BY CRAIG S. MULLINS

Like death and taxes, the prospect of disaster striking your DB2 shop can seem inevitable--especially if you have no disaster recovery plan at the ready. Here are some tips and techniques for creating such a plan

Mastering Disaster

PLANNING FOR DISASTER recovery is a complex procedure in the best of situations. Unfortunately, the best of situations does not exist within a DB2 environment. DB2 has limitations during disaster recovery that require specific solutions. This article will offer techniques, requirements, and cautions that will help you put together a disaster recovery plan for your DB2 applications.

DISASTER STRIKES

The situation is grim. Your data processing shop has been struck by a devastating fire. All of the hardware, software, and data at your site have been destroyed. Are you adequately prepared to recover your DB2 data at a remote processing site?

DB2 disaster recovery happens in two steps: recovery of the DB2 subsystem and recovery of the application data. The primary concern of the DBA should be the recovery of the operational data. To accomplish this, however, you need to recover your DB2 susbsystem first. Therefore, your initial concern should be the development of a comprehensive subsystem recovery plan. We will focus on the recovery of DB2 application data rather than recovery of the subsystem and its related data.

DB2 RECOVERY BASICS

The standard tools you use for DB2 recovery are the image copy backup, the DB2 log tapes, and internal DB2 tables and datasets. Figure 1 shows the flow of normal DB2 recovery. The standard unit of recovery for DB2 is the tablespace. The DB2 copy utility creates an image copy backup of the tablespace dataset(s). All image copy dataset information is recorded in the SYSIBM.SYSCOPY table, which is in the DB2 catalog. It's not necessary to keep track of the image copy datasets externally, since DB2 manages this information independent of the application code.

DB2 also keeps a log of all changes made to tablespaces. With a few exceptions, all updates are recorded in the DB2 active log. When an active log is full, DB2 creates an archive log. Many archive logs are created during normal DB2 application processing. This information is stored in the SYSIBM.SYSLGRNG table in the DB2 Directory and the Boot Strap Dataset (BSDS). (See the sidebar, "Tables and Datasets Internal to DB2," for a more in-depth description of these features.)

The recover utility is invoked to restore the tablespace data. DB2 uses all of the information it stores in active and archive logs, the DB2 Catalog, the DB2 Directory, and the BSDS to recover tablespace data with a minimum of user input. The only input the recover utility requires is the name of the tablespace to be recovered. DB2 does the rest. The less user input required in a recovery situation, the less chance there is of someone making a manual error during the confusion of a disaster. Unfortunately, the automation of the recovery process is just the sort of circumstance that can complicate off-site DB2 disaster recovery planning.

This introduction to DB2 recovery only skims the surface of tools available for DB2 recovery planning. The illustration in Figure 2 gives a short description of all the DB2 recovery utilities.

RECOVERY REQUIREMENTS

Your corporation should have a disaster recovery plan. One specific piece of that plan must deal with the recovery of DB2 data. But what are its overall intentions?

Most disaster recovery plans have four goals:

□ To avoid data loss

□ To avoid the reprocessing of transactions

□ To avoid causing inconsistent data

□ To limit the time needed to restart critical application processing.

However, these goals often conflict with one another. For example, how can critical applications be online quickly when they usually consist of huge databases? How can you avoid the loss of data when thousands of transactions update DB2 tables every second? You have to make trade-offs.

When developing a recovery plan, consider your goals. Business needs are the motivating force behind your planning. It's prudent, therefore, to separate your systems into critical and noncritical applications based on business needs. The definition of a critical system must be made by the people responsible for the business function that system supports.

Develop recovery plans for the critical applications first. These support the functions that are absolutely necessary should your company experience a disaster.

Once you've targeted specific applications for disaster planning, you must then decide on a strategy. Here are three possible strategies for DB2 disaster recovery planning. Each one has its strengths and weaknesses. You can choose one strategy, or mix and match them based on the recovery requirements of each application.

