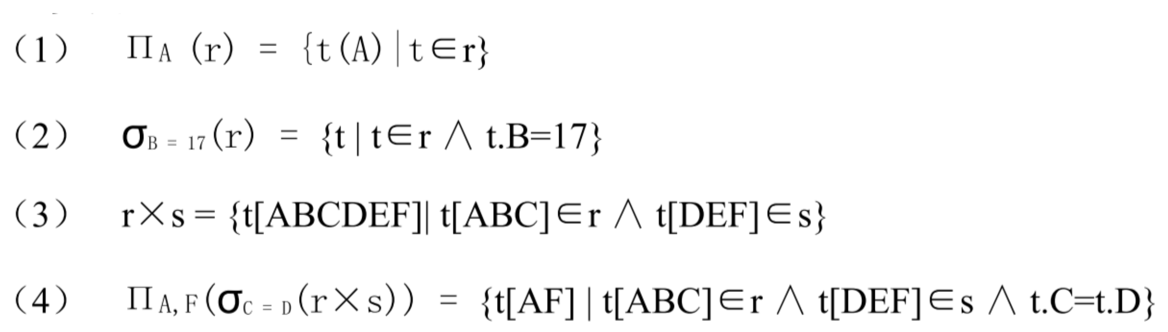
数据库原理常见操作题型

关系演算和关系代数

元组关系演算表达式

R=(A,B,C) S=(D,E,F) 设关系 r(R)和 s(S)已知，分别给出与下列表达式等价的元组关系演算表达式：

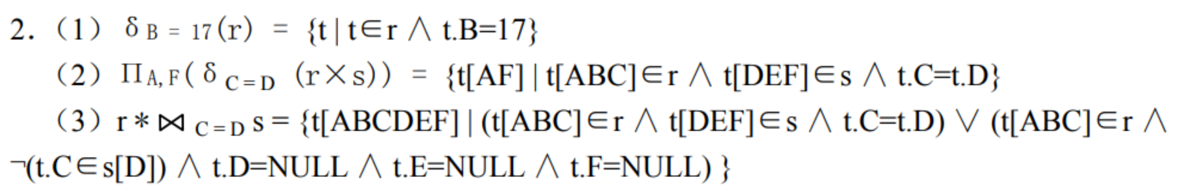


（1） $\Pi_A(r) = \{t(A) \mid t\in r\}$

（2） $\sigma_{B=17}(r) = \{t \mid t\in r \wedge t.B=17\}$

（3） $r\times s = \{t[ABCDEF] \mid t[ABC]\in r \wedge t[DEF]\in s\}$

（4） $\Pi_{A,F}(\sigma_{C=D}(r\times s)) = \{t[AF] \mid t[ABC]\in r \wedge t[DEF]\in s \wedge t.C=t.D\}$



2.（1） $\delta_{B=17}(r) = \{t \mid t\in r \wedge t.B=17\}$

（2） $\Pi_{A,F}(\delta_{C=D}(r\times s)) = \{t[AF] \mid t[ABC]\in r \wedge t[DEF]\in s \wedge t.C=t.D\}$

（3） $r * \bowtie_{C=D} s = \{t[ABCDEF] \mid (t[ABC]\in r \wedge t[DEF]\in s \wedge t.C=t.D) \vee (t[ABC]\in r \wedge \neg(t.C\in s[D]) \wedge t.D=NULL \wedge t.E=NULL \wedge t.F=NULL)\}$

SQL

> 结尾的”； “记得加

预定了**所有**特定内容的人

使用了**EXCEPT**

```
SELECT S.sname FROM Saiillors S
WHERE NOT EXISTS (
    (SELECT B.bid FROM Boats B
    WHERE B.color ='red')
    EXCEPT
    (SELECT R.bid FROM Reserves R
     WHERE R.sid=S.sid)
)
```

不使用**EXCEPT**

```
SELECT S.sname FROM Sailors S
WHERE NOT EXISTS(
    SELECT B.bid FROM Boats B
    WHERE B.color ='red' AND NOT EXISTS(
        SELECT R.bid FROM Reserves R
        WHERE R.bid = B.bid AND R.sid = S.sid
     )
)
```

仅被使用一次 的元组以及其信息

用 Group by

```
Select A.room number, (Select G.name From Guest G Where G.ID number=A.ID number)
From Accommodation A
Where getYear(A.check-in date)=2016 And getMonth(A.check-in date)=January Group by A.room
number
Having count(G.ID number)=1;
```

