When importing only do the csv imports, don’t execute the indexes

Execute the ComboSQL 3 times, change the user name depending on your username and then run the code.

In one of the code it give u the snapshot IDs, save those and replace them in a command that has &begin\_snap\_id, &end\_snap\_id

Also replace the in\_task\_name before executing the last command

Certainly! Here’s a detailed, step-by-step guide for performing SQL Access Advisor analysis using the provided commands, organized into logical sections with proper formatting.

**Grant Required Roles and Privileges**

1. **Grant necessary roles to the user TPUSERNEW:**

sql

Copy code

GRANT DBA TO TPUSERNEW;

GRANT SYSDBA TO TPUSERNEW;

GRANT ADVISOR TO TPUSERNEW;

1. **Verify the roles granted to TPUSERNEW:**

sql

Copy code

SELECT \* FROM DBA\_ROLE\_PRIVS WHERE GRANTEE = 'TPUSERNEW';

**Import Data and Execute Script**

1. **Import your data tables into the database.**
2. **Execute the script containing the 113 queries.**

**Create a SQL Tuning Set for SQL Access Advisor**

1. **Grant necessary privileges to TPUSERNEW:**

sql

Copy code

GRANT ADMINISTER SQL TUNING SET TO TPUSERNEW;

GRANT ADMINISTER ANY SQL TUNING SET TO TPUSERNEW;

GRANT SELECT\_CATALOG\_ROLE TO TPUSERNEW;

GRANT EXECUTE ON DBMS\_WORKLOAD\_REPOSITORY TO TPUSERNEW;

GRANT CREATE ANY SNAPSHOT TO TPUSERNEW;

GRANT SELECT ON DBA\_HIST\_SNAPSHOT TO TPUSERNEW;

**Capture SQL Statements from the Automatic Workload Repository (AWR)**

1. **Create a manual snapshot of the database performance data in the AWR:**

sql

Copy code

EXEC DBMS\_WORKLOAD\_REPOSITORY.CREATE\_SNAPSHOT;

1. **Optionally, view the SQL text captured in the AWR:**

sql

Copy code

SELECT \* FROM DBA\_HIST\_SQLTEXT;

1. **Create a new SQL Tuning Set (STS) to store the SQL statements from the AWR:**

sql

Copy code

BEGIN

DBMS\_SQLSET.CREATE\_SQLSET (

sqlset\_name => 'STS\_AWR\_FOURTH',

description => 'STS to store SQL from the private SQL area'

);

END;

/

1. **Verify the created SQL Tuning Set:**

sql

Copy code

SELECT NAME, STATEMENT\_COUNT AS "SQLCNT", DESCRIPTION

FROM USER\_SQLSET;

1. **Set the timestamp format for better readability:**

sql

Copy code

ALTER SESSION SET NLS\_TIMESTAMP\_FORMAT = 'YYYY-MM-DD HH:MI';

1. **Retrieve the snapshot IDs and their corresponding begin and end times:**

sql

Copy code

COL BEGIN\_SNAP FORMAT A20

COL END\_SNAP FORMAT A20

SELECT SNAP\_ID, BEGIN\_INTERVAL\_TIME BEGIN\_SNAP, END\_INTERVAL\_TIME END\_SNAP

FROM DBA\_HIST\_SNAPSHOT

ORDER BY SNAP\_ID;

1. **Populate the 'STS\_AWR\_FOURTH' SQL Tuning Set with SQL statements from the AWR snapshots between the specified begin and end snapshot IDs:**

sql

Copy code

DECLARE

cur DBMS\_SQLTUNE.SQLSET\_CURSOR;

BEGIN

OPEN cur FOR

SELECT VALUE(P)

FROM DBMS\_SQLTUNE.SELECT\_WORKLOAD\_REPOSITORY(&begin\_snap\_id, &end\_snap\_id, NULL, NULL, NULL, NULL, NULL, NULL, NULL, 'ALL') p;

DBMS\_SQLTUNE.LOAD\_SQLSET(sqlset\_name => 'STS\_AWR\_FOURTH', populate\_cursor => cur);

END;

/

1. **Verify the SQL statements in the 'STS\_AWR\_FOURTH' SQL Tuning Set for TPUSERNEW:**

sql

Copy code

COLUMN SQL\_TEXT FORMAT A30

COLUMN SCH FORMAT A3

COLUMN ELAPSED FORMAT 999999999

SELECT SQL\_ID, PARSING\_SCHEMA\_NAME AS "SCH", SQL\_TEXT,

ELAPSED\_TIME AS "ELAPSED", BUFFER\_GETS

FROM TABLE(DBMS\_SQLTUNE.SELECT\_SQLSET('STS\_AWR\_FOURTH'))

WHERE PARSING\_SCHEMA\_NAME = 'TPUSERNEW';

**Create and Configure the SQL Access Advisor Task (put these into one sql code)**

1. **Declare variables for the task ID and task name:**

sql

Copy code

VARIABLE task\_id NUMBER;

