The data model accompanying Lacroix and Pirotte's queries is shown in Figure R2-1. You will notice that the complexity of the real world has been reduced; a sale and a delivery are assumed to have only one item. The corresponding relational model is shown in Table R2-1, and some data that can be used to test your SQL queries appear at the end of this chapter (see page 311).

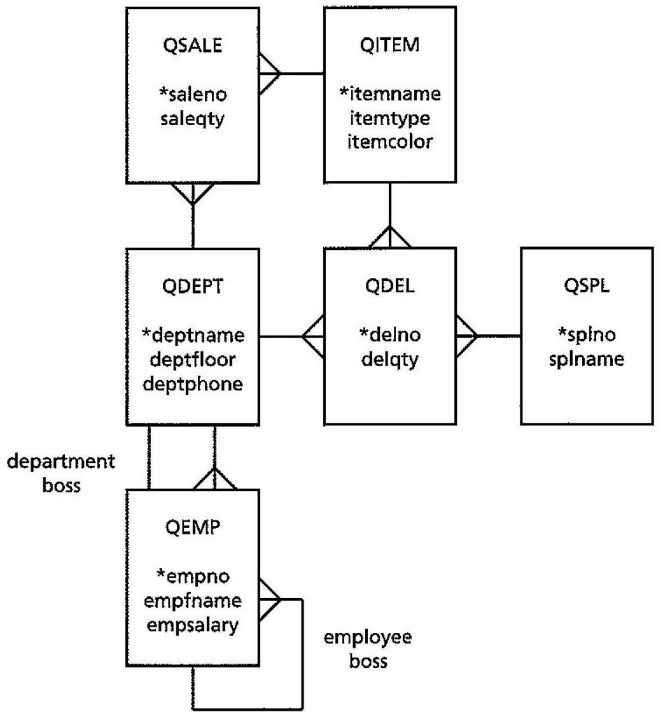

Figure R2-1. Data model

Table R2-1. Relational tables

qsale			
saleno	saleqty	*itemname*	*deptname*

qitem		
itemname	itemtype	itemcolor

qdel				
delno	delqty	*itemname*	*deptname*	*splno*

qspl	
splno	splname

qdept			
deptname	deptfloor	deptphone	*empno*

qemp				
empno	empfname	empsalary	*deptname*	*bossno*

1. A slow full toss

In cricket, a slow full toss is the easiest ball to hit. The same applies to this simple query.

- ***Find the names of employees in the Marketing department.***

```
SELECT empfname
FROM qemp
WHERE deptname = 'Marketing';
```

2. Skinning a cat

Queries often can be solved in several ways. For this query, four possible solutions are presented. Also, we indicate the resource units required to answer the query when using SQL/VM.

- ***Find the items sold by a department on the second floor.***

2.1 Join (545 units)

```
SELECT DISTINCT itemname
FROM qsale, qdept
WHERE qdept.deptname = qsale.deptname AND
      deptfloor      = 2;
```

A join of qsale and qdept is possibly the most obvious way to answer this query.

2.2 In (8)

```
SELECT DISTINCT itemname
FROM qsale
WHERE deptname IN
                 (SELECT deptname
                  FROM qdept
                  WHERE deptfloor =2);
```

Another simple approach is to use an IN clause. First find all the departments on the second floor, and then find a match in qsale. The subquery returns a list of all departments on the second floor. Then use the IN clause to find a match on department name in qsale. Notice that this is the most efficient form of the query.

2.3 Correlated subquery (7,780)

```
SELECT DISTINCT itemname
FROM qsale
WHERE deptname IN
                 (SELECT deptname
                  FROM qdept
                  WHERE qdept,deptname =  qsale.deptname AND
                        deptfloor      =  2);
```

Conceptually, you can think of this correlated query as stepping through qsale one row at a time. The subquery is executed for each row of qsal e, and if there is a match for the current value of qsale. deptname, the current value of itemname is listed.

2.4 Exists (8,080)

```
SELECT DISTINCT itemname
FROM qsale
WHERE EXISTS
           (SELECT *
           FROM qdept
           WHERE qsale.deptname = qdept.deptname AND
                 deptfloor      = 2) ;
```

Conceptually, think of this query as stepping through each row of qsale and evaluating whether the existence test is true. Remember, EXISTS returns *true* if there is at least one row in the inner query for which the condition is true.

The difference between a correlated subquery and existence testing is that a correlated subquery returns either the value of a column or an empty table, while EXISTS returns either *true* or *false*. It may help to think of this version of the query as, "Select item names from sale such that there exists a department relating them to the second floor."

3. Another full toss

- ***Find the names of items sold on floors other than the second floor.***

  This query is a slight adaptation of the second. Just change the equals to a not equals.

```
SELECT DISTINCT itemname
FROM qsale, qdept
WHERE qsale.deptname = qdept.deptname AND
      deptfloor      <> 2;
```

4. Subtracting from all

- ***Find the items sold by no department on the second floor.***

  You may first think this query is the same as the preceding. However, the prior query does not exclude an item that is sold on some other floor in addition to the second floor. For example, if polo sticks are sold on both the third and second floors, they will be reported by the preceding query because they are sold on a floor other than the second floor.

