



An Oracle Technical White Paper
October 2011

Installation Guide for Single Click Configurations of Oracle's MySQL 5 with Oracle Linux 5 for Sun Fire x86 Servers

Introduction	1
Installation Overview	2
Accessing the Installation System	3
Required Pre-Requisites Check.....	3
Hardware Pre-Requisites:.....	3
Software Pre-Requisites:	5
Installation Process	7
Obtaining Oracle's MySQL Installation Media	7
Installing MySQL Server and Client	7
Post-Installation Configuration.....	8
Starting MySQL Server.....	8
Securing MySQL Server for Production Environment	8
Setting MySQL Root User Password.....	9
Starting the MySQL Client	9
Setting Up Replication	9
Setting Replication Configuration on the Master Server.....	9
Setting Replication Configuration on the Slave Server.....	10
Creating a User for Replication.....	11
Obtaining the Replication Master Binary Log Coordinates	12
Setting Up Replication with New Master and Slaves.....	13
Setting Up Replication with an Existing MySQL Database Server	14
Validating MySQL Replication Configuration	16
Conclusion	17
Appendix	18
A.1 Increasing Swap Space Process.....	18
A.2 Installing Packages	20
A.3 Access Rights and Other Information for my.cnf Option Files	21

Introduction

Oracle has developed industry leading applications and operating systems that can only be fully utilized if they are hosted on Oracle hardware and managed by Oracle management tools. Oracle is the only company that can offer tightly integrated x86 clustered systems from applications to disk. Oracle's integrated components greatly simplify the deployment, management, and support for x86 infrastructures to deliver unmatched performance and scalability with superior TCO.

This white paper goes through the process of preparing an Oracle Linux operating system for installation of Oracle's MySQL Enterprise Edition on an Oracle's Sun Fire x86 two-socket rack server, as well as the installation steps required to get MySQL up and running quickly based on Oracle best practices. By following the instructions in this white paper, any user can have their Oracle Sun Fire x86 servers hosting a database running MySQL technology.

In addition, the section, "Setting up Replication," has been included to step through the process of setting up MySQL server replication with two Sun Fire x86 servers. This environment will consist of a master server and a slave server as in the replication environment shown in Figure 1. The slave server will asynchronously replicate the Master server's database to the slave server's database.

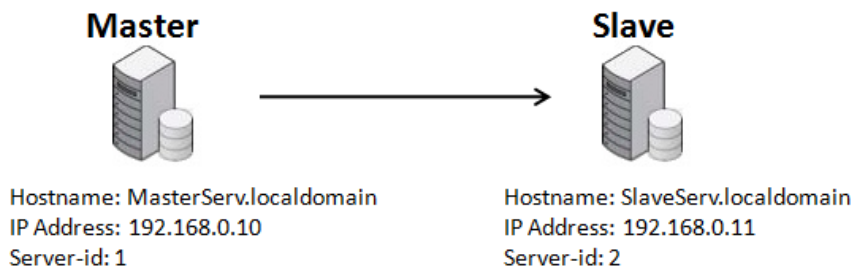


Figure 1. The MySQL replication environment consists of a master and a slave server.

Installation Overview

This installation of MySQL will be performed on a Sun Fire x86 server running Oracle Linux 5 (x86_64) Update 7. For installation of MySQL with hardware or software not listed in this guide, please refer to the MySQL 5.5 Reference Manual at: <http://dev.mysql.com/doc/refman/5.5/en/>.

The operating system was installed using the default Oracle Linux configurations and packages with the following details:

- The Base Package Group, with no optional packages or additional task support groups (Components of the Base Package Group are shown in Table 1)
- IPv4 configured
- Firewall enabled with SSH checked as a trusted service
- SELinux set to Permissive mode
- Registered operating system with Unbreakable Linux Network

TABLE 1. COMPONENTS OF THE BASE PACKAGE GROUP

GROUP	SUB-GROUPS
Desktop Environment	GNOME Desktop Environment
Applications	Editors Games and Entertainment Graphical Internet Graphics Office Productivity Sound and Video
Servers	Printing Support
Base System	Administration Tools Base Java Legacy Software Support X Windows System

Accessing the Installation System

There are several different ways to access the Oracle Linux operating system to perform the Oracle Database installation steps described in this document. For the purpose of this document, the installation system will be accessed using the Oracle Integrated Lights Out Manager (ILOM).