Recovery

ARTWORK: PETER M. MILLER

THE SLEDGEHAMMER

I call this first strategy the sledgehammer, because it's a very basic approach to application backup and recovery. However, it's workable and simple to implement. This strategy should be considered for applications that are noncritical, nonvolatile, and don't run 24 hours a day. It consists of the following steps:

1. Stop the DB2 subsystem to ensure stable application data. This establishes a systemwide point of consistency.

2. Copy all tablespaces using a utility to dump complete DASD volumes. Utilities such as FDR from Innovation Data Processing and DFDSS from IBM work nicely.

3. When all DASD volumes containing DB2 data have been successfully copied, restart the DB2 subsystem.

4. Copy the backup tapes and send them off-site.

5. Recovery at the remote site is then done a complete DASD volume at a time.

However, there are a number of problems inherent in this strategy. For example, many shops require DB2 to be available 24 hours a day. In this situation, completely stopping the DB2 subsystem is not a viable option. Instead, each application could have a regularly scheduled job to stop the application only. The job would need to quiesce—or inactivate—the application tablespaces, the DB2 Catalog (DSNDB01), and the DB2 Directory (DSNDB06), and then stop each application tablespace. (Note: Only an Install System Administrator (SYSADM) can quiesce the DB2 Catalog and DB2 Directory.) You could do the complete volume backup at this point, then restart the application tablespaces.

The sledgehammer approach is effective for those shops willing to trade 24-hour processing capabilities for ease of disaster recovery preparation. Still, this strategy isn't the best solution for most DB2 installations, because most shops are unwilling to make this trade-off. Shutting down DB2 effectively stops the execution of every application that uses DB2 tables. This is usually not possible. Even running the quiesce utility as directed above affects other applications by forcing a point of consistency on the DB2 Catalog and Directory. If you wish to avoid these points of contention, choose another strategy.

THE SCALPEL

This second method uses native DB2 functionality to prepare for disaster recovery. I call this method the scalpel, because it's precise and accurate, much like a surgical instrument.

This approach involves the following steps:

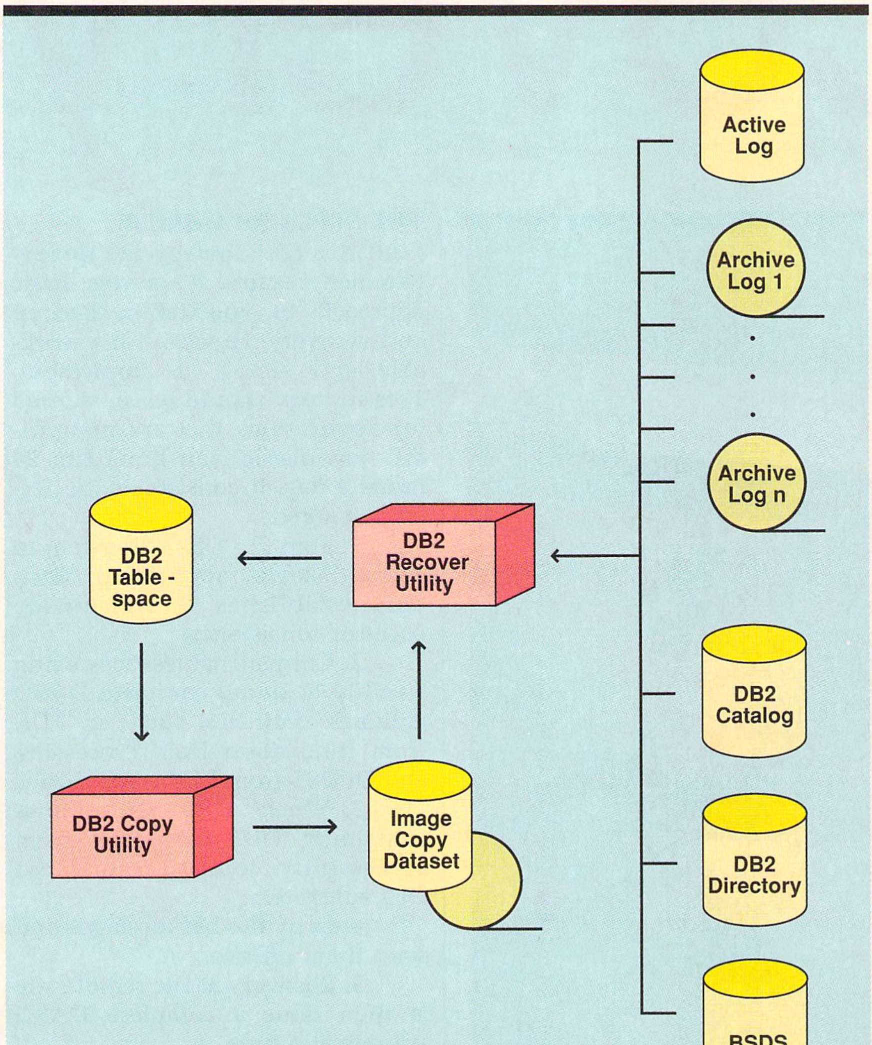

FIGURE 1. *Normal DB2 recovery.*

1. Always produce two image copy backups, at least one of which must be on tape. (This is not as easy as it sounds.)

2. Send the tape image copy backup to the remote site. You should do this as soon as possible after the tape has been created to avoid the chance that the tape could be damaged in a subsequent disaster.

3. Do not backup indexes.

4. Produce a QMF report from the SYSIBM.SYSCOPY table daily, and send a copy of the report to the remote site. (See Figure 3 for a query that will accomplish this.)

5. Produce a BSDS Log Map report using DSNJU004, and send a copy of the report to the remote site.

6. Recovery at the remote site is done a tablespace at a time. Use RECOVER INDEX to rebuild all indexes. Run CHECK DATA to resolve any referential constraint violations.

The scalpel method differs from the sledgehammer in many ways, but perhaps the most important is its reliance on DB2. This means that only application data that's recorded in the DB2 Catalog, DB2 Directory, and BSDS can be recovered. So this approach relies heavily on the ability to recover the DB2 subsystem. Application data will only be as current as the last backup of the DB2 subsystem. This limitation is one of the headaches caused by automation of the DB2 recovery process.

Consider, for example, an application that sends three image copy backups to a remote site daily. One backup is sent off-site in the morning to allow for postbatch recovery, another is sent off-site in the afternoon to allow recovery of all morning transactions, and a third is sent in the evening to allow recovery of all prebatch transactions. However, if only one DB2 Catalog copy is sent off-site daily, say after the morning copy but before the afternoon copy, then remote recovery can proceed only as far as the morning copy plus any additional archive logs that were sent off-site.

For this reason, you should always try to synchronize your application image copies with your

ARCHIVE LOG	Forces an archival of the DB2 log (DB2 v. 2.3 only)
CHECK	Checks the consistency of data for referential constraint violations and checks indexes for consistency with the underlying table data
COPY	Creates an image copy of a tablespace or a dataset of a partitioned tablespace
MERGECOPY	Merges incremental and full tablespace image copies into a new full or incremental image copy
MODIFY	Deletes rows from the SYSIBM.SYSCOPY catalog table and the SYSIBM.SYSLGRNG directory table
QUIESCE	Establishes a point of consistency for a tablespace set
RECOVER	Recovers data (at the page, partition, index, dataset, or tablespace level) to its current state or to a previous point in time
REPAIR	Has many uses but within the context of recovery is used to reset COPY PENDING and RECOVER PENDING status for tablespaces and indexes
REPORT	Reports recovery information from the DB2 Catalog, DB2 Directory, and BSDS
DSNJU003	Changes the BSDS (Change Log Inventory utility)
DSNJU004	Prints a report of information from the BSDS (Print Log Map utility)
DSN1CHKR	Verifies the integrity of the DB2 Catalog and DB2 Directory tables
DSN1COPY	Copies tablespace and index datasets behind the back of DB2
DSN1LOGP	Produces a report from a DB2 log tape (cannot access a log while a DB2 subsystem has it allocated)
DSN1PRNT	Dumps data from a DB2 tablespace, index, image copy, or DSN1COPY dataset

Figure 2. *DB2 recovery utilities.*

DB2 Catalog backups. If you don't, you'll end up with increased data loss and unusable image copies at your remote site. The amount of data lost in an off-site recovery depends on the timeliness of the backup of archive logs and how well the DB2 Catalog backup is synchronized with the application tablespace backups.

