Oracle Backup and Recovery

An Oracle Technical White Paper
February 1999

INTRODUCTION

Backup and recovery is one of the most important aspects of database administration. If a database crashed and there was no way to recover it, the devastating results to a business could include lost data, lost revenue and customer dissatisfaction. Whether companies operate a single database or multiple databases storing hundreds of gigabytes or even terabytes of data, they share one common factor — the need to back up important data and protect themselves from disaster by developing a recovery plan.

ORACLE8*i* SERVER MANAGED RECOVERY PROTECTS DATA

Recovery Manager, a component introduced with Oracle8™, provides a tightly integrated method for creating, managing, restoring and recovering Oracle8 database backups. Other options to Recovery Manager also exist, such as Operating System backups, Export utility, or Oracle® Enterprise Manager backups to protect critical data.

This paper provides a thorough overview of concepts and processes relating to Oracle Backup and Recovery, including a comparison of basic backup and recovery strategies. The overview is followed by an in-depth discussion of the extensive new backup and recovery functionality contained in Oracle8*i*. With new enhancements like multiplexed backup sets, multiplexed archive logs, automatic catalog maintenance, recovery catalog cross check and many other commands and features, Oracle8*i* continues the Oracle tradition of providing the most comprehensive backup and recovery functionality of any database on the market.

BASICS OF BACKUP AND RECOVERY

A backup is a representative copy of data. This copy can include important parts of a database such as the control file, redo logs, and datafiles. A backup protects data from application error and acts as a safeguard against unexpected data loss, by providing a way to restore original data.

Backups are divided into physical backups and logical backups. Physical backups are copies of physical database files. The phrase "backup and recovery" usually refers to the transfer of copied files from one location to another, along with the various operations performed on these files. In contrast, logical backups contain data that is exported using SQL commands and stored in a binary file. Logical backups are used to supplement physical backups.

Restoring a physical backup means reconstructing it and making it available to the Oracle server. To recover a restored backup, data is updated using redo records from the transaction log. The transaction log records changes made to the database after the backup was taken. Oracle performs crash recovery and instance recovery automatically after an instance failure. In the case of media failure, a database administrator (DBA) must initiate a recovery operation.

Recovering a backup involves two distinct operations: rolling the backup forward to a more recent time by applying redo data, and rolling back all changes made in uncommitted transactions to their original state.

In general, recovery refers to the various operations involved in restoring, rolling forward, and rolling back a backup. Backup and recovery refers to the various strategies and operations involved in protecting the database against data loss and reconstructing the database should loss occur.

HOW ORACLE KEEPS RECORDS OF DATABASE TRANSACTIONS

Every time a change in the database occurs, Oracle generates a record of the change in the redo log buffer in memory. Oracle records both committed and uncommitted changes in redo log buffers. Oracle constantly writes the redo log buffers to the online redo log, which is on disk. Depending on whether Oracle runs in ARCHIVELOG or NOARCHIVELOG mode, the system can then archive this redo information by copying the online redo log to other locations on disk. Archived redo logs are crucial for recovery since they constitute a record of changes to the database.

BACKUP AND RECOVERY OPERATIONS

A backup is a snapshot of a datafile, tablespace, or database at a certain time. If periodic backups of the database have been made and data is lost, users can apply the stored redo information to their latest backup to make the database current again.

Oracle enables users to restore an older backup and apply only some redo data, thereby recovering the database to an earlier point in time. This type of recovery is called incomplete media recovery. If the backup was consistent, then users are not required to apply any redo data, at all.

A simple example of media recovery illustrates the concept. Suppose a user makes a backup of the database at noon. Starting at noon, one change to the database is made every minute. At 1 p.m. one of the disk drives fails, causing the loss of all data on that disk. Fortunately, Oracle records all changes in the redo log. The user can then restore the noon backup onto a functional disk drive and use redo data to recover the database to 1 p.m., reconstructing the lost changes.

ELEMENTS OF A BACKUP AND RECOVERY STRATEGY

Although backup and recovery operations can be intricate and vary from one business to another, the basic principles follow these four simple steps:

1. Multiplex the online redo logs
2. Run the database in ARCHIVELOG mode and archive redo logs to multiple locations
3. Maintain multiple concurrent copies of the control file
4. Take frequent backups of physical datafiles and store them in a safe place, making multiple copies if possible

As long as users have backups of the database and archive redo logs in safe storage, the original database can be recreated, even if a fire destroyed the server hardware.

KEY DATA STRUCTURES FOR BACKUP AND RECOVERY

Before users begin to think seriously about backup and recovery strategy, the physical data structures relevant for backup and recovery operations must be identified. This section discusses the following physical data structures:

- Datafiles
- Control Files
- Rollback Segments

- Online Redo Log Files
- Archived Redo Log Files

DATAFILES

Every Oracle database has one or more physical datafiles that belong to logical structures called tablespaces. The datafile is divided into smaller units called data blocks. The data of logical database structures, such as tables and indexes, is physically located in the blocks of the datafiles allocated for a database. Datafiles hold the following characteristics:

- A datafile can be associated with only one database.
- User-defined characteristics allow datafiles to automatically extend when the database runs out of space.
- One or more physical datafiles form a logical database storage unit called a tablespace.

The first block of every datafile is the header. The header includes important information such as file size, block size, tablespace, and creation timestamp. Whenever the database is opened, Oracle checks to see that the datafile header information matches the information stored in the control file. If it does not, then recovery is necessary.

Oracle reads the data in a datafile during normal operation and stores it in the buffer cache. For example, assume that a user wants to access some data in a table. If the requested information is not already in the buffer cache, Oracle reads it from the appropriate datafiles and stores it in memory.

The background process DBWR, known as Database Writer or DB Writer, is in charge of writing buffers to disk. A gap always exists between the time when Oracle changes a block in the buffer cache and the time when DBWR writes it to disk. The more data that accumulates in memory without being written to disk, the longer the recovery time, since a crash or media failure will force Oracle to apply redo data to recover those changes.

