



Oracle 19c Database Administration II (DBAII)

Copyright © 2024 Trivera Technologies LLC. | www.triveratech.com

Experience is Everything

Jumping right In...

- Welcome!
 - **Oracle 19c Database Administration II (DBAII)**
 - All topics and labs are geared for experienced DBAs with prior hands-on exposure to Oracle 19c.
 - Topics or activities may adjust during delivery based on your interests, roles and goals.
- Hours:
 - 10:00 to 6:00 PM Eastern; One Hour for Lunch; A few breaks as needed
- A Bit About Me: Steve Hamilton Steve.Hamilton@triveratech.com
 - Senior Oracle Instructor & Coach, Trivera Technologies www.triveratech.com
- A Bit About You:
 - What's your role / day to day?
 - Are you working with these skills already?
 - What kinds of related things are you working on?
 - What are you most excited to learn about in this class?

Teaming for Success

- **Sharing Feedback – We’re Here to Provide Value!**
 - Feedback is welcome & always encouraged
 - Real time is best
 - Other ways to connect - chat
 - Course Check In – Mid Course Review at the mid-point to gauge how things are going
 - End of Course feedbacks – please complete so we can issue your Certificate of Completion.
- **Course Recordings**
 - Provided by separate link a few days after class
- **Course Certificates**
 - Will be sent out a few days after class after End of course survey is completed.

Course Agenda

- There will be ample hands-on labs and activities (about 50%) throughout.
- We'll focus activities on things that will be useful to you and provide value.

Day One

1. Oracle Database Quick Refresher
 - Lab: Exploring Oracle Database
2. Creating and Managing Tablespaces
 - Lab: Tablespace Management
3. Improving Space Usage
 - Lab: Space Management Practices
4. Managing Undo Data
 - Lab: Undo Data Management
5. Configuring User Resource Limits
 - Lab: User Resource Management

Course Agenda

Day Two

6. Implementing Oracle Database Auditing
 - Lab: Database Auditing Practices
7. Introduction to Loading and Transporting Data
 - Lab: Data Loading and Transport Overview
8. Loading Data
 - Lab: Practical Data Loading
9. Transporting Data
 - Lab: Data Transport Methods
10. Using External Tables to Load and Transport Data
 - Lab: External Table Implementation

Day Three

11. Automated Maintenance Tasks
 - Lab: Maintenance Automation
12. Automated Maintenance Tasks: Managing Tasks and Windows
 - Lab: Managing Maintenance Tasks
13. Database Monitoring and Tuning: Performance Overview
 - Lab: Performance Monitoring Overview
14. Monitoring Database Performance
 - Lab: Practical Performance Monitoring
15. Database Processes
 - Lab: Database Process Management

Course Agenda

Day Four

- 16. Managing Memory
 - Lab: Memory Management Practices
- 17. Analyzing SQL and Optimizing Access Paths
 - Lab: SQL Optimization Techniques
- 18. Backup and Recovery Overview
 - Lab: Backup and Recovery Concepts
- 19. Configuring Backup and Recovery
 - Lab: Backup Configuration
- 20. RMAN (Recovery Manager)
 - Lab: RMAN Configuration

Day Five

- 21. Creating Backups
 - Lab: Creating and Managing Backups
- 22. Recovery
 - Lab: Recovery Operations
- 23. Using Flashback Technologies
 - Lab: Flashback Technologies
- 24. Diagnosing Failures
 - Lab: Failure Diagnosis



Any Questions?

Let's Dive In!

Experience is Everything

Copyright © 2024 Trivera Technologies LLC. | www.triveratech.com



Quick Review: Oracle 19c Database

Copyright © 2024 Trivera Technologies LLC. | www.triveratech.com

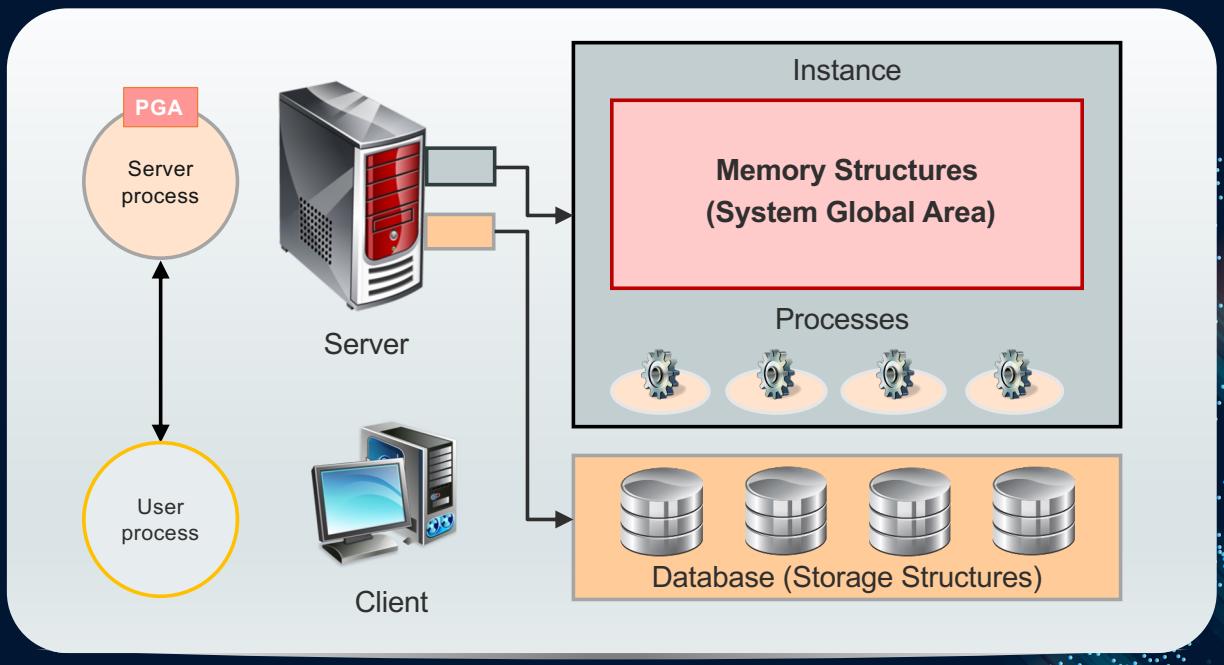
Experience is Everything

Objectives

After completing this lesson, you should be able to:

- List the major architectural components of Oracle Database
- Describe multitenant architecture
- Describe database sharding

Oracle Database Server Architecture: Overview



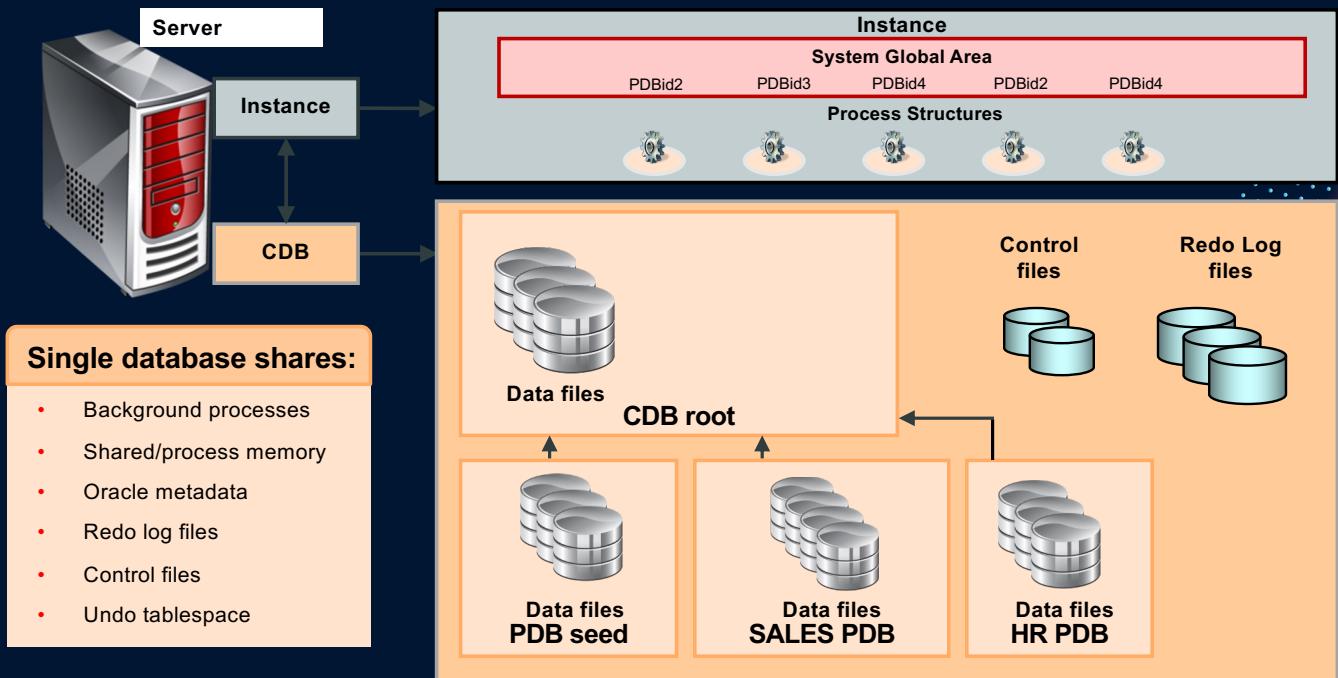
Copyright © 2024 Trivera Technologies LLC. | www.triveratech.com

10

Oracle Multitenant Container Database: Introduction

- *Container*: A logical collection of data or metadata within the multitenant architecture
- *Pluggable database (PDB)*: A portable collection of schemas, schema objects, and nonschema objects
- Containers in a *multitenant container database (CDB)*:
 - CDB root container (also known as the *root*)
 - System container (includes the root and all PDBs)
 - Application containers
 - Root PDB
 - User-created PDBs
- All pluggable databases share:
 - Background processes
 - Shared memory management and some memory structures
 - Some of the Oracle metadata

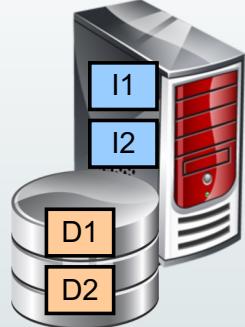
Oracle Multitenant Container Database: Architecture



Copyright © 2024 Trivera Technologies LLC. | www.triveratech.com

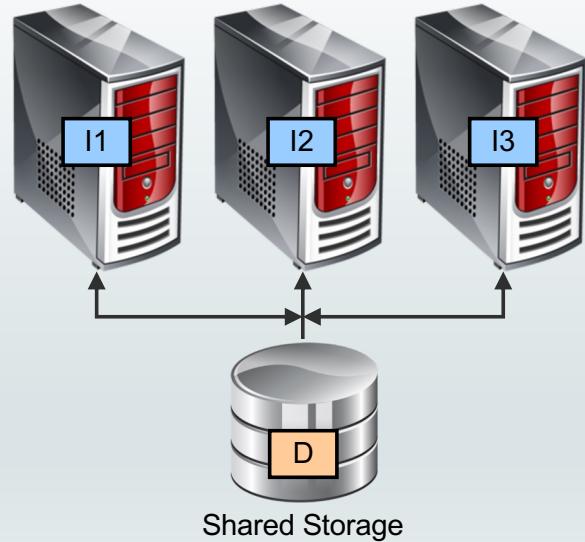
Oracle Database Instance Configurations

Nonclustered System



Local Storage

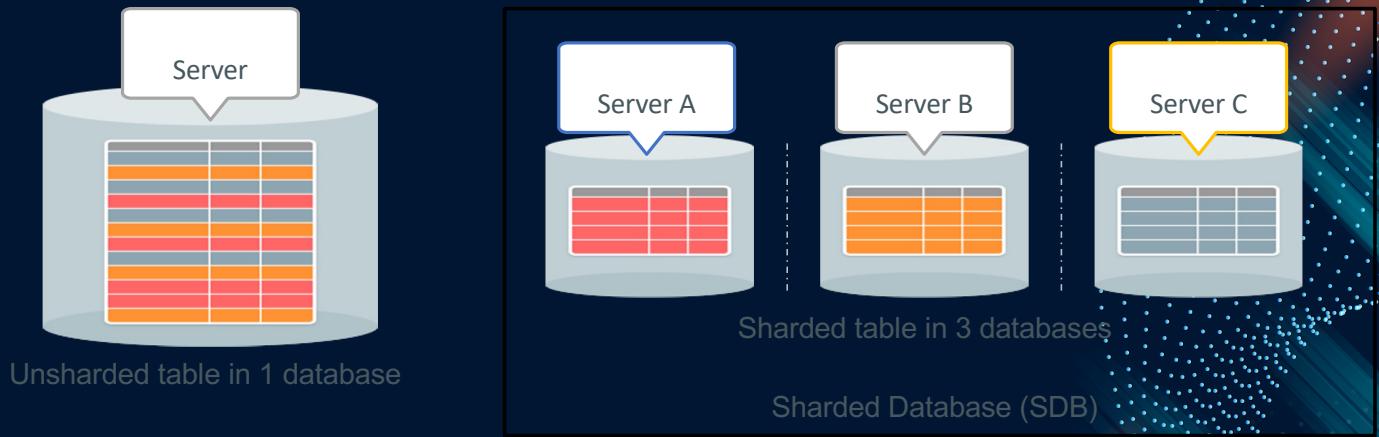
Clustered System



Shared Storage

Database Sharding: Introduction

- A shared-nothing architecture for scalability and availability
- Horizontally partitioned data across independent databases
- Loosely coupled data tier without clusterware



Oracle Database Server: Interactive Architecture Diagram

Access the Interactive Architecture Diagram on the Oracle Help Center Oracle Database “What’s New” page.

<https://tinyurl.com/yepn9ma>

Summary

In this lesson, you should have learned how to:

- List the major architectural components of Oracle Database
- Describe multitenant architecture
- Describe database sharding



Creating and Managing Tablespaces

Copyright © 2024 Trivera Technologies LLC. | www.triveratech.com

Experience is Everything

Objectives

After completing this lesson, you should be able to:

- Create, alter, and drop tablespaces
- View tablespace information
- Implement Oracle Managed Files (OMF)
- Move and rename online data files

Creating Tablespaces

- A tablespace is an allocation of space in the database that can contain schema objects.
- Create a tablespace with the `CREATE TABLESPACE` statement or use a graphical tool.
- You can create three types of tablespaces:
 - Permanent tablespace: Contains persistent schema objects. Objects in permanent tablespaces are stored in data files.
 - Undo tablespace: Is a type of permanent tablespace used by Oracle Database to manage undo data in automatic undo management mode
 - Temporary tablespace: Contains schema objects only for the duration of a session. Objects in temporary tablespaces are stored in temp files.

Creating a Tablespace: Clauses

Include one or more of the following clauses to define various aspects of the tablespace:

Clause	Description
DATAFILE or TEMPFILE	Used to specify the name, location, and initial size of the data file or temp file
ONLINE or OFFLINE	Used to make the tablespace available (or not available) immediately after creation
BLOCKSIZE	Used to specify a nonstandard block size
EXTENT MANAGEMENT	Used to specify how the extents of the tablespace will be managed and where the metadata for allocated and unallocated extents is to be stored
LOGGING	Used to specify the default logging attributes of objects in the tablespace
SEGMENT MANAGEMENT	Used to specify how free space in the segments in the tablespace should be tracked (bitmaps or free lists)

Creating Permanent Tablespaces in a CDB

- Tablespace creation during CDB creation:
 - With DBCA: USERS tablespace created in the CDB root
 - With CREATE DATABASE statement with `USER_DATA TABLESPACE` clause: Your defined tablespace created in the CDB root
- Create a permanent tablespace in the CDB root:

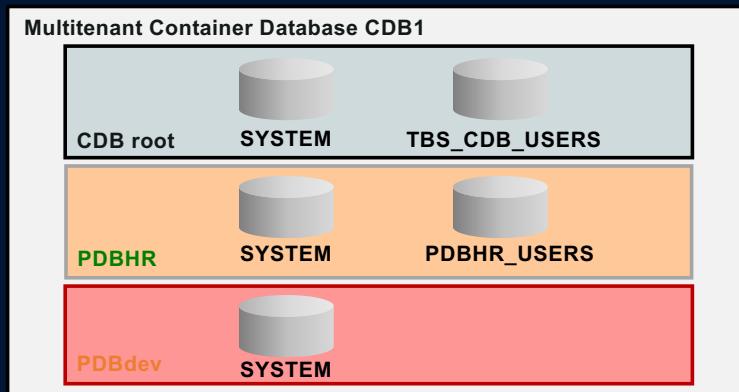
```
SQL> CONNECT system@cdb1
SQL> CREATE TABLESPACE tbs_CDB_users
      DATAFILE '/u1/app/oracle/oradata/cdb/cdb_users01.dbf' SIZE 100M;
```

- Create a permanent tablespace in a PDB:

```
SQL> CONNECT system@PDB1
SQL> CREATE TABLESPACE tbs_PDB1_users
      DATAFILE '/u1/app/oracle/oradata/cdb/pdb1/users01.dbf' SIZE 100M;
```

Defining Default Permanent Tablespaces

- In the CDB
- In the PDB

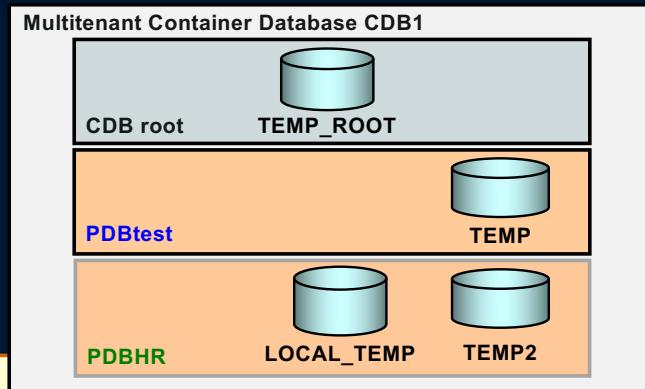


```
SQL> CONNECT system@cdb1
SQL> ALTER DATABASE DEFAULT TABLESPACE tbs_CDB_users;
```

```
SQL> CONNECT pdb1_admin@pdbhr
SQL> ALTER PLUGGABLE DATABASE DEFAULT TABLESPACE pdbhr_users;
```

Temporary Tablespaces

- Only one default temporary tablespace or tablespace group is allowed per CDB or PDB.
- Each PDB can have temporary tablespaces or tablespace groups.
- Define the default temporary tablespace in a PDB:



```
SQL> CONNECT pdb1_admin@pdbhr
SQL> ALTER DATABASE DEFAULT TEMPORARY TABLESPACE local_temp;
```

Altering and Dropping Tablespaces

- When you create a tablespace, it is initially a read/write tablespace.
- Use the `ALTER TABLESPACE` statement to take a tablespace offline or online, add data files or temp files to it, or make it a read-only tablespace.
- A tablespace can be in one of three different statuses or states:
 - Read Write
 - Read Only
 - Offline with one of the following options:
 - NORMAL
 - TEMPORARY
 - IMMEDIATE
- Add space to an existing tablespace by either adding data files to the tablespace or changing the size of an existing data file.
- Use the `DROP TABLESPACE` statement to drop a tablespace and its contents from the database if you no longer need its content.

Viewing Tablespace Information

Tablespace and data file information can be obtained by querying the following views:

- Tablespace information:
 - CDB_TABLESPACES and DBA_TABLESPACES
 - V\$TABLESPACE
- Data file information:
 - CDB_DATA_FILES and DBA_DATA_FILES
 - V\$DATAFILE
- Temp file information:
 - CDB_TEMP_FILES and DBA_TEMP_FILES
 - V\$TEMPFILE
- Tables in a tablespace:
 - ALL_TABLES

Implementing Oracle Managed Files (OMF)

- Specify file operations in terms of database objects rather than file names.

Parameter	Description
DB_CREATE_FILE_DEST	Defines the location of the default file system directory for data files and temporary files
DB_CREATE_ONLINE_LOG_DEST_n	Defines the location for redo log files and control file creation
DB_RECOVERY_FILE_DEST	Gives the default location for the fast recovery area

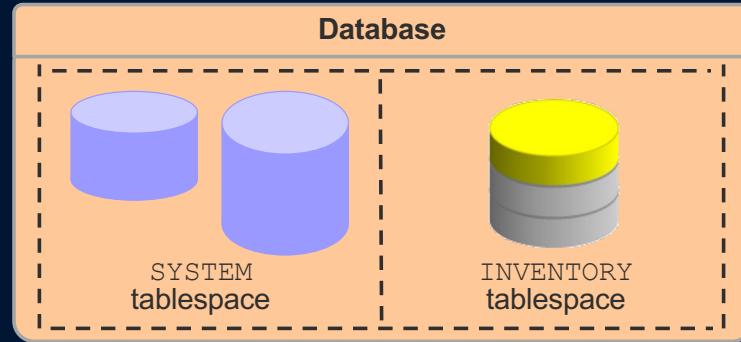
- Example:

```
SQL> ALTER SYSTEM SET DB_CREATE_FILE_DEST='/u01/app/oracle/oradata';
SQL> CREATE TABLESPACE tbs_1;
```

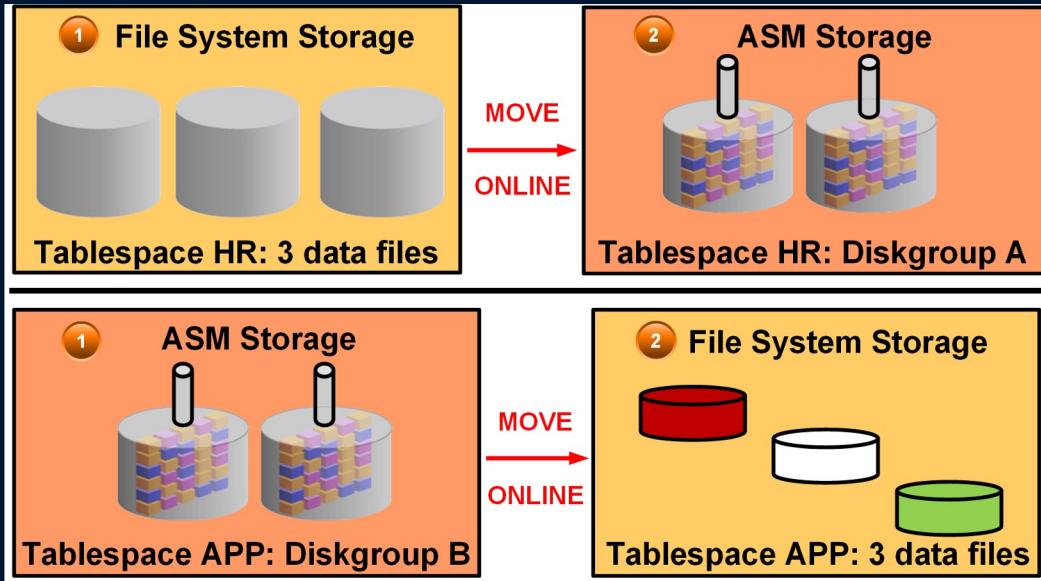
Enlarging the Database

You can enlarge the database in the following ways:

- Create a new tablespace.
- Add a data file to an existing smallfile tablespace.
- Increase the size of a data file.
- Provide for the dynamic growth of a data file.



Moving or Renaming Online Data Files



Examples: Moving and Renaming Online Data Files

- Relocating an online data file:

```
SQL> ALTER DATABASE MOVE DATAFILE '/disk1/myexample01.dbf'  
2 TO '/disk2/myexample01.dbf';
```

- Copying a data file from a file system to Automatic Storage Management (ASM):

```
SQL> ALTER DATABASE MOVE DATAFILE '/disk1/myexample01.dbf'  
2 TO '+DiskGroup2' KEEP;
```

- Renaming an online data file:

```
SQL> ALTER DATABASE MOVE DATAFILE '/disk1/myexample01.dbf'  
2 TO '/disk1/myexample02.dbf';
```

Summary

In this lesson, you should have learned how to:

- Create, alter, and drop tablespaces
- View tablespace information
- Implement Oracle Managed Files (OMF)
- Move and rename online data files

Practice Overview

- Viewing Tablespace Information
- Creating a Tablespace
- Managing Temporary and Permanent Tablespaces



Improving Space Usage

Copyright © 2024 Trivera Technologies LLC. | www.triveratech.com

Experience is Everything

Objectives

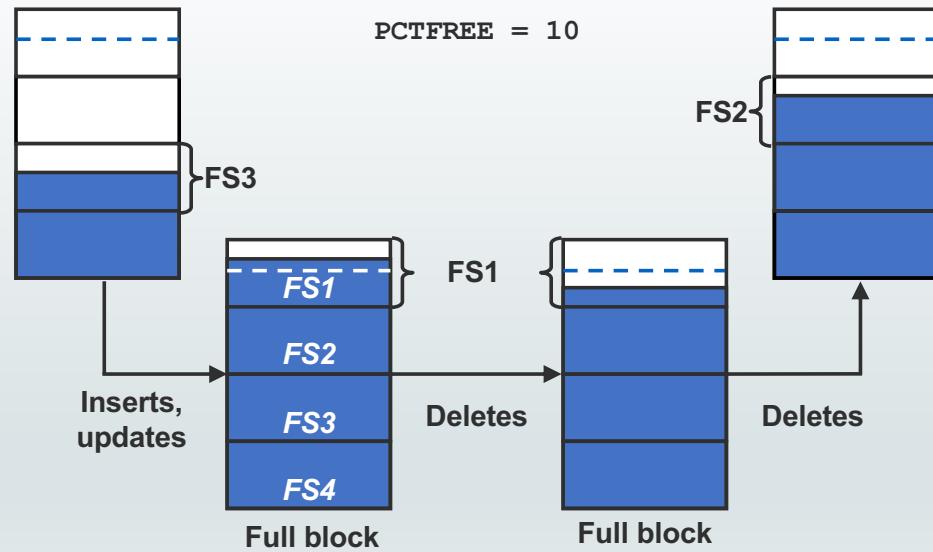
After completing this lesson, you should be able to:

- Describe and use Oracle Database features that save space
- Create private temporary tables
- Save space by using compression
- Reclaim wasted space from tables and indexes by using the segment shrink functionality
- Manage resumable space allocation

Space Management Features

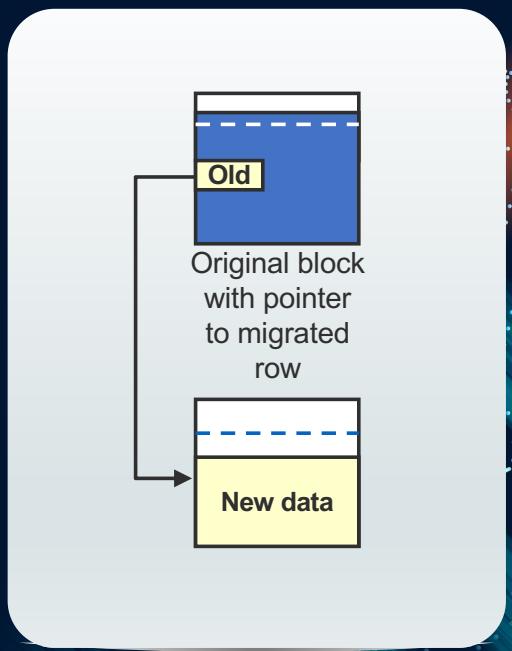
- Space is automatically managed by the Oracle Database server. It generates alerts about potential problems and recommends possible solutions.
- Space management features include:
 - Oracle Managed Files (OMF)
 - Free-space management with bitmaps (“locally managed”) and automatic data file extension
 - Proactive space management (default thresholds and server-generated alerts)
 - Space reclamation (shrinking segments, online table redefinition)
 - Capacity planning (growth reports)

Block Space Management



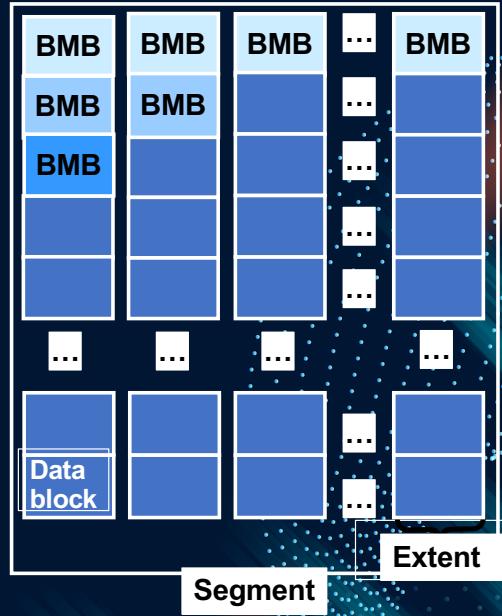
Row Chaining and Migration

- On update: Row length increases, exceeding the available free space in the block.
- Data needs to be stored in a new block.
- Original physical identifier of row (ROWID) is preserved.
- The Oracle Database server needs to read two blocks to retrieve data.
- Segment Advisor finds segments containing the migrated rows.
- There is automatic coalescing of fragmented free space inside the block.



Free Space Management Within Segments

- Tracked by bitmaps in segments
- Benefits:
 - More flexible space utilization
 - Runtime adjustment
 - Multiple process search of bitmap blocks (BMBs)



Allocating Extents

- Searching the data file's bitmap for the required number of adjacent free blocks
- Sizing extents with storage clauses:
 - UNIFORM
 - AUTOALLOCATE
- Viewing the extent map
- Obtaining deallocation advice

Using Unusable Indexes

- Consider using unusable indexes to improve the performance of bulk loads.
- Unusable indexes are ignored by the optimizer.
- When an unusable index is created, no segment is created:

```
CREATE INDEX test_i1 ON seg_test(c) UNUSABLE
```

- When an existing index is altered to unusable, the segment is dropped:

```
ALTER INDEX test_i UNUSABLE
```

- An unusable index can be rebuilt to make it valid again:

```
ALTER INDEX test_i REBUILD
```

Using Temporary Tables

- Temporary tables contain data for the duration of a transaction or session.
- Types of temporary tables:
 - Global: Table definition is visible to all sessions; content is specific to a session.
 - Private: Table definition is visible only to the creating session.
- Segment for a temporary table is allocated with the first `INSERT` or `CREATE TABLE AS SELECT` statement.
- Table definition persists after a `ROLLBACK`.
- Transaction-specific temporary table can only be used by one transaction at a time.

Creating Global Temporary Tables

- Create a global temporary table by using the CREATE GLOBAL TEMPORARY TABLE statement.
- Specify whether the global temporary table applies to a transaction or session by using the ON COMMIT clause:
 - Transaction-specific (default): ON COMMIT DELETE ROWS
 - Session-specific: ON COMMIT PRESERVE ROWS
- Example:

```
SQL> CREATE GLOBAL TEMPORARY TABLE trans_buff_area(date1 DATE,...)
      > ON COMMIT DELETE ROWS;
```

Creating Private Temporary Tables

- Create a private temporary table by using the CREATE PRIVATE TEMPORARY TABLE statement.
- The table name must start with ORA\$PTT_ :

```
PRIVATE_TEMP_TABLE_PREFIX = ORA$PTT_
```

```
SQL> CREATE PRIVATE TEMPORARY TABLE ORA$PTT_mine (c1 DATE, ... c3 NUMBER(10,2));
```

- The CREATE PRIVATE TEMPORARY TABLE statement does not commit a transaction.
- Two concurrent sessions may have a private temporary table with the same name but different shape.
- Private temporary table definition and contents are automatically dropped at the end of a session or transaction.

```
SQL> CREATE PRIVATE TEMPORARY TABLE ORA$PTT_mine (c1 DATE ...)
      ON COMMIT PRESERVE DEFINITION;
```

```
SQL> DROP TABLE ORA$PTT_mine;
```

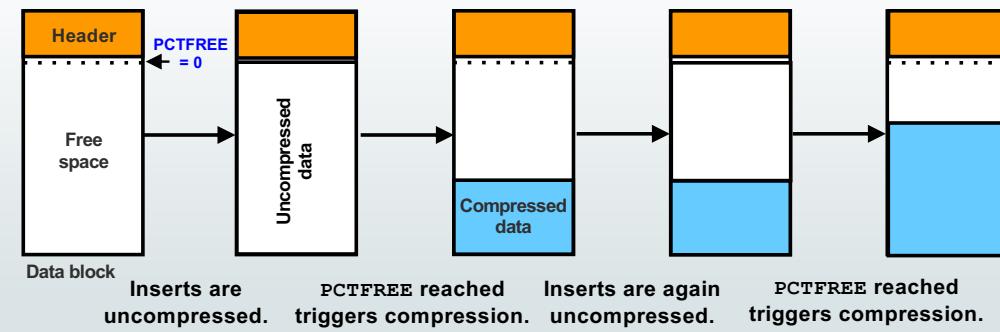
Table Compression: Overview

Reducing storage costs by compressing all data:

- Basic compression for direct-path insert operations: 10x
- Advanced row compression for all DML operations: 2–4x

Compression Method	Compression Ratio	CPU Overhead	CREATE and ALTER TABLE Syntax	Typical Applications
Basic table compression	High	Minimal	COMPRESS or ROW STORE COMPRESS BASIC	DSS
Advanced row compression	High	Minimal	ROW STORE COMPRESS ADVANCED	OLTP, DSS

Table Compression: Concepts



Compression for Direct-Path Insert Operations

- Is enabled with `CREATE TABLE ... COMPRESS BASIC`
- Is recommended for bulk loading data warehouses
- Maximizes contiguous free space in blocks

Advanced Row Compression for DML Operations

- Is enabled with CREATE TABLE ... ROW STORE COMPRESS ADVANCED
- Is recommended for active OLTP environments

Y	Y	Y	Y
G	Y	G	
G	Y	Y	G

Uncompressed block

G	Y			
Y		Y		Y
G	Y	G		
G	Y	Y	Y	G

OLTP compression with the symbol table at the beginning of the block

Specifying Table Compression

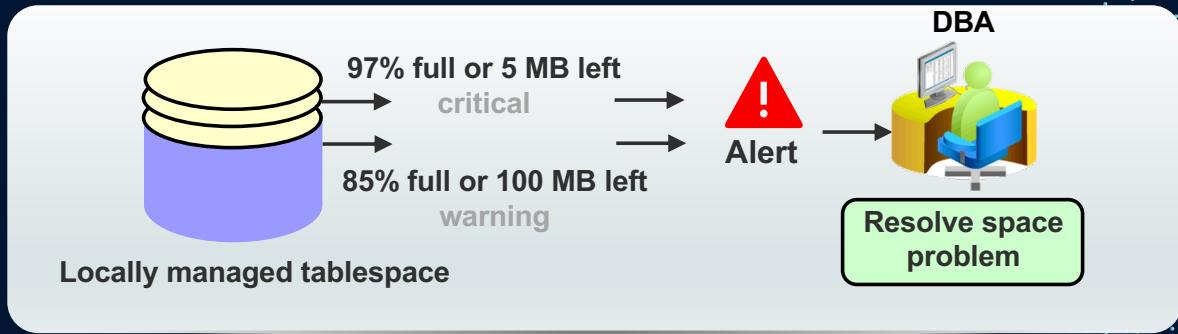
- You can specify table compression for:
 - An entire heap-organized table
 - A partitioned table (each partition can have a different type or level of compression)
 - The storage of a nested table
- You cannot:
 - Specify basic and advanced row compression on tables with more than 255 columns
 - Drop a column if a table is compressed for direct loads, but you can drop it if the table is advance row compressed

Using the Compression Advisor

- Analyzes objects to give an estimate of space savings for different compression methods
- Helps in deciding the correct compression level for an application
- Recommends various strategies for compression
 - Picks the right compression algorithm for a particular data set
 - Sorts on a particular column for increasing the compression ratio
 - Presents tradeoffs between different compression algorithms

Resolving Space Usage Issues

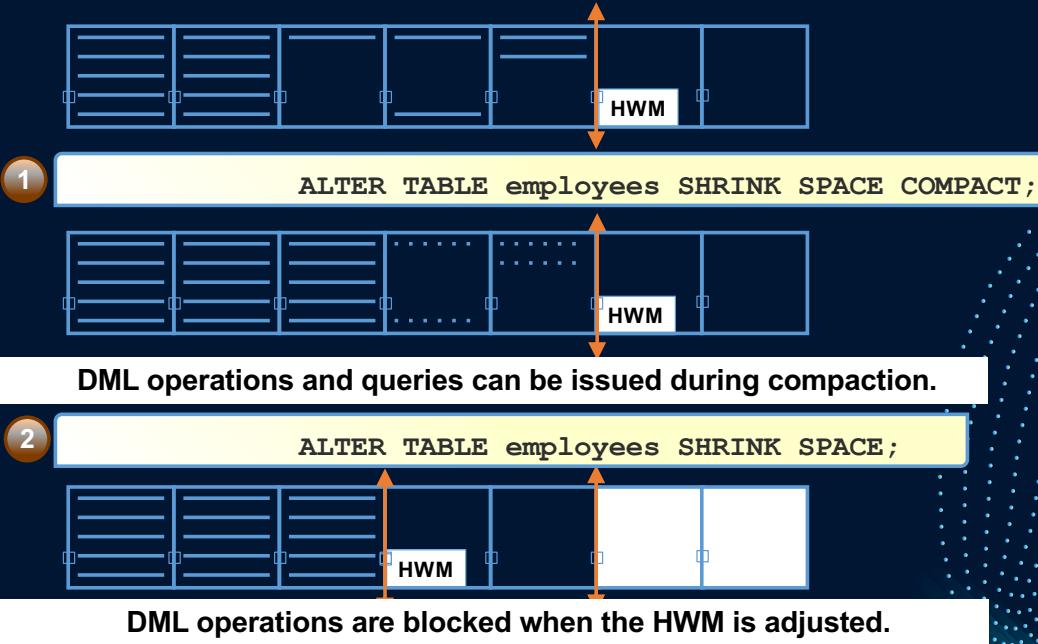
- Resolve space usage issues by:
 - Adding or resizing data files
 - Setting AUTOEXTEND to ON
 - Shrinking objects
 - Reducing UNDO_RETENTION
- Check for long-running queries in temporary tablespaces.



