**SQL面试题大全**

Student(S#学号),Sname,Sage,Ssex) 学生表

Course(C#（课程号）,Cname,T#（老师编号）) 课程表

SC(S#（学号）,C#（课程号）,score) 成绩表

Teacher(T#（老师编号）,Tname) 教师表

**问题：**

1、查询“001”课程比“002”课程成绩高的所有学生的学号；

select a.s# from (select s#.score from sc where c#=’001’)as a,(select s#,score from sc where c#=’002’)as b where a.score>b.score and a.s#=b.s#

  select a.S# from (select s#,score from SC where C#='001') a,(select s#,score

  from SC where C#='002') b

  where a.score>b.score and a.s#=b.s#;

2、查询平均成绩大于60分的同学的学号和平均成绩；

select s# ,avg(score) from sc group by s# having avg(score)>60

    select S#,avg(score)

    from sc

    group by S# having avg(score) >60;

select student.s# ,student.name,sc.count(c#),sc.sum(score)from student left join sc on student.s#=sc.s# group by sc.s#,sc.c#

  select Student.S#,Student.Sname,count(SC.C#),sum(score)

  from Student left Outer join SC on Student.S#=SC.S#

  group by Student.S#,Sname

4、查询姓“李”的老师的个数；

  select count(distinct(Tname))

  from Teacher

  where Tname like '李%';

5、查询没学过“叶平”老师课的同学的学号、姓名；

    select Student.S#,Student.Sname

    from Student

    where S# not in (select distinct( SC.S#) from SC,Course,Teacher where  SC.C#=Course.C# and Teacher.T#=Course.T# and Teacher.Tname='叶平');

6、查询学过“001”并且也学过编号“002”课程的同学的学号、姓名；

  select Student.S#,Student.Sname from Student,SC where Student.S#=SC.S# and SC.C#='001'and exists( Select \* from SC as SC\_2 where SC\_2.S#=SC.S# and SC\_2.C#='002');

7、查询学过“叶平”老师所教的所有课的同学的学号、姓名；

  select S#,Sname

  from Student

  where S# in (select S# from SC ,Course ,Teacher where SC.C#=Course.C# and Teacher.T#=Course.T# and Teacher.Tname='叶平' group by S# having count(SC.C#)=(select count(C#) from Course,Teacher  where Teacher.T#=Course.T# and Tname='叶平'));

8、查询课程编号“002”的成绩比课程编号“001”课程低的所有同学的学号、姓名；

  Select S#,Sname from (select Student.S#,Student.Sname,score ,(select score from SC SC\_2 where SC\_2.S#=Student.S# and SC\_2.C#='002') score2

  from Student,SC where Student.S#=SC.S# and C#='001') S\_2 where score2 <score;

9、查询所有课程成绩小于60分的同学的学号、姓名；

  select S#,Sname

  from Student

  where S# not in (select Student.S# from Student,SC where S.S#=SC.S# and score>60);

10、查询没有学全所有课的同学的学号、姓名；

    select Student.S#,Student.Sname

    from Student,SC

    where Student.S#=SC.S# group by  Student.S#,Student.Sname having count(C#) <(select count(C#) from Course);

11、查询至少有一门课与学号为“1001”的同学所学相同的同学的学号和姓名；

    select S#,Sname from Student,SC where Student.S#=SC.S# and C# in select C# from SC where S#='1001';

12、查询至少学过学号为“001”同学所有一门课的其他同学学号和姓名；

    select distinct SC.S#,Sname

    from Student,SC

    where Student.S#=SC.S# and C# in (select C# from SC where S#='001');

13、把“SC”表中“叶平”老师教的课的成绩都更改为此课程的平均成绩；

    update SC set score=(select avg(SC\_2.score)

    from SC SC\_2

    where SC\_2.C#=SC.C# ) from Course,Teacher where Course.C#=SC.C# and Course.T#=Teacher.T# and Teacher.Tname='叶平');

