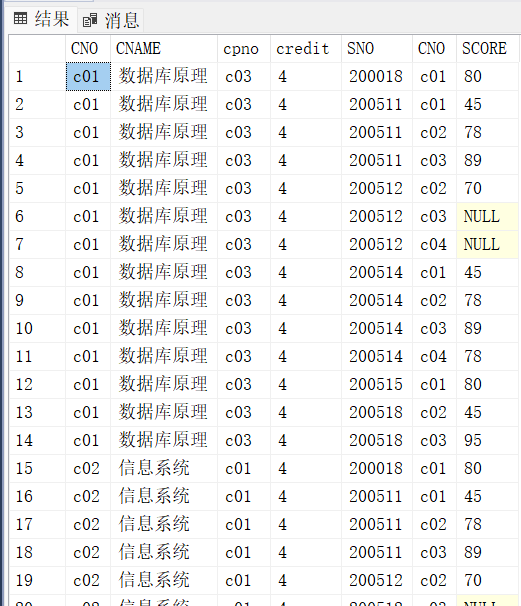
1) 多表的连接查询（相当于做笛卡儿乘积）

SELECT \* FROM C，SC



这里你可以往这两个表中追加大量数据，观察执行的效率

2) 表的等值连接查询

① 查询各学生的选课信息（包括学号、课程名、成绩）

因为学号和成绩在SC表中，而课程名在C表中，因此需要多表查询。

SELECT SNO，CNAME，SCORE

FROM C，SC

WHERE C.CNO=SC.CNO



② 查询学生的选课记录，显示学生的学号、姓名、课程号、成绩）

SELECT S.SNO，SNAME，CNO，SCORE

FROM S，SC

WHERE S.SNO=SC.SNO

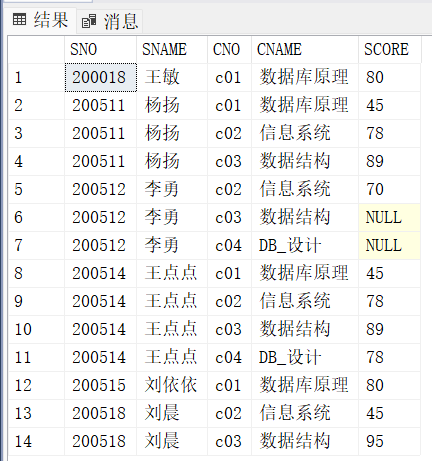


③ 查询学生的选课记录，显示学生的学号、姓名、课程号、课程名、成绩

SELECT S.SNO，SNAME，SC.CNO，CNAME，SCORE

FROM S，SC，C

WHERE S.SNO=SC.SNO AND SC.CNO=C.CNO



④ 左外连接查询：当希望左表（第一张表）中所有记录全部显示出来时，需要用

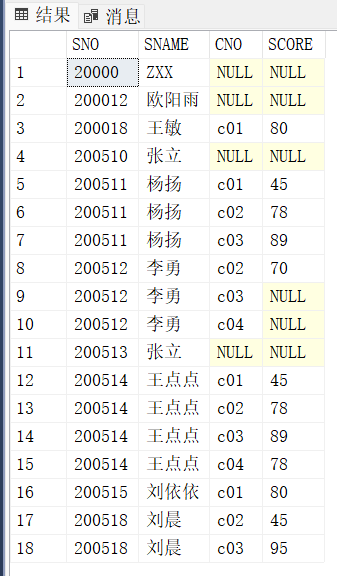
左外连接操作。

INSERT INTO S (SNO,SNAME) VALUES（’20000’,’ZXX’）

SELECT S.SNO，SNAME，CNO，SCORE

FROM S

LEFT OUTER JOIN SC ON S.SNO=SC.SNO



观察与下面等值连接的执行结果有何不同？

SELECT S.SNO，SNAME，CNO，SCORE FROM S, SC WHERE S.SNO=SC.SNO



本条查询语句并不会查询CNO为NULL的信息。

3) 表自身的连接

① 查询与’李勇’同系的学生学号

将学生表S与S本身进行等值连接（系部相等），因为S与S做连接操作时不能区

分，所以，对表取一个别名。然后将第二张表S中名字为’李勇’的记录选择出就可。

SELECT S1.SNO

FROM S AS S1， S AS S2

WHERE S1.DEPT=S2.DEPT AND S2.SNAME=‘李勇’



