MySQL 之终端：管理数据库的基本操作

MySQL 有很多的可视化管理工具，比如“mysql-workbench”和“sequel-pro-”。 现在我写 MySQL 的终端命令操作的文章，是想强化一下自己对于 MySQL 的理解，总会比使用图形化的理解透彻，因为我本来就比较喜欢写代码。同时写出来这些文章，是想要给大家当个参考，希望也能对大家有所帮助，有所提升，这就是我为什么要写终端操作 MySQL 的文章了。

注意：MySQL 数据库命令不区分大小写。但在 MAC 的终端，如果你想使用 tab 自动补全命令，那么你就必须使用大写，这样 MAC 的终端才会帮你补全命令，否则你按 N 遍 tab 都不会有响应。

1、数据库（database）管理

1.1 create 创建数据库

1.2 show 查看所有数据库

1.3 alter 修改数据库

1.4 use 使用数据库

1.5 查看当前使用的数据库

1.6 drop 删除数据库

2、数据表（table）管理

2.1 create 创建表

2.2 show 显示表

2.3 desc 查看表结构

2.4 alter 修改表结构（增、删、改）

2.4.1 insert 在表中添加列（字段）

2.4.2 alter 修改表（列）字段

2.4.3 delete 删除表（列）字段

2.4.4 rename 重命名表名

2.5 create 利用已有数据创建新表

3、数据的操作及管理

3.1 增加数据（增）

3.2 删除数据（删）

3.3 修改数据（改）

3.4 查询数据（查）

4、管理视图

1、数据库（database）管理

1.1 create 创建数据库

```
1create database firstDB;
```

1.2 show 查看所有数据库

```
1 mysql> show databases;
2 +--------------------+
3 | Database |
4 +--------------------+
5 | information_schema |
6 | firstDB |
7 | mysql |
8 | performance_schema |
9 +--------------------+
104 rows in set (0.00 sec)
```

1.3 alter 修改数据库

alter 命令修改数据库编码：

默认创建的数据库默认不支持中文字符，如果我们需要它支持中文字符，则将它的编码设置为 utf8 格式：

```
1mysql> ALTER DATABASE testDB CHARACTER SET UTF8;
```

```
2Query OK, 1 row affected (0.00 sec)
```

1.4 use 使用数据库

```
1mysql> use firstDB;
2Database changed
```

1.5 查看当前使用的数据库

```
1mysql> select database();
2+------------+
3| database() |
4+------------+
5| firstdb |
6+------------+
71 row in set (0.00 sec)
```

1.6 drop 删除数据库

```
1mysql> drop database firstDB;
2Query OK, 0 rows affected (0.00 sec)
```

2、数据表（table）管理

我们首先创建一个数据库，提供我们往后的使用：

```
1mysql> create database testDB;
2Query OK, 1 row affected (0.00 sec)
```

创建后记得用 use 命令进入（使用）数据库，不然后面的操作都会不成功的。

2.1 create 创建表

```
1mysql> create table PEOPLE (
2  -> ID int AUTO_INCREMENT PRIMARY KEY,
3  -> NAME varchar(20) not null,
4  -> AGE int not null,
5  -> BIRTHDAY datetime);
6Query OK, 0 rows affected (0.01 sec)
```

2.2 show 显示表

显示当前数据库所有的数据表

```
1mysql> show tables;
```

```
2+-----------------+
3| Tables_in_testdb |
4+-----------------+
5| PEOPLE |
6+-----------------+
71 row in set (0.00 sec)
```

2.3 desc 查看表结构

```
1 mysql> desc PEOPLE
2    -> ;
3 +----------+-------------+------+-----+---------+----------------+
4 | Field | Type | Null | Key | Default | Extra |
5 +----------+-------------+------+-----+---------+----------------+
6 | ID | int(11) | NO | PRI | NULL | auto_increment |
7 | NAME | varchar(20) | NO | | NULL | |
8 | AGE | int(11) | NO | | NULL | |
9 | BIRTHDAY | datetime | YES | | NULL | |
10+----------+-------------+------+-----+---------+----------------+
114 rows in set (0.01 sec)
```

2.4 alter 修改表结构（增、删、改）

默认创建的表不支持中文字符，所以需将表编码设置为 utf8：

```
1mysql> ALTER TABLE KEYCHAIN CONVERT TO CHARACTER SET UTF8;
2Query OK, 1 row affected (0.02 sec)
3Records: 1 Duplicates: 0 Warnings: 0
```

