（说明：在有些查询中，使用 distinct 对结果元组做去重处理，并不是题目所必须的）

【3.1】Create queries in SQL to answer the requests in Exercise 2.5(a) through (u), at the end of Chapter 2.

(a) Find all (cid, aid, pid) triples for customer, agent, product combinations that are all in the same city. Nothing about orders is involved in this selection.

```
SELECT   cid, aid, pid
FROM   customers c, agents a, products p
WHERE   c.city=a.city and a.city=p.city ;
```

(b) Find all (cid, aid, pid) triples for customer, agent, product combinations that are not all in the same city (any two may be).

```
SELECT   cid, aid, pid
FROM   customers c, agents a, products p
WHERE   c.city<>a.city and a.city<>p.city ;
```

(c) Find all (cid, aid, pid) triples for customer, agent, product combinations, no two of which are in the same city.

```
SELECT   cid, aid, pid
FROM   customers c, agents a, products p
WHERE   c.city<>a.city and a.city<>p.city and c.city<>p.city ;
```

(d) Get cities of agents booking an order from customer c002.

```
SELECT   distinct   a.city
FROM   orders o, agents a
WHERE   o.aid=a.aid and o.cid='c002' ;
```

(e) Get product names ordered by at least one customer based in Dallas through an agent based in Tokyo.

```
SELECT   distinct   p.pname
FROM   customers c, agents a, products p, orders o
WHERE   o.cid=c.cid and o.aid=a.aid and o.pid=p.pid and
          c.city = 'Dallas' and a.city = 'Tokyo' ;
```

(f) Get pids of products ordered through any agent who makes at least one order for a customer in Kyoto. NOTE: The request posed here is not the same as asking for pids of products ordered by a customer in Kyoto.

```
SELECT   distinct   pid   FROM   orders
WHERE   aid   IN   (
      SELECT   o.aid   FROM   orders o, customers c
      WHERE   o.cid=c.cid and c.city='Kyoto' ) ;
```

(g) Display all pairs of aids for agents who live in the same city.

```
SELECT   x.aid, y.aid
FROM   agents x, agents y
WHERE   x.city=y.city and x.aid<y.aid ;
```

(h) Find cids of customers who did not place an order through agent a03.

```
SELECT  cid  FROM  customers
WHERE  cid  NOT IN ( select cid from orders where aid='a03' ) ;
```

【说明】也可以用 not exists 或者 except 谓词来表示此查询：

参考答案 1：

```
SELECT  c.cid  FROM  customers c
WHERE  NOT EXISTS (select * from orders o where o.cid=c.cid and o.aid='a03') ;
```

参考答案 2：

```
(SELECT cid FROM customers) except (SELECT cid FROM orders WHERE aid='a03');
```

(i) Find cids of customers who have the largest discount; separately, find those who have the smallest discount. NOTE: This is quite hard with the operations provided in relational algebra.

① Find cids of customers who have the largest discount

```
SELECT  cid  FROM  customers
WHERE  discnt >= ALL ( select  discnt  from  customers ) ;
```

② Find cids of customers who have the smallest discount

```
SELECT  cid  FROM  customers
WHERE  discnt <= ALL ( select  discnt  from  customers ) ;
```

(j) Find cids of customers who order all products.

```
SELECT  c.cid  FROM  customers c  WHERE  NOT EXISTS (
    SELECT  *  FROM  products p  WHERE  NOT EXISTS (
        SELECT  *  FROM  orders o  WHERE  o.cid=c.cid  and  o.pid=p.pid ) ) ;
```

(k) Find pids of products ordered through agent a03 but not through agent a06.

```
SELECT  pid  FROM  orders
WHERE  aid='a03' and pid NOT IN (SELECT pid FROM orders WHERE aid='a06') ;
```

(l) Get pnames and pids of products that are stored in the same city as one of the agents who sold these products.

```
SELECT  p.pid, p.pname
FROM  products p, orders o, agents a
WHERE  o.pid=p.pid and o.aid=a.aid and p.city=a.city ;
```

(m) Get aids and anames of agents with aname beginning with the letter 'N' who do not place orders for any product in Newark.

```
SELECT  aid, aname  FROM  agents a
WHERE  aname LIKE 'N%'  and  NOT EXISTS (
      SELECT  *  FROM  orders o, products p
      WHERE  o.aid=a.aid and o.pid=p.pid and p.city='Newark' ) ;
```

(n) Get cids of customers who order both product p01 and product p07.

