# 建表代码

CREATE TABLE Class(

class\_id INT(10) PRIMARY KEY NOT NULL UNIQUE AUTO\_INCREMENT,

class\_name VARCHAR(64) NOT NULL

)ENGINE = INNODB DEFAULT CHARSET = utf8;

INSERT INTO `class`

VALUES

( '1', '三年二班' ),

( '2', '三年三班' ),

( '3', '一年二班' ),

( '4', '二年九班' );

CREATE TABLE Subject(

subject\_id INT(10) PRIMARY KEY NOT NULL UNIQUE AUTO\_INCREMENT,

subject\_name VARCHAR(64) NOT NULL,

teacher\_id INT(10)

)ENGINE = INNODB DEFAULT CHARSET = utf8;

INSERT INTO `Subject`

VALUES

( '1', '生物', '1' ),

( '2', '物理', '2' ),

( '3', '体育', '3' ),

( '4', '美术', '2' );

CREATE TABLE score(

score\_id INT(10) PRIMARY KEY NOT NULL UNIQUE AUTO\_INCREMENT,

student\_id INT(10),

subject\_id INT(10) NOT NULL,

score INT(10)

);

INSERT INTO `score`

VALUES

( '1', '1', '1', '10' ),

( '2', '1', '2', '9' ),

( '5', '1', '4', '66' ),

( '6', '2', '1', '8' ),

( '8', '2', '3', '68' ),

( '9', '2', '4', '99' ),

( '10', '3', '1', '77' ),

( '11', '3', '2', '66' ),

( '12', '3', '3', '87' ),

( '13', '3', '4', '99' ),

( '14', '4', '1', '79' ),

( '15', '4', '2', '11' ),

( '16', '4', '3', '67' ),

( '17', '4', '4', '100' ),

( '18', '5', '1', '79' ),

( '19', '5', '2', '11' ),

( '20', '5', '3', '67' ),

( '21', '5', '4', '100' ),

( '22', '6', '1', '9' ),

( '23', '6', '2', '100' ),

( '24', '6', '3', '67' ),

( '25', '6', '4', '100' ),

( '26', '7', '1', '9' ),

( '27', '7', '2', '100' ),

( '28', '7', '3', '67' ),

( '29', '7', '4', '88' ),

( '30', '8', '1', '9' ),

( '31', '8', '2', '100' ),

( '32', '8', '3', '67' ),

( '33', '8', '4', '88' ),

( '34', '9', '1', '91' ),

( '35', '9', '2', '88' ),

( '36', '9', '3', '67' ),

( '37', '9', '4', '22' ),

( '38', '10', '1', '90' ),

( '39', '10', '2', '77' ),

( '40', '10', '3', '43' ),

( '41', '10', '4', '87' ),

( '42', '11', '1', '90' ),

( '43', '11', '2', '77' ),

( '44', '11', '3', '43' ),

( '45', '11', '4', '87' ),

( '46', '12', '1', '90' ),

( '47', '12', '2', '77' ),

( '48', '12', '3', '43' ),

( '49', '12', '4', '87' ),

( '52', '13', '3', '87' );

CREATE TABLE Student(

student\_id INT(10) PRIMARY KEY NOT NULL UNIQUE AUTO\_INCREMENT,

sex VARCHAR(8) NOT NULL,

class\_id INT(10),

student\_name VARCHAR(64)

)ENGINE = INNODB DEFAULT CHARSET = utf8;

INSERT INTO `student`

VALUES

( '1', '男', '1', '理解' ),

( '2', '女', '1', '钢蛋' ),

( '3', '男', '1', '张三' ),

( '4', '男', '1', '张一' ),

( '5', '女', '1', '张二' ),

( '6', '男', '1', '张四' ),

( '7', '女', '2', '铁锤' ),

( '8', '男', '2', '李三' ),

( '9', '男', '2', '李一' ),

( '10', '女', '2', '李二' ),

( '11', '男', '2', '李四' ),

( '12', '女', '3', '如花' ),

( '13', '男', '3', '刘三' ),

( '14', '男', '3', '刘一' ),

( '15', '女', '3', '刘二' ),

( '16', '男', '3', '刘四' );

CREATE TABLE Teacher(

teacher\_id INT(10) PRIMARY KEY NOT NULL UNIQUE AUTO\_INCREMENT,

teacher\_name VARCHAR(64) NOT NULL

)ENGINE = INNODB DEFAULT CHARSET = utf8;

