

---

# **Oracle9i DBA Fundamentals II**

**Student Guide • Volume 2**

---

D11297GC10  
Production 1.0  
May 2001  
D32715

**ORACLE®**

## **Authors**

Donna Keesling  
James Womack

## **Technical Contributors and Reviewers**

Lance Ashdown  
Tammy Bednar  
Louise Beijer  
Howard Bradley  
Senad Dizdar  
Joel Goodman  
Scott Gossett  
Stefan Lindblad  
Howard Ostrow  
Radhanes Petronilla  
Maria Jesus Senise Garcia  
Peter Sharman  
Ranbir Singh  
Harald Van Breederode  
John Watson  
Steven Wertheimer  
Junichi Yamazaki

## **Publisher**

John B Dawson

**Copyright © Oracle Corporation, 2000, 2001. All rights reserved.**

This documentation contains proprietary information of Oracle Corporation. It is provided under a license agreement containing restrictions on use and disclosure and is also protected by copyright law. Reverse engineering of the software is prohibited. If this documentation is delivered to a U.S. Government Agency of the Department of Defense, then it is delivered with Restricted Rights and the following legend is applicable:

### **Restricted Rights Legend**

Use, duplication or disclosure by the Government is subject to restrictions for commercial computer software and shall be deemed to be Restricted Rights software under Federal law, as set forth in subparagraph (c)(1)(ii) of DFARS 252.227-7013, Rights in Technical Data and Computer Software (October 1988).

This material or any portion of it may not be copied in any form or by any means without the express prior written permission of Oracle Corporation. Any other copying is a violation of copyright law and may result in civil and/or criminal penalties.

If this documentation is delivered to a U.S. Government Agency not within the Department of Defense, then it is delivered with "Restricted Rights," as defined in FAR 52.227-14, Rights in Data-General, including Alternate III (June 1987).

The information in this document is subject to change without notice. If you find any problems in the documentation, please report them in writing to Education Products, Oracle Corporation, 500 Oracle Parkway, Box SB-6, Redwood Shores, CA 94065. Oracle Corporation does not warrant that this document is error-free.

Oracle and all references to Oracle products are trademarks or registered trademarks of Oracle Corporation.

All other products or company names are used for identification purposes only, and may be trademarks of their respective owners.

# Contents

## 1 Networking Overview

- Objectives 1-2
- Network Environment Challenges 1-3
- Simple Network: Two-Tier 1-5
- Simple to Complex Network: N-Tier 1-6
- Complex Network 1-7
- Oracle9i Networking Solutions 1-8
- Connectivity: Oracle Net Services 1-9
- Connectivity: Database Connectivity With IIOP and HTTP 1-11
- Directory Naming 1-12
- Directory Services: Oracle Internet Directory 1-13
- Scalability: Oracle Shared Server 1-14
- Scalability: Connection Manager 1-15
- Security: Advanced Security 1-17
- Advanced Security Encryption 1-18
- Security: Oracle Net and Firewalls 1-19
- Accessibility: Heterogeneous Services 1-20
- Accessibility: External Procedures 1-21
- Summary 1-22

## 2 Basic Oracle Net Architecture

- Objectives 2-2
- Oracle Net Connections 2-3
- Client-Server Application Connection: No Middle-Tier 2-4
- Web Client Application Connections 2-6
- Web Client Application Connection: Java Application Client 2-7
- Web Client Application Connection: Java Applet Client 2-8
- Web Client Application Connection: Web Server Middle-Tier 2-9
- Web Client Application Connection: No Middle-Tier 2-10
- Summary 2-12

## 3 Basic Oracle Net Server-Side Configuration

- Objectives 3-2
- Overview: The Listener Process 3-3
- The Listener Responses 3-4
- Configuring the Listener 3-5
- Bequeath Session 3-7
- Redirect Session 3-9
- Static Service Registration: The listener.ora File 3-10
- Static Service Registration: Create a Listener 3-14
- Configure Services 3-15
- Logging and Tracing 3-16
- Dynamic Service Registration: Configure Registration 3-17
- Dynamic Service Registration: Configure PMON 3-18
- Configure the Listener for Oracle9i JVM: IIOP and HTTP 3-19

- Listener Control Utility (LSNRCTL) 3-21
- LSNRCTL Commands 3-22
- LSNRCTL SET and SHOW Modifiers 3-24
- Summary 3-26
- Practice 3 Overview 3-27

## **4 Basic Oracle Net Services Client-Side Configuration**

- Objectives 4-2
- Host Naming 4-3
  - Host Naming Client Side 4-4
  - Host Naming Server Side 4-5
  - Select Host Name Method 4-6
  - Host Naming Method 4-7
- Local Naming 4-8
- Oracle Net Configuration Assistant 4-9
- Choosing Local Naming 4-10
- Configuring Local Net Service Names 4-11
- Working with Net Service Names 4-12
- Specify the Oracle Database Version 4-13
- Database Service Name 4-14
- Network Protocol 4-15
- Host Name and Listener Port 4-16
- Testing the Connection 4-17
- Connection Test Result 4-18
- Net Service Name 4-19
- Save the Net Service Name 4-20
- tnsnames.ora 4-21
- sqlnet.ora 4-22
- Troubleshooting the Client Side 4-23
- Summary 4-25
- Practice 4 Overview 4-26

## **5 Usage and Configuration of the Oracle Shared Server**

- Objectives 5-2
- Server Configurations 5-3
  - Dedicated Server Processes 5-4
  - Oracle Shared Server 5-5
  - Benefits of Oracle Shared Server 5-7
  - Connecting 5-9
  - Processing a Request 5-10
  - The SGA and PGA 5-12
  - Configuring Oracle Shared Server 5-13
- DISPATCHERS 5-14
- SHARED\_SERVERS 5-16
- MAX\_DISPATCHERS 5-18

- MAX\_SHARED\_SERVERS 5-20
- CIRCUITS 5-21
- SHARED\_SERVER\_SESSIONS 5-22
- Related Parameters 5-23
- Verifying Setup 5-24
- Data Dictionary Views 5-26
- Summary 5-27
- Practice 5 Overview 5-28

## **6 Backup and Recovery Overview**

- Objectives 6-2
- Backup and Recovery Issues 6-3
- Categories of Failures 6-4
- Causes of Statement Failures 6-5
- Resolutions for Statement Failures 6-6
- Causes of User Process Failures 6-7
- Resolution of User Process Failures 6-8
- Possible User Errors 6-9
- Resolution of User Errors 6-10
- Causes of Instance Failure 6-11
- Recovery from Instance Failure 6-12
- Causes of Media Failures 6-14
- Resolutions for Media Failures 6-15
- Defining a Backup and Recovery Strategy 6-16
- Business Requirements 6-17
- Operational Requirements 6-18
- Technical Considerations 6-20
- Disaster Recovery Issues 6-22
- Summary 6-24

## **7 Instance and Media Recovery Structures**

- Objectives 7-2
- Overview 7-3
- Large Pool 7-6
- Database Buffer Cache, DBWn, and Datafiles 7-8
- Redo Log Buffer, LGWR, and Redo Log Files 7-10
- Multiplexed Redo Log Files 7-13
- CKPT Process 7-15
- Multiplexed Control Files 7-17
- ARCn Process and Archived Log Files 7-19
- Database Synchronization 7-21
- Phases for Instance Recovery 7-22
- Tuning Instance Recovery Performance 7-24
- Tuning the Duration of Instance and Crash Recovery 7-25

- Initialization Parameters Influencing Checkpoints 7-26
- Tuning the Phases of Instance Recovery 7-28
- Tuning the Rolling Forward Phase 7-29
- Tuning the Rolling Back Phase 7-30
- Fast-Start On-Demand Rollback 7-31
- Fast-Start Parallel Rollback 7-32
- Controlling Fast-Start Parallel Rollback 7-33
- Monitoring Parallel Rollback 7-34
- Summary 7-35
- Practice 7 Overview 7-36

## **8 Configuring the Database Archiving Mode**

- Objectives 8-2
- Redo Log History 8-3
- Noarchivelog Mode 8-4
- Archivelog Mode 8-6
- Changing the Archiving Mode 8-8
- Automatic and Manual Archiving 8-10
- Specifying Multiple ARCn Processes 8-12
- Stop or Start Additional Archive Processes 8-13
- Enabling Automatic Archiving at Instance Startup 8-14
- Enabling Automatic Archiving After Instance Startup 8-15
- Disabling Automatic Archiving 8-16
- Manually Archiving Online Redo Log Files 8-17
- Specifying the Archive Log Destination 8-19
- Specifying Multiple Archive Log Destinations 8-20
- LOG\_ARCHIVE\_DEST\_n Options 8-21
- Specifying a Minimum Number of Local Destinations 8-22
- Controlling Archiving to a Destination 8-24
- Specifying the File Name Format 8-25
- Obtaining Archive Log Information 8-26
- Summary 8-29
- Practice 8 Overview 8-30

## **9 Oracle Recovery Manager Overview and Configuration**

- Objectives 9-2
- Recovery Manager Features 9-3
- Recovery Manager Components 9-5
- RMAN Repository: Using the Control File 9-7
- Channel Allocation 9-8
- Manual Channel Allocation 9-10
- Automatic Channel Allocation 9-12
- Media Management 9-13
- Types of Connections with RMAN 9-15
- Connecting Without a Recovery Catalog 9-16

- Recovery Manager Modes 9-18
- RMAN Commands 9-20
- RMAN Configuration Settings 9-22
- The CONFIGURE Command 9-23
- The SHOW Command 9-25
- LIST Command Operations 9-26
- The LIST Command 9-27
- The REPORT Command 9-28
- The REPORT NEED BACKUP Command 9-29
- Recovery Manager Packages 9-30
- RMAN Usage Considerations 9-31
- Summary 9-33
- Practice 9 Overview 9-34

## **10 User-Managed Backups**

- Objectives 10-2
- Terminology 10-3
- User-Managed Backup and Recovery 10-5
- Querying Views to Obtain Database File Information 10-6
- Backup Methods 10-8
- Consistent Whole Database Backup (Closed Database Backup) 10-9
- Advantages of Making Consistent Whole Database Backups 10-10
- Making a Consistent Whole Database Backup 10-12
- Open Database Backup 10-14
- Advantages of Making Open Database Backups 10-15
- Open Database Backup Requirements 10-16
- Open Database Backup Options 10-17
- Making a Backup of an Online Tablespace 10-18
- Ending the Online Tablespace Backup 10-19
- Backup Status Information 10-20
- Failure During Online Tablespace Backup 10-22
- Read-Only Tablespace Backup 10-24
- Read-Only Tablespace Backup Issues 10-25
- Backup Issues with Logging and Nologging Options 10-26
- Manual Control File Backups 10-27
- Backing Up the Initialization Parameter File 10-29
- Verifying Backups Using the DBVERIFY Utility 10-30
- DBVERIFY Command-Line Interface 10-31
- Summary 10-33
- Practice 10 Overview 10-34

## **11 RMAN Backups**

- Objectives 11-2
- RMAN Backup Concepts 11-3
- Recovery Manager Backups 11-4

Backup Sets	11-5
Characteristics of Backup Sets	11-6
Backup Piece	11-7
The BACKUP Command	11-8
Backup Piece Size	11-11
Parallelization of Backup Sets	11-12
Multiplexed Backup Sets	11-15
Duplexed Backup Sets	11-16
Backups of Backup Sets	11-17
Archived Redo Log File Backups	11-18
Archived Redo Log Backup Sets	11-19
Datafile Backup Set Processing	11-20
Backup Constraints	11-21
Image Copies	11-22
Characteristics of an Image Copy	11-23
Image Copies	11-24
The COPY Command	11-25
Image Copy Parallelization	11-26
Copying the Whole Database	11-27
Making Incremental Backups	11-28
Differential Incremental Backup Example	11-29
Cumulative Incremental Backup Example	11-31
Backup in Noarchivelog Mode	11-32
RMAN Control File Autobackups	11-33
Tags for Backups and Image Copies	11-34
RMAN Dynamic Views	11-35
Monitoring RMAN Backups	11-36
Miscellaneous RMAN Issues	11-38
Summary	11-40
Practice 11 Overview	11-41

## **12 User-Managed Complete Recovery**

Objectives	12-2
Media Recovery	12-3
Recovery Steps	12-4
Restoration and Datafile Media Recovery with User-Managed Procedures	12-5
Archivelog and Noarchivelog Modes	12-6
Recovery in Noarchivelog Mode	12-7
Recovery in Noarchivelog Mode With Redo Log File Backups	12-9
Recovery in Noarchivelog Mode Without Redo Log File Backups	12-10
Recovery in Archivelog Mode	12-11
Complete Recovery	12-12
Complete Recovery in Archivelog Mode	12-13
Determining Which Files Need Recovery	12-14



- User-Managed Recovery Procedures: RECOVER Command 12-16
- Using Archived Redo Log Files During Recovery 12-17
- Restoring Datafiles to a New Location with User-Managed Procedures 12-19
- Complete Recovery Methods 12-20
- Complete Recovery of a Closed Database 12-22
- Closed Database Recovery Example 12-23
- Open Database Recovery When the Database Is Initially Open 12-25
- Open Database Recovery Example 12-26
- Open Database Recovery When the Database Is Initially Closed 12-28
- Open Database Recovery Example 12-29
- Recovery of a Datafile Without a Backup 12-32
- Recovery Without a Backup Example 12-33
- Read-Only Tablespace Recovery 12-35
- Read-Only Tablespace Recovery Issues 12-36
- Loss of Control Files 12-37
- Recovering Control Files 12-38
- Summary 12-39
- Practices 12-1 and 12-2 Overview 12-40

### **13 RMAN Complete Recovery**

- Objectives 13-2
- Restoration and Datafile Media Recovery Using RMAN 13-3
- Using RMAN to Recover a Database in Noarchivelog Mode 13-4
- Using RMAN to Recover a Database in Archivelog Mode 13-6
- Using RMAN to Restore Datafiles to a New Location 13-7
- Using RMAN to Recover a Tablespace 13-8
- Using RMAN to Relocate a Tablespace 13-9
- Summary 13-11
- Practices 13-1 and 13-2 Overview 13-12

### **14 User-Managed Incomplete Recovery**

- Objectives 14-2
- Incomplete Recovery Overview 14-3
- Reasons for Performing Incomplete Recovery 14-4
- Types of Incomplete Recovery 14-5
- Incomplete Recovery Guidelines 14-7
- Incomplete Recovery and the Alert Log 14-9
- User-Managed Procedures for Incomplete Recovery 14-10
- RECOVER Command Overview 14-11
- Time-Based Recovery Example 14-12
- UNTIL TIME Recovery 14-13
- Cancel-Based Recovery Example 14-15
- Using a Backup Control File During Recovery 14-18
- Loss of Current Redo Log Files 14-21
- Summary 14-23
- Practices 14-1 and 14-2 Overview 14-24

## **15 RMAN Incomplete Recovery**

- Objectives 15-2
- Incomplete Recovery of a Database Using RMAN 15-3
- RMAN Incomplete Recovery UNTIL TIME Example 15-4
- RMAN Incomplete Recovery UNTIL SEQUENCE Example 15-6
- Summary 15-7
- Practice 15 Overview 15-8

## **16 RMAN Maintenance**

- Objectives 16-2
- Cross Checking Backups and Copies 16-3
- The CROSSCHECK Command 16-4
- Deleting Backups and Copies 16-5
- The DELETE Command 16-6
- Deleting Backups and Copies 16-7
- Changing the Availability of RMAN Backups and Copies 16-8
- Changing the Status to Unavailable 16-9
- Exempting a Backup or Copy from the Retention Policy 16-10
- The CHANGE ... KEEP Command 16-11
- Cataloging Archived Redo Log Files and User-Managed Backups 16-12
- The CATALOG Command 16-13
- Uncataloging RMAN Records 16-14
- The CHANGE ... UNCATALOG Command 16-15
- Summary 16-16
- Practice 16 Overview 16-17

## **17 Recovery Catalog Creation and Maintenance**

- Objectives 17-2
- Overview 17-4
- Recovery Catalog Contents 17-5
- Benefits of Using a Recovery Catalog 17-7
- Additional Features Which Require the Recovery Catalog 17-8
- Create Recovery Catalog 17-9
- Connecting Using a Recovery Catalog 17-12
- Recovery Catalog Maintenance 17-13
- Resynchronization of the Recovery Catalog 17-14
- Using RESYNC CATALOG for Resynchronization 17-15
- Resetting a Database Incarnation 17-16
- Recovery Catalog Reporting 17-18
- Viewing the Recovery Catalog 17-19
- Stored Scripts 17-21
- Script Examples 17-22
- Managing Scripts 17-23
- Backup of Recovery Catalog 17-24

Recovering the Recovery Catalog 17-25

Summary 17-26

Practice 17 Overview 17-27

## **18 Transporting Data Between Databases**

Objectives 18-2

Oracle Export and Import Utility Overview 18-3

Methods to Run the Export Utility 18-5

Export Modes 18-6

Command-Line Export 18-7

Direct-Path Export Concepts 18-9

Specifying Direct-Path Export 18-10

Direct-Path Export Features 18-11

Direct-Path Export Restrictions 18-12

Uses of the Import Utility for Recovery 18-13

Import Modes 18-14

Command-Line Import 18-15

Invoking Import as SYSDBA 18-17

Import Process Sequence 18-18

National Language Support Considerations 18-19

Summary 18-20

Practice 18 Overview 18-21

## **19 Loading Data Into a Database**

Objectives 19-2

Data Loading Methods 19-3

Direct-Load INSERT 19-4

Serial Direct-Load Inserts 19-5

Parallel Direct-Load Insert 19-7

SQL\*Loader 19-8

Using SQL\*Loader 19-9

Conventional and Direct Path Loads 19-10

Comparing Direct and Conventional Path Loads 19-11

Parallel Direct-Path Load 19-12

SQL\*Loader Control File 19-13

Control File Syntax Considerations 19-16

Input Data and Datafiles 19-17

Logical Records 19-20

Data Conversion 19-21

Discarded or Rejected Records 19-22

Log File Contents 19-23

SQL\*Loader Guidelines 19-25

Summary 19-26

Practice 19 Overview 19-27

## **20 Workshop**

- Objectives 20-2
- Workshop Methodology 20-4
- Workshop Approach 20-6
- Business Requirements 20-7
- Resolving a Database Failure 20-8
- Troubleshooting Methods 20-10
- Enable Tracing 20-11
- Using Trace Files 20-12
- Resolving a Network Failure 20-14
- Summary 20-16

## **Appendix A: Practice Solutions**

## **Appendix B: Workshop Scenarios**

## **Appendix C: Worldwide Support Bulletins**

# 12

## User-Managed Complete Recovery

ORACLE

Copyright © Oracle Corporation, 2001. All rights reserved.

# Objectives

**After completing this lesson, you should be able to do the following:**

- **Describe media recovery**
- **Perform recovery in Noarchivelog mode**
- **Perform complete recovery in Archivelog mode**
- **Restore datafiles to different locations**
- **Relocate and recover a tablespace by using archived redo log files**
- **Describe read-only tablespace recovery**

ORACLE

# Media Recovery

- **Used to recover a lost or damaged current datafile or control file**
- **Requires explicit invocation**
- **Operates as follows:**
  - **Files are restored from backups**
  - **Redo data is applied to the restored files from archived redo log files and online redo logs**

ORACLE

12-3

Copyright © Oracle Corporation, 2001. All rights reserved.

## Media Recovery

Media recovery is used to recover a lost or damaged current datafile or control file. You can also use it to recover changes that were lost when a datafile went offline without the `OFFLINE NORMAL` option.

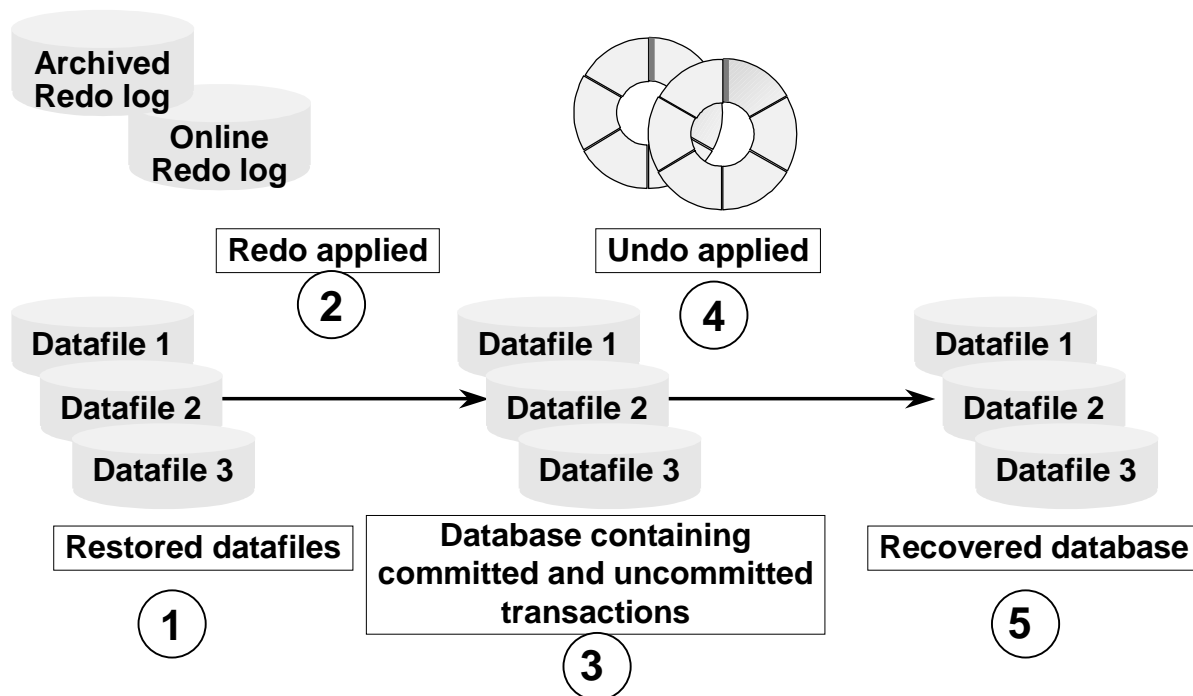
### Restoring Files

When you restore a file, you are replacing a missing or damaged file with a backup copy.

### Recovery of Files

When you recover a file, changes recorded in the redo log files are applied to the restored files.

## Recovery Steps



ORACLE

12-4

Copyright © Oracle Corporation, 2001. All rights reserved.

### Recovery Steps

1. Damaged or missing files are restored from a backup.
2. Changes from the archived redo log files and online redo log files are applied as necessary. Undo blocks are generated at this time. This is referred to as rolling forward or cache recovery.
3. The database may now contain committed and uncommitted changes.
4. The undo blocks are used to roll back any uncommitted changes. This is known as rolling back or transaction recovery.
5. The database is now in a recovered state.



## **Restoration and Datafile Media Recovery with User-Managed Procedures**

- **Restore files using operating system commands**
- **Recover files using the SQL\*Plus RECOVER command**

ORACLE

12-5

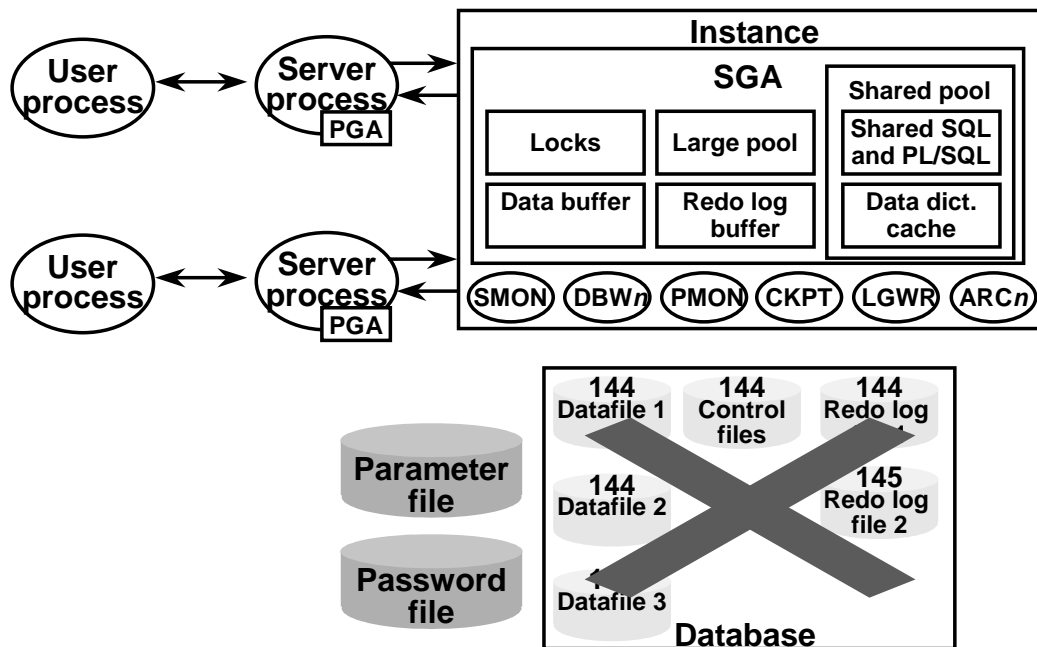
Copyright © Oracle Corporation, 2001. All rights reserved.

### **Restoration and Datafile Media Recovery With User-Managed Procedures**

When you restore a file, you use operating system commands to copy the file from a backup. You can restore datafiles, control files, archived redo log files, and the server parameter file.

You use the SQL\*Plus RECOVER command to apply redo log files to the restored files. You can perform automatic recovery or step through the log files to apply the changes.

# Archivelog and Noarchivelog Modes



ORACLE

12-6

Copyright © Oracle Corporation, 2001. All rights reserved.

## Contrasting Archivelog and Noarchivelog Modes

The Archive mode you choose to operate your database affects your options for recovery if you have a media failure.

The following factors should be considered in setting the archive log mode.

- Noarchivelog mode may be suitable when:
  - Data loss between backups can be tolerated (during development, training, etc.)
  - It is faster to reapply transactions (from batch files)
  - Data rarely changes (non-OLTP)
- Archivelog mode is preferable when:
  - The database cannot be shut down for a closed backup
  - Data loss cannot be tolerated
  - It is easier to recover using archived redo log files than reapplying transactions (OLTP)
- By default, the database is in Noarchivelog mode.

# Recovery in Noarchivelog Mode

**In Noarchivelog mode, you must restore the following database files:**

- **Datafiles**
- **Control files**

**You may also restore the following files:**

- **Redo log files**
- **Password file**
- **Parameter file**

ORACLE

12-7

Copyright © Oracle Corporation, 2001. All rights reserved.

## Recovery When the Database is in Noarchivelog Mode

When media failure occurs in a database operating in Noarchivelog mode, you need a valid closed database backup to recover. In Noarchivelog mode, all Oracle database files must be restored, even if only one datafile is damaged or lost. Make sure you restore all of the following files: datafiles and control files. If you took a backup of the online redo log files when you backed up the datafile and control files, you can also restore those files. Remember, all Oracle database files must be synchronized for the database to open.

You should restore the password and parameter files only if they are corrupt or lost.

**Note:** For a database in Noarchivelog mode, you do not have to restore all Oracle files if no redo log file has been overwritten since the last backup, as illustrated in the following:

- **Scenario**
  - There are two redo logs for a database.
  - A closed database backup was taken at log sequence 144.
  - While the database was at log sequence 145, data file 2 was lost.

- **Result**

Because log sequence 144 has not been overwritten, datafile number 2 can be restored and recovered manually.

# Recovery in Noarchivelog Mode

- **Advantages**
  - Easy to perform, with low risk of error
  - Recovery time is the time it takes to restore all files.
- **Disadvantages**
  - Data is lost and must be reapplied manually.
  - The entire database is restored to the point of the last whole closed backup.

ORACLE

12-8

Copyright © Oracle Corporation, 2001. All rights reserved.

## Recovery in Noarchivelog Mode: Advantages and Disadvantages

If you decide to operate a database in Noarchivelog mode, consider the following advantages and disadvantages with respect to recovery:

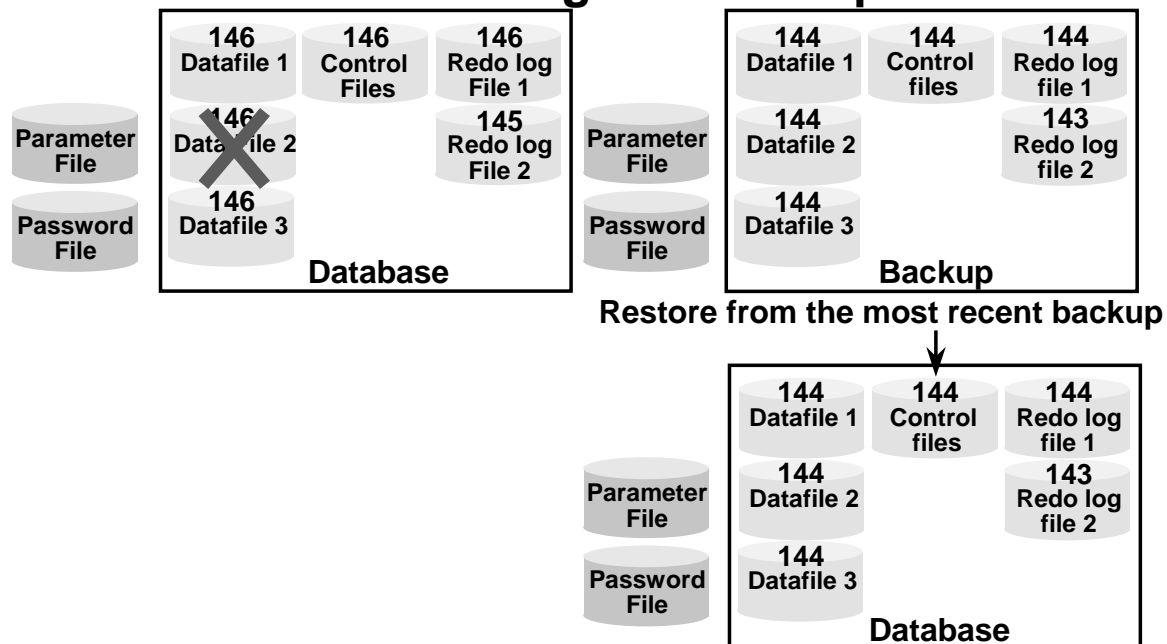
### Advantages

- Easy to perform, because only a restore of all files from a backup is required. The only risks are operational in nature: restoring the wrong backup, overwriting the backup, not shutting down the database before restore, or making invalid backups. Adequate training and tested procedures can help to minimize these risks.
- The time taken for recovery is merely the length of time it takes you to restore all files, given your hardware and operating system.

### Disadvantages

- All data entered by users since the last backup will be lost and must be reapplied manually.
- The entire database must be restored from the last whole closed backup, even if only one data file is lost.

## Recovery in Noarchivelog Mode With Redo Log File Backups



ORACLE

12-9

Copyright © Oracle Corporation, 2001. All rights reserved.

### User-Managed Recovery in Noarchivelog Mode

#### Example

Disk 2 is damaged, losing datafile number 2. Only two online redo log files exist.

The last backup was taken at log sequence 144 and the current log sequence number is 146.

You cannot recover the datafile, because redo log 144 has been overwritten. This would be confirmed if recovery was attempted. Therefore, you must shut down the database and restore all Oracle files.

```
SQL> SHUTDOWN ABORT;
```

To restore files:

```
Unix: cp /BACKUP/* /databases/db01/ORADATA
```

```
NT: copy d:\disk1\backup\*. * d:\disk1\data\
```

When the copy is finished, restart the instance:

```
SQL> CONNECT / as sysdba;
```

```
SQL> STARTUP;
```

Notify users that they will need to reenter data from the time of the last backup.

## Recovery in Noarchivelog Mode Without Redo Log File Backups

1. Shut down the instance.
2. Restore the datafiles and the control file from the most recent whole database backup.
3. Perform cancel-based recovery.
4. Open the database with the **RESETLOGS** option.

ORACLE

12-10

Copyright © Oracle Corporation, 2001. All rights reserved.

### Recovery Without Redo Log File Backups

1. If the database is open, then shut down the database as follows:  

```
SQL> SHUTDOWN IMMEDIATE
```
2. Restore the most recent whole database backup with operating system commands. You must restore all of the datafiles and control files, not just the damaged files. The following example restores a whole database backup:  

```
$ cp /db01/BACKUP/*.dbf /ORADATA/u*/* # restores datafiles  
$ cp /db01/BACKUP/*.ctl /ORADATA/u*/* # restores control file
```
3. Because you did not back up the online redo logs, you cannot restore them with the datafiles and control files. So that Oracle can reset the online redo logs, you must mimic incomplete recovery as follows:  

```
SQL> RECOVER DATABASE UNTIL CANCEL  
SQL> CANCEL
```
4. Then open the database with the **RESETLOGS** option to reset the current redo log sequence to 1 as follows:  

```
SQL> ALTER DATABASE OPEN RESETLOGS ;
```

# Recovery in Archivelog Mode

- **Complete Recovery**
  - Uses redo data or incremental backups
  - Updates the database to the most current point in time
  - Applies all redo changes
- **Incomplete Recovery**
  - Uses backup and redo logs to produce a noncurrent version of the database

ORACLE

12-11

Copyright © Oracle Corporation, 2001. All rights reserved.

## Recovery in Archivelog Mode

### Comparison of Complete and Incomplete Recovery

When you perform media recovery, you update the restored files to the current, or to a user-specified noncurrent, time.

In complete recovery, you use the redo log files or incremental backups to update the restored files to the most current point in time. All of the redo changes contained in the archived and online redo log files are applied to the files. You can perform complete recovery on a database, tablespace, or datafile.

With incomplete recovery, you are recovering the database to a point in time before the current time. Usually, you do not apply all of the redo entries which have been generated since the backup you used for the restore was taken.

# Complete Recovery

- **Make sure that datafiles for restore are offline.**
- **Restore only lost or damaged datafiles.**
- **Do not restore the control files, redo log files, password files, or parameter files.**
- **Recover the datafiles.**

ORACLE

12-12

Copyright © Oracle Corporation, 2001. All rights reserved.

## Complete Recovery When the Database is in Archivelog Mode

When media failure occurs in a database operating in Archivelog mode, you need the following to recover completely up to the time of failure:

- A valid backup containing the lost or damaged datafiles taken after the database was set in Archivelog mode
- All archived logs from the time of the backup that you are using through the present time
- The redo log files that contain transactions not yet archived



# Complete Recovery in Archivelog Mode

- **Advantages**
  - Only need to restore lost files
  - Recovers all data to the time of failure
  - Recovery time is the time it takes to restore lost files and apply all archived log files
- **Disadvantages**
  - Must have all archived log files since the backup from which you are restoring

ORACLE

12-13

Copyright © Oracle Corporation, 2001. All rights reserved.

## Recovery in Archivelog Mode: Advantages and Disadvantages

### Advantages

- Only need to restore lost or damaged files.
- No committed data is lost. Restoring the files, then applying archived and redo logs, brings the database to the current point in time.
- The total recovery time is the length of time required to restore the files and apply all archived and redo logs.
- Recovery can be performed while the database is open (except system tablespace files and datafiles that contain online rollback segments).

### Disadvantages

You must have all archived redo log files from the time of your last backup to the current time. If you are missing one, you cannot perform a complete recovery, because all archived redo log files must be applied in sequence; that is, archived log 144, then 145, then 146, and so on.

## Determining Which Files Need Recovery

- View **V\$RECOVER\_FILE** to determine which datafiles need recovery
- View **V\$ARCHIVED\_LOG** for a list of all archived redo log files for the database
- View **V\$RECOVERY\_LOG** for a list of all archived redo log files required for recovery

ORACLE

12-14

Copyright © Oracle Corporation, 2001. All rights reserved.

### Determining Which Files Need Recovery

#### Identifying Datafiles That Need Recovery

To identify datafiles needing recovery, and from where recovery needs to start, use the **V\$RECOVER\_FILE** view as follows:

```
SQL> SELECT * FROM v$recover_file;
```

FILE#	ONLINE	ERROR	CHANGE#	TIME
2	OFFLINE		288772	02-MAR-01

The **ERROR** column returns two possible values to define the reason why the file needs to be recovered:

- **NULL** if the reason is unknown
- **OFFLINE NORMAL** if recovery is not needed

The **CHANGE#** column returns the SCN (system change number) from where recovery must start.

## Determining Which Files Need Recovery (continued)

### Locating Archived Log Files to Apply

To locate archived log files, view V\$ARCHIVED\_LOG for all archived log files or V\$RECOVERY\_LOG for archived log files needed during recovery:

```
SQL> SELECT * FROM v$recovery_log;
```

THREAD#	SEQUENCE#	TIME	ARCHIVE_NAME
-----	-----	-----	-----
1	34	02-MAR-01	/.../ORADATA/ARCHIVE1/arch_34.arc
...			
1	43	04-MAR-01	/.../ORADATA/ARCHIVE1/arch_43.arc
1	44	04-MAR-01	/.../ORADATA/ARCHIVE1/arch_44.arc

From the above information, archived logs from 34 on are required to recover datafile 2 completely.

**Note:** V\$RECOVERY\_LOG contains useful information only for the Oracle process doing the recovery.

V\$ARCHIVED\_LOG displays archived log information from the control file, including archive log names.

# User-Managed Recovery Procedures: RECOVER Command

## Recover a mounted database:

```
SQL> RECOVER DATABASE;  
OR  
SQL> RECOVER DATAFILE  
2> '/ORADATA/u03/users01.dbf';
```

## Recover an open database:

```
SQL> RECOVER TABLESPACE users;  
OR  
SQL> RECOVER DATAFILE 3;
```

ORACLE

12-16

Copyright © Oracle Corporation, 2001. All rights reserved.

## RECOVER Commands

One of the following commands can be issued to recover the database:

- RECOVER [AUTOMATIC] DATABASE

This command can only be used for a closed database recovery.

- RECOVER [AUTOMATIC] TABLESPACE <NUMBER> | <NAME>

This command can only be used for an open database recovery.

- RECOVER [AUTOMATIC] DATAFILE <NUMBER> | <NAME>

This command can be used for both an open and closed database recovery.

**where:** automatic automatically applies archived and redo log files.

## Using Archived Redo Log Files During Recovery

- To change archive location, use the `ALTER SYSTEM ARCHIVE LOG...` command.
- To apply redo log files automatically:
  - Issue the `SET AUTORECOVERY ON` command before starting media recovery
  - Enter `auto` when prompted for an archived log file
  - Use the `RECOVER AUTOMATIC...` command.

ORACLE

12-17

Copyright © Oracle Corporation, 2001. All rights reserved.

### Using Archived Redo Log Files During Recovery

During recovery, the Oracle server can manually or automatically apply the necessary archived and online redo log files to reconstruct the data files. Before a redo log file is applied, the Oracle server suggests the log file name to apply.

#### Restoring Archives to a Different Location

If archived redo log files are not restored to the `LOG_ARCHIVE_DEST` directory, then the Oracle server needs to be notified before or during recovery, by one of the following methods:

- Specifying the location and name at the recover prompt:  
`Specify log: {<RET>=suggested | filename | AUTO | CANCEL}`
- Using the `ALTER SYSTEM ARCHIVE` command:  
`SQL> ALTER SYSTEM ARCHIVE LOG START TO <new location>;`
- Using the `RECOVER FROM <LOCATION>` command:  
`SQL> RECOVER FROM '<new location>' DATABASE;`

## Using Archived Redo Log Files During Recovery (continued)

### How to Apply Redo Log Files Automatically

You can automatically apply redo log files in the following ways:

- Before starting media recovery, issue the SQL\*Plus statement:

```
SQL> SET AUTORECOVERY ON
```

- Enter auto when prompted for a redo log file:

```
SQL> RECOVER datafile 4;
```

```
ORA-00279: change 308810...03/22/01 17:00:14 needed for  
thread 1
```

```
ORA-00289: suggestion : /ORADATA/ARCHIVE1/arch_35.arc
```

```
ORA-00280: change 308810 for thread 1 is in sequence #35
```

```
Specify log: {<RET>=suggested | filename | AUTO | CANCEL}
```

```
AUTO
```

```
Log applied.
```

```
...
```

- Use the AUTOMATIC option of the RECOVER command:

```
SQL> RECOVER AUTOMATIC datafile 4;
```

```
Media recovery complete.
```

## Restoring Datafiles to a New Location with User-Managed Procedures

- Use operating system commands to restore the datafile to the new location.
- Use the `ALTER DATABASE RENAME FILE` command to record the change in the control file.

ORACLE

12-19

Copyright © Oracle Corporation, 2001. All rights reserved.

### User-Managed Procedures for Restoring Datafiles to a New Location

1. Use operating system commands to restore the file to the new location.

**Note:** In the UNIX environment, the files must exist in the new location prior to issuing the `ALTER DATABASE RENAME` command. This is not the case in an NT environment.

2. Start up the instance and mount the database.
3. Use the `ALTER DATABASE` command to update the control file with the new file name or location:

```
SQL> ALTER DATABASE RENAME FILE
      2>          '/ORADATA/u03/users01.dbf'
      3> to      '/ORADATA/u04/users01.dbf';
```

# Complete Recovery Methods

- **Closed database recovery for:**
  - **System datafiles**
  - **Undo segment datafiles**
  - **Whole database**
- **Open database recovery, with database initially opened (for file loss)**
- **Open database recovery with database initially closed (for hardware failure)**
- **Datafile recovery with no datafile backup**

ORACLE

12-20

Copyright © Oracle Corporation, 2001. All rights reserved.

## Complete Recovery Methods

There are four methods for performing complete recovery:

### Method 1: Recovering a Closed Database

Use this method when:

- The database is not operational 24 hours a day, 7 days a week.
- The recovered files belong to the system or undo segment tablespace.
- The whole database, or a majority of the datafiles, need recovery.

### Method 2: Recovering an Open Database, Initially Opened

This method of recovery is generally used when:

- File corruption, accidental loss of file, or media failure has occurred, which has not resulted in the database being shut down.
- The database is operational 24 hours a day, 7 days a week. Down time for the database must be kept to a minimum.
- Recovered files do not belong to the system or undo segment tablespaces.



## **Complete Recovery Methods (continued)**

### **Method 3: Recovering an Open Database, Initially Closed**

This method of recovery is generally used when:

- A media or hardware failure has shut the system down.
- The database is operational 24 hours a day, 7 days a week. Down time for the database must be kept to a minimum.
- The restored files do not belong to the system or undo segment tablespace.

### **Method 4: Recovering a Datafile with No Backup**

This method of recovery is generally used when:

- Media or user failure has resulted in loss of a datafile that was never backed up.
- All archived logs exist since the file was created.
- The restored files do not belong to the system or undo segment tablespace.

**Note:** During recovery, all archived logs files must be available to the Oracle server on disk. If they are on a backup tape, you must restore them first.

# Complete Recovery of a Closed Database

**Closed database recovery is used for:**

- **System tablespace datafiles**
- **Rollback segment datafiles**
- **Whole database**

ORACLE

12-22

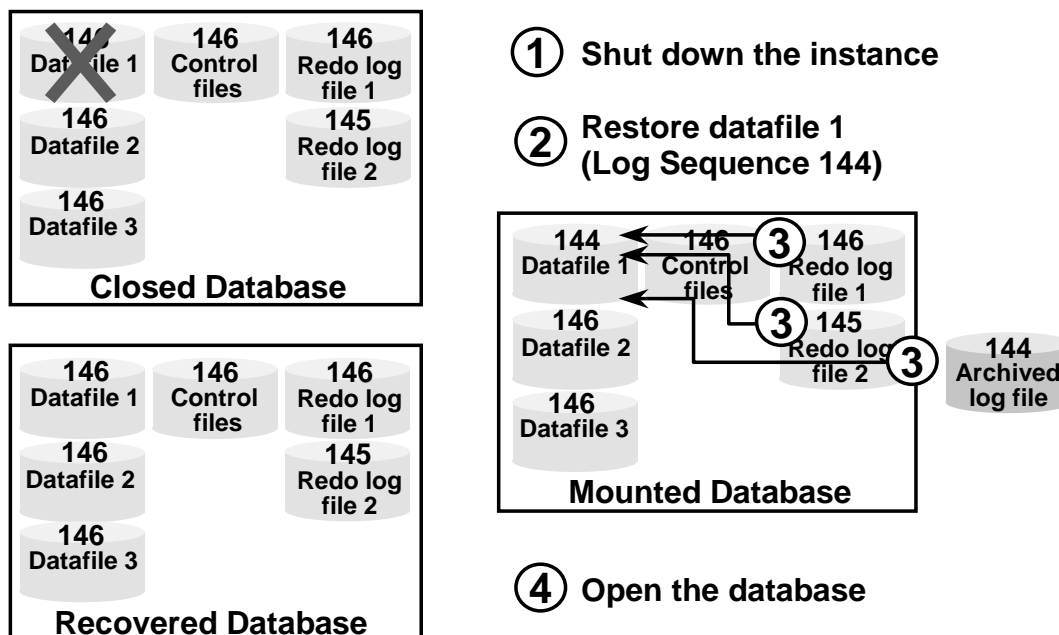
Copyright © Oracle Corporation, 2001. All rights reserved.

## **Closed Database Recovery**

You generally use this method of recovery with either the `RECOVER DATABASE` or `RECOVER DATAFILE` commands when:

- The recovered files belong to the system or rollback segment tablespace.
- The whole database, or a majority of the datafiles, need recovery.
- The database is not operational 24 hours a day, 7 days a week.

## Closed Database Recovery Example



ORACLE

12-23

Copyright © Oracle Corporation, 2001. All rights reserved.

### Closed Database Recovery Example

You have determined that u01 contains corrupt blocks. This is where datafile 1 is stored. By querying V\$DATAFILE and V\$TABLESPACE views, you discover that datafile 1 is one of the files belonging to the system tablespace. Proceed as follows to recover the database:

1. If the instance is not already shut down, issue the SHUTDOWN command as follows:

```
SQL> SHUTDOWN ABORT;
```

2. Restore the file from backup (the most recent if available):

```
UNIX> host cp /BACKUP/system01.dbf /ORADATA/u01
```

```
NT > host copy c:\backup\df1.dbf d:\data\
```

3. Start the instance in Mount mode and recover the data file:

```
SQL> STARTUP MOUNT;
```

```
SQL> RECOVER DATABASE;
```

```
or SQL> RECOVER DATAFILE '/ORADATA/u01/system01.dbf';
```

```
ORA-00279: change 148448 ...03/29/01 17:04:20 needed for thread
```

```
ORA-00289: suggestion : /ORADATA/ARCHIVE1/arch_144.arc
```

```
ORA-00280: change 148448 for thread 1 is in sequence #144
Log applied.
```

```
...
```

```
Media recovery complete.
```

### **Closed Database Recovery Example (continued)**

To bring the data file to the point of failure, all needed archived logs and redo logs are applied.

4. When recovery is finished, all data files are synchronized. Open the database.

```
SQL> ALTER DATABASE OPEN;
```

You can now notify users that the database is available for use, and tell them to reenter any data that was not committed before system failure.

**Note:** During this method of recovery, the database must be closed; the entire database is inaccessible to users during the recovery process.

## **Open Database Recovery When the Database Is Initially Open**

**Use this method when:**

- **The database is currently open**
- **The database will remain open during the recovery**
- **The media failure does not affect the `SYSTEM` tablespace**

**ORACLE**

12-25

Copyright © Oracle Corporation, 2001. All rights reserved.

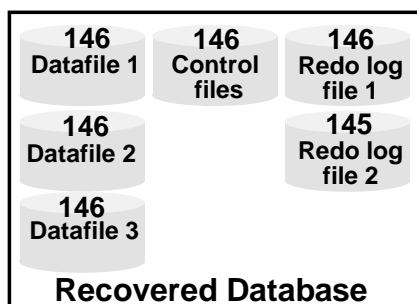
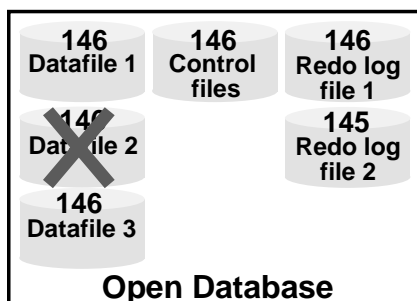
### **Open Database Recovery**

#### **Recovering an Open Database When It Is Initially Open**

This method of recovery is generally used when:

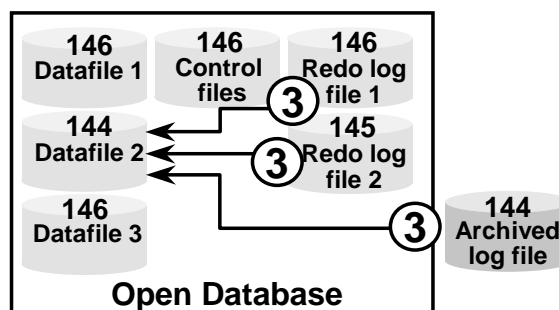
- File corruption, accidental loss of file, or media failure has occurred, which has not resulted in the database being shut down.
- The database is operational 24 hours a day, 7 days a week. Downtime for the database must be kept to a minimum.
- Affected files do not belong to the system or rollback tablespaces.

