第 25 章

25 创建与数据库相关的脚本

本章内容

- 编写数据库shell脚本

到目前为止，我们已经讲述了shell脚本的很多特性。不过这还不够！要想提供先进的特性，还得利用shell脚本之外的高级功能，例如访问数据库、从互联网上检索数据以及使用电子邮件发送报表。本章将为你展示如何在脚本中使用这三个Linux系统中的常见功能。

25.1 MySQL 数据库

shell脚本的问题之一是持久性数据。你可以将所有信息都保存在shell脚本变量中，但脚本运行结束后，这些变量就不存在了。有时你会希望脚本能够将数据保存下来以备后用。

过去，使用shell脚本存储和提取数据需要创建一个文件，从其中读取数据、解析数据，然后将数据存回到该文件中。在文件中搜索数据意味着要读取文件中的每一条记录进行查找。现在由于数据库非常流行，将shell脚本和有专业水准的开源数据库对接起来非常容易。Linux中最流行的开源数据库是MySQL。它是作为Linux-Apache-MySQL-PHP（LAMP）服务器环境的一部分而逐渐流行起来的。许多互联网Web服务器都采用LAMP来搭建在线商店、博客和其他Web应用。

本节将会介绍如何在Linux环境中使用MySQL数据库创建数据库对象以及如何在shell脚本中使用这些对象。

25.1.1 使用 MySQL

绝大多数Linux发行版在其软件仓库中都含有MySQL服务器和客户端软件包，这使得在Linux系统中安装完整的MySQL环境简直小菜一碟。图25-1展示了Ubuntu Linux发行版中的Add Software（添加软件）功能。

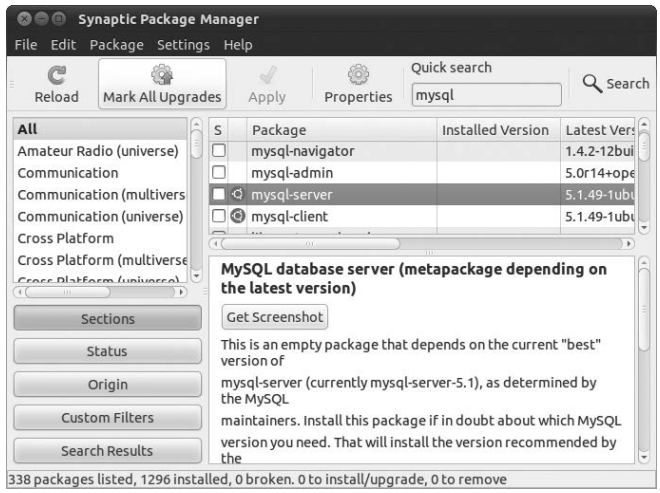

图25-1　在Ubuntu Linux系统上安装MySQL服务器

搜索到mysql-server包之后，只需要选择出现的mysql-server条目就可以了，包管理器会下载并安装完整的MySQL（包括客户端）软件。没什么比这更容易的了！

通往MySQL数据库的门户是`mysql`命令行界面程序。本节将会介绍如何使用mysql客户端程序与数据库进行交互。

1. 连接到服务器

mysql客户端程序允许你通过用户账户和密码连到网络中任何地方的MySQL数据库服务器。默认情况下，如果你在命令行上输入`mysql`，且不加任何参数，它会试图用Linux登录用户名连接运行在同一Linux系统上的MySQL服务器。

大多数情况下，这并不是你连接数据库的方式。通常还是创建一个应用程序专用的账户比较安全，不要用MySQL服务器上的标准用户账户。这样可以针对应用程序用户实施访问限制，即便应用程序出现了偏差，在必要时你也可以删除或重建。可以使用`-u`参数指定登录用户名。

```
$ mysql -u root -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 42
Server version: 5.5.38-0ubuntu0.14.04.1 (Ubuntu)

Copyright (c) 2000, 2014, 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>
```

