

下载 : mysql-5.7.17.tar

重新克隆新的虚拟机:

eth0 网卡:192.168.4.50-192.168.4.57

主机名称:mysql50-mysql57

## 案例一：安装部署 MySQL

### 1. 准备工作（非必须的操作）:

关闭防火墙（如果有的话）

关闭 SELinux（如果有的话）

如果之前有 mariadb，则需要先卸载，并删除对应的配置与数据:

```
systemctl stop mariadb
rm -rf /etc/my.cnf
rm -rf /var/lib/mysql/*
rpm -e --nodeps mariadb mariadb-server mariadb-devel
```

### 2. 安装部署 MySQL

```
[root@mysql50 ~]# tar -xf mysql-5.7.17.tar
```

```
[root@mysql50 ~]# yum -y install mysql-community*
```

### 3. 启动服务

```
[root@mysql50 ~]# systemctl start mysqld
```

提示：第一次启动，需要初始化数据，会比较慢

```
[root@mysql50 ~]# systemctl status mysqld
```

```
[root@mysql50 ~]# systemctl enable mysqld
```

### 4. 配置 MySQL 管理员密码（默认数据库管理员账户为 root）

第一次启动时，mysql 会自动为 root 账户配置随机密码，我们需要通过日志查看该密码

```
[root@mysql50 ~]# grep password /var/log/mysqld.log
```

```
2018-12-25T12:43:41.164573Z 1 [Note] A temporary password is generated for root@localhost: cvAd3af8a<j?
```

```
[root@mysql50 ~]# mysql -uroot -p'cvAd3af8a<j?'
```

```
mysql> show databases;
```

```
ERROR 1820 (HY000): You must reset your password using ALTER USER statement before executing this statement.
```

注意：用该密码登录到服务端后，必须马上修改密码，不然会报上面的错误。

密码策略

策略参数	值	描述
validate_password_policy	0 或者 LOW	长度
	1 或者 MEDIUM（默认）	长度；数字、大写/小写，特殊符号
	2 或者 STRONG	长度；数字、大写/小写，特殊符号；字典文件

```
mysql> set global validate_password_policy=0; //只验证长度
```

```
Query OK, 0 rows affected (0.00 sec)
```

```
mysql> set global validate_password_length=6; //修改密码长度,默认值是 8 个字符
```

```
Query OK, 0 rows affected (0.00 sec)
```

```
mysql> alter user user() identified by "123456"; //修改登陆密码
```

```
Query OK, 0 rows affected (0.00 sec)
```

修改密码后，可以数据库命令了！

```
mysql> show databases;
```

## 案例二：数据库基本管理

### 1. 数据库操作基本流程

- 连接登陆数据库
- 创建数据库
- 创建数据表
- 插入数据记录
- 断开连接

连接 MySQL 数据库的命令语法格式：

```
[root@mysql50 ~]# mysql [ -h 服务器 IP 或域名 -u 用户名 -p 密码 数据库名称 ]
quit 或者 exit 退出数据库
```

```
[root@mysql50 ~]# mysql -h 127.0.0.1 -uroot -p123456
```

注意事项：

操作指令不区分大小写（密码和变量除外）

每条 SQL 语句都以分号;结束

默认不支持 Tab 键补齐（可以自行下载安装 <https://github.com/dbcli/mycli> 工具实现自动补齐功能）

\c 可以取消书写错误的命令

常用的 SQL 命令分类：

DDL 数据定义语言（create, alter, drop）

DML 数据操作语言（insert, update, delete）

DCL 数据控制语言（grant, revoke）

DTL 数据事务语言（commit, rollback, savepoint）

### 2. 数据库相关指令练习

```
mysql> show databases; #查看数据库
mysql> use mysql; #切换数据库
mysql> select database(); #查看当前数据库
+-----+
| database() |
+-----+
| mysql |
+-----+
1 row in set (0.00 sec);
mysql> create database tts character set utf8; #创建数据库
mysql> drop database tts; #删除数据库
```

提示：数据库命名规则

（数字、字母、下划线，不能纯数字；区分大小写；不能使用关键词或特殊符号）

### 3.数据表相关指令练习

创建数据表基本语法格式如下:

```
create table 数据库名称.数据表名称(
    字段名 1 数据类型(宽度) 约束条件,
    字段名 2 数据类型(宽度) 约束条件,
    ....
);
```