# Open Database Recovery Example



① Take datafile 2 offline

② Restore datafile 2  
(Log Sequence 144)



④ Bring datafile 2 online

## Open Database Recovery Example

Your database is currently open and datafile number 2 has accidentally been removed using operating system commands. Because the database is currently open, you can use the following command to determine which tablespace the datafile belongs to:

```
SQL > SELECT file_id f#, file_name,
```

```
2> tablespace_name tablespace, status
```

```
3> FROM dba_data_files;
```

F#	FILE_NAME	TABLESPACE	STATUS
1	/disk1/data/system01.dbf	SYSTEM	AVAILABLE
2	/disk2/data/df2.dbf	USER_DATA	AVAILABLE
3	/disk1/data/rbs01.dbf	RBS	AVAILABLE

...

## Open Database Recovery Example (continued)

1. Issue the following query to determine if you need to take datafile 2 offline:

```
SQL> SELECT d.file# f#, d.name, d.status, h.status
2  FROM v$datafile d, v$datafile_header h
3  WHERE d.file# = h.file#;
```

F#	D.NAME	D.STATUS	H.STATUS
1	/disk1/data/system01.dbf	SYSTEM	ONLINE
2	/disk2/data/df2.dbf	RECOVER	OFFLINE
3	/disk1/data/rbs_01.dbf	ONLINE	ONLINE
...			

In this case, the Oracle server has already taken the file offline.

2. Because the file is offline, you can now restore it from a backup:

```
for UNIX > host cp /disk1/backup/df2.dbf /disk2/data/
for NT > host copy c:\backup\df2.dbf d:\data\
```

3. Use the RECOVER or ALTER DATABASE RECOVER commands to apply the archives and the redo logs to the restored data file.

```
SQL> recover datafile '/disk2/backup/df2.dbf';
or SQL> recover tablespace USER_DATA;
```

4. When recovery is finished, all datafiles are synchronized. Bring the data file online:

```
SQL> alter database datafile '/disk2/data/df2.dbf' online;
or SQL> alter tablespace USER_DATA online;
```

### Note

- The Oracle server sometimes detects a file problem and automatically takes it offline. Before recovery, always check the alert log for any errors and check the status of files; offline files may require recovery.
- When a tablespace is taken offline, all datafiles are taken offline and no data contained in that tablespace can be accessed. For a multifile tablespace, when one datafile is taken offline, only data contained inside that datafile cannot be accessed. The tablespace still remains available.

## **Open Database Recovery When the Database Is Initially Closed**

**Use this method when:**

- **The database is currently closed**
- **The database will be opened during recovery**
- **The media failure does not affect the `SYSTEM` tablespace**

**ORACLE**

**12-28**

Copyright © Oracle Corporation, 2001. All rights reserved.

### **Open Database Recovery**

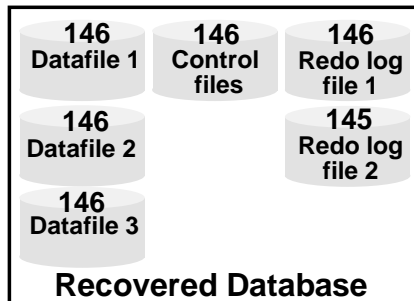
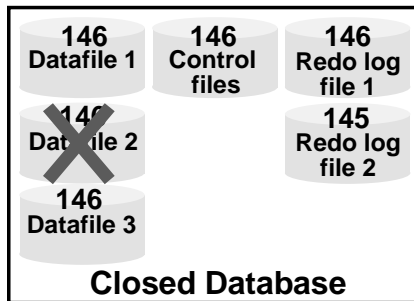
#### **Recovering an Open Database When It Is Initially Closed**

This method of recovery is generally used when:

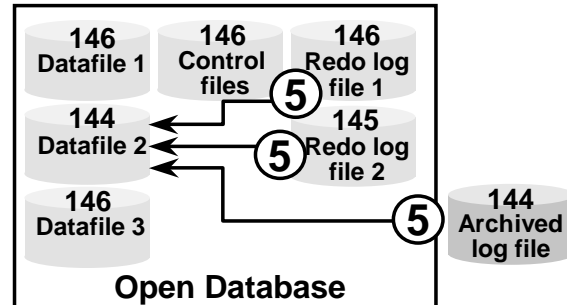
- A media or hardware failure has brought the system down.
- The database is operational 24 hours a day, 7 days a week. Downtime for the database must be kept to a minimum.
- The damaged files do not belong to the system or undo segment tablespace.



# Open Database Recovery Example



- ① Mount the database
- ② Take datafile 2 offline
- ③ Open the database
- ④ Restore datafile 2



- ⑥ Bring datafile 2 online

## Open Database Recovery Example

You have just determined that the media failure was due to a failed disk controller, which contains only disk 2. From your familiarity with the database, you know datafile 2 is not a system or rollback segment datafile, nor will its unavailability prevent users from running their end-of-month reports.

1. Mount the database. It will not open because datafile 2 cannot be opened.

```
SQL> STARTUP MOUNT
```

Database mounted.

If you are not sure of the tablespace number to which the file belongs, issue the following query:

```
SQL> SELECT d.file#, d.ts#, h.tablespace_name, d.name,
2>          h.error
3> FROM v$datafile d, v$datafile_header h
4> WHERE d.file# = h.file#;
```

## Open Database Recovery Example (continued)

FILE#	TS#	TABLES	NAME	Error
1	0	SYSTEM	/disk1/data/system01.dbf	
2	1		/disk2/data/df2.dbf	FILE NOT FOUND
3	2	RBS	/disk1/data/rbs01.dbf	

...

- If the datafile is not offline, the database will not open. Therefore, the file must be taken offline. You have queried V\$DATAFILE and determined that the file is online. The following command must be issued:

```
SQL> ALTER DATABASE datafile '/disk2/data/df2.dbf' offline;
```

**Note:** The ALTER TABLESPACE command cannot be used here because the database is not yet open.

- The database can now be opened so that users can access the system:

```
SQL> ALTER DATABASE OPEN;
```

- Now restore the file. Because it cannot be restored to the damaged disk 2, restore it to disk 3:

```
UNIX> host cp /disk1/backup/df2.dbf /disk3/data/
```

```
NT> host copy c:\backup\df2.dbf e:\data\
```

The Oracle server must now be informed of the new file location:

```
SQL> ALTER DATABASE rename file '/disk2/data/df2.dbf'
2> to '/disk3/data/df2.dbf';
```

When the database is opened and tablespace recovery is required, issue the following query to determine the name of the tablespace that owns the data file:

```
SQL> SELECT file_id f#, file_name,
2> tablespace_name tablespace, status
3> FROM dba_data_files;
```

F#	FILE_NAME	TABLESPACE	STATUS
1	/disk1/data/system_01.dbf	SYSTEM	AVAILABLE
2	/disk3/data/df2.dbf	USER_DATA	AVAILABLE
3	/disk1/data/rbs01.dbf	RBS	AVAILABLE

### **Open Database Recovery Example (continued)**

5. Use the RECOVER or ALTER DATABASE RECOVER command to start applying the archived redo log files and online redo log files to the restored datafile.

```
SQL> RECOVER DATAFILE '/disk3/data/df2.dbf';
```

```
or SQL> RECOVER TABLESPACE user_data;
```

6. When recovery is finished, all datafiles are synchronized. Bring the datafile online:

```
SQL> ALTER DATABASE datafile '/disk3/data/df2.dbf' online;
```

```
or SQL> ALTER TABLESPACE user_data online;
```

## Recovery of a Datafile Without a Backup

- Datafile is lost that was never backed up
- Cannot be used when it is a file from the **SYSTEM** tablespace

ORACLE

12-32

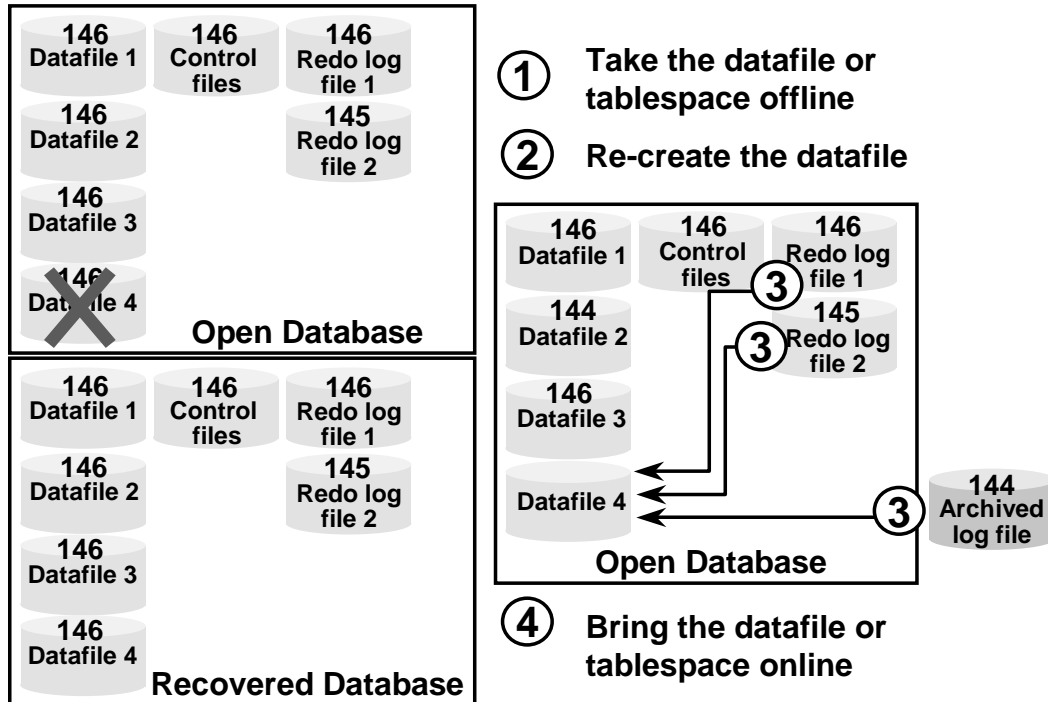
Copyright © Oracle Corporation, 2001. All rights reserved.

### Recovery of a Datafile Without a Backup

This method of recovery is generally used when:

- Media or user failure has resulted in the loss of a datafile that was never backed up.
- All archived logs exist since the file was created.
- The affected files do not belong to the system or undo segment tablespace.

## Recovery Without a Backup Example



### Recovery Without a Backup Example

Datafile 4 (on disk 1) is lost. When restoring the datafile from tape, you receive an error indicating that the file was not backed up. You locate the DBA who created the TABLE\_DATA tablespace two days ago and find that it contains important user data, but it was never included in the backup strategy. Because datafile 4 is not a system or rollback segment data file, and you have all archived logs for the past two days, you can proceed as follows:

1. If the database is closed, then mount the database, take the datafile (with no backup) offline, and open the database. This allows users who do not need the TABLE\_DATA tablespace to work on the system. If the database is open, take the datafile or the tablespace offline.

**Note:** If the database is open, the immediate option must be included, to avoid the database writer trying to write to a file that does not exist:

```
SQL> ALTER TABLESPACE table_data OFFLINE IMMEDIATE;
```

Tablespace altered.

## Recovery Without a Backup Example (continued)

You confirm the recovery status by querying V\$RECOVER\_FILE to check the status of a backup:

```
SQL> SELECT * FROM v$recover_file;
FILE# ONLINE  ERROR              CHANGE# TIME
-----
4      OFFLINE FILE NOT FOUND 0
```

2. You issue the following command to re-create the file:

```
SQL> ALTER DATABASE create datafile '/disk2/DATA/df4.dbf'
2>    as '/disk1/DATA/df4.dbf';
Database altered.
```

```
SQL> SELECT * FROM v$recover_file;
FILE# ONLINE  ERROR              CHANGE# TIME
-----
4      OFFLINE              248621 01-DEC-97
```

3. Use the RECOVER or ALTER DATABASE RECOVER commands to start applying the archived and online redo log files to the re-created datafile:

```
SQL> RECOVER TABLESPACE table_data;
```

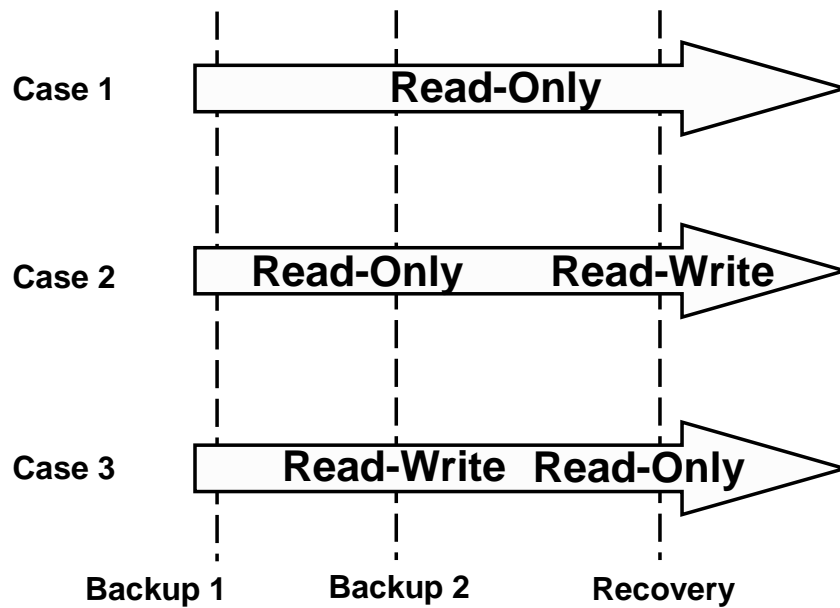
To bring the datafile to the point of failure, all needed archived logs and redo logs are applied. All datafiles are now synchronized.

4. When recovery is finished, bring the tablespace online:

```
SQL> ALTER TABLESPACE table_data ONLINE;
```

All data is now recovered. Include the file in the backup strategy and notify users that the tablespace is ready to be used again.

## Read-Only Tablespace Recovery



ORACLE

12-35

Copyright © Oracle Corporation, 2001. All rights reserved.

### Recovery Cases

**Case 1** The tablespace being recovered is read-only, and was read-only when the last backup occurred. In this case, you can simply restore the tablespace from the backup. There is no need to apply any redo information.

**Case 2** The tablespace being recovered is read-write, but was read-only when the last backup occurred. In this case, you need to restore the tablespace from the backup and apply the redo information from the point when the tablespace was made read-write.

**Case 3** The tablespace being recovered is read-only, but was read-write when the last backup occurred. Because you should always back up a tablespace after making it read-only, you should not experience this situation. However, if this does occur, you must restore the tablespace from the backup and recover up to the time that the tablespace was made read-only.

# Read-Only Tablespace Recovery Issues

**Special considerations must be taken for read-only tablespaces when:**

- **Re-creating a control file**
- **Renaming data files**
- **Using a backup control file**

ORACLE

12-36

Copyright © Oracle Corporation, 2001. All rights reserved.

## **Recovery of Read-Only Tablespaces**

### **Re-creating a Control File**

If you need to re-create a control file with the `CREATE CONTROL FILE` command and your database has read-only tablespaces, you must follow special procedures. Issue the command `ALTER DATABASE BACKUP CONTROLFILE TO TRACE` to get a listing of the procedures.

### **Renaming Datafiles**

If you cannot restore a copy of the datafiles in a read-only tablespace to the correct destination, you can use the `ALTER DATABASE RENAME` command to place the files in a new location.

### **Backup Control File**

The procedure for recovering read-only tablespaces with a backup control file is essentially the same as for offline normal tablespaces, except that you need to bring the tablespace online after the database is open.



# Loss of Control Files

**You may need to create control files if:**

- **All control files are lost because of a failure**
- **The name of a database needs to be changed**
- **The current settings in the control file need to be changed**

ORACLE

12-37

Copyright © Oracle Corporation, 2001. All rights reserved.

## Loss of Control Files

Following are three situations in which you may be confronted with recovering or re-creating a control file:

- All control files are lost because of a failure. You should rarely encounter this problem if you have properly multiplexed your control files and spread them across multiple physical devices.
- You need to change the name of a database. This may be necessary if the database is to become part of a distributed environment and other databases have the same name. It may also be necessary when restoring a database for recovery purposes.
- You need to change a setting in the control file. Many settings in the control file are fixed at the time the control file is created. After the control file is created, these settings cannot be dynamically altered and you must re-create the control file to change them.

# Recovering Control Files

## Methods to recover from loss of control file:

- Use the current control file
- Create a new control file
- Use a backup control file

ORACLE

12-38

Copyright © Oracle Corporation, 2001. All rights reserved.

## Recovering Control Files

There are three ways to restore and recover a control file:

- Use a current copy to restore a lost file. This assumes that you have not lost all of your control files.
- Use the `CREATE CONTROLFILE` command to create a new file. To do this you must know all of the files for the database. Backing up a control file to trace on an occasional basis facilitates this process.
- Use the `RECOVER DATABASE USING BACKUP CONTROLFILE` command. This is necessary if all control files have been lost.

**Note:** If your database is properly configured with mirrored control files, you minimize the likelihood that you will experience the loss of all control files.

# Summary

**In this lesson, you should have learned how to:**

- **Determine what type of recovery is required**
- **Determine which files need to be restored and recovered**
- **Recover a database in Noarchivelog mode**
- **Recover a database in Archivelog mode**
- **Restore datafiles to different locations if the original location is unavailable**

**ORACLE**

## Practices 12-1 and 12-2 Overview

These practices cover the following topics:

- Performing complete database recovery with the database in Noarchivelog mode
- Performing complete database recovery with the database in Archivelog mode

ORACLE

12-40

Copyright © Oracle Corporation, 2001. All rights reserved.

## Practice 12-1 User-Managed Complete Recovery

### Complete Database Recovery: Noarchivelog Mode

1. Shut down the database and disable automatic archiving. Start the instance and mount the database. Set the database in Noarchivelog mode, and then open the database. Confirm the status by issuing the `ARCHIVE LOG LIST` command.
2. Shut down the database and perform a full, closed backup by using operating system commands to copy the files to the `$HOME/BACKUP/NOARCH` directory. Start the instance, mount and open the database.
3. Run the `$HOME/STUDENT/LABS/emphist.sql` script. This script creates a new table named `EMPHIST` in the `HR` schema and adds rows to it. Query the table to obtain a count of the number of rows.
4. Connect as `system/manager` and issue the following query to obtain the names of datafiles that contain the `EMPHIST` table:

```
SELECT f.file_name from dba_tables t,dba_data_files f
WHERE table_name = 'EMPHIST' and
       t.tablespace_name=f.tablespace_name;
```

5. Run the `$HOME/STUDENT/LABS/breakdb.sql` script to simulate failure.
6. Attempt to restart the instance and open the database. What happens?
7. Shut down the database and use the appropriate operating system command to replace the current database with the latest backup (**Hint:** Copy from the `NOARCH` directory to the `ORADATA` directory).
8. Start the instance, mount and open the database.
9. Connect to the database as `hr/hr` and execute a query against the `EMPHIST` table. What happens and why?
10. What conclusions can you make about offline backups and recovery for databases in Noarchivelog mode?

## Practice 12-2 User-Managed Complete Recovery

### Complete Database Recovery: Archivelog Mode

1. Query the `V$DATABASE` view to determine the Archivelog mode of the database. Use the `ARCHIVE LOG LIST` command to check the status of automatic archiving.
2. Shut down the instance and configure automatic archiving. Mount the database and use the `ALTER DATABASE` command to set the database in Archivelog mode.
3. Verify your changes with the `ARCHIVE LOG LIST` command. Note the current log sequence number.
4. Perform a closed database backup. Store the backup in the `$HOME/BACKUP/UMAN` directory.
5. Run the `$HOME/STUDENT/LABS/emphist.sql` script. This script creates a new table named `EMPHIST` in the `HR` schema and adds rows to it. Issue a query against the `EMPHIST` table to determine how many rows it contains.
6. Connect as `system/manager` and run the `$HOME/STUDENT/LABS/checktbs.sql` script and note the datafiles associated with the tablespace that contains the `EMPHIST` table.
7. Run the `$HOME/STUDENT/LABS/breakdb.sql` script to simulate hardware failure.
8. Attempt to start the instance and open the database. What happens?
9. The Oracle server cannot locate the files for the `USERS` tablespace because of a perceived media failure. Because archiving is enabled, you can now perform a complete recovery. Restore the datafiles for the `USERS` tablespace from the backup that you made in Step 4.
10. Use the `RECOVER DATABASE` command to recover the database.
11. When recovery is complete, open the database to make it available for all users.
12. Query the `DBA_TABLESPACES` view to see if the `USERS` tablespace is online.
13. Execute a query against the `HR.EMPHIST` table. What happens?
14. Connect as `system/manager` and query the `V$LOG` view and note the sequence number. Compare this value with the value in step 3. What conclusions can you make about complete recovery?

## Practice 12-3 User-Managed Complete Recovery

### Optional Practice: Open Database Recovery

1. Run the `$HOME/STUDENT/LABS/breakdb.sql` script to simulate hardware failure.
2. Attempt to restart the instance and open the database. What happens?
3. The Oracle server cannot locate the files for the `USERS` tablespace because of a perceived media failure. You can now perform a complete recovery. Take the datafiles for the `USERS` tablespace offline.
4. Open the database to make it available for all users.
5. Take the `USERS` tablespace offline, then restore all datafiles from the backup.
6. Use the `RECOVER TABLESPACE` command to recover the tablespace.
7. Put the `USERS` tablespace back online.
8. Execute a query against the `HR.EMPHIST` table.

## Practice 12-4 User-Managed Complete Recovery

### Optional Practice: Recovery After Adding a New Datafile

1. Run the `$HOME/STUDENT/LABS/newtbs.sql` script as the user `SYSTEM` to:
  - Create a new tablespace with a new datafile
  - Create a table named `NEW_EMP` in the `HR` schema in the new tablespace
  - Simulate the loss of the new datafile

2. Connect as `hr/hr` and update the rows in the `NEW_EMP` table as follows: What happens?

```
SQL> UPDATE new_emp
```

```
2> SET salary = salary * 1.1;
```

The Oracle server cannot locate the file for the `NEW_USERS` tablespace.

3. You can perform a complete recovery after the re-creation of the file for which you have no backup. You can either take the datafile for the `NEW_USERS` tablespace offline or take the tablespace offline, because it only contains one datafile.

**Note:** The `IMMEDIATE` option must be included so that the database writer will not try to write to a nonexistent file.

Confirm the recovery status by querying `V$RECOVER_FILE`.

4. You now must re-create the file.
5. Use the `RECOVER TABLESPACE` command to apply the redo logs to the datafile.
6. When recovery is complete, bring the tablespace online.
7. Again update the rows in the `HR.NEW_EMP` table as follows:

```
SQL> UPDATE new_emp
```

```
2> SET salary = salary * 1.1;
```

8. Connect as `system/manager` and drop the `NEW_USERS` tablespace and associated datafiles in preparation for later practices.



## **Practice 12-5 User-Managed Complete Recovery**

### **Optional Practice: Recovering from a Failure During an Online Backup**

In the practice you will simulate a failure in the database while performing an online backup of the `SAMPLE` tablespace. You will need to issue the appropriate commands to recover and reopen the database.

1. Begin the online backup of the `SAMPLE` tablespace by issuing the appropriate command in `SQL*Plus`.
2. Make an operating system backup of the files belonging to the `SAMPLE` tablespace in the `$HOME/BACKUP/UMAN` directory.
3. Issue the `SHUTDOWN ABORT` command in `SQL*Plus`.
4. Start the instance and mount the database.
5. Query `V$BACKUP` to determine if any files are in an online backup.
6. Issue the appropriate command to end the backup and unfreeze the datafile header. Query `V$BACKUP` to check the status of the datafile.
7. Open the database for users.



# 13

## **RMAN Complete Recovery**

ORACLE

Copyright © Oracle Corporation, 2001. All rights reserved.

# Objectives

**After completing this lesson, you should be able to do the following:**

- **Describe the use of RMAN for restoration and recovery**
- **Perform recovery in Noarchivelog mode**
- **Perform complete recovery in Archivelog mode**
- **Restore datafiles to different locations**
- **Relocate and recover a tablespace by using archived redo log files**

ORACLE

## Restoration and Datafile Media Recovery Using RMAN

- Restore files from backup sets or image copies using the RMAN `RESTORE` command
- Recover files using the RMAN `RECOVER` command

ORACLE

13-3

Copyright © Oracle Corporation, 2001. All rights reserved.

### Restoration and Datafile Media Recovery Using RMAN

RMAN automates the procedure for restoring files. When you issue the `RESTORE` command, RMAN uses a server session to restore the correct backups and copies. The RMAN repository is used to select the best available backup set or image copies to use in the restoration. By default, RMAN does not restore a file if the file is already in the correct place and its header contains the correct information. In releases prior to Oracle9i, the files were always restored.

When you issue the RMAN `RECOVER` command, RMAN applies changes from online redo log files and archived redo log files, or uses incremental backups to recover the restored files.

Using RMAN you can perform recovery at the following levels:

- Database
- Tablespace
- Datafile

In complete recovery, all of the redo entries in the archived redo logs files and online redo log files are used to recover the database. The damaged files are restored from a backup and the log files are used to update the datafiles to the current point in time.

## Using RMAN to Recover a Database in Noarchivelog Mode

```
rman target /  
RMAN> STARTUP MOUNT  
RMAN> RESTORE DATABASE;  
RMAN> RECOVER DATABASE;  
RMAN> ALTER DATABASE OPEN RESETLOGS;
```

ORACLE

13-4

Copyright © Oracle Corporation, 2001. All rights reserved.

### Using RMAN to Restore a Database in Noarchivelog Mode

Recovery Manager uses the recovery catalog or target database control file to determine which full or incremental backups or image copies it will use during restoration.

#### Example

This example assumes that:

- A full backup taken using RMAN is available on disk.
- The current control files were not damaged and do not need to be restored.

## **Using RMAN to Restore a Database in Noarchivelog Mode (continued)**

### **Considerations for Restoring in NOARCHIVELOG Mode**

Consider the following when RMAN restore or recovery operations are performed on a database in Noarchivelog mode:

- You can only restore using RMAN if the backups were taken or registered with RMAN.
- To restore to a previous point in time, you may have to use the backup of an older control file and use the `RESTORE CONTROL FILE` option. The database should be in `NOMOUNT` state to restore the control file.
- The target database must be in mount mode for the restoration of datafiles.
- All of the datafiles must be restored from a backup taken at the same time.
- The `ALTER DATABASE OPEN RESETLOGS` command may be required if a backup of the control file was restored.
- A whole backup is required after an `OPEN WITH RESETLOGS` command.

## Using RMAN to Recover a Database in Archivelog Mode

```
rman target /  
RMAN> STARTUP MOUNT  
RMAN> RESTORE DATABASE;  
RMAN> RECOVER DATABASE;  
RMAN> ALTER DATABASE OPEN;
```

ORACLE

13-6

Copyright © Oracle Corporation, 2001. All rights reserved.

### Using RMAN to Restore a Database in Archivelog Mode

In this example you use RMAN to restore and recover the whole database.

This example assumes that:

- A full backup taken using RMAN is available on disk.
- The current control files were not damaged and do not need to be restored.
- All datafiles are damaged or lost.



## Using RMAN to Restore Datafiles to a New Location

- Use the **SET NEWNAME** command to restore the datafile to the new location.
- Use the **SWITCH** command to record the change in the control file.

ORACLE

13-7

Copyright © Oracle Corporation, 2001. All rights reserved.

### Restoring Datafiles to a New Location

If you have experienced media failure such as the loss of a disk drive, you may need to restore the datafiles to a new location.

#### Example

1. Connect to RMAN.  
`rman target`
2. Mount the database.  
`RMAN> STARTUP MOUNT`
3. Use RMAN to restore the datafile to the new location and record the change in the control file.  

```
run{
  set newname for datafile 1 to '/<newdir>/system01.dbf' ...
  restore database;
  switch datafile all;
  recover database;
  alter database open; }
```

# Using RMAN to Recover a Tablespace

**Use the following RMAN commands to restore and recover a tablespace:**

- **RESTORE TABLESPACE**
- **RECOVER TABLESPACE**

ORACLE

13-8

Copyright © Oracle Corporation, 2001. All rights reserved.

## Using RMAN to Recover a Tablespace

If you have determined that you need to restore and recover the `users` tablespace, issue the RMAN commands as follows:

```
run{
  sql "alter tablespace users offline immediate";
  restore tablespace users;
  recover tablespace users;
  sql "alter tablespace users online";
}
```

## Using RMAN to Relocate a Tablespace

- Use the **SET NEWNAME** command to restore the files.
- Use the **SWITCH** command to record the new names in the control file.
- Use the **RECOVER TABLESPACE** command to recover the datafiles of the tablespace.

ORACLE

13-9

Copyright © Oracle Corporation, 2001. All rights reserved.

### Using RMAN to Relocate a Tablespace

If a datafile cannot be accessed because of disk failure, you need to restore it to a new location or switch to an existing image copy.

The same procedure is useful when you want to relocate the tablespace, because you are running short of disk space in one drive or you are reorganizing the database to improve performance.

#### Example

You notice that u03 has been corrupted and the database is still open. Occasionally users complain that they cannot access information in datafile number 3.

The following is the procedure to restore the datafiles to a new location:

1. Check the location and size of the datafile on u03, using the following command:

```
SQL> SELECT file#, name, bytes FROM v$datafile;
```

FILE#	NAME	BYTES
1	/ORADATA/u01/system01.dbf	120586240
2	/ORADATA/u02/undotbs.dbf	31457280
3	/ORADATA/u03/users01.dbf	5242880

...

You determine that there is enough space on u04 for datafile 3.

## Using RMAN to Relocate a Tablespace (continued)

2. Make sure that the file (tablespace if necessary) is offline so that it can be restored successfully using the RESTORE command.
3. Because the file was copied to a new location (using SET NEWNAME), the file must be made current by notifying the Oracle server of the new file location using the SWITCH command.
4. Use the RECOVER command to start applying the combination of incremental backups, cumulative backups, archived redo log files and online redo logs to the restored file to synchronize the database.
5. When recovery is finished, bring the data file online. Notify users that the database is available for use, and that they should reenter any data that was not committed before system failure.

The following command may be used:

```
RUN{
SQL "alter tablespace users offline immediate";
SET NEWNAME FOR datafile '/ORADATA/u03/users01.dbf'
to '/ORADATA/u04/users01.dbf'; #Specify where to restore the
file
RESTORE (tablespace users); #Restore the datafile
SWITCH datafile 3;#Update the control file and recovery
catalog
RECOVER TABLESPACE users; #Recover the tablespace
SQL "alter tablespace tbs1 online";}

```

# Summary

**In this lesson, you should have learned how to:**

- **Recover a database in Noarchivelog mode**
- **Recover a database in Archivelog mode**
- **Restore datafiles to different locations if the original location is unavailable**

ORACLE

## Practices 13-1 and 13-2 Overview

These practices cover the following topics:

- Using RMAN to recover a tablespace
- Using RMAN to recover relocated datafiles

ORACLE

## Practice 13-1 RMAN Complete Recovery

### Tablespace Recovery Using RMAN

1. Using RMAN, connect to your target database and configure controlfile autobackup using the following format: `$HOME/BACKUP/RMAN/%F.bck`
2. Make a whole database backup specifying the following format: `$HOME/BACKUP/RMAN/df_%d_%s_%p.bus`
3. Connect as `sysdba` in SQL\*Plus and run the `$HOME/STUDENT/LABS/breakdb.sql` script.
4. Invoke SQL\*Plus, connect as `sysdba` and start up your instance.
5. Using RMAN, connect to your target database and restore and recover the `USERS` tablespace.
6. Open the database after recovery completes.

## **Practice 13-2 RMAN Complete Recovery**

### **Recovering Datafiles to a New Location Using RMAN**

1. Connect as sysdba in SQL\*Plus and run the `$HOME/STUDENT/LABS/breakdb.sql` script.
2. You have determined that u03 (`$HOME/ORADATA/u03`) is corrupted. You must relocate the datafile for the USERS tablespace to another location. `$HOME/ORADATA/u04` has sufficient space. Using RMAN, construct a RUN block to relocate the datafile from u03 to u04 and recover the USERS tablespace.
3. Recover the datafile that you have relocated.



# 14

## User-Managed Incomplete Recovery

ORACLE

Copyright © Oracle Corporation, 2001. All rights reserved.

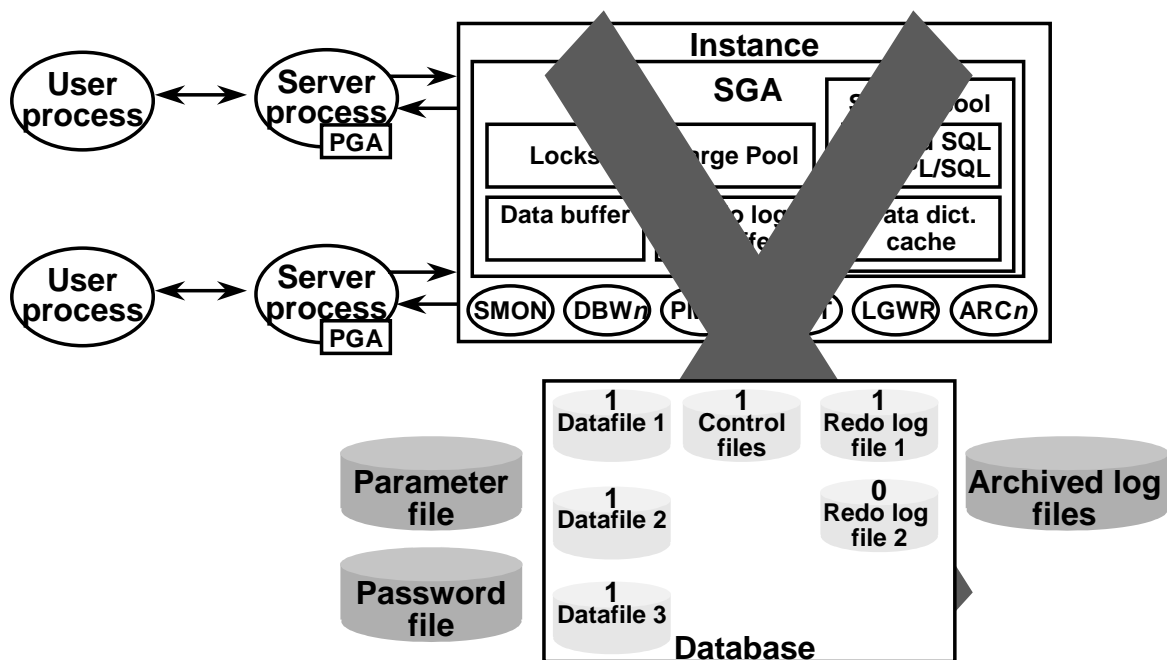
# Objectives

**After completing this lesson, you should be able to do the following:**

- **Describe the steps of incomplete recovery**
- **Perform an incomplete database recovery**
- **Identify the loss of current online redo log files**

ORACLE

# Incomplete Recovery Overview



ORACLE

14-3

Copyright © Oracle Corporation, 2001. All rights reserved.

## Incomplete Recovery

Incomplete recovery reconstructs the database to a prior point in time (before the time of the failure).

**Note:** This situation results in the loss of data from transactions committed after the time of recovery. This data will need to be reentered manually if required. Perform this recovery only when absolutely necessary. Incomplete recovery can be a difficult and time-consuming operation.

To performing incomplete recovery, you need:

- A valid offline or online backup of all of the datafiles made before the recovery point
- All archived logs from the backup until the specified time of recovery

Incomplete recovery is typically used when a complete recovery operation fails.

## Reasons for Performing Incomplete Recovery

- **Complete recovery fails because an archived log is lost.**
- **All control files are lost.**
- **All unarchived redo log files and a datafile are lost.**
- **User error**
  - **An important table was dropped.**
  - **Invalid data was committed in a table.**

ORACLE

14-4

Copyright © Oracle Corporation, 2001. All rights reserved.

### Common Reasons for Performing Incomplete Recovery

- **Missing archive:** A complete recovery operation fails because of a bad or missing archived log. Recovery can only be completed to a time in the past, prior to applying the archived log.
- **Loss of control files:** You did not mirror your control file and you do not know the structure of your database, but you have a backup of an old binary copy.
- **Loss of redo logs:** Redo logs were not mirrored and you lost a redo log before it was archived, along with a datafile. Recovery cannot continue past the lost redo log.
- **User error:** A user drops the wrong table, commits data updated with an incorrect WHERE clause, and so forth.

## Types of Incomplete Recovery

- **There are three types of incomplete recovery:**
  - **Time-based recovery**
  - **Cancel-based recovery**
  - **Change-based recovery**
- **You may need to recover using a restored control file when:**
  - **Control files are lost**
  - **Performing incomplete recovery to a point when the database structure is different than the current**

ORACLE

14-5

Copyright © Oracle Corporation, 2001. All rights reserved.

### Types of Incomplete Recovery

#### Time-Based Recovery

This method of recovery is terminated after all changes up to a specified point in time are committed. Use this approach when:

- Unwanted changes to data were made or important tables dropped, and the approximate time of the error is known. Recovery time and data loss will be minimized if you are notified immediately. Well tested programs, security, and procedures should prevent the need for this type of recovery.
- The approximate time a nonmirrored online redo log becomes corrupt. Mirroring of logs should prevent the need for this type of recovery.

#### Cancel-Based Recovery

This method of recovery is terminated by entering CANCEL at the recovery prompt (instead of a log file name). Use this approach when:

- A current redo log file or group is damaged and is not available for recovery. Mirroring should prevent the need for this type of recovery.
- An archived redo log file needed for recovery is lost. Frequent backups and multiple archive destinations should prevent the need for this type of recovery.

## **Types of Incomplete Recovery (continued)**

### **Change-Based Recovery**

This method of recovery is terminated after all changes up to the specified system change number (SCN) are committed. Use this approach when recovering databases in a distributed environment. This method is not described in more detail in this course.

### **Recovery Using a Backup Control File**

This method of recovery is terminated when the specified method of recovery (cancel-, time-, or change-based) has completed or control files are recovered. You must specify in the `RECOVER DATABASE` command that an old copy of the control file will be used for recovery. Use this approach when:

- All control files are lost, the control file cannot be re-created, and a binary backup of the control file exists. Mirroring the control file (onto different disks) and keeping a current text version of the `CREATE CONTROLFILE` statement reduces the chances of using this method.
- Restoring a database, with a different structure than the current database, to a prior point in time.

# Incomplete Recovery Guidelines

- **Follow all steps carefully.**
- **Take whole database backups before and after recovery.**
- **Always verify that the recovery was successful.**
- **Back up and remove archived logs.**

ORACLE

14-7

Copyright © Oracle Corporation, 2001. All rights reserved.

## Incomplete Recovery Guidelines

- It is important to follow all recovery steps carefully, because most problems with incomplete recovery are caused by a DBA error during the recovery.

Transaction activity can only be rolled forward to the desired time, not back to the desired time. This is the reason why all datafiles must be restored for the database to be taken back in time. Failure to restore all datafiles prevents the (unsynchronized) database from opening.

- Perform a whole closed database backup (including control files and redo logs) before starting incomplete recovery. This is helpful in the following ways:
  - It allows you to recover from error. If your recovery fails (for example, you recover past the desired point of recovery), redo logs and control files cannot be used for the next recovery unless there is a backup of these files.
  - It saves time if the recovery fails. In this situation, you can restore the datafiles from the new backup, rather than from a previous backup which needs archives applied.

**Note:** If a whole backup is not performed, at least archive the current redo log (ALTER SYSTEM ARCHIVE LOG CURRENT) and back up the control file (ALTER DATABASE BACKUP CONTROLFILE TO <LOCATION>).

- Perform a whole closed backup after successful recovery. This is recommended if a recovery is required before completion of the next scheduled backup.

## **Incomplete Recovery Guidelines (continued)**

- Always verify that the problem has been corrected before allowing users to access the system, in case the recovery failed and needs to be performed again.
- Back up (and later remove) archived logs from the system to prevent mixing archives from different database incarnations.

Consider the following example:

- A database at log seq 144 has archived logs from arch\_120.rdo to arch\_143.rdo.
- After performing incomplete recovery, a new database incarnation is created, setting the database log seq to 0.
- Archived logs arch\_120.rdo to arch\_143.rdo are now part of the old database incarnation.
- After 120 log switches, the archived log arch\_120.rdo will be overwritten, and is backed up with all other archives (including the old archived logs arch\_121.rdo to arch\_143.rdo).
- At a later stage, if recovery requires arch\_124.rdo, you need to make sure that the archived log restored from the backup is for the correct database incarnation, otherwise an error will result.



## Incomplete Recovery and the Alert Log

- **Check before and after recovery**
- **Contains error information, hints, and SCNs**

ORACLE

14-9

Copyright © Oracle Corporation, 2001. All rights reserved.

### The Alert Log

During recovery, progress information is stored in the alert log. This file should always be checked before and after recovery. Here is a sample `alert.log` file:

```
$ vi /disk1/BDUMP/alert_DB00.log
...
Media Recovery Log
ORA-279 ... RECOVER    database until time '2001...
Wed Mar 07 11:55:13 2001
RECOVER DATABASE CONTINUE DEFAULT
Media Recovery Log /disk1/archive/arch_34.rdo
Incomplete recovery done UNTIL CHANGE 309121
Media Recovery Complete
Completed: RECOVER DATABASE CONTINUE DEFAULT
Wed Mar 07 11:55:13 2001
alter database open resetlogs
...
```

## **User-Managed Procedures for Incomplete Recovery**

- 1. Shut down and back up the database.**
- 2. Restore all datafiles. Do not restore the control file, redo logs, password file, or parameter file.**
- 3. Mount the database.**
- 4. Recover the datafiles to a point before the time of failure.**
- 5. Open the database with RESETLOGS.**
- 6. Perform a closed database backup.**

ORACLE

14-10

Copyright © Oracle Corporation, 2001. All rights reserved.

### **Incomplete Recovery Steps with User-Managed Procedures**

When a failure occurs requiring incomplete recovery, you must have the following to recover:

- A valid offline or online backup containing all datafiles.
- All archived redo logs, from the restored backup to before the time of failure.

Follow these steps to recover:

1. Perform a full closed backup of the existing database. Shut down the database as all datafiles, including the system tablespace files, will be restored from a backup.
2. Restore all datafiles to take your database back in time.
3. Place the database in mount mode and insure that the datafiles are online.
4. Recover the database.
5. Open the database by using the RESETLOGS option and verify the recovery.
6. Perform a whole closed backup of the database.

## RECOVER Command Overview

### Recover a database until cancel:

```
recover database until cancel
```

### Recover a database until time:

```
recover database  
until time '2001-03-04:14:22:03'
```

### Recover using backup control file:

```
recover database  
until time '2001-03-04:14:22:03'  
using backup controlfile
```

ORACLE

14-11

Copyright © Oracle Corporation, 2001. All rights reserved.

## RECOVER Command for Incomplete Recovery

The following command is used to perform incomplete recovery:

```
recover [automatic] database <Option>
```

**where:**    `automatic`    Automatically applies archived and redo log files.

Option	<code>until time 'YYYY-MM-DD:HH:MI:SS';</code> <code>until cancel;</code> <code>until scn &lt;integer&gt;;</code> <code>using backup control file;</code>
--------	--

### Note

- The `ALTER DATABASE RECOVER` syntax can be used instead.
- To apply redo log files automatically during recovery, you can use the `SQL*Plus SET AUTORECOVERY ON` command, enter `auto` at the recovery prompt, or use the `SQL RECOVER AUTOMATIC` command.

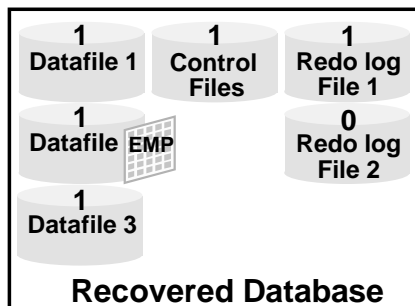
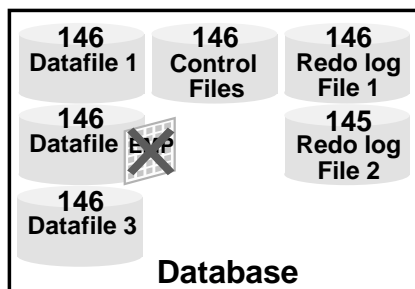
# Time-Based Recovery Example

## Scenario

- The current time is 12:00 p.m. on March 9, 2001.
- The `EMPLOYEES` table has been dropped.
- The table was dropped at approximately 11:45 a.m.
- Database activity is minimal because most staff are currently in a meeting.
- The table must be recovered.

ORACLE

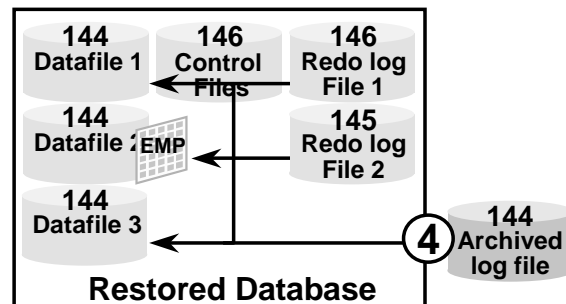
## UNTIL TIME Recovery



① Shut down and back up

② Restore all datafiles

③ Mount the database



⑤ Open with Resetlogs

⑥ Back up the database

ORACLE

14-13

Copyright © Oracle Corporation, 2001. All rights reserved.

### Incomplete Recovery Using UNTIL TIME

1. Shut down the database and begin recovery. Because the approximate time of the failure is known and the database structure has not changed since 11:44 a.m., you can use the `until time` method:

If the database is open, shut it down by using either the `NORMAL`, `IMMEDIATE`, or `TRANSACTIONAL` options.

2. Restore all datafiles from backup (the most recent if possible). You may need to restore archived logs. If there is enough space available, restore to the `LOG_ARCHIVE_DEST` location or use the `ARCHIVE SYSTEM ARCHIVE LOG START TO <LOCATION>` or `SET LOGSOURCE <LOCATION>` to change the location
3. Mount the database.
4. Recover the database:

```
SQL> recover database until time '2001-03-09:11:44:00';
ORA-00279: change 148448 ...02/29/98 17:04:20 needed for
thread ORA-00289: suggestion : /disk1/archive/arch_6 .rdo
ORA-00280: change 148448 for thread 1 is in sequence #6
Log applied.
```

...

Media recovery complete.

## Incomplete Recovery Using UNTIL TIME (continued)

5. To synchronize datafiles with control files and redo logs, open the database by using the RESETLOGS option:

```
SQL> alter database open resetlogs;
```

```
SQL> archive log list;
```

```
...
```

```
Oldest online log sequence 0
```

```
Next log sequence to archive 1
```

```
Current log sequence 1
```

6. Before performing the whole closed database backup, query the EMPLOYEES table to make sure it exists.

When recovery is successful and the backup has completed, notify users that the database is available for use, and any data entered after the recovery time (11:44 a.m.) will need to be reentered.

# Cancel-Based Recovery Example

## Scenario

- The current time is 12:00 p.m. on March 9, 2001.
- The **EMPLOYEES** table was dropped while someone was trying to fix bad blocks.
- Log files exist on the same disk.
- The table was dropped at approximately 11:45 a.m.
- Staff are currently in a meeting.

ORACLE

14-15

Copyright © Oracle Corporation, 2001. All rights reserved.

## Incomplete Recovery Using UNTIL CANCEL

You are concerned about block corruption in the **EMPLOYEES** table resulting from disk error. Because redo logs are contained on the same disk, you decide to check the status of redo logs and archived logs:

```
SQL> select * from v$logfile;
```

GROUP#	STATUS	MEMBER
--------	--------	--------

2		/disk1/data/log2a.rdo
1		/disk1/data/log1a.rdo

```
SQL> select * from v$log;
```

G#	...	SEQ#	BYTES	MEMBERS	ARC	STATUS	...	FIRST_TIME
1	...	49	153600	1	NO	CURRENT	...	01-03-09:11:55
2	...	48	153600	1	NO	INACTIVE	...	01-03-09:11:34

# Cancel-Based Recovery Example

## Findings

- Redo logs are not multiplexed.
- One of the online redo logs is missing.
- The missing redo log is not archived.
- The redo log contained information from 11:34 a.m.
- Twenty-six minutes of data will be lost.
- Users can recover their data.

ORACLE

14-16

Copyright © Oracle Corporation, 2001. All rights reserved.

## Incomplete Recovery Using UNTIL CANCEL (continued)

After searching through the /disk1/data directory, you notice that redo log log2a.rdo cannot be located and has not been archived. Therefore, you cannot recover past this point.