INSERT INTO `teacher`

VALUES

( '1', '张磊老师' ),

( '2', '李平老师' ),

( '3', '刘海燕老师' ),

( '4', '朱云海老师' ),

( '5', '李杰老师' );

# 后续对表的更新，最终表内容

INSERT INTO `Subject`

VALUES

( '5', '数据库', '6' );

/\*INSERT INTO `score`

VALUES

( '53', '13', '5', '100' ); 后来删掉了这一条\*/

INSERT INTO `teacher`

VALUES

( '6', '叶平老师' );

INSERT INTO `student`

VALUES

( '17', '男', '4', '理解' ),

( '18', '女', '4', '钢蛋' ),

( '19', '男', '4', '张三' ),

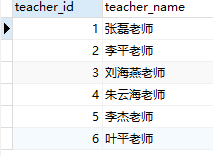
( '20', '男', '5', '张二' ),

( '21', '女', '5', '张二' );

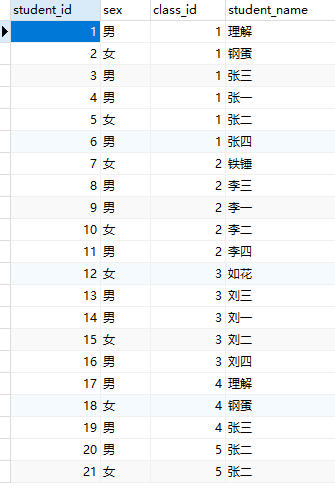
Subject表



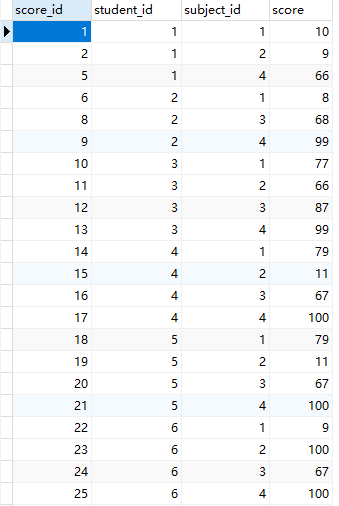
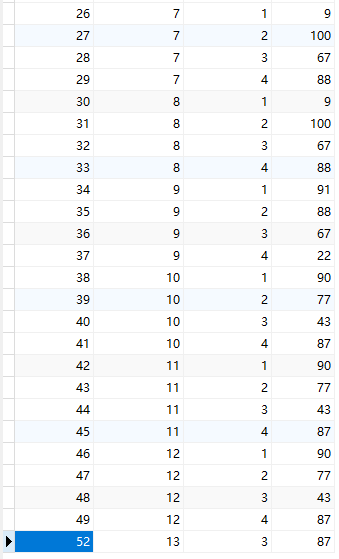
Teacher表



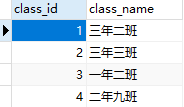
Student表



Score表

Class表



# 题目

## #1.查询男、女生的人数

SELECT

sex,

COUNT(sex) '人数'

FROM

student

GROUP BY

sex;



## #2.查询姓“张”的学生名单

SELECT

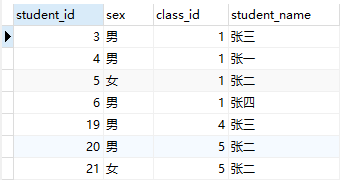
\*

FROM

student

WHERE

student\_name LIKE '张%'



## #3.课程平均分从高到低显示

SELECT

sub.subject\_name '课程名称',

AVG( sc.score ) '平均分排序'

FROM

score sc

LEFT JOIN SUBJECT sub ON sc.subject\_id = sub.subject\_id

GROUP BY

sc.subject\_id

ORDER BY

AVG(sc.score) DESC;



## #4.查询有课程成绩小于60分的同学的学号、姓名

SELECT DISTINCT

stu.student\_id '有成绩小于60分的同学的学号',

stu.student\_name '姓名'

FROM

score sc,

student stu

WHERE

sc.score < 60 AND sc.student\_id = stu.student\_id;



## #5.查询至少有一门课与学号为1的同学所学课程相同的同学的学号和姓名

SELECT DISTINCT

c.student\_id '学号为1的同学所学课程相同的同学的学号',

c.student\_name '姓名'

