DSI303: Advanced Backup, Restore, and Recovery Techniques

Instructor Guide

90018GC11
Production 1.1
April 2000
M010722

ORACLE®

Authors

Shankar Raman
Janet Stern

Technical Contributors and Reviewers

Troy Anthony
Robert Farrington
Lex de Haan
Christine Jeal
Vijay Lunawat
Roderick Manalac
Jose Marco-Dominguez
Michael Möller
Richard Powell

Publishers

Nita Brozowski
Stephanie Jones-Collins

Copyright © Oracle Corporation, 2000. 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 Introduction
Course Objectives 1-2
DSI Curriculum 1-4
Course Prerequisites 1-6

2 Recovery Concepts and Redo Generation
Objectives 2-2
Transactional Properties 2-3
Durability 2-5
Logging Methods 2-6
Redo Architecture 2-8
Physiological Logging 2-9
Redo Generation Architecture 2-10
System Change/Commit number 2-12
SCN 2-13
Redo Byte Address (RBA) 2-14
Change Vector 2-15
Change Vector Example 2-16
Redo Records 2-19
Redo Logs 2-21
Redo Threads 2-22
Redo Log Buffer 2-23
Redo Stream 2-24
Recovery Algorithms 2-25
Page Fix Rule 2-26
Write Ahead Logging 2-27
Log Force at Commit 2-28
Online Log Switching 2-29
Checkpoints 2-30
Normal Checkpoint Mechanics 2-31
Incremental Checkpoint 2-32
Checkpoint Structure 2-34
Applying Redo 2-35
Summary 2-37
References, Acknowledgments, and Further Reading 2-38

3 Internal Recovery Structures

Objectives 3-2
Approaching Recovery Issues 3-3
Diagnostic Trace Files 3-4
Generic Dump Information 3-5
Control File Dump 3-6
Control File Layout 3-8
Control File Dump Reuse Record Section 3-9
Control File Dump 3-11
Control File: File Header 3-12
Control File: Database Entry 3-13
Control File Dump 3-17
Control File: Chkpt Progress 3-18
Control File Dump 3-20
Control File: Redo Thread 3-21
Control File Dump 3-24
Control File: Log File Records 3-25
Control File Dump 3-29
Control File: Data File Records 3-30
Control File Dump 3-33
Control File: Temp File Records 3-34
Control File Dump 3-35
Control File: Tablespace Records 3-36
Control File Dump 3-37
Control File: Log File History Records 3-38
Control File Dump 3-39
Control File: Offline Range Records 3-40
Control File Dump 3-41
Control File: Archived Log Records 3-42
Control File Dump 3-43
Data file Headers Dump 3-44
Datafile Header 3-45
Datafile Header Dump 3-53
Redo Log Headers Dump 3-54
Log File Header 3-55
V$ Views 3-56
Summary 3-59
References 3-60

4 Offline File Recovery
Objectives 4-2
Files After a Clean Shutdown 4-3
Recovering Data Files After a Clean Shutdown 4-4
Files After Shutdown Abort or Crash 4-5
Recovery After Shutdown Abort 4-6
Recovering Files from Cold Backup 4-7
Hot Backups 4-9
Restore From a Hot Backup 4-12
Offline Tablespaces 4-13
Offline Datafiles 4-14
Read-Only Tablespaces 4-15
Read-Only Tablespace Benefits 4-16
Read-Only Tablespace Internals 4-17
Recovery of Read-Only Files 4-20
Recovery of Read-Only Files Current Control File 4-21
Recovery of Read-Only Files Backup Control File 4-22
Recovery of Read-Only Files Backup Control File Caveats 4-23
Summary 4-24
References 4-25

5 Block, Crash, and Instance Recovery
Objectives 5-2
Block Level Recovery 5-3
Block Recovery: Buffer Headers 5-5
Block Recovery: When to Apply Redo 5-6
Block Recovery: Failure 5-7
Redo Application 5-8
Crash/Instance/Thread Recovery 5-12
Tuning Instance Recovery 5-15
Thread Recovery 5-16
LOG_CHECKPOINT_TIMEOUT 5-19
LOG_CHECKPOINT_INTERVAL 5-20
Fast-Start Checkpointing Architecture 5-21
Use of FAST_START_IO_TARGET 5-22
Summary 5-23
References 5-24

6 Media Recovery
Objectives 6-2
Media Recovery 6-3
Media Recovery: Data file Recovery 6-5
Media Recovery: Tablespace Recovery 6-6
Media Recovery: Database Recovery 6-7
When to Use Media Recovery 6-10
Media Recovery Required 6-11
Checkpoint Structure 6-12
Recovery with Archived Logs 6-14
Media Recovery: Threads 6-15
Threads 6-16
Thread Merged Recovery 6-17
Thread Merged Redo 6-18
Media Recovery: Fuzzy Bits 6-19
Media Recovery Fuzzy Bit 6-20
Steps of Media Recovery 6-21
Completion of Media Recovery 6-24
Incomplete Recovery 6-25
Media Recovery with Backup Controlfile 6-26
V$ Views for Recovery 6-27
Useful Recovery Tracing Events 6-29
Summary 6-30
References 6-31

7 Resetlogs
Objectives 7-2
Usage 7-3
Usage: Incomplete Media Recovery 7-4
Usage: Backup Control File 7-5
Usage: Create Control File with Resetlogs 7-6
!!! Warning !!! 7-7
Mechanics of Resetlogs 7-8
Resetlogs Actions 7-9
Resetlogs Consistency Checks 7-10
Failure of Consistency Checking 7-12
Consequences of Resetlogs 7-14
Rollforward Through Resetlogs Example 7-15

Rollforward Through Resetlogs 7-16
Recovering Through Resetlogs 7-17
Forcing the Database Open 7-19
_offline_rollback_segments 7-20
_corrupted_rollback_segments 7-21
_corrupt_blocks_on_stuck_recovery 7-22
_allow_resetlogs_corruption 7-23
_allow_read_only_corruption 7-24
Summary 7-25
References 7-26

8 Point-In-Time Recovery
Objectives 8-2
What Is Object Point-In-Time Recovery? 8-3
Object PITR 8-4
Methodology 8-5
TSPITR: Basics 8-7
TSPITR: Caution! 8-8
TSPITR: Terminology 8-9
TSPITR: Dependency Rules 8-11
TSPITR: Check Views 8-13
Steps Involved in TSPITR 8-15
Prepare the Primary 8-16
Prepare the CLONE 8-17
Configure the CLONE 8-18
Start the CLONE 8-19
Prepare CLONE for Recovery 8-20
Recover and Open CLONE Database 8-21
Prepare the CLONE for Export 8-22
Export from the CLONE 8-23
Export for TSPITR 8-24
Move Datafiles to the Primary 8-26
Import into the Primary 8-27
TSPITR: Finishing Touches 8-28
TSPITR: Limitations 8-29
TSPITR of Partition Tablespaces 8-30
TSPITR with Transportable Tablespaces 8-31
Summary 8-32
References 8-33

9 Recovery Performance and Parallel Recovery
Objectives 9-2
What Is Parallel Recovery? 9-3
Parallel Recovery Usage 9-4
Typical Parallel Recovery Session 9-6
Parallel Recovery Rollforward 9-7
Roll Forward: Benefits 9-9
Parallel Roll Forward Benefits 9-10
FREEZE_DB_FOR_FAST_INSTANCE_ RECOVERY 9-11
Fast-Start Parallel Rollback 9-12
Controlling Fast-Start Parallel Rollback 9-14
Monitoring Parallel Rollback 9-15
On-Demand Parallel Rollback 9-16
Summary 9-17
References 9-18

10 Archiving Internals
Objectives 10-2
Redo Log Concepts 10-3
Automatic Redo Log Archiving 10-4
Benefits of Redo Log Archiving 10-5
Operation of the Archiver 10-6
Enabling Automatic Archiving 10-8
Multiple Archive Destinations 10-9
Interaction of Semantics 10-12
Init.Ora Examples 10-14
Manual Redo Log Archiving 10-15
Archiving Tips 10-16
Managing the Archive Destination 10-17
Archiving and Parallel Server 10-18
Archiving and Parallel Server Force SCN 10-19
ARC0 Inter-Instance Communication 10-20
Control File Structures for Archive 10-21
Archive Data in the Control file Example 10-22
Controlfile Structures Archive Link List 10-23
Control File Dump Visualization 10-24
Using Multiple Buffer Streams 10-25

Sanity Checks 10-26
When ARC0 Detects Corruption 10-27
Events That Post ARC0 10-28
Tuning and Configuring Archive process 10-29
Tuning and Configuring ARC0 10-30
Diagnosing ARC0 Problems 10-31
V$ARCHIVE_DEST COLUMNS 10-34
SUMMARY 10-36

11 Oracle Standby Database
Objectives 11-2
Disaster Recovery Concepts 11-3
Standby Database Features 11-4
Standby Architecture 11-5
Guidelines 11-6
Initialization Parameters 11-7
LOG_ARCHIVE_DEST_n 11-9
Net8 Config Files 11-10
Maintaining the Standby Database 11-12
Managed Recovery Mode 11-13
Maintain Standby Database - Manual Recovery 11-14
Useful Views 11-15
Troubleshoot Archiver Problems 11-16
Reestablish the RFS Process 11-17
Recovering Standby Database 11-18
Read Only Mode of Standby 11-19
Activating the Standby Database 11-20
Effects of Physical Structure Changes 11-21
Altering Physical Structure at Primary Database 11-27
Refreshing Standby Database Control File 11-28
Nologging Operations 11-29
Quick Summary for Standby Databases 11-31
Common Mistakes to Avoid 11-32
Control File Dump Main Section 11-33
Control File Dump Log File Section 11-34
Control File Dumps 11-35
Case Study 1 11-36
Case Study 2 11-37
Summary 11-38
Credits/References 11-39

A Supplemental Information

B Practices

C Solutions

1

Introduction

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Schedule:	Timing	Topic
	10 minutes	Lecture
	10 minutes	Total

Course Objectives

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

- Understand the structures used during recovery operations
- Analyze control file, datafile header and redo log header dumps
- Understand how offline files are recovered.
- Discuss the differences between block, crash, and instance recovery

1-2 Confidential

Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Course Objectives

The three days DSI301 course is the first in a series of courses that targets Oracle Support Services (OSS) technical analysts and customer support. The primary purpose is to provide participants with the tools and utilities to analyze and troubleshoot server-related failures. It is also intended as a preparatory course for courses DSI302 to DSI308.

Course Objectives

- Understand the mechanics behind Media Recovery
- Explain the ramifications of using resetlogs
- Perform Tablespace Point-in-Time recovery
- Understand the internal activities of the archiver process
- Be able to set up a Standby database and open it in read-only mode

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

DSI Curriculum

The Data Server Internals (DSI) curriculum consists of the following eight courses:

- **DSI301: Advanced Server Support Skills**
- **DSI302: Data Management**
- **DSI303: Database Backup and Recovery**
- **DSI304: Query Management**

1-4 Confidential

Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

DSI Curriculum

The DSI curriculum provides internal and confidential information for OSS technical analysts and customer support staff with a minimum of 12 months experience supporting the RDBMS. Each of the following courses lasts four days:

- DSI301 introduces the student to advanced support skills such as reading block dumps, recovering from corruptions and reading the Source Code.
- DSI302 provides insight into data management within the Oracle server and uses demonstrations and lab exercises to show how rollback operations are performed.
- DSI303 teaches participants about backup and restore; the internals of redo generation, application, and archiving; block, crash, instance, and media recovery; the resetlogs operation; the implementation of read-only and offline tablespaces; object (TSPITR) recovery; parallel recovery; and how to gather and analyze recovery diagnostics.
- DSI304 provides insight into query operations within Oracle and how to interpret and modify their execution plans.

DSI Curriculum

- DSI305: Database Tuning
- DSI306: Very Large Databases
- DSI307: Distribution and Replication
- DSI308: Parallel Server

1-5 Confidential

Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

DSI Curriculum (continued)

- DSI305 provides detailed insight into database tuning, enabling students to interpret and evaluate a database profile and to gain an understanding of the internal layers and mechanisms that give rise to the common database tuning profiles.
- DSI306 teaches participants about transaction monitors, features, and relevance in a VLDB environment. Participants also learn about techniques like MTS, partitioning (manual and automatic), and parallel DML. Other topics covered include advanced queuing, sort enhancements, deferred transaction recovery, and tuning data loads.
- In DSI307, participants explore the troubleshooting and design of advanced replication architectures.
- DSI308 discusses Oracle7 and Oracle8*i* Parallel Server advanced topics.

Course Prerequisites

Required classes:

- **DSI 301**
- **DSI 302**

Assumed knowledge:

- **Interpreting block dumps**
- **Rollback segment header structures and transactional concepts**
- **Basic understanding of source code concepts**
- **12 months of Database support experience**

1-6 Confidential

Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Course Prerequisites

DSI 301 covers basic block dump analysis and how to recover from rollback segment corruption.

DSI 302 covers the different datatypes, segment header blocks, and segment data blocks and how to interpret the information contained within block dumps of these items. DSI 302 also covers the transactional properties in great depth, which we will not cover again in this class.

2

Recovery Concepts and Redo Generation

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Schedule:	Timing	Topic
	60 minutes	Lecture
	40 minutes	Lab
	100 minutes	Total

Objectives

After completing of this lesson, you should be able to describe the following:

- **Transactional properties**
- **Redo logging methods**
- **Redo generation architecture**
  - **Structures**
  - **Algorithms**
- **Analyzing redo log dumps**
  LogMiner
- **Applying redo**

2-2

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Objectives

This lesson covers the following topics:

1. Transactional properties—particularly the maintenance of ACID principles with regard to transactions.
2. The implementation of the redo generation architecture adopted by Oracle, with emphasis on such topics as logical redo ordering, write-ahead logging, log force at commit, page fixing principals, coordination of online log switching, and the importance of checkpointing all discussed in the context of physiological log generation.
3. Recovery concepts important to Oracle implementations. including block, instance, crash, and media recovery. This lesson also considers the phases involved in recovering an inconsistent database : roll-forward and rollback.
4. This lesson will also review certain common examples of failure and the means of restoring these systems to a consistent state.

Transactional Properties

In order to maintain consistency at both the database and the transaction level, Oracle must enforce the following properties:

- **Atomicity**
- **Consistency**
- **Isolation**
- **Durability**

2-3

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Transactional Properties

For more information on the concepts of transaction processing, see

"*Transaction Processing: Concepts and Techniques*." by Jim Gray and Andreas Reuters

- Atomicity: A transaction's changes to the state are atomic: either all happen or none happen. These changes include database changes, messages, and actions on transducers.

  Example: A mail note is only sent after one commits. When it is sent, the entire mail note is sent.

- Consistency: A transaction is a correct transformation of the state. The actions taken as a group do not violate any of the integrity constraints associated with the state. This requires that the transaction be a correct program. The view of an object does not change within the transaction.

  Example: While you are reading a mail, the content will not change; the same version of the mail note will remain a constant whether or not there are other concurrent modifications.

- Isolation: Even though transactions execute concurrently, it appears to each transaction, T, that others executed either before T or after T, but not both. Other uncommitted changes or committed changes after a certain timestamp are ignored within your own transaction.

  Example: While writing or sending email, other people can be doing the same thing. Subsequently, there will be an order or sequence of mail notes.

- Durability: Once a transaction completes successfully (commits), its changes to the database state survive failures.

  Example: If the mail system fails, all sent mail is still in your inbox when you open your mailbox again.

Transactional Properties (continued)

Oracle needs to enforce these transactional properties while at the same time allowing for concurrent processing, processing speed, and so forth.

Many of the algorithms implemented for processing capabilities mean that at any stage during normal database operation, different portions of the database (and the memory structures used to access it) are inconsistent with regard to one another.

The database is the physical files in which we store data (the datafiles, the redo log files, and the control file) and the instance the memory structures, shared global area/process global area (SGA/PGA), maintained by the Oracle relational database management system (RDBMS), then we know that database blocks are read from the disk into memory buffers within the SGA. These blocks are then modified while resident in these buffers, with least recently used (LRU) algorithms and other algorithms, meaning that many changes could take place before these blocks are written back into the datafiles. These blocks are therefore from a later state in time than blocks on disk, and perhaps other blocks in memory. The database is therefore inconsistent, and anything other than a controlled (graceful) shutdown will leave the database inconsistent.

The problem that the Recovery module must address is that of transactional durability. Once a transaction commits, the changes that it has made must survive RDBMS failures.

Durability

Transaction durability:

Once a transaction commits it *must* survive failures.

2-5

Confidential

Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Durability

Once a transaction completes successfully (for example, commits), any changes that are made to the database state within the scope of this transaction must survive failures.

A valid backup and recovery strategy enables the database to be restored to a consistent state in almost all scenarios. Where this system falls down is dependent on the planning and implementation by the database administrator, and the value that is placed on the data to be protected. If some loss can be tolerated then a cheaper, more simplistic solution can be provided. However as the data value increases then more expensive (and possibly more complex) solutions may be devised, in an effort to protect against all possible failure scenarios.

The backup and recovery process must be designed to accommodate both planned and unplanned outages, although typically it is considered as a means of restoring consistency after an unplanned outage. Oracle's recovery mechanism copes with both types of outages provided that sufficient resources are available.

Logging Methods

- **Physical logging:**
  - **Block copies**
- **Logical logging:**
  - **Transaction logging**
- **Physiological logging :**
  - **Combination of physical and logical**

2-6

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Logging Methods

Methods of logging changes made within a database include:

Physical logging:

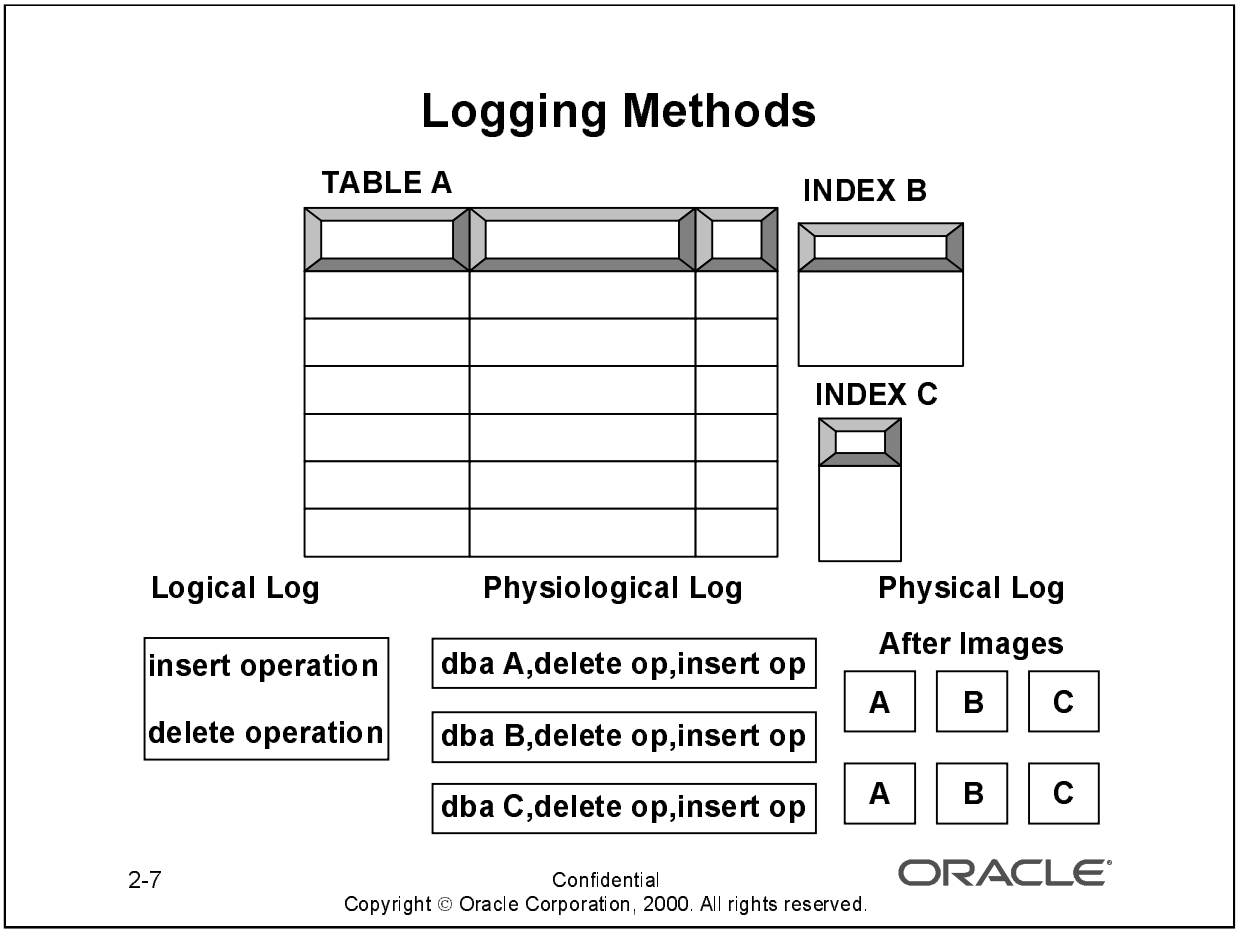
Logs the before and after image of blocks. This is usually achieved by keeping either the changes made to a block or an entire copy of the block, along with the before image, (sometimes called the shadow copy). Physical logging concentrates on one block at a time. Because we need to store multiple images of a block, the amount of information recorded can be quite high (for example, 32 K blocks would require an image for each change).

Logical logging:

Logical logging is often called operational logging. An operation is recorded, such as a delete or an update. The operation to perform or undo a change has also to be recorded, with the operation perhaps affecting many physical blocks (Consider the statement 'DELETE * FROM GIANT_TABLE', which is what would be recorded). A significant drawback of this method of logging is that is difficult to apply during recovery, and does not support *idempotence* (That is, changes must be applied sequentially , and is transaction based, therefore if stopped in the middle of a transaction all changes have to be re-applied).

Physiological logging:

Oracle adopts a combination of physical and logical logging. Structures called redo records which consist of one or more change vectors are maintained. A change vector is the physical change to a database block (an atomic change to a block), while the redo record is a change to the database state (an atomic database change). The redo record therefore tracks the logical changes (or transactional changes).



Logging Methods (continued)

Assume that you have a table (Table A) with two indexes (Index B and Index C), and you issue two statements within the one transaction against that table: an insert operation, followed by a delete operation. This constitutes actions against three different segments in the database: the table and its two indexes. (For simplicity this does not include a discussion on any other structures involved in an Oracle operation such as rollback segments or dictionary objects.)

A logical logging method would store the insert and delete operations. This does not require much storage, but requires that the operations be applied serially, and in their entirety during a recovery scenario (that is not to say that Oracle allows application of part of a transaction, but consider this from a point of stopping and then restarting recovery. A logical logged database may not have applied any of these changes, the transaction would need to be replayed if you start recovery again. Oracle would rollback the transaction if you opened the database.).

A database operating under a physical logging protocol would need to store before and after images of all blocks affected, a significant amount of storage required.

Physiological logging uses the best features of both of these methods, storing the individual block changes as a series of change vectors, plus the opcode to record the operation performed.

Redo Architecture

- Physiological logging is achieved with Oracle's Redo Architecture.
- Coordination exists between:
  - SGA

    buffer cache, log buffer
  - Physical Devices

    data files, redo log files (online + archived)
  - Background processes

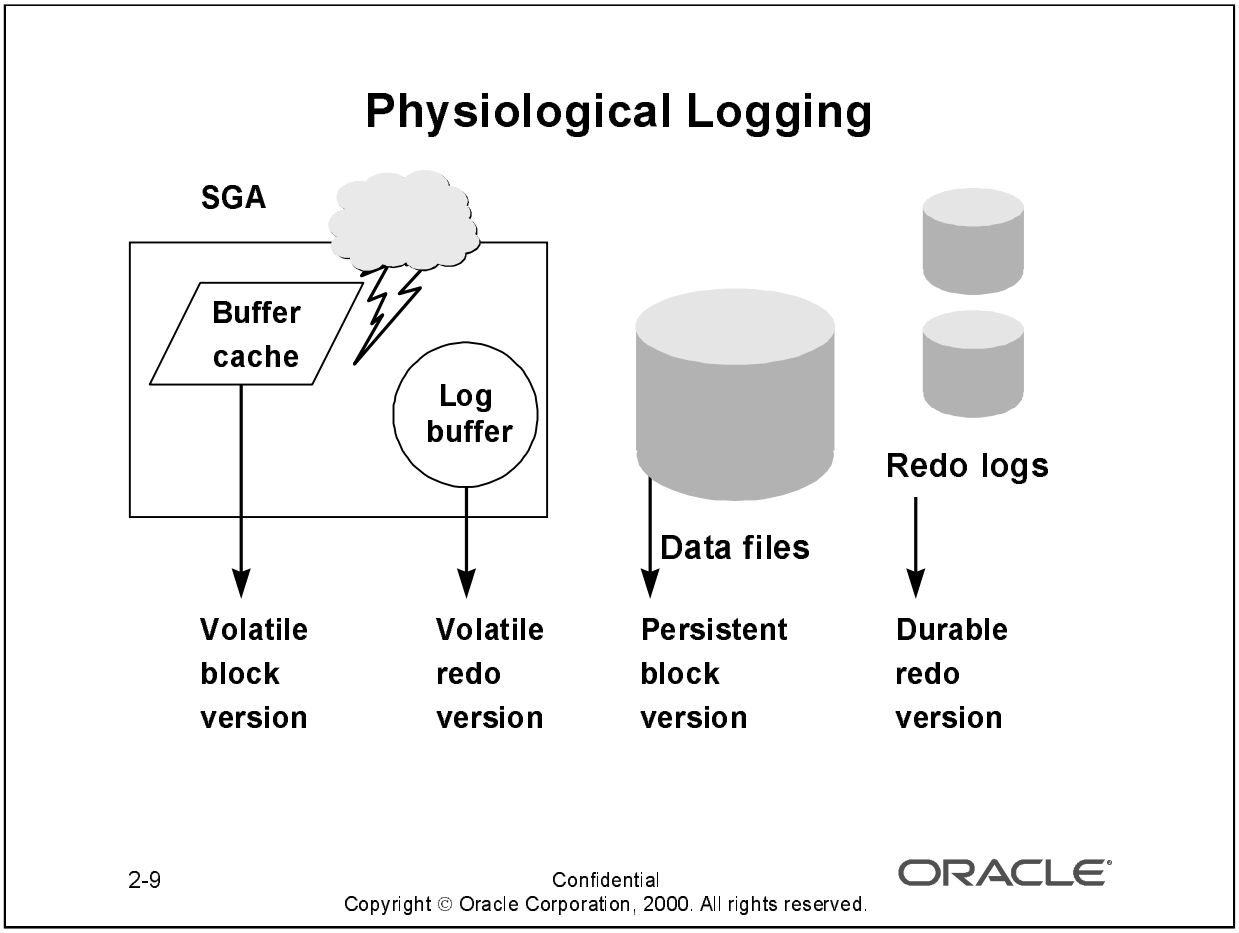
    LGWR, DBW0, CKPT (ARC0)

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Redo Architecture

Changes to database blocks are made within the SGA and stored permanently on disk. The data is transferred between the SGA and the disk devices by various background processes.



Physiological Logging

In the normal operation of an Oracle database, the buffer cache contains the most current version of a data block (contains the latest change). These volatile blocks can be in advance of the block stored on disk (the persistent version of the block). This is due to the fact that Oracle is optimized for OLTP performance. The LRU algorithm maintains a block in memory, as long as it is being modified, with database writer (DBW0) periodically flushing changed buffers (dirty buffers) to disk.

Although this is good from a performance perspective, it leaves a certain vulnerability if any situation causes the loss of the volatile memory areas. For example, an instance failure removes the SGA, and both the buffer cache and the log buffer (and all the changes recorded in them) are lost. It is the responsibility of the redo architecture to reconstruct those changes using only the durable redo (contained in the online redo log files) and the persistent blocks (contained in the data files).

Accepting that the buffer cache contains the latest block version, and the log buffer has the changes to be applied built into it first, then we can say that the volatile block version (VBV) is in advance of the persistent block version (PBV), and the volatile redo version (VRV) is in advance of the durable redo version (DRV) (that is VBV $\geq$ PBV and VRV $\geq$ DRV and VRV $\geq$ VBV). If the instance is lost then DRV $\geq$ PBV and the current commit System Change Number (SCN) will be less than or equal to the highest value in the DRV.

Redo Generation Architecture

Oracle's Redo Architecture consists of:

- **SCN**
- **RBA**
- **Change Vectors and Redo Records**
- **Redo Log Files**
- **Redo Threads**
- **Redo Log Buffer**
- **Checkpoint Structures**

2-10 Confidential

Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Redo Generation Architecture

The following structures are involved in the recovery mechanism.

System Change Number (SCN). This records a consistent version of the database. An SCN is allocated and saved in the header of a redo record that commits a transaction. SCNs are also recorded in other data structures, such as datafile headers, control file structures and block header structures (among others).

Redo Byte Address (RBA): An RBA acts as a pointer to where a particular change vector appears in the redo stream. It consists of three fields, the log sequence number, the block number within the log, and byte offset within the block. The RBA structure is 10 bytes long.

Change vectors/redo records: A change vector describes a change to a single database block. A group of change vectors which comprise a single atomic change to the database are called a redo record.

Redo log files: A series of circular files to which all changes made to database blocks (as well as information to reconstruct some control file changes) are recorded. Each log is identified by its thread number (see next definition), sequence number within a thread, and the range of SCNs contained in it. The information in the redo log files is logically ordered to preserve the order in which changes were made, with the ordering based on SCN. This supports the Isolation rule of the ACID properties of transactions.

Redo Generation Architecture (continued)

- Thread of redo: The redo generated by one instance is called its thread of redo. Each thread is written to its own private set of redo log files. The logs currently being written to are termed "online" and these can be optionally archived (copied) once writing is complete. A thread can therefore have both an online and an archived portion. In the case of Oracle Parallel Server (OPS), each instance has its own independent thread of redo. During recovery a merged thread approach must be taken to provide the logical ordering required.
- Checkpoint structure: A data structure that defines a point in the redo stream, before which all changes to blocks have been written to disk. The checkpoint structure contains a checkpoint SCN, an enabled thread bit vector (a bit mask identifying what threads were enabled at the time of the checkpoint), a timestamp, the number of the thread that initiated the checkpoint (nontrivial in the OPS case) and an RBA.

System Change/Commit Number

- Describes the “version” or the committed version of a database
- 6 bytes long consisting of a BASE (4 bytes) and WRAP (2 bytes)
  - The base increments at the allocation of each new SCN
  - The wrap increments when the base turns over
- Multi-instance databases maintain a global SCN

2-12

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

System Change/Commit Number

An SCN defines a committed version of the database. A query reports the contents of the database as it looked at some specific SCN. An SCN is allocated and saved in the header of a redo record that commits a transaction. An SCN may also be saved in a record when it is necessary to mark the redo as being allocated after a specific SCN. SCN's are also allocated and stored in other data structures such as the control file or datafile headers.

An SCN is at least 48 bits long. Thus they can be allocated at a rate of 16,384 SCN’s per second for over 534 years without running out of them. At this rate, an Oracle database will run out of SCN’s in June, 2522 AD (using 31 day months for time stamps).

In a multi-instance system a distinction is made between the current and recent SCN (on a single instance system they are the same). The current SCN is the most up to date system wide SCN and is maintained in a lock value block of a global lock (type SC). The recent SCN is a cached value of the current SCN.

SCN

- Assigned to a transaction when it commits.
- Increases monotonically.
- May jump in a distributed transaction or in a parallel instance environment.
- Used for:
  - Concurrency control
  - Redo Log Ordering
  - Recovery

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Redo Byte Address (RBA)

```
REDO RECORD - Thread:1 RBA:  0x000065.00000002.0010
LEN: 0x02f0 VLD: 0x01 SCN scn:  0x0000.0000e4d3
05/12/98
15:19:54
```

- Marks the start of a redo record
- 10 bytes long
- Has three components:
  - Log sequence number
  - Block number within the redo log
  - Byte number within the block

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Redo Byte Address (RBA)

An RBA identifies the start of a redo record within an online redo log file or an archive log file. Thus it points to the exact location of the redo record in a particular thread. It is stored in all datafile headers and the data file entry in the control file. It is used to quickly locate the redo to start recovery.

Change Vector

- Describes a single change to one Oracle data block
- Contains information required to move the block from one consistent state to the next consistent state

  DBA, SCN, SEQ, and the operation
- Always created before modifying the data block buffer

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Change Vector

A change vector describes a single change to a single data block. It has a header that gives the data block address (DBA) of the block, the sequence number, the SCN, the change type, the block class, and the operation. After the header is information that depends on the operation.

The DBA and the sequence number are copied from the block header when the change vector is constructed.

Change Vector Example

```
CHANGE #1 TYP:0 CLS:15 AFN:3 DBA:0x00c00002
   SCN:0x0000.0010e2cb SEQ: 1 OP:5.4
```

TYP	**Change Type**
CLS	**Block class**
AFN	**Absolute File number**
DBA	**Relative DBA**

2-16

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Change Vector Example

Example of a change vector header:

```
CHANGE #1 TYP:0 CLS:15 AFN:2 DBA:0x00800002 SCN:0x0000.0000e4d0
SEQ:  1 OP:5.2
```

TYP:	Change type - Normal
KCOCT_NORM	Normal change vector
KCOCT_NEW	Completely new contents and usage of a block
KCOCT_DLOG	Delayed logging change included
KCOCT_IMAGE	Block image for hot backup ¾ not a change
KCOCT_CLEAR	Empty block image ¾ just to new the block
KCOCT_INVLD	Invalidation redo from UNRECOVERABLE direct path
KCOCT_MARKER	Media recovery marker
KCOCT_MAX	Max type for detecting corrupt redo logs

The block class (CLS) gives the class that the block was pinned under when it was changed. This is not the block type that defines the format of the data. The class is used for locking to prevent false deadlocks. The class field in redo is used to validate that the correct lock is invalid during instance recovery. If this is a change of type KCOCT_INVLD then there is no class because no locks are held, and this field contains the number of blocks to invalidate.

The absolute file number (AFN) and the relative DBA define the disk address of the block that is modified by this change. Both may be zero if the change is not for a specific block.

Change Vector Example

```
CHANGE #1 TYP:0 CLS:15 AFN:3 DBA:0x00c00002
   SCN:0x0000.0010e2cb SEQ: 1 OP:5.4
```

SCN System Commit Number

SEQ Sequence Number

OP Opcode

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

Change Vector Example (continued)

The SCN and SEQ describe the version of the block that this change expects to be applied to. They are extracted from the block when creating this change vector. After this change is applied, the block header is at the SCN associated with the log where the redo is located. If there are multiple change vectors at the same SCN in the redo log stream, then they will have different sequence numbers. Note that all changes at the same SCN are in the same thread of redo because a new SCN is needed to ensure correct ordering of redo application. Some changes can be applied to multiple versions of a block. In those cases the change type will be set to a special value, and this SCN will represents the greatest block version that this can be applied to.

The opcode (OP) defines the routines to call when applying this change or doing a symbolic dump of this change vector.

There is a list of a opcodes included as an appendix to this course.

Change Vector Example

```
CHANGE #3 TYP:2 CLS:1 AFN:6 DBA:0x01800003
   SCN:0x0000.0010e2cd SEQ: 1 OP:11.5
KTB Redo
op: 0x11   ver: 0x01
op:  F  xid: 0x0003.00f.00000030    uba:
   0x00c0212a.001a.05
Block cleanout record, scn:  0x0000.0010e2ce  ver:
   0x01, entries follow …
itli: 1  flg: 2  scn: 0x0000.0010e2cd
KDO Op code:  URP   xtype:  XA  bdba: 0x01800003
   hdba: 0x01800002
tabn: 0  slot: 3 (0x3)  flag: 0x2c  lock: 1  ckix: 0
ncol: 3  nnew: 1  size: 0
col 2: [7]  78 c7 0c 1e 01 01 01
```


Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Change Vector Example (continued)

The change vector shown in the slide is part of a redo record for a table update. The DML for this operation is:

```
UPDATE TABLE some_rows set col3='30-dec-99' where col1=4;
```

some_rows has 3 columns: col1 - number, col2 - varchar2, col3 - date.

Redo Records

- Describe a single atomic change to the database
- A collection of one or many change vectors
- Generated before modifying the Oracle data buffer
- Recovery guarantees all or nothing when applying changes from a redo record

  This atomicity guarantee is one of the fundamental jobs of the cache layer
- Ordered by SCNs

2-19 Confidential

Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Redo Records

A redo record is a group of change vectors describing a single atomic change to the database. A transaction can generate multiple redo records. The grouping of change vectors into a redo record allows multiple database blocks to be changed so that either all changes occur or no changes occur, despite arbitrary intervening failures. This atomicity guarantee is one of the fundamental jobs of the cache layer. Recovery preserves redo record atomicity across failures.

Redo records are ordered by SCNs:

- SCNs are monotonically increasing.
- CHANGE SCN is used for ordering the redo.
- COMMIT SCN is used to mark when a transaction was committed.

Redo Records

Example:

```
REDO RECORD - Thread:1 RBA:
   0x00004f.00000002.0010 LEN: 0x0214 VLD: 0x01
   SCN scn: 0x0000.0010e2cb 01/03/00 08:49:09
   CHANGE #1 TYP:0 CLS:15 AFN:3 DBA:0x00c00002
   SCN:0x0000.0010e2c3 SEQ:  1 OP:5.2
          (Information about operation follows)
   CHANGE #2 TYP:0 CLS:16 AFN:3 DBA:0x00c0002e
   SCN:0x0000.0010e2c1 SEQ:  2 OP:5.1
          (Information about operation follows)
   CHANGE #3 TYP:2 CLS: 1 AFN:6 DBA:0x01800003
   SCN:0x0000.0010e2c9 SEQ:  1 OP:11.2
          (Information about operation follows)
    CHANGE #4 MEDIA RECOVERY MARKER
   SCN:0x0000.0010e2cb SEQ:  0 OP:5.19
          (Information about operation follows)
```

2-20

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Redo Records (continued)

As explained earlier, a redo record is a group of change vectors.

How many change vectors does this redo record have and what does each one represent ?

A. Four change vectors:

- CV for the transaction table (rollback segment header) OPCODE 5.2
- CV for the undo block (rollback segment) OPCODE 5.1
- CV for the actual datablock (insert row piece) OPCODE 11.2
- CV for MEDIA RECOVERY purposes OPCODE 5.19

For more information on OPCODES, see ktucts.h

Redo Logs

- Changes to data blocks are made by:
  - Constructing a change vector for the change
  - Constructing a redo record for a set of change vectors
  - Saving this redo record to a redo log
- Redo records in a log are ordered by SCN.
- The header of a redo log contains the low SCN and the next SCN:
  - low SCN—>first redo record
  - Next SCN—>last redo record—>low SCN of the next log

2-21 Confidential

Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Redo Logs

All changes to database blocks are made by constructing a redo record for the change, saving this record in a redo log, and then applying the change vectors to the datablocks. Recovery is the process of applying redo to old versions of datablocks to make them current. This is necessary when the current version has been lost.

When a redo log becomes full, it is closed and a log switch occurs. Each log is identified by its thread number, sequence number (within thread), and the range of SCNs spanned by its redo records. This information is stored in the thread number, sequence number, low SCN, and next SCN fields of the logfile header.

The redo records in a log are ordered by SCN. Moreover, redo records containing change vectors for a given block occur in increasing SCN order across threads (case of Oracle Parallel Server — OPS).

Only some records have SCN's in their header, but every record is applied after the allocation of the SCN appearing with or before it in the log. The header of the log contains the low SCN and the next SCN. The low SCN is the SCN associated with the first redo record (unless there is an SCN in its header). The next SCN is the low SCN of the log with the next higher sequence number for the same thread. The current log of an enabled thread has an infinite next SCN, because there is no log with a higher sequence number.

Redo Threads

- A redo generated by an instance is called a thread of redo.
- A parallel server database with multiple active instances will have multiple threads of redo.
- Each thread of redo generally switches logs independently of the other threads.

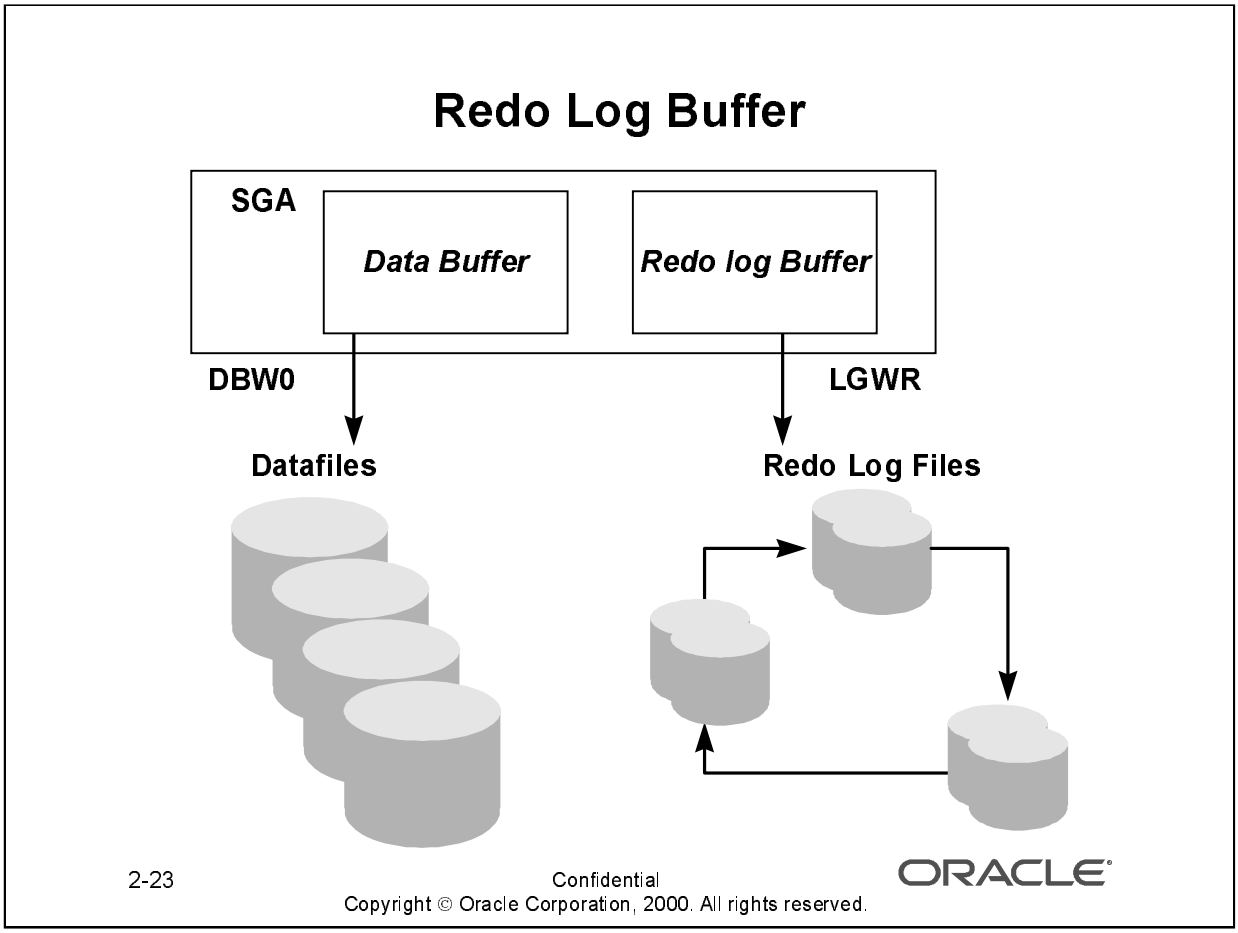
2-22

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Redo Threads

For each logfile, there is a record in the control file that describes the log file. The index of a log's control file record is referred to as its log number. Log numbers are equivalent to log group numbers, and they are unique across all threads. The list of a thread's log file records is anchored in the thread record (that is via head and tail logfile record indices), and linked through the logfile records, each of which stores the thread number. The logfile record also has fields identifying the number of group members, as well as the head and tail (that is filename record indices) of the list (linked through filename records) of filenames in the group.



Redo Log Buffer

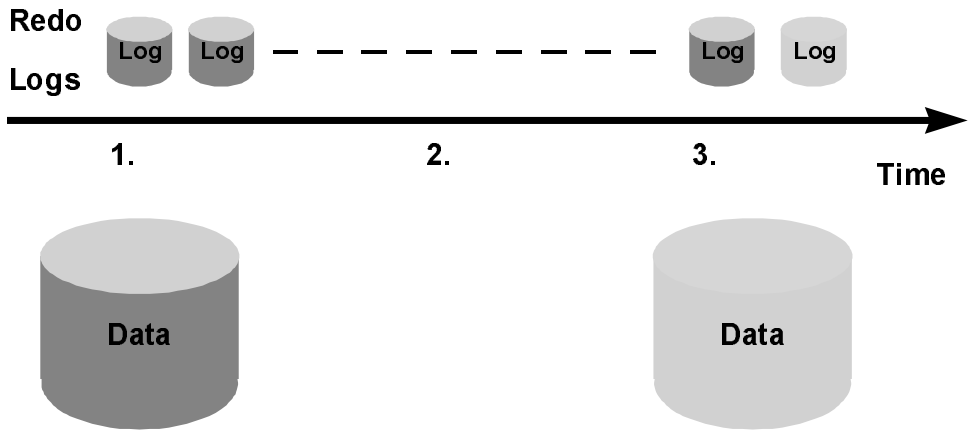
Redo log buffers have the following characteristics:

- Space is allocated in SGA to hold redo record.
- LGWR writes out to the redo logs from these buffers.
- Allocating space in the redo log buffer ends up allocating space in the log files.
- Only after the redo record has been built in the redo buffer can the data block buffers be changed.

Write-ahead log property is discussed later in the lesson. It ensures that LGWR writes the redo log buffer consisting of changes done to a particular block to disk before DBW0 can write out the cache buffer pertaining to that data block.

Redo Stream

The logical life of a database is between two resetlogs points.



2-24

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Redo Stream

The logical lifespan of a database is between two resetlogs points (that is, two incomplete recovery points). As shown in the slide:

1. A Create Database (or open resetlogs command) initializes a database.
2. Redo is generated in a logically ordered stream, where an RBA identifies a point within that stream.
3. If an open resetlogs is issued at some point (that is, an incomplete recovery is performed), then a new incarnation of the database is created. The redo stream is reinitialized and is only applicable to the new incarnation of the database.

Prior to version 7.3 of Oracle it was impossible to recover a database through a resetlogs point. Under certain exact conditions it may be possible to recover through a resetlogs, however the conditions may be difficult to meet. This is the topic of a paper by Lawrence To (WEB IV Note: 50451.1), and will be discussed briefly later in this course.

Recovery Algorithms

In order for recovery to function, a number of algorithms need to interact:

- **Page fix**
- **Write ahead logging**
- **Log force at commit**
- **Online log switching**
- **Checkpointing**

2-25

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Recovery Algorithms

These algorithms function in unison to ensure that the Oracle database can be restored to a consistent state should a processing failure occur.

Page Fix Rule

Oracle's Fix rule:

- **Locks a page (buffer) prior to reading or modifying**
- **Generates redo prior to modifying the page**
- **Maintains consistency and isolation of the page**

2-26 Confidential

Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Page Fix Rule

An Oracle buffer (or page) needs to be prevented from being accessed while it is being modified. The Fix rule prevents this access until all the changes needed to take the buffer to its next consistent state have been applied.

The locking portion of the Fix rule is implemented on UNIX with semaphores:

1. Obtain a page semaphore in exclusive mode prior to altering the page, or obtain a page semaphore in shared or exclusive mode prior to reading.
2. Hold the semaphore until the changes to be applied are recorded in a log buffer and the changes have been made to the page.

A redo record is created prior to changing the buffer to ensure that a consistent state can be restored in case of failure. Oracle guarantees that when a page is unpinned after the page-action completes, the redo records are consistent with the page state or Oracle buffer state.

A failure in a page action (for example process death during the modification of a buffer) which results in an inconsistent page, requires online block recovery. This necessitates the application of redo records to a copy of the block obtained from the data file.

Write Ahead Logging

Write ahead logging guarantees:

- **Before DBW0 writes out a data block buffer, LGWR must have written the necessary redo records to the log files.**
- **Before a block buffer is changed in the buffer cache, redo must have been written to the log buffer.**

2-27

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Write Ahead Logging

Before a change is made to a cache managed block undo for that block (in the rollback segment) must be generated , also redo must be generated for both the undo change and change itself. The redo is copied into the log buffer and this change is then made to the blocks in the buffer cache. The routine that makes the change applies the change vectors to the block(s) in the buffer cache (the same mechanism that occurs during recovery).

The generation of redo, and the subsequent writing of that redo to the log file makes the change real. Once an entry is successfully written in the log file then the change can have been considered to have occurred, regardless of what happens to the database system after this.

Before DBW0 can write a dirty block back to disk, the redo log records that contain the corresponding changes must be written to persistent storage (the online log files). The entries are written to the online logs by LGWR. A commit, filling of the log buffer, or a timeout can also cause LGWR to write entries from the log buffer to the redo log files.

Log Force at Commit

A commit:

- **Signals a completed transaction**
- **Forces redo entries to disk**
  Group commits
- **Does not write data blocks**
  No data block force at commit

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Log Force at Commit

During the commit action an SCN is allocated and a commit record is generated (ktucm). The commit record is formed in the log buffer. To make the committed change permanent, all the redo entries up to and including the commit record must be written to the online logs (on disk). Oracle will optimize the writing of redo such that any transactions which complete during the signaling of a write will be written at the same time (this is termed group, or piggyback, commits).

Recovery is designed to support the Durability property of transactions. It can achieve this by having to write only the redo entries at commit time. It is not necessary to write all the changed datablocks at this time, which is what is meant by the phrase “no data block force at commit”.

Online Log Switching

- If there is not enough space in the current log, a suitable redo log among the other online logs is found.
- The suitable redo should be:
  - Not active
  - Checkpoint complete
  - Archiving not required

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Online Log Switching

If there is insufficient space in the current online log in which to write redo, it is necessary to perform a log switch. In order to do this, a candidate must be found amongst the other online logs. Two doubly - linked lists are visible in the control file linking the online logs with respect to the next available log, and the next log requiring archiving (If the database is operating in Archivelog mode).

The next log to be used must not contain changes required in the event of instance recovery is needed. The current thread checkpoint must be beyond that log file. In other words, the thread checkpoint SCN must be beyond the highest SCN allocated in the log to be switched into. The message “checkpoint not complete” in the alert.log will indicate that LGWR is waiting to switch into an active logfile. You should look at increasing the number of logfiles, the size of the logfiles, or the frequency of checkpointing to alleviate this issue.

If the database is operating in Archivelog mode then the next log to be used must have been archived before it is being switched into. The ARC0 process may be posted to archive the log file if this is not the case.

Checkpoints

- A checkpoint is a point (SCN) in the redo stream before which all blocks that were dirtied at or before that time have been written to disk.
- Checkpoints occur in several contexts:
  - Thread Checkpoint (OPS)
  - Database Checkpoint
  - Datafile Checkpoint
  - Incremental Checkpoint
  - Mini Checkpoint (DDL based)
- Checkpoints are used to determine where recovery should start.

2-30 Confidential

Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Checkpoints

Checkpoints can be triggered by a number of events (log switch, shutdown, thread close, hot backup), which lead to different types of checkpoints:

- Database checkpoint: The Thread Checkpoint that has the lowest checkpoint SCN for all enabled threads becomes the database checkpoint. All blocks in memory that contain changes made prior to this SCN across all instances must be written out to disk. For a single instance database, a database checkpoint is the same as a thread checkpoint.
- Thread checkpoint: All blocks dirtied prior to the checkpoint SCN in that thread for all online data files will be written to disk. This occurs each time a local checkpoint is initiated by the instance (e.g. log switch).
- Datafile checkpoint: All blocks changed prior to the datafile checkpoint SCN have been written to disk. Database checkpoints, hot backups, or taking a tablespace offline can signal datafile checkpoints.
- Incremental checkpoints do not update file headers. They are used to reduce recovery time in case of failure.
- Some DDL, such as drop table, will cause mini-checkpoints to occur. This affects only the blocks belonging to the object covered by the statement.

Checkpoint information is extremely useful in determining where to commence recovery. This topic is covered in detail when recovery algorithms are discussed.

Normal Checkpoint Mechanics

- Get current RBA from the log buffers (completion RBA).
- Mark dirty block buffers in cache below this RBA to be written out.
- Post DBW0.
- DBW0 writes all marked buffers to disk.
- Update datafile headers and control file.

2-31 Confidential

Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Normal Checkpoint Mechanics

A normal checkpoint occurs at a log switch or when triggered by initialization parameters that govern checkpoint intervals. When a normal checkpoint is signaled the information for the checkpoint structure is gathered (current RBA for log buffers, current SCN, and so on). The process that is performing the checkpoint (CKPT, LGWR, or a server process) scans the buffer cache LRUW list a certain number of buffers at a time. It marks all dirty buffers which have a high RBA below the current checkpoint RBA. It then posts DBW0 to write them out to disk. DBW0 will flush only a certain number of buffers at a time.

DBW0 does not necessarily write the dirty buffers as soon as they are marked for write by the checkpoint process. (The DBW0 algorithms are discussed later in the course) The inclusion of multiple LRU lists and ordering the checkpoint list by RBA will allow for a cleaner flushing of dirty buffers.

If the next checkpoint starts before DBW0 has finished flushing all dirty buffers to disk, we cannot start a new checkpoint. If a new checkpoint were to start while another was ongoing, a new RBA would be chosen but there would be no way of determining which dirty buffers belong to which checkpoint, because the blocks not flushed out from the previous checkpoint would also have an RBA lower than the current checkpoint RBA.

A fast checkpoint occurs on a database shutdown, tablespace backup, and alter system checkpoint. In a fast checkpoint the dirty buffers are written out immediately.

Incremental Checkpoint

- Continuously active checkpoint
- No completion RBA
- Checkpoint advanced in memory only
- RBA for incremental checkpoint recorded in control file
- DBW0 writes out dirty buffers to advance the incremental checkpoint
- Used to reduce recovery time after a failure

2-32 Confidential

Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Incremental Checkpoint

An incremental checkpoint is a continuously active checkpoint. DBW0 periodically writes out buffers from the checkpoint queues to advance the incremental checkpoint in memory. Periodically, the CKPT process scans all the checkpoint queues, determines the lowest low RBA of all the buffers at the heads of the checkpoint queues, and writes out this RBA to a thread-private location in the control file. These disjointed activities of writing buffers in ascending low RBA order and periodically recording the lowest low RBA in the control file constitute incremental checkpointing.

Recovery chooses the higher of the incremental checkpoint RBA and the thread checkpoint RBA as the starting position for recovery.

Incremental Checkpoints

There are five criteria that drive incremental checkpoint writes:

- **Upper bound on recovery reads**
- **Size of the smallest log file**
- **Value of log_checkpoint_interval**
- **Value of log_checkpoint_timeout**
- **Total number of dirty buffers in the cache**

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Incremental Checkpoints (continued)

The `fast_start_target_io` parameter can be used to put an upper bound on recovery reads. To avoid recovery stalling during a log switch, an incremental checkpoint is triggered when an amount of redo equal to 90% of the size of the smallest log file has been generated. `log_checkpoint_interval` is based on blocks and `log_checkpoint_timeout` is based on time. If all four of these limits are specified, the most aggressive in setting the target RBA for incremental checkpoints is chosen.

The final criterion is based on the parameter `db_block_max_dirty_target`.

The more aggressive the incremental checkpoint, the more likely you are to encounter ‘write complete waits’ wait events. A hot, diry buffer may also be near the head of the checkpoint queues. Writing such buffers can potentially cause ‘write complete waits’. However, with more frequent incremental checkpoints DBW0 has to perform less scanning of the buffer cache, so overall performance should not be affected.

Checkpoint Structure

A checkpoint structure includes the following information:

- **Checkpoint SCN**
- **Time that the SCN was allocated**
- **Thread that issued the checkpoint**
- **RBA where the SCN was allocated**
- **Enabled thread bit vector (bitvec)**

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Checkpoint Structure

The checkpoint structure consists of several elements - the key elements being the Checkpoint SCN, and an enabled thread bit vector (bitvec).

The checkpoint SCN effectively demarcates a specific location in each enabled thread. For each thread, this location is where redo was being generated at some point in time within the resolution of one commit. The redo record headers in the log can be scanned to find the first redo record that was allocated at the checkpoint SCN or higher.

The enabled thread bitvec is a mask defining which threads were enabled at the time the checkpoint SCN was allocated. Note that a bit is set for each thread that was enabled, regardless of whether it was open or closed. Every thread that was enabled has a redo log that contains the checkpoint SCN. A log containing this SCN is guaranteed to exist, either online or archived.

The checkpoint structure also stores the time that the checkpoint SCN was allocated. This timestamp is only used to print a message to aid a person looking for a log. In addition, the checkpoint structure stores the number of the thread that allocated the checkpoint SCN and the current RBA in that thread when the checkpoint SCN was allocated. Having an explicitly stored thread RBA (as opposed to having only the checkpoint SCN as an implicit thread location pointer) makes the log sequence number (part of the RBA) and archived log name readily available for the single-instance case.

A checkpoint structure for a port that supports up to 1023 threads of redo is 150 bytes long. A VMS checkpoint is 30 bytes and supports up to 63 threads of redo.

Applying Redo

Oracle provides three types of recovery mechanisms:

- **Block level recovery**
- **Thread recovery**
- **Media recovery**

2-35 Confidential

Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Applying Redo

Three types of recovery mechanisms are provided:

1. Block level recovery is the simplest form of recovery. It is performed automatically by Oracle when a process that is changing a buffer dies or if a corrupted block is detected in the buffer cache.
2. Thread recovery is also performed automatically by Oracle when it detects the failure of an instance. This can be termed instance recovery or crash recovery, depending on whether one or more instances are involved, or all instances are involved. In single-instance (exclusive mode) database the terms crash and instance recovery are inter-changeable; in a parallel server database, crash recovery implies all instances have crashed whereas instance recovery requires that some instances survive.
3. Media recovery is performed in response to a recover command. It is used to make a backup more current, and and therefore is not an automatic process.

These three recovery procedures will be discussed in more detail later in the course.

Applying Redo

- Rolls data blocks forward by reapplying the change vectors in the redo logs to the correct version of the block.
- Block and instance recovery use only the online logs.
- Media recovery may require archived logs.
- Checkpoint structures are used to determine at what SCN to start recovery.

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Applying Redo (continued)

Instance and media recovery are covered later in the course.

Summary

In this lesson, you should have learned about the following:

- ACID properties
- Physiological Logging
- Redo Structures
- Recovery Algorithms
- Applying Redo

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

References, Acknowledgments, and Further Reading

- *Oracle 7 Internals - Module 2* by Lawrence To, Oracle Internal Publication, 1996.
- Oracle Source Code - version 8.
- *Oracle7 v7.2 Recovery Outline* by Andrea Borr and Bill Bridge, UK Web IV Note: 41391.1
- *Design Specification for Bounded Recovery Time, RDBMS 8.1* by Bill Bridge and Ashok Joshi, Feb 15. 1998.
- *Transaction Processing: Concepts and Techniques by* Jim Gray and Andreas Reuter
- Recovery through Resetlogs feature (Oracle 7.3.3) WEB IV Note: 50451.1

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

3

Internal Recovery Structures

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Schedule:	Timing	Topic
	90 minutes	Lecture
	60 minutes	Lab
	150 minutes	Total

Objectives

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

- Develop an approach to recovery
- Perform Control File Dump Analysis
- Perform File header Dump Analysis
  - Data file
  - Redo log
- Use V$ Views during recovery

3-2

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Objectives

As mentioned in the previous lesson there are many fundamental recovery structures involved in guaranteeing the ACID properties of the database. This lesson covers about those that are specifically associated with durability and recovery guarantee.

Approaching Recovery Issues

- Gather diagnostic information.
- Develop an understanding of the situation.
- Confirm your diagnosis.
- Construct a list of recovery options.
- Convey recovery options to customer.

3-3 Confidential

Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Approaching Recovery Issues

Diagnostic information can come from the alert log, trace file, V$ views, file header dumps and customer statements. Be careful when collecting information from V$ views, especially if a backup control file is used for recovery. Know where the information in the V$ view is obtained from.

From the gathered evidence, try to picture what happened before you became involved in the problem. If the gathered evidence disagrees with the customer's story, the evidence wins! (assuming it was gathered correctly)

Sometimes it helps to give the customer a choice in how to recover database, but try to persuade customer to choose the most preferable option. The fastest way to open the database may not be the best recovery option (or the quickest in recovering completely).

Let the customer make the final decision; after all, they understand their business needs the best. Just make sure they understand the consequences of their decision.

Diagnostic Trace Files

- Three dump files:
  - Control file
  - Data file headers
  - Log file headers
- Dumps are performed by setting an event.
- Each dump has different levels: The level determines how much information is listed in the trace file.
- Trace file are created in user_dump_dest.

3-4 Confidential

Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Diagnostic Trace Files

Connected Internal or as SYS

- Control file dump

```
alter session set events  'immediate trace name controlf level N';
```

- Datafile header dump

```
alter session set events  'immediate trace name file_hdrs level N';
```

- Redo log header dump

```
alter session set events  'immediate trace name redohdr level N';
```

Taking Dumps

- To dump the information in control file or file headers
- Database must be mounted or open
- Dump level (1–10) specifies the amount of information dumped
- Trace files (the dumps) are created in user_dump_dest

Generic Dump Information

- Every dump contains generic file header information under the heading 'FILE HEADER:' (structure kccfh).
- Useful information found here:
  - Database ID and Database Name
  - Control File Sequence number
  - File size (in Oracle blocks) and Oracle block size
  - File number
  - File type

3-5 Confidential

Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Generic Dump Information

Example

```
FILE HEADER:
   Software vsn=135266304=0x8100000, Compatibility
    Vsn=135217728=0x8000000
    Db Id=1120036664=0x42c26738, Db Name='DB02'
    Control Seq=808=0x328, File size=808=0x328
    File Number=0, Blksiz=4096, File Type=1 CONTROL
```

The database ID and name can be used to verify that the file belongs to this database. The control file sequence number shows the last time the file header was updated. These values can also be obtained from V$DATABASE.

File number is relevant only for datafiles and redo logs. For control files, it is always 0.

File Type: 1= Control file

2 = Redo log

3 = Datafile

4 = Backup control file

6 = Temp file

Control File Dump

- The first part of the dump is the generic file header.
- Database related information:

DATABASE ENTRY	x$kccdi
CHECKPOINT PROGRESS RECORDS	x$kcccp
REDO THREAD RECORDS	x$kccrt
LOG FILE RECORDS	x$kccle
DATAFILE RECORDS	x$kccfe
TEMP FILE RECORDS	x$kcctf
TABLESPACE RECORDS	x$kccts
LOG FILE HISTORY RECORDS	x$kcclh
OFFLINE RANGE RECORDS	x$kccor
ARCHIVED LOG RECORDS	x$kccal
RMAN RECORDS	

3-6

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Control File Dump

To dump a control file, you need to issue this command:

```
ALTER SESSION SET EVENTS 'IMMEDIATE TRACE NAME controlf LEVEL 10';
```

We will only highlight significant components in the dumps that relate to recovery.

Control file and datafile dumps tend to reveal a lot of information. The purpose of the following slides is to describe the wealth of information available in the dumps which could help you debug recovery problems.

The x$tables are the fixed tables where the information in each of these sections comes from. Some V$ views are created on top of the X$ tables, such as V$THREAD, V$LOG, and V$OFFLINE_RANGE.

The amount of information dumped depends on the level specified. The information dumped at each level will be discussed in the next chapter.

Control File Dump

RMAN related information:

- BACKUP SET RECORDS x$kccbs
- BACKUP PIECE RECORDS x$kccbp
- BACKUP DATAFILE RECORDS x$kccbf
- BACKUP LOG RECORDS x$kccbl
- DATAFILE COPY RECORDS x$kccdc
- BACKUP DATAFILE CORRUPTION RECORDS x$kccfc
- DATAFILE COPY CORRUPTION RECORDS x$kcccc
- DELETION RECORDS x$kccdl
- PROXY COPY RECORDS x$kccpd

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Control File Dump (continued)

Each of these sections has a V$ view associated with it.

Control File Layout

Oracle 7	Oracle8*i*
Generic File Header	Generic File Header
Database Entry	Database Entry
Checkpoint Records	Checkpoint Records
Redo Thread Records	Redo Thread Records
Log File Records	Log File Records
Datafile Records	Datafile Records
Log File History Records	Temp File Records
	Tablespace Records
	Log File History Records
	Offline Range Records
	Archived Log Records
	RMAN Information
Static Size	**Dynamic Size**

3-8

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Control File Layout

In Oracle7, the file header, database entry and checkpoint structure were all of a fixed size. MAXINSTANCES controlled the size of the redo thread records area, MAXLOGFILES controlled the size of the log file record area, MAXDATAFILES controlled the size of the datafile record area, and MAXLOGHISTORY determined the size of the log history area.