Querying V\$LOG\_HISTORY confirms the absence of archived log seq 48 (log2a.rdo):

```
SQL> select * from v$log_history;
```

RECID	STAMP	...	FIRST_CHANGE	FIRST_TIME
-----	-----	...	-----	-----
1	318531466	...	88330	01-02-28:12:43
47	319512880	...	309067	01-03-09:11:26

Because this is an OLTP system, the output from V\$LOG shows that an extra 10 minutes of work will be lost if the database is recovered before applying log2a.rdo. Users are not happy about losing work, but can recover that work.

You can recover the database as follows:

1. Shut down the database.
2. Restore all datafiles from the most recent backup.
3. You already have a valid backup, so mount the database.



## Incomplete Recovery Using UNTIL CANCEL (continued)

4. Recover the database until log seq 48:

```
SQL> recover database until cancel
ORA-00279:change 148448...03/02/01 12:45:20 needed for thread
ORA-00289: suggestion : /disk1/archive/arch_34.rdo
ORA-00280: change 148448 for thread 1 is in sequence #34
Log applied.
...
ORA-00279:change 309012...03/09/01 11:33:56 needed for thread
ORA-00289: suggestion : /disk1/archive/arch_48.rdo
ORA-00280: change 309012 for thread 1 is in sequence #48
Specify log: {<RET>=suggested | filename | AUTO | CANCEL}
cancel
Media recovery cancelled.
```

5. Open the database by using the RESETLOGS option.
6. Check that the EMPLOYEES table exists.
7. When recovery is complete, make a backup. Notify users that the database is available for use, and any data entered after the recovery time (11:34 a.m.) will need to be reentered.

# Using a Backup Control File During Recovery

## Scenario

- The current time is 12:00 p.m. on March 9, 2001.
- The tablespace containing the **EMPLOYEES** table has been dropped.
- The error occurred around 11:45 a.m.
- Many employee records were updated this morning, but not since 11:00 a.m.
- Backups are taken every night.

ORACLE

14-18

Copyright © Oracle Corporation, 2001. All rights reserved.

## Incomplete Recovery Using a Backup Control File

1. The tablespace containing the **EMPLOYEES** table has been dropped as follows:  

```
SQL> drop tablespace emp_ts including contents;
```
2. You immediately ask users to log out and keep records of the data entered over the past 15 minutes. While you wait for all users to log out and remaining sessions to be killed, you place the database in restricted mode to prevent further access:  

```
SQL> alter system enable restricted session;
```
3. During your investigation, you locate the binary control file from the backup last night. Because the current control file will be replaced, you carefully gather database structure information in case it is required:  

```
SQL> select * from v$log;
```

GROUP#...	SEQ#	BYTES	...	ARC	STATUS	...	FIRST_TIME
1	...	61	153600...	NO	CURRENT	...	01-03-09:11:55
2	...	60	153600...	NO	INACTIVE	...	01-03-09:11:34

```
SQL> select tablespace_name, file_name from dba_data_files
```

TABLESPACE_NAME	FILE_NAME
EMP_TS	/disk1/data/emp_01.dbf

## Using a Backup Control File During Recovery

### Findings

- The backup from last night contains datafiles and control files required for recovery.
- The `EMP_TS` tablespace has one datafile.
- The current log sequence number is 61.
- You confirm that the tablespace was dropped at 11:44:54 a.m. on March 9, 2001.
- Datafile number 4 is offline.

ORACLE

14-19

Copyright © Oracle Corporation, 2001. All rights reserved.

### Incomplete Recovery Using a Backup Control File (continued)

4. You confirm the time of error by checking the alert log:

```
UNIX> vi /disk1/BDUMP/alert*.log
```

```
Or NT> notepad c:\BDUMP\alert_DB00.log
```

```
...
```

```
Fri Mar09 11:44:54 1999
```

```
drop tablespace emp_ts including contents
```

```
...
```

5. Shut down the database, back up control files, then restore all datafiles and control files for the database at a time when the tablespace existed. After attempting to open the database, the following error informs you that the redo logs and control files are not synchronized:

```
ORA-00314: log 1 of thread 1, expected sequence# doesn't  
match
```

```
ORA-00312: online log 1 thread 1: '/disk1/data/log1a.rdo'
```

## Incomplete Recovery Using a Backup Control File (continued)

6. Verify whether any offline datafiles exist and place them online, because any offline files may be unrecoverable after recovery:

```
SQL> select * from v$recover_file;
```

FILE#	ONLINE	ERROR	CHANGE#	TIME
4	OFFLINE		288772	02-MAR-01

```
SQL> alter database datafile 4 online;
```

7. Perform the recovery:

```
SQL> recover database until time '2001-03-09:11:44:00'  
2> using backup controlfile;
```

```
...
```

Media recovery complete.

**Note:** If you are prompted for an online redo log file, enter the file name at the prompt to continue the recovery.

If the following error appears instead of “Media recovery complete”, it indicates that either datafiles need to be restored from an earlier backup, or more recovery is required (not possible here).

```
ORA-01152: file 7 was not restored from a sufficiently old backup
```

```
ORA-01110: datafile 7: '/disk1/data/new01.dbf'
```

8. To synchronize datafiles with the control files and redo logs, open the database with the RESETLOGS option.
9. Verify that the EMPLOYEES table exists.
10. Make a whole backup.
11. Notify users that the database is available for use, and any data entered after 11:44 a.m. will need to be reentered.

# Loss of Current Redo Log Files

**If the database is closed:**

- **Attempt to open the database.**
- **Find the current log sequence number.**
- **Recover the database until cancel.**
- **Drop and re-create log files if necessary.**
- **Open the database using RESETLOGS.**
- **Perform a whole-database backup.**

ORACLE

14-21

Copyright © Oracle Corporation, 2001. All rights reserved.

## Loss of Current Redo Log Files

If the database is closed, media failure may have occurred or a background process may have terminated. Use the steps below to rectify this situation:

1. Attempting to open the database will immediately notify you of the current redo log group through the following message:

Database mounted.

ORA-00313: open failed for members of log group 2 of thread 1

ORA-00312: online log 2 thread 1:  
'/disk1/archive/log2a.rdo'

ORA-27037: unable to obtain file status

SVR4 Error: 2: No such file or directory

Additional information: 3

Because log group 2 is the current log group, it will not have been archived. Using the CLEAR LOGFILE command is of no use because:

```
SQL> alter database clear unarchived logfile group 2;
```

ORA-01624: log 2 needed for crash recovery of thread 1

ORA-00312: online log 2 thread 1:  
'disk1/archive/log2a.rdo'

## Database Closed Because of Failure (continued)

2. Incomplete recovery is therefore required. First, you must note the current log sequence number:

```
SQL> select * from v$log;
```

GROUP#...	SEQ#	BYTES	...	ARC	STATUS...	FIRST_TIME
-----...	----	-----...	---	-----...	-----	
1 ...	60	153600	...	NO	INACTIVE.	01-03-09:19:34
2 ...	61	153600	...	NO	CURRENT...	01-03-09:19:50

3. Restore all datafiles from a previous backup, use the RECOVER UNTIL CANCEL command, and stop before redo log 61 is applied:

```
SQL> recover database until cancel;
```

```
ORA-00279: change 309043...03/09/01 14:50:14 needed for  
thread 1
```

```
ORA-00289: suggestion : /disk1/archive/arch_46.rdo
```

```
ORA-00280: change 309043 for thread 1 is in sequence #46
```

```
Specify log: {<RET>=suggested | filename | AUTO | CANCEL}
```

```
...
```

```
ORA-00279: change 309141...03/09/01 19:50:14 needed for  
thread 1
```

```
ORA-00289: suggestion : /disk1/archive/arch_61.rdo
```

```
ORA-00280: change 309043 for thread 1 is in sequence #61
```

```
ORA-00278: log file ... no longer needed for this recovery
```

```
Specify log: {<RET>=suggested | filename | AUTO | CANCEL}
```

```
cancel
```

```
Media recovery complete.
```

4. Open the database using the RESETLOGS option.
5. The database should now be operational, because any missing log files will be re-created.  
**Note:** If log files need to be re-created on another disk due to media failure, use the ALTER DATABASE DROP LOG GROUP and ALTER DATABASE ADD LOG GROUP commands to create the log files manually.
6. Because you just performed incomplete recovery, the database should now be backed up.

# Summary

**In this lesson, you should have learned how to:**

- **Perform incomplete database recovery**
- **Recover from the loss of current online redo log files**

**ORACLE**

## Practices 14-1 and 14-2 Overview

These practices cover the following topics:

- Recovery from user failure
- Recovery with lost archived redo log files

ORACLE



## Practice 14-1 User-Managed Incomplete Recovery

### Recovering from User Failure: Incomplete Recovery

1. If you are unsure whether you have a valid backup from the previous practice, then perform either a whole closed or opened database backup. Store the backup in the `$HOME/BACKUP/UMAN` directory. Start the instance and open the database
2. Connect as `hr/hr`. Insert rows into the `EMPHIST` table by issuing the following statement:  

```
INSERT INTO emphist SELECT * FROM emphist;
```
3. Issue a `SELECT` statement to obtain a count of the rows in the `EMPHIST` table. Note the number of rows.
4. Connect as `system/manager` and issue the following query:  

```
SELECT f.file_name FROM dba_tables t, dba_data_files f
      WHERE table_name = 'EMPHIST' AND
      t.tablespace_name = f.tablespace_name;
```

Record the filename of all datafiles for the tablespace.
5. Record the current system time by using an operating system command.
6. Query `V$LOG` to find the current online log sequence number.
7. Connect as `hr/hr` and add rows to the `EMPHIST` table by executing the following command:  

```
INSERT INTO emphist SELECT * FROM emphist;
```
8. Issue a `SELECT` statement to obtain a count of the rows in the `EMPHIST` table. Note the number of rows.
9. Run the `$HOME/STUDENT/LABS/breaktab.sql` script to simulate a user accidentally dropping the `EMPHIST` table.
10. Attempt to query the `EMPHIST` table. What happens?
11. The Oracle server cannot locate the `EMPHIST` table. You need to restore this table to the database. Since archiving is enabled and you know the approximate time of failure, you can now perform an incomplete recovery to restore the table.  

Connect as `sysdba` and shut down the instance. Start the instance and mount the database.
12. Restore all datafiles from the backup that you made in step 1. If you did not make a new backup at the beginning of this practice, be sure to restore the datafiles for the `USERS` tablespace to the proper directory.
13. Recover the database until the time you noted in step 5.
14. When recovery is complete, open the database using the `RESETLOGS` option.
15. Execute a query against the `HR.EMPHIST` table. What happens and why?
16. Connect as `system/manager`, query the `V$LOG` view, and note the sequence number. Compare this value with the value in step 5. What conclusions can you make about incomplete recovery?
17. Take a whole offline backup. Store the backup in the `$HOME/BACKUP/UMAN` directory.

## Practice 14-2 User-Managed Incomplete Recovery (Optional)

### Incomplete Recovery: Recovery with a Lost Archived Log

1. Determine the current system time by using an operating system command.
2. Query the V\$LOG view and record the current online log sequence number.
3. Run the \$HOME/STUDENT/LABS/moredata.sql script to switch the logs and create a new table.
4. Run the \$HOME/STUDENT/LABS/breakarc.sql script to simulate the loss of an archived log file.
5. Run the \$HOME/STUDENT/LABS/breakdb.sql script to simulate hardware failure.
6. Attempt to start the instance and open the database. What happens?
7. The server cannot locate the files for the USERS tablespace because of perceived media failure. Because archiving is enabled, you can attempt to perform a complete recovery. Restore the datafile for the USERS tablespace from the backup you made in Practice 14-1.
8. Use the RECOVER AUTOMATIC DATABASE command to recover the database. Note the name of any files that cannot be found. Issue a CANCEL when the server is unable to locate the specified archived redo log file.
9. Attempt to open the database. What happens?
10. The recovery has been cancelled prior to applying the lost archived redo log file. The datafiles in the USERS tablespace cannot be brought forward to the current database time. Because recovery cannot take the database back in time, you must perform an incomplete recovery. Restore all datafiles from the backup you made in Practice 14-1.
11. Recover the database using the UNTIL CANCEL option, stopping when the Oracle server requests the archived redo log file you noted in step 8.  
**Note:** Do not use the automatic method. Apply each archived redo log file manually as the Oracle server requests it.
12. Enter cancel at the recovery prompt.
13. Once recovery is complete, open the database using the RESETLOGS option.
14. Query V\$DATAFILE to verify that all datafiles are online.
15. Take a whole offline backup. Store the backup in the \$HOME/BACKUP/UMAN directory.
16. Connect as sysdba and start the instance, mount and open the database.

# 15

## **RMAN Incomplete Recovery**

ORACLE

Copyright © Oracle Corporation, 2001. All rights reserved.

# Objectives

**After completing this lesson, you should be able to do the following:**

- **Perform an incomplete database recovery using  
UNTIL TIME**
- **Perform an incomplete database recovery using  
UNTIL SEQUENCE**

ORACLE

## Incomplete Recovery of a Database Using RMAN

1. Mount the database.
2. Allocate multiple channels for parallelization.
3. Restore all datafiles.
4. Recover the database by using `UNTIL TIME`, `UNTIL SEQUENCE`, or `UNTIL SCN`.
5. Open the database by using `RESETLOGS`.
6. Perform a whole database backup.

ORACLE

15-3

Copyright © Oracle Corporation, 2001. All rights reserved.

### Incomplete Recovery of a Database Using RMAN

The restore and recovery process for incomplete recovery follows the same procedure and syntax as complete recovery, except that all datafiles need to be restored from the past backup.

#### Note

- The target database must be in mounted state.
- The files being restored must be offline. You can restore using RMAN only if the backups were taken or registered with RMAN.

## RMAN Incomplete Recovery UNTIL TIME Example

```
RMAN> run {  
  2> allocate channel c1 type DISK;  
  3> allocate channel c2 type DISK;  
  4> set until time = '2000-12-09:11:44:00';  
  5> restore database;  
  6> recover database;  
  7> alter database open resetlogs; }
```

ORACLE

15-4

Copyright © Oracle Corporation, 2001. All rights reserved.

### RMAN Incomplete Recovery UNTIL TIME Example

At 12:00 p.m. on Tuesday, December 9, 2000, you immediately shut down the database and begin recovery after determining that the EMPLOYEES table was dropped. The approximate time of failure is known and the database structure has not changed since 11:44 a.m. You can use the UNTIL TIME method:

1. If the target database is open, perform a clean shut down.
2. Mount the target database. Do not back up the database during the recovery.
3. Start Recovery Manager and connect to the target database. Before starting RMAN, ensure that NLS\_LANG and NLS\_DATE\_FORMAT environment variables are set appropriately:

```
$ NLS_LANG=american  
$ NLS_DATE_FORMAT='YYYY-MM-DD:HH24:MI:SS'  
$ rman target rman/rman@DB00
```

4. You can allocate multiple channels to improve the performance:

```
RMAN> run { allocate channel c1 type DISK;  
          allocate channel c2 type DISK;
```

## **RMAN Incomplete Recovery UNTIL TIME Example (continued)**

5. Specify the time for recovery and restore all datafiles from a backup with RMAN commands. RMAN chooses the correct files based on the SET UNTIL command:

```
RMAN> ... set until time = '2000-12-09:11:44:00';
```

```
RMAN> ... restore database;
```

**Note:** If you need to restore archived redo log files to a new location use the RMAN SET ARCHIVELOG DESTINATION TO <location> command.

6. Recover the database to the time specified in the SET UNTIL command:

```
RMAN> ... recover database;
```

7. Open the database using the RESETLOGS option:

```
RMAN> ... alter database open resetlogs; }
```

8. Check that the table exists and perform a backup.

9. Notify users that the database is available for use, and that they should reenter any data that was not committed before system failure.

10. If using a recovery catalog, register the new incarnation of the database:

```
RMAN> reset database;
```

## RMAN Incomplete Recovery UNTIL SEQUENCE Example

```
RMAN> RUN {  
  2> SET UNTIL SEQUENCE 120 THREAD 1;  
  3> ALTER DATABASE MOUNT;  
  4> RESTORE DATABASE;  
  5> RECOVER DATABASE; # recovers through log 119  
  6> SQL "ALTER DATABASE OPEN RESETLOGS";  
  7> }
```

ORACLE

15-6

Copyright © Oracle Corporation, 2001. All rights reserved.

### RMAN Incomplete Recovery UNTIL SEQUENCE Example

The UNTIL SEQUENCE clause specifies a redo log sequence number and thread as an upper limit. RMAN performs the operation on files up to but not including the specified log sequence number.

This example assumes that log sequence 120 was lost due to a disk crash and the database needs to be recovered using the available archived redo logs.



# Summary

**In this lesson, you should have learned how to:**

- **Perform an incomplete database recovery using  
UNTIL TIME**
- **Perform an incomplete database recovery using  
UNTIL SEQUENCE**

ORACLE

## **Practice 15 Overview**

**This practice covers recovery with lost archived redo log files.**

**ORACLE**

## Practice 15 RMAN Incomplete Recovery

### Incomplete Recovery: Recovery with a Lost Archived Log

1. Make a whole database backup specifying the following format:  
`$HOME/BACKUP/RMAN/df_%d_%s_%p.bus`
2. Run the `$HOME/STUDENT/LABS/moredata.sql` script to switch the logs and create a new table.
3. Run the `$HOME/STUDENT/LABS/breakarc.sql` script to simulate the loss of an archived redo log file.
4. Run the `$HOME/STUDENT/LABS/breakdb.sql` script to simulate hardware failure.
5. Attempt to start the instance and open the database. What happens?
6. The server cannot locate the files for the USERS tablespace because of perceived media failure. Because archiving is enabled, you can attempt to perform a complete recovery. Use RMAN to restore the datafile for the USERS tablespace.
7. Use RMAN to recover the tablespace. Note the name and sequence number of any files that cannot be found.
8. Use RMAN with the `UNTIL LOG SEQUENCE` clause to perform incomplete recovery through the last good archived redo log file.
9. Once recovery is complete, open the database using the `RESETLOGS` option.
10. Make a new backup in the `$HOME/BACKUP/RMAN` directory with the following format:  
`df_%d_%s_%p.bus`



# 16

## **RMAN Maintenance**

ORACLE

Copyright © Oracle Corporation, 2001. All rights reserved.

# Objectives

**After completing this lesson, you should be able to do the following:**

- **Perform cross checking of backups and copies**
- **Update the repository when backups have been deleted**
- **Change the availability status of backups and copies**
- **Make a backup or copy exempt from the retention policy**
- **Catalog backups made with operating system commands**

ORACLE

# Cross Checking Backups and Copies

**Use the CROSSCHECK command to:**

- **Ensure repository information is synchronized with actual files**
- **Check the status of a backup or copy**
- **Update the repository when files have been deleted with operating system commands**

ORACLE

16-3

Copyright © Oracle Corporation, 2001. All rights reserved.

## Cross Checks of RMAN Backups and Copies

Performing a cross check provides you with a way to ensure that data about backup sets and image copies in the RMAN repository is synchronized with corresponding data on disk or in the media management catalog.

You can use the LIST command to obtain a report of the backups and copies that you have made and then use the CROSSCHECK command to check that these files still exist. If RMAN cannot find a file, it updates the repository records to EXPIRED. You can determine which files are marked EXPIRED by issuing a LIST EXPIRED command. Then, you can run DELETE EXPIRED to remove the repository records for all expired files.

If the backup or copy is on disk, then the CROSSCHECK command determines whether the header of the backup piece is valid. If the backup is on tape, then the command simply checks that the backups exist.

# The CROSSCHECK Command

## Cross check all backups in the database:

```
CROSSCHECK BACKUP;
```

## Cross check all copies in the database:

```
CROSSCHECK COPY;
```

ORACLE

16-4

Copyright © Oracle Corporation, 2001. All rights reserved.

## Using the CROSSCHECK Command

You can issue the `CROSSCHECK BACKUP` command to cross check backup sets, backup pieces, and proxy copies. By default, RMAN cross checks backups of the whole database.

You can issue the `CROSSCHECK COPY` command to cross check datafile copies, control file copies, archived redo logs, and image copies of archived redo logs. By default, all files in the database with status `AVAILABLE` or `EXPIRED` are checked.

The `CROSSCHECK` command can also be used with options to restrict the check to a specific backup piece, backupset, datafile, or control file copy.



# Deleting Backups and Copies

Use the **DELETE** command to:

- Delete physical backups and image copies
- Update repository status to **DELETED**
- Remove records from the recovery catalog

ORACLE

16-5

Copyright © Oracle Corporation, 2001. All rights reserved.

## Deleting Backups and Copies

You can use the **DELETE** command to delete backups and image copies and update the repository and recovery catalog. When you issue the command, RMAN displays a list of the files and prompts you to confirm the deletion request.

# The DELETE Command

## Delete a specific backup set:

```
DELETE BACKUPSET 102;
```

## Delete an expired backup without the confirmation:

```
DELETE NOPROMPT EXPIRED BACKUP OF TABLESPACE users;
```

## Delete all backups, copies, and archived redo log files based on the configured retention policy:

```
DELETE OBSOLETE;
```

ORACLE

16-6

Copyright © Oracle Corporation, 2001. All rights reserved.

## Using the DELETE Command

By default, the DELETE command displays a list of the files and prompts you for confirmation before deleting any file in the list. You can use the NOPROMPT option to suppress the confirmation prompt. NOPROMPT is the default when running the DELETE command from a command file.

If you specify the EXPIRED option on the DELETE command, then only the files that were marked EXPIRED by the CROSSCHECK command are removed. You can use the LIST command to determine which backups or copies are expired.

You can specify the OBSOLETE option to remove files considered OBSOLETE by the retention policy. You can specify a retention policy by using CONFIGURE RETENTION POLICY. You can also specify the REDUNDANCY and RECOVERY WINDOW options on the DELETE command as in the following example:

```
RMAN> DELETE OBSOLETE RECOVERY WINDOW OF 7 DAYS;
```

## Deleting Backups and Copies

Use the **BACKUP ... DELETE INPUT** command to:

- Delete input files upon successful creation of the backup set
- Delete archived redo log files, datafile copies, and backup sets

ORACLE

16-7

Copyright © Oracle Corporation, 2001. All rights reserved.

### Using the **DELETE INPUT** Option of the **BACKUP** Command

You can use the **DELETE INPUT** option with the **BACKUP** command to direct RMAN to delete the input files upon successful creation of the backup set. You can specify this option only when backing up archived redo log files, datafile copies, or backup sets. Using this option is equivalent to issuing a **DELETE** command for the input files.

If you specify **DELETE ALL INPUT** when backing up archived redo log files, then RMAN deletes all copies of corresponding archived redo logs that match the selection criteria. The **BACKUP ARCHIVELOG** command only backs up one copy of each distinct log sequence number, so if you specify the **DELETE INPUT** option without the **ALL** keyword, RMAN deletes only the copy of the file that it backs up.

## Changing the Availability of RMAN Backups and Copies

- **Change the status of a backup or copy to Unavailable with the `CHANGE ... UNAVAILABLE` command.**
- **Return the status to Available with the `CHANGE ... AVAILABLE` command.**

ORACLE

16-8

Copyright © Oracle Corporation, 2001. All rights reserved.

### Changing the Availability Status

You can use the `CHANGE ... UNAVAILABLE` command when a backup or copy cannot be found or is unavailable because of hardware maintenance. If a file is marked `UNAVAILABLE`, RMAN will not use the file when a `RESTORE` or `RECOVER` command is issued.

When the file is found or the maintenance is completed, you can mark it available again by issuing the `CHANGE ... AVAILABLE` command.

```
CHANGE BACKUPSET 100 UNAVAILABLE;
```

# Changing the Status to Unavailable

## Change the status of a specific datafile:

```
CHANGE DATAFILECOPY '/DB01/BACKUP/users01.dbf'  
UNAVAILABLE;
```

## Change the status of a control file backup:

```
CHANGE BACKUP OF CONTROLFILE UNAVAILABLE;
```

## Change the status of archived redo log files:

```
CHANGE COPY OF ARCHIVELOG SEQUENCE BETWEEN 230 AND  
240 UNAVAILABLE;
```

ORACLE

## Changing the Status to Unavailable

When you issue the `CHANGE ... UNAVAILABLE` command, the status is marked in the repository. You can view the status with the `LIST` command.

The `BACKUP` keyword of the `CHANGE` command indicates on which backup sets, backup pieces, or proxy copies the command should operate. If you do not specify an option for `BACKUP`, then `CHANGE BACKUP` operates on all backups recorded in the repository.

The `COPY` keyword of the `CHANGE` command indicates on which datafile copies, archived redo logs, or image copies of archived redo logs the command should operate. If you do not specify an option for `COPY`, then `CHANGE COPY` operates on all copies recorded in the repository.

## Exempting a Backup or Copy from the Retention Policy

- Use the **CHANGE ... KEEP** command to exempt a backup or copy from the retention policy
- Use the **CHANGE ... NOKEEP** command to cancel the exemption

ORACLE

16-10

Copyright © Oracle Corporation, 2001. All rights reserved.

### Retention Policy Exemption

A retention policy specifies when RMAN should consider the backups and copies it creates as obsolete.

You can use the **CHANGE ... KEEP** command to make a file exempt from the retention policy and **CHANGE ... NOKEEP** to make the file conform to the retention policy.

**KEEP** overrides any configured retention policy for this backup or copy so that the backup is not obsolete. You can use this option to create a long-term backup. The exempted backup is still a fully valid backup and can be restored just as any other RMAN backup.

**NOKEEP** specifies that the backup or copy expires according to the user's retention policy. This is the default behavior if no **KEEP** option is specified on the **BACKUP** or **COPY** command.

# The CHANGE ... KEEP Command

## Create a long-term backup:

```
CHANGE BACKUPSET 123 KEEP FOREVER NOLOGS;
```

## Make a datafile exempt from the retention policy for 60 days:

```
CHANGE DATAFILECOPY '/DB01/BACKUP/users01.dbf' KEEP  
UNTIL 'SYSDATE+60';
```

ORACLE

16-11

Copyright © Oracle Corporation, 2001. All rights reserved.

## Using the CHANGE...KEEP Command

The FOREVER parameter specifies that the backup or copy never expires. You must use a recovery catalog when FOREVER is specified; otherwise the backup records eventually age out of the control file.

When you specify the LOGS parameter, you have indicated that all of the archived logs required to recover this backup or copy must remain available as long as this backup or copy is available.

NOLOGS specifies that this backup or copy cannot be recovered because the archived logs needed to recover this backup will not be kept. You can only use this backup or copy to restore the database to the point in time that the backup or copy was taken.

You can use the UNTIL TIME = 'date\_string' parameter to specify the date until which the backup or copy must be kept. You can either specify a specific time by using the current NLS\_DATE\_FORMAT, or a SQL date expression, such as 'SYSDATE+365'.

# Cataloging Archived Redo Log Files and User-Managed Backups

**You can use the CATALOG command to add information to the repository about:**

- **An operating system datafile copy**
- **An archived redo log copy**
- **A control file copy**

ORACLE

16-12

Copyright © Oracle Corporation, 2001. All rights reserved.

## Cataloging Files in the Repository

When you make backups with operating system commands no information is recorded in the repository. You must manually notify RMAN when you make backups with an operating system utility.

For an operating system backup to be cataloged, it must be:

- Accessible on disk
- A complete image copy of a single file
- A consistent or inconsistent datafile, control file, or archived redo log backup. If it is inconsistent, then it must have been created with the `BEGIN BACKUP/END BACKUP` statements. If it is a control file backup, then it should have been made with the `ALTER DATABASE BACKUP CONTROLFILE` statement.

RMAN treats the operating system backups as datafile copies. During cataloging, RMAN only checks the file header. It does not check whether the file was correctly copied by the operating system utility.

You can make RMAN aware of the existence of archived redo log files that are not recorded in the repository. You need to do this if you have restored your control file from a backup and you want to update the repository with archived redo log file information.



# The CATALOG Command

**Catalog a backup taken with an operating system command:**

```
CATALOG DATAFILECOPY '/DB01/BACKUP/users01.dbf';
```

**Catalog archived redo log files:**

```
CATALOG ARCHIVELOG '/ORADATA/ARCHIVE1/arch_12.arc',  
'/ORADATA/ARCHIVE1/arch_13.arc';
```

ORACLE

16-13

Copyright © Oracle Corporation, 2001. All rights reserved.

## Using the CATALOG Command

You use the `CONTROLFILECOPY` parameter to specify the file name of a control file copy to be added to or updated in the repository.

`DATAFILECOPY` specifies the filename of a datafile copy to be added to or updated in the repository.

You use the `ARCHIVELOG` parameter to specify the filename of an archived redo log file to be added to or updated in the repository.

# Uncataloging RMAN Records

Use the **CHANGE ... UNCATALOG** command to:

- Update the record in the repository to **DELETED** status
- Delete a specific backup or copy record from the recovery catalog

ORACLE

16-14

Copyright © Oracle Corporation, 2001. All rights reserved.

## Uncataloging RMAN Repository Records

You can use this command when you have deleted a backup or copy through a means other than RMAN. After you issue the **CHANGE ... UNCATALOG** command for an RMAN backup, it is permanently unusable by RMAN.

# The CHANGE ... UNCATALOG Command

## Remove records for deleted archived redo log files:

```
CHANGE ARCHIVELOG ... UNCATALOG;
```

## Remove records for a deleted datafile:

```
CHANGE DATAFILECOPY '/DB01/BACKUP/users01.dbf'  
UNCATALOG;
```

ORACLE

16-15

Copyright © Oracle Corporation, 2001. All rights reserved.

## Using the CHANGE...UNCATALOG Command

The CHANGE ... UNCATALOG command is used to update the repository. It does not remove physical backups or copies. You can use this command to notify RMAN when a file is deleted by some means other than an RMAN DELETE command.

## Summary

**In this lesson, you should have learned how to:**

- **Perform cross checking of backups and copies**
- **Update the repository when backups have been deleted**
- **Change the availability status of backups and copies**
- **Make a backup or copy exempt from the retention policy**
- **Catalog backups made with operating system commands**

ORACLE

## Practice 16 Overview

**This practice covers the following topics:**

- **Performing cross checking**
- **Cataloging files in the repository**

ORACLE

## Practice 16 RMAN Maintenance

1. Connect to your database in the default Nocatalog mode.
2. Use the RMAN REPORT command to generate a listing of your database structure.
3. Use the RMAN LIST and CROSSCHECK commands to generate a listing of the backup sets and the status of the files.
4. Using an operating system command, copy the datafile belonging to the SAMPLE tablespace to your BACKUP directory and then remove it from the RMAN directory to simulate a loss of the backup.
5. Use the RMAN CROSSCHECK command to update the repository with the status of the datafile backup that you moved in the previous step. Be sure to specify the backup set that you moved in the previous step.
6. Issue the LIST EXPIRED command to check the status of your files. Are any of your files expired?
7. Using an operating system command, return the SAMPLE tablespace backup to the correct location.
8. Use the RMAN CROSSCHECK command to update the repository with the status of the datafile backup.
9. Make a backup of the datafile belonging to the SAMPLE tablespace to \$HOME/BACKUP/RMAN directory with user-managed procedures.
10. Use the RMAN CATALOG command to update the repository with this backup information.
11. Use the RMAN LIST COPY command to verify that the backup has been recorded in the repository.

# 17

## **Recovery Catalog Creation and Maintenance**

ORACLE

Copyright © Oracle Corporation, 2001. All rights reserved.

# Objectives

**After completing this lesson, you should be able to do the following:**

- **Describe the contents of the recovery catalog**
- **List the RMAN features which require the recovery catalog**
- **Create the recovery catalog**
- **Maintain the recovery catalog by using RMAN commands**
- **Use RMAN to register, resynchronize, and reset a database**

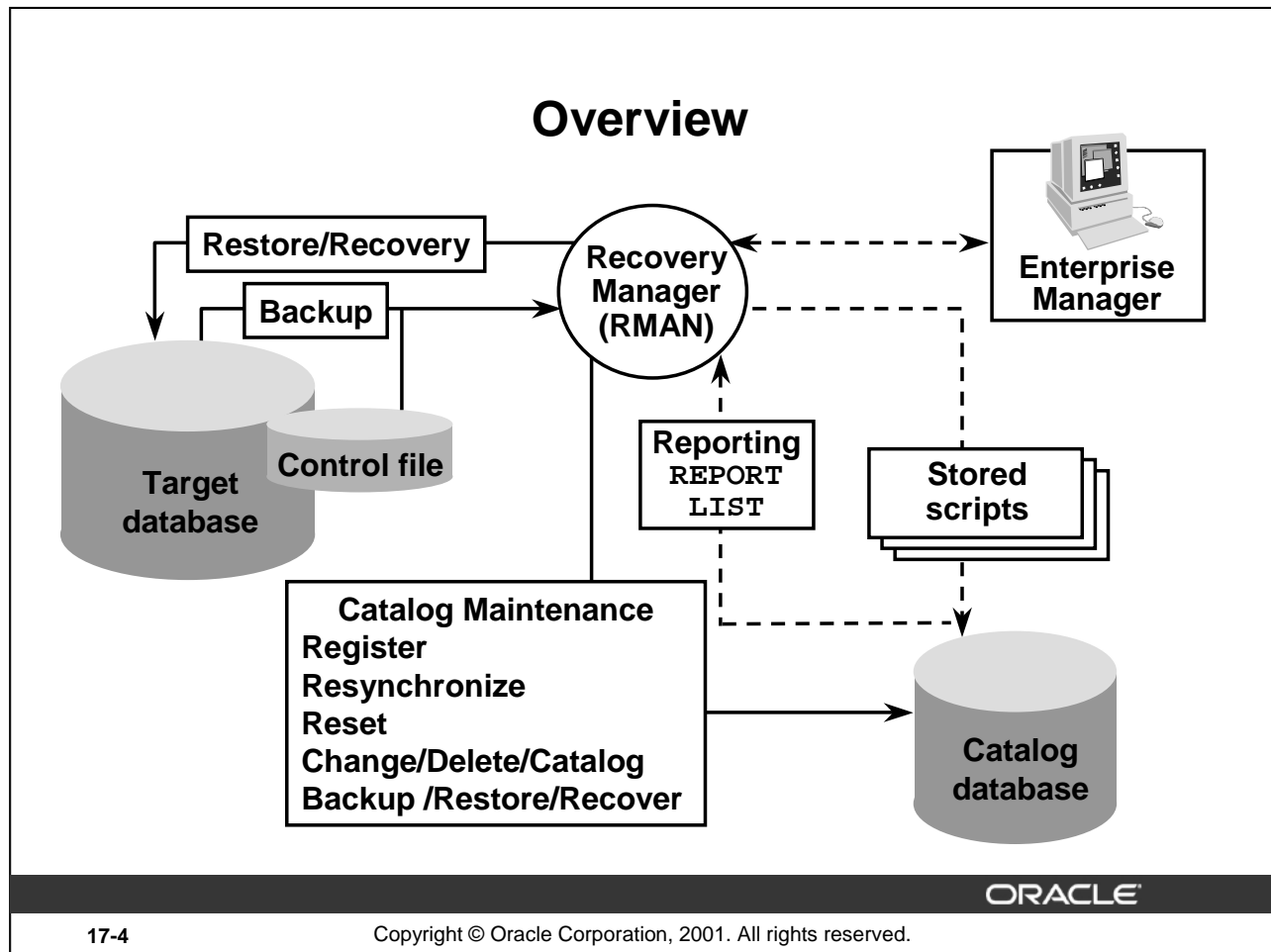
ORACLE



# Objectives

- Query the recovery catalog to generate reports and lists
- Create, store, and run scripts
- Describe methods for backing up and recovering the recovery catalog

ORACLE



## Overview

The recovery catalog is a schema that is created in a separate database. It contains the RMAN metadata obtained from the target database control file. RMAN propagates information about the database structure, archived redo logs, backup sets, and datafile copies into the recovery catalog from the control file of the target database.

You should use a catalog when you have multiple target databases to manage. RMAN stores, uses, and maintains the information in the recovery catalog. The recovery catalog is maintained by RMAN when you do the following:

1. Register the target database in the catalog.
2. Resynchronize the catalog with the control file of the target database.
3. Reset the database to a previous incarnation.
4. Change information about the backups or files.
5. Perform a backup, restore, or recovery operation.

You can use the `REPORT` and `LIST` commands to obtain information from the recovery catalog. You can store scripts in the recovery catalog.

# Recovery Catalog Contents

**The recovery catalog is an optional repository containing information on:**

- **Datafile and archived redo log file backup sets and backup pieces**
- **Datafile copies**
- **Archived redo log files**
- **The physical structure of the target database**

ORACLE

17-5

Copyright © Oracle Corporation, 2001. All rights reserved.

## Recovery Catalog Contents

The recovery catalog contains information about:

- Datafiles and archived redo log file backup sets and backup pieces:

The catalog stores information such as the name and time of the backup set.

- Datafile copies:

The catalog records the time stamp and name of data file copies.

- Archived redo log files and copies of them:

The catalog maintains a record of which archived logs have been created by the server and any copies made by RMAN.

- The physical structure of the target database:

It contains information similar to that contained in the target database control file.

# Recovery Catalog Contents

The recovery catalog can also contain:

- **Persistent RMAN configuration settings**
- **Stored job scripts**

ORACLE

17-6

Copyright © Oracle Corporation, 2001. All rights reserved.

## Recovery Catalog Contents

The recovery catalog can also contain:

- Configuration settings which are persistent across RMAN sessions and are set with the `CONFIGURE` command.
- Stored scripts which are named sequences of commands.

## **Benefits of Using a Recovery Catalog**

**The following features are available only when you use a recovery catalog:**

- **Metadata about multiple target databases in one catalog**
- **Metadata about multiple incarnations of a single target database**
- **Historical metadata**
- **Reporting on the target database at a noncurrent time**

**ORACLE**

17-7

Copyright © Oracle Corporation, 2001. All rights reserved.

### **Benefits of Using a Recovery Catalog**

You can use a recovery catalog to store information about multiple target databases in a single recovery catalog. The recovery catalog can be used to store information about more than one incarnation of a single database. This allows you to report on the target database from a non-current incarnation.

## **Additional Features Which Require the Recovery Catalog**

**You must use a recovery catalog if you want to store the following in the repository:**

- **Scripts**

ORACLE

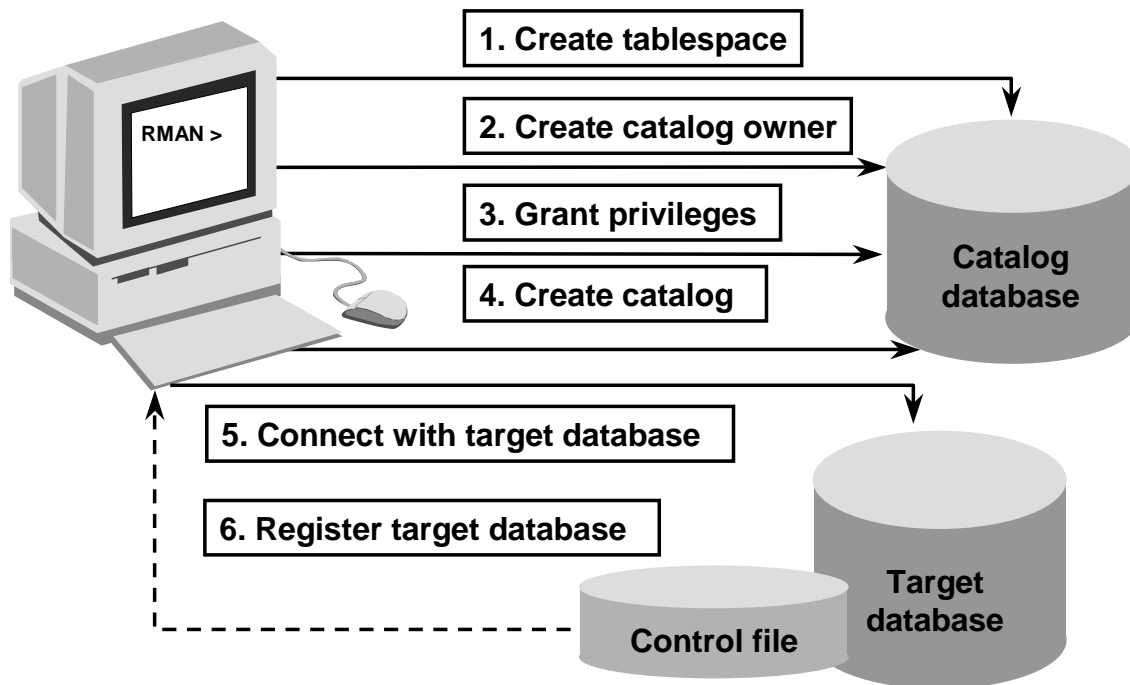
17-8

Copyright © Oracle Corporation, 2001. All rights reserved.

### **Features Which Require the Use of a Recovery Catalog**

If you want to store scripts which contain commands for backup and recovery operations, you must use a recovery catalog. The scripts cannot be stored in the target database control file.

## Create Recovery Catalog



ORACLE

17-9

Copyright © Oracle Corporation, 2001. All rights reserved.

### How to Create a Recovery Catalog

To create the recovery catalog, perform the following steps:

1. Connect to the catalog database and create a tablespace for the catalog:

```

SQL > create tablespace rman_ts datafile '<directory>/<name>'
      2 > size 20M default storage
      3 > (initial 100K next 100K pctincrease 0);
  
```

2. Create a user and schema for the recovery catalog:

```

SQL > create user rman_db01 identified by rman_db01
      2> default tablespace rman_ts
      3> quota unlimited on rman_ts;
  
```

3. Grant the roles and privileges to this user to maintain the recovery catalog and perform the backup and recovery operations.

```

SQL > grant recovery_catalog_owner to rman_db1;
SQL > grant connect, resource to rman_db1;
  
```

## How to Create a Recovery Catalog (continued)

4. Log in to the operating system and issue the RMAN command to invoke the RMAN command interpreter. Create the catalog. Use of the LOG option enables RMAN to output messages and commands to a file.

```
% rman catalog rman_db1/rman_db1@catdb log = catalog.log
create catalog tablespace rman_ts;
exit;
```

**Note:** When you use the LOG option, the output is directed to the file and you may not get the RMAN prompt. So the CREATE CATALOG command should be entered when the cursor appears on the new line. Similarly the exit command should be entered on the next new line. The purpose of the log is to help you review any errors that may arise in the process of creation of catalog so that corrective actions can be taken.

Check the catalog.log file created by RMAN to note if there are any errors in creation of the catalog. If you do find any errors, you should drop all objects under the catalog owner and re-create them from the beginning.

Catalog.log may appear as follows:

```
RMAN-06008: connected to recovery catalog database
RMAN-06428: recovery catalog is not installed
RMAN> create catalog tablespace rman_ts;
RMAN-06431: recovery catalog created
RMAN>
RMAN> exit;
Recovery Manager complete
```

5. Connect to the target database. You must log in as a user with SYSDBA privileges on the target database to perform all the backup and recovery operations.

```
% rman target sys/oracle@db01
RMAN-06005: connected to target database: DB01
(DBID=472633597)
RMAN> connect catalog rman_db01/rman_db01@catdb
RMAN-06008: connected to recovery catalog database
RMAN>
```



## How to Create a Recovery Catalog (continued)

6. Register the target database in the catalog. If the target database is not registered in the recovery catalog, the catalog cannot be used to store information about the database. Recovery Manager uses the internal database identifier (DBID), which is calculated when the database is first created, as a unique identifier for the database. If you attempt to register a new database that has been created by copying an existing database and then changing the `db_name`, the register will fail. You can avoid this problem by using the `DUPLICATE` command, which copies the database from backups and generates a new database identifier.

To back up a copied database, create a new recovery catalog owner, and create the catalog in the new account

RMAN creates rows in the recovery catalog that contain information about the target database. RMAN copies all pertinent data about the target database from the control file into the recovery catalog.

```
RMAN> register database;
```

```
RMAN-03022: compiling command: register
```

```
RMAN-03023: executing command: register
```

```
RMAN-08006: database registered in recovery catalog
```

```
RMAN-03023: executing command: full resync
```

```
RMAN-08002: starting full resync of recovery catalog
```

```
RMAN-08004: full resync complete
```

```
RMAN>
```

# Connecting Using a Recovery Catalog

## Initiating a session on the target database:

**Unix:**

```
$ ORACLE_SID=db01; export ORACLE_SID
$ rman target sys/oracle
RMAN> connect catalog rman_db01/rman_db01@catdb
```

**NT:**

```
C:\> set ORACLE_SID=db01
C:\> rman target sys/oracle
RMAN> connect catalog rman_db01/rman_db01@catdb
```

## Remote connection:

```
rman target sys/oracle@db01
RMAN-6005: connected to target database: ...
RMAN> connect catalog rman_db01/rman_db01@catdb
```

ORACLE

17-12

Copyright © Oracle Corporation, 2001. All rights reserved.

## How to Connect to Recovery Manager

To connect to RMAN using a recovery catalog, follow these steps:

1. Initiate an RMAN session from the target database.

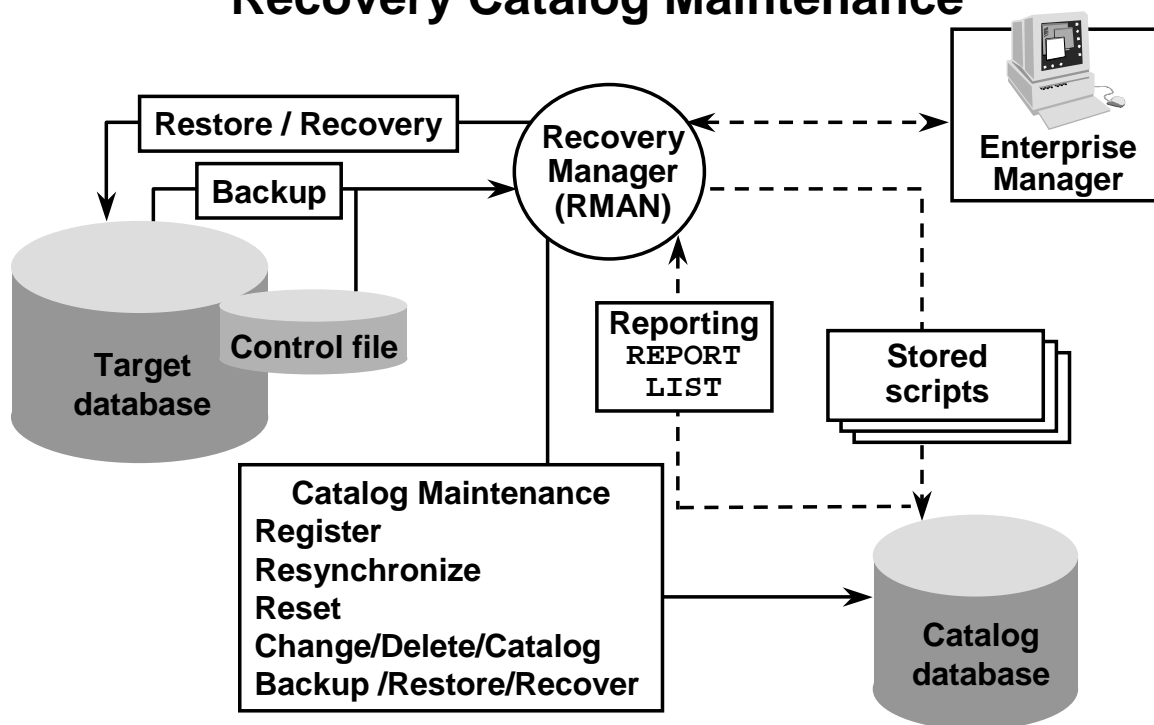
On Windows NT you may invoke the command prompt before issuing the following RMAN command or you can invoke the command line interface using the Run option from Start menu.

```
rman target sys/oracle@db01
RMAN-06005: connected to target database:
DB01(DBID=XXXXXXXXXX)
RMAN>
```

2. Connect to the recovery catalog database.

```
RMAN> connect catalog rman_db01/rman_db01@catdb
RMAN-06008: connected to recovery catalog database
RMAN>
```

## Recovery Catalog Maintenance



ORACLE

17-13

Copyright © Oracle Corporation, 2001. All rights reserved.

### Catalog Maintenance Commands

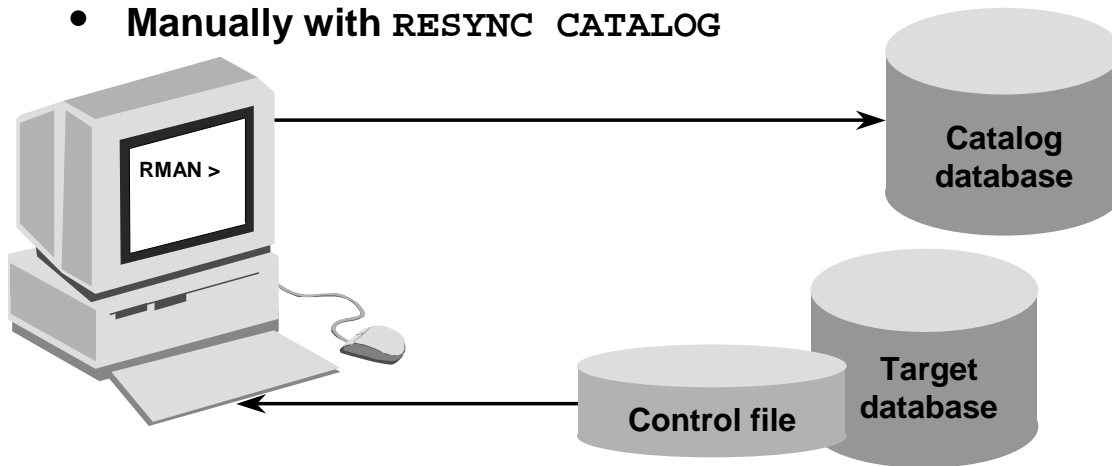
The CATALOG, CHANGE, and DELETE commands can be used to update the recovery catalog manually. These commands were reviewed in a previous lesson.

