CST3604 Lecture Notes

Physical Design

Physical Data Model – Distributed Design

(Part 1 of ?)

(Lecture Notes 3?)

Prof. Abel Angel Rodriguez

Chapter 4 Database Security

4.1 Overview of Database Security Concepts

4.1.1 Database Security Concepts

- Objectives of chapter 4:
  - Discuss the features of DBMSs that are used for implementing database security.:

- Security requirements are usually defined by the following questions:
  - Who or which users can access data in the database?
  - Who or which applications can access data in the database?
  - What portion of data each entitled user can access?
  - What operations each entitled user can perform on this data?

- Based on the questions above, lets define two security terms:
  - **Authentication:**
    - First entry point to security from a user standpoint.
    - Authentication provides a way of identifying a user, typically by having the user enter a valid user name and valid password before access is granted.
    - The process of authentication is based on each user having a unique set of criteria for gaining access.
    - Usually the client application or OS sends a request to a SERVER to service this request. Server compares a user's authentication credentials with other user credentials stored in a database. If the credentials match, the user is granted access to the network. If the authentication fails and network access is denied.
    - Note that applications can also go through this authentication process.
  - **Authorization:**
    - Once the user authenticates, the authorization process determines whether the user has the authority to access the resources on the network.
    - Simply put, authorization is the process of enforcing policies: determining what types or qualities of activities, resources, or services a user or application is permitted.
    - Usually, authorization occurs within the context of authentication. Once you have authenticated a user, they may be authorized for different types of access or activity.
  - **Accounting/Auditing:**
    - The final security measure is accounting or auditing.
    - Auditing measures the resources a user or application consumes during access.
    - This can include the amount of system time or the amount of data a user has sent and/or received during a session.
    - Accounting is carried out by logging of session statistics and usage information and is used for authorization control, billing, trend analysis, resource utilization, and capacity planning activities.

Data Security Levels in the IT environment

- Outer-level Security – The security of data is supported at several different levels, including network, application, and database

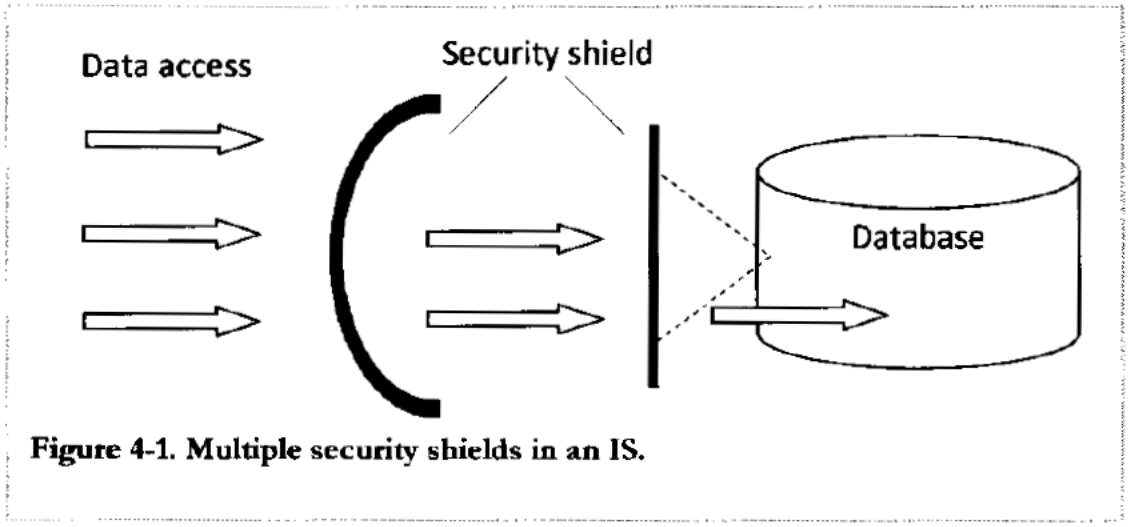


Figure 4-1. Multiple security shields in an IS.

- Outer-level security may not be enough to stop a user from bi-passing the outer security layer and breach the database internally:
  - Security implemented outside the database plays an important role, it is crucial to understand that this outer security shield is not sufficient to protect the data
  - This is where database security plays an important role.

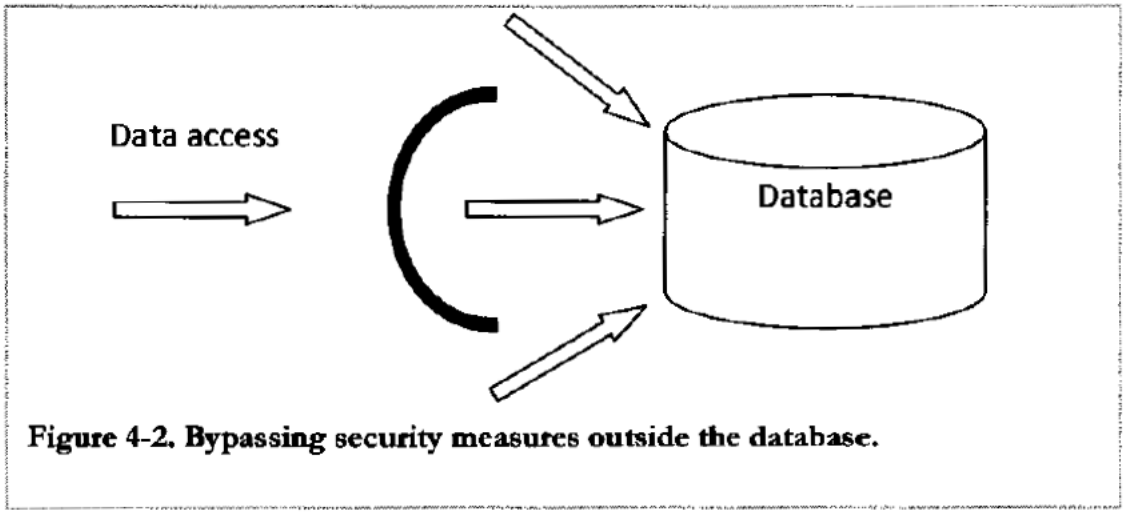


Figure 4-2. Bypassing security measures outside the database.

- Our focus is Database Security:

- In-database security we add another layer of protection on the database itself, thus we make the overall data protection more reliable.

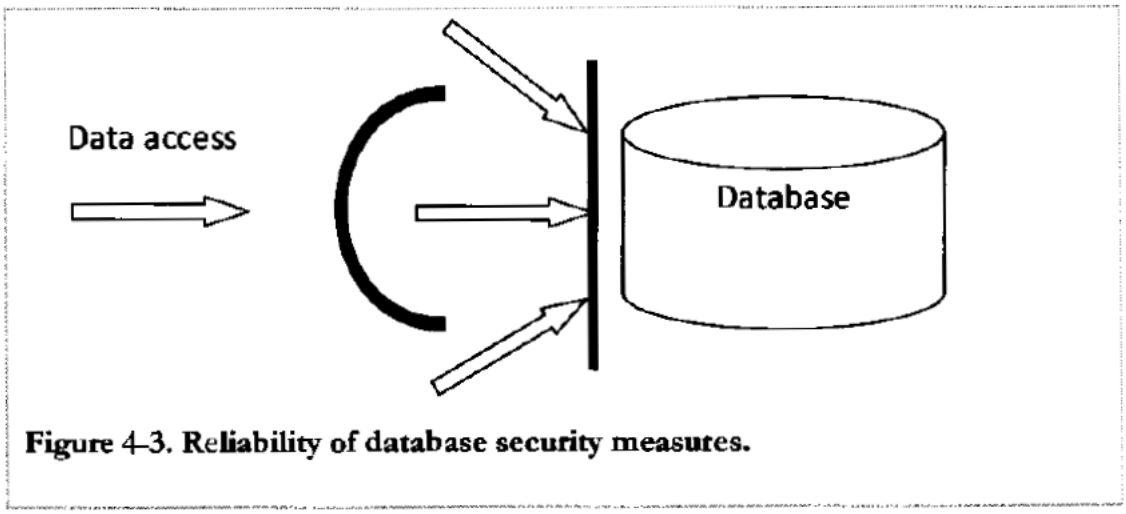


Figure 4-3. Reliability of database security measures.

- Another advantage of implementing security in the database is that security policies are applied once, instead of being implemented repeatedly in different applications on the database

- High-level requirements for data security:

Data security must be supported within the database.

Database security is about ensuring that only authorized users can perform allowed operations on the data.

  - Database security falls within the category of AUTHORIZATION which specifies which users can perform which operations and on what data

- Security can be measured to be valid, by defining the relationships between the following 3 elements:

  - **Users (Who can access)**
  - **Operations (what can they do)**
  - **Data (what data they can access)**.

- For example:

  - If the user John requests data about the age of employees from the table Employee, the system checks whether John has the permission to select on the attribute Age of the table Employee

  - Violations of the relationships between a user and operations the user can perform on particular data results in security threads such as:

    - Theft and fraud
    - Loss of confidentiality
    - Loss of privacy, etc.

Database Security versus Database Integrity

- There is a difference between the concept of security and data integrity contrainst:
  - **Integrity constraints** – prevents actions which compromise the integrity of the data, such as data types, etc.
    - Integrity measures protect data from any authorized changes that can violate the correctness of the data.
    - Integrity constraints DO NOT implement security.
  - **Security measures** –Security measures shield data from unauthorized access

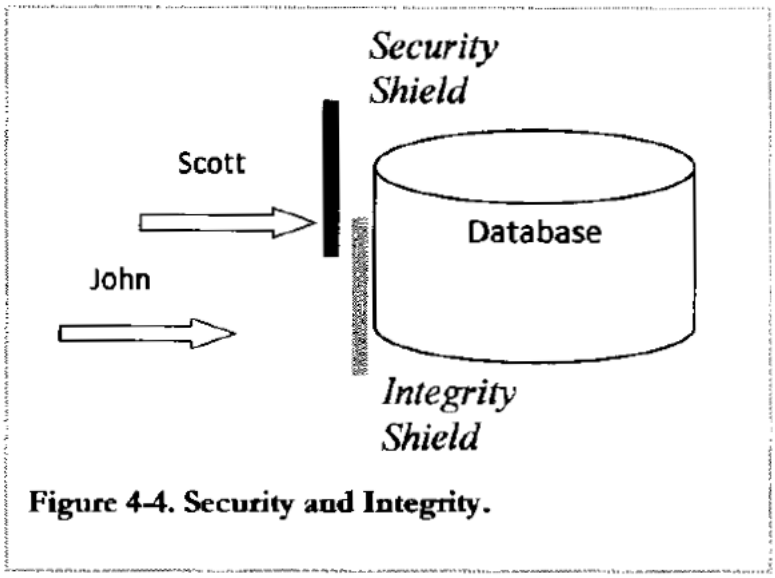


Figure 4-4. Security and Integrity.

- Example from Malyuta's book:

The integrity of data is the correspondence of data to specific structural or action oriented business rules. It must be supported by the appropriate implementation of integrity constraints[11], and it cannot be enforced by security measures. For example, if according to a business rule which states that the values of the attribute Age in the table Employee has to be between 18 and 65, then the CHECK constraint on the attribute Age will guarantee the integrity of this attribute. However, the CHECK constraint does not implement any security control. If the security of data requires that only the user John can modify the attribute Age, then security measures have to prevent any other user from changing values of this attribute. On the other hand, security measures are not related to integrity – even with a successful implementation of security, but without the corresponding integrity constraint, the user John can make a mistake while entering data into the Age attribute and violate the correctness of data.

Database Security & Database Availability

- Database needs to be available :
  - Database needs to be available for security measures/authorization to be effective or work.
  - Database Security cannot make data available to a user, but it can secure the data when data is available.

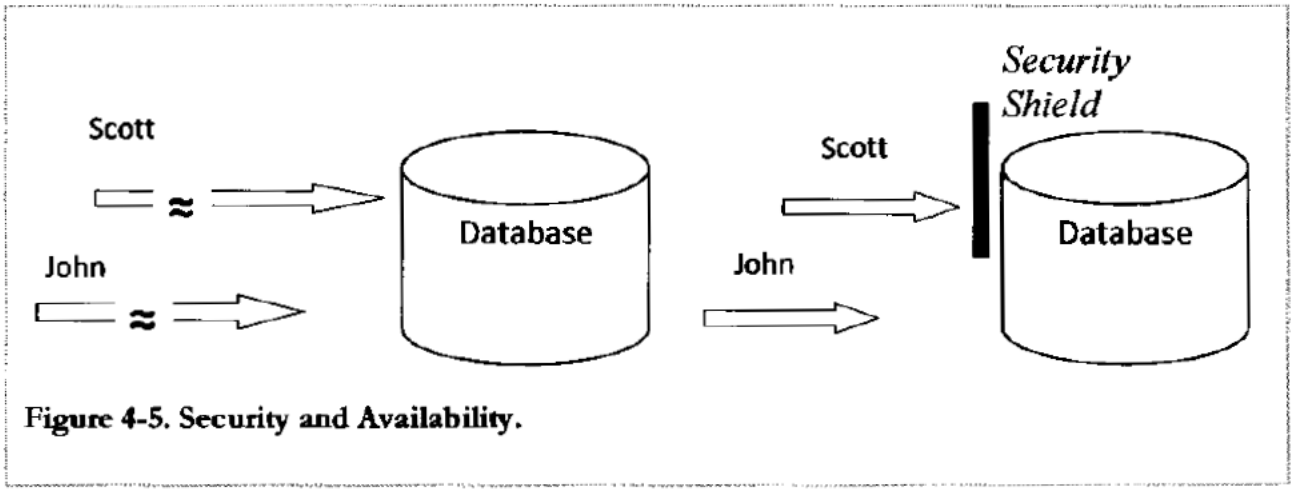


Figure 4-5. Security and Availability.

Summary of DBMS Security Features provided

- Security measures provided by DBMSs include:

- *Authentication of users.* Authentication measures define users who are entitled to work with a database and their credentials which the DBMS uses to authenticate users to a database.
- *Authorization of access to data.* For users who are authenticated for the database, authorization measures define the kind of actions that these users can perform on that database.

4.2 Implementing Security – Users, Operations & Data

4.1.1 Users & Schema (Database)

- In previous section, we discussed that security can be measured to be valid, by defining the relationships between the following 3 elements:

  1. **Users (Who can access)**
  2. **Operations (what can they do)**
  3. **Data (what data they can access)**.

- In the next sections, we will analyze methods databases use to address these three security elements. But first we start with **Users**.

Importance of establishing a Security Policy for your Database

- Is important to develop a security policy for your database:

  - The security policy establishes method for protecting the data from accidental or malicious destruction.
  - Each database should have an administrator or security administrators who is responsible for implementation and maintaining this security policy.
  - The administrator can be one individual if the database is small or a group of individuals if the database is large thus requires several administrators.

