《数据库》课程实践报告

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 院、系 | 计算机学院 | | 年级专业 | 21计科 | 姓名 | 方浩楠 | 学号 | 2127405048 |
| 实验布置日期 | |  | | 提交  日期 |  | | 成绩 |  |

# 实验五

## 实验1.创建表

**目的：**1 掌握利用SQL语言创建表的方法。

2 sp\_help 命令

**要求：**1 创建表 2 修改表结构 3 删除表

**一．新建数据库school\_DB,在该数据库中完成以下操作：**

**二．写出使用 Create Table 语句创建表 student , sc,course 的SQL语句。**

学生表、课程表、选课表属于数据库 School ，其各自得数据结构如下：

学生 Student (Sno,Sname,Ssex,Sage,Sdept)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 序号 | 列名 | 含义 | 数据类型 | 长度 |
| 1 | Sno | 学号 | 字符型(char) | 6 |
| 2 | Sname | 姓名 | 字符型(varchar) | 8 |
| 3 | Ssex | 性别 | 字符型(char) | 2 |
| 4 | Sage | 年龄 | 整数 (smallint) |  |
| 5 | sdept | 系科 | 字符型(varchar) | 15 |

课程表 course(Cno,Cname,Cpno,Ccredit)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 序号 | 列名 | 含义 | 数据类型 | 长度 |
| 1 | Cno | 课程号 | 字符型(char) | 4 |
| 2 | cname | 课程名 | 字符型(varchar) | 20 |
| 3 | Cpno | 先修课 | 字符型(char) | 4 |
| 4 | Ccredit | 学分 | 短整数 (tinyint) |  |

学生选课 SC(Sno,Cno,Grade)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 序号 | 列名 | 含义 | 数据类型 | 长度 |
| 1 | Sno | 学号 | 字符型(char) | 6 |
| 2 | Cno | 课程名 | 字符型(char) | 4 |
| 3 | Grade | 成绩 | 小数(decimal) | 12,1 |

create table Student(

Sno char(6) not null,

Sname varchar(8) not null,

Ssex char(2) not null,

Sage smallint not null,

Sdept varchar(15) not null

)

create table Course(

Cno char(4) not null,

Cname varchar(20) not null,

Cpno char(4) not null,

Ccredit tinyint not null

)

create table SC(

Sno char(6) not null,

Cno char(4) not null,

Grade decimal(12,1) not null

)

文本

中度可信度描述已自动生成

**三 使用 SP\_HELP 查看表 student 的表结构**

表格

描述已自动生成

**四 利用 sql 语句表结构修改**

1 在student 表中添加列：

家庭地址 address 长度为 60 varchar 型

入学日期 inDate 日期型

完成后用sp\_help 查看是否成功。

2 将家庭地址 address 长度为 50

完成后用sp\_help 查看是否成功。

3 删除 student 表的 inDate 列

alter table Student add address varchar(60),inDate date;

alter table Student alter column address varchar(50)

alter table Student drop column inDate

**五 删除表**

1 删除表 sc

2 删除表 student

3 删除表 course

drop table SC

drop table Student

drop table Course

## 实验2.创建数据完整性

**目的：**1掌握创建数据完整性约束的命令。

2 掌握完整性约束的修改、删除。

**要求：**1 能建立完整性约束 2 修改完整性约束 3 删除完整性约束

**一 写出带有完整性约束的 Create Table 命令建立表student、course、sc 。要求：**

1 Student表的主码：sno

student 的约束：

* 姓名不可为空，且唯一
* 性别 不能为空且取值范围为{男，女}
* 年龄大于16岁
* sdept 默认为 ‘JSJ’ 系

2 Course表的主码：cno

course 的约束：

* Ccredit 取值范围{ 0 ,1,2,3,4,5 }
* 课程表的每一行的 Cno 与 cpno 不可相同

3 Sc表的主码：sno，cno 。主码名为 PK\_SC

Sc的外码：

* 外码：SC 表的sno 参照表 student 的 sno
* 外码：sc 表的Cno 参照表 course 的 cno

4 把上述创建表的sql 语句的脚本存储到文件 createSchool.sql 。

--实验二

--1.写出带有完整性约束的 Create Table 命令建立表student、course、sc

create table Student(

Sno char(6) not null,

Sname varchar(8) not null,

Ssex char(2) not null,

Sage smallint not null,

Sdept varchar(15) not null default('JSJ'),

primary key(Sno),

unique(Sname),

check(Ssex in('男','女')),

check(Sage>16),

)

create table Course(

Cno char(4) not null,

Cname varchar(20) not null,

Cpno char(4),

Ccredit tinyint not null check(ccredit IN (0, 1, 2, 3, 4, 5)),

primary key(Cno),

unique(Cno,Cpno)

)

create table SC(

Sno char(6) not null,

Cno char(4) not null,

Grade decimal(12,1) not null,

constraint PK\_SC primary key(Sno,Cno),

foreign key(sno) references student(sno),

foreign key(cno) references course(cno)

) 日程表

描述已自动生成

**二 使用 SP\_HELP 查看表 student 的主码名，约束名，并记录。**

使用 SP\_HELP 查看表 sc 的主码名，外码名，并记录

--2.使用 SP\_HELP 查看表 student 的主码名，约束名，并记录。使用 SP\_HELP 查看表 sc 的主码名，外码名，并记录。

exec sp\_help 'student'

exec sp\_help 'Course'

exec sp\_help 'sc'

图形用户界面, 应用程序, 表格

描述已自动生成

**三 利用alter table 添加、删除完整性约束**

1 删除 SC 的主码，sc表的主码名为 pk\_sc

2 删除 SC 表**参照 course 表**的外码。。

3 添加 SC 表的主码。主码名为 PK\_SC

4 添加 SC 表的Cno 的外码 ，参照表Course的Cno.

5加自定义约束：表 SC的成绩只能在 0 – 100分之间。

--3.利用alter table 添加、删除完整性约束

alter table sc

drop constraint PK\_SC

alter table sc

drop constraint FK\_\_SC\_\_Cno\_\_0F624AF8

alter table sc

add constraint PK\_SC primary key(Sno,Cno)

alter table sc

add foreign key(Cno) references Course(Cno)

alter table sc

add check(grade > 0 and grade < 100)

**四 使用**

select \* from student 查看信息

select \* from course 查看信息

select \* from sc 查看信息

图片包含 图形用户界面

描述已自动生成

## 实验3.数据完整性试验

**目的：**1理解实体完整性、参照完整性、用户自定义完整性的作用

2 特别掌握外码的作用。

**要求：**记录试验中遇到的问题，并写出原因。

**一 实体完整性**

1 student 表数据输入

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 学号 | 姓名 | 性别 | 年龄 | 系科 |
| 3001 | 赵达 | 男 | 20 | SX |
| 3002 | 杨丽 | 女 | 21 | JSJ |
| 3001 | 李寅 | 女 | 21 | SX |
|  |  |  |  |  |

* 输入上述数据，记录出现的问题，说明原因。
* select \* from student 查看你输入了几行数据。

2 course 表数据的输入

|  |  |  |  |
| --- | --- | --- | --- |
| Cno | Cname | Cpno | Ccredit |
| 1081 | 电子商务 |  | 4 |

3 SC 表数据的输入

|  |  |  |
| --- | --- | --- |
| Sno | Cno | Grade |
| 3001 | 1081 | 90 |
| 3001 | 1081 | 79 |
|  |  |  |

输入上述数据，记录出现的问题，说明原因。

4. 说明实体完整性的含义，说明什么情况下会违反实体完整性，以及违约处理情况。可截图辅以说明。

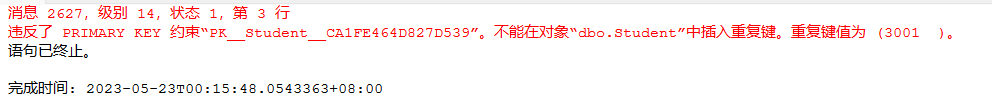
1.

insert into student values

(3001,'赵达','男',20,'SX'),

(3002,'杨丽','女',21,'JSJ'),

(3001,'李寅','女',21,'SX');



2. insert into course values(1081,'电子商务',null,4);

3.

insert into sc values

(3001,1081,90),

(3001,1081,79);

**二 用户自定义完整性约束**

表student 有用户自定义约束：

**性别 不能为空且取值范围为{男，女}**

**年龄大于16岁**

表course 的自定义约束：

**Ccredit 取值范围{ 0 ,1,2,3,4,5 }**

**课程表的每一行的 Cno 与 cpno 不可相同**

1 student 表数据输入

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 学号 | 姓名 | 性别 | 年龄 | 系科 |
| 3005 | 赵达 | 男 | 14 | SX |
| 3006 | 杨丽 | 南 | 21 | JSJ |

