DOYENSYS
VALUE INNOVATION

Data Dictionary and Dynamic Performance Views

Data Dictionary

In Oracle Database, the data dictionary views (also known as data dictionary tables or data dictionary views) are a set of read-only views that provide metadata information about the database.

They are part of the Oracle data dictionary, which is a collection of database tables and views containing information about the database objects, their structures, and various aspects of the database configuration.

The data dictionary views are crucial for database administrators, developers, and users to query and understand the database schema, objects, and configurations.

Location:

Data dictionary views are stored in the SYS schema, and most of them have names prefixed with "DBA_", "ALL_", or "USER_" based on the level of access required.

Data Dictionary

Access Levels:

There are three levels of data dictionary views:

- **DBA_ Views:** Provide information about all objects in the database. Accessible to users with the DBA role.
- **ALL_ Views:** Provide information about objects accessible to the current user. Accessible to all users.
- **USER_ Views:** Provide information about objects owned by the current user. Accessible to the owner.

Data Dictionary

Common Data Dictionary Views:

Some commonly used data dictionary views include:

- DBA_TABLES, ALL_TABLES, USER_TABLES: Information about tables.
- DBA_COLUMNS, ALL_TAB_COLUMNS, USER_TAB_COLUMNS: Information about table columns.
- DBA_INDEXES, ALL_INDEXES, USER_INDEXES: Information about indexes.
- DBA_VIEWS, ALL_VIEWS, USER_VIEWS: Information about views.
- DBA_PROCEDURES, ALL_PROCEDURES, USER_PROCEDURES: Information about stored procedures and functions.
- DBA_USERS, ALL_USERS, USER_USERS: Information about database users.

Dynamic Performance Views:

In addition to the traditional data dictionary views, Oracle Database also includes dynamic performance views, commonly prefixed with "V$". These views provide real-time information about the performance and internal workings of the database.

Some of the commonly user performance views:

V$SESSION - Contains information about sessions connected to the database, such as session ID, username, program, and status.

V$SQL - Provides details about SQL statements currently in the SQL cache, including SQL text, execution plans, and statistics.

V$LOCK - Displays information about locks currently held or being requested.

User Performance Views:

V$TRANSACTION - Shows information about transactions currently active in the database.

V$SESSION_EVENT - Displays information about events currently being waited on by sessions.

V$SYSSTAT - Provides a summary of various statistics for the database.

V$PARAMETER - Contains information about initialization parameters and their current values.

V$DATAFILE - Lists datafiles associated with the database.

V$LOG - Displays information about redo log files.

V$LOGFILE - Provides details about individual members of redo log groups.

User Performance Views:

V$TABLESPACE - Lists information about tablespaces in the database.

V$INSTANCE - Displays information about the Oracle Database instance.

V$BACKUP - Provides information about RMAN backups.

V$DATAFILE_HEADER - Displays information about the headers of datafiles.

V$SESSION_LONGOPS - Shows information about long-running operations currently being executed.

catalog.sql

The catalog.sql script in Oracle Database is a SQL script provided by Oracle that is used to create and install the data dictionary views and tables necessary for the Oracle Data Dictionary.

The Data Dictionary is a collection of database tables and views containing metadata about the database itself, its structure, and various aspects of its configuration.

These views are essential for the functioning of the Oracle Database and for users, administrators, and applications to query information about the database.

catalog.sql

Creation of Data Dictionary Views:

- The primary purpose of catalog.sql is to create the data dictionary views and tables that make up the Oracle Data Dictionary.
- These views provide metadata information about database objects, users, privileges, and other aspects of the database.

Population of Data Dictionary Tables:

- The script may also include statements to populate the data dictionary tables with initial data.
- This data includes information about the database itself, tablespaces, users, roles, and other database-related entities.

catalog.sql

Initialization of the Data Dictionary:

- Running catalog.sql during database creation or upgrade initializes the Data Dictionary.
- The Data Dictionary is a crucial component for the functioning of the Oracle Database, and its proper initialization is essential for the overall health of the database.

Dependency for Other Scripts:

- Various other scripts and components within the Oracle Database depend on the existence and proper functioning of the Data Dictionary.
- Running catalog.sql is often a prerequisite for other scripts and processes that rely on the information stored in the Data Dictionary.

catalog.sql

Maintenance and Upgrade:

- During database upgrades or patching, the catalog.sql script may need to be run to ensure that the Data Dictionary is updated to match the new version of the database software.
- This script helps maintain consistency between the database software and the metadata it manages.

Standardization Across Databases:

- catalog.sql contributes to standardizing the structure of the Data Dictionary across different Oracle Database installations.
- This ensures a consistent set of views and tables, making it easier for developers and administrators to work with Oracle databases.

catproc.sql

The catproc.sql script in Oracle Database is used to create and install the database objects related to the Oracle Database PL/SQL packages and procedures.

