Create, Manage, and control Database Users

Dr. Rudra Pratap Deb Nath [*]

Associate Professor
University of Chittagong, Chittagong

July 9, 2025

[*]Corresponding authors email: rudra@cu.ac.bd

1 Creating users

The '`create user user_name identified by password ;`' command is used to create a user in a database. The user is assigned a password, which is stored encrypted in the database.

For example following statement create a user TEST whose password is TEST.

```
SQL> create user TEST identified by  TEST;
```

In the current version, `Oracle` recommends you to prefix a user name with 'c##'. Therefore, the above statement may arise `''ORA-65096: invalid common user or role name"`. Execute the following statement to avoid this restriction.

```
SQL> alter session set "_ORACLE_SCRIPT"=true;
```

You can also assign other parameters when creating a user.

```
SQL> create user TEST
     identified by TEST
     default tablespace USERS
     temporary tablespace TEMP
     QUOTA 100M on USERS
     QUOTA unlimited on SYSTEM
     password expire
     account unlock;
```

- The password specified is not case sensitive.
- If the 'default tablespace' clause is not assigned, SYSTEM tablespace is assigned as the default tablespace. Use "select tablespace_name from user_tablespaces" to see the available tablespaces in your database.
- Although the default and temporary tablespace are specified, TEST does not initially have any space quota on USERS (or any tablespace). 'QUOTA clause' is userd to allocate quotas on USERS and SYSTEM.
- Unlimited specifies that the quota on the tablespace is not limited.
- Password expire specifies that the user will be prompted for a new password at the fist login.
- Account unlock is default. To initially lock 'account lock' is used.

- The profile clause specify the profile to be assigned. To add a profile, use profile profile_name.

** TEST can connect to the database only if (s)he has the ‘create session’ privilege. This can be done by the following command.

```
SQL> Grant create session to TEST;
```

Now the TEST user can connect to the database using the following command.

```
SQL> connect test/test;
```

It will prompt you to type new password. Now you can change different parameter of the user using ‘alter user’ command. For example,

```
SQL> alter user TEST identified by TEST123;
```

Dropping a table: The `‘drop user TEST cascade;’` command is used to drop a user. If cascade keyword is specified, the `Oracle` drops all the objects owned by the user and then drop the user.

```
SQL> drop user TEST cascade;
```

Data dictionary tables: DBA_USERS, All_USERS, V$SESSION, DBA_TS_QUOTAS

1.1 Profile

Two categories of profile: resource management and password management.

Resource Management: `Oracle` lets you control the following types of resource usage through profiles:

- Concurrent sessions per user
- Elapsed and idle time connected to the database.
- CPU time used.
- Private `SQL` and `PL/SQL` area used in the SGA.
- Logical reads performed.
- Amount of private SGA space used in shared server configuration.

** To enable resource limit restrictions with profiles, first enable them in the database by setting the initialization parameter resource_limit to true.

```
SQL> Alter system set resource_limit = true scope=
   both;
```

To assign resource limits to a profile, '`create profile`' or '`alter profile`' is used. Following parameters can be used to control resources.

SESSIONS_PER_USER : Limits the number of concurrent user sessions.

```
SQL> create/alter profile admin_profile
LIMIT sessions_per_user 2;
```

CPU_PER_SESSION : Limits the amount of CPU time that can be consumed in any session established by a user with a profile. The specified value is in hundredths of a second. When the limit is reached, the current statement fails, the transaction is automatically rolled back, and an exception is raised.

```
SQL> create/alter profile admin_profile
LIMIT cpu_per_sessions 3000;
```

CPU_PER_CALL : Limits the amount of CPU time a single `SQL` statement can use. It also measure in hundredths of a second.

```
SQL> create/alter profile admin_profile
LIMIT cpu_per_call 3000;
```

LOGICAL_READS_PER_SESSION : Limits the number of data blocks read in a session, including the blocks read from memory and from physical reads.

```
SQL> create/alter profile admin_profile
LIMIT logical_reads_per_session 100000;
```

LOGICAL_READS_PER_CALL : Limits the number of data blocks read by a single `SQL` statement.

```
SQL> create/alter profile admin_profile
LIMIT logical_reads_per_call 2500;
```

PRIVATE_SGA : Limits the amount of sapce allocated in the SGA for private areas per session. This limit does not apply to dedicated architecture connections.

```
SQL> create/alter profile admin_profile
LIMIT private_sga 5M;
```

CONNECT_TIME : Specifies the maximum number of minutes a session can stay connected to the database (total elapsed time, not CPU time). [wall clock time- unit is minute]

```
SQL> create/alter profile admin_profile
LIMIT connect_time 120/unlimited;
```

IDLE_TIME : Specifies the maximum number of minutes a session can be continuously idle.

```
SQL> create/alter profile admin_profile
LIMIT idle_time 10/unlimited;
```

COMPOSITE_LIMIT : A weighted sum of four resource limits: CPU_PER_SESSION, LOGICAL_READS_PER_SESSION, CONNECT_TIME, PRIVATE_SGA. The weights are established with the 'alter resource cost' statement and can be viewed from the resource_ cost data dictionary view.

```
SQL> alter resource _cost
logical_reads _per_session 10
connect_time 2;
```

Here CPU_PER_SESSION and PRIVATE_SGA will have a cost of 0. cost= (10* LOGICAL_READS_PER_SESSION)+(2* CONNECT_TIME) =1000240

```
SQL> create/alter profile admin_profile
LIMIT composite_limit 1500000;
```

The user will be restricted when this cost exceeds 1500000 or when the value use for LOGICAL_READS_PER_SESSION or CONNECT_TIME in the profile are reached.

