实验 4　多表查询-连接查询和嵌套查询

一、目的和要求

（1）掌握简单的多表连接查询，了解多表查询的目的
（2）掌握嵌套查询的用法
（3）掌握带有 IN 谓词、ANY、SOME、ALL 谓词、EXISTS 谓词实现嵌套查询的区别
（4）理解嵌套查询时，=和 IN 的区别
（5）掌握外连接的使用方法和目的
（6）理解嵌套查询和连接查询的区别和效率
（7）掌握利用 AS 给表重新命名的方法和目的
（8）掌握集合查询

二、背景知识

当需要查询的信息来自多个表，称为多表查询。多表查询的主要实现方式有二种即连接查询和嵌套查询。

1. 连接查询

连接查询主要包括多表连接查询、自身连接查询、外连接查询等。

用来连接两个表的条件称为连接条件或连接谓词，其一般格式为：

[<表名 1>．]<列名 1> <比较运算符>［<表名 2>.］<列名 2>

连接谓词中的列名称为连接字段。如果不同关系中具有相同的属性名，为避免混淆必须用“表名.属性名”指定属性属于哪一表。

连接运算中有两种特殊情况，一种称为广义笛卡儿积连接，另一种称为自然连接。广义笛卡儿积连接是不带连接谓词的连接。两个表的广义笛卡儿积连接即是两表中元组的交叉乘积，也即其中一表中的每一元组都要与另一表中的每一元组作拼接，因此结果表往往很大。如果是按照两个表中的相同属性进行等值连接，且目标列中去掉了重复的属性列，但保留了所有不重复的属性列，则称之为自然连接。

连接操作不仅可以在两个表之间进行，也可以是一个表与其自己进行连接，这种连接称为表的自身连接。

连接查询也可以在 FROM 子句中使用连接表达式来实现，可分别实现内连接和外连接。

1）内连接

```
SELECT ……
FROM<表 1> INNER JOIN <表 2>
        ON <连接条件>
```

2）外连接

SELECT ……

FROM<表 1> LEFT OUTER JOIN

| RIGHT OUTER JOIN | FULL OUTER JOIN <表 2>

ON <连接条件>

2. 嵌套查询

在 SQL 语言中，一个 SELECT-FROM-WHERE 语句称为一个查询块。将一个查询块嵌套在另一个查询块的 WHERE 子句或 HAVING 短语的条件中的查询称为嵌套查询，嵌套的 SELECT-FROM-WHERE 查询块称为子查询，需要注意的是：①作为子查询的 SELECT 语句必须放在括号之内；②子查询一般不使用 ORDER BY 字句，但当子查询中使用 TOP 选项时，可以使用 ORDER BY 字句。一般有三种形式的子查询：

(1)、由[NOT] IN 引出子查询

(2)、由[NOT] EXISTS 引出子查询

(3)、由比较运算符引出子查询

注意：在比较子查询中，如果没有使用，ALL 或 ANY 修饰，则必须保证子查询所返回的结果集合中只有单行数据。否则将引起查询错误，如果比较操作与 ALL 或 ANY 修饰一起使用，则允许子查询返回多个数据行。

三、实验内容

要求在查询编辑器窗口中选择 XSGL 数据库为当前数据库，且使用 SQL 语句练习多表连接查询和嵌套查询。

四、实验步骤

1．求学号为‘20022037’的同学的每门课的成绩，输出格式为：学号，课程名，课程成绩

```
SELECT SNO AS 学号, CNAME AS 课程名 ,GRADE AS 课程成绩
FROM sc,course
WHERE sc.CNO=course.CNO AND SNO='20022037'
```

也可以使用连接表达式实现查询，格式如下：

```
SELECT SNO AS 学号, CNAME AS 课程名 ,GRADE AS 课程成绩
FROM sc INNER JOIN course ON sc.CNO=course.CNO
WHERE SNO='20022037'
```

2．查询每个学生的每门课程的成绩，要求输出学号，课程名，成绩

```
SELECT SNO,CNAME,GRADE
FROM sc, course
WHERE   sc.CNO=course.CNO
```

也可以使用连接表达式实现查询，格式如下：

```
SELECT SNO,CNAME,GRADE
FROM sc INNER JOIN course ON sc.CNO=course.CNO
```

