The DB2 for z/OS Catalog and Directory

Is There Any Data More Important?

By Willie Favero

The heart of any DB2 subsystem or DB2 data sharing group is the DB2 catalog and directory. Yet, in spite of their importance, they're almost invisible to many, and to others are often considered just another set of tables. The catalog and directory were seldom talked about before DB2 Version 8. Everyone knows they're there and assumes they always will be. They also assume they will always work. Without these tables, however, you aren't going to get far if you try to do anything with DB2 for z/OS. However, what happens if you should find yourself without them because of some data mishap or worse? How can you get them back?

It doesn't matter whether you're dealing with a localized disaster or a full-blown, move to another site-type of catastrophe. At some point, you're going to have to get the DB2 catalog and directory back online and functioning before you can do anything else with DB2. This article takes a look at catalog and directory tables, why they're important, and how you can ensure they will always be there if or when you need them (or at least how you can get them back if they disappear).

Before discussing what's involved to recover the catalog and directory, let's consider what makes up the objects we call DB2's catalog and directory. The DB2 catalog is in database DSNDB06 while the directory lives in DSNDB01. The catalog has literally doubled in size during its lifetime, going from 11 tablespaces in DB2 V1 to a whopping 22 tablespaces in V8. The catalog could possibly grow as large as 32 tablespaces by the time DB2 9 becomes available. The directory, however, even though it's of equal or greater importance, is considerably smaller. It currently consists of only five tablespaces and really hasn't changed much in size (number of objects) over the years. One tablespace was added back in V2.3 and three have been renamed since DB2's inception.

The catalog consists of 82 tables in V8 (80 in V7) that externally have similar characteristics to other tables in DB2. SQL can be run against the catalog to query its contents and even change the data in a few columns in some tables. As of V8, one noticeable difference between user tables and the tables in the catalog is that most of the catalog tables are now encoded using Unicode. However, that should have little, if any, impact on backup and

recovery. The directory is quite different from the catalog. SQL can't be used against any of the directory tables. In fact, in most cases, you probably shouldn't be poking around inside the directory for any reason.

Another thought: Are your data sets managed in SMS? This is a requirement if you want to take advantage of the DB2 V8 features BACKUP SYSTEM and RESTORE SYSTEM. When using SMS, consider how you set up your storage groups. The catalog and directory should be in its own storage group, with the Boot Strap Data Set (BSDS) and active logs in another, all separated from your user data that's in its own storage group.

SYSCOPY is usually the source of information about your recovery assets, but for the DB2 catalog and directory, not all the information you might need can be found there. SYSCOPY doesn't contain details about SYSUTILX, DBD01, SYSLGRNX, and SYSCOPY itself; this information is written to the DB2 logs.

Preparing Your Backups

All the copy methods available in DB2 will work on the catalog and directory. Three specific catalog/directory tablespaces require special handling and you should manage them cautiously as far as the COPY utility is concerned:

- DSNDB06.SYSCOPY
- DSNDB01.DBD01
- DSNDB01.SYSUTILX.

First, only full image copies are allowed. Although incremental image copies are allowed against all other catalog and directory tablespaces, if it's attempted for DSNDB06.SYSCOPY, DSNDB01.DBD01, or DSNDB01.SYSUTILX, DB2 will change the copy to a full image copy. These three tablespaces can't be used together in a list. DSNDB06.SYSCOPY, DSNDB01.DBD01, and DSNDB01.SYSUTILX must each be on its own copy statement. Also, DSNDB01.SYSUTILX must be in its own copy job step. If you specify SHRLEVEL(REFERENCE) when copying SYSUTILX, this copy must be the only copy job running on the Sysplex. The indexes on DSNDB06.SYSCOPY and DSNDB01.SYSUTILX, if defined with the COPY YES attribute, are an exception; they can be copied along with their tablespace.

SHRLEVEL(CHANGE), another popular copy option, is allowed on all the catalog and directory tablespaces. You may want to consider using it when there are concerns about concurrently running copies. The copy utility uses DSNDB06.SYSCOPY, DSNDB01.DBD01, and DSNDB01.SYSUTILX, so be careful to avoid copying these tablespaces while executing COPY for other objects, both user and DB2. If you must copy these three tablespaces simultaneously as copies for other tablespaces, consider using the SHRLEVEL(CHANGE) COPY option. This will minimize contention, although you must use the log along with the image copy and you won't be able to perform a TOCOPY recovery.

