You Mean I Can Have My Cake and Eat It Too? Transparent Archiving with DB2 Version 11

Dawn Kastelic, Sr. Enterprise Architect 2018

Many organizations need to save data for compliance and regulatory changes. With ever-increasing requirements to keep historical data, querying and managing data is becoming more challenging. Performance becomes degraded over time as operational tables grow in size and maintenance also becomes more expensive. There is value in separating data that is still subject to change or accessed very frequently from the historical data that is rarely accessed.

There are many ways to address archiving data, such as application logic, database triggers, database partitioning, log-based technology, system-period temporal tables, and third-party tools. Most of these methods are complex and expensive. A newer, simpler way to save historical data is to take advantage of the new row-based transparent archiving feature of Version 11 DB2 for z/OS.

How does this transparent archiving work in DB2 11? DB2 provides basic archive and retrieval functions using SQL through a two-table approach, with a single image to the end user. The table that contains the current operational data is the archive-enabled table. The archive-enabled table that holds the pre-existing rows is called the archive table. DB2 can automatically move deleted rows from an archive-enabled table to the associated archive table.

What makes this so transparent? You don't need to make any changes to existing applications to include or exclude archive table data. You can include the scope of SQL queries with a global variable. This eliminates the complexity of applications having to write two distinct SQL statements to process both active and archive data. To provide optimal performance, DB2 prepares two paths for an SQL statement that access an archive-enabled table, and then determines which one to use at run time. The user can also control whether deleted data is moved to the archive table with a global variable.

Transparent archiving is also easy to manage for the database administrator. Changes made to the archive-enabled table are automatically made to the archive table. Authorizations are required to be done only on the archive-enabled table (no separate authorizations are needed on the archive table). The user can also choose to move the archive tables to a lower-cost storage device to reduce operating costs if the historical data is accessed infrequently.

So do you have to keep all your historical data and occasionally access some of it as well? Try DB2's transparent archiving feature!