Oracle GRANT Statement

Summary: In this lab, you will learn how to use the Oracle `GRANT` statement to give privileges to a specific user.

The overview of Oracle privileges

After [creating a user], you need to decide which actions the user can do in the Oracle database.

In the `CREATE USER` lab, we used the `GRANT` statement to provide the user `john` the `CREATE SESSION` system privilege to enable the user to log in the Oracle database.

```
GRANT CREATE SESSION TO john;
```

What is a privilege?

By definition, a privilege is a right to execute an SQL statement or a right to access an object of another user.

Oracle defines two main types of privileges: system privileges and object privileges

System privileges

System privileges determine what a user can do in the database. They mainly allow a user to add or modify schema objects in the database like creating tables, creating views, and removing tablespaces.

The most important system privileges are:

- `CREATE SESSION`
- `CREATE TABLE`
- `CREATE VIEW`
- `CREATE PROCEDURE`
- `SYSDBA`
- `SYSOPER`

Object privileges

Object privileges decide how a user can access the data in the database. The object privileges apply to rows in tables or views.

Here are some common object privileges:

- `INSERT`
- `UPDATE`
- `DELETE`
- `INDEX`
- `EXECUTE`

To grant one or more privileges to a user, you use the `GRANT` statement

Oracle `GRANT` statement

The `GRANT` statement assigns one or more privileges to a specific user. The following illustrates the basic syntax of the `GRANT` statement:

```
GRANT {system_privileges | object_privileges }
TO user
[WITH ADMIN OPTION]
```

In this syntax:

First, specify the system or object privileges that you want to assign to a user after the `GRANT` keyword. If you assign more than one privilege, you use a comma-separated list of privileges.

Second, specify the user that receives the privileges after the `TO` keyword.

Third, optionally use the `WITH ADMIN OPTION` if you want the user to be able to perform the following:

- Grant / revoke the privilege to/from another user.
- Alter the privilege to change the authorization needed to access it.
- Drop the privilege.

The user who receives the privileges via the `GRANT` statement is also known as a *grantee*.

Note that the `GRANT` statement also works with roles, which we will cover in the subsequent lab.

Oracle `GRANT` statement examples

Let's practice with the `GRANT` statement to get a better understanding.

1) Use Oracle `GRANT` to grant system and object privileges to a user example

In this lab, we will launch two SQL Developer sessions, one for the user `sys` that will grant privileges and another for the user `john`.

First, launch SQL Developer and log in to the Oracle database using the user `john`. Note that we assigned the user `john` the `CREATE SESSION` system privilege, so it should be able to log in.

In case you're not following the `CREATE USER` lab, you can create a user `john` and grant the `CREATE SESSION` system privilege by using the following statements:

```
CREATE USER john IDENTIFIED BY abcd1234;

GRANT CREATE SESSION TO john;
```

Important: Drop table `T1` as sys user before proceeding.

```
DROP TABLE t1;
```

Second, use the user `john` to log in to the Oracle Database and `create a new table`:

```
CREATE TABLE t1(id NUMBER PRIMARY KEY);
```

Oracle issued the following error:

```
ORA-01031: insufficient privileges
```

To allow the user `john` to create the table, you need to grant the `CREATE TABLE` system privilege to the user as shown in the following statement:

```
GRANT CREATE TABLE TO john;
```

Now, the user `john` can `create a new table`:

```
CREATE TABLE t1(id NUMBER PRIMARY KEY);
```

The following statement shows the privileges of the current user:

```
SELECT * FROM session_privs;
```

Here are the privileges of the user `john`:

```
PRIVILEGE

CREATE SESSION
CREATE TABLE
```

Third, use the user `john` to [insert a new row] into the `t1` table:

```
INSERT INTO t1(id) VALUES(10);
```

Oracle issued the following error:

```
ORA-01950: no privileges on tablespace 'USERS'
```

This is because the user `john` has a quota of zero on the `USERS` tablespace.

