Fastest way for doing INSERTS using IBATIS

Asked 16 years, 1 month ago Modified 16 years, 1 month ago Viewed 12k times

4

I need to insert 20,000 rows in a single table (SQL Server 2005) using iBatis. What's the fastest way to do it ? I'm already using batch mode, but it didn't help much:

```
try {
  sqlMap.startTransaction();
  sqlMap.startBatch();
  // ... execute statements in between
  sqlMap.commitTransaction();
} finally {
  sqlMap.endTransaction();
}
```

java ibatis bulkinsert

Share Improve this question Follow

asked Nov 13, 2008 at 20:56
muriloq
2,712 ● 5 ● 29 ● 32

4 Answers

Sorted by: Highest score (default)

4

Barring the bulk loaders others are referring to, let's consider how to best do it through SQL. (And the bulk loaders don't work well if you're sending intermixed data to different tables.)

First, you shouldn't be using whatever abstraction layer you're using, in this case iBatis, as it effectively will offer you little value, but that abstraction layer will have some (not necessarily much, but some) CPU cost. You should really simply use a raw database connection.

Next, you'll be sending in a mess of INSERT statements. The question is whether you should use a simple string for the statment, (i.e. INSERT INTO TABLE1 VALUES('x','y', 12)) vs a prepared statement (INSERT INTO TABLE1 VALUES(?, ?, ?)).

That will depend on your database and DB drivers.

The issue with using a simple string, is basically the conversion cost from an internal format (assuming you're inserting Java data) to the string. Converting a number or date to a String is actually a reasonably expensive CPU operation. Some databases and drivers will work with the binary data directly, rather than simply the string data. So, in that case a PreparedStatement could net some CPU savings in potentially not having to convert the data.

The downside is that this factor will vary by DB vendor, and potentially even the JDBC vendor. For example, Postgres (I believe) only works with SQL strings, rather than binary, so using a PreparedStatement is a waste over simply building the string yourself.

Next, once you have your statement type, you want to use the addBatch() method of the JDBC Statement class. What addBatch does is it groups up the SQL statements in to, well, a batch. The benefit is that instead of sending several requests to the DB, you send a single LARGE request. This cuts down on network traffic, and will give some noticeable gains in throughput.

The detail is that not all drivers/databases support addBatch (at least not well), but also the size of your batch is limited. You most likely can't addBatch for all 20,000 rows and expect it to work, though that would be the best bet. This limit, also, can vary by database.

For Oracle, in the past, I used a buffer of 64K. Basically I wrote a wrapper function that would take a literal INSERT statement, and accumulate them in 64K batches.

So, if you wanted to bulk insert data through SQL via JDBC, those are the ways to do it. The big improvement is the Batch mode, the Statement vs PreparedStatement is more to potentially conserve some CPU, and maybe network traffic if your driver supports a binary protocol.

Test, rinse, and repeat until you're happy enough.

Share Improve this answer Follow

answered Nov 13, 2008 at 21:51
Will Hartung
118k ● 20 ● 133 ● 207

2

Although this is not specific to your db server, I have previously had success writing the rows out to a local file in csv format and then having the database importing the file. This was considerably faster than insert statements or even a batch insert.

Share Improve this answer Follow

answered Nov 13, 2008 at 21:21
benPearce
38.3k ● 14 ● 67 ● 96

made the same experience with MySQL recently: inserting data using plain INSERT INTO statements loaded from a file was significantly faster than doing a batch insert as describe above. – Benedikt Waldvogel Jan 18, 2010 at 23:10

2

In SQL Server, the fasted way to insert records in batch is using BULK INSERT. However, this method loads the records from a text file rather than directly from your application.

It also doesn't take into the account the time spent creating the file. You may have to weigh if that offsets any speed gains from the actual insert. Keep in mind that even if this is a little slower overall, you'll end up tying up your database server for less time.

The only other thing you might try is inserting (staging) the batch into a completely different table (with no indexes or anything). Then move the record from that staging table to your target table and drop the staging table. This would move the data to server first, so that the final insert could all happen with sql server itself. But again: it's a two step process, so you'll have to count the time for both steps.

Share Improve this answer Follow

answered Nov 13, 2008 at 21:23

Joel Coehoorn
415k ● 114 ● 577 ● 813

1 +1 Bulk Loading from a file. Writing a flat file is usually amazingly fast. – S.Lott Nov 13, 2008 at 21:29

1

Bulk inserts are best done using the database's own bulk loader tools. For Oracle, it's SQL*Loader, for example. Often these are faster than anything you could ever write.

Share Improve this answer Follow

answered Nov 13, 2008 at 21:02

S.Lott
391k ● 82 ● 517 ● 788