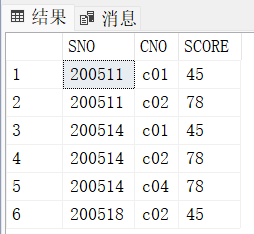
② 显示每个学生的非最高分成绩（学生自己的选课成绩中，不是最高分的选课记

录显示出来）

SELECT SNO, CNO, SCORE FROM SC AS SC1

WHERE SCORE <(SELECT MAX(SCORE) FROM SC AS SC2

WHERE SC2.SNO=SC1.SNO)



4) IN 嵌套子查询

① 不相关的IN 子查询：子查询可以单独执行，与被嵌套的查询无关。

如，查询与’李勇’同系的学生学号

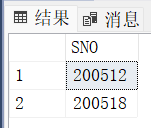
可以先查询出’李勇’所在的系，然后再到S表中查询与上述结果相同的记录。

SELECT SNO

FROM S

WHERE DEPT IN

(SELECT DEPT FROM S WHERE SNAME=‘李勇’）

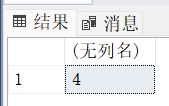


查询’数据库原理’课程的选课人数

SELECT COUNT (\*) FROM SC

WHERE CNO IN

（ SELECT CNO FROM C WHERE CNAME=‘数据库原理’）



② 相关的子查询：子查询中要用到父查询表的信息，子查询不能独立执行。

如，查询选修课程号为“C01”课程且成绩至少高于选修课程号为“C02”的同

学的Cno、Sno和SCORE。

在子查询中，因为要查找该同学’C02’课程的成绩，所以，需要父查询表中该学

生的学号信息。

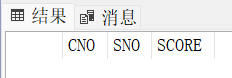
SELECT CNO，SNO，SCORE

FROM SC AS SC1

WHERE CNO=‘C01’ AND SCORE >(

SELECT SCORE FROM SC AS SC 2

WHERE SC2.SNO=SC1.SNO AND SC2.CNO=‘C02’)



查询没有结果

5) \* EXISTS 嵌套子查询

① 执行以下语句，观察显示的两个查询结果

SELECT SNO，SNAME

FROM S

WHERE EXISTS

（SELECT \* FROM SC WHERE CNO=‘C03’）

GO

SELECT SNO，SNAME FROM S



② 执行以下语句，观察显示的两个查询结果

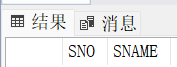
INSERT INTO C VALUES（’C06’，’数据库安全’，NULL，3）

SELECT SNO，SNAME

FROM S

WHERE EXISTS

（SELECT \* FROM SC WHERE CNO=‘C06’）



③ 查询’数据库原理’课程的选课人数

SELECT COUNT（\*）

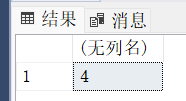
FROM SC

数据库实验指导书 22

WHERE EXISTS

（ SELECT \* FROM C

WHERE C.CNO=SC.CNO AND CNAME=‘数据库原理’)



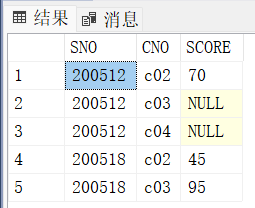
1. 查询“计算机系“学生所教课程的成绩表。

SELECT SC.SNO, SC.CNO, SC.SCORE

FROM SC

INNER JOIN S ON SC.SNO = S.SNO

WHERE S.DEPT = '计算机系';



上述查询语句使用 INNER JOIN 来联接 SC 表和 S 表，基于学生学号（SNO）进行连接。然后，使用 WHERE 子句筛选出系部（DEPT）为 "计算机系" 的学生。最后，选择学号（SNO）、课程号（CNO）和成绩（SCORE）列从 SC 表中检索数据。执行此查询后，将返回 "计算机系" 学生所教课程的成绩表。

1. 查询成绩比该课程平均成绩低的同学的成绩表。

SELECT \*

FROM SC

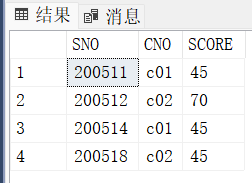
WHERE SCORE < (

SELECT AVG(SCORE)

FROM SC

WHERE CNO = SC.CNO

)



上述查询语句中的子查询计算了每门课程的平均成绩，然后在外部查询中选择成绩低于该课程平均成绩的记录。子查询使用了 WHERE 子句来确保只计算与外部查询中的记录相同的课程号（CNO）的平均成绩。执行此查询后，将返回成绩比本课程平均成绩低的同学的成绩表。

