目录

1. 数据导入/导出：

1.1 检索目录

1.1.1 查看检索目录

安装 MySQL 服务软件时，会自动创建检索目录

```
mysql> show variables like '%file%';          # like模糊查询，文件的检索目录
...
| secure_file_priv                  | /var/lib/mysql-files/          |
...
mysql> system ls /var/lib/mysql-files                  #可以在mysql下执行linux命令
mysql> exit
```

1.1.2 修改检索目录

```
[root@host50 ~]# vim /etc/my.cnf
[mysqld]
secure_file_priv="/myload"
创建目录并修改所有者，启动服务
[root@host50 ~]# mkdir /myload
[root@host50 ~]# chown mysql /myload
[root@host50 ~]# systemctl restart mysqld
连接到数据库查看
[root@host50 ~]# mysql -uroot -p’ 123qqq...A’
mysql> show variables like 'secure_file_priv';
```

1.2 数据导入

--把系统文件的内容存储到数据库的表里，默认只有数据库管理员 root 用户有数据导入权限

数据导入步骤：

1）建表

2）拷贝文件到检索目录下

3）导入数据

语法格式：

mysql> load data infile "/目录名/文件名" into table 库名.表名 fields terminated by "分隔符" lines terminated by "\n";

将 linux 的系统用户的信息存储到 test 库下的 user 表中

```
mysql> create database test;
mysql> use test;
mysql> show tables;
mysql> create table user(name char(50), password char(1), uid int, gid int, comment varchar(150), homedir char(100), shell  char(50));
mysql> desc user;
将passwd 拷贝到数据库的检索目录/myload，用于导入数据
mysql> system cp /etc/passwd /myload
mysql> system ls /myload
导入数据
mysql> load data infile "/myload/passwd"  into table user fields terminated by ":" lines terminated by "\n";
mysql> select * from user;
为了方便管理，在user表的最前面设置行号字段id(主键和自增长)
mysql> alter table user add id int primary key auto_increment first;
mysql> select * from user;
```

数据导入注意事项

1） 字段分隔符要与文件一致

2) 表字段类型和字段个数要与文件内容匹配

3) 导入数据时指定文件的绝对路径

1.3 数据导出

语法格式：

格式 1：select 命令 into outfile "/目录名/文件名";

格式 2：select 命令 into outfile "/目录名/文件名" fields terminated by "分隔符";

格式 3：select 命令 into outfile "/目录名/文件名" fields terminated by "分隔符" lines terminated by "\n";

```
从user表中，查询出name,homedir,shell字段的前3行记录
mysql> select name,homedir,shell from user limit 3;
导出查询出的数据到user.txt文件中，不指定分隔符时，默认以一个tab为分隔符
mysql> select name,homedir,shell from user limit 3 into outfile "/myload/user.txt";
mysql> system cat /myload/user.txt
导出查询出的数据到user2.txt文件中，各字段之间以#分隔
mysql> select name,homedir,shell from user limit 3 into outfile "/myload/user2.txt" \
fields terminated by "###";                    \为换行符
mysql> system cat /myload/user2.txt
导出查询出的数据到user3.txt文件中，指定行的间隔符为！默认以"\n"换行符分隔
mysql> select name,homedir,shell from user limit 3 into outfile "/myload/user3.txt"  \
fields terminated by "###" lines terminated by "!!!";      \为换行符
mysql> system cat /myload/user3.txt
```

数据导出注意事项：

1、导出数据行数由 SQL 查询绝定

2、导出的是表记录，不包含字段名

3、自动创建存储数据的文件

4、存储数据文件，具有唯一性

2. 管理表记录

2.1 插入表记录

语法格式：

格式 1：添加 1 条记录，给所有字段赋值

insert into 表名 values(字段值列表)

格式 2：添加多条记录，给所有字段赋值

insert into 表名 values(字段值列表)，（字段值列表），（字段值列表）

格式 3：添加 1 条记录，给指定字段赋值

insert into 表名（字段名列表）values(字段值列表)

格式 4：添加多条记录，给指定字段赋值

insert into 表名（字段名列表）values(字段值列表)，(字段值列表)，(字段值列表)

```
mysql> insert into user values(30,'bob','x',3001,3001,'test  \
user','/home/bob','/sbin/nologin');              \为换行符
mysql> select * from user;
mysql> insert into user values(31,'bob','x',3001,3001,'test   \
user','/home/bob','/sbin/nologin'),(41,'tom','x',3002,3002,'student','/home/tom','/sbin/
nologin');                                       \为换行符
mysql> select * from user;
mysql> insert into user(name,homedir) values('alice','/home/alice');
mysql> select * from user;
```

```
mysql> insert into user(name) values('alice'),('jack'),('toma');
mysql> select * from user;
```

