ExpressCluster for Linux Ver3.1

Database Agent

2005.12.28
4rd Revision

Revision History

Revision	Revision Date	Revision Page	Description
1	2005/05/30		The 1st Revision was created.
2	2005/07/22		The following description has been added because the Sybase monitor command is supported in R3.0-2 or later.
		13	1) 2.2 Uninstalling Script Templates
			Deleting process for the script template folder for Sybase has been added.
		20	2) 5.1 Monitoring Application
			Sybase has been added to database applications that can be monitored.
		27	3) 7.1 List of Database Monitoring Commands
			Description on Sybase monitor command has been added.
		47	4) clp_sybmon
			The command for Sybase has been added.
		47	5) clp_sybmon
			Descriptions on messages produced by clp_sybmon have been added.
		131	6) 10.17 For Sybase Adaptive Server Enterprise(ASE)
			Description on script template for Sybase has been added.
			The following description has been added/modified.
		47	Information on the library name used by clp_msql41mon has been added to the Note 1 in the MySQL monitor command.
3	2005/09/09	In general	The description is corrected.
4	2005/12/28		The following description is added by monitoring command correspondence for PostgreSQL8.0 after R3.0-3.
		9	The description about the latest version of a script templates is added.
		13	Description of the deletion folder at the time of uninstallation of script templates of the Windows version is changed.
		20	The information on the application for monitoring of R3.0-3 is added.
		27	clp_psql80mon is added to a monitoring command list.
		38	The information on clp_psql80mon is added.
		57	The message of clp_psql80mon is added.
		107	The contents of the script templates for clp_psql80mon are added.

			Description is added and corrected to below by the agent correspondence for IA64 servers.
		8	The hardware requirement table of the Database Agent is corrected.
		9	The installation procedure of the Database Agent for IA64 servers is added.
		13	The folder name uninstalled by the Windows version and trekking tool is corrected.(It united with the information on the newest trekking tool currently opened to Web.)
		20	The table of the application for monitoring is added and corrected.
		22	Description is added to [Important].
		72	Bitmap is corrected.
		72 -	The description about [Start Monitor Wait Time] and [Re-active Threshold] is added.

This construction guide describes "NEC ExpressCluster Database Agent for Linux R3.0-3".

The latest information on system confirmation, system configuration guide, update, and tracking tool is provided in the following URL.
Please obtain the latest version before configuring the system.

Usage on the NEC Internet:
http://soreike.wsd.mt.nec.co.jp/

Usage out of the NEC Internet:
http://www.ace.comp.nec.co.jp/CLUSTERPRO/

Part I The Basics

1 Setting Up the Database Agent

1.1 Setting Up the Monitoring Module

Note:
This document assumes that users use the ExpressCluster CD as the installation media. If you use a different kind of media or trial version, refer to the document that came with your media, and replace paths and others according to your environment settings.

1.1.1 Before Installation

Check the followings before installing the Database Agent on a server.

The Database Agent requirements are as follows. Check each item on all servers on which the Database Agent will be installed.

Requirements to use the Database Agent (monitoring module)	
Hardware	IA32 server, x86_64 server, IA64 server
OS	The same as the ExpressCluster server and allowing operation of the database system to be monitored.
ExpressCluster	ExpressCluster SE for Linux Ver3.1-3 or later ExpressCluster LE for Linux Ver3.1-3 or later ExpressCluster SX for Linux Ver3.1-3 or later ExpressCluster LX for Linux Ver3.1-3 or later
Memory size	7 M byte (per command)
Disk size	1 M byte

Obtain the latest update for the Database Agent. Refer to the Update Manual for how to apply updates.

1.1.2 Installing the Database Agent on Linux System

Set up the Database Agent after installing the ExpressCluster. Apply the latest update for the ExpressCluster.

Log in as a root user and follow the steps below when installing the Database Agent on Linux system.

(1) Insert the CD (ExpressCluster CD) into a CD drive.

(2) Mount the CD.

```
# mount /dev/cdrom
```

(3) Move to the following directory:

```
# cd /mnt/cdrom/Linux/3.0/en/option
```

(4) Install the Database Agent by using the rpm command.

For Linux IA32

```
# rpm -i --nodeps expresscls-dbmon-3.0-1.i386.rpm
```

For Linux x86_64

```
# rpm -i --nodeps expresscls-dbmon-3.0-1.x86_64.rpm
```

For Linux IA64

```
# rpm -i --nodeps expresscls-dbmon-3.0-3.ia64.rpm
```

*You need to check the rpm file name since it may vary depending on the version of the Agent, etc.

(5) After installing the Agent, register the license for the Database Agent.

Register the license by following the procedure in 3 “License Registration”.

Supplement) The command description may vary depending on the types of Linux.

Note) If you set up the Database Agent while monitoring an application using the Database Agent on the ExpressCluster, processes may fail to finish successfully. When setting up the Database Agent, either stop the failover group which is monitoring the database, or move to the server in which set up is not performed.

1.2 Setting Up Script Templates

Script templates are available for Windows version trekking tool and a Linux version trekking tool. Choose and setup scrip templates according to your requirements.

The latest version of a script templates should come to hand. Please apply with reference to the update procedure document for script templates.

1.2.1 Installing Script Templates on Windows

Set up script templates after Windows version trekking tool has been set up. This is because the script templates are installed in a folder having the trekking tool scripts. You cannot install script templates on a terminal where trekking tool is not installed.

When installing the script templates on Windows, log in as a user with administrator's authority.

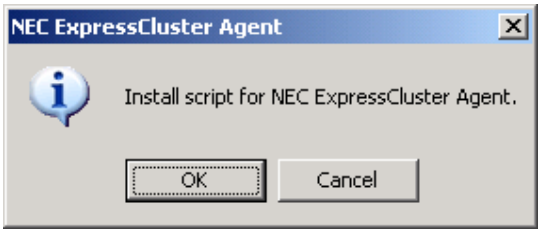
(1) Insert the CD (ExpressCluster CD) into a CD drive.

The Setup Menu window appears automatically. If the window does not automatically start up, execute the menu.exe on the CD drive directly.

(2) Click [NEC ExpressCluster for Linux].

(3) Click [NEC ExpressCluster Template Scripts for Agent].

The setup message appears. Click [OK] to install.



Supplement) SETUP.EXE for script installation automatically searches the folder of trekking tool and then performs installation. Specify the folder name directly if the search cannot be executed correctly or if it takes time. Specify the following folder name when you specify a folder name.

Example: When the trekking tool is installed in C:\Program Files\NEC\clptrek
(clptrek.htmlb is in the C:\Program Files\NEC\clptrek folder)
A:\SETUP.EXE **"C:\Program Files"**

When specifying the folder name where the script templates are to be installed

```
>CD  Q:
>CD  \Linux\3.0\en\script\win
>\SETUP.EXE    [installation_folder_of_the_trekking_tool]
```

*Check the underlined CD drive name by using Windows Explorer, etc. since the name may vary depending on the environment of terminals.

1.2.2 Installing Script Templates on Linux System

Set up script templates after Linux version trekking tool has been set up. The script templates will be installed in the directory where the trekking tool scripts exist.

When installing the script templates on Linux, log in as a root user and follow the steps below.

(1) Insert the CD (ExpressCluster CD) into a CD drive.

(2) Mount the CD.

```
# mount  /dev/cdrom
```

(3) Move to the following directory:

```
# cd  /mnt/cdrom/Linux/3.0/en/script/Linux
```

(4) Install the script templates using the rpm command.

For Linux IA32

```
# rpm  –i  expresscls-monscript-3.0-1.i386.rpm
```

For Linux x86_64

```
# rpm  –i  expresscls-monscript-3.0-1.x86_64.rpm
```

(*) The rpm file name may vary depending on the version of the Agent.
The script templates are for all Agents.

Supplement) The command description may vary depending on the type of Linux.

2 Uninstalling the Database Agent

2.1 Uninstalling the Monitoring Module

When uninstalling a monitoring module, run the following command as a root user.

```
# rpm –e expresscls-dbmon
```

Note) If you uninstall the Database Agent while monitoring an application using the Database Agent on the ExpressCluster, processes may fail to finish successfully. When uninstalling the Database Agent, either stop the failover group that is performing database monitoring, or move to the server in which uninstallation is not perfromed.

When uninstalling the ExpressCluster, the Database Agent will be uninstalled as well but the rpm module information will be left as is. Therefore, it is necessary to forcefully delete the package using the rpm command.

Example: rpm –e --force expresscls-dbmon

2.2 Uninstalling Script Templates

2.2.1 Uninstalling the Script Templates for Windows

To uninstall script templates, run the following command as a user with administrator's authority.

```
>RMDIR /S "C:\Program Files\NEC\clptrek\scripts\linux\db2v8"
>RMDIR /S "C:\Program Files\NEC\clptrek\scripts\linux\db2v8-mon"
>RMDIR /S "C:\Program Files\NEC\clptrek\scripts\linux\http"
>RMDIR /S "C:\Program Files\NEC\clptrek\scripts\linux\http-mon"
>RMDIR /S "C:\Program Files\NEC\clptrek\scripts\linux\mysql"
>RMDIR /S "C:\Program Files\NEC\clptrek\scripts\linux\mysql3.23-mon"
>RMDIR /S "C:\Program Files\NEC\clptrek\scripts\linux\mysql4.0-mon"
>RMDIR /S "C:\Program Files\NEC\clptrek\scripts\linux\mysql4.1-mon"
>RMDIR /S "C:\Program Files\NEC\clptrek\scripts\linux\nfs"
>RMDIR /S "C:\Program Files\NEC\clptrek\scripts\linux\nfs-mon"
>RMDIR /S "C:\Program Files\NEC\clptrek\scripts\linux\oracle9i"
>RMDIR /S "C:\Program Files\NEC\clptrek\scripts\linux\oracle9i-mon"
>RMDIR /S "C:\Program Files\NEC\clptrek\scripts\linux\oracle10g"
>RMDIR /S "C:\Program Files\NEC\clptrek\scripts\linux\oracle10g-mon"
>RMDIR /S "C:\Program Files\NEC\clptrek\scripts\linux\postgresql"
>RMDIR /S "C:\Program Files\NEC\clptrek\scripts\linux\postgresql7.2-mon"
>RMDIR /S "C:\Program Files\NEC\clptrek\scripts\linux\postgresql7.3-mon"
>RMDIR /S "C:\Program Files\NEC\clptrek\scripts\linux\postgresql8.0-mon"
>RMDIR /S "C:\Program Files\NEC\clptrek\scripts\linux\powergres1.1"
>RMDIR /S "C:\Program Files\NEC\clptrek\scripts\linux\powergres1.1-mon"
>RMDIR /S "C:\Program Files\NEC\clptrek\scripts\linux\samba"
>RMDIR /S "C:\Program Files\NEC\clptrek\scripts\linux\samba-mon"
>RMDIR /S "C:\Program Files\NEC\clptrek\scripts\linux\smtp"
>RMDIR /S "C:\Program Files\NEC\clptrek\scripts\linux\smtp-mon"
>RMDIR /S "C:\Program Files\NEC\clptrek\scripts\linux\sybase"
>RMDIR /S "C:\Program Files\NEC\clptrek\scripts\linux\sybase-mon"
>RMDIR /S "C:\Program Files\NEC\clptrek\scripts\linux\tuxedo8.1"
>RMDIR /S "C:\Program Files\NEC\clptrek\scripts\linux\tuxedo8.1-mon"
>RMDIR /S "C:\Program Files\NEC\clptrek\scripts\linux\weblogic8.1"
>RMDIR /S "C:\Program Files\NEC\clptrek\scripts\linux\weblogic8.1-mon"
>RMDIR /S "C:\Program Files\NEC\clptrek\scripts\linux\websphere6.0"
>RMDIR /S "C:\Program Files\NEC\clptrek\scripts\linux\websphere6.0-mon"
```

(*) Check the folder name, etc. since the underlined file name may vary depending on the terminal environment.

2.2.2 Uninstalling the Script Templates for Linux

To uninstall script templates, run the following command as a root user.

```
# rpm –e expresscls-monscript
```

Uninstalling the Linux Trekking Tool also uninstalls the script templates (for Linux TrekkingTool), however, rpm module information is not removed. Therefore, you have to remove the package forcibly with rpm command before reinstalling it.

i.e. rpm -e --force expresscls-monscript

3 License Registration

To use this product, register the license after installing the monitoring module.
To register the license, log in as a root user from the server which has the monitoring module installed. You have to register a unique license for each server.
Register the license by the following steps below.
License registration is not required for script templates.

Note 1:
Depending on your ExpressCluster version, you may not be able to register the license properly. In such a case, you have to update your ExpressCluster.
Note 2:
You have to register a unique Database Agent license key for each server in a cluster. Otherwise, license keys collide in a cluster, consequently, a license error occurs at the Database Agent startup. See "8 Alert Messages" for details of logged messages.

3.1 Interactive License Registration

(1) Run the following command on a server.

```
# clplcnsc –i –p DBMON30
```

(2) Enter 1 for the product division.

```
Selection of product division.
  1. Product
  2. Trial
Select product division. [1 or 2]...1
```

(3) Enter the product serial number.

```
Enter serial number [Ex. XX000000]... xxnnnnnn
```

Specify the number indicated on the license sheet as the serial number.

(4) Enter the product license key.

```
Enter license key
[Ex. XXXXXXXX-XXXXXXXX-XXXXXXXX-XXXXXXXX]... xxxxxx
xx-xxxxxxxx-xxxxxxxx-xxxxxxxx
```

Specify the number indicated on the license sheet as the license key. Enter the license sheet information exactly as indicated because the license key is case sensitive. Note that no "I"s and "O"s in upper case are used in the license key.

After running the command, check that the message "command was success" is indicated and the command is successfully completed. For the other ending messages, refer to a separate ExpressCluster configuration and installation guide, "*Commands.*"

3.2 License Registration By Specifying A License File

(1) Run the following command on a server.

```
# clplcnsc –i filepath –p DBMON30
```

Specify the path to the distributed license file in the file path of –i option.
You can see on the console a message, "command was success." when the command is completed if the registration is successful. See ExpressCluster Construction Guide, "Commands" for details of other completion messages.

A license file may come with the trial version only.

4 Using the Database Agent

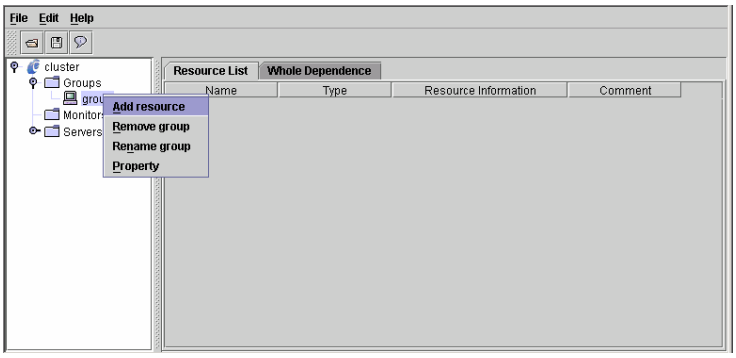
This section briefly explains how to use the Database Agent. For the details, refer to "Part II Details" as necessary.

4.1 Setting Monitoring Commands

The description here assumes a database system is already constructed.

To monitor a database, add an EXEC resource for monitoring to a failover group which starts up and stops the database system. Use trekking tool to add a resource.

Adding an EXEC Resource



Write scripts to start and stop the monitoring command in an EXEC resource. The following is an example of Oracle monitoring commands.

start.sh

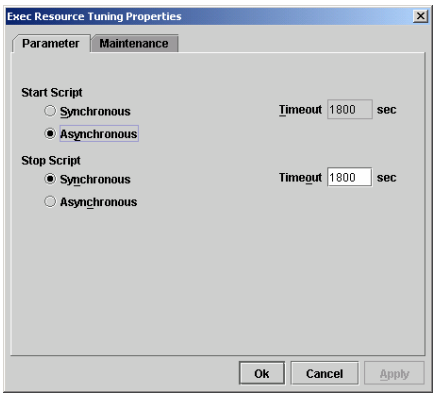
```
clp_ora10mon orawatch –d database
```

stop.sh

```
clp_ora10mon orawatch --stop
```

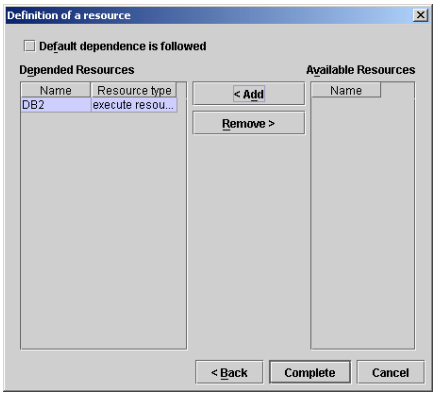
Configure the EXEC resource by choosing "Asynchronous" for start up script and leaving the stop script as "Synchronous."

EXEC Resource Setting



Establish dependency with the EXEC resource of start-up and deactivation of the database system.

Dependency Set Up

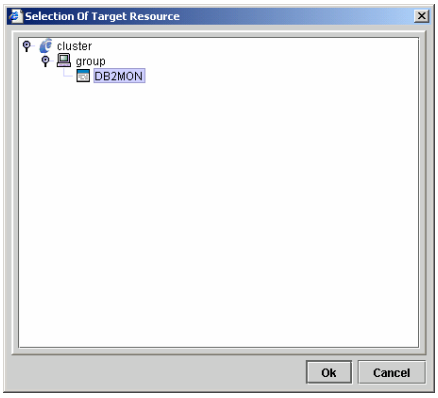


Due to the settings you made, the monitoring command starts up after the database system starts up, and starts monitoring.

4.2 Setting PID Monitoring

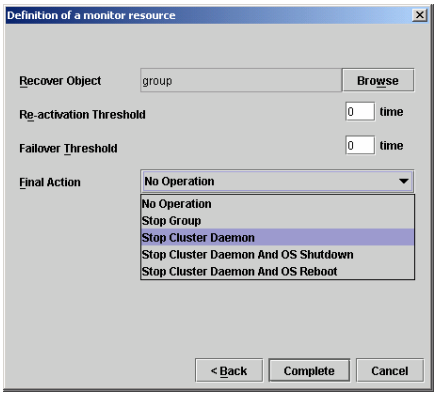
Set up the monitor resource for the EXEC resource which starts up monitoring commands. A monitoring command terminates its process when an error is detected in the database. Therefore, it is possible to detect an error in the database by looking at the termination of the monitoring command process through PID monitoring of a monitor resource.

Setting an EXEC Resource for monitoring command in monitoring PID



Configure how ExpressCluster behaves when an error is detected in the database by the PID monitoring resource. Generally set "0" for "Failover Threshold" and "Stop Cluster Daemon" for "Final Action."

Setting action to be taken when an error is detected



With the settings above, you can start monitoring a database system by using monitoring commands.

Part II Details

5 Database Monitoring

5.1 Monitoring Application

The Database Agent monitors databases working under the ExpressCluster environment. The following chart shows the version of the Database Agent and database application that can be monitored.

For Linux x86

Database	R3.0-1	R3.0-2	R3.0-3
DB2 Universal Database V8.1/V8.2	+	+	+
Oracle9i Database Release 2	+	+	+
Oracle Database 10g Release 1	+	+	+
Oracle Database 10g Release 2			+
PostgreSQL 7.2	+	+	+
PostgreSQL 7.3/7.4	+	+	+
PostgreSQL 8.0			+
PowerGres Plus V1.1	+	+	+
MySQL 3.23	+	+	+
MySQL 4.0	+	+	+
MySQL 4.1	+	+	+
Sybase Adaptive Server Enterprise 12.5.2	-	+	+

+:Supported -:Not supported

For Linux x86_64

Database	R3.0-1	R3.0-2	R3.0-3
DB2 Universal Database V8.2(*1)	+	+	+
Oracle9i Database Release 2	+	+	+
Oracle Database 10g Release 1	+	+	+
PostgreSQL 7.2	-	-	-
PostgreSQL 7.3/7.4	+	+	+
PostgreSQL 8.0	-	-	+
PowerGres Plus V1.1	-	-	-
MySQL 3.23	+	+	+
MySQL 4.0	+	+	+
MySQL 4.1	+	+	+
Sybase Adaptive Server Enterprise 12.5.2	-	+	+

+:Supported -:Not supported

*1: Only the database monitoring of the instance of which the width is 64 bits is supported (the database monitoring of the instance of which the width is 32 bits is not supported).

For Linux IA64

Database	R3.0-1	R3.0-2	R3.0-3
DB2 Universal Database V8.2	-	-	+
Oracle9i Database Release 2	-	-	+
Oracle Database 10g Release 1	-	-	+
PostgreSQL 7.2	-	-	-
PostgreSQL 7.3/7.4	-	-	-
PostgreSQL 8.0	-	-	-
PowerGres Plus V1.1	-	-	-
MySQL 3.23	-	-	-
MySQL 4.0	-	-	-
MySQL 4.1	-	-	-
Sybase Adaptive Server Enterprise 12.5.2	-	-	-

+:Supported -:Not supported

To monitor databases, monitoring commands for each database are provided.

Refer to "7 Database Monitoring Commands" for the detailed information of commands.

For information on versions supporting PostgreSQL and MySQL, visit the ExpressCluster website.

5.2 Overview of Monitoring

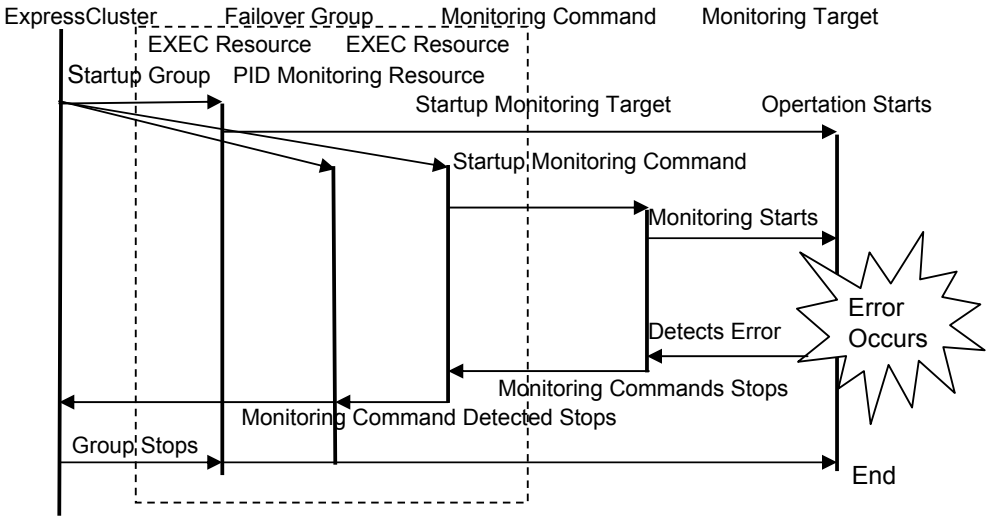
The Database Agent offers commands described in a script to monitor the operation of the database.