-p参数告诉mysql程序提示输入登录用户输入密码。输入root用户账户的密码，这个密码要么是在安装过程中，要么是使用mysqladmin工具获得的。一旦登录了服务器，你就可以输入命令。

2. mysql命令

mysql程序使用两种不同类型的命令：

- 特殊的mysql命令
- 标准SQL语句

mysql程序使用它自有的一组命令，方便你控制环境和提取关于MySQL服务器的信息。这些命令要么是全名（例如status），要么是简写形式（例如\s）。你可以从mysql命令提示符中直接使用命令的完整形式或简形式。

```
mysql> \s
--------------
mysql  Ver 14.14 Distrib 5.5.38, for debian-linux-gnu (i686) using readline 6.3

Connection id:            43
Current database:
Current user:             root@localhost
SSL:                      Not in use
Current pager:            stdout
Using outfile:            ''
Using delimiter:          ;
Server version:           5.5.38-0ubuntu0.14.04.1 (Ubuntu)
Protocol version:         10
Connection:               Localhost via UNIX socket
Server characterset:      latin1
Db     characterset:      latin1
Client characterset:      utf8
Conn.  characterset:      utf8
UNIX socket:              /var/run/mysqld/mysqld.sock
Uptime:                   2 min 24 sec

Threads: 1 Questions: 575 Slow queries: 0 Opens: 421 Flush tables: 1
     Open tables: 41  Queries per second avg: 3.993
--------------

mysql>
```

mysql程序实现了MySQL服务器支持的所有标准SQL（Structured Query Language，结构化查询语言）命令。mysql程序实现的一条很棒的SQL命令是SHOW命令。你可以利用这条命令提取MySQL服务器的相关信息，比如创建的数据库和表。

```
mysql> SHOW DATABASES;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| mysql              |
+--------------------+
2 rows in set (0.04 sec)
```

```
mysql> USE mysql;
Database changed
mysql> SHOW TABLES;
+---------------------------+
| Tables_in_mysql           |
+---------------------------+
| columns_priv              |
| db                        |
| func                      |
| help_category             |
| help_keyword              |
| help_relation             |
| help_topic                |
| host                      |
| proc                      |
| procs_priv                |
| tables_priv               |
| time_zone                 |
| time_zone_leap_second     |
| time_zone_name            |
| time_zone_transition      |
| time_zone_transition_type |
| user                      |
+---------------------------+
17 rows in set (0.00 sec)
mysql>
```

在这个例子中，我们用SQL命令SHOW来显示当前在MySQL服务器上配置过的数据库，然后用SQL命令USE来连接到单个数据库。mysql会话一次只能连一个数据库。

你会注意到，在每个命令后面我们都加了一个分号。在mysql程序中，分号表明命令的结束。如果不用分号，它会提示输入更多数据。

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

mysql>
```

在处理长命令时，这个功能很有用。你可以在一行输入命令的一部分，按下回车键，然后在下一行继续输入。这样一条命令可以占任意多行，直到你用分号表明命令结束。

说明　本章中，我们用大写字母来表示SQL命令，这已经成了编写SQL命令的通用方式，但mysql程序支持用大写或小写字母来指定SQL命令。

3. 创建数据库

MySQL服务器将数据组织成*数据库*。数据库通常保存着单个应用程序的数据，与用这个数据库服务器的其他应用互不相关。为每个shell脚本应用创建一个单独的数据库有助于消除混淆，避免数据混用。

创建一个新的数据库要用如下SQL语句。

```
CREATE DATABASE name;
```

非常简单。当然，你必须拥有在MySQL服务器上创建新数据库的权限。最简单的办法是作为root用户登录MySQL服务器。

```
$ mysql -u root -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 42
Server version: 5.5.38-0ubuntu0.14.04.1 (Ubuntu)