14、查询和“1002”号的同学学习的课程完全相同的其他同学学号和姓名；

    select S# from SC where C# in (select C# from SC where S#='1002')

    group by S# having count(\*)=(select count(\*) from SC where S#='1002');

15、删除学习“叶平”老师课的SC表记录；

    Delect SC

    from course ,Teacher

    where Course.C#=SC.C# and Course.T#= Teacher.T# and Tname='叶平';

16、向SC表中插入一些记录，这些记录要求符合以下条件：没有上过编号“003”课程的同学学号、2、

    号课的平均成绩；

    Insert SC select S#,'002',(Select avg(score)

    from SC where C#='002') from Student where S# not in (Select S# from SC where C#='002');

17、按平均成绩从高到低显示所有学生的“数据库”、“企业管理”、“英语”三门的课程成绩，按如下形式显示： 学生ID,,数据库,企业管理,英语,有效课程数,有效平均分

    SELECT S# as 学生ID

        ,(SELECT score FROM SC WHERE SC.S#=t.S# AND C#='004') AS 数据库

        ,(SELECT score FROM SC WHERE SC.S#=t.S# AND C#='001') AS 企业管理

        ,(SELECT score FROM SC WHERE SC.S#=t.S# AND C#='006') AS 英语

        ,COUNT(\*) AS 有效课程数, AVG(t.score) AS 平均成绩

    FROM SC AS t

    GROUP BY S#

    ORDER BY avg(t.score)

18、查询各科成绩最高和最低的分：以如下形式显示：课程ID，最高分，最低分

    SELECT L.C# As 课程ID,L.score AS 最高分,R.score AS 最低分

    FROM SC L ,SC AS R

    WHERE L.C# = R.C# and

        L.score = (SELECT MAX(IL.score)

                      FROM SC AS IL,Student AS IM

                      WHERE L.C# = IL.C# and IM.S#=IL.S#

                      GROUP BY IL.C#)

        AND

        R.Score = (SELECT MIN(IR.score)

                      FROM SC AS IR

                      WHERE R.C# = IR.C#

                  GROUP BY IR.C#

                    );

19、按各科平均成绩从低到高和及格率的百分数从高到低顺序

    SELECT t.C# AS 课程号,max(course.Cname)AS 课程名,isnull(AVG(score),0) AS 平均成绩

        ,100 \* SUM(CASE WHEN  isnull(score,0)>=60 THEN 1 ELSE 0 END)/COUNT(\*) AS 及格百分数

    FROM SC T,Course

    where t.C#=course.C#

    GROUP BY t.C#

    ORDER BY 100 \* SUM(CASE WHEN  isnull(score,0)>=60 THEN 1 ELSE 0 END)/COUNT(\*) DESC

20、查询如下课程平均成绩和及格率的百分数(用"1行"显示): 企业管理（001），马克思（002），OO&UML （003），数据库（004）

    SELECT SUM(CASE WHEN C# ='001' THEN score ELSE 0 END)/SUM(CASE C# WHEN '001' THEN 1 ELSE 0 END) AS 企业管理平均分

        ,100 \* SUM(CASE WHEN C# = '001' AND score >= 60 THEN 1 ELSE 0 END)/SUM(CASE WHEN C# = '001' THEN 1 ELSE 0 END) AS 企业管理及格百分数

        ,SUM(CASE WHEN C# = '002' THEN score ELSE 0 END)/SUM(CASE C# WHEN '002' THEN 1 ELSE 0 END) AS 马克思平均分

        ,100 \* SUM(CASE WHEN C# = '002' AND score >= 60 THEN 1 ELSE 0 END)/SUM(CASE WHEN C# = '002' THEN 1 ELSE 0 END) AS 马克思及格百分数

        ,SUM(CASE WHEN C# = '003' THEN score ELSE 0 END)/SUM(CASE C# WHEN '003' THEN 1 ELSE 0 END) AS UML平均分

        ,100 \* SUM(CASE WHEN C# = '003' AND score >= 60 THEN 1 ELSE 0 END)/SUM(CASE WHEN C# = '003' THEN 1 ELSE 0 END) AS UML及格百分数