If you run these monitoring commands, monitoring is performed with the intervals specified by the parameter until a database error is detected. These commands stop when a database error is detected.

You use these commands by writing them in a script of an EXEC resource for an ExpressCluster failover group.

Since these commands stop when a database error is detected, you can achieve failover or server shutdown by configuring the EXEC resource.

Overview of Monitoring Operation



A failover group will stop by failover and server shutdown, etc, by PID monitoring resource which detected an error of the EXEC resource which starts the monitoring command.

[Important]

(1) These commands can detect an error which does not cause the monitoring target to end abnormally (mainly stalling). Although it cannot detect an abnormal ending of a target application to be monitored, it can indirectly detect a failure through process of monitoring operation of the target (such as database access) .

(2) The purpose of using these commands is to monitor the operation of a target application to be monitored, but not to investigate or diagnose the cause when an error occurs on the application. When an error occurs, you need to use other means, such as application logs, to see details of its cause.

(3) Whenever it performs monitor processing depending on the application for monitoring, an access log may be outputted to syslog etc., or a log may be outputted to the local directory of the application for monitoring. About these setup, since it is uncontrollable by this monitoring command, please set up suitably with the application for monitoring. However, when not outputting the log of the application for

monitoring, the log at the time of obstacle generating is not outputted, either, but cause investigation may become difficult.

(4) Use a stop command to stop these monitoring commands. If a process was stopped by using the kill command of Linux, the monitoring command may not be restarted since the monitoring command management information will not be initialized.

(5) Since these monitoring commands run as a client application of a database system, configuration to run client application on a server is necessary. Refer to the manuals for each databases system for the details.

5.3 Writing Scripts to an EXEC Resource

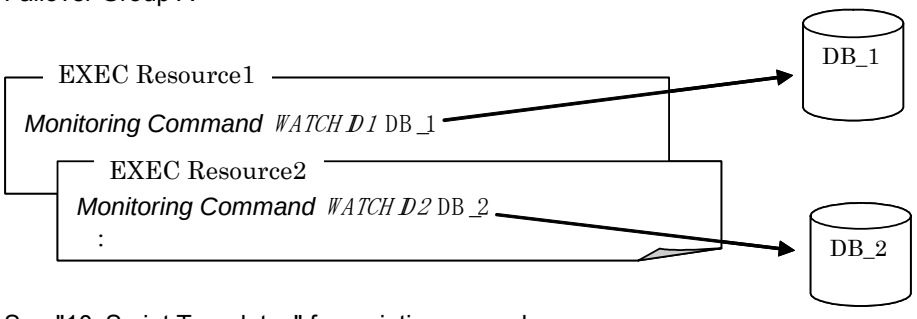
When you write a script on an EXEC resource, the following must be noted.

- Before creating an EXEC resource in which you will write start/end of the Agent, complete the EXEC resource in which start/end of a monitoring target application is written. Then, check if a failover group successfully starts, stops, moves, and fails over. If you create an EXEC resource where the start and end of the Agent are written without checking these issues, it would be difficult to determine the cause when an error was detected in the startup of a failover group by the Agent; If the error really occurred, the environment of the monitoring target application was not correctly set up, or a parameter value of the Agent is not appropriate.

- Configure the dependency relationship of resources so that the EXEC resource for start/stop of the Agent starts up after the EXEC resource in which start/end of an application to be monitored is written starts up. If you make incorrect settings, the Agent may consider the monitoring target application as an error.

These monitoring commands can also be written to monitor multiple databases in one failover group.

Example:
Failover Group A



See "10 Script Templates" for scripting examples.

5.4 Operation Check

Check on the Web Manager screen to see the failover group is correctly operating by following the steps below.

5.4.1 Checking Behavior of Target Application to Be Monitored

Before adding an EXEC resource for monitoring commands, check that a target application to be monitored is operating normally by following the steps below.

5.4.1.1 Start up a Group

Start up a selected failover group.
After selecting the failover group which you wish to start on the tree view on the main screen, display a menu by right clicking, and then select [Start].

5.4.1.2 Stop a Group

Stop the selected failover group.
After selecting the failover group which you wish to stop on the tree view on the main screen, display a menu by right clicking, and then select [Stop].

Check the group starts up and stops on all servers that start up a failover group.

5.4.1.3 Move a Group

Move a failover group among servers.
Start a group and move it to other servers consecutively.
After selecting the failover group which you wish to move on the tree view on the main screen, display a menu by right clicking, and then select [Move].

Depending on the script, it may take a few minutes to complete moving. See the tree view on the main screen to check completion of moving.

5.4.1.4 Failover a Group

Perform failover.
Start up a group and shut down its server. Check that the failover group has failed over to its failover destination server.
After selecting a server which you wish to shut down on the tree view on the main screen, display a menu by right clicking, and then select [reboot].

5.4.2 Checking Monitoring Command Operation

After making sure that the target application to be monitored operates normally, add

an EXEC resource to start up the monitoring commands and a monitor resource which monitors the EXEC resource for starting up monitoring commands. After updating a failover group, check that monitoring commands are running normally by performing the following as the way you checked behavior of a target application to be monitored.

5.4.2.1 Start Up a Group

Start up a selected failover group.

After selecting a failover group, which you wish to start up, on the tree view on the main screen, display a menu by right clicking and then select [start].

If a monitoring command displays an error message in the ExpressCluster manager when the group is started, a parameter value of the monitoring command may not be appropriate. If the error message is displayed on a particular server, there may be an error in the environment settings of the target application to be monitored.

5.4.2.2 Stop a Group

Stop a selected failover group.

After selecting a failover group which you wish to deactivate on the tree view on the main screen, display a menu by right clicking and then select [Stop].

Check the group starts up and stops on all servers that start up a failover group.

5.4.2.3 Move a Group

Move a failover group among servers.

Start a group and move it to other servers consecutively.

After selecting a failover group which you wish to migrate on the tree view on the main screen, display a menu by right clicking and then select [Move].

5.4.2.4 Failover a Group

Perform failover.

Start up a group and shut down its server. Check that the failover group has failed over to its failover destination server.

After selecting a server, which you wish to shut down on the tree view on the main screen, display a menu by right clicking and select [Reboot].

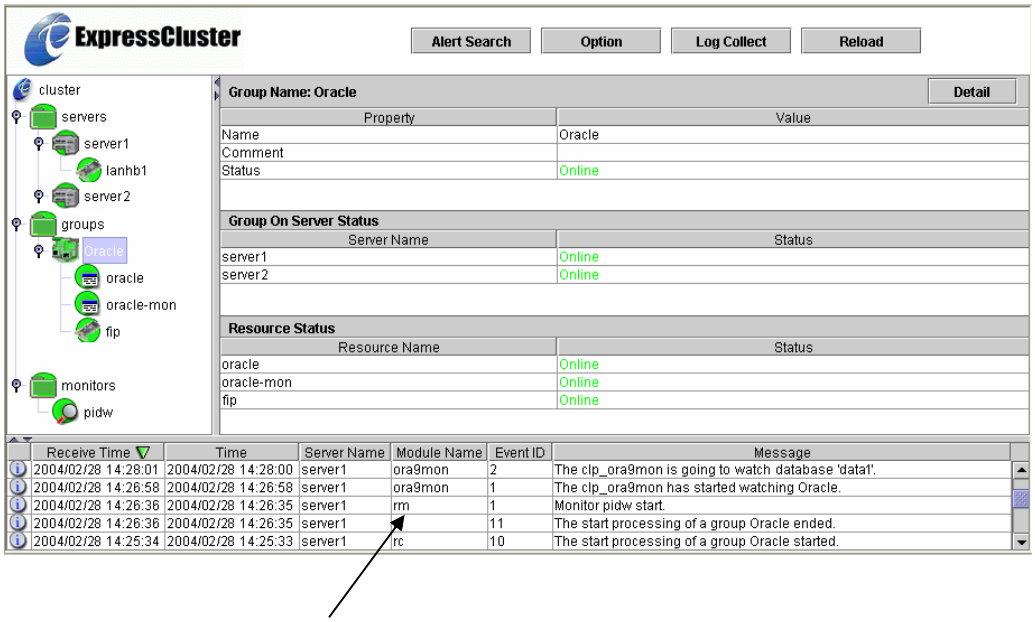
[CAUTION]
Monitoring may not be performed normally depending on the system environment if you invoke the monitoring command with an extremely small value (e.g. 1) specified to the parameters of monitoring interval and response time. Run sufficient operational check before performing these operations.

6 Information Provided by Monitoring Commands

Monitoring commands display a monitoring status on the alert view of the ExpressCluster Web Manager.

The same message as that logged in Alert View is also logged to syslog on the server where the watching command is running.

6.1 Alert Messages



Displayed on an alert view of the Manager.

A single line in a message is up to 400 bytes. If information to be shown in a message is longer than 400 bytes, the message will be displayed in two or more lines. Other message may interrupt a message shown in multiple lines depending on timing.

The same information as shown in the alert message is output to the syslog.

See "8 Alert Messages" for the alert message details.

6.2 Log Collection in Case of an Error

Error logs of monitoring commands will be produced in the same folder where error logs of the ExpressCluster server are produced. Logs are collected in the same way as ExpressCluster logs are collected. For more information, see "ExpressCluster for Linux Ver3.0 Web Manager" and "ExpressCluster for Linux Ver3.0 Commands".

7 Database Monitoring Commands

7.1 List of Database Monitoring Commands

The Database Agent provides database monitoring commands to be written in script.

Command	Used to	See page
clp_db28mon	Monitor DB2. This command supports DB2 Universal Database V8.1/V8.2	30
clp_ora9mon	Monitor Oracle. This command supports Oracle9i Database Release 2	34
clp_ora10mon	Monitor Oracle. This command supports Oracle Database 10g Release 1	34
clp_psql72mon	Monitor PostgreSQL. This command supports Postgre SQL 7.2	38
clp_psql73mon	Monitor PostgreSQL. This command supports Postgre SQL 7.3/7.4 This command supports PowerGres Plus V1.1	38
clp_psql80mon	Monitor PostgreSQL. This command supports Postgre SQL 8.0	38
clp_mysql323mon	Monitor MySQL This command supports MySQL 3.2.	43
clp_mysql40mon	Monitor MySQL This command supports MySQL 4.0	43
clp_mysql41mon	Monitor MySQL This command supports MySQL 4.1	43
clp_sybmon	Monitor Sybase Adaptive Server Enterprise This command supports Sybase Adaptive Server Enterprise 12.5	47

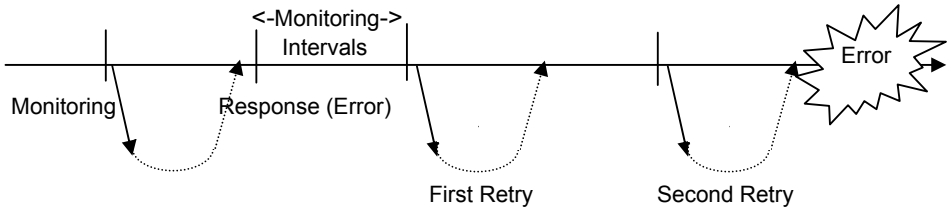
[Important]

(1) These commands should be run by a root user. If a database monitoring command is run by a user other than a root user, acquiring license and other process may fail. Therefore, the command fails to be run.

(2) When running these commands,/usr/sbin must be added to a path. Typically /usr/sbin is added to a path.

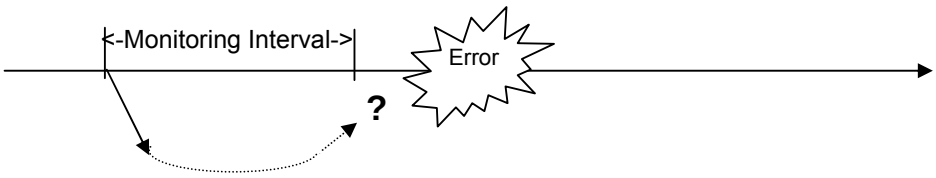
7.2 Monitoring Chart

The Database Agent detects an error by the following timing.



*In this case, the retry count is 2 times.

If there is no response within a specified response wait time, it is immediately regarded detects as an error.



7.3 Interrupting and Resuming Monitoring

You can interrupt/resume monitoring by the Database Agent using the following method. While monitoring is suspended, you can perform database maintenance.

(1) Start the monitoring command to start monitoring.

(2) To interrupt monitoring, run the following command from a server console whenever you wish to interrupt.
"Monitoring Command ***watchid*** --pause "

(3) The following message will be displayed on the ExpressCluster manager indicating that the monitoring process is interrupted.
"The *xxxx* is going to stop monitoring. [ID:***watchid***]"

(4) To resume monitoring, run the following command from a server console whenever you wish to resume.
"Monitoring Command ***watchid*** --continue"

(5) The following message will be displayed on the ExpressCluster manager indicating that the monitoring process is resumed.
"The *xxxx* is going to monitor. [ID:***watchid***]"

7.4 Command Syntax

This section provides information on command functions. Note the following to understand functions of commands.

- Command line
  - Gives an idea what the user actually enters.
  - [ ] indicates that the enclosed parameter may be skipped.
  - | indicates one of parameters separated by this symbol should be chosen.
- Description
  - Gives information what is performed.
- Parameter
  - Explanation about parameters shown in the command line.
- Supplementary information
  Gives supplementary information such as details on parameter settings
- Monitoring method
  - An explanation on how to monitor
- Note
  Gives information on what should be noted when using the command.
- Command usage example
  An example of scripting when using the command

clp_db28mon	Monitors DB2

Command line

Start Monitoring clp_db28mon	*Identifier* -d *Database Name* [-m *Instance Name*] [-u *User Name*] [-p *Password*] [-t *Table Name*] [-i *Monitoring Interval* [-c *Retry Count*] [-r *Response Wait Time*]
Stop Monitoring clp_db28mon	*Identifier* --stop [*Stop Wait Time*]
Interrupt Monitoring clp_db28mon	*Identifier* --pause
Resume Monitoring clp_db28mon	*Identifier* --continue
Display Information clp_db28mon	*identifier* --disp
Delete Information clp_db28mon	*Identifier* --del

Description

When a database name is specified, this command monitors DB2 per database. The command stops when a DB2 error is detected. The command also stops, interrupts and resumes monitoring. Use a root privileged console to specify --pause/--continue/--disp/--del.

Option

Identifier	Specifies an identifier to uniquely identify the monitoring command. You must set an identifier.
-d *Database name*	Specifies the name of a database to be monitored. You must set this option.
-m *Instance name*	Specifies the name of an instance (database manager) in a database to be monitored. Default value: db2inst1
-u *User name*	Specifies the name of a user to log in a database. Default value: db2inst1
-p *Password*	Specifies a password used to log in a database Default value: ibmdb2

-t *Table name*	Specifies the name of a monitoring table to be created on a database. Default value: db2watch
-i *Monitoring Interval*	Specifies an interval of monitoring a database in seconds (between 1 and 10000). Default value: 60
-c *Retry count*	Specifies retry count to be made (between 1 and 10000) when an error is detected during database monitoring Default value: 2
-r *Response Wait Time*	Specifies a response wait time (between 1 and 10000) of a database monitoring Default value:120
--stop	Stops the monitoring command
Stop Wait Time	Specifies time to wait for the monitoring command to stop normally (between 1 and 10000) y. Default value: 60
--pause	Temporary interrupts monitoring.
--continue	Resumes monitoring.
--disp	Displays a process ID (pid) of the monitoring command identifier managed by the monitoring command. Do not use this in ordinary use. (See note 4).
--del	Deletes information about the monitoring command identifier managed by the monitoring command. Do not use this in ordinary use. (See note 4).

Remarks

About identifier

You need to specify an identifier that is unique on the system to control the monitoring command. You cannot use an identifier that is already used by a monitoring command for another monitoring command. Specify an identifier using alphanumeric characters. An identifier should be up to 255 bytes. An identifier is case sensitive.

An identifier must be written as the first parameter of a monitoring command.

About the -u and –p parameters

It is not usually necessary to specify –u and -p parameters when accessing a database under a user name which has the same value as an instance name. When you access a database under a different user from an instance name, specify these parameters.

About the -t parameter

A table of values specified with the –t parameter is created on a database specified by the –d parameter. Make sure the –t parameter table name and the names of tables used in the operation do not overlap.

About the -d,-m,-u,-p, and -t parameters
The maximum length of a character sting for these parameters is 255 bytes. Effective length varies by each parameter but this command does not check the effective length. Effective length is determined by the DB2 specifications.

Monitoring method

This monitoring command performs the following monitoring.
By creating a monitoring table on the database and issuing an SQL statement, the command allows reading and writing of a maximum of 5-digit numerical data.
As a result of monitoring, it is considered an error if:

(1) No response is returned for database access or issue of SQL statement within a response wait time (-r parameter value)
(2) Error is returned for database access or issue of SQL statement.
(3) The written data and read data do not match.

The SQL statement to be used is create/drop/insert/update/select.

Note 1

This monitoring command monitors DB2 using CLI library of DB2. Therefore, it is necessary to run "source *home of instance user*/sqllib/db2profile" as a root user. Write this in a start up script.

Note 2

This monitoring command cannot access a DB2 database if the database code page and the root user code page, which runs this monitoring command, are different. Write "export LANG=ja_JP.ujis" and such in a start up script as necessary. Check the database code page by "db2 get db cfg for *database name*". For more information, see the DB2 manual.

Note 3

You cannot monitor a DB2 if the values for parameters including database name, instance name, user name, or a password are different from the values in the DB2 to be monitored. As you see a message with error information, check your environment.

Note 4

If this monitoring command process is stopped using a kill command during monitoring, a monitoring command of the same identifier may not start because the management information is not successfully initialized. In such case, if you run the "clp_db28mon identifier --disp" command, the process ID which corresponds to the specified identifier will be displayed. Check the run file of the process ID and if it is in process other than the monitoring command, delete the management information using the "clp_db28mon identifier --del" command.
Never specify a normally operating identifier and run the "clp_db28mon identifier --del" command because the monitoring command will malfunction.

Note 5

When monitoring the database of DB2 V8.2 on the x86_64 server, monitor the database created by the instance of which the width is 64 bits. The DB2 monitoring command cannot monitor the database created by the instance of which the width is 32 bits.

Example of Command Usage	[start.sh] export LANG=ja_JP.ujis source /home/db2inst1/sqllib/db2profile clp_db28mon db2watch –d *database name* [stop.sh] source /home/db2inst1/sqllib/db2profile clp_db28mon db2watch --stop This command starts up from an EXEC resource.

clp_ora9mon **clp_ora10mon**	Monitors Oracle9i Monitors Oracle10g

Command line

Start Monitoring clp_ora9mon

Identifier -d *Connection String*
[-u *User Name*] [-p *Password*] [-t *Table Name*]
[-i *Monitoring Interval* [-c *Retry Count*] [-r *Response Wait Time*]

Stop Monitoring clp_ora9mon

Identifier --stop [*Stop Wait Time*]

Interrupt Monitoring clp_ora9mon

Identifier --pause

Resume Monitoring clp_ora9mon

Identifier --continue

Display Information clp_ora9mon

identifier --disp

Delete Information clp_ora9mon

Identifier --del

clp_ora10mon's command syntax is the same as above.

Description

When a connection string is specified, this command monitors Oracle per database. This monitoring command stops when an Oracle error is detected.
The command also stops, interrupts and resumes monitoring.
Use a root privileged console to specify --pause/--continue/--disp/--del.

Option

Identifier	Specifies an identifier to uniquely identify the monitoring command. You must set an identifier.
-d *Connecting String*	Specifies a connection string corresponding to the database to be monitored. You must set this option.
-u *User name*	Specifies the name of a user to log in a database. Default value: sys
-p *Password*	Specifies a password used to log in a database. Default value: change_on_install
-t *Table name*	Specifies a monitoring table name to be created on the database. Default value: orawatch

-i *Monitoring Interval*	Specifies an interval of monitoring a database in seconds (between 1 and 10000). Default value: 60
-c *Retry count*	Specifies retry count to be made (between 1 and 10000) when an error is detected during database monitoring Default value: 2
-r *Response Wait Time*	Specifies a response wait time (between 1 and 10000) of a database monitoring Default value:120
--stop	Stops the monitoring command
Stop Wait Time	Specifies time to wait for the monitoring command to stop normally (between 1 and 10000) y. Default value: 60
--pause	Temporary interrupts monitoring.
--continue	Resumes monitoring.
--disp	Displays a process ID (pid) of the monitoring command identifier managed by the monitoring command. Do not use this in ordinary use. (See note 4).
--del	Deletes information about the monitoring command identifier managed by the monitoring command. Do not use this in ordinary use. (See note 4).

Remarks

About identifier

You need to specify an identifier that is unique on the system to control the monitoring command. You cannot use an identifier that is already used by a monitoring command for another monitoring command. Specify an identifier using alphanumeric characters. An identifier should be up to 255 bytes. An identifier is case sensitive.

An identifier must be written as the first parameter of a monitoring command.

About the -u and –p parameters

For –u and –p parameters, specify an Oracle user who can access the database specified in -d parameter.

About the –t parameter

A table of values specified with the –t parameter is created on a database specified by the –d parameter. Make sure the –t parameter table name and the names of tables used in the operation do not overlap.

About the -d,-u,-p, and -t parameters

The maximum length of a character string for these parameters is 255 bytes. Effective length varies by each parameter but this command does not check the effective length. Effective length is determined by the Oracle specifications.

Monitoring method

This monitoring command performs the following monitoring.
By creating a monitoring table on the database and issuing an SQL statement, the command allows reading and writing of a maximum of 5-digit numerical data.
As a result of monitoring, it is considered an error if:

(1) No response is returned for database access or issue of SQL statement within a response wait time (-r parameter value)
(2) Error is returned for database access or issue of SQL statement.
(3) The written data and read data do not match.

