MySQL 数据库管理员

实践指南

D61762CN30
版本 3.0
2013 年 8 月
D82513

ORACLE

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

免责声明

本文档包含专有权信息，并受版权法和其它知识产权法的保护。您可以复制和打印本文档以供在 Oracle 培训课程中单独使用。不得以任何方式修改或变更本文档。除了在依照版权法中制定的“合理使用”范围内使用本文档外，在未经 Oracle 明确授权的情况下，您不得以全部或部分的形式使用、共享、下载、上载、复制、打印、显示、展示、再版、发布、许可、张贴、传播或散布本文档。

本文档中包含的信息如有更改，恕不另行通知。如果您在本文档中发现任何问题，请书面通知：Oracle University, 500 Oracle Parkway, Redwood Shores, California 94065 USA。不保证本文档中没有错误。

有限权利声明

如果将本文档交付给美国政府或代表美国政府使用本文档的任何人，必须符合以下规定：

U.S. GOVERNMENT RIGHTS
The U.S. Government’s rights to use, modify, reproduce, release, perform, display, or disclose these training materials are restricted by the terms of the applicable Oracle license agreement and/or the applicable U.S. Government contract.

商标声明

Oracle 和 Java 是 Oracle 和/或其附属公司的注册商标。其它名称可能是其各自拥有者的商标。

作者

Jeremy Smyth

技术撰稿人和审稿人

Mark Lewin、Horst Hunger、Andrew Morgan、Todd Farmer、Jonathon Coombes、Sneha Modi、Bob Falasco、Saikumar Vishvanatha、Kim Seong Loh、Mattias Jonsson、Chuck Bell、Ingo Strüwing、Matt Lord、Bertrand Matthelié

本书使用的发布工具：Oracle Tutor

目录

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

第 1 课的练习：MySQL 简介

第 1 章

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

第 1 课的练习

课程概览

本课没有练习。

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

第 2 课的练习：体系结构

第 2 章

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

第 2 课的练习：概览

练习概览

通过这些练习，可测试您对 MySQL 体系结构的掌握程度。上机练习是针对 Oracle Linux 操作系统环境编写的，Oracle 课堂中将提供此环境。对于非 Oracle 课堂环境，可能需要对文件位置进行一些调整。

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

练习 2-1：测验 — 体系结构概览

概览

在本练习中，您将回答有关 MySQL 体系结构的问题。

持续时间

完成本练习大约需要 10 分钟。

测验问题

请从所提供的每道多选题或正/误判断题中选择最佳答案。

1. 所有 MySQL 客户机程序和非客户机程序均可与 MySQL 服务器进行通信。
    a. 正确
    b. 错误
2. 在 Windows 下运行 MySQL 服务器时，访问该服务器的客户机程序也必须在 Windows 下运行。
    a. 正确
    b. 错误
3. 命令行程序通常用于与称为 `mysqld` 的 MySQL 服务器进行通信。
    a. 正确
    b. 错误
4. 请考虑下面列出的用于连接到 MySQL 服务器的方式。哪项不依赖于操作系统？（请选择所有正确的选项。）
    a. TCP/IP
    b. ODBC
    c. 共享内存
    d. 命名管道
    e. UNIX 套接字文件
5. MySQL 使用磁盘空间存储以下对象：
    a. 服务器和客户机程序及其库
    b. 日志文件和状态文件
    c. 数据库
    d. 表格式 (`.frm`) 文件、数据文件和索引文件
    e. 超过了用于从内存中表转换为磁盘上表的大小阈值的内部临时表
    f. 以上都是

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

6. MySQL 服务器为以下对象分配内存：
    a. 连接处理程序（每个连接都使用内存）
    b. 缓冲区和高速缓存
    c. 授权表的副本
    d. 主机高速缓存和表高速缓存
    e. 查询高速缓存
    f. 以上都是
7. MySQL 服务器使用 _____ 作为内存，着手临时保存数据以实现避免成本高昂的磁盘访问 I/O 的目的。
    a. MEMORY 表
    b. 内部临时表
    c. 共享内存
    d. 缓冲区（或高速缓存）

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

练习解答 2-1：测验 — 体系结构概览

测验解答

1. **b.** 错误。有些 MySQL 程序不与服务器进行通信，而是直接处理数据或日志文件。示例包括 `innochecksum` 和 `mysqlbinlog`。
2. **b.** 错误。MySQL 可以用在异构环境中。例如，在 UNIX 主机上运行的服务器可由在 Windows 计算机上运行的客户机进行访问。
3. **b.** 错误。最常用的命令行程序称为 `mysql`，而不是 `mysqld`。后者是 MySQL 服务器。
4. **a、b。**TCP/IP 不依赖于操作系统。ODBC 也不依赖于操作系统，但仅在支持 MySQL Connector/ODBC 的操作系统上可用于 MySQL。
5. **f.** 所列出的所有这些对象都使用磁盘空间进行存储。
6. **f.** MySQL 为所列出的所有这些对象分配内存。
7. **d.** MySQL 服务器使用缓冲区（或高速缓存）作为内存，着手临时保存数据以实现避免成本高昂的磁盘访问 I/O 的目的。

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

第 3 课的练习：系统管理

第 3 章

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

第 3 课的练习：概览

练习概览

通过这些练习，可测试您对 MySQL 系统管理的掌握程度。上机练习是针对 Oracle Linux 操作系统环境编写的，Oracle 课堂中将提供此环境。对于非 Oracle 课堂环境，可能需要对文件位置进行一些调整。

练习使用标准 Linux 用户 (`oracle`) 和 `root` 用户。练习步骤使用 `$` 提示符来指示标准用户，使用 **#** 提示符来指示 `root` 用户。

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

练习 3-1：安装 MySQL 服务器

概览

在本练习中，您将安装各个 Linux RPM 文件。

假设

MySQL 服务器 Linux RPM 文件已经作为 Oracle 课堂环境的一部分而下载，位于 `/stage` 目录中。

持续时间

完成本练习大约需要 10 分钟。

任务

1. 安装 **`/stage/mysql`** 目录中的各个 RPM 文件（以 `root` 身份）。
    a. 对每个文件执行以下命令：
    - `rpm -hi --replacefiles MySQL-server*.rpm`
    - `rpm -hi MySQL-client*.rpm`
    - `rpm -hi MySQL-devel*.rpm`
    - `rpm -hi MySQL-shared*.rpm`
    - `rpm -hi MySQL-test*.rpm`
    – 如果您的系统包含先前版本的预安装 MySQL 库，可以将 `--replacefiles` 选项添加到 `rpm` install 命令。在 Oracle 课堂中，`server` 软件包需要此选项。
2. 启动 MySQL 服务器。
3. 保护 MySQL 服务器。

    注：将需要位置 `/root/.mysql_secret` 中保存的初始 `root` 口令。

    对于您自己的位置处的安装，请参阅“MySQL Downloads（MySQL 下载）”网页上的“Community Edition（社区版）”下载文件：http://dev.mysql.com/downloads/。

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

练习解答 3-1：安装 MySQL 服务器

任务

1. 安装 /stage/mysql 目录中的各个 RPM 文件（以 root 身份）。
   a. 对终端窗口中的每个文件执行以下命令，得到的结果如下所示：

```
$ su -
Password: oracle
# cd /stage/mysql
# rpm -hi --replacefiles MySQL-server*.rpm
...
A RANDOM PASSWORD HAS BEEN SET FOR THE MySQL root USER !
You will find that password in '/root/.mysql_secret'.

You must change that password on your first connect,
no other statement but 'SET PASSWORD' will be accepted.
See the manual for the semantics of the 'password expired' flag.

Also, the account for the anonymous user has been removed.

In addition, you can run:

  /usr/bin/mysql_secure_installation

which will also give you the option of removing the test database.
This is strongly recommended for production servers.
...
New default config file was created as /usr/my.cnf and
will be used by default by the server when you start it.
You may edit this file to change server settings

WARNING: Default config file /etc/my.cnf exists on the system
This file will be read by default by the MySQL server
...
# rpm -hi MySQL-client*.rpm
# rpm -hi MySQL-devel*.rpm
# rpm -hi MySQL-shared*.rpm
# rpm -hi MySQL-test*.rpm
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

2. 启动 MySQL 服务器。

   a. 在终端窗口中输入以下内容，得到的结果如下所示：

```
# service mysql start
Starting MySQL..                                           [ OK ]
```

3. 保护 MySQL 服务器。

   a. 在终端窗口中输入以下内容，输入适合您安装的回答，得到的结果如下所示：

```
# cat /root/.mysql_secret
# The random password set for the root user ...: TeDRFkYP
# mysql -uroot -p
Enter password: TeDRFkYP
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 1
Server version: 5.6.10-enterprise-commercial-advanced

Copyright (c) 2000, 2013, Oracle and/or its affiliates. All rights
reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input
statement.

mysql> SET PASSWORD=PASSWORD('oracle');
Query OK, 0 rows affected (0.00 sec)

mysql> EXIT
Bye
# /usr/bin/mysql_secure_installation
NOTE: RUNNING ALL PARTS OF THIS SCRIPT IS RECOMMENDED FOR ALL MySQL
SERVERS IN PRODUCTION USE! PLEASE READ EACH STEP CAREFULLY!
...
Enter current password for root (enter for none): oracle
OK, successfully used password, moving on...
...
You already have a root password set, so you can safely answer 'n'.

Set root password? [Y/n] n
 ... skipping.
...
Remove anonymous users? [Y/n] Y
 ... Success!
...
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

```
Disallow root login remotely? [Y/n] Y
 ... Success!
...
Remove test database and access to it? [Y/n] Y
 ... Success!
...
Reload privilege tables now? [Y/n] Y
 ... Success!
...
All done!  If you've completed all of the above steps, your MySQL
installation should now be secure.
Thanks for using MySQL!
```

注：虽然此解答显示 `mysql_secure_installation` 脚本以 Linux `root` 用户身份运行，但您可以普通用户身份运行该脚本。

对于您自己的位置处的 MySQL 安装，请参阅“MySQL Downloads（MySQL 下载）”网页上的“Community Edition（社区版）”下载文件：http://dev.mysql.com/downloads/。

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

练习 3-2：MySQL 数据目录

概览

在本练习中，您将查看与 MySQL 服务器关联的数据目录。本练习要求您使用 mysql 客户机装入 world_innodb 数据库。要查看 world_innodb 数据库中预生成的表，请创建该数据库（空的），然后上载包含表数据的文件。

持续时间

本练习应该需要大约五分钟来完成。

任务

1. 创建和填充 world_innodb 数据库。
   a. 进入 mysql 客户机。在终端窗口中输入以下内容：

```
$ mysql -uroot -poracle
```

   使用在 MySQL 服务器安装过程中建立的用户名和口令。
   b. 使用下面的 CREATE DATABASE 语句创建空的 world_innodb 数据库。在 mysql 客户机中，输入以下内容：

```
mysql> CREATE DATABASE world_innodb;
```

   c. 使用 USE 语句选择 world_innodb 数据库：

```
mysql> USE world_innodb
```

   d. 使用 **SOURCE** 语句构建并填充 world_innodb 数据库来运行 world_innodb.sql 脚本：

```
mysql> SET autocommit=0;
Query OK, 0 rows affected (0.00 sec)

mysql> SOURCE /labs/world_innodb.sql
```

   – “事务与锁定”一课中介绍了 autocommit 选项。在此处重置该选项可提高脚本执行速度。
   – 在命令完成之前会滚动出现多条“Query OK…”消息。
   e. 设置 autocommit 选项：

```
mysql> SET autocommit=1;
Query OK, 0 rows affected (0.00 sec)
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

2. 列出当前数据库。

   a. 在 mysql 客户机中，输入以下内容：

```
mysql> SHOW DATABASES;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| mysql              |
| performance_schema |
| world_innodb       |
+--------------------+
4 rows in set (0.00 sec)
```

   – 此时，将列出四个数据库。

3. 找到本地 MySQL 服务器数据目录。

   a. 在 mysql 客户机中，输入以下内容：

```
mysql> SHOW VARIABLES LIKE 'datadir'\G
```

4. 查看数据目录的内容。

   a. 使用前面 SHOW VARIABLES 语句的结果，在另一个终端窗口中输入以下内容：

```
$ cd /var/lib/mysql
$ ls
```

   – MySQL 服务器上的每个数据库都有其自己的目录。

5. 查看 mysql 数据库目录的内容。

   a. 以 root 用户身份，将目录转至 mysql 目录并查看内容。在终端窗口中输入以下内容：

```
$ su -
Password: oracle
# cd /var/lib/mysql/mysql
# ls
```

   – 请注意目录中包含的许多 .frm 文件，表示表格式。此外，还请注意 MyISAM 表的 .MYD 和 .MYI 文件以及 InnoDB 表数据的一些 .ibd 文件和 slow_log 的 .CSV 和 .CSM 文件及其元文件。

6. 查看 world_innodb 数据库目录的内容。

   a. 将目录转至 world_innodb 目录并查看内容。在前面步骤中使用的终端窗口中输入以下内容：

```
# cd ../world_innodb
# ls
```

   – 请注意目录中包含的表 .frm 文件和 InnoDB .idb 文件。

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

练习解答 3-2：MySQL 数据目录

任务

1. 创建和填充 world_innodb 数据库。

   a. 进入 mysql 客户机。在终端窗口中键入以下内容，得到的结果如下所示：

```
$ mysql -uroot -poracle
...
Server version: 5.6.8-enterprise-commercial-advanced-log MySQL
Enterprise Server - Advanced Edition (Commercial)
...
mysql>
```

   b. 创建空的 world_innodb 数据库：

```
mysql> CREATE DATABASE world_innodb;
Query OK, 1 row affected (0.02 sec)
```

   c. 选择 world_innodb 数据库：

```
mysql> USE world_innodb
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
```

   d. 使用 **SOURCE** 语句构建并填充 world_innodb 数据库来运行 world_innodb.sql 脚本：

```
mysql> SET autocommit=0;
Query OK, 0 rows affected (0.00 sec)

mysql> SOURCE /labs/world_innodb.sql
```

   – “事务与锁定”一课中介绍了 autocommit 选项。在此处重置该选项可提高脚本执行速度。
   – 在命令完成之前会滚动出现多条“Query OK…”消息。

   e. 设置 autocommit 选项：

```
mysql> SET autocommit=1;
Query OK, 0 rows affected (0.00 sec)
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

2. 列出当前数据库。

   a. 在 mysql 客户机中，输入以下内容，得到的结果如下所示：

```
mysql> SHOW DATABASES;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| mysql              |
| performance_schema |
| world_innodb       |
+--------------------+
4 rows in set (0.00 sec)
```

   – 此时，服务器包含四个数据库。

3. 找到本地 MySQL 服务器数据目录。

   a. 在 mysql 客户机中，输入以下内容，得到的结果如下所示：

```
mysql> SHOW VARIABLES LIKE 'datadir'\G
*************************** 1. row ***************************
Variable_name: datadir
        Value: /var/lib/mysql/
1 row in set (0.01 sec)
```

4. 查看数据目录的内容。

   a. 在另一个终端窗口中输入以下内容，得到的结果如下所示：

```
$ cd /var/lib/mysql
$ ls
auto.cnf      ib_logfile1     mysql               RPM_UPGRADE_HISTORY
ibdata1       hostname.err    mysql.sock          RPM_UPGRADE_MARKER-
LAST
ib_logfile0   hostname.pid    performance_schema  world_innodb
```

5. 查看 mysql 数据库目录的内容：

   a. 在终端窗口中输入以下内容，得到的结果如下所示：

```
$ su -
Password: oracle
# cd /var/lib/mysql/mysql
# ls
columns_priv.frm   innodb_index_stats.frm    slave_worker_info.ibd
columns_priv.MYD   innodb_index_stats.ibd    slow_log.CSM
columns_priv.MYI   innodb_table_stats.frm    slow_log.CSV
...
help_topic.MYD     slave_relay_log_info.ibd
help_topic.MYI     slave_worker_info.frm
```

   – 请注意目录中包含的许多 .frm 文件，表示表格式。此外，还请注意 MyISAM 表的 .MYD 和 .MYI 文件以及 InnoDB 表数据的一些 .ibd 文件和 slow_log 的 .CSV 和 .CSM 文件及其元文件。

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

6. 查看 world_innodb 数据库目录的内容。

   a. 转至 world_innodb 目录并查看内容。在终端窗口中输入以下内容，得到的结果如下所示：

```
# cd ../world_innodb
# ls
City.frm  Country.frm  CountryLanguage.frm  db.opt
City.ibd  Country.ibd  CountryLanguage.ibd
```

   – 请注意目录中包含的表 .frm 文件和 InnoDB .idb 文件。

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

练习 3-3：启动和停止 MySQL 服务器

概览

在本练习中，将探讨如何在 Linux 环境中启动和停止 MySQL 服务器。

假设

在本练习之前已经完成了 MySQL 服务器安装。

持续时间

本练习应该需要大约五分钟来完成。

任务

1. 在操作服务器之前，以 `admin`（或 `root`）特权登录。
2. 检查 MySQL 服务器的状态。（如果服务器未在运行，请参阅解答注释。）
3. 停止 MySQL 服务器。
4. 再次检查 MySQL 服务器的状态。
5. 再次启动 MySQL 服务器。
6. 再次检查 MySQL 服务器的状态。

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

练习解答 3-3：启动和停止 MySQL 服务器

任务

1. 在操作服务器之前，必须具有 admin（或 root）特权。
   a. 在终端窗口中输入以下内容：

```
$ su -
Password: oracle
```

2. 检查 MySQL 服务器的状态。
   a. 在终端窗口中输入以下内容，得到的结果如下所示：

```
# service mysql status
MySQL running (1117)                                       [ OK ]
```

   – **注：**此时服务器应该正在运行 — 它已在前面的练习中启动。如果由于某种原因，服务器未在运行，请先运行 start 命令，然后再在下一步运行 stop 命令。进程编号可能与前面输出中显示的不同。

3. 停止 MySQL 服务器。
   a. 在终端窗口中输入以下内容，得到的结果如下所示：

```
# service mysql stop
Shutting down MySQL......                                  [ OK ]
```

4. 再次检查 MySQL 服务器的状态。
   a. 在终端窗口中输入以下内容，得到的结果如下所示：

```
# service mysql status
MySQL is not running                                       [FAILED]
```

5. 再次启动 MySQL 服务器。
   a. 在终端窗口中输入以下内容，得到的结果如下所示：

```
# service mysql start
Starting MySQL                                             [ OK ]
```

   – 使服务器保持运行状态，因为将在后面的练习中使用该服务器。

6. 检查 MySQL 服务器的状态。
   a. 在终端窗口中输入以下内容，得到的结果如下所示：

```
# service mysql status
MySQL running (1401)                                       [ OK ]
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

第 4 课的练习：服务器配置

第 4 章

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

第 4 课的练习：概览

练习概览

通过这些练习，可测试您对 MySQL 服务器配置的掌握程度。上机练习是针对 Oracle Linux 操作系统环境编写的，Oracle 课堂中将提供此环境。对于非 Oracle 课堂环境，可能需要对文件位置进行一些调整。

假设

- 已安装 MySQL 服务器。
- 在终端窗口中以 `root` 用户身份登录。
- 您可以通过命令行提示符访问 `mysql` 客户机。
- 您可以熟练使用 Linux 中的 `gedit` 或其他文本编辑器。

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

练习 4-1：测验 – MySQL 服务器配置

概览

在本练习中，您将回答有关 MySQL 服务器配置的问题。

持续时间

完成本练习大约需要 10 分钟。

测验问题

请从所提供的每道多选题或正/误判断题中选择最佳答案。

1. 下列有关 SQL 模式的说法哪些是正确的？（请选择所有正确的选项。）
    a. 只能全局更改 `sql_mode` 的设置。
    b. 如果要设置两种 SQL 模式（例如，`STRICT_ALL_TABLES` 和 `ERROR_FOR_DIVISION_BY_ZERO` 模式），则必须发出两个 `SET sql_mode` 语句。
    c. 除非明确声明为全局，否则设置的 SQL 模式仅会影响设置该模式的客户机会话。
    d. SQL 模式会影响服务器的行为。例如，它们会影响服务器处理无效输入数据的方式。
    e. SQL 模式会影响服务器为客户机提供的功能。例如，您可以使用 SQL 模式启用和禁用 InnoDB 支持。
2. 以下将 SQL 模式更改为 `STRICT_TRANS_TABLES` 和 `PIPES_AS_CONCAT` 的语句语法是否正确？

```
mysql> SET sql_mode = 'STRICT_TRANS_TABLES,PIPES_AS_CONCAT';
```

    a. 正确
    b. 错误
3. 所有 MySQL 程序均会从名为 `my.ini`、`my.cnf` 或 `~/.my.cnf` 的纯文本选项文件中读取启动选项。
    a. 正确
    b. 错误
4. 如果使用 `-h 127.0.0.1` 选项调用 `mysql`，下列哪些说法是正确的？
    a. `mysql` 会建立到远程服务器的连接。
    b. `mysql` 将使用 TCP/IP 连接到本地实例。
    c. `mysql` 只在特定的操作系统上运行。
5. 在 Linux 中，服务器会将错误写入标准错误输出（通常为您的终端）。您可以通过使用 `--log-error=` *<file_name>* 选项启动服务器而改为将错误输出写入指定的文件。
    a. 正确
    b. 错误

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

6. 默认情况下，会以文本格式将错误日志写入到后缀为 _____ 的数据目录，并可以使用所有用于显示文本文件的程序进行查看。
    a. `.log`
    b. `.error_log`
    c. `.err`
    d. `.bin_err`
7. 可以在不指定任何选项的情况下启动 MySQL 服务器。要覆盖默认选项值，需要在命令行上指定新选项。
    a. 正确
    b. 错误
8. 关于二进制日志，下列说法正确的是：
    a. 将根据日志文件名随机启用二进制日志。
    b. 所有操作都记录在一个大型文件中。
    c. 将按升序使用数字扩展名创建二进制日志。
    d. 二进制日志以二进制格式（而非文本格式）存储。

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

练习解答 4-1：测验 — MySQL 服务器配置

测验解答

1. 下列有关 SQL 模式的说法哪些是正确的？

   c. 除非明确声明为全局，否则设置的 SQL 模式仅会影响设置该模式的客户机会话。

   d. SQL 模式会影响服务器的行为；例如，它们会影响服务器处理无效输入数据的方式。

2. 以下将 SQL 模式更改为 `STRICT_TRANS_TABLES` 和 `PIPES_AS_CONCAT` 的语句语法是否正确？

   ```
   mysql> SET sql_mode = 'STRICT_TRANS_TABLES,PIPES_AS_CONCAT';
   ```

   a. 正确

3. 所有 MySQL 程序均会从名为 `my.ini`、`my.cnf` 或 `~/.my.cnf` 的纯文本选项文件中读取启动选项。

   a. 正确

4. 假设使用 `-h 127.0.0.1` 选项调用 `mysql`，下列哪些说法是正确的？

   b. `mysql` 将使用 TCP/IP 连接到本地实例。

5. 在 Linux 中，服务器会将错误写入标准错误输出（通常为您的终端）。您可以通过使用 `--log-error=` *<file_name>* 选项启动服务器而改为将错误输出写入指定的文件。

   a. 正确

6. 默认情况下，会以文本格式将错误日志写入到后缀为 _____ 的数据目录，并可以使用所有用于显示文本文件的程序进行查看。

   c. `.err`

7. 可以在不指定任何选项的情况下启动 MySQL 服务器。要覆盖默认选项值，需要在命令行上指定新选项。

   b. 错误。还可以在启动选项文件中指定选项。

8. 关于二进制日志，下列说法正确的是：

   c. 将按升序使用数字扩展名创建二进制日志。

   d. 二进制日志以二进制格式（而非文本格式）存储。

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

练习 4-2：编辑和创建配置文件

概览

在本练习中，您将编辑 my.cnf 配置文件并创建新的配置文件。

持续时间

完成本练习大约需要 30 分钟。

任务

1. 为使用共同的基线配置，请复制 /etc 目录中的 my.cnf 配置文件模板的副本，并其置于 /root 中。
2. 找到 [mysqld] 部分，并将指定的默认端口更改为 3309。添加 [client] 部分，在其中指定端口 3309。
3. 在 [mysqld] 部分中，指示要启用下列日志：常规、二进制和慢速查询。将二进制文件的基名指定为 mybinlog。
4. 保存并关闭配置 (config) 文件。
5. 以 root 身份停止并重新启动服务器。
6. 确认在配置文件中指定的日志文件目前已创建。在 /var/lib/mysql 目录下查看这些日志文件。
7. 以 oracle 用户身份登录终端并在其中使用以下选项创建附加配置文件：口令 = oracle，用户名 = root，启用压缩模式，显示警告并添加特定的 mysql 提示符（以包括当前系统时间和数据库）。调用文件 my_opts.txt 并将其保存在主目录 (~/) 中。
8. 使用新的配置文件调用 mysql 客户机。
9. 确认新配置文件中的设置：
    a. 检查此 mysql 客户机会话正在使用的选项状态。
    b. 更改数据库设置，使用 world_innodb 数据库。
    c. 退出 mysql 会话。
10. 将 my_opts.txt 中除 password 设置之外的其他选项添加到默认配置文件 (/etc/my.cnf) 中，以便默认情况下每次调用客户机时均会使用这些选项。
11. 以 oracle 用户身份登录，调用 mysql 客户机而不指定新的配置文件。确认设置。
12. 以 oracle 用户身份登录，调用 mysql_config_editor，在默认登录路径中为 root 用户提供登录设置。
13. 使用 mysql_config_editor 显示当前用户的所有已存储登录路径。
14. 使用 **cat** 命令显示 ~/.mylogin.cnf 文件的内容。
15. 以 oracle 用户身份登录，调用 mysql 客户机而不提供任何命令行选项。成功登录后，退出 mysql 客户机。
16. 使用 mysql_config_editor 删除默认登录路径。

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

17. 再次调用 mysql_config_editor，为 root 用户提供登录设置，并将其存储在登录路径 **admin** 中。
18. 调用 mysql 客户机，指定登录路径 admin。
19. 通过将（第 1 步中所创建的）现有 /root/my.cnf 文件复制到 /etc 目录来创建默认配置文件的“干净”（原始）版本。保存新文件后，重新启动服务器。

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

练习解答 4-2：编辑和创建配置文件

任务

1. 为使用共同的基线配置，请复制 /etc 目录中的 my.cnf 配置文件模板的副本，并其置于 /root 中。

   a. 在终端窗口中输入以下内容：

```
$ su -
Password: oracle
```

   b. 将 my.cnf 文件复制到 /root。

```
# cp /etc/my.cnf /root
```

   c. 使用 gedit（或您选择的其他文本编辑器）打开 my.cnf 配置文件。

```
# gedit /etc/my.cnf
```

   如果您习惯使用其他编辑器，可使用 vim 或 emacs（或者其他编辑器）。

2. 找到 [mysqld] 部分并添加端口 3309。添加 [client] 部分，在其中指定端口 3309。修改 **my.cnf** 文件中的以下行：

```
[mysqld]
datadir=/var/lib/mysql
socket=/var/lib/mysql/mysql.sock
port=3309
user=mysql
...
[client]
port=3309
```

   – 在本例中，将侦听 TCP/IP 连接所使用的端口号设置为 3309。除非由 root 系统用户启动服务器，否则端口号必须为 1024 或更高。

3. 在 [mysqld] 部分中，指示要启用下列日志：常规、二进制和慢速查询。向 [mysqld] 部分添加以下行：

```
[mysqld]
...
symbolic-links=0
general_log
log-bin=mybinlog
slow_query_log
...
```

   – 下次重新启动服务器时，MySQL 将启用常规日志、二进制日志和慢速日志。

4. 保存并关闭配置 (config) 文件。

   – 在 gedit 中，依次按 Ctrl + S 和 Ctrl + Q。

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

5. 以 root 身份停止并重新启动服务器。
   a. 在终端窗口中输入以下内容，得到的结果如下所示：
   – 在停止服务器之前，请确保已退出当前正在运行的所有 mysql 客户机。

```
# service mysql restart
Shutting down MySQL....                                    [ OK ]
Starting MySQL....                                         [ OK ]
```

6. 确认在配置文件中指定的日志文件目前已创建。在 /var/lib/mysql 目录下查看这些日志文件。在终端窗口中输入以下内容，得到的结果如下所示：

```
# ls /var/lib/mysql
auto.cnf            ib_logfile0      mysql.sock
host-name.log       ib_logfile1      performance_schema
host-name.pid       mybinlog.000001  RPM_UPGRADE_HISTORY
host-name-slow.log  mybinlog.index   RPM_UPGRADE_MARKER-LAST
ibdata1             mysql            world_innodb
```

7. 以 oracle 用户身份登录终端窗口并在其中使用以下选项创建附加配置文件：口令 = oracle，用户名 = root，启用压缩模式，显示警告并添加特定的 mysql 提示符。调用文件 my_opts.txt 并将其保存在主目录 (~/) 中。
   a. 在 ~ 目录（如果以 oracle 身份登录，则为 /home/oracle）中，使用 gedit（或您选择的其他文本编辑器）创建 my_opts.txt 配置文件。

```
$ cd
$ pwd
/home/oracle
$ gedit my_opts.txt
```

   b. 向 my_opts.txt 文件添加以下内容：

```
[client]
password = oracle
user = root

[mysql]
compress
show-warnings
prompt = \R:\m \d>\_
```

   c. 保存并关闭新配置文件。

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

8. 使用 my_opts.txt 文件调用 mysql 客户机。在终端窗口中输入以下内容，得到的结果如下所示：

```
$ mysql --defaults-extra-file=~/my_opts.txt
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 1
Server version: 5.6.10-enterprise-commercial-advanced-log MySQL
Enterprise Server - Advanced Edition (Commercial)

Copyright (c) 2000, 2013, Oracle and/or its affiliates. All rights
reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input
statement.

10:21 (none)>
```

   – 现在提示符反映了当前的系统时间，并显示当前没有任何数据库正在使用中。

9. 确认新配置文件中的设置：

   a. 检查此 mysql 客户机会话正在使用的选项状态。在 mysql 提示符下输入以下内容，得到的结果如下所示：

```
10:21 (none)> STATUS
--------------
mysql  Ver 14.14 Distrib 5.6.10, for Linux (x86_64) using  EditLine
wrapper

Connection id:          1
Current database:
Current user:           root@localhost
SSL:                    Not in use
Current pager:          stdout
Using outfile:          ''
Using delimiter:        ;
Server version:         5.6.10-enterprise-commercial-advanced-log
MySQL Enterprise Server - Advanced Edition (Commercial)
Protocol version:       10
Connection:             Localhost via UNIX socket
Server characterset:    latin1
Db     characterset:    latin1
Client characterset:    utf8
Conn.  characterset:    utf8
UNIX socket:            /var/lib/mysql/mysql.sock
Protocol:               Compressed
Uptime:                 8 min 40 sec
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

```
Threads: 1  Questions: 8  Slow queries: 0  Opens: 70  Flush tables: 1
Open tables: 63  Queries per second avg: 0.015
--------------
```

– 显示了服务器连接规格列表，其中包括选项文件中的特定设置。请检查列表中的相应选项设置。

b. 更改数据库设置，使用 world_innodb 数据库。在新更改的提示符下输入以下内容，得到的结果如下所示：

```
10:22 (none)> USE world_innodb
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
```

c. 退出 mysql 会话，得到的结果如下所示：

```
10:23 world_innodb> EXIT
Bye
$
```

10. 将 my_opts.txt 中除 password 设置之外的其他选项添加到默认配置文件 (/etc/my.cnf) 中，以便默认情况下每次调用客户机时均会使用这些选项。以 root 身份登录，并向现有的 /etc/my.cnf 文件添加以下内容：

```
...
[client]
port=3309
user = root

[mysql]
compress
show-warnings
prompt = \R:\m \d>\_
```

– 将口令放在默认选项文件中并不安全，因此不是好做法。

– 更改客户机选项后，不必重新启动 mysql 服务器。

11. 以 oracle 用户身份登录，调用 mysql 客户机而不指定新的配置文件。确认设置。

以 oracle 用户身份登录终端窗口并在其中输入以下内容，得到的结果如下所示：

```
$ mysql -p
Enter password: oracle
Welcome to the MySQL monitor. Commands end with ; or \g.
Your MySQL connection id is 2
Server version: 5.6.8-enterprise-commercial-advanced-log MySQL
Enterprise Server - Advanced Edition (Commercial)
12:01 (none)>
```

– 不需要为 **user** 提供值，因为该值已硬编码到配置文件中。

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

12. 以 oracle 用户身份登录，调用 mysql_config_editor，在默认登录路径中为 root 用户提供登录设置。

 以 oracle 用户身份登录新的终端窗口并在其中输入以下内容，得到的结果如下所示：

```
$ mysql_config_editor set --user=root --password
Enter password: oracle
```

13. 使用 mysql_config_editor 显示当前用户的所有已存储登录路径。以 oracle 用户身份登录终端窗口并在其中输入以下内容，得到的结果如下所示：

```
$ mysql_config_editor print --all
[client]
user = root
password = *****
```

   – 该命令将以纯文本格式显示新创建的 ~/.mylogin.cnf 文件的内容，并遮蔽口令。

14. 使用 cat 命令显示 ~/.mylogin.cnf 文件的内容。以 oracle 用户身份登录终端窗口并在其中输入以下内容，得到的结果如下所示：

```
$ cat ~/.mylogin.cnf
     ________________________________-
__________________________________________________________________________
     2□U□1□▯ □OB@1@□□`{□□□$□}Nq □  □E□  Aq□□-
□□)^□|{□j□□s□x□□W□$
```

   – 文件内容不可读，无法看到其中存储的用户名或口令。

15. 以 oracle 用户身份登录，调用 mysql 客户机而不提供任何命令行选项。成功登录后，退出 mysql 客户机。

   a. 以 oracle 用户身份登录终端窗口并在其中输入以下内容，得到的结果如下所示：

```
$ mysql
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 3
Server version: 5.6.10-enterprise-commercial-advanced-log MySQL
Enterprise Server - Advanced Edition (Commercial)
...
```

   – 使用在第 12 步中提供的用户名和口令登录。如果没有提供命令行选项，mysql 客户机将使用默认登录路径。由于在第 12 步中没有指定登录路径名称，因此设置存储在默认路径下。

   b. 要退出，请在 mysql 提示符下输入以下内容：

```
10:36 (none)> EXIT
Bye
```

16. 使用 mysql_config_editor 删除默认登录路径。以 oracle 用户身份登录终端窗口并在其中输入以下内容，得到的结果如下所示：

```
$ mysql_config_editor remove
WARNING : No login path specified, so default login path will be
removed.
 Continue? (Press y|Y for Yes, any other key for No) : y
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

17. 再次调用 mysql_config_editor，为 root 用户提供登录路径设置，并将其存储在登录路径 admin 下。以 oracle 用户身份登录终端窗口并在其中输入以下内容，得到的结果如下所示：

```
$ mysql_config_editor set --login-path=admin --user=root --password
Enter password: oracle
```

18. 调用 mysql 客户机，指定登录路径 admin。

```
$ mysql --login-path=admin
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 4
...
```

- 使用在第 17 步中提供的用户名和口令登录。使用 --login-path 选项时，mysql 客户机将使用存储在该登录路径下的凭证。

要退出，请在 mysql 提示符下输入以下内容：

```
10:39 (none)> EXIT
Bye
```

19. 通过将（第 1 步中所创建的）现有 /root/my.cnf 文件复制到 /etc 目录来创建默认配置文件的“干净”（原始）版本。保存新文件后，重新启动服务器。

a. 以 root 身份登录终端窗口并在其中输入以下内容：

```
# cp /root/my.cnf /etc/
cp: overwrite `/etc/my.cnf'? y
```

- 这样做的部分原因是要恢复默认端口设置。如果保留端口设置 3309（或其他非默认端口），则必须指定用于所有客户机连接的端口，但这样做可能需要较长时间并且可能会导致错误。现在将其恢复为默认端口可避免日后出现误解。

b. 重新启动服务器。在终端窗口中输入以下内容，得到的结果如下所示：

```
# service mysql restart
Shutting down MySQL......                                  [ OK ]
Starting MySQL......                                     [ OK ]
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

练习 4-3：附加练习 — 服务器配置

概览

在本练习中，您将用到 world_innodb 数据库以及本课程中与 MySQL 服务器配置相关的信息。

持续时间

完成本练习大约需要 40 分钟。

任务

1. 使用 SHOW GLOBAL VARIABLES LIKE '%log' 语句检查是否已启用常规和慢速查询日志。如果尚未启用，请使用 SET GLOBAL 语句启用这些日志文件。
2. 通过检查 log_output 变量值，确保这些日志基于表。如果需要，可将该变量的值设置为 **TABLE**，然后使用 TRUNCATE 语句清空表。
3. 为 world_innodb 数据库创建名为 world2 的新副本。要完成此操作，请创建一个名为 world2 的新数据库，并在连接到 world2 时再次获取 world_innodb.sql 数据库文件的数据。
4. 使用以下 SELECT 语句来计算常规日志中的 CREATE TABLE 语句数：

```
mysql> SELECT COUNT(*) FROM mysql.general_log
    -> WHERE argument LIKE 'CREATE TABLE%';
```

5. 创建将记录到慢速查询日志中的查询：
   a. 示例：需要较长时间的 SELECT

```
mysql> SELECT SLEEP(11);
```

   b. 在 mysql 数据库的 slow_log 表中查看此查询。
6. 通过检查 log_bin 变量值，确定是否已启用二进制日志记录。然后退出当前 mysql 会话。
7. 编辑 my.cnf 文件（位于 /etc 目录中）的 [mysqld] 部分以启用二进制日志记录。保存所编辑的内容后，停止并重新启动服务器。
8. 在新的 mysql 会话中，确保已启用二进制日志，并使用 RESET MASTER 语句清除之前的所有日志（如果存在）。
9. 执行数据更改操作：
   a. 例如，创建和删除数据库：

```
mysql> CREATE DATABASE foo;
mysql> DROP DATABASE foo;
```

   b. 使用 SHOW BINLOG EVENTS 语句列出二进制日志中的此操作。
10. 使用 FLUSH BINARY LOGS 显式轮转二进制日志，然后执行其他的数据更改查询（类似于上述步骤）。
11. 使用相应的 SHOW 语句列出所有二进制日志（对于 mysql-bin.000002）以及最近的更改。
12. 清除第一个二进制日志。
13. 执行其他的数据修改查询（类似于上述步骤），但这些查询语句使用数据库名称 foo2。
14. 使用 mysqlbinlog 显示二进制日志中在第 11 步中显示的条目之后发生的所有条目。

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

15. 查看 mysql-bin.000002 二进制日志文件，找到刷新日志后所发生事件的起始部分。删除 foo2 数据库的事件编号是多少？
16. 在 mysql 提示符下装入审计日志插件。
17. 执行一些修改语句（类似于上述步骤）。
18. 查看 audit_log_file 系统变量的值，然后使用该值查找审计日志并查看其内容。
19. 如果出现任何警告，则卸载审计日志插件。
20. 执行一些修改语句（类似于上述步骤）。
21. 退出当前 mysql 会话。
22. 查看审计日志的最后几行。
23. 配置 /etc/my.cnf 文件以在服务器启动时启用审计日志插件并防止它在运行期间被卸载。完成此操作后，重新启动服务器。
24. 尝试卸载审计日志插件，并记录结果。
25. 再次查看审计日志文件，并注意任何更改。
26. 在 /etc/my.cnf 文件中删除与审计日志相关的所有设置，然后重新启动 MySQL 服务器。

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

练习解答 4-3：附加练习 — 服务器配置

任务

1. 使用 SHOW GLOBAL VARIABLES LIKE '%log' 语句检查是否已启用常规和慢速查询日志。如果尚未启用，请使用 SET GLOBAL 语句启用这些日志文件。

   a. 以 oracle 身份登录终端并在其中调用 mysql 客户机，指定登录路径 admin。

```
$ mysql --login-path=admin
Your MySQL connection id is 1
...
```

   b. 查看以“log”结尾的状态变量。

   在 mysql 提示符下输入以下内容，得到的结果如下所示：

```
mysql> SHOW GLOBAL VARIABLES LIKE '%log';
+---------------------------------+-------+
| Variable_name                   | Value |
+---------------------------------+-------+
| back_log                        | 80    |
| general_log                     | OFF   |
| innodb_api_enable_binlog        | OFF   |
| innodb_locks_unsafe_for_binlog | OFF   |
| relay_log                       |       |
| slow_query_log                  | OFF   |
| sync_binlog                     | 0     |
| sync_relay_log                  | 10000 |
+---------------------------------+-------+
8 rows in set (0.00 sec)
```

   c. 启用日志文件。输入以下内容，得到的结果如下所示：

```
mysql> SET GLOBAL general_log = ON;
Query OK, 0 rows affected (0.06 sec)
mysql> SET GLOBAL slow_query_log = ON;
Query OK, 0 rows affected (0.07 sec)
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

d. 查看以“log”结尾的新状态变量。在 mysql 提示符下输入以下内容，得到的结果如下所示：

```
mysql> SHOW GLOBAL VARIABLES LIKE '%log';
+---------------------------------+-------+
| Variable_name                   | Value |
+---------------------------------+-------+
| back_log                        | 80    |
| general_log                     | ON    |
| innodb_api_enable_binlog        | OFF   |
| innodb_locks_unsafe_for_binlog | OFF   |
| relay_log                       |       |
| slow_query_log                  | ON    |
| sync_binlog                     | 0     |
| sync_relay_log                  | 10000 |
+---------------------------------+-------+
8 rows in set (0.00 sec)
```

2. 通过检查 log_output 变量值，确保这些日志基于表。如果需要，可将该变量的值设置为 TABLE，然后使用 TRUNCATE 语句清空表。

a. 检查 log_output 变量的值。输入以下内容，得到的结果如下所示：

```
mysql> SHOW GLOBAL VARIABLES LIKE 'log_output';
+---------------+-------+
| Variable_name | Value |
+---------------+-------+
| log_output    | FILE  |
+---------------+-------+
1 row in set (0.00 sec)
```

b. 如果未将该变量的值设置为 TABLE，则键入以下内容，得到的结果如下所示：

```
mysql> SET GLOBAL log_output = 'TABLE';
Query OK, 0 rows affected (0.01 sec)
```

c. 确认 log_output 变量现在已设置为 TABLE。输入以下内容，得到的结果如下所示：

```
mysql> SHOW GLOBAL VARIABLES LIKE 'log_output';
+---------------+--------+
| Variable_name | Value  |
+---------------+--------+
| log_output    | TABLE  |
+---------------+--------+
1 row in set (0.00 sec)
```

d. 清空日志的表。输入以下内容，得到的结果如下所示：

```
mysql> TRUNCATE mysql.general_log;
Query OK, 0 rows affected (0.01 sec)

mysql> TRUNCATE mysql.slow_log;
Query OK, 0 rows affected (0.01 sec)
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

3. 为 world_innodb 数据库创建名为 world2 的新副本。要完成此操作，请创建一个名为 world2 的新数据库，并在连接到 world2 时再次获取 **world_innodb.sql** 数据库文件的数据。

   a. 创建和填充 world2 数据库。输入以下内容，得到的结果如下所示：

```
mysql> CREATE DATABASE world2;
Query OK, 1 row affected (0.02 sec)

mysql> USE world2;
Database changed
mysql> SET autocommit=0;
Query OK, 0 rows affected (0.00 sec)

mysql> SOURCE /labs/world_innodb.sql
Query OK ...
...
Query OK, 0 rows affected (0.00 sec)

mysql> SET autocommit=1;
Query OK, 0 rows affected (0.04 sec)
```

   – 重置 autocommit 选项可加快导入过程。

4. 使用以下 SELECT 语句来计算常规日志中的 CREATE TABLE 语句数。输入以下内容，得到的结果如下所示：

```
mysql> SELECT COUNT(*) FROM mysql.general_log
    -> WHERE argument LIKE 'CREATE TABLE%';
+----------+
| COUNT(*) |
+----------+
| 3        |
+----------+
```

5. 创建将记录到慢速查询日志中的查询：

   a. 示例：需要较长时间的 SELECT。输入以下内容，得到的结果如下所示：

```
mysql> SELECT SLEEP(11);
+------------+
| SLEEP (11) |
+------------+
|          0 |
+------------+
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

b. 在 mysql 数据库的 slow_log 表中查看此查询。输入以下查询，得到的结果如下所示：

```
mysql> SELECT * FROM mysql.slow_log\G
*************************** 1. row ***************************
    start_time: 2013-02-01 11:06:36
     user_host: root[root] @ localhost []
    query_time: 00:00:11
     lock_time: 00:00:00
     rows_sent: 1
 rows_examined: 0
            db: world2
last_insert_id: 0
     insert_id: 0
     server_id: 0
      sql_text: SELECT SLEEP(11)
     thread_id: 1
1 row in set (0.00 sec)
```

6. 通过检查 log_bin 变量值，确定是否已启用二进制日志记录：

a. 检查日志的状态。输入以下内容，得到的结果如下所示：

```
mysql> SHOW GLOBAL VARIABLES LIKE 'log_bin';
+---------------+-------+
| Variable_name | Value |
+---------------+-------+
| log_bin       | OFF   |
+---------------+-------+
```

b. 然后退出当前 mysql 会话。

```
mysql> EXIT
Bye
```

7. 编辑 my.cnf 文件（位于 /etc 目录中）的 [mysqld] 部分以启用二进制日志记录。保存所编辑的内容后，停止并重新启动服务器。

a. 以 root 身份登录终端并在其中使用 gedit（或您选择的其他文本编辑器）打开 my.cnf 配置文件。

```
# gedit /etc/my.cnf
```

b. 向 [mysqld] 部分添加以下用于控制日志更新的行：

```
[mysqld]
log-bin=mysql-bin
datadir=/var/lib/mysql
...
```

c. 保存并关闭配置文件。

d. 保存所编辑的内容后，停止并重新启动服务器。在终端窗口中输入以下内容，得到的结果如下所示：

```
# service mysql restart
Shutting down MySQL...                                     [  OK  ]
Starting MySQL.                                            [  OK  ]
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

8. 在新的 mysql 会话中，确保已启用二进制日志，并使用 RESET MASTER 语句清除之前的所有日志（如果存在）。

   a. 以 oracle 身份登录 Linux 终端并在其中启动 mysql 客户机：

```
$ mysql --login-path=admin
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 1
...
```

   b. 检查日志的状态。输入以下内容，得到的结果如下所示：

```
mysql> SHOW GLOBAL VARIABLES LIKE 'log_bin';
+---------------+-------+
| Variable_name | Value |
+---------------+-------+
| log_bin       | ON    |
+---------------+-------+
```

   c. 清除所有日志。输入以下内容，得到的结果如下所示：

```
mysql> RESET MASTER;
Query OK, 0 row affected (0.09 sec)
```

9. 执行数据更改操作，并列出二进制日志中的该操作。

   a. 例如，创建和删除数据库。输入以下内容，得到的结果如下所示：

```
mysql> CREATE DATABASE foo;
Query OK, 1 row affected (0.00 sec)

mysql> DROP DATABASE foo;
Query OK, 0 row affected (0.00 sec)
```

   b. 使用 SHOW BINLOG EVENTS 语句列出二进制日志中的此操作。输入以下内容，得到的结果如下所示：

```
mysql> SHOW BINLOG EVENTS;
+------------------+-----+-------------+-----------+----------+----------------------
| Log_name         | Pos | Event_type  | Server_id | End_log… | Info
+------------------+-----+-------------+-----------+----------+----------------------
| mysql-bin.000001 |   4 | Format_desc |         1 |      120 | Server ver: 5.6.10..
| mysql-bin.000001 | 120 | Query       |         1 |      211 | CREATE DATABASE foo
| mysql-bin.000001 | 211 | Query       |         1 |      294 | DROP DATABASE foo
+------------------+-----+-------------+-----------+----------+----------------------
3 rows in set (0.00 sec)
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

10. 使用 FLUSH BINARY LOGS 显式轮转二进制日志，然后执行其他的数据更改查询（类似于上述步骤）。

    a. 刷新二进制日志。输入以下内容：

```
mysql> FLUSH BINARY LOGS;
Query OK, 0 row affected (0.04 sec)
```

    b. 更改数据。输入以下内容，得到的结果如下所示：

```
mysql> CREATE DATABASE foo;
Query OK, 1 row affected (0.00 sec)

mysql> DROP DATABASE foo;
Query OK, 0 row affected (0.02 sec)
```

11. 使用相应的 SHOW 语句列出所有二进制日志（对于 mysql-bin.000002）以及最近的更改。

    a. 列出日志。输入以下内容，得到的结果如下所示：

```
mysql> SHOW MASTER LOGS;
+------------------+-----------+
| Log_name         | File_size |
+------------------+-----------+
| mysql-bin.000001 |       341 |
| mysql-bin.000002 |       294 |
+------------------+-----------+
2 rows in set (0.00 sec)
```

    b. 列出更改。输入以下内容，得到的结果如下所示：

```
mysql> SHOW BINLOG EVENTS IN 'mysql-bin.000002';
+------------------+-----+-------------+-----------+----------+----------------------
| Log_name         | Pos | Event_type  | Server_id | End_log… | Info
+------------------+-----+-------------+-----------+----------+----------------------
| mysql-bin.000002 |   4 | Format_desc |         1 |      120 | Server ver: 5.6.10..
| mysql-bin.000002 | 120 | Query       |         1 |      211 | CREATE DATABASE foo
| mysql-bin.000002 | 211 | Query       |         1 |      294 | DROP DATABASE foo
+------------------+-----+-------------+-----------+----------+----------------------
3 rows in set (0.00 sec)
```

12. 清除第一个二进制日志。输入以下内容，得到的结果如下所示：

```
mysql> PURGE MASTER LOGS TO 'mysql-bin.000002';
Query OK, 0 row affected (0.05 sec)
```

13. 执行其他的数据修改查询（类似于上述步骤），但这些查询语句使用数据库名称 foo2。输入以下内容，得到的结果如下所示：

```
mysql> CREATE DATABASE foo2;
Query OK, 1 row affected (0.00 sec)

mysql> DROP DATABASE foo2;
Query OK, 0 row affected (0.00 sec)
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

14. 使用 mysqlbinlog 显示二进制日志中在第 11 步中显示的条目之后发生的所有条目。

以 root 身份登录终端窗口并在其中输入以下内容，得到的结果如下所示：

```
# mysqlbinlog /var/lib/mysql/mysql-bin.000002
/*!50530 SET @@SESSION.PSEUDO_SLAVE_MODE=1*/;
/*!40019 SET @@session.max_insert_delayed_threads=0*/;
/*!50003 SET
@OLD_COMPLETION_TYPE=@@COMPLETION_TYPE,COMPLETION_TYPE=0*/;
DELIMITER /*!*/;
# at 4
#130202 12:02:05 server id 1  end_log_pos 120 CRC32 0x061acade  Start:
binlog v 4, server v 5.6.10-enterprise-commercial-advanced-log created
130202 12:02:05
# Warning: this binlog is either in use or was not closed properly.
BINLOG '
PQANUQ8BAAAAdAAAAHgAAAABAAQANS42LjEwLWVudGVycHJpc2UtY29tbWVyY2lhbC1hZH
ZhbmNl
ZC1sb2cAAAAAAAAAAAAAAAAAAEzgNAAgAEgAEBAQEEgAAXAAEGggAAAAICAgCAAAACgoKGR
kAAd7K
GgY=
'/*!*/;
# at 120
#130202 12:02:15 server id 1  end_log_pos 211 CRC32 0x9d08e28b  Query
thread_id=1     exec_time=0     error_code=0
SET TIMESTAMP=1359806535/*!*/;
SET @@session.pseudo_thread_id=1/*!*/;
SET @@session.foreign_key_checks=1, @@session.sql_auto_is_null=0,
@@session.unique_checks=1, @@session.autocommit=1/*!*/;
SET @@session.sql_mode=1075838976/*!*/;
SET @@session.auto_increment_increment=1,
@@session.auto_increment_offset=1/*!*/;
/*!\C utf8 *//*!*/;
SET
@@session.character_set_client=33,@@session.collation_connection=33,@@
session.collation_server=8/*!*/;
SET @@session.lc_time_names=0/*!*/;
SET @@session.collation_database=DEFAULT/*!*/;
CREATE DATABASE foo
/*!*/;
# at 211
#130202 12:02:21 server id 1  end_log_pos 294 CRC32 0x24ec5ec1  Query
thread_id=1     exec_time=0     error_code=0
SET TIMESTAMP=1359806541/*!*/;
DROP DATABASE foo
/*!*/;
# at 294
#130202 12:06:19 server id 1  end_log_pos 388 CRC32 0xa8736010  Query
thread_id=1     exec_time=0     error_code=0
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

```
SET TIMESTAMP=1359806779/*!*/;
CREATE DATABASE foo2
/*!*/;
# at 388
#130202 12:06:19 server id 1  end_log_pos 473 CRC32 0xee26c05a  Query
thread_id=1     exec_time=0     error_code=0
SET TIMESTAMP=1359806779/*!*/;
DROP DATABASE foo2
/*!*/;
DELIMITER ;
# End of log file
ROLLBACK /* added by mysqlbinlog */;
/*!50003 SET COMPLETION_TYPE=@OLD_COMPLETION_TYPE*/;
/*!50530 SET @@SESSION.PSEUDO_SLAVE_MODE=0*/;
```

15. 查看 mysql-bin.000002 二进制日志文件，找到刷新日志后所发生事件的起始部分。删除 foo2 数据库的事件编号是多少？388
    - 事件位置编号与二进制日志文件名结合起来可确定特定事件。
16. 在 mysql 提示符下装入审计日志插件。输入以下内容，得到的结果如下所示：

```
mysql> INSTALL PLUGIN audit_log SONAME 'audit_log.so';
Query OK, 0 rows affected (0.19 sec)
```

17. 执行一些修改语句（类似于上述步骤）。输入以下内容，得到的结果如下所示：

```
mysql> CREATE DATABASE foo;
Query OK, 1 row affected (0.00 sec)

mysql> DROP DATABASE foo;
Query OK, 0 row affected (0.00 sec)
```

18. 查看 audit_log_file 系统变量的值，然后使用该值查找审计日志并查看其内容。
    a. 输入以下内容，得到的结果如下所示：

```
mysql> SHOW VARIABLES LIKE 'audit_log_file';
+----------------+-----------+
| Variable_name  | Value     |
+----------------+-----------+
| audit_log_file | audit.log |
+----------------+-----------+
1 row in set (0.20 sec)
```

    - audit.log 文件位于服务器的数据目录中。

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

b. 以 root 身份登录 Linux 终端窗口并在其中输入以下内容，得到的结果如下所示：

```
# cat /var/lib/mysql/audit.log
<?xml version="1.0" encoding="UTF-8"?>
<AUDIT>
  <AUDIT_RECORD TIMESTAMP="2013-02-02T12:22:45" NAME="Audit"
SERVER_ID="1" VERSION="1" STARTUP_OPTIONS="/usr/sbin/mysqld --
basedir=/usr --datadir=/var/lib/mysql --plugin-
dir=/usr/lib64/mysql/plugin --user=mysql --log-
error=/var/log/mysqld.log --pid-file=/var/lib/mysql/host-name.pid --
socket=/var/lib/mysql/mysql.sock" OS_VERSION="x86_64-Linux"
MYSQL_VERSION="5.6.10-enterprise-commercial-advanced-log"/>
  <AUDIT_RECORD TIMESTAMP="2013-02-02T12:22:45" NAME="Query"
CONNECTION_ID="1" STATUS="0" SQLTEXT="INSTALL PLUGIN audit_log SONAME
'audit_log.so'"/>
  <AUDIT_RECORD TIMESTAMP="2013-02-02T12:23:22" NAME="Query"
CONNECTION_ID="1" STATUS="0" SQLTEXT="CREATE DATABASE foo"/>
  <AUDIT_RECORD TIMESTAMP="2013-02-02T12:23:25" NAME="Query"
CONNECTION_ID="1" STATUS="0" SQLTEXT="DROP DATABASE foo"/>
  <AUDIT_RECORD TIMESTAMP="2013-02-02T12:23:25" NAME="Query"
CONNECTION_ID="1" STATUS="0" SQLTEXT="SELECT DATABASE()"/>
  <AUDIT_RECORD TIMESTAMP="2013-02-02T12:23:51" NAME="Query"
CONNECTION_ID="1" STATUS="0" SQLTEXT="SHOW VARIABLES LIKE
'audit_log_file'"/>
```

19. 如果出现任何警告，则卸载审计日志插件。输入以下内容，得到的结果如下所示：

```
mysql> UNINSTALL PLUGIN audit_log;
Query OK, 0 rows affected, 1 warning (0.00 sec)

mysql> SHOW WARNINGS;
+---------+------+-----------------------------------------------------+
| Level   | Code | Message                                             |
+---------+------+-----------------------------------------------------+
| Warning | 1620 | Plugin is busy and will be uninstalled on shutdown |
+---------+------+-----------------------------------------------------+
1 row in set (0.00 sec)
```

20. 执行一些修改语句（类似于上述步骤）。输入以下内容，得到的结果如下所示：

```
mysql> CREATE DATABASE foo;
Query OK, 1 row affected (0.00 sec)

mysql> DROP DATABASE foo;
Query OK, 0 row affected (0.00 sec)
```

21. 退出当前 mysql 会话。输入以下内容，得到的结果如下所示：

```
mysql> EXIT
Bye
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

22. 查看审计日志的最后几行。以 root 身份登录 Linux 终端窗口并在其中输入以下内容，得到的结果如下所示：

```
# cat /var/lib/mysql/audit.log
<?xml version="1.0" encoding="UTF-8"?>
<AUDIT>
...
  <AUDIT_RECORD TIMESTAMP="2013-02-02T12:23:25" NAME="Query"
CONNECTION_ID="1" STATUS="0" SQLTEXT="DROP DATABASE foo"/>
  <AUDIT_RECORD TIMESTAMP="2013-02-02T12:23:25" NAME="Query"
CONNECTION_ID="1" STATUS="0" SQLTEXT="SELECT DATABASE()"/>
  <AUDIT_RECORD TIMESTAMP="2013-02-02T12:23:51" NAME="Query"
CONNECTION_ID="1" STATUS="0" SQLTEXT="SHOW VARIABLES LIKE
'audit_log_file'"/>
  <AUDIT_RECORD TIMESTAMP="2013-02-02T12:27:31" NAME="Query"
CONNECTION_ID="1" STATUS="0" SQLTEXT="UNINSTALL PLUGIN audit_log"/>
  <AUDIT_RECORD TIMESTAMP="2013-02-02T12:27:31" NAME="NoAudit"
SERVER_ID="1"/>
</AUDIT>
```

