Tech Tip #6: Preloading Data After a Server Restart

Overview: Usefulness of preloading data via either BUTIL or BtrvLoad

So, if you followed our last Tech Tip, you now have a fast server with lots of memory allocated to the database cache. However, every time you reboot the server, things slow to a crawl again. Why is that?

The answer is simple – when you restart the server or restart the database engine itself, all of the data that was previously stored up in the database cache is gone. The next time you go to read a given page or set of pages, the database engine has to get it from the disk once again – and as you know, disk can be hundreds of times slower than memory accesses!

If you are rebooting your server, it is likely happening in the middle of the night. You can take advantage of this relatively quiet time of day to ask the database engine to pre-load some of your critical data pages back into memory, so that it will be there when the users sign on a few hours later. Unfortunately, there is no simply way to do this – we actually have to ask the database engine to access the files and get the data into memory.

One way to do this is with the included **Maintenance Utility** tool, which is called **BUTIL** on the command line. The **BUTIL** program provides a number of useful functions for managing files, but we will only be using one of them here – namely the `-RECOVER` option. This option reads each record from a given file in physical order and writes it to an unformatted file. As we really do not care about the unformatted file anyway, we are going to substitute the "null device" for the target file, which essentially just throws away the resulting data, saving us some disk accesses, and thus running faster.

```
D:\Data>butil -recover CredAud.btr NUL

Btrieve Maintenance Utility 11.30.051.000
Copyright (C) Pervasive Software Inc. 2012
All Rights Reserved.

BUTIL-71: BUTIL has recovered 0 records so far.
BUTIL-71: BUTIL has recovered 200 records so far.
BUTIL-71: BUTIL has recovered 277 records so far.

The command completed successfully.
```

This process read each of the 277 records and simply discarded the results. A side effect of this reading process, though, is that the database cache is now full of the data pages from this file, and any subsequent file accesses should execute more quickly, since the disk is not needed to access pages already in cache.

This process can then be extended by writing a simple batch files to load data records from as many tables as you want, like this:

```
BUTIL –RECOVER CredAud.btr NUL
BUTIL –RECOVER CasLdgr2.btr NUL
BUTIL –RECOVER CliLdgr2.btr NUL
```

To run multiple **BUTIL** operations in parallel, you can manually split the process into multiple batch files as well, and then launch all of the batch files at the same time. For best results, try to split the data file sizes evenly across the batch files so that they complete at about the same time.

Another option is to use a tool that was built especially for the purpose of pre-loading data. Goldstar Software's **BtrvLoad** tool is one such tool. Instead of reading one complete record from one file in each database call, **BtrvLoad** is designed to issue a single call to read only 1 byte from each of the next 5000

records at a time. The net result is a rapid set of requests that hits all of the data pages of a file as rapidly as possible. Couple this with the ability to spawn multiple threads (as many as 120 at one time) to read data from many different files at the same time, as well as support for wildcards, and you have a powerful data loading tool indeed. Here is an example of **BtrvLoad** running to load data from a large batch of files…

```
D:\Data>btrvload *.btr /t10
BtrvLoad Version 1.20: 03/09 (C)2013 Goldstar Software Inc.
Registered to Goldstar Software Inc. (Site License GS)
Searching for <*.btr>...
Press <Esc> to terminate the file scanning loop.
Status   0 after       277 records from CredAud.btr
Status   0 after      6530 records from CasLdgr2.btr
Status   0 after      4153 records from CliLdgr2.btr
. . .
Status   0 after     93378 records from Files.btr
Status   0 after     68558 records from ToDo.btr
Status   0 after    212780 records from NotesMem.btr

File loading completed for 119 files.
```

In this case, all 119 files were loaded into the cache at the same time, with a single command. For more information about **BtrvLoad**, check out the Goldstar Software web site at:

http://www.goldstarsoftware.com/btrvload.asp

With such great power comes great responsibility, right? No matter what solution you pick, you should NEVER try to load more data than will fit in the L1 cache, as once the engine has to start throwing away data blocks, you are simply wasting time, energy, and disk accesses. If you've been following along with these Tech Tips, you'll know how to monitor the L1 Cache Usage. Watch the graph as the data loads, and you'll get a good idea if you have enough memory for your L1 cache or not.

Furthermore, a huge data load at a time when people are trying to use the database actively for normal tasks may also slow the system down to a crawl if the disk channel becomes overloaded. In other words, be VERY careful about running any such loading tool, and use it only during non-peak times.