# SQL概述

结构化查询语言(Structured Query Language)简称SQL，是一种特殊目的的编程语言，是一种数据库查询和程序设计语言，用于存取数据以及查询、更新和管理关系数据库系统；同时也是数据库脚本文件的扩展名。

从上可以看出我们数据库相关工作职位大概两种：DBD和DBA

dba是数据库管理员database administrator  
dbd是数据库开发人员database developer

SQL 是1986年10 月由美国国家标准局（ANSI）通过的数据库语言美国标准，接着，国际标准化组织（ISO）颁布了SQL正式国际标准。1989年4月，ISO提出了具有完整性特征的SQL89标准，1992年11月又公布了SQL92标准，在此标准中，把数据库分为三个级别：基本集、标准集和完全集。

至于什么基本集，标准集我们不用管，看看就行，牵扯到数据库原理和数学算法里面了

# SQL语句结构

结构化查询语言包含6个部分：

**一：数据查询语言（DQL:Data Query Language）：**

其语句，也称为“数据检索[语句](http://baike.baidu.com/view/537751.htm" \t "_blank)”，用以从表中获得数据，确定数据怎样在应用程序给出。保留字SELECT是DQL（也是所有SQL）用得最多的动词，其他DQL常用的保留字有WHERE，ORDER BY，GROUP BY和HAVING。这些DQL保留字常与其他类型的SQL语句一起使用。

**二：数据操作语言（DML：Data Manipulation Language）：**

其语句包括动词INSERT，UPDATE和DELETE。它们分别用于添加，修改和删除表中的行。也称为动作查询语言。

**三：事务处理语言（TPL）：跟shell有点类似 由多条sql语句组成的整体**

它的语句能确保被DML语句影响的表的所有行及时得以更新。TPL语句包括BEGIN TRANSACTION，COMMIT和ROLLBACK。

**四：数据控制语言（DCL）：**

它的语句通过GRANT或REVOKE获得许可，确定单个用户和用户组对数据库对象的访问。某些RDBMS可用GRANT或REVOKE控制对表单个列的访问。

**五：数据定义语言（DDL）：**

其语句包括动词CREATE和DROP。在数据库中创建新表或删除表（CREAT TABLE 或 DROP TABLE）；为表加入索引等。DDL包括许多与人数据库目录中获得数据有关的保留字。它也是动作查询的一部分。

**六：指针控制语言（CCL）：**

它的语句，像DECLARE CURSOR，FETCH INTO和UPDATE WHERE CURRENT用于对一个或多个表单独行的操作。([dɪˈkler] [ˈkɜ:rsə(r)]声明游标)[fɛtʃ] into获取到

# MySQL语句

## **关于数据库的操作**

### **查看数据库：**

mysql> show databases;

+--------------------+

| Database |

+--------------------+

| information\_schema |

| mysql |

| performance\_schema |

| sys |

注：

1：information\_schema这数据库保存了MySQL服务器所有数据库的信息。如数据库名，数据库的表，表栏的数据类型不访问权限等。 [ˈskimə]

2：performance\_schema 这是MySQL5.5新增的一个性能优化的引擎：命名PERFORMANCE\_SCHEMA [pəˈfɔ:məns]

主要用于收集数据库服务器性能参数。MySQL用户是不能创建存储引擎为PERFORMANCE\_SCHEMA的表

<http://www.cnblogs.com/hzhida/archive/2012/08/08/2628833.html>

3：mysql库是系统库，里面保存有账户信息，权限信息等。

4：mysql5.7增加了sys 系统数据库，通过这个库可以快速的了解系统的元数据信息

元数据是关于数据信息的数据，如数据库名或表名，列的数据类型，或访问权限等。

mysql> show databases \G #以行的方式显示

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 1. row \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Database: information\_schema

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 2. row \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Database: mysql

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 3. row \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Database: performance\_schema