        ,SUM(CASE WHEN C# = '004' THEN score ELSE 0 END)/SUM(CASE C# WHEN '004' THEN 1 ELSE 0 END) AS 数据库平均分

        ,100 \* SUM(CASE WHEN C# = '004' AND score >= 60 THEN 1 ELSE 0 END)/SUM(CASE WHEN C# = '004' THEN 1 ELSE 0 END) AS 数据库及格百分数

  FROM SC

21、查询不同老师所教不同课程平均分从高到低显示

  SELECT max(Z.T#) AS 教师ID,MAX(Z.Tname) AS 教师姓名,C.C# AS 课程ＩＤ,MAX(C.Cname) AS 课程名称,AVG(Score) AS 平均成绩

    FROM SC AS T,Course AS C ,Teacher AS Z

    where T.C#=C.C# and C.T#=Z.T#

  GROUP BY C.C#

  ORDER BY AVG(Score) DESC

22、查询如下课程成绩第 3 名到第 6 名的学生成绩单：企业管理（001），马克思（002），UML （003），数据库（004）

    [学生ID],[学生姓名],企业管理,马克思,UML,数据库,平均成绩

    SELECT  DISTINCT top 3

      SC.S# As 学生学号,

        Student.Sname AS 学生姓名 ,

      T1.score AS 企业管理,

      T2.score AS 马克思,

      T3.score AS UML,

      T4.score AS 数据库,

      ISNULL(T1.score,0) + ISNULL(T2.score,0) + ISNULL(T3.score,0) + ISNULL(T4.score,0) as 总分

      FROM Student,SC  LEFT JOIN SC AS T1

                      ON SC.S# = T1.S# AND T1.C# = '001'

            LEFT JOIN SC AS T2

                      ON SC.S# = T2.S# AND T2.C# = '002'

            LEFT JOIN SC AS T3

                      ON SC.S# = T3.S# AND T3.C# = '003'

            LEFT JOIN SC AS T4

                      ON SC.S# = T4.S# AND T4.C# = '004'

      WHERE student.S#=SC.S# and

      ISNULL(T1.score,0) + ISNULL(T2.score,0) + ISNULL(T3.score,0) + ISNULL(T4.score,0)

      NOT IN

      (SELECT

            DISTINCT

            TOP 15 WITH TIES

            ISNULL(T1.score,0) + ISNULL(T2.score,0) + ISNULL(T3.score,0) + ISNULL(T4.score,0)

      FROM sc

            LEFT JOIN sc AS T1

                      ON sc.S# = T1.S# AND T1.C# = 'k1'

            LEFT JOIN sc AS T2

                      ON sc.S# = T2.S# AND T2.C# = 'k2'

            LEFT JOIN sc AS T3

                      ON sc.S# = T3.S# AND T3.C# = 'k3'

            LEFT JOIN sc AS T4

                      ON sc.S# = T4.S# AND T4.C# = 'k4'

      ORDER BY ISNULL(T1.score,0) + ISNULL(T2.score,0) + ISNULL(T3.score,0) + ISNULL(T4.score,0) DESC);

23、统计列印各科成绩,各分数段人数:课程ID,课程名称,[100-85],[85-70],[70-60],[ <60]

    SELECT SC.C# as 课程ID, Cname as 课程名称

        ,SUM(CASE WHEN score BETWEEN 85 AND 100 THEN 1 ELSE 0 END) AS [100 - 85]

        ,SUM(CASE WHEN score BETWEEN 70 AND 85 THEN 1 ELSE 0 END) AS [85 - 70]

        ,SUM(CASE WHEN score BETWEEN 60 AND 70 THEN 1 ELSE 0 END) AS [70 - 60]

        ,SUM(CASE WHEN score < 60 THEN 1 ELSE 0 END) AS [60 -]

    FROM SC,Course

    where SC.C#=Course.C#

    GROUP BY SC.C#,Cname;

24、查询学生平均成绩及其名次

      SELECT 1+(SELECT COUNT( distinct 平均成绩)

