The Resource for Users of IBM zSeries & S/390 Systems
zJOURNAL

October / November 2008

DB2 Database Maintenance and Recovery:

Getting Past the Common Misconceptions.

by Brenda Honeycutt and Craig S. Mullins

Every DBA strives to maintain a healthy DB2 environment every day. However, there are numerous lingering misconceptions about database health to which many organizations fall prey. These misconceptions occur because DBAs believe in, or apply concepts that are, objectively false.

Like old wives' tales, some misconceptions exist simply because they have been passed down from older generations. In terms of DB2, this means that even best practices true for older DB2 versions may no longer be best. This article comprises a number of the most common ones about DB2 maintenance and recovery.

"Lies, Damned Lies, and Statistics"

This well-known saying is part of a phrase usually attributed to Benjamin Disraeli and popularized in the U.S. by Mark Twain. Although the origin of this statement may be questioned, its meaning is clear. It refers to the persuasive power of numbers and neatly describes how even accurate statistics can be used to bolster inaccurate arguments. When it comes to DB2, statistics are the lifeblood of database health.

With that said, it is relevant to distinguish between the two types of statistics available to us: catalog statistics and real-time statistics. Although originally intended as food for the DB2 Optimizer, historically we have also needed analytical RUNSTATS that updated catalog statistics in order to drive our maintenance processes. With the advent of real-time statistics there is absolutely no need to waste time and resources by running analytical RUNSTATS. It's time to get back to the future and use RUNSTATS strictly for the purpose that IBM intended: driving access paths for optimal performance.

For the skeptics, it’s time to dispel the delusion that using real-time statistics increases CPU costs. DB2 collects the statistics in real-time anyway, even if you are not using them. Experience and customer polls indicate that externalizing RTS has practically no impact on CPU. And the fact that they are collected in real time not only means that they're available when you need them, it means that they're up-to-date and accurate.

All Backups Are There!

So you've got your statistics straight. They reflect your database activity accurately and you can rely on them as a basis for backups. The real-time statistics that pertain to backups include:

- **COPYLASTTIME** - the timestamp of the last full or incremental image copy on the table space or partition.
- **COPYUPDATEDPAGES** - the number of distinct pages that have been updated since the last COPY.
- **COPYCHANGES** - the number of insert, delete, and update operations since the last COPY.
- **COPYUPDATELRSN** - the LRSN or RBA of the first update after the last COPY.
- **COPYUPDATETIME** - the timestamp of the first update after the last COPY.

Even after progressing to this point, there are other common fallacies regarding database backups outside the realm of determining when to do them.

Migrating too early to tape. Some organizations migrate current backups to tape immediately in order to free up disk space. Well, this approach will free up *some* DASD, but the time needed to recall that tape will have to be added to the recovery time. So which is more important: disk space, or time to recover? As a general rule of thumb, you should consider keeping the most current backups on DASD and migrate them to tape only when a new backup is taken. Of course, your tape versus disk strategy for backups will be heavily dependent on the nature of the data being backed up and the recovery time objectives required for that data.

Some shops use dual backups. Rather than an attempt at cost savings for disk space, you might consider limiting dual backups to objects that require a critical copy. A critical copy is one taken after a REORG or a LOAD LOG NO, as well as a REORG with a rebuild of the compression dictionary for compressed table spaces. In other cases, DB2 always has the opportunity to fall back to a prior copy in the unlikely event of an unusable copy.

Virtual tape storage is as good as DASD. Not. There is a hard-and-fast rule concerning virtual tape systems (VTS) that everyone should follow: although choosing a less expensive storage media, such as VTS, for backups is okay, do not overestimate the speed of VTS. When deciding on the right backup device, always keep mount and recall times in mind. For example, mount and recall times for small objects are often bad in relation to recovery time.

Oops, the archive logs are on tape! If your archive logs are on tape or VTS, a parallel recovery is not possible. Keeping archive logs on DASD isn't an option for some shops. When backing up archive logs to tape, do not follow the rule of thumb pertaining to the 28672 block size normally optimal for tape. Instead, use a block size of 24576. To enable parallel processing for recovery, the archive logs on tape must be copied to DASD, and 24576 is the preferred size; it’s been the ZPARM default since DB2 V8.