参考答案 1：

```
(SELECT cid FROM orders WHERE pid='p01') INTERSECT
    (SELECT cid FROM orders WHERE pid='p07') ;
```

参考答案 2：

```
SELECT cid FROM orders x, orders y WHERE x.cid=y.cid and x.pid='p01' and y.pid='p07' ;
```

(o) Get names of agents who place orders for all products ordered by customer c002.

```
SELECT  a.aname  FROM  agents a
WHERE  NOT EXISTS (
    SELECT  *  FROM  orders x
    WHERE  x.cid='c002'  and  NOT EXISTS (
        SELECT  *  FROM  orders y  WHERE  y.aid=a.aid  and  y.pid=x.pid ) ) ;
```

(p) Get names of agents who place orders for all products that are ordered by any customer at all. (Hint: The phrase 'any customer at all' means the same as 'some customer.')

```
SELECT  aname  FROM  agents a  WHERE  NOT EXISTS (
    SELECT  *  FROM  orders x  WHERE  NOT EXISTS (
        SELECT  *  FROM  orders y  WHERE  y.aid=a.aid and y.pid=x.pid ) ) ;
```

(q) Get (cid, aid, pid) triples for customer, agent, product combinations so that at most two of them are in the same city. (Is this equivalent to any of the first three queries of this exercise, (a), (b), or (c)?)

参考答案 1：

```
SELECT  c.cid, a.aid, p.pid  FROM  customers c, agents a, products p
WHERE  c.city<>a.city or a.city<>p.city ;
```

参考答案 2：

```
(SELECT cid, aid, pid FROM customers, agents, products)  EXCEPT
    ( SELECT  c.cid, a.aid, p.pid
      FROM  customers c, agents a, products p
      WHERE  c.city=a.city and a.city=p.city ) ;
```

(r) Get pids of products ordered by all customers who place any order through agent a03.

参考答案 1：

```
SELECT  pid  FROM  products p  WHERE  NOT EXISTS (
    SELECT  *  FROM  orders x  WHERE  x.aid='a03'  and  NOT EXISTS (
        SELECT  *  FROM  orders y
        WHERE  y.pid=p.pid  and  y.cid=x.cid ) ) ;
```

参考答案 2：

```
SELECT  pid  FROM  products p
WHERE  NOT EXISTS (
    SELECT  *  FROM  customers c
    WHERE  c.cid  IN  ( SELECT cid FROM orders WHERE aid='a03' )
        and  NOT EXISTS (
            SELECT  *  FROM  orders o
            WHERE  o.pid=p.pid  and  o.cid=c.cid ) ) ;
```

(s) Get aids of agents who place individual orders in dollar value greater than $500 for customers living in Kyoto.

```
SELECT  o.aid  FROM  orders o, customers c
WHERE  o.cid=c.cid  and  c.city='Kyoto'  and  o.dollars>500 ;
```

(t) Give all (cname, aname) pairs where the customer places an order through the agent.

```
SELECT  c.cname,  a.aname
FROM  customers c,  agents a,  orders o
WHERE  o.cid=c.cid  and  o.aid=a.aid ;
```

(u) [HARD] Get cids of customers who order all their products through only one agent.

参考答案 1：

```
SELECT  distinct x.cid  FROM  orders x
WHERE  NOT EXISTS(SELECT * FROM orders y WHERE y.cid=x.cid and y.aid<>x.aid) ;
```

参考答案 2：

```
(SELECT  distinct  cid  FROM  orders) EXCEPT
    (SELECT x.cid FROM orders x, orders y WHERE y.cid=x.cid and y.aid<>x.aid) ;
```

参考答案 3：

```
SELECT  cid
FROM  orders
GROUP BY  cid
HAVING  count(distinct aid)=1 ;
```

【3.2】Use one of the All-or-Any predicates in (a) and (b) below.

(a) Retrieve aid values of agents who receive more than the minimum percent commission (column name: percent).

```
SELECT aid FROM agents WHERE percent > ANY (select percent from agents) ;
```

(b) Retrieve aid values of agents who receive the maximum percent commission.

```
SELECT aid FROM agents WHERE percent >= ALL (select percent from agents) ;
```

(c) Explain why the following query fails to answer request (a) above, although it retrieves the right rows from the agents table of Figure 2.1:

```
select aid from agents where percent > 5;
```

对于如图 2.2 所示的 agents 表，所有供应商的最低折扣为 5，执行该查询能够得到 request (a)想要的结果元组；当 agents 表中的元组取值情况发生变化时，特别是当表中的 minimum percent 发生变化时，该查询只是返回折扣大于 5 的供应商，并不是“折扣大于最小折扣”的供应商。

AGENTS

aid	aname	city	percent
a01	Smith	New York	6
a02	Jones	Newark	6
a03	Brown	Tokyo	7
a04	Gray	New York	6
a05	Otasi	Duluth	5
a06	Smith	Dallas	5

Figure 2.1 & 2.2

【3.3】Recall that the two predicates IN and =SOME have the same effect (as explained in Example 3.4.8).

(a) Given this, explain why the predicates NOT IN and <>ANY (not equal any) do not have the same effect, but that <>ALL must be used to have the effect of NOT IN.