The ILOM provides access to remotely monitor and manage Oracle Sun Fire servers without consuming operating system resources. ILOM provides fully featured interfaces, including a browser-based Web interface, a command line interface, an SNMP interface, and an IPMI interface. These interfaces are based on industry standards and are intuitive to use. During this installation process, the ILOM's browser-based Web interface will be used to remotely access the system.

To access the ILOM's Web interface, open a Web browser and enter the ILOM's IP address. When prompted, enter the appropriate user name and associated password.

Select the Remote Control tab in the navigation tabs section. From the Redirection tab, click on the Launch Remote Console. This will open a Java-based interface to the console of the system.

Log into the operating system as root user.

The majority of the installation procedures described in this document will be performed using the terminal unless otherwise noted. To open the terminal window, after logging into the system, right click on the Oracle Linux desktop and select Open Terminal from the menu.

Required Pre-Requisites Check

This section lists the hardware and software pre-requisites to be checked before installation.

Hardware Pre-Requisites:

Before installing MySQL Enterprise Edition for Linux x86-64, the system must meet the following recommended minimum requirements for physical memory, memory swap space, and available disk space.

Physical Memory and Swap Space Requirement Check

Physical Minimum RAM: 512 MB

Swap Space: 256 MB

To check the available physical memory, use the following command:

```
# free
```

[Sample Output]

	total	used	free	shared	buffers	cached
Mem:	24726300	4810596	19915704	0	119712	3709048
-/+ buffers/cache:		981836	23744464			
Swap:	26705912	0	26705912			

In this test environment, the server has 24 GB of memory, more than minimum recommended system.

If the RAM size is less than required size, then please install more physical memory to get to at least the minimum level before continuing.

Swap space is recommended to be at least 256 MB, and the free command run earlier shows the test system currently has 26 GB, again more than the minimum recommended 256 MB. It is recommended to set the appropriate swap space size during the installation of Oracle Linux, but if the swap space size needs to be increased after the Operating system has already been installed, please refer to appendix A.1 on how to increase available swap space.

Physical Disk Space Requirement Check

For available disk space, please confirm the following space is available:

Minimum Physical Disk Space: 1 GB

Note: This does not include the space required for installing the binaries. Please take this sizing into consideration as well.

Again, these numbers are based on MySQL 5 Enterprise Edition for Oracle Linux 5 Update 7 x86-64 bit edition. For other configurations, please refer to the MySQL 5 Reference Guide.

To check disk space, enter the following command:

```
# df -h
```

Depending on how the file system has been created, the output may appear as a single LogicalVolume or individual /boot, /swap, /, etc... file structure.

[Sample Output from df -h]

Filesystem	Size	Used	Avail	Use%	Mounted on
/dev/mapper/VolGroup00-LogVol100	426G	16G	389G	4%	/
/dev/sdal	99M	23M	71M	25%	/boot
tmpfs	12G	0	12G	0%	/dev/shm

Please confirm that under the / directory, there is enough space available. To confirm in this example, look at the volume mounted on /

/dev/mapper/VolGroup00-LogVol100	426G	16G	389G	4%	/
----------------------------------	------	-----	------	----	---

LogicalVolume mounted on / directory has 389 GB of available space, more than enough free space to continue with the installation. If there is not enough physical space, please add additional disk space.

Software Pre-Requisites:

The software pre-requisites listed in this section are for MySQL 5.5 Enterprise Edition for Linux x86-64 on Oracle Linux 5 (x86_64) Update 7. If an installation is being performed on anything other than what is listed, please refer to the MySQL 5.5 Reference Manual, under Chapter 2: “Installation and Upgrading MySQL” for different Operating Systems.

Operating System Package Requirements Check

For this installation of Oracle MySQL 5.5 Enterprise Edition, the following packages (or later versions) need to be installed:

- glibc-2.3
- glibc-2.3 (32 bit)

To check if a package is installed, please use the following command:

```
# rpm -qa package_name
```

For example for the glibc package when already installed on the system, the following command will result in the subsequent system response:

```
# rpm -qa glibc
glibc-2.5-58
glibc-2.5-58
```

If the package name and version are returned, then the package is installed. If the package is not installed, please refer to Appendix A.2 on directions on how to install the needed packages. Most likely though this package has already been installed if the installation of the Linux operating system was installed with the default packages.

Opening Required Firewall Ports

In order to allow access for replication on the MySQL servers, or other connectivity to the MySQL server, the following ports need to be granted permission through the firewall:

- MySQL Port: 3306/tcp

To grant this port through the firewall, from the Oracle Linux desktop, select from the top menu System > Administration > Security Level and Firewall and click on “Other ports” to expand this section. In this section, click the “Add” button and enter the port number 3306 and select protocol tcp as seen in Figure 2.