– XML 文件包含结束标记，可结束 AUDIT 根元素，其中没有最终修改语句记录。

23. 配置 /etc/my.cnf 文件以在服务器启动时启用审计日志插件并防止它在运行期间被卸载。完成此操作后，重新启动服务器。

a. 按如下所示编辑文件 /etc/my.cnf：

```
[mysqld]
plugin-load=audit_log.so
audit-log=FORCE_PLUS_PERMANENT
log-bin=mysql-bin
datadir=/var/lib/mysql
...
```

b. 在终端窗口中输入以下内容，得到的结果如下所示：

```
# service mysql restart
Shutting down MySQL...                                     [  OK  ]
Starting MySQL.                                            [  OK  ]
```

24. 以 root 身份重新连接到 mysql。尝试卸载审计日志插件，并记录结果。输入以下内容，得到的结果如下所示：

```
$ mysql --login-path=admin
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 1
Server version: 5.6.10-enterprise-commercial-advanced-log MySQL
Enterprise Server - Advanced Edition (Commercial)
...
mysql> UNINSTALL PLUGIN audit_log;
ERROR 1702 (HY000): Plugin 'audit_log' is force_plus_permanent and can
not be unloaded
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

25. 再次查看审计日志文件，并注意任何更改。以 root 身份在终端提示符下输入以下内容，得到的结果如下所示：

```
# cat /var/lib/mysql/audit.log
<?xml version="1.0" encoding="UTF-8"?>
<AUDIT>
...
  <AUDIT_RECORD TIMESTAMP="2013-02-02T12:27:31" NAME="Query"
CONNECTION_ID="2" STATUS="0" SQLTEXT="UNINSTALL PLUGIN audit_log"/>
  <AUDIT_RECORD TIMESTAMP="2013-02-02T12:27:31" NAME="NoAudit"
SERVER_ID="1"/>
  <AUDIT_RECORD TIMESTAMP="2013-02-02T12:35:00" NAME="Audit"
SERVER_ID="1" VERSION="1" STARTUP_OPTIONS="/usr/sbin/mysqld --
basedir=/usr --datadir=/var/lib/mysql --plugin-
dir=/usr/lib64/mysql/plugin --user=mysql --log-
error=/var/log/mysqld.log --pid-file=/var/lib/mysql/EDTDR20P1.pid --
socket=/var/lib/mysql/mysql.sock" OS_VERSION="x86_64-Linux"
MYSQL_VERSION="5.6.10-enterprise-commercial-advanced-log"/>
  <AUDIT_RECORD TIMESTAMP="2013-02-02T12:39:16" NAME="Connect"
CONNECTION_ID="1" STATUS="0" USER="root" PRIV_USER="root" OS_LOGIN=""
PROXY_USER="" HOST="localhost" IP="" DB=""/>
  <AUDIT_RECORD TIMESTAMP="2013-02-02T12:39:16" NAME="Query"
CONNECTION_ID="1" STATUS="0" SQLTEXT="select @@version_comment limit
1"/>
  <AUDIT_RECORD TIMESTAMP="2013-02-02T12:40:05" NAME="Query"
CONNECTION_ID="1" STATUS="1702" SQLTEXT="UNINSTALL PLUGIN audit_log"/>
```

- 审计日志插件已重新打开 XML 文件，该文件记录了第二次尝试卸载该插件的操作及其错误编号，但并没有卸载该插件。

26. 在 /etc/my.cnf 文件中删除与审计日志相关的所有设置，然后重新启动 MySQL 服务器。

a. 从 /etc/my.cnf 文件中删除以下行：

```
plugin-load=audit_log.so
audit-log=FORCE_PLUS_PERMANENT
```

文件的顶部应该类似于以下内容：

```
[mysqld]
log-bin=mysql-bin
datadir=/var/lib/mysql
...
```

b. 重新启动 MySQL 服务器。在终端窗口中输入以下内容，得到的结果如下所示：

```
# service mysql restart
Shutting down MySQL...                                     [  OK  ]
Starting MySQL.                                            [  OK  ]
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

第 5 课的练习：客户机和工具

第 5 章

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

第 5 课的练习：概览

练习概览

通过这些练习，可测试您对 MySQL 客户机和工具的掌握程度。上机练习是针对 Oracle Linux 操作系统环境编写的，Oracle 课堂中将提供此环境。对于非 Oracle 课堂环境，可能需要对文件位置进行一些调整。

假设

- 在本练习之前已经完成了 MySQL 服务器安装。
- 在终端窗口中以 `oracle` 用户身份登录。

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

练习 5-1：调用 mysql 客户机

概览

在本练习中，您将使用某些常见选项调用 `mysql` 客户机。

持续时间

完成本练习大约需要 10 分钟。

任务

1. 显示本地主机上的 `mysql` 客户机版本。
2. 显示 `mysql` 客户机选项。
3. 使用用户名和口令选项启动 `mysql` 客户机。退出客户机。
4. 使用 `admin` 登录路径启动 `mysql` 客户机，并在执行命令行时包含 `SELECT` 语句（以 HTML 格式显示当前日期和时间）。
5. 使用用户名、端口（设置为 3306）以及用于此会话的 tee 文件启动 `mysql` 客户机。允许客户机请求口令。出现请求时输入客户机口令。退出客户机并确认已创建 tee 文件。
6. 以安全更新模式启动 `mysql` 客户机。
7. 在 `mysql` 客户机中，显示会话状态。
8. 列出服务器端帮助：
    a. 找到整体类别。
    b. 列出与帐户管理相关的命令。
    c. 显示与设置口令相关的信息。
9. 显示可用数据库。退出客户机。

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

练习解答 5-1：调用 mysql 客户机

任务

1. 显示本地主机上的 mysql 客户机版本。在终端窗口中输入以下内容，得到的结果如下所示：

```
$ mysql -V
mysql  Ver 14.14 Distrib 5.6.10, for Linux (x86_64) using  EditLine wrapper
```

2. 显示 mysql 客户机选项。在终端窗口中输入以下内容，得到的结果如下所示：

```
$ mysql --help
mysql  Ver 14.14 Distrib 5.6.10, for Linux (x86_64) using  EditLine
wrapper
Copyright (c) 2000, 2013, Oracle and/or its affiliates. All rights
reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Usage: mysql [OPTIONS] [database]
  -?, --help          Display this help and exit.
  -I, --help          Synonym for -?
  --auto-rehash       Enable automatic rehashing. One doesn't need to
use
...
max-join-size                     1000000
secure-auth                       TRUE
show-warnings                     FALSE
plugin-dir                        (No default value)
default-auth                      (No default value)
histignore                        (No default value)
binary-mode                       FALSE
server-public-key-path            (No default value)
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

3. 使用用户名和口令选项启动 mysql 客户机。退出客户机。在终端窗口中输入以下内容，得到的结果如下所示：

```
$ mysql -uroot -poracle
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 2
Server version: 5.6.10-enterprise-commercial-advanced-log MySQL
Enterprise Server - Advanced Edition (Commercial)

Copyright (c) 2000, 2013, Oracle and/or its affiliates. All rights
reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input
statement.

mysql> EXIT
Bye
```

   – 请注意，您还可以对用户和口令选项使用以下语法：
   --user=root --password=oracle。另请注意，最佳做法是避免在命令行中使用口令，而应尽可能使用其他方法（例如 --login-path）。

4. 使用 admin 登录路径启动 mysql 客户机，并在执行命令行时包含 SELECT 语句（以 HTML 格式显示当前日期和时间）。在终端窗口中输入以下内容，得到的结果如下所示：

```
$ mysql --login-path=admin --html -e \
    "SELECT CURRENT_DATE(), CURRENT_TIME()"
<TABLE BORDER=1><TR><TH>CURRENT_DATE()</TH><TH>CURRENT_TIME()</TH>
</TR><TR><TD>2013-02-03</TD><TD>09:56:34</TD></TR></TABLE>[prompt]$
```

   – 这将以 HTML 格式（而不是默认的表格格式）返回日期和时间戳查询的输出。还可以通过向该命令添加以下内容来将此输出重定向到文件中：
   > <path>/date_time.html

5. 使用用户名、端口（设置为 3306）以及用于此会话的 tee 文件启动 **mysql** 客户机。允许客户机请求口令。出现请求时输入客户机口令。退出客户机并确认已创建 tee 文件。

   a. 在终端窗口中输入以下内容，得到的结果如下所示：

```
$ mysql -uroot -p -P 3306 --tee=tee_1.txt
Logging to file 'tee_1.txt'
Enter password: oracle
Welcome to the MySQL monitor.  Commands end with ; or \g.
...
mysql> EXIT
Bye
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

b. 通过在终端窗口中输入以下内容来查看 tee 文件以确保其已正确创建，得到的结果如下所示：

```
$ cat tee_1.txt
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 4
Server version: 5.6.10-enterprise-commercial-advanced-log MySQL
Enterprise Server - Advanced Edition (Commercial)

Copyright (c) 2000, 2013, Oracle and/or its affiliates. All rights
reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input
statement.

mysql> EXIT
```

6. 以安全更新模式启动 mysql 客户机。在终端窗口中输入以下内容，得到的结果如下所示：

```
$ mysql --login-path=admin --safe-updates
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 5
...
```

7. 在 mysql 客户机中，显示会话状态。在终端窗口中输入以下内容，得到的结果如下所示：

```
mysql> STATUS;
--------------
mysql  Ver 14.14 Distrib 5.6.10, for Linux (x86_64) using  EditLine
wrapper

Connection id:          5
Current database:
Current user:           root@localhost
SSL:                    Not in use
Current pager:          stdout
Using outfile:          ''
Using delimiter:        ;
Server version:         5.6.10-enterprise-commercial-advanced-log
MySQL Enterprise Server - Advanced Edition (Commercial)
Protocol version:       10
Connection:             Localhost via UNIX socket
Server characterset:    latin1
Db     characterset:    latin1
Client characterset:    utf8
Conn.  characterset:    utf8
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

```
UNIX socket:            /var/lib/mysql/mysql.sock
Uptime:                 2 hours 28 min 59 sec

Threads: 1  Questions: 18  Slow queries: 0  Opens: 71  Flush tables: 1
Open tables: 64  Queries per second avg: 0.000

Note that you are running in safe_update_mode:
UPDATEs and DELETEs that don't use a key in the WHERE clause are not
allowed.
(One can force an UPDATE/DELETE by adding LIMIT # at the end of the
command.)
SELECT has an automatic 'LIMIT 1000' if LIMIT is not used.
Max number of examined row combination in a join is set to: 1000000


--------------
```

8. 列出服务器端帮助：

   a. 找到整体类别。在终端窗口中输入以下内容，得到的结果如下所示：

```
mysql> HELP CONTENTS;
You asked for help about help category: "Contents"
For more information, type 'help <item>', where <item> is one of the
following categories:
   Account Management
   Administration
   Compound Statements
   Data Definition
   Data Manipulation
   Data Types
   Functions
   Functions and Modifiers for Use with GROUP BY
   Geographic Features
   Help Metadata
   Language Structure
   Plugins
   Procedures
   Storage Engines
   Table Maintenance
   Transactions
   User-Defined Functions
   Utility
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

b. 列出与帐户管理相关的命令。在终端窗口中输入以下内容，得到的结果如下所示：

```
mysql> HELP Account Management;
You asked for help about help category: "Account Management"
For more information, type 'help <item>', where <item> is one of the
following topics:
   ALTER USER
   CREATE USER
   DROP USER
   GRANT
   RENAME USER
   REVOKE
   SET PASSWORD
```

c. 显示与设置口令相关的信息。在终端窗口中输入以下内容，得到的结果如下所示：

```
mysql> HELP SET PASSWORD;
Name: 'SET PASSWORD'
Description:
Syntax:
SET PASSWORD [FOR user] =
    {
        PASSWORD('cleartext password')
      | OLD_PASSWORD('cleartext password')
      | 'encrypted password'
    }

The SET PASSWORD statement assigns a password to an existing MySQL
user account. When the read_only system variable is enabled, the SUPER
...
```

9. 显示可用数据库。退出客户机。在终端窗口中输入以下内容，得到的结果如下所示：

```
mysql> SHOW DATABASES;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| mysql              |
| performance_schema |
| world2             |
| world_innodb       |
+--------------------+
5 rows in set (0.00 sec)
mysql> EXIT
Bye
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

练习 5-2：调用 mysqladmin 客户机

概览

在本练习中，您将使用某些常见选项调用 mysqladmin 客户机。

持续时间

完成本练习大约需要 5 分钟。

任务

1. 显示本地主机上运行的 mysqladmin 版本。
2. 显示 mysqladmin 客户机选项。
3. 使用用户名和口令启动 mysqladmin 客户机并列出当前变量设置。

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

练习解答 5-2：调用 mysqladmin 客户机

任务

1. 显示本地主机上运行的 mysqladmin 版本。在终端窗口中输入以下内容，得到的结果如下所示：

```
$ mysqladmin -V
mysqladmin  Ver 8.42 Distrib 5.6.10, for Linux on x86_64
```

2. 显示 mysqladmin 客户机选项。在终端窗口中输入以下内容，得到的结果如下所示：

```
$ mysqladmin --help
mysqladmin  Ver 8.42 Distrib 5.6.10, for Linux on x86_64
Copyright (c) 2000, 2013, Oracle and/or its affiliates. All rights
reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Administration program for the mysqld daemon.
Usage: mysqladmin [OPTIONS] command command....
  --bind-address=name IP address to bind to.
  -c, --count=#       Number of iterations to make. This works with -i
                      (--sleep) only.
  --debug-check       Check memory and open file usage at exit.
  --debug-info        Print some debug info at exit.
...
  status                Gives a short status message from the server
  start-slave           Start slave
  stop-slave            Stop slave
  variables             Prints variables available
  version               Get version info from server
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

3. 使用用户名和口令启动 mysqladmin 客户机并列出当前变量设置。在终端窗口中输入以下内容，得到的结果如下所示：

```
$ mysqladmin -uroot -poracle variables
+------------------------------+-----------------+
| Variable_name                | Value           |
+------------------------------+-----------------+
| audit_log_buffer_size        | 1048576         |
| audit_log_file               | audit.log       |
| audit_log_flush              | OFF             |
| audit_log_policy             | ALL             |
| audit_log_rotate_on_size     | 0               |
| audit_log_strategy           | ASYNCHRONOUS    |
| auto_increment_increment     | 1               |
| auto_increment_offset        | 1               |
| autocommit                   | ON              |
| automatic_sp_privileges      | ON              |
| back_log                     | 80              |
| basedir                      | /usr            |
...
| version_compile_machine      | x86_64          |
| version_compile_os           | Linux           |
| wait_timeout                 | 28800           |
+------------------------------+-----------------+
```

– 还可以通过向该命令添加以下内容来将此输出重定向到文件中：

```
> <path>/mysqladmin_variables.txt
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

练习 5-3：观看 MySQL Enterprise Monitor 演示

概览

在本练习中，您将运行基于 Web 的 Enterprise Monitor 演示，该演示可显示其主要功能。

持续时间

完成本练习大约需要 10 分钟。

注：本练习没有单独的解答。

任务

1. 打开 Mozilla Firefox Web 浏览器。
2. 浏览到“MySQL Enterprise Monitor: Dashboard（MySQL Enterprise Monitor：系统信息显示板）”书签，然后运行演示。
   注：该演示将从以下位置运行：
   file:///labs/demos/memdashboard/memdashboard.html
3. 浏览到“MySQL Enterprise Monitor: Query Analyzer（MySQL Enterprise Monitor：查询分析器）”书签，然后运行演示。
   注：该演示将从以下位置运行：
   file:///labs/demos/memqueryanalyzer/memqueryanalyzer.html

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

练习 5-4：通过 MySQL Workbench 执行系统管理任务

概览

在本练习中，您将安装 MySQL Workbench，用其创建和填充数据库，并使用其系统管理功能。

持续时间

完成本练习大约需要 25 分钟。

任务

1. 通过 `/stage/mysql` 中的相应 RPM 文件安装 MySQL Workbench。
2. 通过选择“Applications（应用程序）”>“Programming（编程）”>“MySQL Workbench”，启动 MySQL Workbench。
3. 在“Home（主页）”屏幕中，创建到 `world_innodb` 模式的新连接，并将其命名为 `World`。
4. 使用 MySQL Workbench 启动“Query Database（查询数据库）”窗口。
5. 通过 MySQL Workbench 查询 `world_innodb.Country` 表。
6. 通过 MySQL Workbench 查询 `world_innodb.City` 表。
7. 同时查询上述两个表。
8. 使用“Plug-ins（插件）”菜单将查询中的关键字转换为大写。
9. 查看包含 `world_innodb.CountryLanguage` 表中的列的列表。
10. 通过 `/labs` 目录中的脚本装入 `sakila` 数据库，然后刷新“Schemas（模式）”列表以确认其已正确装入。
11. 为“Server Administration（服务器管理）”模块创建新的服务器实例。
12. 连接到新的服务器实例。
13. 在“Server Status（服务器状态）”页面上，查看显示系统负载、所使用的内存、所使用的连接、通信流量、查询高速缓存命中率和关键效率的多个状态图表。
14. 选择“Startup / Shutdown（启动/关闭）”页面并查看“Database Server Status（数据库服务器状态）”。
15. 选择“Status and System Variables（状态和系统变量）”页面。
16. 在“Status Variables（状态变量）”选项卡上，选择“InnoDB/Stats”。请注意显示的 InnoDB 状态变量。
17. 选择“Options File（选项文件）”页面。
18. 查看配置文件位置，注意该位置为 `/etc/my.cnf`。
19. 浏览“Admin (Local server)（管理（本地服务器））”选项卡中的其他页面和选项卡。
20. 关闭“Admin (Local server)（管理（本地服务器））”选项卡。

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

练习解答 5-4：通过 MySQL Workbench 执行系统管理任务

1. 通过 /stage/mysql 中的相应 RPM 文件安装 MySQL Workbench。

   在终端窗口中执行以下命令，得到的结果如下所示：

```
$ su -
Password: oracle
# cd /stage/mysql
# rpm -hi --nodeps mysql-workbench*.rpm
```

2. 通过选择“Applications（应用程序）”>“Programming（编程）”>“MySQL Workbench”，启动 MySQL Workbench。
3. 在“Home（主页）”屏幕中，创建到 world_innodb 模式的新连接，并将其命名为 World。
   a. 在“SQL Development（SQL 开发）”下，单击“New Connection（新建连接）”。
   b. 在“Setup New Connection（设置新连接）”对话框中，请记下默认的“Connection Method（连接方法）”为“Standard (TCP/IP)（标准 (TCP/IP)）”，默认的“Hostname（主机名）”为 127.0.0.1，默认的“Port（端口）”为 3306，默认的“Username（用户名）”为 root。
   c. 对于“Connection Name（连接名称）”，请输入 World。
   d. 对于“Default Schema（默认模式）”，请输入 world_innodb。
   e. 选择“Advanced（高级）”选项卡。
   f. 请记下默认设置，但不要做任何更改。
   g. 单击“Test Connection（测试连接）”。
   h. 输入口令 oracle 并单击“OK（确定）”。
   i. 在对话框中单击“OK（确定）”。
   j. 单击“OK（确定）”保存新连接。
4. 使用 MySQL Workbench 启动“Query Database（查询数据库）”窗口。
   a. 右键单击新创建的 World 连接，然后选择“Query Database（查询数据库）”。
   b. 输入口令 oracle 并选择“Save password in keychain（将口令保存到 keychain）”选项。
   c. 单击“OK（确定）”。
   d. 如果这是第一次在该系统上使用 keychain，则会显示另一个对话框。在“Password（口令）”字段和“Confirm password（确认口令）”字段中分别输入口令 oracle，然后单击“Create（创建）”。
   e. 注意以下现象：
      – MySQL Workbench 中将显示新的“SQL Editor (World)（SQL 编辑器 (World)）”选项卡。
      – “SCHEMAS（模式）”窗格中包含 MySQL 服务器中用户数据库的列表，该列表当前包含数据库 world2 和 world_innodb。
      – 将显示一个空的“Query 1（查询 1）”窗格，您可在其中输入某些 SQL 语句。

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

5. 通过 MySQL Workbench 查询 `world_innodb.Country` 表。
   a. 在“Query 1（查询 1）”窗格中输入以下语句：

```
select * from Country;
```

   b. 按 Ctrl + Return 可执行该语句。
6. 通过 MySQL Workbench 查询 `world_innodb.City` 表。
   a. 在“Query 1（查询 1）”窗格中上述语句下面的新行中输入以下语句：

```
select * from City;
```

   b. 按 Ctrl + Return 可执行该语句。请确保在上一步中，光标在查询的结尾处闪烁。
7. 同时查询上述两个表。
   a. 按 Ctrl + Shift + Return 可执行编辑器窗口中的所有语句。

   注：结果窗格中会显示两个选项卡，分别返回查询编辑器窗口中的每个语句所对应的结果集。
8. 使用“Plug-ins（插件）”菜单将查询中的关键字转换为大写。
   a. 在该菜单中，选择“Plugins（插件）”>“Utilities（实用程序）”>“Make keywords in query uppercase（将查询中的关键字转换为大写）”。
9. 查看包含 `world_innodb.CountryLanguage` 表中的列的列表。
   a. 在“SCHEMAS（模式）”窗格中，浏览到“world_innodb”>“Tables（表）”>“CountryLanguage”>“Columns container（列容器）”，然后浏览包含 `CountryLanguage` 表中的列的列表。单击每个列时，请注意“Object Info（对象信息）”窗格中该列的定义。
10. 通过 `/labs` 目录中的脚本装入 `sakila` 数据库，然后刷新“Schemas（模式）”列表以确认其已正确装入。
    a. 选择“File（文件）”>“Open SQL Script（打开 SQL 脚本）”。
    b. 浏览到 `/labs/sakila-db/sakila-schema.sql`。
    c. 按 Ctrl + Shift + Return 可执行整个脚本。
    d. 选择“File（文件）”>“Open SQL Script（打开 SQL 脚本）”。
    e. 浏览到 `/labs/sakila-db/sakila-data.sql`。
    f. 在“Unknown File Encoding（未知文件编码）”对话框中，单击“OK（确定）”。
    g. 按 Ctrl + Shift + Return 可执行整个脚本。
    h. 在“Schemas（模式）”窗格中，单击“Refresh（刷新）”图标。请注意，此时将显示 `sakila` 数据库。
11. 为“Server Administration（服务器管理）”模块创建新的服务器实例。
    a. 单击“Home（主页）”选项卡返回主屏幕。在“Server Administration（服务器管理）”下，单击“New Server Instance（新建服务器实例）”。
    b. 此时将显示“Create New Server Instance Profile（创建新的服务器实例配置文件）”对话框，显示与“MySQL 客户机”课程中所介绍的页面类似的一系列页面中的第一个页面。
    c. 第一个此类页面的标题为“Specify the Host Machine the Database Server is running on（指定正在运行数据库服务器的主机）”。单击“Next（下一步）”接受默认值 localhost。

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

d. 在“Set the Database Connection Values（设置数据库连接值）”页面上，接受默认值并单击“Next（下一步）”。要测试连接详细信息，必须输入相应的口令。

e. 在“Please enter password for the following service:（请输入以下服务的口令：）”对话框中，输入口令 `oracle`。

f. 在“Testing the Database Connection（测试数据库连接）”页面上，注意到“Database connection tested successfully（已成功测试数据库连接）”消息后单击“Next（下一步）”。

g. 在“Specify the installation type for you target operation system（为您的目标操作系统指定安装类型）”页面上，选择以下设置：

   – 操作系统：Linux（已选定）
   – MySQL 安装类型：通用 Linux（MySQL tar 软件包）

h. 单击“Next（下一步）”。

i. 安装过程将检查配置设置，并转向标题为“Testing Host Machine Settings（测试主机设置）”的屏幕。注意到“Testing host machine settings is done（已完成对主机设置的测试）”消息后单击“Next（下一步）”。

j. 在“Review settings（查看设置）”对话框中，单击“Continue（继续）”。

k. 在“Create the Instance Profile（创建实例配置文件）”页面上，为服务实例名称输入 `Local server`，然后单击“Finish（完成）”。

12. 连接到新的服务器实例。

    a. 在主屏幕上，注意“Server Administration（服务器管理）”下出现了新的“Local server（本地服务器）”项。双击“Local server（本地服务器）”项。

    b. 如果出现提示，则输入口令 **`oracle`**。

    c. 此时 Workbench 中将显示一个标题为“Admin (Local server)（管理（本地服务器））”的新选项卡，其中左侧显示多个页面，分为“Management（管理）”、“Configuration（配置）”、“Security（安全）”和“Data Export/Restore（数据导出/恢复）”几个部分。

13. 在“Server Status（服务器状态）”页面上，查看显示系统负载、所使用的内存、所使用的连接、通信流量、查询高速缓存命中率和关键效率的多个状态图表。

14. 选择“Startup / Shutdown（启动/关闭）”页面并查看“Database Server Status（数据库服务器状态）”。

15. 选择“Status and System Variables（状态和系统变量）”页面。

16. 在“Status Variables（状态变量）”选项卡中，选择“InnoDB/Stats”。请注意显示的 InnoDB 状态变量。

17. 选择“Options File（选项文件）”页面。

18. 查看配置文件位置，注意该位置为 `/etc/my.cnf`。

19. 浏览“Admin (Local server)（管理（本地服务器））”选项卡中的其他页面和选项卡。

20. 关闭“Admin (Local server)（管理（本地服务器））”选项卡。

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

第 6 课的练习：数据类型

第 6 章

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

第 6 课的练习：概览

练习概览

通过这些练习，可测试您对 MySQL 数据类型的掌握程度。上机练习是针对 Oracle Linux 操作系统环境编写的，Oracle 课堂中将提供此环境。对于非 Oracle 课堂环境，可能需要对文件位置进行一些调整。

假设

- 在本练习之前已经完成了 MySQL 服务器安装。
- 您可以通过命令行提示符访问 `mysql` 客户机。

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

练习 6-1：测验 – MySQL 数据类型

概览

在本练习中，您将回答有关 MySQL 数据类型的问题。

持续时间

完成本练习大约需要 5 分钟。

测验问题

请从所提供的每道多选题或正/误判断题中选择最佳答案。

1. 四种主要数据类型类别包括以下哪些项？
    a. 时间
    b. 数值
    c. 二进制
    d. 字符
    e. 以上都是
2. 判断题：创建数据库时，作为 `CREATE DATABASE` 语句的一部分，会进行数据类型声明。
    a. 正确
    b. 错误
3. 判断题：MySQL 中的数据类型不支持小数。
    a. 正确
    b. 错误
4. `BLOB` 数据类型属于 _____ 类别。
    a. 字符
    b. 数值
    c. 时间
    d. 二进制
    e. 以上都是
5. 判断题：MySQL 中的数据类型支持 `NULL` 列值。
    a. 正确
    b. 错误
6. 以下属性适用于所有数据类型：
    a. `NULL`、`NOT NULL` 和 `DEFAULT`
    b. `INTEGER`
    c. 列
    d. 数值、字符串和常规

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

练习解答 6-1：测验 — MySQL 数据类型

测验解答

1. **e.** 以上都是。
2. **b. 错误。**数据类型在 **`CREATE TABLE`** 语句中初次指定是作为列声明的一部分。
3. **b. 错误。**浮点值（包含整数部分、小数部分或同时包括二者的近似值数值）是受支持的数值类别数据类型之一。
4. **d.** 二进制。
5. **a. 正确**。
6. **a.** **`NULL`**、**`NOT NULL`** 和 **`DEFAULT`**。

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

练习 6-2：设置数据类型

概览

在本练习中，您将设置和查看表列的数据类型。

假设

- 您已登录到 mysql 客户机。

持续时间

完成本练习大约需要 10 分钟。

任务

1. 创建新数据库 test，以便创建表和设置数据类型。
2. 在新的 test 数据库中创建一个名为 integers 的表，其中包含一个名为 n（较小的无符号整数）的列。然后向表列中添加数据值 5。
3. 使用 **SELECT** 语句确认 integers 表存在并包含指定的值。
4. 显示 integers 表的表状态并记录数据类型值的数据长度。
5. 在 world_innodb 数据库中，使用以下语句创建 City 表的副本：

```
CREATE TABLE CityCopy LIKE City;
INSERT INTO CityCopy SELECT * FROM City;
```

6. 显示 CityCopy 表的结构（使用 DESC 语句）以确认当前的数据类型设置，并显示当前的表状态。记录行平均长度和数据平均长度。
7. 将 CityCopy 表的 Population 列的数据类型更改为 BIGINT（原始设置为 INT）。然后再次检查表结构和状态，记录修改后的行平均长度和数据平均长度。
8. 删除 CityCopy 表。

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

练习解答 6-2：设置数据类型

任务

1. 创建新数据库 test，以便创建表和设置数据类型。在终端窗口中输入以下内容，得到的结果如下所示：

```
mysql> CREATE DATABASE test;
Query OK, 1 row affected (0.01 sec)
```

2. 在新的 test 数据库中创建一个名为 integers 的表，其中包含一个名为 n（较小的无符号整数）的列。然后向表列中添加数据值 5。在终端窗口中输入以下内容，得到的结果如下所示：

```
mysql> USE test
Database changed
mysql> CREATE TABLE integers(n SMALLINT UNSIGNED);
Query OK, 0 rows affected (0.97 sec)

mysql> INSERT INTO integers VALUES (5);
Query OK, 1 rows affected (0.03 sec)
```

3. 使用 SELECT 语句确认 integers 表存在并包含指定的值。在终端窗口中输入以下内容，得到的结果如下所示：

```
mysql> SELECT * FROM integers;
+------+
| n    |
+------+
|    5 |
+------+
1 row in set (0.01 sec)
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

4. 显示 integers 表的表状态并记录数据类型值的数据长度。在终端窗口中输入以下内容，得到的结果如下所示：

```
mysql> SHOW TABLE STATUS LIKE 'integers'\G
*************************** 1. row ***************************
           Name: integers
         Engine: InnoDB
        Version: 10
     Row_format: Compact
           Rows: 1
 Avg_row_length: 16384
    Data_length: 16384
Max_data_length: 0
   Index_length: 0
      Data_free: 0
 Auto_increment: NULL
    Create_time: 2012-10-04 19:53:31
    Update_time: NULL
     Check_time: NULL
      Collation: latin1_swedish_ci
       Checksum: NULL
 Create_options:
        Comment:
1 row in set (0.00 sec)
```

5. 在 world_innodb 数据库中，使用以下语句创建 City 表的副本。在终端窗口中输入以下内容，得到的结果如下所示：

```
mysql> USE world_innodb;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql> CREATE TABLE CityCopy LIKE City;
Query OK, 0 rows affected (0.21 sec)

mysql> INSERT INTO CityCopy SELECT * FROM City;
Query OK, 4079 rows affected, 1 warning (0.44 sec)
Records: 4079  Duplicates: 0  Warnings: 1
```

MySQL 会生成一个警告，因为基于语句的日志记录没有以安全方式复制自动增量列。在本课中，您可以忽略此警告。

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

6. 显示 CityCopy 表的结构（使用 DESC 语句）以确认当前的数据类型设置，并显示当前的表状态。记录行平均长度和数据平均长度。

   a. 在终端窗口中输入以下内容，得到的结果如下所示：

```
mysql> DESC CityCopy;
+-------------+----------+------+-----+---------+----------------+
| Field       | Type     | Null | Key | Default | Extra          |
+-------------+----------+------+-----+---------+----------------+
| ID          | int(11)  | NO   | PRI | NULL    | auto_increment |
| Name        | char(35) | NO   |     |         |                |
| CountryCode | char(3)  | NO   | MUL |         |                |
| District    | char(20) | NO   |     |         |                |
| Population  | int(11)  | NO   |     | 0       |                |
+-------------+----------+------+-----+---------+----------------+
5 rows in set (0.07 sec)

mysql> SHOW TABLE STATUS LIKE 'CityCopy'\G
*************************** 1. row ***************************
           Name: CityCopy
         Engine: InnoDB
        Version: 10
     Row_format: Compact
           Rows: 4188
 Avg_row_length: 97
    Data_length: 409600
Max_data_length: 0
   Index_length: 131072
      Data_free: 0
 Auto_increment: 4080
    Create_time: 2012-10-04 20:05:13
    Update_time: NULL
     Check_time: NULL
      Collation: latin1_swedish_ci
       Checksum: NULL
 Create_options:
        Comment:
1 row in set (0.00 sec)
```

7. 将 CityCopy 表的 Population 列的数据类型更改为 BIGINT（原始设置为 INT）。然后再次检查表结构和状态，记录修改后的行平均长度和数据平均长度。

   a. 在终端窗口中输入以下内容，得到的结果如下所示：

```
mysql> ALTER TABLE CityCopy MODIFY Population BIGINT;
Query OK, 4079 rows affected (0.02 sec)
Records: 4079  Duplicates: 0  Warnings: 0
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

b. 然后再次检查表状态。在终端窗口中输入以下内容，得到的结果如下所示：

```
mysql> DESC CityCopy;
+-------------+-----------+------+-----+---------+----------------+
| Field       | Type      | Null | Key | Default | Extra          |
+-------------+-----------+------+-----+---------+----------------+
| ID          | int(11)   | NO   | PRI | NULL    | auto_increment |
| Name        | char(35)  | NO   |     |         |                |
| CountryCode | char(3)   | NO   | MUL |         |                |
| District    | char(20)  | NO   |     |         |                |
| Population  | bigint(20)| YES  |     | 0       |                |
+-------------+-----------+------+-----+---------+----------------+

mysql> SHOW TABLE STATUS LIKE 'CityCopy'\G
*************************** 1. row ***************************
           Name: CityCopy
         Engine: InnoDB
        Version: 10
     Row_format: Compact
           Rows: 4118
 Avg_row_length: 103
    Data_length: 425984
Max_data_length: 0
   Index_length: 131072
      Data_free: 0
 Auto_increment: 4080
    Create_time: 2012-10-04 20:13:28
    Update_time: NULL
     Check_time: NULL
      Collation: latin1_swedish_ci
       Checksum: NULL
 Create_options:
        Comment:
1 row in set (0.00 sec)
```

由于 Population 列的大小增加，因此行平均长度和数据平均长度也有所增加。

8. 删除 CityCopy 表。在终端窗口中输入以下内容，得到的结果如下所示：

```
mysql> DROP TABLE CityCopy;
Query OK, 0 rows affected (0.04 sec)
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

第 7 课的练习：获取元数据

第 7 章

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

第 7 课的练习：概览

练习概览

通过这些练习，可测试您对“获取 MySQL 数据库元数据”方面知识的掌握程度。上机练习是针对 Oracle Linux 操作系统环境编写的，Oracle 课堂中将提供此环境。对于非 Oracle 课堂环境，可能需要对文件位置进行一些调整。

假设

- 在进行这些练习之前已经完成了 MySQL 服务器安装。
- 在终端窗口中以 `oracle` 用户身份登录。
- 您可以通过命令行提示符访问 `mysql` 客户机。

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

练习 7-1：使用 INFORMATION_SCHEMA 获取元数据

概览

在本练习中，您将在 INFORMATION_SCHEMA 数据库中查询元数据。

持续时间

完成本练习大约需要 15 分钟。

任务

1. 通过针对 INFORMATION_SCHEMA 数据库中的 SCHEMATA 表使用 SELECT 语句，获取有关 world_innodb 数据库（模式）的元数据。
2. 通过针对 INFORMATION_SCHEMA 数据库中的 TABLE_SCHEMA 表使用 SELECT 语句，选择 world_innodb 数据库（模式）中表的名称和引擎。
3. 通过针对 INFORMATION_SCHEMA 数据库中的 TABLE_SCHEMA 表使用包含 GROUP BY 子句的 SELECT 语句，列出每个数据库（模式）中每个存储引擎的表数。
4. 通过针对 INFORMATION_SCHEMA 数据库中的 TABLES 表使用 SELECT 语句，列出 world_innodb 数据库中 City 表的数据长度。
5. 通过针对 INFORMATION_SCHEMA 数据库中的 COLUMNS 表使用包含 COUNT(*) 选项的 SELECT 语句，列出 world_innodb 数据库中使用 CHAR 或 VARCHAR 数据类型的表列数。

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

练习解答 7-1：使用 INFORMATION_SCHEMA 获取元数据

任务

1. 通过针对 INFORMATION_SCHEMA 数据库中的 SCHEMATA 表使用 SELECT 语句，获取有关 world_innodb 数据库（模式）的元数据。在终端窗口中输入以下内容，得到的结果如下所示：

```
$ mysql -uroot -poracle
...

mysql> SELECT *
    -> FROM INFORMATION_SCHEMA.SCHEMATA
    -> WHERE SCHEMA_NAME = 'world_innodb'\G
*************************** 1. row ***************************
              CATALOG_NAME: def
               SCHEMA_NAME: world_innodb
DEFAULT_CHARACTER_SET_NAME: latin1
    DEFAULT_COLLATION_NAME: latin1_swedish_ci
                  SQL_PATH: NULL
1 row in set (0.00 sec)
```

2. 通过针对 INFORMATION_SCHEMA 数据库中的 TABLE_SCHEMA 表使用 SELECT 语句，选择 world_innodb 数据库（模式）中表的名称和引擎。在终端窗口中输入以下内容，得到的结果如下所示：

```
mysql> USE INFORMATION_SCHEMA
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql> SELECT TABLE_NAME, ENGINE
    -> FROM TABLES
    -> WHERE TABLE_SCHEMA = 'world_innodb';
+-----------------+--------+
| TABLE_NAME      | ENGINE |
+-----------------+--------+
| City            | InnoDB |
| Country         | InnoDB |
| CountryLanguage | InnoDB |
+-----------------+--------+
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

3. 通过针对 INFORMATION_SCHEMA 数据库中的 TABLE_SCHEMA 表使用包含 GROUP BY 子句的 SELECT 语句，列出每个数据库（模式）中每个存储引擎的表数。在终端窗口中输入以下内容，得到的结果如下所示：

```
mysql> SELECT TABLE_SCHEMA, ENGINE, COUNT(*)
    -> FROM TABLES
    -> GROUP BY TABLE_SCHEMA, ENGINE;
+--------------------+--------------------+----------+
| TABLE_SCHEMA       | ENGINE             | COUNT(*) |
+--------------------+--------------------+----------+
| information_schema | MEMORY             |       50 |
| information_schema | MyISAM             |       10 |
| mysql              | CSV                |        2 |
| mysql              | InnoDB             |        5 |
| mysql              | MyISAM             |       21 |
| performance_schema | PERFORMANCE_SCHEMA |       52 |
| sakila             | NULL               |        7 |
| sakila             | InnoDB             |       15 |
| sakila             | MyISAM             |        1 |
| test               | InnoDB             |        1 |
| world2             | InnoDB             |        3 |
| world_innodb       | InnoDB             |        3 |
+--------------------+--------------------+----------+
12 rows in set (0.01 sec)
```

4. 通过针对 INFORMATION_SCHEMA 数据库中的 TABLES 表使用 SELECT 语句，列出 world_innodb 数据库中 City 表的数据长度。在终端窗口中输入以下内容，得到的结果如下所示：

```
mysql> SELECT TABLE_NAME, DATA_LENGTH
    -> FROM TABLES
    -> WHERE TABLE_SCHEMA = 'world_innodb'
    -> AND TABLE_NAME = 'City';
+------------+-------------+
| TABLE_NAME | DATA_LENGTH |
+------------+-------------+
| City       |      409600 |
+------------+-------------+
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

5. 通过针对 INFORMATION_SCHEMA 数据库中的 COLUMNS 表使用包含 COUNT(*) 选项的 SELECT 语句，列出 world_innodb 数据库中使用 CHAR 或 VARCHAR 数据类型的表列数。在终端窗口中输入以下内容，得到的结果如下所示：

```
mysql> SELECT DATA_TYPE, COUNT(*)
    -> FROM COLUMNS
    -> WHERE TABLE_SCHEMA = 'world_innodb'
    -> AND DATA_TYPE IN ('CHAR', 'VARCHAR')
    -> GROUP BY DATA_TYPE;
+-----------+----------+
| DATA_TYPE | COUNT(*) |
+-----------+----------+
| char      |       12 |
+-----------+----------+
```

   – world_innodb 数据库中不存在 VARCHAR 数据类型，因此不会显示在输出中。

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

练习 7-2：使用 SHOW 和 DESCRIBE 获取元数据

概览

在本练习中，您将使用 SHOW 和 DESCRIBE 语句获取数据库元数据。

持续时间

完成本练习大约需要 10 分钟。

任务

1. 列出所有可用的数据库。
2. 列出所有名称中包含字母“o”的数据库。
3. 列出所有可用的 INFORMATION_SCHEMA 数据库表。
4. 显示有关 world_innodb 数据库中 City 表的列的详细信息。
5. 显示 world_innodb 数据库中 City 表的索引信息。
6. 显示 world_innodb 数据库中 CountryLanguage 表的结构。
7. 列出所有可用的字符集。
8. 列出所有可用的整理。
9. 退出 mysql 客户机。

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

练习解答 7-2：使用 SHOW 和 DESCRIBE 获取元数据

任务

1. 列出所有可用的数据库。使用在上一练习中打开的 mysql 客户机在终端窗口中输入以下内容，得到的结果如下所示：

```
mysql> SHOW DATABASES;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| mysql              |
| performance_schema |
| sakila             |
| test               |
| world2             |
| world_innodb       |
+--------------------+
7 rows in set (0.00 sec)
```

2. 列出所有名称中包含字母“o”的数据库。在终端窗口中输入以下内容，得到的结果如下所示：

```
mysql> SHOW DATABASES LIKE '%o%';
+--------------------+
| Database (%o%)     |
+--------------------+
| information_schema |
| performance_schema |
| world2             |
| world_innodb       |
+--------------------+
4 rows in set (0.00 sec)
```

3. 列出所有可用的 INFORMATION_SCHEMA 数据库表。在终端窗口中输入以下内容，得到的结果如下所示：

   a. 将数据库更改为 information_schema：

```
mysql> USE information_schema;
Database changed
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

b. 列出 information_schema 数据库中的表：

```
mysql> SHOW TABLES;
+---------------------------------------+
| Tables_in_information_schema          |
+---------------------------------------+
| CHARACTER_SETS                        |
| COLLATIONS                            |
| COLLATION_CHARACTER_SET_APPLICABILITY |
| COLUMNS                               |
| COLUMN_PRIVILEGES                     |
...
| INNODB_BUFFER_POOL_STATS              |
| INNODB_SYS_COLUMNS                    |
| INNODB_SYS_FOREIGN                    |
+---------------------------------------+
60 rows in set (0.00 sec)
```

4. 显示有关 world_innodb 数据库中 City 表的列的详细信息。在终端窗口中输入以下内容，得到的结果如下所示：

a. 将数据库更改为 world_innodb：

```
mysql> USE world_innodb
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
```

b. 列出 City 表的所有列信息：

```
mysql> SHOW FULL COLUMNS FROM City\G
*************************** 1. row ***************************
     Field: ID
      Type: int(11)
 Collation: NULL
      Null: NO
       Key: PRI
   Default: NULL
     Extra: auto_increment
Privileges: select,insert,update,references
   Comment:
*************************** 2. row ***************************
     Field: Name
      Type: char(35)
 Collation: latin1_swedish_ci
      Null: NO
       Key:
   Default:
     Extra:
Privileges: select,insert,update,references
   Comment:
*************************** 3. row ***************************
     Field: CountryCode
      Type: char(3)
 Collation: latin1_swedish_ci
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

```
      Null: NO
       Key: MUL
   Default:
     Extra:
Privileges: select,insert,update,references
   Comment:
*************************** 4. row ***************************
     Field: District
      Type: char(20)
 Collation: latin1_swedish_ci
      Null: NO
       Key:
   Default:
     Extra:
Privileges: select,insert,update,references
   Comment:
*************************** 5. row ***************************
     Field: Population
      Type: int(11)
 Collation: NULL
      Null: NO
       Key:
   Default: 0
     Extra:
Privileges: select,insert,update,references
   Comment:
5 rows in set (0.00 sec)
```

5. 显示 world_innodb 数据库中 City 表的索引信息。在终端窗口中输入以下内容，得到的结果如下所示：

```
mysql> SHOW INDEX FROM City\G
*************************** 1. row ***************************
        Table: City
   Non_unique: 0
     Key_name: PRIMARY
 Seq_in_index: 1
  Column_name: ID
    Collation: A
  Cardinality: 4188
     Sub_part: NULL
       Packed: NULL
         Null:
   Index_type: BTREE
      Comment:
Index_comment:
*************************** 2. row ***************************
        Table: City
   Non_unique: 1
     Key_name: CountryCode
 Seq_in_index: 1
  Column_name: CountryCode
    Collation: A
  Cardinality: 465
     Sub_part: NULL
       Packed: NULL
         Null:
   Index_type: BTREE
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

```
      Comment:
Index_comment:
2 rows in set (0.01 sec)
```

注：INFORMATION_SCHEMA 表的 TABLE_CONSTRAINTS 和 STATISTICS 也包含索引元数据。

6. 显示 world_innodb 数据库中 CountryLanguage 表的结构。在终端窗口中输入以下内容，得到的结果如下所示：

```
mysql> DESCRIBE CountryLanguage;
+-------------+---------------+------+-----+---------+-------+
| Field       | Type          | Null | Key | Default | Extra |
+-------------+---------------+------+-----+---------+-------+
| CountryCode | char(3)       | NO   | PRI |         |       |
| Language    | char(30)      | NO   | PRI |         |       |
| IsOfficial  | enum('T','F') | NO   |     | F       |       |
| Percentage  | float(4,1)    | NO   |     | 0.0     |       |
+-------------+---------------+------+-----+---------+-------+
4 rows in set (0.03 sec)
```

7. 列出所有可用的字符集。在终端窗口中输入以下内容，得到的结果如下所示：

```
mysql> SHOW CHARACTER SET;
+---------+-----------------------------+---------------------+--------+
| Charset | Description                 | Default collation   | Maxlen |
+---------+-----------------------------+---------------------+--------+
| big5    | Big5 Traditional Chinese    | big5_chinese_ci     |      2 |
| dec8    | DEC West European           | dec8_swedish_ci     |      1 |
| cp850   | DOS West European           | cp850_general_ci    |      1 |
| hp8     | HP West European            | hp8_english_ci      |      1 |
| koi8r   | KOI8-R Relcom Russian       | koi8r_general_ci    |      1 |
...
| eucjpms | UJIS for Windows Japanese | eucjpms_japanese_ci |      3 |
+---------+-----------------------------+---------------------+--------+
40 rows in set (0.02 sec)
```

– 此时将显示当前系统上的完整字符集列表。

8. 列出所有可用的整理。在终端窗口中输入以下内容，得到的结果如下所示：

```
mysql> SHOW COLLATION;
+---------------------+---------+-----+---------+----------+---------+
| Collation           | Charset | Id  | Default | Compiled | Sortlen |
+---------------------+---------+-----+---------+----------+---------+
| big5_chinese_ci     | big5    |   1 | Yes     | Yes      |       1 |
| big5_bin            | big5    |  84 |         | Yes      |       1 |
| dec8_swedish_ci     | dec8    |   3 | Yes     | Yes      |       1 |
| dec8_bin            | dec8    |  69 |         | Yes      |       1 |
| cp850_general_ci    | cp850   |   4 | Yes     | Yes      |       1 |
| cp850_bin           | cp850   |  80 |         | Yes      |       1 |
...
| eucjpms_bin         | eucjpms |  98 |         | Yes      |       1 |
+---------------------+---------+-----+---------+----------+---------+
219 rows in set (0.00 sec)
```

– 此时将显示当前系统上的完整整理列表。

9. 退出 mysql 客户机。在终端窗口中输入以下内容，得到的结果如下所示：

```
mysql> EXIT
Bye
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

练习 7-3：使用 mysqlshow 获取元数据

概览

在本练习中，您将使用 mysqlshow 客户机获取数据库元数据。

持续时间

完成本练习大约需要 5 分钟。

任务

1. 在命令行中使用 mysqlshow 应用程序列出所有可用的数据库。
2. 在命令行中列出 world_innodb 数据库中所有的表。
3. 在命令行中显示 world_innodb 数据库中 CountryLanguage 表的结构。

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

练习解答 7-3：使用 mysqlshow 获取元数据

任务

1. 在命令行中使用 mysqlshow 应用程序列出所有可用的数据库。在终端窗口中输入以下内容，得到的结果如下所示：

```
$ mysqlshow -uroot -poracle
Warning: Using a password on the command line interface can be
insecure.
+--------------------+
|     Databases      |
+--------------------+
| information_schema |
| mysql              |
| performance_schema |
| sakila             |
| test               |
| world2             |
| world_innodb       |
+--------------------+
```

2. 在命令行中列出 world_innodb 数据库中所有的表。在终端窗口中输入以下内容，得到的结果如下所示：

```
$ mysqlshow world_innodb -uroot -poracle
+-----------------+
|     Tables      |
+-----------------+
| City            |
| Country         |
| CountryLanguage |
+-----------------+
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

3. 在命令行中显示 world_innodb 数据库中 CountryLanguage 表的结构。在终端窗口中输入以下内容，得到的结果如下所示：

```
# mysqlshow world_innodb CountryLanguage -uroot -oracle
Database: world_innodb  Table: CountryLanguage
+-------------+--------------+------------------+------+-----+------
---+-------+----------------------------------+---------+
| Field       | Type         | Collation        | Null | Key |
Default | Extra | Privileges                       | Comment |
+-------------+--------------+------------------+------+-----+------
---+-------+----------------------------------+---------+
| CountryCode | char(3)      | latin1_swedish_ci | NO   | PRI |
|       | select,insert,update,references |         |
| Language    | char(30)     | latin1_swedish_ci | NO   | PRI |
|       | select,insert,update,references |         |
| IsOfficial  | enum('T','F') | latin1_swedish_ci | NO   |     | F
|       | select,insert,update,references |         |
| Percentage  | float(4,1)   |                  | NO   |     | 0.0
|       | select,insert,update,references |         |
+-------------+--------------+------------------+------+-----+------
---+-------+----------------------------------+---------+
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

第 8 课的练习：事务与锁定

第 8 章

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

第 8 课的练习：概览

练习概览

通过这些练习，可测试您对 MySQL 中事务和锁定的掌握程度。上机练习是针对 Oracle Linux 操作系统环境编写的，Oracle 课堂中将提供此环境。对于非 Oracle 课堂环境，可能需要对文件位置进行一些调整。

假设

- 在进行这些练习之前已经完成了 MySQL 服务器安装。
- 在终端窗口中以 `oracle` 用户身份登录。
- 您可以通过命令行提示符访问 `mysql` 客户机。

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

练习 8-1：测验 – 事务与锁定

概览

在本练习中，您将回答使用 MySQL 服务器时与事务和锁定有关的问题。

持续时间

完成本练习大约需要 10 分钟。

测验问题

请从所提供的每道多选题或正/误判断题中选择最佳答案。

1. 事务是被视为一个工作单元的数据操作执行步骤集合。
    a. 正确
    b. 错误
2. 使用术语“遵从 ACID”时，该术语指 MySQL 中的事务具有 _____、一致性、_____ 和持久性。
    a. 原子性、隔离性
    b. 自动性、隔离性
    c. 原子性、独立性
    d. 始终打开、针对事务
3. 如果 AUTOCOMMIT 未启用，_____。分别使用 COMMIT 和 ROLLBACK 语句显式提交或回滚事务。事务终止后，将隐式启动新事务。
    a. 事务在某一时间只包含一个语句
    b. 服务器要求您输入 AUTOCOMMIT 设置的值，然后再继续
    c. 默认情况下事务会跨越多个语句
4. 四个事务隔离级别包括 READ UNCOMMITTED、READ COMMITTED、REPEATABLE 和 ISOLATED。
    a. 正确
    b. 错误
5. 两个并发事务同时访问相同的表时会发生三种类型的不一致问题：_____、不可重复读和（或）虚读/虚行。
    a. 可重复读
    b. “脏”读
    c. “污染”读
6. SERIALIZABLE 级别允许不可重复读。
    a. 正确
    b. 错误

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

7. 锁定是避免并发问题的一种机制，由 MySQL 服务器管理，锁定一个客户机是为了 _____。
    a. 与主客户机隔离
    b. 移除任何非 MySQL 客户机
    c. 限制其他客户机
8. InnoDB 支持的两个锁定修饰符为 `LOCK IN SHARE MODE` 和 `FOR UPDATE`。
    a. 正确
    b. 错误
9. 两种锁定类型锁定选择的 _____ 来避免并发访问表数据时会发生的问题。
    a. 锁
    b. 表
    c. 读
    d. 行

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

练习解答 8-1：测验 — 事务与锁定

测验解答

1. **a.** 正确。
2. **a.** ACID = 原子性、一致性、隔离性、持久性。
3. **c.** 默认情况下会跨越多个语句。
4. **b.** 错误。最后两个实际上为 `REPEATABLE READ` 和 `SERIALIZABLE`。
5. **b.** “脏”读。
6. **b.** 错误。`SERIALIZABLE` 将一个事务的结果与其他事务完全隔离。该级别与 `REPEATABLE READ` 类似，但其限制性更强，即一个事务所选的行不能由其他事务更改，直到第一个事务完成为止。
7. **c.** 限制其他客户机。MySQL 使用共享锁或互斥锁实现这一点。
8. **a.** 正确。
9. **d.** 行。

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

练习 8-2：使用事务控制语句

概览

在本练习中，您将使用事务控制语句启动、处理和提交 SQL 事务。

持续时间

完成本练习大约需要 15 分钟。

任务

1. 使用 `SHOW ENGINES` 命令确定系统中是否有任何事务存储引擎可用以及哪个是默认引擎。
2. 使用 `SET AUTOCOMMIT` 语句启用 `AUTOCOMMIT`。
3. 为使用 `world_innodb` 数据库做准备，确认 `City` 表使用事务存储引擎 InnoDB。
4. 使用 `START TRANSACTION` 语句显式启动新事务。
5. 删除一行。
6. 使用 `ROLLBACK` 语句回滚打开的事务。
7. 启动另一个新事务。
8. 再次删除同一行。
9. 使用 `COMMIT` 语句停止并提交事务。
10. 确认该行现已不存在。
11. 尝试回滚已提交的事务。

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

练习解答 8-2：使用事务控制语句

任务

1. 使用 SHOW ENGINES 命令确定系统中是否有任何事务存储引擎可用以及哪个是默认引擎。在终端窗口中输入以下内容，得到的结果如下所示：

```
$ mysql -uroot -poracle
...

mysql> SHOW ENGINES\G
...
*************************** 5. row ***************************
      Engine: MyISAM
     Support: YES
     Comment: MyISAM storage engine
Transactions: NO
          XA: NO
  Savepoints: NO
...
*************************** 8. row ***************************
      Engine: InnoDB
     Support: DEFAULT
     Comment: Supports transactions, row-level locking, and foreign
keys
Transactions: YES
          XA: YES
  Savepoints: YES
...
```

   – 各个系统的列表顺序可能有所不同。无论顺序如何，InnoDB 存储引擎都显示为默认引擎且包含事务。

2. 使用 SET AUTOCOMMIT 语句启用 AUTOCOMMIT。在终端窗口中输入以下内容，得到的结果如下所示：

   a. 将 AUTOCOMMIT 模式设置为开启：

```
mysql> SET AUTOCOMMIT = 1;
Query OK, 0 rows affected (0.06 sec)
```

   b. 检查值进行确认：

```
mysql> SELECT @@AUTOCOMMIT;
+--------------+
| @@AUTOCOMMIT |
+--------------+
|            1 |
+--------------+
1 row in set (0.02 sec)
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

3. 为使用 world_innodb 数据库做准备，确认 City 表使用事务存储引擎 InnoDB。在终端窗口中输入以下内容，得到的结果如下所示：

   a. 更改为 world_innodb 数据库：

```
mysql> USE world_innodb
...
Database changed
```

   b. 显示 City 表结构，并确认 ENGINE 的值是 InnoDB：

```
mysql> SHOW CREATE TABLE City\G
*************************** 1. row ***************************
       Table: City
Create Table: CREATE TABLE `City` (
  `ID` int(11) NOT NULL AUTO_INCREMENT,
  `Name` char(35) NOT NULL DEFAULT '',
  `CountryCode` char(3) NOT NULL DEFAULT '',
  `District` char(20) NOT NULL DEFAULT '',
  `Population` int(11) NOT NULL DEFAULT '0',
  PRIMARY KEY (`ID`),
  KEY `CountryCode` (`CountryCode`),
  CONSTRAINT `city_ibfk_1` FOREIGN KEY (`CountryCode`) REFERENCES
`Country` (`Code`)
) ENGINE=InnoDB AUTO_INCREMENT=4080 DEFAULT CHARSET=latin1
1 row in set (0.00 sec)
```

4. 使用 START TRANSACTION 语句显式启动新事务。在终端窗口中输入以下内容，得到的结果如下所示：

```
mysql> START TRANSACTION;
Query OK, 0 rows affected (0.00 sec)
```

5. 删除一行。在终端窗口中输入以下内容，得到的结果如下所示：

   a. 选择包含“Manta”城市的一行：

```
mysql> SELECT * FROM City WHERE Name = 'Manta';
+-----+-------+-------------+----------+------------+
| ID  | Name  | CountryCode | District | Population |
+-----+-------+-------------+----------+------------+
| 600 | Manta | ECU         | Manabí   |     164739 |
+-----+-------+-------------+----------+------------+
1 row in set (0.06 sec)
```

   b. 选择包含“Manta”城市的一行：

```
mysql> DELETE FROM City WHERE Name = 'Manta';
Query OK, 1 row affected (0.02 sec)
```

   c. 选择包含“Manta”城市的一行：