Reclaiming Space by Shrinking Segments

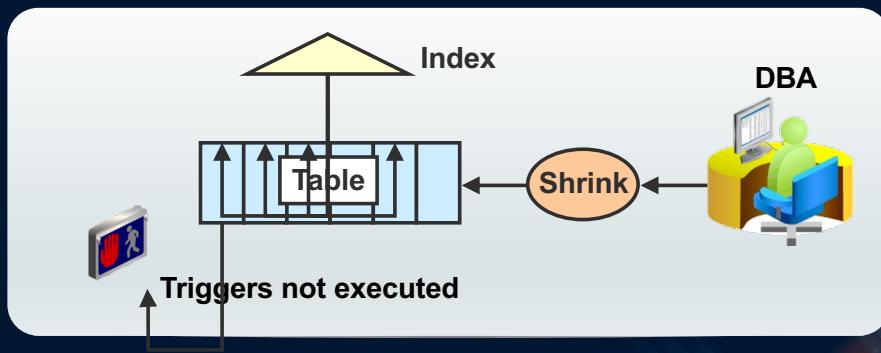
- Shrink is an online and in-place operation.
- It is applicable only to segments residing in ASSM tablespaces.
- Candidate segment types:
 - Heap-organized tables and index-organized tables
 - Indexes
 - Partitions and subpartitions
 - Materialized views and materialized view logs

Shrinking Segments



Results of a Shrink Operation

- Improved performance and space utilization
- Indexes maintained
- Triggers not executed
- Number of migrated rows may be reduced
- Rebuilding secondary indexes on IOTs recommended



Copyright © 2024 Trivera Technologies LLC. | www.triveratech.com

Managing Resumable Space Allocation

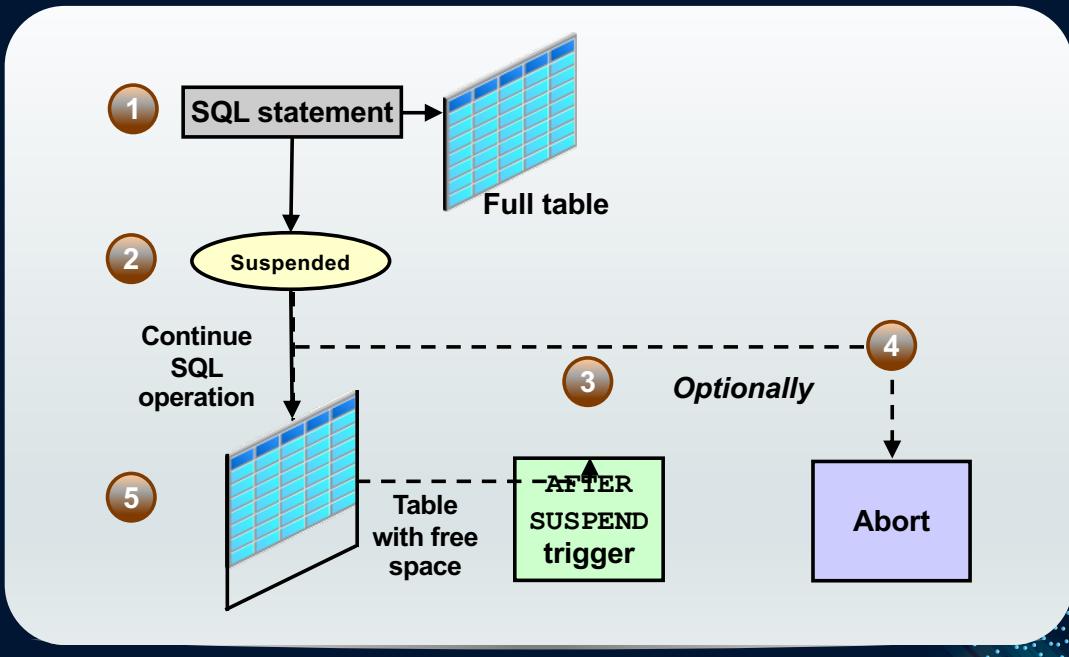
A resumable statement:

- Enables you to suspend large operations instead of receiving an error
- Gives you a chance to fix the problem while the operation is suspended, rather than starting over
- Is suspended for the following conditions:
 - Out of space
 - Maximum extents reached
 - Space quota exceeded
- Can be suspended and resumed multiple times

Using Resumable Space Allocation

- Queries, DML operations, and certain DDL operations can be resumed if they encounter an out-of-space error.
- A resumable statement can be issued through SQL, PL/SQL, SQL*Loader, and Data Pump utilities, or Oracle Call Interface (OCI).
- A statement executes in resumable mode only if its session has been enabled by one of the following actions:
 - The `RESUMABLE_TIMEOUT` initialization parameter is set to a nonzero value.
 - An `ALTER SESSION ENABLE RESUMABLE` statement is issued.

Resuming Suspended Statements



What Operations Are Resumable?

The following operations are resumable:

- Queries: SELECT statements that run out of temporary space (for sort areas)
- DML: INSERT, UPDATE, and DELETE statements
- The following DDL statements:
 - CREATE TABLE ... AS SELECT
 - CREATE INDEX
 - ALTER INDEX ... REBUILD
 - ALTER TABLE ... MOVE PARTITION
 - ALTER TABLE ... SPLIT PARTITION
 - ALTER INDEX ... REBUILD PARTITION
 - ALTER INDEX ... SPLIT PARTITION
 - CREATE MATERIALIZED VIEW

Summary

In this lesson, you should have learned how to:

- Describe and use Oracle Database features that save space
- Create private temporary tables
- Save space by using compression
- Reclaim wasted space from tables and indexes by using the segment shrink functionality
- Manage resumable space allocation

Practice Overview

- Managing Space in Tablespaces
- Using Compression
- Enabling the Resumable Space Allocation Feature



Managing Undo Data

Copyright © 2024 Trivera Technologies LLC. | www.triveratech.com

Experience is Everything

Objectives

After completing this lesson, you should be able to:

- Explain DML and undo data generation
- Monitor and administer undo data
- Describe the difference between undo data and redo data
- Configure undo retention
- Guarantee undo retention
- Enable temporary undo

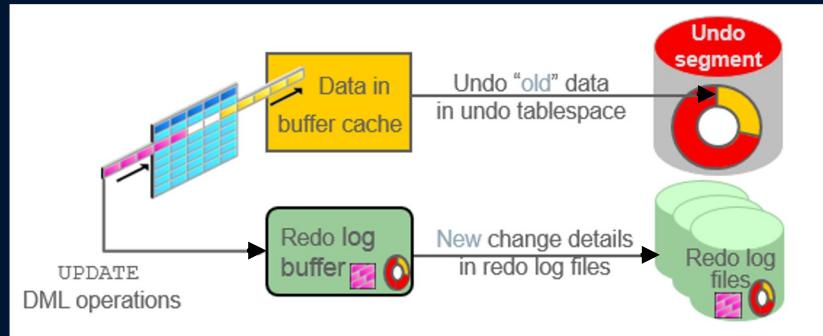
Undo Data: Overview

Undo data is:

- A record of the action of a transaction
- Captured for every transaction that changes data
- Retained at least until the transaction is ended
- Used to support:
 - Rollback operations
 - Read-consistent queries
 - Oracle Flashback Query, Oracle Flashback Transaction, and Oracle Flashback Table
 - Recovery from failed transactions

Transactions and Undo Data

- Each transaction is assigned to only one undo segment.
- An undo segment can service more than one transaction at a time.



Storing Undo Information

- Undo information is stored in undo segments, which are stored in an undo tablespace.
- Undo tablespaces:
 - Are used only for undo segments
 - Have special recovery considerations
 - May be associated with only a single instance
 - Require that only one of them be the current writable undo tablespace for a given instance at any given time

Comparing Undo Data and Redo Data

	Undo	Redo
Record of	How to undo a change	How to reproduce a change
Used for	Rollback, read consistency, flashback	Rolling forward of database changes
Stored in	Undo segments	Redo log files



Copyright © 2024 Trivera Technologies LLC. | www.triveratech.com

Managing Undo

- Automatic undo management:
 - Fully automated management of undo data and space in a dedicated undo tablespace
 - For all sessions
 - Self-tuning in AUTOEXTEND tablespaces to satisfy long-running queries
 - Self-tuning in fixed-size tablespaces for best retention
- DBA tasks in support of Flashback operations:
 - Configuring undo retention
 - Changing the undo tablespace to a fixed size
 - Avoiding space and “snapshot too old” errors

Comparing SHARED Undo Mode and LOCAL Undo Mode

- There are two undo modes in the multitenant architecture: SHARED and LOCAL.
 - There is only one SHARED undo tablespace (in CDB root).
 - There can be a LOCAL undo tablespace in each PDB.



- When is LOCAL undo mode required?
 - Hot cloning
 - Near-zero down time PDB relocation

```
SQL> STARTUP UPGRADE;
SQL> ALTER DATABASE LOCAL UNDO ON;
```

Configuring Undo Retention

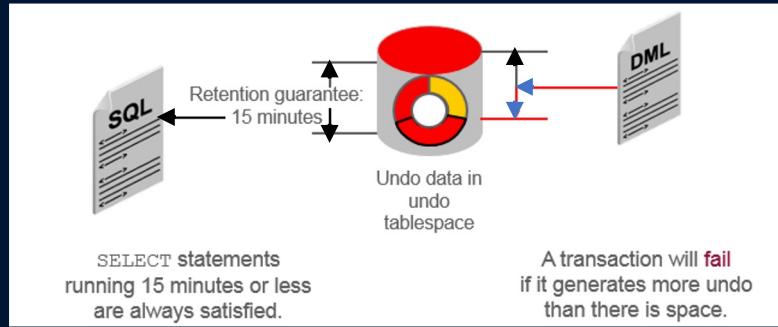
- `UNDO_RETENTION` specifies (in seconds) how long already committed undo information is to be retained.
- Set this parameter when:
 - The undo tablespace has the `AUTOEXTEND` option enabled
 - You want to set undo retention for LOBS
 - You want to guarantee retention

Categories of Undo

Category	Description
Active: Uncommitted undo information	Supports an active transaction and is never overwritten
Unexpired: Committed undo information	Is required to meet the undo retention interval
Expired: Expired undo information	Overwritten when space is required for an active transaction

Guaranteeing Undo Retention

```
SQL> ALTER TABLESPACE undotbs1 RETENTION GUARANTEE;
```

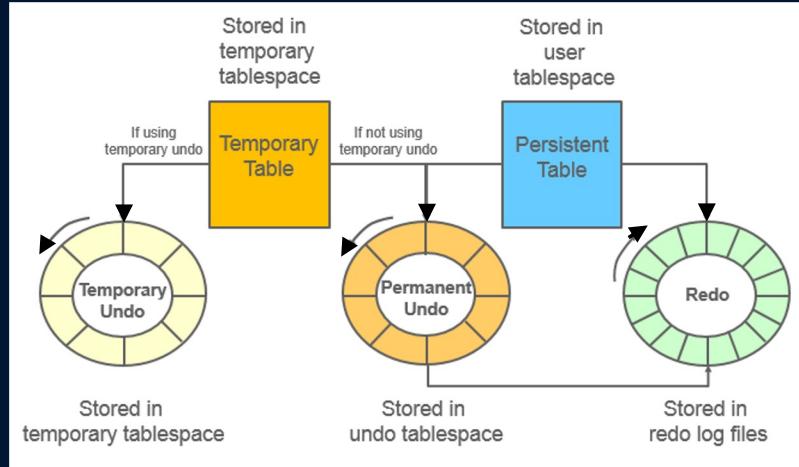


This example is based on an `UNDO_RETENTION` setting of 900 seconds (15 minutes).

Changing an Undo Tablespace to a Fixed Size

- Rationale:
 - Supporting Flashback operations
 - Limiting tablespace growth
- Steps:
 - Run the regular workload.
 - The self-tuning mechanism establishes the minimum required size.
 - (Optional) Use the Enterprise Manager Cloud Control Undo Advisor, which calculates the required size for future growth.
 - (Optional) Change the undo tablespace to a fixed size.

Temporary Undo: Overview



Temporary Undo Benefits

- Reduces the amount of undo stored in the undo tablespaces
- Reduces the amount of redo data written to the redo log
- Enables DML operations on temporary tables in a physical standby database with the Oracle Active Data Guard option

Enabling Temporary Undo

- Enable temporary undo for a session:

```
SQL> ALTER SESSION SET temp_undo_enabled = true;
```

- Enable temporary undo for the database instance:

```
SQL> ALTER SYSTEM SET temp_undo_enabled = true;
```

- Temporary undo mode is selected when a session first uses a temporary object.

Monitoring Temporary Undo

```
SQL> SELECT to_char(BEGIN_TIME,'dd/mm/yy hh24:mi:ss') "BEGIN TIME",
  2  txncount "TXNCNT", maxconcurrency, undoblkcnt, uscount "USCNT",
  3  nospaceerrcnt "NOSPEERRCNT"
  4  FROM  v$tempundostat;

BEGIN TIME          TXNCNT MAXCONCURRENCY UNDOBLKCNT USCNT NOSPEERRCNT
-----  -----
...
19/08/12 22:19:44      0            0        0      0      0
19/08/12 22:09:44      0            0        0      0      0
...
19/08/12 13:09:44      0            0        0      0      0
19/08/12 12:59:44      3            1        24     1      0
576 rows selected.
SQL>
```

Summary

In this lesson, you should have learned how to:

- Explain DML and undo data generation
- Monitor and administer undo data
- Describe the difference between undo data and redo data
- Configure undo retention
- Guarantee undo retention
- Enable temporary undo

Practice Overview

- Managing Undo Tablespaces in a PDB



Configuring User Resource Limits

Copyright © 2024 Trivera Technologies LLC. | www.triveratech.com

Experience is Everything

Objectives

After completing this lesson, you should be able to:

- Create and assign profiles to:
 - Control resource consumption
 - Manage account status and password expiration
- Use Oracle-supplied password functions in profiles

Profiles and Users

- Users are assigned only one profile at a time.
- Profiles:
 - Control resource consumption
 - Manage account status and password expiration
- `RESOURCE_LIMIT` must be set to `TRUE` (default) for profiles to impose resource limitations.

Creating Profiles in a Multitenant Architecture

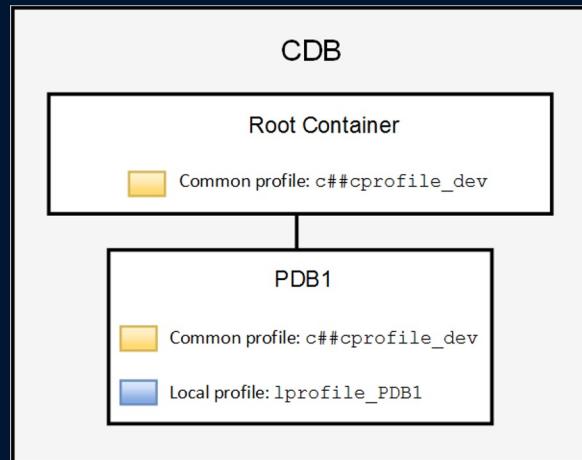
- Common profile: Replicated in all current and future containers

```
SQL> CREATE PROFILE c##cprofile_dev  
2 limit ... CONTAINER=ALL;
```

- Local profile: Created in a single PDB and can be used within that PDB only

```
SQL> CREATE PROFILE lprofile_PDB1  
2 limit ... ;
```

Creating Profiles: Example



Copyright © 2024 Trivera Technologies LLC. | www.triveratech.com

88

Profile Parameters: Resources

- In a profile, you can control:
 - CPU resources: May be limited to a per-session or per-call basis
 - Network and memory resources (Connect time, Idle time, Concurrent sessions, Private SGA)
- Disk I/O resources: Limit the amount of data a user can read at the per-session level or per-call level.
- Profiles cannot impose resource limitations on users unless the RESOURCE_LIMIT initialization parameter is set to TRUE. With RESOURCE_LIMIT at its default value of FALSE, profile resource limitations are ignored.
- Profiles also allow composite limits, which are based on weighted combinations of CPU/session, reads/session, connect time, and private SGA.

Profile Parameters: Locking and Passwords

- In a profile, specific parameters control account locking, password aging and expiration, and password history.
- Profile password settings are always enforced.
- Account locking enables automatic locking of accounts for a set duration when users fail to log in to the system in the specified number of attempts or when accounts sit inactive for a predefined number of days (users have not attempted to log in to their accounts).
- Password aging and expiration enables user passwords to have a lifetime, after which the passwords expire and must be changed.
- Password history checks the new password to ensure that the password is not reused for a specified amount of time or a specified number of password changes.
- Password complexity verification makes a complexity check on the password to verify that it meets certain rules.

Oracle-Supplied Password Verification Functions

- Complexity verification checks that each password is complex enough to provide reasonable protection against intruders who try to break into the system by guessing passwords.
- You can create your own password verification functions.
- Oracle Database provides the following functions by default:
 - ORA12C_VERIFY_FUNCTION
 - ORA12C_STRONG_VERIFY_FUNCTION
 - ORA12C_STIG_VERIFY_FUNCTION
- Password complexity checking is not enforced for the `SYS` user.

Assigning Profiles in a Multitenant Architecture

- Commonly: The profile assignment is replicated in all current and future containers.

```
SQL> CONNECT / AS SYSDBA  
SQL> ALTER USER <common user> PROFILE <common profile> CONTAINER=ALL;
```

- Locally: The profile assignment occurs in one PDB (stand-alone or application container) only.

```
SQL> CONNECT SYS@PDB1 AS SYSDBA  
SQL> ALTER USER <common or local user> PROFILE <common or local profile>;
```

Summary

In this lesson, you should have learned how to:

- Create and assign profiles to:
 - Control resource consumption
 - Manage account status and password expiration
- Use Oracle-supplied password functions in profiles

Practice Overview

- Using SQL Developer to Create a Local Profile
- Using SQL Developer to Create Local Users
- Configuring a Default Role for a User



Implementing Oracle Database Auditing

Copyright © 2024 Trivera Technologies LLC. | www.triveratech.com

Experience is Everything

Objectives

After completing this lesson, you should be able to:

- Describe DBA responsibilities for security and auditing
- Enable unified auditing
- Create unified audit policies
- Maintain the audit trail

Database Security

A secure system ensures the confidentiality of the data that it contains. There are several aspects of security:

- Restricting access to data and services
- Authenticating users
- Monitoring for suspicious activity

Monitoring for Compliance

- Monitoring or auditing must be an integral part of your security procedures.
- Review the following:
 - Mandatory auditing
 - Standard database auditing
 - Value-based auditing
 - Fine-grained auditing (FGA)

101

Types of Activities to be Audited

You can audit the following types of activities:

- User accounts, roles, and privileges
- Object actions
- Application context values
- Oracle Data Pump
- Oracle Database Real Application Security
- Oracle Database Vault
- Oracle Label Security
- Oracle Recovery Manager
- Oracle SQL*Loader direct path events

Mandatorily Audited Activities

The following activities are audited:

- CREATE/ALTER/DROP AUDIT POLICY
- AUDIT/NOAUDIT
- EXECUTE of:
 - DBMS_FGA
 - DBMS_AUDIT_MGMT
- ALTER TABLE **against AUDSYS audit trail table**
- **Top-level statements by administrative users (SYS, SYSDBA, SYSOPER, SYSASM, SYSBACKUP, SYSDG, and SYSKM) until the database opens**

Understanding Auditing Implementation

- *Mixed mode auditing* is the default when a new database is created.
- Mixed mode auditing enables the use of:
 - Pre-Oracle Database 12c auditing features
 - *Unified auditing* features
- The recommendation from Oracle is to migrate to pure unified auditing.
- Query V\$OPTION to determine if the database has been migrated to unified auditing:

```
SELECT VALUE FROM V$OPTION WHERE PARAMETER = 'Unified Auditing';

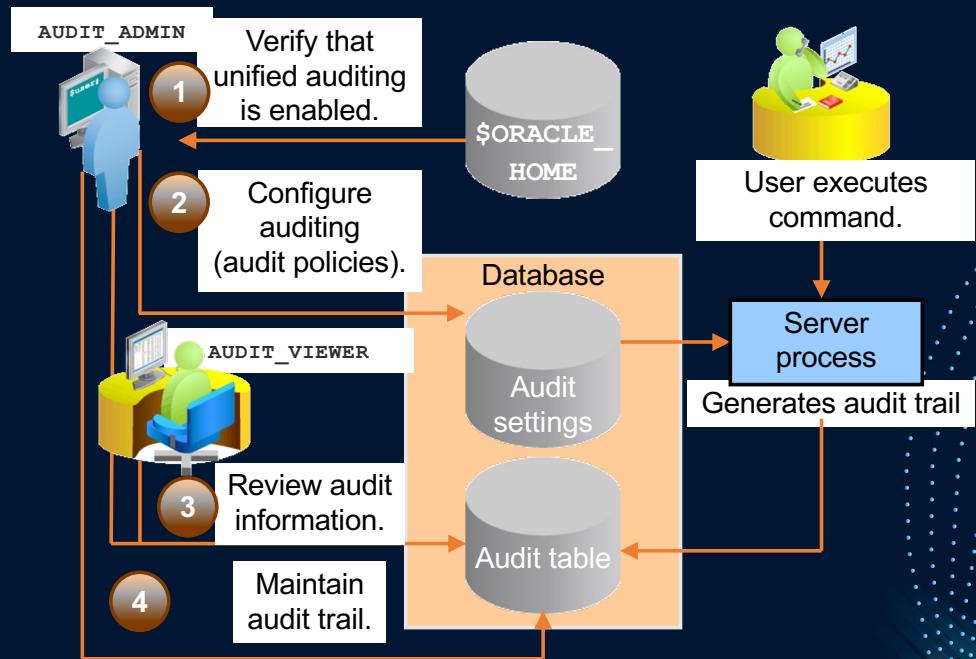
PARAMETER          VALUE
-----
Unified Auditing  TRUE
```

Administering the Roles Required for Auditing

A user must be granted one of the following roles to perform auditing:

- **AUDIT_ADMIN** enables the user to:
 - Create unified and fine-grained audit policies
 - Execute the `AUDIT` and `NOAUDIT` SQL statements
 - View audit data
 - Manage the audit trail (table in the `AUDSYS` schema)
- **AUDIT_VIEWER** enables the user to:
 - View and analyze audit data
 - Execute the `DBMS_AUDIT_UTIL` PL/SQL package

Database Auditing: Overview



Configuring Auditing

Method	Description
Unified audit policies	Group audit settings into a policy
Predefined unified audit policies	Commonly used security-relevant audit settings
Fine-grained audit policies	Define specific conditions that must be met for auditing to take place

Creating a Unified Audit Policy

- Use the CREATE AUDIT POLICY statement to create a unified audit policy:

```
CREATE AUDIT POLICY select_emp_pol  
  ACTIONS select on hr.employees
```

- To simplify policy management, group related options into a single policy.

Creating an Audit Policy: System-Wide Audit Options

- System privileges:

```
CREATE AUDIT POLICY audit_syspriv_pol1
PRIVILEGES SELECT ANY TABLE, CREATE LIBRARY
```

- Actions:

```
CREATE AUDIT POLICY audit_actions_pol2
ACTIONS AUDIT, ALTER TRIGGER
```

- Roles:

```
CREATE AUDIT POLICY audit_role_pol3
ROLES mgr_role
```

- System privileges, actions, and roles:

```
CREATE AUDIT POLICY audit_mixed_pol4
PRIVILEGES DROP ANY TABLE
ACTIONS      CREATE TABLE, DROP TABLE, TRUNCATE TABLE
ROLES        emp_role
```

Creating an Audit Policy: Object-Specific Actions

Create audit policies based on object-specific options.

```
CREATE AUDIT POLICY audit_objpriv_pol5  
  ACTIONS SELECT, UPDATE, LOCK ON hr.employees
```

```
CREATE AUDIT POLICY audit_objpriv_pol6  
  ACTIONS ALL
```

```
CREATE AUDIT POLICY audit_objpriv_pol7  
  ACTIONS EXECUTE, GRANT ON hr.raise_salary_proc
```

Creating an Audit Policy: Specifying Conditions

- Condition and evaluation **PER SESSION**

```
CREATE AUDIT POLICY audit_mixed_pol5
ACTIONS RENAME ON hr.employees,ALTER ON hr.jobs,
WHEN 'SYS_CONTEXT (''USERENV'', ''SESSION_USER'')='JIM''
EVALUATE PER SESSION
```

- Condition and evaluation **PER STATEMENT**

```
CREATE AUDIT POLICY audit_objpriv_pol6
ACTIONS ALTER ON OE.ORDERS
WHEN 'SYS_CONTEXT(''USERENV'', ''CLIENT_IDENTIFIER'')='OE''
EVALUATE PER STATEMENT
```

- Condition and evaluation **PER INSTANCE**

```
CREATE AUDIT POLICY audit_objpriv_pol7
ROLES dba
WHEN SYS_CONTEXT(''USERENV'', ''INSTANCE_NAME'')='sales''
EVALUATE PER INSTANCE
```

Enabling and Disabling Audit Policies

- Enable audit policies:

- Apply to all users.

```
AUDIT POLICY audit_syspriv_pol1;
```

- Apply only to some users.

```
AUDIT POLICY audit_pol2 BY scott, oe;  
AUDIT POLICY audit_pol3 BY sys;
```

- Exclude some users.

```
AUDIT POLICY audit_pol4 EXCEPT jim, george;
```

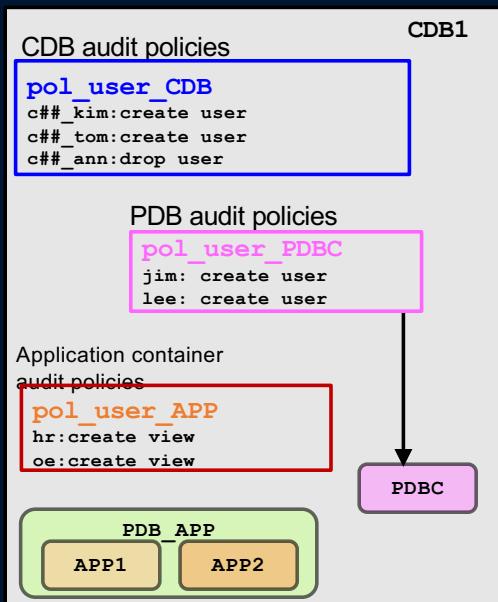
- Audit the recording based on failed or succeeded actions.

```
AUDIT POLICY audit_syspriv_pol1 WHENEVER SUCCESSFUL ;  
AUDIT POLICY audit_objpriv_pol2 WHENEVER NOT SUCCESSFUL ;
```

```
AUDIT POLICY auditpol5 BY joe WHENEVER SUCCESSFUL ;
```

- Disable audit policies by using the NOAUDIT command.

Auditing Actions in the CDB and PDBs



1. Connect to the CDB root or to an application root or to a regular PDB.
2. Create common or local unified audit policies:
 1. For all PDBs (*connect to CDB root*)
 2. For all application PDBs of an application container (*connect to the application root*)
 3. For a regular PDB or a specific application PDB (*connect to the PDB*)
3. Enable/disable audit policies:
 1. Define users or users being granted roles to be audited (*DBA role*)
 2. Use `AUDIT POLICY` and `NOAUDIT POLICY` commands

Modifying a Unified Audit Policy

- Enabled and disabled unified audit policies can be modified.
- Use the ALTER AUDIT POLICY statement to modify a unified audit policy:

```
ALTER AUDIT POLICY select_emp_pol  
ADD ACTIONS select on hr.job_history
```

Auditing Top-Level Statements Only

Top-level statement unified auditing enables you to:

- Audit a top-level user or direct user activities in the database without collecting indirect user activity

```
SQL> CREATE AUDIT POLICY actions_all_pol ACTION ALL ONLY TOLEVEL;  
SQL> AUDIT POLICY actions_all_pol BY SYS;
```

```
SQL> CREATE AUDIT POLICY update_emp_pol ACTIONS UPDATE ON HR.EMPLOYEES ONLY TOLEVEL;  
SQL> AUDIT POLICY update_emp_pol;
```

- Minimize audit records

```
SQL> CONNECT user1@PDB1  
SQL> UPDATE hr.employees SET salary = salary * 0.1 WHERE empno = 100;
```

Direct Audited

```
SQL> EXEC hr.salary_emp_raise (empno => 100, increase => '0.1')
```

Not direct
 Not audited

Viewing Audit Policy Information

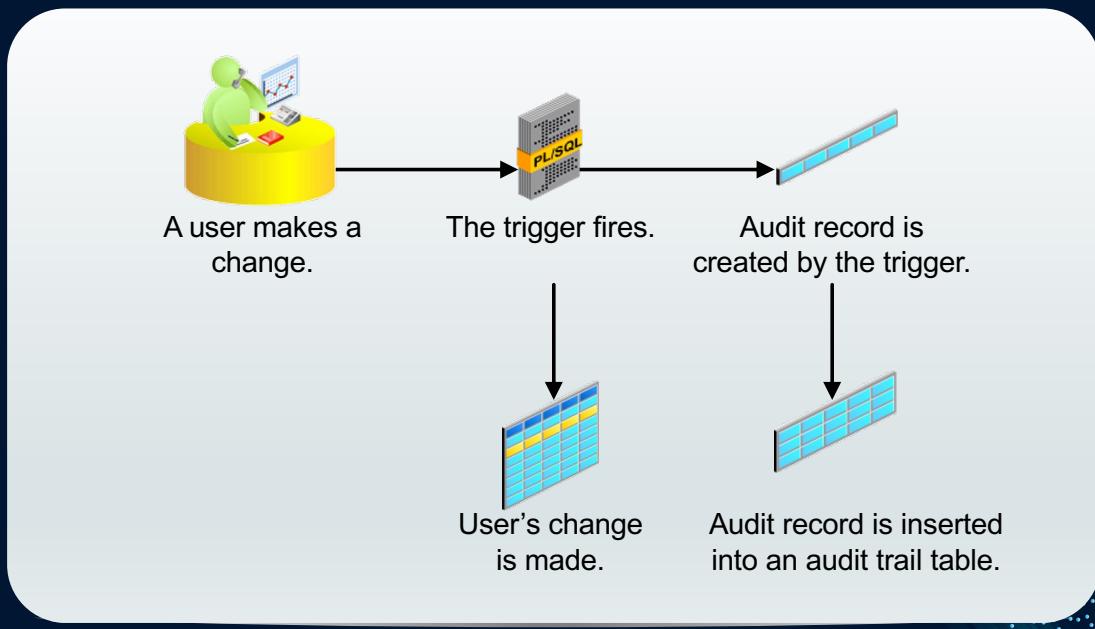
```
SQL> SELECT policy_name, audit_option, condition_eval_opt  
  2  FROM audit_unified_policies;
```

POLICY_NAME	AUDIT_OPTION	CONDITION_EVAL_OPT
POL1	DELETE	INSTANCE
POL2	TRUNCATE TABLE	NONE
POL3	RENAME	SESSION
POL4	ALL ACTIONS	STATEMENT

```
SQL> SELECT policy_name, enabled_opt, user_name, success, failure  
  2  FROM audit_unified_enabled_policies;
```

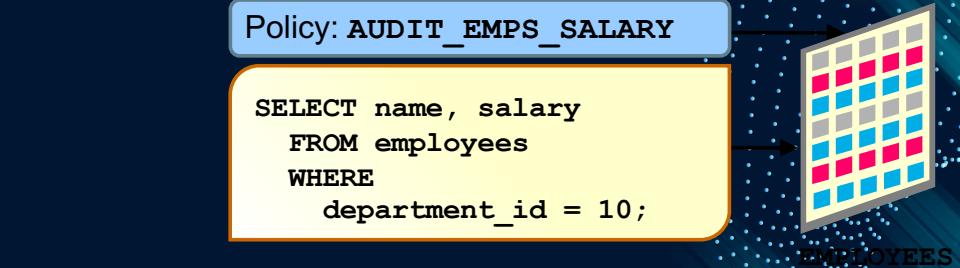
POLICY_NAME	ENABLED_	USER_NAME	SUC	FAI
POL3	BY	PM	NO	YES
POL2	EXCEPT	SYSTEM	NO	YES
POL4	BY	SYS	YES	
POL6	BY	ALL USERS	YES	NO

Value-Based Auditing



Fine-Grained Auditing

- Monitors data access on the basis of content
- Audits SELECT, INSERT, UPDATE, DELETE, and MERGE
- Can be linked to one or more columns in a table or view
- May execute a procedure
- Is administered with the DBMS_FGA package



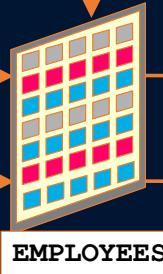
FGA Policy

- Defines:
 - Audit criteria
 - Audit action
- Is created with
DBMS_FGA.ADD_POLICY

```
dbms_fga.add_policy (
    object_schema =>      'HR',
    object_name      =>      'EMPLOYEES',
    policy_name     =>      'audit_emps_salary',
    audit_condition=>      'department_id=10',
    audit_column    =>      'SALARY,COMMISSION_PCT',
    handler_schema   =>      'secure',
    handler_module   =>
        'log_emps_salary',
    enable           =>      TRUE,
    statement_types =>      'SELECT,UPDATE');
```

SELECT name, job_id
FROM employees
WHERE
 department_id = 20;

SELECT name, salary
FROM employees
WHERE
 department_id = 10;



Not audited

SECURE.LOG_EMPS_SALARY

Audited DML Statement: Considerations

- Records are audited if the FGA predicate is satisfied and the relevant columns are referenced.
- DELETE statements are audited regardless of columns specified.
- MERGE statements are audited with the underlying INSERT, UPDATE, and DELETE generated statements.

```
UPDATE hr.employees  
SET salary = 1000  
WHERE commission_pct = .2;
```

Not audited because
none of the employees
are in department 10



```
UPDATE hr.employees  
SET salary = 1000  
WHERE employee_id = 200;
```

Audited because the
employee is in
department 10



FGA Guidelines

- To audit all rows, use a `null` audit condition.
- To audit all columns, use a `null` audit column.
- Policy names must be unique.
- The audited table or view must already exist when you create the policy.
- If the audit condition syntax is invalid, an `ORA-28112` error is raised when the audited object is accessed.
- If the audited column does not exist in the table, no rows are audited.
- If the event handler does not exist, no error is returned and the audit record is still created.

Archiving and Purging the Audit Trail

- Periodically archive and purge the audit trail to prevent it from growing too large.
- Create an archive by:
 - Copying audit trail records to a database table
 - Using Oracle Audit Vault and Database Firewall
- Purge the audit trail by:
 - Creating and scheduling a purge job to run at a specified time by using the DBMS_AUDIT_MGMT.CREATE_PURGE_JOB PL/SQL procedure
 - Manually by using the DBMS_AUDIT_MGMT.CLEAN_AUDIT_TRAIL PL/SQL procedure

Purging Audit Trail Records

- Schedule an automatic purge job:

```
DBMS_AUDIT_MGMT.CREATE_PURGE_JOB
(AUDIT_TRAIL_TYPE=> DBMS_AUDIT_MGMT.AUDIT_TRAIL_UNIFIED,
AUDIT_TRAIL_PURGE_INTERVAL => 12,
AUDIT_TRAIL_PURGE_NAME => 'Audit_Trail_PJ',
USE_LAST_ARCH_TIMESTAMP => TRUE,
CONTAINER => DBMS_AUDIT_MGMT.CONTAINER_CURRENT);
```

- Manually purge the audit records:

```
DBMS_AUDIT_MGMT.CLEAN_AUDIT_TRAIL(
AUDIT_TRAIL_TYPE => DBMS_AUDIT_MGMT.AUDIT_TRAIL_UNIFIED)
```

Summary

In this lesson, you should have learned how to:

- Describe DBA responsibilities for security and auditing
- Enable unified auditing
- Create unified audit policies
- Maintain the audit trail

Practice Overview

- Enabling Unified Auditing
- Creating Audit Users
- Creating an Audit Policy



Introduction to Loading and Transporting Data

Copyright © 2024 Trivera Technologies LLC. | www.triveratech.com

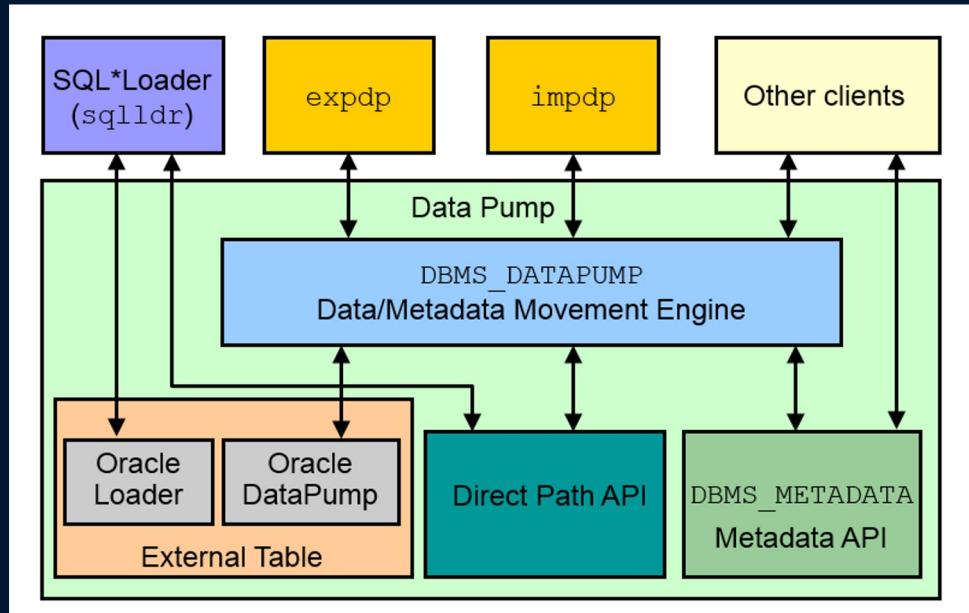
Experience is Everything

Objectives

After completing this lesson, you should be able to:

- Describe some ways to move data
- Explain the general architecture of Oracle Data Pump and SQL*Loader

Moving Data: General Architecture



Oracle Data Pump: Overview

As a server-based facility for high-speed data and metadata movement, Oracle Data Pump:

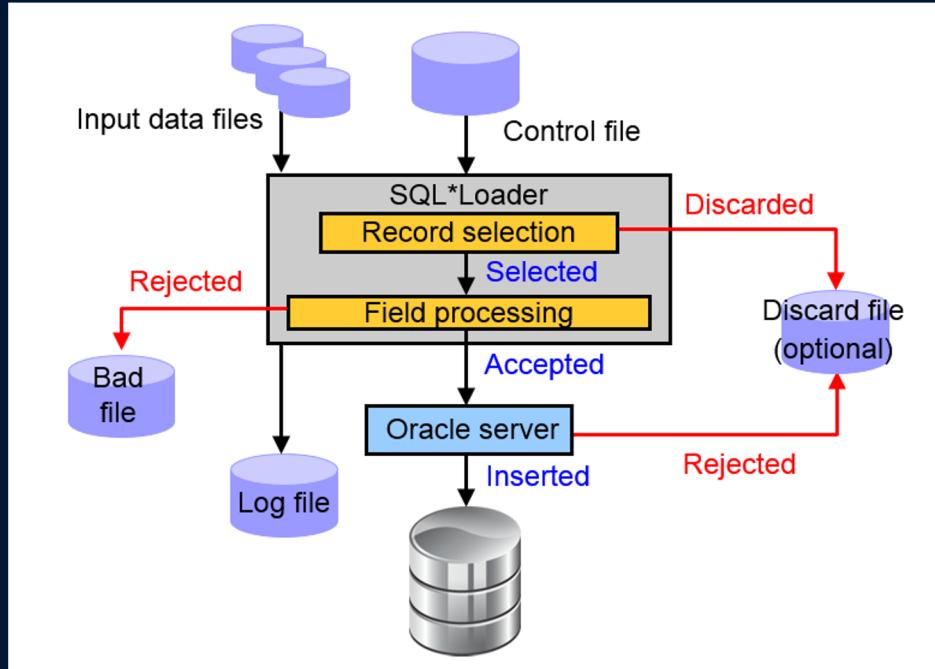
- Can be called via `DBMS_DATAPUMP`
- Provides the following tools:
 - `expdp` and `impdp`
 - GUI interface in Enterprise Manager Cloud Control
- Provides several data movement methods:
 - Conventional path load
 - Direct path
 - External tables
 - Transportable tablespace
 - Network link support
- Detaches from and reattaches to long-running jobs
- Restarts Data Pump jobs

Oracle Data Pump: Benefits

Data Pump offers many benefits and features, such as:

- Fine-grained object and data selection
- Explicit specification of database version
- Parallel execution
- Network mode in a distributed environment
- Remapping capabilities
- Data sampling and metadata compression
- Compression of data during a Data Pump export
- Security through encryption
- Ability to export XMLType data as CLOBs

SQL Loader: Overview



Summary

In this lesson, you should have learned how to:

- Describe some ways to move data
- Explain the general architecture of Oracle Data Pump and SQL*Loader



Loading Data

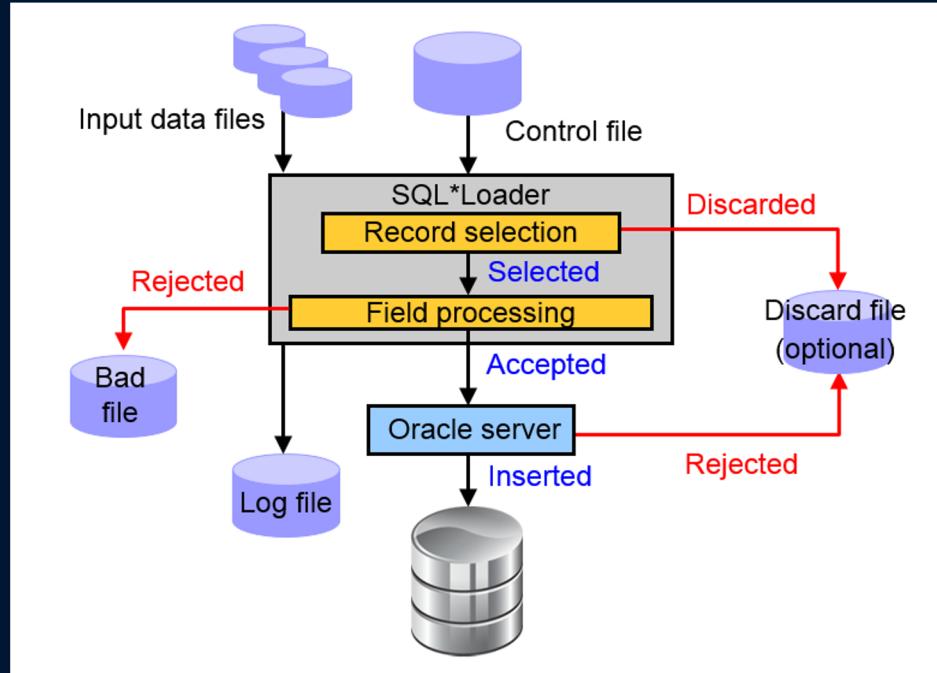
Experience is Everything

Copyright © 2024 Trivera Technologies LLC. | www.triveratech.com

Objectives

After completing this lesson, you should be able to use SQL*Loader to load data from a non-Oracle database (or user files).