注意事项：

1）字段值要与字段类型相匹配

2）字符类型的字段，要用""括起来

3）依次给所有字段赋值时，字段名可以省略

4）只给部分字段赋值时，必须明确写出对应的字段名称

5）没有赋值的字段使用默认值或自增长赋值

6）新纪录追加在末尾

2.2 查询表记录

语法格式：

格式 1：查看所有记录

```
select  字段 1,字段 1,....,字段 N  from  库名.表名
```

格式 2：条件查询

```
select  字段 1,字段 1,....,字段 N  from  库名.表名  where  条件表达式
```

```
查看user表中，所有字段下的所有记录
mysql> select * from user;
查看user表中，id和name字段下的所有记录
mysql> select id,name from test.user;
查看user表中，id小于等于2的 id和name字段下的所有记录
mysql> select id,name from test.user where id<=2;
查看user表中，id小于4的 id和name字段下的所有记录
mysql> select id,name from test.user where id<4;
```

注意事项：

1）* 代表所有字段

2）查看当前库表记录时库名可以省略

3）字段列表决定显示列个数

4）条件决定显示行个数

2.3 更新表记录

语法格式：

格式 1：批量更新

update 库名.表名 set 字段名=值,字段名=值,字段名=值.....

格式 2：条件匹配更新

update 库名.表名 set 字段名=值,字段名=值,字段名=值..... where 条件表达式

```
修改user表中password字段所有记录修改为A, comment字段所有记录修改为 student
mysql> update user set password='A',comment='student';
将user表中name字段的值为"root"的password字段的值改为“x”,comment字段的值改为"root"
mysql> update user set password='x',comment='root' where name='root';
```

注意事项：

1）字段值要与字段类型相匹配

2）对于字符类型的字段，值要用双引号括起来

3）若不使用 where 限定条件，会更新所有记录

4）限定条件时，只更新匹配条件的记录

2.4 删除表记录

语法格式：

格式 1：条件匹配删除

delete from 库名.表名 where 条件表达式

格式 2：删除所有记录**（慎用）**

delete from 库名.表名

删除user表中，id号大于20的表记录

mysql> delete from user where id > 20;

3. 基本匹配条件

3.1 数值比较

--字段必须是数值类型

类 型	比较	例子
=	相等	id = 3
>	大于	uid > 3
>=	大于或等于	uid >= 3
<	小于	uid < 3
<=	小于或等于	uid <= 3
!=	不相等	uid != 3

查询user表中，uid小于10的name和uid字段

mysql> select name,uid from user where uid < 10;

当uid字段的值和gid字段的值相等时，显示user表中name,uid,gid字段的值

```
mysql> select name,uid,gid from user where uid=gid;
查询user表中，id字段的值等于1的，所有记录
mysql> select * from user where id=1;
查询user表中，uid字段的值不等于0的，所有记录
mysql> select * from user where uid!=0;
```

3.2 字符比较

--字段必须是字符类型

类 型	比较	例子
=	相等	name = “root”
!=	不相等	name != “root”
is null	空	shell is null
is not null	非空	shell is not null

```
查询user表中，shell字段的值等于"/sbin/nologin"的name字段
mysql> select name from user where shell='/sbin/nologin';
查询user表中，shell字段的值不是"/sbin/nologin"的name和shell字段
mysql> select name,shell from user where shell!='/sbin/nologin';
查询user表中，name字段的值等于"root"的name和shell字段
mysql> select name,shell from user where name='root';
查询显示user表中，name字段的值等于"root"的id和name字段
mysql> select name,id from user where name='root';
#null和NULL 指空值，没有数据
#"null"    引号，指字符串
#""        指空字符，没有数据，只是占位
mysql> insert into user(name) values(null),("null"),(""), (NULL);
查询user表中，name字段的所有记录信息
```

```
mysql> select name from user;
查询显示user表中name字段的值为null的，id和name字段
mysql> select id,name from user where name is null;
查询显示user表中name字段的值不为null的id和name字段
mysql> select id,name from user where name is not  null;
查询显示，user表中name字段的值为""(空字符)的，id和name字段的记录
mysql> select id,name from user where name="";
```