```
mysql> SELECT * FROM City WHERE Name = 'Manta';
Empty set (0.00 sec)
```

   – 第一个 SELECT 语句确认存在城市 Manta。删除操作返回 OK，表示有一行受到影响。第二个 SELECT 语句显示该行现在是空的。

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

6. 使用 ROLLBACK 语句回滚打开的事务。在终端窗口中输入以下内容，得到的结果如下所示：

   a. 回滚打开的事务：

```
mysql> ROLLBACK;
Query OK, 0 rows affected (0.05 sec)
```

   b. 确认该行现在恢复到其原始状态：

```
mysql> SELECT * FROM City WHERE Name = 'Manta';
+-----+-------+-------------+----------+------------+
| ID  | Name  | CountryCode | District | Population |
+-----+-------+-------------+----------+------------+
| 600 | Manta | ECU         | Manabí   |     164739 |
+-----+-------+-------------+----------+------------+
1 row in set (0.06 sec)
```

   – 结果显示该行现在已恢复。

7. 启动另一个新事务。在终端窗口中输入以下内容，得到的结果如下所示：

```
mysql> START TRANSACTION;
Query OK, 0 rows affected (0.00 sec)
```

8. 再次删除同一行。在终端窗口中输入以下内容，得到的结果如下所示：

```
mysql> DELETE FROM City WHERE Name = 'Manta';
Query OK, 1 row affected (0.02 sec)
```

9. 使用 COMMIT 语句停止并提交事务。在终端窗口中输入以下内容，得到的结果如下所示：

```
mysql> COMMIT;
Query OK, 0 rows affected (0.04 sec)
```

10. 确认该行现已不存在。在终端窗口中输入以下内容，得到的结果如下所示：

```
mysql> SELECT * FROM City WHERE Name = 'Manta';
Empty set (0.00 sec)
```

   – 最后一个 SELECT 语句显示该行不再存在，且该 DELETE 操作无法撤消。

11. 尝试回滚已提交的事务。在终端窗口中输入以下内容，得到的结果如下所示：

   a. 回滚打开的事务：

```
mysql> ROLLBACK;
Query OK, 0 rows affected (0.05 sec)
```

   b. 确认该行的当前状态：

```
mysql> SELECT * FROM City WHERE Name = 'Manta';
Empty set (0.00 sec)
```

   – 结果显示该行仍为空，因为事务在此回滚操作发生前进行了提交。

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

练习 8-3：有关事务和锁定的附加练习

概览

在本练习中，您将使用事务和锁定语句。

持续时间

完成本练习大约需要 30 分钟。

任务

1. 在 mysql 客户机会话中，检查当前隔离级别。
2. 使用 PROMPT 语句，将上一步打开的 mysql 会话中的提示符更改为 t1，以便将其与后面的客户机会话区别开来。
3. 在 t1 mysql 会话中启动一个新事务。
4. 在 t1 会话中，选择 City 表中 ID 大于 4070 的所有行。
5. 在另一个终端窗口中，打开第二个 mysql 会话。将该 mysql 会话中的提示符更改为 t2。
6. 在 t2 mysql 会话中启动一个事务。
7. 在 t2 会话中，从 world_innodb 数据库中选择 City 表中 ID 大于 4070 的所有行。
8. 在 t2 会话中，向 City 表插入新的一行。确认新行已加入。在 t2 终端窗口中输入以下内容：

```
t2> INSERT INTO City (Name, CountryCode) VALUES ('New City', 'ATA');
```

9. 在 t1 会话中，再次选择 ID 大于 4070 的城市。现在是否看到这一行？
10. 在 t2 会话中，提交事务。
11. 在 t1 会话中，提交事务。再次选择 ID 大于 4070 的城市。
12. 在 t1 mysql 会话中启动一个新事务。
13. 在 t1 会话中，删除向 City 表插入的行。确认该行已删除。

更改隔离级别：

14. 在 t2 会话中，使用 SET SESSION tx_isolation 命令将隔离级别更改为 READ UNCOMMITTED，然后确认更改。
15. 在 t2 会话中启动一个新事务。选择 ID 大于 4070 的城市。此次读取出现了什么问题？
16. 在 t1 会话中，使用 ROLLBACK 语句取消事务。再次选择 ID 大于 4070 的城市。
17. 在 t2 会话中，选择 ID 大于 4070 的城市。结果是否与上一步中在 t1 会话中所查询的结果一样？
18. 在 t2 会话中，取消此会话注册的任何事务。
19. 退出两个 mysql 会话以及两个终端窗口，为接下来的练习做好准备。

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

练习解答 8-3：有关事务和锁定的附加练习

任务

1. 在 mysql 客户机会话中，检查当前隔离级别。在终端窗口中输入以下内容，得到的结果如下所示：

```
$ mysql -uroot -poracle
...

mysql> SELECT @@global.tx_isolation;
+-----------------------+
| @@global.tx_isolation |
+-----------------------+
| REPEATABLE-READ       |
+-----------------------+
1 row in set (0.00 sec)
```

2. 将上一步打开的 mysql 会话中的提示符更改为 t1，以便将其与后面的客户机会话区别开来。在终端窗口中输入以下内容，得到的结果如下所示：

```
mysql> PROMPT t1> ;
PROMPT set to 't1> '
t1>
```

3. 在 t1 mysql 会话中启动一个新事务。在 t1 终端窗口中输入以下内容，得到的结果如下所示：

   a. 启动新事务：

```
t1> START TRANSACTION;
Query OK, 0 rows affected (0.00 sec)
```

4. 在 t1 会话中，从 world_innodb 数据库中选择 City 表中 ID 大于 4070 的所有行。在 t1 终端窗口中输入以下内容，得到的结果如下所示：

```
t1> USE world_innodb
...
Database changed
t1> SELECT * FROM City WHERE ID > 4070;
+------+--------------+-------------+------------+------------+
| ID   | Name         | CountryCode | District   | Population |
+------+--------------+-------------+------------+------------+
| 4071 | Mount Darwin | ZWE         | Harare     |     164362 |
| 4072 | Mutare       | ZWE         | Manicaland |     131367 |
| 4073 | Gweru        | ZWE         | Midlands   |     128037 |
| 4074 | Gaza         | PSE         | Gaza       |     353632 |
| 4075 | Khan Yunis   | PSE         | Khan Yunis |     123175 |
| 4076 | Hebron       | PSE         | Hebron     |     119401 |
| 4077 | Jabaliya     | PSE         | North Gaza |     113901 |
| 4078 | Nablus       | PSE         | Nablus     |     100231 |
| 4079 | Rafah        | PSE         | Rafah      |      92020 |
+------+--------------+-------------+------------+------------+
9 rows in set (0.00 sec)
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

5. 在另一个终端窗口中，打开第二个 mysql 会话。将该 mysql 会话中的提示符更改为 t2。在终端窗口中输入以下内容，得到的结果如下所示：

```
$ mysql -uroot -poracle
...
mysql> PROMPT t2> ;
PROMPT set to 't2> '
t2>
```

6. 在 t2 mysql 会话中启动一个事务。在 t2 终端窗口中输入以下内容，得到的结果如下所示：

```
t2> START TRANSACTION;
Query OK, 0 rows affected (0.00 sec)
```

7. 在 t2 会话中，从 world_innodb 数据库中选择 City 表中 ID 大于 4070 的所有行。在 t2 终端窗口中输入以下内容，得到的结果如下所示：

```
t2> USE world_innodb
Database changed

t2> SELECT * FROM City WHERE ID > 4070;
+------+--------------+-------------+------------+------------+
| ID   | Name         | CountryCode | District   | Population |
+------+--------------+-------------+------------+------------+
| 4071 | Mount Darwin | ZWE         | Harare     |     164362 |
| 4072 | Mutare       | ZWE         | Manicaland |     131367 |
| 4073 | Gweru        | ZWE         | MIDlands   |     128037 |
| 4074 | Gaza         | PSE         | Gaza       |     353632 |
| 4075 | Khan Yunis   | PSE         | Khan Yunis |     123175 |
| 4076 | Hebron       | PSE         | Hebron     |     119401 |
| 4077 | Jabaliya     | PSE         | North Gaza |     113901 |
| 4078 | Nablus       | PSE         | Nablus     |     100231 |
| 4079 | Rafah        | PSE         | Rafah      |      92020 |
+------+--------------+-------------+------------+------------+
9 rows in set (0.00 sec)
```

8. 在 t2 会话中，向 City 表插入新的一行。确认新行已加入。
   a. 在 t2 终端窗口中输入以下内容，得到的结果如下所示：

```
t2> INSERT INTO City (Name, CountryCode) VALUES ('New City', 'ATA');
Query OK, 1 row affected (0.00 sec)
```

   – 插入操作已完成，并显示该操作影响了一行。

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

b. 选择 ID 大于 4070 的城市：

```
t2> SELECT * FROM City WHERE ID > 4070;
+------+--------------+-------------+------------+------------+
| ID   | Name         | CountryCode | District   | Population |
+------+--------------+-------------+------------+------------+
| 4071 | Mount Darwin | ZWE         | Harare     |     164362 |
| 4072 | Mutare       | ZWE         | Manicaland |     131367 |
| 4073 | Gweru        | ZWE         | MIDlands   |     128037 |
| 4074 | Gaza         | PSE         | Gaza       |     353632 |
| 4075 | Khan Yunis   | PSE         | Khan Yunis |     123175 |
| 4076 | Hebron       | PSE         | Hebron     |     119401 |
| 4077 | Jabaliya     | PSE         | North Gaza |     113901 |
| 4078 | Nablus       | PSE         | Nablus     |     100231 |
| 4079 | Rafah        | PSE         | Rafah      |      92020 |
| 4080 | New City     | ATA         |            |          0 |
+------+--------------+-------------+------------+------------+
10 rows in set (0.00 sec)
```

9. 在 t1 会话中，再次选择 ID 大于 4070 的城市。

a. 在 t1 终端窗口中输入以下内容，得到的结果如下所示：

```
t1> SELECT * FROM City WHERE ID > 4070;
+------+--------------+-------------+------------+------------+
| ID   | Name         | CountryCode | District   | Population |
+------+--------------+-------------+------------+------------+
...
| 4079 | Rafah        | PSE         | Rafah      |      92020 |
+------+--------------+-------------+------------+------------+
9 rows in set (0.00 sec)
```

b. 现在是否看到这一行？该行仍然不在。这是因为隔离级别是 REPEATABLE READ。

10. 在 t2 会话中，提交事务。在 t2 终端窗口中输入以下内容，得到的结果如下所示：

```
t2> COMMIT;
Query OK, 0 rows affected (0.01 sec)
```

11. 在 t1 会话中，提交事务。再次选择 ID 大于 4070 的城市。在 t1 终端窗口中输入以下内容，得到的结果如下所示：

```
t1> COMMIT;
Query OK, 0 rows affected (0.01 sec)

t1> SELECT * FROM City WHERE ID > 4070;
+------+--------------+-------------+------------+------------+
| ID   | Name         | CountryCode | District   | Population |
+------+--------------+-------------+------------+------------+
...
| 4080 | New City     | ATA         |            |          0 |
+------+--------------+-------------+------------+------------+
10 rows in set (0.00 sec)
```

– 请注意，在 t2 会话中插入的行现在对 t1 会话可见，这是因为在每个会话中提交了事务。

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

12. 在 t1 mysql 会话中启动一个新事务。在 t1 终端窗口中输入以下内容，得到的结果如下所示：

```
t1> START TRANSACTION;
Query OK, 0 rows affected (0.00 sec)
```

13. 在 t1 会话中，删除向 City 表插入的行。确认该行已删除。在 t1 终端窗口中输入以下内容，得到的结果如下所示：

```
t1> DELETE FROM City WHERE ID = 4080;
Query OK, 1 row affected (0.00 sec)

t1> SELECT * FROM City WHERE ID > 4070;
+------+--------------+-------------+------------+------------+
| ID   | Name         | CountryCode | District   | Population |
+------+--------------+-------------+------------+------------+
...
| 4079 | Rafah        | PSE         | Rafah      |      92020 |
+------+--------------+-------------+------------+------------+
9 rows in set (0.00 sec)
```

- 该行已被删除。

更改隔离级别：

14. 在 t2 会话中，使用 SET SESSION tx_isolation 命令将隔离级别更改为 READ UNCOMMITTED，然后确认更改。在 t1 终端窗口中输入以下内容，得到的结果如下所示：

```
t2> SET SESSION tx_isolation = 'READ-UNCOMMITTED';
Query OK, 0 rows affected (0.00 sec)

t2> SELECT @@tx_isolation;
+-----------------------+
| @@global.tx_isolation |
+-----------------------+
| READ-UNCOMMITTED      |
+-----------------------+
1 row in set (0.00 sec)
```

15. 在 t2 会话中启动一个新事务。选择 ID 大于 4070 的城市。在 t2 终端窗口中输入以下内容，得到的结果如下所示：

```
t2> START TRANSACTION;
Query OK, 0 rows affected (0.00 sec)

t2> SELECT * FROM City WHERE ID > 4070;
+------+--------------+-------------+------------+------------+
| ID   | Name         | CountryCode | District   | Population |
+------+--------------+-------------+------------+------------+
...
| 4079 | Rafah        | PSE         | Rafah      |      92020 |
+------+--------------+-------------+------------+------------+
9 rows in set (0.00 sec)
```

a. 此次读操作出现了什么问题？这是一个脏读。由于隔离级别是 READ UNCOMMITTED，SELECT 结果反映了 t1 会话中未提交的事务。

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

16. 在 t1 会话中，使用 ROLLBACK 语句取消事务。再次选择 ID 大于 4070 的城市。在 t1 终端窗口中输入以下内容，得到的结果如下所示：

```
t1> ROLLBACK;
Query OK, 0 rows affected (0.01 sec)

t1> SELECT * FROM City WHERE ID > 4070;
+------+-------------+-------------+------------+------------+
| ID   | Name        | CountryCode | District   | Population |
+------+-------------+-------------+------------+------------+
...
| 4080 | New City    | ATA         |            |          0 |
+------+-------------+-------------+------------+------------+
10 rows in set (0.00 sec)
```

   – DELETE 事务被回滚。因此，之前插入的行仍然存在。

17. 在 t2 会话中，选择 ID 大于 4070 的城市。在 t2 终端窗口中输入以下内容，得到的结果如下所示：

```
t2> SELECT * FROM City WHERE ID > 4070;
+------+-------------+-------------+------------+------------+
| ID   | Name        | CountryCode | District   | Population |
+------+-------------+-------------+------------+------------+
...
| 4080 | New City    | ATA         |            |          0 |
+------+-------------+-------------+------------+------------+
10 rows in set (0.00 sec)
```

   a. 结果是否与上一步中在 t1 会话中所查询的结果一样？是的，表内容和执行 DELETE 之前一样。

18. 在 t2 会话中，取消此会话注册的任何事务。在 t2 终端窗口中输入以下内容，得到的结果如下所示：

```
t2> ROLLBACK;
Query OK, 0 rows affected (0.01 sec)
```

19. 退出两个 mysql 会话以及两个终端窗口，为接下来的练习做好准备：

```
t1> EXIT
Bye
$ exit

t2> EXIT
Bye
$ exit
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

第 9 课的练习：存储引擎

第 9 章

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

第 9 课的练习：概览

练习概览

通过这些练习，可测试您对 MySQL 中的 InnoDB 和其他存储引擎的掌握程度。上机练习是针对 Oracle Linux 操作系统环境编写的，Oracle 课堂中将提供此环境。对于非 Oracle 课堂环境，可能需要对文件位置进行一些调整。

假设

- 在进行这些练习之前已经完成了 MySQL 服务器安装。
- 在终端窗口中以 `oracle` 用户身份登录。
- 您可以通过命令行提示符访问 `mysql` 客户机。

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

练习 9-1：测验 – InnoDB 存储引擎

概览

在本练习中，您将回答有关 InnoDB 存储引擎的问题。

持续时间

完成本练习大约需要 6 分钟。

测验问题

请从所提供的每道多选题或正/误判断题中选择最佳答案。

1. 如何禁用 InnoDB 存储引擎？
    a. 仅当从源编译 MySQL 时，通过使用 `--without-innodb` 选项
    b. 通过使用 `--skip-innodb` 选项启动服务器
    c. 通过发出语句 `SET GLOBAL have_innodb = 0`。需要 **SUPER** 特权来执行此操作
    d. 不能禁用存储引擎
2. 在默认 InnoDB 配置中，`.ibd` 文件的数量等于 InnoDB 表数量。
    a. 正确
    b. 错误
3. 可能会出现 _____，因为 InnoDB 在事务过程中直到需要锁时才获取锁。
    a. 行级别锁定
    b. 死锁
    c. 虚读
    d. 以上都是
4. 可以从系统表空间中删除数据文件来减小其大小。
    a. 正确
    b. 错误
5. InnoDB 通过使用 _____ 和 _____ 键加强表关系来提供引用完整性。
    a. 候选和唯一
    b. 主和唯一
    c. 主和外
    d. 唯一和外

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

练习解答 9-1：测验 — InnoDB 存储引擎

测验解答

1. **b.** 通过使用 `--skip-innodb` 选项启动服务器。

   注：使用 `--skip-innodb` 选项时，应将 `default-storage-engine` 选项设置为除 InnoDB 之外的引擎。自版本 5.5 起，InnoDB 是默认引擎。如果不配置 `default-storage-engine`，则使用 `--skip-innodb` 选项时服务器不启动
2. **a.** 正确
3. **c.** 死锁
4. **b.** 错误
5. **c.** 主和外

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

练习 9-2：设置和确认 InnoDB 设置

概览

在本练习中，使用各种方法设置和显示当前 MySQL 会话的 InnoDB 设置。

持续时间

完成本练习大约需要 30 分钟。

任务

1. 编辑当前配置文件（来自前面练习的 `my.cnf`）的 `[mysqld]` 部分，将 InnoDB 缓冲区池设置为您的物理 RAM 的大约 50% 并在缓冲区池中按 RAM 的 GB 创建一个缓冲区池实例。
2. 停止并重新启动 MySQL 服务器来实现配置文件更改。
3. 在新终端窗口中启动 `mysql` 客户机，使用 `SELECT` 语句确认当前会话的默认存储引擎。
4. 确认 `INFORMATION_SCHEMA` 数据库包含与 InnoDB 相关的表。
5. 列出与 InnoDB 相关的所有变量的设置。
    a. 多个表空间 (file-per-table) 设置是 ON 还是 OFF？
    b. 什么是自动扩展增量设置？
    c. InnoDB 数据文件路径是什么？
    d. 是否存在任何清除滞后？
    e. 缓冲区池大小是多少？
    f. 存在多少缓冲区池实例？
6. 编辑现有配置文件 (`/etc/my.cnf`) 将自动扩展增量更改为 128 MB。
7. 通过再次列出与 InnoDB 相关的所有变量来确认对配置文件进行的更改。将结果与前面的列表进行比较。
    a. 什么是自动扩展增量设置？
8. 创建名为 `CityLanguage` 的新表，并将存储引擎设置为 MEMORY。
    – 这将使用通过特定存储引擎设置创建新表的方法。
    a. 使用下面的 `CREATE TABLE` 语句创建包含 `City`、`Country`、`CountryCode` 和 `Language` 列的表：

```
mysql> CREATE TABLE CityLanguage (
    -> City CHAR(35),
    -> Country CHAR(35),
    -> CountryCode CHAR(3),
    -> Language CHAR(30)
    -> ) ENGINE=MEMORY;
```

    b. 确认存储引擎设置。
9. 更改新的 `CityLanguage` 表，从而其使用 InnoDB 存储引擎并确认其已经更改。

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

练习解答 9-2：设置和确认 InnoDB 设置

任务

1. 编辑当前配置文件（来自前面练习的 my.cnf）的 [mysqld] 部分，将 InnoDB 缓冲区池设置为您的物理 RAM 的大约 50% 并在缓冲区池中按 RAM 的 GB 创建一个缓冲区池实例。

   a. 要确定总物理内存，请在终端窗口中输入以下内容：

```
$ su -
Password: oracle
# head -1 /proc/meminfo
MemTotal:        4059764 kB
```

   前面输出中显示的总物理内存大约是 4 GB。如果服务器具有 4 GB RAM，请在下面的步骤中将缓冲区池大小设置为 2 GB，将缓冲区池实例的数量设置为 2。根据需要选择不同值。例如，如果服务器具有 8 GB RAM，则选择缓冲区池大小为 4 GB，四个缓冲区池实例。

   b. 使用 gedit（或您选择的其他文本编辑器）打开 my.cnf 配置文件。

```
# gedit /etc/my.cnf
```

   c. 向 [mysqld] 部分添加以下行。选择适合步骤 1a 中确定的系统的值：

```
[mysqld]
innodb_buffer_pool_size=2GB
innodb_buffer_pool_instances=2
...
```

   d. 保存并关闭配置文件。

2. 停止并重新启动 MySQL 服务器来实现配置文件更改。在终端窗口中输入以下内容，得到的结果如下所示：

```
# service mysql restart
Shutting down MySQL.....                   [ OK ]
Starting MySQL.....                        [ OK ]
```

3. 在新终端窗口中启动 mysql 客户机，使用 SELECT 语句确认当前会话的默认存储引擎。在终端窗口中输入以下内容，得到的结果如下所示：

```
$ mysql -uroot -poracle
...
mysql> SELECT @@default_storage_engine;
+--------------------------+
| @@default_storage_engine |
+--------------------------+
| InnoDB                   |
+--------------------------+
1 row in set (0.00 sec)
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

4. 确认 INFORMATION_SCHEMA 数据库包含与 InnoDB 相关的表。在终端窗口中输入以下内容，得到的结果如下所示：

```
mysql> USE INFORMATION_SCHEMA
...
Database changed

mysql> SHOW TABLES LIKE 'INNODB%';
+---------------------------------------+
| Tables_in_information_schema (INNODB%) |
+---------------------------------------+
| INNODB_LOCKS                          |
| INNODB_TRX                            |
| INNODB_SYS_DATAFILES                  |
| INNODB_LOCK_WAITS                     |
| INNODB_SYS_TABLESTATS                 |
| INNODB_CMP                            |
| INNODB_FT_BEING_DELETED               |
| INNODB_CMP_RESET                      |
| INNODB_CMP_PER_INDEX                  |
| INNODB_CMPMEM_RESET                   |
| INNODB_FT_DELETED                     |
| INNODB_BUFFER_PAGE_LRU                |
| INNODB_FT_INSERTED                    |
| INNODB_CMPMEM                         |
| INNODB_SYS_INDEXES                    |
| INNODB_SYS_TABLES                     |
| INNODB_SYS_FIELDS                     |
| INNODB_CMP_PER_INDEX_RESET            |
| INNODB_BUFFER_PAGE                    |
| INNODB_FT_DEFAULT_STOPWORD            |
| INNODB_FT_INDEX_TABLE                 |
| INNODB_FT_INDEX_CACHE                 |
| INNODB_SYS_TABLESPACES                |
| INNODB_METRICS                        |
| INNODB_SYS_FOREIGN_COLS               |
| INNODB_FT_CONFIG                      |
| INNODB_BUFFER_POOL_STATS              |
| INNODB_SYS_COLUMNS                    |
| INNODB_SYS_FOREIGN                    |
+---------------------------------------+
29 rows in set (0.01 sec)
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

5. 列出与 InnoDB 相关的所有变量的设置。在终端窗口中输入以下内容，得到的结果如下所示：

```
mysql> SHOW VARIABLES LIKE '%innodb%';
+-----------------------------------------+------------------------+
| Variable_name                           | Value                  |
+-----------------------------------------+------------------------+
| ignore_builtin_innodb                   | OFF                    |
...
| innodb_api_trx_level                    | 0                      |
| innodb_autoextend_increment             | 64                     |
| innodb_autoinc_lock_mode                | 1                      |
| innodb_buffer_pool_dump_at_shutdown     | OFF                    |
| innodb_buffer_pool_dump_now             | OFF                    |
| innodb_buffer_pool_filename             | ib_buffer_pool         |
| innodb_buffer_pool_instances            | 2                      |
| innodb_buffer_pool_load_abort           | OFF                    |
| innodb_buffer_pool_load_at_startup      | OFF                    |
| innodb_buffer_pool_load_now             | OFF                    |
| innodb_buffer_pool_size                 | 2147483648             |
| innodb_change_buffer_max_size           | 25                     |
...
| innodb_concurrency_tickets              | 5000                   |
| innodb_data_file_path                   | ibdata1:12M:autoextend |
| innodb_data_home_dir                    |                        |
...
| innodb_file_format_max                  | Antelope               |
| innodb_file_per_table                   | ON                     |
| innodb_flush_log_at_timeout             | 1                      |
...
| innodb_max_dirty_pages_pct_lwm          | 0                      |
| innodb_max_purge_lag                    | 0                      |
| innodb_max_purge_lag_delay              | 0                      |
...
| innodb_version                          | 1.2.10                 |
| innodb_write_io_threads                 | 4                      |
+-----------------------------------------+------------------------+
115 rows in set (0.01 sec)
```

a. 多个表空间 (file-per-table) 设置是 ON 还是 OFF？**ON**

b. 什么是自动扩展增量设置？**64**

c. InnoDB 数据文件路径是什么？**ibdata1:12M:autoextend**

d. 是否存在任何清除滞后？否；**innodb_max_purge_lag** 的值是 **0**

e. 缓冲区池大小是多少？前面输出中的 **2147483648** 字节。此问题和后面问题的答案取决于步骤 1c 中使用的值

f. 存在多少缓冲区池实例？前面输出中的**二**

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

6. 编辑现有配置文件 (/etc/my.cnf) 将自动扩展增量更改为 128 MB。

   注：必须从以 root 身份登录的终端中启动编辑器。

   a. 按如下方式编辑 /etc/my.cnf 文件：

```
[mysqld]
innodb_autoextend_increment=128
innodb_buffer_pool_size=2GB
...
```

   b. 保存并关闭配置文件。

   c. 停止并重新启动服务器来实现更改。在终端窗口中输入以下内容，得到的结果如下所示：

   – 在停止服务器之前，请确保已退出当前正在运行的所有 mysql 客户机。

```
# service mysql restart
Shutting down MySQL......                  [ OK ]
Starting MySQL......                       [ OK ]
```

7. 通过再次列出与 InnoDB 相关的所有变量来确认对配置文件进行的更改。将结果与前面的列表进行比较。在终端窗口中输入以下内容，得到的结果如下所示：

```
$ mysql -uroot -poracle
...
mysql> SHOW VARIABLES LIKE '%innodb%';
+---------------------------------+------------------------+
| Variable_name                   | Value                  |
+---------------------------------+------------------------+
| ignore_builtin_innodb           | OFF                    |
| innodb_adaptive_flushing        | ON                     |
...
| innodb_autoextend_increment     | 128                    |
...
+---------------------------------+------------------------+
```

   a. 什么是自动扩展增量设置？**128**

8. 创建名为 CityLanguage 的新表（将存储引擎设置为 MEMORY）并确认设置。在 mysql 提示符下输入以下内容，得到的结果如下所示：

   a. 使用下面的 **CREATE TABLE** 语句创建包含 City、Country、CountryCode 和 Language 列的表：

```
mysql> USE world_innodb;
...
Database changed
mysql> CREATE TABLE CityLanguage (
    -> City CHAR(35),
    -> Country CHAR(35),
    -> CountryCode CHAR(3),
    -> Language CHAR(30)
    -> ) ENGINE=MEMORY;
Query OK, 0 rows affected (0.11 sec)
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

b. 确认存储引擎设置：

```
mysql> SHOW CREATE TABLE CityLanguage\G
*************************** 1. row ********************
       Table: CityLanguage
Create Table: CREATE TABLE `citylanguage` (
  `City` char(35) DEFAULT NULL,
  `Country` char(35) DEFAULT NULL,
  `CountryCode` char(3) DEFAULT NULL,
  `Language` char(30) DEFAULT NULL
) ENGINE=MEMORY DEFAULT CHARSET=latin1
1 row in set (0.00 sec)
```

9. 更改新的 CityLanguage 表，从而其使用 InnoDB 存储引擎并确认其已经更改。在 mysql 提示符下输入以下内容，得到的结果如下所示：

```
mysql> ALTER TABLE CityLanguage ENGINE=InnoDB;
Query OK, 0 rows affected (0.80 sec)
Records: 0  Duplicates: 0  Warnings: 0

mysql> SHOW CREATE TABLE CityLanguage\G
*************************** 1. row ***************************
       Table: CityLanguage
Create Table: CREATE TABLE `CityLanguage` (
  `City` char(35) DEFAULT NULL,
  `Country` char(35) DEFAULT NULL,
  `CountryCode` char(3) DEFAULT NULL,
  `Language` char(30) DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1
1 row in set (0.00 sec)
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

第 10 课的练习：分区

第 10 章

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

第 10 课的练习：概览

练习概览

通过这些练习，可测试您对 MySQL 中表分区的掌握程度。上机练习是针对 Oracle Linux 操作系统环境编写的，Oracle 课堂中将提供此环境。对于非 Oracle 课堂环境，可能需要对文件位置进行一些调整。

假设

- 在进行这些练习之前已经完成了 MySQL 服务器安装。
- 在终端窗口中以 `oracle` 用户身份登录。
- 您可以通过命令行提示符访问 `mysql` 客户机。

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

练习 10-1：测验 – MySQL 分区

概览

在本测验中，您将回答有关 MySQL 分区的问题。

持续时间

完成本练习大约需要 10 分钟。

测验问题

请从所提供的每道多选题或正/误判断题中选择最佳答案。

1. 使用分区的主要原因是为了提高数据库性能。
    a. 正确
    b. 错误
2. 将表的不同行数分配到不同的物理分区称为 _____。
    a. 数据库分区
    b. 物理分区
    c. 水平分区
    d. 以上都不是
3. 按如下所示执行用于确定您的当前 MySQL 服务器是否支持分区的语句：

```
SHOW ENGINES;
```

    a. 正确
    b. 错误
4. 根据与离散值集之一匹配的列选择分区时使用 _____ 分区类型。
    a. LIST
    b. 子分区
    c. RANGE
    d. LINEAR
5. 使用 COLUMNS 分区变体（以及 RANGE 和 LIST 类型）来允许使用多个列。在以下情况时将考虑这些列：将行放入分区中，以及确定将检查哪些分区来匹配分区删改中的行。
    a. 正确
    b. 错误
6. 可以对按以下分区类型进行了分区的表进行子分区，以便进一步划分每个分区。
    a. LIST 和 LINEAR
    b. HASH 和 KEY
    c. 仅 RANGE
    d. LIST 和 RANGE

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

7. 查询 INFORMATION_SCHEMA 数据库的 PARTITIONS 表来列出特定表（使用 WHERE 子句）的分区名称（以及各个分区说明）。
    a. 正确
    b. 错误

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

练习解答 10-1：测验 — MySQL 分区

测验解答

1. **a**. 正确
2. **c**. 水平分区
3. **b**. 错误。语句如下所示：

```
SHOW PLUGINS;
```

4. **a**. **LIST**
5. **a**. 正确
6. **d**. LIST 和 RANGE。子分区自身可以使用 HASH、LINEAR HASH、KEY 或 LINEAR KEY 分区
7. **a**. 正确

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

练习 10-2：创建和修改分区表

概览

在本练习中，您将创建分区表并修改该表。

持续时间

完成本练习大约需要 15 分钟。

任务

1. 启动新 `mysql` 客户机会话。确认全局变量 `innodb_file_per_table` 已启用（这是自 MySQL 5.6 以来的默认设置）。
2. 在 `world_innodb` 数据库内创建名为 `City_part` 的新表，其具有与 `City` 相同的列定义。
3. 使用 `SHOW TABLE STATUS` 语句确定该新表是否已分区。
4. 通过使用 `ALTER TABLE` 语句修改该新表来添加四个 `RANGE` 类型分区（使用 `ID` 列）：
   - `p0`（小于 1000 的值）
   - `p1`（小于 2000 的值）
   - `p2`（小于 3000 的值）
   - `p3`（小于最大值的值）
5. 现在您已经修改了该新表，请再次显示表分区状态。是否指示分区？
6. 通过插入原始表中的所有行，完成 `City` 表到 `City_part` 的复制。
7. 在单独的终端窗口中，检查新表文件的（`.par` 和 `.ibd`）的 MySQL 数据目录。注意 `.ibd` 文件的确切数量及其文件名和大小。

   注：由于 `info_file_per_table` 设置为 `ON`，因此将创建单独的 `.ibd` 分区文件。
8. 通过使用 `EXPLAIN PARTITIONS` 显示用于查询所有表数据的分区来确认 `City_part` 表分区。在结果中列出分区。
9. 确定将用于查询 `City_part` 表数据的分区，其中 `ID` 值小于 `2000`。其是否使用所有分区？如果不是，哪些分区用于此级别的查询？
10. 重新定义 `City_part` 表来将 `KEY` 分区用于三个单独的分区。
11. 通过使用 `EXPLAIN PARTITIONS` 显示用于查询所有表数据的分区来确认表分区修改。
12. 检查已修改表文件的 MySQL 数据目录。注意 `.ibd` 文件的确切数量及其文件名和大小。这些文件与先前使用 `RANGE` 分区的表文件有什么区别？
13. 查询 `INFORMATION_SCHEMA` 数据库中的 `PARTITIONS` 表，以了解 `City_part` 表中的分区名称。列出了多少分区？它们的名称是什么？

    注：将当前 `City_part` 表保持完好以在下一练习中使用。

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

练习解答 10-2：创建和修改分区表

任务

1. 启动新 mysql 客户机会话。确认全局变量 innodb_file_per_table 已启用（这是自 MySQL 5.6 以来的默认设置）。

   a. 在终端窗口中输入以下内容，得到的结果如下所示：

```
$ mysql --login-path=admin
...

mysql> SHOW VARIABLES LIKE 'innodb_file_per_table';
+-----------------------+-------+
| Variable_name         | Value |
+-----------------------+-------+
| innodb_file_per_table | ON    |
+-----------------------+-------+
1 row in set (0.00 sec)
```

2. 在 world_innodb 数据库内创建名为 City_part 的新表，其具有与 City 相同的列定义。在终端窗口中输入以下内容，得到的结果如下所示：

```
mysql> USE world_innodb
...
Database changed
mysql> CREATE TABLE City_part LIKE City;
Query OK, 0 rows affected (0.00 sec)
```

3. 使用 SHOW TABLE STATUS 语句确定该新表是否已分区。在终端窗口中输入以下内容，得到的结果如下所示：

```
mysql> SHOW TABLE STATUS LIKE 'City_part'\G
*************************** 1. row ***************************
           Name: City_part
         Engine: InnoDB
        Version: 10
     Row_format: Compact
           Rows: 0
 Avg_row_length: 0
    Data_length: 16384
Max_data_length: 0
   Index_length: 16384
      Data_free: 0
 Auto_increment: 1
    Create_time: 2012-10-15 18:00:02
    Update_time: NULL
     Check_time: NULL
      Collation: latin1_swedish_ci
       Checksum: NULL
 Create_options:
        Comment:
1 row in set (0.00 sec)
```

   – 没有指示此表中进行了分区。

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

4. 通过使用 ALTER TABLE 语句修改该新表来添加四个 RANGE 类型分区（使用 ID 列）。在终端窗口中输入以下内容，得到的结果如下所示：

```
mysql> ALTER TABLE City_part PARTITION BY RANGE (ID) (
    -> PARTITION p0 VALUES LESS THAN (1000),
    -> PARTITION p1 VALUES LESS THAN (2000),
    -> PARTITION p2 VALUES LESS THAN (3000),
    -> PARTITION p3 VALUES LESS THAN MAXVALUE
    -> );
Query OK, 0 rows affected (4.34 sec)
Records: 0  Duplicates: 0  Warnings: 0
```

5. 现在您已经修改了该新表，请再次显示表分区状态。

   a. 在终端窗口中输入以下内容，得到的结果如下所示：

```
mysql> SHOW TABLE STATUS LIKE 'City_part'\G
*************************** 1. row ***************************
           Name: City_part
         Engine: InnoDB
        Version: 10
     Row_format: Compact
           Rows: 4
 Avg_row_length: 16384
    Data_length: 65536
Max_data_length: 0
   Index_length: 65536
      Data_free: 25165824
 Auto_increment: 1
    Create_time: NULL
    Update_time: NULL
     Check_time: NULL
      Collation: latin1_swedish_ci
       Checksum: NULL
 Create_options: partitioned
        Comment:
1 row in set (0.00 sec)
```

   b. 是否指示分区？是

6. 通过插入原始表中的所有行，完成 City 表到 City_part 的复制。在终端窗口中输入以下内容，得到的结果如下所示：

```
mysql> INSERT INTO City_part SELECT * FROM City;
Query OK, 4079 rows affected, 1 warning (1.63 sec)
Records: 4079  Duplicates: 0  Warnings: 1
```

该语句发出警告，这是因为以下事实情况：您使用的是基于语句的日志记录，同时将另一个表中的记录插入使用自动增量列的表中。我们不保证此组合可以安全执行复制。由于您没有对复制使用二进制日志，所以可以放心地忽略本课中的此警告。

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

7. 在单独的终端窗口中，检查新表文件的（.par 和 .ibd）的 MySQL 数据目录。

   a. 在单独的终端窗口中输入以下内容，得到的结果如下所示：

```
$ su -
Password: oracle
# cd /var/lib/mysql/world_innodb; ls -l
...
-rw-rw---- 1 mysql mysql   8710 Feb  6 04:55 City_part.frm
-rw-rw---- 1 mysql mysql     32 Feb  6 04:55 City_part.par
-rw-rw---- 1 mysql mysql 212992 Feb  6 04:57 City_part#P#p0.ibd
-rw-rw---- 1 mysql mysql 212992 Feb  6 04:57 City_part#P#p1.ibd
-rw-rw---- 1 mysql mysql 212992 Feb  6 04:57 City_part#P#p2.ibd
-rw-rw---- 1 mysql mysql 212992 Feb  6 04:57 City_part#P#p3.ibd
...
```

   b. 注意 .ibd 文件的确切数量及其文件名和大小：

   – .frm 和 .par 文件非常小。
   – 您的文件大小可能与上面显示的那些不同。最重要的是注意存在哪些文件及其大小差异。

8. 通过使用 EXPLAIN PARTITIONS 显示用于查询所有表数据的分区来确认 City_part 表分区。在 mysql 窗口中输入以下内容，得到的结果如下所示：

```
mysql> EXPLAIN PARTITIONS SELECT * FROM City_part\G
*************************** 1. row ***************************
           id: 1
  select_type: SIMPLE
        table: City_part
   partitions: p0,p1,p2,p3
         type: ALL
possible_keys: NULL
          key: NULL
      key_len: NULL
          ref: NULL
         rows: 4079
        Extra:
1 row in set (0.00 sec)
```

   – EXPLAIN 返回用于查询的检查行的估计数量，其可能与实际行数有很大差别。

9. 确定将用于查询 City_part 表数据的分区，其中 **ID** 值小于 **2000**。

   a. 在终端窗口中输入以下内容，得到的结果如下所示：

```
mysql> EXPLAIN PARTITIONS SELECT * FROM City_part WHERE ID < 2000\G
*************************** 1. row ***************************
           id: 1
  select_type: SIMPLE
        table: City_part
   partitions: p0,p1
         type: range
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

```
possible_keys: PRIMARY
          key: PRIMARY
      key_len: 4
          ref: NULL
         rows: 1996
        Extra: Using where
1 row in set (0.00 sec)
```

b. 其是否使用所有分区？否。仅 p0 和 p1 显示在 partitions 列中。

10. 重新定义 City_part 表来将 **KEY** 分区用于三个单独的分区。在终端窗口中输入以下内容，得到的结果如下所示：

```
mysql> ALTER TABLE City_part
    -> PARTITION BY KEY (ID) PARTITIONS 3;
Query OK, 4079 rows affected (5.24 sec)
Records: 4079  Duplicates: 0  Warnings: 0
```

11. 通过使用 EXPLAIN PARTITIONS 显示用于查询所有表数据的分区来确认表分区修改。在终端窗口中输入以下内容，得到的结果如下所示：

```
mysql> EXPLAIN PARTITIONS SELECT * FROM City_part\G
*************************** 1. row ***************************
           id: 1
  select_type: SIMPLE
        table: City_part
   partitions: p0,p1,p2
         type: ALL
possible_keys: NULL
          key: NULL
      key_len: NULL
          ref: NULL
         rows: 2734
        Extra:
1 row in set (0.00 sec)
```

12. 检查已修改表文件的 MySQL 数据目录。注意 .ibd 文件的确切数量及其文件名和大小。

a. 以 root 身份登录终端窗口并在其中输入以下内容，得到的结果如下所示：

```
# cd /var/lib/mysql/world_innodb; ls -l
...
-rw-rw---- 1 mysql mysql   8710 Feb  6 05:06 City_part.frm
-rw-rw---- 1 mysql mysql     32 Feb  6 05:06 City_part.par
-rw-rw---- 1 mysql mysql 245760 Feb  6 05:06 City_part#P#p0.ibd
-rw-rw---- 1 mysql mysql 278528 Feb  6 05:06 City_part#P#p1.ibd
-rw-rw---- 1 mysql mysql 245760 Feb  6 05:06 City_part#P#p2.ibd
...
```

b. 这些文件与先前使用 **RANGE** 分区的表文件有什么区别？现在有三个 .ibd 文件，每个包含大约三分之一的表数据。

– 您的文件大小可能与上面显示的那些不同。最重要的是注意存在哪些文件及其大小差异。

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

13. 查询 INFORMATION_SCHEMA 数据库中的 PARTITIONS 表，以了解 City_part 表中的分区名称。在 mysql 窗口中输入以下内容，得到的结果如下所示：

```
mysql> SELECT TABLE_NAME,
    -> GROUP_CONCAT(PARTITION_NAME)
    -> FROM INFORMATION_SCHEMA.PARTITIONS
    -> WHERE TABLE_SCHEMA='world_innodb'
    -> AND TABLE_NAME='City_part';
+------------+------------------------------+
| TABLE_NAME | GROUP_CONCAT(PARTITION_NAME) |
+------------+------------------------------+
| City_part  | p0,p1,p2                     |
+------------+------------------------------+
1 row in set (0.00 sec)
```

注：将当前 City_part 表保持完好以在下一练习中使用。

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

练习 10-3：删除表中的分区

概览

在本练习中，您将从分区表中删除分区并从表中删除所有分区。

持续时间

完成本练习大约需要 10 分钟。

任务

1. 使用上一练习中的 `mysql` 客户机，从 `City_part` 表中删除第一个分区 (`p0`)。此操作是否起作用？如果否，它为何不起作用？
2. 将当前 `City_part` 表恢复为第一个 `RANGE` 分区的配置：
   - `p0`（小于 1000 的值）
   - `p1`（小于 2000 的值）
   - `p2`（小于 3000 的值）
   - `p3`（小于最大值的值）
3. 验证每个新分区的文件大小。
4. 尝试再次从 `City_part` 表中删除第一个分区 (`p0`)。此操作是否起作用？为什么？
5. 通过使用 `EXPLAIN PARTITIONS` 显示现在用于查询所有表数据的分区来确认对 `City_part` 表分区进行的修改。哪些分区被留下？
6. 检查 MySQL 数据目录。已删除的分区文件是否仍然存在？剩余 `.ibd` 文件的大小是否与删除分区之前相同？
   - 与练习 10-2 步骤 7 中的输出结果进行比较。
7. 删除 `City_part` 表的分区并将其恢复为其原始的非分区状态。
8. 现在您已经再次修改了 `City_part`，请再次显示表分区状态。输出是否指示表进行了分区？
9. 通过使用 `EXPLAIN PARTITIONS`，验证 `City_part` 表分区现在是否已不存在。
10. 最后一次检查 MySQL 数据目录。哪些 `City_part` 文件被留下？其名称和大小是什么？

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

练习解答 10-3：删除表中的分区

任务

1. 使用上一练习中的 mysql 客户机，从 City_part 表中删除第一个分区 (p0)。

   a. 在终端窗口中输入以下内容，得到的结果如下所示：

```
mysql> ALTER TABLE City_part DROP PARTITION p0;
ERROR 1512 (HY000): DROP PARTITION can only be used on RANGE/LIST
partitions
```

   b. 此操作是否起作用？否，因为这是 KEY 分区表。

2. 将当前 City_part 表恢复为第一个 RANGE 分区的配置。在终端窗口中输入以下内容，得到的结果如下所示：

```
mysql> ALTER TABLE City_part PARTITION BY RANGE (id) (
    -> PARTITION p0 VALUES LESS THAN (1000),
    -> PARTITION p1 VALUES LESS THAN (2000),
    -> PARTITION p2 VALUES LESS THAN (3000),
    -> PARTITION p3 VALUES LESS THAN MAXVALUE
    -> );
Query OK, 4079 rows affected (7.73 sec)
Records: 4079  Duplicates: 0  Warnings: 0
```

3. 验证每个新分区的文件大小。以 root 身份登录终端窗口并在其中输入以下内容，得到的结果如下所示：

```
# cd /var/lib/mysql/world_innodb; ls -l
...
-rw-rw---- 1 mysql mysql   8710 Feb  6 05:31 City_part.frm
-rw-rw---- 1 mysql mysql     32 Feb  6 05:31 City_part.par
-rw-rw---- 1 mysql mysql 327680 Feb  6 05:31 City_part#P#p0.ibd
-rw-rw---- 1 mysql mysql 311296 Feb  6 05:31 City_part#P#p1.ibd
-rw-rw---- 1 mysql mysql 360448 Feb  6 05:31 City_part#P#p2.ibd
-rw-rw---- 1 mysql mysql 311296 Feb  6 05:31 City_part#P#p3.ibd
...
```

4. 尝试再次从 City_part 表中删除第一个分区 (p0)。

   a. 在终端窗口中输入以下内容，得到的结果如下所示：

```
mysql> ALTER TABLE City_part DROP PARTITION p0;
Query OK, 0 rows affected (0.02 sec)
Records: 0  Duplicates: 0  Warnings: 0
```

   b. 此操作是否起作用？是。RANGE 分区表允许使用 DROP PARTITION。

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

5. 通过使用 EXPLAIN PARTITIONS 显示现在用于查询所有表数据的分区来确认对 City_part 表分区进行的修改。在终端窗口中输入以下内容，得到的结果如下所示：

```
mysql> EXPLAIN PARTITIONS SELECT * FROM City_part\G
*************************** 1. row ***************************
           id: 1
  select_type: SIMPLE
        table: City_part
   partitions: p1,p2,p3
         type: ALL
possible_keys: NULL
          key: NULL
      key_len: NULL
          ref: NULL
         rows: 3081
        Extra: NULL
1 row in set (0.00 sec)
```

哪些分区被留下？在前面步骤中删除了分区 p0，将留下分区 p1、p2 和 p3。

6. 检查 MySQL 数据目录。

   a. 以 root 身份登录终端窗口并在其中输入以下内容，得到的结果如下所示：

```
# cd /var/lib/mysql/world_innodb; ls -l
...
-rw-rw---- 1 mysql mysql   8710 Feb  6 05:21 City_part.frm
-rw-rw---- 1 mysql mysql     32 Feb  6 05:21 City_part.par
-rw-rw---- 1 mysql mysql 311296 Feb  6 05:12 City_part#P#p1.ibd
-rw-rw---- 1 mysql mysql 360448 Feb  6 05:12 City_part#P#p2.ibd
-rw-rw---- 1 mysql mysql 311296 Feb  6 05:12 City_part#P#p3.ibd
...
```

   b. 已删除的分区文件是否仍然存在？否

   c. 剩余 .ibd 文件的大小是否与删除分区之前相同？是，当表首次更改为 RANGE 表时，这三个分区的大小与之前相同。但是，p0 分区及其所有内容现在都已不存在。

   – 您的文件大小可能与上面显示的那些不同。最重要的是注意存在哪些文件及其大小差异。

7. 删除 City_part 表的分区并将其恢复为其原始的非分区状态。在终端窗口中输入以下内容，得到的结果如下所示：

```
mysql> ALTER TABLE City_part REMOVE PARTITIONING;
Query OK, 3080 rows affected (0.12 sec)
Records: 3080  Duplicates: 0  Warnings: 0
```

8. 现在您已经再次修改了 City_part，请再次显示表分区状态。在终端窗口中输入以下内容，得到的结果如下所示：

```
mysql> SHOW TABLE STATUS LIKE 'City_part'\G
*************************** 1. row ***************************
           Name: City_part
         Engine: InnoDB
        Version: 10
     Row_format: Compact
           Rows: 3081
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

```
 Avg_row_length: 101
    Data_length: 311296
Max_data_length: 0
   Index_length: 98304
      Data_free: 0
 Auto_increment: 4081
    Create_time: 2013-02-06 05:35:01
    Update_time: NULL
     Check_time: NULL
      Collation: latin1_swedish_ci
       Checksum: NULL
 Create_options:
        Comment:
1 row in set (0.00 sec)
```

a. 输出是否指示表进行了分区？否。Create_options 列为空并且不包含字符串“partitioned”来指示已启用分区。

9. 通过使用 EXPLAIN PARTITIONS，验证 City_part 表分区现在是否已不存在。在终端窗口中输入以下内容，得到的结果如下所示：

```
mysql> EXPLAIN PARTITIONS SELECT * FROM City_part\G
*************************** 1. row ***************************
           id: 1
  select_type: SIMPLE
        table: City_part
   partitions: NULL
         type: ALL
possible_keys: NULL
          key: NULL
      key_len: NULL
          ref: NULL
         rows: 3081
        Extra:
1 row in set (0.00 sec)
```

10. 最后一次检查 MySQL 数据目录。哪些 City_part 文件被留下？其名称和大小是什么？在终端窗口中输入以下内容，得到的结果如下所示：

```
# cd /var/lib/mysql/world_innodb; ls -l
...
-rw-rw---- 1 mysql mysql   8710 Mar 23 19:12 City_part.frm
-rw-rw---- 1 mysql mysql 475136 Mar 23 19:12 City_part.ibd
...
```

– 您的文件大小可能与上面显示的那些不同。最重要的是注意存在哪些文件及其大小差异。

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

第 11 课的练习：用户管理

第 11 章

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

第 11 课的练习：概览

练习概览

通过这些练习，可测试您对 MySQL 用户管理的掌握程度。上机练习是针对 Oracle Linux 操作系统环境编写的，Oracle 课堂中将提供此环境。对于非 Oracle 课堂环境，可能需要对文件位置进行一些调整。

假设

- 在进行这些练习之前已经完成了 MySQL 服务器安装。
- 在终端窗口中以 `oracle` 用户身份登录。
- 您可以通过命令行提示符访问 `mysql` 客户机。

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

练习 11-1：测验 — MySQL 用户管理

概览

在本测验中，您将回答有关 MySQL 用户管理的问题。

持续时间

本练习应该需要大约五分钟来完成。

测验问题

请从所提供的每道多选题或正/误判断题中选择最佳答案。

1. 要确定哪些帐户可以在不指定口令的情况下使用，可使用以下语句：

```
SELECT Host, User FROM mysql.user WHERE Password = '';
```

   a. 正确
   b. 错误

2. 假定与指定 MySQL 用户名关联的帐户有以下特权设置，其中 **Select_priv** 列指示 **SELECT** 全局特权的设置：

```
mysql> SELECT Host, User, Select_priv FROM mysql.user
    -> WHERE User = 'Sasha';
+--------------+-------+-------------+
| Host         | User  | Select_priv |
+--------------+-------+-------------+
| 62.220.12.66 | Sasha | N           |
| 62.220.12.%  | Sasha | Y           |
| 62.220.%     | Sasha | N           |
+--------------+-------+-------------+
```

   注：Select_priv 列指示已在全局范围 (***.***) 授予了第二个条目的 **SELECT** 特权。假定未向 **Sasha** 帐户授予其他任何授权表中的特权。

   用户 **Sasha** 在从以下主机连接时，能否从 MySQL 服务器上的任何表中选择数据？

   a. **62.220.12.66**
   b. **62.220.12.43**
   c. **62.220.42.43**
   d. **localhost**

3. 以下哪条用户处理语句是正确的？

   a. 要向现有用户授予其他特权，可使用 **ALTER USER** 或 **GRANT** 语句。
   b. 要删除用户，可使用 **DROP USER** 或 **REVOKE** 语句。
   c. 要创建用户，可使用 **CREATE USER** 语句或 **GRANT** 语句。

4. 假定创建了一个新用户，但尚未向该用户帐户授予任何特权。此时，该用户能做什么？

   a. 连接到服务器。
   b. 发出语句，如 **SELECT**。
   c. 发出语句，如 **SHOW DATABASES**。

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

5. 以下有关现有连接的特权更改效果的陈述，哪些是正确的？
    a. 在删除了用户后，由该用户发起的到服务器的连接会自动终止。
    b. 全局特权更改不会影响用户的现有连接。此类更改仅在下一次用户尝试连接时生效。
    c. 数据库特权更改仅在用户下一次尝试连接时生效。
    d. 数据库特权更改仅在用户下一次发出 **USE *<database>*** 语句时生效。
    e. 表和列特权会立即生效。
6. 具有 **root** 操作系统登录帐户的特权的服务器拥有不必要的文件系统访问权限，构成了安全风险。
    a. 正确
    b. 错误
7. 要禁用 TCP/IP 连接，可使用 _____ 选项启动服务器。
    a. **--skip-tcp_ip**
    b. **--skip-networking**
    c. **--tcp-ip_skip**

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

练习解答 11-1：测验 – MySQL 用户管理

测验解答

1. **a**. 正确
2. 用户 **Sasha** 在从以下主机连接时，能否从 MySQL 服务器上的任何表中选择数据？
   a. **不能**。**62.220.12.66** 是与用户 **Sasha** 正尝试从中连接的主机匹配的最具体的条目。因为该条目的 **SELECT** 特权为 **N**，因此 **Sasha** 无法从服务器上的任何表中选择。
   b. **能**。与 **62.220.12.43** 匹配的最具体的条目是 **62.220.12.%**。因为该条目的 **SELECT** 特权为 **Y**，因此 **Sasha** 可以从服务器上的任何表中选择。
   c. **不能**。与 **62.220.42.43** 匹配的最具体的条目是 **62.220.%**。因为该条目的 **SELECT** 特权为 **N**，因此 **Sasha** 不能从服务器上的任何表中选择。
   d. **不能**。没有与 **Sasha@localhost** 匹配的条目。因此，用户 **Sasha** 甚至不能从 **localhost** 连接到服务器。
3. **c**. 要创建用户，可使用 **CREATE USER** 语句或 **GRANT** 语句。**ALTER USER** 只能用于让用户的口令失效；如果要授予其他特权，应使用 **GRANT**。可以使用 **DROP USER** 语句彻底删除用户。使用 **REVOKE** 时，您也许可以撤消用户的所有特权，但用户帐户会保留，如以下示例中所示：

```
mysql> SHOW GRANTS FOR 'lennart';
+-------------------------------------+
| Grants for lennart@%                |
+-------------------------------------+
| GRANT USAGE ON *.* TO 'lennart'@'%' |
+-------------------------------------+
mysql> REVOKE usage ON *.* FROM 'lennart';
mysql> SHOW GRANTS FOR 'lennart';
+-------------------------------------+
| Grants for lennart@%                |
+-------------------------------------+
| GRANT USAGE ON *.* TO 'lennart'@'%' |
+-------------------------------------+
mysql> DROP USER 'lennart';
mysql> SHOW GRANTS FOR 'lennart';
ERROR 1141 (42000): There is no such grant defined for user 'lennart'
on host '%'
```

4. **a**. 连接到服务器。该用户帐户可用于连接到服务器以及在服务器上查看。像其他任何用户一样，该用户会获取显示有关服务器信息的语句（如 **SHOW VARIABLES** 或 **SHOW STATUS**）的完整输出。该用户无法获取有关数据库对象的信息，因为该用户没有任何针对数据库对象的特权。**SHOW DATABASES** 仅显示 **INFORMATION_SCHEMA** 数据库（这是在文件系统中没有物理表示形式的数据库）。

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

5. 以下有关现有连接的特权更改效果的陈述，哪些是正确的？
   a. **错误**。在删除了用户后，由该用户发起的到服务器的连接不会自动终止。
   b. **正确**。全局特权更改不会影响用户的现有连接。此类更改仅在下一次用户尝试连接时生效。
   c. **错误**。请参见下一项。
   d. **正确**。数据库特权更改仅在用户下一次发出 **USE *<database>*** 语句时生效。
   e. **正确**。表和列特权会立即生效。
6. **a**. 正确
7. **b**. --skip-networking

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

练习 11-2：创建、验证和删除用户

概览

在本练习中，您将创建、验证和删除一个匿名用户。

持续时间

本练习应该需要大约五分钟来完成。

任务

1. 使用 `SELECT` 语句验证服务器上没有匿名帐户。
2. 在本地主机上使用 `CREATE USER` 语句创建一个匿名帐户。
3. 再次运行上一条 `SELECT` 语句以确认现在存在该匿名帐户。
4. 使用 `DROP USER` 语句删除该匿名帐户。确认该匿名帐户不再存在。

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

练习解答 11-2：创建、验证和删除用户

任务

1. 使用 SELECT 语句验证服务器上没有匿名帐户。

   在终端窗口中输入以下内容，得到的结果如下所示：

```
$ mysql --login-path=admin
...

mysql> SELECT user, host, password FROM mysql.user
    -> WHERE user = '';
Empty set (0.00 sec)
```

2. 在本地主机上使用 CREATE USER 语句创建一个匿名帐户。

   在 mysql 提示符下输入以下内容，得到的结果如下所示：

```
mysql> CREATE USER ''@'localhost';
Query OK, 0 rows affected (0.00 sec)
```

3. 再次运行上一条 **SELECT** 语句以确认现在存在该匿名帐户。

   在 mysql 提示符下输入以下内容，得到的结果如下所示：

```
mysql> SELECT user, host, password FROM mysql.user
    -> WHERE user = '';
+------+-----------+----------+
| user | host      | password |
+------+-----------+----------+
|      | localhost |          |
+------+-----------+----------+
```

4. 使用 **DROP USER** 语句删除该匿名帐户。

   a. 在 mysql 提示符下输入以下内容，得到的结果如下所示：

```
mysql> DROP USER ''@'localhost';
Query OK, 0 rows affected (0.00 sec)
```

   b. 确认该匿名帐户不再存在：

