数据库实验报告 – CAP 数据库

一、用 Select 语句完成查询

(1) 找出住在纽约的代理商的 aid 值和名字

```
SELECT aid,aname
FROM dbo.agents
WHERE city='NEW YORK'
```

(2) 检索订货记录中所有的零件的 pid 值。

```
SELECT DISTINCT pid
FROM dbo.orders;
```

(3) 检索有关住在 Duluth 或 Dallas 的代理商的所有信息。

```
SELECT *
FROM dbo.agents
WHERE city='Duluth' or city='Dallas';
```

(4) 检索居住地名以“D”开始的客户的信息。

```
SELECT *
FROM dbo.customers
WHERE city LIKE 'D%';
```

(5) 检索所有客户的最高折扣率

```
SELECT MAX(discnt)
FROM dbo.customers;
```

(6) 求出所有的订货交易的总金额。

```
SELECT SUM(dollars)
FROM dbo.orders;
```

(7) 求出产品 p01 的订购总量。

```
SELECT SUM(qty)
FROM dbo.orders
WHERE pid='p01';
```

(8) 求出有顾客居住的城市的数目

```
SELECT COUNT(DISTINCT city)
FROM dbo.customers;
```

二、数据库查询与修改

(1) 找出所有客户、代理商和商品都在同一城市的三元组（cid, aid, pid）

```
SELECT  cid,aid,pid
FROM agents,customers,products
WHERE               agents.city=customers.city              AND
customers.city=products.city
```

(2) 找出所有客户、代理商和商品不都在同一城市（可能有两个在同一城市）的三元组（cid, aid, pid）。

```
SELECT  cid,aid,pid
FROM agents,customers,products
WHERE               agents.city!=customers.city             OR
customers.city!=products.city
```

(3) 找出所有在同一城市的代理商的 aid 对。

```
SELECT  A.aid,B.aid
FROM agents A,agents B
WHERE A.city=B.city
```

(4) 找出同时订购了商品 p01 和 p07 的客户的 cid 值。（若找出客户的 cname 呢？）

(5) 统计各个产品的销售总量。

```
SELECT pid,SUM(qty)
FROM orders
GROUP BY pid;
```

(6) 当某个代理商所订购的某样产品的总量超过 1000 时，打印出所有满足条件的产品和代理商的 ID 以及这个总量。

```
SELECT aid,sum(qty)
FROM orders
GROUP BY aid
HAVING sum(qty)>1000;
```

(7) **找出订购了产品 p05 的顾客的名字。**

```
SELECT distinct(cname)
FROM orders,customers
WHERE orders.pid = 'p05' and orders.cid = customers.cid;
```

(8) 检索满足以下条件的顾客-代理商的姓名对（cname,aname），其中的顾客 cname 通过代理商 aname 订了货。

```
SELECT distinct cname,aname
FROM orders A,orders B,agents,customers
WHERE A.aid = agents.aid and B.cid = customers.cid and A.ordno =
B.ordno;
```

(9) 找出至少被两个顾客订购的产品的 pid 值。

```
SELECT pid
FROM orders
GROUP BY pid
HAVING COUNT(DISTINCT cid)>=2;
```

(10)在 customers 表中插入一个新行。

```
INSERT INTO customers(cid,cname,city)
VALUES ('c007','WinDix','Dallas');
```

(11)检索 customers 表中 discnt 值为空的行。

```
SELECT *
FROM customers
WHERE discnt IS NULL;
```

(12)检索客户以及他们订购商品的详细信息。（用外联接）

```
SELECT customers.cid, customers.cname, customers.city,
    products.pid, products.pname, products.price, products.city,
    orders.qty, orders.dollars
FROM orders
    LEFT OUTER JOIN customers ON (customers.cid=orders.cid)
    LEFT OUTER JOIN products ON (orders.pid=products.pid)
```

(13)检索有关住在 Duluth 或 Dallas 的代理商的所有信息。（要求使用 IN 谓词实现）

```
SELECT *
FROM agents
WHERE city IN ('Duluth','Dallas');
```