FROM

score b,

student c

WHERE

b.subject\_id IN ( SELECT subject\_id FROM score WHERE student\_id = 1 )

AND c.student\_id = b.student\_id;



## #6.查询出只选修了一门课程的全部学生的学号和姓名

SELECT

stu.student\_id '学号',

stu.student\_name '姓名'

FROM

(SELECT student\_id,COUNT(subject\_id) '选修的课程数' FROM score GROUP BY student\_id) a,

student stu

WHERE

a.`选修的课程数` = 1

AND a.student\_id = stu.student\_id;



## #7.查询各科成绩最高和最低的分：以如下形式显示：课程ID，最高分，最低分

SELECT

subject\_id '课程ID',

MAX(score) '最高分',

MIN(score) '最低分'

FROM

score

GROUP BY

subject\_id;



## #8.查询课程编号“2”的成绩比课程编号“1”课程低的所有同学的学号、姓名；

SELECT DISTINCT

stu.student\_id '学号',

stu.student\_name '姓名'

FROM

(SELECT score\_id,score FROM score WHERE subject\_id = 2) a,

score sc,

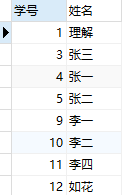
student stu

WHERE

sc.subject\_id = 1

AND a.score < sc.score

AND sc.student\_id = stu.student\_id;



## #9.查询“生物”课程比“物理”课程成绩高的所有学生的学号

SELECT

a.`学号` ,

a.`生物成绩`,

b.`物理成绩`

FROM

(

SELECT

sc.score '生物成绩',

sc.student\_id '学号'

FROM

SUBJECT sub,

score sc

WHERE

sub.subject\_name = '生物'

AND sub.subject\_id = sc.subject\_id

) a,

(

SELECT

sc.score '物理成绩',

sc.student\_id '学号'

FROM

SUBJECT sub,

score sc

WHERE

sub.subject\_name = '物理'

AND sub.subject\_id = sc.subject\_id

) b

WHERE

a.`生物成绩` > b.`物理成绩`

AND a.`学号` = b.`学号`;



## #10.查询平均成绩大于60分的同学的学号和平均成绩

SELECT

student\_id,

AVG(score) avgGrade

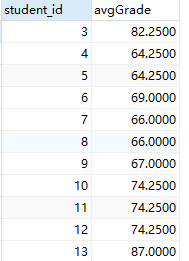
FROM

score

GROUP BY

student\_id

HAVING avgGrade > 60;



## #11.查询所有同学的学号、姓名、选课数、总成绩

SELECT

stu.student\_id '学号',

stu.student\_name '姓名',

COUNT(sc.subject\_id) '选课数',

SUM(sc.score)

FROM

student stu

LEFT JOIN score sc ON stu.student\_id = sc.student\_id

GROUP BY

stu.student\_id;



## #12.查询姓“李”的老师的个数

SELECT

COUNT(teacher\_name) '姓李的老师个数'

FROM

teacher

WHERE

teacher\_name LIKE '李%'



## #13.查询没学过“张磊老师”课的同学的学号、姓名

SELECT

stu.student\_id,

stu.student\_name ,

te.teacher\_name '教过这个学生的老师'

FROM

student stu

LEFT JOIN score sc ON stu.student\_id = sc.student\_id

LEFT JOIN subject sub ON sc.subject\_id = sub.subject\_id

LEFT JOIN teacher te ON sub.teacher\_id = te.teacher\_id

GROUP BY

stu.student\_id

HAVING

te.teacher\_name NOT IN ( '张磊老师' )

OR te.teacher\_name IS NULL;

/\*

执行子查询时，MYSQL需要创建临时表，查询完毕后再删除这些临时表，所以，子查询的速度会受到一定的影响，这里多了一个创建和销毁临时表的过程。

优化方式：

可以使用连接查询（JOIN）代替子查询，连接查询不需要建立临时表，因此其速度比子查询快。在连接查询中左右连接效率高于内连接

\*/



## #14.查询学过“1”并且也学过编号“2”课程的同学的学号、姓名

#方法1

SELECT

sca.student\_id '学号',

stu.student\_name '姓名'

FROM

score sca

LEFT JOIN score scb ON sca.subject\_id = 1

AND scb.subject\_id = 2

AND sca.student\_id = scb.student\_id

INNER JOIN student stu ON stu.student\_id = sca.student\_id

AND stu.student\_id = scb.student\_id;