3) 查询选修 C01 课程的学生学号、课程名、成绩，并要求对查询结果按学号的降

序排列，如果学号相同则按成绩的升序排列。

SELECT S.SNO, C.CNAME, SC.SCORE

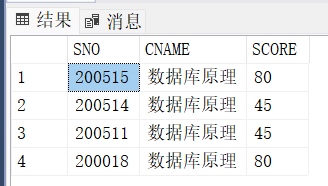
FROM S

INNER JOIN SC ON S.SNO = SC.SNO

INNER JOIN C ON SC.CNO = C.CNO

WHERE SC.CNO = 'C01'

ORDER BY S.SNO DESC, SC.SCORE ASC;



上述查询语句使用了多个表之间的内部连接（INNER JOIN）来获取学生表（S）、成绩表（SC）和课程表（C）之间的相关信息。通过 WHERE 子句筛选出课程号为 'C01' 的记录。然后，使用 ORDER BY 子句按照学号（SNO）的降序和成绩（SCORE）的升序排列结果。执行此查询后，将返回选修课程 'C01' 的学生学号、课程名和成绩，并按照指定的排序方式进行排列。

4) 查询选修两门及两门以上课程的学生学号及姓名。

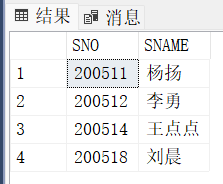
SELECT S.SNO, S.SNAME

FROM S

INNER JOIN SC ON S.SNO = SC.SNO

GROUP BY S.SNO, S.SNAME

HAVING COUNT(DISTINCT SC.CNO) >= 2;



上述查询语句使用了学生表（S）和成绩表（SC）之间的内部连接（INNER JOIN），并通过 GROUP BY 子句按学生的学号（SNO）和姓名（SNAME）进行分组。然后，通过 HAVING 子句筛选出选修了两门及两门以上课程的学生。HAVING COUNT(DISTINCT SC.CNO) >= 2 表示课程号（CNO）的数量不少于 2。执行此查询后，将返回选修了两门及两门以上课程的学生的学号和姓名。

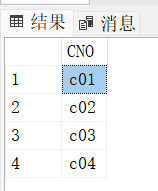
5) 查询年龄在 20-22 之间的男生的选修的课程号。

SELECT DISTINCT SC.CNO

FROM S

INNER JOIN SC ON S.SNO = SC.SNO

WHERE S.AGE BETWEEN 20 AND 22 AND S.SEX = '男';



上述查询语句使用了学生表（S）和成绩表（SC）之间的内部连接（INNER JOIN），并通过 WHERE 子句筛选出年龄在 20-22 之间且性别为男（SEX = '男'）的学生。然后，通过 SELECT DISTINCT 语句选择唯一的课程号（CNO）。执行此查询后，将返回年龄在 20-22 之间的男生选修的课程号列表。

6) 查询选修’数据结构’课程的学生人数。

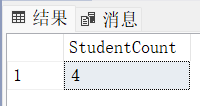
SELECT COUNT(DISTINCT S.SNO) AS StudentCount

FROM S

INNER JOIN SC ON S.SNO = SC.SNO

INNER JOIN C ON SC.CNO = C.CNO

WHERE C.CNAME = '数据结构';



上述查询语句使用了学生表（S）、成绩表（SC）和课程表（C）之间的内部连接（INNER JOIN）。首先，通过 INNER JOIN 将 S 表和 SC 表关联起来，然后将 SC 表与 C 表关联，以获取课程的信息。通过 WHERE 子句筛选出课程名为'数据结构'（C.CNAME = '数据结构'）的记录。最后，通过 COUNT(DISTINCT S.SNO) 对学生学号进行计数，得到选修了'数据结构'课程的学生人数。

7) 查询每门选课成绩在 80 分以上的学生学号、姓名。

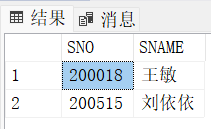
SELECT S.SNO, S.SNAME

FROM S

INNER JOIN SC ON S.SNO = SC.SNO

GROUP BY S.SNO, S.SNAME

HAVING MIN(SC.SCORE) >= 80;



上述查询语句使用了学生表（S）和成绩表（SC）之间的内部连接（INNER JOIN）。通过 GROUP BY 子句按学生的学号和姓名进行分组。然后，通过 HAVING 子句筛选出选修课程的最低成绩（MIN(SC.SCORE)）都在80分以上的学生。执行此查询后，将返回学号和姓名满足条件的学生信息。