SQL Loader: Review



Creating the SQL*Loader Control File

The SQL*Loader control file contains:

- Location of the data to be loaded
- Data format
- Configuration details:
 - Memory management
 - Record rejection
 - Interrupted load handling details
- Data manipulation details

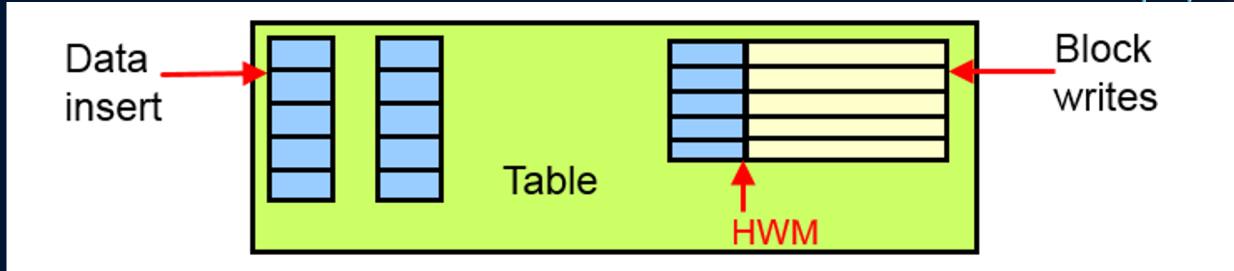
```
1      -- This is a sample control file
2 LOAD DATA
3 INFILE 'SAMPLE.DAT'
4 BADFILE 'sample.bad'
5 DISCARDFILE 'sample.dsc'
6 APPEND
7 INTO TABLE emp
8 WHEN (57) = '. '
9 TRAILING NULLCOLS
10 (hiredate SYSDATE,
     deptno POSITION(1:2) INTEGER EXTERNAL(3)
     NULLIF deptno=BLANKS,
     ...
     empno POSITION(45) INTEGER EXTERNAL
TERMINATED BY whitespace,
     ...
)
```

SQL*Loader Loading Methods

Conventional Load	Direct Path Load
Uses COMMIT	Uses data saves (faster operation)
Always generates redo entries	Generates redo only under specific conditions
Enforces all constraints	Enforces only PRIMARY KEY, UNIQUE, and NOT NULL constraints
Fires INSERT triggers	Does not fire INSERT triggers
Can load into clustered tables	Does not load into clusters
Allows other users to modify tables during load operation	Prevents other users from making changes to tables during load operation
Maintains index entries on each insert	Merges new index entries at the end of the load

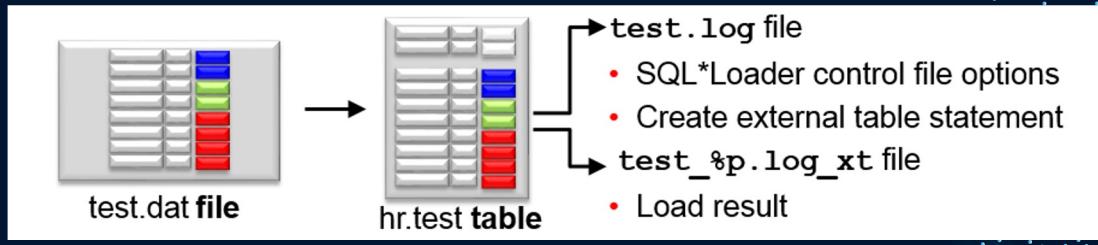
Protecting Against Data Loss

- Use data saves to protect against loss of data due to instance failure.
- Use the SQL*Loader ROWS parameter to specify when a data save should occur during a direct path load.



SQL*Loader Express Mode

- Specify a table name to initiate an express mode load.
- Table columns must be scalar data types (character, number, or datetime).
- A data file can contain only delimited character data.
- SQL*Loader uses table column definitions to determine input data types.
- There is no need to create a control file.

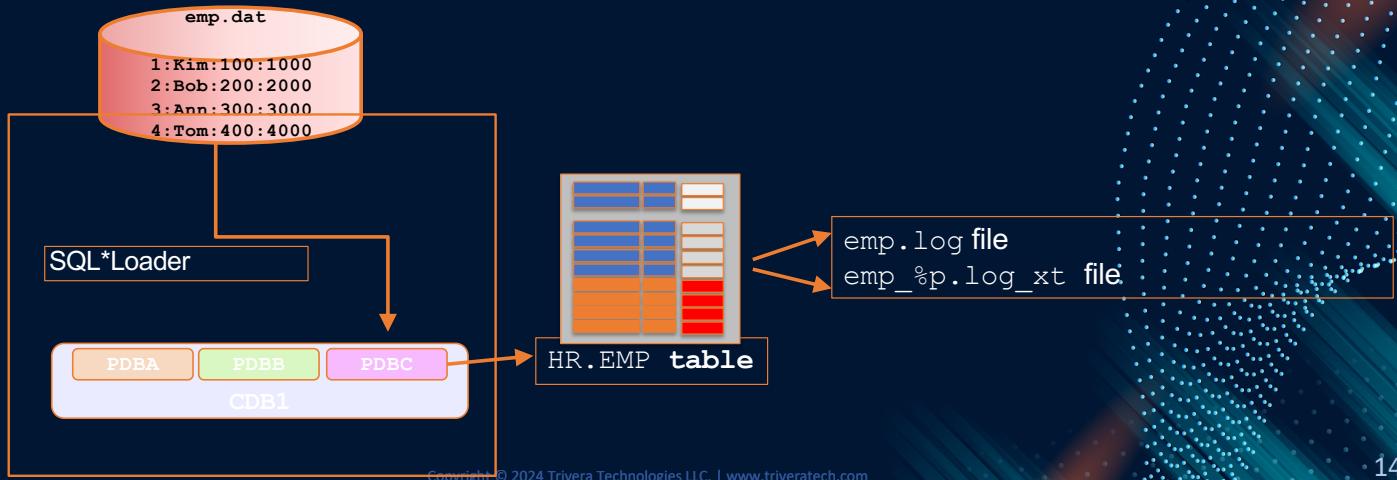


Using SQL*Loader to Load a Table in a PDB

1. Use SQL*Loader express mode to insert rows into HR.EMP in PDBC.

- No need to prepare a control file. `$ sqlldr system@PDBC TABLE=hr.emp`
- The table columns must be scalar data types (character, number, or datetime).
- SQL*Loader uses table column definitions to determine input data types.

2. Use log files to verify load operation.



146

Summary

In this lesson, you should have learned how to use external tables to move data via platform-independent files.

Practice Overview

- Loading Data into a PDB from an External File



Transporting Data

Copyright © 2024 Trivera Technologies LLC. | www.triveratech.com

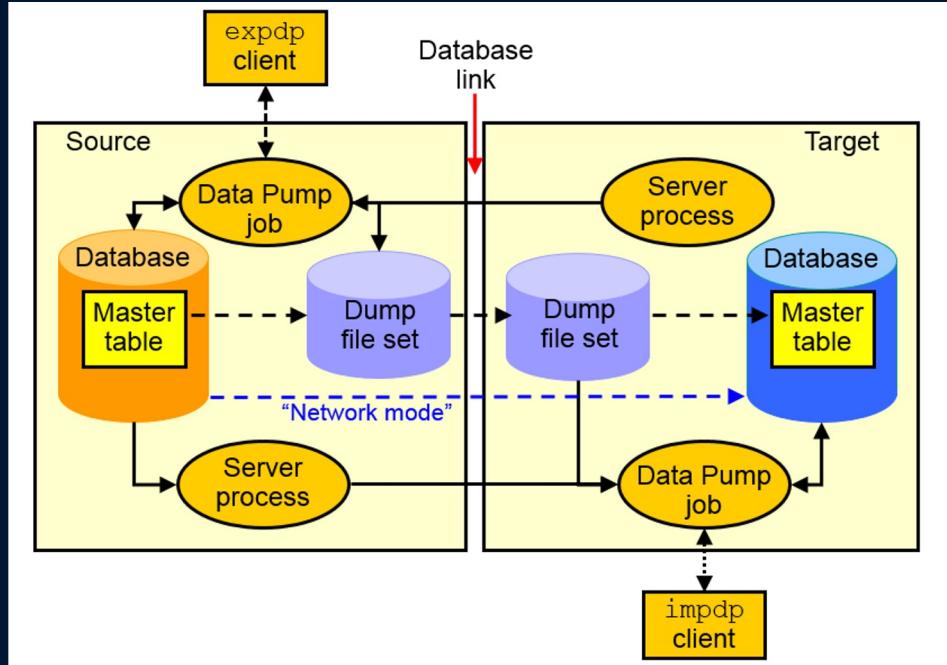
Experience is Everything

Objectives

After completing this lesson, you should be able to:

- Explain the general architecture of Oracle Data Pump
- Use Data Pump Export and Import to move data between Oracle databases
- Transport tablespaces between databases by using image copies or backup sets
- Transport databases by using data files or backup sets

Data Pump Export and Import Clients



Copyright © 2024 Trivera Technologies LLC. | www.triveratech.com

151

Data Pump Interfaces and Modes

- Data Pump Export and Import interfaces:
 - Command line
 - Parameter file
 - Interactive command line
 - Enterprise Manager Cloud Control
- Data Pump Export and Import modes:
 - Full
 - Schema
 - Table
 - Tablespace
 - Transportable tablespace
 - Transportable database

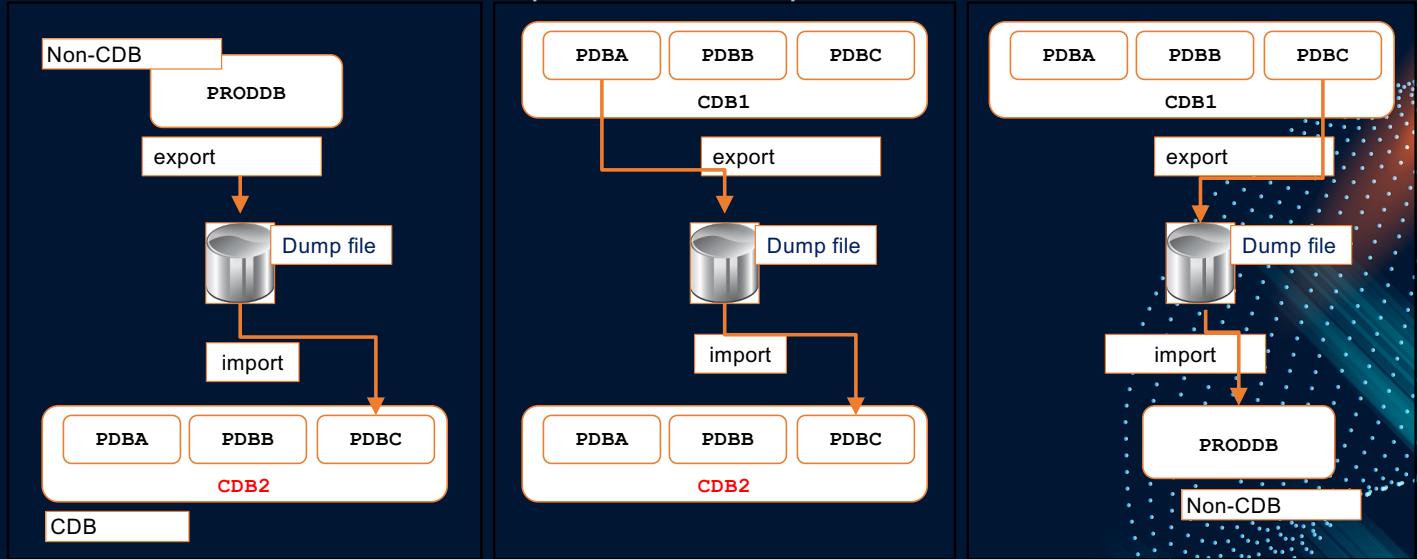
Data Pump Import Transformations

You can remap:

- Data files by using REMAP_DATAFILE
- Tablespaces by using REMAP_TABLESPACE
- Schemas by using REMAP_SCHEMA
- Tables by using REMAP_TABLE
- Data by using REMAP_DATA
- Directory by using REMAP_DIRECTORY

Using Oracle Data Pump with PDBs

Use the PDB service name to export from or import into a PDB.



Exporting from a Non-CDB and Importing into a PDB

1. Export PRODDB with the FULL=Y parameter:

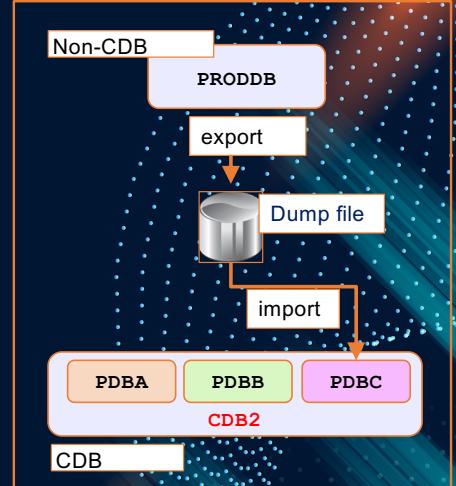
```
$ expdp system@PRODDB FULL=Y DUMPFILE=proddb.dmp
```

1. If PDBC does not exist in CDB2, create PDBC in CDB2:

```
SQL> CONNECT sys@CDB1
SQL> CREATE PLUGGABLE DATABASE PDBC ...;
```

1. Open PDBC.
2. Create a Data Pump directory in PDBC.
3. Copy the dump file to the Data Pump directory.
4. Create the same PRODDB tablespaces in PDBC for new local users' objects.
5. Import into PDBC with FULL=Y and REMAP parameter:

```
$ impdp system@PDBC FULL=Y DUMPFILE=proddb.dmp
```



Exporting and Importing Between PDBs

1. Export PDBA from CDB1 with the FULL=Y parameter:

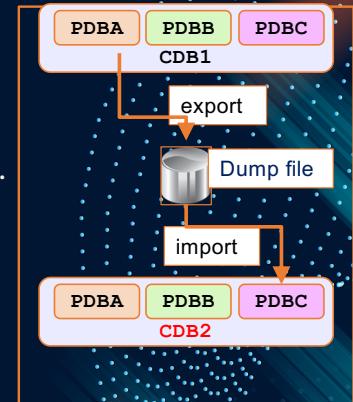
```
$ expdp system@PDBA FULL=Y ...
```

1. If PDBC does not exist in CDB2, create PDBC in CDB2:

```
SQL> CONNECT sys@CDB2  
SQL> CREATE PLUGGABLE DATABASE PDBC ...;
```

1. Open PDBC.
2. Create a Data Pump directory in PDBC.
3. Copy the dump file to the Data Pump directory.
4. Create the same PDBA tablespaces in PDBC for new local users' objects.
5. Import into PDBC of CDB2 with the FULL and REMAP parameters:

```
$ impdp system@PDBC FULL=Y REMAP_SCHEMA=c##u:lu...
```

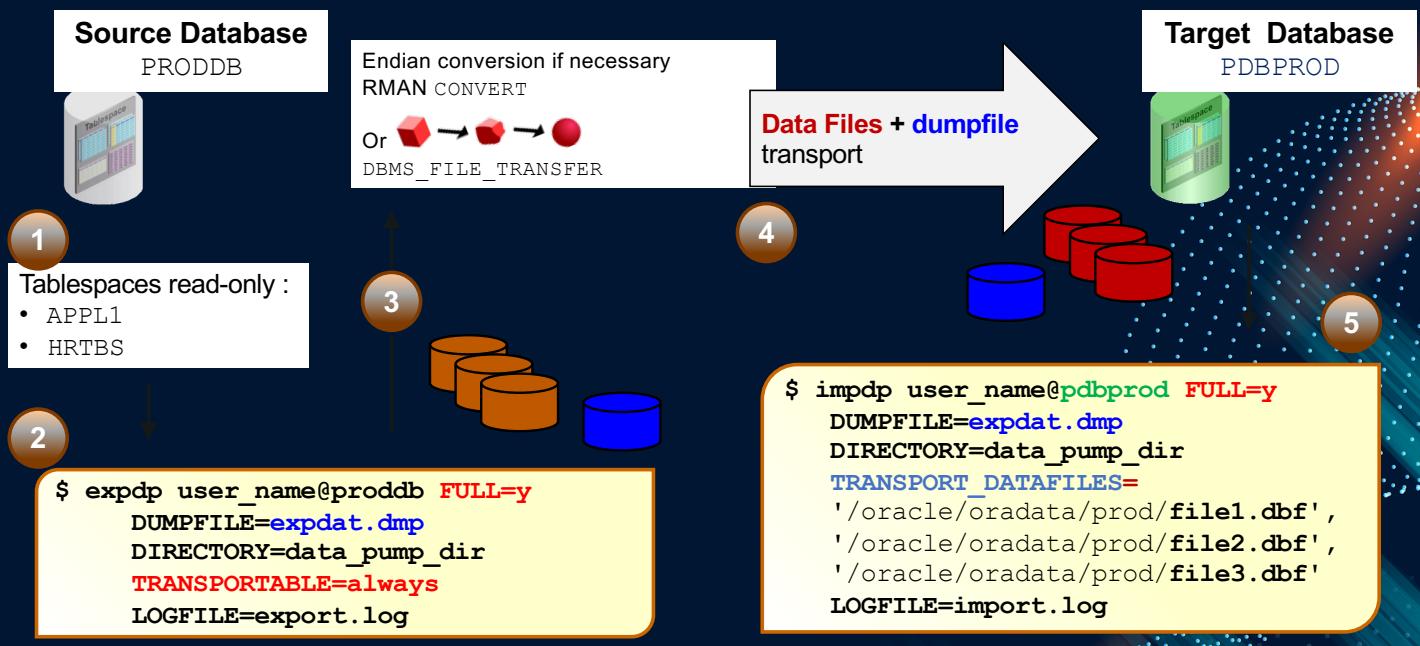


Full Transportable Export/Import

- A full transportable export exports all objects and data necessary to create a complete copy of the database. Specify these parameter values:
 - TRANSPORTABLE=ALWAYS
 - FULL=Y
- A full transportable import imports a dump file only if it has been created using the transportable option during export.
 - TRANSPORT_DATAFILES
 - If the TRANSPORTABLE parameter is specified, the NETWORK_LINK parameter is required.

```
$ expdp user_name@pdb FULL=y DUMPFILE=expdat.dmp DIRECTORY=data_pump_dir  
TRANSPORTABLE=always
```

Full Transportable Export/Import: Example



Transporting a Database Over the Network: Example

To transport a database over the network, perform an import using the `NETWORK_LINK` parameter.

1. Create a database link in the target to the source database.
2. Make the user-defined tablespaces in the source database read-only.
3. Transport the datafiles for all of the user-defined tablespaces from the source to the target location.
4. Perform conversion of the datafiles if necessary.
5. Import in the target database.

```
$ impdp username@dbname full=Y network_link = sourcedb  
    transportable = always  
    transport_datafiles = '/oracle/oradata/prod/sales01.dbf',  
                        '/oracle/oradata/prod/cust01.dbf'  
    logfile=import.log
```

Using RMAN to Transport Data Across Platforms

Transporting databases, datafiles, and tablespaces across platforms:

- Cross-platform transport (with different endian formats)
- Based on image copies and backup sets
- Use of inconsistent tablespace backups

Benefits:

- Reduced down time for platform migrations
- Choice of compression and multisession
- Not cataloged in control file, not used for regular restore operations

RMAN CONVERT Command

RMAN:

- Converts tablespaces, data files, or databases to the format of a destination platform
- Does not change input files
- Writes converted files to output destination
- Can convert on source or destination platform
- Assumes you initiate the data transfer

```
rman target sys@orcl
RMAN> ALTER TABLESPACE bartbs READ ONLY;

RMAN> CONVERT TABLESPACE bartbs
      TO PLATFORM 'Solaris Operating System (x86-64)'
      FORMAT '/tmp/transport/%U';
```

Transporting Data with Minimum Down Time

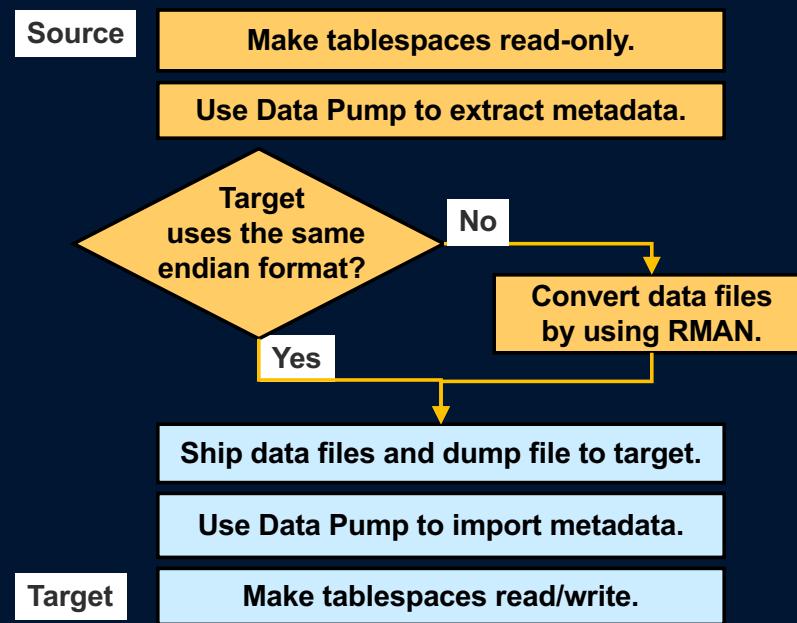
Consider the required database open mode and endian format:

- Database transport: READ ONLY, same endian format
- Tablespace transport: READ WRITE, different endian format

Example:

1. Create a database incremental level 0 backup and apply it to the destination.
2. Create incremental backups and apply them to the destination.
3. Repeat: Create and apply incremental backups.
4. Perform the final incremental backup in READ ONLY mode, apply it, and open both databases consistent with each other.

Transporting a Tablespace by Using Image Copies



Determining the Endian Format of a Platform

- Cross-platform transportable tablespaces:
 - Simplify moving data between data warehouse and data marts
 - Allow database migration from one platform to another
 - Allow the same character set on source and target platforms
- List of supported platforms and their endian formats:

```
SELECT * FROM V$TRANSPORTABLE_PLATFORM;
```

- Determine the endian format of source and target platforms:

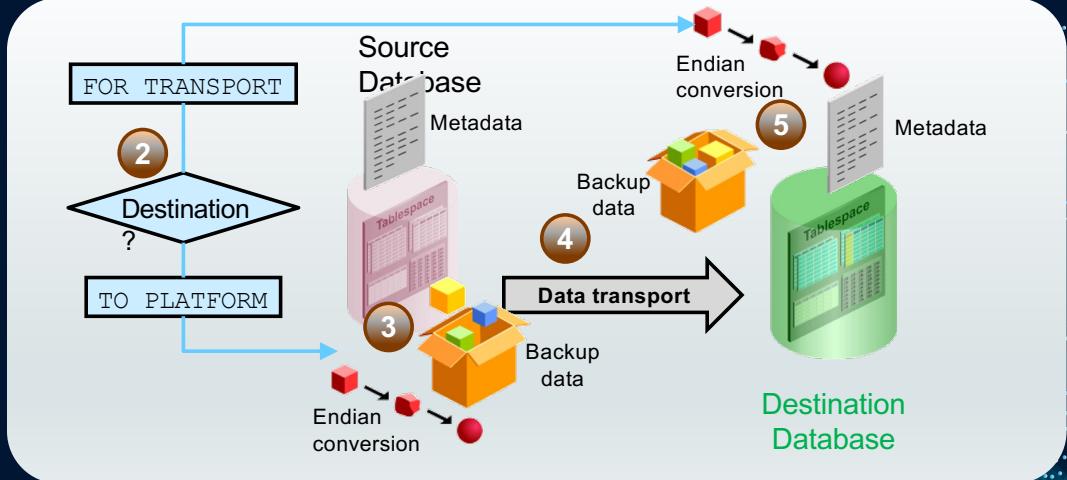
```
SELECT tp.endian_format
FROM   v$transportable_platform tp, v$database sp
WHERE  tp.platform_name = sp.platform_name;
```

Transporting Data with Backup Sets



Prerequisites

- COMPATIBLE=12.0 (or greater)
- READ ONLY mode for creation of cross-platform database backup
- Database in READ WRITE mode for creation of cross-platform tablespace backup



Transporting a Tablespace

Metadata



Source Database

1. Verify the prerequisites.
2. Start an RMAN session in the source database.
3. Query the exact name of the destination platform.
4. Change the tablespace to **read-only**.

```
RMAN> ALTER TABLESPACE test READ ONLY;
```

1. Perform a cross-platform transportable backup and a Data Pump export:
 - Conversion on the destination host

```
RMAN> BACKUP FOR TRANSPORT FORMAT '/bkp/test.bck'  
      DATAPUMP FORMAT '/bkp/test_meta.bck' TABLESPACE test;
```

- Conversion on the source host

```
RMAN> BACKUP TO PLATFORM 'HP Tru64 UNIX'  
      FORMAT '/bkp/test.bck'  
      DATAPUMP FORMAT '/bkp/test_meta.bck' TABLESPACE test;
```

Transporting a Tablespace

6. Move the backup sets and the Data Pump export dump file to the destination host.
7. Connect to the destination host as TARGET.
8. Restore the cross-transportable backup and the Data Pump export.



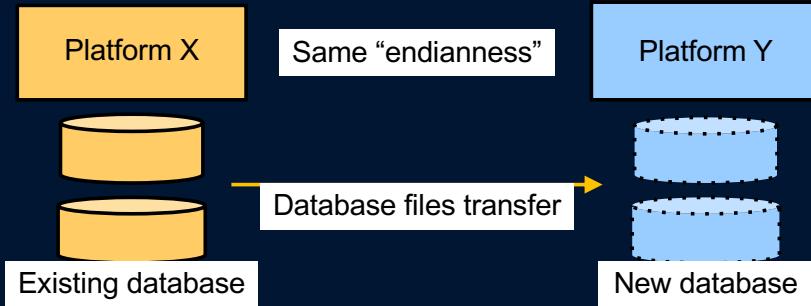
```
RMAN> RESTORE FOREIGN TABLESPACE test  
      FORMAT '/oracle/test.dbf'  
      FROM BACKUPSET '/bkp/test.bck'  
      DUMP FILE FROM BACKUPSET '/bkp/test_meta.bck' ;
```

Transporting Inconsistent Tablespaces

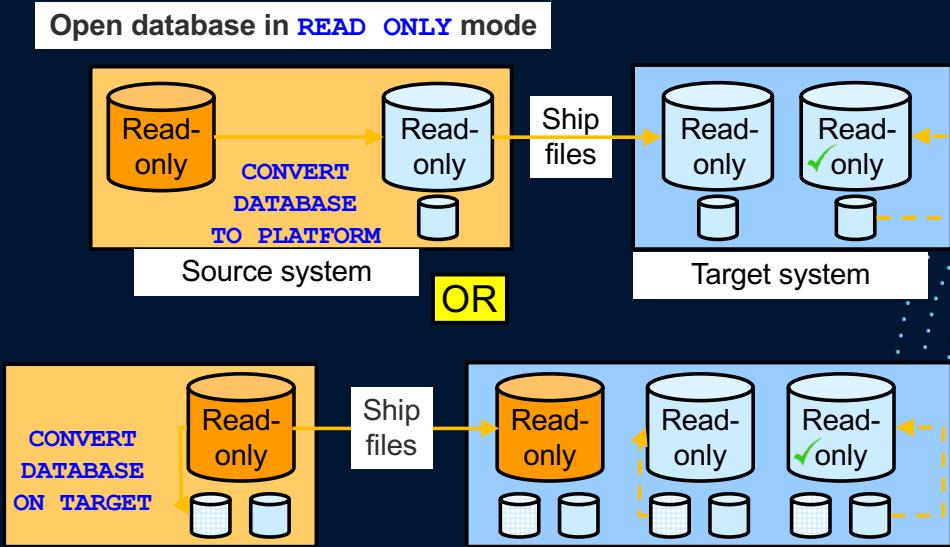
- Create cross-platform inconsistent incremental backups with the `ALLOW INCONSISTENT` clause.
- Restore the inconsistent cross-platform tablespace backup with the `RESTORE FOREIGN TABLESPACE` command.
- Recover restored data files copies with cross-platform incremental backups with the `RECOVER FOREIGN DATAFILECOPY` command.

Database Transport: Data Files

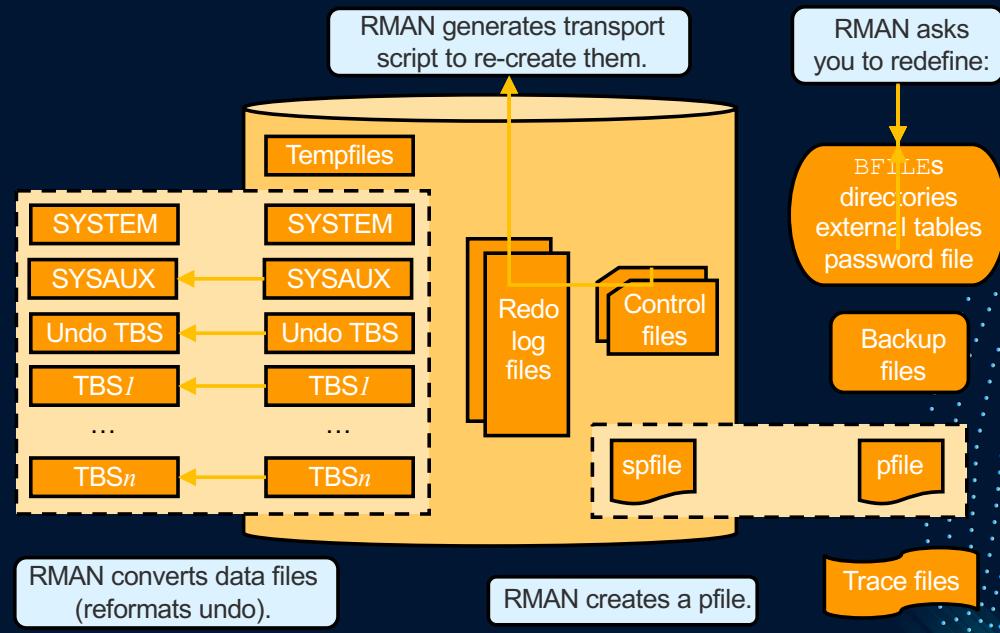
- Generalize the transportable tablespace feature.
- Data can easily be distributed from a data warehousing environment to data marts, which are usually on smaller platforms.
- A database can be migrated from one platform to another very quickly.



Transporting a Database



Transporting a Database: Conversion



Transporting a Database: Example 1

```
SQL> startup mount;
SQL> alter database open read only;
SQL> SET SERVEROUTPUT ON
SQL> DECLARE
  db_ready BOOLEAN;
BEGIN
  db_ready := DBMS_TDB.CHECK_DB('Microsoft Windows IA (32-bit)');
END ;
/
SQL> host rman target=/
RMAN> CONVERT DATABASE TRANSPORT SCRIPT 'crdb.sql' NEW DATABASE
'newdb' TO PLATFORM 'Microsoft Windows IA (32-bit)' FORMAT '/tmp/%U';

```

Source

1

2

3

4 Ship data files, PFILE, and crdb.sql

5

```
$ sqlplus / as sysdba
SQL> @crdb.sql
```

Target

Transporting a Database: Example 2

```
$ sqlplus / as sysdba
SQL> startup mount;
SQL> alter database open read only;
SQL> SET SERVEROUTPUT ON
SQL> DECLARE
db_ready BOOLEAN;
BEGIN
db_ready := DBMS_TDB.CHECK_DB('Microsoft Windows IA (32-bit)');
END;
/
SQL> host rman target=/
RMAN> CONVERT DATABASE ON TARGET PLATFORM CONVERT SCRIPT 'cnvt.sql'
TRANSPORT SCRIPT 'crdb.sql' NEW DATABASE 'newdb' FORMAT '/tmp/%U';
```

4 Ship data files, PFILE, cnvt.sql, crdb.sql

5 \$ sqlplus / as sysdba Target
SQL> host rman target=/
RMAN> @cnvt.sql
RMAN> exit;
SQL> @crdb.sql

Transporting a Database: Considerations

- Create the password file on the target platform.
- Transport the BFILEs used in the source database.
- The generated PFILE and transport script use Oracle Managed Files (OMF).
- Use DBNEWID to change the DBID.

176

Transporting a Database with Backup Sets

Metadata



Source Database

1. Verify the prerequisites:
 - COMPATIBLE: Greater or equal 12.0
 - OPEN_MODE: Read only
2. Start an RMAN session to connect to the source database.

```
RMAN> CONNECT TARGET sys/p@orcl
```

1. Query the exact name of the destination platform in V\$TRANSPORTABLE_PLATFORM:
2. Back up the source database:

Or:

```
RMAN> BACKUP TO PLATFORM='HP Tru64 UNIX'  
        FORMAT '/bkp_dir/trans_U%' DATABASE;
```

```
RMAN> BACKUP FOR TRANSPORT FORMAT '/bkp_dir/trans_U'%  
        DATABASE;
```

Transporting a Database with Backup Sets

5. Disconnect from the source database.
6. Move the backup sets and the Data Pump export dump file to the destination host.
7. Connect to the destination host as TARGET.

```
RMAN> CONNECT TARGET sys/p@orcl2
```

7. Restore the full backup set with the RESTORE command.

```
RMAN> RESTORE FOREIGN DATABASE TO NEW  
      FROM BACKUPSET '/bkp_dir/trans_U%';
```



Summary

In this lesson, you should have learned how to:

- Explain the general architecture of Oracle Data Pump
- Use Data Pump Export and Import to move data between Oracle databases
- Transport tablespaces between databases by using image copies or backup sets
- Transport databases by using data files or backup sets

Practice Overview

- Moving Data from One PDB to Another PDB
- Transporting a Tablespace



Using External Tables to Load and Transport Data

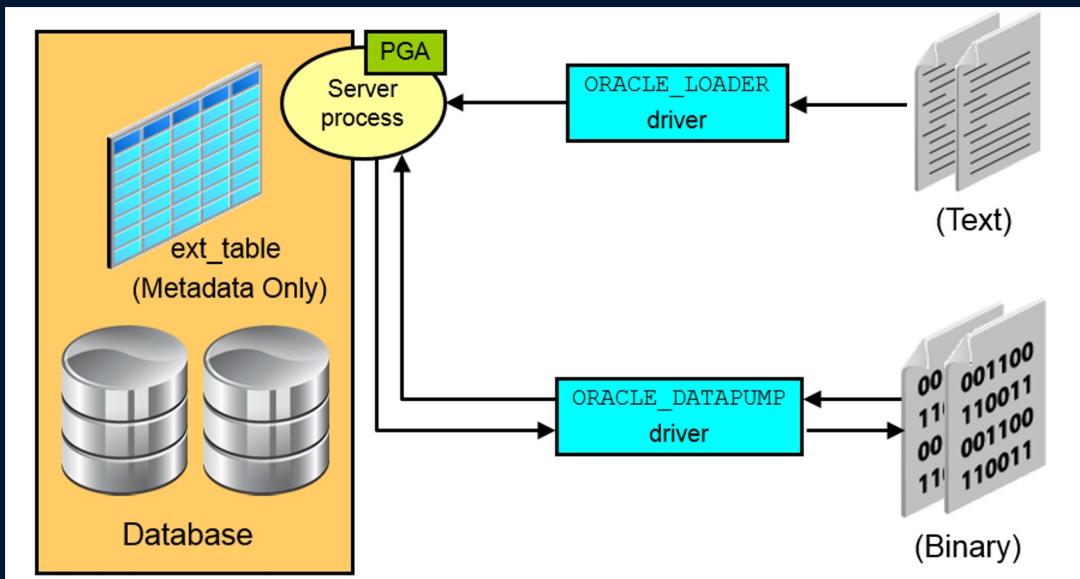
Copyright © 2024 Trivera Technologies LLC. | www.triveratech.com

Experience is Everything

Objectives

After completing this lesson, you should be able to use external tables to move data via platform-independent files.