* 输入上述数据，记录出现的问题，说明原因。
* select \* from student 查看你输入了那些数据。

2 course 表数据的输入

|  |  |  |  |
| --- | --- | --- | --- |
| Cno | Cname | Cpno | Ccredit |
| 1085 | C++ |  | 9 |
| 1086 | 语文 | 1086 | 3 |

* 输入上述数据，记录出现的问题，说明原因。
* select \* from student 查看你输入了那些数据。

3 SC 表数据的输入

|  |  |  |
| --- | --- | --- |
| Sno | Cno | Grade |
| 3002 | 1081 | 128 |
|  |  |  |

* 输入上述数据，记录出现的问题，说明原因。
* select \* from student 查看你输入了那些数据。

4.说明用户定义完整性的含义，基于这三张表，举例说明什么情况下会违反用户定义完整性，以及违约处理情况。可截图辅以说明。

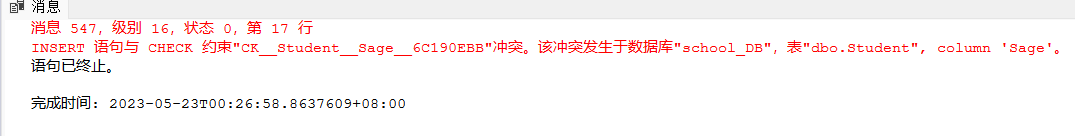
1.

insert into student values

(3005,'赵达','男',14,'SX'),

(3006,'杨丽','南',21,'JSJ');

select \* from student;



图形用户界面, 应用程序, 表格

描述已自动生成

2.

insert into course values

(1085,'C++',null,9),

(1086,'语文','1086',3);

图片包含 图形用户界面

描述已自动生成

图形用户界面, 应用程序

描述已自动生成

3.

insert into sc values

(3002,1081,128);

图片包含 文本

描述已自动生成

图形用户界面, 应用程序, Word

描述已自动生成

4.

当数据库中某一实体所包含的属性值出现于约束相违背时，实体完整性就会受到破坏。这种情况下，通常会采取限制用户访问权限，禁止用户添加和修改非法数据，并且及时进行数据修复。

**三 参照完整性约束**

* 掌握表之间建立外码后，对被参照表的如下操作会有何影响：
* 对参照表添加新行、删除行、修改外码值有何影响？
* 掌握级联修改、级联删除的概念。

**注意：**

**表SC的 Sno是外码，参照student的sno。**

**表SC的 Cno是外码，参照course的cno。**

**1 输入实验前的数据**

学生表 Student

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Sno | Sname | Ssex | Sage | Sdept |
| 4001 | 赵尹 | 男 | 20 | SX |
| 4002 | 杨开 | 女 | 20 | JSJ |

课程表 course

|  |  |  |  |
| --- | --- | --- | --- |
| Cno | Cname | Cpno | Ccredit |
| 1088 | Java |  | 5 |
| 1089 | 数学 |  | 3 |

学生选课 SC

|  |  |  |
| --- | --- | --- |
| Sno | Cno | grade |
| 4001 | 1088 | 90 |
| 4002 | 1088 | 86 |
|  |  |  |

**2 试验过程**

**1) 在SC表中添加新行：**

|  |  |  |
| --- | --- | --- |
| Sno | Cno | Grade |
| 4001 | 1066 | 76 |

记录试验结果.,写出出现此结果的原因.

**2) 在student表中添加新行**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Sno | Sname | Ssex | Sage | Sdept |
| 4003 | 赵辉 | 男 | 21 | SX |

记录试验结果.,写出出现此结果的原因.

**3) 删除student 表的 4001 ，4002学生**

记录试验结果.,写出出现此结果的原因.

思考：

* 删除SC表的记录有限制吗？
* 采取什么技术能使不能成功执行的命令变得可以执行，且使数据库保持数据完整性。

**4) 把 student 表的学号 4003 改为 4018 , 4001改为4021。**

记录试验结果.,写出出现此结果的原因.

思考：采取什么技术能使本题不能执行的命令可以执行，且使数据库保持数据完整性。

**5) 把sc表中的如下记录的学号从4001改为4011。**

|  |  |  |
| --- | --- | --- |
| Sno | Cno | Grade |
| 4001 | 1088 | 90 |

记录试验结果.,写出出现此结果的原因.

* 如不成功，则可以采取什么方法来实现此要求。
* 如不成功，那么把4001修改为4003，能成功吗？

6）说明参照完整性的含义，基于这三张表，举例说明哪些情况下会违反参照完整性，每种情况举一例；以及违约处理情况（含三种违约处理）。可截图辅以说明。

1.

insert into student values

(4001,'赵尹','男',20,'SX'),

(4002,'杨开','女',20,'JSJ');

insert into course values

(1088,'Java',null,5),

(1089,'数学',null,3)

insert into sc values

(4001,1088,90),

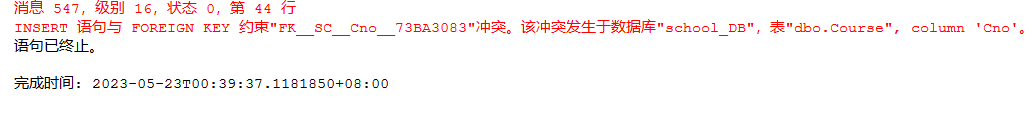
(4002,1088,86)

2.

（1）

insert into sc values

(4001,1066,76)



（2）

insert into student values

(4003,'赵辉','男',21,'SX')

文本

低可信度描述已自动生成

（3）

delete from student

where sno = '4001' or sno = '4002'

文本

中度可信度描述已自动生成

（4）

update student

set sno = '4018'

where sno = '4003';

update student

set sno = '4021'

where sno = '4001'

图形用户界面, 应用程序

描述已自动生成

学号为4001的学生的信息已经被删除，因此无法被update

（5）

update sc

set sno = '4001'

where sno = '4001' and cno = '1088' and grade = 90;

（6）

当外码输入自己参照的另一个表格中的没有的数据时，会报错；当外码已经记录下数据，然后在参考的表格中修改列值，也会报错；当外码已经记录下数据，然后在参考的表格中删除对应的行，也会报错。

设置空值|联级处理(cascade）|拒绝处理(No Action).

# 实验6

# 第二部分 SQL语言

## 实验四 建表语句和完整性约束

**目的：**1 掌握利用SQL语言创建表的方法。

2 sp\_help 命令

**要求：**1 创建表 2 修改表结构 3 删除表

**一．创建grade数据库，在grade数据库中使用Create Table 语句创建以下表：（参考表中需要输入的数据，自定表中各列的数据类型）**

1）DEP（deptid，depname，depzr）

含义：系表（系号，系名，系主任）

要求：设置主键.

2） 建立S表并为该表设置主键，设age的默认值为20；姓名不可为空，且唯一。

S(sno,sname,age,deptid)

含义：学生表(学号,姓名,年龄，系号)

3）建立SC表并为该表设置主键和外键，要求score应该在0-100之间；

SC(sno,cno,score)

含义：选修表(学号,课程号,成绩)

4）建立C表；

Course(cno,cname,pcno)

含义：课程表(课程号,课程名,先行课程号)

要求：

1. 设置主键和外键。

2. 课程表的每一行的 Cno 与 cpno 不可相同

5）T(TNO,TN,PROF, SA,deptid)

含义：教师表(教师号,教师姓名,职称,工资,系号)

要求：设置主键和外键 工资大于等于1000；

6）TC(TNO,CNO,book)

含义：教课表（教师号，课程号，所用书）

要求：设置主键和外键；

**二．把创建表的sql 语句的脚本存储到文件grade.sql**

grade.sql:

create database grade  
  
create table DEP(  
 DEPtid char(10) not null,  
 DEPname char(10) not null,  
 DEPzr char(10) not null,  
   
 primary key(DEPtid)  
)  
  
create table S(  
 Sno char(10) not null,  
 Sname char(10) not null,  
 age int default(20),  
 DEPtid char(10) not null,  
  
 primary key(Sno),  
 unique(Sname)  
)  
  
create table Course(  
 Cno char(10) not null,  
 Cname char(10) not null,  
 PCno char(10),  
  
 primary key(Cno),  
 unique(Cno,PCno)  
)  
  
create table SC(  
 Sno char(10) not null,  
 Cno char(10) not null,  
 score int check(score < 100 and score > 0),  
  
 primary key(Sno,Cno),  
 foreign key(Sno) references S(Sno),  
 foreign key(Cno) references Course(Cno)  
)  
  
create table T(  
 Tno char(10) not null,  
 Tn char(10) not null,  
 PROF char(10) not null,  
 SA DECIMAL(10, 2) not null,  
 DEPtid char(10),  
  
 primary key(Tno),  
 foreign key(DEPtid) references DEP(DEPtid),  
 check(SA >= 1000)  
)  
  
create table TC(  
 TNO char(10) not null,  
 CNO char(10) not null,  
 book char(10),  
  
 primary key(TNO,CNO),  
 foreign key(TNO) references T(tno),  
 foreign key(cno) references Course(cno)  
  
