

# 数据库实验

## 第一次实验——SQL 语言中的单表查询与连接查询

曹广杰

数据科学与计算机学院

2017/9/30

任课教师：刘玉葆。

关键词：数据库；SQL 语言；连接查询；

## Contents

Task1 .....	1
实验结果: .....	2
Task2 .....	2
实验结果 .....	3
Task3 .....	3
实验结果: .....	4
Task4 .....	4
实验结果: .....	4
Task5 .....	4
实验结果: .....	5
Task6 .....	5
实验结果: .....	6
Task7 .....	6
实验结果: .....	6
Task8 .....	7
实验结果: .....	7
Task9 .....	7
实验结果: .....	8
Task10 .....	8
实验结果: .....	8

本次实验基于 school 数据,联系 SQL 查询语言进行实验。该数据文件由 4 种关系构成,分别为 courses、students、choices 和 teachers;

## Task1

要求: 查询选修 C++课程的成绩比姓名为 ZNK00 的学生高的所有学生的编号和姓名。

-- 查找 c++科目的成绩

```
select students.sid, score
from CHOICES, STUDENTS, COURSES
where STUDENTS.sid = CHOICES.sid
and COURSES.cid = CHOICES.cid
```

```

and cname = 'c++'

-- 分数比 znkoo 高的限定条件

and score >

    (select score -- znkoo 的分数

    from CHOICES, STUDENTS, COURSES
    where STUDENTS.sid = CHOICES.sid
    and COURSES.cid = CHOICES.cid
    and sname = 'znkoo'
    and cname = 'c++')
order by score

```

句法：

1. 结构：此处使用子查询语言，在表达 znkoo 的分数时候，由于根据学生姓名不能直接获得其分数，故而需要查询。在这种条件下，使用子查询语言；
2. 对于子查询语句，可以使用在 where 中附加的条件语句限定，但是在添加参与查询的对象时，有时候系统需要查询几个对象交集的笛卡尔集，这使得计算复杂度直线上升。而使用子查询语言的时候，则分割为两个查询任务，此时，要注意子查询语言的返回值应该有且只有明确的一个值。

## 实验结果：

结果		消息
	sid	score
1	816083740	83
2	801957670	83
3	824234651	83
4	896946689	83
5	844645359	83
6	805319995	83
7	815392783	83
8	827802499	83
9	871667680	83
10	863758872	83

✓ 查询已成功执行。

## Task2

要求：找出和学生 883794999 或学生 850955252 的年级一样的学生的姓名。

```

-- 其中一个所在的年级

select sname, grade
from STUDENTS
where grade = (
    select grade

```

```

        from STUDENTS
        where sid = '883794999')
union
-- 另一个所在的年级

select sname, grade
from STUDENTS
where grade = (
    select grade
    from STUDENTS
    where sid = '850955252')

```

查询两个不同的条件，不应该使用‘或’语言——带有比较运算符的子查询，该子查询必须返回单值，否则引起编译错误。所以，使用或语言的时候，返回值会不止一个，查询系统无从辨认。

## 实验结果

结果		消息
	sname	grade
1	aaaaadsgr	2001
2	aabskm	2001
3	aacef	2001
4	aactnzls	2001
5	aactzdn	1997
6	aagendsis	2001
7	aagtfy	1997
8	aahklft	1997
9	aanjnthr	1997
10	aahdeh	1997

✓ 查询已成功执行。

查询结果显示，两个年份分别为 2001 & 1997。

## Task3

要求：查询没有选修 Java 的学生名称。

```

-- 查询所有的学生

select sid
from STUDENTS

-- 排除选择 Java 的学生

where sid not in(

    select sid -- 选择 Java

    from CHOICES, COURSES

```

```
where CHOICES.cid = COURSES.cid  
and COURSES.cname = 'java')
```

使用 not in 语句，表明“不在……范围中”

## 实验结果：

结果		消息
	sid	
1	800006682	
2	800006941	
3	800009249	
4	800014991	
5	800016416	

✓ 查询已成功执行。

## Task4

要求：找出课时最少的课程的详细信息。

```
select *  
from COURSES  
where COURSES.hour <=  
-- 子查询，小于每一个  
    all(select hour  
        from COURSES  
        where hour is not null)
```

## 实验结果：

结果				消息
	cid	cname	hour	
1	10024	use case	18	
2	10034	windows	18	

✓ 查询已成功执行。

可以看到，最少课时为 18。

## Task5

要求：查询工资最高的教师的编号和开设的课程号。

```
-- 教师 ID 与 课程 ID 一一对应  
select TEACHERS.tid, cid
```

```

from TEACHERS, CHOICES
where TEACHERS.tid = CHOICES.tid
-- 限定条件：工资最高
and salary >= all(
    select salary
    from TEACHERS
    where salary is not null)

