Craig S. Mullins

October / November 2004

The Resource for Users of IBM zSeries & S/390 Systems

z/JOURNAL

zData Perspectives

by Craig S. Mullins

The Top Ten Features of DB2 for z/OS V8

Every new release of DB2 is burgeoning with new features and functionality that will be of use to someone. And everyone has their own opinion as to which features are the most important. I'm using this installment of my column to comment on the top ten new features of the latest version of DB2 that will be most useful to the most shops. So, as is standard with such lists, let's start with number 10 and work our way up to number 1…

10. Online Schema Change: It has been a long-standing desire of DBAs to be able to change DB2 database schemas using simple ALTERs instead of having to drop and re-create

DDL. Well, V8's online schema change capability is IBM's first step in making that dream a reality. But it ranks so low on this Top 10 list because the job is not complete and there are underlying issues with what had been delivered. The net result is that some types of changes are now easier, but application performance can suffer after a change when DB2 converts between the old format and the new format.

9. Recursive SQL: Recursion is implemented in DB2 using common table expressions (CTEs), which also are new to DB2 V8. A CTE can be thought of as a named temporary table within a SQL statement that is retained for the duration of that statement. A CTE is defined at the beginning of a query using the WITH clause. After it is defined, the CTE can be referenced by name within the rest of the SQL statement. Although it may be a bit complicated to learn, recursive SQL adds great power to our DB2 querying capabilities. If you have a business need to walk or explode hierarchies in DB2, recursive SQL helps immensely.

8. Dynamic Scrollable Cursors: Although scrollable cursors were introduced in V7, V8 boosts their value by making them more usable with better performance. Dynamic scrollable cursors enable direct access to the data in the base table instead of having to use a declared temporary table – and they do not materialize. Each FETCH returns the most current data and is sensitive to all INSERTs, UPDATEs, and DELETEs – and that means no more delete holes and missing updates.

7. Multi-row FETCH and INSERT: With this great new feature you can use arrays as a target for your FETCH or to bulk INSERT data. By accessing data in sets, instead of row by row, you can greatly speed up application performance when large numbers of rows need to be processed.

6. Materialized Query Tables: A materialized query table, or MQT, is basically a view whose data is materialized and saved. So, you can turn a view into a table using MQTs, providing potential benefit for your complex business intelligence queries performing summaries and analytics. Of course, there are management issues involved, but you can use MQTs to create “denormalized” structures to improve query performance – and leave those base tables fully normalized.

5. Sequence Objects: Sequence objects improve upon DB2’s ability to automatically assign sequential values to a column. Like identity columns, sequence objects control the starting point for generating values and how to increment; but sequence objects are more flexible and generally useful than identity columns. Sequence objects are not forced onto a single table, can be used by multiple data types, and are easier to control during program testing.

4. DPSIs: Data-partitioned secondary indexes, or DPSIs, help to alleviate the management problems of NPIs. Because the index partitioning scheme aligns with the data partitioning scheme, even when the key is not the same, index partitions can be managed independently by utilities. Of course, there are performance issues that need to be addressed in your applications if you switch from NPIs to DPSIs.

3. Stage 1 for Unlike Data Types: In past versions of DB2, whenever the data type and length of the two arguments of a predicate did not match, DB2 would degrade the predicate to Stage 2. Now, for like data types (numeric to numeric, or character to character, for example), DB2 will not degrade predicates to Stage 2 due to non-matching data type and length. All you need to do is rebind!

2. Partitioning changes: V8 introduced many useful changed to partitioning. Collectively, I group these together as #2. Changes include the ability to define up to 4,096 partitions to a single table space, to define the partitioning limit keys in the table instead of in an index, and the ability to separate clustering from partitioning. These modifications greatly improve our ability to define, use, and manage partitioned table spaces in our applications.

1. 2M SQL Limit: And finally, at #1, we have the ability to write very long and complex SQL statements, without getting the nebulous –101 return code, which provides the very useful explanation "SQL statement too complex." By changing the size of the largest SQL statement from 32,765 bytes to 2,097,152 bytes, V8 makes it possible to write much more complex – and therefore much more useful – DB2 SQL statements. And that is definitely a #1 priority!

Summary

Feel free to disagree with me, but I think these are the ten features of V8 that will have the biggest, immediate impact on DB2 systems and applications.

From zJournal, October / November 2004
.

Home.