Also, you should always keep at least three image copy backup tapes at your remote site. This will provide a sufficient number of backups should one or more of your tapes be damaged. DB2 will automatically fall back to previous image copy backups in the event of a damaged tape. Changes will be applied from the archive logs to recreate the data lost by falling back to the previous image copy.

You should also note that any updates recorded on the DB2 active logs at the time of the disaster will be lost. Recovery can only be done through the last archive log that's available at the remote site.

The final point to consider with the scalpel method concerns the creation of the underlying tablespace and indexspace datasets. If you are using native VSAM, you must use AMS to create the datasets before recovering each tablespace and its related indexes. If you're using STOGROUPs for your production datasets, simply ensure that the STOGROUPs have been altered to point to valid DASD volumes at the remote site. The recover utility will create the underlying VSAM datasets for you.

DSN1COPY

This third strategy is not recommended because it operates "behind the back" of DB2 and therefore sacrifices the rigorous control provided by DB2 backup and recovery procedures. However, at times you may want to use this approach for a limited number of noncritical applications.

This strategy is close to the sledgehammer approach, but it's a little more complicated. Follow these steps for each DSN1COPY that must be executed:

□ In each tablespace set, quiesce all of the tablespaces that will be backed up using the DSN1COPY utility.

□ In each tablespace set, stop all of the tablespaces that will be backed up using DSN1COPY.

□ Execute the DSN1COPY utility for each tablespace being copied.

□ In each tablespace set, start all of the tablespaces that will be backed up using DSN1COPY.

Recovery at the remote site must be done using DSN1COPY, because these backup datasets are not recorded in the DB2 Catalog. Therefore, each tablespace and indexspace dataset must be created using AMS before the DSN1COPY can be executed to restore the application data.

This is a complex and potentially error-prone process that should be avoided. If your application data is very stable, however, you may want to avoid recording backups in the DB2 Catalog. This could simplify your Catalog maintenance procedures. You should execute the modify utility to clean up the SYSIBM.SYSCOPY table and the SYSIBM.SYSLGRNG table periodically. Modify is run specifying a tablespace and a date range that will delete all image copy and log information for that date range's tablespace. Each application must supply the appropriate date range for image copy deletion.

If your date range is unknown, unstable, or random, you may wish to avoid using the DB2 Catalog for recovery altogether. You could simply create four DSN1COPY backups every time your (stable) application data changes. Retaining two of them on-site and sending two off-site should suffice. Remember, this method should only be used for stable data and is not generally recommended. The proper method is to use the DB2 copy and recover utilities and to execute the modify utility on a tablespace-by-tablespace basis for each application.

NONCRITICAL APPLICATIONS

You should be concerned with recovering noncritical applications only after you have implemented complete disaster recovery procedures for the critical ones. If you follow the same procedures outlined earlier, you'll have an exemplary disaster recovery plan for all of your applications. Sometimes, though, simple DSN1COPY datasets for each tablespace in the noncritical application will suffice for off-site recovery purposes. These datasets should be taken when DB2 is not operational (or when the application has been stopped). Because the application is noncritical, the DSN1COPY may need to be done less frequently. You'll have to decide this on an application-by-application basis.

For some noncritical applications the decision may be made that no disaster recovery procedures will be developed. This is a valid decision only if the company can afford to lose the system entirely—a rare circumstance indeed.