- 使用 sub-sql 来获取元组信息

查询秋季学期有 2门以上课成绩为 90 分以上的学生的姓名

```
SELECT SNAME FROM STUDENT
WHERE SNO IN(
  SELECT SNO FROM SC
  WHERE GRADE>90 AND CNO IN (
      SELECT CNO FROM COURSE
    WHERE SEMESTER='  秋'
  ) GROUP BY SNO HAVING COUNT(*)>=2
);
```

查询所学每一门课程成绩 均高于等于 该课程平均成绩的学生的姓名及 相应课程号

- 使用 不存在 小于来写较为容易

```
SELECT SNAME, CNO
FROM STUDENT, SC
WHERE STUDENT.SNO = SC.SNO AND STUDENT.SNO NOT IN(
    SELECT SNO FROM SC SCX
    WHERE SCX.GRADE < (SELECT AVG(GRADE)
                    FROM SC SCY
                    WHERE SCY.CNO = SCX.CNO
                   )
    );
```

查秋季学期有一门以上课程考 90 以上的学生的姓名

连接查询

```
SELECT sname
FROM Student S, Course C, Enroll E
WHERE S.sid=E.sid AND E.cid=C.cid AND E.grade>90 AND C.semester='秋季' ;
```

嵌套查询

```
SELECT sname
FROM Student S
WHERE S.sid IN (
  SELECT E.sid
  FROM Course C, Enroll E
WHERE E.cid=C.cid AND E.grade>90 AND C.semester='秋季' );
```

关联嵌套

```
SELECT sname
 FROM Student S
 WHERE EXISTS (
   SELECT E.sid
   FROM Course C, Enroll E
   WHERE S.sid=E.sid AND E.cid=C.cid AND E.grade>90 AND
 C.semester='秋季' );
```

查询所学每一门课程成绩均不低于该门课程平均成绩的学生的姓名及相应课程号

```
SELECT S.sname, E.cid
FROM Enroll E, Student S
WHERE E.sid = S.sid AND
          S.sid NOT IN ( SELECT sid
                         FROM Enroll E1
                         WHERE grade < (SELECT AVG(grade)
                                        FROM Enroll E2
                                        WHERE E2.cid = E1.cid));
```

```
SELECT S.sname, E.cid
FROM Enroll E, Student S
WHERE E.sid = S.sid AND NOT EXISTS
          ( SELECT E1.cid
            FROM Enroll E1
            WHERE E1.sid = S.sid AND
                  grade < (SELECT AVG(grade)
                           FROM Enroll E2
                           WHERE E2.cid = E1.cid));
```

按医生职称查询 2019 年各职称级别完成手术次数最多的医生姓名及其完成的手术次数

```
SELECT 职称，姓名，手术次数
FROM 医生，（
     SELECT 医生编号，COUNT（*）AS 手术次数
     FROM 医生，手术医生安排 AS A
     WHERE 手术日期.GetYear（）=2019 and 医生.医生编号=手术医生安排.医生编号
     and NOT EXISTS
     (SELECT * FROM 医生，手术医生安排 AS B
     WHERE 医生.医生编号=手术医生安排.医生编号 and 手术日期.GetYear（）
     =2019 and COUNT(B.医生编号) > COUNT(A.医生编号) and A.职称=B.职称
     GROUP BY 医生编号)
     GROUP BY 医生编号
     } AS S
     WHERE 医生.医生编号=S.医生编号;
```

