数据的管理：增删改查

1、数据的插入

准备工作：

创建一个数据库 myseconddb:
create database myseconddb;
指定在 myseconddb 数据库下做操作：
use myseconddb;

建表：

```
create table student(
    sid int primary key auto_increment, #自动增长
    name varchar(20) not null,
    grade int,
    gender varchar(5)
);
```

注解：

auto_increment 是约束的一种，表示自动增长——如果你在表中插入行（记录）的时候，没有给该字段赋值，那么，MySQL 会自动的产生一个唯一的标识符。每次增加一个,默认为 1

为 student 表插入数据：

语法格式 1：一次插入一条数据

insert into 表名称(字段 1,字段 2,……字段 n) values(值 1,值 2,……值 n);

举例：

```
insert into student(name,gender,grade) values('zhangsan','man',90);
#因为建表时,sid 设置了自动增长,所以不需要赋值也会给予一个相应的值
```

```
insert into student(name,gender,grade) values('lisi','man',88);

select * from student;

desc student;
```

注解：

1）字段名称和字段的值要一一对应，不能张冠李戴====>要按照顺序来；
2）sid 的值设置了 auto_increment，可以自动生成及增长，我们不用专门插入；
3）如果插入的数据有中文，插入或数据查询时可能会乱码，这是因为建库的时候
我们使用的默认字符集可能没有支持中文；
4）字符串类型赋值需加引号。

语法格式 2：一次插入多条数据

```
insert into  表名称(字段 1,字段 2,……字段 n) values(值 1,值 2,……值 n),
(值 1,值 2,……值 n),
……
(值 1,值 2,……值 n);
```

举例：

```
insert into student(name,grade,gender)
values('wangwu',77,'f'),
('xiaohong',67,'m'),
('xiaoming',55,'m');
```

需求：

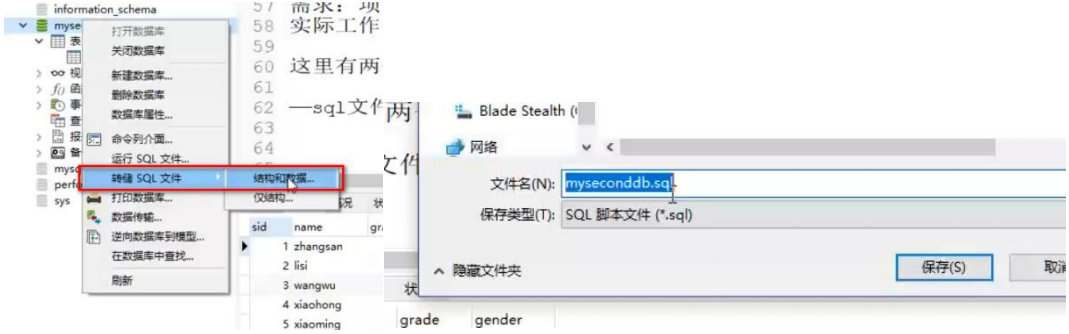
项目的数据成千上万，测试纯手工的方式做数据的插入将耗费大量的人力时间开销，
实际工作中，通常使用 sql 文件的方式一下子导入数据。

这里有两个方面：sql 文件的生成和 sql 文件的导入

——sql 文件的生成：数据的导出。步骤如下:

双击想要导出数据的数据库将其激活——》在该数据库上鼠标右击，选择“转储 SQL 文件”
——》选择“结构和数据”，保存在某个目录下，等待出现“……successfully”，关闭弹窗。
——》在保存的目录下，就可以找到该 sql 文件。

1.　　2.



3.

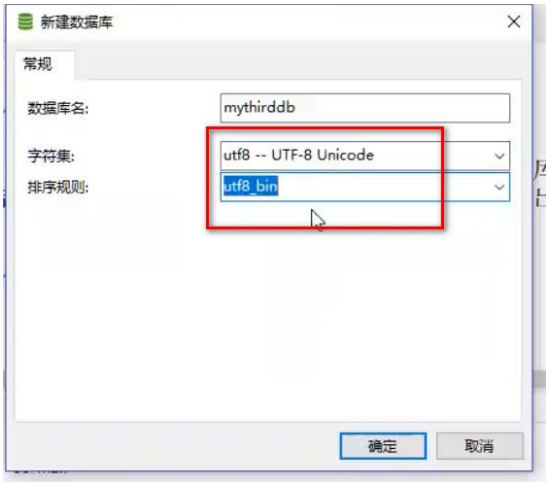
——sql 文件的导入：数据的导入。

在你的连接下建一个数据库，双击激活该数据库——》在该数据库上鼠标右击，选择“运行 SQL 文件”——》选中你要导入的 sql 文件，运行，当出现“…… successfully”，
即表示导入成功。此时刷新数据库下的表，就可以看到导入的表及数据。

新建数据库:

1.

