FUJITSU Software
Symfoware Server V12.0.0

Operation Guide

Linux

Preface

Purpose of this document

The Symfoware Server database system extends the PostgreSQL features and runs on the Linux platform.

This document is the Symfoware Server Operation Guide.

Intended readers

This document is intended for those who install and operate Symfoware Server.

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

- PostgreSQL

- SQL

- Linux

Structure of this document

This document is structured as follows:

Issue date and version

```
First edition: November 2013
```

Copyright

```
First edition: November 2013
```

Contents

Chapter 1 Operating Symfoware Server

This chapter describes how to operate Symfoware Server.

1.1 Operating Methods

There are two methods of managing Symfoware Server operations:

- Operation management using GUI tools

- Operation management using commands

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 or maintaining the database, you may need to manipulate the database objects defined in the database. For this, use pgAdmin, which is a GUI tool that performs this task easily.

The NCHAR type is available in Symfoware Server pgAdmin.

📑 See
....

- Refer to "Support for National Characters" in the Application Development Guide for information on the NCHAR type.

- Refer to "1.7 Notes on pgAdmin" if using the features below in pgAdmin:

    - Query tool

    - Data view
....

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.
....

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 Windows

The figure below shows the flow of WebAdmin GUI windows.

```
Login
   ├─→ Monitor ─────────→ Create
   │      ├─→ Delete
   │      ├─→ Start/Stop
   │      ├─→ Backup
   │      ├─→ Data Recovery
   │      └─→ Message
   │
   └─→ Setting
          ├─→ Character Code
          ├─→ Client Authentication
          ├─→ Communication
          ├─→ SQL Options
          └─→ Memory
```
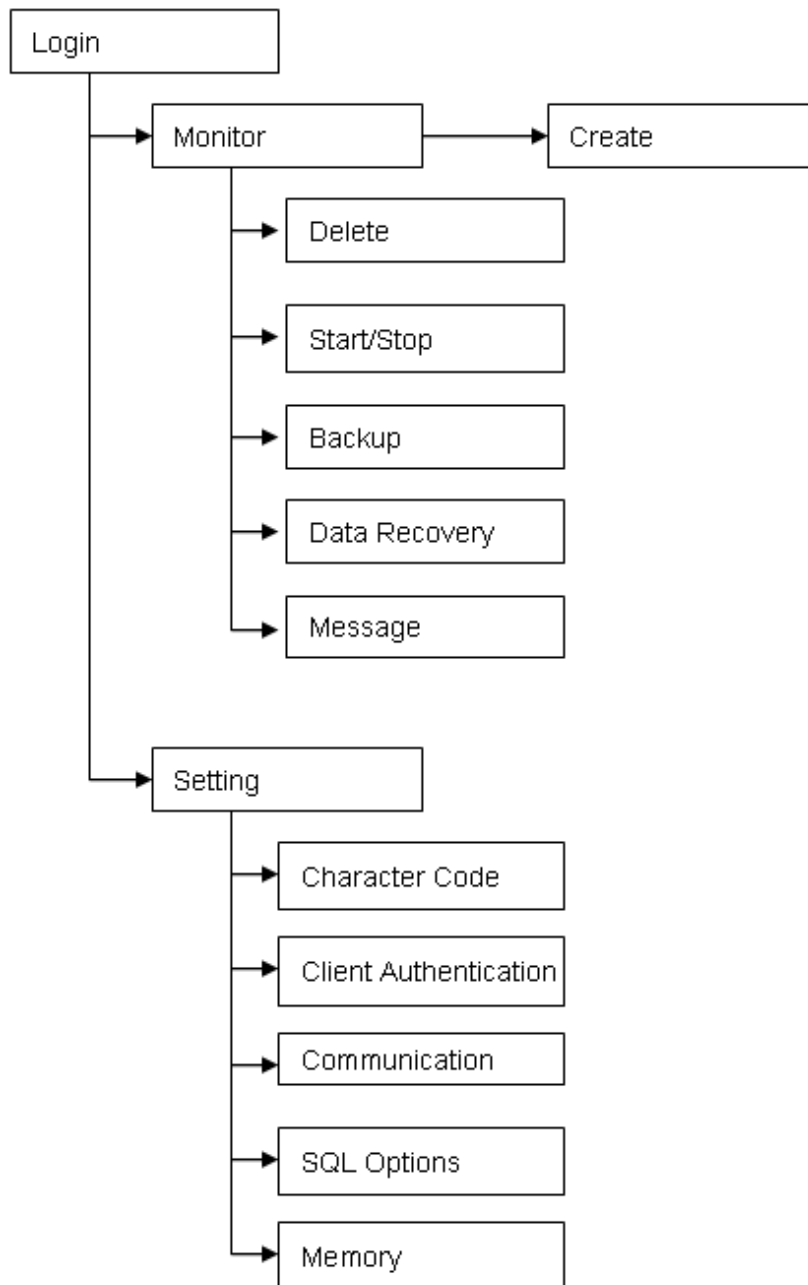
Monitor menu

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

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, 9.0, 10.0

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 Symfoware Server is installed.

- *portNumber*:The port number of WebAdmin. The default port number is 26515.

Example

For a server with IP address "192.0.2.0" and port number "26515"

```
http://192.0.2.0:26515/
```

The activation URL window shown below is displayed.
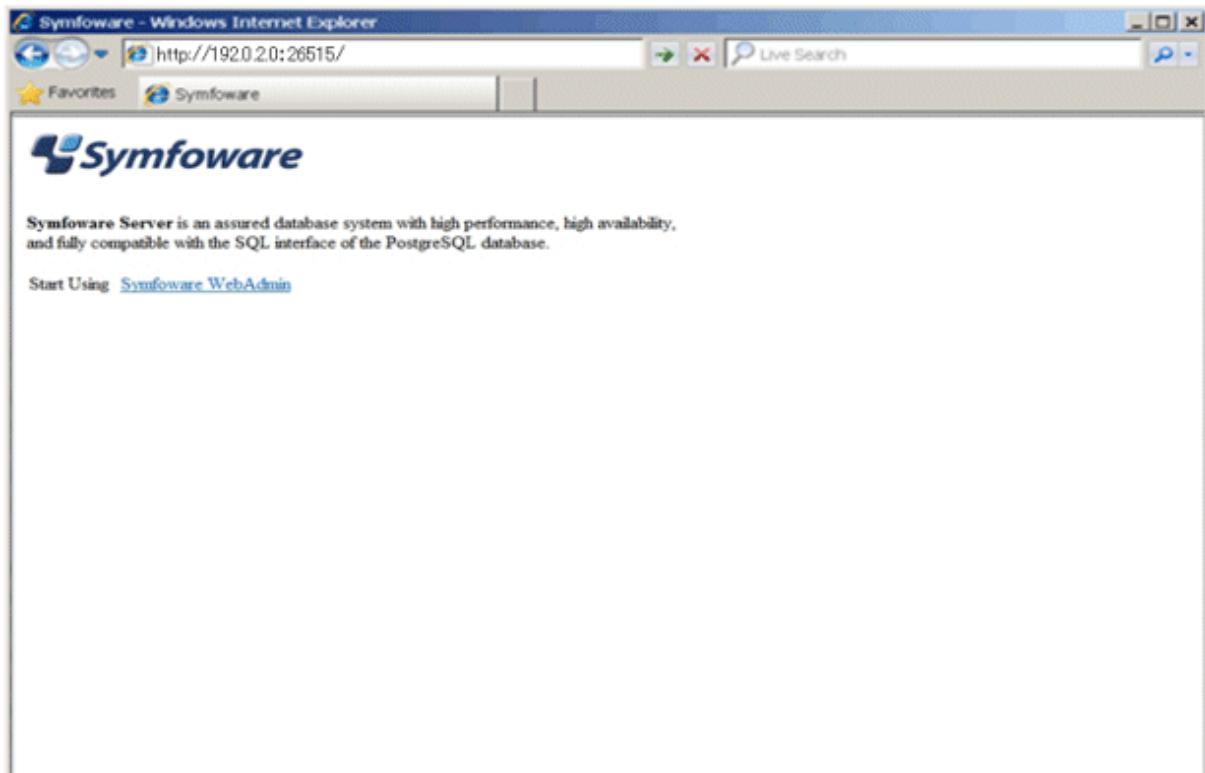
![Note]

- 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 [Symfoware 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

P Point
....

Use the OS user account as the user ID of the instance administrator. Refer to "Creating an Instance Administrator" in the Installtion 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 describes how to start pgAdmin.

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

- 64-bit products

From the [Start] window >> [pgAdmin III(64bit)(V12.0.0)].

- 32-bit products

From the [Start] window >> [pgAdmin III(32bit)(V12.0.0)].

Other operating systems

- 64-bit products

From the [Start] menu, click [All Programs] >> [Symfoware Server Client (Open Interface) (64bit) V12.0.0] >> [pgAdmin III(64bit) (V12.0.0)].

- 32-bit products

From the [Start] menu, click [All Programs] >> [Symfoware Server Client (Open Interface) (32bit) V12.0.0] >> [pgAdmin III(32bit) (V12.0.0)].

