October 2006 Volume II Issue 4

ORAtips Journal

ORAtips
On Database

Survey of Oracle® Recovery Options

By Steve Callan

***Editor's Note:** Losing data is like riding a roller-coaster, your stomach drops, you break out in a cold sweat, and you take a new interest in religion. ORAtips Associate Editor Steve Callan's latest contribution won't help you battle your amusement ride fears, but he can definitely take some of the fear out of losing data with his tips on recovery options and getting back to where you started.*

Introduction

From an applications or Applications standpoint, this article about recovery options is an example of where general knowledge is useful, but knowing the exact details isn't necessary because you have the resources to get those if needed. Unless you are in a down-in-the-trenches database administrator position, knowing the exact step-by-step details of how to recover a dropped table isn't essential. However, what is essential from an Applications or managerial standpoint is knowing the recovery operations at your disposal. It is also important to note that not all recovery operations may be available.

Dropping a table isn't likely (or not even possible depending on the application) in a Forms environment as your interaction with the database is primarily at the statement level. Statement level operations are those associated with insert, update, and delete, or in database terms, data manipulation language (DML) operations. An example would be deleting the salary record of an employee and then issuing a commit. If the record you just deleted happens to be that of your supervisor, you may be experiencing that unsettling "oops" feeling way down deep inside. Your protestation of "I didn't really mean to delete that information" is likely to be countered with a "Why were you looking at that in the first place" interrogative. Plenty of time for guilt and recrimination later, so for now, let's focus on retrieving the deleted data.

Oracle itself has built in mechanisms to recover process and instance failures, and human intervention is required for media failure.

Types Of Errors

As stated in Chapter 1 of Oracle Database Concepts, Oracle identifies errors as one of five types. These types are (and a brief description or example):

- User - Dropping a table
- Statement - Logical error in a procedure
- Process - Abnormal disconnection
- Instance - Power failure
- Media - Disk-related problems, generally requires media recovery

Statement errors, in the context defined in the documentation, should be minimized, if not eliminated, during the development, quality assurance review/testing, and user acceptance test phases of whichever software development lifecycle methodology is being used. After release into production, statement errors (bugs) may cause some data errors. Validation, integrity, and anomalies are types of data-related errors that can be difficult, if not impossible, to recover from. A series of bad statement errors (assuming there are intermediate commits in a procedure before an error is returned) may require an extensive effort to effect recovery.

Process, instance, and media errors will be immediately noticed by administrators, either system or database. These types of errors are immediate attention getters because across the board, they all result in the database (or instance, depending upon how precise you want to be in describing the problem) coming to an immediate halt or shutdown. Oracle itself has built in mechanisms to recover process and instance failures, and human intervention is required for media failure.

Statement, process, instance, and media errors are less likely to occur than user errors, and as mentioned, there are other processes, safeguards, or mechanisms in place to perform recovery. This leaves us with users as the main source of error-generating entities in an organization. How do users create errors? Errors are generated via statements issued by the user. In this context, a statement is valid in that it was successfully applied and there were no logic errors in terms of the statement's syntax. What is in error is what the statement performed.

Before discussing what your recovery options are, a brief overview of what Oracle uses for recovery will help frame the difference between what is immediately versus what is not immediately recoverable.

Structures Used In Recovery

There are three mechanisms used by Oracle for recovery: redo logs, undo records, and backups. These are the "proper" means of what is used for backup and recovery. Importing an export dump file is a fourth way of recovering a database (or table), but to be pedagogical and consistent with Oracle's definition of what constitutes a backup, export/import is not considered as a true backup strategy.

Immediate changes to data are stored in the redo logs and undo records. These structures allow for near instantaneous recovery (or the start of it, anyway). A recovery operation may take a significant amount of time if the original user operation involved an extensive amount of DML. Each of these structures is time-limited in that, because they are after all just datafiles, they can contain only so much information. Once a redo log is "filled up" to a certain point, it is archived off as an archived redo log and stored elsewhere on the file system (or eventually written to tape).

