IBM DATABASE MAGAZINE » Powering business information

Quarter 2, 2007 Vol. 12, Issue 2

http://www.dbmag.intelligententerprise.com/iduguserview/showArticle.jhtml?articleID=201200471

IDUG User View: Locking Up DB2 Performance

By David Beulke

A new locking feature in DB2 9 for z/OS improves the performance outlook.

Supporting multiple systems and dealing with application developers and managers who often don't have a clue makes the DBA's job a challenge. Fortunately, DB2 9 for z/OS offers a new optimistic locking feature that can improve system performance-and perhaps the DBA's (and application developer's) mood.

Database locking is a necessary overhead and a core component of all DBMSs. Locking maintains integrity by preventing multiple transactions from changing the same data at the same time. But taking and maintaining database locks can be expensive, especially for complex systems, applications, or transactions.

Optimistic locking now uses new features defined within DB2 tables to reduce deadlocks and overall locking overhead, improving system and application performance. (Read Programmers Only for more on optimistic locking.)

To use the new feature for optimistic locking, you need to define a new `ROW CHANGE TIMESTAMP` column in a DB2 table with new parameters (`GENERATED ALWAYS, FOR EACH ROW ON UPDATE, AS ROW CHANGE TIMESTAMP`) as follows:

```
CREATE TABLE BEULKE.PRODUCT_TBL (

PROD_NBR INTEGER NOT NULL,

PROD_INVENTORY INTEGER NOT NULL,
```

```
PROD_LAST_UPD NOT NULL

GENERATED ALWAYS

FOR EACH ROW ON UPDATE

AS ROW CHANGE

TIMESTAMP);
```

These parameters tell DB2 to always populate and pay special attention to the timestamp and table. The last update timestamp has been embedded in some applications for years; now IBM has endorsed and improved on this technique.

These new features enable DB2 to retrieve rows from a specific time period and understand when they were last modified. DB2 not only notes the row timestamp information but also the record ID (RID) and change token information. Noting the row attributes allows applications and users to query the database via timestamp to get a specific row or group of rows based on `WHERE` timestamp clause criteria.

The new column feature reduces locking overhead by allowing the majority of applications to be rebound and reduces locking profiles from Repeatable Read (RR), Read Stability (RS), or Cursor Stability (CS) to Uncommitted Read (UR). Uncommitted Read avoids database locks; the application can maintain database transaction integrity by using the new timestamp column within the application `UPDATE SQL` statements. The new timestamp column provides both the timestamp and the record ID (RID) of the row that DB2 can use to verify that no other application has changed the data in question.

Another DB2 9 SQL phrase, `SKIP LOCKED DATA`, also helps avoid locks by not retrieving or affecting data rows with incompatible locks. You can use the phrase within `SELECT, UPDATE`, and `DELETE SQL` statements to avoid deadlocks.

Use both the isolation level UR and the `SKIP LOCKED DATA` phrase cautiously. Although the techniques can dramatically reduce locking and improve performance, they require that you thoroughly know your applications. Research each application in detail before using this performance boost, and read the DB2 manuals for full details. Because these techniques can significantly reduce deadlocks and locking overhead, especially in a data sharing environment, they're worth the research and implementation time.

These techniques and other performance topics are covered at International DB2 Users Group conferences around the world (www.idug.org).

David Beulke *[davebeulke@cs.com], a past president of IDUG, is a DB2 consultant, author, and lecturer who specializes in database performance, data warehouses, and Internet applications.*

Return to Article