RESYNC and RESET are covered in this lesson.

# Resynchronization of the Recovery Catalog

**Resynchronization of the recovery catalog happens:**

- **Automatically with RMAN commands**
- **Manually with `RESYNC CATALOG`**



ORACLE

17-14

Copyright © Oracle Corporation, 2001. All rights reserved.

## Resynchronizing the Recovery Catalog

Resynchronization of the recovery catalog ensures that the metadata is current with the target control file.

Resynchronizations can be full or partial. In a partial resynchronization, RMAN reads the current control file to update changed data, but does not resynchronize metadata about the database physical schema: datafiles, tablespaces, redo threads, rollback segments, and online redo logs. In a full resynchronization, RMAN updates all changed records, including schema records.

RMAN automatically detects when it needs to perform a full or partial resynchronization and executes the operation as needed. You can also force a full resynchronization by issuing a `RESYNC CATALOG` command.

To ensure that the catalog stays current, run the `RESYNC CATALOG` command periodically. A good rule of thumb is to run it at least once every  $n$  days, where  $n$  is the setting for the initialization parameter `CONTROL_FILE_RECORD_KEEP_TIME`. Because the control file employs a circular reuse system, backup and copy records eventually get overwritten. Resynchronizing the catalog ensures that these records are stored in the catalog and are not lost.

# Using RESYNC CATALOG for Resynchronization

**Issue the RESYNC CATALOG command when you:**

- **Add or drop a tablespace**
- **Add or drop a datafile**
- **Relocate a database file**

```
$ rman target / catalog rman/rman@catdb  
RMAN> RESYNC CATALOG;
```

ORACLE

17-15

Copyright © Oracle Corporation, 2001. All rights reserved.

## Using the RESYNC CATALOG Command

Any structural changes to the database cause the control file and recovery catalog to become “out of synch.” The catalog will be synchronized automatically when a BACKUP or COPY command is issued with a connection to the catalog. However, this synchronization can cause a delay in the backup operation.

The RESYNC CATALOG command updates the following records:

- **Log history:** Created when a log switch occurs. Recovery Manager tracks this information so that it knows what archive logs it should expect to find.
- **Archived redo log:** Associated with archived logs that were created by archiving an online log, by copying an existing archived log, or by restoring an archived log backup set.
- **Backup history:** Associated with backup sets, backup pieces, backup set members, proxy copies, and image copies.
- **Physical schema:** Associated with datafiles and tablespaces.

# Resetting a Database Incarnation

**Use the RESET DATABASE command:**

- **When the database is opened with the RESETLOGS option**
- **To direct RMAN to create a new database incarnation record**
- **To distinguish between opening with RESETLOGS and an accidental restore operation of an old control file**

ORACLE

17-16

Copyright © Oracle Corporation, 2001. All rights reserved.

## Using the RESET DATABASE Command

An incarnation of a database is a number used to identify a version of the database prior to the log sequence number being reset to zero. This prevents archived and online redo logs from being applied to an incorrect incarnation of the database. The RESET DATABASE command is used by Recovery Manager to store database incarnation information in the recovery catalog. All subsequent backups and log archives are associated with the new database incarnation.

If the target database is recovered to a point in the past, the database must be opened with the RESETLOGS option. In this case, Recovery Manager cannot use the recovery catalog again until a RESET DATABASE command is issued. This enables Recovery Manager to distinguish between a RESETLOGS and an accidental restore operation of an old control file.

### Example

```
RMAN> reset database;
RMAN-03022: compiling command: reset
RMAN-03023: executing command: reset
RMAN-08006: database registered in recovery catalog
RMAN-03023: executing command: full resync
RMAN-08002: starting full resync of recovery catalog
RMAN-08004: full resync complete
```

## Using the RESET DATABASE Command (continued)

### RESET DATABASE TO INCARNATION Key Command

The RESET DATABASE TO INCARNATION <identifier> command is used to undo the effects of a RESETLOGS operation by restoring backups of a prior incarnation of the database. You must specify the primary key of the record for the database incarnation to which you return:

```
RMAN> reset database to incarnation <identifier>;
```

**Note:** The identifier is obtained by the LIST INCARNATION OF DATABASE command.

### Example

```
RMAN> list incarnation of database;
```

```
RMAN-03022: compiling command: list
```

```
RMAN-06240: List of Database Incarnations
```

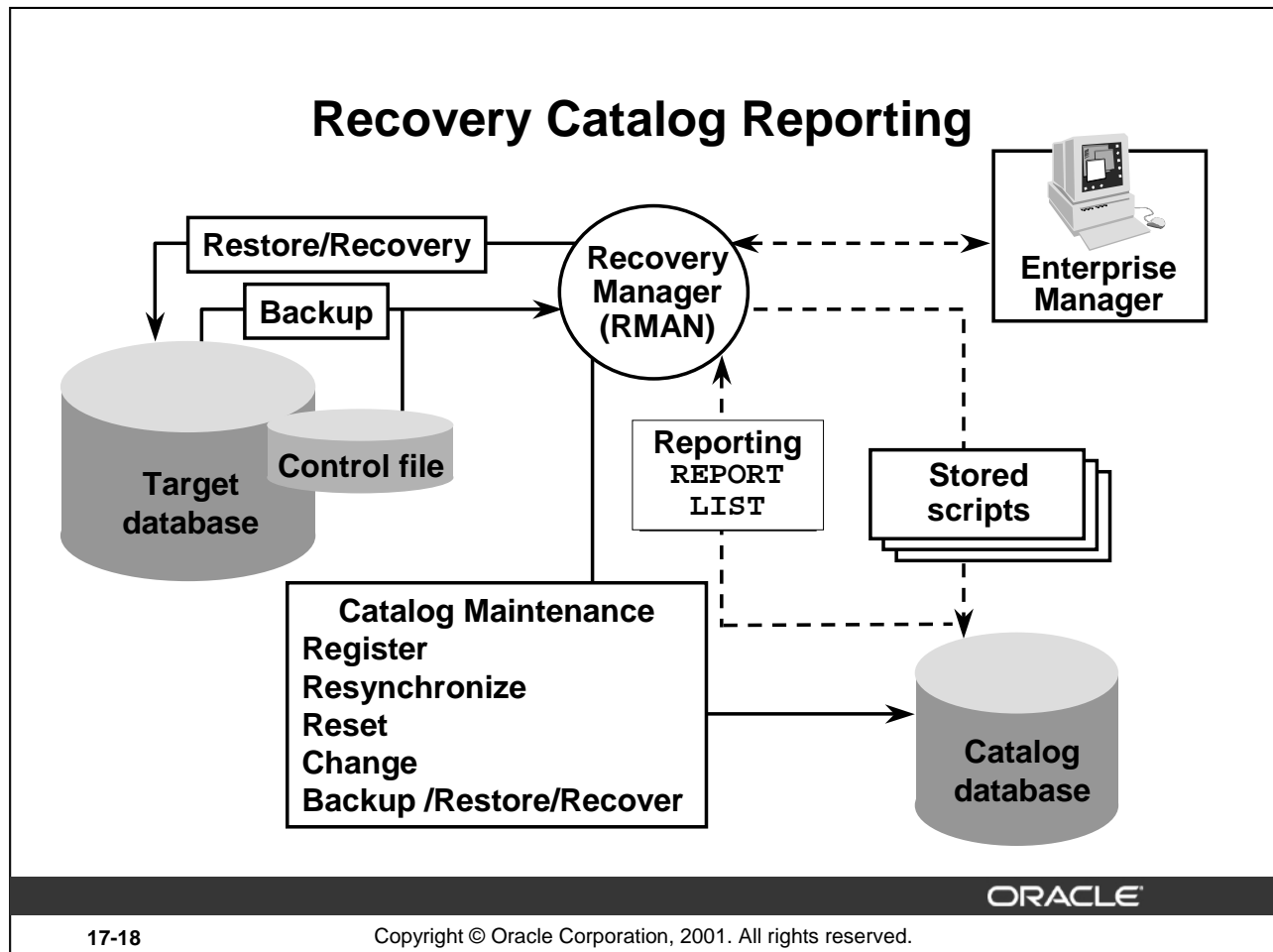
```
RMAN-06241: DB Key Inc Key DB Name          DB ID CUR Reset SCN
```

```
RMAN-06242: -----
```

```
RMAN-06243:      1          2  ORACLE 1186311932 YES      25730
```

```
RMAN-06243:      1        421  ORACLE 1186311932 NO       172279
```

```
RMAN> reset database to incarnation 421;
```



## Recovery Catalog Reporting

These commands analyze and list information contained inside the recovery catalog.

### **REPORT Command**

You can use the `REPORT` command to analyze various aspects of the backup, copy, restore, and recovery operations.

### **LIST Command**

You can use the `LIST` command to display information on backup sets, file copies, and archived logs, which are stored in the recovery catalog.

### **Views**

In addition to the `REPORT` and `LIST` commands, you can use SQL commands to query the data dictionary and dynamic views that are created when the recovery catalog is created.



# Viewing the Recovery Catalog

## Data dictionary views:

- RC\_DATABASE
- RC\_DATAFILE
- RC\_STORED\_SCRIPT
- RC\_STORED\_SCRIPT\_LINE
- RC\_TABLESPACE



ORACLE

17-19

Copyright © Oracle Corporation, 2001. All rights reserved.

## Data Dictionary Views

The recovery catalog views are nonnormalized views that are optimized for RMAN usage rather than user queries. You can usually use the RMAN reporting commands to obtain the needed information from the recovery catalog.

You can use the recovery catalog views as shown in the following examples:

### Example 1

To determine which databases are currently registered in the recovery catalog:

```
SQL> select * from rc_database;
```

DB_KEY	DBINC_KEY	DBID	NAME	CHANGE#	RESETLOGS
1	2	1943591421	DB01	1	20-APR-99

## Data Dictionary Views (continued)

### Example 2

To determine which tablespaces are currently stored in the recovery catalog for the target database:

```
SQL > select DB_KEY, DBINC_KEY, DB_NAME, TS#, NAME,
           CREATION_CHANGE# CHANGE#, CREATION_TIME CRE_DATE
       from rc_tablespace;
```

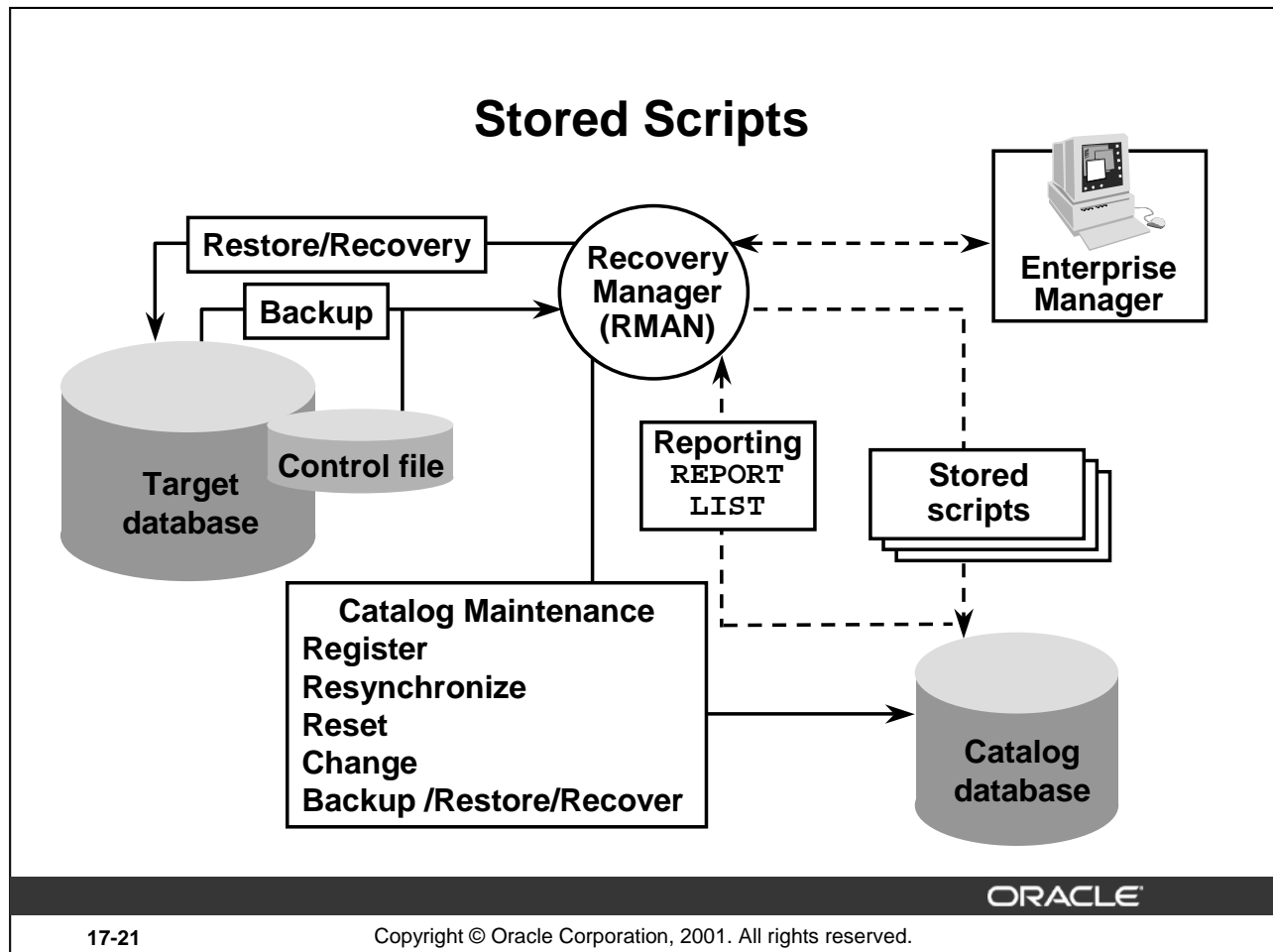
DB_KEY	DBINC_KEY	DB_NAME	TS#	NAME	CHANGE#	CRE_DATE
-----	-----	-----	--	-----	-----	-----
1	2	DB01	3	DATA01	9611	20-APR-99
1	2	DB01	1	RBS	9599	20-APR-99
1	2	DB01	4	RMAN_TS	14023	29-APR-99
1	2	DB01	0	SYSTEM	3	20-APR-99
1	2	DB01	2	TEMP	9605	20-APR-99

### Example 3

To determine which scripts are currently stored in the recovery catalog for the target database:

```
SQL> select * from rc_stored_script;
```

DB_KEY	DB_NAME	SCRIPT_NAME
-----	-----	-----
1	DB01	nightlybackup
1	DB01	archivebackup



## Stored Scripts

A Recovery Manager script is a set of commands that:

- Specify frequently used backup, recover, and restore operations
- Are created using the `CREATE SCRIPT` command
- Are stored in the recovery catalog
- Can be called only by using the `RUN` command
- Enable you to plan, develop, and test a set of commands for backing up, restoring, and recovering the database
- Minimize the potential for operator errors

## Storing and Viewing Scripts

As an example, a script named `level0backup` can be created and stored in the recovery catalog to make an incremental level 0 backup. Storing the script in the recovery catalog enables any DBA using Recovery Manager to access the scripts.

You can display a list of stored scripts by querying the `RC_STORED_SCRIPT` view.

You can query the `RC_STORED_SCRIPT_LINE` view to list the text of a specified stored script or you can use the `PRINT SCRIPT` command.

# Script Examples

## Use CREATE SCRIPT to store a script.

```
RMAN> create script Level0Backup {  
    backup  
    incremental level 0  
    format '/u01/db01/backup/%d_%s_%p'  
    fileperset 5  
    (database include current controlfile);  
    sql 'alter database archive log current';  
}
```

## Use EXECUTE SCRIPT to run a script.

```
RMAN > run {execute script Level0Backup;}
```

ORACLE

## Creating and Using Stored Scripts

Backup, restore, and recovery operations are generally automated using scripts. RMAN provides a way of storing these scripts in the recovery catalog. You create scripts by using the CREATE SCRIPT command. Use the RUN command to execute the script.

# Managing Scripts

## Use REPLACE SCRIPT to rewrite a script

```
RMAN> REPLACE SCRIPT Level0Backup {  
    ...  
    fileperset 3  
    ...  
}
```

## Use DELETE SCRIPT to remove a script

```
RMAN> DELETE SCRIPT Level0Backup;
```

## Use PRINT SCRIPT to display a script

```
RMAN> PRINT SCRIPT Level0Backup;
```

ORACLE

## Managing Stored Scripts

You can rewrite a script with the REPLACE SCRIPT command. You must supply the entire script, not just the changed lines.

## Backup of Recovery Catalog

- **Whole database backup of the database containing the recovery catalog**
- **Tablespace backup of the tablespace containing the recovery catalog**
- **Export:**
  - **If catalog database is not very large, you can export the database at regular intervals.**
  - **If catalog database is large, export the schema containing the recovery catalog.**

ORACLE

17-24

Copyright © Oracle Corporation, 2001. All rights reserved.

### Backup of Recovery Catalog

It is critical to have a tested backup strategy for the recovery catalog. The recovery catalog is a schema of objects stored in a database. The considerations for backup of the recovery catalog are similar to those of a schema.

You could use one of the following strategies to back up the recovery catalog:

- **Whole database backup:** You can take a whole database backup using RMAN or operating system commands.
- **Tablespace backup:** If the recovery catalog is stored in a separate tablespace (as recommended) and the catalog database is operated in `Archivelog` mode, you can take an online backup of the tablespace containing the recovery catalog.
- **Export:** If the database containing the catalog is not very large, you can export the database at regular intervals. However, when the catalog database is quite large, export may take a very long time and consume a large amount of disk storage. Then you can export the schema containing the recovery catalog.

Always store the recovery catalog in a separate database from your target database. Also ensure that the files related to the catalog database are isolated on disks different from those containing the target database.

## Recovering the Recovery Catalog

- **Create a database from a previous backup of the recovery catalog database.**
- **Relocate the catalog into another database and import the data.**
- **Import the entire database from an export.**

ORACLE

17-25

Copyright © Oracle Corporation, 2001. All rights reserved.

### Recovering the Recovery Catalog

The strategy to recover a recovery catalog would depend on the nature of failure and the backup strategy in place.

If the database containing the recovery catalog is damaged, and has to be rebuilt, then you should consider the following recovery options:

- You can create a database from a previous backup of the recovery catalog database.
- You can decide to locate the catalog in another database. In that database, create a user and grant the user the `RECOVERY_CATALOG_OWNER` privilege. You can import the data from the export of the previous catalog owner into the schema of the newly created user.
- You can create a new database and import the entire database from an export of the recovery catalog database.

When the recovery catalog has been rebuilt, you should resynchronize the catalog with the control file of the target database immediately.

During resynchronization, Recovery Manager may add records for files that no longer exist, because files being recataloged are not verified. Remove such records by issuing the `CHANGE . . . UNCATALOG` command.

# Summary

**In this lesson, you should have learned that:**

- **Before using the recovery catalog, you must register the target database**
- **You should resynchronize the catalog frequently using the control file**
- **Scripts can be stored in the recovery catalog**

ORACLE



## Practice 17 Overview

**This practice covers the following topics:**

- **Creating the recovery catalog**
- **Registering a target database with the recovery catalog**
- **Listing the incarnation of a target database**
- **Storing a script in the recovery catalog and executing it**

ORACLE

## Practice 17 RMAN Recovery Catalog

1. Execute the `$HOME/STUDENT/LABS/crercts.sql` script to create the `recat` tablespace for the recovery catalog and the `rcuser` schema.
2. Connect to the recovery catalog database using RMAN. Create the catalog in the `recat` tablespace.
3. Connect to your target database and recovery catalog using RMAN.
4. Execute the command to resynchronize the control file and recovery catalog. What happens? Why?
5. Register the target database in the recovery catalog.
6. Using RMAN, list all the database incarnations registered in the catalog.
7. Enter the `RESET DATABASE` command at the RMAN prompt. What happens?
8. View the `$HOME/LABS/crebkup.sql` script. In `SQL*Plus` connect to your target database as `system/manager` and execute the script to create an online operating system copy of the `SAMPLE` tablespace datafile in your `$HOME/BACKUP` directory.
9. Using RMAN, add the backup made in step 8 to the recovery catalog.
10. Using RMAN, confirm that the datafile has been added to the recovery catalog.
11. Use the RMAN command to remove the backup of the `SAMPLE` tablespace datafile from the recovery catalog. Do not remove the file from the operating system.
12. Using `SQL*Plus`, connect to your recovery catalog database and query the `RC_DATAFILE_COPY` view to confirm that the datafile has been removed from the recovery catalog.
13. Create a script to make a whole database backup with following information:

Name of script:	nightback
Channel name:	dbnD (n is the student account number)
Channel type :	Disk
Format:	<code>\$HOME/BACKUP/RMAN/%b%d%s%p</code>
Level:	Database (No archive logs)
tag:	nback

DO NOT RUN THIS SCRIPT NOW.
14. Use the `PRINT` command to query the recovery catalog and verify the script creation.

# 18

## Transporting Data Between Databases

ORACLE

Copyright © Oracle Corporation, 2001. All rights reserved.

# Objectives

**After completing this lesson, you should be able to do the following:**

- **Describe the uses of the Export and Import utilities**
- **Describe Export and Import concepts and structures**
- **Perform simple Export and Import operations**
- **List guidelines for using Export and Import**

ORACLE

# Oracle Export and Import Utility Overview

**You can use these utilities to do the following:**

- **Archive historical data**
- **Save table definitions to protect them from user error failure**
- **Move data between machines and databases or between different versions of the Oracle server**
- **Transport tablespaces between databases**

ORACLE

18-3

Copyright © Oracle Corporation, 2001. All rights reserved.

## Export and Import Utility Overview

The Export utility provides a simple way for you to transfer data objects between Oracle databases, even if they reside on platforms with different hardware and software configurations. The Export utility can provide a logical backup of:

- Database objects
- A tablespace
- An entire database

The Import utility is used to read a valid Export file for moving data into a database. Redo log history cannot be applied to objects that are imported from an export file, therefore data loss may occur, but can be minimized. The DBA can use the Export and Import utilities to complement normal operating system backups by using them to:

- Create a historical archive of a database object or entire database; for example, when a schema is modified to support changing business requirements.
- Save table definitions in a binary file. This may be useful for creating and maintaining a baseline of a given schema structure.
- Move data from one Oracle database version to another, such as upgrading from Oracle8i to Oracle9i.

## Export and Import Utility Overview (continued)

You use these utilities to protect against:

- User errors where a user may accidentally drop or truncate a table
- A table that has become logically corrupted
- An incorrect batch job or other DML statement that has affected only a subset of the database

You use these utilities to recover:

- A logical database to a point different from the rest of the physical database when multiple logical databases exist in separate tablespaces of one physical database
- A tablespace in a very large database (VLDB) when tablespace point-in-time recovery (TSPITR) is more efficient than restoring the whole database from a backup and rolling it forward

**Note:** This lesson addresses the Export and Import utilities and discusses how they affect backup and recovery operations. For a detailed description of these utilities, refer to the Oracle9i Server Utilities manual.

## Methods to Run the Export Utility

- **An interactive dialog**
- **The command-line interface**
- **Parameter files**
- **Oracle Enterprise Manager**

ORACLE

18-5

Copyright © Oracle Corporation, 2001. All rights reserved.

### Export Methods

You can invoke Export and specify parameters by using any of the following methods:

- Command-line entries
- Interactive Export prompts
- Parameter files
- Oracle Enterprise Manager

To use Export, you must have the `CREATE SESSION` privilege on an Oracle database. To export tables owned by another user, you must have the `EXP_FULL_DATABASE` role enabled. This role is granted to all DBAs. If you do not have the system privileges contained in the `EXP_FULL_DATABASE` role, you cannot export objects contained in another user's schema.

**Note:** Many options are only available by using the command-line interface. However, you can use a parameter file with command line.

## Export Modes

Table Mode	User Mode	Tablespace Mode	Full Database Mode
Table definitions	Tables definitions	Table definitions	Tables definitions
Table data (all or selected rows)	Tables data		Tables data
Owner's table grants	Owner's grants	Grants	Grants
Owner's table indexes	Owner's indexes	Indexes	Indexes
Table constraints	Tables constraints	Table constraints	Tables constraints
		Triggers	

ORACLE

18-6

Copyright © Oracle Corporation, 2001. All rights reserved.

### Table Mode

Table mode exports specified tables in the user's schema, rather than exporting all tables. A privileged user can export specified tables owned by other users.

### User Mode

User mode exports all objects for a user's schema. Privileged users can export all objects in the schemas of a specified set of users. This mode can be used to complement a full database export.

### Tablespace Mode

You can use transportable tablespaces to move a subset of an Oracle database and plug it into another Oracle database, essentially moving tablespaces between the databases.

Moving data by way of transportable tablespaces can be much faster than performing either an import/export of the same data, because transporting a tablespace only requires the copying of data files and integrating the tablespace structural information. You can also use transportable tablespaces to move index data, thereby avoiding the index rebuilds you would have to perform when importing table data.

### Full Database Mode

Full database mode exports all database objects, except those in the SYS schema. Only privileged users can export in this mode.

**Note:** This is a partial treatment. For a full treatment of modes and objects, see *Oracle9i Utilities Guide*, Part No. A86728-01.



# Command-Line Export

## Syntax

```
exp keyword = (value, value2, ... ,valuen)
```

## Example

```
exp hr/hr TABLES=(employees,departments) rows=y  
file=exp1.dmp
```

```
exp system/manager OWNER=hr direct=y  
file=expdat.dmp
```

```
exp system/manager TRANSPORT_TABLESPACE=y  
TABLESPACES=(ts_emp) log=ts_emp.log
```

```
exp system/manager FULL=y inctype=cumulative  
file=expcum1.dmp
```

ORACLE

18-7

Copyright © Oracle Corporation, 2001. All rights reserved.

## Command Line Export

You can copy database data to an operating system file by using the command line mode of the Export utility. This file is only readable by the Import utility.

### Example

Create an export file named `exp1.dmp` that includes the tables `EMPLOYEES` and `DEPARTMENTS` from `HR` schema including rows:

```
$ exp hr/hr tables=(employees,departments) rows=y  
file=exp1.dmp
```

Create a fast export file named `expdat.dmp` that includes all objects for `HR` schema:

```
$ exp system/manager owner=hr direct=y
```

If file is not specified, the default export file name is `expdat.dmp`.

Create an export file named `expdat.dmp` that includes all definitions of objects belonging to the tablespace `ts_employees` and generate a log file named `ts_employees.log`:

```
$ exp system/manager TRANSPORT_TABLESPACE=y  
TABLESPACES=(ts_employees) LOG=ts_employees.log
```

Create an export file named `expcum1.dmp` that includes all definitions and data modified in the database since the last cumulative or complete export.

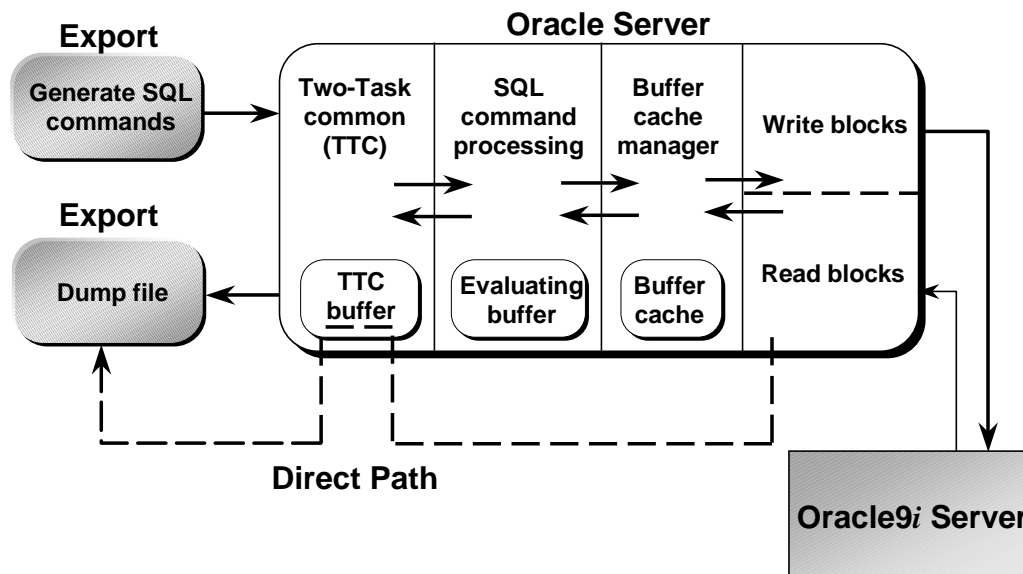
```
$ exp system/manager FULL=y INCTYPE=cumulative  
FILE=expcum1.dmp
```

## Export Parameters

Parameter	Description
USERID	Username/password of schema objects to export
FILE	Name of output file
ROWS	Include table rows in export file: (Y)es/(N)o
FULL	Export entire database: (Y)es/(N)o
OWNER	Users to export: Username
TABLES	Tables to export: List of tables
INDEXES	Indexes to export: (Y)es/(N)o
DIRECT	Specify direct mode export: (Y)es/(N)o
INCTYPE	Type of export level
PARFILE	Name of file in which parameters are specified
HELP	Display export parameters in interactive mode (Y)
LOG	Name of file for informational and error messages
CONSISTENT	Read-consistent view of the database when data is updated during an export: (Y)es/(N)o
BUFFER	Size of the data buffer in bytes: (Integer)
TRANSPORT_TABLESPACE	Enables the export of transportable tablespace metadata (release 8.1 only)
TABLESPACES	Tablespaces to be transported (release 8.1 only)
POINT_IN_TIME_RECOVER	Indicates whether or not the Export utility exports one or more tablespaces in an Oracle database (release 8.0 only)
RECOVERY_TABLESPACES	Specifies the tablespaces that will be recovered by using point-in-time recovery (release 8.0 only) Refer to the Oracle Server Readme, release 8.0.4
COMPRESS	Specified to include all data in one extent: (Y)es/(N)o

**Note:** The parameters listed above are not a complete list of all Export utility parameters but are those that a DBA may often use for restoring a database.

## Direct-Path Export Concepts



ORACLE

18-9

Copyright © Oracle Corporation, 2001. All rights reserved.

### Direct-Path Export Concepts

By using the Direct-Path feature, you can extract data much faster. When the `DIRECT=Y` parameter is specified, the Export utility reads directly from the data layer instead of going through the SQL-command processing layer.

#### Mechanics of Direct-Path Export

- Direct mode of export can be set by specifying the `DIRECT=Y` parameter.
- Direct-path export does not compete with other resources of the Instance.
- If in direct-read mode, it reads database blocks into a private area used by the session.
- Rows are transferred directly into the Two-Task Common (TTC) buffer for transport.
- The data in the TTC buffer is in the format that the Export utility expects.

## Specifying Direct-Path Export

As command line argument to the Export command:

```
exp userid=hr/hr full=y direct=y
```

As a keyword in a parameter file:

```
exp parfile=<Parameter file>
```

### Parameter file

```
.....(Other Paramaters)  
DIRECT = Y  
.....(Other Paramaters)
```

ORACLE

18-10

Copyright © Oracle Corporation, 2001. All rights reserved.

## Specifying Direct-Path Export

Before direct-path Export can be used, you must make sure the `catexp.sql` script has been run. It can be found in `$HOME/rdbms/admin/`.

### Using the DIRECT Parameter

#### Command-Line Option

You can invoke direct-path export by using the `DIRECT` command-line parameter at the operating system prompt.

```
$ exp user=hr/hr full=y direct=y
```

#### Parameter File

An example of a parameter file, `exp_par.txt`:

```
USERID=hr/hr  
TABLES=(employees,departments)  
FILE=exp_one.dmp  
DIRECT=Y
```

To execute the parameter from the operating system prompt:

```
$ exp parfile=exp_param.txt
```

## Direct-Path Export Features

- The type of Export is indicated on the screen output, export dump file, and the log file.
- Data is already in the format that Export expects, avoiding unnecessary data conversion.
- Uses an optimized SQL `SELECT` statement.

ORACLE

18-11

Copyright © Oracle Corporation, 2001. All rights reserved.

### Direct-Path Export

The direct-path option of the Export utility introduces some features that differentiate it from the conventional-path Export.

#### Direct-Path Features

- The type of export is indicated on the screen output, export dump file, and the log file specified with the LOG parameter.
- Data is already in the format that Export expects, thus avoiding unnecessary data conversion. The data is transferred to Export client, which then writes the data into the Export file.
- The direct-path Export uses an optimized `SELECT * FROM table` without any predicate.

**Note:** The format of the column data and specification in the Export dump file differ from those of conventional-path Export.

## Direct-Path Export Restrictions

- The direct-path option cannot be invoked interactively.
- Client-side and server-side character sets must be the same.
- The `BUFFER` parameter has no affect.
- You cannot use the direct-path option to export rows containing `LOB`, `BFILE`, `REF`, or object types.

ORACLE

18-12

Copyright © Oracle Corporation, 2001. All rights reserved.

### Direct-Path Restrictions

The direct-path option of the Export utility has some restrictions that differentiate it from the conventional-path Export.

- The direct-path export feature cannot be invoked by using an interactive EXP session.
- When the direct-path option is used, the client-side character set must match the character set of the server side. Use the `NLS_LANG` environment variable to set the same character set as the server one.
- The `BUFFER` parameter of the Export utility has no effect on direct-path Export; it is used only by the conventional-path option.
- You cannot direct-path Export rows that contain the datatypes `LOB`, `BFILE`, `REF`, or object type columns, including `VARRAY` columns and nested tables. Only the data definition to create the table is exported, not the data.

## Uses of the Import Utility for Recovery

- **Create table definitions**
- **Extract data from a valid Export file**
- **Import from a complete or cumulative Export file**
- **Recover from user-error failures**

ORACLE

18-13

Copyright © Oracle Corporation, 2001. All rights reserved.

### Import Utility

The Import utility reads the object definitions and table data from an Export dump file. It inserts the data objects into an Oracle database. Import functionality includes:

- Creating table definitions since the table definitions are stored in the Export file. Choosing to import data without rows will create just the table definitions.
- Extracting data from a valid export file by using the Table, User, Tablespace, or Full Import modes
- Importing data from a complete or cumulative Export file
- Recovering from user failure errors where a table is accidentally dropped or truncated by using one the previously mentioned methods

# Import Modes

Mode	Description
Table	Import specified tables into a schema.
User	Import all objects that belong to a schema
Tablespace	Import all definitions of the objects contained in the tablespace
Full Database	Import all objects from the export file

ORACLE

18-14

Copyright © Oracle Corporation, 2001. All rights reserved.

## IMPORT Modes

### Table Mode

Table mode imports all specified tables in the user's schema, rather than all tables. A privileged user can import specified tables owned by other users.

### User Mode

User mode imports all objects for a user's schema. Privileged users can import all objects in the schemas of a specified set of users.

### Tablespace Mode

Tablespace mode allows a privileged user to move a set of tablespaces from one Oracle database to another.

### Full Database Mode

Full database mode imports all database objects, except those in the SYS schema. Only privileged users can import in this mode.



# Command-Line Import

## Syntax

```
imp keyword = value or keyword = (value,  
value2, ... value n)
```

## Example

```
imp hr/hr TABLES=(employees,departments) rows=y  
file=exp1.dmp
```

```
imp system/manager FROMUSER=hr file=exp2.dmp
```

```
imp system/manager TRANSPORT_TABLESPACE=y  
TABLESPACES=ts_employees
```

ORACLE

18-15

Copyright © Oracle Corporation, 2001. All rights reserved.

## Examples

Import into HR schema, the tables EMPLOYEES and DEPARTMENTS, including rows, by using the Export file named exp1.dmp.

```
$ imp hr/hr tables=(employees,departments) rows=y \  
file=exp1.dmp
```

Import all objects belonging to the HR schema, including rows, by using the export file named exp2.dmp.

```
$ imp system/manager FROMUSER=hr file=exp2.dmp
```

Import all definitions of objects belonging to the tablespace ts\_employees, by using the export file named expdat.dmp.

```
$ imp system/manager TRANSPORT_TABLESPACE=y  
TABLESPACES=ts_employees
```

If the file parameter is not specified, Import looks for the default file; expdat.dmp.

**Note:** Command-line mode options are similar to interactive mode options but offer more functionality.

## Import Parameters

Parameter	Description
USERID	Username/password of schema objects to export
FILE	Name of the input file; must be a valid Export Utility file
ROWS	Include table rows in import file
IGNORE	Ignore create errors due to an object's existence
FULL	Import entire file
TABLES	Tables to import
INDEXES	Indexes to import
INCTYPE	Specifies the type of incremental import; options are <code>SYSTEM</code> and <code>RESTORE</code>
PARFILE	Parameter specification file
HELP	Display export parameters in interactive mode
LOG	File for informational and error messages
DESTROY	Specifies whether or not the existing data file making up the database should be reused
FROMUSER	A list of schemas containing objects to import
TOUSER	Specifies a list of usernames whose schemas will be imported
INDEXFILE	Specifies a file to receive index-creation
TRANSPORT_TABLESPACE	Instructs Import to import transportable tablespace metadata from an export file
TABLESPACES	List of tablespaces to be transported into the database
DATAFILES	List datafiles to be transported into the database
TTS_OWNERS	List the users who own the data in the transportable tablespace set

**Note:** The parameters listed above are not a complete list of all Import utility parameters but are those that a DBA will often use to restore a database.

## Invoking Import as SYSDBA

**You do not need to invoke Import as SYSDBA except:**

- **At the request of Oracle technical support**
- **When importing a transportable tablespace set**

**To invoke Import as SYSDBA:**

```
imp \ 'username/password AS  
SYSDBA\ '
```

ORACLE

18-17

Copyright © Oracle Corporation, 2001. All rights reserved.

### Importing as SYSDBA

SYSDBA is used internally and has specialized functions. Its behavior is not the same as for generalized users. Therefore, you should not typically need to invoke Import as SYSDBA, except in the following situations:

- At the request of Oracle technical support
- When importing a transportable tablespace set

To invoke Import as SYSDBA, use the following syntax, adding any desired parameters or parameter file names:

```
imp \ 'username/password AS SYSDBA\ '
```

or, optionally:

```
imp \ 'username/password@instance AS SYSDBA\ '
```

If either the username or password is omitted, Import will prompt you for it. This example shows the entire connect string enclosed in quotation marks and backslashes. This is because the string, AS SYSDBA, contains a blank; a situation for which most operating systems require that the entire connect string be placed in quotation marks or marked as a literal by some method. Additionally, some operating systems also require that quotation marks on the command line be preceded by an escape character. In this example, backslashes are used as the escape character. If the backslashes were not present, the command line parser that Export uses would not understand the quotation marks and would remove them before calling Export.

## Import Process Sequence

1. New tables are created
2. Data is imported
3. Indexes are built
4. Triggers are imported
5. Integrity constraints are enabled on the new tables
6. Any bitmap, functional, and/or domain indexes are built

ORACLE

18-18

Copyright © Oracle Corporation, 2001. All rights reserved.

### Import Process Sequence

When importing a table, the export file is read and the table and data are created in the following order:

1. New tables are created
2. Data is imported
3. Indexes are built
4. Triggers are imported
5. Integrity constraints are enabled on the new tables
6. Any bitmap, functional, and/or domain indexes are built

This sequence prevents data from being rejected due to the order in which tables are imported. This sequence also prevents redundant triggers from firing twice on the same data (once when it is originally inserted and again during the import).

The order in which you import tables may be important if you do not import all the objects that a user owns. For, example, if the table with the foreign key has a referential check on the table with the primary key, and the foreign key table is imported first, then all rows that reference the primary key that have not been imported will be rejected if the constraints were enabled. For a full database Export this is not a problem.

## National Language Support Considerations

- **The Export file identifies the character encoding scheme used for the character data in the file**
- **The Import utility translates data to the character set of its host system**
- **A multibyte character set Export file must be imported into a system that has the same characteristics**

ORACLE

18-19

Copyright © Oracle Corporation, 2001. All rights reserved.

### National Language Support Considerations

When moving data from one Oracle database by one character set to a database with a different character set, ensure the data conversion is handled appropriately. You can do this by setting the `NLS_LANG` environment variable to the character set definition of the database from which the data is being exported. Not setting this correctly could cause unwanted conversion of characters in the data, possibly causing loss of data.

#### Examples

Converting from a 7-bit ASCII character set, such as American English, to an 8-bit character set such as Danish, no conversion is needed because all characters have an equivalent character in the Danish alphabet.

When converting from an 8-bit ASCII character set, such as Danish, to a 7-bit character set such as American English, the extra Danish characters that are not found in the American alphabet may be converted to question marks (?). In this case the question marks are substituted for the unknown Danish characters which, is the appropriate result.

In the 8-bit to 8-bit data movement, whether characters are lost depends upon the specifics of the languages used to enter the data. For example, the Spanish alphabet has letters that are not in the Danish alphabet, so moving data from a Spanish database to a Danish one might result in possible conversion and therefore possible loss of those characters.

# Summary

**In this lesson, you should have learned how to:**

- **Describe the uses of Export and Import**
- **Describe Export and Import concepts and structures**
- **Perform simple Export and Import operations**
- **List guidelines for using Export and Import**

**ORACLE**

## Practice 18 Overview

This practice covers the following topics:

- Use of the Export utility
- Use of the Import utility

ORACLE

## **Practice 18**

1. Invoke the Export utility to export the EMPLOYEES and DEPARTMENTS tables in the hr schema.
2. Start SQL\*Plus and connect as HR. Drop the EMPLOYEES and DEPARTMENTS tables.
3. Restore the EMPLOYEES and DEPARTMENTS tables by using the import utility.
4. Query the EMPLOYEES and DEPARTMENTS tables to get the number of rows in each of those tables.



# 19

## Loading Data into a Database

ORACLE

Copyright © Oracle Corporation, 2001. All rights reserved.

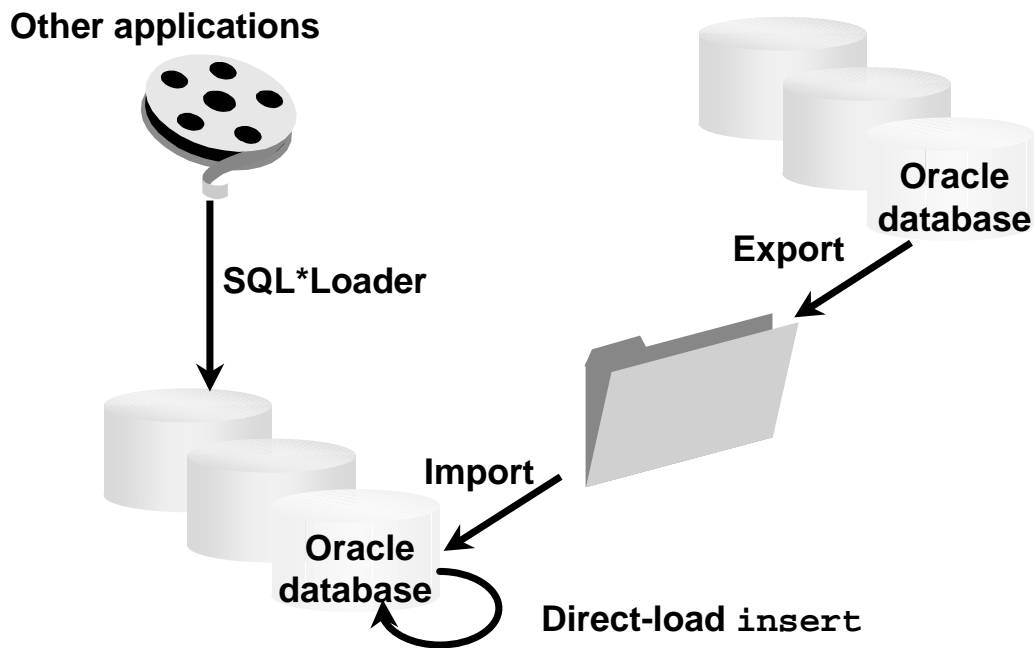
# Objectives

**After completing this lesson, you should be able to do the following:**

- **Demonstrate usage of direct-load `insert` operations**
- **Describe the usage of SQL\*Loader**
- **Perform basic SQL\*Loader operations**
- **List guidelines for using SQL\*Loader and direct-load `insert`**

ORACLE

# Data Loading Methods



ORACLE

19-3

Copyright © Oracle Corporation, 2001. All rights reserved.

## Loading Data

Several methods are available for loading data into tables in an Oracle database. Of the methods available **Direct-Load insert** and **SQL\*Loader** are discussed here. **Export** and **Import** will be discussed in another lesson.

### SQL\*Loader

**SQL\*Loader** loads data from external files into tables of an Oracle database. It has a powerful data parsing engine that puts little limitation on the format of the data in the datafile.

### Direct-Load insert

**Direct-load insert** can be used to copy data from one table to another table within the same database. It speeds up the insert operation, bypassing the buffer cache and writing data directly into the data files.

# Direct-Load INSERT

**Direct-load INSERT can be performed in the following ways:**

- **Normal (serially) or in parallel**
- **Into partitioned tables, nonpartitioned tables, or single partitions of a table**
- **With or without logging of redo data**

ORACLE

19-4

Copyright © Oracle Corporation, 2001. All rights reserved.

## Direct-Load INSERT

Direct-load INSERT (serial or parallel) can only support the INSERT ... SELECT syntax of an INSERT statement, not the INSERT ... values syntax. The parallelism for INSERT ... SELECT is determined from either parallel hints or parallel table definition. Oracle9i provides syntax extensions that extend the scope of the INSERT ... SELECT statement, so that you can insert rows into multiple tables as part of a single DML statement.

### **Serial direct-load INSERT into a nonpartitioned, partitioned, or subpartitioned table.**

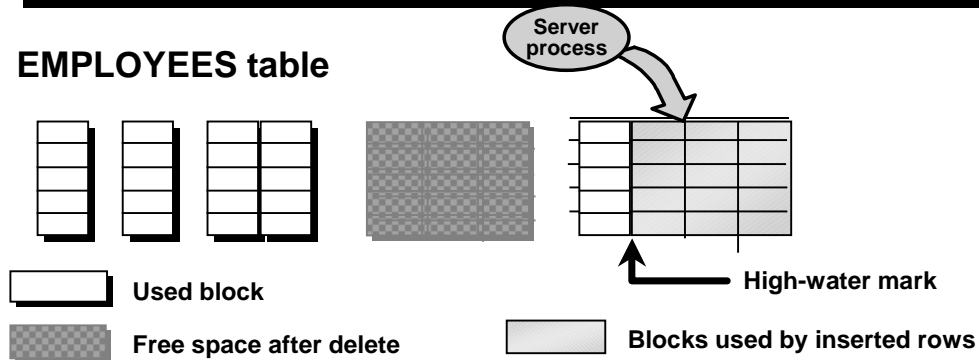
Data is inserted beyond the current high water mark of the table segment or each partition segment. The *high-water mark* is the level at which blocks have never been formatted to receive data. When a statement executes, the high water mark is updated to the new value, making the data visible to others. When loading a partitioned or subpartitioned table, SQL\*Loader partitions the rows and maintains indexes (which can also be partitioned). A direct path load of a partitioned or subpartitioned table can be quite resource-intensive.

**Parallel direct-load INSERT into a nonpartitioned table.** Each parallel execution server allocates a new temporary segment and inserts data into the temporary segment. When a statement executes, the parallel execution coordinator merges the new temporary segments into the primary table segment.

## Serial Direct-Load Inserts

```
INSERT /*+ APPEND */ INTO emp
NOLOGGING
SELECT * FROM t_employees;
COMMIT;
```

**EMPLOYEES table**



ORACLE

19-5

Copyright © Oracle Corporation, 2001. All rights reserved.

### Direct-Load INSERT (continued)

**Parallel direct-load INSERT into a partitioned table.** Each parallel execution server is assigned one or more partitions, with no more than one process working per partition. The parallel execution server inserts data beyond the current high water mark of the partition segments assigned to it. When a statement executes, the high water mark of each partition segment is updated by the parallel execution coordinator to the new value, making the data visible to others.