答：这三个谓词的语义如下：

result / expression	true	false
expr NOT IN (Subquery)	Expr is not found in the set returned by the Subquery.	Expr is an element of the set returned by the Subquery.
expr <>ALL (Subquery)	All elements of the set returned by the Subquery are not equal to expr.	At least one element of the set returned by the Subquery is equal to expr.
expr <>ANY (Subquery)	At least one element of the set returned by the Subquery is not equal to expr.	All elements of the set returned by the Subquery are equal to expr.

从上表中可以看出，NOT IN 和<>ALL 的语义是完全一样的，但它们和<>ANY 的语义不同。

另外，当涉及到空值(the value of expr is NULL)或空集(the set returned by the Subquery is empty.)问题时，需要单独定义它们的处理规则：

	expr is NULL	the set returned by the Subquery is empty
expr IN (Subquery)	false	false **
expr NOT IN (Subquery)	false	true **
expr θ ALL (Subquery) *	false	false ***
expr θ ANY (Subquery) *	false	false ***

说明：

*：θ可以是>, >=, <, <=, =, <>等六种比较运算中的任意一种，SOME 和 ANY 的语义是一样的；

**：左式 expr 的取值不是 NULL，而子查询的结果集合是空集，按照谓词 IN 和 NOT IN 的原始语义进行判断；

***：在数据库系统中，如果一个子查询的结果集是空集，那么所有基于该子查询的量化比较的判断结果都是 false。

(b) Execute the two queries of Example 3.4.13 with predicates NOT IN and <>ALL. Then execute the query with the predicate <>ALL replaced by <>ANY and state in words what will be retrieved as a result.

答：例 3.4.13 是“查询从没有通过 a05 号供应商去购买过商品的顾客的名字”，如果使用<>ALL 谓词，该查询可表示如下：

```
SELECT cname FROM customers
WHERE cid <>ALL (SELECT cid FROM orders WHERE aid = 'a05') ;
```

其执行过程如下：1）先执行内部的子查询，得到一个顾客编号的集合 S，其中的每一位顾客，都是曾经通过 a05 供应商去购买过商品；2）用集合 S 去执行外部查询，返回所有满足下述条件的顾客 c 的名字：顾客 c 的编号与集合 S 中的所有元素(顾客编号)都不相等。

如果将其中的谓词<>ALL 替换为<>ANY，则得到如下的查询 q_0：

```
SELECT cname FROM customers
WHERE cid <>ANY (SELECT cid FROM orders WHERE aid = 'a05') ;
```

查询 q_0 的执行过程如下：1）先执行内部的子查询，得到一个顾客编号的集合 S，其中的每一位顾客，都是曾经通过 a05 供应商去购买过商品；2）用集合 S 去执行外部查询，返回所有满足下

述条件的顾客 c 的名字：在集合 S 中至少存在一个元素(顾客编号)，与客户 c 的编号是不相等。综上所述，查询 q_0 的执行结果如下：

1) 如果“不存在通过 a05 供应商去购买过商品的顾客”，即子查询的查询结果为空集，则 q_0 返回一个空的结果集合；（基于前面的说明***）
2) 如果“a05 只向唯一的一位顾客 c 销售过商品”，即子查询只返回单个客户的编号（顾客 c 的编号），则 q_0 返回 customers 表中除顾客 c 之外的其他所有顾客的名字；
3) 如果“a05 向多位不同的顾客销售过商品”，即子查询返回的结果集中包含多个不同的客户编号，则 q_0 返回 customers 表中的所有顾客的名字。

(c) Execute the query of Example 3.4.7, which uses the predicate <=ALL. What quantified comparison predicate would you substitute in this query to retrieve exactly those rows that are not retrieved using <=ALL? Demonstrate by execution that this query returns the proper rows.

答：例 3.4.7(Find aid values of agents with a minimum percent commission.)是要查询具有最小 percent 值的供应商的编号，其补集是“percent 值不是最低的供应商的编号”，也可理解为：查询满足下述条件的顾客 c 的编号：至少存在一位顾客 t，满足 c.percent>t.percent。因此，可以用谓词>ANY 来代替例 3.4.7 中的谓词<=ALL，将查询改写如下：

```
SELECT aid FROM agents
WHERE percent >ANY ( SELECT percent FROM agents ) ;
```

(d) Refer to the definitions of predicates <ANY and <ALL and explain why these predicates have the same meaning as < in the condition expr<(Subquery), when the Subquery returns a single element.

答：

expr<ANY(Subquery) is true if and only if at least one element of the set returned by the Subquery is more than expr.

expr<ALL(Subquery) is true if and only if all elements of the set returned by the Subquery is more than expr.