Managing Users and Database Resources (Objects)

- Users will manage and consume the database resources:

  - A user account with the right permission and privileges is required to access or manage the resources.
  - Therefore, our objectives is as follows:

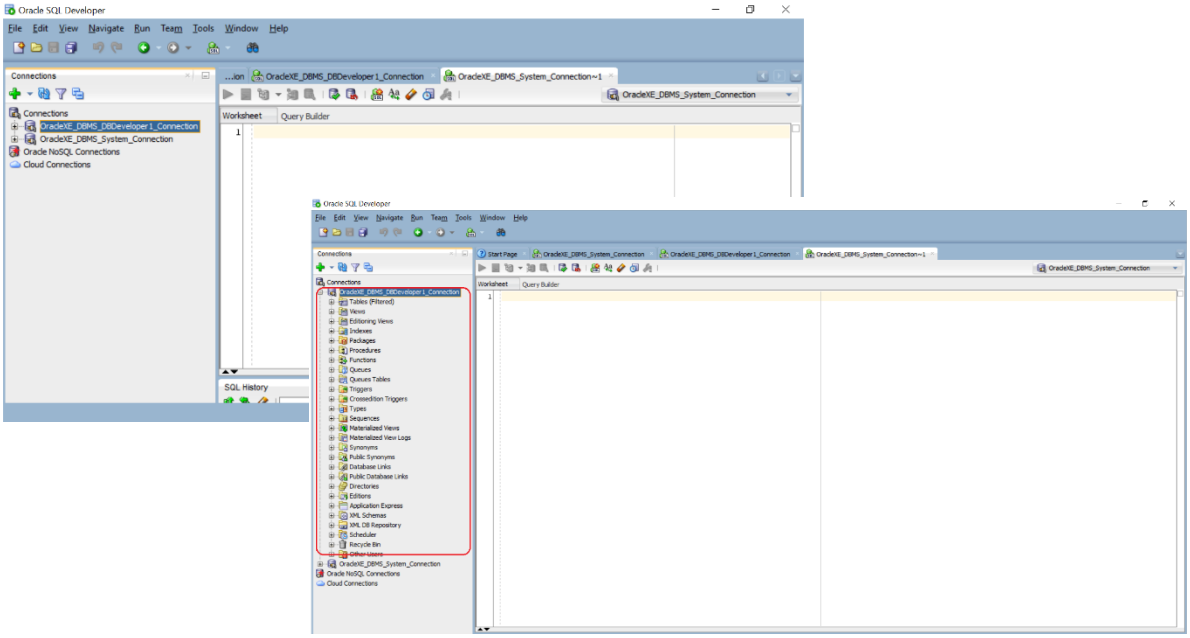
    - Create users
    - Assign Roles to grant privileges to designated objects.
    - Assign Privileges to designated database objects

Users & Schema

The Schema

- A **Schema**:
  - A schema is a collection of objects, storage, disk etc., tied to one **USER**.
  - It can also be considered a database created within the DBMS.
  - Some other DBMS systems allow you to create multiple databases in one DBMS instance or server application.
  - In Oracle, you create separate schemas. So, you could look at a schema as an individual database in Oracle like some other DBMS have.
  - The important thing to point out is that is owned by the one **USER** account that created it.
  - If you recall from your projects, you created a user, then a connection using the user account. The result was a SCHEMA with database objects, etc.

- For example, below diagrams shows a SCHEMA in SQL Developer named DBMS_DBDeveloper1_connection & all the objects within this schema



The User Account

- The first part of security is to authenticate the **USER account**.
  - Username & Password are the primary authentication security tools.
  - There are additional security measures that can be taken to protect the username & password such as: password length policies, locking system after several incorrect passwords, etc.

- The **USER account**:
  - A **USER** is a database object.
  - Must be created in the database & given authentication credentials (username & password)
  - The fact a user exists in the database does not mean they can perform any operation.
  - Explicit **permission & privilege** must be assigned to a user to dictate what action or access the user account has after it's been created. Such as relationship between users, database objects (data) and what operations can be done on the objects.

  - Users can be granted permission to do the following:
    - Perform actions on all objects (all tables, indexes, stored procedures, etc.)
    - Perform actions on all objects of a type (tables only or indexes only or stored procedures only etc.)
    - Perform actions on one object only
    - Etc.

  - Privileges can be taken away or revoked.

Type of Database Users in Oracle

- Let's look at some of the type of database users in Oracle:

Database Administrator

- This is the **DBA** or **database administrator**:
  - Every DBMS must have a DBA. The default DBA account as you know is **system** and you provided a password during the installation of Oracle
  - A database system can be large and can have many users so database administration may require a group of DBAs who share responsibility for managing the database.
  - This means, you may have to create many DBA user accounts with many levels of permissions to share the database administrative tasks.
  - A database administrator's responsibilities can include the following tasks:
    - Installing and upgrading the Oracle Database server and application tools
    - Allocating system storage and planning future storage requirements for the database system
    - Creating primary database storage structures (tablespaces) after application developers have designed an application
    - Creating primary objects (tables, views, indexes) once application developers have designed an application
    - Modifying the database structure, as necessary, from information given by application developers
    - Enrolling users and maintaining system security
    - Ensuring compliance with Oracle license agreements
    - Controlling and monitoring user access to the database
    - Monitoring and optimizing the performance of the database
    - Planning for backup and recovery of database information
    - Maintaining archived data on tape
    - Backing up and restoring the database
    - Contacting Oracle for technical support

Security Offices

- In some cases, security officers or user accounts to manage user security such as:
  - Create users
  - Control and monitor user access (permissions and privileges)
  - Maintain security.
  - The DBA may be responsible for these tasks or may assign a security office to handle this instead of DBA.
  - Security officers are example of other admins created to share the security responsibility.

Network Administrators

- In some cases, where other Oracle networking products are being used such as Oracle Net Services, a separate group of administrators are created to administer such networking devices and products.

Application Developers

- These types of users design and implement database applications:
  - Their responsibility can include:
    - Designing and developing the database application
    - Designing the database structure for an application
    - Estimating storage requirements for an application
    - Specifying modifications of the database structure for an application
    - Relaying this information to a database administrator
    - Tuning the application during development
    - Establishing security measures for an application during development
  - Application developers perform some of these tasks in collaboration with DBAs

Application Administrators

- In some cases, administrators are created to administer one application:
  - DBAs are created for that one application only.
  - So each application has it's own Application DBA

Database Users

- Database users interacts with the database through applications created by the database developers or utility programs that allow database users to perform some operation:
  - Their responsibility & usage can include.
    - Entering, modifying, and deleting data, where permitted
    - Generating reports from the data

- The **Database Administrator USER account**:
  - A user account with the many privileges.
  - It is an administrative account that can create users, assign privileges/permissions, revoke permissions etc.
  - For example: **system** account is the main administrator account in Oracle.

Pre-defined User Accounts in Oracle

- Oracle includes the following predefined or built-in accounts:

  - **Built-in administrative accounts: SYS, SYSTEM, SYSMAN & DBSNMP**

    - SYS:
      - Automatically created and granted DBA role
      - Default password = CHANGE_ON_INSTALL
      - **Can perform all administrative functions**.
      - All the base or special tables, views etc., that are essential to the operation of Oracle are stored in the SYS schema (database)
      - These tables should never be modified by any administrator or user.
      - No tables should be created in this SYS schema
      - Make sure most users except the DBA be able to connect to the Oracle Database using SYS account.

    - SYSTEM:
      - Automatically created and granted DBA role
      - Default password = MANAGER
      - Peforms all administrative functions except:
        - Backup & Recovery
        - Database upgrade

      - This DBA account is used to create additional tables and views for Oracle functionalities & tools
      - Again, do not store table in the SYSTEM schema

    - SYSMAN:
      - Automatically created and granted special privileges
      - Used to perform Oracle Enterprise Manager administrative tasks (Server Agent related admin tasks).
      - Do not delete this account.
      - Oracle recommends you creating individual database administrative accounts and not use this generic SYSTEM account to administer the database.

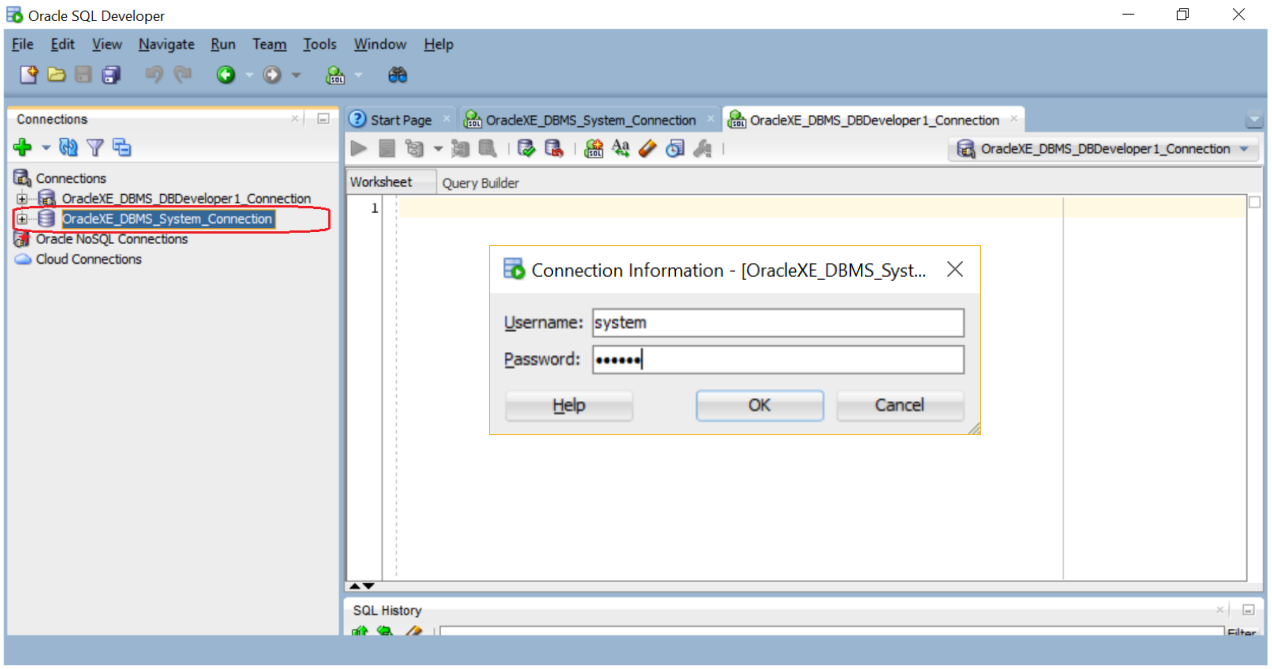
    - DBSNMP:
      - Automatically created and granted special privileges
      - Used to monitor and manage the database. (Server Agent related admin tasks).
      - Do not delete this account.

- **Sample Schema Accounts**
  - There are a few accounts created automatically that are used for the sample schema or database that come with Oracle for teaching purpose.
  - Example are users: HR, SH & OE.
- **Internal Accounts**
  - Accounts that manage special schemas for individual Oracle features or components.
  - Do not delete these or try to use them. Meant for internal Oracle operations.

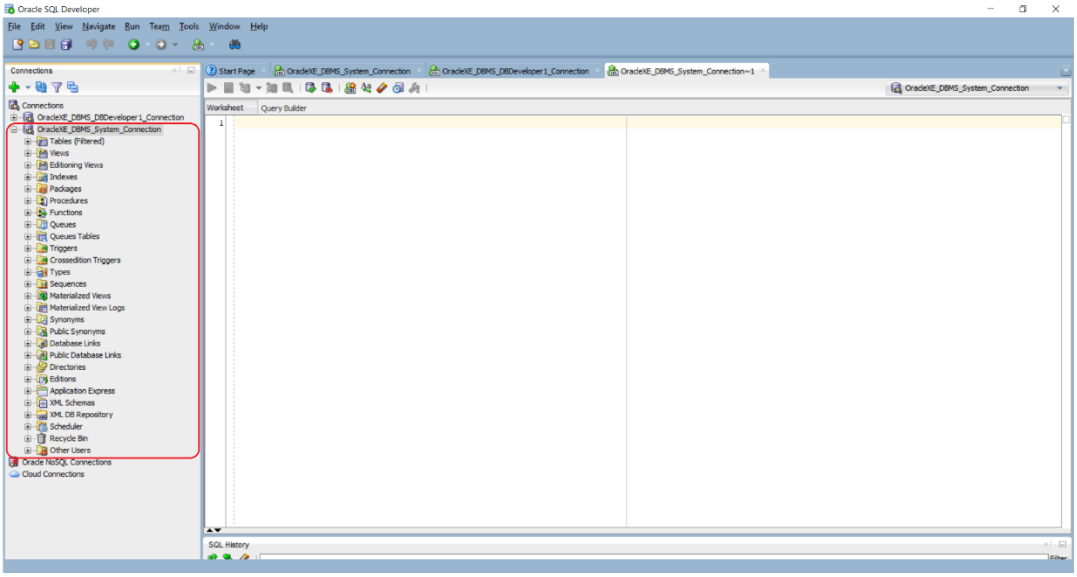
- All other accounts must be created!