[root@server01 ~]# mysql -e 'show databases' -usystem -p123456

+--------------------+

| Database |

+--------------------+

| information\_schema |

| mysql |

| performance\_schema |

+--------------------+

**mysql –e 后面我们接SQL语句，直接终端运行，后面写sql 相关shell可以用到**

[root@xuegod63 ~]# mysqlshow -uroot -p123456

+--------------------+

| Databases |

+--------------------+

| information\_schema |

| mysql |

| performance\_schema |

+--------------------+

### **创建数据库：**

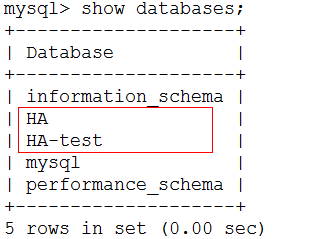
**语法：create database 数据库名;**

**创建数据库注意事项：**

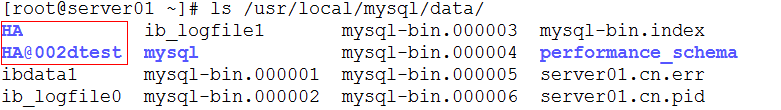
1. **在文件系统中，MySQL的数据存储区将以目录方式表示MySQL数据库。因此，上面命令中的数据库名字必须与操作系统的约束的目录名字一致。例如不允许文件和目录名中有\,/,:,\*,?,”,<,>,|这些符号，在MySQL数据库名字中这些字母会被自动删除。<遵从目录的约束>**
2. **数据库的名字不能超过64个字符，包含特殊字符的名字或者是全部由数字或保留字组成的名字必须用单引号``包起来。**
3. **数据库不能重名。**

mysql> create database HA; #创建一个名为HA的数据库

mysql> create database `HA-test`;



[root@server01 ~]# ls /usr/local/mysql/data/ #查看数据库存放目录



### **选择要操作的数据库：**

使用USE语句将会选择一个数据库成为当前数据库。后面的操作默认都在被选择的数据库中操作。

mysql> use HA-test;

Database changed

**查看自己所处的位置及默认所在的位置**

mysql> select database();

+------------+

| database() |

+------------+

| HA-test |

+------------+

1 row in set (0.00 sec)

默认

mysql> select database();

+------------+

| database() |

+------------+

| NULL |

+------------+

1 row in set (0.00 sec)

Null意味着没有选择数据库

Null在数据库中表示 不知道的数据，主要有3种意思：

1）知道数据存在，但不知道具体值.

2）不知道数据是否存在.

3）数据不存在.

**在命令行选择默认的数据库**

mysql -uroot -p123456 HA

mysql> select now(),user(),database();

+---------------------+------------------+------------+

| now() | user() | database() |

+---------------------+------------------+------------+

| 2015-10-06 10:52:48 | system@localhost | HA |

+---------------------+------------------+------------+

### 删除数据库：

mysql> drop database `HA-test`;

删除没有任何提示，要慎重操作

**方法2：直接到数据库存放目录移出就行**

cd /usr/local/mysql/data/

mv HA@002dtest /tmp

mysql> show databases;

课间时间

**使用****IF** **EXISTS 子句以避免删除不存在的数据库时出现的MySQL错误信息**

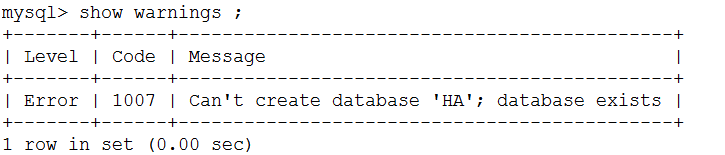


mysql> drop database if exists `HA-test`;

IF EXISTS：如果存在

**同理我们创建数据库时也可以使用**





mysql> create database if not exists HA;

## **关于表的操作：**

### **创建表：**

**语法：create table 表名 (字段名 类型, 字段名 类型, 字段名 类型);**

mysql> create table student(id int(20),name char(40),age int);

### **查看表相关信息：**

查看表：

要进入到数据库再查看

mysql> use mysql;

Database changed

mysql> show tables;

+--------------+

| Tables\_in\_HA |

+--------------+

| student |

**查看表的结构：**

Describe

mysql> desc student;

+-------+----------+------+-----+---------+-------+

| Field | Type | Null | Key | Default | Extra |

+-------+----------+------+-----+---------+-------+

| id | int(20) | YES | | NULL | |

| name | char(40) | YES | | NULL | |

| age | int(11) | YES | | NULL | |

+-------+----------+------+-----+---------+-------+

3 rows in set (0.00 sec)

mysql> explain mysql.user;

mysql> show columns from mysql.user;

mysql> show fields from mysql.user;

mysql> show columns from mysql.user like '%user';