The following window is displayed when pgAdmin starts.
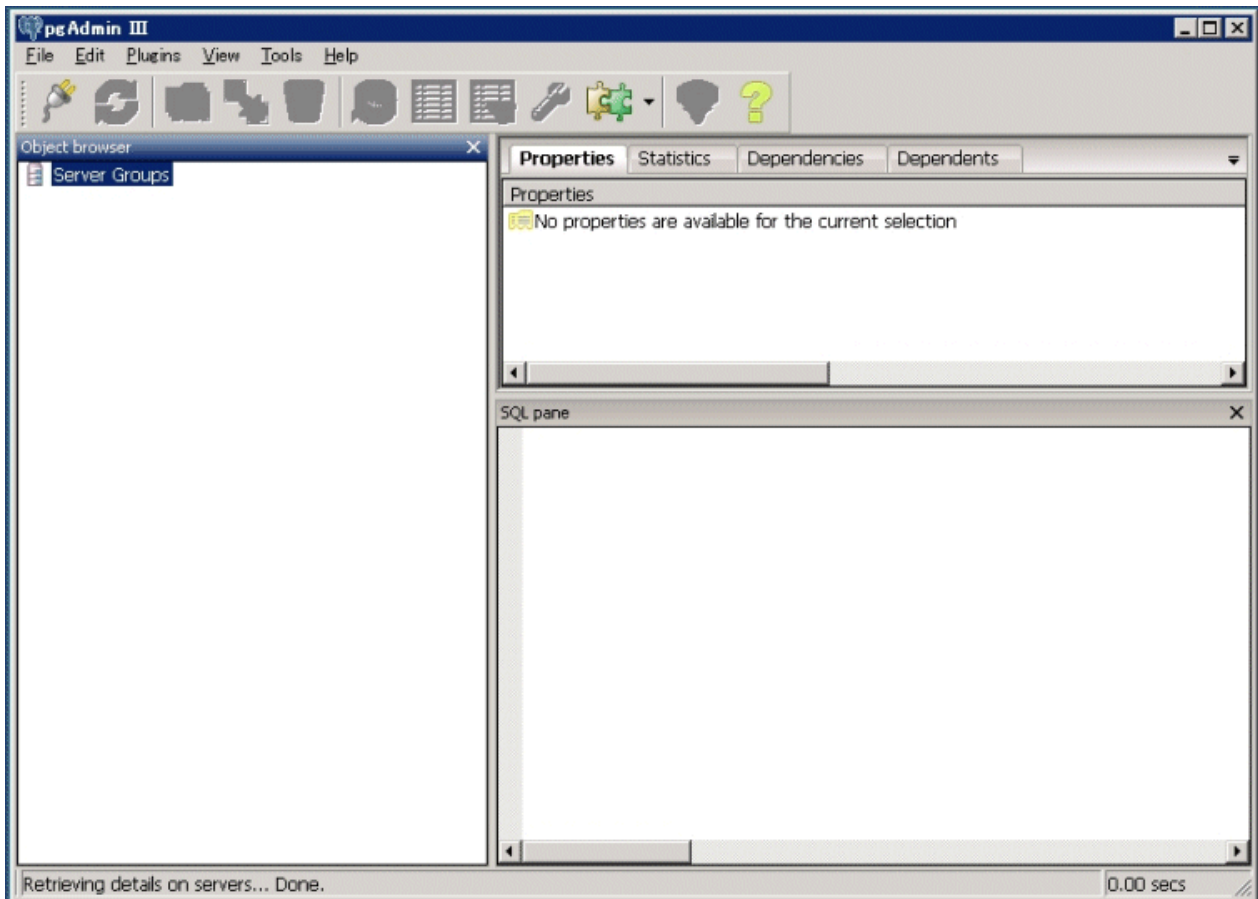
Note

- 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 or later is required for browsing the manual from [Symfoware 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 Symfoware Server is installed

- [Port]: Port number of the instance

- [Username]: User ID of the instance administrator

- [Password]: Password for the user ID specified in [Username]

If you add an instance using pgAdmin, pgAdmin is automatically connected to that instance.

📒 Note
. . . .

If you select [Store password], a file storing the Symfoware Server 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.con
. . . .

1.3.3 Connecting/Disconnecting an Instance

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

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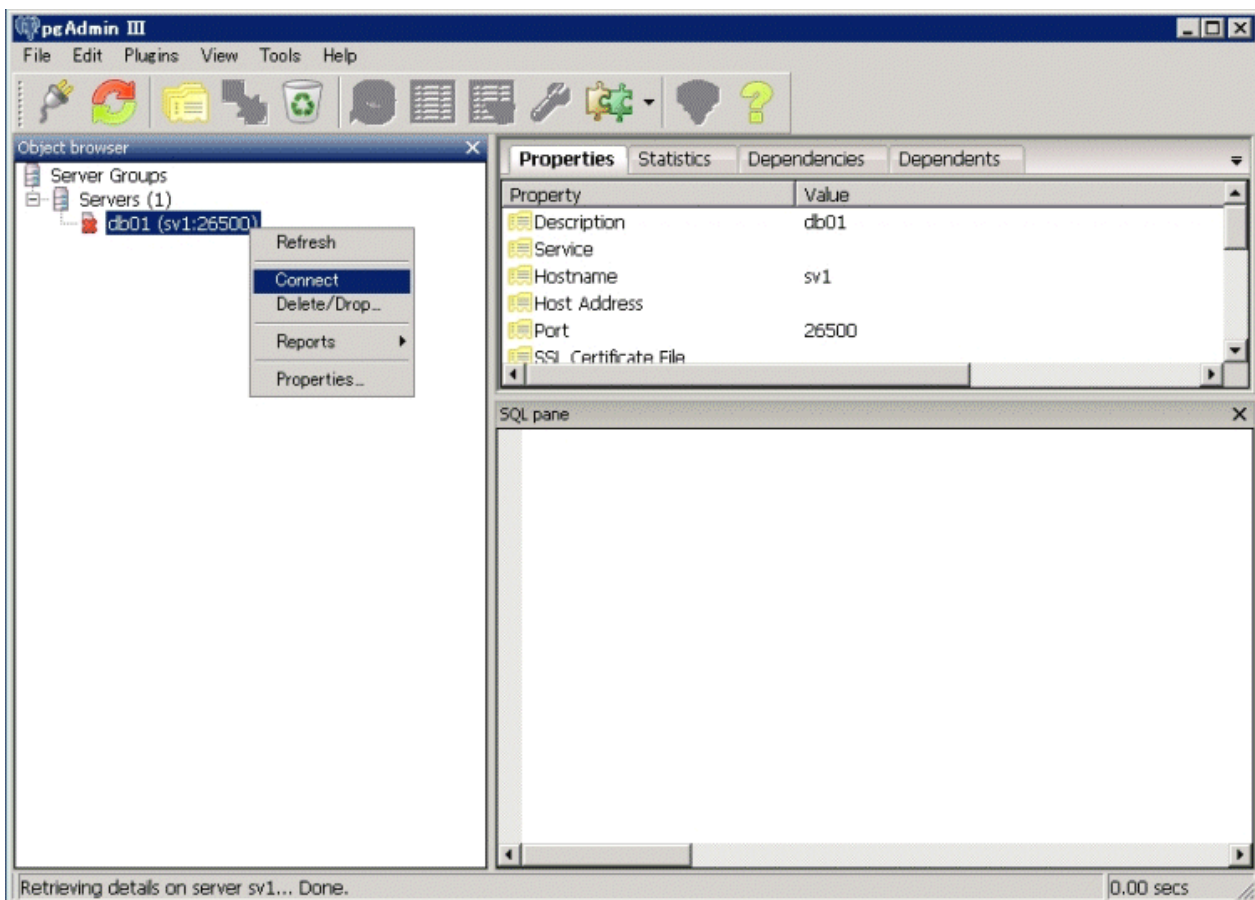
If a password was not saved when the instance was added, the following password entry window is displayed.

Connect to Server

Please enter password for user symfo
on server db01 (sv1)

☐ Store password

Help   OK   Cancel

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. You can run these commands on a server that can connect to the database.

🛢 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 Symfoware Server

This section describes the operating environment and the file composition of Symfoware Server.

1.5.1 Operating Environment

The following figure shows the configuration of the Symfoware Server operating environment. The tables given below list the roles of the OS resources and Symfoware Server resources.
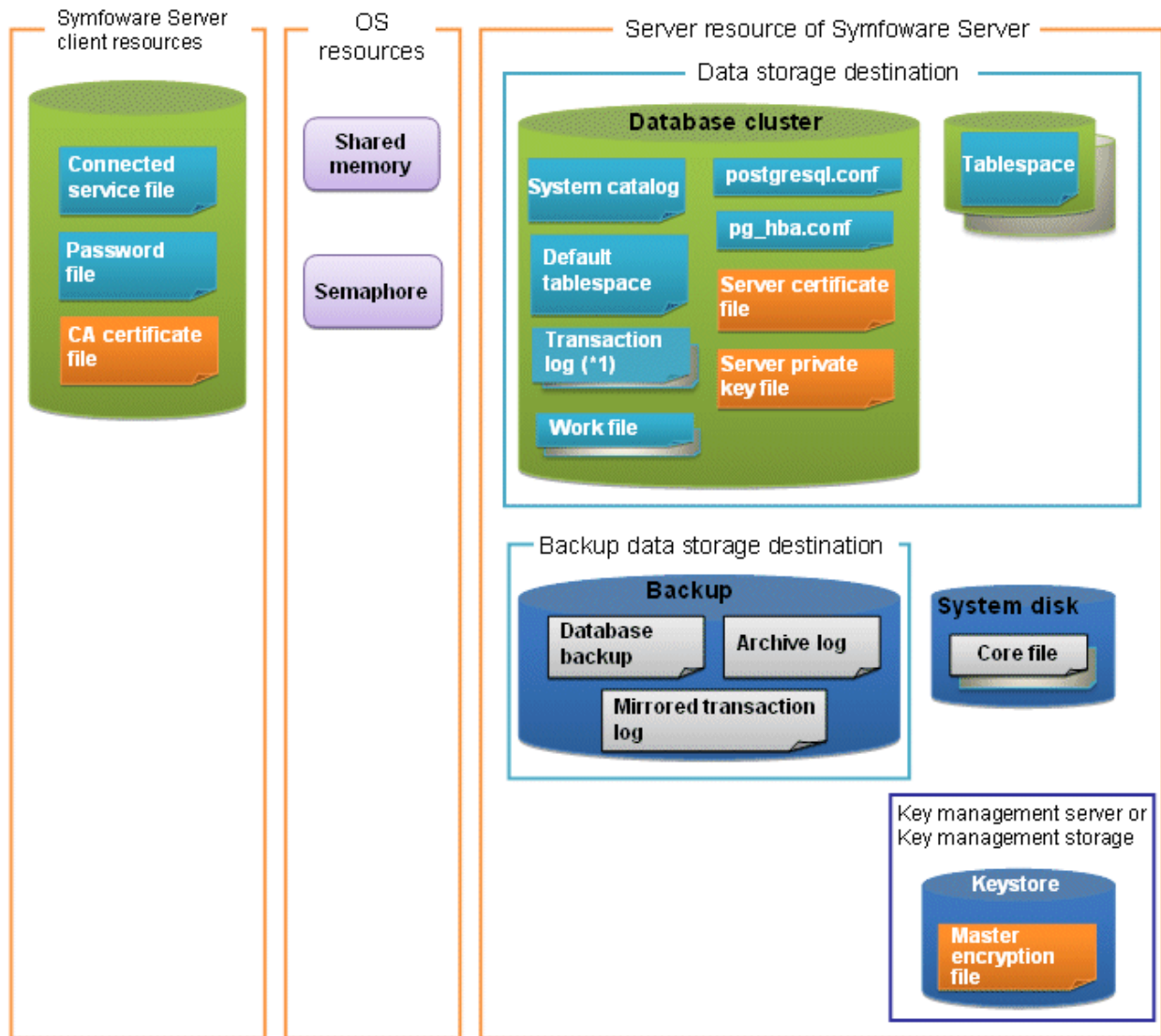
*1: To distribute the I/O load, place the transaction log on a different disk from the data storage destination.