Creating a User Account in SQL Developer

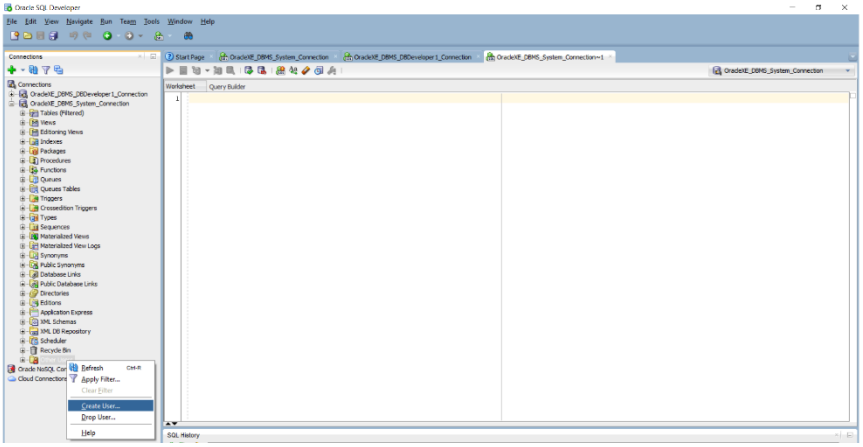
Step 1 - Log in to the schema (This case login into the system schema)



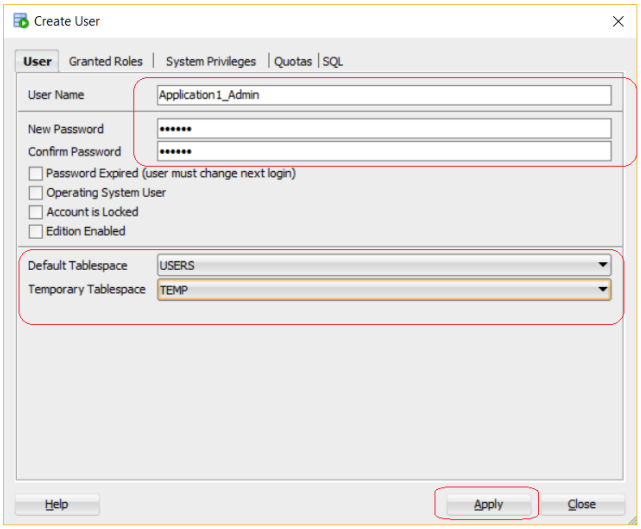
Step 2 – Expand schema to see all objects



Step 3 – Go to Other Users folder & *Right-click* and select Create User…

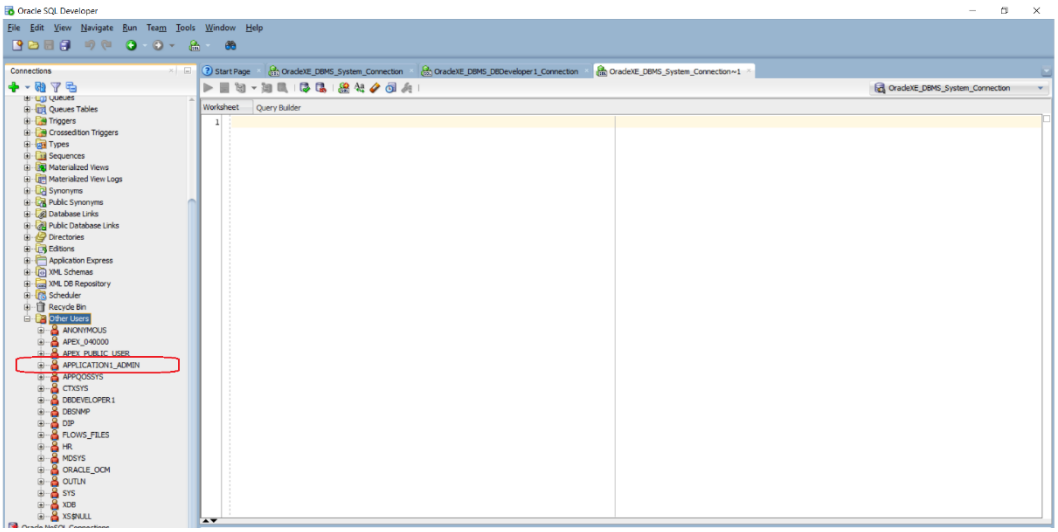


Step 4 – In Create User screen, select the User tab and enter username & password and apply or confirm



- Select the Tablespace = Users
- Select the Temporary Tablespace = TEMP

Step 5 – User account Object is created

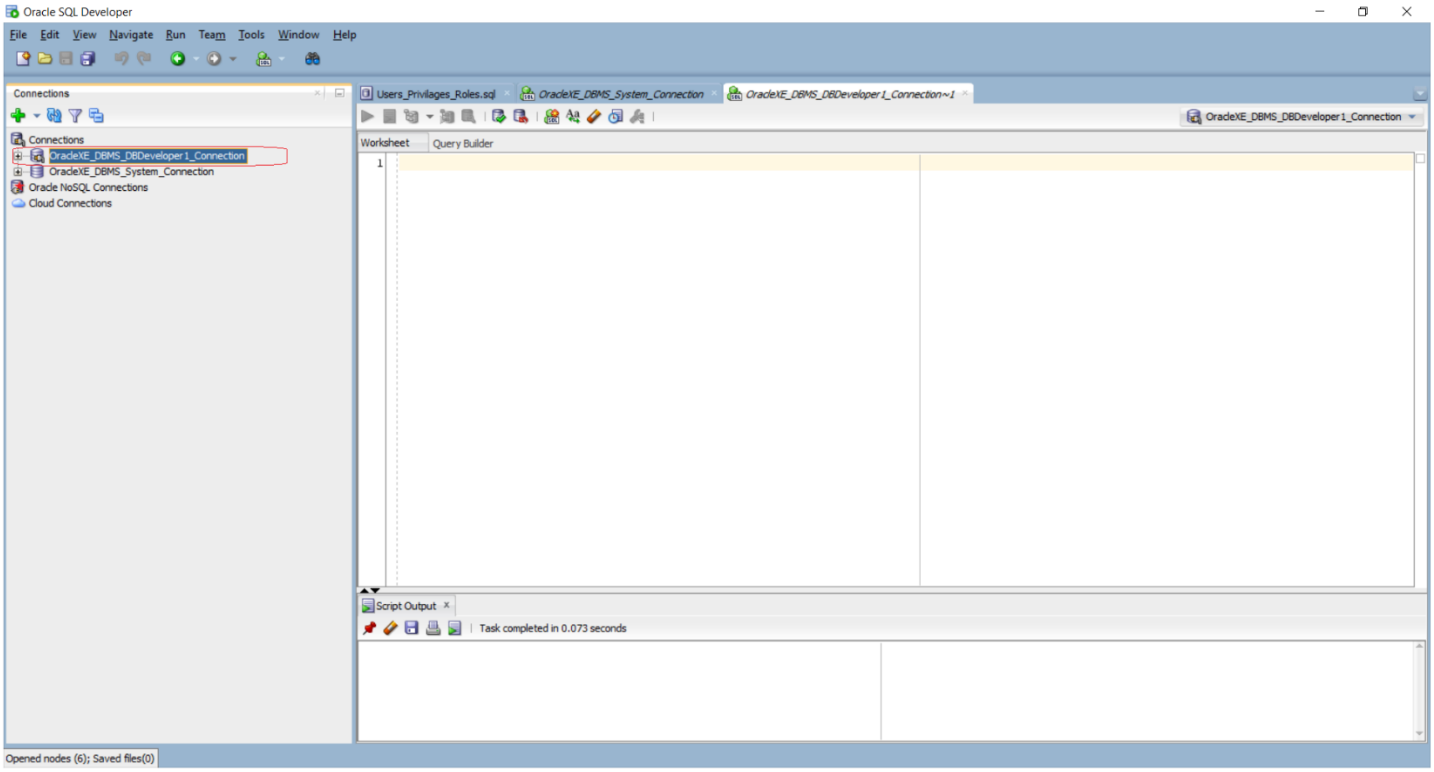


- Note the following:
  - This **Application 1_Admin** account exists but has no access or privileges granted.
  - It is just a user account.

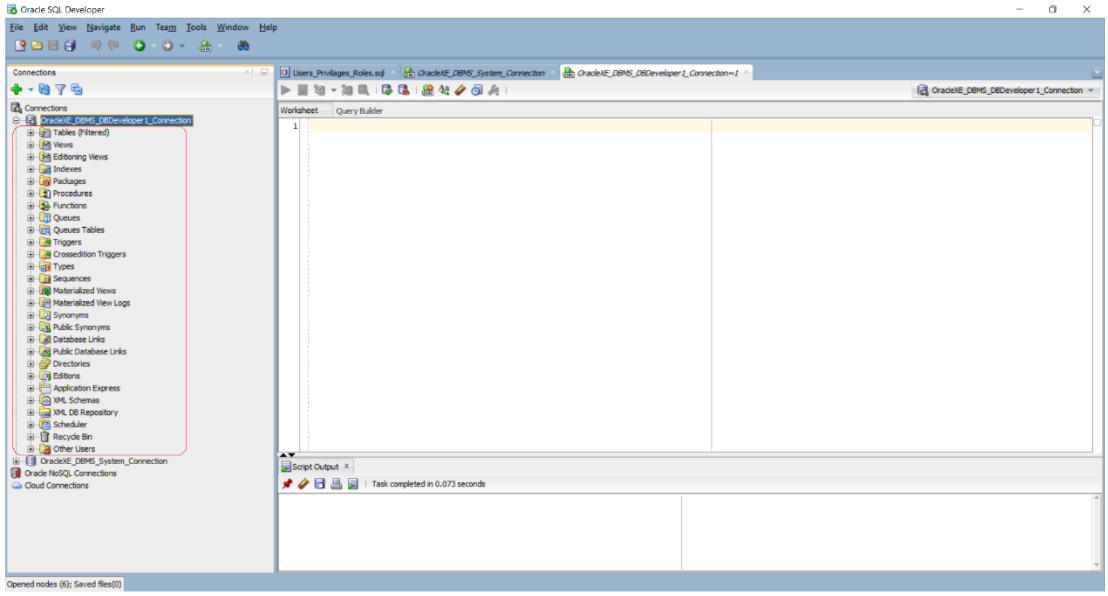
Creating a User Account Using the DBDeveloper1 Connection in SQL Developer

- In this example, we create a user account using the DBDeveloper1 account which also has privileges to perform many administrative functions.
- We will create a user name User1.

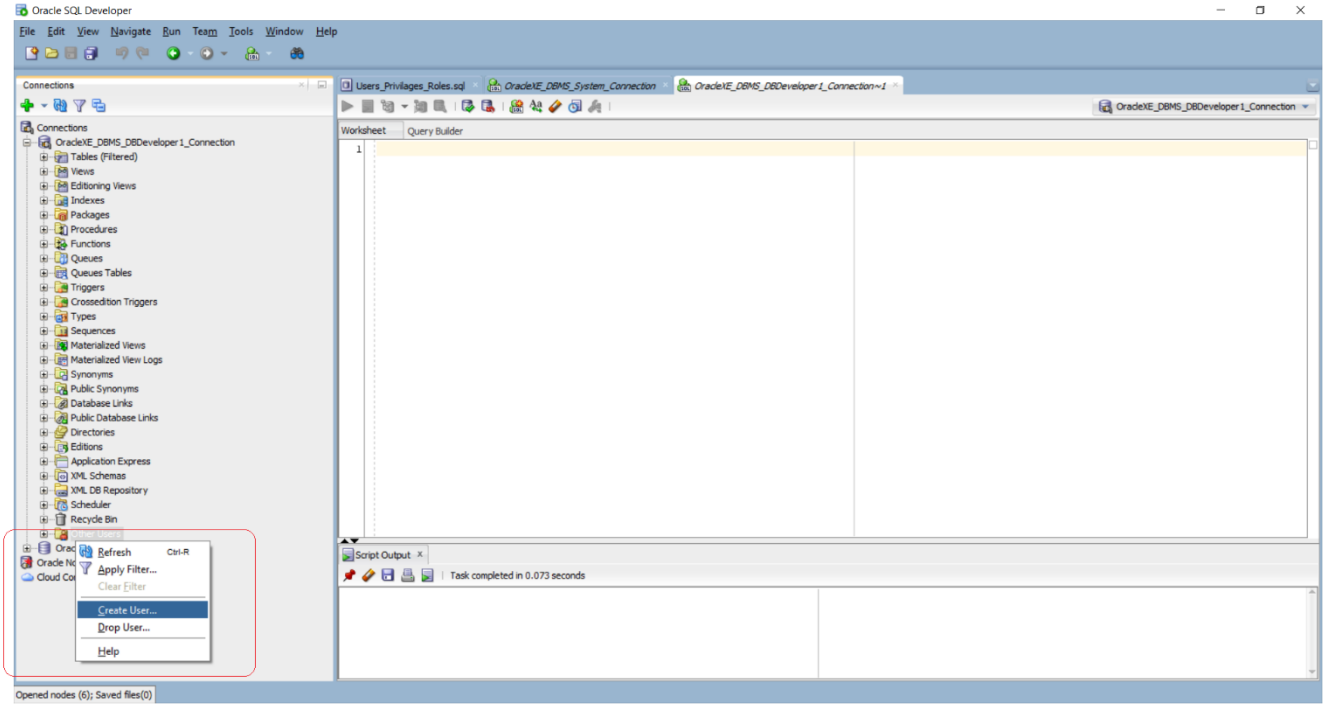
Step 1 - Log in to the schema (This case login into the DBDeveloper1 schema)



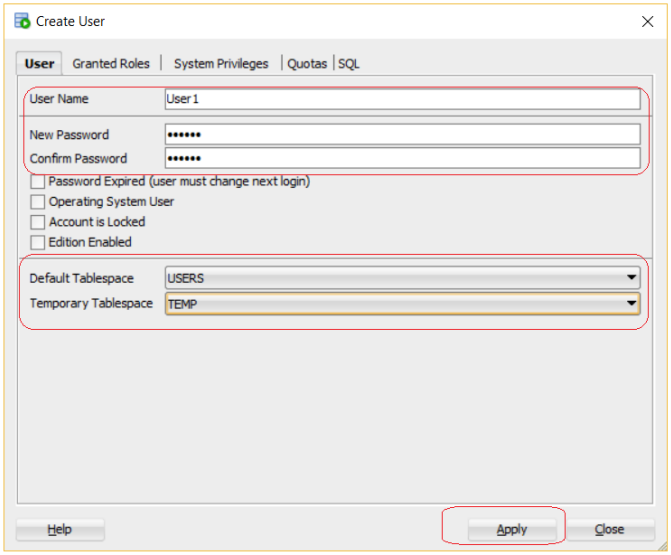
Step 2 – Expand schema to see all objects



Step 3 – Go to Other Users folder & *Right-click* and select Create User…

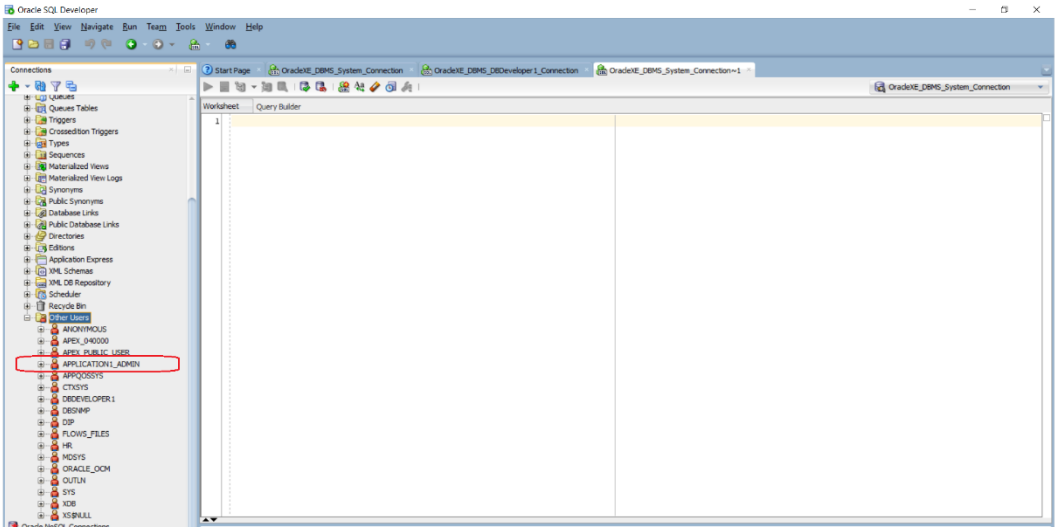


Step 4 – In Create User screen, select the User tab and enter username & password and apply or confirm