```
mysql> SELECT user, host, password FROM mysql.user
    -> WHERE user = '';
Empty set (0.00 sec)
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

练习 11-3：设置 world_innodb 数据库的用户

概览

在本练习中，您将专门为 world_innodb 数据库创建一个新用户，并对其执行删除和验证操作。

持续时间

完成本练习大约需要 10 分钟。

任务

1. 使用上一练习中的 mysql 客户机会话，在主机 pc.example.com 上创建一个新用户 student。
2. 使用 SELECT 语句确认该新用户现在已包括在 mysql 数据库中。
3. 使用 GRANT 语句为用户 student 分配一个口令以及对 world_innodb 数据库的特权（SELECT、INSERT、DELETE、UPDATE）。
4. 使用 SHOW GRANT 语句显示用户 student 在 pc.example.com 主机上设置的授权。
5. 撤消当前在 pc.example.com 主机上授予 student 的 DELETE 和 UPDATE 特权。确认用户 student 在 pc.example.com 主机上现在设置的更新后授权。
6. 使用 GRANT 语句将 student 在 pc.example.com 主机上的口令更改为 NewPass。
7. 在 pc.example.com 主机上将 ALL 特权授予 student，并设置口令 NewPass，并且每小时最多仅允许 10 次连接。确认用户 student 在 pc.example.com 主机上现在设置的更新后授权。
8. 在主机 pc.example.com 上删除用户 student。确认该帐户不再存在。

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

练习解答 11-3：设置 world_innodb 数据库的用户

任务

1. 使用上一练习中的 mysql 客户机会话，在主机 pc.example.com 上创建一个新用户 student。

   在 mysql 提示符下输入以下内容，得到的结果如下所示：

```
mysql> CREATE USER 'student'@'pc.example.com';
Query OK, 0 rows affected (0.00 sec)
```

2. 使用 SELECT 语句确认该新用户现在已包括在 mysql 数据库中。

   在 mysql 提示符下输入以下内容，得到的结果如下所示：

```
mysql> SELECT user, host, password
    -> FROM mysql.user WHERE user = 'student';
+---------+----------------+----------+
| user    | host           | password |
+---------+----------------+----------+
| student | pc.example.com |          |
+---------+----------------+----------+
1 row in set (0.00 sec)
```

3. 使用 GRANT 语句为用户 student 分配一个口令以及对 world_innodb 数据库的特权（SELECT、INSERT、DELETE、UPDATE）。

   在 mysql 提示符下输入以下内容，得到的结果如下所示：

```
mysql> GRANT SELECT, INSERT, DELETE, UPDATE ON world_innodb.*
    -> TO 'student'@'pc.example.com'
    -> IDENTIFIED BY 'student_pass';
Query OK, 0 rows affected (0.00 sec)
```

4. 使用 SHOW GRANTS 语句显示用户 student 在 pc.example.com 主机上设置的授权。

   在 mysql 提示符下输入以下内容，得到的结果如下所示：

```
mysql> SHOW GRANTS FOR 'student'@'pc.example.com'\G
*************************** 1. row ***************************
Grants for student@pc.example.com: GRANT USAGE ON *.* TO
'student'@'pc.example.com' IDENTIFIED BY PASSWORD
'*832CA3798551FEA378AD34E02713320E28194DA2'
*************************** 2. row ***************************
Grants for student@pc.example.com: GRANT SELECT, INSERT, UPDATE,
DELETE ON `world_innodb`.* TO 'student'@'pc.example.com'
2 rows in set (0.00 sec)
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

5. 撤消当前在 pc.example.com 主机上授予 student 用户的 DELETE 和 UPDATE 特权。确认用户 student 在 pc.example.com 主机上现在设置的更新后授权。

   在 mysql 提示符下输入以下内容，得到的结果如下所示：

```
mysql> REVOKE DELETE, UPDATE ON world_innodb.*
    -> FROM 'student'@'pc.example.com';
Query OK, 0 rows affected (0.00 sec)

mysql> SHOW GRANTS for 'student'@'pc.example.com'\G
*************************** 1. row ***************************
Grants for student@pc.example.com: GRANT USAGE ON *.* TO
'student'@'pc.example.com' IDENTIFIED BY PASSWORD
'*832CA3798551FEA378AD34E02713320E28194DA2'
*************************** 2. row ***************************
Grants for student@pc.example.com: GRANT SELECT, INSERT ON
`world_innodb`.* TO 'student'@'pc.example.com'
2 rows in set (0.00 sec)
```

6. 使用 GRANT 语句将 student 用户在 pc.example.com 主机上的口令更改为 NewPass。

   在 mysql 提示符下输入以下内容，得到的结果如下所示：

```
mysql> GRANT USAGE ON *.* TO 'student'@'pc.example.com'
    -> IDENTIFIED BY 'NewPass';
Query OK, 0 rows affected (0.00 sec)
```

7. 在 pc.example.com 主机上将 ALL 特权授予 student 用户，并设置口令 NewPass，并且每小时最多仅允许 10 次连接。确认用户 student 在 pc.example.com 主机上现在设置的更新后授权。

   在 mysql 提示符下输入以下内容，得到的结果如下所示：

```
mysql> GRANT ALL ON world_innodb.* TO 'student'@'pc.example.com'
    -> IDENTIFIED BY 'NewPass'
    -> WITH MAX_CONNECTIONS_PER_HOUR 10;
Query OK, 0 rows affected (0.00 sec)

mysql> SHOW GRANTS for 'student'@'pc.example.com'\G
*************************** 1. row ***************************
Grants for student@pc.example.com: GRANT USAGE ON *.* TO
'student'@'pc.example.com' IDENTIFIED BY PASSWORD
'*9EE65E227E34770F6F707EC991CFF926CB294BC7' WITH
MAX_CONNECTIONS_PER_HOUR 10
*************************** 2. row ***************************
Grants for student@pc.example.com: GRANT ALL PRIVILEGES ON
`world_innodb`.* TO 'student'@'pc.example.com'
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

8. 在主机 pc.example.com 上删除用户 student。确认该帐户不再存在。

   a. 在 mysql 提示符下输入以下内容，得到的结果如下所示：

```
mysql> DROP USER 'student'@'pc.example.com';
Query OK, 0 rows affected (0.00 sec)
```

   b. 确认该帐户不再存在：

```
mysql> SELECT user, host, password FROM mysql.user
    -> WHERE user='student';
Empty set (0.00 sec)
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

练习 11-4：使用 PAM 验证插件

概览

在本练习中，您将启用 PAM 验证插件，并使用操作系统凭证登录到 MySQL。

持续时间

完成本练习大约需要 20 分钟。

任务

注：本练习需要使用多个终端窗口，用于多个 `mysql` 客户机会话。

1. 在终端窗口（以 `root` 身份登录）中执行以下命令以创建 Linux 用户 `pamuser1` 并设置口令 `oracle1`。

```
useradd pamuser1
passwd pamuser1
```

2. 使用文本编辑器在 `/etc/pam.d/` 目录中创建一个文件 `mysql-dba-course`，其中包含以下文本：

```
#%PAM-1.0
auth            include         password-auth
account         include         password-auth
```

3. 使文件 `/etc/shadow` 可组读取，并将其组设置为 `mysql`，从而使 MySQL 服务器进程可以使用 PAM 进行验证。使用以下 Linux 命令：

```
chmod 440 /etc/shadow
chgrp mysql /etc/shadow
```

4. 安装 `authentication_pam` 插件。
5. 显示所有插件以验证是否启用了 PAM 验证插件。
6. 创建一个 MySQL 用户 `pamuser1`，该用户可以从本地主机登录。使用 PAM 服务 `mysql-dba-course`，确保使用 PAM 验证确认了该用户。
7. 在 `world_innodb.City` 表上将 `SELECT` 特权授予新创建的 `pamuser1` 用户。
8. 在一个新终端窗口中，尝试使用 `cleartext` 插件和口令 `oracle1` 以 `pamuser1` 身份登录到 MySQL。
9. 执行一条语句以显示 `CURRENT_USER()` 函数的值。
10. 退出以 `pamuser1` 身份登录的 mysql 提示符。
11. 使用与本练习开始时相同的技术，创建一个 Linux 用户 `pamuser2` 并设置口令 `oracle2`。
12. 在 `root` 终端提示符下使用以下命令，创建一个 Linux 组 `dba` 并向该组中添加新创建的 `pamuser2` 用户：

```
groupadd dba
usermod -G dba pamuser2
```

13. 创建一个可从本地主机登录的 `mysql` 用户 `world_admin`。为该用户指定一个加密的口令。向该新用户授予 `world_innodb` 数据库的所有特权。

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

14. 使用 PAM 服务 `mysql-dba-course` 和映射 `dba=world_admin`，创建一个默认的代理帐户，该帐户的用户名为空，主机名经过 PAM 验证确认。
15. 向该默认代理帐户授予之前创建的 `world_admin@localhost` 帐户的 `PROXY` 特权。
16. 在一个新终端窗口中，尝试使用 `cleartext` 插件和口令 `oracle2` 以 `pamuser2` 身份登录到 MySQL。
17. 执行一条语句以显示 `CURRENT_USER()` 函数的值。
18. 退出所有 `mysql` 提示符。

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

练习解答 11-4：使用 PAM 验证插件

任务

注：本练习需要使用多个终端窗口，用于多个 mysql 客户机会话。

1. 在终端窗口（以 root 身份登录）中执行以下命令以创建 Linux 用户 pamuser1 并设置口令 oracle1。

   在终端窗口中输入以下内容，得到的结果如下所示：

```
$ su -
Password: oracle
# useradd pamuser1
# passwd pamuser1
Changing password for user pamuser1.
New password: oracle1
BAD PASSWORD: it is based on a dictionary word
Retype new password: oracle1
passwd: all authentication tokens updated successfully.
```

2. 使用文本编辑器在 /etc/pam.d/ 目录中创建一个文件 mysql-dba-course，其中包含以下文本：

```
#%PAM-1.0
auth            include         password-auth
account         include         password-auth
```

   在上一步使用的终端窗口中输入以下内容，并将以上内容添加到该文件中，然后将该文件另存为 /etc/pam.d/mysql-dba-course：

```
# gedit /etc/pam.d/mysql-dba-course
```

   注：可使用任何您习惯使用的文本编辑器，但请确保以 root 身份运行编辑器，以便将文件保存到正确的位置。

3. 使文件 /etc/shadow 可组读取，并将其组设置为 mysql，从而使 MySQL 服务器进程可以使用 PAM 进行验证。

   在上一步使用的终端窗口中输入以下内容，得到的结果如下所示：

```
# chmod 440 /etc/shadow
# chgrp mysql /etc/shadow
```

4. 安装 authentication_pam 插件。

   在 mysql 提示符下输入以下内容，得到的结果如下所示：

```
mysql> INSTALL PLUGIN authentication_pam
    -> SONAME 'authentication_pam.so';
Query OK, 0 rows affected (0.00 sec)
```

   注：要让插件在服务器重启后保持装入状态，可在选项文件中装入插件。

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

5. 显示所有插件以验证是否启用了 PAM 验证插件。

   在 mysql 提示符下输入以下内容，得到的结果如下所示：

```
mysql> SHOW PLUGINS;
+--------------------------+----------+--------------------+----------------------+-------------+
| Name                     | Status   | Type               | Library              | License     |
+--------------------------+----------+--------------------+----------------------+-------------+
| binlog                   | ACTIVE   | STORAGE ENGINE     | NULL                 | PROPRIETARY |
| mysql_native_password    | ACTIVE   | AUTHENTICATION     | NULL                 | PROPRIETARY |
...
| daemon_memcached         | ACTIVE   | DAEMON             | libmemcached.so      | PROPRIETARY |
| authentication_pam       | ACTIVE   | AUTHENTICATION     | authentication_pam.so | PROPRIETARY |
+--------------------------+----------+--------------------+----------------------+-------------+
45 rows in set (0.00 sec)
```

6. 创建一个 MySQL 用户 pamuser1，该用户可以从本地主机登录。使用 PAM 服务 mysql-dba-course，确保使用 PAM 验证确认了该用户。

   在 mysql 提示符下输入以下内容，得到的结果如下所示：

```
mysql> CREATE USER pamuser1@localhost
    -> IDENTIFIED WITH authentication_pam AS 'mysql-dba-course';
Query OK, 0 rows affected (0.00 sec)
```

   请注意，您尚未为此用户提供 MySQL 口令。另请注意，必须将包含特殊字符的名称放在引号中，但是，如果用户或主机的名称中没有特殊字符，则不需要加引号，如 pamuser 和 localhost。

7. 在 world_innodb.City 表上将 SELECT 特权授予新创建的 pamuser1 用户。

   在 mysql 提示符下输入以下内容，得到的结果如下所示：

```
mysql> GRANT SELECT ON world_innodb.City TO pamuser1@localhost;
Query OK, 0 rows affected (0.00 sec)
```

8. 在一个新终端窗口中，尝试使用 cleartext 插件和口令 oracle1 以 pamuser1 身份登录到 MySQL。

   在一个新的终端窗口中输入以下内容，得到的结果如下所示：

```
$ mysql --enable-cleartext-plugin -upamuser1 -p
Enter password: oracle1
Welcome to the MySQL monitor.  Commands end with ; or \g.
...
```

9. 执行一条语句以显示 CURRENT_USER() 函数的值。

   在上一步启动的 mysql 提示符下输入以下内容，得到的结果如下所示：

```
mysql> SELECT CURRENT_USER();
+--------------------+
| CURRENT_USER()     |
+--------------------+
| pamuser1@localhost |
+--------------------+
1 row in set (0.00 sec)
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

10. 退出以 pamuser1 身份登录的 mysql 提示符。

    在上一步使用的 mysql 提示符下输入以下内容，得到的结果如下所示：

```
mysql> EXIT
Bye
```

11. 使用与本练习开始时相同的技术，创建一个 Linux 用户 pamuser2 并设置口令 oracle2。

    在先前使用的 root 终端窗口中输入以下内容，得到的结果如下所示：

```
# useradd pamuser2
# passwd pamuser2
Changing password for user pamuser2.
New password: oracle2
BAD PASSWORD: it is based on a dictionary word
Retype new password: oracle2
passwd: all authentication tokens updated successfully.
```

12. 在 root 终端提示符下使用以下命令，创建一个 Linux 组 dba 并向该组中添加新创建的 pamuser2 用户：

    在上一步使用的 root 终端窗口中输入以下内容，得到的结果如下所示：

```
# groupadd dba
# usermod -G dba pamuser2
```

13. 创建一个可从本地主机登录的 mysql 用户 world_admin。为该用户指定一个加密的口令。向该新用户授予 world_innodb 数据库的所有特权。

    在 mysql 提示符（以 root 身份登录）下输入以下内容，得到的结果如下所示：

```
mysql> CREATE USER world_admin@localhost IDENTIFIED BY 'u=aX;yö#Qq';
Query OK, 0 rows affected (0.00 sec)
mysql> GRANT ALL PRIVILEGES ON world_innodb.*
    -> TO world_admin@localhost;
Query OK, 0 rows affected (0.00 sec)
```

    该帐户不是直接使用的，因此要尽可能使用加密的口令。

14. 使用 PAM 服务 mysql-dba-course 和映射 dba=world_admin，创建一个默认的代理帐户，该帐户的用户名为空，主机名经过 PAM 验证确认。

    在 mysql 提示符下输入以下内容，得到的结果如下所示：

```
mysql> CREATE USER ''@''
    ->   IDENTIFIED WITH authentication_pam
    -> AS 'mysql-dba-course, dba=world_admin';
Query OK, 0 rows affected (0.00 sec)
```

15. 向该默认代理帐户授予之前创建的 world_admin@localhost 帐户的 PROXY 特权。

```
mysql> GRANT PROXY ON world_admin@localhost TO ''@'';
Query OK, 0 rows affected (0.00 sec)
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

16. 在一个新终端窗口中，尝试使用 cleartext 插件和口令 oracle2 以 pamuser2 身份登录到 MySQL。

    在一个新的终端窗口中输入以下内容，得到的结果如下所示：

```
$ mysql --enable-cleartext-plugin -upamuser2 -p
Enter password: oracle2
Welcome to the MySQL monitor.  Commands end with ; or \g.
...
```

17. 执行一条语句以显示 CURRENT_USER() 函数的值。

    在上一步启动的 mysql 提示符下输入以下内容，得到的结果如下所示：

```
mysql> SELECT CURRENT_USER();
+-----------------------+
| CURRENT_USER()        |
+-----------------------+
| world_admin@localhost |
+-----------------------+
1 row in set (0.00 sec)
```

    以 pamuser2 身份登录后，已经使用 PAM 和 Linux 口令对您进行了验证，并且已将您映射到使用默认代理帐户的 world_admin MySQL 用户。

18. 退出所有 mysql 提示符。

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

练习 11-5：附加练习

概览

在本练习中，您将使用本课中介绍的信息，通过使用 mysql 客户机和 world_innodb 数据库，实施用户帐户和特权。

持续时间

完成本练习大约需要 15 分钟。

任务

注：本练习需要使用多个终端窗口，用于多个 mysql 客户机会话。

1. 使用 root 用户帐户登录到 mysql 客户机。验证服务器上是否存在任何匿名帐户。如果存在，则立即将其删除。
2. 创建一个只能进行本地连接的新用户帐户 Stefan，并设置口令 weak。确认新用户现在已包括在 mysql 数据库中。
3. 向 Stefan 提供所有 world_innodb 数据库特权。确认经过更新的特权。
4. 再打开一个终端窗口，并以 Stefan 身份登录到 mysql。验证新帐户有效。
5. Stefan 帐户的口令是 weak。设置一个新口令 new_pass。注销此 Stefan 会话。
6. 使用以 root 身份登录的 mysql 客户机会话，撤消之前授予 Stefan 帐户的所有 City 表特权。
7. 允许 Stefan 对 City 表中的所有列执行 SELECT 操作。注销此 root 会话。确认经过更新的特权。
8. 使用第二个终端窗口（先前为用户 Stefan 打开的窗口），再次以 Stefan 身份登录，并通过对每个 world_innodb 表执行 SELECT 验证其仍然有效。注销此 Stefan 会话。
9. 从前一个 root 会话窗口创建一个新用户 UserGroup4_01，该用户拥有与 Stefan 相同的特权，并将口令设置为 0004nq2。确认该新用户存在。
10. 删除新用户 UserGroup4_01。通过尝试显示对此用户的授权，确认该用户不再存在。
11. 在不同的终端窗口中启动两个单独的查询，然后在其运行时使用第三个窗口显示相应进程。

    注：为了顺利进行本练习，必须严格遵循以下说明的顺序和时间。仔细阅读整套说明，然后快速连续地执行这些说明。

    a. 从上一个 root 会话窗口将提示符更改为 t1 以将其与其他两个窗口区分出来。执行仅显示语句本身的初始 SHOW PROCESSLIST 语句。

    b. 再打开一个终端窗口。使用新创建的用户 Stefan（用户名 Stefan，口令 new_pass）启动 mysql 客户机。将此 mysql 会话提示符更改为 t2。将数据库更改为使用 world_innodb 数据库。

    c. 打开第三个终端窗口。使用 admin 登录路径启动 mysql 客户机。将此 mysql 会话提示符更改为 t3。将数据库更改为使用 world_innodb 数据库。

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

12. 从单独的会话窗口启动查询。

a. 在 t1 会话中，执行以下 SELECT 语句：

```
t1> SELECT SLEEP(60);
```

b. 在 t2 会话中，发送以下 SELECT 语句：

```
t2> SELECT * FROM City, Country, CountryLanguage
-> LIMIT 10000000;
```

c. 在 t3 会话中，在几秒时间后重新发送 SHOW PROCESSLIST 语句，并记下两个新进程的存在情况。

d. 在 t3 会话中，再过几秒后再次重新发送 SHOW PROCESSLIST（使用 \G 终结符以方便读取结果），并记下 Time 列中的差异，这表示进程已经运行的秒数。

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

练习解答 11-5：附加练习

任务

注：本练习需要使用多个终端窗口，用于多个 mysql 客户机会话。

1. 使用 root 用户帐户登录到 mysql 客户机。验证服务器上是否存在任何匿名帐户。如果存在，则立即将其删除。

   在终端窗口中输入以下内容，得到的结果如下所示：

```
$ mysql --login-path=admin
...

mysql> SELECT user, host, password FROM mysql.user
    -> WHERE user='';
+------+------+----------+
| user | host | password |
+------+------+----------+
|      |      |          |
+------+------+----------+
1 row in set (0.00 sec)
mysql> DROP USER ''@'';
Query OK, 0 rows affected (0.00 sec)
```

2. 创建一个只能进行本地连接的新用户帐户 Stefan，并设置口令 weak。

   a. 在 mysql 提示符下输入以下内容，得到的结果如下所示：

```
mysql> CREATE USER 'Stefan'@'localhost'
    -> IDENTIFIED BY 'weak';
Query OK, 0 rows affected (0.00 sec)
```

   b. 确认新用户现在已包括在 mysql 数据库中：

```
mysql> SELECT user, host, password FROM mysql.user
    -> WHERE user = 'Stefan';
+--------+-----------+-------------------------------------------+
| user   | host      | password                                  |
+--------+-----------+-------------------------------------------+
| Stefan | localhost | *2924FCAB604BD710DD8B0B223B67ABEBF826D10D |
+--------+-----------+-------------------------------------------+
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

3. 向 Stefan 提供所有 world_innodb 数据库特权。

   a. 在 mysql 提示符下输入以下内容，得到的结果如下所示：

```
mysql> GRANT ALL ON world_innodb.* TO 'Stefan'@'localhost';
Query OK, 0 rows affected (0.00 sec)
```

   b. 确认经过更新的特权：

```
mysql> SHOW GRANTS FOR 'Stefan'@'localhost'\G
*************************** 1. row ***************************
Grants for Stefan@localhost: GRANT USAGE ON *.* TO
'Stefan'@'localhost' IDENTIFIED BY PASSWORD
'*2924FCAB604BD710DD8B0B223B67ABEBF826D10D'
*************************** 2. row ***************************
Grants for Stefan@localhost: GRANT ALL PRIVILEGES ON `world_innodb`.*
TO 'Stefan'@'localhost'
2 rows in set (0.00 sec)
```

4. 再打开一个终端窗口，并以 Stefan 身份登录到 mysql。

   在 mysql 提示符下输入以下内容，得到的结果如下所示：

```
$ mysql -uStefan  -pweak
Welcome to the MySQL monitor.  Commands end with ; or \g.
...
mysql>
```

5. Stefan 帐户的口令是 weak。设置一个新口令 new_pass。注销此 Stefan 会话。

   a. 在 mysql 提示符下输入以下内容，得到的结果如下所示：

```
mysql> SET PASSWORD FOR 'Stefan'@'localhost' = PASSWORD('new_pass');
Query OK, 0 rows affected (0.00 sec)

mysql> EXIT
Bye
```

   b. 验证新口令有效：

```
$ mysql -uStefan -pnew_pass
Welcome to the MySQL monitor.  Commands end with ; or \g.
...
mysql> EXIT
Bye
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

6. 使用以 root 身份登录的 mysql 客户机会话，撤消之前授予 Stefan 帐户的所有 City 表特权。

   注：必须拥有授予的特权以及 GRANT OPTION 特权才能授予或撤消特权。这就是您使用 root 帐户的原因。

   a. 在终端窗口中输入以下内容，得到的结果如下所示：

```
$ mysql -uroot -p
Enter password: oracle
...

mysql> REVOKE ALL ON world_innodb.* FROM 'Stefan'@'localhost';
Query OK, 0 rows affected (0.00 sec)

mysql> GRANT ALL ON world_innodb.Country TO 'Stefan'@'localhost';
Query OK, 0 rows affected (0.00 sec)

mysql> GRANT ALL ON world_innodb.CountryLanguage
    -> TO 'Stefan'@'localhost';
Query OK, 0 rows affected (0.00 sec)
```

   b. 确认经过更新的特权：

```
mysql> SHOW GRANTS FOR 'Stefan'@'localhost'\G
*************************** 1. row ***************************
Grants for Stefan@localhost: GRANT USAGE ON *.* TO
'Stefan'@'localhost' IDENTIFIED BY PASSWORD
'*B6408F4D32E8BEC631EF224B6F743F3340E6E744'
*************************** 2. row ***************************
Grants for Stefan@localhost: GRANT ALL PRIVILEGES ON
`world_innodb`.`CountryLanguage` TO 'Stefan'@'localhost'
*************************** 3. row ***************************
Grants for Stefan@localhost: GRANT ALL PRIVILEGES ON
`world_innodb`.`Country` TO 'Stefan'@'localhost'
3 rows in set (0.00 sec)
```

   该用户拥有 City 表以外的所有 world_innodb 表的特权。

7. 允许 Stefan 对 City 表中的所有列执行 SELECT 操作。注销此 root 会话。

   a. 在 mysql 提示符下输入以下内容，得到的结果如下所示：

```
mysql> GRANT SELECT (Name, Population) ON world_innodb.City
    -> TO 'Stefan'@'localhost';
Query OK, 0 rows affected (0.00 sec)
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

b. 确认经过更新的特权：

```
mysql> SHOW GRANTS FOR 'Stefan'@'localhost'\G
*************************** 1. row ***************************
Grants for Stefan@localhost: GRANT USAGE ON *.* TO
'Stefan'@'localhost' IDENTIFIED BY PASSWORD
'*B6408F4D32E8BEC631EF224B6F743F3340E6E744'
*************************** 2. row ***************************
Grants for Stefan@localhost: GRANT ALL PRIVILEGES ON
`world_innodb`.`CountryLanguage` TO 'Stefan'@'localhost'
*************************** 3. row ***************************
Grants for Stefan@localhost: GRANT SELECT (Population, Name) ON
`world_innodb`.`City` TO 'Stefan'@'localhost'
*************************** 4. row ***************************
Grants for Stefan@localhost: GRANT ALL PRIVILEGES ON
`world_innodb`.`Country` TO 'Stefan'@'localhost'
4 rows in set (0.00 sec)

mysql> EXIT
Bye
```

该用户现在拥有所有 world_innodb 表（包括 City 表的 Name 和 Population 列）的特权。

8. 使用第二个终端窗口（先前为用户 Stefan 打开的窗口），再次以 Stefan 身份登录，并通过对每个 world_innodb 表执行 SELECT 验证其仍然有效。

a. 在 mysql 提示符下输入以下内容，得到的结果如下所示：

```
$ mysql -uStefan -pnew_pass
Welcome to the MySQL monitor.  Commands end with ; or \g.
...
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

b. 测试对 Country 表的访问权限：

```
mysql> USE world_innodb
...
Database changed
mysql> SELECT * from Country\G
...
*************************** 239. row ***************************
          Code: ZWE
          Name: Zimbabwe
     Continent: Africa
        Region: Eastern Africa
   SurfaceArea: 390757.00
     IndepYear: 1980
    Population: 11669000
LifeExpectancy: 37.8
           GNP: 5951.00
        GNPOld: 8670.00
     LocalName: Zimbabwe
GovernmentForm: Republic
   HeadOfState: Robert G. Mugabe
       Capital: 4068
         Code2: ZW
239 rows in set (0.00 sec)
```

c. 测试对 CountryLanguage 表的访问权限：

```
mysql> SELECT * from CountryLanguage;
...
| ZMB         | Bemba                     | F          |        29.7 |
| ZMB         | Chewa                     | F          |         5.7 |
| ZMB         | Lozi                      | F          |         6.4 |
| ZMB         | Nsenga                    | F          |         4.3 |
| ZMB         | Nyanja                    | F          |         7.8 |
| ZMB         | Tongan                    | F          |        11.0 |
| ZWE         | English                   | T          |         2.2 |
| ZWE         | Ndebele                   | F          |        16.2 |
| ZWE         | Nyanja                    | F          |         2.2 |
| ZWE         | Shona                     | F          |        72.1 |
+-------------+---------------------------+------------+-------------+
984 rows in set (0.01 sec)
```

d. 测试对 City 表的访问权限：

```
mysql> SELECT * FROM City;
ERROR 1143 (42000): SELECT command denied to user 'Stefan'@'localhost'
for column 'ID' in table 'city'
mysql> SELECT Name FROM City;
...
| Nablus                           |
| Rafah                            |
| New City                         |
+----------------------------------+
4079 rows in set (0.00 sec)
```

请注意，使用授予的特权，不能选择 City 的所有列 (*)。

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

e. 注销此 Stefan 会话：

```
mysql> EXIT
Bye
```

9. 从前一个 root 会话窗口创建一个新用户 UserGroup4_01，该用户拥有与 Stefan 相同的特权，并将口令设置为 0004nq2。

a. 在 mysql 提示符（以 root 身份登录）下输入以下内容，得到的结果如下所示：

```
mysql> CREATE USER 'UserGroup4_01'@'localhost'
    -> IDENTIFIED BY '0004nq2';
Query OK, 0 rows affected (0.00 sec)

mysql> GRANT ALL ON world_innodb.Country
    -> TO 'UserGroup4_01'@'localhost';
Query OK, 0 rows affected (0.00 sec)

mysql> GRANT ALL ON world_innodb.CountryLanguage
    -> TO 'UserGroup4_01'@'localhost';
Query OK, 0 rows affected (0.00 sec)

mysql> GRANT SELECT ON world_innodb.City
    -> TO 'UserGroup4_01'@'localhost';
Query OK, 0 rows affected (0.00 sec)
```

b. 确认该新用户存在：

```
mysql> SHOW GRANTS FOR 'UserGroup4_01'@'localhost'\G
*************************** 1. row ***************************
Grants for UserGroup4_01@localhost: GRANT USAGE ON *.* TO
'UserGroup4_01'@'localhost' IDENTIFIED BY PASSWORD
'*926ED64069CED09E0AAAED7677CAE9C44CF5C469'
*************************** 2. row ***************************
Grants for UserGroup4_01@localhost: GRANT ALL PRIVILEGES ON
`world_innodb`.`Country` TO 'UserGroup4_01'@'localhost'
*************************** 3. row ***************************
Grants for UserGroup4_01@localhost: GRANT SELECT ON
`world_innodb`.`City` TO 'UserGroup4_01'@'localhost'
*************************** 4. row ***************************
Grants for UserGroup4_01@localhost: GRANT ALL PRIVILEGES ON
`world_innodb`.`CountryLanguage` TO 'UserGroup4_01'@'localhost'
4 rows in set (0.00 sec)
```

10. 删除新用户 UserGroup4_01。

a. 在 mysql 提示符下输入以下内容，得到的结果如下所示：

```
mysql> DROP USER 'UserGroup4_01'@'localhost';
Query OK, 0 rows affected (0.00 sec)
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

b. 通过尝试显示对此用户的授权，确认该用户不再存在：

```
mysql> SHOW GRANTS FOR 'UserGroup4_01'@'localhost'\G
ERROR 1141 (42000): There is no such grant defined for user
'UserGroup4_01' on host 'localhost'
```

11. 在不同的终端窗口中启动两个单独的查询，然后在其运行时使用第三个窗口显示相应进程。

注：为了顺利进行本练习，必须严格遵循以下说明的顺序和时间。仔细阅读整套说明，然后快速连续地执行这些说明。

a. 从上一个 root 会话窗口将提示符更改为 t1 以将其与其他两个窗口区分出来。执行仅显示语句本身的初始 SHOW PROCESSLIST 语句。

```
mysql> PROMPT t1> ;
PROMPT set to 't1> '
t1> SHOW PROCESSLIST;
+----+------+-----------+------+---------+------+-------+------------------+
| Id | User | Host      | db   | Command | Time | State | Info             |
+----+------+-----------+------+---------+------+-------+------------------+
|  4 | root | localhost | NULL | Query   |    0 | init  | SHOW PROCESSLIST |
+----+------+-----------+------+---------+------+-------+------------------+
1 row in set (0.00 sec)
```

b. 再打开一个终端窗口。使用新创建的用户 Stefan（用户名 Stefan，口令 new_pass）启动 mysql 客户机。将此 mysql 会话提示符更改为 t2。将数据库更改为使用 world_innodb 数据库。

```
$ mysql -uStefan -pnew_pass
...
mysql> PROMPT t2> ;
PROMPT set to 't2> '
t2> USE world_innodb;
...
Database changed
```

c. 打开第三个终端窗口。使用 admin 登录路径启动 mysql 客户机。将此 mysql 会话提示符更改为 t3。将数据库更改为使用 world_innodb 数据库。输入以下内容，得到的结果如下所示：

```
$ mysql --login-path=admin
...
mysql> PROMPT t3> ;
PROMPT set to 't3> '
t3> USE test;
...
Database changed
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

12. 从单独的会话窗口启动查询。输入以下内容，得到的结果如下所示：

a. 在 t1 会话中，执行 SELECT SLEEP(60); 语句：

```
t1> SELECT SLEEP(60);
（在处理期间暂停一分钟）
+------------+
| SLEEP(60)  |
+------------+
|          0 |
+------------+
1 row in set (1 min 0.08 sec)
```

b. 在 t2 会话中，执行以下语句：

```
t2> SELECT Code FROM City, Country, CountryLanguage
    -> LIMIT 10000000;
（在处理期间输出暂停大约 40 秒）
...
10000000 rows in set (41.91 sec)
```

c. 在 t3 会话中，在几秒时间后重新发送 SHOW PROCESSLIST 语句，并记下两个新进程的存在情况：

```
t3> SHOW PROCESSLIST;
+----+--------+-----------+--------------+---------+------+--------------+---------------+
| Id | User   | Host      | db           | Command | Time | State        | Info          |
+----+--------+-----------+--------------+---------+------+--------------+---------------+
|  4 | root   | localhost | NULL         | Query   |    4 | User sleep   | SELECT SLEEP… |
| 10 | Stefan | localhost | world_innodb | Query   |    2 | Sending data | SELECT Code F…|
| 12 | root   | localhost | test         | Query   |    0 | init         | SHOW PROCESSL…|
+----+--------+-----------+--------------+---------+------+--------------+---------------+
3 rows in set (0.00 sec)
```

为了适合页面，前面的输出已截断。请注意，Time 列值取决于执行此语句的时间。

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

d. 在 t3 会话中，再过几秒后再次重新发送 SHOW PROCESSLIST（使用 **\G** 终结符以方便读取结果），并记下 Time 列中的差异，这表示进程已经运行的秒数：

```
t3> SHOW PROCESSLIST\G
*************************** 1. row ***************************
     Id: 8
   User: root
   Host: localhost
     db: NULL
Command: Query
   Time: 22
  State: User sleep
   Info: SELECT SLEEP(60)
*************************** 2. row ***************************
     Id: 10
   User: Stefan
   Host: localhost
     db: world_innodb
Command: Sleep
   Time: 20
  State:
   Info: NULL
*************************** 3. row ***************************
     Id: 12
   User: root
   Host: localhost
     db: test
Command: Query
   Time: 0
  State: init
   Info: SHOW PROCESSLIST
3 rows in set (0.00 sec)
```

现在，您可以退出每个终端 mysql 客户机，并关闭窗口。不要删除 Stefan 用户帐户，因为以后的练习中还要使用该帐户。

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

第 12 课的练习：安全

第 12 章

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

第 12 课的练习：概览

练习概览

通过这些练习，可测试您对 MySQL 安全知识的掌握程度。上机练习是针对 Oracle Linux 操作系统环境编写的，Oracle 课堂中将提供此环境。对于非 Oracle 课堂环境，可能需要对文件位置进行一些调整。

假设

- 在进行这些练习之前已经完成了 MySQL 服务器安装。
- 在终端窗口中以 `root` 用户身份登录。
- 您可以通过命令行提示符访问 `mysql` 客户机。

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

练习 12-1：测验 – MySQL 安全

概览

在本测验中，您将回答有关 MySQL 安全的问题。

持续时间

完成本练习大约需要八分钟。

测验问题

请从所提供的每道多选题或正/误判断题中选择最佳答案。

1. `root` 帐户拥有可执行任何数据库操作的完全权限，因此不应该向其授予对 `mysql` 数据库中的 `user` 表的访问权限。
    a. 正确
    b. 错误
2. 以下是 MySQL 服务器和数据安全风险的类型。
    a. 窃听
    b. 更改
    c. 播放
    d. 拒绝服务
    e. 以上都是
3. 对于所有连接、查询和其他操作，MySQL 使用基于 _____ 的安全。
    a. 原子控制线
    b. 访问控制列表
    c. 数据访问安全
4. 最常见的 MySQL 安装安全风险属于以下类别：
    a. 网络、操作系统和文件系统
    b. 网络、安全连接和用户
    c. 用户和数据
5. 通过使用单引号引住数值和字符串值，可保护数据免受未经授权的访问。
    a. 正确
    b. 错误
6. 只要用户拥有口令，未加密连接便可以满足在网络上安全传输数据的要求。
    a. 正确
    b. 错误

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

7. MySQL 支持在 MySQL 客户机和服务器之间使用 _____ 连接加密数据。
    a. 仅 SSL
    b. SSL 和 SSH
    c. `--compress`
8. 所有 MySQL 安装都支持 SSL 连接。
    a. 正确
    b. 错误

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

练习解答 12-1：测验 — MySQL 安全

测验解答

1. **b**. 错误。请勿向任何人（MySQL `root` 帐户除外）授予对 `mysql` 数据库中的 `user` 表的访问权限！
2. **e**. 以上都是。
3. **b**. 访问控制列表。
4. **a**. 网络、操作系统和文件系统。
5. **a**. 正确。
6. **b**. 错误。如果在客户机和服务器之间使用未加密的连接，则对网络拥有访问权限的用户将可以看到所有通信，并可以查看（和修改）正在发送或接收的数据。
7. **b**. SSL（安全套接字层）和 SSH（安全 shell，专用于 Windows OS）协议。
8. **b**. 错误。要使用 SSL 连接，系统必须支持 OpenSSL 或 yaSSL（随 MySQL 标准安装提供），但不是 MySQL 二进制文件的所有形式都包含必要的设置。

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

练习 12-2：确定 SSL 连接的状态

概览

在本练习中，您将使用特定 SQL 语句来确定 MySQL 服务器是否支持 SSL 连接。

持续时间

完成本练习大约需要七分钟。

任务

1. 在 `mysql` 客户机中，检查正在运行的 `mysqld` 服务器是否支持 SSL；这通过显示 `have_ssl` 系统变量的值来实现。
2. 检查当前服务器连接是否使用了 SSL；这通过检查 `Ssl_cipher` 状态变量的值来实现。然后关闭 `mysql` 客户机。

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

练习解答 12-2：确定 SSL 连接的状态

任务

1. 在 mysql 客户机中，检查 MySQL 服务器是否支持 SSL；这通过显示 have_ssl 系统变量的值来实现。在终端窗口中输入以下内容，得到的结果如下所示：

```
$ mysql --login-path=admin
Welcome to the MySQL monitor.  Commands end with ; or \g.
...

mysql> SHOW VARIABLES LIKE 'have_ssl';
+---------------+----------+
| Variable_name | Value    |
+---------------+----------+
| have_ssl      | DISABLED |
+---------------+----------+
```

   – 结果表明已编译 SSL 功能，但目前尚未启用。

2. 检查当前服务器连接是否使用了 SSL；这通过检查 Ssl_cipher 状态变量的值来实现。然后关闭 mysql 客户机。在终端窗口中（从 mysql 客户机）输入以下内容，得到的结果如下所示：

```
mysql> SHOW STATUS LIKE 'Ssl_cipher';
+---------------+-------+
| Variable_name | Value |
+---------------+-------+
| Ssl_cipher    |       |
+---------------+-------+

mysql> EXIT
Bye
```

   – 结果表明未分配任何 SSL 连接加密算法。

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

练习 12-3：附加练习 — 为 MySQL 启用 SSL 连接支持

概览

在本练习中，您将设置相应的变量以允许 MySQL 服务器和客户机使用 SSL 连接。

持续时间

完成本练习大约需要 10 分钟。

任务

1. 按照解答中的步骤，创建启用 SSL 连接所需的 OpenSSL 文件。将新文件夹命名为 `newcerts`，然后将其置于当前的 `/etc/` 目录中。
2. 确认已创建 SSL 连接所需的八个 SSL 文件：
   - `ca-cert.pem`
   - `client-cert.pem`
   - `client-req.pem`
   - `server-key.pem`
   - `ca-key.pem`
   - `client-key.pem`
   - `server-cert.pem`
   - `server-req.pem`
3. 使用 `Stefan` 用户帐户（在“用户管理”一课的练习中创建）在 `mysql` 客户机中设置 SSL。在 `root` 帐户中，向 `Stefan` 帐户授予对 `mysql` 数据库中的 `user` 表的 `SELECT` 特权，并在同一 `GRANT` 语句中要求 SSL 加密连接。
4. 使用可启用 SSL 证书颁发机构、服务器证书和服务器密钥的选项重新启动服务器。
5. 尝试以 `Stefan` 身份执行对 **`mysql`** 客户机的标准登录。结果是什么？
6. 登录到 `Stefan` 帐户，添加用于启用 SSL 连接的选项。
7. 列出 `have_ssl` 和 `Ssl_ciper` 变量的当前值。此信息能否确认 SSL 连接状态？
8. 退出 `mysql` 会话。
9. 为准备下一练习，在不使用 SSL 连接选项的情况下停止并重新启动服务器。

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

练习解答 12-3：附加练习 — 为 MySQL 启用 SSL 连接支持

任务

1. 按照解答中的步骤，创建启用 SSL 连接所需的 OpenSSL 文件。将新文件夹命名为 newcerts，然后将其置于当前的 /etc/ 目录中。在终端窗口中输入以下内容，得到的结果如下所示：

```
$ su -
Password: oracle
# cd /etc
# mkdir newcerts
# cd newcerts
# openssl genrsa 2048 > ca-key.pem
Generating RSA private key, 2048 bit long modulus
.........................+++
...........+++
e is 65537 (0x10001)
# openssl req -new -x509 -nodes -days 1000 \
        -key ca-key.pem > ca-cert.pem
You are about to be asked to enter information that will be
incorporated into your certificate request.
What you are about to enter is what is called a Distinguished Name or
a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [XX]: US
State or Province Name (full name) []: Texas
Locality Name (eg, city) [Default City]: Dallas
Organization Name (eg, company) [Default Company Ltd]:
Organizational Unit Name (eg, section) []:
Common Name (eg, your name or your server's hostname) []:
Email Address []:
# openssl req -newkey rsa:2048 -days 1000 -nodes \
        -keyout server-key.pem > server-req.pem
Generating a 2048 bit RSA private key
.............+++
..+++
writing new private key to 'server-key.pem'
-----
You are about to be asked to enter information that will be
incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or
a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [XX]:
State or Province Name (full name) []:
Locality Name (eg, city) [Default City]:
Organization Name (eg, company) [Default Company Ltd]:
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

```
Organizational Unit Name (eg, section) []:
Common Name (eg, your name or your server's hostname) []:
Email Address []:
Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:
An optional company name []:
# openssl x509 -req -in server-req.pem -days 1000 -CA ca-cert.pem \
         -CAkey ca-key.pem -set_serial 01 > server-cert.pem
Signature ok
subject=/C=XX/L=Default City/O=Default Company Ltd
Getting CA Private Key
# openssl req -newkey rsa:2048 -days 1000 -nodes \
     -keyout client-key.pem > client-req.pem
Generating a 2048 bit RSA private key
.....................................................................
.....................................+++
.....................................................................
.................................................+++
writing new private key to 'client-key.pem'
-----
You are about to be asked to enter information that will be
incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or
a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [XX]:
State or Province Name (full name) []:
Locality Name (eg, city) [Default City]:
Organization Name (eg, company) [Default Company Ltd]:
Organizational Unit Name (eg, section) []:
Common Name (eg, your name or your server's hostname) []:
Email Address []:
Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:
An optional company name []:
# openssl x509 -req -in client-req.pem -days 1000 -CA ca-cert.pem \
         -CAkey ca-key.pem -set_serial 01 > client-cert.pem
Signature ok
subject=/C=XX/L=Default City/O=Default Company Ltd
Getting CA Private Key
```

2. 确认已创建 SSL 连接所需的八个 SSL 文件。在终端窗口中（与上一步相同的 shell 提示符下）输入以下内容，得到的结果如下所示：

```
# ls
ca-cert.pem  client-cert.pem  client-req.pem   server-key.pem
ca-key.pem   client-key.pem   server-cert.pem  server-req.pem
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

3. 使用 Stefan 用户帐户（在“用户管理”一课的练习中创建）在 mysql 客户机中设置 SSL。在 root 帐户中，向 Stefan 帐户授予对 mysql 数据库中的 user 表的 SELECT 特权，并在同一 GRANT 语句中要求 SSL 加密连接。确认特权。在终端窗口中（mysql 客户机提示符下）输入以下内容，得到的结果如下所示：

```
$ mysql --login-path=admin
Welcome to the MySQL monitor.  Commands end with ; or \g.
...

mysql> GRANT SELECT ON mysql.user TO 'Stefan'@'localhost'
     > IDENTIFIED BY 'new_pass' REQUIRE SSL;
Query OK, 0 rows affected (0.00 sec)

mysql> SHOW GRANTS FOR 'Stefan'@'localhost'\G
*************************** 1. row ***************************
Grants for Stefan@localhost: GRANT USAGE ON *.* TO
'Stefan'@'localhost' IDENTIFIED BY PASSWORD
'*B6408F4D32E8BEC631EF224B6F743F3340E6E744' REQUIRE SSL
*************************** 2. row ***************************
Grants for Stefan@localhost: GRANT SELECT ON `mysql`.`user` TO
'Stefan'@'localhost'
*************************** 3. row ***************************
Grants for Stefan@localhost: GRANT ALL PRIVILEGES ON
`world_innodb`.`Country` TO 'Stefan'@'localhost'
*************************** 4. row ***************************
Grants for Stefan@localhost: GRANT ALL PRIVILEGES ON
`world_innodb`.`CountryLanguage` TO 'Stefan'@'localhost'
*************************** 5. row ***************************
Grants for Stefan@localhost: GRANT SELECT (Population, Name) ON
`world_innodb`.`City` TO 'Stefan'@'localhost'
5 rows in set (0.00 sec)

mysql> EXIT
```

4. 使用可启用 SSL 证书颁发机构、服务器证书和服务器密钥的选项重新启动服务器。在终端窗口中（以 root 身份登录的 Linux 终端提示符下）输入以下内容，得到的结果如下所示：

```
# service mysql restart --ssl-ca=/etc/newcerts/ca-cert.pem \
     --ssl-cert=/etc/newcerts/server-cert.pem \
     --ssl-key=/etc/newcerts/server-key.pem
Shutting down MySQL..                                      [  OK  ]
Starting MySQL...                                          [  OK  ]
```

5. 尝试以 Stefan 身份执行对 **mysql** 客户机的标准登录。结果是什么？

```
$ mysql -uStefan -pnew_pass
ERROR 1045 (28000): Access denied for user 'Stefan'@'localhost' (using
password: YES)
```

   – 将出现针对此用户的错误，因为现在这是一个需要 SSL 的帐户，需要使用 SSL 选项来登录该帐户。

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

6. 登录到 Stefan 帐户，添加用于启用 SSL 连接的选项。

```
$ mysql -uStefan -pnew_pass --ssl-ca=/etc/newcerts/ca-cert.pem
Welcome to the MySQL monitor.  Commands end with ; or \g.
...
```

7. 列出 have_ssl 和 Ssl_ciper 变量的当前值。此信息能否确认 SSL 连接状态？

```
mysql> SHOW VARIABLES LIKE 'have_ssl';
+---------------+-------+
| Variable_name | Value |
+---------------+-------+
| have_ssl      | YES   |
+---------------+-------+
1 row in set (0.01 sec)

mysql> SHOW STATUS LIKE 'Ssl_cipher';
+---------------+--------------------+
| Variable_name | Value              |
+---------------+--------------------+
| Ssl_cipher    | DHE-RSA-AES256-SHA |
+---------------+--------------------+
1 row in set (0.00 sec)
```

8. 退出 mysql 会话。

```
mysql> EXIT
Bye
```

9. 为准备下一练习，在不使用 SSL 连接选项的情况下停止并重新启动服务器。以 root 身份登录终端并输入以下内容。

```
# service mysql restart
Shutting down MySQL...                                     [  OK  ]
Starting MySQL.                                            [  OK  ]
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

第 13 课的练习：表维护

第 13 章

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

第 13 课的练习：概览

练习概览

通过这些练习，可测试您对表维护知识的掌握程度。上机练习是针对 Oracle Linux 操作系统环境编写的，Oracle 课堂中将提供此环境。对于非 Oracle 课堂环境，可能需要对文件位置进行一些调整。

假设

- 在进行这些练习之前已经完成了 MySQL 服务器安装。
- 在终端窗口中以 `oracle` 用户身份登录。
- 您可以通过命令行提示符访问 `mysql` 客户机。

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

练习 13-1：测验 — 表维护

概览

在本测验中，您将回答有关表维护的问题。

持续时间

本练习应该需要大约五分钟来完成。

测验问题

请从所提供的每道多选题或正/误判断题中选择最佳答案。

1. `myisamchk` 和 `mysqlcheck` 命令行客户机实用程序均可用于检查 InnoDB 表。
    a. 正确
    b. 错误
2. `myisamchk` 和 `mysqlcheck` 均直接访问表，而不与 MySQL 服务器通信。
    a. 正确
    b. 错误
3. 建议您首先使用 _____ 选项运行 `mysqlcheck`。如果需要修复，则再次运行。
    a. `--force`
    b. 无
    c. `--scan --quick`
4. 对于 InnoDB 表，可执行以下哪种维护操作？
    a. `ANALYZE TABLE`
    b. `CHECK TABLE`
    c. `CHECKSUM TABLE`
    d. `REPAIR TABLE`
    e. `OPTIMIZE TABLE`
    f. 以上都是
5. 优化表时，如果从表中删除了多行，则将回收未使用的空间。要优化表，可以使用 `OPTIMIZE TABLE` 语句或 _____ 实用程序（带有 _____ 选项）。
    a. `mysqlcheck --optimize`
    b. `myisamchk --optimize`
    c. `mysqlcheck --reclaim`
    d. 以上都不是
6. 对 MyISAM 表执行 `myisamchk` 之前，应锁定表或停止服务器，确保服务器不会访问正在进行处理的表。
    a. 正确
    b. 错误

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

7. 在系统崩溃并导致 InnoDB 无法自动恢复之后，要修复 InnoDB 表，请使用 _____ 选项重新启动服务器。
    a. `--recover`
    b. `-recover -f`
    c. `--innodb_force_recovery`

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

练习解答 13-1：测验 — 表维护

测验解答

1. b. 错误。在这两个实用程序中，只有 `mysqlcheck` 可用于检查 InnoDB 表。
2. b. 错误。`mysqlcheck` 不会直接连接到表，而是在 MySQL 服务器运行时通过它进行通信。
3. b. 无
4. a. `ANALYZE TABLE`、b. `CHECK TABLE`、c. `CHECKSUM TABLE` 和 e. `OPTIMIZE TABLE`。InnoDB 不支持 `REPAIR TABLE`。
5. a. `mysqlcheck`（带有 `--optimize` 选项）
6. a. 正确
7. c. `--innodb_force_recovery`

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

练习 13-2：使用表维护 SQL 语句

概览

在本练习中，您将使用 SQL 语句检查和优化表。

持续时间

完成本练习大约需要 15 分钟。

任务

1. 在 `world_innodb` 数据库中，新建一个名为 `City_temp` 的表，该表是 `City` 表的副本。将 `City` 表的内容插入 `City_temp` 表中：

```
CREATE TABLE City_temp LIKE City;
INSERT INTO City_temp SELECT * FROM City;
```

   - 为了日后进行比较，请使用 `SHOW TABLE STATUS` 语句显示当前的表状态。

2. 检查 `City_temp` 表是否存在问题。该表当前是否存在问题？
3. 通过使用 `ID` 列删除 `City_temp` 表数据中的多个行来创建“洞”（由于删除或更新而产生的行间隔），如下所示：

```
DELETE FROM City_temp WHERE ID BETWEEN 1001 AND 2000;
```

   - 确认已删除行，并且 `Data_Length` 和 `Index_Length` 统计信息已更改。

4. 通过对 `City_temp` 表运行 `ANALYZE TABLE` 来更新表的统计信息。确认已进行分析。然后退出 `mysql` 客户机会话。

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

练习解答 13-2：使用表维护 SQL 语句

任务

1. 在 world_innodb 数据库中，新建一个名为 City_temp 的表，该表是 City 表的副本。将 City 表的内容插入 City_temp 表中。

   a. 在终端窗口中输入以下内容，得到的结果如下所示：

```
$ mysql --login-path=admin
Welcome to the MySQL monitor.  Commands end with ; or \g.
...

mysql> USE world_innodb
Database changed

mysql> CREATE TABLE City_temp LIKE City;
Query OK, 0 rows affected (0.10 sec)
```

   b. 输入以下语句来填充 City_temp 表，得到的结果如下所示：

```
mysql> INSERT INTO City_temp SELECT * FROM City;
Query OK, 4079 rows affected, 1 warning (2.02 sec)
Records: 4079  Duplicates: 0  Warnings: 1
```

   该警告表明，在启用二进制日志记录的情况下，您将一个表中的多个行插入另一个包含自动递增字段的表中。在本课中，您可以放心地忽略此警告。

   c. 为了日后进行比较，请使用 SHOW TABLE STATUS 语句显示当前的表状态：

```
mysql> SHOW TABLE STATUS LIKE 'City_temp' \G
*************************** 1. row ***************************
           Name: City_temp
         Engine: InnoDB
        Version: 10
     Row_format: Compact
           Rows: 4188
 Avg_row_length: 97
    Data_length: 409600
Max_data_length: 0
   Index_length: 131072
...
```

   – 请记录 Rows、Data_Length 和 Index_Length 的当前值。

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

2. 检查 City_temp 表是否存在问题。该表当前是否存在问题？

   在当前 mysql 客户机会话中输入以下内容，得到的结果如下所示：

```
mysql> CHECK TABLE City_temp\G
*************************** 1. row ***************************
   Table: world_innodb.City_temp
      Op: check
Msg_type: status
Msg_text: OK
1 row in set (0.02 sec)
```

3. 通过删除 City_temp 表数据中的多个行来创建“洞”（由于删除或更新而产生的行间隔）。确认已删除行，并且 Data_Length 和 Index_Length 统计信息已更改。

   a. 在当前 mysql 客户机会话中输入以下内容，得到的结果如下所示：

```
mysql> DELETE FROM City_temp WHERE Id BETWEEN 1001 AND 2000;
Query OK, 1000 rows affected (0.03 sec)
```

   b. 显示表状态，确认已删除行：

```
mysql> SHOW TABLE STATUS LIKE 'City_temp' \G
*************************** 1. row ***************************
           Name: City_temp
         Engine: InnoDB
        Version: 10
     Row_format: Compact
           Rows: 3109
 Avg_row_length: 131
    Data_length: 409600
Max_data_length: 0
   Index_length: 131072
...
```

   – 请注意行数的变化。尽管此数字只是 InnoDB 表的估计值（并且每次执行 SHOW TABLE STATUS 语句可能都会有所不同），但通过该数字可大概了解到行数已减少。

   – Data_Length 和 Index_Length 数值保持不变。

4. 通过对 City_temp 表运行 ANALYZE TABLE 来更新表的统计信息。确认已进行分析。然后退出 mysql 客户机会话。

   a. 在当前 mysql 客户机会话中输入以下内容，得到的结果如下所示：

```
mysql> ANALYZE TABLE City_temp;
+------------------------+---------+----------+----------+
| Table                  | Op      | Msg_type | Msg_text |
+------------------------+---------+----------+----------+
| world_innodb.City_temp | analyze | status   | OK       |
+------------------------+---------+----------+----------+
1 row in set (1.13 sec)
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

b. 显示新表的状态以确认优化：

```
mysql> SHOW TABLE STATUS LIKE 'City_temp' \G
*************************** 1. row ***************************
           Name: City_temp
         Engine: InnoDB
        Version: 10
     Row_format: Compact
           Rows: 3079
 Avg_row_length: 106
    Data_length: 327680
Max_data_length: 0
   Index_length: 98304
...
```

请注意，执行分析后，行数、行平均长度以及其他值均已变化。

c. 退出会话：

```
mysql> EXIT
Bye
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

练习 13-3：使用表维护实用程序

概览

在本练习中，您将使用 MySQL 命令行实用程序检查并修复表。

持续时间

完成本练习大约需要 15 分钟。

任务

1. 在 shell 提示符中，使用 `mysqlcheck` 客户机程序（使用标准的用户名 (`-u`) 和口令 (`-p`) 选项）对 `world_innodb` 数据库运行检查。
2. 运行与上一步类似的检查，但这次使用登录路径代替 `-u` 和 `-p`，然后使用 `mysqlcheck` 检查当前所有数据库。
3. 修改上一步中用于分析所有数据库中的所有表的语句。请注意每个表所提供的消息以及这些消息之间的差异。

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

练习解答 13-3：使用表维护实用程序

任务

1. 在 shell 提示符中，使用 mysqlcheck 客户机程序（使用标准的用户名 (-u) 和口令 (-p) 选项）对 world_innodb 数据库运行检查。在终端窗口中输入以下内容，得到的结果如下所示：

```
$ mysqlcheck -uroot -p --databases world_innodb
Enter password: oracle
world_innodb.City                                  OK
world_innodb.Country                               OK
world_innodb.CountryLanguage                       OK
```

   – 所有合格的表均显示 **OK** 状态。

2. 运行与上一步类似的检查，但这次使用登录路径代替 -u 和 -p，然后使用 mysqlcheck 检查当前所有数据库。

   在终端窗口中输入以下内容，得到的结果如下所示：

```
$ mysqlcheck --login-path=admin --all-databases
mysql.columns_priv                                 OK
mysql.db                                           OK
mysql.event                                        OK
...
world_innodb.City_temp                             OK
world_innodb.Country                               OK
world_innodb.CountryLanguage                       OK
```