CONTROL FILES

Every Oracle database has a control file containing the operating system filenames of all other files that constitute the database. This important file also contains consistency information that is used during recovery, such as the:

- Database name
- Timestamp of database creation
- Names of the database's datafiles and online and archived redo log files
- Checkpoint, a record indicating the point in the redo log where all database changes prior to this point have been saved in the datafiles
- Information on backed up files (when using the Recovery Manager utility)

Users can multiplex the control file, allowing Oracle to write multiple copies of the control file to protect it against disaster. If the operating system supports disk mirroring, the control file can also be mirrored, allowing the O/S to write a copy of the control file to multiple disks.

Every time a user mounts an Oracle database, its control file is used to identify the datafiles and online redo log files that must be opened for database operation. If the physical makeup of the database changes, a new datafile or redo log file is created. Oracle then modifies the database's control file to reflect the change.

The control file should be backed up whenever the structure of the database changes. Structural changes can include adding, dropping, or altering datafiles or tablespaces and adding or dropping online redo logs.

ROLLBACK SEGMENTS

Every database contains one or more rollback segments, the logical structures contained in datafiles. Whenever a transaction modifies a data block, a rollback segment records the state of the information before it changed.

Oracle uses rollback segments for a variety of operations. In general, the rollback segments of a database store the old values of data changed by uncommitted transactions. Oracle can use the information in a rollback segment during database recovery to undo any uncommitted changes applied from the redo log to the datafiles, putting the data into a consistent state.

For example, a user changes a row value in a table from 5 to 7. The redo log keeps a record of this change, while the rollback segment stores the old value. Before the user can commit the transaction, the power goes out. After power is restored, the database performs a crash recovery operation: it uses the redo log to roll forward the value from 5 to 7, then uses the rollback segment to undo the uncommitted change from 7 to 5.

ONLINE REDO LOG FILES

Redo logs are absolutely crucial for recovery. For example, imagine that a power outage prevents Oracle from permanently writing modified data to the datafiles. In this situation, an old version of the data in the datafiles can be combined with the recent changes recorded in the online redo log to reconstruct what was lost.

Every Oracle database contains a set of two or more online redo log files. Oracle assigns every redo log file a log sequence number to uniquely identify it. The set of redo log files for a database is collectively known as the database's redo log. Oracle uses the redo log to record all changes made to the database.

Oracle records every change in a redo record, an entry in the redo buffer describing what has changed. For example, assume a user updates a column value in a payroll table from 5 to 7. Oracle records the old value in a rollback segment and the new value in a redo record. Since the redo log stores every change to the database, the redo record for this transaction actually contains three parts:

- The change to the transaction table of the rollback segment
- The change to the rollback segment data block
- The change to the payroll table data block

If the user then commits the update to the payroll table — to make permanent the changes executed by SQL statements — Oracle generates another redo record. In this way, the system maintains a careful watch over everything that occurs in the database.

Circular Use of Redo Log Files

Log Writer (LGWR) writes redo log entries to disk. Redo log data is generated in the redo log buffer of the system global area. As transactions commit and the log buffer fills, LGWR writes redo log entries into an online redo log file. LGWR writes to online redo log files in a circular fashion: when it fills the current online redo log file, called the active file, LGWR writes to the next available inactive redo log file. LGWR cycles through the online redo log files in the database, writing over old redo data. Filled redo log files are available for reuse depending on whether archiving is enabled:

- If archiving is disabled, a filled online redo log is available once the changes recorded in the log have been saved to the datafiles.
- If archiving is enabled, a filled online redo log is available once the changes have been saved to the datafiles and the file has been archived.

System Change Number (SCN)

The SCN is an ever-increasing value that uniquely identifies a committed version of the database. Every time a user commits a transaction, Oracle assigns it a new SCN. The SCN then becomes a type of ID stamp for purposes of recovery. Oracle uses SCNs in control files, datafile headers, and redo records. Every redo log file has both a log sequence number and low and high SCN. The low SCN records the lowest SCN recorded in the log file, while the high SCN records the highest SCN in the log file. SCNs are important in situations when the user must specify the exact point to which the database must recover.

ARCHIVED REDO LOG FILES

Archived log files are redo logs that Oracle has filled with redo entries, rendered inactive, and copied to one or more log archive destinations. Oracle can be run in either of two modes:

- ARCHIVELOG — Oracle archives the filled online redo log files before reusing them in the cycle.
- NOARCHIVELOG — Oracle does not archive the filled online redo log files before reusing them in the cycle.

Running the database in ARCHIVELOG mode has the following benefits:

- The database can be completely recovered from both instance and media failure.
- The user can perform online backups, i.e., back up tablespaces while the database is open and available for use.
- Archived redo logs can be transmitted and applied to a standby database, which is an exact replica of the primary database.

– Oracle8*i* supports multiplexed archive logs to avoid any possible single point of failure on the archive logs.

– The user has more recovery options, such as the ability to perform tablespace-point-in-time recovery (TSPITR).

– Additional administrative operations can be performed to store and keep track of the archived redo logs.

– Extra disk space is required to store the archived logs.

Running the database in NOARCHIVELOG mode has the following consequences:

– The user can only back up the database while it is completely closed after a clean shutdown.

– Typically, the only media recovery option is to restore the whole database, which causes the loss of all transactions issued since the last backup.

UNDERSTANDING BASIC BACKUP STRATEGY

A backup strategy provides a safeguard against data loss. Answering the following questions can help database administrators develop a strong backup strategy:

- What types of backup failure can occur?
- What information should be backed up?
- Which backup method should be used?
- Should backups be made online or offline?
- How often should backups be made?
- How can dangerous backup techniques be avoided?
- Should a standby database be created?

WHAT TYPES OF BACKUP FAILURE CAN OCCUR?