)

**三．生成grade数据库中6张表的数据库关系图**

图形用户界面, 图示, 表格

描述已自动生成

**四．在表中输入以下数据：**

Dep表：

01 计算机系 张荣

02 信息管理系 李力

S表：

s1 王立 16 01

s2 李楠 18 02

C表：

c1 计算机原理 NULL

c2 网络工程 c1

c3 数据库原理 c2

SC表：

s1 c1 88

s1 c2 70

s2 c1 NULL

s2 c2 56

s1 c3 89

T表：

t1 张荣 讲师 1800.00 01

t2 李力 助教 1200.00 02

t3 刘伟 副教授 2000.00 NULL

t4 刘明 助教 1000.00 NULL

TC表：

t1 c1 计算机原理

t2 c2 网络工程

t3 c1 计算机原理

t4 c2 网络工程

insert into dep (deptid, depname, depzr) values  
 (01, '计算机系', '张荣'),  
 (02, '信息管理系', '李力');  
  
insert into s (sno, sname, age, deptid) values  
 ('s1', '王立', 16, 01),  
 ('s2', '李楠', 18, 02);  
  
insert into course(cno, cname, pcno) values  
 ('c1', '计算机原理', null),  
 ('c2', '网络工程', 'c1'),  
 ('c3', '数据库原理', 'c2');  
  
insert into sc (sno, cno, score) values  
 ('s1', 'c1', 88),  
 ('s1', 'c2', 70),  
 ('s2', 'c1', null),  
 ('s2', 'c2', 56),  
 ('s1', 'c3', 89);  
  
insert into t (tno, tn, prof, sa, deptid) values  
 ('t1', '张荣', '讲师', 1800.00, 01),  
 ('t2', '李力', '助教', 1200.00, 02),  
 ('t3', '刘伟', '副教授', 2000.00, null),  
 ('t4', '刘明', '助教', 1000.00, null);  
  
insert into tc (tno, cno, book) values  
 ('t1', 'c1', '计算机原理'),  
 ('t2', 'c2', '网络工程'),  
 ('t3', 'c1', '计算机原理'),  
 ('t4', 'c2', '网络工程');

**五．修改表：利用alter table 修改表结构**

1. 为S表添加外键fk\_S。

alter table S  
add constraint fk\_s foreign key(DEPtid) references DEP(deptid)

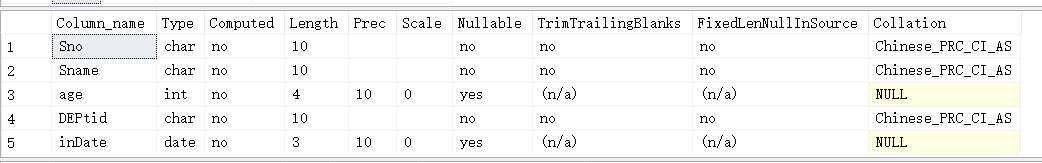
2. 为T表添加default约束：职称默认为“副教授”。

alter table t  
add constraint d\_t default '副教授' for PROF

3. 在S表中添加列：入学日期 inDate 日期型

完成后用sp\_help 查看是否成功。

alter table s  
add inDate date



4. 删除 student 表的 inDate 列

alter table s  
drop column inDate

5. 删除T表中的default约束。

alter table t  
drop constraint d\_t

思考：

1. 以grade数据库为例，分别测试参照完整性中，默认（拒绝）策略、级联策略和置空策略的效果。
2. 以grade数据库为例，说明外键的取值有几种情况？外键和所参照的主键必须是在两张表吗？

## 试验六： 查询语句

**目的: 掌握 Select 查询语句。**

参考资料“SQL.pdf”或中国大学MOOC上“第十二讲：数据查询-单表查询”完成以下查询：

**第一部分： 单表查询**

1. 在school数据库中完成以下查询（服务器上没有school数据库的，需先附加school数据库；或执行school数据库建表语句），将文件保存为：school数据查询.sql文件。

1查询年龄在19至21岁之间的女生的学号,姓名,年龄,按年龄从大到小排列。

2查询姓名中第2个字为“明”字的学生学号、性别。

3查询 1001课程没有成绩的学生学号、课程号

4查询JSJ 、SX、WL 系的年龄大于25岁的学生学号,姓名，结果按系及学号排列

5按10分制查询学生的sno,cno,10分制成绩

（1-10分 为1 ，11-20分为2 ，30-39分为3，。。。90-100为10）

6查询 student 表中的学生共分布在那几个系中。（distinct）

7查询0001号学生1001，1002课程的成绩。

school数据查询.sql

*--1.查询年龄在19至21岁之间的女生的学号,姓名,年龄,按年龄从大到小排列*select sno,sname,sage  
from student  
where ssex = '女' and sage between 19 and 21  
order by sage desc;  
  
*--2.查询姓名中第2个字为“明”字的学生学号、性别*select sno,ssex  
from student  
where sname like '\_明%';  
  
*--3.查询 1001课程没有成绩的学生学号、课程号*select sno,cno  
from sc  
where cno = '1001' and grade is null;  
  
*--4.查询JSJ 、SX、WL 系的年龄大于25岁的学生学号,姓名，结果按系及学号排列*select sno,sname  
from student  
where sdept in ('jsj','sx','wl') and sage>25  
order by sdept,sno  
  
*--5.按10分制查询学生的sno,cno,10分制成绩 （1-10分 为1 ，11-20分为2 ，30-39分为3，。。。90-100为10）*select sno,cno,  
case  
 when grade between 1 and 10 then 1  
 when grade between 11 and 20 then 2  
 when grade between 21 and 30 then 3  
 when grade between 31 and 40 then 4  
 when grade between 41 and 50 then 5  
 when grade between 51 and 60 then 6  
 when grade between 61 and 70 then 7  
 when grade between 71 and 80 then 8  
 when grade between 81 and 90 then 9  
 when grade between 91 and 100 then 10  
end as grade\_10  
from sc;  
  
*--6.查询 student 表中的学生共分布在那几个系中。（distinct）*select distinct sdept  
from student;  
  
*--7.查询0001号学生1001，1002课程的成绩。*select grade  
from sc  
where sno = '0001' and cno in('1001','1002')

1. 在SPJ数据库中，完成如下查询：（如没有SPJ数据库文件，可先按以下要求创建）

SPJ数据库，包含SB、PB，JB，SPJB表。其中供应商表SB由供应商代码SN、供应商名SNAME、所在城市CITY组成。零件表PB由零件代码PN，零件名PNAME，颜色COLOR和重量WEIGHT组成。工程项目表JB由项目代码JN，项目名JNAME，项目所在城市CITY组成。供应情况表SPJB由供应商代码SN，零件代码PN，项目代码JN，供应数量QTY组成。

**文字图案

中度可信度描述已自动生成**

创建以上四张表，数据类型参考表中数据。要求定义出主键和外键信息，另外，PB表中的WEIGHT要求大于0；SPJB表中的QTY要求大于0。

SPJ数据库中的查询：

1. 按供应商编号升序、数量降序输出SPJB信息
2. 取出s1供应商供应的最大数量、最小数量、及其两者之差，平均数量
3. 求各供应商供应零件的平均数量，并按供应商号降序排序
4. 查询供应零件总数量在800以上的工程编号和其供应的总数量
5. 查询所有工程的全部细节；
6. 查询所在城市为上海的所有工程的全部细节；
7. 查询提供零件数量大于300的供应商的编号；
8. 查询为工程J1提供零件的供应商代号：
9. 查询为工程J1提供零件P1的供应商代号；
10. 取出为工程J1或J2提供零件的供应商代号；
11. 取出由供应商S1提供零件的工程的代号；
12. 取出零件重量在10-20之间的零件信息
13. 取出城市以“北”开头的供应商的编号、名称、城市

create database SPJ  
  
CREATE TABLE SB (  
 SN CHAR(4),  
 SNAME VARCHAR(50),  
 CITY VARCHAR(50),  
 PRIMARY KEY (SN)  
);  
  
CREATE TABLE PB (  
 PN CHAR(4),  
 PNAME VARCHAR(50),  
 COLOR VARCHAR(50),  
 WEIGHT DECIMAL(10,2),  
 PRIMARY KEY (PN)  
);  
  
CREATE TABLE JB (  
 JN CHAR(4),  
 JNAME VARCHAR(50),  
 CITY VARCHAR(50),  
 PRIMARY KEY (JN)  
);  
  