In Oracle8 and Oracle8*i* the control file size is variable and areas such as the datafile records area can be extended as needed. The V$CONTROLFILE_RECORD_SECTION view can be used to view how many records are in each section and how many bytes each section occupies.

Control File Dump
Reuse Record Section

Non-circular reuse record section:

- **Checkpoint progress records**
- **Redo thread records**
- **Log file records**
- **Datafile records**
- **Temp file records**
- **Tablespace records**

3-9 Confidential

Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Control File Dump Reuse Record Section

Two classes of records are maintained within the control file:

- Circular reuse records
- Non-circular reuse records

Non-circular reuse records have an identifying record-index, which does not change. The records in these sections may be overwritten, and the records contained within each section always reflect the most current state of the database (as long as the control file is not a backup).

Reuse of a record in this class is done by logical deletion and re-use. For example, you can drop a data file and add a data file to the database, and the data file record area can be reused. In the Log File records, when a log switch occurs, the previous control file entry for that log is overwritten.

Prior to Oracle8, the amount of space in the control file allocated to each section was determined by control file parameters:

- MAXLOGFILES
- MAXLOGMEMBERS
- MAXDATAFILES

Control File Dump Reuse Record Section

Circular reuse record section:

- Log history records
- Offline range record
- Archived log records
- Backup set records (RMAN)
- Backup datafile records (RMAN)
- Backup log records (RMAN)
- Data file copy records (RMAN)
- Backup data file corruption records (RMAN)
- Data file copy corruption records (RMAN)
- Deletion records (RMAN)
- Proxy copy records (RMAN)

3-10 Confidential

Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Control File Dump Reuse Record Section (continued)

Circular reuse records are uniquely identified by a *recid*. Recid is a monotonically increasing sequence number that uniquely identifies the record, even if the record is relocated to a new record-index (entry slot). Reuse of a record in this class is based on the user-supplied timestamp and space requirements. The entries are overwritten in the same order they were created. Each section contains historical data.

The CONTROL_FILE_RECORD_KEEP_TIME parameter specifies the minimum age in days that a record must be stored before it can be reused. Default value is 7 days.

For example, RMAN records information about data file copies in the control file. The system data file is backed up every day, starting with 15-Mar-99. After the 22nd of March, the datafile copy entry for 15-Mar-99 would be overwritten. In this way, the control file keeps all data file copy records dating back one week.

Control File Dump

```
alter session set events 'immediate trace
name controlf level <n>';
```

- **Level 1**

  Generic file header
- **Level 2**

  Level 1 + database information + checkpoint progress records
- **Level 3 or higher**

  Level 2 + reuse record sections

3-11

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Control File Dump

Every dump contains a generic file header (level 1). Additional levels give additional information.

Level 10 is most commonly used. Currently it provides the same information as a level 3 dump.

Normally the highest level is used in recovery situations to get all possible information up front. Lower levels might be used to reduce the size of the trace files being produced.

The next slides discuss some of the structures in the control file. See Appendix for sample dump.

Control File: File Header

- Seq#
  - Incremented each time the control file header is updated.
  - Identifies the version of the control file.
- DB Id
  - Uniquely identifies the database.
  - Set at creation time.
  - Combines DB name, length and date.
  - Written in every file header of the database.

3-12

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Control File: File Header

The dump of the control file starts with the Seq#, which is a number that is incremented each time the control file header is updated. It can be considered as indicating the version of the control file. This number is stored in the generic file header, so each file has it stored in its header.

The Database Id uniquely identifies the database (even when database names are the same). It is fixed during creation time and is a combination of db name, length, and date. At database creation time, we call, `kccmdi`, to create the `db` id. The `db` id is never modified even when the database name is changed. The `db` id is written in every Oracle file header and is primarily used to confirm referenced files are truly members of the same database environment.

Control File: Database Entry

- Version
  - Creation version
  - Compatibility version
- Database creation date
- Database Flags

  Indicate current state of the database.

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Control File: Database Entry

In the sample dump output one can see that the database was created with version 8.1 and is compatible with version 8.0. Compatible represents the lowest version which this database can read and apply change vectors and redo for recovery. More specifically some new features of the current version involve operations that will produce change vectors containing redo opcodes not contained in older versions. This makes it impossible to use an older version of Oracle (below the compatibility version) for media recovery. Database Flags (from kcc3.h)

```
#define KCCDIMRE 0x0001  /* whether media recovery enabled */
#define KCCDICKD 0x0002  /* if dictionary must be checked with control file */
#define KCCDIRLR 0x0004  /* DB OPEN RESETLOGS required */
#define KCCDIJNK  0x0008  /* junk value from beta - do not reuse this flag */
#define KCCDIMRC 0x0010  /* was/is last mounted READ_COMPATIBLE */
#define KCCDICNV  0x0020 /* control file was just created by convert from v6 */
#define KCCDIIRA   0x0040  /* Incomplete Recovery Allowed when resetting logs */
#define KCCDICCF  0x0100  /* Controlfile was created with CREATE CONTROLFILE */
#define KCCDIINV  0x0200   /* Invalid control file or database - still creating */
#define KCCDISBD 0x0400   /* StandBy Database - control file for hot standby */
#define KCCDIORL 0x0800   /* Opened ResetLogs - set until dictionary check */
#define KCCDICFC 0x1000   /* valid ControlFile Checkpoint in backup cf */
#define KCCDISSN 0x2000   /* SnapShot controlfile fileName pointer valid */
#define KCCDIUCD 0x4000   /* lazy file header Update Checkpoint cycle Done */
#define KCCDICLO 0x8000    /* clone database */
```

Control File: Database Entry

- Incomplete recovery SCN
- Resetlogs SCN
- Resetlogs timestamp or count
- Database checkpoint
  - Thread
  - SCN

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

Control File: Database Entry (continued)

Incomplete recovery scn: The last SCN applied during incomplete db recovery. If recovery needs to be restarted, it will start from this incomplete recovery SCN. This is cleared when database is opened.

Resetlogs SCN: Uniquely identifies each execution of a RESETLOG open. This is compared with File Headers and Log Files Headers when database is opened.

Resetlogs timestamp or count: Timestamp of the last resetlogs. Redo logs cannot be applied by recovery if its resetlogs data does not match

Resetlogs characteristics: Datafile cannot be recovered or accessed if its resetlogs data does not match that of the database information record of the control files. Exceptions include read-only and offline normal files in which case, the STOP SCN is compared. This ensures that changes discarded by resetlogs do not get back into the database. Resetlogs and recovery past resetlogs are discussed later in this course.

The resetlogs information can also be viewed through V$DATABASE.

Database checkpoint structure: Thread checkpoint of the thread that has the lowest checkpoint SCN of all open threads.

thread=thread owning the database checkpoint

If threads are open, the database checkpoint is the lowest thread checkpoint. If all threads are closed, the database checkpoint is the highest thread checkpoint.

scn=scn of the checkpoint structure

Redo created prior to the database checkpoint SCN in all instances has been written to all online datafiles. Online datafiles are checkpointed at the database checkpoint. All changes prior to the checkpoint SCN are written to disk prior to updating headers. Use V$DATABASE to view the database checkpoint.

Control File: Database Entry

- Threads
  - Enabled
  - Open
  - Head
  - Tail
- Enabled threads bitvec
- Force SCN

3-15 Confidential

Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Control File: Database Entry (continued)

Database Entry

Threads describe how many enabled, opened threads are present. Thread status is available in V$THREAD.

- Enabled = the number of enabled threads
- Open = the number of open threads
- Head = head of thread link list
- Tail = tail of thread link list

Enabled thread bitvec

1 = enabled

0 = closed/disabled

Usually has a max of 64 but the SSTNTHRD value is port specific. MPP has more allocated to it. BITVEC[SSTNTHRD+1]

Force Archive SCN or Force SCN

System strives to archive any log (from any enabled thread) that contains low SCN <= Force SCN. This helps ensure that all redo logs necessary for the recovery of a hot backup are archived and is essential for standby databases and parallel server. It is updated at log switch as follows:

Prev SCN of current log = FORCE SCN

The force SCN is used to force log switches in idle enabled threads; thereby, the log can be archived. Archiver can not archive current logs. This will be discussed in more detail in the archiving section.

This is from kccdifas in kcc.c.

Control File: Database Entry

Control File checkpoint SCN:

- **Updated only when database is open**
- **Updated after *every* checkpoint**
- **Always > = database checkpoint**

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Control File: Database Entry (continued)

The Control File Checkpoint SCN is updated after every checkpoint (may it be tablespace, thread, or database) and is always greater than or equal to the database checkpoint when the database is open.

Control File Dump

- End of the datafile entry section in the control file.
- Next slide starts with the checkpoint progress records.

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Control File Dump

This slide is here for demarcation of the various sections in the control file dump.

Control File: Chkpt Progress

Checkpoint progress records:

- **Record contains information about the buffer cache of the owning instance.**
- **Not read or written via control file transactions.**
- **Each record is in its own control file block:**
  - **Not versioned via the bitmap**
  - **Does not have a shadow block**
- **Ownership of open thread enqueue grants write access to the checkpoint progress record for that thread.**

  Therefore there is no contention other than for the disk itself.

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

Control File: Chkpt Progress

Changes to the control file are managed through *shadow block* structures. Copies of each block in the control file are stored in the same file with a flag to indicate which block copy is the current copy. With this method, Oracle is able to detect if a control file is out of synch with the other control files. The shadow block also serves as rollback for the control file.

Control File: Chkpt Progress

- Status
- Flags
- Dirty
- Low cache RBA
- on disk RBA
- on disk SCN
- Resetlogs SCN
- Heartbeat and MountID

3-19 Confidential

Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Control File: Chkpt Progress (continued)

If there are multiple threads for the database (OPS), then there will be multiple checkpoint progress records in the control file.

Status is set when the record is written and may not reflect the current state. It is used to determine how useful the rest of the record is.

0x00 - invalid, 0x01 - thread closed, 0x02 - thread open and can be used for recovery

The dirty count is the length of the checkpoint queue. (# buffers in the cache dirtied by redo)

Low cache RBA determines where recovery needs to start for the thread

All changes prior to this RBA have been written to disk

On disk RBA is the highest RBA on disk. During recovery, all redo prior to this RBA must be applied

The on disk RBA is a high RBA so it points just beyond the redo generated. The SCN at the on disk RBA is also needed in case the on disk RBA points to a block that was never written. In this situation the on disk SCN will become the high SCN of the thread.

Thread heartbeat updates are done in CKPT's timeout action (every 3 seconds). A changing heartbeat indicates that the thread is mounted by a live instance; this property is used to validate the sanity of the distributed lockmgr (DLM) environment when a new instance attempts to mount the database.

From kcc.h.

Control File Dump

- End of the checkpoint progress section in the control file.
- Next slide starts with the redo threads.

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

Control File: Redo Thread

- **Thread status:**
  - **Enabled threads are seen as active.**
  - **All active threads are subject to archiving.**
- **Thread links forward and back**

  Links all open threads

3-21 Confidential

Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Control File: Redo Thread

Status of the thread:

- 0x01 - Thread is open
- 0x02 - Thread is enabled
- 0x04 - Thread is enabled publicly

For example, 0x07 implies a publicly enabled, opened thread

Oracle considers an enabled thread to be active. All enabled threads are available for recovery. At times, a log switch may need to be forced of an enabled but not opened thread to archive the log.

Enabling a thread is required before it can be mounted. If the thread is enabled publicly then it may be chosen by any instance that did not specify a thread to mount. The thread must not be enabled already, and must have two online logs. The database must be open and the caller must hold the instance state enqueue. Note that this call generates a redo record.

Disabling a thread prevents any instance from mounting it and takes it out from the enabled bit vector. This allows all the logs to be deleted after archiving. A thread may be disabled only if it is not mounted and is closed. The database does not need to be open.

Thread links:

- Forward: Points to the next thread on the open thread linked list (thread#)
- Back: Points to the previous thread on the open thread link list

Control File: Redo Thread

- Log linked list
  First, last, current
- Enabled at SCN, disabled at SCN
- Checkpoint structure
- Enabled threads bitvec

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Control File: Redo Thread (continued)

Online redo log link list

- First: log file link list head
- Last: log file link list tail
- Current: current log in the thread

The list of a thread's log file records is anchored in the thread record (that is, using head and tail log file record indices) and linked through the log file records, each of which stores the thread number. The log file record also has fields identifying the number of group members, as well as the head and tail (that is, the file name record indices) of the list (linked through file name records).

last used seq# = The last sequence number allocated to this thread (current log seq#)

Thread Checkpoint Structure

- SCN: Checkpoint SCN for the thread
- Time: Time at which the SCN was allocated for the checkpoint
- Thread: Thread number
- RBA: Redo byte address for this thread

Enabled Thread Bit Vector

- Bit vector of all enabled threads
- Needed for recovery and archiving

Control File: Redo Thread

Thread Checkpoint

- **Forces to disk all dirty buffers in that instance that contain changes before a designated SCN (thread checkpoint SCN).**
- **Updated each time an instance checkpoints its thread.**
- **Instance recovery starts applying at the last thread checkpoint for that instance.**

3-23 Confidential

Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Control File: Redo Thread (continued)

Other information in the control file dump:

- opened at The time stamp at which this thread was last opened
- by instance The SID of the last instance to open this thread

The checkpoint structure demarcates the point in time (redo byte address and database SCN) in which all changes are guaranteed to be on disk. If the file is online and changeable, then the file is still online fuzzy. Recovery will start from the last successful checkpoint.

Control File Dump

- End of the redo threads section in the control file.
- Next slide starts with the log file records.

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

Control File: Log File Records

- Log File Entries:
  - Log file number
  - File name
  - Multiple file names for mirrored log groups
- Redo Log links:
  - Forward
  - Backward
- Log Sequence number

3-25 Confidential

Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Control File: Log File Records

Log File Entries

- Log File # : log file number (If it has been deleted, it is marked <*deleted*>)
- Thread: Thread number of this log

Redo Log Links

- Forward: The number of the next valid online redo log
- Backward: The number of the previous valid online redo log

For each log file, there is a control file record that describes it. The index of a log's control file record is referred to as its log number. Note that log numbers are equivalent to log group numbers, and are globally unique across all threads).

The forward pointer usually points to the newly created log. The order of the switching from online redo log to online redo log is described by the redo log links. Although it is usually the same order (for example, 1,2,3,4), it can change. The next log needs to satisfy the following conditions:

Database checkpoint SCN >= next SCN

- Has been archived or spooled if archivelog mode = TRUE
- Not being cleared (will discuss this later).

At times, the redo log link may change its pattern (for example, 1,2,3,1,4,2,3,..) and so on.

Seq (log sequence number), along with thread #, makes the file unique.

Control File: Log File Records

- Log file flags
- Archive links (not used)
- SCNs
  - Previous
  - Low
  - Next

3-26 Confidential

Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Control File: Log File Records (continued)

Log File Records

flg = Log file flag

- 0X0001 : Set if the log file is spooled or archived
- 0X0002 : Set if no more allocation can be done in the log
- 0X0004 : USE ME flag which indicates to use it to be the next current log (brand new log)
- 0X0008 : Current Log
- 0X0010 : Set if log is being cleared

cleared logfile = Inactive online logfile is discarded and reinitialized

There is a command that allows you to clear an inactive online redo log. You can even clear a log that has not been archived. You may chose to clear instead of taking an outage to turn off archivelog and dropping the inactive log.

Why would you need to clear a log?

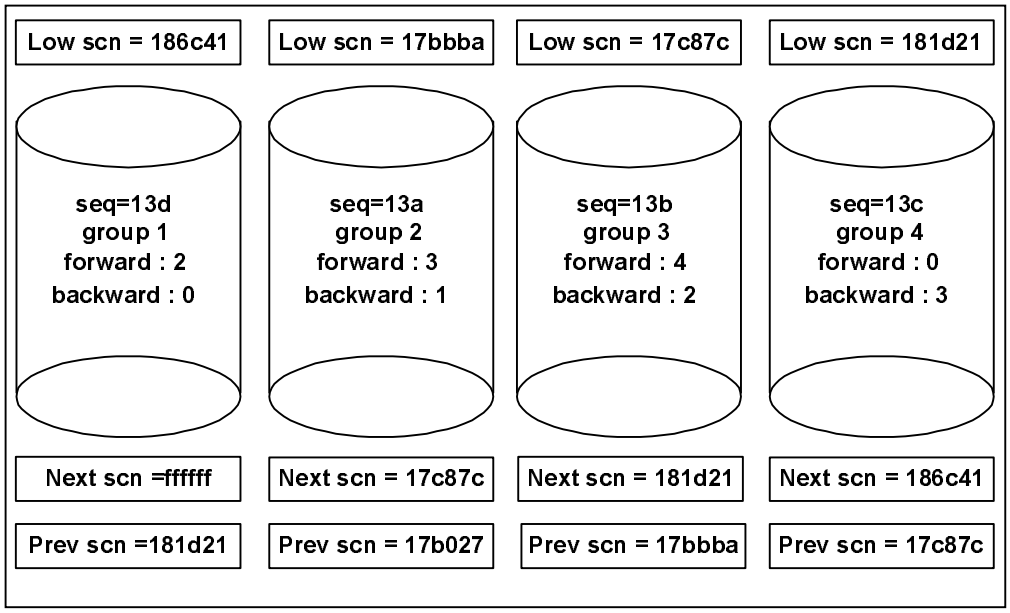
- Inactive redo log (not a current log) is corrupted due to media failure
- Archiver cannot archive redo log because corruption

Prev SCN = low SCN of the previous log

Low SCN = when switched into, the first SCN allocated to the online redo log.

redo in this log will have SCN > low SCN

Control File: Log File Records



3-27

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Control File: Log File Records (continued)

Next SCN = The "high" SCN of this log

- redo in this log will have SCN < Next SCN
- Next SCN = Low SCN of the next log
- ffff.ffffffff indicates the current online log

All SCNs include a timestamp. Low and Next SCN describe the non-inclusive endpoints of each log.

Try to answer the following questions:

Which log is current?

Follow the redo log link from oldest to current log:

- Which log will be the next current log?
- What will be the sequence number of the next log?
- What is the lowest SCN that you can recover from using the online logs?

Refer to the dump if you need help.

Answers:

1, 2-3-4-1, 2, 13e or 318, 17bbba

Control File: Log File Records

- Which log file is current?
- Has either log group been archived?

```
LOG FILE #1:
  (name #3) /proj/p97052/ITSRMAN/redo1a.rdo
  (name #4) /proj/p97052/ITSRMAN/redo1b.rdo
 Thread 1 redo log links: forward: 2 backward: 0
 siz: 0x4800 seq: 0x0000013d hws: 0x5 bsz: 512 nab:
0xffffffff flg: 0x8 dup: 2
 Archive links: fwrd: 0 back: 0 Prev scn: 0x0000.00181d21
 Low scn: 0x0000.00186c41 04/14/98 14:24:06
 Next scn: 0xffff.ffffffff 02/28/98 05:44:01
LOG FILE #2:
  (name #5) /proj/p97052/ITSRMAN/redo2a.rdo
  (name #6) /proj/p97052/ITSRMAN/redo2b.rdo
 Thread 1 redo log links: forward: 3 backward: 1
 siz: 0x4800 seq: 0x0000013a hws: 0xc bsz: 512 nab: 0x4801
flg: 0x1 dup: 2
 Archive links: fwrd: 0 back: 0 Prev scn: 0x0000.0017b027
 Low scn: 0x0000.0017bbba 02/28/98 05:44:01
 Next scn: 0x0000.0017c87c 02/28/98 06:30:47
```

3-28 Confidential

Copyright © Oracle Corporation, 2000. All rights reserved.

Control File: Log File Records (continued)

Log file 1 is the current log as indicated by the Next scn value 0xffff.ffffffff and the flg value of 0x8.

Since the flg for Log File #2 is set to 0x1, this shows that the log group has been archived.

Control File Dump

- End of the Log file section in the Control file.
- Next slide starts with the Datafile records.

3-29

Confidential

Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Control File Dump

This slide is used for demarcation of the various sections in the control file.

Control File: Data File Records

- File name
- Size at creation
- Block size
- Status

3-30 Confidential

Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Control File: Data File Records

Status:

0x0002 file is ONLine

0x0004 ReaDing is Enabled

0x0008 ChanGing is Enabled

0x0010 Media Recovery Required

0x0020 Generate End hot backup Marker at next open

0x0040 File record generated by check dictionary

0x0080 Save Offline scn Range at next checkpoint

0x0100 Renamed Missing File

0x0200 Generate Off-line Immediate marker

0x0400 Checkpoint by instance where UnVerified

0x0800 offline to be DRoPped

0x1000 Was at Clean Chkpt: clean up to ctrlfile chkpt

0x2000 Online at Dictionary Check if read/only tblspc

0x4000 entry created by DBMS_BACKUP_RESTORE

Example : 0xe = 0x8 + 0x4 + 0x2

Change/Write + Read + Online—>normal status of open files.

In Oracle7, the flag 0x0001 was also used to indicate a SYSTEM tablespace data file.

Control File: Data File Records

- Tablespace number
- prev_file
- Checkpoint Counter
- Checkpoint SCN
- Stop SCN
- Offline SCN
- Online SCN

3-31 Confidential

Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Control File: Data File Records (continued)

prev_file points to the file number of the previous file in the same tablespace.

Checkpoint count

- Number of checkpoints to online files
- Used to detect backup data files or control files
- Incremented every time checkpoints of the online files are being advanced

The old value of the checkpoint counter is also remembered in the file header. It is usually one less than the current counter in the header. The old checkpoint counter value may differ from the current counter by more than one if the previous data file checkpoint counter update failed after the file header was written but before the control file *transaction* committed.

Checkpoint SCN

- Last successful checkpoint SCN of online files
- All dirty buffers prior to SCN have been written to disk

Stop SCN indicates that no more redo is needed after this SCN:

- Infinite for online, changeable files
- Infinite when checkpoint was not completed
- During shutdown (shutdown abort)

Control File: Data File Records

Offline SCN Range

- **Offline SCN (kccfeofs)**
- **Online checkpoint SCN (kccfeonc)**
- **No redo generated for the file during this interval**
- **Recovery optimization**

3-32

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Control File: Data File Records (continued)

Offline SCN: SCN where a file was cleanly offline normal or made read-only set equal to the clean STOP SCN saved when setting the file offline or read-only

- Used to optimize recovery
- This is the range of SCNs that can be ignored during recovery for this file because it was offline during that time.

Online checkpoint SCN:

- SCN when a file was online from a previous offline
- SCN when a read-only file was toggled to read-write

This file is not checked when recovering the range of redo between the offline and online SCNs.

Control File Dump

- End of the Datafile records section in the control file.
- Next slide starts with the Temp file records.

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

Control File: Temp File Records

- Based on format of data file records
- Does not contain recovery related information
- File name
- Creation size and block size
- Status
- Tablespace number
- krfil: file's tablespace relative file number
- prev_file

3-34 Confidential

Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Control File: Temp File Records

Temporary files have file numbers ranging from 1 to *db_files*, which is calculated separately from the file number for a permanent datafile. In the SGA file array, permanent files use numbers from 1 to db_files, and temp files from (db_files + *n*) to (2 * db_files), where *n* is the absolute file number of the temporary file.

Temporary files are not stored as normal datafiles in the data dictionary. There is an alternative to FILE$ (X$KCCTF), which is used to populate the view V$TEMPFILE.

The status field is different from that for permanent files:

- 0x01 - Created in primary database
- 0x02 - File is Online
- 0x04 - Reading is Enabled
- 0x08 - Changing is Enabled

prev_file refers to the previous file in the tablespace (index to a temp file number).

Control File Dump

- End of the Temp file records section in the control file.
- Next slide starts with the Tablespace records.

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

Control File: Tablespace Records

- Tablespace number
- Tablespace name
- First data file link
- Flag (rollback segments)
- TSPITR start SCN and timestamp
- Last completed TSPITR SCN and timestamp

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

Control File: Tablespace Records

The tablespace number is the same as the ts# in the data dictionary. Tablespace name and number are viewable through V$TABLESPACE.

The first data file link is a pointer to the number of the first data file defined for this tablespace. For temporary tablespaces, this points only to temp files.

The TSPITR records are used to record when tablespace point-in-time recovery operations were performed on the tablespace.

Starting in 8*i*, there is a status flag for the tablespace records:

0x01 - Temporary tablespace—contains only temp files

In Oracle8, this field was used to store a counter for the number of rollback segments in the tablespace, however that functionality was never implemented.

Control File Dump

- End of the Tablespace records section in the control file.
- Next slide starts the Log File History records.

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Control File: Log File History Records

- Earliest Log History:
  - Record
  - Thread
  - Sequence Number
  - Link
  - Low SCN
  - Next SCN
- Latest Log History

3-38 Confidential

Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Control File: Log File History Records

The entire log history can be retrieved from V$LOG_HISTORY.

Log history is extremely helpful when doing media recovery in an Oracle Parallel Server (multiple threads) environment. It helps in thread switch for locating which log (from another thread) needs to be applied next.

Earliest Log History

- Record: Slot number in the table
- Thread: Thread number
- Seq#: Sequence number of the redo (archive) log
- Link: Forward pointer to slot of the next oldest sequence number within the thread
- Low SCN: Low SCN of the archived log
- Next SCN: Next SCN of the archived log

Latest Log history

- The most recent entry
- Data has same meaning as “Earliest log history”

Control File Dump

- End of the Log history records section in the control file.
- Next slide starts the Offline Range records.

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Control File: Offline Range Records

- Records created similar to log history records.
- Earliest and latest records are highlighted.
- Contains:
  - File index
  - Link-recid
  - Offline SCN
  - Online SCN checkpoint
  - Thread and RBA

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

Control File: Offline Range Records

Records are created only when a file is taken offline and then brought back online. This information can be viewed through V$OFFLINE_RANGE.

Example:

Earliest Record

RECID #1 Recno 1 Record timestamp 12/15/99 09:54:32 File=2 Link-Recid=0

Offline scn: 0x0000.000219a6

Online checkpointed at scn: 0x0000.000219a7 11/15/99 13:00:35

thread:1 rba:(0x1.2.0)

Latest Record

RECID #14 Recno 14 Record timestamp 01/06/00 13:21:24 File=6 Link-Recid=12

Offline scn: 0x0000.000e6f0c

Online checkpointed at scn: 0x0000.000e6f37 12/20/99 13:54:51

thread:1 rba:(0x46.583.1a0)

The file index tells you which file the record pertains to. Link-recid points to the previous record in the offline range records *for that same file*. Thread and RBA tell you where in the redo stream the file was brought online.

Control File Dump

- End of the Offline range records section in the control file.
- Next slide starts the Archived Log records.

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Control File: Archived Log Records

- Records are created as in log history records.
- Earliest and Latest records are highlighted.
- Contains:
  - Thread
  - Seq#
  - Flags
  - Resetlogs SCN and timestamp
  - OS file name
  - Low and next SCN
  - Block count and block size

3-42 Confidential

Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Control File: Archived Log Records

Records are created only if archiving is enabled.

Thread refers to the thread which archived the log. Seq# is the sequence number of the archived log.

Resetlogs information is used to determine which version of the database the archived log can be applied to.

The filename stored is the full path name of where the log was created at the OS level.

The low SCN and next SCN delineate the recovery records contained in the archived log file.

The block count and block size can be used to calculate the size of the archived log. The block count can be different from the online log size. The block size is the same as the block size used for the online logs (OS dependent).

The information here can be viewed in V$ARCHIVED_LOG

Control File Dump

- Control File Dump END
- File Headers dump START

3-43

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Control File Dump

RMAN records are not covered in this course.

Data File Headers Dump

```
alter session set events 'immediate trace
name file_hdrs level <n>';
```

- Level 1

  Control file's data file entry
- Level 2

  Level 1 + generic file header
- Level 3 or greater

  Level 2 + data file header
- Level 10 most commonly used

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Data File Headers Dump

Currently level 10 is the maximum level. It provides the same output as level 3.

Datafile Header

- Control file information
- Generic file header
  - DB ID
  - File number
  - File type
- Tablespace number and name
- rel_fn

3-45

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Datafile Header

The control file information was covered previously. The information after this section comes from the header of each datafile (as opposed to coming from the control file).

The generic file header for each datafile contains the DB Id field which is used to verify that this file is associated with the correct database. The file number is the datafile number (as appears in the data dictionary) and the file type is 3 for datafile and 6 for temp files.

The tablespace number and name gives the tablespace information for this datafile.

rel_fn is the relative file number for the datafile.

Datafile Header

- Creation SCN
- Backup SCN
- Resetlogs Data
- Status
  - Defined in kcv.h
  - Different interpretation from status in control file
  - Critical during recovery

3-46

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Datafile Header (continued)

Creation SCN is when the file was created. No redo exists prior to this SCN for this file.

Backup taken at SCN is updated at BEGIN BACKUP and indicates when a hot backup started.

Resetlogs Data lists the timestamp of the most recent resetlogs and the SCN of the file after resetlogs. This information is compared with control file (database entry) information when the file is brought online.

Status Definitions (from kcv3.h) :			Version
#define	KCVFHHBP 0x01	/* hotbackup-in-progress on file (fuzzy file) */	common
#define	KCVFHHBM 0x02	/* end hot backup marker encountered */	7.3 only
#define	KCVFHOFZ 0x04	/* Online FuZzy because it was online and db open */	common
#define	KCVFHCRM 0x08	/* crash recovery marker encountered */	7.3 only
#define	KCVFHMFZ 0x10	/* Media recovery FuZzy - file in media recovery */	common
#define	KCVFHCMF 0x20	/* Clear media recovery. Fuzzy at end of recovery */	7.3 only
#define	KCVFHAFZ 0x40	/* Absolutely FuZzy - fuzzyness from file scan */	8.x only
#define	KCVFHBCP 0x100	/* Bad Checkpoint - no enabled thread bitvec */	8.x only
#define	KCVFHSTS 0x100	/* System Tablespace file */	7.3 only
#define	KCVFHFMH 0x200	/* Freshly Munged Header. resetlogs not finished */	common
#define	KCVFHXCH 0x400	/* eXternally CacHed by operating system */	common
#define	KCVFHZBA 0x800	/* Zeroed Blocks Allowed */	8.x only
#define	KCVFHPCP 0x1000	/* Proxy Copy in Progress */	8.1 only

Datafile Header

- Fuzzy file:
  - File that may have changes after its checkpoint.
  - Fuzziness extends over a range of SCNs starting from the checkpoint.
  - End of this range is the fuzzy SCN.
- Datafiles cannot be opened if a fuzzy bit is set. Crash or Media recovery will be required.

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Datafile Header (continued)

When a file is being updated by an instance that has the database open it is fuzzy because there may be some dirty buffers in the cache. Its fuzziness extends to the current SCN+1.

If the instance were to crash and a copy of the file was saved, the copy would be online fuzzy up to the time of the next crash recovery.

A hot backup is fuzzy up to the SCN associated with its end hot backup marker.

A similar marker is created after crash recovery so that it is possible to know the extent of fuzziness for backups taken after a crash. Online fuzziness is also left in a file when it goes offline immediate.

A file that is in media recovery is fuzzy because buffers are not written after every change is applied.

If a marker is encountered that indicates the end of hot backup fuzziness or online fuzziness, then the next media recovery checkpoint can clear that fuzziness. Every block in a file contains an SCN in its header. For blocks modified by redo, this is the SCN of the last change applied to the block. By looking at all the blocks in a file it is possible to determine the highest SCN in any block header. This SCN + 1 is called the absolute fuzzy scn. It is clear that there are no changes in the file that are at the absolute fuzzy SCN or greater, so it can replace any other fuzziness indicators.

Datafile Header

- The three Fuzzy bits:
  - Hot backup in Progress
  - Online fuzzy
  - Media Recovery fuzzy
- Fuzzy bits are part of the datafile status in the file header (0x01, 0x04, 0x10).

3-48 Confidential

Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Datafile Header (continued)

Three fuzzy bits are used by recovery processes:

- Hot backup in progress (bit 0x01): Applies only to hot backups made using utilities outside the kernel. This flag indicates that the file is in fuzzy state from its checkpoint until its end hot backup marker. It is set at begin hot backup. Recovery clears this flag when it crosses the end hot backup marker.
- Online fuzzy (bit 0x04): Indicates that the file is in fuzzy state from its checkpoint until the crash recovery marker. It is set when the data file is first open or onlined or made read-write. It is cleared when the data file is closed, offlined (normal) or made read-only.
- Media Recovery fuzzy (bit 0x10): This means that the file is in the process of having media recovery applied to it. It is set when media recovery is being applied. It is cleared when all the redo logs for that file have been applied.

There is a fourth fuzzy bit (Absolute fuzzy) used by RMAN: (bit 0x40)
Server managed backups (RMAN) calculate the absolute fuzziness of the backup. Absolute fuzzy SCN is the greatest of SCNs of all the blocks that are changed after the checkpoint. The absolute fuzzy flag is cleared when there is a file checkpoint that advances the SCN beyond the absolute fuzzy SCN.

Datafile Header

Other file status bits:

- **Bad checkpoint**
- **Freshly Munged Header**
- **Externally cached by operating system**
- **Zeroed blocks allowed**

3-49 Confidential

Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Datafile Header (continued)

Other bits:

- Bad checkpoint: Set when the file checkpoint does not contain an enabled thread bitvec so that the checkpoint is not used to start recovery. Happens when the checkpoint is reconstructed from an SCN in the dictionary.
- Freshly Munged header: Set when the file header is updated with the new resetlogs stamp and checkpoint. Cleared by any checkpoint. Used to allow open resetlogs to be repeated if there is a failure after writing the data file headers and before committing the control file transaction.
- Externally cached by operating system: Set when there is an external cache at the OS level that can delay writes to permanent media. Calculated when the file is brought online when the database is open and by the first instance to open the database when the file is online. If crash recovery finds a file that needs recovery from more than one thread and was externally cached, then the file needs media recovery.
- Zeroed blocks allowed: This is set for files that have been converted from Oracle7 where blocks were initialized to zero. If this is not set, then a block of all zero is media corrupt.

Datafile Header

- Checkpoint count
- Checkpoint Structure
- Backup Checkpoint SCN
- Absolute Fuzzy SCN
- Recovery Fuzzy SCN

3-50 Confidential

Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Datafile Header (continued)

Checkpoint Count

- Allow detection of a restored datafile or control file
- Incremented at every checkpoint

Checkpoint Structure

- Records the last checkpoint information
- Frozen when file is in hot backup mode

Backup checkpoint SCN

- Updated during hot backups
- Used in conjunction with the alter database datafile end backup command:

If the checkpoint count and the backup checkpoint structure match the information found in the control file, the end backup command succeeds and clears the hot backup fuzzy bit.

Datafile Header (continued)

Absolute Fuzzy SCN

- Set to 0 when not set
- Server managed backups calculate absolute fuzziness
- Set to the highest SCN+1 when scanning block headers for backups
- Cleared when the file checkpoint is advanced beyond the current value

Recovery Fuzzy SCN

- Set to 0 when the MR fuzzy bit is not set
- Set to a non-zero value when media recovery sees changes beyond MR fuzzy SCN
- MR fuzzy SCN is advanced for all files in recovery as soon as any redo is encountered
- When not set to 0, it is always >= checkpoint SCN
- Cleared when checkpoint is advanced by media recovery

Datafile Header

- The temp files are listed after the permanent files.
- Control file information
- Generic File header:
  - DB Id
  - File number and type
- Temp file header information

  Except for tablespace information, all fields are set to binary zero.

3-52

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Datafile Header (continued)

The temp file information is listed at the end of the datafile dump. The temp file numbers are independent of the permanent datafile numbers.

Other than the control file section (discussed earlier) the other information to obtain is the DB ID and the file number (relevant to temp files) and type, which should be 6 for temp datafiles.

The rest of the file header information is set to zero.

Datafile Header Dump

- Datafile header dump END
- Log file header START

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Redo Log Headers Dump

```
alter session set events 'immediate trace
name redohdr level <n>';
```

- Level 1
  Control file's redo log entry
- Level 2
  Level 1 + generic file header
- Level 3 or greater
  Level 2 + log file header
- Level 10 most commonly used

3-54

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Redo Log Headers Dump

Currently, level 10 is the maximum level. Itprovides the same information as a level 3 dump.

Log File Header

- Control file record
- Generic File header:
  - DB Id
  - File number and type
- Thread
- Sequence number
- End of Thread Marker
- Resetlogs information
- Low SCN and Next SCN
- Checksums

3-55 Confidential

Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Log File Header

The DBID can be used to ascertain if the logs are associated with the correct database. The file number is the log file number, and the file type is 2 for redo logs.

The Thread number uniquely identifies the thread# of the log. The sequence number uniquely identifies the seq# of the log with the redo stream for the database.

The current log in every enabled thread has the end-of-thread (EOT) flag set in its header. Complete recovery tries to continue redo application through the EOT in all threads.

The resetlogs SCN and count are recorded to prevent the redo from being applied to the wrong incarnation of the database.

A redo log cannot be applied by recovery if its resetlogs data does not match the database information record of the control file. However, backups taken just before the open resetlogs can be used if there were no transactions between the time the backup was taken and the time the resetlogs was issued. That is, if there is absolutely no change between the incomplete recovery SCN and the resetlogs SCN. This is accomplished internally by placing the recently recovered files offline before the database is opened with resetlogs.

Low SCN is the SCN associated with the first redo record for this log. The next SCN is the low SCN of the log with the next higher sequence number for the same thread.

The thread enable and closed information is also stored in the log file header.

For corruption prevention, block checksums are recorded in the file headers.

V$ Views

The following V$ views contain the most useful recovery-related information:

- **Database**

  V$DATABASE
- **Control file**
  - **V$CONTROLFILE**
  - **V$CONTROLFILE_RECORD_SECTION**

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

V$ Views

Starting with Oracle8, there is an additional class of fixed views, the GV$ (Global V$) fixed views. For almost every V$ view there is a GV$ view. In OPS, querying a GV$ view retrieves the V$ view information from all qualified instances (OPS_ADMIN_GROUP). In addition to the V$ information, each GV$ view has an additional column named INST_ID which displays the instance number from which the associated V$ view information was obtained.

To use GV$ views you must:

- Set PARALLEL_MAX_SERVERS > 0 on all instances mounting the database (Oracle uses PQ slaves for inter-instance communication).
- Have the OPS_ADMIN_GROUP parameter defined with at least one member.

V$DATABASE externalizes database information from the control file. CONTROLFILE_TYPE is CURRENT for a current control file. This is important because it determines whether information in other V$ views is current.

V$CONTROLFILE shows the control file names, useful to cross check against the init.ora file.

V$CONTROLFILE_RECORD_SECTION can be used to determine the size occupied by the various portions of the control file.

V$ Views

- **Datafile information**
  - **V$DATAFILE , V$DBFILE**
  - **V$RECOVER_FILE**
  - **V$DATAFILE_HEADER**
  - **V$BACKUP**
  - **V$OFFLINE_RANGE**
- **Tempfile information**

  V$TEMPFILE
- **Tablespace information**

  V$TABLESPACE

3-57 Confidential

Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

V$ Views (continued)

- V$DATAFILE gets all its information from the control file. STATUS shows the status of each datafile and the CREATION_CHANGE# is the SCN of first redo to apply if the datafile is re-created (in case there is no backup and file is recreated from redo logs only). BYTES is the file size Oracle expects to see. CHECKPOINT_CHANGE# is the SCN of the last checkpoint issued against this file, useful in determining if the file can be recovered using only the online logs. V$DBFILE gives the OS file name.
- V$RECOVER_FILE lists files that need media recovery. ONLINE gives the current status of the file, which is different from V$DATAFILE. For example, a file may have RECOVER status in V$DATAFILE but V$RECOVER_FILE may show it as OFFLINE. CHANGE# is the SCN of first redo needed to start recovery.
- V$DATAFILE_HEADER externalizes file_hdrs dump information. Its information is derived from the data files themselves. Main columns to check are STATUS, RECOVER, FUZZY, and ERROR.
- V$BACKUP shows data files that are in hot backup mode.
- V$OFFLINE_RANGE lists the offline ranges for the datafiles.
- V$TEMPFILE gives file number and size information for temp files in the database.
- V$TABLESPACE gives the tablespace number and name only.

V$ Views

- Online redo logs
  - V$THREAD
  - V$LOGFILE
  - V$LOG
- Offline (archived) redo log
  - V$LOG_HISTORY
  - V$ARCHIVED_LOG
  - V$RECOVERY_LOG

3-58 Confidential

Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

V$ Views (continued)

- V$THREAD gives information on the status of all available threads.
- V$LOGFILE lists names and statuses of online logs. The information comes from the control file.
- V$LOG lists online log information including thread#, sequence#, archival status, file status, low SCN, and file size in bytes. It also gets its information from the control file.
- V$LOG_HISTORY externalizes the control file's log history table. Log history information is stored *only* in the control file.
- V$ARCHIVED_LOG contains the same information as V$LOG_HISTORY, and also lists the archive log names.
- V$RECOVERY_LOG is a subset of V$LOG_HISTORY and it lists information about archived logs that are needed to complete media recovery

Summary

In this lesson, you should have learned about the following:

- Approaching recovery issues
- Control file dump structures
- File header dump structures
  - Data file
  - Log files
- Fuzzy bits
- V$ Views

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

References

- *Internals Module 2 - Backup and Recovery (v7)* by Lawrence To
- Source code for Oracle8
- *Oracle8 v8.1 Recovery Outline* by Andrea Borr, Bill Bridge, and Ron Obermarck, Sept. 1999.
- *Design Specification for Temporary File Support, RDBMS 8.1* by Ron Obermarck, Dec. 1997.

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

4

Offline File Recovery

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Schedule:	Timing	Topic
	40 minutes	Lecture
	30 minutes	Lab
	70 minutes	Total

Objectives

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

- Describe the different states that a file can be in after a shutdown
- Recover using a cold backup
- Recover using a hot backup
- Recover read-only/offline tablespaces

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Files After a Clean Shutdown

- Before shutdown, all uncommitted transactions are rolled back.
- Dirty blocks in buffer cache are written out to disk.
- Files are checkpointed and synchronized with control file.
- Datafile header status is set to 0x00.

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Recovering Data Files After a Clean Shutdown

- No recovery needed.
- All files are consistent with one another and with the control file.
- There are no committed changes that need recovering.
- Can open database even if online logs are not available (using resetlogs).

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Files After Shutdown Abort or Crash

- Dirty blocks in buffer cache are not written to disk.
- Files are not checkpointed during shutdown.
- Control file may be ahead of data files.
- File status is not changed.
- Uncommitted transactions are not rolled back.

  Blocks on disk may contain changes from uncommitted transactions.

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

Recovery After Shutdown Abort

- Files are not consistent with each other or with control file.
- Data file may be missing committed transactions from the redo stream.
- Data file may contain active uncommitted transactions which need to be rolled back.
- File status indicates recovery needed.

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

Recovering Files From Cold Backup

- Shutdown normal or immediate cleans out the buffer cache and checkpoints the files:
  - Files are consistent with one another.
  - No changes are missing.
- Uncommitted transactions are rolled back during a clean shutdown.

  No uncommitted transactions are stored on disk.
- File status is changed to indicate that files are offline and consistent.

4-7

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Recovering Files From Cold Backup

The database cannot be shut down cleanly if a tablespace is in hot backup mode:

```
ORA-01149: cannot shutdown - file 1 has online backup set
ORA-01110: data file 1 : '/u02/oracle/or8i/system01.dbf'
```

Recovering Files From Cold Backup

- To recover, no online logs are needed.
  Resetlogs can be used.
- Control file is not needed.
  Control file can be recreated with SQL or script.
- To roll forward, specify "using backup controlfile" option in recover command; or use a more recent control file.

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

Recovering Files From Cold Backup (continued)

A cold backup does not necessarily include the control file or redo logs. Having the control file is helpful, but it can always be re-created. Only a complete set of data files are needed to recover a cold backup.

If a cold backup is copied over an existing backup for recovery, but the online logs are not removed, Oracle will not be able to open the database without RESETLOGS, since the consistency checks against the online logs will fail. (seq# in control file < seq# in log file headers).

When issuing a RESETLOGS, Oracle writes the log files sequentially in 64K blocks. This can have a serious impact on recovery time. If many or large sized redo logs are normally used with the database, drop all but two or three online logs, and open the database with RESETLOGS. Add in more redo logs after the database is open and available.

If the database was not shutdown cleanly when the cold backup was taken (also called fuzzy backup) , the datafiles will not have the proper file status (0x00) and recovery using this backup set may not be possible. Attempting to recover with a fuzzy backup set will succeed only if complete recovery is attempted (to the end of all redo threads for the database) and all generated redo is available. A message such as the following will appear in the alert log:

```
ALTER DATABASE RECOVER database until cancel;
Media Recovery Start
WARNING! Recovering data file 1 from a fuzzy backup. It might
   be an online backup taken without entering the begin
   backup command.
```

Hot Backups

- Copied while the database is open.
- Tablespaces being backed up MUST be in hot backup mode.

```
alter tablespace <tbsp> begin backup;
```

- As soon as the backup has completed at OS level, take the tablespace out of hot backup.

```
alter tablespace <tbsp> end backup;
```


Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Hot Backups

To take hot backups, the database must be in ARCHIVELOG mode.

If you are using RMAN, the tablespace does not need to be put into backup mode.

Hot Backups

- Mechanisms of hot backup:
  - Data file checkpoint structure is frozen.
  - Backup is transparent to users.
- Block splits during hot backups:
  - Data block is written by DBW0 while it is read by the O/S copy utility.
  - Can cause fractured or split blocks.
  - Redo for whole blocks is logged to the redo stream.
  - More redo is generated during hot backups.
  - Tablespace should be taken out of backup mode as soon as the file copy is completed.

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Hot Backups (continued)

Fractured blocks may occur when there is an update to a database block while the block is being backed up at the OS level. Because Oracle blocks normally consist of more than one OS block, it is possible that part of the block is backed up, the block is changed, and then the rest of the block is backed up. The two halves of the block are not consistent with each other.

To prevent this, Oracle copies the entire block to the redo logs. During recovery, the block is copied from the redo log and any further changes logged in the redo are applied to the recovered block.

Because Oracle is now logging entire blocks to the redo logs, hot backups should not be taken during a time of heavy DML activity. Also, you should put only one tablespace at a time into hot backup mode.

Hot Backups

- If the database crashes during a hot backup, files in hot backup mode will require media recovery on startup:
  - Take datafiles out of hot backup mode.
  - V$backup shows datafiles in hot backup mode.
- If any redo logs are missing, recovery is impossible.
- Very important to thoroughly test hot backup scripts.

4-11 Confidential

Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Hot Backups (continued)

To take a tablespace out of backup on startup, mount the database and issue the following command:

```
ALTER DATABASE DATAFILE '<filespec>' END BACKUP;
```

This updates the data file header and control file..

Alternatively, you can issue the RECOVER DATABASE command but this will involve checking all the redo logs for missing transactions since the hot backup started.

When taking hot backups, securing all the redo necessary to restore the datafiles to a consistent point in time is critical. To do this, follow these steps:

1. Alter system switch logfile. Make note of the current log.
2. Put tablespaces in backup mode one by one and copy at OS level.
3. After all files have been backed up, make a backup copy of the control file.
4. Issue an ARCHIVE LOG CURRENT so that all changes in the online logs are archived. Make note of the last log to be archived.

The backup set now consists of all datafiles, the control file backup, and all the archive logs from the current log in step 1 to the last archived log in step 4. These are all archive logs that could possibly contain information needed to recover the backup set.

Restore From a Hot Backup

Some form of recovery must be performed because the datafiles are inconsistent with each other.

- **Complete recovery:**
  - **Requires the current online log and all archived redo since the first checkpoint for this hot backup.**
  - **See v$recover_file and/or v$recovery_log.**
- **Incomplete recovery**

  Requires all redo from the first checkpoint to the end backup redo marker for the last data file in the backup set.

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Restore From a Hot Backup

When taking hot backups, the file header is frozen while the file is being copied. This means that each data file can have a different checkpoint in your backup set. To make the backup set consistent, all files need to be recovered until their SCNs match again and the fuzzy bits have been reset.

Oracle stores a marker in the redo stream when a tablespace is taken out of hot backup mode.

```
REDO RECORD - Thread:1 RBA: 0x000033.00000002.0010 LEN: 0x0038 VLD:
0x01
SCN scn:  0x0000.000e6abf 12/20/99 08:50:03
CHANGE #1 MEDIA RECOVERY MARKER SCN:0x0000.00000000 SEQ:  0 OP:17.1
End backup marker - file:2 scn:  0x0000.000e6abb
```

Offline Tablespaces

- Three different modes of offline:
  - NORMAL: No recovery needed
  - IMMEDIATE: Recovery needed
  - FOR RECOVER: For TSPITR
- All files belonging to the tablespace are made offline.
- There should be no active rollback segments in an offline tablespace.
- Once a tablespace is offline, it can be dropped.
- Datafiles can be taken offline separate from the tablespace but not dropped separately.
- No writes or reads to offline files unless they need recovery.

4-13 Confidential

Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Offline Tablespaces

If a tablespace or data file is taken offline with IMMEDIATE option, while the database is open, you must perform media recovery before bringing it back online, because a checkpoint is not performed on the datafile before it is taken offline.

A tablespace can be taken offline even if there are active transactions in the tablespace. A subsequent select on these files will return ORA-376, ORA-1110. However, the DML statement will complete.

Example: What happens if you do this?

Session 1	Session 2
1. Update a table in a tablespace	Alter tablespace offline
2. Commit	
3. Select updated rows from table	Alter tablespace online

Read-only tablespaces cannot be put in hot backup mode (ORA-1642).

Tablespace point in time recovery (TSPITR) and Media Recovery are covered later in this course.

Offline Datafiles

- Control file status of offline files
  - NORMAL: 0x80
  - IMMEDIATE: 0x10
  - FOR RECOVER: 0X10
- Data file status of offline files
  - NORMAL: 0x0
  - IMMEDIATE: 0x4
  - FOR RECOVER: 0X4
- Once a tablespace is offline, it can be dropped. Datafiles cannot be dropped unless the tablespace is dropped.

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Read-Only Tablespaces

- Tablespace must be online, no RBS.
- All files belonging to the tablespace are made read-only.
- Tablespace or data files cannot be in backup mode.
- Cannot contain any active transactions or rollback segments.
- No writes to read-only files unless they need recovery.
- Read-only tablespaces are still available for queries.
- Can be moved to read-only media.

4-15 Confidential

Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Read-Only Tablespaces

A tablespace cannot be made read-only if there are any active transactions in the DATABASE. This is because there is no specific link between a transaction and the tablespaces which it modifies. In 8*i*, the ALTER TABLESPACE command will wait until all active transactions have completed. Prior to 8*i* the command would fail with ORA-1640: " cannot make tablespace read-only with active transactions"

Making a tablespace read-only while a file is

OFFLINE : ORA-01230: cannot make read only - file %n is offline

Making a tablespace read-only while tablespace is

HOT BACKUP : ORA-01150: cannot prevent writes - file %n has online backup set

OFFLINE : ORA-01539: tablespace 'HIST' is not online

Read-Only Tablespace Benefits

- Only one backup of the tablespace is needed after it is made read-only.
- Read-only files are similar to offline normal files, and recovery ignores them.
- After being set Read-only files are no longer checkpointed.

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Read-Only Tablespace Benefits

Read Only files are similar to offline normal files from a recovery perspective: recovery ignores files in the offline range. The rest of the lesson applies to offline normal files (that is, offline normal or read-only).

There are other optimizations on read only files

For example, in a Parallel Server environment you can have only 1 PCM lock per read-only file.

This lesson will only look at recovery aspects of read-only/Offline files.

Read-Only Tablespace Internals

When a tablespace is changed from read-write to read-only:

1. All the files in the tablespace are checkpointed.
2. Redo for all these files is written to disk.
3. For each file in the tablespace, the file status in the control file is updated to 0x86.

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Read-Only Tablespace Internals

4. Stop SCN for data files is set to the last checkpoint SCN.
5. Online fuzzy bit is cleared from file headers.
6. TS$ is updated.

   SCN BAS and SCN WRP are set to the last checkpoint SCN for the tablespace.

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Read-Only Tablespace Internals (continued)

The STOP SCN for a file is the SCN at which recovery stops for this file.

Review Questions:

1. What is 0x86 file status (from Chapter 3) ?
2. What does save offline range mean ? (this is after you have answered 1)

Read-Only Tablespace Internals

When the tablespace is changed from read-only to read-write the following happens:

1. **Online checkpoint SCN for the files is set to current checkpoint SCN.**
2. **Offline SCN range is stored .**
3. **Stop SCN is set to infinite (0xffff.ffffffff).**
4. **File header fuzzy bit is set.**
5. **Stop SCN is cleared from TS$.**

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Read-Only Tablespace Internals (continued)

The offline range is the range marking when the file was offline. This range is updated only when the file is made read-write; the offline SCN is set to the stop SCN and the online checkpoint SCN is set to the SCN when the file was brought online.

Recovery of Read-Only Files

Crash / Instance recovery ignores read-only files if:

- **Checkpoint SCN in the file headers = stop SCN for the file in the control file.**
- **File status is not fuzzy.**

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Recovery of Read-Only Files Current Control File

- Read-only files are online but do not generate redo.
- Media recovery applies redo to online, read-write files.
- Files where recovery SCN falls in the offline range are ignored during the recovery process.
- End SCN of the offline range tells when to start applying redo.

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Recovery of Read-Only Files Backup Control File

Since the backup control file does not reflect the status of the file through the entire recovery, you have the following options:

- **Take the file offline before starting media recovery and make it online after open with resetlogs.**
- **Recreate control file without the read-only files then use the ALTER DATABASE RENAME FILE for the read-only files.**
- **Use a control file with read-write status for the read-only files (required if files were altered read-write during the recovery period).**

4-22 Confidential

Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Recovery of Read-Ony Files Backup Control File

Recovery is determined by the control file. If you attempt to recover a datafile that is marked "read only" with a backup control file, you will fail with ORA-01233.

ORA-01233: "file %s is read only - cannot recover using backup controlfile"

Cause: An attempt to do media recovery using a backup control file found that one of the files is marked read only. Read only files do not normally need to be recovered, but recovery with a backup control file must recover all online files.

Action: If the file really is read only, take it offline before the recovery, and bring the read only tablespace online after the database is open. If the file does need recovery use a control file from the time the file was read-write. If the correct control file is not available, use CREATE CONTROLFILE to make one.

When the ALTER DATABASE BACKUP CONTROLFILE TO TRACE command is used, read-only files are not listed in the datafile section. At the end of the trace file is the command:

```
ALTER DATABASE RENAME FILE 'MISSING00006' TO
'/u02/oracle/hist512.dbf';
```

Recovery of Read-Only Files Backup Control File Caveats

- Create control file script does not include read-only files.
- Read-only files are made online and open after database is opened.
- Checkpoint SCN must be same as Stop SCN in TS$.

If read-only files need recovery they should be included in the control file.

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Recovery of Read-Ony Files Backup Control File Caveats

The files must be current (checkpoint SCN = Stop SCN) before they can be brought online.

Summary

In this lesson, you should have learned the following:

- Redo logs are used to maintain consistency of hot backup files and to avoid fractured blocks
- File status flag controls operations performed on data files:
  - Hot backup: log full blocks to redo
  - Offline: no queries allowed
  - Read-only: no checkpoints performed
- Read-only/offline files are ignored during recovery and need to be backed up only once
- The current control file must be used for media recovery of read-only files

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

References

- *Oracle7 Internals Module 2* - Lawrence To, COE
- *Database Recovery (Prevention, Detection, & Repair Concepts)*, Lawrence To, 1995
- "*Oracle8 v8.1 Recovery Outline*" - Andrea Borr, Bill Bridge, and Ron Obermarck, 1999 (`http://www-sup.ca.oracle.com/spicy`)

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

5

Block, Crash, and Instance Recovery

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Schedule:	Timing	Topic
	60 minutes	Lecture
	60 minutes	Total

Objectives

After completing this lesson, you should be able to describe features and implementation of:

- **Block recovery**
- **Crash recovery**
- **Instance recovery**
- **Recovering from block corruption**

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Block Level Recovery

- Block level recovery:
  - Results from a process death
  - Is automatic and transparent to the user
  - Uses only online logs and online data files
- Oracle8*i* features advanced block level recovery
  - Recovers interested blocks first
  - Avoids waiting for complete transaction recovery

5-3

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

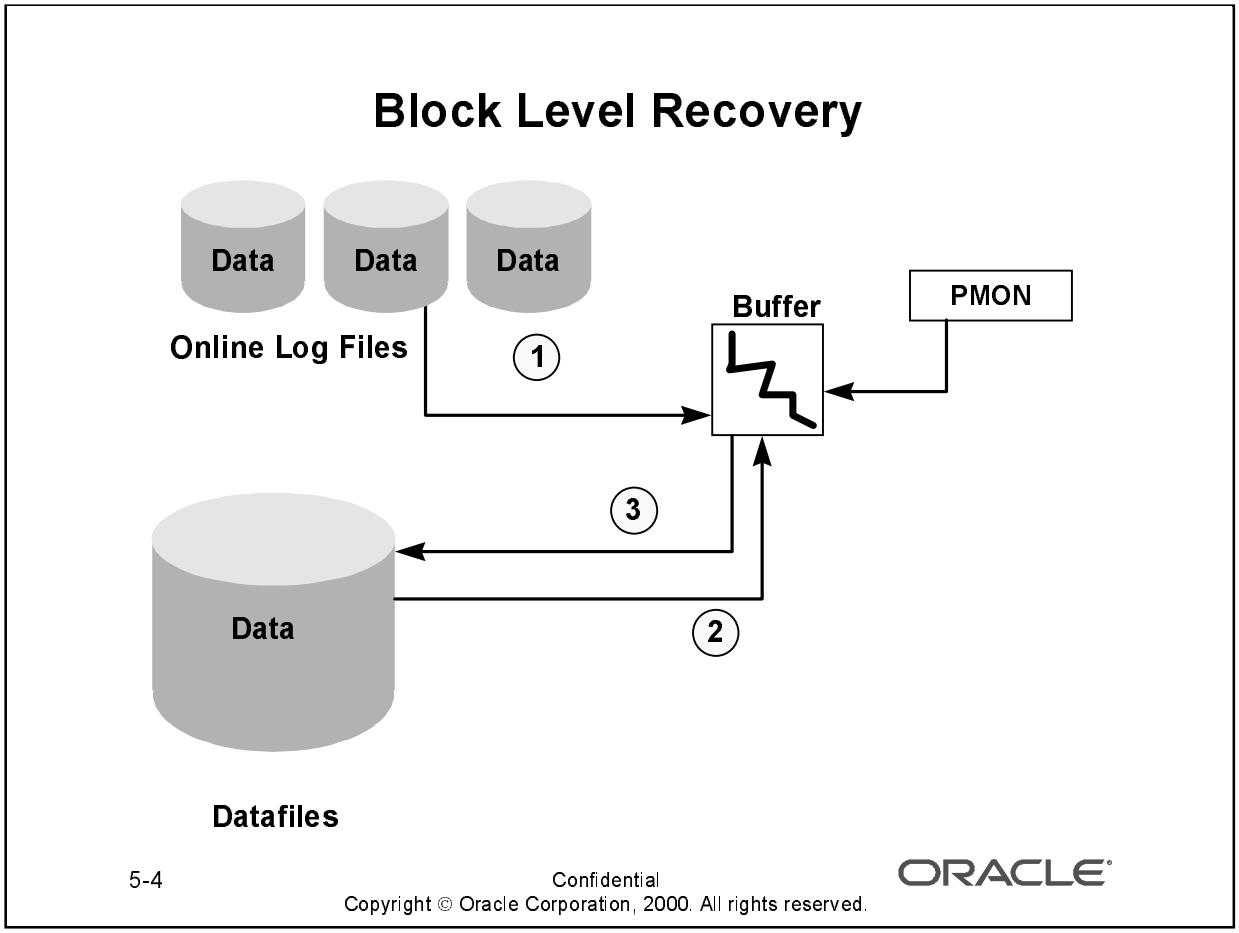
Block Level Recovery

Block Recovery

Block recovery occurs as part of the normal operation of the Oracle database. If Oracle detects a corrupted buffer in the cache it attempts to reconstruct the changes made to that buffer prior to the occurrence of the problem. In order to do this, Oracle needs to read the block that this buffer corresponds to from the datafile and then applying any redo information applicable to this block from after the last checkpoint. This information will be required only from the online logs as these logs are checkpointed prior to them being overwritten Block recovery cannot require redo from another thread or redo prior to the last checkpoint.

Block Level Recovery in Oracle8*i*

The feature of Block Level Recovery implemented in Oracle 8*i* improves the availability of data in the Oracle RDBMS after a failure. Block level recovery immediately recovers only the data block under consideration, leaving the rest of the dead transaction to be recovered in the background. This improves the availability of the database for users accessing data that is locked by large dead transactions.



Block Level Recovery (continued)

The `kcrabr()` function is called when a process dies or is interrupted while it was changing a block in memory. It is normally performed by PMON but the task can be undertaken by SMON or a foreground process. SMON will perform block recovery as part of the rollforward stage of recovery (discussed later), and a foreground process may detect an error during the redo callback routine.

Block recovery is guaranteed to be able to be completed. The rredo after the last checkpoint from the current one thread is required.. This is because no other thread can “dirty” the same buffer, and the cache coherency policy (used by Oracle Parallel Server) protects against inconsistencies where multiple caches are involved (essentially, where two instances wish to modify the same block, Oracle will write the dirty block back to disk before reading it in to the other instance’s cache).

The steps involved are:

1. Read the block from disk
2. Use the current thread’s online redo to reconstruct the buffer to a consistent state.
3. Write the recovered block back to disk.

Block Recovery: Buffer Headers

Fields in the buffer header:

- **Low-RBA**
- **High-RBA**
- **Recovery-RBA**

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Block Recovery: Buffer Headers

Structures pertaining to recovery for a data block are stored in the buffer header. Block recovery proceeds from the Low-RBA to the High-RBA. These fields delineate the range of redo necessary to apply to the disk version of the block to make it consistent with the redo already generated.

If PMON is performing block recovery then the amount of redo read is limited to 20 buffers by KCRABRCNT. This prevents PMON from becoming tied up recovering one block, and ignoring other duties. The RBA corresponding to the redo already applied is stored in the Recovery-RBA structure. This prevents PMON from unnecessarily trying to reapply the same redo.

Setting the event 10293 will show trace information on block recovery.

Block Recovery: When to Apply Redo

- Redo application starts at the Low-RBA
- Redo is applied only to the correct version of the block
  - Determined by DBA, SCN, SEQ
  - Block must not be physically corrupt

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Block Recovery: When to Apply Redo

Block version is determined by the SCN and the SEQ number of that block. SCN, SEQ, and DBA are stored in the header of each block; and in each change vector.

If the DBA, SCN, and SEQ are equal in both the block and the change vector, then the change vector is applied to the block. Cache checking is performed before and after the redo application to determine whether or not the block is corrupted .

If the block is at a higher SCN, then the redo has already been applied and can be discarded. If the redo advances the SCN, then the SEQ is reset to 1, otherwise the SEQ number is advanced..

If the SCN in the block header is behind the SCN in the change vector then we are missing some changes. This may indicate the system has lost writes, or a bug has been encountered.

It is possible for logically and media corrupted blocks to be corrected by a NEWING action during recovery. During a new of a block (when it is utilized within a new segment) we reinitialize the block header and reset the SCN. Depending on why the block was corrupt this action could clear the problem. In many cases, recovery ceases when a corrupted block is encountered - nothing will save the block in this situation.

Block Recovery: Failure

- If block recovery fails, retry
- If there is a subsequent failure:
  - Block marked logically corrupt
  - Internal error signaled

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Block Recovery: Failure

It is possible for block recovery to fail. If this happens Oracle attempts block recovery again. If there is a second failure then the block is marked logically corrupt, by setting the scn field to zero. A trace file will also be generated at this point.

Other internal errors can occur during block recovery (and redo application); for example (from kcra.c and kco.c):

- ORA-600 [3001] - a log containing the low RBA could not be found.
- ORA-600 [3015] - recovery did not complete successfully
- ORA-600 [2021] - ORA-600[2024] - failure in a validation check
- ORA-600 [kcoapl_2] - failure with block version and delayed logging block cleanout.

Redo Application

- **The redo application algorithm has changed from V7.**
- **In essence the concept is the same, however the checks applied are more complex.**

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

Redo Application

In Oracle version 7, the application of redo, in pseudo code, looked like this:
(Given a block and redo entry that corresponds to that block)

```
If block INC# > redo INC#
   1. continue         /* irrelevant redo entry */
Else if block INC# < redo INC#
        1. if redo SEQ# = 1  /* 1st change to new incarnation of a block
*/
             apply redo change
             increment block SEQ# to redo SEQ#
             continue
            else
              signal error   /* recovery stuck - missing changes */
       Else                   /* block INC# = redo INC#  */
         1. If block SEQ# = redo SEQ#
              apply redo change
              increment block SEQ#
              continue
            Else if block SEQ# > redo SEQ#
                    continue         /* redo already applied */
                 Else                 /* block SEQ# < redo SEQ# */
                    signal error   /* recovery stuck  - missing changes */
                 endif
          endif
```

In V8 we now perform checks along the following lines:
(Given a block and redo entry that correspond to that block)
Based on change type, it is determined if special kinds of processing may be needed when applying this change vector. The change types are defined in kco.h.