(14)找出通过住在 Duluth 或 Dallas 的代理商订货的所有顾客的姓名和折扣率。(要求使用 IN 谓词实现)

```
SELECT DISTINCT customers.cname,customers.discnt
FROM customers,orders
WHERE customers.cid=orders.cid AND
    aid IN (
    SELECT aid
    FROM agents
    WHERE city IN ('Duluth','Dallas'))
```

(15)求所有满足以下条件的顾客的 cid 值：该顾客的 discnt 的值小于任一住在 Duluth 的顾客的 discnt 值。

```
SELECT cid
FROM customers
WHERE discnt <ALL (
        SELECT discnt
        FROM customers
        WHERE city='Duluth')
```

(16)检索没有通过代理商 a05 订货的所有顾客的名字。

```
SELECT cname
FROM customers
WHERE NOT EXISTS
    (SELECT *
     FROM orders
     WHERE cid=customers.cid AND aid='a05')
```

(17)检索一个包含顾客所在的或者代理商所在的城市的名称。(使用 UNION 实现)

```
SELECT city
FROM customers
UNION
SELECT city
FROM agents
```

(18)在 orders 表中插入一个新行。

```
INSERT INTO orders(ordno,month,cid,aid,pid)
VALUES (1107,'aug','c006','a04','p01');
```

(19)创建一个名为 swcusts 的表，它包含住在西南部的所有顾客，并向该表中插入所有来自 Dallas 或 Austin 的顾客。

```
INSERT INTO swcusts
SELECT *
FROM customers
WHERE city = 'Dallas' or city = 'Austin';
```

(20)将所有住在 New York 的代理商的佣金率提高 10%。

```
UPDATE agents
SET per = per*1.1
WHERE city = 'New York';
```

(21)删除所有住在 New York 的代理商。

```
DELETE
FROM agents
WHERE city = 'New York'
```

(22)创建一个 agentorders 视图，它扩展了表 orders 的行，包括订货的代理商的详细信息。

```
CREATE VIEW agentorders
AS
SELECT orders.*,aname,city,per
FROM orders,agents
```

(23)利用 agentorders 视图查询代理商 Brown 的所有订单信息

```
SELECT distinct *
FROM agentorders
WHERE aname ='Brown';
```

(24)创建 cacities 视图，该视图列出表 customers 和表 agents 中所有配对的城市，其中该顾客通过该代理商订购了商品。

```
CREATE VIEW cacities
AS
SELECT customers.city
FROM customers,agents,orders A,orders B
WHERE A.cid = customers.cid and B.aid = agents.aid and A.ordno =
B.ordno;
```

(25)创建 custs 视图

```
CREATE VIEW custs
AS
SELECT *
FROM customers
WHERE discnt<=15.0 with check option;
```

(26)对 custs 视图进行更新操作

```
UPDATE custs
SET discnt=discnt+10;
```

数据库实验报告 – StudentCourse 数据库

一、表的建立

(1) 建立 Student 表

```
CREATE TABLE Student
(   Sno CHAR(10) PRIMARY KEY,
    Sname CHAR(10) NOT NULL,
    Ssex CHAR(2) NOT NULL,
    Sage SMALLINT,
    Sdept CHAR(4) NOT NULL
);
```

(2) 建立 Course 表

```
CREATE TABLE Course
(
    Cno CHAR(3) PRIMARY KEY,
    Cname CHAR(30) NOT NULL,
    Credit SMALLINT,
    Pcno CHAR(3),
    FOREIGN KEY (Pcno) REFERENCES Course(Cno)
);
```

(3) 建立 SC 表

```
CREATE TABLE SC
(
    Sno CHAR(10),
    Cno CHAR(3),
    Grade SMALLINT,
    PRIMARY KEY(Sno,Cno),
    FOREIGN KEY (Sno) REFERENCES Student(Sno),
    FOREIGN KEY (Cno) REFERENCES Course(Cno)
);
```