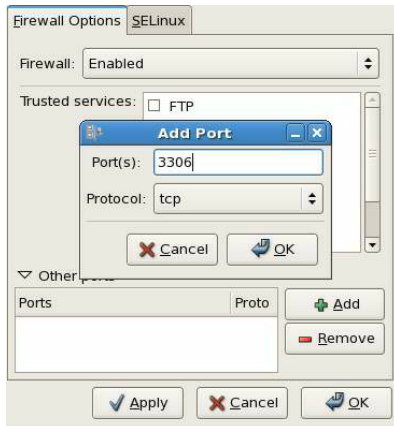


Figure 2. Menu for adding ports that are granted permission through the firewall.

When completed, click the “OK” button in the Security Level Configuration window.

Update Hosts Resolution Files for Name Configuration (for Environments Without DNS)

If setting up MySQL Replication in an environment without a DNS server, it is recommended to modify the `/etc/hosts` files on both servers.

To do this, open the `/etc/hosts` file on the Master server using the `vi` editor:

```
# vi /etc/hosts
```

And add the IP address and associated hostname of the slave server with the following format:

```
IP_ADDRESS <FQDN> <SHORT_NAME>
```

For example, in this test environment, the following line is added to the Master server's `/etc/hosts` file:

```
192.168.0.11 SlaveServ.localdomain SlaveServ
```

Next, open the slave server's `/etc/hosts` file using the `vi` editor:

```
# vi /etc/hosts
```

And add the IP address and associated hostname of the Master server with the following format:

```
IP_ADDRESS <FQDN> <SHORT_NAME>
```

For example, in this test environment, the following line is added to the slave server's `/etc/hosts` file

```
192.168.0.10 MasterServ.localdomain MasterServ
```

Installation Process

This section covers how to obtain the installation media as well as the installation procedure.

Obtaining Oracle's MySQL Installation Media

Open your Web browser and enter the following URL in the address bar:

https://edelivery.oracle.com/EPD/GetUserInfo/get_form?caller=WelcomePage

Enter the requested information and review and agree to the Trial License Agreement and Export Restrictions. Click “Continue” when completed.

From the “Select a Product Pack” dropdown list, select MySQL Database.

From the “Platform” dropdown list, select the appropriate platform. For this installation, the platform is Linux x86-64, since the installation is being performed on Oracle Linux 5 Update 7 (x86-64).

Click “OK” when completed.

Download the following item:

- MySQL Advanced Server 5.5.x RPM Linux 5 x86 (64bit)

Unzip the file using the following command:

```
# unzip V*.zip
```

After completing the unzip command for the file, the following RPM files should be extracted:

- MySQL-server-advanced-VERSION.x86_64.rpm
- MySQL-client-advanced-VERSION.x86_64.rpm
- MySQL-devel-advanced-VERSION.x86_64.rpm
- MySQL-shared-advanced-VERSION.x86_64.rpm
- MySQL-shared-compat-advanced-VERSION.x86_64.rpm
- MySQL-embedded-advanced-VERSION.x86_64.rpm
- MySQL-test-advanced-VERSION.x86_64.rpm

Installing MySQL Server and Client

During RPM installation, a user named mysql and a group named mysql are created on the system.

This is done using the useradd, groupadd, and usermod commands. Those commands require appropriate administrative privileges, which is required for locally managed users and groups (as listed in the /etc/passwd and /etc/group files) by the RPM installation process being run by root. So please confirm that you are running the installation as root user.

Install the MySQL server package:

```
# rpm -ivh MySQL-server-advanced-VERSION.x86_64.rpm
```

Then proceed to install the MySQL Client package:

```
# rpm -ivh MySQL-client-advanced- VERSION.x86_64.rpm
```

Post-Installation Configuration

There are some required operations that need to be performed after installing MySQL. Proceed through the following sections for the required steps.

Starting MySQL Server

After installing MySQL server, please start MySQL server if it is not already running. To check if the MySQL server is running, issue the following command:

```
# /etc/init.d/mysql status
```

If the output shows MySQL is not running, please issue the following command:

```
# /etc/init.d/mysql start
```

This will start the MySQL server. So if the `mysql status` command is run again, it should report the following:

```
MySQL running (PID#)
```

Securing MySQL Server for Production Environment

If this MySQL server is intended for production use, not just lab or testing usage, it is recommended to run the `mysql_secure_installation` script. This script sets the MySQL root password, removes anonymous users, disallows remote login by root user, and removes test databases that are created during the MySQL server installation. If this MySQL server is intended for non-production use, this next command can be skipped. To run the `mysql_secure_installation` script, execute the following command:

```
# /usr/bin/mysql_secure_installation
```

Please read each step carefully and enter the appropriate responses when prompted. Note that by default there is no root password.