```

句法：

条件 All（集合），表示集合内的所有数据都满足条件。

**实验结果：**

结果		
	tid	cid
1	204711560	10032
2	277877392	10041
3	204711560	10001
4	204711560	10026
5	287866460	10005

✓ 查询已成功执行。

## Task6


要求：找出选修课程 ERP 成绩最高的学生编号。

```

-- 成绩信息与课程信息锁定
select sid, score
from CHOICES, COURSES
where CHOICES.cid = COURSES.cid
and COURSES.cname = 'erp'
-- 大于所有科目最高成绩，
-- 则必然是 erp 的最高成绩
and score >= all
    (select score
    from CHOICES
    where score is not null)

```

## 实验结果：



	sid	score
1	862654622	99
2	839342232	99
3	865296034	99
4	843643589	99
5	896273784	99

查询已成功执行。

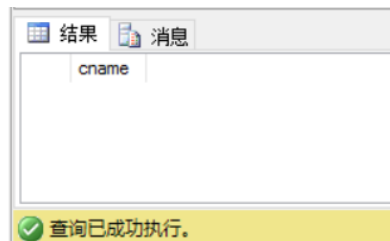
## Task7

要求：查询没有学生选修的课程名称。

```
select cname
from COURSES
where cid not in
      (select cid
       from CHOICES)
```

查询选课列表中不存在的课程。

## 实验结果：

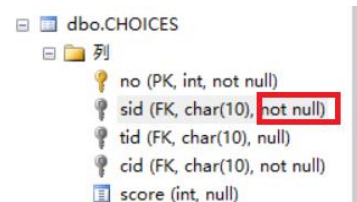


cname
-------

查询已成功执行。

查询学生的考核信息：

现在显示结果是，所有的课程都出现在选课列表中。而选课列表中，每一门课都有学生选修。



列
no (PK, int, not null)
sid (FK, char(10), not null)
tid (FK, char(10), null)
cid (FK, char(10), not null)
score (int, null)

这就说明，不存在没有学生选修的课程。

即：

1. 所有课程都在选课列表中；

2. 选课列表的课都有学生选修；

则，所有课程都有学生选修——即，不存在没有学生选修的课程。

## Task8

要求：查询讲授课程 UML 的教师所讲授的所有课程名称。

```
-- 课程 ID 与老师 ID 对应
select cname
from COURSES, CHOICES
where CHOICES.cid = COURSES.cid
-- 教授 UML 课程的老师 ID
and CHOICES.tid =
    ANY(select tid
        from CHOICES
-- UML 课程 ID
        where cid =
            (select cid
             from COURSES
             where cname = 'uml'))
```

实验结果：

结果	
	cname
4	information security
5	operating system
6	compiling principle
7	struts
8	computer storage

✓ 查询已成功执行。

关联嵌套的信息查询，需要的信息都不易获得，在此前提下使用子查询语句逐一获得需要的信息即可，注意使用 any/all/not in 等关系谓词。

## Task9

要求：使用集合交运算，查询既选修了 database 又选修了 UML 课程的学生编号。

```
-- database 的查询
select sid
from CHOICES, COURSES
where cname = 'database'
and CHOICES.cid = COURSES.cid
intersect -- 巴啦啦能量！连接！
-- UML 的查询
select sid
from CHOICES, COURSES
where cname = 'uml'
```

```
where cname = 'uml'
and CHOICES.cid = COURSES.cid
```

实验结果：

	sid
1	822863200
2	862162560
3	846105377
4	885608789
5	886966253

✓ 查询已成功执行。

实际上就是分别查询，再使用一个连接词。

## Task10

要求：使用集合减运算，查询选修了 database 却没有选修 UML 课程的学生编号。

```
-- database 的查询
select sid
from CHOICES, COURSES
where cname = 'database'
and CHOICES.cid = COURSES.cid
except -- 巴啦啦能量！减法！
-- UML 的查询
select sid
from CHOICES, COURSES
where cname = 'uml'
and CHOICES.cid = COURSES.cid
```

表明，选择了 database，但是没有选择 UML 的学生而已。对此，使用 except 语句，进行子集的分割清除。

实验结果：

	sid
1	803870334
2	815496921
3	815855012
4	856608411
5	866200069

✓ 查询已成功执行。