

MySQL之终端（Terminal）管理数据库、数据表、数据的基本操作

8 回复 42 查看



(<https://www.shiyanlou.com/user/8490>) 实验楼管理员 (<https://www.shiyanlou.com/vip>) 6小时前

技术分享 (<https://www.shiyanlou.com/questions/?tag=技术分享>)

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。

分享到微博

全部回答



实验楼管理员 (<https://www.shiyanlou.com/user/8490>) (<https://www.shiyanlou.com/vip>)

(<https://www.shiyanlou.com/user/8490>)

1、数据库（database）管理

1.1 create 创建数据库

```
create database firstDB;
```

1.2 show 查看所有数据库

```
mysql> show databases;

+-----+
| Database          |
+-----+
| information_schema |
| firstDB            |
| mysql              |
| performance_schema |
+-----+
4 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，则将其编码设置为utf8格式：

```
mysql> ALTER DATABASE testDB CHARACTER SET UTF8;
Query OK, 1 row affected (0.00 sec)
```

1.4 use 使用数据库

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

1.5 查看当前使用的数据库

```
mysql> select database();
+-----+
| database() |
+-----+
| firstdb    |
+-----+
1 row in set (0.00 sec)
```

1.6 drop 删除数据库

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

6小时前



实验楼管理员 (<https://www.shiyanlou.com/user/8490>) [vip](https://www.shiyanlou.com/vip)

(<https://www.shiyanlou.com/user/8490>)

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：

```
mysql> create database testDB;
Query 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

```
mysql> create table PEOPLE (
-> ID int AUTO_INCREMENT PRIMARY KEY,
-> NAME varchar(20) not null,
-> AGE int not null,
-> BIRTHDAY datetime);
Query 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

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

2.3 desc 查看表结构

```
mysql> desc PEOPLE
-> ;
+-----+-----+-----+-----+-----+-----+
| Field      | Type          | Null | Key | Default | Extra          |
+-----+-----+-----+-----+-----+-----+
| ID         | int(11)       | NO   | PRI | NULL    | auto_increment |
| NAME       | varchar(20)   | NO   |     | NULL    |                |
| AGE        | int(11)       | NO   |     | NULL    |                |
| BIRTHDAY   | datetime      | YES  |     | NULL    |                |
+-----+-----+-----+-----+-----+-----+
4 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：

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

6小时前



实验楼管理员 (<https://www.shiyanlou.com/user/8490>) [vip](https://www.shiyanlou.com/vip)

(<https://www.shiyanlou.com/user/8490>)

2.4.1 Insert 在表中添加列 (字段)

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

```
mysql> desc PEOPLE;
+-----+-----+-----+-----+-----+-----+
| Field      | Type          | Null | Key | Default | Extra          |
+-----+-----+-----+-----+-----+-----+
| ID         | int(11)       | NO   | PRI | NULL    | auto_increment |
| NAME      | varchar(20)   | NO   |     | NULL    |                |
| AGE       | int(11)       | NO   |     | NULL    |                |
| BIRTHDAY  | datetime      | YES  |     | NULL    |                |
| star      | tinyint(1)    | YES  |     | NULL    |                |
+-----+-----+-----+-----+-----+-----+
5 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

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

```
mysql> desc PEOPLE;
+-----+-----+-----+-----+-----+-----+
| Field      | Type          | Null | Key | Default | Extra          |
+-----+-----+-----+-----+-----+-----+
| ID         | int(11)       | NO   | PRI | NULL    | auto_increment |
| NAME      | varchar(20)   | NO   |     | NULL    |                |
| AGE       | int(11)       | NO   |     | NULL    |                |
| BIRTHDAY  | datetime      | YES  |     | NULL    |                |
| star      | int(11)       | YES  |     | NULL    |                |
+-----+-----+-----+-----+-----+-----+
5 rows in set (0.00 sec)
```

6小时前



实验楼管理员 (<https://www.shiyanlou.com/user/8490>) [vip](https://www.shiyanlou.com/vip)

(<https://www.shiyanlou.com/user/8490>)

2.4.3 delete 删除表 (列) 字段

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

删除后，再次查看PEOPLE表结构：

```
mysql> desc PEOPLE;
+-----+-----+-----+-----+-----+-----+
| Field      | Type          | Null | Key | Default | Extra          |
+-----+-----+-----+-----+-----+-----+
| ID         | int(11)       | NO   | PRI | NULL    | auto_increment |
| NAME      | varchar(20)   | NO   |     | NULL    |                |
| AGE       | int(11)       | NO   |     | NULL    |                |
| BIRTHDAY  | datetime      | YES  |     | NULL    |                |
+-----+-----+-----+-----+-----+-----+
4 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

```
mysql> RENAME TABLE PEOPLE TO NEW_PEOPLE;
Query 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：

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

2.5 create 利用已有数据创建新表

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

我们查看一下目前数据库存在的表：