Table 1.1 OS resources

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

Table 1.2 Symfoware Server client resources

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

Table 1.3 Server resources of Symfoware Server

Type	Role
Database cluster	Database storage area on the database storage disk. It is a collection of databases managed by an instance.

Type	Role
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 Symfoware Server
pg_hba.conf	Symfoware Server 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	Symfoware Server process core file that is output when an error occurs during a Symfoware Server 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

Symfoware Server 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/symfoserver64 **32-bit product** /opt/symfoserver32
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.
Server certificate file (*1)	N	1	Specify using postgresql.conf.

File type	Required	Quantity	How to specify the location
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 Symfoware Server 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 Symfoware Server 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 Symfoware Server are scanned for viruses. Alternatively, if the files that comprise Symfoware Server are to be scanned for viruses, stop Symfoware Server and perform the scan when tasks that use Symfoware Server are not operating.

. . . .

1.6 Notes on Compatibility of Applications Used for Operations

When you upgrade Symfoware Server 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 Symfoware Server.

📚 See

. . . .

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

. . . .

1.7 Notes on pgAdmin

When using the following pgAdmin features, the data may not be displayed if the column in the referenced data is too long - if this happens, export the data to a file before checking it.

- Query tool

- Data view

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

P 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.
. . . .

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.

Typically, the status of an active instance appears as Started. If an error is detected, the status of the instance appears as Stopped and a message is output to the system log.

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
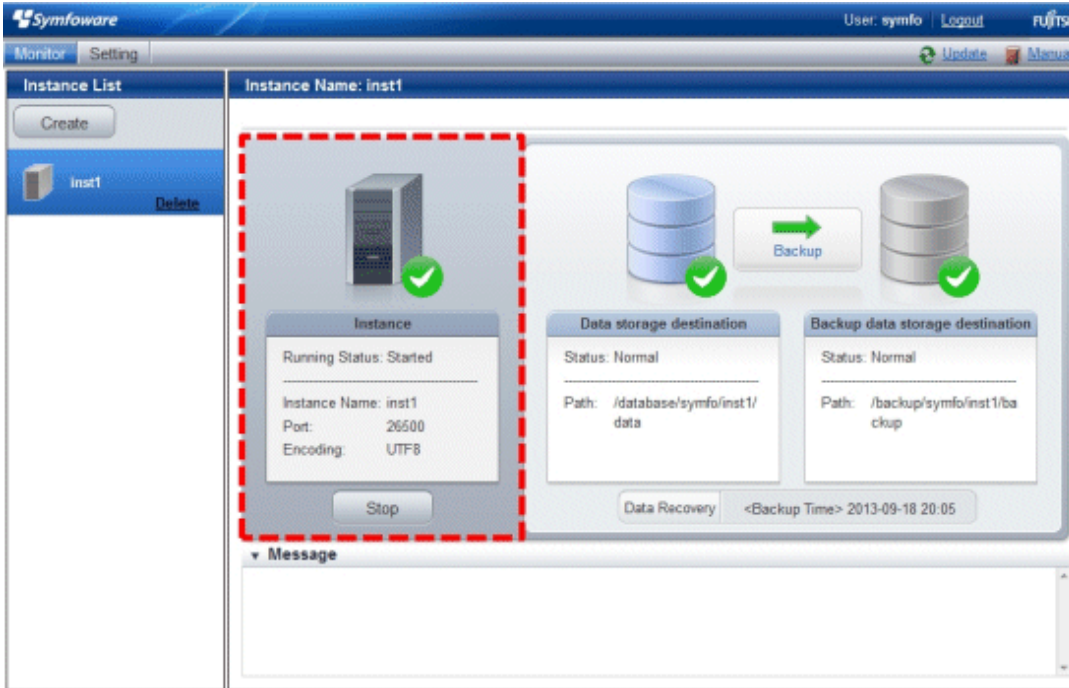
Figure 2.2 Status when an instance is stopped
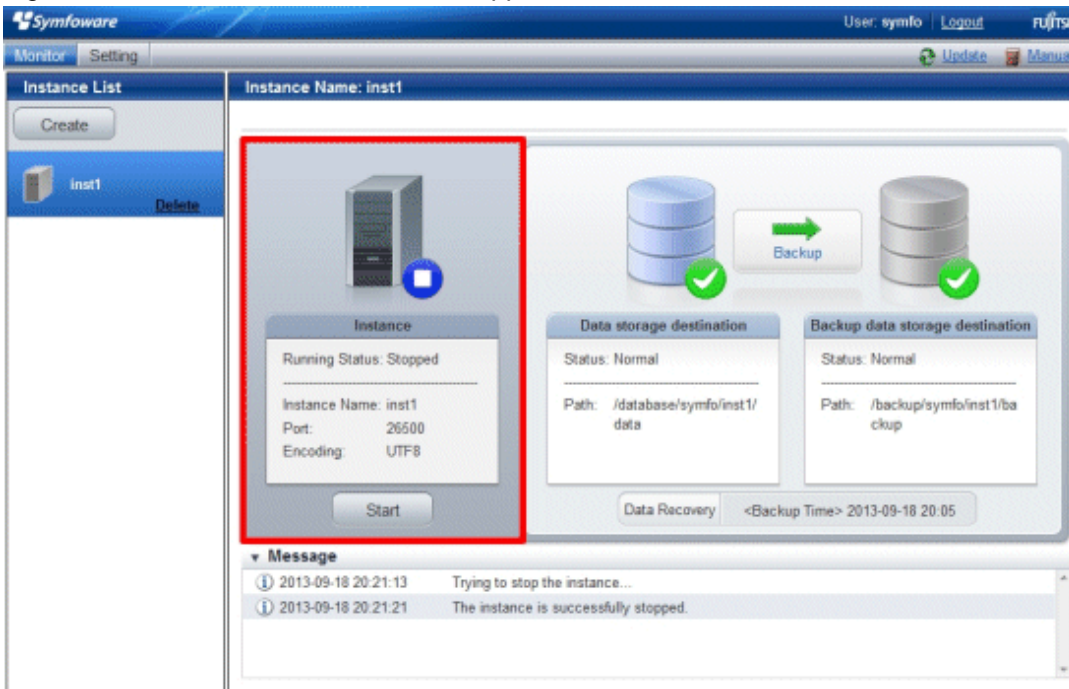
Note

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.

Starting an instance

Use the pg_ctl command to start an instance.

Configure the environment variables before using the commands.

See

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.

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.

- Preferably, specify the -w option. This option ensures that instance startup completes successfully before the command is returned. If you omit the -w option, it may be impossible to tell if the instance started successfully.

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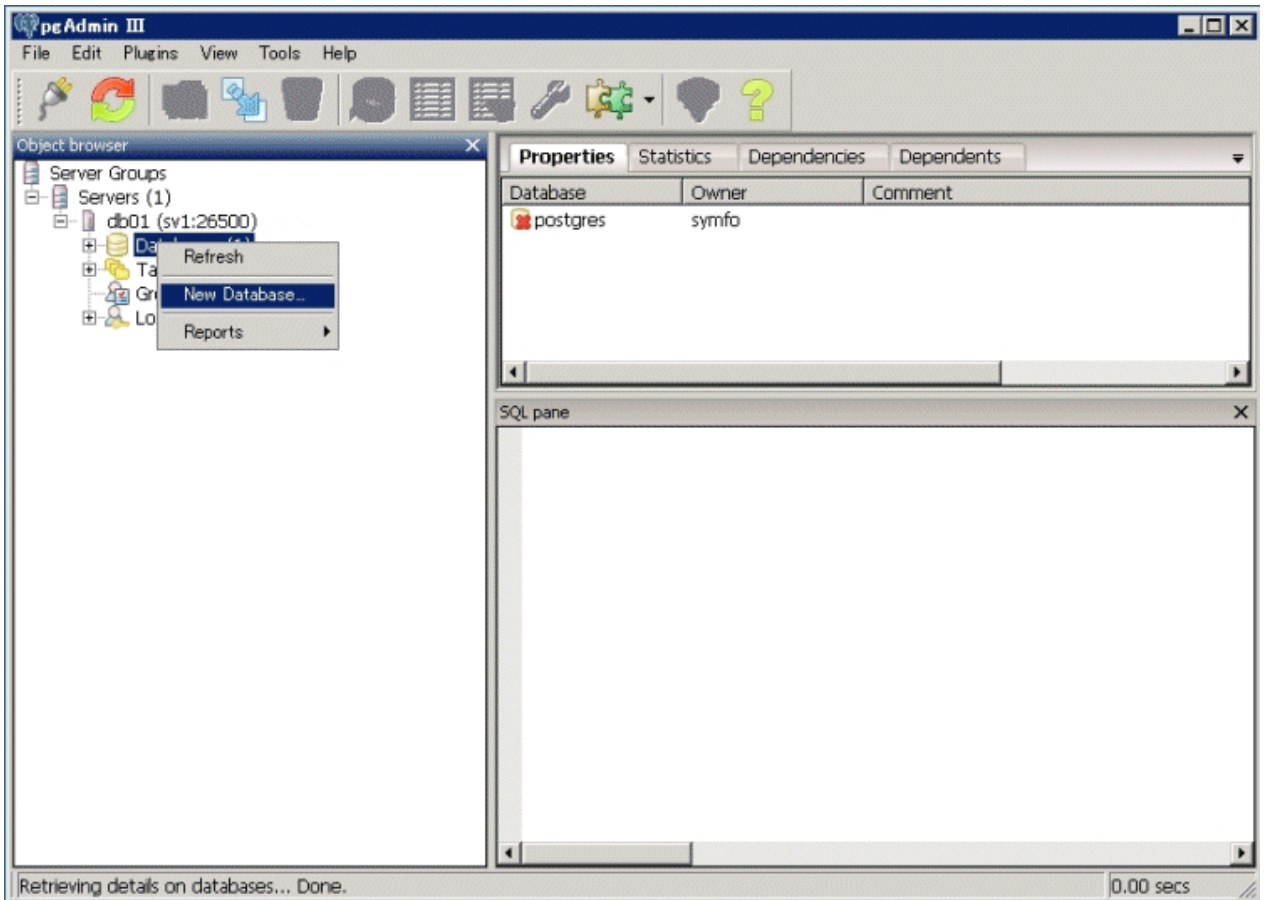
2. Specify appropriate values for the following items in the new database window.