  The correct way to approach this is to first get a list of all items sold on the second floor (the second query) and then subtract this result from all items sold. Clearly, the difference must be all items not sold on the second floor.

```
SELECT DISTINCT itemname
FROM qsale
WHERE itemname NOT IN
            (SELECT DISTINCT itemname
             FROM qsale, qdept
             WHERE qsale.deptname = qdept.deptname AND
                   deptfloor      = 2);
```

5. Dividing

- ***Find the items sold by all departments on the second floor.***

  This is the relational algebra divide or the SQL double NOT EXISTS encountered in Chapter 5. You can think of this query as

  Select items from sales such that there does not exist a department on the second floor that does not sell this item.

  However, it is easier to apply the SQL template described in Chapter 5, which is

```
SELECT targetl
FROM target
WHERE NOT EXISTS
     (SELECT *
      FROM source
      WHERE NOT EXISTS
            (SELECT *
             FROM target-source
             WHERE target-source.targettt = target.targett AND
                   target-source.sourceft = source.sourcett));
```

  The substitutions are straightforward.

targetl	=	itemname
target	=	qitem
source	=	qdept
target-source	=	qsale
target*	=	itemname
source#	=	deptname

Some additional code needs to be added to handle the restriction to items on the second floor. The query becomes

```
SELECT DISTINCT itemname
FROM qitem
WHERE NOT EXISTS
      (SELECT *
       FROM qdept
       WHERE deptfloor =2 AND NOT EXISTS
             (SELECT *
              FROM qsale
              WHERE qsale.itemname = qitem.itemname AND
                    qsale.deptname = qdept.deptname));
```

Here is an alternative approach to formulating the problem:

- ***Find all the items for which the number of second-floor departments that sett them is equal to the total number of departments on the second floor,***

```
SELECT qsale.itemname
FROM qsale, qdept
WHERE qsale.deptname  = qdept.deptname AND
      qdept.deptfloor = 2
GROUP BY qsale.itemname
   HAVING COUNT(DISTINCT qdept.deptname) =
         (SELECT COUNT(DISTINCT deptname)
          FROM qdept
          WHERE deptfloor = 2);
```

6. At least some number

- ***Find the items sold by at least two departments on the second floor.***

The GROUP BY and HAVING clauses make it easy for you to count the number of rows that meet some condition.

```
SELECT itemname
FROM qsale, qdept
WHERE qsale.deptname = qdept.deptname AND
      deptfloor      = 2
GROUP BY itemname
   HAVING COUNT(DISTINCT qdept.deptname) > 1;
```

7. A friendly IN for an SQL traveler

- ***Find the salary of Clare's manager.***

Tills query is readily handled with the IN clause. The inner query gets the employee number of Clare's manager, and this locates the row containing that person's salary.

```
SELECT empfname, empsalary
FROM qemp
WHERE empno IN
     (SELECT bossno
      FROM qemp
      WHERE empfname = 'Clare');
```

8. Joining a table with itself

- ***Find numbers and names of those employees who make more than their manager.***

This query is a simple play once you realize that you can conceptually create two copies of a table and join them. This type of query is discussed in more depth in Chapter 6.

```
SELECT wrk.empno, wrk.empfname
FROM qemp wrk, qemp boss
WHERE wrk.bossno     = boss.empno AND
      boss.empsalary < wrk.empsalary;
```

9. A combination of subtract from all and a self-join

- ***Find the departments where all the employees earn less than their manager.***

  The key is to first find the departments where at least one employee earns more than or the same as the manager, and then subtract these departments from the set of all departments. Also, we exclude Management because it has no boss. What is left must be the departments where no employee earns more than the manager. This query is a combination, conceptually, of queries 4 and 8.

```
SELECT DISTINCT deptname
FROM qemp
WHERE deptname <> 'Management1 AND
      deptname NOT IN
               (SELECT wrk.deptname
                FROM qemp wrk, qemp boss
                WHERE wrk.bossno    = boss.empno AND
                      wrk.empsalary >= boss.empsalary);
```

10. Self-join with GROUP BY

- ***Count the number of direct employees of each manager.***

  Join the table with itself by matching employees and their managers and then group by manager's name with a count. Just in case two bosses have the same first name, employee number is included in the selection and grouping clauses.

```
SELECT boss.empno, boss.empfname, COUNT(*)
FROM qemp wrk, qemp boss
WHERE wrk.bossno = boss.empno
GROUP BY boss.empno, boss.empfname;
```

11. A self-join with two matching conditions

- ***Find the names of employees who are in the same department as their manager (as an employee). Report the name of the employee, the department, and the boss's name.***

  There is no reason why a join cannot have matching on more than one common column. Here we have two conditions: one for employee number and one for department name.

```
SELECT wrk.empfname, wrk.deptname, boss.einpfname
FROM qemp wrk, qemp boss
WHERE wrk.bossno   = boss.empno AND
      wrk.deptname = boss.deptname;
```

12. Averaging with GROUP BY

- ***List the departments having an average salary over $25,000.***

```
SELECT deptname, AVG(empsalary)
FROM qemp
GROUP BY deptname
   HAVING AVG(empsalary) > 25000;
```

13. Inner query GROUP BY and HAVING

- ***List the departments where the average salary of the employees of each manager is more than $25,000.***

