DB Security

Authentication: **Authentication is a process of verifying the credentials (username & password) of a user to login into the system.**

Authorization: **Authorization is process of verifying whether the user as permissions to perform any operation on the database.**

Data Control Language: **DCL commands are used to enforce database security in multiple users' database environment. These are two types….**

- **GRANT**
- **REVOKE**

GRANT: **Grant command is used for giving a privilege or permission for a user to perform operations on the database.**

Syntax: **GRANT <Privilege Name> on <object name> To {User} ;**

Privilege Name: **Used to granted permission to the users for some rights are ALL and SELECT.**

Object Name: **It is the name of database objects like Table, Views and Stored Procedure etc….**

User: **Used for to whom an access rights is being granted.**

REVOKE: **Revoke command removes user access rights / privileges to the database OR taking back the permission that is given to a user.**

Syntax: **Revoke <privilege name> on <object name > from {user};**

WORKING WITH DCL COMMANDS:

STEP1: CONNECT WITH SYSTEM DBA:

SQL> CONN SYSTEM / MANAGER;

CONNECTED.

STEP2: CREATE A NEW USER IN ORACLE DB:

SYNTAX:

SQL> CREATE USER <USER NAME> IDENTIFIED BY <PASSWORD>;

EX:

SQL> CREATE USER A IDENTIFIED BY A;

STEP3: CONNECT WITH USER "A":

SQL> CONN A/A;

ERROR:

ORA-01045: user A lacks CREATE SESSION privilege; logon denied

Warning: You are no longer connected to ORACLE.

NOTE: USER WAS CREATED BUT THIS USER IS DUMMY USER BECAUSE IS NOT HAVING CONNECT PERMISSION.SO PERMISSION MUST BE GIVEN TO USER "A".

STEP4: GRANTING CONNECT PERMISSIONS TO USER "A":

SQL> CONN SYSTEM / MANAGER;

CONNECTED.

SQL> GRANT CONNECT TO A;

CONNECT = TO CONNECT TO ORACLE DB.

STEP5: NOW CONNECT WITH USER " A":

SQL> CONN A/A;

CONNECTED.

STEP6: HOW TO CHANGING USER PASSWORD:

SQL> PASSWORD

Changing password for A

Old password: A

New password: 123

Retype new password: 123

Password changed

SQL> CONN A/123;

Connected.

NOTE:

1. PASSWORD CAN BE CHANGED BOTH USER & DBA.

2. PASSWORD CAN BE CHANGED BUT USERNAME CANNOT BE CHANGED.

3. USER NAME IS NOT CASE SENSITIVE BUT PASSWORD IS CASE SENSITIVE.

STEP7: NOW USER "A" CAN CREATE ITS OWN TABLES:

EX:

SQL> CREATE TABLE TEST1(SNO INT, NAME VARCHAR2(10));

Error: INSUFFICIENT PRIVILEGE.

STEP8: GRANTING CREATE TABLE PERMISSION FROM DBA:

SQL> CONN SYSTEM / MANAGER;

CONNECTED.

SQL> GRANT CREATE TABLE TO A;

GRANTED.

SQL> CONN A/123

EX:

SQL> CREATE TABLE TEST1(SNO INT, NAME VARCHAR2(10));

TABLE CREATED

STEP9: NOW USER "A" CAN INSERT ROWS INT A TABLE:

SQL> INSERT INTO TEST1 VALUES (1021,'SAI');

ERROR: INSUFFICIENT PRIVILEGE ON TABLE SPACE.

STEP10: GRANTING TABLE SPACE PERMISSION FROM DBA:

SQL> GRANT UNLIMITED TABLE SPACE TO A;

GRANTED.

SQL> CONN A/123

SQL> INSERT INTO TEST1 VALUES (1021,'SAI');

SQL> INSERT INTO TEST1 VALUES (1022,'JONES');

SQL> COMMIT;

SQL> UPDATE TEST1 SET NAME='MILLER' WHERE SNO=1022;

SQL> COMMIT;

SQL> DELETE FROM TEST1 WHERE SNO=1022;

SQL> COMMIT;

SQL> SELECT * FROM TEST1;

PRIVILEGES: **PRIVILEGE IS RIGHT / PERMISSION GIVEN TO THE USERS.ALL DATABASES ARE HAVING TWO TYPES OF PRIVILEGES.**

i) SYSTEM PRIVILEGES

ii) OBJECT PRIVILEGES

i) SYSTEM PRIVILEGES:

> SYSTEM PRIVILEGES ARE GIVEN BY DBA.SUCH AS CREATE SYNONYM, CREATE VIEW, CREATE MATERIALIZED VIEW, CREATE INDEX ……etc.

