

Re: Help with Master/Detail UI in ADPs

Source:

<http://www.tech-archive.net/Archive/Access/microsoft.public.access.adp.sqlserver/2006-02/msg00164.html>

- *From:* "Tom Ellison" <tellison@xxxxxxxxxxxx>
 - *Date:* Tue, 14 Feb 2006 12:23:13 -0600
-

Dear Patrick:

If I had a nickel for every time . . .

Answering your last question first:

No, MSDE is not limited to 5 concurrent users, it is limited to 5 concurrent processes. It is commonly used with 50 and even 100 users.

5 concurrent processes means 5 query threads (queries) running at any one time. If each user runs a query every 10 seconds (and that's a very high average. I usually use 30 seconds) and each query takes half a second (not unlikely if properly optimized) then with 100 users the engine is busy with an average of 2 threads at any one time. It would be statistically rare even in this rather extreme case to have 5 concurrent queries running. Even then, it doesn't bomb, it throttles. Everybody go get a cup of coffee, and it will be back in a minute.

The users whose connection to the server is idle at any moment don't count. So, unless the usage is incredibly high, or the query demands are very complex (or not optimized) your 20-some user scenario is almost certainly no strain. You'll want good server hardware and plenty of memory in it for cache to improve this even more. You may get the average processing time (not including the time to transmit results over the network, just the time to prepare them in the server's memory) down well under the half second estimate.

For an application such as you described, this can be the best bang for the buck.

Don't quote me on it, but the new SQL Express does not appear to have that limit or any throttle. Also, it has doubled the database size limit and has some great new features, and improved performance. You might research that. I know I'm looking forward to trying it when I get a chance. I still haven't figured out what the licensing restrictions are for it, but I know when I download it I'll have work for my lawyer to figure that out. Ouch!

More, inline, below.

Re: Help with Master/Detail UI in ADPs

Tom Ellison

"Patrick Jackman" <pjackman@xxxxxxxx> wrote in message
news:ObvfyoYMGHA.524@xxxxxxxxxxxxxxxxxxxxxxxxxxxx

Thanks for the ideas Tom.

I think I would prefer temporary tables to avoid having to add filters everywhere to exclude the "tentative" records. Temporary tables on the server would need to hold the user's log in along with the screen's hWnd.

You can still create temp tables on the server. I use the computer network name and the hWnd (window handle number) of the instance of Access to create this, along with a name for that "local" table. If there are multiple instance of the application open on the same client, this still ensures uniqueness.

Putting temp records in the table is superior for performance. Deleting them from one place and inserting them another is a lot more work for the server than just changing a column. You may want to re-think how much work is involved in implementing the Tentative column. I wouldn't think that's much work at all.

If I were to have a screen with an unbound parent and 4 child screens bound to temporary tables on the server, would this consume 4 database connections?

Again, counting connections is not the point in any case.

Regarding MSDE, I thought it was performance limited to 5 concurrent users.

Is it commonly used with 20 users?

Not so at all, as I explained.

Patrick.

"Tom Ellison" <tellison@xxxxxxxx> wrote in message
news:eNOES1HMGHA.1132@xxxxxxxxxxxxxxxxxxxxxxxxxxxx

Re: Help with Master/Detail UI in ADPs

Dear Patrick:

You have a very good design paradigm going there.

Here's an approach I prefer for ADPs with MSDE or SQL Server.

Make sure user's log in, and keep track of who is posting records. Post the user to all new records. Allow updates to take effect immediately.

Mark records as being tentative (another column in the table) and put them right into the database immediately. Having a record in the table, but tentative, is really just the same thing as having it in a local table, except that if the user's computer breaks down, that user has access to those same records from any other system though his login. All the business rules can be in affect immediately through the data being placed directly in the public database, so if another user begins entering, say, an invoice already entered by tentatively, it is blocked initially. That user will know the invoice has been at least partially entered, and by whom.

When you query for reports, you must determine whether the tentative data belongs on the report or not, and filter it out if not.

All forms are then bound (if desired).

Given decent server hardware, moving from Jet to MSDE, you'll find the performance is unlikely to be an issue. 20 users may be a pretty full load for Jet, but will be a light load for most applications with MSDE. This is especially true with a "small" back end of 700 MB. If you have a gigabyte of memory in the server, it will quickly cache all the repeatedly accessed data, and you'll be flying.

Perhaps this approach will be attractive to you, and can be adapted to your needs.

Tom Ellison

"Patrick Jackman" <pjackman@xxxxxxxx> wrote in message
news:eiZNRjHMGHA.2216@xxxxxxxxxxxxxxxxxxxxxxxxxxxxx

In Access DAO/Jet applications, I handle Master/Detail relationships by using unbound fields for the Master table and a subform bound to a local temp table for the Detail rows with OK, Apply and Cancel buttons on these forms. If the user clicks OK or Apply, I do concurrency checks then I write from the unbound fields to the Master table on the server

Re: Help with Master/Detail UI in ADPs

and from the local temp table to the Detail table on the server for rows that are new, dirty or deleted. If the user clicks Cancel, I just close the form.

In some applications I have up to 5 Detail subforms against a Master without any issues. Using temp tables allows me to validate business rules between the various M/D entities before saving any changes. I also use multi-instance forms to allow users to have multiple views of M/D data open concurrently and I keep track of what data in a local temp table belongs to which form instance by loading the subform's hWnd along with its data.

Using local temp tables for Detail entities and local tables for lookups allows me to have acceptable performance with 20 – 25 concurrent users against a 700 MB backend. This approach is code intensive but I've had time to streamline it over the last 12 years of working with Access full-time.

I would like to start using SQL Server 2000. If I connect with ODBC I can continue with the same design approach using local temp Jet tables.

But I would like to consider using ADPs to avoid the reported performance penalty of ODBC.

Is there a "best practice" for Master/Detail form design with ADPs?

I've tried several approaches without success:

1. Bind ADO adLockBatchOptimistic recordsets to the Master and Detail forms then set the ActiveConnection = Nothing. When I reconnect and issue UpdateBatch, the M will update in certain situations but the D never updates.

2. Create an in memory ADO recordset for the Detail, load it

Re: Help with Master/Detail UI in ADPs

with data
and bind it to the Detail subform. The binding fails with
#Error in each
field of the subform.

3. The "Access 2002 Enterprise Developer's Handbook"
approach on p. 281:
"Using Transactions with Bound Forms". It works until I
attempt to open
a second form instance on the same row while the 1st is still
in a
transaction. And I've read posts here suggesting this is not a
best
practice.

What is the preferred approach for dealing with this issue in
ADPs? Has
anyone written about it from a real world perspective?

Patrick