              FROM (SELECT S#,AVG(score) AS 平均成绩

                      FROM SC

                  GROUP BY S#

                  ) AS T1

            WHERE 平均成绩 > T2.平均成绩) as 名次,

      S# as 学生学号,平均成绩

    FROM (SELECT S#,AVG(score) 平均成绩

            FROM SC

        GROUP BY S#

        ) AS T2

    ORDER BY 平均成绩 desc;

25、查询各科成绩前三名的记录:(不考虑成绩并列情况)

      SELECT t1.S# as 学生ID,t1.C# as 课程ID,Score as 分数

      FROM SC t1

      WHERE score IN (SELECT TOP 3 score

              FROM SC

              WHERE t1.C#= C#

            ORDER BY score DESC

              )

      ORDER BY t1.C#;

26、查询每门课程被选修的学生数

  select c#,count(S#) from sc group by C#;

27、查询出只选修了一门课程的全部学生的学号和姓名

  select SC.S#,Student.Sname,count(C#) AS 选课数

  from SC ,Student

  where SC.S#=Student.S# group by SC.S# ,Student.Sname having count(C#)=1;

28、查询男生、女生人数

    Select count(Ssex) as 男生人数 from Student group by Ssex having Ssex='男';

    Select count(Ssex) as 女生人数 from Student group by Ssex having Ssex='女'；

29、查询姓“张”的学生名单

    SELECT Sname FROM Student WHERE Sname like '张%';

30、查询同名同性学生名单，并统计同名人数

  select Sname,count(\*) from Student group by Sname having  count(\*)>1;;

31、1981年出生的学生名单(注：Student表中Sage列的类型是datetime)

    select Sname,  CONVERT(char (11),DATEPART(year,Sage)) as age

    from student

    where  CONVERT(char(11),DATEPART(year,Sage))='1981';

32、查询每门课程的平均成绩，结果按平均成绩升序排列，平均成绩相同时，按课程号降序排列

    Select C#,Avg(score) from SC group by C# order by Avg(score),C# DESC ;

33、查询平均成绩大于85的所有学生的学号、姓名和平均成绩

    select Sname,SC.S# ,avg(score)

    from Student,SC

    where Student.S#=SC.S# group by SC.S#,Sname having    avg(score)>85;

34、查询课程名称为“数据库”，且分数低于60的学生姓名和分数

    Select Sname,isnull(score,0)

    from Student,SC,Course

    where SC.S#=Student.S# and SC.C#=Course.C# and  Course.Cname='数据库'and score <60;

35、查询所有学生的选课情况；

    SELECT SC.S#,SC.C#,Sname,Cname

    FROM SC,Student,Course

    where SC.S#=Student.S# and SC.C#=Course.C# ;

36、查询任何一门课程成绩在70分以上的姓名、课程名称和分数；

    SELECT  distinct student.S#,student.Sname,SC.C#,SC.score

    FROM student,Sc

    WHERE SC.score>=70 AND SC.S#=student.S#;

37、查询不及格的课程，并按课程号从大到小排列

    select c# from sc where scor e <60 order by C# ;

38、查询课程编号为003且课程成绩在80分以上的学生的学号和姓名；

    select SC.S#,Student.Sname from SC,Student where SC.S#=Student.S# and Score>80 and C#='003';

39、求选了课程的学生人数

    select count(\*) from sc;

40、查询选修“叶平”老师所授课程的学生中，成绩最高的学生姓名及其成绩

    select Student.Sname,score

    from Student,SC,Course C,Teacher

    where Student.S#=SC.S# and SC.C#=C.C# and C.T#=Teacher.T# and Teacher.Tname='叶平' and SC.score=(select max(score)from SC where C#=C.C# );

41、查询各个课程及相应的选修人数

    select count(\*) from sc group by C#;

42、查询不同课程成绩相同的学生的学号、课程号、学生成绩

  select distinct  A.S#,B.score from SC A  ,SC B where A.Score=B.Score and A.C# <>B.C# ;