CREATE TABLE SPJB (  
 SN CHAR(4),  
 PN CHAR(4),  
 JN CHAR(4),  
 QTY INT,  
 PRIMARY KEY (SN, PN, JN, QTY),  
 FOREIGN KEY (SN) REFERENCES SB (SN),  
 FOREIGN KEY (PN) REFERENCES PB (PN),  
 FOREIGN KEY (JN) REFERENCES JB (JN)  
);  
  
insert into sb values  
('S1','N1','上海'),  
('S2','N2','北京'),  
('S3','N3','北京'),  
('S4','N4','上海'),  
('S5','N5','南京');  
  
insert into pb values  
('P1','PN1','红','12'),  
('P2','PN2','绿','18'),  
('P3','PN3','蓝','20'),  
('P4','PN4','红','13'),  
('P5','PN5','蓝','11'),  
('P6','PN6','绿','15');  
  
insert into jb values  
('J1','JN1','上海'),  
('J2','JN2','广州'),  
('J3','JN3','南京'),  
('J4','JN4','南京'),  
('J5','JN5','上海'),  
('J6','JN6','武汉'),  
('J7','JN7','上海');  
  
insert into spjb values  
('S1','P1','J1',200),  
('S1','P1','J4',700),  
('S2','P3','J1',400),  
('S2','P3','J2',200),  
('S2','P3','J3',200),  
('S2','P3','J4',500),  
('S2','P3','J5',600),  
('S2','P3','J6',400),  
('S2','P3','J7',800),  
('S2','P3','J2',100),  
('S3','P3','J1',200),  
('S3','P3','J2',500),  
('S4','P6','J3',300),  
('S4','P6','J7',300),  
('S5','P2','J2',200),  
('S5','P2','J4',100),  
('S5','P5','J5',500),  
('S5','P5','J7',100),  
('S5','P6','J2',200),  
('S5','P1','J4',1000),  
('S5','P3','J4',1200),  
('S5','P4','J4',800),  
('S5','P5','J4',400),  
('S5','P6','J4',500);  
  
*--1.按供应商编号升序、数量降序输出SPJB信息*select *\**from spjb  
order by sn asc,qty desc;  
  
*--2.取出s1供应商供应的最大数量、最小数量、及其两者之差，平均数量*select *max*(qty) as '最大数量',*min*(qty) as '最小数量',(*max*(qty) - *min*(qty)) as '两者之差',*avg*(qty) as '平均数量'  
from spjb  
where sn = 's1';  
  
*--3.求各供应商供应零件的平均数量，并按供应商号降序排序*select sn,*avg*(qty) as '平均数量'  
from spjb  
group by sn  
order by sn;  
  
*--4.查询供应零件总数量在800以上的工程编号和其供应的总数量*select pn, *sum*(qty) as '总数量'  
from spjb  
group by pn  
having *sum*(qty)>800;  
  
*--5.查询所有工程的全部细节*select *\**from jb;  
  
*--6.查询所在城市为上海的所有工程的全部细节*select *\**from jb  
where city = '上海'  
  
*--7.查询提供零件数量大于300的供应商的编号*select sn  
from spjb  
group by sn  
having *sum*(qty)>300;  
  
*--8.查询为工程J1提供零件的供应商代号*select sn  
from spjb  
where jn = 'J1';  
  
*--9.查询为工程J1提供零件P1的供应商代号*select distinct sn  
from spjb  
where jn = 'J1' and pn = 'P1';  
  
*--10.取出为工程J1或J2提供零件的供应商代号*select distinct sn  
from spjb  
where jn in ('J1','J2');  
  
*--11.取出由供应商S1提供零件的工程的代号*select jn   
from spjb   
where sn='s1' group by jn  
  
*--12.取出零件重量在10-20之间的零件信息*select *\**from pb  
where weight between 10 and 20;  
  
*--13.取出城市以“北”开头的供应商的编号、名称、城市*select sn,sname,city  
from sb   
where city like'北%'

**第二部分：T-SQL**

1. **函数：**
   1. 根据文件T-SQL.pdf中“9.4系统内置函数.pdf”文件，熟悉函数的功能，特别是字符串函数和日期函数；

二） 完成以下习题：

* 1. **预测以下陈述的输出：Select Round(1234.567,1)**

1. **1234.5**
2. **1234.6**
3. **1234**
4. **1234.56**
   1. 某个高级学校以十进制数形式存储学生的考试分数。他们使用称为

float的数据类型，以便他们存储十进制值。以同样格式用以下语句输入分 数到所创建的表：

CREATE TABLE StudentMarks

(cRegistrationNo char(6) primary key,

cBatchCode char(6) not null,

fTest1 float not null,

fTest2 float not null,

fPractical float not null,

fTotal float)

显示分数的报告要舍入到最近的整数。为产生此报告应使用以下查询中哪个？

1.SELECT cRegistrationNo, round (fTotal) FROM StudentMarks

**2.SELECT cRegistrationNo, round (fTotal,0) FROM StudentMarks**

3.SELECT cRegistrationNo, round (fTotal,-1) FROM StudentMarks

4.SELECT cRegistrationNo, round (fTotal,1) FROM StudentMarks

* 1. 识别按以下格式显示当前日期的SQL语句。 dd.mm.yyyy

* 1. Select date= ‘dd.mm.yy’, getdate()
  2. Select convert(char(12),4,getdate())
  3. Select convert(char(12),getdate(),4)
  4. **Select convert(char(12),getdate(),104)**
  5. **预测以下SQL 语句的输出。**

**Select floor(1234.567)**

**1）1234.56**

**2）1234**

**3）1235**

**4) 12345.67**

* 1. **一名学生执行以下SELECT语句：**

**SELECT substring('Microsoft SQL Sever is a great product',2,4).**

**此substring 函数将返回以下串中哪一个？**

* 1. **icro**
  2. **icr**
  3. **ir**
  4. **ro**

6．

Hugh and Co中所有职工的评估信息保存在称为Appraisal的

表中。

用以下语句创建此表：

CREATE TABLE Appraisal

( cEmployeeCode char (6) not null,

dDateOfAppraisal datetime not null,

cReviewer char(15) not null,

cStatus char(3) not null)

每季度评估职工。dDateOfAppraisal属性包含最近评估的日期。

以下查询中哪一个可用来发现他下次评估的日期？

1.SELECT dateadd(qq, 3,dDateofAppraisal)FROM Appraisal

2.SELECT datepart (mm,dDateOfAppraisal) +3 FROM Appraisal

3.SELECT datepart (mm,dDateOfAppraisal ) FROM Appraisal

**4.SELECT dateadd (mm,3,dDateOfAppraisal) FROM Appraisal**

**7.预测以下SQL 语句的输出：**

**Select \* from sales where tran\_date >=dateadd(dd, -3, getdate())**

**a. 显示销售日期在当前系统日期之后3天的所有行。**

**b. 显示销售日期在当前系统日期之前3天的所有行。**

**c. 显示销售日期是当前系统日期的所有行。**

**d. 显示销售日期在当前系统日期之后3周的所有行。**

**8.预测以下SQL 语句的输出，如果给定产品的销售日期是July 13, 2000 ，**

**定单日期是July 1, 2000：**

**Select datediff(yy, sale\_dt, order\_dt) from transaction where**

**prod\_id = ‘10202’**

**A. 1**

**B. -1**

**C. 0**

**D. 13**

9.识别可用来显示当前日期的季度、月的名、年的天数 的SQL语句？

1. **Select ‘Quarter’=datepart(qq,getdate()),**

**‘Month’=datename(mm,getdate()), datepart(dy,getdate())**

B. Select ‘Quarter’=datename(qq,getdate()),

‘Month’=datepart(mm,getdate()), datepart(dd,getdate())

C. Select ‘Quarter’=datepart(qq,getdate()),

‘Month’=datename(mm,getdate()), datepart(dw,getdate())

D. Select ‘Quarter’=datepart(qq,getdate()),

‘Month’=datename(mm,getdate())

