

Not Your Standard Sorting Requirement

Sometimes the requirements of a particular application dictate that data needs to be sorted using some irregular collating sequence. These odd needs sometimes cause developers to sit and scratch their heads for hours, searching for ways to make DB2 do something that seems to be "unnatural." But often you can create an answer just by understanding the problem and applying some creative SQL.

At this point, some of you might be asking "What the heck is he talking about?"• Fair enough. Let's take a look at an example to bring the issue into focus.

Assume that you have a table containing transactions, or some other type of interesting facts. The table has a CHAR(3) column containing an abbreviation for the name of the day on which the transaction happened; let's call this column DAY_NAME. So, for example, the DAY_NAME column would contain MON for Monday data, and so on.

Now, let's further assume that we want to write queries against this table that orders the results by DAY_NAME. We'd want Sunday first, followed by Monday, Tuesday, Wednesday, and so on. How can this be done?

Well, the first step is usually to write the first query that comes to mind, or something like this:

```
SELECT DAY_NAME, COL1, COL2 . . .
FROM TXN_TABLE
ORDER BY DAY_NAME;
```

Of course, the results will be sorted improperly. ORDER BY will sort the results alphabetically; in other words: FRI MON SAT SUN THU TUE WED

This is what I mean by an irregular sorting requirement. • Here we have an example that occurs commonly enough, but without an obvious immediate solution. Furthermore, many businesses and applications have similar requirements for which the business needs dictate a different sort order than strictly alphabetical or numeric. So what is the solution here?

Of course, one solution would be to design the table with an additional numeric or alphabetic column that would sort properly. By this, I mean that we could add a DAY_NUM column that would be 1 for Sunday, 2 for Monday, and so on. But this requires a database design change, and it becomes possible for the DAY_NUM and DAY_NAME to get out of sync if you are not careful.

There is another solution that is both elegant and does not require any change to the database. To implement this solution, all you need is an understanding of SQL and SQL functions -- in this case, the LOCATE function. Consider this SQL:

```
SELECT DAY_NAME, COL1, COL2 . . .
FROM TXN_TABLE
ORDER BY LOCATE('SUNMONTUEWEDTHUFRISAT', DAY_NAME);
```

The trick here is to understand how the LOCATE function works. The LOCATE function returns the starting position of the first occurrence of one string within another string.

So, in our example, LOCATE finds the position of the DAY_NAME value within the string 'SUNMONTUEWEDTHUFRISAT', and returns the integer value of that position. So, if DAY_NAME is WED, the LOCATE function returns 10. (Note: Some other database

systems have a similar function called INSTR.) Sunday would return 1, Monday 4, Tuesday 7, Wednesday 10, Thursday 13, Friday 16, and Saturday 19. This means that our results would be in the order we require.

Of course, you can go one step further if you'd like. Some queries may need to actually return the day of week. You can use the same technique with a twist to return the day of week value, given only the day's name. To turn this into the appropriate day of the week number (that is, a value of 1 through 7), we divide by three, use the INT function on the result to return only the integer portion of the result, and then add one:

```
INT (LOCATE ('SUNMONTUEWEDTHUFRISAT', DAY_NAME) / 3) + 1;
```

Let's use our previous example of Wednesday again. The LOCATE function returns the value 10. So, $INT(10/3) = 3$ and add 1 to get 4. And sure enough, Wednesday is the fourth day of the week.

Summary

With a sound understanding of the features of DB2 SQL and a little imagination many irregular requirements are achievable using nothing more than SQL!.

About the Author

Craig S. Mullins is the president and principal consultant of Mullins Consulting, Inc., specializing in DB2 and database performance, strategy and research. He is also the publisher and editor for [The Database Site](#). Craig has more than two decades of experience in all facets of database systems development and has worked with DB2 for z/OS since Version 1. Craig is the author of the book [DB2 Developer's Guide](#), currently in its 5th edition with over 1500 pages of in-depth technical information on DB2 for z/OS. He is also the author of [Database Administration: The Complete Guide to Practices and Procedures](#), the industry's only comprehensive guide to heterogeneous database administration. And Craig was appointed by IBM as an [Information Champion](#). You can reach Craig thru his web site at <http://www.CraigSMullins.com>.