Password Management: Following parameters can be used to control a password.

FAILED_LOGIN_ATTEMPTS : Specifies the maximum number of consecutive invalid login attempts allowed before the user account is locked.

PASSWORD_LOCK_TIME : Specifies the number of days the user account will remain locked after the user has made FAILED_LOGIN_ATTEMPTS number of consecutive failed login attempts.

PASSWORD_LIFE_TIME : Specifies the number of days a user can use one password. If the user doesn't change the password within the number of days specified, all connection requests return an error. The DBA then has to reset the password.

PASSWORD_GRACE_TIME : Specifies the number of days the user will get a warning before the password expires. This is a reminder for the user to change the password.

PASSWORD_REUSE_TIME : Specifies the number of days a password can not be used again after changing it.

PASSWORD_REUSE_MAX : Specifies the number of password changes required before a password can be reused. Both PASSWORD_REUSE_TIME and PASSWORD_REUSE_MAX can not be used, one should always be set to unlimited.

PASSWORD_VERIFY_FUNCTION : `Oracle` provides a default script which can be modified.

The following `SQL` shows an example profile.

```
SQL> alter profile admin_profile LIMIT
        sessions_per_user 6
        connect_time 1440
        idle_time 120
        logical_reads_per_call 10000
        password_reuse_time 90
        password_reuse_max unlimited
        failed_login_attempts 6
        password_lock_time 10/1440;

SQL> alter user test profile admin_profile;
```

Dropping a profile:

```
SQL> drop profile admin_profile cascade;
```

By assigning cascade keyword, the users who have that profile will be assigned the default profile.
Data Dictionary: DBA_PROFILES, RESOURCE_COST, USER_PASSWORD_LIMITS.

2 Managing Privileges

Privileges allow a user to access database objects or execute stored programs that are owned by another user. Privileges are assigned to a user, to the special user public, or to a role.

Generally, a database has three types of privileges:

1 Object privileges: Permission on schema objects such as tables, views, sequences, procedures, packages.

2 System privileges: Permission on database level operations, such as connecting to the database, creating users, altering the database or consuming unlimited amounts of tablespaces.

3 Role privileges: Object ans system privileges that a user has by way of a role.

Object privileges: can be granted individually, a grouped in a list, or with the keyword 'all' to implicitly grant all available object privileges for a particular schema object.

Table object privileges: SELECT, INSERT, UPDATE, DELETE, ALTER, DEBUG, INDEX, REFERENCES

View object privileges: SELECT, INSERT, UPDATE, DELETE, DEBUG, REFERENCES

Sequence object privileges: SELECT, ALTER

Stored functions, procedure, packages and java object privileges: DEBUG, EXECUTE

```
SQL> Grant select, insert on HR.employees to test;

SQL> Grant select, insert on HR.employees to public;
```

If privileges are granted to the special user PUBLIC, they are made available to all current and future database users.

When you extend a privilege to another user or role, you can also extend the ability for that grantee to turn around and grant the privilege to others. To extend this extra option, include the keywords WITH GRANT OPTION in the GRANT statement.

```
SQL> Grant select, insert on HR.employees to test
   with grant option;
```

WITH GRANT OPTION only be used if the grantee is either a user or special account Public, not a role.

If you grant an object privilege using WITH GRANT OPTION and later revoke that privilege, the revoke cascades, and the privileges created by the grantee are also revoked.

With object privileges, the database records both the grantor and grantee. Therefore, a grantee can obtain a privilege from more that one grantor. When this multiple grant of the same privilege occurs, revoking one of these grants does not remove the privilege. To remove the privilege, all grants must be revoked.

System Privileges: `Oracle` has more than 170 system privileges of which are listed in the data dictionary view SYSTEM_PRIVIEW_MAP. Some important system privileges are mentioned below:

Database: Alter database, alter system, audit system, audit any

Profiles: Create profile, alter profile, drop profile

Roles: Create role, alter any role, drop any role, grant any role

Sessions: Create session, alter session, alter resource cost, restricted session.

Tables: Create table, create any table, alter any table, drop any table, comment any table, select any table, update any table, delete any table, lock any table, flashback any table.

Tablespaces: Create tablespace, alter tablespace, drop tablespace, manage tablespace, unlimited tablespace.

Triggers: Create trigger, create any trigger, alter any trigger, drop any trigger, administer database trigger.

Users: Create user, alter user, drop user

Views: Create view, create any view, drop any view, comment any table, flashback any table

Powerful role privileges are SYSDBA, SYSOPER (less powerful than SYSDBA)

The optional keywords WITH ADMIN OPTION are required to additionally allow the grantee to confer these privileges on other users and roles.

If you grant a system privilege WITH ADMIN OPTION and later revoke that privilege, the privilege created by the grantee will not be revoked.

Role in data dictionary: DBA_SYS_PRIVS, DBA_TAB_PRIVS, USER_ROLE_PRIVS.