当 Subquery 返回的结果集中只含单个结果元素时，‘at least one element of the set...’和‘all elements of the set...’具有相同的语义，且等价于 expr<(Subquery)。

【3.4】

(a) Compose an SQL statement that solves the problem of Example 3.4.1 without using a Subquery. (The FROM clause should reference all tables involved.)

答：Example 3.4.1: Get cid values of customers who place orders with agents in Duluth or Dallas.
不使用 Subquery，该查询可表示如下：

```
SELECT o.cid
FROM orders o, agents a
WHERE o.aid=a.aid and (a.city='Duluth' or a.city='Dallas') ;
```

(b) Is it always possible to avoid using a Subquery as we did in part (a)? Assume that we have a table S with attributes $A_1,...,A_n$, a table T with attributes $B_1,...,B_m$, and constants c and k, where A_i and B_i are from the same domain, for i=1,2,3, and c and k are both from the same domain as A_2 and B_2. Consider the query

select A_1,..., A_n from S where A_2=k and A_1 in (select B_1 from T where B_2=c);

Rewrite this Select statement to get the same result but without using a Subquery. Don't forget to qualify attributes when needed.

答：Yes.

```
select S.A1, ..., S.An
from S, T
where S.A2 = k and T.B2 = c and S.A1 = T.B1 ;
```

(c) Repeat part (b), rewriting without a Subquery, the query

select A_1,..., A_n from S where A_2=k and A_1 in (select B_1 from T where B_2=c and B_3=S.A_3);

答：

```
select S.A1, ..., S.An
from S, T
where S.A2 = k and T.B2 = c and T.B3 = S.A3 and S.A1 = T.B1 ;
```

【3.5】

Consider the problem to find all (cid, aid) pairs where the customer does not place an order through the agent. This can be accomplished with the Select statement

```
select cid, aid from customers c, agents a
    where not exists (select * from orders x
        where x.cid = c.cid and x.aid = a.aid);
```

Is it possible to achieve this result using the NOT IN predicate in place of the NOT EXISTS predicate with a single Subquery? With more than one Subquery? Explain your answer and demonstrate any equivalent form by execution.

答：可用 NOT IN 谓词改写成如下查询：

```
select cid, aid from customers c, agents a
    where (cid, aid) not in (select cid, aid from orders) ;
```

（略）

【3.6】Look again at the pseudo-code in Figure 3.3.

(a) Write comparable pseudo-code to show how the query of Exercise 3.4(b) can be evaluated without the use of nested loops. You should have two loops performed at distinct times (not one within another)--the first loop, corresponding to the Subquery, placing results (A, values) into a list of values L, and the second loop using a predicate to test if a value of A_1 is in L. We do not have a rigorous definition of what pseudo-code should look like, but try to make it clear what is happening, by analogy with the pseudo-code of Figure 3.3.

答：pseudo-code for Exercise 3.4(b):

```
Set the list L is null;
FOR t FROM ROWS 1 TO LAST OF T
    IF (t.B2=c) THEN place t.B1 into a list of values L
END FOR t
FOR s FROM ROWS 1 TO LAST OF S
    IF (s.A2=k and s.A1 is an element of the list L)
    THEN PRINT OUT SELECT_LIST VALUES: s.A1, ..., s.An
END FOR s
```

(b) Explain why we cannot avoid nested loops in pseudo-code for the query of Exercise 3.4(c). The reason is based on the fact that this query has what is known as a correlated Subquery.

答：在练习 3.4(c)中，是要在关系 S 中查询满足以下条件的元组 s：$s.A_2=k$，并且满足“在关系 T 中至少存在一个元组 t，他们之间满足 $t.B_2=c$ and $t.B_3=s.A_3$ and $t.B_1=s.A_1$”

由于在内层嵌套的子查询中，使用了外部查询中的元组 s，针对外层查询中检查到的每一个元组 s，都要用 $s.A_3$ 参与到内层嵌套子查询的执行，因此该子查询是一个相关子查询。即使将练习 3.4(c)改造成不使用子查询的表连接查询，但其执行过程仍然是两层的嵌套循环(nested loops).

【3.7】

How would we show that SQL Select actually offers all the power of relational algebra? Recall from Section 2.8 that all eight operations of relational algebra (listed at the end of Section 2.5) can be expressed in terms of the five basic operations: UNION, DIFFERENCE, PRODUCT, PROJECT, and SELECT. Consider any two tables R and S existing in an SQL database that have been created using the Create Table statement, which we will call base tables.

Explain how you would use SQL Select to retrieve any of the answer tables:

(a) R UNION S

答：(select * from R) UNION (select * from S) ;

(b) R MINUS S, R TIMES S, R[subset of columns], R WHERE <condition> as in relational algebra. Use only the NOT EXISTS predicate for Subqueries. Assume that R and S have compatible headings where necessary.