A direct-load insert can be invoked by using the APPEND hint, as shown in the command below:

```
INSERT /*+APPEND */ INTO [ schema. ] table
[ [NO]LOGGING ]
sub-query;
```

**where:** *schema* is the owner of the table

*table* is the name of the table

*sub-query* is the subquery used to select the columns and rows for insert

## **Direct-Load INSERT (continued)**

### **LOGGING Mode**

When inserting using the `LOGGING` option, which is the default, the operation generates redo log entries, making complete recovery possible in case of failures. If the `NOLOGGING` option is used, changes to data are not recorded in the redo log buffer. Some minimal logging still occurs for operations that update the data dictionary. The `NOLOGGING` mode is used if this attribute has been set for the table.

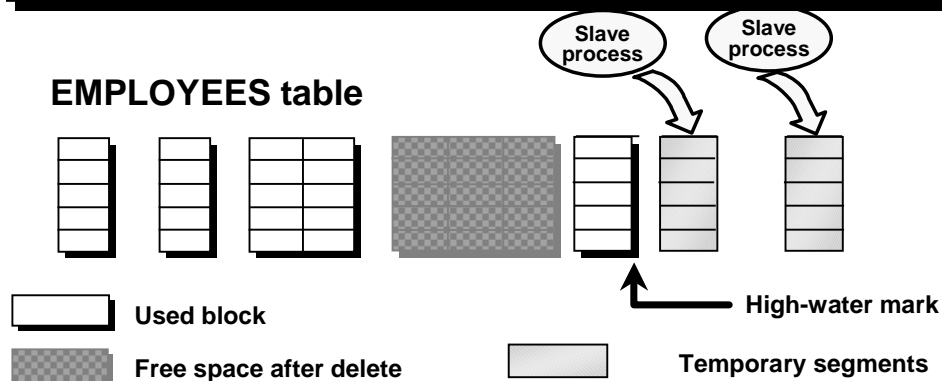
If several online modifications to the data in the table are likely to occur subsequently, it might be advisable to set the `NOLOGGING` attribute before the load and reset it to `LOGGING` once the load is completed.

### **Other Considerations**

All data loaded using direct-load `insert` is loaded above the high-water mark. If the table contains many blocks where rows have been deleted, space may be wasted and full table scans may be slower.

## Parallel Direct-Load Insert

```
ALTER SESSION ENABLE PARALLEL DML;  
INSERT /*+PARALLEL(hr.employees,2) */  
      INTO hr.employees NOLOGGING  
SELECT * FROM hr.old_employees;
```



ORACLE

19-7

Copyright © Oracle Corporation, 2001. All rights reserved.

### Parallel Direct-Load INSERT

Direct-load inserts can be made in parallel using one of the following methods:

- Using a `PARALLEL` hint in the `INSERT` statement, as in the example
- Creating the table or altering it to specify the `PARALLEL` clause

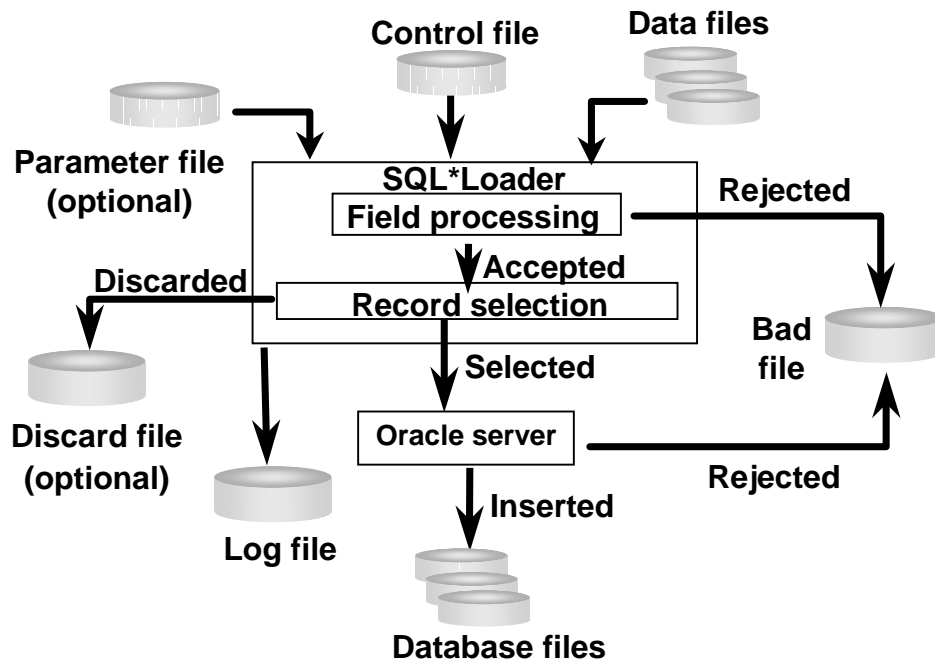
When parallel direct-load inserts are made, the Oracle server uses several processes, known as parallel query slaves, to insert data into the table. Temporary segments are allocated to store the data inserted by each slave process. When the transaction commits, the extents in these individual segments become a part of the table in which records are inserted.

### Note

- The `ALTER SESSION ENABLE PARALLEL DML` command must be executed at the beginning of a transaction.
- An object that is modified using parallel direct-load insert cannot be queried or modified again within the same transaction.

For a detailed discussion of parallel direct-load inserts, see the *Oracle9i Reference*, "Parallel Execution."

# SQL\*Loader



19-8

Copyright © Oracle Corporation, 2001. All rights reserved.

## SQL\*Loader Features

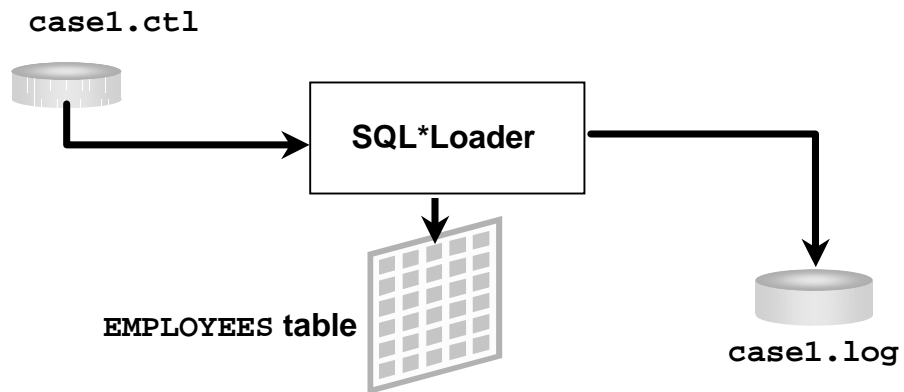
SQL\*Loader loads data from external files into tables in an Oracle database. SQL\*Loader has the following features:

- SQL\*Loader can use one or more input files
- Several input records can be combined into one logical record for loading
- Input fields can be fixed or variable length
- Input data can be in any format: character, binary, packed decimal, date, and zoned decimal
- Data can be loaded from different types of media such as disk, tape, or named pipes
- Data can be loaded into several tables in one run
- Options are available to replace or to append to existing data in the tables
- SQL functions can be applied on the input data before the row is stored in the database
- Column values can be autogenerated based on rules. For example, a sequential key value can be generated and stored in a column
- Data can be loaded directly into the table, bypassing the database buffer cache



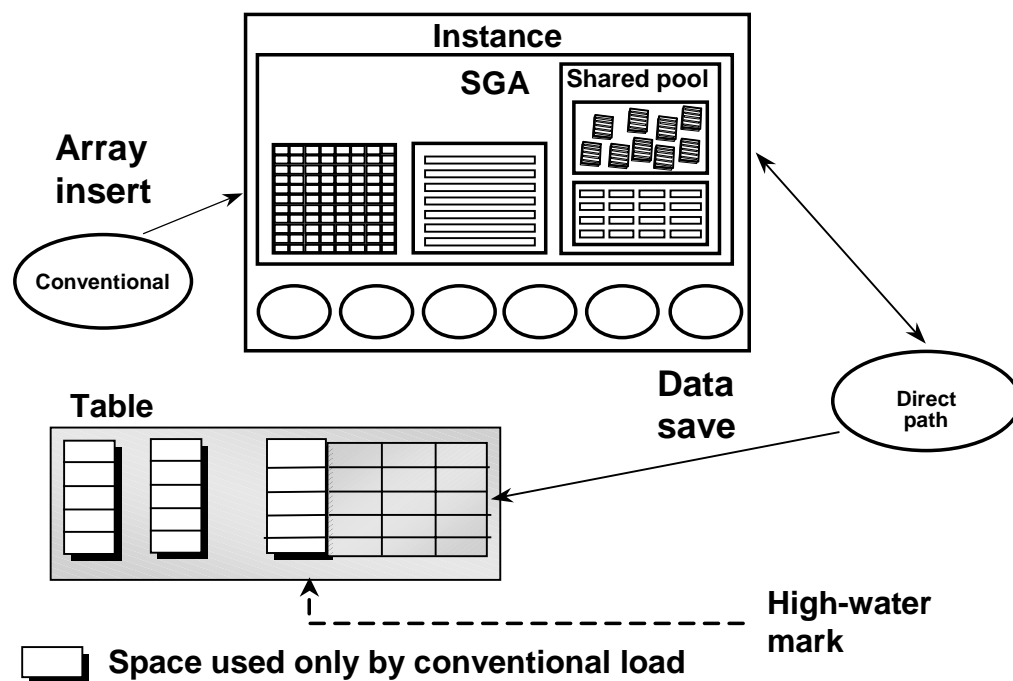
## Using SQL\*Loader

```
$sqlldr hr/hr \  
> control=case1.ctl \  
> log=case1.log direct=Y
```



ORACLE

## Conventional and Direct Path Loads

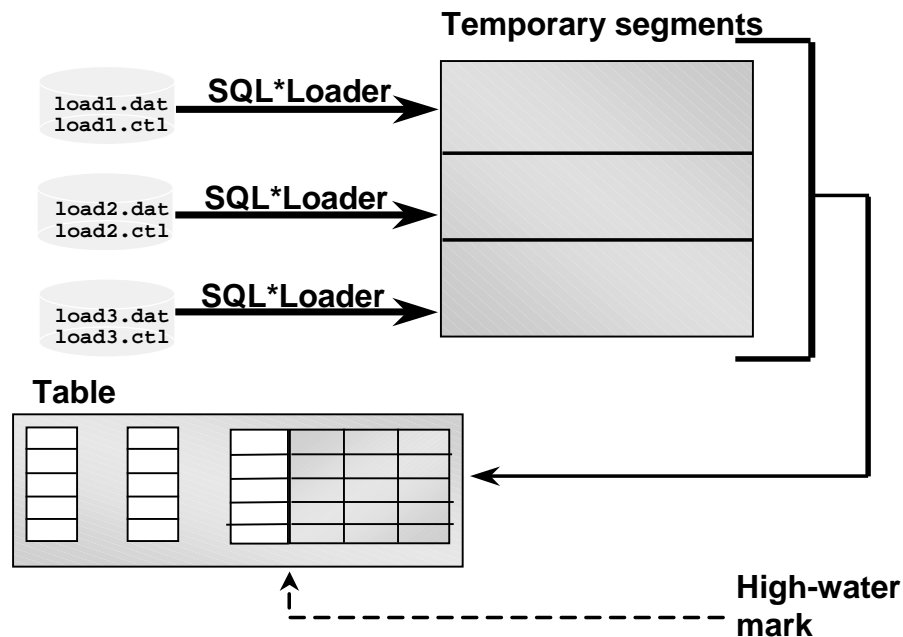


## Comparing Direct and Conventional Path Loads

Conventional Load	Direct Path Load
Uses <b>COMMITs</b> to make changes permanent	Uses data saves
Redo log entries always generated	Generates redo only under specific conditions
Enforces all constraints	Enforces only primary key, unique, and NOT NULL
<b>INSERT</b> triggers fire	<b>INSERT</b> triggers do not fire
Can load into clustered tables	Cannot load into clustered tables
Other users can make changes to tables	Other users cannot make changes to tables

ORACLE

## Parallel Direct-Path Load



ORACLE

19-12

Copyright © Oracle Corporation, 2001. All rights reserved.

### Parallel Direct-Path Load

Multiple SQL\*Loader sessions improve the performance of a direct path load. There are three models of concurrency that can be used to minimize the time required for data loading:

- Parallel conventional path loads
- Intersegment concurrency with direct path load method
- Intrasegment concurrency with direct path load method

#### Concurrent conventional path

If triggers or integrity constraints pose a problem, but faster loading is desired, you should consider using concurrent conventional path loads. Use multiple load sessions executing concurrently on a multiple-CPU system. Split the input datafiles into separate files on logical record boundaries, then load each such input datafile with a conventional path load session.

#### Intersegment concurrency

Intersegment concurrency can be used for concurrent loading of different objects. This technique can be applied for concurrent direct path loading of different tables, or to concurrent direct path loading of different partitions of the same table.

#### Intrasegment concurrency

A parallel direct path load allows multiple direct path load sessions to concurrently load the data into the same table, or into the same partition of a partitioned table allowing intrasegment parallelism.

# SQL\*Loader Control File

**The control file tells SQL\*LOADER:**

- **Where to find the load data**
- **The data format**
- **Configuration details**
  - **Memory management**
  - **Record rejection**
  - **Interrupted load handling details**
- **How to manipulate the data**

ORACLE

19-13

Copyright © Oracle Corporation, 2001. All rights reserved.

## The SQL\*Loader Control File

The SQL\*Loader control file is a text file that contains data definition language (DDL) instructions. DDL is used to control the following aspects of a SQL\*Loader session:

- Where SQL\*Loader finds the data to load
- How SQL\*Loader expects that data to be formatted
- How SQL\*Loader configures (memory management, rejecting records, interrupted load handling, and so on) as it loads the data
- How SQL\*Loader manipulates the data being loaded

Although not precisely defined, a control file can be said to have three sections.

- The first section contains session wide information, for example:
  - Global options such as bindsize, rows, records to skip, and so on
  - INFILE clauses to specify where the input data is located
  - How data is to be loaded
- The second section consists of one or more INTO TABLE blocks. Each of these blocks contains information about the table into which the data is to be loaded, such as the table name and the columns of the table.
- The third section is optional and, if present, contains input data.

## The SQL\*Loader Control File (continued)

The example below illustrates a typical SQL\*Loader control file.

```
1  -- This is a sample control file
2  LOAD DATA
3  INFILE 'SAMPLE.DAT'
4  BADFILE 'sample.bad'
5  DISCARDFILE 'sample.dsc'
6  APPEND
7  INTO TABLE emp
8  WHEN (57) = '.'
9  TRAILING NULLCOLS
10 (hiredate SYSDATE,
    deptno POSITION(1:2) INTEGER EXTERNAL(3)
    NULLIF deptno=BLANKS,
    job POSITION(7:14) CHAR TERMINATED BY WHITESPACE
    NULLIF job=BLANKS "UPPER(:job)",
    mgr POSITION(28:31) INTEGER EXTERNAL
    TERMINATED BY WHITESPACE, NULLIF mgr=BLANKS,
    ename POSITION(34:41) CHAR
    TERMINATED BY WHITESPACE "UPPER(:ename)",
    empno POSITION(45) INTEGER EXTERNAL
    TERMINATED BY WHITESPACE,
    sal POSITION(51) CHAR TERMINATED BY WHITESPACE
    "TO_NUMBER(:sal, '$99,999.99')",
    comm INTEGER EXTERNAL ENCLOSED BY '(' AND '%'
    ":comm * 100"
)
```

### Sample control file explanation

1. This is how comments are entered in a control file. Comments can appear anywhere in the command section of the file, but they should not appear within the data.
2. The LOAD DATA statement tells SQL\*Loader that this is the beginning of a new data load. If you were continuing a load that had been interrupted in progress, you would use the CONTINUE LOAD DATA statement.
3. The INFILE keyword specifies the name of a datafile containing data that you want to load.

### **The SQL\*Loader Control File (continued)**

4. The **BADFILE** keyword specifies the name of a file into which rejected records are placed.
5. The **DISCARDFILE** keyword specifies the name of a file into which discarded records are placed.
6. The **APPEND** keyword is one of the options you can use when loading data into a table that is not empty. To load data into a table that is empty, you would use the **INSERT** keyword.
7. The **INTO TABLE** keyword allows you to identify tables, fields, and datatypes. It defines the relationship between records in the datafile and tables in the database.
8. The **WHEN** clause specifies one or more field conditions, which each record must match before SQL\*Loader will load the data. In this example SQL\*Loader will only load the record if the 57<sup>th</sup> character is a decimal point. That decimal point delimits dollars and cents in the field and causes records to be rejected if SAL has no value.
9. The **TRAILING NULLCOLS** clause tells SQL\*Loader to treat any relatively positioned columns that are not present in the record as null columns.
10. The remainder of the control file contains the field list, which provides information about column formats in the table being loaded.

## Control File Syntax Considerations

- **The syntax is free-format**
- **Syntax is case insensitive**
- **Comments extend from the two hyphens ( -- ) that mark the beginning of the comment to the end of the line**
- **The `CONSTANT` keyword is reserved**

ORACLE

19-16

Copyright © Oracle Corporation, 2001. All rights reserved.

### Control File Syntax Considerations

- The syntax is free-format (statements can extend over multiple lines).
- It is case insensitive; however, strings enclosed in single or double quotation marks are taken literally, including case.
- In control file syntax, comments extend from the two hyphens (--) that mark the beginning of the comment to the end of the line. The optional third section of the control file is interpreted as data rather than as control file syntax; consequently, comments in this section are not supported.
- The `CONSTANT` keyword has special meaning to SQL\*Loader and is therefore reserved. Therefore, to avoid potential conflicts, It is recommended that you do not use the word `CONSTANT` as a name for any tables or columns.



# Input Data and Datafiles

- **SQL\*Loader reads data from one or more files specified in the control file**
- **From SQL\*Loader's perspective, the data in the datafile is organized as *records***
- **A datafile can be in one of three formats:**
  - **Fixed-record format**
  - **Variable-record format**
  - **Stream-record format**

ORACLE

19-17

Copyright © Oracle Corporation, 2001. All rights reserved.

## Datafile Formats

### Fixed Record Format

A file is in fixed record format when all records in a datafile are the same byte length. Although this format is the least flexible, it results in better performance than variable or stream format. Fixed-record format is also simple to specify. For example:

```
INFILE <datafile_name> "fix n"
```

This example specifies that SQL\*Loader should interpret the particular datafile as being in fixed-record format where every record is *n* bytes long.

The example below shows a control file that specifies a fixed record format datafile. The datafile contains four physical records. The first record is [0001, abcd] which is exactly nine bytes long (using a single-byte character set) and the carriage return is the tenth byte.

```
load data
infile 'example.dat' "fix 10"
into table example
fields terminated by ','
(col1, col2)

example.dat:
0001,abcd
0002,fg
hi
0003,klmn
```

## Datafile Formats (continued)

### Variable-Record Format

A file is in variable-record format when the length of each record in a character field is included at the beginning of each record in the datafile. This format provides some added flexibility over the fixed-record format and a performance advantage over the stream record format. For example, you can specify a datafile that is to be interpreted as being in variable-record format as follows:

```
INFILE "datafile_name" "var n"
```

In this example, *n* specifies the number of bytes in the record length field. If *n* is not specified, SQL\*Loader assumes a length of 5. Specifying *n* larger than 40 will result in an error. The following example shows a control file specification that tells SQL\*Loader to look for data in the datafile `example.dat` and to expect variable-record format where the record length fields are 3 bytes long. The `example.dat` datafile consists of three physical records. The first is specified to be 009 (that is, 9) bytes long, the second is 010 bytes long (including a 1-character newline), and the third is 012 bytes long. This example also assumes a single-byte character set for the datafile.

```
load data
infile 'example.dat' "var 3"
into table example
fields terminated by ',' optionally enclosed by '"'
(col1 char(5),
col2 char(7))
```

```
example.dat:
009hello,cd,
010world,im,
012my,name is,
```

### Stream-Record Format

A file is in stream record format when the records are not specified by size; instead SQL\*Loader forms records by scanning for the *record terminator*. Stream record format is the most flexible format, but there can be a negative effect on performance. The specification of a datafile to be interpreted as being in stream-record format looks similar to the following:

```
INFILE <datafile_name> ["str terminator_string"]
```

The `terminator_string` is specified as either `'char_string'` or `X'hex_string'` where:

`'char_string'` is a string of characters enclosed in single or double quotation marks

`X'hex_string'` is a byte string in hexadecimal format

## Datafile Formats (continued)

When the `terminator_string` contains special (nonprintable) characters, it should be specified as a X'hex\_string'. However, some nonprintable characters can be specified as ('char\_string') by using a backslash. For example:

<code>\n</code>	linefeed (newline)
<code>\t</code>	horizontal tab
<code>\f</code>	formfeed
<code>\v</code>	vertical tab
<code>\r</code>	carriage return

If the character set specified with the `NLS_LANG` parameter for your session is different from the character set of the datafile, character strings are converted to the character set of the datafile.

Hexadecimal strings are assumed to be in the character set of the datafile, so no conversion is performed. If no `terminator_string` is specified, it defaults to the newline (end-of-line) character (line feed in UNIX-based platforms, carriage return followed by a line feed on Microsoft platforms, and so on). The newline character is connected to the character set of the datafile.

The following example illustrates loading data in stream record format where the terminator string is specified using a character string, '`|\n`'. The use of the backslash character allows the character string to specify the nonprintable linefeed character.

```
load data
infile 'example.dat' "str '|\n'"
into table example
fields terminated by ',' optionally enclosed by '"'
(col1 char(5),
col2 char(7))
example.dat:
hello,world,|
james,bond,|
```

# Logical Records

**SQL\*Loader can be instructed to follow one of the following two logical record-forming strategies:**

- **Combine a fixed number of physical records to form each logical record**
- **Combine physical records into logical records while a certain condition is true**

ORACLE

19-20

Copyright © Oracle Corporation, 2001. All rights reserved.

## Logical Records

SQL\*Loader organizes the input data into physical records, according to the specified record format. By default a physical record is a logical record, but for added flexibility, SQL\*Loader can be instructed to combine a number of physical records into a logical record. SQL\*Loader can do this in one of two ways:

- Combine a fixed number of physical records to form each logical record
- Combine physical records into logical records while a certain condition is true

### Using **CONCATENATE** to Assemble Logical Records

CONCATENATE is used when SQL\*Loader should always add the same number of physical records to form one logical record. The following is an example of using CONCATENATE, in which *integer* specifies the number of physical records to combine:

CONCATENATE *integer*

### Using **CONTINUEIF** to Assemble Logical Records

CONTINUEIF must be used if the number of physical records to be continued varies. The keyword CONTINUEIF is followed by a condition that is evaluated for each physical record as it is read. For example, two records might be combined if there were a pound sign (#) in character position 80 of the first record. If any other character were there, the second record would not be added to the first.

# Data Conversion

**During a conventional path load, datafields in the datafile are converted into columns in the database in two steps:**

- **The field specifications in the control file are used to interpret the format of the datafile and convert it to a `SQL INSERT` statement using that data**
- **The Oracle database server accepts the data and executes the `INSERT` statement to store the data in the database**

ORACLE

## Discarded or Rejected Records

- **Bad file**
  - SQL\*Loader rejects records when the input format is invalid
  - If the Oracle database finds that the row is invalid, the record is rejected and SQL\*Loader puts it in the bad file
- **Discard file**
  - This can be used only if it has been enabled
  - This file contains records that were filtered out because they did not match any record-selection criteria specified in the control file

ORACLE

19-22

Copyright © Oracle Corporation, 2001. All rights reserved.

### Handling Discarded or Rejected Records

#### The Bad File

The bad file contains records that were rejected, either by SQL\*Loader or by the Oracle database.

#### SQL\*Loader Rejects

Records are rejected by SQL\*Loader when the input format is invalid. For example, if the second enclosure delimiter is missing, or if a delimited field exceeds its maximum length, SQL\*Loader rejects the record. Rejected records are placed in the bad file.

#### Oracle Rejects

After a record is accepted for processing by SQL\*Loader, a row is sent to Oracle for insertion. If Oracle determines that the row is valid, then the row is inserted into the database. If not, the record is rejected, and SQL\*Loader puts it in the bad file. The row may be rejected, for example, because a key is not unique, because a required field is null, or because the field contains invalid data for the Oracle datatype.

#### The Discard File

As SQL\*Loader executes, it may create a file called the discard file. This file is created only when it is needed, and only if you have specified that a discard file should be enabled. The discard file contains records that were filtered out of the load because they did not match any record-selection criteria specified in the control file. The discard file therefore contains records that were not inserted into any table in the database. You can specify the maximum number of such records that the discard file can accept.

# Log File Contents

- **Header Information**
- **Global Information**
- **Table Information**
- **Datafile Information**
- **Table Load Information**
- **Summary Statistics**
- **Additional statistics for direct path loads and multithreading Information**

ORACLE

19-23

Copyright © Oracle Corporation, 2001. All rights reserved.

## Contents of the Log File

The Header Information Section contains the following entries:

- Date of the run
- Software version number

The Global Information section contains the following entries:

- Names of all input/output files
- Echo of command-line arguments
- Continuation character specification

The Table Information Section provides the following entries for each table loaded:

- Table name
- Load conditions, if any. That is, whether all records were loaded or only those meeting WHEN-clause criteria.
- INSERT , APPEND, or REPLACE specification
- The following column information:
  - If found in datafile, the position, length, datatype, and delimiter.
  - If specified, RECNUM , SEQUENCE , CONSTANT, or EXPRESSION
  - If specified, DEFAULTIF or NULLIF

## Contents of the Log File (continued)

If the SQL\*Loader control file contains any directives for loading datetime/interval datatypes, then the log file contains the keyword `DATETIME` or `INTERVAL` under the datatype heading. If applicable, the keyword `DATETIME` or `INTERVAL` is followed by the corresponding mask.

The Datafile Information section appears only for datafiles with data errors and provides the following entries:

- SQL\*Loader and Oracle data record errors
- Records discarded

The Table Load Information section provides the following entries for each table that was loaded:

- Number of rows loaded
- Number of rows that qualified for loading but were rejected due to data errors
- Number of rows that were discarded because they met no `WHEN`-clause tests
- Number of rows whose relevant fields were all null

The Summary Statistics section displays the following data:

- Amount of space used:
  - For bind array (what was actually used based on `BINDSIZE` specified)
  - For other overhead (always required, independent of `BINDSIZE`)
- Cumulative load statistics. That is, for all datafiles, the number of records that were skipped, read, or rejected

The following statistics are logged when a table is loaded:

- Direct-path load of a partitioned table reports per-partition statistics.
- Conventional-path load cannot report per-partition statistics.

If media recovery is not enabled, the load is not logged. That is, if media recovery is disabled the request for a logged operation is overridden.



## SQL\*Loader Guidelines

- **Use a parameter file to specify commonly used command line options**
- **Place data within the control file only for a small, one-time load**
- **Improve performance by:**
  - **Allocating sufficient space**
  - **Sorting the data on the largest index**
  - **Specifying different files for temporary segments for parallel loads**

ORACLE

19-25

Copyright © Oracle Corporation, 2001. All rights reserved.

### SQL\*Loader Usage Guidelines

Use the following guidelines when using SQL\*Loader to minimize errors and improve performance:

- Use a parameter file to specify commonly used command line options. For example, if loading into a data warehouse every week, all options except the names of the files may be the same.
- Separating the control file and the data file permits reusing control files for several load sessions.
- Pre-allocating space based on the expected data volume prevents dynamic allocation of extents during the load and improves the speed of the load.
- When direct loads are used, temporary segments are used to generate indexes for the new data. These indexes are merged with the existing indexes at the end of the load. By sorting the input data on the keys in the largest index, use of sort space can be minimized.
- With parallel direct loads, you can specify the location of temporary segments used for inserting data. For each load session, specify a different database file to achieve maximum performance.

# Summary

In this lesson, you should have learned how to:

- Describe the usage of SQL\*Loader
- Perform basic SQL\*Loader operations
- Demonstrate proficiency using direct-load `INSERT` operations
- List guidelines for using SQL\*Loader and direct-load `INSERT`

ORACLE

## Practice 19 Overview

**This practice covers the following topics:**

- **Using SQL\*Loader to restore data**
  - Using a control file
  - Using a data file
- **Using direct-load insert to load data**

ORACLE

## Practice 19

Use the account HR for all the questions in this practice. Examine the files `case1.ctl`, `case2.ctl` and `case2.dat` to become familiar with the control and data file formats. As user HR, perform the following steps and get acquainted with using SQL\*Loader.

1. Create table DEPARTMENTS2 as a copy of the DEPARTMENTS table.
2. Delete all the records in the DEPARTMENTS2 table.
3. Run SQL\*Loader to load data into the DEPARTMENTS2 table using the control file `case1.ctl` located in your LABS directory. Examine the log file generated, and query the DEPARTMENTS2 table to check that all the data loaded properly.
4. Delete all the records in the DEPARTMENTS2 table.
5. Run SQL\*Loader in direct-path mode to load data into the DEPARTMENTS2 table using the control file `case2.ctl`. Notice that this run uses an input data file to load data. Examine the log file generated, and query the DEPARTMENTS2 table to check the data loaded.
6. Create a table EMPLOYEES2 as a copy of the EMPLOYEES table. When complete, truncate EMPLOYEES2, then restore the data with a direct-load insert from the EMPLOYEES table.
7. Truncate EMPLOYEES2 once again, then restore the data with a parallel direct-load insert from the EMPLOYEES table. Specify a degree of parallelism of two. Verify the data when finished.

# 20

## Workshop

ORACLE

Copyright © Oracle Corporation, 2001. All rights reserved.

# Objectives

**After completing this lesson, you should be able to do the following:**

- **Document a database configuration by using the Database Configuration Worksheet**
- **Configure an Oracle9i database to support stated business requirements**
- **Recover a failed database while minimizing down time and data loss**

ORACLE

# Objectives

- **Enable and use trace output for troubleshooting**
- **Identify and troubleshoot:**
  - **Listener problems**
  - **Client configuration issues**

ORACLE

# Workshop Methodology

- **Group-oriented and interactive**
- **Intensive hands-on diagnosis and problem resolution**
- **Variety of failure scenarios**
- **Recovery solutions**
- **Variety of configuration errors**
- **Develop troubleshooting skills**

ORACLE

20-4

Copyright © Oracle Corporation, 2001. All rights reserved.

## **Group-Oriented and Interactive Structure**

The workshop is structured to allow individuals to work in groups to perform database backup, restore, and recovery operations. Each group is encouraged to share their approach to resolving database failures with other groups in the class.

## **Intensive Hands-On Diagnosis and Problem Resolution**

The intent is to provide you with as much hands-on experience as possible to diagnose and work through backup and recovery scenarios. Experience and knowledge gained from the course will play a major role toward successfully completing the objectives of each session.



## **Variety of Failure Scenarios**

During this workshop, the instructor will induce configuration errors by running a series of shell scripts. The objective is to diagnose the nature of the failure and to make the necessary corrections.

During this workshop, the instructor will also induce failure scenarios without informing the class. The objective is to diagnose the nature of the failure and to perform the appropriate recovery process. The types of failures that you may encounter include:

- Loss of a redo log group
- Loss of datafiles
- Loss of control files
- Media loss
- Loss of a table
- Loss of an online undo segment datafile

## **Recovery Solutions**

This workshop simulates a real-world environment in that exact solutions to problems may not be readily available in the event of a database failure. Therefore, only cursory instructions are noted in Appendix B for performing restore and recovery operations. In some cases you may use associated Oracle Worldwide Support bulletins included in Appendix C. The intent is to familiarize you with available documentation and how to interpret it to perform a successful database recovery.

# Workshop Approach

- **Physical investigation:**
  - Use views and tools to derive information
  - View trace files and log files
  - View command output and log files
  - Use views and tools to confirm proper database configuration
- **Database configuration:**
  - Archiving is enabled
  - Control files and log files are mirrored
  - Control file is backed up

ORACLE

20-6

Copyright © Oracle Corporation, 2001. All rights reserved.

## Physical Investigation

Use the tools in the Oracle9i environment, such as the V\$ views, data dictionary views, and facilities in SQL\*Plus, and Oracle Enterprise Manager (if available) to derive information about your database environment. Keep the business requirements in mind and note any deficiencies that you feel will need to be corrected or implemented to support them.

You will use the command (`tnsping`, `sqlplus`, `lsnrctl`, etc) output and log files to perform the necessary troubleshooting. For the second part, you can use the tools in the Oracle9i environment, such as the V\$ views and facilities in SQL\*Plus, to set up and test your configuration.

## Database Configuration

Physically modify the database configuration to ensure that:

- Archiving is enabled
- Control files and log files are mirrored and distributed across multiple devices
- The control file is backed up both to trace and in a binary format

# Business Requirements

- **Twenty-four hour availability**
- **Peak usage varies across all time zones**
- **Daily backups are required**
- **Complete database recovery is required**

ORACLE

20-7

Copyright © Oracle Corporation, 2001. All rights reserved.

## Business Requirements

The following business requirements should be familiar to you when configuring your database for backup and recovery.

**Twenty-Four Hour Availability** The database must be available 24 hours a day, 7 days a week. An eight-hour maintenance window is scheduled for the first Saturday of each month when the instance can be shut down.

**Peak Usage Varies Across Available Time Frame** This database is accessed globally, so it is used throughout the 24-hour period of one day.

**Daily Backups** Full database backups are required on a daily basis.

**Complete Database Recovery** This is a critical business application and data loss cannot be tolerated. A high number of transactions occur over the 24-hour time frame.

## Resolving a Database Failure

- **Phase I: Diagnose the problem**
- **Phase II: Restore appropriate files**
- **Phase III: Recover the database**
- **Phase IV: Back up the database**

ORACLE

20-8

Copyright © Oracle Corporation, 2001. All rights reserved.

### Resolving a Workshop Failure Scenario

The failure and recovery scenarios are hands-on exercises, where each group has the flexibility to perform the restore and recovery operation that they determine is appropriate. Approximately six failure and recovery scenarios will be conducted during the workshop.

The instructor will not tell you what scenario is being conducted. When resolving failures, follow the diagnostic steps presented in the lesson.

## **Phase I: Diagnose the Problem**

1. The first phase is to research the nature of the failure. Use V\$ views, data dictionary views, trace and log files, basic operating system commands, and Oracle Enterprise Manager to diagnose the problem.
2. Determine if the database instance is available and the database is open.
3. Attempt to start the instance.
4. Shut down the instance if a problem occurs while starting it or when opening the database.
5. Check the trace files and alert log file.
6. Determine the appropriate recovery method:
  - Closed database recovery
  - Open database, offline tablespace recovery
  - Open database, offline tablespace, individual data file recovery
  - Cancel-based recovery
  - Time-based recovery
  - Change-based recovery
7. Once you have determined what type of failure you are dealing with, refer to Appendix B for additional instructions and information.

## **Phase II: Restore Appropriate Files**

Before you perform a recovery scenario, determine what files to restore and what state the instance and database must be in to perform the recovery. Remember that the objective is to minimize down time and loss of data, so do not restore files if it is not necessary.

## **Phase III: Recover the Database**

Once the appropriate files are restored, initiate your recovery operation. After completing the recovery, note any proactive measures that can be taken to prevent that type of failure in the future.

## **Phase IV: Back Up the Database**

Not all recovery operations require a database backup when they are complete. However, determine whether your database needs to be backed up and, if so, perform another backup.

## Troubleshooting Methods

- Use OS utilities like ping and telnet to test network connectivity
- Use Oracle Net utilities like `tnsping` to test service connectivity
- Check logfiles initially to diagnose problems
- Use trace files to get detailed information
- Use trace files sparingly or trim them regularly because of their potential for rapid growth

ORACLE

# Enable Tracing

- Oracle Net Manager
- Edit SQLNET.ORA:

```
TRACE_DIRECTORY_CLIENT = /u01/user01/NETWORK/LOG
NAMES.DEFAULT_DOMAIN = us.oracle.com
TRACE_UNIQUE_CLIENT = on
TRACE_FILE_CLIENT = client.trc
TRACE_LEVEL_CLIENT = SUPPORT
NAMES.DIRECTORY_PATH= (TNSNAMES)
```

ORACLE

20-11

Copyright © Oracle Corporation, 2001. All rights reserved.

## Tracing

Client tracing is controlled by the following `sqlnet.ora` parameters:

<code>TRACE_DIRECTORY_CLIENT</code>	(sets directory where trace files will be written)
<code>TRACE_FILE_CLIENT</code>	(sets trace file name)
<code>TRACE_UNIQUE_CLIENT</code>	(forces creation of a new trace file for each connection)
<code>TRACE_LEVEL_CLIENT</code>	(controls trace output volume, values can be: OFF, USER, ADMIN and SUPPORT, ordered from least to most)

To configure tracing on the client using Oracle Net Manager:

Click on the Profile icon in the directory tree. Select General from the pull down menu and Click on the Tracing tab. Select a trace level from the menu located in the Client Information section. Enter a valid directory name in the Trace Directory field, then enter a file name in the Trace File field. Click on the Unique File Trace Name checkbox if you want a unique identifier appended to each new trace file created and then save your configuration.

# Using Trace Files

**Trace files will give you a better understanding of:**

- **The flow of packets between network nodes**
- **Which component of Net8 is failing**
- **Pertinent error codes**

ORACLE

20-12

Copyright © Oracle Corporation, 2001. All rights reserved.

## Using Trace Files

Evaluating trace files will help you to diagnose and troubleshoot network problems by giving you a better understanding of packet flow, component failure and error codes. The ability to inspect packet data and determine packet types can be an invaluable troubleshooting tool.

The packet type codes are listed below:

- 1 - Connect packet
- 2 - Accept packet
- 3 - Acknowledge packet
- 4 - Refuse packet
- 5 - Redirect packet
- 6 - Data Packet
- 7 - Null Packet, empty data
- 9 - Abort packet
- 11 - Re-send packet
- 12 - Marker packet
- 13 - Attention packet
- 14 - Control information packet



## Using Trace Files (continued)

Below is a sample excerpt from an actual trace file:

```
1. nspsend: plen=287, type=1
2. ntpwr: entry
3. ntpwr: exit
4. nspsend: 287 bytes to transport
5. nspsend: packet dump
6. nspsend: 01 1F 00 00 01 00 00 00 | .....|
7. ...
8. nspsend: 00 00 28 44 45 53 43 52 | ..(DESCR|
9. nspsend: 49 50 54 49 4F 4E 3D 28 | IPTION=(|
10. nspsend: 41 44 44 52 45 53 53 3D | ADDRESS=|
11. nspsend: 28 50 52 4F 54 4F 43 4F | (PROTOCO|
12. nspsend: 4C 3D 62 65 71 29 28 50 | L=beq)(P|
13. nspsend: 52 4F 47 52 41 4D 3D 2F | ROGRAM=|
14. nspsend: 75 30 33 2F 6F 72 61 39 | u03/ora9|
15. nspsend: 69 2F 62 69 6E 2F 6F 72 | i/bin/or|
16. nspsend: 61 63 6C 65 29 28 41 52 | acle)(AR|
17. nspsend: 47 56 30 3D 6F 72 61 63 | GV0=orac|
18. nspsend: 6C 65 55 30 31 29 28 41 | leU01)(A|
19. nspsend: 52 47 53 3D 27 28 44 45 | RGS='(DE|
20. nspsend: 53 43 52 49 50 54 49 4F | SCRIPTIO|
21. nspsend: 4E 3D 28 4C 4F 43 41 4C | N=(LOCAL|
22. nspsend: 3D 59 45 53 29 28 41 44 | =YES)(AD|
23. nspsend: 44 52 45 53 53 3D 28 50 | DRESS=(P|
24. nspsend: 50 52 4F 47 52 41 4D 3D | PROGRAM=|
25. nspsend: 29 28 48 4F 53 54 3D 73 | )(HOST=s|
26. nspsend: 74 63 2D 73 75 6E 30 32 | tc-sun02|
27. nspsend: 2E 75 73 2E 6F 72 61 63 | .us.orac|
28. nspsend: 6C 65 2E 63 6F 6D 29 28 | le.com)(|
29. nspsend: 55 53 45 52 3D 75 73 65 | USER=use|
30. nspsend: 72 30 31 29 29 29 29 00 | r01))))|.|
31. nspsend: normal exit
```

This is the first packet sent by a client to the listener to establish a connection. From line 1 we can see that this is a connect packet (TYPE 1). From the rest of the output we can see the packet length and initialization and connect data sent to the listener. Although too large to display in full, you can locate connection negotiation data and important error codes. This kind of information can be very valuable troubleshooting difficult network problems.

## Resolving a Network Failure

- **Phase I: Diagnose the problem**
- **Phase II: Correct configuration information in the appropriate files**
- **Phase III: Use Oracle commands (`tnsping`, `sqlplus`, `lsnrctl`, etc.) to test the corrections**

ORACLE

## Network Workshop

This workshop introduces eight failure scenarios. Each script introduces a single error in one of the configuration files for Oracle network products. After the instructor introduces an error in your configuration, try to make a connection using the network, then troubleshoot and fix the problem before going on to the next script. For example, to start the first scenario, you would run the command below:

```
$ sqlplus system/manager@your_service_name
```

(Should produce some error)

Use the command output and log files to solve the error. Use an editor to fix the file containing the error (`tnsnames.ora`, `sqlnet.ora` or `listener.ora`) and try to connect again.

If you cannot solve the problem, and wish to continue to the next scenario, three scripts are provided to restore the original files; `FIX_CLIENT.sh` (`tnsnames.ora`), `FIX_NAMING.sh` (`sqlnet.ora`) and `FIX_LISTENER.sh` (`listener.ora`). Run the `FIX` script related to your problem only if you can't fix it yourself, then tell your instructor that you wish to go on to the next scenario.

# Summary

- **Instructor-facilitated workshop**
- **Group-oriented**
- **Hands-on approach**
- **Simulated “real-world” environment**
- **Minimize down time and data loss**
- **Use tools and diagnostics to determine the type of failure**

ORACLE

20-16

Copyright © Oracle Corporation, 2001. All rights reserved.

## **Instructor-Facilitated Workshop**

The instructor will facilitate the workshop by providing guidance and additional information as appropriate.

## **Group-Oriented Emphasis**

A strong emphasis is placed on teaming with other students in the workshop for purposes of diagnosing and resolving failures. The ability to successfully complete each scenario is based upon the cumulative knowledge and problem resolution skills of each group.

## **Hands-On Approach**

This is meant to be a hands-on workshop, providing you with the maximum allowable time to be involved in a lab situation.

## **Simulated Real-World Environment**

Real-world problems are not as easy to resolve as hypothetical lab problems. Therefore, this workshop provides numerous database failures that will be unknown to you and must be diagnosed and resolved by each group using available resources and tools. The primary objective of each scenario is to perform a timely recovery with no or minimal down time and no data loss.

## **Minimized Down Time and Data Loss**

For many of the failure scenarios there may be more than one method to restore and recover the database. Keep in mind that when resolving the failure, you want to minimize down time and data loss, so select the best restore and recovery method.

## **Use of Tools and Diagnostics**

Use the various tools and diagnostics that were addressed during the course to help determine and resolve the type of failure.

## Practice 20

1. Shutdown the instance and restore your database files from the backup in the `$HOME/DONTOUCH` directory.
2. Start your instance, mount and open the database.
3. Connect as sysdba and use the data dictionary views and SQL\*Plus commands to complete the Database Configuration Checklist which follows.
4. Verify that your control files are mirrored..
5. Verify that your online redo log files are multiplexed.
6. Verify that your database is in Archivelog mode.
7. Run the `$HOME/STUDENT/LABS/emphist.sql` and `$HOME/STUDENT/LABS/moreemphist.sql` scripts.
8. Use operating system commands or RMAN to make a whole database backup in the `$HOME/BACKUP` directory.
9. Remove any archived redo log files that are no longer needed.
10. Backup up your control file to trace.
11. Ensure that your instance is started and your database is open before beginning the workshop scenarios.

## Practice 20 – Database Configuration Checklist

### Tablespace and Datafile Information

Tablespace Name	Datafile Name (full path)

### Online Redo Log File Information

Group #	Redo Log File Name (full path)	Size	Status

### Control File Information

Control File Name

## Practice 20 – Database Configuration Checklist

### Initialization Parameters

Parameter Name	Value
BACKGROUND_DUMP_DEST	
CORE_DUMP_DEST	
DB_FILES	
DB_NAME	
LOG_ARCHIVE_DEST	
LOG_ARCHIVE_DUPLEX_DEST	
LOG_ARCHIVE_FORMAT	
LOG_ARCHIVE_MIN_SUCCEED_DEST	
LOG_ARCHIVE_START	
USER_DUMP_DEST	



---

**A**

---

# **Practice Solutions**

## Practice 3 Solutions

1. Create a listener `listener $nn$`  ( $nn$  will be a two digit number assigned to you by your instructor) using Oracle Net Manager. The listener must be configured for the server as provided by the instructor; this server contains an Oracle database `Unn`. The listener must be configured for the TCP/IP protocol only and must listen for incoming connections on the port provided by the instructor.

**Note:** If Oracle9i Oracle Net software is loaded on the student PC's, the listener configuration file will be created on the client PC using Oracle Net Manager and, in later steps, transferred via FTP or similar file transfer application on the server.

For this practice and successive network practices, the `TNS_ADMIN` environment variable **must** point to `$HOME/NETWORK/ADMIN` on the host where your Unix account resides. Look in your `.profile` (located in your home directory) and search for an entry like this:

```
TNS_ADMIN=$HOME/NETWORK/ADMIN
export TNS_ADMIN
```

Edit the file and add the lines above if they don't already exist. Log out and log back in again for the changes to take effect.

If Oracle9i client software is not available on your workstation, the `listener.ora` must be edited by hand. Sample networking files can be found in your `$HOME/network/admin` directory. The sample files will all have `.sam` extensions. Copy `listener.sam` to `listener.ora` and edit by hand using `vi`.

```
$ cd $TNS_ADMIN
$ cp listener.sam listener.ora
```

- a. If Oracle9i is available on the PC, go to the NT Start menu on the client PC, select Programs—>Oracle - *Oracle Home*—>Network Administration—>Oracle Net Manager.
- b. Click on the “Local” icon in the Net Configuration tree then click the Listeners folder. Select “Create” from the Edit menu item, or click the “+” icon.
- c. Enter a name for your listener (`listener $nn$` ) in the Choose Listener Name dialog box that appears and click OK. The name of the new listener will appear below the Listeners folder in the left-hand pane of Net Manager.
- d. Click the new listener name and select Listening Locations from the pull-down menu on the right-hand pane in Net Manager if not already selected.
- e. Click the Add Address button. A tab for the address details of the listener will appear.
- f. Select TCP as the protocol, if not already selected.
- g. Enter the name of the server in the Host field, and the port number assigned for your listener in the Port field (the port number is provided by your instructor). Leave TCP as the default value for the Protocol.
- h. Select Database Services from the pull-down menu on the right side of the screen in Net Manager.
- i. Click the Add Database button. A tab for the database on behalf of which the listener will listen for incoming connections will appear.

### Practice 3 Solutions (continued)

- j. Enter a name for the global database in the Global Database Name field (the Global Database Name is provided by your instructor).
- k. Enter the directory, defined as your `$ORACLE_HOME` on the server, in the Oracle Home Directory field (Issue the `env` command from the UNIX prompt to get the home directory).
- l. Enter your database system identifier (*Unn*) in the SID field.
- m. From the pull-down menu on the right side of the screen in Net Manager, select “General Parameters” and then choose the “Logging and Tracing” tab. Make sure logging is enabled and the log file to be used is `$HOME/NETWORK/LOG/listenernn.log`.
- n. Save your configuration by selecting “Save Network Configuration” from the File menu item in Net Manager.

### Manual Configuration

If you are manually editing the `listener.ora` file, follow the steps below.

- a. Change directories to `$TNS_ADMIN` and make sure to copy the `listener.sam` file to `listener.ora`

```
$ cd $TNS_ADMIN
```

```
$ cp listener.sam listener.ora
```

- b. Using `vi`, edit the sample `listener.ora` file and define the following entries:

<code>SID_LIST_LISTENERnn</code>	(where <code>LISTENERnn</code> is your listener name)
<code>GLOBAL_DBNAME</code>	(instance_name + domain name)
<code>ORACLE_HOME</code>	(should be same as your <code>\$ORACLE_HOME</code> )
<code>SID_NAME</code>	(your instance name)
<code>LISTENERnn</code>	(your listener name)
<code>PROTOCOL</code>	(should be <code>tcp</code> )
<code>HOST</code>	(your host name)
<code>PORT</code>	(the port assigned by your instructor)
<code>LOG_DIRECTORY_LISTENERnn</code>	( <code>YOUR_HOME_DIR/NETWORK/LOG</code> )
<code>LOG_FILE_ LISTENERnn</code>	( <code>YOUR_LISTENER_NAME.log</code> )

## Practice 3 Solutions (continued)

For example:

```
SID_LIST_LISTENER01 =
(SID_LIST =
  (SID_DESC =
    (GLOBAL_DBNAME = U01.us.oracle.com)
    (ORACLE_HOME = /u03/ora9i)
    (SID_NAME = U01)
  )
)
LISTENER01 =
DESCRIPTION =
  (ADDRESS = (PROTOCOL = TCP)(HOST = stc-sun02)(PORT = 1701))
)
LOG_DIRECTORY_LISTENER01 = /dbclass7/user01/NETWORK/LOG
LOG_FILE_LISTENER01 = listener01.log
```

2. View the contents of the listener.ora file to verify the configuration details.
  - a. `$ cd $TNS_ADMIN`
  - b. `$ view listener.ora` (you can use `more` or `pg` also)
3. If you have created the listener.ora file on your pc, then use FTP (ASCII mode) to transfer it to your \$TNS\_ADMIN directory on the Unix server.
4. When the listener.ora file is properly placed, start your listener by issuing `lsnrctl start listenernn` from your prompt. If you encounter difficulties, use the `lsnrctl` command output and the listener log file to troubleshoot.

```
$ lsnrctl start listener01 (substitute your listener name here)
```

5. Start up or stop and restart your database instance.

```
$ sqlplus /nolog
SQL> connect / as sysdba
SQL> shutdown
SQL> startup
```

6. View the contents of the listener log file. Is the instance registered ? Why not ?

```
$ more $HOME/NETWORK/LOG/listener01.log
TIMESTAMP * CONNECT DATA [* PROTOCOL INFO] * EVENT [* SID] *
RETURN CODE
23-APR-2001 21:28:15 *
(CONNECT_DATA=(CID=(PROGRAM=)(HOST=stc-sun02) ...
```

For instance registration, the init.ora parameters `SERVICE_NAMES` and `INSTANCE_NAME` must be defined properly. If your listener is non-default, then `LOCAL_LISTENER` must also be defined. This has not been done yet.

