DB2 update

January 1995

Lock Avoidance in DB2 V3

By Craig S. Mullins

Lock avoidance is a mechanism used by DB2 V3 to access data without taking a lock. However, the lock avoidance mechanism preserves data integrity. Prior DB2 versions and releases always locked data pages whenever data on the page was accessed. DB2 V3 incorporates intelligent data access logic that can determine when locks can be avoided. It is important to note that DB2 V3 can avoid taking locks only under certain circumstances that will be explained in this article.

Lock avoidance is a desirable capability because it increases the availability of DB2 data. Pages that were locked and therefore, unavailable for access, in prior releases, can be accessed by DB2 V3. Furthermore, fewer instructions are needed to access data when a lock can be avoided, because the instructions required to lock a data page are not expended. When a latch is taken instead of a lock, it is handled by the DB2 code, so the cross-memory service calls to the IRLM are eliminated. For these reasons,

avoiding locks should have a positive impact on query performance.

Which Locks Can be Avoided?

Not all locks are avoided under DB2 V3. Locking, now typically referred to as transaction locking in the IBM manuals, is still required to enforce semantics. Additionally, not all types of pages are candidates for lock avoidance. Only data pages in user tables are potential beneficiaries of the lock avoidance technique. Lock avoidance can not be employed when accessing index pages or data pages in the DB2 Catalog and DB2 Directory.

When will DB2 V3 utilize lock avoidance? Simply stated, if DB2 can determine that the data to be accessed is committed and that no semantics will be violated by failing to acquire the lock, then lock avoidance will be used.

But perhaps this is not as clear as it can be. Let's be more specific. Locks will always be taken for all pages accessed by any application bound specifying ISOLATION(RR). The repeatable read isolation level requires that data will not change over the duration of a commit scope, regardless of the number of times it is accessed. Failure to take a lock in this situation would cause a violation of data semantics.

Locks can be avoided for pages accessed by cursors having the FOR FETCH ONLY parameter that were bound specifying ISOLATION(CS) and CURRENTDATA NO. Locks can be avoided for unqualified rows accessed by queries bound with ISOLATION(CS).

Additionally, when system-managed referential integrity is used, locks can be avoided when checking for dependent rows when:

- either the parent primary key is being updated, or;

- the parent row is being deleted and the ON DELETE RESTRICT rule is specified.

Additionally, locking is always avoided when SHRLEVEL(CHANGE) is specified for the COPY and RUNSTATS utilities.

How Are Locks Avoided?

Because lock avoidance, by its very nature, avoids locks, an alternate method of determining whether data is committed or not must be employed by DB2. To do this two new features are used by DB2 V3: CLSN and the PUNC bit.

CLSN is the Commit Log Sequence Number. It is the log RBA for uncommitted data for the oldest active unit of recovery. DB2 can compare the CLSN with the log RBA in the page header to determine if it has been committed.

The PUNC bit is stored in the record header for each row of a DB2 table. PUNC stands for Possibly UNCommitted. DB2 periodically resets PUNC bits when it has documentation that the page on which the record is stored has been committed.

But, how can DB2 utilize the CLSN and PUNC bits to determine whether locks can be avoided?

For a page being accessed, the RBA of the last page update (that is, PGLOGRBA in the page header) is compared with the CLSN. If PGLOGRBA is less than CLSN, DB2 knows that the data on that page has been committed. Therefore, can avoid taking a lock and no further checking is required.

If CLSN is less than PGLOGRBA, then the page is examined to determine if any of its rows qualify (based upon the WHERE clause of the SQL statement being processed). If no rows qualify, a lock is not needed.

If rows qualify, the PUNC bit for the row being accessed is checked to see if it is off. This indicates that the row has not been updated since the last time the bit was turned off. In this can a lock can be avoided because the data is committed.

If all of these tests fail, DB2 can not avoid taking a lock.

Lock Avoidance and Performance

To determine the impact of lock avoidance on your system, DB2 V3 provides additional trace records. IFCIDs 218 and 223 provide CLSN information, and IFCIDs 226 and 227 provide “wait for page latch” information.

Synopsis

Lock avoidance is a new technique that can potentially enhance data availability and query performance. To encourage its usage in application programs, specify ISOLATION(CS) and CURRENTDATA NO when binding packages and plans. Additionally, specifically code the FOR FETCH ONLY clause to avoid ambiguous cursors.

From DB2 Update, January 1995.

© 2004 Craig S. Mullins, All rights reserved.

Home.