Oracle® Solaris Cluster Data Service for MySQL Guide

Part No: E69331
October 2019

ORACLE

Oracle Solaris Cluster Data Service for MySQL Guide

Part No: E69331

Access to Oracle Support

Oracle customers that have purchased support have access to electronic support through My Oracle Support. For information, visit `http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info` or visit `http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs` if you are hearing impaired.

Référence: E69331

Accès aux services de support Oracle

Les clients Oracle qui ont souscrit un contrat de support ont accès au support électronique via My Oracle Support. Pour plus d'informations, visitez le site `http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info` ou le site `http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs` si vous êtes malentendant.

Contents

Tables

Examples

Using This Documentation

- **Overview** – Describes how to install and configure the HA for MySQL data service
- **Audience** – Technicians, system administrators, and authorized service providers
- **Required knowledge** – Advanced experience troubleshooting and replacing hardware

Product Documentation Library

Documentation and resources for this product and related products are available at http://docs.oracle.com/cd/E69294_01.

Feedback

Provide feedback about this documentation at http://www.oracle.com/goto/docfeedback.

♦ ♦ ♦ CHAPTER 1

Installing and Configuring HA for MySQL

This chapter explains how to install and configure HA for MySQL in the global cluster or in a zone cluster.

Note - As of the Oracle Solaris Cluster 4.4 release, support for HA for MySQL in a high-availability zone, also called a failover zone, is removed.

This chapter contains the following sections.

Installing and Configuring HA for MySQL

Table 1, "Task Map: Installing and Configuring HA for MySQL," on page 13 lists the tasks for installing and configuring HA for MySQL. Perform these tasks in the order that they are listed.

TABLE 1 Task Map: Installing and Configuring HA for MySQL

Task	For Instructions, Go To
1. Plan the installation.	"HA for MySQL Overview" on page 14

Task	For Instructions, Go To
	“Planning the HA for MySQL Installation and Configuration” on page 14
2. Install and configure MySQL.	“How to Install and Configure MySQL” on page 23
3. Verify installation and configuration.	“How to Verify the Installation and Configuration of MySQL” on page 25
4. Install HA for MySQL Packages.	“Installing the HA for MySQL Package” on page 26
5. Register and Configure HA for MySQL.	“How to Register and Configure HA for MySQL as a Failover Service” on page 27
6. Verify HA for MySQL Installation and Configuration.	“How to Verify the HA for MySQL Installation and Configuration” on page 37
7. Understand HA for MySQL fault monitor.	“Understanding the HA for MySQL Fault Monitor” on page 38
8. Debug HA for MySQL.	“Debugging the HA for MySQL” on page 39

HA for MySQL Overview

The MySQL software delivers a fast, multi-threaded, multi-user, and robust SQL (Structured Query Language) database server. MySQL Server is intended for mission-critical, heavy-load production systems as well as for embedding into mass-deployed software.

MySQL is freely available under the GNU General Public License. You can download it from `http://www.mysql.com`.

The HA for MySQL data service provides a mechanism for orderly startup and shutdown, fault monitoring and automatic failover of the MySQL service. The following MySQL components are protected by the HA for MySQL data service.

TABLE 2 Protection of Components

Component	Protected by
MySQL server	HA for MySQL

Planning the HA for MySQL Installation and Configuration

This section contains the information you need to plan your HA for MySQL installation and configuration.

MySQL and Oracle Solaris Zones

HA for MySQL is supported in Oracle Solaris Zones. Oracle Solaris Cluster offers the following concepts for Oracle Solaris Zones.

- HA for MySQL is supported to run in an Oracle Solaris Cluster zone cluster. Zone clusters are containers that are running after a reboot of the node. These containers form virtual clusters with their own node names.
- As of the Oracle Solaris Cluster 4.4 release, support for HA for MySQL in a high-availability zone, also called a failover zone, is removed.

Configuration Restrictions

This section provides a list of software and hardware configuration restrictions that apply to HA for MySQL only.

For restrictions that apply to all data services, see the *Planning and Administering Data Services for Oracle Solaris Cluster 4.4* and the *Oracle Solaris Cluster 4.4 Release Notes*.

Caution - Your data service configuration might not be supported if you do not observe these restrictions.

Restriction for the HA for MySQL Data Service Configuration

A single MySQL Server is to be configured as a failover resource only. A MySQL server is configured within a MySQL cluster, can be configured as a failover, scalable, or multiple-master resource. Scalable or multiple-master data services are typically deployed when you configure the MySQL server for MySQL Cluster.

Restrictions on the MySQL Configuration File

The MySQL configuration file, `my.cnf`, should be placed only in the MySQL Database directory. If `my.cnf` has to be placed in a local file system, then create a symbolic link from the MySQL Database directory.

The configuration file `my.cnf` should *not* be placed in the `/etc` directory, because it will override command line options.

Restrictions for the MySQL Configurations

The following configurations are supported with the HA for MySQL data service.

- Single or multiple MySQL instances in master configuration
- Single or multiple MySQL instances in slave configuration

Restrictions on the MySQL Database Directory

Regardless of which MySQL delivery method you have chosen, whether from `http://www.mysql.com` or from other source, each MySQL instance must have a unique Database directory. You can mount this Database directory as either a highly available local file system or cluster file system.

Tip - It is always a good practice to mount the cluster file systems with the `/global` prefix, and to mount the highly available local file systems with the `/local` prefix.

Configuration Requirements

The requirements in this section apply to HA for MySQL only. You must meet these requirements before you proceed with your HA for MySQL installation and configuration.

Caution - Your data service configuration might not be supported if you do not adhere to these requirements.

Components and Their Dependencies for HA for MySQL

MySQL components and their dependencies – You can configure HA for MySQL data service to protect a MySQL instance and its respective components. The following are the dependencies for a MySQL resource in the global cluster or in a zone cluster:

- `SUNW.HAStoragePlus` – Required only if the configuration uses a failover file system or uses file systems in a zone cluster. It is a `resource_offline_restart_dependency`.
- `SUNW.LogicalHostName`
- `SUNW.SharedAddress`

Registration and Configuration File for HA for MySQL

The MySQL component has a configuration file in `/opt/SUNWscmys/util`. This file enables you to prepare a MySQL instance to be registered with Oracle Solaris Cluster.

Within this file, the appropriate dependencies have been applied.

EXAMPLE 1 The `mysql_config` File

```
# more /opt/SUNWscmys/util/mysql_config
::::::::::::::
mysql_config
::::::::::::::
#
# CDDL HEADER START
#
# The contents of this file are subject to the terms of the
# Common Development and Distribution License (the License).
# You may not use this file except in compliance with the License.
#
# You can obtain a copy of the license at usr/src/CDDL.txt
# or http://www.opensolaris.org/os/licensing.
# See the License for the specific language governing permissions
# and limitations under the License.
#
# When distributing Covered Code, include this CDDL HEADER in each
# file and include the License file at usr/src/CDDL.txt.
# If applicable, add the following below this CDDL HEADER, with the
# fields enclosed by brackets [] replaced with your own identifying
# information: Portions Copyright [yyyy] [name of copyright owner]
#
# CDDL HEADER END
#

#
# Copyright (c) 2006, 2012, Oracle and/or its affiliates. All rights reserved.
#
#

#ident  "@(#)mysql_config.ksh   1.11    12/08/20"

# This file will be sourced in by mysql_register and the parameters
# listed below will be used.
#

# Where is mysql installed (BASEDIR)
```

```
MYSQL_BASE=

# Mysql admin-user for localhost (Default is root)
MYSQL_USER=

# Password for mysql admin user, if you do not want to disclose this password in
# a file, leave the MYSQL_PASSWD variable empty, or take it out.
MYSQL_PASSWD=

# Configured logicalhost. For scalable or multiple master resources leave it empty.
MYSQL_HOST=

# Specify a username for a faultmonitor user
FMUSER=

# Pick a password for that faultmonitor user, if you do not want to disclose this
# password in a file, leave the FMPASS variable empty, or take it out.
FMPASS=

# Socket name for mysqld ( Should be /tmp/<logical-host>.sock )
MYSQL_SOCK=

# Specify the physical hostname for the physical NIC that this logicalhostname
# belongs to for every node in the cluster this Resource group can get located on.
# If you use the mysql_geocontrol features to implement the MySQL replication as
# the replication protocol in Oracle Solaris Cluster geographic edition, specify all
# physical nodes of all clusters, specify at least all the nodes on both sites
# where the mysql databases can be hosted.
# IE: The logicalhost lh1 belongs to hme1 for physical-node phys-1 and
# hme3 for  physical-node phys-2. The hostname for hme1 is phys-1-hme0 and
# for hme3 on phys-2 it is phys-2-hme3.
# IE: MYSQL_NIC_HOSTNAME="phys-1-hme0 phys-2-hme3"
# IE: If two clusters are tied together by the mysql_geocontrol features, assuming the
# mysql database on cluster one belongs to cl1-phys1-hme0 and cl1-phys2-hme3, the
# mysql database on cluster two belongs to cl2-phys1-hme2 and cl2-phys2-hme4. Then the
# MYSQL_NIC_HOSTNAME variable needs to be set to:
# MYSQL_NIC_HOSTNAME="cl1-phys1-hme0 cl1-phys2-hme3 cl2-phys1-hme2 cl2-phys2-hme4"

MYSQL_NIC_HOSTNAME=

# Where are your databases installed, (location of my.cnf)
MYSQL_DATADIR=

# Is MySQL Cluster installed?
# Any entry here triggers the ndb engine check preparation. If no MySQL cluster should
 be
# checked, leave it empty.
NDB_CHECK=
```

Explanation of the `my.cnf` File

The `my.cnf` file – The HA for MySQL data service provides two sample `my.cnf` files, one sample file for a master configuration and one for a slave configuration. However, ensure that at least the following parameters are set.