创建入下图所示的数据表, 数据库名称为 school, 数据表名称为 student。

学号	姓名	性别	手机号	通信地址
NSD131201	张三	男	13012345678	朝阳区劲松南路 ..
NSD131202	韩梅梅	女	13722223333	海淀区北三环西路 ..
NSD131203	王五	男	18023445678	丰台区兴隆中街 ..

```
mysql> show character set;           #查看所有可用编码
mysql> create database school character set utf8; #创建数据库
mysql> create table school.student(
    学号 char(20),
    姓名 char(20),
    性别 char(5),
    手机号 int(11),
    通信地址 char(50));              #创建数据表
```

查看数据表结构语法格式: desc 数据表名称;

```
mysql> desc school.student;
+-----+-----+-----+-----+-----+-----+
| Field          | Type      | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| 学号           | char(20)  | YES  |     | NULL    |       |
| 姓名           | char(20)  | YES  |     | NULL    |       |
| 性别           | char(5)   | YES  |     | NULL    |       |
| 手机号         | int(11)   | YES  |     | NULL    |       |
| 通信地址       | char(50)  | YES  |     | NULL    |       |
+-----+-----+-----+-----+-----+-----+

```

插入数据的语法格式: insert into 数据库名称.数据表名称 values(值列表);

```
mysql> insert into school.student values('NSD181001','葫芦娃','男',1388888888,'北京');
```

一次插入 1 条数据。

```
mysql> insert into school.student values('NSD181002','蛇精','女',1389999999,'上海'),
    ('NSD181003','爷爷','男',1387777777,'长白山');
```

一次插入多条数据 (多条数据使用逗号分隔)。

查看数据:

```
mysql> select * from school.student;
```

学号	姓名	性别	手机号	通信地址
NSD181001	葫芦娃	男	13888888888	北京
NSD181002	蛇精	女	13899999999	上海
NSD181003	爷爷	男	13877777777	长白山

更新数据语法格式: update 数据库名称.数据表名称 set 字段=值 [where 条件]

```
mysql> update school.student set 性别='女'; #更新所有数据
```

```
mysql> update school.student set 性别='男' where 姓名='葫芦娃'; #更新满足条件的数据
```

删除数据:

```
mysql> delete from school.student where 学号='NSD181003'; #删除表中满足条件的数据
```

```
mysql> delete from school.student; #删除表中所有数据
```

删除数据表:

```
mysql> drop table school.student; #删除整个数据表
```

## 案例三：MySQL 数据类型

### 1. 数字类型

类型	大小	范围(有符号)	范围(无符号)	用途
<b>tinyint</b>	1 字节	-128~127	0~255	微小整数
<b>smallint</b>	2 字节	-32768~32767	0~65535	小整数
<b>mediumint</b>	3 字节	$-2^{23} \sim 2^{23} - 1$	$0 \sim 2^{24} - 1$	中整数
<b>int</b>	4 字节	$-2^{31} \sim 2^{31} - 1$	$0 \sim 2^{32} - 1$	大整数
<b>bigint</b>	8 字节	$-2^{63} \sim 2^{63} - 1$	$0 \sim 2^{64} - 1$	极大整数
<b>float</b>	4 字节			单精度浮点数(小数点)
<b>double</b>	8 字节			双精度浮点数(小数点)
<b>decimal</b>	Decimal(M,D), 其中 M 为有效位数, D 为小数位数, M 应大于 D, 占用 M+2 字节			
<b>unsigned</b>	标记使用无符号存储			

```
mysql> create table school.num(
  id tinyint,
  age int(3),
  score float(4,2));
```

Query OK, 0 rows affected (0.16 sec)

```
mysql> desc school.num;
```

```
+-----+-----+-----+-----+-----+
| Field | Type          | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+
| id    | tinyint(4)   | YES  |     | NULL    |       |
| age   | int(3)       | YES  |     | NULL    |       |
| score | float(4,2)   | YES  |     | NULL    |       |
+-----+-----+-----+-----+-----+
```

3 rows in set (0.00 sec)

```
mysql> insert into school.num values(1111,22,11.2);
```