- Select the Tablespace = Users
- Select the Temporary Tablespace = TEMP

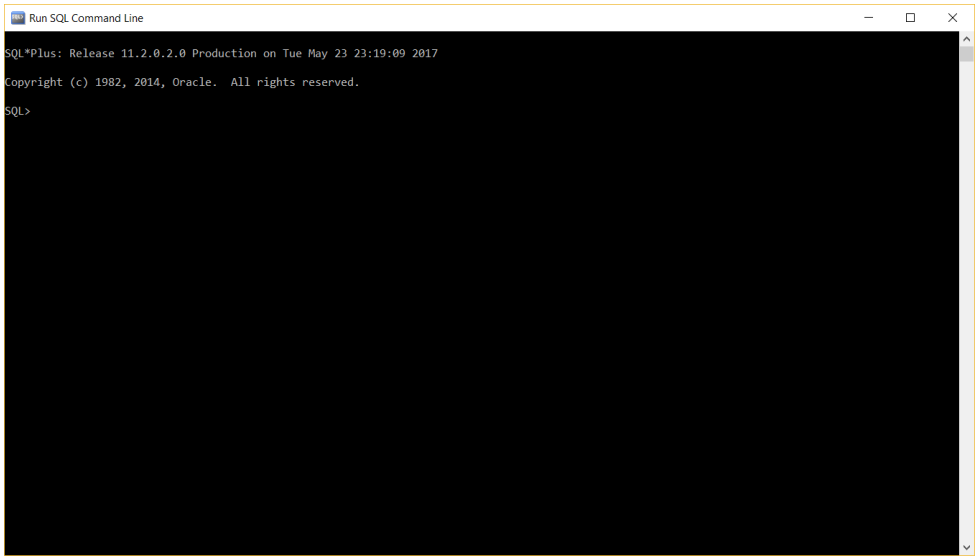
Step 5 – User account Object is created



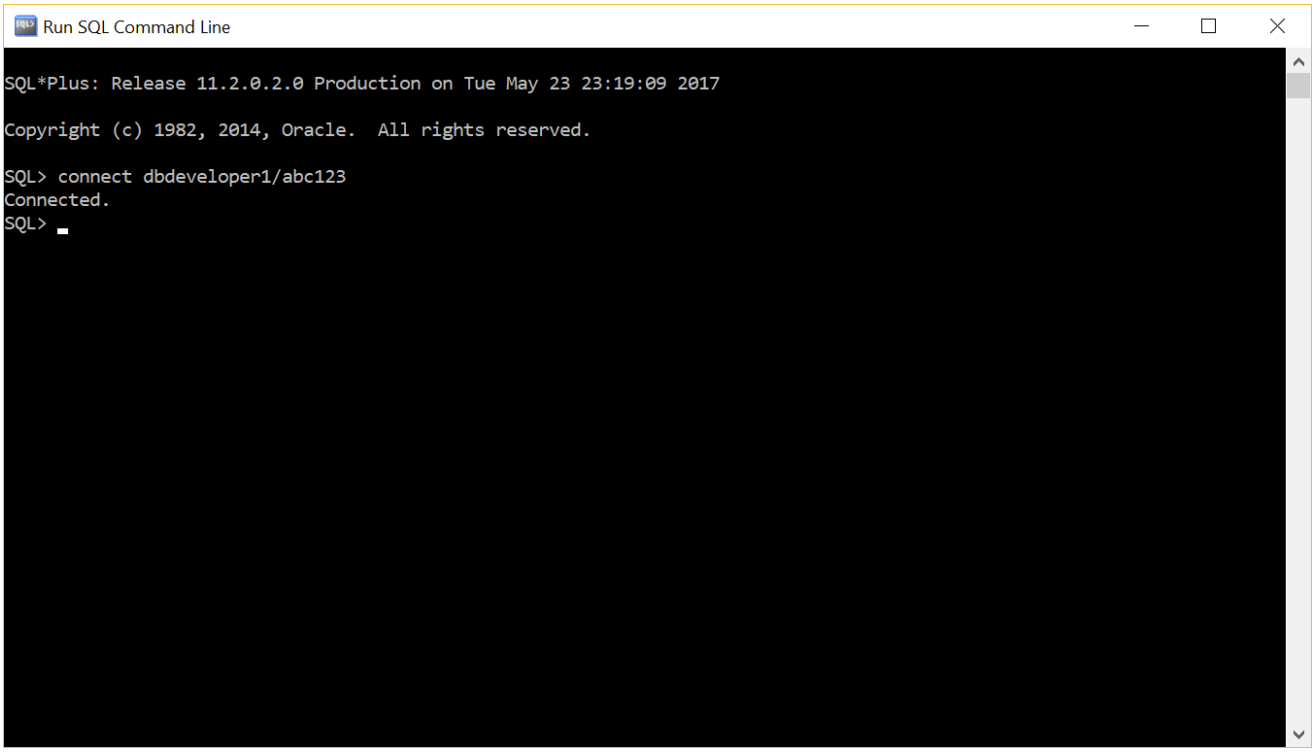
- Note the following:
  - This **User 1** account exists but has no access or privileges granted.

Creating a User Account Using the DBDeveloper1 Connection in SQL Plus (SQL Command Line)

Step 1 – Open the SQL Command Line or SQL Plus



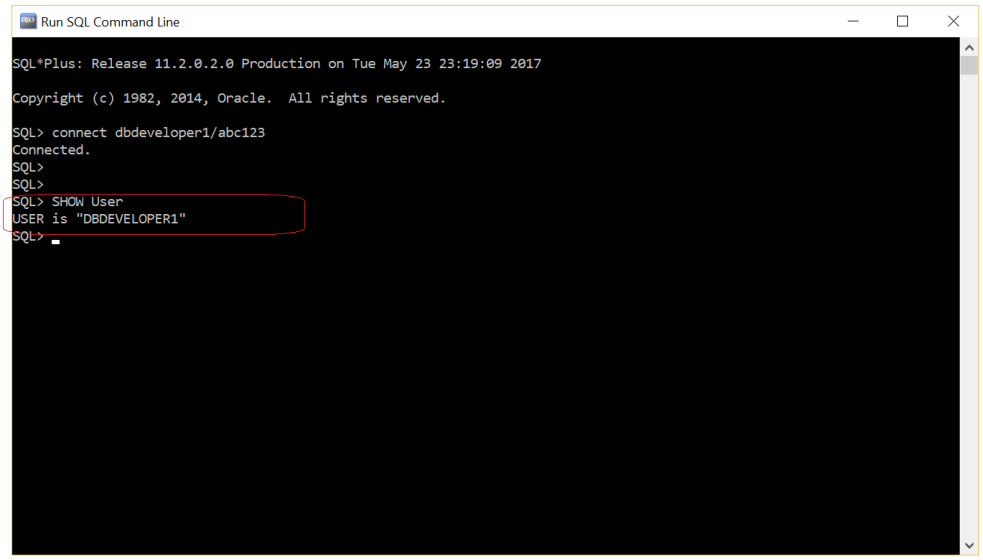
Step 2 – Connect to the Database as DBDeveloper1



- Command used:

  Syntax: **Connect Username/password**
  Command: **Connect dbdeveloper1/abc123**

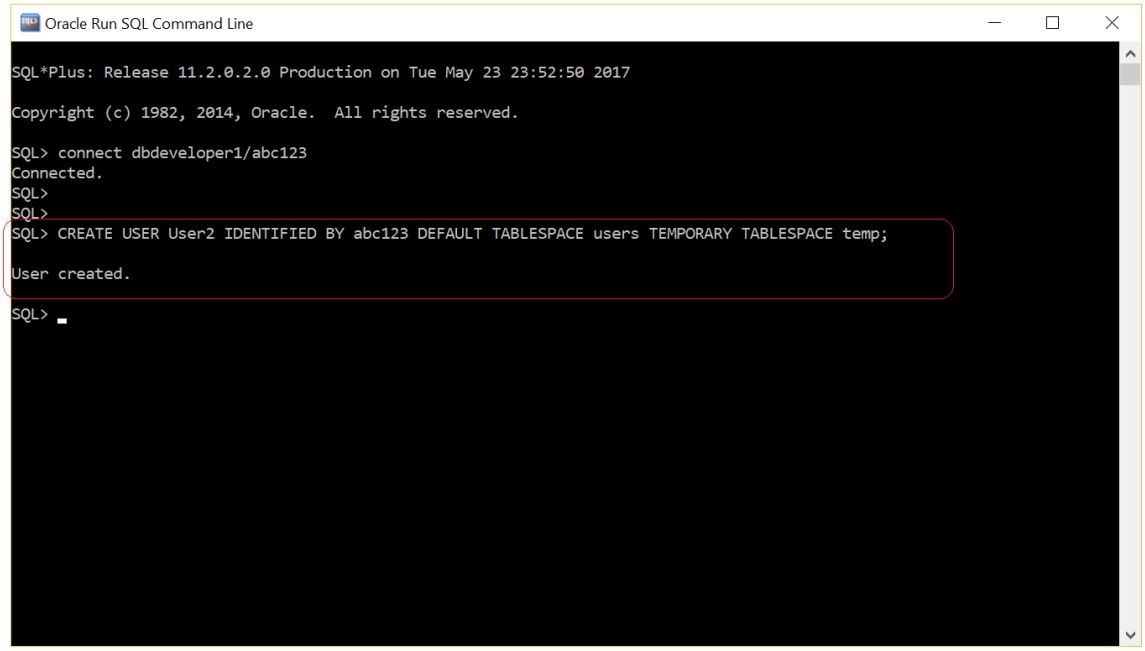
Step 3 – To prove you are connected as DBDeveloper1, you can use the SHOW User command:



- Command used:

```
SHOW User
```

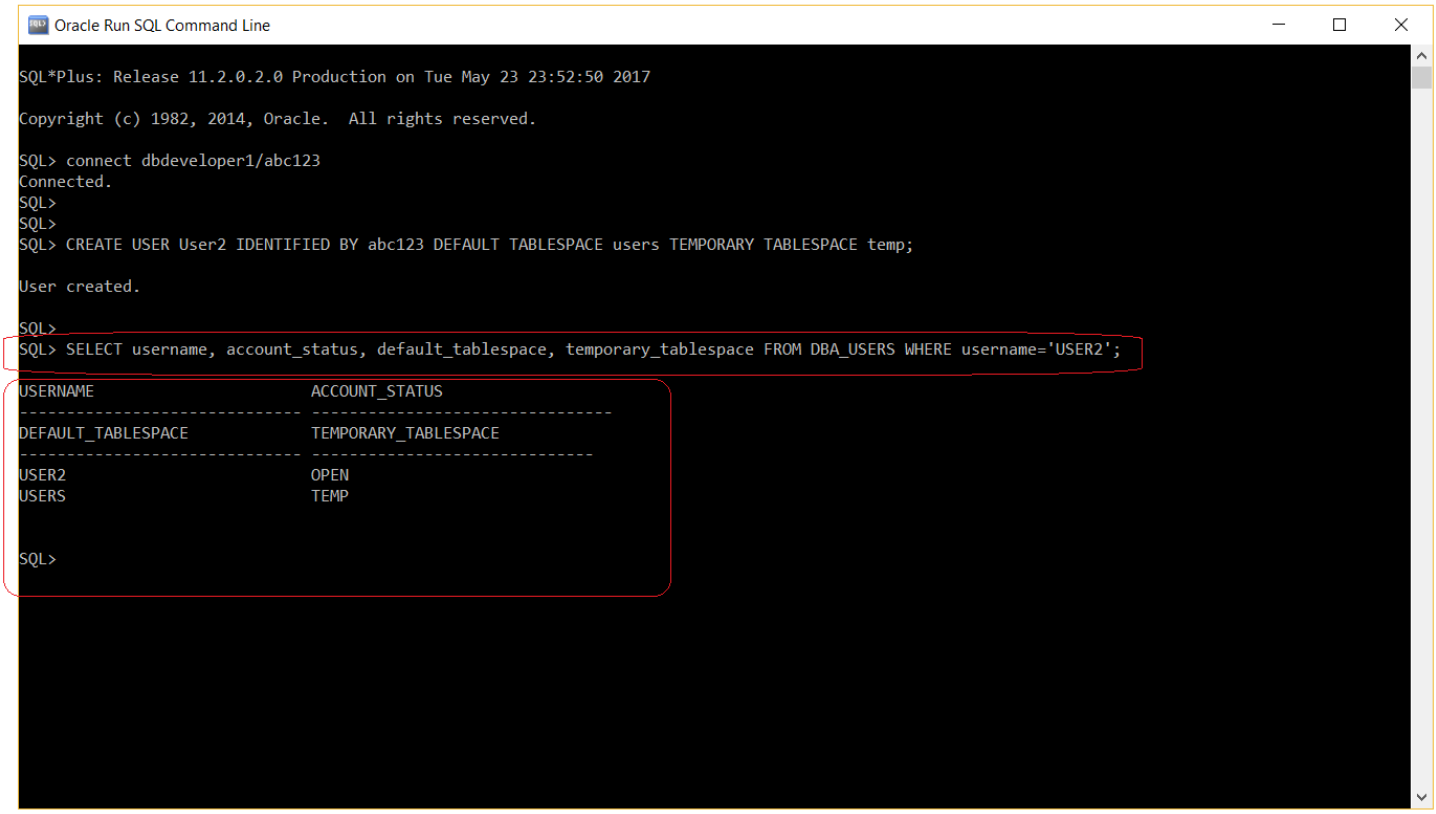
Step 4 – Enter Command to Create User with Username, Password, Default Tablespace and Temporary Tablespace values



- Command used:

```
Syntax:
CREATE USER username                -- username
IDENTIFIED BY password              -- assigning a password via IDENTIFIED BY keyword
DEFAULT TABLESPACE TablespaceName   -- tablespace where user will reside.
TEMPORARY TABLESPACE TEMP           -- Temporary tablespace

Command:
CREATE USER User2 IDENTIFIED BY abc123 DEFAULT TABLESPACE users TEMPORARY TABLESPACE temp;
```

Step 5 – Verify the User exists by running a query on the DBA_USERS table which contains all user information



Syntax:
SELECT QUERY on DBA_USERS table

Command:

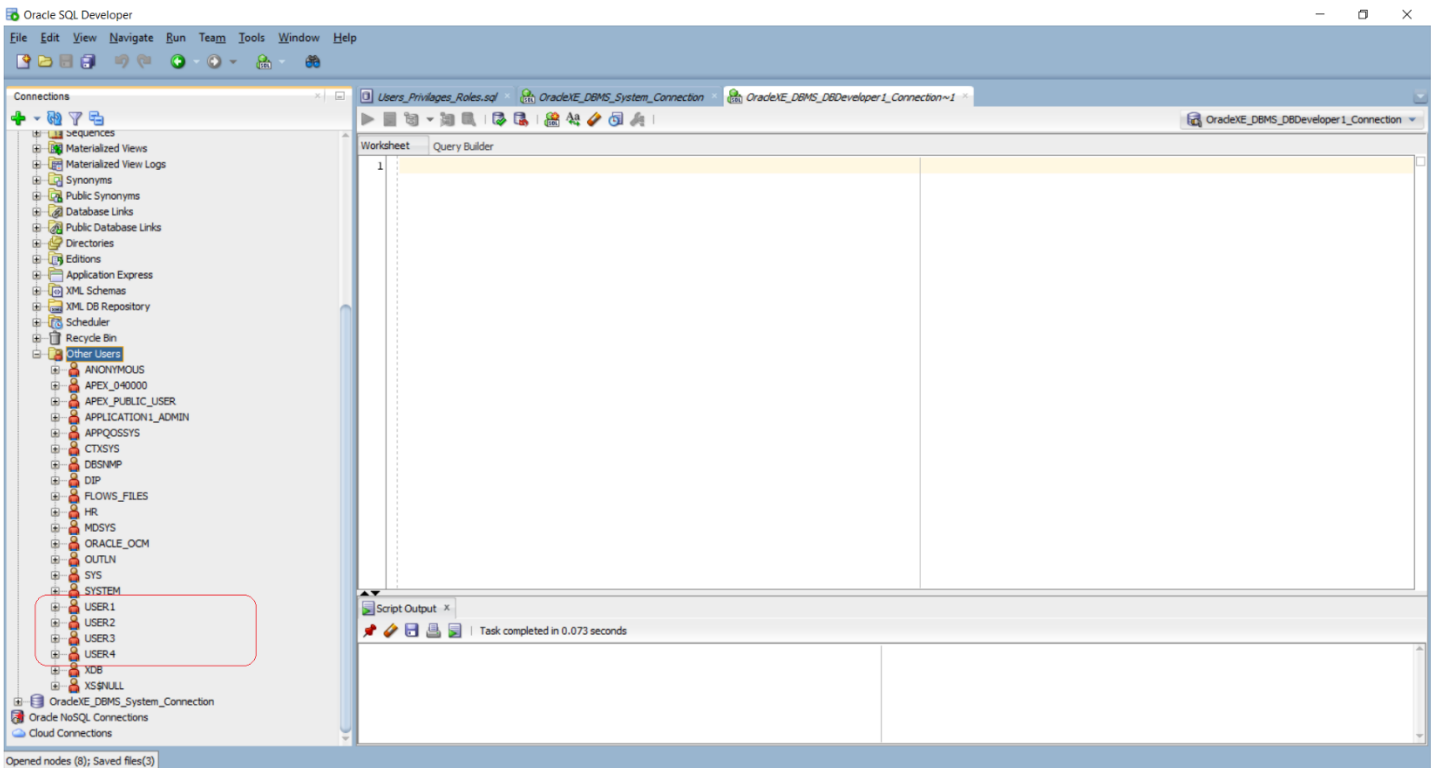
```
SELECT username, account_status, default_tablespace, temporary_tablespace
FROM DBA_USERS
WHERE username='USER2';
```

- Note that: the username must be CAPITAL LETTERS since this is how is stored in Oracle.

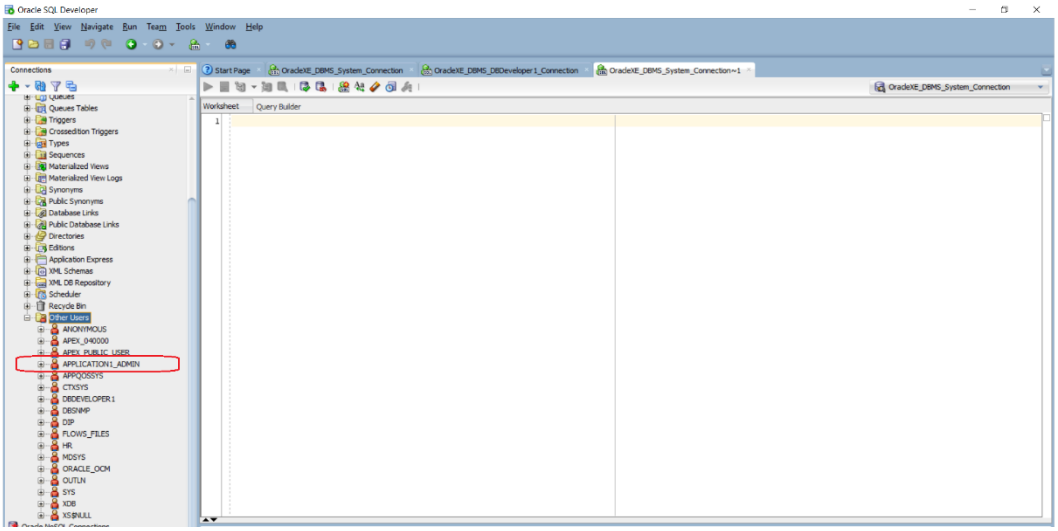
Create User3 & User4 using either SQL Developer or SQL Plus

Step 1 – Repeat the steps in previous section to create two additional users: USER3 & USER4

Step 2 – At the end, we will have the 4 users object created



Step 5 – User account Object is created



- Note the following:
  - This **User 1** account exists but has no access or privileges granted.

User account & schema ownership

- Some **USER account** characteristics:
  - All objects created by a user, are owned by this user.
  - A **USER account** can be given permission to create other objects or schema (connection)
  - A schema is created via a user account with the right permissions, who owns that schema, thus all objects in the schema.
  - The name of the schema is the same as the name of the user who created the connection or schema.
  - Other users in the schema can be given permissions and privileges to perform DBA functionalities etc.
  - Example below:

    - Scott has access & permissions to all object in Scott's schema (Create, alter, drop, insert, delete etc.), but not John's.
    - John has access & permissions to all object in John's schema (Create, alter, drop, insert, delete etc.), but not Scott's.

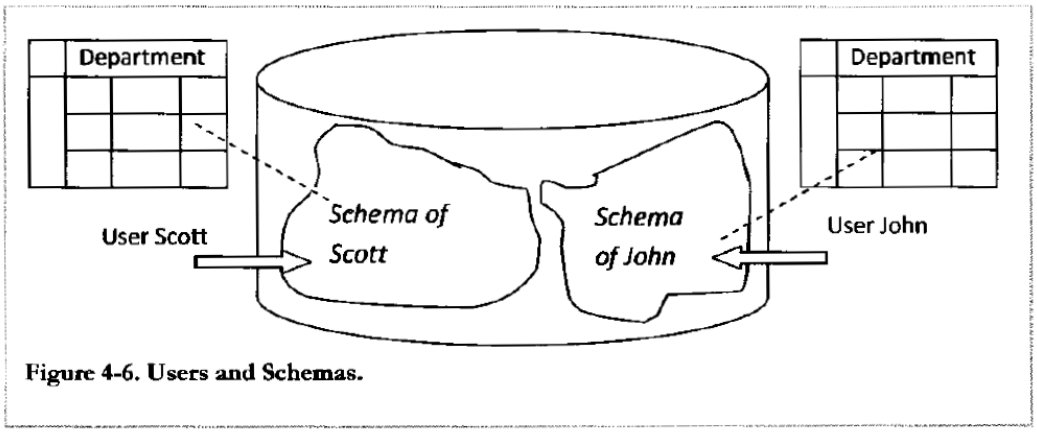


Figure 4-6. Users and Schemas.

Summary up to this point

- Here is a summary and outcome of what we have accomplished:

  - We created 4 user accounts (USER1, USER2, USER3 & USER4)
  - Users were created using SQL Developer and SQLPlus so you can see both methods.
  - There is a reason why both methods are shown. We will be testing our user objects using both admin tools.

- **VERY IMPORTANT INFORMATION:**

  - At this point, we CANNOT connect to the database with any of these 4 user accounts created either with Oracle SQL Developer or SQLPlus.
  - Reason being that any of these accounts have the privileges to connect to the database.
  - In the next sections, we will discuss how to GRANT privileges to these accounts so they can connect to the database.

4.1.2 Managing User Privileges and Roles

- In this section, we will address how to grant privileges and roles to users.

Managing User Privileges

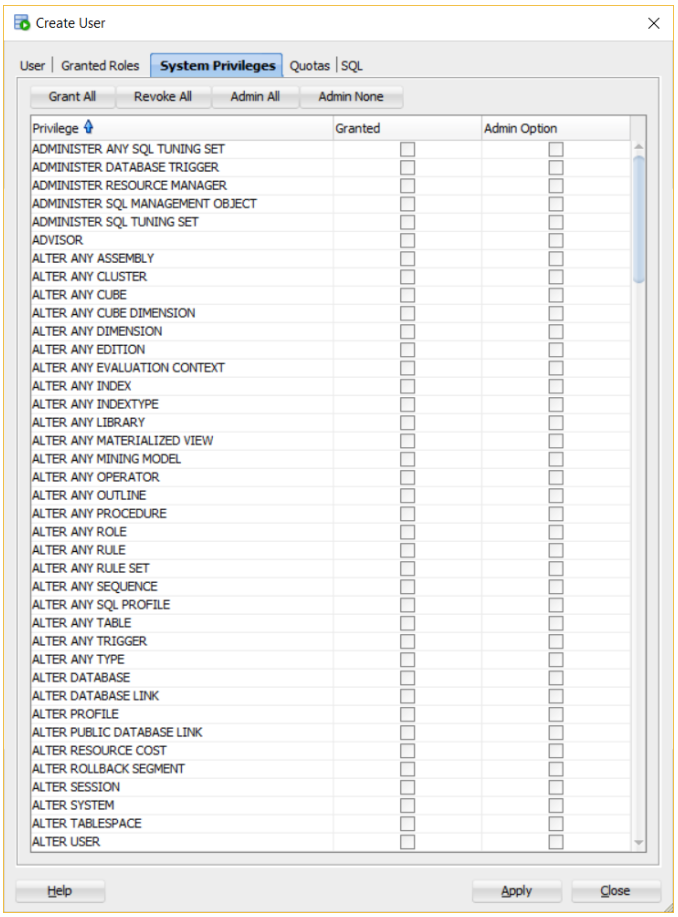
- You use privileges and roles to control user access to data and actions that can be taken by a user.
  - Privileges are the rights to perform a specific action on a schema object (tables, views, indexes, etc.)
  - There are 3 types of privileges:

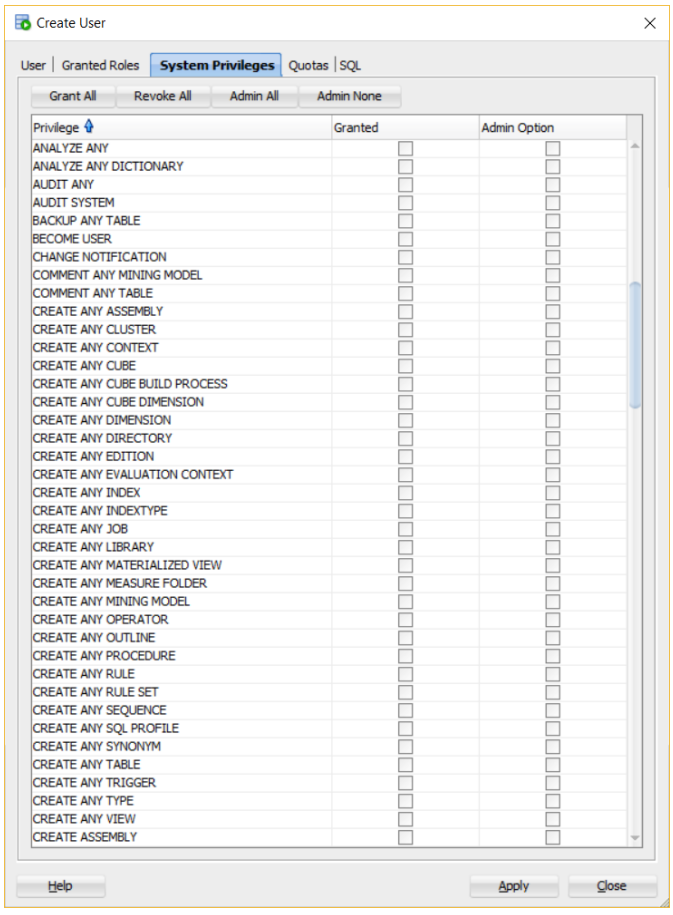
Type	Description
System privilege	A system-defined privilege usually granted only by administrators. These privileges allow users to perform specific database operations.
Object privilege	A system-defined privilege that controls access to a specific object.
Role	A collection of privileges and other roles. Some system-defined roles exist, but most are created by administrators. Roles group together privileges and other roles, which facilitates the granting of multiple privileges and roles to users.