Note - The `my.cnf` file is a important file within MySQL. Refer to the MySQL documentation for complete configuration information on the parameters that follow.

The MySQL `my.cnf` file in `[mysqld]`section in a master configuration:

- The `bind address` parameter must be set to the defined logical host's IP name.

  Note - Some of the MySQL parameters do not work with `bind address` if the name of the logical host is set. In these cases do not set the `bind address` parameter, or use the absolute address of the logical hostname.

- The **socket** parameter is defined as `/tmp/`*logical-host-ip-name*`.sock` in case of a failover deployment. If you deploy a scalable or multiple-master configuration, specify `/tmp/`*physical-hostname*`.sock`.
- The `binlog-ignore-db` parameter contains `sc3_test_database`, if `log-bin` option is being used.

The MySQL `my.cnf` file in `[mysqld]`section in a slave configuration:

- The `bind address` parameter must be set to the defined logical host's IP name.

  Note - Some MySQL builds do not work with names in the `bind address` parameter. In these cases do not set the `bind address` parameter, or use the absolute IP address of the logical hostname.

- The `socket` parameter is defined as `/tmp/`*logical-host-IP name*`.sock`.
- The `binlog-ignore-db` parameter contains `sc3_test_database` if `log-bin` option is being used.
- The `master-host` parameter is the hostname where the master instance resides.
- The `master-user` parameter is the username the slave will use for identification to the master.
- The `master-password` parameter is the password the slave will use for identification to the master.

Note - Beginning in MySQL server 4.4, the use of master-host, `master-user` and `master-password` in the `my.cnf` file is deprecated. Configure this information in the MySQL client with the `change_master` statement.

- The `master-info-file` parameter is the location of the file that remembers where MySQL left off on the master during the replication process. This file must be placed on a global file system or a failover file system.

Installing and Configuring MySQL

This section contains the procedures you need to install and configure MySQL in the global cluster or in a zone cluster.

References will be made to certain directories for MySQL. The following list shows common path names for these references. For more information, refer to the "Configuration Restrictions" on page 15 section.

- MySQL installed from `http://www.mysql.com` on a cluster file system, with a MySQL instance on a cluster file system:
  - **MySQL Base directory** – `/global/mysql`
  - **MySQL Database directory** – `/global/mysqldata`
- MySQL installed on a local file system, with `mysql` instances on a highly available local file system:
  - **MySQL Base directory** – `/usr/local/mysql`
  - **MySQL Database directory** – `/local/mysqldata`

Determine how MySQL will be deployed in an Oracle Solaris Cluster configuration.

- Determine how many MySQL instances will be deployed.
- Determine which cluster file system will be used by each MySQL instance.
- Determine where you will install MySQL, in the global cluster or in a zone cluster.

To install and configure MySQL in a global cluster or zone-cluster configuration, complete the following tasks:

- "Enabling MySQL" on page 21

- “How to Install and Configure MySQL” on page 23

You will find installation examples in the following appendixes:

- Appendix A, “Deployment Example: Installing MySQL in a Failover Configuration”
- Appendix B, “Deployment Example: Installing MySQL in a Scalable or Multiple-Master Configuration”

Enabling MySQL

Determine whether you need to install MySQL in a failover, scalable, or multiple-master configuration and follow the appropriate procedure:

- “How to Enable MySQL for a Failover Resource” on page 21
- “How to Enable MySQL to Run in a Scalable Configuration” on page 22
- “How to Enable MySQL to Run in a Multiple-Master Configuration” on page 22

▼ How to Enable MySQL for a Failover Resource

1. **On one of the nodes in the cluster that will host MySQL, become an administrator that provides `solaris.cluster.admin` authorization.**

2. **Register the `ORCL.mysql` and `SUNW.HAStoragePlus` resource types.**

```
# clresourcetype register ORCL.mysql SUNW.HAStoragePlus
```

3. **Create a failover resource group.**

```
# clresourcegroup create MySQL-failover-resource-group
```

4. **Create a resource for the MySQL disk storage.**

```
# clresource create \
-g MySQL-failover-resource-group \
-t SUNW.HAStoragePlus \
-p FilesystemMountPoints=MySQL-instance-mount-points MySQL-has-resource
```

5. **Create a resource for the MySQL logical hostname.**

```
# clreslogicalhostname \
-g MySQL-failover-resource-group \
```

```
-h MySQL-logical-hostname \
MySQL-lh-resource
```

6. **Enable the failover resource group that now includes the MySQL disk storage and Logical Hostname resources.**

```
# clresourcegroup online -M -n current-node MySQL-failover-resource-group
```

▼ How to Enable MySQL to Run in a Scalable Configuration

1. **On one of the nodes in the cluster that will host MySQL, become an administrator that provides `solaris.cluster.admin` authorization.**

2. **Register the `ORCL.mysql` resource type.**

```
# clresourcetype register ORCL.mysql
```

3. **Create a failover resource group.**

```
# clresource create SharedAddress-failover-resource-group
```

4. **Create the shared address resource.**

```
# clressharedaddress create \
-g SharedAddress-failover-resource-group \
-h MySQL-logical-hostname \
MySQL-lh-resource
```

5. **Enable the failover resource group that now includes the shared address resources.**

```
# clresourcegroup online -M -n current-node MySQL-failover-resource-group
```

6. **Create a resource group to run on at minimum two nodes.**

```
 # clresourcegroup create -p Maximum_primaries=2 \
-p Desired_primaries=2 MySQL-scalable-resource-group
```

▼ How to Enable MySQL to Run in a Multiple-Master Configuration

1. **On one of the nodes in the cluster that will host MySQL, become an administrator that provides `solaris.cluster.admin` authorization.**

2. **Register the `ORCL.mysql` resource type.**

```
# clresourcetype register ORCL.mysql
```

3. **Create a resource group to run on at least two nodes.**

```
# clresourcegroup create -p Maximum_primaries=2 \
-p Desired_primaries=2 MySQL-scalable-resource-group
```

How to Install and Configure MySQL

Before You Begin Perform this procedure to install and configure MySQL in the global cluster or in a zone cluster.

Note - If the HA for MySQL package (`ha-cluster/data-service/mysql`) was not installed during your initial Oracle Solaris Cluster installation, proceed to "Installing the HA for MySQL Package" on page 26 to install it on your cluster. Then return here to continue the installation and configuration of MySQL.

---

1. **Ensure that you are on the node of the global cluster or zone cluster where you have enabled your resource group.**

2. **Install MySQL onto all nodes within Oracle Solaris Cluster.**

   MySQL should be installed onto a cluster file system. For a discussion of the advantages and disadvantages of installing the software on local versus cluster file systems, see "Determining the Location of the Application Binaries" in *Planning and Administering Data Services for Oracle Solaris Cluster 4.4*.

   Download MySQL from `http://www.mysql.com`. If you intend to use local disks for the MySQL software, you will need to repeat this step on all nodes within Oracle Solaris Cluster.

3. **Create a `mysql-user` and `mysql-group` for MySQL on all nodes in the cluster that will run MySQL.**

   Create an entry in `/etc/group` on all nodes with Oracle Solaris Cluster.

```
# groupadd -g 1000 mysql
```

   Create an entry in `/etc/passwd` on all nodes within Oracle Solaris Cluster. This user should have a locked password.

```
# useradd -u 1000 -g 1000 -d /global/mysql -s /bin/sh mysql
```

4. **Change the owner and the group for MySQL binaries.**

If MySQL binaries are on all nodes, then repeat this step on every node.

```
# chown -R mysql:mysql /global/mysql
```

5. **Create your MySQL Database directory for your MySQL instance or instances.**

```
# mkdir MySQL-Database-directory
```

Note - Refer to "Configuration Restrictions" on page 15 for a description of the *MySQL Database directory* and to "Installing and Configuring MySQL" on page 20 for a list of common path names. For a multiple-master operating system scalable deployment, perform the remaining steps of this procedure on all nodes of the cluster that are about to host the MySQL database.