SYNTAX:

SQL> GRANT <SYSTEM PRIVILEGE> TO <USER1>;

EX:

SQL> CONN SYSTEM/MANAGER;

Connected.

SQL> GRANT CREATE SYNONYM, CREATE VIEW TO A;

Grant succeeded.

SQL> CONN A/123;

Connected.

SQL> CREATE SYNONYM SYN1 FOR TEST1;

Synonym created.

SQL> CREATE VIEW V1 AS SELECT * FROM TEST1;

View created.

NOTE:IF WE WANT TO VIEW SYSTEM PRIVILEGES RELATED TO USER IN ORACLE DB_THEN WE FOLLOW THE FOLLOWING DATADICTIONARY IS " SESSION_PRIVS "SYNTAX:

SQL> SELECT * FROM SESSION_PRIVS;

ii) OBJECT PRIVILEGES: OBJECT PRIVILEGES ARE USED TO USERS TO ALLOWED TO PERFORM SOME OPERATIONS ON OBJECT.THESE PRIVILEGES ARE GIVEN BY EITHER DBA (OR) DB DEVELOPER. ORACLE HAVING THE FOLLOWING OBJECT PRIVILEGES ARE INSERT, UPDATE, DELETE, SELECT.

> THESE FOUR OBJECT PRIVILEGES ARE REPRESENTED BY USING "ALL" KEYWORD.

SYNTAX:

GRANT <OBJECT PRIVILEGES> ON <OBJECT NAME> TO USER1;

EX:

SQL> CONN A/123;

SQL> SELECT * FROM DEPT;

(OR)

SQL> SELECT * FROM SYSTEM.DEPT;

ERROR at line 1:

ORA-00942: table or view does not exist

NOTE:USER "A" CANNOT SELECT / ACCESS DATA FROM DEPT TABLE BECAUSE USER "A" IS NOT HAVING PERMISSION OF ACCESSING DATA FROM DEPT.SO THAT WE WANT TO TAKE GRANT SELECT PERMISSION FROM DBA.

EX:

SQL> CONN SYSTEM/MANAGER

Connected.

SQL> GRANT SELECT ON DEPT TO A;

Grant succeeded.

SQL> CONN A/123;

Connected.

SQL> SELECT * FROM SYSTEM.DEPT; -----ALLOWED

SQL> INSERT INTO SYSTEM.DEPT VALUES (50,'SAP','INDIA'); ---NOT ALLOW

SQL> UPDATE SYSTEM.DEPT SET LOC='HYD' WHERE DEPTNO=50; --- NOT ALLOW

SQL> DELETE FROM SYSTEM.DEPT WHERE DEPTNO=50; ---NOT ALLOW

NOTE: USER "A" CANNOT PERFORM DML OPERATIONS ON DEPT TABLE BECAUSE USER "A" DID NOT HAVE PERMISSION FROM DBA.

EX:

SQL> CONN SYSTEM/MANAGER

SQL> GRANT INSERT, UPDATE, DELETE ON DEPT TO A;

SQL> CONN A/123;

SQL> INSERT INTO SYSTEM.DEPT VALUES (50,'SAP','INDIA'); --- ALLOW

SQL> UPDATE SYSTEM.DEPT SET LOC='HYD' WHERE DEPTNO=50; --- ALLOW

SQL> DELETE FROM SYSTEM.DEPT WHERE DEPTNO=50; --- ALLOW

<u>NOTE:</u>TO VIEW THE INFORMATION ABOUT PRIVILEGE AND ALSO GRANTER THEN WE USE FOLLOWING DATADICTIONARY " USER_TAB_PRIVS_MADE ".