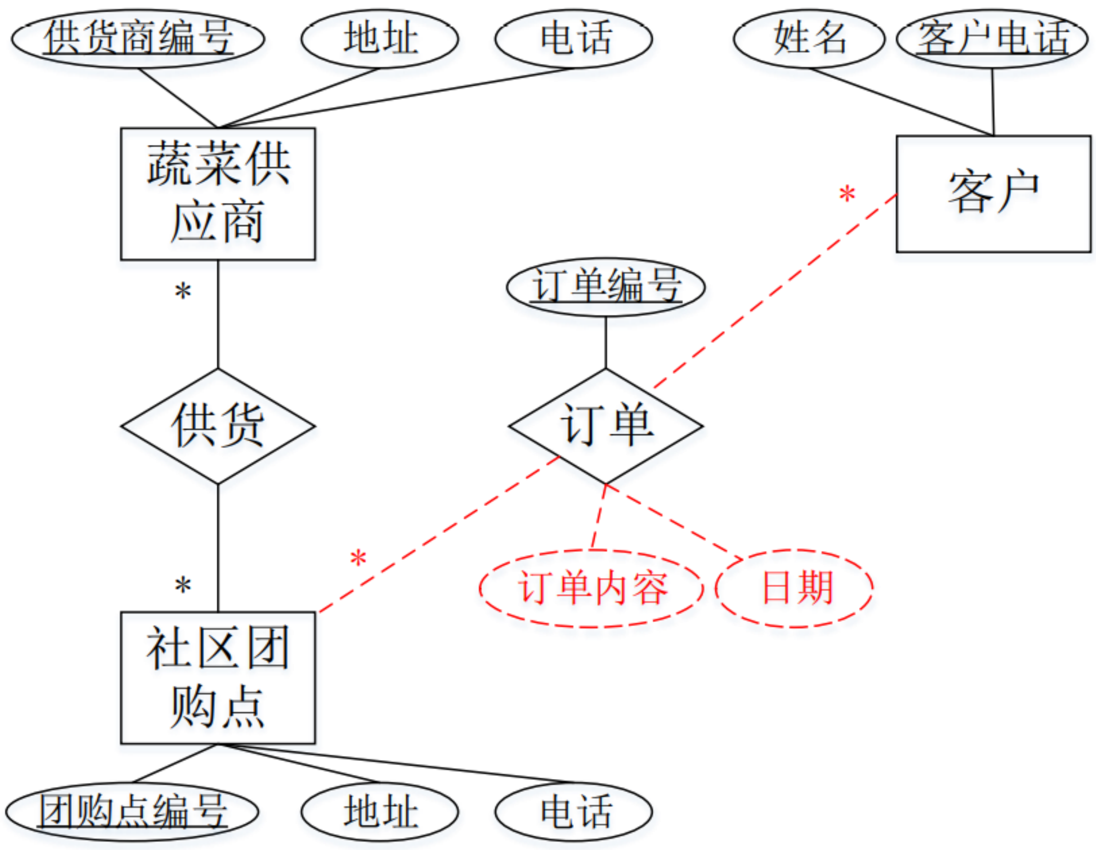
触发器

> 见书本P191

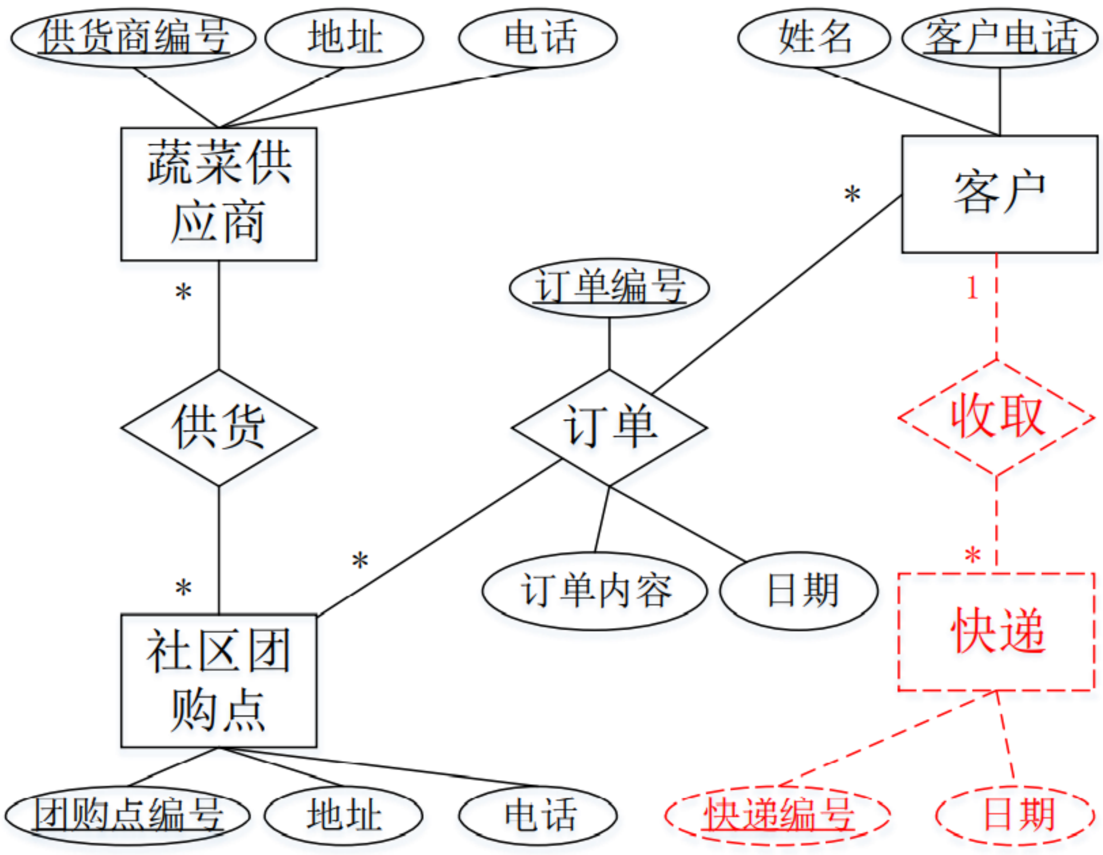
```
Create Trigger insertRollback
Before insert on Accommodation
Referencing New as A
For each statement
When A.check-out date=Null Rollback;
```

```
Create trigger checkReserves
Before insert on reserves
Referencing New as N
For each row
When (GetMon(N.say) in {7,8}) and (N.sid not in (
   select sid from reserves
   group by sid
   having count(*)>9)
)Rollback;
```