---

6. **Create the MySQL `my.cnf` file.**

The HA for MySQL data service provides three sample `my.cnf` files for MySQL. One sample configuration file is for a master configuration, one sample file is for a slave configuration, and one is for the server configuration in a MySQL cluster.

The content of `/opt/SUNWscmys/etc/my.cnf_sample_[master|slave|mysqld_cluster]`provides a sample MySQL configuration file that you can use to create your MySQL instance *MySQL-Database-directory*`/my.cnf`. You must still edit that file to reflect your configuration values.

```
# cp /opt/SUNWscmys/etc/my.cnf_sample_master \
MySQL-Database-directory/my.cnf
```

Note - If you are about to configure a multiple-master or a scalable configuration, set the socket directive in the `my.cnf` files to `socket=/tmp/`*physical-host*`.sock`.

---

7. **Bootstrap the MySQL instance.**

This creates the privilege tables `db`, `host`, `user`, `tables_priv`, and `columns_priv` in the `mysql database`, as well as the `func` table.

```
# cd MySQL-Base-directory
# ./bin/mysqld --initialize --datadir=MySQL-Database-directory
```

8. **Create a log file directory in *MySQL-Database-Directory*.**

```
# mkdir MySQL-Database-Directory/logs
```

9. **Create directories for your storage engines.**

```
# mkdir MySQL-Database-Directory/innodb
```

10. **Change owner and group for** *MySQL-Database-Directory***.**

```
# chown -R mysql:mysql MySQL-Database-Directory
```

11. **Change file permission for** *MySQL-Database-Directory*`/my.cnf`**.**

```
# chmod 644 MySQL-Database-Directory/my.cnf
```

Verifying the Installation and Configuration of MySQL

This section contains the procedure you need to verify the MySQL installation and configuration.

▼ How to Verify the Installation and Configuration of MySQL

This procedure does not verify that your application is highly available because you have not yet installed your data service.

Note - Before verifying the installation and configuration of MySQL, ensure that the logical hostname for the MySQL is available. You will need to complete the appropriate registration and configuration procedure in this document according to your zone type.

---

1. **Start the MySQL Server for this instance.**

```
# cd MySQL-Base-directory
# ./bin/mysqld --defaults-file=MySQL-Database-directory/my.cnf \
--basedir=MySQL-Base-directory \
--datadir=MySQL-Database-directory \
--user=mysql --pid-file=MySQL-Database-directory/mysqld.pid &
```

2. **Connect to the MySQL instance.**

```
# MySQL-Base-directory/bin/mysql -S /tmp/Logical-host.sock -uroot -p
Enter password:
```

Then enter the root password which is used at "Example: Enabling the MySQL Software to Run in the Cluster" on page 45.

3. **Stop the MySQL server instance.**

```
# kill -TERM $(cat MySQL-Database-directory/mysqld.pid)
```

Installing the HA for MySQL Package

If you did not install the HA for MySQL package during your initial Oracle Solaris Cluster installation, perform this procedure to install the package.

▼ How to Install the HA for MySQL Package

Perform this procedure on each cluster node where you want the HA for MySQL software to run.

1. **On the cluster node where you are installing the data service package, assume the `root` role.**

2. **Ensure that the data service package is available from the configured publisher and that the `solaris` and `ha-cluster` publishers are valid.**

```
# pkg list -a ha-cluster/data-service/mysql
# pkg publisher
PUBLISHER                             TYPE     STATUS   P  LOCATION
solaris                               origin   online   F  solaris-repository
ha-cluster                            origin   online   F  ha-cluster-repository
```

For information about setting the `solaris` publisher, see "Adding, Modifying, or Removing Package Publishers" in *Updating Systems and Adding Software in Oracle Solaris 11.4*.

Tip - Use the `-nv` options whenever you install or update to see what changes will be made, such as which versions of which packages will be installed or updated and whether a new BE will be created.

If you do not get any error messages when you use the `-nv` options, run the command again without the `-n` option to actually perform the installation or update. If you do get error messages, run the command again with more `-v` options (for example, `-nvv`) or more of the package FMRI pattern to get more information to help you diagnose and fix the problem. For troubleshooting information, see Appendix A, "Troubleshooting Package Installation and Update," in *Updating Systems and Adding Software in Oracle Solaris 11.4*.

3. **Install the HA for MySQL software package.**

```
# pkg install ha-cluster/data-service/mysql
```

4. **Verify that the package installed successfully.**

   ```
   $ pkg info ha-cluster/data-service/mysql
   ```

   Installation is successful if output shows that `State` is `Installed`.

5. **Perform any necessary updates to the Oracle Solaris Cluster software.**
   For instructions about updating your software, see Chapter 10, "Updating Software Packages" in *Updating Your Oracle Solaris Cluster 4.4 Environment*.

Registering and Configuring HA for MySQL

This section contains the procedures you need to configure HA for MySQL.

- "How to Register and Configure HA for MySQL as a Failover Service" on page 27
- "How to Add an HA for MySQL Resource as a Scalable or Multiple-Master Service" on page 32

▼ How to Register and Configure HA for MySQL as a Failover Service

This procedure assumes that you installed the data service packages during your initial Oracle Solaris Cluster installation.

If you did not install the HA for MySQL packages as part of your initial Oracle Solaris Cluster installation, go to "Installing the HA for MySQL Package" on page 26.

1. **On a node in the cluster that hosts MySQL, become an administrator that provides `solaris.cluster.admin` authorization.**

2. **Start the MySQL Server instance manually.**

   ```
   # cd MySQL-Base- directory

   # ./bin/mysqld --defaults-file=MySQL-Database-directory/my.cnf \
   --basedir=MySQL-Base-directory \
    --datadir=MySQL-Database-directory \
    --user=mysql \
   ```

```
--pid-file=MySQL-Database-directory/mysqld.pid &
```

3. **Configure the administrator password for the administrative user.**

```
# MySQL-Database-directory/bin/mysqladmin \
-S /tmp/logical-host.sock password 'admin-password'
```

4. **Add the administrative user for locally accessing a MySQL instance with a *MySQL logical-host-ip-name*.**

Note - If you want to access the MySQL instance only through the socket (`localhost`), omit this step.

When bootstrapping MySQL the command `mysqld --initialize` creates two administrative users, one belonging to `localhost` and one belonging to the node on which `mysqld --initialize` was executed.

Add an administrative user for every global-cluster node that runs this MySQL instance.

Note - If the node name and the hostname for the physical interface are different, use the hostname for the physical interface.

The following is an example for a MySQL instance.

```
# mysql -S /tmp/hahostix1.sock -uroot -padmin-password
mysql> use mysql;
mysql> GRANT ALL ON *.* TO 'root'@'clusterix2' IDENTIFIED BY 'rootpasswd';
mysql> GRANT ALL ON *.* TO 'root'@'clusterix1' IDENTIFIED BY 'rootpasswd';
mysql> exit;
```

Note - You must manually add `Grant_priv` to the administrative users. See the MySQL administration documentation.

The following is an example for a MySQL 4.*x* instance.

```
# mysql -S /tmp/hahostix1.sock -uroot -padmin-password
mysql> use mysql;
mysql> UPDATE user SET Grant_priv='Y' WHERE User='root' AND Host='clusterix1';
mysql> UPDATE user SET Grant_priv='Y' WHERE User='root' AND Host='clusterix2';
mysql> exit;
```

The following is an example for a MySQL 5.*x* instance.

```
# mysql -S /tmp/hahostix1.sock -uroot -padmin-password
mysql> use mysql;
```

```
mysql> GRANT ALL ON *.* TO 'root'@'clusterix2' IDENTIFIED BY 'rootpasswd';
mysql> GRANT ALL ON *.* TO 'root'@'clusterix1' IDENTIFIED BY 'rootpasswd';
mysql> exit;
```

The following is an example for a MySQL 8.*x* instance.

```
# mysql -S /tmp/hahostix1.sock -uroot -padmin-password
mysql> use mysql;
mysql> CREATE USER 'root'@'clusterix2' IDENTIFIED BY 'rootpasswd';
mysql> CREATE USER 'root'@'clusterix1' IDENTIFIED BY 'rootpasswd';
mysql> GRANT ALL ON *.* TO 'root'@'clusterix2' IDENTIFIED BY 'rootpasswd';
mysql> GRANT ALL ON *.* TO 'root'@'clusterix1' IDENTIFIED BY 'rootpasswd';
mysql> UPDATE user SET Grant_priv='Y' WHERE User='root' AND Host='clusterix1';
mysql> UPDATE user SET Grant_priv='Y' WHERE User='root' AND Host='clusterix2';
mysql> exit;
```

Note - If you experience any problems here, refer to the MySQL administration documentation.

