尚硅谷大数据技术之Hive SQL题库-初级

（作者：尚硅谷研究院）

版本：V1.0

# 第一章 环境准备

## 1.1 建表语句

**hive>**

**-- 创建学生表**

DROP TABLE IF EXISTS student;

create table if not exists student(

stu\_id string COMMENT '学生id',

stu\_name string COMMENT '学生姓名',

birthday date COMMENT '出生日期',

sex string COMMENT '性别'

)

row format delimited fields terminated by ','

stored as textfile;

**-- 创建课程表**

DROP TABLE IF EXISTS course;

create table if not exists course(

course\_id string COMMENT '课程id',

course\_name string COMMENT '课程名',

tea\_id string COMMENT '任课老师id'

)

row format delimited fields terminated by ','

stored as textfile;

**-- 创建老师表**

DROP TABLE IF EXISTS teacher;

create table if not exists teacher(

tea\_id string COMMENT '老师id',

tea\_name string COMMENT '学生姓名'

)

row format delimited fields terminated by ','

stored as textfile;

**-- 创建分数表**

DROP TABLE IF EXISTS score;

create table if not exists score(

stu\_id string COMMENT '学生id',

course\_id string COMMENT '课程id',

course int COMMENT '成绩'

)

row format delimited fields terminated by ','

stored as textfile;

## 1.2 数据准备

（1）创建/opt/module/data目录

[atguigu@hadoop102 module]$ mkdir data

（2）将如下4个文件放到/opt/module/data目录下



（3）数据样式说明

①student表数据

001,彭于晏,1995-05-16,男

002,胡歌,1994-03-20,男

003,周杰伦,1995-04-30,男

004,刘德华,1998-08-28,男

005,唐国强,1993-09-10,男

②course表数据

01,语文,1003

02,数学,1001

03,英语,1004

04,体育,1002

05,音乐,1002

③teacher表数据

1001,张高数

1002,李体音

1003,王子文

1004,刘丽英

④score表数据

001,01,94

002,01,74

004,01,85

005,01,64

006,01,71

007,01,48

008,01,56

009,01,75

## 1.3 插入数据

（1）插入数据

**hive>**

load data local inpath '/opt/module/data/student.txt' into table student;

load data local inpath '/opt/module/data/course.txt' into table course;

load data local inpath '/opt/module/data/teacher.txt' into table teacher;

load data local inpath '/opt/module/data/score.txt' into table score;

（2）验证插入数据情况

**hive>**

select \* from student limit 5;

select \* from course limit 5;

select \* from teacher limit 5;

select \* from score limit 5;

# 第二章 简单查询

## 2.1 查找特定条件

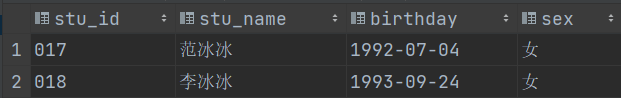
### 2.1.1 查询姓名中带“冰”的学生名单

select

\*

from student

where stu\_name like '%冰%';



### 2.1.2 查询姓“王”老师的个数

select

count(\*)

from teacher

where tea\_name like '王%';



### 2.1.3 检索课程编号为“04”且分数小于60的学生学号，结果按分数降序排列

select

stu\_id,

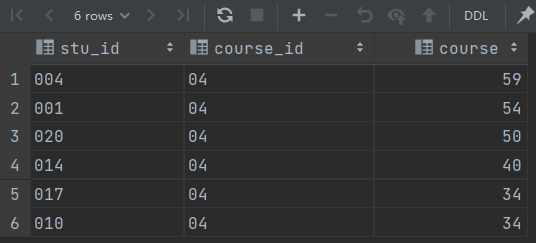
course\_id,

course

from score

where course\_id = '04' and course < 60

order by course desc;



### 2.1.4 查询数学成绩不及格的学生和其对应的成绩，按照学号升序排序

select

t1.stu\_id,

t1.stu\_name,

t2.course

from student as t1

join (

select

stu\_id,

course\_id,

course

from score

where course\_id = '2' and course < 60

) as t2

on t1.stu\_id = t2.stu\_id

order by t1.stu\_id;



## 2.2日期时间相关

### 2.2.1 查询各学生的年龄（精确到月份）

select

t.stu\_name,

concat(

if(month>=0,year,year-1),'岁',

if(month>=0,month,12+month),'个月'

) as age

from(

select

stu\_name,

year(`current\_date`())-year(birthday) year,

month(`current\_date`())-month(birthday) month

from student

) t;



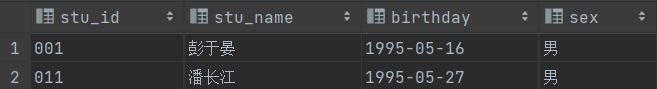
### 2.2.2 查询本月过生日的学生

select

\*

from student

where month(birthday) = month(current\_date());



# 第三章 汇总分析

## 3.1 汇总分析

### 3.1.1 查询课程编号为“02”的总成绩

select

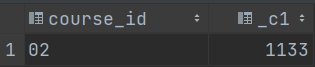
course\_id,

sum(course)

from score

where course\_id="02"

group by course\_id;



### 3.1.2 查询参加考试的学生个数

思路：对成绩表中的学号做去重并count

select

count(distinct stu\_id)

from score;



## 3.2 分组

### 3.2.1 查询各科成绩最高和最低的分，以如下的形式显示：课程号，最高分，最低分

思路：按照学科分组并使用max和min。

select

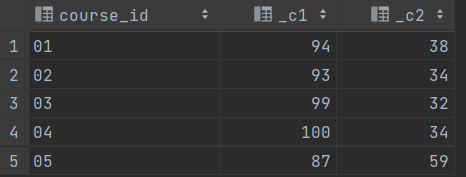
course\_id,

max(course),

min(course)

from score

group by course\_id;



### 3.2.2 查询每门课程有多少学生参加了考试（有考试成绩）

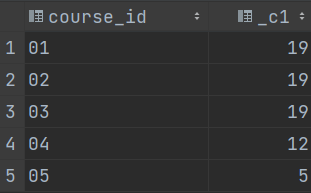
select

course\_id,

count(\*)

from score

group by course\_id;



### 3.2.3 查询男生、女生人数

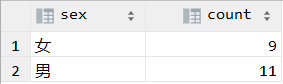
select

sex,

count(\*)

from student

group by sex;



## 3.3 分组结果的条件

### 3.3.1 查询平均成绩大于60分学生的学号和平均成绩

思路分析：

平均成绩：展开来说就是计算每个学生的平均成绩

这里涉及到“每个”就是要分组了

平均成绩大于60分，就是对分组结果指定条件

首先要分组求出每个学生的平均成绩，筛选高于60分的，并反查出这批学生，统计出这些学生总的平均成绩。

select

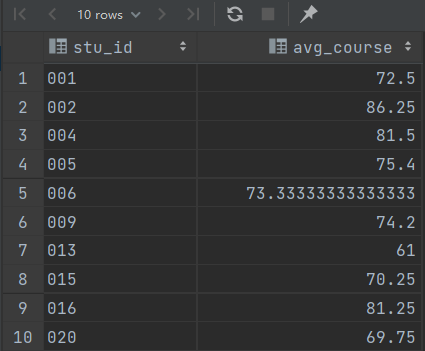
stu\_id,

avg(course) avg\_course

from score

group by stu\_id

having avg\_course > 60;



