### 1、表结构

–1.学生表

student(s\_id,s\_name,s\_birth,s\_sex) –学生编号,学生姓名, 出生年月,学生性别

–2.课程表

course(c\_id,c\_name,t\_id) – –课程编号, 课程名称, 教师编号

–3.教师表

Teacher(t\_id,t\_name) –教师编号,教师姓名

–4.成绩表

Score(s\_id,c\_id,s\_score) –学生编号,课程编号,分数

### 2、测试数据

--建表

--学生表

CREATE TABLE `student`(

`s\_id` VARCHAR(20),

`s\_name` VARCHAR(20) NOT NULL DEFAULT '',

`s\_birth` VARCHAR(20) NOT NULL DEFAULT '',

`s\_sex` VARCHAR(10) NOT NULL DEFAULT '',

PRIMARY KEY(`s\_id`)

) ENGINE=InnoDB DEFAULT CHARSET=utf8;

--课程表

CREATE TABLE `course`(

`c\_id` VARCHAR(20),

`c\_name` VARCHAR(20) NOT NULL DEFAULT '',

`t\_id` VARCHAR(20) NOT NULL,

PRIMARY KEY(`c\_id`)

) ENGINE=InnoDB DEFAULT CHARSET=utf8;

--教师表

CREATE TABLE `teacher`(

`t\_id` VARCHAR(20),

`t\_name` VARCHAR(20) NOT NULL DEFAULT '',

PRIMARY KEY(`t\_id`)

) ENGINE=InnoDB DEFAULT CHARSET=utf8;

--成绩表

CREATE TABLE `score`(

`s\_id` VARCHAR(20),

`c\_id` VARCHAR(20),

`s\_score` INT(3),

PRIMARY KEY(`s\_id`,`c\_id`)

) ENGINE=InnoDB DEFAULT CHARSET=utf8;

--插入学生表测试数据

insert into student values('01' , '赵雷' , '1990-01-01' , '男');

insert into student values('02' , '钱电' , '1990-12-21' , '男');

insert into student values('03' , '孙风' , '1990-05-20' , '男');

insert into student values('04' , '李云' , '1990-08-06' , '男');

insert into student values('05' , '周梅' , '1991-12-01' , '女');

insert into student values('06' , '吴兰' , '1992-03-01' , '女');

insert into student values('07' , '郑竹' , '1989-07-01' , '女');

insert into student values('08' , '王菊' , '1990-01-20' , '女');

--课程表测试数据

insert into course values('01' , '语文' , '02');

insert into course values('02' , '数学' , '01');

insert into course values('03' , '英语' , '03');

--教师表测试数据

insert into teacher values('01' , '张三');

insert into teacher values('02' , '李四');

insert into teacher values('03' , '王五');

--成绩表测试数据

insert into score values('01' , '01' , 80);

insert into score values('01' , '02' , 90);

insert into score values('01' , '03' , 99);

insert into score values('02' , '01' , 70);

insert into score values('02' , '02' , 60);

insert into score values('02' , '03' , 80);

insert into score values('03' , '01' , 80);

insert into score values('03' , '02' , 80);

insert into score values('03' , '03' , 80);

insert into score values('04' , '01' , 50);

insert into score values('04' , '02' , 30);

insert into score values('04' , '03' , 20);

insert into score values('05' , '01' , 76);

insert into score values('05' , '02' , 87);

insert into score values('06' , '01' , 31);

insert into score values('06' , '03' , 34);

insert into score values('07' , '02' , 89);

insert into score values('07' , '03' , 98);

### 3、练习题

#### -- 1、查询"01"课程比"02"课程成绩高的学生的信息及课程分数

SQL:

SELECT s.\* , s1.s\_score 01\_score ,s2.s\_score 02\_score

FROM student s , score s1, score s2

WHERE

s.s\_id = s1.s\_id AND

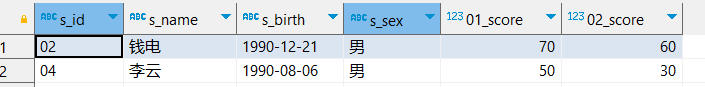
s.s\_id = s2.s\_id AND

s1.c\_id = 01 AND

s2.c\_id = 02 AND

s1.s\_score > s2.s\_score

结果：



#### -- 2、查询"01"课程比"02"课程成绩低的学生的信息及课程分数

SQL:

SELECT s.\* , s1.s\_score 01\_score ,s2.s\_score 02\_score

FROM student s , score s1, score s2

WHERE

s.s\_id = s1.s\_id AND

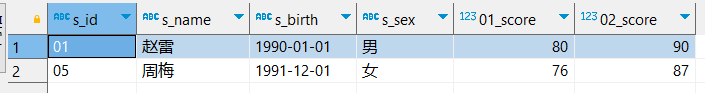
s.s\_id = s2.s\_id AND

s1.c\_id = 01 AND

s2.c\_id = 02 AND

s1.s\_score < s2.s\_score

结果：



#### -- 3、查询平均成绩大于等于60分的同学的学生编号和学生姓名和平均成绩

SQL:

SELECT s.s\_id , s.s\_name , AVG(sc.s\_score)

FROM student s , score sc

WHERE s.s\_id = sc.s\_id

GROUP BY sc.s\_id

HAVING AVG(sc.s\_score) > 60

结果：



#### -- 4、查询平均成绩小于60分的同学的学生编号和学生姓名和平均成绩

-- (包括有成绩的和无成绩的)

SQL:

SELECT s.s\_id , s.s\_name ,

CASE

WHEN AVG(sc.s\_score) = NULL THEN

0

ELSE

AVG(sc.s\_score)

END AS avgscore

FROM student s

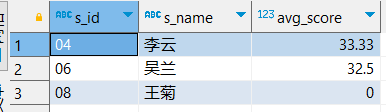
LEFT JOIN score sc

ON s.s\_id = sc.s\_id

GROUP BY sc.s\_id

HAVING AVG(sc.s\_score) < 60 OR AVG(sc.s\_score) IS NULL

结果：



#### -- 5、查询所有同学的学生编号、学生姓名、选课总数、所有课程的总成绩

SQL:

SELECT s.s\_id , s.s\_name , COUNT(sc.c\_id),

CASE

WHEN SUM(sc.s\_score) = NULL THEN

0

ELSE

SUM(sc.s\_score)

END AS SUMscore

FROM student s

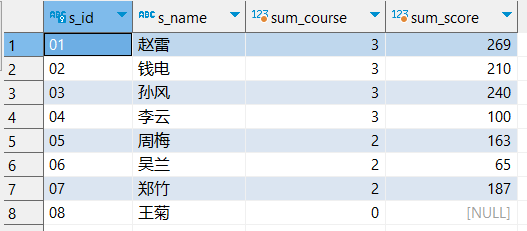
LEFT JOIN score sc

ON s.s\_id = sc.s\_id

GROUP BY sc.s\_id

HAVING SUM(sc.s\_score) OR SUM(sc.s\_score) IS NULL

结果：



#### -- 6、查询学过"张三"老师授课的同学的信息

SQL:

SELECT s.\*

FROM student s,course c,score sc,teacher t

WHERE

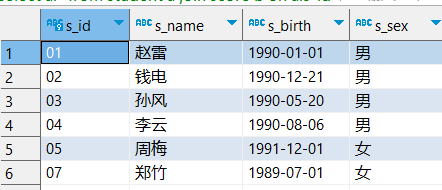
s.s\_id = sc.s\_id AND

sc.c\_id = c.c\_id AND

c.t\_id = t.t\_id AND

t.t\_name = '张三'

结果：



#### -- 7、查询学过编号为"01"并且也学过编号为"02"的课程的同学的信息

SQL:

SELECT s.\*

FROM student s , score sc1 , score sc2

WHERE

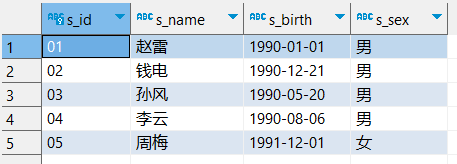
s.s\_id = sc1.s\_id AND

s.s\_id = sc2.s\_id AND

sc1.c\_id = '01' AND

sc2.c\_id = '02'

结果：



#### -- 8、查询没有学全所有课程的同学的信息

SQL:

SELECT s.\*

FROM student s

LEFT JOIN score sc1 ON s.s\_id = sc1.s\_id AND sc1.c\_id = 01

LEFT JOIN score sc2 ON s.s\_id = sc2.s\_id AND sc2.c\_id = 02

LEFT JOIN score sc3 ON s.s\_id = sc3.s\_id AND sc3.c\_id = 03

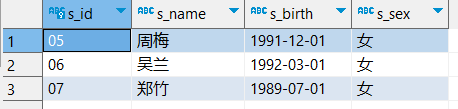
WHERE

sc1.c\_id IS NULL OR

sc2.c\_id IS NULL OR

sc3.c\_id IS NULL

结果：



#### -- 9、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

SQL:

SELECT s.s\_id , s.s\_name , AVG(sc.s\_score)