The SQL statement to be used is create/drop/insert/update/select.

Note 1

This command monitors Oracle by using the Oracle interface (Oracle Call Interface). Therefore, it is necessary to have the library for interface (libclntsh.so) installed on the server to be monitored.

Note 2

You cannot monitor Oracle if values for parameters including connection string, user name, password are different from the values in the Oracle environment for monitoring. As you see a message with error information, check the environment.

Note 3

When only the OS authentication is available as DBA user authentication,
If REMOTE_LOGIN_PASSWORDFIL is set to NONE in Oracle initialization parameter file, specify a database user who does not have DBA right for –u and -p parameters of clp_ora9mon and clp_ora10mon.
Otherwise, monitoring is impossible because ORA-01031 error occurs at clp_ora9mon or clp_ora10mon startup.

Note 4

If this monitoring command process is stopped using a kill commands during monitoring, a monitoring command of the same identifier may not start because the management information was not successfully initialized. In such case, if you run the "clp_ora9mon identifier --disp" command, the process ID which corresponds to the specified identifier will be displayed. Check the run file of the process ID. If it is in process other than monitoring command, delete the management information using the "clp_ora9mon identifier --del" command.
Never specify a normally operating identifier and run the "clp_ora9mon identifier --del" command because the monitoring command will malfunction.

Note 5

Use the same character set supported by the OS when you create database.
If NLS_LANGUAGE is set to Japanese in Oracle initialization parameter file, set NLS_LANG (an Oracle environment variable) to English, and set the character set to that of database before you start the Database Agent.
If these character sets fail to match, alert messages of event ID (0) cannot properly be displayed on alert view.
However, in some cases, database connection errors such as

invalid use name may not be displayed properly even if two character sets are the same.
See an Oracle document "Globalization Support Guide" for details of NLS parameter and NLS_LANG settings.
See Section "8 Alert Messages " for details of alert messages.

Example of Command Usage

[start.sh]
export ORACLE_HOME=/u01/app/oracle/product/10.1.0/db_1
export LD_LIBRARY_PATH=$ORACLE_HOME/lib
export NLS_LANG=AMERICAN_AMERICA.JA16EUC
clp_ora10mon orawatch –d *database name*
[stop.sh]
export ORACLE_HOME=/u01/app/oracle/product/10.1.0/db_1
export LD_LIBRARY_PATH=$ORACLE_HOME/lib
clp_ora10mon orawatch --stop

This command starts up from an EXEC resource.
It is necessary to set environment variables, ORACLE_HOME and LD_LIBRARY_PATH, to run this monitoring command.

clp_psql72mon	Monitors PostgreSQL 7.2
clp_psql73mon	Monitors PostgreSQL 7.3/7.4
clp_psql80mon	Monitors PowerGres Plus V1.1 Monitors PostgreSQL 8.0

Command line

Start Monitoring clp_psql72mon

Identifier -d *Database Name*
[-a *IPAddress*] [-n *Port Number*]
[-u *User Name*] [-p *Password*] [-t *Table Name*]
[-i *Monitoring Interval* [-c *Retry Count*] [-r *Response Wait Time*]

Stop Monitoring clp_psql72mon

Identifier --stop [*Stop Wait Time*]

Interrupt Monitoring clp_psql72mon

Identifier --pause

Resume Monitoring clp_psql72mon

Identifier --continue

Display Information clp_psql72mon

identifier --disp

Delete Information clp_psql72mon

Identifier --del

The command syntax of clp_psql73mon and clp_psql80mon are the same as above.

Description

When a database name is specified, this command monitors PostgreSQL per database. This command stops when a PostgreSQL error is detected.
The command also stops, interrupts and resumes monitoring.
Use a root privileged console to specify --pause/--continue/--disp/--del.

Option

Identifier	Specifies an identifier to uniquely identify the monitoring command. You must set an identifier.
-d *Database name*	Specifies the name of a database to be monitored. You must set this option.
-a *IP address*	Specifies an IP address used to access PostgreSQL from a PostgreSQL client. Default value: 127.0.0.1

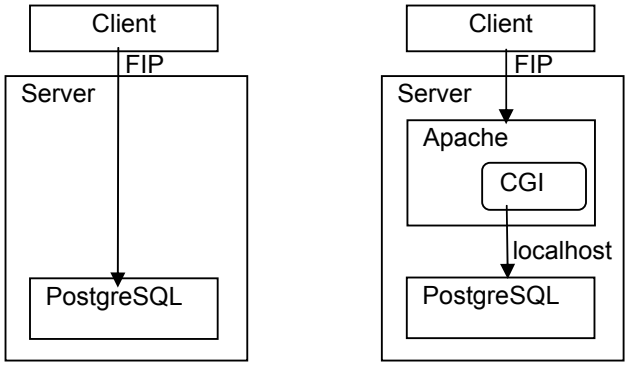
-n *Port number*	Specifies a PostgreSQL port number. Default value: 5432 (PG PORT value if the PGPORT environment variable is set)
-u *User name*	Specifies a user name used to log in a database. Default value: postgres
-p *Password*	Specifies a password used to log in a database. Default value: None
-t *Table name*	Specifies the name of a monitoring table to be created on the database. Default value: psqlwatch
-i *Monitoring Interval*	Specifies an interval of monitoring a database in seconds (between 1 and 10000). Default value: 60
-c *Retry count*	Specifies retry count to be made (between 1 and 10000) when an error is detected during database monitoring Default value: 2
-r *Response Wait Time*	Specifies a response wait time (between 1 and 10000) of a database monitoring Default value:120
--stop	Stops the monitoring command
Stop Wait Time	Specifies time to wait for the monitoring command to stop normally (between 1 and 10000) y. Default value: 60
--pause	Temporary interrupts monitoring.
--continue	Resumes monitoring.
--disp	Displays a process ID (pid) of the monitoring command identifier managed by the monitoring command. Do not use this in ordinary use. (See note 3).
--del	Deletes information about the monitoring command identifier managed by the monitoring command. Do not use this in ordinary use. (See note 3).

Remarks

About identifier
You need to specify an identifier that is unique on the system to control the monitoring command. You cannot use an identifier that is already used by a monitoring command for another monitoring command. Specify an identifier using alphanumeric characters. An identifier should be up to 255 bytes. An identifier is case sensitive.
An identifier must be written as the first parameter of a monitoring command.

About the –a parameter
Specify an FIP if access to PostgreSQL is made by an FIP. You do not need to specify this parameter for the localhost connection.

Example



The IP address specified by this parameter needs to be authorized for access by the pg_hba.conf file.

About the -n parameter
Specify this parameter if the port number is specified when PostgreSQL starts up. If no port number is specified when PostgreSQL starts up, 5432 is typically used.

About the -u –p parameters
Specify the user name and password that have been set up by PostgreSQL (they are not the user name and password on Linux) for these parameters. These parameters are specified when monitoring databases which limit the accessible users in pg_hba.conf.

About the –t parameter
A table of values specified with the –t parameter is created on a database specified by the –d parameter. Make sure the –t parameter table name and the names of table used in the operation do not overlap.

About the -d,-a,-u,-p, and -t parameters
The maximum length of a character sting for these parameters is 255 bytes. Effective length varies by each parameter but this command does not check the effective length. Effective length is determined by the PostgreSQL specifications.

Monitoring method

This monitoring command performs the following monitoring.
By creating a monitoring table on the database and issuing an SQL statement, the command allows reading and writing of a maximum of 5-digit numerical data.
As a result of monitoring, it is considered an error if:

(1) No response is returned for database access or issue of SQL statement within a response wait time (-r parameter value.)
(2) Error is returned for database access or issue of SQL statement.
(3) The written data and read data do not match.

The SQL statement to be used is create/drop/insert/update/select.

Note 1

This monitoring command monitors PostgreSQL using the libpq library of PostgreSQL.
If you cannot run this watching command (i.e., "Fail to start Group xxx application (watcing ID=xxx)" is displayed on ExpressCluster Manager Alert View), set the application library path to the place where PostgreSQL libpq library exists.
This watching command requires the PostgreSQL library of the following versions.

clp_psql72mon	libpq.so.2
clp_psql73mon	libpq.so.3
clp_psql80mon	libpq.so.4

Example

```
export LD_LIBRARY_PATH=/usr/local/pgsql/lib
```

A library name appears in a standard error output of a server as: "clp_psql72mon: error while loading shared libraries: libpq.so.2.0: cannot open shared object file: No such file or directory." If there is no library for a given file name, soft-link to the entity file of PostgreSQL library.
Example

```
ln –s libpq.so.2.2 libpq.so.2.0
```

Note 2

If a specified parameter value is different from a value in the PostgreSQL environment, a message with error information will be displayed on the alert view of ExpressCluster manager. In such case, check the environment.

Note 3

If this monitoring command process is stopped using a kill command during monitoring, a monitoring command of the same identifier may not start because the management information is not successfully initialized. In such case, if you run the "clp_psql*xx*mon identifier --disp" command, the process ID which corresponds to the specified identifier will be displayed. Check the run file of the process ID. If it is in a process other than the monitoring command, delete the management information using the "clp_psqlxxmon identifier --del" command.
Never specify a normally operating identifier and run the "clp_psqlxxmon identifier --del" command because the monitoring command will malfunction.

Note 4	Client authentication: The following authentication methods can be specified in pg_hba.conf file. These methods have been confirmed proper behaviors in this watching command. trust,md5,password
Example of Command Usage	[start.sh] export LD_LIBRARY_PATH=/usr/local/pgsql/lib clp_psql72mon psqlwatch –d *database name* [stop.sh] clp_psql72mon psqlwatch --stop This command starts up from an EXEC resource.

clp_mysql323mon **clp_mysql40mon** **clp_mysql41mon**	Monitors MySQL 3.23 Monitors MySQL 4.0 Monitors MySQL4.1

Command line

Start Monitoring

clp_mysql323mon *identifier* -d D*atabase Name*
[-a *IP Address*] [-n *Port Number*]
[-u *User Name*] [-p *Password*] [-t *Table Name*]
[-i *Monitoring Interval* [-c *Retry Count*] [-r *Response Wait Time*]

Stop Monitoring

clp_mysql323mon *Identifier* --stop [*Stop Wait Time*]

Interrupt Monitoring

clp_mysql323mon *Identifier* --pause

Resume Monitoring

clp_mysql323mon *Identifier* --continue

Display Information

clp_mysql323mon *identifier* --disp

Delete Information

clp_mysql323mon *Identifier* --del

The command syntax of clp_mysql40mon and clp_mysql41mon are the same as above.

Description

When a database name is specified, this command monitors MySQL per database. This command stops when a MySQL error is detected.
The command also stops, interrupts and resumes monitoring.
Use a root privileged console to specify --pause/--continue/--disp/--del.

Option

Identifier
Specifies an identifier to uniquely identify the monitoring command.
You must set an identifier.

-d *Database name*
Specifies a name of a database to be monitored.
You must set this option.

-a *IP address*
Specifies an IP address used to access MySQL from a MySQL client.
Default value: localhost

-n *Port number*	Specifies a MySQL port number. Default value: 3306
-u *User name*	Specifies a user name used to log in a database. Default value: None
-p *Password*	Specifies a password used to log in a database. Default value: None
-t *Table name*	Specifies a name of monitoring table to be created on the database. Default value: mysqlwatch
-i *Monitoring Interval*	Specifies an interval of monitoring a database in seconds (between 1 and 10000). Default value: 60
-c *Retry count*	Specifies retry count to be made (between 1 and 10000) when an error is detected during database monitoring Default value: 2
-r *Response Wait Time*	Specifies a response wait time (between 1 and 10000) of a database monitoring Default value:120
--stop	Stops the monitoring command
Stop Wait Time	Specifies time to wait for the monitoring command to stop normally (between 1 and 10000) y. Default value: 60
--pause	Temporary interrupts monitoring.
--continue	Resumes monitoring.
--disp	Displays a process ID (pid) of the monitoring command identifier managed by the monitoring command. Do not use this in ordinary use. (See note 3).
--del	Deletes information about the monitoring command identifier managed by the monitoring command. Do not use this in ordinary use. (See note 3).

Remarks

About identifier

You need to specify an identifier that is unique on the system to control the monitoring command. You cannot use an identifier that is already used by a monitoring command for another monitoring command. Specify an identifier using alphanumeric characters. An identifier should be up to 255 bytes. An identifier

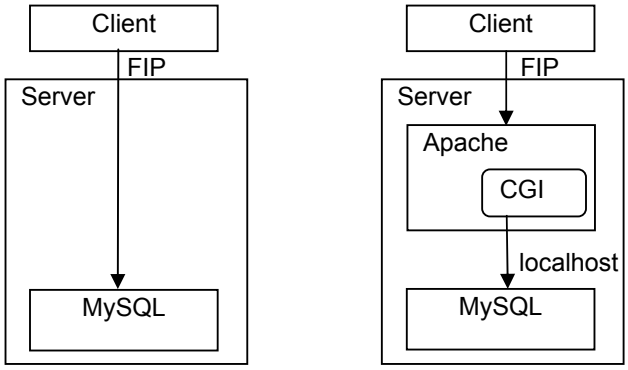
is case sensitive.
An identifier must be written as the first parameter of a monitoring command.

About the –a parameter
If this parameter value is localhost (not including when 127.0.0.1 is specified), no TCP/IP communication is not used to access MySQL. Because of this, use a script to start up this command to configure the name of a file for socket specified by my.cnf such as
export MYSQL_UNIX_PORT=/var/lib/mysql/mysql.sock

If you specify an IP address (including 127.0.0.1) by this parameter, the specified IP address needs to be authorized for ccess by a grant statement.

Specify an FIP when MySQL is accessed by an FIP. For localhost connection, specify either none or 127.0.0.1.

Example



About the –n parameter
Specify this parameter if a port number connected to MySQL is not the default value (3306) of MySQL.

About the –t parameter
A table of values specified with the –t parameter is created on a database specified by the –d parameter. Make sure the –t parameter table name and the name of the table used in the operation do not overlap.

About the -d,-a,-u,-p, and -t parameters
The maximum length of a character sting for these parameters is 255 bytes. Effective length varies by each parameter but this command does not check the effective length. Effective length is determined by the MySQL specifications.

Monitoring method

This monitoring command performs the following monitoring.
By creating a monitoring table on the database and issuing an SQL statement, the command allows reading and writing of a maximum of 5-digit numerical data.
As a result of monitoring, it is considered an error if:
(1) No response is returned for database access or issue of SQL statement within a response wait time (-r parameter value.)

(2) Error is returned for database access or issue of SQL statement.
(3) The written data and read data do not match.
The SQL statement to be used is create/drop/insert/update/select.

Note 1

This monitoring command monitors MySQL using libmysqlclient library of MySQL.
If you cannot run this command ("Failed to start an application of the group xxx (Monitoring ID=xxx)" appears on the alert view of ExpressCluster manager), check that libmysqlclient.so.xx exists in the install directory of the MySQL library.
A library name appears in a standard error output of a server as: "clp_mysql323mon: error while loading shared libraries: libmysqlclient.so.10: cannot open shared object file: No such file or directory."
This watching command requires the MySQL library of the following versions.

clp_mysql323mon	libmysqlclient.so.10
clp_mysql40mon	libmysqlclient.so.12
clp_mysql41mon	libmysqlclient.so.14

For MySQL version 4.0, if "MySQL-shared-compat" rpm is installed on the server, the above two libraries are also installed, therefore, you can monitor MySQL4.0 database in clp_mysql323mon.

Note 2

If a specified parameter value is different from a value in the MySQL environment for monitoring, a message with error information will be displayed on the alert view of the ExpressCluster manager. In such case, check the environment.

Note 3

If this monitoring command process is stopped using a kill command during monitoring, a monitoring command of the same identifier may not start because the management information is not correctly initialized. In such case, if you run the "clp_mysqlxxmon identifier --disp" command, the process ID which corresponds to the specified identifier will be displayed. Check the run file of the process ID. If it is in process other than the monitoring command, delete the management information using the "clp_mysqlxxmon identifier --del" command.
Never specify a normally operating identifier and run the "clp_mysqlxxmon identifietr --del" command because the monitoring command will malfunction.

Example of Command Usage

[start.sh]
export MYSQL_UNIX_PORT=/var/lib/mysql/mysql.sock
clp_mysql323mon mysqlwatch –d *database name*
[stop.sh]
clp_mysql323mon mysqlwatch --stop

This command starts up from an EXEC resource.

clp_sybmon	Monitors Sybase Adaptive Server Enterprise (hereinafter referred to as ASE)

Command line

Start Monitoring

clp_sybmon *Identifier* -d *Database Name* -s *Database Server Name* [-u *User Name*] [-p *Password*] [-t *Table Name*] [-i *Monitoring Interval*] [-c R*etry Count*] [-r *Response Wait Time*]

Stop Monitoring

clp_sybmon *Identifier* --stop [*Stop Wait Time*]

Interrupt Monitoring

clp_sybmon *Identifier* --pause

Resume Monitoring

clp_sybmon *Identifier* --continue

Display Information

clp_sybmon *Identifier* --disp

Delete Information

clp_sybmon *Identifier* --del

Description

This command specifies a database name and database server name, and monitors ASE per database. The command terminates when ASE error is detected.
The command also stops, interrupts and resumes monitoring.
Use a root authorization console to specify --pause/--continue/--disp/--del.

Option

Identifier	Specifies an identifier to uniquely identify the monitoring command. You must set this option.
-d *Database name*	Specifies the name of a database to be monitored. You must set this option.
-s *Database server name*	Specifies the name of a database server to be monitored. You must set this option.
-u *User name*	Specifies the user name to log in a database. Default value: sa
-p *Password*	Specifies the password to log in a database. Default value: None
-t *Table name*	Specifies the name of monitoring table to be created on a database. Default value: sybasemonitor

-i *Monitoring Interval*	Specifies an interval of monitoring database in seconds (between 1 and 10000). Default value: 60
-c *Retry count*	Specifies the retry count (between 1 and 10000) for when an error is detected during database monitoring. Default value: 2
-r *Response Wait Time*	Specifies a response wait time (between 1 and 10000) of a database monitoring. Default value: 120
--stop	Terminates the monitoring command.
Stop Wait Time	Specifies the time to wait for the monitoring command to terminate normally (between 1 and 10000). Default value: 60
--pause	Interrupts monitoring temporarily.
--continue	Resumes monitoring.
--disp	Displays a process ID (pid) of the monitoring command identifier managed by the monitoring command. Do not use this in ordinary cases. (See Note 3)
--del	Deletes information about the monitoring command identifier managed by the monitoring command. Do not use this in ordinary cases. (See Note 3)

Remarks

About identifier
You need to specify an identifier that is unique in the system to control the monitoring command. You cannot invoke the monitoring command by using the identifier already used.
Specify an identifier using alphanumeric characters up to 255 bytes. Case sensitive.
You need to write an identifier as the first parameter of a monitoring command.

About the -s parameter
Specify the database server name set at ASE installation for the -s parameter. The database server name can be checked in interfaces located directly under the ASE installation directory.

About the -u, -p parameters
Specify the user name and password that are set for ASE (they are not the user name and password on Linux) for these parameters. Connect to the database server using the “sa” user account with a blank password if the user name and password are not specified.

About the -t parameter
A table of values specified with the -t parameter is created on a database specified by the -d parameter. Make sure the -t parameter table name and the names of table used in the operation do not overlap.

About the -d, -s,-u,-p,-t parameters
The maximum length of a character string which can be specified for these parameters is 255 bytes. Effective length varies by each parameter but this command does not check the effective length. Effective length is determined by the ASE specifications.

Monitoring method

This monitoring command performs the following monitoring.
By creating a monitoring table on the database and issuing an SQL statement, the command allows reading and writing of a maximum of 5-digit numerical data (decimal).
As a result of monitoring, it is considered as an error if:
(1) No response is returned for database access or issuance of SQL statement within a response wait time (-r parameter value.)
(2) Error is returned for database access or issuance of SQL statement.
(3) The written data and read data do not match.
The SQL statement to be used is create/drop/insert/update/select.

Note 1

This monitoring command monitors ASE using the Open Client DB-Library/C of ASE.
If you cannot run this monitoring command (“Monitor XXX failed. (1:Process does not exist.(pid=XXX))” is displayed on EXPRESSCLUSTER Manager Alert View), set the application library path to the path where libsybdb.so exists.

Example:
```
export LD_LIBRARY_PATH=/opt/sybase/OCN-12_5/lib/
```
A library name appears in a standard error output of a server as: “clp_sybmon: error while loading shared libraries: libsybdb.so: cannot open shared object file: No such file or directory”.

Note 2

If a specified parameter value is different from the value in the ASE environment, a message with error information will be displayed on the Alert View of EXPRESSCLUSTER Manager. In such case, check the environment.

Note 3

If this monitoring command process is stopped using a kill command during monitoring, a monitoring command of the same identifier may not start because the management information is not successfully initialized. In such case, if you run the "clp_sybmon identifier --disp" command, the process ID which corresponds to the specified identifier will be displayed. Check the run file of the process ID using the ps command. If it is in a process other than the monitoring command, delete the management information using the "clp_sybmon identifier --del" command.
Never specify a normally operating identifier to run the "clp_sybmon identifier --del" command because the monitoring command will malfunction.

Example of Command Usage	[start.sh] export LD_LIBRARY_PATH=/opt/sybase/OCS-12_5/lib clp_sybmon sybwatch -d *database_name* -s d*atabase_ server_name* [stop.sh] clp_sybmon sybwatch --stop This command starts up from an EXEC resource.