  This query is more challenging than simple averaging. Note that the query implies that you should exclude the manager's salary from the calculation. An employee is the manager of a particular department if that person's employee number and the department name are in the same row in qdept (i.e., wrk. empno = qdept. empno AND wrk. cieptname = qdept. deptname). Once the department manager has been excluded, then use grouping to get the employee data by department.

```
SELECT wrk.deptname, AVG(wrk.empsalary)
FROM qemp wrk
WHERE wrk.empno NOT IN
            (SELECT qdept.empno
             FROM qdept
             WHERE wrk.empno    = qdept.empno AND
                   wrk.deptname = qdept.deptname)
GROUP BY wrk.deptname
   HAVING AVG(wrk.empsalary) > 25000;
```

14. An IN with GROUP BY and COUNT

- ***List the name and salary of the managers with more than two employees.***

  The inner query uses grouping with counting to identify the employee numbers of managers with more than two employees. The outer query reports details of these managers.

```
SELECT empfname, empsalary
FROM qemp
WHERE empno IN
      (SELECT bossno FROM qemp
       GROUP BY bossno
       HAVING COUNT(*) > 2);
```

15. A self-join with some conditions

- ***List the name, salary, and manager of the employees of the Marketing department who have a salary over $25,000.***

  A join gets workers and bosses together in the same row, and then the various conditions are applied to restrict the rows reported.

```
SELECT wrk.empfname, wrk.empsalary, boss.empfname
FROM qemp wrk, qemp boss
WHERE wrk.bossno    = boss.empno    AND
      wrk.deptname  = 'Marketing' AND
      wrk.empsalary > 25000;
```

16. Making comparisons

- ***List the names of the employees who earn more than any employee in the Marketing department.***

  The first step is to determine the maximum salary of anyone in the Marketing department. Then, find anyone with a larger salary.

```
SELECT empfname, empsalary
FROM qemp
WHERE empsalary >
      (SELECT MAX(empsalary)
       FROM qemp
       WHERE deptname = 'Marketing');
```

17. An IN with GROUP BY and SUM

- ***Among all the departments with total salary greater than $25,000, find the departments that sell Stetsons.***

  This is very similar to query 16. First, find the departments that satisfy the condition, and then select from that list any departments that sell Stetsons.

```
SELECT DISTINCT deptname
FROM qsale
WHERE itemname = 'Stetson' AND
      deptname IN
      (SELECT deptname
       FROM qemp
       GROUP BY deptname
         HAVING SUM(empsalary) > 25000);
```

18. A double divide!

- ***List the items delivered by every supplier that delivers all items of type N.***

  SQL programmers who tackle this query without blinking an eye are superheroes. This is a genuinely true-blue, tough query because it is two divides. There is an inner divide that determines the suppliers that deliver all items of type N. The SQL for this query is

```
SELECT *
FROM qspl
WHERE NOT EXISTS
      (SELECT * FROM qitem
      WHERE itemtype = 'N' AND
            NOT EXISTS
                  (SELECT *
                   FROM qdel
                   WHERE qdel.itemname = qitem.itemname AND
                         qdel.Splno    = qspl.splno));
```

  Then there is an outer divide that determines which of the suppliers (returned by the inner divide) provide all these items. If we call the result of the inner query qspn, the SQL for the outer query is

```
SELECT DISTINCT itemname
FROM qdel del
WHERE NOT EXISTS
      (SELECT *
       FROM qspn
       WHERE NOT EXISTS
            (SELECT *
             FROM qdel
             WHERE qdel.itemname = del.itemname AND
                   qdel.splno    = qspn.splno));
```

The complete query is

```
SELECT DISTINCT itemname
FROM qdel del
WHERE NOT EXISTS
      (SELECT *
       FROM qspl
       WHERE NOT EXISTS
             (SELECT *
              FROM qitem
              WHERE itemtype = 'N' AND
                    NOT EXISTS
                    (SELECT *
                     FROM qdel
                     WHERE qdel.itemname = qitem.itemname AND
                           qdel.splno    = qspl.splno)
             )
           AND NOT EXISTS
             (SELECT *
              FROM qdel
              WHERE qdel.itemname = del.itemname AND
                    qdel.splno    = qspl.splno)
      );
```

19. A slam dunk

- ***Find the suppliers that deliver compasses.***

A simple query to recover from the double divide. It is a good idea to include supplier number since splname could possibly be nonunique.

```
SELECT DISTINCT qspl.splno, splname
FROM qspl, qdel
WHERE qspl.splno = qdel.splno AND
      itemname   = 'Compass';
```

20. A 6-inch putt for a birdie

- ***Find the suppliers that do not deliver compasses.***

This is a relatively straightforward subtract (and take one off par for a birdie).

```
SELECT splno, splname
FROM qspl
WHERE splno NOT IN
      (SELECT splno
       FROM qdel
       WHERE itemname = 'Compass');
```

21. Making the count

- ***Find the suppliers that deliver both compasses and an item other than compasses.***

A simple approach is to find those suppliers that supply items other than compasses (i.e., itemname <> 'compass') and also supply compasses (the subquery).

```
SELECT DISTINCT qdel.splno, splname
FROM qspl, qdel
WHERE qdel.splno = qspl.splno AND
      itemname   <> 'Compass' AND
      qdel.splno IN
           (SELECT splno
            FROM qdel
            WHERE itemname = 'Compass');
```

A more general approach is to find suppliers that have delivered compasses and more than one item (i.e., COUNT (DISTINCT itemname) > 1). This means they deliver at least an item other than compasses. Note that the GROUP BY clause includes supplier number to cope with the situation where two suppliers have the same name. The DISTINCT item-name clause must be used to guard against multiple deliveries of compasses from the same supplier.

```
SELECT DISTINCT qdel.splno, splname
FROM qspl, qdel
WHERE qdel.splno = qspl.splno AND
      qdel,splno IN
                 (SELECT splno
                  FROM qdel
                  WHERE itemname = 'Compass')
GROUP BY qdel.splno, splname
   HAVING COUNT(DISTINCT itemname) > 1;
```

The more general approach enables you to solve queries, such as *Find suppliers that deliver three items other than compasses,* by changing the HAVING clause to COUNT (DISTINCT itemname > 3).