答：

1) R MINUS S

设 R 和 S 是一对相容的关系表，$A_1,A_2,...A_k$ 是其中的属性，则‘R MINUS S’可以被表示为：

```
(select * from R) EXCEPT (select * from S) ;
```

也可被表示为：

```
select * from R
where not exists (select * from S
                  where S.A1=R.A1 and S.A2=R.A2 and ... and S.Ak=R.Ak) ;
```

2) R TIMES S

设 $A_1,A_2,...,A_m$ 和 $B_1,B_2,...,B_n$ 分别是关系 R 和 S 中的属性，则‘R TIMES S’可以被表示为：

```
select R.A1, R.A2, ..., R.Am, S.B1, S.B2, ..., S.Bn
from R, S ;
```

3) R[subset of columns]

设 $A_1,A_2,...,A_k$ 是关系 R 的一个属性子集，则‘$R[A_1, A_2, ..., A_k]$’可以被表示为：

```
select  A1, A2, ..., Ak  from  R ;
```

4) R WHERE <condition>

可以被表示为：select * from R where <condition> ;

(c) [HARD] But we are not finished demonstrating the power of the SQL Select, because recursive expressions are possible in relational algebra where recursion is not possible in SQL Select. For example, if R, S and T are all compatible base tables with headings $A_1,...,A_n$, then explain how we can express, using the SQL Select, the relational algebra expression (R UNION S) MINUS T.

答：因为 (R UNION S) MINUS T 等价于 (R MINUS T) UNION (S MINUS T) ，因此可以用 SQL 表示如下：

(select * from R where not exists(select * from T where T.A_1=R.A_1 and ... and T.A_n=R.A_n)) UNION
(select * from S where not exists(select * from S where S.A_1=R.A_1 and ... and S.A_n=R.A_n)) ;

(d) [VERY HARD; REQUIRES MATH BACKGROUND] Prove that if U and V represent any arbitrarily recursive relational algebra expressions achieved by SQL Select statements in terms of base tables, we can also achieve by SQL Select statements the deeper recursions:
U UNION V, U MINUS V, U TIMES V, U[subset of columns], U WHERE <condition>

证：设 U 和 V 所对应的 SQL 查询语句分别是 $SubQuery_U$ 和 $SubQuery_V$ ，其结果关系表的表头分别是 $Head(U)=\{A_1,A_2,...,A_m\}$ 和 $Head(V)=\{B_1,B_2,...,B_n\}$ 。

1) U UNION V, U MINUS V

要求 U 和 V 是相容的关系表，即 Head(U)=Head(V)。因此假设：m=n，且 A_1 对应 B_1，A_2 对应 B_2，...，A_m 对应 B_m，则：

- U UNION V 可以表示为：

(select * from (*Subquery*$_U$) as U(A_1,...,A_m)) UNION (select * from (*Subquery*$_V$) as V(B_1,...,B_m)) ;

- U MINUS V 可以表示为：

方法一：

(select * from (*Subquery*$_U$) as U(A_1,...,A_m)) EXCEPT (select * from (*Subquery*$_V$) as V(B_1,...,B_m)) ;

方法二：

```
select  *  from (Subquery_U) as U(A1,...,Am)
where not exists (
      select  *  from (Subquery_V) as V(B1,...,Bm)
      where  V.B1=U.A1 and ... and V.Bm=U.Am  ) ;
```

2) U TIMES V

```
select U.A1, U.A2, ..., U.Am, V.B1, V.B2, ..., V.Bn
from (Subquery_U) as U(A1,...,Am), (Subquery_V) as V(B1,...,Bm) ;
```

3) U[subset of columns]

```
select  <subset of columns>
from (Subquery_U) as U(A1,...,Am) ;
```

4) U WHERE <condition>

```
select  *  from (Subquery_U) as U(A1,...,Am)  where <condition> ;
```

(e) [VERY HARD] Recall from Theorem 2.8.3 that R DIVIDEBY S can be expressed in terms of projection, difference, and product. Let R stand for the SQL statement “select cid, aid from orders;” and let S stand for “select aid from agents where city=’New York’;”. Then R DIVIDEBY S gives the same answer as Example 3.5.2. Use the formula of 2.8.3 and express this in terms of an SQL Select, then execute the resulting statement and verify that it gives the right answer.

答：Example 3.5.2: Find cids of customers who place orders with ALL agents based in New York.