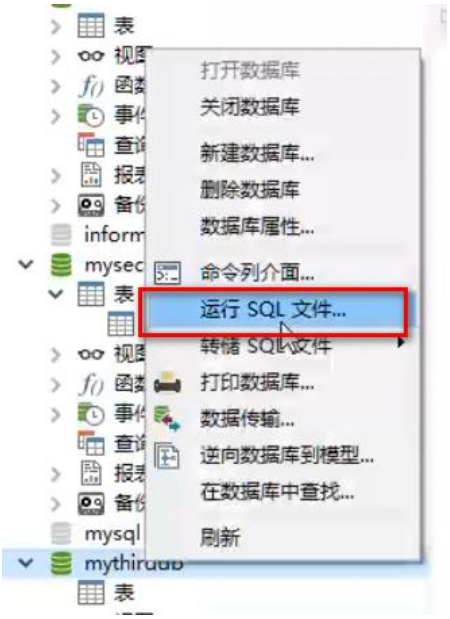
2.支持中文的数据库 utf-8

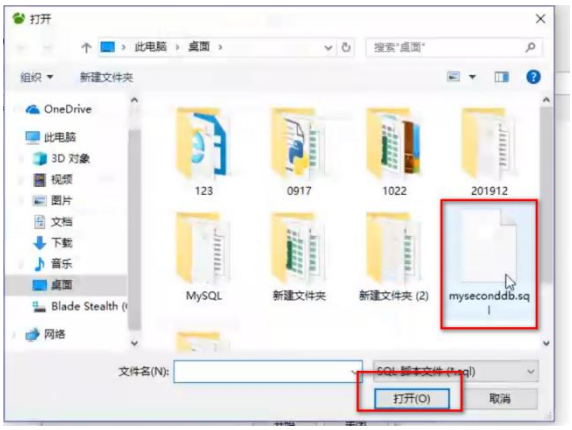


导入数据库文件:

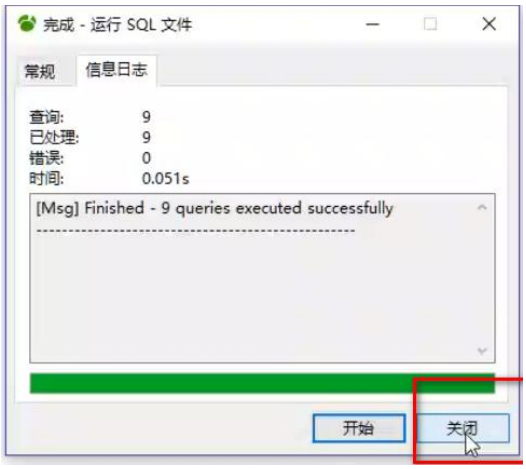
1.

2.





3.



2.数据查询：

准备工作：导入已经准备好的 sql 文件

导入了三张表：
xsb——学生表
xh 学号
xb 性别
xm 姓名
jq 籍贯
nl 年龄
bj 班级
sfzh 身份证号
zcrq 注册日期

kcb——课程表
kch 课程号
kcm 课程名

cjb——成绩表

kch 课程号
xh 学号
cj 成绩

需求：

注解：

成绩表里的 kch 必须参照课程表里的 kch 才有意义，
成绩表里的 xh 必须参照学生表里的 xh 才有意义。

解决办法：

通过外键约束。

达到的效果：

一旦在成绩表里添加了外键约束，那么在往成绩表里插入数据的时候，
它就会去课程表和学生表里去看是否有对应的课程号和学号，
如果没有则拒绝录入，并给出错误提示信息。

数据的查询操作：

Select 后面是查询的字段(列),form 后面是表格名称,where 后面是记录(行)

（1）简单查询

语法：
select *|字段 1,字段 2……字段 n from 表名;
查询数据库的某些表格内的字段信息

示例：
查询所有的学员信息
```
use mythirddb;
select * from xsb;
select xh,xm,xb,jg,nl,bj,sfzh,zcrq from xsb;
```
查询学生的学号和姓名
```
select xh,xm from xsb;
```
查询学生的籍贯信息（重复的籍贯只显示一次）
```
select distinct jg from xsb;
```

注解：
1）从表中选取所有的字段使用*
2）如果想过滤重复的行（记录），使用 distinct

3）select 子句：作用在于过滤出满足条件的列（字段）
4）from 子句：作用在于确定我们要查询的数据来自于哪些表

（2）条件查询

语法：

select *|字段 1,字段 2……字段 n
from 表名
where 查询条件;

注解：where 子句：作用在于过滤出满足条件的行（记录）

示例：

查询张三这个学员的所有信息
select * from xsb where xm='张三';

查询张三的性别和籍贯
select xb,jg from xsb where xm='张三';

查询年龄大于 20 岁的学员信息
select * from xsb where nl>20;

查询年龄小于等于 20 岁的学员信息
select * from xsb where nl<=20;

条件查询还支持多条件：多条件常用的三个逻辑运算符：

与：并且的意思——and
或：或者的意思——or
非：取反的意思——not

举例：

查询张三和李四的学员信息
select * from xsb where xm='张三' or xm='李四';
因为 xm 字段内不可能包含张三和李四同时存在,是要么张三要么李四,所以选择 or
select * from xsb where xm in('张三','李四');
In 是包含不包含的意思,表示张三和李四在不在 xm 这个字段内,如果在就显示