二、表数据的添加

```
INSERT
INTO Student(Sno,Sname,Ssex,Sage,Sdept)
VALUES
    ('95001','李勇','男',20,'CS');

    ('95002','刘晨','女',19,'IS');
    ('95003','王敏','女',18,'MA');
    ('95004','张立','男',19,'IS');
    ('95005','刘云','女',18,'CS');
```

```
INSERT
INTO Course(Cno,Cname,Credit,Pcno)
VALUES ('1','数据库',4,'5');

VALUES ('2','数学',6,NULL);
VALUES ('3','信息系统',3,'1');
VALUES ('4','操作系统',4,'6');
VALUES ('5','数据结构',4,'7');
VALUES ('6','数据处理',3,NULL);
VALUES ('7','PASCAL 语言',4,'6');
```

```
INSERT
INTO SC(Sno,Cno,Grade)
VALUES ('95001','1',92);

VALUES ('95001','2',85);
VALUES ('95001','3',88);
VALUES ('95002','2',90);
VALUES ('95002','3',80);
VALUES ('95003','2',85);
VALUES ('95004','1',58);
VALUES ('95004','2',85);
```

三、表数据的修改

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

```
UPDATE Course
SET Credit=4
WHERE Cno='4';
```

```
UPDATE Course
SET Pcno=NULL
WHERE Cno='7';
```

```
UPDATE SC
SET Grade=Grade+3
WHERE Sno='95001' AND Cno='1';
```

四、表数据的删除

```
DELETE
FROM SC
WHERE Sno='95005';

DELETE
FROM Student
WHERE Sno='95005';
```

```
DELETE
FROM SC;
```

```
DELETE
FROM SC
WHERE Grade<60;
```

五、SQL 语句查询练习题

(1) 查询全体学生的学号和姓名

```
SELECT Sno,Sname
FROM Student;
```

(2) 查询选修了课程名为'数据库原理' 的学生的学号和姓名

```
SELECT Student.Sno,Sname
FROM Student,SC
WHERE Student.Sno=SC.Sno AND SC.Cno IN
(
    SELECT Cno
    FROM Course
    WHERE Cname='数据库'
);
```

或者:

```
SELECT Student.Sno,Sname
FROM Student,SC,Course
WHERE Student.Sno=SC.Sno AND SC.Cno=Course.Cno AND Cname='数据库
';
```

(3) 查询全体学生的姓名，出生年份,和所在系，并用小写字母表示所有系名，并给各列指定列名。

```
SELECT Sname '姓名', (2023-Sage) '出生年份', Lower(Sdept) '所在系'
FROM Student;
```

(4) 查询有多少名学生的数据库课程成绩不及格

```
SELECT DISTINCT COUNT(Student.Sno)
FROM Student,SC
WHERE Student.Sno=SC.Sno
    AND Grade<60
    AND SC.Cno IN
    (
        SELECT Cno
        FROM Course
        WHERE Cname='数据库'
    );
```

(5) 查找所有姓'李'的学生的姓名，学号和性别

```
SELECT Sname,Sno,Ssex
From Student
WHERE Sname LIKE '李%';
```

(6) 求没有选修数学课程的学生学号

```
SELECT Student.Sno
From Student
WHERE Sno NOT IN
    (
        SELECT Sno
        FROM SC
        GROUP BY Sno
        HAVING
            EXISTS
            (SELECT Cno FROM Course WHERE Cname='数学')
    );
```

(7) 查询选修了课程的学生的学号

```
SELECT DISTINCT Student.Sno
From Student,SC
WHERE Student.Sno=SC.Sno;
```

(8) 计算 1 号课程的学生的平均成绩，最高分和最低分

```
SELECT AVG(Grade) '平均',MAX(Grade) '最高',MIN(Grade) '最低'
FROM SC
WHERE Cno='1';
```

(9) 查询数学系和信息系的学生的信息;

```
SELECT *
FROM Student
WHERE Sdept IN ('MA','IS');
```

(10)将年龄为 19 岁的学生的成绩置零

```
UPDATE SC
SET Grade=0
WHERE Sno IN
(
    SELECT Sno
    FROM Student
    WHERE Sage=19
)
```

(11)查询所有选修了 1 号课程的学生姓名

```
SELECT Sname
FROM Student,SC
WHERE Student.Sno=SC.Sno AND SC.Cno='1';
```

(12)对每一个性别，求学生的平均年龄，并把结果存入数据库

先创建表，再插入数据

```
CREATE TABLE Sex_age
(   Ssex CHAR(2) PRIMARY KEY,
    Avg_age SMALLINT
);

INSERT
INTO Sex_age(Ssex,Avg_age)
    SELECT Ssex,AVG(Sage)
    FROM Student
    GROUP BY Ssex;
```