5. Copy the MySQL configuration file to your home directory.

```
# cp /opt/SUNWscmys/util/mysql_config /home-dir
```

6. Create a `faultmonitor-user` and a `test-database` for the MySQL instance.

```
# cd home-dir
```

Edit the `mysql_config` file and follow the comments within that file.

```
#
# CDDL HEADER START
#
# The contents of this file are subject to the terms of the
# Common Development and Distribution License (the License).
# You may not use this file except in compliance with the License.
#
# You can obtain a copy of the license at usr/src/CDDL.txt
# or http://www.opensolaris.org/os/licensing.
# See the License for the specific language governing permissions
# and limitations under the License.
#
# When distributing Covered Code, include this CDDL HEADER in each
# file and include the License file at usr/src/CDDL.txt.
# If applicable, add the following below this CDDL HEADER, with the
# fields enclosed by brackets [] replaced with your own identifying
# information: Portions Copyright [yyyy] [name of copyright owner]
#
# CDDL HEADER END
```

```
#

#
# Copyright (c) 2006,2012, Oracle and/or its affiliates.  All rights reserved.
#

#ident  "@(#)mysql_config.ksh   1.11   12/08/20"

# This file will be sourced in by mysql_register and the parameters
# listed below will be used.
#

# Where is mysql installed (BASEDIR)
MYSQL_BASE=

# MySQL admin-user for localhost (Default is root)
MYSQL_USER=

# Password for mysql admin user, if you do not want to disclose this passowrd in
# a file, leave the MYSQL_PASSWD variable empty, or take it out.
MYSQL_PASSWD=

# Configured logicalhost. For scalable or multiple-master resources, leave it empty.
MYSQL_HOST=

# Specify a username for a faultmonitor user
FMUSER=

# Pick a password for that faultmonitor user, if you do not want to disclose this
# password in a file, leave the FMPASS variable empty, or take it out.
FMPASS=

# Socket name for mysqld ( Should be /tmp/<logical-host>.sock )
MYSQL_SOCK=

# Specify the physical hostname for the physical NIC that this logical hostname
# belongs to for every node in the cluster this resource group can be located on.
# If you use the mysql_geocontrol features to implement the MySQL replication as
# the replication protocol in Oracle Solaris Cluster geographic edition, specify all
# physical nodes of all clusters. Specify at least all the nodes on both sites
# where the MySQL databases can be hosted.
# IE: The logicalhost lh1 belongs to hme1 for physical-node phys-1 and
# hme3 for  physical-node phys-2. The hostname for hme1 is phys-1-hme0 and
# for hme3 on phys-2 it is phys-2-hme3.
# IE: MYSQL_NIC_HOSTNAME="phys-1-hme0 phys-2-hme3"
# IE: If two clusters are tied together by the mysql_geocontrol features, assuming the
# MySQL database on cluster one belongs to cl1-phys1-hme0 and cl1-phys2-hme3, the
# MySQL database on cluster two belongs to cl2-phys1-hme2 and cl2-phys2-hme4. Then the
```

```
# MYSQL_NIC_HOSTNAME variable needs to be set to:
# MYSQL_NIC_HOSTNAME="cl1-phys1-hme0 cl1-phys2-hme3 cl2-phys1-hme2 cl2-phys2-hme4"

MYSQL_NIC_HOSTNAME=

# Where are your databases installed? (location of my.cnf)
MYSQL_DATADIR=

# Is MySQL Cluster installed?
# Any entry here triggers the ndb engine check preparation. If no MySQL cluster should
 be
# checked, leave it empty.
NDB_CHECK=
```

The following is an example for a MySQL instance.

```
MYSQL_BASE=/global/mysql
MYSQL_USER=root
MYSQL_PASSWD=
MYSQL_HOST=hahostix1
FMUSER=fmuser
FMPASS=
MYSQL_SOCK=/tmp/hahostix1.sock
MYSQL_NIC_HOSTNAME="clusterix1 clusterix2"
MYSQL_DATADIR=/global/mysql-data
```

After editing `mysql_config`, you must run the `mysql_register` script.

```
# /opt/SUNWscmys/util/mysql_register -f home-dir/mysql_config
```

7. Stop the MySQL Server instance manually.

```
# kill -TERM `cat MySQL-Database-directry/mysqld.pid
```

8. Create and register MySQL as a failover data service.

a. Assume the `root` role on one node.

b. Encrypt the password of the fault-monitoring database user.

```
# clpstring create -t resource -b MySQL-resource  MySQL-resource
```

c. Register the MySQL resource.

```
# clresource create -d -g MySQL-failover-resource-group -t ORCL.mysql \
-p Mysql_basedir=MySQL-base-directory \
-p Mysql_datadir=MySQL-databse-directory \
-p Mysql_user=MySQL-OS-User \
```

```
-p Mysql_host=MySQL-logical-host \
-p Mysql_fmuser=MySQL-database-faul-monitoring-user \
-p Mysql_logdir=MySQL-database-log-directory \
-p Resource_dependencies=MySQL-logical-host-resource \
-p Resource_dependencies_offline_restart=MySQL-storage-resource \
MySQL-resource
```

9. Enable each MySQL resource.

Repeat this step for each MySQL instance, if multiple instances were created.

```
# clresource status
# clresource enable MySQL-resource
```

Example 2 Creating a Failover HA for MySQL Resource

The following example encrypts a password for the `MySQL-resource` resource, then creates and registers the failover resource.

```
# clpstring create -t resource -b MySQL-resource MySQL-resource

# clresource create -d -g MySQL-failover-resource-group \
-t ORCL.mysql \
-p Mysql_basedir=MySQL-base-directory \
-p Mysql_datadir=MySQL-database-directory \
-p Mysql_user=MySQL-OS-user \
-p Mysql_host=MySQL-logical-host \
-p Mysql_fmuser=MySQL-database-fault-monitoring-user \
-p Mysql_logdir=MySQL-database-log-directory \
-p Resource_dependencies=MySQL-logical-host-resource \
-p Resource_dependencies_offline_restart=MySQL-storage-resource \
MySQL-resource
```

▼ How to Add an HA for MySQL Resource as a Scalable or Multiple-Master Service

Before You Begin Ensure that you installed the data service packages during your initial Oracle Solaris Cluster installation. If you did not install the HA for MySQL packages as part of your initial Oracle Solaris Cluster installation, go to "Installing the HA for MySQL Package" on page 26.

Note - Perform the following steps, including stopping the MySQL server on every node that hosts the MySQL server.

1. **On the cluster node that hosts MySQL, become an administrator that provides `solaris.cluster.admin` RBAC authorization.**

2. **Start the MySQL Server instance manually.**

```
# cd MySQL-Base-directory
# ./bin/mysqld --defaults-file=MySQL-Database-directry/my.cnf \
--basedir=MySQL-Base-directory \
--datadir=MySQL-Database-directory --user=mysql \
--pid-file=MySQL-Database-directory/mysqld.pid &
```

3. **Configure the administrator password for the administrative user.**

```
# MySQL-Database-directory/bin/mysqladmin \
-S /tmp/Logicalhost.sock -uroot password 'admin-password'
```

4. **Add the administrative user for locally accessing a MySQL instance with a *MySQL-logical-host-ip-name*.**

   Add an administrative user for every global-cluster node that runs this MySQL instance.

   Note - If the node name and the hostname for the physical interface are different, use the hostname for the physical interface.

   The following is an example for a MySQL instance.

```
# mysql -S /tmp/clusterix2.sock -uroot -padmin-password
mysql> use mysql;
mysql> GRANT ALL ON *.* TO 'root'@'clusterix2' IDENTIFIED BY 'rootpasswd';
mysql> UPDATE user SET Grant_priv='Y' WHERE User='root' AND Host='clusterix2';
mysql> exit;
```

   The following is an example for a MySQL 5.x instance.

```
# mysql -S /tmp/clusterix2.sock -uroot -padmin-password
mysql> use mysql;
mysql> GRANT ALL ON *.* TO 'root'@'clusterix2' IDENTIFIED BY 'rootpasswd';
mysql> UPDATE user SET Grant_priv='Y' WHERE User='root' AND Host='clusterix2';
mysql> exit;
```

   The following is an example for a MySQL 8.x instance.

```
# mysql -S /tmp/clusterix2.sock -uroot -padmin-password
mysql> use mysql;
mysql> CREAT USER 'root'@'clusterix2' IDENTIFIED BY 'rootpasswd';
mysql> GRANT ALL ON *.* TO 'root'@'clusterix2' IDENTIFIED BY 'rootpasswd';
mysql> UPDATE user SET Grant_priv='Y' WHERE User='root' AND Host='clusterix2';
mysql> exit;
```

Note - If you experience any problems here, refer to the MySQL administration documentation.

