

四、帶有EXISTS的子查詢

EXISTS代表存在量詞，帶有**EXISTS**的子查詢不返回任何數據，只產生邏輯真值“true”或邏輯假值“false”

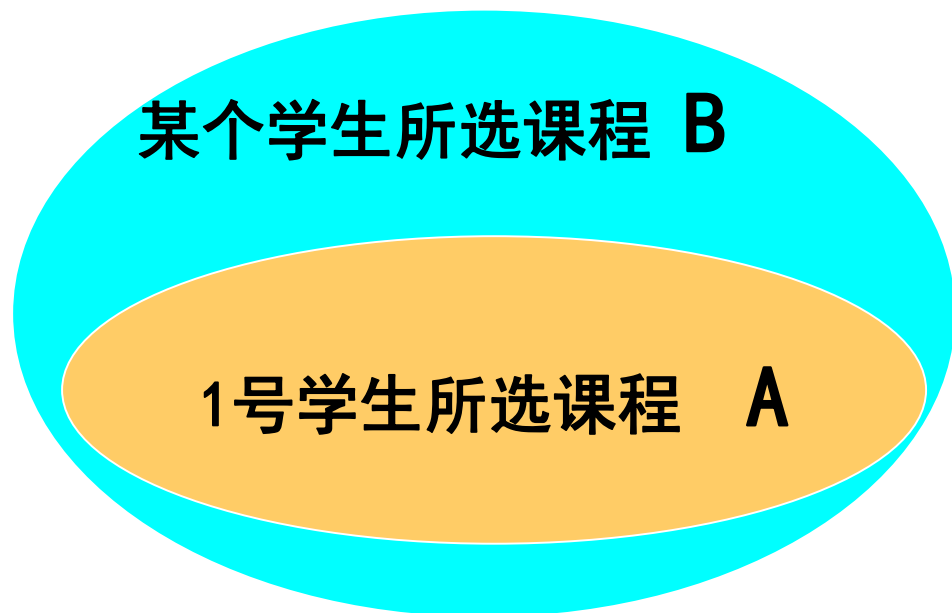
查询同时选修了1号和2号课程的学生学号

```
select Sno  
from SC as A  
where Cno='1'  
and Exists (  
    select *  
    from SC as B  
    where Cno='2'  
    and A.Sno = B.Sno)
```

查询选修了1号课而没有选修2号课的学生学号

```
select Sno  
from SC as A  
where Cno='1'  
and Not Exists (  
        select *  
        from SC as B  
        where Cno='2'  
        and A.Sno = B.Sno)
```

查询至少选修了1号学生所选全部课程的学生编号



应该满足：

Not Exists (A-B) 为真

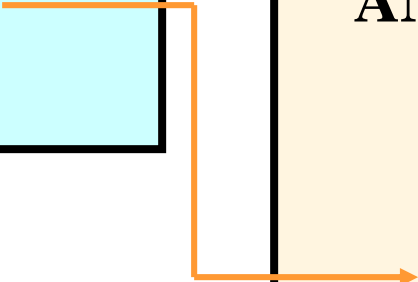
```
SELECT Cno
FROM SC
WHERE Sno='1'
AND Cno not IN
(SELECT Cno
FROM SC
WHERE Sno= 'x')
```

```
SELECT Cno
FROM SC as A
WHERE Sno='1'
AND not Exists
(SELECT Cno
FROM SC as B
WHERE Sno= 'x'
AND A.Cno= B.Cno)
```