8 Alert Messages

8.1 Messages produced by clp_db28mon

8.1.1 Messages Indicating Normal Operation

ID	Message	Explanation	Supplement
1	The clp_db28mon has started watching DB2UDB. [ID:xx]	clp_db28mon has started.	-
2	The clp_db28mon is going to watch database ' *xxx* '. [ID:xx]	clp_db28mon will start monitoring the database xxx.	When this message is not displayed immediately after the message shown in the previous column, an error may have occurred. In the case, an error message may be displayed after a while. Take actions according to the error message.
3	The clp_db28mon will stop watching DB2UDB database ' *xxx* '. [ID:xx]	clp_db28mon will stop.	-
7	The clp_db28mon is going to stop monitoring. [ID:xx]	clp_db28mon monitoring is interrupted.	This message is displayed when monitoring is interrupted by using the --pause command.
8	The clp_db28mon is going to monitor. [ID:xx]	clp_db28mon monitoring is resumed.	This message is displayed when monitoring is resumed by using the --continue command.
52	Trial Period of clp_db28mon is Till *nn*/*nn*/*nnnn* (mm/dd/yyyy).	Operation is under the trial license.	-

8.1.2 Messages Due to Setting Error

ID	Message	Explanation	Supplement
5	The clp_db28mon is not going to watch DB2UDB database ' *xxx* '. [ID:xx]	Monitoring is not performed due to a setting error, etc.	Take actions according to the message displayed immediately before this message.
11	Invalid parameter in the clp_db28mon. [ID:xx]	The parameter value of clp_db28mon is in invalid format.	Check the parameter value of the monitoring command.
12	'-d' parameter is not specified at the clp_db28mon command. [ID:xx]	The -d parameter is not specified for clp_db28mon.	Check the parameter of the monitoring command.
13	The monitor id is not specified at the clp_db28mon command.	No identifier is specified for clp_db28mon.	Check the parameter of the monitoring command.
14	The specified monitor id is already under use in other processes. [ID:xx]	clp_db28mon cannot newly be started since the same identifier is already running.	Check the parameter of the monitoring command.
15	The clp_db28mon has not performed end processing. [ID:xx]	The ending process could not be performed with the --stop parameter.	Check the parameter of the monitoring command.

ID	Message	Explanation	Supplement
21	Failed to connect to the database ' *xxx* '. [ID:xx]	There is an error with the database connection function of DB2.	The database name specified by the -d parameter may be incorrect. Take action according to the message of ID=0, displayed immediately before this message.
24	Specified user name ' *xxx* ' does not exist. The clp_db28mon will terminate. [ID:xx]	The user name specified by the -u parameter does not exist.	Check whether the specified user name exists.
25	The DB2 instance ' *xxx* ' has not been started. [ID:xx]	The instance specified by the -m parameter has not started.	Check whether the specified instance is started.
26	The code page of database ' *xxx* ' is not correct. [ID:xx]	The code page of the environmental which is running this monitoring command does not match the code page of a database.	Specify "export LANG=ja_JP.ujis," etc. after checking the database code page with "db2 get db cfg for database name."
51	The licence of clp_db28mon is not registered.	The license is not registered.	Register the license.
53	The licence of trial expired by *nn*/*nn*/*nnnn* (mm/dd/yyyy).	The trial term of the trial version license has expired.	-
55	The licence of trial is valid from *nn*/*nn*/*nnnn* (mm/dd/yyyy).	The start date of the trial license is yet to come.	-
56	The registration license overlaps.	The registered license key overlaps.	Register a different license key for each server.

8.1.3 Messages Produced when an Error is Detected in Database Monitoring

ID	Message	Explanation	Supplement
6	The clp_db28mon will terminate. [ID:xx]	An error was detected. The command terminates.	Take an action according to the message displayed immediately before this message.
21	Failed to connect to the database ' *xxx* '. [ID:xx]	There is an error with the database connection function of DB2.	Take actions according to the ID=0 message displayed immediately before this message.
31	The clp_db28mon has detected an error in DB2UDB database ' *xxx* ' (stall). [ID:xx]	No response is returned after executing DB2 monitoring.	Check if there is an error in the database system.
32	The clp_db28mon has detected an error in DB2UDB database ' *xxx* ' (data access error). [ID:xx]	When a DB2 database is read, the data read and the data written just before do not match.	Check if there is an error in the database system.
33	The clp_db28mon has detected an error in DB2UDB database ' *xxx* '. [ID:xx]	There was an error in DB2.	Take actions based on the message of ID=0 displayed immediately before this message.
34	Failed to execute SQL statement(*xxx*). [ID:xx]	An attempt to run an SQL statement failed.	Take actions based on the message of ID=0 displayed immediately before this message.

ID	Message	Explanation	Supplement
35	Error occured in DB2UDB API *xxx*. [ID:xx]	There was an error in API of DB2.	Take actions based on the message of ID=0 displayed immediately before this message.
0	SQL*nnnn* SQLSTATE=*nnnnn* xxxxxxxx	This is an error code produced by DB2.	Refer to the DB2 message manual.

8.1.4 Messages Due to a System Error

ID	Message	Explanation	Supplement
42	The clp_db28mon has detected system error (*xxx nn*). [ID:xx]	There is an error on the Linux system. xxx indicates a function name and nn indicates an error code.	Check the status of the system based on the error code.
54	Failed to check license of the clp_db28mon.	Checking the license information has failed.	The license management module of the ExpressCluster server may be old. Check whether there is any module update.

8.2 Messages produced by clp_ora9mon/clp_ora10mon

Messages logged by clp_ora9mon and clp_ora10mon are identical except for module names.

8.2.1 Messages Indicating Normal Operation

ID	Message	Explanation	Supplement
1	The clp_ora10mon has started watching Oracle. [ID:xx]	clp_ora10mon has started.	-
2	The clp_ora10mon is going to watch database ' *xxx* '. [ID:xx]	clp_ora10mon will start monitoring the database xxx.	When this message is not displayed immediately after the message shown in the previous column, an error may have occurred. In the case, an error message may be displayed after a while. Take actions according to the error message.
3	The clp_ora10mon will stop watching Oracle database ' *xxx* '. [ID:xx]	clp_ora10mon will stop.	-
7	The clp_ora10mon is going to stop monitoring. [ID:xx]	clp_ora10mon monitoring is interrupted.	This message is displayed when monitoring is interrupted by using the --pause command.
8	The clp_ora10mon is going to monitor. [ID:xx]	clp_ora10mon monitoring is resumed.	This message is displayed when monitoring is resumed by using the --continue command.
52	Trial Period of clp_ora10mon is Till *nn*/*nn*/*nnnn* (mm/dd/yyyy).	Operation is under the trial license.	-

8.2.2 Messages Due to Setting Error

ID	Message	Explanation	Supplement
5	The clp_ora10mon is not going to watch Oracle database ' *xxx* '. [ID:xx]	Monitoring is not performed due to a setting error, etc.	Take an action according to the message displayed immediately before this message.
11	Invalid parameter in the clp_ora10mon. [ID:xx]	The parameter value of clp_ora10mon is in invalid format.	Check the parameter value of the monitoring command.
12	'-d' parameter is not specified at the clp_ora10mon command. [ID:xx]	The -d parameter is not specified for clp_ora10mon.	Check the parameter of the monitoring command.
13	The monitor id is not specified at the clp_ora10mon command.	No identifier is specified for clp_ora10mon	Check the parameter of the monitoring command.
14	The specified monitor id is already under use in other processes. [ID:xx]	clp_ora10mon cannot newly be started since the same identifier is already running.	Check the parameter of the monitoring command.

ID	Message	Explanation	Supplement
15	The clp_ora10mon has not performed end processing. [ID:xx]	The ending process could not be performed with the --stop parameter.	Check the parameter of the monitoring command.
22	Failed to connect with the server *xxx*. [ID:xx]	There is an error with the server connection function of Oracle.	The connection string specified by the –d parameter may be incorrect. Take action according to the message of ID=0, displayed immediately before this message.
23	Could not start the user session %s. [ID:xx]	There is an error in the session start function of Oracle.	Check whether the specified user name/password is correct.
51	The licence of clp_ora10mon is not registered.	The license is not registered.	Register the license.
53	The licence of trial expired by *nn*/*nn*/*nnnn* (mm/dd/yyyy).	The trial term of the trial version license has expired.	-
55	The licence of trial is valid from *nn*/*nn*/*nnnn* (mm/dd/yyyy).	The start date of the trial license is yet to come.	-
56	The registration license overlaps.	The registered license key overlaps.	Register a different license key for each server.

8.2.3 Messages Produced when an Error is Detected in Database Monitoring

ID	Message	Explanation	Supplement
6	The clp_ora10mon will terminate. [ID:xx]	An error was detected. The command terminates.	Take actions according to the message displayed immediately before this message.
22	Failed to connect with the server *xxx*. [ID:xx]	There is an error with the database connection function of Oracle.	Take actions according to the ID=0 message, displayed immediately before this message.
31	The clp_ora10mon has detected an error in Oracle database ' *xxx* ' (stall). [ID:xx]	No response is returned after executing Oracle monitoring.	Check if there is an error in the database system.
32	The clp_ora10mon has detected an error in Oracle database ' *xxx* ' (data access error). [ID:xx]	When Oracle database is read, the data read and the data written just before do not match.	Check if there is an error in the database system.
33	The clp_ora10mon has detected an error in Oracle database ' *xxx* '. [ID:xx]	There was an error in Oracle.	Take actions based on the message of ID=0, displayed immediately before this message.
34	Failed to execute SQL statement(*xxx*). [ID:xx]	An attempt to run an SQL statement failed.	Take actions based on the message of ID=0, displayed immediately before this message.
35	Error occured in Oracle API *xxx*. [ID:xx]	There was an error in API of Oracle.	Take actions based on the message of ID=0, displayed immediately before this message.
0	ORA- *nnnnn*:xxxxxxxx	This is an error code produced by Oracle.	Refer to the Oracle message manual.

8.2.4 Messages Due to a System Error

ID	Message	Explanation	Supplement
42	The clp_ora10mon has detected system error (*xxx nn*). [ID:xx]	There was an error on the Linux system. xxx indicates a function name and nn indicates an error code.	Check the state of the system based on the error code.
54	Failed to check license of the clp_ora10mon. [ID:xx]	Checking the license information has failed.	The license management module of ExpressCluster server may be old. Check whether there is any module update.

8.3 Messages produced by clp_psql72mon/clp_psql73mon/ clp_psql80mon

By the message which clp_psql72mon, and clp_psql73mon and clp_psql80mon output, although only the portions of a module name differ, the same contents are outputted about the other portion.

8.3.1 Messages Indicating Normal Operation

ID	Message	Explanation	Supplement
1	The clp_psql72mon has started watching PostgreSQL. [ID:xx]	clp_psql72mon has started.	-
2	The clp_psql72mon is going to watch database ' *xxx* '. [ID:xx]	clp_psql72mon will start monitoring the database xxx.	When this message is not displayed immediately after the message indicated above, an error may have occurred. In the case, an error message may be displayed after a while. Take actions based on the error message.
3	The clp_psql72mon will stop watching PostgreSQL database ' *xxx* '. [ID:xx]	clp_psql72mon monitoring is interrupted.	-
7	The clp_psql72mon is going to stop monitoring. [ID:xx]	clp_psql72mon monitoring is interupted.	This message is displayed when monitoring is interrupted by using the --pause command.
8	The clp_psql72mon is going to monitor. [ID:xx]	clp_psql72mon monitoring is resumed.	This message is displayed when monitoring is resumed by using the --continue command.
52	Trial Period of clp_psql72mon is Till *nn*/*nn*/*nnnn* (mm/dd/yyyy).	Operation is under the trial license.	-

8.3.2 Messages Due to Setting Error

ID	Message	Explanation	Supplement
5	The clp_psql72mon is not going to watch PostgreSQL database ' *xxx* '. [ID:xx]	Monitoring is not performed due to a setting error, etc.	Take actions according to the message displayed immediately before this message.
11	Invalid parameter in the clp_psql72mon. [ID:xx]	The parameter value of clp_psql72mon is in invalid format.	Check the parameter value of the monitoring command.
12	'-d' parameter is not specified at the clp_psql72mon command. [ID:xx]	The -d parameter is not specified for clp_psql72mon	Check the parameter of the monitoring command.
13	The monitor id is not specified at the clp_psql72mon command.	No identifier is specified for clp_ psql72mon	Check the parameter of the monitoring command.
14	The specified monitor id is already under use in other processes. [ID:xx]	clp_psql72mon cannot newly be started since the same identifier is already running.	Check the parameter of the monitoring command.

ID	Message	Explanation	Supplement
15	The clp_psql72mon has not performed end processing. [ID:xx]	The ending process could not be performed with the --stop parameter.	Check the parameter of the monitoring command.
21	Failed to connect to the database ' *xxx* '. [ID:xx]	There is an error with the database connection functions of PostgreSQL.	The database name specified by the -d parameter, user name or password may be incorrect. Take actions according to the message of ID=0, displayed immediately before this message.
51	The licence of clp_psql72mon is not registered.	The license is not registered.	Register the license.
53	The licence of trial expired by *nn*/*nn*/*nnnn* (mm/dd/yyyy).	The trial term of the trial version license has expired.	-
55	The licence of trial is valid from *nn*/*nn*/*nnnn* (mm/dd/yyyy).	The start date of the trial license is yet to come.	-
56	The registration license overlaps.	The registered license key overlaps.	Register a different license key for each server.

8.3.3 Messages Produced when an Error is Detected in Database Monitoring

ID	Message	Explanation	Supplement
6	The clp_psql72mon will terminate. [ID:xx]	An error was detected. The command terminates.	Take action according to the message displayed immediately before.
21	Failed to connect to the database ' *xxx* '. [ID:xx]	There is an error with the database connection function of PostgreSQL.	Take an action according to the ID=0 message, displayed immediately before this message.
31	The clp_psql72mon has detected an error in PostgreSQL database ' *xxx* ' (stall). [ID:xx]	No response is returned after executing PostgreSQL monitoring.	Check if there is an error in the database system.
32	The clp_psql72mon has detected an error in PostgreSQL database ' *xxx* ' (data access error). [ID:xx]	When a PostgreSQL database is read, the data read and the data written just before do not match.	Check if there is an error in the database system.
33	The clp_psql72mon has detected an error in PostgreSQL database ' *xxx* '. [ID:xx]	There was an error in PostgreSQL.	Take actions based on the message of ID=0, displayed immediately before this message.
34	Failed to execute SQL statement(*xxx*). [ID:xx]	An attempt to run an SQL statement failed.	Take actions based on the message of ID=0, displayed immediately before this message.
35	Error occured in PostgreSQL API *xxx*. [ID:xx]	There was an error in API of PostgreSQL.	Take actions based on the message of ID=0, displayed immediately before this message.
0	*xxxxxxxx*	This is an error code produced by PostgreSQL.	Refer to the PostgreSQL message manual.

8.3.4 Messages Due to a System Error

ID	Message	Explanation	Supplement
42	The clp_psql72mon has detected system error (*xxx nn*). [ID:xx]	There was an error on the Linux system. xxx indicates a function name and nn indicates an error code.	Check the state of the system based on the error code.
54	Failed to check license of the clp_psql72mon.	Checking the license information has failed.	The license management module of the ExpressCluster server may be old. Check whether there is any module update.

8.4 Messages produced by clp_mysql323mon/clp_mysql40mon/ clp_mysql41mon

Messages output by clp_mysql323mon, clp_mysql40mon, or clp_mysql41mon are identical except for module names.

8.4.1 Messages Indicating Normal Operation

ID	Message	Explanation	Supplement
1	The clp_mysql41mon has started watching MySQL. [ID:xx]	clp_mysql41mon has started.	-
2	The clp_mysql41mon is going to watch database ' *xxx* '. [ID:xx]	clp_mysql41mon will start monitoring the database xxx.	When this message is not displayed immediately after the message indicated above, an error may have occurred. In such case, the error message may be displayed after a while. Take actions based on the error message.
3	The clp_mysql41mon will stop watching MySQL database ' *xxx* '. [ID:xx]	clp_mysql41mon will stop.	-
7	The clp_mysql41mon is going to stop monitoring. [ID:xx]	clp_mysql41mon monitoring is interrupted.	This message is displayed when monitoring is interrupted by using the --pause command.
8	The clp_mysql41mon is going to monitor. [ID:xx]	clp_mysql41mon monitoring is resumed.	This message is displayed when monitoring is resumed by using the --continue command.
52	Trial Period of clp_mysql41mon is Till *nn*/*nn*/*nnnn* (mm/dd/yyyy).	Operation is under the trial license.	-

8.4.2 Messages Due to Setting Error

ID	Message	Explanation	Supplement
5	The clp_mysql41mon is not going to watch MySQL database ' *xxx* '. [ID:xx]	Monitoring processing is not performed due to a setting error, etc.	Take actions according to the message displayed immediately before this message.
11	Invalid parameter in the clp_mysql41mon. [ID:xx]	The parameter value of clp_mysql41mon is in invalid format.	Check the parameter of the monitoring command.
12	'-d' parameter is not specified at the clp_mysql41mon command. [ID:xx]	The -d parameter is not specified for clp_mysql41mon	Check the parameter of the monitoring command.
13	The monitor id is not specified at the clp_mysql41mon command.	No identifier is specified for clp_mysql41mon.	Check the parameter of the monitoring command.

ID	Message	Explanation	Supplement
14	The specified monitor id is already under use in other processes. [ID:xx]	clp_mysql41mon cannot newly be started since the same identifier is already running.	Check the parameter of the monitoring command.
15	The clp_mysql41mon has not performed end processing. [ID:xx]	The ending process could not be performed with the --stop parameter.	Check the parameter of the monitoring command.
21	Failed to connect to the database ' *xxx* '. [ID:xx]	There is an error with the database connection function of MySQL.	The database name specified by the -d parameter, user name or password may be incorrect. Take actions according to the message of ID=0, displayed immediately before this message.
51	The licence of clp_mysql41mon is not registered. [ID:xx]	The license is not registered.	Register the license.
53	The licence of trial expired by *nn*/*nn*/*nnnn* (mm/dd/yyyy).	The trial term of the trial version license has expired.	-
55	The licence of trial is valid from *nn*/*nn*/*nnnn* (mm/dd/yyyy).	The start date of the trial license is yet to come.	-
56	The registration license overlaps.	The registered license key overlaps.	Register a different license key for each server.

8.4.3 Messages Produced when an Error is Detected in Database Monitoring

ID	Message	Explanation	Supplement
6	The clp_mysql41mon will terminate. [ID:xx]	An error was detected. The command stops.	Take actions according to the message displayed immediately before this message.
21	Failed to connect to the database ' *xxx* '. [ID:xx]	There is an error with the database connection function of MySQL.	Take an action according to the ID=0 message, displayed immediately before this message.
31	The clp_mysql41mon has detected an error in MySQL database ' *xxx* ' (stall). [ID:xx]	No response is returned after executing MySQL monitoring.	Check if there is an error in the database system.
32	The clp_mysql41mon has detected an error in MySQL database ' *xxx* ' (data access error). [ID:xx]	When a MySQL database is read, the data read and the data written just before do not match.	Check if there is an error in the database system.
33	The clp_mysql41mon has detected an error in MySQL database ' *xxx* '. [ID:xx]	There was an error in MySQL.	Take actions based on the message of ID=0, displayed immediately before this message.
34	Failed to execute SQL statement(*xxx*). [ID:xx]	An attempt to run an SQL statement failed.	Take actions based on the message of ID=0, displayed immediately before this message.
35	Error occured in MySQL API *xxx*. [ID:xx]	There was an error in API of MySQL.	Take actions based on the message of ID=0 displayed immediately before this message.

ID	Message	Explanation	Supplement
0	*xxxxxxxx*	This is an error code produced by MySQL.	Refer to the MySQL message manual.

8.4.4 Messages Due to a System Error

ID	Message	Explanation	Supplement
42	The clp_mysql41mon has detected system error (*xxx nn*). [ID:xx]	There was an error on the Linux system. xxx indicates a function name and nn indicates an error code.	Check the state of the system based on an error code.
54	Failed to check license of the clp_mysql41mon.	Checking the license information has failed.	The license management module of the ExpressCluster server may be old. Check whether there is any module update.

8.5 Messages produced by clp_sybmon

8.5.1 Messages Indicating Normal Operation

ID	Message	Explanation	Supplement
1	The clp_sybmon has started watching ASE. [ID:xx]	clp_sybmon has started.	-
2	The clp_sybmon is going to watch database ' *xxx* '. [ID:xx]	clp_sybmon will start monitoring the database xxx.	When this message is not displayed immediately after the message indicated above, an error may have occurred. In such case, the error message may be displayed after a while. Take actions based on the error message.
3	The clp_sybmon will stop watching ASE database ' *xxx* '. [ID:xx]	clp_sybmon will stop.	-
7	The clp_sybmon is going to stop monitoring. [ID:xx]	clp_sybmon monitoring is interrupted.	This message is displayed when monitoring is interrupted by using the --pause command.
8	The clp_sybmon is going to monitor. [ID:xx]	clp_sybmon monitoring is resumed.	This message is displayed when monitoring is resumed by using the --continue command.
52	Trial Period of clp_sybmon is Till *nn*/*nn*/*nnnn* (mm/dd/yyyy).	Operation is under the trial license.	-

8.5.2 Messages Due to Setting Error

ID	Message	Explanation	Supplement
5	The clp_sybmon is not going to watch ASE database ' *xxx* '. [ID:xx]	Monitoring processing is not performed due to a setting error, etc.	Take actions according to the message displayed immediately before this message.
11	Invalid parameter in the clp_sybmon. [ID:xx]	The parameter value of clp_sybmon is in invalid format.	Check the parameter of the monitoring command.
12	'-d' parameter is not specified at the clp_sybmon command. [ID:xx]	The -d parameter is not specified for clp_sybmon	Check the parameter of the monitoring command.
13	The monitor id is not specified at the clp_sybmon command.	No identifier is specified for clp_sybmon.	Check the parameter of the monitoring command.
14	The specified monitor id is already under use in other processes. [ID:xx]	clp_sybmon cannot newly be started since the same identifier is already running.	Check the parameter of the monitoring command.
15	The clp_sybmon has not performed end processing. [ID:xx]	The ending process could not be performed with the --stop parameter.	Check the parameter of the monitoring command.
21	Failed to connect to the database ' *xxx* '. [ID:xx]	There is an error with the database connection function of ASE.	Take actions according to the message of ID=0, displayed im mediately before this message.
51	The licence of clp_sybmon is not registered.	The license is not registered.	Register the license.

ID	Message	Explanation	Supplement
53	The licence of trial expired by *nn*/*nn*/*nnnn* (mm/dd/yyyy).	The trial term of the trial version license has expired.	-
55	The licence of trial is valid from *nn*/*nn*/*nnnn* (mm/dd/yyyy).	The start date of the trial license is yet to come.	-
56	The registration license overlaps.	The registered license key overlaps.	Register a different license key for each server.