#方法2

SELECT

sca.student\_id '学号',

stu.student\_name '姓名'

FROM

student stu

INNER JOIN score sca ON sca.subject\_id = 1

AND stu.student\_id = sca.student\_id

INNER JOIN score scb ON scb.subject\_id = 2

AND stu.student\_id = scb.student\_id;



## #15.查询学过“李平老师”所教的所有课的同学的学号、姓名

SELECT

ta.id '学号',

stu.student\_name '姓名'

FROM

(

SELECT DISTINCT

sc.student\_id id,

COUNT( sc.subject\_id ) cnt\_class

FROM

teacher te

LEFT JOIN SUBJECT sub ON te.teacher\_id = sub.teacher\_id

AND te.teacher\_name = '李平老师'

LEFT JOIN score sc ON sc.subject\_id = sub.subject\_id

GROUP BY

sc.student\_id

) ta

INNER JOIN student stu ON ta.id = stu.student\_id

AND ta.cnt\_class = (

SELECT

COUNT( sub.teacher\_id )

FROM

teacher te

LEFT JOIN SUBJECT sub ON te.teacher\_id = sub.teacher\_id

AND te.teacher\_name = '李平老师'

);

/\*

首先第一个线索要获得李平老师id，之后

通过id获得所交的科目的编号，再者获得科目编号对应的学生，会有重复要去重

\*/



## #16.查询没有学全所有课的同学的学号、姓名

SELECT

sc.student\_id '学号',

stu.student\_name,

COUNT( sc.subject\_id ) '所学的课程数'

FROM

score sc,

student stu

WHERE

stu.student\_id = sc.student\_id

GROUP BY

sc.student\_id

HAVING

COUNT( sc.subject\_id ) < ( SELECT COUNT( \* ) FROM `subject` );



## #17.查询和“003”号的同学学习的课程完全相同的其他同学学号和姓名

#方法1 正面刚

SELECT DISTINCT

ta.stu\_id '学号',

ta.stu\_name '姓名'

FROM

(

SELECT

sc.student\_id stu\_id,

stu.student\_name stu\_name,

count( \* ) cnt\_sub

FROM

score sc,

student stu

WHERE

sc.student\_id != 3

AND subject\_id IN ( SELECT subject\_id FROM score WHERE student\_id = 3 )

AND stu.student\_id = sc.student\_id

GROUP BY

sc.student\_id

) ta,

( SELECT student\_id stu\_id, count( \* ) cnt\_sub FROM score GROUP BY student\_id ) tb,

(SELECT COUNT( \* ) cnt\_x FROM score WHERE student\_id = 3) tc

WHERE

ta.cnt\_sub = tb.cnt\_sub

AND ta.cnt\_sub = tc.cnt\_x

AND ta.stu\_id = tb.stu\_id;

/\*

1-处理出一张所有同学学习与3号同学相同的科数的表，排除3号同学

2-处理一个所有同学学习的科目的总数

3-得到3号同学学习科目数

然后判断完全相同的：

条件1表的科目数=2表的科目数 表示就是全部科目，且只含有2号同学学的科目

条件1表的科目数=3表的科目数

表1的学生id = 表2学生id

\*/

#方法2 逆向思维

SELECT

sc.student\_id '学号',

stu.student\_name '姓名'

FROM

score sc,

student stu

WHERE

sc.student\_id NOT IN ( SELECT student\_id FROM score WHERE subject\_id NOT IN ( SELECT subject\_id FROM score WHERE student\_id = 3 ) )

AND sc.student\_id != 3

AND sc.student\_id = stu.student\_id

GROUP BY

sc.student\_id

HAVING

count( sc.subject\_id ) = ( SELECT count( subject\_id ) FROM score WHERE student\_id = 3 );/\*

双重否定，找出不学3号同学学过的科目的同学，然后再排除这些同学，

接下来就剩下一定学过3号同学学的科目，再从中找出学习的科目数与

3号学习的科目数一样

\*/



## #18.删除学习“叶平”老师课的SC表记录；

INSERT INTO `Subject`

VALUES

( '5', '数据库', '6' );

INSERT INTO `teacher`

VALUES

( '6', '叶平老师' );

SELECT

sc.score\_id

FROM

score sc

INNER JOIN SUBJECT sub ON sc.subject\_id = sub.subject\_id

INNER JOIN teacher te ON te.teacher\_id = sub.teacher\_id

AND te.teacher\_name = '叶平老师';

