IBM DATABASE MAGAZINE » Powering business information

Quarter 4, 2008 Vol. 13, Issue 4

http://www.dbmag.intelligententerprise.com/story/showArticle.jhtml?articleID=211300322

IDUG User Views: DB2 XML and LOB Design

By David Beulke

Guidelines for DB2 for z/OS applications.

Application developers are starting to leverage the ability to put all types of large XML and unstructured data objects alongside main transaction data in DB2. This marriage of data and large objects (LOBs) has increased the flexibility and functionality of new applications.

Of course, XML and other LOBs require additional performance considerations. When designing and tuning LOB applications, keep the following factors in mind.

LOB reference. The most important factor when working with LOBs is how the LOB is referenced when it materializes in DB2 or in the user address space. This materialization is determined by the LOB design and reference method you choose.

Using a LOB file reference variable provides the best performance. This approach prevents LOB materialization within the DBM1 address space and the application while avoiding restrictions on memory capacity or allocation limits within the application itself.

LOB logging. LOBs are notoriously big; logging them can cause performance issues with your active and archive logs. DB2 9 provides the ability to log LOBs greater than 1GB; the bigger the LOB, the greater the potential performance impact. Whenever possible, adjust your LOB table space definition to `NOT LOGGED` to avoid the tremendous amount of data that would arise from logging LOB update activity. Also, if you can, use utilities with parameters that turn off logging when loading LOB data types.

LOB size. Because many new Java and distributed applications are using LOBs, DB2 9 and the JDBC Universal Driver Type 4 (JCC T4) now optimize the DRDA LOB data flow. The optimization, called progressive streaming, handles LOBs differently based on the LOB size and JCC T4 and DRDA settings. Progressive streaming optimizes the processing of the smaller LOBs especially well; small LOBs (less than 32KB) are transferred with the query block, much like a long varchar column.

For medium LOBs (larger than 32KB but smaller than 1MB), DB2 9 performs better with non-LOB data being transferred in the query block and the LOB transferred in external overflow data blocks. This approach gives the client all the query data and gives the server the LOB data for processing.

You can adjust small and medium LOB handling through the `MEDDTASZ` DRDA parameter setting in conjunction with JCC T4 properties. The new DB2 Universal JCC T4 Driver has two important property settings. The first parameter is `progressiveStreaming`, which provides a new handling option when enabled. The other, `streamBufferSize`, adjusts the LOB buffer staging area handling. These property parameters cause the JCC `fullyMaterializeLobData` setting to be ignored when `progressiveStreaming` is enabled. The two parameters can improve overall LOB performance by enabling you to tune the LOB materialization and handling for your DRDA application workflows.

For large LOBs, the query block is returned along with the LOB locator variables. Best performance still comes from using locators for large LOBs to avoid LOB materialization within the application workflow.

There's no single best way to manage LOBs, but handling LOB data types correctly can improve the performance of your system, database, and application.

David Beulke has more than 22 years experience in architecture, design, and development of high performance DB2 systems across all platforms.

Return to Article