8.5.3 Messages Produced when an Error is Detected in Database Monitoring

ID	Message	Explanation	Supplement
6	The clp_sybmon will terminate. [ID:xx]	An error was detected. The command stops.	Take actions according to the message displayed immediately before this message.
22	Failed to connect to the database ' *xxx* '. [ID:xx]	There is an error with the database connection function of ASE.	Take an action according to the ID=0 message, displayed immediately before this message.
31	The clp_sybmon has detected an error in ASE database ' *xxx* ' (stall). [ID:xx]	No response is returned after executing ASE monitoring.	Check if there is an error in the database system.
32	The clp_sybmon has detected an error in ASE database ' *xxx* ' (data access error). [ID:xx]	When a ASE database is read, the data read and the data written just before do not match.	Check if there is an error in the database system.
33	The clp_sybmon has detected an error in ASE database ' *xxx* '. [ID:xx]	There was an error in ASE.	Take actions based on the message of ID=0, displayed immediately before this message.
34	Failed to execute SQL statement(*xxx*). [ID:xx]	An attempt to run an SQL statement failed.	Take actions based on the message of ID=0, displayed immediately before this message.
35	Error occured in ASE API *xxx*. [ID:xx]	There was an error in API of ASE.	Take actions based on the message of ID=0 displayed immediately before this message.

8.5.4 Messages Due to a System Error

ID	Message	Explanation	Supplement
42	The clp_sybmon has detected system error (*xxx nn*). [ID:xx]	There was an error on the Linux system. *xxx* indicates a function name and *nn* indicates an error code.	Check the state of the system based on an error code.
54	Failed to check license of the clp_sybmon.	Checking the license information has failed.	The license management module of the ExpressCluster server may be old. Check whether there is any module update.

9 Environment Construction using a Trekking Tool

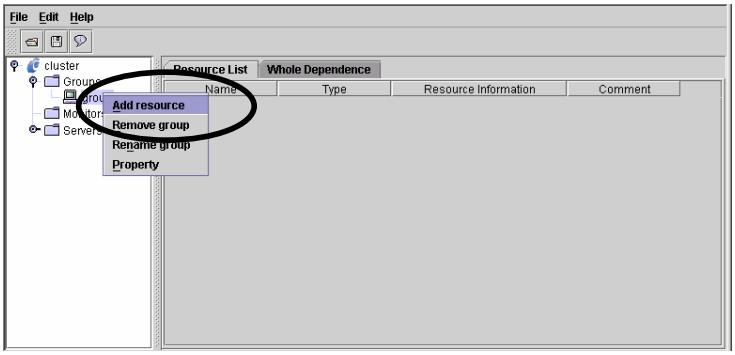
When you create a failover group of a database system by using trekking tool, follow the steps below.

1) Add a failover group for the application to be monitored.
2) Add a disk resource or an IP resource
3) Add an EXEC resource for starting the application to be monitored.
4) Make what you have done take effect in the ExpressCluster server and check that the application to be monitored operates successfully. (See "5.4.1 Checking Behavior of Target Application to Be Monitored")
5) Add an EXEC resource for starting the monitoring command.
6) Add a monitor resource to monitor an EXEC resource of the monitoring command.
7) Make what you have done take effect in the ExpressCluster server and check that the application to be monitored and the monitoring command operate normally. (See "5.4.2 Checking Monitoring Command Operation ")

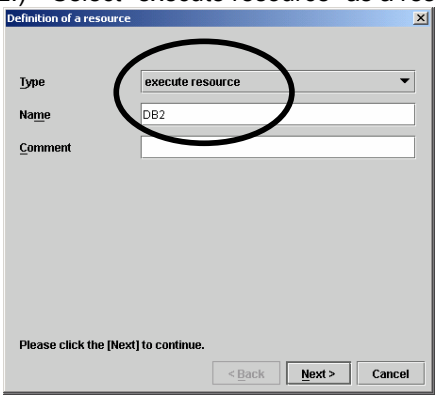
The following provides detailed information of Step 3), 5), and 6).

9.1 Adding an EXEC Resource for the Application to be Monitored

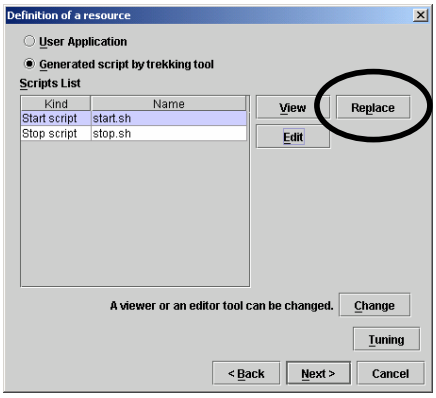
Run trekking tool and perform "Add Resource" in the failover group of an application to be monitored.



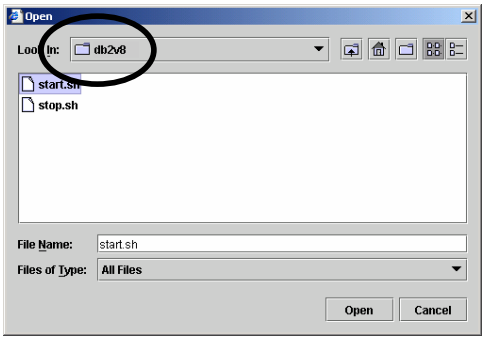
Add an EXEC resource for the monitored application. (In the following example, it is DB2.) Select "execute resource" as a resource type.



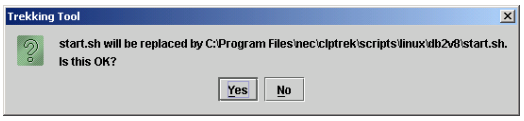
Click the [Replace] button and replace start.sh and stop.sh by script templates of the Agent.



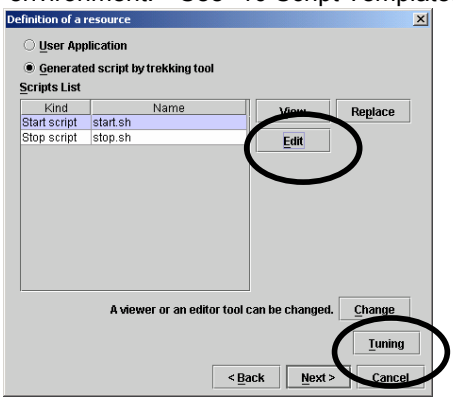
Specify and replace a script of the monitored application. Replace it by a template for monitoring application.



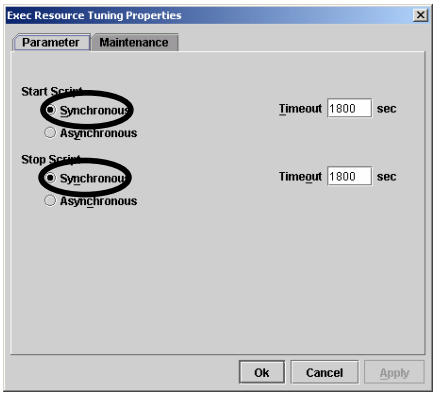
The following dialogue asking for confirmation appears. Check the original file to be replaced and then select "Yes."



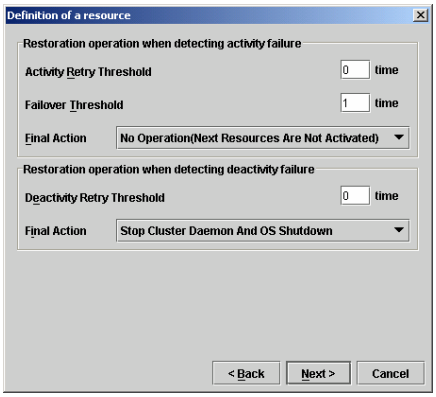
If you click the "Edit" button, the editor will open and you can modify the script according to the environment. See "10 Script Templates" for what to be modified.



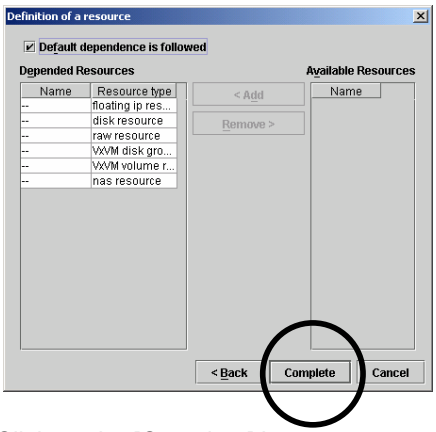
Click the "Tuning" button and check that both start script and stop script are "Synchronous" (The default value is Synchronous).



Configure items in the following window according to your environment. You do not have to change.



No changes are made in the following window. Check that a disk resource and an IP resource are displayed in the [Depended Resources] box.

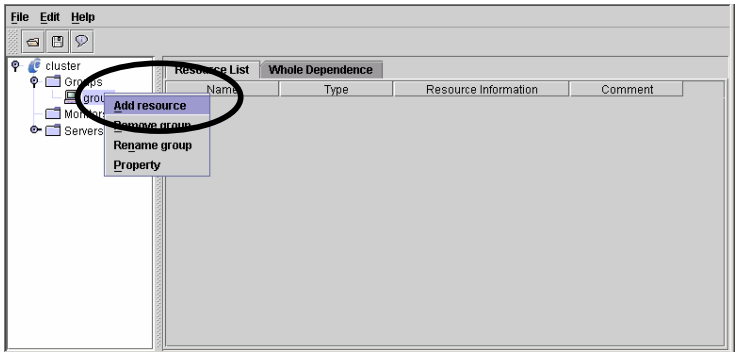


Click on the [Complete] button to create an EXEC resource for the monitored application.

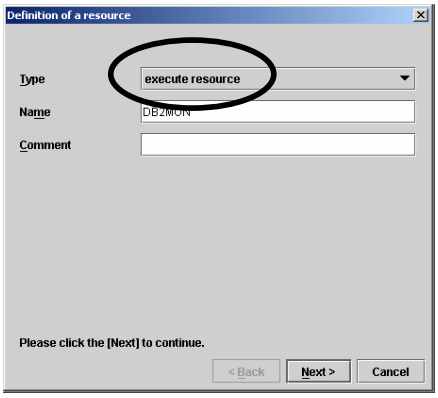
When the EXEC resource of the monitored application is added, make it take effect in the ExpressCluster server and check it works successfully.

9.2 Adding an EXEC Resource for a Monitoring Command

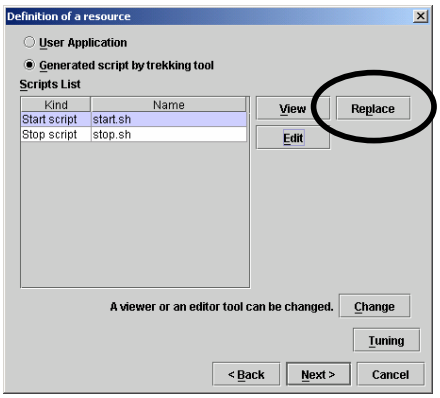
Add an EXEC resource for a monitoring command to a failover group of the monitored application.



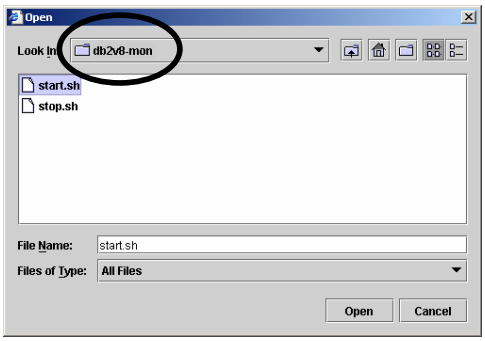
Select "execute resource" as a resource type. Specify a value that is different from the name selected previously for the [Name.]



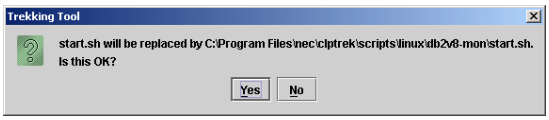
Click the [Replace] button and replace start.sh and stop.sh by script templates of the Agent.



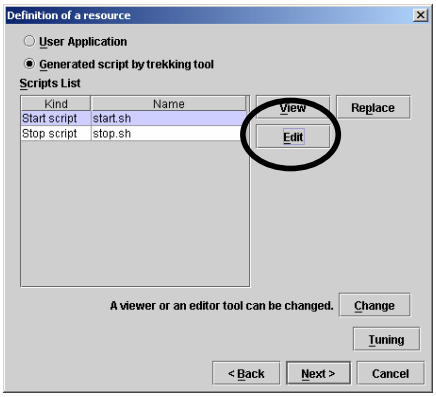
Specify and replace a script for the monitoring command. Replace it by a template for monitoring command.



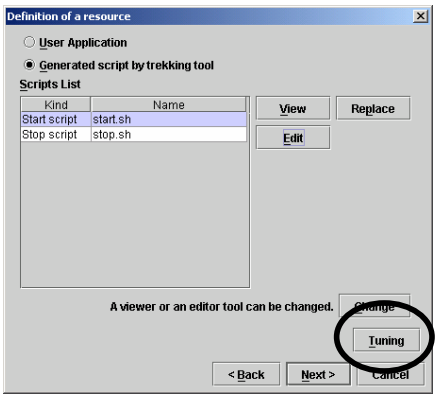
The following dialogue asking for confirmation appears. Check the original file to be replaced and then select "Yes."



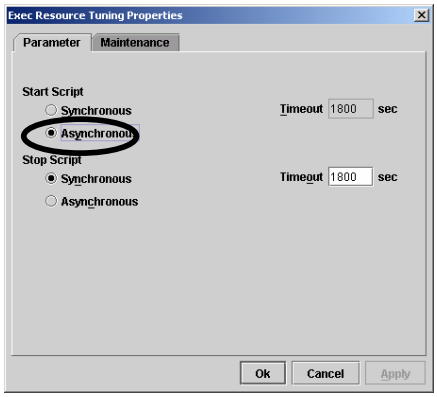
If you click the "Edit" button, the editor will open and you can modify the script according to the environment. See "10 Script Templates" for what to be modified.



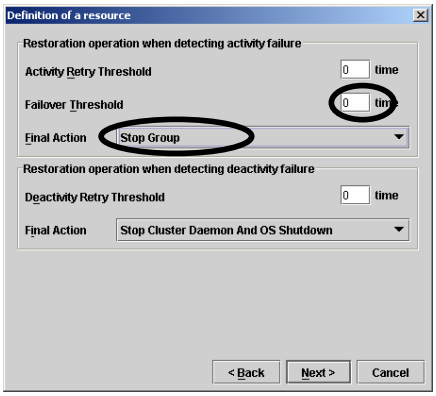
Click the "Tuning" button.



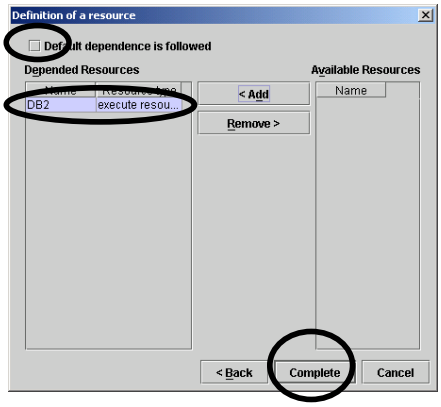
Select [Asynchronous] for the start script.



Set 0 for the [Failover Threshold] of [Restoration operation when detecting activity failure], and "Stop Group" for [Final Action.]



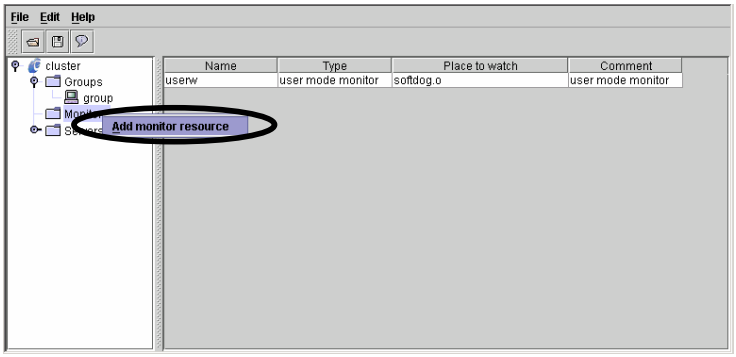
Clear the [Default dependence is followed] check box and add an EXEC resource of the monitored application to the depended resource.



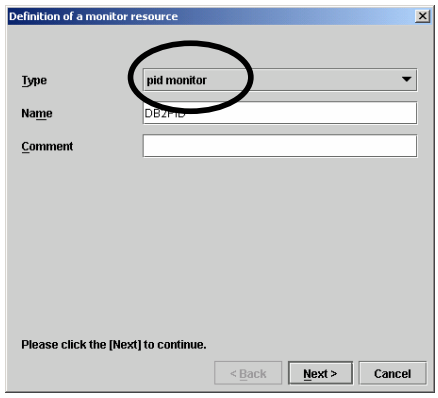
Click the [Complete] button and create an EXEC resource for the monitoring command.

9.3 Setting a Monitor Resource

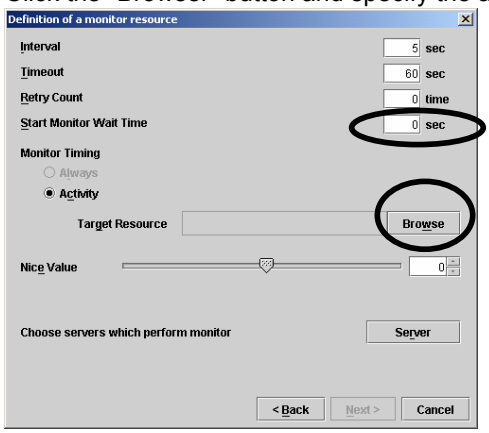
Add a monitor resource.



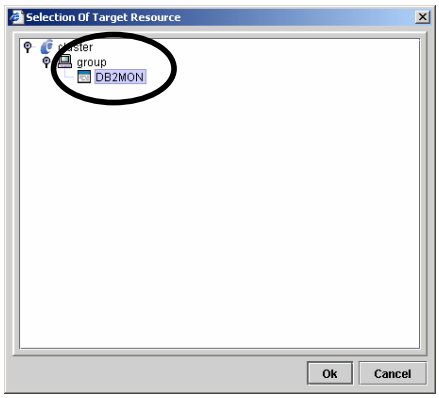
Select "pid monitor" as a monitoring type.



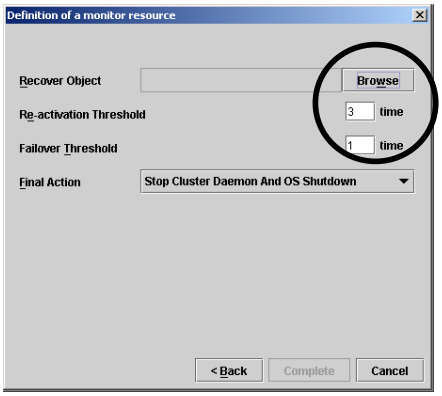
Click the "Browser" button and specify the application target for pid.



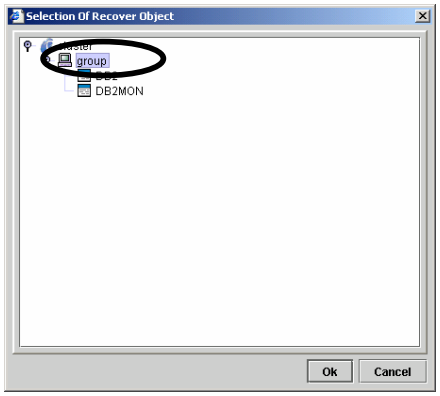
Select the EXEC resource of the monitoring command.



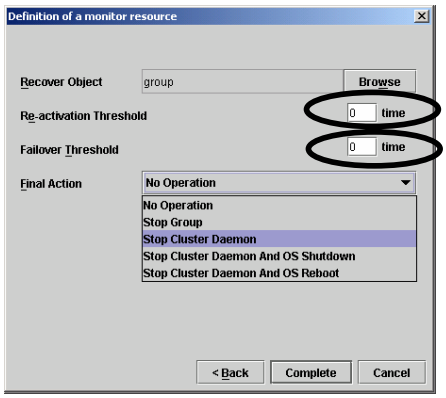
Click the [Browse] button and select a recovery target.



Select a failover group with a monitored application.



Select an action to be taken for [Final Action.] This selection determines the operation of ExpressCluster when the monitoring command detects an error in the monitored application.



*1

*2

***1**

When you detect an obstacle, please specify 0 as a [Re-active Threshold] to perform failover immediately. On the contrary, when you detect an obstacle, please specify values other than zero as a [Re-active Threshold] to revitalize a group.

When setting a [Re-active Threshold] as values other than zero, it is necessary to set up the value of [Start Monitor Wait Time] for a long time than the following time.

- Time after Agent starts until Agent is completed more unusually

Time until Agent is completed is influenced by the value of -i parameter (Monitoring Interval), -c parameter (Retry count), and -r parameter (Response Wait Time).

[Example] When abnormalities are notified from DBMS and Agent is completed

-i (Monitoring Interval) : 60 (seconds)
-c (Retry count) : 2 (Number of times)
-r (Response Wait Time) : 120 (seconds)

Near time until Agent is completed
60(seconds) * 2(Number of times) + Time concerning monitoring = 120 + α(seconds)

* Monitoring Interval * Retry count + Time concerning monitoring

With contents of the error, the time to the end of Agent changes a little.

Please refer to "7.2 Monitoring Chart" about operation of the Agent according to the kind of error.

***2**

If you want to perform failover without stopping a server as a cluster at the time of error detection, specify 1 in [Failover Threshold.] In this case, other failover groups continue operation in the server in which an error is detected. When an error is also detected in the failover destination server, failover occurs again and the failover group comes back.

If you want to perform failover by stopping a server as a cluster at the time of an error detection, specify 0 in [Failover Threshold.] In addition, specify "Stop Cluster Daemon", "Stop Cluster Daemon And OS Shutdown", or "Stop Cluster Daemon And OS Reboot" for [Final Action.] In this case, since the server which detected an error stops operating as a cluster, other failover group terminates or failover occurs.

It is recommended to specify 0 for "Failover Threshold" and "Stop Cluster Daemon"other failover group terminates or failover occurs.