**会一种常用的就行**

**查看创建表执行了哪些命令：**

mysql> show create table student \G

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 1. row \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Table: student

Create Table: CREATE TABLE `student` (

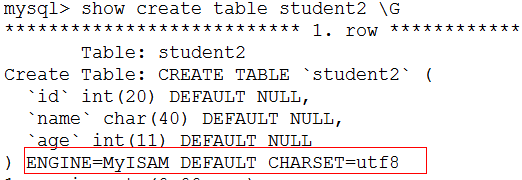
`id` int(20) DEFAULT NULL,

`name` char(40) DEFAULT NULL,

`age` int(11) DEFAULT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8

mysql> create table student2(id int(20),name char(40),age int)ENGINE=MyISAM DEFAULT CHARSET=utf8; 可以指定表的默认存储引擎和字符集



这两个是默认存储引擎和默认字符集

### **删除表：**

mysql> drop table student2;

### **禁止预读表信息：**

**没有禁止前的提示**

**mysql> use performance\_schema;**

**Reading table information for completion of table and column names**

**You can turn off this feature to get a quicker startup with -A**

**登录的时候加上-A参数**

**mysql -uroot –p123456 -A**

### **修改表名称alter：**

**语法：alter table 表名 rename 新表名;**

**mysql>** **alter table student rename students; #studen表名修改为students**

**mysql> show tables;**

**+--------------+**

**| Tables\_in\_HA |**

**+--------------+**

**| students |**

**+--------------+**

### **修改表中的字段类型：**

**语法：alter table 表名 modify 要修改的字段名 要修改的类型;**

**mysql> desc students;**

**+-------+----------+------+-----+---------+-------+**

**| Field | Type | Null | Key | Default | Extra |**

**+-------+----------+------+-----+---------+-------+**

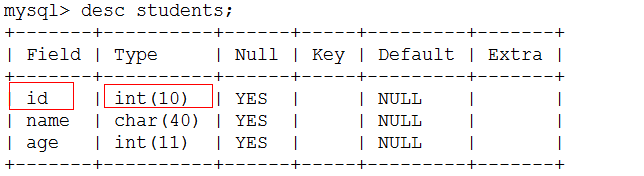
**| id | int(20) | YES | | NULL | |**

**| name | char(40) | YES | | NULL | |**

**| age | int(11) | YES | | NULL | |**

**+-------+----------+------+-----+---------+-------+**

**mysql>** **alter table students modify id int(10);**



### **修改表中的字段类型和字段名称：**

**语法：alter table 表名 change 原字段名 新字段名 新字段类型;**

**查了一下官方文档，发现mysql还真的不支持同时修改多个字段，**

MODIFY [COLUMN] ***col\_name*** ***column\_definition***

[FIRST | AFTER ***col\_name***]