### 3.3.2 查询至少选修两门课程的学生学号

思路：

第1步，需要先计算出每个学生选修的课程数据，需要按学号分组

第2步，至少选修两门课程：也就是每个学生选修课程数目>=2，对分组结果指定条件

select

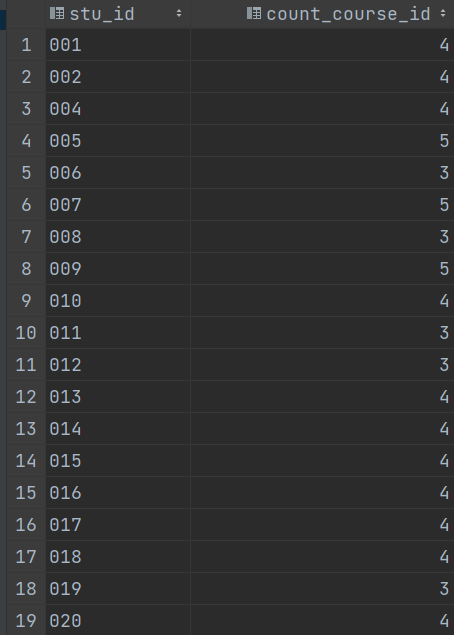
stu\_id,

count( distinct course\_id) count\_course\_id

from score

group by stu\_id

having count\_course\_id >=2;



### 3.3.3 查询同姓（假设每个学生姓名的第一个字为姓）的学生名单并统计同名人数

思路：先提取出每个学生的姓并分组，如果分组的count>=2则为同姓

select

t1.first\_stu\_name,

count(\*) count\_first\_stu\_name

from (

select

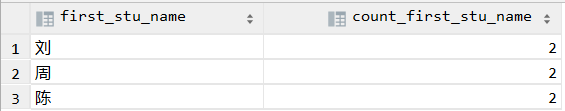
substring(stu\_name,1,1) first\_stu\_name

from student

)t1

group by t1.first\_stu\_name

having count\_first\_stu\_name >= 2;



### 3.3.4 查询每门课程的平均成绩，结果按平均成绩升序排序，平均成绩相同时，按课程号降序排列

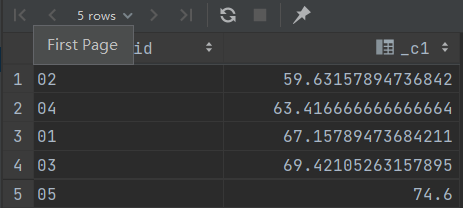
思路：按照课程号分组并求组内的平均值

select course\_id,avg(course)

from score

group by course\_id

order by avg(course) asc,course\_id desc;



### 3.3.5 统计参加考试人数大于等于15的学科

按课程分组并统计组内人数，过滤条件大于等于15

select

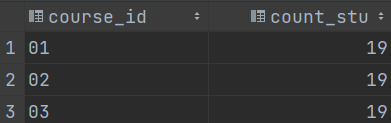
course\_id,

count(stu\_id) as count\_stu

from score

group by course\_id

having count\_stu > 15;



## 3.4 查询结构排序&分组指定条件

### 3.4.1 查询学生的总成绩并进行排名

思路：分组、sum、排序

select

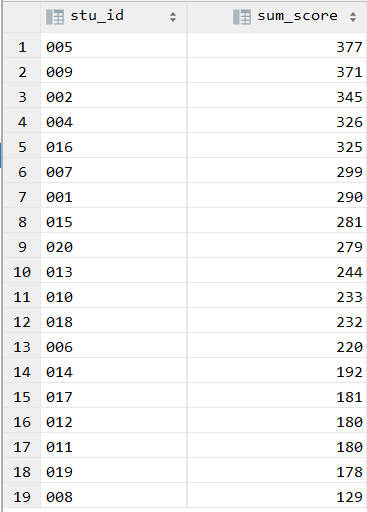
stu\_id,

sum(course) sum\_score

from score

group by stu\_id

order by sum\_score desc;



### 3.4.2 查询平均成绩大于60分的学生的学号和平均成绩

思路：分组，avg，过滤>=60

select tmp.stu\_id,avg(tmp.course) from

(

select

student.stu\_id,

score.stu\_id,

score.course\_id,

score.course

from student

inner join score

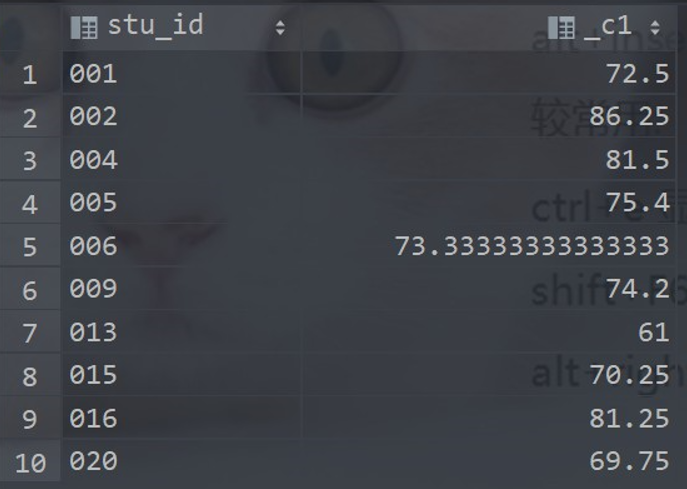
on

student.stu\_id = score.stu\_id

) tmp

group by tmp.stu\_id

having avg(tmp.course) > 60;



### 3.4.3 按照如下格式显示学生的语文、数学、英语三科成绩，没有成绩的输出为0，按照学生的有效平均成绩降序显示

学生id 学生姓名 语文 数学 英语 有效课程数 有效平均成绩

select stu\_id as `学生ID`

,(select nvl(course,0) as course from score where score.stu\_id = t.stu\_id and course\_id = '01') as `语文`

,(select nvl(course,0) as course from score where score.stu\_id = t.stu\_id and course\_id = '02') as `数学`

,(select nvl(course,0) as course from score where score.stu\_id = t.stu\_id and course\_id = '03') as `英语`

,count(\*) as `有效课程数`,avg(t.course) as `平均成绩`

FROM score as t

group by stu\_id

order by avg(t.course);



### 3.4.4 查询一共参加两门课程且其中一门为语文课程的学生的id和姓名

select

t1.stu\_id,

count(course\_id) as c

from (

select stu\_id,

course\_id

from score

where stu\_id in (select stu\_id

from score

where course\_id = '01')

) as t1

group by t1.stu\_id

having c = 2;

答案：空

参加三门的课程：

select

t1.stu\_id,

count(course\_id) as c

from (

select stu\_id,

course\_id

from score

where stu\_id in (select stu\_id

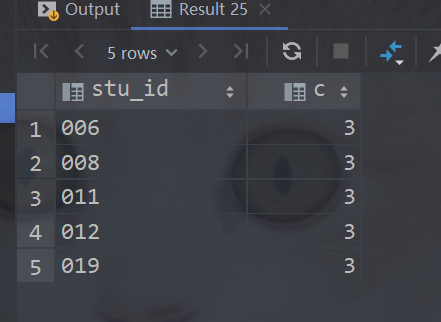
from score

where course\_id = '01')

) as t1

group by t1.stu\_id

having c = 3;