Redo Application

Checks are employed based on the type of redo.
From kco.h, the types are:

- **KCOT_NORM**
- **KCOCT_NEW**
- **KCOCT_DLOG**
- **KCOCT_IMAGE**
- **KCOCT_CLEAR**
- **KCOCT_INVLD**
- **KCOCT_MARKER**

5-9 Confidential

Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Redo Application (continued)

From kco.h the types of change vectors are defined as kcohdtyp:

KCOCT_NORM

This is the most common type of change. The block version must match kcohdscn and kcohdseq exactly. This cannot be applied to a corrupt block of any type.

KCOCT_NEW

This change will new the block at the scn of this redo. The redo can be applied if kcohdscn and kcohdseq are greater than or equal to the value in the block header. Thus a media corrupt block can always be repaired by applying this change, and a logically corrupt block can be repaired if kcohdscn is a higher scn. Before calling the redo application callback, the block will be formatted at scn kcohdscn with a zero data area and seq# one. It is possible that the formatted version of the block was written to disk thus allowing the case where this is applied to a block that matches kcohdscn and kcohdseq exactly. Note that unlike most changes at a different scn than the previous log entry, the block that results from applying this change may have seq# two rather than seq# one. That happens when kcohdscn equals the scn this redo is generated at. Note that kcohdseq will always be one for a change of this type.

KCOCT_DLOG

This change was made from a block that had a delayed logging change in it. A delayed logging change is a modification to a buffer in the cache that does not generate redo. This can be done when the change may be lost without any information loss. The best example of this type of change is a block cleanout.

Redo Application (continued)

The changes will be recalculated again if lost. The first logged change, after a delayed logging change, must include the delayed change as well as what ever other change is needed. The SCN and SEQ of the block with the last logged change, is not available, but it must be less than this kcohdscn and kcohdseq. This change may be applied to either version. If the change is applied to the version without the unlogged change then the redo application callback must reconstruct the unlogged change. If the change is applied to the version with the unlogged changed then it must be marked as logged. The callback does this by noticing the unlogged change in the block. In order to ensure that the next version of the block has the same header, kcohdscn and kcohdseq must be stored in the header before advancing it to indicate this change has been applied.

KCOCT_IMAGE

This change vector does not represent a new version of the block. It is a block image that was logged during a backup, just in case the block in the backup is fractured. It can be used to repair a corrupt version of the block. Applying this redo does not increment the block version to the SCN of this redo. The block header must contain kcohdscn and kcohdseq so that the next change can be applied.

KCOCT_CLEAR

This change vector is exactly like a block image except that it does not have any information for the data area. It is applied just like KCOCT_IMAGE except that the block header is marked as new. This type of redo is needed because there are times when we read a block that no longer contains what we are expecting. Blocks containing temp data have images logged with this type of redo so that readers for a previous usage of the block will not see a media corrupt block if a hot backup has been restored and recovered.

KCOCT_INVLD

This invalidation redo is generated during unrecoverable direct path operations. The DBA gives the first block and the class gives the number of blocks that were loaded without logging the new block contents. This redo is applied only by media recovery. If the block being recovered is at a version before this redo then it is marked as logically corrupt at kcohdscn. Otherwise the redo is ignored. This is like newing a block except that it immediately becomes logically corrupt.

KCOCT_MARKER

This is not a change to a data block. It is a marker that is used only during media recovery. This represents an operation involving a control file transaction that needs to be done during media recovery. Examples include controlfile redo, thread enable, file resize, and end hot backup. The DBA will always be zero for this type of redo. The change vector is ignored except during media recovery.

Redo Application

Considering the NORMAL type (the most common), structures that are considered include:

- **Block SCN**
- **Block Sequence (up to 254 per SCN)**
- **Redo SCN**
- **Redo Sequence**
- **Commit SCN**

5-11

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Redo Application (continued)

All structures except the commit SCN are found in the block header or the redo change vectors. Oracle allows up to 254 changes per SCN and record this as the sequence (SEQ). A corrupt block will have a SEQ = 255, and / or an SCN of 0 (depending on logical or media corruption). The commit SCN is returned by the function kcrfgs() and is the SCN allocated when a commit redo record was last generated or applied by this process.

Considering the NORMAL change type (the most common) , given a change vector and the block it applies to (assuming the block IS NOT corrupt):

```
NORM -    If the block SCN  < redo SCN
          error  /* redo in the future */
     If block SCN = redo SCN
          If block SEQ = redo SEQ         Apply the change
          If block SEQ < redo SEQ         error  /*  redo in the future*/
     If block SCN  > redo SCN
           if block SCN < commit SCN
                if Delayed logging -> Apply change
                else error  /*  possibly lost writes */
           if block SCN >= commit SCN
                block SEQ < commit SEQ
                     error  /*  redo in future */
           else redo already applied
```

Crash/Instance/Thread Recovery

- Required when there is an instance failure resulting in loss of the instance's buffer cache.
- Crash failure refers to failure of all instances in an Oracle Parallel Server environment.
- Instance failure refers to a failure when there is one or more surviving instance in an OPS environment.

5-12

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Crash/Instance/Thread Recovery

The terms thread, crash and instance recovery refer to very similar operations, although there are some differences. Crash recovery is the failure of all instances (the only possibility in exclusive mode). Instance recovery is the failure of one or more instances in an OPS environment (at least one instance remains and this instance performs the recovery operation for the failed instances). Thread recovery applies to either situation. Mostly this section will discuss crash recovery, leaving a detailed discussion of instance recovery in an OPS environment to DSI308.

In either case the corrective actions to be taken are very similar. Instance or crash failure affects database integrity because the instance's dirty buffer cache is lost. This means that the memory version of certain blocks is different from their on-disk version, as explained in the next slide.

Crash/Instance/Thread Recovery

A failure could leave the database in one of the following states:

- **Data files on disk contain some but not all of the changes of a single atomic change.**
- **Data files have some changes from uncommitted transactions.**
- **Data files on disk do not have changes from committed transactions.**

5-13

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Crash/Instance/Thread Recovery (continued)

Crash / Instance / Thread Recovery has to guarantee the structural integrity of the database, which after a crash could fail into one of the states listed in the slide.

The recovery layer repairs conditions listed first and third; and transaction layer repairs the second condition.

Crash/Instance/Thread Recovery

- **Oracle's buffer cache is optimized more for OLTP performance than for crash tolerance.**
- **Cache and recovery components provide a mechanism for crash tolerance while maintaining OLTP performance.**

5-14

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Crash/Instance/Thread Recovery (continued)

- Log Force at Commit: Guarantees that at commit time all redo records including the commit redo record for the transaction are written to disk.
- Checkpointing: Bounds the amount of redo that needs to be applied for recovery.
- Online Switch Management: Ensures that instance/crash recovery can be accomplished by online logs only.
- Write-Ahead Log: Ensures that no changes are written to the data files that are not already written to a redo log. This ensures that redo for the undo has been generated so the change can be undone should a crash occur before the change is committed.

Refer to the previous slide and understand how these four characteristics allow recovery from the three failures mentioned on that slide.

Tuning Instance Recovery

- There are several methods for tuning instance and crash recovery:
  - Using initialization parameters to influence the number of redo log records and data blocks involved in recovery
  - Sizing the redo log file to influence checkpointing frequency
  - Using SQL statements to initiate checkpoints
  - Parallelizing instance recovery operations to further shorten the recovery duration
- Oracle8*i* Enterprise Edition offers fast-start fault recovery functionality to control instance recovery

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

Thread Recovery

If an instance crash occurs it is necessary to replay any changes that occurred after the last completed checkpoint:

- **Termed crash, instance, or thread recovery**
- **Automatically performed at time of next startup (no operator intervention)**
- **Uses only online logs (of the thread involved) and data files**

5-16

Confidential

Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Thread Recovery

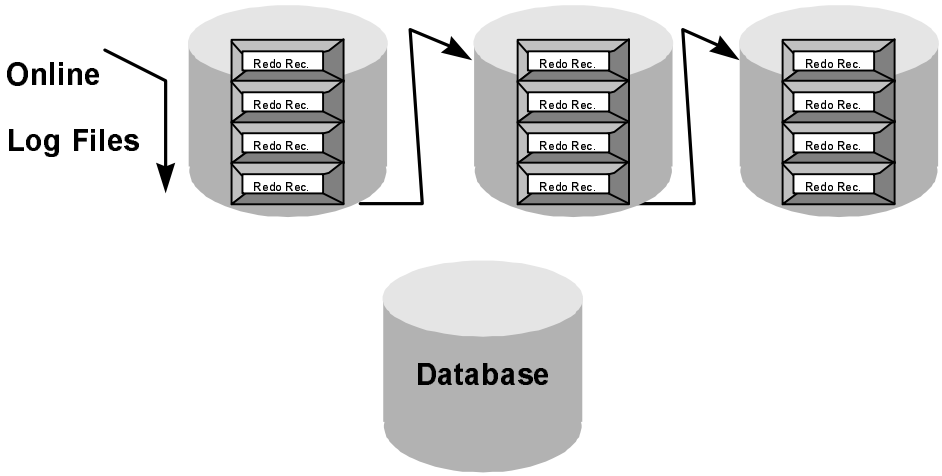
Thread recovery is used to recover from a crash failure (where all instances die) or an instance failure, in an OPS situation, where a surviving instance will recover a failed one.

The aim of thread recovery is to restore the data block images that were in the cache of the failed instance, but had not yet been checkpointed back to disk, and to close the open thread. This *must* be accomplished with only the online log files and the current online data files.

The detection of the need for thread recovery, and the repair itself must proceed autonomously, without operator intervention. It must occur either implicitly at the next startup or through the actions of another instance in the OPS case.

Thread Recovery

Thread recovery applies the change vectors one thread at a time.



5-17

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Thread Recovery (continued)

One redo thread is processed at a time, beginning at its last checkpoint, and continuing to the end of thread. All redo is applied from each thread, before the next thread is started. This algorithm depends on the cache coherence algorithm mentioned earlier that allows for only one instance at any one time to have a modified version of a block in its cache (between modifications the block is written back to disk).

The diagram above applies equally to single or multiple instance Oracle operations: Consider the online logs to represent different groups in the single instance case, or different threads in the multiple instance case. In OPS, recovery proceeds from the thread with the lowest checkpoint SCN, which ensures that the database checkpoint advances as each thread is recovered.

Detection of the need for thread recovery, for single instance, is the presence of an open thread flag in the control file at the next startup. The data files will also have fuzzy bits set in their headers, indicating the files are possibly in advance of their respective checkpoints .

In OPS, the detection of an “invalid block lock” by a foreground process is one of the common indicators of instance failure. This occurs when the detecting instance tries to bring a block into its cache, but there is a lock (indicating that the block may have changed) and no owner exists for that lock. The SMON process in the surviving instance is alerted and begins to search for failed instances. It does this by trying to acquire all RT (thread open) locks, which are normally held by an active instance. Once all the RT locks for a particular instance are acquired, the SMON process recovers that instance.

Thread Recovery

- Duration depends on rollforward and rollback phase.
- At the completion of thread recovery:
  - All redo has been applied.
  - All fuzzy bits are cleared.
  - An end-crash-recovery marker has been generated.

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

Thread Recovery (continued)

At the end of thread recovery the thread is checkpointed and closed. If this occurs as part of a startup, the close is transient. All fuzzy bits in the file header are cleared.

A crash recovery marker is generated for use by Media Recovery to clear fuzzy bits during subsequent recovery operations.

The duration of thread recovery is dependent on both the rollforward and rollback phases. From Oracle version 7.3 onward the database is opened once the rollforward phase (application of redo) completes. Rollforward completion is marked by the "Cache recovery complete" message appearing in the alert.log. Rollback of uncommitted transactions will be processed by SMON at the rate of cleanup_rollback_entries records at a time. Rollback can also be performed by a foreground process which requires access to a block from a transaction that is yet to be rolled back. The foreground process will rollback the transaction in its entirety, not just the block it needs.

Note: cleanup_rollback_entries is a hidden parameter in Oracle8*i*.

LOG_CHECKPOINT_TIMEOUT

- Helps to ensure that the latest checkpoint position follows the most recent redo block by no more than specified seconds.
- An upper bound on the time a buffer can be dirty in the cache before DBW0 must write it to disk.
- The default value is 1800.
- The minimum value in the Standard Edition is 900. If set below 900 in the Standard Edition, Oracle rounds it to 900.

5-19 Confidential

Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

LOG_CHECKPOINT_TIMEOUT

Set the LOG_CHECKPOINT_TIMEOUT initialization parameter to a value *n* (where *n* is an integer) to require that the latest checkpoint position follow the most recent redo block by no more than *n* seconds. In other words, at most, n seconds' worth of logging activity can occur between the most recent checkpoint position and the end of the redo log. This forces the checkpoint position to keep pace with the most recent redo block

You can also interpret LOG_CHECKPOINT_TIMEOUT as specifying an upper bound on the time a buffer can be dirty in the cache before DBWn must write it to disk. For example, if you set LOG_CHECKPOINT_TIMEOUT to 60, then no buffers remain dirty in the cache for more than 60 seconds. The default value for LOG_CHECKPOINT_TIMEOUT is 1800. The minimum value for LOG_CHECKPOINT_TIMEOUT in the Standard Edition is 900. If you set the value below 900 in the Standard Edition, Oracle rounds it to 900.

LOG_CHECKPOINT_INTERVAL

- Helps to ensure that the checkpoint position never lags behind the most recent redo block by more than n redo blocks.
- Oracle limits the maximum value of LOG_CHECKPOINT_INTERVAL to 90% of the smallest log.
- LOG_CHECKPOINT_INTERVAL is specified in redo blocks.
- Use the view V$INSTANCE_RECOVERY to assess the checkpoint interval being applicable.

5-20 Confidential

Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

LOG_CHECKPOINT_INTERVAL

Set the LOG_CHECKPOINT_INTERVAL initialization parameter to a value n (where n is an integer) to require that the checkpoint position never follow the most recent redo block by more than n blocks. In other words, at most n redo blocks can exist between the checkpoint position and the last block written to the redo log. In effect, you are limiting the amount of redo blocks that can exist between the checkpoint and the end of the log.

Oracle limits the maximum value of LOG_CHECKPOINT_INTERVAL to 90% of the smallest log to ensure that the checkpoint advances far enough to eliminate “log wrap.” Log wrap occurs when Oracle fills the last available redo log file and cannot write to any other log file because the checkpoint has not advanced far enough. By ensuring that the checkpoint never gets too far from the end of the log, Oracle never has to wait for the checkpoint to advance before it can switch logs.

LOG_CHECKPOINT_INTERVAL is specified in redo blocks. Redo blocks are the same size as operating system blocks. Use the LOG_FILE_SIZE_REDO_BLKS column in V$INSTANCE_RECOVERY to see the number of redo blocks corresponding to 90% of the size of the smallest log file.

Fast-Start Checkpointing Architecture

New dynamic parameter to limit data file I/O during recovery:

```
fast_start_io_target = 10000
```

- **Useful in establishing service-level agreements with users**
- **Used in conjunction with other parameters to determine target for checkpointing**

5-21

Confidential

Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

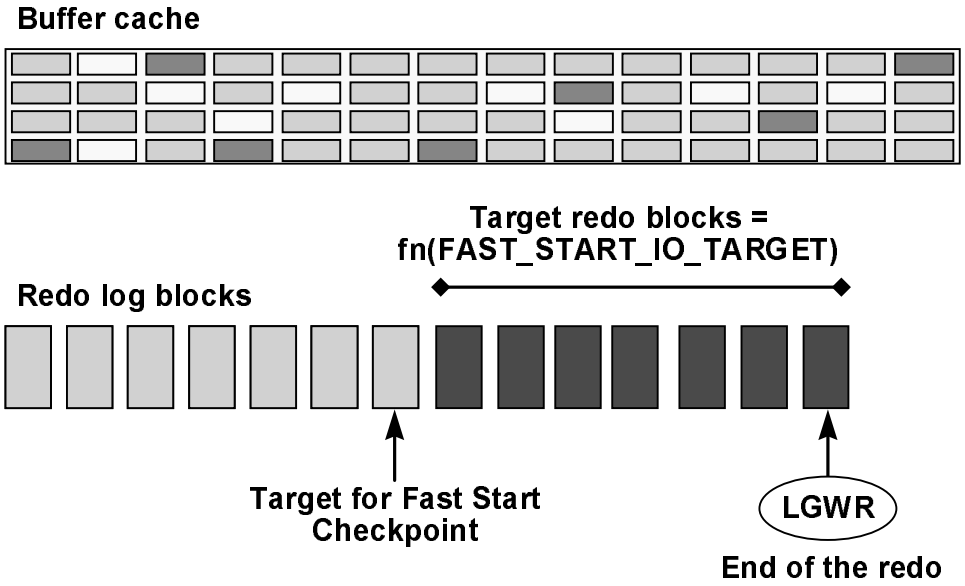
Fast-Start Checkpointing Architecture

Prior to Oracle8*i*, it was difficult to control the amount of time an instance took to perform instance recovery, because it depends on transaction load at the time of failure.

Oracle8*i* introduces some new optimizations to bound the recovery time required. It also introduces a new parameter FAST_START_IO_TARGET that allows a database administrator to specify the maximum number of datafile blocks that should be processed during instance recovery. Because the time to recover from an instance failure is primarily dependent on datafile I/O, and the average I/O time can be estimated from instance statistics, this parameter allows a database administrator to establish service level agreements with users.

This parameter is dynamic and is used with other parameters by the Oracle Server to determine the target redo log file block up to which a checkpoint must progress.

Use of FAST_START_IO_TARGET



5-22

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

Use of FAST_START_IO_TARGET

Fast-Start Checkpoints

Fast-start checkpoints cause database writer (DBWn) to write blocks from the buffer cache so that the earliest buffer to be dirtied gets written first.

Fast-start checkpointing causes DBWn to continually write data blocks so that the checkpoint position in the redo log can advance and satisfy the target set by the FAST_START_IO_TARGET initialization parameter.

Impact of FAST_START_IO_TARGET on Checkpoints

FAST_START_IO_TARGET defines the number of I/O operations (data blocks) that Oracle need to process (read and write) during recovery. Oracle automatically examines the blocks in the redo log and dynamically calculates the target redo blocks given a value of FAST_START_IO_TARGET. Oracle then continually writes dirty buffers to advance the checkpoint position such that the number of redo records that would need to be processed in the event of recovery (those between the checkpoint position and the end of the redo log) does not exceed the calculated target redo blocks.

Setting FAST_START_IO_TARGET attempts to restrict the number of data blocks that will need to be processed during recovery if instance failure were to occur. However, the actual number of blocks read or written may vary slightly because of the time taken to reach the target.

Summary

In this lesson, you should have learned about:

- Block recovery
- Redo application
- Instance / thread recovery

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

References

- *Oracle 7 Internals, Module 2*, Lawrence To, COE
- *Internals of Recovery*, Bill Bridge and Andrea Borr-WebIV note 41391.1
- Oracle 8*i* Documentation:
  - SQL Reference (ALTER SYSTEM)
  - Administrator's Guide
  - Reference Manual (db_block_checking)

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

6

Media Recovery

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Schedule:	Timing	Topic
	60 minutes	Lecture
	90 minutes	Practice
	150 minutes	Total

Objectives

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

- Media recovery concepts
- Checkpoint structure revisited
- Thread Merged Recovery
- Steps of Media Recovery
- V$ Views
- Tracing events

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

Media Recovery

- Performed using commands entered by database administrator:
  - Can be used to recover a control file.
  - Used to recover from disk failures or corruption.
- Brings backup data f iles up to date (advances the file header checkpoints).
- Applies changes from online or archived redo.
- Data files being recovered must be offline or mounted.

6-3

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Media Recovery

Media recovery is the only form of recovery which requires intervention by the database administrator. Instance and block recovery are both invoked automatically by Oracle.

The next few slides review some of the basics of media recovery.

Media Recovery

There are three forms of Media Recovery:

- **Data file recovery**
- **Tablespace recovery**
- **Database recovery**

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Media Recovery (continued)

The different types of media recovery determine which datafiles are to be recovered and also the mode that the database needs to be in to allow media recovery to take place.

Media Recovery: Data File Recovery

- Recovers the specified data files to the same checkpoint as the rest of the database.
- One or many data files can be recovered with one command.
- Database can be mounted or open.
- Data files being recovered must be offline if database is open.
- Only complete recovery is possible.

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Media Recovery: Data File Recovery

The command used to recover datafiles, whether the database is open or closed, is:

```
SVRMGR>recover datafile '<file_name> (,<file_name>)';
```

Datafile recovery is used to restore a file that was taken offline with `immediate` option because of an I/O error or to restore a file after disk failure.

Media Recovery: Tablespace Recovery

- Recovers one or more restored data files belonging to specified tablespaces.
- Can recover multiple tablespaces in the same command.
- Must be performed while the database is open.
- The tablespaces being recovered must be offline.
- Only complete recovery is possible with SQL.
- Incomplete recovery is possible with TSPITR.

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Media Recovery: Tablespace Recovery

Tablespace recovery is the one form of media recovery that is specified at a logical level. Oracle needs to be able to establish which datafiles need recovery by querying the data dictionary that is why the database needs to be open.

The command used to invoke tablespace recovery is:

```
SVRMGR>recover tablespace <tablespace_name> (,<tablespace_name>);
```

Media Recovery: Database Recovery

- Recovers of some or all of the data files in a database.
- Can be used to recover the control file.
- Returns all data files and the control file to the same point in time (SCN) through redo application.
- Requires the database to be mounted but not open.
- Files must be online to be recovered.
- Offline files are not recovered.

6-7

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Media Recovery: Database Recovery

Database recovery is invoked by the following command:

```
SVRMGR>recover database (until time 'xxxx'/change xxxx/cancel);
```

If the database is open, then data file or tablespace recovery can be performed, except on the `SYSTEM` tablespace.

Media Recovery: Database Recovery

- Can be complete or incomplete.
- All files must be recovered to a consistent state.
  - The current database checkpoint
  - A clean stop SCN
- Incomplete recovery is available only when the database has been operating in `ARCHIVELOG` mode.

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Media Recovery: Database Recovery (continued)

Incomplete recovery means bringing the whole database to a point behind the current time. This means that all data files must be restored so that they can be rolled forward to the same point.

Media Recovery: Database Recovery

- Can be performed using a non-current control file.
  - The syntax `using backup controlfile` must be used.
  - Requires a RESETLOGS to open the database.
- The structure of the database should be matched by the control file.

  Match data file listings to sys.file$.

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Media Recovery: Database Recovery (continued)

Resetlogs and the consistency checks done during that operation are explained in detail in a separate lesson. The following is one of the checks that you should understand as far as media recovery is concerned

- When the database is opened with resetlogs option, a comparison is done between the data files in the controlfile and those in the data dictionary table, sys.file$. If a data file is found to be in sys.file$ but not in the controlfile then an entry will be created for it in the control file with the name of '`MISSINGnnn`', where `nnn` is the file_id in decimal. The file will be marked offline and needing recovery. This file, if it exists, can then be renamed to its correct name with the syntax:

  ```
  SVRMGR>alter database rename file 'MISSINGnnn' to
  '<file_name>';
  ```

- The data file is then recovered and brought online.

If a datafile is listed in the controlfile but is not listed in sys.file$ then it is removed from the controlfile.

When to Use Media Recovery

- When a data file has been restored from backup
- When a data file has been taken offline without a checkpoint
- When a backup control file is restored, even if the data files are current

  Will require the use of resetlogs option to open database

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

When to Use Media Recovery

Media recovery is required in these cases even if recovery can be accomplished using only the online logs.

If media recovery is invoked before a database is opened in a case where crash recovery would have been sufficient, then crash recovery is invoked automatically even if it has nothing to do.

Media Recovery Required

- Oracle establishes that media recovery is required by comparing the checkpoint structure in the datafile header with that in the control file.
- A mismatch in these structures indicates that redo needs to be applied.
- Even though detection of the need for media recovery is automatic, the actual recovery is done manually.

6-11 Confidential

Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Media Recovery Required

Even if media recovery can be resolved using the online redo log files only , it is still not done automatically.

If the database is more current than the controlfile, media recovery is required—to recover the control file, not the data files.

```
RECOVER DATABASE USING BACKUP CONTROLFILE;
```

Oracle's recovery layer code is designed to "tolerate" a discrepancy in checkpoint SCN between the file header and the control file record. These values could get out of synch if an instance failure occurs between the time the file header was updated and the time the control file transaction committed.

Checkpoint Structure

```
FILE HEADER:
    Software vsn=134217728=0x8000000, Compatibility Vsn=134217728=0x8000000
    Db Id=107678589=0x66b0b7d, Db Name='RCAT'
    Control Seq=2067=0x813, File size=512=0x200
    File Number=2, Blksiz=2048, File Type=3
Tablespace #1 - USERS  rel_fn:2
Creation   at   scn: 0x0000.00005ed7 06/30/98 14:53:36
Backup taken at scn: 0x0000.00000000 01/01/88 00:00:00 thread:0
 reset logs count:0x13ffc3fd scn: 0x0000.00000001 recovered at 01/01/88 00:00:00
 status:0x4 root dba:0x00000000 chkpt cnt: 8 ctl cnt:7
begin-hot-backup file size: 00x0000.00000000 01/01/88 00
Checkpointed at scn: 0x0000.00005f0e 06/30/98 15:20:36
 thread:1 rba:(0x2a9.3.10)
 enabled  threads:  01000000 00000000 00000000 00000000 00000000
    00000000
  00000000 00000000
Backup Checkpointed at scn: :00:00
 thread:0 rba:{0x0.0.0}
 enabled  threads:  00000000 00000000 00000000 00000000 00000000 00000000
  00000000 00000000
```

6-12

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Checkpoint Structure

Checkpoint structure components:

- Checkpoint SCN: Defines the committed version of the database prior to which all changes have been written to disk.
- Timestamp: The time this SCN was allocated
- Thread number: There is a one-to-one relationship between ‘enabled’ thread and the number of instances in your database. This is usually relevant only when Oracle Parallel Server is implemented.
- RBA: Redo byte address associated with this SCN. Used to quickly find the redo stream in the thread to start recovery.
- Enabled thread bitvec: A bit vector that outlines the number of enabled threads present in the database. The recovery process requires this information when undergoing thread switching during media recovery.

A mismatch in checkpoint counters between the data file header and the datafile’s control file record is used to detect when a backup data file or a backup control file has been restored.

Checkpoint Structure

- Differences in the checkpoint counter between the data file header and the control file indicate that recovery is needed.
- Checkpoint counter in file header < control file
  - Backup file has been restored
  - `ORA-01113 "file %s needs media recovery"`
- Checkpoint counter in file header > control file
  - Backup control file is being used
  - `ORA-01207 "file is more recent than controlfile - old controlfile"`

6-13

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Checkpoint Structure (continued)

In the first case, you should issue a `recover datafile` command and supply the necessary archived log files.

In the second case you should issue a `recover database using backup controlfile` command or recreate the control file if all the data files and log files exist.

Recovery with Archived Logs

- Automatically prompted for first log file needed.
- Can specify the `AUTOMATIC` option of `RECOVER` command to have Oracle apply logs without input from the database administrator.
- Uses log sequence number and the log_archive_dest and log_archive_format parameters to reconstruct log file names and location.
- Log files must be on disk.
- Oracle Parallel Server uses log history records to reconstruct order log usage.

6-14

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Recovery With Archived Logs

If it is not possible to determine the initial log sequence number, the user will have to guess and try logs until the right one is accepted. The timestamp from the media-recovery-start checkpoint is reported to aid in this effort.

Media Recovery: Threads

- The enabled thread bitvec indicates how many threads are enabled when recovery is started.
- Oracle performs thread-merged media recovery rather than single-threaded recovery, as during crash recovery.
- This means that all threads are recovered simultaneously.

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Media Recovery: Threads

A thread is a stream of redo. On single-instance databases there is usually only one thread although it is possible to specify more under certain circumstances. Multiple threads are usually only seen in situations where parallel server is being used. In parallel server there is generally one thread per instance and the thread applicable to that instance is defined in the `init<SID>.ora`.

Media recovery needs to find the start SCN to recover from. This is the lowest SCN of the data file header checkpoints, not the thread checkpoints.

In crash recovery, redo is applied from one thread at a time because only one instance at a time can dirty a block in cache; in between block modifications the block is written to disk. Therefore a block in a current online file can need redo for at most one thread. This assumption can not be made in media recovery as more than one instance may have made changes to a block so changes must be applied to blocks in ascending SCN order, switching between threads where necessary.

Threads

- If media recovery encounters a thread enable record then a new media recovery buffer is allocated for that thread and recovery continues , merging that thread with the other open threads.
- If recovery hits a thread disable record it will deallocate the buffer and stop recovery of that thread.
- The end of thread (EOT) flag in a redo log always indicates the log that contains the end of media recovery for that thread.

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

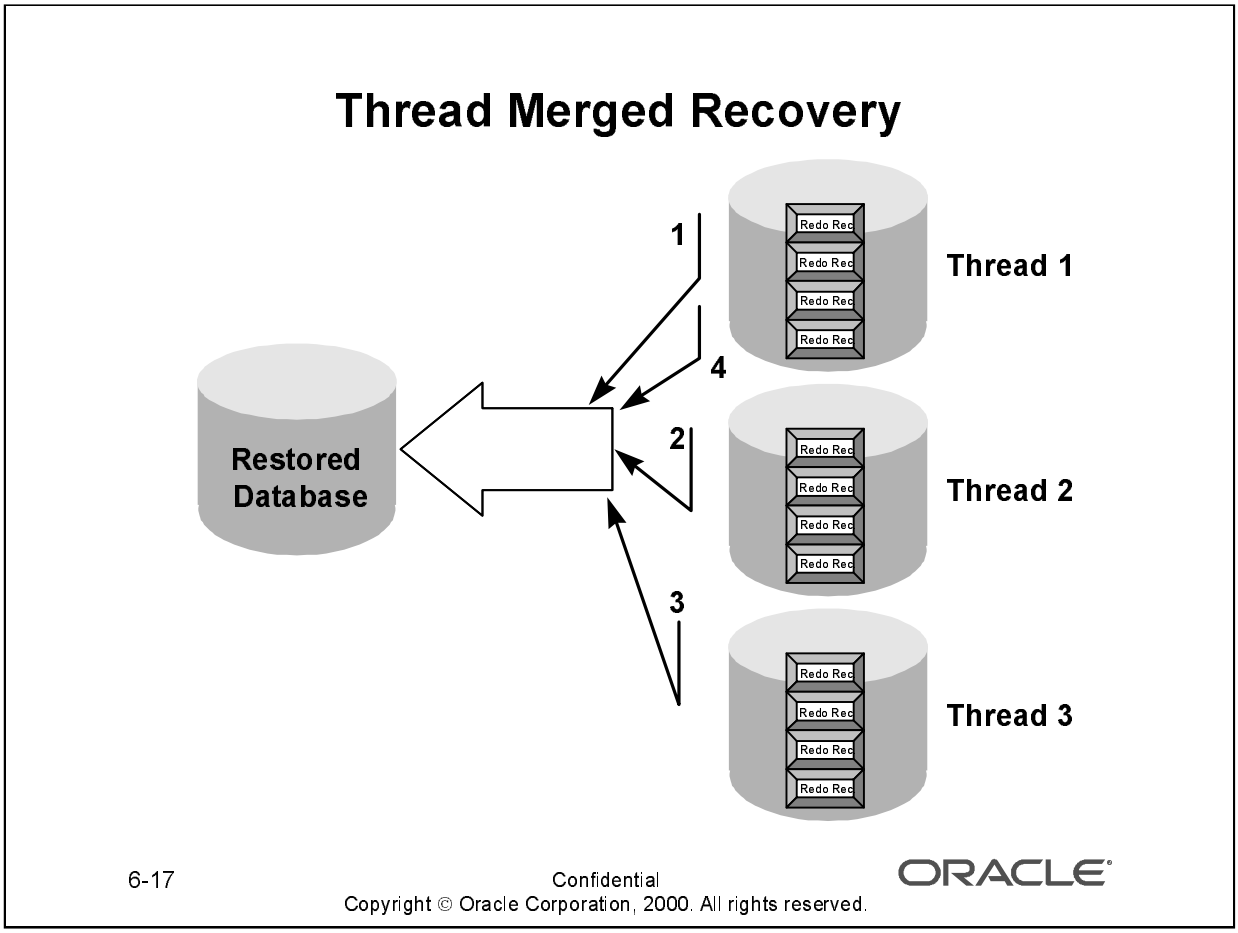
Threads

A thread enable record is written when a new thread of redo is created that is, when a new instance is started. A thread disable record is written when that instance is shutdown.

The EOT flag is stored in the log header. For this reason Oracle always advises against backing up online logs because of the danger of recovering to the wrong stopping point. If backup copies of the logs were restored with the data files then Oracle would recover to the EOT for each thread. The EOT recovered to, however, would be incorrect for full recovery to the current point in time.

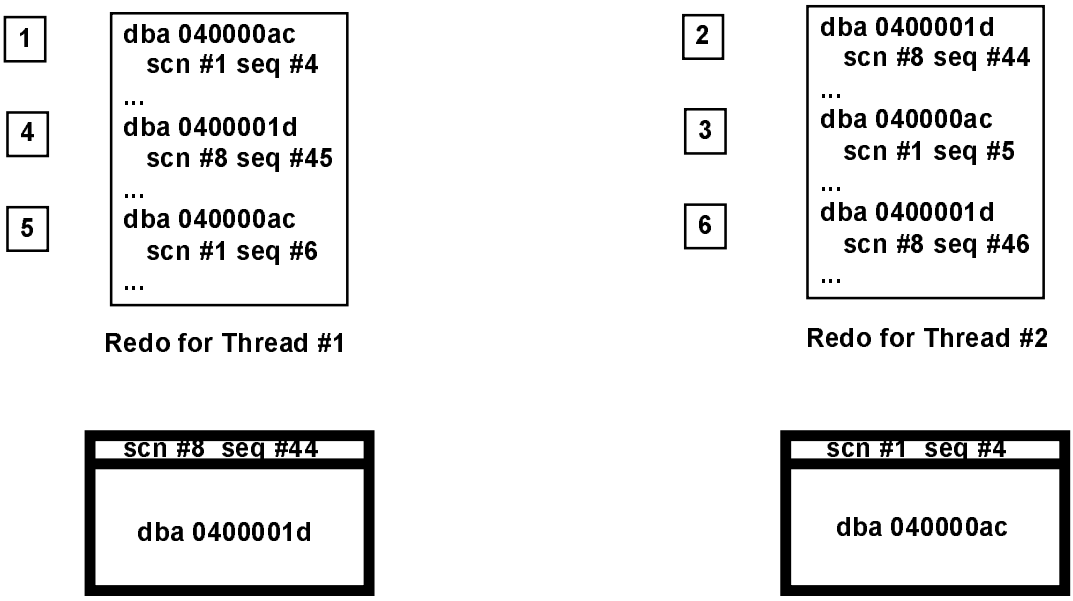
The EOT flag is highlighted below (this is from a dump of the redo log header):

```
descrip:"Thread 0001, Seq# 0000003514, SCN 0x04d900144312-
0xffffffffffff"
 thread: 1 nab: 0xffffffff seq: 0x00000dba hws: 0x1 eot: 1 dis: 0
```



Thread Merged Recovery

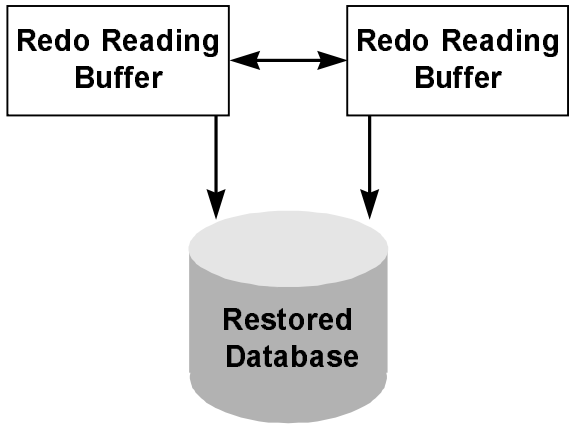
The numbers indicate how recovery progresses through the different threads, switching to the next thread in the list when it detects redo in the future of a block. Consider the following example:



Order of redo records applied... ('*' indicates thread switch)
1, *, 2, 3, *,4,5, ...(sometime in the future) *,6

Thread Merged Redo

Redo covers the enabling and disabling of threads. Buffers are opened/closed during recovery.



1. Recovery starts
2. Thread enabled
3. Thread switching
4. Thread closed

6-18

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Thread Merged Redo

The redo records associated with events such as thread enable, and other controlfile activity, have no data block address connected with them. These are essentially instructions to media recovery to perform some action. Here is an example of the change vector that enabled a new redo thread:

```
REDO RECORD - Thread:1 RBA: 0x00021b.00000002.0010 LEN: 0x004c VLD:
0x01
SCN scn: 0x0000.000a5c99 07/03/98 07:36:55
CHANGE #1 MEDIA RECOVERY MARKER SCN:0x0000.00000000 SEQ:  0 OP:17.2
thread enable: Checkpointed at scn: 0x0000.000a5c98 07/03/98
07:36:56
 thread:2 rba:(0x1.2.0)
 enabled  threads:  01100000 00000000 00000000 00000000 00000000
00000000
  00000000 00000000
END OF REDO DUMP
```

Media Recovery: Fuzzy Bits

- The database cannot be opened while any fuzzy bit is set in any data file.
- A file is fuzzy if it may have changes written to disk at an SCN later than its checkpoint.
- The fuzziness of a file is particularly important for incomplete recovery.

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Media Recovery: Fuzzy Bits

The fuzzy bits were explained in an earlier lesson. For review the three types of fuzzy bits are:

1. Hot backup fuzzy
2. Online fuzzy
3. Media recovery fuzzy

Media Recovery Fuzzy Bit

- Set when a file is in the process of having media recovery applied.
- The bit and the SCN are set when media recovery sees redo beyond the media recovery fuzzy SCN.
- It is fuzzy because media recovery buffers are not written after every change applied.

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Media Recovery Fuzzy Bit

The media recovery fuzzy SCN is advanced for all files in the recovery as soon as any redo is encountered. The SCN is set to the beginning of the next log that will be opened. If there are multiple threads of redo then the lowest next scn is used. The flag and SCN are cleared when the checkpoint is advanced by media recovery and recovery of this file is going to pause, for example when getting a log for a thread enable. The fuzziness is also cleared when recovery has applied all redo generated for the file.

Media recovery fuzzy bit prevents the file from opening before all the redo has been applied to that file.

Steps of Media Recovery

- Get the media-recovery-start SCN.

  The lowest data file header checkpoint SCN of all datafiles being recovered.
- The checkpoint structure at the media-recovery-start SCN is checked to see how many threads are enabled at that point.
- A buffer for reading redo is allocated for each enabled thread.
- A log is opened for each enabled thread of redo:
  - If online, it is opened automatically.
  - If archived, the database administrator is prompted for the log name.

6-21

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Steps of Media Recovery

If a data file's checkpoint is in its offline range, then the offline-end checkpoint is used instead of the data file header checkpoint as its media-recovery-start SCN.

The stop SCNs are saved for all data files being recovered. If finite, the highest stop SCN can be used to allow recovery to terminate without needlessly searching for redo beyond that SCN to apply.

Except in the case of cancel-based incomplete recovery and backup control file recovery, the next online log sequence is accessed automatically if it is on disk. If not, the user is prompted for the next log.

Steps of Media Recovery

- An exclusive media recovery lock is taken out on each data file being recovered by the process performing media recovery.
- Media Recovery fuzzy bit is set for each file.
- The checkpoint bitvec determines the initial enabled threads but this can change during the recovery process.
- Thread merged redo is applied.

6-22 Confidential

Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Steps of Media Recovery (continued)

Media recovery causes an exclusive MR lock on the files undergoing recovery. This prevents two processes from starting MR simultaneously. The lock is acquired by your session, which started the recovery.

If an instance has a file open then it has the MR lock in Share mode so that no other process can acquire it in exclusive mode to start MR on that file.

The MR fuzzy bit will prevent the files from being opened in an inconsistent state (that is, before all the redo has been applied).

Thread merged redo guarantees that the redo from all the enabled threads has been applied and has been applied in the right order—ascending SCN.

Steps of Media Recovery

- Media recovery checkpoints occur:
  - At log boundaries
  - When the stop SCN of a data file with a finite stop SCN is reached
- Recovery can therefore be stopped and restarted.
- If the stop SCN has been reached, the data file header checkpoints are advanced to that SCN.

6-23

Confidential

Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Steps of Media Recovery (continued)

Media recovery does checkpointing during its operation. These checkpoints are like normal checkpoints:

- All dirty (recovery) buffers are written to disk
- Checkpoint structures are updated with the checkpoint SCN in the datafile headers
- Media recovery can be restarted from this SCN

If a stop SCN has been reached for a file this means that the file was taken offline or made read-only at this SCN and has no redo beyond this point.

Completion of Media Recovery

- Complete MR continues until the EOT flag is encountered for all enabled threads of redo.
- Incomplete media recovery continues until the incomplete recovery SCN is reached.
- If a file has a finite stop SCN, then recovery of the file stops at that point.
- The media recovery fuzzy bit is cleared.
- File header and control file checkpoints are updated.

6-24 Confidential

Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Completion of Media Recovery

Media recovery will proceed until the current log in all threads has been applied. The EOT flag in the last log guarantees this condition. Because this flag is in the logfile header complete MR can be achieved even without the current controlfile.

The incomplete recovery SCN can be specified during incomplete recovery as a valid SCN or a date and time, or can be set at a log boundary during MR by entering 'CANCEL'. Transactions that committed at the specified SCN are rolled back.

There is no danger in backing up online redologs. The danger is restoring them accidentally over the current logs impairing recovery or forcing incomplete recovery when full recovery was an option. You can back up online redo log files along with a cold backup so that a complete recovery can be done using that set and an open can be done without RESETLOGS. More on this during RESETLOGS chapter!

With the database open, taking a data file offline produces a finite stop SCN for that datafile. This allows Oracle to determine when to stop media recovery of that file. Otherwise, there is no way of knowing when to stop because the Stop SCN will be infinite.

Incomplete Recovery

When a database is recovered to a point before the EOT for all enabled threads:

- **The point at which recovery should stop is defined by the DBA.**
- **Recovery can be stopped:**
  - **At a defined time or SCN**
  - **When a '*cancel*' request is issued**
- **All files must be recovered to the highest datafile checkpoint SCN.**
- **Hot backup files must be recovered past the point where END BACKUP was issued.**

6-25

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Incomplete Recovery

Even when incomplete recovery is performed, all files in the database must be recovered to the same point. The lowest point to which the files can be recovered is the highest checkpoint SCN among all the files in the database because only then will all the files be synchronized.

Media Recovery with Backup Controlfile

- A noncurrent control file is missing information on:
  - Current online logs
  - Data file stop SCN
  - Enabled threads
- Media recovery assumes an infinite stop SCN.
- Media recovery stops when the EOT marker in the redo logs is read.
- File headers provide recovery information for enabled threads and start SCNs.

6-26

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Media Recovery With Backup Controlfile

If the controlfile is from a backup then Oracle does not know for certain that files marked read-only will still be read-only when recovery is completed. If the file really is read-only, or if it is marked read-write when it is read-only at the end of recovery, manual work around is possible. If it is marked read-only but is really read-write then the DBA needs to recreate the control file to reflect the data file's read-write status.

V$ Views for Recovery

- V$ARCHIVED_LOG
- V$BACKUP
- V$CONTROLFILE
- V$DATAFILE, V$DBFILE, V$DATAFILE_HEADER
- VLOG, VLOGFILE
- V$RECOVER_FILE
- V$RECOVERY_FILE_STATUS
- V$RECOVERY_LOG
- V$RECOVERY_PROGRESS
- V$RECOVERY_STATUS
- V$THREAD

6-27 Confidential

Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

V$ Views for Recovery

V$ARCHIVED_LOG

Displays archived log information from the control file including archive log names

V$BACKUP

Displays the backup status of all online data files. Check that no files are in backup mode before starting recovery

V$DATAFILE, V$DBFILE

- Data file information from the control file
- V$DBFILE gives the path names for the data files
- Check that all files needing recovery are online before starting incomplete recovery

V$DATAFILE_HEADER

Data file information from the data file headers

VLOG, VLOGFILE

- Contains log file information from the control files.
- V$LOGFILE gives the file names

V$RECOVER_FILE

- Displays status of files needing media recovery
- Gives the start SCN for each file needing recovery
- Can be used to make sure all files are recovered to a consistent SCN

V$ Views for Recovery (continued)

V$RECOVERY_FILE_STATUS

- Contains one row for each data file for each RECOVER command
- Contains useful information only for the Oracle process doing the recovery
- Displays status of IN RECOVERY, CURRENT, or NOT RECOVERED

V$RECOVERY_LOG

- Lists information about archived logs that are needed to complete media recovery
- Information is derived from the log history view, V$LOG_HISTORY

V$RECOVERY_PROGRESS

- Can be used to track database recovery operations to ensure that they are not stalled, and also to estimate the time required to complete the operation in progress
- V$RECOVERY_PROGRESS is a subview of V$SESSION_LONGOPS

V$RECOVERY_STATUS

- Contains statistics of the current recovery process
- Contains useful information only for the Oracle process doing the recovery
- Lists recovery checkpoints, status of logs used previously and current control status (NEED LOG; LOG REUSED; THREAD DISABLED)

V$THREAD

- Thread information from the control file
- Lists status, enabled, instance, checkpoint SCN, and timestamp

Useful Recovery Tracing Events

- 10228
  - Traces the application of redo
  - Has the same output as 'alter system dump logfile'
- 10015
  - Shows undo being applied during trx recover.
  - Can be used to increase SC.
- 10218

  Shows the undo block addresses for the undo applied during recovery

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

Summary

In this lesson, you should have learned about the following:

- Media recovery requires intervention by the database administrator.
- Media recovery can be complete or incomplete.
- To perform incomplete recovery *all* data files must be restored before recovery is started.
- Media recovery uses thread-merged redo application.
- Media recovery is successful if the database is in a consistent state once media recovery has completed.

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

References

- *Oracle 7 Internals - Module 2* by Lawrence To , Oracle Internal Publication 1996
- Oracle8 Documentation Manuals
- *Oracle8 Backup and Recovery Handbook by* Rama Velpuri & Anand Adkoli, Oracle Press, 1998
- "EVENT: Common Numeric Events" UK Note: 28719.1
- "*Oracle8 v8.1 Recovery Outline*", Andrea Borr, Bill Bridge, & Ron Obermarck, Sept. 1999

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

7

Resetlogs

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Schedule:	Timing	Topic
	45 minutes	Lecture
	30 minute	Lab
	75 minutes	Total

Objectives

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

- When and why resetlogs is used
- How resetlogs works
- The impact of using resetlogs on future recovery
- The implications of forcing the database open with underscore parameters

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Usage

The resetlogs option is required when opening the database after one of the following has been performed:

- **Incomplete media recovery**
- **Recovery using a backup control file**
- **Recovery with a control file created with the resetlogs option**

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Usage

When using incomplete recovery to recover to a point in time before a data file was dropped, the control file must contain the file to be recovered. Therefore you need to either use a backup control file or recreate it.

Usage: Incomplete Media Recovery

- During incomplete media recovery the most recent redo is not applied. It is necessary to clean these records from the redo logs.
- Resetlogs purges this redo from the redo logs before opening the database.

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Usage: Incomplete Media Recovery

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

Usage:
Backup Control File

- **Backup control files will not contain the most recent control file information and may be out of sync with the data and redo logs.**
- **Resetlogs will resync the files after a recovery with a backup control file.**

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Usage: Backup Control File

```
SQL> alter database backup controlfile to 'control.bak';
```

or copy the control file when the database is closed.

```
SQL> recover database using backup controlfile <options> ;
```

options = until CANCEL
until SCN
until TIME

Usage:
Create Control File with Resetlogs

- Used when all the copies of the control file and all members of the redo logs are lost.
- Resetlogs purges or re-create the online logs during the recreation of the control file.
- If only the control file is lost, and the redo logs are current, then use create control file.. NORESETLOGS

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

!!! Warning !!!

- **It is normally not possible to recover past a resetlogs.**
- **You should take a complete database backup immediately after database database is opened with the resetlogs option.**

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

!!! Warning !!!

Example

1. Backup taken at 8 a.m.
2. Table dropped at 11 a.m.
3. Decision made to recover the table using point in time recovery.
4. Backup from 8 a.m restored, database is rolled forward till 10.59 and opened with resetlogs option at noon.
5. Disk crash at 4 p.m causes a data file failure.
6. Last backup from 8 a.m cannot be used.
7. Backup should have been taken at noon after the open with resetlogs.

Note: The guidelines necessary for recovery past resetlogs is possible are discussed later in this lesson. For more information please refer to the note :

"7.3 rollforward through resetlogs" by Lawrence To in the site `http://spicy.us.oracle.com/classes/`.

Mechanics of Resetlogs

- How:
  - Record the current SCN and a counter known as ‘resetlogs data’.
  - Store these values in the control file, the data file headers, and the redo log headers.
  - Identifies when execution of database open with resetlogs took place.
- Why:
  - Resetlogs ensures that old files are not. introduced into the database.

7-8

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Mechanics of Resetlogs

Resetlogs prevents introduction of old files into the database which is also the reason why old files (from before resetlogs) cannot be used in recovery after the database is opened with resetlogs option. The exception to this is read only and offline normal files whose file headers will not have the resetlogs data.

Resetlogs Actions

- All online log file information is reset in the control file.
- One log file is picked to be the current log file for every enabled thread.
- Log header is updated to Log Seq # 1.
- The file headers for all online datafiles are updated with the new checkpoint and new ‘resetlogs data’.
- Offline datafiles are marked needing media recovery.

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Resetlogs Actions

If all files for a tablespace are offline prior to resetlogs, then the files in this tablespace are marked as needing recovery afterwards. The tablespace can be brought online after the database is opened with resetlogs. If only one file of a tablespace is offline prior to resetlogs but the tablespace itself is online, the file will need media recovery which cannot be performed past a resetlogs.

Resetlogs Consistency Checks

- On open, file$ and the control file are cross checked.
- If a file exists in file$ but not in control file a MISSINGxxx place holder is created in the control file.
- If a file exists in control file but not in file$. File entry is removed from control file.

  (Remember! file$ is always right).
- If a datafile was added during the recovery period and recovery is using an old control file, then the file will have to be recreated.

7-10 Confidential

Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Resetlogs Consistency Checks

In case of mismatch file$ is considered right because the entries are never removed from file$. If the file is read only or offline normal then the file can be accessed after the open by renaming the MISSINGxxx to the filename.

```
SVRMGR> ALTER DATABASE RENAME FILE 'MISSINGxxx' to '/...';
```

If a datafile was added to the database after the last backup was taken and the current control file is missing, then restore the backup set and the old control file. Start recovering the database. At the point the new datafile was added, recovery will error out with:

```
ORA-00283: Recovery session canceled due to errors
ORA-01244: unnamed datafile(s) added to controlfile by media recovery
ORA-01110: data file 5: '/u01/oracle/or8i/testtbs.dbf'
```

Recreate the missing datafile by first selecting from v$datafile the created name of the file, and then issuing the command:

```
ALTER DATABASE CREATE DATAFILE 'UNNAMED0005' AS
'/u01/oracle/or8i/testtbs.dbf';
```

Then rename the file:

```
ALTER DATABASE RENAME FILE 'UNNAMED0005' TO
 '/u01/oracle/or8i/testtbs.dbf';
```

Then continue with the recovery of the database.

Alternatively the file could have been added to the database before starting recovery. Note: 29430.1

Resetlogs Consistency Checks

Before opening with resetlogs the following conditions must be true:

- **Datafile compatibility must not be greater than the current compatible setting for the database.**
- **Offline immediate files must be onlined or dropped.**
- **Files must not have the fuzzy bit set and must all be at the same checkpoint SCN.**

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Resetlogs Consistency Checks (continued)

Consistent Checkpoint SCN guarantees that the files will be recoverable to the same PIT called the resetlogs SCN. Fuzzy bit set means that the file has updates from after the checkpoint SCN.

Exception to the Rule

1. Read-Only files will have no redo between checkpoint SCN and resetlogs SCN.

Failure of Consistency Checking

- The first two points (on previous page) can be easily remedied sometimes.
- If the consistency checking fails because the data files are not at the same point in time then the database cannot be guaranteed to be consistent hence it will not open.

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

Failure of Consistency Checking

Example

ORA-01113 file SYSTEM needs media recovery.

// Cause: An attempt was made to open a data file that is in need of media recovery.

// Action: First apply media recovery to the data file identified in the message, then retry the operation.

This could indicate either that the file is not consistent with other files, or that it still has its hot backup fuzzy bit set.

Failure of Consistency Checking

In situations where database has to be opened without the consistency checks:

- **Use _allow_resetlogs_corruption.**
- **Bypasses the consistency checks so files can have different SCNs or have the fuzzy bit set.**
- ***DATABASE WILL BE IN AN INCONSISTENT STATE* and must be rebuilt!**

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Failure of Consistency Checking (continued)

The `_allow_resetlogs_corruption` parameter is unsupported and should be used with extreme caution and in consultation with Senior Analysts. The customer should be made fully aware of the implications of setting this parameter and should be asked to make a decision after being informed of all the ramifications.

Consequences of Resetlogs

- Cannot recover beyond the point of issuing resetlogs in many cases
- Any files offline during recovery will cause the ALTER DATABASE OPEN RESETLOGS command to fail:

```
ORA-01245: offline file 8 will be lost
  if RESETLOGS is done
ORA-01110: data file 8:
  'D:\ORACLE\RBS.DBF'
```

Such files will have to be dropped or recovered in order to proceed.

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Consequences of Resetlogs

If you perform incomplete recovery on a database and a file is offline, but still needs recovery, a warning message will appear:

- ORA-01547: warning: RECOVER succeeded but OPEN RESETLOGS would get error below
- ORA-01245: offline file 8 will be lost if RESETLOGS is done
- ORA-01110: data file 8: '/dev/oracle/product/V815/dbs/indx01.dbf'

Rollforward Through Resetlogs Example

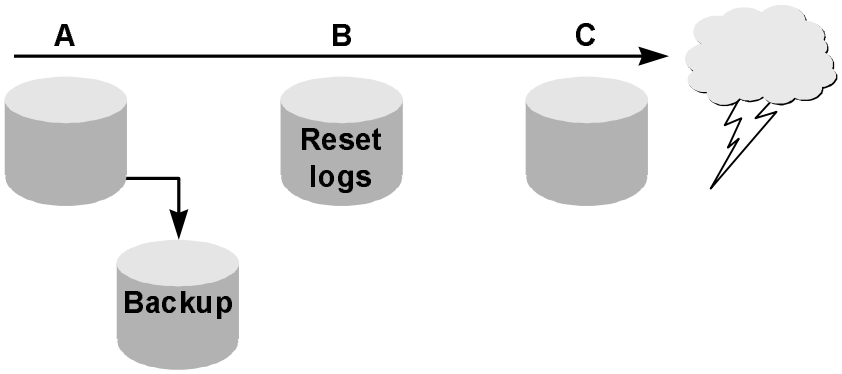
- Database was opened with resetlogs on a 24x7 system.
- Immediately afterwards a hot backup was started.
- A disk failure lead to the database crashing before it was completed.
- Can this be database be recovered now?
- From 7.3.3 onwards this can be achieved in the right circumstances.

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

Rollforward Through Resetlogs

The following slides assume the following:

- **Backup was taken at point A.**
- **Resetlogs at point B.**
- **The current database is at point C.**



7-16

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Rollforward Through Resetlogs

Requirements for performing a rollforward through resetlogs:

- A consistent backup of the database, taken before you open the database with the resetlogs option (includes the control file)
- A control file that is valid after you open the database with resetlogs option.

Recovering Through Resetlogs

- Take a backup of the control file at point C.
- Restore the data files from point A (excluding read-only data files).
- Restore the control file from point A to a different location from point C and update the init.ora to point to the new control file.
- Startup mount and recover the database using backup control file until change number found in the alert log from previous open RESETLOGS.

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Recovering Through Resetlogs

- When recovery completes perform a normal shutdown.
- Change the init.ora to point back to the control files at point C.
- Startup mount and then do a complete recovery using the current control file.

The database should be opened WITHOUT resetlogs because the redo logs are current on the system.

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Recovering Through Resetlogs (continued)

All the log files must be available from before RESETLOGS was issued and afterwards (starting over with log sequence #1) for the database to be recovered.

Forcing The Database Open

- Exceptional circumstances.
- Database is not consistent.
- Use hidden parameters that bypass consistency checks.
- Applies to a particular stage of recovery:
  - Rollforward
  - Rollback
- Not supported! Not tested! Not guaranteed to work!

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

Forcing The Database Open

Unsupported means that:

- The product was not designed to work with these parameters.
- The parameters have not been tested, and so you may be the first to try them in this combination!
- The consistency of the database will always be in-doubt afterwards.
- There is a high risk of unexpected and undesirable side effects.

_offline_rollback_segments

- TRN TBL is not checked on startup.
- RBS can be dropped with active transactions.
- Provides read access to specified rollback segments, but not write access.
- Needs to be used until all data blocks with changes in the specified rollback segments have been recovered.
- Database has to be rebuilt after using this parameter.

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

_offline_rollback_segments

Rollback segments specified in _offline_rollback_segments are marked offline. Their TRN TBLs are not read on startup, and they are not available for future updates or transactions.

Verify that all blocks have been recovered by determining which objects are affected from a transaction table dump, and then perform export on all involved objects. Because `_offline_rollback_segments` allows the rollback segments to be read, a read-consistent version of the table can be created. However, the uncommitted transactions in the table cannot be rolled back, so the objects have to be treated as if corrupted. Once all active transactions have been cleaned up, the rollback segments can be dropped and the parameter removed.

See WEB IV Note 16736.1: PARAMETER:OFFLINE_ROLLBACK_SEGMENTS

_corrupted_rollback_segments

- Used when specified rollback segments are not available at all, or are not readable (RBS header block corruption).
- Any transactions stored in the specified RBS are treated as committed and complete. TRN TBL is not accessed.
- Leads to inconsistencies throughout the database.
- The database must be rebuilt after using this parameter.

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

_corrupted_rollback_segments

See PRE 1025624.6 “V7: ROLLBACK SEGMENT CORRUPTION - HOW TO RECOVER” or Note: 34004.1 “PARAMETER: INIT.ORA: CORRUPTED_ROLLBACK_SEGMENTS”

_corrupt_blocks_on_stuck_recovery

- Can be used if an ORA-600 [3020] is preventing recovery from proceeding
- Usage is to set this parameter to the number of times to corrupt a block:

  _corrupt_blocks_on_stuck_recovery = n
- Allows recovery to proceed on blocks where the redo corresponds to a more recent version of the block
- Can corrupt the database

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

_corrupt_blocks_on_stuck_recovery

The most common reason for getting an ORA-600 [3020] is recovering the database, opening the database, and then attempting to recover the database further without restoring the data files again.

This parameter may not be implemented on all versions of the RDBMS, or on all platforms.

If you are performing parallel recovery, then each slave process can skip <n> number of blocks.

Refer to WEB IV Notes:

- 16527.1 PARAMETER:CORRUPT_BLOCKS_ON_STUCK_RECOVERY
- 30866.1 OERI:3020

_allow_resetlogs_corruption

- **Bypasses the consistency checks so files can have different SCNs or have the fuzzy bit set.**
- **DATABASE WILL BE IN AN INCONSISTENT STATE and must be rebuilt!**

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

_allow_resetlogs_corruption

For more information, see WEB IV Note: 33969.1
PARAMETER:ALLOW_RESETLOGS_CORRUPTION

This parameter is only effective if you issue:

```
ALTER DATABASE OPEN RESETLOGS;
```

If you set the parameter and just OPEN the database (without RESETLOGS) the parameter has no effect.

_allow_read_only_corruption

- Allows the database to be opened in read-only mode even if it is corrupt
- Available only in 8*i*
- Data will not be consistent

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Summary

In this lesson, you should have learned about the following:

- Use resetlogs when doing recovery to a point in the past or if online redo logs are not current.
- Always take a backup after using resetlogs.
- Use resetlogs with caution as it can lead to further recovery not being possible.
- There are ways to force the database open, but each has its own set of risks.

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

References

- `www-sup.ca.oracle.com/spicy/handouts.html`
  - *Oracle 8.1 Recovery Outline* - Andrea Borr, Bill Bridge and Ron Obermarck
  - `Resetlog.ppt` - Cynthia Yip and Richard Exley
  - `73reset2.ppt` - Lawrence To
- WEB IV
  - Note 41399.1: TECH: Summary for forcing the database open with `_ALLOW_RESETLOGS_CORRUPTION`
  - Note 29016.1: QREF: Recovery Scenarios - Quick Reference

7-26

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

References

See also the *Oracle8i Server Backup and Recovery Guide*.

8

Point-In-Time Recovery

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Schedule:	Timing	Topic
	45 minutes	Lecture
	45 minutes	Lab exercise
	90 minutes	Total

Objectives

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

- Perform object Point-In-Time-Recovery (PITR) in Oracle7 and Oracle8
- Describe TSPITR methodology
- Perform Tablespace Point-in-Time-Recovery (TSPITR)
- Use different views for TSPITR
- Describe the limitations of TSPITR

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

What Is Object Point-In-Time Recovery?

- Returning an object to a previous state.
- For example, you might use object PITR after performing:
  - Drop table
  - Truncate table
  - Erroneous committed data changes

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

What Is Object Point-In-Time Recovery?

Requirements:

- Backups of all files that are needed to recover specified objects. Tablespaces other than the SYSTEM tablespace, that do not need recovery are not necessary.
- All redo needed to recover the backup set to a previous point in time is available.
- Space is available to store copies of the files being recovered.
- There is enough memory available to start a recovery instance.
- DBA is able to make the table logically consistent with other database structures in case of damage caused by a user or by applications.

Object PITR

- Works at an object level.

  That is, one or more tables.
- The user is responsible for restoring and maintaining integrity constraints, sequences, indexes, and so on.
- Can be used in all versions of Oracle.

8-4

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Object PITR

Recovery Time

The time taken using this method depends on the time it takes to set up and recover a backup instance, plus additional time for the export/import of the table data. When dealing with very large tables (as in a VLDB environment) this approach may not be feasible.

Methodology

1. Restore a subset of the primary database to a separate location.
2. The subset needs the following:
   - All SYSTEM tablespace datafiles.
   - All ROLLBACK SEGMENT tablespace datafiles.
   - All CONTROL FILES.
   - All datafiles from the tablespaces holding extents for objects being recovered.

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Methodology

Note that when using this method, you will need to re-create the indexes for the recovered tables, either with import, or through SQL—using CREATE INDEX command, after the import is finished.

It is strongly advised that the restore of the backup take place on a different machine from the production database to avoid any possible overwrites of production data. This also helps to avoid ORACLE_SID conflicts.

Methodology

3. Use standard recovery techniques to recover the subset to a point in time.
4. Open database with resetlogs option.
5. Export the objects from the subset database.
6. Import the objects into the primary database.

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

Methodology (continued)

If the database backup was restored on the same machine as the production database, be extremely careful that the production online logs are not overwritten by the `ALTER DATABASE OPEN RESETLOGS` command.

TSPITR: Basics

- The tablespace is recovered to a previous point-in-time on a clone database.
- The datafiles from the recovered tablespace are copied to the primary database.
- The primary database is then modified by import to accept these files.
- Available only in release Oracle8 or later.

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

TSPITR: Basics

The recovery to point in time on the clone database is done using normal recovery techniques.

The metadata for the recovered tablespaces on the clone database is exported using new export parameters (`, TABLESPACES`)

The primary database is modified by import using a new parameter (`POINT_IN_TIME_RECOVER`), which imports only metadata about the recovered tablespaces, so is much faster than the old method.

TSPITR: Caution!

- TSPITR is complicated.
- It should not be attempted without practice.
- There is a potential for data loss.
- Customers should understand the implications before using it.

8-8

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

TSPITR: Caution!

TRPITR Caution

TSPITR should not be attempted for the first time in a production environment.

TSPITR should not attempt for the first time when under pressure to produce results.

Potential Data Loss

The most important thing to understand about TSPITR is that it is TABLESPACE recovery; once you have recovered to a point in time all objects in the tablespace will be as they were at that time (not just the table(s) needing recovery). Any objects or data added to the tablespace after the chosen recovery time will no longer exist after TSPITR is performed.

It is very important to take this into account when deciding whether to use TSPITR or not. If TSPITR is not suitable then object point-in-time recovery can be used.

TSPITR: Terminology

- Primary: The database which holds the tablespaces to be recovered.
- Clone: A copy database used for point-in-time recovery of the tablespaces. It is also called the auxiliary instance when using RMAN.
- RECOVERY SET: All tablespaces that need point-in-time recovery.