```
mysql> show tables;
+-----+
| Tables_in_testdb |
+-----+
| PEOPLE           |
| newTable         |
+-----+
2 rows in set (0.00 sec)
```

6小时前



实验楼管理员 (<https://www.shiyanlou.com/user/8490>) (<https://www.shiyanlou.com/vip>)

(<https://www.shiyanlou.com/user/8490>)

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：

```
mysql> insert into PEOPLE VALUES (null, 'Anny', 22, '1992-05-22');
Query 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：

```
mysql> select * from PEOPLE;
+-----+-----+-----+-----+
| ID | NAME | AGE | BIRTHDAY          |
+-----+-----+-----+-----+
| 1 | Anny | 22 | 1992-05-22 00:00:00 |
+-----+-----+-----+-----+
1 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：

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

3.2 删除数据（删）

delete 命令删除数据：

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


再次查询PEOPLE表：

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

已经看不到名为“Lisa”的数据了。

6小时前



实验楼管理员 (<https://www.shiyanlou.com/user/8490>)  (<https://www.shiyanlou.com/vip>)

(<https://www.shiyanlou.com/user/8490>)

3.3 修改数据（改）

update 命令修改数据：

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

查询PEOPLE表内容：

```
mysql> select * from PEOPLE;
+-----+-----+-----+-----+
| ID | NAME | AGE | BIRTHDAY |
+-----+-----+-----+-----+
| 1 | Anny | 22 | 1992-05-22 00:00:00 |
| 2 | Calvin | 23 | 1991-05-22 00:00:00 |
| 4 | Nick | 24 | 1990-05-22 00:00:00 |
| 5 | Rick | 24 | 1991-05-22 00:00:00 |
+-----+-----+-----+-----+
4 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：

```
mysql> select * from PEOPLE;
+-----+-----+-----+-----+
| ID | NAME | AGE | BIRTHDAY |
+-----+-----+-----+-----+
| 1 | Anny | 22 | 1992-05-22 00:00:00 |
| 2 | Calvin | 23 | 1991-05-22 00:00:00 |
| 4 | Nick | 24 | 1990-05-22 00:00:00 |
| 5 | Rick | 24 | 1991-05-22 00:00:00 |
+-----+-----+-----+-----+
4 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：

```
mysql> select NAME, AGE, BIRTHDAY from PEOPLE;
+-----+-----+-----+
| NAME | AGE | BIRTHDAY |
+-----+-----+-----+
| Anny | 22 | 1992-05-22 00:00:00 |
| Calvin | 23 | 1991-05-22 00:00:00 |
| Nick | 24 | 1990-05-22 00:00:00 |
| Rick | 24 | 1991-05-22 00:00:00 |
+-----+-----+-----+
4 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。

6小时前



实验楼管理员 (<https://www.shiyanlou.com/user/8490>) 💛 (<https://www.shiyanlou.com/vip>)

(<https://www.shiyanlou.com/user/8490>)

4、管理视图

4.1 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。

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

创建成功后查看视图。

```

PEOPLE          PEOPLE.AGE      PEOPLE.BIRTHDAY PEOPLE.ID      PEOPLE.NAME
mysql> SELECT * FROM PEOPLE_VIEW
    -> ;
+-----+-----+
| NAME   | AGE |
+-----+-----+
| Anny   | 22  |
| Calvin | 23  |
| Nick   | 24  |
| Rick   | 24  |
+-----+-----+
4 rows in set (0.00 sec)

```

我们也可以使用 DESC 命令查看视图的结构。

```

mysql> DESC PEOPLE_VIEW;
+-----+-----+-----+-----+-----+-----+
| Field | Type   | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| ID    | int(11)| NO   |     | 0        |       |
+-----+-----+-----+-----+-----+
1 row in set (0.01 sec)

```

4.2 替换视图

创建或替换原有视图。

```

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

```

创建或替换后查看视图。

```

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

```

6小时前



实验楼管理员 (<https://www.shiyanlou.com/user/8490>) 💎 (<https://www.shiyanlou.com/vip>)

(<https://www.shiyanlou.com/user/8490>)

4.3 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。

```

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