Because DISTINCT colname is not supported by MS Access, for that DBMS you must first create a view containing distinct splno, itemname pairs and then substitute the name of the view for qdel and drop the DISTINCT clause in the COUNT statement.

22. Minus and divide

- ***List the departments that have not recorded a sale for all the items of type N.***

This query has two parts, an inner divide and an outer minus. The inner query finds departments that have sold all items of type N. These are then subtracted from all departments to leave only those that have not sold all items of type N.

```
SELECT deptname
FROM qdept
WHERE deptname NOT IN
      (SELECT deptname
       FROM qdept
       WHERE NOT EXISTS
            (SELECT *
             FROM qitem
             WHERE itemtype = 'N' AND
                   NOT EXISTS
                   (SELECT *
                    FROM qsale
                    WHERE qsale.deptname  =  cjdept.deptname AND
                          qsale.itemname  =  qitem.itemname)
            )
      );
```

23. Division with copies

- ***List the departments that have at least one sale of all the items delivered to them.***

This is a variation on the divide concept. Normally with a divide, you have three tables representing a many-to-many (m:m) relationship. In this case, you only have two tables, qdel and qsale. You can still construct the query, however, by creating two copies of qdel (del1 and del 2 in this case) and then proceeding as if you had three different tables. Also, you must match on deptname so that you get the correct (deptname, itemname) pair for comparing with qsale.

```
SELECT DISTINCT deptname
FROM qdel del1
WHERE NOT EXISTS
      (SELECT *
       FROM qdel del2
       WHERE del2.deptname = del1.deptname AND
             NOT EXISTS
                  (SELECT *
                   FROM qsale
                   WHERE del2.itemname  = qsale.itemname AND
                         del1 .deptname = qsale.deptname)
      );
```

This query can also be written as shown next. Observe how NOT IN functions like NOT EXISTS. We will use this variation on divide with some of the upcoming queries.

```
SELECT DISTINCT deptname
FROM qdel del1
WHERE NOT EXISTS
      (SELECT *
       FROM qdel del2
       WHERE del2.deptname = del1.deptname AND
             itemname NOT IN
                  (SELECT itemname
                   FROM qsale
                   WHERE deptname = del1.deptname)
      );
```

24. A difficult pairing

- ***List the supplier-department pairs -where the department sells all items delivered to it by the supplier.***

This query is yet another variation on divide. An additional complication is that you have to match the department name and item name of sales and deliveries.

```
SELECT splname, deptname
FROM qdel del1, qspl
WHERE del1.splno = qspl.splno AND
      NOT EXISTS
           (SELECT *
            FROM qdel
            WHERE qdel.deptname = del1.deptname AND
                  qdel.splno    = del1.splno    AND
                  itemname NOT IN
                       (SELECT itemname
                        FROM qsale
                        WHERE qsale.deptname = del1.deptname)
           );
```

25. Two divides and an intersection

- ***List the items delivered to all departments by all suppliers.***

  This query has three parts. First, find the items delivered by all suppliers (the first divide); then find the items delivered to all departments (the second divide). Finally, find the items that satisfy both conditions—the function of the AND connection between the two divides. The items reported must be the ones both delivered by all suppliers and delivered to all departments. The administrative departments (Management, Marketing, Personnel, Accounting, and Purchasing) should be excluded because they do not sell items.

```
SELECT itemname
FROM qitem
WHERE NOT EXISTS
            (SELECT *
             FROM qspl
             WHERE NOT EXISTS
                  (SELECT *
                  FROM qdel
                  WHERE qdel. itemname = qitem. itemname AND
                        qdel.splno     = qspl.splno)
            )
      AND NOT EXISTS
            (SELECT *
             FROM qdept
             WHERE deptname NOT IN
                     ('Management', 'Marketing', 'Personnel',
                      'Accounting', 'Purchasing')
                   AND NOT EXISTS
                     (SELECT *
                     FROM qdel
                     WHERE qdel.itemname = qitem.itemname AND
                     qdel.deptname       = qdept.deptname)
            );
```

26. A divide with a matching condition

- ***List the items sold only by departments that sell all the Items delivered to them.***