External Tables



External Tables: Benefits

- Data can be used directly from the external file or loaded into another database.
- External data can be queried and joined directly in parallel with tables residing in the database, without requiring it to be loaded first.
- The results of a complex query can be unloaded to an external file.
- You can combine generated files from different sources for loading purposes.

ORACLE_LOADER Access Driver

```
CREATE TABLE extab_employees
  (employee_id      NUMBER(4),
   first_name       VARCHAR2(20),
   last_name        VARCHAR2(25),
   hire_date        DATE)
  ORGANIZATION EXTERNAL
    (TYPE ORACLE_LOADER
     DEFAULT DIRECTORY extab_dat_dir
     ACCESS PARAMETERS
       (records delimited by newline
        badfile extab_bad_dir:'empxt%a_%p.bad'
        logfile extab_log_dir:'empxt%a_%p.log'
        fields terminated by ','
        missing field values are null
        (employee_id, first_name, last_name,
         hire_date char date format date mask "dd-mon-yyyy"))
     LOCATION ('empxt1.dat', 'empxt2.dat'))
PARALLEL REJECT LIMIT UNLIMITED;
```

ORACLE_DATAPUMP Access Driver

```
CREATE TABLE ext_emp_query_results
  (first_name, last_name, department_name)
ORGANIZATION EXTERNAL
(
  TYPE ORACLE_DATAPUMP
  DEFAULT DIRECTORY ext_dir
  LOCATION ('emp1.exp','emp2.exp','emp3.exp')
)
PARALLEL
AS
SELECT e.first_name,e.last_name,d.department_name
FROM employees e, departments d
WHERE e.department_id = d.department_id AND
      d.department_name in ('Marketing', 'Purchasing');
```

External Tables

- Querying an external table:

```
SELECT * FROM extab_employees;
```

- Querying and joining an external table with an internal table:

```
SELECT e.employee_id, e.first_name, e.last_name, d.department_name  
FROM departments d, extab_employees e  
WHERE d.department_id = e.department_id;
```

- Appending data from an external table to an internal table:

```
INSERT /*+ APPEND */ INTO hr.employees  
SELECT * FROM extab_employees;
```

Viewing Information About External Tables

Dictionary Views	Description
[DBA ALL USER]_EXTERNAL_TABLES	Specific attributes
[DBA ALL USER]_EXTERNAL_LOCATIONS	Data sources
[DBA ALL USER]_TABLES	All tables
[DBA ALL USER]_TAB_COLUMNS	Columns of tables
[DBA ALL]_DIRECTORIES	Directory objects

Summary

- In this lesson, you should have learned how to use external tables to move data via platform-independent files.

Practice Overview

- Querying External Tables
- Unloading External Tables



Automated Maintenance Tasks: Overview

Copyright © 2024 Trivera Technologies LLC. | www.triveratech.com

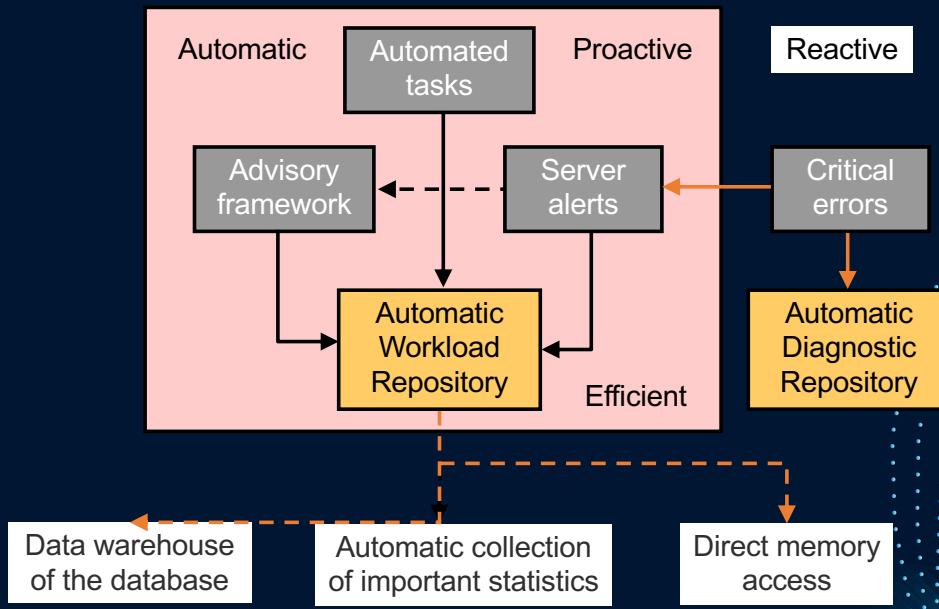
Experience is Everything

Objectives

After completing this lesson, you should be able to:

- Describe Oracle Database's proactive database maintenance infrastructure
- Discuss automated maintenance tasks
- Explain maintenance windows

Proactive Database Maintenance Infrastructure



Automated Maintenance Tasks: Components

- Automated maintenance tasks: Tasks that are started automatically at regular intervals (maintenance windows) to perform maintenance operations on the database
- Maintenance windows: Predefined time intervals that are intended to occur during a period of low system load
- Oracle Scheduler
 - An enterprise job scheduler
 - A job is created for each maintenance task scheduled to run in a maintenance window when it opens.
- Oracle Database Resource Manager
 - Enables you to manage resource allocation for a database
 - Predefined maintenance windows use the `DEFAULT_MAINTENANCE_PLAN` resource plan by default.

Predefined Automated Maintenance Tasks

Task	Description
Automatic Optimizer Statistics Collection	Collects statistics for all schema objects which have no statistics or only stale statistics
Optimizer Statistics Advisor	Analyzes how statistics are being gathered and suggests changes
Automatic Segment Advisor	Identifies segments that have reclaimable space and makes defragmenting recommendations
Automatic SQL Tuning Advisor	Examines the performance of high-load SQL statements and makes tuning recommendations
SQL Plan Management (SPM) Evolve Advisor	Evolves plans that have recently been added to the SQL plan baseline

Maintenance Windows

Maintenance windows are:

- Contiguous time intervals during which automated maintenance tasks are run
- Oracle Scheduler windows that belong to the window group named MAINTENANCE_WINDOW_GROUP

Predefined Maintenance Windows

Task	Description
MONDAY_WINDOW	Starts at 10 PM on Monday and ends at 2 AM
TUESDAY_WINDOW	Starts at 10 PM on Tuesday and ends at 2 AM
WEDNESDAY_WINDOW	Starts at 10 PM on Wednesday and ends at 2 AM
THURSDAY_WINDOW	Starts at 10 PM on Thursday and ends at 2 AM
FRIDAY_WINDOW	Starts at 10 PM on Friday and ends at 2 AM
SATURDAY_WINDOW	Starts at 6 AM on Saturday and is 20 hours long
SUNDAY_WINDOW	Starts at 6 AM on Sunday and is 20 hours long

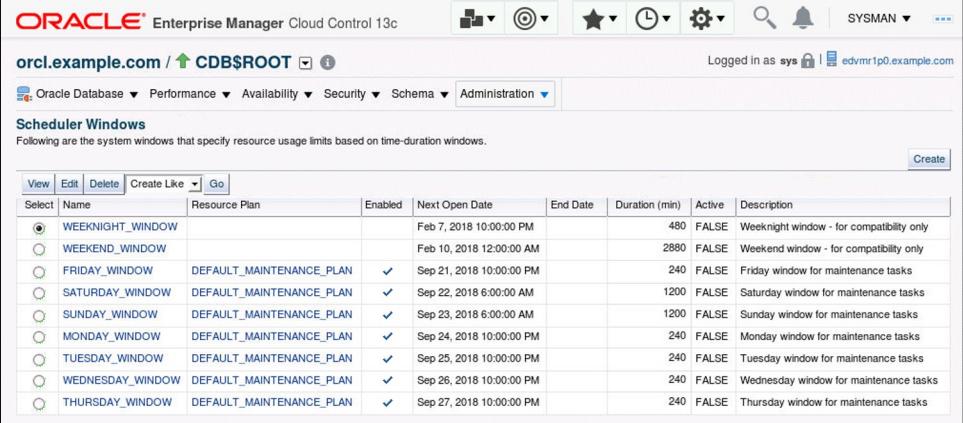
Viewing Maintenance Window Details

Maintenance Window Group

10 PM – 2 AM Mon to Fri



6 AM – 2 AM Sat to Sun



Select	Name	Resource Plan	Enabled	Next Open Date	End Date	Duration (min)	Active	Description
<input checked="" type="radio"/>	WEEKNIGHT_WINDOW			Feb 7, 2018 10:00:00 PM		480	FALSE	Weeknight window - for compatibility only
<input type="radio"/>	WEEKEND_WINDOW			Feb 10, 2018 12:00:00 AM		2880	FALSE	Weekend window - for compatibility only
<input type="radio"/>	FRIDAY_WINDOW	DEFAULT_MAINTENANCE_PLAN	✓	Sep 21, 2018 10:00:00 PM		240	FALSE	Friday window for maintenance tasks
<input type="radio"/>	SATURDAY_WINDOW	DEFAULT_MAINTENANCE_PLAN	✓	Sep 22, 2018 6:00:00 AM		1200	FALSE	Saturday window for maintenance tasks
<input type="radio"/>	SUNDAY_WINDOW	DEFAULT_MAINTENANCE_PLAN	✓	Sep 23, 2018 6:00:00 AM		1200	FALSE	Sunday window for maintenance tasks
<input type="radio"/>	MONDAY_WINDOW	DEFAULT_MAINTENANCE_PLAN	✓	Sep 24, 2018 10:00:00 PM		240	FALSE	Monday window for maintenance tasks
<input type="radio"/>	TUESDAY_WINDOW	DEFAULT_MAINTENANCE_PLAN	✓	Sep 25, 2018 10:00:00 PM		240	FALSE	Tuesday window for maintenance tasks
<input type="radio"/>	WEDNESDAY_WINDOW	DEFAULT_MAINTENANCE_PLAN	✓	Sep 26, 2018 10:00:00 PM		240	FALSE	Wednesday window for maintenance tasks
<input type="radio"/>	THURSDAY_WINDOW	DEFAULT_MAINTENANCE_PLAN	✓	Sep 27, 2018 10:00:00 PM		240	FALSE	Thursday window for maintenance tasks

198

Automated Maintenance Tasks

Autotask maintenance process:

1. The maintenance window opens.
2. The Autotask background process schedules jobs.
3. Oracle Scheduler initiates jobs.
4. Oracle Resource Manager limits the impact of Autotask jobs.

Summary

In this lesson, you should have learned how to:

- Describe Oracle Database's proactive database maintenance infrastructure
- Discuss automated maintenance tasks
- Explain maintenance windows



Automated Maintenance Tasks: Managing Tasks and Windows

Copyright © 2024 Trivera Technologies LLC. | www.triveratech.com

Experience is Everything

Objectives

After completing this lesson, you should be able to:

- Enable and disable maintenance tasks
- Create, modify, and remove maintenance windows
- Reduce or increase resource allocation to automated maintenance tasks

Configuring Automated Maintenance Tasks

You can perform the following configuration tasks:

- Adjust the duration and start time of the maintenance window.
- Control the resource plan that allocates resources to automated maintenance tasks during each window.
- Enable or disable individual tasks in some or all maintenance windows.



Enabling and Disabling Maintenance Tasks

- Enable or disable maintenance tasks for all maintenance windows:
 - Use the `ENABLE` and `DISABLE` procedures of the `DBMS_AUTO_TASK_ADMIN` package with the `WINDOW_NAME` argument set to `NULL`.
 - Use the `ENABLE` and `DISABLE` procedures with no arguments to enable or disable all automated maintenance tasks for all windows.
- Enable or disable maintenance tasks for specific maintenance windows:
 - Use the `ENABLE` and `DISABLE` procedures of the `DBMS_AUTO_TASK_ADMIN` package with the `WINDOW_NAME` argument set to a window name.



Creating and Managing Maintenance Windows

- To create a new maintenance window:
 1. Use the `DBMS_SCHEDULER.CREATE_WINDOW` procedure to create the window.
 2. Use the `DBMS_SCHEDULER.ADD_GROUP_MEMBER` procedure to add the new window to the `MAINTENANCE_WINDOW_GROUP` window group.
- To change the attributes of a maintenance window:
 1. Use the `DBMS_SCHEDULER.DISABLE` procedure to disable the window.
 2. Use the `DBMS_SCHEDULER.SET_ATTRIBUTE` procedure to modify the window attributes.
 3. Use the `DBMS_SCHEDULER.ENABLE` procedure to re-enable the window.
- To remove a maintenance window, use the `DBMS_SCHEDULER.REMOVE_GROUP_MEMBER` procedure.

Resource Allocations for Automated Maintenance Tasks

- Automated maintenance tasks run under the `ORA$AUTOTASK` subplan of the `DEFAULT_MAINTENANCE_PLAN` resource plan.
- Any resource allocation that is unused by sessions in `SYS_GROUP` is shared by sessions belonging to `OTHER_GROUPS` and `ORA$AUTOTASK` in the percentages shown below:

Consumer Group/subplan	Level 1	Maximum Utilization Limit
<code>ORA\$AUTOTASK</code>	5%	90
<code>OTHER_GROUPS</code>	20%	-
<code>SYS_GROUP</code>	75%	-

- `ORA$AUTOTASK` cannot be allocated more than 90% of the CPU resources..

Changing Resource Allocations for Maintenance Tasks

- Change the percentage of resources allocated to the ORA\$AUTOTASK subplan in the resource plan for the window of interest.
- Adjust the resource allocation for one or more subplans or consumer groups in the window's resource plan so that the resource allocation at the top level of the plan adds up to 100%.

Summary

In this lesson, you should have learned how to:

- Enable and disable maintenance tasks
- Create, modify, and remove maintenance windows
- Reduce or increase resource allocation to automated maintenance tasks

Practice Overview

- Enabling and Disabling Automated Maintenance Tasks
- Modifying the Duration of a Maintenance Window



Database Monitoring and Tuning Performance Overview

Copyright © 2024 Trivera Technologies LLC. | www.triveratech.com

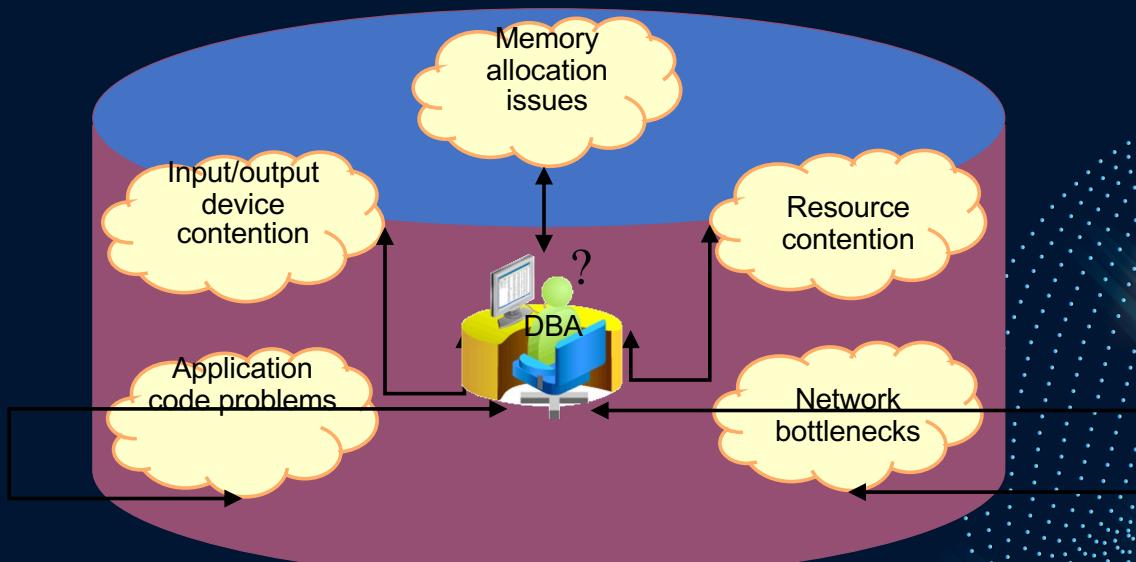
Experience is Everything

Objectives

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

- The activities that you perform to manage database performance
- The Oracle performance tuning methodology

Performance Management Activities



Copyright © 2024 Trivera Technologies LLC. | www.triveratech.com

212

Performance Planning Considerations



System Architecture Investment



Scalability



Workload Testing

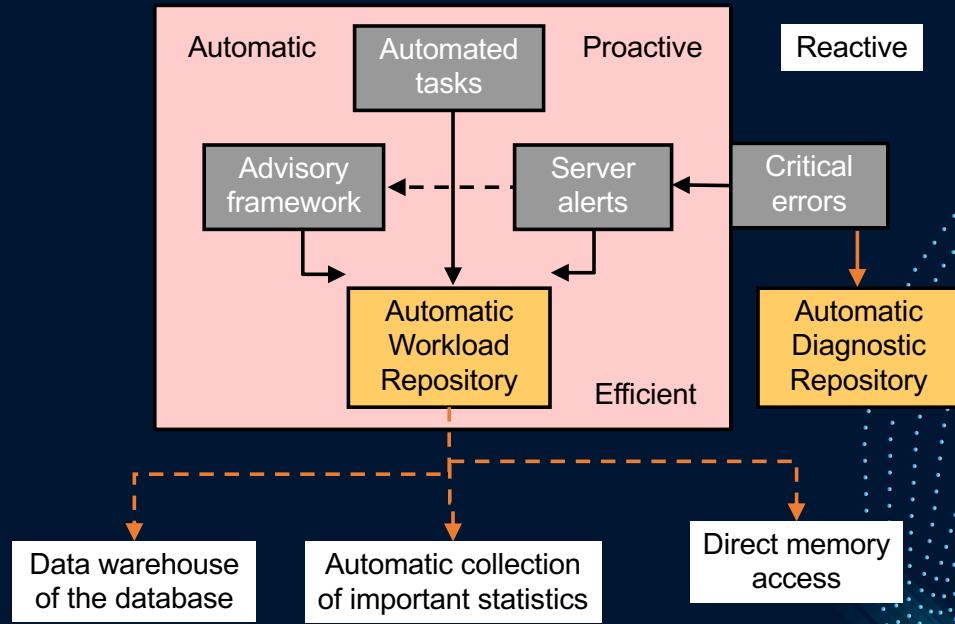


Application Design Principles



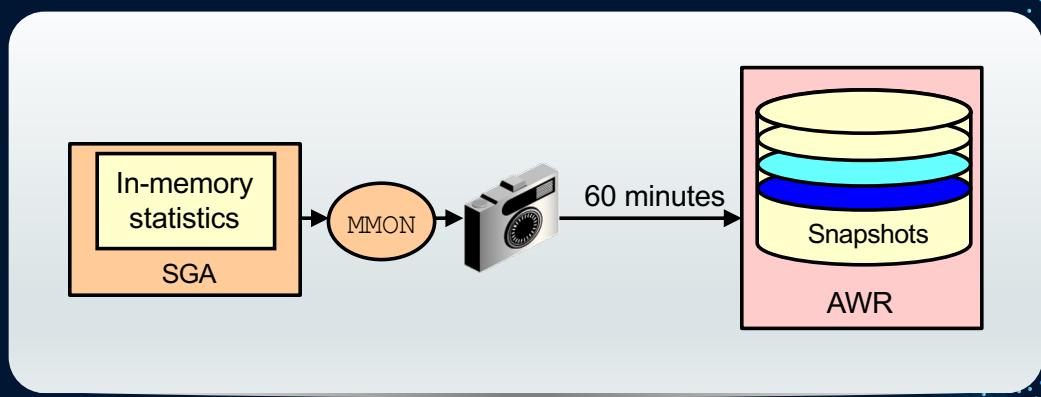
New Application Deployment

Database Maintenance



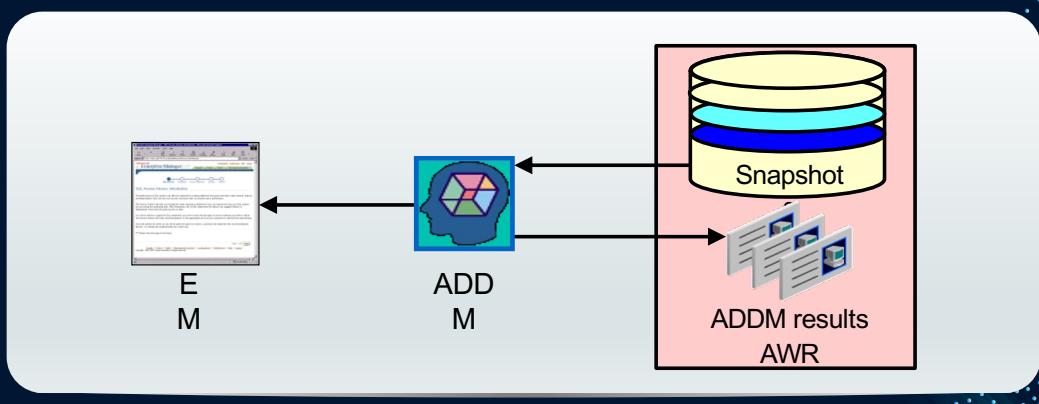
Automatic Workload Repository (AWR)

- Built-in repository of performance information
- Snapshots of database metrics taken every 60 minutes and retained for eight days
- Foundation for all self-management functions



Automatic Database Diagnostic Monitor (ADDM)

- Runs after each AWR snapshot
- Monitors the instance, detects bottlenecks
- Stores results in the AWR



Configuring Automatic ADDM Analysis at the PDB Level

1. Enable PDB AWR snapshot creation on the CDB root and on each PDB:

```
SQL> ALTER SYSTEM SET awr_pdb_autoflush_enabled = TRUE;
```

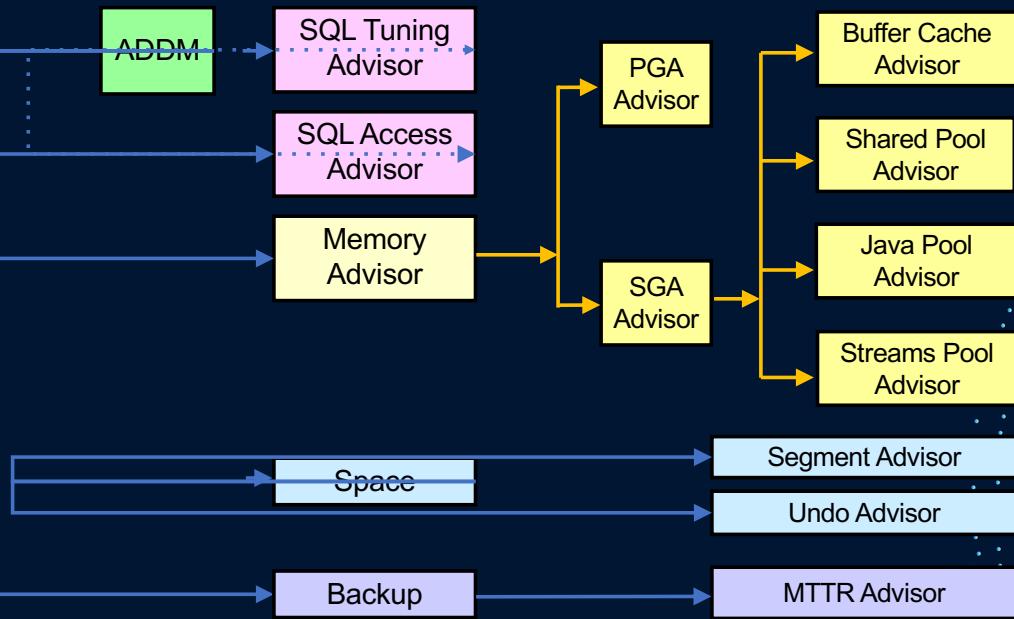
2. Set the AWR snapshot interval to greater than 0 at the PDB level:

```
SQL> CONNECT sys@PDB1 AS SYSDBA
SQL> EXEC dbms_workload_repository.modify_snapshot_settings(interval => 60)
```

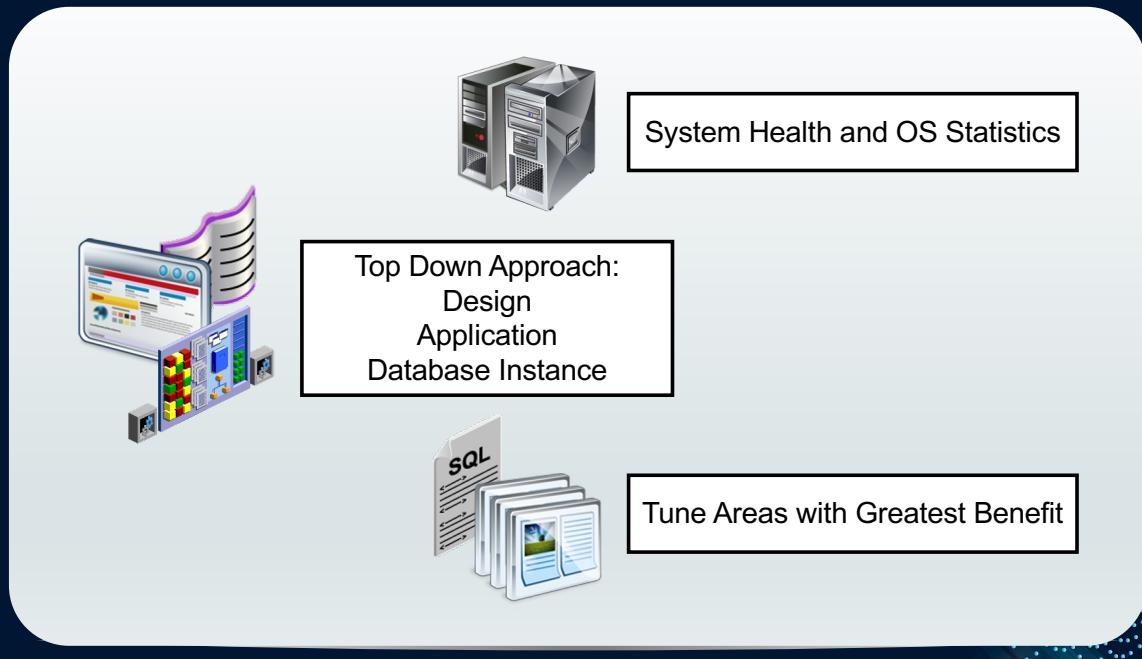
3. Execute the ADDM task (manually when required):

```
SQL> CONNECT sys@PDB1 AS SYSDBA
SQL> EXEC DBMS_ADDM.ANALYZE_DB(:tname, begin_snapshot =>1, end_snapshot =>2)
```

Advisory Framework



Performance Tuning Methodology



Copyright © 2024 Trivera Technologies LLC. | www.triveratech.com

221

Summary

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

- The activities that you perform to manage database performance
- The Oracle performance tuning methodology



14. Monitoring Database Performance

Copyright © 2024 Trivera Technologies LLC. | www.triveratech.com

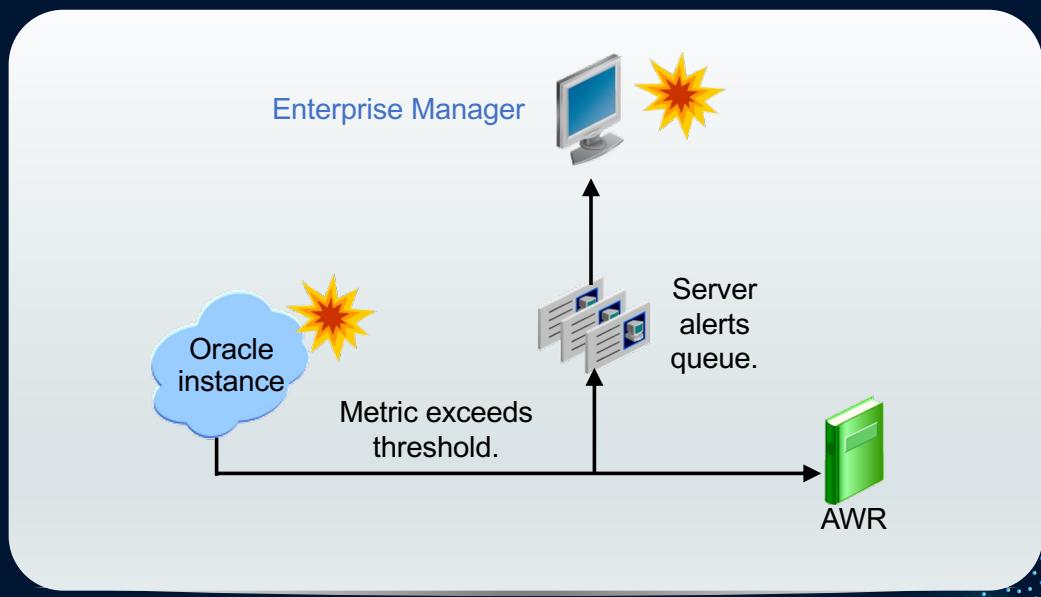
Experience is Everything

Objectives

After completing this lesson, you should be able to:

- Use performance views and tools to monitor database instance performance
- Describe the server statistics and metrics that are collected by the Oracle Database server

Server-Generated Alerts



Setting Metric Thresholds

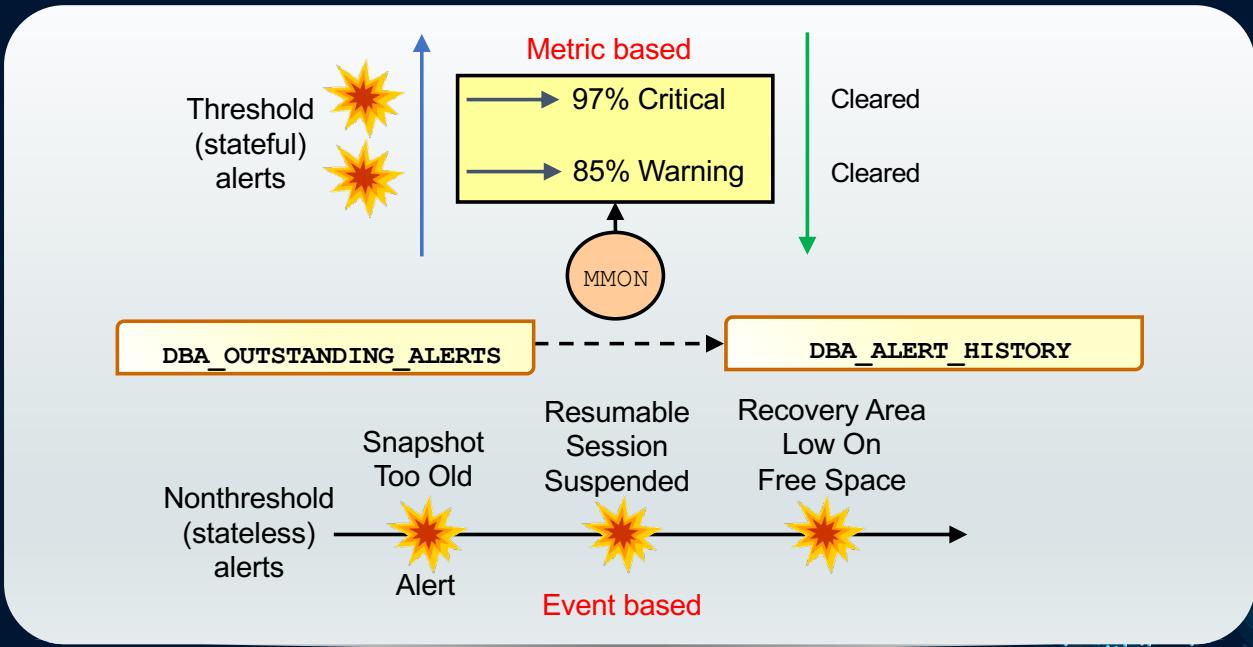
View and change threshold settings for the server alert metrics by using:

- The `GET_THRESHOLD` and `SET_THRESHOLD` procedures of the `DBMS_SERVER_ALERT` PL/SQL package
- The Metric and Collection Settings page in Enterprise Manager Cloud Control

Reacting to Alerts

- If necessary, you should gather more input (for example, by running ADDM or another advisor).
- Investigate critical errors.
- Take corrective measures.
- Acknowledge alerts that are not automatically cleared.

Alert Types and Clearing Alerts



Database Server Statistics and Metrics

Cumulative statistics:

- Wait events with time information
- Time model



Metrics: Statistic rates



Sampled statistics:

- Active session history
- Statistics by session, SQL, and service
- Other dimensions



Performance Monitoring

- Enterprise Manager Database Express
- Enterprise Manager Cloud Control
- Performance views

Instance/Database

V\$DATABASE
V\$INSTANCE
V\$PARAMETER
V\$SPPARAMETER
V\$SYSTEM_PARAMETER
V\$PROCESS
V\$BGPROCESS
V\$PX_PROCESS_SYSSTAT
V\$SYSTEM_EVENT

Disk

V\$DATAFILE
V\$FILESTAT
V\$LOG
V\$LOG_HISTORY
V\$DBFILE
V\$TEMPFILE
V\$TEMPSEG_USAGE
V\$SEGMENT_STATISTICS

Memory

V\$BUFFER_POOL_STATISTICS
V\$LIBRARYCACHE
V\$SGAINFO
V\$PGASTAT

Contention

V\$LOCK
V\$UNDOSTAT
V\$WAITSTAT
V\$LATCH

Viewing Statistics Information

V\$SYSSTAT

- STATISTIC#
- NAME
- CLASS
- VALUE
- STAT_ID

V\$SYSTEM_WAIT_CLASSES

- WAIT_CLASS_ID
- WAIT_CLASS#
- WAIT_CLASS
- TOTAL_WAITS
- TIME_WAITED

V\$SGASTAT

- POOL
- NAME
- BYTES

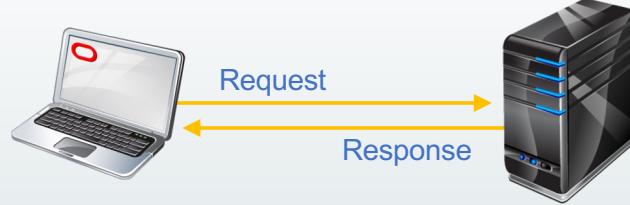
V\$EVENT_NAME

- EVENT_NUMBER
- EVENT_ID
- NAME ←
- PARAMETER1
- PARAMETER2
- PARAMETER3
- WAIT_CLASS

V\$SYSTEM_EVENT

- EVENT
- TOTAL_WAITS
- TOTAL_TIMEOUTS
- TIME_WAITED
- AVERAGE_WAIT
- TIME_WAITED_MICRO

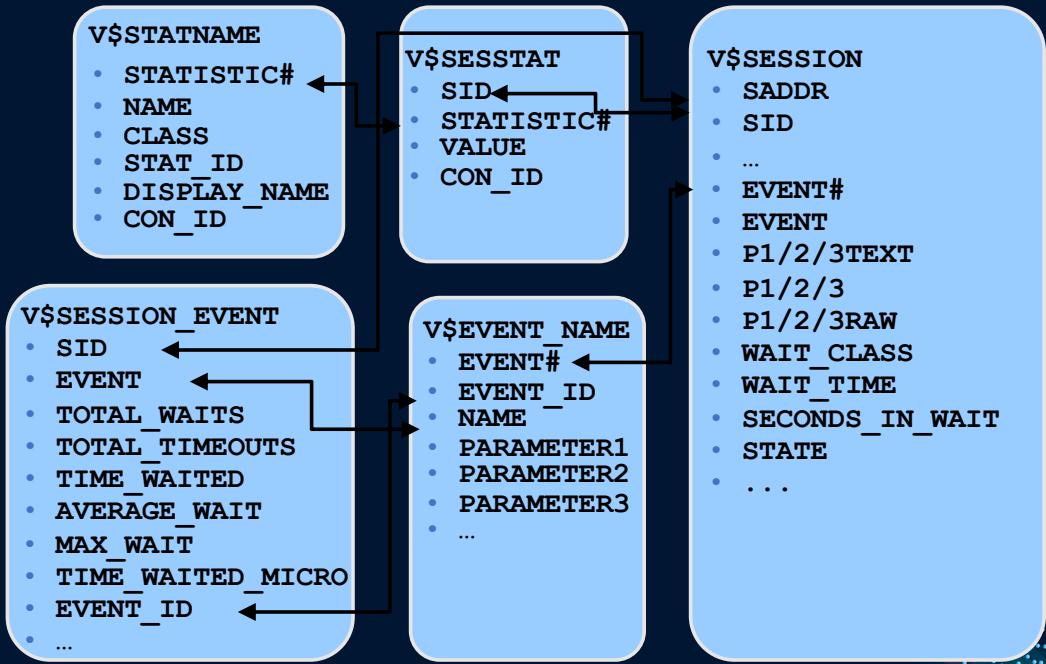
Monitoring Wait Events



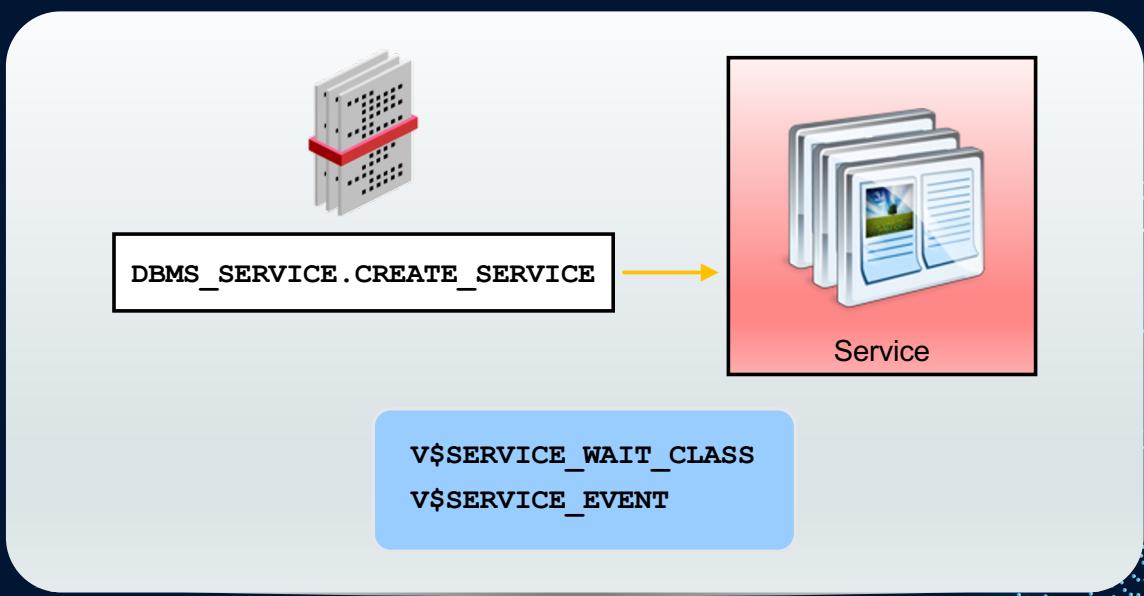
Wait events: Statistics indicating the server process had to wait for an event to complete