# 第四章 复杂查询

## 4.1 子查询

### 4.1.1 查询所有课程成绩小于60分学生的学号、姓名

select

stu\_id,stu\_name

from

student

where stu\_id not in (

select

student.stu\_id

from

student

inner join

score

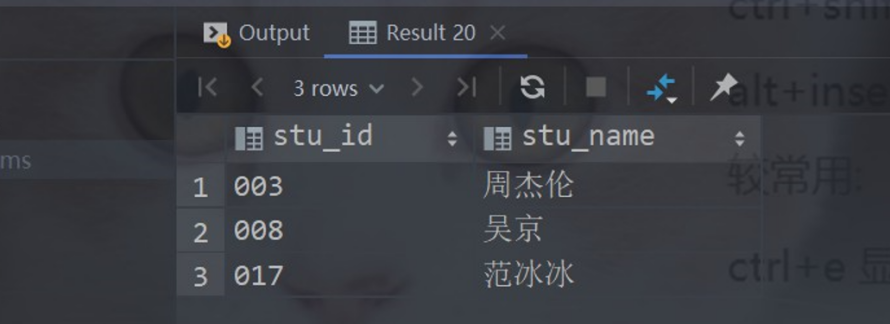
on

student.stu\_id = score.stu\_id

where

score.course > 60

);



### 4.1.2 查询没有学全所有课的学生的学号、姓名

解释：没有学全所有课，也就是该学生选修的课程数 < 总的课程数

select

student.stu\_id,student.stu\_name

from

student

inner join

score

on

student.stu\_id = score.stu\_id

group by student.stu\_id,student.stu\_name

having count(course\_id) < (select

count(course\_id)

from

course);



### 4.1.3 查询出只选修了两门课程的全部学生的学号和姓名

解释：学生选修的课程数=2

select

stu\_id,stu\_name

from student

where stu\_id in(

select

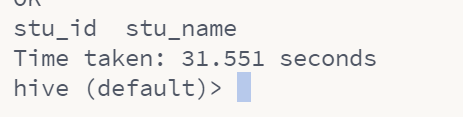
stu\_id

from score

group by stu\_id

having count(1) ==2

);



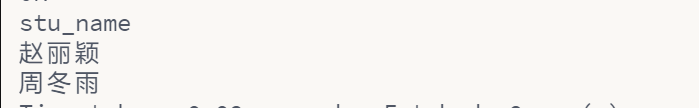
### 4.1.4 查找1990年出生的学生名单

select

stu\_name

from student

where date\_format(birthday,yyyy)==1990;



# 第五章 多表查询

## 5.1 表联结

### 5.1.1 查询两门以上不及格课程的同学的学号及其平均成绩

先找出有两门以上不及格的学生名单，按照学生分组，过滤组内成绩低于60的并进行count，count>=2。

接着做出一张表查询学生的平均成绩并和上一个子查询中的学生学号进行连接

select

t3.stu\_id sur\_id,t2.avg\_course avg

from (

select

t1.stu\_id

from (

select

stu\_id,count(course) count\_num

from score

where course<60

group by stu\_id

) t1

where t1.count\_num >=2

)t3

join (

select

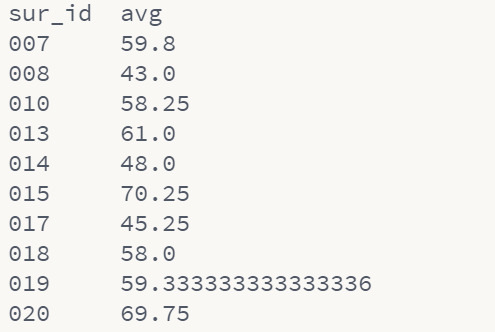
stu\_id,avg(course) avg\_course

from score

group by stu\_id

) t2

on t2.stu\_id = t3.stu\_id;



### 5.1.2 查询所有学生的学号、姓名、选课数、总成绩

select

s.stu\_id,s.stu\_name,t1.cor\_num,t1.sum

from student s

join (

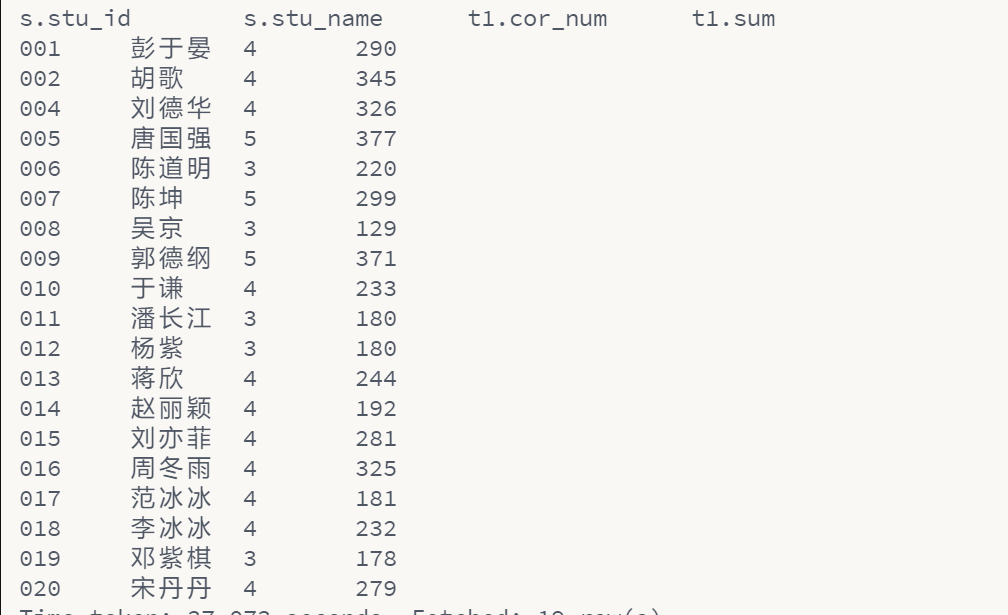
select stu\_id,count(course\_id) cor\_num ,sum(course) sum

from score

group by stu\_id

) t1

on t1.stu\_id = s.stu\_id;



### 5.1.3 查询平均成绩大于85的所有学生的学号、姓名和平均成绩

select

distinct

stu\_id,

stu\_name,

ag

from(

select

stu.stu\_id,

stu.stu\_name,

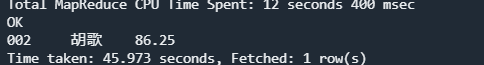
avg(course) over(partition by stu.stu\_id) ag

from score sc left join student stu

on sc.stu\_id = stu.stu\_id

) t1

where ag > 85;



### 5.1.4 查询学生的选课情况：学号，姓名，课程号，课程名称

select

t1.stu\_id,

stu\_name,

t1.course\_id,

t1.course\_name

from

(select

stu\_id,

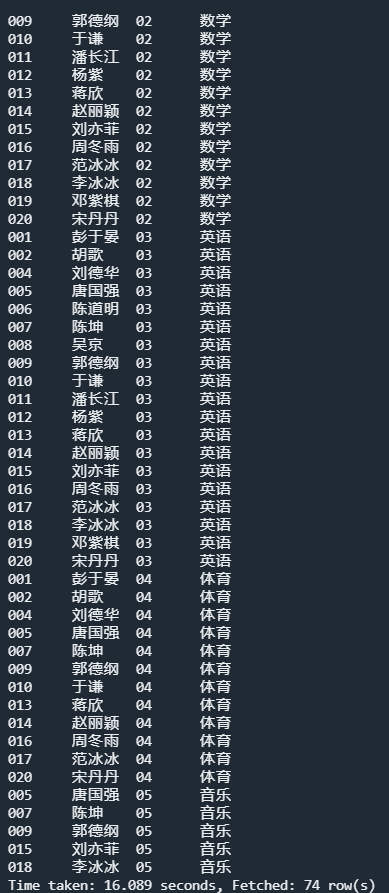
co.course\_id,

co.course\_name

from score sc left join course co

on sc.course\_id = co.course\_id) t1 left join student stu

on stu.stu\_id = t1.stu\_id;