Yet another variation on divide—which is why you needed a break. There are two parts to this query. First, look for items sold by departments that sell all items delivered to them, and then make sure that no other department sells the same item.

```
SELECT DISTINCT iternnaine
FROM gsale sale
WHERE deptname IN
      (SELECT deptname
       FROM qdept dept1
       WHERE NOT EXISTS
            (SELECT *
             FROM qdel
             WHERE qdel.deptname  = dept1.deptname AND
                   itemnaine NOT IN
                        (SELECT itemname
                         FROM qsale
                         WHERE qsale.deptname = dept1.deptname)
            )
      )
      AND NOT EXISTS
            (SELECT *
             FROM qsale
             WHERE itemname = sale.itemname AND
                   deptname NOT IN
                        (SELECT deptname
                         FROM qdept dept2
                         WHERE NOT EXISTS
                              (SELECT *
                               FROM qdel
                               WHERE qdel.deptname = dept2.deptname AND
                                     itemname NOT IN
                                          (SELECT itemname
                                           FROM qsale
                                           WHERE gsale.deptname =
                                                 dept2.deptname)
                              )
                        )
            );
```

27. Restricted divide

- ***Who are the suppliers that deliver all the items of type N?***

A slight variation on the standard divide to restrict consideration to the type N items.

```
SELECT splno, splname
FROM qspl
WHERE NOT EXISTS
      (SELECT *
       FROM qitem
       WHERE itemtype = 'N' AND
             NOT EXISTS
                  (SELECT *
                   FROM qdel
                   WHERE qdel.splno    = qspl.splno AND
                         gdel.itemname = qitem.itemname));
```

28. A NOT IN variation on divide

- ***List the suppliers that deliver only the items sold by the Books department.***

This query may be rewritten as, *Select suppliers where there does not exist a delivery that does not include the items sold by the Books department.* Note the use of the IN clause to limit consideration to those suppliers that have made a delivery. Otherwise, a supplier that has never delivered an item will be reported as delivering only the items sold by the Books department.

```
SELECT splname
 FROM qspl
 WHERE splno IN
            (SELECT splno
             FROM qdel)
       AND NOT EXISTS
            (SELECT *
             FROM qdel
             WHERE qdel.splno = qspl.splno AND
                   itemname NOT IN
                        (SELECT itemname
                         FROM qsale
                         WHERE deptname = 'Books'));
```

29. All and only

- ***List the suppliers that deliver all and only the items sold by the Equipment department.***

This is one query with three parts. The first part identifies suppliers that deliver all items sold by the Equipment department (they could also deliver other items, but these are not sold in the Equipment department). The second part identifies suppliers that deliver only items sold by the Equipment department (i.e., they do not deliver any other items). This part is similar to the previous query. The third part is the intersection of the first two queries to indicate suppliers that satisfy both conditions.

```
SELECT splname
FROM qspl
WHERE NOT EXISTS
            (SELECT *
             FROM qsale
             WHERE deptname = 'Equipment' AND
                   itemname NOT IN
                        (SELECT itemname
                         FROM qdel
                         WHERE qdel.splno = qspl.splno))
      AND NOT EXISTS
            (SELECT *
             FROM qdel
             WHERE qdel.splno = qspl.splno AND
                   itemname NOT IN
                        (SELECT itemname
                         FROM qsale
                         WHERE deptname = 'Equipment'));
```

30. Divide with an extra condition

- ***List the suppliers that deliver every item of type C to the same department on the second floor.***

  This is a divide in which there are three WHERE conditions in the innermost query. The extra condition handles the "same department" requirement.

```
SELECT splname
FROM qspl
WHERE EXISTS
      (SELECT *
       FROM qdept
       WHERE deptfloor = 2 AND
             NOT EXISTS
                  (SELECT *
                   FROM qitem
                   WHERE itemtype = 'C' AND
                   NOT EXISTS
                        (SELECT *
                         FROM qdel
                         WHERE qdel.splno    = qspl.splno     AND
                               qdel.itemname = qitem,itemname AND
                               qdel.deptname = qdept.deptname)));
```

31. At least some COUNT

- ***List the suppliers that deliver at least two items of type N to departments.***

  First, do a three-way join to get the data for deliveries, suppliers, and items. Then, group with a COUNT condition. This can be easily extended to a variety of conditions based on counts.

```
SELECT qspl.splno, splname
FROM qdel, qspl, qitem
WHERE itemtype      = 'N' AND
      qdel.splno    = qspl.splno AND
      qdel.itemname = qitem.itemname
GROUP BY qspl.splno, splname
   HAVING COUNT(DISTINCT qdel.itemname) > 1;
```

32. Double divide with a restriction

- ***List the suppliers that deliver all the items of type B to departments on the second floor -who sell all the items of type R.***

  Break this query into two parts. First, create a view of departments on the second floor that sell all items of type R.

```
CREATE VIEW v32 AS
      (SELECT deptname
       FROM qdept
       WHERE deptfloor = 2 AND
             NOT EXISTS
            (SELECT *
             FROM qitem
             WHERE itemtype = 'R1 AND
                   NOT EXISTS
                  (SELECT *
                   FROM qsale
                   WHERE qsale.itemname = qitem.itemname AND
                         qsale.deptname = qdept.deptname)));
```

Second, report all the suppliers that deliver all the items of type B to the departments designated in the previously created view.

```
SELECT splname
FROM qspl
WHERE NOT EXISTS
      (SELECT *
       FROM qitem
       WHERE itemtype = 'B1 AND
             NOT EXISTS
                  (SELECT *
                   FROM qdel
                   WHERE qdel.itemname = qitem.itemname AND
                        qdel.splno    = qspl.splno AND
                        deptname IN
                                    (SELECT deptname FROM v32)));
```

33. Triple divide with an intersection

- ***List the suppliers that deliver all the items of type B to the departments that also sell all the items of type N.***