Copyright (c) 2000, 2014, 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> CREATE DATABASE mytest;
Query OK, 1 row affected (0.02 sec)

mysql>
```

可以使用SHOW命令来查看新数据库是否创建成功。

```
mysql> SHOW DATABASES;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| mysql              |
| mytest             |
+--------------------+
3 rows in set (0.01 sec)

mysql>
```

好了，它已经成功创建了。现在你可以创建一个新的用户账户来访问新数据库了。

4. 创建用户账户

到目前为止，你已经知道了如何用root管理员账户连接到MySQL服务器。这个账户可以完全控制所有的MySQL服务器对象（就和Linux的root账户可以完全控制Linux系统一样）。

在普通应用中使用MySQL的root账户是极其危险的。如果有安全漏洞或有人弄到了root用户账户的密码，各种糟糕事情都可能发生在你的系统（以及数据）上。

为了阻止这种情况的发生，明智的做法是在MySQL上创建一个仅对应用中所涉及的数据库有权限的独立用户账户。可以用GRANT SQL语句来完成。

```
mysql> GRANT SELECT,INSERT,DELETE,UPDATE ON test.* TO test IDENTIFIED
by 'test';
Query OK, 0 rows affected (0.35 sec)

mysql>
```

这是一条很长的命令。让我们看看命令的每一部分都做了什么。

第一部分定义了用户账户对数据库有哪些权限。这条语句允许用户查询数据库数据（select权限）、插入新的数据记录以及删除和更新已有数据记录。

test.*项定义了权限作用的数据库和表。这通过下面的格式指定。

```
database.table
```

正如在这个例子中看到的，在指定数据库和表时可以使用通配符。这种格式会将指定的权限作用在名为test的数据库中的所有表上。

最后，你可以指定这些权限应用于哪些用户账户。grant命令的便利之处在于，如果用户账户不存在，它会创建。identified by部分允许你为新用户账户设定默认密码。

可以直接在mysql程序中测试新用户账户。

```
$ mysql mytest -u test -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 42
Server version: 5.5.38-0ubuntu0.14.04.1 (Ubuntu)

Copyright (c) 2000, 2014, 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>
```

第一个参数指定使用的默认数据库（mytest）。如你所见，-u选项定义了登录的用户，-p用来提示输入密码。输入test用户账户的密码后，你就连到了服务器。

现在已经有了数据库和用户账户，可以为数据创建一些表了。

5. 创建数据表

MySQL是一种关系数据库（relational database）。在关系数据库中，数据按照字段、记录和表进行组织。数据字段是信息的单个组成部分，比如员工的姓或工资。记录是相关数据字段的集合，比如员工ID号、姓、名、地址和工资。每条记录都代表一组数据字段。

表含有保存相关数据的所有记录。因此，你会使用一个叫作Employees的表来保存每个员工的记录。

要在数据库中新建一张新表，需要用SQL命令CREATE TABLE。

```
$ mysql mytest -u root -p
Enter password:
mysql> CREATE TABLE employees (
  -> empid int not null,
  -> lastname varchar(30),
  -> firstname varchar(30),
  -> salary float,
  -> primary key (empid));
Query OK, 0 rows affected (0.14 sec)

mysql>
```

首先要注意，为了新建一张表，我们需要用root用户账户登录到MySQL上，因为test用户没有新建表的权限。接下来，我们在mysql程序命令行上指定了test数据库。不这么做的话，就需要用SQL命令USE来连接到test数据库。

警告 在创建新表前，很重要的一点是，要确保你使用了正确的数据库。另外还要确保使用管理员用户账户（MySQL中的root用户）登录来创建表。

表中的每个数据字段都用数据类型来定义。MySQL和PostgreSQL数据库支持许多不同的数据类型。表25-1列出了其中较常见的一些数据类型。

表25-1 MySQL的数据类型

数据类型	描 述
char	定长字符串值
varchar	变长字符串值
int	整数值
float	浮点值
boolean	布尔类型true/false值
date	YYYY-MM-DD格式的日期值
time	HH:mm:ss格式的时间值
timestamp	日期和时间值的组合
text	长字符串值
BLOB	大的二进制值，比如图片或视频剪辑

empid数据字段还指定了一个数据约束（data constraint）。数据约束会限制输入什么类型数据可以创建一个有效的记录。not null数据约束指明每条记录都必须有一个指定的empid值。

最后，primary key定义了可以唯一标识每条记录的数据字段。这意味着每条记录中在表中都必须有一个唯一的empid值。

创建新表之后，可以用对应的命令来确保它创建成功了，在mysql中是用show tables命令。