EX:

SQL> CONN SYSTEM/MANAGER;

SQL> DESC USER_TAB_PRIVS_MADE;

SQL> SELECT GRANTEE, TABLE_NAME, GRANTOR, PRIVILEGE FROM USER_TAB_PRIVS_MADE;

GRANTEE	TABLE_NAME	GRANTOR	PRIVILEGE
A	DEPT	SYSTEM	DELETE

NOTE:TO VIEW THE INFORMATION ABOUT PRIVILEGE AND ALSO WHO RECEIVIED PERMISSION(GRANTEE) THEN WE USE FOLLOWING DATADICTIONARY " USER_TAB_PRIVS_RECD".

EX:

```
SQL> CONN A/123;

SQL> DESC USER_TAB_PRIVS_RECD;

SQL> SELECT GRANTOR, PRIVILEGE, TABLE_NAME FROM
USER_TAB_PRIVS_RECD;
```

GRANTOR	PRIVILEGE	TABLE_NAME
SYSTEM	UPDATE	DEPT

EX:

```
SQL> CONN SYSTEM/MANAGER

SQL> REVOKE ALL ON DEPT FROM A;
```

> DBA(SYSTEM) CANCELLED ALL PERMISSIONS OF USER " A ".

WITH GRANT OPTION:

WHEN A USER RECEIVING PERMISSIONS "WITH GRANT OPTION" THEN THAT USER ALSO ALLOW TO GIVE OBJECT PRIVILEGE TO ANOTHER USER.EX:

```
SQL> CONN SYSTEM/MANAGER

SQL> GRANT SELECT ON DEPT TO U1;
```

SQL> CONN U1/U1;

SQL> GRANT SELECT ON SYSTEM.DEPT TO U2;

ERROR:

INSUFFICIENT PRIVIKEGES TO GRANT TO U2.

> TO OVERCOME THE ABOVE PROBLEM THEN USE "WITH GRANT OPTION " BY SYSTEM.

EX:

SQL> CONN SYSTEM/MANAGER

SQL> GRANT SELECT ON DEPT TO U1 WITH GRANT OPTION;

SQL> CONN U1/U1;

SQL> GRANT SELECT ON SYSTEM.DEPT TO U2;

Grant succeeded.

SQL> CONN U2/U2;

SQL> SELECT * FROM SYSTEM.DEPT; -----ALLOWED

CREATE SESSION:BY USING CREATE SESSION SYSTEM PRIVILEGE ONLY USER ARE ALLOWED TO CONNECT TO ORACLE DB OTHERWISE ORACLE SERVER RETURNS AN ERROR.

EX:

SQL> CONN SYSTEM/MANAGER

SQL> CREATE USER U3 IDENTIFIED BY U3;

User created.

SQL> CONN U3/U3;

ERROR:

ORA-01045: user U3 lacks CREATE SESSION privilege; logon denied

> TO OVERCOME THE ABOVE "CREATE SESSION" PRIVILEGE PROBLEM THEN WE FOLLOW THE FOLLOWING SOLUTION IS,

SQL> CONN SYSTEM/MANAGER

SQL> GRANT CREATE SESSION TO U3;

SQL> CONN U3/U3;

CONNECTED.

SQL> CREATE TABLE TEST11(SNO INT, NAME VARCHAR2(10));

ERROR at line 1:

ORA-01031: insufficient privileges

SQL> CONN SYSTEM/MANAGER

SQL> GRANT CREATE TABLE TO U3;

SQL> GRANT UNLIMITED TABLESPACE TO U3;

SQL> CONN U3/U3;

SQL> CREATE TABLE TEST11(SNO INT, NAME VARCHAR2(10));

TABLE CREATED.

SQL> INSERT INTO TEST11 VALUES(1021,'SAI');

SQL> UPDATE TEST11 SET NAME='SAI KUMAR' WHERE SNO=1021;

SQL> DELETE FROM TEST11 WHERE SNO=1021;

SQL> SELECT * FROM TEST11;

NOTE:TO VIEW ALL USERS DETAILES IN ORACLE THEN WE FOLLOW THE FOLLOWING DATADICTIONARY IS "ALL_USERS".