DB2 ENVIRONMENT

Sometimes, recovery is targeted to be done at an alternate site that's already running DB2. This is not advisable. In the event of a disaster, your whole machine could be lost. This means that not only DB2, but MVS, JES, TSO, and all of the other system software will need to be recovered. Your disaster recovery plan will become needlessly complex if you plan to recover to an existing system. Reconfiguring software that is already operational is usually more difficult than bringing everything up from scratch.

```
QMF Query

 SELECT      DBNAME, TSNAME, DSNUM, TIMESTAMP, ICTYPE,
             DSNAME, FILESEQNO, SHRLEVEL, DSVOLSER
 FROM        SYSIBM.SYSCOPY
 ORDER BY    DBNAME, TSNAME, DSNUM, TIMESTAMP


QMF Form

Total Width of Report Columns: 150

NUM    COLUMN HEADING     USAGE     INDENT     WIDTH      EDIT      SEQ

 1     DATABASE           BREAK1    1            8        C         1
 2     TABLESPACE         BREAK2    1            8        C         2
 3     DS_NUM             BREAK3    1            3        L         3
 4     TIMESTAMP                    1           26       TSI        4
 5     IC_TYPE                      1            4        C         5
 6     DATASET NAME                 1           44        C         6
 7     FIL_SEQ_NO                   1            3        L         7
 8     SHR_LVL                      1            3        C         8
 9     VOL SERIAL LIST              1           42        C         9
```

Figure 3. *SYSIBM.SYSCOPY image copy report.*

If you insist on recovering to a DB2 subsystem that already exists, remember the following: All databases, tablespaces, tables, and indexes would need to be created at the remote site. This could be done either at the time of the disaster, which would be a complex and error-prone procedure, or prior to the disaster, which would be easy but resource-consuming. At any rate, all DB2 objects must exist before the image copy datasets can be restored. This can be accomplished only by using the DSN1COPY service aid program with the OBIDXLAT option.

You should maintain a comprehensive report that lists the DBID for each database, the PSID for each tablespace, and the OBID for each table in both DB2 subsystems. (See Figure 4 for a QMF query that will produce this report.) DBIDs, PSIDs, and OBIDs are identifiers that are stored in the DB2 Catalog to identify each object to DB2. This list can then be used to execute the DSN1COPY service aid program using the OBIDXLAT option. *This is the only way to accomplish recovery to a different DB2 subsystem.*

You must also consider dataset mangement techniques. If you allocate VSAM datasets for all of your production tablespaces and indexes, then you must use AMS to create the underlying datasets prior to recovery at the remote site. If you use STOGROUPs, though, the datasets will already have been allocated when the tablespaces and indexes were created.

DB2 HURDLES

You must address many potential problems when developing a DB2 disaster recovery plan. Of these, DB2's inability to produce two duplicate backup image copy datasets is the most frustrating. DB2 allows backup datasets to be taken only one at a time. What then should be done to produce the backup dataset that's to be sent to the remote site?

Remember, tablespace image copy information is recorded in the DB2 Catalog when the copy utility is executed. What happens if a second copy utility is executed directly following the first copy? Well, another entry is made in the DB2 catalog. If the second copy is then sent off-site for disaster recovery purposes, how will this affect normal, on-site recovery?

When the recover utility is invoked, DB2 will issue a mount request for the tape that was sent off-site. The operator would have to reply that the tape could not be mounted. If the tape isn't mounted, then the DB2 recovery will fail. If it is mounted, then the recovery will continue, finding the previous image copy dataset that is still on-site.

This approach is not recommended. First, if two DB2 copies are run, you should always send the first one off-site. This will reduce the necessity for manual intervention at the local site. Manual intervention will be required only when the second image copy dataset is damaged. Furthermore, you can avoid making two DB2 copies altogether by taking one image copy and then running IEBGENER to produce a second copy. (IEBGENER is an IBM utility program that copies datasets.)

If the image copy dataset is not cataloged, then DB2 will store the volume serial number in the SYSCOPY table; if it is cataloged, it won't store it. Knowing this, a strategy can be developed for dealing with the IEBGENER copy that has been sent off-site. Catalog all image copy datasets. Use IEBGENER to make an uncataloged copy using a different volume serial number. Catalog the tape at the remote site using the same volume serial number as was supplied to the IEBGENER job. This is the easiest method for accomplishing dual image copies without having both recorded in the DB2 Catalog.

Ideally, IBM should solve this problem. But in the meantime, simply providing two additonal options for the copy utility and one additional option for the recover utility should suffice.

The copy utility should have two new options: dual and remote.