43、查询每门功成绩最好的前两名

    SELECT t1.S# as 学生ID,t1.C# as 课程ID,Score as 分数

      FROM SC t1

      WHERE score IN (SELECT TOP 2 score

              FROM SC

              WHERE t1.C#= C#

            ORDER BY score DESC

              )

      ORDER BY t1.C#;

44、统计每门课程的学生选修人数（超过10人的课程才统计）。要求输出课程号和选修人数，查询结果按人数降序排列，查询结果按人数降序排列，若人数相同，按课程号升序排列

    select  C# as 课程号,count(\*) as 人数

    from  sc

    group  by  C#

    order  by  count(\*) desc,c#

45、检索至少选修两门课程的学生学号

    select  S#

    from  sc

    group  by  s#

    having  count(\*)  >  =  2

46、查询全部学生都选修的课程的课程号和课程名

    select  C#,Cname

    from  Course

    where  C#  in  (select  c#  from  sc group  by  c#)

47、查询没学过“叶平”老师讲授的任一门课程的学生姓名

    select Sname from Student where S# not in (select S# from Course,Teacher,SC where Course.T#=Teacher.T# and SC.C#=course.C# and Tname='叶平');

48、查询两门以上不及格课程的同学的学号及其平均成绩

    select S#,avg(isnull(score,0)) from SC where S# in (select S# from SC where score <60 group by S# having count(\*)>2)group by S#;

49、检索“004”课程分数小于60，按分数降序排列的同学学号

    select S# from SC where C#='004'and score <60 order by score desc;

50、删除“002”同学的“001”课程的成绩

delete from Sc where S#='001'and C#='001';

问题描述：

本题用到下面三个关系表：

CARD     借书卡。   CNO 卡号，NAME  姓名，CLASS 班级

BOOKS    图书。     BNO 书号，BNAME 书名,AUTHOR 作者，PRICE 单价，QUANTITY 库存册数

BORROW   借书记录。 CNO 借书卡号，BNO 书号，RDATE 还书日期

备注：限定每人每种书只能借一本；库存册数随借书、还书而改变。

要求实现如下15个处理：

  1． 写出建立BORROW表的SQL语句，要求定义主码完整性约束和引用完整性约束。

  2． 找出借书超过5本的读者,输出借书卡号及所借图书册数。

  3． 查询借阅了"水浒"一书的读者，输出姓名及班级。

  4． 查询过期未还图书，输出借阅者（卡号）、书号及还书日期。

  5． 查询书名包括"网络"关键词的图书，输出书号、书名、作者。

  6． 查询现有图书中价格最高的图书，输出书名及作者。

  7． 查询当前借了"计算方法"但没有借"计算方法习题集"的读者，输出其借书卡号，并按卡号降序排序输出。

  8． 将"C01"班同学所借图书的还期都延长一周。

  9． 从BOOKS表中删除当前无人借阅的图书记录。

  10．如果经常按书名查询图书信息，请建立合适的索引。

  11．在BORROW表上建立一个触发器，完成如下功能：如果读者借阅的书名是"数据库技术及应用"，就将该读者的借阅记录保存在BORROW\_SAVE表中（注ORROW\_SAVE表结构同BORROW表）。

  12．建立一个视图，显示"力01"班学生的借书信息（只要求显示姓名和书名）。

  13．查询当前同时借有"计算方法"和"组合数学"两本书的读者，输出其借书卡号，并按卡号升序排序输出。

  14．假定在建BOOKS表时没有定义主码，写出为BOOKS表追加定义主码的语句。

  15．对CARD表做如下修改：

    a. 将NAME最大列宽增加到10个字符（假定原为6个字符）。

    b. 为该表增加1列NAME（系名），可变长，最大20个字符。

1. 写出建立BORROW表的SQL语句，要求定义主码完整性约束和引用完整性约束

--实现代码：

CREATE TABLE BORROW(

    CNO int FOREIGN KEY REFERENCES CARD(CNO),

    BNO int FOREIGN KEY REFERENCES BOOKS(BNO),

    RDATE datetime,

    PRIMARY KEY(CNO,BNO))