**10.预测以下SQL语句的输出，如果某产品销售的日期**

**是July 23, 2001 （2001年7月23日），**

**定单日期是 July 1, 2001（2001年7月1日）。**

**Select datediff(dd, sale\_dt, order\_dt)**

**from transaction where prod\_id = ‘10202’**

1. **22**
2. **-22**
3. **18**
4. **21**
5. **流程语句**

熟悉文件：T-SQL.pdf中“9.3常量和变量”和“9.5批处理和流程控制语句”，重点是变量部分和流程控制语句部分。

并完成以下题目：

1. 使用WHILE循环计算1-100的和

2. 使用CASE来判断当前日期是否是闰年？

3. 用T-SQL流程控制语句编写程序，求斐波那契数列中小于100的所有数（斐波那契数列为1,2,3,5,8,13，…）。

**第三部分：统计**

在school数据库中完成以下查询：

1查询姓名中有“明”字的学生人数。

2计算‘JSJ’系的平均年龄及最大年龄。

3查询学生中姓名为张明、赵英的人数

4计算每一门课的总分、平均分，最高分、最低分，按**平均分**由高到低排列

5 计算 1001,1002 课程的平均分。

6 查询平均分大于80分的学生学号及平均分

7 统计选修课程超过 2 门的学生学号

8 统计有10位成绩大于85分以上的课程号。

9 统计平均分不及格的学生学号

10 统计有大于两门课不及格的学生学号

*--第三部分：统计*use school  
*--1查询姓名中有“明”字的学生人数。*select *count*(sno) as '人数'  
from student  
where sname = '%明%';  
*--2计算‘JSJ’系的平均年龄及最大年龄。*select *avg*(sage),*max*(sage)  
from student  
where sdept = 'jsj';  
*--3查询学生中姓名为张明、赵英的人数*select *count*(sno)  
from student  
where sname in ('张明','赵英');  
*--4计算每一门课的总分、平均分，最高分、最低分，按平均分由高到低排列*select cno,*sum*(grade),*avg*(grade),*max*(grade),*min*(grade)   
from sc  
group by cno   
order by *avg*(grade) desc  
*--5 计算 1001,1002 课程的平均分。*select cno , *avg*(grade)   
from sc  
where cno in ('1001','1002') Group by cno  
*--6 查询平均分大于80分的学生学号及平均分*select sno,*avg*(grade)  
from sc  
where grade>80;  
*--7 统计选修课程超过 2 门的学生学号*select sno  
from sc  
group by sno  
having *count*(*\**)>2;  
*--8 统计有10位成绩大于85分以上的课程号。*select distinct cno  
from sc  
where grade>85  
group by cno  
having *count*(*\**) > 10;  
*--9 统计平均分不及格的学生学号*select distinct sno  
from sc  
group by sno  
having *avg*(grade)<60;  
*--10 统计有大于两门课不及格的学生学号*select sno   
from sc  
where grade<60   
group by sno   
having *count*(*\**) >2

**第四部分： 连接**

在school数据库中完成以下查询：

1查询 JSJ 系的学生选修的课程号

2查询选修1002 课程的学生的学生姓名 (不用嵌套及嵌套2种方法）

3查询数据库原理不及格的学生学号及成绩

4查询选修“数据库原理”课且成绩 80 以上的学生姓名(不用嵌套及嵌套2种方法）

5查询平均分不及格的学生的学号，姓名,平均分。

6查询女学生平均分高于75分的学生姓名。

7查询男学生学号、姓名、课程号、成绩。(一门课程也没有选修的男学生也要列出，不能遗漏)

*--第四部分：连接  
  
--1查询 JSJ 系的学生选修的课程号*select distinct cno  
from student join sc on student.sno = sc.sno  
where sdept = 'jsj';  
  
*--2查询选修1002 课程的学生的学生姓名 (不用嵌套及嵌套2种方法）*select sname  
from student join sc on student.sno = sc.sno  
where cno = '1002'  
  
select sname  
from student  
where sno in(  
 select sno  
 from sc  
 where cno = '1002'  
);  
  
*--3查询数据库原理不及格的学生学号及成绩*select sno,grade  
from course join sc on course.cno = sc.cno  
where cname = '数据库原理' and grade<60;  
*--4查询选修“数据库原理”课且成绩 80 以上的学生姓名(不用嵌套及嵌套2种方法）*select sname  
from student,sc,course  
where student.sno = sc.sno and sc.cno = course.cno and cname = '数据库原理' and grade > 80;  
  
select sname  
from student  
where sno in(  
 select sno  
 from sc  
 where cno in(  
 select cno  
 from course  
 where cname = '数据库原理'  
 )  
);  
*--5查询平均分不及格的学生的学号，姓名,平均分。*select sc.sno,sname,*avg*(grade)  
from student join sc on student.sno = sc.sno  
group by sc.sno  
having *avg*(grade)<60;  
*--6查询女学生平均分高于75分的学生姓名。*select sname  
from student join sc on student.sno = sc.sno  
where ssex = '女'  
group by student.sno  
having *avg*(grade) > 75;  
*--7查询男学生学号、姓名、课程号、成绩。(一门课程也没有选修的男学生也要列出，不能遗漏)*select student.sno,sname,cno,grade  
from student left outer join sc on student.sno = sc.sno  
where student.ssex = '男';

**第五部分： 嵌套、相关及其他**

1. 在school数据库中完成以下查询：

1 查询平均分不及格的学生人数

2 查询没有选修1002 课程的学生的学生姓名

3 查询平均分最高的学生学号及平均分 （2种方法 TOP , any , all）

\*4 查询没有选修1001，1002课程的学生姓名。

5 查询1002课程第一名的学生学号（2种方法）

6 查询平均分前三名的学生学号

7 查询 JSJ 系的学生与年龄不大于19岁的学生的差集

8 查询1001号课程大于90分的学生学号、姓名及平均分大于85分的学生学号、姓名

9 查询每门课程成绩都高于该门课程平均分的学生学号

10 查询大于本系科平均年龄的学生姓名

*--第五部分：嵌套、相关及其他  
  
--一.在school数据库中完成以下查询：  
--1 查询平均分不及格的学生人数*select *count*(*\**)  
from sc  
group by sno  
having *avg*(grade)<60  
*--2 查询没有选修1002 课程的学生的学生姓名*select sname  
from student  
where sno not in(  
 select sno  
 from sc  
 where cno = '1002'  
);  
*--3 查询平均分最高的学生学号及平均分 （2种方法 TOP , any , all）*select top 1 sno,*avg*(grade)  
from sc  
group by sno  
order by *avg*(grade) desc;  
  
select sno,*avg*(grade)  
from sc  
where *avg*(grade)>= all(  
 select *avg*(grade)  
 from sc  
 group by sno  
);  
  
*--\*4 查询没有选修1001，1002课程的学生姓名*select sname  
from student  
where sno not in(  
 select sno  
 from sc  
 where cno in('1001','1002')  
);  
*--5 查询1002课程第一名的学生学号（2种方法）*select top 1 sno  
from sc  
where cno = '1002'  
order by grade desc;  
  
with max\_grade(value) as(  
 select *max*(grade)  
 from sc  
)  
select sno  
from sc,max\_grade  
where grade = max\_grade.value  
*--6 查询平均分前三名的学生学号*select top 3 sno  
from sc  
group by *avg*(grade);  
*--7 查询 JSJ 系的学生与年龄不大于19岁的学生的差集*select *\**from student  
where sdept = 'jsj'  
except  
select *\**from student  
where sage<19;  
*--8 查询1001号课程大于90分的学生学号、姓名及平均分大于85分的学生学号、姓名*with avg\_grade\_greater\_85(sno) as(  
 select sno  
 from sc  
 group by sno  
 having *avg*(grade) > 85  
)  
select sc.sno,student.sname  
from sc,avg\_grade\_greater\_85,student  
where sc.sno = avg\_grade\_greater\_85.sno and sc.sno = student.sno and cno = '1001' and grade>90;   
*--9 查询每门课程成绩都高于该门课程平均分的学生学号*with avg\_grade(grade) as(  
 select *avg*(grade)  
 from sc  
 group by sno  
)  
select sno  
from sc,avg\_grade  
where sc.grade > avg\_grade.grade;  
*--10查询大于本系科平均年龄的学生姓名*with avg\_age(age) as(  
 select *avg*(sage)  
 from student  
)  
select sname  
from student,avg\_age  
where student.sage > avg\_age.age;

1. SPJ数据库中的查询：
2. 取出供应商S1提供的零件的颜色；
3. 取出为工程J1提供红色零件的供应商代号；
4. 取出为所在城市为上海的工程提供零件的供应商代号
5. 取出为所在城市为上海或北京的工程提供红色零件的供应商代号；
6. 取出供应商与工程所在城市相同的供应商提供的零件代号；
7. 取出上海的供应商提供给上海的任一工程的零件的代号；
8. 取出所有这样的一些＜CITY，CITY＞二元组，使得第1个城市的供应商为第2个城市的工程提供零件；
9. 取出所有这样的三元组＜CITY，PN，CITY＞，使得第1个城市的供应商为第2个城市的工程提供指定的零件；
10. 重复8题，但不检索两个CITY值相同的三元组。