### 5.1.5 查询出每门课程的及格人数和不及格人数

select

tmp1.course\_id,

tmp1.course\_name,

tmp1.stu\_num,

tmp2.u\_stu\_num

from

(select

t1.course\_id,

co.course\_name,

t1.stu\_num

from

(select

count(stu\_id) stu\_num,

course\_id

from score

where course >=60

group by course\_id

)t1 left join course co

on t1.course\_id = co.course\_id) tmp1

left join

(select

t1.course\_id,

co.course\_name,

t1.u\_stu\_num

from

(select

count(stu\_id) u\_stu\_num,

course\_id

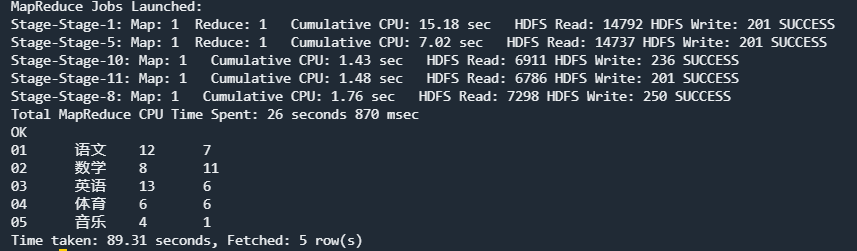
from score

where course < 60

group by course\_id

)t1 left join course co

on t1.course\_id = co.course\_id) tmp2

on tmp1.course\_id=tmp2.course\_id;

### 5.1.6 使用分段[100-85],[85-70],[70-60],[<60]来统计各科成绩，分别统计：各分数段人数，课程号和课程名称

select

sec,

stu\_num,

course\_id,

course\_name

from(

select

'1' sec,

t1.stu\_num,

t1.course\_id,

co.course\_name

from

(select

count(stu\_id) stu\_num,

course\_id

from score

where course >85 and course <=100

group by course\_id

)t1 left join course co

on t1.course\_id = co.course\_id

union all

select

'2' sec,

t1.stu\_num,

t1.course\_id,

co.course\_name

from(

select

count(stu\_id) stu\_num,

course\_id

from score

where course >70 and course <=85

group by course\_id

)t1 left join course co

on t1.course\_id = co.course\_id

union all

select

'3' sec,

t1.stu\_num,

t1.course\_id,

co.course\_name

from(

select

count(stu\_id) stu\_num,

course\_id

from score

where course >60 and course <=70

group by course\_id

)t1 left join course co

on t1.course\_id = co.course\_id

union all

select

'4' sec,

t1.stu\_num,

t1.course\_id,

co.course\_name

from(

select

count(stu\_id) stu\_num,

course\_id

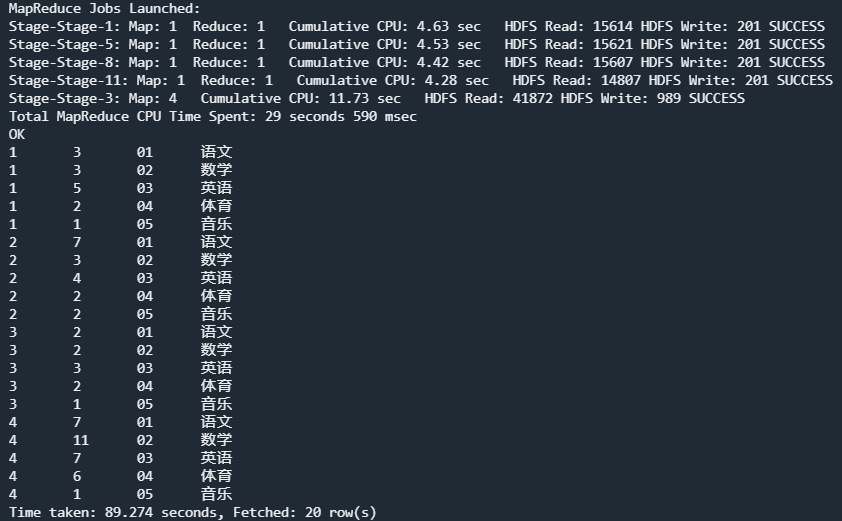
from score

where course <= 60

group by course\_id

)t1 left join course co

on t1.course\_id = co.course\_id)tmp;



### 5.1.7 查询课程编号为03且课程成绩在80分以上的学生的学号和姓名

select

st.stu\_id,

st.stu\_name,

sc.course,

co.course\_id,

co.course\_name

from student as st

inner join score as sc

on st.stu\_id = sc.stu\_id

inner join course as co

on sc.course\_id = co.course\_id

where co.course\_id = '03' and sc.course > 80;

结果：



### 5.1.8 （重要！行转列）使用sql实现将该表行转列为下面的表结构

如果没有该课程成绩用0代替。

学号 课程01 课程02 课程03 课程04

001 94 63 79 54

select stu\_id,

max(case course\_id when '01' then course else 0 end),

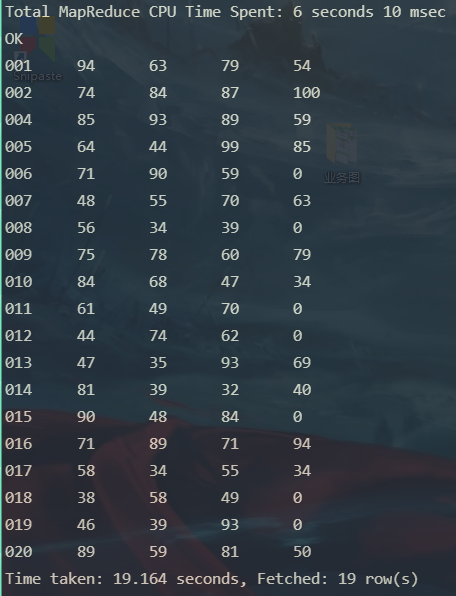
max(case course\_id when '02' then course else 0 end),

max(case course\_id when '03' then course else 0 end),

max(case course\_id when '04' then course else 0 end)

from score

group by stu\_id;



## 5.2 多表连接

### 5.2.1 检索"01"课程分数小于60，按分数降序排列的学生信息

select st.\*,sc.course

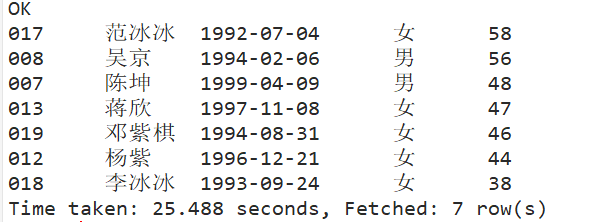
from student st

left join score sc

on sc.stu\_id = st.stu\_id

where sc.course\_id = '01' AND sc.course< 60

order by sc.course desc;