   – 所有表均显示 **OK** 状态。

3. 修改上一步中用于分析所有数据库中的所有表的语句。请注意每个表所提供的消息以及这些消息之间的差异。

   在终端窗口中输入以下内容，得到的结果如下所示：

```
$ mysqlcheck --login-path=admin --all-databases --analyze
mysql.columns_priv                                 Table is already up to
date
mysql.db                                           Table is already up to
date
mysql.event                                        Table is already up to
date
mysql.func                                         Table is already up to
date
mysql.general_log
note     : The storage engine for the table doesn't support analyze
mysql.help_category                                Table is already up to
date
...
mysql.slave_worker_info                            OK
mysql.slow_log
note     : The storage engine for the table doesn't support analyze
mysql.tables_priv                                  Table is already up to
date
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

```
mysql.time_zone                                 Table is already up to
date
...
sakila.film_actor                               OK
sakila.film_category                            OK
sakila.film_text                                Table is already up to
date
sakila.inventory                                OK
sakila.language                                 OK
...
world_innodb.City_part                          OK
world_innodb.City_temp                          OK
world_innodb.Country                            OK
world_innodb.CountryLanguage                    OK
```

被分析的表返回 OK 状态。前一练习中分析的表（或者自上次执行 ANALYZE 后未更改的表）报告 Table is already up to date，而 CSV 表（如 mysql.general_log）则报告 The storage engine for the table doesn't support analyze。

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

第 14 课的练习：导出和输入数据

第 14 章

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

第 14 课的练习：概览

练习概览

通过这些练习，可测试您对导出和导入数据的掌握程度。上机练习是针对 Oracle Linux 操作系统环境编写的，Oracle 课堂中将提供此环境。对于非 Oracle 课堂环境，可能需要对文件位置进行一些调整。

假设

- 在进行这些练习之前已经完成了 MySQL 服务器安装。
- 在终端窗口中以 `oracle` 用户身份登录。
- 您可以通过命令行提示符访问 `mysql` 客户机。

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

练习 14-1：导出 MySQL 数据

概览

在本练习中，您将使用 SELECT ... INTO OUTFILE 语句导出数据。要实现该目标，请执行以下操作：

- 使用默认导出选项导出 world_innodb 数据库中 CountryLanguage 表的内容。
- 使用逗号分隔值输出导出 world_innodb 数据库中 CountryLanguage 表的内容。

持续时间

本练习应该需要大约五分钟来完成。

任务

1. 使用 mysql 客户机将 world_innodb 数据库中 CountryLanguage 表的内容导出到 /tmp 目录下名为 CountryLanguage.txt 的文件中。使用默认导出选项。
2. 在第二个终端窗口中，查看 /tmp/countryLanguage.txt 文件的内容。
   - 不同的列是如何隔开的？

     _____
3. 在第一个终端窗口的 mysql 客户机中，将 world_innodb 数据库中 CountryLanguage 表的内容导出到 /tmp 目录下名为 CountryLanguage.csv 的文件中。使用逗号分隔值输出。
4. 在第二个终端窗口中，查看 /tmp/countryLanguage.csv 文件的内容。
   - 不同的列是如何隔开的？

     _____

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

练习解答 14-1：导出 MySQL 数据

任务

1. 使用 mysql 客户机将 world_innodb 数据库中 CountryLanguage 表的内容导出到 /tmp 目录下名为 CountryLanguage.txt 的文件中。使用默认导出选项。

   在 Linux 终端提示符下发出以下命令，得到的结果如下所示：

```
$ mysql -uroot -poracle
Welcome to the MySQL monitor.  Commands end with ; or \g.
...

mysql> USE world_innodb;
...
Database changed
mysql> SELECT * INTO OUTFILE '/tmp/CountryLanguage.txt'
    -> FROM CountryLanguage;
Query OK, 984 rows affected (0.00 sec)
```

   上述命令在 /tmp 目录中创建了一个文件。

2. 在第二个终端窗口中，查看 /tmp/countryLanguage.txt 文件的内容。

   在 Linux 终端提示符下发出以下命令，得到的结果如下所示：

```
$ more /tmp/CountryLanguage.txt
ABW     Dutch   T       5.3
ABW     English F       9.5
ABW     Papiamento      F       76.7
ABW     Spanish F       7.4
AFG     Balochi F       0.9
AFG     Dari    T       32.1
AFG     Pashto  T       52.4
AFG     Turkmenian      F       1.9
AFG     Uzbek   F       8.8
AGO     Ambo    F       2.4
...
```

   - 按 <space> 向下翻页，然后单击 q 退出 more 窗口。
   - 不同的列是如何隔开的？列由制表符分隔。

3. 在第一个终端窗口的 mysql 客户机中，将 world_innodb 数据库中 CountryLanguage 表的内容导出到 /tmp 目录下名为 CountryLanguage.csv 的文件中。使用逗号分隔值输出。

   在 mysql 提示符下发出以下命令，得到的结果如下所示：

```
mysql> SELECT * INTO OUTFILE '/tmp/CountryLanguage.csv'
    -> FIELDS TERMINATED BY ',' ENCLOSED BY '"'
    -> LINES TERMINATED BY '\n'
    -> FROM CountryLanguage;
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

4. 在第二个终端窗口中，查看 /tmp/CountryLanguage.csv 文件的内容。

在 Linux 终端提示符下发出以下命令，得到的结果如下所示：

```
$ more /tmp/CountryLanguage.csv
"ABW","Dutch","T","5.3"
"ABW","English","F","9.5"
"ABW","Papiamento","F","76.7"
"ABW","Spanish","F","7.4"
"AFG","Balochi","F","0.9"
"AFG","Dari","T","32.1"
"AFG","Pashto","T","52.4"
"AFG","Turkmenian","F","1.9"
"AFG","Uzbek","F","8.8"
"AGO","Ambo","F","2.4"
...
```

– 不同的列是如何隔开的？列由逗号分隔。

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

练习 14-2：导入数据

概览

在本练习中，您将使用 LOAD DATA INFILE 语句导入数据。要实现该目标，请执行以下操作：

- 在 world_innodb 数据库中，基于 CountryLanguage 表创建新表。
- 将上一练习中所创建的文本文件的内容导入新表。

持续时间

本练习应该需要大约五分钟来完成。

任务

1. 使用 mysql 客户机在连接到 world_innodb 数据库时创建和原始 CountryLanguage 表结构相同的新表（名为 CountryLanguage2）。
2. 显示表列表，确认新表目前已存在。
3. 发出 LOAD DATA INFILE 语句，将文件 /tmp/CountryLanguage.txt 装入 CountryLanguage2 表中。
4. 选择新的 CountryLanguage2 表中的所有数据来确认该表目前已填充。然后退出 mysql 会话。

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

练习解答 14-2：导入数据

任务

1. 使用 mysql 客户机在连接到 world_innodb 数据库时创建和原始 CountryLanguage 表结构相同的新表（名为 CountryLanguage2）。

   在 mysql 提示符下发出以下命令：

```
mysql> CREATE TABLE CountryLanguage2 LIKE CountryLanguage;
Query OK, 0 rows affected (0.86 sec)
```

2. 显示表列表，确认新表目前已存在。

   在 mysql 提示符下发出以下命令，得到的结果如下所示：

```
mysql> SHOW TABLES;
+------------------------+
| Tables_in_world_innodb |
+------------------------+
| City                   |
| CityLanguage           |
| City_part              |
| City_temp              |
| Country                |
| CountryLanguage        |
| CountryLanguage2       |
+------------------------+
7 rows in set (0.00 sec)
```

3. 发出 LOAD DATA INFILE 语句，将文件 /tmp/CountryLanguage.txt 装入 CountryLanguage2 表中。

   在 mysql 提示符下发出以下命令，得到的结果如下所示：

```
mysql> LOAD DATA INFILE '/tmp/CountryLanguage.txt' INTO TABLE
    -> CountryLanguage2;
```

4. 选择新的 CountryLanguage2 表中的所有数据来确认该表目前已填充。然后退出 mysql 会话。

   在 mysql 提示符下发出以下命令，得到的结果如下所示：

```
mysql> SELECT * FROM CountryLanguage2;
+-------------+---------------------------+------------+------------+
| CountryCode | Language                  | IsOfficial | Percentage |
+-------------+---------------------------+------------+------------+
| ABW         | Dutch                     | T          |        5.3 |
| ABW         | English                   | F          |        9.5 |
...
| ZWE         | Ndebele                   | F          |       16.2 |
| ZWE         | Nyanja                    | F          |        2.2 |
| ZWE         | Shona                     | F          |       72.1 |
+-------------+---------------------------+------------+------------+
984 rows in set (0.00 sec)

mysql> EXIT
Bye
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

第 15 课的练习：在 MySQL 中编程

第 15 章

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

第 15 课的练习：概览

练习概览

通过这些练习，可测试您对在 MySQL 中使用编程工具的相关知识的掌握程度。上机练习是针对 Oracle Linux 操作系统环境编写的，Oracle 课堂中将提供此环境。对于非 Oracle 课堂环境，可能需要对文件位置进行一些调整。

假设

- 在进行这些练习之前已经完成了 MySQL 服务器安装。
- 在终端窗口中以 `oracle` 用户身份登录。
- 您可以通过命令行提示符访问 `mysql` 客户机。

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

练习 15-1：创建存储例程

概览

在本练习中，您将创建并执行存储过程和存储函数。要实现该目标，请执行以下操作：

- 使用 world_innodb 数据库。
- 创建一个存储过程，以显示 world_innodb 数据库中 City、Country 和 CountryLanguage 表的记录数，并执行该存储过程。
- 创建一个存储函数以通过输入两个值（基本工资和税率）来确定税后工资，并执行该存储函数。

持续时间

完成本练习大约需要 20 分钟。

任务

1. 使用 mysql 客户机在 world_innodb 数据库中输入以下 SQL 语句以创建 record_count 存储过程：

```
DELIMITER //
CREATE PROCEDURE record_count ()
BEGIN
   SELECT 'Country count ', COUNT(*) FROM Country;
   SELECT 'City count ', COUNT(*) FROM City;
   SELECT 'CountryLanguage count', COUNT(*) FROM
    CountryLanguage;
END//
DELIMITER ;
```

2. 发出适当命令以执行 record_count 存储过程。
3. 转到 test 数据库，然后尝试执行 record_count 存储过程。
   - record_count 过程为何失败？

     _____
     _____
4. 执行 record_count 存储过程，这次使用 world_innodb 数据库的名称进行限定。
   - record_count 存储过程是否正确执行？ _____

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

5. 转到 world_innodb 数据库并发出以下 SQL 语句以创建 pay_check 存储函数：

```
DELIMITER //
CREATE FUNCTION pay_check (gross_pay FLOAT(9,2),
  tax_rate FLOAT(3,2))
RETURNS FLOAT(9,2)
NO SQL
BEGIN
   DECLARE net_pay FLOAT(9,2) DEFAULT 0;
   SET net_pay=gross_pay - gross_pay * tax_rate;
   RETURN net_pay;
END//
DELIMITER ;
```

6. 执行 pay_check 存储函数，方法为发出适当的 SQL 语句并传递参数 100000 和 0.05。

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

练习解答 15-1：创建存储例程

任务

1. 使用 mysql 客户机在 world_innodb 数据库中输入以下 SQL 语句以创建 record_count 存储过程。

   在 Linux 终端中输入以下内容，得到的结果如下所示：

```
$ mysql --login-path=admin
Welcome to the MySQL monitor.  Commands end with ; or \g.
...
mysql> USE world_innodb;
...
Database changed
mysql> DELIMITER //
mysql> CREATE PROCEDURE record_count ()
    -> BEGIN
    ->   SELECT 'Country count ', COUNT(*) FROM Country;
    ->   SELECT 'City count ', COUNT(*) FROM City;
    ->   SELECT 'CountryLanguage count', COUNT(*) FROM
    ->    CountryLanguage;
    -> END//
Query OK, 0 rows affected (0.06 sec)

mysql> DELIMITER ;
```

   world_innodb 数据库现在包含名为 record_count 的存储过程。

2. 发出适当命令以执行 record_count 存储过程。

   在 mysql 提示符下输入以下语句，得到的结果如下所示：

```
mysql> CALL record_count();
+----------------+----------+
| Country count  | COUNT(*) |
+----------------+----------+
| Country count  |      239 |
+----------------+----------+
1 row in set (0.00 sec)

+-------------+----------+
| City count  | COUNT(*) |
+-------------+----------+
| City count  |     4079 |
+-------------+----------+
1 row in set (0.01 sec)

+-----------------------+----------+
| CountryLanguage count | COUNT(*) |
+-----------------------+----------+
| CountryLanguage count |      984 |
+-----------------------+----------+
1 row in set (0.01 sec)

Query OK, 0 rows affected (0.01 sec)
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

3. 转到 test 数据库，然后尝试执行 record_count 存储过程。
   在终端窗口中输入以下内容，得到的结果如下所示：

```
mysql> USE test;
...
Database changed
mysql> CALL record_count();
ERROR 1305 (42000): PROCEDURE test.record_count does not exist.
```

   - record_count 过程为何失败？
     存储过程 record_count 是 world_innodb 数据库的一部分。MySQL 在 test 数据库中查找 record_count 存储过程。

4. 执行 record_count 存储过程，这次使用 world_innodb 数据库的名称进行限定。
   在终端窗口中输入以下内容，得到的结果如下所示：

```
mysql> CALL world_innodb.record_count();
+----------------+----------+
| Country count  | COUNT(*) |
+----------------+----------+
| Country count  |      239 |
+----------------+----------+
1 row in set (0.01 sec)

+-------------+----------+
| City count  | COUNT(*) |
+-------------+----------+
| City count  |     4079 |
+-------------+----------+
1 row in set (0.01 sec)

+-----------------------+----------+
| CountryLanguage count | COUNT(*) |
+-----------------------+----------+
| CountryLanguage count |      984 |
+-----------------------+----------+
1 row in set (0.01 sec)

Query OK, 0 rows affected (0.01 sec)
```

   - record_count 存储过程是否正确执行？是

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

5. 转到 world_innodb 数据库并发出以下 SQL 语句以创建 pay_check 存储函数。
   在 mysql 提示符下输入以下语句，得到的结果如下所示：

```
mysql> USE world_innodb;
...
Database changed
mysql> DELIMITER //
mysql> CREATE FUNCTION pay_check (gross_pay FLOAT(9,2),
    ->  tax_rate FLOAT(3,2))
    -> RETURNS FLOAT(9,2)
    -> NO SQL
    -> BEGIN
    ->   DECLARE net_pay FLOAT(9,2) DEFAULT 0;
    ->   SET net_pay=gross_pay - gross_pay * tax_rate;
    ->   RETURN net_pay;
    -> END//
Query OK, 0 rows affected (0.06 sec)

mysql> DELIMITER ;
```

6. 执行 pay_check 存储函数，方法为发出适当的 SQL 语句并传递参数 100000 和 0.05。
   在 mysql 提示符下输入以下语句，得到的结果如下所示：

```
mysql> SELECT pay_check (100000, 0.05);
+--------------------------+
| pay_check (100000,0.05) |
+--------------------------+
|                 95000.00 |
+--------------------------+
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

练习 15-2：查看存储例程

概览

在本练习中，您将查看在前面的练习中创建的存储例程。要实现该目标，请执行以下操作：

- 使用 `SHOW CREATE PROCEDURE` 命令查看位于 `world_innodb` 数据库中的特定存储例程。
- 使用 `SHOW { PROCEDURE | FUNCTION } STATUS` 命令查看 MySQL 服务器中的存储例程。
- 使用 `INFORMATION_SCHEMA` 数据库查看 MySQL 服务器中的所有存储例程。

持续时间

完成本练习大约需要 10 分钟。

任务

1. 使用 `mysql` 客户机在 `world_innodb` 数据库中发出 `SHOW CREATE PROCEDURE` 命令以查看 `record_count` 存储过程的详细信息。
2. 发出 `SHOW PROCEDURE STATUS` 命令以查看 MySQL 服务器上所有名称以 `record` 开头的存储过程的状态。
3. 发出 `SHOW FUNCTION STATUS` 命令以查看 MySQL 服务器上所有存储函数的状态。
4. 使用 `INFORMATION_SCHEMA` 数据库发出 `SELECT` 语句以查看 MySQL 服务器上所有存储例程的详细信息。

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

练习解答 15-2：查看存储例程

任务

1. 使用 mysql 客户机在 world_innodb 数据库中发出 SHOW CREATE PROCEDURE 命令以查看 record_count 存储过程的详细信息。

   在连接到 world_innodb 数据库的 mysql 提示符下输入以下语句，得到的结果如下所示：

```
mysql> SHOW CREATE PROCEDURE record_count\G
*************************** 1. row ***************************
           Procedure: record_count
            sql_mode: STRICT_TRANS_TABLES,NO_ENGINE_SUBSTITUTION
    Create Procedure: CREATE DEFINER=`root`@`localhost` PROCEDURE
`record_count`()
BEGIN
  SELECT 'Country count ', COUNT(*) FROM Country;
  SELECT 'City count ', COUNT(*) FROM City;
  SELECT 'CountryLanguage count', COUNT(*) FROM
   CountryLanguage;
END
character_set_client: utf8
collation_connection: utf8_general_ci
  Database Collation: latin1_swedish_ci
1 row in set (0.00 sec)
```

2. 发出 SHOW PROCEDURE STATUS 命令以查看 MySQL 服务器上所有名称以 record 开头的存储过程的状态。

   在 mysql 提示符下输入以下语句，得到的结果如下所示：

```
mysql> SHOW PROCEDURE STATUS LIKE 'record%'\G
*************************** 1. row ***************************
                  Db: world_innodb
                Name: record_count
                Type: PROCEDURE
             Definer: root@localhost
            Modified: 2013-02-10 13:48:11
             Created: 2013-02-10 13:48:11
       Security_type: DEFINER
             Comment:
character_set_client: utf8
collation_connection: utf8_general_ci
  Database Collation: latin1_swedish_ci
1 row in set (0.00 sec)
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

3. 发出 SHOW FUNCTION STATUS 命令以查看 MySQL 服务器上所有存储函数的状态。
   在 mysql 提示符下输入以下语句，得到的结果如下所示：

```
mysql> SHOW FUNCTION STATUS\G
*************************** 1. row ***************************
                  Db: sakila
                Name: get_customer_balance
                Type: FUNCTION
             Definer: root@localhost
            Modified: 2013-02-03 15:25:01
             Created: 2013-02-03 15:25:01
       Security_type: DEFINER
             Comment:
character_set_client: utf8
collation_connection: utf8_general_ci
  Database Collation: latin1_swedish_ci
*************************** 2. row ***************************
                  Db: sakila
                Name: inventory_held_by_customer
                Type: FUNCTION
             Definer: root@localhost
            Modified: 2013-02-03 15:25:01
             Created: 2013-02-03 15:25:01
       Security_type: DEFINER
             Comment:
character_set_client: utf8
collation_connection: utf8_general_ci
  Database Collation: latin1_swedish_ci
*************************** 3. row ***************************
                  Db: sakila
                Name: inventory_in_stock
                Type: FUNCTION
             Definer: root@localhost
            Modified: 2013-02-03 15:25:01
             Created: 2013-02-03 15:25:01
       Security_type: DEFINER
             Comment:
character_set_client: utf8
collation_connection: utf8_general_ci
  Database Collation: latin1_swedish_ci
*************************** 4. row ***************************
                  Db: world_innodb
                Name: pay_check
                Type: FUNCTION
             Definer: root@localhost
            Modified: 2013-02-10 13:52:32
             Created: 2013-02-10 13:52:32
       Security_type: DEFINER
             Comment:
character_set_client: utf8
collation_connection: utf8_general_ci
  Database Collation: latin1_swedish_ci
4 rows in set (0.00 sec)
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

4. 使用 INFORMATION_SCHEMA 数据库发出 SELECT 语句以查看 MySQL 服务器上所有存储例程的详细信息。

   在 mysql 提示符下输入以下语句，得到的结果如下所示：

```
mysql> USE INFORMATION_SCHEMA;
...
Database changed
mysql> SELECT * FROM ROUTINES\G
*************************** 1. row ***************************
           SPECIFIC_NAME: film_in_stock
         ROUTINE_CATALOG: def
          ROUTINE_SCHEMA: sakila
            ROUTINE_NAME: film_in_stock
...
*************************** 7. row ***************************
           SPECIFIC_NAME: pay_check
         ROUTINE_CATALOG: def
          ROUTINE_SCHEMA: world_innodb
            ROUTINE_NAME: pay_check
            ROUTINE_TYPE: FUNCTION
               DATA_TYPE: float
CHARACTER_MAXIMUM_LENGTH: NULL
  CHARACTER_OCTET_LENGTH: NULL
       NUMERIC_PRECISION: 9
           NUMERIC_SCALE: 2
      DATETIME_PRECISION: NULL
      CHARACTER_SET_NAME: NULL
          COLLATION_NAME: NULL
          DTD_IDENTIFIER: float(9,2)
            ROUTINE_BODY: SQL
      ROUTINE_DEFINITION: BEGIN
  DECLARE net_pay FLOAT(9,2) DEFAULT 0;
  SET net_pay=gross_pay - gross_pay * tax_rate;
  RETURN net_pay;
END
           EXTERNAL_NAME: NULL
       EXTERNAL_LANGUAGE: NULL
         PARAMETER_STYLE: SQL
        IS_DETERMINISTIC: NO
         SQL_DATA_ACCESS: NO SQL
                SQL_PATH: NULL
           SECURITY_TYPE: DEFINER
                 CREATED: 2013-02-10 13:52:32
            LAST_ALTERED: 2013-02-10 13:52:32
                SQL_MODE: STRICT_TRANS_TABLES,NO_ENGINE_SUBSTITUTION
         ROUTINE_COMMENT:
                 DEFINER: root@localhost
    CHARACTER_SET_CLIENT: utf8
    COLLATION_CONNECTION: utf8_general_ci
      DATABASE_COLLATION: latin1_swedish_ci
*************************** 8. row ***************************
           SPECIFIC_NAME: record_count
         ROUTINE_CATALOG: def
          ROUTINE_SCHEMA: world_innodb
            ROUTINE_NAME: record_count
            ROUTINE_TYPE: PROCEDURE
               DATA_TYPE:
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

```
CHARACTER_MAXIMUM_LENGTH: NULL
  CHARACTER_OCTET_LENGTH: NULL
       NUMERIC_PRECISION: NULL
           NUMERIC_SCALE: NULL
      DATETIME_PRECISION: NULL
      CHARACTER_SET_NAME: NULL
          COLLATION_NAME: NULL
          DTD_IDENTIFIER: NULL
            ROUTINE_BODY: SQL
      ROUTINE_DEFINITION: BEGIN
  SELECT 'Country count ', COUNT(*) FROM Country;
  SELECT 'City count ', COUNT(*) FROM City;
  SELECT 'CountryLanguage count', COUNT(*) FROM
   CountryLanguage;
END
           EXTERNAL_NAME: NULL
       EXTERNAL_LANGUAGE: NULL
         PARAMETER_STYLE: SQL
        IS_DETERMINISTIC: NO
         SQL_DATA_ACCESS: CONTAINS SQL
                SQL_PATH: NULL
           SECURITY_TYPE: DEFINER
                 CREATED: 2013-02-10 13:48:11
            LAST_ALTERED: 2013-02-10 13:48:11
                SQL_MODE: STRICT_TRANS_TABLES,NO_ENGINE_SUBSTITUTION
         ROUTINE_COMMENT:
                 DEFINER: root@localhost
    CHARACTER_SET_CLIENT: utf8
    COLLATION_CONNECTION: utf8_general_ci
      DATABASE_COLLATION: latin1_swedish_ci
8 rows in set (0.00 sec)
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

练习 15-3：创建触发器

概览

在本练习中，您将创建一个触发器以捕获删除的数据。要实现该目标，请执行以下操作：

- 在 world_innodb 表中创建一个表以存储对 City 表中已删除数据的引用。
- 在创建的表中进行删除后，创建一个触发器以存储对已删除数据的引用。
- 从 world_innodb 表中删除来自 City 表中的数据。
- 确认所创建的表包含对已删除数据的引用。

持续时间

完成本练习大约需要 20 分钟。

任务

1. 使用 mysql 客户机在 world_innodb 数据库中发出以下 SQL 语句，以创建用于存储对 City 表中已删除记录的引用的表：

```
CREATE TABLE DeletedCity
( ID INT UNSIGNED,
  Name VARCHAR(50),
  When_Deleted timestamp);
```

2. 发出以下 SQL 语句以创建用于在从 City 表中删除记录后捕获数据的触发器，并将所捕获的数据置于 DeletedCity 表中：

```
CREATE TRIGGER City_AD AFTER DELETE ON City
FOR EACH ROW
INSERT INTO DeletedCity (ID, Name)
   VALUES (OLD.ID, OLD.Name);
```

3. 发出适当的 SHOW 命令以确定已创建触发器。
4. 删除 City 表中城市名为 Dallas 的记录。
5. 尝试选择名为 Dallas 的所有 City 表记录以确认正确执行了删除操作。
6. 查看 DeletedCity 表的内容以确定触发器是否捕获了对第 4 步中所删除记录的引用。

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

练习解答 15-3：创建触发器

任务

1. 使用 mysql 客户机在 world_innodb 数据库中发出以下 SQL 语句，以创建用于存储对 City 表中已删除记录的引用的表。

   在连接到 world_innodb 数据库的 mysql 提示符下输入以下语句，得到的结果如下所示：

```
mysql> USE world_innodb;
...
Database changed
mysql> CREATE TABLE DeletedCity
    -> (ID INT UNSIGNED,
    ->  Name VARCHAR(50),
    ->  When_Deleted timestamp);
);
Query OK, 0 rows affected (0.06 sec)
```

2. 发出以下 SQL 语句以创建用于在从 City 表中删除记录后捕获数据的触发器，并将所捕获的数据置于 DeletedCity 表中。

   在 mysql 提示符下输入以下语句，得到的结果如下所示：

```
mysql> CREATE TRIGGER City_AD AFTER DELETE ON City
    -> FOR EACH ROW
    -> INSERT INTO DeletedCity (ID, Name)
    ->  VALUES (OLD.ID, OLD.Name);
Query OK, 0 rows affected (0.06 sec)
```

3. 发出适当的 SHOW 命令以确定已创建触发器。

   在 mysql 提示符下输入以下语句，得到的结果如下所示：

```
mysql> SHOW TRIGGERS\G
*************************** 1. row ***************************
             Trigger: City_AD
               Event: DELETE
               Table: City
           Statement: INSERT INTO DeletedCity (ID, Name)
   VALUES (OLD.ID, OLD.Name)
              Timing: AFTER
             Created: NULL
            sql_mode: STRICT_TRANS_TABLES,NO_ENGINE_SUBSTITUTION
             Definer: root@localhost
character_set_client: utf8
collation_connection: utf8_general_ci
  Database Collation: latin1_swedish_ci
1 row in set (0.01 sec)
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

4. 删除 City 表中城市名为 Dallas 的记录。

   在 mysql 提示符下输入以下语句，得到的结果如下所示：

```
mysql> DELETE FROM City WHERE Name = 'Dallas';
Query OK, 1 row affected (0.13 sec)
```

5. 尝试选择名为 Dallas 的所有 City 表记录以确认正确执行了删除操作。

   在 mysql 提示符下输入以下语句，得到的结果如下所示：

```
mysql> SELECT * FROM City WHERE Name = 'Dallas';
Empty set (0.00 sec)
```

6. 查看 DeletedCity 表的内容以确定触发器是否捕获了对第 4 步中所删除记录的引用。

   在 mysql 提示符下输入以下语句，得到的结果如下所示：

```
mysql> SELECT * FROM DeletedCity;
+------+--------+---------------------+
| ID   | Name   | When_Deleted        |
+------+--------+---------------------+
| 3800 | Dallas | 2013-02-10 14:01:51 |
+------+--------+---------------------+
1 row in set (0.00 sec)
```

   已删除的记录及其删除时间将显示在结果集中。

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

练习 15-4：创建和测试事件

概览

在本练习中，您将创建一个事件以定位在 MySQL 服务器上运行时间过长的进程并终止其连接。要实现该目标，请执行以下操作：

- 创建一个事件，以便在非 `root` 用户的查询运行时间超过 30 秒后将其断开。
- 启用事件调度程序。
- 创建非 `root` 用户以测试所创建的事件。
- 测试所创建的事件。

持续时间

完成本练习大约需要 20 分钟。

注：本练习在创建事件时使用了超出本课程范围的高级编程。但是，本示例旨在显示高级编程的作用，以及了解有关本主题的更多信息如何为 MySQL 服务器的日常管理任务带来的益处。

任务

1. 使用 `mysql` 客户机在 `world_innodb` 数据库中发出以下 SQL 语句以创建一个事件，在非 `root` 用户的查询运行时间超过 30 秒时将其断开：

```
DELIMITER //
CREATE EVENT kill_user
  ON SCHEDULE EVERY 20 SECOND
DO
BEGIN
  DECLARE var_id INT;
  DECLARE procs CURSOR FOR SELECT id
  FROM INFORMATION_SCHEMA.PROCESSLIST WHERE TIME>30
    AND Command != 'Sleep' AND USER != 'root';
  OPEN procs;
  BEGIN
    DECLARE EXIT HANDLER FOR NOT FOUND
    BEGIN
      CLOSE procs;
    END;
    LOOP
      FETCH procs INTO var_id;
      KILL var_id;
    END LOOP;
  END;
END//
DELIMITER ;
```

2. 启用事件调度程序。
3. 创建名为 `tester` 的非 `root` 用户，该用户使用口令“`secret`”从 `localhost` 登录。
4. 授予新用户对所有数据库的访问权限。

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

5. 在第二个终端窗口中，以新用户身份登录到 mysql 客户机。
6. 在第二个终端窗口中，输入以下命令以创建尝试运行超过 30 秒的 SELECT 语句：

```
mysql> SELECT SLEEP(60);
```

   – 服务器终止连接用了多长时间？

   _____.

7. 在第二个终端窗口中退出 mysql 客户机。

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

练习解答 15-4：创建和测试事件

任务

1. 使用 mysql 客户机在 world_innodb 数据库中发出以下 SQL 语句以创建一个事件，在非 root 用户的查询运行时间超过 30 秒时将其断开：

```
mysql> DELIMITER //
mysql> CREATE EVENT kill_user
    -> ON SCHEDULE EVERY 20 SECOND
    -> DO
    -> BEGIN
    ->  DECLARE var_id INT;
    ->  DECLARE procs CURSOR FOR SELECT id
    ->  FROM INFORMATION_SCHEMA.PROCESSLIST WHERE TIME>30
    ->   AND Command != 'Sleep' AND USER != 'root';
    ->  OPEN procs;
    ->  BEGIN
    ->    DECLARE EXIT HANDLER FOR NOT FOUND
    ->    BEGIN
    ->      CLOSE procs;
    ->    END;
    ->    LOOP
    ->      FETCH procs INTO var_id;
    ->      KILL var_id;
    ->    END LOOP;
    ->  END;
    -> END//
Query OK, 0 rows affected (0.06 sec)

mysql> DELIMITER ;
```

2. 启用事件调度程序。

```
mysql> SET GLOBAL event_scheduler = ON;
Query OK, 0 rows affected (0.06 sec)
```

3. 创建名为 tester 的非 root 用户，该用户使用口令“secret”从 localhost 登录。

```
mysql> CREATE USER tester@localhost IDENTIFIED BY 'secret';
Query OK, 0 rows affected (0.06 sec)
```

4. 授予新用户对所有数据库的访问权限。

```
mysql> GRANT SELECT ON *.* TO tester@localhost;
Query OK, 0 rows affected (0.06 sec)
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

5. 在第二个终端窗口中，以新用户身份登录到 mysql 客户机。

```
$ mysql -utester -p
Enter password: secret
Welcome to the MySQL monitor.  Commands end with ; or \g.
...
```

6. 在第二个终端窗口中，输入以下命令以创建尝试运行超过 30 秒的 SELECT 语句：

```
mysql> SELECT SLEEP(60);
ERROR 2013 (HY000): Lost connection to MySQL server during query
```

   – 服务器终止连接用了多长时间？
   查询开始运行后大约 30 至 50 秒，服务器终止了连接。由于将此事件安排为每隔 20 秒运行一次，因此该事件会在达到 30 秒限制后的 20 秒内终止进程。

7. 在第二个终端窗口中退出 mysql 客户机：

```
mysql> EXIT
Bye
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

第 16 课的练习：MySQL 备份和恢复

第 16 章

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

第 16 课的练习：概览

练习概览

通过这些练习，可测试您对备份和恢复主题的掌握程度。上机练习是针对 Oracle Linux 操作系统环境编写的，Oracle 课堂中将提供此环境。对于非 Oracle 课堂环境，可能需要对文件位置进行一些调整。

假设

- 在进行这些练习之前已经完成了 MySQL 服务器安装。
- 在终端窗口中以 `root` 用户身份登录。
- 您可以通过命令行提示符访问 `mysql` 客户机。

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

练习 16-1：测验 — 备份简介

概览

在本练习中，您将回答有关 MySQL 备份策略的问题。

持续时间

本练习应该需要大约五分钟来完成。

测验问题

对于下面的每个选项，写下要使用的备份策略类型：**磁盘**、**逻辑/文本**或**二进制**。（每个答案可以包含一种、两种或全部三种类型的策略。）

1. 模式 — 版本控制 _____
2. 数据分析/报告 _____
3. 病毒/SQL 注入（坏磁盘） _____
4. 用户失误/错误 _____
5. 用户破坏性操作（故意的） _____
6. 电源故障 _____
7. 系统故障（磁盘、内存、网络崩溃） _____
8. 灾难（人为、自然） _____
9. 归档记录（合规性） _____
10. 测试数据/创建开发服务器 _____
11. 迁移 _____
12. 软件更新 _____
13. 复制 _____
14. 证据（用于法院/政府诉讼） _____

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

练习解答 16-1：测验 – 备份简介

测验解答

1. 磁盘、逻辑/文本

2. 逻辑/文本

3. 磁盘、逻辑/文本

4. 磁盘、逻辑/文本、二进制日志

5. 磁盘、逻辑/文本、二进制日志

6. 磁盘、逻辑/文本、二进制日志

7. 磁盘、逻辑/文本

8. 磁盘、逻辑/文本

9. 逻辑/文本

10. 磁盘

11. 逻辑/文本

12. 磁盘、逻辑/文本

13. 二进制日志

14. 逻辑/文本

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

练习 16-2：MySQL Enterprise Backup

概览

在本练习中，您将使用 mysqlbackup 应用程序创建一个备份。要实现该目标，请执行以下操作：

- 修改 my.cnf 文件。
- 在 MySQL 服务器上创建一个负载。
- 在 MySQL 服务器上有负载的情况下，执行 mysqlbackup 应用程序。
- 从 /var/lib/mysql 目录中删除文件。
- 执行备份以恢复删除的文件。

假设

- 已安装并且正在运行 MySQL 服务器。
- 已安装 world_innodb 数据库。
- 已安装 MySQL Enterprise Backup 软件。

持续时间

完成本练习大约需要 30 分钟。

任务

1. 创建 /backups 目录，并将其所有者更改为 mysql 用户。
2. 在 /etc/my.cnf 文件中，记下 datadir 服务器参数的值。
3. 设置以下 **my.cnf** 选项，如下所示：

```
[mysqld]
log-bin=mybinlog
innodb_log_files_in_group = 2
innodb_log_file_size = 256M
innodb_data_file_path=ibdata1:10M:autoextend
```

4. 添加一个 [mysqlbackup] 部分，然后添加以下设置：
   - backup-dir: **/backups/meb1**
   - user: **backupuser**
   - socket: **/var/lib/mysql/mysql.sock**
5. 保存对 /etc/my.cnf 文件的更改并关闭该文件。
6. 删除 InnoDB 日志文件。
7. 重新启动 MySQL 服务器以应用更新后的 /etc/my.cnf。
8. 再打开一个终端窗口并启动 mysql 客户机。
9. 在第二个终端窗口中，创建一个 MySQL 用户 backupuser 并设置口令 oracle，然后授予该用户使用 mysqlbackup 所需的所有权限。
10. 在第二个终端窗口中，使用 world_innodb 数据库并创建名为 City_Large 的 City 表的副本。

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

11. 在第二个终端窗口中，发出以下命令将 City 表中的所有记录插入到 City_Large 表中。

```
INSERT INTO City_Large (Name, CountryCode, District,
Population) SELECT Name, CountryCode, District,
Population FROM City;
```

12. 在第二个终端窗口中，发出以下命令复制 City_Large 表中的记录：

```
INSERT INTO City_Large (Name, CountryCode, District,
Population) SELECT Name, CountryCode, District,
Population FROM City_Large;
```

13. 在第二个终端窗口中，执行上一步中的 SQL 语句七次。
14. 在第二个终端窗口中，在运行备份之前，执行一次 FLUSH LOGS 命令。
15. 在第二个终端窗口中，再一次发出第 12 步中的命令。不要等待此步骤完成，立即开始下一步。
16. 在第一个终端窗口中，发出以下命令（同时继续执行上一步中的 mysql 命令）以 mysql 用户身份运行 mysqlbackup 应用程序。

```
# su - mysql
$ mysqlbackup --defaults-file=/etc/my.cnf -p backup-and-apply-log
```

17. 在第一个终端窗口中，列出 /backups/meb1 目录中的文件。
18. 在第一个终端窗口中，从 mysql 用户 shell 退出，返回到 root shell。然后，在 /var/lib/mysql 目录中，删除 ibdata1 文件和 world_innodb 目录。
19. 在第二个终端窗口中，显示 MySQL 服务器包含的所有数据库。
20. 在第二个终端窗口中，退出 mysql 客户机。
21. 在第一个终端窗口中，关闭 MySQL 服务器。
22. 在第一个终端窗口中，执行以下命令将创建的备份恢复到 /var/lib/mysql 目录。

```
# su - mysql
$ mysqlbackup --defaults-file=/etc/my.cnf copy-back
```

23. 在第一个终端窗口中，从 mysql shell 退出并启动 MySQL 服务器。
24. 在第二个终端窗口中，启动 mysql 客户机并显示 MySQL 服务器包含的所有数据库。
25. 在第二个终端窗口中，查看 world_innodb 数据库包含的所有表。
26. 在第二个终端窗口中，退出 mysql 客户机并关闭终端窗口。

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

练习解答 16-2：MySQL Enterprise Backup

任务

1. 创建 **/backups** 目录，并将其所有者更改为 mysql 用户。

   在 Linux 终端上输入以下内容：

```
$ su -
Password: oracle
# mkdir /backups
# chown mysql:mysql /backups
```

2. 在 /etc/my.cnf 文件中，记下 datadir 服务器参数的值。
   - 查找标题 [mysqld] 并记下 datadir 参数的值：

```
[mysqld]
...
datadir = /var/lib/mysql
...
```

   - **注：**要执行冷备份，datadir 参数必须在文件中。目录路径必须是绝对路径。对于热备份或温备份，mysqlbackup 将从服务器查询该值；它不会假定任何文件位置默认值。

3. 设置以下 my.cnf 选项，如下所示。
   - 查找标题 [mysqld]，修改 log-bin 的值，并添加剩余行，如下所示：

```
[mysqld]
...
log-bin=mybinlog
innodb_log_files_in_group = 2
innodb_log_file_size = 256M
innodb_data_file_path=ibdata1:12M:autoextend
...
```

4. 添加一个 **[mysqlbackup]** 部分，然后添加以下设置：
   - backup-dir: **/backups/meb1**
   - user: **backupuser**
   - socket: **/var/lib/mysql/mysql.sock**
     - 在 **/etc/my.cnf** 末尾添加以下行：

```
...
[mysqlbackup]
backup-dir=/backups/meb1
user=backupuser
socket=/var/lib/mysql/mysql.sock
```

5. 保存对 /etc/my.cnf 文件的更改并关闭该文件。
6. 删除 InnoDB 日志文件。

   在终端窗口中输入以下内容：

```
# rm -f /var/lib/mysql/ib_logfile*
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

7. 重新启动 MySQL 服务器以应用更新后的 /etc/my.cnf。

   在终端窗口中输入以下内容，得到的结果如下所示：

```
# service mysql restart
Shutting down MySQL...                                     [  OK  ]
Starting MySQL.                                            [  OK  ]
```

8. 再打开一个终端窗口并启动 mysql 客户机。
   得到的结果如下所示：

```
$ mysql --login-path=admin
Welcome to the MySQL monitor.  Commands end with ; or \g.
...
```

9. 在第二个终端窗口中，创建一个 MySQL 用户 backupuser 并设置口令 oracle，然后授予该用户使用 mysqlbackup 所需的所有权限。

   在终端窗口中输入以下内容，得到的结果如下所示：

```
mysql> CREATE USER 'backupuser'@'localhost' IDENTIFIED BY 'oracle';
Query OK, 0 rows affected (0.00 sec)

mysql> GRANT RELOAD ON *.* TO 'backupuser'@'localhost';
Query OK, 0 rows affected (0.00 sec)

mysql> GRANT CREATE, INSERT, DROP ON mysql.ibbackup_binlog_marker
     > TO 'backupuser'@'localhost';
Query OK, 0 rows affected (0.00 sec)

mysql> GRANT CREATE, INSERT, DROP ON mysql.backup_progress
     > TO 'backupuser'@'localhost';
Query OK, 0 rows affected (0.00 sec)

mysql> GRANT CREATE, INSERT, DROP ON mysql.backup_history
     > TO 'backupuser'@'localhost';
Query OK, 0 rows affected (0.00 sec)

mysql> GRANT REPLICATION CLIENT ON *.* TO 'backupuser'@'localhost';
Query OK, 0 rows affected (0.00 sec)

mysql> GRANT SUPER ON *.* TO 'backupuser'@'localhost';
Query OK, 0 rows affected (0.00 sec)

mysql> GRANT CREATE TEMPORARY TABLES ON mysql.*
     > TO 'backupuser'@'localhost';
Query OK, 0 rows affected (0.00 sec)
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

10. 在第二个终端窗口中，使用 world_innodb 数据库并创建名为 City_Large 的 City 表的副本。在终端窗口中输入以下内容，得到的结果如下所示：

```
mysql> USE world_innodb;
...
Database changed
mysql> CREATE TABLE City_Large LIKE City;
```

11. 在第二个终端窗口中，发出以下命令将 City 表中的所有记录插入到 City_Large 表中。得到的结果如下所示：

```
mysql> INSERT INTO City_Large (Name, CountryCode, District,
    ->  Population) SELECT Name, CountryCode, District,
    ->  Population FROM City;
Query OK, 4078 rows affected, 1 warning (1.78 sec)
Records: 4078  Duplicates: 0  Warnings: 1
```

12. 在第二个终端窗口中，发出以下命令复制 City_Large 表中的记录，得到的结果如下所示：

```
mysql> INSERT INTO City_Large (Name, CountryCode, District,
    ->  Population) SELECT Name, CountryCode, District,
    ->  Population FROM City_Large;
Query OK, 4078 rows affected, 1 warning (1.68 sec)
Records: 4078  Duplicates: 0  Warnings: 1
```

13. 在第二个终端窗口中，执行上一步中的 SQL 语句七次。在终端窗口中输入以下内容，得到的结果如下所示：
    - **注：**操作完成的时间会由于添加到 City_Large 表中的记录数量指数级成倍增长而逐渐增加。

```
mysql> INSERT INTO City_Large (Name, CountryCode, District,
    ->  Population) SELECT Name, CountryCode, District,
    ->  Population FROM City_Large;
Query OK, 8156 rows affected, 1 warning (0.34 sec)
Records: 8156  Duplicates: 0  Warnings: 1

mysql> INSERT INTO City_Large (Name, CountryCode, District,
    ->  Population) SELECT Name, CountryCode, District,
    ->  Population FROM City_Large;
Query OK, 16312 rows affected, 1 warning (0.64 sec)
Records: 16312  Duplicates: 0  Warnings: 1

mysql> INSERT INTO City_Large (Name, CountryCode, District,
    ->  Population) SELECT Name, CountryCode, District,
    ->  Population FROM City_Large;
Query OK, 32624 rows affected, 1 warning (0.77 sec)
Records: 32624  Duplicates: 0  Warnings: 1

mysql> INSERT INTO City_Large (Name, CountryCode, District,
    ->  Population) SELECT Name, CountryCode, District,
    ->  Population FROM City_Large;
Query OK, 65248 rows affected, 1 warning (1.54 sec)
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

```
Records: 65248  Duplicates: 0  Warnings: 1

mysql> INSERT INTO City_Large (Name, CountryCode, District,
    ->  Population) SELECT Name, CountryCode, District,
    ->  Population FROM City_Large;
Query OK, 130496 rows affected, 1 warning (2.63 sec)
Records: 130496  Duplicates: 0  Warnings: 1

mysql> INSERT INTO City_Large (Name, CountryCode, District,
    ->  Population) SELECT Name, CountryCode, District,
    ->  Population FROM City_Large;
Query OK, 260992 rows affected, 1 warning (4.58 sec)
Records: 260992  Duplicates: 0  Warnings: 1

mysql> INSERT INTO City_Large (Name, CountryCode, District,
    ->  Population) SELECT Name, CountryCode, District,
    ->  Population FROM City_Large;
Query OK, 521984 rows affected, 1 warning (11.33 sec)
Records: 521984  Duplicates: 0  Warnings: 1
```

14. 在第二个终端窗口中，在运行备份之前，执行一次 FLUSH LOGS 命令。这将迫使当前的二进制日志关闭，并打开下一个增量二进制日志。

```
mysql> FLUSH LOGS;
```

15. 在第二个终端窗口中，再一次发出第 12 步中的命令。不要等待此步骤完成，立即开始下一步。
注：该操作的运行时间应需要 20–40 秒。在此期间，返回到第一个终端窗口并执行 **mysqlbackup** 命令。这将提供一个模拟的 InnoDB 表“热”备份。

在终端窗口中输入以下内容，得到的结果如下所示：

```
mysql> INSERT INTO City_Large (Name, CountryCode, District,
    ->  Population) SELECT Name, CountryCode, District,
    ->  Population FROM City_Large;
Query OK, 1043968 rows affected, 1 warning (29.49 sec)
Records: 1043968  Duplicates: 0  Warnings: 1
```

16. 在第一个终端窗口中，发出以下命令（同时继续执行上一步中的 mysql 命令）以 mysql 用户身份运行 mysqlbackup 应用程序。输入以下命令，得到的结果如下所示：

```
# su mysql
$ mysqlbackup --defaults-file=/etc/my.cnf -p backup-and-apply-log
MySQL Enterprise Backup version 3.8.1 [2013/01/28]
Copyright (c) 2003, 2012, Oracle and/or its affiliates. All Rights
Reserved.

 mysqlbackup: INFO: Starting with following command line ...
 mysqlbackup --defaults-file=/etc/my.cnf -p backup-and-apply-log

Enter password: oracle
 mysqlbackup: INFO: MySQL server version is '5.6.10-enterprise-
commercial-advanced-log'.
 mysqlbackup: INFO: Got some server configuration information from
running server.
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

```
IMPORTANT: Please check that mysqlbackup run completes successfully.
           At the end of a successful 'backup-and-apply-log' run
mysqlbackup
           prints "mysqlbackup completed OK!".

--------------------------------------------------------------------
                      Server Repository Options:
--------------------------------------------------------------------
  datadir = /var/lib/mysql/
  innodb_data_home_dir =
  innodb_data_file_path = ibdata1:12M:autoextend
  innodb_log_group_home_dir = /var/lib/mysql/
  innodb_log_files_in_group = 2
  innodb_log_file_size = 268435456
  innodb_page_size = 16384
  innodb_checksum_algorithm = innodb
  innodb_undo_directory = /var/lib/mysql/
  innodb_undo_tablespaces = 0
  innodb_undo_logs = 128

--------------------------------------------------------------------
                      Backup Config Options:
--------------------------------------------------------------------
  datadir = /backups/meb1/datadir
  innodb_data_home_dir = /backups/meb1/datadir
  innodb_data_file_path = ibdata1:12M:autoextend
  innodb_log_group_home_dir = /backups/meb1/datadir
  innodb_log_files_in_group = 2
  innodb_log_file_size = 268435456
  innodb_page_size = 16384
  innodb_checksum_algorithm = innodb
  innodb_undo_directory = /backups/meb1/datadir
  innodb_undo_tablespaces = 0
  innodb_undo_logs = 128

 mysqlbackup: INFO: Unique generated backup id for this is
13605079305675972

 mysqlbackup: INFO: Uses posix_fadvise() for performance optimization.
 mysqlbackup: INFO: Creating 14 buffers each of size 16777216.
130210 14:52:12 mysqlbackup: INFO: Full Backup operation starts with
following threads
                1 read-threads    6 process-threads    1 write-threads
130210 14:52:12 mysqlbackup: INFO: System tablespace file format is
Antelope.
130210 14:52:12 mysqlbackup: INFO: Starting to copy all innodb
files...
130210 14:52:12 mysqlbackup: INFO: Copying /var/lib/mysql/ibdata1
(Antelope file format).
130210 14:52:12 mysqlbackup: INFO: Found checkpoint at lsn 309944491.
130210 14:52:12 mysqlbackup: INFO: Starting log scan from lsn
309944320.
130210 14:52:12 mysqlbackup: INFO: Copying log...
130210 14:52:12 mysqlbackup: INFO: Log copied, lsn 309944491.
130210 14:52:12 mysqlbackup: INFO: Copying
/var/lib/mysql/mysql/innodb_index_stats.ibd (Antelope file format).
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

```
130210 14:52:12 mysqlbackup: INFO: Copying
/var/lib/mysql/mysql/innodb_table_stats.ibd (Antelope file format).
...
130210 14:52:20 mysqlbackup: INFO: Copying the database directory
'world_innodb'
130210 14:52:20 mysqlbackup: INFO: Completing the copy of all non-
innodb files.
130210 14:52:20 mysqlbackup: INFO: A copied database page was modified
at 309944491.
          (This is the highest lsn found on page)
          Scanned log up to lsn 309947815.
          Was able to parse the log up to lsn 309947815.
          Maximum page number for a log record 430
130210 14:52:20 mysqlbackup: INFO: All tables unlocked
130210 14:52:20 mysqlbackup: INFO: All MySQL tables were locked for
0.496 seconds.
130210 14:52:20 mysqlbackup: INFO: Full Backup operation completed
successfully.
130210 14:52:20 mysqlbackup: INFO: Backup created in directory
'/backups/meb1'
130210 14:52:20 mysqlbackup: INFO: MySQL binlog position: filename
mybinlog.000003, position 2671

------------------------------------------------------------
   Parameters Summary
------------------------------------------------------------
   Start LSN                  : 309944320
   End LSN                    : 309947815
------------------------------------------------------------


 mysqlbackup: INFO: Creating 14 buffers each of size 65536.
130210 14:52:20 mysqlbackup: INFO: Apply-log operation starts with
following threads
                1 read-threads    1 process-threads
130210 14:52:20 mysqlbackup: INFO: ibbackup_logfile's creation
parameters:
          start lsn 309944320, end lsn 309947815,
          start checkpoint 309944491.
 mysqlbackup: INFO: InnoDB: Starting an apply batch of log records to
the database...
InnoDB: Progress in percent: 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16
17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39
40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62
63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80 81 82 83 84 85
86 87 88 89 90 91 92 93 94 95 96 97 98 99 Setting log file size to
268435456
InnoDB: Progress in MB: 100 200
Setting log file size to 268435456
InnoDB: Progress in MB: 100 200
130210 14:52:30 mysqlbackup: INFO: We were able to parse
ibbackup_logfile up to
          lsn 309947815.
 mysqlbackup: INFO: Last MySQL binlog file position 0 2671, file name
mybinlog.000003
130210 14:52:30 mysqlbackup: INFO: The first data file is
'/backups/meb1/datadir/ibdata1'
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

```
          and the new created log files are at '/backups/meb1/datadir'
130210 14:52:30 mysqlbackup: INFO: Apply-log operation completed
successfully.
130210 14:52:30 mysqlbackup: INFO: Full backup prepared for recovery
successfully.

mysqlbackup completed OK!
```

17. 在第一个终端窗口中，列出 /backups/meb1 目录中的文件，得到的结果如下所示：

```
$ ls /backups/meb1/
backup-my.cnf   datadir   meta
```

18. 在第一个终端窗口中，从 mysql 用户 shell 退出，返回到 root shell。然后，在 /var/lib/mysql 目录中，删除 ibdata1 文件和 world_innodb 目录。

    在终端窗口中输入以下内容，得到的结果如下所示：

```
$ exit
# cd /var/lib/mysql
# rm -f ibdata1
# rm -rf world_innodb
```

19. 在第二个终端窗口中，显示 MySQL 服务器包含的所有数据库。

    在 mysql 窗口中输入以下内容，得到的结果如下所示：

```
mysql> SHOW DATABASES;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| mysql              |
| performance_schema |
| sakila             |
| test               |
| world2             |
+--------------------+
6 rows in set (0.00 sec)
```

    – 数据库列表是否包含 world_innodb 数据库？否

20. 在第二个终端窗口中，退出 mysql 客户机。

```
mysql> EXIT;
```

21. 在第一个终端窗口中，关闭 MySQL 服务器。

```
# service mysql stop
Shutting down MySQL..                                      [  OK  ]
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

22. 在第一个终端窗口中，执行以下命令将创建的备份恢复到 /var/lib/mysql 目录，得到的结果如下所示：

```
# su mysql
$ mysqlbackup --defaults-file=/etc/my.cnf copy-back
MySQL Enterprise Backup version 3.6.0 [2011/07/01]
Copyright (c) 2003, 2011, Oracle and/or its affiliates. All Rights
Reserved.