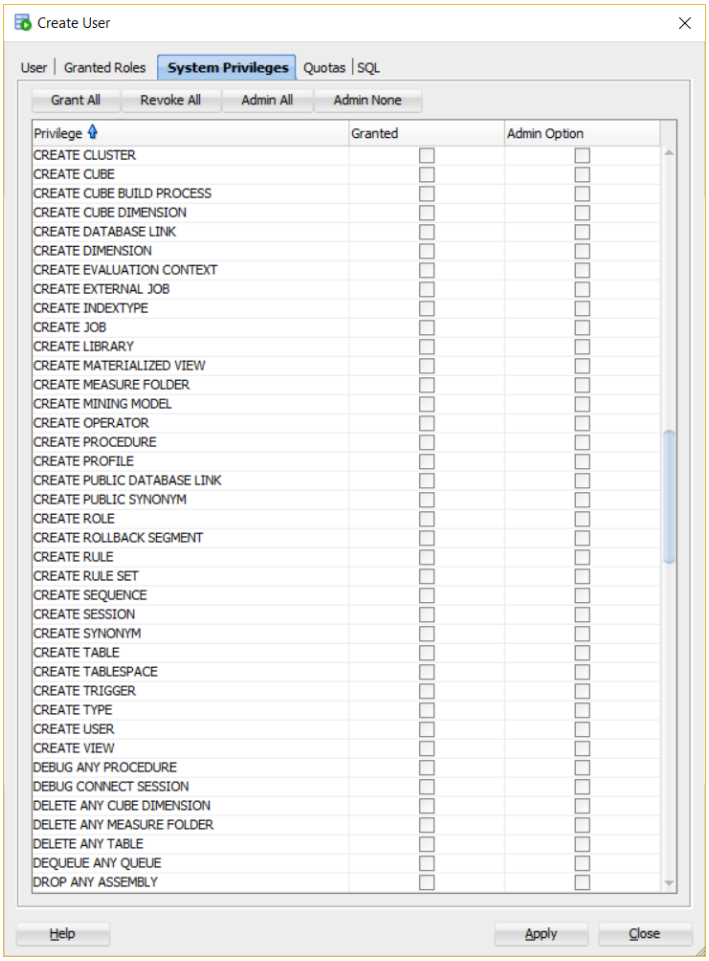
- Guidelines for granting privileges:
  - **Use principle of least privileges** – Only give users the privilege they actually need only to perform their jobs. No more.
  - **Restrict the following**:
    - The number of system object privileges granted to a user
    - The number of people given administrative SYS or SYSTEM privileges

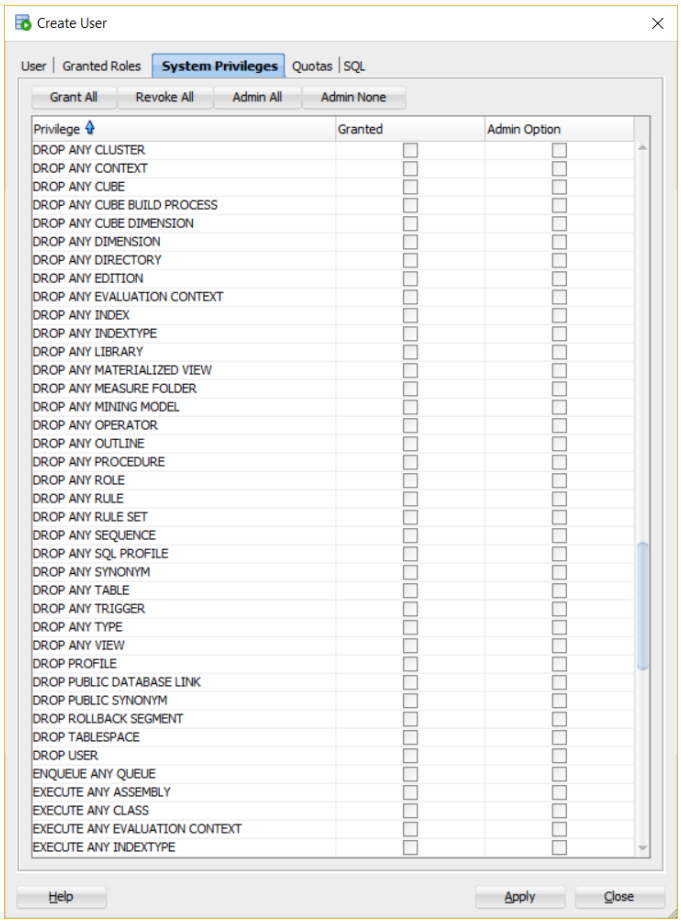
List of PRIVILEGES in Oracle 11g

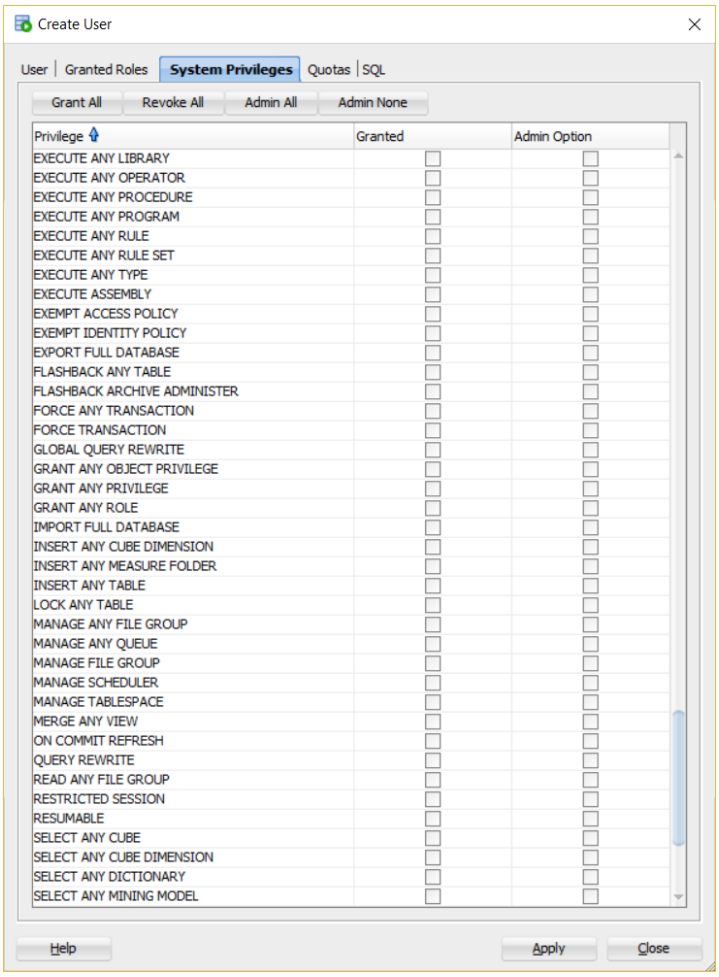
- List of PRIVILEGES in Oracle SQL Developer:

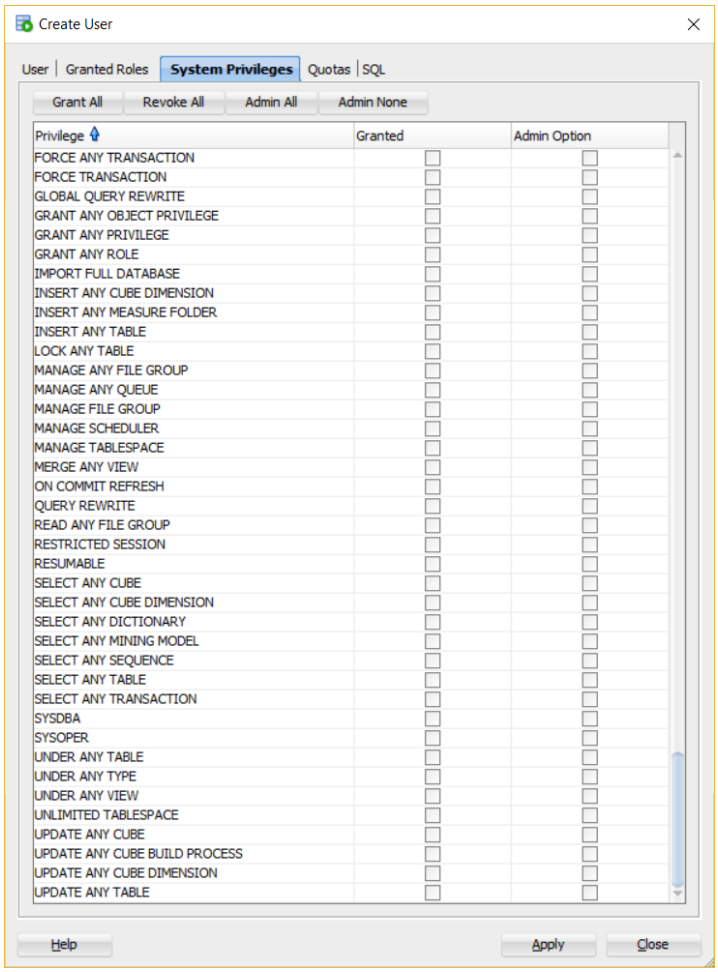





Create User
User | Granted Roles | System Privileges | Quotas | SQL
Grant All
Revoke All
Admin All
Admin None
Privilege
Granted
Admin Option
CREATE CLUSTER
CREATE CUBE
CREATE CUBE BUILD PROCESS
CREATE CUBE DIMENSION
CREATE DATABASE LINK
CREATE DIMENSION
CREATE EVALUATION CONTEXT
CREATE EXTERNAL JOB
CREATE INDEXTYPE
CREATE JOB
CREATE LIBRARY
CREATE MATERIALIZED VIEW
CREATE MEASURE FOLDER
CREATE MINING MODEL
CREATE OPERATOR
CREATE PROCEDURE
CREATE PROFILE
CREATE PUBLIC DATABASE LINK
CREATE PUBLIC SYNONYM
CREATE ROLE
CREATE ROLLBACK SEGMENT
CREATE RULE
CREATE RULE SET
CREATE SEQUENCE
CREATE SESSION
CREATE SYNONYM
CREATE TABLE
CREATE TABLESPACE
CREATE TRIGGER
CREATE TYPE
CREATE USER
CREATE VIEW
DEBUG ANY PROCEDURE
DEBUG CONNECT SESSION
DELETE ANY CUBE DIMENSION
DELETE ANY MEASURE FOLDER
DELETE ANY TABLE
DEQUEUE ANY QUEUE
DROP ANY ASSEMBLY
Help
Apply
Close


Create User
User | Granted Roles | System Privileges | Quotas | SQL
Grant All
Revoke All
Admin All
Admin None
Privilege
Granted
Admin Option
DROP ANY CLUSTER
DROP ANY CONTEXT
DROP ANY CUBE
DROP ANY CUBE BUILD PROCESS
DROP ANY CUBE DIMENSION
DROP ANY DIMENSION
DROP ANY DIRECTORY
DROP ANY EDITION
DROP ANY EVALUATION CONTEXT
DROP ANY INDEX
DROP ANY INDEXTYPE
DROP ANY LIBRARY
DROP ANY MATERIALIZED VIEW
DROP ANY MEASURE FOLDER
DROP ANY MINING MODEL
DROP ANY OPERATOR
DROP ANY OUTLINE
DROP ANY PROCEDURE
DROP ANY ROLE
DROP ANY RULE
DROP ANY RULE SET
DROP ANY SEQUENCE
DROP ANY SQL PROFILE
DROP ANY SYNONYM
DROP ANY TABLE
DROP ANY TRIGGER
DROP ANY TYPE
DROP ANY VIEW
DROP PROFILE
DROP PUBLIC DATABASE LINK
DROP PUBLIC SYNONYM
DROP ROLLBACK SEGMENT
DROP TABLESPACE
DROP USER
ENQUEUE ANY QUEUE
EXECUTE ANY ASSEMBLY
EXECUTE ANY CLASS
EXECUTE ANY EVALUATION CONTEXT
EXECUTE ANY INDEXTYPE
Help
Apply
Close


Create User
User
Granted Roles
System Privileges
Quotas
SQL
Grant All
Revoke All
Admin All
Admin None
Privilege
Granted
Admin Option
EXECUTE ANY LIBRARY
EXECUTE ANY OPERATOR
EXECUTE ANY PROCEDURE
EXECUTE ANY PROGRAM
EXECUTE ANY RULE
EXECUTE ANY RULE SET
EXECUTE ANY TYPE
EXECUTE ASSEMBLY
EXEMPT ACCESS POLICY
EXEMPT IDENTITY POLICY
EXPORT FULL DATABASE
FLASHBACK ANY TABLE
FLASHBACK ARCHIVE ADMINISTER
FORCE ANY TRANSACTION
FORCE TRANSACTION
GLOBAL QUERY REWRITE
GRANT ANY OBJECT PRIVILEGE
GRANT ANY PRIVILEGE
GRANT ANY ROLE
IMPORT FULL DATABASE
INSERT ANY CUBE DIMENSION
INSERT ANY MEASURE FOLDER
INSERT ANY TABLE
LOCK ANY TABLE
MANAGE ANY FILE GROUP
MANAGE ANY QUEUE
MANAGE FILE GROUP
MANAGE SCHEDULER
MANAGE TABLESPACE
MERGE ANY VIEW
ON COMMIT REFRESH
QUERY REWRITE
READ ANY FILE GROUP
RESTRICTED SESSION
RESUMABLE
SELECT ANY CUBE
SELECT ANY CUBE DIMENSION
SELECT ANY DICTIONARY
SELECT ANY MINING MODEL
Help
Apply
Close


Create User
User
Granted Roles
System Privileges
Quotas
SQL
Grant All
Revoke All
Admin All
Admin None
Privilege
Granted
Admin Option
FORCE ANY TRANSACTION
FORCE TRANSACTION
GLOBAL QUERY REWRITE
GRANT ANY OBJECT PRIVILEGE
GRANT ANY PRIVILEGE
GRANT ANY ROLE
IMPORT FULL DATABASE
INSERT ANY CUBE DIMENSION
INSERT ANY MEASURE FOLDER
INSERT ANY TABLE
LOCK ANY TABLE
MANAGE ANY FILE GROUP
MANAGE ANY QUEUE
MANAGE FILE GROUP
MANAGE SCHEDULER
MANAGE TABLESPACE
MERGE ANY VIEW
ON COMMIT REFRESH
QUERY REWRITE
READ ANY FILE GROUP
RESTRICTED SESSION
RESUMABLE
SELECT ANY CUBE
SELECT ANY CUBE DIMENSION
SELECT ANY DICTIONARY
SELECT ANY MINING MODEL
SELECT ANY SEQUENCE
SELECT ANY TABLE
SELECT ANY TRANSACTION
SYSDBA
SYSOPER
UNDER ANY TABLE
UNDER ANY TYPE
UNDER ANY VIEW
UNLIMITED TABLESPACE
UPDATE ANY CUBE
UPDATE ANY CUBE BUILD PROCESS
UPDATE ANY CUBE DIMENSION
UPDATE ANY TABLE
Help
Apply
Close

List of System Privileges & Description

- A system privilege is the right to perform actions on any object of a particular type.
- Objects include tables, views, materialized views, synonyms, indexes, sequences, cache groups, replication schemes and PL/SQL functions, procedures and packages.
- Only the instance administrator or a user with ADMIN privilege can grant or revoke system privilege
- List of Oracle system privileges available in Oracle 11g:

Privilege	Description
ADMIN	Enables a user to perform administrative tasks including checkpointing, backups, migration, and user creation and deletion.
ALTER ANY CACHE GROUP	Enables a user to alter any cache group in the database.
ALTER ANY INDEX	Enables a user to alter any index in the database. **Note**: There is no ALTER INDEX statement.
ALTER ANY MATERIALIZED VIEW	Enables a user to alter any materialized view in the database. **Note**: There is no ALTER MATERIALIZED VIEW statement.
ALTER ANY PROCEDURE	Enables a user to alter any PL/SQL procedure, function or package in the database.
ALTER ANY SEQUENCE	Enables a user to alter any sequence in the database. **Note**: There is no ALTER SEQUENCE statement.
ALTER ANY TABLE	Enables a user to alter any table in the database.
ALTER ANY VIEW	Enables a user to alter any view in the database. **Note**: There is no ALTER VIEW statement.
CACHE_MANAGER	Enables a user to perform operations related to cache groups.
CREATE ANY CACHE GROUP	Enables a user to create a cache group owned by any user in the database.
CREATE ANY INDEX	Enables a user to create an index on any table or materialized view in the database.
CREATE ANY MATERIALIZED VIEW	Enables a user to create a materialized view owned by any user in the database.
CREATE ANY PROCEDURE	Enables a user to create a PL/SQL procedure, function or package owned by any user in the database.
CREATE ANY SEQUENCE	Enables a user to create a sequence owned by any user in the database.
CREATE ANY SYNONYM	Enables a user to create a private synonym owned by any user in the database.
CREATE ANY TABLE	Enables a user to create a table owned by any user in the database.
CREATE ANY VIEW	Enables a user to create a view owned by any user in the database.
CREATE CACHE GROUP	Enables a user to create a cache group owned by that user.
CREATE MATERIALIZED VIEW	Enables a user to create a materialized view owned by that user.
CREATE PROCEDURE	Enables a user to create a PL/SQL procedure, function or package owned by that user.
CREATE PUBLIC SYNONYM	Enables a user to create a public synonym.
CREATE SEQUENCE	Enables a user to create a sequence owned by that user.
CREATE SESSION	Enables a user to create a connection to the database.
CREATE SYNONYM	Enables a user to create a private synonym.
CREATE TABLE	Enables a user to create a table owned by that user.
CREATE VIEW	Enables a user to create a view owned by that user.
DELETE ANY TABLE	Enables a user to delete from any table in the database.
DROP ANY CACHE GROUP	Enables a user to drop any cache group in the database.
DROP ANY INDEX	Enables a user to drop any index in the database.
DROP ANY MATERIALIZED VIEW	Enables a user to drop any materialized view in the database.
DROP ANY PROCEDURE	Enables a user to drop any PL/SQL procedure, function or package in the database.
DROP ANY SEQUENCE	Enables a user to drop any sequence in the database.
DROP ANY SYNONYM	Enables a user to drop a synonym owned by any user in the database.
DROP ANY TABLE	Enables a user to drop any table in the database.
DROP ANY VIEW	Enables a user to drop any view in the database.
DROP PUBLIC SYNONYM	Enables a user to drop a public synonym.
EXECUTE ANY PROCEDURE	Enables a user to execute any PL/SQL procedure, function or package in the database.