   - [Properties] tab

     The following example illustrates creation of the database "db01".
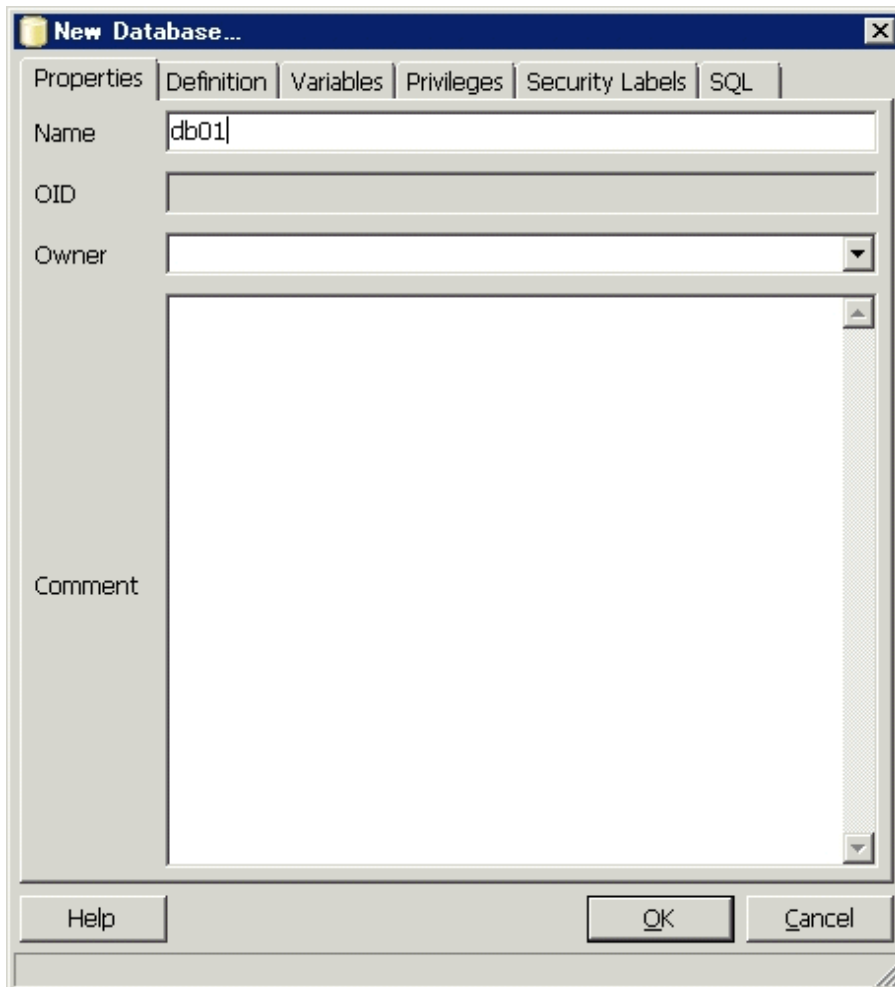
     - [Name]: Name of the database to be managed

3. Click [OK] to create the database.

2.2.2 Using Client Commands

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

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

```
> psql postgres
psql (9.2.4)
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 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 Symfoware Server 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.

Backup operation

Follow the procedure below to back up the database.

1. Select database backup

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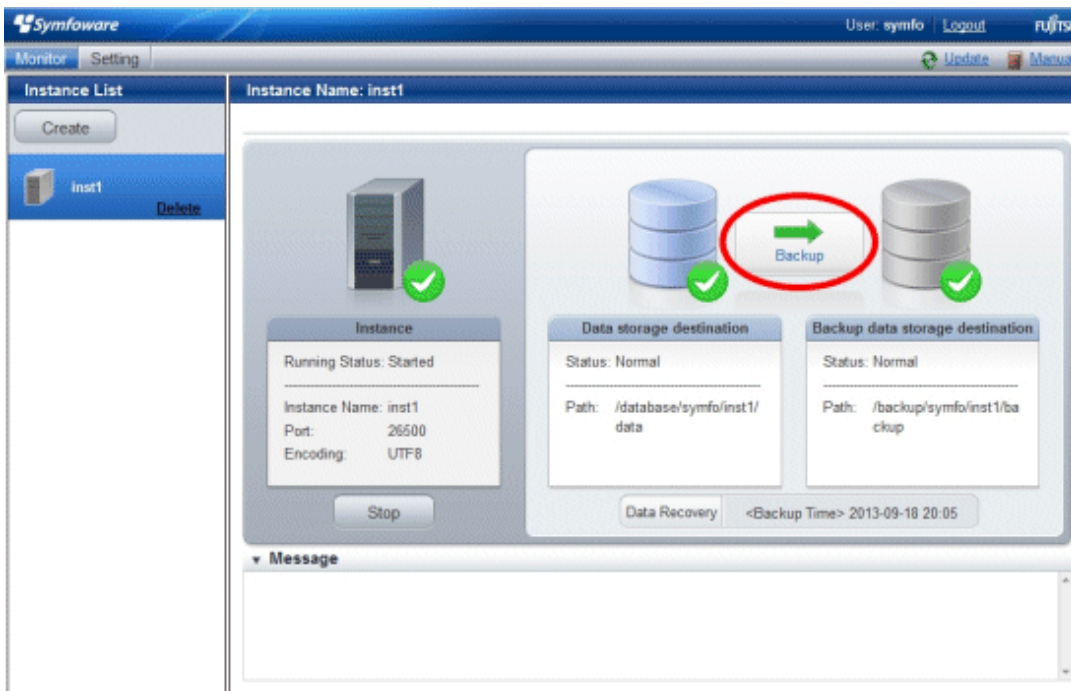
2. Back up the database

   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.
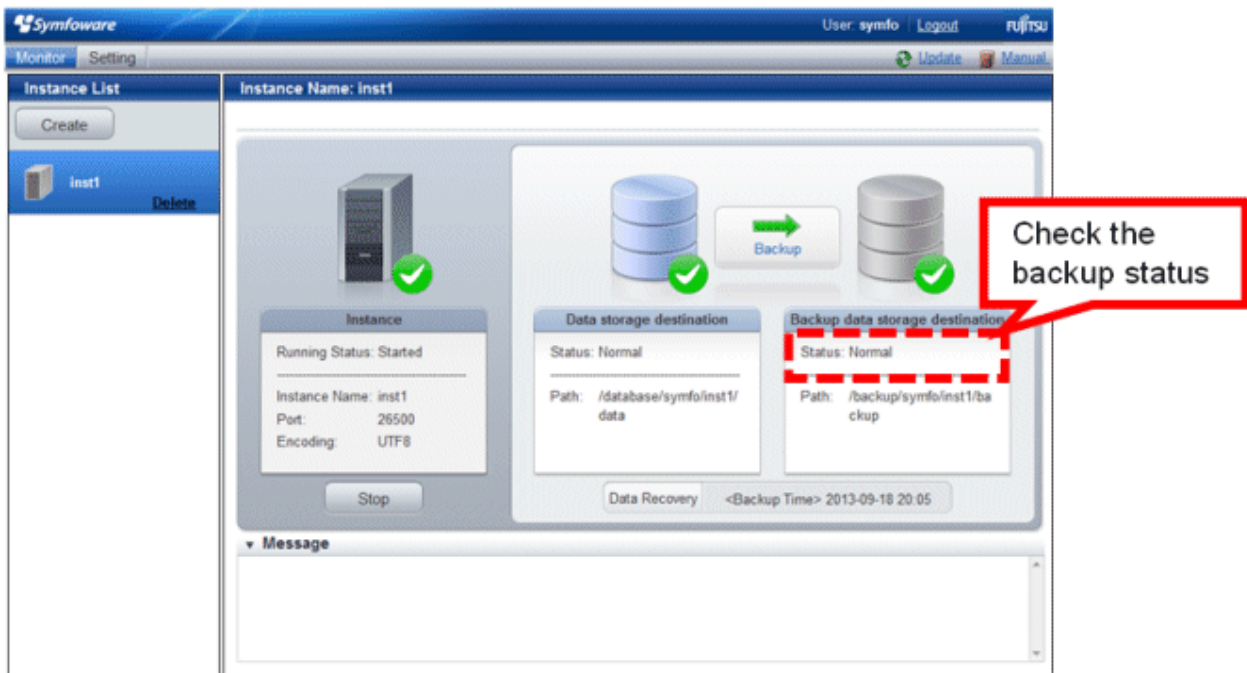
Note

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 symfo:symfo /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	Specify the output level for the transaction log.
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.

Setting a restore point

You can create a named point in the database for use when restoring data.

Use the psql command to set a point.

Set a restore point before starting operations if you want to recover the database to a point other than when a backup was taken, or the latest point preceding an error. At recovery, specify this restore point as the recovery end point. It is useful to set a restore point, for example, before you execute an application.

🔷 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_20130703_1');
LOG:  restore point "batch_20130703_1" created at 0/20000E8
STATEMENT:  select pg_create_restore_point('batch_20130703_1');
 pg_create_restore_point
------------------------
 0/20000E8
(1 row)
```

. . . .

Refer to "7.3.2 Using the pgx_rcvall Command" for information on using a restore point to recover the database.

- 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.

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

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.

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

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
2013-07-01 13:30:40    COMPLETE       /backup/inst1/2013-07-01_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.