VARIABLE task\_name VARCHAR2(255);

1. **Create a new SQL Access Advisor task:**

sql

Copy code

EXEC :task\_name := 'STS\_AWR\_FOURTH\_TASK';

EXEC DBMS\_ADVISOR.CREATE\_TASK('SQL Access Advisor', :task\_id, :task\_name);

1. **Set the analysis scope for the task:**

sql

Copy code

EXEC DBMS\_ADVISOR.SET\_TASK\_PARAMETER(:task\_name, 'ANALYSIS\_SCOPE', 'ALL');

1. **Add the 'STS\_AWR\_FOURTH' SQL Tuning Set as a reference for the task:**

sql

Copy code

EXECUTE DBMS\_ADVISOR.ADD\_STS\_REF(:task\_name, 'TPUSERNEW', 'STS\_AWR\_FOURTH');

1. **Execute the SQL Access Advisor task:**

sql

Copy code

EXECUTE DBMS\_ADVISOR.EXECUTE\_TASK(:task\_name);

**View SQL Access Advisor Task Results**

1. **Check the task status and status message:**

sql

Copy code

COL TASK\_ID FORMAT 999

COL TASK\_NAME FORMAT A25

COL STATUS\_MESSAGE FORMAT A25

SELECT TASK\_ID, TASK\_NAME, STATUS, STATUS\_MESSAGE

FROM USER\_ADVISOR\_LOG;

1. **View the tasks:**

sql

Copy code

SELECT \* FROM USER\_ADVISOR\_TASKS;

1. **View the recommendations for the task:**

sql

Copy code

SELECT \* FROM USER\_ADVISOR\_RECOMMENDATIONS WHERE TASK\_ID = :task\_id;

1. **View the actions for the recommendations:**

sql

Copy code

SELECT \* FROM USER\_ADVISOR\_ACTIONS WHERE TASK\_ID = :task\_id;

1. **Commit the changes:**

sql

Copy code

COMMIT;

1. **Execute the provided show\_recm procedure to display the recommendations and their attributes:**

CREATE OR REPLACE PROCEDURE show\_recm (in\_task\_name IN VARCHAR2) IS

CURSOR curs IS

SELECT DISTINCT action\_id, command, attr1, attr2, attr3, attr4, attr5 -- Added missing comma and attr5

FROM user\_advisor\_actions

WHERE task\_name = in\_task\_name

ORDER BY action\_id;

v\_action NUMBER;

v\_command VARCHAR2(32);

v\_attr1 VARCHAR2(4000);

v\_attr2 VARCHAR2(4000);

v\_attr3 VARCHAR2(4000);

v\_attr4 VARCHAR2(4000);

v\_attr5 VARCHAR2(4000);

BEGIN

OPEN curs;

DBMS\_OUTPUT.PUT\_LINE('=========================================');

DBMS\_OUTPUT.PUT\_LINE('Task\_name = ' || in\_task\_name); -- Concatenate strings using ||

LOOP

FETCH curs INTO

v\_action, v\_command, v\_attr1, v\_attr2, v\_attr3, v\_attr4, v\_attr5; -- Added missing comma and v\_attr5

EXIT WHEN curs%NOTFOUND;

DBMS\_OUTPUT.PUT\_LINE('Action ID: ' || v\_action); -- Concatenate strings using ||

DBMS\_OUTPUT.PUT\_LINE('Command : ' || v\_command); -- Concatenate strings using ||

DBMS\_OUTPUT.PUT\_LINE('Attr1 (name) : ' || SUBSTR(v\_attr1, 1, 30)); -- Concatenate strings using || and fix SUBSTR position

DBMS\_OUTPUT.PUT\_LINE('Attr2 (tablespace): ' || SUBSTR(v\_attr2, 1, 30)); -- Concatenate strings using || and fix SUBSTR position

DBMS\_OUTPUT.PUT\_LINE('Attr3 : ' || SUBSTR(v\_attr3, 1, 30)); -- Concatenate strings using || and fix SUBSTR position

DBMS\_OUTPUT.PUT\_LINE('Attr4 : ' || v\_attr4); -- Concatenate strings using ||

DBMS\_OUTPUT.PUT\_LINE('Attr5 : ' || v\_attr5); -- Concatenate strings using ||

DBMS\_OUTPUT.PUT\_LINE('----------------------------------------');

END LOOP;

CLOSE curs; -- Added missing semicolon

END show\_recm;

/ This guide should help you execute the SQL Access Advisor analysis efficiently, step-by-step.

**To clear:**

BEGIN

DBMS\_ADVISOR.DELETE\_TASK(task\_name => 'STS\_AWR\_FOURTH\_TASK');

END;

/

BEGIN

DBMS\_SQLSET.DROP\_SQLSET(sqlset\_name => 'STS\_AWR\_FOURTH');

END;

/