Tech Tip: Monitoring With Continuous Operations Mode

Written by Bill Bach, President of Goldstar Software Inc.

Several Tech Tips and White Papers have been written over the years about using Continuous Operations mode (ContOps) in the database to ensure proper backups for your database environment. However, little has been said about monitoring your system while it is in ContOps mode, so we have created this Tech Tip to provide some simple pointers on what to watch out for to make your backup process as seamless as possible.

What is Continuous Operations Mode?

Continuous Operations Mode is a special mode within the Pervasive database, available since the release of Btrieve 6 over 15 years ago, that allows you to perform a snapshot backup of your environment at any time of the day or night, without getting all users out of the database.   Briefly, ContOps mode queues up disk writes to a separate physical file with a ^^^ extension, called a "delta file", while you take a backup of your files. By segregating the writes for a period of time, you can obtain a backup that includes the state of the database from a single moment in time.

If you have never tried an online backup, or are otherwise are unfamiliar with ContOps mode, then you may wish to check out some of these links before continuing with this Tech Tip:

http://ww1.pervasive.com/library/docs/psql/950/advops/advops-09-5.html
http://www.goldstarsoftware.com/papers/ValidBackups.pdf

What Must I Worry About With Continuous Operations Mode?

As a special database mode that "queues up" the disk writes to the database while it is active, you should be aware that ContOps does add a certain amount of overhead to your environment for every record insert, update, or delete. As such, you need to be careful of the following situations:

- **Running Out of Disk Space**
- **Mass Updates or Purges**
- **Server Crashes While In Continuous Operations Mode**
- **Failing to Exit Continuous Operations Mode**

Let's look at each of these issues separately.

How Do I Avoid Running Out Of Disk Space?

Each database write (an insert, update, or delete) will generate one or more disk writes to the database files, which will be sent to the separate Delta File. This means that even updates and deletes can start to rapidly use up your free disk space. Running out of free space while in Continuous Mode can cause a fair number of problems, so it should be avoided at all costs.

You should always monitor the disk space on the database volume. From experience, we know that database volumes tend to collect "gunk" in the form of backup files, old converted data, and so on.

Sometimes, you need to sit down with the application developer and the various system administrators to determine which files are really needed and which can be deleted.

Keeping the system clean is the first part of the puzzle. The second part is making sure that the data volume doesn't get used for storing downloads from users or huge log files from applications. Your Pervasive database server, if running as a dedicated box, should be ONLY used for database files. Keep the other files (such as application installation files, music video downloads, and other sundry things that tend to collect on a server) segregated onto file servers. This not only keeps the system clean, but it allows the server to tune its memory usage for the database and not try to swap out the database for file system cache.

You should be checking your disk free space at least weekly, and daily (or more frequently) for any mission-critical system. While you are at it, you should also monitor the disk free space on the boot volume, as running a Windows server out of space on the C: drive is a big problem, too.

How Do I Avoid Mass Updates or Purges?

As mentioned above, each database write translates to multiple page writes at the data file level. While a database is in Continuous Operations mode, you want to avoid massive numbers of record updates or deletes. Usually, this is not something that is planned. Instead, a company will schedule backups for off hours, and then decide that the same time frame would be an ideal time to archive and purge old data. The resulting collision caused the delta files to grow very rapidly, raising the possibility of running out of disk space.

However, the problem is worse than a pure disk space issue. Because the way the database engine handles the delta files is far less efficient than the way it handles production PSQL files, doing a lot of changes (or deltas) in a database can have a severe impact on the performance of the server overall. In extreme cases, this can cause the entire server to become so bottlenecked at the disk subsystem that the console becomes unresponsive. Ugh!

There's no easy way to *always* avoid this issue. Good communications between the database or server administrator (who schedules the backups) and the application key users (who are likely to schedule such archive jobs) is one real key to success. Another is ensuring that the amount of time that you spend in ContOps mode is always minimized, as we'll explain in the next section.

How Do I Avoid Server Crashes While In Continuous Operations Mode?

If the server crashes while in ContOps, you may need to take some manual steps to properly recover the environment and get it ready for the next backup. These steps are spelled out in the Goldstar Software white paper on proper backups, via the link provided above. Pervasive's Backup Agent can help with this recovery process by keeping track of which files were in ContOps mode at the time of the failure, and then opening these files long enough to allow the roll-in to complete.

Unfortunately, there is no way to tell when a power failure, OS crash, or even user-induced failure will occur, so you cannot completely avoid this issue. However, you can take some special steps to minimize the likelihood of this situation arising in the first place. The most important thing to do is to minimize the time spent in ContOps mode in the first place. If you can get into ContOps, make your backup copies, and get out in 10 minutes instead of 60 minutes, then the odds of crashing while in this state are reduced by a factor of 6.