This script plays a crucial role in setting up the infrastructure for Oracle Database features such as PL/SQL packages, procedures, and functions.

Running catproc.sql is often part of the post-installation steps or database creation process, and it helps ensure that the database has the necessary components for PL/SQL functionality.

Installation of PL/SQL Packages:

- The primary purpose of catproc.sql is to create and install PL/SQL packages and procedures that are essential for various features and functionalities within the Oracle Database.
- It includes packages related to data dictionary management, dynamic SQL execution, and other PL/SQL utilities.

catproc.sql

Dependency for Other Scripts and Components:

- Various Oracle Database components, applications, and scripts depend on the PL/SQL packages and procedures provided by catproc.sql.
- Running this script ensures that these components have access to the required PL/SQL functionality.

Enhancement of PL/SQL Features:

- catproc.sql may include enhancements and updates to existing PL/SQL features, providing bug fixes, optimizations, or new functionalities.
- This script is often part of the overall maintenance and improvement process for the PL/SQL environment.

catproc.sql

Standardization Across Databases:

- catproc.sql contributes to standardizing the PL/SQL environment across different Oracle Database installations.
- It ensures that the necessary PL/SQL packages and procedures are consistently present, making it easier to develop and maintain applications across different databases.

Support for Database Tools and Applications:

- Many Oracle Database tools, utilities, and applications rely on the PL/SQL infrastructure provided by catproc.sql.
- Running this script is important for ensuring compatibility with various Oracle Database development and management tools.

Table of contents

Maintaining the Control File

What are we going to see ?

- Control File Contents
- Multiplexing the Control File
- Obtaining Control File Information

Maintaining the Control File

Control File Contents

Definition and Purpose:

The control file is a critical component that maintains metadata about the database. It includes information about database checkpoints, the database name, creation timestamp, archive redo log history, and more.

Information Stored in Control Files:

- Database name
- Creation timestamp
- Archive redo log history
- Redo log file information
- Datafile information
- Checkpoint information
- Server parameter information

Maintaining the Control File

Role in Database Recovery:

Control files play a crucial role in database recovery, aiding in restoring the database to a consistent state after a failure.

View control file location
SQL> show parameter control_files;

Extract control file details
SELECT * FROM V$CONTROLFILE;

V$CONTROLFILE_RECORD_HISTORY

Task : How to view the control file contents.

Maintaining the Control File

Multiplexing the Control File

Importance of Multiplexing:

• Multiplexing control files involves creating multiple copies of control files. This is essential for data recovery in case of control file corruption or loss.

Multiplexing:

- Oracle Multiplexing (Static & Dynamic)

Best Practices for Control File Multiplexing:

- Store copies on separate disks.
- Maintain at least two copies.

Control Files

DEMO SESSION

Maintaining Redo Log Files

What are we going to see ?

- Using Redo Log Files
- Adding, Drop, Relocating, Renaming Redo logfiles
- Obtaining Group and Member Information

Maintaining Redo Log Files

Redo log File

- They capture changes made to the database so that in the event of a failure, the database can be recovered to a consistent state.
- The redo log consists of two or more redo log groups, each containing one or more redo log files.
- Redo log files are typically multiplexed to provide redundancy and avoid a single point of failure.
- Redo log files contain a record of changes, known as redo entries or redo records.

Maintaining Redo Log Files

Redo log File

- Each redo entry includes information about the change, such as the type of operation, the table affected, and the data before and after the change.
- Redo log files are used in a circular fashion, and when one log file is filled, Oracle switches to the next available file.
- Archived redo log files are copies of redo log files that have been archived for backup and recovery purposes.
- During instance recovery or media recovery, Oracle uses the redo log to reapply changes made since the last checkpoint.

Maintaining Redo Log Files

Redo log File - Switching ,Archiving , Recovery:

- They capture changes made to the database so that in the event of a failure, the database can be recovered to a consistent state.
- The redo log consists of two or more redo log groups, each containing one or more redo log files.
- Redo log files are typically multiplexed to provide redundancy and avoid a single point of failure.
- Redo log files contain a record of changes, known as redo entries or redo records.
- Each redo entry includes information about the change, such as the type of operation, the table affected, and the data before and after the change.

Maintaining Redo Log Files

Redo log File - Switching ,Archiving , Recovery:

- Redo log files are used in a circular fashion, and when one log file is filled, Oracle switches to the next available file.
- Archived redo log files are copies of redo log files that have been archived for backup and recovery purposes.
- During instance recovery or media recovery, Oracle uses the redo log to reapply changes made since the last checkpoint.

Maintaining Redo Log Files

Limitations of Redo Log Files:

1.Limited Size:

- Redo log files have a finite size, and once they are filled, the database must switch to the next available log file. If the size is not properly managed, it can lead to issues.

2.I/O Overhead:

- Writing to the redo log imposes I/O overhead on the database. Excessive I/O operations on redo log files can affect performance.