## Practice 4 Solutions

1. Use the Oracle Net Manager and configure your client to use the local naming method. Select “TNSNAMES” as the *only* naming method. If you are unsure of the name of your client, please ask the instructor how to obtain the name.
  - a. If Oracle9i is available on the PC, go to the NT Start menu on the client PC, select Programs—>Oracle - *Oracle Home*—>Network Administration—>Oracle Net Manager.
  - b. Select the “Profile” icon in the Net Configuration tree and then choose “NAMING” from the pull down menu.
  - c. Select the “Methods” tab and make sure that “TNSNAMES” is the only method promoted.
  - d. From the file menu, choose “Save Network Configuration”.
  - e. Select “Service Naming” from the directory tree and click on “+” to add a service name.
  - f. Enter a net service name and click “Next” to continue.
  - g. Select TCP/IP as the protocol, and click “Next” to continue.
  - h. Enter your Unix hostname and the port given to you by your instructor. Click “Next” to continue.
  - i. Enter the service name (Oracle8i or better) and click “Next” to continue.
  - j. Test the service. If test fails, check login being used by clicking the “Change Login” button. Make sure account being used exists and try again.
  - k. From the file menu, choose “Save Network Configuration”.

## Manual Configuration

If Oracle9i is not available on your client, manually configure and test the client connection from the Unix host. Change directories to \$TNS\_ADMIN and copy `sqlnet.sam` and `tnsnames.sam` to `sqlnet.ora` and `tnsnames.ora` respectively. Edit them manually with `vi`.

- a. Edit `sqlnet.ora` and make sure `NAMES.DIRECTORY_PATH` is set to “TNSNAMES”.

```
$ cd $TNS_ADMIN
$ vi sqlnet.ora

NAMES.DEFAULT_DOMAIN = us.oracle.com
NAMES.DIRECTORY_PATH= (TNSNAMES)
```

## Practice 4 Solutions (continued)

- b. Edit tnsnames.ora and add a net service name for your database.

```
$ cd $TNS_ADMIN
$ vi sqlnet.ora
U01 =          {your instance name here}
(DESCRIPTION =
  (ADDRESS_LIST =
    (ADDRESS = (PROTOCOL = TCP)(HOST = stc-sun02)(PORT =
1701))
  ) {protocol must be tcp, your host name and assigned
port}
  (CONNECT_DATA =
    (SERVICE_NAME = U01.us.oracle.com) {SID+DOMAIN or
GLOBAL_DBNAME}
  )
)
```

2. Test that the service is reachable using tnsping.

```
$ tnsping U01
Used parameter files:
/u01/user01/NETWORK/ADMIN/sqlnet.ora
/u01/user01/NETWORK/ADMIN/tnsnames.ora
Used TNSNAMES adapter to resolve the alias
Attempting to contact(ADDRESS=(PROTOCOL=TCP)(HOST=stc-
sun02)(PORT=1501))
OK (10 msec)
```

3. Connect to the server as system/manager using SQL\*Plus and verify that you are connected to the correct instance by querying the V\$INSTANCE view.

```
$ sqlplus system/manager@u01
SQL*Plus:Release 9.0.0.0.0 - Beta on Tue Mar 27 22:32:46
(c) Copyright 2001 Oracle Corporation. All rights reserved.
Connected to:
Oracle9i Enterprise Edition Release 9.0.0.0.0 - Beta
With the Partitioning option
JServer Release 9.0.0.0.0 - Beta
SQL> select instance_name from v$instance;
INSTANCE_NAME
-----
U01
```

## Practice 5 Solutions

1. Start a Telnet session connecting to the server where your database resides. Configure and start up Oracle Shared Server for your database so that you have one dispatcher listening for TCP/IP connections and one shared server to serve requests. Specify the maximum dispatchers as two and maximum shared servers as six.

**Note:** Since the listener you are using is not listening on the default port of 1521, you must define the `local_listener` parameter in your `init.ora` and include a listener alias and address in your `tnsnames.ora` file. If this parameter is not properly defined, the instance will not start since the dispatcher processes will not know how to register with the listener.

- a. Shut down your instance and add the following parameters to your `init.ora` file located in `$HOME/ADMIN/PFILE`.

```
local_listener=your_listener_name
dispatchers = "(PROTOCOL=TCP)(DISPATCHERS=1)"
shared_servers=1
max_dispatchers=2
max_shared_servers=6
```

- b. Add the following lines to your `tnsnames.ora` file:

```
your_listener_name.us.oracle.com =
(description =
(address=(PROTOCOL=tcp)(HOST=your_host.us.oracle.com)
(RT = 1701)))
```

- c. Restart your instance

2. To verify that a dispatcher is associated with your listener, use the `LSNRCTL` utility. Issue the command: `lsnrctl services your_listener_name`

```
$ lsnrctl services listener01
LSNRCTL for Solaris: Version 9.0.0.0.0
Copyright (c) 1991, 2001, Oracle Corporation.
Connecting to
(DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)(HOST=stc-sun02)
Services Summary...
Service "u01.us.oracle.com" has 1 instance(s)
...
"D000" established:0 refused:0 current:0 max:992
state:ready
DISPATCHER <machine: stc-sun02.us.oracle.com, pid: 14277>
(ADDRESS=(PROTOCOL=tcp)(HOST=stc-sun02)(PORT=35204))
The command completed successfully
```

## Practice 5 Solutions (continued)

- Before making a network connection, query the view V\$CIRCUIT from SQL\*Plus connecting as system/manager in your telnet session to see if it contains data. This view has an entry for each connection session currently using shared servers.

```
$ sqlplus system/manager
SQL> select circuit, dispatcher, server from v$circuit;
no rows selected
```

- Make a connection using SQL\*Plus, connecting as system/manager from your client to the server, and query V\$CIRCUIT view again. After you have verified the connection, exit SQL\*Plus.

```
$ sqlplus system/manager@u01
SQL> select circuit, dispatcher, server from v$circuit;
CIRCUIT  DISPATCH  SERVER
-----  -
80DE1B50 8108FDA8 8108F728
```

- Query the V\$SHARED\_SERVER view to see how many shared servers have been started.

```
SQL> select name, status, circuit from v$shared_server;
NAME STATUS          CIRCUIT
-----  -
S000 EXEC            80DE1B50
```

- Query the V\$DISPATCHER view to see how many dispatchers have been started.

```
SQL> select name, status from v$dispatcher;
NAME STATUS
-----  -
D000 WAIT
```

- Make two connections using SQL\*Plus, connecting as system/manager from your client to the server using shared servers. Has the number of shared servers increased? Why or why not?

```
$ sqlplus system/manager@u01
SQL>! (This allows the user to "shell out" of SQL*Plus and start another session)
$ sqlplus system/manager@u01
SQL> select name, status, circuit from v$shared_server;
NAME STATUS          CIRCUIT
-----  -
S000 EXEC            80DE1B50
```

Another shared server has not been started. The shared server is not considered to be overloaded and is well below the operating system determined maximum number of connections.



## Practice 5 Solutions (continued)

8. Add one more dispatcher to handle TCP requests and verify that the additional dispatcher has been added.

- a. Shutdown your instance and change the “dispatchers” values from 1 to 2 in your init.ora file:

```
$ vi $HOME/ADMIN/PFILE/initUnn.ora
dispatchers = "(PROTOCOL=TCP)(DISPATCHERS=2)
```

- b. Restart the instance and use the lsnrctl command to check services:

```
$ lsnrctl services listener01 (use your listener name
here)
```

```
LSNRCTL for Solaris: Version 9.0.0.0.0
```

```
Copyright (c) 1991, 2001, Oracle Corporation.
```

```
Connecting to
```

```
(DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)(HOST=stc-sun02)
```

```
Services Summary...
```

```
Service "U01.us.oracle.com" has 1 instance(s).
```

```
...
```

```
"D000" established:0 refused:0 current:0 max:992
state:ready
```

```
DISPATCHER <machine: stc-sun02.us.oracle.com, pid:
14277>
```

```
(ADDRESS=(PROTOCOL=tcp)(HOST=stc-sun02)(PORT=35204))
```

```
"D001" established:0 refused:0 current:0 max:992
state:ready
```

```
DISPATCHER <machine: stc-sun02, pid: 14596>
```

```
(ADDRESS=(PROTOCOL=tcp)(HOST=stc-sun02)(PORT=35310))
```

```
The command completed successfully
```

## Practice 7 Solutions

1. Query the V\$ view that you use to find the names of all datafiles in the database.

```
SQL> SELECT name FROM v$datafile;
```

```
NAME
```

```
-----  
/databases/db01/ORADATA/u01/system01.dbf  
/databases/db01/ORADATA/u02/undotbs.dbf  
/databases/db01/ORADATA/u03/users01.dbf  
/databases/db01/ORADATA/u03/indx01.dbf  
/databases/db01/ORADATA/u02/sample01.dbf  
/databases/db01/ORADATA/u01/querydata01.dbf
```

2. Query the V\$ views that you use to find the current online redo log group and names of all redo log files in the database.

```
SQL> SELECT group#, status FROM v$log;
```

```
GROUP# STATUS
```

```
-----  
1          CURRENT  
2          INACTIVE
```

```
SQL> SELECT member FROM v$logfile;
```

```
MEMBER
```

```
-----  
/databases/db01/ORADATA/u03/log01a.rdo  
/databases/db01/ORADATA/u03/log02a.rdo
```

3. Query the V\$ view that you use to find the names of all control files in the database.

```
SQL> SELECT name FROM v$controlfile;
```

```
NAME
```

```
-----  
/databases/db01/ORADATA/u01/ctrl01.ctl
```

4. Query the V\$ view that you use to find the name of the database before dropping tables or shutting down the database.

```
SQL> SELECT name FROM v$database;
```

```
NAME
```

```
-----  
DB01
```

## Practice 7 Solutions (continued)

5. Query the V\$ view that you use to locate processes still connected to the instance before shutting down the database.

```
SQL> SELECT pid, username FROM v$process;
```

```
PID USERNAME
```

```
--- -
```

```
1
2    db01
3    db01
4    db01
5    db01
6    db01
7    db01
8    db01
```

```
8 rows selected.
```

6. Which initialization parameter configures the memory area in the SGA that buffers recovery information in memory before being written to disk?

```
LOG_BUFFER
```

7. What is the large pool, when is it used, and what initialization parameter configures it?

The large pool is an area of the SGA which can be used for buffering information in memory for Recovery Manager when IO slaves are required. This increases the speed and efficiency of backups and restores when using RMAN.

The `LARGE_POOL_SIZE` parameter specifies the number of bytes allocated from the SGA.

8. Describe the significance of the `FAST_START_MTTR_TARGET` parameter during instance recovery.

A target (bounded) time to complete the roll forward phase of recovery is specified by means of the parameter `FAST_START_MTTR_TARGET`, and Oracle automatically varies the checkpoint writes to meet that target.

## Practice 7 Solutions (continued)

9. Set up mirroring of control files so you have two control files. Place your second control file in the \$HOME/ORADATA/u02 directory.

To add a new control file or change the number or location of the control file, follow these steps:

- a. Shut down the database:

```
SQL> SHUTDOWN IMMEDIATE;
```

- b. Copy the existing control file to a different device using operating system commands:

```
$cp -p $HOME/ORADATA/u01/ctrl01.ctl
```

```
$HOME/ORADATA/u02/ctrl02.ctl
```

```
$chmod g+wx $HOME/ORADATA/u02/ctrl02.ctl
```

- c. Edit or add the CONTROL\_FILES parameter and specify names for all the control files:

```
$vi $HOME/ADMIN/PFILE/init<sid>.ora
```

```
control_files=$HOME/ORADATA/u01/ctrl01.ctl,
```

```
$HOME/ORADATA/u02/ctrl02.ctl
```

- d. Start up the instance and open the database:

```
SQL> STARTUP PFILE=$HOME/ADMIN/PFILE/init<sid>.ora
```

```
SQL> show parameter control_files
```

NAME	TYPE	VALUE
control_files	string	\$HOME/ORADATA/u01/ctrl01.ctl, \$HOME/ORADATA/u02/ctrl02.ctl

10. Set up mirroring of online redo log files so you have two members per group. Place the second member of each group in the \$HOME/ORADATA/u04 directory.

```
SQL> alter database add logfile member
```

```
2    '$HOME/ORADATA/u04/log01b.rdo' to group 1,
```

```
3    '$HOME/ORADATA/u04/log02b.rdo' to group 2;
```

Database altered.

```
SQL> select member from v$logfile;
```

MEMBER

```
-----  
/databases/ed21/ORADATA/u03/log01a.rdo
```

```
/databases/ed21/ORADATA/u03/log02a.rdo
```

```
/databases/ed21/ORADATA/u04/log01b.rdo
```

```
/databases/ed21/ORADATA/u04/log02b.rdo
```

## Practice 8 Solutions

1. Invoke SQL\*Plus, connect as sysdba , and shut down the instance with the Immediate option.

```
$ sqlplus /nolog
SQL> connect / as sysdba
SQL> shutdown immediate
```

2. Edit the init.ora file to:

- Enable archiving
  - Archive log files to two destinations: \$HOME/ORADATA/ARCHIVE1 and \$HOME/ORADATA/ARCHIVE2 directories. The \$HOME/ORADATA/ARCHIVE1 is mandatory, and \$HOME/ORADATA/ARCHIVE2 is optional.
  - Use the archiving format of arch\_%s.arc
  - Spawn two archive processes at instance start
- ```
init.ora file
...
log_archive_start=true
log_archive_dest_1="LOCATION=$HOME/ORADATA/ARCHIVE1/
MANDATORY"
log_archive_dest_2="LOCATION=$HOME/ORADATA/ARCHIVE2/
OPTIONAL"
log_archive_format = arch_%s.arc
log_archive_max_processes = 2
```

3. Start up the database in Mount state.

```
SQL> startup mount pfile=$HOME/ADMIN/PFILE/init<sid>.ora
ORACLE instance started.
```

## Practice 8 Solutions (continued)

4. List the parameters LOG\_ARCHIVE\_DEST, LOG\_ARCHIVE\_START, and LOG\_ARCHIVE\_FORMAT, and note the values.

```
SQL> show parameter log_archive
```

| NAME                         | TYPE    | VALUE                                           |
|------------------------------|---------|-------------------------------------------------|
| log_archive_dest             | string  |                                                 |
| log_archive_dest_1           | string  | LOCATION=\$HOME/ORADATA/<br>ARCHIVE1/ MANDATORY |
| log_archive_dest_10          | string  |                                                 |
| log_archive_dest_2           |         | LOCATION=\$HOME/ORADATA/<br>ARCHIVE2/ OPTIONAL  |
| ...                          |         |                                                 |
| log_archive_dest_9           | string  |                                                 |
| log_archive_dest_state_1     | string  | enable                                          |
| log_archive_dest_state_10    | string  | enable                                          |
| ...                          |         |                                                 |
| log_archive_dest_state_9     | string  | enable                                          |
| log_archive_duplex_dest      | string  |                                                 |
| log_archive_format           | string  | arch_%s.arc                                     |
| log_archive_max_processes    | integer | 2                                               |
| log_archive_min_succeed_dest | integer | 1                                               |
| log_archive_start            | boolean | TRUE                                            |
| log_archive_trace            | integer | 0                                               |

5. Execute the ARCHIVE LOG LIST command. Note the database log mode of the database and whether automatic archiving is enabled.

```
SQL> archive log list;
```

|                            |                                   |
|----------------------------|-----------------------------------|
| Database log mode          | No Archive Mode                   |
| Automatic archival         | Enabled                           |
| Archive destination        | /databases/db01/ORADATA/ARCHIVE2/ |
| Oldest online log sequence | 62                                |
| Current log sequence       | 63                                |

6. Set the database in Archivelog mode.

```
SQL> alter database archivelog;
```

```
Database altered.
```

7. Open the database.

```
SQL> alter database open;
```

```
Database altered.
```

## Practice 8 Solutions (continued)

8. Execute the ARCHIVE LOG LIST command. Verify that two archive processes are running.

```
SQL> archive log list;
```

| Database log mode            | Archive Mode                     |
|------------------------------|----------------------------------|
| Automatic archival           | Enabled                          |
| Archive destination          | /databases/db01/ORADATA/ARCHIVE2 |
| Oldest online log sequence   | 62                               |
| Next log sequence to archive | 63                               |
| Current log sequence         | 63                               |

Enter the following command at the operation system prompt:

```
$ ps -ef|grep arc
```

```
oracle 29296      1  0 03:19:51 ?        0:00 ora_arc0_db01
oracle 29298      1  0 03:19:51 ?        0:00 ora_arc1_db01
```

9. Execute the ALTER SYSTEM SWITCH LOGFILE command twice, then show the values of the ARCHIVE parameters. Do you see any archived log files? What is the format of the filename?

```
SQL> alter system switch logfile;
```

System altered.

```
SQL> alter system switch logfile;
```

System altered.

```
SQL> select name, value
```

```
2  from v$parameter
```

```
3  where name like 'log_archive_dest%';
```

| NAME | VALUE |
|------|-------|
|------|-------|

|                   |      |
|-------------------|------|
| log_archive_start | TRUE |
|-------------------|------|

log\_archive\_dest

log\_archive\_duplex\_dest

log\_archive\_dest\_1 LOCATION=\$HOME/ORADATA/ARCHIVE1/  
MANDATORY

log\_archive\_dest\_2 LOCATION=\$HOME/ORADATA/ARCHIVE2/ OPTIONAL

...

log\_archive\_dest\_10

log\_archive\_dest\_state\_1 enable

log\_archive\_dest\_state\_2 enable

...

## Practice 8 Solutions (continued)

```
log_archive_dest_state_10    enable
log_archive_max_processes    2
log_archive_min_succeed_dest 1
log_archive_trace            0
log_archive_format            arch_%s.arc

SQL> !ls -l $HOME/ORADATA/ARCHIVE1 $HOME/ORADATA/ARCHIVE2
/databases/db01/ORADATA/ARCHIVE1/:
total 150
-rw-rw----    1 oracle    dba 77824 Mar 23 03:28 arch_63.arc
-rw-rw----    1 oracle    dba 1024  Mar 23 03:28 arch_64.arc
/databases/db01/ORADATA/ARCHIVE2/:
total 150
-rw-rw----    1 oracle    dba 77824 Mar 23 03:28 arch_63.arc
-rw-rw----    1 oracle    dba 1024  Mar 23 03:28 arch_64.arc
```

10. Stop automatic archiving by executing the ALTER SYSTEM ARCHIVE LOG STOP command.

```
SQL> alter system archive log stop;
System altered.
```

11. Execute the ALTER SYSTEM SWITCH LOGFILE command enough times to cycle through all the online redo log groups. What happened and why?

```
SQL> alter system switch logfile;
System altered.
```

```
SQL> alter system switch logfile;
```

The database is in Archivelog mode. Automatic archiving is disabled the next redo log file cannot be used since it has not been archived.

12. Establish a second telnet session and invoke SQL\*Plus. Connect as sysdba.
13. Enable automatic archiving by using the ALTER SYSTEM ARCHIVE LOG START command.

```
SQL> alter system archive log start;
System altered.
```

14. Return to your first session. What happened and why? You now have the “System Altered” message followed by the SQL prompt. The archiver process has been restarted. The log switch is now possible because the redo log files were archived when automatic archiving was restarted.



## Practice 9 Solutions

1. List some of the benefits of using RMAN rather than user-managed backup and recovery procedures.

---

---

---

---

2. Describe some of the ways that RMAN uses the control file of the target database.

---

---

---

---

3. Connect to your database as the target database in the default Nocatalog mode.

```
$rman
```

```
RMAN> connect target
```

```
Recovery Manager: Release 9.0.0.0.0 - Beta
```

```
(c) Copyright 2000 Oracle Corporation. All rights reserved.
```

```
connected to target database: DB01 (DBID=1125003950)
```

4. Use the RMAN REPORT command to generate a listing of your database structure.

```
RMAN> report schema;
```

```
using target database controlfile instead of recovery  
catalog
```

```
Report of database schema
```

```
Fi K-bytes Tablespace Datafile Name
```

```
-- -----
```

|   |        |            |                                              |
|---|--------|------------|----------------------------------------------|
| 1 | 128000 | SYSTEM     | /databases/db01/ORADATA/u01/system.dbf       |
| 2 | 30720  | UNDOTBS    | /databases/db01/ORADATA/<br>u02/undotbs.dbf  |
| 3 | 5120   | USERS      | /databases/db01/ORADATA/<br>u03/users01.dbf  |
| 4 | 5120   | INDX       | /databases/db01/ORADATA/u03/indx01.dbf       |
| 5 | 81920  | SAMPLE     | /databases/db01/ORADATA/<br>u02/sample01.dbf |
| 6 | 1024   | QUERY_DATA | /databases/db01/ORADATA/<br>u01/querydata01f |

## Practice 9 Solutions (continued)

5. Use the RMAN SHOW command to generate a listing of the RMAN configuration settings.

```
RMAN> show all;

RMAN configuration parameters are:
CONFIGURE RETENTION POLICY TO REDUNDANCY 1; # default
CONFIGURE BACKUP OPTIMIZATION OFF; # default
CONFIGURE DEFAULT DEVICE TYPE TO DISK; # default
CONFIGURE CONTROLFILE AUTOBACKUP OFF; # default
CONFIGURE CONTROLFILE AUTOBACKUP FORMAT FOR DEVICE TYPE DISK
TO '%F'; # default
CONFIGURE DEVICE TYPE DISK PARALLELISM 1; # default
CONFIGURE DATAFILE BACKUP COPIES FOR DEVICE TYPE DISK TO 1;
# default
CONFIGURE ARCHIVELOG BACKUP COPIES FOR DEVICE TYPE DISK TO
1; # default
CONFIGURE MAXSETSIZE TO UNLIMITED; # default
CONFIGURE SNAPSHOT CONTROLFILE NAME TO
'/databases/oracle9i/dbs/snapcf_db01.f';t
```

6. Use the RMAN CONFIGURE command to set the backup retention policy to a recovery window of 14 days.

```
RMAN> configure retention policy to recovery
2> window of 14 days;

new RMAN configuration parameters:
CONFIGURE RETENTION POLICY TO RECOVERY WINDOW OF 14 DAYS;
new RMAN configuration parameters are successfully stored
```

7. Verify the setting for the backup retention policy.

```
RMAN> SHOW RETENTION POLICY;

RMAN configuration parameters:
CONFIGURE RETENTION POLICY TO RECOVERY WINDOW OF 14 DAYS;
```

8. Set the backup retention policy back to the default value.

```
RMAN> CONFIGURE RETENTION POLICY CLEAR;

old RMAN configuration parameters:
CONFIGURE RETENTION POLICY TO RECOVERY WINDOW OF 14 DAYS;
RMAN configuration parameters are successfully reset to
default value
```

## Practice 10 Solutions

1. While the database is open, connect to the database as `sys` or `system` and using `V$` and data dictionary views, make a list of all of the files that must be backed up for a whole offline database backup.

**Note:** Copy the redo logs for ease of restore/recovery in `noarchive` mode.

```
SQL> select name from v$controlfile;
```

```
NAME
```

```
-----  
/databases/db01/ORADATA/u01/ctrl01.ctl  
/databases/db01/ORADATA/u02/ctrl02.ctl
```

```
SQL> select member from v$logfile;
```

```
MEMBER
```

```
-----  
/databases/db01/ORADATA/u03/log01a.rdo  
/databases/db01/ORADATA/u03/log02a.rdo  
/databases/db01/ORADATA/u04/log01b.rdo  
/databases/db01/ORADATA/u04/log02b.rdo
```

```
SQL> select name from v$datafile;
```

```
NAME
```

```
-----  
/databases/db01/ORADATA/u01/system01.dbf  
/databases/db01/ORADATA/u02/undotbs.dbf  
/databases/db01/ORADATA/u03/users01.dbf  
/databases/db01/ORADATA/u03/indx01.dbf  
/databases/db01/ORADATA/u02/sample01.dbf  
/databases/db01/ORADATA/u01/querydata01.dbf
```

## Practice 10 Solutions (continued)

2. Shut down the database with the IMMEDIATE option. Make a whole offline database backups into the \$HOME/DONTTOUCH directory using operating system commands.

```
$ cp -rp $HOME/ORADATA/u* $HOME/DONTTOUCH
```

```
$ cp $ORACLE_HOME/dbs/orapw<sid> $HOME/DONTTOUCH
```

**Note:** Do not place in or remove files from the DONTTOUCH directory. This copy will be used during the workshop.

3. Start the instance, mount and open the database.

```
SQL> startup pfile=$HOME/ADMIN/PFILE/initdb01.ora
```

4. Connect as system/manager and make an open backup of the SAMPLE tablespace. Copy the file to \$HOME/BACKUP/UMAN directory. Make sure that you do not overwrite another copy.

```
SQL> connect system/manager
```

```
SQL> ALTER TABLESPACE sample BEGIN BACKUP;
```

```
SQL> !cp $HOME/ORADATA/u02/sample01.dbf  
$HOME/BACKUP/UMAN/sample01.dbf
```

```
SQL> ALTER TABLESPACE sample END BACKUP;
```

5. Use the ALTER DATABASE command to back up the control file to trace. Execute the \$HOME/STUDENT/LABS/spid.sql script to identify the trace file. Exit to the operating system and copy the trace file to \$HOME/BACKUP/UMAN/cntrl.sql. Using an editor, remove the comments from the trace file.

```
SQL> alter database backup controlfile to trace;
```

```
SQL> @$HOME/STUDENT/LABS/spid.sql
```

| USERNAME | SPID         |
|----------|--------------|
| -----    | -----        |
| SYSTEM   | <process ID> |

```
SQL> exit
```

```
$ cd $HOME/ADMIN/UDUMP
```

```
$ cp db01_ora_<process ID>.trc $HOME/BACKUP/UMAN/cntrl.trc
```

```
$ vi $HOME/BACKUP/UMAN/cntrl.trc
```

Remove all comments from the trace file.

6. Create a binary copy of the control file and put it in the \$HOME/BACKUP/UMAN directory. Name the backup copy cntrl1.bkp.

```
SQL> alter database backup controlfile to
```

```
2> '$HOME/BACKUP/UMAN/cntrl1.bkp';
```

## Practice 11 Solutions

1. What are the two supported backup types for Recovery Manager? List some of the differences between the two backup types.

The two types of backups supported by the recovery manager are backup set and image copy.

A backup set is a backup of one or more database files, while the image copy contains a backup of only one datafile.

An image copy can be made to a disk only while backup set can be taken to disk or tape.

An image copy contains all the blocks of the input file (even the unused blocks in datafiles) while backup set may contain only the used blocks.

Image copies operate on single files at file level while backup sets operate on files and their logical groups (such as Tablespace, Database).

2. Use RMAN to back up the datafiles belonging to the SAMPLE and USERS tablespace. Be sure you also make a copy of the current control file. Your backups should be placed in the \$HOME/BACKUP/RMAN directory and should use the format df\_%d\_%s\_%p.bus for the file names.

```
RMAN> BACKUP TABLESPACE sample INCLUDE CURRENT CONTROLFILE  
FORMAT '$HOME/BACKUP/RMAN/df_%d_%s_%p.bus';
```

```
Starting backup at 21-MAR-01 using target database  
controlfile instead of recovery catalog allocated channel:  
ORA_DISK_1
```

```
channel ORA_DISK_1: sid=12 devtype=DISK
```

```
channel ORA_DISK_1: starting full datafile backupset
```

```
channel ORA_DISK_1: specifying datafile(s) in backupset  
input datafile fno=00006
```

```
name=/databases/db01/ORADATA/u02/sample01.dbf
```

```
including current controlfile in backupset
```

```
channel ORA_DISK_1: starting piece 1 at 21-MAR-01
```

```
channel ORA_DISK_1: finished piece 1 at 21-MAR-01
```

```
piece handle=/databases/db01/BACKUP/RMAN/df_DB01_1_1.bus  
comment=NONE
```

```
channel ORA_DISK_1: backup set complete, elapsed time:  
00:00:03
```

```
Finished backup at 21-MAR-01
```

## Practice 11 Solutions (continued)

3. Create an image copy of the datafiles belonging to the SYSTEM tablespace. The copy should be placed in the \$HOME/BACKUP/RMAN directory with the name of sys0101.cpy. The tag should be SYSTEM01.

```
RMAN> COPY DATAFILE '$HOME/ORADATA/u01/system01.dbf'
TO '$HOME/BACKUP/RMAN/sys0101.cpy'
TAG 'SYSTEM01';
```

Starting copy at 21-MAR-01 using target database controlfile  
instead of recovery catalog

allocated channel: ORA\_DISK\_1

channel ORA\_DISK\_1: sid=9 devtype=DISK

channel ORA\_DISK\_1: copied datafile 1

output filename=/databases/db01/BACKUP/RMAN/sys0101.cpy

recid=1 stamp=424947138

Finished copy at 21-MAR-01

4. Using RMAN, back up the archived logs generated today to the \$HOME/BACKUP/RMAN directory.

```
RMAN> BACKUP ARCHIVELOG FROM TIME 'SYSDATE-1';
```

Starting backup at 21-MAR-01

current log archived

using channel ORA\_DISK\_1

channel ORA\_DISK\_1: starting archive log backupset

channel ORA\_DISK\_1: specifying archive log(s) in backup set

input archive log thread=1 sequence=33 recid=5

stamp=424878058

...

input archive log thread=1 sequence=43 recid=25

stamp=424948086

channel ORA\_DISK\_1: starting piece 1 at 21-MAR-01

channel ORA\_DISK\_1: finished piece 1 at 21-MAR-01

piece handle=/databases/oracle9i/dbs/02cl8cbn\_1\_1

comment=NONE

channel ORA\_DISK\_1: backup set complete, elapsed time:

00:00:03

Finished backup at 21-MAR-01

## Practice 11 Solutions (continued)

5. Obtain a listing of all data files that have not been backed up.

```
RMAN> REPORT NEED BACKUP;
```

```
RMAN retention policy will be applied to the command
```

```
RMAN retention policy is set to redundancy 1
```

```
Report of files with less than 1 redundant backups
```

| File | #bkps | Name                                        |
|------|-------|---------------------------------------------|
| 2    | 0     | /databases/db01/ORADATA/u02/undotbs.dbf     |
| 3    | 0     | /databases/db01/ORADATA/u03/users01.dbf     |
| 4    | 0     | /databases/db01/ORADATA/u03/indx01.dbf      |
| 6    | 0     | /databases/db01/ORADATA/u01/querydata01.dbf |

## Practice 12-1 Solutions

### Complete Database Recovery: Archivelog Mode

1. Shut down the database and disable automatic archiving. Start the instance and mount the database. Set the database in Noarchivelog mode and then open the database. Confirm the status by issuing the ARCHIVE LOG LIST command.

```
SQL> shutdown immediate
Database closed.
Database dismounted.
ORACLE instance shut down.
SQL> exit
$ vi $HOME/ADMIN/PFILE/init<sid>.ora
comment out the log_archive_start parameter
$ sqlplus /nolog
SQL> connect / as sysdba
Connected to an idle instance.
SQL> startup mount pfile=$HOME/ADMIN/PFILE/init<sid>.ora
SQL> alter database noarchivelog;
sql> alter database open;
sql> archive log list;
Database log mode                No Archive Mode
Automatic archival                Disabled
Archive destination               /databases/db01/ORADATA/ARCHIVE2/
Oldest online log sequence       69
Current log sequence              70
```

2. Shut down the database and perform a full closed backup using operating system commands to copy the files to the \$HOME/BACKUP/NOARCH directory. Verify the your copy is complete. Start the instance, mount and open the database.

```
SQL> shutdown immediate
SQL> exit
$ cp -r $HOME/ORADATA/u* $HOME/BACKUP/NOARCH
$ ls -alr $HOME/BACKUP/NOARCH/*
$ sqlplus /nolog
SQL> connect / as sysdba
SQL> startup pfile=$HOME/ADMIN/PFILE/init<sid>.ora
```



## Practice 12-1 Solutions (continued)

3. Run the `$HOME/STUDENT/LABS/emphist.sql` script. This script creates a new table named EMPHIST in the HR schema and adds rows to it. Query the table to obtain a count of the number of rows.

```
SQL> @$HOME/STUDENT/LABS/emphist.sql
SQL> SELECT count(*) FROM hr.emphist;
COUNT(*)
-----
45
1 row selected
```

4. Connect as system/manager and issue the following query to obtain the names of datafiles that contain the EMPHIST table:

```
SQL> select f.file_name from dba_tables t, dba_data_files f
2> where table_name = 'EMPHIST' and
3> t.tablespace_name=f.tablespace_name;
FILE_NAME
-----
/databases/db01/ORADATA/u03/users01.dbf
```

5. Run the `$HOME/STUDENT/LABS/breakdb.sql` in SQL\*Plus to simulate failure.

```
SQL> @$HOME/STUDENT/LABS/breakdb.sql
```

6. Attempt to restart the database normally. What happened?

```
SQL> startup pfile=$HOME/ADMIN/PFILE/init<sid>.ora
ORACLE instance started.

...
Database mounted.
ORA-01157: cannot identify/lock data file 3-see DBWR trace
file
ORA-01110: data file
3: '/databases/db01/ORADATA/u03/users01.dbf'
```

The Oracle server cannot open datafile number 3. Therefore, the database is left in the mount state. The files for the USERS tablespace cannot be located because of perceived media failure.

7. Shut down the database and use the appropriate operating system command to replace the current database with the latest backup (Hint: Copy from the NOARCH directory to the ORADATA directory).

```
SQL> connect / as sysdba;
SQL> shutdown abort;
SQL> !cp -rp $HOME/BACKUP/NOARCH/u* $HOME/ORADATA
```

## Practice 12-1 Solutions (continued)

8. Start up and open the database so that it will be available to all users.

```
SQL> connect / as sysdba;
```

```
SQL> startup pfile=$HOME/ADMIN/PFILE/init<sid>.ora
```

9. Connect to the database as hr/hr and execute a query against the EMPHIST table. What happened and why?

```
SQL> connect hr/hr;
```

```
SQL> SELECT * FROM emphist;
```

```
ORA-00942: table or view does not exist
```

The table does not exist because it was created after the last backup was taken.

10. What conclusions can you make about offline backups and recovery for databases in Noarchivelog mode?

Offline backups can be used to restore the database. Databases in noarchivelog mode do not have archived redo log files that can be used to recover to the point of failure.

Therefore, all changes after the previous backup have been lost. This explains why the EMPHIST table no longer exists.

## Practice 12-2 Solutions

### Complete Database Recovery: Archivelog Mode

1. Query the V\$DATABASE view to determine the archive log mode of the database. Use ARCHIVE LOG LIST to check the status of automatic archiving.

```
SQL> select dbid, name, log_mode from v$database;
```

| DBID       | NAME  | LOG_MODE     |
|------------|-------|--------------|
| -----      | ----- | -----        |
| 1943591421 | DB01  | NOARCHIVELOG |

1 row selected.

```
SQL> archive log list;
```

|                            |                                   |
|----------------------------|-----------------------------------|
| Database log mode          | No Archive Mode                   |
| Automatic archival         | Disabled                          |
| Archive destination        | /databases/db01/ORADATA/ARCHIVE2/ |
| Oldest online log sequence | 69                                |
| Current log sequence       | 70                                |

2. Shut down the instance and configure automatic archiving. Mount the database and use the ALTER DATABASE command to set the database in Archivelog mode.

```
SQL> shutdown immediate;
```

```
SQL> exit
```

Edit the init.ora file to set the LOG\_ARCHIVE\_START parameter.

```
$ sqlplus /nolog
```

```
SQL> connect / as sysdba
```

```
SQL> startup mount pfile=$HOME/ADMIN/PFILE/init<sid>.ora
```

```
SQL> alter database archivelog;
```

```
SQL> alter database open;
```

3. Verify your changes with the ARCHIVE LOG LIST command. Note the current log sequence number.

```
SQL> archive log list;
```

|                              |                                   |
|------------------------------|-----------------------------------|
| Database log mode            | Archive Mode                      |
| Automatic archival           | Enabled                           |
| Archive destination          | /databases/db01/ORADATA/ARCHIVE2/ |
| Oldest online log sequence   | 69                                |
| Next log sequence to archive | 70                                |
| Current log sequence         | 70                                |

## Practice 12-2 Solutions (continued)

4. Perform a closed database backup. Store the backup in the \$HOME/BACKUP/UMAN directory.

```
SQL> shutdown immediate;
SQL> !cp -rp $HOME/ORADATA/u* $HOME/BACKUP/UMAN
SQL> exit
$ sqlplus /nolog
SQL> connect / as sysdba
SQL > startup pfile=$HOME/ADMIN/PFILE/init<sid>.ora
```

5. Run the \$HOME/STUDENT/LABS/emphist.sql script. This script creates a new table named EMPHIST in the HR schema and adds rows to it. Issue a query against the EMPHIST table to determine how many rows it contains.

```
SQL> @$HOME/STUDENT/LABS/emphist.sql
SQL> SELECT count(*) FROM hr.emphist;
COUNT(*)
-----
45
1 row selected
```

6. Connect as system/manager and run the \$HOME/STUDENT/LABS/checktbs.sql script and note the datafiles associated with the tablespace that contains the EMPHIST table.

```
SQL> connect system/manager
SQL> @$HOME/STUDENT/LABS/checktbs
FILE_NAME
-----
/databases/db01/ORADATA/u03/users01.dbf
1 row selected.
```

7. Run the \$HOME/STUDENT/LABS/breakdb.sql script to simulate hardware failure.

```
SQL> @$HOME/STUDENT/LABS/breakdb.sql
```

8. Attempt to start the database normally. What happened?

```
SQL> startup pfile=$HOME/ADMIN/PFILE/init<sid>.ora
ORACLE instance started.
Database mounted.
ORA-01157: cannot identify/lock data file 3 - see DBWR trace file
ORA-01110: data file 3:
'/databases/db01/ORADATA/u03/users01.dbf'
```

The Oracle server cannot open datafile number 3. Therefore, the database is left in mount mode.

## Practice 12-2 Solutions (continued)

9. The Oracle server cannot locate the files for the USERS tablespace because of perceived media failure. Because archiving is enabled, you can now perform a complete recovery.

Restore the data files for the USERS tablespace from the backup that you made in step 4.

```
$ cp -p $HOME/BACKUP/UMAN/u03/users01.dbf
$HOME/ORADATA/u03/users01.dbf
```

10. Use the RECOVER DATABASE command to recover the database.

```
SQL> recover automatic database;
```

11. When recovery is complete, open the database to make it available for all users.

```
SQL> alter database open;
```

12. Query the DBA\_TABLESPACES view to see if the USERS tablespace is online.

```
SQL > select tablespace_name, status from dba_tablespaces
2 > where tablespace_name = 'USERS';
```

| TABLESPACE_NAME | STATUS |
|-----------------|--------|
| USERS           | ONLINE |

1 row selected.

13. Execute a query against the HR.EMPHIST table. What happened?

```
SQL> SELECT count(*) FROM hr.emphist;
COUNT(*)
-----
55
1 row selected
```

**Note:** The breakdb.sql script executes the moreemphist.sql script which inserts additional rows into the HR.EMPHIST table.

14. Connect as system/manager and query the V\$LOG view and note the sequence number. Compare the values with the values found in step 3. What conclusions can you make about complete recovery?

| GROUP# | THREAD# | SEQUENCE# | BYTES    | MEMBERS | ARC | STATUS... |
|--------|---------|-----------|----------|---------|-----|-----------|
| 1      | 1       | 77        | 10485760 | 2       | YES | INACTIVE  |
| 2      | 1       | 78        | 10485760 | 2       | NO  | CURRENT   |

The log sequence numbers are higher than in step 3 when the database backup was taken. During recovery, archived redo log files have been applied, and the database has been brought forward to the current point in time.

## Practice 12-3 Solutions

1. Run the \$HOME/STUDENT/LABS/breakdb.sql script to simulate hardware failure.

```
SQL> @$HOME/STUDENT/LABS/breakdb.sql
```

2. Attempt to restart the instance and open the database. What happened?

```
SQL> startup pfile=$HOME/ADMIN/PFILE/init<sid>.ora
```

```
ORACLE instance started.
```

```
Database mounted.
```

```
ORA-01157: cannot identify data file 3- file not found
```

```
ORA-01110: data file 3: '/.../ORADATA/u03/users01.dbf'
```

The Oracle server cannot open datafile number 3. The database is left in mount mode.

3. You can now perform complete recovery. Take the datafiles for the USERS tablespace offline.

```
SQL > alter database datafile
```

```
    '$HOME/ORADATA/u03/users01.dbf' offline;
```

4. Open the database to make it available for all users.

```
SQL> alter database open;
```

5. Take the USERS tablespace offline, then restore all datafiles from the backup.

```
SQL> ALTER TABLESPACE users OFFLINE IMMEDIATE;
```

```
$ cp $HOME/BACKUP/UMAN/u03/users01.dbf $HOME/ORADATA/u03
```

6. Use the RECOVER TABLESPACE command to recover the tablespace.

```
SQL> RECOVER AUTOMATIC TABLESPACE users;
```

7. Put the USERS tablespace back online.

```
SQL> ALTER TABLESPACE users ONLINE;
```

8. Execute a query against the HR.EMP HIST table.

```
SQL> connect hr/hr;
```

```
SQL> SELECT COUNT(*) FROM hr.employ;
COUNT(*)
```

```
-----
```

```
65
```

```
1 row selected
```

## Practice 12-4 Solutions

1. Run the `$HOME/STUDENT/LABS/newtbs.sql` script as the user `SYSTEM` to:
  - Create a new tablespace with a new datafile
  - Create a table named `NEW_EMP` in the `HR` schema in the new tablespace
  - Simulate the loss of the new datafile

```
SQL> @$HOME/STUDENT/LABS/newtbs.sql
```

2. Connect as `hr/hr` and update the rows in the `NEW_EMP` table as follows: What happened?

```
SQL> UPDATE new_emp
```

```
2> SET salary = salary * 1.1;
```

```
UPDATE new_emp
```

```
*
```

```
ERROR at line 1:
```

```
ORA-01116: error in opening database file 7
```

```
ORA-01110: data file 7:
```

```
'/databases/db01/ORADATA/u04/newusers01.dbf'
```

```
ORA-27041: unable to open file
```

```
SVR4 Error: 2: No such file or directory
```

```
Additional information: 3
```

The Oracle server cannot locate the file for the `NEW_USERS` tablespace.

3. You can perform a complete recovery after the re-creation of the file for which you have no backup. Connect as `sysdba`. You can either take the datafile for the `NEW_USERS` tablespace offline, or take the tablespace offline, because it only contains one datafile.

**Note:** The immediate option must be included to avoid a checkpoint trying to write to a file which does not exist:

```
SQL> CONNECT / AS SYSDBA
```

```
SQL> ALTER TABLESPACE new_users OFFLINE IMMEDIATE;
```

```
Tablespace altered.
```

Confirm the recovery status by querying `V$RECOVER_FILE`.

```
SQL> select * from v$recover_file;
```

| FILE# | ONLINE | ERROR | CHANGE# | TIME |
|-------|--------|-------|---------|------|
|-------|--------|-------|---------|------|

-----

|   |         |                |  |   |
|---|---------|----------------|--|---|
| 7 | OFFLINE | FILE NOT FOUND |  | 0 |
|---|---------|----------------|--|---|

## Practice 12-4 Solutions (continued)

4. You must now re-create the file.

```
SQL > ALTER DATABASE CREATE DATAFILE  
2 > '$HOME/ORADATA/u04/newusers01.dbf';
```

Database altered.

```
SQL> select * from v$recover_file;
```

| FILE# | ONLINE  | ERROR | CHANGE# | TIME      |
|-------|---------|-------|---------|-----------|
| 7     | OFFLINE |       | 248621  | 22-MAR-01 |

5. Use the RECOVER TABLESPACE command to apply the redo logs to the datafile.

```
SQL > RECOVER TABLESPACE new_users;
```

6. When recovery is complete, bring the tablespace online.

```
SQL > ALTER TABLESPACE new_users ONLINE;
```

All data is now recovered. Include the file in the backup strategy and notify users that the tablespace is ready to be used again.

7. Try again to update the rows in the HR.NEW\_EMP table as follows:

```
SQL> UPDATE new_emp  
2> SET salary = salary * 1.1;
```

8. Drop the NEW\_USERS tablespace and associated datafiles in preparation for later practices.

```
SQL > DROP TABLESPACE new_users  
2 > INCLUDING CONTENTS AND DATAFILES;
```



## Practice 12-5 Solutions

In this practice you will simulate a failure in the database while performing an online backup of the SAMPLE tablespace. You will need to issue the appropriate commands to recover and reopen the database.

1. Begin the online backup of the SAMPLE tablespace by issuing the appropriate command in SQL\*Plus.

```
SQL> ALTER TABLESPACE sample BEGIN BACKUP;
```

```
Tablespace altered.
```

2. Make an OS backup of the files belonging to the SAMPLE tablespace in the \$HOME/BACKUP/UMAN directory.

```
$ cp $HOME/ORADATA/u02/sample01.dbf $HOME/BACKUP/UMAN
```

3. Issue the SHUTDOWN ABORT command in SQL\*Plus.

```
SQL> shutdown abort;
```

4. Start the instance and mount the database.

```
SQL> connect / as sysdba;
```

```
SQL> startup mount pfile=$HOME/ADMIN/PFILE/init<sid>.ora;
```

```
ORACLE instance started.
```

```
Total System Global Area      21797632 bytes
```

```
Fixed Size                      285440 bytes
```

```
Variable Size                  16777216 bytes
```

```
Database Buffers               4194304 bytes
```

```
Redo Buffers                    540672 bytes
```

```
Database mounted.
```

5. Query V\$BACKUP to determine if any files are in an online backup.

```
SQL> SELECT * FROM v$backup;
```

| FILE# | STATUS     | CHANGE# | TIME      |
|-------|------------|---------|-----------|
| 1     | NOT ACTIVE | 0       |           |
| 2     | NOT ACTIVE | 0       |           |
| 3     | NOT ACTIVE | 0       |           |
| 4     | NOT ACTIVE | 0       |           |
| 5     | ACTIVE     | 107167  | 22-MAR-01 |
| 6     | NOT ACTIVE | 0       |           |

This indicates that file number 5 is currently in online backup mode.