#方法1

CREATE TABLE tmp AS SELECT

sc.score\_id

FROM

score sc

INNER JOIN SUBJECT sub ON sc.subject\_id = sub.subject\_id

INNER JOIN teacher te ON te.teacher\_id = sub.teacher\_id

AND te.teacher\_name = '叶平老师';

#方法2

CREATE TABLE tmp AS SELECT

sc.score\_id

FROM

score sc

INNER JOIN SUBJECT sub ON sc.subject\_id = sub.subject\_id

INNER JOIN teacher te ON te.teacher\_id = sub.teacher\_id

AND te.teacher\_name = '叶平老师';

DELETE

FROM

score

WHERE

score\_id IN (

SELECT \* FROM tmp

);

DROP TABLE tmp;

/\*

不能先将select出表中的某些值，再update这个表(在同一语句中)

所以先建立一个临时表，删除相应数据后删除表，或者直接嵌套一层，使其成为临时表

\*/

## #19.向SC表中插入一些记录，这些记录要求符合以下条件：

#①没有上过编号“002”课程的同学学号；

#②插入“002”号课程的平均成绩

INSERT INTO score ( student\_id, subject\_id, score ) SELECT

stu.student\_id,

2,

( SELECT AVG( score ) FROM score WHERE subject\_id = 2 )

FROM

student stu

WHERE

stu.student\_id IN ( SELECT student\_id FROM score WHERE subject\_id != 2 );

## #20.按平均成绩从低到高显示所有学生的“生物”、“物理”、“体育”三门的课程成绩，按如下形式显示： 学生ID,生物,物理,体育,有效课程数,有效平均分

#方法1

SELECT

stu.student\_id,

a.score biological,

b.score physical,

c.score sports,

d.score arts,

e.count\_subject number\_class,

e.avg\_score avg\_grade

FROM

student stu

LEFT JOIN ( SELECT student\_id, score FROM score WHERE subject\_id = ( SELECT subject\_id FROM `subject` WHERE subject\_name = '生物' ) ) a ON stu.student\_id = a.student\_id

LEFT JOIN ( SELECT student\_id, score FROM score WHERE subject\_id = ( SELECT subject\_id FROM `subject` WHERE subject\_name = '物理' ) ) b ON stu.student\_id = b.student\_id

LEFT JOIN ( SELECT student\_id, score FROM score WHERE subject\_id = ( SELECT subject\_id FROM `subject` WHERE subject\_name = '体育' ) ) c ON stu.student\_id = c.student\_id

LEFT JOIN ( SELECT student\_id, score FROM score WHERE subject\_id = ( SELECT subject\_id FROM `subject` WHERE subject\_name = '美术' ) ) d ON stu.student\_id = d.student\_id

LEFT JOIN ( SELECT student\_id, avg( score ) avg\_score, count( score ) count\_subject FROM score GROUP BY student\_id ) e ON stu.student\_id = e.student\_id;

#方法2

SELECT

stu.student\_id,

SUM( ( CASE sc.subject\_id WHEN ( SELECT subject\_id FROM SUBJECT WHERE subject\_name = '生物' ) THEN sc.score ELSE 0 END ) #case语句形式1

) biological,

SUM(

( CASE WHEN sc.subject\_id IN ( SELECT subject\_id FROM SUBJECT WHERE subject\_name = '物理' ) THEN sc.score ELSE 0 END ) #case语句形式2

) physical,

SUM( ( CASE sc.subject\_id WHEN ( SELECT subject\_id FROM SUBJECT WHERE subject\_name = '体育' ) THEN sc.score ELSE 0 END ) ) sports,

SUM( ( CASE sc.subject\_id WHEN ( SELECT subject\_id FROM SUBJECT WHERE subject\_name = '美术' ) THEN sc.score ELSE 0 END ) ) arts,

tmp.avg\_grade,

tmp.count\_grade

FROM

student stu

LEFT JOIN (

SELECT

sca.student\_id,

AVG( sca.score ) avg\_grade,

COUNT( sca.score ) count\_grade

FROM

score sca

GROUP BY

sca.student\_id

) tmp ON stu.student\_id = tmp.student\_id,

score sc

WHERE

stu.student\_id = sc.student\_id

GROUP BY

stu.student\_id

ORDER BY

score DESC;