Undo records, present since version 9i, can ensure a specific amount of time in the past to be readily available, and may have more than the specified time. An undo retention period of 60 minutes results in Oracle maintaining an undo file (or space) large enough to store or record 60 minutes worth of change. Because the space tends to grow and shrink, you may have 70 minutes worth of look-back recovery time, but no guarantee of that extra ten minutes.

The third structure consists of database backups. Which backup method or utility you use depends on your situation or environment. If you can afford to lose data, your options are limited, but the steps to generate a backup are easier. If you cannot afford to lose data, which is more likely than not, then you (or more precisely, your DBA) have more options, but the steps are more involved.

A fourth structure listed in the documentation is the control file, but its role for the purpose of this article can be ignored. However, when talking with your DBA during a recovery operation, you should know that control files are essential to the operation of all databases and it is extremely important to have more than one copy of this file distributed across multiple disks. Casually ask where the control files are located for the production database, and if your DBA tells you they are all on the same drive or partition, you could be in serious trouble.

At this point in time, let's discuss how "point in time" applies in a recovery.

Immediate changes to data are stored in the redo logs and undo records.

ARCHIVELOG vs. NOARCHIVELOG Mode

If your database is operating in NOARCHIVELOG mode, you should first of all understand why it is acceptable to operate in this manner. Changes to the database are recorded in the redo logs, and if your system is not archiving the redo logs, there is no copy of them once they are overwritten. Let's suppose your database is shutdown at 2 AM every morning and that the relevant files are copied off somewhere before the database is restarted. Known as a cold backup, your ability to recover is limited to that point in time when the database was backed up. All changes since that point in time are lost. If you lose a disk at 2 PM, you can only recover to the cold backup taken at 2 AM. Can you afford to lose 12 hours worth of changes?

As a side note to should this be how your database is being backed up, ask if the redo logs are also copied during the cold backup. If so, ask why this takes place since copying the redo logs in a cold backup is in direct contravention of Oracle's guidance.

Another single point in time recovery option is offered via the export utility. The "exp" utility can create a logical record or backup of your database (tables, schemas, or the entire database). Suppose a user drops a table. If that table were included in the export dump file, then it can be imported back into your database. You may encounter integrity or constraint errors when importing directly into the owning schema, so it may be necessary to import the table into another schema (have the table be owned by another user) and then work on getting the data back into the original schema.

What if you don't need the entire table, but just a single record? You cannot import based on a query (but

ORAtips
On Database

you can export data filtered by a query), so you'll still need the entire table. How else can you retrieve the data for a single record?

Oracle's LogMiner utility "enables you to query redo log files through a SQL interface. All changes made to user data or to the database dictionary are recorded in the Oracle redo log files." One of the stated potential uses is for "Detecting and whenever possible, correcting user error," where user error is defined as a user "deleting the wrong rows because of incorrect values in a WHERE clause, [and] updating rows with incorrect values." [From the Oracle Database Concepts guide.]

LogMiner is frequently overlooked as a means of data recovery. A user operating in a well-functioning/well-operating database (meaning, you are following best practices with respect to backup and recovery) is not likely to have to use LogMiner, but it is a utility to keep tucked away in the back of your mind because of "what-if" situations, such as what if your database is not flashback enabled or what if your database version is 9i and you don't have a recycle bin? If you have to resort to LogMiner, it will help your DBA if you can provide an approximate time of when the user error occurred.