**来源：http://dev.mysql.com/doc/refman/5.5/en/alter-table.html**

**mysql> desc students;**

**+-------+----------+------+-----+---------+-------+**

**| Field | Type | Null | Key | Default | Extra |**

**+-------+----------+------+-----+---------+-------+**

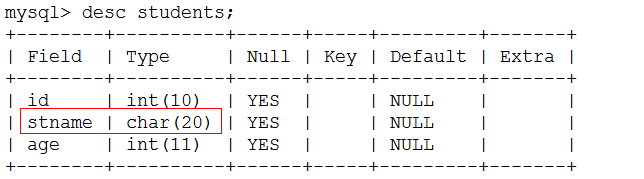
**| id | int(10) | YES | | NULL | |**

**| name | char(40) | YES | | NULL | |**

**| age | int(11) | YES | | NULL | |**

**+-------+----------+------+-----+---------+-------+**

**mysql> alter table students change name stname char(20);**



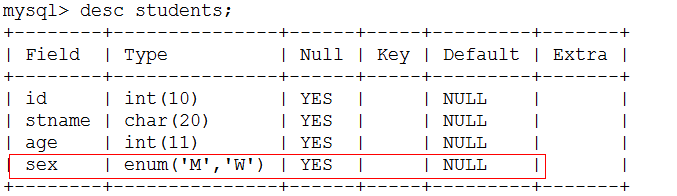
**注：CHANGE 和MODIFY的区别：**

**CHANGE 对列进行重命名和更改列的类型，需给定旧的列名称和新的列名称、当前的类型。 MODIFY 可以改变列的类型，此时不需要重命名（不需给定新的列名称）**

### **在表中添加字段：**

**语法：alter table 表名 add 字段名 字段类型；**

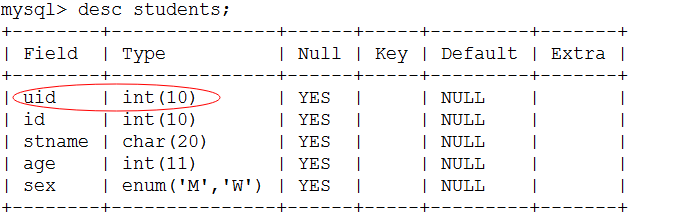
**mysql> alter table students add sex enum('M','W');**



### **指定位置添加字段：**

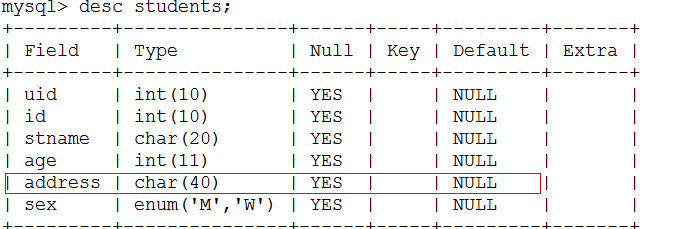
**在第一列添加一个字段：**

**mysql> alter table students add uid int(10) first;**



**在age后面添加一个address字段：**

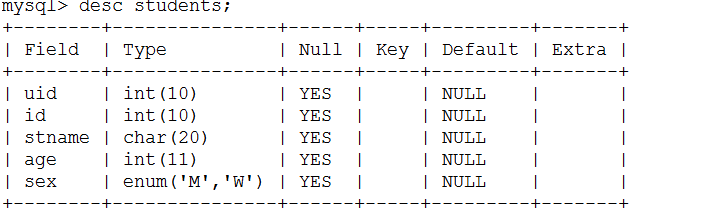
**mysql>** **alter table students add address char(40) after age;**