Setting MySQL Root User Password

If the MySQL root user password has not yet been changed due to skipping the previous step, it is strongly recommended to set a password for the MySQL root user. To do this, please run the `/usr/bin/mysqladmin -u root password 'new-password'` command. For example:

```
# /usr/bin/mysqladmin -u root password mypassword
```

Starting the MySQL Client

To start the MySQL Client from the server where MySQL server was installed, run the following command:

```
# mysql -u root -p
```

Enter the correct password for MySQL root user.

To exit out of the MySQL client, type the exit command:

```
mysql> exit
```

Setting Up Replication

This section describes the process to configure MySQL Replication. This includes creating the proper `my.cnf` file that will setup binary logging and establish the unique server IDs. Also covered are the steps to create a dedicated replication user, connecting the master server and slave server, and validating the replication setup. This section will address two potential configurations that represent a new master-slave replication configuration and the integration of an existing MySQL database server to a replication configuration.

Setting Replication Configuration on the Master Server

On the master server, stop the MySQL server:

```
# /etc/init.d/mysql stop
```

Create `my.cnf` option file from `my-size.cnf` file:

- Locate `.cnf` files

```
# find / -name '*.cnf'
```

- o Most likely in `/usr/share/mysql/` directory

- Copy proper `my-size.cnf` to one of the following four locations as `my.cnf`:

```
/etc/my.cnf, /etc/mysql/my.cnf, /usr/local/mysql/etc/my.cnf, or ~/.my.cnf
```

o Example:

```
# cp /usr/share/mysql/my-large.cnf /etc/my.cnf
```

o For further details on access right types (global or user) based on location, and additional information on option files, please refer to appendix A.3

Modify created my.cnf file:

```
# vi /location/my.cnf
```

- Add the following lines in the [mysqld] section:

```
log-bin=mysql.bin  
sync_binlog=1
```

- Locate the following line in the [mysqld] section:

```
server-id=<value> (typically server-id=1)
```

o If this line does not exist, add the following line to the [mysqld] section:

```
server-id=1
```

o If the server-id=value is not 1, please change the server-id=value to 1.

Uncomment the following line:

```
innodb_flush_log_at_trx_commit=1
```

Save and exit my.cnf with wq option.

Start the MySQL server:

```
# /etc/init.d/mysql start
```

Setting Replication Configuration on the Slave Server

On the slave server, stop the MySQL server:

```
# /etc/init.d/mysql stop
```

Create my.cnf option file from my-size.cnf file:

- Locate .cnf files

```
# find / -name '*.cnf'
```

o Most likely in /usr/share/mysql/ directory

- Copy proper my-size.cnf to one of the following 4 locations as my.cnf:

/etc/my.cnf, /etc/mysql/my.cnf, /usr/local/mysql/etc/my.cnf, or ~/.my.cnf

o Example:

```
# cp /usr/share/mysql/my-large.cnf /etc/my.cnf
```

o For further details on access right types (global or user) based on location, and additional information on option files, please refer to appendix A.3.

Modify created my.cnf file:

```
# vi /<location>/my.cnf
```

- Locate the following line in the [mysqld] section:

```
server-id=<value> (typically server-id=1)
```

o If this line does not exist, add the following line to the [mysqld] section:

```
server-id=2
```

o If the server-id=value is 1, please change the server-id=value to 2 or another unique number that is not used by any other server in the replication configuration:

- Note that numbering is up to the customer to manage. Server ID can be any value between 1 and $(2^{32}) - 1$.

- Save and exit my.cnf with wq option.

- Restart the MySQL server:

```
# /etc/init.d/mysql start
```

NOTES:

- If setting up multiple slaves, each slave must have a unique ID value that differs from the master ID and other slave IDs.
- On slaves, it is not mandatory to enable binary logging (log-bin=mysql.bin) for replication. However, if binary logging is enabled on the slaves, the binary logs can be used for data backups and crash recovery on the slave. Also, the slave can be used as part of a more complex replication topology (i.e.: where the slave acts as a master to other slaves).