5. **Copy the MySQL configuration files to your home directory.**

   # **cp /opt/SUNWscmys/util/mysql_config** */home-dir*

6. **Create a fault-monitor user and a test database for the MySQL instance.**

   # **cd** *home-dir*

7. **Edit the `mysql_config` file and follow the comments within that file.**

```
##
CDDL HEADER START
##
#The contents of this file are subject to the terms of the
# Common Development and Distribution License (the License).
# You may not use this file except in compliance with the License.
##
#You can obtain a copy of the license at usr/src/CDDL.txt
# or http://www.opensolaris.org/os/licensing.
# See the License for the specific language governing permissions
# and limitations under the License.
##
#When distributing Covered Code, include this CDDL HEADER in each
# file and include the License file at usr/src/CDDL.txt.
# If applicable, add the following below this CDDL HEADER, with the
# fields enclosed by brackets [] replaced with your own identifying
# information: Portions Copyright [yyyy] [name of copyright owner]
##
CDDL HEADER END
#
##
#Copyright 2012 Oracle Corporation. All rights reserved.
# Use is subject to license terms.
#
#ident "@(#)ds_template.sgm 1.41 12/10/12"
# This file will be sourced in by mysql_register and the parameters
# listed below will be used.
#
# Where is MySQL installed (BASEDIR)
MYSQL_BASE=
# MySQL admin-user for localhost (Default is root)
MYSQL_USER=
# Password for MySQL admin user
MYSQL_PASSWD=
# Configured logicalhost. For scalable or multiple-master resources leave it empty.
```

```
MYSQL_HOST=
# Specify a username for a faultmonitor user
FMUSER=
# Pick a password for that faultmonitor user
FMPASS=
# Socket name for mysqld ( Should be /tmp/logical-host.sock )
MYSQL_SOCK=
# Specify the physical hostname for the physical NIC that this logicalhostname
# belongs to for every node in the cluster this resource group can be located on.
# If you use the mysql_geocontrol features to implement the MySQL replication as
# the replication protocol in Oracle Solaris Cluster geographic edition, specify all
# physical nodes of all clusters. Specify at least all the nodes on both sites
# where the MySQL databases can be hosted.
# IE: The logicalhost lh1 belongs to hme1 for physical-node phys-1 and
# hme3 for physical-node phys-2. The hostname for hme1 is phys-1-hme0 and
# for hme3 on phys-2 it is phys-2-hme3.
# IE: MYSQL_NIC_HOSTNAME="phys-1-hme0 phys-2-hme3"
# IE: If two clusters are tied together by the mysql_geocontrol features, assuming the
# MySQL database on cluster one belongs to cl1-phys1-hme0 and cl1-phys2-hme3, the
# MySQL database on cluster two belongs to cl2-phys1-hme2 and cl2-phys2-hme4. Then the
# MYSQL_NIC_HOSTNAME variable needs to be set to:
# MYSQL_NIC_HOSTNAME="cl1-phys1-hme0 cl1-phys2-hme3 cl2-phys1-hme2 cl2-phys2-
#hme4"
MYSQL_NIC_HOSTNAME=
# Where are your databases installed, (location of my.cnf)
MYSQL_DATADIR=
# Is MySQL Cluster installed?
# Any entry here triggers the ndb engine check preparation. If no MySQL cluster should
 be
# checked, leave it empty.
NDB_CHECK=
```

The following is an example for a MySQL instance.

```
MYSQL_BASE=/global/mysql
MYSQL_USER=root
MYSQL_PASSWD=root
MYSQL_HOST=hahostix1
FMUSER=fmuser
FMPASS=
MYSQL_SOCK=/tmp/hahostix1.sock
MYSQL_NIC_HOSTNAME="clusterix1 clusterix2"
MYSQL_DATADIR=/global/mysql-data
NDB_CHECK=
```

If you want to monitor the `ndb` tables of a MySQL cluster, set `NDB_CHECK` to yes.

If you are about to prepare the database servers for a scalable or multiple-master configuration, set `MYSQL_SOCK=`*hostname*`.sock`.

8. **After editing `mysql_config`, run the `mysql_register` script.**

```
# /opt/SUNWscmys/util/mysql_register -f home-dir/mysql_config
```

9. **Stop the MySQL Server instance manually.**

```
# kill -TERM $(cat MySQL-Database-directory/mysqld.pid)
```

10. **After finishing the previous steps on every node in the cluster, continue in the global cluster unless you configure the MySQL server resource in a zone cluster.**

11. **Create and register MySQL as a scalable or multiple-master data service.**

    a. **Assume the `root` role on one node.**

    b. **Encrypt the password of the fault-monitoring database use.**

```
# clpstring create -t resource -b MySQL-resource  MySQL-resource
```

    c. **Register the MySQL resource.**

Note - If you register a multiple-master resource, omit the `Mysql_host`, `Scalable`, and `Port_list` properties.

To monitor the `ndb` tables, set the `Mysql_cluster` property to `true`.

```
# clresource create -d -g MySQL-failover-resource-group \
-t ORCL.mysql \
-p Mysql_basedir=MySQL-Base-directory \
-p Mysql_datadir=MySQL-database-directory \
-p Mysql_user=MySQL-OS-user \
-p Mysql_host=MySQL-scalable-address \
-p Mysql_fmuser=MySQL-database-fault-monitoring-user \
-p Mysql_logdir=MySQL-database-log-directory \
-p Scalable=true \
-p Port_list=mysql_port/tcp \
-p Resource_dependencies=MySQL-logical-host-resource \
-p Resource_dependencies_offline_restart=MySQL-storage-resource \
MySQL-resource
```

12. **Switch the resource group to a managed online state.**

```
# clresourcegroup online -eM MySQL-Scalable-resource-group
```

Example 3 Creating a Scalable HA for MySQL Resource

The following example encrypts a password for the `MySQL-resource` resource, then creates and registers the scalable resource in the `mysql-rg` resource group.

```
# clpstring create -t resource -b MySQL-resource MySQL-resource

# clresource create -d -g mysql-rg \
-t ORCL.mysql \
-p Mysql_basedir=/usr/local/mysql \
-p Mysql_datadir=/global/mysql-data \
-p Mysql_user=mysql \
-p Mysql_host=hahostix1 \
-p Mysql_fmuser=fmuser \
-p Mysql_logdir=/global/mysql-data/logs \
-p Scalable=true \
-p Port_list=3306/tcp \
-p Resource_dependencies=hahostix1 \
-p Resource_dependencies_offline_restart=mysql-has-res \
MySQL-resource
```

Verifying the HA for MySQL Installation and Configuration

This section contains the procedure you need to verify that you installed and configured your data service correctly.

▼ How to Verify the HA for MySQL Installation and Configuration

1. **Assume the `root` role on one of the nodes in the cluster that will host MySQL.**

2. **Ensure all the MySQL resources are online.**

   ```
   # cluster status
   ```

   For each MySQL resource that is not online, use the `clresource` command as follows.

   ```
   # clresource enable MySQL-resource
   ```

3. **Run the `clresourcegroup` command to switch the MySQL resource group to another cluster node, such as `node2`.**

```
# clresourcegroup -h node2  MySQL-failover-resource-group
```

Understanding the HA for MySQL Fault Monitor

This section describes the HA for MySQL fault monitor's probing algorithm or functionality, states the conditions, messages, and recovery actions associated with unsuccessful probing.

For conceptual information on fault monitors, see "Data Services Fault Monitors" in *Concepts for Oracle Solaris Cluster 4.4*.

Resource Properties

The HA for MySQL fault monitor uses the same resource properties as resource type `SUNW.gds`. Refer to the `SUNW.gds(7)` man page for a complete list of resource properties used.

Probing Algorithm and Functionality

The following are the MySQL probing steps:

- Sleeps for `Thorough_probe_interval`.
- Tries to connect to the MySQL instance, using the `mysqladmin` command with argument `ping`, with the fault-monitor user that is defined with the `Mysql_fmuser` property. If this fails, then the probe will restart the MySQL resource.
- Every 300 seconds, the probe also checks the following:
  - If the MySQL instance is a slave configuration, the probe checks whether the MySQL instance is connected to its master. If the Slave is not connected, the probe will write an error message to `syslog` and sets a status message.
  - Verifies that the probe can list all databases and tables, not the contents. If the probe receives any errors the probe writes an error message to `syslog`.
  - Conducts function tests on the defined test-database, Create Table, Insert into Table, Update Table, Delete from Table and Drop Table. If any of those operations fails, then the probe restarts the MySQL resource.
- If all MySQL processes have died, `pmf` interrupts the probe to immediately restart the MySQL resource.