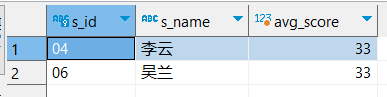
FROM student s INNER JOIN score sc on s.s\_id = sc.s\_id

WHERE sc.s\_score < 60

GROUP BY sc.s\_id

HAVING COUNT(\*) > 1

结果：



#### -- 10、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

SQL:

SELECT s.s\_id , sc1.s\_score AS '语文', sc2.s\_score AS '数学', sc3.s\_score AS '英语', AVG(sc.s\_score)

FROM student s

INNER JOIN score sc on s.s\_id = sc.s\_id

LEFT JOIN score sc1 ON s.s\_id = sc1.s\_id AND sc1.c\_id = 01

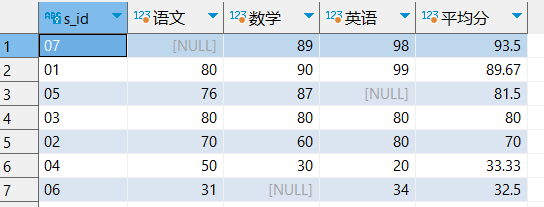
LEFT JOIN score sc2 ON s.s\_id = sc2.s\_id AND sc2.c\_id = 02

LEFT JOIN score sc3 ON s.s\_id = sc3.s\_id AND sc3.c\_id = 03

GROUP BY sc.s\_id

ORDER BY AVG(sc.s\_score) DESC

结果：



#### -- 11、查询各科成绩最高分、最低分和平均分：以如下形式显示：课程ID，课程name，最高分，最低分，平均分，及格率，中等率，优良率，优秀率

--及格为>=60，中等为：70-80，优良为：80-90，优秀为：>=90

SQL:

SELECT c.c\_id,c.c\_name,MAX(sc.s\_score),MIN(sc.s\_score),ROUND(AVG(sc.s\_score),2) AS '平均分',

ROUND(100 \* SUM(CASE WHEN sc.s\_score >= 60 THEN 1 ELSE 0 END ) / SUM(CASE WHEN sc.s\_score THEN 1 ELSE 0 END ),2)AS '及格率',

ROUND(100 \* SUM(CASE WHEN sc.s\_score >= 70 AND sc.s\_score < 80 THEN 1 ELSE 0 END ) / SUM(CASE WHEN sc.s\_score THEN 1 ELSE 0 END ),2)AS '中等率',

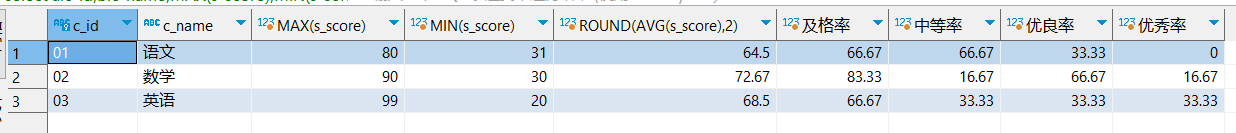
ROUND(100 \* SUM(CASE WHEN sc.s\_score >= 80 AND sc.s\_score < 90 THEN 1 ELSE 0 END ) / SUM(CASE WHEN sc.s\_score THEN 1 ELSE 0 END ),2)AS '优良率',

ROUND(100 \* SUM(CASE WHEN sc.s\_score >= 90 THEN 1 ELSE 0 END ) / SUM(CASE WHEN sc.s\_score THEN 1 ELSE 0 END ),2)AS '优秀率'

FROM course c INNER JOIN score sc ON c.c\_id = sc.c\_id

GROUP BY sc.c\_id

结果：



#### -- 12、查询学生的总成绩并进行排名

SQL:

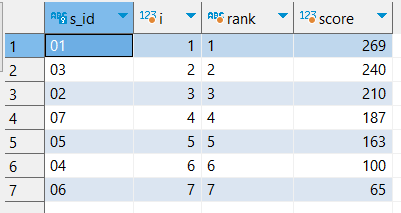
SELECT s\_id,SUM(s\_score),RANK() over (ORDER BY SUM(s\_score) DESC) AS 'rank',ROW\_NUMBER() over (ORDER BY SUM(s\_score) DESC) AS '123'

FROM score

GROUP BY s\_id

ORDER BY SUM(s\_score) DESC

结果：



#### -- 13、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

SQL:

SELECT s.\*,sc.排名,sc.s\_score,sc.c\_id

FROM student s INNER JOIN

(SELECT s\_id,s\_score,c\_id,ROW\_NUMBER() over (PARTITION BY c\_id ORDER BY s\_score DESC) AS 排名 FROM score