V\$EVENT_NAME

Monitoring Sessions



Monitoring Services



Copyright © 2024 Trivera Technologies LLC. | www.triveratech.com

235

Summary

In this lesson, you should have learned how to:

- Use performance views and tools to monitor database instance performance
- Describe statistics and metrics that are collected by the Oracle Database server

Practice Overview

- Using Enterprise Manager Database Express to Manage Performance



Database Processes

Copyright © 2024 Trivera Technologies LLC. | www.triveratech.com

Experience is Everything

Objectives

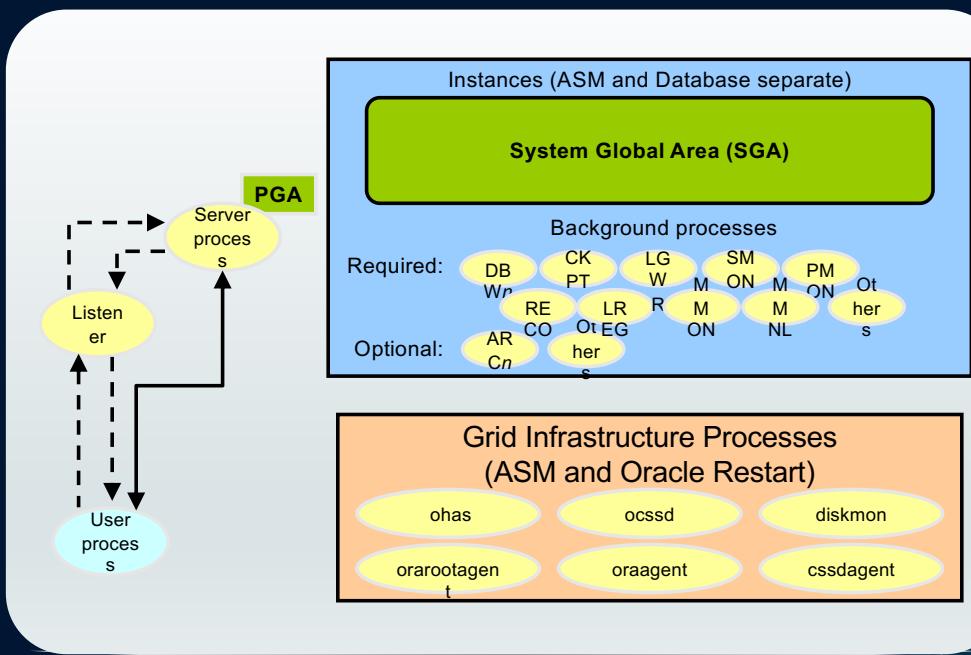
After completing this lesson, you should be able to

- Describe background processes

Process Architecture

- User process
 - Is the application or tool that connects to the Oracle database
- Database processes
 - Server process: Connects to the Oracle instance and is started when a user establishes a session
 - Background processes: Are started when an Oracle instance is started
- Daemon / Application processes
 - Networking listeners
 - Grid Infrastructure daemons

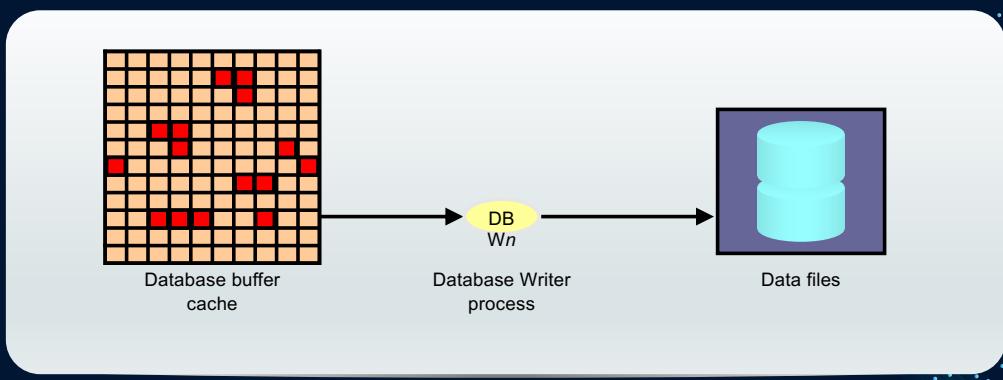
Process Structures



Database Writer Process (DBW n & BW nn)

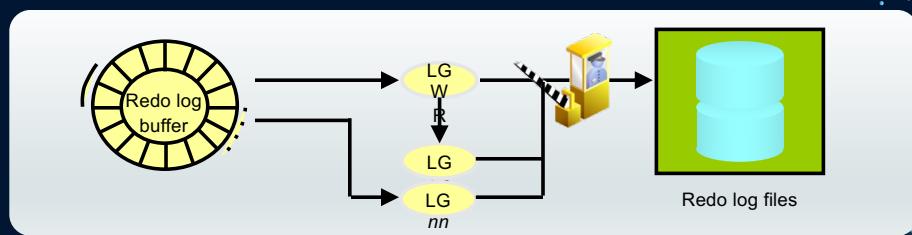
Writes modified (dirty) buffers in the database buffer cache to disk:

- Asynchronously while performing other processing
- To advance the checkpoint



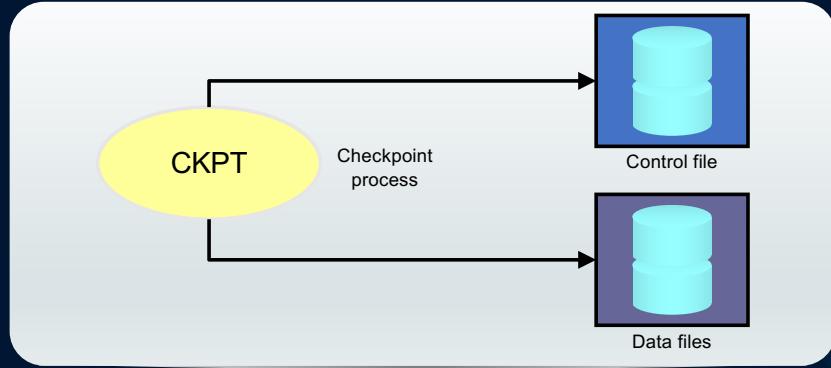
Log Writer Process (LGWR & LGnn)

- Writes the redo log buffer to a redo log file on disk:
 - When a user process commits a transaction
 - When an online redo log switch occurs
 - When the redo log buffer is one-third full or contains 1 MB of buffered data
 - Before a DBW n process writes modified buffers to disk
 - When three seconds have passed since the last write
- Serves as coordinator of LGnn processes and ensures correct order for operations that must be ordered



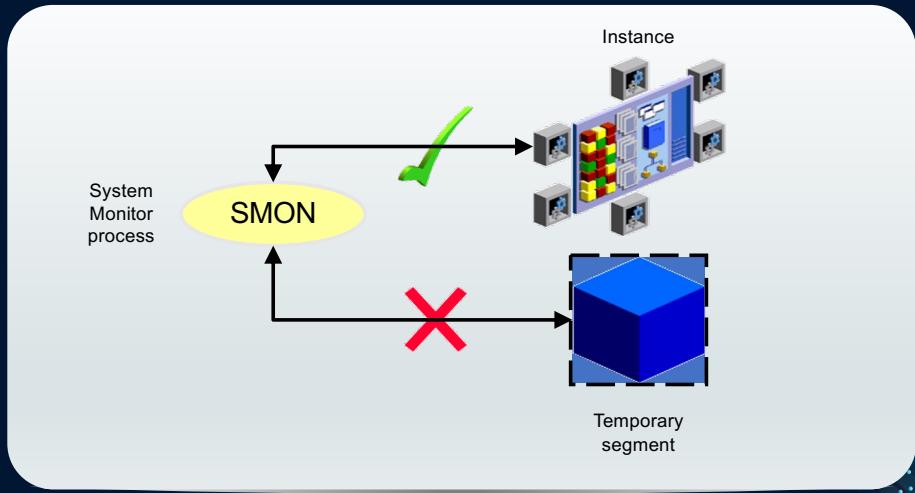
Checkpoint Process (CKPT)

- Records checkpoint information in:
 - The Control file
 - Each data file header
- Signals DBW n to write blocks to disk



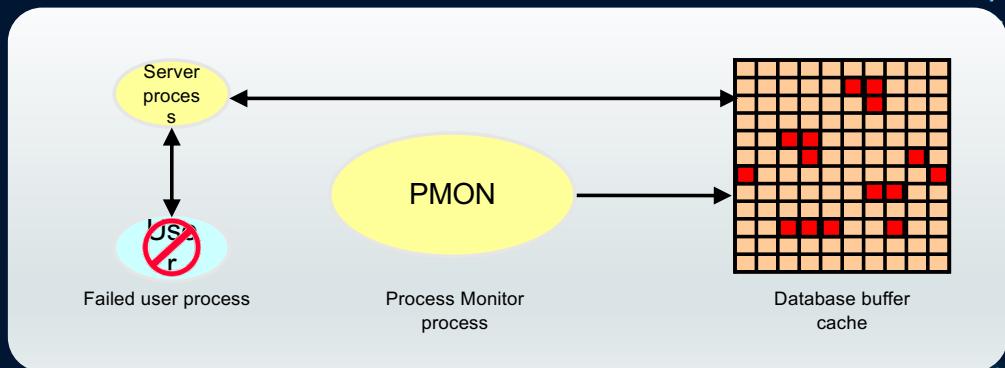
System Monitor Process (SMON)

- Performs recovery at instance startup
- Cleans up unused temporary segments



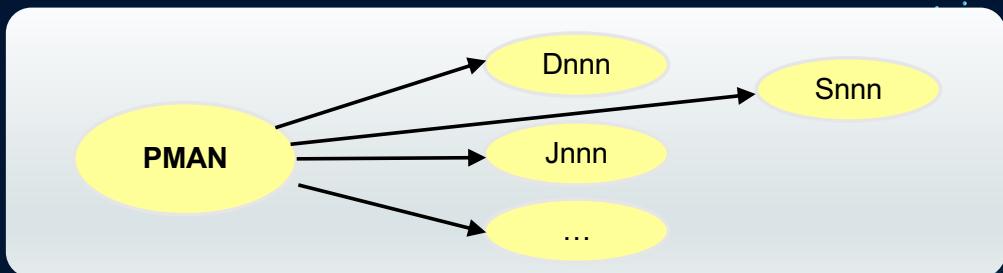
Process Monitor Process (PMON)

- Performs process recovery when a user process fails
 - Cleans up the database buffer cache
 - Frees resources that are used by the user process
- Monitors sessions for idle session timeout



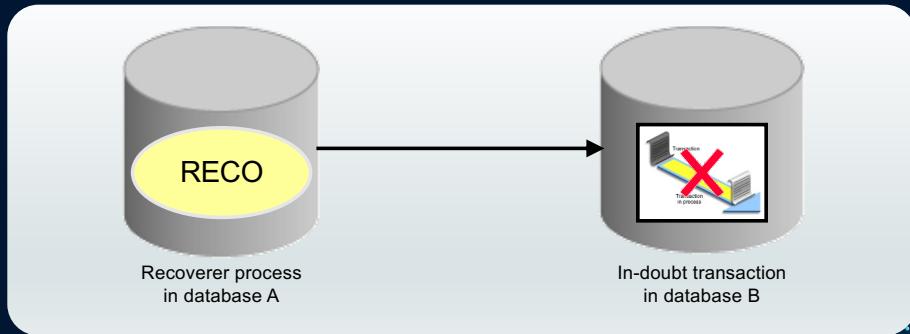
Process Manager (PMAN)

- PMAN monitors, spawns, and stops the following types of processes as needed:
 - Dispatcher and shared server processes
 - Connection broker and pooled server processes for database resident connection pools
 - Job queue processes
 - Restartable background processes



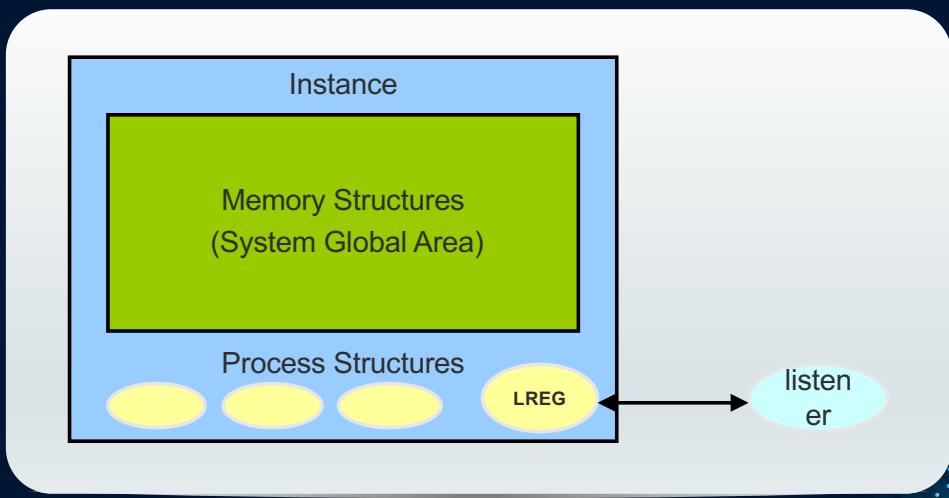
Recoverer Process (RECO)

- Used with the distributed database configuration
- Automatically connects to other databases involved in in-doubt distributed transactions
- Automatically resolves all in-doubt transactions
- Removes any rows that correspond to in-doubt transactions



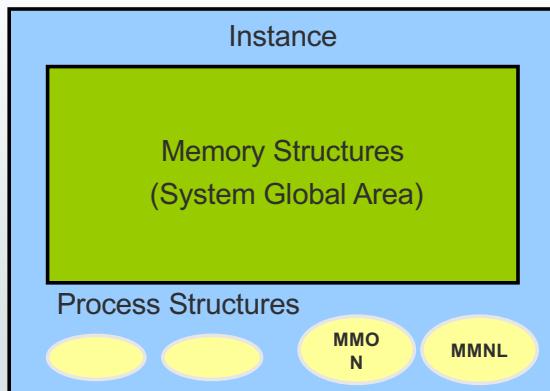
Listener Registration Process (LREG)

Registers information about the database instance and dispatcher processes with Oracle Net Listener



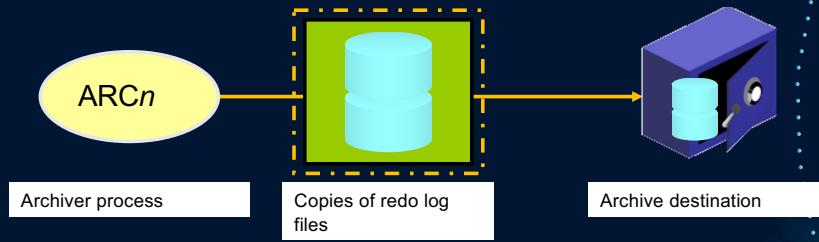
Manageability Monitor Process (MMON)

Performs or schedules many manageability task

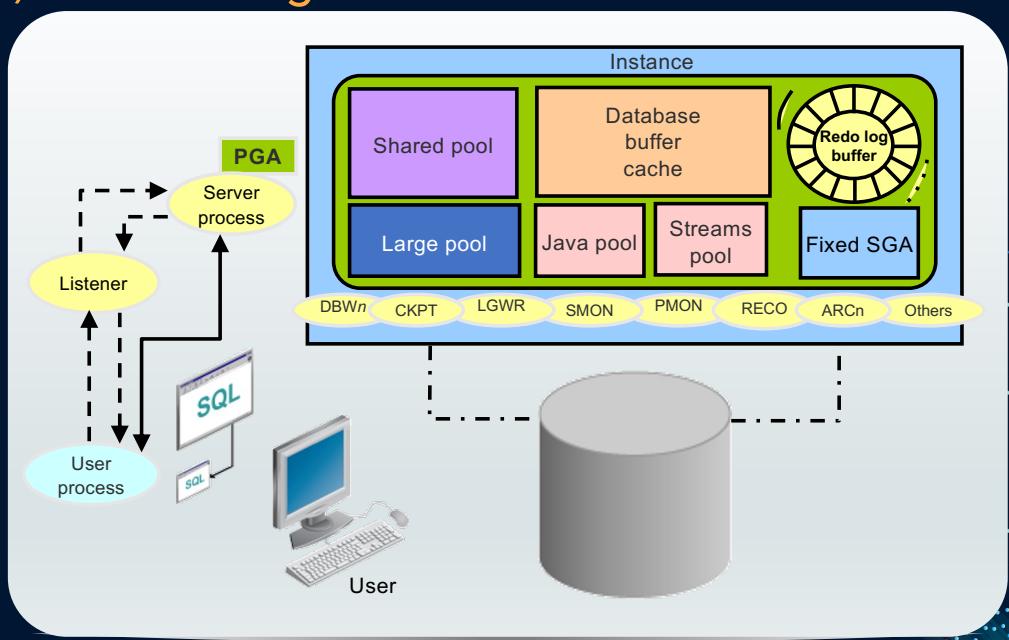


Archiver Processes (ARCn)

- Copy redo log files to a designated storage device after a log switch has occurred
- Can collect transaction redo data and transmit that data to standby destinations



Interacting with an Oracle Database: Memory, Processes, and Storage



Copyright © 2024 Trivera Technologies LLC. | www.triveratech.com

255

Summary

In this lesson, you should have learned how to configure and monitor memory components for optimal performance.

Practice Overview

- Examining the Database Background Processes
- Identifying the Database Foreground Server Processes



Managing Memory

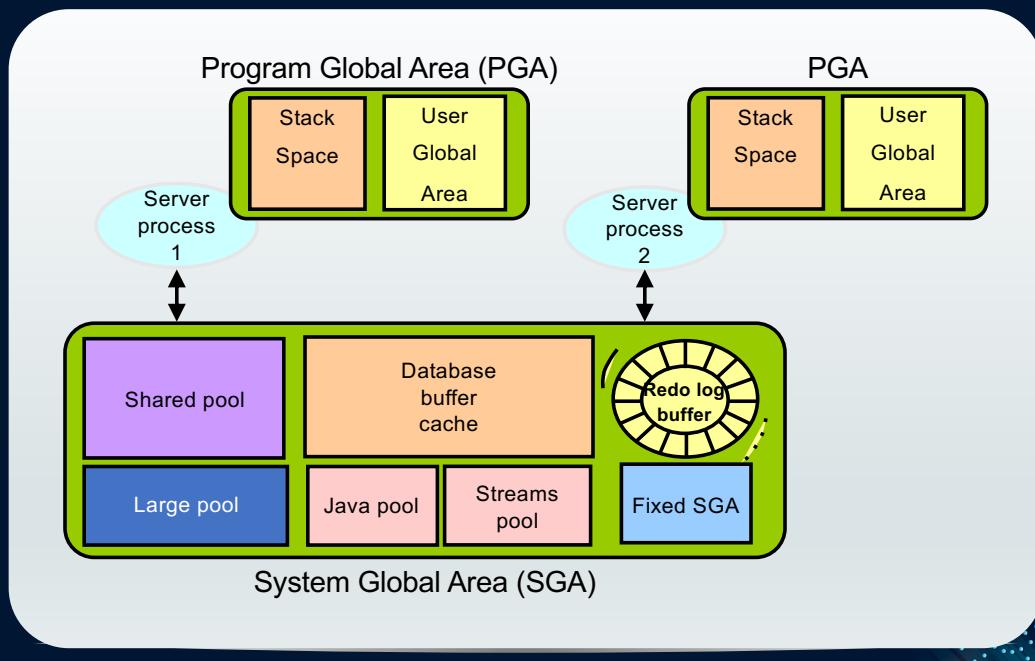
Copyright © 2024 Trivera Technologies LLC. | www.triveratech.com

Experience is Everything

Objectives

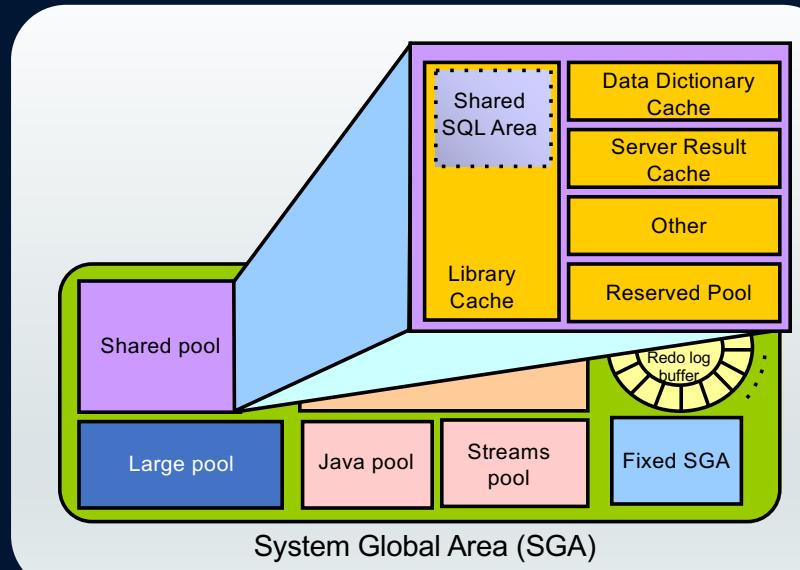
After completing this lesson, you should be able to configure and monitor memory components for optimal performance.

Managing Memory Components



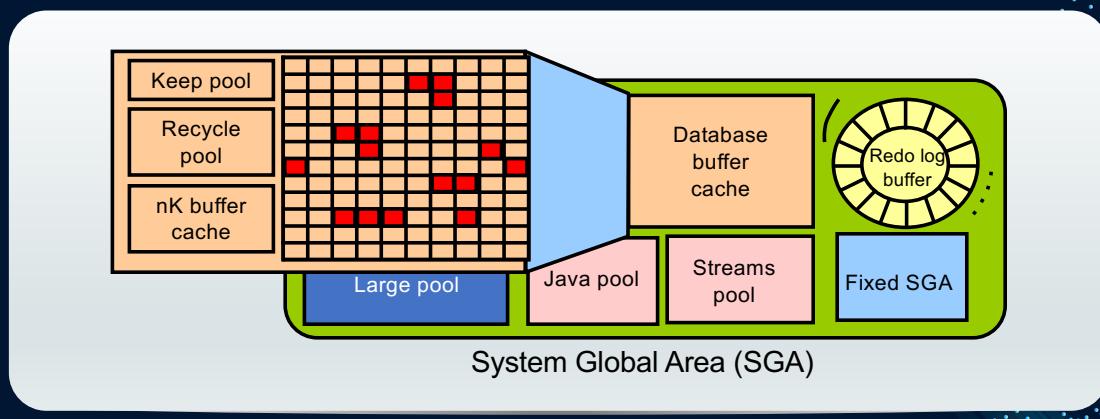
Shared Pool

- Is a portion of the SGA
- Contains:
 - Library cache
 - Shared SQL area
 - Data dictionary cache
 - Server result cache



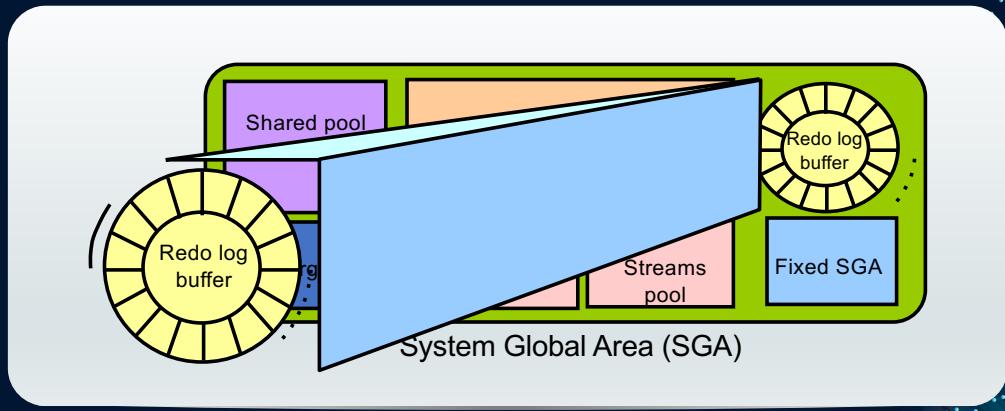
Database Buffer Cache

- Is part of the SGA
- Holds copies of data blocks that are read from data files
- Is shared by all concurrent users



Redo Log Buffer

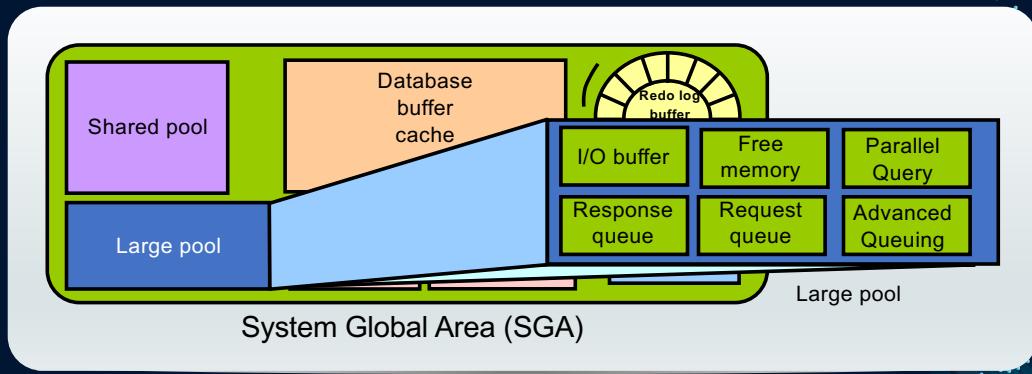
- Is a circular buffer in the SGA
- Holds information about changes made to the database
- Contains redo entries that have the information to redo changes made by operations such as DML and DDL



Large Pool

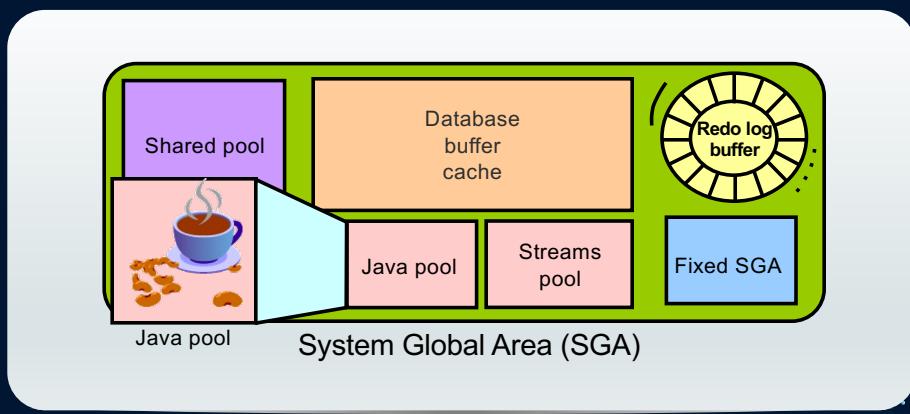
Provides large memory allocations for:

- Session memory for the shared server and the Oracle XA interface
- I/O server processes
- Oracle Database backup and restore operations



Java Pool

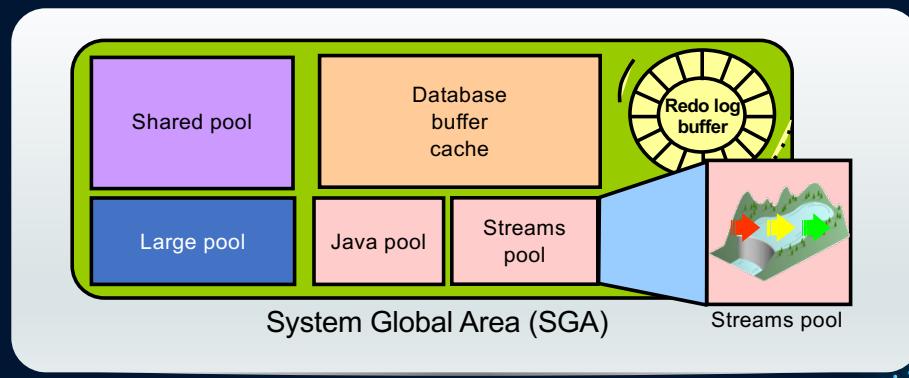
Java pool memory is used to store all session-specific Java code and data in the JVM.



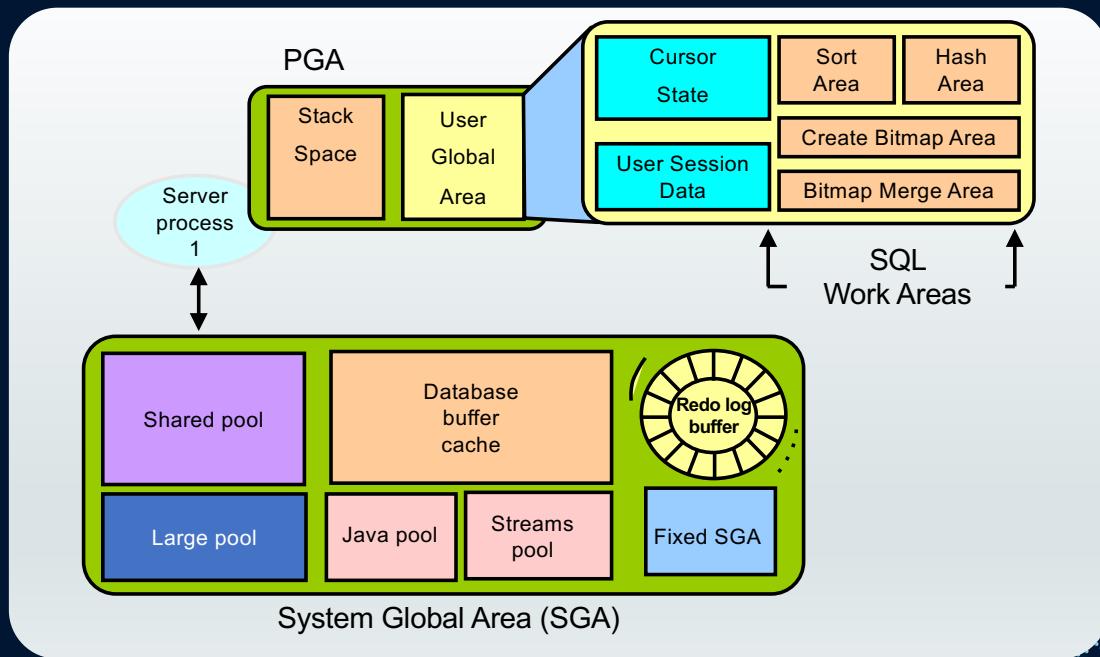
Streams Pool

Streams pool memory is used exclusively by Oracle Streams to:

- Store buffered queue messages
- Provide memory for Oracle Streams processes



Program Global Area (PGA)



Managing Memory Components

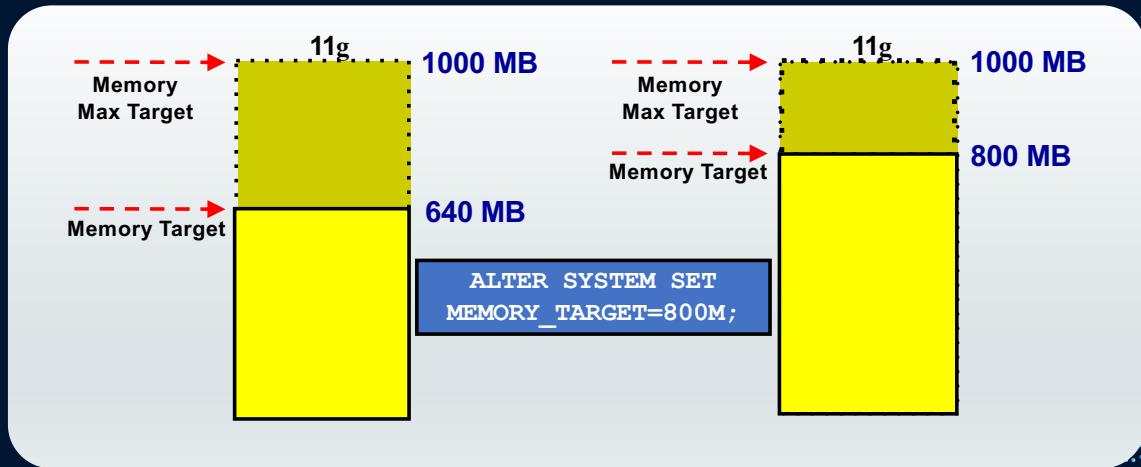
- Automatic Memory Management (AMM) enables you to specify total memory allocated to instance (including both SGA and PGA)
- Automatic Shared Memory Management (ASMM):
 - Enables you to specify total SGA memory through one initialization parameter
 - Enables the Oracle server to manage the amount of memory allocated to the shared pool, Java pool, buffer cache, streams pool, and large pool
- Manually setting shared memory management:
 - Sizes the components through multiple individual initialization parameters
 - Uses the appropriate Memory Advisor to make recommendations

Efficient Memory Usage: Guidelines

- Fit the SGA into physical memory.
- Use the Memory Advisors.
- Tune for the most efficient use of memory
 - Reduce overall physical I/O
 - Reduce the total memory needs

Automatic Memory Management

With Automatic Memory Management, the database server can size the SGA and PGA automatically according to your workload.



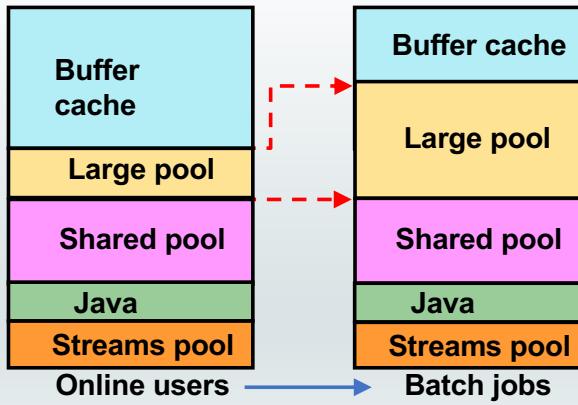
Monitoring Automatic Memory Management

View	Description
V\$MEMORY_DYNAMIC_COMPONENTS	Shows the current sizes of all dynamically tuned memory components
V\$MEMORY_RESIZE_OPS	Shows a circular history buffer of the last 800 memory resize requests
V\$MEMORY_TARGET_ADVICE	Provides tuning advice for the MEMORY_TARGET initialization parameter

Automatic Shared Memory Management

- Automatically adapts to workload changes
- Maximizes memory utilization
- Helps eliminate out-of-memory errors

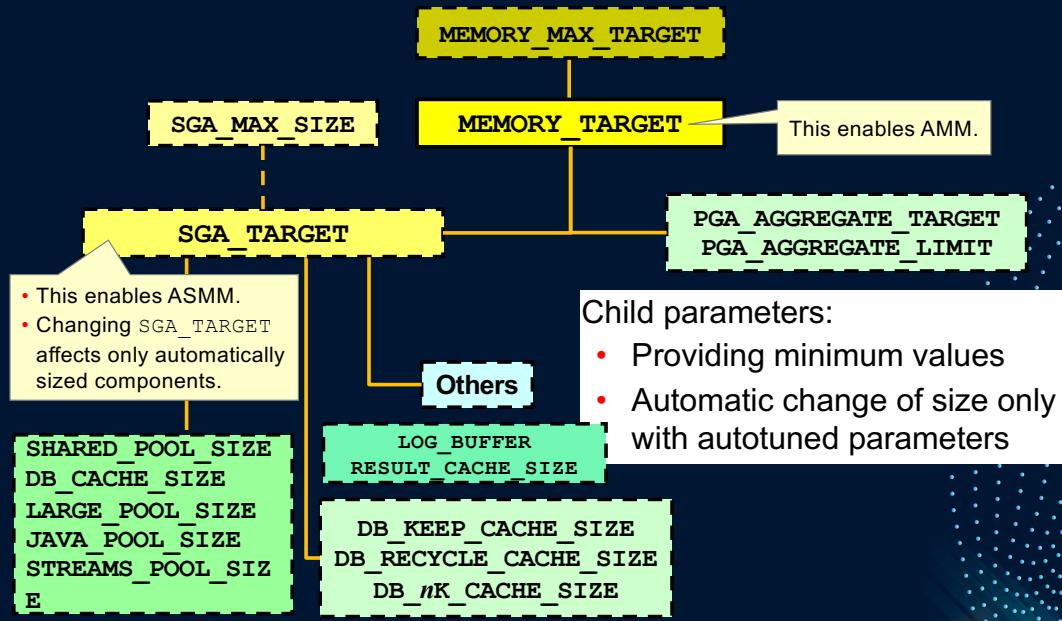
Example:



Understanding Automatic Shared Memory Management

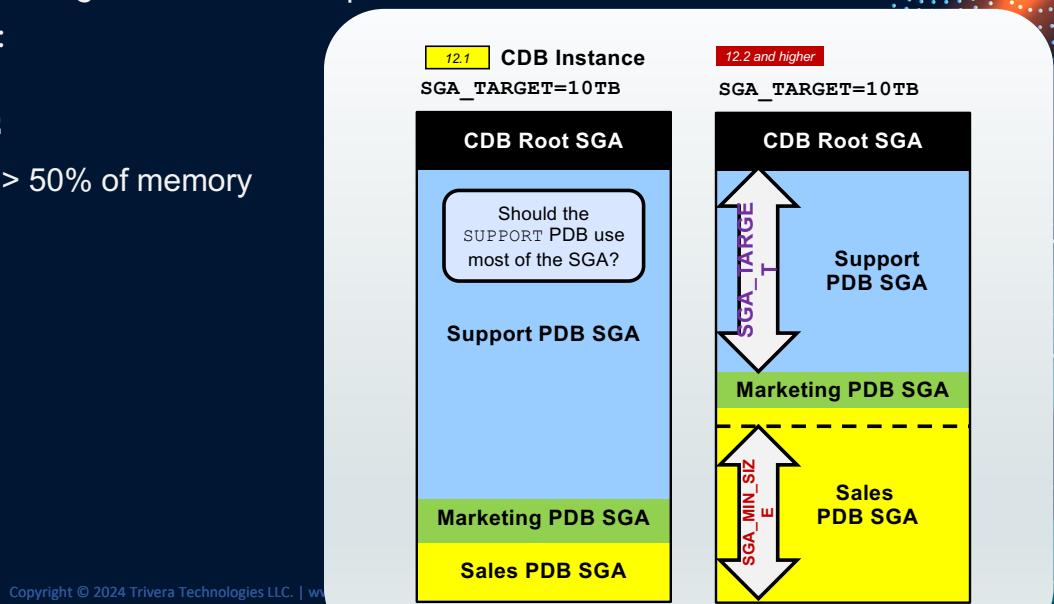
- ASMM is based on workload information that MMON captures in the background.
- MMON uses memory advisors.
- Memory is moved to where it is needed the most by MMAN.
- If an SPFILE is used (which is recommended):
 - Component sizes are saved across shutdowns
 - Saved values are used to bootstrap component sizes
 - There is no need to relearn optimal values

Oracle Database Memory Parameters



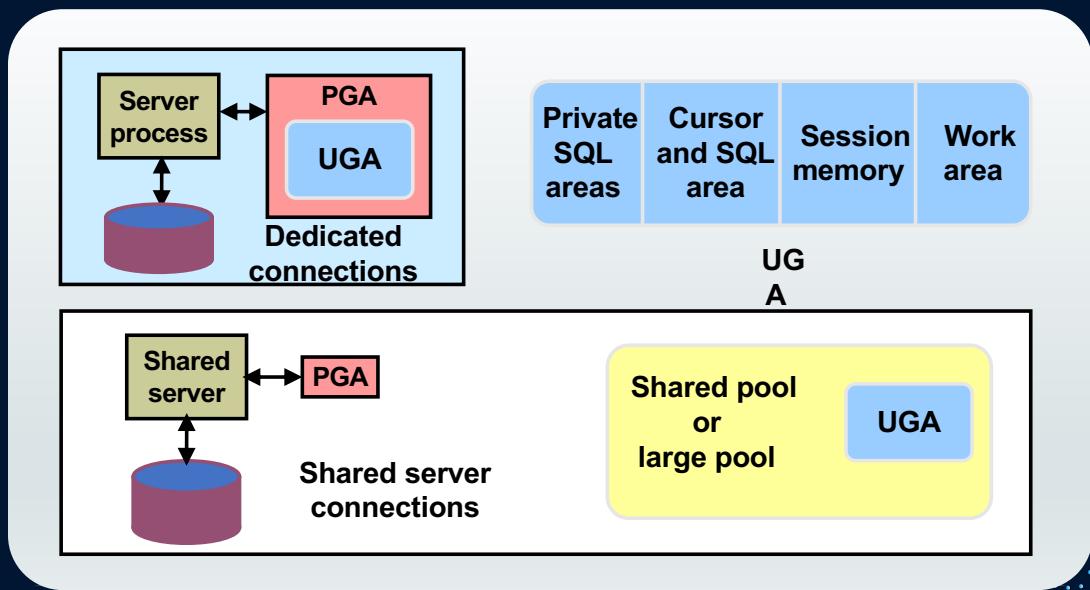
Managing the SGA for PDBs

- **SGA_TARGET** set at PDB level enforces a hard limit for the PDB's SGA.
- **SGA_TARGET** at PDB level provides more SGA for other containers.
- **SGA_MIN_SIZE** set for a PDB guarantees SGA space for the PDB.
- Parameters at PDB level:
 - **DB_CACHE_SIZE**
 - **SHARED_POOL_SIZE**
- PDB minimums cannot be > 50% of memory



Copyright © 2024 Trivera Technologies LLC. | www.triveratech.com

Managing the Program Global Area (PGA)



Automatic PGA memory management is enabled by default.

