很多查询都可以用经典的学生－课程－成绩案例来表示，下面是一些我在笔试或面试中遇到的经典题型。这些题目没有涉及到索引，函数，存储过程等高级功能，都是一些基本的查询语句，但难度却不小。

表结构

学生表student(id,name)

课程表course(id,name)

学生课程表student\_course(sid,cid,score)

sql

create table student(

id int unsigned primary key auto\_increment,

name char(10) not null

);

insert into student(name) values('张三'),('李四');

create table course(

id int unsigned primary key auto\_increment,

name char(20) not null

);

insert into course(name) values('语文'),('数学');

create table student\_course(

sid int unsigned,

cid int unsigned,

score int unsigned not null,

foreign key (sid) references student(id),

foreign key (cid) references course(id),

primary key(sid, cid)

);

insert into student\_course values(1,1,80),(1,2,90),(2,1,90),(2,2,70);

问题

查询student表中重名的学生，结果包含id和name，按name,id升序

select id,name

from student

where name in (

select name from student group by name having(count(\*) > 1)

) order by name;

我们经常需要查询某一列重复的行，一般通过group by(有重复的列)然后取count>1的值。 关系型数据库有他的局限性， 有些看似简单的查询写出来的sql很复杂，而且效率也会很低。

在student\_course表中查询平均分不及格的学生，列出学生id和平均分

select sid,avg(score) as avg\_score

from student\_course

group by sid having(avg\_score<60);

group by和having是最常考的。 where子句中不能用聚集函数作为条件表达式，但是having短语可以，where和having的区别在于对用对象不同，where作用于记录，having作用于组。

在student\_course表中查询每门课成绩都不低于80的学生id

select distinct sid

from student\_course

where sid not in (

select sid from student\_course

where score < 80);

用到反向思想，其实就是数理逻辑中的∀x:P和¬∃x:¬P是等价的。

查询每个学生的总成绩，结果列出学生姓名和总成绩 如果使用下面的sql会过滤掉没有成绩的人

select name,sum(score) total

from student,student\_course

where student.id=student\_course.sid

group by sid;

更保险的做法应该是使用 左外连接

select name,sum(score)

from student left join student\_course

on student.id=student\_course.sid

group by sid;

总成绩最高的学生，结果列出学生id和总成绩 下面的sql效率很低，因为要重复计算所有的总成绩。

select sid,sum(score) as sum\_score

from student\_course group by sid having sum\_score>=all

(select sum(score) from student\_course group by sid);

因为order by中可以使用聚集函数，最简单的方法是：

select sid,sum(score) as sum\_score

from student\_course group by sid

order by sum\_score desc limit 1;

同理可以查总成绩的前三名。

在student\_course表查询课程1成绩第2高的学生，如果第2高的不止一个则列出所有的学生

这是个查询 第N大数 的问题。 我们先查出第2高的成绩：

select min(score) from student\_course where cid = 1 group by score order by score desc limit 2;

使用这种方式是错的，因为作用的先后顺序是group by->min->order by->limit，mysql提供了limit offset,size这种方式来取第N大的值，因此正确的做法是：

select score from student\_course where cid = 1 group by score order by score desc limit 1,1;

然后再取出该成绩对应的学生：

select \* from student\_course

where cid=1 and score = (

select score from student\_course where cid = 1 group by score order by score desc limit 1,1

);

类似的，可以查询 某个值第N高 的记录。

在student\_course表查询各科成绩最高的学生，结果列出学生id、课程id和对应的成绩 你可能会这样写：

select sid,cid,max(score) from student\_course group by cid;

然而上面是不对的，因为 使用了group by的查询字段只能是group by中的字段或者聚集函数或者是每个分组内均相同的字段。 虽然不会报错，但是sid是无效的，如果去掉sid的话只能查出没门课程的最高分，不包含学生id。 本题的正确解法是使用相关嵌套查询：

select \* from student\_course as x where score>=

(select max(score) from student\_course as y where cid=x.cid);

相关嵌套查询也就是在进行内层查询的时候需要用到外层查询，有一些注意事项：

子查询一定要有括号

as可以省略

使用相关查询；>=max等价于>=all，但是聚合函数比使用any或all效率高

在student\_course表中查询每门课的前2名，结果按课程id升序，同一课程按成绩降序 这个问题也就是取每组的前N条纪录，类似的查询在csdn上也有征集答案

select \* from student\_course x where

2>(select count(\*) from student\_course y where y.cid=x.cid and y.score>x.score)

order by cid,score desc;

这也是一个相关嵌套查询，对于每一个分数，如果同一门课程下只有0个、1个分数比这个分数还高，那么这个分数肯定是前2名之一

一个叫team的表，里面只有一个字段name,一共有4条纪录，分别是a,b,c,d,对应四个球队，两两进行比赛，用一条sql语句显示所有可能的比赛组合

select a.name, b.name

from team a, team b

where a.name < b.name

其实就是一个表和自己连接查询。

题目：数据库中有一张如下所示的表，表名为sales。

年 季度 销售

1991 1 11

1991 2 12

1991 3 13

1991 4 14

1992 1 21

1992 2 22

1992 3 23

1992 4 24

要求：写一个SQL语句查询出如下所示的结果。

年 一季度 二季度 三季度 四季度

1991 11 12 13 14

1992 21 22 23 24

select 年,

sum(case when 季度=1 then 销售量 else 0 end) as 一季度,

sum(case when 季度=2 then 销售量 else 0 end) as 二季度,

sum(case when 季度=3 then 销售量 else 0 end) as 三季度,

sum(case when 季度=4 then 销售量 else 0 end) as 四季度

from sales group by 年;

同理，如果要查询每个人的每门课的成绩可以使用如下sql

create view temp as select student.name as sname,course.name as cname,score

from student\_course join (student,course)

on(student\_course.sid=student.id and student\_course.cid=course.id)

;

select sname,

sum(case when cname='语文' then score else 0 end) as 语文,

sum(case when cname='数学' then score else 0 end) as 数学

from temp

group by sname;

当然如果新增了一门课，第二条sql就需要跟着变。