Data loss can occur for various reasons. Here are some of the most common types of failures that can lead to data loss:

A *statement failure* is a logical failure in the handling statement in an Oracle program. For example, a user issues a statement that is not a valid SQL construction. When statement failure occurs, Oracle automatically undoes any effects of the statement and returns control to the user.

A *process failure* is a failure in a user process accessing Oracle, i.e., an abnormal disconnection or process termination. The failed user process cannot continue work, although Oracle and other user processes can. If the user process fails while modifying the database, Oracle background processes undo the effects of uncommitted transactions.

An *instance failure* is a problem that prevents an Oracle instance, i.e., the SGA and background processes, from continuing to function. Instance failure can result from a hardware problem such as a power outage, or a software problem such as an operating system crash. When an instance fails, Oracle does not write the data in the buffers of the SGA to the datafiles.

A *user or application error* is a user mistake that results in the loss of data. For example, a user can accidentally delete data from a payroll table. Such user errors can require the database to be recovered to a point in time before the error occurred. To allow recovery from user error and accommodate other unique recovery requirements, Oracle provides for point-in-time recovery (PITR). PITR allows the database to be recovered to the instant in time before the data was deleted.

A *media failure* is a physical problem that arises when Oracle tries to write or read a file that is required to operate the database. A common example is a disk head crash that causes the loss of all data on a disk drive. Disk failure can affect a variety of files, including datafiles, redo log files, and control files. Because the database instance cannot continue to function properly, it cannot write the data in the database buffers of the SGA to the datafiles.

WHAT INFORMATION SHOULD BE BACKED UP?

A database contains a wide variety of types of data. When developing backup strategy, DBAs must decide what information they want to copy. The basic backup types include:

- Whole Database
- Tablespace

- Datafile
- Control File
- Archived Redo Log
- Parameter File
- Password File

In deciding what to back up, the basic principle is to prioritize data depending on its importance and the degree to which it changes. Archive logs do not change, for example, but they are crucial for recovering the database, so multiple copies should be maintained, if possible. Expense account tables, however, are constantly updated by users. Therefore, this tablespace should be backed up frequently to prevent having to apply as much redo data during recovery.

Backups can be combined in a variety of ways. For example, a DBA can decide to take weekly whole database backups, to ensure a relatively current copy of original database information, but take daily backups of the most accessed tablespaces. The DBA can also multiplex the all important control file and archived redo log as an additional safeguard.

Whole Database Backup

The most common type of backup, a whole database backup contains the control file along with all database files that belong to a database. If operating in ARCHIVELOG mode, the DBA also has the option of backing up different parts of the database over a period of time, thereby constructing a whole database backup piece by piece.

Consistent Whole Database Backup

In this backup, all datafiles and control files are consistent to the same point in time — consistent with respect to the same SCN, for example. The only tablespaces in a consistent backup that are allowed to have older SCNs are read-only and offline-normal tablespaces, which are consistent with the other datafiles in the backup. This type of backup allows the user to open the set of files created by the backup without applying redo logs, since the data is already consistent.

The only way to perform this type of backup is to shut down the database cleanly and make the backup while the database is closed. A consistent whole database backup is the only valid backup option for databases running in NOARCHIVELOG mode.

Inconsistent Whole Database

An inconsistent whole database backup can be used if 24x7 access to the database is a requirement. Also called an online backup, this backup is inconsistent because portions of the database are being modified and written to disk while the backup is progressing. The database must be run in ARCHIVELOG mode to make open backups.

Inconsistent Closed Backups

An inconsistent closed backup can be done after a system crash failure or SHUTDOWN ABORT. This type of backup is valid if the database is running in ARCHIVELOG mode, since all logs are available to make the backup consistent. Oracle does not recommend taking inconsistent closed backups.

Tablespace Backups

A tablespace backup is a backup of a subset of the database. Tablespace backups are only valid if the database is operating in ARCHIVELOG mode. The only time a tablespace backup is valid for a database running in NOARCHIVELOG mode is when that tablespace is read-only or offline-normal.

Datafile Backups

A datafile backup is a backup of a single datafile. Datafile backups, which are not as common as tablespace backups, are only valid if the database is run in ARCHIVELOG mode. The only time a datafile backup is valid for a database running in NOARCHIVELOG mode is if that datafile is the only file in a tablespace. For example, the backup is a tablespace backup, but the tablespace only contains one file and is read-only or offline-normal.

Control File Backups

A control file backup is a backup of a database's control file. If a database is open, the user can issue the following SQL statement:

```
ALTER DATABASE BACKUP CONTROLFILE to 'location';
```

Recovery Manager can also be used to back up the control file. If the user is going to make an O/S backup of the control file, the database must be shut down first.

WHICH BACKUP METHOD SHOULD BE USED?

Oracle gives users a choice of several basic methods for making backups. The methods include:

- *Recovery Manager (RMAN)* — A component that establishes a connection with a server process and automates the movement of data for backup and recovery operations.
- *Operating System (O/S)* — The database is backed up manually by executing commands specific to the user's *operating system*.
- *Oracle Export Utility* — The utility makes logical backups by writing data from an Oracle database to operating system files in a proprietary format. This data can later be imported into a database.
- *Oracle® Enterprise Manager* — A GUI interface that invokes Recovery Manager.

Making Recovery Manager Backups and Copies

Recovery Manager is a powerful and versatile program that allows users to make a backup or copy of their data. When the user specifies files or archived logs using the Recovery Manager backup command, Recovery Manager creates a backup set as output. A backup set is a file or files in a Recovery Manager-specific format that requires the use of the Recovery Manager *restore* command for recovery operations. In contrast, when the *copy* command is used to create an image copy of a file, it is in an instance-usable format — the user does not need to invoke Recovery Manager to restore or recover it.

When a Recovery Manager command is issued, such as *backup* or *copy*, Recovery Manager establishes a connection to an Oracle server process. The server process then backs up the specified datafile, control file, or archived log from the target database. Recovery Manager obtains the information it needs from either the control file or the optional recovery catalog. The recovery catalog is a central repository containing a variety of information useful for backup and recovery.