### **删除表中字段：**

**语法：alter table 表名 drop 字段名 ;**

**mysql> alter table students drop address;**



## **关于记录的操作：**

### **插入字段<记录>insert：**

**语法：insert into 表名values (字段值1,字段值2, 字段值3);**

**mysql> insert into student values(1,'zhangs',21);**

**插入记录时要对应相对的类型**

**mysql> insert into student values(2,'lis',24),(3,'wange',26);**

**同时插入多条，使用，分开**

**mysql> insert into students (id,name)values(4,'hangl');**

**指定字段插入**

### **查询表中记录：**

**语法：select \* from 表名称；**

**mysql> select \* from student; \*表示所有**

**+------+--------+------+**

**| id | name | age |**

**+------+--------+------+**

**| 1 | zhangs | 21 |**

**| 2 | lis | 24 |**

**| 3 | wange | 26 |**

**| 4 | hangl | NULL |**

**+------+--------+------+**

**当字段比较多的时候我们也可以使用\G**

**mysql> select \* from student\G**

**只查询表中某个字段的内容：**

**mysql> select name from student;**

**+--------+**

**| name |**

**+--------+**

**| zhangs |**

**| lis |**

**| wange |**

**| hangl |**

|  |
| --- |
|  |

**mysql> select id,name from student;**

**+------+--------+**

**| id | name |**

**+------+--------+**

**| 1 | zhangs |**

**| 2 | lis |**

**| 3 | wange |**

**| 4 | hangl |**

**+------+--------+**

**查看别的数据库的表或者不在本数据库上进行查看：**

**语法：SELECT 字段 FROM 数据库名.表名;**

mysql> select \*from HA.student; **查看某个数据库下指定的表内容，数据库名.表名**

+------+--------+------+

| id | name | age |

+------+--------+------+

| 1 | zhangs | 21 |

| 2 | lis | 24

|

| 3 | wange | 26 |

| 4 | hangl | NULL |

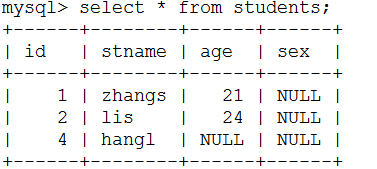
+------+--------+------+

这样等效于先use 数据库，然后查看

### 删除记录：

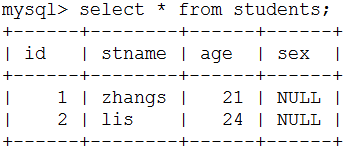
**删除id为3的行**

**mysql> delete from students where id=3;**



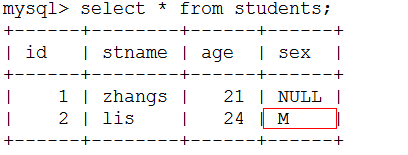
**删除age为空的行**

**mysql> delete from students where age is null;**

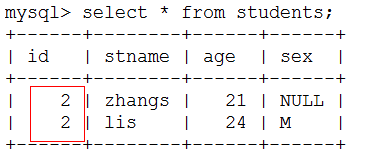


**更新记录**

**mysql> update students set sex='M' where id=2;**



**mysql> update students set id=2; 所有的都变为2**



**update students set stname='zhangsan',age=21 where uid=1;**

**同时更新多个字段时候用，号隔开**

## **SQL基础条件查询语句**

**语法：select 字段名1，字段名2 from 表名 [where 条件];**

### 1；查询students表中的name,age

**mysql> select name,age from students;**

**+--------+------+**

**| name | age |**

**+--------+------+**

**| zhangs | 21 |**

**| lis | 24 |**

**| jk | 24 |**

**| lo | 26 |**

**| io | 25 |**

**| jk | 24 |**

**+--------+------+**

2：去重复查询distinct [dɪˈstɪŋkt]