```
mysql> show tables;
+----------------+
| Tables_in_test |
+----------------+
| employees      |
+----------------+
1 row in set (0.00 sec)

mysql>
```

有了新建的表，现在你可以开始保存一些数据了。下一节将会介绍应该怎么做。

6. 插入和删除数据

毫不意外，你需要使用SQL命令INSERT向表中插入新的记录。每条INSERT命令都必须指定数据字段值来供MySQL服务器接受该记录。

SQL命令INSERT的格式如下。

```
INSERT INTO table VALUES (...)
```

每个数据字段的值都用逗号分开。

```
$ mysql mytest -u test -p
Enter password:

mysql> INSERT INTO employees VALUES (1, 'Blum', 'Rich', 25000.00);
Query OK, 1 row affected (0.35 sec)
```

上面的例子用-u命令行选项以mytest用户账户登录。

INSERT命令会将指定的数据写入表中的数据字段里。如果你试图添加另外一条包含相同的empid数据字段值的记录，就会得到一条错误消息。

```
mysql> INSERT INTO employees VALUES (1, 'Blum', 'Barbara', 45000.00);
ERROR 1062 (23000): Duplicate entry '1' for key 1
```

但如果你将empid的值改成唯一的值，那就没问题了。

```
mysql> INSERT INTO employees VALUES (2, 'Blum', 'Barbara', 45000.00);
Query OK, 1 row affected (0.00 sec)
```

现在表中应该有两条记录了。

如果你需要从表中删除数据，可以用SQL命令DELETE，但要非常小心。

DELETE命令的基本格式如下。

```
DELETE FROM table;
```

其中*table*指定了要从中删除记录的表。这个命令有个小问题：它会删除该表中所有记录。

要想只删除其中一条或多条数据行，必须用WHERE子句。WHERE子句允许创建一个过滤器来指定删除哪些记录。可以像下面这样使用WHERE子句。

```
DELETE FROM employees WHERE empid = 2;
```

这条命令只会删除empid值为2的所有记录。当你执行这条命令时，mysql程序会返回一条消息来说明有多少个记录符合条件。

```
mysql> DELETE FROM employees WHERE empid = 2;
Query OK, 1 row affected (0.29 sec)
```

跟期望的一样，只有一条记录符合条件并被删除。

7. 查询数据

一旦将所有数据都放入数据库，就可以开始提取信息了。

所有查询都是用SQL命令SELECT来完成。SELECT命令非常强大，但用起来也很复杂。

SELECT语句的基本格式如下。

```
SELECT datafields FROM table
```

*datafields*参数是一个用逗号分开的数据字段名称列表，指明了希望查询返回的字段。如果你要提取所有的数据字段值，可以用星号作通配符。

你还必须指定要查询的表。要想得到有意义的结果，待查询的数据字段必须对应正确的表。

默认情况下，SELECT命令会返回指定表中的所有记录。

```
mysql> SELECT * FROM employees;
+-------+----------+------------+--------+
| empid | lastname | firstname  | salary |
+-------+----------+------------+--------+
|     1 | Blum     | Rich       |  25000 |
|     2 | Blum     | Barbara    |  45000 |
|     3 | Blum     | Katie Jane |  34500 |
|     4 | Blum     | Jessica    |  52340 |
+-------+----------+------------+--------+
4 rows in set (0.00 sec)

mysql>
```

可以用一个或多个修饰符定义数据库服务器如何返回查询数据。下面列出了常用的修饰符。

- WHERE：显示符合特定条件的数据行子集。
- ORDER BY：以指定顺序显示数据行。
- LIMIT：只显示数据行的一个子集。

WHERE子句是最常用的SELECT命令修饰符。它允许你指定查询结果的过滤条件。下面是一个使用WHERE子句的例子。

