MySQL

```sql
-- 查询王燕的信息
select * from xs where xs_name ='王燕';

-- 查询并给重命名
select xs_id as '学号',xs_name as '姓名',xs_zy as '专业',xs_rq as '出生日期' ,xs_xf as '学分',xs_bz as '备注'
 from xs as '学生' where xs_name ='王燕';

-- 模糊查询法  '_'代表一个字符，‘%’代表多个字符
select * from xs where xs_name like '王_';
select * from xs where xs_name like '李%';

-- 查询在某一个范围内between A and B ：查询A到B之间的数据
select * from xs where xs_rq between '1990-01-01' and '1990-12-31';

-- 等价于
select * from xs where xs_rq > '1990-01-01' and xs_rq < '1990-12-31';

-- 指定具体几个范围数值
select * from xs where xs_rq in('1990-02-10','1990-11-20');

-- 不在指定范围内的
select * from xs where xs_rq not in('1990-02-10','1990-11-20');

-- 空值判断 is NULL
select * from xs where xs_bz is null;

-- 非空判定
select * from xs where xs_bz is not null;

-- 去重复  distinct
select distinct * from xs where xs_id < 81108;

-- 对查询结果进行排序  asc:正序     desc:倒序
select * from xs order by xs_rq asc;
select * from xs order by xs_rq desc;

-- 复合排序 先按专业排序，当专业相同时，按日期排序
select * from xs order by xs_zy,xs_rq;

-- 查询前几条记录 limit
-- 查询学号小于81110的前5个学生信息
select * from xs where xs_id <81110 limit 5;

-- 查询学号小于81110的学生信息，列出从下标为2开始的三个学生信息
select * from xs where xs_id <81110 limit 2,3;

-- 分组查询 group by 将查询结果分为男女两个组
select * from xs where xs_id < 81110 group by xs_xb;

-- 查询所有学生中男生和女生年龄最大的两个学生的信息
-- 先将学生分组，获得男生、女生最大的日期
select min(xs_rq) from xs group by xs_xb;
```

```
-- 将上一步的查询结果作为条件，合并
select * from xs where
 xs_rq in (
   select min(xs_rq) from xs group by xs_xb);

-- having 语句的用法 ---where语句里面不能有统计函数，而且where条件不能对group by 进行分组
select * from xs having xs_xf>50;

-- having 一般与group by 结合使用
-- 查询所有学生的平均成绩
select xs_id as '学号',avg(kc_cj)as '平均成绩' from xs_kc group by xs_id;

-- 查询平均成绩大于80的学生信息
-- 1、查询平均成绩大于80的学生id‘
select xs_id from xs_kc group by xs_id having avg(kc_cj)>80;

-- 2、利用查询到的id查询学生的基本信息
select * from xs where
 xs_id in(select xs_id from xs_kc group by xs_id having avg(kc_cj)>80);

-- 多表查询
-- 查询所有学生的课程及成绩
select xs.xs_name,kc_name,xs_kc.kc_cj from xs,xs_kc,kc where xs_kc.kc_id = kc.kc_id and xs_kc.xs_id = xs.xs_id;

-- 多表连接
-- 查询选修了206课程且成绩在80分以上的学生姓名及成绩
select xs.xs_name,xs_kc.kc_cj from xs,xs_kc where xs.xs_id = xs_kc.xs_id and xs_kc.kc_cj>80 and xs_kc.kc_id=206

-- join on
select xs.xs_name,xs_kc.kc_cj from xs join xs_kc on xs.xs_id = xs_kc.xs_id where xs_kc.kc_cj>80 and xs_kc.kc_id=206

-- 查询选修了“计算机基础”课程并且成绩在80分以上的学生学号、姓名、课程名及成绩
select xs.xs_id as '学号',xs.xs_name as '姓名' ,kc.kc_name as '课程名',xs_kc.kc_cj as '成绩'
 from xs join xs_kc on xs.xs_id=xs_kc.xs_id
  join kc on kc.kc_id=xs_kc.kc_id
   where kc.kc_name='计算机基础' and xs_kc.kc_cj>80;

-- 查询数据库中课程不同、成绩相同的学生的学号、课程号及成绩
select a.xs_id as '学号',a.kc_id as '课程号',a.kc_cj as '成绩'
 from xs_kc as a join xs_kc as b on a.kc_cj=b.kc_cj
  and a.xs_id = b.xs_id
   and a.kc_id!=b.kc_id;

-- LEFT OUTER JOIN : 左外连接。。。在结果表里除了匹配行外，还包括左表有的但右表中不匹配的行，对于这样的行，从右表被选择的列设置为NULL
-- 查询所有学生情况及他们选修的课程号，若学生未选修任何课，也要包括其情况
select xs.*,kc_id from xs left outer join xs_kc on xs.xs_id=xs_kc.xs_id;

-- 交叉连接：CROSS JOIN 范例：列出所有学生所有可能的选课情况
select xs.xs_id as '学号',xs_name as '姓名',kc.kc_id as '课程号',kc.kc_name as '课程名'
 from xs cross join kc;

-- 去重复 distinct  范例：查找所有学生选过的课程名和课程号
select distinct xs_kc.kc_id as '课程号',kc.kc_name as '课程名' from xs_kc,kc where kc.kc_id=xs_kc.kc_id;

-- 替换查询结果中的数据
--
```

```
select xs.xs_id as '学号' ,xs.xs_name as '姓名',xs.xs_xf as '学分' ,
 case
 when xs.xs_xf is null then '未选课'
 when xs.xs_xf < 50 then '不及格'
 when xs.xs_xf >= 50 and xs.xs_xf <=52 then '合格'
 else '优秀'
 end
 as '等级'
from xs where xs.xs_zy='计算机';

-- 运算
-- 将学生成绩的百分制换算为150分制
select xs_kc.xs_id,xs_kc.kc_cj*1.5 from xs_kc where xs_kc.xs_id = '81101'

-- 查询学生表中比所有计算机系学生都大的学生学号，姓名，专业名，出生日期
select xs.xs_id,xs.xs_name,xs.xs_zy,xs.xs_rq from xs where xs.xs_rq < all (select xs.xs_rq from xs where xs.xs_zy='计算机')

-- 查询与81101性别相同，总学分相同的学生学号和姓名
select xs.xs_id,xs.xs_name from xs where (xs.xs_xb,xs.xs_xf) = (select xs.xs_xb,xs.xs_xf from xs where xs.xs_id='81103')

-- 查询学号为81101,81103两位同学的信息

select * from xs where xs.xs_id='81103'
union
select * from xs where xs.xs_id='81104'
```