Recovery Manager automatically establishes the names and locations of all the files needed to back up. Recovery Manager also supports incremental backups — backups of only those blocks that have changed since a previous backup. In traditional backup methods, all the datablocks ever used in a datafile must be backed up.

Making O/S Backups

If the user does not want to use Recovery Manager, operating system commands can be used such as the UNIX *dd* or *tar* command to make backups. Backup operations can also be automated by writing scripts. The user can make a backup of the whole database at once or back up individual tablespaces, datafiles, control files, or archived logs. A whole database backup can be supplemented with backups of individual tablespaces, datafiles, control files, and archived logs. O/S commands can also be used to perform these backups if the database is down or if the necessary commands are entered to take an online backup.

Using the Export Utility for Supplemental Backup Protection

Physical backups can be supplemented by using the Export utility to make logical backups of data. Logical backups store information about the schema objects created for a database. The Export utility writes data from a database into Oracle files in a proprietary format, which can then be imported into a database using the Import utility.

Oracle Enterprise Manager

Although Recovery Manager is commonly used as a command-line utility, Recovery Manager can also perform backups using Oracle Enterprise Manager. The Backup Manager in Oracle Enterprise Manager is a GUI interface to Recovery Manager that enables backup and recovery via a point-and-click method.

Feature	*Recovery Manager*	*Operating System*	*Export*	*Oracle Enterprise Manager*
Closed Database Backups	*Supported. Requires instance to be mounted.*	*Supported.*	*Not supported.*	*GUI supported for RMAN.*
Open Database Backups	*DBA does not use BEGIN/END BACKUP commands.*	*Use BEGIN/END BACKUP commands.*	*Requires RBS to generate consistent backups.*	*GUI supported for RMAN.*
Incremental Backups	*Supported. Backs up all modified blocks.*	*Not supported.*	*Supported, but not true incremental. Backs up a whole table even if only one block is modified.*	*GUI supported for RMAN.*
Corrupt Block Detection	*Supported. Identifies corrupt blocks and writes to V$BACKUP_CORRUPTIO N or V$COPY_CORRUPTION.*	*Not supported.*	*Supported. Identifies corrupt blocks in the export log.*	*GUI supported for RMAN.*
Automatic Backup	*Supported. Establishes the name and locations of all files to be backed up (whole database, tablespace, datafile or control file backup).*	*Not supported. Files to be backed up must be specified manually.*	*Supported. Performs either full, user or table backups.*	*RMAN scripts get scheduled using the Oracle Job System.*
Backup Catalogs	*Supported. Backups are cataloged to the recovery catalog and to the control file, or just to the control file.*	*Not supported.*	*Not supported.*	*GUI supported for RMAN.*
Backups to Sequential Media	*Supported. Interfaces with a media manager.*	*Supported. Backup to tape is manual or managed by a media manager.*	*Supported.*	*GUI supported for RMAN.*
Backs up init.ora and Password Files	*Not supported.*	*Supported.*	*Not supported.*	*Not supported.*
Operating System Independent Language	*O/S independent scripting language*	*O/S dependent*	*O/S independent scripting language*	*O/S independent scripting language*

Feature Comparison of Backup Methods

SHOULD BACKUPS BE MADE ONLINE OR OFFLINE?

Online backups or offline backups can be made using either Recovery Manager or O/S commands. An online backup, also known as an open backup, is a backup of one or more database files that is made while the database is open. An offline backup, also known as a closed backup, is a backup of one or more database files that made after the database has been closed cleanly.

If the database must be open and available all the time, then online backups are the only option. If there is a time of little or no activity on the database, then the user may decide to take periodic offline backups of the whole database, then supplement them with online backups of tablespaces that generate a great deal of activity.

HOW OFTEN SHOULD BACKUPS BE MADE?

A backup strategy should be tailored to company needs. For example, if the company can afford to lose data in the event of a disk failure, backups may not need to be performed very often. The advantage of taking infrequent backups is that the user frees Oracle's resources for other operations. The disadvantage is the company may end up either losing data or recovering data but spending a long time doing so.

If the database must be available twenty-four hours a day, seven days a week, it should be backed up frequently. Otherwise, a disaster can wipe out the database and destroy the business — or at least cause a serious setback. In this case, it may be necessary to take daily backups, multiplex online redo logs, and archive redo logs to several different locations. The user can even maintain a standby database in a different city that is a constantly updated replica of the original database.

HOW CAN DANGEROUS BACKUP TECHNIQUES BE AVOIDED?

Although it may seem that online redo logs should be backed up along with the datafiles and control file, this technique is dangerous. Online redo logs should not be backed up for the following reasons:

- If the database is in ARCHIVELOG mode, ARCH is already archiving the redo logs.
- Online logs can be protected against media failure by multiplexing them, i.e., having multiple log members per group, on different disks and disk controllers.

- If the database is in NOARCHIVELOG mode, the only type of backups that should be performed are closed, consistent, whole database backups. The files in this type of backup are all consistent and do not need recovery, so the online logs are not needed.

Backing up online redo logs can be dangerous since the user may accidentally restore them. There are a number of situations where restoring the online logs can cause significant problems in the recovery process. For example, it's easy to make a simple mistake when a crisis occurs. When restoring the whole database, it is possible to accidentally restore the online redo logs, thus overwriting the current logs with older, useless backups. This action forces the DBA to perform an incomplete recovery instead of the intended complete recovery, thereby losing the ability to recover valuable transactions.

SHOULD A STANDBY DATABASE BE CREATED?

A standby database maintains a duplicate, or standby copy of the primary (also known as production) database and provides continued primary database availability in the event of a disaster. A standby database is constantly in recovery mode. If a disaster occurs, the standby database can be taken out of recovery mode and activated for online use.

