**eg:Query the information of students.**

select \* from student;

图形用户界面, 文本, 应用程序, 电子邮件

描述已自动生成

**Q1: Query the information and course scores of students whose grades in the "01" course are higher than those in the "02" course**

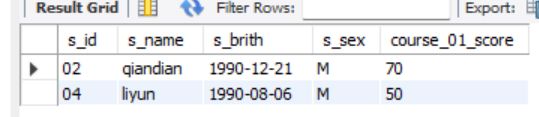
SELECT st.\*, sc1.s\_score AS course\_01\_score

FROM student st

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

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

WHERE sc1.s\_score > sc2.s\_score;



**Q2:** **Query the information and course scores of students whose grades in the "01" course are lower than those in the "02" course**

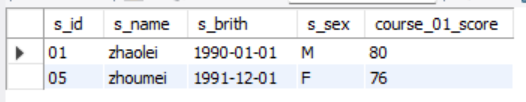
SELECT st.\*, sc1.s\_score AS course\_01\_score

FROM student st

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

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

WHERE sc1.s\_score < sc2.s\_score;



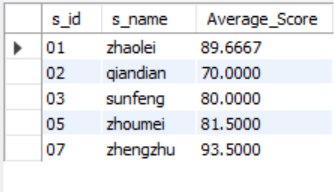
**Q3:** **Query the student number, student name and average score of students whose average score is greater than or equal to 60 points**

SELECT st.s\_id, st.s\_name, AVG(sc.s\_score) AS Average\_Score FROM student st

JOIN score sc ON st.s\_id = sc.s\_id

GROUP BY st.s\_id

HAVING Average\_Score >= 60;



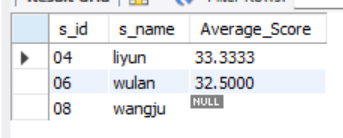
**Q4: Query the student number, name and average score of students whose average score is less than 60 points (including those with and without scores)**

SELECT st.s\_id, st.s\_name, AVG(sc.s\_score) AS Average\_Score FROM student st

LEFT JOIN score sc ON st.s\_id = sc.s\_id

GROUP BY st.s\_id

HAVING Average\_Score IS NULL OR Average\_Score < 60;



**Q5: Query the student number, student name, total number of courses selected, and total grades of all courses (including those without selected courses) of all students**

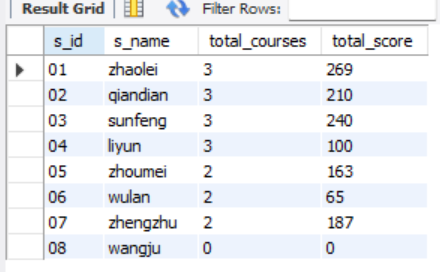
SELECT st.s\_id, st.s\_name, COUNT(DISTINCT sc.c\_id) AS total\_courses,

SUM(IFNULL(sc.s\_score, 0)) AS total\_score

FROM student st

LEFT JOIN score sc ON st.s\_id = sc.s\_id

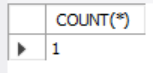
GROUP BY st.s\_id, st.s\_name;



**Q6: Query the number of teachers with the surname "li"**

SELECT COUNT(\*) FROM teacher

WHERE t\_name LIKE 'li%';



**Q7: Query the information of students who have studied with teacher "zhangsan"**

SELECT st.\*

FROM student st

WHERE st.s\_id IN (

SELECT DISTINCT sc.s\_id

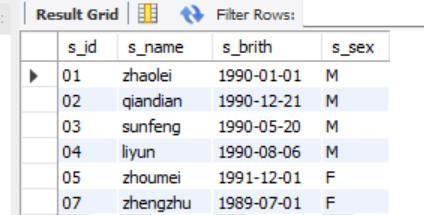
FROM score sc

JOIN course c ON sc.c\_id = c.c\_id

JOIN teacher t ON c.t\_id = t.t\_id

WHERE t.t\_name LIKE 'zhangsan'

);



**Q8: Query the information of students who have not studied under teacher "zhangsan"**

SELECT st.\*

FROM student st

WHERE st.s\_id NOT IN (

SELECT DISTINCT sc.s\_id

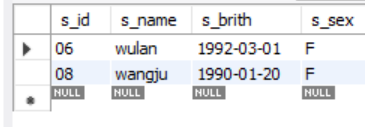
FROM score sc

JOIN course c ON sc.c\_id = c.c\_id

JOIN teacher t ON c.t\_id = t.t\_id

WHERE t.t\_name LIKE 'zhangsan'

);



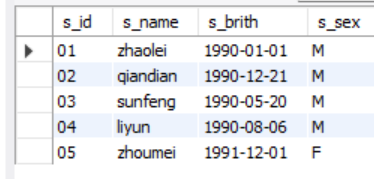
**Q9: Query the information of students who have studied the course numbered "01" and also studied the course numbered "02"**