Managing the PGA for PDBs

Instance PGA_AGGREGATE_LIMIT

- No more PGA can be allocated.
- Calls or sessions of the largest PGA users are terminated.

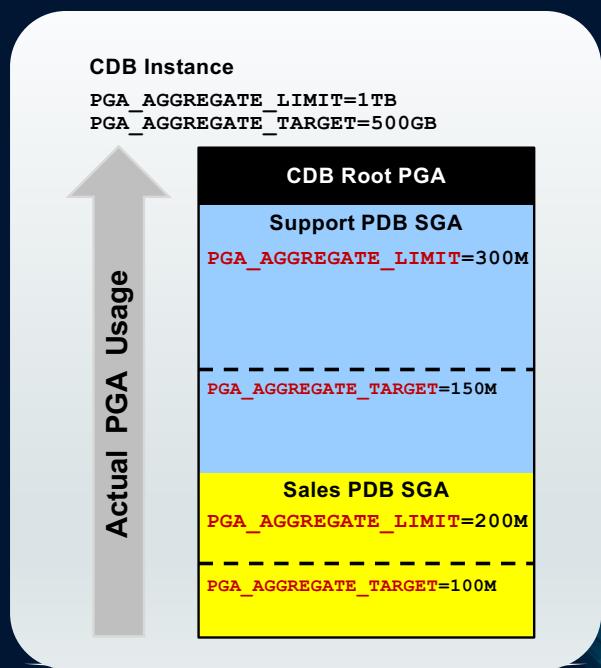
Instance PGA_AGGREGATE_TARGET

- All sessions must use TEMP rather than PGA.

PDB PGA_AGGREGATE_LIMIT

PDB PGA_AGGREGATE_TARGET

- These parameters set the same behavior at the PDB level.



Summary

In this lesson, you should have learned how to configure and monitor memory components for optimal performance.

Practice Overview

- Viewing Memory Configurations



Analyzing SQL and Optimizing Access Paths

Copyright © 2024 Trivera Technologies LLC. | www.triveratech.com

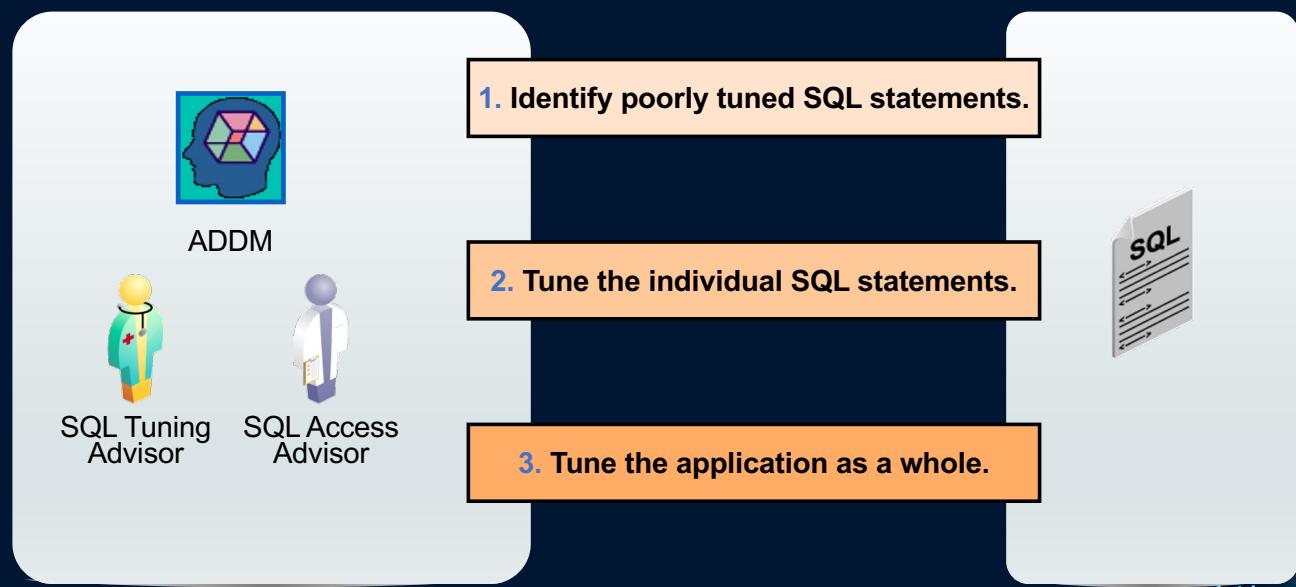
Experience is Everything

Objectives

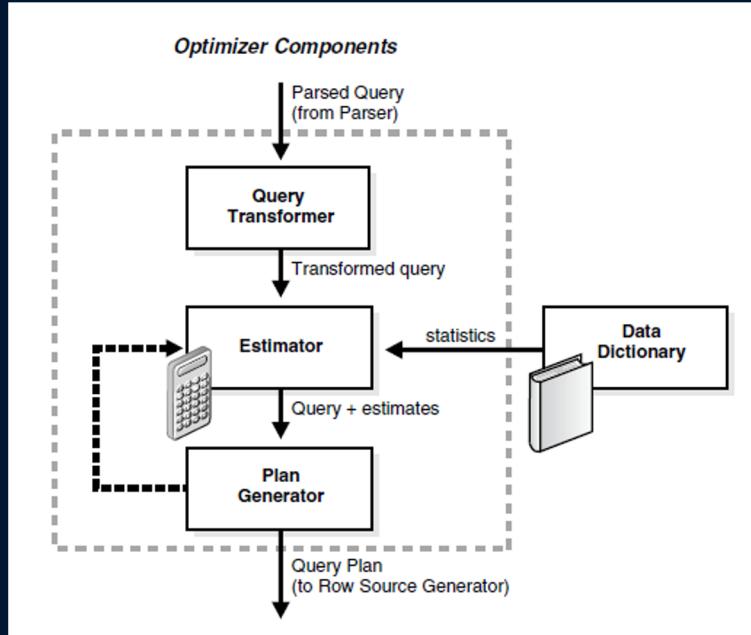
After completing this lesson, you should be able to:

- Describe the SQL tuning methodology
- Manage optimizer statistics
- Use SQL Tuning Advisor to identify and tune SQL statements that are using the most resources
- Use SQL Access Advisor to tune a workload

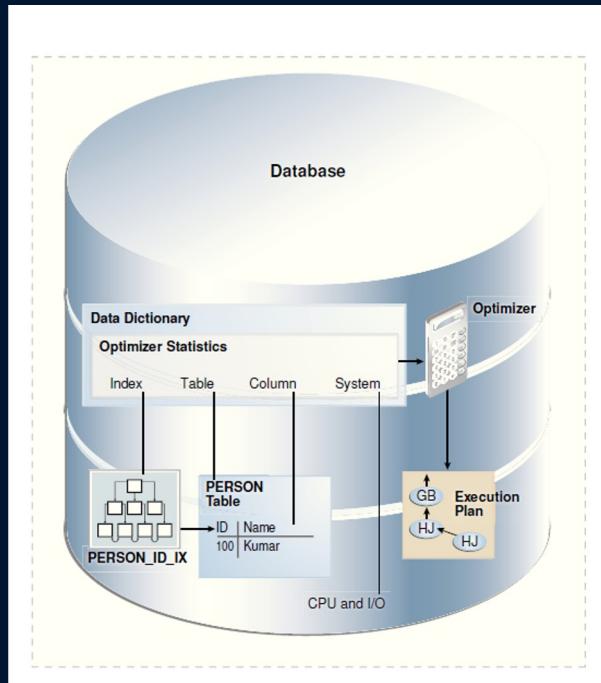
SQL Tuning Process



Oracle Optimizer



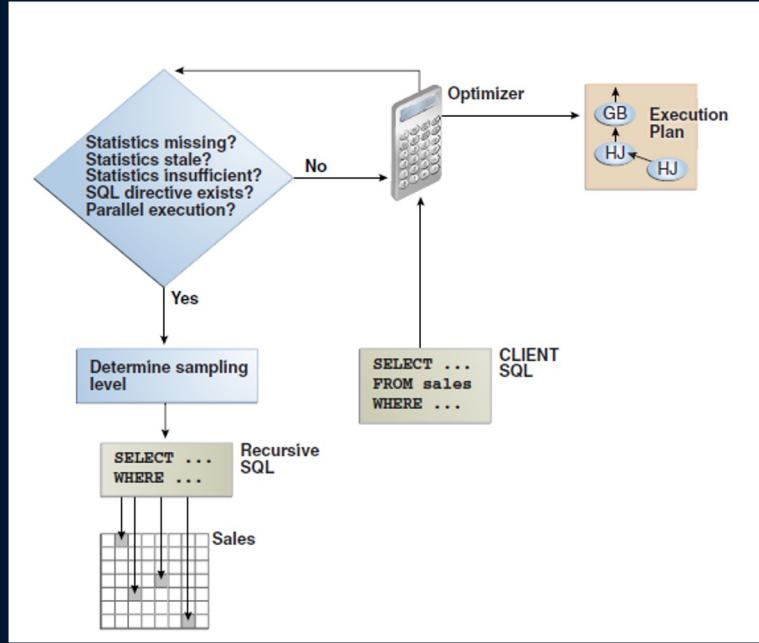
Optimizer Statistics



Copyright © 2024 Trivera Technologies LLC. | www.triveratech.com

289

Optimizer Statistics Collection



Setting Optimizer Statistics Preferences

DBMS_STATS.GATHER_*_STATS procedures: Gather statistics for an entire database or for individual objects using default values

Use the SET_*_PREFS procedures to create preference values for any object that is not owned by SYS or SYSTEM

Query DBA_TAB_STAT_PREFS to view object-level preferences

Execute the DBMS_STATS.GET_PREFS procedure for each preference to see the global preferences

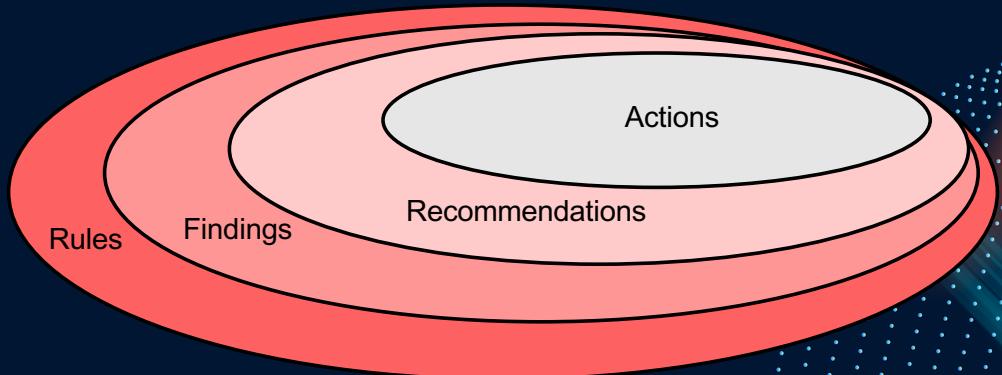
Optimizer Statistics Advisor

- If best practices change in a new release, Optimizer Statistics Advisor encodes these practices in its **rules**.
- The advisor always provides the most up-to-date recommendations.
- Track and analyze how statistics are collected.
 - Class of findings: System, Operations, Objects
- Scope of findings
 - Problems with gathering of statistics
 - Status of automatic statistic gathering jobs
 - Quality of current statistics
- Suggestion for changes to the statistics collection

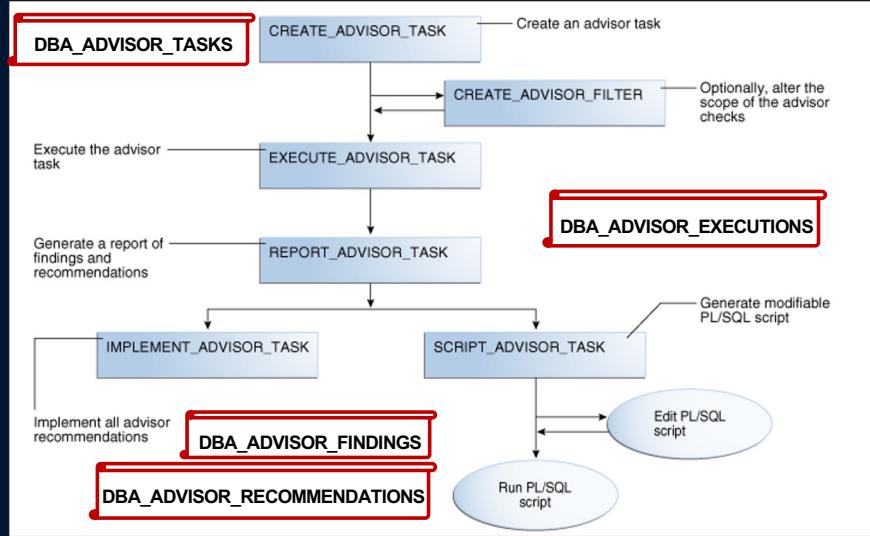
Optimizer Statistics Advisor Report

Report sections:

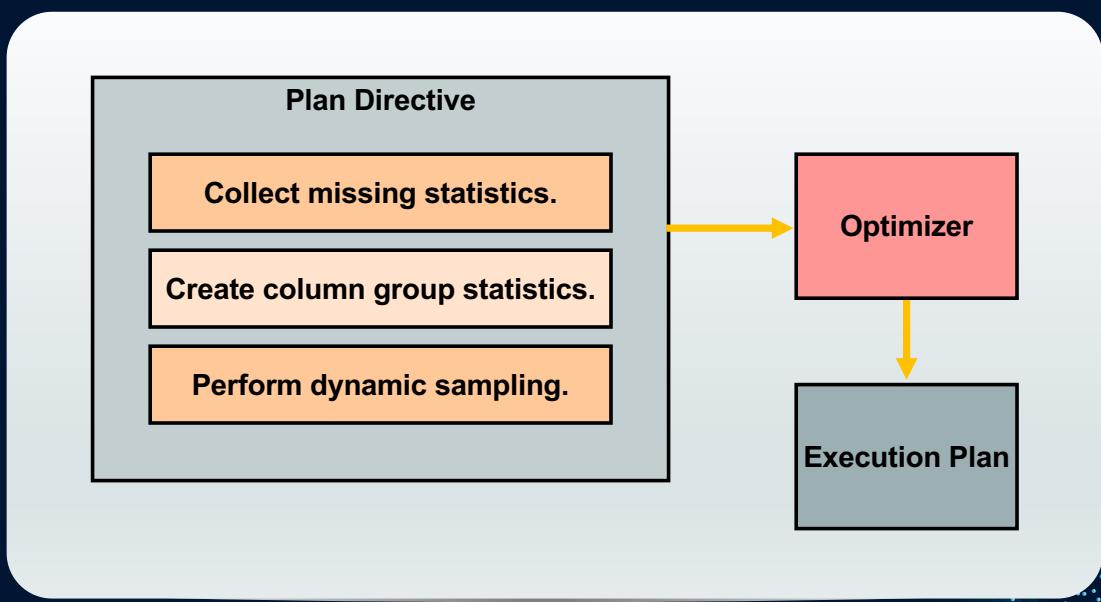
- Header
- Summary
- Errors
- Findings



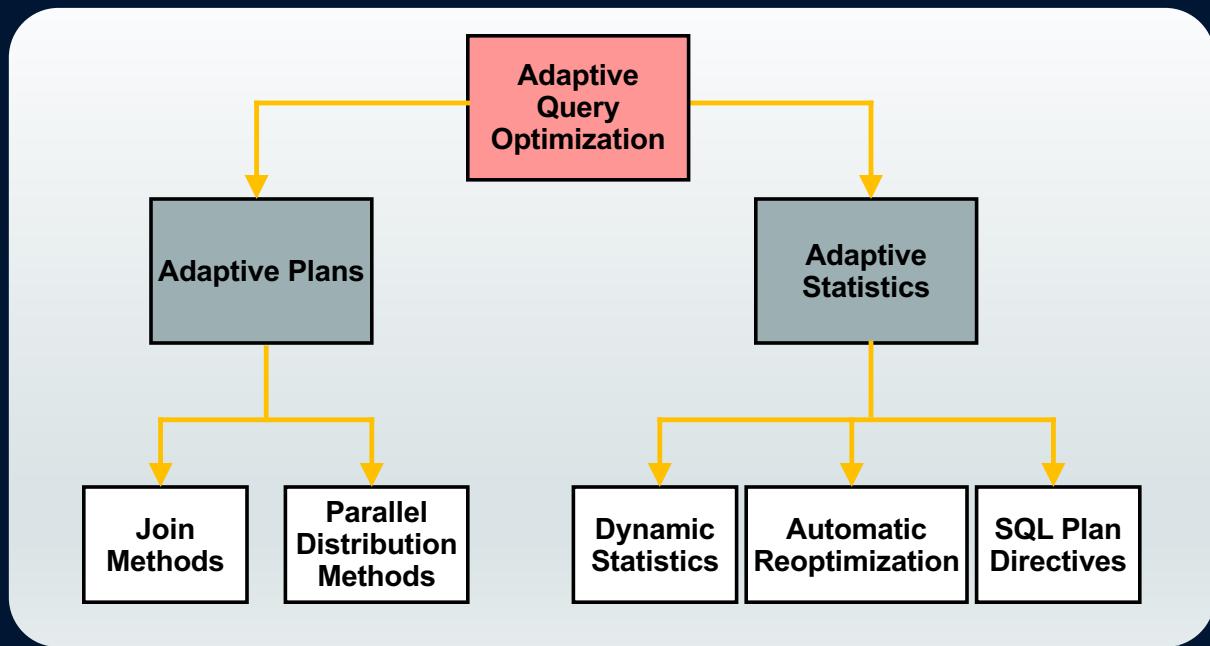
Executing Optimizer Statistics Advisor Tasks



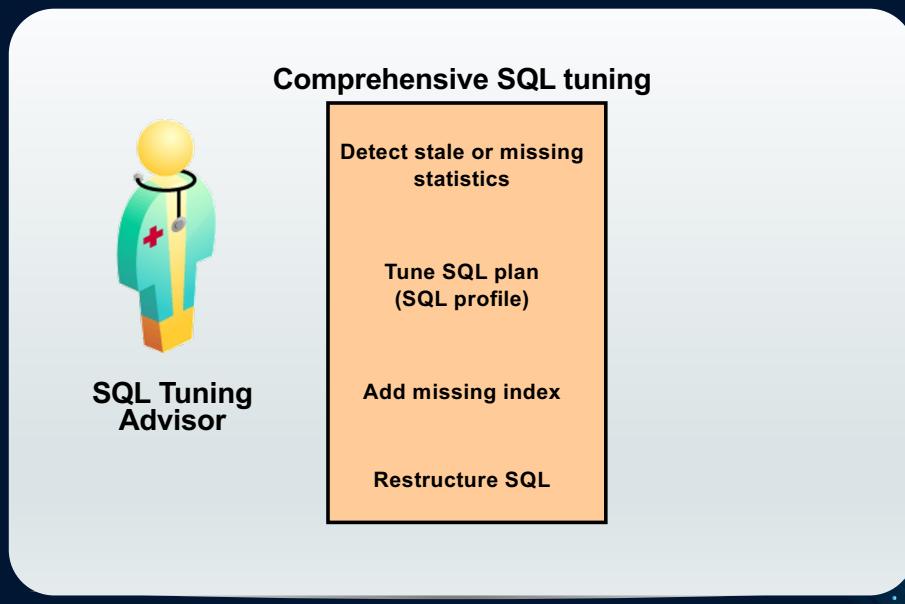
SQL Plan Directives



Adaptive Execution Plans



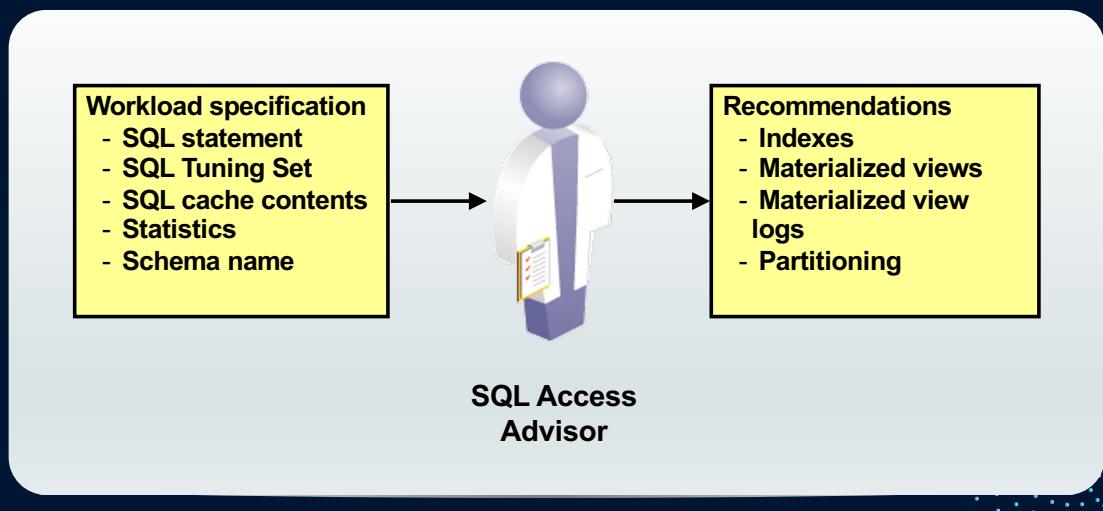
SQL Tuning Advisor: Overview



Copyright © 2024 Trivera Technologies LLC. | www.triveratech.com

300

SQL Access Advisor: Overview



SQL Performance Analyzer: Overview

Predicts the impact of system changes



**SQL
Performance
Analyzer**

Builds different
versions of SQL
workload performance

Executes SQL serially

Analyzes performance
differences

Offers fine-grained
performance analysis
on individual SQL

Managing Automated Tuning Tasks

- Use the DBMS_AUTO_TASK_ADMIN.ENABLE procedure to manage automatic space and performance tuning tasks
- Set the CLIENT_NAME parameter to the following values based on the task:
 - Automatic statistics collection: auto optimizer stats collection
 - Automatic SQL Tuning task: sql tuning advisor
 - Segment shrink: auto space advisor

```
BEGIN
    DBMS_AUTO_TASK_ADMIN.ENABLE (
        client_name => 'auto optimizer stats collection',
        operation    => NULL, window_name => NULL);
END;
```

Summary

In this lesson, you should have learned how to:

- Describe the SQL tuning methodology
- Manage optimizer statistics
- Use SQL Tuning Advisor to identify and tune SQL statements that are using the most resources
- Use SQL Access Advisor to tune a workload

Practice Overview

- Using the SQL Tuning Advisor
- Using the Optimizer Statistics Advisor



Backup and Recovery: Overview

Copyright © 2024 Trivera Technologies LLC. | www.triveratech.com

Experience is Everything

Objectives

After completing this lesson, you should be able to:

- Identify the responsibilities of a DBA in database backup and recovery
- Identify the types of failure that can occur in an Oracle database
- Describe instance recovery
- Describe complete and incomplete recovery

DBA Responsibilities

- Protect the database from failure wherever possible.
- Increase the mean time between failures (MTBF).
- Protect critical components by using redundancy.
- Decrease the mean time to recover (MTTR).
- Minimize the loss of data.

Separation of DBA Duties

The SYSBACKUP administrative privilege:

- Includes permissions for backup and recovery (connecting to a closed database)
- Does not include data access privileges such as SELECT ANY TABLE
- Is granted to the SYSBACKUP user that is created during database installation
- Can be explicitly used in RMAN connections by a SYSBACKUP privileged user

```
$ rman target ''/ as sysbackup"  
connected to target database: ORCL (DBID=1297344416)
```

Assessing Your Recovery Requirements

- Identify and prioritize critical data.
- Base recovery requirements on data criticality.
 - Recovery Point Objective (RPO): Tolerance for data loss
 - How frequently should backups be taken?
 - Is point-in-time recovery required?
 - Recovery Time Objective (RTO): Tolerance for down time
 - Downtime: Problem identification + recovery planning + systems recovery
 - Tiered RTO per level of granularity (database, tablespace, table, row)
 - Determine backup retention policy for on-site, off-site, and long-term backups.
- Assess data protection requirements.
 - Physical: Disasters, outages, failures, corruptions
 - Logical: Human errors, application errors

Categories of Failure

Failures can generally be divided into the following categories:

- Statement failure
- User process failure
- Network failure
- User error
- Instance failure
- Media failure



Statement Failure

Typical Problems	Possible Solutions
Attempts to enter invalid data into a table	Work with users to validate and correct data.
Attempts to perform operations with insufficient privileges	Provide the appropriate object or system privileges.
Attempts to allocate space that fails	Enable resumable space allocation. Increase owner quota. Add space to the tablespace.
Logic errors in applications	Work with developers to correct program errors.

User Process Failure

Typical Problems	Possible Solutions
A user performs an abnormal disconnect.	A DBA's action is not usually needed to resolve user process failures.
A user's session is abnormally terminated.	Instance background processes roll back uncommitted changes and release locks.
A user experiences a program error that terminates the session.	The DBA should watch for trends.

Network Failure

Typical Problems	Possible Solutions
Listener fails	Configure a backup listener and connect-time failover.
Network interface card (NIC) fails	Configure multiple network cards.
Network connection fails	Configure a backup network connection.

User Error

Typical Problems	Possible Solutions
User inadvertently deletes or modifies data	Roll back a transaction and dependent transactions or rewind the table
User drops a table	Recover the table from recycle bin Recover the table from a backup

Use Oracle LogMiner to query your online redo logs and archived redo logs through an Enterprise Manager or SQL interface.

Instance Failure

Typical Causes	Possible Solutions
Power outage	Restart the instance by using the STARTUP command. Recovering from instance failure is automatic, including rolling forward changes in the redo logs and then rolling back any uncommitted transactions.
Hardware failure	Investigate the causes of failure by using the alert log, trace files, and Enterprise Manager.
Failure of one of the critical background processes	
Emergency shutdown procedures	

Media Failure

Typical Causes	Possible Solution
Failure of a disk drive	1. Restore the affected file from backup.
Failure of a disk controller	2. Inform the database server of a new file location (if necessary).
Deletion or corruption of a file needed for a database operation	3. Recover the file by applying redo information (if necessary).
Storage network failure	
Solid state storage corruption	

Data Failures

- **Inaccessible components:** Missing data files at the OS level, incorrect access permissions, offline tablespace
- **Physical corruptions:** Block checksum failures, invalid block header field values
- **Logical corruptions:** Inconsistent dictionary, corrupt row piece, index entry, or transaction
- **Inconsistencies:** Control file older or newer than the data files and online redo logs
- **I/O failures:** Limit on the number of open files exceeded, inaccessible channels, network or I/O error



Instance Recovery

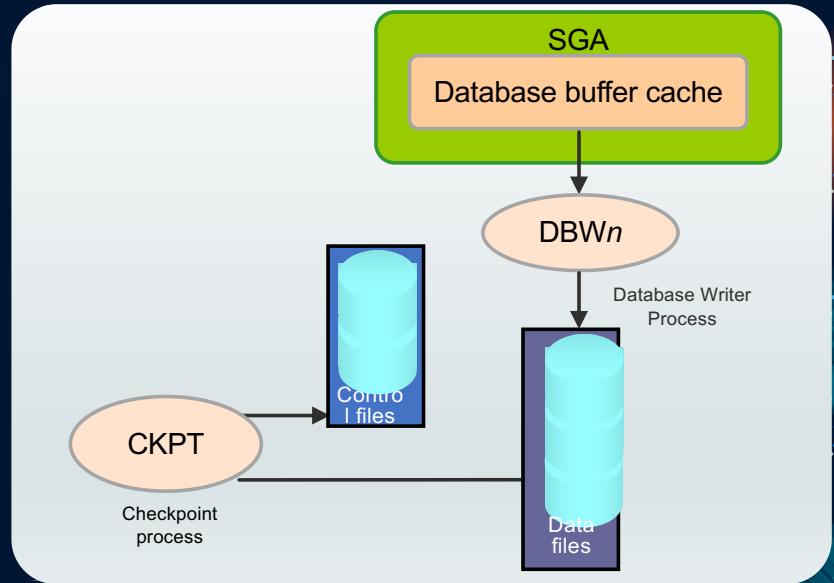
You can understand instance recovery by becoming familiar with the following concepts and procedures:

- The checkpoint (CKPT) process
- Redo log files and the Log Writer (LGWR) process
- Automatic instance or crash recovery
- Phases of instance recovery
- Tuning instance recovery
- Using the MTTR Advisor

The Checkpoint (CKPT) Process

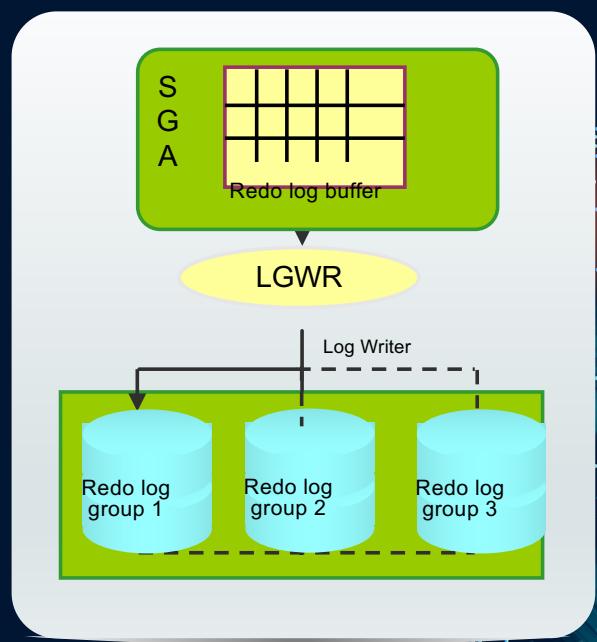
CKPT is responsible for:

- Updating data file headers with checkpoint information
- Updating control files with checkpoint information
- Signaling DBW n at full checkpoints

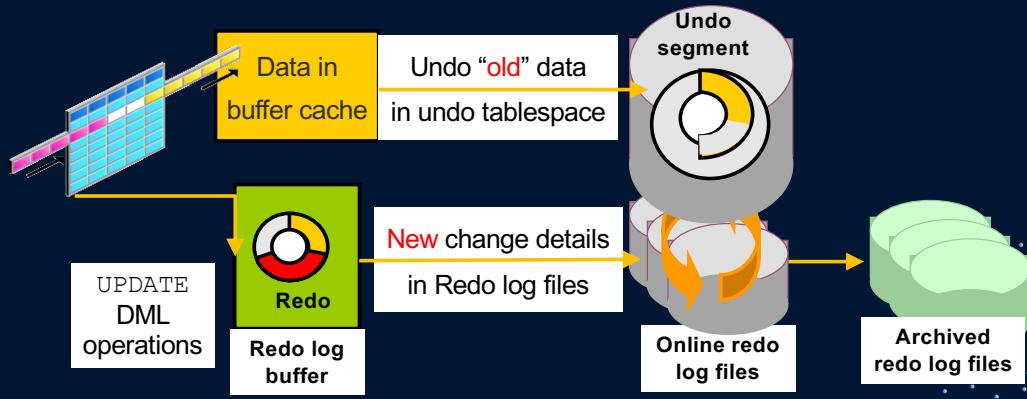


Redo Log Files and the Log Writer (LGWR) Process

- Redo log files:
 - Record changes to the database
 - Should be multiplexed to protect against loss
- Log Writer (LGWR) writes:
 - At commit
 - When the redo log buffer is one-third full
 - Every three seconds
 - Before DBWn writes
 - Before clean shutdowns



Database Log Mode



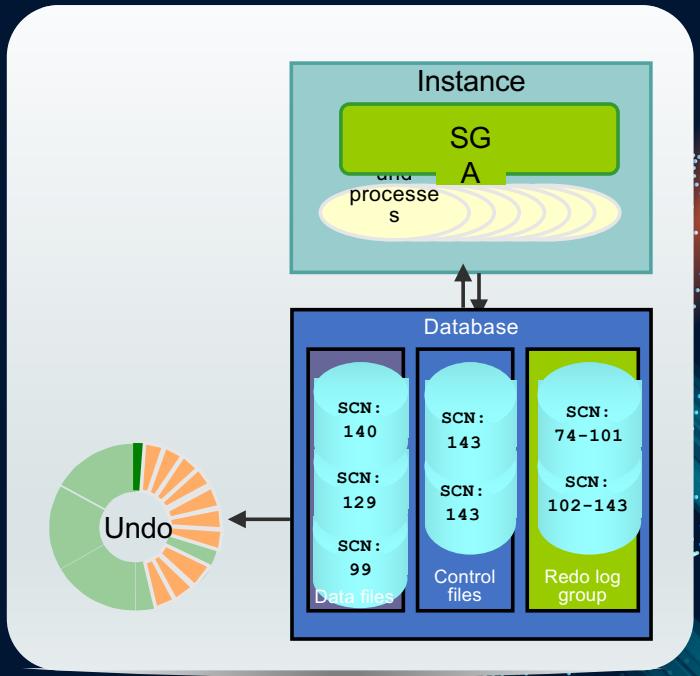
NOARCHIVELOG mode	ARCHIVELOG mode
Closed database	Open database
Recovery to the last backup	Recovery to last committed transaction
Suitable for training and test environments, for data warehouses with infrequent loads	Suitable for production environments

Automatic Instance Recovery or Crash Recovery

- Is caused by attempts to open a database whose files are not synchronized on shutdown
- Uses information stored in redo log groups to synchronize files
- Involves two distinct operations:
 - **Rolling forward:** Redo log changes (both committed and uncommitted) are applied to data files.
 - **Rolling back:** Changes that are made but not committed are returned to their original state.

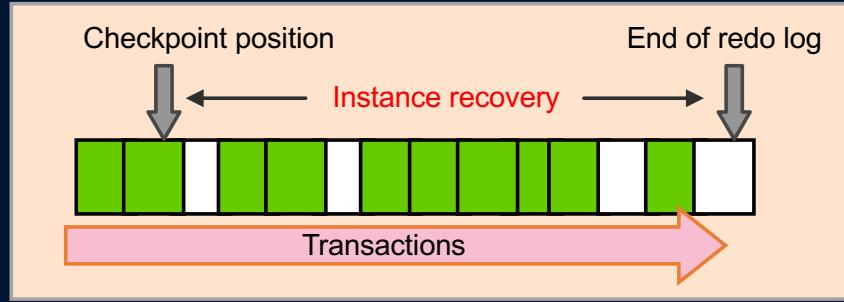
Phases of Instance Recovery

1. Instance startup (data files are out of sync)
2. Roll forward (redo)
3. Committed and uncommitted data in files
4. Database opened
5. Roll back (undo)
6. Committed data in files



Tuning Instance Recovery

- During instance recovery, the transactions between the checkpoint position and the end of the redo log must be applied to data files.
- You tune instance recovery by controlling the difference between the checkpoint position and the end of the redo log.



Using the MTTR Advisor

- Specify the desired time in seconds or minutes.
- The default value is 0 (disabled).
- The maximum value is 3,600 seconds (one hour).