*--二．SPJ数据库中的查询：*use SPJ  
*--1.取出供应商S1提供的零件的颜色；*select distinct color  
from pb join spjb on pb.pn = spjb.pn  
where spjb.sn = 's1';  
*--2.取出为工程J1提供红色零件的供应商代号；*select distinct sn  
from pb join spjb on pb.pn = spjb.pn  
where jn = 'J1' and color = '红色';  
*--3.取出为所在城市为上海的工程提供零件的供应商代号*select distinct SPJB.sn  
from SPJB join JB J on J.JN = SPJB.JN join SB S on S.SN = SPJB.SN  
where J.CITY = '上海';  
*--4.取出为所在城市为上海或北京的工程提供红色零件的供应商代号；*select distinct SPJB.SN  
from SPJB join PB P on P.PN = SPJB.PN join JB J on J.JN = SPJB.JN join SB S on J.CITY = S.CITY  
where P.COLOR = '红色' and J.CITY in ('上海','北京');  
*--5.取出供应商与工程所在城市相同的供应商提供的零件代号；*select distinct SPJB.PN  
from SPJB join JB J on J.JN = SPJB.JN join SB S on S.SN = SPJB.SN  
where j.CITY = s.CITY;  
*--6.取出上海的供应商提供给上海的任一工程的零件的代号；*select distinct SPJB.PN  
from SPJB join JB J on J.JN = SPJB.JN join SB S on S.SN = SPJB.SN  
where j.CITY = '上海';  
*--7.取出所有这样的一些＜CITY，CITY＞二元组，使得第1个城市的供应商为第2个城市的工程提供零件；*select distinct s.CITY,j.CITY  
from SPJB join JB J on J.JN = SPJB.JN join SB S on S.SN = SPJB.SN  
where s.CITY != j.CITY  
*--8.取出所有这样的三元组＜CITY，PN，CITY＞，使得第1个城市的供应商为第2个城市的工程提供指定的零件；*select distinct s.CITY,p.PN,j.CITY  
from SPJB join PB P on SPJB.PN = P.PN join JB J on J.JN = SPJB.JN join SB S on S.SN = SPJB.SN  
where s.CITY = '城市1' and j.CITY = '城市2' and p.PN = '零件号';  
*--9.重复8题，但不检索两个CITY值相同的三元组。*select distinct s.CITY,p.PN,j.CITY  
from SPJB join PB P on SPJB.PN = P.PN join JB J on J.JN = SPJB.JN join SB S on S.SN = SPJB.SN  
where s.CITY = '城市1' and j.CITY = '城市2' and p.PN = '零件号' and s.CITY != j.CITY;

## 实验七 视图

**目的: 掌握视图的建立、使用。**

在school数据库中，完成以下操作：

1建立学生学号、姓名、性别、课程号、成绩的视图 v\_sc

查看V\_sc中的数据。

create view v\_sc as(  
 select s.sno,s.sname,s.ssex,sc.cno,sc.grade  
 from student as s join sc on s.sno = sc.sno  
);

select *\**from v\_sc



2 建立学生学号、姓名、出生年份的视图 v\_age

查看V\_age中的数据。

create view v\_age as(  
 select sno,sname,sage  
 from student  
);

select *\**from v\_age

应用程序, 表格, Excel

描述已自动生成

3 建立 ‘JSJ’ 系的学生学号、姓名、性别、年龄的视图 V\_JSJ

create view v\_jsj as(  
 select sno,sname,ssex,sage  
 from student  
 where sdept = 'jsj'  
);

4 建立每门课程的平均分的视图 V\_avggrade

create view v\_avggrade as(  
 select cno,*avg*(grade) as avg\_grade  
 from sc  
 group by cno  
);

5 将 视图 v\_jsj 中 李文庆 的年龄改为21岁

update v\_jsj  
set sage = 21  
where sname = '李文庆';

6 察看 student 中李文庆的年龄

查看 v\_age 中李文庆的出生年月

select sage  
from student  
where sname = '李文庆';

select sage  
from v\_jsj  
where sno = '李文庆';

7 查询每门课程的及格率

create view v1 as(  
 select cno, *count*(*\**) as a  
 from sc  
 group by cno  
)  
create view v2 as(  
 select cno,*count*(*\**) as b  
 from sc  
 where grade>= 60  
 group by cno  
)   
  
select v2.cno,b/a\*1.0  
from v1 join v2 on v1.cno = v2.cno

思考:

1 利用 V\_JSJ 视图,可以更新SX 的学生的年龄吗? 写出理由

如: update v\_jsj set sage=25 where sno=’ 0004’

0004 号学生为 SX 系.

2 create view v\_student (sno,sname,ssex,sage,sdept) as

select sno,sname,ssex,sage,sdept from student

学籍管理系统使用此视图。现把student 的列名 sname 改为 xm,请问原来的学籍管理系统能正常运行吗？如不能，则如何处理能使系统正常运行。

**实验八：数据库综合实验**

1. 在Product 表中查出所有产品信息（三列：Name, ProductNumber, ListPrice） ，按产品名升序排列。
2. 在Product表中查出标价大于$1000的产品，求出这些产品的均价并按类型号分组。
3. 在Product表中查出每种产品类型均价等于该类产品最高标价的产品类型号。
4. 在SalesOrderDetail表中查出每个销售订单的销售总额。两列：订单号、销售总额。
5. 在SalesOrderDetail表中查出每种产品销售平均价格以及该产品迄今为止的总销售额。三列：ID、均价、总额。
6. 在SalesOrderDetail表中查出总销量大于5的产品 ID，并排序。 必须使用 HAVING。
7. 在SalesOrderDetail表中查出单价小于25，且平均订单量大于5的产品 ID。
8. 在SalesOrderDetail表中查出总销售额大于$10000 且平均单笔销量小于3的产品。三列：ID、均价、总销 售额。
9. 在SalesOrderDetail表中查出单笔销售产品数量大于 10 的订单销售的产品 ID 和均价。分组。
10. 在SalesOrderDetail表中查出完成了相同单笔销售额的订单的平均销售产品数量和这个销售总额。
11. 查出每种产品的销售总额和折扣总额。涉及表：Product、SalesOrderDetail，按产品名升序排列。共三列，必须使用 join。
12. 查出销售订单中每种产品的收入金额（扣除折扣后）。涉及表 ： Product、 SalesOrderDetail，按产品名降序排列。共三列。必须使用 join。
13. 查出产品型号为“Long-Sleeve Logo Jersey\*” （\*为任意字符）的每种产品名。涉及表：Product、ProductModel。一列，使用 EXISTS 和 IN 两种方法完成。
14. 在SalesOrderDetail表中统计订单中至少包含9项产品的产品ID及其销售总额。

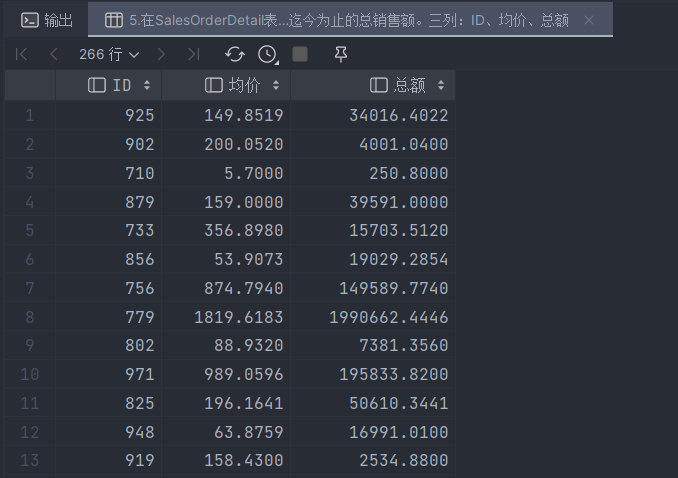
分析讨论：EXISTS返回的是布尔值，它不返回实际的结果。In相当于用一个表中的数据到另一个表中进行对比。Having用于统计函数的条件设定，这时不能用where，当要对数据进行分组计算时，要注意各个条件间的聚合问题。

*--1.在Product 表中查出所有产品信息（三列：Name, ProductNumber, ListPrice） ，按产品名升序排列*select Name,ProductNumber,ListPrice  
from Production.Product  
order by Name asc;  
电脑萤幕的截图

描述已自动生成  
*--2.在Product表中查出标价大于$1000的产品，求出这些产品的均价并按类型号分组*select ProductModelID,*avg*(ListPrice) as '均价'  
from Production.product  
where ListPrice>1000  
group by ProductModelID;  
图形用户界面, 文本

描述已自动生成  
*--3.在Product表中查出每种产品类型均价等于该类产品最高标价的产品类型号*select ProductModelID  
from Production.Product  
group by ProductModelID  
having *avg*(ListPrice) = *max*(ListPrice);  
文本

描述已自动生成  
*--4.在SalesOrderDetail表中查出每个销售订单的销售总额。两列：订单号、销售总额*select ProductID as '订单号',*sum*(UnitPrice) as '销售总额'  
from Sales.SalesOrderDetail  
group by ProductID;  
图形用户界面, 文本

