Database Management System

EXPERIMENT 15 CONTROLLING USER ACCESS

NAME : PRATHESHA J

ROLL NO : 241001172

1. What privilege should a user be given to log on to the Oracle Server? Is this a system or an object privilege?

Answer: CREATE SESSION - This is a system privilege.

2. What privilege should a user be given to create tables?

Answer: CREATE TABLE - This is a system privilege.

3. If you create a table, who can pass along privileges to other users on your table?

Answer: The table owner or users who have been granted the privilege WITH GRANT OPTION.

4. You are the DBA. You are creating many users who require the same system privileges. What should you use to make your job easier?

Answer: Create a ROLE and grant the system privileges to the role, then grant the role to multiple users.

5. What command do you use to change your password?

Answer: ALTER USER username IDENTIFIED BY new_password;

6. Grant another user access to your DEPARTMENTS table. Have the user grant you query access to his or her DEPARTMENTS table.

As DBA or table owner:

```
-- Grant SELECT privilege on DEPARTMENTS to another user
GRANT SELECT ON departments TO user2;
```

Expected Output: `Grant`

```
succeeded.
```

As the other user (user2), grant SELECT back:

```
GRANT SELECT ON departments TO original_user;
```

Expected Output: `Grant`

```
succeeded.
```

7. Query all the rows in your DEPARTMENTS table.

```
SELECT * FROM departments;
```

Expected Output:

```
DEPARTMENT_ID DEPARTMENT_NAME                MANAGER_ID
LOCATION_ID
------------- ------------------------------ ---------- ----
------
           10 Administration                        200
```

```
1700
           20 Marketing                                      201
1800
           30 Purchasing                                     114
1700
           40 Human Resources                                203
2400
           50 Shipping                                       121
1500
           60 IT                                             103
1400
           70 Public Relations                               204
2700
           80 Sales                                          145
2500
           90 Executive                                      100
1700
          100 Finance                                        108
1700
          110 Accounting                                     205
1700
          120 Treasury                                        -
1700
          130 Corporate Tax                                   -
1700
          140 Control And Credit                              -
1700
          150 Shareholder Services                            -
1700
          160 Benefits                                        -
1700
```

```
          170 Manufacturing                           -
1700
          180 Construction                            -
1700
          190 Contracting                             -
1700
          200 Operations                              -
1700
          210 IT Support                              -
1700
          220 NOC                                     -
1700
          230 IT Helpdesk                             -
1700
          240 Government Sales                        -
1700
          250 Retail Sales                            -
1700
          260 Recruiting                              -
1700
          270 Payroll                                 -
1700
```

8. Add new rows to DEPARTMENTS table Team 1:

```
INSERT INTO departments (department_id, department_name,
location_id)
VALUES (500, 'Education', 1700);
```

Expected Output: 1 row

```
created.
```

Team 2:

```
INSERT INTO departments (department_id, department_name,
location_id)
VALUES (510, 'Human Resources', 1700);
```

Expected Output: 1 row

```
created.
```

Query other team's table (if privileges granted):

```
-- If granted SELECT privilege on the other team's table
SELECT * FROM team2.departments WHERE department_id = 510;
```

9. Query USER_TABLES data dictionary

```
SELECT table_name, tablespace_name, num_rows
FROM user_tables
WHERE table_name = 'DEPARTMENTS';
```

Expected Output:

```
TABLE_NAME                     TABLESPACE_NAME
NUM_ROWS
------------------------------ -----------------------------
- ----------
```

```
DEPARTMENTS                     USERS
27
```

10. Revoke SELECT privilege from the other team

```
REVOKE SELECT ON departments FROM user2;
```

Expected Output: Revoke

```
succeeded.
```

11. Remove inserted row and save changes

```
DELETE FROM departments WHERE department_id = 500;
COMMIT;
```

Expected Output:

```
1 row deleted.

Commit complete.
```

Complete User Management Examples:

1. Create a new user

```
-- As DBA
CREATE USER new_user IDENTIFIED BY password123;
```

Expected Output: `User`

```
created.
```

2. Grant system privileges to user

```
GRANT CREATE SESSION, CREATE TABLE, CREATE VIEW TO new_user;
```

Expected Output:

```
Grant succeeded.
```

3. Grant object privileges with GRANT OPTION

```
GRANT SELECT, INSERT, UPDATE ON employees TO new_user WITH
GRANT OPTION;
```

Expected Output: `Grant`

```
succeeded.
```

4. Create and use roles

```
-- Create role
CREATE ROLE manager_role;

-- Grant privileges to role
GRANT CREATE TABLE, CREATE VIEW, CREATE PROCEDURE TO
manager_role;

-- Grant role to user
```

```
GRANT manager_role TO new_user;
```

Expected Output:

```
Role created.

Grant succeeded.

Grant succeeded.
```

5. View user privileges

```
-- View system privileges
SELECT * FROM user_sys_privs;

-- View role privileges
SELECT * FROM user_role_privs;

-- View object privileges
SELECT * FROM user_tab_privs;
```

Expected Output (sample):

```
USERNAME                       PRIVILEGE
ADM
------------------------------ -----------------------------
----------- ---
NEW_USER                       CREATE SESSION
NO
NEW_USER                       CREATE TABLE
NO
NEW_USER                       CREATE VIEW
```

```
NO
```

6. Change user password

```
ALTER USER new_user IDENTIFIED BY newpassword456;
```

Expected Output: User

```
altered.
```

7. Grant privileges to PUBLIC

```
GRANT SELECT ON countries TO PUBLIC;
```

Expected Output: Grant

```
succeeded.
```

8. Revoke privileges

```
REVOKE INSERT ON employees FROM new_user;
```

Expected Output: Revoke

```
succeeded.
```

9. Drop user

```
-- If user has objects
DROP USER new_user CASCADE;
```

```
-- If user has no objects
DROP USER new_user;
```

Expected Output: User

dropped.

Security Best Practices Examples:

1. Create a read-only role

```
CREATE ROLE read_only_role;
GRANT CREATE SESSION TO read_only_role;
GRANT SELECT ON employees TO read_only_role; GRANT SELECT ON
departments TO read_only_role;
GRANT read_only_role TO report_user;
```

2. Create an application role

```
CREATE ROLE hr_app_role;
GRANT SELECT, INSERT, UPDATE ON employees TO hr_app_role;
GRANT SELECT, INSERT, UPDATE ON departments TO hr_app_role;
GRANT SELECT ON jobs TO hr_app_role;
GRANT SELECT ON locations TO hr_app_role;
```

3. View all privileges in the system

```
-- View all users
SELECT username, account_status, created FROM dba_users;
```

```
-- View all roles
SELECT role, password_required FROM dba_roles;

-- View role privileges
SELECT grantee, granted_role FROM dba_role_privs;
```

Key Security Concepts Demonstrated:

1. **System Privileges** - Database operations (CREATE TABLE, CREATE SESSION, etc.)
2. **Object Privileges** - Operations on specific objects (SELECT, INSERT, UPDATE, DELETE)
3. **Roles** - Named groups of privileges for easier management
4. **GRANT** - Gives privileges to users or roles
5. **REVOKE** - Removes privileges from users or roles
6. **WITH GRANT OPTION** - Allows grantee to grant privileges to others
7. **PUBLIC** - Special group that includes all users
8. **Data Dictionary Views** - USER_TAB_PRIVS, USER_SYS_PRIVS, USER_ROLE_PRIVS