3．查询每个学生的每门课程的成绩，要求输出学号，姓名，课程号，成绩

```
SELECT student.SNO,SNAME,CNO,GRADE
FROM student, sc
WHERE   student.SNO=sc.SNO
```

也可以使用连接表达式实现查询，格式如下：

```
SELECT student.SNO,SNAME,CNO,GRADE
FROM student INNER JOIN sc ON student.SNO=sc.SNO
```

以上查询中如果学生没有选课，查询结果中不显示该学生的信息。如果要查询每个学生的信息及其选课信息，如要求输出学号，姓名，课程号，成绩，则可用外连接实现。

```
SELECT student.SNO,SNAME,CNO,GRADE
FROM student LEFT OUTER JOIN sc ON student.SNO=sc.SNO
```

思考：比较以上两个查询的结果有何不同？为什么？

4．查询选修了'线性代数'课程的学生学号、姓名

```
SELECT student.SNO, SNAME
FROM sc, course,student
WHERE sc.SNO=student.SNO AND sc.CNO=course.CNO
    AND   course.CNAME='线性代数'
```

5．查询'线性代数'的所有授课班级的平均成绩，并列出授课班号、教师名、平均成绩，且按平均成绩排序

```
SELECT sc.CNO,course.TNAME,AVG(GRADE)   AS   平均成绩
FROM sc,course
WHERE sc.CNO=course.CNO   AND   CNAME='线性代数'
GROUP BY sc.CNO,course.TNAME
ORDER BY   AVG(GRADE)
```

6．使用多表连接方法，查询和学号为‘20000156’的同学同年同月同日出生的所有学生的学号、姓名、生日。

```
SELECT a.SNO,a.SNAME,a.BIRTHDAY
FROM  student  AS  a,  student  b
WHERE a.BIRTHDAY =b.BIRTHDAY  AND  b.SNO='20000156'
```

7．使用嵌套查询方法，查询和学号为‘20000156’的同学同年同月出生的所有学生的学号、姓名、生日。

```
SELECT SNO,SNAME,BIRTHDAY
FROM    student
WHERE  YEAR(BIRTHDAY)+MONTH(BIRTHDAY)=
      (
        SELECT YEAR(BIRTHDAY)+MONTH(BIRTHDAY)    FROM student
        WHERE SNO='20000156'
      )
```

说明：该嵌套子查询只执行一次，整个查询效率比第 6 题快

8．使用嵌套查询方法，查询“赵蓉”教师任课的学生成绩，并按成绩递增排列

```
SELECT CNO,SNO,GRADE
FROM sc
WHERE CNO  IN
   ( SELECT CNO    FROM  course
     WHERE  TNAME='赵蓉' )
ORDER BY GRADE
```

说明：该嵌套子查询只执行一次，执行效率比多表连接查询效率高

9．使用嵌套查询方法，查询课程最低分大于 70，最高分小于 90 的学生学号和姓名

```
SELECT SNO,SNAME
FROM student
WHERE SNO IN
      ( SELECT SNO
      FROM sc
      GROUP BY sc.SNO
      HAVING MIN(GRADE)>70 AND MAX(GRADE)<90 )
```

10．用嵌套法查询选修了“线性代数“的学生学号和姓名

```
SELECT SNO,SNAME
FROM student
WHERE SNO IN
    (
    SELECT SNO FROM   sc
    WHERE CNO IN
          ( SELECT CNO     FROM   course
          WHERE CNAME='线性代数' )
    )
```

说明：该查询使用了两层嵌套查询，查询次序为从里向外执行

11．从选修’218801’课程的同学中，选出成绩高于’季莹’的学生的学号和成绩

```
SELECT SNO,GRADE
FROM sc
WHERE CNO='218801'     AND GRADE >
      (
       SELECT GRADE   FROM sc
       WHERE      CNO='218801'      AND   SNO=
            ( SELECT SNO     FROM student
             WHERE SNAME='季莹 ' )
       )
```

说明：先执行子查询，再执行主查询，该子查询只执行一次

12．查询成绩比该课程平均成绩低的学生成绩表

```
SELECT SNO,CNO,GRADE
FROM sc AS   a
WHERE GRADE<
    ( SELECT AVG(GRADE)
      FROM sc   AS   b
      WHERE   a.CNO=b.CNO
    )
```