ERROR 1264 (22003): Out of range value for column 'id' at row 1

#提示值超出范围 (tinyint 只能存-128~127 或者 0~255 之间的值)。

```
mysql> insert into school.num values(130,22,11.2);
```

ERROR 1264 (22003): Out of range value for column 'id' at row 1

#130 也提示错误, 因为默认使用的是有符号的存储, 如果需要无符号需要添加 unsigned 标记。

```
mysql> insert into school.num values(-125,22,11.2);
```

```
Query OK, 1 row affected (0.06 sec) #正确
```

```
mysql> insert into school.num values(-125,22,143.434);
```

```
ERROR 1264 (22003): Out of range value for column 'score' at row 1
```

提示：错误，小数的总长度为 4 位，也就是整数为 2 位，小数为 2 位。当整数写 3 位就报错。

```
mysql> insert into school.num values(-125,22,14.43433333);
```

```
ERROR 1264 (22003): Out of range value for column 'score' at row 1
```

提示：不报错误，整数位合法，小数位超出，系统会自动把多余的删除，进行四舍五入。

## 2. 字符类型

类型	描述
char(字符数)	固定长度，最大长度 255 字符，不够指定的字符数时自动在右边填补空格，超出指定字符数则无法写入。
varchar(字符数)	可变长度，根据实际数据大小分配存储空间，超出指定字符数则无法写入。
text/blob	字符数大于 65535 时使用

```
mysql> create table school.info(
```

```
name char(4),
```

```
email varchar(30));
```

```
mysql> desc school.info;
```

```
mysql> insert into school.info values('tom', 'tom@163.com');
```

```
Query OK, 1 row affected (0.06 sec) #注意：字符串需要使用引号！
```

```
mysql> select * from school.info;
```

## 3. 字符类型

datetime 日期时间类型，占 8 个字节

范围 1000-01-01 00:00:00:000000~9999-12-31 23:59:59.999999

**如果不给该类型的数据赋值，则默认为 NULL**

timestamp 日期时间类型，占 4 个字节

范围 1970-01-01 00:00:00:000000~2038-01-19 03:14:07.999999

**如果不给该类型的数据赋值，则 mysql 自动为其分配当前的系统时间**

YYYYmmddhhmmss

date 日期类型，占用 4 个字节

范围 0001-01-01~9999-12-31

**默认使用 4 位数字表示，当只用 2 位数字负值时：**

**01~69 自动识别为 2001~2069**

## 70~99 自动识别为 1970~1999

year 年份类型，占用 1 个字节  
范围 1901-2155

time 时间类型，占用 3 个字节  
范围 HH:MM:SS

创建学员信息表：姓名、出生日期、入学年份、上课时间、下课时间

```
mysql> create table school.stuinfo(  
    name char(5),  
    birth datetime,  
    start year,  
    begin time,  
    end time);
```

```
mysql> insert into school.stuinfo values ('tom',20011010121200,2018,0800,1800);
```

```
mysql> select * from school.stuinfo;
```

```
+-----+-----+-----+-----+-----+  
| name | birth           | start | begin  | end    |  
+-----+-----+-----+-----+-----+  
| tom  | 2001-10-10 12:12:00 | 2018 | 00:08:00 | 00:18:00 |  
+-----+-----+-----+-----+-----+
```

```
mysql> insert into school.stuinfo values ( 'lucy',20011010121200,2555,0800,1800);
```

```
ERROR 1264 (22003): Out of range value for column 'start' at row 1 #year 年份的有效范围是 1901-2155
```

几个 mysql 内置的时间函数

```
mysql> select now(),sysdate();
```

#### 4. 字符类型

枚举类型（选择类型）

enum(值 1, 值 2, 值 3...) #单选项

set(值 1, 值 2, 值 3...) #多选项

```
mysql> create table school.tea(  
    name char(5),  
    gender enum('boy', 'girl'),  
    interest set('book','film','music','football'),
```

```
);  
mysql> desc school.tea;  
  
mysql> insert into school.tea values ('tom', 'man', 'it,boot')  
错误，超出了可选择的范围  
  
mysql> insert into school.tea values ('tom', 'man', 'it,boot')
```