描述已自动生成  
*--5.在SalesOrderDetail表中查出每种产品销售平均价格以及该产品迄今为止的总销售额。三列：ID、均价、总额*select ProductID as ID,*avg*(UnitPrice) as '均价', *sum*(UnitPrice) as '总额'  
from Sales.SalesOrderDetail  
group by ProductID;  
  
*--6在SalesOrderDetail表中查出总销量大于5的产品 ID，并排序。 必须使用 HAVING*select ProductID,*sum*(OrderQty)  
from Sales.SalesOrderDetail  
group by ProductID  
having *sum*(OrderQty)>5  
order by *sum*(OrderQty) desc;  
电脑的屏幕截图

描述已自动生成  
*--7.在SalesOrderDetail表中查出单价小于25，且平均订单量大于5的产品 ID*select ProductID  
from Sales.SalesOrderDetail  
group by ProductID  
having *avg*(OrderQty)>5 and *max*(UnitPrice)<25  
图形用户界面, 文本, 应用程序

描述已自动生成  
*--8.在SalesOrderDetail表中查出总销售额大于$10000 且平均单笔销量小于3的产品。三列：ID、均价、总销售额*select ProductID as ID,*avg*(UnitPrice) as '均价',*sum*(LineTotal) as '总销售额'  
from Sales.SalesOrderDetail  
group by ProductID  
having *sum*(LineTotal) > 10000 and *avg*(OrderQty)<3;  
图形用户界面, 文本, 应用程序

描述已自动生成  
*--9.在SalesOrderDetail表中查出单笔销售产品数量大于 10 的订单销售的产品 ID 和均价。分组。*select ProductID as ID,*avg*(UnitPrice) as '均价'  
from Sales.SalesOrderDetail  
where OrderQty>10  
group by ProductID;  
图形用户界面, 文本

描述已自动生成  
*--10.在SalesOrderDetail表中查出完成了相同单笔销售额的订单的平均销售产品数量和这个销售总额。*select *avg*(OrderQty),*sum*(LineTotal)  
from Sales.SalesOrderDetail  
group by LineTotal  
having *count*(*\**)>1;

图形用户界面, 文本

描述已自动生成  
*--11.查出每种产品的销售总额和折扣总额。涉及表：Product、SalesOrderDetail，按产品名升序排列。共三列，必须使用 join。*select p.Name as '产品名字',*sum*(LineTotal) as '销售总额',*sum*(s.UnitPriceDiscount \* s.OrderQty\*UnitPrice) as '折扣总额'  
from Production.Product as p join Sales.SalesOrderDetail as s on p.ProductID = s.ProductID  
group by p.ProductID,p.Name  
order by p.Name asc;

图形用户界面, 文本, 应用程序

描述已自动生成  
*--12查出销售订单中每种产品的收入金额（扣除折扣后）。涉及表 ： Product、 SalesOrderDetail，按产品名降序排列。共三列。必须使用 join。*select p.Name,*sum*((1-UnitPriceDiscount)\*UnitPrice\*s.OrderQty) as '收入金额'  
from Production.Product as p join Sales.SalesOrderDetail as s on p.ProductID = s.ProductID  
group by p.ProductID,p.Name  
order by p.Name desc;  
图形用户界面, 文本

描述已自动生成  
*--13.查出产品型号为“Long-Sleeve Logo Jersey\*” （\*为任意字符）的每种产品名。涉及表：Product、ProductModel。一列，使用 EXISTS 和 IN 两种方法完成。*select p.Name  
from Production.Product as p  
where p.ProductModelID in(  
 select pm.ProductModelID  
 from Production.ProductModel as pm  
 where Name like 'Long-Sleeve Logo Jersey%'  
);  
select distinct Name  
from Production.Product as p  
where *exists*(  
 select *\** from Production.ProductModel as pm  
 where p.ProductModelID = pm.ProductModelID and pm.Name like 'Long-Sleeve Logo Jersey%'  
);

图形用户界面, 文本

描述已自动生成  
*--14.在SalesOrderDetail表中统计订单中至少包含9项产品的产品ID及其销售总额。*select ProductID,*sum*(LineTotal)  
from Sales.SalesOrderDetail as SOD  
group by ProductID  
having *count*(*\**) >= 9;

图形用户界面, 文本, 应用程序

描述已自动生成

# 实验九

在实验四所建的grade数据库中，完成以下查询，文件保存为grade查询.sql。

附grade数据库表结构：

DEP（deptid，depname，depzr）含义：系表（系号，系名，系主任）

S(sno,sname,age,deptid) 含义：学生表(学号,姓名,年龄，系号)

SC(sno,cno,score) 含义：选修表(学号,课程号,成绩)

Course(cno,cname,pcno) 含义：课程表(课程号,课程名,先行课程号)

T(TNO,TN,PROF, SA,deptid) 含义：教师表(教师号,教师姓名,职称,工资,系号)

TC(TNO,CNO,book) 含义：教课表（教师号，课程号，所用书）

*--完成以下的SQL查询，将对应的SQL语句写在每道题目的下方*use grade  
*--1、查询学生的全部信息。*select *\**from s;  
  
*--2、查询选修了课程的学生号。*select distinct sno  
from sc;  
  
*--3、查询选修课程号为‘C1’的学生的学号和成绩。*select sno,score  
from sc  
where cno = 'C1';  
  
*--4、查询成绩高于85分的学生的学号、课程号和成绩。*select sno,cno,score  
from sc  
where score>85;  
  
*--5、查询选修C1或C2且分数大于等于85分学生的的学号、课程号和成绩。*select sno,cno,score  
from sc  
where cno in('C1','C2') and score>85;  
  
*--6、查询工资在1000至1500之间的教师的教师号、姓名及职称。*select tno,tn,prof  
from t  
where sa between 1000 and 1500;  
  
*--7、查询没有选修C1，也没有选修C2的学生的学号、课程号和成绩。*select sno, cno, score   
from SC   
where sno not in (  
 select sno   
 from SC   
 where cno in ('c1','c2'));  
  
*--8、查询所有姓张的教师的教师号和姓名。*select tno,tn  
from t  
where tn like '张%';  
  
*--9、查询姓名中第二个汉字是“力”的教师号和姓名。*select tno,tn  
from t  
where tn like '\_力%';  
  
*--10、查询没有考试成绩的学生的学号和相应的课程号。*select sno,cno  
from sc  
where score is null;  
  
*--11、查询选修C1 的学生学号和成绩，并按成绩降序排列。*select sno,score  
from sc  
where cno = 'c1'  
order by score desc;  
  
*--12、查询选修C1、C2的学号、课程号和成绩，查询结果按学号升序排列，学号相同再按成绩降序排列。*select sno,cno,score  
from sc  
where cno in ('c1','c2')  
order by sno,score desc;  
  
*--13. 求学号为S1学生的总分和平均分。*select *max*(score) as max\_score,*avg*(score) as avg\_score  
from sc  
where sno = 's1'；  
  
*--14、求选修C1号课程的最高分、最低分及之间相差的分数*select *max*(score),*min*(score),*max*(score)-*min*(score)  
from sc  
where cno = 'C1';  
  
*--15、求计算机系学生的总数*select *count*(*\**)  
from s join dep on s.DEPtid = dep.DEPtid  
where depname = '计算机系';  
  
*--16、查询各位教师的教师号及其任课的门数。*select tno,*count*(distinct cno)  
from tc   
group by tno;  
  
*--17、查询选修两门以上课程的学生学号和选课门数*select sno,*count*(distinct cno)  
from sc  
group by sno  
having *count*(distinct sno)>2;  
  
*--18、求选课在三门以上且各门课程均及格的学生的学号及其总成绩，查询结果按总成绩降序列出。*select sno,*sum*(score)  
from sc  
group by sno  
having *count*(distinct cno)>3 and *min*(score)>=60  
order by *sum*(score);  
  
*--19、查询与刘明教师职称相同的教师号、姓名。*select t1.tno,t1.tn  
from t as t1,t as t2  
where t1.prof = t2.prof and t2.tn = '刘明' and t1.tno != t2.tno  
  
*--20、查询讲授课程号为C1的教师姓名。*select tn  
from t join tc on t.tno=tc.tno  
where cno = 'C1';  
  
*--21、查询讲授课程号为C1的教师姓名，系名和课程名。*select tn,deptid,cname  
from t,tc,course  
where t.tno = tc.tno and course.cno = tc.cno and tc.cno = 'c1'  
  
*--22、查询计算机系的学生选修课程的课程名、学号、成绩*select course.cname,s.sno,sc.score  
from course,s,sc,dep  
where course.cno = sc.cno and sc.sno = s.sno and dep.DEPtid = s.DEPtid and dep.DEPname = '计算机系'  
  
*--23、查询‘王立’选修的课程名。*select cname  
from s,sc,course  
where s.sno = sc.sno and sc.cno = course.cno and s.sname = '王立';  
  