) sc

ON s.s\_id = sc.s\_id

WHERE 排名 IN(2,3)

结果：

score



#### -- 14、统计各科成绩各分数段人数：课程编号,课程名称,[100-85],[85-70],[70-60],[0-60]及所占百分比

SQL:

SELECT c.c\_name,c.c\_id,

SUM(CASE WHEN sc.s\_score > 85 AND sc.s\_score <= 100 THEN 1 ELSE 0 END) AS '85-100',

ROUND(100 \* SUM(CASE WHEN sc.s\_score > 85 AND sc.s\_score <= 100 THEN 1 ELSE 0 END) / SUM(CASE WHEN sc.s\_score THEN 1 ELSE 0 END),2) AS '百分比',

SUM(CASE WHEN sc.s\_score > 70 AND sc.s\_score <= 85 THEN 1 ELSE 0 END) AS '70-85',

ROUND(100 \* SUM(CASE WHEN sc.s\_score > 70 AND sc.s\_score <= 85 THEN 1 ELSE 0 END) / SUM(CASE WHEN sc.s\_score THEN 1 ELSE 0 END),2) AS '百分比',

SUM(CASE WHEN sc.s\_score > 60 AND sc.s\_score <= 70 THEN 1 ELSE 0 END) AS '60-70',

ROUND(100 \* SUM(CASE WHEN sc.s\_score > 60 AND sc.s\_score <= 70 THEN 1 ELSE 0 END) / SUM(CASE WHEN sc.s\_score THEN 1 ELSE 0 END),2) AS '百分比',

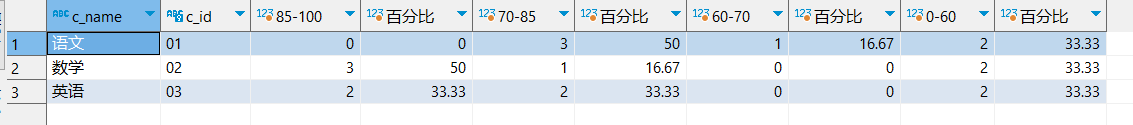
SUM(CASE WHEN sc.s\_score > 0 AND sc.s\_score <= 60 THEN 1 ELSE 0 END) AS '0-60',

ROUND(100 \* SUM(CASE WHEN sc.s\_score > 0 AND sc.s\_score <= 60 THEN 1 ELSE 0 END) / SUM(CASE WHEN sc.s\_score THEN 1 ELSE 0 END),2) AS '百分比'

FROM course c LEFT JOIN score sc ON c.c\_id = sc.c\_id

GROUP BY sc.c\_id

结果：



#### -- 15、查询各科成绩前三名的记录

SQL:

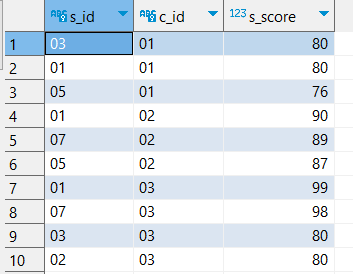
SELECT \*

FROM score s1

WHERE (SELECT COUNT(1) FROM score s2 WHERE s1.c\_id = s2.c\_id AND s1.s\_score<s2.s\_score) < 3

ORDER BY s1.c\_id,s1.s\_score DESC

结果：



#### -- 16、查询选修"张三"老师所授课程的学生中，成绩最高的学生信息及其成绩

SQL:

SELECT s.\*,sc.s\_score,sc.c\_id,c.c\_name

FROM student s,score sc,course c,teacher t

WHERE s.s\_id = sc.s\_id AND

sc.c\_id = c.c\_id AND

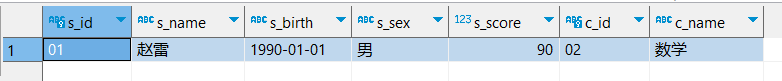
c.t\_id = t.t\_id AND

t.t\_name = '张三'

ORDER BY sc.s\_score DESC

LIMIT 1

结果：



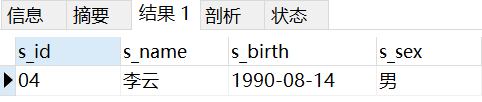
#### -- 17、查询本周过生日的学生

SQL:

SELECT \* FROM student

WHERE WEEK(CONCAT(YEAR(CURDATE()),"-" ,MID(s\_birth,6,5))) = WEEK(NOW())

结果：



#### -- 18、查询下个月过生日的学生

SQL:

SELECT \* FROM student

WHERE MONTH(CONCAT(YEAR(CURDATE()),"-",MID(s\_birth,6,5))) = MONTH(CURDATE())+1

结果：