EX:

SQL> DESC ALL_USERS;

SQL> SELECT USERNAME FROM ALL_USERS;

SYNTAX TO DROP A USER:

SQL> DROP USER <USER NAME> CASCADE;

EX:

SQL> DROP USER A CASCADE;

ROLE: ROLE IS COLLECTION OF SYSTEM / OBJECT PRIVILEGES AND CREATED BY DBA.

WHY WE NEED TO CREATE ROLE:

IN REALTIME ENVIRONMENT NO. OF USERS WORKING ON SAME PROJECT IN THIS SOME GROUP OF USERS REQUIRES COMMON SET OF SYSTEM PRIVILEGES OR OBJECT PRIVILEGES AT THIS TIME DBA CREATING ROLE AND ASSIGNING THAT ROLE TO THE NO. OF USERS.

STEPS TO CREATE A ROLE:

STEP1: CREATE A ROLE:

SYNTAX:

CREATE ROLE <ROLE NAME>;

STEP2: ASSIGN SYSTEM / OBJECT PRIVILEGES TO A ROLE:

SYNTAX:

GRANT SYSTEM PRIVILEGES / OBJECT PRIVILEGES TO <ROLE NAME>;

STEP3: ASSIGN ROLE TO THE NO. OF USERS:

SYNTAX:

GRANT ROLENAME TO USER1, USER2, USER3, ……;

EX:

SQL> CONN SYSTEM/MANAGER

SQL> CREATE ROLE R1;

Role created.

SQL> GRANT CREATE SYNONYM TO R1;

Grant succeeded.

SQL> GRANT R1 TO U1, U2;

Grant succeeded.

NOTE1:TO VIEW SYSTEM PRIVILEGES RELATED TO ROLE THEN WE ARE USING THE FOLLOWING DATADICTIONARY IS " ROLE_SYS_PRIVS ".

EX:

SQL> DESC ROLE_SYS_PRIVS;

SQL> SELECT ROLE, PRIVILEGE FROM ROLE_SYS_PRIVS WHERE ROLE='R1';

ROLE	PRIVILEGE
R1	CREATE SYNONYM

NOTE2:TO VIEW OBJECT PRIVILEGES RELATED TO ROLE THEN WE USE THE FOLLOWING DATADICTIONARY IS " ROLE_TAB_PRIVS "

EX:

SQL> DESC ROLE_TAB_PRIVS;

SQL> SELECT ROLE, PRIVILEGE, TABLE_NAME FROM ROLE_TAB_PRIVS WHERE ROLE='R1';

ROLE PRIVILEGE TABLE_NAME

--------- ---------- ----------

R1 SELECT DEPT

<u>NOTE3:</u>TO VIEW ROLES GRANTED TO A USER IN ORACLE DB THEN USE THE FOLLOWING DATADICTIONARY IS "USER_ROLE_PRIVS".

EX:

SQL> CONN U1/U1;

SQL> DESC USER_ROLE_PRIVS;

SQL> SELECT USERNAME, GRANTED_ROLE FROM USER_ROLE_PRIVS;

USERNAME GRANTED_ROLE

--------- ----------

U1 R1

NOTE: "WITH GRANT OPTION " DOESN'T WORK ON ROLE.

EX:

SQL> CONN SYSTEM/MANAGER

SQL> GRANT INSERT ON DEPT TO R1 WITH GRANT OPTION;

ERROR at line 1:

ORA-01926: cannot GRANT to a role WITH GRANT OPTION

<u>SYNTAX TO DROP A ROLE:</u>

SQL> DROP ROLE <ROLE NAME>;

EX:

SQL> DROP ROLE R1;

GRANTING PERMISSIONS TO DIFFERENT LEVELS:

SYNTAX:

GRANT <SYSTEM PRIVILEGES> TO <USER> / <ROLE> / <PUBLIC>;

&

GRANT <OBJECT PRIVILEGES> ON <OBJECT NAME> TO <USER> / <ROLE> / <PUBLIC>;

REVOKING PERMISSIONS TO DIFFERENT LEVELS:

SYNTAX:

REVOKE <SYSTEM PRIVILEGES> FROM <USER> / <ROLE> / <PUBLIC>;

&

REVOKE <OBJECT PRIVILEGES> ON <OBJECT NAME> FROM <USER> / <ROLE> / <PUBLIC>;