A standby database is intended for recovery of the primary database. However, with Oracle8*i*, the standby database can be opened in query mode so batch reporting can be offloaded from the primary site. Once the standby database is activated after disasters, it cannot be returned to standby recovery mode unless it is re-created as another standby database.

Warning: Activating a standby database resets the online logs of the standby database. Therefore, after activation, the logs from the standby database and production database are incompatible. The user must place the data files, log files, and control files of the primary and standby databases on separate physical media. Therefore, it is impossible to use the same control file for both the primary and standby databases.

UNDERSTANDING BASIC RECOVERY STRATEGY

Basic recovery involves two parts: restoring a physical backup and then updating it with the changes made to the database since the last backup. The most important aspect of recovery is making sure all data files are consistent with respect to the same point in time. Oracle has integrity checks that prevent the user from opening the database until all data files are consistent with one another.

When preparing a recovery strategy, it is critical to understand the answers to these questions:

- How does recovery work?
- What are the types of recovery?
- Which recovery method should be used?

HOW DOES RECOVERY WORK?

In every type of recovery, Oracle sequentially applies redo data to data blocks. Oracle uses information in the control file and datafile headers to ascertain whether recovery is necessary.

Recovery has two parts: rolling forward and rolling back. When Oracle rolls forward, it applies redo records to the corresponding data blocks. Oracle systematically goes through the redo log to determine which changes it needs to apply to which blocks, and then changes the blocks. For example, if a user adds a row to a table, but the server crashes before it can save the change to disk, Oracle can use the redo record for this transaction to update the data block to reflect the new row.

Once Oracle has completed the rolling forward stage, it begins rolling back. The rollback information is stored in transaction tables. Oracle searches through the table for uncommitted transactions, undoing any that it finds. For example, if the user never committed the SQL statement that added the row, then Oracle will discover this fact in a transaction table and undo the change.

WHAT ARE THE TYPES OF RECOVERY?

There are three basic types of recovery: instance recovery, crash recovery, and media recovery. Oracle performs the first two types of recovery automatically at instance startup. Only media recovery requires the user to issue commands.

An *instance recovery*, which is only possible in an Oracle® Parallel Server configuration, occurs in an open database when one instance discovers that another instance has crashed. A surviving instance automatically uses the redo log to recover the committed data in the database buffers that was lost when the instance failed. Oracle also undoes any transactions that were in progress on the failed instance when it crashed, then clears any locks held by the crashed instance after recovery is complete.

A *crash recovery* occurs when either a single-instance database crashes or all instances of a multi-instance database crash. In crash recovery, an instance must first open the database and then execute recovery operations. In general, the first instance to open the database after a crash or SHUTDOWN ABORT automatically performs crash recovery.

Unlike crash and instance recovery, a *media recovery* is executed on the user's command, usually in response to media failure. In media recovery, online or archived redo logs can be used to make a restored backup current or to update it to a specific point in time.

Media recovery can restore the whole database, a tablespace or a datafile and recover them to a specified time. Whenever redo logs are used or a database is recovered to some non-current time, media recovery is being performed.

A restored backup can always be used to perform the recovery. The principal division in media recovery is between complete and incomplete recovery. Complete recovery involves using redo data combined with a backup of a database, tablespace, or datafile to update it to the most current point in time. It is called complete because Oracle applies all of the redo changes to the backup. Typically, media recovery is performed after a media failure damages datafiles or the control file.

Recovery Options

If the user does not completely recover the database to the most current time, Oracle must be instructed how far to recover. The user can perform:

- Tablespace point-in-time recovery (TSPITR), which enables users to recover one or more tablespaces to a point-in-time that is different from the rest of the database.
- Time-based recovery, also called point-in-time recovery (PITR), which recovers the data up to a specified point in time.
- Cancel-based recovery, which recovers until the CANCEL command is issued.
- Change-based recovery or log sequence recovery. If O/S commands are used, change-based recovery recovers up to a specified SCN in the redo record. If Recovery Manager is used, log sequence recovery recovers up to a specified log sequence number.

When performing an incomplete recovery, the user must reset the online redo logs when opening the database. The new version of the reset database is called a new incarnation. Opening the database with the RESETLOGS option tells Oracle to discard some redo and prevents Oracle from ever applying the discarded redo in any recovery the user might do in the future. Since the existing backups are no longer usable for recovery, the user should make a whole database backup immediately after resetting the online redo log.

WHICH RECOVERY METHOD SHOULD BE USED?

Users have a choice between two basic methods for recovering physical files. They can:

- Use the Recovery Manager utility to automate recovery.
- Recover the database manually by using Oracle Enterprise Manager or executing SQL commands.

Recovering with Recovery Manager

The basic Recovery Manager commands are restore and recover. Recovery Manager can be used to restore datafiles from backup sets or from file copies on disk, either to their current location or to a new location. Backup sets containing archived redo logs can also be restored.

In a recovery catalog, Recovery Manager keeps a record containing all the essential information concerning every backup ever taken. If a recovery catalog is not used, Recovery Manager uses the control file for necessary information. The Recovery Manager recover command can be used to perform complete media recovery and apply incremental backups, and to perform incomplete media recovery. Recovery Manager completely automates the procedure for recovering and restoring backups and copies. Recovery Manager is described further in the sections below.

Recovering with SQL and SQL*Plus®

Administrators can also use SQL commands or the SQL*Plus utility at the command line to restore and perform media recovery on your files. These include the: SQL ALTER DATABASE RECOVER command and the SQL*Plus RECOVER command.

If the operating system supports Oracle Enterprise Manager, the user can execute SQL restore and recovery commands in a GUI environment. In each case, users can recover a database, tablespace, or datafile.

Before performing recovery, users need to:

- Determine which files to recover. Often the table V$RECOVER_FILE can be used.
- Restore backups of files permanently damaged by media failure. If the user does not have a backup, recovery can still be performed if the user has the necessary redo log files and the control file contains the name of the damaged file.
- If a file cannot be restored to its original location, then the user must relocate the restored file and inform the control file of the new location.
- Restore necessary archived redo log files.