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

TSPITR: Terminology

Auxiliary Set

- **Backups of the `SYSTEM` tablespace, any rollback segment tablespaces and the control files.**
- **Backup of `TEMPORARY` tablespace is optional.**
- **The control file backups must be from the same time as the other file backups and should have been made with the command.**

```
ALTER DATABASE BACKUP CONTROLFILE TO
   '<filename>';
```

8-10

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

TSPITR: Terminology

Auxiliary Set

Including the TEMPORARY tablespace as part of the recovery set is mentioned only because export does do some minimal sorting operations and will need sort space. If the CLONE database is to be discarded and has some room available in the SYSTEM tablespace for sorting, then it does not need to be included.

If the control file backed up by normal O/S copy and was not created with the command

`ALTER DATABASE BACKUP CONTROLFILE TO 'file_name',`

then the problem can be worked around. This is discussed later in the chapter.

TSPITR: Dependency Rules

The recovery set must consist of all data files that contain extents for:

- **Tables or clusters to be recovered, and their indexes**
- **All partitions and subpartitions of a partitioned table**
- **Primary and foreign key relationships**
- **All elements of a LOB : segment, index, and locator**

8-11

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

TSPITR: Dependency Rules

If tables and indexes are in different tablespaces, then both tablespaces will have to be included in the recovery set. This is what is meant by fully contained.

TSPITR: Dependency Rules

- An Oracle8*i* recovery set should not contain:
  - Replicated master tables
  - Snapshot logs
  - Snapshot tables
- Oracle8 may have additional restrictions for:
  - Tables with VARRAY columns
  - Nested tables
  - Tables with external BFILES
  - Index-only tables (IOTs)

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

TSPITR: Check Views

- **TS_PITR_CHECK**

  Objects that violate TSPITR dependency rules
  - **OBJ<n>_OWNER VARCHAR2(30)**
  - **OBJ<n>_NAME VARCHAR2(30)**
  - **OBJ<n>_SUBNAME VARCHAR2(30)**
  - **OBJ<n>_TYPE VARCHAR2(30)**
  - **TS<n>_NAME VARCHAR2(30)**
  - **CONSTRAINT_NAME VARCHAR2(30)**
  - **REASON VARCHAR2(30)**

  (< n> is 1 or 2)

8-13

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

TSPITR: Check Views

TS_PITR_CHECK

This view is created from two others views: STRADDLING_RS_OBJECTS and TS_PITR_INFO (see CATPITR.SQL)

Assuming a recovery set consisting of tablespaces TS_A and TS_B, any dependency violations can be checked using the following query:

```
SELECT   *
FROM     SYS.TS_PITR_CHECK
WHERE    (TS1_NAME IN ('TS_A','TS_B')
     AND
     TS2_NAME NOT IN ('TS_A','TS_B'))
OR   (TS1_NAME NOT IN ('TS_A','TS_B')
     AND
     TS2_NAME IN ('TS_A','TS_B'))
```

TSPITR: Check Views

- **TS_PITR_OBJECTS_TO_BE_DROPPED**
  Objects that are dropped as a result of TSPITR
  - **OWNER** **VARCHAR2(30)**
  - **NAME** **VARCHAR2(30)**
  - **CREATION_TIME** **DATE**
  - **TABLESPACE_NAME** **VARCHAR2(30)**

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

TSPITR: Check Views (continued)

TS_PITR_OBJECTS_TO_BE_DROPPED

Because TSPITR operates at the tablespace level, anything created after the recovery time specified during TSPITR is lost.

For example, assuming a recovery set of tablespaces TS_A and TS_B and a proposed recovery time of 12-JUN-1998 12:34:56 the following query can be run to see what objects (if any) will be lost:

```
SELECT   *
FROM     SYS.TS_PITR_OBJECTS_TO_BE_DROPPED
WHERE    TABLESPACE_NAME IN ('TS1','TS2')
AND CREATION_TIME
             > TO_DATE(     '12-JUN-1998:12:34:56',
         'DD-MON-YY:HH24:MI:SS')
```

Also any data added to objects in the recovery set after the recovery time is lost. There is no Oracle-provided method of determining what this data is.

Steps Involved in TSPITR

- High-level description:
  - Create a clone database.
  - Recover a subset of the clone database.
  - Copy the datafiles to the target database.
  - Export the tablespace meta data from the clone.
  - Import the tablespace meta data into the primary.
- The following slides work through the individual steps of TSPITR.

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

Prepare the Primary

- Query TS_PITR_CHECK
  - Make sure there are no dependency issues
  - Note the objects violating the dependency rules because they will need to be dropped on the CLONE database as well.
- `ALTER SYSTEM ARCHIVE LOG CURRENT`

  In case the recovery process needs changes from the current redo log.
- `ALTER ROLLBACK SEGMENT <name> OFFLINE`

  For any rollback segments in the RECOVERY SET.
- `ALTER TABLESPACE <name> OFFLINE`

  For all tablespaces in the RECOVERY SET.

8-16 Confidential

Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Prepare the Primary

ARCHIVE LOG CURRENT

Flush all changes for the data files to disk and archive the changes.

ALTER ROLLBACK SEGMENT <name> OFFLINE

You cannot take a tablespace with active rollback segments.

ALTER TABLESPACE <name> OFFLINE

This is done to stop any further changes to data in the recovery set as these changes will be lost when TSPITR is complete. If some data in the recovery set is not corrupt and users wish to access it then the tablespaces can be made read-only by using the command `ALTER TABLESPACE <name> READ ONLY`, but they will have to be made offline before the datafile copy phase of TSPITR begins.

When taking tablespaces offline, you can only use `OFFLINE NORMAL` or `OFFLINE FOR RECOVER` (if the files are not available). Doc Bug 513825.

Note: To use TSPITR the database must be in `ARCHIVELOG` mode. You cannot take a tablespace offline if the database is in `NOARCHIVELOG` mode since doing so will make the tablespace unrecoverable.

Prepare the CLONE

- **Restore the auxiliary set.**

  Must be a different location than primary.
- **Restore the recovery set.**

  This can be restored over the primary as long as the tablespaces are offline in the primary.
- **Restore the CONTROL FILES.**
  - **These must have been made by the command `ALTER DATABASE BACKUP CONTROLFILE TO '<filename>'`.**
  - **Must be more current than the files in backup.**

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

Prepare the CLONE

Restore the Recovery Set

If it is decided to have the clone recovery set in the same location as the primary then the tablespaces must be altered OFFLINE and not merely READ ONLY.

!!! Take great care if you chose this option as the potential for errors resulting in serious damage is much higher !!!

Restore the Control Files

Made with `ALTER DATABASE BACKUP CONTROLFILE TO '<filename>'`.

In Oracle8 checkpoints are stored in control files as well as the datafiles. This gives the potential for error if the control file being used was not backed up at the same time as the datafiles. The way to avoid these errors is to ensure that the control file backup is taken at the same time as the datafile backups.

Configure the CLONE

- Copy parameter files from the primary.
- Configure the init.ora file for the clone (`initCLONE.ora`).
  - CONTROL_FILES: Must point to a new location.
  - LOCK_NAME_SPACE: Set to the SID for the clone to allow the database to start without error.
  - Use DB_FILE_NAME_CONVERT and LOG_FILE_NAME_CONVERT if needed.

8-18

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Configure the CLONE

Configuration Files

These are the `init.ora`, `config.ora`, and `orapwd` files. If resources are an issue, SGA settings can be reduced on the clone database.

CONTROL_FILES

Make sure this points to the location of the clone database control files, not the control files for the primary database.

LOCK_NAME_SPACE

This parameter should be set to a unique value to avoid problems mounting two databases of the same name on the same machine.

DB_FILE_NAME_CONVERT = “FROM”,”TO”

This parameter can be used to convert parts of the data file names substituting the string specified in the TO clause for the string specified in the FROM clause.

LOG_FILE_NAME_CONVERT = “FROM”,”TO”

This parameter can be used to convert parts of the LOGFILE names substituting the string specified in the TO clause for the string specified in the FROM clause.

Start the CLONE

- Set the OS environment to access the CLONE
  - ORACLE_SID
  - ORACLE_HOME (possibly the same)
- `STARTUP NOMOUNT PFILE=initCLONE.ora`
- `ALTER DATABASE MOUNT CLONE DATABASE`
  - All datafiles will be offline
  - Database will be in `NOARCHIVELOG` mode

8-19

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Start the CLONE

ALTER DATABASE MOUNT CLONE DATABASE

After this step ALL datafiles will be offline. This is done for two reasons:

1. Many of the files will probably not be needed; the recovery set is typically a minority of the database as a whole.
2. It cannot be assumed that the files will be present even after the conversion is performed by the FILE_NAME_CONVERT parameters

When Control file was not backed up with ALTER command

If the control file being used was not created with `ALTER DATABASE BACKUP CONTROLFILE TO '<filename>'` mounting the CLONE database will fail with ORA-1696. If you get this error and none of the conditions from the error message seem to apply you can work around it as follows:

1. `STARTUP NOMOUNT PFILE=initCLONE.ora`
2. `ALTER DATABASE MOUNT`

   Note: Do not use CLONE keyword here.
3. `ALTER DATABASE BACKUP CONTROLFILE TO '<filename>';`
4. `SHUTDOWN`
5. Replace the original control file with that created in step 3 above
6. `STARTUP NOMOUNT PFILE=initCLONE.ora`
7. `ALTER DATABASE MOUNT CLONE DATABASE;`

Prepare CLONE for Recovery

- Check status of files in V$DATAFILE
- `ALTER DATABASE RENAME FILE '<name>'`
  If any files need renaming
- `ALTER DATABASE DATAFILE '<name>' ONLINE`
  For all files in the recovery set and auxiliary set

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Prepare CLONE for Recovery

ALTER DATABASE RENAME FILE 'name'

This step is needed for any files in the recovery set or auxiliary set which have not been fully renamed by the parameter DB_FILE_NAME_CONVERT parameter.

IMPORTANT: - If you do not rename files correctly you could be overwriting production files!

ALTER DATABASE DATAFILE 'name' ONLINE

Files have to be online to be recovered. Also, the export phase of TSPITR does not work unless all datafiles in the recovery set are online.

Recover and Open CLONE Database

- Recover to a point in time with `'USING BACKUP CONTROLFILE'` clause:
  - UNTIL CANCEL
  - UNTIL CHANGE <scn>
  - UNTIL TIME 'YYYY-MM-DD:HH24:MI:SS'
- Check V$LOGFILE to make sure redo log filenames are correct.
- Open the CLONE database:
  - `ALTER DATABASE OPEN RESETLOGS`
  - Only SYSTEM rollback segment will be available
  - No DML allowed

8-21

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Recover and Open CLONE Database

Recover to Point-in-Time

If the `RECOVER DATABASE` statement reports that recovery is not needed check that the `RECOVERY SET` datafiles have been brought online.

ALTER DATABASE OPEN RESETLOGS

Before opening the database with resetlogs option, carry out one last check that the clone database has the correct redo log file names (not the production log file names):

```
SELECT  NAME FROM V$LOGFILE;
```

Prepare the CLONE for Export

Query TS_PITR_CHECK on CLONE database:

- **Drop any objects that have already been dropped on the primary.**
- **Make sure that there are no other dependency issues in the backup set.**

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Prepare the CLONE for Export

For the export phase to succeed there can be no references out of or into the recovery set. All data to be recovered has to be completely contained within the recovery set of tablespaces. You can verify this is true by looking at `TS_PITR_CHECK` on the clone database.

Likewise, import will not succeed if the recovery set is not fully contained.

There are some problems with the information displayed in this table in Oracle8.0.

See bugs: 494744, 465207, and 486730.

Export From the CLONE

- EXP has extra parameters for TSPITR:
  - POINT_IN_TIME_RECOVER=Y
  - TABLESPACES=list
- Example:
  - For two tablespaces TS_A and TS_B

```
exp sys/<pwd> point_in_time_recover=y
tablespaces='TS_A','TS_B'
```

  - The export must be run as SYS or it errors out with ORA-29303

8-23

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Export From the CLONE

The `TABLESPACES` parameter originated in Oracle8.0.2, but was changed to `RECOVERY_TABLESPACES` (bug 465257). In 8*i* it was changed back to `TABLESPACES` (bug 671845). This parameter can now be used for both TSPITR and transportable tablespaces (with `TRANSPORTABLE_TABLESPACES = Y` instead of `POINT_IN_TIME_RECOVER`)

Export: POINT_IN_TIME_RECOVER=Y

This export is a modified version of a normal export. It extracts:

1. Dictionary metadata for all objects in the recovery set
2. Recovery information for datafile, tablespace and database verification

If ORA-29308 : "view TS_PITR_CHECK failure" is encountered during the export run the check query again and make sure no rows are returned. If not, resolve any dependencies.

The FILE=<name> parameter can still be specified as in a normal export.

Export for TSPITR

Export dump file contains the following information:

- **Database version**
- **Export method (TSPITR)**
- **Block size**
- **Series of calls to DBMS_PITR:**
  - **Set compatibility restrictions**
  - **Tablespace number and SCN**
  - **File verification information**

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Export for TSPITR

Example

```
EXPORT:V08.01.05
DSYS  RTPITR  2048  0  28  4000   Wed Dec 15 15:23:50 1999
EXECUTE
sys.dbms_pitr.beginImport('8.0.0.0.0',2552309165,137639,381589235,804074);
EXECUTE sys.dbms_pitr.adjustCompatibility('NWIMGFMT','8.1.0.0.0');
EXECUTE sys.dbms_pitr.adjustCompatibility('JAVA','8.1.0.0.0');
EXECUTE sys.dbms_pitr.adjustCompatibility('OPERATOR','8.1.0.0.0');
EXECUTE sys.dbms_pitr.adjustCompatibility('INDXTYP','8.1.0.0.0');
EXECUTE sys.dbms_pitr.adjustCompatibility('ASSOC','8.1.0.0.0');
EXECUTE sys.dbms_pitr.beginTablespace(6,783139);
EXECUTE sys.dbms_pitr.doFileVerify(7,6,804074,804018,384276085,1048576);
EXECUTE sys.dbms_pitr.endTablespace(804074,804018,384276085);
EXECUTE sys.dbms_pitr.commitPitr;
CONNECT PETER
TABLE "EMP"
CREATE TABLE "EMP" (
  "EMPNO" NUMBER(4, 0),
  "ENAME" VARCHAR2(10),
  "JOB" VARCHAR2(9), ...
```

Export for TSPITR (Continued)

```
PROCEDURE beginImport(packageVersion, databaseID, resetSCN, resetStamp,
highestSCN)
```

This procedure checks package version, database ID, resetlog SCN, and stamp. It also adjusts primary database SCN if necessary, and enables pseudo create syntax.

Input parameters:

- PackageVersion: The version number of the package that emitted the PL/SQL anonymous block
- databaseID: 32 bits database ID
- resetSCN: Reset SCN expected in primary database
- resetStamp: Reset timestamp expected in primary database
- highestSCN: Highest clean SCN of the tablespaces in the clone database. The production database needs to adjust the SCN if the highest clean SCN is larger than the SCN in production database

```
PROCEDURE adjustCompatibility( comID, comRL)
```

This is called once for each entry in the clone database compatibility segment other than undo, bootstrap, and the compatibility segment itself.

Input parameters:

- comID: Compatibility type id.
- comRL: Current release level used by type comID.

```
PROCEDURE beginTablespace( tsid, createSCN);
```

Input parameters:

- tsid: The tablespace number undergoing point-in-time recovery.
- createSCN: Tablespace creation SCN.

```
PROCEDURE doFileVerify( fno, tsid, ckptSCN,  resetSCN, resetStamp,
filesize);
```

This procedure must follow a beginTablespace call. The file must be for the tablespace. There must be one call for each datafile that is part of the tablespace (in the clone database).

Input parameters:

- fno: Absolute datafile number
- tsid: Corresponding tablespace number.
- ckptSCN: The datafile checkpoint SCN.
- resetSCN: Reset SCN in file header.
- resetStamp: Reset timesatmp in file header.
- filesize: The size of the file in blocks.

```
PROCEDURE endTablespace( cleanSCN, resetSCN, resetStamp);
```

This procedure must follow the last doFileVerify call. Each beginTablespace call must have a matching endTablespace call. It builds a list of the files added between the current and the recovery point-in-time.

Input parameters:

- cleanSCN: tablespace clean SCN from the clone database.
- resetSCN: tablespace resetlog SCN from the clone database.
- resetStamp: tablespace resetlog time stamp from the clone database.

See DBMSPITR.SQL for more information.

Move Datafiles to the Primary

- Shutdown the CLONE.
- Alter all tablespaces of the recovery set are offline in the primary.
- Copy all the data files of the recovery set from the clone to the primary.

  Make sure that the tablespaces have been altered offline before attempting this.

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

Move Datafiles to the Primary

Shutdown the CLONE

This step is *ABSOLUTELY VITAL*. If TSPITR has been carried out with the recovery set remaining in the production database.

Copy Data files

All tablespaces in the recovery set must be offline before the files from the clone database are copied across. Check this before copying the files as follows:

```
SELECT    TABLESPACE_NAME, STATUS
FROM      DBA_TABLESPACES
```

If it is not possible to set the tablespace offline, for reasons such as missing a datafile , use the command `ALTER TABLESPACE <name> OFFLINE FOR RECOVER`. Make sure all tablespaces including those which are read only are set offline.

Import Into the Primary

- IMP has an extra parameter for TSPITR
  - POINT_IN_TIME_RECOVER=Y
  - `IMP SYS/<pwd> POINT_IN_TIME_RECOVER=Y`
- If errors occur, resolve the issue and reattempt import.

8-27

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Import Into the Primary

Import: POINT_IN_TIME_RECOVER=Y

The import performs the following steps (not necessarily in this order):

1. Drop all objects in the recovery set, thus removing all entries from UET$.
2. Retain tablespace and data file definitions with one FET$ entry per file.
3. Verify that the clone database was a version of the primary.
4. Verify the data file and tablespace information.
5. Make the data files part of the primary.
6. Bring back all the objects in the recovery set.

The last step is done using a special form of DDL syntax which creates the dictionary entries for the objects in the recovery set. The export contains the information needed to rebuild the entry in SEG$ and the segment header is used to rebuild all the entries of UET$.

Any failures during the import can be handled by sorting out the problem and running the import again.

TSPITR: Finishing Touches

- Update the primary:
  - Bring all recovery set tablespaces online and read-write (if altered read-only)
  - Rebuild indexes/constraints
  - Recalculate table/index statistics
- Back up the recovery set tablespaces on the PRIMARY database
  - Recovery through TSPITR is not allowed
  - ORA-1247 “database recovery through TSPITR of tablespace”
- Discard the CLONE

8-28 Confidential

Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

TSPITR: Finishing Touches

Backup the Recovery Set on the Primary Database

Attempting to perform media recovery through TSPITR will fail with ORA-1246. Attempting to perform database recovery through TSPITR will result in ORA-1247.

To avoid any possible recovery problems a backup of all the recovery set tablespaces should be taken as soon as possible after using TSPITR.

Discard the Clone Database

Check that the correct files are being removed.

Remember that if the recovery set was left in situ (that is, as part of the primary database) then the files *should not be removed* when the clone is discarded.

TSPITR: Limitations

- Cannot recover the SYSTEM tablespace
- Cannot undo `DROP TABLESPACE`
- Cannot recover a tablespace that has been dropped and recreated with the same name
- Cannot remove data files added to a tablespace
- Cannot recover optimizer statistics
- Cannot recover through a TSPITR operation

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

TSPITR: Limitations

Physical Tablespace Structure

TSPITR uses the current physical structure of a tablespace. If a file has been added to a tablespace after the time to which TSPITR is being done, the file will still appear as part of the tablespace and it will be empty.

.

TSPITR of Partition Tablespaces

- TSPITR cannot recover individual partitions.
- Use the following steps:
  - Convert the partition to a standalone table on both the PRIMARY and CLONE databases.
  - Use TSPITR as detailed to recover the table.
  - Swap the standalone table back into the partition on the PRIMARY database.
  - Correct any partition indexes.
- Or, make the entire partition table part of the Recovery Set.

8-30

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

TSPITR of Partition Tablespaces

TSPITR OF Individual Partitions

Remember that TSPITR works at the tablespace level and partitioned tables usually reside in several tablespaces.

This approach works only if the individual partition being recovered is in a separate tablespace from the remainder of the partitioned table.

The approach consists of converting the individual partition into a normal table. This allows TSPITR to proceed as normal; there is no need to consider the implications of partitions. Once the partition has been recovered, it can be exchanged back into a partition of the main table.

Partition Indexes

Once the TSPITR has been completed, the indexes on the partitioned table should be checked and any that are unusable should be corrected.

TSPITR with Transportable Tablespaces

- Uses the same methodology
- Transportable tablespace set is used instead of recovering files from the clone database
- Available in Oracle8*i*

  Compatible on target and source database must be 8.1.0.0.0 or greater

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

Summary

In this lesson, you should have learned about the following:

- How to implement object point-in-time-recovery (PITR) for Oracle7 and Oracle8.
- TSPITR basics, terminology, and limitations.
- Which TSPITR views to use.
- How to perform TSPITR.

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

References

- *Oracle8 Backup and Recovery Guide*

  For release 8.0.4 (part no A58396-01)
- Oracle Press, *Oracle8 Backup and Recovery Handbook*, Rama Velpuri and Anand Adkoli
- `$ORACLE_HOME/rdbms/doc/README.pit` (8.0.3 only)
- "*Tablespace Point-in-Time Recovery*", Alok Pareek, Oracle Technical Review, 1998, Volume 1.

8-33

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

References

See Also:

- "*TSPITR using RMAN*", Stefan Pommerenk, COE, Oracle Technical Review, 1998, Volume 2
- "*Tablespace Recovery Using an OS Backup*", Brenda Matrella, James Stone

9

Recovery Performance and Parallel Recovery

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Schedule:	Timing	Topic
	30 minutes	Lecture
	30 minutes	Total

Objectives

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

- Describe the advantages of parallel recovery
- Describe how parallel recovery works
- Implement a parallel recovery strategy

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

What Is Parallel Recovery?

- Mechanism to reduce recovery time
- Uses I/O parallelism to reduce recovery times
- Can be used for instance or media recovery
- Most effective when several data files spanning several disks are involved
- Parallel Recovery can be initiated several ways:
  - Starting several recovery sessions

    Each session recovers a separate set of files.

    Every session reads the entire set of redo logs.
  - By setting the RECOVERY_PARALLELISM parameter
  - Using the RECOVER PARELLEL command

9-3

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

What Is Parallel Recovery?

Parallel Recovery Using Multiple Sessions Examples

SESSION 1	SESSION 2	SESSION 3
recover datafile `sys01.dbf` `users01.dbf`	recover datafile `rbs01.dbf`	recover datafile

Parallel Recovery Usage

RECOVERY_PARALLELISM

- `init.ora` parameter.
- Specifies the number of processes to be used automatically during instance or crash recovery.
- Default is operating system dependent.
- Set greater than 1 if using the PARALLEL clause in RECOVERY.
- Should not be greater than the PARALLEL_MAX_SERVERS.
- Check Web IV Notes: 404843.9 2078384.6 1020568.102 and 2066675.102.

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

Parallel Recovery Usage

RECOVERY_PARALLELISM Initialization Parameter

This parameter specifies the number of processes to be used for instance or media recovery. A large value can greatly reduce instance recovery time. A value of 0 or 1 indicates that parallel recovery will not be performed and that recovery will be serial. This parameter works in conjunction with PARALLEL_MAX_SERVERS parameter. RECOVERY_PARALLELISM should not be set greater than PARALLEL_MAX_SERVERS parameter. Please check WebIV Note 404843.9 for further information

Parallel Recovery Usage

- Using the RECOVER command

  RECOVER DATABASE PARALLEL DEGREE d INSTANCES i
- DEGREE d
  - Specifies the number of processes used during the recovery for each instance
  - Should be less than the value set for PARALLEL_MAX_SERVERS parameter
  - Default value is calculated based on number of CPUs, devices and so on
- INSTANCES i
  - Specifies the number of instances in recovery
  - By default all available instances are used

9-5

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Parallel Recovery Usage (continued)

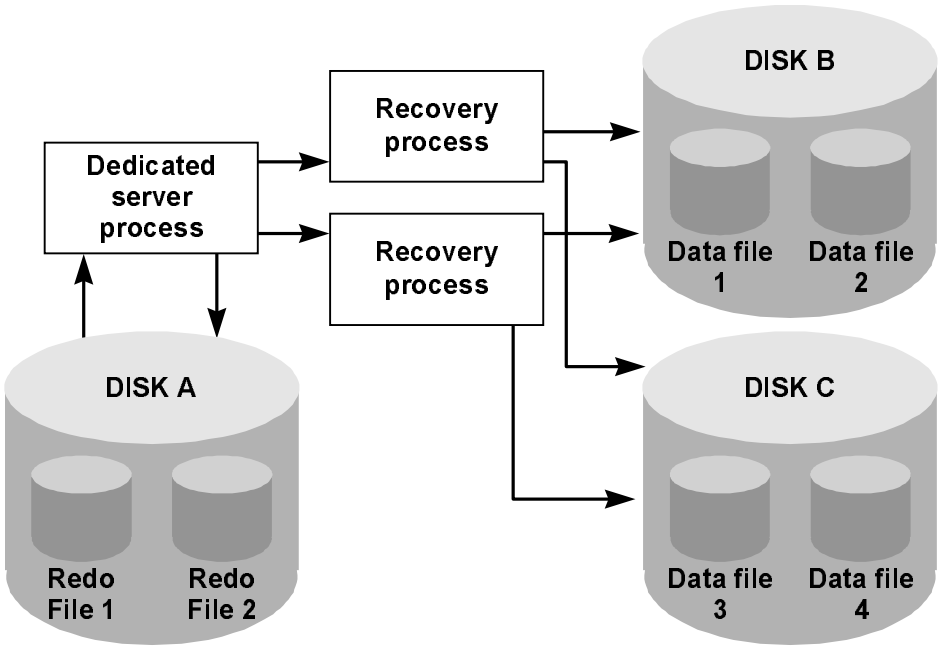
Example:

`RECOVER DATABASE PARALLEL DEGREE 6 INSTANCES 5;` will start up 30 slaves processes, 6 for each instance. So, PARALLEL_MAX_SERVERS must be set to 6 or higher for each instance.

ORA-00600: internal error code, arguments: [`ksmovrflow`], [`qdv`] can occur if you do not include DEGREE or INSTANCES in the command. (bug 739804)

If DEGREE parallel slaves could not be obtain on one instance, it is possible the parallel server will dump core (bug 545417 - fixed 8.0.4)

Typical Parallel Recovery Session



Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

Parallel Recovery Rollforward

- Rollforward Phase of the recovery process involves two stages:
  - Reading the redo
  - Applying the change vectors
- Reading the redo logs is done serially and thread. merging must be done in case of media recovery.
- We can parallelize the application of change vectors.
- The process that issues the RECOVER command reads the redo logs and sends the change vectors to the parallel slaves.
- Similar to parallel query architecture.

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

Parallel Recovery Rollforward

- Change vectors are assigned based on a hash function (DBA % # of recovery slave processes).
- Each slave process is responsible for loading the blocks into memory and applying the change vectors to the blocks that hash to its bucket.

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Roll Forward: Benefits

- Reduced elapsed recovery time for crash, instance and media recovery.
- Most effective when datafiles are on several disks under different controllers.
- If operating system supports asynchronous I/O then the performance improvements are not as large.
- If asynchronous I/O is not supported recovery elapsed times can be greatly reduced.

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Parallel Roll Forward Benefits

- Comparing parallel to serial recovery:
  - With synchronous I/O: 9 times faster
  - With asynchronous I/O: 2 times faster
- 1 or 2 parallel processes per disk with Asynch I/O.
- Recovery Elapsed time is reduced when the cost of CPU usage and I/O parallelism is less than process management cost on the system.
- Only one recovery session needs to be started, parallel processes are started automatically.
- The recovery process is disk intensive not CPU intensive.
- Redo log files are read only once by a single process.

9-10 Confidential

Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Parallel Roll Forward Benefits

Parallel Recovery Benefits

From *Oracle8i Backup and Recovery Guide*: "Parallel Recovery requires a minimum of 8 recovery processes to improve upon serial recovery."

FREEZE_DB_FOR_FAST_INSTANCE_ RECOVERY

- FREEZE_DB_FOR_FAST_INSTANCE_RECOVERY is an Oracle Parallel Server parameter.
- When set to TRUE, Oracle freezes the entire database during instance recovery.
- Stops all other disk activities except those for instance recovery. This lets instance recovery complete faster.
- The entire database becomes unavailable during instance recovery.
- If all online data files use hash locks, the default value of this parameter is FALSE. If any data files use fine-grain locks, the default is TRUE.

9-11 Confidential

Copyright © Oracle Corporation, 2000. All rights reserved.

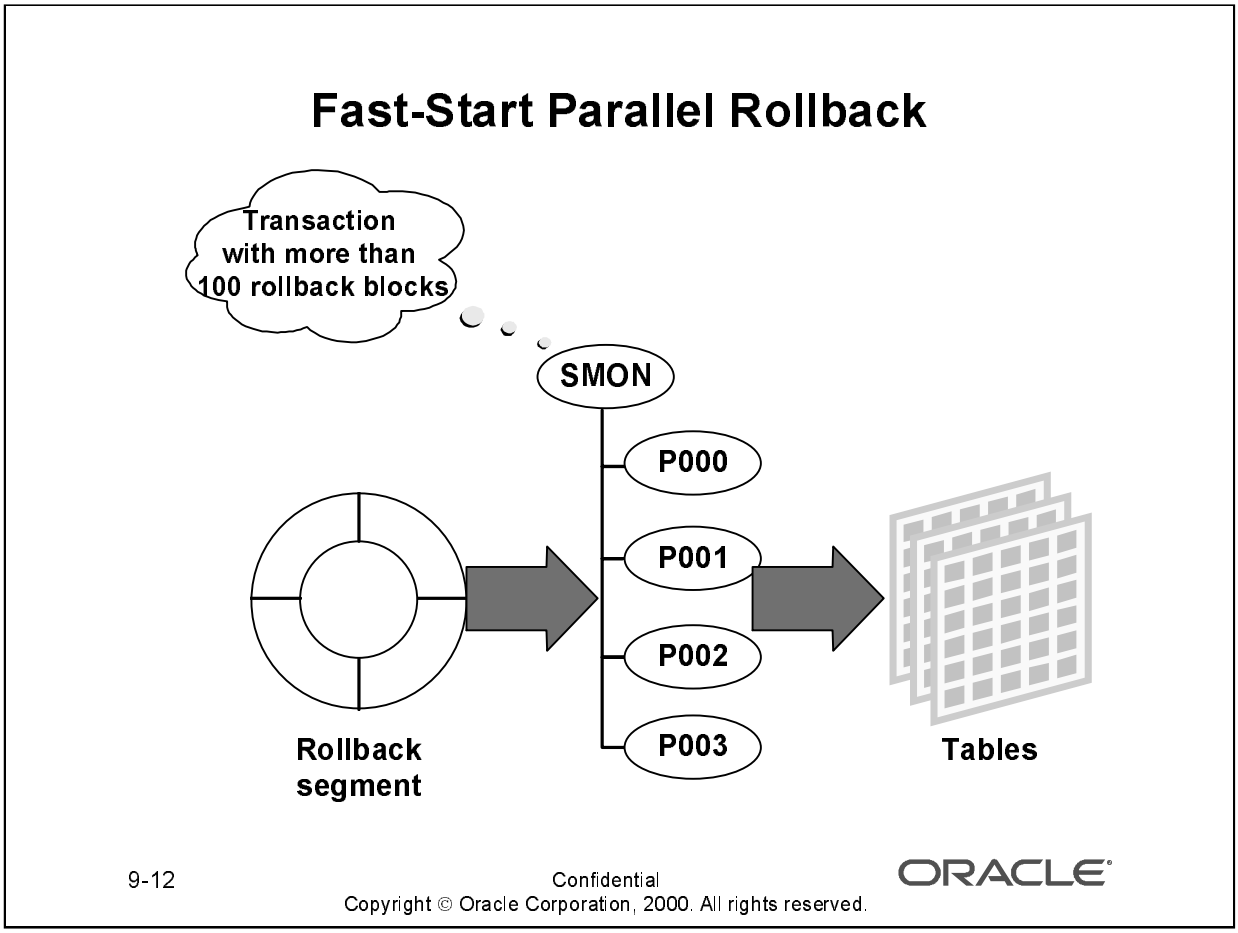
ORACLE

FREEZE_DB_FOR_FAST_INSTANCE_RECOVERY

The value of this parameter lets the database administrator control whether Oracle freezes the entire database during instance recovery. When this parameter is set to TRUE, Oracle freezes the entire database during instance recovery. The advantage of freezing the whole database is that it stops all other disk activities except those for instance recovery. This lets instance recovery complete faster. The drawback of freezing the whole database is that the entire database becomes unavailable during instance recovery.

When this parameter is set to FALSE, Oracle does not freeze the entire database, unless Oracle is responsible for resilvering some of the mirrored data files. Resilvering means ensuring data consistency of mirrored data files after a node crash. When Oracle does not freeze the entire database, part of the unaffected database is accessible during instance recovery.

If all online data files use hash locks, the default value of this parameter is FALSE. If any data files use fine-grain locks, the default is TRUE.



Fast-Start Parallel Rollback

Deferred Transaction Rollback

Deferred rollback was introduced in Oracle7.3.1. It enables a database to be opened as soon as the roll forward using the redo log files is completed. The rollback of any uncommitted transactions is done after opening the database to users.The rollback could be done in one of the following ways:

- By SMON, which periodically scans all the rollback segments and rolls back aborted transactions as needed.
- By user processes, whereby a user process encounters a row lock held by a dead transaction and recovers the transaction, which frees the row lock, after which the user process continues processing.

Prior to Oracle8*i*

Deferred rollback could take a long time when parallel transactions are terminated by a system crash, because rollback is predominantly a serial operation. For example, suppose that a parallel update transaction with a degree of parallelism of 6 runs for 10 minutes, but fails due to a system crash. Rolling back this transaction serially by SMON could take one hour or more. This feature was referred to as Parallel Transaction Recovery

With Oracle8*i*

In Oracle8*i*, the parallel transaction recovery feature has been renamed fast start parallel rollback. In addition to change in name, Fast start parallel rollback in Oracle8*i* enables SMON to use parallel query slaves to complete the rollback operation. Parallel rollback is automatically started when SMON determines that the dead transaction had generated a large number of rollback blocks. The current threshold for determining whether a transaction is large is 100 rollback blocks.

Fast-Start Parallel Rollback (continued)

Fast Start Rollback

Fast-start parallel rollback is mainly useful when a system has transactions that run a long time before committing, especially parallel INSERT, UPDATE, and DELETE operations.

When SMON discovers that the amount of recovery work is above a certain threshold, it automatically begins parallel rollback by dispersing the work among several parallel processes: process 1 rolls back one transaction, process 2 rolls back a second transaction, and so on.

The threshold is the point at which parallel recovery becomes cost effective;, in other words, when parallel recovery takes less time than serial recovery.

Intra-Transaction recovery

One special form of Fast-start parallel rollback is intra-transaction recovery, in which a single transaction is divided among several processes. Intra-transaction parallelism kicks in only when more than four slaves are available.

For example, assume that eight transactions require recovery with one parallel process assigned to each transaction. The transactions are all similar in size except for transaction 5, which is quite large.

This means that it takes longer for one process to roll this transaction back than for the other processes to roll back their transactions.

In this situation, Oracle automatically begins intra-transaction recovery by dispersing transaction 5 among the processes: process 1 takes one part, process 2 takes another part, and so on.

Parallel Rollback in an OPS Configuration

In OPS, you can perform Fast-start parallel rollback on each instance. Within each instance, you can perform parallel rollback on transactions that are:

- Online on a given instance
- Offline and not being recovered on instances other than the given instance

Once a rollback segment is online for a given instance, only that instance can perform parallel rollback on transactions on that segment.

Controlling Fast-Start Parallel Rollback

- Define dynamic parameter
- FAST_START_PARALLEL_ROLLBACK

Value	Maximum Parallel Recovery Servers
FALSE	None
LOW	2 * CPU_COUNT
HIGH	4 * CPU_COUNT

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Controlling Fast-Start Parallel Rollback

Parallel rollback is initiated by the FAST_START_PARALLEL_ROLLBACK initialization parameter. The valid values for this parameter and its impact on fast start parallel rollback are shown in the table.

Monitoring Parallel Rollback

- V$FAST_START_SERVERS
  - STATE: recovering or idle
  - PID, UNDOBLKSDONE
- V$FAST_START _TRANSACTIONS
  - USN, SLT, SEQ: Transaction ID
  - UNDOBLKSDONE
  - UNDOBLKSTOTAL
  - CPUTIME: Time in seconds

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

Monitoring Parallel Rollback

Dynamic Performance Views

Two new dynamic performance views are available to monitor the progress of fast start parallel rollback and the processes used in performing the rollback.

Monitor Use of Parallel Query Slaves for Fast Start Parallel Rollback

```
SELECT * FROM v$fast_start_servers;
STATE         UNDOBLOCKSDONE PID
----------    -------------- ----
RECOVERING                99 10
IDLE0                        11
IDLE0                        12
IDLE0                        13
```

Verify the Status of Fast Start Rollback

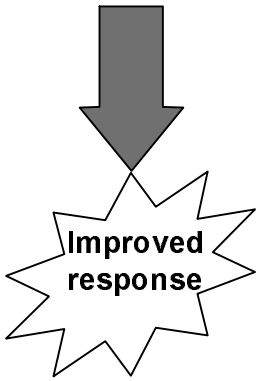
```
SELECT usn, state, undoblksdone, undoblkstotal
FROM v$fast_start_transactions;
 USN  STATE    UNDOBLOCKSDONE UNDOBLOCKSTOTAL
 ---  -----    -------------- ----------------
   2  RECOVERING          82               365
```

The USN column specifies which rollback segment from which the rollback is taking place; the UNDOBLKSDONE and UNDOBLKSTOTAL indicate the amount of work done and the total amount of work, respectively.

On-Demand Parallel Rollback

The server process encountering data to be rolled back performs the following:

- **Rolls back the block containing the required row.**
- **Hands off further recovery, which may be in parallel to SMON**



9-16

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

On-Demand Parallel Rollback

Fast-start parallel rollback speeds up the rolling back of transactions by SMON. This does not fix the problem of a user transaction having to wait after initiating recovery of large transactions. This problem could occur when user processes detect locks held by a dead transaction that SMON has not yet rolled back. On-demand parallel rollback solves this problem.

In on-demand block level recovery, user transactions initiate rollback only on the block the user transaction is attempting to access. The rest of the blocks are recovered in the background by SMON, potentially in parallel

Summary

In this lesson, you should have learned about the following:

- Parallel Recovery can be performed using various methods.
- Parallel recovery is set up with an init.ora parameter and the RECOVER DATABASE command.
- Only redo application is parallelized.
- Under the right conditions parallelism can reduce recovery times.

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

References

- Web IV Note 41391.1 "*Internals of Recovery: Oracle7 v7.2 Recovery Outline*" - Andrea Borr and Bill Bridge
- Web IV Notes 2078384.6 and 1020568.102 for 7.3.4
- Oracle8*i* Documentation

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

10

Archiving Internals

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Schedule:	Timing	Topic
	60 minutes	Lecture
	60 minutes	Total

Objectives

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

- **Describe how archiving is implemented**
- **Configure multiple archive destinations in Oracle8*i***
- **Use the new views to monitor archive process**
- **Describe redo log block checksums**
- **Debug archiving problems**

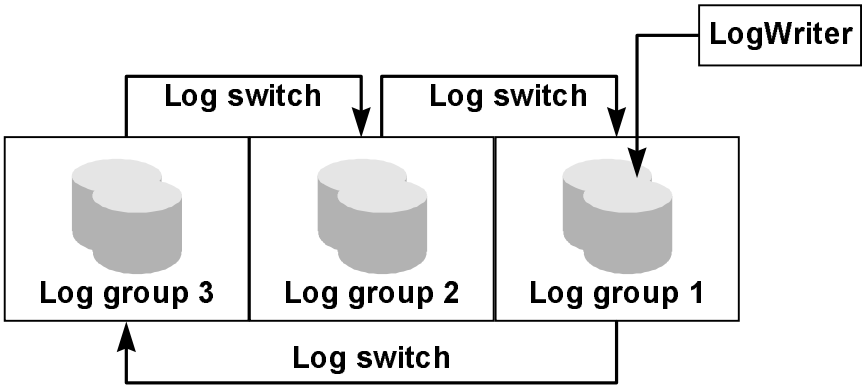
Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Objectives

In Oracle8*i*, multiple archive processes can be configured for an instance. Unlike in Oracle8.0 and before, these processes are now named ARC0, ARC1, and so on. In general, the Archive process is referred to as ARC0.

Redo Log Concepts



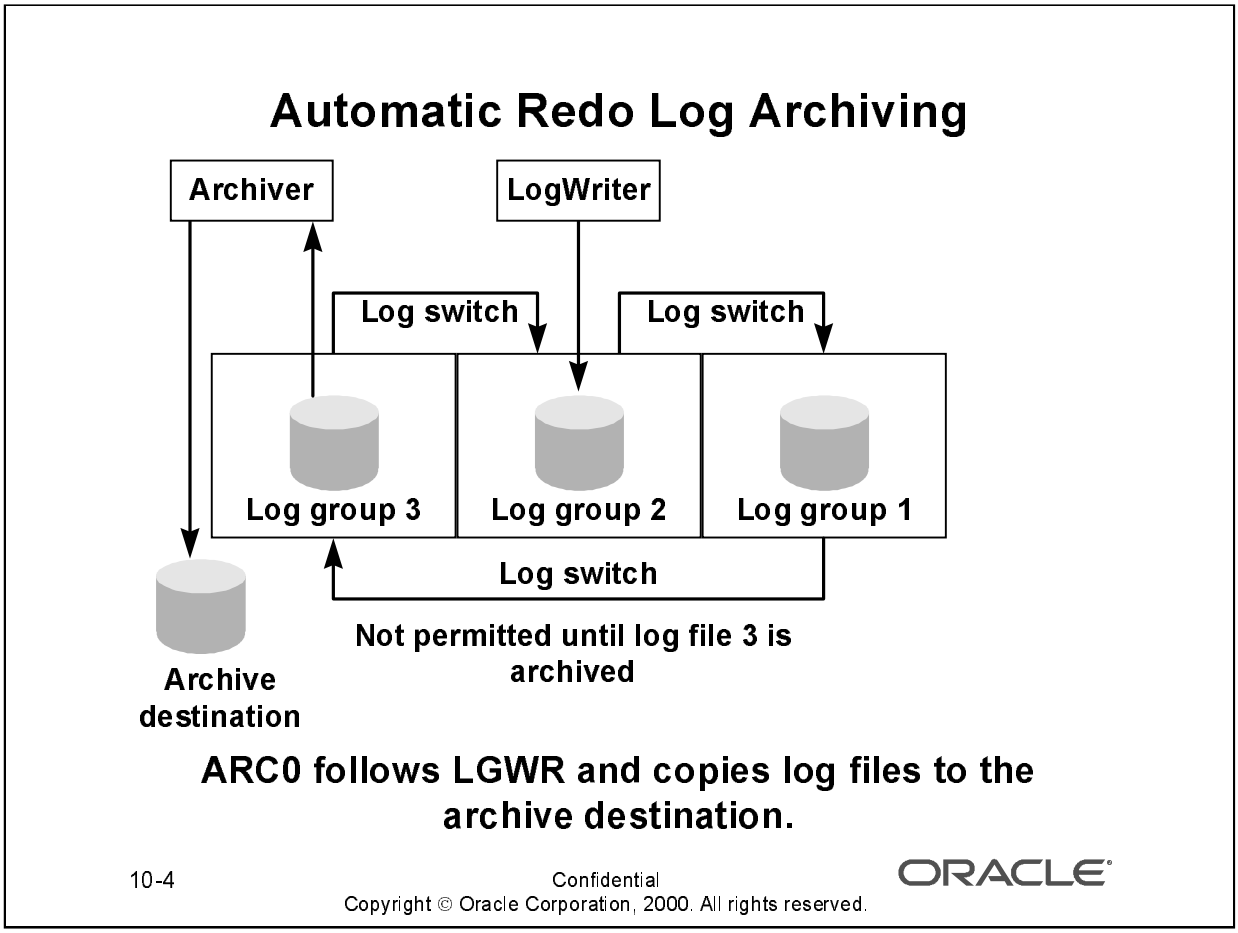
LGWR writes sequentially to each log file and then rewrites over the file it started with.

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

Redo Log Concepts

LGWR writes changes to the log files on disk as a mechanism to protect data changes performed by committed transactions:

- All data changes written to online redo log files by LGWR
- Checkpoints protect committed transactions by ensuring LGWR writes ahead of DBW0
- Minimum of two redo logs
- Redo logs are used in a circular fashion
- LGWR overwrites the initial log file after all logs written to



Automatic Redo Log Archiving

If automatic archiving is enabled:

- All online redo logs are copied before being overwritten by LGWR
- All data changes to the database are available in the archived logs
- ARC0 process copies the online redo logs to archive destination

If automatic archiving is not enabled, then the DBA intervention is required to archive the online logs.

Benefits of Redo Log Archiving

- Online backups: Reduces downtime
- Standby databases
- Point-in-time recovery (PITR)
- Oracle8 tablespace PITR
- More database recovery options
- Minimal database performance impact if configured correctly

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Operation of the Archiver

- ARC0 keeps up with LGWR in its own thread of redo.
- ARC0 has secondary roles in OPS configurations:
  - Posting quiescent instances to get their ARCn processes to archive log files for that instance
  - Performing log switches and archiving logs for idle instances

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Operation of the Archiver

Archive process will perform only one OPS operation before returning to the control file to see if any redo from its own instance needs archiving.

Operation of the Archiver

When ARC0 is posted to do some work, it:

- **Reads the control file to determine which, if any, logs need archiving**
- **Opens the online log member files for archiving**
- **Performs reads from the online log files**
- **Writes to archive log buffers**
- **Creates new archive log file**
- **Asynchronously writes to the archive file from the buffers**
- **Updates the control file**

10-7

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Operation of the Archiver (continued)

Oracle stops writing to the log buffers when they are full or when end-of-file is reached.

New archive files are created as needed. Only one archive file is being written at any time.

The archive link list and redo log status fields in the control file are updated as ARC0 moves from one archive file to the next

Enabling Automatic Archiving

- Configure init.ora parameters:
  - log_archive_start = true
  - log_archive_dest = (directory)
  - log_archive_format = (file format)
- STARTUP MOUNT
- ALTER DATABASE ARCHIVELOG
- ALTER DATABASE OPEN
- ARC0 process starts up and archives redo logs written to by LGWR

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

Multiple Archive Destinations

- New parameters in Oracle8*i*
  - LOG_ARCHIVE_DEST_n defines up to five destinations (n=1..5)
  - LOG_ARCHIVE_DEST_STATE_n
  - LOG_ARCHIVE_MAX_PROCESSES
- Can be changed dynamically
- Obsolete Parameters:
  - LOG_ARCHIVE_BUFFERS
  - LOG_ARCHIVE_BUFFER_SIZE
  - ARCHIVE_IO_SLAVES

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Multiple Archive Destinations

Oracle8*i* allows to archive online logs to multiple destinations. Up to 5 destinations can be specified. Destinations can be local and remote. In order to support this feature, Oracle8*i* introduces the following parameters:

- **LOG_ARCHIVE_DEST_n** (where n is an integer from 1 to 5)

  Local destinations are defined using the LOCATION keyword followed by the archive directory path. Remote destinations are defined by the SERVICE keyword followed by the TNS alias of the standby database.

  Examples:

  LOG_ARCHIVE_DEST_1="LOCATION=/oracle/arch_1 MANDATORY"

  LOG_ARCHIVE_DEST_2="LOCATION=/oracle/arch_2 OPTIONAL"

  LOG_ARCHIVE_DEST_3="SERVICE=uksn117_V813b REOPEN=60"

- **LOG_ARCHIVE_DEST_STATE_n** (where n is an integer from 1 to 5)

  Defines the status of each of the archive destinations. These parameters can take one of 2 values: ENABLE or DEFER. If a destination is enabled, Oracle8*i* archives to that destination. If deferred, the destination is not used for archiving.

- **LOG_ARCHIVE_MIN_SUCCEED_DEST** = n (where n is an integer from 1 to 5)

  –This is an 8.0 parameter, but is affected by the new Oracle8*i* parameters. It determines the minimum number of destinations to which Oracle8*i* must successfully archive a redo log group before it can reuse the online log files. The minimum value for LOG_ARCHIVE_MIN_SUCCEED_DEST is 1.

Multiple Archive Destinations (continued)

Multiple Archive Processes

LOG_ARCHIVE_MAX_PROCESSES = n (where n is an integer from 1 to 10)

Specifies the number of archive processes to be invoked at instance startup when the parameter LOG_ARCHIVE_START is set to TRUE. If LOG_ARCHIVE_MAX_PROCESSES is not set explicitly one archiver (ARC0) will be started. If LOG_ARCHIVE_MAX_PROCESSES is explicitly set to a value (range 1–10) then it will start additional archiver processes named arc0, arc1, arc2. . . . . and so on. There is no need to change this parameter from its default value of 1, because the system determines how many ARCn processes are needed and LGWR automatically starts up more ARCn processes when the database workload requires more.

Any successful archival in Oracle8*i* generates a trace file with the archiver process name.

If LOG_ARCHIVE_PROCESS is set to TRUE, Oracle starts up a single archiver process named ARC0. Subsequently if the parameter is changed using the ALTER SYSTEM command it will start the specified number of archive processes. For example:

```
SQLPLUS>alter system set LOG_ARCHIVE_MAX_PROCESSES=4;
```

will invoke additional processes ARC , ARC2, and ARC3 in the following sequence:

1. The shadow process contacts the primary archive process(ARC0 here) and sends a message to increase the number of processes.
2. The archiver process invokes the kcrrschd function. (kcrrschd: Schedule Multiple archive Processes).
3. It checks whether the requested(new) number of processes is lower than the current(old) number of archive processes. If it is lower or if ARCHIVING is DISABLED then it simply returns. Else if it is more than the supported limit (10) it limits the requested number of processes to 10.
4. The scheduler function acquires a latch on the kcrrxs{} structure(ARC0 Activation status) structure.
5. The function then loops over the specified number of processes and updates the structure to change the state to scheduled (KCRRSCHED)
6. It then releases the latch and hands over to the kcrrsmp(kcrrsmp: Start Multiple archive Processes) function for invocation.
7. The kcrrsmp function acquires a latch prematurely on the kcrrxs{} structure (Archive Activation status) to serialize access to the code section.
8. It schedules any pending archiver processes and cleans up dead processes, if any.
9. The function then loops over the specified number of processes and changes the state of each from KCRRSCHED to KCRRSTART (that is, the Archiver process is pending activation).
10. The latch is then released and ARCn processes are started .
11. The kcrrsmp function then reacquires the latch. Individual archiver processes are then notified to activate themselves. The archiver processes activate themselves and update the alert file.
12. The calling function sleeps while all the archiver processes to activate themselves and finally when all are active it updates the state in the kcrrxs structure.
13. Finally when the current number of archiver processes match the requested number of archiver processes, the latch and breaks are released.

Multiple Archive Destinations (continued)

The alert. log reflects the sequence of events as follows:

```
sql: prodding the archiver
ALTER SYSTEM SET log_archive_max_processes=4;
Tue Jul 13 02:15:14 1999
ARC0: changing ARC1 KCRRNOARCH->KCRRSCHED
ARC0: changing ARC2 KCRRNOARCH->KCRRSCHED
ARC0: changing ARC3 KCRRNOARCH->KCRRSCHED
ARC0: STARTING ARCH PROCESSES
ARC0: changing ARC1 KCRRSCHED->KCRRSTART
ARC0: changing ARC2 KCRRSCHED->KCRRSTART
ARC0: changing ARC3 KCRRSCHED->KCRRSTART
ARC0: invoking ARC1
Tue Jul 13 02:15:15 1999
ARC1: changing ARC1 KCRRSTART->KCRRACTIVE
Tue Jul 13 02:15:15 1999
ARC0: Initializing ARC1
```

The same activation sequence follows for ARC2 and ARC3.

Reduction in Number of Processes

The number of processes can also be reduced. When the command `alter system set LOG_ARCHIVE_MAX_PROCESSES=2;` is issued, the sequence of states is recorded in the alert. log file as follows:

```
sql: prodding the archiver
Tue Jul 13 00:34:20 1999
ARC3: changing ARC0 KCRRACTIVE->KCRRSHUTDN
ARC3: sending ARC0 shutdown message
ARC3: changing ARC1 KCRRACTIVE->KCRRSHUTDN
ARC3: sending ARC1 shutdown message
ARC3: received prod
Tue Jul 13 00:34:20 1999
ALTER SYSTEM SET log_archive_max_processes=2;
Tue Jul 13 00:34:20 1999
ARCH shutting down
ARC0: Archival stopped
ARC0: changing ARC0 KCRRSHUTDN->KCRRDEAD
Tue Jul 13 00:34:20 1999
ARCH shutting down
ARC1: Archival stopped
ARC1: changing ARC1 KCRRSHUTDN->KCRRDEAD
```

Interaction of Semantics

log_archive_dest_state_n	log_archive_dest_n	status
undefined	defined, not null	valid
enable	defined, not null	valid
defer	defined, not null	deferred
-	not defined, null	inactive

- The parameter log_archive_dest_state_n sets the initial destination status when instance starts.
- Valid destinations will be used in archiving.
- Deferred or inactive destinations will not be used.

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

Interaction of Semantics

If the complete directory path specified in the destination of the parameter LOG_ARCHIVE_DEST_n is not available, then the archived files can be written to the super directory.

For example if the following parameters are set:

- LOG_ARCHIVE_START = TRUE
- LOG_ARCHIVE_FORMAT = arc%S.%T
- LOG_ARCHIVE_DEST_1="LOCAL=/u02/oradata/test"

and if the directory path /u02/oradata/ does not have a subdirectory test, when a log switch occurs and the archive log, say number 1234, is written, it will be written to the /u02/oradata directory as file testarc1234.001. This can be an irritant when recovery is performed.

While with LOG_ARCHIVE_MIN_SUCCEED_DEST parameter you can broadly specify the number of successful archive destinations, with LOG_ARCHIVE_DEST_STATE_n parameter DBA can specify which of the destinations DBA would like to be definitely successful (MANDATORY) and which may be optional (OPTIONAL). This gives a better control of the archive destinations.

Interaction of Semantics (continued)

LOG_ARCHIVE_DEST_n and LOG_ARCHIVE_DEST_STATE_n parameters can be changed dynamically. This enables the DBA to defer archiving to a particular destination if it is filling up. The DBA can then make additional space available by purging archive logs and then restart the archival to the destination. However, any archives generated during the time that a particular destination is deferred, are not copied to those destinations. If such a destination is necessary during recovery, the DBA should copy the unavailable archive logs from where they are available.

Init.Ora Examples

Example 1:

- log_archive_dest_1="location=/u01/data"
- log_archive_dest_2="service=farout reopen=300"
- log_archive_dest_3="location=/user reopen=300"

Example 2:

- log_archive_dest_1="location=/u03/data mandatory"
- log_archive_dest_2="location=/user mandatory"
- log_archive_dest_3="location=/u01/data"
- log_archive_dest_5="service=far mandatory"
- log_archive_dest_state_2=defer
- log_archive_min_succeed_dest=2

10-14

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Init.Ora Examples

Example 1

- A typical configuration with local destinations duplexed and one remote host destination
- One of the two Local destinations must succeed
- Because the paramer log_archive_dest_state_n is not specified, log_archive_min_succeed_dest defaults to 1
- We cannot say which one would always be successful

Example 2

- Both dest_1 and dest_3 must succeed since dest_2 is deferred since log_archive_min_succeed_dest is set to 2.
- Dest_5 must also succeed
- No reopen on any destination
- To activate dest_2 as a destination:

```
alter system set log_archive_dest_state_2=enable
```

Manual Redo Log Archiving

- Archivelog mode.
- log_archive_start = false.
- No automatic archiving is performed.
- Enter archiving commands manually.
- Oracle will stop when all logs are filled and online logs not archived.
- Use the `ALTER SYSTEM` command with the following:
  - ARCHIVE LOG ALL
  - ARCHIVE LOG CURRENT
  - ARCHIVE LOG NEXT

10-15 Confidential

Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Manual Redo Log Archiving

If the database is in archivelog mode and log_archive_start = false in the init.ora file, then no ARC0 process starts up with the database and all archiving will have to be performed manually .

Although manual archiving is too onerous to be practical, the manual archiving commands can be used in conduction with the ARC0 process. Issuing manual archiving commands invokes the shadow process for the svrmgr session to archive some logs, allowing ARCH to catch up with LGWR.

If ARC0 does not archive the log files before they are needed by LGWR, the database can come to a complete stop, waiting for ARC0 to finish archiving the log.

`ARCHIVE LOG ALL` command archives all redo logs except the current log being used by LGWR and any logs already archived. When this command is issued, the shadow process for the session starts archiving log files until ALL logs have been copied. This command can be issued to help ARC0 catch up with LGWR.

`ARCHIVE LOG CURRENT` does the same as `ARCHIVE LOG ALL` but archives the current redo lo as well. This may not be desirable as the current log may be almost empty if a log switch has just occurred.

`ARCHIVE LOG NEXT` tries and archives the next log file the database is due to write to.

For parallel instances, you would use the thread clause as well:

`ALTER SYSTEM ARCHIVE LOG` *`THREAD 2`* `CURRENT`

Archiving Tips

- Archiving should happen faster than log switches occur.
- Use multiple log members in each group so that ARC0 can multiplex file reads.
- The archive destination should have sufficient disk bandwidth.
- Manage the archive destination area.

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Managing the Archive Destination

- The archive destination should be on separate disk from the online logs.
- Monitor available free space.
- Offload archived logs regularly to tape with backups or in batches as needed.
- Purge unnecessary archive logs.

10-17

Confidential

Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Managing the Archive Destination

If you copy your archive logs to offline media, you need to do so at a rate faster than they are being produced by the database.

Datafile performance tuning methods should be applied to the archive log destination site.

Any archived log files older than the oldest complete backup set should be deleted as they are unusable.

While purging take care to see that no copies / backups are over-written inadvertently. Any over-writing may leave a gap in the redo stream (archive logs sequence) and may become a cause of concern during media recovery.

Archiving and Parallel Server

- Multiple instances
- Multiple sets of online and archived logs
- Each instance has a separate redo thread:
  - Unique and independent
  - Open or closed, enabled, or disabled

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Archiving and Parallel Server

With Oracle Parallel Server configurations there are multiple instances all accessing the same set of Oracle database files. Each instance has its own set of online redo logs and can create its own archived log files.

A redo thread consists of the online portion of the redo plus any archived logs.

Each redo thread is independent of all other threads. They can be configured differently from each other—one performing manual archiving, the other automatic. Archive log mode is a database mode, all instances accessing the same database in an OPS environment inherit the archive mode. However, each instance may be configured to use its own location and scheme of archival.

Archiving and Parallel Server Force SCN

- SCN values for all instances kept within a set range of the Force SCN.
- Quiet instances may be kicked.
- Active instances can archive idle instances (shutdown instances).

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

Archiving and Parallel Server Force SCN

Log switches are generally independent of other threads and instances. However, Oracle tries to keep all of the instances within a certain range of SCN values. Quiet instances my be kicked to switch log files and increase their SCNs. Idle instances may have their logs switched and archived by other instances in order to force their SC's to keep up.

Oracle maintains a force SCN to keep all instances within a certain SCN range. This range is not tunable.

For more information on Archiving in an Oracle Parallel Server environment, refer to DSI 308.

ARC0 Inter-Instance Communication

- Post other ARC0 processes in quiet instances to switch logs to increase the checkpoint SCN for that instance.
- Force a log switch and archive logs in a closed instance if its SCN gets below the force SCN value.
- Primary responsibility is still to archive its own redo thread.
- May be posted to interrupt its secondary tasks to attend to its own thread.

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Arco Inter-Instance Communication

Archive process is responsible for keeping all SCNs for all enabled threads in the database within a certain range of the force SCN. If an instance is going to be down for an extended period of time (recovery, tuning, reconfiguration, hardware upgrades, and so on), then disable the thread for that instance to reduce the workload of other ARCn processes.

Control File Structures for Archive

- Used to communicate with the instance.
- Force SCN (OPS only).

  Enforces a minimum SCN for all instances.
- Archive link list contains pointers to the previous and next (fwrd) log files to be archived.

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Archive Data in the Control File Example

- Dump the contents of a control file using the command:

```
alter session 'set events immediate
  trace name controlf level 10'
```

- Find the redo and archive link lists in the log file section.

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

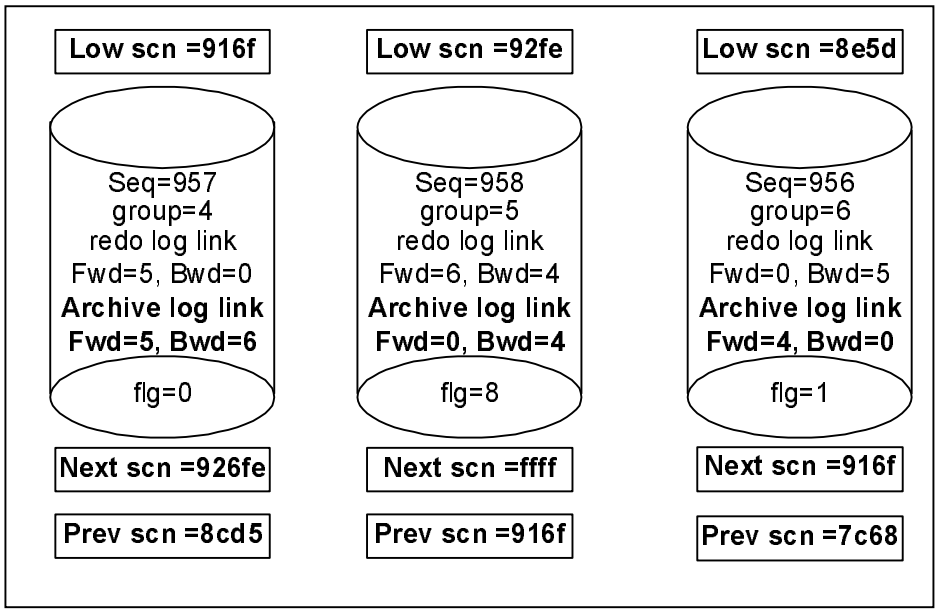
Controlfile Structures
Archive Link List

- One entry for each active redo log group.
- Pointers to the previous and next log group to be archived.
- Implemented as a doubly linked list.

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Control File Dump Visualization



Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

Control File Dump Visualization

Some points:

Which log is current?

Follow the redo log link:

4, 5, 6

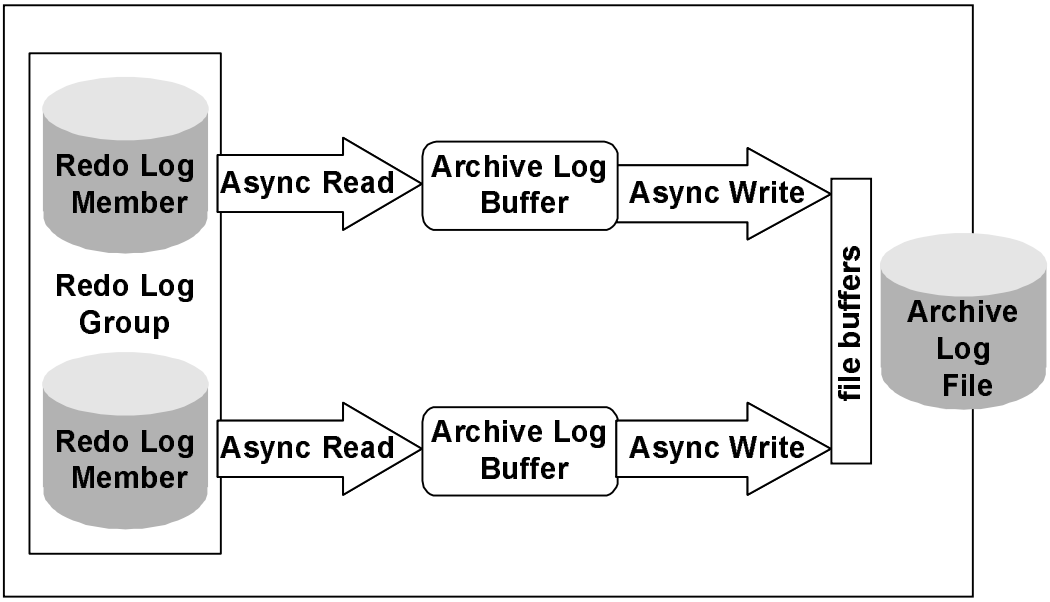
Which log is the next current log?

Group 6

Which group will be archived first?

Group 6

Using Multiple Buffer Streams



Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Sanity Checks

ARC0 performs some basic checks when copying the online log files:

- **Log file headers**
  - **Database id**
  - **Create date**
  - **Low and high SCN numbers**
- **If the LOG_BLOCK_CHECKSUM parameter is set to true:**
  - **Calculates checksums when writing to the redo log files**
  - **Can help in identifying the source of corrupt archive log files**
  - **Performance overhead**

10-26

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Sanity Checks

The LOG_BLOCK_CHECKSUM parameter can be used to provide an additional check against corruption in the online and archive logs.

