

设教学数据库 Education 有三个关系:

学生关系 S (SNO, SNAME, AGE, SEX, SDEPT); 学习关系 SC (SNO, CNO, GRADE); 课程关系 C (CNO, CNAME, CDEPT, TNAME)

查询问题:

- (1) 检索计算机系的全体学生的学号, 姓名和性别;
- (2) 检索学习课程号为 C2 的学生学号与姓名;
- (3) 检索选修课程名为“DS”的学生学号与姓名;
- (4) 检索选修课程号为 C2 或 C4 的学生学号;
- (5) 检索至少选修课程号为 C2 和 C4 的学生学号;
- (6) 检索不学 C2 课的学生姓名和年龄;
- (7) 检索学习全部课程的学生姓名;
- (8) 查询所学课程包含学生 S3 所学课程的学生学号。

- (1) 检索计算机系的全体学生的学号, 姓名和性别;

```
SELECT Sno, Sname, Sex
FROM S
WHERE Sdept = 'CS';
```

- (2) 检索学习课程号为 C2 的学生学号与姓名;

1. SELECT Sno, Sname FROM S WHERE Sno IN (SELECT Sno FROM SC WHERE Cno= 'C2')	2. SELECT S.Sno, Sname FROM S, SC WHERE S.Sno=SC.Sno AND SC.Cno= 'C2' ;
------------------------------------------------------------------------------------------------	----------------------------------------------------------------------------------

- (3) 检索选修课程名为“DS”的学生学号与姓名

本查询涉及到学号、姓名和课程名三个属性, 分别存放在 S 和 C 表中, 但 S 和 C 表没有直接联系, 必须通过 SC 表建立它们二者的联系。C → SC → S

基本思路:

- (1) 首先在 C 表中找出“DS”课程的课程号 Cno;
- (2) 然后在 SC 表中找出 Cno 等于第一步给出的 Cno 集合中的某个元素 Cno;
- (3) 最后在 S 关系中选出 Sno 等于第二步中 Sno 集合中某个元素的元组, 取出 Sno 和 Sname 送入结果表列。

```
SELECT Sno, Sname
FROM S
WHERE Sno IN
( SELECT Sno
FROM SC
WHERE Cno IN
( SELECT Cno
FROM C
WHERE Cname= 'DS' ));
```

- (4) 检索选修课程号为 C2 或 C4 的学生学号;

```
SELECT Sno
FROM SC
WHERE Cno= 'C2' OR Cno= 'C4';
```

- (5) 检索至少选修课程号为 C2 和 C4 的学生学号;

```
SELECT Sno
FROM SC X, SC Y
WHERE X.Sno=Y.Sno AND X.Cno= 'C2' AND Y.Cno= 'C4' ;
```

- (6) 检索不学 C2 课的学生姓名和年龄;

```
1. SELECT Sname
FROM S
WHERE Sno NOT IN
( SELECT Sno
FROM SC
WHERE Cno= 'C2' );
```

```
2. SELECT Sname
FROM S
WHERE NOT EXISTS
( SELECT *
FROM SC
WHERE SC.Sno=S.Sno
AND Cno= 'C2' );
```

- (7) 检索学习全部课程的学生姓名;

在表 S 中找学生, 要求这个学生学了全部课程。换言之, 在 S 表中找学生, 在 C 中不存在一门课程, 这个学生没有学。

```
SELECT Sname
FROM S
WHERE NOT EXISTS
( SELECT *
FROM C
WHERE NOT EXISTS
( SELECT *
FROM SC
WHERE SC.Sno=S.Sno AND SC.Cno=C.Cno ) );
```

- (8) 查询所学课程包含学生 S3 所学课程的学生学号。

分析: 不存在这样的课程 Y, 学生 S3 选了 Y, 而其他学生没有选。

```
SELECT DISTINCT Sno
FROM SC AS X
WHERE NOT EXISTS
( SELECT *
FROM SC AS Y
WHERE Y.Sno= 'S3' AND NOT EXISTS
( SELECT *
FROM SC AS Z
WHERE Z.Sno=X.Sno AND Z.Cno=Y.Cno ) );
```