DB2 v. 2.3 will provide a quicker recover utility

Specifying dual would cause the copy utility to produce two identical image copy datasets, both recorded in the DB2 catalog. (Note: BMC Software, a vendor specializing in database software, provides a product that creates dual image copies recorded in the DB2 Catalog.) The second option, remote, could then be specified only in combination with the dual option. The remote option would flag the first of the two image copies as unavailable because it was sent off-site. This information should be stored in an additional column in the SYSIBM.SYSCOPY table to allow the recover utility to access it.

Remote would also be the additional option needed by the recover utility. Image copies marked "remote" in the DB2 Catalog could only be processed by recover utilities executed with the remote option. It's important that if the remote option is specified, only remote image copies can be used. You shouldn't allow mixing and matching of local and remote copies. Avoiding this will minimize confusion and enforce "backup" integrity. The recover utility would default to "local." Only nonremote image copies would be used by local recoveries.

```
QMF Query

     SELECT    S.DBNAME, S.DBID, S.NAME, S.PSID, T.CREATOR, T.NAME, T.OBID
     FROM      SYSIBM.SYSTABLESPACE  S, SYSIBM.SYSTABLES T
     WHERE     S.DBNAME = T.DBNAME
     AND       S.NAME   = T.TSNAME
     AND       T.TYPE   = 'T'
     ORDER BY  S.DBNAME, S.DBID, S.NAME, S.PSID, T.CREATOR, T.NAME


QMF Form

Total Width of Report Columns: 61

NUM      COLUMN HEADING       USAGE       INDENT      WIDTH      EDIT      SEQ

 1       DATABASE             BREAK1      1           8          C         1
 2       DBID                 BREAK1      1           4          L         2
 3       TABLE_SPACE          BREAK2      1           8          C         3
 4       PSID                 BREAK2      1           4          L         4
 5       TABLE_CREATOR                    1           8          C         5
 6       TABLE NAME                       1           18         C         6
 7       OBID                             1           4          L         7
```

Figure 4. *DBID, PSID, OBID report.*

Other problems also arise during DB2 disaster recovery planning. DB2 cannot force an archive log. This complicates the off-site archive log backup procedures. Archive logs can be copied only when they are archived to tape by DB2. The rate of archival is not predictable, though. Manual effort is required to analyze the rate of log archival. When necessary, a job is submitted to copy the archive logs that have yet to be copied for off-site storage. If log archival could be forced, however, a job could be periodically submitted in order to force an archive log and make a copy for off-site recovery.

This is a significant problem, because the BSDS and the SYSIBM.SYSCOPY table are backed up at log archival time. These two items play a substantial role in the recovery process.

Logging poses another problem as well. If a disaster strikes it will never be possible to restore all DB2 data to its state just prior to the disaster. At best, data can be restored only back to the last archive log that was sent off-site. This is one reason to have small active logs, thereby forcing more frequent log archival. The smaller the active log, the less data you'll lose. If DB2 provided remote-log and remote-copy capabilities, then it would be technically possible to recover data back to its most recent state.

With DB2 v. 2.3, announced last October, IBM has addressed some of the problems associated with DB2 disaster recovery. The copy utility has been enhanced to permit the creation of up to four image copies per execution. Two of the image copies can be designated for the local DB2 and the other two for the remote DB2. Taking two local and two remote image copies will speed recovery in the event of a bad image copy tape at either site. (Note: DB2 v. 2.3 also provides a quicker recover utility due to improved algorithms for reading archive log records.) In addition, a new command, ARCHIVE LOG, will force a log archival. These enhancements will go a long way toward establishing an optimum environment for enabling DB2 disaster recovery. DB2 v. 2.3, however, won't be generally be available until fourth quarter 1991.

HELPFUL HINTS

The following tips can help you prepare your disaster recovery plan:

□ Ensure that you have an adequate disaster recovery plan for the DB2 subsystem. This involves backing up system datasets and system tablespaces, and integrating the timing of the backups with the needs of each DB2 application.

□ Remember that the recover utility can only recover using the backup tapes that have been sent to the remote site. Updates on the active log at the time of the disaster will be lost, as will all archive logs and image copy backup tapes that were not sent off-site.