/\*

形式1

CASE 列名name

WHEN x1 THEN -- 当name=x1时,这一列取值y1

y1

WHEN x2 THEN -- 当name=x2时,这一列取值y2

y2

ELSE -- 否则取值y3

y3

END ;

形式2

CASE

WHEN name = x1 THEN -- 当name=x1时,这一列取值y1

y1

WHEN name = x2 THEN -- 当name=x2时,这一列取值y2

y2

WHEN name = x3 THEN -- 当name=x3时,这一列取值y3

y3

ELSE -- 否则取值y4

y4

END ;

\*/

#方法3

SELECT

stu.student\_id,

SUM( IF ( sc.subject\_id IN ( SELECT subject\_id FROM SUBJECT WHERE subject\_name = '生物' ), sc.score, 0 ) ) biological,

SUM( IF ( sc.subject\_id IN ( SELECT subject\_id FROM SUBJECT WHERE subject\_name = '物理' ), sc.score, 0 ) ) physical,

SUM( IF ( sc.subject\_id IN ( SELECT subject\_id FROM SUBJECT WHERE subject\_name = '体育' ), sc.score, 0 ) ) sports,

SUM( IF ( sc.subject\_id IN ( SELECT subject\_id FROM SUBJECT WHERE subject\_name = '美术' ), sc.score, 0 ) ) arts,

tmp.avg\_grade,

tmp.count\_grade

FROM

student stu

LEFT JOIN (

SELECT

sca.student\_id,

AVG( sca.score ) avg\_grade,

COUNT( sca.score ) count\_grade

FROM

score sca

GROUP BY

sca.student\_id

) tmp ON stu.student\_id = tmp.student\_id,

score sc

WHERE

stu.student\_id = sc.student\_id

GROUP BY

stu.student\_id

ORDER BY

score DESC;

/\* 方式1 示例

IF stu\_grade>=90 THEN

SELECT stu\_grade,'A';

ELSEIF stu\_grade<90 AND stu\_grade>=80 THEN

SELECT stu\_grade,'B';

ELSEIF stu\_grade<80 AND stu\_grade>=70 THEN

SELECT stu\_grade,'C';

ELSEIF stu\_grade70 AND stu\_grade>=60 THEN

SELECT stu\_grade,'D';

ELSE

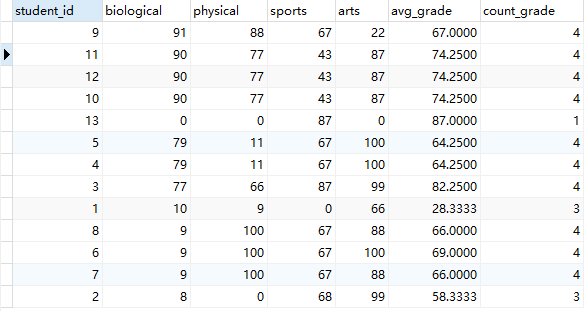
SELECT stu\_grade,'E';

END IF;

方式2

IF(clum\_name,x1,x2) clum\_name不为空且不等于0时，取值x1，否则x2

\*/



## #21.查询各科成绩最高和最低的分：以如下形式显示：课程ID，最高分，最低分

SELECT

subject\_id '课程ID',

MAX(score) '最高分',

MIN(score) '最低分'

FROM

score

GROUP BY

subject\_id;



## #22.按各科平均成绩从低到高和及格率的百分数从高到低顺序

/\*让外部的sc表的id一个一个id的去指定临时表内的数据id来源显示，相当于多次查询\*/

SELECT

sc.subject\_id,

AVG( score ) avgGrade,

( SELECT COUNT( \* ) FROM score sca WHERE sca.subject\_id = sc.subject\_id AND sca.score >= 60 ) / ( SELECT COUNT( \* ) FROM score sca WHERE sca.subject\_id = sc.subject\_id ) pass

FROM

score sc

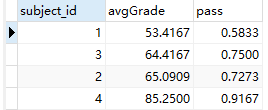
GROUP BY

sc.subject\_id

ORDER BY

avgGrade,

pass DESC;



## #23.查询各科成绩前三名的记录:(不考虑成绩并列情况)

SELECT

sub.subject\_id,

sub.subject\_name,

(SELECT score FROM score sc WHERE sc.subject\_id = sub.subject\_id GROUP BY score ORDER BY score DESC LIMIT 0,1) 'one',

