Lab 6.1: Working with Initialization Parameters and Viewing Diagnostic Information

Objective:

To investigate initialization parameter files, view initialization parameters using SQL*Plus, and modify initialization parameters.

Part A: Investigating Initialization Parameter Files

Steps:

1. **Locate the Initialization Parameter Files**

   Initialization parameter files are typically found in the `$ORACLE_HOME/dbs` directory. List the contents of this directory:

   ```
   cd /u01/app/oracle/product/19.3.0/dbhome_1/dbs
   ls -l
   ```

   Expected Files:

   - `initCDBLAB.ora` : Initialization parameter file for `CDBLAB` .
   - `spfileCDBLAB.ora` : Server parameter file for `CDBLAB` .

2. **View the Contents of the Initialization Parameter File**

   Use a text editor or a command like `cat` to view the contents of `initCDBLAB.ora` :

   ```
   cat /u01/app/oracle/product/19.3.0/dbhome_1/dbs/initCDBLAB.ora
   ```

   Key Parameters:

   - `db_name` : Name of the database.
   - `control_files` : Location of control files.
   - `db_block_size` : Database block size.
   - `db_create_file_dest` : Default location for data files.
   - `db_recovery_file_dest` : Location for the recovery area.

3. **View the Contents of the Server Parameter File**

   Use the `strings` command to view the `spfileCDBLAB.ora` as it is a binary file:

   ```
   strings /u01/app/oracle/product/19.3.0/dbhome_1/dbs/spfileCDBLAB.ora
   ```

Part B: Viewing Initialization Parameters by Using SQL*Plus

Steps:

1. **Connect to SQL*Plus as SYSDBA**

   Start SQL*Plus and connect as SYSDBA:

   ```
   sqlplus / as sysdba
   ```

2. **View Initialization Parameters Using SHOW PARAMETER**

   Use the `SHOW PARAMETER` command to view parameters:

```
SHOW PARAMETER db_name;
SHOW PARAMETER control_files;
```

Explanation:

- `SHOW PARAMETER db_name` : Displays the database name.
- `SHOW PARAMETER control_files` : Displays the locations of the control files.

3. **Query V$PARAMETER View**

   Query the `V$PARAMETER` view to get information about initialization parameters:

```
SELECT name, value, isdefault, ismodified, issys_modifiable
FROM v$parameter
WHERE name = 'db_name';
```

Explanation:

- `name` : Parameter name.
- `value` : Current value of the parameter.
- `isdefault` : Indicates if the parameter value is the default.
- `ismodified` : Indicates if the parameter has been modified.
- `issys_modifiable` : Indicates if the parameter is modifiable at the system level.

4. **Query V$SPPARAMETER View**

   Query the `V$SPPARAMETER` view to get information about server parameters:

```
SELECT name, value
FROM v$spparameter
WHERE name = 'control_files';
```

5. **Query V$PARAMETER2 View**

   Query the `V$PARAMETER2` view for additional details about parameters:

```
SELECT name, value, type
FROM v$parameter2
WHERE name = 'db_block_size';
```

6. **Query V$SYSTEM_PARAMETER View**

   Query the `V$SYSTEM_PARAMETER` view to see system-wide parameters:

```
SELECT name, value, issys_modifiable
FROM v$system_parameter
WHERE name = 'db_recovery_file_dest';
```

Part C: Modifying Initialization Parameters by Using SQL*Plus

Steps:

1. **Modify a Session-Level Parameter**

   Change a session-level parameter using the `ALTER SESSION` command:

```
ALTER SESSION SET sort_area_size = 65536;
```

Verification:

```
SHOW PARAMETER sort_area_size;
```

2. **Modify a Dynamic System-Level Parameter**

   Change a dynamic system-level parameter using the `ALTER SYSTEM` command:

```
ALTER SYSTEM SET shared_pool_size = '500M' SCOPE=BOTH;
```

   Explanation:

   - `SCOPE=BOTH` : Applies the change both in memory and updates the SPFILE.

   Verification:

```
SHOW PARAMETER shared_pool_size;
```

3. **Modify a Static System-Level Parameter**

   Change a static system-level parameter, which requires a database restart:

```
ALTER SYSTEM SET db_cache_size = '2G' SCOPE=SPFILE;
```

   Restart the Database:

```
SHUTDOWN IMMEDIATE;
STARTUP;
```

   Verification:

```
SHOW PARAMETER db_cache_size;
```

Summary

In this lab, you investigated initialization parameter files, viewed initialization parameters using SQL*Plus, and modified initialization parameters at the session and system levels. This practice is essential for understanding and managing the configuration and performance of an Oracle database.

Part D: Exploring the Automatic Diagnostic Repository (ADR) and Logging DDL Statements

Objective:

To examine the structure of the Automatic Diagnostic Repository (ADR), view the alert log using both a text editor and ADRCI, and enable DDL logging to log DDL statements in the DDL log file.

Steps:

1. **Examine the Structure of the Automatic Diagnostic Repository (ADR)**

   The ADR is located in the `$ORACLE_BASE/diag/rdbms/<db_name>/<SID>/trace` directory. Navigate to this directory to examine its structure:

```
cd $ORACLE_BASE/diag/rdbms/CDBLAB/CDBLAB/trace
ls -l
```

Expected Output:

- `alert_CDBLAB.log` : The alert log file.
- Various trace files (e.g., `*.trc` files): These files contain diagnostic information for different Oracle processes.

2. **View the Alert Log Using a Text Editor**

   Use a text editor to view the alert log file:

```
vi $ORACLE_BASE/diag/rdbms/CDBLAB/CDBLAB/trace/alert_CDBLAB.log
```

   Explanation:

   - The alert log provides a chronological log of database messages and errors. It is useful for diagnosing issues with the database.

3. **View the Alert Log Using ADRCI**

   Start ADRCI and use it to view the alert log:

```
adrci
```

   In ADRCI:

```
ADRCI:> show homes
ADRCI:> set home diag/rdbms/CDBLAB/CDBLAB
ADRCI:> show alert -tail 50
```

   Explanation:

   - `show homes` : Lists the ADR homes available.
   - `set home diag/rdbms/CDBLAB/CDBLAB` : Sets the ADR home to the current database.
   - `show alert -tail 50` : Displays the last 50 lines of the alert log.

4. **Enable DDL Logging**

   Enable DDL logging by setting the appropriate initialization parameter:

```
ALTER SYSTEM SET enable_ddl_logging = TRUE SCOPE=BOTH;
```

   Explanation:

   - `enable_ddl_logging = TRUE` : Enables logging of DDL statements.

5. **Log Some DDL Statements**

   Execute some DDL statements to generate log records:

```
CREATE TABLE test_table (id NUMBER, name VARCHAR2(50));
ALTER TABLE test_table ADD (email VARCHAR2(100));
DROP TABLE test_table;
```

6. **Verify the DDL Log File**

Navigate to the directory containing the DDL log file and view its contents:

```
cd $ORACLE_BASE/diag/rdbms/CDBLAB/CDBLAB/log/ddl
ls -l
cat ddl.log
```

Expected Output:

- Entries for each DDL statement executed.

Summary

In this part of the lab, you examined the structure of the Automatic Diagnostic Repository (ADR), viewed the alert log using both a text editor and ADRCI, and enabled DDL logging to log DDL statements in the DDL log file. Understanding the ADR and logging mechanisms is crucial for diagnosing and troubleshooting Oracle database issues.