查询年龄在 19 岁到 21 岁区间的学员信息
select * from xsb where nl>=19 and nl<=21;
既要大于等于 19,又要小于等于 21
select * from xsb where nl between 19 and 21;
Between 表示,既然在这个区间内又要包含这个区间的数值;

注意：between and 的用法：between 较小值 and 较大值，
它表示的是一个在较小值到较大值之间的范围，
且包含较小值和较大值(左侧小值,右侧大值)

查询除了张三以外的学员信息
```
select * from xsb where xm!='张三';
select * from xsb where not xm='张三';
```

条件查询还支持模糊查询：使用 like

常用的两个通配符：

%表示此处有 0 个或 1 个或多个字符
_表示此处有 1 个字符

举例：查询姓张的学员的信息
```
select * from xsb where xm like '张%';
select * from xsb where xm like '张_';      #表示的该学员名字只有两个字
```

查询名字里包含“张”的学员信息
```
select * from xsb where xm like '%张%';
```

条件查询还支持空值查询

is null:是空值
is not null:不是空值

举例：查询成绩表里没有成绩的学员学号
```
select xh from cjb where cj is null;
```

（3）排序显示

语法格式：
```
select *|字段 1,字段 2……字段 n
from  表名
where  查询条件
order by  字段  asc|desc;
```

注解：
asc ascend 升序
desc descend 降序

举例：成绩表信息按成绩降序排列
```
select * from cjb order by cj desc;
```

升序：
```
select * from cjb order by cj asc;
select * from cjb order by cj;
```

查询学员信息，按年龄从大到小排序，如果年龄相同，
按学号从小到大排序：

```
select * from xsb order by nl desc,xh asc;
```

009	班长	男	湖南	21	1班	34032219
001	张三	男	北京	20	1班	34032219
007	王钊	男	山西	20	2班	34032219
008	邢华	男	河南	20	3班	34032219
005	郭杨	男	天津	19	3班	34032219

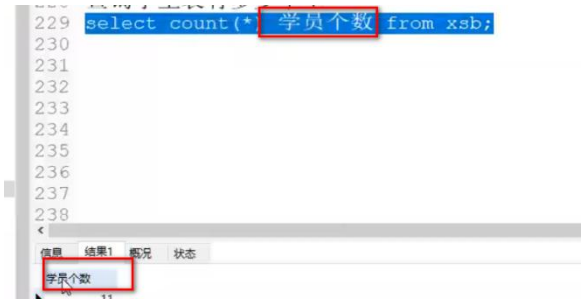
（4）分组查询

聚合（统计）函数——用于做分组统计

count() 统计个数
avg() 求平均值
sum() 求和
max() 求最大值
min() 求最小值

举例：
查询学生表有多少学生

```
select count(*) 学员个数 from xsb;
```



如果 count(*)后面加名称,那么就相当于给这个函数加入叠名,如果不加名称,那么这个就会显示 count(*)

查询‘001’号学员的最大成绩、最小成绩、平均成绩和成绩总分

```
select max(cj),min(cj),avg(cj),sum(cj) from cjb
where xh='001';
```

分组统计语法格式：

```
select *|字段 1,字段 2……字段 n
from 表名
where 查询条件
group by 分组字段
having 过滤条件
order by 字段 asc|desc;
```

注解:
group by 子句：按某个字段做分组
having 子句：对分组后的数据做进一步的过滤
where 子句：对分组前的数据做过滤

举例：
查询每个学员的成绩总分
```
select xh,sum(cj)
from cjb
group by xh;
```

3.数据的更新操作：

语法：
```
update 表名 set 修改的内容 [where 条件];
```

注解：如果不跟条件，那么表里所有的记录都会更新，
如果跟了条件，那么只有满足条件的记录会被更新。

举例：将成绩表里所有人的成绩都减少 2 分
```
update cjb set cj=cj-2;
```

将“001”学员的分数减少 2 分
```
update cjb set cj=cj-2 where xh='001';
```

将张三的性别更新为女，年龄更新为 21：
```
update xsb set xb='女',nl=21 where xm='张三';
```

4.数据的删除：

语法：delete from 表名 [where 条件];

```
[SQL]delete from xsb;
[Err] 1451 - Cannot delete or update a parent row: a foreign key constraint fails (`mythirddb`.`cjb`, CONSTRAINT `cjb_ibfk_2` FOREIGN KEY (`xh`) REFERENCES `xsb` (`xh`))
```

注解：如果不跟条件，那么删除表里的所有数据（保留表结构），
如果跟条件，则只会删除满足条件的记录。

举例：
删除 001 号学员的信息
```
delete from xsb where xh='001';
```

删除所有学员信息
```
delete from xsb;
```

注意：由于 xsb 表里的学号被 cjb 里的学号所参照，那么在做数据删除的时候，
通常是先删除子表（cjb）里的相关记录，再删除附表（xsb:被参照的那张表）里

的
相关记录。
delete from cjb;
delete from xsb;