(SELECT score FROM score sc WHERE sc.subject\_id = sub.subject\_id GROUP BY score ORDER BY score DESC LIMIT 1,1) 'two',

(SELECT score FROM score sc WHERE sc.subject\_id = sub.subject\_id GROUP BY score ORDER BY score DESC LIMIT 2,1) 'three'

FROM

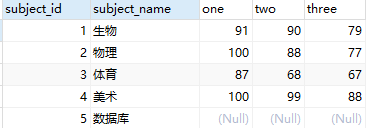
subject sub;

/\*

LIMIT x,y 意思是y条数据，索引从x到y+x-1，第x+1条记录到第x+y+1条记录

SELECT score FROM score sc WHERE sc.subject\_id GROUP BY score ORDER BY score DESC LIMIT 1,1

\*/



## #24.查询每门课程被选修的学生数

SELECT

subject\_id '课程ID',

COUNT(student\_id) '选修人数'

FROM

score

GROUP BY

subject\_id;



## #25.查询同名同姓学生名单，并统计同名人数

INSERT INTO `student`

VALUES

( '17', '男', '4', '理解' ),

( '18', '女', '4', '钢蛋' ),

( '19', '男', '4', '张三' ),

( '20', '男', '5', '张二' ),

( '21', '女', '5', '张二' );

SELECT

ta.student\_name '同名同姓的姓名',

COUNT( ta.student\_name ) '同名个数'

FROM

(

SELECT DISTINCT

sta.student\_id,

sta.student\_name

FROM

student sta

INNER JOIN student stb ON sta.student\_name = stb.student\_name

AND sta.student\_id != stb.student\_id

) ta

GROUP BY

ta.student\_name;



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

SELECT

AVG(IF(ISNULL(score),0,score) ) avgGrade,

subject\_id

FROM

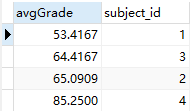
score

GROUP BY

subject\_id

ORDER BY

avgGrade, subject\_id DESC;



## #27.查询平均成绩大于70的所有学生的学号. 姓名和平均成绩

SELECT

stu.student\_id id,

stu.student\_name name,

AVG(sc.score) avgGrade

FROM

score sc

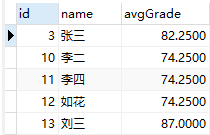
LEFT JOIN student stu ON sc.student\_id = stu.student\_id

GROUP BY

sc.student\_id

HAVING

avgGrade > 70;



## #28.查询课程名称为“物理”，且分数低于60的学生姓名和分数

SELECT

stu.student\_id id,

stu.student\_name name,

sc.score grade

FROM

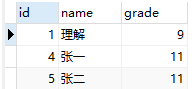
subject sub

INNER JOIN score sc ON sub.subject\_name = '物理'

AND sub.subject\_id = sc.subject\_id

AND sc.score < 60

LEFT JOIN student stu ON sc.student\_id = stu.student\_id;



## #29.查询课程编号为003且课程成绩在80分以上的学生的学号和姓名

SELECT

stu.student\_id id,

stu.student\_name name,

sc.subject\_id,

sc.score

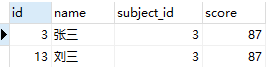
FROM

student stu

INNER JOIN score sc ON sc.subject\_id = 3

AND sc.score > 80

AND stu.student\_id=sc.student\_id;



## #30.求选了课程的学生人数

SELECT

COUNT( \* ) '选了课的学生人数'

FROM

( SELECT student\_id FROM score GROUP BY student\_id ) ta;



## #31.查询选修“李平”老师所授课程的学生中，成绩最高的学生姓名及其成绩；

SELECT

stu.student\_id,

stu.student\_name,

a.subject\_id,

a.maxGrade

FROM

(

SELECT

sc.subject\_id,

MAX( sc.score ) maxGrade

FROM

teacher te

INNER JOIN SUBJECT sub ON te.teacher\_name = '李平老师'

AND te.teacher\_id = sub.teacher\_id

INNER JOIN score sc ON sub.subject\_id = sc.subject\_id

GROUP BY

sc.subject\_id

) a

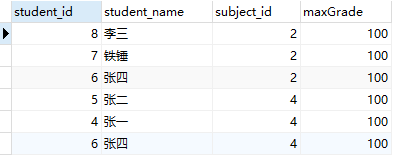
INNER JOIN score sca ON a.subject\_id = sca.subject\_id