SELECT st.\*

FROM student st

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

JOIN score sc2 ON st.s\_id = sc2.s\_id AND sc2.c\_id = '02';



**Q10: Query the information of students who have studied the course numbered "01" but not the course numbered "02"**

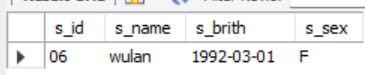
SELECT st.\*

FROM student st

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

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

WHERE sc2.s\_id IS NULL;

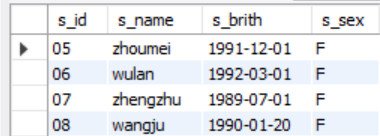


**Q11: Query information about students who have not completed all courses**

SELECT st.\* FROM student st

WHERE (SELECT COUNT(DISTINCT c\_id) FROM course) >

(SELECT COUNT(DISTINCT c\_id) FROM score WHERE s\_id = st.s\_id);



**Q12: Query the information of students who have at least one course that is the same as the student with student number "01"**

SELECT DISTINCT st.\*

FROM student st

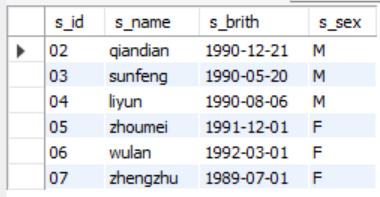
JOIN score sc ON st.s\_id = sc.s\_id

WHERE EXISTS (

SELECT 1 FROM score sc2

WHERE sc.c\_id = sc2.c\_id AND sc2.s\_id = '01'

) AND st.s\_id NOT LIKE '01';



**Q13: Query the information of other students who are studying the same course as the student numbered "01"**

SELECT DISTINCT st.\*

FROM student st

JOIN score sc ON st.s\_id = sc.s\_id

WHERE sc.c\_id IN (

SELECT c\_id FROM score WHERE s\_id = '01'

) AND st.s\_id NOT LIKE '01';



**Q14: Query the names of students who have not studied any course taught by "zhangsan"**

SELECT DISTINCT st.s\_name

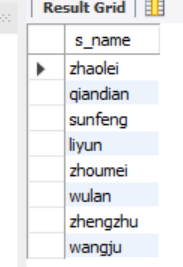
FROM student st

LEFT JOIN score sc ON st.s\_id = sc.s\_id

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

LEFT JOIN teacher t ON c.t\_id = t.t\_id AND t.t\_name = 'zhangsan'

WHERE t.t\_id IS NULL;



**Q15: Query the student ID number, name and average grade of students who failed two or more courses (Even if you did not take the exam, your score will be recorded as zero, so it should be considered a failure.)**

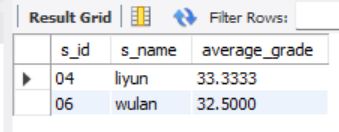
SELECT st.s\_id, st.s\_name, AVG(sc.s\_score) AS average\_grade FROM student st

JOIN score sc ON st.s\_id = sc.s\_id

WHERE sc.s\_score < 60

GROUP BY st.s\_id, st.s\_name

HAVING COUNT(\*) >= 2;



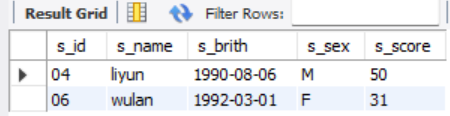
**Q16: Query the student information of "01" course whose score is less than 60 and sorted by score in descending order**

SELECT st.\*, sc.s\_score FROM student st

JOIN score sc ON st.s\_id = sc.s\_id

WHERE sc.c\_id = '01' AND sc.s\_score < 60

ORDER BY sc.s\_score DESC;



**Q17: Displays the grades for all students in all courses and the average grade by average grade from highest to lowest (Students who did not take the exam will have a score of 0)**

SELECT st.s\_id, st.s\_name, c.c\_id, c.c\_name, IFNULL(sc.s\_score, 0) AS score,

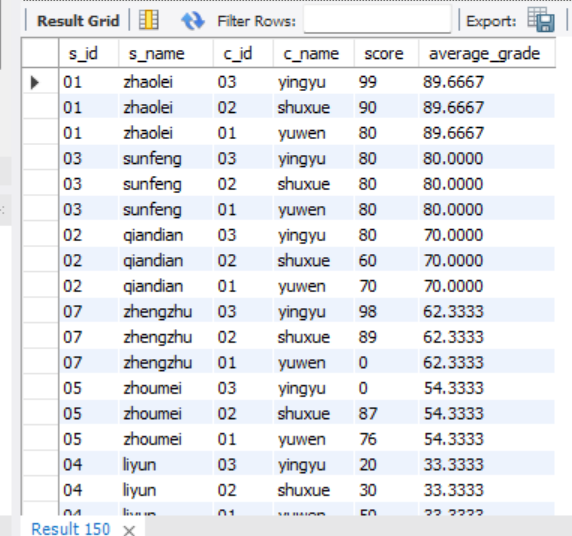
AVG(IFNULL(sc.s\_score, 0)) OVER (PARTITION BY st.s\_id) AS average\_grade