2.4.1 insert 在表中添加列（字段）

```
1mysql> alter table PEOPLE add star BOOL;
2Query OK, 0 rows affected (0.02 sec)
3Records: 0 Duplicates: 0 Warnings: 0
```

提示：在 MySQL 里，布尔类型会自动转换为 tinyint(1)类型。

我们不妨使用 desc 去查看一下 PEOPLE 表结构：

```
1 mysql> desc PEOPLE;
2 +----------+-------------+------+-----+---------+----------------+
3 | Field | Type | Null | Key | Default | Extra |
4 +----------+-------------+------+-----+---------+----------------+
5 | ID | int(11) | NO | PRI | NULL | auto_increment |
6 | NAME | varchar(20) | NO | | NULL | |
```

```
7 | AGE | int(11) | NO | | NULL | |
8 | BIRTHDAY | datetime | YES | | NULL | |
9 | star | tinyint(1) | YES | | NULL | |
10+----------+-------------+------+-----+---------+----------------+
115 rows in set (0.00 sec)
```

现在，你该相信我了吧？

2.4.2 alter 修改表（列）字段

```
1mysql> alter table PEOPLE MODIFY star int;
2Query OK, 0 rows affected (0.01 sec)
3Records: 0 Duplicates: 0 Warnings: 0
```

也可以指定 int(n) 的长度，比如 int(2)。

我们再次使用 desc 查看 PEOPLE 表结构：

```
1 mysql> desc PEOPLE;
2 +----------+-------------+------+-----+---------+----------------+
3 | Field | Type | Null | Key | Default | Extra |
4 +----------+-------------+------+-----+---------+----------------+
5 | ID | int(11) | NO | PRI | NULL | auto_increment |
6 | NAME | varchar(20) | NO | | NULL | |
7 | AGE | int(11) | NO | | NULL | |
8 | BIRTHDAY | datetime | YES | | NULL | |
9 | star | int(11) | YES | | NULL | |
10+----------+-------------+------+-----+---------+----------------+
115 rows in set (0.00 sec)
```

2.4.3 delete 删除表（列）字段

```
1mysql> alter table PEOPLE DROP column star;
2Query OK, 0 rows affected (0.02 sec)
3Records: 0 Duplicates: 0 Warnings: 0
```

删除后，再次查看 PEOPLE 表结构：

```
1 mysql> desc PEOPLE;
2 +----------+-------------+------+-----+---------+----------------+
3 | Field | Type | Null | Key | Default | Extra |
4 +----------+-------------+------+-----+---------+----------------+
5 | ID | int(11) | NO | PRI | NULL | auto_increment |
6 | NAME | varchar(20) | NO | | NULL | |
7 | AGE | int(11) | NO | | NULL | |
```

```
8 | BIRTHDAY | datetime | YES | | NULL | |
9 +----------+------------+------+-----+---------+----------------+
104 rows in set (0.00 sec)
```

删除字段成功，现在我们已经不能看到 star 的字段了。

2.4.4 rename 重命名表名

```
1mysql> RENAME TABLE PEOPLE TO NEW_PEOPLE;
2Query OK, 0 rows affected (0.00 sec)
```

2.4.5 null or not null

修改表字段允许为空或不允许为空：

```
1mysql> ALTER TABLE PEOPLE MODIFY AGE INT(3) NULL;
2Query OK, 0 rows affected (0.01 sec)
3Records: 0 Duplicates: 0 Warnings: 0
```

把 PEOPLE 表的 AGE 字段设置成“允许为空”，即插入记录时这个字段可以不录入。否则相反。

它的格式为：ALTER TABLE <TALBE_NAME> MODIFY <CLOUMN> <NULL ‘OR’ NOT NULL>

2.5 create 利用已有数据创建新表

```
1mysql> create table newTable select * from PEOPLE;
2Query OK, 0 rows affected (0.01 sec)
3Records: 0 Duplicates: 0 Warnings: 0
```

我们查看一下目前数据库存在的表：

```
1mysql> show tables;
2+-----------------+
3| Tables_in_testdb |
4+-----------------+
5| PEOPLE |
6| newTable |
7+-----------------+
82 rows in set (0.00 sec)
```

3、数据的操作及管理

数据表的基本操作，包含增、删、改、查数据。

以下命令均在 PEOPLE 表上操作。

3.1 增加数据（增）

PEOPLE 表目前是没有数据的，它是空的数据表，我们现在先添加一些数据。

insert into 命令添加数据：

```
1mysql> insert into PEOPLE VALUES (null, 'Anny', 22, '1992-05-22');
2Query OK, 1 row affected (0.00 sec)
```

