UNIVERSITY OF TECHNOLOGY, JAMAICA

Lab Manual

FACULTY:	Engineering & Computing (FENC)
SCHOOL/DEPT:	School of Computing & Information Technology
COURSE OF STUDY:	Bachelor of Science in Computing
YEAR:	Four (3)
MODULE TITLE:	**Database Administration**

Topic: Administering Roles, Profiles, Procedures, Encryption

Objectives:

1. Decentralized Administration with
    - Admin Option
    - Grant Option
2. Using Profiles for Password Management

Granting System Privileges and Roles

You can grant system privileges and roles to other users and roles using the `GRANT` statement. The following privileges are required:

- To grant a system privilege, you must have been granted the system privilege with the `ADMIN OPTION` or have been granted the `GRANT ANY PRIVILEGE` system privilege.
- To grant a role, you must have been granted the role with the `ADMIN OPTION` or have been granted the `GRANT ANY ROLE` system privilege.

Granting the ADMIN OPTION

A user or role that is granted a privilege or role, which specifies the `WITH ADMIN OPTION` clause, has several expanded capabilities:

- The grantee can grant or revoke the system privilege or role to or from any user or other role in the database. Users cannot revoke a role from themselves.
- The grantee can further grant the system privilege or role with the `ADMIN OPTION`.
- The grantee of a role can alter or drop the role.

Example: `GRANT new_dba TO michael WITH ADMIN OPTION;`

Note: The user `michael` cannot only use all of the privileges implicit in the `new_dba` role, but can also grant, revoke, and drop the `new_dba` role as deemed necessary.
When a user creates a role, the role is automatically granted to the creator with the `ADMIN OPTION`

Please Note that the SQL solution for all exercises should be provided.

Exercise 1:

1. Use Labuser to create a new user "adminop_<yourid>". Grant all necessary privileges to connect and create tables.

   Observation:

   - Which privileges did you grant?
   - Which system tables showed that you have the privileges?

2. Allow your lab partner to log in with the new user "adminop_<yourid>" and attempt creating a **new** user "adminop_1_<yourid>". (Do not grant any privileges to create a user)

   Observation:
   - What did you observe?
   - Please write all error messages and codes along with your observation. Research the error codes you observed.

3. Using Labuser Grant the user "adminop_<yourid>" the admin option to create users.
   Observation:
   What did you observe?

4. Ask your lab partner to try re-creating the user "adminop_1_<yourid>" using the "adminop_<yourid>" user. Ensure that the create user privilege with admin option is used given.
   Observation:
   What did you observe that is different from task 2?

5. Allow your Lab Partner to use "adminop_1_<yourid>" to create "adminop_2_<yourid>". Ensure that the create user privilege with admin option is used given.

   Observation:
   What did you observe?

6. Use Labuser to revoke the privileges of "adminop_1_<yourid>".
   Observation:
   Was adminop_2_<yourid> removed from your database ? Which system tables did you use to confirm this? State what is cascading revoke and does it apply now? Are users who were created dropped?

7. Use "adminop_2_<yourid>" to create "adminop_3_<yourid>" with connect privileges.
   Observation:
   Was adminop_3_<yourid> created ? Which system tables did you use to confirm this? Does cascading revoke apply? Can adminop_2_<yourid> still create users although the user that gave it the privilege (adminop_1_<yourid>) had its privilege revoked?

Granting the GRANT OPTION

Specify `WITH GRANT OPTION` to enable the grantee to grant the object privileges to other users and roles. The user whose schema contains an object is automatically granted all associated object privileges with the `GRANT OPTION`. This special privilege allows the grantee several expanded privileges:

- The grantee can grant the object privilege to any users in the database, with or without the `GRANT OPTION`, and to any role in the database.
- If both of the following conditions are true, then the grantee can create views on the table and grant the corresponding privileges on the views to any user or role in the database.
    - The grantee receives object privileges for the table with the `GRANT OPTION`.
    - The grantee has the `CREATE VIEW` or `CREATE ANY VIEW` system privilege.

Exercise 2:

1. Use Labuser to create a new user "grantop_<yourid>". Grant all necessary privileges to connect and create tables.
   Observation:
   What did you observe? State the system tables for confirmation.

2. Create the sales_<your_id> table in the labuser schema with 5 records.
   Observation:
   What did you observe?

3. Grant the SELECT,UPDATE, DELETE and INSERT permission on the sales table in labuser to studentA with grant option.
   Observation:
   Which system tables show the privileges that were assigned? If there are

multiple ones, list them and tell the difference between the tables.

4. Allow your lab partner to log on as **studentA**.
   Observation: Can they access the sales table in the labuser schema?

5. Allow your lab partner to log on as grantop.
   Observation: Can they access the sales table in the labuser schema?

6. Use **studentA** to grant all the privileges to the sales table in the labuser schema to grantop.
   Observation: Can grantop access the sales table in the labuser schema?

7. Use the labuser to revoke the permissions from studentA.
   Observation: Can studentA access the sales table in the labuser schema?

8. Use grantop to access the sales table in the labuser scheme.
   Observation: Can grantop access the sales table in the labuser schema? Does Cascading revoke apply here and why?

Profile

You can set up limits on the system resources used by setting up profiles with defined limits on resources. Profiles are very useful in large, complex organizations with many users. It allows you to regulate the amount of resources used by each database user by creating and assigning profiles to users.

Syntax

```
CREATE PROFILE <profile_name> LIMIT
<profile_item_name> <value>
<profile_item_name> <value>;
```

Exercise 3

1. Login in as labuser and create the below profile, search the system table to find where this profile is stored.

```
CREATE PROFILE studentProf LIMIT
PASSWORD_LIFE_TIME 60
PASSWORD_GRACE_TIME 10
PASSWORD_REUSE_TIME 2
PASSWORD_REUSE_MAX 2
FAILED_LOGIN_ATTEMPTS 2
PASSWORD_LOCK_TIME 2
CPU_PER_CALL 3000
PRIVATE_SGA 500K
LOGICAL_READS_PER_CALL 1000;
```

2. Assign **studentB** to the **studentProf** profile

3. Query the System table to show which profile **studentB** is on.

4. Attempt to log on as student 3 times with the incorrect password. Then the 4th time with the right password
   Observation: What happens?

5. Restore normal service to **studentB** using labuser.

6. Change the password twice for student using labuser.
   (Use account expire. Use the same new password each time)