第一步：A-B的表示

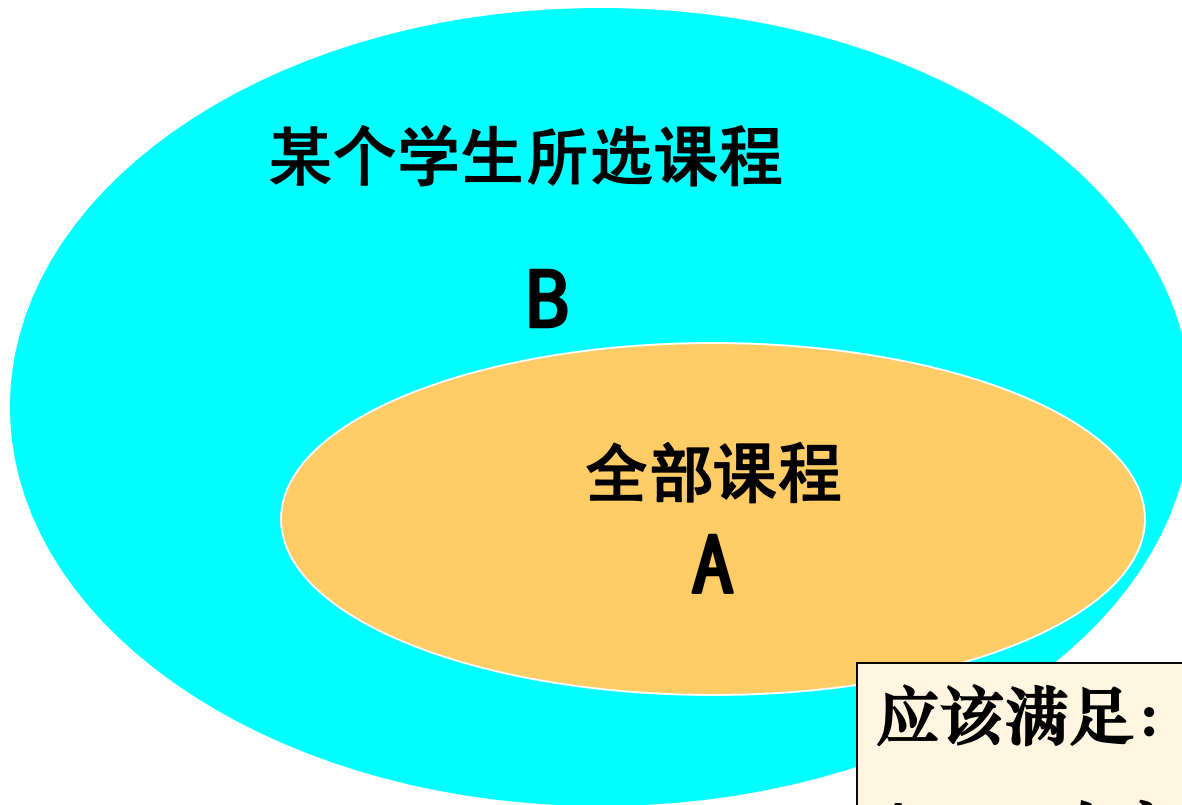
第二步：Not Exists (A - B) 的表示

```
SELECT Cno
FROM SC as A
WHERE Sno='1'
AND not Exists
(SELECT Cno
FROM SC as B
WHERE Sno= 'x'
AND A.Cno= B.Cno)
```



```
SELECT distinct Sno
FROM SC as C
WHERE Not Exists
(SELECT *
FROM SC as A
WHERE Sno='1'
AND not Exists
(SELECT *
FROM SC as B
WHERE C.Sno = B.Sno
AND A.Cno = B.Cno))
```

查询选修了全部课程的学生编号



应该满足：

$A - B$ 为空

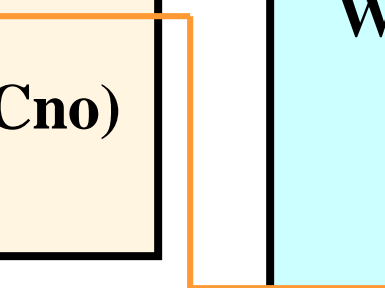
即：Not Exists ($A - B$) 为真

第一步：A-B的表示

```
SELECT Cno
FROM Course
WHERE not Exists
      (SELECT Cno
       FROM SC
       WHERE Sno= 'x'
        AND Cno=Course.Cno)
```


第二步：Not Exists (A - B) 的表示

```
SELECT Cno
FROM Course
WHERE not Exists
  (SELECT Cno
   FROM SC
   WHERE Sno = 'x'
    AND Cno = Course.Cno)
```



```
SELECT Sno
FROM Student
WHERE not Exists
  (SELECT *
   FROM Course
   WHERE not Exists
     (SELECT *
      FROM SC
      WHERE Sno = Student.Sno
       AND Cno = Course.Cno))
```

查询被所有学生选修的课程编号

```
SELECT Cno
FROM   Course
WHERE  not Exists
(SELECT Sno
     FROM   Student
     WHERE  not Exists
           (SELECT Sno
            FROM   SC
            WHERE Course.Cno=Cno
            AND   Student.Sno=Sno))
```

学生 (学号、姓名、性别、院系)