(13)查询每个学生已获得的学分

```
SELECT Student.Sno,Sname,SUM(Credit) 'Credit'
FROM Student
        LEFT OUTER JOIN SC ON(Student.Sno=SC.Sno)
        LEFT OUTER JOIN Course ON(SC.Cno=Course.Cno)
GROUP BY Student.Sno,Sname
```

或者：

```
CREATE VIEW ALL_INFO
AS
    SELECT Student.Sno, Sname, Ssex, Sage, Sdept, Course.Cno,
Cname, Grade, Credit, Pcno
    FROM Student,SC,Course
    WHERE Student.Sno=SC.Sno AND SC.Cno=Course.Cno;

SELECT Sno,Sname,SUM(Credit) 'Credit'
FROM ALL_INFO
GROUP BY Sno,Sname;
```

(14)将所有女生的记录定义为一个视图

```
CREATE VIEW Female_INFO
AS
    SELECT Student.Sno, Sname, Ssex, Sage, Sdept, Course.Cno,
Cname, Grade, Credit, Pcno
    FROM Student,SC,Course
    WHERE Student.Sno=SC.Sno
        AND SC.Cno=Course.Cno
        AND Ssex='女';
```

(15)查询没有选修了 1 号课程的学生姓名

```
SELECT Sno,Sname
FROM Student
WHERE Sno NOT IN
(   SELECT Sno
    FROM SC
    WHERE Cno='1'
);
```

(16)将所有选修了数据库课程的学生的成绩加 5 分

```
UPDATE SC
SET Grade=Grade+5
WHERE Cno IN
(
    SELECT Cno
    FROM Course
    WHERE Cname='数据库'
)
```

(17)查询各系的男女生学生总数，并按系升序排列，女生排在前

```
SELECT Sdept,SSex,COUNT(Sno)
FROM STUDENT
GROUP BY Sdept,Ssex
ORDER BY Sdept ASC,SSex DESC;
```

(18)查询'信息系'(IS)学生"数据结构"课程的平均成绩

```
SELECT AVG(Grade)
FROM SC LEFT OUTER JOIN Student ON (Student.Sno=SC.Sno)
WHERE Sdept='IS'
```

(19)创建一个反映学生出生年份的视图

```
CREATE VIEW Student_Birth
AS
SELECT *,2023-Sage Birth
FROM Student
```

(20)查询与'王田'在同一个系学习的学生的信息

```
SELECT *
FROM Student
WHERE Sdept IN
(   SELECT Sdept
    FROM Student
    WHERE Sname='王田'
);
```

(21)查询年龄在 20 岁以下的学生的姓名及其年龄

```
SELECT Sname,Sage
FROM Student
WHERE Sage<20;
```

(22)查询当前至少选修数据库和信息系统其中一门课的学生的学号

```
SELECT DISTINCT Student.Sno
FROM Student,SC,Course
WHERE Student.Sno=SC.Sno
    AND SC.Cno=Course.Cno
    AND Cname IN ('数据库','信息系统');
```

(23)查询每个学生的学号，姓名，选修的课程名和成绩:

```
SELECT DISTINCT Student.Sno,Sname,Cname,Grade
FROM Student,SC,Course
WHERE Student.Sno=SC.Sno
    AND SC.Cno=Course.Cno;
```

(24)查找名字中包括“俊”的学生的姓名，学号，选课课程和成绩

```
SELECT DISTINCT Student.Sno,Sname,Cname,Grade
FROM Student,SC,Course
WHERE Student.Sno=SC.Sno
    AND SC.Cno=Course.Cno
    AND Sname LIKE '%俊%'
```

(25)查询学分大于 8 的学生，输出学生的学号和学分

```
SELECT Student.Sno,Sname,SUM(Credit) 'Credit'
FROM Student
        LEFT OUTER JOIN SC ON(Student.Sno=SC.Sno)
        LEFT OUTER JOIN Course ON(SC.Cno=Course.Cno)
GROUP BY Student.Sno,Sname
HAVING SUM(Credit)>8
```

(26)查询 IS,CS,MA 系的所有学生的姓名和性别

```
SELECT Sname,Ssex
FROM Student
WHERE Sdept IN ('IS','CS','MA')
```