- If the MySQL resource is repeatedly restarted and subsequently exhausts the `Retry_count` within the `Retry_interval` then a failover is initiated for the Resource Group onto another node if `Failover_enabled` is set to TRUE.

Debugging the HA for MySQL

This section describes how to activate and deactivate debugging for HA for MySQL. HA for MySQL has a file named `config` that enables you to activate debugging for MySQL resources. This file is in the `/opt/SUNWscmys/etc` directory.

▼ How to Activate Debugging for HA for MySQL

1. **Determine whether debugging for HA for MySQL is active.**

```
# grep daemon /etc/syslog.conf
*.err;kern.debug;daemon.notice;mail.crit        /var/adm/messages
*.alert;kern.err;daemon.err                             operator
#
```

If debugging is inactive, `daemon.notice` is set in the file `/etc/syslog.conf`.

2. **If debugging is inactive, edit the `/etc/syslog.conf` file to change `daemon.notice` to `daemon.debug`.**

3. **Confirm that debugging for HA for MySQL is active.**

If debugging is active, `daemon.debug` is set in the file `/etc/syslog.conf`.

```
# grep daemon /etc/syslog.conf
*.err;kern.debug;daemon.debug;mail.crit        /var/adm/messages
*.alert;kern.err;daemon.err                             operator
#
```

4. **Restart the `syslogd` daemon.**

```
# svcadm restart svc:/system/system-log:default
```

5. **Set the `Debug_level` property for the MySQL resource.**

```
# clresource set -p Debug_level=N MySQL-resource
```

The following are the valid values for the `Debug_level` property:

0	No debug information
1	Issue start and stop messages for the executed functions
2	Include a Korn shell trace on the console for start and stop messages of the functions

Example 4 Setting Debugging

The following example enables debugging for the `MySQL-resource` resource using debug level 2.

```
# clresource set -p Debug_level=2 MySQL-resource
```

How to Deactivate Debugging for HA for MySQL

1. **Determine whether debugging for HA for MySQL is active.**

```
# grep daemon /etc/          :operator
#
```

If debugging is active, `daemon.debug` is set in the file `/etc/syslog.conf`.

2. **If debugging is active, edit the `/etc/syslog.conf` file to change `daemon.debug` to `daemon.notice`.**

3. **Confirm that debugging for HA for MySQL is inactive.**

If debugging is inactive, `daemon.notice` is set in the file `/etc/syslog.conf`.

```
# grep daemon /etc/syslog.conf
*.err;kern.debug;daemon.notice;auth.none;mail.crit        /var/adm/messages
*.alert;kern.err;daemon.err                                 operator
#
```

4. **Restart the `syslogd` daemon.**

```
# svcadm restart svc:/system/system-log:syslog
```

5. **Set the `Debug_level` property for the MySQL resource to `0`.**

```
# clresource set -p Debug_level=0 MySQL-resource
```

♦ ♦ ♦ APPENDIX A

Deployment Example: Installing MySQL in a Failover Configuration

This appendix presents a complete example of how to install and configure the MySQL application and data service in the global cluster or in a zone cluster. It presents a simple two-node cluster configuration. If you need to install the application in any other configuration, refer to the general purpose procedures presented elsewhere in this manual.

For an example of MySQL installation in scalable or multiple-master configuration, see Appendix B, “Deployment Example: Installing MySQL in a Scalable or Multiple-Master Configuration”.

Target Cluster Configuration

This example uses a two-node cluster with the following node names:

- `phys-schost-1` (a physical node, which owns the file system)
- `phys-schost-2` (a physical node)

This configuration also uses the logical host name `ha-host-1`.

Software Configuration

This deployment example uses the following software products and versions:

- Oracle Solaris 11.4 software for SPARC or x86 platforms
- Oracle Solaris Cluster 4.4 core software
- Oracle HA for MySQL
- MySQL - Version 5.7.12

This example assumes that you have already installed and established your cluster. It illustrates installation and configuration of the data service application only.

Assumptions

The instructions in this example were developed with the following assumptions:

- **Shell environment:** All commands and the environment setup in this example are for the bash shell environment. If you use a different shell, replace any bash shell-specific information or instructions with the appropriate information for you preferred shell environment.
- **User login:** Unless otherwise specified, perform all procedures by assuming a role that provides `solaris.cluster.admin`, `solaris.cluster.modify`, and `solaris.cluster.read` authorization.

Installing and Configuring MySQL on Local Storage

The tasks you must perform to install and configure MySQL in the global cluster or in a zone cluster are as follows:

▼ Example: Preparing the Cluster for MySQL

1. **Install and configure the cluster as instructed in the *Installing and Configuring an Oracle Solaris Cluster 4.4 Environment*.**

   Install the following cluster software components on both nodes.

   - Oracle Solaris Cluster core software
   - Oracle Solaris Cluster data service for MySQL

2. **Beginning on the node that owns the file system, add the `mysql` user.**

```
phys-schost-1# groupadd -g 1000 mysql
phys-schost-1# useradd -g 1000 -d /global/mnt3/mysql -m -s /bin/ksh mysql
phys-schost-2# groupadd -g 1000 mysql
phys-schost-2# useradd -g 1000 -d /global/mnt3/mysql -m -s /bin/ksh mysql
```

Example: Configuring Cluster Resources for MySQL

1. **Register the necessary data types on one of the nodes.**

   ```
   phys-schost-1# clresourcetype register ORCL.mysql SUNW.HAStoragePlus
   ```

2. **Create the MySQL resource group.**

   ```
   phys-schost-1# clresourcegroup create RG-MYS
   ```

3. **Create the logical host.**

   ```
   phys-schost-1# clreslogicalhostname create -g RG-MYS ha-host-1
   ```

4. **Create the `HAStoragePlus` resource in the `RG-MYS` resource group.**

   ```
   phys-schost-1# clresource create -g RG-MYS -t SUNW.HAStoragePlus -p AffinityOn=TRUE \
   -p FilesystemMountPoints=/global/mnt3,/global/mnt4 RS-MYS-HAS
   ```

5. **Enable the resource group.**

   ```
   phys-schost-1# clresourcegroup online -M RG-MYS
   ```

Example: Installing and Bootstrapping the MySQL Software on Local Storage

These steps illustrate how to install the MySQL software in the default directory `/usr/local/mysql`. If only one node is mentioned, it needs to be the node where your resource group is online.

1. **Install the MySQL binaries on both nodes.**

   ```
   phys-schost-1# cd /usr/local
   phys-schost-1# tar xvf mysql-max-5.7.12-solaris11-architecture_64.tar.gz
   ```

```
phys-schost-1# ln -s mysql-max-5.7.12-solaris11-architecture_64 mysql
```

2. **Change the ownership of the MySQL binaries on both nodes.**

```
phys-schost-1# chown -R mysql:mysql /usr/local/mysql
```

3. **Create your database directories.**

```
phys-schost-1# mkdir -p /global/mnt3/mysql-data/logs
phys-schost-1# mkdir /global/mnt3/mysql-data/innodb
```

4. **Bootstrap MySQL.**

```
phys-schost-1# cd /usr/local/mysql
phys-schost-1# ./bin/mysqld --initialize --datadir=/global/mnt3/mysql-data
```

5. **Create your `my.cnf` `config-file` in `/global/mnt3/mysql-data`.**

```
phys-schost-1# cat > /global/mnt3/mysql-data/my.cnf << EOF
[mysqld]
server-id=1
#port=3306
# 10.18.5.1 is the address of the logical host
bind-address=10.18.5.1 # this is the address of the logical host
socket=/tmp/hahostix1.sock
log=/global/mnt3/mysql-data/logs/log1
log-bin=/global/mnt3/mysql-data/logs/bin-log
binlog-ignore-db=sc3_test_database
log-slow-queries=/global/mnt3/mysql-data/logs/log-slow-queries
#log-update=/global/mnt3/mysql-data/logs/log-update

# Innodb
#skip-innodb
innodb_data_home_dir = /global/mnt3/mysql-data/innodb
innodb_data_file_path = ibdata1:10M:autoextend
innodb_log_group_home_dir = /global/mnt3/mysql-data/innodb
innodb_log_arch_dir = /global/mnt3/mysql-data/innodb
# You can set .._buffer_pool_size up to 50 - 80 %
# of RAM but beware of setting memory usage too high
set-variable = innodb_buffer_pool_size=50M
set-variable = innodb_additional_mem_pool_size=20M
# Set .._log_file_size to 25 % of buffer pool size
set-variable = innodb_log_file_size=12M
set-variable = innodb_log_buffer_size=4M
innodb_flush_log_at_trx_commit=1
set-variable = innodb_lock_wait_timeout=50

# Replicating Slave
#server-id=2
```

```
#master-host=administerix
#master-user=repl
#master-password=repl
#master-info-file=/global/mnt3/mysql-data/logs/master.info
```