### Practice 12-5 (continued)

6. Issue the appropriate command to end the backup mode and unfreeze the datafile header. Query V\$BACKUP to check the status of the datafile.

```
SQL> ALTER DATABASE datafile 5 END BACKUP;
```

```
Database altered.
```

```
SQL> SELECT * FROM v$backup;
```

| FILE# | STATUS     | CHANGE# | TIME      |
|-------|------------|---------|-----------|
| 1     | NOT ACTIVE | 0       |           |
| ...   |            |         |           |
| 5     | ACTIVE     | 107167  | 22-MAR-01 |
| ...   |            |         |           |

7. Open the database for users.

```
SQL> ALTER DATABASE OPEN;
```

```
Database altered.
```

## Practice 13-1 Solutions

### Tablespace Recovery Using RMAN

1. Configure controlfile autobackup using the following format:  
\$HOME/BACKUP/RMAN/%F.bck  
  
RMAN> configure controlfile autobackup format for device type disk  
  
2> to '\$HOME/BACKUP/RMAN/%F.bck';  
  
using target database controlfile instead of recovery catalog  
  
new RMAN configuration parameters:  
CONFIGURE CONTROLFILE AUTOBACKUP FORMAT FOR DEVICE TYPE DISK  
TO '\$HOME/BACKUP/R';  
  
new RMAN configuration parameters are successfully stored  
RMAN> configure controlfile autobackup on;  
  
new RMAN configuration parameters:  
CONFIGURE CONTROLFILE AUTOBACKUP ON;  
  
new RMAN configuration parameters are successfully stored
2. Make a whole database backup specifying the following format:  
\$HOME/BACKUP/RMAN/df\_%d\_%s\_%p.bus  
  
RMAN> backup database  
  
2> format '\$HOME/BACKUP/RMAN/df\_%d\_%s\_%p.bus';  
  
Starting backup at 25-APR-01  
allocated channel: ORA\_DISK\_1  
channel ORA\_DISK\_1: sid=13 devtype=DISK  
channel ORA\_DISK\_1: starting full datafile backupset  
channel ORA\_DISK\_1: specifying datafile(s) in backupset  
input datafile fno=00001 name=/.../ORADATA/u01/system01.dbf  
input datafile fno=00005 name=/.../ORADATA/u02/sample01.dbf  
input datafile fno=00002 name=/.../ORADATA/u02/undotbs.dbf  
input datafile fno=00003 name=/.../ORADATA/u03/users01.dbf  
input datafile fno=00004 name=/.../ORADATA/u03/indx01.dbf  
input datafile fno=00006 name=/.../ORADATA/u01/querydata01.dbf  
channel ORA\_DISK\_1: starting piece 1 at 25-APR-01  
channel ORA\_DISK\_1: finished piece 1 at 25-APR-01  
piece handle=/.../BACKUP/RMAN/df\_ED21\_3\_1.bus comment=NONE  
channel ORA\_DISK\_1: backup set complete, elapsed time:  
00:00:35

## Practice 13-1 Solutions (continued)

```
Finished backup at 25-APR-01
Starting Control File Autobackup at 25-APR-01
piece handle=/.../BACKUP/RMAN/c-1125003950-20010425-00.bck
comment=NE
Finished Control File Autobackup at 25-APR-01
```

### Tablespace Recovery Using RMAN

3. Connect as sysdba in SQL\*Plus and run the  
\$HOME/STUDENT/LABS/breakdb.sql script.  
SQL> @\$HOME/STUDENT/LABS/breakdb.sql
4. Startup your instance in SQL\*Plus.  
SQL> startup mount pfile=\$HOME/ADMIN/PFILE/init<sid>.ora
5. Use RMAN to restore and recover the USERS tablespace.  
RMAN> connect target  
connected to target database (not started)  
RMAN> restore tablespace users;  
Starting restore at 02-APR-01  
allocated channel: ORA\_DISK\_1  
channel ORA\_DISK\_1: sid=11 devtype=DISK  
channel ORA\_DISK\_1: starting datafile backupset restore  
channel ORA\_DISK\_1: specifying datafile(s) to restore from  
backup set  
restoring datafile 00003 to  
/databases/db01/ORADATA/u03/users\_01\_db01.dbf  
channel ORA\_DISK\_1: restored backup piece 1  
piece handle=/databases/db01/BACKUP/RMAN/df\_DB01\_3\_1.bus  
tag=null params=NULL  
channel ORA\_DISK\_1: restore complete  
Finished restore at 02-APR-01  
RMAN> recover tablespace users;  
Starting recover at 02-APR-01  
using channel ORA\_DISK\_1  
starting media recovery  
archive  
logfilename=/databases/db01/ORADATA/ARCHIVE1/arch\_90.arc  
thread=1 sequ6  
...

### Practice 13-1 Solutions (continued)

```
archive  
logfilename=/databases/db01/ORADATA/ARCHIVE1/arch_95.arc  
thread=1 sequ3  
media recovery complete  
Finished recover at 02-APR-01
```

6. Open the database after recovery completes.

```
RMAN> alter database open;  
using target database controlfile instead of recovery  
catalog  
database opened
```

## Practice 13-2 Solutions

### Relocating a datafile

1. Connect as sysdba in SQL\*Plus and run the \$HOME/STUDENT/LABS/breakdb.sql script.  
SQL> @\$HOME/STUDENT/LABS/breakdb.sql
2. You have determined that u03 (\$HOME/ORADATA/u03) is corrupted. You must relocate the datafile for the USERS tablespace to another location. \$HOME/ORADATA/u04 has sufficient space. Using RMAN, construct a RUN block to relocate the datafile from u03 to u04 and recover the USERS tablespace.

```
RMAN> startup mount pfile=$HOME/ADMIN/PFILE/init<sid>.ora;
run{
  set newname for datafile 3 to
  '$HOME/ORADATA/u04/users01.dbf';
  restore tablespace users;
  switch datafile all;
  recover tablespace users;
  sql 'alter database open';
}
executing command: SET NEWNAME
Starting restore at 02-APR-01
using channel ORA_DISK_1
channel ORA_DISK_1: starting datafile backupset restore
channel ORA_DISK_1: specifying datafile(s) to restore from
backup set
restoring datafile 00003 to
/databases/db01/ORADATA/u04/users01.dbf
channel ORA_DISK_1: restored backup piece 1
piece handle=/databases/db01/BACKUP/RMAN/df_DB01_3_1.bus
tag=null params=NULL
channel ORA_DISK_1: restore complete
Finished restore at 02-APR-01
datafile 3 switched to datafile copy
input datafilecopy recid=4 stamp=425996991
filename=/databases/ed01/ORADATA/u04f
Starting recover at 02-APR-01
using channel ORA_DISK_1
starting media recovery
archive
logfilename=/databases/db01/ORADATA/ARCHIVE1/arch_90.arc
thread=1 sequ6
archive
logfilename=/databases/db01/ORADATA/ARCHIVE1/arch_103.arc
thread=1 sequ4
media recovery complete
Finished recover at 02-APR-01
```

## Practice 14-1 Solutions

### Recovering from User Failure: Incomplete Recovery

1. If you are unsure whether you have a valid backup from the previous practices, then perform either a whole closed or opened database backup. Store the backup in the \$HOME/BACKUP/UMAN directory. Start the instance and mount the database.

```
SQL> shutdown immediate;
```

```
SQL> !cp -rp $HOME/ORADATA/u* $HOME/BACKUP/UMAN
```

```
SQL> startup pfile=$HOME/ADMIN/PFILE/init<sid>.ora;
```

2. Connect as hr/hr. Insert rows into the EMPHIST table by issuing the following statement:

```
SQL> INSERT INTO emphist SELECT * FROM emphist;
```

```
SQL> COMMIT;
```

3. Issue a SELECT statement to obtain a count of the rows in the EMPHIST table. Note the number of rows.

```
SQL> SELECT COUNT(*) FROM emphist;
```

```
COUNT(*)
```

```
-----
```

```
170
```

4. Connect as system/manager and issue the following query:

```
SQL> SELECT f.file_name FROM dba_tables t, dba_data_files f
      2> WHERE table_name = 'EMPHIST' AND
      3> t.tablespace_name = f.tablespace_name;
```

Record the filename of all datafiles for the tablespace.

```
$HOME/ORADATA/u04/users01.dbf
```

5. Record the current system time using an operating system command.

```
SQL> !date
```

```
Thu Mar 22 14:34:41 PST 2001
```

6. Query V\$LOG to find the current online log sequence number..

```
SQL> SELECT * FROM v$log;
```

| GROUP# | THREAD# | SEQUENCE# | BYTES    | MEMBERS | ARC | STATUS... |
|--------|---------|-----------|----------|---------|-----|-----------|
| 1      | 1       | 105       | 10485760 | 2       | YES | INACTIVE  |
| 2      | 1       | 106       | 10485760 | 2       | NO  | CURRENT   |

7. Connect as hr/hr and add rows to the EMPHIST table by executing the following command:

```
$ sqlplus hr/hr
```

```
SQL> INSERT INTO emphist SELECT * FROM emphist;
```

```
SQL> COMMIT;
```

## Practice 14-1 Solutions (continued)

8. Issue a SELECT statement to obtain a count of the rows in the EMPHIST table. Note the number of rows.

```
SQL> SELECT COUNT(*) FROM emphist;
COUNT(*)
-----
340
```

9. Run the \$HOME/STUDENT/LABS/breaktab.sql script to simulate a user accidentally dropping the EMPHIST table.

```
SQL> @$HOME/STUDENT/LABS/breaktab.sql
```

10. Attempt to query the EMPHIST table. What happened?

```
SQL> SELECT * FROM hr.emphist;
ORA-00942: table or view does not exist
The table does not exist any more.
```

11. The Oracle server cannot locate the EMPHIST table. You need to restore this table to the database. Since archiving is enabled and you know the approximate time of failure, you can now perform an incomplete recovery to restore the table.

Shut down the instance.

```
SQL> shutdown immediate
```

12. Restore all datafiles from the backup that you made in step 1.

```
SQL> !cp $HOME/BACKUP/UMAN/u01/*.dbf $HOME/ORADATA/u01
SQL> !cp $HOME/BACKUP/UMAN/u02/*.dbf $HOME/ORADATA/u02
SQL> !cp $HOME/BACKUP/UMAN/u03/*.dbf $HOME/ORADATA/u03
```

If you did not take a backup at the beginning of this practice, you need to restore the datafile for the USERS tablespace as follows:

```
SQL> !cp $HOME/BACKUP/UMAN/u03/users01.dbf $HOME/ORADATA/u04
```

13. Start the instance and mount the database. Recover the database until the time you noted in step 4.

```
SQL> startup mount pfile=$HOME/ADMIN/PFILE/init<sid>.ora
SQL> RECOVER DATABASE UNTIL TIME '2001-03-22:14:34:41'
```

14. When recovery is complete, open the database using the Resetlogs option.

```
SQL> ALTER DATABASE OPEN RESETLOGS;
```



## Practice 14-1 Solutions (continued)

15. Connect as hr/hr and execute a query against the EMPHIST table. What happened and why?

```
SQL> connect hr/hr
SQL> SELECT COUNT(*) FROM emphist;
COUNT(*)
-----
170
1 row selected
```

The table exists again, because the entire database is taken back to a time before the table was dropped. However, you lost the rows that were inserted after the time to which you recovered the database.

16. Connect as system/manager, query the V\$LOG view, and note the sequence number. Compare this value with the value in step 5. What conclusions can you make about incomplete recovery?

```
SQL> SELECT * FROM v$log;
GROUP#  THREAD#  SEQUENCE#  BYTES      MEMBERS  ARC  STATUS...
-----  -
1         1           0 10485760         2 YES  UNUSED
2         1           1 10485760         2 NO   CURRENT
```

The sequence numbers are reset to 1.

17. Take a whole offline backup. Store the backup in the \$HOME/BACKUP/UMAN directory.

```
SQL> connect / as sysdba
SQL> shutdown immediate
SQL> !cp -rp $HOME/ORADATA/u* $HOME/BACKUP/UMAN
```

## Practice 14-2 Solutions

### Recovery with a Lost Archived Log: Incomplete Recovery

1. Start the instance and open the database. Determine the current system time using an operating system command.

```
SQL> startup
```

```
SQL> !date
```

```
Fri Mar 23 07:35:46 PST 2001
```

2. Query the V\$LOG view and record the current online log sequence number.

```
SQL> SELECT * FROM v$log;
```

| GROUP# | THREAD# | SEQUENCE# | BYTES    | MEMBERS | ARC | STATUS... |
|--------|---------|-----------|----------|---------|-----|-----------|
| 1      | 1       | 0         | 10485760 | 2       | YES | UNUSED    |
| 2      | 1       | 1         | 10485760 | 2       | NO  | CURRENT   |

3. Run the \$HOME/STUDENT/LABS/moredata.sql script to switch the logs and create a new table.

```
SQL> @$HOME/STUDENT/LABS/moredata.sql
```

4. Run the \$HOME/STUDENT/LABS/breakarc.sql script to simulate the loss of an archived redo log file.

```
SQL> @$HOME/STUDENT/LABS/breakarc.sql
```

5. Run the \$HOME/STUDENT/LABS/breakdb.sql script to simulate hardware failure.

```
SQL> @$HOME/STUDENT/LABS/breakdb.sql
```

6. Attempt to restart the database normally. What happened?

```
SQL> startup
```

```
ORACLE instance started.
```

```
...
```

```
Database mounted.
```

```
ORA-01157:cannot identify/lock data file 3 - see DBWR trace file
```

```
ORA-01110:data file 3:
```

```
'/databases/db01/ORADATA/u04/users01.dbf'
```

The Oracle server cannot open datafile number 3. The database is left in mount mode.

7. The Oracle server cannot locate the files for the USERS tablespace because of perceived media failure. Since archiving is enabled, you can attempt to perform a complete recovery.

Restore the data files for the USERS tablespace from the backup you made in Practice 14-1.

```
$cp $HOME/BACKUP/UMAN/u04/users01.dbf $HOME/ORADATA/u04
```

## Practice 14-2 Solutions (continued)

8. Use the RECOVER AUTOMATIC DATABASE command to recover the database. Note the name of any files that cannot be found. Issue a CANCEL when the Oracle server is unable to locate the specified archive log.

```
SQL> RECOVER AUTOMATIC DATABASE
ORA-00279: change 87937 generated at 03/23/2001 08:49:39
needed for thread 1
ORA-00289:suggestion: /.../ORADATA/ARCHIVE2/arch_32.arc
ORA-00280: change 87937 for thread 1 is in sequence #3
ORA-00278: log file
'/databases/db01/ORADATA/ARCHIVE2/arch_32.arc' no longer
needed for this recovery
ORA-00308: cannot open archived log
'/databases/db01/ORADATA/ARCHIVE2/arch_32.arc'
ORA-27037: unable to obtain file status
SVR4 Error: 2: No such file or directory
Additional information: 3
Specify log: {<RET>=suggested | filename | AUTO | CANCEL}
CANCEL
Media recovery cancelled.
```

9. Attempt to open the database. What happened?

```
SQL> ALTER DATABASE OPEN;
ORA-01113: file 3 needs media recovery
ORA-01110: data file 3: '/.../ORADATA/u04/users01.dbf'
```

The datafile requires more recovery to become synchronized with the other datafiles.

10. The recovery has been cancelled prior to applying the lost archived log. The datafiles in the USERS tablespace cannot be brought forward to the current database time. Since recovery cannot take the database back in time, you must perform an incomplete recovery.

Restore all data files from the backup you made in Practice 14-1.

```
SQL> !cp $HOME/BACKUP/UMAN/u01/*.dbf $HOME/ORADATA/u01
SQL> !cp $HOME/BACKUP/UMAN/u02/*.dbf $HOME/ORADATA/u02
SQL> !cp $HOME/BACKUP/UMAN/u03/*.dbf $HOME/ORADATA/u03
SQL> !cp $HOME/BACKUP/UMAN/u04/*.dbf $HOME/ORADATA/u04
```

## Practice 14-2 Solutions (continued)

11. Recover the database using the UNTIL CANCEL option, stopping when the Oracle server requests the archived log file you noted in step 8.

**Note:** Do not use the automatic method. Apply each archived log manually as the Oracle server requests it.

```
SQL> RECOVER DATABASE UNTIL CANCEL
```

```
ORA-00279: change 87837 generated at 03/22/2001 15:06:39  
needed for thread 1
```

```
ORA-00289: suggestion :  
/databases/db01/ORADATA/ARCHIVE2/arch_1.arc
```

```
ORA-00280: change 87837 for thread 1 is in sequence #1
```

```
...
```

```
ORA-00279: change 87937 generated at 03/23/2001 08:49:39  
needed for thread 1
```

```
ORA-00289: suggestion :  
/databases/db01/ORADATA/ARCHIVE2/arch_3.arc
```

```
ORA-00280: change 87937 for thread 1 is in sequence #3
```

```
ORA-00278: log file  
'/databases/db01/ORADATA/ARCHIVE2/arch_2.arc' no longer  
needed for this recovery
```

12. Type cancel at the recovery prompt.

```
CANCEL
```

```
Media recovery cancelled.
```

13. Once recovery is complete, open the database using the RESETLOGS option.

```
SQL> ALTER DATABASE OPEN RESETLOGS;
```

```
Statement processed.
```

14. Query V\$DATAFILE to verify that all datafiles are online

```
SQL> SELECT name, status FROM v$datafile;
```

| NAME                                       | STATUS |
|--------------------------------------------|--------|
| /databases/db01/ORADATA/u01/system01.dbf   | SYSTEM |
| /databases/db01/ORADATA/u02/undotbs.dbf    | ONLINE |
| /databases/db01/ORADATA/u04/users01.dbf    | ONLINE |
| /databases/db01/ORADATA/u03/indx01.dbf     | ONLINE |
| /databases/db01/ORADATA/u02/sample01.dbf   | ONLINE |
| /databases/db01/ORADATA/u01/querydata0.dbf | ONLINE |

15. Take a whole offline backup. Store the backup in the \$HOME/BACKUP/UMAN directory.

```
SQL> connect / as sysdba
```

```
SQL> shutdown immediate
```

```
SQL> !cp -rp $HOME/ORADATA/u* $HOME/BACKUP/UMAN
```

## Practice 15 Solutions

### RMAN Recovery with a Lost Archived Log: Incomplete Recovery

1. Make a whole database backup using RMAN specifying  
RMAN> backup database  
2> format '\$HOME/BACKUP/RMAN/df\_%d\_%s\_%p.bus' ;
2. Run the \$HOME/STUDENT/LABS/moredata.sql script to switch the logs and create a new table.  
SQL> @\$HOME/STUDENT/LABS/moredata.sql
3. Run the \$HOME/STUDENT/LABS/breakarc.sql script to simulate the loss of an archived redo log file.  
SQL> @\$HOME/STUDENT/LABS/breakarc.sql
4. Run the \$HOME/STUDENT/LABS/breakdb.sql script to simulate hardware failure.  
SQL> @\$HOME/STUDENT/LABS/breakdb.sql
5. Attempt to restart the database normally. What happened?  
SQL> startup  
ORACLE instance started.  
  
...  
Database mounted.  
ORA-01157:cannot identify/lock data file 3 - see DBWR trace file  
ORA-01110:data file 3:  
'/databases/db01/ORADATA/u04/users01.dbf'  
  
The Oracle server cannot open datafile number 3. The database is left in mount mode.
6. The Oracle server cannot locate the files for the USERS tablespace because of perceived media failure. Since archiving is enabled, you can attempt to perform a complete recovery. Use RMAN to restore the datafiles for the USERS tablespace.  
RMAN> restore tablespace users;  
Starting restore at 26-APR-01  
using target database controlfile instead of recovery catalog  
allocated channel: ORA\_DISK\_1  
channel ORA\_DISK\_1: sid=11 devtype=DISK  
channel ORA\_DISK\_1: starting datafile backupset restore  
channel ORA\_DISK\_1: specifying datafile(s) to restore from backup set  
restoring datafile 00003 to  
/databases/ed21/ORADATA/u04/users01.dbf  
channel ORA\_DISK\_1: restored backup piece 1

## Practice 15 Solutions (continued)

```
piece handle=/databases/ed21/BACKUP/RMAN/df_ED21_5_1.bus
tag=null params=NULL
```

```
channel ORA_DISK_1: restore complete
```

```
Finished restore at 26-APR-01
```

7. Use RMAN to recover the tablespace. Note the name and sequence number of any files that cannot be found.

```
RMAN> recover tablespace users;
```

```
Starting recover at 26-APR-01
```

```
using channel ORA_DISK_1
```

```
starting media recovery
```

```
archive log thread 1 sequence 1 is already on disk as file
/databases/ed21/ORADc
```

```
RMAN
```

```
00571:=====
===
```

```
RMAN-00579: the following error occurred at 04/26/2001
13:12:50
```

```
RMAN-03002: failure during compilation of command
```

```
RMAN-03013: command type: recover
```

```
RMAN-03002: failure during compilation of command
```

```
RMAN-03013: command type: recover(4)
```

```
RMAN-06053: unable to perform media recovery because of
missing log
```

```
RMAN-06025: no backup of log thread 1 seq 2 scn 168396 found
to restore
```

8. Use RMAN with the UNTIL LOG SEQUENCE clause to perform incomplete recovery through the last good archived redo log file.

```
RMAN> RUN {
```

```
2> SET UNTIL SEQUENCE 2 THREAD 1;
```

```
3> RESTORE DATABASE;
```

```
4> RECOVER DATABASE;
```

```
5> }
```

```
executing command: SET until clause
```

```
Starting restore at 26-APR-01
```

```
using channel ORA_DISK_1
```

## Practice 15 Solutions (continued)

```
datafile 6 not processed because file is read-only
skipping datafile 3; already restored to file
/databases/ed21/ORADATA/u04/users01.dbf
channel ORA_DISK_1: starting datafile backupset restore
channel ORA_DISK_1: specifying datafile(s) to restore from
backup set
restoring datafile 00001 to
/databases/ed21/ORADATA/u01/system01.dbf
restoring datafile 00002 to
/databases/ed21/ORADATA/u02/undotbs.dbf
restoring datafile 00004 to
/databases/ed21/ORADATA/u03/indx01.dbf
restoring datafile 00005 to
/databases/ed21/ORADATA/u02/sample01.dbf
channel ORA_DISK_1: restored backup piece 1
piece handle=/databases/ed21/BACKUP/RMAN/df_ED21_5_1.bus
tag=null params=NULL
channel ORA_DISK_1: restore complete
Finished restore at 26-APR-01
Starting recover at 26-APR-01
using channel ORA_DISK_1
datafile 6 not processed because file is read-only
starting media recovery
archive log thread 1 sequence 1 is already on disk as file
/databases/ed21/ORADc
archive log
filename=/databases/ed21/ORADATA/ARCHIVE1/arch_1.arc
thread=1 sequel
media recovery complete
Finished recover at 26-APR-01
```

9. Once recovery is complete, open the database using the RESETLOGS option.

```
RMAN> ALTER DATABASE OPEN RESETLOGS;
database opened
```

10. Make a new backup in the \$HOME/BACKUP/RMAN directory with the following format:

```
df_%d_%s_%p.bus
RMAN> backup database
2> format '$HOME/BACKUP/RMAN/df_%d_%s_%p.bus';
```

## Practice 16 Solutions

1. Connect to your database in the default Nocatlog mode.

```
$ rman target /
Recovery Manager: Release 9.0.0.0.0 - Beta
(c) Copyright 2000 Oracle Corporation. All rights reserved.
connected to target database: DB01 (DBID=1121888154)
```

2. Use the RMAN REPORT command to generate a listing of your database structure.

```
RMAN> REPORT SCHEMA;
using target database controlfile instead of recovery
catalog
Report of database schema
File K-bytes Tablespace RB Datafile Name
-----
1      102400 SYSTEM      *** /.../ORADATA/u01/system01.dbf
2       30720 UNDOTBS     *** /.../ORADATA/u02/undotbs.dbf
3       51200 USERS       *** /.../ORADATA/u03/users01.dbf
4        5120 INDX        *** /.../ORADATA/u03/indx01.dbf
5        1024 QUERY_DATA *** /.../ORADATA/u01/querydata01.dbf
6       10240 SAMPLE      *** /.../ORADATA/u02/sample01.dbf
```

3. Use the RMAN LIST and CROSSCHECK commands to generate a listing of the backup sets and the status of the files.

```
RMAN> LIST BACKUP;
List of Backup Sets
=====
BS Key Type LV SizeDevice Type Elapsed Time Completion Time
-----
1      Full 1M          DISK 00:00:01      21-MAR-01
BP Key: 1 Status: AVAILABLE Tag:
Piece Name: /databases/db01/BACKUP/RMAN/df_DB01_1_1.bus
Controlfile Included: Ckp SCN: 66090 Ckp time: 21-MAR-01
List of Datafiles in backup set 1
File LV Type Ckp SCN Ckp Time Name
-----
6      Full          66091 21-MAR-01 /.../ORADATA/u03/sample01.dbf
```



## Practice 16 Solutions (continued)

```
RMAN> CROSSCHECK BACKUPSET 1;
using channel ORA_DISK_1
crosschecked backup piece: found to be 'AVAILABLE'
backup piece
handle=/databases/db01/BACKUP/RMAN/df_DB01_1_1.bus recid=1
stamp=46
```

4. Using an operating system command, copy the backup file belonging to the SAMPLE tablespace to your BACKUP directory and then remove it from the RMAN directory to simulate a loss of the backup.

```
$cp $HOME/BACKUP/RMAN/df_DB01_1_1.bus $HOME/BACKUP
$rm $HOME/BACKUP/RMAN/df_DB01_1_1.bus
```

5. Use the RMAN CROSSCHECK command to update the repository with the status of the datafile backup that you moved in the previous step. Be sure to specify the backup set that you moved in the previous step.

```
RMAN> CROSSCHECK BACKUPSET 1;
using target database controlfile instead of recovery
catalog
allocated channel: ORA_DISK_1
channel ORA_DISK_1: sid=9 devtype=DISK
crosschecked backup piece: found to be 'EXPIRED'
backup piece
handle=/databases/db01/BACKUP/RMAN/df_DB01_1_1.bus recid=1
stamp=46
```

6. Issue the LIST EXPIRED command to check the status of your files. Are any of your files expired?

```
RMAN> LIST EXPIRED BACKUP;
List of Backup Sets
=====
BS Key Type LV Size Device Type Elapsed Time Completion Time
--- --
1 Full 1M DISK 00:00:01 21-MAR-01
BP Key: 1 Status: EXPIRED Tag:
Piece Name: /databases/db01/BACKUP/RMAN/df_DB01_1_1.bus
Controlfile Included: Ckp SCN: 66090 Ckp time: 21-MAR-01
List of Datafiles in backup set 1
File LV Type Ckp SCN Ckp Time Name
-- --
6 Full 66091 21-MAR-01 /.../ORADATA/u02/sample01.dbf
```

## Practice 16 Solutions (continued)

7. Using an operating system command, return the SAMPLE tablespace backup to the correct location.

```
$mv $HOME/BACKUP/df_DB01_1_1.bus $HOME/BACKUP/RMAN
```

8. Use the RMAN CROSSCHECK command to update the repository with the status of the datafile backup.

```
RMAN> CROSSCHECK BACKUPSET 1;
using target database controlfile instead of recovery
catalog
allocated channel: ORA_DISK_1
channel ORA_DISK_1: sid=9 devtype=DISK
crosschecked backup piece: found to be 'AVAILABLE'
backup piece
handle=/databases/db01/BACKUP/RMAN/df_DB01_1_1.bus recid=1
stamp=46
```

9. Make a backup of the datafile belonging to the SAMPLE tablespace to the \$HOME/BACKUP/RMAN directory with user-managed procedures.

```
SQL> ALTER TABLESPACE sample BEGIN BACKUP;
Tablespace altered.
cp $HOME/ORADATA/u02/sample01.dbf $HOME/BACKUP/RMAN
SQL> ALTER TABLESPACE sample END BACKUP;
Tablespace altered.
```

10. Use the RMAN CATALOG command to update the repository with this backup information.

```
RMAN> CATALOG DATAFILECOPY
'$HOME/BACKUP/RMAN/sample01.dbf';
using target database controlfile instead of recovery
catalog
cataloged datafile copy
datafile copy
filename=/databases/db01/BACKUP/RMAN/sample01.dbf recid=3 s7
```

## Practice 16 Solutions (continued)

11. Use the RMAN LIST COPY command to verify that the backup has been recorded in the repository.

```
RMAN> list copy;
```

```
List of Datafile Copies
```

| Key | F | S | Completion | Ckp SCN | Ckp Time  | Name                             |
|-----|---|---|------------|---------|-----------|----------------------------------|
| --- | - | - | -----      | -----   | -----     | ----                             |
| 1   | 1 | A | 21-MAR-01  | 66107   | 21-MAR-01 | /.../BACKUP/RMAN/<br>sys0101.cpy |
| 3   | 6 | A | 23-MAR-01  | 88207   | 23-MAR-01 | /.../BACKUP/RMAN/<br>sample01... |

## Practice 17 Solutions

1. Execute the `crercts.sql` script to create the `recat` tablespace for the recovery catalog and the `rcuser` schema.

```
SQL> @$HOME/STUDENT/LABS/crercts
```

2. Connect to the recovery catalog database using RMAN. Create the catalog in the `recat` tablespace.

```
rman catalog rcuser/rcuser@<service name>
```

```
Recovery Manager: Release 9.0.0.0.0 - Beta
```

```
(c) Copyright 2000 Oracle Corporation. All rights reserved.
```

```
connected to recovery catalog database
```

```
recovery catalog is not installed
```

```
RMAN> CREATE CATALOG;
```

3. Connect to your target database and recovery catalog using RMAN.

```
rman target / catalog rcuser/rcuser@<service name>
```

```
Recovery Manager: Release 9.0.0.0.0 - Beta
```

```
(c) Copyright 2000 Oracle Corporation. All rights reserved.
```

```
connected to target database: DB01 (DBID=1122749761)
```

```
connected to recovery catalog database
```

4. Execute the command to resynchronize the control file and recovery catalog. What happened? Why?

```
RMAN> resync catalog;
```

```
RMAN-00571:=====
```

```
RMAN-00569: ===== ERROR MESSAGE STACK FOLLOWS =====
```

```
RMAN-00571:=====
```

```
RMAN-00579: the following error occurred at 03/23/2001  
12:47:15
```

```
RMAN-03006: non-retryable error occurred during execution of  
command: resync
```

```
RMAN-12004: unhandled exception during command execution on  
channel default
```

```
RMAN-20001: target database not found in recovery catalog
```

```
The target database is not registered in the catalog.
```

5. Register the target database in the recovery catalog at the RMAN prompt.

```
RMAN> register database;
```

```
database registered in recovery catalog
```

```
starting full resync of recovery catalog
```

```
full resync complete
```

## Practice 17 Solutions (continued)

6. Using RMAN, list all the database incarnations registered in the catalog.

```
RMAN> list incarnation of database;
RMAN-03022: compiling command: list
List of Database Incarnations
DB K Inc K DB Name DB ID CUR Reset SCN Reset Time
----
1 2 DB01 1121888154 YES 87938 23-MAR-01
```

7. Enter the RESET DATABASE command at the RMAN prompt. What happens?

```
RMAN> reset database;
RMAN-00571:=====
RMAN-00569:===== ERROR MESSAGE STACK FOLLOWS=====
RMAN-00571:=====
RMAN-03006: non-retryable error occurred during execution of
command: reset
RMAN-07004: unhandled exception during command execution on
channel default
RMAN-20009: database incarnation already registered
```

8. View the \$HOME/STUDENT/LABS/crebkup.sql script. In SQL\*Plus connect to your target database as system/manger and execute the script to create an online operating system copy of the SAMPLE tablespace datafile in your \$HOME/BACKUP directory.

```
SQL> @$HOME/STUDENT/LABS/crebkup.sql
```

9. Using RMAN, add the backup made in step 6 to the catalog.

```
$rman target / catalog rcuser/rcuser@<service name>
Recovery Manager: Release 9.0.0.0.0 - Beta
(c) Copyright 2000 Oracle Corporation. All rights reserved.
connected to target database: DB01 (DBID=1121888154)
connected to recovery catalog database
RMAN> catalog datafilecopy
2> '$HOME/BACKUP/sample01.cpy'
3> tag 'SAMPLECPY1';
cataloged datafile copy datafile copy
filename=/databases/db01/BACKUP/sample01.cpy recid=4
stamp=4251353
```

## Practice 17 Solutions (continued)

10. Using RMAN, confirm that the data file has been added to the recovery catalog.

```
RMAN> list copy;
```

```
List of Datafile Copies
```

```
Key          File S Completion CkpSC Ckp time  Name
```

```
-----
```

|     |   |   |           |       |           |                          |
|-----|---|---|-----------|-------|-----------|--------------------------|
| 163 | 1 | A | 21-MAR-01 | 66107 | 21-MAR-01 | /.../BACKUP/...          |
| 168 | 6 | A | 23-MAR-01 | 90320 | 23-MAR-01 | /.../BACKUP/...          |
| 165 | 6 | A | 23-MAR-01 | 88207 | 23-MAR-01 | /.../BACKUP/sample01.cpy |

11. Use the RMAN command to remove the backup of the SAMPLE tablespace datafile from the recovery catalog. Do not remove the file from the operating system.

```
RMAN> CHANGE DATAFILECOPY
```

```
2> '$HOME/BACKUP/sample01.cpy'
```

```
3> UNCATALOG;
```

```
uncataloged datafile copy datafile copy
filename=/databases/db01/BACKUP/sample01.cpy recid=4
stamp=4251353
```

12. Using SQL\*Plus, connect to your recovery catalog database and query the RC\_DATAFILE\_COPY view to confirm that the datafile has been removed from the recovery catalog.

```
$ sqlplus rcuser/rcuser@<service name>;
```

```
SQL> SELECT name, db_name, file#
```

```
2> FROM rc_datafile_copy;
```

```
NAME
```

| NAME                                     | DB_NAME | FILE# |
|------------------------------------------|---------|-------|
| /databases/db01/BACKUP/RMAN/sys0101.cpy  | UNKNOWN | 1     |
| /databases/db01/BACKUP/RMAN/sample01.dbf | DB01    | 6     |

## Practice 17 Solutions (continued)

13. Create a script to make a whole database backup with following information:

Name of script: nightback  
Channel name: dbnD (n is the student account number)  
Channel type Disk  
Format \$HOME/BACKUP/RMAN/%b%d%s%p  
Level Database (No archive logs)  
tag nback

DO NOT RUN THIS SCRIPT NOW.

```
RMAN> CREATE SCRIPT nightback {  
    2> allocate channel db01D type disk;  
    3> backup format '$HOME/BACKUP/RMAN/%d%s%p'  
    4> (database);  
    5> release channel db01D;  
    6> }
```

created script nightback

14. Use the PRINT command to query the recovery catalog and verify the script creation.

```
RMAN> PRINT SCRIPT nightback;  
printing stored script: nightback  
{  
allocate channel db01D type disk;  
backup format '$HOME/BACKUP/RMAN/%d%s%p'  
(database);  
release channel db01D;  
}
```

## Practice 18 Solutions

1. Invoke the Export utility to export the EMPLOYEES and DEPARTMENTS tables in the HR schema.

```
$ exp hr/hr file=$HOME/BACKUP/export.dmp  
tables=employees,departments
```

```
Export: Release 9.0.0.0.0 - Beta on Thu Mar 29 12:48:25 2001  
(c) Copyright 2001 Oracle Corporation. All rights reserved.  
Connected to: Oracle9i Enterprise Edition Release 9.0.0.0.0  
With the Partitioning option
```

```
JServer Release 9.0.0.0.0 - Beta
```

```
Export done in US7ASCII character set and AL16UTF16 NCHAR  
character set
```

```
server uses WE8ISO8859P1 character set (possible charset  
conversion)
```

```
About to export specified tables via Conventional Path ...
```

```
. . exporting table          EMPLOYEES          107 rows  
exported
```

```
EXP-00091: Exporting questionable statistics.
```

```
. . exporting table          DEPARTMENTS        27 rows  
exported
```

```
EXP-00091: Exporting questionable statistics.
```

```
Export terminated successfully without warnings.
```

2. Start SQL\*Plus and connect as SYSDBA. Run the catexp.sql script from \$HOME/STUDENT/LABS.

```
sqlplus /nolog
```

```
SQL>connect / as sysdba
```

```
SQL> @$HOME/STUDENT/LABS/catexp.sql
```

```
Connect as HR and drop the EMPLOYEES and DEPARTMENTS tables.
```

```
SQL>connect hr/hr
```

```
SQL> drop table employees cascade constraints;
```

```
Table dropped.
```

```
SQL> drop table departments cascade constraints;
```

```
Table dropped.
```



## Practice 18 Solutions (continued)

3. Restore the EMPLOYEES and DEPARTMENTS tables by using the import utility.

```
$ imp hr/hr file=$HOME/BACKUP/export.dmp
TABLES=employees,departments
Import: Release 9.0.0.0.0 - Beta on Thu Mar 29 20:57:30
Connected to: Oracle9i Enterprise Edition Release 9.0.0.0.0
With the Partitioning option
JServer Release 9.0.0.0.0 - Beta
Export file created by EXPORT:V09.00 via conventional path
Import in US7ASCII and AL16UTF16 NCHAR character set
import server uses WE8ISO8859P1 character set
importing SYSTEM's objects into SYSTEM
. . importing table                "EMPLOYEES"        107 rows
imported
. . importing table                "DEPARTMENTS"       27 rows
imported
About to enable constraints...
Import terminated successfully without warnings
```

4. Query the EMPLOYEES and DEPARTMENTS tables to determine the number of rows in each of those tables.

```
SQL> select count(*) from employees;
COUNT(*)
-----
      107

SQL> select count(*) from departments;
COUNT(*)
-----
       27
```

## Practice 19 Solutions

Use the account HR for all the questions in this practice. Examine the files case1.ctl, case2.ctl and case2.dat to become familiar with the control and data file formats. As user HR, perform the following steps and get acquainted with using SQL\*Loader.

1. Create table DEPARTMENTS2 as a copy of the DEPARTMENTS table.

```
$ sqlplus hr/hr
SQL*Plus: Release 9.0.0.0.0 - Beta on Wed Mar 28 22:11:07
2001
(c) Copyright 2001 Oracle Corporation. All rights reserved.
Connected to:
Oracle9i Enterprise Edition Release 9.0.0.0.0 - Beta
With the Partitioning option
JServer Release 9.0.0.0.0 - Beta
SQL> create table departments2
2 As select * from departments;
Table created.
```

2. Delete all the records in the DEPARTMENTS2 table.

```
SQL> truncate table departments2;
Table truncated.
```

3. Run SQL\*Loader to load data into the DEPARTMENTS2 table using the control file case1.ctl located in your LABS directory. Examine the log file generated, and query the DEPARTMENTS2 table to check that all the data loaded properly.

```
$ cd $HOME/STUDENT/LABS
$ sqlldr hr/hr control=case1.ctl log=$HOME/case1.log
SQL*Loader: Release 9.0.0.0.0 - Beta on Wed Mar 28 22:24:15
(c) Copyright 2001 Oracle Corporation. All rights reserved.
Commit point reached - logical record count 27
SQL> select * from departments2;
DEPARTMENT_ID DEPARTMENT_NAME      MANAGER_ID LOCATION_ID
-----
20 Marketing                201          1800
30 Purchasing                14           1700
40 Human Resources          203          2400
50 Shipping                 121          1500
60 IT                       103          1400
70 Public Relations         204          2700
80 Sales                    145          2500
...
27 rows selected.
```

## Practice 19 Solutions (continued)

4. Delete all the records in the DEPARTMENTS2 table.

```
SQL> truncate table departments2;
```

Table truncated.

5. Run SQL\*Loader in direct-path mode to load data into the DEPARTMENTS2 table using the control file case2.ctl. Notice that this run uses an input data file to load data. Examine the log file generated and query the DEPARTMENTS2 table to check the data loaded.

- a. Run the following commands:

```
$ cd $HOME/STUDENTS/LABS
```

```
$ sqlldr hr/hr control=case2.ctl direct=true
```

```
log=$HOME/case2.log
```

```
SQL*Loader: Release 9.0.0.0.0 - Beta on Wed Mar 28  
22:39:24
```

```
(c) Copyright 2001 Oracle Corporation.
```

```
Commit point reached - logical record count 27
```

```
SQL> select * from departments2;
```

| DEPARTMENT_ID | DEPARTMENT_NAME  | MANAGER_ID | LOCATION_ID |
|---------------|------------------|------------|-------------|
| 20            | Marketing        | 201        | 1800        |
| 30            | Purchasing       | 14         | 1700        |
| 40            | Human Resources  | 203        | 2400        |
| 50            | Shipping         | 121        | 1500        |
| 60            | IT               | 103        | 1400        |
| 70            | Public Relations | 204        | 2700        |
| 80            | Sales            | 145        | 2500        |
| ...           |                  |            |             |

27 rows selected.

- b. Inspect the log file.

```
$ view $HOME/ulcase2.log
```

```
Control File: case2.ctl
```

```
Data File: case2.dat
```

```
...
```

```
Continuation: none specified
```

```
Path used: Direct
```

```
Table DEPARTMENTS2, loaded from every logical record.
```

```
...
```

## Practice 19 Solutions (continued)

Table DEPARTMENTS2:

27 Rows successfully loaded.

0 Rows not loaded due to data errors.

0 Rows not loaded because all WHEN clauses were failed.

...

Total logical records read: 27

Total logical records rejected: 0

Total logical records discarded: 0

6. Create a table EMPLOYEES2 as a copy of the EMPLOYEES table. When complete, truncate EMPLOYEES2, then restore the data with a direct-load insert from the EMPLOYEES table.

- a. Connect as HR and create table EMPLOYEES2.

```
$ sqlplus hr/hr
```

```
SQL> create table employees2
```

```
2 as select * from employees;
```

```
Table created.
```

- b. Truncate table EMPLOYEES2.

```
SQL> truncate table employees2;
```

```
Table truncated.
```

```
SQL> select * from employees2;
```

```
no rows selected
```

- c. Perform a direct-load insert into EMPLOYEES2 from EMPLOYEES and query EMPLOYEES2 to verify the load.

```
SQL> insert /*+ append */ into employees2
```

```
2 nologging
```

```
3 select * from employees;
```

```
107 rows created.
```

```
SQL> commit
```

```
SQL> select employee_id, first_name, last_name from  
employees2;
```

| EMPLOYEE_ID | FIRST_NAME | LAST_NAME |
|-------------|------------|-----------|
|-------------|------------|-----------|

|     |        |        |
|-----|--------|--------|
| 139 | John   | Seo    |
| 140 | Joshua | Patel  |
| 141 | Trenna | Rajs   |
| 142 | Curtis | Davies |

...

```
107 rows selected.
```

## Practice 19 Solutions (continued)

7. Truncate EMPLOYEES2 once again, then restore the data with a parallel direct-load insert from the EMPLOYEES table. Specify a degree of parallelism of two. Verify the data when finished.

- a. Truncate the EMPLOYEES2 table and then query it to be certain it contains no data.

```
SQL> truncate table employees2;
```

```
Table truncated.
```

```
SQL> select * from employees2;
```

```
no rows selected
```

- b. Enable parallel DML and execute the parallel direct-load insert into EMPLOYEES2 from the EMPLOYEES table. Don't forget to commit when finished. Query EMPLOYEES2 to verify the load.

```
SQL> alter session enable parallel dml;
```

```
Session altered.
```

```
SQL> insert /*+parallel(employees2,2) */
```

```
2  into employees2 nologging
```

```
3  select * from employees;
```

```
107 rows created.
```

```
SQL> commit;
```

```
Commit complete.
```

```
SQL> select employee_id, first_name, last_name from  
employees2;
```

```
EMPLOYEE_ID FIRST_NAME          LAST_NAME
```

```
-----
```

```
139          John              Seo
```

```
140          Joshua            Patel
```

```
141          Trena             Rajs
```

```
...
```

```
107 rows selected.
```



---

# B

---

## Workshop Scenarios

## **Network Workshop Scenarios**

Scenario 1: Bad port in TNSNAMES.ORA

Scenario 2: Incorrect "CONNECT\_DATA" punctuation in TNSNAMES.ORA.

Scenario 3: PROTOCOL error in TNSNAMES.ORA

Scenario 4: SERVICE\_NAME error in TNSNAMES.ORA

Scenario 5: Incorrect NAMES.DIRECTORY\_PATH value in SQLNET.ORA

Scenario 6: PORT error in the listener definition in LISTENER.ORA

Scenario 7: LISTENER error in SID\_LIST in LISTENER.ORA

Scenario 8: Listener name error in LISTENER.ORA



## Scenario 1: “Broken” port in TNSNAMES.ORA

### Solution Outline

This script “breaks” the port by replacing the original port # with 17DD in TNSNAMES.ORA. Replace :17DD” with the originally assigned port number.

```
U01.us.oracle.com =  
  (DESCRIPTION =  
    (ADDRESS_LIST =  
      (ADDRESS = (PROTOCOL = TCP)(HOST = stc-  
sun02.us.oracle.com)(PORT = 17DD))  
    )  
  )  
  . . .
```

## Scenario 2: Incorrect “CONNECT\_DATA” punctuation in TNSNAMES.ORA

### Solution Outline

This scenario “breaks” the punctuation in TNSNAMES.ORA. It removes the left parenthesis preceding “CONNECT\_DATA.” Replace the missing left parenthesis.

```
(ADDRESS_LIST =  
  (ADDRESS = (PROTOCOL = TCP)(HOST = stc-sun02)(PORT =  
    1701))  
  )  
  CONNECT_DATA =      {Replace the left parenthesis: (CONNECT_DATA }  
  (SERVICE_NAME = U01.us.oracle.com)  
  . . .
```

### Scenario 3: PROTOCOL error in TNSNAMES.ORA

#### Solution Outline

(netbrk\_3.sh) This scenario introduces a PROTOCOL error (removes TCP) in tnsnames.ora. Replace TCP in the PROTOCOL parameter definition.

```
U01.us.oracle.com =  
  (DESCRIPTION =  
    (ADDRESS_LIST =  
      (ADDRESS = (PROTOCOL = )(HOST = stc-sun02)(PORT = 1701))  
      Should be: (PROTOCOL = TCP)
```

...

## Scenario 4: SERVICE\_NAME error in TNSNAMES.ORA

### Solution Outline

This scenario comments out the SERVICE\_NAME line in TNSNAMES.ORA.  
Uncomment the line.

```
U01.us.oracle.com =  
  (DESCRIPTION =  
    (ADDRESS_LIST =  
      (ADDRESS = (PROTOCOL = )(HOST = stc-sun02)(PORT = 1701))  
    )  
    (CONNECT_DATA =  
      #      (SERVVICE_NAME = U01.us.oracle.com)  
    )  
  )  
...
```

## Scenario 5: Incorrect NAMES.DIRECTORY\_PATH value in SQLNET.ORA

### Solution Outline

This scenario changes NAMES.DIRECTORY\_PATH from TNSNAMES to NAMES in SQLNET.ORA. Replace (NAMES) with (TNSNAMES).

```
NAMES.DEFAULT_DOMAIN = us.oracle.com
```

```
NAMES.DIRECTORY_PATH= ( NAMES )
```

*Should be (TNSNAMES)*

...

## Scenario 6: PORT error in the listener definition in LISTENER.ORA

### Solution Outline

This scenario misspells PORT in the listener definition in LISTENER.ORA. Change to “PORT”.

```
LISTENER01=
  (DESCRIPTION =
    (ADDRESS = (PROTOCOL = TCP)(HOST = stc-sun02)(PRT = 1701))
    Should be (PORT =
  . . .
```

## Scenario 7: LISTENER error in SID\_LIST in LISTENER.ORA

### Solution Outline

This scenario misspells “LISTENER” in SID\_LIST\_LISTENER in LISTENER.ORA.  
Replace the “T” in “LISENER”

```
SID_LIST_LISENER01 =  
  (SID_LIST =  
    (SID_DESC =  
      (GLOBAL_DBNAME = U01.us.oracle.com)  
    . . .
```

## Scenario 8: Listener name error in LISTENER.ORA

### Solution Outline

This scenario changes the name of the listener to LISTENERX in LISTENER.ORA.  
Change it to your listener name.

**LISTENERX=**

(DESCRIPTION =

(ADDRESS = (PROTOCOL = TCP)(HOST = stc-sun02)(PORT = 1701))

)



## **Backup and Recovery Workshop Scenarios**

Scenario 1: Loss of INACTIVE Online Redo Log Group

Scenario 2: Loss of CURRENT Online Redo Log Group

Scenario 3: Loss of Control Files

Scenario 4: Loss of Media

Scenario 5: Loss of an Online Undo Segment Datafile (Open or Closed Database)

Scenario 6: Loss of a System Tablespace Datafile

Scenario 7: Loss of a Non-System, Non-Rollback Segment Datafile

Scenario 8: Recover from User Errors

Scenario 9: Failure During Online Backup

Scenario 10: Missing Data File

Scenario 11: Loss of a Datafile and Missing Archive Log File

Scenario 12: Loss of Non-Essential Datafile When Database Is Down

Scenario 13: Recover a Lost Datafile with No Backup

Scenario 14: Missing Mirrored Online Redo Log Files

Scenario 15: Loss of a Control File and Read-Only Tablespace

## Scenario 1: Loss of INACTIVE Online Redo Log Group

### Solution Outline

1. Shut down the instance.
2. Mount the database.
3. Check the V\$LOG view to determine if the file has been archived.
4. Check V\$DATAFILE to determine if there is an offline datafile that requires the unarchived log to bring it online. Issue the ALTER DATABASE CLEAR LOGFILE command; the keywords UNRECOVERABLE DATAFILE are required. The datafile and its entire tablespace must be dropped from the database because the redo necessary to bring it online is being cleared, and there is no copy of it.
5. Add a new redo log group by using the information noted on the Database Configuration Checklist as follows:

```
ALTER DATABASE ADD LOGFILE GROUP 3  
'$HOME/ORADATA/u03/log03a.rdo' SIZE 2M
```

6. Drop the damaged redo log file group:

```
SQL>ALTER DATABASE DROP LOGFILE GROUP n ;  
where n is the appropriate group number
```

7. Add a member to the log file group 3:

```
ALTER DATABASE ADD LOGFILE MEMBER  
'$HOME/ORADATA/u04/log03b.rdo' TO GROUP 3;
```

8. Determine if a full offline backup is required, and perform one if necessary.
9. Ensure that the instance is started and that the database is open.
10. Run the moreemphist.sql script.