设教学数据库 Education 有三个关系：

学生关系 S (SNO, SNAME, AGE, SEX, SDEPT);

学习关系 SC (SNO, CNO, GRADE);

课程关系 C (CNO, CNAME, CDEPT, TNAME)

查询问题：

- 1: 查所有年龄在 20 岁以下的学生姓名及年龄。
- 2: 查考试成绩有不及格的学生的学号
- 3: 查所年龄在 20 至 23 岁之间的学生姓名、系别及年龄。
- 4: 查计算机系、数学系、信息系的学生姓名、性别。
- 5: 查既不是计算机系、数学系、又不是信息系的学生姓名、性别
- 6: 查所有姓“刘”的学生的姓名、学号和性别。
- 7: 查姓“上官”且全名为 3 个汉字的学生姓名。
- 8: 查所有不姓“张”的学生的姓名。
- 9: 查 DB_Design 课程的课程号。
- 10: 查缺考的学生的学号和课程号。
- 11: 查年龄为空值的学生的学号和姓名。
- 12: 查计算机系 20 岁以下的学生的学号和姓名。
- 13: 查计算机系、数学系、信息系的学生姓名、性别。
- 14: 查询选修了 C3 课程的学生的学号和成绩，其结果按分数的降序排列。
- 15: 查询全体学生的情况，查询结果按所在系升序排列，对同一系中的学生按年龄降序排列。
- 16: 查询学生总人数。
- 17: 查询选修了课程的学生人数。
- 18: 计算选修了 C1 课程的学生平均成绩。
- 19: 查询学习 C3 课程的学生最高分数。
- 20: 查询各个课程号与相应的选课人数。
- 21: 查询计算机系选修了 3 门以上课程的学生学号。
- 22: 求基本表 S 中男同学的每一年龄组（超过 50 人）有多少人？要求查询结果按人数升序排列，人数相同按年龄降序排列。
- 23: 查询每个学生及其选修课程的情况。
- 24: 查询选修了 C2 课程且成绩在 90 分以上的所有学生。
- 25: 查询每个学生选修的课程名及其成绩。
- 26: 统计每一年龄选修课程的学生人数。
- 27: 查询选修了 C2 课程的学生姓名。
- 28: 查询与“张三”在同一个系学习的学生学号、姓名和系别。
- 29: 查询选修课程名为“数据库”的学生学号和姓名。
- 30: 查询与“张三”在同一个系学习的学生学号、姓名和系别。
- 31: 查询选修课程名为“数据库”的学生学号和姓名。
- 32: 查询选修了 C2 课程的学生姓名。
- 33: 查询所有未选修 C2 课程的学生姓名。
- 34: 查询与“张三”在同一个系学习的学生学号、姓名和系别。
- 35: 查询选修了全部课程的学生姓名。

36: 查询所学课程包含学生 S3 所学课程的学生学号

(1) 比较

例 1: 查所有年龄在 20 岁以下的学生姓名及年龄。

```
SELECT Sname, Sage
FROM S
WHERE Sage<20;    (NOT age>=20)
```

例 2: 查考试成绩有不及格的学生的学号

```
SELECT DISTINCT Sno
FROM SC
WHERE grade<60;
```

(2) 确定范围

例 3: 查所年龄在 20 至 23 岁之间的学生姓名、系别及年龄。

```
SELECT Sname, Sdept, Sage
FROM S
WHERE Sage BETWEEN 20 AND 23;
```

(3) 确定集合

例 4: 查计算机系、数学系、信息系的学生姓名、性别。

```
SELECT Sname, Ssex
FROM S
WHERE Sdept IN ('CS', 'IS', 'MATH');
```

例 5: 查既不是计算机系、数学系、又不是信息系的学生姓名、性别

```
SELECT Sname, Ssex
FROM S
WHERE Sdept NOT IN ('CS', 'IS', 'MATH');
```

