一. 多表查询(联合查询, 子查询, 联结查询)

```
简单介绍: eg.union/union all(联合查询)
 eg.子查询:select中嵌套select
 eg.连接查询join
```

1. 联合查询

```
#两个表的字段数量必须相等
select name,age,phone from student
union all
select name,age,phone from people;
```

union all:所有数据联合起来

union :所有数据结合在一起, 并去重

```
create table if not exists class(
 id int primary key auto_increment comment '班级号',
 name varchar(10) not null comment '班级名'
)comment '班级表';
```

2. 子查询

```
#查询id为100的学生所学的专业
select class_id from student where id=100;
select *from class where id=(select class_id from student where id=100);
```

```
查询班级为H5班级人数
select count(*)
from student
where class_id=(select id from class where name='H5');
```

```
#查询年龄最大的同学所在班级的所有学生信息
select *from student where class_id in
(select class_id from student where age1=(select max(age1) from
student));

```

```
#查询和张三同班的成年女同学
select *from student where age='女'and age>=18 and class_id in
(select class_id from student where name='张三');
```

```
4
5 in代表多个
```

二. 联结查询

包含: 1. 内联结

2. 外联结

左外联结

右外联结

3. 自联结:内联结的特殊形式

4. 全联结:full join

前提: 笛卡尔积:一次查询多张表,结果的列数=每个表的列数之和

结果的条数数=每个表的条数之乘

```
1 select *from student,class;
2 select *from class,student;
```

当表中有相同的字段,使用这些字段需要添加前缀(表名)

1. 内联结(join)

```
1 select *from class,student where class_id=class.id;
2 select student.*from class,student where class_id=class.id;
3 select class.*from class,student where class_id=class.id;
4 select s.name,c.name from student s ,class c where class_id=c.id;
```

多张组合的临时表联结

```
1 select *from student s inner join class c on s.class_id = c.id;
2
3 inner可以省略
```

查询学生及所在班级的信息

```
1 select *from student s join class c on s.class_id = c.id;
```

查询学生的分班情况

2. 左外连接,左边部分,中间部分

```
select *from student s left join class c on s.class_id = c.id;
select *from student s right join class c on s.class_id = c.id;
select *from class c left join student s on c.id = s.class_id ;
```

3. 全联结:

```
full join
```

4. 自联结

查询和张三性别相同的所有学生

```
select *from student where gender in (select gender from student where name ='张三');

select *from student s1,student s2
where s1.gender=s2.gender and s2.name='张三';
#条件
```

三. DCL

数据控制语言

事物:一组SQL语句,要么都执行,要么都不执行

```
#购物:
#产品表,店铺表,订单表,用户表,地址表

#外键
#not action:没有操作,当主键的值修改,外键不变
#cascade:级联操作,当主键的值修改,外键和主键保持一致;当主键的值删除,对应的外键也删除
#set null:设置为空,当主键的值修改,外键变为null
#set default:设置为默认值,当主键的值修改,外键变为默认
#restrict:和no action一样
 #修改Java班的编号
```

1.开启事物

start transaction ;

```
update class set id=10 where name = 'java';

update student set class_id=10 where class_id=5;
```

2.提交

```
commit ;
```

3.回滚

```
rollback ;
```

三.数据库三大范式

```
1.原子性:数据库的每一列必须是不可拆分的最小单元
2.唯一性:所有的数据必须依赖于主键
3.降低冗余：数据库中每一列不能依赖于其他列
```

四.表设计

1.　　ER图:实体关系图

2.用SQL语句创建四个表并完成相关题目。

3.数据库的表结构

```
# student (学生表)
# 属性名 数据类型 可否为空 含义
# id varchar(20) 否 学号（主键）
# name varchar(20) 否 学生姓名
# gender varchar(20) 否 学生性别
# birthday datetime 可 学生出生年月
# class varchar(20) 可 学生所在班级
#
# course（课程表）
 # 属性名 数据类型 可否为空 含义
 # id varchar (20) 否 课程号（主键）
 # name varchar (20) 否 课程名称
 # teacher_id varchar (20) 否 教工编号（外键）
```

```
#
# score(成绩表)
# 属性名 数据类型 可否为空 含义
# student_id varchar (20) 否 学号（外键）
# course_id varchar (20) 否 课程号（外键）
# degree decimal(4,1) 可 成绩
# 主键：student_id + course_id

# teacher(教师表)
# 属性名 数据类型 可否为空 含 义
# id varchar (20) 否 教工编号（主键）
# name varchar (20) 否 教工姓名
# gender varchar (20) 否 教工性别
# birthday datetime 可 教工出生年月
# title varchar (20) 可 职称
# department varchar (20) 否 教工所在部门
```

```
以下是Navicat可视化工具的安装与激活：
 https://www.jianshu.com/p/5f693b4c9468#comment-20147185
```