  Defeat this by dividing—*that* word again — the query into parts. First, identify the departments that sell all items of type N, and save as a view.

```
CREATE VIEW v33a AS
      (SELECT deptname
       FROM qdept
       WHERE NOT EXISTS
            (SELECT *
             FROM qitem
             WHERE itemtype = 'N1 AND
                   NOT EXISTS
                        (SELECT *
                        FROM qsale
                        WHERE qsale.deptname = qdept.deptname AND
                              qsale.itemname = qitem.itemname)));
```

  Next, select the departments to which all items of type B are delivered, and save as a view.

```
CREATE VIEW v33b AS
      (SELECT deptname
       FROM qdept
       WHERE NOT EXISTS
            (SELECT *
             FROM qitem
             WHERE itemtype = 'B1 AND
                   NOT EXISTS
                        (SELECT *
                         FROM qdel
                         WHERE qdel.deptname = qdept.deptname AND
                               qdel.itemname = qitem.itemname)));
```

Now, find the suppliers that supply all items of type B to the departments that appear in both views.

```
SELECT splname
FROM qspl
WHERE NOT EXISTS
      (SELECT *
       FROM qitem
       WHERE itemtype = 'B' AND
             NOT EXISTS
                  (SELECT *
                   FROM qdel
                   WHERE qdel.splno = qspl.splno AND
                         qdel.itemname = qitem.itemname
                         AND EXISTS
                              (SELECT *
                               FROM v33a
                               WHERE qdel.deptname = v33a.deptname)
                         AND EXISTS
                              (SELECT *
                               FROM v33b
                               WHERE qdel.deptname = v33b.deptname)
                  )
      );
```

34. An easy one COUNT

- ***List the items delivered by exactly one supplier (i.e., list the items always delivered by the same supplier).***

A reasonably straightforward GROUP BY with an exact count.

```
SELECT itemname
FROM qdel
GROUP BY itermame
   HAVING COUNT(DISTINCT splno) = 1;
```

35. The only one

- ***List the supplier and the item, where the supplier is the only deliverer of some item.***

For each item delivered, check to see if there is no other delivery of this item by another supplier.

```
SELECT DISTINCT qspl.splno, splname, itemname
FROM qspl, qdel dell
WHERE qspl.splno = dell.splno AND
      itemname NOT IN
                  (SELECT itemname
                   FROM qdel
                   WHERE qdel.splno <> dell.splno);
```

36. At least some number

- ***List the suppliers that deliver at least 10 items.***

This is an easy GROUP BY with a count condition.

```
SELECT qspl.splno, splname
FROM qdel, qspl
WHERE qdel.splno = qspl.splno
GROUP BY qspl.splno, splname
   HAVING COUNT(DISTINCT qdel.itemname) >= 10;
```

37. A three-table join

- ***For each item, give its type, the departments that sell the item, and the floor location of these departments.***

A three-table join is rather easy after all the divides.

```
SELECT qitem.itemname, itemtype, qdept.deptname, deptfloor
FROM qitem, qsale, qdept
WHERE qsale.itemname = qitem.itemname AND
      qsale.deptname = qdept.deptname;
```

38. Using NOT IN like NOT EXISTS

- ***List the departments for which each item delivered to the department is delivered to some other department as well.***

  This is another variation on double negative logic. The query can be rewritten as, *Find departments where there is not a delivery where a supplier does not deliver the item to some other department.* In this situation, the NOT IN clause is like a NOT EXISTS.

```
SELECT DISTINCT deptname
FROM qdel del1
WHERE NOT EXISTS
      (SELECT *
       FROM qdel del2
       WHERE del2.deptname = del1.deptname AND
             itemname NOT IN
                       (SELECT itemname
                        FROM qdel del3
                        WHERE del3.deptname <> del1.deptname));
```

39. Minus after GROUP BY

- ***List each item delivered to at least two departments by each supplier that delivers it.***

  The inner query uses grouping to identify items delivered by the same supplier to one department at most. The remaining items must be delivered by the same supplier to more than one department.

```
SELECT DISTINCT itemname
FROM qdel
WHERE itemname NOT IN
                  (SELECT itemname
                   FROM qdel
                   GROUP BY itemname, splno
                      HAVING COUNT(DISTINCT deptname) < 2);
```

40. Something to all

- ***List the items that are delivered only by the suppliers that deliver something to all the departments.***

  This is a variation on query 25 with the additional requirement, handled by the innermost query, that the supplier delivers to all departments. Specifying that all departments get a delivery means that the number of departments to whom a supplier delivers (GROUP BY splno HAVING COUNT (DISTINCT deptname)) must equal the number of departments that sell items (SELECT COUNT(*) FROM qdept WHERE deptname NOT IN ('Management', 'Marketing', 'Personnel', 'Accounting', 'Purchasing')).

```
SELECT DISTINCT itemname
FROM qdel del1
WHERE NOT EXISTS
            (SELECT *
             FROM qdel del2
             WHERE del2.itemname = del1.itemname AND
                   splno NOT IN
                        (SELECT splno
                         FROM qdel
                         GROUP BY splno
                            HAVING COUNT(DISTINCT deptname) =
                                        (SELECT COUNT(*)
                                         FROM qdept
                                         WHERE deptname NOT IN
                                                   ('Management',
                                                    'Marketing',
                                                    'Personnel',
                                                    'Accounting',
                                                    'Purchasing')
                                        )));
```