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
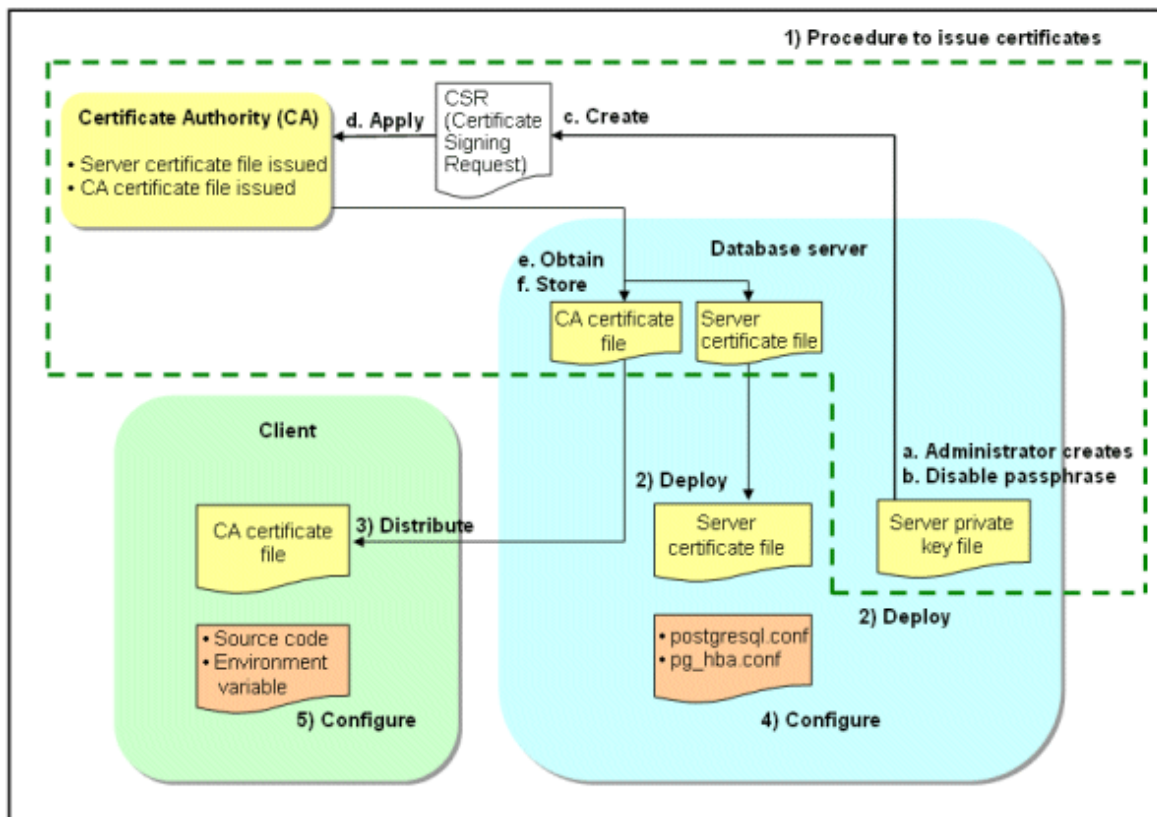
4.1.1 Issuing a Certificate

For authenticating servers, you must acquire a certificate issued by the certificate authority (CA).
Symfoware Server 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"
. . . .

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 Symfoware Server, 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. Symfoware Server 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 trade off 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.

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.

3. Use the pg_ctl command to restart an instance.

   Specify the following in the pg_ctl command:

   - Specify "restart" as the mode.

   - 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.

   - It is recommended to specify the -w option. This means that the command returns after waiting for the instance to start. If the -w 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
```

....

📌 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 SQL 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.

If recovery is to be performed during instance startup, the instance cannot be started using the above method. This is because the encrypted WAL must be decrypt ed. In this case, specify the --keystore-passphrase option in the pg_ctl command instead of executing the pgx_open_keystore function. This will display the prompt for the passphrase to be entered.

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

P 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';
```

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.

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.

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;
```

See

....

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

....

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)
```

....

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 symfo:symfo /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 backup 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 July 1, 2013.

```
> pgx_dmpall -D /database/inst1
> cp -p /key/store/location/keystore.ks /keybackup/keystore_20130701.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 July 5, 2013.

```
> psql -c "SELECT pgx_set_master_key('passphrase')" postgres
> cp -p /key/store/location/keystore.ks /keybackup/keystore_20130705.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 July 5, 2013.

```
> cp -p /keybackup/keystore_20130705.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 July 2013.

```
> cp -p /keybackup/keystore_20130701.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

Symfoware Server 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 July 1, 2013.

```
> pgx_dmpall -D /database/inst1
> cp -p /key/store/location/keystore.ks /keybackup/keystore_20130701.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 July 1, 2013.

```
> cp -p /keybackup/keystore_20130701.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 and systems that use streaming replication.

5.10.1 HA Clusters that do not Use Streaming Replication

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

Location of the keystore file

The primary and standby servers can use the same keystore file on the key management server or the key management storage. This is because the standby server is inactive while the primary server is active, so both servers cannot simultaneously read or write to the same file.

A copy of the keystore file can also be placed on both servers. In this case, you must copy the keystore file to the standby server if you change the master encryption key and the passphrase on the primary server.

Using an automatically opening keystore on the standby server

When using an automatically opening keystore, enable automatic opening on the standby server. An automatically opening keystore (keystore.aks) created on the primary server cannot automatically open even if you copy it to the standby server.

5.10.2 Streaming Replication

Take the following points into account when using transparent data encryption in a streaming replication environment.

Location of the keystore file

Use different keystore files on the primary server and the standby server. Both servers cannot share the same keystore file on the file server. This is because both servers may read or write to the file simultaneously. The same applies to automatically opening keystores.

Using an automatically opening keystore on the standby server

When using an automatically opening keystore, enable automatic opening on the standby server. An automatically opening keystore (keystore.aks) created on the primary server cannot automatically open even if you copy it to the standby server.

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.

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

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

Symfoware Server 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 Symfoware Server to collect logs from its log files (for example, log_destination) before operating Symfoware Server.

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.
....

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 10 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

Symfoware Server does not immediately delete the updated or deleted data. If the VACUUM determines there are no transactions that reference the database, Symfoware Server 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            | 16384
datname          | db01
pid              | 16875
usesysid         | 10
usename          | symfo
application_name | apl01
client_addr      | 192.33.44.15
client_hostname  |
client_port      | 51793
backend_start    | 2013-05-31 17:40:24.161826+09
xact_start       | 2013-05-31 17:40:27.636134+09
query_start      | 2013-05-31 17:40:27.636134+09
state_change     | 2013-05-31 17:40:27.636402+09
waiting          | f
state            | idle in transaction
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].
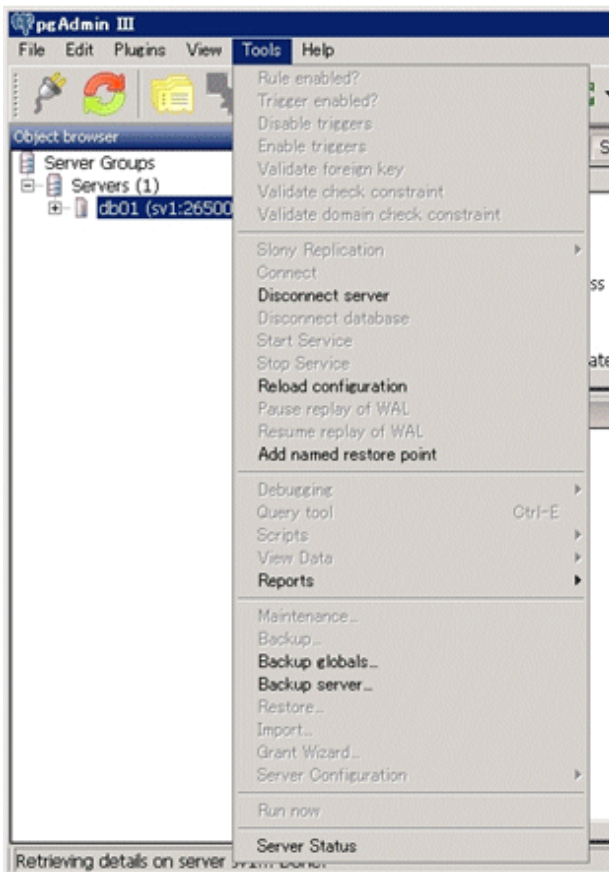
2. Identify client connections that have been in the waiting state for an extended period.

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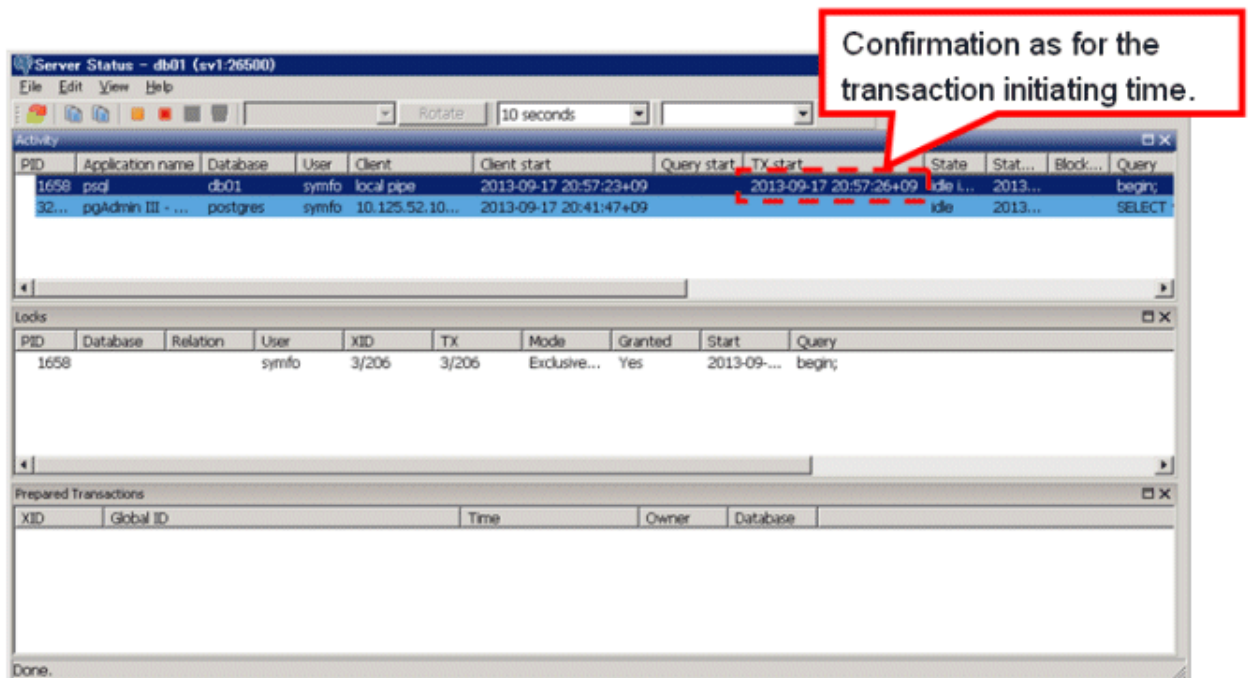
6.5 Reorganizing Indexes

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.