Creating a User for Replication

Each slave must connect to the master using a MySQL user name and password, so there must be a user account on the master that the slave can use to connect. Any account can be used for this operation, providing it has been granted the REPLICATION SLAVE privilege. However, it is recommended to create a dedicated replication user as to minimize the possibility of compromise to other MySQL accounts.

Log into MySQL server (master):

```
# mysql -u root -p
```

Use the following command to create a replication user named *repl* that will be connecting from slave server 192.168.0.11 with the password replpass.

```
mysql> CREATE USER 'repl'@'192.168.0.11' IDENTIFIED BY 'replpass';
```

Grant the replication user REPLICATION SLAVE privilege.

```
mysql> GRANT REPLICATION SLAVE ON *.* TO 'repl'@'192.168.0.11';
```

Obtaining the Replication Master Binary Log Coordinates

To configure replication on the slave server, the master's current coordinates within its binary log must be determined. This information is required so that when the slave starts the replication process, it is able to start processing events from the binary log at the correct point.

If there is existing data on the master server that needs to be synchronized on the slaves before starting the replication process, processing statements on the master need to be stopped, and then obtain its current binary log coordinates and dump its data before permitting the master to continue executing statements. If the execution of statements is not stopped, the data dump and the master status information that are used will not match and will end up with inconsistent or corrupted databases on the slaves.

To obtain the master binary log coordinates, start a client session on the master:

```
# mysql -u root -p
```

And flush all tables and block write statements by executing the FLUSH TABLES WITH READ LOCK statement:

```
mysql> FLUSH TABLES WITH READ LOCK;
```

NOTE: For InnoDB tables, the FLUSH TABLES WITH READ LOCK blocks COMMIT operations as well.

Leave this current mysql client session running so that the read lock remains in effect. If this client session, where the FLUSH TABLES command was run, is closed the lock will be released.

Next, open another client session on the master and use the SHOW MASTER STATUS statement to determine the currently binary log file name and position:

```
mysql> SHOW MASTER STATUS;
```

[Sample Output]

```
+-----+-----+-----+-----+
| File           | Position | Binlog_Do_DB | Binlog_Ignore_DB |
| mysql-bin.000002 |      428 |              |                  |
+-----+-----+-----+-----+
1 row in set (0.00 sec)
```

The File column shows the name of the log file and the Position shows the position within the file. In this example, the binary log file is mysql-bin.000002 and the position is 428. Make note of these values. They will be needed later when setting up the slave as they are the replication coordinates at which the slave should begin processing new updates from the master.

If there is existing data that needs to be synchronized with the slave before starting replication, leave the initial client session running so that the lock remains in place and proceed to the, “Setting up Replication with an Existing MySQL Database Server,” section. Otherwise, if this is a brand new master-slave replication group, the client lock session can be closed to release the lock and progress to the, “Setting up Replication with New Master and Slaves,” section.

Setting Up Replication with New Master and Slaves

The easiest and most straightforward method for setting up replication is to use new master and slave servers.

To setup replication between a new master and slave server, the master configuration needs to be set on the slave. To do this, log into the MySQL slave server:

```
# mysql -u root -p
```

When logged in, execute the following commands:

```
mysql> CHANGE MASTER TO
-> MASTER_HOST='master_host_name or master_ip_address',
-> MASTER_USER='replication_user_name',
-> MASTER_PASSWORD='replication_password',
-> MASTER_LOG_FILE='recorded_log_file_name',
-> MASTER_LOG_POS=recorded_log_position;
```

[Example]

```
mysql> CHANGE MASTER TO
-> MASTER_HOST='192.168.0.10',
-> MASTER_USER='repl',
-> MASTER_PASSWORD='replpass',
-> MASTER_LOG_FILE='mysql-bin.000002',
-> MASTER_LOG_POS=428;
```

After successfully running the CHANGE MASTER TO command, start the slave threads:

```
mysql> START SLAVE;
```

Confirm that the slave thread is communicating with the MySQL master server. Check the status of the slave server:

```
mysql> SHOW SLAVE STATUS \G;
```

There should not be any active errors and the Slave_IO_State should show the following:

```
Slave_IO_State: Waiting for master to send event
```

This completes the setup of setting replication with new master and slave servers.

Setting Up Replication with an Existing MySQL Database Server

This section describes setting up replication for an existing MySQL database server such that the existing MySQL database server will become the master server and the database information will be replicated to a slave server. This is done by first creating a snapshot of the existing database, importing this snapshot to the slave server, noting the file name and position from the master status command ran earlier, and setting the master configuration on the slave server, which initializes replication.