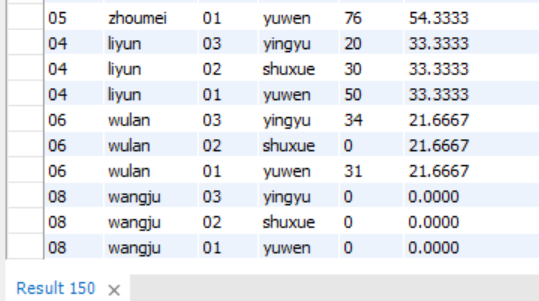
FROM student st

JOIN course c

LEFT JOIN score sc ON st.s\_id = sc.s\_id AND c.c\_id = sc.c\_id

ORDER BY average\_grade DESC, st.s\_id;





**Q18: Query the highest score, lowest score and average score of each subject: displayed in the following form: course ID, course name, highest score, lowest score, average score, passing rate, medium rate, good rate, excellent rate – passing is >=60 , medium is: 70-80, good is: 80-90, excellent is: >=90**

SELECT c.c\_id, c.c\_name, MAX(sc.s\_score) AS highest\_score, MIN(sc.s\_score) AS lowest\_score,

AVG(sc.s\_score) AS average\_score, COUNT(CASE WHEN sc.s\_score >= 60 THEN 1 END) / COUNT(\*) AS passing\_rate, COUNT(CASE WHEN sc.s\_score >= 70 AND sc.s\_score < 80 THEN 1 END) / COUNT(\*) AS medium\_rate,

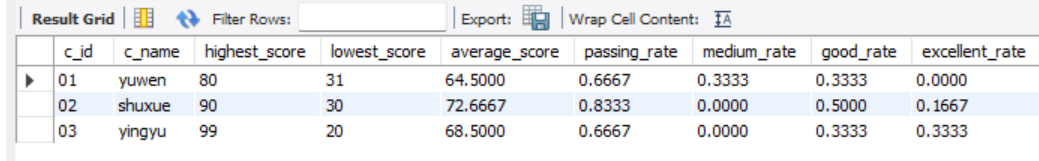
COUNT(CASE WHEN sc.s\_score >= 80 AND sc.s\_score < 90 THEN 1 END) / COUNT(\*) AS good\_rate, COUNT(CASE WHEN sc.s\_score >= 90 THEN 1 END) / COUNT(\*) AS excellent\_rate

FROM student st

JOIN score sc ON st.s\_id = sc.s\_id

JOIN course c ON sc.c\_id = c.c\_id

GROUP BY c.c\_id, c.c\_name;



**Q19: Query the student information, Sort by the scores of each subject and display the ranking**

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

(SELECT COUNT(\*) + 1 FROM score sc2

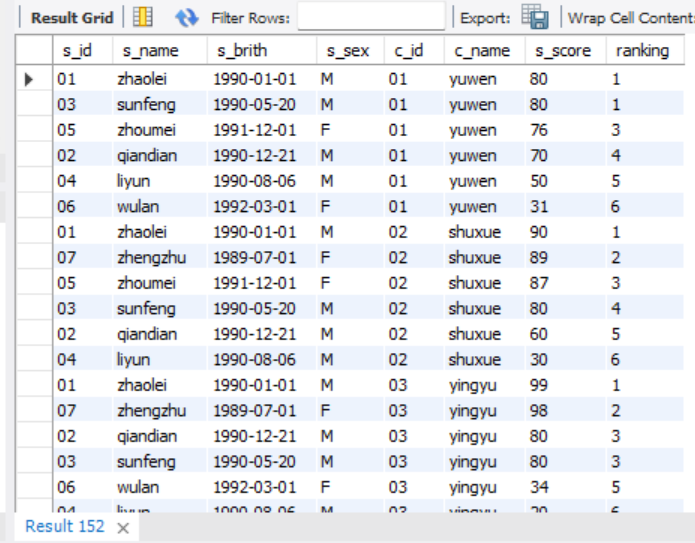
WHERE c.c\_id = sc2.c\_id AND sc2.s\_score > sc.s\_score) AS ranking

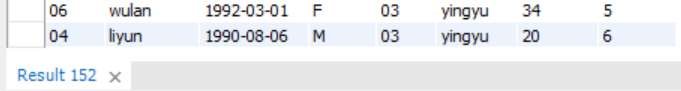
FROM student st

JOIN score sc ON st.s\_id = sc.s\_id

JOIN course c ON sc.c\_id = c.c\_id

ORDER BY c.c\_id, ranking;





**Q20: Query students’ total scores and rank them**

☹