(4) 字符匹配

例 6: 查所有姓“刘”的学生的姓名、学号和性别。

```
SELECT Sname, Sno, Ssex
FROM S
WHERE Sname LIKE '刘%';
```

例 7: 查姓“上官”且全名为 3 个汉字的学生姓名。

```
SELECT Sname
FROM S
WHERE Sname LIKE '上官_ _';
```

例 8: 查所有不姓“张”的学生的姓名。

```
SELECT Sname, Sno, Ssex
FROM S
WHERE Sname NOT LIKE '张%';
```

例 9: 查 DB_Design 课程的课程号。

```
SELECT Cno
FROM C
WHERE Cname LIKE 'DB\Design' ESCAPE '\';
```

(5) 涉及空值的查询

例 10: 查缺考的学生的学号和课程号。

```
SELECT Sno, Cno
FROM SC
WHERE Grade IS NULL;      (不能用=代替)
{ 有成绩的 WHERE Grade IS NOT NULL; }
```

例 11: 查年龄为空值的学生的学号和姓名。

```
SELECT Sno, Sname
FROM S
WHERE Sage IS NULL;
```

(6) 多重条件查询

例 12: 查计算机系 20 岁以下的学生的学号和姓名。

```
SELECT Sno, Sname
FROM S
WHERE Sdept= 'CS' AND Sage<20;
```

例 13: 查计算机系、数学系、信息系的学生姓名、性别。

```
SELECT Sname, Ssex
FROM S
WHERE Sdept= 'CS' OR Sdept= 'IS' OR Sdept= 'MATH';
```

3、对查询结果排序

例 14: 查询选修了 C3 课程的学生的学号和成绩，其结果按分数的降序排列。

```
SELECT Sno, Grade
FROM SC
WHERE Cno= 'C3'
ORDER BY Grade DESC;
```

例 15: 查询全体学生的情况，查询结果按所在系升序排列，对同一系中的学生按年龄降序排列。

```
SELECT *
FROM S
ORDER BY Sdept, Sage DESC;
```

4. 聚合函数的使用

例 16: 查询学生总人数。

```
SELECT COUNT (*)
FROM S
```

例 17: 查询选修了课程的学生人数。

```
SELECT COUNT (DISTINCT Sno)
FROM SC
```

例 18: 计算选修了 C1 课程的学生平均成绩。

```
SELECT AVG (Grade)
FROM SC
WHERE Cno= 'C1';
```

例 19: 查询学习 C3 课程的学生最高分数。

```
SELECT MAX (Grade)
FROM SC
WHERE Cno= 'C3';
```

5、对查询结果分组

例 20: 查询各个课程号与相应的选课人数。

```
SELECT Cno, COUNT (Sno)
FROM SC
GROUP BY Cno;
```

该 SELECT 语句对 SC 表按 Cno 的取值进行分组，所有具有相同 Cno 值的元组为一组，然后对每一组作用聚合函数 COUNT 以求得该组的学生人数。

如果分组后还要求按一定的条件对这些组进行筛选，最终只输出满足指定条件 组，则可以使用 HAVING 短语指定筛选条件。

例 21: 查询计算机系选修了 3 门以上课程的学生的学号。

```
SELECT Sno
FROM SC
WHERE Sdept= 'CS'
GROUP BY Sno
HAVING COUNT (*) >3;
```

WHERE 子句与 HAVING 短语的根本区别在于作用对象不同。WHERE 子句作用于基本表或视图，从中选择满足条件的元组。HAVING 短语作用于组，从中选择满足条件的组。

例 22: 求基本表 S 中男同学的每一年龄组（超过 50 人）有多少人？要求查询结果按人数升序排列，人数相同按年龄降序排列。

```
SELECT Sage, COUNT (Sno)
FROM S
WHERE Ssex= 'M'
GROUP BY Sage
HAVING COUNT (*) > 50
ORDER BY 2, Sage DESC;
```

二、多表查询

1、联接查询

例 23: 查询每个学生及其选修课程的情况。

```
SELECT S. Sno, Sname, Sage, Ssex, Sdept, Cno, Grade
FROM S, SC
WHERE S. Sno=SC. Sno;
```