课程 (课程号、课程名)

学习 (学号、课程号、成绩)

讲授 (课程号、教师号、教师姓名)

- 1 找出计算机系女同学名单
- 2 求选修课程号为2的学生名单
- 3 求选修数据库课程的学生姓名及院系
- 4 求选修了张宁老师所授全部课程的学生名单
- 5 求选修了张宁老师所授课程的学生名单
- 6 没有选修任何课程的学生名单及院系
- 7 没有被任何人选修的课程名

第四节 聚集函数

COUNT ([DISTINCT ALL] *)	统计元组个数
COUNT ([DISTINCT ALL] 列名)	一列中值的个数
SUM ([DISTINCT ALL] 列名)	按列求和
AVG ([DISTINCT ALL] 列名)	按列求平均值
MAX ([DISTINCT ALL] 列名)	求一列中最大值
MIN ([DISTINCT ALL] 列名)	求一列中最小值

```
SELECT    Count (*)  
FROM      学生;
```

```
SELECT    AVG (年龄)  
FROM      学生;
```

```
SELECT    MAX (年龄)  
FROM      学生;
```

```
SELECT 课程号, Count(学号)  
FROM 学习  
GROUP BY 课程号;
```

```
SELECT 学号, Count(课程号)  
as 不及格门次  
FROM 学习  
WHERE 成绩 < 60  
GROUP BY 学号  
HAVING Count(课程号) > 1;
```

WHERE与HAVING
的区别在于作
用对象不同。
WHERE作用于表，
选择满足条件
的元组。
HAVING作用于
组，从中选择
满足条件的组。

3.4 数据更新

一、插入数据

1 插入单个元组

INSERT

INTO <表名> [(<属性列1>[, <属性列2>...])]

VALUES (<常量1>[, ,<常量2>]...);

2 插入子查询结果

INSERT

INTO <表名> [(<属性列1>[, <属性列2>...])]

子查询;

```
INSERT
```

```
INTO Student (Sno,Sname,Ssex,Sage,Sdept)
```

```
VALUES('4','丽丽' , '女' ,19,'计算机' );
```

指定列名

```
INSERT
```

```
INTO Student
```

```
VALUES('4','丽丽' , '女' ,19,'计算机' );
```

不指定
列名

对每一个系，求学生的平均年龄，并存入数据库

```
CREATE TABLE Dept_age  
(Sdept CHAR(15),  
Avg_age SMALLINT);
```

建立新表

```
INSERT  
INTO Dept_age (Sdept, Avg_age)  
SELECT Sdept, AVG(Sage)  
FROM Student  
GROUP BY Sdept;
```

插入数据

二、删除数据

删除语句的一般格式:

DELETE

FROM <表名>

[WHERE <条件>];

DELETE语句删除表中的数据，
但表的定义仍然留在数据字典中

删除学号为1号的学生记录

```
DELETE FROM Student  
WHERE Sno='1';
```

删除单个
元组

删除计算机学院所有学生的选课记录

```
DELETE FROM SC  
WHERE Sno IN  
(SELECT Sno  
FROM Student  
WHERE Sdept='计算机' );
```

带子查询
的修改语
句

三、修改数据

修改操作语句的一般格式：

UPDATE <表名>

SET <列名>=<表达式> [, <列名>=<表达式>]
>]...

[WHERE <条件>];

将1号学生的年龄改为22岁

```
UPDATE Student  
SET Sage=22  
WHERE Sno='1';
```

修改单个
元组

将所有学生的年龄都增加1岁

```
UPDATE Student  
SET Sage=Sage+1;
```

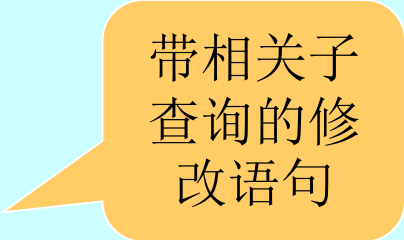
修改多个
元组

将计算机学院全体学生的成绩置零

```
UPDATE SC  
SET Grade=0  
WHERE Sno IN  
  (SELECT Sno  
   FROM Student  
   WHERE Sdept='计算机' );
```

带子查询
的修改语
句

```
UPDATE SC
SET Grade=0
WHERE 'CS' =
(SELECT Sdept
FROM Student
WHERE Student.Sno=SC.Sno);
```



带相关子
查询的修
改语句