□ Ensure that every tablespace has at least one and preferably three valid off-site image copy backups.

□ When running the copy utility, always use SHRLEVEL REFERENCE. Running copy with SHRLEVEL CHANGE could cause inconsistent data and will make the recover utility take longer to execute.

□ Catalog all image copy backups, and run IEBGENER to produce a second backup copy on magnetic tape.

□ Document the backup strategy for each tablespace.

□ Document the state of each DB2 application and the state of the DB2 subsystem by producing DB2 Catalog, DB2 Directory, and BSDS reports after producing your off-site backups. Send this information to your remote site daily.

□ Keep the active log relatively small, but not so small as to impact system performance.

□ Backup each application's tablespaces at the remote site immediately after each application has been recovered.

□ Run a battery of SELECT statements against the recovered application tables to validate the state of their data.

□ Test your disaster recovery plan before a disaster strikes. This way, you can work out any problems before it's too late.

DB2 disaster recovery is a complex topic that deserves substantial attention. Each application needs to be analyzed to determine its optimal disaster recovery strategy. These guidelines, coupled with a comprehensive DB2 subsystem disaster plan, will provide a satisfactory disaster recovery mechanism for your corporation. ▮▮▮▮

REFERENCES

DB2 Offsite Recovery Restart, May 1990, white paper available from IBM Corp., 580 Walnut St., Cincinnati, OH 45202.

Haupert, R. "DB2 Recovery Considerations," *IDUG Proceedings,* May 1990.

"Operation and Recovery," *DB2 Administration Guide,* Version 2, Release 2, Chapter 5, document number SC26-4374-1, IBM Corp.

Craig S. Mullins is a database and systems administrator specializing in DB2 at Mellon Bank in Pittsburgh. He is also a cofounder and vice president of ASSET Inc., a customized-software and technical consulting firm, and a DB2 and SQL instructor.

Tables and Datasets Internal to DB2

A CERTAIN NUMBER OF control structures are necessary for DB2 to execute. DB2 uses several of these structures to accomplish data recovery. The following descriptions provide information about each of the internal tables and datasets mentioned in the disaster recovery article.

The DB2 Catalog

DB2's Catalog is composed of DB2 tables that collectively describe every object defined to a particular DB2 subsystem. Executing DDL statements or utility jobs will cause rows in DB2 Catalog tables to be updated, inserted, deleted, or selected. Because the DB2 Catalog is composed of DB2 tables, it can be accessed using SQL.

Of particular interest to the recovery process is the SYSIBM.SYSCOPY catalog table. It records any information about tablespace image copies, loads, reorganizations, and quiesces.

The DB2 Directory

DB2's Directory consists of four "tables" that are not truly DB2 tables. SQL can't be used to access the DB2 Directory. Information pertaining to the normal operation of DB2 is stored in these "tables."

DB2 recovery relies on the SYSIBM.SYSLGRNG "table." It's short for log range and contains tablespace relative byte addresses (RBAs). DB2 writes a row to SYSIBM.SYSLGRNG each time a tablespace is opened and updated. DB2 will update the row when the tablespace is closed.

Boot Strap Dataset

The Boot Strap Dataset, or BSDS for short, is a VSAM key-sequenced dataset. It's used to manage DB2's inventory of active and archive logs, and contains the RBA range on each log.

DB2 writes a record to the BSDS each time a new archive log dataset is defined. When an active log dataset is reused, DB2 will update the RBA range in the BSDS.

Active Logs

The active log consists of multiple VSAM entry-sequenced datasets. There can be anywhere from two to 53 active logs defined to a DB2 subsytem. Usually, DB2 will dual log (requiring from four to 106 VSAM ESDS datasets). The active log datasets must reside on DASD.

In most cases, whenever data in a DB2 table changes, DB2 will record the changes in the active log. The only exception to this rule is when the LOG(NO) option of the LOAD or REORG utility is used.

Archive Logs

The archive log consists of many physical sequential datasets. Up to 1,000 archive log datasets can be maintained on DASD, mass storage, or tape.

Active log datasets are copied to the archive log datasets when the active log dataset is full. This process is called off-loading.