Think about the trade-off in saving tape versus recovery speed, and your decision should be obvious. Before backing up any logs to tape, it’s a good idea to always have at least 24 hours covered by the active log and the least 48 hours by archive logs on DASD.

I can use mirroring instead of dual logging. Some organizations mistakenly believe that single active logging is fine if mirroring is used. But what happens if a technical error with the channel, the device, or the firmware of the device occurs? The data will be mirrored as inconsistently as the source. You should always do dual logging, *without exception*. The DB2 log is the most important resource to assure data consistency.

Fast, Faster, Flash It. IBM introduced BACKUP SYSTEM and RESTORE SYSTEM in V8 to take full advantage of the immense speed gains in FlashCopy I and II technology. With FlashCopy, the I/O subsystem does the copy; freeing the CPU to do other more important work. FlashCopies enable ultra-fast image copies and now support incremental copies at the volume level and over multi-volumes when using "consistency groups".

Some people have hailed FlashCopy as "the end of the image copy," but that isn't the case for all installations. Indeed, it's absolutely the best for select, large, enterprise-critical objects, but you should choose those objects wisely. FlashCopy can be expensive in terms of the hardware cost and the system topology. You must have twice the number of disks to flash a given "pool," which can make keeping generations of backups extremely expensive. Even if you then off-load the flash pool to tape, you fill up tapes with gigabytes of unchanged data. The introduction of incremental FlashCopies has alleviated some of this space management nightmare, but not all. Incremental flashes are at the volume level. A data set that is multi-volume must be handled with great care—unless Persistent, Change Recording, Background Nocopy, and Inhibit Target write options are all in effect.

FlashCopies are not registered anywhere, unless from the BACKUP SYSTEM DB2 command. You must remember that you did a flash, and remember what was flashed. Flashes do not reset the COPY PENDING status. Do not, however, allow these facts to discourage you from using FlashCopy outside of DB2. Some third-party tools keep track of FlashCopies and reset the copy pending status. Maybe yours is one of these.

Copying Indexes? Some folks blindly continue on, administering their DB2 systems with little or no changes even as new versions roll in with new functionality. We've been able to use COPY to make image copy backups of DB2 indexes for some time now, but many shops ignore this capability completely. If the index was created (or altered) with the COPY YES parameter, then it can be recovered using either the REBUILD or RECOVER utilities. An index defined as COPY NO only can be recovered using the REBUILD utility.

It can make sense to backup indexes for recovery because the REBUILD utility can take a long time to complete for very large amounts of data or when a large number of indexes exist. For example, in one simple test, recovering a 50 million row index substantially outperformed rebuilding that same index. Of course, your mileage may vary.

Another reason for backing up indexes is in the case of a recovery of both the data object and the index. COPY YES indexes can be restored at the same time as the data is being restored, thus reducing the overall outage time.

Keep in mind, however, that all index copies must be full copies as incremental image copies are not permitted for indexes.

Recovery Ready, Set, Go

If you prepare well, there should be no recovery problem since all backups are available and current. You are carefully aligning your backup frequency to update rates using automation, thresholds, and monitoring. You are likely convinced that your backup strategy is fine-tuned.

What is wrong with this picture? You probably haven't the slightest inkling of whether the recovery duration supports the business needs. Threshold-based backups simply neglect recovery time objectives. A look at two scenarios found in most shops illustrates this best.

There is a misguided notion that recovering small objects, especially those with minimal updates, will always be fast. But this does not take into consideration situations where single updates to a page cover several archive logs. Recovering a large object with high update rates, all of which are active, can be much faster.

On the other hand, what about those large objects with high update rates that contain your most critical enterprise data? Although a well-adjusted procedure to back them up every night is fine and good, large volumes of transactions during the day may lead to the fact that the update rates are already exceeded by midday. Being forced to recover using excessive log applies leads to a significantly increased recovery time. An intelligent use of real-time statistics combined with, for example, IBM's online COPY, will insure just-in-time backups that support business availability requirements.

In terms of recovery-time objectives, consider several other factors like ZPARMs, use of multi-volume data sets, and the coupling facility.

