September, 1998

Advisor Answers

Visual FoxPro 6.0, 5.0 and 3.0

Visual FoxPro 6.0, 5.0 and 3.0 and FoxPro 2.x

Q: I have a table containing purchases (Operacion="C") and sales (Operacion="V") of a pharmaceutical product. I need to total the amount sold and purchased by date, so I wrote a SQL SELECT with a UNION clause.

In my sample data, there are only two purchases, with a different date for each, so I should get two records for Operacion="C". Instead of that, I get only one record for Operacion="C".

Could you explain what's going on?

–Germán Turriziani (via Advisor.com)

A: When I tested the query you sent with your sample data, I got different results than what you report. One possible explanation for the difference in our results is the setting of SET COLLATE. For more on the problems associated with collation sequences, see the Ask Advisor column from February '98.

However, what I saw also wasn't quite right. The secret to getting the kind of results you're looking for is in understanding the relationships among several components of the query.

Your table contains two kinds of records (sales and purchases), with a code differentiating them. As I understand it, the results you want have two records for each date, one containing total amount sold and the other containing total amount purchased. (The quantities are in milligrams.)

You're correct that UNION is the way to do this. You need to combine the records containing the total sales by date with the records containing the total purchases by date.

To get the total sales by date, you'd use something like this. (Although your sample data has Spanish field and table names, I'm using English, to make it easier for more readers to follow.)

```
SELECT ddate, SUM(quantity) ;
   FROM transactions ;
   WHERE operation="V" ;
   GROUP BY ddate
```

You can, of course, add an expression to the WHERE clause to limit the query to a particular date range.

A similar query gives us total amount sold by date:

```
SELECT ddate, SUM(quantity) ;
```

```
   FROM transactions ;
   WHERE operation="C" ;
   GROUP BY ddate
```

Again, we can filter on date to limit the results.

In order to combine the two, we use UNION, which takes the results of multiple queries and creates a single result set. One of the key points here is that GROUP BY is executed before the UNION clause, so each query in the UNION needs its own GROUP BY clause.

When we combine the two results, we need something to indicate which records came from which result set. Here's the query so far:

```
SELECT ddate, SUM(quantity) AS Amount, "V" AS TransType ;
   FROM transactions ;
   WHERE operation="V" ;
   GROUP BY ddate ;
UNION ;
SELECT ddate, SUM(quantity) AS Amount, "C" AS TransType ;
   FROM transactions ;
   WHERE operation="C" ;
   GROUP BY ddate
```

The next issue is ordering the result. Unlike GROUP BY, ORDER BY is executed only once in a UNIONed query, so we can add that clause at the end. We also, presumably, want to put the results somewhere. I generally use a cursor. So here's the query I'd write:

```
SELECT ddate, SUM(quantity) AS Amount, "V" AS TransType ;
   FROM transactions ;
   WHERE operation="V" ;
   GROUP BY ddate ;
UNION ;
SELECT ddate, SUM(quantity) AS Amount, "C" AS TransType ;
   FROM transactions ;
   WHERE operation="C" ;
   GROUP BY ddate ;
ORDER BY dDate ;
INTO CURSOR Trans
```

Add whatever additional filtering you need in each of the WHERE clauses and you should be set.

–Tamar