使用 select 命令查看表（会在后面介绍），现在我们查看 PEOPLE 数据表的数据：

```
1mysql> select * from PEOPLE;
2+----+------+-----+---------------------+
3| ID | NAME | AGE | BIRTHDAY |
4+----+------+-----+---------------------+
5| 1 | Anny | 22 | 1992-05-22 00:00:00 |
6+----+------+-----+---------------------+
71 row in set (0.00 sec)
```

数据表现在有一条数据。

我们多添加几条数据，如：

```
1 mysql> select * from PEOPLE;
2 +----+--------+-----+---------------------+
3 | ID | NAME | AGE | BIRTHDAY |
4 +----+--------+-----+---------------------+
5 | 1 | Anny | 22 | 1992-05-22 00:00:00 |
6 | 2 | Garvey | 23 | 1991-05-22 00:00:00 |
7 | 3 | Lisa | 25 | 1989-05-22 00:00:00 |
8 | 4 | Nick | 24 | 1990-05-22 00:00:00 |
9 | 5 | Rick | 24 | 1991-05-22 00:00:00 |
10+----+--------+-----+---------------------+
115 rows in set (0.00 sec)
```

3.2 删除数据（删）

delete 命令删除数据：

```
1mysql> delete from PEOPLE where name = 'Lisa';
2Query OK, 1 row affected (0.01 sec)
```

再次查询 PEOPLE 表：

```
1 mysql> select * from PEOPLE;
2 +----+--------+-----+---------------------+
3 | ID | NAME | AGE | BIRTHDAY |
4 +----+--------+-----+---------------------+
5 | 1 | Anny | 22 | 1992-05-22 00:00:00 |
6 | 2 | Garvey | 23 | 1991-05-22 00:00:00 |
7 | 4 | Nick | 24 | 1990-05-22 00:00:00 |
8 | 5 | Rick | 24 | 1991-05-22 00:00:00 |
9 +----+--------+-----+---------------------+
104 rows in set (0.00 sec)
```

已经看不到名为“Lisa”的数据了。

3.3 修改数据（改）

update 命令修改数据：

```
1mysql> update PEOPLE set name='Calvin' where name = 'Garvey';
2Query OK, 1 row affected (0.00 sec)
3Rows matched: 1 Changed: 1 Warnings: 0
```

查询 PEOPLE 表内容：

```
1 mysql> select * from PEOPLE;
2 +----+--------+-----+---------------------+
3 | ID | NAME | AGE | BIRTHDAY |
4 +----+--------+-----+---------------------+
5 | 1 | Anny | 22 | 1992-05-22 00:00:00 |
6 | 2 | Calvin | 23 | 1991-05-22 00:00:00 |
7 | 4 | Nick | 24 | 1990-05-22 00:00:00 |
8 | 5 | Rick | 24 | 1991-05-22 00:00:00 |
9 +----+--------+-----+---------------------+
104 rows in set (0.00 sec)
```

名为“Garvey”的记录已经修改为“Calvin”。

3.4 查询数据（查）

select 命令查询数据，最简单的就是查询表的所有数据，也就是我们最初使用到的那条命令：

```
1 mysql> select * from PEOPLE;
2 +----+--------+-----+---------------------+
3 | ID | NAME | AGE | BIRTHDAY |
4 +----+--------+-----+---------------------+
```

```
5 | 1 | Anny | 22 | 1992-05-22 00:00:00 |
6 | 2 | Calvin | 23 | 1991-05-22 00:00:00 |
7 | 4 | Nick | 24 | 1990-05-22 00:00:00 |
8 | 5 | Rick | 24 | 1991-05-22 00:00:00 |
9 +----+--------+-----+---------------------+
104 rows in set (0.00 sec)
```

格式：select * from <表名>，*代表所有字段。

查询数据时也可指定显示的（列）字段：

```
1 mysql> select NAME, AGE, BIRTHDAY from PEOPLE;
2 +--------+-----+---------------------+
3 | NAME | AGE | BIRTHDAY |
4 +--------+-----+---------------------+
5 | Anny | 22 | 1992-05-22 00:00:00 |
6 | Calvin | 23 | 1991-05-22 00:00:00 |
7 | Nick | 24 | 1990-05-22 00:00:00 |
8 | Rick | 24 | 1991-05-22 00:00:00 |
9 +--------+-----+---------------------+
104 rows in set (0.00 sec)
```