(27)查询至少选修了 2 门课程的学生的平均成绩

```
SELECT Student.Sno,Sname,AVG(Grade) 'AvgGrade'
FROM Student
       LEFT OUTER JOIN SC ON(Student.Sno=SC.Sno)
       LEFT OUTER JOIN Course ON(SC.Cno=Course.Cno)
GROUP BY Student.Sno,Sname
HAVING COUNT(SC.Cno)>=2;
```

(28)查询每个学生所选课程的平均成绩，最高分，最低分,和选课门数

```
SELECT Student.Sno,Sname,AVG(Grade) 'AvgGrade',
    MAX(Grade) 'MaxGrade',MIN(Grade) 'MinGrade',
    COUNT(SC.Cno) 'CourseCount'
FROM Student
       LEFT OUTER JOIN SC ON(Student.Sno=SC.Sno)
       LEFT OUTER JOIN Course ON(SC.Cno=Course.Cno)
GROUP BY Student.Sno,Sname;
```

(29)删除年龄大于 21 岁所有学生的选课记录

```
DELETE
FROM SC
WHERE Sno IN
(   SELECT Sno
    FROM Student
    WHERE Sage>21
);
```

(30)查询没有先行课的课程的课程号 cno 和课程名 cname

```
SELECT Cno,Cname
FROM Course
WHERE Pcno IS NULL;
```

(31)创建信息系学生信息的视图

```
CREATE VIEW Stu_IS
AS
    SELECT *
    FROM Student
    WHERE Sdept='IS'
;
```

(32)在信息系的学生视图中查询年龄小于 20 岁的学生

```
SELECT *
FROM Stu_IS
WHERE Sage<20;
```

(33)删除马朝阳同学的所有选课记录

```
DELETE
FROM SC
WHERE Sno IN
(   SELECT Sno
    FROM Student
    WHERE Sname='马朝阳'
);
```

(34)查询选修了 3 号课程的学生的学号和成绩，并按分数降序排列

```
SELECT Sno,Grade
FROM SC
WHERE Cno='3'
ORDER BY Grade DESC;
```

(35)查询数据库课程成绩不及格的学生，输入其学号，姓名和成绩

```
SELECT Student.Sno,Sname,Grade
FROM Student,SC
WHERE
    Student.Sno=SC.Sno
    AND Grade<60
    AND Cno IN(
        SELECT Cno
        FROM Course
        WHERE Cname='数据库'
        )
;
```

(36)查询全体学生的情况,查询结果按所在系号升序排列，同一系中的学生按年龄降序排列

```
SELECT *
FROM Student
ORDER BY Sdept ASC,Sage DESC;
```

(37)查询每个学生及其选修课程的情况

```
SELECT Student.Sno,Student.Sname,Course.Cname
FROM Student
       LEFT OUTER JOIN SC ON(Student.Sno=SC.Sno)
       LEFT OUTER JOIN Course ON(SC.Cno=Course.Cno)
;
```

(38)查询每一门课程的间接先行课

```
SELECT Fir.Cno,Sec.Pcno
FROM Course Fir,Course Sec
WHERE Fir.Pcno=Sec.Cno;
```

(39)查询选修 1 号课程且成绩在 85 分以上的所有学生的学号、姓名

```
SELECT Student.Sno,Sname
FROM Student
    LEFT OUTER JOIN SC ON(Student.Sno=SC.Sno)
WHERE Cno='1' AND Grade>85;
```

(40)查询全体学生的所有信息

```
SELECT *
FROM Student;
```

(41)查询选修了课程'1'和课程'2'的学生的学号

```
SELECT Student.Sno
FROM Student
WHERE 2=
    (
    SELECT COUNT(*)
    FROM SC
    WHERE
        Student.Sno=SC.Sno
        AND(SC.Cno='1' OR SC.Cno='2')
    )
;
```

(42)创建信息系选修了 1 号课程的学生的视图

```
CREATE VIEW Stu_IS_Course1
AS
    SELECT Student.Sno,Sname,Sdept,Grade
    FROM Student
        LEFT OUTER JOIN SC ON(Student.Sno=SC.Sno)
    WHERE
        Sdept='IS'
        AND Cno='1'
;
```