LISTDEF is another utility function to be careful with if used in conjunction with the catalog and directory. DSNDB01 and DSNDB06 can't be specified on the DATABASE option, and pattern matching is prohibited on the TABLESPACE and INDEXSPACE options if a catalog or directory tablespace is specified. Fully qualified

catalog and directory names are allowed only on the TABLE and INDEX options. The TABLE and INDEX options also don't allow pattern matching for catalog and directory objects. Also, no SYSUTILX related object can be specified in a LISTDEF statement.

Finally, a record of all recovery resources (i.e., image copies, quiesce points, etc.) for user, catalog, and directory objects is recorded in the catalog table SYSIBM.SYSCOPY—with the exception of the three tablespaces just mentioned. The catalog tablespace DSNDB06.SYSCOPY and the directory tablespaces DSNDB01.DBD01 and DSNDB01.SYSUTILX have their copy information recorded on the DB2 log, not in SYSCOPY. The information necessary to recover these three objects is still required even when the catalog isn't available. If there's no catalog available, no access to SYSCOPY and no way to determine what resources are available to recover these objects, a recovery must still be performed. The log, however, should always be available. Without the log, you can't restart DB2 or, in most cases, recover an object. Recording the copy information for DSNDB06.SYSCOPY, DSNDB01.DBD01, and DSNDB01.SYSUTILX on the log ensures that it's always accessible to the RECOVER utility, regardless of the state of the catalog.

If you have the batch window to perform a quiesce of the catalog and directory, it's a great way to set up DB2 for a traditional point-in-time recovery. There are, however, precautions you must take. If you quiesce SYSUTILX, it must be in its own separate job step. If you're running QUIESCE specifically to prepare for a potential point-in-time recovery, you must quiesce DSNDB06.SYSCOPY after quiesces for all the other catalog and directory tablespaces have completed.

Most of the processes available today to minimize the time to recover from a disaster and recover your data to a point as close to current as possible also apply to the catalog and directory. No matter what method you use to create a backup of the catalog and directory, you must have a process in place and tested. Again, the catalog and directory are the most important pieces of information in your entire DB2, so treat them appropriately. Recovering the catalog and directory to a current point in time or to a prior point in time could be interesting if you lack any backups of the catalog or directory. Even DB2 needs something to start with.

Performing a Recovery

A recommendation found in the DB2 documentation states that you should avoid recovering the catalog and directory to a prior point in time. Although this is an excellent recommendation, it's often hard to follow if you find yourself offsite recovering from a disaster. In the case of a disaster, a prior point in time is all you may have. So it's important that you be familiar with the correct procedure to follow if you need to recover your catalog and directory to a prior point in time.

One of the reasons it's not advisable to recover the catalog to a prior point in time is the relationship between objects; there are just so many relationships that exist in the catalog and directory, both Referential Integrity (RI) and other. Databases, along with all the tablespaces, tables, indexes, etc. that are created in them, become Database Descriptors (DBDs). The DBD is in DSNDB01.DBD01 while most of the other definitions are in DSNDB06.SYSDBASE. Plan and package detail can be found in DSNDB06.SYSPLAN and DSNDB06.SYSPKAGE while the actual plans and packages reside in DSNDB01.SCT01 and DSNDB01.SPT01. The recovery process requires access to certain information that must be present before you can run the RECOVER utility. So when attempting a point-in-time recovery, be aware of your recovery order.

Another important relationship is the one between the user data and objects that contain it. If you recover the catalog and directory to a prior point in time, you'll definitely lose some information; maybe not always user data, but certain things will disappear. If you recover the catalog and directory to a week ago, the descriptions for anything created or altered will be lost, as well as information about binds and the packages and plans created, stored procedure entries, etc. For example, take something as simple as an ALTER ADD of a new column. Also assume you have processed data to this table since the new column was added. If you recover the catalog and directory to a time prior to the add column, then that column no longer exists in that table. What happens to the data in that column? It's still in the VSAM data set. Something has to happen now to make everything match up again.

If you're recovering the catalog to a prior point in time, you must recover the catalog tablespaces in the order listed in the *DB2 Utility Guide and Reference* (SC18-7427). You must follow the recovery sequence for the catalog and directory. Consider treating your entire DB2 (catalog, directory, and all user data) as one giant chunk that must be recovered together to the same point in time. With a point-in-time recovery, it's also best to bring down DB2 cleanly and restart DB2, specifying ACCESS(MAINT) at DB2 start-up. This will prevent anyone else from getting into DB2 while the recovery occurs.