设计Schema E-R 图



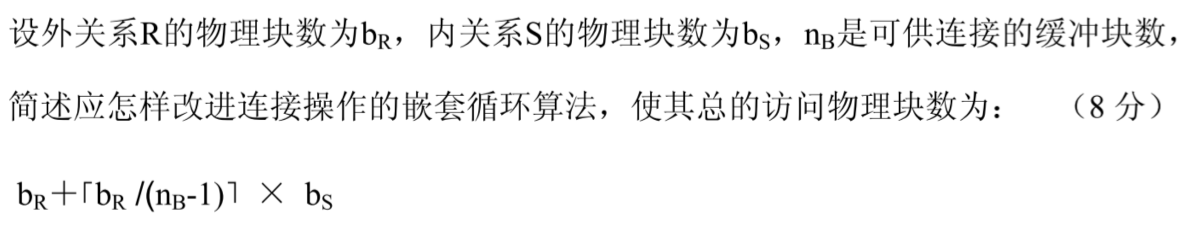
供货商编号
地址
电话
姓名
客户电话
蔬菜供应商
客户
*
订单编号
供货
订单
*
*
订单内容
日期
*
社区团购点
团购点编号
地址
电话



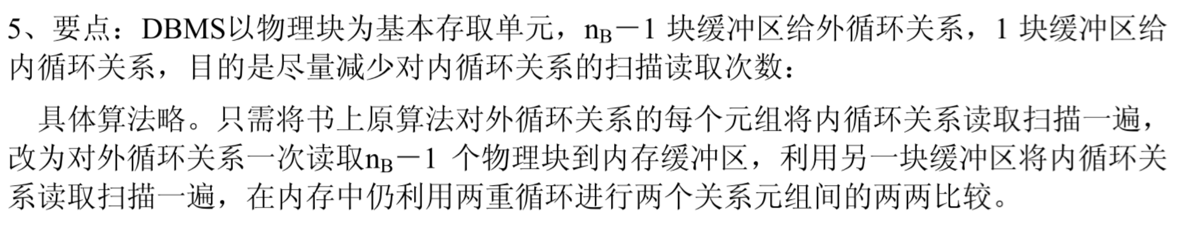
供货商编号
地址
电话
姓名
客户电话
蔬菜供应商
客户
*
*
订单编号
1
供货
订单
收取
*
*
*
订单内容
日期
快递
社区团购点
团购点编号
地址
电话
快递编号
日期

操作优化

嵌套循环


设外关系R的物理块数为b_R，内关系S的物理块数为b_S，n_B是可供连接的缓冲块数，简述应怎样改进连接操作的嵌套循环算法，使其总的访问物理块数为：　（8 分）

$b_R + \lceil b_R /(n_B-1) \rceil \times b_S$


5、要点：DBMS以物理块为基本存取单元，n_B-1 块缓冲区给外循环关系，1 块缓冲区给内循环关系，目的是尽量减少对内循环关系的扫描读取次数；

具体算法略。只需将书上原算法对外循环关系的每个元组将内循环关系读取扫描一遍，改为对外循环关系一次读取n_B-1 个物理块到内存缓冲区，利用另一块缓冲区将内循环关系读取扫描一遍，在内存中仍利用两重循环进行两个关系元组间的两两比较。

嵌入式SQL

> 书本 P77

```
Declare Cursor CR1
As
select reserves.bid, reserves.day, boats.bname from reserves, boats where reserves.bid=boats.bid
and sid=:id and bid= :bid
...
Open cursor CR1
While(true)
{
...
Fetch ... into...
...
}
Close CR1
```

- 得分点 `定义游标`，`打开游标`，`取数`，`关闭游标`