```
mysql> SELECT * FROM employees WHERE salary > 40000;
+-------+----------+-----------+--------+
| empid | lastname | firstname | salary |
+-------+----------+-----------+--------+
|     2 | Blum     | Barbara   |  45000 |
|     4 | Blum     | Jessica   |  52340 |
+-------+----------+-----------+--------+
2 rows in set (0.01 sec)

  mysql>
```

现在你可以看到将数据库访问功能添加到shell脚本中的强大之处了！只要使用几条SQL命令和mysql程序就可以轻松应对你的数据管理需求。下一节将会介绍如何将这些功能引入shell脚本。

25.1.2　在脚本中使用数据库

现在你已经有了一个可以正常工作的数据库，终于可以将精力放回shell脚本编程了。本节将会介绍如何用shell脚本同数据库交互。

1. 登录到服务器

如果你为自己的shell脚本在MySQL中创建了一个特定的用户账户，那你需要使用mysql命令，以该用户的身份登录。实现的方法有好几种，其中一种是使用-p选项，在命令行中加入密码。

```
mysql mytest -u test -p test
```

不过这并不是一个好做法。所有能够访问你脚本的人都会知道数据库的用户账户和密码。

要解决这个问题，可以借助mysql程序所使用的一个特殊配置文件。mysql程序使用$HOME/.my.cnf文件来读取特定的启动命令和设置。其中一项设置就是用户启动的mysql会话的默认密码。

要想在这个文加中设置默认密码，只需要像下面这样。

```
$ cat .my.cnf
[client]
password = test
$ chmod 400 .my.cnf
$
```

可以使用chmod命令将.my.cnf文件限制为只能由本人浏览。现在可以在命令行上测试一下。

```
$ mysql mytest -u test
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 44
Server version: 5.5.38-0ubuntu0.14.04.1 (Ubuntu)

Copyright (c) 2000, 2014, 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>
```

棒极了！这样就不用在shell脚本中将密码写在命令行上了。

2. 向服务器发送命令

在建立起到服务器的连接后，接着就可以向数据库发送命令进行交互。有两种实现方法：

- 发送单个命令并退出；
- 发送多个命令。

要发送单个命令，你必须将命令作为mysql命令行的一部分。对于mysql命令，可以用-e选项。

```
$ cat mtest1
#!/bin/bash
# send a command to the MySQL server

MYSQL=$(which mysql)

$MYSQL mytest -u test -e 'select * from employees'
$ ./mtest1
+-------+----------+------------+---------+
| empid | lastname | firstname  | salary  |
+-------+----------+------------+---------+
|     1 | Blum     | Rich       | 25000   |
|     2 | Blum     | Barbara    | 45000   |
|     3 | Blum     | Katie Jane | 34500   |
|     4 | Blum     | Jessica    | 52340   |
+-------+----------+------------+---------+
$
```

数据库服务器会将SQL命令的结果返回给shell脚本，脚本会将它们显示在STDOUT中。

如果你需要发送多条SQL命令，可以利用文件重定向（参见第15章）。要在shell脚本中重定向多行内容，就必须定义一个结束（end of file）字符串。结束字符串指明了重定向数据的开始和结尾。

下面的例子定义了结束字符串及其中数据。

