Take a Plunge Into SQL-Transactions

By Christopher Hardekopf

Transactions in SQL are a useful tool for preserving data integrity. With BASIS' ESQL tables, BBj® developers are now able to exploit that tool without the need to use a third party database. To use transactions appropriately, it is necessary to understand the important concepts in this article regarding their benefits and limitations.

Transactions Reviewed

Simply stated, transactions are a mechanism for grouping database operations into a single unit. This might be as straightforward as a single update or as complicated as several selects, inserts, and updates on different ESQL tables. This group of operations can then be either committed or rolled back. Committing a transaction makes every operation persistent, while rolling the transaction back will reverse every operation. Either will unlock all resources locked by the transaction. If the server machine crashes while a transaction is still in progress, the transaction operations automatically roll back, leaving the ESQL tables in a known good state. This means that either all of the operations in a transaction will persist in the database or none of the operations will take effect. The developer never has to worry that only part of a list of operations will persist in the database.

In addition, developers can create savepoints – intermediate stages of a transaction to which they can roll back – leaving the transaction as a whole open rather than rolling back the entire transaction. Later, they can choose to either commit or roll back all of the other operations in the transaction. Transactions are in auto-commit mode unless the developer chooses to use transactions explicitly. This simply means that each individual SQL statement runs inside its own transaction, generally transparent, that is automatically committed when the statement completes.

Things to Keep in Mind

Using transactions requires ESQL tables. BBj transactions need substantial file support to work properly so they do not work with traditional BBx® files. BBj transactions are table-oriented and do not handle SQL meta operations. Developers cannot roll back operations such as create database, drop database, create table, and drop table in a transaction.

Because of the data integrity guarantees provided by transactions, they must lock various database resources as the data is read and changed. After all, two transactions cannot change the same data since the client might choose to roll back either transaction. This would lead to a lot of confusion and unpredictable behavior. As a result, individual transactions should only be maintained for short periods of time. In particular, never leave a transaction open while an application waits for user input. While this is possible, it leads to slowness as it blocks several users waiting on another user. It is also valuable for clients to use read-only database connections when they do not need to write data to the database. Transaction locking for a read-only connection is much less restrictive than locking in a read-write connection. Read-only database connections allow clients to share database resources much more efficiently.

Transaction Isolation

The primary configuration for individual transactions is the transaction isolation level, which controls accessibility of the data that other transactions are concurrently modifying. In other words, transactions view data based on their own isolation levels rather than the isolation levels of other transactions. The Java API provides five different transaction isolation levels, each of which has different blocking behavior in accordance with the transactions' visibility. In general, the more restrictive the reads, the more blocking is required. The transaction levels range from TRANSACTION_NONE, which indicates that transactions are not used or are unsupported, to TRANSACTION_SERIALIZABLE, which is the strictest.

Summary

If you have not dipped your toes into ESQL files, now is definitely the time to dive in, head first. With the latest addition of transactional support, ESQL files further the concept of data availability and reliability. Because the JDBC API and many third party tools support SQL transactions, you can take the big plunge and start working with transactions today!

For more about ESQL tables and isolation levels, refer to the online documentation at www.basis.com/onlinedocs/documentation/flash and search the index for:

ESQL Overview
Connecting Using the JDBC Driver

Read the *BASIS International Advantage* article
Implementing Journaled Files and Transaction Tracking
www.basis.com/advantage/mag-v6n1/journaled.html

Chris Hardekopf
Software Engineer