It is recommended to specify 0 for "Failover Threshold" and "Stop Cluster Daemon" for "Final Action" given recovery work will be done when an error occurs.

See the construction guide of the ExpressCluster server about the details of the final operation.

If a setup is completed, click the [Complete] button to complete creation of monitor resource.

10 Script Templates

This product is shipped with script templates for trekking tool. Since the templates may be improved and modified from time to time, make sure to check if there is a newer version of the templates on the ExpressCluster home page. Use the newer one if there is any.

Where modification can be made are represented as comments in the script templates. See these comments for your modification.

10.1 For DB2

Two templates, start.sh and stop.sh, are provided for the DB2 scripts.
Modify and use them according to the operational environment and other requirements. Where modification is made are indicated in underlined boldface italics in the scripts in the following section.

10.1.1start.sh

```
#! /bin/sh
#****************************************
#*              start.sh                *
#****************************************

if [ "$CLP_EVENT" = "START" ]
then
 if [ "$CLP_DISK" = "SUCCESS" ]
 then
    echo "NORMAL1"
    if [ "$CLP_SERVER" = "HOME" ]
    then
       echo "NORMAL2"
    else
       echo "ON_OTHER1"
    fi
    echo "DB2 start"
#
# Modify the instance name to an appropriate value.
#
    su - db2inst1 -c "DB2INSTANCE=db2inst1;db2start"
 else
    echo "ERROR_DISK from START"
 fi
elif [ "$CLP_EVENT" = "FAILOVER" ]
then
 if [ "$CLP_DISK" = "SUCCESS" ]
 then
    echo "FAILOVER1"
    if [ "$CLP_SERVER" = "HOME" ]
```

```
    then
       echo "FAILOVER2"
    else
       echo "ON_OTHER2"
    fi
    echo "DB2 start"
#
# Modify the instance name to appropriate value.
#
    su - db2inst1 -c "DB2INSTANCE=db2inst1;db2start"
 else
    echo "ERROR_DISK from FAILOVER"
 fi
else
 echo "NO_CLP"
fi
echo "EXIT"
exit 0
```

10.1.2stop.sh

```
#! /bin/sh
#***************************************
#*               stop.sh               *
#***************************************

if [ "$CLP_EVENT" = "START" ]
then
 if [ "$CLP_DISK" = "SUCCESS" ]
 then
    echo "NORMAL1"
    if [ "$CLP_SERVER" = "HOME" ]
    then
       echo "NORMAL2"
    else
       echo "ON_OTHER1"
    fi
    echo "DB2 stop"
#
# Modify the instance name to an appropriate value.
#
    su - db2inst1 -c "db2stop force"

 else
    echo "ERROR_DISK from START"
 fi
elif [ "$CLP_EVENT" = "FAILOVER" ]
then
 if [ "$CLP_DISK" = "SUCCESS" ]
 then
    echo "FAILOVER1"
    if [ "$CLP_SERVER" = "HOME" ]
    then
       echo "FAILOVER2"
    else
       echo "ON_OTHER2"
    fi
    echo "DB2 stop"
#
# Modify the instance name to an appropriate value.
#
    su - db2inst1 -c "db2stop force"

 else
    echo "ERROR_DISK from FAILOVER"
 fi
else
 echo "NO_CLP"
fi
```

```
echo "EXIT"
exit 0
```

10.2 For DB2 Monitoring

Two templates, start.sh and stop.sh, are provided for the DB2 monitoring scripts. Modify and use them according to the operational environment and other requirements. Where modification is made are indicated in underlined boldface italics in the scripts in the following section.

10.2.1start.sh

```
#! /bin/sh
#***************************************
#*              start.sh               *
#***************************************

if [ "$CLP_EVENT" = "START" ]
then
 if [ "$CLP_DISK" = "SUCCESS" ]
 then
    echo "NORMAL1"
    if [ "$CLP_SERVER" = "HOME" ]
    then
       echo "NORMAL2"
    else
       echo "ON_OTHER1"
    fi
    echo "clp_db28mon start"
#
# Specify according to the database code page.
#
    export LANG=ja_JP.eucJP
#
# Modify the instance home name to an appropriate value.
#
    source /home/db2inst1/sqllib/db2profile
#
# Modify the interface name and/or database name to an appropriate value.
#
    clp_db28mon db2watch -d XXXX -m db2inst1
 else
    echo "ERROR_DISK from START"
 fi
elif [ "$CLP_EVENT" = "FAILOVER" ]
then
 if [ "$CLP_DISK" = "SUCCESS" ]
 then
    echo "FAILOVER1"
    if [ "$CLP_SERVER" = "HOME" ]
    then
       echo "FAILOVER2"
    else
```

```
      echo "ON_OTHER2"
   fi
   echo "clp_db28mon start"
#
# Specify according to the code page of a database.
#
   export LANG=ja_JP.eucJP
#
# Modify the instance home name to appropriate value.
#
   source /home/db2inst1/sqllib/db2profile
#
# Modify the instance name and database name to an appropriate value.
#
   clp_db28mon db2watch -d XXXX -m db2inst1
 else
   echo "ERROR_DISK from FAILOVER"
 fi
else
 echo "NO_CLP"
fi
echo "EXIT"
exit 0
```

10.2.2stop.sh

```
#! /bin/sh
#***************************************
#*               stop.sh               *
#***************************************

if [ "$CLP_EVENT" = "START" ]
then
 if [ "$CLP_DISK" = "SUCCESS" ]
 then
    echo "NORMAL1"
    if [ "$CLP_SERVER" = "HOME" ]
    then
       echo "NORMAL2"
    else
       echo "ON_OTHER1"
    fi
    echo "clp_db28mon stop"
#
# Modify the instance home name to an appropriate value.
#
    source /home/db2inst1/sqllib/db2profile
    clp_db28mon db2watch --stop
 else
    echo "ERROR_DISK from START"
 fi
elif [ "$CLP_EVENT" = "FAILOVER" ]
then
 if [ "$CLP_DISK" = "SUCCESS" ]
 then
    echo "FAILOVER1"
    if [ "$CLP_SERVER" = "HOME" ]
    then
       echo "FAILOVER2"
    else
       echo "ON_OTHER2"
    fi
    echo "clp_db28mon stop"
#
# Modify the instance home name to an appropriate value.
#
    source /home/db2inst1/sqllib/db2profile
    clp_db28mon db2watch --stop
 else
    echo "ERROR_DISK from FAILOVER"
 fi
else
 echo "NO_CLP"
fi
```

```
echo "EXIT"
exit 0
```

10.3 For Oracle9i

Two templates, start.sh and stop.sh, are provided for the Oracle9i scripts.
Modify and use them according to the operational environment and other requirements.
Where modification is made are indicated in underlined boldface italics in the scripts in the following section.

10.3.1start.sh

```
#! /bin/sh
#***************************************
#*              start.sh               *
#***************************************

if [ "$CLP_EVENT" = "START" ]
then
 if [ "$CLP_DISK" = "SUCCESS" ]
 then
    echo "NORMAL1"
    if [ "$CLP_SERVER" = "HOME" ]
    then
       echo "NORMAL2"
    else
       echo "ON_OTHER1"
    fi
    echo "Oracle9i start"
#
# Modify the user name and listener name to appropriate values.
#
    su - oracle -c "lsnrctl start LISTENER"
#
# Modify the user name, SID name, start script full path name to appropriate values.
#
    su - oracle -c "export ORACLE_SID=orcl;sqlplus /nolog @/XXXX/startup.sql"
 else
    echo "ERROR_DISK from START"
 fi
elif [ "$CLP_EVENT" = "FAILOVER" ]
then
 if [ "$CLP_DISK" = "SUCCESS" ]
 then
    echo "FAILOVER1"
    if [ "$CLP_SERVER" = "HOME" ]
    then
       echo "FAILOVER2"
    else
       echo "ON_OTHER2"
    fi
    echo "Oracle9i start"
#
```

```
 # Modify the user name, and listener name to appropriate values.
 #
    su - oracle -c "lsnrctl start LISTENER"
 #
 # Modify the user name, SID name, and start script full path name to appropriate
values.
 #
    su - oracle -c "export ORACLE_SID=orcl;sqlplus /nolog @/XXXX/startup.sql"
  else
    echo "ERROR_DISK from FAILOVER"
  fi
 else
  echo "NO_CLP"
 fi
 echo "EXIT"
 exit 0
```

10.3.2stop.sh

```
#! /bin/sh
#****************************************
#*               stop.sh                *
#****************************************

if [ "$CLP_EVENT" = "START" ]
then
 if [ "$CLP_DISK" = "SUCCESS" ]
 then
    echo "NORMAL1"
    if [ "$CLP_SERVER" = "HOME" ]
    then
       echo "NORMAL2"
    else
       echo "ON_OTHER1"
    fi
    echo "Oracle9i stop"
#
# Modify the user name, SID name, and stop script full path name to appropriate
values.
#
    su - oracle -c "export ORACLE_SID=orcl;sqlplus /nolog @/XXXX/shutdown.sql"
#
# Modify the user name and listener name to appropriate values.
#
    su - oracle -c "lsnrctl stop LISTENER"
 else
    echo "ERROR_DISK from START"
 fi
elif [ "$CLP_EVENT" = "FAILOVER" ]
then
 if [ "$CLP_DISK" = "SUCCESS" ]
 then
    echo "FAILOVER1"
    if [ "$CLP_SERVER" = "HOME" ]
    then
       echo "FAILOVER2"
    else
       echo "ON_OTHER2"
    fi
    echo "Oracle9i stop"
#
# Modify the user name, SID name, and stop script full path name to appropriate
values.
#
    su - oracle -c "export ORACLE_SID=orcl;sqlplus /nolog @/XXXX/shutdown.sql"
#
# Modify the user name and listener name to appropriate values.
```

```
#
   su - oracle -c "lsnrctl stop LISTENER"
 else
   echo "ERROR_DISK from FAILOVER"
 fi
else
 echo "NO_CLP"
fi
echo "EXIT"
exit 0
```

10.3.3 startup.sql/shutdown.sql

To start up Oracle9i, both startup.sql and shutdown.sql files used by sqlplus are required. You can choose any name and path for these files but make sure they match with the full path name specified with start.sh/stop.sh.

Write the following information in each file. In addition, there is no template for this information.

startup.sql

```
connect / as sysdba
startup pfile=initialization file name.
exit
```

shutdown.sql

```
connect / as sysdba
shutdown immediate
exit
```

10.4 For Oracle9i Monitoring

Two templates, start.sh and stop.sh, are provided for the Oracle9i monitoring scripts. Modify and use them according to the operational environment and other requirements. Where modification is made are indicated in underlined boldface italics in the scripts in the following section.

10.4.1start.sh

```
#! /bin/sh
#***************************************
#*              start.sh               *
#***************************************

if [ "$CLP_EVENT" = "START" ]
then
 if [ "$CLP_DISK" = "SUCCESS" ]
 then
    echo "NORMAL1"
    if [ "$CLP_SERVER" = "HOME" ]
    then
       echo "NORMAL2"
    else
       echo "ON_OTHER1"
    fi
    echo "clp_ora9mon start"
#
# Modify ORACLE_HOME to an appropriate value.
#
    export ORACLE_HOME=/opt/oracle/product/9.2.0
    export LD_LIBRARY_PATH=$ORACLE_HOME/lib
#
# Modify the database name to an appropriate value.
#
    clp_ora9mon orawatch -d XXXX
 else
    echo "ERROR_DISK from START"
 fi
elif [ "$CLP_EVENT" = "FAILOVER" ]
then
 if [ "$CLP_DISK" = "SUCCESS" ]
 then
    echo "FAILOVER1"
    if [ "$CLP_SERVER" = "HOME" ]
    then
       echo "FAILOVER2"
    else
       echo "ON_OTHER2"
    fi
    echo "clp_ora9mon start"
```

```
#
# Modify ORACLE_HOME to an appropriate value.
#
    export ORACLE_HOME=/opt/oracle/product/9.2.0
    export LD_LIBRARY_PATH=$ORACLE_HOME/lib
#
# Modify the database name to an appropriate value.
#
    clp_ora9mon orawatch -d XXXX
 else
    echo "ERROR_DISK from FAILOVER"
 fi
else
 echo "NO_CLP"
fi
echo "EXIT"
exit 0
```

10.4.2stop.sh

```
#! /bin/sh
#***************************************
#*               stop.sh               *
#***************************************

if [ "$CLP_EVENT" = "START" ]
then
 if [ "$CLP_DISK" = "SUCCESS" ]
 then
    echo "NORMAL1"
    if [ "$CLP_SERVER" = "HOME" ]
    then
       echo "NORMAL2"
    else
       echo "ON_OTHER1"
    fi
#
# Modify ORACLE_HOME to an appropriate value.
#
    export ORACLE_HOME=/opt/oracle/product/9.2.0
    export LD_LIBRARY_PATH=$ORACLE_HOME/lib

    echo "clp_ora9mon stop"
    clp_ora9mon orawatch --stop
 else
    echo "ERROR_DISK from START"
 fi
elif [ "$CLP_EVENT" = "FAILOVER" ]
then
 if [ "$CLP_DISK" = "SUCCESS" ]
 then
    echo "FAILOVER1"
    if [ "$CLP_SERVER" = "HOME" ]
    then
       echo "FAILOVER2"
    else
       echo "ON_OTHER2"
    fi
#
# Modify ORACLE_HOME to an appropriate value.
#
    export ORACLE_HOME=/opt/oracle/product/9.2.0
    export LD_LIBRARY_PATH=$ORACLE_HOME/lib

    echo "clp_ora9mon stop"
    clp_ora9mon orawatch --stop
 else
    echo "ERROR_DISK from FAILOVER"
```

```
 fi
else
 echo "NO_CLP"
fi
echo "EXIT"
exit 0
```

10.5 For Oracle10g

Two templates, start.sh and stop.sh, are provided for the Oracle10g scripts.
Modify and use them according to the operational environment and other requirements. Where modification is made are indicated in underlined boldface italics in the scripts in the following section.

10.5.1start.sh

```
#! /bin/sh
#****************************************
#*              start.sh                *
#****************************************

if [ "$CLP_EVENT" = "START" ]
then
 if [ "$CLP_DISK" = "SUCCESS" ]
 then
    echo "NORMAL1"
    if [ "$CLP_SERVER" = "HOME" ]
    then
       echo "NORMAL2"
    else
       echo "ON_OTHER1"
    fi
    date +"%Y/%m/%d %T"
    echo "Oracle10g start"
#
# Modify the user name and listener name to appropriate values.
#
    su - oracle -c "lsnrctl start LISTENER"
#
# Modify the user name, SID name, start script full path name to appropriate values.
#
    su - oracle -c "export ORACLE_SID=orcl;sqlplus /nolog @/XXXX/startup.sql"
 else
    echo "ERROR_DISK from START"
 fi
elif [ "$CLP_EVENT" = "FAILOVER" ]
then
 if [ "$CLP_DISK" = "SUCCESS" ]
 then
    echo "FAILOVER1"
```

```
    if [ "$CLP_SERVER" = "HOME" ]
    then
       echo "FAILOVER2"
    else
       echo "ON_OTHER2"
    fi
    date +"%Y/%m/%d %T"
    echo "Oracle10g start"
  #
  # Modify the user name, and listener name to appropriate values.
  #
    su - oracle -c "lsnrctl start LISTENER"
  #
  # Modify the user name, SID name, and start script full path name to appropriate
values.
  #
    su - oracle -c "export ORACLE_SID=orcl;sqlplus /nolog @/XXXX/startup.sql"
   else
    echo "ERROR_DISK from FAILOVER"
   fi
  else
   echo "NO_CLP"
  fi
  echo "EXIT"
  exit 0
```

10.5.2stop.sh

```
  #! /bin/sh
  #****************************************
  #*               stop.sh                *
  #****************************************

  if [ "$CLP_EVENT" = "START" ]
  then
   if [ "$CLP_DISK" = "SUCCESS" ]
   then
    echo "NORMAL1"
    if [ "$CLP_SERVER" = "HOME" ]
    then
       echo "NORMAL2"
    else
       echo "ON_OTHER1"
    fi
    date +"%Y/%m/%d %T"
    echo "Oracle10g stop"
  #
  # Modify the user name, SID name, and stop script full path name to appropriate
values.
  #
    su - oracle -c "export ORACLE_SID=orcl;sqlplus /nolog @/XXXX/shutdown.sql"
```

```
  #
  # Modify the user name and listener name to appropriate values.
  #
     su - oracle -c "lsnrctl stop LISTENER"
   else
     echo "ERROR_DISK from START"
   fi
  elif [ "$CLP_EVENT" = "FAILOVER" ]
  then
   if [ "$CLP_DISK" = "SUCCESS" ]
   then
     echo "FAILOVER1"
     if [ "$CLP_SERVER" = "HOME" ]
     then
        echo "FAILOVER2"
     else
        echo "ON_OTHER2"
     fi
     date +"%Y/%m/%d %T"
     echo "Oracle10g stop"
  #
  # Modify the user name, SID name, and stop script full path name to appropriate
values.
  #
     su - oracle -c "export ORACLE_SID=orcl;sqlplus /nolog @/XXXX/shutdown.sql"
  #
  # Modify the user name and listener name to appropriate values.
  #
     su - oracle -c "lsnrctl stop LISTENER"
   else
     echo "ERROR_DISK from FAILOVER"
   fi
  else
   echo "NO_CLP"
  fi
  echo "EXIT"
  exit 0
```

10.5.3startup.sql/shutdown.sql

To start up Oracle10g, both startup.sql and shutdown.sql files used by sqlplus are required. You can choose any name and path for these files but make sure they match with the full path name specified with start.sh/stop.sh.

Write the following information in each file. In addition, there is no template for this information.

startup.sql

```
connect / as sysdba
startup pfile=initialization file name.
exit
```

shutdown.sql

```
connect / as sysdba
shutdown immediate
exit
```

10.6 For Oracle10g Monitoring

Two templates, start.sh and stop.sh, are provided for the Oracle10g monitoring scripts. Modify and use them according to the operational environment and other requirements. Where modification is made are indicated in underlined boldface italics in the scripts in the following section.

10.6.1start.sh

```
#! /bin/sh
#***************************************
#*              start.sh               *
#***************************************

if [ "$CLP_EVENT" = "START" ]
then
 if [ "$CLP_DISK" = "SUCCESS" ]
 then
    echo "NORMAL1"
    if [ "$CLP_SERVER" = "HOME" ]
    then
       echo "NORMAL2"
    else
       echo "ON_OTHER1"
    fi
    date +"%Y/%m/%d %T"
    echo "clp_ora10mon start"
#
# Modify ORACLE_HOME to an appropriate value.
#
    export ORACLE_HOME=/u01/app/oracle/product/10.1.0/db_1
    export LD_LIBRARY_PATH=$ORACLE_HOME/lib
#
# Change NLS_LANG parameter to an appropriate value.
#
#     export NLS_LANG=AMERICAN_AMERICA.JA16EUC
#
```

```
# Modify the database name to an appropriate value.
#
   clp_ora10mon orawatch -d XXXX
 else
   echo "ERROR_DISK from START"
 fi
elif [ "$CLP_EVENT" = "FAILOVER" ]
then
 if [ "$CLP_DISK" = "SUCCESS" ]
 then
   echo "FAILOVER1"
   if [ "$CLP_SERVER" = "HOME" ]
   then
      echo "FAILOVER2"
   else
      echo "ON_OTHER2"
   fi
   date +"%Y/%m/%d %T"
   echo "clp_ora10mon start"
#
# Modify ORACLE_HOME to an appropriate value.
#
   export ORACLE_HOME=/u01/app/oracle/product/10.1.0/db_1
   export LD_LIBRARY_PATH=$ORACLE_HOME/lib
#
# Change NLS_LANG parameter to an appropriate value.
#
#     export NLS_LANG=AMERICAN_AMERICA.JA16EUC
#
# Modify the database name to an appropriate value.
#
   clp_ora10mon orawatch -d XXXX
 else
   echo "ERROR_DISK from FAILOVER"
 fi
else
 echo "NO_CLP"
fi
echo "EXIT"
exit 0
```

10.6.2stop.sh

```
#! /bin/sh
#***************************************
#*               stop.sh               *
#***************************************

if [ "$CLP_EVENT" = "START" ]
then
```

```
if [ "$CLP_DISK" = "SUCCESS" ]
then
   echo "NORMAL1"
   if [ "$CLP_SERVER" = "HOME" ]
   then
      echo "NORMAL2"
   else
      echo "ON_OTHER1"
   fi
#
# Modify ORACLE_HOME to an appropriate value.
#
   export ORACLE_HOME=/u01/app/oracle/product/10.1.0/db_1
   export LD_LIBRARY_PATH=$ORACLE_HOME/lib

   date +"%Y/%m/%d %T"
   echo "clp_ora10mon stop"
   clp_ora10mon orawatch --stop
else
   echo "ERROR_DISK from START"
fi
elif [ "$CLP_EVENT" = "FAILOVER" ]
then
if [ "$CLP_DISK" = "SUCCESS" ]
then
   echo "FAILOVER1"
   if [ "$CLP_SERVER" = "HOME" ]
   then
      echo "FAILOVER2"
   else
      echo "ON_OTHER2"
   fi
#
# Modify ORACLE_HOME to an appropriate value.
#
   export ORACLE_HOME=/u01/app/oracle/product/10.1.0/db_1
   export LD_LIBRARY_PATH=$ORACLE_HOME/lib

   date +"%Y/%m/%d %T"
   echo "clp_ora10mon stop"
   clp_ora10mon orawatch --stop
else
   echo "ERROR_DISK from FAILOVER"
fi
else
echo "NO_CLP"
fi
echo "EXIT"
exit 0
```

10.7 For PostgreSQL

Two templates, start.sh and stop.sh, are provided for the PostgreSQL monitoring scripts.
Modify and use them according to the operational environment and other requirements. Where modification is made are indicated in underlined boldface italics in the scripts in the following section.
When starting multiple PostgreSQLs in a single server, in a system such as bidirectional standby, set identifiers, database domain directory and port number in the way they do not overlap.