AND a.maxGrade = sca.score

INNER JOIN student stu ON stu.student\_id = sca.student\_id

ORDER BY

subject\_id;



## #32.查询各个课程及相应的选修人数；

SELECT

sub.subject\_name,

COUNT(sc.student\_id) sum

FROM

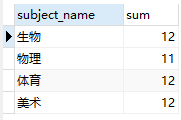
score sc

INNER JOIN

subject sub ON sc.subject\_id = sub.subject\_id

GROUP BY

sc.subject\_id;



## #33.查询不同课程但成绩相同的学生的学号、课程号、学生成绩；

SELECT

sca.student\_id,

sca.subject\_id,

sca.score

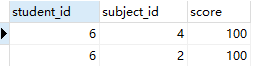
FROM

score sca

INNER JOIN score scb ON sca.subject\_id != scb.subject\_id

AND sca.score = scb.score

AND sca.student\_id = scb.student\_id;



## #34.查询每门课程成绩最好的前两名；

#方法1

SELECT

sc.student\_id,

sc.subject\_id,

sc.score

FROM

score sc

LEFT JOIN (

SELECT

score\_id,

subject\_id,

( SELECT score FROM score sca WHERE sca.subject\_id = scd.subject\_id ORDER BY score DESC LIMIT 0, 1 ) first\_num,

( SELECT score FROM score sca WHERE sca.subject\_id = scd.subject\_id ORDER BY score DESC LIMIT 1, 1 ) second\_num

FROM

score scd

) ta ON sc.score\_id = ta.score\_id

WHERE

sc.score = ta.first\_num

OR sc.score = ta.second\_num

ORDER BY

sc.subject\_id;

#方法2

SELECT

subject\_id,

score

FROM

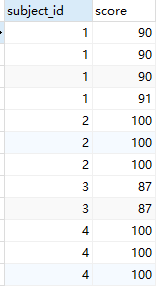
score sc1

WHERE

( SELECT COUNT( \* ) FROM score sc2 WHERE sc1.score < sc2.score AND sc1.subject\_id = sc2.subject\_id ) < 2

ORDER BY

subject\_id;



## #35.检索至少选修两门课程的学生学号；

SELECT

student\_id,

COUNT(\*) count\_subject

FROM

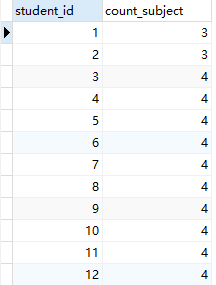
score

GROUP BY

student\_id

HAVING

count\_subject > 1;



## #36.查询全部学生都选修的课程的课程号和课程名；

SELECT

sc.subject\_id,

sub.subject\_name

FROM

score sc,

`subject` sub

WHERE

sub.subject\_id = sc.subject\_id

GROUP BY

sc.subject\_id

HAVING

COUNT( sc.subject\_id ) IN ( SELECT COUNT( \* ) FROM student );



## #37.查询没学过“李平”老师讲授的任一门课程的学生姓名；

SELECT

student\_name,

student\_id

FROM

student

WHERE

student\_id NOT IN (

SELECT DISTINCT

sc.student\_id

FROM

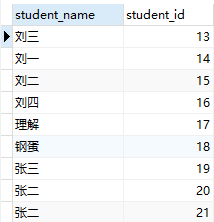
teacher te

INNER JOIN SUBJECT sub ON te.teacher\_name = '李平老师'

AND te.teacher\_id = sub.teacher\_id

INNER JOIN score sc ON sc.subject\_id = sub.subject\_id

);



## #38.查询两门或两门以上不及格课程的同学的学号及其平均成绩；

SELECT

sca.student\_id,

AVG( sca.score ) avgScore

FROM

score sca

INNER JOIN ( SELECT student\_id FROM score WHERE score < 60 GROUP BY student\_id HAVING COUNT( \* ) >= 2 ) tmp ON tmp.student\_id = sca.student\_id;



## #39.检索“004”课程分数小于60，按分数降序排列的同学学号；

SELECT

student\_id

FROM

score

WHERE

score < 60

AND subject\_id = 4

ORDER BY

student\_id;



## #40.删除“002”同学的“001”课程的成绩；

DELETE

FROM

score

WHERE

student\_id = 2

AND subject\_id = 1;