Performance overhead of using LOG_BLOCK_CHECKSUM should be minimal, just 1% to 2%

When ARC0 Detects Corruption

- Attempts to read from another member of the same redo log group.
- If all group members are corrupt, ARC0 signals an error and gives up.
- If one member in the group is not corrupt, then no error is flagged.
- If ARC0 stops due to an error, the database stops when LGWR catches it.

10-27

Confidential

Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

When ARC0 Detects Corruption

If one member of a redo log group could not be written to, but at least one member could, then a fatal error is not signaled and the database continues processing. However, an alert is posted to the alert log that LGWR failed to write to an online log.

If the redo logs are not mirrored within Oracle and corruption is found, the error is signaled and the instance will crash.

Events That Post ARC0

- Time-outs every five minutes.
- Log switches.
- ARC0 can post itself while archiving if another log needs to be archived and LGWR is waiting.
- Instance recovery inhibits ARC0.

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Event That Post ARC0

The ARC0 process will remain idle unless it is posted to look and see if anything needs archiving.

Tuning and Configuring Archive Process

- Size the online redo log files correctly
- In Oracle 8.0 and 7.3 use LOG_ARCHIVE_BUFFERS and LOG_ARCHIVE_BUFFER_SIZE to tune archive process
  - Increase LOG_ARCHIVE_BUFFERS to improve archiver performance
  - Increase LOG_ARCHIVE_BUFFER_SIZE to speed up archiver
- LOG_ARCHIVE_BUFFERS and LOG_ARCHIVE_BUFFER_SIZE are obsolete in Oracle8i

10-29 Confidential

Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Tuning and Configuring Archive Process

Archive process should be tuned so that it does not detract from the performance of the database by working too fast, or working too slow so that LGWR is constantly waiting on the archiver to finish.

In Oracle8.0 and Oracle7.3, you can increase `LOG_ARCHIVE_BUFFERS` to improve performance of archive processing and use `LOG_ARCHIVE_BUFFER_SIZE` to tune how fast it works. If archiv process is taking too many CPU cycles, decrease `LOG_ARCHIVE_BUFFER_SIZE` and/or `LOG_ARCHIVE_BUFFER`. There should be at least as many log archive buffers as there are log file members.

Tuning and Configuring ARC0

- Use multiple process to archive as needed.
- Ensure that operating system asynchronous I/O is enabled.
- Use log_archive_duplex_dest and log_archive_min_succeed_dest to mirror archived logs for very critical systems.
- In Oracle8i, use log_archive_dest_state_n parameter to effectively control the archival.

10-30 Confidential

Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Tuning and Configuring ARC0

If ARC0 is having trouble keeping up with LGWR it may help to increase the size of the online log files. This will help ARCH keep up with LGWR, especially if redo generation is in bursts, as there will be more time between log switches. You can also write scripts that monitor LGWR and ARC0. These scripts can issue an ALTER SYSTEM ARCHIVE LOG ALL command when ARC0 falls behind. This additional process performing archiving may help ARC0 catch up.

Asynchronous I/O at the OS level can have a big impact on archiving performance.

Additional archiving processes can be used by using manual archiving commands in Oracle7, or by using I/O slaves in Oracle8 with the ARCH_IO_SLAVES parameter. The ARCH_IO__SLAVES parameter is obsolete in Oracle8i.

With the `log_archive_duplex_dest` parameter you can specify an alternate archive destination for archive logs. With `log_archive_ min_succeed_dest` you can determine whether copies of logs written to this destination are optional (set to 1) or critical (set to 2).

Diagnosing ARC0 Problems

- Get a full description of the problem.
- Use `archive log list` *command* to confirm that the database is in archivelog mode.
- Check the next and current logs.
- Check for trace files and entries in the alert log.
- Check for hardware or disk problems.
- Try to archive to a different location.

10-31

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Diagnosing ARC0 Problems

Check for any trace file dumps in the *background_dump_dest* directory and in the database alert log. Problems with the archive destination directory not being writable will usually show themselves here.

If space is the cause of ARC0 problems, consider automating the monitoring of space in the archive destination so the problem does not occur again.

If ARC0 is stuck, do not delete the archive files that will be needed to perform recovery, if one is necessary. Copy the files to tape or to another disk first.

You can send archive logs to a different location with the `ALTER SYSTEM ARC HIVE LOG NEXT TO '<directory>` command. This allows you to focus on the disk as the problem and start archiving (and thus database processing) to another location without shutting down the database.

Diagnosing ARC0 Problems

- V$ views
  - V$LOG
  - V$LOGFILE
  - V$DATABASE
  - V$ARCHIVE
  - V$ARCHIVE_DEST
- If still stuck and evidence points to ARC0, use oradebug to get a stack trace for ARC0.

10-32 Confidential

Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Diagnosing ARC0 Problems (continued)

V$LOG

Check the status of online logs, have they been archived, the start times, the SCN's for logs, and look for invalid or stale in the status field.

V$LOGFILE

Lists the database online redo logs. Check for invalid or stale or deleted in the status field.

V$DATABASE

Confirms in archivelog mode, ARCHIVE_CHANGE# gives the SCN of the last change to be archived.

V$ARCHIVE

Lists all the logs that need archiving from all threads. This is useful if there are many entries in this table because it indicates that the LGWR might be switching logs faster than ARC0 can archive them out.

Diagnosing ARC0 Problems

If there are problems with the archiver and there are no hardware or disk bandwidth issues on your system or any alerts or trace files, you will need to find out what ARC0 is doing internally:

- Use oradebug to dump a stack trace for the ARC0 process.
- Determine what ARC0 is doing.
- Check for platform/version specific bugs.

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

V$ARCHIVE_DEST COLUMNS

- Status
  - valid #defined and enabled state
  - inactive #not defined
  - deferred #defined but deferred state
  - error #an archive error has occurred

    Not considered for future archiving until cleared
  - disabled #manually disabled following error
- Binding: Optional or mandatory

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

V$ARCHIVE_DEST COLUMNS

- Target: Primary or standby
- Destination: Location directory or service name
- Fail_sequence: Redo sequence that has failed
  Not used for future archiving until cleared
- Error: Text of last error
- Reopen_secs: Reopen option

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

SUMMARY

In this lesson you have learned:

- Implementation of how archiving.
- Configuring Multiple Archive Destinations in Oracle8*i*.
- The use of new views to monitor archive process.
- Configuring redo log block checksums.
- How to debug archiving problems.

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

11

Oracle Standby Database

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Schedule:	Timing	Topic
	75 minutes	Lecture
	75 minutes	Total

Objectives

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

- Explain the use of a standby database
- Configure initialization parameters
- Create, maintain, and activate a standby database
- Describe Managed Recovery mode
- Set a standby database in Read Only mode
- Describe the process of propagating structural changes to the standby database
- Propagate “nologging” transactions on the primary database to the standby database
- Observe the dump of the control file and the data file headers from the standby database

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

Disaster Recovery Concepts

- Provide higher availability
- Provide product usability
- Resilient configuration from catastrophic failures of primary database:
  - Outages due to environment: Fire, earthquake
  - Physical outages: Loss of entire machine

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Disaster Recovery Concepts

Standby databases are usually used as a part of a solution but they do not resolve all problems. Standby databases do not resolve the problem of losing the online redo log; other than giving you quick recovery up to the corruption.

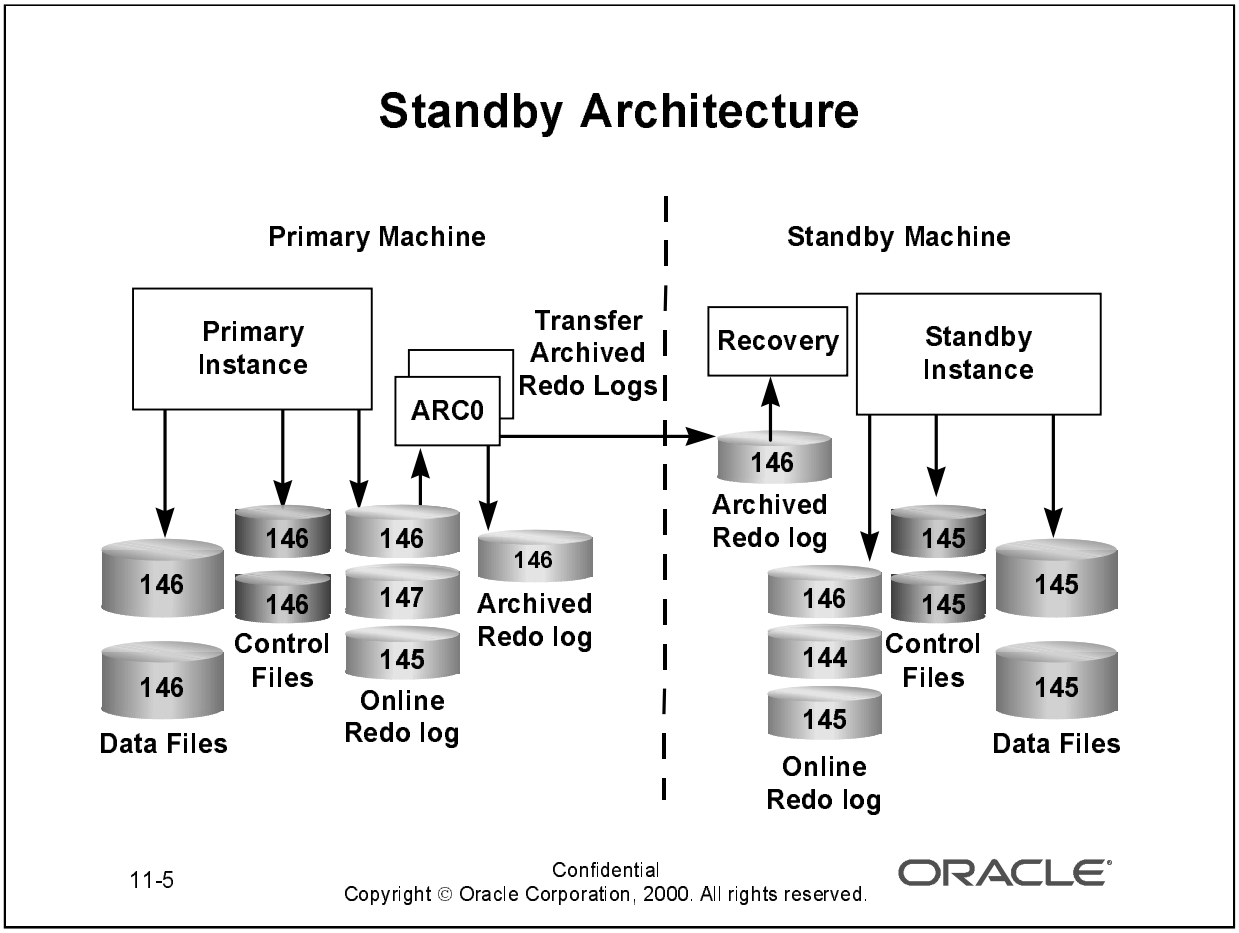
Please read lto's bulletin on checklist of outages.

Standby Database Features

A standby database:

- Is a copy of the primary database on a separate machine
- Is kept in recovery mode for immediate activation
- Can be in Managed Recovery mode
- Can be opened in READ ONLY mode for queries
- Becomes the primary database after activation

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.



Standby Architecture

With Oracle8*i*, the standby database has undergone significant changes architecturally.

First, the facility to configure automatic archival to a remote location has been incorporated. Therefore by appropriately configuring the initialization parameter and the Net8 configuration files, Archived redo logs can be written to a remote (standby) location automatically. In this, the ARCn process performs the local archiving, and communicates with remote file sharing (RFS) process through the listener on the remote (standby) host to write the archive files in the standby location. One RFS process is created for each standby location. For this to function, the standby database should be in MOUNTED (STANDBY) or OPEN (READ ONLY) mode.

Second is the ability to maintain the standby in Sustained recovery mode, where the recovery process on the standby automatically applies archivelogs when they have been written to the standby archive destination. The automatic recovery is initiated as the RFS process updates the control file on the standby database when each archive file is written.

Third is, with 8*i*, the recovery can be suspended and resumed without having to recreate the standby database.

Finally, the standby database can be opened in read only mode and can be used for query purposes. During the time that the database is opened, recovery is suspended.

Guidelines

- **Prerequisite:**

  The same version, release, and patch of the OS and RDBMS on primary and standby hosts.
- **To create a standby:**
  - **Configure the initialization parameters on standby and primary.**
  - **Configure the Net8 configuration files (tnsnames on primary and listener on standby).**
  - **Create a standby control file and copy it to standby.**
  - **Copy a backup of data files to standby.**
  - **Start the standby database and initiate recovery.**

11-6

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Guidelines

Standby Database Guidelines

The databases (both primary and standby) should be in Archivelog mode

The standby database operates on Oracle release 7.3 and later

Standby and primary databases should ideally be on different machines

Use the same version, release, and patch of the operating systems and RDBMS on the primary and standby hosts. The standby host may use a different disk configuration

Standby can be opened in READ ONLY mode and returned to standby mode (in Oracle8i)

When standby is in READ ONLY mode, recovery is suspended

Creation Guidelines

Configure the initialization parameters on the primary and standby databases

Configure the Net8 files `tnsnames.ora` on the primary and `listener.ora` on the standby database to automatically transfer archive logs to the standby database

Create control file using the command:

```
ALTER DATABASE CREATE STANDBY CONTROLFILE AS <filename>
```

Backup the datafiles and copy them over to standby location

Do not copy the online redo logs from primary to the standby.

Use either or both of the following to to correctly indicate the file locations in standby

- `ALTER DATABASE RENAME FILE … command`
- `DB_FILE_NAME_CONVERT/LOG_FILE_NAME_CONVERT` parameters in the `init.ora`.

Initialization Parameters

Same	Only on Standby
COMPATIBLE	DB_FILE_NAME_CONVERT
DB_NAME	LOG_FILE_NAME_CONVERT
DB_BLOCK_SIZE	STANDBY_ARCHIVE_DEST
LOG_ARCHIVE_FORMAT	LOCK_NAME_SPACE

Preferably different

CONTROL_FILES
SERVICE_NAMES
INSTANCE_NAME
LOG_ARCHIVE_DEST_n
BACKGROUND_DUMP_DEST
USER_DUMP_DEST

11-7

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Initialization Parameters

The initialization parameters for configuring the standby database can be grouped into three categories.

Same

Parameters such as COMPATIBLE, DB_NAME, DB_BLOCK_SIZE and LOG_ARCHIVE_FORMAT must be the same in both primary and standby database instances.

Only on Standby

Parameters such as DB_FILE_NAME_CONVERT, LOG_FILE_NAME_CONVERT and STANDBY ARCHIVE_DEST are to be configured only on the standby instance.

Parameters LOG_FILE_NAME_CONVERT and DB_FILE_NAME_CONVERT reduce the effort of renaming files on the standby site especially if similar (not necessarily the same) directory structure is used at standby and primary databases. Oracle server automatically parses the file name and replaces arguments appropriately. In cases where these parameters are not sufficient, the DBA can always manually rename the files in their standby site by using *alter database rename file "file name" to "new name"*.

Initialization Parameters (continued)

May Be Different

Parameters such as CONTROL_FILE, USER_DUMP_DEST, BACKGROUND_DUMP_DEST, LOG_ARCHIVE_DEST_n, SERVICE_NAMES, and INSTANCE_NAME, would be different for the standby and primary instances, especially if you configure standby database on the same machine (or cluster) as the primary database.

Many customers also optimize their init.ora parameters for faster recovery. They can increase db_block_buffers, db_file_simultaneous_writes, the number of database writers, and so on. Furthermore, recovery is done with higher degrees of parallelism if possible.

The drawback to this option is that you may have to switch to the original init.ora parameter prior to activating the standby database.

LOG_ARCHIVE_DEST_n

- LOG_ARCHIVE_DEST_n (n=1..5) to replace LOG_ARCHIVE_DEST and LOG_ARCHIVE_DUPLEX_DEST
- LOG_ARCHIVE_DEST_n =“destination option1 option2”
  - Destination - service=nnnn for remote
  - Option1: mandatory or optional
  - Option2: REOPEN=nnn
- Can be changed at session or system level
- One local disk destination is required
- LOG_ARCHIVE_DEST_STATE_n parameter to replace LOG_ARCHIVE_MIN_SUCCEED_DEST

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

LOG_ARCHIVE_DEST_n

Remote Archival

Previous implementations of a standby database required the DBA to copy the archived redo logs from the primary host to the standby host and manually apply them. This feature automates the transmission of redo log files for potential use in standby database recovery, automated or manual.

To configure remote archival, the following steps have to be taken.

1. Configure LOG_ARCHIVE_DEST_n and LOG_ARCHIVE_DEST_STATE_n parameters on the primary database.
   - Log_archive_dest_n replaces log_archive_dest and log_archive_duplex_dest although, for backward computability, the latter two parameters are supported.
   - Log_archive_min_succeed_dest has been reinterpreted to mean the number of local destinations that must succeed. It cannot be less than 1 or more than the available (defined and not deferred) local destinations.
   - Optional attributes
     - optional/mandatory #required for redo log reuse?
     - reopen=n #allow future archiving in n secs
     - Reopen responds to inability to create the file and allows a subsequent archive operation to occur after n secs. Resets error condition
     - Example : Log_archive_dest_3=“service=STDBY reopen=60”
   - The key word SERVICE indicates that the specified destination is remote.
   - In this example, the service name STDBY is the TNS alias for the standby database.
   - If an attempt to write archive log fails, next attempt is kept waiting for 60 seconds.

Net8 Config Files

```
tnsnames.ora on primary
stdb=
(DESCRIPTION =
 (ADDRESS_LIST =
  (ADDRESS = (PROTOCOL = TCP)(HOST = ..)(PORT
= 1521)))
  (CONNECT_DATA =      (SID = stdb)   )   )

listener.ora on Standby
LISTENER=
 (ADDRESS_LIST =
  (ADDRESS = (PROTOCOL = TCP)  (Host = …)
(Port = 1521) ) )
SID_LIST_LISTENER=
 (SID_LIST =
   (SID_DESC =
     (SID_NAME = stdb) (ORACLE_HOME = …) ) )
```

11-10

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Net8 Config Files

Remote Archival (continued)

Configure Net8 configuration files on Primary and Standby for automatic transfer of Archive files to standby.

2. Configure tnsnames.ora on the primary. Ensure that the alias for Standby instance is as mentioned in the LOG_ARCHIVE_DEST_n parameter.

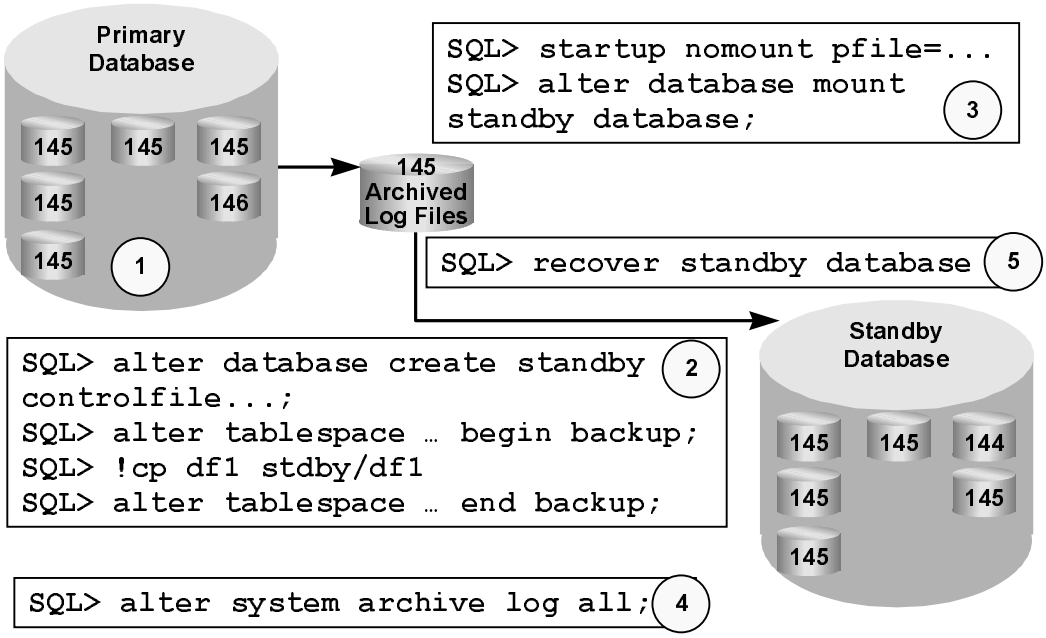
```
STDBY=
 (DESCRIPTION =
  (ADDRESS_LIST =  (ADDRESS=(PROTOCOL=TCP)(HOST=….)(PORT=...)) )
  (CONNECT_DATA=(SID=...))
 )
```

3. Configure listener.ora on the primary

Ensure that the parameters such as PORT, and SID match those in the tnsnames.ora of the primary.

```
LISTENER=
  (ADDRESS_LIST =
   (ADDRESS = (PROTOCOL = TCP) (Host = …) (Port = 1532) ) )
SID_LIST_LISTENER=
  (SID_LIST =
    (SID_DESC =
      (SID_NAME = stdb) (ORACLE_HOME = ….)  )  )
```

Creating a Standby Database



11-11

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Creating a Standby Database

- Create a standby control file on the primary and copy it to the appropriate location on the standby.
- Back up the data files and copy the backup to appropriate location for standby.
- Startup the standby instance.
- Mount the standby database.
- Initiate recovery.

Maintaining the Standby Database

- Regularly transfer archived logs from the primary to standby database.
- Apply the archived logs to the standby database through recovery.
- Ensure that the archive files do not fill up standby destination.
- If needed backup on to tape and purge directory.
- Checking all files are online that needs recovery.
- Taking unnecessary data files offline.

```
ALTER DATABASE DATAFILE 'filename' OFFLINE DROP
```

11-12

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Maintaining the Standby Database

The standby database should be synchronized frequently with the primary database. This is achieved through the following process:

- Regularly transfer archived logs from the primary database to the standby database. The frequency will depend on how fast the standby database is required after the primary database fails.
- After the archived logs have been transferred, use the recover command to apply them to the standby database.
- Avoid incomplete recovery and clearing unarchived log files on the primary database, because these will invalidate the standby database. If such activities are required, then the standby database may have to be re-created.
- When operations involving NOLOGGING option are performed on the primary database, you have to take a backup of the tablespace involved and copy it to the standby database.

Note: Do not transfer redo logs. Transferring redo logs from the primary database will cause an error.

Many customers only want a subset of their Primary or Production database in their Standby environment. One needs to offline drop these data files if you do not want to recover them. It is important that all files that you want to recover has the correct file path and are online.

Managed Recovery Mode

- Standby automatically applies the archived logs once the files become available.
- Manual or automated transfer of archived logs.
- No [from location] option; uses `standby_log_dest` in standby database.

```
RECOVER MANAGED STANDBY DATABASE TIMEOUT 60
```

- Recovery operation waits the specified (TIMEOUT) number of minutes for applying archive logs.
- To terminate recovery, use the command from a separate session.

```
RECOVER MANAGED STANDBY DATABASE CANCEL
```

11-13

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Managed Recovery Mode

Standby Database in Managed Recovery Mode

While manual recovery mode of the standby database is possible, it is preferable to use sustained recovery so that the standby closely synchronizes with the primary at all times. In Managed Recovery mode, a standby database waits for archived log files from a target database and then automatically applies the redo logs when the files become available. This feature eliminates the need for you to interactively provide the recovery process with filenames of the archived redo logs. The task of copying archive logs should be done by the DBA and can be automated separately.

How to Place the Standby Database in Managed Recovery Mode

Use the MANAGED option of RECOVER command to put the standby in managed recovery. Following is an example.

```
RECOVER MANAGED STANDBY DATABASE TIMEOUT 60;
```

When you specify TIMEOUT option, the recovery operation waits for specified number of minutes for the Oracle server to write the requested archived log entry to the standby control file's directory. If TIMEOUT option is not specified, recovery process will wait indefinitely for the next archive log. There is no [from 'location'] option to this recovery command. RFS updates the controlfile with the full pathname of the place where it put the archive log files. This is the value given in standby_archive_dest or the default location ORACLE_HOME/dbs.

The managed recovery must be stopped from a different session with the command

```
RECOVER MANAGED STANDBY DATABASE CANCEL;
```

Maintain Standby Database - Manual Recovery

- Use FROM ‘location’ option.

```
RECOVER FROM ‘location’ STANDBY DATABASE;
```

- Suggested file to apply is returned.
- Log_archive_dest_1 is default location.
- Apply logs initially brought over to sync up with primary.
- Manual or sustained maintenance mode may now be used.

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Maintain Standby Database - Manual Recovery

By default, the manual recovery command looks for archived redo logs in log_archive_dest_1 if you do not provide the [from ‘location’] parameter. Using this option to point to the directory that standby_archive_dest points to will allow manual recovery to use the files placed by RFS.

As long as the standby database is mounted, in a maintenance mode, or in read only, the remote archive process should work, automatically putting archive files out on the standby_archive_dest directory. These can be applied via the manual process if desired.

Useful Views

- V$ARCHIVE
  - Redo log files in need of archiving
  - One row per thread
- V_$ARCHIVED_LOG
  - Record inserted after successful archival
  - If cleared, name column is NULL
  - Record inserted when archive log is restored
- V_$ARCHIVE_DEST

  Complete status of each destination
- V_$ARCHIVED_LOG
- V_$ARCHIVE_PROCESSES

11-15

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Useful Views

V$ARCHIVE

This view contains information on redo log files in need of archiving. Each row provides information for one thread.

V$ARCHIVED_LOG

This view displays archived log information from the controlfile including archive log names. An archive log record is inserted after the online redo log is successfully archived or cleared (name column is NULL if the log was cleared). If the log is archived twice, there will be two archived log records with the same THREAD#, SEQUENCE#, and FIRST_CHANGE#, but with a different name. An archive log record is also inserted when an archive log is restored from a backup set or a copy.

V$ARCHIVE_DEST

This view, new in Oracle8*i*, describes, for the current instance, all the archive log destinations, their current value, mode, and status.

Troubleshoot Archiver Problems

- Monitor alert.log file and use the views to check if remote archive is functional.
- Ensure REOPEN option is included in the log_archive_dest_n parameter.
- Use `alter system set log_archive_dest_n` command to reset the parameter dynamically.
- Optional destination?

  Manually recover missing archive logs on standby
- Mandatory destination?

  Use `archive log all` to initiate archive process.

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Troubleshoot Archive Problems

Usually when archive fails to archive to a destination, this event is posted in the alert log. So periodic monitoring of alert.log file is essential. You could set up a scheduled job to notify you if any errors are reported in alert.log

Use the show parameter command or the view V$ARCHIVE_DEST periodically to ensure that remote archive process has not failed.

If the parameter has not been set or needs change, then use 'alter system set log_archive_dest_n' command to reset the parameter dynamically.

If the remote destination is set as OPTIONAL, then ensure that any missing archive logs are restored on standby manually.

Reestablish the RFS Process

- If remote archival fails with ORA-3113, take the following steps.
- Make sure the standby database is mounted.
- Make sure the listener on the standby host is up and listening for the standby database.
- Restart the ARCH process on the PRIMARY database to recreate the RFS process on the remote.
- On the primary, reset the failed destination.

11-17

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Reestablish the RFS Process

If the remote archival fails with ORA3113 error, recreate the RFS process on the standby host, and re-synchronize the standby database so that automatic updating can continue.

1. Make sure the standby database is mounted as a standby.
   - `sqlplus> startup nomount pfile="/xx/xx/initSID.ora"`
   - `sqlplus> alter database mount standby database`
2. Make sure the listener on the standby host is listening for the standby database.
   - `$ lsnrctl start`
   - `$ lsnrctl services`
3. Bounce the ARC0 process on the primary to re-create the RFS process.
   - `sqlplus> alter system archive log stop;`
   - `sqlplus> alter system archive log start;`
4. On the primary, reset the failed destination. This will clear the errors against this particular destination and continues automatic archiving to the destination. View the alert_SID.log (tail) in the primary host to make sure this is happening.

```
alter system set log_archive_dest_2="service=XXXX reopen=120"
```

Recovering Standby Database

- Manually recover missing archives on standby
  - ftp missing archive logs from primary
  - Manually apply using `recover standby database` command
- Resume sustained recovery mode
- Recover from 'location' standby database
  - Archive files must be transferred from Primary to Standby
  - Set AUTORECOVERY on
  - Restart recovery
  - Monitor the progress and for errors

11-18

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Recovering Standby Database

Restore archivelogs to remote location

There may be one or more archive logs missing on the standby database; a "gap" in the sequence. You must use manual recovery on the standby to bridge the gap, then resume sustained recovery.

1. Review v$log_history of both primary and standby or look in the archive destinations to see what logs are missing.
2. Manually transfer the missing logs to the standby host.
3. Apply the missing logs manually using

   ```
   sqlplus> recover standby database;
   ```

   Note: You must use manual recovery because the remote archival process has not updated the standby database control file for the missing logs, so sustained recovery mode would not see the archive files.
4. When application of the missing logs is complete, resume sustained recovery:

   ```
   sqlplus> recover managed standby database;
   ```

Read Only Mode of Standby

- Create locally managed temporary tablespaces on primary.
- Assign locally managed temporary tablespaces to the users on primary.
- Recreate the standby control file and archive the current redo log.
- Mount the standby database and recover to suitable point in time.
- Open the standby in read only mode.
- Add tempfile to the temporary tablespace.
- Perform sort queries.

11-19

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Read Only Mode of Standby Database

Standby database can be opened in read only mode to run queries. However, if the queries require a sort operation, then the sort has to be performed in locally managed temp files. The procedure to accomplish this is illustrated below:

- Create locally managed temporary tablespaces on primary
- CREATE TEMPORARY TABLESPACE READ_ONLY_SORT TEMPFILE'...' SIZE 10M EXTENT MANAGEMENT LOCAL UNIFORM SIZE 1M;
- Assign locally managed temporary tablespaces to the users on primary
- ALTER USER SCOTT TEMPORARY TABLESPACE READ_ONLY_SORT;
- Recreate the standby control file and archive the current redo log. You need not copy the temporary file over to the standby. It can be created.
- ALTER DATABASE CREATE STANDBY CONTROLFILE AS ... ;
- Mount the standby database and recover to suitable point in time
- STARTUP NOMOUNT PFILE=....
  ALTER DATABASE MOUNT STANDBY DATABASE;
  RECOVER STANDBY DATABASE UNTIL CANCEL;
  ...
  CANCEL
- Open the standby in read only mode
- ALTER DATABASE OPEN READ ONLY;
- Add tempfile to the temporary tablespace
- ALTER TABLESPACE READ_ONLY_SORT ADD TEMPFILE '...' SIZE 10M;
- Connect as the normal user and run the queries.

Activating the Standby Database

If the primary database fails, the standby database can be used:

- Archive the primary database's current online redo logs (if possible).
- Clear the online redo logs on standby database.
- Transfer and apply the archives to the standby database.
- Activate the standby database.

  Open reset logs option may be necessary.
- Allowing future user connections to be established on the standby database.

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

Effects of Physical Structure Changes

PRIMARY DATABASE	STANDBY DATABASE
Adding data files	
• Control file change (Redo created)	• Control file is updated, automatically
• Data dictionary change (Redo crated)	• Manually create data file and restart recovery
Dropping data files	
• Control file change (Redo created)	• Control file is updated automatically
• Data dictionary change (Redo crated	• Manually delete file at the OS
Resizing data files	
• Control file, dictionary change	• Automatically propagated
Status of data files	
• Read-only, Offline, etc.,	• Automatically propagated

11-21

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Effects of Physical Structure Change

As the standby is in mounted state, only data file has relevance not tablespace. Generally adding or dropping online redo logs at primary does not affect standby.

Adding Data Files

When data files are added in the primary database, a special redo record is written to update the data dictionary. This gets propagated through the archive log to the standby database. However, if the file itself is not manually copied over to the standby, the recovery process will fail when it attempts to use the non existent file. So file may have to be created with alter database create datafile command.

- REDO RECORD - Thread:1 RBA: 0x000058.00000007.007b LEN: 0x00bd VLD: 0x03
- COMMIT SCN scn: 0x0000.0000176c 11/09/95 12:31:10
- CHANGE #1 CLASS: 15 DBA: 0x08000106 INC: 0x00001001 SEQ: 0x000003f9 OPCODE 5.4
- ktucm redo: slt: 0x001a sqn: 0x0000000a srt: 0 sta: 9 flg: 0x2
- ktucf redo: uba: 0x08000139.0003.05 ext: 0 spc: 1623 fbi: 0
- CHANGE #2 CLASS: 0 DBA: 0x00000000 INC: 0x00000000 SEQ: 0x00000000 OPCODE 17.9
- Create of tablespace #7: 'ADD_FILE'

Effects of Physical Structure Changes (continued)

Dropping Data Files

When datafile or tablespace is dropped from the primary database, a special redo produced to update the data dictionary. This, when propagated to the standby database, will update the control file in the standby database. However, the file itself may have to be removed at the OS level.

- REDO RECORD - Thread:1 RBA: 0x00005b.00000219.0083 LEN: 0x0053 VLD: 0x03
- COMMIT SCN scn: 0x0000.0000177a 11/09/95 13:26:30
- CHANGE #1 CLASS: 11 DBA: 0x08000002 INC: 0x00000001 SEQ: 0x0000050e OPCODE 5.4
- ktucm redo: slt: 0x0014 sqn: 0x0000000d srt: 0 sta: 9 flg: 0x0

File Status Changes

File status changes including resizing are propagated to the standby database via archive logs so no additional effort is required..

REDO RECORD - Thread:1 RBA: 0x00005c.0000000a.0198 LEN: 0x008b VLD: 0x03

COMMIT SCN scn: 0x0000.00001781 11/09/95 13:55:00

CHANGE #1 CLASS: 13 DBA: 0x08000084 INC: 0x00000801 SEQ: 0x0000043f OPCODE 5.4

ktucm redo: slt: 0x0008 sqn: 0x0000000b srt: 0 sta: 9 flg: 0x2

ktucf redo: uba: 0x080000da.0004.06 ext: 1 spc: 1316 fbi: 1

CHANGE #2 CLASS: 0 DBA: 0x00000000 INC: 0x00000000 SEQ: 0x00000000 OPCODE 17.6

Tablespace offline normal

Checkpointed at scn: 0x0000.00001780 11/09/95 13:55:00

thread:1 rba:(0x5c.9.e1)

enabled threads: 01000000 00000000 00000000 00000000 00000000 00000000

00000000 00000000

file #3. created at scn: 0x0000.0000110d

REDO RECORD - Thread:1 RBA: 0x000063.00000004.0190 LEN: 0x008b VLD: 0x03

COMMIT SCN scn: 0x0000.00001a72 11/09/95 14:57:14

CHANGE #1 CLASS: 17 DBA: 0x08000188 INC: 0x00001801 SEQ: 0x0000099d OPCODE 5.4

ktucm redo: slt: 0x001b sqn: 0x00000012 srt: 0 sta: 9 flg: 0x2

ktucf redo: uba: 0x080001ef.000e.06 ext: 1 spc: 1342 fbi: 0

CHANGE #2 CLASS: 0 DBA: 0x00000000 INC: 0x00000000 SEQ: 0x00000000 OPCODE 17.8

Tablespace read only

Checkpointed at scn: 0x0000.00001a71 11/09/95 14:57:14

thread:1 rba:(0x63.4.47)

enabled threads: 01000000 00000000 00000000 00000000 00000000 00000000

00000000 00000000

file #3. created at scn: 0x0000.0000110d

REDO RECORD - Thread:1 RBA: 0x000063.00000006.01c9 LEN: 0x0083 VLD: 0x03

Effects of Physical Structure Changes (continued)

File Status Changes (continued)

COMMIT SCN scn: 0x0000.00001a74 11/09/95 14:57:26

CHANGE #1 CLASS: 11 DBA: 0x08000002 INC: 0x00000001 SEQ: 0x0000074d OPCODE 5.4

ktucm redo: slt: 0x0001 sqn: 0x00000013 srt: 0 sta: 9 flg: 0x0

CHANGE #2 CLASS: 0 DBA: 0x00000000 INC: 0x00000000 SEQ: 0x00000000 OPCODE 17.7

Tablespace read write after read only at scn: 0x0000.00001a71

Checkpointed at scn: 0x0000.00001a73 11/09/95 14:57:26

thread:1 rba:(0x63.6.1c9)

enabled threads: 01000000 00000000 00000000 00000000 00000000 00000000

00000000 00000000

file #3. created at scn: 0x0000.0000110d

Resize File

- REDO RECORD - Thread:1 RBA: 0x00008b.0000029e.0010 LEN: 0x0027 VLD: 0x01
- CONTINUE SCN scn: 0x0000.00001b5c 11/10/95 12:39:17
- CHANGE #1 CLASS: 0 DBA: 0x00000000 INC: 0x00000000 SEQ: 0x00000000 OPCODE17.4
- Datafile resize marker - file: 3 old size: 275 new size: 300
- In recovery process trace file:
  - Fri Nov 10 12:43:28 1995
  - *** SESSION ID:(6.3) 1995.11.10.12.43.28.000
  - point 1: new_fst_blk=0, old_size=565248, new_size=275
  - point 2: old_size=275, new_size=300
  - point 3: old_size=275, new_size=300

Effects of Physical Structure Changes

- Altering log files
  - Adding/dropping online log files will have no effect on standby
  - Invalidating standby

    Alter database clear unarchived logfile

    Resetlogs on the Primary
- Enabling/Disabling Threads
  - No effect on standby

11-24

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Effects of Physical Structure Changes (continued)

Altering log files affects the standby site if you refresh the control file. If you refresh the control file, then you need to drop any extra log groups and members to reduce the database activate (resetlogs) interval. Again, in 7.3.3 and higher, pre-clear the logs.

Prior to 7.3, the commands shown in the slide should not affect your standby database. However, if you refresh the control file, one need to drop any extra log groups and members to reduce the open resetlogs interval. There is no clear logfile command prior to 7.3.

Do not ever disable threads in your standby database because media recovery needs to know all available enabled threads for thread switching and merging.

Effects of Physical Structure Changes

Altering Control Files

- **Create controlfile noresetlogs**
  - **Stop recovery on Standby site**
  - **Refresh standby control file is required**
  - **Restart recovery**
- **Create controlfile resetlogs**

  Recreating standby database is required

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Effects of Physical Structure Changes

Altering Initialization Parameters may affect performance or effectiveness of standby

- Compatible, db_block_buffers, log_archive_buffer_size, max_data_files, max_rollback_segments, and so on
- Stop recovery
- Restart standby instance
- Mount standby database
- Start recovery

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Effects of Physical Structure Changes (continued)

This is true for both 7.3 and prior to 7.3 configurations.

Altering Physical Structure at Primary Database

Update the standby database's control file.

- Methods for adding data files:
  - Copy the data file from the primary to the standby database.
  - Add the data file to the standby database using `alter database create datafile`.
  - You could re-create the control file for the standby database.
- Altering Log Files:
  - Adding or dropping log file groups does not affect the standby database.
  - Clearing log files on the primary database invalidates standby.

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

Refreshing Standby Database Control File

- Refresh or create an updated copy of the standby database's control file.
  1. Halt the recovery process at the standby database.
  2. Re-create the standby control file from the primary database.
  3. Archive the current online redo logs from the primary database.
  4. Transfer control file and archived logs.
  5. Mount the standby database and restart the recovery process.
- Re-creating control file on primary requires refreshing control file on the standby.

11-28

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Refreshing Standby Database Control File

You can invalidate the control file if the primary site recreates its controlfile and changes one of the following:

- Maximum number of redo log file groups or members
- Maximum number of data files
- Maximum number of instances that can concurrently mount and open database

Creating a controlfile with resetlogs option will invalidate the entire standby database.

Nologging Operations

Consider the scenario:

- Data is loaded in a table with nologging option.
- Archive logs are applied at the standby.
- Media recovery is canceled.
- Standby is opened in read only mode for queries, but get errors.
  - ERROR at line 1:

    ORA-01578: ORACLE data block corrupted (file # 3, block # 753)
  - ORA-01110: data file 3: '/u03/user/db02/STDBY/DISK3/data01.dbf'
  - ORA-26040: Data block was loaded using the NOLOGGING option

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

Nologging Operations

Data from nologging operations at primary are not recorded in log files. Propagate these as follows:

- **Take an online backup of all the data files comprising the tablespace(s) involved from the primary database**
- **Archive the current redo log from the primary.**
- **Take the affected data files offline at the standby database**
- **Transfer the data file copies and current archived logs from the primary database to the standby database and perform recovery at standby.**

11-30

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Nologging Operations

Operation with the Nologging (Unrecoverable) Option

When commands such as direct load; create table/index with nologging; insert with APPEND hint are used on the primary database, these operations are not recorded in the redo logs and hence are not propagated to the standby database through the archive logs.

These transactions may have to be propagated separately using the following steps:

- Take an online backup of all the data files comprising the tablespaces involved from the primary database.
- Archive the current redo log from the primary
- If necessary, take the affected data files offline at standby database and drop the tablespace.
- Transfer the data file copies and current archived logs from the primary database to the standby database and perform recovery at standby.

Quick Summary for Standby Databases

PRIMARY DATABASE	STANBY DATABASE
Recreate control file	
• without resetlogs option	• refresh control file (may drop logs pre-clear groups, rename files)
• with resetlogs	• refresh entire standby database
Data Files	
• add a data file	• alter database create datafile, then restart recovery
• drop a data file	• delete file at operating system
• status change	• no maintenance needed
Log files	
• add / drop	• no maintenance needed
• clear unarchived log	• refresh entire standby database
• reset logs	• refresh entire standby database

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

Common Mistakes to Avoid

- Do not propagate incomplete archive logs.
- Use the correct recovery command.

  `recover standby database ...`
- Ensure that all data files needed are online during recovery.
- Unless required to activate the standby database, do not apply online redo logs from the primary database.

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Control File Dump Main Section

Instance name: sb02

```
DUMP OF CONTROL FILES, Seq # 779 = 0x30b
FILE HEADER:
        Software vsn=135266304=0x8100000, Compatibility
Vsn=134217728=0x8000000
        Db Id=1120036664=0x42c26738, Db Name='DB02'
        Control Seq=779=0x30b, File size=808=0x328
        File Number=0, Blksiz=4096, File Type=4 BACKUP
CONTROL
        DATABASE ENTRY:
         (blkno = 0x1, size = 192, max = 1, in-use = 1,
last-recid= 0)
DF Version: creation=0x8100000 compatible=0x8000000,
Date  08/04/99 13:03:52
 DB Name "DB02"
 Database flags = 0x00005447
 Controlfile Creation Timestamp  08/04/99 13:03:53
 Incmplt recovery scn: 0x0000.00054ef7
```


Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

Control File Dump Main Section

Standby Control File Dump

The following points may be of interest in the control file dump of the standby database.

Instance Name, DB Name : The two values would be different.

File Type: 4 BACKUP CONTROL

Database flags =

0x0000400 /* StandBy Database - control file for hot standby */

0x0000040 /* Incomplete Recovery Allowed when resetting logs */

0x0000004 /* DB OPEN RESETLOGS required */

0x0000002 /* if dictionary must be checked with control file */

0x0000001 /* whether media recovery enabled */

Control File Dump Log File Section

```
Redo Version: creation=0x8100000 compatable=0x8100000
 #Data files = 7, #Online files = 7
 Database checkpoint: Thread=1 scn: 0x0000.00054e8e
 Threads: #Enabled=1, #Open=1, Head=1, Tail=1
 enabled  threads:  01000000 00000000 00000000 00000000
00000000 00000000  00000000 00000000
 Max log members = 3, Max data members = 1
LOG FILE #1:
  (converted name #1)
/u03/user/db02/STDBY/DISK3/redo0101.log
  (converted name #2)
/u03/user/db02/STDBY/DISK4/redo0102.log
 Thread 1 redo log links: forward: 2 backward: 0
 siz: 0x1000 seq: 0x00000091 hws: 0x1 bsz: 512 nab:
0xffffffff flg: 0x8 dup: 2
 Archive links: fwrd: 0 back: 0 Prev scn: 0x0000.00054e8c
 Low scn: 0x0000.00054e8e 11/29/99 14:18:12
 Next scn: 0xffff.ffffffff 11/29/99 14:17:43
```

11-34

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Control File Dump Log File Section

Thread is enabled publicly but not opened.

Control File Dumps

```
DATA FILE #1:
  (converted name #5)
/u03/user/db02/STDBY/DISK1/system01.dbf
creation size=25600 block size=4096 status=0xe head=5
tail=5 dup=1
 tablespace 0, index=1 krfil=1 prev_file=0
 unrecoverable scn: 0x0000.00000000 01/01/88 00:00:00
 Checkpoint cnt:238 scn: 0x0000.00054ef7 12/01/99 12:42:19
 Stop scn: 0xffff.ffffffff 11/29/99 11:29:47
 Creation Checkpointed at scn:  0x0000.00000004 08/04/99
13:04:19
 thread:1 rba:(0x1.3.10)
 enabled  threads:  01000000 00000000 00000000 00000000
00000000 00000000  00000000 00000000
 Offline scn: 0x0000.00000000 prev_range: 0
 Online Checkpointed at scn:  0x0000.00000000
 thread:0 rba:(0x0.0.0)
 enabled  threads:  00000000 00000000 00000000 00000000
00000000 00000000  00000000 00000000
 Hot Backup end marker scn: 0x0000.00000000
```


Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

Control File Dumps

```
ub2   kccfesta;                          /* file status */
#define   KCCFESTS 0x0001                /* belongs to System TableSpace */
#define   KCCFEONL 0x0002                /* file is ONLine */
#define   KCCFERDE 0x0004                /* ReaDing is Enabled */
#define   KCCFECGE 0x0008                /* ChanGing is Enabled */
#define   KCCFEMRR 0x0010                /* Media Recovery Required */
#define   KCCFEGEM 0x0020   /* Generate End hot backup Marker at next open */
#define   KCCFECKD 0x0040   /* File entry generated by check dictionary */
#define   KCCFESOR 0x0080   /* Save Offline scn Range at next checkpoint */
#define   KCCFERMF 0x0100                /* Renamed Missing File */
#define   KCCFEGOI 0x0200                /* Generate Off-line Immediate marker */
#define   KCCFECUV 0x0400   /* Checkpoint by instance where UnVerified */
#define   KCCFEDRP 0x0800                /* offline to be DRoPped */
#define   KCCFEWCC 0x1000   /* Was at Clean Chkpt: clean up to ctrlfile chkpt */
#define   KCCFEORL 0x2000   /* Offlined by ResetLogs: dict check may online */
```

I think this actually only means that the file was in a read only tablespace when the standby controlfile was created. The enable bits are ignored during recovery. In a standby database (or backup controlfile) the stop SCN is infinite because we do not know if the tablespace will become read write again some day. Recovery still needs to apply redo generated after the file goes read write. Basically standby recovery uses totally different mechanisms for dealing with read only and offline normal tablespaces.

Case Study 1

Problem:

- Corrupted archive file
- Recovery session bombs with the following errors

```
ORA-00283: Recovery session canceled due
   to errors
ORA-00354: corrupt redo log block header
...
```

Solution:

- Find a good copy of the archive log
- Start the recovery process again and reapply the archive log

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

Case Study 2

Situation:

- Primary Database adds a new data file.
- Standby database has not been adjusted.

Solution:

- Use the command `alter database create datafile 'filename'`.
- Start recovery again.

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Summary

In this lesson, you should have learned about the following:

- Use of standby database to minimize downtime
- How to configure a standby database
- How to propagate data loading operations with nologging option to the standby
- How to propagate structural changes to the standby database
- How to use the standby database in read only mode

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

ORACLE

Credits/References

- *Oracle8i Backup and Recovery*
- *Oracle8i Reference Guide*
- *Standby Brownbags* by Lawrence To

Confidential
Copyright © Oracle Corporation, 2000. All rights reserved.

A
Supplemental Information

Opcodes for Oracle8

Layers from kco.h

Layer 1: KCOCOTCT - Transaction control
Layer 2: KCOCOTRD - Transaction read
Layer 3: KCOCOTUP - Transaction update
Layer 4: KCOCOTBK - Transaction block

opcodes from ktbcts.h

opcode 1 : KTBOPCLN - CLeaNout opcode
opcode 2 : KTBPHCLN - PHysical CLeaNout opcode
opcode 3 : KTBSARC - Single ARray Change
opcode 4 : KTBMARC - Multiple Changes to an Array
opcode 5 : KTBOPFMB - ForMat Block

Layer 5: KCOCOTUN - Transaction undo

opcodes from ktucts.h

opcode 1 : KTURDB - Undo block or undo segment header (Kernel Transaction Undo Redo Data Block)
opcode 2 : KTURDH - Update rollback segment header (Kernel Transaction Undo Redo Data Head)
opcode 3 : KTURBG - Rollout a transaction begin (Kernel Transaction Undo Redo BeGin transaction)
opcode 4 : KTURCM - Commit Transaction (transaction table update) (Kernel Transaction Undo Redo ComMit)
opcode 5 : KTUFMT - Create Rollback Segment (format) -no undo record
opcode 6 : KTUIRB - Rollback record index in an undo block
opcode 7 : KTUUBG - Begin transaction (transaction table update)
opcode 8 : KTURMR - Mark transaction as dead
opcode 9 : KTUUAE - Undo routine to rollback the extend of a RBS
opcode 10 : KTUREH - Redo to perform the rollback of extend of RBS to the segment header
opcode 11 : KTUBRB - Rollback DBA in transaction table entry
opcode 12 : KTURST - Change transaction state (transaction table entry)
opcode 13 : KTURCT - Convert rollback segment format (V6 -> V7)
opcode 14 : KTURUC - Change extent allocation parameters in RBS
opcode 15 : KTURCTS
opcode 16 : KTURCTU
opcode 17 : KTURCTM
opcode 18 : KTURPX
opcode 19 : KTUTSL - Tx Start audit Log record
opcode 20 : KTUTSC - Tx Continue audit Log record
opcode 21 : KTURCVD
opcode 22 : KTURPHC

Layer 6: KCOCODCF - Control file

opcodes from tbs.h

opcode 1 : TBSCRDF
opcode 2 : TBSCADF
opcode 3 : TBSCOFL
opcode 4 : TBSCONL
opcode 5 : TBSCRDW
opcode 6 : TBSCRDO
opcode 7 : TBSCRTS
opcode 8 : TBSCATS
opcode 9 : TBSCUTP

Layer 10: KCOCODIX - Index

opcodes from kdi.h

opcode 1	: KDICPDO	-	Physical do - load index block (Loader with direct mode)
opcode 2	: KDICLIN	-	Insert leaf row
opcode 3	: KDICLPU	-	Purge leaf row
opcode 4	: KDICLDE	-	Mark leaf row deleted
opcode 5	: KDICLRE	-	Restore leaf row (clear leaf delete flags)
opcode 6	: KDICLOK	-	Lock index block
opcode 7	: KDICULO	-	Unlock index block
opcode 8	: KDICLNE	-	Initialize new leaf block
opcode 9	: KDICAIR	-	Apply ITL Redo : call ktbair only (trans layer do)
opcode 10	: KDICLNX	-	Set leaf block next link.
opcode 11	: KDICLPR	-	Set leaf block previous link
opcode 12	: KDICRSP	-	Init root block after split
opcode 13	: KDICLEM	-	Make leaf block empty
opcode 14	: KDICIMA	-	Restore block before image
opcode 15	: KDICBIN	-	Branch block row insert
opcode 16	: KDICBPU	-	Branch block row purge
opcode 17	: KDICBNE	-	Initialize new branch block
opcode 18	: KDICLUP	-	Update keydata in row
opcode 19	: KDICLCL	-	Clear row's split flag
opcode 20	: KDICLSE	-	Set row's split flag
opcode 21	: KDICUGE	-	General undo above the cache (undo callback)
opcode 23	: KDICREB	-	Restore block to b-tree
opcode 24	: KDICSIT	-	Shrink ITL (transaction entries)
opcode 25	: KDICFRB	-	Format root block redo
opcode 26	: KDICUFB	-	Undo of format root block
opcode 27	: KDICUFR	-	Redo for undo of format root block
opcode 28	: KDICUMG	-	Undo for migrating block
opcode 29	: KDICMG	-	Redo for migrating block
opcode 30	: KDICLNU	-	IOT leaf block nonkey update
opcode 31	: KDICDLR	-	Direct load root redo
opcode 32	: KDICCOM	-	Combine operation for insert and restore rows

Layer 11: KCOCODRW - Row Access

opcodes from kdocts.h

opcode 1	: KDOIUR	-	Interpret Undo Redo (Undo)
opcode 2	: KDOIRP	-	Insert Row Piece
opcode 3	: KDODRP	-	Drop Row Piece
opcode 4	: KDOLKR	-	Lock Row Piece
opcode 5	: KDOURP	-	Update Row Piece
opcode 6	: KDOORP	-	Overwrite Row Piece
opcode 7	: KDOMFC	-	Manipulate First Column (add or delete the 1st column)
opcode 8	: KDOCFA	-	Change Forwarding Address
opcode 9	: KDOCKI	-	Change cluster Key Index
opcode 10	: KDOSKL	-	Set Key Links (change the forward & backward key links on a cluster key)
opcode 11	: KDOQMI	-	Quick Multi-Insert (ex. insert as select...)
opcode 12	: KDOQMD	-	Quick Multi-Delete
opcode 13	: KDOTBF	-	Toggle Block header Flags

Layer 12: KCOCODCL - Cluster

Layer 13: KCOCOTSG- Transasction segment

opcodes from ktscts.h

opcode 1 : KTSDSF - Data Segment Format
opcode 2 : KTSFFB - Format Freelist Block
opcode 3 : KTSRCTU - Redo for Convert To Unlimited extents format
opcode 4 : KTSRFSH - Fix Segment Header by moving its extent to ext 0
opcode 5 : KTSFRBFMT - Format data block
opcode 6 : KTSFRBLNK - Set link value on block
opcode 7 : KTSFRGRP - Freelist related fgroup/segheader redo
opcode 8 : KTSFUGRP - Freelist related fgroup/segheader undo
opcode 9 : KTSFUNLK - Undo for linking block to trx freelist
/* BITMAP redo/undo */
opcode 10 : KTSBSFO - Format Segment header
opcode 11 : KTSBBFO - Format bitmap Block
opcode 12 : KTSBIFO - Format bitmap Index block
opcode 13 : KTSBBREDO - Redo for bitmap block (BMB)
opcode 14 : KTSBBUNDO - Undo for BMB
opcode 15 : KTSBIREDO - Redo for Index map
opcode 16 : KTSBIUNDO - Undo for Index map

Layer 14: KCOCOTEX - Transaction extent

opcodes from kte2.h

opcode 1 : KTECUSH - ? Unlock Segment Header
opcode 2 : KTECRLK - ? Lock Segment Header
opcode 3 : KTEFRCU - Redo for Conversion to Unlimited format
opcode 4 : KTEOPEMREDO - Extent Operation Redo
opcode 5 : KTEOPEUNDO - Extent Operation Undo
opcode 6 : KTEOPEFREDO - Extent Map Format Redo
opcode 7 : KTECNV - Convert Segment Header
opcode 8 : KTEOPUTRN - Undo for Truncate operations, flush the object
opcode 9 : KTEFUCTL - Undo for reformat of a ctl block
opcode 10 : KTEFRCTL - Redo to facilitate above undo

Layer 15: KCOCOTTS - Table space (ktt.h)

Layer 16: KCOCOQRC- Row cache

Layer 17: KCOCORCV - Recovery (redo)

opcodes from kcv.h

opcode 1 : KCVOPEHB - End Hot Backup (clears the hot backup in-progress flags in the indicated list of files)
opcode 2 : KCVOPENT - ENable Thread (creates a redo record signalling that a thread has been enabled)
opcode 3 : KCVOPCRM - Crash Recovery Marker
opcode 4 : KCVOPRSZ - ReSiZeable datafiles
opcode 5 : KCVOPONL - tablespace ONline
opcode 6 : KCVOPOFF - tablespace OFFline
opcode 7 : KCVOPRDW - tablespace ReaD Write
opcode 8 : KCVOPRDO - tablespace ReaD Only
opcode 9 : KCVOPADD - ADDing datafiles to database
opcode 10 : KCVOPDRP - tablespace DRoP
opcode 11 : KCVOPTPR - Tablespace PitR

Layer 18: KCOCOHLB - Hot-backup Log Blocks

opcodes from kcb.h

opcode 1 : KCBKCOLB - Log Block image
opcode 2 : KCBKCORV - Recovery testing

Layer 19: KCOCODLB - Direct loader log blocks

opcodes from kcbl.h

opcode 1 : KCBLCOLB - Direct Block Logging

opcode 2 : KCBLCOIR - Invalidate Range
opcode 3 : KCBLCRLB - Direct Block Re-logging
opcode 4 : KCBLCRIR - Invalidate Range Re-logging

Layer 20: KCOCOKCK - Compatibility segment operations

opcodes from kck.h

opcode 1 : KCKFCS - Format Compatibility Segment
opcode 2 : KCKUCS - Update Compatibility Segment

Layer 21: KCOCOLFS - LOB segment operations

opcodes from kdl2.h

opcode 1 : KDLOPWRI - WRIte data into ILOB data block

Highlighted Trace Files

Controlfile dump

```
1.  Dump file /u02/home/usupport/udump/ora_12357.trc
2.  Oracle8i Enterprise Edition Release 8.1.5.0.0 - Production
3.  With the Partitioning, Parallel Server and Java options
4.  PL/SQL Release 8.1.5.0.0 - Production
5.  ORACLE_HOME = /u02/app/oracle/product/8.1.5
6.  System name:        HP-UX
7.  Node name:  ops1
8.  Release:    B.11.00
9.  Version:    A
10. Machine:    9000/889
11. Instance name: V815A
12. Redo thread mounted by this instance: 1
13. Oracle process number: 10
14. Unix process pid: 12357, image: oracle@ops1 (TNS V1-V3)

15. *** 2000.01.03.17.28.34.526
16. DUMP OF CONTROL FILES, Seq # 1170 = 0x492
17.
18. << Generic File Header starts >>
19.
20.  FILE HEADER:
21.          Software vsn=135266304=0x8100000, Compatibility Vsn=134217728=0x8000000
22.          Db Id=3453975259=0xcddf7adb, Db Name='V815'
23.          Control Seq=1170=0x492, File size=3714=0xe82
24.          File Number=0, Blksiz=2048, File Type=1 CONTROL
25.
26. << Generic file header stops, database information starts >>
27.
28.          DATABASE ENTRY:
29.           (blkno = 0x1, size = 192, max = 1, in-use = 1, last-recid= 0)
30. DF Version: creation=0x8100000 compatible=0x8000000, Date  09/18/99 11:48:43
31. DB Name "V815"
32. Database flags = 0x00004000
33. Controlfile Creation Timestamp  09/18/99 11:48:45
34. Incmplt recovery scn: 0x0000.00000000
35. Resetlogs scn: 0x0000.00000001 Resetlogs Timestamp  09/18/99 11:48:43
36. Prior resetlogs scn: 0x0000.00000000 Prior resetlogs Timestamp  01/01/88 00:00:00
37. Redo Version: creation=0x8100000 compatable=0x8100000
38. #Data files = 8, #Online files = 8
39. Database checkpoint: Thread=1 scn: 0x0000.000460c2
40. Threads: #Enabled=2, #Open=2, Head=1, Tail=2
41. enabled  threads:  01100000 00000000 00000000 00000000 00000000 00000000
42.   00000000
43. Max log members = 3, Max data members = 1
44. Arch list: Head=0, Tail=0, Force scn: 0x0000.00040f61
45. Controlfile Checkpointed at scn:  0x0000.0006a482 12/14/99 17:22:10
46. thread:0 rba:(0x0.0.0)
47. enabled  threads:  00000000 00000000 00000000 00000000 00000000 00000000
```

48. 00000000
49.
50. ***<< Database information stops, checkpoint progress records starts >>***
51.
52. CHECKPOINT PROGRESS RECORDS:
53. (blkno = 0x12, size = 104, max = 32, in-use = 2, last-recid= 2)
54. THREAD #1 - status:0x2 flags:0x0 dirty:3
55. low cache rba:(0x15d.3b.0) on disk rba:(0x15d.3e.0)
56. on disk scn: 0x0000.0006a47d 12/09/99 08:09:21
57. resetlogs scn: 0x0000.00000001 09/18/99 11:48:43
58. heartbeat: 384021191 mount id: 3460566510
59. THREAD #2 - status:0x2 flags:0x0 dirty:0
60. low cache rba:(0xffffffff.ffffffff.ffff) on disk rba:(0x7.6.0)
61. on disk scn: 0x0000.0004af2e 12/01/99 18:04:14
62. resetlogs scn: 0x0000.00000001 09/18/99 11:48:43
63. heartbeat: 384763071 mount id: 3460566510
64. THREAD #3 - status:0x0 flags:0x0 dirty:0
65. low cache rba:(0x0.0.0) on disk rba:(0x0.0.0)
66. on disk scn: 0x0000.00000000 01/01/88 00:00:00
67. resetlogs scn: 0x0000.00000000 01/01/88 00:00:00
68. heartbeat: 0 mount id: 0
69. THREAD #4 - status:0x0 flags:0x0 dirty:0
70. low cache rba:(0x0.0.0) on disk rba:(0x0.0.0)
71. on disk scn: 0x0000.00000000 01/01/88 00:00:00
72. resetlogs scn: 0x0000.00000000 01/01/88 00:00:00
73. heartbeat: 0 mount id: 0
74.
75. [records deleted for readability]
76.
77. THREAD #31 - status:0x0 flags:0x0 dirty:0
78. low cache rba:(0x0.0.0) on disk rba:(0x0.0.0)
79. on disk scn: 0x0000.00000000 01/01/88 00:00:00
80. resetlogs scn: 0x0000.00000000 01/01/88 00:00:00
81. heartbeat: 0 mount id: 0
82. THREAD #32 - status:0x0 flags:0x0 dirty:0
83. low cache rba:(0x0.0.0) on disk rba:(0x0.0.0)
84. on disk scn: 0x0000.00000000 01/01/88 00:00:00
85. resetlogs scn: 0x0000.00000000 01/01/88 00:00:00
86. heartbeat: 0 mount id: 0
87.
88. ***<< Checkpoint progress record section stops, redo thread record section starts >>***
89.
90. REDO THREAD RECORDS:
91. (blkno = 0x12, size = 104, max = 32, in-use = 2, last-recid= 2)
92. THREAD #1 - status:0x7 thread links forward:2 back:0
93. #logs:2 **first:1 last:2 current:1** last used seq#:0x15d
94. enabled at scn: 0x0000.00000001 09/18/99 11:48:46
95. disabled at scn: 0x0000.00000000 01/01/88 00:00:00
96. opened at 12/01/99 16:17:51 by instance V815A
97. Checkpointed at scn: 0x0000.000460c2 12/01/99 16:17:51
98. thread:1 rba:(0x15d.33.10)
99. enabled threads: 01100000 00000000 00000000 00000000 00000000 00000000
100. 00000000
101. log history: 353

102. THREAD #2 - status:0x7 thread links forward:0 back:1
103. #logs:2 first:3 last:4 current:4 last used seq#:0x8
104. enabled at scn: 0x0000.00008845 09/18/99 12:16:05
105. disabled at scn: 0x0000.00000000 01/01/88 00:00:00
106. opened at 12/14/99 17:22:10 by instance V815B
107. Checkpointed at scn: 0x0000.0006a482 12/14/99 17:22:10
108. thread:2 rba:(0x8.2.10)
109. enabled threads: 01100000 00000000 00000000 00000000 00000000 00000000
110. 00000000
111. log history: 355
112.
113. ***<<Redo thread record section stops, log file record section starts >>***
114.
115. LOG FILE RECORDS:
116. (blkno = 0x14, size = 72, max = 64, in-use = 4, last-recid= 4)
117. LOG FILE #1:
118. (name #1) **/dev/usupport_vg/rV815_log1_01a.log**
119. (name #2) /dev/usupport_vg/rV815_log1_01b.log
120. Thread 1 redo log links: forward: 2 backward: 0
121. siz: 0x1f4 **seq: 0x0000015d** hws: 0x8 **bsz: 1024** nab: 0xffffffff **flg: 0x8** dup: 2
122. Archive links: fwrd: 0 back: 0 **Prev scn: 0x0000.000088c4**
123. Low scn: 0x0000.00040f75 11/13/99 13:58:56
124. Next scn: 0xffff.ffffffff 09/18/99 12:17:30
125. LOG FILE #2:
126. (name #3) /dev/usupport_vg/rV815_log1_02a.log
127. (name #4) /dev/usupport_vg/rV815_log1_02b.log
128. Thread 1 redo log links: forward: 0 backward: 1
129. siz: 0x1f4 seq: 0x0000015c hws: 0x1b bsz: 1024 nab: 0xfa flg: 0x0 dup: 2
130. Archive links: fwrd: 0 back: 0 Prev scn: 0x0000.000087e4
131. Low scn: 0x0000.000088c4 09/18/99 12:17:30
132. Next scn: 0x0000.00040f75 11/13/99 13:58:56
133. LOG FILE #3:
134. (name #11) /dev/usupport_vg/rV815_log2_01a.log
135. (name #12) /dev/usupport_vg/rV815_log2_01b.log
136. Thread 2 redo log links: forward: 4 backward: 0
137. siz: 0x800 seq: 0x00000007 hws: 0x5 bsz: 1024 nab: 0x6 flg: 0x0 dup: 2
138. Archive links: fwrd: 0 back: 0 Prev scn: 0x0000.00040f61
139. Low scn: 0x0000.0004af26 12/01/99 18:04:11
140. Next scn: 0x0000.0006a481 12/14/99 17:22:10
141. LOG FILE #4:
142. (name #13) /dev/usupport_vg/rV815_log2_02a.log
143. (name #14) /dev/usupport_vg/rV815_log2_02b.log
144. Thread 2 redo log links: forward: 0 backward: 3
145. siz: 0x800 seq: 0x00000008 hws: 0x2 bsz: 1024 nab: 0xffffffff flg: 0x8 dup: 2
146. Archive links: fwrd: 0 back: 0 Prev scn: 0x0000.0004af26
147. Low scn: 0x0000.0006a481 12/14/99 17:22:10
148. Next scn: 0xffff.ffffffff 12/01/99 18:04:11
149.
150. ***<< Log file records section stops, data file record section starts >>***
151.
152. DATA FILE RECORDS:
153. (blkno = 0x17, size = 180, max = 254, in-use = 9, last-recid= 10)
154. DATA FILE #1:

155. (name #5) **/dev/usupport_vg/rV815_system_01.dbf**
156.**creation size=89600 block size=2048 status=0xe** head=5 tail=5 dup=1
157. **tablespace 0**, index=1 krfil=1 **prev_file=0**
158. unrecoverable scn: 0x0000.0000885a 09/18/99 12:16:21
159. Checkpoint cnt:378 scn: 0x0000.000460c2 12/01/99 16:17:51
160. Stop scn: 0xffff.ffffffff 12/01/99 15:50:40
161. Creation Checkpointed at scn: 0x0000.00000004 09/18/99 11:49:17
162. thread:1 rba:(0x1.3.10)
163. enabled threads: 01000000 00000000 00000000 00000000 00000000 00000000
164. 00000000
165. Offline scn: 0x0000.00000000 prev_range: 0
166. Online Checkpointed at scn: 0x0000.00000000
167. thread:0 rba:(0x0.0.0)
168. enabled threads: 00000000 00000000 00000000 00000000 00000000 00000000
169. 00000000
170. Hot Backup end marker scn: 0x0000.00000000
171.DATA FILE #2:
172. (name #6) /dev/usupport_vg/rV815_query_01.dbf
173.creation size=2560 block size=2048 **status=0x86** head=6 tail=6 dup=1
174. tablespace 1, index=2 krfil=2 prev_file=0
175. unrecoverable scn: 0x0000.00000000 01/01/88 00:00:00
176. **Checkpoint cnt:290 scn: 0x0000.000460c2** 12/01/99 16:17:51
177. **Stop scn: 0x0000.000460c2** 12/01/99 16:17:51
178. Creation Checkpointed at scn: 0x0000.000050a9 09/18/99 11:58:14
179. thread:1 rba:(0x59.1e5.b0)
180. enabled threads: 01000000 00000000 00000000 00000000 00000000 00000000
181. 00000000
182. Offline scn: 0x0000.00000000 prev_range: 0
183. Online Checkpointed at scn: 0x0000.00000000
184. thread:0 rba:(0x0.0.0)
185. enabled threads: 00000000 00000000 00000000 00000000 00000000 00000000
186. 00000000
187. Hot Backup end marker scn: 0x0000.00000000
188.DATA FILE #3:
189. (name #7) /dev/usupport_vg/rV815_rbs_01.dbf
190.creation size=6144 block size=2048 status=0xe head=7 tail=7 dup=1
191. tablespace 2, index=3 krfil=3 prev_file=0
192. unrecoverable scn: 0x0000.00000000 01/01/88 00:00:00
193. Checkpoint cnt:290 scn: 0x0000.000460c2 12/01/99 16:17:51
194. Stop scn: 0xffff.ffffffff 12/01/99 15:50:40
195. Creation Checkpointed at scn: 0x0000.000050b2 09/18/99 11:58:16
196. thread:1 rba:(0x59.1ed.a8)
197. enabled threads: 01000000 00000000 00000000 00000000 00000000 00000000
198. 00000000
199. Offline scn: 0x0000.00000000 prev_range: 0
200. Online Checkpointed at scn: 0x0000.00000000
201. thread:0 rba:(0x0.0.0)
202. enabled threads: 00000000 00000000 00000000 00000000 00000000 00000000
203. 00000000
204. Hot Backup end marker scn: 0x0000.00000000
205.DATA FILE #4:
206. (name #8) /dev/usupport_vg/rV815_temp_01.dbf
207.creation size=5120 block size=2048 status=0xe head=8 tail=8 dup=1

```
208. tablespace 3, index=4 krfil=4 prev_file=0
209. unrecoverable scn: 0x0000.00000000 01/01/88 00:00:00
210. Checkpoint cnt:289 scn: 0x0000.000460c2 12/01/99 16:17:51
211. Stop scn: 0xffff.ffffffff 12/01/99 15:50:40
212. Creation Checkpointed at scn:  0x0000.000050bf 09/18/99 11:58:19
213. thread:1 rba:(0x5a.4.a8)
214. enabled  threads:  01000000 00000000 00000000 00000000 00000000 00000000
215.  00000000
216. Offline scn: 0x0000.00000000 prev_range: 0
217. Online Checkpointed at scn:  0x0000.00000000
218. thread:0 rba:(0x0.0.0)
219. enabled  threads:  00000000 00000000 00000000 00000000 00000000 00000000
220.  00000000
221. Hot Backup end marker scn: 0x0000.00000000
222.DATA FILE #5:
223.  (name #9) /dev/usupport_vg/rV815_users_01.dbf
224.creation size=5120 block size=2048 status=0xe head=9 tail=9 dup=1
225. tablespace 4, index=5 krfil=5 prev_file=0
226. unrecoverable scn: 0x0000.00000000 01/01/88 00:00:00
227. Checkpoint cnt:289 scn: 0x0000.000460c2 12/01/99 16:17:51
228. Stop scn: 0xffff.ffffffff 12/01/99 15:50:40
229. Creation Checkpointed at scn:  0x0000.000050cb 09/18/99 11:58:21
230. thread:1 rba:(0x5a.f.a8)
231. enabled  threads:  01000000 00000000 00000000 00000000 00000000 00000000
232.  00000000
233. Offline scn: 0x0000.00000000 prev_range: 0
234. Online Checkpointed at scn:  0x0000.00000000
235. thread:0 rba:(0x0.0.0)
236. enabled  threads:  00000000 00000000 00000000 00000000 00000000 00000000
237.  00000000
238. Hot Backup end marker scn: 0x0000.00000000
239.DATA FILE #6:
240.  (name #10) /dev/usupport_vg/rV815_index_01.dbf
241.creation size=5120 block size=2048 status=0xe head=10 tail=10 dup=1
242. tablespace 5, index=6 krfil=6 prev_file=0
243. unrecoverable scn: 0x0000.00000000 01/01/88 00:00:00
244. Checkpoint cnt:289 scn: 0x0000.000460c2 12/01/99 16:17:51
245. Stop scn: 0xffff.ffffffff 12/01/99 15:50:40
246. Creation Checkpointed at scn:  0x0000.000050d6 09/18/99 11:58:24
247. thread:1 rba:(0x5a.17.a8)
248. enabled  threads:  01000000 00000000 00000000 00000000 00000000 00000000
249.  00000000
250. Offline scn: 0x0000.00000000 prev_range: 0
251. Online Checkpointed at scn:  0x0000.00000000
252. thread:0 rba:(0x0.0.0)
253. enabled  threads:  00000000 00000000 00000000 00000000 00000000 00000000
254.  00000000
255. Hot Backup end marker scn: 0x0000.00000000
256.DATA FILE #7:
257.  (name #15) /dev/usupport_vg/rjaymz.dbf
258.creation size=5632 block size=2048 status=0xe head=15 tail=15 dup=1
259. tablespace 6, index=7 krfil=7 prev_file=0
260. unrecoverable scn: 0x0000.00000000 01/01/88 00:00:00
261. Checkpoint cnt:4 scn: 0x0000.000460c2 12/01/99 16:17:51
```

262. Stop scn: 0xffff.ffffffff 12/01/99 15:50:40
263. Creation Checkpointed at scn: 0x0000.0004607b 12/01/99 15:46:48
264. thread:1 rba:(0x15d.1e.a8)
265. enabled threads: 01100000 00000000 00000000 00000000 00000000 00000000
266. 00000000
267. Offline scn: 0x0000.00000000 prev_range: 0
268. Online Checkpointed at scn: 0x0000.00000000
269. thread:0 rba:(0x0.0.0)
270. enabled threads: 00000000 00000000 00000000 00000000 00000000 00000000
271. 00000000
272. Hot Backup end marker scn: 0x0000.00000000
273. DATA FILE #8:
274. (name #16) /dev/jaymz_vg/rjaymz2.dbf
275. creation size=3584 block size=2048 status=0xe head=16 tail=16 dup=1
276. **tablespace 6, index=7 krfil=8 prev_file=7**
277. unrecoverable scn: 0x0000.00000000 01/01/88 00:00:00
278. Checkpoint cnt:2 scn: 0x0000.000460fb 12/01/99 17:06:46
279. Stop scn: 0xffff.ffffffff 12/01/99 17:06:46
280. Creation Checkpointed at scn: 0x0000.000460fa 12/01/99 17:06:46
281. thread:2 rba:(0x6.21.158)
282. enabled threads: 01100000 00000000 00000000 00000000 00000000 00000000
283. 00000000
284. Offline scn: 0x0000.00000000 prev_range: 0
285. Online Checkpointed at scn: 0x0000.00000000
286. thread:0 rba:(0x0.0.0)
287. enabled threads: 00000000 00000000 00000000 00000000 00000000 00000000
288. 00000000
289. DATA FILE #9: <deleted>
290. creation size=3584 block size=2048 status=0x810 head=0 tail=0 dup=0
291. tablespace 7, index=8 krfil=9 prev_file=0
292. unrecoverable scn: 0x0000.00000000 01/01/88 00:00:00
293. Checkpoint cnt:2 scn: 0x0000.000460fb 12/01/99 17:06:46
294. Stop scn: 0x0000.0004783e 12/21/99 12:06:46
295. Creation Checkpointed at scn: 0x0000.000460fa 12/01/99 17:06:46
296. thread:2 rba:(0x6.21.158)
297. enabled threads: 01100000 00000000 00000000 00000000 00000000 00000000
298. 00000000
299. Offline scn: 0x0000.00000000 prev_range: 0
300. Online Checkpointed at scn: 0x0000.00000000
301. thread:0 rba:(0x0.0.0)
302. enabled threads: 00000000 00000000 00000000 00000000 00000000 00000000
303. 00000000
304.
305. ***<< Data file record section stops, temp file record section starts>>***
306.
307. TEMP FILE RECORDS:
308. (blkno = 0x9b, size = 56, max = 254, in-use = 1, last-recid= 1)
309. **TEMP FILE #1**: External File #101
310. (name #18) **/dev/usupport_vg/rV815/tempfile1.dbf**
311. **creation size=2560 block size=2048 status=0xe** head=18 tail=18 dup=1
312. **tablespace=9**, index=8 **krfil=1** prev_file=0
313. unrecoverable scn: 0x0000.00000000 01/01/88 00:00:00
314.

315. ***<<Temp file record section stops, Tablespace record section starts >>***
316.
317. TABLESPACE RECORDS:
318. (blkno = 0x92, size = 68, max = 254, in-use = 8, last-recid= 8)
319. **TABLESPACE #0 SYSTEM**: recno=1
320. First datafile link=1 Tablespace Flag=0
321. Tablespace PITR mode start scn: 0x0000.00000000 01/01/88 00:00:00
322. Tablespace PITR last completion scn: 0x0000.00000000 01/01/88 00:00:00
323. TABLESPACE #1 QUERY_DATA: recno=2
324. First datafile link=2 Tablespace Flag=0
325. Tablespace PITR mode start scn: 0x0000.00000000 01/01/88 00:00:00
326. Tablespace PITR last completion scn: 0x0000.00000000 01/01/88 00:00:00
327. TABLESPACE #2 RBS: recno=3
328. First datafile link=3 Tablespace Flag=0
329. Tablespace PITR mode start scn: 0x0000.00000000 01/01/88 00:00:00
330. Tablespace PITR last completion scn: 0x0000.00000000 01/01/88 00:00:00
331. TABLESPACE #3 TEMP: recno=4
332. First datafile link=4 Tablespace Flag=0
333. Tablespace PITR mode start scn: 0x0000.00000000 01/01/88 00:00:00
334. Tablespace PITR last completion scn: 0x0000.00000000 01/01/88 00:00:00
335. TABLESPACE #4 USERS: recno=5
336. First datafile link=5 Tablespace Flag=0
337. Tablespace PITR mode start scn: 0x0000.00000000 01/01/88 00:00:00
338. Tablespace PITR last completion scn: 0x0000.00000000 01/01/88 00:00:00
339. TABLESPACE #5 INDX: recno=6
340. First datafile link=6 Tablespace Flag=0
341. Tablespace PITR mode start scn: 0x0000.00000000 01/01/88 00:00:00
342. Tablespace PITR last completion scn: 0x0000.00000000 01/01/88 00:00:00
343. TABLESPACE #6 JAYMZ: recno=7
344. First datafile link=8 Tablespace Flag=0
345. Tablespace PITR mode start scn: 0x0000.00000000 01/01/88 00:00:00
346. Tablespace PITR last completion scn: 0x0000.00000000 01/01/88 00:00:00
347. TABLESPACE #7 TEMP_TS: recno=8
348. First datafile link=1 Tablespace Flag=1
349. Tablespace PITR mode start scn: 0x0000.00000000 01/01/88 00:00:00
350. Tablespace PITR last completion scn: 0x0000.00000000 01/01/88 00:00:00
351.
352. ***<< Tablespace record section stops, log file history record section starts>>***
353.
354. LOG FILE HISTORY RECORDS:
355. (blkno = 0xa4, size = 36, max = 3204, in-use = 355, last-recid= 355)
356. Earliest record:
357. RECID #1 **Recno 1** Record timestamp 09/18/99 11:49:20 **Thread=1 Seq#=1 Link-Recid=0**
358. Low scn: 0x0000.00000001 09/18/99 11:48:46 Next scn: 0x0000.000000d1
359. Latest record:
360. RECID #355 Recno 355 Record timestamp 12/14/99 17:22:10 Thread=2 Seq#=7 Link-Recid=354
361. Low scn: 0x0000.0004af26 12/01/99 18:04:11 Next scn: 0x0000.0006a481
362. RECID #354 Recno 354 Record timestamp 12/01/99 18:04:11 Thread=2 Seq#=6 Link-Recid=352
363. Low scn: 0x0000.00040f61 11/05/99 07:25:54 Next scn: 0x0000.0004af26
364. RECID #353 Recno 353 Record timestamp 11/13/99 13:58:56 Thread=1 Seq#=348 Link-Recid=347
365. Low scn: 0x0000.000088c4 09/18/99 12:17:30 Next scn: 0x0000.00040f75
366. RECID #352 Recno 352 Record timestamp 11/05/99 07:25:54 Thread=2 Seq#=5 Link-Recid=351
367. Low scn: 0x0000.0003731e 11/04/99 16:14:42 Next scn: 0x0000.00040f61
368. RECID #351 Recno 351 Record timestamp 11/04/99 16:14:43 Thread=2 Seq#=4 Link-Recid=350

```
369.  Low scn: 0x0000.000324ed 11/04/99 16:13:19 Next scn: 0x0000.0003731e
370. RECID #350 Recno 350 Record timestamp  11/04/99 16:13:19 Thread=2 Seq#=3 Link-Recid=349
371.  Low scn: 0x0000.0002d6bc 11/04/99 16:08:13 Next scn: 0x0000.000324ed
372. RECID #349 Recno 349 Record timestamp  11/04/99 16:08:13 Thread=2 Seq#=2 Link-Recid=348
373.  Low scn: 0x0000.000287fe 11/04/99 15:30:02 Next scn: 0x0000.0002d6bc
374. RECID #348 Recno 348 Record timestamp  11/04/99 15:30:02 Thread=2 Seq#=1 Link-Recid=0
375.  Low scn: 0x0000.00008845 09/18/99 12:16:05 Next scn: 0x0000.000287fe
376. RECID #347 Recno 347 Record timestamp  09/18/99 12:17:30 Thread=1 Seq#=347 Link-Recid=346
377.  Low scn: 0x0000.000087e4 09/18/99 12:15:19 Next scn: 0x0000.000088c4
378. RECID #346 Recno 346 Record timestamp  09/18/99 12:15:19 Thread=1 Seq#=346 Link-Recid=345
379.  Low scn: 0x0000.000087ad 09/18/99 12:15:16 Next scn: 0x0000.000087e4
380.
381.  [ records deleted for readability ]
382.
383.  << Log history record section stops, offline range record section starts>>
384.
385.          OFFLINE RANGE RECORDS:
386.           (blkno = 0xdd, size = 56, max = 289, in-use = 0, last-recid= 0)
387.
388.  <<Offline range record section stops, archived log record section starts>>
389.
390.          ARCHIVED LOG RECORDS: (records added for demonstration only)
391.           (blkno = 0xe5, size = 584, max = 3202, in-use = 0, last-recid= 0)
392.  Earliest record:
393.   RECID #54 Recno 8 Record timestamp 10/21/99 15:27:42 Thread=1 Seq#=128
394.    Flags:
395.    Resetlogs scn and time scn: 0x0000.00000001 09/18/99 13:03:52
396.    filename /dev/usupport_vg/rV815/arch/arch_128.arc
397.    Low scn: 0x0000.0004309f 10/21/99 11:38:02 Next scn: 0x0000.00043be8 10/21/99 15:27:41
398.    Block count=808 Blocksize=1024
399.  Latest record:
400.  RECID #80 Recno 7 Record timestamp 12/28/99 14:17:13 Thread=1 Seq#=154
401.    Flags:
402.    Resetlogs scn and time scn: 0x0000.00000001 09/18/99 13:03:52
403.    filename /dev/usupport_vg/rV815/arch/arch_154.arc
404.    Low scn: 0x0000.00048d2c 12/28/99 14:12:02 Next scn: 0x0000.00049185 12/28/99 14:17:13
405.    Block count=14 Blocksize=1024
406.
407.  <<Archived log record section stops, RMAN information starts >>
408.
409.          BACKUP SET RECORDS:
410.           (blkno = 0x481, size = 40, max = 303, in-use = 0, last-recid= 0)
411.          BACKUP PIECE RECORDS:
412.           (blkno = 0x487, size = 736, max = 508, in-use = 0, last-recid= 0)
413.          BACKUP DATAFILE RECORDS:
414.           (blkno = 0x540, size = 116, max = 523, in-use = 0, last-recid= 0)
415.          BACKUP LOG RECORDS:
416.           (blkno = 0x55e, size = 76, max = 133, in-use = 0, last-recid= 0)
417.          DATAFILE COPY RECORDS:
418.           (blkno = 0x563, size = 660, max = 509, in-use = 0, last-recid= 0)
419.          BACKUP DATAFILE CORRUPTION RECORDS:
420.           (blkno = 0x609, size = 44, max = 276, in-use = 0, last-recid= 0)
421.          DATAFILE COPY CORRUPTION RECORDS:
422.           (blkno = 0x60f, size = 40, max = 303, in-use = 0, last-recid= 0)
```

```
423.            DELETION RECORDS:
424.             (blkno = 0x615, size = 20, max = 3238, in-use = 0, last-recid= 0)
425.            PROXY COPY RECORDS:
426.             (blkno = 0x635, size = 852, max = 636, in-use = 0, last-recid= 0)
427.*** END OF DUMP ***
```

Data File Headers Dump

```
Dump file /u02/home/usupport/udump/ora_12408.trc
Oracle8i Enterprise Edition Release 8.1.5.0.0 - Production
With the Partitioning, Parallel Server and Java options
PL/SQL Release 8.1.5.0.0 - Production
ORACLE_HOME = /u02/app/oracle/product/8.1.5
System name:        HP-UX
Node name:  ops1
Release:    B.11.00
Version:    A
Machine:    9000/889
Instance name: V815A
Redo thread mounted by this instance: 1
Oracle process number: 10
Unix process pid: 12408, image: oracle@ops1 (TNS V1-V3)

*** 2000.01.03.17.30.39.783
*** SESSION ID:(9.21548) 2000.01.03.17.30.39.766
DUMP OF DATA FILES: 8 files in database

<< Control file information starts >>

DATA FILE #1:
  (name #5) /dev/usupport_vg/rV815_system_01.dbf
  creation size=89600 block size=2048 status=0xe head=5 tail=5 dup=1
   tablespace 0, index=1 krfil=1 prev_file=0
     unrecoverable scn: 0x0000.0000885a 09/18/99 12:16:21
     Checkpoint cnt:378 scn: 0x0000.000460c2 12/01/99 16:17:51
     Stop scn: 0xffff.ffffffff 12/01/99 15:50:40
     Creation Checkpointed at scn:  0x0000.00000004 09/18/99 11:49:17
     thread:1 rba:(0x1.3.10)
     enabled  threads:  01000000 00000000 00000000 00000000 00000000 00000000
        00000000
     Offline scn: 0x0000.00000000 prev_range: 0
     Online Checkpointed at scn:  0x0000.00000000
     thread:0 rba:(0x0.0.0)
     enabled  threads:  00000000 00000000 00000000 00000000 00000000 00000000
        00000000
        Hot Backup end marker scn: 0x0000.00000000

<< Control file information ends, Generic file header starts >>

     FILE HEADER:
                Software vsn=135266304=0x8100000, Compatibility Vsn=134217728=0x8000000
                Db Id=3453975259=0xcddf7adb, Db Name='V815'
                Control Seq=1157=0x485, File size=89600=0x15e00
                File Number=1, Blksiz=2048, File Type=3 DATA

<< Generic file header ends, Data file header information starts >>

     Tablespace #0 - SYSTEM  rel_fn:1
     Creation   at   scn: 0x0000.00000004 09/18/99 11:49:17
```

```
478.  Backup taken at scn: 0x0000.00000000 01/01/88 00:00:00 thread:0
479.   reset logs count:0x1670bd9b scn: 0x0000.00000001 recovered at 11/05/99 07:25:53
480.   status:0x4 root dba:0x00400232 chkpt cnt: 378 ctl cnt:377
481.  begin-hot-backup file size: 0
482.  Checkpointed at scn:  0x0000.000460c2 12/01/99 16:17:51
483.   thread:1 rba:(0x15d.33.10)
484.   enabled  threads:  01100000 00000000 00000000 00000000 00000000 00000000
485.    00000000
486.  Backup Checkpointed at scn:  0x0000.00000000
487.   thread:0 rba:(0x0.0.0)
488.   enabled  threads:  00000000 00000000 00000000 00000000 00000000 00000000
489.    00000000
490.  External cache id: 0x0 0x0 0x0 0x0
491.  Absolute fuzzy scn: 0x0000.00000000
492.  Recovery fuzzy scn: 0x0000.00000000 01/01/88 00:00:00
493.
494.  << Data file header information ends, control file information starts >>
495.
496.  DATA FILE #2:
497.    (name #6) /dev/usupport_vg/rV815_query_01.dbf
498.  creation size=2560 block size=2048 status=0x86 head=6 tail=6 dup=1
499.   tablespace 1, index=2 krfil=2 prev_file=0
500.   unrecoverable scn: 0x0000.00000000 01/01/88 00:00:00
501.   Checkpoint cnt:290 scn: 0x0000.000460c2 12/01/99 16:17:51
502.   Stop scn: 0x0000.000460c2 12/01/99 16:17:51
503.   Creation Checkpointed at scn:  0x0000.000050a9 09/18/99 11:58:14
504.   thread:1 rba:(0x59.1e5.b0)
505.   enabled  threads:  01000000 00000000 00000000 00000000 00000000 00000000
506.    00000000
507.   Offline scn: 0x0000.00000000 prev_range: 0
508.   Online Checkpointed at scn:  0x0000.00000000
509.   thread:0 rba:(0x0.0.0)
510.   enabled  threads:  00000000 00000000 00000000 00000000 00000000 00000000
511.    00000000
512.   Hot Backup end marker scn: 0x0000.00000000
513.
514.   << Control file information ends, Generic file header starts >>
515.
516.   FILE HEADER:
517.           Software vsn=135266304=0x8100000, Compatibility Vsn=134217728=0x8000000
518.           Db Id=3453975259=0xcddf7adb, Db Name='V815'
519.           Control Seq=1161=0x489, File size=2560=0xa00
520.           File Number=2, Blksiz=2048, File Type=3 DATA
521.
522.    << Generic file header ends, data file header information starts >>
523.
524.  Tablespace #1 - QUERY_DATA  rel_fn:2
525.  Creation   at   scn: 0x0000.000050a9 09/18/99 11:58:14
526.  Backup taken at scn: 0x0000.00000000 01/01/88 00:00:00 thread:0
527.   reset logs count:0x1670bd9b scn: 0x0000.00000001 recovered at 11/05/99 07:25:53
528.   status:0x0 root dba:0x00000000 chkpt cnt: 290 ctl cnt:289
529.  begin-hot-backup file size: 0
530.  Checkpointed at scn:  0x0000.000460c2 12/01/99 16:17:51
531.   thread:1 rba:(0x15d.33.10)
```

```
532.    enabled  threads:  01100000 00000000 00000000 00000000 00000000 00000000
533.     00000000
534.   Backup Checkpointed at scn:  0x0000.00000000
535.    thread:0 rba:(0x0.0.0)
536.    enabled  threads:  00000000 00000000 00000000 00000000 00000000 00000000
537.     00000000
538.   External cache id: 0x0 0x0 0x0 0x0
539.   Absolute fuzzy scn: 0x0000.00000000
540.   Recovery fuzzy scn: 0x0000.00000000 01/01/88 00:00:00
541.   DATA FILE #3:
542.     (name #7) /dev/usupport_vg/rV815_rbs_01.dbf
543.   creation size=6144 block size=2048 status=0xe head=7 tail=7 dup=1
544.    tablespace 2, index=3 krfil=3 prev_file=0
545.    unrecoverable scn: 0x0000.00000000 01/01/88 00:00:00
546.    Checkpoint cnt:290 scn: 0x0000.000460c2 12/01/99 16:17:51
547.    Stop scn: 0xffff.ffffffff 12/01/99 15:50:40
548.    Creation Checkpointed at scn:  0x0000.000050b2 09/18/99 11:58:16
549.    thread:1 rba:(0x59.1ed.a8)
550.    enabled  threads:  01000000 00000000 00000000 00000000 00000000 00000000
551.     00000000
552.    Offline scn: 0x0000.00000000 prev_range: 0
553.    Online Checkpointed at scn:  0x0000.00000000
554.    thread:0 rba:(0x0.0.0)
555.    enabled  threads:  00000000 00000000 00000000 00000000 00000000 00000000
556.     00000000
557.    Hot Backup end marker scn: 0x0000.00000000
558.    FILE HEADER:
559.            Software vsn=135266304=0x8100000, Compatibility Vsn=134217728=0x8000000
560.            Db Id=3453975259=0xcddf7adb, Db Name='V815'
561.            Control Seq=1157=0x485, File size=6144=0x1800
562.            File Number=3, Blksiz=2048, File Type=3 DATA
563.   Tablespace #2 - RBS  rel_fn:3
564.   Creation   at   scn: 0x0000.000050b2 09/18/99 11:58:16
565.   Backup taken at scn: 0x0000.00000000 01/01/88 00:00:00 thread:0
566.    reset logs count:0x1670bd9b scn: 0x0000.00000001 recovered at 11/05/99 07:25:53
567.    status:0x4 root dba:0x00000000 chkpt cnt: 290 ctl cnt:289
568.   begin-hot-backup file size: 0
569.   Checkpointed at scn:  0x0000.000460c2 12/01/99 16:17:51
570.    thread:1 rba:(0x15d.33.10)
571.    enabled  threads:  01100000 00000000 00000000 00000000 00000000 00000000
572.     00000000
573.   Backup Checkpointed at scn:  0x0000.00000000
574.    thread:0 rba:(0x0.0.0)
575.    enabled  threads:  00000000 00000000 00000000 00000000 00000000 00000000
576.     00000000
577.   External cache id: 0x0 0x0 0x0 0x0
578.   Absolute fuzzy scn: 0x0000.00000000
579.   Recovery fuzzy scn: 0x0000.00000000 01/01/88 00:00:00
580.   DATA FILE #4:
581.     (name #8) /dev/usupport_vg/rV815_temp_01.dbf
582.   creation size=5120 block size=2048 status=0xe head=8 tail=8 dup=1
583.    tablespace 3, index=4 krfil=4 prev_file=0
584.    unrecoverable scn: 0x0000.00000000 01/01/88 00:00:00
585.    Checkpoint cnt:289 scn: 0x0000.000460c2 12/01/99 16:17:51
```

```
586.    Stop scn: 0xffff.ffffffff 12/01/99 15:50:40
587.    Creation Checkpointed at scn:  0x0000.000050bf 09/18/99 11:58:19
588.    thread:1 rba:(0x5a.4.a8)
589.    enabled  threads:  01000000 00000000 00000000 00000000 00000000 00000000
590.     00000000
591.    Offline scn: 0x0000.00000000 prev_range: 0
592.    Online Checkpointed at scn:  0x0000.00000000
593.    thread:0 rba:(0x0.0.0)
594.    enabled  threads:  00000000 00000000 00000000 00000000 00000000 00000000
595.     00000000
596.    Hot Backup end marker scn: 0x0000.00000000
597.    FILE HEADER:
598.            Software vsn=135266304=0x8100000, Compatibility Vsn=134217728=0x8000000
599.            Db Id=3453975259=0xcddf7adb, Db Name='V815'
600.            Control Seq=1157=0x485, File size=5120=0x1400
601.            File Number=4, Blksiz=2048, File Type=3 DATA
602.   Tablespace #3 - TEMP  rel_fn:4
603.   Creation   at   scn: 0x0000.000050bf 09/18/99 11:58:19
604.   Backup taken at scn: 0x0000.00000000 01/01/88 00:00:00 thread:0
605.    reset logs count:0x1670bd9b scn: 0x0000.00000001 recovered at 11/05/99 07:25:53
606.    status:0x4 root dba:0x00000000 chkpt cnt: 289 ctl cnt:288
607.   begin-hot-backup file size: 0
608.   Checkpointed at scn:  0x0000.000460c2 12/01/99 16:17:51
609.    thread:1 rba:(0x15d.33.10)
610.    enabled  threads:  01100000 00000000 00000000 00000000 00000000 00000000
611.     00000000
612.   Backup Checkpointed at scn:  0x0000.00000000
613.    thread:0 rba:(0x0.0.0)
614.    enabled  threads:  00000000 00000000 00000000 00000000 00000000 00000000
615.     00000000
616.   External cache id: 0x0 0x0 0x0 0x0
617.   Absolute fuzzy scn: 0x0000.00000000
618.   Recovery fuzzy scn: 0x0000.00000000 01/01/88 00:00:00
619.   DATA FILE #5:
620.     (name #9) /dev/usupport_vg/rV815_users_01.dbf
621.   creation size=5120 block size=2048 status=0xe head=9 tail=9 dup=1
622.    tablespace 4, index=5 krfil=5 prev_file=0
623.    unrecoverable scn: 0x0000.00000000 01/01/88 00:00:00
624.    Checkpoint cnt:289 scn: 0x0000.000460c2 12/01/99 16:17:51
625.    Stop scn: 0xffff.ffffffff 12/01/99 15:50:40
626.    Creation Checkpointed at scn:  0x0000.000050cb 09/18/99 11:58:21
627.    thread:1 rba:(0x5a.f.a8)
628.    enabled  threads:  01000000 00000000 00000000 00000000 00000000 00000000
629.     00000000
630.    Offline scn: 0x0000.00000000 prev_range: 0
631.    Online Checkpointed at scn:  0x0000.00000000
632.    thread:0 rba:(0x0.0.0)
633.    enabled  threads:  00000000 00000000 00000000 00000000 00000000 00000000
634.     00000000
635.    Hot Backup end marker scn: 0x0000.00000000
636.    FILE HEADER:
637.            Software vsn=135266304=0x8100000, Compatibility Vsn=134217728=0x8000000
638.            Db Id=3453975259=0xcddf7adb, Db Name='V815'
639.            Control Seq=1157=0x485, File size=5120=0x1400
```

```
640.            File Number=5, Blksiz=2048, File Type=3 DATA
641.    Tablespace #4 - USERS  rel_fn:5
642.    Creation   at   scn: 0x0000.000050cb 09/18/99 11:58:21
643.    Backup taken at scn: 0x0000.00000000 01/01/88 00:00:00 thread:0
644.     reset logs count:0x1670bd9b scn: 0x0000.00000001 recovered at 11/05/99 07:25:53
645.     status:0x4 root dba:0x00000000 chkpt cnt: 289 ctl cnt:288
646.    begin-hot-backup file size: 0
647.    Checkpointed at scn:  0x0000.000460c2 12/01/99 16:17:51
648.     thread:1 rba:(0x15d.33.10)
649.     enabled  threads:  01100000 00000000 00000000 00000000 00000000 00000000
650.      00000000
651.    Backup Checkpointed at scn:  0x0000.00000000
652.     thread:0 rba:(0x0.0.0)
653.     enabled  threads:  00000000 00000000 00000000 00000000 00000000 00000000
654.      00000000
655.    External cache id: 0x0 0x0 0x0 0x0
656.    Absolute fuzzy scn: 0x0000.00000000
657.    Recovery fuzzy scn: 0x0000.00000000 01/01/88 00:00:00
658.    DATA FILE #6:
659.      (name #10) /dev/usupport_vg/rV815_index_01.dbf
660.    creation size=5120 block size=2048 status=0xe head=10 tail=10 dup=1
661.     tablespace 5, index=6 krfil=6 prev_file=0
662.     unrecoverable scn: 0x0000.00000000 01/01/88 00:00:00
663.     Checkpoint cnt:289 scn: 0x0000.000460c2 12/01/99 16:17:51
664.     Stop scn: 0xffff.ffffffff 12/01/99 15:50:40
665.     Creation Checkpointed at scn:  0x0000.000050d6 09/18/99 11:58:24
666.     thread:1 rba:(0x5a.17.a8)
667.     enabled  threads:  01000000 00000000 00000000 00000000 00000000 00000000
668.      00000000
669.     Offline scn: 0x0000.00000000 prev_range: 0
670.     Online Checkpointed at scn:  0x0000.00000000
671.     thread:0 rba:(0x0.0.0)
672.     enabled  threads:  00000000 00000000 00000000 00000000 00000000 00000000
673.      00000000
674.     Hot Backup end marker scn: 0x0000.00000000
675.     FILE HEADER:
676.            Software vsn=135266304=0x8100000, Compatibility Vsn=134217728=0x8000000
677.            Db Id=3453975259=0xcddf7adb, Db Name='V815'
678.            Control Seq=1157=0x485, File size=5120=0x1400
679.            File Number=6, Blksiz=2048, File Type=3 DATA
680.    Tablespace #5 - INDX  rel_fn:6
681.    Creation   at   scn: 0x0000.000050d6 09/18/99 11:58:24
682.    Backup taken at scn: 0x0000.00000000 01/01/88 00:00:00 thread:0
683.     reset logs count:0x1670bd9b scn: 0x0000.00000001 recovered at 11/05/99 07:25:53
684.     status:0x4 root dba:0x00000000 chkpt cnt: 289 ctl cnt:288
685.    begin-hot-backup file size: 0
686.    Checkpointed at scn:  0x0000.000460c2 12/01/99 16:17:51
687.     thread:1 rba:(0x15d.33.10)
688.     enabled  threads:  01100000 00000000 00000000 00000000 00000000 00000000
689.      00000000
690.    Backup Checkpointed at scn:  0x0000.00000000
691.     thread:0 rba:(0x0.0.0)
692.     enabled  threads:  00000000 00000000 00000000 00000000 00000000 00000000
693.      00000000
```

```
694.  External cache id: 0x0 0x0 0x0 0x0
695.  Absolute fuzzy scn: 0x0000.00000000
696.  Recovery fuzzy scn: 0x0000.00000000 01/01/88 00:00:00
697.  DATA FILE #7:
698.    (name #15) /dev/usupport_vg/rjaymz.dbf
699.  creation size=5632 block size=2048 status=0xe head=15 tail=15 dup=1
700.   tablespace 6, index=7 krfil=7 prev_file=0
701.   unrecoverable scn: 0x0000.00000000 01/01/88 00:00:00
702.   Checkpoint cnt:4 scn: 0x0000.000460c2 12/01/99 16:17:51
703.   Stop scn: 0xffff.ffffffff 12/01/99 15:50:40
704.   Creation Checkpointed at scn:  0x0000.0004607b 12/01/99 15:46:48
705.   thread:1 rba:(0x15d.1e.a8)
706.   enabled  threads:  01100000 00000000 00000000 00000000 00000000 00000000
707.    00000000
708.   Offline scn: 0x0000.00000000 prev_range: 0
709.   Online Checkpointed at scn:  0x0000.00000000
710.   thread:0 rba:(0x0.0.0)
711.   enabled  threads:  00000000 00000000 00000000 00000000 00000000 00000000
712.    00000000
713.   Hot Backup end marker scn: 0x0000.00000000
714.   FILE HEADER:
715.           Software vsn=135266304=0x8100000, Compatibility Vsn=134217728=0x8000000
716.           Db Id=3453975259=0xcddf7adb, Db Name='V815'
717.           Control Seq=1157=0x485, File size=5632=0x1600
718.           File Number=7, Blksiz=2048, File Type=3 DATA
719.  Tablespace #6 - JAYMZ  rel_fn:7
720.  Creation   at   scn: 0x0000.0004607b 12/01/99 15:46:48
721.  Backup taken at scn: 0x0000.00000000 01/01/88 00:00:00 thread:0
722.   reset logs count:0x1670bd9b scn: 0x0000.00000001 recovered at 01/01/88 00:00:00
723.   status:0x4 root dba:0x00000000 chkpt cnt: 4 ctl cnt:3
724.  begin-hot-backup file size: 0
725.  Checkpointed at scn:  0x0000.000460c2 12/01/99 16:17:51
726.   thread:1 rba:(0x15d.33.10)
727.   enabled  threads:  01100000 00000000 00000000 00000000 00000000 00000000
728.    00000000
729.  Backup Checkpointed at scn:  0x0000.00000000
730.   thread:0 rba:(0x0.0.0)
731.   enabled  threads:  00000000 00000000 00000000 00000000 00000000 00000000
732.    00000000
733.  External cache id: 0x0 0x0 0x0 0x0
734.  Absolute fuzzy scn: 0x0000.00000000
735.  Recovery fuzzy scn: 0x0000.00000000 01/01/88 00:00:00
736.  DATA FILE #8:
737.    (name #16) /dev/jaymz_vg/rjaymz2.dbf
738.  creation size=3584 block size=2048 status=0xe head=16 tail=16 dup=1
739.   tablespace 6, index=7 krfil=8 prev_file=7
740.   unrecoverable scn: 0x0000.00000000 01/01/88 00:00:00
741.   Checkpoint cnt:2 scn: 0x0000.000460fb 12/01/99 17:06:46
742.   Stop scn: 0xffff.ffffffff 12/01/99 17:06:46
743.   Creation Checkpointed at scn:  0x0000.000460fa 12/01/99 17:06:46
744.   thread:2 rba:(0x6.21.158)
745.   enabled  threads:  01100000 00000000 00000000 00000000 00000000 00000000
746.    00000000
747.   Offline scn: 0x0000.00000000 prev_range: 0
```

```
748.    Online Checkpointed at scn:  0x0000.00000000
749.    thread:0 rba:(0x0.0.0)
750.    enabled  threads:  00000000 00000000 00000000 00000000 00000000 00000000
751.     00000000
752.    Hot Backup end marker scn: 0x0000.00000000
753.    FILE HEADER:
754.            Software vsn=135266304=0x8100000, Compatibility Vsn=134217728=0x8000000
755.            Db Id=3453975259=0xcddf7adb, Db Name='V815'
756.            Control Seq=1160=0x488, File size=3584=0xe00
757.            File Number=8, Blksiz=2048, File Type=3 DATA
758.   Tablespace #6 - JAYMZ  rel_fn:8
759.   Creation   at   scn: 0x0000.000460fa 12/01/99 17:06:46
760.   Backup taken at scn: 0x0000.00000000 01/01/88 00:00:00 thread:0
761.    reset logs count:0x1670bd9b scn: 0x0000.00000001 recovered at 01/01/88 00:00:00
762.    status:0x4 root dba:0x00000000 chkpt cnt: 2 ctl cnt:1
763.   begin-hot-backup file size: 0
764.   Checkpointed at scn:  0x0000.000460fb 12/01/99 17:06:46
765.    thread:2 rba:(0x6.21.158)
766.    enabled  threads:  01100000 00000000 00000000 00000000 00000000 00000000
767.     00000000
768.   Backup Checkpointed at scn:  0x0000.00000000
769.    thread:0 rba:(0x0.0.0)
770.    enabled  threads:  00000000 00000000 00000000 00000000 00000000 00000000
771.     00000000
772.   External cache id: 0x0 0x0 0x0 0x0
773.   Absolute fuzzy scn: 0x0000.00000000
774.   Recovery fuzzy scn: 0x0000.00000000 01/01/88 00:00:00
775.   DATA FILE #9: <deleted>
776.    creation size=3584 block size=2048 status=0x810 head=0 tail=0 dup=1
777.    tablespace 7, index=8 krfil=9 prev_file=0
778.    unrecoverable scn: 0x0000.00000000 01/01/88 00:00:00
779.    Checkpoint cnt:2 scn: 0x0000.000460fb 12/01/99 17:06:46
780.    Stop scn: 0x0000.0004783e 12/21/99 12:06:46
781.    Creation Checkpointed at scn:  0x0000.000460fa 12/01/99 17:06:46
782.    thread:2 rba:(0x6.21.158)
783.    enabled  threads:  01100000 00000000 00000000 00000000 00000000 00000000
784.     00000000
785.    Offline scn: 0x0000.00000000 prev_range: 0
786.    Online Checkpointed at scn:  0x0000.00000000
787.    thread:0 rba:(0x0.0.0)
788.    enabled  threads:  00000000 00000000 00000000 00000000 00000000 00000000
789.     00000000
790.    Hot Backup end marker scn: 0x0000.00000000
791.   DUMP OF TEMP FILES: 1 files in database
792.
793.   << Control file information starts >>
794.
795.   TEMP FILE #1: External File #101
796.     (name #18) /dev/usupport_vg/rV815/tempfile1.dbf
797.   creation size=2560 block size=2048 status=0xe head=18 tail=18 dup=1
798.    tablespace 9, index=8 krfil=1 prev_file=0
799.    unrecoverable scn: 0x0000.00000000 01/01/88 00:00:00
800.
801.   << Control file information ends, Generic file header information starts >>
```

```
802.
803.  FILE HEADER:
804.          Software vsn=135266304=0x8100000, Compatibility Vsn=134217728=0x8000000
805.          Db Id=3453975259=0xcddf7adb, Db Name='V815'
806.          Control Seq=739=0x2e3, File size=2560=0xa00
807.          File Number=1, Blksiz=2048, File Type=6 DATA
808.
809.  << Generic file header information ends, data file header information starts >>
810.
811.  Tablespace #7 - TEMP_TS  rel_fn:1
812.  Creation   at   scn: 0x0000.00000000 01/01/88 00:00:00
813.  Backup taken at scn: 0x0000.00000000 01/01/88 00:00:00 thread:0
814.   reset logs count:0x0 scn: 0x0000.00000000 recovered at 01/01/88 00:00:00
815.   status:0x0 root dba:0x00000000 chkpt cnt: 0 ctl cnt:0
816.  begin-hot-backup file size: 0
817.  Checkpointed at scn:  0x0000.00000000
818.   thread:0 rba:(0x0.0.0)
819.   enabled  threads:  00000000 00000000 00000000 00000000 00000000 00000000
820.    00000000 00000000
821.  Backup Checkpointed at scn:  0x0000.00000000
822.   thread:0 rba:(0x0.0.0)
823.   enabled  threads:  00000000 00000000 00000000 00000000 00000000 00000000
824.    00000000 00000000
825.  External cache id: 0x0 0x0 0x0 0x0
826.  Absolute fuzzy scn: 0x0000.00000000
827.  Recovery fuzzy scn: 0x0000.00000000 01/01/88 00:00:00
```

Redo Log Headers Dump

```
801.Dump file /u02/home/usupport/udump/ora_12426.trc
802.Oracle8i Enterprise Edition Release 8.1.5.0.0 - Production
803.With the Partitioning, Parallel Server and Java options
804.PL/SQL Release 8.1.5.0.0 - Production
805.ORACLE_HOME = /u02/app/oracle/product/8.1.5
806.System name:        HP-UX
807.Node name:  ops1
808.Release:    B.11.00
809.Version:    A
810.Machine:    9000/889
811.Instance name: V815A
812.Redo thread mounted by this instance: 1
813.Oracle process number: 10
814.Unix process pid: 12426, image: oracle@ops1 (TNS V1-V3)

815.*** 2000.01.03.17.32.18.766
816.*** SESSION ID:(9.21550) 2000.01.03.17.32.18.748
817.DUMP OF LOG FILES: 4 logs in database
818.
819.<< Control file information starts >>
820.
821.LOG FILE #1:
822.  (name #1) /dev/usupport_vg/rV815_log1_01a.log
823.  (name #2) /dev/usupport_vg/rV815_log1_01b.log
824. Thread 1 redo log links: forward: 2 backward: 0
825. siz: 0x1f4 seq: 0x0000015d hws: 0x8 bsz: 1024 nab: 0xffffffff flg: 0x8 dup: 2
826. Archive links: fwrd: 0 back: 0 Prev scn: 0x0000.000088c4
827. Low scn: 0x0000.00040f75 11/13/99 13:58:56
828. Next scn: 0xffff.ffffffff 09/18/99 12:17:30
829.
830. << Control file information ends, Generic file header information starts >>
831.
832. FILE HEADER:
833.        Software vsn=135266304=0x8100000, Compatibility Vsn=135266304=0x8100000
834.        Db Id=3453975259=0xcddf7adb, Db Name='V815'
835.        Control Seq=1157=0x485, File size=500=0x1f4
836.        File Number=1, Blksiz=1024, File Type=2 LOG
837.
838.  << Generic file header information ends, log file header information starts >>
839.
840. descrip:"Thread 0001, Seq# 0000000349, SCN 0x000000040f75-0xffffffffffff"
841. thread: 1 nab: 0xffffffff seq: 0x0000015d hws: 0x8 eot: 1 dis: 0
842. reset logs count: 0x1670bd9b scn: 0x0000.00000001
843. Low scn: 0x0000.00040f75 11/13/99 13:58:56
844. Next scn: 0xffff.ffffffff 09/18/99 12:17:30
845. Enabled scn: 0x0000.00000001 09/18/99 11:48:46
846. Thread closed scn: 0x0000.000460c1 12/01/99 15:50:40
847. Log format vsn: 0x8000000 Disk cksum: 0xed88 Calc cksum: 0xed88
848.
849. << Log file information ends, control file information starts >>
```

```
850.
851.LOG FILE #2:
852.  (name #3) /dev/usupport_vg/rV815_log1_02a.log
853.  (name #4) /dev/usupport_vg/rV815_log1_02b.log
854. Thread 1 redo log links: forward: 0 backward: 1
855. siz: 0x1f4 seq: 0x0000015c hws: 0x1b bsz: 1024 nab: 0xfa flg: 0x0 dup: 2
856. Archive links: fwrd: 0 back: 0 Prev scn: 0x0000.000087e4
857. Low scn: 0x0000.000088c4 09/18/99 12:17:30
858. Next scn: 0x0000.00040f75 11/13/99 13:58:56
859. FILE HEADER:
860.          Software vsn=135266304=0x8100000, Compatibility Vsn=135266304=0x8100000
861.          Db Id=3453975259=0xcddf7adb, Db Name='V815'
862.          Control Seq=1157=0x485, File size=500=0x1f4
863.          File Number=2, Blksiz=1024, File Type=2 LOG
864. descrip:"Thread 0001, Seq# 0000000348, SCN 0x0000000088c4-0x000000040f75"
865. thread: 1 nab: 0xfa seq: 0x0000015c hws: 0x1b eot: 0 dis: 0
866. reset logs count: 0x1670bd9b scn: 0x0000.00000001
867. Low scn: 0x0000.000088c4 09/18/99 12:17:30
868. Next scn: 0x0000.00040f75 11/13/99 13:58:56
869. Enabled scn: 0x0000.00000001 09/18/99 11:48:46
870. Thread closed scn: 0x0000.0003c13b 11/05/99 06:35:06
871. Log format vsn: 0x8000000 Disk cksum: 0xf7b4 Calc cksum: 0xf7b4
872.LOG FILE #3:
873.  (name #11) /dev/usupport_vg/rV815_log2_01a.log
874.  (name #12) /dev/usupport_vg/rV815_log2_01b.log
875. Thread 2 redo log links: forward: 4 backward: 0
876. siz: 0x800 seq: 0x00000007 hws: 0x5 bsz: 1024 nab: 0x6 flg: 0x0 dup: 2
877. Archive links: fwrd: 0 back: 0 Prev scn: 0x0000.00040f61
878. Low scn: 0x0000.0004af26 12/01/99 18:04:11
879. Next scn: 0x0000.0006a481 12/14/99 17:22:10
880. FILE HEADER:
881.          Software vsn=135266304=0x8100000, Compatibility Vsn=135266304=0x8100000
882.          Db Id=3453975259=0xcddf7adb, Db Name='V815'
883.          Control Seq=1168=0x490, File size=2048=0x800
884.          File Number=3, Blksiz=1024, File Type=2 LOG
885. descrip:"Thread 0002, Seq# 0000000007, SCN 0x00000004af26-0x00000006a481"
886. thread: 2 nab: 0x6 seq: 0x00000007 hws: 0x5 eot: 0 dis: 0
887. reset logs count: 0x1670bd9b scn: 0x0000.00000001
888. Low scn: 0x0000.0004af26 12/01/99 18:04:11
889. Next scn: 0x0000.0006a481 12/14/99 17:22:10
890. Enabled scn: 0x0000.00008845 09/18/99 12:16:05
891. Thread closed scn: 0x0000.0006a478 12/06/99 04:17:42
892. Log format vsn: 0x8000000 Disk cksum: 0x4cb1 Calc cksum: 0x4cb1
893.LOG FILE #4:
894.  (name #13) /dev/usupport_vg/rV815_log2_02a.log
895.  (name #14) /dev/usupport_vg/rV815_log2_02b.log
896. Thread 2 redo log links: forward: 0 backward: 3
897. siz: 0x800 seq: 0x00000008 hws: 0x2 bsz: 1024 nab: 0xffffffff flg: 0x8 dup: 2
898. Archive links: fwrd: 0 back: 0 Prev scn: 0x0000.0004af26
899. Low scn: 0x0000.0006a481 12/14/99 17:22:10
900. Next scn: 0xffff.ffffffff 12/01/99 18:04:11
901. FILE HEADER:
902.          Software vsn=135266304=0x8100000, Compatibility Vsn=135266304=0x8100000
903.          Db Id=3453975259=0xcddf7adb, Db Name='V815'
```

```
Control Seq=1169=0x491, File size=2048=0x800
File Number=4, Blksiz=1024, File Type=2 LOG
descrip:"Thread 0002, Seq# 0000000008, SCN 0x00000006a481-0xffffffffffff"
thread: 2 nab: 0xffffffff seq: 0x00000008 hws: 0x2 eot: 1 dis: 0
reset logs count: 0x1670bd9b scn: 0x0000.00000001
Low scn: 0x0000.0006a481 12/14/99 17:22:10
Next scn: 0xffff.ffffffff 12/01/99 18:04:11
Enabled scn: 0x0000.00008845 09/18/99 12:16:05
Thread closed scn: 0x0000.0006a481 12/14/99 17:22:10
Log format vsn: 0x8000000 Disk cksum: 0xc72f Calc cksum: 0xc72f
```

B
Practices

Lesson 2 Practice: Viewing Redo Records

In this practice we will issue some SQL operations against a table in the database. Then we will dump the redo logs to view the redo records generated for this DML.

1. Connect as SCOTT/TIGER and query the BONUS table.
2. Perform an ALTER SYSTEM SWITCH LOGFILE.
3. Insert a row into the BONUS table.
4. Create an index on the ENAME column of the BONUS table.

   HINT: CREATE INDEX … ON BONUS(ENAME);
5. Insert another row into the BONUS table.
6. Commit.
7. Delete a row from the BONUS table.
8. Commit.
9. Delete all rows from the BONUS table.
10. Rollback.
11. Switch the logs again.
12. Find out which log is the current log and which log was previously the current log.
13. Dump the previous log.

    HINT: ALTER SYSTEM DUMP LOGFILE '<full path name'>;
14. Find the first Insert command. How many change vectors are contained within this redo record? Why are there more than one change vectors? _____

    _____
15. Locate the commit marker for the second insert.
16. How many redo records were generated for the CREATE INDEX command? _____
17. Was redo generated for the ROLLBACK? _____

Lesson 3 Practices: Recovery Diagnostics

Exercise 1 - Control File dump

Refer to the trace file entitled 'Controlfile dump' (starting on pg.B-5), which was dumped with the syntax:

SVRMGR> alter session set events 'immediate trace name controlf level N';

Q1 What level was specified in the controlf dump command?

_____

Q2 What minimum level should be specified in order to dump out datafile records?

_____

Q3 Has this database been opened with resetlogs? If so, when?

_____

Q4 What do the database flags translate to?

_____

Q5 When was the database last checkpointed?

_____

Q6 What is the controlfile's size (in bytes)?

_____

Exercise 2 - Datafile Header dump

Refer to the trace file entitled 'Datafile headers dump' (starting on pg.B-7), which was dumped with the syntax:

SVRMGR> alter session set events 'immediate trace name file_hdrs level N';

Q1 What level was used in the file_hdrs dump command?

_____

Q2 What are the statuses of the datafiles, and what does each status translate to? (You may need to reference
the controlfile status as well as the datafile status)

_____

Q3 What was the log sequence at the last checkpoint of file #1?

_____

Q4 ….and which thread performed the checkpoint?

_____

Q5 How many times has file #2 been checkpointed?

_____

Q6 What is the database name to which these files belong?

_____

Exercise 3 - Redo log header dump

Refer to the trace file entitled 'Redo log headers dump' (starting on pg.B-13), which was dumped with the syntax:

SVRMGR> alter session set events 'immediate trace name redohdr level N';

Q1 What level was used in the redohdr dump command?

_____

Q2 How many redo threads are in this database?

_____

Q3 What are the range of log sequences in each thread?

_____

Q4 What is the range of SCNs in each redo log?

_____

Q5 Identify the two checksum fields in each logfile

_____

Q6 How big is each redo log (in bytes)?

_____

Control file dump

```
Dump file /oracle/app/oracle/admin/V8041/udump/v8041_ora_12186.trc
Oracle8 Enterprise Edition Release 8.0.4.1.0 - Production
With the Partitioning, Objects and Parallel Server options
PL/SQL Release 8.0.4.1.0 - Production
ORACLE_HOME = /oracle/app/oracle/product/8.0.4
System name:      SunOS
Node name:  uksn116
Release:    5.5.1
Version:    Generic_103640-14
Machine:    sun4u
Instance name: V8041
Redo thread mounted by this instance: 1
Oracle process number: 14
Unix process pid: 12186, image: oracleV8041

Mon Jun  8 10:26:43 1998
*** SESSION ID:(14.851) 1998.06.08.10.26.43.000
DUMP OF CONTROL FILES, Seq # 1219 = 0x4c3
 FILE HEADER:
      Software vsn=134217728=0x8000000, Compatibility Vsn=134217728=0x8000000
      Db Id=3253243709=0xc1e88f3d, Db Name='V804'
      Control Seq=1219=0x4c3, File size=202=0xca
      File Number=0, Blksiz=8192, File Type=1
      DATABASE ENTRY:
       (blkno = 0x1, size = 192, max = 1, in-use = 1, last-recid= 0)
 DF Version: creation=0x8000000 compatible=0x8000000, Date  02/13/98 17:02:53
 DB Name "V804"
 Database flags = 0x00006001
 Controlfile Creation Timestamp  02/13/98 17:02:54
 Incmplt recovery scn: 0x0000.00000000
 Resetlogs scn: 0x04d8.5d142764 Resetlogs Timestamp  05/26/98 13:57:40
 Prior resetlogs scn: 0x04d8.5d142740 Prior resetlogs Timestamp  05/26/98
13:17:58
 Redo Version: creation=0x8000000 compatable=0x8000000
 #Data files = 8, #Online files = 8
 Database checkpoint: Thread=1 scn: 0x04d8.fea98ff9
 Threads: #Enabled=2, #Open=2, Head=1, Tail=2
 enabled  threads:  01100000 00000000 00000000 00000000 00000000 00000000
  00000000 00000000
 Max log members = 2, Max data members = 1
 Arch list: Head=1, Tail=4, Force scn: 0x04d8.fea98ff7
 Snapshot Controlfile filename
  (name #15) /oracle/app/oracle/product/8.0.4/dbs/snapcf_V8041.f
 Snapshot Controlfile checkpoint scn: 0x04d8.5d1562a8 06/02/98 16:50:39
 Controlfile Checkpointed at scn: 0x04d8.fea994d2 01/01/88 00:00:00
 thread:0 rba:(0x0.0.0)
 enabled  threads:  00000000 00000000 00000000 00000000 00000000 00000000
  00000000 00000000
      CHECKPOINT PROGRESS RECORDS:
       (blkno = 0x6, size = 104, max = 8, in-use = 2, last-recid= 2)
THREAD #1 - status:0x2 flags:0x0 dirty:3
low cache rba:(0xa.5.10) on disk rba:(0xa.7.0)
on disk scn: 0x04d8.fea9941f 06/07/98 11:27:39
THREAD #2 - status:0x2 flags:0x0 dirty:16
low cache rba:(0xb.ae.10) on disk rba:(0xb.d0.0)
on disk scn: 0x04d8.fea994f3 06/08/98 09:44:04
THREAD #3 - status:0x0 flags:0x0 dirty:0
low cache rba:(0x0.0.0) on disk rba:(0x0.0.0)
on disk scn: 0x0000.00000000 01/01/88 00:00:00
```

```
THREAD #4 - status:0x0 flags:0x0 dirty:0
low cache rba:(0x0.0.0) on disk rba:(0x0.0.0)
on disk scn: 0x0000.00000000 01/01/88 00:00:00
THREAD #5 - status:0x0 flags:0x0 dirty:0
low cache rba:(0x0.0.0) on disk rba:(0x0.0.0)
on disk scn: 0x0000.00000000 01/01/88 00:00:00
THREAD #6 - status:0x0 flags:0x0 dirty:0
low cache rba:(0x0.0.0) on disk rba:(0x0.0.0)
on disk scn: 0x0000.00000000 01/01/88 00:00:00
THREAD #7 - status:0x0 flags:0x0 dirty:0
low cache rba:(0x0.0.0) on disk rba:(0x0.0.0)
on disk scn: 0x0000.00000000 01/01/88 00:00:00
THREAD #8 - status:0x0 flags:0x0 dirty:0
low cache rba:(0x0.0.0) on disk rba:(0x0.0.0)
on disk scn: 0x0000.00000000 01/01/88 00:00:00
*** END OF DUMP ***
```