格式：select <字段名,字段名,…> from <表名>。

select 查询命令还有很多的高级用法，比如用来查找不重复（distinct）的数据，使数据按条件排序（order by），按查询条件显示数据（where）等等。这些都会在下一篇文章作重点介绍，请大家继续留意我的博客，谢谢。

4、管理视图

创建视图

视图是从数据库里导出一个或多个表的虚拟表，是用来方便用户对数据的操作。

```
1mysql> CREATE VIEW PEOPLE_VIEW (
2  -> NAME, AGE)
3  -> AS SELECT NAME, AGE FROM PEOPLE;
```

创建成功后查看视图。

```
1 PEOPLE PEOPLE.AGE PEOPLE.BIRTHDAY PEOPLE.ID PEOPLE.NAME
2 mysql> SELECT * FROM PEOPLE_VIEW
3   -> ;
4 +--------+-----+
5 | NAME | AGE |
```

```
6 +--------+-----+
7 | Anny | 22 |
8 | Calvin | 23 |
9 | Nick | 24 |
10| Rick | 24 |
11+--------+-----+
124 rows in set (0.00 sec)
```

我们也可以使用 DESC 命令查看视图的结构。

```
1mysql> DESC PEOPLE_VIEW;
2+-------+---------+------+-----+---------+-------+
3| Field | Type | Null | Key | Default | Extra |
4+-------+---------+------+-----+---------+-------+
5| ID | int(11) | NO | | 0 | |
6+-------+---------+------+-----+---------+-------+
71 row in set (0.01 sec)
```

替换视图

创建或替换原有视图。

```
 mysql> CREATE OR REPLACE VIEW
1PEOPLE_VIEW(PEOPLE_ID,PEOPLE_NAME,PEOPLE_AGE) AS SELECT ID,NAME,AGE
2FROM PEOPLE;
 Query OK, 0 rows affected (0.00 sec)
```

创建或替换后查看视图。

```
1 mysql> SELECT * FROM PEOPLE_VIEW;
2 +-----------+-------------+------------+
3 | PEOPLE_ID | PEOPLE_NAME | PEOPLE_AGE |
4 +-----------+-------------+------------+
5 | 1 | Anny | 22 |
6 | 2 | Calvin | 23 |
7 | 4 | Nick | 24 |
8 | 5 | Rick | 24 |
9 +-----------+-------------+------------+
104 rows in set (0.00 sec)
```

操作视图

当视图数据有变化时（增、删、改），真实的表数据也会随着改变。也就是说，对视图的操作就是对表的数据，所以我们可以把视图当作表。

例：往视图插入一条数据。

```
1mysql> INSERT INTO PEOPLE_VIEW VALUES(NULL, 'Kerry', '33');
2Query OK, 1 row affected (0.00 sec)
```

插入数据成功后查看视图。

```
1 mysql> SELECT * FROM PEOPLE_VIEW ;
2 +-----------+-------------+------------+
3 | PEOPLE_ID | PEOPLE_NAME | PEOPLE_AGE |
4 +-----------+-------------+------------+
5 | 1 | Anny | 22 |
6 | 2 | Calvin | 23 |
7 | 4 | Nick | 24 |
8 | 5 | Rick | 24 |
9 | 6 | Kerry | 33 |
10+-----------+-------------+------------+
115 rows in set (0.00 sec)
```

可以在视图上看到我们刚刚插入的数据，现在我们就来验证一下真实的表是否也会作出变化。

```
1 mysql> SELECT * FROM PEOPLE;
2 +----+--------+-----+---------------------+
3 | ID | NAME | AGE | BIRTHDAY |
4 +----+--------+-----+---------------------+
5 | 1 | Anny | 22 | 1992-05-22 00:00:00 |
6 | 2 | Calvin | 23 | 1991-05-22 00:00:00 |
7 | 4 | Nick | 24 | 1990-05-22 00:00:00 |
8 | 5 | Rick | 24 | 1991-05-22 00:00:00 |
9 | 6 | Kerry | 33 | NULL |
10+----+--------+-----+---------------------+
115 rows in set (0.00 sec)
```

可见，真实的表数据也已经有所改变，刚刚往视图里插入的那一条数据存在于真实表中，真理便是：对视图的操作就是对表的数据。

删除视图

```
1mysql> DROP VIEW PEOPLE_VIEW;
2Query OK, 0 rows affected (0.00 sec)
```