**mysql> select distinct name,age from students;**

**+--------+------+**

**| name | age |**

**+--------+------+**

**| zhangs | 21 |**

**| lis | 24 |**

**| jk | 24 |**

**| lo | 26 |**

**| io | 25 |**

**mysql> select distinct id,name,age from students where id=3;**

**+------+------+------+**

**| id | name | age |**

**+------+------+------+**

**| 3 | jk | 24 |**

**select distinct \* from students; mysql的distinct可以对\*使用**

### 3：使用and和or进行多条件查询

**or和and 同时存在时，先算and的两边值,逻辑与先执行**

**mysql> select id,name,age from students where id>3 and age>25;**

**+------+------+------+**

**| id | name | age |**

**+------+------+------+**

**| 5 | lo | 26 |**

**+------+------+------+**

**mysql> select id,name,age from students where id>3 or age>25;**

**+------+------+------+**

**| id | name | age |**

**+------+------+------+**

**| 5 | lo | 26 |**

**| 6 | io | 25 |**

**+------+------+------+**

**select \* from students where stname='zhangsan' and (age=21 or age=24);**

**注意and和or都是用的时候的逻辑关系**

### 4：MySQL区分大小写查询：

**Mysql查询默认是不区分大小写的**

**mysql> select name from students where name='jk';**

**+------+**

**| name |**

**+------+**

**| jk |**

**| jk |**

**| JK |**

**+------+**

**解决**

**mysql> select \* from students where binary name='jk'; [ˈbaɪnəri]**

**+------+------+------+------+**

**| id | name | age | sex |**

**+------+------+------+------+**

**| 3 | jk | 24 | W |**

**| 3 | jk | 24 | W |**

**+------+------+------+------+**

**BINARY是类型转换运算符，它用来强制它后面的字符串为一个二进制字符串，可以理解为在字符串比较的时候区分大小写。**

### 5：MySQL查询排序：

**语法：select distinct 字段1，字段2 from 表名order by 字段名;**

**默认为升序 asc**

**mysql> select distinct id from students order by id asc;**

**+------+**

**| id |**

**+------+**

**| 2 |**

**| 3 |**

**| 5 |**

**| 6 |**

**| 8 |**

**+------+**

**mysql> select distinct id from students order by id desc;**

**+------+**

**| id |**

**+------+**

**| 8 |**

**| 6 |**

**| 5 |**

**| 3 |**

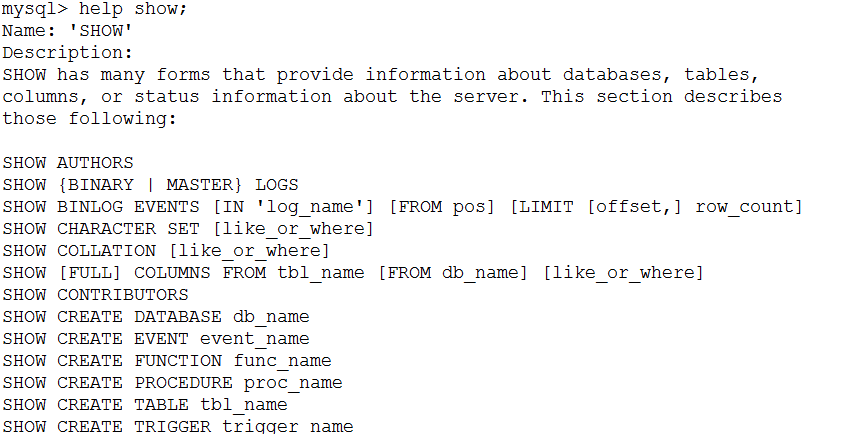
**| 2 |**

**+------+**

## 关于MySQL命令帮助

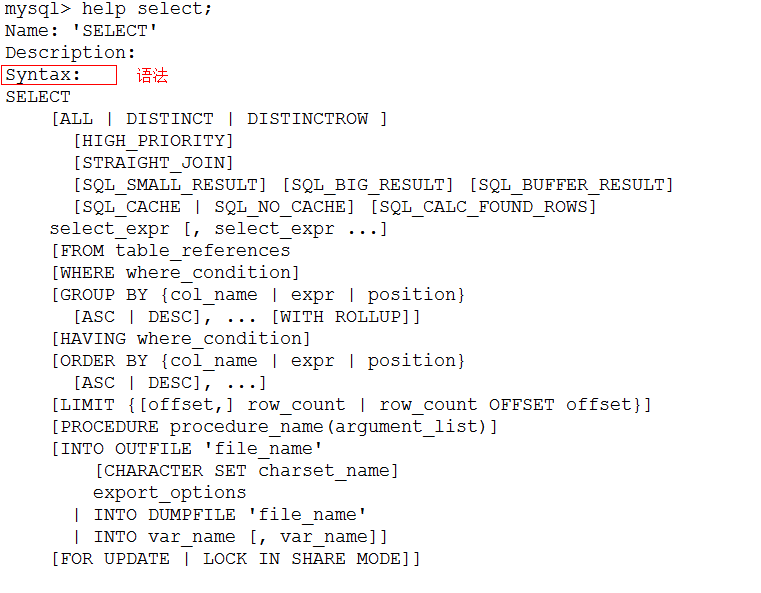
**help**

**mysql> help show;**



**会告诉我们很多使用方法和信息**

**mysql> help select;**



**Yum 安装mysql5.7的方法：**

**CentOS 7版本下载rpm -Uvh** [**https://repo.mysql.com//yum/mysql-5.7-community/el/7/x86\_64/mysql57-community-release-el7-10.noarch.rpm**](https://repo.mysql.com//yum/mysql-5.7-community/el/7/x86_64/mysql57-community-release-el7-10.noarch.rpm)

**Yum list**

**yum -y install mysql-community-server**

**第一次通过# grep "password" /var/log/mysqld.log 命令获取MySQL的临时密码**

**用该密码登录到服务端后，必须马上修改密码，不然操作查询时报错误**

**刚开始设置的密码必须符合长度，且必须含有数字，小写或大写字母，特殊字符。**

**如果想设置简单密码，如下操作：**

**方法一：首先，修改validate\_password\_policy参数的值**

**mysql> set global validate\_password\_policy=0; #定义复杂度**

**mysql> set global validate\_password\_length=1; #定义长度 默认是8**

**mysql>set password for 'root'@'localhost'=password('123456');**

**mysql> flush privileges;**

**方法二：在/etc/my.cnf 可关闭密码强度审计插件，重启mysql服务**

**validate-password=ON/OFF/FORCE/FORCE\_PLUS\_PERMANENT: 决定是否使用该插件(及强制/永久强制使用)。**