2. 找出借书超过5本的读者,输出借书卡号及所借图书册数

--实现代码：

SELECT CNO,借图书册数=COUNT(\*)

FROM BORROW

GROUP BY CNO

HAVING COUNT(\*)>5

3. 查询借阅了"水浒"一书的读者，输出姓名及班级

--实现代码：

SELECT \* FROM CARD c

WHERE EXISTS(

    SELECT \* FROM BORROW a,BOOKS b

    WHERE a.BNO=b.BNO

        AND b.BNAME=N'水浒'

        AND a.CNO=c.CNO)

4. 查询过期未还图书，输出借阅者（卡号）、书号及还书日期

--实现代码：

SELECT \* FROM BORROW

WHERE RDATE<GETDATE()

5. 查询书名包括"网络"关键词的图书，输出书号、书名、作者

--实现代码：

SELECT BNO,BNAME,AUTHOR FROM BOOKS

WHERE BNAME LIKE N'%网络%'

6. 查询现有图书中价格最高的图书，输出书名及作者

--实现代码：

SELECT BNO,BNAME,AUTHOR FROM BOOKS

WHERE PRICE=(

    SELECT MAX(PRICE) FROM BOOKS)

7. 查询当前借了"计算方法"但没有借"计算方法习题集"的读者，输出其借书卡号，并按卡号降序排序输出

--实现代码：

SELECT a.CNO

FROM BORROW a,BOOKS b

WHERE a.BNO=b.BNO AND b.BNAME=N'计算方法'

    AND NOT EXISTS(

        SELECT \* FROM BORROW aa,BOOKS bb

        WHERE aa.BNO=bb.BNO

            AND bb.BNAME=N'计算方法习题集'

            AND aa.CNO=a.CNO)

ORDER BY a.CNO DESC

8. 将"C01"班同学所借图书的还期都延长一周

--实现代码：

UPDATE b SET RDATE=DATEADD(Day,7,b.RDATE)

FROM CARD a,BORROW b

WHERE a.CNO=b.CNO

    AND a.CLASS=N'C01'

9. 从BOOKS表中删除当前无人借阅的图书记录

--实现代码：

DELETE A FROM BOOKS a

WHERE NOT EXISTS(

    SELECT \* FROM BORROW

    WHERE BNO=a.BNO)

10. 如果经常按书名查询图书信息，请建立合适的索引

--实现代码：

CREATE CLUSTERED INDEX IDX\_BOOKS\_BNAME ON BOOKS(BNAME)

11. 在BORROW表上建立一个触发器，完成如下功能：如果读者借阅的书名是"数据库技术及应用"，就将该读者的借阅记录保存在BORROW\_SAVE表中（注ORROW\_SAVE表结构同BORROW表）

--实现代码：

CREATE TRIGGER TR\_SAVE ON BORROW

FOR INSERT,UPDATE

AS

IF @@ROWCOUNT>0

INSERT BORROW\_SAVE SELECT i.\*

FROM INSERTED i,BOOKS b

WHERE i.BNO=b.BNO

    AND b.BNAME=N'数据库技术及应用'

12. 建立一个视图，显示"力01"班学生的借书信息（只要求显示姓名和书名）

--实现代码：

CREATE VIEW V\_VIEW

AS

SELECT a.NAME,b.BNAME

FROM BORROW ab,CARD a,BOOKS b

WHERE ab.CNO=a.CNO

    AND ab.BNO=b.BNO

    AND a.CLASS=N'力01'

13. 查询当前同时借有"计算方法"和"组合数学"两本书的读者，输出其借书卡号，并按卡号升序排序输出

--实现代码：

SELECT a.CNO

FROM BORROW a,BOOKS b

WHERE a.BNO=b.BNO

    AND b.BNAME IN(N'计算方法',N'组合数学')

GROUP BY a.CNO

HAVING COUNT(\*)=2

ORDER BY a.CNO DESC

14. 假定在建BOOKS表时没有定义主码，写出为BOOKS表追加定义主码的语句

--实现代码：

ALTER TABLE BOOKS ADD PRIMARY KEY(BNO)