Chapter 7 Actions when an Error Occurs

This chapter describes the actions to take when an error occurs in the database or an application, while Symfoware Server 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

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

7.1 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:

- 7.1.1 Using WebAdmin

- 7.1.2 Using Server Command

· ·

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.

· ·

7.1.1 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.
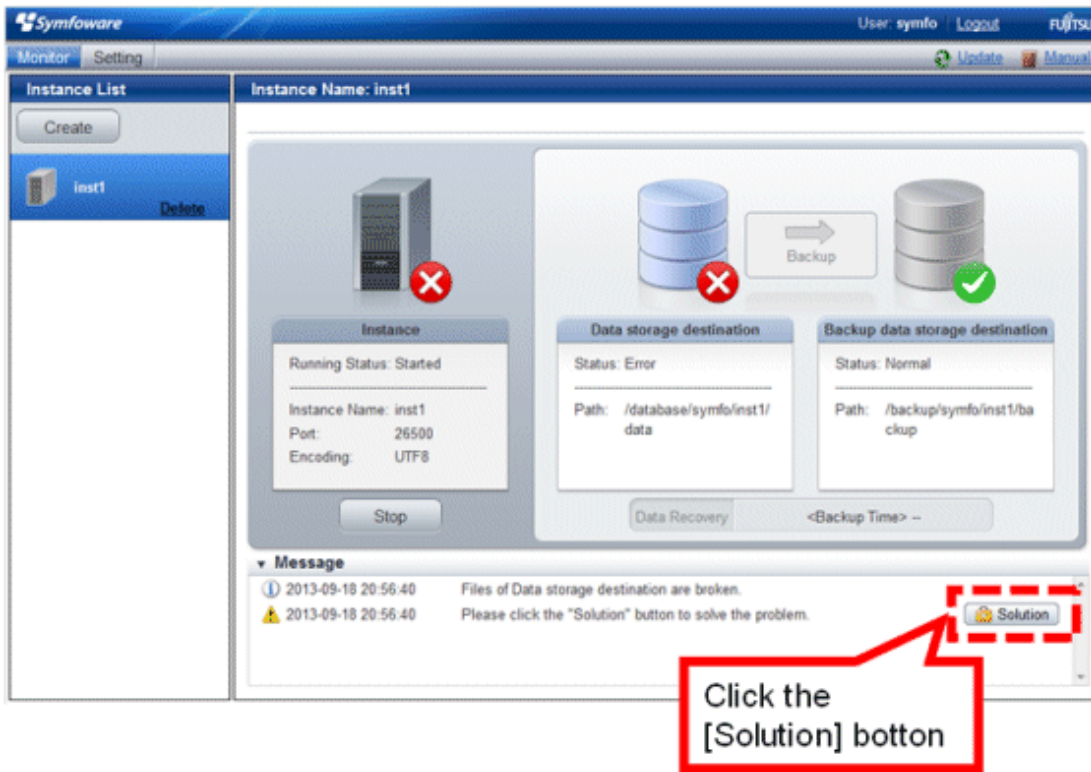
7. Run recovery

   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

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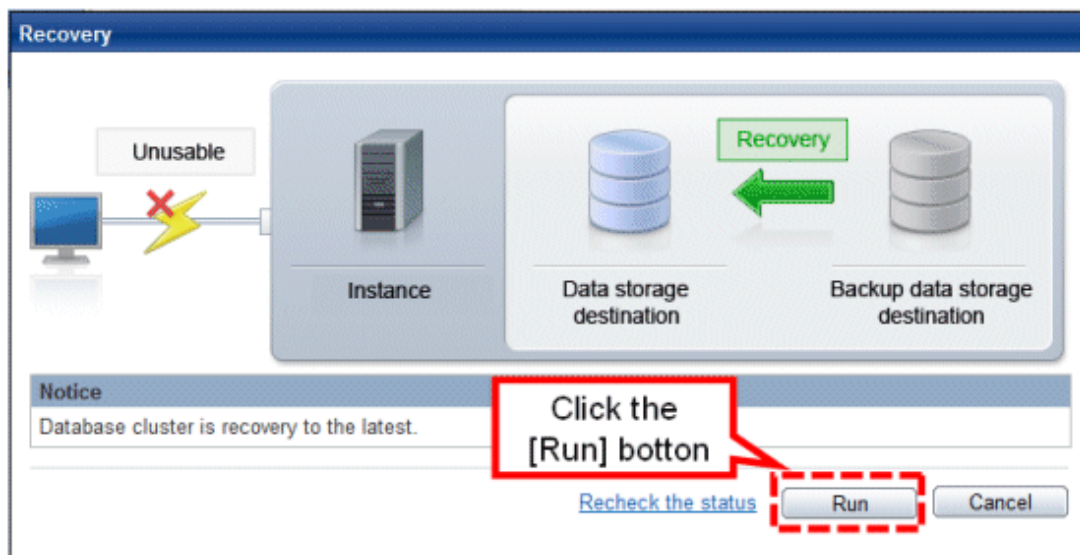
WebAdmin cannot accurately recover a hash index. If you are using a hash index, then after recovery, execute the REINDEX command for the appropriate index.

. . . .

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 "7.10.3 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

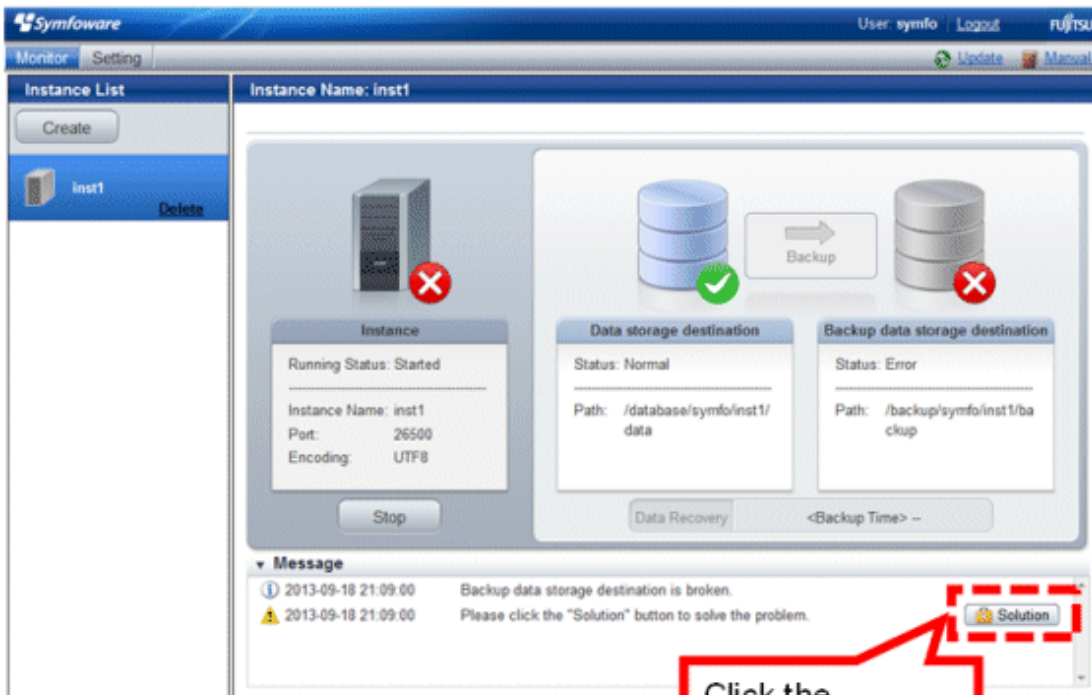   Log in to WebAdmin, and in the [Monitor] window, click the [Solution] button for the error message.

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.

![P] 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.

7.1.2 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 "7.11 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

Example

To create a data storage destination directory:

```
$ mkdir /database/inst1
$ chown symfo:symfo /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 (therefore the user does not need not be concerned).

```
FATAL: The database system is starting (XXXXX)
```

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 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 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 symfo:symfo /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 "7.11 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 symfo:symfo /backup/inst1
# chmod 700 /backup/inst1
```

Refer to "3.2.2 Using Server Commands" for detailes.

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_mode.

Refer to "B.1 WAL Mirroring Control Functions" for information on pgx_resume_wal_multiplexing.

....

7.2 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:

- 7.2.1 Using WebAdmin

- 7.2.2 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.

....

7.2.1 Using WebAdmin

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

Refer to "7.1.1 Using WebAdmin" for details.

7.2.2 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 "7.11 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.

   - pecify 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
   2013-07-20 10:00:00    COMPLETE         /backup/inst1/2013-07-20_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, "July 20, 2013 10:00:00" is specified as the recovery time.

   ```
   > pgx_rcvall -D /database/inst1 -B /backup/inst1 -e '2013-07-20 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 (therefore the user does not need not be concerned).