(43)建立信息系选修了 1 号课程且成绩在 90 分以上的学生的视图

```
CREATE VIEW Stu_IS_Course1_90
AS
    SELECT Student.Sno,Sname,Sdept,Grade
    FROM Student
        LEFT OUTER JOIN SC ON(Student.Sno=SC.Sno)
    WHERE
        Sdept='IS'
        AND Cno='1'
        AND Grade>90
;
```

(44)查询修课总学分在 10 学分以下的学生姓名

```
SELECT Student.Sno,Sname,SUM(Credit) 'Credit'
FROM Student
        LEFT OUTER JOIN SC ON(Student.Sno=SC.Sno)
        LEFT OUTER JOIN Course ON(SC.Cno=Course.Cno)
GROUP BY Student.Sno,Sname
HAVING SUM(Credit)<10
```

(45)查询比'刘晨'年龄小的所有学生的信息

```
SELECT *
FROM Student
WHERE Sage<
(   SELECT Sage
    FROM Student
    WHERE Sname='刘晨'
)
```

(46)查询所有选修了 2 号课程的学生的姓名

```
SELECT Sname
FROM Student,SC
WHERE Student.Sno=SC.Sno
    AND Cno='2';
```

(47)查询其他系中比信息系(IS)某一学生年龄小的学生姓名和年龄

```
SELECT Sname,Sage
FROM Student
WHERE
    Sdept != 'IS'
    AND Sage<ANY
    (   SELECT Sage
        FROM Student
        WHERE Sdept='IS'
    );
```

(48)查询学生 2 号课程的成绩，并按照成绩由高到低输出

```
SELECT Sname,Grade
FROM Student,SC
WHERE
    Student.Sno=SC.Sno
    AND Cno='2'
ORDER BY Grade DESC;
```

(49)查询考试成绩有不及格的学生的学号

```
SELECT DISTINCT Student.Sno
FROM Student,SC
WHERE
    Student.Sno=SC.Sno
    AND Grade<60
```

(50)查询其他系中比信息系(IS)学生年龄都小的学生姓名和年龄

```
SELECT Sname,Sage
FROM Student
WHERE
    Sdept != 'IS'
    AND Sage<ALL
    (   SELECT Sage
        FROM Student
        WHERE Sdept='IS'
    );
```

(51)将所有学生的学号和他的平均成绩定义为一个视图

```
CREATE VIEW Stu_AVG
AS
    SELECT Student.Sno,AVG(Grade) Average
    FROM Student
        LEFT OUTER JOIN SC ON(Student.Sno=SC.Sno)
    GROUP BY Student.Sno
```

(52)在视图 S_G 中查询平均成绩在 90 分以上的学生的学号和平均成绩:

```
SELECT Sno,Average
FROM Stu_AVG
WHERE Average>90;
```

(53)查询与计算机系(CS)系所有学生的年龄均不同的学生学号，姓名和年龄

```
SELECT Sno,Sname,Sage
FROM Student
WHERE Sage NOT IN
(
    SELECT Sage
    FROM Student
    WHERE Sdept='CS'
)
```

(54)查询信息系选修了 1 号课程的学生

```
SELECT Student.Sno,Sname
FROM Student,SC
WHERE
    Student.Sno=SC.Sno
    AND Cno='1'
    AND Sdept='IS'
```

(55)查询与其他所有学生年龄均不同的学生学号，姓名和年龄

```
SELECT Sno,Sname,Sage
FROM Student Stu
WHERE 1=
(
    SELECT COUNT(*)
    FROM Student
    GROUP BY Sage
    HAVING Sage=Stu.Sage
)
```

(56)查询选修了全部课程的学生姓名

```
SELECT Sname
FROM Student
    LEFT OUTER JOIN SC ON(Student.Sno=SC.Sno)
GROUP BY Student.Sname
HAVING Count(Cno)=
(
    SELECT Count(*)
    FROM Course
);
```

(57)求没有人选修的课程号 cno 和 cname

```
SELECT Cno,Cname
FROM Course
WHERE Cno NOT IN
(
    SELECT Cno
    FROM SC
)
```