Creating a Database Snapshot Using mysqldump Command

Create a snapshot of the data in an existing master database using the mysqldump tool. Once the data dump has been completed, this data dump can be imported into the slave before starting the replication process.

Obtain a snapshot of the data using mysqldump command from a shell session, other than the client lock; create a dump of all the databases or of selected individual databases using the following commands:

Example for mysqldump for all databases:

```
# mysqldump -u root -p --all-databases --master-data > dbdump.db
```

Example of mysqldump for a selected database (for example, my_test_db):

```
# mysqldump -u root -p --databases my_test_db --master-data > dbdump.db
```

At this point, release the lock on the client lock session by running the following command from the client lock session:

```
mysql> UNLOCK TABLES;
```

At this point, copy the dbdump.db file over to the slave. When completed moving the dump file over to the slave, go to the, “Setting up Replication with Existing Data,” section.

Setting Up Replication with Existing Data

On the slave, stop the MySQL server if it is currently running. To stop the MySQL server, enter the following command:

```
# /etc/init.d/mysql stop
```

Open another terminal session from the slave server and start the MySQL server with the `--skip-slave-start` option so that replication does not start. Enter the following command to start the MySQL server with this option:

```
# mysqld_safe --skip-slave-start
```

Go back to the original slave terminal session and import the snapshot file that was generated earlier with the `mysqldump` command.

```
# mysql -u root -p < dbdump.db
```

Next, set the master configuration on the slave. To do this, log into the MySQL slave server:

```
# mysql -u root -p
```

When logged in, execute the following commands:

```
mysql> CHANGE MASTER TO
-> MASTER_HOST='master_host_name or master_ip_address',
-> MASTER_USER='replication_user_name',
-> MASTER_PASSWORD='replication_password',
-> MASTER_LOG_FILE='recorded_log_file_name',
-> MASTER_LOG_POS=recorded_log_position;
```

[Example]

```
mysql> CHANGE MASTER TO
-> MASTER_HOST='192.168.0.10',
-> MASTER_USER='repl',
-> MASTER_PASSWORD='replpass',
-> MASTER_LOG_FILE='mysql-bin.000002',
-> MASTER_LOG_POS=428;
```

After successfully running the `CHANGE MASTER TO` command, start the slave threads:

```
mysql> START SLAVE;
```

Close the terminal session where the `--skip-slave-start` option was run from.

Confirm that the slave thread is communicating with the MySQL master server. Check the status of the slave server:

```
mysql> SHOW SLAVE STATUS \G;
```

There should not be any active errors and the `Slave_IO_State` should show the following:

```
Slave_IO_State: Waiting for master to send event
```

This completes the setup of setting replication with an existing MySQL database server.

Validating MySQL Replication Configuration

To confirm that replication has been configured correctly, a temporary test database can be created on the master server and it should automatically be replicated on the slave server. To perform this operation, log in to the mysql client on the master server.

```
# mysql -u root -p
```

Run the following mysql command to create a database:

```
mysql> CREATE DATABASE TempTest;
```

Now, check that the TempTest database exists on the slave server. Log into the MySQL client from the slave server:

```
# mysql -u root -p
```

Run the following command to show the available databases:

```
mysql> SHOW DATABASES;
```

The database TempTest should appear in the list of databases.

To remove the TempTest database, go back to the MySQL client session on the master server and run the following command:

```
mysql> DROP DATABASE TempTest;
```

This will remove the database from the master server as well as the replicated database on the slave.

Conclusion

The installation of Oracle's MySQL Enterprise Edition and MySQL replication configuration can be performed fairly quickly and easily, especially when utilizing the Oracle stack. Support of Oracle applications, operating systems, and hardware is available from one location, as well as patches, software updates, and driver updates. This simplifies not only the installation and setup of the Oracle stack, but also reduces the complication of maintaining the environment. For further information and available product feature details, please visit the Oracle Web sites listed in Table 2.