```
FATAL: The database system is starting (XXXXX)
```

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.

7.3 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:

- 7.3.1 Using WebAdmin

- 7.3.2 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.

7.3.1 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].
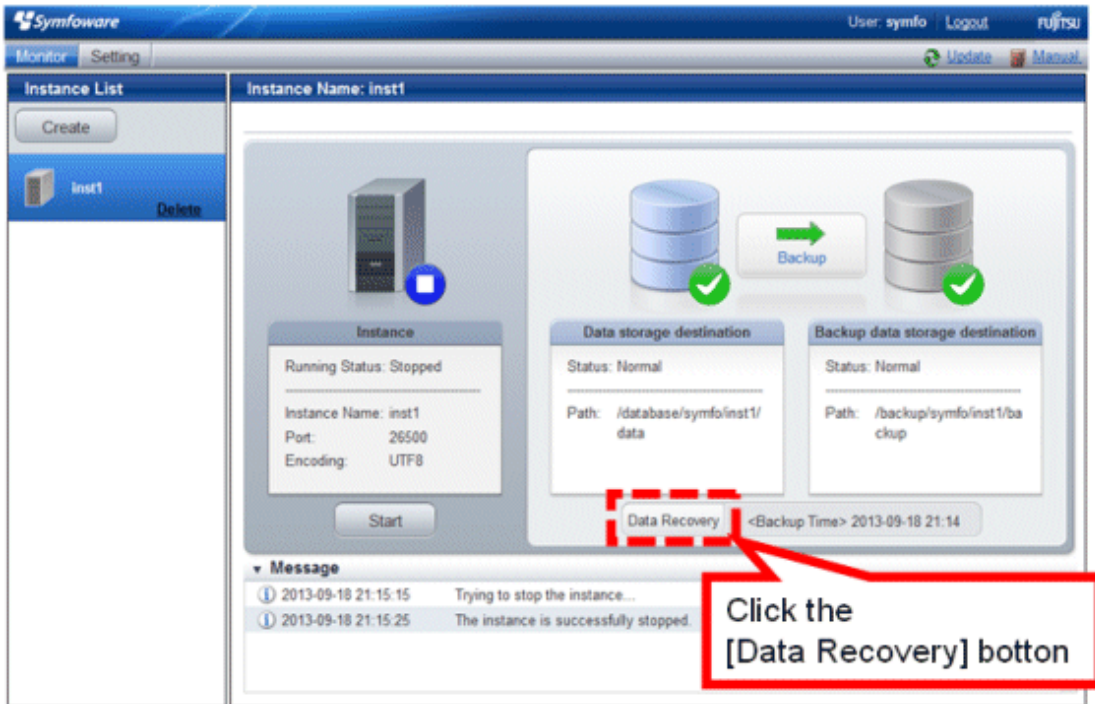
5. Recover to the backup point

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**

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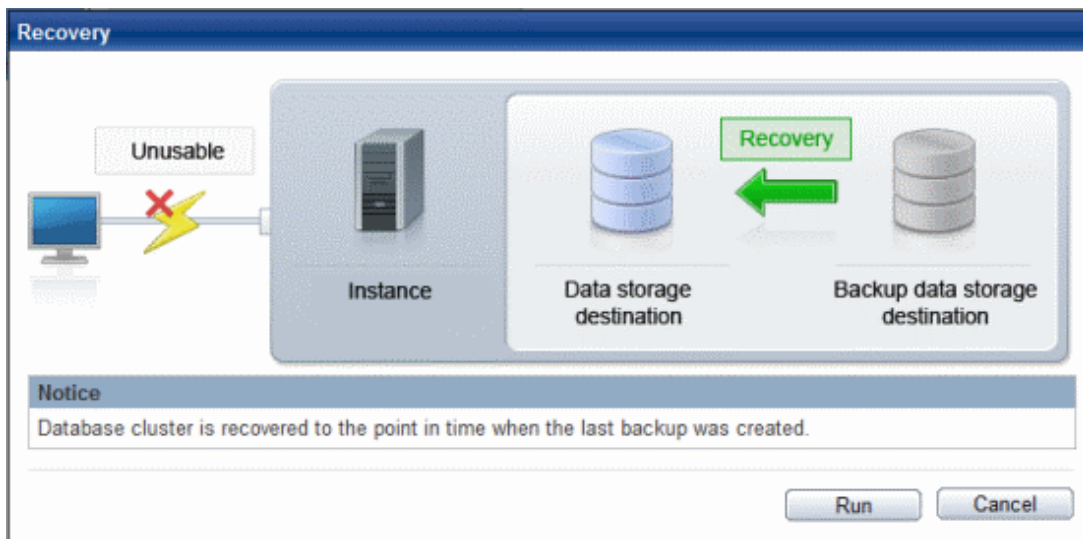
WebAdmin cannot accurately recover a hash index. If you are using a hash index, then after recovery, execute the REINDEX command for the appropriate index.

....

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.

7.3.2 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 "7.11 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 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
    2013-07-01 10:00:00   COMPLETE        /backup/inst1/2013-07-01_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_20130703_1".

    ```
    > pgx_rcvall -D /database/inst1 -B /backup/inst1 -n batch_20130703_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 (XXXXX)
```

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.

7.4 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 "7.4.1 When using the view (pg_stat_activity)")

- ps command (refer to "7.4.2 Using the ps Command")

- pgAdmin (refer to "7.4.3 Using pgAdmin")

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

7.4.1 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 (9.2.4)
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             | symfo
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.

....

7.4.2 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
symfo    19174 18027 pts/1                  \_ grep postgres
symfo    20517     1 ?          /opt/symfoserver64/bin/postgres -D /disk01/data
symfo    20518 20517 ?           \_ postgres: logger process
symfo    20520 20517 ?           \_ postgres: checkpointer process
symfo    20521 20517 ?           \_ postgres: writer process
symfo    20522 20517 ?           \_ postgres: wal writer process
symfo    20523 20517 ?           \_ postgres: autovacuum launcher process
symfo    20524 20517 ?           \_ postgres: archiver process
symfo    20525 20517 ?           \_ postgres: stats collector process
symfo    18673 20517 ?           \_ postgres: symfo postgres 192.168.100.1(49448) idle
symfo    16643 20517 ?           \_ postgres: symfo db01 192.168.100.11(49449) UPDATE waiting
symfo    16644 20517 ?           \_ postgres: symfo 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
-------------------
```
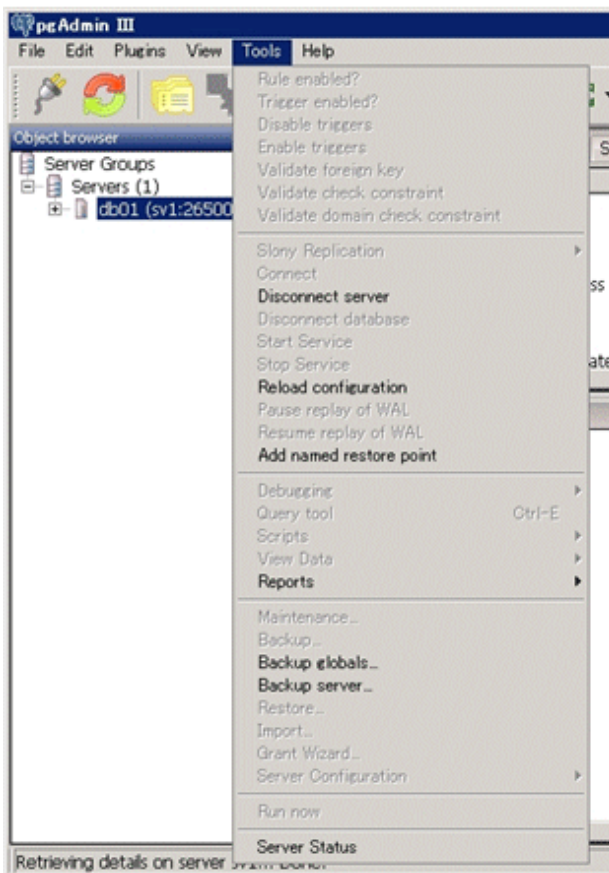
```
t
(1 row)
```

7.4.3 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.

The connection is terminate.
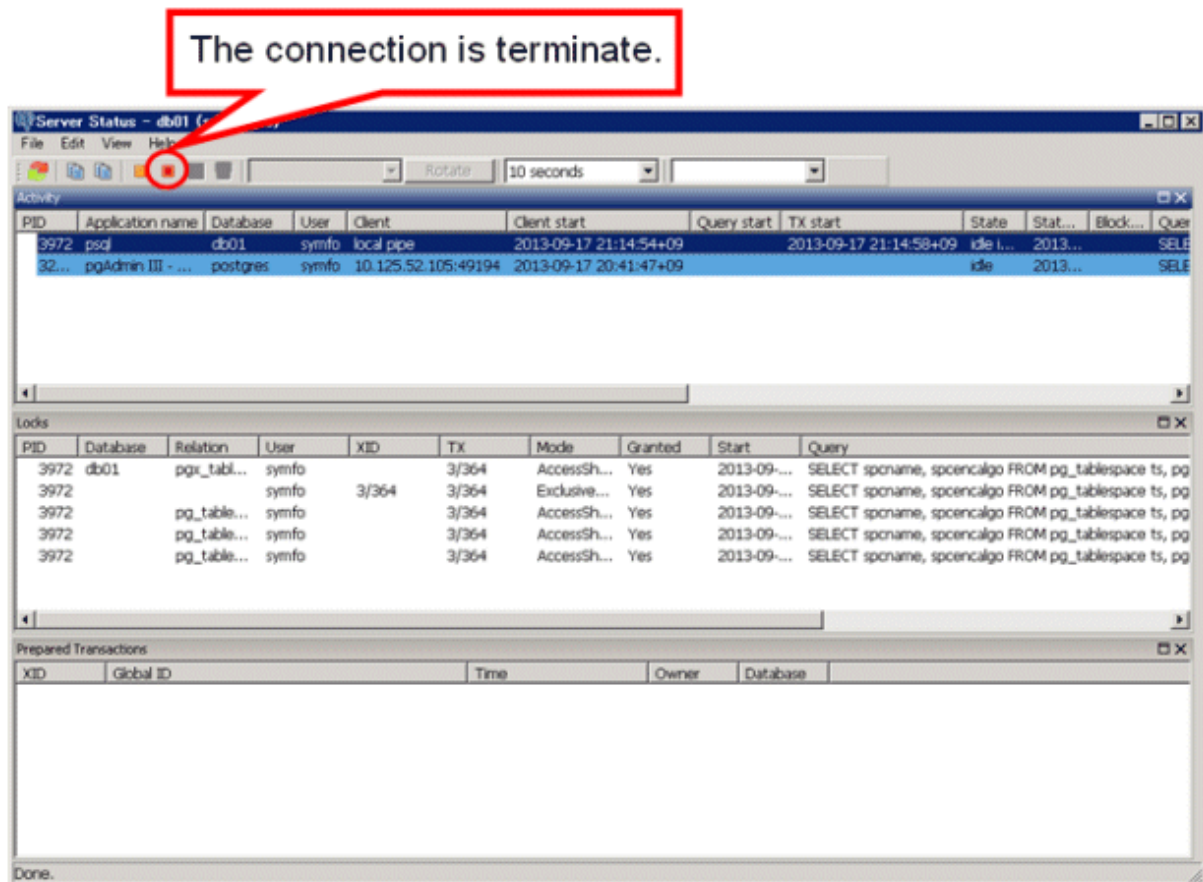
7.5 Actions in Response to an Access Error

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.

. . . .

7.6 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:

- 7.6.1 Using a Tablespace

- 7.6.2 Replacing the Disk with a Larger Capacity Disk

7.6.1 Using a Tablespace

Symfoware Server 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.
. . . .

7.6.2 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:

- 7.6.2.1 Using WebAdmin

- 7.6.2.2 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.
. . . .

7.6.2.1 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 "7.1.1 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.

7.6.2.2 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 "7.11 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 symfo:symfo /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 (XXXXX)
```

📛 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.