(58)查询满足条件的(sno,cno)对，其中该学号的学生没有选修该课程号 cno 的课程

```
SELECT DISTINCT Student.Sno,Course.Cno
FROM Student,Course
EXCEPT
    SELECT Sno,Cno
    FROM SC;
```

(59)查询每个学生的课程成绩最高的成绩信息(sno,cno,grade)

```
SELECT STU.Sno,Cno,Grade
FROM Student Stu,SC
WHERE
    Stu.Sno=SC.Sno
    AND Grade=
    (
        SELECT MAX(Grade)
        FROM Student
                LEFT OUTER JOIN SC ON(Student.Sno=SC.Sno)
        GROUP BY Student.Sno
        HAVING Stu.Sno=Student.Sno
    );
```

(60)查询学生总人数

```
SELECT Count(*)
FROM Student
```

(61)查询年龄在 20-30 岁直接的学生的姓名，姓名，所在系

```
SELECT Sname,Sdept
FROM Student
WHERE Sage>=20 AND Sage<=30
```

(62)查询所有课程的总学分数和平均学分数,以及最高学分和最低学分

```
SELECT SUM(credit) ,AVG(credit), MAX(credit), MIN(credit)
FROM Course;
```

(63)求成绩低于该门课程平均成绩的学生的成绩信息(sno,cno,grade)

```
SELECT A.Sno Sno,A.Cno Cno,A.Grade Grade
FROM SC A
WHERE grade < (
    SELECT AVG(grade)
    FROM SC B
    WHERE A.Sno = B.Sno;
```

(64)查询各系的学生的人数并按人数从多到少排序

```
SELECT count(Sno),Sdept
FROM Student
GROUP BY sdept
ORDER BY count(sno) DESC;
```

(65)创建年龄大于 23 岁的学生的视图

```
CREATE VIEW age23
AS
SELECT *
FROM Student
WHERE Sage > 23;
```

(66)查询选修了课程的学生总数

```
SELECT DISTINCT COUNT(sno)
FROM SC;
```

(67)查询选修了 3 门课程以上的学生的学号和姓名

```
SELECT Sno,Sname
FROM Student
WHERE Sno in(
    SELECT SC.Sno
    FROM SC,Student
    WHERE SC.Sno = Student.Sno
    GROUP BY SC.Sno
    HAVING count(Cno) >= 3
);
```

(68)查询平均分超过 80 分的学生的学号和平均分

```
SELECT Sno,AVG(grade) AvgGrade
FROM SC
GROUP BY sno
HAVING AVG(grade) > 80
```

(69)比较：求各学生的 60 分以上课程的平均分

```
SELECT A.sno,AVG(A.grade) avggrade
FROM SC A
WHERE 60 < all(
    select grade
    from sc B
    where A.sno = B.sno
)
group by A.sno
having AVG(A.grade) > 80;
```

(70)查询”信息系”(IS)中选修了 2 门课程以上的学生的学号

```
select sc.sno
from sc,student
where sdept = 'IS' and sc.sno = student.sno
group by sc.sno
having count(cno) >= 2
```

(71)查询选修了 1 号课程或 2 号课程的学生的学号

```
select distinct sno
from sc
where cno = '1' or cno = '2'
```

(72)查询平均成绩少于 70 分的学生的学号

```
select sno
from sc
group by sno
having AVG(grade) < 70
```

(73)将信息系学生视图 IS_Student 中学号为”95002”的学生姓名改为”刘辰”

```
update IS_Student
set sname = '刘辰'
where sno = '95002'
```

比较: update IS_Student set sname=’刘辰’ where sno=’95003’ 此语句不能实现数据的更新.

(74)向信息系学生视图 IS_Student 中插入一个新的学生记录，学号为 95029,姓名为”刘一梦”，年龄为 20 岁

```
insert
into IS_Student(sno,sname,sage,ssex,sdept)
values('95029','刘一梦',20,'女','IS')
```

(75)删除信息系学生视图 IS_Student 中学号为 95004 的学生的记录

```
delete
from IS_Student
where sno = '95004';
```

数据库实验报告 – 授权与完整性控制

一、管理权限

(1) 把查询、插入表 customers 和修改 cid 的权限授予用户 u01

```
GRANT SELECT,INSERT,UPDATE(cid)
ON customers
TO u01;
```