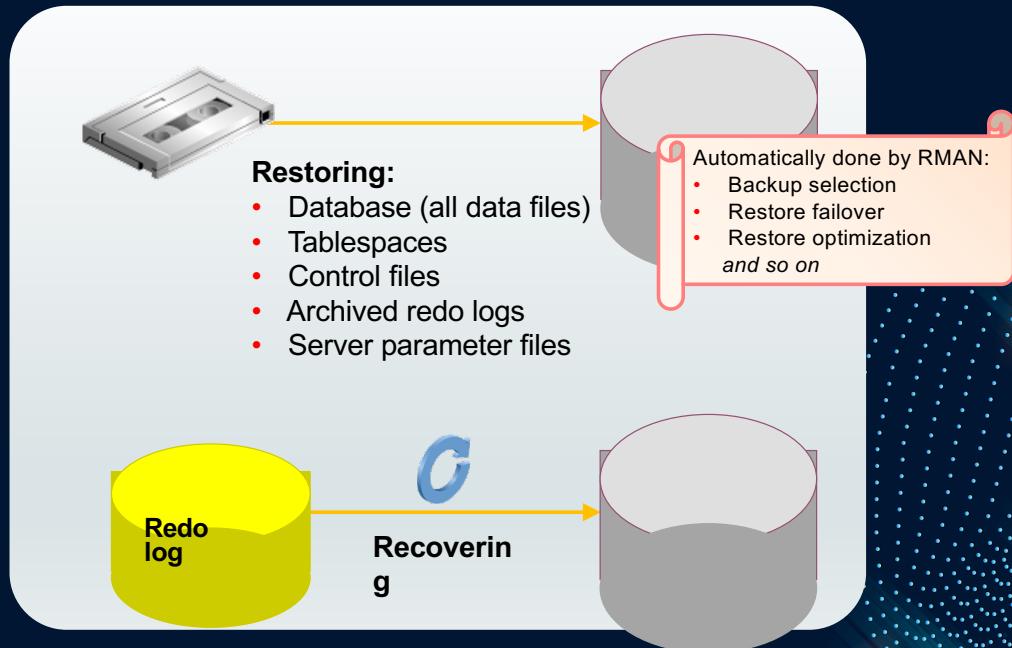
The screenshot shows the 'Recovery Settings' page with the following details:

- Logged in as DBA1**
- Recovery Settings**
- Show SQL | Revert | Apply**
- Instance Recovery**
- The fast-start checkpointing feature is enabled by specifying a non-zero desired mean-time to recover (MTTR) value, which will be used to set the FAST_START_MTTR_TARGET initialization parameter. This parameter controls the amount of time the database takes to perform crash recovery for a single instance. When fast-start checkpointing is enabled, Oracle automatically maintains the speed of checkpointing so that the requested MTTR is achieved. Setting the value to 0 will disable this functionality.
- Current Estimated Mean Time To Recover (seconds) 19
- Desired Mean Time To Recover Minutes

Copyright © 2024 Trivera Technologies LLC. | www.triveratech.com

329

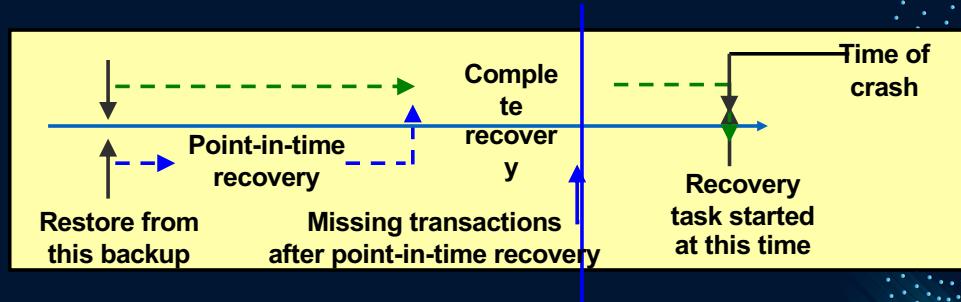
Restoring and Recovering



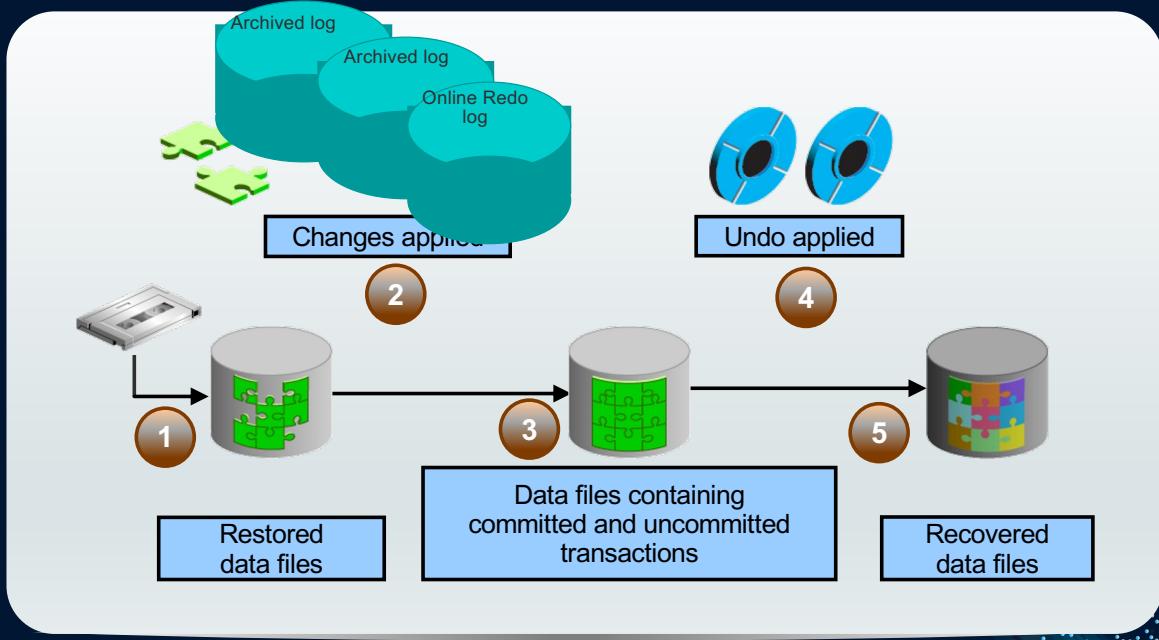
Comparing Complete and Incomplete Recovery

Recovery can have two kinds of scope:

- **Complete recovery:** Brings the database or tablespace up to the present, including all committed data changes made to the point in time when the recovery was requested
- **Incomplete or point-in-time recovery (PITR):** Brings the database or tablespace up to a specified point in time in the past, before the recovery operation was requested



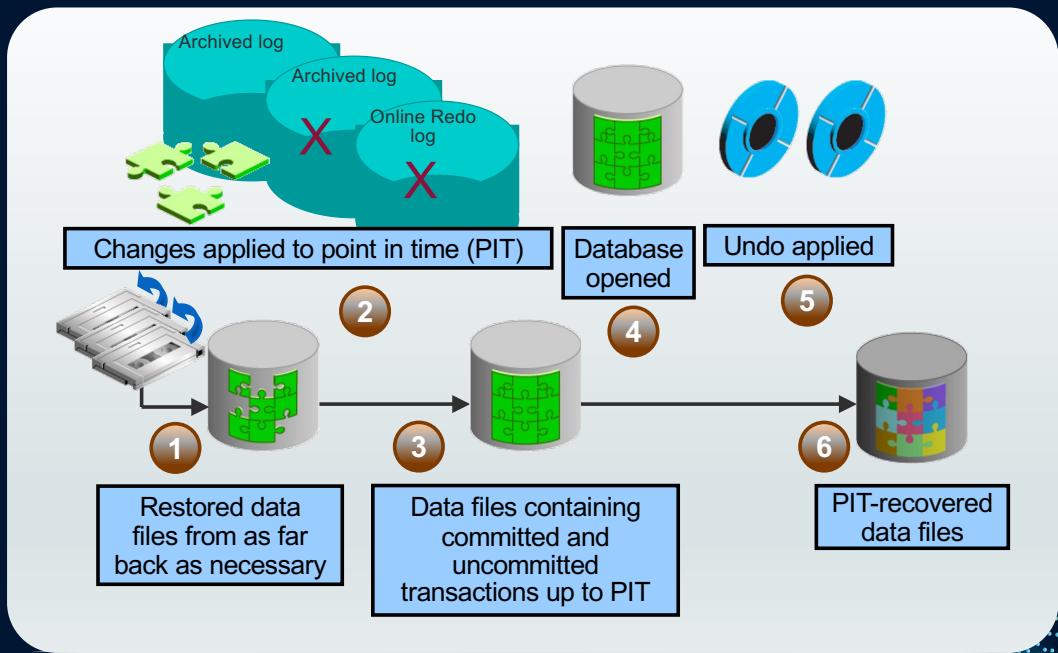
The Complete Recovery Process



Copyright © 2024 Trivera Technologies LLC. | www.triveratech.com

332

The Point-in-Time Recovery Process



Copyright © 2024 Trivera Technologies LLC. | www.triveratech.com

333

Oracle Data Protection Solutions

Backup and Recovery Objective	Recovery Time Objective (RTO)	Oracle Solution
Physical data protection	Hours/Days	Recovery Manager Oracle Secure Backup
Logical data protection	Minutes/Hours	Flashback Technologies
Recovery analysis	Minimize time for problem identification and recovery planning	Data Recovery Advisor

Disaster Recovery Objective	Recovery Time Objective (RTO)	Oracle Solution
Physical data protection	Seconds/Minutes	Data Guard Active Data Guard

Flashback Technology

Use Flashback technology for:

- Viewing past states of data
- Winding data back and forth in time
- Assisting users in error analysis and recovery



For error analysis:

Oracle Flashback Query

Oracle Flashback Versions Query

Oracle Flashback Transaction Query

For error recovery:

Oracle Flashback Transaction Backout

Oracle Flashback Table

Oracle Flashback Drop

Oracle Flashback Database

Summary

In this lesson, you should have learned how to:

- Identify the responsibilities of a DBA in database backup and recovery
- Identify the types of failure that can occur in an Oracle database
- Describe instance recovery
- Describe complete and incomplete recovery



Backup and Recovery Configuration

Copyright © 2024 Trivera Technologies LLC. | www.triveratech.com

Experience is Everything

Objectives

After completing this lesson, you should be able to:

- Configure the fast recovery area
- Multiplex the control file
- Multiplex redo log files
- Configure ARCHIVELOG mode

Configuring for Recoverability

Configure your database for maximum recoverability by:

- Scheduling regular backups
- Multiplexing control files
- Multiplexing redo log groups
- Retaining archived copies of redo logs

Configuring the Fast Recovery Area

- Fast recovery area:
 - Strongly recommended for simplified backup storage management
 - Storage space (separate from working database files)
 - Location specified by the `DB_RECOVERY_FILE_DEST` parameter
 - Size specified by the `DB_RECOVERY_FILE_DEST_SIZE` parameter
 - Large enough for backups, archived logs, flashback logs, multiplexed control files, and multiplexed redo logs
 - Automatically managed according to your retention policy
- Configuration of the fast recovery area includes specifying the location, size, and retention policy.

```
ALTER SYSTEM SET db_recovery_file_dest = directory | disk group
ALTER SYSTEM SET db_recovery_file_destsize = integer [K | M | G]
```

Monitoring the Fast Recovery Area

- Monitor the fast recovery area to ensure that it does not reach its capacity.
- The instance will pause if there isn't enough space in the fast recovery area to create an archived log.
- Query the `V$RECOVERY_FILE_DEST` view to determine the current location, disk quota, space in use, space reclaimable by deleting files, and total number of files in the fast recovery area.
- Query the `V$RECOVERY_AREA_USAGE` view to determine the percentage of the total disk quota used by different types of files.
- You can also use GUI tools such as Enterprise Manager Cloud Control to monitor the space usage.

Multiplexing Control Files

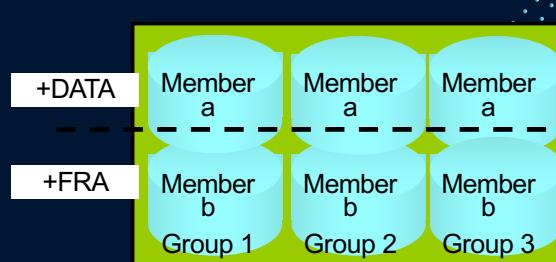
To protect against database failure, your database should have multiple copies of the control file.

	ASM Storage	File System Storage
Best Practice	One copy on each disk group (such as +DATA and +FRA)	At least two copies, each on a separate disk (at least one on a separate disk controller)
Steps to create additional control files	No additional control file copies required	<ol style="list-style-type: none">1. Alter the SPFILE with the ALTER SYSTEM SET control_files command.2. Shut down the database.3. Copy the control file to a new location.4. Open the database and verify the addition of the new control file.

Redo Log Files

Multiplex redo log groups to protect against media failure and loss of data. This increases database I/O. It is suggested that redo log groups have:

- At least two members (files) per group
- Each member:
 - On a separate disk or controller if using file system storage
 - In a separate disk group (such as +DATA and +FRA) if using ASM



Note: Multiplexing redo logs may impact overall database performance.

Multiplexing the Redo Log

Add a member to an existing log group:

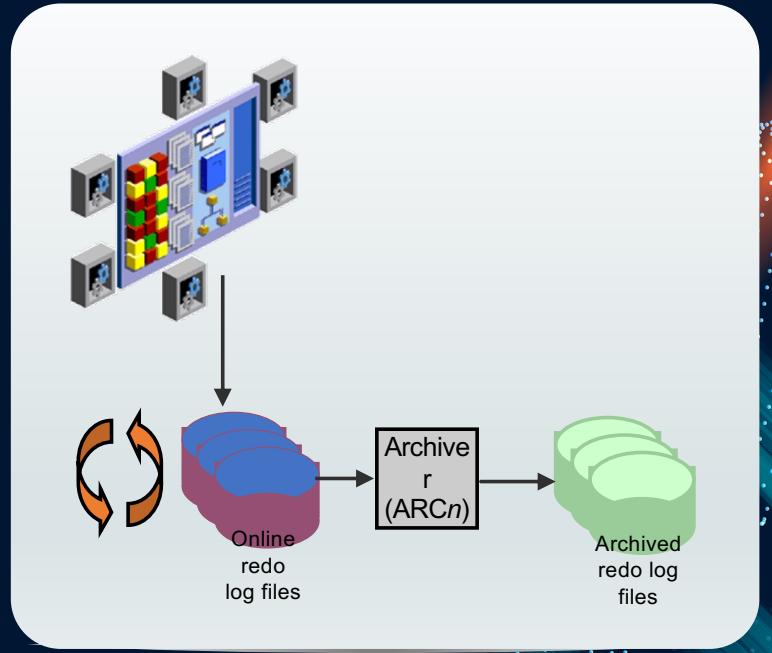
- Navigate to the Redo Log Groups page in Enterprise Manager Database Express.
- Use the `ALTER DATABASE` command:

```
SQL> ALTER DATABASE
  2  ADD LOGFILE MEMBER '/u01/app/oracle/oradata/orcl/redola.log'
  3  TO GROUP 1;
```

Creating Archived Redo Log Files

To preserve redo information, create archived copies of redo log files by performing the following steps:

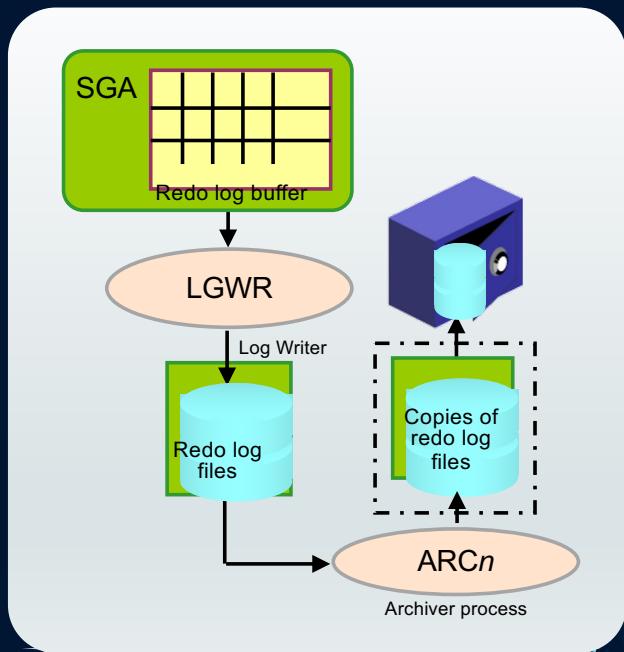
1. Specify the archived redo log file-naming convention.
2. Specify one or more archived redo log file locations.
3. Place the database in ARCHIVELOG mode.



Archiver (ARCn) Process

Archiver (ARCn):

- Automatically archives online redo log files when the database is in ARCHIVELOG mode
- Preserves a record of all changes made to the database



Archived Redo Log Files: Naming and Destinations

- Use the `LOG_ARCHIVE_DEST` initialization parameter to specify a single destination.
- Use the `LOG_ARCHIVE_DEST_n` initialization parameters to archive to two or more locations.
- If you are using file system storage, it is recommended that you add multiple locations across different disks.
- If the fast recovery area is enabled, `USE_DB_RECOVERY_FILE_DEST` is specified by default as an archived redo log file destination.

Configuring ARCHIVELOG Mode

- You can use SQL commands as follows:
 1. Shut down the database instance if it is open.
 2. Mount the database.
 3. Issue the `ALTER DATABASE ARCHIVELOG` command.
 4. Open the database.

```
SQL> shutdown immediate
SQL> startup mount
SQL> alter database archivelog;
SQL> alter database open;
```

- You can also use Enterprise Manager Cloud Control to place the database in ARCHIVELOG mode.

Summary

In this lesson, you should have learned how to:

- Configure the Fast Recovery Area
- Multiplex the control file
- Multiplex redo log files
- Configure ARCHIVELOG mode

Practice Overview

- Verifying that the Control File is Multiplexed
- Configuring the Size of the Fast Recovery Area
- Verifying that the Redo Log File Is Multiplexed
- Configuring ARCHIVELOG Mode



Using Recovery Manager (RMAN)

Copyright © 2024 Trivera Technologies LLC. | www.triveratech.com

Experience is Everything

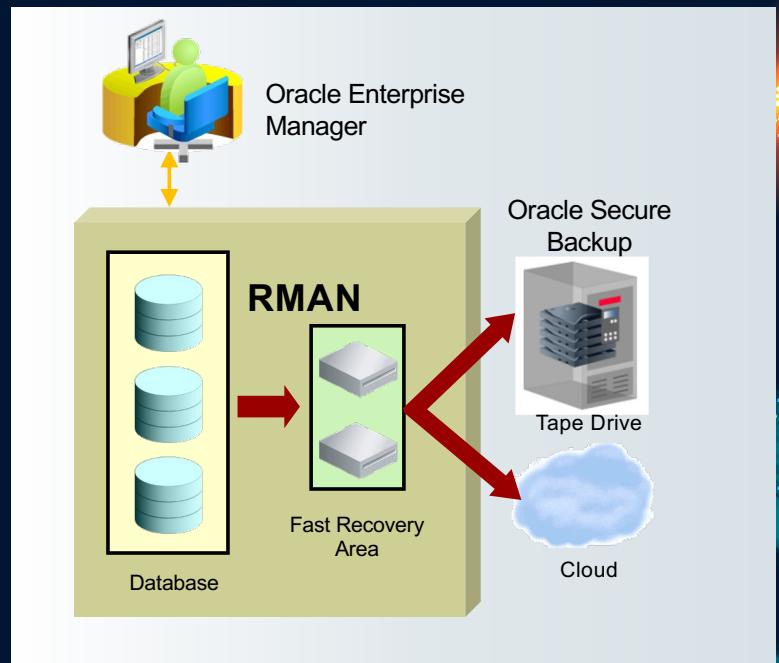
Objectives

After completing this lesson, you should be able to:

- Describe the features and functions of Recovery Manager (RMAN)
- Configure and manage RMAN settings

Integrated Oracle Recovery Manager (RMAN)

- Intrinsic knowledge of database file formats and recovery procedures
 - Tablespace and data file recovery
 - Block validation
 - Online block-level recovery
 - Online, multistreamed backup
 - Unused block compression
 - Native encryption
- Integrated disk, tape, and cloud backup leveraging the fast recovery area and Oracle Secure Backup



Connecting to RMAN and a Target Database

```
. oraenv
orcl
$ rman target ''/ as sysbackup''

RMAN> BACKUP DATABASE;
Starting backup at 10-OCT-18
.
.

RMAN> LIST BACKUP;
BS Key  Type LV Size    Device Type Elapsed Time Completion Time
-----  --  --  --  -----
1       Full   1.06G  DISK            00:01:49   10-OCT-18
.

.

RMAN> DELETE OBSOLETE;
.

Do you really want to delete the above objects (enter YES or NO)? YES
deleted archived log
.

.
```

Using SQL in RMAN

From the RMAN command line:

- Execute SQL commands and PL/SQL procedures
- Use the optional `SQL` prefix to avoid ambiguity

```
RMAN> SELECT NAME, DBID, LOG_MODE  
2> FROM V$DATABASE;  
  
NAME          DBID LOG_MODE  
-----  
ORCL        1297344416 NOARCHIVELOG
```

Types of RMAN Commands

RMAN commands are of the following types:

- Stand-alone command:
 - Is executed individually at the RMAN prompt
 - Cannot appear as subcommands within `RUN`
- Job command:
 - Must be within the braces of a `RUN` command
 - Is executed as a group

Some commands can be executed as both types.

Job Commands: Example

Job commands appear inside a `RUN` command block:

```
RMAN> RUN
2> {
3>   ALLOCATE CHANNEL c1 DEVICE TYPE DISK
4>   FORMAT "/disk2/%U";
5>   BACKUP AS BACKUPSET DATABASE;
6>   SQL 'alter system archive log current';
7> }
```

Execution of the entire block starts when this line is entered.

Deallocated after the RUN block completes

Configuring Persistent Settings for RMAN

- RMAN is preset with default configuration settings.
- Use the `CONFIGURE` command to:
 - Configure automatic channels
 - Specify the backup retention policy
 - Specify the number of backup copies to be created
 - Set the default backup type to `BACKUPSET` or `COPY`
 - Limit the size of backup pieces
 - Exempt a tablespace from backup
 - Enable and disable backup optimization
 - Configure automatic backups of control files
 - Define the archive log deletion policy
 - Specify the parallelism for a device
 - Set the encryption and compression parameters to be used for backups

Viewing Persistent Settings

To examine the persistent RMAN settings for a database:

- Use the RMAN SHOW ALL command to view all configuration settings
- Query the V\$RMAN_CONFIGURATION view to display configuration settings that have been explicitly set

Managing Persistent Settings

- Use multiple streams of data to and from a device:

```
RMAN> CONFIGURE DEVICE TYPE sbt PARALLELISM 3;
```

- Use the SHOW command to list current settings:

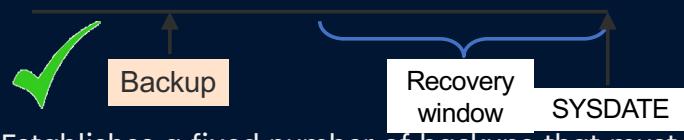
```
RMAN> SHOW CONTROLFILE AUTOBACKUP FORMAT;
RMAN> SHOW EXCLUDE;
RMAN> SHOW ALL;
```

- Use the CLEAR option of the CONFIGURE command to reset any persistent setting to its default value:

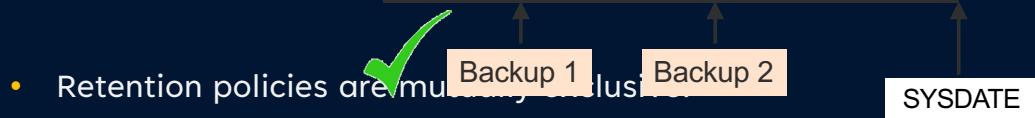
```
RMAN> CONFIGURE BACKUP OPTIMIZATION CLEAR;
RMAN> CONFIGURE MAXSETSIZE CLEAR;
RMAN> CONFIGURE DEFAULT DEVICE TYPE CLEAR;
```

Specifying a Retention Policy

- A retention policy describes which backups will be kept and for how long.
- There are two types of retention policies:
 - **Recovery window:** Establishes a period of time within which point-in-time recovery must be possible



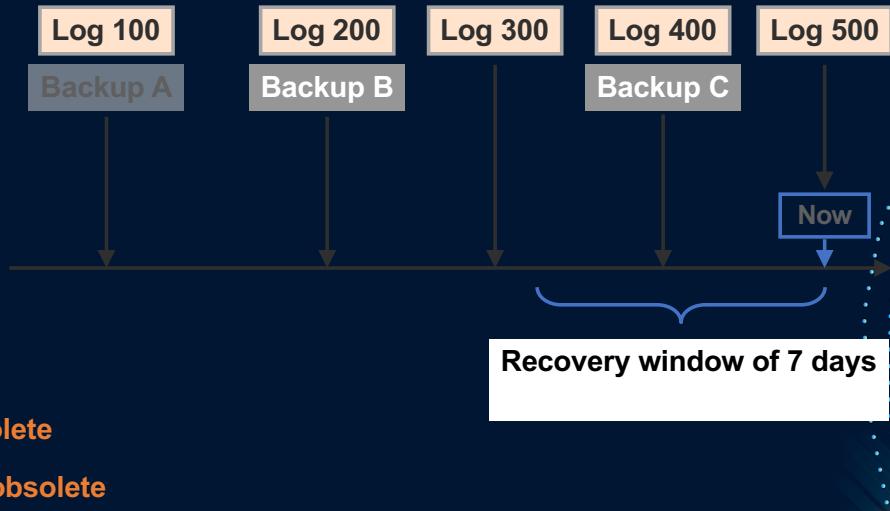
- **Redundancy:** Establishes a fixed number of backups that must be kept



- Retention policies are cumulative, plus

Recovery Window Retention Policy: Example

Backup B and archive logs 201 through 500 are required to satisfy this retention policy.



Summary

In this lesson, you should have learned how to:

- Describe the features and functions of Recovery Manager (RMAN)
- Configure and manage RMAN settings

Practice Overview

- Configuring the Default Backup Destination
- Setting the Date and Time Format for RMAN
- Configuring RMAN Settings



Creating Database Backups

Copyright © 2024 Trivera Technologies LLC. | www.triveratech.com

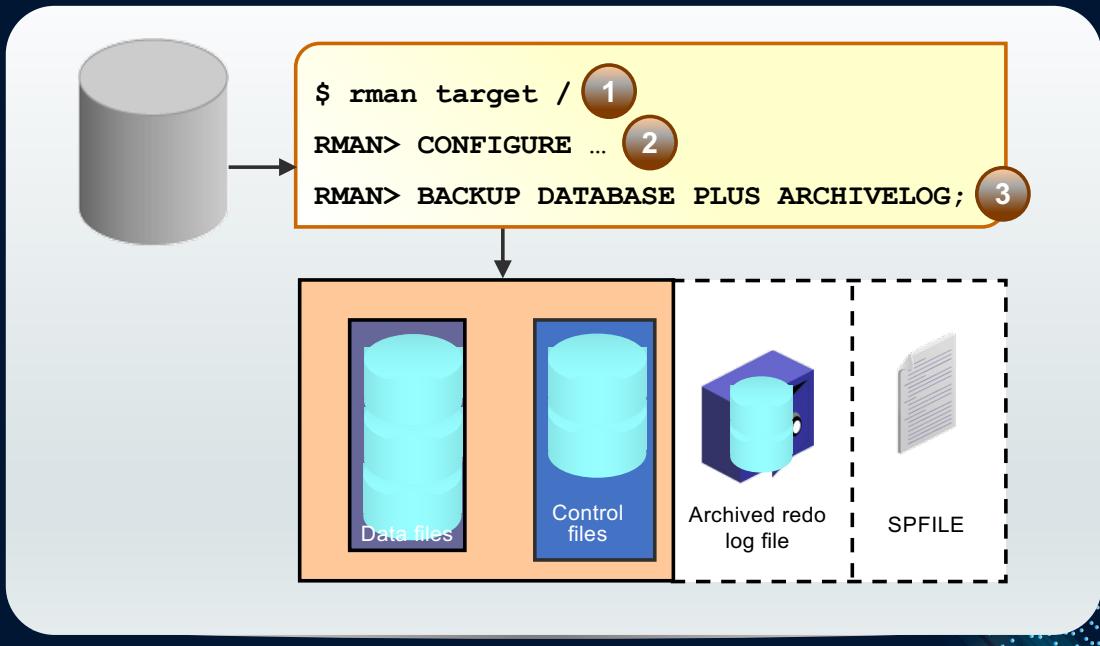
Experience is Everything

Objectives

After completing this lesson, you should be able to:

- Create whole backups
- Create full and incremental backups
- Configure block change tracking
- Use Oracle-suggested backup strategy
- Back up the control file to a trace file
- Report and manage backups

Using RMAN Commands to Create Backups



Syntax and Clauses in RMAN

```
$ export ORACLE_SID=cdb1  
$ rman TARGET /           $ rman TARGET jim@pdb1
```

- DATABASE keyword operates on all PDBs and CDB root or on only one PDB.

```
RMAN> BACKUP DATABASE;  
RMAN> RECOVER DATABASE;
```

- PLUGGABLE DATABASE keywords operate on individual PDBs.

```
RMAN> BACKUP PLUGGABLE DATABASE hr_pdb, sales_pdb;  
RMAN> RECOVER PLUGGABLE DATABASE hr_pdb;
```

- Back up, restore, recover the CDB root using CDB\$ROOT keyword.

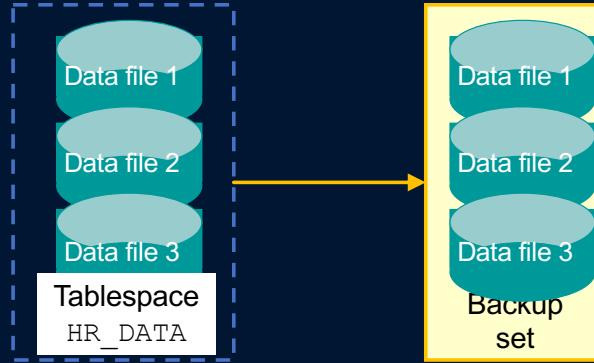
```
RMAN> BACKUP PLUGGABLE DATABASE "CDB$ROOT";
```

- Qualify tablespace of PDB with PDB name.

```
RMAN> BACKUP TABLESPACE sales_pdb:tbs2;  
RMAN> RESTORE TABLESPACE system;
```

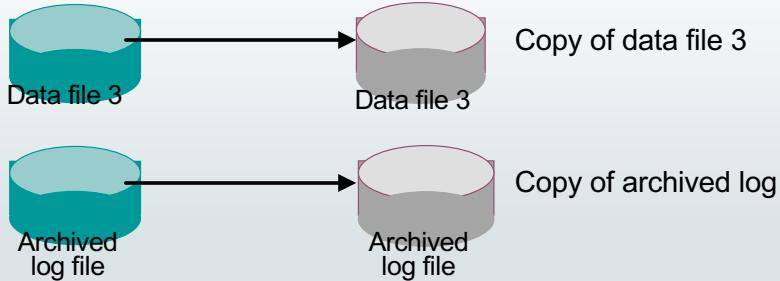
Creating Backup Sets

```
RMAN> BACKUP AS BACKUPSET  
2> FORMAT '/BACKUP/df_%d_%s_%p.bus'  
3> TABLESPACE hr_data;
```

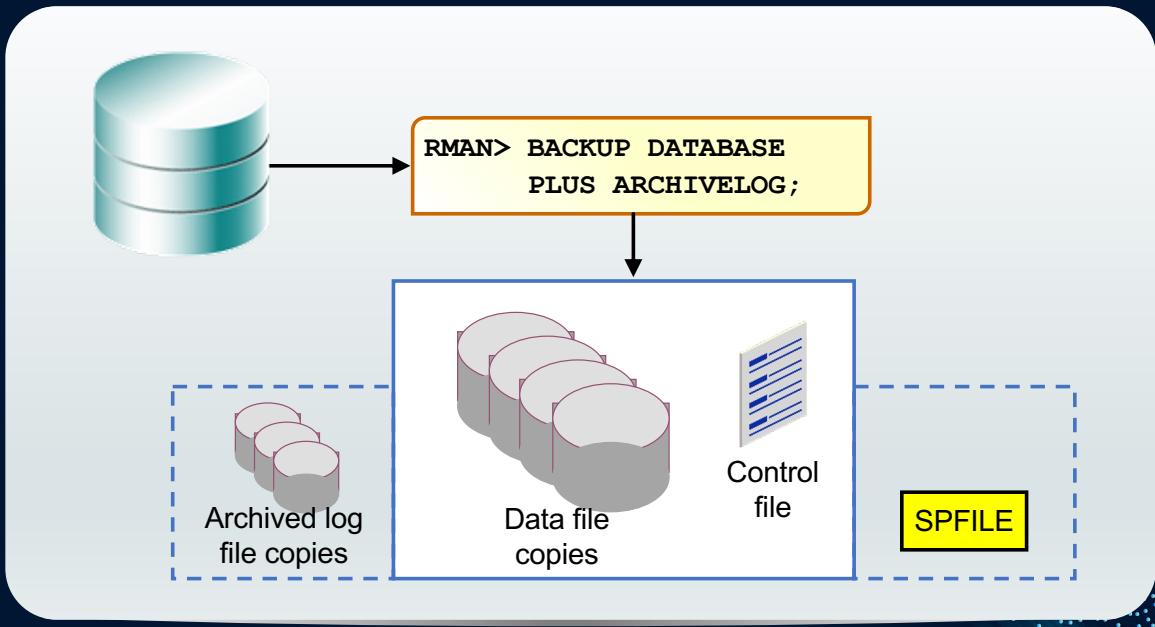


Creating Image Copies

```
RMAN> BACKUP AS COPY DATAFILE '/ORADATA/users_01_db01.dbf';
RMAN> BACKUP AS COPY ARCHIVELOG LIKE '/arch%';
```



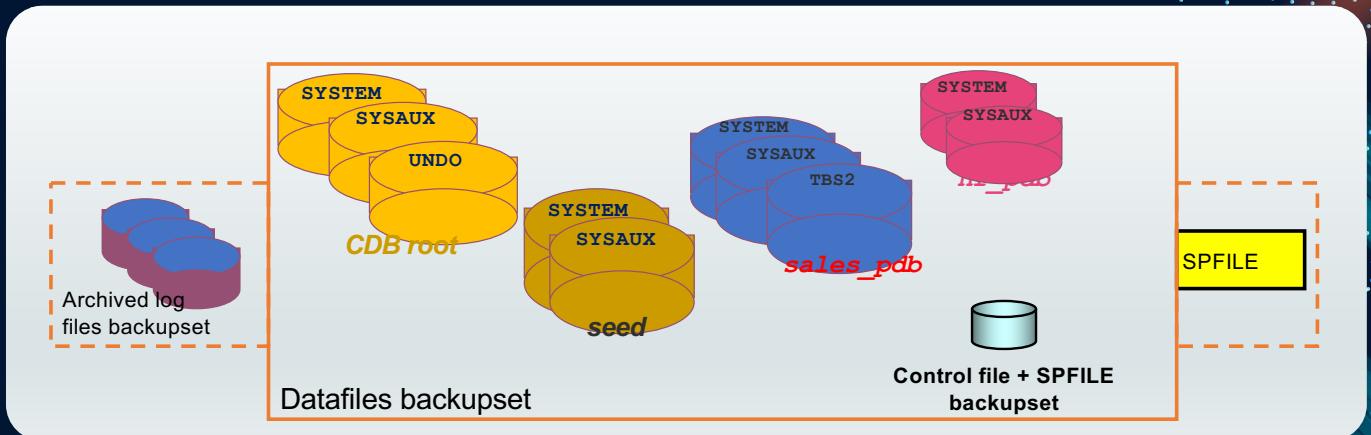
Creating a Whole Database Backup



CDB Backup: Whole CDB Backup

Back up all PDBs datafiles and CDB root files.

```
$ rman TARGET /  
RMAN> BACKUP DATABASE PLUS ARCHIVELOG;
```

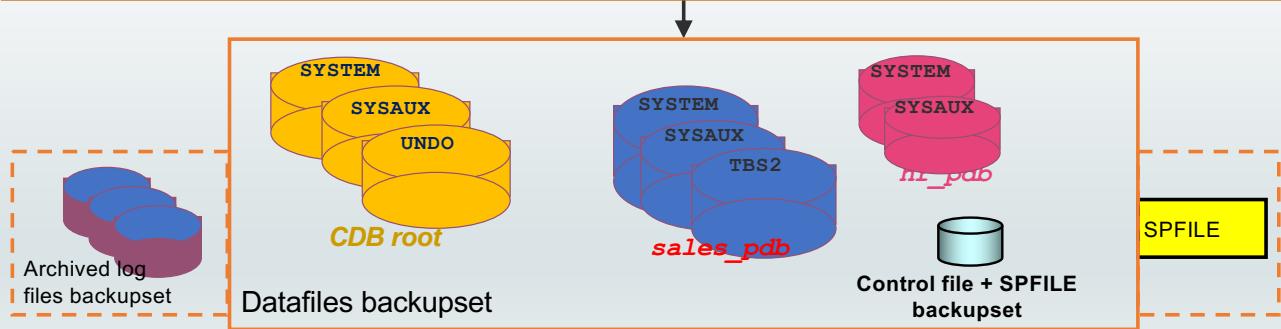


CDB Backup: Partial CDB Backup

Back up the CDB root and/or individual PDBs.