### 5.2.2 查询任何一门课程成绩在70分以上的学生的姓名、课程名称和分数

1）只要有一门课程超70分：

select

stu\_name,

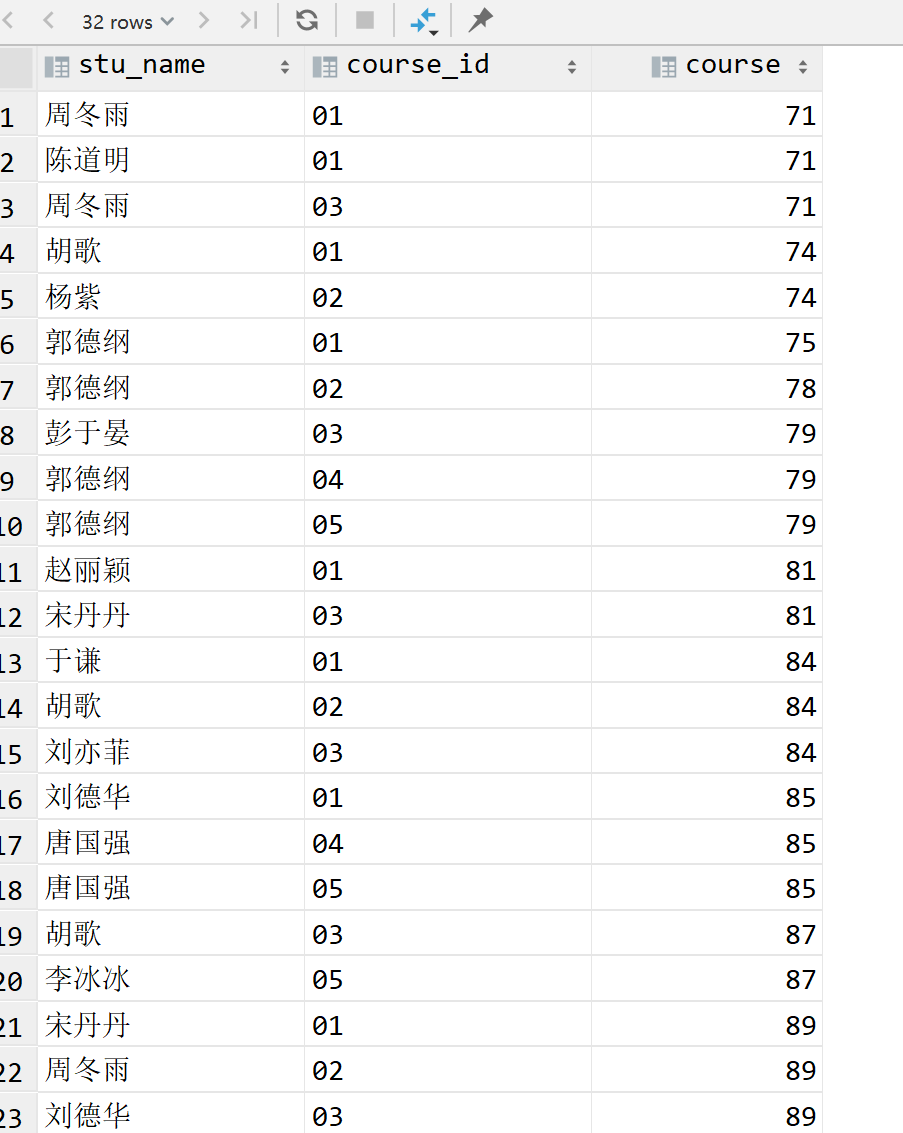
c.course\_id,

course

from student join score s on student.stu\_id = s.stu\_id join course c on s.course\_id = c.course\_id

where course >70

group by stu\_name, c.course\_id, course;



2）所有的课程都在70分以上：

select

st2.stu\_id,

st2.stu\_name,

c2.course\_name,

sc2.course

from student st2

left join score sc2 on sc2.stu\_id = st2.stu\_id

left join course c2 on c2.course\_id = sc2.course\_id

where st2.stu\_id IN

(select

stu\_id

from

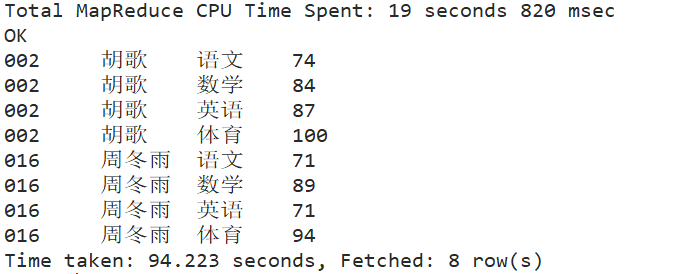
student st

left join score sc on sc.stu\_id = st.stu\_id

group by st.stu\_id having MIN(sc.course)>=70

order by stu\_id);

结果：



### 5.2.3 查询两门及其以上不及格课程的同学的学号，姓名及其平均成绩

知识点：分组 + 条件 + 多表连接

思路：计算每个学号不及格分数个数，筛选出大于2个的学号并找出姓名，平均成绩

select

stu\_id,

stu\_name,

avgcourse

from (

select

stu\_id,

stu\_name,

count(1) num,

avg(course) avgcourse

from (

select s.stu\_id,

stu\_name,

course

from student

join score s on student.stu\_id = s.stu\_id

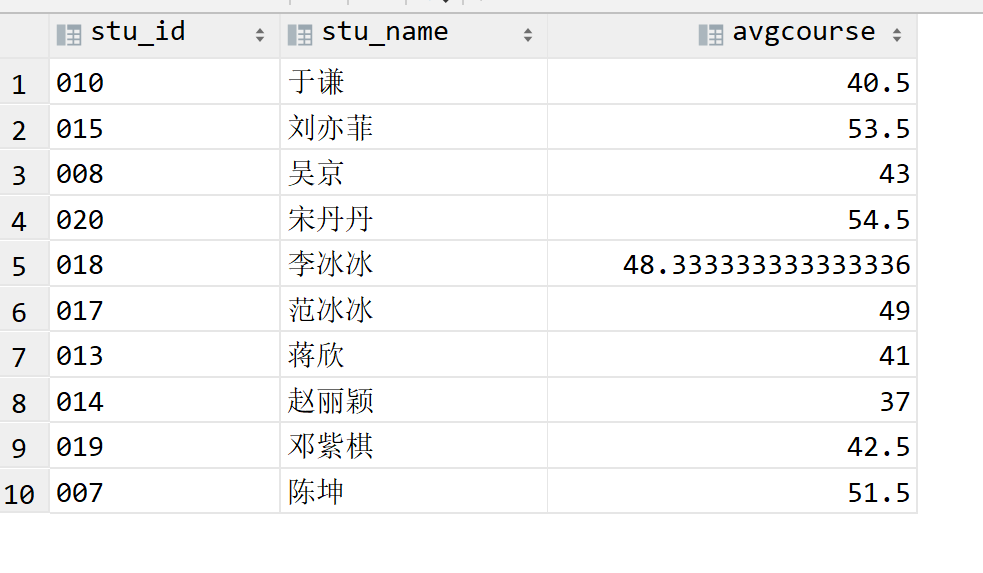
where course < 60

group by s.stu\_id, stu\_name, course

) t1

group by stu\_id, stu\_name ) t2

where num >=2;