R: select cid, aid from orders;

S: select aid from agents where city=’New York’;

Theorem 2.8.3: R÷S = R[cid]－((R[cid]×S)－R)[cid]

将上述的除法推导公式用 SQL 来表示，具体步骤如下：

1) (R[cid]×S) 可以表示为：select R.cid, S.aid from R, S

2) (R[cid]×S)－R 可以表示为：

```
select R.cid, S.aid   from R, S
where not exists (select *   from R y   where y.cid=R.cid and y.aid=S.aid)
```

3) ((R[cid]×S)－R)[cid] 可以表示为：

```
select R.cid   from R, S
where not exists (select *   from R y   where y.cid=R.cid and y.aid=S.aid)
```

4) R[cid]－((R[cid]×S)－R)[cid] 可以表示为：

```
(select cid from R)   EXCEPT
    (select R.cid   from R, S
        where not exists (select *   from R y   where y.cid=R.cid and y.aid=S.aid));
```

将 R 和 S 所对应的 SQL 语句代入后得到如下的 SQL 语句：

```
(select R.cid from (select cid, aid from orders) as R(cid, aid))   EXCEPT
    ( select   R.cid
    from   (select cid, aid from orders) as R(cid, aid),
           (select aid from agents where city=’New York’) as S(aid)
    where   not exists (
        select   *
        from   (select cid, aid from orders) as y(cid, aid)
        where   y.cid=R.cid and y.aid=S.aid) );
```

对上述 SQL 语句进行 rewriting 后的结果如下：

```
(select cid from orders)   EXCEPT
    ( select   R.cid
      from   orders as R, agents as S
      where   S.city=’New York’   and
          not exists (
              select   *
              from   orders as y
              where   y.cid=R.cid and y.aid=S.aid) );
```

【3.8】

(a) Write a Select statement with no WHERE clause to retrieve all customer cids and the maximum money each spends on any product. Label the columns of the resulting table: cid, MAXSPENT.

答：select cid, max(dollars) as MAXSPENT from orders group by cid ;

(b) Write a query to retrieve the AVERAGE value (over all customers) of the MAXSPENT of query (a).

答：
```
select   avg(MAXSPENT)
 from   (select cid, max(dollars) as MAXSPENT from orders group by cid) ;
```

(c) Is it possible to solve (b) without the new capabilities presented in Section 3.6 ?

答：可以不用在 FROM clause 中嵌入子查询，具体方法如下：先查出每个客户订单金额 dollars 最高的订单(在该客户 dollars 最大的那些订单中，只取 ordno 最高的哪一份订单，来做最后的平均值统计)，然后对这些客户订单的 dollars 调用 AVG 函数。具体表示如下：

```
select   avg ( dollars )
from   orders o
where   not exists ( select * from orders x
                        where x.cid=o.cid and x.dollars>o.dollars)   and
     o.ordno >= ALL ( select y.ordno from orders y
                        where y.cid=o.cid and y.dollars=o.dollars) ;
```

说明：还有另外一种方法，可以用(a)中的查询来创建一个视图(view)，然后直接在该视图上执行(b)中的统计查询。

【3.10】

(a) Assume we have a small subset C of customers rows (10 rows out of 500,000 customers) determined with a search_condition, WHERE C, and separately we have an sporders table for orders by customers that are approximately (but maybe not exactly) the same subset C. We want to print out a report giving: cid, cname, TOTDOLL, where TOTDOLL is the SUM(dollars) ordered by each customer in sporders. We want to be able to tell when a customer shows up in sporders but not in C (by showing a null cname); but we do not want to see customers (whether in C or not) who have made no orders in sporders (i.e., we very much want to avoid displaying approximately 500,000-10 customers with nulls in TOTDOLL. What SQL would you write?

答：
```
select cid, cname, TOTDOLL
from (select cid, sum(dollars) as TOTDOLL from sporders group by cid)
      left outer join   (select cid, cname from customers where C)   using(cid) ;
```

(b) Now assume that we do want to see all customers in the subset C who have not made sales in sporders, with nulls in TOTDOLL. What SQL would you write ?

答：将上一题中的 left outer join 改为使用 right outer join.

【3.11】

Compose and executre SQL statements to perform the following set of tasks, using the Basic SQL standard we have provided (outside of section 3.6), and optionally note ways to write them using the extended features of section 3.6.

(a) **For each agent taking an order, list the product pid and the total quantity ordered by all customers from that agent.**(检索每个经销商销售每一种产品的总数量)

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

(b) **We say that a customer x orders a product y in an average quantity A if A is avg(qty) for all orders rows with cid=x and pid=y. Is it possible in a single SQL statement to retrieve cid values of customers who order all the products that they receive in average quantities (by product) of at least 300?**（*检索符合下述要求的客户的编号：*在该客户订购过的所有商品中，每一种商品的平均每笔订单的订购数量均达到或超过 **300**）

```
SELECT  cid
FROM  ( SELECT  cid, pid, avg(qty)
          FROM  orders
          GROUP BY  cid, pid )  as  T(cid, pid, q_avg)
GROUP BY  cid
HAVING  min(q_avg)>=300 ;
```