41. Intersection (AND)

- ***List the items delivered by Nepalese Corp. and sold In the Navigation department,***

The two parts to the query—the delivery and the sale — are intersected using AND.

```
SELECT DISTINCT itemname
FROM qdel, qspl
WHERE qdel.splno = qspl.splno       AND
      splname    = 'Nepalese Corp.' AND
      itemname IN
            (SELECT itemname
             FROM qsale
             WHERE deptname = 'Navigation');
```

42. Union (OR)

- ***List the items delivered by Nepalese Corp, or sold in the Navigation department.***

The two parts are the same as query 41, but the condition is OR rather than AND.

```
SELECT DISTINCT itemname
FROM qdel, qspl
WHERE qdel.splno = qspl.splno    AND
      splname = 'Nepalese Corp.' OR
      itemname IN
            (SELECT itemname
             FROM qsale
             WHERE deptname = 'Navigation');
```

43. Intersection/union

- ***List the departments selling items of type E that are delivered by Nepalese Corp. and/or that are sold by the Navigation department.***

The inner query handles the and/or with OR. Remember OR can mean that the items are in both tables or one table. The outer query identifies the departments that receive the delivered items, which satisfy the inner query.

```
SELECT DISTINCT deptname
FROM qsale
WHERE itemname IN
(SELECT qitern.itemname
 FROM qitem, qdel, qspl
 WHERE qitem.itemname = qdel.itemname AND
       qdel.splno     = qspl.splno    AND
       splname = 'Nepalese Corp.'     AND
       itemtype = 'E')                OR
       itemname IN
                 (SELECT itemname
                  FROM qsale
                  WHERE deptname = 'Navigation');
```

44. Averaging with a condition

- ***Find the average salary of the employees in the Clothes department.***

This is very easy, especially after conquering the divides.

```
SELECT AVG(empsalary)
FROM qemp
WHERE deptname = 'Clothes';
```

45. Averaging with grouping

- ***Find, for each department, the average salary of the employees.***

Another straightforward averaging query.

```
SELECT deptname, AVG(empsalary)
FROM qemp
GROUP BY deptname;
```

46. Average with a join, condition, and grouping

- ***Find, for each department on the second floor, the average salary of the employees.***

A combination of several averaging queries.

```
SELECT qdept.deptname, AVG(empsalary)
FROM qemp, qdept
WHERE qemp.deptname = qdept.deptname AND
      deptfloor     = 2
GROUP BY qdept.deptname;
```

47. Averaging with multiple joins

- ***Find, for each department that sells items of type E, the average salary of the employees.***

Four joins, a condition, and grouping—this is not a particularly challenging query. The multiple joins are needed to get the data required for the answer into a single row.

```
SELECT qdept.deptname, AVG(empsalary)
FROM gerap, gdept, qsale, qitem
WHERE qemp.deptname = qfiept.deptname AND
      qdept.deptname = qsale.deptname AND
      gsale.itemname = qitem.itemname AND
      itemtype = 'E'
GROUP BY qdept.deptname;
```

48. Complex counting

- ***What are the number of different items delivered by each supplier that delivers to all departments?***

First, determine the suppliers that deliver to each department, excluding administrative departments. The inner query handles this part of the main query. Second, count the number of different items delivered by the suppliers identified by the inner query,

```
SELECT splname, COUNT(DISTINCT itemname)
FROM qdel del1, gspl
WHERE del1.splno = qspl.splno AND
      NOT EXISTS
            (SELECT *
             FROM gdept
             WHERE deptname NOT IN
                              (SELECT deptname
                               FROM qdel
                               WHERE gdel.splno = del1.splno)
                   AND deptname NOT IN
                                   ('Management', 'Marketing',
                                    'Personnel', 'Accounting',
                                    'Purchasing')
            )
GROUP BY splname;
```

49. Summing with joins and conditions

- ***Find the total number of items of type E sold by the departments on the second floor.***

Summing is very similar to averaging (see query 47).

```
SELECT SUM(saleqty)
FROM qitem, qsale, qdept
WHERE qitem.itemname = gsale.itemname AND
      qdept.deptname = gsale.deptname AND
      itemtype = 'E'                  AND
      deptfloor = 2;
```

50. Summing with joins, conditions, and grouping

- ***Find, for each item, the total quantity sold by the departments on the second floor.***

Conceptually, this query is similar to query 47.

```
SELECT itemname, SUM(saleqty)
FROM qsale, qdept
WHERE qsale.deptname = qdept.deptname AND
      deptfloor      = 2
GROUP BY itemname;
```

51. Advanced summing

- ***List suppliers that deliver a total quantity of items of types C and N that is altogether greater than 100.***

The difficult part of this query, and it is not too difficult, is to write the condition for selecting items of type C and N. Notice that the query says C and N, but don't translate this to (itemtype = 'C' AND itemtype = 'N') because an item cannot be both types simultaneously. The query means that for any delivery, the item should be type C or type N.

```
SELECT qdel.splno, splname
FROM qspl, qdel, qitem
WHERE qspl.splno     = qdel.splno    AND
      qitem.itemname = qdel.itemname AND
     (itemtype = 'C' OR itemtype = 'N')
GROUP BY qdel.splno, splname
   HAVING SUM(delqty) > 100;
```