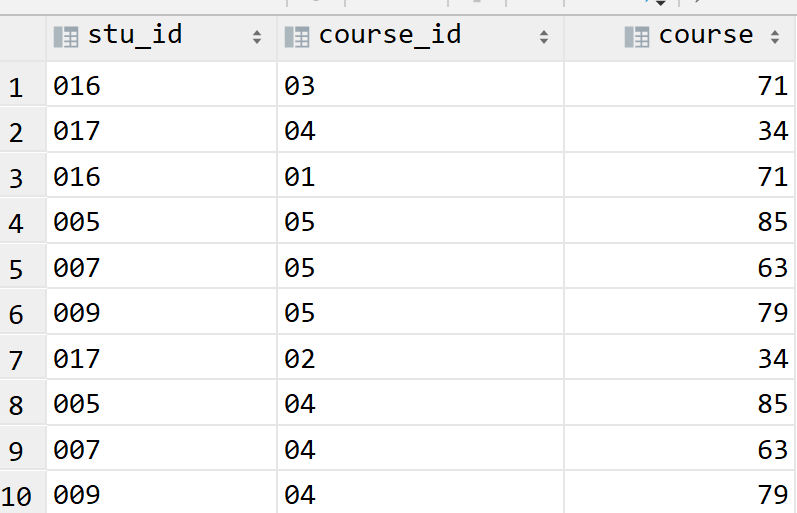


### 5.2.4 查询不同课程成绩相同的学生的学生编号、课程编号、学生成绩

select

s1.\*

from score s1 inner join score s2 on s1.stu\_id=s2.stu\_id and s1.course\_id<>s2.course\_id and s1.course=s2.course;



### 5.2.5 查询课程编号为“01”的课程比“02”的课程成绩高的所有学生的学号

知识点：多表连接 + 条件

select

stu\_id

from (

select

s1.stu\_id,s1.stu\_name,s1.course 01course,s2.course 02course

from (select

s.stu\_id,stu\_name,c.course\_id,course

from student join score s on student.stu\_id = s.stu\_id join course c on s.course\_id = c.course\_id

where c.course\_id ='02'

) s1 join (

select

s.stu\_id,stu\_name,c.course\_id,course

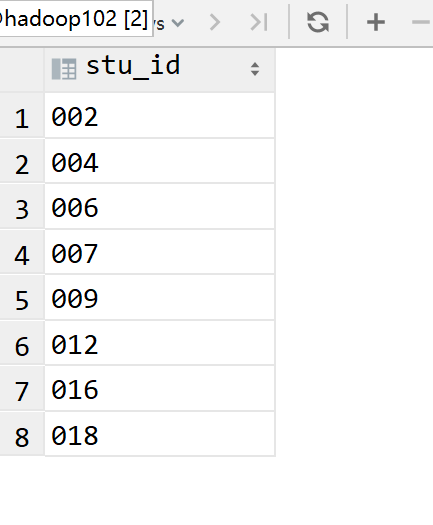
from student join score s on student.stu\_id = s.stu\_id join course c on s.course\_id = c.course\_id

where c.course\_id ="01"

)s2 on s1.stu\_id=s2.stu\_id

) s3

where 01course > 02course;



### 5.2.6 查询学过编号为“01”的课程并且也学过编号为“02”的课程的学生的学号、姓名

select

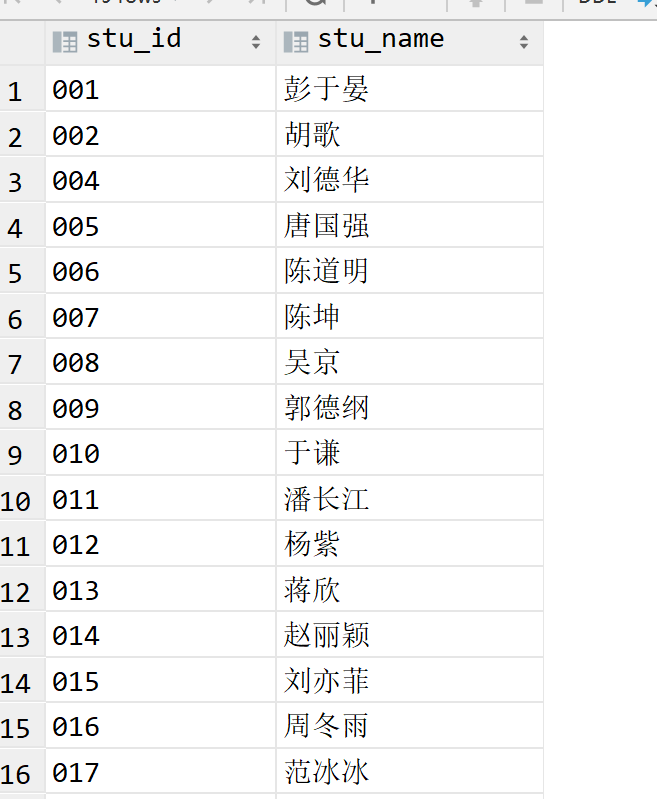
student.stu\_id,

student.stu\_name

from student,score

where student.stu\_id = score.stu\_id and score.course\_id='01'

and exists (select \* from score as sc\_2 where sc\_2.stu\_id = score.stu\_id and sc\_2.course\_id='02');



### 5.2.7 查询学过“李体音”老师所教的所有课的同学的学号、姓名

select

st.stu\_id,

st.stu\_name

from student st

join (select

stu\_id

from score s join (select

course\_id

from course c

join teacher t

on c.tea\_id = t.tea\_id

where tea\_name='李体音'

) t1

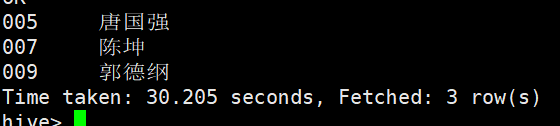
on s.course\_id = t1.course\_id

) t2

on st.stu\_id = t2.stu\_id

group by st.stu\_id, st.stu\_name

having count(st.stu\_id) = 2;



### 5.2.8 查询学过“李体音”老师所讲授的任意一门课程的学生的学号、姓名

select

distinct st.stu\_id,

st.stu\_name

from student st

join (select

stu\_id

from score s join (select

course\_id

from course c join teacher t

on c.tea\_id = t.tea\_id

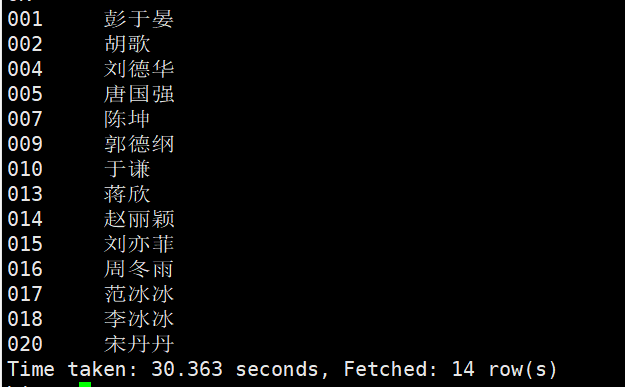
where tea\_name='李体音'

) t1

on s.course\_id = t1.course\_id

) t2

on st.stu\_id = t2.stu\_id;



### 5.2.9 查询没学过"李体音"老师讲授的任一门课程的学生姓名

select

stu\_id,

stu\_name

from student

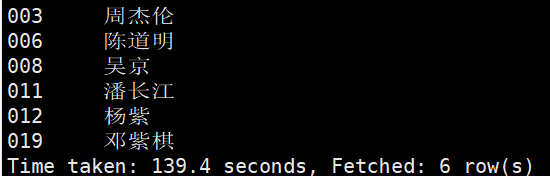
where stu\_id not in (

select distinct (stu\_id)

from score sc,course c,teacher t

where sc.course\_id = c.course\_id and t.tea\_id = c.tea\_id and t.tea\_name = '李体音'

);



### 5.2.10 查询选修“李体音”老师所授课程的学生中成绩最高的学生姓名及其成绩（与上题类似,用成绩排名，用 limit 1得出最高一个）

select

st.stu\_name,

t2.course

from student st

join (select

stu\_id,

course

from score s join (select

course\_id

from course c join teacher t

on c.tea\_id = t.tea\_id

where tea\_name='李体音'