10.7.1start.sh

```
#! /bin/sh
#****************************************
#*              start.sh                *
#****************************************

if [ "$CLP_EVENT" = "START" ]
then
 if [ "$CLP_DISK" = "SUCCESS" ]
 then
    echo "NORMAL1"
    if [ "$CLP_SERVER" = "HOME" ]
    then
       echo "NORMAL2"
    else
       echo "ON_OTHER1"
    fi
    echo "PostgreSQL start"
#
# Modify a PostgreSQL user name, database domain directory, and port number to appropriate values.
#
    su - postgres -c "pg_ctl start -D /mnt/pgsql/data -l /dev/null -o '-i -p 5432'"
 else
    echo "ERROR_DISK from START"
 fi
elif [ "$CLP_EVENT" = "FAILOVER" ]
then
 if [ "$CLP_DISK" = "SUCCESS" ]
 then
    echo "FAILOVER1"
    if [ "$CLP_SERVER" = "HOME" ]
    then
       echo "FAILOVER2"
    else
       echo "ON_OTHER2"
    fi
    echo "PostgreSQL start"
```

```
  #
  # Modify a PostgreSQL user name, database domain directory, and port number to
appropriate values.
  #
     su - postgres -c "pg_ctl start -D /mnt/pgsql/data -l /dev/null -o '-i -p
5432"
   else
      echo "ERROR_DISK from FAILOVER"
   fi
  else
   echo "NO_CLP"
  fi
  echo "EXIT"
  exit 0
```

10.7.2 stop.sh

```
#! /bin/sh
#****************************************
#*               stop.sh                *
#****************************************

if [ "$CLP_EVENT" = "START" ]
then
 if [ "$CLP_DISK" = "SUCCESS" ]
 then
    echo "NORMAL1"
    if [ "$CLP_SERVER" = "HOME" ]
    then
       echo "NORMAL2"
    else
       echo "ON_OTHER1"
    fi
    echo "PostgreSQL stop"
#
# Modify a PostgreSQL user name and database domain directory to appropriate
values.
#
    su - postgres -c 'pg_ctl stop -D /mnt/pgsql/data -m fast'
 else
    echo "ERROR_DISK from START"
 fi
elif [ "$CLP_EVENT" = "FAILOVER" ]
then
 if [ "$CLP_DISK" = "SUCCESS" ]
 then
    echo "FAILOVER1"
    if [ "$CLP_SERVER" = "HOME" ]
    then
       echo "FAILOVER2"
    else
       echo "ON_OTHER2"
    fi
    echo "PostgreSQL stop"
#
# Modify a PostgreSQL user name and database domain directory to appropriate
values.
#
    su - postgres -c 'pg_ctl stop -D /mnt/pgsql/data -m fast'
 else
    echo "ERROR_DISK from FAILOVER"
 fi
else
 echo "NO_CLP"
fi
```

```
echo "EXIT"
exit 0
```

10.8 PostgreSQL Monitoring (clp_psql72mon)

Two templates, start.sh and stop.sh, are provided for the PostgreSQL monitoring (clp_psql72mon) scripts.
Modify and use them according to the operational environment and other requirements. Where modification is made are indicated in underlined boldface italics in the scripts in the following section.

10.8.1 start.sh

```
#! /bin/sh
#****************************************
#*              start.sh                *
#****************************************

if [ "$CLP_EVENT" = "START" ]
then
 if [ "$CLP_DISK" = "SUCCESS" ]
 then
    echo "NORMAL1"
    if [ "$CLP_SERVER" = "HOME" ]
    then
       echo "NORMAL2"
    else
       echo "ON_OTHER1"
    fi
    date +"%Y/%m/%d %T"
    echo "clp_psql72mon start"
#
# Modify the PostgreSQL library path to an appropriate value.
#
#     export LD_LIBRARY_PATH=/usr/local/pgsql/lib
#
# Modify a database name, port number, and etc. to appropriate values.
#
    clp_psql72mon psqlwatch -d XXXX -n 5432
 else
    echo "ERROR_DISK from START"
 fi
elif [ "$CLP_EVENT" = "FAILOVER" ]
then
 if [ "$CLP_DISK" = "SUCCESS" ]
 then
    echo "FAILOVER1"
    if [ "$CLP_SERVER" = "HOME" ]
    then
       echo "FAILOVER2"
```

```
    else
       echo "ON_OTHER2"
    fi
    date +"%Y/%m/%d %T"
    echo "clp_psql72mon start"
#
# Modify the PostgreSQL library path to an appropriate value.
#
#     export LD_LIBRARY_PATH=/usr/local/pgsql/lib
#
# Modify a database name, port number and etc to appropriate values.
#
    clp_psql72mon psqlwatch -d XXXX -n 5432
 else
    echo "ERROR_DISK from FAILOVER"
 fi
else
 echo "NO_CLP"
fi
echo "EXIT"
exit 0
```

10.8.2 stop.sh

```
#! /bin/sh
#****************************************
#*               stop.sh                *
#****************************************

if [ "$CLP_EVENT" = "START" ]
then
 if [ "$CLP_DISK" = "SUCCESS" ]
 then
    echo "NORMAL1"
    if [ "$CLP_SERVER" = "HOME" ]
    then
       echo "NORMAL2"
    else
       echo "ON_OTHER1"
    fi
    date +"%Y/%m/%d %T"
    echo "clp_psql72mon stop"
#
# Modify the PostgreSQL library path to an appropriate value.
#
#     export LD_LIBRARY_PATH=/usr/local/pgsql/lib
#
# Modify a database name, port number, and etc. to appropriate values.
#
    clp_psql72mon psqlwatch --stop
```

```
 else
    echo "ERROR_DISK from START"
 fi
elif [ "$CLP_EVENT" = "FAILOVER" ]
then
 if [ "$CLP_DISK" = "SUCCESS" ]
 then
    echo "FAILOVER1"
    if [ "$CLP_SERVER" = "HOME" ]
    then
       echo "FAILOVER2"
    else
       echo "ON_OTHER2"
    fi
    date +"%Y/%m/%d %T"
    echo "clp_psql72mon stop"
#
# Modify the PostgreSQL library path to an appropriate value.
#
#     export LD_LIBRARY_PATH=/usr/local/pgsql/lib
#
# Modify a database name, port number and etc to appropriate values.
#
    clp_psql72mon psqlwatch --stop
 else
    echo "ERROR_DISK from FAILOVER"
 fi
else
 echo "NO_CLP"
fi
echo "EXIT"
exit 0
```

10.9 PostgreSQL Monitoring (clp_psql73mon)

Two templates, start.sh and stop.sh, are provided for the PostgreSQL monitoring (clp_psql73mon) scripts.
Modify and use them according to the operational environment and other requirements. Where modification is made are indicated in underlined boldface italics in the scripts in the following section.

10.9.1start.sh

```
#! /bin/sh
#****************************************
#*              start.sh                *
#****************************************

if [ "$CLP_EVENT" = "START" ]
then
 if [ "$CLP_DISK" = "SUCCESS" ]
```

```
then
   echo "NORMAL1"
   if [ "$CLP_SERVER" = "HOME" ]
   then
      echo "NORMAL2"
   else
      echo "ON_OTHER1"
   fi
   date +"%Y/%m/%d %T"
   echo "clp_psql73mon start"
#
# Modify the PostgreSQL library path to an appropriate value.
#
#     export LD_LIBRARY_PATH=/usr/local/pgsql/lib
#
# Modify a database name, port number, and etc. to appropriate values.
#
   clp_psql73mon psqlwatch -d XXXX -n 5432
 else
   echo "ERROR_DISK from START"
 fi
elif [ "$CLP_EVENT" = "FAILOVER" ]
then
 if [ "$CLP_DISK" = "SUCCESS" ]
 then
   echo "FAILOVER1"
   if [ "$CLP_SERVER" = "HOME" ]
   then
      echo "FAILOVER2"
   else
      echo "ON_OTHER2"
   fi
   date +"%Y/%m/%d %T"
   echo "clp_psql73mon start"
#
# Modify the PostgreSQL library path to an appropriate value.
#
#     export LD_LIBRARY_PATH=/usr/local/pgsql/lib
#
# Modify a database name, port number and etc to appropriate values.
#
   clp_psql73mon psqlwatch -d XXXX -n 5432
 else
   echo "ERROR_DISK from FAILOVER"
 fi
else
 echo "NO_CLP"
fi
echo "EXIT"
exit 0
```

10.9.2 stop.sh

```
#! /bin/sh
#****************************************
#*               stop.sh                *
#****************************************

if [ "$CLP_EVENT" = "START" ]
then
 if [ "$CLP_DISK" = "SUCCESS" ]
 then
    echo "NORMAL1"
    if [ "$CLP_SERVER" = "HOME" ]
    then
       echo "NORMAL2"
    else
       echo "ON_OTHER1"
    fi
    date +"%Y/%m/%d %T"
    echo "clp_psql73mon stop"
#
# Modify the PostgreSQL library path to an appropriate value.
#
#     export LD_LIBRARY_PATH=/usr/local/pgsql/lib
#
# Modify a database name, port number, and etc. to appropriate values.
#
    clp_psql73mon psqlwatch --stop
 else
    echo "ERROR_DISK from START"
 fi
elif [ "$CLP_EVENT" = "FAILOVER" ]
then
 if [ "$CLP_DISK" = "SUCCESS" ]
 then
    echo "FAILOVER1"
    if [ "$CLP_SERVER" = "HOME" ]
    then
       echo "FAILOVER2"
    else
       echo "ON_OTHER2"
    fi
    date +"%Y/%m/%d %T"
    echo "clp_psql73mon stop"
#
# Modify the PostgreSQL library path to an appropriate value.
#
#     export LD_LIBRARY_PATH=/usr/local/pgsql/lib
#
# Modify a database name, port number and etc to appropriate values.
```

```
#
   clp_psql73mon psql watch --stop
 else
   echo "ERROR_DISK from FAILOVER"
 fi
else
 echo "NO_CLP"
fi
echo "EXIT"
exit 0
```

10.10 PostgreSQL Monitoring (clp_psql80mon)

Two templates, start.sh and stop.sh, are provided for the PostgreSQL monitoring (clp_psql80mon) scripts.
Modify and use them according to the operational environment and other requirements. Where modification is made are indicated in underlined boldface italics in the scripts in the following section.

10.10.1 start.sh

```
#! /bin/sh
#****************************************
#*              start.sh                *
#****************************************

if [ "$CLP_EVENT" = "START" ]
then
 if [ "$CLP_DISK" = "SUCCESS" ]
 then
    echo "NORMAL1"
    if [ "$CLP_SERVER" = "HOME" ]
    then
       echo "NORMAL2"
    else
       echo "ON_OTHER1"
    fi
    date +"%Y/%m/%d %T"
    echo "clp_psql80mon start"
#
# Modify the PostgreSQL library path to an appropriate value.
#
#     export LD_LIBRARY_PATH=/usr/local/pgsql/lib
#
# Modify a database name, port number, and etc. to appropriate values.
#
    clp_psql80mon psqlwatch -d XXXX -n 5432
 else
    echo "ERROR_DISK from START"
 fi
elif [ "$CLP_EVENT" = "FAILOVER" ]
then
 if [ "$CLP_DISK" = "SUCCESS" ]
 then
    echo "FAILOVER1"
    if [ "$CLP_SERVER" = "HOME" ]
    then
       echo "FAILOVER2"
    else
       echo "ON_OTHER2"
    fi
    date +"%Y/%m/%d %T"
```

```
    echo "clp_psql80mon start"
#
# Modify the PostgreSQL library path to an appropriate value.
#
#      export LD_LIBRARY_PATH=/usr/local/pgsql/lib
#
# Modify a database name, port number, and etc. to appropriate values.
#
    clp_psql80mon psqlwatch -d XXXX -n 5432
 else
    echo "ERROR_DISK from FAILOVER"
 fi
else
 echo "NO_CLP"
fi
echo "EXIT"
exit 0
```

10.10.2 stop.sh

```
#! /bin/sh
#***************************************
#*               stop.sh               *
#***************************************

if [ "$CLP_EVENT" = "START" ]
then
 if [ "$CLP_DISK" = "SUCCESS" ]
 then
    echo "NORMAL1"
    if [ "$CLP_SERVER" = "HOME" ]
    then
       echo "NORMAL2"
    else
       echo "ON_OTHER1"
    fi
    date +"%Y/%m/%d %T"
    echo "clp_psql80mon stop"
#
# Modify the PostgreSQL library path to an appropriate value.
#
#     export LD_LIBRARY_PATH=/usr/local/pgsql/lib
    clp_psql80mon psqlwatch --stop
 else
    echo "ERROR_DISK from START"
 fi
elif [ "$CLP_EVENT" = "FAILOVER" ]
then
 if [ "$CLP_DISK" = "SUCCESS" ]
 then
    echo "FAILOVER1"
    if [ "$CLP_SERVER" = "HOME" ]
    then
       echo "FAILOVER2"
    else
       echo "ON_OTHER2"
    fi
    date +"%Y/%m/%d %T"
    echo "clp_psql80mon stop"
#
# Modify the PostgreSQL library path to an appropriate value.
#
#     export LD_LIBRARY_PATH=/usr/local/pgsql/lib
    clp_psql80mon psqlwatch --stop
 else
    echo "ERROR_DISK from FAILOVER"
 fi
else
 echo "NO_CLP"
```

```
fi
echo "EXIT"
exit 0
```

10.11 For PowerGres Plus

Two script templates, start.sh and stop.sh, are provided for the PowerGres Plus scripts.
Modify and use them according to the operational environment and other requirements.
Where modification is made are indicated with underlined boldface italics in the scripts in the following sections.
If multiple PowerGres Plus are started on one server for multi-directional standby, etc., specify a different indentifier, database area directory, and port number for each PowerGres Plus.

10.11.1 start.sh

```
#! /bin/sh
#****************************************
#*              start.sh                *
#****************************************

pidfile_check()
{
 echo "pidfile_check ENTRY"
 if [ -f /PlusData/data1/postmaster.pid ]
 then
    echo "Evacuation of a postmaster.pid file /tmp/postmaster.pid.$$"
    mv /PlusData/data1/postmaster.pid /tmp/postmaster.pid.$$
 fi
#
# Modify the PowerGres Plus user name, database area directory, and
# port number to appropriate values.
#
    su - postgres -c "export PGPORT=5432;/usr/local/pgsqlplus/bin/pg_ctl start
 -D /PlusData/data1 -w -l /dev/null -o '-i -p 5432'"
 if [ $? -ne 0 ]
 then
    if [ ! -f /PlusData/data1/postmaster.pid ]
    then
       echo "A postmaster.pid file does not exist."
       if [ -f /tmp/postmaster.pid.$$ ]
       then
          echo "Restoration of a postmaster.pid file /tmp/postmaster.pid.$$"
          mv /tmp/postmaster.pid.$$ /PlusData/data1/postmaster.pid
       fi
    fi
    if [ -f /tmp/postmaster.pid.$$ ]
    then
       echo "Deletion of an evacuation file /tmp/postmaster.pid.$$"
       rm /tmp/postmaster.pid.$$
    fi
    echo "pidfile_check EXIT(1)"
```

```
    return 1
  fi
  if [ ! -f /PlusData/data1/postmaster.pid ]
  then
    echo "A postmaster.pid file does not exist."
    if [ -f /tmp/postmaster.pid.$$ ]
    then
      echo "Restoration of a postmaster.pid file /tmp/postmaster.pid.$$"
      mv /tmp/postmaster.pid.$$ /PlusData/data1/postmaster.pid
    fi
  fi
  if [ -f /tmp/postmaster.pid.$$ ]
  then
    echo "Deletion of an evacuation file /tmp/postmaster.pid.$$"
    rm /tmp/postmaster.pid.$$
  fi
  echo "pidfile_check EXIT(0)"
  return 0
 }

 if [ "$CLP_EVENT" = "START" ]
 then
  if [ "$CLP_DISK" = "SUCCESS" ]
  then
    echo "NORMAL1"
    if [ "$CLP_SERVER" = "HOME" ]
    then
      echo "NORMAL2"
    else
      echo "ON_OTHER1"
    fi
    date +"%Y/%m/%d %T"
    echo "PostgreSQL Plus start"
 #
 # Modify the PowerGres Plus user name, database area directory, and
 # port number to appropriate values.
 #
    su - postgres -c "export PGPORT=5432;/usr/local/pgsqlplus/bin/pg_ctl start
-D /PlusData/data1 -w -l /dev/null -o '-i -p 5432'"
    if [ $? -ne 0 ]
    then
      pidfile_check
      if [ $? -ne 0 ]
      then
        echo "START FAILED"
      fi
    fi
  else
    echo "ERROR_DISK from START"
  fi
 elif [ "$CLP_EVENT" = "FAILOVER" ]
 then
```

```
  if [ "$CLP_DISK" = "SUCCESS" ]
  then
     echo "FAILOVER1"
     if [ "$CLP_SERVER" = "HOME" ]
     then
        echo "FAILOVER2"
     else
        echo "ON_OTHER2"
     fi
     date +"%Y/%m/%d %T"
     echo "PostgreSQL Plus start"
  #
  # Modify the PowerGres Plus user name, database area directory, and
  # port number to appropriate values.
  #
     su - postgres -c "export PGPORT=5432;/usr/local/pgsqlplus/bin/pg_ctl start
-D /PlusData/data1 -w -l /dev/null -o '-i -p 5432'"
     if [ $? -ne 0 ]
     then
        pidfile_check
        if [ $? -ne 0 ]
        then
           echo "FAILOVER FAILED"
        fi
     fi
  else
     echo "ERROR_DISK from FAILOVER"
  fi
 else
  echo "NO_CLP"
 fi
 echo "EXIT"
 exit 0
```

10.11.2 stop.sh

```
#! /bin/sh
#****************************************
#*               stop.sh                *
#****************************************

if [ "$CLP_EVENT" = "START" ]
then
 if [ "$CLP_DISK" = "SUCCESS" ]
 then
    echo "NORMAL1"
    if [ "$CLP_SERVER" = "HOME" ]
    then
       echo "NORMAL2"
    else
       echo "ON_OTHER1"
    fi
    date +"%Y/%m/%d %T"
    echo "PostgreSQL Plus stop"
#
# Modify the PowerGres Plus user name and database area directory
# to appropriate values.
#
    su - postgres -c '/usr/local/pgsqlplus/bin/pg_ctl stop -D /PlusData/data1
-m fast'
 else
    echo "ERROR_DISK from START"
 fi
elif [ "$CLP_EVENT" = "FAILOVER" ]
then
 if [ "$CLP_DISK" = "SUCCESS" ]
 then
    echo "FAILOVER1"
    if [ "$CLP_SERVER" = "HOME" ]
    then
       echo "FAILOVER2"
    else
       echo "ON_OTHER2"
    fi
    date +"%Y/%m/%d %T"
    echo "PostgreSQL Plus stop"
#
# Modify the PowerGres Plus user name and database area directory
# to appropriate values.
#
    su - postgres -c '/usr/local/pgsqlplus/bin/pg_ctl stop -D /PlusData/data1
-m fast'
 else
    echo "ERROR_DISK from FAILOVER"
```

```
 fi
else
 echo "NO_CLP"
fi
echo "EXIT"
exit 0
```

10.12 For PowerGres Plus monitoring

Two script templates, start.sh and stop.sh, are provided for the PowerGres Plus monitoring scripts.
Modify and use them according to the operational environment and other requirements. Where modification is made are indicated with underlined boldface italics in the scripts in the following sections.

10.12.1 start.sh

```
#! /bin/sh
#****************************************
#*              start.sh                *
#****************************************

if [ "$CLP_EVENT" = "START" ]
then
 if [ "$CLP_DISK" = "SUCCESS" ]
 then
    echo "NORMAL1"
    if [ "$CLP_SERVER" = "HOME" ]
    then
       echo "NORMAL2"
    else
       echo "ON_OTHER1"
    fi
    date +"%Y/%m/%d %T"
    echo "clp_psql73mon start"
#
# Modify the PowerGres Plus library path to an appropriate value.
#
    export LD_LIBRARY_PATH=/usr/local/pgsqlplus/lib
#
# Modify a database name, port number, etc. to appropriate values.
#
    clp_psql73mon pgpluswatch -d XXXX -n 5432
 else
    echo "ERROR_DISK from START"
 fi
elif [ "$CLP_EVENT" = "FAILOVER" ]
then
 if [ "$CLP_DISK" = "SUCCESS" ]
 then
    echo "FAILOVER1"
    if [ "$CLP_SERVER" = "HOME" ]
    then
       echo "FAILOVER2"
    else
       echo "ON_OTHER2"
    fi
```

```
    date +"%Y/%m/%d %T"
    echo "clp_psql73mon start"
#
# Modify the PowerGres Plus library path to an appropriate value.
#
    export LD_LIBRARY_PATH=/usr/local/pgsqlplus/lib
#
# Modify a database name, port number, etc. to appropriate values.
#
    clp_psql73mon pgpluswatch -d XXXX -n 5432
 else
    echo "ERROR_DISK from FAILOVER"
 fi
else
 echo "NO_CLP"
fi
echo "EXIT"
exit 0
```

10.12.2 stop.sh

```
#! /bin/sh
#****************************************
#*               stop.sh                *
#****************************************

if [ "$CLP_EVENT" = "START" ]
then
 if [ "$CLP_DISK" = "SUCCESS" ]
 then
    echo "NORMAL1"
    if [ "$CLP_SERVER" = "HOME" ]
    then
       echo "NORMAL2"
    else
       echo "ON_OTHER1"
    fi
    date +"%Y/%m/%d %T"
    echo "clp_psql73mon stop"
#
# Modify the PowerGres Plus library path to an appropriate value.
#
    export LD_LIBRARY_PATH=/usr/local/pgsqlplus/lib

    clp_psql73mon pgpluswatch --stop
 else
    echo "ERROR_DISK from START"
 fi
elif [ "$CLP_EVENT" = "FAILOVER" ]
then
 if [ "$CLP_DISK" = "SUCCESS" ]
 then
    echo "FAILOVER1"
    if [ "$CLP_SERVER" = "HOME" ]
    then
       echo "FAILOVER2"
    else
       echo "ON_OTHER2"
    fi
    date +"%Y/%m/%d %T"
    echo "clp_psql73mon stop"
#
# Modify the PowerGres Plus library path to an appropriate value.
#
    export LD_LIBRARY_PATH=/usr/local/pgsqlplus/lib

    clp_psql73mon pgpluswatch --stop
 else
    echo "ERROR_DISK from FAILOVER"
```

```
 fi
else
 echo "NO_CLP"
fi
echo "EXIT"
exit 0
```

10.13For MySQL

Two scripts, start.sh and stop.sh, and my.cnf, a template for the file to define MySQL starting option, are provided for the MySQL scripts.
Modify and use them according to the operational environment and other requirements. Where modification is made are indicated in underlined boldface italics in the scripts in the following section.