INFO: Starting with following command line ...
./mysqlbackup --defaults-file=/etc/my.cnf --backup-dir=/backups/meb1
       copy-back

IMPORTANT: Please check that mysqlbackup run completes successfully.
           At the end of a successful 'copy-back' run mysqlbackup
           prints "mysqlbackup completed OK!".

mysqlbackup: INFO: Server repository configuration:
 datadir                         =  /var/lib/mysql
 innodb_data_home_dir            =  /var/lib/mysql
 innodb_data_file_path           =  ibdata1:10M:autoextend
 innodb_log_group_home_dir       =  /var/lib/mysql
 innodb_log_files_in_group       =  2
 innodb_log_file_size            =  256M

mysqlbackup: INFO: Backup repository configuration:
 datadir                         =  /backups/meb1/datadir
 innodb_data_home_dir            =  /backups/meb1/datadir
 innodb_data_file_path           =  ibdata1:10M:autoextend
 innodb_log_group_home_dir       =  /backups/meb1/datadir
 innodb_log_files_in_group       =  2
 innodb_log_file_size            =  268435456

mysqlbackup: INFO: Starting to copy back files
mysqlbackup: INFO: in '/backups/meb1/datadir' directory
mysqlbackup: INFO: back to original data directory '/var/lib/mysql'
mysqlbackup: INFO: Copying back directory
'/backups/meb1/datadir/mysql'
mysqlbackup: INFO: Copying back directory
'/backups/meb1/datadir/performance_schema'
mysqlbackup: INFO: Copying back directory '/backups/meb1/datadir/test'
mysqlbackup: INFO: Copying back directory
'/backups/meb1/datadir/world_innodb'
mysqlbackup: INFO: Starting to copy back InnoDB tables and indexes
in '/backups/meb1' back to original InnoDB data directory:
/var/lib/mysql
mysqlbackup: INFO: Copying back file '/backups/meb1/datadir/ibdata1'
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

```
mysqlbackup: INFO: Starting to copy back InnoDB log files
in '/backups/meb1/datadir' back to original InnoDB log directory
'/var/lib/mysql'
mysqlbackup: INFO: Copying back file
'/backups/meb1/datadir/ib_logfile0'
mysqlbackup: INFO: Copying back file
'/backups/meb1/datadir/ib_logfile1'
mysqlbackup: INFO: Finished copying backup files.
```

23. 在第一个终端窗口中，从 mysql shell 退出并启动 MySQL 服务器。

```
$ exit
# service mysql start
```

24. 在第二个终端窗口中，启动 mysql 客户机并显示 MySQL 服务器包含的所有数据库。在终端窗口中输入以下内容，得到的结果如下所示：

```
$ mysql --login-path=admin
...

mysql> SHOW DATABASES;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| mysql              |
| performance_schema |
| sakila             |
| test               |
| world2             |
| world_innodb       |
+--------------------+
7 rows in set (0.00 sec)
```

25. 在第二个终端窗口中，查看 world_innodb 数据库包含的所有表。在终端窗口中输入以下内容，得到的结果如下所示：

```
mysql> USE world_innodb
Database changed.

mysql> SHOW TABLES;
+------------------------+
| Tables_in_world_innodb |
+------------------------+
| City                   |
| CityLanguage           |
| City_Large             |
| City_part              |
| City_temp              |
| Country                |
| CountryLanguage        |
| CountryLanguage2       |
| DeletedCity            |
+------------------------+
9 rows in set (0.00 sec)
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

26. 在第二个终端窗口中，退出 **mysql** 客户机并关闭终端窗口。

```
mysql> EXIT
Bye
$ exit
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

练习 16-3：mysqldump

概览

在本练习中，您将使用 mysqldump 应用程序创建 world_innodb 数据库的备份副本，然后将大多数内容恢复到另一个数据库。要实现该目标，请执行以下操作：

- 使用 mysqldump 应用程序备份 world_innodb 数据库。
- 创建一个名为 world3 的新数据库。
- 使用 mysql 客户机创建曾备份的大多数表。
- 使用 mysqlimport 命令将备份的数据装入到创建的表中。
- 验证备份的表是否与从备份恢复到新数据库中的表相同。

假设

- 已安装并且正在运行 MySQL 服务器。
- 已安装 world_innodb 数据库。

持续时间

完成本练习大约需要 20 分钟。

任务

1. 使用 mysqldump 应用程序，在 /backups 中生成 world_innodb 数据库的制表符分隔的备份。
2. 查看 /backups 目录的内容。
   - 有多少 *.sql 文件位于此目录中？

     _____
   - 有多少 *.txt 文件位于此目录中？

     _____
3. 查看 Country.sql 文件的内容。
4. 查看 /backups/Country.txt 文件的前几行。
   - 查看数据，文件格式是什么？

     _____
5. 使用 mysqladmin 客户机创建一个名为 world3 的新数据库。
6. 使用 mysql 应用程序，将第 1 步中创建的 Country、City 和 CountryLanguage 表的每个 *.sql 文件装入到 world3 数据库中。
7. 使用 mysqlimport 应用程序，装入第 1 步中创建的 Country、City 和 CountryLanguage 表的每个 *.txt 文件。
8. 使用 mysql 客户机查看 world3 数据库中的表。
9. 查看 world3.City 表和 world_innodb.City 表的记录计数以验证这两个表包含相同的行数。
   - 这两个表的行数是否相同？ _____

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

10. 查看 world3.Country、world_innodb.Country、world3.CountryLanguage 和 world_innodb.CountryLanguage 表的记录计数以确认各个表包含相同的行数。
    - 各个 Country 表的行数是否相同？_____
    - 各个 CountryLanguage 表的行数是否相同？_____
11. 退出 mysql 客户机会话。

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

练习解答 16-3：mysqldump

任务

1. 使用 mysqldump 应用程序，在 /backups 中生成 world_innodb 数据库的制表符分隔的备份。

```
# mysqldump -uroot -p --tab=/backups world_innodb
Enter password: oracle
```

2. 查看 /backups 目录的内容。在终端窗口中输入以下内容，得到的结果如下所示：

```
# ls /backups -l
total 77828
-rw-r--r-- 1 root  root      1469 Feb 10 16:50 CityLanguage.sql
-rw-rw-rw- 1 mysql mysql        0 Feb 10 16:50 CityLanguage.txt
-rw-r--r-- 1 root  root      1616 Feb 10 16:50 City_Large.sql
-rw-rw-rw- 1 mysql mysql 79207921 Feb 10 16:51 City_Large.txt
-rw-r--r-- 1 root  root      1610 Feb 10 16:51 City_part.sql
-rw-rw-rw- 1 mysql mysql   107645 Feb 10 16:51 City_part.txt
-rw-r--r-- 1 root  root      2620 Feb 10 16:50 City.sql
-rw-r--r-- 1 root  root      1610 Feb 10 16:51 City_temp.sql
-rw-rw-rw- 1 mysql mysql   108797 Feb 10 16:51 City_temp.txt
-rw-rw-rw- 1 mysql mysql   143528 Feb 10 16:50 City.txt
-rw-r--r-- 1 root  root      1608 Feb 10 16:51 CountryLanguage2.sql
-rw-rw-rw- 1 mysql mysql    18234 Feb 10 16:51 CountryLanguage2.txt
-rw-r--r-- 1 root  root      1702 Feb 10 16:51 CountryLanguage.sql
-rw-rw-rw- 1 mysql mysql    18234 Feb 10 16:51 CountryLanguage.txt
-rw-r--r-- 1 root  root      2038 Feb 10 16:51 Country.sql
-rw-rw-rw- 1 mysql mysql    31755 Feb 10 16:51 Country.txt
-rw-r--r-- 1 root  root      1489 Feb 10 16:51 DeletedCity.sql
-rw-rw-rw- 1 mysql mysql       32 Feb 10 16:51 DeletedCity.txt
drwx------ 4 mysql mysql     4096 Feb 10 14:52 meb1
```

- 有多少 *.sql 文件位于此目录中？九个，world_innodb 数据库中的每个表各占一个
- 有多少 *.txt 文件位于此目录中？九个，world_innodb 数据库中的每个表各占一个

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

3. 查看 Country.sql 文件的内容。在终端窗口中输入以下内容，得到的结果如下所示：

```
# more /backups/Country.sql
-- MySQL dump 10.13  Distrib 5.6.10, for Linux (x86_64)
--
-- Host: localhost    Database: world_innodb
-- ------------------------------------------------------
-- Server version       5.6.10-enterprise-commercial-advanced-log

/*!40101 SET @OLD_CHARACTER_SET_CLIENT=@@CHARACTER_SET_CLIENT */;
/*!40101 SET @OLD_CHARACTER_SET_RESULTS=@@CHARACTER_SET_RESULTS */;
/*!40101 SET @OLD_COLLATION_CONNECTION=@@COLLATION_CONNECTION */;
/*!40101 SET NAMES utf8 */;
/*!40103 SET @OLD_TIME_ZONE=@@TIME_ZONE */;
/*!40103 SET TIME_ZONE='+00:00' */;
/*!40101 SET @OLD_SQL_MODE=@@SQL_MODE, SQL_MODE='' */;
/*!40111 SET @OLD_SQL_NOTES=@@SQL_NOTES, SQL_NOTES=0 */;

--
-- Table structure for table `Country`
--

DROP TABLE IF EXISTS `Country`;
/*!40101 SET @saved_cs_client     = @@character_set_client */;
/*!40101 SET character_set_client = utf8 */;
CREATE TABLE `Country` (
  `Code` char(3) NOT NULL DEFAULT '',
  `Name` char(52) NOT NULL DEFAULT '',
  `Continent` enum('Asia','Europe','North
America','Africa','Oceania','Antarctica','South America') NOT NULL
DEFAULT 'Asia',
  `Region` char(26) NOT NULL DEFAULT '',
  `SurfaceArea` float(10,2) NOT NULL DEFAULT '0.00',
  `IndepYear` smallint(6) DEFAULT NULL,
  `Population` int(11) NOT NULL DEFAULT '0',
  `LifeExpectancy` float(3,1) DEFAULT NULL,
  `GNP` float(10,2) DEFAULT NULL,
  `GNPOld` float(10,2) DEFAULT NULL,
  `LocalName` char(45) NOT NULL DEFAULT '',
  `GovernmentForm` char(45) NOT NULL DEFAULT '',
  `HeadOfState` char(60) DEFAULT NULL,
  `Capital` int(11) DEFAULT NULL,
  `Code2` char(2) NOT NULL DEFAULT '',
  PRIMARY KEY (`Code`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
/*!40101 SET character_set_client = @saved_cs_client */;

/*!40103 SET TIME_ZONE=@OLD_TIME_ZONE */;

/*!40101 SET SQL_MODE=@OLD_SQL_MODE */;
/*!40101 SET CHARACTER_SET_CLIENT=@OLD_CHARACTER_SET_CLIENT */;
/*!40101 SET CHARACTER_SET_RESULTS=@OLD_CHARACTER_SET_RESULTS */;
/*!40101 SET COLLATION_CONNECTION=@OLD_COLLATION_CONNECTION */;
/*!40111 SET SQL_NOTES=@OLD_SQL_NOTES */;

-- Dump completed on 2013-02-10 16:51:01
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

4. 查看 /backups/Country.txt 文件的前几行。

   在终端窗口中输入以下内容，得到的结果如下所示：

```
# head /backups/Country.txt
ABW     Aruba   North America   Caribbean       193.00  \N      103000
78.4      828.00   793.00   Aruba    Nonmetropolitan Territory of The
Netherlands    Beatrix 129     AW
AFG         Afghanistan         Asia       Southern and Central Asia
652090.00           1919      22720000            45.9      5976.00  \N
Afganistan/Afqanestan    Islamic Emirate Mohammad Omar   1       AF
...
```

   – 查看数据，文件格式是什么？这是制表符分隔的文本文件

5. 使用 mysqladmin 客户机创建一个名为 world3 的新数据库。

```
# mysqladmin -uroot -p create world3
Enter password: oracle
```

6. 使用 mysql 应用程序，将第 1 步中创建的 Country、City 和 CountryLanguage 表的每个 *.sql 文件装入到 world3 数据库中。在终端窗口中输入以下内容，得到的结果如下所示：

```
# cd /backups
# mysql -uroot -poracle world3 < Country.sql
Warning: Using a password on the command line interface can be
insecure.
# mysql -uroot -poracle world3 < CountryLanguage.sql
Warning: Using a password on the command line interface can be
insecure.
# mysql -uroot -poracle world3 < City.sql
Warning: Using a password on the command line interface can be
insecure.
```

7. 使用 mysqlimport 应用程序，装入第 1 步中创建的 Country、City 和 CountryLanguage 表的每个 *.txt 文件。在终端窗口中输入以下内容，得到的结果如下所示：

```
# mysqlimport -uroot -poracle world3 /backups/Country.txt
Warning: Using a password on the command line interface can be
insecure.
world3.Country: Records: 239  Deleted: 0  Skipped: 0  Warnings: 0
# mysqlimport -uroot -poracle world3 /backups/CountryLanguage.txt
Warning: Using a password on the command line interface can be
insecure.
world3.CountryLanguage: Records: 984  Deleted: 0  Skipped: 0
Warnings: 0
# mysqlimport -uroot -poracle world3 /backups/City.txt
Warning: Using a password on the command line interface can be
insecure.
world3.City: Records: 4078  Deleted: 0  Skipped: 0  Warnings: 0
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

8. 使用 mysql 客户机查看 world3 数据库中的表。在终端窗口中输入以下内容，得到的结果如下所示：

```
# mysql -uroot -poracle
Welcome to the MySQL monitor.  Commands end with ; or \g.
...
mysql> USE world3
Database changed
mysql> SHOW TABLES;
+------------------+
| Tables_in_world3 |
+------------------+
| City             |
| Country          |
| CountryLanguage  |
+------------------+
3 rows in set (0.01 sec)
```

9. 查看 world3.City 表和 world_innodb.City 表的记录计数以验证这两个表包含相同的行数。在终端窗口中输入以下内容，得到的结果如下所示：

```
mysql> SELECT COUNT(*) FROM world3.City;
+----------+
| COUNT(*) |
+----------+
|     4078 |
+----------+
1 row in set (0.00 sec)

mysql> SELECT COUNT(*) FROM world_innodb.City;
+----------+
| COUNT(*) |
+----------+
|     4078 |
+----------+
1 row in set (0.00 sec)
```

– 这两个表的行数是否相同？是

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

10. 查看 world3.Country、world_innodb.Country、world3.CountryLanguage 和 world_innodb.CountryLanguage 表的记录计数以确认各个表包含相同的行数。在终端窗口中输入以下内容，得到的结果如下所示：

```
mysql> SELECT COUNT(*) FROM world3.Country;
+-----------+
| COUNT(*) |
+-----------+
|      239 |
+-----------+
1 row in set (0.00 sec)

mysql> SELECT COUNT(*) FROM world_innodb.Country;
+-----------+
| COUNT(*) |
+-----------+
|      239 |
+-----------+
1 row in set (0.00 sec)

mysql> SELECT COUNT(*) FROM world3.CountryLanguage;
+-----------+
| COUNT(*) |
+-----------+
|      984 |
+-----------+
1 row in set (0.00 sec)

mysql> SELECT COUNT(*) FROM world_innodb.CountryLanguage;
+-----------+
| COUNT(*) |
+-----------+
|      984 |
+-----------+
1 row in set (0.00 sec)
```

- 各个 Country 表的行数是否相同？是
- 各个 CountryLanguage 表的行数是否相同？是

11. 退出 mysql 客户机会话。

```
mysql> EXIT
Bye
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

练习 16-4：使用 LVM 快照和二进制日志的备份和恢复

概览

在本练习中，您将执行对 MySQL 服务器数据的完整恢复。要实现该目标，请执行以下操作：

- 配置一个逻辑卷（在环路设备上），其中包含用于支持本练习的数据目录的一个副本。
- 生成 MySQL 数据目录的快照。
- 从快照备份。
- 对 `world_innodb` 数据库进行更改。
- 模拟严重的存储故障。
- 从备份恢复 MySQL 数据目录。
- 运行到某个点的二进制日志以恢复自快照以来所做的更改，并避免再次犯错。

假设

- 已安装并且正在运行 MySQL 服务器。
- `world_innodb.sql` 文件位于 `/labs` 目录中。

持续时间

完成本练习大约需要 40 分钟。

任务

1. 停止 MySQL 服务器。
2. 在终端窗口（以 `root` 身份登录）中执行以下命令，在环路设备 LVM 逻辑卷上创建 MySQL 数据目录的副本。

```
dd if=/dev/zero of=/var/local/mysqldisk bs=1M count=2000
losetup /dev/loop2 /var/local/mysqldisk
vgcreate VG_MYSQL /dev/loop2
lvcreate -l50%VG -n lv_datadir VG_MYSQL
mkfs.ext4 /dev/VG_MYSQL/lv_datadir
mkdir -p /datadir
mount /dev/VG_MYSQL/lv_datadir /datadir
rmdir /datadir/lost+found
cp -a /var/lib/mysql/* /datadir/
chown mysql:mysql /datadir
```

3. 创建名为 `/binlogs` 的目录，并将所有者和组更改为 `mysql`。
4. 将 MySQL 的数据目录设置更改为指向新的 `/datadir` 数据目录，并将二进制日志记录设置为记录到 `/binlogs/mysql-bin`。
5. 重新启动 MySQL。
6. 查看 `/binlogs` 目录的内容。MySQL 服务器是否正在将 binlog 写入此目录？

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

7. 从新的终端窗口使用 mysql 客户机，删除 world_innodb 数据库并使用 /labs/world_innodb.sql 文件重新创建该数据库，就像在“系统管理”一课的练习中所做的那样。
8. 查看 world_innodb 数据库包含的表数量。world_innodb 数据库包含多少表？
9. 通过发出以下命令，查看对 MySQL 服务器运行的进程：

```
SHOW PROCESSLIST;
```

   – 是否有正在对服务器运行的查询？

10. 通过执行“MySQL Enterprise Backup”练习中的步骤 9–12 重新创建表 City_Large，在服务器上模拟大通信流量。确保在以下步骤中创建快照时，有一个大型慢速 INSERT 操作在执行。在插入操作过程中将 autocommit 设置为 0 以延长事务的持续时间。
11. 在上一步仍在执行期间，通过在以 root 身份登录的终端窗口中发出以下命令，刷新 MySQL 日志并创建 MySQL 数据目录的 LVM 快照：

```
mysqladmin -uroot -poracle flush-logs
lvcreate -s -n lv_datadirbackup -L 500M /dev/VG_MYSQL/lv_datadir
```

12. 通过将 autocommit 重新设置为 1，提交在第 10 步中启动的事务。
13. 向 City 表添加一个新行，并设置国家/地区代码 SWE 和名称 Sakila。用适合的查询确认插入。然后，删除 City 表。
14. 通过将快照挂载到本地目录，使用 **tar** 命令生成快照内容的备份。在完成备份后，卸载快照。
15. 使用以下命令删除快照：

```
lvremove VG_MYSQL/lv_datadirbackup -f
```

16. 要模拟灾难，可执行以下命令：

```
lvresize -fL 1M VG_MYSQL/lv_datadir
```

   此命令将逻辑卷大小调整为 1 MB，从而有效地破坏其文件系统中的所有数据。

17. 在 MySQL 提示符下执行一些语句以查看 MySQL 是否仍在运行，然后退出客户机。
18. 尝试使用服务命令以及使用 mysqladmin 关闭 MySQL。
19. 通过发出 ps 命令验证 MySQL 是否在运行。
20. 终止 mysqld_safe 进程。
21. 重新执行第 19 步中的命令以确认 MySQL 不再运行。
22. 要模拟修复发生故障的文件系统，可在以 root 身份登录的终端窗口中执行以下命令，删除被破坏的 LVM 逻辑卷并创建一个新的 LVM 逻辑卷：

```
umount /datadir
lvremove -f VG_MYSQL/lv_datadir
lvcreate -l50%VG -n lv_datadir VG_MYSQL
mkfs.ext4 /dev/VG_MYSQL/lv_datadir
mount /dev/VG_MYSQL/lv_datadir /datadir
rmdir /datadir/lost+found
chown mysql:mysql /datadir
```

   注：这些命令类似于第 2 步中最初创建逻辑卷时的命令。您不会重新创建卷组、/datadir 目录或旧数据目录的副本。在此步骤结束时，数据目录将再次变空。

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

23. 恢复在第 14 步中生成的备份内容。
24. 重新启动 MySQL。
25. 从以 oracle 身份登录的终端登录到 mysql 客户机，并查看 world_innodb 数据库中的表。请确保存在 City 表，并确定第 13 步中添加的 Sakila 行是否存在。
26. 在终端窗口中，查看 /binlogs 目录的内容。
    - 在备份生成后启动了哪个 binlog？
27. 从备份生成后启动的 binlog 开始，查看每个 binlog 的内容以查找包含删除 world_innodb 数据库中 City 表的语句的二进制日志？
    - 哪个 binlog 包含 DROP TABLE 命令？
    - 包含 DROP TABLE 命令的位置是什么？
28. 使用在上一步中记录的数据，创建一个 SQL 脚本并在 mysql 客户机上执行，从而将 world_innodb 数据库恢复到 DROP TABLE 命令的位置。
29. 使用 mysql 客户机查看 world_innodb 数据库的 City 表中 ID 大于 4070 的所有记录。
    - City 表中是否有在第 13 步中添加的记录？
30. 将 MySQL 的数据目录设置改回原始值，然后重新启动 MySQL 以应用更改。

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

练习解答 16-4：使用 LVM 快照和二进制日志的备份和恢复

任务

1. 停止 MySQL 服务器。

   在终端窗口（以 root 身份登录）中执行以下命令，得到的结果如下所示：

```
# service mysql stop
Shutting down MySQL..                                      [  OK  ]
```

2. 在终端窗口（以 root 身份登录）中执行以下命令，在环路设备 LVM 逻辑卷上创建 MySQL 数据目录的副本。

```
dd if=/dev/zero of=/var/local/mysqldisk bs=1M count=2000
losetup /dev/loop2 /var/local/mysqldisk
vgcreate VG_MYSQL /dev/loop2
lvcreate -l50%VG -n lv_datadir VG_MYSQL
mkfs.ext4 /dev/VG_MYSQL/lv_datadir
mkdir -p /datadir
mount /dev/VG_MYSQL/lv_datadir /datadir
rmdir /datadir/lost+found
cp -a /var/lib/mysql/* /datadir/
chown mysql:mysql /datadir
```

3. 创建名为 /binlogs 的目录，并将所有者和组更改为 mysql。

```
# mkdir /binlogs
# chown mysql:mysql /binlogs
```

4. 将 MySQL 的数据目录设置更改为指向新的 /datadir 数据目录，并将二进制日志记录设置为记录到 /binlogs/mysql-bin。

   以 root 身份打开 /etc/my.cnf 文件，并进行以下更改：

```
[mysqld]
...
log-bin=/binlogs/mysql-bin
...
datadir=/datadir
...
```

5. 重新启动 MySQL。

   在终端窗口（以 root 身份登录）中执行以下命令，得到的结果如下所示：

```
# service mysql start
Starting MySQL.                                            [  OK  ]
```

6. 查看 /binlogs 目录的内容。在终端窗口中输入以下内容，得到的结果如下所示：

```
# ls /binlogs
mysql-bin.000001  mysql-bin.index
```

   – MySQL 服务器是否正在将 binlog 写入此目录？是

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

7. 从新的终端窗口使用 mysql 客户机，删除 world_innodb 数据库并使用 /labs/world_innodb.sql 文件重新创建该数据库，就像在“系统管理”一课的练习中所做的那样。

```
$ mysql --login-path=admin
Welcome to the MySQL monitor.  Commands end with ; or \g.
...
mysql> DROP DATABASE world_innodb;
Query OK, 4 rows affected (1.45 sec)

mysql> CREATE DATABASE world_innodb;
Query OK, 1 row affected (0.00 sec)

mysql> USE world_innodb
Database changed
mysql> SET autocommit=0;
Query OK, 0 rows affected (0.00 sec)

mysql> SOURCE /labs/world_innodb.sql
Query OK, 0 rows affected (0.00 sec)

Query OK, 0 rows affected (0.00 sec)
...
mysql> SET autocommit=1;
Query OK, 0 rows affected (0.04 sec)
```

8. 查看 world_innodb 数据库包含的表数量。在 mysql 窗口中输入以下内容，得到的结果如下所示：

```
mysql> SHOW TABLES;
+-------------------------+
| Tables_in_world_innodb |
+-------------------------+
| City                    |
| Country                 |
| CountryLanguage         |
+-------------------------+
3 rows in set (0.00 sec)
```

- world_innodb 数据库包含多少表？三个 — 这是新创建的 world_innodb 数据库中的正确表数目。

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

9. 通过发出以下命令查看对 MySQL 服务器运行的进程，得到的结果如下所示：

```
mysql> SHOW PROCESSLIST;
+----+------+-----------+-----+---------+------+-------+--------------
---+
| Id | User | Host      | db  | Command | Time | State | Info
|
+----+------+-----------+-----+---------+------+-------+--------------
---+
|  2 | root | localhost | NULL| Query   |    0 | NULL  | SHOW
PROCESSLIST|
+----+------+-----------+-----+---------+------+-------+--------------
---+
1 row in set (0.00 sec)
```

– 是否有正在对服务器运行的查询？否

10. 通过执行“MySQL Enterprise Backup”练习中的步骤 9–12 重新创建表 City_Large，在服务器上模拟大通信流量。确保在以下步骤中创建快照时，有一个大型慢速 INSERT 操作在执行。在插入操作过程中将 **autocommit** 设置为 0 以延长事务的持续时间。

```
mysql> CREATE TABLE City_Large LIKE City;
Query OK, 0 rows affected (2.82 sec)

mysql> INSERT INTO City_Large (Name, CountryCode, District,
    ->  Population) SELECT Name, CountryCode, District,
    ->  Population FROM City;
Query OK, 4079 rows affected (0.14 sec)
Records: 4079  Duplicates: 0  Warnings: 0

mysql> SET autocommit=0;
```

发出以下命令 6–10 次以极大地增加 City_Large 表中的记录数量，从而导致在快照生成期间服务器上出现通信流量。使用键盘上的向上箭头可快速重复命令。

```
mysql> INSERT INTO City_Large (Name, CountryCode, District,
    ->  Population) SELECT Name, CountryCode, District,
    ->  Population FROM City_Large;
```

11. 在上一步仍在执行期间，通过在以 root 身份登录的终端窗口中发出以下命令，刷新 MySQL 日志并创建 MySQL 数据目录的 LVM 快照。

输入以下命令，得到的结果如下所示：

```
# mysqladmin -uroot -poracle flush-logs ; \
  lvcreate -s -n lv_datadirbackup -L 500M /dev/VG_MYSQL/lv_datadir
  Logical volume "lv_datadirbackup" created
```

注：此技术不允许获取二进制日志位置，因此在某些情况下（如创建复制从属服务器时）不适用于创建备份。

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

12. 通过将 autocommit 重新设置为 1，提交在第 10 步中启动的事务。

```
mysql> SET autocommit=1;
Query OK, 0 rows affected (0.66 sec)
```

13. 向 City 表添加一个新行，并设置国家/地区代码 SWE 和名称 Sakila。用适合的查询确认插入。然后，删除 City 表。

    在 mysql 窗口中输入以下内容，得到的结果如下所示：

```
mysql> INSERT INTO City(Name, CountryCode) VALUES ('Sakila', 'SWE');
Query OK, 1 row affected (0.11 sec)

mysql> SELECT * FROM City ORDER BY ID DESC LIMIT 1;
+------+--------+-------------+----------+------------+
| ID   | Name   | CountryCode | District | Population |
+------+--------+-------------+----------+------------+
| 4080 | Sakila | SWE         |          |          0 |
+------+--------+-------------+----------+------------+
1 row in set (0.00 sec)

mysql> DROP TABLE City;
Query OK, 0 rows affected (0.31 sec)
```

    注：这些步骤在快照之后、备份操作之前执行。以下备份从快照复制，而不是从活动服务器复制。

14. 通过将快照挂载到本地目录，使用 **tar** 命令生成快照内容的备份。在完成备份后，卸载快照。

    在 Linux 提示符（以 root 身份登录）下输入以下命令：

```
mkdir /root/snapshot
mount /dev/VG_MYSQL/lv_datadirbackup /root/snapshot
cd /root/snapshot
tar -czf /root/mysqldatadir.tgz .
cd
umount /root/snapshot
```

15. 使用以下命令删除快照：

    在 Linux 提示符（以 root 身份登录）下输入以下命令，得到的结果如下所示：

```
# lvremove VG_MYSQL/lv_datadirbackup -f
  Logical volume "lv_datadirbackup" successfully removed
```

16. 要模拟灾难，可执行以下命令。

    在 Linux 提示符（以 root 身份登录）下输入以下命令：

```
# lvresize -fL 1M VG_MYSQL/lv_datadir
```

    此命令将逻辑卷的大小调整为最小的物理盘区（4 MB，从指定的 1 MB 向上舍入而得），从而有效地破坏其文件系统中的所有数据。

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

17. 在 MySQL 提示符下执行一些语句以查看 MySQL 是否仍在运行，然后退出客户机。

```
mysql> SHOW DATABASES;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| mysql              |
| performance_schema |
| world3             |
| world_innodb       |
+--------------------+
5 rows in set (0.00 sec)

mysql> DROP DATABASE world3;
No connection. Trying to reconnect...
ERROR 2002 (HY000): Can't connect to local MySQL server through socket
'/var/lib/mysql/mysql.sock' (2)

mysql> EXIT
Bye
```

注：崩溃的早晚取决于 MySQL 缓存的信息量。请先尝试 `STATUS` 和 `SHOW DATABASES` 之类的命令及几条 **`USE`** 语句，然后尝试修改数据库。一旦 MySQL 无法记录更改，就会发生崩溃。

注：可能会看到各种不同的错误，例如“ERROR 2013 (HY000): Lost connection to MySQL server during query”（错误 2013 (HY000)：在查询过程中丢失与 MySQL 服务器的连接）或者“ERROR 2006 (HY000): MySQL server has gone away”（错误 2006 (HY000)：MySQL 服务器已脱机）。还可能会看到一些操作系统消息；例如，“kernel:journal commit I/O error”（内核：日志提交 I/O 错误）是指由卷缩小导致的严重文件系统问题。

18. 尝试使用服务命令以及使用`mysqladmin`关闭 MySQL。

在 Linux 提示符（以 `root` 身份登录）下输入以下命令：

```
# service mysql stop
MySQL server process #17951 is not running!                [FAILED]
rm: cannot remove `/datadir/<hostname>.pid': Read-only file system
# mysqladmin -uroot -poracle shutdown
mysqladmin: connect to server at 'localhost' failed
error: 'Can't connect to local MySQL server through socket
'/var/lib/mysql/mysql.sock' (2)'
Check that mysqld is running and that the socket:
'/var/lib/mysql/mysql.sock' exists!
```

注：您已经模拟了一次严重的硬盘故障，因此错误消息可能与所示的消息不同。

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

19. 通过发出 ps 命令验证 MySQL 是否在运行。

在 Linux 提示符（以 root 身份登录）下输入以下命令：

```
# ps ax | grep mysqld
17738 pts/2    S      2:11 /bin/sh /usr/bin/mysqld_safe --
datadir=/datadir --pid-file=/datadir/<hostname>.pid
17846 pts/2    R      0:00 /usr/sbin/mysqld --basedir=/usr --
datadir=/datadir --plugin-dir=/usr/lib64/mysql/plugin --user=mysql --
log-error=/var/log/mysqld.log --pid-file=/datadir/<hostname>.pid --
socket=/var/lib/mysql/mysql.sock
17849 pts/2    S+     0:00 grep mysqld
```

MySQL 仍在运行，因为每次崩溃时 mysqld_safe 脚本都会将其重新启动。如果多次运行此语句，将会注意到 mysqld 有时运行，有时不运行。

20. 终止 mysqld_safe 进程。

在 Linux 提示符（以 root 身份登录）下输入以下命令：

```
# killall -9 mysqld_safe
```

21. 重新执行第 19 步中的命令以确认 MySQL 不再运行。

在 Linux 提示符（以 root 身份登录）下输入以下命令：

```
# ps ax | grep mysqld
11528 pts/2    S+     0:00 grep mysqld
```

MySQL 不再运行。

22. 要模拟修复发生故障的文件系统，可在以 root 身份登录的终端窗口中执行以下命令，删除被破坏的 LVM 逻辑卷并创建一个新的 LVM 逻辑卷：

```
umount /datadir
lvremove -f VG_MYSQL/lv_datadir
lvcreate -l50%VG -n lv_datadir VG_MYSQL
mkfs.ext4 /dev/VG_MYSQL/lv_datadir
mount /dev/VG_MYSQL/lv_datadir /datadir
rmdir /datadir/lost+found
chown mysql:mysql /datadir
```

注：这些命令类似于第 2 步中最初创建逻辑卷时的命令。您不会重新创建卷组、/datadir 目录或旧数据目录的副本。在此步骤结束时，数据目录将再次变空。

23. 恢复在第 14 步中生成的备份内容。

在 Linux 提示符（以 root 身份登录）下输入以下命令：

```
# cd /datadir
# tar -xzf /root/mysqldatadir.tgz
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

24. 重新启动 MySQL。

在终端窗口（以 root 身份登录）中执行以下命令，得到的结果如下所示：

```
# service mysql start
Starting MySQL....                                         [  OK  ]
```

您是否能启动 MySQL 服务器？是

注：如果在有大量活动期间创建了快照，则该启动操作可能需要一些时间，因为 InnoDB 需要在异常关闭后启动时执行恢复，而正在运行的系统的快照与断电或其他崩溃无法区分。

25. 从以 oracle 身份登录的终端登录到 mysql 客户机，并查看 world_innodb 数据库中的表。请确保存在 City 表，并确定第 12 步中添加的 Sakila 行是否存在。

在终端窗口中输入以下内容，得到的结果如下所示：

```
$ mysql --login-path=admin
Welcome to the MySQL monitor.  Commands end with ; or \g.
...
mysql> USE world_innodb
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql> SELECT * FROM City ORDER BY ID DESC LIMIT 1;
+------+-------+-------------+----------+------------+
| ID   | Name  | CountryCode | District | Population |
+------+-------+-------------+----------+------------+
| 4079 | Rafah | PSE         | Rafah    |      92020 |
+------+-------+-------------+----------+------------+
1 row in set (0.00 sec)
```

City 表存在，但 Sakila 行不存在，因为备份来自于在创建行并删除 City 表之前生成的快照。可以使用二进制日志恢复自从备份以来所做的更改。

26. 在终端窗口中，查看 **/binlogs** 目录的内容。

在终端窗口中输入以下内容，得到的结果如下所示：

```
# ls /binlogs -l
total 856
-rw-rw---- 1 mysql mysql 863061 Feb 11 05:09 mysql-bin.000001
-rw-rw---- 1 mysql mysql    940 Feb 11 05:42 mysql-bin.000002
-rw-rw---- 1 mysql mysql    120 Feb 11 05:42 mysql-bin.000003
-rw-rw---- 1 mysql mysql     78 Feb 11 05:42 mysql-bin.index
```

– 在备份生成后启动了哪个 binlog？mysql-bin.000002

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

27. 从快照之后创建的第一个 binlog 开始，查看每个 binlog 的内容以查找包含删除 world_innodb 数据库中 City 表的语句的二进制日志？

在终端窗口中输入以下内容，得到的结果如下所示：

```
# mysqlbinlog /binlogs/mysql-bin.000002 | more
/*!50530 SET @@SESSION.PSEUDO_SLAVE_MODE=1*/;
/*!40019 SET @@session.max_insert_delayed_threads=0*/;
/*!50003 SET
@OLD_COMPLETION_TYPE=@@COMPLETION_TYPE,COMPLETION_TYPE=0*/;
DELIMITER /*!*/;
# at 4
#130211  5:09:50 server id 1  end_log_pos 120 CRC32 0xc405eb90  Start:
binlog v 4, server v 5.6
.10-enterprise-commercial-advanced-log created 130211  5:09:50
BINLOG '
Hn0YUQ8BAAAAdAAAAHgAAAAAAQANS42LjEwLWVudGVycHJpc2UtY29tbWVyY2lhbC1hZH
ZhbmNl
ZC1sb2cAAAAAAAAAAAAAAAAAAEzgNAAgAEgAEBAQEEgAAXAAEGggAAAAICAgCAAAACgoKGR
kAAZDr
BcQ=
'/*!*/;
# at 120
#130211  5:09:36 server id 1  end_log_pos 215 CRC32 0xa312ddd4  Query
thread_id=3     exec_ti
me=28   error_code=0
SET TIMESTAMP=1360559376/*!*/;
SET @@session.pseudo_thread_id=3/*!*/;
SET @@session.foreign_key_checks=1, @@session.sql_auto_is_null=0,
@@session.unique_checks=1, @@
session.autocommit=1/*!*/;
SET @@session.sql_mode=1075838976/*!*/;
SET @@session.auto_increment_increment=1,
@@session.auto_increment_offset=1/*!*/;
/*!\C utf8 *//*!*/;
SET
@@session.character_set_client=33,@@session.collation_connection=33,@@
session.collation_ser
ver=8/*!*/;
SET @@session.lc_time_names=0/*!*/;
SET @@session.collation_database=DEFAULT/*!*/;
BEGIN
/*!*/;
# at 215
# at 247
#130211  5:09:36 server id 1  end_log_pos 247 CRC32 0x3fd40695  Intvar
SET INSERT_ID=524277/*!*/;
#130211  5:09:36 server id 1  end_log_pos 468 CRC32 0xccb3fceb  Query
thread_id=3     exec_ti
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

```
me=28   error_code=0
use `world_innodb`/*!*/;
SET TIMESTAMP=1360559376/*!*/;
INSERT INTO City_Large (Name, CountryCode, District,
 Population) SELECT Name, CountryCode, District,
 Population FROM City_Large
/*!*/;
# at 468
#130211  5:09:36 server id 1  end_log_pos 499 CRC32 0x5c1a7c2e  Xid =
5572
COMMIT/*!*/;
# at 499
#130211  5:12:46 server id 1  end_log_pos 594 CRC32 0xc74b18b0  Query
thread_id=3     exec_ti
me=0    error_code=0
SET TIMESTAMP=1360559566/*!*/;
BEGIN
/*!*/;
# at 594
# at 626
#130211  5:12:46 server id 1  end_log_pos 626 CRC32 0x36ec76bb  Intvar
SET INSERT_ID=4080/*!*/;
#130211  5:12:46 server id 1  end_log_pos 776 CRC32 0xdaffdd6e  Query
thread_id=3     exec_ti
me=0    error_code=0
SET TIMESTAMP=1360559566/*!*/;
INSERT INTO City(Name, CountryCode) VALUES ('Sakila', 'SWE')
/*!*/;
# at 776
#130211  5:12:46 server id 1  end_log_pos 807 CRC32 0x0deff828  Xid =
5626
COMMIT/*!*/;
# at 807
#130211  5:13:01 server id 1  end_log_pos 940 CRC32 0xce845fd6  Query
thread_id=3     exec_ti
me=2    error_code=0
SET TIMESTAMP=1360559581/*!*/;
DROP TABLE `City` /* generated by server */
/*!*/;
DELIMITER ;
# End of log file
ROLLBACK /* added by mysqlbinlog */;
/*!50003 SET COMPLETION_TYPE=@OLD_COMPLETION_TYPE*/;
/*!50530 SET @@SESSION.PSEUDO_SLAVE_MODE=0*/;
```

– 哪个 binlog 包含 DROP TABLE 命令？mysql-bin.000002

– 包含 DROP TABLE 命令的位置是什么？DROP TABLE 命令发生在位置 807。

注：在您的系统上，这些值可能有所不同。

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

28. 使用在上一步中记录的数据，创建一个 SQL 脚本并在 mysql 客户机上执行，从而将 world_innodb 数据库恢复到 DROP TABLE 命令的位置。

```
# cd /binlogs
# mysqlbinlog --disable-log-bin --stop-position=807 \
    mysql-bin.000002 | mysql -uroot -poracle
```

29. 使用 mysql 客户机查看 world_innodb 数据库的 City 表中 **ID** 大于 4070 的所有记录。

在终端窗口中输入以下内容，得到的结果如下所示：

```
mysql> SELECT * FROM world_innodb.City WHERE ID > 4070;
+------+--------------+-------------+------------+------------+
| ID   | Name         | CountryCode | District   | Population |
+------+--------------+-------------+------------+------------+
| 4071 | Mount Darwin | ZWE         | Harare     |     164362 |
| 4072 | Mutare       | ZWE         | Manicaland |     131367 |
| 4073 | Gweru        | ZWE         | Midlands   |     128037 |
| 4074 | Gaza         | PSE         | Gaza       |     353632 |
| 4075 | Khan Yunis   | PSE         | Khan Yunis |     123175 |
| 4076 | Hebron       | PSE         | Hebron     |     119401 |
| 4077 | Jabaliya     | PSE         | North Gaza |     113901 |
| 4078 | Nablus       | PSE         | Nablus     |     100231 |
| 4079 | Rafah        | PSE         | Rafah      |      92020 |
| 4080 | Sakila       | SWE         |            |          1 |
+------+--------------+-------------+------------+------------+
10 rows in set (0.00 sec)
```

– City 表中是否有在第 13 步中添加的记录？是。

30. 将 MySQL 的数据目录设置改回原始值，然后重新启动 MySQL 以应用更改。

在终端窗口（以 root 身份登录）中执行以下命令，得到的结果如下所示：

```
# service mysql stop
Shutting down MySQL..                                      [  OK  ]
```

以 root 身份打开 **/etc/my.cnf** 文件，并进行以下更改：

```
[mysqld]
...
datadir=/var/lib/mysql
...
```

保存前面的更改之后，在终端窗口中输入以下内容，得到的结果如下所示：

```
# service mysql start
Starting MySQL.                                            [  OK  ]
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

第 17 课的练习：复制

第 17 章

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

第 17 课的练习：概览

练习概览

通过这些练习，可测试您对复制知识的掌握程度。上机练习是针对 Oracle Linux 操作系统环境编写的，Oracle 课堂中将提供此环境。对于非 Oracle 课堂环境，可能需要对文件位置进行一些调整。

假设

- 在进行这些练习之前已经完成了 MySQL 服务器安装。
- 已安装 MySQL 实用程序。
- 存在文件 `/labs/repl.cnf`，包含第二个练习中显示的内容。
- 在终端窗口中以 `root` 用户身份登录。
- 您可以通过命令行提示符访问 `mysql` 客户机。

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

练习 17-1：测验 — 复制

概览

在本练习中，您将回答有关复制的问题。

持续时间

本练习应该需要大约五分钟来完成。

测验问题

请从所提供的每道多选题或正/误判断题中选择最佳答案。

1. 下列关于 MySQL 主服务器的描述中，哪些是正确的？
    a. 一个主服务器可以具有的从属服务器数量没有限制。
    b. 从属服务器可以具有与主服务器不同的 MySQL 版本。
    c. 在大多数生产设置中通常将从属服务器的数量限制为少于 30。
    d. 以上都是
2. 简单来说，MySQL 复制通过单向、日志传送、异步机制来工作，使其成为主服务器-从属服务器关系。
    a. 正确
    b. 错误
3. 下列哪项常用于复制？
    a. 向外扩展解决方案
    b. 高可用性
    c. Analytics
    d. 以上都是
4. MySQL 复制使用日志传送系统，其中主服务器上发生的所有数据更改都存储在日志中，然后通过从属服务器检索这些数据更改，并根据接收到的这些日志文件执行更改。MySQL 中此日志文件的名称是什么？
    a. 从属服务器日志
    b. 主服务器日志
    c. 二进制日志
    d. 错误日志
5. 从属服务器需要永久连接来接收来自主服务器的更新。
    a. 正确
    b. 错误
6. 使用基于语句的复制的缺点是什么？
    a. 复制的磁盘空间使用率和带宽要求较大。
    b. 复制在行级别发生。
    c. 一些功能可能无法正确复制到不同版本的远程服务器。
    d. 以上都不是

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

7. 哪个线程负责将二进制日志从主服务器下载到称为中继日志的本地文件集？
    a. BINARY_THREAD
    b. IO_THREAD
    c. SQL_THREAD
    d. MASTER_THREAD

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

练习解答 17-1：测验 — 复制

测验解答

1. **d.** 以上都是
2. **a.** 正确
3. **d.** 以上都是
4. **c.** 二进制日志
5. **b.** 错误。从属服务器不不需要永久连接即可接收来自主服务器的更新。
6. **c.** 一些功能可能无法正确复制到不同版本的远程服务器。
7. **b.** IO_THREAD

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

练习 17-2：配置复制

概览

在本练习中，您将启动 MySQL 的四个服务器实例，将一个服务器配置为另一个服务器的从属服务器，在主服务器上创建一些数据，并看到其复制到该从属服务器。

假设

已安装 MySQL 服务器，并且存在文件 `/labs/repl.cnf`，其中包含此练习的步骤 2 解答中所示的内容。

持续时间

完成本练习大约需要 10 分钟。

任务

1. 停止 MySQL 服务器。
2. 查看文件 `/labs/repl.cnf` 的内容。
3. 使用 `mysqld_multi`，启动 `/labs/repl.cnf` 中定义的四个服务器。
4. 在新的终端窗口中，使用 `mysql` 客户机以 `root` 身份连接第一个服务器，并将提示符设置为“`1>`”。**注：**为了培训环境简单起见，这四个服务器没有 `root` 口令。
5. 执行查询来查找第一个服务器的日志坐标。
6. 在第一个服务器上，创建名为 `repl` 的用户，口令为 `oracle`，并授予该用户 `REPLICATION SLAVE` 权限。
7. 在第一个服务器上，通过脚本创建 `world_innodb` 数据库，如您在“系统管理”一课的练习中所做的那样。
8. 在新终端窗口中，使用 `mysql` 客户机以 `root` 身份连接第二个服务器，并将提示符设置为“`2>`”。
9. 在第二个 `mysql` 窗口中，发出 `CHANGE MASTER TO…` 命令将第二个服务器配置为第一个服务器的从属服务器，使用步骤 5 中指出的日志坐标和步骤 6 中创建的用户。显示生成的任何警告的文本。
10. 显示第二个服务器上存在的数据库。
11. 在第二个服务器上启动从属服务器线程。
12. 显示第一个和第二个服务器上的进程。
13. 再次显示第二个服务器上的数据库。注意差别。

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

练习解答 17-2：配置复制

任务

1. 停止 MySQL 服务器。

   在以 root 身份登录的终端中输入以下内容，得到的结果如下所示：

```
# service mysql stop
Shutting down MySQL..                                      [  OK  ]
```

2. 查看文件 /labs/repl.cnf 的内容。

   在以 root 身份登录的终端中输入以下内容，得到的结果如下所示：

```
# cat /labs/repl.cnf
[mysqld1]
datadir=/var/lib/mysql1
port=3311
socket=/var/lib/mysql1/mysql.sock
server-id=1
user=mysql
log-bin=mysql1-bin
relay-log=mysql1-relay-bin
log-slave-updates
log-error=mysql1
report-host=localhost
report-port=3311
relay-log-recovery=1
master-info-repository=TABLE
relay-log-info-repository=TABLE
# gtid-mode=ON
# enforce-gtid-consistency

[mysqld2]
datadir=/var/lib/mysql2
port=3312
socket=/var/lib/mysql2/mysql.sock
server-id=2
user=mysql
log-bin=mysql2-bin
relay-log=mysql2-relay-bin
log-slave-updates
log-error=mysql2
report-host=localhost
report-port=3312
relay-log-recovery=1
master-info-repository=TABLE
relay-log-info-repository=TABLE
# gtid-mode=ON
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

```
# enforce-gtid-consistency

[mysqld3]
datadir=/var/lib/mysql3
port=3313
socket=/var/lib/mysql3/mysql.sock
server-id=3
user=mysql
log-bin=mysql3-bin
relay-log=mysql3-relay-bin
log-slave-updates
log-error=mysql3
report-host=localhost
report-port=3313
relay-log-recovery=1
master-info-repository=TABLE
relay-log-info-repository=TABLE
# gtid-mode=ON
# enforce-gtid-consistency

[mysqld4]
datadir=/var/lib/mysql4
port=3314
socket=/var/lib/mysql4/mysql.sock
server-id=4
user=mysql
log-bin=mysql4-bin
relay-log=mysql4-relay-bin
log-slave-updates
log-error=mysql4
report-host=localhost
report-port=3314
relay-log-recovery=1
master-info-repository=TABLE
relay-log-info-repository=TABLE
# gtid-mode=ON
# enforce-gtid-consistency
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

3. 使用 mysqld_multi，启动 /labs/repl.cnf 中定义的四个服务器。

   在以 root 身份登录的终端中输入以下内容，得到的结果如下所示：

```
# mysqld_multi --defaults-file=/labs/repl.cnf start 1-4

Installing new database in /var/lib/mysql1

Installing MySQL system tables...2012-12-16 13:08:17 0 [Warning]
TIMESTAMP with implicit DEFAULT value is deprecated. Please use --
explicit_defaults_for_timestamp server option (see documentation for
more details).
2012-12-16 13:08:17 6876 [Note] InnoDB: The InnoDB memory heap is
disabled
2012-12-16 13:08:17 6876 [Note] InnoDB: Mutexes and rw_locks use GCC
atomic builtins
2012-12-16 13:08:17 6876 [Note] InnoDB: Compressed tables use zlib
1.2.3
2012-12-16 13:08:17 6876 [Note] InnoDB: CPU does not support crc32
instructions
2012-12-16 13:08:17 6876 [Note] InnoDB: Using Linux native AIO
2012-12-16 13:08:17 6876 [Note] InnoDB: Initializing buffer pool, size
= 128.0M
2012-12-16 13:08:17 6876 [Note] InnoDB: Completed initialization of
buffer pool
2012-12-16 13:08:17 6876 [Note] InnoDB: The first specified data
file ./ibdata1 did not exist: a new database to be created!
...
```

   注：此命令使用 /labs/repl.cnf 中存储的设置启动四个服务器实例。服务器第一次启动时，进程创建在每个服务器的选项中指定的数据目录。

4. 在新的终端窗口中，使用 mysql 客户机以 root 身份连接第一个服务器，并将提示符设置为“1>”。**注：**为了培训环境简单起见，这四个服务器没有 root 口令。

   在新的终端中输入以下内容，得到的结果如下所示：

```
$ mysql -uroot -h127.0.0.1 -P3311
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 1
...
mysql> PROMPT 1> ;
PROMPT set to '1> '
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

5. 执行查询来查找第一个服务器的日志坐标。

   在前面步骤中创建的 mysql 提示符下，输入以下命令，得到的结果如下所示：

```
1> SHOW MASTER STATUS \G
*************************** 1. row ***************************
             File: mysql-bin.000001
         Position: 120
     Binlog_Do_DB:
 Binlog_Ignore_DB:
Executed_Gtid_Set:
1 row in set (0.00 sec)
```

   请注意，日志文件为 mysql-bin.000001，日志位置为 120。

6. 在第一个服务器上，创建名为 repl 的用户，口令为 oracle，并授予该用户 REPLICATION SLAVE 权限。

   在前面步骤中使用的 mysql 提示符下，输入以下命令，得到的结果如下所示：

```
1> CREATE USER 'repl'@'127.0.0.1' IDENTIFIED BY 'oracle';
Query OK, 0 rows affected (0.00 sec)

1> GRANT REPLICATION SLAVE ON *.* TO 'repl'@'127.0.0.1';
Query OK, 0 rows affected (0.00 sec)
```

7. 在第一个服务器上，通过脚本创建 world_innodb 数据库，如您在“系统管理”一课的练习中所做的那样。

   在前面步骤中使用的 mysql 提示符下，输入以下命令，得到的结果如下所示：

```
1> CREATE DATABASE world_innodb;
Query OK, 1 row affected (0.00 sec)

1> USE world_innodb
Database changed
1> SET autocommit=0;
Query OK, 0 rows affected (0.00 sec)

1> SOURCE /labs/world_innodb.sql
Query OK, 1 row affected (0.00 sec)

Query OK, 1 row affected (0.00 sec)
...
1> SET autocommit=1;
Query OK, 0 rows affected (0.06 sec)
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

8. 在新终端窗口中，使用 mysql 客户机以 root 身份连接第二个服务器，并将提示符设置为“2>”。

   在新的终端中输入以下内容，得到的结果如下所示：

```
$ mysql -uroot -h127.0.0.1 -P3312
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 1
...
mysql> PROMPT 2> ;
PROMPT set to '2> '
```

9. 在第二个 mysql 窗口中，发出 CHANGE MASTER TO… 命令将第二个服务器配置为第一个服务器的从属服务器，使用步骤 5 中指出的日志坐标和步骤 6 中创建的用户。显示生成的任何警告的文本。

   在前面步骤中创建的 mysql 提示符下，输入以下命令，得到的结果如下所示：

```
2> CHANGE MASTER TO
    -> MASTER_HOST='127.0.0.1',
    -> MASTER_PORT=3311,
    -> MASTER_LOG_FILE='mysql1-bin.000001',
    -> MASTER_LOG_POS=120,
    -> MASTER_USER='repl',
    -> MASTER_PASSWORD='oracle';
Query OK, 0 rows affected, 2 warnings (0.51 sec)

2> SHOW WARNINGS\G
*************************** 1. row ***************************
  Level: Note
   Code: 1759
Message: Sending passwords in plain text without SSL/TLS is extremely
insecure.
*************************** 2. row ***************************
  Level: Note
   Code: 1760
Message: Storing MySQL user name or password information in the
master.info repository is not secure and is therefore not recommended.
Please see the MySQL Manual for more about this issue and possible
alternatives.
2 rows in set (0.00 sec)
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

10. 显示第二个服务器上存在的数据库。

    在前面步骤中使用的 mysql 提示符下，输入以下命令，得到的结果如下所示：

```
2> SHOW DATABASES;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| mysql              |
| performance_schema |
| test               |
+--------------------+
4 rows in set (0.00 sec)
```

11. 在第二个服务器上启动从属服务器线程。

    在前面步骤中使用的 mysql 提示符下，输入以下命令，得到的结果如下所示：

```
2> START SLAVE;
Query OK, 0 rows affected (0.05 sec)
```

12. 显示第一个和第二个服务器上的进程。

    a. 在第一个 mysql 提示符下，发出以下命令，得到的结果如下所示：

```
1> SHOW PROCESSLIST\G
*************************** 1. row ***************************
     Id: 1
   User: root
   Host: localhost:35112
     db: world_innodb
Command: Query
   Time: 0
  State: init
   Info: SHOW PROCESSLIST
*************************** 2. row ***************************
     Id: 2
   User: repl
   Host: localhost:35115
     db: NULL
Command: Binlog Dump
   Time: 1944
  State: Master has sent all binlog to slave; waiting for binlog to be
updated
   Info: NULL
2 rows in set (0.00 sec)
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

b. 在第二个 mysql 提示符下，发出以下命令，得到的结果如下所示：

```
2> SHOW PROCESSLIST\G
*************************** 1. row ***************************
     Id: 1
   User: root
   Host: localhost:39075
     db: NULL
Command: Query
   Time: 0
  State: init
   Info: SHOW PROCESSLIST
*************************** 2. row ***************************
     Id: 2
   User: system user
   Host:
     db: NULL
Command: Connect
   Time: 1811
  State: Waiting for master to send event
   Info: NULL
*************************** 3. row ***************************
     Id: 3
   User: system user
   Host:
     db: NULL
Command: Connect
   Time: 2311
  State: Slave has read all relay log; waiting for the slave I/O
thread to update it
   Info: NULL
3 rows in set (0.00 sec)
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

13. 再次显示第二个服务器上的数据库。注意差别。

在前面步骤中使用的 mysql 提示符下，输入以下命令，得到的结果如下所示：

```
2> SHOW DATABASES;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| mysql              |
| performance_schema |
| test               |
| world_innodb       |
+--------------------+
5 rows in set (0.00 sec)
```

world_innodb 数据库位于列表中，因为它是在您在步骤 5 中记下日志坐标之后在第一个服务器上创建的。第二个服务器上的从属服务器进程复制这些坐标之后的每个更改，包括 world_innodb 数据库中的所有数据以及在步骤 6 中创建的用户。

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

练习 17-3：添加新从属服务器

概览

在本练习中，您将新服务器置备为现有从属服务器的新从属服务器（使用 `mysqldump`），更改主服务器上的一些数据，并看到其复制到两个从属服务器。

假设

您已成功执行了练习 17-2 中的步骤。

持续时间

完成本练习大约需要 10 分钟。

任务

1. 使用 `mysqldump`，备份第二个服务器上的 `world_innodb` 数据库（包括创建和使用数据库所需的信息），并配置从属服务器。
2. 编辑前面步骤中创建的备份文件以更改 `CHANGE MASTER TO …` 行，从而其指向第二个服务器并且被取消注释。
3. 使用 `mysql` 客户机连接到第三个服务器，并将提示符设置为“`3>`”。
4. 将在步骤 1 中执行的并在步骤 2 中更改的备份应用于第三个服务器。
5. 在第三个服务器上启动从属服务器并显示从属服务器状态。
6. 从第一个服务器上的 `City` 表中删除 ID 大于 4070 的所有行，并确保更改复制到第二个和第三个服务器。
7. 为了准备下面的练习，在第三个服务器上创建名为 `repl` 的用户，使用口令 `oracle`，并授予该用户 `REPLICATION SLAVE` 权限。

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

练习解答 17-3：添加新从属服务器

任务

1. 使用 mysqldump，备份第二个服务器上的 world_innodb 数据库（包括创建和使用数据库所需的信息），并配置从属服务器。

   在终端窗口中输入以下内容，得到的结果如下所示：

```
$ mysqldump -uroot -h127.0.0.1 -P3312 --master-data=2 \
     -B world_innodb > /tmp/server2.sql
```

2. 编辑前面步骤中创建的备份文件以更改 CHANGE MASTER TO … 行，从而其指向第二个服务器并且被取消注释。

   a. 使用 gedit 等编辑器，打开 /tmp/server2.sql 文件并找到以下行：

```
-- CHANGE MASTER TO MASTER_LOG_FILE='mysql2-bin.000001',
MASTER_LOG_POS=860071;
```

   b. 按如下所示更改该行：

```
CHANGE MASTER TO MASTER_HOST='127.0.0.1', MASTER_PORT=3312,
MASTER_USER='repl', MASTER_PASSWORD='oracle',
MASTER_LOG_FILE='mysql2-bin.000001', MASTER_LOG_POS=860071;
```

   请注意，日志坐标可能与您的输出中显示的坐标不同，并且第二个服务器上的复制用户最初是在第一个服务器上创建并在前面练习中复制的。

3. 使用 mysql 客户机连接到第三个服务器，并将提示符设置为“3>”。

   在新的终端窗口中输入以下内容，得到的结果如下所示：

```
$ mysql -uroot -h127.0.0.1 -P3313
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 1
...
mysql> PROMPT 3> ;
PROMPT set to '3> '
```

4. 将在步骤 1 中执行的并在步骤 2 中更改的备份应用于第三个服务器。

   在前面步骤中创建的 mysql 提示符下，输入以下命令，得到的结果如下所示：

```
3> SOURCE /tmp/server2.sql
Query OK, 0 rows affected (0.00 sec)

Query OK, 0 rows affected (0.00 sec)

Query OK, 0 rows affected (0.00 sec)
...
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

5. 在第三个服务器上启动从属服务器并显示从属服务器状态。

   在前面步骤中使用的 mysql 提示符下，输入以下命令，得到的结果如下所示：

```
3> START SLAVE;
Query OK, 0 rows affected (0.05 sec)

3> SHOW SLAVE STATUS\G
*************************** 1. row ***************************
               Slave_IO_State: Waiting for master to send event
                  Master_Host: 127.0.0.1
                  Master_User: repl
                  Master_Port: 3312
                Connect_Retry: 60
              Master_Log_File: mysql2-bin.000001
          Read_Master_Log_Pos: 860071
               Relay_Log_File: mysql3-relay-bin.000002
                Relay_Log_Pos: 284
        Relay_Master_Log_File: mysql2-bin.000001
             Slave_IO_Running: Yes
            Slave_SQL_Running: Yes
              Replicate_Do_DB:
...
```

6. 从第一个服务器上的 City 表中删除 ID 大于 4070 的所有行，并确保更改复制到第二个和第三个服务器。

   a. 在第一个服务器上，输入以下命令，得到的结果如下所示：

```
1> DELETE FROM world_innodb.City WHERE ID > 4070;
Query OK, 9 rows affected (0.05 sec)
```

   b. 在第二个服务器上，输入以下命令，得到的结果如下所示：

```
2> SELECT * FROM world_innodb.City ORDER BY Id DESC LIMIT 5;
+------+-----------------+-------------+----------------+------------+
| ID   | Name            | CountryCode | District       | Population |
+------+-----------------+-------------+----------------+------------+
| 4070 | Chitungwiza     | ZWE         | Harare         |     274912 |
| 4069 | Bulawayo        | ZWE         | Bulawayo       |     621742 |
| 4068 | Harare          | ZWE         | Harare         |    1410000 |
| 4067 | Charlotte Amalie | VIR        | St Thomas      |      13000 |
| 4066 | Charleston      | USA         | South Carolina |      89063 |
+------+-----------------+-------------+----------------+------------+
5 rows in set (0.00 sec)
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

c. 在第三个服务器上，输入以下命令，得到的结果如下所示：

```
3> SELECT * FROM world_innodb.City ORDER BY Id DESC LIMIT 5;
+------+------------------+-------------+----------------+------------+
| ID   | Name             | CountryCode | District       | Population |
+------+------------------+-------------+----------------+------------+
| 4070 | Chitungwiza      | ZWE         | Harare         |     274912 |
| 4069 | Bulawayo         | ZWE         | Bulawayo       |     621742 |
| 4068 | Harare           | ZWE         | Harare         |    1410000 |
| 4067 | Charlotte Amalie | VIR         | St Thomas      |      13000 |
| 4066 | Charleston       | USA         | South Carolina |      89063 |
+------+------------------+-------------+----------------+------------+
5 rows in set (0.00 sec)
```

7. 为了准备下面的练习，在第三个服务器上创建名为 repl 的用户，使用口令 oracle，并授予该用户 REPLICATION SLAVE 权限。

在前面步骤中使用的 mysql 提示符下，输入以下命令，得到的结果如下所示：

```
3> CREATE USER 'repl'@'127.0.0.1' IDENTIFIED BY 'oracle';
Query OK, 0 rows affected (0.00 sec)