3.Performance Impact during Log Switch:

- Log switches, where the database switches from one redo log file to another, can impact performance. It's crucial to monitor and manage log switches to avoid performance degradation.

4.Archiving Overhead:

- Archiving redo log files for backup and recovery purposes introduces additional I/O overhead and storage requirements.

5.Risk of Data Loss:

- If a failure occurs before changes are written to the redo log, there is a risk of data loss. Therefore, proper sizing and monitoring of the redo log files are essential.

6.Multiplexing Complexity:

- While multiplexing redo log files provides redundancy, it adds complexity to the administration of the database.

Maintaining Redo Log Files

Redo Log Files Commands:

To view redo log information
SELECT * FROM V$LOG;
SELECT * FROM V$LOGFILE;

Managing Redo Log Files:

Adding New Redo Log Groups
ALTER DATABASE ADD LOGFILE GROUP ...;
Dropping Redo Log Groups
ALTER DATABASE DROP LOGFILE GROUP ...;
Relocating and Renaming Redo Log Files
ALTER DATABASE RENAME FILE ... TO ...;

Task: Log group & Log member status

Redo Log Files

DEMO SESSION

Tablespace

What are we going to see ?

- Overview of Tablespaces
- Different Types of Tablespaces
- Adding Datafiles to existing Tablespace
- Renaming and Resizing Datafiles
- Creating and Managing Temporary Tablespace and Temporary Tablespace groups

Tablespace

Overview of Tablespaces:

- Tablespaces play a crucial role in managing the physical and logical storage of data objects such as tables, indexes, and clusters.

- A tablespace is a logical storage container within an Oracle database. It consists of one or more data files and is used to group related logical structures.

Tablespace

Types of Tablespaces:

1. System Tablespace
    - Contains the data dictionary and core database functionality.
2. User Tablespaces
    - Created by users to store their data.
3. Temporary Tablespaces
    - Used for sorting and joining operations.
4. Undo Tablespaces
    - Stores undo information for rollback operations.

Tablespace

Tablespace commands:

View existing tablespaces

```
SELECT tablespace_name FROM dba_tablespaces;
```

Create a new tablespace

```
CREATE TABLESPACE tablespace_name DATAFILE 'path_datafile' SIZE 100M;
```

Add datafile to a tablespace

```
ALTER TABLESPACE tablespace_name ADD DATAFILE 'path_to_new_datafile' SIZE 50M;
```

Rename a datafile

```
ALTER DATABASE RENAME FILE 'old_path' TO 'new_path';
```

Tablespace commands

Resize a datafile

```
ALTER DATABASE DATAFILE 'path_datafile' RESIZE 200M;
```

Create a temporary tablespace

```
CREATE TEMPORARY TABLESPACE temp_ts TEMPFILE 'path_tempfile' SIZE 50M;
```

Set a tablespace as the default temporary tablespace

```
ALTER DATABASE DEFAULT TEMPORARY TABLESPACE temp_ts;
```

Setting and managing user quotas on a specific tablespace:

```
ALTER USER username QUOTA quota_in_MB ON tablespace_name;
```

Querying information about tablespaces:

```
SELECT tablespace_name, file_name, bytes, status FROM dba_data_files;
```

Tablespace commands

Tablespace commands:

Creating an Undo Tablespace:

```
CREATE UNDO TABLESPACE undotbs1 DATAFILE '/path/to/undotbs1.dbf' SIZE 200M
AUTOEXTEND ON NEXT 20M MAXSIZE 1G;
```

Setting the Default Undo Tablespace:

```
ALTER SYSTEM SET UNDO_TABLESPACE = undotbs1;
```

Changing the Size of an Undo Tablespace:

```
ALTER DATABASE DATAFILE 'path_datafile.dbf' RESIZE new_size_in_MB;
```

Dropping an Undo Tablespace:

```
DROP TABLESPACE undo_tablespace_name INCLUDING CONTENTS AND DATAFILES;
```

Tablespace commands

Monitoring Undo Tablespace Usage:

SELECT tablespace_name, status, contents, bytes, maxbytes FROM dba_tablespaces WHERE tablespace_name = 'undo_tablespace_name’;

Creating a Temporary Tablespace Group:

CREATE TEMPORARY TABLESPACE GROUP temp_group_name TEMPFILE 'path_to_tempfile1.dbf' SIZE size_in_MB, 'path_to_tempfile2.dbf' SIZE size_in_MB;

Adding a Temporary Tablespace to a Group:

ALTER TEMPORARY TABLESPACE GROUP temp_group_name ADD TEMPORARY TABLESPACE temp_tablespace_name;

Dropping a Temporary Tablespace from a Group:

ALTER TEMPORARY TABLESPACE GROUP temp_group DROP TEMPORARY TABLESPACE temp_ts;

Tablespace commands

DEMO SESSION