(c) **Get aid values of agents not taking orders from any customer in Duluth for any product in Dallas.**（检索没有为居住在 **Duluth** 的任何客户订购过任何商品的经销商的编号）

```
SELECT  aid  FROM  agents  WHERE  aid  not in  (
      SELECT  aid
      FROM  customers c, products p, orders o
      WHERE  c.cid=o.cid  and  p.pid=o.pid  and
             c.city='Duluth'  and  p.city='Dallas' ) ;
```

(d) **Get aid values of agents who order at least one common product for each customer who is based in Duluth or Kyoto.**（检索为居住在 **Duluth** 和 **Kyoto** 的所有客户订购过同一种商品的经销商的编号）

```
SELECT  aid  FROM  agents  a
WHERE  not exists (
    SELECT  *  FROM customers c
    WHERE  (c.city='Duluth' or c.city='Kyoto')  and
       not exists (  SELECT  *  FROM orders o
                     WHERE  o.aid=a.aid  and  o.cid=c.cid ) ) ;
```

(e) **Get cid values of customers who make orders only through agent a03 or a05.**（检索仅通过 **a03** 和 **a05** 两个经销商订购过商品的客户编号）

```
SELECT  distinct  cid  FROM  orders
WHERE  cid  not in (
      SELECT  cid  FROM  orders
      WHERE  aid <> 'a03'  and  aid <> 'a05' ) ;
```

【注】结果包括只通过 **a03** 购买过商品、只通过 **a05** 购买过商品、以及只通过 **a03** 和 **a05** 两个供应商购买过商品的客户；使用 **distinct** 是为了去除重复的结果 **cid**（也可不用 **distinct**，保留重复的 **cid**）

(f) Get pid values of products that are ordered by all customers in Dallas.（检索居住在 Dallas 的所有客户都订购过的商品编号）

```
SELECT  pid  FROM  products p
WHERE  not exist (
      SELECT  *  FROM  customers c
      WHERE  c.city = 'Dallas'  and  not exists (
            SELECT  o.cid  FROM  Orders o
            WHERE  o.cid=c.cid  and  o.pid = p.pid ) ) ;
```

(g) Find agents with the highest percent (percent commission), using the max set function.（检索享有最高佣金比率的经销商）

```
SELECT  *
FROM  Agents
WHERE  percent  IN  (SELECT  max(percent)  FROM  Agents ) ;
```

(h) In the agents table, delete the row with the agent named Gray, print out the resulting table in full, then put Gray back, using the Insert statement.（在 agents 表中删除名字为 Gray 的供应商元组，然后再用 INSERT 命令恢复被删除的 Gray 元组）

```
SELECT  *  FROM  agents  WHERE  aname =  'Gray' ;
Delete  FROM  Agents  WHERE  aname =  'Gray' ;
Insert  into  agents  values(.....);
Commit;
```

（注：省咯号处应是被删除的 Gray 元组的值）

(i) Use the Update statement to change Gray's percent to 11. Then change it back.

```
Update  pgents  Set  percent = 11  WHERE  aname =  'Gray' ;
Rollback;
```

（注：这里用事务回滚来撤销之前的 UPDATE 操作的修改结果）

(j) Use a single Update statement to raise the prices of all products warehoused in Duluth or Dallas by 10%. Then restore the original values by rerunning the procedure that you originally used to create and load the products table.

```
Update  products
Set  price = price * 1.1
WHERE  city =  'Duluth'  or city =  'Dallas' ;
```

（注：表中数据的恢复处理步骤被省略了）

(k) Write an SQL query to retrieve cid values for customers who place at least one order, but only through agent a04. On the same line with each cid, your query should list the total dollar amount of orders placed.（检索仅仅通过 a04 号经销商订购过商品的客户编号，并给出每个客户的订购总金额）

```
SELECT  cid, sum(dollars)
FROM  Orders
WHERE  cid  not in ( SELECT  cid  FROM  orders  WHERE aid<>'a04' )
GROUP BY  cid ;
```

(l) **Write an SQL query to get aid and percent values of agents who take orders from all customers who live in Duluth. The aid values should be reported in order by decreasing percent. (Note that if percent is not retrieved in the SELECT list, we cannot order by these values.)**（检索为居住在 **Duluth** 的所有客户订购过商品的经销商的编号及其佣金百分比，并按照佣金百分比的降序输出查询结果）

```
SELECT aid, percent FROM agents a
WHERE not exist (
      SELECT * FROM customers c
      WHERE c.city = 'Duluth' and c.cid not in (
            SELECT o.cid FROM orders o WHERE o.aid = a.aid ) )
ORDER BY percent desc ;
```

(m) **Write an SQL query to get pid values of products ordered by at least one customer who lives in the same city as the agent taking the order.**（*检索符合下述条件的商品的编号：至少有一个客户通过与该客户位于同一个城市的经销商订购过该商品*）

```
SELECT O.pid
FROM orders O, customers C, agents A
WHERE O.cid = C.cid and O.aid = A.aid and C.city = A.city ;
```

【3.13】

Here is an exercise to investigate the question: What SQL queries are guaranteed to return answer tables with no duplicate rows without use of the DISTINCT keyword? This can be important, because using the DISTINCT keyword when it isn't necessary can cause unwanted resource use. We present a series of rules for queries that will not return duplicate rows and ask you to (i) explain why each rule works, and (ii) give an example of a query that returns duplicate rows when the rule fails. In parts (a) and (b), we consider only queries with no Subqueries and no GROUP BY clauses.