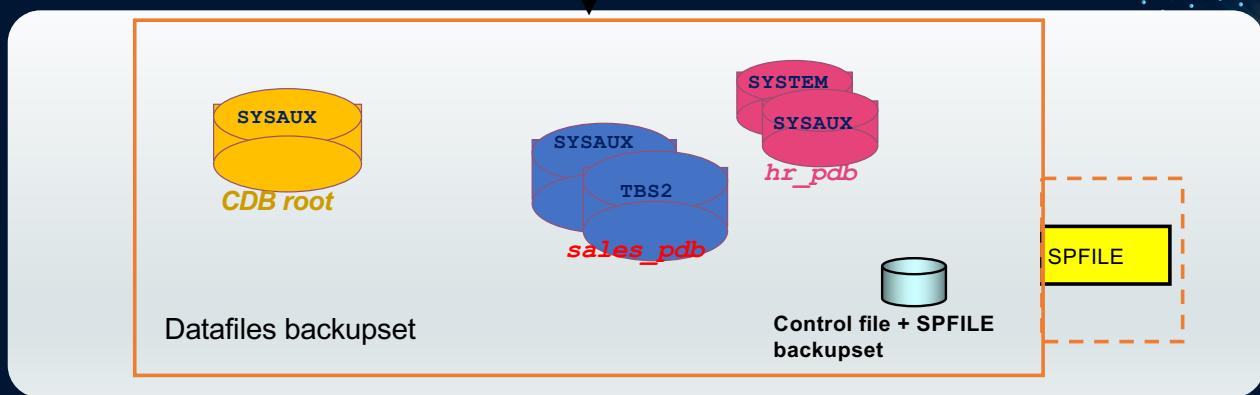
```
$ rman TARGET /
RMAN> BACKUP PLUGGABLE DATABASE "CDB$ROOT", sales_pdb;
RMAN> BACKUP PLUGGABLE DATABASE hr_pdb PLUS ARCHIVELOG;
```

```
$ rman TARGET sys@hr_pdb
RMAN> BACKUP DATABASE;
```



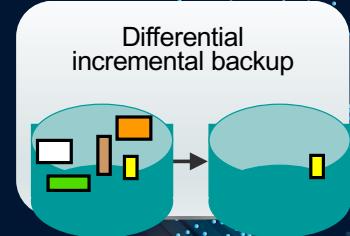
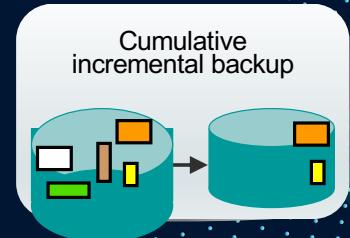
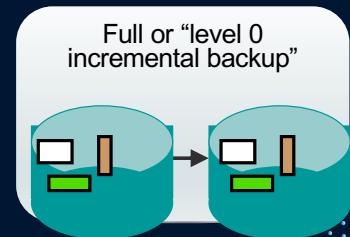
PDB Backup: Partial PDB Backup

```
$ rman TARGET /
RMAN> REPORT SCHEMA;
RMAN> BACKUP TABLESPACE sales_pdb:tbs2;
RMAN> BACKUP TABLESPACE hr_pdb:system, sales_pdb:sysaux;
RMAN> BACKUP TABLESPACE sysaux, hr_pdb:sysaux;
```



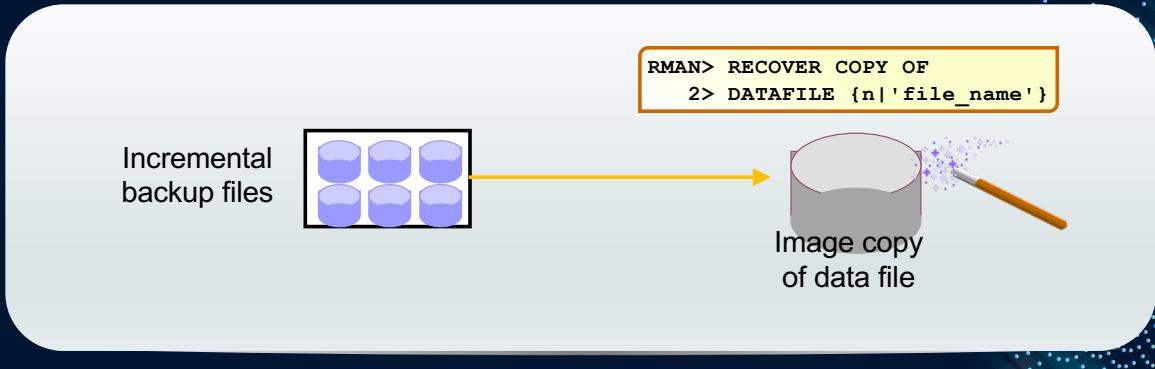
Review: RMAN Backup Types

- A *full backup* contains all used data file blocks.
- A *level 0 incremental backup* is equivalent to a full backup that has been marked as level 0.
- A *cumulative level 1 incremental backup* contains only blocks modified since the last level 0 incremental backup.
- A *differential level 1 incremental backup* contains only blocks modified since the last incremental backup.



Incrementally Updated Backups

- Image copies are updated with all changes up to the incremental backup SCN.
- Incremental backup reduces the time required for media recovery.
- With incrementally updated backups, you can use the `SWITCH` command during the recovery operation.



Incrementally Updated Backups: Example

If you execute these commands daily:

```
RMAN> recover copy of database with tag 'daily_inc';
RMAN> backup incremental level 1 for recover of copy
2> with tag 'daily_inc' database;
```

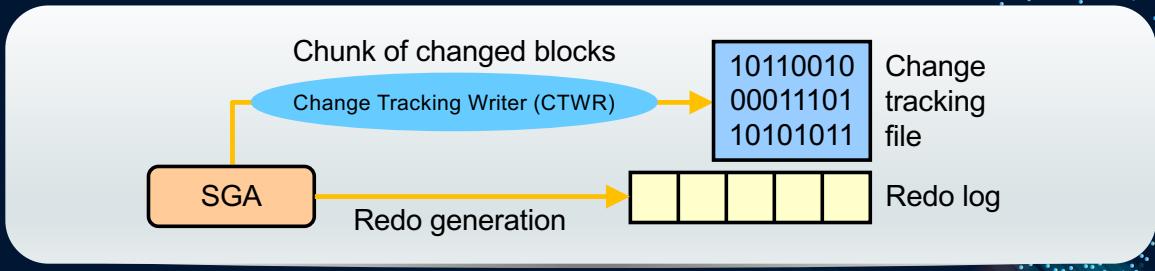
This is the result:

	RECOVER	BACKUP
Day 1	Nothing	Create image copies
Day 2	Nothing	Create incremental level 1
Day 3 and onward	Recover copies based on incremental	Create incremental level 1

Fast Incremental Backup

Implemented by block change tracking, which:

- Maintains a record of block chunks that have changed since the last backup
- Writes this record to a file, as redo is generated
- Is automatically accessed when a backup is done and can make the backup complete more quickly
- Is optimized for up to eight incremental backups
- Is recommended if the changes are less than 20 percent



Maintaining the Block Change Tracking File

- The `DB_CREATE_FILE_DEST` initialization parameter provides the default destination.
- Enable or disable with:

```
ALTER DATABASE
{ENABLE|DISABLE} BLOCK CHANGE TRACKING
[USING FILE '...']
```

- Rename the block change tracking file with the `ALTER DATABASE RENAME` command.
(The database must be in `MOUNT` state.)

Monitoring Block Change Tracking

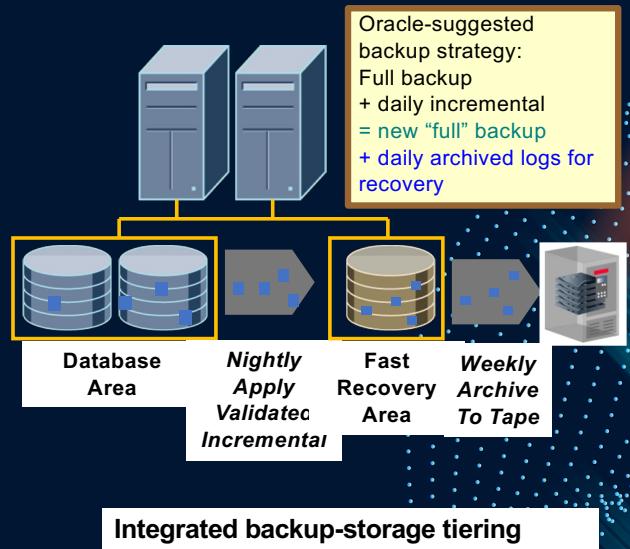
```
SQL> SELECT filename, status, bytes
  2  FROM    v$block_change_tracking;
```

```
SQL> SELECT file#, avg(datafile_blocks),avg(blocks_read),
  2      avg(blocks_read/datafile_blocks) * 100 AS PCT_READ_FOR_BACKUP,
  3      avg(blocks)
  4  FROM    v$backup_datafile
  5 WHERE   used_change_tracking = 'YES' AND incremental_level > 0
  6 GROUP   BY file#;
```

FILE#	BLOCKS_IN_FILE	BLOCKS_READ	PCT_READ_FOR_BACKUP	BLOCKS_BACKED_UP
1	56320	4480	7	462
2	3840	2688	70	2408
3	49920	16768	33	4457

Automatic Disk-to-Disk Backup and Recovery

- Integrated disk-to-disk backup and recovery:
Low-cost disks used for fast recovery area
- Fast incremental backups:
Back up only changed blocks
- Nightly incremental backup rolls forward recovery area backup:
No need to do full backups



Oracle-Suggested Backup

- Provides an out-of-the-box backup strategy based on the backup destination
- Sets up recovery window for backup management
- Schedules recurring and immediate backups

Full backup
+ daily incremental
= new “full” backup
+ daily archived logs for recovery



Backing Up the Control File to a Trace File

- A control file trace backup contains the SQL statement required to re-create the control files in the event that all control files are lost.
- It is recommended to do after each change in the physical structure of the database.
- Control file trace backups may be used to recover from loss of all control files.
- Choose your DBA tool: EM Express, Cloud Control, or command line.

Backing Up the Control File to a Trace File

- Control files can be backed up to a trace file, generating a SQL command to re-create the control file.
- Control file trace backups may be used to recover from the loss of all control files.

```
ALTER DATABASE BACKUP CONTROLFILE TO TRACE
```

388

Cataloging Additional Backup Files

Using the CATALOG command:

- To catalog existing backup files that are no longer listed in the control file
- To catalog files that were never included in the control file or recovery catalog
- To add the following file types to the recovery catalog:
 - CONTROLFILECOPY: Control file copies
 - DATAFILECOPY: Data file copies
 - BACKUPPIECE: Backup pieces
 - ARCHIVELOG: Archived redo log files
- With the START WITH option:

```
RMAN> CATALOG ARCHIVELOG '/disk1/arch_logs/archive1_731.log',  
'/disk1/arch_logs/archive1_732.log';  
RMAN> CATALOG START WITH '/tmp/arch_logs/';
```

Reporting on Backups

RMAN commands:

- **LIST:** Displays information about backup sets, proxy copies, and image copies recorded in the repository
- **REPORT:** Produces a detailed analysis of the repository
- **REPORT NEED BACKUP:** Lists all data files that require a backup
- **REPORT OBSOLETE:** Identifies files that are no longer needed to satisfy backup retention policies

Enterprise Manager Cloud Control:

- Graphical, customizable interface

Using Dynamic Views

Query the following dynamic views in the target database to obtain information about your backups:

- **V\$BACKUP_SET:** Backup sets created
- **V\$BACKUP_PIECE:** Backup pieces that exist
- **V\$DATAFILE_COPY:** Copies of data files on disk
- **V\$BACKUP_FILES:** Information about all files created when creating backups

Summary

In this lesson, you should have learned how to:

- Create whole backups
- Create full and incremental backups
- Configure block change tracking
- Use Oracle-suggested backup strategy
- Back up the control file to a trace file
- Report and manage backups

Practice Overview

- Backing up the Control File
- Verifying Automatic Backups of the Control File and SPFILE
- Creating a Whole Database Backup
- Creating Partial Database Backups
- Configuring Block Change Tracking
- Using Incremental Backup
- Backing Up Additional Database Files



Recovery

Experience is Everything

Copyright © 2024 Trivera Technologies LLC. | www.triveratech.com

Objectives

After completing this lesson, you should be able to describe the use of the RMAN recovery catalog.

RMAN Repository Data Storage: Comparison of Options

Control file:

- Simpler administration
- Default

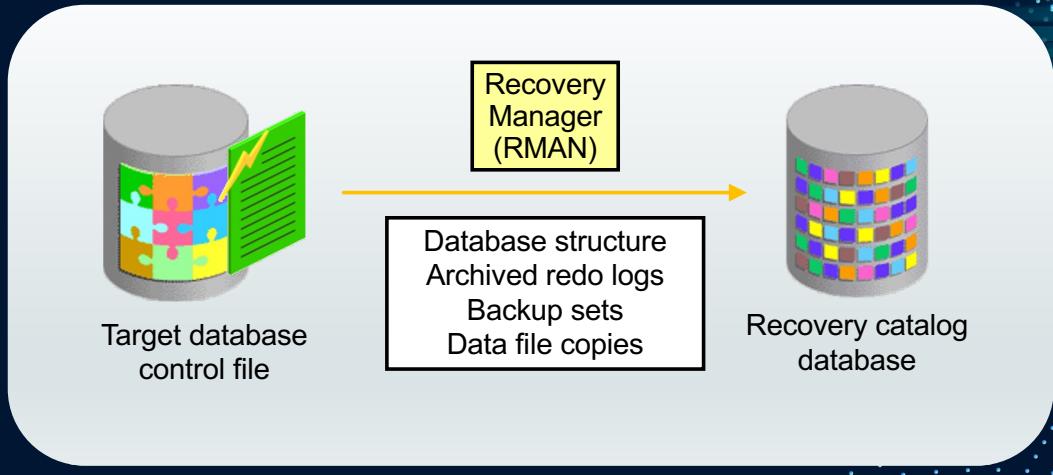
- Recovery catalog:
 - Replicates control file data
 - Stores more backup history
 - Services many targets
 - Stores RMAN scripts
 - Provides more protection options for metadata



Copyright © 2024 Trivera Technologies LLC. | www.triveratech.com

396

Storing Information in the Recovery Catalog



Copyright © 2024 Trivera Technologies LLC. | www.triveratech.com

397

Reasons to Use a Recovery Catalog

- Stores more historical information than the control file
- Enables you to use RMAN-stored scripts
- Enables you to create customized reports for all registered targets
- Enables you to use the `KEEP FOREVER` clause of the `BACKUP` command
- Allows you to list the data files and tablespaces that are or were in the target database *at a given time*
- Enables you to restore and recover following the loss of the control file because it preserves RMAN repository metadata

Summary

In this lesson, you should have learned how to describe the use of the RMAN recovery catalog.



Restore and Recovery Concepts and Flashback Technologies

Copyright © 2024 Trivera Technologies LLC. | www.triveratech.com

Experience is Everything

Objectives

After completing this lesson, you should be able to:

- Explain how to employ the best Oracle Database recovery technology for your failure situation
- Describe instance or crash recovery
- Describe complete recovery
- Describe point-in-time recovery
- Describe recovery with `RESETLOGS`

File Loss

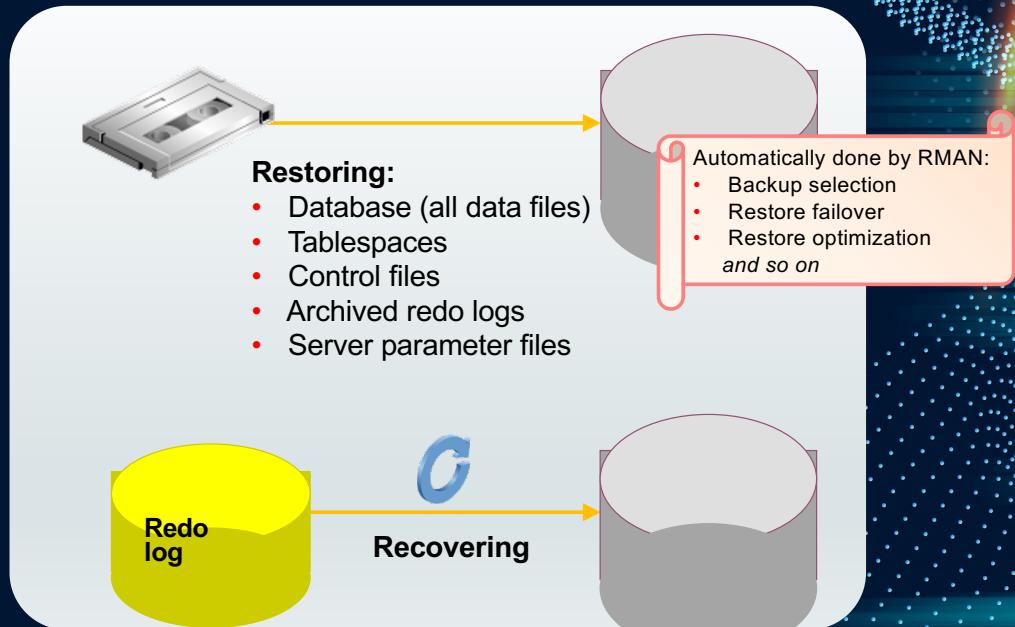
- File loss can be caused by:
 - User error
 - Application error
 - Media failure
- A *noncritical file* loss is one where the database remains open and available.
- The loss of a noncritical file can be addressed by:
 - Creating a new file
 - Rebuilding the file
 - Recovering the lost or damaged file

Data Repair Techniques

To respond to potential data loss:

- Physical failure (missing or corrupted data file):
 - Data Recovery Advisor
 - Data File Media Recovery
 - Block Recovery
- Logical failure (application or user error):
 - Logical Flashback Features
 - Oracle Flashback Database
 - Point-in-Time Recovery:
 - Database Point-in-Time Recovery (DBPITR)
 - Tablespace Point-in-Time Recovery (TSPITR)
 - Table Point-in-Time Recovery (TPITR)

Restoring and Recovering



Copyright © 2024 Trivera Technologies LLC. | www.triveratech.com

.404

Using RMAN RESTORE and RECOVER Commands

- **RESTORE command:** Restores database files from backup
- **RECOVER command:** Recovers the restored files by applying changes recorded in incremental backups and redo log files

```
RMAN> ALTER TABLESPACE inv_tbs OFFLINE IMMEDIATE;
RMAN> RESTORE TABLESPACE inv_tbs;
RMAN> RECOVER TABLESPACE inv_tbs;
RMAN> ALTER TABLESPACE inv_tbs ONLINE;
```

- The Enterprise Manager Cloud Control Recovery Wizard creates and runs an RMAN script to perform the recovery.

Instance Failure

Typical Causes	Possible Solutions
Power outage	Restart the instance by using the STARTUP command. Recovering from instance failure is automatic, including rolling forward changes in the redo logs and then rolling back any uncommitted transactions.
Hardware failure	Investigate the causes of failure by using the alert log, trace files, and Cloud Control.
Failure of one of the critical background processes	
Emergency shutdown procedures	

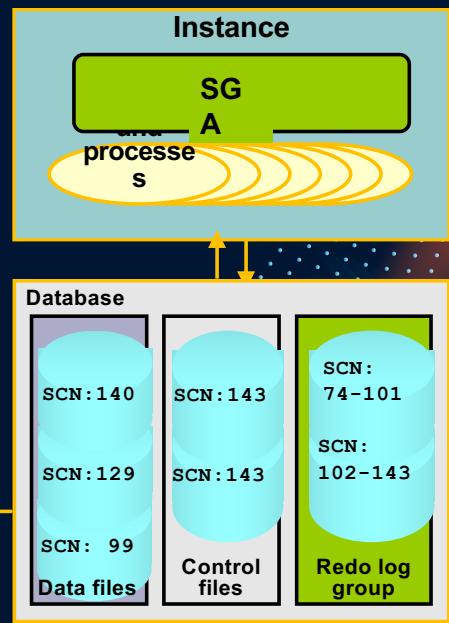
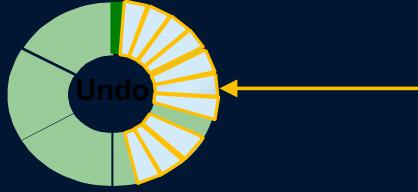
Instance Recovery

Automatic instance or crash recovery:

- Is caused by attempts to open a database whose files are not synchronized on shutdown
- Uses information stored in redo log groups to synchronize files
- Involves two distinct operations:
 - **Rolling forward:** Redo log changes (both committed and uncommitted) are applied to data files.
 - **Rolling back:** Changes that are made but not committed are returned to their original state.

Phases of Instance Recovery

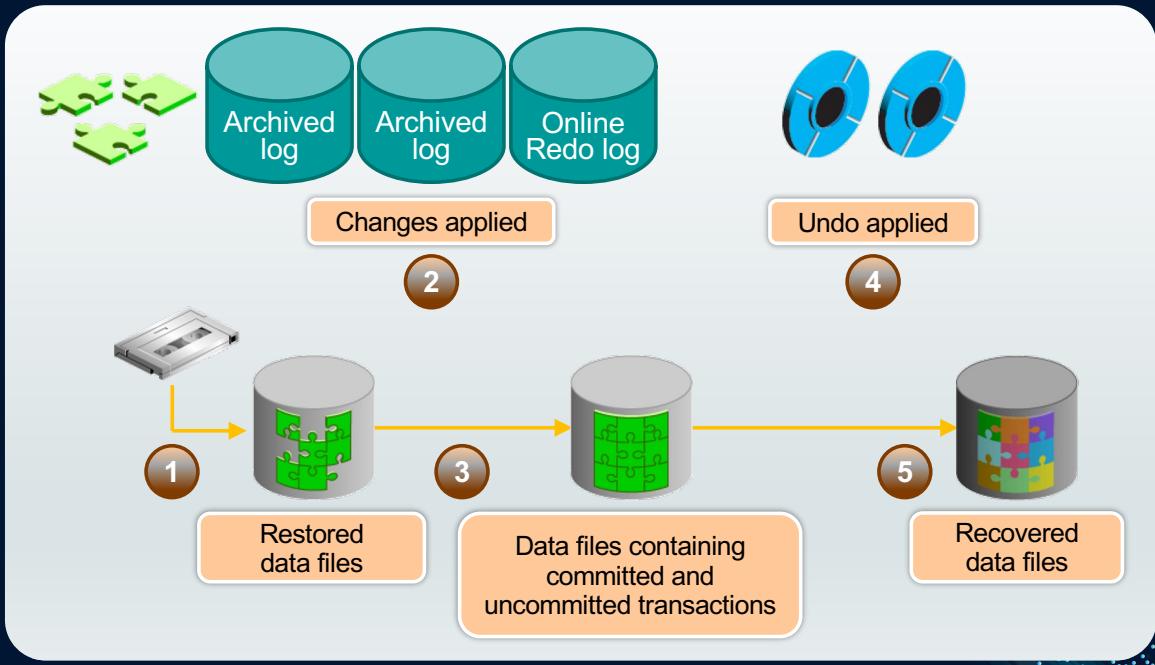
1. Startup instance (data files are out of sync)
2. Roll forward (redo)
3. Committed and uncommitted data in files
4. Database opened
5. Roll back (undo)
6. Committed data in files



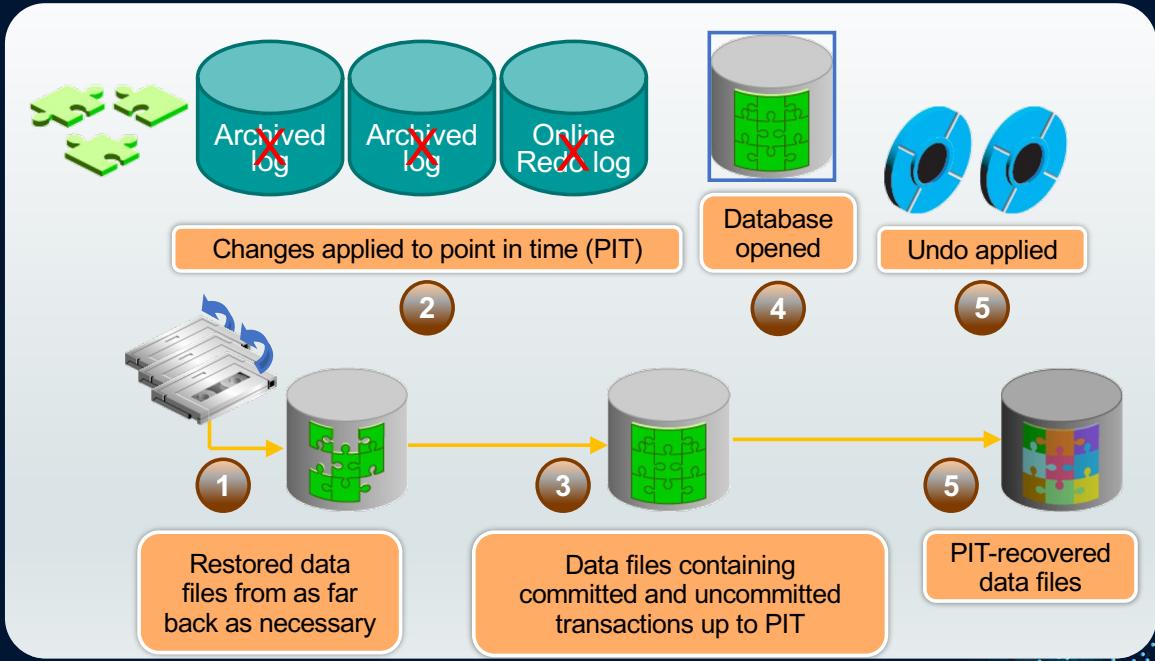
Media Failure

Typical Causes	Possible Solutions
Failure of disk drive	<ol style="list-style-type: none">1. Restore the affected file from backup.
Failure of disk controller	<ol style="list-style-type: none">2. Inform the database about a new file location (if necessary).
Deletion or corruption of a file needed for database operation	<ol style="list-style-type: none">3. Recover the file by applying redo information (if necessary).
Logical unit (LUN) in a storage array going offline	

Complete Recovery Process



Point-in-Time Recovery Process



Copyright © 2024 Trivera Technologies LLC. | www.triveratech.com

411

Recovery with the RESETLOGS Option

- Issue: Missing archive logs for target recovery SCN
- Workflow:
 1. Restore backups.
 2. Recover as far forward as the unbroken series of archive logs allows.
 3. Open the database with the `RESETLOGS` option.
A new `database incarnation` is automatically created to avoid confusion when two different redo streams have the same SCNs, but occurred at different times.

Note: Changes after the last applied archive log **are lost**.

Restore and Recovery Performance: Best Practices

- Minimize the number of archive logs to be applied by using incremental backups.
 - **Cumulative incremental backups:** Only the most recent cumulative incremental backup must be applied. This reduces tape library requests for media backups.
 - **Differential incremental backups:** All differential incremental level 1 backups since the restored data file backup must be applied.
- Use block media recovery for isolated block corruptions.
- Keep an adequate number of archived logs on disk.
- Increase RMAN buffer memory usage.
- Tune the database for I/O, DBWR performance, and CPU utilization.

Summary

In this lesson, you should have learned how to:

- Determine the best Oracle Database recovery technology for your failure situation
- Describe instance or crash recovery
- Describe complete recovery
- Describe point-in-time recovery
- Describe recovery with `RESETLOGS`



Diagnosing Failures

Copyright © 2024 Trivera Technologies LLC. | www.triveratech.com

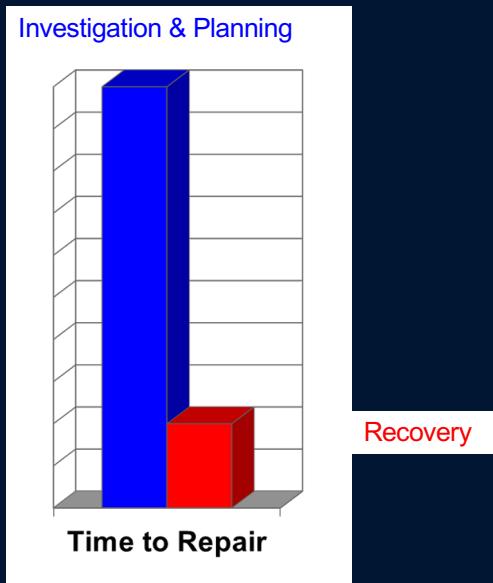
Experience is Everything

Objectives

After completing this lesson, you should be able to:

- Detect and repair database corruption
- Use the Automatic Diagnostic Repository
- Analyze instance recovery with ADRCI
- Use Data Recovery Advisor

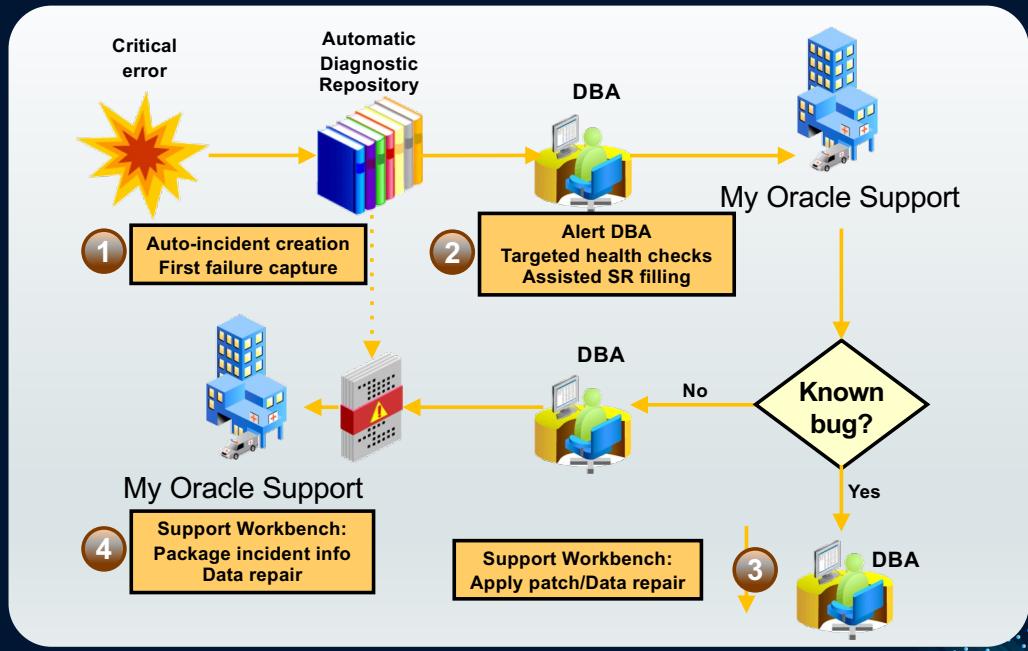
Reducing Problem Diagnosis Time



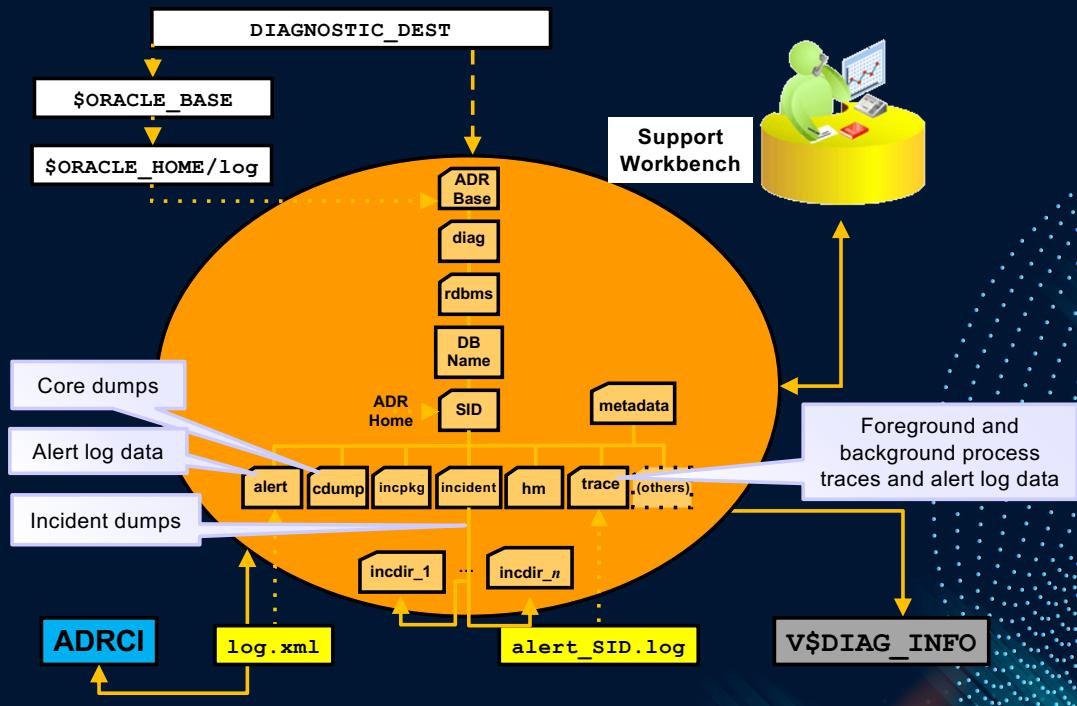
Oracle tools for data repair include:

- RMAN for physical media loss or corruptions
- Flashback for logical errors
- Data Guard for physical problems
- **Data Recovery Advisor** addresses:
 - Problem diagnosis (choosing the right solution can be error prone and time-consuming)
 - Incorrect choices (errors more likely during emergencies)

Automatic Diagnostic Workflow



Automatic Diagnostic Repository



ADR Command-Line Tool (ADRCI)

- ADRCI provides interaction with ADR from an operating system prompt.
- Using ADRCI, you can view diagnostic data within the ADR.

```
$ adrci
ADRCI: Release 12.1.0.1.0 - Production on Thu Nov 29 21:15:27 2012
Copyright (c) 1982, 2012, Oracle and/or its affiliates.
ADR base = "/u01/app/oracle"
ADRCI>
ADRCI> show incident
ADRCI> set editor gedit
ADRCI> show alert
...
ADR Home = /u01/app/oracle/diag/rdbms/em12rep/em12rep:
*****
INCIDENT_ID PROBLEM_KEY          CREATE_TIME
-----
4985        ORA 4031            2012-11-21 00:57:43.823000 +00:00
5161        ORA 4031            2012-11-21 00:58:17.284000 +00:00
2 incident info records fetched
```

V\$DIAG_INFO View

```
SQL> SELECT NAME, VALUE FROM V$DIAG_INFO;
```

NAME	VALUE
Diag Enabled	TRUE
ADR Base	/u01/app/oracle
ADR Home	/u01/app/oracle/diag/rdbms/orcl/orcl
Diag Trace	/u01/app/oracle/diag/rdbms/orcl/orcl/trace
Diag Alert	/u01/app/oracle/diag/rdbms/orcl/orcl/alert
Diag Incident	/u01/app/oracle/diag/rdbms/orcl/orcl/incident
Diag Cdump	/u01/app/oracle/diag/rdbms/orcl/orcl/cdump
Health Monitor	/u01/app/oracle/diag/rdbms/orcl/orcl/hm
Default Trace File	/u01/app/oracle/diag/.../trace/orcl_ora_11424.trc
Active Problem Count	3
Active Incident Count	8

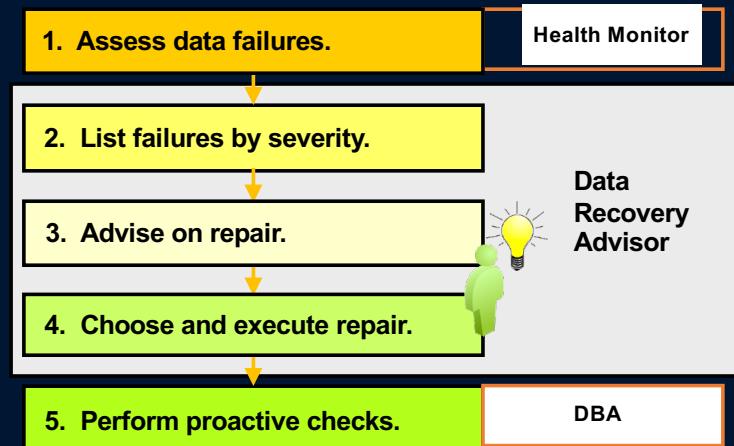


Data Recovery Advisor

- Fast detection, analysis, and repair of failures
- Minimizing disruptions for users
- Down time and runtime failures
- User interfaces:
 - EM GUI interface (several paths)
 - RMAN command line
- Supported database configurations:
 - Single-instance
 - Not RAC
 - Supporting failover to standby, but not analysis and repair of standby databases

Data Recovery Advisor

Reducing down time by eliminating confusion:



Data Failure: Examples

- Not accessible components, for example:
 - Missing data files at the OS level
 - Incorrect access permissions
 - Offline tablespace and so on
- Physical corruptions, such as block checksum failures or invalid block header field values
- Logical corruptions, such as inconsistent dictionary, corrupt row piece, corrupt index entry, or corrupt transaction
- Inconsistencies, such as control file is older or newer than the data files and online redo logs
- I/O failures, such as a limit on the number of open files exceeded, channels inaccessible, network or I/O error



Data Recovery Advisor RMAN Command-Line Interface

RMAN Command	Action
LIST FAILURE	Lists previously executed failure assessment
ADVISE FAILURE	Displays recommended repair option
REPAIR FAILURE	Repairs and closes failures (after ADVISE in the same RMAN session)
CHANGE FAILURE	Changes or closes one or more failures

List Data Failures

The RMAN LIST FAILURE command lists previously executed failure assessment:

- Including newly diagnosed failures
- Removing closed failures (by default)

Syntax:

```
LIST FAILURE
[ ALL | CRITICAL | HIGH | LOW | CLOSED |
  failnum[,failnum,...] ]
[ EXCLUDE FAILURE failnum[,failnum,...] ]
[ DETAIL ]
```



Advising on Repair

The RMAN ADVISE FAILURE command:

- Displays a summary of input failure list
- Includes a warning, if new failures appeared in ADR
- Displays a manual checklist
- Lists a single recommended repair option
- Generates a repair script (for automatic or manual repair)

```
Repair script:  
/u01/app/oracle/diag/rdbms/orcl/orcl/hm/reco_2979  
128860.hm  
RMAN>
```

Executing Repairs

The RMAN REPAIR FAILURE command:

- Follows the ADVISE FAILURE command
- Repairs the specified failure
- Closes the repaired failure

Syntax:

```
REPAIR FAILURE
[USING ADVISE OPTION integer]
[ { NOPROMPT | PREVIEW} ]...
```



Classifying (and Closing) Failures

The RMAN CHANGE FAILURE command:

- Changes the failure priority (except for CRITICAL)
- Closes one or more failures

Example:

```
RMAN> change failure 5 priority low;
List of Database Failures
=====
Failure ID Priority Status      Time Detected Summary
-----
5          HIGH    OPEN       20-DEC-18   one or more datafiles are missing
Do you really want to change the above failures (enter YES or NO)? yes
changed 1 failures to LOW priority
```

Data Recovery Advisor Views

Querying V\$ views:

- **V\$IR_FAILURE:** List of all failures, including closed ones (result of the LIST FAILURE command)
- **V\$IR_MANUAL_CHECKLIST:** List of manual advice (result of the ADVISE FAILURE command)
- **V\$IR_REPAIR:** List of repairs (result of the ADVISE FAILURE command)
- **V\$IR_FAILURE_SET:** Cross-reference of failure and advice identifiers



Summary

In this lesson, you should have learned how to:

- Detect and repair database corruption
- Use the Automatic Diagnostic Repository
- Analyze instance recovery with ADRCI
- Use Data Recovery Advisor

Practice Overview

- Diagnosing and Repairing Database Failure

Thanks again for joining us!

- We truly appreciate your time. Please complete the End of Course Survey.
- Any questions?
 - Review the full Course Guide for Course Tips, Resources & Next Step Learning Plans
 - Feel Free to Reach Out: Info@triveratech.com
 - See full list of Oracle, Database, Tooling, Coding, AI / Machine Learning, Data Science Courses & SkillJourneys: www.triveratech.com
- Free Courses, Articles, Resources & Offers



Let's Connect! Follow Us for Free Courses, Articles, Resources & Offers:
LinkedIn: @TriveraTech



Subscribe to our Channel for Free Courses & Events
YouTube: @TriveraTech



Copyright © 2024 Trivera Technologies LLC. | www.triveratech.com

436