6. **Change the ownership of the MySQL data directory.**

```
phys-schost-1# chown -R mysql:mysql /global/mnt3/mysql-data
```

7. **Change the permission of the `my.cnf` file.**

```
phys-schost-1# chmod 644 /global/mnt3/mysql-data/my.cnf
```

▼ Example: Modifying the MySQL Configuration Files

1. **Copy the MySQL configuration file from the agent directory to its deployment location.**

```
phys-schost-1# cp /opt/SUNWscmys/util/mysql_config /global/mnt3
```

2. **Add this cluster's information to the `mysql_config` configuration file.**

The following listing shows the relevant file entries and the values to assign to each entry.

```
…
MYSQL_BASE=/usr/local/mysql
MYSQL_USER=root
MYSQL_PASSWD=
MYSQL_HOST=ha-host-1
FMUSER=fmuser
FMPASS=
MYSQL_SOCK=/tmp/ha-host-1.sock
MYSQL_NIC_HOSTNAME="phys-schost-1 phys-schost-2"
MYSQL_DATADIR=/global/mnt3/mysql-data
NDB_CHECK=
```

▼ Example: Enabling the MySQL Software to Run in the Cluster

1. **Start the MySQL database manually on the node where the resource group is online.**

```
phys-schost-1# cd /usr/local/mysql
phys-schost-1# ./bin/mysqld --defaults-file=/global/mnt3/mysql-data/my.cnf \
--basedir=/usr/local/mysql --datadir=/global/mnt3/mysql-data \
--pid-file=/global/mnt3/mysql-data/mysqld.pid \
--user=mysql >> /global/mnt3/mysql-data/logs/ha-host-1.log 2>&1 &
```

2. **Set the password for `localhost` in MySQL to root.**

```
phys-schost-1# /usr/local/mysql/bin/mysqladmin -S /tmp/ha-host-1.sock -uroot \
password 'root'
```

3. **Add an administrative user in the MySQL database for the logical host.**

```
phys-schost-1# /usr/local/mysql/bin/mysql -S /tmp/ha-host-1.sock -uroot -proot
mysql> use mysql;
mysql> GRANT ALL ON *.* TO 'root'@'phys-schost-1' IDENTIFIED BY 'root';
mysql> GRANT ALL ON *.* TO 'root'@'phys-schost-2' IDENTIFIED BY 'root';
mysql> UPDATE user SET Grant_priv='Y' WHERE User='root' AND Host='phys-schost-1';
mysql> UPDATE user SET Grant_priv='Y' WHERE User='root' AND Host='phys-schost-2';
mysql> exit
```

4. **Prepare the Oracle Solaris Cluster specific test database.**

```
phys-schost-1# ksh /opt/SUNWscmys/util/mysql_register \
-f /global/mnt3/mysql_config
```

   a. **Enter `root` when you are asked for the MySQL admin password.**

      You set it to `root` in Step 3.

   b. **Enter a password when you are prompted for the fault monitor user password.**

5. **Stop the MySQL database.**

```
phys-schost-1# kill -TERM $(cat /global/mnt3/mysql-data/mysqld.pid)
```

6. **Encrypt the fault monitor user password on one node to run the MySQL database.**

```
phys-schost-1# clpstring create -t resource -b RS-MYS RS-MYS
```

   Enter the same password you provided in Step 4 when you are prompted for the fault monitoring user password.

7. **Register the MySQL resource.**

```
phys-schost-1# clresource create -d -g RG-MYS \
-t ORCL.mysql \
-p Mysql_basedir=/usr/local/mysql \
-p Mysql_datadir=/global/mnt3/mysql-data \
-p Mysql_user=mysql \
-p Mysql_host=ha-host-1 \
-p Mysql_fmuser=fmuser \
-p Mysql_logdir=/global/mnt3/mysql-data/logs \
-p Resource_dependencies=RS-MYS-HAS,ha-host-1 \
RS-MYS
```

♦ ♦ ♦ APPENDIX B

Deployment Example: Installing MySQL in a Scalable or Multiple-Master Configuration

This appendix presents a complete example of how to install and configure the MySQL application and data service in a scalable or multiple-master configuration, in the global cluster or in a zone cluster. It presents a simple two-node cluster configuration. If you need to install the application in any other configuration, refer to the general-purpose procedures presented elsewhere in this manual.

For an example of MySQL installation in a failover configuration, see Appendix A, “Deployment Example: Installing MySQL in a Failover Configuration”.

Target Cluster Configuration

This example uses a two-node cluster with the following node names:

- `phys-schost-1` (a physical node)
- `phys-schost-2` (a physical node)

The scalable configuration also uses the shared address host name `ha-host-1`.

Software Configuration

This deployment example uses the following software products and versions:

- Oracle Solaris 11.4 software for SPARC or x86 platforms
- Oracle Solaris Cluster 4.4 core software
- HA for MySQL

- MySQL - Version 5.7.12

This example assumes that you have already installed and established your cluster. It illustrates installation and configuration of the data service application only.

Assumptions

The instructions in this example make the following assumptions:

- Shell environment: All commands and the environment setup in this example are for the bash shell environment. If you use a different shell, replace any bash shell-specific information or instructions with the appropriate information for you preferred shell environment.
- User login: Unless otherwise specified, perform all procedures by assuming a role that provides `solaris.cluster.admin`, `solaris.cluster.modify`, and `solaris.cluster.read` RBAC authorization.

Installing and Configuring MySQL on Local Storage

These instructions assume that you are installing the MySQL software as the `mysql` user on the local storage.

The tasks you must perform to install and configure MySQL are as follows:

- “Example: How to Prepare the Cluster for MySQL” on page 50
- “Example: How to Configure MySQL in a Scalable Resource Group” on page 51
- “Example: How to Configure MySQL in a Multiple-Master Resource Group” on page 52
- “Example: How to Install and Bootstrap the MySQL Software on Local Storage” on page 52
- “Example: How to Modify the MySQL Configuration Files” on page 54
- “Example: How to Enable the MySQL Software to Run in the Cluster” on page 54

▼ Example: How to Prepare the Cluster for MySQL

Before You Begin Install and configure the cluster as instructed in the *Installing and Configuring an Oracle Solaris Cluster 4.4 Environment*.

1. **Install the following cluster software components on both nodes.**

   - Oracle Solaris Cluster core software
   - Oracle Solaris Cluster data service for MySQL

2. **Add the `mysql` group and user.**

```
phys-schost-1# groupadd -g 1000 mysql
phys-schost-1# useradd -g 1000 -d /local/mysql -m -s /bin/ksh mysql

phys-schost-2# groupadd -g 1000 mysql
phys-schost-2# useradd -g 1000 -d /local/mysql -m -s /bin/ksh mysql
```

Example: How to Configure MySQL in a Scalable Resource Group

This procedure describes how to configure MySQL in a scalable resource group. If you need to configure MySQL in a multiple-master group, see “Example: How to Configure MySQL in a Multiple-Master Resource Group” on page 52.

1. **Register the necessary data types on one of the nodes.**

```
phys-schost-1# clresourcetype register ORCL.mysql
```

2. **Create the failover resource group for the shared address resource.**

```
phys-schost-1# clresourcegroup create RG-ACCESS
```

3. **Create the shared address resource.**

```
phys-schost-1# clressharedaddress create ha-host-1
```

4. **Create the MySQL resource group.**

```
phys-schost-1# clresourcegroup create RG-MYSQL
```

5. **Enable the resource groups.**

```
phys-schost-1# clresourcegroup online -M RG-MYSQL
phys-schost-1# clresourcegroup online -M RG-ACCESS
```

Example: How to Configure MySQL in a Multiple-Master Resource Group

This procedure describes how to configure MySQL in a multiple-master resource group. If you need to configure MySQL in a scalable group, see "Example: How to Configure MySQL in a Scalable Resource Group" on page 51.

1. **Register the necessary data types on one of the nodes.**

```
phys-schost-1# clresourcetype register ORCL.mysql
```

2. **Create the MySQL resource group.**

```
phys-schost-1# clresourcegroup create -p Maximum_primaries=2 \
-p desired_primaries=2 RG-MYS
```

3. **Enable the resource groups.**

```
phys-schost-1# clresourcegroup online -M RG-MYS
```

Example: How to Install and Bootstrap the MySQL Software on Local Storage

These steps illustrate how to install the MySQL software in the default directory `/usr/local/mysql`.