## Scenario 2: Loss of CURRENT Online Redo Log Group

### Solution Outline

1. Start the instance if necessary.
2. Attempt to alter the database and drop the redo log group. You will receive an error stating that you cannot drop the current redo log.
3. Shut down the instance.
4. Review the `alert.log` file and any relevant trace files.
5. Copy all of the datafiles and the missing redo log files from the backup directory into their respective `un` directories.
6. Mount the database.
7. Query the `V$LOG` view to determine the `sequence#` of the CURRENT log group.
8. Issue the `ARCHIVE LOG LIST` command.
9. Recover the database using the `CANCEL` option. Cancel when the current log is suggested.
10. When recovery is complete, open the database with the `RESETLOGS` option.
11. View the `alert.log` file for the recovery that was applied.
12. Determine if a full offline backup is required and perform one if necessary. Remove the archived redo log files from the `ARCHIVE1` and `ARCHIVE2` directories.
13. Remove the `alert.log` and trace files from the `$HOME/ADMIN/BDUMP` directory.
14. Ensure that the instance is started and the database is open.
15. Run the `moreemphist.sql` script.

### **Scenario 3: Loss of Control Files**

#### **Solution Outline**

1. Start the instance if necessary.
2. Shut down the instance if the start failed.
3. Log in to SQL\*Plus and start the instance in Nomount mode.
4. Run the trace file script to recreate the control file.
5. Determine if a full offline backup is required and perform one if necessary.
6. Ensure that the instance is started and the database is open.
7. Run the `moreemphist.sql` script.

## Scenario 4: Loss of Media

### Solution Outline

1. Mount the database.
2. Determine which files to recover using V\$RECOVER\_FILE and V\$DATAFILE.
3. Use the ALTER DATABASE DATAFILE OFFLINE command to take the datafiles offline so you can open the database.
4. Once the database is open, use the ALTER TABLESPACE <tablespace\_name> OFFLINE IMMEDIATE command. Restore missing files from to another available device (un directory).
5. Rename the files so the changes are recorded in the control file.
6. Issue the RECOVER DATAFILE command to recover each individual data file, or if all files of a tablespace are involved, then use the command:  

```
'RECOVER TABLESPACE <tablespace_name>'
```

  
to recover all datafiles for a specific tablespace.
7. When the files have been recovered, bring the tablespace(s) online.
8. Query the V\$DATAFILE view to check the status of the files.
9. Query the V\$RECOVER\_FILE view to check the status of damaged files.
10. From the \$HOME directory, create the subdirectory that was removed as follows:  

```
mkdir $HOME/ORADATA/un
```

  
Also remember to set the correct privileges for Oracle to write to that directory by issuing  

```
chmod 775 $HOME/DATA/un
```
11. Take the tablespaces offline and make a physical copy of the datafiles to their original location.
12. Use the ALTER DATABASE RENAME FILE command to record the structural change in the control file.
13. Bring the tablespaces online.
14. Ensure that the instance is started and the database is open.
15. Determine if a full offline backup is required and perform one if necessary.

For more information, see the following publications:

- *Oracle9i Server Administrator's Guide*
- *Oracle9i SQL Reference Manual*
- Bulletin 1012943.6 in Appendix C

## **Scenario 5: Loss of File Containing Online Undo Segment**

### **Solution Outline**

1. Start the instance.
2. Shut down the instance if you receive any errors.
3. Restore the datafile(s).
4. Use the `Mount` option to mount the database.
5. Perform database recovery until you receive the message, “Media recovery complete.”
6. Determine if a full offline backup is required and perform one if necessary.
7. Open the database.
8. Ensure that the instance is started and the database is open.
9. Run the `moreemphist.sql` script.

### **Solution Outline for Rollback Segments**

1. Start the instance. Review the `alert.log` and any trace files if you receive errors.
2. Reference the bulletins listed below for resolving rollback segment data file recovery.
3. Determine if a full offline backup is required and perform one if necessary.
4. Ensure that the instance is started and the database is open.
5. Run the `moreemphist.sql` script.

For more information, see the following publications:

- Bulletins 1013221.6 and 1010700.6 in Appendix C

## **Scenario 6: Loss of a Datafile of System Tablespace**

### **Solution Outline**

1. Start the instance.
2. Shut down the instance if you receive any errors.
3. Restore the datafile belonging to the SYSTEM tablespace.
4. Use the `Mount` option to mount the database.
5. Perform database recovery until you receive the message, "Media recovery complete."
6. Determine if a full offline backup is required and perform one if necessary.
7. Open the database.
8. Ensure that the instance is started and the database is open.
9. Run the `moreemphist.sql` script.

## **Scenario 7: Loss of a Non-System, Non-Rollback Segment Datafile**

### **Solution Outline**

1. Start the instance if necessary.
2. Query `V$RECOVER_FILE`.
3. Take the missing datafiles offline.
4. Open the database.
5. Restore the datafiles.
6. Perform recovery of the datafiles.
7. Bring the datafiles online.
8. Run the `moreemphist.sql` script.



## Scenario 8: Recover from User Errors

### Solution Outline

1. Three recovery scenarios pertain to this failure. Remember, however, that the objective is to minimize downtime and data loss when determining which recovery method to use.
  - a. You may restore the entire database using a point-in-time recovery, which means you will lose any transactions that occurred after the time to which you recover.
  - b. You may restore the database at another location, export the table, then import the individual table into the primary database.
  - c. Restore the table from an export file.
2. Ensure that the instance is started and the database is open.
3. Run the `moreemphist.sql` script.
4. Determine if a full offline backup is required and perform one if necessary.

## Scenario 9: Failure During Online Backup

### Solution Outline

1. Mount the database.
2. Query the view V\$RECOVER\_FILE.
3. Query the view V\$BACKUP.
4. Determine which files were in backup mode when the database crashed.
5. Take the datafile out of backup mode  
`ALTER DATABASE DATAFILE 'file_name' END BACKUP;`  
or to recover more quickly you may simply issue:  
`ALTER DATABASE END BACKUP;`
6. Open the database.
7. Query the V\$RECOVER\_FILE, V\$BACKUP, and V\$DATAFILE views.
8. Determine if a full offline backup is required and perform one if necessary.
9. Ensure that the instance is started and the database is open.
10. Run the `moreemphist.sql` script.

## Scenario 10: Missing Datafile

### Solution Outline

- Perform recovery in accordance with the attached bulletin.
- Determine if a full offline backup is required and perform one if necessary.
- Ensure that the instance is started and the database is open.
- Run the `moreemp.sql` script.

For more information, see the following publications:

- *Oracle9i Server Administrator's Guide*
- *Oracle9i SQL Reference Manual*
- Bulletin 1005254.6 in Appendix C

## **Scenario 11: Loss of a Datafile and Missing Archived Log File**

### **Solution Outline**

1. Restore the datafile to the correct directory.
2. Start the instance and mount the database.
3. Begin database recovery. You will discover that you are missing an archived log file.
4. Shut down the instance.
5. Determine which archived redo log file you are missing. Check V\$RECOVERY\_LOG for the archival information and check the LOG\_ARCHIVE\_DEST\_ *n* directories.
6. Restore all of your datafiles from the \$HOME/BACKUP directory. Be sure to keep your current control files.
7. Perform a cancel-based recovery, canceling the operation at the appropriate point.
8. Open the database using the RESETLOGS option.
9. Perform a full offline backup and remove the archived redo log files that are no longer needed..
10. Ensure that the instance is started and the database is open.
11. Run the `moreemphist.sql` script.

## Scenario 12: Loss of Non-Essential Datafile When Database Is Down

### Solution Outline

1. Start the instance if necessary.
2. Shut down the instance if you receive errors.
3. Mount the database.
4. Open the database.
5. Query the `V$RECOVER_FILE` view and note the filename.
6. Alter the database to take the file offline and drop it.
7. Open the database.
8. Drop the `INDX` tablespace.
9. Create the `INDX` tablespace using the same file name noted in step 5, using a size of 500K and the `reuse` option.
10. Run the `$HOME/STUDENT/LABS/index.sql` script.
11. Determine if a full offline backup is required and perform one if necessary.
12. Ensure that the instance is started and that the database is open.
13. Run the `moreemphist.sql` script.

## Scenario 13: Recover a Lost Datafile With No Backup

### Solution Outline

1. Use the Mount command to mount the database.
2. Query the V\$RECOVER\_FILE view.
3. Query the V\$DATAFILE view.
4. Alter the database and create the new datafile as a new filespec. The filename is \$HOME/ORADATA/u06/new01.dbf, and the size is 500K.
5. Recover the datafile.
6. Open the database.
7. Query the V\$RECOVER\_FILE view.
8. Query the V\$DATAFILE view.
9. Determine if a full offline backup is required and perform one if necessary.
10. Ensure that the instance is started and that the database is open.
11. Verify that the HR.NEW\_EMPHIST table exists.

## Scenario 14: Missing Mirrored Online Redo Log Files

### Solution Outline

1. Start the instance if necessary.
2. Review the `alert.log` file and trace files for abnormal conditions.
3. Query the `V$LOGFILE` view.
4. Switch the log files and then query the `V$LOGFILE` view. Only the mirrored redo log files are corrupt.
5. Correct the problem by adding new redo log files using the naming conventions on the Database Configuration Checklist.
6. Determine if a full offline backup is required and perform one if necessary.
7. Ensure that the instance is started and that the database is open.
8. Run the `moreemphist.sql` script.

## Scenario 15: Loss of a Control File and Read-Only Tablespace

### Solution Outline

1. Start the instance if necessary.
2. Shut down the instance if you receive errors.
3. View the `alert.log` file.
4. Restore the missing datafile and control files from your backup.
6. Mount the database.
7. Recover the database using the `BACKUP CONTROLFILE` option. (You may need to apply one of the online redo log files.)
8. Open the database using the `RESETLOGS` option.
9. Put the `QUERY_DATA` tablespace into `READ ONLY` mode.
10. Determine if a full offline backup is required and perform one if necessary.
11. Ensure that the instance is started and that the database is open.
12. Run the `moreemphist.sql` script.





# **Worldwide Support Bulletins**

ORACLE

Copyright © Oracle Corporation, 2001. All rights reserved.

## Oracle Corporate Support Problem Repository

### Missing Data File

1. Prob# 1005254.6 CANNOT STARTUP THE DATABASE BECAUSE A DATAFILE WAS REMO
2. Soln# 2031159.6 WORKAROUNDS TO OPEN UP THE DATABASE WITH A REMOVED DATA
3. Prob# 1005254.6 CANNOT STARTUP THE DATABASE BECAUSE A DATAFILE WAS REMO

**Summary** Cannot start up the database because a data file was removed from the file directory

**Problem Description** If a file is physically removed from the operating system directory, you may either get errors while the database is up or while the DBA is trying to start up the database.

**Problem Explanation** The Oracle Server verifies the existence and consistency of each data file registered in the control file after the database is successfully mounted. If the file is bad or is being unintentionally removed, the following error may occur:

```
ORA-01157: cannot identify data file %n - file not found
ORA-01110: data file %n: '%s'
```

The following errors may also occur when trying to do a normal shutdown or when DBWR attempts to write to the file that is being removed:

```
ORA-01116: error in opening database file %n
ORA-01110: data file %n: '%s'
ORA-07368: sfofi: open error, unable to open database file.
```

These errors are followed by an operating system specific error (for instance, error number 2 in most UNIX platforms).

### Diagnostics and References

```
* {1942.6,Y,100}   ORA-1110 DATA FILE 2: '/.../RBSDSV1.DBF"
* {2170.6,Y,100}   ORA-1116 ERROR IN OPENING DATABASE FILE 2
* {4303.6,Y,100}   ORA-11157
* {6179.6,Y,100}
```

LOST A DATAFILE

Soln# 2031159.6 WORKAROUNDS TO OPEN UP THE DATABASE WITH A REMOVED DATA

Solution ID: 2031159.6  
For Problem: 1005254.6  
Affected Platforms: Generic: not platform specific  
Affected Products: Oracle7 Server  
Affected Components: RDBMS V07.XX  
Affected Oracle Vsn: V07.XX

**Summary** Workarounds to open up the database with a removed data file

**Solution Description** *Warning: This solution can only be applied if the removed data file does NOT belong to the system tablespace or to a rollback tablespace. If the file belongs to the system tablespace or to a rollback tablespace, please contact Oracle customer support.*

There are cases in which the DBA inadvertently removes a data file from the file directory, maybe with the incorrect assumption that by removing the file, any reference to it from the Oracle Server is also removed. It may also be due to the fact that an operating system error or hardware problem rendered the file unreadable or inaccessible.

If the file is inaccessible by the Oracle Server, the DBWR may force the data file to go offline, in which case you would get the following error when trying to access the data file by any means:

```
ORA-01135, 00000, "file %s accessed for DML/query is offline"  
// *Cause: Attempted to access a data file that is offline  
// *Action: Bring the data file back online
```

In either case, the easiest way is to drop the entire tablespace that contains the data file. The steps to be executed from within SQL\*DBA are:

1. STARTUP MOUNT
2. For each deleted data file, issue the command  
ALTER DATABASE DATAFILE 'full path of filename' OFFLINE  
[DROP];

**Note:** You must use the DROP option if the database is in NOARCHIVELOG mode, because you cannot recover this file if you apply incomplete media recovery on it via the command ALTER DATABASE OPEN RESETLOGS. See the *SQL Reference Manual* for details.

3. ALTER DATABASE OPEN;
4. DROP TABLESPACE <tablespace> INCLUDING CONTENTS [CASCADE CONSTRAINTS];

## Data Block Corruption

1. Prob# 1010640.6 DATA BLOCK CORRUPTION BULLETIN
2. Soln# 2058665.6 POSSIBLE WORKAROUNDS FOR ORA-1578 - BULLETIN 108491.543

Prob# 1010640.6 DATA BLOCK CORRUPTION BULLETIN

Problem ID: 1010640.6  
Affected Platforms: Generic: not platform specific  
Affected Products: Oracle7 Server  
Affected Components: RDBMS Generic  
Affected Oracle Vsn: Generic

**Summary** Data block corruption bulletin

+=+

**Problem Description** When there is a corrupt data block in the database, one of the most common errors you might receive when you try to access that corrupted block is ORA-1578. Other errors you might also receive to indicate a corruption are:

ora-600 [3339]

ora-600 [3398]

### Problem Explanation

```
ORA-01578, 00000, "ORACLE data block corrupted (file # %s,
block # %s)"
// *Cause: The data block indicated was corrupted, mostly
due to software // errors.
// *Action: Try to restore the segment containing the
block indicated. This
// may involve dropping the segment and recreating it. If
there
// is a trace file, report the errors in it to your ORACLE
// representative.
```

The bulletin 108491.543 by EPITT discusses how to resolve the ora-1578, although the concepts may also be applied to the other corruption errors as well.

===+

## Diagnostics and References

\* {2003.6,Y,100} ORA-1578

\* {6085.6,Y,100} CORRUPTED DATABASE BLOCKS

Soln# 2058665.6 POSSIBLE WORKAROUNDS FOR ORA-1578 - BULLETIN 108491.543

Solution ID: 2058665.6  
For Problem: 1010640.6  
Affected Platforms: Generic: not platform specific  
Affected Products: Oracle7 Server  
Affected Components: RDBMS Generic  
Affected Oracle Vsn: Generic

**Summary** Possible workarounds for ORA-1578 - Bulletin 108491.543

+=+

See bulletin 108491.543.

+=+=+

## References

### I/O Error Reading Block

1. Prob# 1013621.6 ORA-1115 I/O ERROR READING BLOCK
2. Soln# 2061743.6 SOLVING ORA-1115

Prob# 1013621.6 ORA-1115 I/O ERROR READING BLOCK

Problem ID: 1013621.6  
Affected Platforms: Generic: not platform specific  
Affected Products: Oracle7 Server  
Affected Components: RDBMS Generic  
Affected Oracle Vsn: Generic

### Summary

ORA-1115 I/O ERROR READING BLOCK

**Problem Description** An ORA-1115 is issued whenever the Oracle server is unable to read from an open data file because of an I/O error:

```
01115, 00000, "IO error reading block from file %s (block  
# %s)"
```

```
// *Cause: Device on which the file resides is probably  
offline
```

```
// *Action: Restore access to the device
```

ORA-1115s are usually followed by:

- ORA-1110
- An operating system-level Oracle error message such as ORA-737X
- An operating system error (such as error number 5 in UNIX)

## Problem Explanation

What causes ORA-1115?

The Oracle server delivers read-from-file requests to the underlying operating system (except if raw devices are being used). A read request specifies a data file and a block number to be accessed. If a low-level I/O error prevents the read from completing successfully, the Oracle server signals an ORA-1115.

The main causes for an ORA-1115 are:

1. Hardware problems
  - Disk controller problems (the most common, and usually intermittent)
  - Disk problems (including bad blocks and disk malfunctioning)
2. Data block corruption (at the physical level)  
Usually caused by previous hardware problems.
3. Problems handling very large data files

In Oracle 7.1.4 and lower on Sun Solaris, bug 233569 causes ORA-1115 and ORA-7371 when handling data files bigger than 2GB.

Typical scenarios where ORA-1115 can happen include:

- On execution of DML statements
- During exports or imports
- At startup or shutdown

## Diagnostics and References

Soln# 2061743.6 SOLVING ORA-1115

|                      |                                |
|----------------------|--------------------------------|
| Solution ID:         | 2061743.6                      |
| For Problem:         | 1013621.6                      |
| Affected Platforms:  | Generic: not platform specific |
| Affected Products:   | Oracle7 Server                 |
| Affected Components: | RDBMS Generic                  |
| Affected Oracle Vsn: | Generic                        |

**Summary** Solving ORA-1115

+=+

**Solution Description** Because most ORA-1115s are caused by hardware problems, the solution consists in first isolating the hardware problems, and then addressing the problem at the database level, if necessary.

Performing hardware checks is essential. If hardware problems are not fixed, trying to solve the problem at the database level will be useless. Run operating system level utilities and diagnostic tools that check for the sanity of disks, controllers, and the I/O subsystem. Pay special attention to the disk where the data file referenced in the ORA-1115 resides. Your system administrator should be able to assist you in this task. Such diagnostics should be done in parallel with the steps recommended here, if feasible, or as soon as possible thereafter.

Determining the exact cause of an ORA-1115 is not always trivial. Approaches differ according to whether you know the cause of the problem or not.

### **Steps for Solving the Problem When the Cause Is Not Known**

1. Try to assess the cause and extent of the problem.

Examine the `alert.log` file for this instance. Scan the last few days' entries for other occurrences of ORA-1115. If you find them, determine the following:

- a. Do they reference files in different disks?  
If so, it is likely that there you have controller problems. Go to Scenario II.A below.
- b. Do they reference different files in the same disk?  
If so, it is likely that there are problems with that disk. Go to Scenario II. B below.
- c. Do they always reference the same data file?  
If so, it is likely that the data file contains bad blocks. Go to Scenario II.C below.  
If the file is bigger than 2GB and you are running 7.1.4 or lower on Solaris platform, see Scenario II.D below.
- d. If none of the above apply, go to step 2.

2. If the data file is in the SYSTEM tablespace, or the database is in NOARCHIVELOG mode, shut the database down. Go to step 4.

If shutdown immediate fails, do a shutdown abort.

3. If the database is in ARCHIVELOG mode, you should still shut the database down. If the database cannot be shut down, take the data file offline.

```
ALTER DATABASE DATAFILE '<full_path_file_name>' OFFLINE;
```

4. Try to copy the data file to another disk (managed by a different controller, if possible).

5. If the copy fails, even after you retry, the data file must be considered lost at this point. The next action depends on the tablespace to which the lost file belongs. See the following Solution References to PR entries, according to the different types of tablespaces, for instructions on how to proceed.

**Important:** While going through the PR entries below, keep in mind that if you restore the data file from backup, you need to place it in another disk, preferably under a different controller, and rename it inside the Oracle server (see the solution Reference to PR entry 1013480.6 for details). If you recreate any tablespace, make sure its data files are created in another disk, preferably under a different controller.

| TABLESPACE     | PR ENTRY  |
|----------------|-----------|
| -----          | -----     |
| system         | 1013182.6 |
| rollback       | 1013221.6 |
| user 1013173.6 |           |
| index          | 1013115.6 |
| temporary      | 1013104.6 |
| read-only      | 1013129.6 |

6. If the database is down, mount it.
7. Rename the data file that you succeeded in copying inside Oracle.

```
ALTER DATABASE RENAME FILE '<old_full_path_file_name>'
TO '<new_full_path_file_name>';
```
8. If the database is mounted, open it. If you took the data file offline, perform media recovery on it, and then bring it online.

```
RECOVER DATAFILE '<full_path_file_name>';
ALTER DATABASE DATAFILE '<full_path_file_name>' ONLINE;
```



## Steps for Solving the Problem When the Cause Is Known

**Controller Problems** These are typically intermittent. Usually, there is no damage to the data files. Unless you can quickly fix the controller and restore access to the data file, follow these steps:

1. Find out which data files are under the bad controller.  
Query V\$DATAFILE to obtain the names of all data files in the database. You may need the help of the system administrator to determine which data files reside in disks managed by this controller.
2. If any of the data files under the bad controller belongs to the SYSTEM tablespace, or if the database is in NOARCHIVELOG mode, shut the database down. Go to step 4.  
If shutdown immediate fails, do a shutdown abort.
3. If the database is in ARCHIVELOG mode and none of the data files under the bad controller are in the SYSTEM tablespace, you should shut the database down. If the database cannot be shut down, take all the data files under the bad controller offline.  

```
ALTER DATABASE DATAFILE '<full_path_file_name>' OFFLINE;
```
4. Try to copy all the data files under the bad controller to disks managed by different controllers.
5. If the database is down, mount it.
6. Rename all the files that you succeeded in copying inside the Oracle server.  

```
ALTER DATABASE RENAME FILE '<old_full_path_file_name>'  
TO '<new_full_path_file_name>';
```
7. If the copy fails for one or more of the data files even after you try to copy them, those data files must be considered lost at this point. See the following Solution References to PR entries, according to the tablespaces to which the lost data files belong, for instructions on how to proceed.

**Important:** While going through the PR entries below, keep in mind that if you restore data files from backup, you must place them in disks under other controllers and rename them inside the Oracle Server (see the solution Reference to PR entry 1013480.6 for details). If you recreate any tablespace, make sure its data files are created under other controllers.

| TABLESPACE | PR ENTRY  |
|------------|-----------|
| -----      | -----     |
| system     | 1013182.6 |
| rollback   | 1013221.6 |
| user       | 1013173.6 |
| index      | 1013115.6 |
| temporary  | 1013104.6 |
| read-only  | 1013129.6 |

8. If the database is mounted, open it. If any of the moved data files is offline, apply media recovery to it, and then put it online:

```
RECOVER DATAFILE '<full_path_file_name>';
ALTER DATABASE DATAFILE '<full_path_file_name>' ONLINE;
```

**Disk Problems** If a disk has bad blocks or is malfunctioning, you should focus on moving its data files to a different disk, if possible. If not, you must consider the files lost and address the issue according to the tablespaces to which they belong, while you fix the disk. The steps to follow in this scenario are like those in Scenario II.A above.

**Data Block Corruption** If you are certain that the data file has bad blocks, the data file should be considered LOST if it belongs to the SYSTEM tablespace or to a ROLLBACK or READ-ONLY tablespace. See the following Solution References to PR entries, depending on the tablespace to which the data file belongs.

**Important:** While going through the PR entries below, keep in mind that if you restore data files from backup, you must place them in different disks (preferably under other controllers) and rename them inside the Oracle server (see the solution Reference to PR entry 1013480.6 for details). If you re-create any tablespace, make sure its data files are created on different disks (preferably under other controllers).

| TABLESPACE | PR ENTRY  |
|------------|-----------|
| -----      | -----     |
| system     | 1013182.6 |
| rollback   | 1013221.6 |
| user       | 1013173.6 |
| index      | 1013115.6 |
| temporary  | 1013104.6 |
| read-only  | 1013129.6 |

If the data file belongs to a user or index tablespace, you may also address the problem as an object re-creation issue if the ORA-1115 occurs consistently against the same objects (tables, indexes, and so on.). The following query returns the object in which the bad block is:

```
SELECT SEGMENT_NAME, SEGMENT_TYPE FROM DBA_EXTENTS
WHERE FILE_ID = <file_number> and <block_number> BETWEEN
BLOCK_ID AND BLOCK_ID + BLOCKS - 1;
```

where *<file\_number>* and *<block\_number>* are those listed in the ORA-1115. If this query consistently points to a table or index, you may try re-creating them, if possible, in a different tablespace. For further details on this scenario, see the Solution Reference to PR entry 1010640.6.

**Very Large Data File Problems on Solaris** If you are running Oracle 7.1.4 or lower on a Solaris platform, and you get an ORA-7371 with the ORA-1115 and the file is bigger than 2GB, you are likely experiencing bug 233569. This bug is fixed in 7.1.6, and patches are available for 7.1.3 (bug 233569) and 7.1.4 (bug 281904).

## **Rollback Segment Needs Recovery**

Prob# 1010700.6 BULLETIN: ROLLBACK SEGMENT NEEDS RECOVERY

Problem ID: 1010700.6  
Affected Platforms: Generic: not platform specific  
Affected Products: Oracle7 Server  
Affected Components: RDBMS Generic  
Affected Oracle Vsn: Generic

**Summary** Bulletin: Rollback segment needs recovery

### **Problem Description**

Document ID: 107693.969  
Title: ROLLBACK SEGMENT NEEDS RECOVERY  
Department: RDBMS SUPPORT  
Creation Date: 13-February-1995  
Last Revision Date: 7-June-1995  
Expiration Date:  
Revision Number: 1  
Distribution Code: EXTERNAL  
Category:  
Product: GENERIC  
Product Version: GENERIC  
Platform: GENERIC  
Information Type: ADVISORY  
Impact: MEDIUM  
Abstract: This article discusses what it means when a rollback segment needs recovery and how to resolve it.  
Keywords: ROLLBACK;SEGMENT;NEEDS;RECOVERY;  
STATUS;CORRUPT

## Overview

This bulletin discusses why a rollback segment has the status of “needs recovery,” what the status means, and how to resolve it.

## Introduction

Rollback segments can be monitored through the data dictionary view, `DBA_ROLLBACK_SEGS`. There is a status column that describes what state the rollback segment is currently in. Normal states are either online or offline. Occasionally, the status of needs recovery appears.

When a rollback segment is in this state, bringing the rollback segment offline or online either through the alter rollback segment command, or by removing it from the `ROLLBACK_SEGMENTS` parameter in the `init.ora`, usually has no effect.

## Understanding

A rollback segment falls into this status of needs recovery whenever the Oracle server tries to roll back an uncommitted transaction in its transaction table and fails.

Here are some examples of why a transaction may need to rollback:

1. A user may do a DML transaction and decide to issue rollback.
2. A shutdown abort occurs, and the database must do an instance recovery, in which case, the Oracle server has to roll back all uncommitted transactions.

When a rollback of a transaction occurs, undo must be applied to the data block in which the modified rows are found. If that data block is unavailable, the undo cannot be applied. The result is a corrupted rollback segment with the status of needs recovery.

What could be some reasons a data block is inaccessible for undo?

1. If a tablespace or a data file is offline or missing
2. If the object the data block belongs to is corrupted
3. If the data block that is corrupt is in the rollback segment itself rather than the object

## How to Resolve the Needs Recovery Status

1. Verify that all tablespaces and all data files are online. This can be checked through V\$DATAFILE, under the STATUS column. For tablespaces associated with the data files, look in DBA\_TABLESPACES.

If that still does not resolve the problem, continue with the following steps.

2. Put the following in the `init.ora`- `event = "10015 trace name context forever, level 10"`.

Setting this event will generate a trace file that reveals the necessary information about the transaction that the Oracle server is trying to roll back and most importantly, what object the Oracle server is trying to apply the undo to.

3. Shut down the database (if the NORMAL mode does not work, try IMMEDIATE or ABORT) and bring it back up.

**Note:** An ora-1545 or other errors may be encountered. If the database cannot start up, contact customer support.

4. Check in the directory that is specified by the USER\_DUMP\_DEST parameter (in the `init.ora` or SHOW Parameter command) for a trace file that was generated at startup time.
5. In the trace file, there should be a message similar to error recovery tx(##) object #. TX(##) refers to transaction information. The object # is the same as the object\_id in sys.dba\_objects.
6. Use the following query to determine what object the Oracle server is trying to perform recovery on.  

```
select owner, object_name, object_type, status
from dba_objects where object_id = <object #>;
```
7. This object must be dropped so that the undo can be released. An export, or relying on a backup, may be necessary to restore the object after the corrupted rollback segment disappears.
8. After dropping the object, put the rollback segment back in the `init.ora` parameter ROLLBACK\_SEGMENTS, remove the event, and shut down and start up the database.

In most cases, the above steps will resolve the problematic rollback segment. If this still does not resolve the problem, it may be likely that the corruption is in the actual rollback segment. If the problem is not resolved, please contact customer support.

### Problem Explanation

#### Diagnostics and References

- \* {6123.6,Y,100}ROLLBACK SEGMENT NEEDS RECOVERY
- \* {6124.6,Y,100}ORA-1545 ON STARTUP

## Lost Data File in Rollback Segment

Prob# 1013221.6 RECOVERING FROM A LOST DATAFILE IN A ROLLBACK TABLESPACE

|                      |                                |
|----------------------|--------------------------------|
| Problem ID:          | 1013221.6                      |
| Affected Platforms:  | Generic: not platform specific |
| Affected Products:   | Oracle7 Server                 |
| Affected Components: | RDBMS Generic                  |
| Affected Oracle Vsn: | Generic                        |

**Summary** Recovering from a lost data file in a rollback tablespace

**Problem Description** This is a recovery scenario in which a data file in a rollback segment tablespace has been lost or damaged to a point that the Oracle server cannot recognize it anymore. Trying to start up the database will result in ORA-1157, ORA-1110, and possibly an operating system level error such as ORA-7360. Trying to shut down the database in normal or immediate mode will result in ORA-1116, ORA-1110, and possibly an operating system level error such as ORA-7368.

**Solution Description** This recovery situation requires extra caution. Please call Oracle Customer Support if you have questions or need assistance.

The main issue in solving this problem is trying to make sure that the active transactions in the rollback segments do not get lost.

**Solution Explanation** The approach depends on the specific scenario in which the loss of the rollback data file is detected.

## The Database Is Down

Attempting to start up the database will result in ORA-1157 and ORA-1110. The solution here depends on whether the database was cleanly shut down or not.

**The Database Was Cleanly Shut Down** If you are absolutely positive that the database was cleanly shut down, that is, closed with either shutdown NORMAL or IMMEDIATE, then the simplest solution is to drop the missing data file offline, open the database in restricted mode, and then drop and recreate the rollback tablespace to which the file belonged. Do not follow this procedure if the database was shut down in ABORT mode or if it crashed.

The steps are:

1. Make sure the database was last cleanly shut down.

Check the `alert.log` file for this instance. Go to the bottom of the file and make sure the last time you shut the database down you received the messages:

```
"alter database dismount
Completed: alter database dismount"
```

This also includes the case of a clean shutdown followed by a failed attempt to start up the database. In that case, the Oracle server will issue error messages and shut itself down abort. For the purposes of this solution, though, this counts as a clean shutdown.

If that is not the case, that is, if the last time you shut the database down, it was in ABORT mode, or the database crashed itself, it is *not* safe to proceed. You should follow the instructions for case I.B below.

2. Remove all the rollback segments in the tablespace to which the lost data file belongs from the `ROLLBACK_SEGMENTS` parameter in the `init.ora` file for this instance. If you are not sure about which rollbacks are in that tablespace, simply comment out the whole `ROLLBACK_SEGMENTS` entry.

3. Mount the database in restricted mode.

```
STARTUP RESTRICT MOUNT
```

4. Drop the lost data file offline.

```
ALTER DATABASE DATAFILE '<full_path_file_name>'
OFFLINE DROP;
```

5. Open the database.

```
ALTER DATABASE OPEN
```

If you receive the message "Statement processed," go to step 7.

If instead you get ORA-604, ORA-376, and ORA-1110, go to step 6.



6. Because opening the database failed, shut the database down and edit the `init.ora` file for this instance.

Comment out the `ROLLBACK_SEGMENTS` parameter and add the following line:

```
_corrupted_rollback_segments = ( <rollback1>,...,
<rollbackN> )
```

For example, the above list should contain all the rollbacks originally listed in the `ROLLBACK_SEGMENTS` parameter.

Use this parameter only in this specific scenario or as instructed by Oracle Customer Support, then start up the database in restricted mode:

```
STARTUP RESTRICT
```

7. Drop the rollback tablespace to which the data file belonged.

```
DROP TABLESPACE <tablespace_name> INCLUDING CONTENTS;
```
8. Recreate the rollback tablespace with all its rollback segments. Remember to bring the rollbacks online after you create them.
9. Make the database available to all users.

```
ALTER SYSTEM DISABLE RESTRICTED SESSION;
```
10. Reininclude the rollbacks you just recreated in the `ROLLBACK_SEGMENTS` parameter in the `init.ora` file for this instance. If you had commented out the whole `ROLLBACK_SEGMENTS` entry, simply uncomment it now. If you had to go through step 6, remove the corrupted `ROLLBACK_SEGMENTS` parameter now.

**The Database Was Not Cleanly Shut Down** This is the situation where the database was last shut down in ABORT or CRASHED mode. In this case, it is almost certain that the rollback segments that had extents in the lost data file still contain active transactions. Therefore, the file cannot be taken offline or dropped. You must restore the lost data file from a backup and apply media recovery to it. If the database is in NOARCHIVELOG mode, you will only succeed in recovering the data file if the redo to be applied is within the range of your online logs. If a backup of the data file is not available, please contact Oracle Customer Support.

These are the steps:

1. Restore the lost file from a backup.
2. Mount the database.
3. Issue the following query:

```
SELECT FILE#, NAME, STATUS FROM V$DATAFILE;
```

If the status of the file you just restored is OFFLINE, you must take it online before proceeding:

```
ALTER DATABASE DATAFILE '<full_path_file_name>' ONLINE;
```

4. Issue the following query:

```
SELECT V1.GROUP#, MEMBER, SEQUENCE#, FIRST_CHANGE#  
FROM V$LOG V1, V$LOGFILE V2  
WHERE V1.GROUP# = V2.GROUP# ;
```

This will list all your online redo log files and their respective sequence and first change numbers.

5. If the database is in NOARCHIVELOG mode, issue the query:

```
SELECT FILE#, CHANGE# FROM V$RECOVER_FILE;
```

If the CHANGE# is greater than the minimum FIRST\_CHANGE# of your logs, the data file can be recovered. Remember that all the logs to be applied will be online logs, and go to step 6.

If the CHANGE# is lesser than the minimum FIRST\_CHANGE# of your logs, the file cannot be recovered. Your options at this point include restoring a full backup if one is available or forcing the database to open in an inconsistent state to get a full export out of it. For further details and to assist you in your decision, please contact Oracle Customer Support.

6. Recover the data file:

```
RECOVER DATAFILE '<full_path_file_name>'
```

7. Confirm each of the logs that you are prompted for until you receive the message “Media recovery complete.” If you are prompted for a non-existing archived log, the Oracle server probably needs one or more of the online logs to proceed with the recovery. Compare the sequence number referenced in the ORA-280 message with the sequence numbers of your online logs. Then enter the full path name of one of the members of the redo group whose sequence number matches the one you are being asked for. Continue to enter online logs as requested until you receive the message, “Media recovery complete.”
8. Open the database.

## **The Database Is Up**

If you have detected the loss of the rollback data file and the database is still up and running, do not shut it down. In most cases, it is simpler to solve this problem with the database up than with it down.

Two approaches are possible in this scenario:

1. The first one involves taking the lost data file offline, restoring it from backup, and then applying media recovery to it to make it consistent with the rest of the database. This method can only be used if the database is in ARCHIVELOG mode.
2. The other approach involves taking offline all the rollback segments in the tablespace to which the lost data file belongs, dropping the tablespace, and then recreating it. You may need to kill sessions that have transactions in the rollbacks involved to force the rollbacks to go offline.

In general, the first approach is simpler to apply. It will also be faster if the data file and the necessary archived logs can be quickly restored from backup. However, more user transactions will error out and be rolled back than with the second approach. Because of read-consistency, queries against certain tables may fail with the first approach, because the rollback extents from which the Oracle server would retrieve the data may be in the offlined data file.

**Approach A: Restoring the Data File from Backup** As mentioned before, this approach can only be followed if the database is in ARCHIVELOG mode. Here are the steps:

1. Take the lost data file offline.

```
ALTER DATABASE DATAFILE '<full_path_file_name>' OFFLINE;
```

**Note:** Depending on the current amount of database activity, you may need to create additional rollback segments in a different tablespace to keep the database going while you take care of the problem.

2. Restore the data file from a backup.
3. Issue the following query:

```
SELECT V1.GROUP#, MEMBER, SEQUENCE#  
FROM V$LOG V1, V$LOGFILE V2  
WHERE V1.GROUP# = V2.GROUP# ;
```

This will list all your online redo log files and their respective sequence numbers.

4. Recover the data file:

```
RECOVER DATAFILE '<full_path_file_name>'
```

5. Confirm each of the logs that you are prompted for until you receive the message, "Media recovery complete." If you are prompted for a non-existing archived log, the Oracle server probably needs one or more of the online logs to proceed with the recovery. Compare the sequence number referenced in the ORA-280 message with the sequence numbers of your online logs. Then enter the full path name of one of the members of the redo group whose sequence number matches the one you are being asked for. Continue to enter online logs as requested until you receive the message "Media recovery complete."
6. Bring the data file back online.

```
ALTER DATABASE DATAFILE '<full_path_file_name>' ONLINE;
```

**Approach B: Re-creating the Rollback Tablespace** This approach can be used regardless of the archival mode of the database. The steps are:

1. Try to take offline all of the rollback segments in the tablespace to which the lost data file belongs.

```
ALTER ROLLBACK SEGMENT <rollback_segment> OFFLINE;
```

Repeat this statement for all rollbacks in the tablespace.

**Note:** Depending on the current amount of database activity, you may need to create additional rollback segments in a different tablespace to keep the database going while you take care of the problem.

2. Check the status of the rollbacks.

They must all be offline before they can be dropped. Issue the query:

```
SELECT SEGMENT_NAME, STATUS FROM DBA_ROLLBACK_SEGS
WHERE TABLESPACE_NAME = '<TABLESPACE_NAME>';
```

3. Drop all offlined rollback segments.

For each rollback returned by the query in step 2 with status OFFLINE, issue the statement:

```
DROP ROLLBACK SEGMENT <rollback_segment>;
```

4. Handle the rollbacks that remain online.

Repeat the query in step 2.

If any of the rollbacks you tried to offline still has an ONLINE status, it means there are still active transactions in it. You may confirm that by issuing the query:

```
SELECT SEGMENT_NAME, XACTS_ACTIVE_TX, V.STATUS
FROM V$ROLLSTAT V, DBA_ROLLBACK_SEGS
WHERE TABLESPACE_NAME = '<TABLESPACE_NAME>' AND
SEGMENT_ID = USN;
```

If the above query returns no rows, it means all the rollbacks in the affected tablespace are already offline. Repeat the query in step 2 to retrieve the names of the rollbacks that just became offline and then drop them as described in step 3.

If the above query returns one or more rows, they should show the status of PENDING OFFLINE.

Next, check the ACTIVE\_TX column for each rollback. A value of 0 implies that there are no pending transactions left in the rollback, and it should go offline shortly. Repeat the query in step 2 a few more times until it shows the rollback being offline, and then drop it as described in step 3. Go to step 6.

If any of the PENDING OFFLINE rollbacks has a value of 1 or greater in the ACTIVE\_TX column, go to step 5.

5. Force rollbacks with active transactions to go offline.  
At this point, the only way to move forward is to have the PENDING OFFLINE rollbacks released. The active transactions in these rollbacks must either be committed or rolled back. The following query shows which users have transactions assigned to which rollbacks:

```
SELECT S.SID, S.SERIAL#, S.USERNAME, R.NAME "ROLLBACK"  
FROM V$SESSION S, V$TRANSACTION T, V$ROLLNAME R  
WHERE R.NAME IN ( '<PENDING_ROLLBACK_1>', ... ,  
                  '<PENDING_ROLLBACK_N>' )  
AND S.TADDR = T.ADDR AND T.XIDUSN = R.USN;
```

You may directly contact the users with transactions in the PENDING OFFLINE rollbacks and ask them to commit (preferably) or rollback immediately. If that is not feasible, you can force that to happen by killing their sessions. For each of the entries returned by the above query, issue the statement:

```
ALTER SYSTEM KILL SESSION '<SID>, <SERIAL#>';
```

Where *<SID>* and *<SERIAL#>* are those returned by the previous query. After the sessions are killed, it may take a few minutes before the Oracle server finishes rolling back and doing cleanup work. Go back to step 2 and repeat the query periodically until all rollbacks in the affected tablespace are offline and ready to be dropped.

6. Drop the rollback tablespace.  

```
DROP TABLESPACE <tablespace_name> INCLUDING CONTENTS;
```

If this statement fails, please contact Oracle Customer Support. Otherwise, proceed to step 7.
7. Re-create the rollback tablespace.
8. Re-create the rollback segments in the tablespace and bring them online.

1. Prob# 1012943.6 ORA-1113 FILE NEEDS MEDIA RECOVERY
2. Soln# 2061115.6 PERFORM MEDIA RECOVERY ON THE DATAFILE(S)

Prob# 1012943.6 ORA-1113 FILE NEEDS MEDIA RECOVERY

Problem ID: 1012943.6  
Affected Platforms: Generic: not platform specific  
Affected Products: Oracle7 Server  
Affected Components: RDBMS Generic  
Affected Oracle Vsn: Generic

**Summary** ORA-1113 file needs media recovery

**Problem Description** An ORA-1113 will be issued whenever a data file is not in sync with the rest of the database: 01113, 00000, "file %s needs media recovery"

```
// *Cause: An attempt was made to online or open a
database with a file that
// is in need of media recovery.
// *Action: First apply media recovery to the file.
```

Often, ORA-1113 occurs together with ORA-1110. The most common scenarios for an ORA-1113 are:

1. At startup time (usually followed by ORA-1110)
  - The database crashed or was shut down in ABORT mode, or the machine was rebooted while the data file's tablespace was in hot backup mode. At startup, you get ORA-1113.
  - You attempt to open the database with an old version of a data file that was restored from a backup without first bringing it up-to-date.
2. Trying to online a data file  
You try to bring an offline data file back online and get ORA-1113.

**Problem Explanation** Oracle's architecture is tightly coupled in the sense that all database files, data files, redolog files, and control files must be in sync when the database is opened or at the end of a checkpoint. This implies that the checkpoint System Commit Number (SCN) of all data files must be the same. If that is not the case for a particular data file, an ORA-1113 will be generated. For example, when you put a tablespace in hot backup mode, the checkpoint SCN of all its data files is frozen at the current value until you issue the corresponding end backup. If the database crashes during a hot backup and you try to restart it without doing recovery, you will likely get ORA-1113 for at least one of the data files in the tablespace that was being backed up, because its SCN will probably be lower than that of the control file and the data files in other tablespaces. Likewise, offlining a data file causes its checkpoint SCN to freeze. If you simply attempt to take the file online without recovering it first, its SCN will likely be much older than that of the online data files, and thus an ORA-1113 will result.

### Diagnostics and References

- \* {6687.6,Y,100} IS THIS HAPPENING ON STARTUP?
- \* {6690.6,Y,100} DID YOU TRY AND ONLINE THE DATAFILE?
- \* {7241.6,Y,100} BRINGING A TABLESPACE ONLINE

Soln# 2061115.6 PERFORM MEDIA RECOVERY ON THE DATAFILE(S)

|                      |                                |
|----------------------|--------------------------------|
| Solution ID:         | 2061115.6                      |
| For Problem:         | 012943.6                       |
| Affected Platforms:  | Generic: not platform specific |
| Affected Products:   | Oracle7 Server                 |
| Affected Components: | RDBMS Generic                  |
| Affected Oracle Vsn: | Generic                        |

**Summary** Perform media recovery on the data files

**Solution Description** The solution for an ORA-1113 is to perform media recovery on the files having problems using the RECOVER DATAFILE command. If you know that most or all of the files in a tablespace need to be recovered and the database is open, use RECOVER TABLESPACE. If a number of tablespaces need recovery, use RECOVER DATABASE with the database mounted.

The way to do that varies slightly according to the scenario.

**Solution Explanation** Start by querying V\$LOG and V\$LOGFILE. If the database is down, you must mount it first. Connect internal in SQL\*DBA or Server Manager and issue the query:

```
SELECT V1.GROUP#, MEMBER, SEQUENCE#, FIRST_CHANGE#
FROM V$LOG V1, V$LOGFILE V2
WHERE V1.GROUP# = V2.GROUP# ;
```

This will list all your online redolog files and their respective sequence and first change numbers.



The steps to take next depend on the scenario in which the ORA-1113 was issued:

1. At startup after crash with tablespaces in hot backup
  - a. With Oracle 7.1 or lower
    - i. Mount the database.
    - ii. Apply media recovery to the database.  

```
RECOVER DATABASE
```
    - iii. Confirm each of the archived logs that you are prompted for until you receive the message, "Media recovery complete." If you are prompted for an archived log that does not exist, the Oracle server probably needs one or more of the online logs to proceed with the recovery. Compare the sequence number referenced in the ORA-280 message with the sequence numbers of your online logs. Then enter the full path name of one of the members of the redo group whose sequence number matches the one you are being asked for. Keep entering online logs as requested until you receive the message "Media recovery complete."
    - iv. Open the database.
  - b. With Oracle 7.2 or higher
    - i. Mount the database.
    - ii. Find out which data files were in hot backup mode when the database crashed or was shut down in ABORT mode, or the machine was rebooted by running the query:  

```
SELECT V1.FILE#, NAME  
FROM V$BACKUP V1, V$DATAFILE V2  
WHERE V1.STATUS = 'ACTIVE' AND V1.FILE# = V2.FILE# ;
```
    - iii. For each of the files returned by the above query, issue the command:  

```
ALTER DATABASE DATAFILE '<full path name>' END BACKUP;
```
    - iv. Open the database.
2. At startup after restoring a data file or tablespace from a backup
  - a. With the database in ARCHIVELOG mode
    - i. Mount the database.
    - ii. Recover the data file:  

```
RECOVER DATAFILE '<full path name>'
```

  
If recovering more than one data file, issue the following:  

```
RECOVER DATABASE
```

- iii. Confirm each of the archived logs that you are prompted for until you receive the message, "Media recovery complete." If you are prompted for an archived log that does not exist, the Oracle server probably needs one or more of the online logs to proceed with the recovery. Compare the sequence number referenced in the ORA-280 message with the sequence numbers of your online logs. Then enter the full path name of one of the members of the redo group whose sequence number matches the one you are being asked for. Keep entering online logs as requested until you receive the message, "Media recovery complete."
    - iv. Open the database.
  - b. With the database in NOARCHIVELOG mode
 

In this case, you will only succeed in recovering the data file or tablespace if the redo to be applied to it is within the range of your online logs. Issue the query:

```
SELECT FILE#, CHANGE# FROM V$RECOVER_FILE;
```

Compare the change number you obtain with the FIRST\_CHANGE# of your online logs.

If the CHANGE# is greater than the minimum FIRST\_CHANGE# of your logs, the data file can be recovered. In this case, the procedure to be followed is like that of scenario II.A above, except that you must always enter the appropriate online log when prompted, until recovery is finished.

If the CHANGE# is lesser than the minimum FIRST\_CHANGE# of your logs, the file cannot be recovered. Your options at this point include:

    - If the data file is in a temporary or index tablespace, you may drop it with an ALTER DATABASE DATAFILE '<full path name>' OFFLINE DROP statement and then open the database. Once the database is up, you must drop the tablespace to which the data file belongs and re-create it.
    - If the data file is in the SYSTEM or in a rollback tablespace, restore an up-to-date copy of the data file (if available) or your most recent full backup. If a full, consistent backup is not available, please contact Oracle Customer Support.
    - For all other cases in this scenario, you must weigh the cost of going to a backup versus the cost of recreating the tablespace involved, as described in the two previous cases. For more details or to assist you in your decision, please contact Oracle Customer Support.
3. Trying to online a data file or tablespace
  - a. Recover the data file:
 

```
RECOVER DATAFILE '<full path name>'
```

If recovering a tablespace, do the following:

```
RECOVER TABLESPACE <tablespace>
```

- b. Confirm each of the archived logs that you are prompted for until you receive the message, "Media recovery complete." If you are prompted for an archived log that does not exist, the Oracle server probably needs one or more of the online logs to proceed with the recovery. Compare the sequence number referenced in the ORA-280 message with the sequence numbers of your online logs. Enter the full path name of one of the members of the redo group whose sequence number matches the one you are being asked for. Continue to enter online logs as requested until you receive the message, "Media recovery complete."