52. Comparing to the average with a join

- ***List the employees in the Accounting department and the difference between their salaries and the average salary of the department.***

The key to solving this query is placing the average salary for Accounting employees in the same row as the department salary data. This is a two-stage process. You first need to determine the average salary of the employees in all departments and save this as a view. Then, join this view to the qemp table matching the Accounting department's name. Once the average departmental salary has been concatenated to each row, the query is straightforward.

```
CREATE VIEW v52(deptname, dpavgsal) AS
      SELECT deptname, AVG(empsalary)
      FROM qemp GROUP BY deptname;

      SELECT empfname, (empsalary - dpavgsal)
      FROM v52, qemp
      WHERE v52.deptname  = qemp.deptname AND
            qemp.deptname = 'Accounting';
```

53. Comparing to the average with a product

- ***List the employees in the Accounting department and the difference between their salaries and the average salary of all the departments.***

This is a slight variation on the previous query except that a join is not used to combine the data from the view with the employee table. The view is a single row and column table containing the average salary for the organization. To concatenate this row with the data for employees in the Accounting department, we use a product instead of a join. Remember, a product is specified by simply listing the names of the two tables.

```
CREATE VIEW v53(allavgsal) AS
      SELECT AVG(empsalary) FROM qemp;

SELECT empfname, (empsalary - allavgsal)
FROM v53, qemp
WHERE deptname = 'Accounting';
```

54. Averaging with multiple grouping

- ***What is, for each supplier, the average number of items per department that the supplier delivers?***

Here, the averaging is broken into two levels: department within supplier.

```
SELECT qdel.splno, splname, deptname, AVG(delqty)
FROM qspl, qdel
WHERE qspl.splno = qdel.splno
GROUP BY qdel.splno, splname, deptname;
```

55. More than the average with grouping

- ***For each department, find the average salary of the employees who earn more than the average salary of the department.***

The inner query determines the average salary of each department. Look carefully at how it handles matching departments.

```
SELECT deptname, AVG(empsalary)
FROM qemp out
WHERE empsalary >
            (SELECT AVG(empsalary)
             FROM qemp inn
             WHERE out.deptname = inn.deptname)
GROUP BY deptname;
```

56. The simplest average

- ***Give the overall average of the salaries in all departments.***

This is a very simple query.

```
SELECT AVG(empsalary) FROM qemp;
```

Another possible interpretation is that you have to find the total average salary after you determine the average salary for each department. To do this, you would first create a view containing the average salary for each department (see query 52) and then find the average of these average salaries.

```
SELECT AVG(dpavgsal) FROM v52;
```

57. Difference from the average

- ***List each employee's salary, the average salary within that person's department, and the difference between the employees' salaries and the average salary of the department.***

This is reasonably easy once you have created a view of departmental average salaries.

```
SELECT empfname, empsalary, dpavgsal, (empsalary - dpavgsal)
FROM v52, qemp
WHERE v52.deptname = qemp.deptname;
```

58. Averaging with multiple joins, multiple grouping, and a condition

- ***What is the average delivery quantity of items of type N delivered by each company who delivers them?***

This is similar to query 54.

```
SELECT qdel.splno, splname, qdel.itemname, AVG(delqty)
FROM qdel, qspl, qitem
WHERE qdel.splno    = qspl.splno AND
      qdel.itemname = qitem.itemname AND
      iterntype     = 'N'
GROUP BY qdel.splno, splname, qdel.itemname;
```

59. Detailed averaging

- ***What is the average delivery quantity of items of type N delivered by each supplier to each department (given that the supplier delivers items of type N to the department)?***

  Now we take averaging to three levels — supplier, department, item. You can take averaging to as many levels as you like.

```
SELECT qdel.splno, splname, deptname, qdel.itemname, AVG(delqty)
FROM qdel, qspl, qitem
WHERE qdel.splno    = qspl.splno AND
      qdel.itemname = qitem.itemname AND
      itemtype      = 'N'
GROUP BY qdel.splno, splname, deptname, qdel.itemname;
```

60, Counting pairs

- ***What is the number of supplier-department pairs in which the supplier delivers at least one item of type E to the department?***

  First, find all the supplier-department pairs. Without DISTINCT, you would get duplicates, which would make the subsequent count wrong.

```
CREATE VIEW v60 AS
      (SELECT DISTINCT splno, deptname
       FROM qdel, qitem
       WHERE qdel.itemname = qitem.itemname AND
             itemtype      = 'E');
```

  Now, it is a simple count.

```
SELECT COUNT(*) FROM v60;
```

61. No Booleans

- ***Is it true that all the departments that sell Items of type C are located on the third floor? (The result can be a Boolean 1 or 0, meaning yes or no.)***

  SQL cannot return *true or false;* it always returns a table. But you can get close to Boolean results by using counts. If we get a count of zero for the following query, there are no departments that are not on the third floor that sell items of type C.

```
SELECT COUNT(*)
FROM qdept
WHERE deptfloor <> 3 AND
      EXISTS
            (SELECT *
             FROM qsale, qitem
             WHERE qsale.itemname = qitem.itemname AND
                   qsale.deptname = qdept.deptname AND
                   itemtype       = 'C');
```

  Then you need to check that departments on the third floor sell items of type C. If the second query returns a nonzero value, then it is true that departments that sell items of type C are located on the third floor.