How do you minimize the backup time? There are several ways. First, don't back up to tape, which is usually notoriously slow. Instead, consider backing up to a second hard drive on the same server FIRST, and then copying the backup to tape after you have exited ContOps mode. Second, don't use a RAID5 volume as a target for the copy, which is notoriously slow for disk writes. If you have a RAID controller and can set up a separate striped (i.e. RAID0) volume, then you can get a LOT better throughput on the disk writes. If you don't have a RAID controller, then even a separate single drive directly attached to the server can be quite helpful in reducing backup times. Again, once the data is in the staging location, then you have until your next backup to copy the data to a different server, to tape, or even to an off-site location. With today's terabyte-sized drives, you may even have room for multiple backup copies on a single drive, providing tiered backups for very little cost.

How Do I Avoid Failing to Exit Continuous Operation Mode?

Whether you use the BUTIL -STARTBU and -ENDBU commands or the Pervasive Backup Agent to handle your move into and out of Continuous Operations mode, it is imperative to make sure that the script runs to completion and brings the database out of ContOps mode when it is finished. Why is this critical? Imagine if the script failed, and the database were left in ContOps all day long. First, your risk of crashing the server while in ContOps just skyrocketed. Second, the chance of running out of disk space also just skyrocketed. Finally, the lower efficiency of the delta files means that disk utilization will continue to increase and increase until the entire server gets bogged down with Disk I/O that it cannot handle many other chores -- and your 24x7 environment will come to its knees. In fact, if you don't address this before the NEXT backup, then your problems will compound and your backup will now be two days old. Ugh!

Let's review a simple scripts that would commonly be used for ContOps backups:

```
BUTIL -STARTBU @C:\BACKUP\FILES.LST
XCOPY D:\APPLICATION\DATA\*.DAT F:\BACKUP1
BUTIL -ENDBU /A
```

This script has three simple lines -- one that puts the database from the list of files (FILES.LST) into ContOps mode, one that copies the data (*.DAT) over to the backup location, and one that takes the database out of ContOps at the end. For most purposes, this will suffice perfectly. However, what happens:

- if the application removes a file that is in the list, and the STARTBU fails?
- if there is a new file that is in use and cannot be copied?
- if the ENDBU never runs?
- if any files get "stuck" in continuous operations mode?

Let's see how to handle these issues one at a time.

Step 1: Getting Into Continuous Operations Mode Properly

The first problem is all about ensuring that the "right" set of files gets into ContOps mode for the backup. If you are using a file list like in this script, then you should review the list periodically, especially after any application updates are provided, to make sure that the file list is still accurate. You can detect the problem (after the fact) by checking the status code coming back from BUTIL -STARTBU, using an additional line like this:

```
BUTIL -STARTBU @C:\BACKUP\FILES.LST
IF ERRORLEVEL 1 ECHO Error: The STARTBU Failed!
```

Obviously, having a simple message print on an unattended process doesn't buy you a whole lot, but you could change the ECHO to a GOTO statement and perform some sort of notification.

Another solution is to dynamically create your FILES.LST file on the fly, using a command like this:

```
DIR *.DAT /S /B >C:\BACKUP\FILES.LST
BUTIL -STARTBU @C:\BACKUP\FILES.LST
IF ERRORLEVEL 1 ECHO Error: The STARTBU Failed!
```

However, this solution has its difficulties, too. What if someone creates a backup of a corrupted file and calls it BADDATA.DAT? Now this file is going into ContOps every backup, and because it gets a new date stamp, you'll think it is in use by the application.

The easiest way to handle this problem is to leverage Pervasive's Backup Agent to put the files into and out of ContOps mode, as this ensures that only files are open and present get put into ContOps mode:

```
PVBACKUP -ON
```

Of course, you may wish to check the ERRORLEVEL and handle any Backup Agent failures, too.

Step 2: Copying All Files

The second problem is all about copying the data. If a file is in use or is otherwise inaccessible, then the XCOPY line will fail, and it will never terminate. This halts the script and causes problems in the environment. You can avoid some of this by adding a few simple command line switches to the XCOPY line, including /C (continue on error), /R (overwrite read-only files), /Y (Suppress prompting), like this:

```
XCOPY /C /R /Y D:\APPLICATION\DATA\*.DAT F:\BACKUP1
```

Be sure to check ERRORLEVEL on this process, too!