) t1

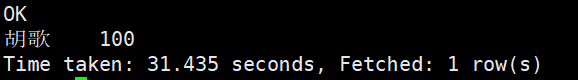
on s.course\_id = t1.course\_id

) t2

on st.stu\_id = t2.stu\_id

order by t2.course desc

limit 1;



### 5.2.11 查询至少有一门课与学号为“001”的学生所学课程相同的学生的学号和姓名

select

distinct s3.stu\_id,

s3.stu\_name

from student s3 join score s4

on s3.stu\_id = s4.stu\_id

where s3.stu\_id != '001'

and s4.course\_id in (

select

s2.course\_id

from student s1 join score s2

on s1.stu\_id = s2.stu\_id

where s1.stu\_id = '001');



### 5.2.12查询所学课程与学号为“001”的学生所学课程完全相同的学生的学号和姓名

select

t3.stu\_id,

t3.stu\_name

from (

select

\*

from (

select

stu\_id,

concat\_ws(',',collect\_list(course\_id)) as c\_w

from score

group by stu\_id

) as t1

where t1.c\_w in (

select

concat\_ws(',',collect\_list(course\_id))

from (

select

\*

from score

where stu\_id = '001'

order by course\_id

) as t1

group by t1.stu\_id

)

) as t2

inner join student t3

on t2.stu\_id = t3.stu\_id

where t3.stu\_id != '001';



### 5.2.13 按平均成绩从高到低显示所有学生的所有课程的成绩以及平均成绩

select

s.stu\_name,

a\_c.course\_name,

a\_c.course,

a\_c.avg\_c

from (

select a\_s.stu\_id,

c.course\_name,

a\_s.course,

a\_s.avg\_c

from ( select

stu\_id,

course\_id,

course,

avg(course) over(partition by stu\_id ) avg\_c

from score

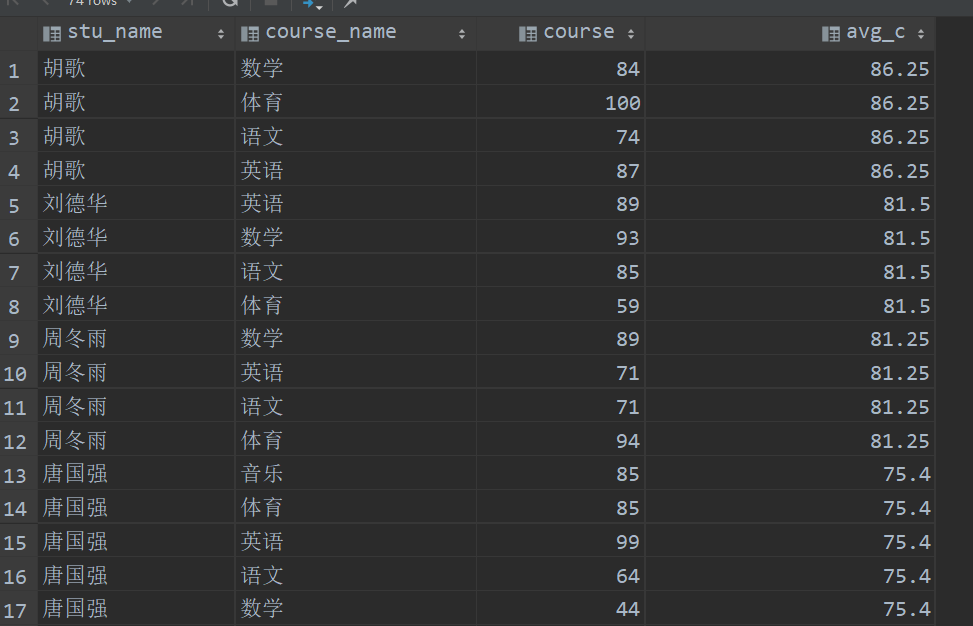
) a\_s

join course c

on a\_s.course\_id = c.course\_id

) a\_c join student s on a\_c.stu\_id = s.stu\_id

order by a\_c.avg\_c desc;



# 第六章 窗口查询

## 6.1 窗口的使用

### 6.1.1 查询每个学生的学生平均成绩及其名次

窗口函数排名

select student.stu\_id

,stu\_name

,birthday

,sex

,avg\_score

,rank() over(order by avg\_score desc)

from student

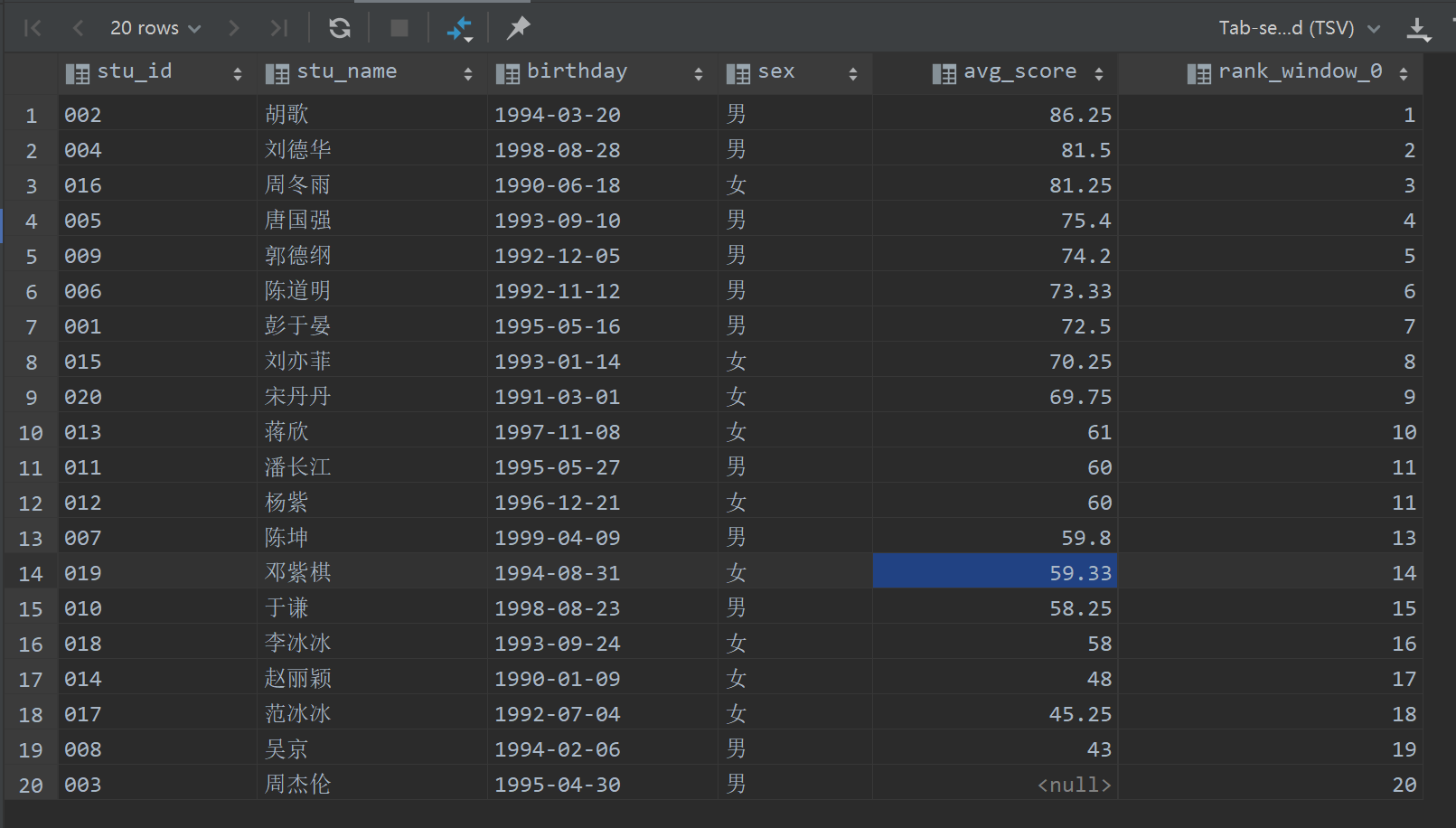
left join

(select stu\_id,round(avg(course),2) avg\_score

from score

group by stu\_id) t1

on student.stu\_id=t1.stu\_id;