Note: The Server Manager RECOVER command can also be used, although Oracle plans to stop supporting Server Manager in future releases.

BENEFITS OF RECOVERY MANAGER

The key objective of Recovery Manager is to provide greater ease of management and administration of the backup and recovery operations, while maintaining superior performance and increased availability of the database. Recovery Manager and the Oracle server minimize operational errors, by automating the backup and recovery process, and by performing multiple consistency checks throughout the procedure. The release of Oracle8*i* provides the customer with even more features and functionality designed to leverage Recovery Manager even further.

Recovery Manager is a command line component (similar to SQL*Plus) that interacts with the Oracle server code to perform backup and recovery operations. Recovery Manager decides how to execute backup, restore, or recovery operations based on information in the database's recovery catalog, control file, and the current state of the datafiles, then instructs and coordinates Oracle server processes to perform the desired operation. Recovery Manager calls the Oracle server to back up and recover the database rather than using operating system products such as the UNIX *dd* command.

The Oracle server maintains records of all successful backups initiated by Recovery Manager within that database's control file. At the end of a backup, this information is propagated to an optional recovery catalog for long-term storage.

The recovery catalog is a set of Oracle schema that stores current and historical information about the target database, and also records information regarding Recovery Manager backups and copies. The tables, views, and code that constitute the recovery catalog are stored in an Oracle database. The recovery catalog maintains both a current and historical view of the structure of the database, successful backups and file copies and archived redo logs generated by the database.

KEY FEATURES OF RECOVERY MANAGER

There are a number of significant benefits to using Recovery Manager that are only possible because an Oracle server process performs the backup. These include:

- An increase in usability, reliability and performance
- Automation of backup, restore and recovery operations
- Whole database backups (or backups of any logical unit: control file, datafile, tablespace or archive log)
- Open or closed backups
- Two types of backup: image copies to disk or backup sets to disk or to tape with media management software
- Intelligent management of the archived redo logs for both backup and recovery
- Corrupt block detection
- Tablespace Point-in-Time Recovery support
- Ability to catalog on disk operating system backups
- Integration with Oracle Enterprise Manager's Backup Manager GUI
- Incremental backups at the Oracle block level for high performance backup and recovery
- Omission of empty blocks during backup for optimization
- Recoverability redundancy when both incrementals and archiving are used
- No extra redo is generated during open database backup
- Intelligent buffering with multiplexing of multiple input files for high speed tape streaming
- Support for Oracle Parallel Server backup and recovery
- Ability to recover through unrecoverable operations, when using incremental backups
- O/S-independent scripting language

As an alternative to using the Recovery Manager language, Recovery Manager is supported by Oracle Enterprise Manager's Backup Manager, a utility that provides a graphical user interface for performing Recovery Manager tasks. Oracle Enterprise Manager also provides job scheduling to facilitate those tasks.

AUTOMATION OF BACKUP AND RECOVERY

Recovery Manager automates backup and recovery by querying information in the recovery catalog, the database's control file, and any datafiles affected by the operations requested. Recovery Manager decides the most efficient method of executing the requested backup, restore, or recovery operation and then issues these steps to the Oracle server. Recovery Manager and the server achieve this by automatically identifying changes in the structure of the database, and dynamically adjusting the current operation to cater to the changes.

Backup

During a whole database backup operation, Recovery Manager responds to changes in the structure of the database by backing up all available datafiles. For example, the addition of a new tablespace (or datafile) would not require any modification in the following database backup script:

```
run {
allocate channel c1 type ‘sbt_tape’;
allocate channel c2 type ‘sbt_tape’;
backup database;
 }
```

In this example, Recovery Manager automatically backs up all files belonging to the database, including any new files added. The allocate channel command establishes a connection to a target database instance. Each connection operates on one backup set at a time (for backup, restore, or recover) or one file copy at a time (for copy). If multiple connections are established, then each connection operates on a separate backup set or file copy.

Restore and Recovery

Recovery Manager also uses the Recovery Catalog to decide which backups are most applicable for restoration and recovery, then hands control over to the Oracle server, which performs the required actions. The user does not need to maintain records of which files were backed up or maintain historical information on the structure of the database at any point in time, as this information is inherent to the recovery catalog.

Recovery Manager greatly simplifies recovery by automatically identifying the most appropriate backups (full and incrementals) and archived redo logs to use, then restores and recovers them. Database point-in-time recovery is made simple, as Recovery Manager intelligently restores only the files that existed in the database at that time.

If a DBA were to perform a database point in time recovery without using Recovery Manager, the DBA would have to perform the same functions manually. If the backups of the database were not whole database backups, determining "which files were backed up when" would only be possible with excellent record keeping by the DBA and operations staff.

Recovery Manager automatically plans and executes the recovery procedure. The question of which backups to use, the application of incremental backups and restoration and application of archivelogs is all transparent to the DBA. The recovery is stopped at the time specified by the "until time" clause, making recovery a very simple operation.

Recovery Manager also backs up archived redo logs. Only redo logs that have been completely archived are backed up by Recovery Manager. This safeguard prevents the common problem of backing up an incomplete archived redo log (i.e., backing up the log before it has been completely written by the ARCH process). Once successfully backed up, the archived redo log can optionally be deleted by Recovery Manager.

BENEFITS OF AN ORACLE SERVER PERFORMING THE BACKUP

Using an Oracle server to perform backups provides many important advantages to customers. This section discusses several key features provided by Oracle.

CORRUPT BLOCK DETECTION

All I/O for backup and restore is performed by an Oracle server process. The integrity checks that are normally performed when reading a datafile to satisfy a SQL operation, are also performed when creating or restoring from a backup. Therefore, many types of corruption are detected and noted by the server process while the backup is in progress.