The first tablespace to recover is DSNDB01.SYSUTILX. When the utility starts, it's registered in SYSUTILX and the row is updated as the utility does its work. SYSUTILX also contains necessary information should a utility need to be restarted. If SYSUTLX doesn't exist, DB2 doesn't let any utility run except for the RECOVER utility for SYSUTILX. SYSUTILX isn't referenced when it's being recovered. SYSUTILX must be in its own recovery step; no other utility process can run while it's being recovered because all other utility processes use SYSUTILX. A subsequent recovery can't run until the step recovering SYSUTILX completes.

Once SYSUTILX has been successfully recovered, all indexes on the tables in SYSUTILX can be recovered or rebuilt. If the indexes have the COPY YES attribute on, they should be recovered. If not, they should be rebuilt. Either process is acceptable.

The tablespace name is one of several things about SYSUTILX that might be important in a recovery. A few versions ago, DSNDB01.SYSUTIL was renamed to DSNDB01.SYSUTILX, but one of the tables in this tablespace retained its name SYSUTIL. Don't get tablespace and table names confused and try to clean up old objects. Another misconception is that SYSUTILX doesn't have to be recovered and can simply be re-created. Don't assume that this directory tablespace should be empty or that its contents are no longer of any use; that could be the furthest thing from the truth. You recover SYSUTILX so you can determine what utilities were running at the time of the disaster in case other actions need to be performed.

The next object that needs to be recovered, following recovery of the

SYSUTILX and its indexes, is DSNDB01.DBD01. DBD01 contains the descriptors, or DBDs, for all remaining objects (catalog, directory, workfiles, and user data) you need to recover. The RECOVER utility will use these DBDs to recover the remaining objects. Recovering DBD01 also must be in its own job step and must complete before any other recovery task executes. There are no indexes on DBD01, so once this recovery step completes, you can move on to the next recovery.

Now that you can manage the recovery jobs (SYSUTILX recovered) and you have all the DBDs (DBD01 recovered) for objects needing recovery, you need to recover the object that contains the list of all the recovery resources needed to recover the rest of the catalog, DSNDB06.SYSCOPY. SYSCOPY has the list of image copies and possible quiesce points available. When recovery for SYSCOPY completes, the indexes on SYSCOPY should be rebuilt. The indexes also can be recovered if COPY YES is specified for the index and a full copy is available. When rebuilding the indexes on the catalog, the DB2-defined indexes should be rebuilt first. You can do this with the INDEX(ALL) control card on the REBUILD utility. If non-DB2 indexes are defined (that's user-defined indexes), then the DB2 indexes should be rebuilt separately followed by rebuilds for the user-defined indexes. This is true for all the recovery steps where indexes exist on the catalog or directory tables.

You now have utility control, DBDs, and an inventory of recovery resources in place. Now the log ranges can be restored. DSNDB01.SYLGRNX is the next object to recover. Again, once the tablespace recovery completes, rebuild the indexes on the SYSLGRNX table. SYSLGRNX has a row containing the start and end Relative Byte Address (RBA) for the open and close of every page set that can be recovered. That includes tablespaces and indexes with COPY YES. In a data-sharing environment, the row contains the start and end Log Record Sequence Number (LRSN). Having these ranges available means the RECOVER utility can read only the log records pertaining to that page set. DSNDB01.SYSUTILX, DSNDB01.DBD01, DSNDB01.SYSLGRNX, DSNDB06.SYSCOPY, DSNDB06.SYSGROUP, DSNDB01.SCT02, and DSNDB01.SPT01 and their associated indexes, even if defined with COPY YES, don't record any information in SYSLGRNX. These objects are assumed open since the last image copy and all log records are read from that point.

The next tablespace in the recovery sequence is DSNDB06.SYSALTER followed by its indexes. This tablespace contains the catalog table, SYSOBDS, and the information about page sets (old DBDs) that might be needed for a point-in-time recovery. It has a row for every page set that can be recovered to an image copy made before the first version of the page set was generated. This tablespace isn't used until DB2 V8 Enabling New Function Mode (ENFM).

DSNDB06.SYSDBAUT, DSNDB06.SYSUSER, and DSNDB06.SYSDBASE are the next tablespaces to recover and must be recovered in the order listed. As each tablespace is recovered, you also recover or rebuild the appropriate indexes for the tablespaces.