Datafile headers dump

```
Dump file /oracle/app/oracle/admin/V8041/udump/v8041_ora_20715.trc
Oracle8 Enterprise Edition Release 8.0.4.1.0 - Production
With the Partitioning, Objects and Parallel Server options
PL/SQL Release 8.0.4.1.0 - Production
ORACLE_HOME = /oracle/app/oracle/product/8.0.4
System name:      SunOS
Node name:  uksn116
Release:    5.5.1
Version:    Generic_103640-14
Machine:    sun4u
Instance name: V8041
Redo thread mounted by this instance: 1
Oracle process number: 14
Unix process pid: 20715, image: oracle

Wed Jun  3 15:37:24 1998
*** SESSION ID:(10.1) 1998.06.03.15.37.24.000
DUMP OF DATA FILES: 8 files in database
DATA FILE #1:
  (name #4) /dev/vx/rdsk/datadg/v804_sys
creation size=6398 block size=8192 status=0xe head=4 tail=4 dup=1
 tablespace 0, index=1 krfil=1 prev_file=0
 unrecoverable scn: 0x0000.00000000 04/15/98 14:24:16
 Checkpoint cnt:198 scn: 0x04d8.5d15b0e8 06/03/98 15:36:24
 Stop scn: 0xffff.ffffffff 05/28/98 18:02:11
 Creation Checkpointed at scn: 0x0000.00000003 02/13/98 17:03:05
 thread:1 rba:(0x1.3.10)
 enabled  threads:  01000000 00000000 00000000 00000000 00000000 00000000
  00000000 00000000
 Offline scn: 0x04d8.5d142763 prev_range: 0
 Online Checkpointed at scn: 0x04d8.5d142764 05/26/98 13:57:40
 thread:1 rba:(0x1.2.0)
 enabled  threads:  01100000 00000000 00000000 00000000 00000000 00000000
  00000000 00000000
 Hot Backup end marker scn: 0x0000.00000000
 FILE HEADER:
      Software vsn=134217728=0x8000000, Compatibility Vsn=134217728=0x8000000
      Db Id=3253243709=0xc1e88f3d, Db Name='V804'
      Control Seq=1170=0x492, File size=6398=0x18fe
      File Number=1, Blksiz=8192, File Type=3
Tablespace #0 - SYSTEM  rel_fn:1
Creation   at   scn: 0x0000.00000003 02/13/98 17:03:05
Backup taken at scn: 0x04d8.5d15b0e8 06/03/98 15:36:24 thread:1
 reset logs count:0x13ed7fd4 scn: 0x04d8.5d142764 recovered at 05/26/98
12:06:49
 status:0x1 root dba:0x004000a9 chkpt cnt: 198 ctl cnt:197
begin-hot-backup file size: 6398
Checkpointed at scn: 0x04d8.5d15b0e8 06/03/98 15:36:24
 thread:1 rba:(0x8.13.10)
 enabled  threads:  01100000 00000000 00000000 00000000 00000000 00000000
  00000000 00000000
Backup Checkpointed at scn: 0x04d8.5d15b0e8 06/03/98 15:36:24
 thread:1 rba:(0x8.13.10)
 enabled  threads:  01100000 00000000 00000000 00000000 00000000 00000000
  00000000 00000000
External cache id: 0x0 0x0 0x0 0x0
Absolute fuzzy scn: 0x0000.00000000
Recovery fuzzy scn: 0x0000.00000000 01/01/88 00:00:00
DATA FILE #2:
```

```
  (name #5) /dev/vx/rdsk/datadg/v804_1_rbs
creation size=5118 block size=8192 status=0xe head=5 tail=5 dup=1
 tablespace 1, index=2 krfil=2 prev_file=0
 unrecoverable scn: 0x0000.00000000 01/01/88 00:00:00
 Checkpoint cnt:187 scn: 0x04d8.5d156282 05/29/98 12:42:08
 Stop scn: 0xffff.ffffffff 05/28/98 18:02:11
 Creation Checkpointed at scn: 0x0000.00002005 02/13/98 17:14:37
 thread:1 rba:(0x7.1576.bc)
 enabled  threads:  01000000 00000000 00000000 00000000 00000000 00000000
  00000000 00000000
 Offline scn: 0x04d8.5d142763 prev_range: 0
 Online Checkpointed at scn: 0x04d8.5d142764 05/26/98 13:57:40
 thread:1 rba:(0x1.2.0)
 enabled  threads:  01100000 00000000 00000000 00000000 00000000 00000000
  00000000 00000000
 Hot Backup end marker scn: 0x0000.00000000
 FILE HEADER:
      Software vsn=134217728=0x8000000, Compatibility Vsn=134217728=0x8000000
      Db Id=3253243709=0xc1e88f3d, Db Name='V804'
      Control Seq=1121=0x461, File size=5118=0x13fe
      File Number=2, Blksiz=8192, File Type=3
Tablespace #1 - RBS1  rel_fn:2
Creation   at   scn: 0x0000.00002005 02/13/98 17:14:37
Backup taken at scn: 0x0000.00000000 01/01/88 00:00:00 thread:0
 reset logs count:0x13ed7fd4 scn: 0x04d8.5d142764 recovered at 05/26/98
12:06:49
 status:0x4 root dba:0x00000000 chkpt cnt: 187 ctl cnt:186
begin-hot-backup file size: 0
Checkpointed at scn: 0x04d8.5d156282 05/29/98 12:42:08
 thread:1 rba:(0x7.2.10)
 enabled  threads:  01100000 00000000 00000000 00000000 00000000 00000000
  00000000 00000000
Backup Checkpointed at scn: 0x0000.00000000 01/01/88 00:00:00
 thread:0 rba:(0x0.0.0)
 enabled  threads:  00000000 00000000 00000000 00000000 00000000 00000000
  00000000 00000000
External cache id: 0x0 0x0 0x0 0x0
Absolute fuzzy scn: 0x0000.00000000
Recovery fuzzy scn: 0x0000.00000000 01/01/88 00:00:00
DATA FILE #3:
  (name #6) /dev/vx/rdsk/datadg/v804_2_rbs
creation size=5118 block size=8192 status=0xe head=6 tail=6 dup=1
 tablespace 2, index=3 krfil=3 prev_file=0
 unrecoverable scn: 0x0000.00000000 01/01/88 00:00:00
 Checkpoint cnt:185 scn: 0x04d8.5d15629f 06/01/98 10:29:41
 Stop scn: 0xffff.ffffffff 05/28/98 18:02:11
 Creation Checkpointed at scn: 0x0000.0000200b 02/13/98 17:14:43
 thread:1 rba:(0x7.1581.bc)
 enabled  threads:  01000000 00000000 00000000 00000000 00000000 00000000
  00000000 00000000
 Offline scn: 0x04d8.5d142763 prev_range: 0
 Online Checkpointed at scn: 0x04d8.5d142764 05/26/98 13:57:40
 thread:1 rba:(0x1.2.0)
 enabled  threads:  01100000 00000000 00000000 00000000 00000000 00000000
  00000000 00000000
 Hot Backup end marker scn: 0x0000.00000000
 FILE HEADER:
      Software vsn=134217728=0x8000000, Compatibility Vsn=134217728=0x8000000
      Db Id=3253243709=0xc1e88f3d, Db Name='V804'
      Control Seq=1146=0x47a, File size=5118=0x13fe
      File Number=3, Blksiz=8192, File Type=3
Tablespace #2 - RBS2  rel_fn:3
```

```
Creation   at   scn: 0x0000.0000200b 02/13/98 17:14:43
Backup taken at scn: 0x0000.00000000 01/01/88 00:00:00 thread:0
 reset logs count:0x13ed7fd4 scn: 0x04d8.5d142764 recovered at 05/26/98
12:06:49
 status:0x4 root dba:0x00000000 chkpt cnt: 185 ctl cnt:184
begin-hot-backup file size: 0
Checkpointed at scn: 0x04d8.5d15629f 06/01/98 10:29:41
 thread:1 rba:(0x7.24.10)
 enabled  threads:  01100000 00000000 00000000 00000000 00000000 00000000
  00000000 00000000
Backup Checkpointed at scn: 0x0000.00000000 01/01/88 00:00:00
 thread:0 rba:(0x0.0.0)
 enabled  threads:  00000000 00000000 00000000 00000000 00000000 00000000
  00000000 00000000
External cache id: 0x0 0x0 0x0 0x0
Absolute fuzzy scn: 0x0000.00000000
Recovery fuzzy scn: 0x0000.00000000 01/01/88 00:00:00
DATA FILE #4:
  (name #7) /dev/vx/rdsk/datadg/v804_1_temp
creation size=5118 block size=8192 status=0x80 head=7 tail=7 dup=1
 tablespace 3, index=4 krfil=4 prev_file=0
 unrecoverable scn: 0x0000.00000000 01/01/88 00:00:00
 Checkpoint cnt:193 scn: 0x04d8.5d15b0ef 06/03/98 15:36:45
 Stop scn: 0x04d8.5d15b0ef 06/03/98 15:36:45
 Creation Checkpointed at scn: 0x0000.00002012 02/13/98 17:14:50
 thread:1 rba:(0x7.1592.c0)
 enabled  threads:  01000000 00000000 00000000 00000000 00000000 00000000
  00000000 00000000
 Offline scn: 0x04d8.5d15629c prev_range: 2
 Online Checkpointed at scn: 0x04d8.5d1562a8 06/02/98 16:50:39
 thread:1 rba:(0x7.26.70)
 enabled  threads:  01100000 00000000 00000000 00000000 00000000 00000000
  00000000 00000000
 Hot Backup end marker scn: 0x0000.00000000
 FILE HEADER:
      Software vsn=134217728=0x8000000, Compatibility Vsn=134217728=0x8000000
      Db Id=3253243709=0xc1e88f3d, Db Name='V804'
      Control Seq=1172=0x494, File size=5118=0x13fe
      File Number=4, Blksiz=8192, File Type=3
Tablespace #3 - TEMP1  rel_fn:4
Creation   at   scn: 0x0000.00002012 02/13/98 17:14:50
Backup taken at scn: 0x0000.00000000 01/01/88 00:00:00 thread:0
 reset logs count:0x13ed7fd4 scn: 0x04d8.5d142764 recovered at 06/01/98
08:24:04
 status:0x0 root dba:0x00000000 chkpt cnt: 193 ctl cnt:192
begin-hot-backup file size: 0
Checkpointed at scn: 0x04d8.5d15b0ef 06/03/98 15:36:45
 thread:1 rba:(0x8.5f.130)
 enabled  threads:  01100000 00000000 00000000 00000000 00000000 00000000
  00000000 00000000
Backup Checkpointed at scn: 0x0000.00000000 01/01/88 00:00:00
 thread:0 rba:(0x0.0.0)
 enabled  threads:  00000000 00000000 00000000 00000000 00000000 00000000
  00000000 00000000
External cache id: 0x0 0x0 0x0 0x0
Absolute fuzzy scn: 0x0000.00000000
Recovery fuzzy scn: 0x0000.00000000 06/01/98 08:19:23
DATA FILE #5:
  (name #8) /dev/vx/rdsk/datadg/v804_2_temp
creation size=5118 block size=8192 status=0xe head=8 tail=8 dup=1
 tablespace 4, index=5 krfil=5 prev_file=0
 unrecoverable scn: 0x0000.00000000 01/01/88 00:00:00
```

```
 Checkpoint cnt:183 scn: 0x04d8.5d156282 05/29/98 12:42:08
 Stop scn: 0xffff.ffffffff 05/28/98 18:02:11
 Creation Checkpointed at scn: 0x0000.00002018 02/13/98 17:14:56
 thread:1 rba:(0x7.159d.c0)
 enabled  threads:  01000000 00000000 00000000 00000000 00000000 00000000
  00000000 00000000
 Offline scn: 0x04d8.5d142763 prev_range: 0
 Online Checkpointed at scn: 0x04d8.5d142764 05/26/98 13:57:40
 thread:1 rba:(0x1.2.0)
 enabled  threads:  01100000 00000000 00000000 00000000 00000000 00000000
  00000000 00000000
 Hot Backup end marker scn: 0x0000.00000000
 FILE HEADER:
      Software vsn=134217728=0x8000000, Compatibility Vsn=134217728=0x8000000
      Db Id=3253243709=0xc1e88f3d, Db Name='V804'
      Control Seq=1121=0x461, File size=5118=0x13fe
      File Number=5, Blksiz=8192, File Type=3
Tablespace #4 - TEMP2  rel_fn:5
Creation   at   scn: 0x0000.00002018 02/13/98 17:14:56
Backup taken at scn: 0x0000.00000000 01/01/88 00:00:00 thread:0
 reset logs count:0x13ed7fd4 scn: 0x04d8.5d142764 recovered at 05/26/98
12:06:49
 status:0x4 root dba:0x00000000 chkpt cnt: 183 ctl cnt:182
begin-hot-backup file size: 0
Checkpointed at scn: 0x04d8.5d156282 05/29/98 12:42:08
 thread:1 rba:(0x7.2.10)
 enabled  threads:  01100000 00000000 00000000 00000000 00000000 00000000
  00000000 00000000
Backup Checkpointed at scn: 0x0000.00000000 01/01/88 00:00:00
 thread:0 rba:(0x0.0.0)
 enabled  threads:  00000000 00000000 00000000 00000000 00000000 00000000
  00000000 00000000
External cache id: 0x0 0x0 0x0 0x0
Absolute fuzzy scn: 0x0000.00000000
Recovery fuzzy scn: 0x0000.00000000 01/01/88 00:00:00
DATA FILE #6:
  (name #9) /dev/vx/rdsk/datadg/v804_tools
creation size=2558 block size=8192 status=0xe head=9 tail=9 dup=1
 tablespace 5, index=6 krfil=6 prev_file=0
 unrecoverable scn: 0x0000.00000000 04/01/98 13:04:55
 Checkpoint cnt:183 scn: 0x04d8.5d156282 05/29/98 12:42:08
 Stop scn: 0xffff.ffffffff 05/28/98 18:02:11
 Creation Checkpointed at scn: 0x0000.0000201e 02/13/98 17:14:59
 thread:1 rba:(0x7.15a8.c0)
 enabled  threads:  01000000 00000000 00000000 00000000 00000000 00000000
  00000000 00000000
 Offline scn: 0x04d8.5d142763 prev_range: 0
 Online Checkpointed at scn: 0x04d8.5d142764 05/26/98 13:57:40
 thread:1 rba:(0x1.2.0)
 enabled  threads:  01100000 00000000 00000000 00000000 00000000 00000000
  00000000 00000000
 Hot Backup end marker scn: 0x0000.00000000
 FILE HEADER:
      Software vsn=134217728=0x8000000, Compatibility Vsn=134217728=0x8000000
      Db Id=3253243709=0xc1e88f3d, Db Name='V804'
      Control Seq=1121=0x461, File size=2558=0x9fe
      File Number=6, Blksiz=8192, File Type=3
Tablespace #5 - TOOLS  rel_fn:6
Creation   at   scn: 0x0000.0000201e 02/13/98 17:14:59
Backup taken at scn: 0x0000.00000000 01/01/88 00:00:00 thread:0
 reset logs count:0x13ed7fd4 scn: 0x04d8.5d142764 recovered at 05/26/98
12:06:49
```

```
 status:0x4 root dba:0x00000000 chkpt cnt: 183 ctl cnt:182
begin-hot-backup file size: 0
Checkpointed at scn: 0x04d8.5d156282 05/29/98 12:42:08
 thread:1 rba:(0x7.2.10)
 enabled  threads:  01100000 00000000 00000000 00000000 00000000 00000000
  00000000 00000000
Backup Checkpointed at scn: 0x0000.00000000 01/01/88 00:00:00
 thread:0 rba:(0x0.0.0)
 enabled  threads:  00000000 00000000 00000000 00000000 00000000 00000000
  00000000 00000000
External cache id: 0x0 0x0 0x0 0x0
Absolute fuzzy scn: 0x0000.00000000
Recovery fuzzy scn: 0x0000.00000000 01/01/88 00:00:00
DATA FILE #7:
  (name #10) /dev/vx/rdsk/datadg/v804_users
creation size=2558 block size=8192 status=0x86 head=10 tail=10 dup=1
 tablespace 6, index=7 krfil=7 prev_file=0
 unrecoverable scn: 0x0000.00000000 03/25/98 15:30:48
 Checkpoint cnt:188 scn: 0x04d8.5d15b0f1 06/03/98 15:36:57
 Stop scn: 0x04d8.5d15b0f1 06/03/98 15:36:57
 Creation Checkpointed at scn: 0x0000.00002025 02/13/98 17:15:03
 thread:1 rba:(0x7.15ba.c0)
 enabled  threads:  01000000 00000000 00000000 00000000 00000000 00000000
  00000000 00000000
 Offline scn: 0x04d8.5d142763 prev_range: 0
 Online Checkpointed at scn: 0x04d8.5d142764 05/26/98 13:57:40
 thread:1 rba:(0x1.2.0)
 enabled  threads:  01100000 00000000 00000000 00000000 00000000 00000000
  00000000 00000000
 Hot Backup end marker scn: 0x0000.00000000
 FILE HEADER:
      Software vsn=134217728=0x8000000, Compatibility Vsn=134217728=0x8000000
      Db Id=3253243709=0xc1e88f3d, Db Name='V804'
      Control Seq=1175=0x497, File size=2558=0x9fe
      File Number=7, Blksiz=8192, File Type=3
Tablespace #6 - USERS  rel_fn:7
Creation   at   scn: 0x0000.00002025 02/13/98 17:15:03
Backup taken at scn: 0x0000.00000000 01/01/88 00:00:00 thread:0
 reset logs count:0x13ed7fd4 scn: 0x04d8.5d142764 recovered at 05/29/98
11:16:49
 status:0x0 root dba:0x00000000 chkpt cnt: 188 ctl cnt:187
begin-hot-backup file size: 0
Checkpointed at scn: 0x04d8.5d15b0f1 06/03/98 15:36:57
 thread:1 rba:(0x8.74.178)
 enabled  threads:  01100000 00000000 00000000 00000000 00000000 00000000
  00000000 00000000
Backup Checkpointed at scn: 0x0000.00000000 01/01/88 00:00:00
 thread:0 rba:(0x0.0.0)
 enabled  threads:  00000000 00000000 00000000 00000000 00000000 00000000
  00000000 00000000
External cache id: 0x0 0x0 0x0 0x0
Absolute fuzzy scn: 0x0000.00000000
Recovery fuzzy scn: 0x0000.00000000 05/29/98 11:16:39
DATA FILE #8:
  (name #14) /dev/vx/rdsk/datadg/v804_rcvcat
creation size=2558 block size=8192 status=0xe head=14 tail=14 dup=1
 tablespace 7, index=8 krfil=8 prev_file=0
 unrecoverable scn: 0x0000.00000000 01/01/88 00:00:00
 Checkpoint cnt:173 scn: 0x04d8.5d156282 05/29/98 12:42:08
 Stop scn: 0xffff.ffffffff 05/28/98 18:02:11
 Creation Checkpointed at scn: 0x0000.00002aad 02/17/98 16:39:05
 thread:1 rba:(0xd.1552.c0)
```

```
 enabled  threads:  01100000 00000000 00000000 00000000 00000000 00000000
  00000000 00000000
 Offline scn: 0x04d8.5d142763 prev_range: 0
 Online Checkpointed at scn: 0x04d8.5d142764 05/26/98 13:57:40
 thread:1 rba:(0x1.2.0)
 enabled  threads:  01100000 00000000 00000000 00000000 00000000 00000000
  00000000 00000000
 Hot Backup end marker scn: 0x0000.00000000
 FILE HEADER:
      Software vsn=134217728=0x8000000, Compatibility Vsn=134217728=0x8000000
      Db Id=3253243709=0xc1e88f3d, Db Name='V804'
      Control Seq=1121=0x461, File size=2558=0x9fe
      File Number=8, Blksiz=8192, File Type=3
Tablespace #7 - RCVCAT  rel_fn:8
Creation   at   scn: 0x0000.00002aad 02/17/98 16:39:05
Backup taken at scn: 0x0000.00000000 01/01/88 00:00:00 thread:0
 reset logs count:0x13ed7fd4 scn: 0x04d8.5d142764 recovered at 05/26/98
12:06:49
 status:0x4 root dba:0x00000000 chkpt cnt: 173 ctl cnt:172
begin-hot-backup file size: 0
Checkpointed at scn: 0x04d8.5d156282 05/29/98 12:42:08
 thread:1 rba:(0x7.2.10)
 enabled  threads:  01100000 00000000 00000000 00000000 00000000 00000000
  00000000 00000000
Backup Checkpointed at scn: 0x0000.00000000 01/01/88 00:00:00
 thread:0 rba:(0x0.0.0)
 enabled  threads:  00000000 00000000 00000000 00000000 00000000 00000000
  00000000 00000000
External cache id: 0x0 0x0 0x0 0x0
Absolute fuzzy scn: 0x0000.00000000
Recovery fuzzy scn: 0x0000.00000000 01/01/88 00:00:00
```

Redo log headers dump

```
Dump file /oracle/app/oracle/admin/V8041/udump/v8041_ora_12282.trc
Oracle8 Enterprise Edition Release 8.0.4.1.0 - Production
With the Partitioning, Objects and Parallel Server options
PL/SQL Release 8.0.4.1.0 - Production
ORACLE_HOME = /oracle/app/oracle/product/8.0.4
System name:      SunOS
Node name:  uksn116
Release:    5.5.1
Version:    Generic_103640-14
Machine:    sun4u
Instance name: V8041
Redo thread mounted by this instance: 1
Oracle process number: 14
Unix process pid: 12282, image: oracleV8041

Mon Jun  8 10:33:26 1998
*** SESSION ID:(14.855) 1998.06.08.10.33.26.000
DUMP OF LOG FILES: 6 logs in database
LOG FILE #1:
  (name #1) /dev/vx/rdsk/datadg/v804_1_redo_1
 Thread 1 redo log links: forward: 2 backward: 0
 siz: 0x1fe0 seq: 0x0000000a hws: 0x1 bsz: 512 nab: 0xffffffff flg: 0x8 dup: 1
 Archive links: fwrd: 4 back: 0 Prev scn: 0x04d8.fea9885c
 Low scn: 0x04d8.fea98ff9 06/05/98 08:39:59
 Next scn: 0xffff.ffffffff 06/03/98 15:24:41
 FILE HEADER:
      Software vsn=134217728=0x8000000, Compatibility Vsn=134217728=0x8000000
      Db Id=3253243709=0xc1e88f3d, Db Name='V804'
      Control Seq=1206=0x4b6, File size=8160=0x1fe0
      File Number=1, Blksiz=512, File Type=2
 descrip:"Thread 0001, Seq# 0000000010, SCN 0x04d8fea98ff9-0xffffffffffff"
 thread: 1 nab: 0xffffffff seq: 0x0000000a hws: 0x1 eot: 1 dis: 0
 reset logs count: 0x13ed7fd4 scn: 0x04d8.5d142764
 Low scn: 0x04d8.fea98ff9 06/05/98 08:39:59
 Next scn: 0xffff.ffffffff 06/03/98 15:24:41
 Enabled scn: 0x04d8.5d142764 05/26/98 13:57:40
 Thread closed scn: 0x04d8.fea98ff9 06/05/98 08:39:59
 Log format vsn: 0x8000000 Disk cksum: 0xeb6f Calc cksum: 0xeb6f
LOG FILE #2:
  (name #2) /dev/vx/rdsk/datadg/v804_1_redo_2
 Thread 1 redo log links: forward: 3 backward: 1
 siz: 0x1fe0 seq: 0x00000008 hws: 0x4 bsz: 512 nab: 0x98 flg: 0x1 dup: 1
 Archive links: fwrd: 0 back: 0 Prev scn: 0x04d8.5d156282
 Low scn: 0x04d8.5d15b0de 06/03/98 15:24:41
 Next scn: 0x04d8.fea9885c 06/03/98 16:19:56
 FILE HEADER:
      Software vsn=134217728=0x8000000, Compatibility Vsn=134217728=0x8000000
      Db Id=3253243709=0xc1e88f3d, Db Name='V804'
      Control Seq=1192=0x4a8, File size=8160=0x1fe0
      File Number=2, Blksiz=512, File Type=2
 descrip:"Thread 0001, Seq# 0000000008, SCN 0x04d85d15b0de-0x04d8fea9885c"
 thread: 1 nab: 0x98 seq: 0x00000008 hws: 0x4 eot: 0 dis: 0
 reset logs count: 0x13ed7fd4 scn: 0x04d8.5d142764
 Low scn: 0x04d8.5d15b0de 06/03/98 15:24:41
 Next scn: 0x04d8.fea9885c 06/03/98 16:19:56
 Enabled scn: 0x04d8.5d142764 05/26/98 13:57:40
 Thread closed scn: 0x04d8.5d15b0de 06/03/98 15:24:41
 Log format vsn: 0x8000000 Disk cksum: 0xd7a4 Calc cksum: 0xd7a4
LOG FILE #3:
```

```
  (name #3) /dev/vx/rdsk/datadg/v804_1_redo_3
 Thread 1 redo log links: forward: 0 backward: 2
 siz: 0x1fe0 seq: 0x00000009 hws: 0x5 bsz: 512 nab: 0x18f flg: 0x1 dup: 1
 Archive links: fwrd: 0 back: 0 Prev scn: 0x04d8.5d15b0de
 Low scn: 0x04d8.fea9885c 06/03/98 16:19:56
 Next scn: 0x04d8.fea98ff9 06/05/98 08:39:59
 FILE HEADER:
      Software vsn=134217728=0x8000000, Compatibility Vsn=134217728=0x8000000
      Db Id=3253243709=0xc1e88f3d, Db Name='V804'
      Control Seq=1206=0x4b6, File size=8160=0x1fe0
      File Number=3, Blksiz=512, File Type=2
 descrip:"Thread 0001, Seq# 0000000009, SCN 0x04d8fea9885c-0x04d8fea98ff9"
 thread: 1 nab: 0x18f seq: 0x00000009 hws: 0x5 eot: 0 dis: 0
 reset logs count: 0x13ed7fd4 scn: 0x04d8.5d142764
 Low scn: 0x04d8.fea9885c 06/03/98 16:19:56
 Next scn: 0x04d8.fea98ff9 06/05/98 08:39:59
 Enabled scn: 0x04d8.5d142764 05/26/98 13:57:40
 Thread closed scn: 0x04d8.fea98863 06/04/98 14:17:38
 Log format vsn: 0x8000000 Disk cksum: 0xd1d2 Calc cksum: 0xd1d2
LOG FILE #4:
  (name #11) /dev/vx/rdsk/datadg/v804_2_redo_1
 Thread 2 redo log links: forward: 5 backward: 0
 siz: 0x1fe0 seq: 0x0000000b hws: 0x1 bsz: 512 nab: 0xffffffff flg: 0x8 dup: 1
 Archive links: fwrd: 0 back: 1 Prev scn: 0x04d8.fea9924b
 Low scn: 0x04d8.fea994b9 06/08/98 09:24:35
 Next scn: 0xffff.ffffffff 06/05/98 08:40:24
 FILE HEADER:
      Software vsn=134217728=0x8000000, Compatibility Vsn=134217728=0x8000000
      Db Id=3253243709=0xc1e88f3d, Db Name='V804'
      Control Seq=1216=0x4c0, File size=8160=0x1fe0
      File Number=4, Blksiz=512, File Type=2
 descrip:"Thread 0002, Seq# 0000000011, SCN 0x04d8fea994b9-0xffffffffffff"
 thread: 2 nab: 0xffffffff seq: 0x0000000b hws: 0x1 eot: 1 dis: 0
 reset logs count: 0x13ed7fd4 scn: 0x04d8.5d142764
 Low scn: 0x04d8.fea994b9 06/08/98 09:24:35
 Next scn: 0xffff.ffffffff 06/05/98 08:40:24
 Enabled scn: 0x04d8.5d142764 05/26/98 13:57:40
 Thread closed scn: 0x04d8.fea994b9 06/08/98 09:24:35
 Log format vsn: 0x8000000 Disk cksum: 0xae06 Calc cksum: 0xae06
LOG FILE #5:
  (name #12) /dev/vx/rdsk/datadg/v804_2_redo_2
 Thread 2 redo log links: forward: 6 backward: 4
 siz: 0x1fe0 seq: 0x00000009 hws: 0x2 bsz: 512 nab: 0x1fdf flg: 0x1 dup: 1
 Archive links: fwrd: 0 back: 0 Prev scn: 0x04d8.fea98b50
 Low scn: 0x04d8.fea98ff7 06/05/98 08:40:24
 Next scn: 0x04d8.fea9924b 06/05/98 08:40:55
 FILE HEADER:
      Software vsn=134217728=0x8000000, Compatibility Vsn=134217728=0x8000000
      Db Id=3253243709=0xc1e88f3d, Db Name='V804'
      Control Seq=1212=0x4bc, File size=8160=0x1fe0
      File Number=5, Blksiz=512, File Type=2
 descrip:"Thread 0002, Seq# 0000000009, SCN 0x04d8fea98ff7-0x04d8fea9924b"
 thread: 2 nab: 0x1fdf seq: 0x00000009 hws: 0x2 eot: 0 dis: 0
 reset logs count: 0x13ed7fd4 scn: 0x04d8.5d142764
 Low scn: 0x04d8.fea98ff7 06/05/98 08:40:24
 Next scn: 0x04d8.fea9924b 06/05/98 08:40:55
 Enabled scn: 0x04d8.5d142764 05/26/98 13:57:40
 Thread closed scn: 0x04d8.fea98ff7 06/05/98 08:40:24
 Log format vsn: 0x8000000 Disk cksum: 0xc2a1 Calc cksum: 0xc2a1
LOG FILE #6:
  (name #13) /dev/vx/rdsk/datadg/v804_2_redo_3
 Thread 2 redo log links: forward: 0 backward: 5
```

```
siz: 0x1fe0 seq: 0x0000000a hws: 0x2 bsz: 512 nab: 0x1fe1 flg: 0x1 dup: 1
Archive links: fwrd: 0 back: 0 Prev scn: 0x04d8.fea98ff7
Low scn: 0x04d8.fea9924b 06/05/98 08:40:55
Next scn: 0x04d8.fea994b9 06/08/98 09:24:35
FILE HEADER:
     Software vsn=134217728=0x8000000, Compatibility Vsn=134217728=0x8000000
     Db Id=3253243709=0xc1e88f3d, Db Name='V804'
     Control Seq=1216=0x4c0, File size=8160=0x1fe0
     File Number=6, Blksiz=512, File Type=2
descrip:"Thread 0002, Seq# 0000000010, SCN 0x04d8fea9924b-0x04d8fea994b9"
thread: 2 nab: 0x1fe1 seq: 0x0000000a hws: 0x2 eot: 0 dis: 0
reset logs count: 0x13ed7fd4 scn: 0x04d8.5d142764
Low scn: 0x04d8.fea9924b 06/05/98 08:40:55
Next scn: 0x04d8.fea994b9 06/08/98 09:24:35
Enabled scn: 0x04d8.5d142764 05/26/98 13:57:40
Thread closed scn: 0x04d8.fea9924b 06/05/98 08:40:55
Log format vsn: 0x8000000 Disk cksum: 0xcb4f Calc cksum: 0xcb4f
```

Lesson 4 Practice: Troubleshooting Recovery Operations

Exercise 1

This exercise will test your understanding of controlfile and datafile header structures. We will introduce some simple scenarios that require you to use standard dumps and V$ views to resolve.

1. Run the shell script $HOME/LABS/case1.sh

Do not look at what the shell script does as it will make the exercise trivial !!!

2. Startup mount the database and take a file header dump.

Alter session set events 'immediate trace name file_hdrs level 10' ;

3. Explain the status of each of the datafiles, how could they have ended in this state? Confirm your observations with V$ view entries.

_____

_____

_____

4. Is the database open or closed? Open the database if necessary. What command did you use? Why?

_____

_____

_____

Exercise 2

This exercise will test an understanding of recovery. Before attempting this exercise ensure we need to make certain SCOTT's object are created in the default tablespace USERS. Run the demobld.sql script as SYSTEM in order to recreate the SCOTT user and his objects ($HOME/LABS/demobld.sql)

1. Record the current log sequence number. _____

2. Place the USERS tablespace in hot backup mode.
   ALTER TABLESPACE users BEGIN BACKUP;

3. Switch logfiles three times -
   ALTER SYSTEM SWITCH LOGFILE;

4. Take an O/S backup of the USERS tablespace and then end hot backup mode.

5. Create a table nologging as SCOTT within the USERS tablespace
   CREATE TABLE nolog_table (col1 NUMBER) NOLOGGING;
   Insert some rows into this table.
   Create another table as select from the EMP table.
   CREATE TABLE log_table AS SELECT * FROM emp;

6. Switch some log files (3 times).

7. Select the block number from dba_segments for the header block of the EMP table.

   _____

8. Shutdown abort

9. Run the shell script case2.sh providing the arguments ORACLE_SID and header block from step 7.
   eg. **%** case2.sh ST22 292

10. Startup the database and SELECT * FROM scott.emp (or whomever your user is).
    What happens? What is wrong?

    _____

11. What can you do ? Perform any means of recovery deemed necessary.

12. What tables are missing if any (look at nolog_table and log_table)? Why?

    _____

    _____

Lesson 8 Practice: Tablespace Point-in-Time Recovery

Lab exercise - A step by step example to TSPITR.

(Simplified verion of WebIV note 62332.1 by NEVANS.UK)

PURPOSE

The purpose of this document is to illustrate the steps involved in Tablespace Point In Time Recovery (TSPITR) via a working example.

This practice implemented here is a relatively common scenario of a user inadvertantly dropping a table that they still require. In order to simplify the issue the user (PITR) creates all objects apart from indexes in a single tablespace.

Our starting point will be the first startup after we have taken a full offline backup of all our datafiles and the controlfiles (by whichever means). It is assumed that the database is running in ARCHIVELOG mode.

Part 1 - Setup

~~~~~~~~~~~~~~~~~~~~~~~~~~~~

1) Create a user PITR with a default tablespace of USERS as the test user. Grant CONNECT and RESOURCE privileges to PITR. As the user PITR create an EMP, DEPT, and BONUS table by selecting from the SCOTT schema. Then shutdown the database and take a full backup.

2) Restart the instance and generate a backup control file (not a trace file for control file creation).

Note: The backup controlfile used for the TSPITR should be created at the same time as the backup of the recovery set and must, without exception, be created with this command as you will get the following error when starting the clone database...:

```
ORA1696, 00000, "controlfile is not a clone controlfile"
// *Cause:  Attempting to mount, a database as a clone when it is already
//          mounted by another instance not as a clone or attempting to use
//          a current controlfile for a clone.
// *Action: Mount without the clone option or use a backup controlfile and
//          shutdown the other instances before mounting as a clone.
```

3) Connect as PITR and list the tables created and the tablespace(s) in which they reside.

4) Query the rows in the BONUS and DEPT tables.

5) Insert the following rows into the BONUS table. Don't forget to commit your work.

```
SQL> insert into bonus values ('SMITH','MANAGER',30000,2000);
1 row processed.
SQL> insert into bonus values ('JONES','ADMIN',15000,1000);
1 row processed.
SQL> insert into bonus values ('BROWN','SECURITY',16000,500);
1 row processed.
```

6) Optional: Create an index on the ENAME column of the BONUS table in the SYSTEM tablespace.

7) Get the current time.
~~~~~~~~~~~~~~~~~~~~~~~~~~~~

8) Create the problem by dropping the DEPT table. Not realizing your mistake, create the table SALGRADE with three columns of type NUMBER: GRADE, LOSAL, and HISAL.

   Note: This is to highlight the fact any committed transactions that are made after the point in time at which we stop recovery, (just before the DEPT table was dropped), will not be recovered. Hence, SALGRADE will not be recovered.

Part 2 - Diagnosing the Situation

~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

The user has realised his error and now approaches the DBA to get the table retrieved. The user informs the DBA that the DEPT table was dropped around 1:30 pm today.

9) Check which objects will be lost after performing TSPITR with SQL similar to the following. See README.pit (in $ORACLE_HOME/rdbms/doc) for further details...:

10) Deciding to proceed with TSPITR, prevent further data loss on the database by taking the tablespaces to be recovered offline. Then, in preparation for performing TSPITR, make certain the contents of the current log will be available as an archive file if needed.

Part 3 - Preparing the CLONE database for TSPITR export

~~~~~~~~~~~~~~~~~~~

16) Recover the database until the time recorded before the DEPT table was dropped (Step 7). Open the database.

17) Connect as the user PITR and query the objects contained in the recovered file. Verify the DEPT exists.

18) We must now ascertain what relationships, if any, exist between objects across the bounds of the recovery set. Further details of this can be found in README.pit. In short, the dictionary view provided for gathering this information is TS_PITR_CHECK. In our example we would use the following SQL to query this view:

If you did not create the index on BONUS, then skip from here until you see … *Skip until here*

19) Since it is an index, drop BONUS_IDX rather than alter the recovery set.

…. Skip until here

20) Verify that there are no objects which are not fully contained.

Part 4 - Performing TSPITR export

~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

21) Now we are ready to export the required tablespace from the clone database. Perform the TSPITR export for the USERS tablespace.

Part 5 - Performing TSPITR import

~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

The next step is to import the tablespace metatdata back into our production instance.

22) Reset the Unix environment to point to the Production database.

23) Prepare the Production database for TSPITR. Remember to perform any actions here that were performed on the CLONE database when preparing for TSPITR export. Then offline the tablespace to be recovered.

24) Replace the recovery set datafiles on the production database with the datafiles we have recovered on the clone database.

!!! Be safe: Save the production files first!!!

25) Now perform the TSPITR import as SYS.

Part 6 - Cleanup

~~~~~~~~~~~~~~~

26) Make the USERS tablespace available again.

Optional: Now we have to recreate the dropped index.
Skip from here until you see …*Skip until here*

27) Recreate the index we had drop.

… Skip until here

28) The TSPITR recovery is now complete. To check this we can connect as the user PITR and see what objects have been recovered. Notice that the SALGRADE table no longer exists.
~~~~~~~~~~~~~~~

Part 7 - Final Task

~~~~~~~~~~~~~~~~~~

29) After TSPITR has been done on a tablespace, it is VITALLY important that it is backed up. Failure to do so could result in loss of that tablespace in the event of media failure, because the archived logs from the last backup taken of that database do not logically link to the recovered tablespaces. If an attempt is made to recover any recovery set tablespaces from a backup taken before TSPITR, it will fail with ORA-01246: 'recovering files through TSPITR of tablespace %s'.

RELATED DOCUMENTS

<Note:46090.1> 3. Terminology
<Note:46091.1> 4. Restrictions of TSPITR
<Note:46092.1> 5. Database requirements of Tablespace Point-In-Time
<Note:46094.1> 7. Steps involved in Tablespace Point-In-Time Recovery
<Note:46095.1> 8.1 Advisory Note
<Note:46098.1> 9. Addendum 'Tablespace Point-In-Time Recovery Alerts'
README.pit file: $ORACLE_HOME/rdbms/doc on UNIX installations
%ORACLE_HOME%\Rdbms or %ORACLE_HOME%\rdbms80\doc
on Windows NT installations
~~~~~~~~~~~~~~~~~~

C
Solutions

Lesson 2 Practice Solution: Viewing Redo Records

In this practice we will issue some SQL operations against a table in the database. Then we will dump the redo logs to view the redo records generated for this DML.

1. Connect as SCOTT/TIGER and query the BONUS table.
   SQL> connect scott/tiger
   Connected.
   SQL> describe bonus;

Name	Null?	Type
ENAME		VARCHAR2(10)
JOB		VARCHAR2(9)
SAL		NUMBER
COMM		NUMBER

   SQL> select * from bonus;
   no rows selected
2. Perform an ALTER SYSTEM SWITCH LOGFILE.
   SQL> connect internal
   Connected.
   SQL> alter system switch logfile;
   System altered.
3. Insert a row into the BONUS table.
   SQL> connect scott/tiger
   Connected.
   SQL> insert into bonus values ('SMITH', 'CLERK', 22000, 0);
   1 row created.
4. Create an index on the ENAME column of the BONUS table.
   HINT: CREATE INDEX … ON BONUS(ENAME);
   SQL> create index bonus_name_indx on bonus(ename);
   Index created.
5. Insert another row into the BONUS table.
   SQL> insert into bonus values ('JONES', 'SALES', 18000, 15);
   1 row created.
6. Commit.
7. Delete a row from the BONUS table.
   SQL> delete from bonus where ename = 'JONES';
   1 row deleted.
8. Commit.
9. Delete all rows from the BONUS table.
   SQL> delete from bonus;
   1 row deleted.
10. Rollback.
    SQL> rollback;
    Rollback complete.

11. Switch the logs again.
    SQL> connect internal
    Connected.
    SQL> alter system switch logfile;
    System altered.
12. Find out which log is the current log and which log was previously the current log.
    SQL> archive log list;
    Database log mode Archive Mode
    Automatic archival Enabled
    Archive destination D:\Oracle8i\oradata\or8i\archive
    Oldest online log sequence 18
    Next log sequence to archive 21
    Current log sequence 21
    SQL> select a.group#, b.member, a.sequence# from v$log a, v$logfile b
    2 where a.group# = b.group#;

GROUP#	MEMBER	SEQUENCE#
1	D:\ORACLE8I\ORADATA\OR8I\REDO04.LOG	21
2	D:\ORACLE8I\ORADATA\OR8I\REDO03.LOG	18
3	D:\ORACLE8I\ORADATA\OR8I\REDO02.LOG	19
4	D:\ORACLE8I\ORADATA\OR8I\REDO01.LOG	20

13. Dump the previous log.
    HINT: ALTER SYSTEM DUMP LOGFILE ‘<full path name’>;
    SQL> alter system dump logfile 'd:\oracle8i\oradata\or8i\redo01.log';
    System altered.
14. View the trace file and find the first Insert command. How many change vectors are contained within this redo record? Why are there more than one change vectors?
    SQL> select object_id from user_objects where object_name = 'BONUS';
    OBJECT_ID

    12236
    % vi v815_ora_2709.trc
    :/12236

    There are 4 change vectors for this redo record:
    #1 - Begin transaction - Opcode 5.2 [KTURDH - Update rollback segment header]
    #2 - Insert rollback information - Opcode 5.1 [KTURDB - Undo block or undo segment header]
    #3 - Insert Row Piece redo - Opcode 11.2 [KDOIRP - Insert Row Piece]
    #4 - Start audit log record - Opcode 5.19 [KTUTSL - Tx Start audit Log record]

    REDO RECORD - Thread:1 RBA: 0x000014.00000002.00f0 LEN: 0x01d4 VLD: 0x01
    SCN scn: 0x0000.000491b1 01/31/00 13:47:41
    CHANGE #1 TYP:0 CLS:27 AFN:3 DBA:0x00c0041c SCN:0x0000.0004919c SEQ: 1 **OP:5.2**
    ktudh redo: slt: 0x0009 sqn: 0x0000001d flg: 0x0012 siz: 80 fbi: 0
    uba: 0x00c00499.000a.06 pxid: 0x0000.000.00000000
    CHANGE #2 TYP:0 CLS:28 AFN:3 DBA:0x00c00499 SCN:0x0000.0004919b SEQ: 6 **OP:5.1**
    ktudb redo: siz: 80 spc: 1124 flg: 0x0012 seq: 0x000a rec: 0x06
    xid: 0x0008.009.0000001d
    ktubl redo: slt: 9 rci: 0 opc: 11.1 **objn: 12236** objd: 12236 tsn: 1
    Undo type: Regular undo **Begin trans** Last buffer split: No

Temp Object: No
Tablespace Undo: No
0x00000000 prev ctl uba: 0x00c00498.000a.01
prev ctl max cmt scn: 0x0000.0002b8c4 prev tx cmt scn: 0x0000.0002b8d9
KDO undo record:
KTB Redo
op: 0x03 ver: 0x01
op: Z
KDO Op code: DRP xtype: XA bdba: 0x00800111 hdba: 0x00800110
itli: 1 ispac: 0 maxfr: 1177
tabn: 0 slot: 0(0x0)
CHANGE #3 TYP:0 CLS: 1 AFN:2 DBA:0x00800111 SCN:0x0000.000491b1 SEQ: 4 **OP:11.2**
KTB Redo
op: 0x01 ver: 0x01
op: F xid: 0x0008.009.0000001d uba: 0x00c00499.000a.06
KDO Op code: IRP xtype: XA bdba: 0x00800111 hdba: 0x00800110
itli: 1 ispac: 0 maxfr: 1177
tabn: 0 slot: 0(0x0) size/delt: 21
fb: --H-FL-- lb: 0x1 cc: 4
null: ----
col 0: [5] 53 4d 49 54 48
col 1: [5] 43 4c 45 52 4b
col 2: [3] c3 03 15
col 3: [1] 80
CHANGE #4 MEDIA RECOVERY MARKER SCN:0x0000.00000000 SEQ: 0 **OP:5.19**
session number = 14
serial number = 1695
current username = SCOTT
login username = SCOTT
client info =
OS username = usupport
Machine name = tcsol1
OS terminal = pts/17
OS process id = 2707
OS program name = sqlplus@tcsol1 (TNS V1-V3)

15. Locate the commit marker for the second insert.
Scan through the dump file after the insert record above until you find the first record of the type shown below:

```
REDO RECORD - Thread:1 RBA: 0x000014.00000004.0010 LEN: 0x0050 VLD: 0x01
SCN scn:  0x0000.000491b3 01/31/00 13:48:28
CHANGE #1 TYP:0 CLS:27 AFN:3 DBA:0x00c0041c SCN:0x0000.000491b1 SEQ:  1
OP:5.4
ktucm redo: slt: 0x0009 sqn: 0x0000001d srt: 0 sta: 9 flg: 0x2
ktucf redo: uba: 0x00c00499.000a.06 ext: 2 spc: 1042 fbi: 0
```

Notice this redo record only has one change vector - Opcode 5.4: KTURCM - Commit Transaction (transaction table update)

16. How many redo records were generated for the CREATE INDEX command?
Answers may vary depending on the activity level of the system.
28 redo records
    1) Update (opcode 11.5) _Next_Object row of OBJ$ (objn 18) - increment new object number
    col 0: [2] c1 02 → OBJ# = 1

col 1: [4] c3 02 17 28 → DATAOBJ = 12239

2) Commit record
3) Insert row (opcode 11.2) into OBJ$ (objn 18) - for the index we are creating
   col 0: [4] c3 02 17 27 → OBJ# = 12238
   col 1: [4] c3 02 17 27 → DATAOBJ# = 12238
   col 2: [2] c1 1b → OWNER# = 26
   col 3: [15] 42 4f 4e 55 53 5f 4e 41 4d 45 5f 49 4e 44 58
   → NAME= BONUS_NAME_INDX
   …
4) Insert leaf row (opcode 10.2) for index I_OBJ1 (objn: 33)
5) Insert leaf row (opcode 10.2) for index I_OBJ2 (objn: 34)
6) Insert row (opcode 11.2) into FET$ (objn: 12)
   col 0: [2] c1 03 → FILE# = 2
   col 1: [3] c2 0f 17 → BLOCK# = 1422
   col 2: [3] c2 02 10 → LENGTH = 115
7) Update row (opcode 11.5, KDO Op code: URP) in FET$ (objn: 12) to remove space for index
   OLD: col 2: [3] c2 02 15 → LENGTH = 12
   NEW: col 2: [2] c1 06 -> LENGTH = 5
8) Commit marker
9) Insert row (opcode 11.2, KDO Op code: IRP) into SEG$ (objn: 14)
   null: ---
   col 0: [2] c1 02 → 1
   col 1: [2] c1 03 → FILE# = 2
   col 2: [3] c2 0f 12 → BLOCK# = 1417
10) Insert leaf row (Opcode 10.2) into I_FILE#_BLOCK# (objn: 9)
11) Commit marker
12) Insert row (opcode 11.2, KDO Op code: IRP) into SEG$ (objn: 14)
   col 0: [2] c1 04 → 3
   col 1: [2] c1 06 → BLOCKS = 5
   col 2: [2] c1 02 → EXTENTS = 1
   col 3: [2] c1 06 → INIEXTS = 5
   col 4: [2] c1 02 → MINEXTS = 1
   col 5: [3] c2 02 16 → MAXEXTS = 121
   col 6: [2] c1 06 → EXTSIZE = 5
   col 7: [2] c1 33 → EXTPCT = 50
   col 8: [2] c1 1b → USER# = 26
   col 9: [1] 80 → 0
   col 10: [1] 80 → 0
   col 11: [1] 80 → 0
   col 12: [1] 80 → 0
   col 13: [1] 80 → 0
   col 14: [1] 80 → 0
13) Insert row (opcode 11.2) into UET$ (objn: 13)
   col 0: [1] 80 → 0
   col 1: [2] c1 03 → file# = 2
   col 2: [3] c2 0f 12 → block# = 1417
   col 3: [2] c1 06 → length = 5
14) Update row (opcode 11.5) in TSQ$ (objn: 57) - update user's tablespace quota
   OLD: col 2: [2] c1 2e → BLOCKS = 45
   NEW: col 2: [2] c1 33 → BLOCKS = 50
15) Delete row (opcode 11.3) from FET$ (objn: 12).

16) Commit marker
17) Data segment format (Op code 13.1) for objn: 12238 (our new index)
18) Format new block (op code 13.5)
19) Modify freelist for new segment (opcode 13.7)
20) Add extent to segment (opcode 14.4)
21) Index root block redo (opcode 10.31) - for creating the index
22) Insert row (opcode 11.2) into ICOL$ (objn: 20)
    col 0: [4] c3 02 17 27 → OBJ# = 12238
    col 1: [2] c1 02 → COL# = 1
    col 2: [2] c1 02 → POS# = 1
    col 3: [1] 80 → SEGCOL = 0
    col 4: [1] 80 → SEGCOLLENGTH = 0
    col 5: [1] 80 → OFFSET = 0
    col 6: [2] c1 02 → INTCOL# = 1
    col 7: [1] 80 → SPARE1 = 0
23) Insert leaf row (opcode 10.2) into I_ICOL1 (objn: 37)
24) Insert row (opcode 11.2) into IND$ (objn: 19)
25) Insert leaf row (opcode 10.2) into I_IND1 (objn: 36)
26) Update row (opcode 11.5) in OBJ$ (objn: 18)
    OLD: col 8: [7] 78 64 01 1f 0e 2d 16 → MTIME= "31-JAN-2000 13:44:21 AD"
    NEW: col 8: [7] 78 64 01 1f 0e 31 1d → MTIME= "31-JAN-2000 13:48:28 AD"
27) Update row (opcode 11.5) in SEG$ (objn: 14)
    OLD: col 0: [2] c1 04 → TYPE# = 3
    NEW: col 0: [2] c1 07 → TYPE# = 6
28) Commit marker.

15. Was redo generated for the ROLLBACK?

<u>Yes. From the end of the log file dump we have several records which show the deletion and subsequent insertion (to undo the deletion) followed by a commit record.</u>

REDO RECORD - Thread:1 RBA: 0x000014.0000001b.0010 LEN: 0x0214 VLD: 0x01
SCN scn: 0x0000.000491c3 01/31/00 13:50:55
CHANGE #1 TYP:0 CLS:19 AFN:3 DBA:0x00c00160 SCN:0x0000.000491a7 SEQ: 1 OP:5.2
ktudh redo: slt: 0x0006 sqn: 0x00000026 flg: 0x0012 siz: 168 fbi: 0
    uba: 0x00c02349.0017.05 pxid: 0x0000.000.00000000
CHANGE #2 TYP:0 CLS:20 AFN:3 DBA:0x00c02349 SCN:0x0000.000491a6 SEQ: 5 OP:5.1
ktudb redo: siz: 168 spc: 1426 flg: 0x0012 seq: 0x0017 rec: 0x05
    xid: 0x0004.006.00000026
…
CHANGE #3 TYP:2 CLS: 1 AFN:2 DBA:0x00800111 SCN:0x0000.000491c2 SEQ: 1 **OP:11.3**
KTB Redo
op: 0x11 ver: 0x01
op: F xid: 0x0004.006.00000026 uba: 0x00c02349.0017.05
Block cleanout record, scn: 0x0000.000491c3 ver: 0x01, entries follow...
 itli: 1 flg: 2 scn: 0x0000.000491c2
KDO Op code: DRP xtype: XA bdba: 0x00800111 hdba: 0x00800110
itli: 1 ispac: 0 maxfr: 1177
tabn: 0 slot: 0(0x0)
CHANGE #4 MEDIA RECOVERY MARKER SCN:0x0000.00000000 SEQ: 0 OP:5.19
…
REDO RECORD - Thread:1 RBA: 0x000014.0000001c.0070 LEN: 0x00d8 VLD: 0x01
SCN scn: 0x0000.000491c3 01/31/00 13:50:55
CHANGE #1 TYP:0 CLS:20 AFN:3 DBA:0x00c02349 SCN:0x0000.000491c3 SEQ: 1 OP:5.1

```
ktudb redo: siz: 100 spc: 1256 flg: 0x0022 seq: 0x0017 rec: 0x06
          xid:  0x0004.006.00000026
ktubu redo: slt: 6 rci: 5 opc: 10.22 objn: 12238 objd: 12238 tsn: 1
Undo type:  Regular undo   Last buffer split:  No
Tablespace Undo:  No
          0x00000000
index undo for leaf key operations
KTB Redo
op: 0x04  ver: 0x01
op: L  itl: scn:  0x0003.00f.0000002b uba: 0x00c00134.0019.06
                  flg: C---    lkc:  0     scn: 0x0000.000491c2
Dump kdilk : itl=2, kdxlkflg=0x1 sdc=0 indexid=0x800589 block=0x0080058a
restore leaf row (clear leaf delete flags)
key :(13):  05 53 4d 49 54 48 06 00 80 01 11 00 00
CHANGE #2 TYP:0 CLS: 1 AFN:2 DBA:0x0080058a SCN:0x0000.000491c3 SEQ:  1 OP:10.4
index redo (kdxlde):  delete leaf row
KTB Redo
op: 0x01  ver: 0x01
op: F  xid:  0x0004.006.00000026    uba: 0x00c02349.0017.06
REDO: CLEAR nonkey data flag
itl: 2, sno: 0, row size 17

REDO RECORD - Thread:1 RBA: 0x000014.0000001d.0010 LEN: 0x0090 VLD: 0x01
SCN scn:  0x0000.000491c4 01/31/00 13:50:57
CHANGE #1 TYP:0 CLS: 1 AFN:2 DBA:0x0080058a SCN:0x0000.000491c3 SEQ:  2 OP:10.5
index redo (kdxlre):  restore leaf row
KTB Redo
op: 0x04  ver: 0x01
op: L  itl: scn:  0x0003.00f.0000002b uba: 0x00c00134.0019.06
                  flg: C---    lkc:  0     scn: 0x0000.000491c2
UNDO: SINGLE CLEAR split flag/ -- / --
itl: 2, sno: 0, row size 17
CHANGE #2 TYP:0 CLS:20 AFN:3 DBA:0x00c02349 SCN:0x0000.000491c3 SEQ:  2 OP:5.6
ktubu redo: slt: 6 rci: 5 opc: 10.22 objn: 12238 objd: 12238 tsn: 1
Undo type:  Regular undo   User undo done   Last buffer split:  No
Tablespace Undo:  No
          0x00000000
ktuxvoff: 0x04a0  ktuxvflg: 0x0002

REDO RECORD - Thread:1 RBA: 0x000014.0000001d.00a0 LEN: 0x00dc VLD: 0x01
SCN scn:  0x0000.000491c4 01/31/00 13:50:57
CHANGE #1 TYP:0 CLS: 1 AFN:2 DBA:0x00800111 SCN:0x0000.000491c3 SEQ:  1 OP:11.2
KTB Redo
op: 0x04  ver: 0x01
op: L  itl: scn:  0x0003.00f.0000002b uba: 0x00c00134.0019.05
                  flg: C---    lkc:  0     scn: 0x0000.000491c2
KDO Op code: IRP  xtype: XR  bdba: 0x00800111  hdba: 0x00800110
itli: 1  ispac: 0  maxfr: 1177
tabn: 0 slot: 0(0x0) size/delt: 21
fb: --H-FL-- lb: 0x0 cc: 4
null: ----
```

```
col  0: [ 5]  53 4d 49 54 48
col  1: [ 5]  43 4c 45 52 4b
col  2: [ 3]  c3 03 15
col  3: [ 1]  80
CHANGE #2 TYP:0 CLS:19 AFN:3 DBA:0x00c00160 SCN:0x0000.000491c3 SEQ:  1 OP:5.11
ktubu redo: slt: 6 rci: 0 opc: 11.1 objn: 12236 objd: 12236 tsn: 1
Undo type:  Regular undo   User undo done    Begin trans    Last buffer split:  No
Tablespace Undo:  No
          0x00000000

REDO RECORD - Thread:1 RBA: 0x000014.0000001d.017c LEN: 0x003c VLD: 0x01
SCN scn:  0x0000.000491c5 01/31/00 13:50:57
CHANGE #1 TYP:0 CLS:19 AFN:3 DBA:0x00c00160 SCN:0x0000.000491c4 SEQ:  1 OP:5.4
ktucm redo: slt: 0x0006 sqn: 0x00000026 srt: 0 sta: 9 flg: 0x0
END OF REDO DUMP
```

Lesson 3 Practice Solutions: Recovery Diagnostics

Exercise 1 Solution - Control File dump

Refer to the annotated trace file entitled 'Control file dump' starting on pg. C-5.

Q1 What level was specified in the controlf dump command

A1> Level 2. The generic header, database information, and checkpoint progress records have been dumped, but the reuse record section has not.

Q2 What minimum level should be specified in order to dump out datafile records?

A2> Level 3.

Q3 Has this database been opened with resetlogs? If so, when?

A3> Yes, on 05/26/98 at 13:57:40

Q4 What do the database flags translate to?

A4> Database flags (see notes p.20 for all flags) = 0x00006001 = KCCDIMRE | KCCDISSN | KCCDISSN

```
#define KCCDIMRE 0x0001 /* whether media recovery enabled */
#define KCCDISSN 0x2000 /* SnapShot controlfile fileName pointer valid */
#define KCCDIUCD 0x4000 /* lazy file header Update Checkpoint cycle Done */
```

Q5 When was the database last checkpointed?

A5> Database checkpoint: Thread=1 scn: 0x04d8.fea98ff9

Q6 What is the controlfile's size (in bytes)?

A6> 202*8192=1654784 bytes

From the generic file header
Control Seq=1219=0x4c3, **File size=202=0xca**
File Number=0, **Blksiz=8192**, File Type=1

Exercise 2 Solution- Datafile Header dump

Refer to the highlighted trace file entitled 'Datafile headers dump' starting on pg. C-7.

Q1 What level was used in the file_hdrs dump command?

A1> Level 3 (or above). The datafile headers have been dumped

Q2 What are the statuses of the datafiles, and what does each status translate to? (You may need to reference
the controlfile status as well as the datafile status)

A2> See notes page 26 for all file header status
Datafile #1 File header status=0x1 (KCVFHHBP) so the file is in hot backup mode
Datafiles #2, #3, #5, #6, #8 File header status=0x4 (KCVFHOFZ) so the file is online fuzzy (either the database was open when the file headers were dumped, or is mounted after an unclean shutdown)
Datafiles #4 & #7 File header status=0x0 so the file is offline. The controlfile status for each of these files will tell you whether the datafile is part of a read-only tablespace (file enabled for reading, but not writing) or whether it was offlined 'manually' (file disabled for reading & writing)

Q3 What was the log sequence at the last checkpoint of file #1?

A3> From the checkpoint structure of datafile 1, the log sequence in the rba is 0x8 which indicates log sequence number 8
Checkpointed at scn: 0x04d8.5d15b0e8 06/03/98 15:36:24
thread:1 rba:(0x8.13.10)

Q4 ….and which thread performed the checkpoint?

A4> Thread #1

Q5 How many times has file #2 been checkpointed?

A5> From the checkpoint count in the file header, the file has been checkpointed 187 times
chkpt cnt: 187

Q6 What is the database name to which these files belong?

A6> From any of the generic file headers, Db Name='V804'

Exercise 3 Solution- Redo log header dump

Refer to the trace file entitled 'Redo log headers dump' starting on pg. C-13

Q1 What level was used in the redohdr dump command?

A1> Level 3 (or above). The redo log headers have been dumped

Q2 How many redo threads are in this database?

A2> Two. Look for the number of distinct threads in each of the controlfile's redo log records

Q3 What are the range of log sequences in each thread?

A3> For all redo logs, look in the log header for the seq (or Seq# in the 'descrip' string).
For example (log file #1):
descrip:"Thread 0001, **Seq# 0000000010**, SCN 0x04d8fea98ff9-0xffffffffffff"
thread: 1 nab: 0xffffffff **seq: 0x0000000a** hws: 0x1 eot: 1 dis: 0

Q4 What is the range of SCNs in each redo log?

A4> For all redo logs, look in the redo log header for low scn and next scn fields. A next scn value of ffff.ffffffff (infinity) means that this is the current log for this thread. For example (log file #1):
Low scn: **0x04d8.fea98ff9** 06/05/98 08:39:59
Next scn: **0xffff.ffffffff** 06/03/98 15:24:41

Q5 Identify the two checksum fields in each logfile

A5> The checksum is written on the last line of the log file header dump. For example (log file #1):
Log format vsn: 0x8000000 Disk cksum: **0xeb6f** Calc cksum: **0xeb6f**

Q6 How big is each redo log (in bytes)?

A6> The generic file header will give this. For eample (log file #1):
Control Seq=1206=0x4b6, **File size=8160**=0x1fe0
File Number=1, **Blksiz=512**, File Type=2
Note that the Blksiz is O/S blocks, not Oracle blocks

Control file dump

```
Dump file /oracle/app/oracle/admin/V8041/udump/v8041_ora_12186.trc
Oracle8 Enterprise Edition Release 8.0.4.1.0 - Production
With the Partitioning, Objects and Parallel Server options
PL/SQL Release 8.0.4.1.0 - Production
ORACLE_HOME = /oracle/app/oracle/product/8.0.4
System name:      SunOS
Node name:  uksn116
Release:    5.5.1
Version:    Generic_103640-14
Machine:    sun4u
Instance name: V8041
Redo thread mounted by this instance: 1
Oracle process number: 14
Unix process pid: 12186, image: oracleV8041

Mon Jun  8 10:26:43 1998
*** SESSION ID:(14.851) 1998.06.08.10.26.43.000
DUMP OF CONTROL FILES, Seq # 1219 = 0x4c3
 FILE HEADER:
      Software vsn=134217728=0x8000000, Compatibility Vsn=134217728=0x8000000
      Db Id=3253243709=0xc1e88f3d, Db Name='V804'
      Control Seq=1219=0x4c3, File size=202=0xca
      File Number=0, Blksiz=8192, File Type=1
      DATABASE ENTRY:
       (blkno = 0x1, size = 192, max = 1, in-use = 1, last-recid= 0)
 DF Version: creation=0x8000000 compatible=0x8000000, Date  02/13/98 17:02:53
 DB Name "V804"
 Database flags = 0x00006001
 Controlfile Creation Timestamp  02/13/98 17:02:54
 Incmplt recovery scn: 0x0000.00000000
 Resetlogs scn: 0x04d8.5d142764 Resetlogs Timestamp  05/26/98 13:57:40
 Prior resetlogs scn: 0x04d8.5d142740 Prior resetlogs Timestamp  05/26/98
13:17:58
 Redo Version: creation=0x8000000 compatable=0x8000000
 #Data files = 8, #Online files = 8
 Database checkpoint: Thread=1 scn: 0x04d8.fea98ff9
 Threads: #Enabled=2, #Open=2, Head=1, Tail=2
 enabled  threads:  01100000 00000000 00000000 00000000 00000000 00000000
  00000000 00000000
 Max log members = 2, Max data members = 1
 Arch list: Head=1, Tail=4, Force scn: 0x04d8.fea98ff7
 Snapshot Controlfile filename
  (name #15) /oracle/app/oracle/product/8.0.4/dbs/snapcf_V8041.f
 Snapshot Controlfile checkpoint scn: 0x04d8.5d1562a8 06/02/98 16:50:39
 Controlfile Checkpointed at scn: 0x04d8.fea994d2 01/01/88 00:00:00
 thread:0 rba:(0x0.0.0)
 enabled  threads:  00000000 00000000 00000000 00000000 00000000 00000000
  00000000 00000000
      CHECKPOINT PROGRESS RECORDS:
       (blkno = 0x6, size = 104, max = 8, in-use = 2, last-recid= 2)
THREAD #1 - status:0x2 flags:0x0 dirty:3
low cache rba:(0xa.5.10) on disk rba:(0xa.7.0)
on disk scn: 0x04d8.fea9941f 06/07/98 11:27:39
THREAD #2 - status:0x2 flags:0x0 dirty:16
low cache rba:(0xb.ae.10) on disk rba:(0xb.d0.0)
on disk scn: 0x04d8.fea994f3 06/08/98 09:44:04
THREAD #3 - status:0x0 flags:0x0 dirty:0
low cache rba:(0x0.0.0) on disk rba:(0x0.0.0)
on disk scn: 0x0000.00000000 01/01/88 00:00:00
```

```
THREAD #4 - status:0x0 flags:0x0 dirty:0
low cache rba:(0x0.0.0) on disk rba:(0x0.0.0)
on disk scn: 0x0000.00000000 01/01/88 00:00:00
THREAD #5 - status:0x0 flags:0x0 dirty:0
low cache rba:(0x0.0.0) on disk rba:(0x0.0.0)
on disk scn: 0x0000.00000000 01/01/88 00:00:00
THREAD #6 - status:0x0 flags:0x0 dirty:0
low cache rba:(0x0.0.0) on disk rba:(0x0.0.0)
on disk scn: 0x0000.00000000 01/01/88 00:00:00
THREAD #7 - status:0x0 flags:0x0 dirty:0
low cache rba:(0x0.0.0) on disk rba:(0x0.0.0)
on disk scn: 0x0000.00000000 01/01/88 00:00:00
THREAD #8 - status:0x0 flags:0x0 dirty:0
low cache rba:(0x0.0.0) on disk rba:(0x0.0.0)
on disk scn: 0x0000.00000000 01/01/88 00:00:00
*** END OF DUMP ***
```

Datafile headers dump

```
Dump file /oracle/app/oracle/admin/V8041/udump/v8041_ora_20715.trc
Oracle8 Enterprise Edition Release 8.0.4.1.0 - Production
With the Partitioning, Objects and Parallel Server options
PL/SQL Release 8.0.4.1.0 - Production
ORACLE_HOME = /oracle/app/oracle/product/8.0.4
System name:      SunOS
Node name:  uksn116
Release:    5.5.1
Version:    Generic_103640-14
Machine:    sun4u
Instance name: V8041
Redo thread mounted by this instance: 1
Oracle process number: 14
Unix process pid: 20715, image: oracle

Wed Jun  3 15:37:24 1998
*** SESSION ID:(10.1) 1998.06.03.15.37.24.000
DUMP OF DATA FILES: 8 files in database
DATA FILE #1:
  (name #4) /dev/vx/rdsk/datadg/v804_sys
creation size=6398 block size=8192 status=0xe head=4 tail=4 dup=1
 tablespace 0, index=1 krfil=1 prev_file=0
 unrecoverable scn: 0x0000.00000000 04/15/98 14:24:16
 Checkpoint cnt:198 scn: 0x04d8.5d15b0e8 06/03/98 15:36:24
 Stop scn: 0xffff.ffffffff 05/28/98 18:02:11
 Creation Checkpointed at scn: 0x0000.00000003 02/13/98 17:03:05
 thread:1 rba:(0x1.3.10)
 enabled  threads:  01000000 00000000 00000000 00000000 00000000 00000000
  00000000 00000000
 Offline scn: 0x04d8.5d142763 prev_range: 0
 Online Checkpointed at scn: 0x04d8.5d142764 05/26/98 13:57:40
 thread:1 rba:(0x1.2.0)
 enabled  threads:  01100000 00000000 00000000 00000000 00000000 00000000
  00000000 00000000
 Hot Backup end marker scn: 0x0000.00000000
 FILE HEADER:
      Software vsn=134217728=0x8000000, Compatibility Vsn=134217728=0x8000000
      Db Id=3253243709=0xc1e88f3d, Db Name='V804'
      Control Seq=1170=0x492, File size=6398=0x18fe
      File Number=1, Blksiz=8192, File Type=3
Tablespace #0 - SYSTEM  rel_fn:1
Creation   at   scn: 0x0000.00000003 02/13/98 17:03:05
Backup taken at scn: 0x04d8.5d15b0e8 06/03/98 15:36:24 thread:1
 reset logs count:0x13ed7fd4 scn: 0x04d8.5d142764 recovered at 05/26/98
12:06:49
 status:0x1 root dba:0x004000a9 chkpt cnt: 198 ctl cnt:197
begin-hot-backup file size: 6398
Checkpointed at scn: 0x04d8.5d15b0e8 06/03/98 15:36:24
 thread:1 rba:(0x8.13.10)
 enabled  threads:  01100000 00000000 00000000 00000000 00000000 00000000
  00000000 00000000
Backup Checkpointed at scn: 0x04d8.5d15b0e8 06/03/98 15:36:24
 thread:1 rba:(0x8.13.10)
 enabled  threads:  01100000 00000000 00000000 00000000 00000000 00000000
  00000000 00000000
External cache id: 0x0 0x0 0x0 0x0
Absolute fuzzy scn: 0x0000.00000000
Recovery fuzzy scn: 0x0000.00000000 01/01/88 00:00:00
```

