

Re: Homegrown synchronization

Source:

<http://www.tech-archive.net/Archive/Access/microsoft.public.access.replication/2007-01/msg00036.html>

- *From:* "rdemyan" <rdemyan@xxxxxxxxxxx>
 - *Date:* 4 Jan 2007 09:34:00 -0800
-

Currently I'm thinking about whether to continue to just copy the downloaded unzipped backend over the existing remote backend or to use the downloaded backend to simply update the existing remote backend.

The latter method helps with a potential latency issue; namely the downloaded production backend may not contain changes made by the remote user that were sent up to the server (presumably a short time earlier).

OTOH, comparing the downloaded production backend with the existing remote backend will be a slow process. While the updates/additions can be filtered resulting in fast performance, the deletes cannot. So for checking for deletes it is a table by table comparison. Since I'm already doing this when the local MyApp creates the incremental update file to send to the server, I have an idea of how long this will take (of order 30 seconds on my machine). Four of the 100 or so tables are responsible for half of this time. While I suggested a method to handle these four separately when creating the update DB from MyApp, that won't work for updating the local backend from the downloaded server backend. I can't see anyway around going table by table in an unfiltered way.

Again, this points to the inherent table design problem from the beginning; namely not having a deleteflag in the tables to identify deletions as opposed to, what I'm currently doing, which is to simply delete the records from the table. To others who may be considering implementing their own synchronization, I would recommend that you take David's advice and implement the deleteflag method in tables. It will speed up creation of updates and updating dramatically. I do want to point however, that actually deleting from tables has not been a problem for us until I wanted to implement this "homegrown synchronization".

.