例 24: 查询选修了 C2 课程且成绩在 90 分以上的所有学生。

```
SELECT S. Sno, Sname
FROM S, SC
WHERE S. Sno=SC. Sno
AND SC. Cno= 'C2'
AND SC. Grade > 90;
```

例 25: 查询每个学生选修的课程名及其成绩。

```
SELECT S. Sno, Sname, Cname, SC. Grade
FROM S, SC, C
WHERE S. Sno=SC. Sno AND SC. Cno=C. Cno
```

例 26: 统计每一年龄选修课程的学生人数。

```
SELECT Sage, COUNT (DISTINCT S. Sno)
FROM S, SC
```

```
WHERE S.Sno=SC.Sno
GROUP BY S;
```

由于要统计每一个年龄的学生人数，因此要把满足 WHERE 子句中条件的查询结果按年龄分组，在每一组中的学生年龄相同。此时的 SELECT 子句应对每一组分开进行操作，在每一组中，年龄只有一个值，统计的人数是这一组中的学生人数。

1、嵌套查询

(1) 带有 IN 谓词的子查询

指父查询与子查询之间用 IN 进行联接，判断某个属性列值是否在子查询的结果中。

例 27：查询选修了 C2 课程的学生姓名。

```
SELECT Sname
FROM S
WHERE Sno IN
      ( SELECT Sno
        FROM SC
        WHERE Cno= 'C2' );
```

例 28：查询与“张三”在同一个系学习的学生学号、姓名和系别。

分析：

- (1) 确定“张三”所在的系；
- (2) 查找所有在 X 系学习的学生。

```
SELECT Sdept
FROM S
WHERE Sname= '张三';
```

```
SELECT Sno, Sname, Sdept
FROM S
WHERE Sdept= 'X'
```

把第一步查询嵌入到第二步查询中，用以构造第二步查询的条件。

```
SELECT Sno, Sname, Sdept
FROM S
WHERE Sdept IN
      (SELECT Sdept
       FROM S
       WHERE Sname= '张三');
```

```
FROM S AS S1, S AS S2
WHERE S1.Sdept=S2.Sdept
AND S2.Sname= '张三'
```

例 29：查询选修课程名为“数据库”的学生学号和姓名。

本查询涉及到学号、姓名和课程名三个属性，分别存放在 S 和 C 表中，但 S 和 C 表没有直接联系，必须通过 SC 表建立它们二者的联系。

$C \rightarrow SC \rightarrow S$

基本思路：

- (1) 首先在 C 表中找出“DB”课程的课程号 Cno；
- (2) 然后在 SC 表中找出 Cno 等于第一步给出的 Cno 集合中的某个元素 Cno；
- (3) 最后在 S 关系中选出 Sno 等于第二步中 Sno 集合中某个元素的元组，取出 Sno 和 Sname 送入结果表列。

```
SELECT Sno, Sname
FROM S
WHERE Sno IN
      (SELECT Sno
       FROM SC
       WHERE Cno IN
            (SELECT Cno
             FROM C
             WHERE Cname= 'DB'));
```

联接查询方式

(2) 带有比较运算符的子查询

例 30：查询与“张三”在同一个系学习的学生学号、姓名和系别。

```
SELECT Sno, Sname, Sdept
FROM S
WHERE Sdept =
      (SELECT Sdept
       FROM S
       WHERE Sname= '张三');
```

例 31：查询选修课程名为“数据库”的学生学号和姓名。

```
SELECT Sno, Sname
FROM S
WHERE Sno IN
      (SELECT Sno
       FROM SC
       WHERE Cno =
            (SELECT Cno
             FROM C
             WHERE Cname= 'DB'));
```