(a) In a query with only one table in the FROM clause, there will be no duplicate rows returned if the column names in the select list form a superkey for the table.

答：

(i) 在没有 Subquery 和 GROUP clause 的情况下，该查询就是一个单表上的“元组选择”查询，由于查询的目标属性集合是原关系表的一个 superkey（同理，他们也是查询结果关系的一个 superkey），因此，查询的结果元组与原关系表中的元组之间存在“1 对 1”的映射关系。关系表中不存在重复的元组（因为有 superkey），因此，查询结果中也不会存在重复元组。

(ii) 假设要查询通过 a01 供应商去购买过商品的客户编号，其 SQL 语句如下：select cid from orders where aid='a01'; 由于在 select clause 中没有使用 DISTINCT 谓词且不含 orders 表的关键字 ordno，因而，其查询结果中通常会含有很多重复的 cid 值。

(b) In a query with multiple tables in the FROM clause, there will be no duplicate rows returned if the column names in the select list contain subsets that form superkeys for each of the tables involved.

答：

(i) 在没有 Subquery 和 GROUP clause 的情况下，该查询是一个多表连接查询，其执行过程是一个多层的嵌套 FOR 循环：从每一张表中各取一个元组，如果他们的取值能够使得 where clause 中的条件成立，则投影生成一条结果元组。由于在查询的结果属性中包含着各个表的 superkey，因此，每一条结果元组在每个关系表中只能对应着唯一的一条元组。而在一张关系表中，任取两个互不相同的元组，如果他们都能参与到结果元组的投影中去，那么他们投影生成的结果元组肯定也互不相同。因此，最终的查询结果中不会存在重复的结果元组。

(ii) 可能返回重复结果元组的例子：

```
select cid, cname, aid, aname
from orders o, customers c, agents a
where o.cid=c.cid and o.aid=a.aid;
```

(c) Is it true that no query with a GROUP BY clause will have duplicate rows? Explain why or give a counterexample.

答：当使用 GROUP BY 执行分组统计查询时，其结果集中不可能出现重复的结果元组。原因如下：

当使用 GROUP BY 进行分组统计查询时，

1）在分组属性上的取值相同的元组，将被划分到同一个 group 中去；而在分组属性上的取值不同的元组，则被划分到不同的 group 中去。

2）每一个 group 将生成一条结果元组。在也有 HAVING clause 的情况下，只有那些能够使得 HAVING 条件成立的 group，才会被用于最终结果元组的生成。

3）在 select clause 中，除了可能的统计结果外，‘必须包含且仅包含’对所有分组属性的投影。

基于上述的 3 点原因，可以确信，在最终返回的查询结果中，不会出现重复的结果元组。

(d) In Select statements that contain Subqueries in their WHERE clause, queries should be guaranteed unique rows in at least the same situations as (a) and (b) above. The transformation of Exercise 3.4(b) from a Subquery form to a join form without a Subquery may result in duplicate rows, however, unless the DISTINCT keyword is used before the select list. Give an example where a Select statement without duplicates gains duplicates during such a transformation.

答：假设我们要查询“通过 a01 号供应商去购买过商品的客户的编号和名称”，我们可以有两种不同的查询表示方法：

Q_1 （不使用 Subquery）	select c.cid, c.cname from customers c, orders o where c.cid=o.cid and o.aid='a01';
Q_2 （不使用 Subquery）	select c.cid, c.cname from customers c where c.cid in (select o.cid from orders o where o.aid='a01');

在查询 Q_1 中，结果属性中不含 orders 表的关键字，因而可能存在重复的结果元组。原因是：如果某个客户与 a01 之间有多份订单，那么该客户元组将与他和 a01 之间的每一份订单都会连接形成一条结果元组，并投影出相同的的 cid 和 cname 值。

在查询 Q_2 中，符合本小题的假设，其外部查询是一个在 customers 表上的单表查询，且结果属性中含 customers 表的关键字 cid。在 Q_2 的执行过程中，对 customers 表只进行一遍扫描(FOR 循环)，每一个客户元组也只被检查一次，如果满足条件则投影出其 cid 和 cname。由于结果含顾客表的关键字属性 cid，因而针对不同的满足查询条件的顾客元组，将会被投影为不同的结果元组。