3> GRANT REPLICATION SLAVE ON *.* TO 'repl'@'127.0.0.1';
Query OK, 0 rows affected (0.00 sec)
```

此时，repl 用户存在于三个服务器上，其在第一个服务器上创建，复制到第二个服务器，最后在第三个服务器上创建。

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

练习 17-4：启用 GTID 并配置循环复制

概览

在本练习中，您将在三个服务器上启用 GTID，连接到主服务器，从而其变为第二个从属服务器的从属服务器，并通过更改一些数据来测试新创建的循环拓扑。

假设

您已成功执行了练习 17-3 中的步骤。

持续时间

完成本练习大约需要 10 分钟。

任务

1. 使用 `mysqld_multi`，停止所有四个服务器。
2. 编辑 `/labs/repl.cnf` 文件以对所有带注释的行取消注释。
3. 使用 `mysqld_multi`，启动所有四个服务器。
4. 停止从属服务器。
5. 在每个服务器上发出 `RESET MASTER` 命令，从而日志文件仅包含使用 GTID 的事件。
6. 在从属服务器上发出适当的 `CHANGE MASTER TO…` 命令以使用 GTID 复制协议。
7. 重新启动从属服务器。
8. 从第一个服务器上的 `City` 表中删除 ID 大于 4060 的所有行，并确保更改复制到第二个和第三个服务器。
9. 记下第一个、第二和第三个服务器的服务器 UUID。
10. 在第三个服务器上，查看从属服务器状态。
11. 在第一个服务器上发出适当 `CHANGE MASTER TO…` 语句，将其配置为第三个服务器的从属服务器，并启动从属服务器线程。
12. 从第二个服务器上的 `City` 表中删除 ID 大于 4050 的所有行，并确保更改复制到第一个和第三个服务器。

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

练习解答 17-4：启用 GTID 并配置循环复制

任务

注：在以 root 身份登录的提示符下执行本练习中的所有 Linux 终端命令。

1. 使用 mysqld_multi，停止所有四个服务器。

   在终端窗口中输入以下内容：

```
# mysqld_multi --defaults-file=/labs/repl.cnf --user=root stop 1-4
```

2. 编辑 /labs/repl.cnf 文件以对所有带注释的行取消注释。

   a. 使用 gedit 等编辑器，打开 /labs/repl.cnf 文件并查找所有带注释的行：

```
# gtid-mode=ON
# enforce-gtid-consistency
```

   b. 按如下所示删除注释：

```
gtid-mode=ON
enforce-gtid-consistency
```

   c. 对所有此类带注释的行重复（每个服务器选项组一对），然后保存并关闭该文件。

3. 使用 mysqld_multi，启动所有四个服务器。

   在终端窗口中输入以下内容：

```
# mysqld_multi --defaults-file=/labs/repl.cnf start 1-4
```

   请注意，该命令不创建数据目录（如前面的练习那样），因为这些目录已经存在。

4. 停止从属服务器。

   a. 在第二个服务器上，输入以下命令，得到的结果如下所示：

```
2> STATUS
ERROR 2013 (HY000): Lost connection to MySQL server during query
2> STOP SLAVE;
ERROR 2006 (HY000): MySQL server has gone away
No connection. Trying to reconnect...
Connection id:    2
Current database: world_innodb

Query OK, 0 rows affected (0.15 sec)
```

   b. 在第三个服务器上，输入以下命令，得到的结果如下所示：

```
3> STATUS
ERROR 2013 (HY000): Lost connection to MySQL server during query
3> STOP SLAVE;
ERROR 2006 (HY000): MySQL server has gone away
No connection. Trying to reconnect...
Connection id:    2
Current database: world_innodb

Query OK, 0 rows affected (0.12 sec)
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

5. 在每个服务器上发出 RESET MASTER 命令，从而日志文件仅包含使用 GTID 的事件。
   a. 在第一个服务器连接中，发出任何命令（例如 STATUS）来重新连接，然后重置主服务器设置，如下所示：

```
1> STATUS
ERROR 2013 (HY000): Lost connection to MySQL server during query
1> RESET MASTER;
ERROR 2006 (HY000): MySQL server has gone away
No connection. Trying to reconnect...
Connection id:    2
Current database: world_innodb

Query OK, 0 rows affected (0.21 sec)
```

   b. 在第二个服务器连接中，发出以下命令，得到的结果如下所示：

```
2> RESET MASTER;
Query OK, 0 rows affected (0.21 sec)
```

   c. 在第三个服务器连接中，发出以下命令，得到的结果如下所示：

```
3> RESET MASTER;
Query OK, 0 rows affected (0.22 sec)
```

6. 在从属服务器上发出适当的 CHANGE MASTER TO… 命令以使用 GTID 复制协议。
   a. 在第二个服务器连接中，发出以下命令，得到的结果如下所示：

```
2> CHANGE MASTER TO MASTER_AUTO_POSITION=1;
Query OK, 0 rows affected (0.24 sec)
```

   b. 在第三个服务器连接中，发出以下命令，得到的结果如下所示：

```
3> CHANGE MASTER TO MASTER_AUTO_POSITION=1;
Query OK, 0 rows affected (0.25 sec)
```

7. 重新启动从属服务器。
   a. 在连接到第二个服务器的 mysql 提示符下，输入以下命令，得到的结果如下所示：

```
2> START SLAVE;
Query OK, 0 rows affected (0.05 sec)
```

   b. 在连接到第三个服务器的 mysql 提示符下，输入以下命令，得到的结果如下所示：

```
3> START SLAVE;
Query OK, 0 rows affected (0.04 sec)
```

8. 从第一个服务器上的 City 表中删除 ID 大于 4060 的所有行，并确保更改复制到第二个和第三个服务器。
   a. 在第一个服务器上，输入以下命令，得到的结果如下所示：

```
1> DELETE FROM world_innodb.City WHERE ID > 4060;
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

b. 在第二个服务器上，输入以下命令，得到的结果如下所示：

```
2> SELECT * FROM world_innodb.City ORDER BY Id DESC LIMIT 5;
+------+--------------+-------------+----------------+------------+
| ID   | Name         | CountryCode | District       | Population |
+------+--------------+-------------+----------------+------------+
| 4060 | Santa Monica | USA         | California     |      91084 |
| 4059 | Cary         | USA         | North Carolina |      91213 |
| 4058 | Boulder      | USA         | Colorado       |      91238 |
| 4057 | Visalia      | USA         | California     |      91762 |
| 4056 | San Mateo    | USA         | California     |      91799 |
+------+--------------+-------------+----------------+------------+
5 rows in set (0.00 sec)
```

c. 在第三个服务器上，输入以下命令，得到的结果如下所示：

```
3> SELECT * FROM world_innodb.City ORDER BY Id DESC LIMIT 5;
+------+--------------+-------------+----------------+------------+
| ID   | Name         | CountryCode | District       | Population |
+------+--------------+-------------+----------------+------------+
| 4060 | Santa Monica | USA         | California     |      91084 |
| 4059 | Cary         | USA         | North Carolina |      91213 |
| 4058 | Boulder      | USA         | Colorado       |      91238 |
| 4057 | Visalia      | USA         | California     |      91762 |
| 4056 | San Mateo    | USA         | California     |      91799 |
+------+--------------+-------------+----------------+------------+
5 rows in set (0.00 sec)
```

9. 记下第一个、第二和第三个服务器的服务器 UUID。

a. 在第一个服务器上，输入以下命令，得到的结果如下所示：

```
1> SELECT @@server_uuid;
+--------------------------------------+
| @@server_uuid                        |
+--------------------------------------+
| ba1e7829-7416-11e2-b285-0019b944b7f7 |
+--------------------------------------+
1 row in set (0.00 sec)
```

b. 在第二个服务器上，输入以下命令，得到的结果如下所示：

```
2> SELECT @@server_uuid;
+--------------------------------------+
| @@server_uuid                        |
+--------------------------------------+
| c701d6c5-7416-11e2-b285-0019b944b7f7 |
+--------------------------------------+
1 row in set (0.00 sec)
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

c. 在第三个服务器上，输入以下命令，得到的结果如下所示：

```
3> SELECT @@server_uuid;
+--------------------------------------+
| @@server_uuid                        |
+--------------------------------------+
| d3f1fc98-7416-11e2-b286-0019b944b7f7 |
+--------------------------------------+
1 row in set (0.00 sec)
```

注：您的 UUID 不同于显示的那些 UUID；根据设计，服务器 UUID 是唯一的。

10. 在第三个服务器上，查看从属服务器状态。

在第三个服务器上，输入以下命令，得到的结果如下所示：

```
3> SHOW SLAVE STATUS\G
*************************** 1. row ***************************
               Slave_IO_State: Waiting for master to send event
                  Master_Host: 127.0.0.1
                  Master_User: repl
                  Master_Port: 3312
                Connect_Retry: 60
              Master_Log_File: mysql2-bin.000001
          Read_Master_Log_Pos: 460
               Relay_Log_File: mysql3-relay-bin.000002
                Relay_Log_Pos: 672
        Relay_Master_Log_File: mysql2-bin.000001
             Slave_IO_Running: Yes
            Slave_SQL_Running: Yes
              Replicate_Do_DB:
          Replicate_Ignore_DB:
           Replicate_Do_Table:
       Replicate_Ignore_Table:
      Replicate_Wild_Do_Table:
  Replicate_Wild_Ignore_Table:
                   Last_Errno: 0
                   Last_Error:
                 Skip_Counter: 0
          Exec_Master_Log_Pos: 460
              Relay_Log_Space: 877
              Until_Condition: None
               Until_Log_File:
                Until_Log_Pos: 0
           Master_SSL_Allowed: No
           Master_SSL_CA_File:
           Master_SSL_CA_Path:
              Master_SSL_Cert:
            Master_SSL_Cipher:
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

```
               Master_SSL_Key:
        Seconds_Behind_Master: 0
Master_SSL_Verify_Server_Cert: No
                Last_IO_Errno: 0
                Last_IO_Error:
               Last_SQL_Errno: 0
               Last_SQL_Error:
  Replicate_Ignore_Server_Ids:
             Master_Server_Id: 2
                  Master_UUID: c701d6c5-7416-11e2-b285-0019b944b7f7
             Master_Info_File: mysql.slave_master_info
                    SQL_Delay: 0
          SQL_Remaining_Delay: NULL
      Slave_SQL_Running_State: Slave has read all relay log; waiting
for the slave I/O thread to update it
           Master_Retry_Count: 86400
                  Master_Bind:
      Last_IO_Error_Timestamp:
     Last_SQL_Error_Timestamp:
               Master_SSL_Crl:
           Master_SSL_Crlpath:
           Retrieved_Gtid_Set: ba1e7829-7416-11e2-b285-0019b944b7f7:1
            Executed_Gtid_Set: ba1e7829-7416-11e2-b285-0019b944b7f7:1
                Auto_Position: 1
1 row in set (0.00 sec)
```

请注意，主服务器 UUID 标识第二个服务器，但是步骤 8 中数据修改的 GTID 标识第一个服务器；事务是全局标识的。

11. 在第一个服务器上发出适当 CHANGE MASTER TO… 语句，将其配置为第三个服务器的从属服务器，并启动从属服务器线程。

    在第一个服务器上，输入以下命令，得到的结果如下所示：

```
1> CHANGE MASTER TO MASTER_HOST='127.0.0.1', MASTER_PORT=3313,
    > MASTER_USER='repl', MASTER_PASSWORD='oracle',
    > MASTER_AUTO_POSITION=1;
Query OK, 0 rows affected, 2 warnings (0.59 sec)
1> START SLAVE;
Query OK, 0 rows affected (0.04 sec)
```

12. 从第二个服务器上的 City 表中删除 ID 大于 4050 的所有行，并确保更改复制到第一个和第三个服务器。

    a. 在第二个服务器上，输入以下命令，得到的结果如下所示：

```
2> DELETE FROM world_innodb.City WHERE ID > 4050;
Query OK, 10 rows affected (0.24 sec)
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

b. 在第一个服务器上，输入以下命令，得到的结果如下所示：

```
1> SELECT * FROM world_innodb.City ORDER BY Id DESC LIMIT 5;
+------+----------+-------------+---------------+------------+
| ID   | Name     | CountryCode | District      | Population |
+------+----------+-------------+---------------+------------+
| 4050 | Roanoke  | USA         | Virginia      |      93357 |
| 4049 | Brockton | USA         | Massachusetts |      93653 |
| 4048 | Albany   | USA         | New York      |      93994 |
| 4047 | Richmond | USA         | California    |      94100 |
| 4046 | Norman   | USA         | Oklahoma      |      94193 |
+------+----------+-------------+---------------+------------+
5 rows in set (0.00 sec)
```

c. 在第三个服务器上，输入以下命令，得到的结果如下所示：

```
3> SELECT * FROM world_innodb.City ORDER BY Id DESC LIMIT 5;
+------+----------+-------------+---------------+------------+
| ID   | Name     | CountryCode | District      | Population |
+------+----------+-------------+---------------+------------+
| 4050 | Roanoke  | USA         | Virginia      |      93357 |
| 4049 | Brockton | USA         | Massachusetts |      93653 |
| 4048 | Albany   | USA         | New York      |      93994 |
| 4047 | Richmond | USA         | California    |      94100 |
| 4046 | Norman   | USA         | Oklahoma      |      94193 |
+------+----------+-------------+---------------+------------+
5 rows in set (0.00 sec)
```

循环复制有效，第二个服务器上的更改通过第三个服务器复制回第一个服务器。

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

练习 17-5：使用 MySQL 实用程序并执行故障转移

概览

在本练习中，您将使用 MySQL 实用程序置备新从属服务器，模拟服务器故障并执行自动故障转移。

假设

您已成功执行了练习 17-4 中的步骤。

持续时间

本练习应该需要大约五分钟来完成。

任务

1. 在第一个服务器上，停止并重置从属服务器。
2. 使用 `mysql` 客户机连接到第四个服务器，并将提示符设置为“`4>`”。
3. 将前面练习中执行的备份（保存到 `/tmp/server2.sql`）应用于第四个服务器。
4. 发出以下命令来比较服务器一和四中的数据库，将输出保存到 `/tmp/diff.sql`。

```
/usr/share/mysql-workbench/python/mysqldbcompare \
     --server1=root@127.0.0.1:3311 \
     --server2=root@127.0.0.1:3314 --changes-for=server2 \
     --difftype=sql -a world_innodb:world_innodb > /tmp/diff.sql
```

5. 查看 `/tmp/diff.sql` 的内容。
6. 对第四个服务器上的文件 `/tmp/diff.sql` 进行溯源，使其与第一个服务器保持同步。
7. 在第四个服务器上，更改主服务器设置来指向第一个服务器并使用 GTID，然后启动从属服务器。
8. 在终端窗口中，发出以下命令来显示复制拓扑。

```
/usr/share/mysql-workbench/python/mysqlrplshow \
     --master=root@127.0.0.1:3311 \
     --discover-slaves-login=root --recurse
```

9. 为了准备自动故障转移，在第四个服务器上创建 `repl` 用户，使用口令 `oracle`，并授予该用户 `REPLICATION SLAVE` 权限。
10. 使用主机 `localhost`（而不是回送地址）重复前面的步骤。
11. 通过使用以下命令在 `auto` 模式下以交互方式启动 `mysqlfailover`。

```
/usr/share/mysql-workbench/python/mysqlfailover \
     --master=root@127.0.0.1:3311 \
     --discover-slaves-login=root --rpl-user=repl:oracle@127.0.0.1
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

12. 通过按 G 和 U 键查看不同屏幕。
13. 从单独的终端窗口关闭第一个服务器。
14. 返回包含 mysqlfailover 进程的终端，并观察执行自动故障转移的工具。
15. 发出 mysqlrplshow 命令来显示复制拓扑，第四个服务器作为主服务器。
16. 停止本课中使用的所有四个服务器，并启动正常 mysql 服务。

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

练习解答 17-5：使用 MySQL 实用程序并执行故障转移

任务

1. 在第一个服务器上，停止并重置从属服务器。

   在第一个 mysql 提示符下输入以下内容，得到的结果如下所示：

```
1> STOP SLAVE;
Query OK, 0 rows affected (0.10 sec)

1> RESET SLAVE ALL;
Query OK, 0 rows affected (0.32 sec)
```

2. 使用 mysql 客户机连接到第四个服务器，并将提示符设置为“4>”。

   在新的终端窗口中输入以下内容，得到的结果如下所示：

```
$ mysql -uroot -h127.0.0.1 -P3314
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 1
...
mysql> PROMPT 4> ;
PROMPT set to '4> '
```

3. 将前面练习中执行的备份（保存到 /tmp/server2.sql）应用于第四个服务器。

   在前面步骤中创建的 mysql 提示符下，输入以下命令，得到的结果如下所示：

```
4> SOURCE /tmp/server2.sql
Query OK, 0 rows affected (0.00 sec)

Query OK, 0 rows affected (0.00 sec)

Query OK, 0 rows affected (0.00 sec)

Query OK, 0 rows affected (0.00 sec)
...
```

   请注意，此备份包括 CHANGE MASTER TO… 语句，它指定日志坐标，而非 GTID。

4. 发出以下命令来比较服务器一和四中的数据库，将输出保存到 /tmp/diff.sql。

   在终端窗口中输入以下内容：

```
$ /usr/share/mysql-workbench/python/mysqldbcompare \
     --server1=root@127.0.0.1:3311 \
     --server2=root@127.0.0.1:3314 --changes-for=server2 \
     --difftype=sql -a world_innodb:world_innodb > /tmp/diff.sql
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

5. 查看 /tmp/diff.sql 的内容。

在终端窗口中输入以下内容，得到的结果如下所示：

```
$ cat /tmp/diff.sql
# server1 on 127.0.0.1: ... connected.
# server2 on 127.0.0.1: ... connected.
# Checking databases world_innodb on server1 and world_innodb on
server2
#
#                                                    Defn    Row
Data
# Type      Object Name                              Diff    Count
Check
# ------------------------------------------------------------------------
-----
# TABLE     City                                     FAIL    FAIL
FAIL
#
# Transformation for --changes-for=server2:
#

ALTER TABLE world_innodb.City
  DROP INDEX CountryCode,
  DROP PRIMARY KEY,
  ADD PRIMARY KEY(ID),
  ADD INDEX CountryCode (CountryCode),
AUTO_INCREMENT=4071;

# Row counts are not the same among world_innodb.City and
world_innodb.City.
#
# Transformation for --changes-for=server2:
#

# Data differences found among rows:
UPDATE world_innodb.City  WHERE ID = '694';
UPDATE world_innodb.City SET ID = '785', Name = 'Pasay', CountryCode =
'PHL', District = 'National Capital Reg', Population = '354908' WHERE
ID = '785';
DELETE FROM world_innodb.City WHERE ID = '4079';
...
DELETE FROM world_innodb.City WHERE ID = '4068';



# TABLE     Country                                  pass    pass
pass
# TABLE     CountryLanguage                          pass    pass
pass
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

```
# Database consistency check failed.
#
# ...done
```

6. 对第四个服务器上的文件 /tmp/diff.sql 进行溯源，使其与第一个服务器保持同步。
   在连接到第四个服务器的 mysql 提示符下，输入以下命令，得到的结果如下所示：

```
4> SOURCE /tmp/diff.sql
Query OK, 0 rows affected (0.04 sec)
Records: 0  Duplicates: 0  Warnings: 0

ERROR 1064 (42000): You have an error in your SQL syntax; check the
manual that corresponds to your MySQL server version for the right
syntax to use near 'WHERE ID = '694'' at line 1
Query OK, 0 rows affected (0.06 sec)
Rows matched: 1  Changed: 0  Warnings: 0

Query OK, 1 row affected (0.05 sec)

...
Query OK, 1 row affected (0.05 sec)
```

   前面输出中的错误是由于 mysqldbcompare 输出中的语法错误。

7. 在第四个服务器上，更改主服务器设置来指向第一个服务器并使用 GTID，然后启动从属服务器。

   a. 在连接到第四个服务器的 mysql 提示符下，输入以下命令，得到的结果如下所示：

```
4> CHANGE MASTER TO MASTER_PORT=3311, MASTER_AUTO_POSITION=1;
Query OK, 0 rows affected (0.34 sec)
4> START SLAVE;
Query OK, 0 rows affected (0.05 sec)
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

8. 在终端窗口中，发出以下命令来显示复制拓扑。

   在终端窗口中输入以下内容，得到的结果如下所示：

```
$ /usr/share/mysql-workbench/python/mysqlrplshow \
     --master=root@127.0.0.1:3311 \
     --discover-slaves-login=root --recurse
# master on 127.0.0.1: ... connected.
# Finding slaves for master: 127.0.0.1:3311
# master on localhost: ... connected.
# Finding slaves for master: localhost:3312
# master on localhost: ... connected.
# Finding slaves for master: localhost:3313
# master on localhost: ... connected.
# Finding slaves for master: localhost:3314

# Replication Topology Graph
127.0.0.1:3311 (MASTER)
   |
   +--- localhost:3312 - (SLAVE + MASTER)
   |    |
   |    +--- localhost:3313 - (SLAVE)
   |
   +--- localhost:3314 - (SLAVE)
```

   输出显示一个图表，表示从属服务器与主服务器的关系。

9. 为了准备自动故障转移，在第四个服务器上创建 repl 用户，使用口令 oracle，并授予该用户 REPLICATION SLAVE 权限。

   在前面步骤中使用的 mysql 提示符下，输入以下命令，得到的结果如下所示：

```
4> CREATE USER 'repl'@'127.0.0.1' IDENTIFIED BY 'oracle';
Query OK, 0 rows affected (0.00 sec)

4> GRANT REPLICATION SLAVE ON *.* TO 'repl'@'127.0.0.1';
Query OK, 0 rows affected (0.00 sec)
```

   此时，repl 用户存在于所有四个服务器上。

10. 使用主机 localhost（而不是回送地址）重复前面的步骤。

    在前面步骤中使用的 mysql 提示符下，输入以下命令，得到的结果如下所示：

```
4> CREATE USER 'repl'@'localhost' IDENTIFIED BY 'oracle';
Query OK, 0 rows affected (0.00 sec)

4> GRANT REPLICATION SLAVE ON *.* TO 'repl'@'localhost';
Query OK, 0 rows affected (0.00 sec)
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

11. 通过使用以下命令在 auto 模式下以交互方式启动 mysqlfailover。

a. 在终端窗口中输入以下内容，得到的结果如下所示：

```
$ /usr/share/mysql-workbench/python/mysqlfailover \
     --master=root@127.0.0.1:3311 \
     --discover-slaves-login=root --rpl-user=repl:oracle
# Discovering slaves for master at 127.0.0.1:3311
# Checking privileges.
# WARNING: You may be mixing host names and IP addresses. This may
result in negative status reporting if your DNS services do not
support reverse name lookup.
#
# Failover console will start in 10 seconds.
```

b. 控制台显示时，其显示内容类似如下：

```
MySQL Replication Failover Utility
Failover Mode = auto     Next Interval = Mon Feb 11 07:55:33 2013

Master Information
------------------
Binary Log File    Position  Binlog_Do_DB  Binlog_Ignore_DB
mysql1-bin.000002  679

GTID Executed Set
ba1e7829-7416-11e2-b285-0019b944b7f7:1-4 [...]

Replication Health Status
+------------+-------+---------+--------+------------+---------+
| host       | port  | role    | state  | gtid_mode  | health  |
+------------+-------+---------+--------+------------+---------+
| 127.0.0.1  | 3311  | MASTER  | UP     | ON         | OK      |
| localhost  | 3312  | SLAVE   | UP     | ON         | OK      |
| localhost  | 3314  | SLAVE   | UP     | ON         | OK      |
+------------+-------+---------+--------+------------+---------+
Q-quit R-refresh H-health G-GTID Lists U-UUIDs
```

请注意，该显示内容不显示循环，也不递归。它仅显示命令行上提供的主服务器的从属服务器。

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

12. 通过按 G 和 U 键查看不同屏幕。

    a. 按 G 键查看“Master GTID Executed Set（执行的主服务器 GTID 集）”，得到的结果如下所示：

```
MySQL Replication Failover Utility
Failover Mode = auto     Next Interval = Mon Feb 11 07:57:03 2013

Master Information
------------------
Binary Log File    Position  Binlog_Do_DB  Binlog_Ignore_DB
mysql1-bin.000002  679

GTID Executed Set
ba1e7829-7416-11e2-b285-0019b944b7f7:1-4 [...]

Master GTID Executed Set
+-----------------------------------------+
| gtid                                    |
+-----------------------------------------+
| ba1e7829-7416-11e2-b285-0019b944b7f7:1-4  |
| c701d6c5-7416-11e2-b285-0019b944b7f7:1    |
+-----------------------------------------+
Q-quit R-refresh H-health G-GTID Lists U-UUIDs
```

    b. 再次按 G 键查看“Transactions executed on the servers（服务器上执行的事务）”，得到的结果如下所示：

```
MySQL Replication Failover Utility
Failover Mode = auto     Next Interval = Mon Feb 11 07:57:21 2013

Master Information
------------------
Binary Log File    Position  Binlog_Do_DB  Binlog_Ignore_DB
mysql1-bin.000002  679

GTID Executed Set
ba1e7829-7416-11e2-b285-0019b944b7f7:1-4 [...]

Transactions executed on the servers:
+------------+-------+---------+-----------------------------------------+
| host       | port  | role    | gtid                                    |
+------------+-------+---------+-----------------------------------------+
| 127.0.0.1  | 3311  | MASTER  | ba1e7829-7416-11e2-b285-0019b944b7f7:1-4  |
| 127.0.0.1  | 3311  | MASTER  | c701d6c5-7416-11e2-b285-0019b944b7f7:1    |
| localhost  | 3312  | SLAVE   | ba1e7829-7416-11e2-b285-0019b944b7f7:1-4  |
| localhost  | 3312  | SLAVE   | c701d6c5-7416-11e2-b285-0019b944b7f7:1    |
| localhost  | 3314  | SLAVE   | 85bae58f-741f-11e2-b2be-0019b944b7f7:1-49 |
| localhost  | 3314  | SLAVE   | ba1e7829-7416-11e2-b285-0019b944b7f7:1-4  |
| localhost  | 3314  | SLAVE   | c701d6c5-7416-11e2-b285-0019b944b7f7:1    |
+------------+-------+---------+-----------------------------------------+
Q-quit R-refresh H-health G-GTID Lists U-UUIDs Up|Down-scroll
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

c. 再次按 G 键查看“Transactions purged from the servers（从服务器清除的事务）”，得到的结果如下所示：

```
MySQL Replication Failover Utility
Failover Mode = auto     Next Interval = Mon Feb 11 07:57:57 2013

Master Information
------------------
Binary Log File    Position  Binlog_Do_DB  Binlog_Ignore_DB
mysql1-bin.000002  679

GTID Executed Set
ba1e7829-7416-11e2-b285-0019b944b7f7:1-4 [...]

Transactions purged from the servers:
0 Rows Found.
Q-quit R-refresh H-health G-GTID Lists U-UUIDs
```

d. 再次按 G 键查看“Transactions owned by another server（其他服务器拥有的事务）”，得到的结果如下所示：

```
MySQL Replication Failover Utility
Failover Mode = auto     Next Interval = Mon Feb 11 07:58:33 2013

Master Information
------------------
Binary Log File    Position  Binlog_Do_DB  Binlog_Ignore_DB
mysql1-bin.000002  679

GTID Executed Set
ba1e7829-7416-11e2-b285-0019b944b7f7:1-4 [...]

Transactions owned by another server:
0 Rows Found.
Q-quit R-refresh H-health G-GTID Lists U-UUIDs
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

e. 按 U 键查看“UUIDs (UUID)”，得到的结果如下所示：

```
MySQL Replication Failover Utility
Failover Mode = auto     Next Interval = Mon Feb 11 07:58:51 2013

Master Information
------------------
Binary Log File    Position  Binlog_Do_DB  Binlog_Ignore_DB
mysql1-bin.000002  679

GTID Executed Set
ba1e7829-7416-11e2-b285-0019b944b7f7:1-4 [...]

UUIDs
+------------+-------+---------+--------------------------------------+
| host       | port  | role    | uuid                                 |
+------------+-------+---------+--------------------------------------+
| 127.0.0.1  | 3311  | MASTER  | ba1e7829-7416-11e2-b285-0019b944b7f7 |
| localhost  | 3312  | SLAVE   | c701d6c5-7416-11e2-b285-0019b944b7f7 |
| localhost  | 3314  | SLAVE   | 85bae58f-741f-11e2-b2be-0019b944b7f7 |
+------------+-------+---------+--------------------------------------+
Q-quit R-refresh H-health G-GTID Lists U-UUIDs
```

f. 按 H 键返回 Health 屏幕。

13. 从单独的终端窗口关闭第二个服务器。

在终端窗口中输入以下内容。

```
$ mysqladmin -uroot -h127.0.0.1 -P3311 shutdown
```

14. 返回包含 mysqlfailover 进程的终端，并观察执行自动故障转移的工具。

输出内容应类似如下：

```
Failover starting in 'auto' mode...
# Candidate slave localhost:3314 will become the new master.
# Preparing candidate for failover.
# Creating replication user if it does not exist.
# Stopping slaves.
# Performing STOP on all slaves.
# Switching slaves to new master.
# Starting slaves.
# Performing START on all slaves.
# Checking slaves for errors.
# Failover complete.
# Discovering slaves for master at localhost:3314

Failover console will restart in 5 seconds.
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

15. 发出 mysqlrplshow 命令来显示复制拓扑，第四个服务器作为主服务器。
    在终端窗口中输入以下内容，得到的结果如下所示。

```
$ /usr/share/mysql-workbench/python/mysqlrplshow \
     --master=root@127.0.0.1:3314 \
     --discover-slaves-login=root --recurse
# master on 127.0.0.1: ... connected.
# Finding slaves for master: 127.0.0.1:3314
# master on localhost: ... connected.
# Finding slaves for master: localhost:3312
# master on localhost: ... connected.
# Finding slaves for master: localhost:3313

# Replication Topology Graph
127.0.0.1:3314 (MASTER)
   |
   +--- localhost:3312 - (SLAVE + MASTER)
        |
        +--- localhost:3313 - (SLAVE)
```

16. 停止本课中使用的所有四个服务器，并启动正常 mysql 服务。
    在终端窗口中输入以下内容，得到的结果如下所示。

```
# mysqld_multi --defaults-file=/labs/repl.cnf --user=root stop 1-4
# service mysql start
Starting MySQL..                                           [  OK  ]
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

第 18 课的练习：性能调节简介

第 18 章

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

第 18 课的练习：概览

练习概览

通过这些练习，可测试您对性能调节主题的掌握程度。这些上机练习是针对 Oracle Linux 操作系统环境编写的，Oracle 课堂中将提供此环境。对于非 Oracle 课堂环境，可能需要对文件位置进行一些调整。

假设

- 在进行这些练习之前已经完成了 MySQL 服务器安装。
- 在终端窗口中以 `oracle` 用户身份登录。
- 您可以通过命令行提示符访问 `mysql` 客户机。

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

练习 18-1：测验 — 性能调节简介

概览

在本练习中，您将回答有关性能调节的问题。

持续时间

本练习应该需要大约五分钟来完成。

测验问题

请从所提供的每道多选题或正/误判断题中选择最佳答案。

1. 以下哪项不是标准化的优点？
    a. 使读取查询更为容易
    b. 消除冗余数据
    c. 最大限度地减少数据不一致情况
    d. 提供对数据的灵活访问
2. 随着数据的增长，选择大于所需空间的数据类型可能会成为一个大问题。
    a. 正确
    b. 错误
3. 查询没有索引或索引不足的表会导致 _____。
    a. 损坏数据
    b. 全表扫描
    c. 性能提高
    d. 以上都是
4. _____ 命令描述 MySQL 打算如何执行指定的 SQL 语句，但不会返回数据集的任何数据。
    a. `ANALYZE`
    b. `DESCRIBE`
    c. `EXPLAIN`
    d. `SHOW`
5. `PROCEDURE ANALYZE` 分析应用程序中的存储过程。
    a. 正确
    b. 错误
6. 哪个状态变量用于显示当前打开的连接数？
    a. `Key_reads`
    b. `Max_used_connections`
    c. `Open_tables`
    d. `Threads_connected`

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

7. 哪个服务器系统变量用于定义将 InnoDB 日志缓冲区写出到日志文件中的频率以及对日志文件执行刷新到磁盘操作的频率？
    a. `innodb_buffer_pool_size`
    b. `innodb_flush_log_at_trx_commit`
    c. `innodb_log_buffer_size`
    d. `innodb_log_file_size`

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

练习解答 18-1：测验 — 性能调节简介

测验解答

1. **a.** 使读取查询更为容易
2. **a.** 正确
3. **b.** 全表扫描
4. **c.** `EXPLAIN`
5. **b.** 错误。`PROCEDURE ANALYZE` 用于分析给定查询的列并提供有关每个字段的调节反馈。
6. **d.** `Threads_connected`
7. **b.** `innodb_flush_log_at_trx_commit`

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

练习 18-2：EXPLAIN

概览

在本练习中，您将使用 EXPLAIN 命令查看多个 SELECT 语句的执行计划。要实现此目标，请执行以下操作：

- 使用 world_innodb 数据库查看多个 SELECT 语句的执行计划。
- 更改现有表以改进查询的执行。

假设

- 已安装并且正在运行 MySQL 服务器。
- 已安装 world_innodb 数据库。

持续时间

完成本练习大约需要 10 分钟。

任务

1. 打开终端窗口，使用 admin 登录路径启动 mysql 客户机，然后使用 world_innodb 数据库。
2. 通过使用 world_innodb.sql 文件重新创建 world_innodb 数据库，使用该数据库的干净副本重新开始。
3. 执行以下命令以查看针对 world_innodb 数据库中 City 表的查询的执行计划：

```
EXPLAIN SELECT Name FROM City WHERE Name LIKE 'A%'\G
```

   - 此查询的选择类型是什么？_____
   - 是否有可供此查询使用的任何索引？_____
   - 至少需要检查多少行才能完成此查询？_____

4. 执行以下命令以查看针对 world_innodb 数据库中 City 表的另一个查询的执行计划：

```
EXPLAIN SELECT Name FROM City WHERE ID > 4070\G
```

   - 此查询的选择类型是什么？_____
   - 是否有可供此查询使用的任何索引？_____
   - 至少需要检查多少行才能完成此查询？_____

5. 比较第一个和第二个 EXPLAIN 语句的输出，确定哪个执行计划更好。

   _____

   _____

6. 通过修改 City 表提高第一个查询的性能，然后返回 EXPLAIN 语句。
   - 此查询的选择类型是什么？_____
   - 是否有可供此查询使用的任何索引？

     _____
   - 至少需要检查多少行才能完成此查询？_____

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

练习解答 18-2：EXPLAIN

任务

1. 打开终端窗口，使用 admin 登录路径启动 mysql 客户机，然后使用 world_innodb 数据库。在终端窗口中输入以下内容，得到的结果如下所示：

```
$ mysql --login-path=admin
Welcome to the MySQL monitor.  Commands end with ; or \g.
...
mysql> USE world_innodb
...
Database changed
```

2. 通过使用 world_innodb.sql 文件重新创建 world_innodb 数据库，使用该数据库的干净副本重新开始。

   在 mysql 提示符下输入以下内容，得到的结果如下所示：

```
mysql> DROP DATABASE world_innodb;
Query OK, 4 rows affected (0.00 sec)

mysql> CREATE DATABASE world_innodb;
Query OK, 1 row affected (0.00 sec)

mysql> USE world_innodb;
Database changed
mysql> SET AUTOCOMMIT=0;
Query OK, 0 rows affected (0.00 sec)

mysql> SOURCE /labs/world_innodb.sql
Query OK, 0 rows affected (0.00 sec)

Query OK, 0 rows affected (0.00 sec)
...
Query OK, 0 rows affected (0.00 sec)

Query OK, 0 rows affected (0.00 sec)

Query OK, 0 rows affected (0.00 sec)

mysql> SET AUTOCOMMIT=1;
Query OK, 0 rows affected (0.12 sec)
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

3. 执行以下命令以查看针对 world_innodb 数据库中 City 表的查询的执行计划。
   在 mysql 提示符下输入以下内容，得到的结果如下所示：

```
mysql> EXPLAIN SELECT Name FROM City WHERE Name LIKE 'A%'\G
*************************** 1. row ***************************
           id: 1
  select_type: SIMPLE
        table: City
         type: ALL
possible_keys: NULL
          key: NULL
      key_len: NULL
          ref: NULL
         rows: 4188
        Extra: Using where
1 row in set (0.00 sec)
```

   - 此查询的选择类型是什么？ALL
   - 是否有可供此查询使用的任何索引？没有
   - 至少需要检查多少行才能完成此查询？大约 3982 行。您的输出可能有所不同。表中有 4079 行，因此 EXPLAIN 将执行评估。

4. 执行以下命令以查看针对 world_innodb 数据库中 City 表的另一个查询的执行计划。
   在 mysql 提示符下输入以下内容，得到的结果如下所示：

```
mysql> EXPLAIN SELECT Name FROM City WHERE ID > 4070\G
*************************** 1. row ***************************
           id: 1
  select_type: SIMPLE
        table: City
         type: range
possible_keys: PRIMARY
          key: PRIMARY
      key_len: 4
          ref: NULL
         rows: 9
        Extra: Using where
1 row in set (0.00 sec)
```

   - 此查询的选择类型是什么？SIMPLE
   - 是否有可供此查询使用的任何索引？
     是，可以使用该表的主键。
   - 至少需要检查多少行才能完成此查询？九行。这是查询返回的实际行数。

5. 比较第一个和第二个 EXPLAIN 语句的输出，确定哪个执行计划更好。
   第二个查询利用了索引，因此不需要执行全表扫描。

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

6. 通过修改 City 表提高第一个查询的性能，然后返回 EXPLAIN 语句。

   在 mysql 提示符下输入以下语句，得到的结果如下所示：

```
mysql> CREATE INDEX iName ON City (Name) USING BTREE;
Query OK, 0 rows affected (0.12 sec)
Records: 0  Duplicates: 0  Warnings: 0

mysql> EXPLAIN SELECT Name FROM City WHERE Name LIKE 'A%'\G
*************************** 1. row ***************************
           id: 1
  select_type: SIMPLE
        table: City
         type: range
possible_keys: iName
          key: iName
      key_len: 35
          ref: NULL
         rows: 258
        Extra: Using where; Using index
1 row in set (0.00 sec)
```

   - 此查询的选择类型是什么？SIMPLE
   - 是否有可供此查询使用的任何索引？
     有。该执行计划将使用在“Name”列上创建的索引。
   - 至少需要检查多少行才能完成此查询？258

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

练习 18-3：PROCEDURE ANALYSE

概览

在本练习中，您将使用 PROCEDURE ANALYSE 选项执行多个 SELECT 语句。要实现此目标，请执行以下操作：

- 使用 world_innodb 数据库执行多个 SELECT 语句。
- 评估现有表以改进该表的设计。

假设

- 已安装并且正在运行 MySQL 服务器。
- 已安装 world_innodb 数据库。

持续时间

完成本练习大约需要 10 分钟。

任务

1. 使用上一练习中的终端（已运行 mysql 客户机并使用了 world_innodb 数据库）执行以下命令：

```
SELECT Name, CountryCode, Population FROM City PROCEDURE ANALYSE()\G
```

   - 根据表中的现有数据，PROCEDURE ANALYSE 选项如何声明才是 City 表中 Name 列的最佳字段类型？

   - 根据表中的现有数据，PROCEDURE ANALYSE 选项如何声明才是 City 表中 CountryCode 列的最佳字段类型？

   - 根据表中的现有数据，PROCEDURE ANALYSE 选项如何声明才是 City 表中 Population 列的最佳字段类型？

2. 执行以下命令以查看 City 表设计：

```
DESC City;
```

   将 City 表的设计与第 1 步中运行的 PROCEDURE ANALYSE 选项的建议进行比较，然后评估每条建议。

3. 执行以下命令以更改 PROCEDURE ANALYSE() 建议的 ENUM 类型，并将建议与之前步骤中的建议进行比较：

```
SELECT Name, CountryCode, Population FROM City
PROCEDURE ANALYSE(256,1024)\G
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

4. 执行以下命令以评估 CountryLanguage 表设计：

```
SELECT * FROM CountryLanguage PROCEDURE ANALYSE(256,1024)\G
```

5. 执行以下命令以查看 CountryLanguage 表设计：

```
DESC CountryLanguage;
```

将 CountryLanguage 表的设计与第 3 步中运行的 PROCEDURE ANALYSE 选项的建议进行比较，您认为应实施哪条建议？

_____

_____

_____

_____

6. 退出 **mysql** 客户机会话。

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

练习解答 18-3：PROCEDURE ANALYSE

任务

1. 使用上一练习中的终端（已运行 mysql 客户机并使用了 world_innodb 数据库）执行以下命令。

   在 mysql 提示符下输入以下语句，得到的结果如下所示：

```
mysql> SELECT Name, CountryCode, Population FROM City
    -> PROCEDURE ANALYSE()\G
*************************** 1. row ***************************
             Field_name: world_innodb.City.Name
              Min_value: A Coruña (La Coruña)
              Max_value: ´s-Hertogenbosch
             Min_length: 3
             Max_length: 33
       Empties_or_zeros: 0
                  Nulls: 0
Avg_value_or_avg_length: 8.5295
                    Std: NULL
      Optimal_fieldtype: VARCHAR(33) NOT NULL
*************************** 2. row ***************************
             Field_name: world_innodb.City.CountryCode
              Min_value: ABW
              Max_value: ZWE
             Min_length: 3
             Max_length: 3
       Empties_or_zeros: 0
                  Nulls: 0
Avg_value_or_avg_length: 3.0000
                    Std: NULL
      Optimal_fieldtype: ENUM('ABW','AFG',...,'ZMB','ZWE') NOT NULL
*************************** 3. row ***************************
             Field_name: world_innodb.City.Population
              Min_value: 42
              Max_value: 10500000
             Min_length: 2
             Max_length: 8
       Empties_or_zeros: 0
                  Nulls: 0
Avg_value_or_avg_length: 350468.2236
                    Std: 723686.9870
      Optimal_fieldtype: MEDIUMINT(8) UNSIGNED NOT NULL
3 rows in set (0.00 sec)
```

   – 根据表中的现有数据，PROCEDURE ANALYSE 选项如何声明才是 City 表中 Name 列的最佳字段类型？
     VARCHAR(33) NOT NULL
   – 根据表中的现有数据，PROCEDURE ANALYSE 选项如何声明才是 City 表中 CountryCode 列的最佳字段类型？ ENUM('ABW','AFG',...,'ZMB','ZWE') NOT NULL
   – 根据表中的现有数据，PROCEDURE ANALYSE 选项如何声明才是 City 表中 Population 列的最佳字段类型？ MEDIUMINT(8) UNSIGNED NOT NULL

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

2. 执行以下命令以查看 City 表设计。将 City 表的设计和第 1 步中运行的 PROCEDURE ANALYSE 选项的建议进行比较，然后评估每个建议。

   在 mysql 提示符下输入以下语句，得到的结果如下所示：

```
mysql> DESC City;
+-------------+------------+------+-----+---------+------------+
| Field       | Type       | Null | Key | Default | Extra      |
+-------------+------------+------+-----+---------+------------+
| ID          | int(11)    | NO   | PRI | NULL    | auto_i ...|
| Name        | char(35)   | NO   | MUL |         |            |
| CountryCode | char(3)    | NO   | MUL |         |            |
| District    | char(20)   | NO   |     |         |            |
| Population  | int(11)    | NO   |     | 0       |            |
+-------------+------------+------+-----+---------+------------+
5 rows in set (0.00 sec)
```

   - 将 City 表的设计与第 1 步中运行的 PROCEDURE ANALYSE 选项的建议进行比较，您认为应实施哪条建议？
     - 将 Population 列更改为 MEDIUMINT UNSIGNED 字段是适合当前数据的建议。MEDIUMINT UNSIGNED 字段可以处理的最大数值为 16,777,215，并且仅占用 3 个字节。相比之下，BIGINT UNSIGNED 字段可以处理的最大数值为 18,446,744,073,709,555,615，但要占用 8 个字节。但是，此列对于将来值的最佳字段类型是 INT UNSIGNED，因为可以处理的最大数值为 4,294,967,295 且仅占用 4 个字节。对于具有 3500 万人口的东京（截止 2013 年）等大城市，MEDIUMINT UNSIGNED 字段无法容纳现在和未来所有城市的人口。
     - 更改 Name 列的建议不会节省多少空间，当表随着新数据的出现而增长时，您应该监视该类型。
     - 不应实施将 CountryCode 列更改为 ENUM 字段类型的建议；包含大量值的 ENUM 字段会增加应用程序的复杂性且难以维护。

3. 执行以下命令以更改 PROCEDURE ANALYSE() 建议的 ENUM 类型，并将建议与之前步骤中的建议进行比较。

   在 mysql 提示符下输入以下语句，得到的结果如下所示：

```
mysql> SELECT Name, CountryCode, Population FROM City
    > PROCEDURE ANALYSE(256,1024)\G
*************************** 1. row ***************************
             Field_name: world_innodb.City.Name
              Min_value: A Coruña (La Coruña)
              Max_value: ´s-Hertogenbosch
             Min_length: 3
             Max_length: 33
       Empties_or_zeros: 0
                  Nulls: 0
Avg_value_or_avg_length: 8.5295
                    Std: NULL
      Optimal_fieldtype: VARCHAR(33) NOT NULL
*************************** 2. row ***************************
             Field_name: world_innodb.City.CountryCode
              Min_value: ABW
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

```
              Max_value: ZWE
            Min_length: 3
            Max_length: 3
       Empties_or_zeros: 0
                  Nulls: 0
Avg_value_or_avg_length: 3.0000
                    Std: NULL
      Optimal_fieldtype: CHAR(3) NOT NULL
*************************** 3. row ***************************
            Field_name: world_innodb.City.Population
             Min_value: 42
             Max_value: 10500000
            Min_length: 2
            Max_length: 8
       Empties_or_zeros: 0
                  Nulls: 0
Avg_value_or_avg_length: 350468.2236
                    Std: 723686.9870
      Optimal_fieldtype: MEDIUMINT(8) UNSIGNED NOT NULL
3 rows in set (0.01 sec)
```

对 CountryCode 的建议已更改为 CHAR(3)，这是最合适的类型。

4. 执行以下命令以评估 CountryLanguage 表设计：

```
mysql> SELECT * FROM CountryLanguage PROCEDURE ANALYSE(256,1024)\G
*************************** 1. row ***************************
            Field_name: world_innodb.CountryLanguage.CountryCode
             Min_value: ABW
             Max_value: ZWE
            Min_length: 3
            Max_length: 3
       Empties_or_zeros: 0
                  Nulls: 0
Avg_value_or_avg_length: 3.0000
                    Std: NULL
      Optimal_fieldtype: CHAR(3) NOT NULL
*************************** 2. row ***************************
            Field_name: world_innodb.CountryLanguage.Language
             Min_value: Abhyasi
             Max_value: [South]Mande
            Min_length: 2
            Max_length: 25
       Empties_or_zeros: 0
                  Nulls: 0
Avg_value_or_avg_length: 7.1606
                    Std: NULL
      Optimal_fieldtype: VARCHAR(25) NOT NULL
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

```
*************************** 3. row ***************************
             Field_name: world_innodb.CountryLanguage.IsOfficial
              Min_value: F
              Max_value: T
             Min_length: 1
             Max_length: 1
       Empties_or_zeros: 0
                  Nulls: 0
Avg_value_or_avg_length: 1.0000
                    Std: NULL
      Optimal_fieldtype: ENUM('F','T') NOT NULL
*************************** 4. row ***************************
             Field_name: world_innodb.CountryLanguage.Percentage
              Min_value: 0.1
              Max_value: 100.0
             Min_length: 3
             Max_length: 5
       Empties_or_zeros: 65
                  Nulls: 0
Avg_value_or_avg_length: 20.4
                    Std: 30.8
      Optimal_fieldtype: FLOAT(4,1) NOT NULL
4 rows in set (0.00 sec)
```

5. 执行以下命令以查看 CountryLanguage 表设计：

```
mysql> DESC CountryLanguage;
+-------------+---------------+------+-----+---------+-------+
| Field       | Type          | Null | Key | Default | Extra |
+-------------+---------------+------+-----+---------+-------+
| CountryCode | char(3)       | NO   | PRI |         |       |
| Language    | char(30)      | NO   | PRI |         |       |
| IsOfficial  | enum('T','F') | NO   |     | F       |       |
| Percentage  | float(4,1)    | NO   |     | 0.0     |       |
+-------------+---------------+------+-----+---------+-------+
4 rows in set (0.00 sec)
```

将 CountryLanguage 表的设计与第 3 步中运行的 PROCEDURE ANALYSE 选项的建议进行比较，您认为应实施哪条建议？

唯一更改的建议是将语言名称从 CHAR(30) 设置为 VARCHAR(25)。这是一个很好的建议，因为 VARCHAR 更节省空间，但您应该确保该列的所有未来值均符合 25 个字符的限制。

6. 退出 **mysql** 客户机会话：

```
mysql> EXIT;
Bye
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

第 19 课的练习：总结

第 19 章

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

第 19 课的练习

课程概览

本课没有练习。

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

附录：EXPLAIN 输出列

第 20 章

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

EXPLAIN 语句

EXPLAIN 语句针对所分析语句中每个 **SELECT** 所指定的每个表生成一行输出，如果使用了子查询或 **UNION**，则一条语句可以有多个 **SELECT**。要有效使用 **EXPLAIN**，有必要了解输出的每个行中各列的含义：

- **id**：指示 **EXPLAIN** 输出行所涉及的分析语句中的 **SELECT**。
- **select_type**：对输出行所涉及的 **SELECT** 进行分类。该列可以使用下表所示的任一值。单词 **DEPENDENT** 指示子查询关联有外部查询。

select_type 值	含义
SIMPLE	简单 SELECT 语句（无子查询或联合）
PRIMARY	外部 SELECT 语句
UNION	联合中的第二个（或后面的）SELECT
DEPENDENT UNION	联合中的第二个（或后面的）SELECT
UNION RESULT	联合中的第二个（或后面的）SELECT
SUBQUERY	子查询中的第一个 SELECT
DEPENDENT SUBQUERY	依赖于外部子查询的子查询中的第一个 SELECT
DERIVED	WHERE 子句中的子查询
MATERIALIZED	实体化子查询
UNCACHEABLE SUBQUERY	无法缓存子查询结果，必须针对外部查询的每一行重新计算
UNCACHEABLE UNION	属于不可缓存的子查询的 UNION 中的非起始语句

- **table**：是对应于行中信息的表的名称。表的顺序指示 MySQL 读取表以处理查询的顺序。这不一定是在 **FROM** 子句中指定的顺序，因为优化程序将尝试确定哪一种顺序的处理最高效。
- **partitions**：指示与查询匹配的数据所在的分区，**NULL** 指示表未分区。仅当使用 **PARTITIONS** 关键字时才会显示此列。
- **type**：指示联接类型。该值可衡量 MySQL 扫描表的效率。本节稍后将介绍可能的类型值。
- **possible_keys**：指示 MySQL 将表的哪一个索引看做用于识别满足查询的行的候选索引。该值可以是包含一个或多个索引名称的列表，也可以是 **NULL**（如果没有候选索引）。单词 **PRIMARY** 指示 MySQL 将表的主键看做候选索引。
- **key**：指示优化程序关于 **possible_keys** 中列出的哪一个候选索引将产生最高效查询执行的决策。如果 key 值是 **NULL**，则意味着未选择索引。发生这种情况不是因为没有候选索引，就是因为优化程序认为它扫描表行的速度与使用任何可能的索引一样快。如果表很小，或者因为索引会在表中产生较大比例的、没有什么用途的行，则可能选择表扫描而不是索引扫描。

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

- **key_len：**指示 MySQL 所用的关键字的长度。可以通过该值推导出所用的索引列数。例如，如果某个索引包含三个 **INT** 列，则每个索引行包含三个 4 个字节的值。如果 **key_len** 是 12，这表示优化程序使用三列索引处理查询。如果 **key_len** 是 4 或 8，则仅使用第一或第二列（即，使用索引最左侧的前缀）。

  如果索引了字符串列的部分值，则访问 **key_len** 值时请将其考虑在内。假设使用了两个 **CHAR(8)** 列上的复合索引，且仅索引每列的前四个字节。在这种情况下，**key_len** 值为 8 意味着将使用索引的两列，而不仅仅是第一列。
- **ref：**指示用来选择表中行的一个或一些索引列。**const** 意味着将索引中的关键字值与常数表达式进行比较，例如在 **Code='FRA'** 中。**NULL** 指示未使用常数和另一个列，指示通过表达式或值的范围进行选择。这也可能指示列不包含常数表达式所指定的值。如果未显示 **NULL** 或 **const**，而显示了 **table_name.column_name** 组合，指示优化程序在 **table_name** 所返回的行中查找 **column_name** 来确定当前表的行。
- **rows：**是优化程序对需要检查的表行数的估计。该值是一个近似值，因为：一般来说，没有实际执行查询，MySQL 便无法知道确切的行数。对于多表查询，行数的结果是对需要读取的行组合总数的估计。通过此结果可粗略衡量查询性能。值越小越好。
- **filtered：**指示按表条件筛选出的表行数（如 **rows** 列所示）所占的估计百分比。使用 **EXPLAIN EXTENDED** 语法时，会显示此列。
- **Extra：**提供有关联接的其他信息。本节稍后将介绍可能的值。

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

联接的 EXPLAIN 输出列

EXPLAIN 输出的 **type** 列中的值指示联接类型，但联接可能会以不同的效率执行。通过指示在每个表中选择行的基础，**type** 值可用于衡量此效率。下面的列表显示了可能的值（从最佳类型到最差类型）：

- **system**：表中只有一行。
- **const**：表中只有一个匹配行。此类型值与 **system** 类似，但表可能有其他不匹配行。这种情况的一个示例是包含 **WHERE Code='FRA'** 的查询的 **EXPLAIN** 输出：

```
mysql> EXPLAIN SELECT * FROM Country WHERE Code = 'FRA'\G
*************************** 1. row ***************************
           id: 1
  select_type: SIMPLE
        table: Country
         type: const
possible_keys: PRIMARY
          key: PRIMARY
      key_len: 3
          ref: const
         rows: 1
        Extra:
```

因为只需读取所有行中的一行，所以该查询使用了 **const** 类型值。如果表中仅包含 **France** 行，表中将没有匹配行，且类型值应该是 **system**，而不是 **const**。

对于 **system** 和 **const**，因为仅匹配一行，所以处理查询的其余部分时，可以一次读取该行所需的任何列，并将列视为常数。

- **eq_ref**：在 **EXPLAIN** 前面所列出的表中，对于每个行组合，只从表中读取一行。常用于联接，其中 MySQL 可以使用主键来确定表行。
- **ref**：在 **EXPLAIN** 前面所列出的表中，对于每个行组合，可能会从表中读取多行。这与 **eq_ref** 类似，但是可能发生在使用非唯一索引来确定表行或者仅使用索引最左侧的前缀的时候。例如，**CountryLanguage** 表的主键为 **CountryCode** 和 **Language** 列。如果只使用 **CountryCode** 值进行搜索，MySQL 可将该列用作最左侧的前缀，但是如果一个国家使用多种语言，则可能有多行。
- **fulltext**：指示联接使用 **FULLTEXT** 索引
- **ref_or_null**：与 ref 类似，但是 MySQL 也查找包含 **NULL** 的行
- **index_merge**：MySQL 使用索引合并算法。
- **unique_subquery**：与 **ref** 类似，但用于从单个表的主键列中进行选择的 **IN** 子查询。
- **index_subquery**：与 **unique_subquery** 类似，但用于从单个表的索引列中进行选择的 **IN** 子查询。
- **range**：使用索引来选择给定索引值范围内的行。常用于不等式比较，如 **id < 10**。

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

- **index**：MySQL 执行完全扫描，但仅扫描索引，而不是数据行。索引扫描更可取：索引是经过排序的，索引行通常比数据行短，所以可按顺序读取索引行，且一次读取更多行。
- **ALL**：对所有数据行进行完整表扫描。通常，这表示没有进行优化，是最坏情况。当 **EXPLAIN** 输出中后面列出的表联接类型为 **ALL** 时，情况特别不利，因为这指示针对联接中前面所处理的表中所选的每个行组合进行表扫描。

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

表处理的 EXPLAIN 输出列

EXPLAIN 输出中的 **Extra** 列提供有关如何处理表的其他信息。一些值表示查询是高效的：

- 使用 **index**：通过从索引读取值而不读取对应的数据行，MySQL 可以优化查询。当查询仅选择索引中的列时，才能实现此优化。
- 使用 **where**：MySQL 使用 **WHERE** 子句来确定满足查询的行。没有 **WHERE** 子句，您将获取表中所有的行。
- **Distinct**：在 **EXPLAIN** 输出中前面所列出的表中，对于每个行组合，MySQL 只从表中读取一行。
- **Not exists**：MySQL 可以执行 **LEFT JOIN**“缺少行”优化，从考虑范围中迅速消除行。

与之相对，一些 **Extra** 值表示查询不够高效：

- **Using filesort**：必须对满足查询的行进行排序，这增加了额外的处理步骤。
- **Using temporary**：必须创建临时表来处理查询。
- **Range checked for each record**：MySQL 无法提前确定要使用表中的哪个索引。对于前面表中所选的每个行组合，将检查表中的索引来确定哪个索引最好。这不是最好的方法，但是比完全不使用索引要好。

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

联接的 EXPLAIN 输出列

要使用 **EXPLAIN** 进行查询分析，请查看其输出以寻找改进查询的方法。修改查询，然后再次运行 **EXPLAIN** 来查看其输出如何变化。变化可能涉及重写查询或修改表的结构。

下列查询重写方法非常有用：

- 如果关键字值是 **NULL**，即使有可用的索引，也可以尝试将 **USE INDEX** 选项作为提示添加到其索引与查询相关的优化程序。要强制 MySQL 使用索引，请使用 **FORCE INDEX**。要让 MySQL 忽略所选的索引而选择一个不同的索引，请使用 **IGNORE INDEX**。这些选项都在 **FROM** 子句中使用，放在包含要控制的索引的表名后面。选项后跟用圆括号括住的、包含一个或多个索引名称的逗号分隔列表。**PRIMARY** 表示表的主键。

```
mysql> SELECT Name FROM CountryList USE INDEX(PRIMARY)
    -> WHERE Code > 'M';
mysql> SELECT Name FROM CountryList IGNORE INDEX(Population)
    -> WHERE Code < 'B' AND Population > 50000000;
```