(3) 带有 EXISTS 谓词的子查询

(1) 带有 EXISTS 谓词的子查询不返回任何实际数据，它只产生逻辑值。

例 32：查询选修了 C2 课程的学生姓名。

```
1. SELECT Sname
   FROM S
   WHERE Sno IN
        ( SELECT Sno
          FROM SC
          WHERE Cno= 'C2' );

2. SELECT Sname
   FROM S
   WHERE EXISTS
        ( SELECT *
          FROM SC
          WHERE SC.Sno=S.Sno AND Cno= 'C2' );
```

例 33：查询所有未选修 C2 课程的学生姓名。

```
SELECT Sname
```

```

FROM S
WHERE NOT EXISTS
(
    SELECT *
    FROM SC
    WHERE SC.Sno=S.Sno AND Cno= 'C2' );

```

[NOT]EXISTS 实际上是一种内、外层互相关的嵌套查询，只有当内层引用了外层的值，这种查询才有意义。

例 34：查询与“张三”在同一个系学习的学生学号、姓名和系别。

```

SELECT Sno, Sname, Sdept
FROM S AS S1
WHERE EXISTS
(
    SELECT *
    FROM S AS S2
    WHERE S2.Sdept=S1. Sdept AND S2.Sname= '张三' );

```

相关子查询

例 35：查询选修了全部课程的学生姓名。

在表 S 中找学生，要求这个学生学了全部课程。换言之，在 S 表中找学生，在 C 中不存在一门课程，这个学生没有学。

```

SELECT Sname
FROM S
WHERE NOT EXISTS
(
    SELECT *
    FROM C
    WHERE NOT EXISTS
    (
        SELECT *
        FROM SC
        WHERE SC.Sno=S.Sno AND SC.Cno=C.Cno ) );

```

例 36：查询所学课程包含学生 S3 所学课程的学生学号

分析：不存在这样的课程 Y，学生 S3 选了 Y，而其他学生没有选。

```

SELECT DISTINCT Sno
FROM SC AS X
WHERE NOT EXISTS
(
    SELECT *
    FROM SC AS Y
    WHERE Y.Sno= 'S3' AND NOT EXISTS
    (
        SELECT *
        FROM SC AS Z
        WHERE Z.Sno=X.Sno AND Z.Cno=Y.Cno ) );

```

自写参考答案

1. SELECT SNO, SNAME, SEX FROM S Where SDEPT = 'CS'
 2. SELECT SNO, SNAME FROM S WHERE SNO IN
(SELECT SNO FROM SC WHERE CNO = 'C2')
 3. SELECT SNO,SNAME FORM S WHERE SNO IN
(SELECT SNO FROM SC WHERE CNO IN
(SELECT CNO FROM C WHERE CNAME = 'DS'))
 4. SELECT SNO FROM SC WHERE CNO = 'C2' OR CNO = 'C4'
 5. SELECT SNO FROM SC X, SC Y
WHERE X.SNO = Y.SNO AND X.CNO = 'C2' AND Y.CNO = 'C4'
 6. SELECT SNAME FROM S WHERE SNO NOT IN
(SELECT SNO FROM SC WHERE CNO = 'C2')
 7. SELECT SNAME FROM S WHERE NOT EXISTS
(SELECT * FROM C WHERE NOT EXISTS
(SELECT * FROM SC WHERE SC.SNO = S.SNO AND SC.CNO = C.CNO))
 8. SELECT SNO WHERE SC AS X WHERE NO EXISTS
(SELECT * FORM SC AS Y WHERE Y.SNO = 'S3' AND NOT EXISTS
(SELECT * FROM SC AS Z WHERE Z.SNO = X.SNO AND Z.CNO = Y.CNO))
-
1. SELECT SNAME, AGE FROM S WHERE AGE < 20
 2. SELECT SNO FROM SC WHERE GRAGE < 60
 3. SELECT SNAME, SDEPT, SEX FROM S WHERE AGE < 24 AND AGE > 19
 4. SELECT SNAME, SEX FROM S