A, B, Zs of ZPARMs: Don't depend on ZPARM defaults … or *any* defaults, for that matter. Defaults might seem to make sense for "one size fits all" shops, but even if your shop is an average fit, the recommended values change with new versions of DB2. Migrating to a new version does not apply the new defaults.

Fast Log Apply (LOGAPSTG) represents the log apply storage and it should be set to the maximum of 100MB. Although most tuning experts agree with the old maxim "never say never," there are exceptions to every rule; so never define a value of less than 100MB for LOGAPSTG. This is the default value, but it was changed only since V8. If DB2 gets less, it just takes less. Limiting this value critically restricts the available log apply storage and may slow down DB2 restart or a recovery. Assigning anything less than 100 therefore just doesn’t make any sense at all.

The check frequency parameter (CHKFREQ) determines the system checkpoint frequency in minutes or in number of log records. The default is 500,000 log records. Consider changing this to a time interval in order to maximize performance. Using log intervals can miss important checkpoints if your logging rates vary significantly. A reasonable starting point is to set CHKFREQ at 2 to 5 minutes.

If data sharing is in use, Retained Lock Wait (RETLWAIT) represents a multiplier on how long a member of a data-sharing group waits for locks on a resource held by another member that has failed. The multiplier is applied to the connection's normal time-out value and should be set to 2. The default is 0, which means the lock request is immediately rejected and the application receives a resource unavailable SQLCODE.

The OUTBUFF parameter is the size of the output buffer that is used for writing active log data sets. Believe it or not, many shops are still using the V7 default of 400, which was changed to 4000 for V8. In order to decrease the number of forced I/O operations that occur because there are no more buffers, you should select as large a size as your system can tolerate. Some popular health checks recommend a minimum value of at least 40000K in order to increase the speed that DB2 can rollback to. Reading the output buffer is always much faster than reading the active log.

These are only a few examples of relevant ZPARMS, specifically the ones that impact your recovery times. The bottom line is to know your ZPARMs well and, because no installation is static, always do periodic checks on the values you choose. The best resource for DSNZPARM information is the IBM DB2 Installation Guide (GC18-9846-03).

The coupling facility performs great. Don't be fooled! Trying to draw a correlation between how the coupling facility (CF) performs during normal operations and how it recovers is more likely to cause blindness than a certain activity in an old wives' tale, because keeping the CF in tiptop shape can be like grappling in the dark.
When a member and/or a coupling facility comes crashing down, all of the data sets are put into LPL or GRECP. To get the objects out of LPL or GRECP, you will have to build and submit lots of jobs. Each job should contain up to 99 –STA commands and up to 10 jobs can run in parallel. Why all these odd numbers? It boils down to LOGAPSTG being used as much and as efficiently as possible. Once you have built and submitted all these jobs, the coupling

facility and the SCA, which holds the LPL, starts thrashing for its very life. The following guidelines are indicative of a CF that will stand the test of a speedy recovery:

- "CF transfer time" is less than 2,000 microseconds.
- "Number of messages rejected due to lack of message buffer" is zero.
- "Sub channel busy" percentage is less than 10%.
- "False lock contention" percentage is less than 3%.
- "All path busy termination count" is zero.
- GBP "cross invalidations due to directory reclaims" is zero.
- GBP "asynchronous failed writes due to lack of storage" is zero.

The Group Buffer Pools are easy to check with a DISPLAY command. For GBP problems, the use of the "Auto Alter" in z/OS is a very good starting-point. The others, however, require RMF reports for false lock contention, and deep knowledge of system internals like CF assembler macro APIs, of which there are many. The following macros will get you started:

- IXCQUERY requests information about resources the coupling facility manages.
- IXLMG requests measurement data related to the use of a coupling facility.
- IXLZSTR retrieves control information and structure data from a dump.

If the coupling facility metrics are obscure or exceeded, you may well be heading for real trouble when you attempt a recovery.

Multi-volume data sets rule. Often, and perhaps unbeknownst to the DBA, DB2 table spaces are spreading out over an unforeseeable number of volumes. If this gets out of control, kiss your RTOs goodbye while you end up fiddling with multi-volume data sets before the real recovery activities can even start.