说明：主查询在判断每个待选行时，唤醒子查询，告诉它该学生选修的课程号，并由子查询计算该课程的平均成绩，然后将该学生的成绩与平均成绩进行比较，找出符合条件的记录，这种子查询称为相关子查询。

13．查询选修了'线性代数'这门课程的学生学号

```
SELECT SNO,SNAME
FROM student
WHERE  EXISTS
      ( SELECT *
        FROM  sc  ,course
        WHERE      sc.CNO=course.CNO      AND  student.SNO=sc.SNO  AND
course.CNAME='线性代数'
      )
```

说明：主查询在判断每个学生时，执行子查询，根据主查询中的当前行的学号，在子查询中，从头到尾进行扫描，判断是否存在该学生的选课记录，如果存在这样的行，EXISTS 子句返回真，主查询选中当前行；如果子查询未找到这样的行，EXISTS 子句返回假，主查询不选中当前行。

14．查询所有学生都选修的课程名

```
SELECT CNAME
FROM course
WHERE not EXISTS
   (
      SELECT *    FROM student
      WHERE not EXISTS
      ( SELECT *   FROM sc
        WHERE SNO=student.SNO AND CNO=course.CNO )
   )
```

15．查询选修了'线性代数'课程或'英语口语'课程的学生学号、姓名。

```
SELECT DISTINCT student.SNO,SNAME
FROM sc,course,student
WHERE    sc.SNO=student.SNO AND sc.CNO=course.CNO AND
   (course.CNAME='线性代数'    OR   course.CNAME='英语口语')
```

16. 用集合操作符 UNION 查询选修了'线性代数'课程或'英语口语'课程的学生学号、姓名。

```
SELECT student.SNO,SNAME    FROM sc,course,student
WHERE  sc.SNO=student.SNO AND sc.CNO=course.CNO AND  course.CNAME='线性
代数'
UNION
```

```
SELECT student.SNO,SNAME     FROM sc,course,student
WHERE     sc.SNO=student.SNO AND sc.CNO=course.CNO
AND    course.CNAME='英语口语'
```

本查询也可以用以下查询实现：

```
SELECT student.SNO,SNAME     FROM sc,course,student
WHERE    sc.SNO=student.SNO AND sc.CNO=course.CNO
    AND    course.CNAME='线性代数' OR course.CNAME='英语口语'
```

17．查询选修了'218801'课程但没有选修'216301'课程的学生学号。

此查询可通过集合差 EXCEPT 实现。

```
SELECT SNO，SNAME
FROM student,sc
WHERE    sc.SNO=student AND CNO='218801'
EXCEPT
SELECT SNO，SNAME
FROM student,sc
WHERE    sc.SNO=student AND CNO='216301'
```

本查询也可以用以下子查询实现：

```
SELECT SNO，SNAME    FROM student
WHERE SNO IN
    ( SELECT SNO     FROM sc
      WHERE    CNO='218801'    )
    AND    SNO NOT IN
        ( SELECT SNO     FROM sc
          WHERE CNO='216301' )
```

18．求同时选修'218801'课程和'216301'课程的学生学号、姓名。

此查询可通过集合交 INTERSECT 实现。

```
SELECT student. SNO，SNAME
FROM student,sc
WHERE    sc.SNO=student AND CNO='218801'
INTERSECT
SELECT student. SNO，SNAME
FROM student,sc
WHERE    sc.SNO=student AND CNO='216301'
```

本查询也可以用以下子查询实现：

```
SELECT SNO，SNAME  FROM student
WHERE SNO IN
      (SELECT  SNO   FROM  sc
       WHERE CNO='218801'   AND     SNO  IN
             ( SELECT SNO   FROM sc
               WHERE CNO='216301' ))
```

19．查询所有学生及其选课信息

```
SELECT student.SNO,SNAME,CNO,GRADE
FROM student left outer JOIN sc
ON student.SNO=sc.SNO
```

20．创建课程平均分视图

```
CREATE VIEW 查询课程平均分 AS
SELECT (AVG(GRADE)) AS 平均分, CNAME
FROM sc,course
WHERE sc.CNO=course.CNO
GROUP BY CNAME
```