### 6.1.2 按各科成绩进行排序，并显示在这个学科中的排名

select t1.stu\_name,t1.course\_name,t1.course

,rank() over(partition by course\_name order by course desc) rk

from (

select \*

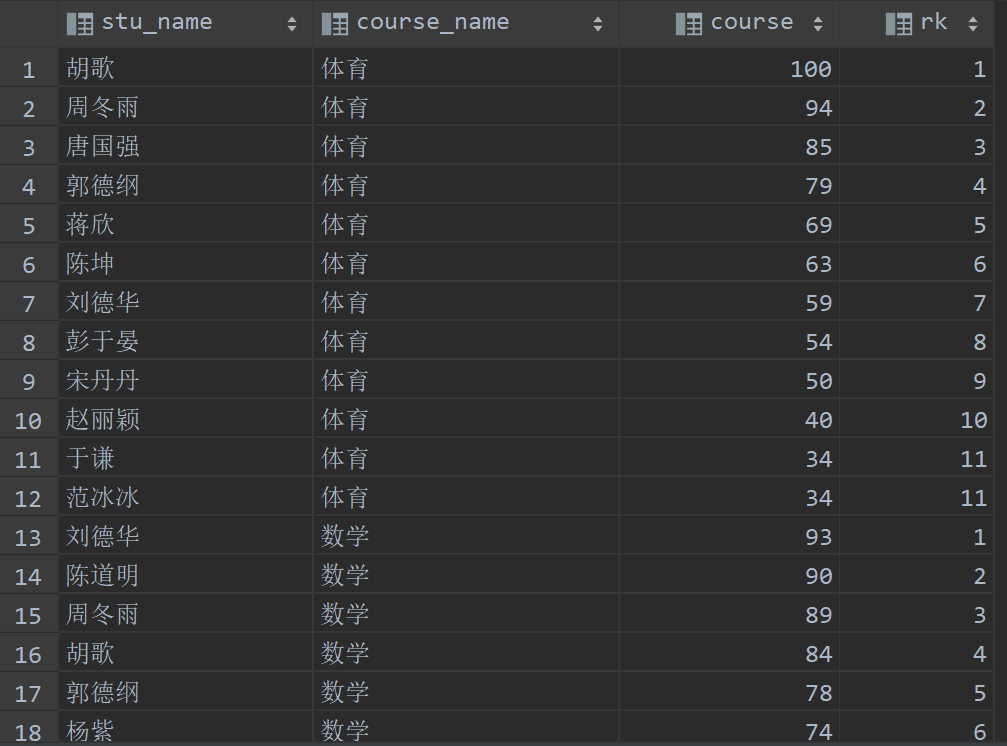
from student

left join score on student.stu\_id = score.stu\_id

left join course on score.course\_id = course.course\_id

where course is not null

) t1;



### 6.1.3 查询每门课程成绩最好的前两名学生姓名

窗口函数排名+多表连接+条件

select t2.stu\_name

from

(

select t1.stu\_name

, t1.course\_name

, t1.course

, rank() over (partition by course\_name order by course desc) rk

from (

select \*

from student

left join score on student.stu\_id = score.stu\_id

left join course on score.course\_id = course.course\_id

where course is not null

) t1

)t2

where rk <= 2 ;



### 6.1.4 查询所有课程的成绩第2名到第3名的学生信息及该课程成绩

select stu\_name,course\_name,tea\_name,course,rk

from

(

select t1.stu\_name

, t1.course\_name

, t1.course

,t1.tea\_name

, rank() over (partition by course\_name order by course desc) rk

from (

select \*

from student

left join score on student.stu\_id = score.stu\_id

left join course on score.course\_id = course.course\_id

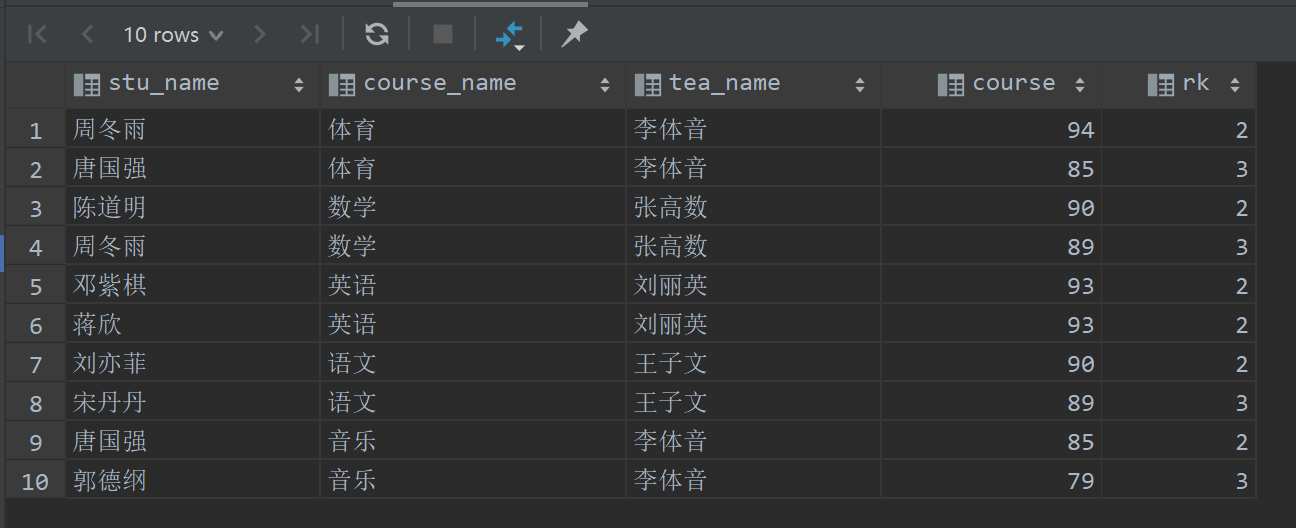
left join teacher on teacher.tea\_id = course.tea\_id

where course is not null

) t1

)t2

where rk = 2 or rk = 3;



### 6.1.5 查询各科成绩前三名的记录（如果有并列，则全部展示，例如如果前7名为：80,80,80,79,79,77,75,70，则统计结果为数字的前三名，结果为80,80,80,79,79,77）

select stu\_name,course\_name,tea\_name,course,rk

from

(

select t1.stu\_name

, t1.course\_name

, t1.course

, t1.tea\_name

, rank() over (partition by course\_name order by course desc) rk

from (

select \*

from student

left join score on student.stu\_id = score.stu\_id

left join course on score.course\_id = course.course\_id

left join teacher on teacher.tea\_id = course.tea\_id

where course is not null

) t1

)t2

where rk <= 3;