  在所有三个选项中，都可以使用关键字 **KEY** 替代 **INDEX**。
- 要强制 MySQL 以特定的顺序联接表，请以 **SELECT STRAIGHT_JOIN**（而不是 **SELECT**）开始查询，然后在 **FROM** 子句中按照所需顺序列出各表。
- 有时，查询中的表有可用的索引，但是编写的查询采用了防止使用索引的方式。如果可以，请采用允许使用索引的等效格式重写查询。

更改表的结构是另一种向优化程序提供更优的决策基础信息的方法：

- 如果 **EXPLAIN** 输出中的 **possible_keys** 值是 **NULL**，则意味着 MySQL 找不到用于处理查询的合适索引。请查看能否向标识要检索的记录的列添加索引。例如，如果通过将一个表中的某列与另一个表中的一列相匹配来执行联接，但是未为这两列创建索引，请尝试为其创建索引。
- 使表索引统计信息保持最新可帮助 MySQL 选择最佳索引。如果表是 InnoDB 和 MyISAM 表，您可以使用 **ANALYZE TABLE** 语句更新其统计信息。当表中的内容更改时，统计信息会过时，从而很难帮助优化程序做出有关查询执行策略的决定。比起很少更新的表，应该更频繁地对经常更改的表运行 **ANALYZE TABLE**。
- 使用 **EXPLAIN** 来分析 **FROM** 子句中包含子查询的语句时，如果子查询自身运行缓慢，请特别注意。对于此类子查询，MySQL 必须执行其来确定其返回内容，以便优化程序可以弄清外部查询的执行计划。

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

附录：练习解答脚本

第 21 章

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

第 1 课：MySQL 简介

本课没有脚本

第 2 课：MySQL 体系结构

本课没有脚本

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

第 3 课：系统管理

练习 3-1：安装 MySQL 服务器

```
-- Step 1
-- (at the linux terminal)
su -
cd /stage/mysql
rpm -hi --replacefiles MySQL-server*.rpm
rpm -hi MySQL-client*.rpm
rpm -hi MySQL-devel*.rpm
rpm -hi MySQL-shared*.rpm
rpm -hi MySQL-test*.rpm

-- Step 2
service mysql start

-- Step 3
cat /root/.mysql_secret
mysql -uroot -p

-- (at the mysql prompt)
SET PASSWORD=PASSWORD('oracle');
EXIT

-- (at the linux terminal)
/usr/bin/mysql_secure_installation
```

练习 3-2：MySQL 数据目录

```
-- Step 1
-- (at the linux terminal)
mysql -uroot -poracle

-- (at the mysql prompt)
CREATE DATABASE world_innodb;
USE world_innodb
SET autocommit=0;
SOURCE /labs/world_innodb.sql
SET autocommit=1;

-- Step 2
SHOW DATABASES;

-- Step 3
SHOW VARIABLES LIKE 'datadir'\G

-- Step 4
-- (at the linux terminal)
cd /var/lib/mysql
ls

-- Step 5
su -
cd /var/lib/mysql/mysql
ls
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

```
-- Step 6
cd ../world_innodb
ls
```

练习 3-3：启动和停止 MySQL 服务器

```
-- Step 1
su -

-- Step 2
service mysql status

-- Step 3
service mysql stop

-- Step 4
service mysql status

-- Step 5
service mysql start

-- Step 6
service mysql status
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

第 4 课：服务器配置

练习 4-1：测验 – MySQL 服务器配置

本练习没有脚本

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

练习 4-2：编辑和创建配置文件

```
-- Step 1
--  (in a Linux terminal)
-- Type "su -" and enter the password
cp /etc/my.cnf /root
gedit /etc/my.cnf

-- Steps 2, 3, 4
-- New contents of /etc/my.cnf repeated here for convenience
[mysqld]
datadir=/var/lib/mysql
socket=/var/lib/mysql/mysql.sock
port=3309
user=mysql
# Disabling symbolic-links is recommended to prevent assorted security
risks
symbolic-links=0
general_log
log-bin=mybinlog
slow_query_log

[mysqld_safe]
log-error=/var/log/mysqld.log
pid-file=/var/run/mysqld/mysqld.pid

[client]
port=3309

-- Step 5
--  (in a Linux terminal)
service mysql restart

-- Step 6
--  (in a Linux terminal)
ls /var/lib/mysql

-- Step 7
--  (in a Linux terminal)
cd
pwd
gedit my_opts.txt

-- New contents of my-opts.txt repeated here for convenience
[client]
password = oracle
user = root

[mysql]
compress
show-warnings
prompt = \R:\m \d>\_

-- Step 8
--  (in a Linux terminal)
mysql --defaults-extra-file=~/my_opts.txt
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

```
-- Step 9
--  (in a mysql prompt)
STATUS
USE world_innodb
EXIT

-- Step 10
-- New contents of /etc/my.cnf repeated here for convenience
[mysqld]
datadir=/var/lib/mysql
socket=/var/lib/mysql/mysql.sock
port=3309
user=mysql
# Disabling symbolic-links is recommended to prevent assorted security
risks
symbolic-links=0
general_log
log-bin=mybinlog
slow_query_log

[mysqld_safe]
log-error=/var/log/mysqld.log
pid-file=/var/run/mysqld/mysqld.pid

[client]
port=3309
user=root

[mysql]
compress
show-warnings
prompt = \R:\m \d>\_

-- Step 11
--  (in a Linux terminal)
mysql -p

-- Step 12
--  (in a Linux terminal)
mysql_config_editor set --user=root --password

-- Step 13
--  (in a Linux terminal)
mysql_config_editor print --all

-- Step 14
--  (in a Linux terminal)
cat ~/.mylogin.cnf

-- Step 15
--  (in a Linux terminal)
mysql
-- (in the mysql prompt)
EXIT
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

```
-- Step 16
--  (in a Linux terminal)
mysql_config_editor remove

-- Step 17
--  (in a Linux terminal)
mysql_config_editor set --login-path=admin --user=root --password

-- Step 18
--  (in a Linux terminal)
mysql --login-path=admin

-- Step 19
--  (in a Linux terminal)
cp /root/my.cnf /etc/
service mysql restart
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

练习 4-3：附加练习 — 服务器配置

```
-- Step 1
--  (in a Linux terminal)
mysql --login-path=admin

-- (in the mysql prompt)
SHOW GLOBAL VARIABLES LIKE '%log';
SET GLOBAL general_log = ON;
SET GLOBAL slow_query_log = ON;
SHOW GLOBAL VARIABLES LIKE '%log';


-- Step 2
-- (in the mysql prompt)
SHOW GLOBAL VARIABLES LIKE 'log_output';
SET GLOBAL log_output = 'TABLE';
SHOW GLOBAL VARIABLES LIKE 'log_output';
TRUNCATE mysql.general_log;
TRUNCATE mysql.slow_log;

-- Step 3
-- (in the mysql prompt)
CREATE DATABASE world2;
USE world2;
SET autocommit=0;
SOURCE /labs/world_innodb.sql
SET autocommit=1;

-- Step 4
-- (in the mysql prompt)
SELECT COUNT(*) FROM mysql.general_log
WHERE argument LIKE 'CREATE TABLE%';

-- Step 5
-- (in the mysql prompt)
SELECT SLEEP(11);
SELECT * FROM mysql.slow_log\G

-- Step 6
-- (in the mysql prompt)
SHOW GLOBAL VARIABLES LIKE 'log_bin';
EXIT

-- Step 7
--  (in a Linux terminal)
gedit /etc/my.cnf
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

```
-- New contents of /etc/my.cnf repeated here for convenience
[mysqld]
log-bin=mysql-bin
datadir=/var/lib/mysql
socket=/var/lib/mysql/mysql.sock
user=mysql
# Disabling symbolic-links is recommended to prevent assorted security
risks
symbolic-links=0

[mysqld_safe]
log-error=/var/log/mysqld.log
pid-file=/var/run/mysqld/mysqld.pid

--  (in a Linux terminal)
service mysql restart

-- Step 8
--  (in a Linux terminal)
mysql --login-path=admin


-- (in the mysql prompt)
SHOW GLOBAL VARIABLES LIKE 'log_bin';
RESET MASTER;

-- Step 9
-- (in the mysql prompt)
CREATE DATABASE foo;
DROP DATABASE foo;
SHOW BINLOG EVENTS;

-- Step 10
-- (in the mysql prompt)
FLUSH BINARY LOGS;
CREATE DATABASE foo;
DROP DATABASE foo;

-- Step 11
-- (in the mysql prompt)
SHOW MASTER LOGS;
SHOW BINLOG EVENTS IN 'mysql-bin.000002';

-- Step 12
-- (in the mysql prompt)
PURGE MASTER LOGS TO 'mysql-bin.000002';

-- Step 13
-- (in the mysql prompt)
CREATE DATABASE foo2;
DROP DATABASE foo2;

-- Step 14
--  (in a Linux terminal)
mysqlbinlog /var/lib/mysql/mysql-bin.000002
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

```
-- Step 15
-- No steps

-- Step 16
-- (in the mysql prompt)
INSTALL PLUGIN audit_log SONAME 'audit_log.so';

-- Step 17
-- (in the mysql prompt)
CREATE DATABASE foo;
DROP DATABASE foo;

-- Step 18
-- (in the mysql prompt)
SHOW VARIABLES LIKE 'audit_log_file';

--  (in a Linux terminal)
cat /var/lib/mysql/audit.log

-- Step 19
-- (in the mysql prompt)
UNINSTALL PLUGIN audit_log;
SHOW WARNINGS;

-- Step 20
-- (in the mysql prompt)
CREATE DATABASE foo;
DROP DATABASE foo;

-- Step 21
-- (in the mysql prompt)
EXIT

-- Step 22
--  (in a Linux terminal)
cat /var/lib/mysql/audit.log

-- Step 23
-- New contents of /etc/my.cnf repeated here for convenience
[mysqld]
plugin-load=audit_log.so
audit-log=FORCE_PLUS_PERMANENT
log-bin=mysql-bin
datadir=/var/lib/mysql
socket=/var/lib/mysql/mysql.sock
user=mysql
# Disabling symbolic-links is recommended to prevent assorted security
risks
symbolic-links=0

[mysqld_safe]
log-error=/var/log/mysqld.log
pid-file=/var/run/mysqld/mysqld.pid

--  (in a Linux terminal)
service mysql restart
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

```
-- Step 24
--  (in a Linux terminal)
mysql --login-path=admin

-- (in the mysql prompt)
UNINSTALL PLUGIN audit_log;

-- Step 25
--  (in a Linux terminal)
cat /var/lib/mysql/audit.log
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

第 5 课：客户机和工具

练习 5-1：调用 mysql 客户机

```
-- Step 1
-- (at the linux terminal)
mysql -V

-- Step 2
-- (at the linux terminal)
mysql --help

-- Step 3
-- (at the linux terminal)
mysql -uroot -poracle

-- (at the mysql prompt)
EXIT


-- Step 4
-- (at the linux terminal)
mysql --login-path=admin --html -e \
    "SELECT CURRENT_DATE(), CURRENT_TIME()"

-- Step 5
-- (at the linux terminal)
mysql -uroot -p -P 3306 --tee=tee_1.txt

-- (at the mysql prompt)
EXIT

-- (at the linux terminal)
cat tee_1.txt

-- Step 6
-- (at the linux terminal)
mysql --login-path=admin --safe-updates

-- Step 7
-- (at the mysql prompt)
STATUS

-- Step 8
-- (at the mysql prompt)
HELP CONTENTS;
HELP Account Management;
HELP SET PASSWORD;

-- Step 9
-- (at the mysql prompt)
SHOW DATABASES;
EXIT
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

练习 5-2：调用 mysqladmin 客户机

```
-- Step 1
-- (at the linux terminal)
mysqladmin -V

-- Step 2
-- (at the linux terminal)
mysqladmin --help

-- Step 3
-- (at the linux terminal)
mysqladmin -uroot -poracle variables
```

练习 5-3：观看 MySQL Enterprise Monitor 演示

请按照练习中的说明进行操作。

练习 5-4：通过 MySQL Workbench 执行系统管理任务

请按照练习中的说明进行操作。要替换练习，请使用以下步骤。

```
-- Step 1
-- (at the linux terminal)
su -
cd /stage/mysql
rpm -hi --nodeps mysql-workbench*.rpm

-- Practice uses MySQL Workbench to load the sakila database
-- To replace this practice:
-- (at the mysql prompt)
SOURCE /labs/sakila-db/sakila-schema.sql
SOURCE /labs/sakila-db/sakila-data.sql
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

第 6 课：数据类型

练习 6-1：测验

本练习没有脚本

练习 6-2：设置数据类型

```
-- All steps carried out at the mysql prompt

-- Step 1
CREATE DATABASE test;

-- Step 2
USE test
CREATE TABLE integers(n SMALLINT UNSIGNED);
INSERT INTO integers VALUES (5);

-- Step 3
SELECT * FROM integers;

-- Step 4
SHOW TABLE STATUS LIKE 'integers'\G

-- Step 5
USE world_innodb;
CREATE TABLE CityCopy LIKE City;
INSERT INTO CityCopy SELECT * FROM City;

-- Step 6
DESC CityCopy;
SHOW TABLE STATUS LIKE 'CityCopy'\G

-- Step 7
ALTER TABLE CityCopy MODIFY Population BIGINT;
DESC CityCopy;
SHOW TABLE STATUS LIKE 'CityCopy'\G

-- Step 8
DROP TABLE CityCopy;
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

第 7 课：获取元数据

练习 7-1：使用 INFORMATION_SCHEMA 获取元数据

```
-- Step 1
-- (at the linux terminal)
mysql -uroot -poracle

-- (at the mysql prompt)
SELECT *
FROM INFORMATION_SCHEMA.SCHEMATA
WHERE SCHEMA_NAME = 'world_innodb'\G


-- Step 2
-- (at the mysql prompt)
USE INFORMATION_SCHEMA
SELECT TABLE_NAME, ENGINE
FROM TABLES
WHERE TABLE_SCHEMA = 'world_innodb';

-- Step 3
-- (at the mysql prompt)
SELECT TABLE_SCHEMA, ENGINE, COUNT(*)
FROM TABLES
GROUP BY TABLE_SCHEMA, ENGINE;

-- Step 4
-- (at the mysql prompt)
SELECT TABLE_NAME, DATA_LENGTH
FROM TABLES
WHERE TABLE_SCHEMA = 'world_innodb'
AND TABLE_NAME = 'City';

-- Step 5
-- (at the mysql prompt)
SELECT DATA_TYPE, COUNT(*)
FROM COLUMNS
WHERE TABLE_SCHEMA = 'world_innodb'
AND DATA_TYPE IN ('CHAR', 'VARCHAR')
GROUP BY DATA_TYPE;
```

练习 7-2：使用 SHOW 和 DESCRIBE 获取元数据

```
-- All steps at the mysql prompt

-- Step 1
SHOW DATABASES;

-- Step 2
SHOW DATABASES LIKE '%o%';

-- Step 3
USE information_schema;
SHOW TABLES;
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

```
-- Step 4
USE world_innodb
SHOW FULL COLUMNS FROM City\G

-- Step 5
SHOW INDEX FROM City\G

-- Step 6
DESCRIBE CountryLanguage;

-- Step 7
SHOW CHARACTER SET;

-- Step 8
SHOW COLLATION;

-- Step 9
EXIT
```

练习 7-3：使用 mysqlshow 获取元数据

```
-- All steps at the linux terminal

-- Step 1
mysqlshow -uroot -poracle

-- Step 2
mysqlshow world_innodb -uroot -poracle

-- Step 3
mysqlshow world_innodb CountryLanguage -uroot -poracle
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

第 8 课：事务和锁定

练习 8-1：测验 — 事务和锁定

本练习没有脚本

练习 8-2：使用事务控制语句

```
-- After Step 1, all steps use the mysql client

-- Step 1
-- (at the linux terminal)
mysql -uroot -poracle

-- (at the mysql prompt)
SHOW ENGINES\G

-- Step 2
SET AUTOCOMMIT = 1;
SELECT @@AUTOCOMMIT;

-- Step 3
USE world_innodb
SHOW CREATE TABLE City\G

-- Step 4
START TRANSACTION;

-- Step 5
SELECT * FROM City WHERE Name = 'Manta';
DELETE FROM City WHERE Name = 'Manta';
SELECT * FROM City WHERE Name = 'Manta';

-- Step 6
ROLLBACK;
SELECT * FROM City WHERE Name = 'Manta';

-- Step 7
START TRANSACTION;

-- Step 8
DELETE FROM City WHERE Name = 'Manta';

-- Step 9
COMMIT;

-- Step 10
SELECT * FROM City WHERE Name = 'Manta';

-- Step 11
ROLLBACK;
SELECT * FROM City WHERE Name = 'Manta';
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

练习 8-3：事务和锁定的附加练习

```
-- Step 1
-- (at the linux terminal)
mysql -uroot -poracle

-- (at the mysql prompt)
SELECT @@global.tx_isolation;

-- Step 2
PROMPT t1> ;

-- Step 3
START TRANSACTION;

-- Step 4
USE world_innodb
SELECT * FROM City WHERE ID > 4070;

-- Step 5
-- (at a new linux terminal)
mysql -uroot -poracle

-- (at the new mysql prompt)
PROMPT t2> ;

-- Step 6
START TRANSACTION;

-- Step 7
USE world_innodb
SELECT * FROM City WHERE ID > 4070;

-- Step 8
INSERT INTO City (Name, CountryCode) VALUES ('New City', 'ATA');
SELECT * FROM City WHERE ID > 4070;

-- Step 9
-- (at the t1 mysql prompt)
SELECT * FROM City WHERE ID > 4070;

-- Step 10
-- (at the t2 mysql prompt)
COMMIT;

-- Step 11
-- (at the t1 mysql prompt)
COMMIT;
SELECT * FROM City WHERE ID > 4070;

-- Step 12
START TRANSACTION;

-- Step 13
DELETE FROM City WHERE ID = 4080;
SELECT * FROM City WHERE ID > 4070;
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

```
-- Step 14
-- (at the t2 mysql prompt)
SET SESSION tx_isolation = 'READ-UNCOMMITTED';
SELECT @@tx_isolation;

-- Step 15
START TRANSACTION;
SELECT * FROM City WHERE ID > 4070;

-- Step 16
-- (at the t1 mysql prompt)
ROLLBACK;
SELECT * FROM City WHERE ID > 4070;

-- Step 17
-- (at the t2 mysql prompt)
SELECT * FROM City WHERE ID > 4070;

-- Step 18
ROLLBACK;

-- Step 19
-- (at the t1 mysql prompt)
EXIT

-- (at the Linux terminal)
exit

-- (at the t2 mysql prompt)
EXIT

-- (at the Linux terminal)
exit
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

第 9 课：存储引擎

练习 9-1：测验 — InnoDB 存储引擎

本练习没有脚本

练习 9-2：设置并确认 InnoDB 设置

```
-- Step 1
-- (at the linux terminal)
su -
head -1 /proc/meminfo
gedit /etc/my.cnf

-- The new contents of /etc/my.cnf are reproduced here for convenience
[mysqld]
innodb_buffer_pool_size=2GB
innodb_buffer_pool_instances=2
log-bin=mysql-bin
datadir=/var/lib/mysql
socket=/var/lib/mysql/mysql.sock
user=mysql
# Disabling symbolic-links is recommended to prevent assorted security
risks
symbolic-links=0

[mysqld_safe]
log-error=/var/log/mysqld.log
pid-file=/var/run/mysqld/mysqld.pid

-- Step 2
-- (at the linux terminal)
service mysql restart

-- Step 3
-- (at the linux terminal)
mysql -uroot -poracle

-- (at the mysql prompt)
SELECT @@default_storage_engine;

-- Step 4
USE INFORMATION_SCHEMA
SHOW TABLES LIKE 'INNODB%';

-- Step 5
SHOW VARIABLES LIKE '%innodb%';
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

```
-- Step 6
-- The new contents of /etc/my.cnf are reproduced here for convenience
[mysqld]
innodb_autoextend_increment=128
innodb_buffer_pool_size=2GB
innodb_buffer_pool_instances=2
log-bin=mysql-bin
datadir=/var/lib/mysql
socket=/var/lib/mysql/mysql.sock
user=mysql
# Disabling symbolic-links is recommended to prevent assorted security
risks
symbolic-links=0

[mysqld_safe]
log-error=/var/log/mysqld.log
pid-file=/var/run/mysqld/mysqld.pid

-- (at the linux terminal)
service mysql restart

-- Step 7
mysql -uroot -poracle

-- (at the mysql prompt)
SHOW VARIABLES LIKE '%innodb%';

-- Step 8
USE world_innodb
CREATE TABLE CityLanguage (
City CHAR(35),
Country CHAR(35),
CountryCode CHAR(3),
Language CHAR(30)
) ENGINE=MEMORY;

SHOW CREATE TABLE CityLanguage\G

-- Step 9
ALTER TABLE CityLanguage ENGINE=InnoDB;
SHOW CREATE TABLE CityLanguage\G
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

第 10 课：分区

练习 10-1：测验 – MySQL 分区

本练习没有脚本

练习 10-2：创建并修改分区表

```
-- Step 1
-- (at the linux terminal)
mysql --login-path=admin

-- (at the mysql prompt)
SHOW VARIABLES LIKE 'innodb_file_per_table';

-- Step 2
USE world_innodb
CREATE TABLE City_part LIKE City;

-- Step 3
SHOW TABLE STATUS LIKE 'City_part'\G

-- Step 4
ALTER TABLE City_part PARTITION BY RANGE (ID) (
PARTITION p0 VALUES LESS THAN (1000),
PARTITION p1 VALUES LESS THAN (2000),
PARTITION p2 VALUES LESS THAN (3000),
PARTITION p3 VALUES LESS THAN MAXVALUE
);

-- Step 5
SHOW TABLE STATUS LIKE 'City_part'\G

-- Step 6
INSERT INTO City_part SELECT * FROM City;

-- Step 7
-- (at the linux terminal)
su -
cd /var/lib/mysql/world_innodb; ls -l

-- Step 8
-- (at the mysql prompt)
EXPLAIN PARTITIONS SELECT * FROM City_part\G

-- Step 9
EXPLAIN PARTITIONS SELECT * FROM City_part WHERE ID < 2000\G

-- Step 10
ALTER TABLE City_part
PARTITION BY KEY (ID) PARTITIONS 3;

-- Step 11
EXPLAIN PARTITIONS SELECT * FROM City_part\G
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

```
-- Step 12
-- (at the linux terminal)
cd /var/lib/mysql/world_innodb; ls -l

-- Step 13
-- (at the mysql prompt)
SELECT TABLE_NAME,
GROUP_CONCAT(PARTITION_NAME)
FROM INFORMATION_SCHEMA.PARTITIONS
WHERE TABLE_SCHEMA='world_innodb'
AND TABLE_NAME='City_part';
```

练习 10-3：从表中删除分区

```
-- Step 1
ALTER TABLE City_part DROP PARTITION p0;

-- Step 2
ALTER TABLE City_part PARTITION BY RANGE (id) (
PARTITION p0 VALUES LESS THAN (1000),
PARTITION p1 VALUES LESS THAN (2000),
PARTITION p2 VALUES LESS THAN (3000),
PARTITION p3 VALUES LESS THAN MAXVALUE
);

-- Step 3
-- (at the linux terminal)
cd /var/lib/mysql/world_innodb; ls -l

-- Step 4
-- (at the mysql prompt)
ALTER TABLE City_part DROP PARTITION p0;

-- Step 5
EXPLAIN PARTITIONS SELECT * FROM City_part\G

-- Step 6
-- (at the linux terminal)
cd /var/lib/mysql/world_innodb; ls -l

-- Step 7
-- (at the mysql prompt)
ALTER TABLE City_part REMOVE PARTITIONING;

-- Step 8
SHOW TABLE STATUS LIKE 'City_part'\G

-- Step 9
EXPLAIN PARTITIONS SELECT * FROM City_part\G

-- Step 10
-- (at the linux terminal)
cd /var/lib/mysql/world_innodb; ls -l
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

第 11 课：用户管理

练习 11-1：测验 – MySQL 用户管理

本练习没有脚本

练习 11-2：创建、验证和删除用户

```
-- Step 1
-- (at the linux terminal)
mysql --login-path=admin

-- (at the mysql prompt)
SELECT user, host, password FROM mysql.user
WHERE user = '';

-- Step 2
CREATE USER ''@'localhost';

-- Step 3
SELECT user, host, password FROM mysql.user
WHERE user = '';

-- Step 4
DROP USER ''@'localhost';
SELECT user, host, password FROM mysql.user
WHERE user = '';
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

练习 11-3：设置 world_innodb 数据库的用户

```
-- All steps at the mysql prompt
-- Step 1
CREATE USER 'student'@'pc.example.com';

-- Step 2
SELECT user, host, password
FROM mysql.user WHERE user = 'student';

-- Step 3
GRANT SELECT, INSERT, DELETE, UPDATE ON world_innodb.*
TO 'student'@'pc.example.com'
IDENTIFIED BY 'student_pass';

-- Step 4
SHOW GRANTS FOR 'student'@'pc.example.com'\G

-- Step 5
REVOKE DELETE, UPDATE ON world_innodb.*
FROM 'student'@'pc.example.com';
SHOW GRANTS for 'student'@'pc.example.com'\G

-- Step 6
GRANT USAGE ON *.* TO 'student'@'pc.example.com'
IDENTIFIED BY 'NewPass';

-- Step 7
GRANT ALL ON world_innodb.* TO 'student'@'pc.example.com'
IDENTIFIED BY 'NewPass'
WITH MAX_CONNECTIONS_PER_HOUR 10;
SHOW GRANTS for 'student'@'pc.example.com'\G

-- Step 8
DROP USER 'student'@'pc.example.com';
SELECT user, host, password FROM mysql.user
WHERE user='student';
```

练习 11-4：使用 PAM 身份验证插件

```
-- Step 1
-- (at the linux terminal)
su -
useradd pamuser1
passwd pamuser1

-- Step 2
-- The new contents of /etc/pam.d/mysql-dba-course are reproduced here
for convenience.
#%PAM-1.0
auth            include         password-auth
account         include         password-auth

-- Step 3
chmod 440 /etc/shadow
chgrp mysql /etc/shadow
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

```
-- Step 4
-- (at the mysql prompt)
INSTALL PLUGIN authentication_pam
SONAME 'authentication_pam.so';

-- Step 5
SHOW PLUGINS;

-- Step 6
CREATE USER pamuser1@localhost
IDENTIFIED WITH authentication_pam AS 'mysql-dba-course';

-- Step 7
GRANT SELECT ON world_innodb.City TO pamuser1@localhost;

-- Step 8
-- (at the linux terminal)
mysql --enable-cleartext-plugin -upamuser1 -p

-- Step 9
-- (at the mysql prompt)
SELECT CURRENT_USER();

-- Step 10
EXIT

-- Step 11
-- (at the linux terminal)
useradd pamuser2
passwd pamuser2

-- Step 12
groupadd dba
usermod -G dba pamuser2

-- Step 13
-- (at the mysql prompt)
CREATE USER world_admin@localhost IDENTIFIED BY 'u=aX;yö#Qq';
GRANT ALL PRIVILEGES ON world_innodb.*
TO world_admin@localhost;

-- Step 14
CREATE USER ''@''
  IDENTIFIED WITH authentication_pam
AS 'mysql-dba-course, dba=world_admin';

-- Step 15
GRANT PROXY ON world_admin@localhost TO ''@'';

-- Step 16
-- (at the linux terminal)
mysql --enable-cleartext-plugin -upamuser2 -p
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

```
-- Step 17
-- (at the mysql prompt)
SELECT CURRENT_USER();

-- Step 18
EXIT
```

练习 11-5：附加练习

```
-- Step 1
-- (at the linux terminal)
mysql --login-path=admin

-- (at the mysql prompt)
SELECT user, host, password FROM mysql.user
WHERE user='';
DROP USER ''@'';

-- Step 2
CREATE USER 'Stefan'@'localhost'
IDENTIFIED BY 'weak';
SELECT user, host, password FROM mysql.user
WHERE user = 'Stefan';

-- Step 3
GRANT ALL ON world_innodb.* TO 'Stefan'@'localhost';
SHOW GRANTS FOR 'Stefan'@'localhost'\G

-- Step 4
mysql -uStefan -pweak

-- Step 5
SET PASSWORD FOR 'Stefan'@'localhost' = PASSWORD('new_pass');
EXIT

-- (at the linux terminal)
mysql -uStefan -pnew_pass

-- Step 6
mysql -uroot -p

-- (at the mysql prompt)
REVOKE ALL ON world_innodb.* FROM 'Stefan'@'localhost';
GRANT ALL ON world_innodb.Country TO 'Stefan'@'localhost';
GRANT ALL ON world_innodb.CountryLanguage
TO 'Stefan'@'localhost';
SHOW GRANTS FOR 'Stefan'@'localhost'\G

-- Step 7
GRANT SELECT (Name, Population) ON world_innodb.City
TO 'Stefan'@'localhost';
SHOW GRANTS FOR 'Stefan'@'localhost'\G
EXIT

-- Step 8
-- (at the linux terminal)
mysql -uStefan -pnew_pass
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

```
-- (at the mysql prompt)
USE world_innodb
SELECT * from Country\G
SELECT * from CountryLanguage;
SELECT * FROM City;
SELECT Name FROM City;
EXIT

-- Step 9
CREATE USER 'UserGroup4_01'@'localhost'
IDENTIFIED BY '0004nq2';
GRANT ALL ON world_innodb.Country
TO 'UserGroup4_01'@'localhost';
GRANT ALL ON world_innodb.CountryLanguage
TO 'UserGroup4_01'@'localhost';
GRANT SELECT ON world_innodb.City
TO 'UserGroup4_01'@'localhost';
SHOW GRANTS FOR 'UserGroup4_01'@'localhost'\G

-- Step 10
DROP USER 'UserGroup4_01'@'localhost';
SHOW GRANTS FOR 'UserGroup4_01'@'localhost'\G

-- Step 11
-- (at the mysql prompt)
PROMPT t1> ;
SHOW PROCESSLIST;

-- (at another Linux prompt)
mysql -uStefan -pnew_pass
PROMPT t2> ;
USE world_innodb;

-- (at yet another Linux prompt)
mysql --login-path=admin
PROMPT t3> ;
USE test;

-- Step 12
-- at the t1 prompt
SELECT SLEEP(60);

-- at the t2 prompt
SELECT Code FROM City, Country, CountryLanguage
LIMIT 10000000;

-- at the t3 prompt
SHOW PROCESSLIST;
SHOW PROCESSLIST\G
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

第 12 课：安全

练习 12-1：测验 — MySQL 安全

本练习没有脚本

练习 12-2：确定 SSL 连接的状态

```
-- Step 1
-- (at the linux terminal)
mysql --login-path=admin

-- (at the mysql prompt)
SHOW VARIABLES LIKE 'have_ssl';

-- Step 2
SHOW STATUS LIKE 'Ssl_cipher';
EXIT
```

练习 12-3：附加练习 — 为 MySQL 启用 SSL 连接支持

```
-- Step 1
-- (at the linux terminal)
su -

cd /etc
mkdir newcerts
cd newcerts
openssl genrsa 2048 > ca-key.pem
openssl req -new -x509 -nodes -days 1000 \
        -key ca-key.pem > ca-cert.pem
-- follow the sequence in the practice
openssl req -newkey rsa:2048 -days 1000 -nodes \
        -keyout server-key.pem > server-req.pem
-- follow the sequence in the practice
openssl x509 -req -in server-req.pem -days 1000 -CA ca-cert.pem \
         -CAkey ca-key.pem -set_serial 01 > server-cert.pem
openssl req -newkey rsa:2048 -days 1000 -nodes \
     -keyout client-key.pem > client-req.pem
-- follow the sequence in the practice
openssl x509 -req -in client-req.pem -days 1000 -CA ca-cert.pem \
         -CAkey ca-key.pem -set_serial 01 > client-cert.pem

-- Step 2
ls
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

```
-- Step 3
-- (at the linux terminal)
mysql --login-path=admin

-- (at the mysql prompt)
GRANT SELECT ON mysql.user TO 'Stefan'@'localhost'
IDENTIFIED BY 'new_pass' REQUIRE SSL;

SHOW GRANTS FOR 'Stefan'@'localhost'\G

EXIT


-- Step 4
-- (at the linux terminal)
service mysql restart --ssl-ca=/etc/newcerts/ca-cert.pem \
     --ssl-cert=/etc/newcerts/server-cert.pem \
     --ssl-key=/etc/newcerts/server-key.pem

-- Step 5
mysql -uStefan -pnew_pass

-- Step 6
mysql -uStefan -pnew_pass --ssl-ca=/etc/newcerts/ca-cert.pem

-- Step 7
-- (at the mysql prompt)
SHOW VARIABLES LIKE 'have_ssl';
SHOW STATUS LIKE 'Ssl_cipher';

-- Step 8
EXIT

-- Step 9
-- (at the linux terminal)
service mysql restart
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

第 13 课：表维护

练习 13-1：测验 — 表维护

本练习没有脚本

练习 13-2：使用表维护 SQL 语句

```
-- Step 1
-- (at the linux terminal)
mysql --login-path=admin

-- (at the mysql prompt)
USE world_innodb
CREATE TABLE City_temp LIKE City;
INSERT INTO City_temp SELECT * FROM City;
SHOW TABLE STATUS LIKE 'City_temp' \G

-- Step 2
CHECK TABLE City_temp\G

-- Step 3
DELETE FROM City_temp WHERE Id BETWEEN 1001 AND 2000;
SHOW TABLE STATUS LIKE 'City_temp' \G

-- Step 4
ANALYZE TABLE City_temp;
SHOW TABLE STATUS LIKE 'City_temp' \G
EXIT
```

练习 13-3：使用表维护实用程序

```
-- Step 1
-- (at the linux terminal)
mysqlcheck -uroot -p --databases world_innodb

-- Step 2
mysqlcheck --login-path=admin --all-databases

-- Step 3
mysqlcheck --login-path=admin --all-databases --analyze
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

第 14 课：导出和导入数据

练习 14-1：导出 MySQL 数据

```
-- Step 1
-- (at the linux terminal)
mysql -uroot -poracle

-- (at the mysql prompt)
USE world_innodb;

SELECT * INTO OUTFILE '/tmp/CountryLanguage.txt'
FROM CountryLanguage;

-- Step 2
-- (at the linux terminal)
more /tmp/CountryLanguage.txt

-- Step 3
-- (at the mysql prompt)
SELECT * INTO OUTFILE '/tmp/CountryLanguage.csv'
FIELDS TERMINATED BY ',' ENCLOSED BY '"'
LINES TERMINATED BY '\n'
FROM CountryLanguage;

-- Step 4
-- (at the linux terminal)
more /tmp/CountryLanguage.csv
```

练习 14-2：导入数据

```
-- Step 1
-- (at the mysql prompt)
CREATE TABLE CountryLanguage2 LIKE CountryLanguage;

-- Step 2
SHOW TABLES;

-- Step 3
LOAD DATA INFILE '/tmp/CountryLanguage.txt' INTO TABLE
CountryLanguage2;

-- Step 4
SELECT * FROM CountryLanguage2;
EXIT
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

第 15 课：在 MySQL 中编程

练习 15-1：创建存储例程

```
-- Step 1
-- (at the linux terminal)
mysql --login-path=admin

-- (at the mysql prompt)
USE world_innodb;

DELIMITER //
CREATE PROCEDURE record_count ()
BEGIN
  SELECT 'Country count ', COUNT(*) FROM Country;
  SELECT 'City count ', COUNT(*) FROM City;
  SELECT 'CountryLanguage count', COUNT(*) FROM
   CountryLanguage;
END//

DELIMITER ;

-- Step 2
CALL record_count();

-- Step 3
USE test;
CALL record_count();

-- Step 4
CALL world_innodb.record_count();

-- Step 5
USE world_innodb;

DELIMITER //
CREATE FUNCTION pay_check (gross_pay FLOAT(9,2),
 tax_rate FLOAT(3,2))
RETURNS FLOAT(9,2)
NO SQL
BEGIN
  DECLARE net_pay FLOAT(9,2) DEFAULT 0;
  SET net_pay=gross_pay - gross_pay * tax_rate;
  RETURN net_pay;
END//

DELIMITER ;

-- Step 6
SELECT pay_check (100000, 0.05);
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

练习 15-2：查看存储例程

```
-- Step 1
SHOW CREATE PROCEDURE record_count\G

-- Step 2
SHOW PROCEDURE STATUS LIKE 'record%'\G

-- Step 3
SHOW FUNCTION STATUS\G

-- Step 4
USE INFORMATION_SCHEMA;
SELECT * FROM ROUTINES\G
```

练习 15-3：创建触发器

```
-- Step 1
USE world_innodb;

CREATE TABLE DeletedCity
( ID INT UNSIGNED,
  Name VARCHAR(50),
  When_Deleted timestamp);

-- Step 2
CREATE TRIGGER City_AD AFTER DELETE ON City
FOR EACH ROW
INSERT INTO DeletedCity (ID, Name)
   VALUES (OLD.ID, OLD.Name);


-- Step 3
SHOW TRIGGERS\G

-- Step 4
DELETE FROM City WHERE Name = 'Dallas';

-- Step 5
SELECT * FROM City WHERE Name = 'Dallas';

-- Step 6
SELECT * FROM DeletedCity;
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

练习 15-4：创建和测试事件

```
-- Step 1
DELIMITER //
CREATE EVENT kill_user
  ON SCHEDULE EVERY 20 SECOND
DO
BEGIN
  DECLARE var_id INT;
  DECLARE procs CURSOR FOR SELECT id
  FROM INFORMATION_SCHEMA.PROCESSLIST WHERE TIME>30
    AND Command != 'Sleep' AND USER != 'root';
  OPEN procs;
  BEGIN
    DECLARE EXIT HANDLER FOR NOT FOUND
    BEGIN
      CLOSE procs;
    END;
    LOOP
      FETCH procs INTO var_id;
      KILL var_id;
    END LOOP;
  END;
END//
DELIMITER ;

-- Step 2
SET GLOBAL event_scheduler = ON;

-- Step 3
CREATE USER tester@localhost IDENTIFIED BY 'secret';

-- Step 4
GRANT SELECT ON *.* TO tester@localhost;

-- Step 5
-- (at the linux terminal)
mysql -utester -p

-- Step 6
-- (at the mysql prompt)
SELECT SLEEP(60);

-- Step 7
EXIT
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

第 16 课：MySQL 备份和恢复

练习 16-1：测验 — 备份简介

本练习没有脚本

练习 16-2：MySQL Enterprise Backup

```
-- Step 1
-- (at the linux terminal)
su -
mkdir /backups
chown mysql:mysql /backups

-- Step 2, Step 3, Step 4, Step 5
-- The contents of the new my.cnf file are reproduced here for
convenience
[mysqld]
innodb_autoextend_increment=128
innodb_buffer_pool_size=2GB
innodb_buffer_pool_instances=2
log-bin=mybinlog
innodb_log_files_in_group = 2
innodb_log_file_size = 256M
innodb_data_file_path=ibdata1:12M:autoextend
datadir=/var/lib/mysql
socket=/var/lib/mysql/mysql.sock
user=mysql
# Disabling symbolic-links is recommended to prevent assorted security
risks
symbolic-links=0

[mysqld_safe]
log-error=/var/log/mysqld.log
pid-file=/var/run/mysqld/mysqld.pid

[mysqlbackup]
backup-dir=/backups/meb1
user=backupuser
socket=/var/lib/mysql/mysql.sock

-- Step 6
rm -f /var/lib/mysql/ib_logfile*

-- Step 7
service mysql restart

-- Step 8
mysql --login-path=admin
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

```
-- Step 9
-- (at the mysql prompt)
CREATE USER 'backupuser'@'localhost' IDENTIFIED BY 'oracle';
GRANT RELOAD ON *.* TO 'backupuser'@'localhost';
GRANT CREATE, INSERT, DROP ON mysql.ibbackup_binlog_marker
TO 'backupuser'@'localhost';
GRANT CREATE, INSERT, DROP ON mysql.backup_progress
TO 'backupuser'@'localhost';
GRANT CREATE, INSERT, DROP ON mysql.backup_history
TO 'backupuser'@'localhost';
GRANT REPLICATION CLIENT ON *.* TO 'backupuser'@'localhost';
GRANT SUPER ON *.* TO 'backupuser'@'localhost';
GRANT CREATE TEMPORARY TABLES ON mysql.*
TO 'backupuser'@'localhost';

-- Step 10
USE world_innodb;
CREATE TABLE City_Large LIKE City;

-- Step 11
INSERT INTO City_Large (Name, CountryCode, District,
 Population) SELECT Name, CountryCode, District,
 Population FROM City;

-- Step 12
INSERT INTO City_Large (Name, CountryCode, District,
 Population) SELECT Name, CountryCode, District,
 Population FROM City_Large;

-- Step 13
INSERT INTO City_Large (Name, CountryCode, District,
 Population) SELECT Name, CountryCode, District,
 Population FROM City_Large;
INSERT INTO City_Large (Name, CountryCode, District,
 Population) SELECT Name, CountryCode, District,
 Population FROM City_Large;
INSERT INTO City_Large (Name, CountryCode, District,
 Population) SELECT Name, CountryCode, District,
 Population FROM City_Large;
INSERT INTO City_Large (Name, CountryCode, District,
 Population) SELECT Name, CountryCode, District,
 Population FROM City_Large;
INSERT INTO City_Large (Name, CountryCode, District,
 Population) SELECT Name, CountryCode, District,
 Population FROM City_Large;
INSERT INTO City_Large (Name, CountryCode, District,
 Population) SELECT Name, CountryCode, District,
 Population FROM City_Large;
INSERT INTO City_Large (Name, CountryCode, District,
 Population) SELECT Name, CountryCode, District,
 Population FROM City_Large;

-- Step 14
FLUSH LOGS;
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

```
-- Step 15
INSERT INTO City_Large (Name, CountryCode, District,
 Population) SELECT Name, CountryCode, District,
 Population FROM City_Large;

-- Step 16
-- (at the linux terminal)
su mysql
mysqlbackup --defaults-file=/etc/my.cnf -p backup-and-apply-log

-- Step 17
ls /backups/meb1/

-- Step 18
exit
cd /var/lib/mysql
rm -f ibdata1
rm -rf world_innodb

-- Step 19
-- (at the mysql prompt)
SHOW DATABASES;

-- Step 20
EXIT;

-- Step 21
-- (at the linux terminal)
service mysql stop

-- Step 22
su mysql
mysqlbackup --defaults-file=/etc/my.cnf copy-back


-- Step 23
exit
service mysql start

-- Step 24
mysql --login-path=admin

-- (at the mysql prompt)
SHOW DATABASES;

-- Step 25
USE world_innodb
SHOW TABLES;

-- Step 26
EXIT

-- (at the linux terminal)
exit
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

练习 16-3：mysqldump

```
-- Step 1
-- (at the linux terminal)
mysqldump -uroot -p --tab=/backups world_innodb

-- Step 2
ls /backups -l

-- Step 3
more /backups/Country.sql

-- Step 4
head /backups/Country.txt

-- Step 5
mysqladmin -uroot -p create world3

-- Step 6
cd /backups
mysql -uroot -poracle world3 < Country.sql
mysql -uroot -poracle world3 < CountryLanguage.sql
mysql -uroot -poracle world3 < City.sql

-- Step 7
mysqlimport -uroot -poracle world3 /backups/Country.txt
mysqlimport -uroot -poracle world3 /backups/CountryLanguage.txt
mysqlimport -uroot -poracle world3 /backups/City.txt

-- Step 8
mysql -uroot -poracle

-- (at the mysql prompt)
USE world3
SHOW TABLES;

-- Step 9
SELECT COUNT(*) FROM world3.City;
SELECT COUNT(*) FROM world_innodb.City;

-- Step 10
SELECT COUNT(*) FROM world3.Country;
SELECT COUNT(*) FROM world_innodb.Country;
SELECT COUNT(*) FROM world3.CountryLanguage;
SELECT COUNT(*) FROM world_innodb.CountryLanguage;

-- Step 11
EXIT
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

练习 16-4：使用 LVM 快照和二进制日志进行备份和恢复

```
-- Step 1
-- (at the linux terminal)
service mysql stop

-- Step 2
dd if=/dev/zero of=/var/local/mysqldisk bs=1M count=2000
losetup /dev/loop2 /var/local/mysqldisk
vgcreate VG_MYSQL /dev/loop2
lvcreate -l50%VG -n lv_datadir VG_MYSQL
mkfs.ext4 /dev/VG_MYSQL/lv_datadir
mkdir -p /datadir
mount /dev/VG_MYSQL/lv_datadir /datadir
rmdir /datadir/lost+found
cp -a /var/lib/mysql/* /datadir/
chown mysql:mysql /datadir

-- Step 3
mkdir /binlogs
chown mysql:mysql /binlogs

-- Step 4
-- The contents of the my.cnf file are reproduced here for convenience
[root@EDTDR20P1 backups]# cat /etc/my.cnf
[mysqld]
innodb_autoextend_increment=128
innodb_buffer_pool_size=2GB
innodb_buffer_pool_instances=2
log-bin=/binlogs/mysql-bin
innodb_log_files_in_group = 2
innodb_log_file_size = 256M
innodb_data_file_path=ibdata1:12M:autoextend
datadir=/datadir
socket=/var/lib/mysql/mysql.sock
user=mysql
# Disabling symbolic-links is recommended to prevent assorted security
risks
symbolic-links=0

[mysqld_safe]
log-error=/var/log/mysqld.log
pid-file=/var/run/mysqld/mysqld.pid

[mysqlbackup]
backup-dir=/backups/meb1
user=backupuser
socket=/var/lib/mysql/mysql.sock

-- Step 5
service mysql start

-- Step 6
ls /binlogs
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

```
-- Step 7
mysql --login-path=admin

-- (at the mysql prompt)
DROP DATABASE world_innodb;
CREATE DATABASE world_innodb;
USE world_innodb
SET autocommit=0;
SOURCE /labs/world_innodb.sql
SET autocommit=1;

-- Step 8
SHOW TABLES;

-- Step 9
SHOW PROCESSLIST;

-- Step 10
CREATE TABLE City_Large LIKE City;
INSERT INTO City_Large (Name, CountryCode, District,
 Population) SELECT Name, CountryCode, District,
 Population FROM City;

SET autocommit=0;

INSERT INTO City_Large (Name, CountryCode, District,
 Population) SELECT Name, CountryCode, District,
 Population FROM City_Large;
INSERT INTO City_Large (Name, CountryCode, District,
 Population) SELECT Name, CountryCode, District,
 Population FROM City_Large;
INSERT INTO City_Large (Name, CountryCode, District,
 Population) SELECT Name, CountryCode, District,
 Population FROM City_Large;
INSERT INTO City_Large (Name, CountryCode, District,
 Population) SELECT Name, CountryCode, District,
 Population FROM City_Large;
INSERT INTO City_Large (Name, CountryCode, District,
 Population) SELECT Name, CountryCode, District,
 Population FROM City_Large;
INSERT INTO City_Large (Name, CountryCode, District,
 Population) SELECT Name, CountryCode, District,
 Population FROM City_Large;
INSERT INTO City_Large (Name, CountryCode, District,
 Population) SELECT Name, CountryCode, District,
 Population FROM City_Large;
INSERT INTO City_Large (Name, CountryCode, District,
 Population) SELECT Name, CountryCode, District,
 Population FROM City_Large;

-- Step 11
-- (at the linux terminal)
mysqladmin -uroot -poracle flush-logs
lvcreate -s -n lv_datadirbackup -L 500M /dev/VG_MYSQL/lv_datadir

-- Step 12
SET autocommit=1;
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

```
-- Step 13
-- (at the mysql prompt)
INSERT INTO City(Name, CountryCode) VALUES ('Sakila', 'SWE');
SELECT * FROM City ORDER BY ID DESC LIMIT 1;
DROP TABLE City;

-- Step 14
-- (at the linux terminal)
mkdir /root/snapshot
mount /dev/VG_MYSQL/lv_datadirbackup /root/snapshot
cd /root/snapshot
tar -czf /root/mysqldatadir.tgz .
cd
umount /root/snapshot

-- Step 15
lvremove VG_MYSQL/lv_datadirbackup -f

-- Step 16
lvresize -fL 1M VG_MYSQL/lv_datadir

-- Step 17
-- (at the mysql prompt)
SHOW DATABASES;
DROP DATABASE world3;

-- Step 18
-- (at the linux terminal)
service mysql stop
mysqladmin -uroot -poracle shutdown

-- Step 19
ps ax | grep mysqld

-- Step 20
killall -9 mysqld_safe

-- Step 21
ps ax | grep mysqld

-- Step 22
umount /datadir
lvremove -f VG_MYSQL/lv_datadir
lvcreate -l50%VG -n lv_datadir VG_MYSQL
mkfs.ext4 /dev/VG_MYSQL/lv_datadir
mount /dev/VG_MYSQL/lv_datadir /datadir
rmdir /datadir/lost+found
chown mysql:mysql /datadir

-- Step 23
cd /datadir
tar -xzf /root/mysqldatadir.tgz

-- Step 24
service mysql start
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

```
-- Step 25
mysql --login-path=admin

-- (at the mysql prompt)
USE world_innodb
SELECT * FROM City ORDER BY ID DESC LIMIT 1;

-- Step 26
-- (at the linux terminal)
ls /binlogs -l

-- Step 27
mysqlbinlog /binlogs/mysql-bin.000002 | more

-- Step 28
cd /binlogs
mysqlbinlog --disable-log-bin --stop-position=807 \
    mysql-bin.000002 | mysql -uroot -poracle


-- Step 29
-- (at the mysql prompt)
SELECT * FROM world_innodb.City WHERE ID > 4070;

-- Step 30
-- (at the linux terminal)
service mysql stop
-- change the my.cnf file as in the practice
service mysql start
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

第 17 课：复制

练习 17-1：测验 — 复制

本练习没有脚本

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

练习 17-2：配置复制

```
-- Step 1
-- (at the linux terminal)
service mysql stop

-- Step 2
cat /labs/repl.cnf

-- Step 3
mysqld_multi --defaults-file=/labs/repl.cnf start 1-4

-- Step 4
mysql -uroot -h127.0.0.1 -P3311

-- (at the mysql prompt)
PROMPT 1> ;

-- Step 5
SHOW MASTER STATUS \G

-- Step 6
CREATE USER 'repl'@'127.0.0.1' IDENTIFIED BY 'oracle';
GRANT REPLICATION SLAVE ON *.* TO 'repl'@'127.0.0.1';

-- Step 7
CREATE DATABASE world_innodb;
USE world_innodb
SET autocommit=0;
SOURCE /labs/world_innodb.sql
SET autocommit=1;

-- Step 8
-- (at the linux terminal)
mysql -uroot -h127.0.0.1 -P3312

-- (at the mysql prompt)
PROMPT 2> ;

-- Step 9
CHANGE MASTER TO
 MASTER_HOST='127.0.0.1',
 MASTER_PORT=3311,
 MASTER_LOG_FILE='mysql1-bin.000001',
 MASTER_LOG_POS=120,
 MASTER_USER='repl',
 MASTER_PASSWORD='oracle';

SHOW WARNINGS\G

-- Step 10
SHOW DATABASES;
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

```
-- Step 11
START SLAVE;

-- Step 12
SHOW PROCESSLIST\G
SHOW PROCESSLIST\G

-- Step 13
SHOW DATABASES;
```

练习 17-3：添加新从属服务器

```
-- Step 1
-- (at the linux terminal)
mysqldump -uroot -h127.0.0.1 -P3312 --master-data=2 \
     -B world_innodb > /tmp/server2.sql

-- Step 2
-- Edit the file as described in the practice

-- Step 3
mysql -uroot -h127.0.0.1 -P3313

-- (at the mysql prompt)
PROMPT 3> ;

-- Step 4
SOURCE /tmp/server2.sql

-- Step 5
START SLAVE;
SHOW SLAVE STATUS\G

-- Step 6
-- (on the 1st server)
DELETE FROM world_innodb.City WHERE ID > 4070;

-- (on the 2nd server)
SELECT * FROM world_innodb.City ORDER BY Id DESC LIMIT 5;

-- (on the 3rd server)
SELECT * FROM world_innodb.City ORDER BY Id DESC LIMIT 5;

-- Step 7
CREATE USER 'repl'@'127.0.0.1' IDENTIFIED BY 'oracle';
GRANT REPLICATION SLAVE ON *.* TO 'repl'@'127.0.0.1';
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

练习 17-4：启用 GTID 并配置循环复制

```
-- Step 1
-- (at the linux terminal)
mysqld_multi --defaults-file=/labs/repl.cnf --user=root stop 1-4

-- Step 2
-- Follow the step in the practice

-- Step 3
mysqld_multi --defaults-file=/labs/repl.cnf start 1-4

-- Step 4
-- (on the second server)
STATUS
STOP SLAVE;

-- (on the third server)
STATUS
STOP SLAVE;

-- Step 5
-- (on the first server)
STATUS
RESET MASTER;

-- (on the second server)
RESET MASTER;

-- (on the third server)
RESET MASTER;

-- Step 6
-- (on the second server)
CHANGE MASTER TO MASTER_AUTO_POSITION=1;

-- (on the third server)
CHANGE MASTER TO MASTER_AUTO_POSITION=1;

-- Step 7
-- (on the second server)
START SLAVE;

-- (on the third server)
START SLAVE;

-- Step 8
-- (on the first server)
DELETE FROM world_innodb.City WHERE ID > 4060;

-- (on the second server)
SELECT * FROM world_innodb.City ORDER BY Id DESC LIMIT 5;

-- (on the third server)
SELECT * FROM world_innodb.City ORDER BY Id DESC LIMIT 5;
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

```
-- Step 9
-- (on the first server)
SELECT @@server_uuid;

-- (on the second server)
SELECT @@server_uuid;

-- (on the third server)
SELECT @@server_uuid;

-- Step 10
SHOW SLAVE STATUS\G

-- Step 11
-- (on the first server)
CHANGE MASTER TO MASTER_HOST='127.0.0.1', MASTER_PORT=3313,
MASTER_USER='repl', MASTER_PASSWORD='oracle',
MASTER_AUTO_POSITION=1;

START SLAVE;

-- Step 12
-- (on the second server)
DELETE FROM world_innodb.City WHERE ID > 4050;

-- (on the first server)
SELECT * FROM world_innodb.City ORDER BY Id DESC LIMIT 5;

-- (on the third server)
SELECT * FROM world_innodb.City ORDER BY Id DESC LIMIT 5;
```

练习 17-5：使用 MySQL 实用程序并执行故障转移

```
-- Step 1
-- (on the first server)
STOP SLAVE;
RESET SLAVE ALL;

-- Step 2
-- (at the linux terminal)
mysql -uroot -h127.0.0.1 -P3314

-- (at the mysql prompt)
PROMPT 4> ;

-- Step 3
-- (at the fourth server)
SOURCE /tmp/server2.sql

-- Step 4
-- (at the linux terminal)
/usr/share/mysql-workbench/python/mysqldbcompare \
    --server1=root@127.0.0.1:3311 \
        --server2=root@127.0.0.1:3314 --changes-for=server2 \
        --difftype=sql -a world_innodb:world_innodb > /tmp/diff.sql
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

```
-- Step 5
cat /tmp/diff.sql

-- Step 6
-- (at the fourth server mysql prompt)
SOURCE /tmp/diff.sql

-- Step 7
CHANGE MASTER TO MASTER_PORT=3311, MASTER_AUTO_POSITION=1;
START SLAVE;

-- Step 8
-- (at the linux terminal)
/usr/share/mysql-workbench/python/mysqlrplshow \
     --master=root@127.0.0.1:3311 \
     --discover-slaves-login=root --recurse

-- Step 9
-- (at the fourth server)
CREATE USER 'repl'@'127.0.0.1' IDENTIFIED BY 'oracle';
GRANT REPLICATION SLAVE ON *.* TO 'repl'@'127.0.0.1';

-- Step 10
CREATE USER 'repl'@'localhost' IDENTIFIED BY 'oracle';
GRANT REPLICATION SLAVE ON *.* TO 'repl'@'localhost';

-- Step 11
-- (at the linux terminal)
/usr/share/mysql-workbench/python/mysqlfailover \
     --master=root@127.0.0.1:3311 \
     --discover-slaves-login=root --rpl-user=repl:oracle@127.0.0.1


-- Step 12
-- Follow the steps in the practice

-- Step 13
-- (at the linux terminal)
mysqladmin -uroot -h127.0.0.1 -P3311 shutdown

-- Step 14
-- No script

-- Step 15
/usr/share/mysql-workbench/python/mysqlrplshow \
     --master=root@127.0.0.1:3314 \
     --discover-slaves-login=root --recurse

-- Step 16
mysqld_multi --defaults-file=/labs/repl.cnf --user=root stop 1-4
service mysql start
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

第 18 课：性能调节简介

练习 18-1：测验 — 性能调节简介

本练习没有脚本

练习 18-2：EXPLAIN

```
-- Step 1
-- (at the linux terminal)
mysql --login-path=admin

-- (at the mysql prompt)
USE world_innodb


-- Step 2
DROP DATABASE world_innodb;
CREATE DATABASE world_innodb;
USE world_innodb;
SET AUTOCOMMIT=0;
SOURCE /labs/world_innodb.sql
SET AUTOCOMMIT=1;

-- Step 3
EXPLAIN SELECT Name FROM City WHERE Name LIKE 'A%'\G

-- Step 4
EXPLAIN SELECT Name FROM City WHERE ID > 4070\G

-- Step 5
-- No script

-- Step 6
CREATE INDEX iName ON City (Name) USING BTREE;
EXPLAIN SELECT Name FROM City WHERE Name LIKE 'A%'\G
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。

练习 18-3：PROCEDURE ANALYSE

```
-- Step 1
-- (at the mysql prompt)
SELECT Name, CountryCode, Population FROM City
PROCEDURE ANALYSE()\G

-- Step 2
DESC City;

-- Step 3
SELECT Name, CountryCode, Population FROM City
PROCEDURE ANALYSE(256,1024)\G

-- Step 4
SELECT * FROM CountryLanguage PROCEDURE ANALYSE(256,1024)\G

-- Step 5
DESC CountryLanguage;

-- Step 6
EXIT
```

版权所有 © 2013，Oracle 和/或其附属公司。保留所有权利。