```

插入数据成功后查看视图。

```
mysql> SELECT * FROM PEOPLE_VIEW ;
+-----+-----+-----+
| PEOPLE_ID | PEOPLE_NAME | PEOPLE_AGE |
+-----+-----+-----+
|          1 | Anny        |          22 |
|          2 | Calvin      |          23 |
|          4 | Nick        |          24 |
|          5 | Rick        |          24 |
|          6 | Kerry       |          33 |
+-----+-----+-----+
5 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。

```
mysql> SELECT * FROM PEOPLE;
+----+-----+-----+-----+
| ID | NAME  | AGE | BIRTHDAY          |
+----+-----+-----+-----+
|  1 | Anny  |  22 | 1992-05-22 00:00:00 |
|  2 | Calvin |  23 | 1991-05-22 00:00:00 |
|  4 | Nick  |  24 | 1990-05-22 00:00:00 |
|  5 | Rick  |  24 | 1991-05-22 00:00:00 |
|  6 | Kerry |  33 | NULL               |
+----+-----+-----+-----+
5 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。

4.4 删除视图

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

博文作者：GarveyCalvin

博文出处：<http://www.cnblogs.com/GarveyCalvin/>

6小时前

登录后才能回答问题哟~

我要提问

标签

Linux (<https://www.shiyanlou.com/questions/?tag=Linux>) Python (<https://www.shiyanlou.com/questions/?tag=Python>)

C/C++ (<https://www.shiyanlou.com/questions/?tag=C/C++>) 实验环境 (<https://www.shiyanlou.com/questions/?tag=实验环境>)

技术分享 (<https://www.shiyanlou.com/questions/?tag=技术分享>) 功能建议 (<https://www.shiyanlou.com/questions/?tag=功能建议>)

课程需求 (<https://www.shiyanlou.com/questions/?tag=课程需求>) Java (<https://www.shiyanlou.com/questions/?tag=Java>)

其他 (<https://www.shiyanlou.com/questions/?tag=其他>) SQL (<https://www.shiyanlou.com/questions/?tag=SQL>)

Hadoop (<https://www.shiyanlou.com/questions/?tag=Hadoop>) NodeJS (<https://www.shiyanlou.com/questions/?tag=NodeJS>)

常见问题 (<https://www.shiyanlou.com/questions/?tag=常见问题>) Web (<https://www.shiyanlou.com/questions/?tag=Web>)

Shell (<https://www.shiyanlou.com/questions/?tag=Shell>) PHP (<https://www.shiyanlou.com/questions/?tag=PHP>)

Git (<https://www.shiyanlou.com/questions/?tag=Git>) HTML (<https://www.shiyanlou.com/questions/?tag=HTML>)

HTML5 (<https://www.shiyanlou.com/questions/?tag=HTML5>) 信息安全 (<https://www.shiyanlou.com/questions/?tag=信息安全>)

网络 (<https://www.shiyanlou.com/questions/?tag=网络>) GO (<https://www.shiyanlou.com/questions/?tag=GO>)

NoSQL (<https://www.shiyanlou.com/questions/?tag=NoSQL>) 训练营 (<https://www.shiyanlou.com/questions/?tag=训练营>)

Android (<https://www.shiyanlou.com/questions/?tag=Android>) Ruby (<https://www.shiyanlou.com/questions/?tag=Ruby>)

Perl (<https://www.shiyanlou.com/questions/?tag=Perl>)

相关问题

C++静态库与动态库 (<https://www.shiyanlou.com/questions/3017>)

谈Runtime机制和使用的整体化梳理 (<https://www.shiyanlou.com/questions/3010>)

JavaScript：彻底理解同步、异步和事件循环(Event Loop) (<https://www.shiyanlou.com/questions/3009>)

Github上的十大深度学习项目 (<https://www.shiyanlou.com/questions/3000>)

git基础知识整理 (<https://www.shiyanlou.com/questions/2999>)

动手做实验，轻松学IT。

实验楼-通过动手实践的方式学会IT技术。

公司简介 (<https://www.shiyanlou.com/aboutus>) 联系我们 (<https://www.shiyanlou.com/contact>) 常见问题 (<https://www.shiyanlou.com/faq#howtostart>)
我要开课 (<https://www.shiyanlou.com/labs>) 隐私协议 (<https://www.shiyanlou.com/privacy>) 会员条款 (<https://www.shiyanlou.com/terms>)
友情链接 (<https://www.shiyanlou.com/friends>)
站长统计 (http://www.cnzz.com/stat/website.php?web_id=5902315) 蜀ICP备13019762号 (<http://www.miibeian.gov.cn/>)



QQ群



微信



微博
(<http://weibo.com/shiyanlou2013>)