WHERE SDEPT = '计算机系' OR SDEPT = '数学系' OR SDEPT = '信息系'
 5. SELECT SNAME, SEX FROM S

WHERE SDEPT <> '计算机系' AND SDEPT <> '数学系' AND SDEPT <> '信息系'
 6. SELECT SNAME, SNO, SEX FROM S WHERE SNAME LIKE '刘%'
 7. SELECT SNAME FROM S WHERE SNAME LIKE '上官%' AND LENGTH(SNAME) = 3
 8. SELECT SNAME, SNO, SEX FROM S WHERE SNAME NOT LIKE '张%'
 9. SELECT CNO FROM C WHERE CNAME = 'DB_Design'
 10. SELECT SNO, CNO FROM SC WHERE GRADE IS NULL
 11. SELECT SNO, SNAME FROM S WHERE AGE IS NULL
 12. SELECT SNO, SNAME FROM S WHERE SDEPT = '计算机系' AND AGE < 20
 13. SELECT SNAME, SEX FROM S

WHERE SDEPT = '计算机系' OR SDEPT = '数学系' OR SDEPT = '信息系'
 14. SELECT SNO, GRADE FROM SC WHERE CNO = 'C3' ORDER BY GRADE DESC

15. SELECT * FROM S ORDER BY SDEPT, AGE DESC
16. SELECT COUNT(*) FROM S
17. SELECT COUNT(SNO) FROM SC
18. SELECT AVG(GRADE) FROM SC WHERE CNO = 'C1'
19. SELECT MAX(GRADE) FROM SC WHERE CNO = 'C3'
20. SELECT CNO, COUNT(SNO) FROM SC GROUP BY CNO
21. SELECT SNO FROM SC
WHERE SNO IN (SELECT SNO FROM S WHERE SDEPT = '计算机系')
GROUP BY SNO HAVING COUNT(*) > 3
22. SELECT AGE, COUNT(SNO) FROM S WHERE SEX = 'male'
GROUP BY AGE HAVING COUNT(*) > 50
ORDER BY 2, AGE DESC
23. SELECT S.SNO, SNAME, AGE, SEX, SDEPT, CNO, GRADE FROM S, SC
WHERE S.SNO = SC.SNO
24. SELECT SNO FROM SC WHERE CNO = 'C2' AND GRADE > 90
25. SELECT S.SNO, SNAME, CNAME, GRADE FROM S, SC, C
WHERE S.SNO = SC.SNO AND SC.CNO = C.CNO
26. SELECT AGE, COUNT(S.SNO) FROM S, SC WHERE S.SNO = SC.SNO
GROUP BY AGE
27. SELECT SNAME FROM S WHERE SNO IN
(SELECT SNO FROM SC WHERE CNO = 'C2')
28. SELECT SNO, SNAME, SDEPT FROM S WHERE SDEPT IN
(SELECT SDEPT FROM S WHERE SNAME = '张三')
29. SELECT SNO, SNAME FROM S WHERE SNO IN
(SELECT SNO FROM SC WHERE CNO IN
(SELECT CNO FROM C WHERE CNAME = 'DB'))
32. SELECT SNAME FROM S WHERE SNO IN
(SELECT SNO FROM SC WHERE CNO = 'C2') 或者
SELECT SNAME FROM S WHERE EXISTS
(SELECT * FROM SC WHERE SC.SNO = S.SNO AND CNO = 'C2')
33. SELECT SNAME FROM S WHERE SNO NOT IN
(SELECT SNO FROM SC WHERE CNO = 'C2') 或者
SELECT SNAME FROM S WHERE NOT EXISTS
(SELECT * FROM SC WHERE SC.SNO = S.SNO AND CNO = 'C2')
35. SELECT SNAME FROM S WHERE NOT EXISTS
(SELECT * FROM C WHERE NOT EXISTS
(SELECT * FROM SC WHERE SC.SNO = S.SNO AND SC.CNO = C.CNO))
36. SELECT SNO FROM SC AS X WHERE NOT EXISTS
(SELECT * FROM SC AS Y WHERE Y.SNO = 'S3' AND NOT EXISTS
(SELECT * FROM SC AS Z WHERE Z.SNO = X.SNO AND Z.CNO = Y.CNO))