SPLIT BLOCK DETECTION AND ONLINE BACKUP MODE

The Oracle server can detect "split blocks" while the backup is in progress, so "online backup mode" is not needed when using Recovery Manager to perform online backups. As a result, online backups using Recovery Manager do not produce any extra redo.

CHECKSUMS FOR ADDITIONAL RELIABILITY

As an additional reliability measure, a checksum is computed for each block written to a backup. The checksum is verified during restore and recovery. This makes it possible for Oracle to identify whether a block in the backup has been in any way corrupted or tampered with in the time between when it was first written out to backup, and the time when it is being restored.

INCREMENTAL BACKUPS AND RECOVERY

An incremental backup is a backup of a datafile that includes only Oracle blocks that were changed since a previous incremental. Performing incremental backups typically results in reduced backup size and time.

Incremental backups can only be performed on datafiles, and not on control files or archived redo logs. There are five incremental levels: 0 and levels 1 to 4 . The level 0 incremental is a baseline backup — all blocks which contain data are backed up and empty blocks are omitted. Subsequent incrementals at levels greater than 0 will only back up changed blocks.

If the incremental keyword is not included, by default Recovery Manager will perform a full backup. A full backup means the backup will include all blocks which contain data, but the backup will not affect subsequent incrementals. The keyword full* refers to whether an incremental is performed, and does not refer to how much of the database is backed up.

There are two types of incremental backups: cumulative and differential. By default, incremental backups are differential. Cumulative backups may take more time and space than non-cumulative, however only one cumulative backup from any level will be needed for recovery.

During recovery, incremental backups are chosen by Recovery Manager, and applied automatically by the Oracle server to recover the datafile. If the application of incrementals does not recover the file to the specified time, any archived redo logs that must be applied to recover the datafile will be automatically restored and applied by Recovery Manager.

Incremental backups also offer an additional level of redundancy for recovery. In the case that archived redo logs were not successfully backed up (or were corrupted), incremental backups offer a way of making the datafile newer without needing this redo. Conversely, if the tape that an incremental backup resides on is corrupt, archived redo logs can be used to roll forward.

The application of incremental backups is very efficient because:

- Incremental backups can be applied in parallel to multiple datafiles concurrently.
- The application of a block image will be faster than applying redo.
- Much less redo is applied during recovery (as the application of an incremental may obsolete a considerable amount of redo).

Incremental backups completed after UNRECOVERABLE operations are performed can be used to recover those datafiles. Without incrementals, a full backup of all files involved in the UNRECOVERABLE operation must be performed after the operation completes or recovery of these files is not possible.

MULTIPLEXING AND EMPTY BLOCK OMISSION

To keep very high performance tape drives streaming, it is often necessary to perform multiple reads on multiple devices. Recovery Manager automatically multiplexes input files in backup sets by performing multiple concurrent reads of distinct files. This provides ample data to keep the tape devices busy, without saturating a single datafile with too many read requests.

Recovery Manager also omits backing up empty blocks in backup sets, resulting in much smaller and faster backups (as less is written out during backup). During restore, Recovery Manager then restores the files and recreates the empty blocks skipped during backup.

SUPPORT FOR ORACLE PARALLEL SERVER

Recovery Manager is able to distribute backups, restores and recoveries over multiple instances in an Oracle Parallel Server configuration. To do this, the DBA specifies multiple connect strings in the Recovery Manager backup or restore command — server processes on the instances specified will then be started to perform the actions specified.

There are a number of requirements imposed by Recovery Manager in an Oracle Parallel Server environment:

- Recovery Manager assumes all backup files are equally available and restorable on all nodes configured to be able to restore files. This is only an issue as some media management products assume that a file backed up on one node will only be restored on that node.

- Archivelog backups can easily be distributed over all desired nodes, however the initial instance Recovery Manager connects to must be able to read the archivelog headers as a preparation for their backup, and also must be able to read the archivelogs during recovery. The requirement that the node performing the recovery is able to read archivelogs is not specific to Recovery Manager, but has always been required in an Oracle Parallel Server environment.

SUPPORT FOR CATALOGING O/S BACKUPS

Recovery Manager provides a facility whereby it is possible to register in the Recovery Catalog on-disk backups that were created using O/S utilities. To achieve this, Recovery Manager inspects the datafile copies, and registers information about the copies in the Recovery Catalog (the reason for requiring the backups on disk).

Cataloging O/S copies is most applicable at sites that break mirrors as the first part of a backup strategy. The volumes belonging to the broken mirror half can be cataloged as O/S backups. Once cataloged, Recovery Manager can backup the O/S copies to tape. Once the backups are cataloged, Recovery Manager can restore and recover them.

SUPPORT FOR TABLESPACE POINT-IN-TIME RECOVERY (TSPITR)

Oracle8 introduced the ability to recover a tablespace to a point-in-time that is inconsistent with the remaining database, and then to integrate that tablespace into the primary database. Recovery Manager assists in this process by automating the restore and recovery of the clone instance.

BACKUPS TO TAPE WITH RECOVERY MANAGER

Recovery Manager is able to create backup sets and datafile copies on disk. However, the Oracle8*i* server does not contain code to manage writing to, or reading from, sequential storage devices, such as magnetic tapes.

To solve this requirement, Recovery Manager and Oracle8*i* create an architecture that allows Oracle to manage the process of database backup and recovery, yet integrate with tape storage management subsystems for tape backup. Oracle designed an interface specification that allows the Oracle server to call third party Media Management products to backup to and restore from tape.

The interface specification is called the Media Management API (or System Backup to Tape API), and is a programming interface published by Oracle. Media management vendors who are members of Oracle's Backup Solutions Program are provided with a Software Developers Kit (SDK) which contains this interface specification, and information which assists with their development of the interface library.