Quite regularly, application problems become database-administration problems, which in turn become DASD space-management problems. It's not too unusual for a DBA to knock on the door of the Space Management Department in order to ask for relief. Make sure that the "help" you get doesn't worsen your problems! Over the last decades and years, DASD space management has advanced considerably. SMS can offer so much relief that one DATA CLASS and parameters was even named after this major effect: Space Constraint Relief.

Yet space problems are space problems. If there is a permanent shortage of DASD space, even the most sophisticated settings of SMS DATA CLASS parameters are not going to help. Do pay close attention when somebody starts mumbling terms like Volume Count, Space Constraint Relief, Dynamic Volume Count, or Extent Constraint Removal. The settings for those parameters determine whether or not a data set or cluster will be multi-volume.

If you set DATACLAS to enable multi-volume allocations (SPACE CONSTRAINT RELIEF = Y *and* DYNAMIC VOLUME COUNT to more than one), and then reorganize everything, you could well end up with thousands of small, multi-volume data sets that just soak up below-the-line storage and make I/O, especially OPEN and CLOSE, slow. This is especially true of highly fragmented disks.

Only hand-picked, large VSAM clusters should be multi-volume. They must be clearly documented, and they must be part of your recovery planning and recovery procedures.

VSAM Extent Constraint Removal was introduced with z/OS 1.7. It removes the 255 extent limit (255 extents per component, or per stripe, respectively), whereas the limit of 123 extents per volume remains. The maximum number of extents is now 7257, 59 volumes with 123 extents each. You don't want that, do you?

On Logging and Auditing

Probably one of the biggest misconceptions is that logging supports auditing requirements. Transaction logs are an integral component for ensuring database integrity. Logs function to record all changes as they are made to DB2 data. DB2 will log every modification made to every piece of DB2 data. Log records are written for every modification that is successfully executed and committed (with a few exceptions, such as LOAD LOG NO). Log records are written to the active logs. There are usually two active log data sets to safeguard against physical device errors. The active logs must reside on disk (they cannot reside on tape). The active log data sets are managed by DB2 using the BSDS.

As the active logs are filled, a process called log offloading is invoked by DB2 to move the log information offline to archive log data sets. This process prohibits the active logs from filling up during DB2 processing which would stifle the DB2 environment.

The logged data can be used for rolling back transactions and recovering table spaces and indexes. Indeed, this is its primary and most important purpose – to ensure the integrity of the data in your DB2 databases. However, some organizations use the logs for database auditing.

Database auditing is the process of monitoring access to and modification of selected database objects and resources within operational databases and retaining a detailed record of the access where said record can be used to proactively trigger actions and can be retrieved and analyzed as needed. But it is a bad idea to rely on the transaction logs for your auditing.

The biggest problem when relying on logs for auditing is that reads are not recorded. Sometimes regulations and policies require auditing reads. But if you need to know who accessed a particular table and when, the log cannot answer that question.

Another problem with using logs for auditing is a lack of separation of duties. Understanding what is on the logs and which logs are needed requires DBAs. But we must also audit DBA activity. This is the type of problem that auditors like to avoid.

Perhaps the biggest problem with log-based auditing though is one of volume. Database auditing should be performed only on those database objects containing sensitive information. You do not want to audit everything. But the DB2 logs are non-discriminating; that is, *every* change to *every* database object is logged. If you rely on the logs for auditing, the resulting mountain of logs that must be kept can quickly become an administrative burden.

To do database auditing justice requires a software solution that captures database requests from the server without impacting database logs or recovery objectives.

Summary

Recognizing misconceptions requires, as a first step, recognizing them. After that, changing them is a matter of finding better ways to perform your processes. Today's business availability requirements are higher than ever and unless you've already been forced into a corner, you may have been lucky enough not to even know that some practices might not be the best ones. Lots of resources are at your disposal and tools are available, too. Although the notion has been passed down for generations, eating carrots probably won't improve your eyesight; but hopefully the contents of this article have improved your *insight*!

From zJournal, October / November 2008.

DB2PORTAL.com