To fix this, you use the `ALTER USER` command to change the quota of the user john on the `USERS` tablespace:

```
ALTER USER john QUOTA UNLIMITED ON USERS;
```

Now, the user `john` should be able to `insert a row` into the `t1` table:

```
INSERT INTO t1(id) VALUES(10);
```

And [query data] from the `t1` table as well:

```
SELECT * FROM t1;
```

Here is the output:

```
        ID

        10
```

2) Use Oracle `GRANT` to assign privileges WITH ADMIN OPTION example

First, create a new user called `jack` and grant the user the `CREATE SESSION` so that the user can log in:

```
CREATE USER jack IDENTIFIED BY abcd1234
    QUOTA UNLIMITED ON users;

GRANT CREATE SESSION TO jack;
```

Second, grant the `CREATE TABLE` system privilege to `john`, but this time, use the `WITH ADMIN OPTION`:

```
GRANT CREATE TABLE TO john WITH ADMIN OPTION;
```

Now, the user `john` can grant the `CREATE TABLE` system privilege to another user e.g. `jack`.

Third, login as `john` and grant the `CREATE TABLE` system privilege to `jack`:

```
GRANT CREATE TABLE TO jack;
```

Finally, login as `jack` and create a new table:

```
CREATE TABLE t2(id NUMBER PRIMARY KEY);
```

The user `jack` can create the table.

3) Using Oracle GRANT to assign privileges which has `ANY` option example

Some system privileges have the keyword `ANY` that enables a user to perform the corresponding action on any objects in the database.

For example, `SELECT ANY TABLE` allows a user to select data from any table in any schema in the database.

Consider the following example.

First, log in as `jack` and [select] the data from `t1` table in the `john`'s schema:

```
SELECT * FROM john.t1;
```

Oracle issued the following error:

```
ORA-00942: table or view does not exist
```

Second, login as `sys` and grant the `SELECT ANY TABLE` system privilege to `jack`:

```
GRANT SELECT ANY TABLE TO jack;
```

Third, from the session of `john`, execute the `SELECT` statement:

```
SELECT * FROM john.t1;
```

Here is the output:

```
        ID

        10
```

Now the user `jack` can select data from any table in any schema in the Oracle database.

4) Using Oracle GRANT to grant object privileges to a user example

First, launch the first SQL Developer session, log in as `sys` user and `create a new table` named `t2`:

```
DROP TABLE t2;
```

```
CREATE TABLE t2(id INT);
```

Second, insert some values into the `t2` table:

```
INSERT INTO t2(id) VALUES(1);
INSERT INTO t2(id) VALUES(2);
```

Third, launch the second SQL Developer session, log in as `john`, and query data from the `sys.t2` table:

```
SELECT * FROM sys.t2;
```

Oracle issued the following error:

```
ORA-00942: table or view does not exist
```

This is because the user `john` does not have the privilege to query data from the `sys.t2` table.

Fourth, go back to the first SQL Developer session and grant the `SELECT` object privilege on `sys.t2` to `john`:

```
GRANT SELECT ON sys.t2 TO john;
```

Fifth, go to the second session SQL Developer, and query data from the `sys.t2` table:

```
SELECT * FROM sys.t2
```

Now, `john` should be able to query data from the `sys.t2` table.

Sixth, try to insert some rows into the `sys.t2` table:

```
INSERT INTO sys.t2(id) VALUES(3)
```

Oracle issued the following error:

```
ORA-01031: insufficient privileges
```

To allow `john` to insert and update data in the `sys.t2` table, you need to grant the `[INSERT]` and `[UPDATE]` object privilege to `john`:

```
GRANT INSERT, UPDATE ON sys.t2 TO john;
```

Now, `john` should be able to insert and update data in the `sys.t2` table.

In this lab, you have learned how to use the Oracle `GRANT` statement to assign system and object privileges to a specific user.