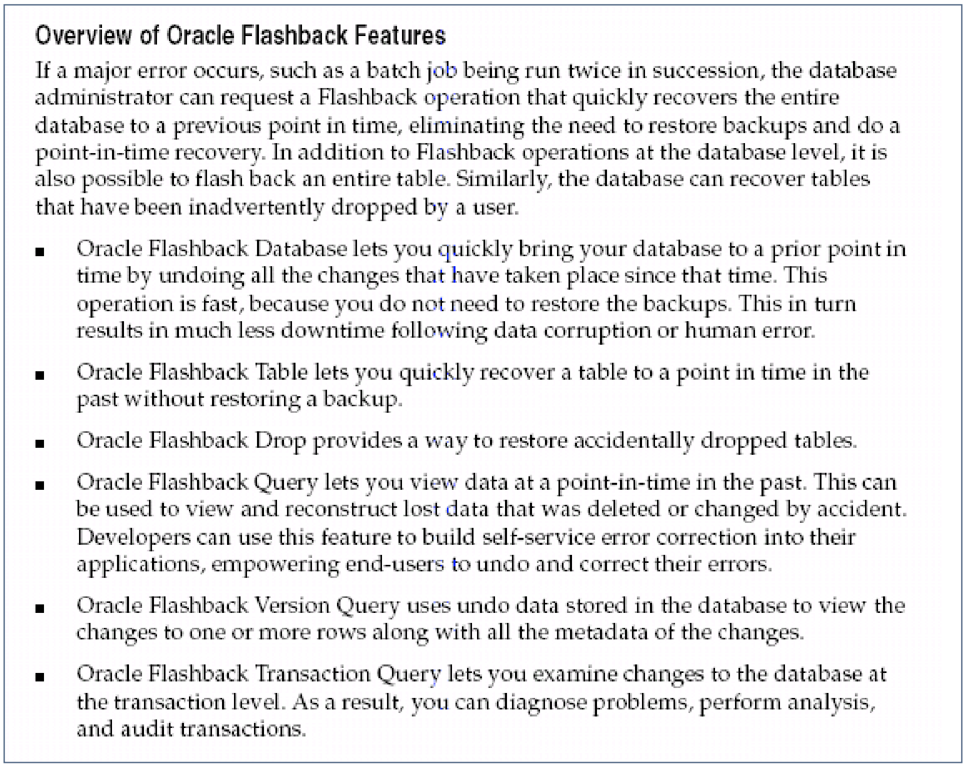
So far, we've been limited to a single point in time recovery, which is better than nothing, but one feature, which makes Oracle so powerful, is its ability to provide an almost infinite range of points in time, that is, you can recover to practically any point in time of your choosing. To facilitate this ability, your database will need to be in ARCHIVELOG mode.

In ARCHIVELOG mode, the database is still using redo logs, but as mentioned, once a redo log is filled up and cycled out of the rotation, it is archived off to a separate location. The original redo log stays in the same place; it's just that the file is copied to one or more locations. While in archive mode, your backup options now include the traditional hot backup and RMAN (Recovery Manager). One way to frame the difference here is that of user versus server-based or managed recovery. In other words, you can do the recovery work or let Oracle do it. Again, the point of this article is not to make you an expert on syntax, but to point out what your options are in a recovery situation – options that your DBA should be experienced and knowledgeable in.

Speaking of DBAs, your on-call DBA may be a new junior DBA. In a panic situation, it can help if others have some knowledge of what the possibilities are. Instead of guessing or remembering possibilities, your DBAs should have an "emergency procedures" guide for recovery operations. Although all of this information is available on the Internet, from Oracle or otherwise, and in numerous database administration books, you're goal is to minimize the research (and learn) time and maximize the "do" time.

Flashback

Flashback deserves two entries because there are two types of flashback available in Oracle 10g. Probably the more commonly known flashback has to do with flashback table or flashback database. The "other" flashback is flashback query. Flashback query does not undo anything, but it does allow you to look back in time (again, up to a certain point or limit). If a user made an error, one option would be to flashback query the affected table or data and store the results of that query in a working table. Then, at a more convenient time, the original data can be fixed or restored. But, since we are likely to



Overview of Oracle Flashback Features

If a major error occurs, such as a batch job being run twice in succession, the database administrator can request a Flashback operation that quickly recovers the entire database to a previous point in time, eliminating the need to restore backups and do a point-in-time recovery. In addition to Flashback operations at the database level, it is also possible to flash back an entire table. Similarly, the database can recover tables that have been inadvertently dropped by a user.