The remaining catalog and directory tablespaces can now be recovered, along with their indexes. For the catalog, this will include SYSGROUP, SYSGPAUT, SYSOBJ, SYSPLAN, SYSPKAGE, SYSSEQ, SYSSEQ2, SYSSTATS, SYSSTR, SYSVIEWS, SYSDDF, SYSHIST, SYSGRTNS, SYSJAVA, SYSJAUXA, SYSJAUXB, and SYSEBCDC. The directory objects that still need to be recovered are DSNDB01.SCT02 and DSNDB01.SPT01 plus their indexes. A list of all catalog indexes is included in Appendix F of the *DB2 Version 8 SQL Reference* (SC18-7426) and the directory indexes can be found in Chapter 26 of the *DB2 Version 8 Diagnosis Guide and Reference* (LY37-3201).

There are a few additional tablespaces being added in DB2 9 with some new relationships that need to be retained. When DB2 9 is available, make sure you re-address your DR and local recovery plans for the catalog and directory to ensure all the newest objects are included. Also be aware of new restrictions that might be introduced with some of the new objects.

When you recover DSNDB06.SYSCOPY, DSNDB01.SYSLGRNX, DSNDB06.SYSDBAUT, DSNDB06.SYSUSER, and DSNDB06.SYSDBASE, no other utilities can be running. However, other utility statements can be coded in the same job step. At this point, the formal catalog and directory tablespaces, along with all DB2-defined and user-defined indexes should be recovered. Any other objects considered part of DB2 but not part of the catalog and directory (such as the objects used by real-time statistics and the resource limit facility) should now be recovered followed by all of the user tablespaces.

The recovery information for DSNDB01.SYSUTILX, DSNDB01.DBD01, and DSNDB06.SYSCOPY isn't contained in the SYSCOPY catalog table; it's stored on the DB2 log. Why? When we recovered DBD01, for example, did SYSCOPY even exist yet? No. So there would have been no way to determine what recovery resources were available from SYSCOPY when there wasn't a SYSCOPY.

What about the inventory of recovery resources necessary to recover SYSCOPY? How could that kind of information possibly be stored in SYSCOPY? The solution is to store the image copy and quiesce information on the DB2 log. The log must be available during a recovery anyway, so why not store the image copy information there so it's always available when DSNDB06.SYSCOPY isn't? Even though this information is stored on the log, you can still use the REPORT RECOVERY utility to obtain SYSCOPY details for DSNDB01.SYSUTILX, DSNDB01.DBD01, and DSNDB06.SYSCOPY. REPORT RECOVERY should be a task in both your backup and recovery procedures. After completing a backup of the entire catalog and directory, running REPORT RECOVERY will yield a list of recovery resources, the recovery history from the SYSIBM.SYSCOPY, log ranges from the SYSIBM.SYSLGRNX, and volume serial numbers for the archive log data sets from the BSDS that you can use as a record to verify the recoverability of the catalog and directory.

Everything up until now has examined recovering the catalog and directory. However, to run RECOVER, the log information and the BSDS must be accurate. If it's necessary to recover any logs or repair the BSDS, make sure this occurs before you attempt to recover the catalog and directory.

When recovering the catalog and directory at a DR site, you should consider checking it out before you let anyone use the system to ensure all the pieces were recovered correctly. A good set of SQL with expected results works well as a starter for checking out the catalog. If you don't have your own set of queries, there are some starter queries in member DSNTESQ in DB2's sample library hlq.SDSNSAMP. However, this article briefly discusses

performing a catalog health check. The procedures discussed should be applied to any catalog after a point-in-time recovery.

What Privileges Are Necessary?

Secondary authorization identifiers (authids) should be strongly considered for SYSADM, Installation SYSADM, and Installation SYSOPR access. These authorizations should be tightly controlled because of their capabilities. At a minimum, the install authids defined in DSNZPARM should use secondary authids. Also, consider setting up special catalog recovery secondary authids with DBRECOVER, DBADM, or DBCTRL. Granting DBRECOVER, DBADM, or DBCTRL on DSNDB06 to a catalog recovery secondary authid would give that authid all the privileges needed to manage the backup and recovery processes for the catalog. Granting these authorization sets on DSNDB06 gives that authid the same privileges on DSNDB01. You can't grant a privilege set directly on DSNDB01.

If you're recovering your entire catalog and directory, start your DB2 subsystem specifying ACCESS(MAINT). In this mode, only Install SYSADM or Install SYSOPR are allowed access to the catalog. This prevents unintentional access to the catalog while it's being recovered. For those in a data-sharing environment, remember that ACCESS(MAINT) affects only the DB2 member started in MAINT mode; other members of the group are unaffected, although they all have access to the same single catalog.