TABLE 2: LINKS TO ADDITIONAL INFORMATION

ORACLE PRODUCT OFFERING	WEB SITE URL
Oracle's Sun Fire x86 servers:	http://www.oracle.com/goto/x86
Oracle Linux:	http://www.oracle.com/us/technologies/linux/
Oracle's MySQL:	http://www.mysql.com/
Oracle's MySQL Enterprise Edition:	http://www.mysql.com/products/enterprise/
Oracle 's MySQL 5.5 Reference Manual:	http://dev.mysql.com/doc/refman/5.5/en/

Appendix

A.1 Increasing Swap Space Process

Check the current available Swap Space by running the following command:

```
# free
```

[Sample Output]

	total	used	free	shared	buffers	cached
Mem:	24726300	4810596	19915704	0	119712	3709048
-/+ buffers/cache:		981836	23744464			
Swap:	26705912	0	26705912			

There is about 26 GB (26705912 kilobytes) of Swap Space as seen in the following line:

```
Swap:      26705912          0          26705912
```

In this example, the Swap Space will be increased by 11 GB. To do so, a supplementary Swap File needs to be created. This file can be created by using the data dump command (dd). This command can only be run under root, so if not already logged in as root, please do so or enter super user role with the command:

```
# su - root
```

And enter the root password when prompted.

When ready, carefully type in the dd command:

```
# dd if=/dev/zero of=<outfile_location_and_filename> bs=<bit_size>
count=<#_of_times_ran>
```

In the following data dump command:

```
# dd if=/dev/zero of=/moreswap bs=1G count=10
```

[Sample Output]

```
10+0 records in
10+0 records out
10737418240 bytes (11 GB) copied, 104.207 seconds, 103 MB/s
```

An outfile titled moreswap in the / directory will be created with zeros written to the output file. This moreswap file will be 11 GB in size since bit size was specified as 1G and run times were 10 (0 – 10). To prepare this output file as a swap file, the command mkswap is run against the moreswap file to assign the file as a swap-consumable for the Linux kernel.

```
# mkswap /moreswap
```

[Sample Output]

```
Setting up swapspace version 1, size = 10737414 kB
```

After the moreswap file is assigned as swap space, the supplemental swap file needs to be enabled. To do this, run the swapon command on the new swap file

```
# swapon /moreswap
```

Confirm that the swap space has been increased by running the free command again.

```
# free
```

[Sample Output]

```

      total        used        free      shared    buffers  cached
Mem:   24726300   14518900   10207400          0     82316   13165320
-/+ buffers/cache: 1271264   23455036
Swap:   37191664          0   37191664
```

In order for this swap space to remain available after a system reboot, the fstab file needs to be modified to mount this swap space at boot. Be careful when modifying the fstab file, as mistaken changes can render the OS unbootable. It is also recommended to create a backup of the fstab file before making any modifications.

To modify the fstab file, open the file with the vi editor

```
# vi /etc/fstab
```

Locate the line that references your previously existing swap file

[Sample Line Entry]

```
/dev/VolGroup00/LogVol01 swap                    swap    defaults        0 0
```

Under this line, enter a similar entry pointing to the newly created swap file

[Sample New Line Entry]

```
/moreswap                swap                    swap    defaults        0 0
```

So the fstab file should appear something like the following:

[Sample Line Entry]

```
/dev/VolGroup00/LogVol01 swap                    swap    defaults        0 0
/moreswap                swap                    swap    defaults        0 0
```

Exit the fstab file saving the needed changes with the “wq” command.

A.2 Installing Packages

To install Oracle Linux packages, it is recommended to have added the server to the Oracle Support Network. This will allow the download and installation of the latest Linux packages and patches. To install the packages, the following command is used:

```
# yum install package_name
```

Example: This example demonstrates the installation of the `elfutils-libelf-devel.x86_64` package. If this package is already installed and is at the latest level, yum installer will state this and exit out of the installation. If not, yum will continue with the installation of the package as well as other package dependencies.

```
# yum install elfutils-libelf-devel.x86_64
Loaded plugins: rhnplugin, security
Setting up Install Process
Resolving Dependencies
--> Running transaction check
---> Package elfutils-libelf-devel.x86_64 0:0.137-3.el5 set to be updated
--> Processing Dependency: elfutils-libelf-devel-static-x86_64 = 0.137-3.el5 for
package: elfutils-libelf-devel
--> Running transaction check
---> Package elfutils-libelf-devel-static.x86_64 0:0.137-3.el5 set to be updated
--> Finished Dependency Resolution
```

Dependencies Resolved

```
=====
Package                Arch      Version      Repository    Size
=====
Installing:
elfutils-libelf-devel   x86_64    0.137-3.el5  localinfo    24 k
Installing for dependencies:
elfutils-libelf-devel-static x86_64    0.137-3.el5  localinfo    64 k
```

Transaction Summary

```
=====
Install      2 Package(s)
Upgrade     0 Package(s)
```