- Oracle Flashback Database lets you quickly bring your database to a prior point in time by undoing all the changes that have taken place since that time. This operation is fast, because you do not need to restore the backups. This in turn results in much less downtime following data corruption or human error.
- Oracle Flashback Table lets you quickly recover a table to a point in time in the past without restoring a backup.
- Oracle Flashback Drop provides a way to restore accidentally dropped tables.
- Oracle Flashback Query lets you view data at a point-in-time in the past. This can be used to view and reconstruct lost data that was deleted or changed by accident. Developers can use this feature to build self-service error correction into their applications, empowering end-users to undo and correct their errors.
- Oracle Flashback Version Query uses undo data stored in the database to view the changes to one or more rows along with all the metadata of the changes.
- Oracle Flashback Transaction Query lets you examine changes to the database at the transaction level. As a result, you can diagnose problems, perform analysis, and audit transactions.

Figure 1: Overview of Oracle Flashback Features

be more interested in a more immediate recovery, let's move on to the other type of flashback.

With a table or database operating in flashback mode, recovery is about as simple as issuing one command. You can flashback a table or database to a point in time without missing a beat. Figure 1 (from Chapter 17 in the Concepts guide) summarizes the different flashback features found in Release 2 of Oracle 10g.

As far as a user is concerned, barring any special privileges, is DBA intervention required to use a flashback feature? If you are in your own schema and drop a table you own, can you flashback table to recover it, or do you need DBA intervention? Fortunately, a user can recover objects owned by that user. However, you may want to clarify what is meant by user. There is a huge difference between "Scott" and a schema owner/user name "application_mgr." The impact of a Scott-level type of user dropping and recovering his own table is likely to be minimal, if significant at all. The main schema user performing the same step should be done in coordination with a DBA (and what if Scott can log on as application_mgr?).

You can flashback a table or database to a point in time without missing a beat.

So far, the user error somewhat implicitly implied here is an erroneous drop table statement. Unless there is a dropped version of the table with some rows of interest, flashback table basically applies to an entire table, not just a few rows. That is why flashback query deserves special mention as it can perform the look back in time to retrieve the original record set.

Part of what enables flashback table to work is a recycle bin. When a user drops a table, it can go into a recycle bin, and when flashback table is used, the user is retrieving the dropped object from the bin. A user can control his own recycle bin by setting a session parameter value. Further, a user can see the contents of his own bin by querying the USER_RECYCLEBIN view.

From a managerial standpoint, you may want to ensure that the necessary parameters for this feature are set by default in user sessions. In other words, you want to make it difficult for users to truly and permanently erase or alter data.

Conclusion

Oracle's, and hence your, ability to recover lost data has greatly improved with the release of Oracle 10g. One of the problems with new releases is that DBAs are usually so busy with maintaining the status quo that it is hard to assimilate the myriad of features found in a new release. No doubt, there are companies still using rollback segments and hot backups because things work and why fix what isn't broken reasoning. New features become old features, typically at the cost of making some older feature obsolete (say goodbye to rollback segments because they are going, going, gone in the near future). Backup and recovery skills, individually or jointly, can easily be argued as the key skill set needed by a DBA. Poor performance can slow things down, but having a less than optimally performing database is much better than having a database corrupted due to data loss.

One rule to live by when it comes to recovery is to never put yourself in an unrecoverable situation. A corollary to that is to not do anything, which makes the situation worse. Timely identification of an error goes a long way towards keeping the situation recoverable. Now that you're armed with a better understanding of recovery operations, you stand a better chance of recovering your manager's lost data.

Steve Callan – Steve is an Oracle DBA and developer. His Oracle experience includes several versions of the RDBMS, Forms & Reports, and Application Server. In addition to working with Oracle, Steve also spends time on researching other database systems such as SQL Server and DB2 and would someday like to start his own software company. Steve may be contacted at Steve.Callan@ERPtips.com.

ORAtips Journal