

Re: Confusion about database updates

Source: <http://coding.derkeiler.com/Archive/Java/comp.lang.java.databases/2004-04/0115.html>

From: David Harper (*devnull_at_obliquity.u-net.com*)

Date: 04/12/04

Date: Mon, 12 Apr 2004 15:03:27 +0100

Burhan Khalid wrote:

> *Greetings:*

>

> *I'm developing an application that will be running on many different
> clients, all connecting to the same database server (MySQL).*

That sounds good. MySQL can easily handle many simultaneous clients. I'm currently running speed/resilience tests where eight nodes of an HP AlphaCluster all open multiple connections to a MySQL server running on a Pentium/Linux box and collectively throw about 2 gigabytes of data at it. The server barely breaks into a sweat, and even with a somewhat complicated set of tables, it loads the data in under an hour.

> *I have a few concerns about updating databases. Currently, when a new
> record is inserted, a unique id, which is auto-updated is generated in
> the database. This works well, since there is very little possibility of
> two or more clients submitting a request at the same point in time.*

I'm assuming that when you say "a unique id, which is auto-updated", you actually mean that you've defined primary key with the auto_increment qualifier. In this case, MySQL will allocate each new record an ID that is unique within the table. Your clients can retrieve the value of the ID using the getGeneratedKeys method of the java.sql.Statement which executed the insert operation.

> *The problem occurs when connection to the database server is lost. In
> this situation, the application should still continue to work in
> offline" mode, using either an internal database (hsqldb) or writing to
> a XML file. I have not decided what yet. Now when the connection to the
> database server is restored, how would I update the "main" database
> server with records from each client?*

>

> *Consider the following scenario :*

>

> *User A and User B are both connected to the database server, and the
> current record id is 200. Now the database connection is lost. Both User
> A and User B create a new record (internally). Both now are on record id
> 201. When the database connection is restored, how would I insert the*

comp.lang.java.databases: Re: Confusion about database updates

- > *records of both clients into the database? There cannot be two records*
- > *with the same primary key of 201.*

The answer is **not** to allocate an ID to the records that you are storing internally. Let the MySQL server do that when your client application manages to re-establish a connection.

After all, unless I've misunderstood you, User A and User B won't be aware of one another's records until both client programs are able to re-connect to the MySQL server and insert the records that they have been keeping "on hold". The database is the only way for one user to know **anything** about the data that other users are holding.

But taking a broader view, you have to ask yourself what scenarios are likely to lead to your client programs losing their connection to the database server?

The MySQL server itself could crash, though this is **very** unlikely. Speaking from personal experience, I've run MySQL servers for nine months at a stretch, and only had to shut them down because our sysops wanted to upgrade the operating system on the host machine.

The client program could crash, but in that case you don't have time to write the unsaved records to an XML file or alternate database anyhow!

You could lose the network connection between the client program and the MySQL server. Is your application running on a corporate intranet? Or across the global Internet? [And do you have crazed back-hoe operators in your neighbourhood? ;-)] In this case, both the MySQL server and the client program are still running, so you **can** write the unsaved records to an alternate datastore.

David Harper
Cambridge, England