While attempting to recover a catalog or directory page set, you could receive the message DSNT500I RESOURCE UNAVAILABLE. If you have the correct set of privileges, then DSNDB06.SYSDBASE or DSNDB06.SYSUSER is probably not available. If you see this message, you should ensure the two tablespaces are available. If they aren't, you can start them or run the recovery using the Install SYSADM or Install SYSOPR authids.

Prevention

As Henry de Bracton, the writer on English law, stated, "An ounce of prevention is worth a pound of cure." This applies to the catalog and directory, too. There are some basic health check tasks you can perform periodically (besides making regularly scheduled backups) that will help ensure the catalog and directory remain in good working form. The golden oldie of them all is validating that the internal links used by the tablespaces SYSDBASE, SYSDBAUT, SYSGROUP, SYSPLAN, and SYSVIEWS in DSNDB06 and tablespace DBD01 in DSNDB01 are in good working order. These tablespaces use a series of forward and backward pointers, not accessible via SQL, to link certain elements in the aforementioned catalog and directory tablespaces. For example, in SYSDBASE, SYSTABLES has pointers from a table definition to that table's first column, the next table definition, and to the tablespace where it belongs. SYSCOLUMNS has pointers from column to column within a table and to the table that owns the columns.

For more on these internal links and pointers, see the Catalog Formats chapter of the *DB2 for z/OS Version 8 Diagnostics Guide and Reference* (LY37-3201). These links can be checked periodically by running the DB2 stand-alone utility, DSN1CHKR, which is straightforward to run, although its output can sometimes be challenging to read if a problem is discovered. Fixing a found problem isn't something that should be attempted by the faint of heart or those less skilled. The major rule to remember when executing DSN1CHKR is to never run it against an active catalog or directory tablespace.

CHKR reads only the underlying VSAM data set. If changed pages are still in the buffers and haven't been externalized to disk, CHKR will think there are broken links because it can't check the pages in the buffers. The best practice is to copy the catalog/directory tablespaces (after using the STOP DATABASE SPACENAM command to stop them) to another location using DSN1COPY. Then run CHKR against the copies. This lets you restart the catalog and directory tablespace as soon as the copy completes. Also, when running DSN1COPY to make a copy, always specify the check option to validate page integrity.

DSN1COPY is another DB2 stand-alone utility with several uses. For now, we're concerned only with its ability to check the validity of a data page when the CHECK option is specified on the EXEC statement. It performs this process one page at a time and reports on any page errors. If a copy of the catalog or directory isn't being made, DSN1COPY's output DD statement, SYSUT2, can be set to DUMMY. This lets pages be checked without the need to write (copy) the pages to another page set. This practice of running DSN1COPY with the CHECK option should be done periodically for all critical page sets.

Running the DB2 CHECK DATA and CHECK INDEX utilities against the catalog is another preventive maintenance task you should complete. Once DB2 V2.1 was released, links were no longer used by new tablespaces added to DB2. DB2 used the RI feature. Because some of the catalog tablespaces use RI, the CHECK DATA utility should be run against the appropriate catalog tablespace. In addition, the DB2 CHECK INDEX utility should be executed against all the catalog and directory tablespaces to check all the indexes DB2 used. V8 has more than 130 indexes and 100 constraints defined on the catalog and directory tables, so this is neither trivial in size nor importance.

Conclusion

The catalog and directory are vital to DB2's existence. They need to be treated as equally or more important than your most critical production user tablespace. Ensure you're monitoring their health, you have a solid backup strategy, and you know how to recover one or more of their tablespaces should the need arise. To learn more, refer to:

- *DB2 Version 8 Utility Guide and Reference* (SC18-7427)
- *DB2 Version 8 Diagnostics Guide and Reference* (LY37-3201)
- *DB2 Administration Guide* (SC18-7413).

Keep these handy and make sure you always have the most current copies of the IBM materials, which can downloaded at www.ibm.com/software/data/db2/zos/v8books.html. Finally, don't forget to check out my blog occasionally at http://blogs.ittoolbox.com/database/db2zos for the latest information on DB2 for z/OS.

About the Author

In the past 29 years, Willie Favero has been a customer, worked for IBM, worked for a vendor, and is now an IBM employee again. He has always worked with databases and has more than 20 years of DB2 experience. A well-known, frequent speaker at conferences and author of numerous articles on DB2, he's currently with North American Lab Services for DB2 for z/OS, part of IBM's Software Group.
Email: wfavero@attlgobal.net