1. **Install the MySQL binaries on both nodes.**

```
phys-schost-1# cd /usr/local
phys-schost-1# tar xvf mysql-max-5.0.12-solaris11-architecture_64.tar.gz
phys-schost-1# ln -s mysql-max-5.0.12-solaris11-architecture_64 mysql
```

2. **Change the ownership of the MySQL binaries on both nodes.**

```
phys-schost-1# chown -R mysql:mysql /usr/local/mysql
```

3. **Create your database directories.**

```
phys-schost-1# mkdir -p /local/mysql-data/logs
phys-schost-1# mkdir /local/mysql-data/innodb
```

4. **Bootstrap MySQL on both nodes.**

```
phys-schost-1# cd /usr/local/mysql
phys-schost-1# ./bin/mysqld --initialize --datadir=/local/mysql-data
```

5. **Create your `my.cnf config-file in /local/mysql-data`.**

```
phys-schost-1# cat > /local/mysql-data/my.cnf << EOF
[mysqld]
server-id=1
#port=3306
# 10.18.5.1 is the address of the logical host
#bind-address=10.18.5.1 #
socket=/tmp/phys-schost-1 .sock
log=/local/mysql-data/logs/log1
log-bin=/local/mysql-data/logs/bin-log
binlog-ignore-db=sc3_test_database
log-slow-queries=/local/mysql-data/logs/log-slow-queries
#log-update=/local/mysql-data/logs/log-update
# Innodb
#skip-innodb
innodb_data_home_dir = /local/mysql-data/innodb
innodb_data_file_path = ibdata1:10M:autoextend
innodb_log_group_home_dir = /local/mysql-data/innodb
innodb_log_arch_dir = /local/mysql-data/innodb
# You can set .._buffer_pool_size up to 50 - 80 %
# of RAM but beware of setting memory usage too high
set-variable = innodb_buffer_pool_size=50M
set-variable = innodb_additional_mem_pool_size=20M
# Set .._log_file_size to 25 % of buffer pool size
set-variable = innodb_log_file_size=12M
set-variable = innodb_log_buffer_size=4M
innodb_flush_log_at_trx_commit=1
set-variable = innodb_lock_wait_timeout=50
# Replication of Slave
#server-id=2
#master-host=administerix
#master-user=repl
#master-password=repl
#master-info-file=/local/mysql-data/logs/master.info
# MySQL 4.x
#relay-log=/local/mysql-data/logs/slave-bin.log
#relay-log-info-file=/local/mysql-data/logs/slave-info
```

Note - If you are running multiple slaves, ensure that the parameters `server-id` and `socket` differ for all the MySQL resources.

---

6. **Change the ownership of the MySQL data directory.**

```
phys-schost-1# chown -R mysql:mysql /local/mysql-data
```

7. **Change the permission of the `my.cnf` file.**

```
phys-schost-1# chmod 644 /local/mysql-data/my.cnf
```

Example: How to Modify the MySQL Configuration Files

1. **Copy the MySQL configuration file from the agent directory to its deployment location on both nodes.**

```
phys-schost-1# cp /opt/SUNWscmys/util/mysql_config /local
```

2. **Add this cluster's information to the `mysql_config` configuration file on both nodes.**

   The following listing shows the relevant file entries for a scalable service and the values to assign to each entry.

```
…
MYSQL_BASE=/usr/local/mysql
MYSQL_USER=root
MYSQL_PASSWD=
MYSQL_HOST= phys-schost-1
FMUSER=fmuser
FMPASS=
MYSQL_SOCK=/tmp/phys-schost-1.sock
MYSQL_NIC_HOSTNAME="phys-schost-1"
MYSQL_DATADIR=/local/mysql-data
NDB_CHECK=
```

Note - Make sure that the parameter `MYSQL_HOST` reflects the physical hostname on both nodes.

Example: How to Enable the MySQL Software to Run in the Cluster

1. **Start the MySQL database manually on both nodes.**

```
phys-schost-1# cd /usr/local/mysql
```

```
phys-schost-1# ./bin/mysqld --defaults-file=/local/mysql-data/my.cnf \
--basedir=/usr/local/mysql --datadir=/local/mysql-data \
--pid-file=/local/mysql-data/mysqld.pid \
--user=mysql >> /local/mysql-data/logs/phys-schost-1.log 2>&1
```

Note - Make sure to change `phys-schost-1` to `phys-schost-2` on the second node.

2. **Set the password for `localhost` in MySQL to root on both nodes.**

```
phys-schost-1# /usr/local/mysql/bin/mysqladmin -S /tmp/phys-schost-1.sock -uroot \
 password 'root'
```

Note - Make sure to change `phys-schost-1` to `phys-schost-2` on the second node.

3. **Add an administrative user in the MySQL database for the physical host on both nodes.**

```
phys-schost-1# /usr/local/mysql/bin/mysql -S /tmp/phys-schost-1.sock -uroot -proot
mysql> use mysql;
mysql> GRANT ALL ON *.* TO 'root'@'phys-schost-1' IDENTIFIED BY 'root';
mysql> UPDATE user SET Grant_priv='Y' WHERE User='root' AND Host='phys-schost-1';
mysql> exit
```

Note - Make sure to change `phys-schost-1` to `phys-schost-2` on the second node.

4. **Prepare the Oracle Solaris Cluster specific test database on both nodes.**

```
phys-schost-1# ksh /opt/SUNWscmys/util/mysql_register -f /local/mysql_config
```

5. **Stop the MySQL database.**

```
phys-schost-1# kill -TERM $(cat /local/mysql-data/mysqld.pid)
```

6. **Encrypt the fault monitor user password on one node to run the MySQL database.**

```
phys-schost-1# clpstring create -t resource -b RS-MYSQL RS-MYSQL
```

Enter the same password you provided in Step 4 when you are prompted for the fault monitoring user password.

7. **Register the MySQL resource.**

   - **For a scalable resource, use the following command:**

```
phys-schost-1# clresource create -d -g RG-MYSQL \
-t ORCL.mysql \
-p Mysql_basedir=/usr/local/mysql \
-p Mysql_datadir=/local/mysql-data \
-p Mysql_user=mysql \
-p Mysql_host=ha-host-1 \
-p Mysql_fmuser=fmuser \
-p Mysql_logdir=/local/mysql-data/logs \
-p Scalable=true \
-p Port_list=3306/tcp \
-p Resource_dependencies=ha-host-1 \
RS-MYSQL
```

- **For a multiple-master resource, use the following command:**

```
phys-schost-1# clresource create -d -g RG-MYSQL \
-t ORCL.mysql \
-p Mysql_basedir=/usr/local/mysql \
-p Mysql_datadir=/local/mysql-data \
-p Mysql_user=mysql \
-p Mysql_fmuser=fmuser \
-p Mysql_logdir=/local/mysql-data/logs \
RS-MYSQL
```

8. **Enable the resource.**

```
phys-schost-1# clresource enable RS-MYSQL
```

♦ ♦ ♦ APPENDIX C

HA for MySQL Extension Properties

The ORCL.mysql resource type inherits the algorithms and properties from the `ORCL.gds` resource type. Only the additional extension properties for the ORCL.mysql resource type are described in this Appendix. For a description of all other properties, see *Developing Agents With the Generic Data Service (GDS)*.

`ORCL.mysql` Extension Properties

The additional extension properties of this resource type are as follows:

`Debug_level`

This property indicates the level to which debug messages for the MySQL resources are logged. When the debug level is increased, more debug messages are written to the system log `/var/adm/messages` as follows:

0	No debug messages
1	Function Begin and End messages
2	All debug messages and function Begin and End messages

You can specify a different value of the `Debug_level` extension property for each node that can master the resource.

Data type	Integer
Default	0
Range	0-2
Tunable	Any time

`Mysql_basedir`

This property specifies the base directory, which is the directory where the MySQL binaries are installed. This property is mandatory.

Data type String

Default None defined

Range Not applicable

Tunable When disabled

`Mysql_check`

This property specifies the flag if the probe should do an intensive check for the MySQL objects. This property is mandatory.

Data type Boolean

Default True

Range Not applicable

Tunable When disabled

`Mysql_cluster`

This property specifies the flag if the probe should do an intensive check for the MySQL `ndb` objects. This property is mandatory.

Data type Boolean

Default False

Range Not applicable

Tunable When disabled

`Mysql_datadir`

This property specifies the directory where the data is stored. This property is mandatory.

Data type String

Default None defined

Range Not applicable

Tunable When disabled

`Mysql_fmuser`

This property specifies the MySQL database user for fault monitoring. This property is mandatory.

Data type String

Default None defined

Range Not applicable

Tunable When disabled

`Mysql_host`

This property specifies the IP alias to use for the MySQL probes.

Data type String

Empty string None defined

Range Not applicable

Tunable When disabled

`Mysql_logdir`

This property specifies the directory where the MySQL logs are stored. This property is mandatory.

Data type String

Default None defined

Range Not applicable

Tunable When disabled

`Mysql_user`

This property specifies the MySQL operating system user. This property is mandatory.

Data type String

Default	None defined
Range	Not applicable
Tunable	When disabled

Index

O

P

R

S