15.1 将NAME最大列宽增加到10个字符（假定原为6个字符）

--实现代码：

ALTER TABLE CARD ALTER COLUMN NAME varchar(10)

15.2 为该表增加1列NAME（系名），可变长，最大20个字符

--实现代码：

ALTER TABLE CARD ADD 系名 varchar(20)

问题描述：

为管理岗位业务培训信息，建立3个表:

S (S#,SN,SD,SA)   S#,SN,SD,SA 分别代表学号、学员姓名、所属单位、学员年龄

C (C#,CN )        C#,CN       分别代表课程编号、课程名称

SC ( S#,C#,G )    S#,C#,G     分别代表学号、所选修的课程编号、学习成绩

要求实现如下5个处理：

  1. 使用标准SQL嵌套语句查询选修课程名称为’税收基础’的学员学号和姓名

  2. 使用标准SQL嵌套语句查询选修课程编号为’C2’的学员姓名和所属单位

  3. 使用标准SQL嵌套语句查询不选修课程编号为’C5’的学员姓名和所属单位

  4. 使用标准SQL嵌套语句查询选修全部课程的学员姓名和所属单位

  5. 查询选修了课程的学员人数

  6. 查询选修课程超过5门的学员学号和所属单位

1. 使用标准SQL嵌套语句查询选修课程名称为’税收基础’的学员学号和姓名

--实现代码：

SELECT SN,SD FROM S

WHERE [S#] IN(

    SELECT [S#] FROM C,SC

    WHERE C.[C#]=SC.[C#]

        AND CN=N'税收基础')

2. 使用标准SQL嵌套语句查询选修课程编号为’C2’的学员姓名和所属单位

--实现代码：

SELECT S.SN,S.SD FROM S,SC

WHERE S.[S#]=SC.[S#]

    AND SC.[C#]='C2'

3. 使用标准SQL嵌套语句查询不选修课程编号为’C5’的学员姓名和所属单位

--实现代码：

SELECT SN,SD FROM S

WHERE [S#] NOT IN(

    SELECT [S#] FROM SC

    WHERE [C#]='C5')

4. 使用标准SQL嵌套语句查询选修全部课程的学员姓名和所属单位

--实现代码：

SELECT SN,SD FROM S

WHERE [S#] IN(

    SELECT [S#] FROM SC

        RIGHT JOIN C ON SC.[C#]=C.[C#]

    GROUP BY [S#]

    HAVING COUNT(\*)=COUNT(DISTINCT [S#]))

5. 查询选修了课程的学员人数

--实现代码：

SELECT 学员人数=COUNT(DISTINCT [S#]) FROM SC

6. 查询选修课程超过5门的学员学号和所属单位

--实现代码：

SELECT SN,SD FROM S

WHERE [S#] IN(

    SELECT [S#] FROM SC

    GROUP BY [S#]

    HAVING COUNT(DISTINCT [C#])>5)

if not object\_id('cj')is null

   drop table cj

go

create table cj(stuName nvarchar(10),KCM nvarchar(10),cj numeric(5,2))

insert into cj select '张三','语文',98

union select '李四','语文',89

union select '王五','语文',67

union select '周攻','语文',56

union select '张三','数学',89

union select '李四','数学',78

union select '王五','数学',90

union select '周攻','数学',87

方法一：

select stuname from

    (select stuName,kcm,(select count(\*) from cj where stuname!=a.stuname and kcm=a.kcm and cj>a.cj) cnt from cj a) x

    group by stuname having max(cnt)<=1

go

方法二：

SELECT stuname FROM cj1 a

where cj IN(SELECT TOP 2 cj FROM cj1 WHERE kcm=a.kcm ORDER BY cj desc)

GROUP BY stuname HAVING(count(1)>1)

方法三：

select distinct stuname from cj a

    where not exists(select kcm from cj b where a.stuname=stuname

                and (select count(\*) from cj where kcm=b.kcm and stuname!=a.stuname and cj>b.cj)>1)