7.7 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:

- 7.7.1 Temporarily Saving Backup Data

- 7.7.2 Replacing the Disk with a Larger Capacity Disk

7.7.1 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:

- 7.7.1.1 Using WebAdmin

- 7.7.1.2 Using Server Commands

7.7.1.1 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.
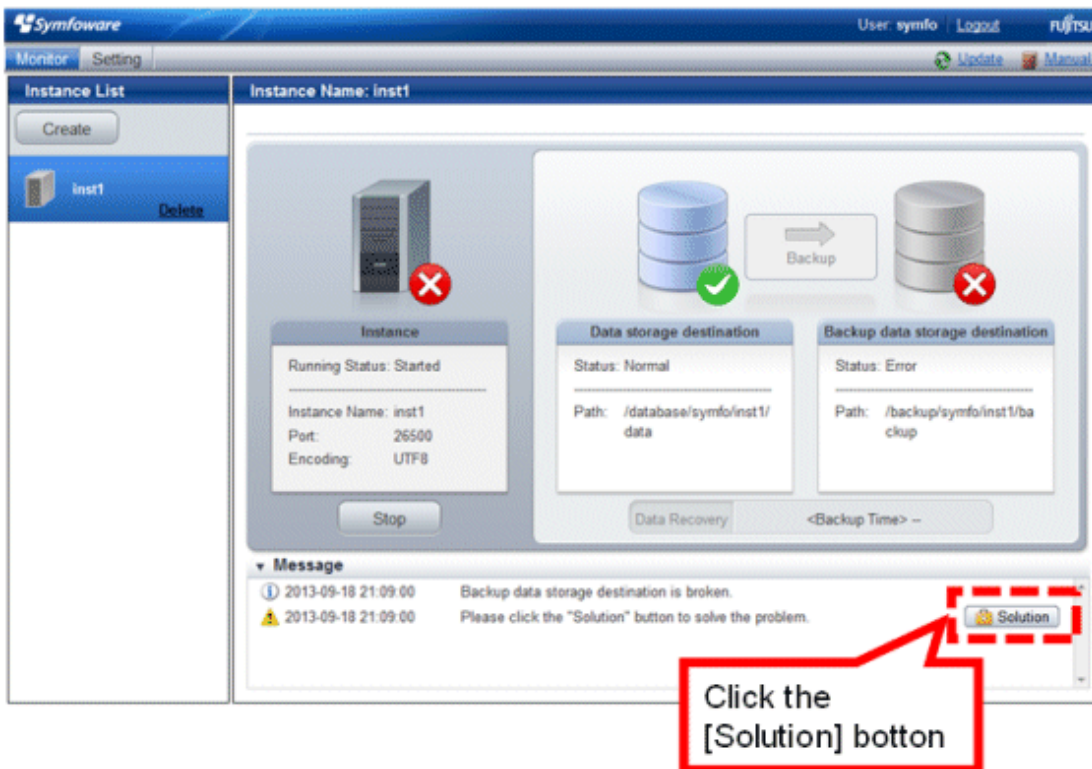
   The above screen is displayed if moving and saving backup data causes WebAdmin to detect that the content at the backup data storage destination is missing. If you use another method to save the data and no abnormality is detected, click [->] next to the "Backup" caption in the [Monitor] menu window.

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.
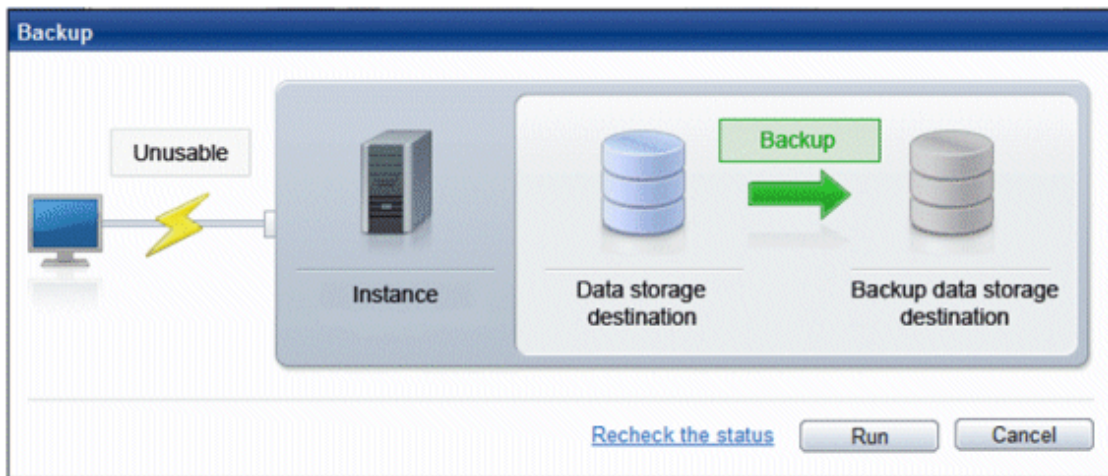

4. Delete temporarily saved backup data

   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
   ```

7.7.1.2 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 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.

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 "7.11 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_mode.

Refer to "B.1 WAL Mirroring Control Functions" for information on pgx_resume_wal_multiplexing.

7.7.2 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:

- 7.7.2.1 Using WebAdmin

- 7.7.2.2 Using Server Commands

Note

Before replacing the disk, stop applications that are using the database.

7.7.2.1 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 backup data") and 3 ("Run backup") under "If failure occurred on the backup storage disk" in "7.1.1 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
   ```

7.7.2.2 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 symfo:symfo /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 "7.11 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.

```
# mkdir /backup/inst1
# chown symfo:symfo /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 "pg_ctl" under "Reference" in the PostgreSQL Documentation for information on the pg_ctl command.

Refer to "Write Ahead Log" under "Server Administration" in the PostgreSQL Documentation for information on archive_mode.

Refer to "B.1 WAL Mirroring Control Functions" for information on pgx_is_wal_multiplexing_paused and pgx_resume_wal_multiplexing.

7.8 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.

7.8.1 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:

- 7.8.1.1 Using WebAdmin

- 7.8.1.2 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 icon] **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.

. . . .

7.8.1.1 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 "7.1.1 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.

7.8.1.2 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 "7.11 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 symfo:symfo /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 (therefore the user does not need not be concerned).

```
FATAL: The database system is starting (XXXXX)
```

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.

7.9 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 destination disk	Transaction log storage destination disk	Backup data storage destination disk
Error	-	Error
-	Error	Error

7.10 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.

7.10.1 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

7.10.2 Unmounted Disk

If the server restart, for example, results in any storage disk not being mounted, instance startup fails. Refer to the system log and the server log, check if it contains any message relating to the storage destinations.

7.10.3 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:

- 7.10.3.1 Using WebAdmin

- 7.10.3.2 Using Server Commands

7.10.3.1 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 "7.2.1 Using WebAdmin" for details.

7.10.3.2 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 "7.2.2 Using the pgx_rcvall Command" for details.

7.11 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:

7.11.1 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.

7.11.2 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.

7.11.2.1 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 "7.11.2.2 Stopping the Instance Using the Immediate Mode" or "7.11.2.3 Forcibly Stopping the Server Process".

📝 Example
. . . .
```
> pg_ctl stop -D /database/inst1 -m fast
```
. . . .

7.11.2.2 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 "7.11.2.3 Forcibly Stopping the Server Process".

📝 Example
. . . .
```
> pg_ctl stop -D /database/inst1 -m immediate
```
. . . .

7.11.2.3 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
symfo    19174 18027 pts/1                \_ grep postgres
symfo    20517     1 ?          /opt/symfoserver64/bin/postgres -D /database/inst1
symfo    20518 20517 ?              \_ postgres: logger process
```

```
symfo    20520  20517 ?              \_ postgres: checkpointer process
symfo    20521  20517 ?              \_ postgres: writer process
symfo    20522  20517 ?              \_ postgres: wal writer process
symfo    20523  20517 ?              \_ postgres: autovacuum launcher process
symfo    20524  20517 ?              \_ postgres: archiver process
symfo    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
   ```

7.12 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 | 2013-08-06 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.

P 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
```

. . . .

Appendix A Parameters

This appendix describes the parameters to be set in the postgresql.conf file of Symfoware Server.

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.
. . . .

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 Symfoware Server.

📚 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 View

This appendix describes how to use the system view in Symfoware Server.

🔖 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

Appendix D Activating and Stopping the Web Server Feature of WebAdmin

To use WebAdmin for creating and managing a Symfoware Server instance on a server where Symfoware Server 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 Symfoware Server is installed in /opt/symfoserver64:

   ```
   # cd /opt/symfoserver64/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 Symfoware Server is installed in /opt/symfoserver64:

   ```
   # cd /opt/symfoserver64/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 Symfoware Server

  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 (26500) 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.

- SYMFO_HOME

  Set the Symfoware Server installation directory.

Index