*--24、查询‘王立’选修的课程名和成绩。*select cname,score  
from s,sc,course  
where s.sno = sc.sno and sc.cno = course.cno and s.sname = '王立';  
  
*--25、查询'王立'的总分、平均分*select *sum*(score) as '总分', *avg*(score) as '平均分'  
from s,sc,course  
where s.sno = sc.sno and sc.cno = course.cno and s.sname = '王立';  
  
*--26、显示学生姓名及该学生的平均分*select sname,*avg*(score) as '平均分'  
from s join sc on s.sno = sc.sno  
group by s.sname;  
  
*--27、显示学生学号、学生姓名及该学生的平均分*select s.sno,sname,*avg*(score) as '平均分'  
from s join sc on s.sno = sc.sno  
group by s.sno,sname;  
  
*--28、查询每一门课的间接先修课（即先修课的先修课）*select c1.cname as '课程名',c2.cname '先修课程名'  
from course as c1,course as c2  
where c1.pcno = c2.cno;  
  
*--29、查询选修c1号课程且成绩在70分以上的所有学生的详细信息*select s.sno,sname,age,deptid  
from s join sc on s.sno = sc.sno  
where sc.cno = 'c1' and score > 70;  
  
*--30. 查询与“王立”在同一个系学习的学生姓名*select s1.sname  
from s as s1,s as s2  
where s2.Sname = '王立' and s1.sno != s2.Sno and s1.DEPtid = s2.DEPtid;  
  
*--31.查询选修了课程名为“计算机原理”的学生学号和姓名*select s.sno,sname   
from s,sc,course  
where s.sno = sc.sno and sc.cno = course.cno and cname = '计算机原理';  
  
*--32.查询其他系中比计算机系所有学生年龄都小的学生姓名及年龄。*with min\_age\_cmop(min\_age) as(  
 select *min*(age)  
 from s,dep  
 where s.DEPtid = dep.DEPtid and dep.DEPname = '计算机系'  
)  
select sname,age  
from s,dep,min\_age\_cmop  
where s.DEPtid = dep.DEPtid and dep.DEPname != '计算机系' and s.age < min\_age;   
  
*--33.查询没有选课的学生的姓名*select sname  
from s left outer join sc on s.sno = sc.Sno  
where cno IS NULL;  
  
*--34.找出成绩高于所有选修课程平均成绩的学号和成绩。*with avg\_score(value) as(  
 select *avg*(score)  
 from sc  
)  
select s.sno,sc.score  
from s,sc,avg\_score  
where s.sno = sc.sno and sc.score > avg\_score.value;  
  
*--35、查询其他系中比计算机系所有教师工资都高的教师的姓名和工资。*with max\_salary(salary) as(  
 select *max*(sa)  
 from t join dep on t.DEPtid = dep.DEPtid  
 where depname = '计算机系'  
)  
select tn,sa  
from t,dep,max\_salary  
where t.DEPtid = DEP.DEPtid and DEP.DEPname = '计算机系' and t.SA > max\_salary.salary  
  
*--36.检索至少选修了刘伟老师所授课程中一门课程的学生姓名；*with liuwei\_class(cno) as(  
 select cno  
 from t join tc on t.tno = tc.tno  
 where t.tn = '刘伟'  
)   
select sname  
from s,sc,liuwei\_class  
where sc.sno = s.sno and sc.cno = liuwei\_class.cno;  
  
*--37.检索李楠同学不学的课程的课程号*select distinct sc.cno  
from sc  
except  
select distinct sc.cno  
from sc join s on sc.sno = s.sno  
where s.sname = '李楠'  
  
*--20.求没有选修”数据库原理”课程的学生的姓名。*select sname   
from s   
where sno not in (  
 select sno   
 from sc,course   
 where sc.cno=course.cno and course.cname='数据库原理'  
)  
  
*--38.查询计算机系的学生与年龄不大于19岁的学生的交集*select s.sno  
from s join dep on s.DEPtid = dep.DEPtid  
where dep.DEPname = '计算机系'  
intersect  
select sno  
from s  
where age<=19  
  
*--39.查询选修课程c1的学生集合与选修课程c2的学生集合的交集*select sno   
from sc   
where cno='c1'   
intersect   
select sno   
from sc   
where cno='c2'  
  
*--40.查询计算机系的学生与年龄不大于19岁的学生的差集。*select s.sno  
from s join dep on s.DEPtid = dep.DEPtid  
where dep.DEPname = '计算机系'  
except  
select sno  
from s  
where age<=19

# 实验十：更新数据

1. **在school数据库中完成以下语句：**

**一 insert**

1 写出把下述学生的信息添加到student表中的命令。

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 学号 | 姓名 | 性别 | 年龄 | 系科 |
| 4001 | 赵茵 | 男 | 20 | SX |
| 4002 | 杨华 | 女 | 21 |  |

insert into student values  
(4001,'赵茵','男',20,'SX'),  
(4002,'杨华','女',21,null);

**2 批量插入数据**

1） 建立一个新表 sc\_name ，有属性 sno , sname , ssex , cno , grade 。

2） 把 SX 系学生的sno,sname,ssex, cno , grade 插入到表 sc\_name 中。

3） 察看 sc\_name 表的数据

create table sc\_name(  
 sno char(6),  
 sname varchar(8),  
 ssex char(2),  
 cno char(6),  
 grade int  
)  
  
insert into sc\_name  
select student.sno,sname,ssex,cno,grade  
from Student join sc on student.sno = sc.Cno and Sdept = 'SX';  
  
select *\**from sc\_name;

**二 Update**

1 修改 0001 学生的系科为: JSJ

2 把陈小明的年龄加1岁，性别改为女。

2 修改李文庆的1001课程的成绩为 93 分

3 把“数据库原理”课的成绩减去1分

update Student  
set sdept = 'JSJ'  
where Sno = '0001';  
  
update Student  
set Sage = Sage + 1, Ssex = '女'  
where sname = '陈小明';  
  
update sc  
set Grade = 93  
where sno in (  
 select sno  
 from Student  
 where sname = '李文庆'  
 );  
  
update sc\_name  
set grade = grade - 1  
where cno in(  
 select cno  
 from Course  
 where cname = '数据库原理'  
 );

**三 Delete**

1 删除所有 JSJ 系的男生

2 删除“数据库原理”的课的选课纪录

delete from Student  
where Sdept = 'JSJ';  
  
delete from sc\_name  
where cno in(  
 select cno  
 from Course  
 where cname = '数据库原理'  
 )

思考：修改数据的命令与修改表结构的命令有何区别？

**二）在grade数据库中完成以下语句：**

–1.在S表中插入一条学生记录（学号：S7；姓名：郑冬；年龄：21；系别：01）

--2.在SC表中插入一条选课记录（’S7’,’C1’）

--3.求出各系教师的平均工资，把结果存放在新表AVGSAL中。

--4、把刘伟教师转到信息管理系。

--5、将所有学生年龄增加1岁

--6、把教师表中工资小于等于1000元的讲师的工资提高20%。

--7、把教师表中工资低于平均工资的教师的工资提高1000元。

--8、把讲授C5课程的教师的岗位津贴增加100元。

--9、把所有教师的工资提高到平均工资的1.2倍

--10、将刘伟教师的工资置为空值

--11、删除刘伟教师授课的记录。

use grade  
  
*--1*insert into s values  
('s7','郑东',21,01);  
  
*--2*insert into SC values  
('s7','c1',null);  
  
*--3*create table avgsal(  
 department char(10),  
 avg\_sa int  
);  
insert into avgsal  
select DEP.depname,*avg*(sa) as '平均工资'  
from DEP join T on dep.DEPtid = T.DEPtid  
group by DEP.DEPtid,dep.depname  
  
*--4*update T  
set DEPtid = (  
 select DEPtid  
 from DEP  
 where DEPname = '信息管理系'  
 )  
where Tn = '刘伟';  
  
*--5*update S  
set age = age + 1;  
  
*--6*update t  
set SA = SA \* 1.2  
where SA<=1000 and PROF = '讲师';  
  
*--7*update t  
set SA = SA + 1000  
where SA < (  
 select *avg*(sa)  
 from t  
 );  
  
*--8*update t  
set sa = sa + 100  
where tno in (  
 select TNO  
 from TC  
 where cno = 'c5'  
 );  
  
*--9、把所有教师的工资提高到平均工资的1.2倍*update t  
set sa = 1.2 \* (  
 select *avg*(sa)  
 from t  
 );  
*--10、将刘伟教师的工资置为空值*update t  
set sa = null  
where tn = '刘伟';  
*--11、删除刘伟教师授课的记录。*delete from tc  
where TNO in (  
 select TNO  
 from T  
 where Tn = '刘伟'  
 );