```
$ cat mtest2
#!/bin/bash
# sending multiple commands to MySQL

MYSQL=$(which mysql)
$MYSQL mytest -u test <<EOF
show tables;
select * from employees where salary > 40000;
EOF
$ ./mtest2
Tables_in_test
employees
empid    lastname    firstname    salary
2        Blum        Barbara      45000
4        Blum        Jessica      52340
$
```

shell会将EOF分隔符之间的所有内容都重定向给mysql命令。mysql命令会执行这些命令行，就像你在提示符下亲自输入的一样。用了这种方法，你可以根据需要向MySQL服务器发送任意多条命令。但你会注意到，每条命令的输出之间没有没有任何分隔。在25.2.3节中，你会看到如何解决这个问题。

说明 你应该也注意到了，当使用输入重定向时，mysql程序改变了默认的输出风格。mysql程序检测到了输入是重定向过来的，所以它只返回了原始数据而不是在数据两边加上ASCII符号框。这非常有利于提取个别的数据元素。

当然，并不是只能从数据表中提取数据。你可以在脚本中使用任何类型的SQL命令，比如INSERT语句。

```
$ cat mtest3
#!/bin/bash
# send data to the table in the MySQL database

MYSQL=$(which mysql)

if [ $# -ne 4 ]
then
 echo "Usage: mtest3 empid lastname firstname salary"
else
 statement="INSERT INTO employees VALUES ($1, '$2', '$3', $4)"
 $MYSQL mytest -u test << EOF
 $statement
EOF
 if [ $? -eq 0 ]
 then
    echo Data successfully added
 else
    echo Problem adding data
 fi
fi
$ ./mtest3
Usage: mtest3 empid lastname firstname salary
$ ./mtest3 5 Blum Jasper 100000
Data added successfully
$
$ ./mtest3 5 Blum Jasper 100000
ERROR 1062 (23000) at line 1: Duplicate entry '5' for key 1
Problem adding data
$
```

这个例子演示了使用这种方法的一些注意事项。在指定结束字符串时，它必须是该行唯一的内容，并且该行必须以这个字符串开头。如果我们将EOF文本缩进以和其余的if-then缩进对齐，它就不会起作用了。

注意，在INSERT语句里，我们在文本值周围用了单引号，在整个INSERT语句周围用了双引号。一定不要弄混引用字符串值的引号和定义脚本变量文本的引号。

还有，注意我们是怎样使用$?特殊变量来测试mysql程序的退出状态码的。它有助于你判断命令是否成功执行。

将这些命令的结果发送到STDOUT并不是管理和操作数据最简单的方法。下一节将会为你展示一些技巧，帮助脚本获取从数据库中检索到的数据。

3. 格式化数据

mysql命令的标准输出并不太适合提取数据。如果要对提取到的数据进行处理，你需要做一些特别的操作。本节将会介绍一些技巧来帮你从数据库报表中提取数据。

提取数据库数据的第一步是将mysql命令的输出重定向到一个环境变量中。这允许你在其他

命令中使用输出信息。这里有个例子。

```
$ cat mtest4
#!/bin/bash
# redirecting SQL output to a variable

MYSQL=$(which mysql)

dbs=$($MYSQL mytest -u test -Bse 'show databases')
for db in $dbs
do
 echo $db
done
$ ./mtest4
information_schema
test
$
```

这个例子在mysql程序的命令行上用了两个额外参数。-B选项指定mysql程序工作在批处理模式运行，-s（silent）选项用于禁止输出列标题和格式化符号。

通过将mysql命令的输出重定向到一个变量，此例可以逐步输出每条返回记录里的每个值。

mysql程序还支持另外一种叫作可扩展标记语言（Extensive Markup Language，XML）的流行格式。这种语言使用和HTML类似的标签来标识数据名和值。

对于mysql程序，可以用-X命令行选项来输出。

```
$ mysql mytest -u test -X -e 'select * from employees where empid = 1'
<?xml version="1.0"?>

<resultset statement="select * from employees">
<row>
    <field name="empid">1</field>
    <field name="lastname">Blum</field>
    <field name="firstname">Rich</field>
    <field name="salary">25000</field>
</row>
</resultset>
$
```

通过使用XML，你能够轻松标识出每条记录以及记录中的各个字段值。然后你就可以使用标准的Linux字符串处理功能来提取需要的数据。