10.13.1 start.sh

```
#! /bin/sh
#****************************************
#*              start.sh                *
#****************************************

if [ "$CLP_EVENT" = "START" ]
then
 if [ "$CLP_DISK" = "SUCCESS" ]
 then
    echo "NORMAL1"
    if [ "$CLP_SERVER" = "HOME" ]
    then
       echo "NORMAL2"
    else
       echo "ON_OTHER1"
    fi
    echo "MySQL start"
#
# Modify the file for defining starting option to an appropriate value.
# When registering this as a script file of ExpressCluster, specify the group name of a
path.
#
    safe_mysqld
--defaults-file=/opt/nec/clusterpro/scripts/group/groupname/my.cnf &
 else
    echo "ERROR_DISK from START"
 fi
elif [ "$CLP_EVENT" = "FAILOVER" ]
then
 if [ "$CLP_DISK" = "SUCCESS" ]
 then
    echo "FAILOVER1"
```

```
      if [ "$CLP_SERVER" = "HOME" ]
      then
         echo "FAILOVER2"
      else
         echo "ON_OTHER2"
      fi
      echo "MySQL start"
  #
  # Modify the file for defining starting option to an appropriate value.
  # When registering this as a script file of ExpressCluster, specify the group name of a
path.
  #
      safe_mysqld
--defaults-file=/opt/nec/clusterpro/scripts/group/groupname/my.cnf &
   else
      echo "ERROR_DISK from FAILOVER"
   fi
  else
   echo "NO_CLP"
  fi
  echo "EXIT"
  exit 0
```

10.13.2 stop.sh

```
#! /bin/sh
#***************************************
#*               stop.sh               *
#***************************************

if [ "$CLP_EVENT" = "START" ]
then
 if [ "$CLP_DISK" = "SUCCESS" ]
 then
    echo "NORMAL1"
    if [ "$CLP_SERVER" = "HOME" ]
    then
       echo "NORMAL2"
    else
       echo "ON_OTHER1"
    fi
    echo "MySQL stop"
#
# Modify the file for defining starting option to an appropriate value.
# When registering this as a script file of ExpressCluster, specify the group name of a
path.
#
    mysqladmin
--defaults-file=/opt/nec/clusterpro/scripts/group/groupname/my.cnf shutdown
 else
    echo "ERROR_DISK from START"
 fi
elif [ "$CLP_EVENT" = "FAILOVER" ]
then
 if [ "$CLP_DISK" = "SUCCESS" ]
 then
    echo "FAILOVER1"
    if [ "$CLP_SERVER" = "HOME" ]
    then
       echo "FAILOVER2"
    else
       echo "ON_OTHER2"
    fi
    echo "MySQL stop"
#
# Modify the file for defining starting option to an appropriate value.
# when registering this as a script file of ExpressCluster, specify the group name of a
path.
#
    mysqladmin
--defaults-file=/opt/nec/clusterpro/scripts/group/groupname/my.cnf shutdown
 else
    echo "ERROR_DISK from FAILOVER"
 fi
```

```
else
 echo "NO_CLP"
fi
echo "EXIT"
exit 0
```

10.13.3 my.cnf

If a file for defining starting option created at the time of MySQL startup, parameter specification at startup becomes easier. You can choose any path or file name for this file but make sure the name you choose matches with the full path name specified with start.sh/stop.sh. You may register as a script of a failover group, or may create on each server apart from a script file.

***If you register as a script, do not modify the information directly on each server. Always make modifications from ExpressCluster trekking tool.**

```
[client]
port            = 3306
socket      = /var/lib/mysql/mysql.sock

[mysqld]
port            = 3306
socket      = /var/lib/mysql/mysql.sock
pid-file    = /var/lib/mysql/mysql.pid
datadir     = /mnt/mysql/
```

Specify a directory for storage in datadir. Typically, a directory on a shared disk is specified.

In a bidirectional standby system, you must specify different values in items above for each failover group.

Example:

my.cnf for the failover group 1

```
[client]
port            = 3306
socket          = /var/lib/mysql/mysql1.sock
[mysqld]
port            = 3306
socket          = /var/lib/mysql/mysql1.sock
pid-file        = /var/lib/mysql/mysql1.pid
datadir         = /mnt/mysql1/
```

my.cnf for the failover group 2

```
[client]
port            = 3307
socket          = /var/lib/mysql/mysql2.sock
[mysqld]
port            = 3307
socket          = /var/lib/mysql/mysql2.sock
pid-file        = /var/lib/mysql/mysql2.pid
datadir         = /mnt/mysql2/
```

10.14For MySQL Monitoring (clp_mysql323mon)

Two templates, start.sh and stop.sh, are provided for the MySQL monitoring (clp_mysql323mon) scripts.
Modify and use them according to the operational environment and other requirements. Where modification is made are indicated in underlined boldface italics in the scripts in the following section.

10.14.1 start.sh

```
#! /bin/sh
#****************************************
#*              start.sh                *
#****************************************

if [ "$CLP_EVENT" = "START" ]
then
 if [ "$CLP_DISK" = "SUCCESS" ]
 then
    echo "NORMAL1"
    if [ "$CLP_SERVER" = "HOME" ]
    then
       echo "NORMAL2"
    else
       echo "ON_OTHER1"
    fi
    date +"%Y/%m/%d %T"
    echo "clp_mysql323mon start"
#
# If you do not specify the -a parameter, modify the file name for socket to an appropriate value.
#
#     export MYSQL_UNIX_PORT=/var/lib/mysql/mysql.sock
#
# Modify the database name, the IP address, and etc. to appropriate values.
#
    clp_mysql323mon mysqlwatch -d XXXX -a nnn.nnn.nnn.nnn
 else
    echo "ERROR_DISK from START"
 fi
elif [ "$CLP_EVENT" = "FAILOVER" ]
then
 if [ "$CLP_DISK" = "SUCCESS" ]
 then
    echo "FAILOVER1"
    if [ "$CLP_SERVER" = "HOME" ]
    then
       echo "FAILOVER2"
    else
       echo "ON_OTHER2"
```

```
    fi
    date +"%Y/%m/%d %T"
    echo "clp_mysql323mon start"
#
# If you do not specify the -a parameter, modify the file name for socket to an
appropriate value.
#
#     export MYSQL_UNIX_PORT=/var/lib/mysql/mysql.sock
#
# Modify the database name, the IP address, and etc. to the appropriate values.
#
    clp_mysql323mon mysqlwatch -d XXXX -a nnn.nnn.nnn.nnn
 else
    echo "ERROR_DISK from FAILOVER"
 fi
else
 echo "NO_CLP"
fi
echo "EXIT"
exit 0
```

10.14.2 stop.sh

```
#! /bin/sh
#****************************************
#*               stop.sh                *
#****************************************

if [ "$CLP_EVENT" = "START" ]
then
 if [ "$CLP_DISK" = "SUCCESS" ]
 then
    echo "NORMAL1"
    if [ "$CLP_SERVER" = "HOME" ]
    then
       echo "NORMAL2"
    else
       echo "ON_OTHER1"
    fi
    date +"%Y/%m/%d %T"
    echo "clp_mysql323mon stop"
    clp_mysql323mon mysqlwatch --stop
 else
    echo "ERROR_DISK from START"
 fi
elif [ "$CLP_EVENT" = "FAILOVER" ]
then
 if [ "$CLP_DISK" = "SUCCESS" ]
 then
    echo "FAILOVER1"
    if [ "$CLP_SERVER" = "HOME" ]
```

```
    then
       echo "FAILOVER2"
    else
       echo "ON_OTHER2"
    fi
    date +"%Y/%m/%d %T"
    echo "clp_mysql323mon stop"
    clp_mysql323mon mysqlwatch --stop
 else
    echo "ERROR_DISK from FAILOVER"
 fi
else
 echo "NO_CLP"
fi
echo "EXIT"
exit 0
```

10.15For MySQL Monitoring (clp_mysql40mon)

Two templates, start.sh and stop.sh, are provided for the MySQL monitoring (clp_mysql40mon) scripts.
Modify and use them according to the operational environment and other requirements. Where modification is made are indicated in underlined boldface italics in the scripts in the following section.

10.15.1 start.sh

```
#! /bin/sh
#****************************************
#*              start.sh                *
#****************************************

if [ "$CLP_EVENT" = "START" ]
then
 if [ "$CLP_DISK" = "SUCCESS" ]
 then
    echo "NORMAL1"
    if [ "$CLP_SERVER" = "HOME" ]
    then
       echo "NORMAL2"
    else
       echo "ON_OTHER1"
    fi
    date +"%Y/%m/%d %T"
    echo "clp_mysql40mon start"
#
# If you do not specify the -a parameter, modify the file name for socket to an appropriate value.
#
#     export MYSQL_UNIX_PORT=/var/lib/mysql/mysql.sock
#
```

```
# Modify the database name, the IP address, and etc. to the appropriate values.
#
    clp_mysql4Omon mysqlwatch -d XXXX -a nnn.nnn.nnn.nnn
 else
    echo "ERROR_DISK from START"
 fi
elif [ "$CLP_EVENT" = "FAILOVER" ]
then
 if [ "$CLP_DISK" = "SUCCESS" ]
 then
    echo "FAILOVER1"
    if [ "$CLP_SERVER" = "HOME" ]
    then
       echo "FAILOVER2"
    else
       echo "ON_OTHER2"
    fi
    date +"%Y/%m/%d %T"
    echo "clp_mysql4Omon start"
#
# If you do not specify the -a parameter, modify the file name for socket to an
appropriate value.
#
#     export MYSQL_UNIX_PORT=/var/lib/mysql/mysql.sock
#
# Modify the database name, the IP address, and etc. to appropriate values.
#
    clp_mysql4Omon mysqlwatch -d XXXX -a nnn.nnn.nnn.nnn
 else
    echo "ERROR_DISK from FAILOVER"
 fi
else
 echo "NO_CLP"
fi
echo "EXIT"
exit 0
```

10.15.2 stop.sh

```
#! /bin/sh
#****************************************
#*               stop.sh                *
#****************************************

if [ "$CLP_EVENT" = "START" ]
then
 if [ "$CLP_DISK" = "SUCCESS" ]
 then
    echo "NORMAL1"
    if [ "$CLP_SERVER" = "HOME" ]
```

```
    then
       echo "NORMAL2"
    else
       echo "ON_OTHER1"
    fi
    date +"%Y/%m/%d %T"
    echo "clp_mysql4Omon stop"
    clp_mysql4Omon mysqlwatch --stop
 else
    echo "ERROR_DISK from START"
 fi
elif [ "$CLP_EVENT" = "FAILOVER" ]
then
 if [ "$CLP_DISK" = "SUCCESS" ]
 then
    echo "FAILOVER1"
    if [ "$CLP_SERVER" = "HOME" ]
    then
       echo "FAILOVER2"
    else
       echo "ON_OTHER2"
    fi
    date +"%Y/%m/%d %T"
    echo "clp_mysql4Omon stop"
    clp_mysql4Omon mysqlwatch --stop
 else
    echo "ERROR_DISK from FAILOVER"
 fi
else
 echo "NO_CLP"
fi
echo "EXIT"
exit 0
```

10.16 For MySQL Monitoring (clp_mysql41mon)

Two templates, start.sh and stop.sh, are provided for the MySQL monitoring (clp_mysql41mon) scripts.
Modify and use them according to the operational environment and other requirements. Where modification is made are indicated in underlined boldface italics in the scripts in the following section.

10.16.1 start.sh

```
#! /bin/sh
#****************************************
#*              start.sh                *
#****************************************

if [ "$CLP_EVENT" = "START" ]
```

```
then
 if [ "$CLP_DISK" = "SUCCESS" ]
 then
    echo "NORMAL1"
    if [ "$CLP_SERVER" = "HOME" ]
    then
       echo "NORMAL2"
    else
       echo "ON_OTHER1"
    fi
    date +"%Y/%m/%d %T"
    echo "clp_mysql41mon start"
#
# If you do not specify the -a parameter, modify the file name for socket to an
# appropriate value.
#
#      export MYSQL_UNIX_PORT=/var/lib/mysql/mysql.sock
#
# Modify the database name, the IP address, and etc. to the appropriate values.
#
    clp_mysql41mon mysqlwatch -d XXXX -a nnn.nnn.nnn.nnn
 else
    echo "ERROR_DISK from START"
 fi
elif [ "$CLP_EVENT" = "FAILOVER" ]
then
 if [ "$CLP_DISK" = "SUCCESS" ]
 then
    echo "FAILOVER1"
    if [ "$CLP_SERVER" = "HOME" ]
    then
       echo "FAILOVER2"
    else
       echo "ON_OTHER2"
    fi
    date +"%Y/%m/%d %T"
    echo "clp_mysql41mon start"
#
# If you do not specify the -a parameter, modify the file name for socket to an
# appropriate value.
#
#      export MYSQL_UNIX_PORT=/var/lib/mysql/mysql.sock
#
# Modify the database name, the IP address, and etc. to appropriate values.
#
    clp_mysql41mon mysqlwatch -d XXXX -a nnn.nnn.nnn.nnn
 else
    echo "ERROR_DISK from FAILOVER"
 fi
else
 echo "NO_CLP"
fi
```

```
echo "EXIT"
exit 0
```

10.16.2 stop.sh

```
#! /bin/sh
#****************************************
#*               stop.sh                *
#****************************************

if [ "$CLP_EVENT" = "START" ]
then
 if [ "$CLP_DISK" = "SUCCESS" ]
 then
    echo "NORMAL1"
    if [ "$CLP_SERVER" = "HOME" ]
    then
       echo "NORMAL2"
    else
       echo "ON_OTHER1"
    fi
    date +"%Y/%m/%d %T"
    echo "clp_mysql41mon stop"
    clp_mysql41mon mysqlwatch --stop
 else
    echo "ERROR_DISK from START"
 fi
elif [ "$CLP_EVENT" = "FAILOVER" ]
then
 if [ "$CLP_DISK" = "SUCCESS" ]
 then
    echo "FAILOVER1"
    if [ "$CLP_SERVER" = "HOME" ]
    then
       echo "FAILOVER2"
    else
       echo "ON_OTHER2"
    fi
    date +"%Y/%m/%d %T"
    echo "clp_mysql41mon stop"
    clp_mysql41mon mysqlwatch --stop
 else
    echo "ERROR_DISK from FAILOVER"
 fi
else
 echo "NO_CLP"
fi
echo "EXIT"
exit 0
```

10.17 For Sybase Adaptive Server Enterprise(ASE)

Two templates, start.sh and stop.sh, are provided for the ASE scripts.
Modify and use them according to the operational environment and other requirements. Where modification is made are indicated in underlined boldface italics in the scripts in the following section.

10.17.1 start.sh

```
#! /bin/sh
#***************************************
#*    start.sh       *
#***************************************
if [ "$CLP_EVENT" = "START" ]
then
 if [ "$CLP_DISK" = "SUCCESS" ]
 then
    echo "NORMAL1"
    if [ "$CLP_SERVER" = "HOME" ]
    then
       echo "NORMAL2"
    else
       echo "ON_OTHER1"
    fi

    date +"%Y/%m/%d  %T"
    echo "Sybase ASE start"

 #
 # correct the start-faile path for the ASE data server
 # correct the install path of syb and the library pass
 #
    run_script=/opt/sybase/ASE-12_5/install/RUN_SERVER
    export LD_LIBRARY_PATH=/opt/sybase/OCS-12_5/lib
    export SYBASE=/opt/sybase

 #
 # correct the sybase account and the install path of ASE
 #
    su - sybase -c "export SYBASE=/opt/sybase; $run_script" > /dev/null 2>&1
&
    sleep 10

 else
    echo "ERROR_DISK from START"
 fi
elif [ "$CLP_EVENT" = "FAILOVER" ]
then
 if [ "$CLP_DISK" = "SUCCESS" ]
 then
```

```
      echo "FAILOVER1"
      if [ "$CLP_SERVER" = "HOME" ]
      then
         echo "FAILOVER2"
      else
         echo "ON_OTHER2"
      fi

      date +"%Y/%m/%d  %T"
      echo "Sybase ASE start"

    #
    # correct the start-faile path for the ASE data server
    # correct the install path of syb and the library pass
    #
      run_script=/opt/sybase/ASE-12_5/install/RUN_SERVER
      export LD_LIBRARY_PATH=/opt/sybase/OCS-12_5/lib
      export SYBASE=/opt/sybase

    #
    # correct the sybase account and the install path of ASE
    #
      su - sybase -c "export SYBASE=/opt/sybase; $run_script" > /dev/null 2>&1
&
      sleep 10

    else
      echo "ERROR_DISK from FAILOVER"
    fi
   else
    echo "NO_CLP"
   fi
   echo "EXIT"
   exit 0
```

10.17.2 stop.sh

```
   #! /bin/sh
   #***************************************
   #*     stop.sh        *
   #***************************************
   if [ "$CLP_EVENT" = "START" ]
   then
    if [ "$CLP_DISK" = "SUCCESS" ]
    then
      echo "NORMAL1"
      if [ "$CLP_SERVER" = "HOME" ]
      then
         echo "NORMAL2"
      else
```

```
        echo "ON_OTHER1"
      fi

      date +"%Y/%m/%d  %T"
      echo "Sybase ASE stop"
      #
      # correct the path of "isql" command
      #
      ISQL=/opt/sybase/OCS-12_5/bin/isql

      #
      # correct the path of shutdown script for ASE dataserver
      #
      shutdown=/home/sybase/shutdown
      sleep 10

      #
      # correct the server name of ASE data server
      # correct the sybase account and the install path of ASE
      #
      su - sybase -c "export SYBASE=/opt/sybase; $ISQL -S SERVER -U sa -P -i
$shutdown"
      sleep 10

   else
      echo "ERROR_DISK from START"
   fi
  elif [ "$CLP_EVENT" = "FAILOVER" ]
  then
   if [ "$CLP_DISK" = "SUCCESS" ]
   then
      echo "FAILOVER1"
      if [ "$CLP_SERVER" = "HOME" ]
      then
         echo "FAILOVER2"
      else
         echo "ON_OTHER2"
      fi

      date +"%Y/%m/%d  %T"
      echo "Sybase ASE stop"
      #
      # correct the path of "isql" command
      #
      ISQL=/opt/sybase/OCS-12_5/bin/isql

      #
      # correct the path of shutdown script for ASE dataserver
      #
      shutdown=/home/sybase/shutdown
      sleep 10
```

```
      #
      # correct the server name of ASE data server
      # correct the sybase account and the install path of ASE
      #
      su - sybase -c "export SYBASE=/opt/sybase; $ISQL -S SERVER -U sa -P -i
$shutdown"
      sleep 10

   else
      echo "ERROR_DISK from FAILOVER"
   fi
  else
   echo "NO_CLP"
  fi
  echo "EXIT"
  exit 0
```

10.17.3 A start file and shutdown file

You can use the start file created in installation when ASE is started. (In the example shown above, RUN_SERVER) The shutdown file is executed by accessing the database server when stopping. You can use any path or name for this file as long as the name does not contradict the full path name specified by start.sh/stop.sh.

Describe the following in each file. No template is provided for the start and shutdown files.

shutdown

```
shutdown with nowait
go
```

10.18For ASE Monitoring

Two templates, start.sh and stop.sh, are provided for the ASE monitoring scripts. Modify and use them according to the operational environment and other requirements. Where modification is made are indicated in underlined boldface italics in the scripts in the following section.

10.18.1 start.sh

```
#!/bin/sh
#**************************************
#*    start.sh                        *
#**************************************

#
# correct the install path of syb and the library pass
#
export SYBASE=/opt/sybase
export LD_LIBRARY_PATH=/opt/sybase/OCS-12_5/lib

if [ "$CLP_EVENT" = "START" ]
then
 if [ "$CLP_DISK" = "SUCCESS" ]
 then
    echo "NORMAL1"
    if [ "$CLP_SERVER" = "HOME" ]
    then
       echo "NORMAL2"
    else
       echo "ON_OTHER1"
    fi
    #
    # correct or add the prameters,server naem, databsyb name, etc...
    #
    clp_sybmon sybwatch -d XXXX -s SERVER -u user

 else
    echo "ERROR_DISK from START"
 fi
elif [ "$CLP_EVENT" = "FAILOVER" ]
then
 if [ "$CLP_DISK" = "SUCCESS" ]
 then
    echo "FAILOVER1"
    if [ "$CLP_SERVER" = "HOME" ]
    then
       echo "FAILOVER2"
    else
       echo "ON_OTHER2"
    fi
```

```
   #
   # correct or add the prameters,server naem, databsyb name, etc...
   #
   clp_sybmon sybwatch -d XXXX -s SERVER -u user

 else
   echo "ERROR_DISK from FAILOVER"
 fi
else
 echo "NO_CLP"
fi
echo "EXIT"
exit 0
```

10.18.2 stop.sh

```
#! /bin/sh
#****************************************
#*     stop.sh        *
#****************************************

#
# correct the library pass
#
export LD_LIBRARY_PATH=/opt/sybase/OCS-12_5/lib

if [ "$CLP_EVENT" = "START" ]
then
 if [ "$CLP_DISK" = "SUCCESS" ]
 then
   echo "NORMAL1"
   if [ "$CLP_SERVER" = "HOME" ]
   then
      echo "NORMAL2"
   else
      echo "ON_OTHER1"
   fi
   #
   # correct the watch id
   #
   clp_sybmon sybwatch --stop

 else
   echo "ERROR_DISK from START"
 fi
elif [ "$CLP_EVENT" = "FAILOVER" ]
then
 if [ "$CLP_DISK" = "SUCCESS" ]
 then
   echo "FAILOVER1"
   if [ "$CLP_SERVER" = "HOME" ]
```

```
   then
      echo "FAILOVER2"
   else
      echo "ON_OTHER2"
   fi
   #
   # correct the watch id
   #
   clp_sybmon sybwatch --stop

 else
   echo "ERROR_DISK from FAILOVER"
 fi
else
 echo "NO_CLP"
fi
echo "EXIT"
exit 0
```