(2) 把对表 orders 的所有操作权限授予用户 u01

```
GRANT ALL PRIVILEGES
ON orders
TO u01;
```

(3) 把对表 customers 的查询权限授予所有用户

```
GRANT SELECT
ON customers
TO PUBLIC;
```

(4) 把对表 agents 的插入权限授予用户 u01，并允许将此权限再授予其他用户

```
GRANT INSERT
ON agents
TO u01
WITH GRANT OPTION;
```

(5) 以新用户 U01 的身份登录并向 customers 表插入数据

```
INSERT
INTO customers
VALUES('c007','Marshuni','Wuhan',15);
```

(6) 以新用户 U01 的身份登录并将对表 agents 的插入权限再授予 u02

```
GRANT INSERT
ON agents
TO u02;
```

(7) 以用户 U02 身份登录向 agents 表中插入数据

```
INSERT
INTO agents
VALUES('a07','Jianxiao LIU','Wuhan',999);
```

(8) 以 DBA 的身份登录并把用户 u01 对 agents 表的插入权限撤销

```
REVOKE INSERT
ON agents
FROM u02;
```

(9) DBA 把在数据库 CAP 中建立表的权限授予用户 u01

```
GRANT CREATE TABLE
TO u01;
```

(10) 以 u01 的身份登录并在 CAP 中创建新表。

```
CREATE TABLE table01
(   tid CHAR(10),
    tname CHAR(16),
    tinfo CHAR(32)
);
```

二、完整性控制

(1) 以 DBA 身份登录并在 orders 表中插入新行。

```
INSERT
INTO orders
VALUES('1025', 'may', 'c007','a01','p02',1000,450);
```

执行结果是什么?原因是什么?

```
违反了 PRIMARY KEY 约束'PK__orders__023D5A04'。不能在对象
'dbo.orders'中插入重复键。
```

(2) 要求 customers 表中的 discnt 值不高于 15.0

```
ALTER TABLE customers
ADD CONSTRAINT discnt_max
CHECK (discnt<=15.0);
```

(3) 执行以下语句，查看执行结果，并分析原因。

```
UPDATE customers
SET discnt=20
WHERE cid='c002';
```

将 discnt 设置为了 20，违反了上一题设置的 discnt_max 约束，系统拒绝执行。

(4) 要求 orders 表中 qty 列的值必须大于或等于 0。

```
ALTER TABLE orders
ADD
CONSTRAINT qtyck CHECK (qty>=0),
CONSTRAINT dollarsck CHECK (dollars>=0);
```

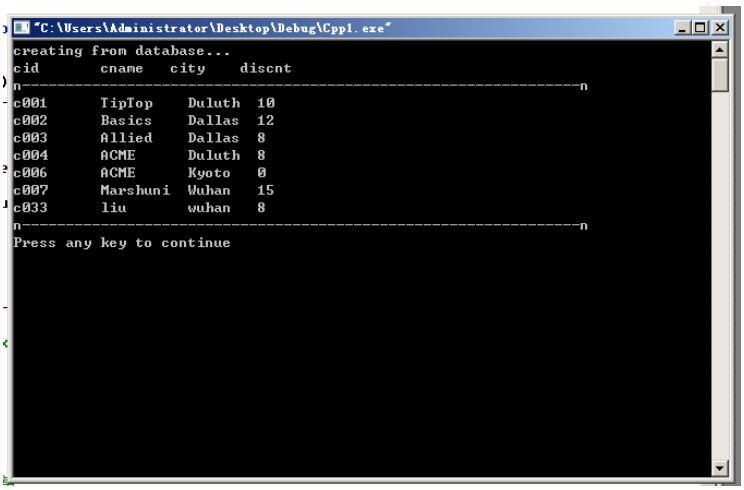
(5) 定义 CAP 数据库中的参照完整性。

```
ALTER TABLE orders
ADD
CONSTRAINT cidref FOREIGN KEY(cid) REFERENCES customers,
CONSTRAINT aidref FOREIGN KEY(aid) REFERENCES agents,
CONSTRAINT pidref FOREIGN KEY(pid) REFERENCES products;
```

数据库实验报告 – 使用 VC 访问数据库

一、使用 VC 访问数据库

(1) 编程实现与 customers 表的连接，获取表中数据



口做准备