While XCOPY is built into the operating system, it isn't the most robust application that you can use for copying files. Other ones to check out include XXCOPY (http://www.xxcopy.com), ROBOCOPY (built into Windows Vista, 7, and 2008), Beyond Compare (http://www.scootersoftware.com), and many others. Many of these other tools are either free or very low cost and well worth the investment. In any event, you'll want to spend some time with the documentation, as options and switches abound. (In fact, XXCOPY has over 230 command line options!)

Step 3: Making Sure the ENDBU Runs

When the copy is done (and you have checked for the ERRORLEVEL accordingly), the last step is to take the database out of ContOps mode. This is quite simple, being done with either a BUTIL -ENDBU command, or a PVBACKUP -OFF command, depending on your tool of choice.

Seems simple, right? Well, what about a check to make sure that THIS step works correctly, too? Again, it is imperative to check the ERRORLEVEL coming from the command to see if it succeeded or not. Even better, though, is to provide some sort of positive verification that the entire backup process really finished. Instead of reporting errors here, you may wish to ALSO report success as well.

Reporting success from a script can be done in a few different ways. You can do something simple like write a "Backup Completed" entry to a log file somewhere, then check that log file every morning and make sure that it was updated on the last backup. If you see that the log file date is not changing, then your script isn't running. Another solution is BMail, a freeware program that can send Email right from your batch file, like this:

```
BUTIL -ENDBU /A
BMAIL -s smtpsrv -t to@co.com -f from@co.com -h -a "Backup Status %ERRORLEVEL%"
```

This command sends an Email message after the process finishes with the status of the BUTIL. Of course, a more robust solution will have error checking each step of the way and report the exact problems encountered in the Email, and perhaps include the backup log for additional information. With a script like this, if a backup goes by without sending you an Email, then you'll know that a fatal scripting problem has occurred.

However, the termination of the command doesn't always mean that the files have all come out of continuous mode.

Step 4: Making Sure No Files Are Stuck In Continuous Operations Mode

We should note that successful completion of the backup script does NOT guarantee that all files have been removed from ContOps mode successfully. Since this process can take from just a few seconds to several hours to complete (depending on the number of database writes that were queued up), it is often very difficult to even script this task.

Luckily, there are some ways that you can find out if any files are in ContOps mode or not. One such solution simply checks for the existence of the delta files. You can run this script about an hour after your backup is supposed to finish and it will send you an Email if there are any delta files left:

```
BUTIL -ENDBU /A
D:\APPLICATION\DATA\*.^^^^^^
IF ERRORLEVEL 1 GOTO :END
BMAIL -s smtpsrv -t to@co.com -f from@co.com -h -a "Backup Incomplete!"
:END
```

Let's review this short script. The first line seems out of place -- why do we do this? Simple -- if you know that your backup was supposed to finish an hour earlier, then this BUTIL command will return an error, which we safely ignore. However, if the backup script did get stuck for any reason, this should serve to start removing files from ContOps right now. The second line checks the directory for any delta files. Note that the caret (^) is an escape character in Windows, so you have to double it to get it to work in the batch file. The third line checks the return from the DIR command. If the DIR returned "File Not Found", then ERRORLEVEL will be 1, which means that there are NO delta files, so we cleanly exit. If the ERRORLEVEL is 0, then files were found, and we want to send ourselves an Email with this notification, so that we have time to resolve it before the next backup.

Another thing that you can do is to ask the database engine if any files are still in ContOps mode. Although the Pervasive Monitor tool and the Java-based BMON tool don't report this attribute when you look at the Open Files information, the DTI interface DOES make this flag available, so developers can write their own applications that expose it, such as Goldstar Software's PSConfig tool:

```
D:\APPLICATION\DATA>psconfig /mf
PSConfig Version 3.31: 06/26 (C)2008 Goldstar Software Inc.

Number of Open Files: 1

Mode LkTrRoCoRI Handles Open Time   FileName
Norm N N N Y N        1 03-13 15:17 D:\APPLICATION\DATA\MYFILE.MKD
```

Note the column "Co" with the "Y" in it, indicating that the file is currently Continuous Operations mode. It would also be possible to parse this output and detect ContOps files from a script.

Summary

As we've seen, the process of getting snapshot backups on your Pervasive PSQL database can be easily scripted with just a few lines. Adding a few lines more, though, can help ensure that your backups continue to work properly, that your server doesn't get bogged down, and that you have some clear visibility into your script, to help keep your data safe.

Author Information:

Bill Bach is the Founder and President of Goldstar Software Inc., a Pervasive reseller in the Chicago area that specializes in providing Pervasive products, services, and training to its customers in North America and abroad. Bill has written numerous tools and utilities to help system administrators and database developers work with their Pervasive database environments, and his training classes for Pervasive PSQL and DataExchange are the most comprehensive classes available. Get more information from http://www.goldstarsoftware.com.