Total download size: 88 k

Is this ok [y/N]: y

Downloading Packages:

```
-----
Total                    57 MB/s | 88 kB    00:00
Running rpm_check_debug
Running Transaction Test
Finished Transaction Test
Transaction Test Succeeded
Running Transaction
  Installing      : elfutils-libelf-devel                1/2
```

```
Installing      : elfutils-libelf-devel-static
```

```
2/2
```

```
Installed:
```

```
elfutils-libelf-devel.x86_64 0:0.137-3.el5
```

```
Dependency Installed:
```

```
elfutils-libelf-devel-static.x86_64 0:0.137-3.el5
```

```
Complete!
```

A.3 Access Rights and Other Information for my.cnf Option Files

MySQL provides a number of preconfigured option files that can be used as a basis for tuning the MySQL server. Sample files such as `my-small.cnf`, `my-medium.cnf`, `my-large.cnf`, and `my-huge.cnf`, which are sample option files for small, medium, large, and very large systems.

On Linux, MySQL programs read startup options from the files listed in Table 3. Files are processed in the specified order (top items first).

TABLE 3. SOURCES FOR STARTUP OPTIONS IN LINUX IMPLEMENTATIONS

FILE NAME	PURPOSE
<code>/etc/my.cnf</code>	Global options
<code>/etc/mysql/my.cnf</code>	Global options
<code>\$MYSQL_HOME/my.cnf</code>	Server-specific options
<code>~/my.cnf</code>	User-specific options

Note: The symbol "~" represents the current user's home directory (the value of `$HOME`).

`MYSQL_HOME` is an environment variable containing the path to the directory in which the server-specific `my.cnf` file resides. If `MYSQL_HOME` is not set and you start the server using the `mysqld_safe` program, `mysqld_safe` attempts to set `MYSQL_HOME` as follows:

- Let `BASEDIR` and `DATADIR` represent the path names of the MySQL base directory and data directory, respectively.
- If there is a `my.cnf` file in `DATADIR` but not in `BASEDIR`, `mysqld_safe` sets `MYSQL_HOME` to `DATADIR`.
- Otherwise, if `MYSQL_HOME` is not set and there is no `my.cnf` file in `DATADIR`, `mysqld_safe` sets `MYSQL_HOME` to `BASEDIR`.

In MySQL 5.5, use of `DATADIR` as the location for `my.cnf` is deprecated.

Typically, DATADIR is /usr/local/mysql/data for a binary installation or /usr/local/var for a source installation. Note that this is the data directory location that was specified at configuration time, not the one specified with the --datadir option when mysqld starts. Use of --datadir at runtime has no effect on where the server looks for option files, because it looks for them before processing any options.

MySQL looks for option files in the order just described and reads any that exist. If an option file that you want to use does not exist, create it with a plain text editor.

If multiple instances of a given option are found, the last instance takes precedence. There is one exception: For mysqld, the first instance of the --user option is used as a security precaution, to prevent a user specified in an option file from being overridden on the command line.

Any long option that may be given on the command line when running a MySQL program can be given in an option file as well. To get the list of available options for a program, run it with the --help option.

Further details can be found at the following link:

<http://dev.mysql.com/doc/refman/5.5/en/option-files.html>



Installation Guide for Single Click
Configurations of Oracle's MySQL 5 with Oracle
Linux 5 for Sun Fire x86 Servers
October 2011, Version 1.0

Oracle Corporation
World Headquarters
500 Oracle Parkway
Redwood Shores, CA 94065
U.S.A.

Worldwide Inquiries:
Phone: +1.650.506.7000
Fax: +1.650.506.7200

oracle.com



Oracle is committed to developing practices and products that help protect the environment

Copyright © 2011, Oracle and/or its affiliates. All rights reserved. This document is provided for information purposes only and the contents hereof are subject to change without notice. This document is not warranted to be error-free, nor subject to any other warranties or conditions, whether expressed orally or implied in law, including implied warranties and conditions of merchantability or fitness for a particular purpose. We specifically disclaim any liability with respect to this document and no contractual obligations are formed either directly or indirectly by this document. This document may not be reproduced or transmitted in any form or by any means, electronic or mechanical, for any purpose, without our prior written permission.

Oracle and Java are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

Intel and Intel Xeon are trademarks or registered trademarks of Intel Corporation. All SPARC trademarks are used under license and are trademarks or registered trademarks of SPARC International, Inc. AMD, Opteron, the AMD logo, and the AMD Opteron logo are trademarks or registered trademarks of Advanced Micro Devices. UNIX is a registered trademark licensed through X/Open Company, Ltd. 0611

Hardware and Software, Engineered to Work Together