21．以列的方式统计每门课程的分数段人数。分数段为：不及格、60-70、70-80、80-90、90-100

```
(SELECT CNAME ,'不及格' AS fsd ,COUNT(*) AS rs
FROM sc,course
WHERE sc.CNO=course.CNO AND GRADE <60
GROUP BY   CNAME)
UNION
(SELECT CNAME ,'60-70' AS fsd , COUNT (*)
FROM sc,course
WHERE sc.CNO=course.CNO  AND GRADE BETWEEN 60 AND 70
GROUP BY   CNAME)
UNION
(SELECT CNAME ,'70-80' AS fsd , COUNT (*)
FROM sc,course
WHERE sc.CNO=course.CNO  AND GRADE BETWEEN 70 AND 80
GROUP BY  CNAME)
UNION
```

```
(SELECT CNAME ,'80-90' AS fsd , COUNT (*)
FROM sc,course
WHERE sc.CNO=course.CNO    AND GRADE BETWEEN 80 AND 90
GROUP BY    CNAME)
UNION
(SELECT CNAME ,'90-100' AS fsd , COUNT (*)
FROM sc,course
WHERE sc.CNO=course.CNO    AND GRADE BETWEEN 90 AND 100
GROUP BY    CNAME)
```

思考模仿题：

1．查询所有选课学生的姓名

```
SELECT TNAME
FROM student AS    a
WHERE    EXISTS
      ( SELECT *
        FROM    sc    AS    b
        WHERE    a.SNO= b.SNO )
```

2．查询所有未选课的学生的姓名

```
SELECT TNAME
FROM student AS    a
WHERE    not EXISTS
  ( SELECT *
    FROM    sc    AS    b
    WHERE    a.SNO = b.SNO )
```

3．按学生分类查询其选修课程的平均分，输出学号、姓名和平均成绩

```
SELECT student.SNO,student.SNAME,AVG(GRADE)
FROM student,sc
WHERE student.SNO=sc.SNO
GROUP BY student.SNO,student.SNAME
```

4．查询所有课程的平均分，输出课程名和平均成绩，并按平均成绩递增

```
SELECT CNAME,AVG(GRADE)
FROM sc,course
WHERE sc.CNO=course.CNO
```

```
GROUP     BY    CNAME
ORDER BY    AVG(GRADE)
```

5．查询少于 10 名同学选修的课程名称，授课班号，教师名，选课人数

```
SELECT sc.CNO,CNAME,TNAME,COUNT(*)
FROM sc,course
WHERE sc.CNO=course.CNO
GROUP BY sc.CNO,CNAME,TNAME
HAVING COUNT(*)<10
ORDER BY sc.CNO
```

6．按学号显示信息学院，‘通信专业’或‘电子科学专业’的每个学生的每门课程的成绩明细，并统计每个学生的总成绩，平均成绩

```
SELECT SNO,CNO,GRADE
FROM sc
WHERE SUBSTRING(SNO,5,2) IN('22','24')
ORDER BY SNO
COMPUTE SUM(GRADE),AVG(GRADE),MAX(GRADE)    BY SNO
```

7．统计每门课的不及格人数，列出课程名和不及格人数

```
SELECT CNAME ,'不及格分数段' AS fsd ,COUNT(*) AS rs
FROM sc, course
WHERE sc.CNO= course.CNO AND GRADE<60
GROUP BY     CNAME
```

思考题

1、在学生管理数据库中，完成以下查询：

（1） 使用嵌套方法查询存在有 95 分以上成绩的课程 CNO
（2） 查询成绩比该课程平均成绩低的学生成绩表
（3） 按课程名称统计每一门课程的平均分，输出课程名称和平均分
（4） 按学生姓名统计其选修课程的总学分，输出学生姓名和总学分
（5） 查询同时选修了‘203402’和‘244501’课程的同学名称
（6） 求最高分学生的学号
（7） 查询“线性代数”的所有授课班级的平均成绩，列出课程名和平均成绩
（8） 查询“线性代数”成绩最高的前 5 名学生的姓名及成绩，结果按成绩降序
（9） 查询学生“20002059”选修课程的总学分数
（10）对每个同学，查找其获得最高成绩的课程号

2、完成书后作业的查询。