The interface software developed by the Media Manager is known as the Media Management Library, or Media Management Layer (MML). The MML is called by Oracle when it needs to write to, or read from, devices supported by the Media Manager. The MML then communicates with the Media Management Server to write the backup to tape. The location of the backup files is managed by the media manager — the Oracle server simply requests a particular backup file be written or restored by the media manager.

This architecture exploits the robustness and expertise of both Oracle and Media Management and allows customers to choose from the many tape storage management products which comply with Oracle's Backup Solutions Program (BSP). For information on Media Management vendors who are partners in Oracle's BPS, please visit http://www.oracle.com, contact your local Oracle representative, or ask your Media Management Vendor whether they are BPS members.

COMPARING RECOVERY MANAGER IN ORACLE8*i* TO ORACLE8*i* ENTERPRISE EDITION

There are a number of restrictions in the Recovery Manager product distributed with the Oracle8*i* Server. These are:

- Only one stream of data (only one backup server process) is allowed.
- Incremental backups are not available.
- The Enterprise Edition allows both incremental backups and an unlimited number of data streams. The number of streams (server processes performing backups) possible is only limited by the hardware or operating system.

NEW RECOVERY MANAGER FEATURES IN ORACLE8*i*

In addition to all the features already available in Recovery Manager, several new pieces of functionality have been added to Recovery Manager for Oracle8*i*. These features include:

- Proxy Copy
- Disk affinity
- LIST command enhancements
- Recovery catalog crosscheck
- Multiplexed backup sets
- Control file character set specification
- STARTUP and SHUTDOWN commands
- DUPLICATE command
- Automatic catalog maintenance

PROXY COPY

Recovery Manager controls the integration of the database server directly, using a tape media management system which is compliant with Oracle's "System Backup to Tape (SBT)" tape media management application programming interface (API). This allows customers to choose from a variety of tape media management subsystems and enables Oracle to read or write tape volumes — perhaps in a tape library — directly.

The Oracle8*i* server can integrate with any media management software that is compliant with Oracle's Media Management API. This allows customers to choose from a variety of tape media management subsystems, and enables Oracle to read or write backups on non-disk media, such as tapes, and to use automated tape libraries.

In Oracle8*i*, the Media Management API has been enhanced to allow the bulk I/O of creating and restoring database backups to be offloaded from the Oracle host to the media manager. This makes it possible, during full online or offline backups, for the Oracle database server to not be directly involved in the reading or writing process. Therefore, media managers that directly integrate disk and tape storage subsystems can perform data movement between disk and tape at the speed of the underlying devices, freeing host CPU resources, and reducing the impact on application response time of backup or restore workloads.

This capability is known as "proxy copy" because Oracle selects the media manager as a proxy to actually copy data to or from tape. Proxy Copy will soon be supported by a number of media management vendor partners in Oracle's Backup Solutions Program (BSP).

DISK AFFINITY

In Oracle Parallel Server configurations, the speed of access to disks may not be uniform across the nodes of the cluster. One or more nodes may have "affinity" for a particular set of disks. Consequently, access to these disks from these nodes is more efficient than from other nodes. Recovery Manager in Oracle8*i* is aware of this affinity when scheduling backups and restores and will attempt to schedule datafile backups on channels allocated at nodes that have affinity to those files.

LIST ENHANCEMENTS

The LIST command in Oracle8*i* now supports multiple objects, not just one. The list backup set output has been enhanced to show backup pieces and files included in the set. The *all* keyword is added to list all backup sets in the catalog, whether usable by the current database incarnation or not.

RECOVERY CATALOG CROSSCHECK

The media management layer has its own catalog of backup files that maps a file to tape volumes and contains metadata about the backup files. Many media managers support automatic tape expiration. This can cause earlier versions of Recovery Manager's recovery catalog to become out of sync. In Oracle8*i*, Recovery Manager provides a means to re-synchronize the recovery catalog with the media management catalog.

MULTIPLEXED BACKUP SETS

A new command, SET DUPLEX, is used to specify the number of copies of output files to create. Each channel will then create N copies of each backup piece, where N is the value specified in the SET DUPLEX command. If a tag was specified, then all backup pieces share the same tag.

The value of N for SET DUPLEX can create up to four concurrent copies of each backup piece. This feature is particularly useful for archivelog backups. If a datafile backup is bad, an earlier backup of the same datafile can usually still be used for recovery, but if an archivelog backup is bad, then no recovery beyond the point in time when that log was created is possible. Because archivelogs are critical to recovery, customers should use the new Oracle8*i* capability supporting multiplexed archive logs. This prevents the archive logs from being a single point of failure.

This feature is most effective when used in conjunction with a third-party media manager that supports the Oracle8*i* media management interface, because the new interface guarantees that each copy will be written to separate physical media. This behavior is not guaranteed with a media manager that supports the Oracle8 media management interface.

CONTROL FILE CHARACTER SET SPECIFICATION

Starting with Oracle Version 8.0.5, the database character set ID is stored in the control file. This allows access to the database character set before a database open and is necessary for Recovery Manager to correctly interpret tablespace names prior to the database open if the character set happens to be something other than the default (US7ASCII). The CREATE CONTROLFILE syntax has been expanded to optionally accept the database character set name so that character set information is available at mount time.

STARTUP AND SHUTDOWN COMMANDS

STARTUP and SHUTDOWN commands have been added to Recovery Manager for starting up and shutting down the target database.

DUPLICATE COMMAND

The DUPLICATE command allows you to create a replicated database using the backups stored in the media manager from another database. This replicated database can be given a different DB_NAME or have the same DB_NAME as the one which the user just cloned. One possible use of a clone database would be to create a safe environment for testing backup and recovery procedures.

AUTOMATIC CATALOG MAINTENANCE

This feature allows recovery catalog creation and maintenance to be done via Recovery Manager commands instead of running separate SQL scripts. The new commands are: CREATE CATALOG, UPGRADE CATALOG, and DROP CATALOG.

RESTORE PREVIEW

You can use the RESTORE … VALIDATE command to check that backups and copies can be restored.