```
DATA FILE #2:
  (name #5) /dev/vx/rdsk/datadg/v804_1_rbs
creation size=5118 block size=8192 status=0xe head=5 tail=5 dup=1
 tablespace 1, index=2 krfil=2 prev_file=0
 unrecoverable scn: 0x0000.00000000 01/01/88 00:00:00
 Checkpoint cnt:187 scn: 0x04d8.5d156282 05/29/98 12:42:08
 Stop scn: 0xffff.ffffffff 05/28/98 18:02:11
 Creation Checkpointed at scn: 0x0000.00002005 02/13/98 17:14:37
 thread:1 rba:(0x7.1576.bc)
 enabled  threads:  01000000 00000000 00000000 00000000 00000000 00000000
  00000000 00000000
 Offline scn: 0x04d8.5d142763 prev_range: 0
 Online Checkpointed at scn: 0x04d8.5d142764 05/26/98 13:57:40
 thread:1 rba:(0x1.2.0)
 enabled  threads:  01100000 00000000 00000000 00000000 00000000 00000000
  00000000 00000000
 Hot Backup end marker scn: 0x0000.00000000
 FILE HEADER:
      Software vsn=134217728=0x8000000, Compatibility Vsn=134217728=0x8000000
      Db Id=3253243709=0xc1e88f3d, Db Name='V804'
      Control Seq=1121=0x461, File size=5118=0x13fe
      File Number=2, Blksiz=8192, File Type=3
Tablespace #1 - RBS1  rel_fn:2
Creation   at   scn: 0x0000.00002005 02/13/98 17:14:37
Backup taken at scn: 0x0000.00000000 01/01/88 00:00:00 thread:0
 reset logs count:0x13ed7fd4 scn: 0x04d8.5d142764 recovered at 05/26/98
12:06:49
 status:0x4 root dba:0x00000000 chkpt cnt: 187 ctl cnt:186
begin-hot-backup file size: 0
Checkpointed at scn: 0x04d8.5d156282 05/29/98 12:42:08
 thread:1 rba:(0x7.2.10)
 enabled  threads:  01100000 00000000 00000000 00000000 00000000 00000000
  00000000 00000000
Backup Checkpointed at scn: 0x0000.00000000 01/01/88 00:00:00
 thread:0 rba:(0x0.0.0)
 enabled  threads:  00000000 00000000 00000000 00000000 00000000 00000000
  00000000 00000000
External cache id: 0x0 0x0 0x0 0x0
Absolute fuzzy scn: 0x0000.00000000
Recovery fuzzy scn: 0x0000.00000000 01/01/88 00:00:00
DATA FILE #3:
  (name #6) /dev/vx/rdsk/datadg/v804_2_rbs
creation size=5118 block size=8192 status=0xe head=6 tail=6 dup=1
 tablespace 2, index=3 krfil=3 prev_file=0
 unrecoverable scn: 0x0000.00000000 01/01/88 00:00:00
 Checkpoint cnt:185 scn: 0x04d8.5d15629f 06/01/98 10:29:41
 Stop scn: 0xffff.ffffffff 05/28/98 18:02:11
 Creation Checkpointed at scn: 0x0000.0000200b 02/13/98 17:14:43
 thread:1 rba:(0x7.1581.bc)
 enabled  threads:  01000000 00000000 00000000 00000000 00000000 00000000
  00000000 00000000
 Offline scn: 0x04d8.5d142763 prev_range: 0
 Online Checkpointed at scn: 0x04d8.5d142764 05/26/98 13:57:40
 thread:1 rba:(0x1.2.0)
 enabled  threads:  01100000 00000000 00000000 00000000 00000000 00000000
  00000000 00000000
 Hot Backup end marker scn: 0x0000.00000000
 FILE HEADER:
      Software vsn=134217728=0x8000000, Compatibility Vsn=134217728=0x8000000
      Db Id=3253243709=0xc1e88f3d, Db Name='V804'
      Control Seq=1146=0x47a, File size=5118=0x13fe
      File Number=3, Blksiz=8192, File Type=3
```

```
Tablespace #2 - RBS2  rel_fn:3
Creation   at   scn: 0x0000.0000200b 02/13/98 17:14:43
Backup taken at scn: 0x0000.00000000 01/01/88 00:00:00 thread:0
 reset logs count:0x13ed7fd4 scn: 0x04d8.5d142764 recovered at 05/26/98
12:06:49
 status:0x4 root dba:0x00000000 chkpt cnt: 185 ctl cnt:184
begin-hot-backup file size: 0
Checkpointed at scn: 0x04d8.5d15629f 06/01/98 10:29:41
 thread:1 rba:(0x7.24.10)
 enabled  threads:  01100000 00000000 00000000 00000000 00000000 00000000
  00000000 00000000
Backup Checkpointed at scn: 0x0000.00000000 01/01/88 00:00:00
 thread:0 rba:(0x0.0.0)
 enabled  threads:  00000000 00000000 00000000 00000000 00000000 00000000
  00000000 00000000
External cache id: 0x0 0x0 0x0 0x0
Absolute fuzzy scn: 0x0000.00000000
Recovery fuzzy scn: 0x0000.00000000 01/01/88 00:00:00
DATA FILE #4:
  (name #7) /dev/vx/rdsk/datadg/v804_1_temp
creation size=5118 block size=8192 status=0x80 head=7 tail=7 dup=1
 tablespace 3, index=4 krfil=4 prev_file=0
 unrecoverable scn: 0x0000.00000000 01/01/88 00:00:00
 Checkpoint cnt:193 scn: 0x04d8.5d15b0ef 06/03/98 15:36:45
 Stop scn: 0x04d8.5d15b0ef 06/03/98 15:36:45
 Creation Checkpointed at scn: 0x0000.00002012 02/13/98 17:14:50
 thread:1 rba:(0x7.1592.c0)
 enabled  threads:  01000000 00000000 00000000 00000000 00000000 00000000
  00000000 00000000
 Offline scn: 0x04d8.5d15629c prev_range: 2
 Online Checkpointed at scn: 0x04d8.5d1562a8 06/02/98 16:50:39
 thread:1 rba:(0x7.26.70)
 enabled  threads:  01100000 00000000 00000000 00000000 00000000 00000000
  00000000 00000000
 Hot Backup end marker scn: 0x0000.00000000
 FILE HEADER:
      Software vsn=134217728=0x8000000, Compatibility Vsn=134217728=0x8000000
      Db Id=3253243709=0xc1e88f3d, Db Name='V804'
      Control Seq=1172=0x494, File size=5118=0x13fe
      File Number=4, Blksiz=8192, File Type=3
Tablespace #3 - TEMP1  rel_fn:4
Creation   at   scn: 0x0000.00002012 02/13/98 17:14:50
Backup taken at scn: 0x0000.00000000 01/01/88 00:00:00 thread:0
 reset logs count:0x13ed7fd4 scn: 0x04d8.5d142764 recovered at 06/01/98
08:24:04
 status:0x0 root dba:0x00000000 chkpt cnt: 193 ctl cnt:192
begin-hot-backup file size: 0
Checkpointed at scn: 0x04d8.5d15b0ef 06/03/98 15:36:45
 thread:1 rba:(0x8.5f.130)
 enabled  threads:  01100000 00000000 00000000 00000000 00000000 00000000
  00000000 00000000
Backup Checkpointed at scn: 0x0000.00000000 01/01/88 00:00:00
 thread:0 rba:(0x0.0.0)
 enabled  threads:  00000000 00000000 00000000 00000000 00000000 00000000
  00000000 00000000
External cache id: 0x0 0x0 0x0 0x0
Absolute fuzzy scn: 0x0000.00000000
Recovery fuzzy scn: 0x0000.00000000 06/01/98 08:19:23
DATA FILE #5:
  (name #8) /dev/vx/rdsk/datadg/v804_2_temp
creation size=5118 block size=8192 status=0xe head=8 tail=8 dup=1
```

```
 tablespace 4, index=5 krfil=5 prev_file=0
 unrecoverable scn: 0x0000.00000000 01/01/88 00:00:00
 Checkpoint cnt:183 scn: 0x04d8.5d156282 05/29/98 12:42:08
 Stop scn: 0xffff.ffffffff 05/28/98 18:02:11
 Creation Checkpointed at scn: 0x0000.00002018 02/13/98 17:14:56
 thread:1 rba:(0x7.159d.c0)
 enabled  threads:  01000000 00000000 00000000 00000000 00000000 00000000
  00000000 00000000
 Offline scn: 0x04d8.5d142763 prev_range: 0
 Online Checkpointed at scn: 0x04d8.5d142764 05/26/98 13:57:40
 thread:1 rba:(0x1.2.0)
 enabled  threads:  01100000 00000000 00000000 00000000 00000000 00000000
  00000000 00000000
 Hot Backup end marker scn: 0x0000.00000000
 FILE HEADER:
      Software vsn=134217728=0x8000000, Compatibility Vsn=134217728=0x8000000
      Db Id=3253243709=0xc1e88f3d, Db Name='V804'
      Control Seq=1121=0x461, File size=5118=0x13fe
      File Number=5, Blksiz=8192, File Type=3
Tablespace #4 - TEMP2  rel_fn:5
Creation   at   scn: 0x0000.00002018 02/13/98 17:14:56
Backup taken at scn: 0x0000.00000000 01/01/88 00:00:00 thread:0
 reset logs count:0x13ed7fd4 scn: 0x04d8.5d142764 recovered at 05/26/98
12:06:49
 status:0x4 root dba:0x00000000 chkpt cnt: 183 ctl cnt:182
begin-hot-backup file size: 0
Checkpointed at scn: 0x04d8.5d156282 05/29/98 12:42:08
 thread:1 rba:(0x7.2.10)
 enabled  threads:  01100000 00000000 00000000 00000000 00000000 00000000
  00000000 00000000
Backup Checkpointed at scn: 0x0000.00000000 01/01/88 00:00:00
 thread:0 rba:(0x0.0.0)
 enabled  threads:  00000000 00000000 00000000 00000000 00000000 00000000
  00000000 00000000
External cache id: 0x0 0x0 0x0 0x0
Absolute fuzzy scn: 0x0000.00000000
Recovery fuzzy scn: 0x0000.00000000 01/01/88 00:00:00
DATA FILE #6:
  (name #9) /dev/vx/rdsk/datadg/v804_tools
creation size=2558 block size=8192 status=0xe head=9 tail=9 dup=1
 tablespace 5, index=6 krfil=6 prev_file=0
 unrecoverable scn: 0x0000.00000000 04/01/98 13:04:55
 Checkpoint cnt:183 scn: 0x04d8.5d156282 05/29/98 12:42:08
 Stop scn: 0xffff.ffffffff 05/28/98 18:02:11
 Creation Checkpointed at scn: 0x0000.0000201e 02/13/98 17:14:59
 thread:1 rba:(0x7.15a8.c0)
 enabled  threads:  01000000 00000000 00000000 00000000 00000000 00000000
  00000000 00000000
 Offline scn: 0x04d8.5d142763 prev_range: 0
 Online Checkpointed at scn: 0x04d8.5d142764 05/26/98 13:57:40
 thread:1 rba:(0x1.2.0)
 enabled  threads:  01100000 00000000 00000000 00000000 00000000 00000000
  00000000 00000000
 Hot Backup end marker scn: 0x0000.00000000
 FILE HEADER:
      Software vsn=134217728=0x8000000, Compatibility Vsn=134217728=0x8000000
      Db Id=3253243709=0xc1e88f3d, Db Name='V804'
      Control Seq=1121=0x461, File size=2558=0x9fe
      File Number=6, Blksiz=8192, File Type=3
Tablespace #5 - TOOLS  rel_fn:6
Creation   at   scn: 0x0000.0000201e 02/13/98 17:14:59
Backup taken at scn: 0x0000.00000000 01/01/88 00:00:00 thread:0
```

```
 reset logs count:0x13ed7fd4 scn: 0x04d8.5d142764 recovered at 05/26/98
12:06:49
 status:0x4 root dba:0x00000000 chkpt cnt: 183 ctl cnt:182
begin-hot-backup file size: 0
Checkpointed at scn: 0x04d8.5d156282 05/29/98 12:42:08
 thread:1 rba:(0x7.2.10)
 enabled  threads:  01100000 00000000 00000000 00000000 00000000 00000000
  00000000 00000000
Backup Checkpointed at scn: 0x0000.00000000 01/01/88 00:00:00
 thread:0 rba:(0x0.0.0)
 enabled  threads:  00000000 00000000 00000000 00000000 00000000 00000000
  00000000 00000000
External cache id: 0x0 0x0 0x0 0x0
Absolute fuzzy scn: 0x0000.00000000
Recovery fuzzy scn: 0x0000.00000000 01/01/88 00:00:00
DATA FILE #7:
  (name #10) /dev/vx/rdsk/datadg/v804_users
creation size=2558 block size=8192 status=0x86 head=10 tail=10 dup=1
 tablespace 6, index=7 krfil=7 prev_file=0
 unrecoverable scn: 0x0000.00000000 03/25/98 15:30:48
 Checkpoint cnt:188 scn: 0x04d8.5d15b0f1 06/03/98 15:36:57
 Stop scn: 0x04d8.5d15b0f1 06/03/98 15:36:57
 Creation Checkpointed at scn: 0x0000.00002025 02/13/98 17:15:03
 thread:1 rba:(0x7.15ba.c0)
 enabled  threads:  01000000 00000000 00000000 00000000 00000000 00000000
  00000000 00000000
 Offline scn: 0x04d8.5d142763 prev_range: 0
 Online Checkpointed at scn: 0x04d8.5d142764 05/26/98 13:57:40
 thread:1 rba:(0x1.2.0)
 enabled  threads:  01100000 00000000 00000000 00000000 00000000 00000000
  00000000 00000000
 Hot Backup end marker scn: 0x0000.00000000
 FILE HEADER:
      Software vsn=134217728=0x8000000, Compatibility Vsn=134217728=0x8000000
      Db Id=3253243709=0xc1e88f3d, Db Name='V804'
      Control Seq=1175=0x497, File size=2558=0x9fe
      File Number=7, Blksiz=8192, File Type=3
Tablespace #6 - USERS  rel_fn:7
Creation   at   scn: 0x0000.00002025 02/13/98 17:15:03
Backup taken at scn: 0x0000.00000000 01/01/88 00:00:00 thread:0
 reset logs count:0x13ed7fd4 scn: 0x04d8.5d142764 recovered at 05/29/98
11:16:49
 status:0x0 root dba:0x00000000 chkpt cnt: 188 ctl cnt:187
begin-hot-backup file size: 0
Checkpointed at scn: 0x04d8.5d15b0f1 06/03/98 15:36:57
 thread:1 rba:(0x8.74.178)
 enabled  threads:  01100000 00000000 00000000 00000000 00000000 00000000
  00000000 00000000
Backup Checkpointed at scn: 0x0000.00000000 01/01/88 00:00:00
 thread:0 rba:(0x0.0.0)
 enabled  threads:  00000000 00000000 00000000 00000000 00000000 00000000
  00000000 00000000
External cache id: 0x0 0x0 0x0 0x0
Absolute fuzzy scn: 0x0000.00000000
Recovery fuzzy scn: 0x0000.00000000 05/29/98 11:16:39
DATA FILE #8:
  (name #14) /dev/vx/rdsk/datadg/v804_rcvcat
creation size=2558 block size=8192 status=0xe head=14 tail=14 dup=1
 tablespace 7, index=8 krfil=8 prev_file=0
 unrecoverable scn: 0x0000.00000000 01/01/88 00:00:00
 Checkpoint cnt:173 scn: 0x04d8.5d156282 05/29/98 12:42:08
 Stop scn: 0xffff.ffffffff 05/28/98 18:02:11
```

```
 Creation Checkpointed at scn: 0x0000.00002aad 02/17/98 16:39:05
 thread:1 rba:(0xd.1552.c0)
 enabled  threads:  01100000 00000000 00000000 00000000 00000000 00000000
  00000000 00000000
 Offline scn: 0x04d8.5d142763 prev_range: 0
 Online Checkpointed at scn: 0x04d8.5d142764 05/26/98 13:57:40
 thread:1 rba:(0x1.2.0)
 enabled  threads:  01100000 00000000 00000000 00000000 00000000 00000000
  00000000 00000000
 Hot Backup end marker scn: 0x0000.00000000
 FILE HEADER:
      Software vsn=134217728=0x8000000, Compatibility Vsn=134217728=0x8000000
      Db Id=3253243709=0xc1e88f3d, Db Name='V804'
      Control Seq=1121=0x461, File size=2558=0x9fe
      File Number=8, Blksiz=8192, File Type=3
Tablespace #7 - RCVCAT  rel_fn:8
Creation   at   scn: 0x0000.00002aad 02/17/98 16:39:05
Backup taken at scn: 0x0000.00000000 01/01/88 00:00:00 thread:0
 reset logs count:0x13ed7fd4 scn: 0x04d8.5d142764 recovered at 05/26/98
12:06:49
 status:0x4 root dba:0x00000000 chkpt cnt: 173 ctl cnt:172
begin-hot-backup file size: 0
Checkpointed at scn: 0x04d8.5d156282 05/29/98 12:42:08
 thread:1 rba:(0x7.2.10)
 enabled  threads:  01100000 00000000 00000000 00000000 00000000 00000000
  00000000 00000000
Backup Checkpointed at scn: 0x0000.00000000 01/01/88 00:00:00
 thread:0 rba:(0x0.0.0)
 enabled  threads:  00000000 00000000 00000000 00000000 00000000 00000000
  00000000 00000000
External cache id: 0x0 0x0 0x0 0x0
Absolute fuzzy scn: 0x0000.00000000
Recovery fuzzy scn: 0x0000.00000000 01/01/88 00:00:00
```

Redo log headers dump

```
Dump file /oracle/app/oracle/admin/V8041/udump/v8041_ora_12282.trc
Oracle8 Enterprise Edition Release 8.0.4.1.0 - Production
With the Partitioning, Objects and Parallel Server options
PL/SQL Release 8.0.4.1.0 - Production
ORACLE_HOME = /oracle/app/oracle/product/8.0.4
System name:      SunOS
Node name:  uksn116
Release:    5.5.1
Version:    Generic_103640-14
Machine:    sun4u
Instance name: V8041
Redo thread mounted by this instance: 1
Oracle process number: 14
Unix process pid: 12282, image: oracleV8041

Mon Jun  8 10:33:26 1998
*** SESSION ID:(14.855) 1998.06.08.10.33.26.000
DUMP OF LOG FILES: 6 logs in database
LOG FILE #1:
  (name #1) /dev/vx/rdsk/datadg/v804_1_redo_1
 Thread 1 redo log links: forward: 2 backward: 0
 siz: 0x1fe0 seq: 0x0000000a hws: 0x1 bsz: 512 nab: 0xffffffff flg: 0x8 dup: 1
 Archive links: fwrd: 4 back: 0 Prev scn: 0x04d8.fea9885c
 Low scn: 0x04d8.fea98ff9 06/05/98 08:39:59
 Next scn: 0xffff.ffffffff 06/03/98 15:24:41
 FILE HEADER:
      Software vsn=134217728=0x8000000, Compatibility Vsn=134217728=0x8000000
      Db Id=3253243709=0xc1e88f3d, Db Name='V804'
      Control Seq=1206=0x4b6, File size=8160=0x1fe0
      File Number=1, Blksiz=512, File Type=2
 descrip:"Thread 0001, Seq# 0000000010, SCN 0x04d8fea98ff9-0xffffffffffff"
 thread: 1 nab: 0xffffffff seq: 0x0000000a hws: 0x1 eot: 1 dis: 0
 reset logs count: 0x13ed7fd4 scn: 0x04d8.5d142764
 Low scn: 0x04d8.fea98ff9 06/05/98 08:39:59
 Next scn: 0xffff.ffffffff 06/03/98 15:24:41
 Enabled scn: 0x04d8.5d142764 05/26/98 13:57:40
 Thread closed scn: 0x04d8.fea98ff9 06/05/98 08:39:59
 Log format vsn: 0x8000000 Disk cksum: 0xeb6f Calc cksum: 0xeb6f
LOG FILE #2:
  (name #2) /dev/vx/rdsk/datadg/v804_1_redo_2
 Thread 1 redo log links: forward: 3 backward: 1
 siz: 0x1fe0 seq: 0x00000008 hws: 0x4 bsz: 512 nab: 0x98 flg: 0x1 dup: 1
 Archive links: fwrd: 0 back: 0 Prev scn: 0x04d8.5d156282
 Low scn: 0x04d8.5d15b0de 06/03/98 15:24:41
 Next scn: 0x04d8.fea9885c 06/03/98 16:19:56
 FILE HEADER:
      Software vsn=134217728=0x8000000, Compatibility Vsn=134217728=0x8000000
      Db Id=3253243709=0xc1e88f3d, Db Name='V804'
      Control Seq=1192=0x4a8, File size=8160=0x1fe0
      File Number=2, Blksiz=512, File Type=2
 descrip:"Thread 0001, Seq# 0000000008, SCN 0x04d85d15b0de-0x04d8fea9885c"
 thread: 1 nab: 0x98 seq: 0x00000008 hws: 0x4 eot: 0 dis: 0
 reset logs count: 0x13ed7fd4 scn: 0x04d8.5d142764
 Low scn: 0x04d8.5d15b0de 06/03/98 15:24:41
 Next scn: 0x04d8.fea9885c 06/03/98 16:19:56
 Enabled scn: 0x04d8.5d142764 05/26/98 13:57:40
 Thread closed scn: 0x04d8.5d15b0de 06/03/98 15:24:41
 Log format vsn: 0x8000000 Disk cksum: 0xd7a4 Calc cksum: 0xd7a4
```

```
LOG FILE #3:
  (name #3) /dev/vx/rdsk/datadg/v804_1_redo_3
 Thread 1 redo log links: forward: 0 backward: 2
 siz: 0x1fe0 seq: 0x00000009 hws: 0x5 bsz: 512 nab: 0x18f flg: 0x1 dup: 1
 Archive links: fwrd: 0 back: 0 Prev scn: 0x04d8.5d15b0de
 Low scn: 0x04d8.fea9885c 06/03/98 16:19:56
 Next scn: 0x04d8.fea98ff9 06/05/98 08:39:59
 FILE HEADER:
      Software vsn=134217728=0x8000000, Compatibility Vsn=134217728=0x8000000
      Db Id=3253243709=0xc1e88f3d, Db Name='V804'
      Control Seq=1206=0x4b6, File size=8160=0x1fe0
      File Number=3, Blksiz=512, File Type=2
 descrip:"Thread 0001, Seq# 0000000009, SCN 0x04d8fea9885c-0x04d8fea98ff9"
 thread: 1 nab: 0x18f seq: 0x00000009 hws: 0x5 eot: 0 dis: 0
 reset logs count: 0x13ed7fd4 scn: 0x04d8.5d142764
 Low scn: 0x04d8.fea9885c 06/03/98 16:19:56
 Next scn: 0x04d8.fea98ff9 06/05/98 08:39:59
 Enabled scn: 0x04d8.5d142764 05/26/98 13:57:40
 Thread closed scn: 0x04d8.fea98863 06/04/98 14:17:38
 Log format vsn: 0x8000000 Disk cksum: 0xd1d2 Calc cksum: 0xd1d2
LOG FILE #4:
  (name #11) /dev/vx/rdsk/datadg/v804_2_redo_1
 Thread 2 redo log links: forward: 5 backward: 0
 siz: 0x1fe0 seq: 0x0000000b hws: 0x1 bsz: 512 nab: 0xffffffff flg: 0x8 dup: 1
 Archive links: fwrd: 0 back: 1 Prev scn: 0x04d8.fea9924b
 Low scn: 0x04d8.fea994b9 06/08/98 09:24:35
 Next scn: 0xffff.ffffffff 06/05/98 08:40:24
 FILE HEADER:
      Software vsn=134217728=0x8000000, Compatibility Vsn=134217728=0x8000000
      Db Id=3253243709=0xc1e88f3d, Db Name='V804'
      Control Seq=1216=0x4c0, File size=8160=0x1fe0
      File Number=4, Blksiz=512, File Type=2
 descrip:"Thread 0002, Seq# 0000000011, SCN 0x04d8fea994b9-0xffffffffffff"
 thread: 2 nab: 0xffffffff seq: 0x0000000b hws: 0x1 eot: 1 dis: 0
 reset logs count: 0x13ed7fd4 scn: 0x04d8.5d142764
 Low scn: 0x04d8.fea994b9 06/08/98 09:24:35
 Next scn: 0xffff.ffffffff 06/05/98 08:40:24
 Enabled scn: 0x04d8.5d142764 05/26/98 13:57:40
 Thread closed scn: 0x04d8.fea994b9 06/08/98 09:24:35
 Log format vsn: 0x8000000 Disk cksum: 0xae06 Calc cksum: 0xae06
LOG FILE #5:
  (name #12) /dev/vx/rdsk/datadg/v804_2_redo_2
 Thread 2 redo log links: forward: 6 backward: 4
 siz: 0x1fe0 seq: 0x00000009 hws: 0x2 bsz: 512 nab: 0x1fdf flg: 0x1 dup: 1
 Archive links: fwrd: 0 back: 0 Prev scn: 0x04d8.fea98b50
 Low scn: 0x04d8.fea98ff7 06/05/98 08:40:24
 Next scn: 0x04d8.fea9924b 06/05/98 08:40:55
 FILE HEADER:
      Software vsn=134217728=0x8000000, Compatibility Vsn=134217728=0x8000000
      Db Id=3253243709=0xc1e88f3d, Db Name='V804'
      Control Seq=1212=0x4bc, File size=8160=0x1fe0
      File Number=5, Blksiz=512, File Type=2
 descrip:"Thread 0002, Seq# 0000000009, SCN 0x04d8fea98ff7-0x04d8fea9924b"
 thread: 2 nab: 0x1fdf seq: 0x00000009 hws: 0x2 eot: 0 dis: 0
 reset logs count: 0x13ed7fd4 scn: 0x04d8.5d142764
 Low scn: 0x04d8.fea98ff7 06/05/98 08:40:24
 Next scn: 0x04d8.fea9924b 06/05/98 08:40:55
 Enabled scn: 0x04d8.5d142764 05/26/98 13:57:40
 Thread closed scn: 0x04d8.fea98ff7 06/05/98 08:40:24
 Log format vsn: 0x8000000 Disk cksum: 0xc2a1 Calc cksum: 0xc2a1
```

```
LOG FILE #6:
  (name #13) /dev/vx/rdsk/datadg/v804_2_redo_3
 Thread 2 redo log links: forward: 0 backward: 5
 siz: 0x1fe0 seq: 0x0000000a hws: 0x2 bsz: 512 nab: 0x1fe1 flg: 0x1 dup: 1
 Archive links: fwrd: 0 back: 0 Prev scn: 0x04d8.fea98ff7
 Low scn: 0x04d8.fea9924b 06/05/98 08:40:55
 Next scn: 0x04d8.fea994b9 06/08/98 09:24:35
 FILE HEADER:
      Software vsn=134217728=0x8000000, Compatibility Vsn=134217728=0x8000000
      Db Id=3253243709=0xc1e88f3d, Db Name='V804'
      Control Seq=1216=0x4c0, File size=8160=0x1fe0
      File Number=6, Blksiz=512, File Type=2
 descrip:"Thread 0002, Seq# 0000000010, SCN 0x04d8fea9924b-0x04d8fea994b9"
 thread: 2 nab: 0x1fe1 seq: 0x0000000a hws: 0x2 eot: 0 dis: 0
 reset logs count: 0x13ed7fd4 scn: 0x04d8.5d142764
 Low scn: 0x04d8.fea9924b 06/05/98 08:40:55
 Next scn: 0x04d8.fea994b9 06/08/98 09:24:35
 Enabled scn: 0x04d8.5d142764 05/26/98 13:57:40
 Thread closed scn: 0x04d8.fea9924b 06/05/98 08:40:55
 Log format vsn: 0x8000000 Disk cksum: 0xcb4f Calc cksum: 0xcb4f
```

Lesson 4 Practice Solutions: Troubleshooting Recovery Operations

Exercise 1 Solution

1. Run the shell script $HOME/LABS/case1.sh

```
SVRMGR> Connected.
SVRMGR> ORACLE instance started.
Total System Global Area                          6311312 bytes
Fixed Size                                          64912 bytes
Variable Size                                     5304320 bytes
Database Buffers                                   409600 bytes
Redo Buffers                                       532480 bytes
Database mounted.
Database opened.
SVRMGR> Database closed.
Database dismounted.
ORACLE instance shut down.
SVRMGR> ORACLE instance started.
Total System Global Area                          6311312 bytes
Fixed Size                                          64912 bytes
Variable Size                                     5304320 bytes
Database Buffers                                   409600 bytes
Redo Buffers                                       532480 bytes
Database mounted.
Database opened.
SVRMGR> Statement processed.
SVRMGR> Statement processed.
SVRMGR> ORACLE instance shut down.
SVRMGR>
Server Manager complete.
```

2. Startup mount the database and take a file header dump.

Alter session set events 'immediate trace name file_hdrs level XX' ;

```
SQL*Plus: Release 8.1.5.0.0 - Production on Mon Jan 31 20:33:43 2000
(c) Copyright 1999 Oracle Corporation.  All rights reserved.

SQL> connect internal
Connected to an idle instance.

SQL> startup
ORACLE instance started.

Total System Global Area    6311312 bytes
Fixed Size                    64912 bytes
Variable Size               5304320 bytes
Database Buffers             409600 bytes
Redo Buffers                 532480 bytes
Database mounted.
ORA-01113: file 1 needs media recovery
ORA-01110: data file 1: '/home/DATA/system01.dbf'


SQL> alter session set events 'immediate trace name file_hdrs level 10';

Session altered.

SQL>
```

3. Explain the status of each of the datafiles, how could they have ended in this state? Confirm your observations with V$ view entries.

```
DATA FILE #1:
  (name #10) /home/db01/DATA/system01.dbf
creation size=0 block size=4096 status=0xe head=10 tail=10 dup=1
 tablespace 0, index=1 krfil=1 prev_file=0
....
Hot Backup end marker scn: 0x0000.00000000
 FILE HEADER:
        Software vsn=135286784=0x8105000, Compatibility
Vsn=134217728=0x8000000
        Db Id=384896786=0x16f10f12, Db Name='DB01'
        Control Seq=198=0xc6, File size=12800=0x3200
        File Number=1, Blksiz=4096, File Type=3 DATA
Tablespace #0 - SYSTEM  rel_fn:1
Creation   at   scn: 0x0000.00000004 06/24/99 13:33:12
Backup taken at scn: 0x0000.00029878 01/31/00 20:29:19 thread:1
 reset logs count:0x172553cf scn: 0x0000.0000c33a recovered at 01/31/00
20:27:55
 status:0x1 root dba:0x00400123 chkpt cnt: 63 ctl cnt:62
begin-hot-backup file size: 12800
Checkpointed at scn: 0x0000.00029878 01/31/00 20:29:19
 thread:1 rba:(0x7.11.88)
 enabled  threads:  01000000 00000000 00000000 00000000 00000000 00000000
  00000000 00000000
Backup Checkpointed at scn: 0x0000.00029878 01/31/00 20:29:19
 thread:1 rba:(0x7.11.88)
```

...

FILE 1 is in HOT BACKUP mode. The status in the file header is set to `0x1`, there is a value in the begin-hot-backup file size (`12800`), and the Backup Checkpointed at scn field is set `(0x0000.00029878 01/31/00 09:44:12 thread:1 rba:(0x7.11.88)`). The status from the controlfile is `0xe` (online, read, write), note that system datafiles no longer have an additional bit set for them (in version 7, the status from the controlfile would have read `0xf`)

```
DATA FILE #2:
  (name #9) /home/db01/DATA/rbs01.dbf
creation size=0 block size=4096 status=0xe head=9 tail=9 dup=1
 tablespace 1, index=2 krfil=2 prev_file=0
 unrecoverable scn: 0x0000.00000000 01/01/88 00:00:00
 Checkpoint cnt:58 scn: 0x0000.00029872 01/31/00 20:29:17
 Stop scn: 0xffff.ffffffff 01/31/00 20:29:04
…
 Hot Backup end marker scn: 0x0000.00000000
 FILE HEADER:
        Software vsn=135286784=0x8105000, Compatibility
Vsn=134217728=0x8000000
        Db Id=384896786=0x16f10f12, Db Name='DB01'
        Control Seq=197=0xc5, File size=3840=0xf00
        File Number=2, Blksiz=4096, File Type=3 DATA
Tablespace #1 - RBS  rel_fn:2
Creation   at   scn: 0x0000.00000f3e 06/24/99 13:34:18
Backup taken at scn: 0x0000.00000000 01/01/88 00:00:00 thread:0
 reset logs count:0x172553cf scn: 0x0000.0000c33a recovered at 01/31/00
20:27:55
 status:0x4 root dba:0x00000000 chkpt cnt: 58 ctl cnt:57
begin-hot-backup file size: 0
Checkpointed at scn: 0x0000.00029872 01/31/00 20:29:17
```

File 2 is online, read, write (from the controlfile status `0xe`) and online fuzzy (from the datafile status `0x4`. File is open and in normal operation.

```
DATA FILE #3:
  (name #8) /home/db01/DATA/users01.dbf
creation size=12800 block size=4096 status=0x10 head=8 tail=8 dup=1
 tablespace 7, index=3 krfil=3 prev_file=0
unrecoverable scn: 0x0000.00000000 01/01/88 00:00:00
 Checkpoint cnt:7 scn: 0x0000.00029872 01/31/00 20:29:17
 Stop scn: 0x0000.00029880 01/31/00 20:29:20
 Creation Checkpointed at scn:  0x0000.00024a41 01/31/00 20:27:07
 thread:1 rba:(0x6.18.1c)
 enabled  threads:  01000000 00000000 00000000 00000000 00000000 00000000
  00000000 00000000
 Offline scn: 0x0000.00000000 prev_range: 0
 Online Checkpointed at scn:  0x0000.00000000
 ….
FILE HEADER:
    Software vsn=135286784=0x8105000, Compatibility Vsn=134217728=0x8000000
    Db Id=384896786=0x16f10f12, Db Name='DB01'
    Control Seq=197=0xc5, File size=12800=0x3200
    File Number=3, Blksiz=4096, File Type=3 DATA
Tablespace #7 - USERS  rel_fn:3
Creation   at   scn: 0x0000.00024a41 01/31/00 20:27:07
Backup taken at scn: 0x0000.00000000 01/01/88 00:00:00 thread:0
 reset logs count:0x172553cf scn: 0x0000.0000c33a recovered at 01/31/00
20:27:55
 status:0x4 root dba:0x00000000 chkpt cnt: 7 ctl cnt:6
begin-hot-backup file size: 0
Checkpointed at scn:  0x0000.00029872 01/31/00 20:29:17
 thread:1 rba:(0x7.c.10)
 enabled  threads:  01000000 00000000 00000000 00000000 00000000 00000000
  00000000 00000000
Backup Checkpointed at scn:  0x0000.00000000
 thread:0 rba:(0x0.0.0)
…
```

File 4 has a status in the controlfile portion of `0x10` (Media recovery required), and a status in the datafile header status of `0x4` (online fuzzy [normal]). The **checkpoint count** in the controlfile is

104 and this matches the **chkpt cnt** in the datafile portion. The checkpoint scns match as well (`0x0000.00029872`). Why does the file need media recovery? The stop SCN is set to a finite value (`0x0000.00029880`), this is true only when a tablespace is taken offline normal or the database is shutdown. The STOP SCN indicates the highest value change made in this datafile. The file is checkpointed at this SCN. Looking at the file header shows an earlier checkpoint value (`0x0000.00029872`), and the file is still online fuzzy indicating that changes are still expected to be made. This is indicative of a file that has been taken offline without checkpointing (ie. Offline Immediate). When a file is taken offline immediately, any datablocks currently dirtied within the buffer cache are not flushed to disk. These changes have to be reconstructed from redo in the redo logs, and as we don't know when this will happen, the file is flagged as needing media recovery, therefore redo may come from online or archived redo logs.

Query V$BACKUP and V$DATAFILE to confirm findings

```
SQL> alter database open;
alter database open
*
ORA-01113: file 1 needs media recovery
ORA-01110: data file 1: '/home/db01/DATA/system01.dbf'

SQL> select * from v$backup;

     FILE# STATUS                CHANGE# TIME
---------- ------------------ ---------- ---------
         1 ACTIVE                 170104 31-JAN-00
         2 NOT ACTIVE                  0
         4 NOT ACTIVE                  0
         5 NOT ACTIVE                  0
         6 NOT ACTIVE                  0
         7 NOT ACTIVE                  0

SQL> alter database datafile '/home/db01/DATA/system01.dbf'
     2> end backup;
Database altered.
SQL> alter database open;
Database altered.
SQL> select file#, ts#, status, enabled from v$datafile;

     FILE#        TS# STATUS  ENABLED
---------- ---------- ------- ----------
         1          0 SYSTEM  READ WRITE
         2          1 ONLINE  READ WRITE
         3          2 RECOVER DISABLED
         4          3 ONLINE  READ WRITE
         5          4 ONLINE  READ WRITE
         6          5 ONLINE  READ WRITE
         7          6 ONLINE  READ ONLY
```

4. Is the database open or closed? Open the database if necessary. What command did you use? Why?

See above. Could have used RECOVER DATABASE as this would have cleared the Media recovery requirement of the offline file as well. We now have to do that manually.

```
SQL> recover datafile '/home/db01/DATA/users01.dbf';
SQL> select file#, ts#, status, enabled from v$datafile;

     FILE#        TS# STATUS  ENABLED
---------- ---------- ------- ----------
         1          0 SYSTEM  READ WRITE
```

```
         2          1 ONLINE  READ WRITE
         3          2 OFFLINE DISABLED
         4          3 ONLINE  READ WRITE
         5          4 ONLINE  READ WRITE
         6          5 ONLINE  READ WRITE
         7          6 ONLINE  READ ONLY
SQL> alter tablespace users online;
Tablespace altered.
SQL> select file#, ts#, status, enabled from v$datafile;

     FILE#        TS# STATUS  ENABLED
---------- ---------- ------- ----------
         1          0 SYSTEM  READ WRITE
         2          1 ONLINE  READ WRITE
         3          2 ONLINE  READ WRITE
         4          3 ONLINE  READ WRITE
         5          4 ONLINE  READ WRITE
         6          5 ONLINE  READ WRITE
         7          6 ONLINE  READ ONLY
```

Exercise 2 Solution

Run the demobld.sql script as SYSTEM in order to recreate the SCOTT user and his objects ($HOME/LABS/demobld.sql)

% sqlplus system/manager

```
SQL*Plus: Release 8.1.5.0.0 - Production on Tue Feb 1 05:15:23 2000
(c) Copyright 1999 Oracle Corporation.  All rights reserved.

Connected to:
Oracle8i Enterprise Edition Release 8.1.5.0.0 - Production
With the Partitioning and Java options
PL/SQL Release 8.1.5.0.0 - Production

SQL> @$HOME/LABS/demobld
Disconnected from Oracle8i Enterprise Edition Release 8.1.5.0.0 - Production
With the Partitioning and Java options
PL/SQL Release 8.1.5.0.0 - Production

% sqlplus scott/tiger

SQL*Plus: Release 8.1.5.0.0 - Production on Tue Feb 1 05:15:23 2000
(c) Copyright 1999 Oracle Corporation.  All rights reserved.

Connected to:
Oracle8i Enterprise Edition Release 8.1.5.0.0 - Production
With the Partitioning and Java options
PL/SQL Release 8.1.5.0.0 - Production

SQL> select table_name, tablespace_name from user_tables;

TABLE_NAME                     TABLESPACE_NAME
------------------------------ ------------------------------
BONUS                          USERS
DEPT                           USERS
EMP                            USERS
SALGRADE                       USERS
```

1. Record the current log sequence number.

```
SQL> connect internal
Connected.
SQL> archive log list
Database log mode              Archive Mode
Automatic archival             Enabled
Archive destination            /home/teach4/ARCHIVE
Oldest online log sequence     7
Next log sequence to archive   8
Current log sequence           8
```

2. Place the USERS tablespace in hot backup mode.

```
SQL> alter tablespace users begin backup;

Tablespace altered.
```

3. Switch logfiles three times -

```
SQL> alter system switch logfile;
System altered.

SQL> /
System altered.

SQL> /
```

```
System altered.
```

4. Take an O/S backup of the USERS tablespace and then end hot backup mode.

```
SQL> host
% cp users01.dbf ../BACKUP
% ls ../BACKUP
users01.dbf
% exit
SQL> alter tablespace users end backup;
Tablespace altered.
```

5. Create a table nologging as SCOTT within the USERS tablespace

```
SQL> connect scott/tiger
Connected.
SQL> create table nolog_table (col1 number) nologging;
Table created.
```

Insert some rows into this table.

```
SQL> insert into nolog_table values (1);
1 row created.

SQL> c/1/2
  1* insert into nolog_table values (2)
SQL> /
1 row created.

SQL> c/2/3
  1* insert into nolog_table values (3)
SQL> /
1 row created.

SQL> c/3/4
  1* insert into nolog_table values (4)
SQL> /
1 row created.
```

Create another table as select from the EMP table.

```
SQL> create table log_table as select * from emp;
Table created.
```

6. Switch some log files (3 times).

```
SQL> alter system switch logfile;
System altered.
SQL> /
System altered.
SQL> /
System altered.
```

7. Select the block number from dba_segments for the header block of the EMP table.

```
SQL> select header_file, header_block from dba_segments
  2  where segment_name = 'EMP';

HEADER_FILE HEADER_BLOCK
----------- ------------
          3          252
```

8. Shutdown abort

```
SQL> shutdown abort
ORACLE instance shut down.
```

9. Run the shell script case2.sh providing the arguments ORACLE_SID and header block from step 7.

eg. **%** case2.sh DB01 252

10. Startup the database and SELECT * FROM scott.emp (or whomever your user is). What happens? What is wrong?

    The shell script case2.sh corrupts the block that is passed to it, which is expected to be the segment header block of the EMP table. Any attempts to select the segment header result in ORA-1578 datablock corrupted.

```
SVRMGR> select * from scott.emp;
EMPNO      ENAME      JOB       MGR        HIREDATE  SAL        COMM
DEPTNO
---------- ---------- --------- ---------- --------- ---------- --------
ORA-01578: ORACLE data block corrupted (file # 3, block # 252)
ORA-01110: data file 3: '/home/db01/DATA/users01.dbf'
```

    From the alert.log we can also see something like the following indicating that the block is corrupt.

```
Corrupt block relative dba: 0x0100012e file=4. blocknum=302.
Bad header found during buffer read
Data in bad block - type:6. format:2. rdba:0x01000125
last change scn:0x0000.00016981 seq:0x1 flg:0x02
consistancy value in tail 0x69810601
check value in block header: 0x0, check value not calculated
spare1:0x0, spare2:0x0, spare2:0x0
```

    Access to the data via the segment header is not possible.

11. What can you do ? Perform any means of recovery deemed necessary.
    Options include, dropping the table, selecting from an index (those columns that are indexed), restoring a backup and rolling through the corruption (provided that the corruption is not due to a bug that causes it to be replayed). We will use the backup restoration possibility, for this exercise.

    Restore the datafile from the users tablespace and rollforward.

```
SQL> startup mount
ORACLE instance started.
Total System Global Area       4430016 bytes
Fixed Size                       47296 bytes
Variable Size                  4177920 bytes
Database Buffers                122880 bytes
Redo Buffers                     81920 bytes
Database mounted.
SQL> recover datafile '/home/db01/DATA/users01.dbf'
ORA-00279: change 152746 generated at 08/13/98 13:24:29 needed for thread 1
ORA-00289: suggestion : /home/db01/ARCHIVE/T0001S000000008.ARC
ORA-00280: change 152746 for thread 1 is in sequence #8
Specify log: {<RET>=suggested | filename | AUTO | CANCEL}
Log applied.
ORA-00279: change 152752 generated at 08/13/98 13:28:09 needed for thread 1
ORA-00289: suggestion : /home/db01/ARCHIVE/T0001S000000009.ARC
ORA-00280: change 152752 for thread 1 is in sequence #9
ORA-00278: log file '/home/db01/ARCHIVE/T0001S000000008.ARC' no
 longer needed for this recovery
Specify log: {<RET>=suggested | filename | AUTO | CANCEL}
Log applied.
….
Log applied.
ORA-00279: change 152755 generated at 08/13/98 13:28:24 needed for thread 1
ORA-00289: suggestion : /home/db01/ARCHIVE/T0001S0000000010.ARC
ORA-00280: change 152755 for thread 1 is in sequence #10
```

```
ORA-00278: log file '/home/db01/ARCHIVE/T0001S000000009.ARC' no
 longer needed for this recovery
Specify log: {<RET>=suggested | filename | AUTO | CANCEL}

Log applied.
Media recovery complete.
```

12. What tables are missing if any (look at nolog_table and log_table)? Why?

Open the database and determine how successful the recovery was.

```
SQL> select * from emp;

     EMPNO ENAME      JOB              MGR HIREDATE   SAL    COMM DEPTNO
---------- ---------- --------- ---------- --------- -----  ----- ------
      7369 SMITH      CLERK           7902 17-DEC-80   80             20
….
14 rows selected


SQL> select * from nolog_table;

      COL1
----------
         1
         2
         3
         4

4 rows selected.
```

```
SQL> select * from log_table;
select * from log_table
              *
ERROR at line 1:
ORA-01578: ORACLE data block corrupted (file # 3, block # 252)
ORA-01110: data file 3: '/home/db01/DATA/users01.dbf'
```

As the above select statements show, recovery was successful in restoring the corrupted segment header for the EMP table. However, other tables within the tablespace were also affected. The table created with NOLOGGING has the rows present after recovery as they were inserted via standard DML, which is logged via redo. The table, LOG_TABLE was created with CTAS (create table as select), an operation that can be performed via nologging, and as that is the default for our tablespace, that method is used. Tables created via nologging operations are left in a MEDIA CORRUPT state (similar to the direct path state caused by failed SQL*Loader sessions in V7) by a media recovery operation. It is necessary to back up the datafiles which belong to tablespaces where nologging operations are performed (if it is required that the objects exist after media recovery operations). The controlfile has an entry that records when an unrecoverable operation is performed -

```
DATA FILE #3:
  (name #7) /home/db01/DATA/users01.dbf
creation size=10240 block size=4096 status=0xe head=7 tail=7 dup=1
 tablespace 3, index=3 krfil=3 prev_file=0
 unrecoverable scn: 0x0000.000254f1 08/14/98 10:23:12
 Checkpoint cnt:136 scn: 0x0000.000254e9 08/14/98 10:22:41
 Stop scn: 0xffff.ffffffff 08/14/98 10:01:29
 Creation Checkpointed at scn: 0x0000.0000226a 08/09/98 12:36:21
 thread:1 rba:(0x3a.372.c0)
 enabled  threads:  01000000 00000000 00000000 00000000 00000000 00000000
00000000 00000000
```

Investigating the redo entries will show information similar to this below. DBA 0x00c000fd is the first block of the table containing rows, after the segment header (the table only contains one block of data).

```
REDO RECORD - Thread:1 RBA: 0x000018.00000015.01ec LEN: 0x003c VLD: 0x01
SCN scn: 0x0000.000254f0 08/14/98 10:23:11
CHANGE #1 TYP:0 CLS:15 AFN:2 DBA:0x00800002 SCN:0x0000.000254ef SEQ:  2
OP:5.4
ktucm redo: slt: 0x0005 sqn: 0x00000065 srt: 0 sta: 9 flg: 0x0

REDO RECORD - Thread:1 RBA: 0x000018.00000016.0044 LEN: 0x0028 VLD: 0x01
SCN scn: 0x0000.000254f1 08/14/98 10:23:11
CHANGE #1 INVLD AFN:3 DBA: 0x00c000fd BLKS:0x0001 SCN:0x0000.000254f1 SEQ:
1 OP:
19.2
Direct Loader invalidate block range redo entry


REDO RECORD - Thread:1 RBA: 0x000018.00000016.006c LEN: 0x00ac VLD: 0x01
SCN scn: 0x0000.000254f1 08/14/98 10:23:11
CHANGE #1 TYP:0 CLS:15 AFN:2 DBA:0x00800002 SCN:0x0000.000254f0 SEQ:  1
OP:5.2
ktudh redo: slt: 0x0011 sqn: 0x00000067 flg: 0x0009 siz: 52 fbi: 1
            uba: 0x008001a3.0034.01    pxid: xid: 0x0000.000.00000000
```

Lesson 8 Practice: Tablespace Point-in-Time Recovery

Lab exercise - A step by step example to TSPITR.

(Simplified verion of WebIV note 62332.1 by NEVANS.UK)

PURPOSE

The purpose of this document is to illustrate the steps involved in Tablespace Point In Time Recovery (TSPITR) via a working example.

This practice implemented here is a relatively common scenario of a user inadvertantly dropping a table that they still require. In order to simplify the issue the user (PITR) creates all objects apart from indexes in a single tablespace.

Our starting point will be the first startup after we have taken a full offline backup of all our datafiles and the controlfiles (by whichever means). It is assumed that the database is running in ARCHIVELOG mode.

Part 1 - Setup

~~~~~~~~~~~~~~~~~~~~~~~~~~~~

1) Create a user PITR with a default tablespace of USERS as the test user. Grant CONNECT and RESOURCE privileges to PITR. As the user PITR create an EMP, DEPT, and BONUS table by selecting from the SCOTT schema. Then shutdown the database and take a full backup.

```
SQL> create user PITR identified by PITR default tablespace USERS temporary tablespace TEMP;
User created.
SQL> grant connect, resource to pitr;
Grant suceeded.
SQL> connect scott/tiger
Connected.
SQL> grant select on emp to PITR;
Grant suceeded.
SQL> grant select on dept to PITR;
Grant suceeded.
SQL> grant select on bonus to PITR;
Grant suceeded.
SQL> connect PITR/pitr
Connected.
SQL> create table EMP as select * from SCOTT.EMP;
Table created.
SQL> create table DEPT as select * from SCOTT.DEPT;
Table created.
SQL> create table BONUS as select * from SCOTT.BONUS;
Table created.
SQL> connect internal
Connected.
SQL> shutdown immediate
Database closed.
Database dismounted.
ORACLE instance shut down.
```

2) Restart the instance and generate a backup control file (not a trace file for control file creation).

Note: The backup controlfile used for the TSPITR should be created at the same time as the backup of the recovery set and must, without exception, be
~~~~~~~~~~~~~~~~~~~~~~~~~~~~

created with this command as you will get the following error when starting the clone database:

```
ORA1696, 00000, "controlfile is not a clone controlfile"
// *Cause:  Attempting to mount, a database as a clone when it is already
//          mounted by another instance not as a clone or attempting to use
//          a current controlfile for a clone.
// *Action: Mount without the clone option or use a backup controlfile and
//          shutdown the other instances before mounting as a clone.
```

```
SQL> connect internal
Connected to an idle instance.
SQL> startup
ORACLE instance started.

Total System Global Area    5881232 bytes
Fixed Size                    64912 bytes
Variable Size               4874240 bytes
Database Buffers             409600 bytes
Redo Buffers                 532480 bytes
Database mounted.
Database opened.
```

3) Connect as PITR and list the tables created and the tablespace(s) in which they reside.

```
SQL> connect pitr/pitr
Connected.
SQL> select table_name, tablespace_name from user_tables;

TABLE_NAME                     TABLESPACE_NAME
-------------------------------  ----------------------------------------
EMP                            USERS
DEPT                           USERS
BONUS                          USERS
```

4) Query the rows in the BONUS and DEPT tables.

```
SQL> select * from BONUS;
SQL> select * from DEPT;
```

5) Insert the following rows into the BONUS table. Don't forget to commit your work.

```
SQL> insert into bonus values ('SMITH','MANAGER',30000,2000);
1 row processed.
SQL> insert into bonus values ('JONES','ADMIN',15000,1000);
1 row processed.
SQL> insert into bonus values ('BROWN','SECURITY',16000,500);
1 row processed.
SQL> commit;
```

6) Optional: Create an index on the ENAME column of the BONUS table in the SYSTEM tablespace.

```
SQL> create index bonus_idx on bonus(ename) tablespace SYSTEM;
Statement processed.
```

7) Get the current time.

```
SQL> select to_char(sysdate,'YYYY-MM-DD:HH24:MI:SS') from dual;
```

```
TO_CHAR(SYSDATE,'Y
------------------
2000-01-31:13:38:37
1 row selected.
```

8) Create the problem by dropping the DEPT table. Not realizing your mistake, create the table SALGRADE with three columns of type NUMBER: GRADE, LOSAL, and HISAL.

```
SQL> drop table dept;      <<== OOPS!!!
Statement processed.

SQL> create table salgrade (grade number,losal number,hisal number);
Statement processed.
```

Note: This is to highlight the fact any committed transactions that are made after the point in time at which we stop recovery, (just before the DEPT table was dropped), will not be recovered. Hence, SALGRADE will not be recovered.

Part 2 - Diagnosing the Situation

~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

The user has realised his error and now approaches the DBA to get the table retrieved. The user informs the DBA that the DEPT table was dropped around 1:30 pm today.

9) Check which objects will be lost after performing TSPITR with SQL similar to the following. See README.pit (in $ORACLE_HOME/rdbms/doc) for further details...:

```
SQL> connect internal
Connected.

SQL> select owner, name, tablespace_name,
  2>       to_char (creation_time,'YYYY-MM-DD:HH24:MI:SS')
  3> from sys.ts_pitr_objects_to_be_dropped
  4> where tablespace_name in (' USERS ')
  5> and creation_time > to_date('00-JAN-31:13:30:00','YY-MON-DD:HH24:MI:SS')
  6> order by tablespace_name, creation_time;
```
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

```
OWNER                          NAME                           TABLESPACE_NAME    TO_CHAR(CREATION_TI
------------------------------ ------------------------------ ---------------- ----------- -------------------
TSPITR                         SALGRADE                       USERS              2000-01-31:13:47:13
1 row selected.
```

10) Deciding to proceed with TSPITR, prevent further data loss on the database by taking the tablespaces to be recovered offline. Then, in preparation for performing TSPITR, make certain the contents of the current log will be available as an archive file if needed.

```
SQL> alter tablespace users offline immediate;
Statement processed.

SQL> alter system archive log current;
Statement processed.
```

Part 3 - Preparing the CLONE database for TSPITR export

We must now prepare the clone database in order to recover the tablespace with the dropped table.

11) Set the Unix environment to point to a CLONE instance. The name should be CLxx, where xx is your assigned number. Copy the configuration files from the production database to files for the clone database.

```
% setenv ORACLE_SID CL01
% cp initDB01.ora initCL01.ora
% cp configDB01.ora configCL01.ora
```

12) Edit the clone database parameter files and add the following parameters:

control_files = (/home/DB01/CLONE/backup.ctl)
lock_name_space = CL01

The control_files parameter must point at a backup copy of the controlfile or you will be unable to open the clone database. The lock_name_space parameter allows the clone database to start up even though it has the same name as the primary (the db_name parameter should NOT be changed).

There are also some optional parameters which will automatically update the clone controlfile with the locations of the datafiles and logfiles after we have restored them, an example of this is:

db_file_name_convert="DATA","CLONE"
log_file_name_convert="DATA","CLONE"

Alternatively you could you use "ALTER DATABASE RENAME FILE '<...>' TO '<...>';" when the clone database is in a mount state.

13) Start the clone database in NOMOUNT mode using the pfile option to specify the modified parameter file.

```
SQL> startup nomount pfile=/home/DB01/CLONE/initCL01.ora
ORACLE instance started.
Total System Global Area      4852460 bytes
Fixed Size                      47852 bytes
Variable Size                 4321280 bytes
Database Buffers               409600 bytes
Redo Buffers                    73728 bytes
```

14) Using the special CLONE option of the mount command, alter the clone database into the mount state. Then check the status of the datafiles for this database.

```
SQL> alter database mount CLONE database;
Statement processed.

SQL> select name, status from v$datafile;
NAME                                                                    STATUS
------------------------------------------------------------------------ -------
/home/DB01/DATA/system01.dbf                                            SYSOFF
/home/DB01/DATA/rbs01.dbf                                               OFFLINE
/home/DB01/DATA/temp01.dbf                                              OFFLINE
/home/DB01/DATA/query01.dbf                                             OFFLINE
/home/DB01/DATA/users01.dbf                                             OFFLINE
```

15) Rename any files which were relocated and not modified by the DB_FILE_NAME_CONVERT parameter. Then bring all the files in the RECOVERY set and AUXILLIARY set online.

```
SQL> alter database datafile
  2> '/home/DB01/DATA/system01.dbf' to '/home/DB01/DATA/system01.dbf';
Statement processed.

Repeat as needed for each file.

SQL> alter database datafile '/home/DB01/CLONE/system01.dbf' online;
Statement processed.

Repeat as needed for each file.
```

16) Recover the database until the time recorded before the DEPT table was dropped (Step 7). Open the database.

```
SQL> recover database using backup controlfile until time '1998-08-18:13:38: 37';
ORA-00279: change 12599 generated at 08/18/98 13:25:48 needed for thread 1
ORA-00289: suggestion : /home/DB01/ARCHIVE/arch129.arc
ORA-00280: change 12599 for thread 1 is in sequence #129
Specify log: {<RET>=suggested | filename | AUTO | CANCEL}

Log applied.
Media recovery complete.

SQL> alter database open resetlogs;
Statement processed.
```

17) Connect as the user PITR and query the objects contained in the recovered file. Verify the DEPT exists.

```
SQL> connect pitr/pitr
SQL> select * from dept;

DEPTNO    DNAME         LOC
---------- --------------  -------------
     10 ACCOUNTING    NEW YORK
     20 RESEARCH          DALLAS
     30 SALES             CHICAGO
     40 OPERATIONS    BOSTON
4 rows selected.
```

18) We must now ascertain what relationships, if any, exist between objects across the bounds of the recovery set. Further details of this can be found in README.pit. In short, the dictionary view provided for gathering this information is TS_PITR_CHECK. In our example we would use the following SQL to query this view:

If you did not create the index on BONUS, then skip from here until you see … *Skip until here*

```
SQL> select *
  2> from sys.ts_pitr_check
  3> where
  4> (ts1_name in ('USERS')
  5> and ts2_name not in ('USERS'))
  6> or (ts1_name not in ('USERS')
  7> and ts2_name in ('USERS'));

OBJ1_OWNER          OBJ1_NAME  OBJ1_TYPE      TS1_NAME                OBJ2_NAME
 OBJ2_TYPE   OBJ2_OWNER  TS2_NAME                CONSTRAINT_NAME            REASON

------------------------------ ------------------------------ ------------------
------------ --------------- ------------------------------ --------------------
TSPITR                  BONUS          TABLE         USERS                BONUS_IDX
INDEX         TSPITR       SYSTEM

Tables and associated indexes not fully contained in the recovery set
```

Note: README.pit gives details of suggested output formatting within SQL*Plus in order to make the output a little more user friendly. The formatted output would look something like:

```
owner1   name1  obj1typ    ts1_name        name2   obj2typ   owner2  ts2_name
---------   --------  ---------  ---------------  ---------------  ----------  ----------  ------------
TSPITR  BONUS  TABLE        USERS   BONUS_IDX   INDEX   TSPITR   SYSTEM  ...

    Tables and associated indexes not fully contained in the recovery set
```

From this output we can see that the BONUS table has an index BONUS_IDX that is contained in a separate tablespace, we must decide whether to include the other tablespace in the Recovery set or drop this index and recreate it after recovery. This must be resolved, and any actions taken at this point should be noted and performed on the PROD database as well, or TSPITR import will fail with:

```
ORA-29308, 00000, "view TS_PITR_CHECK failure"
// *Cause: Some objects which crossed the boundary of the recovery set were
//         not allowed in the tabelspace point-in-time recovery.
// *Action: Query TS_PITR_CHECK and resolve the boundary crossing objects.
```

19) Since it is an index, drop BONUS_IDX rather than alter the recovery set.

```
SQL> drop index pitr.bonus_idx;
Statement processed.
```

…. Skip until here

20) Verify that there are no objects which are not fully contained.

```
SQL> select *
```

```
2> from sys.ts_pitr_check
3> where
4> (ts1_name in ('USERS')
5> and ts2_name not in ('USERS'))
6> or (ts1_name not in ('USERS')
7> and ts2_name in ('USERS'));
```

No rows selected.

Part 4 - Performing TSPITR export

21) Now we are ready to export the required tablespace from the clone database. Perform the TSPITR export for the USERS tablespace.

```
% exp sys/change_on_install point_in_time_recover=y tablespaces=users

Export: Release 8.1.0.5.0 - Production on Tue Feb 1 11:09:09 2000

(c) Copyright 1999 Oracle Corporation.  All rights reserved.

Connected to: Oracle8i Enterprise Edition Release 8.1.5.0.0 - Production
With the Partitioning and Java options
PL/SQL Release 8.1.5.0.0 - Production
Export done in WE8ISO8859P1 character set and WE8ISO8859P1 NCHAR character set
Note: table data (rows) will not be exported

About to export Tablespace Point-in-time Recovery objects...
For tablespace USERS ...
. exporting cluster definitions
. exporting table definitions
. . exporting table                    EMP
. . exporting table                    BONUS
. . exporting table                    DEPT
. exporting referential integrity constraints
. exporting triggers
. end point-in-time recovery
```

Note: The TSPITR export has to be done as user SYS, if not it will fail with the following errors:

```
EXP-00008: ORACLE error 29300 encountered
ORA-29300: ORACLE error, tablespace point-in-time recovery
ORA-29303: user does not login as SYS
EXP-00000: Export terminated unsuccessfully
```

Part 5 - Performing TSPITR import

~~~~~~

The next step is to import the tablespace metatdata back into our production instance.

22) Reset the Unix environment to point to the Production database.
% setenv ORACLE_SID DB01

23) Prepare the Production database for TSPITR. Remember to perform any actions here that were performed on the CLONE database when preparing for TSPITR export. Then offline the tablespace to be recovered.

Optional: If BONUS_IDX was created, then perform the following step:
SQL> drop index pitr.bonus_idx;
Statement processed.

SQL> connect internal
Connected.
We now need to prepare the tablespace in question for recovery - failing to do so will cause the import to fail.

SQL> alter tablespace users offline for recover;
Statement processed.

24) Replace the recovery set datafiles on the production database with the datafiles we have recovered on the clone database.

!!! Be safe: Save the production files first!!!

% mv /home/DB01/DATA/users01.dbf /home/DB01/DATA/users01.old

Copy the tablespace datafiles from the clone to the primary.

% cp /home/DB01/CLONE/users01.dbf /home/DB01/DATA/users01.dbf

25) Now perform the TSPITR import as SYS.

% imp sys/change_on_install point_in_time_recover=Y

```
Import: Release 8.1.5.0.0 - Production on Tue Feb 01 15:8:12 2000
(c) Copyright 1999 Oracle Corporation.  All rights reserved.

Connected to: Oracle8 Enterprise Edition Release 8.1.5.0.0 - Production
With the Partitioning and Java options
PL/SQL Release 8.1.5.0.0 - Production

Export file created by EXPORT:V08.01.05 via conventional path
About to import Tablespace Point-in-time Recovery objects...
. importing PITR's objects into PITR
. . importing table                "EMP"
. . importing table                "BONUS"
. . importing table                "DEPT"
. importing SYSTEM's objects into SYSTEM
. importing SCOTT's objects into SCOTT
Import terminated successfully with warnings.
```
~~~~~~

Part 6 - Cleanup

~~~~~~~~~~~~~~~

26) Make the USERS tablespace available again.

```
SQL> connect internal
Connected.

SQL> select tablespace_name, status from dba_tablespaces;
TABLESPACE_NAME                STATUS
------------------------------ ---------
SYSTEM                         ONLINE
RBS                            ONLINE
TEMP                           ONLINE
QUERY                          ONLINE
USERS                          OFFLINE

SQL> alter tablespace users online;
Statement processed.
```

Optional: Now we have to recreate the dropped index.
Skip from here until you see …*Skip until here*

27) Recreate the index we had drop.

```
SQL> create index pitr.bonus_idx on pitr.bonus(ename) tablespace system;
```

… *Skip until here*

28) The TSPITR recovery is now complete. To check this we can connect as the user PITR and see what objects have been recovered. Notice that the SALGRADE table no longer exists.

```
SQL> connect pitr/pitr
Connected.

SQL> select object_name from user_objects;

OBJECT_NAME
-----------
EMP
BONUS
BONUS_IDX     (will not be present if not recreated)
DEPT
4 rows selected

SQL> select * from dept;

DEPTNO     DNAME         LOC
---------- -------------- -------------
      10 ACCOUNTING     NEW YORK
      20 RESEARCH       DALLAS
      30 SALES          CHICAGO
      40 OPERATIONS     BOSTON
4 rows selected.
```
~~~~~~~~~~~~~~~

```
SQL> select *from bonus;
ENAME     JOB       SAL       COMM
---------- --------- ---------- ----------
SMITH     MANAGER      30000      2000
JONES     ADMIN        15000      1000
BROWN     SECURITY     16000       500
3 rows selected.
```

Part 7 - Final Task

~~~~~~~~~~~~~~~~~~

29) After TSPITR has been done on a tablespace, it is VITALLY important that it is backed up. Failure to do so could result in loss of that tablespace in the event of media failure, because the archived logs from the last backup taken of that database do not logically link to the recovered tablespaces. If an attempt is made to recover any recovery set tablespaces from a backup taken before TSPITR, it will fail with ORA-01246: 'recovering files through TSPITR of tablespace %s'.

RELATED DOCUMENTS

<Note:46090.1> 3. Terminology
<Note:46091.1> 4. Restrictions of TSPITR
<Note:46092.1> 5. Database requirements of Tablespace Point-In-Time
<Note:46094.1> 7. Steps involved in Tablespace Point-In-Time Recovery
<Note:46095.1> 8.1 Advisory Note
<Note:46098.1> 9. Addendum 'Tablespace Point-In-Time Recovery Alerts'
README.pit file: $ORACLE_HOME/rdbms/doc on UNIX installations
%ORACLE_HOME%\Rdbms or %ORACLE_HOME%\rdbms80\doc
on Windows NT installations
~~~~~~~~~~~~~~~~~~