Privilege	Description
FLUSH ANY CACHE GROUP	Enables a user to flush any cache group in the database.
INSERT ANY TABLE	Enables a user to insert into any table in the database. It also enables the user to insert into any table using the synonym, public or private, to that table.
LOAD ANY CACHE GROUP	Enables a user to load any cache group in the database.
REFRESH ANY CACHE GROUP	Enables a user to flush any cache group in the database.
SELECT ANY SEQUENCE	Enables a user to select from any sequence or synonym on a sequence in the database.
SELECT ANY TABLE	Enables a user to select from any table, view, materialized view, or synonym in the database.
UNLOAD ANY CACHE GROUP	Enables a user to unload any cache group in the database.
UPDATE ANY TABLE	Enables a user to update any table or synonym in the database.
XLA	Enables a user to connect to a database as an XLA reader.

List of Object Privileges & Description

- An object privilege is the right to perform a particular action on an object or to access another user's object.
- Objects include tables, views, materialized views, indexes, synonyms, sequences, cache groups, replication schemes and PL/SQL functions, procedures and package
- An object's owner has all object privileges for that object, and those privileges cannot be revoked.
- The object's owner can grant object privileges for that object to other database users.
- user with ADMIN privilege can grant and revoke object privileges from users who do not own the objects on which the privileges are granted
- List of Oracle object privileges available in Oracle 11g:

Privilege	Object type	Description
DELETE	Table	Enables a user to delete from a table.
EXECUTE	PL/SQL package, procedure or function	Enables a user to execute a PL/SQL package, procedure or function directly.
FLUSH	Cache group	Enables a user to flush a cache group.
INDEX	Table or materialized view	Enables a user to create an index on a table or materialized view.
INSERT	Table or synonym	Enables a user to insert into a table or into the table through a synonym.
LOAD	Cache group	Enables a user to load a cache group.
REFERENCES	Table or materialized view	Enables a user to create a foreign key dependency on a table or materialized view. The REFERENCES privilege on a parent table implicitly grants SELECT privilege on the parent table.
REFRESH	Cache group	Enables a user to refresh a cache group.
SELECT	Table, sequence, view, materialized view, or synonym	Enables a user to select from a table, sequence, view, materialized view, or synonym. The SELECT privilege enables a user to perform all operations on a sequence. A user can be granted the SELECT privilege on a synonym or a view without being explicitly granted the SELECT privilege on the originating table.
UNLOAD	Cache group	Enables a user to unload a cache group.
UPDATE	Table	Enables a user to update a table.

Privileges Hierarchy

- Some privileges also grant or include other privileges.
- This means assigning a privilege, will also include another.
- The list below shows the privileges and what other privilege they include in Oracle 11g:

Privilege	Confers these privileges
ADMIN	All other privileges including CACHE_MANAGER
CREATE ANY INDEX	INDEX (any table or materialized view)
CREATE ANY MATERIALIZED VIEW	CREATE MATERIALIZED VIEW
CREATE ANY PROCEDURE	CREATE PROCEDURE
CREATE ANY SEQUENCE	CREATE SEQUENCE
CREATE ANY SYNONYM	CREATE SYNONYM
CREATE ANY TABLE	CREATE TABLE
CREATE ANY VIEW	CREATE VIEW
DELETE ANY TABLE	DELETE (any table)
EXECUTE ANY PROCEDURE	EXECUTE (any procedure)
INSERT ANY TABLE	INSERT (any table)
SELECT ANY SEQUENCE	SELECT (any sequence)
SELECT ANY TABLE	SELECT (any table, view or materialized view)
UPDATE ANY TABLE	UPDATE (any table)

- Two special administrative privileges required for an administrator to perform basic database operations are granted via two special system privileges:

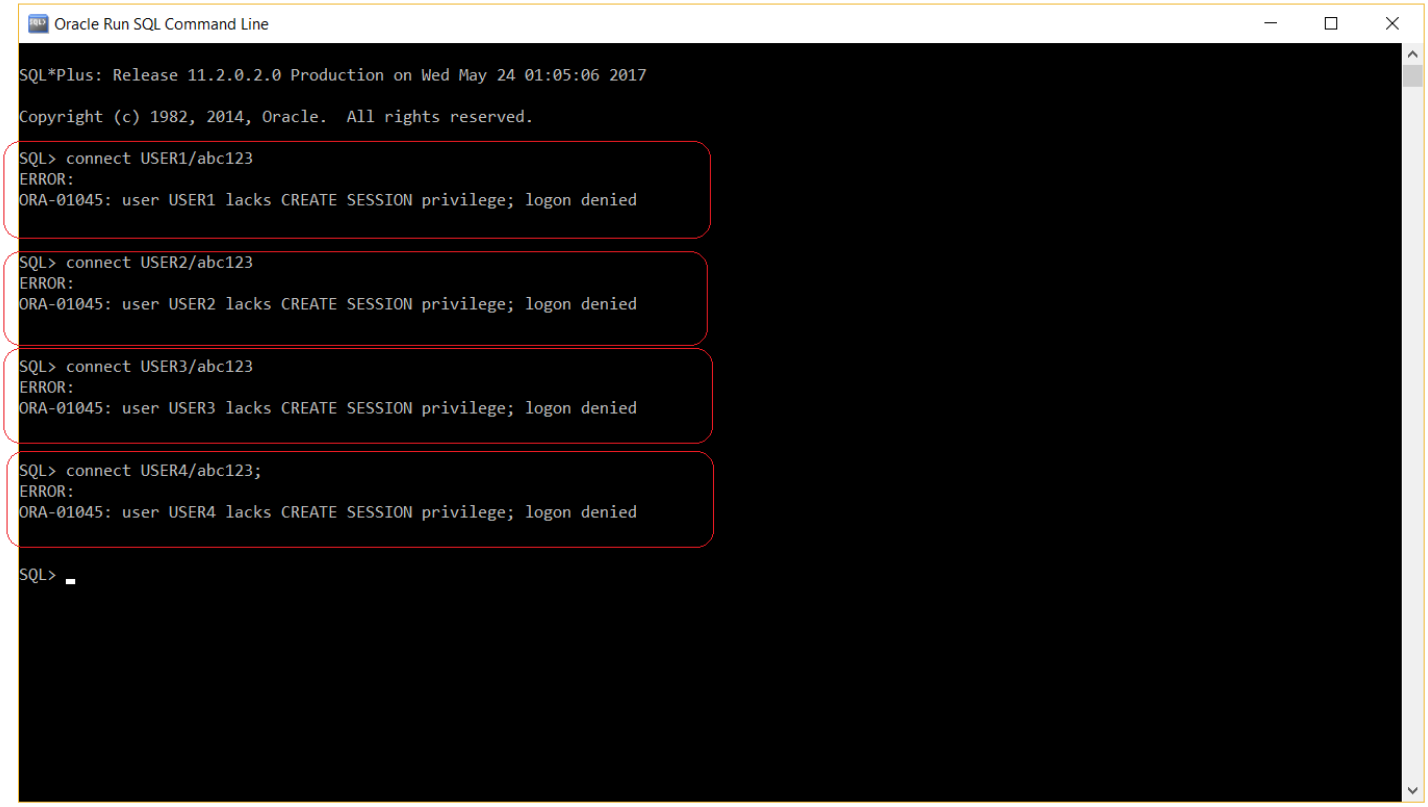
System Privilege	Operations Authorized
SYSDBA	▪ Perform STARTUP and SHUTDOWN operations ▪ ALTER DATABASE: open, mount, back up, or change character set ▪ CREATE DATABASE ▪ DROP DATABASE ▪ CREATE SPFILE ▪ ALTER DATABASE ARCHIVELOG ▪ ALTER DATABASE RECOVER ▪ Includes the RESTRICTED SESSION privilege Effectively, this system privilege allows a user to connect as user SYS.
SYSOPER	▪ Perform STARTUP and SHUTDOWN operations ▪ CREATE SPFILE ▪ ALTER DATABASE OPEN/MOUNT/BACKUP ▪ ALTER DATABASE ARCHIVELOG ▪ ALTER DATABASE RECOVER (Complete recovery only. Any form of incomplete recovery, such as UNTIL TIME|CHANGE|CANCEL|CONTROLFILE requires connecting as SYSDBA.) ▪ Includes the RESTRICTED SESSION privilege This privilege allows a user to perform basic operational tasks, but without the ability to look at user data.

(IMPORTANT) VERIFYING The Users Created Cannot Connect to the Database Because They have not been GRANTED the privilege

- When we create the users, we pointed out the following **IMPORTANT INFORMATION**:
  - At this point, we CANNOT connect to the database with any of these 4 user accounts created either with Oracle SQL Developer or SQLPlus.
  - Reason being that any of these accounts have the privileges to connect to the database.
  - In the next sections, we will discuss how to GRANT privileges to these accounts so they can connect to the database.

- We are now going to prove this by trying to connect as the newly created users to the database.

Step 1 – Attempt to connect to the database using USER1, USER2, USER3 & USER4



Step 2 – As you can see, every attempt FAILED. That is because the USER DON'T have the CREATE SESSION PRIVILEGE

Step 3 – You need to GRANT to each user the CREATE SESSION PRIVILEGE

- In the next section we will show you have to do this

GRANTING System Privileges

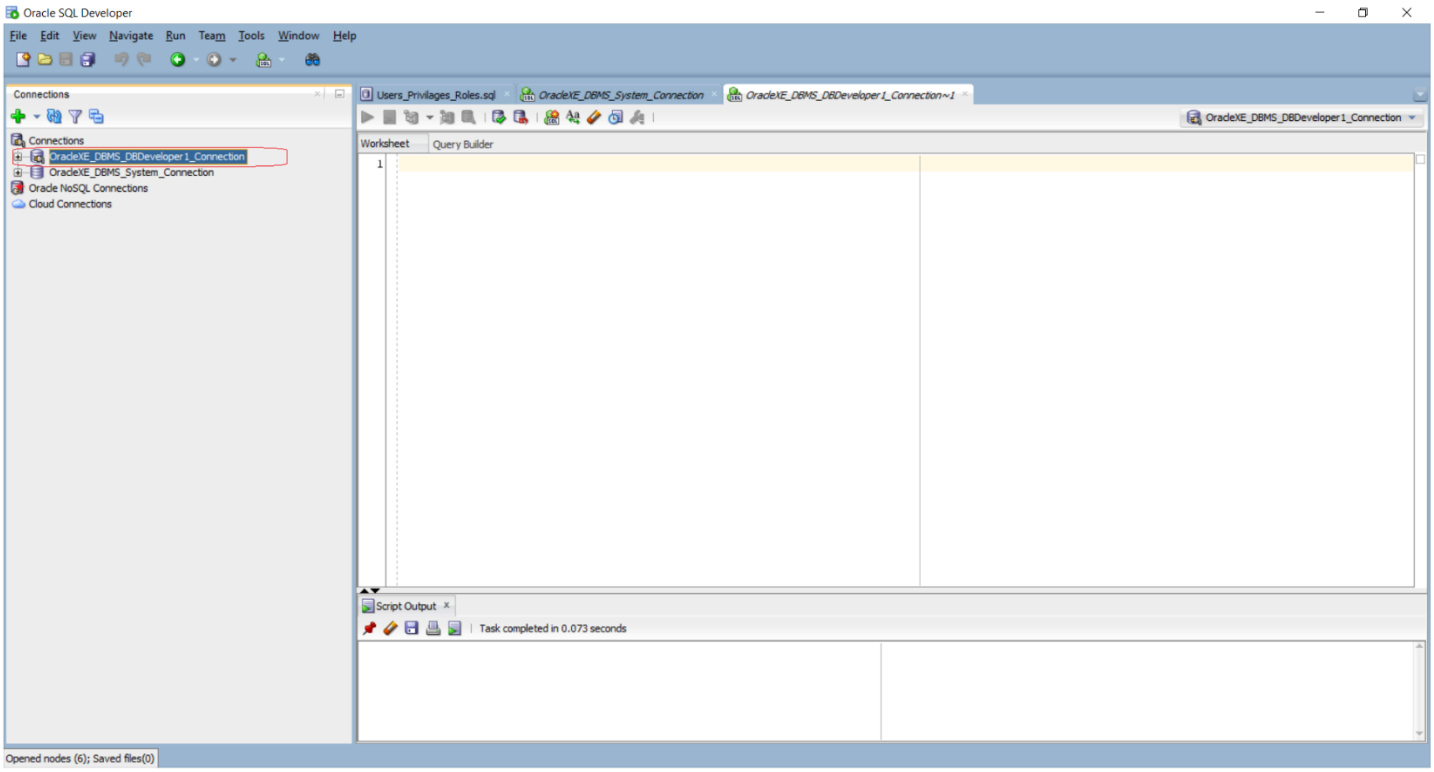
- We now look at the syntax for granting user privileges
- We can do this as follows:

1. Oracle SQL Developer graphically
2. Oracle SQL Developer via Script
3. SQLPlus SQL Command Line

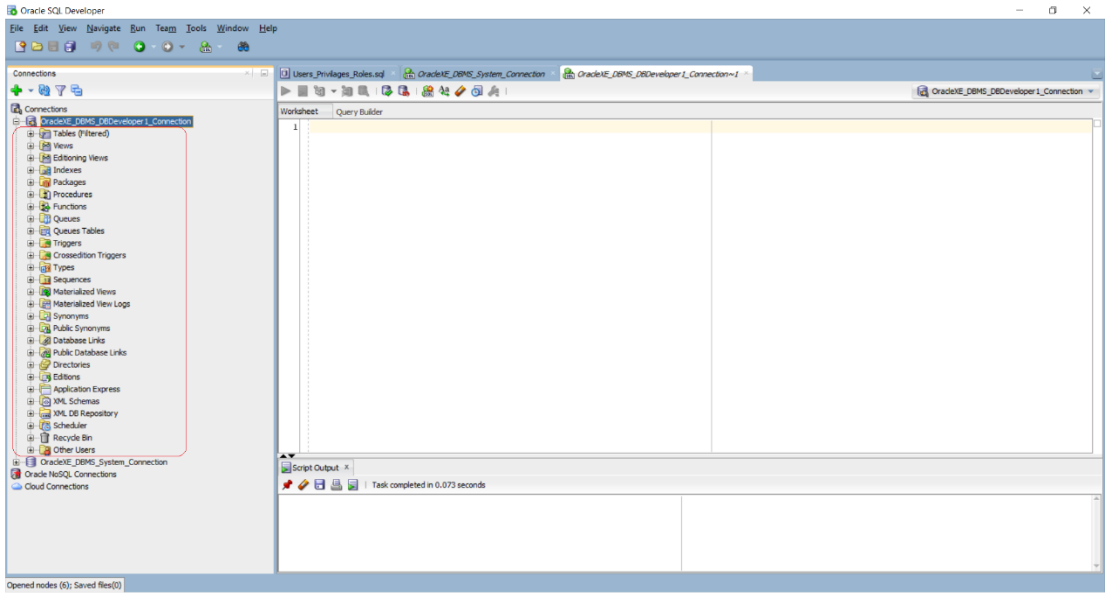
Option 1 – Granting User System Privileges Using Oracle SQL Developer Graphically

- We will grant **CREATE SESSION** privilege to **USER1**.

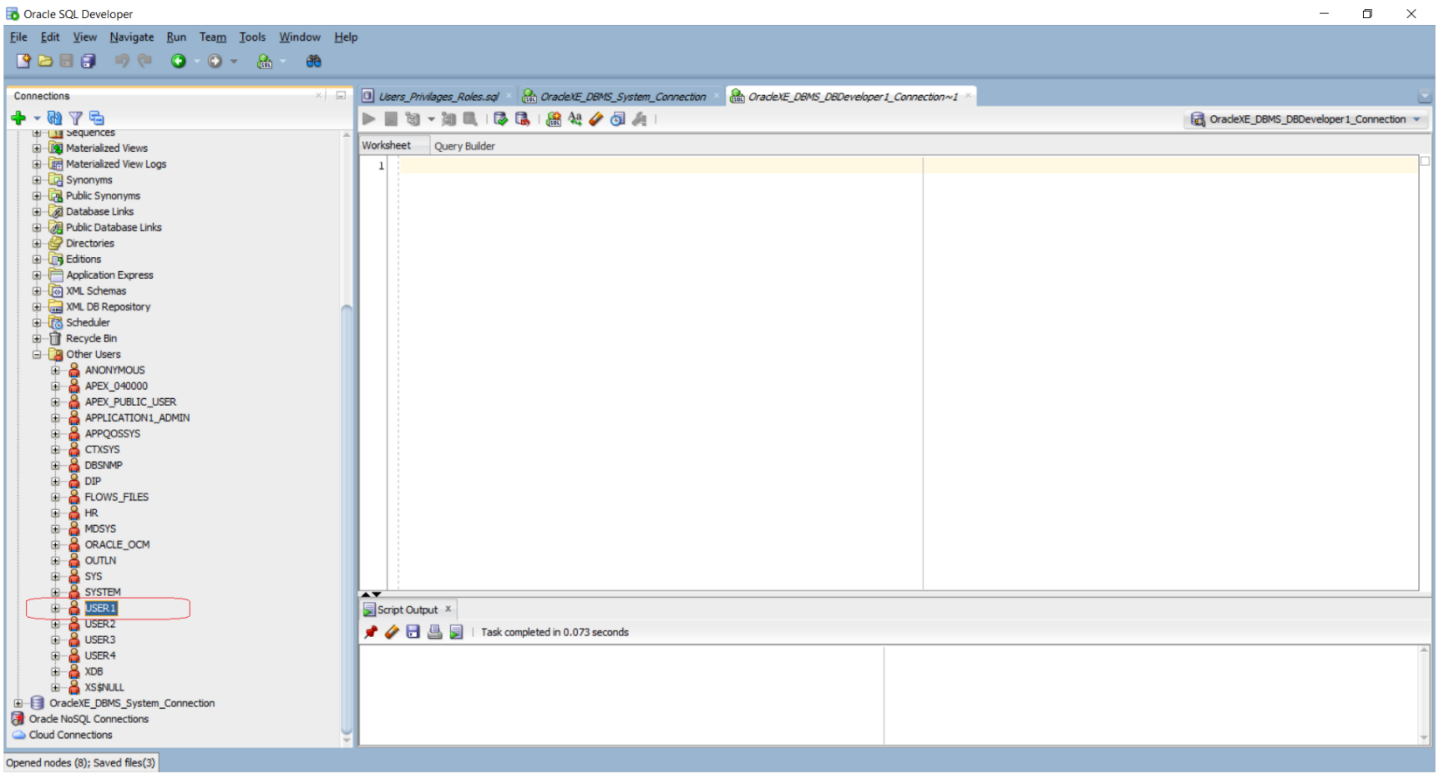
Step 1 - Log in to the schema (This case login into the DBDeveloper1 schema)



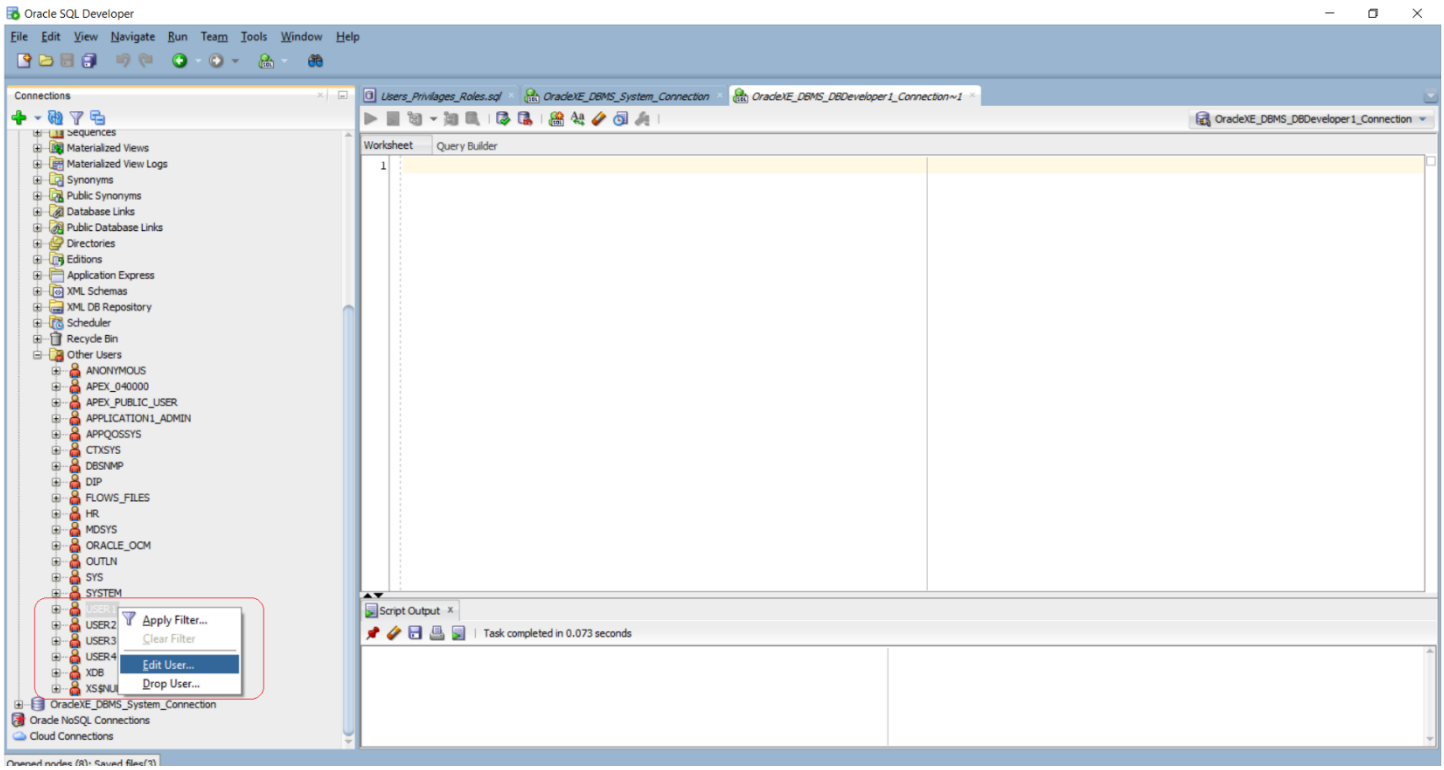
Step 2 – Expand schema to see all objects



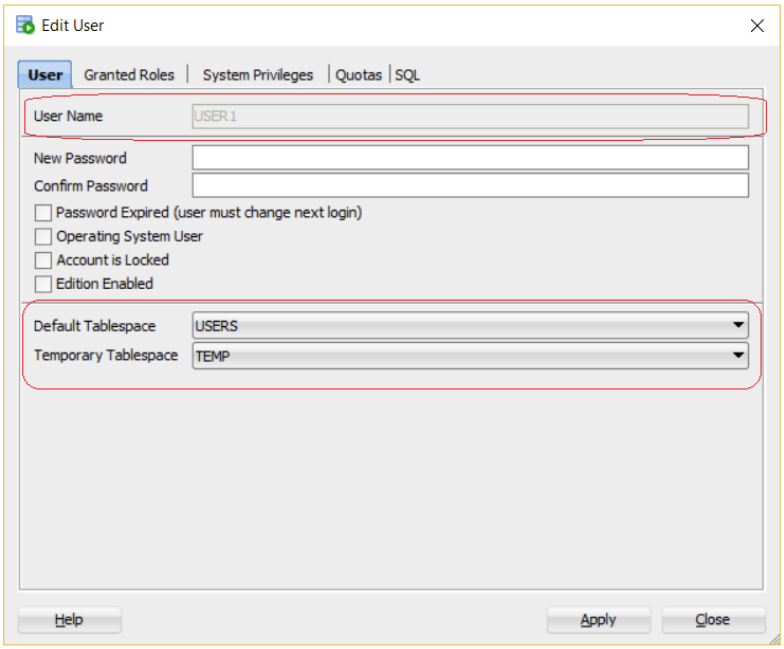
Step 3 – Expand the Other users folder & *Right-click* and select USER1



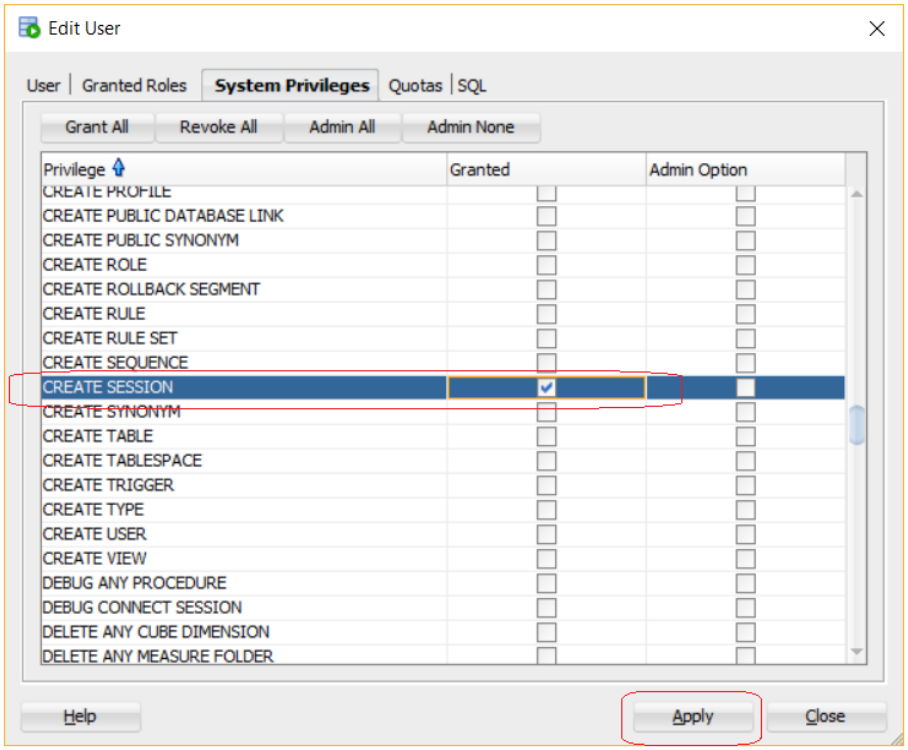
Step 4 – *Right-click* USER1 select Edit User…

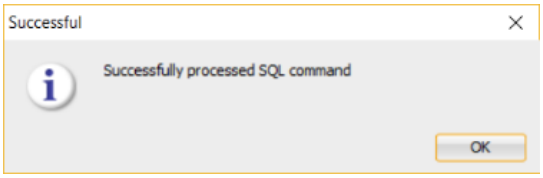


Step 5 – The Edit User screen shows the basic information selecting during creation of the user account



Step 6 – Select the System Privileges Tab and





Option 2 – Granting User System Privileges Using Oracle SQL Developer Via Script

- We will grant **CREATE SESSION** privilege to **USER2**.

Syntax for Granting User Privileges

- We will now look at some of the syntax for Granting privileges to a user:

```
-- Grant one system privilege to single user
GRANT Privilege_Name TO user;
```

```
--Grant more than one system privileges to single user in a single GRANT statement
GRANT Privilege_Name1, Privilege_Name2, Privilege_Name3, Privilege_Name(n) TO user;
```

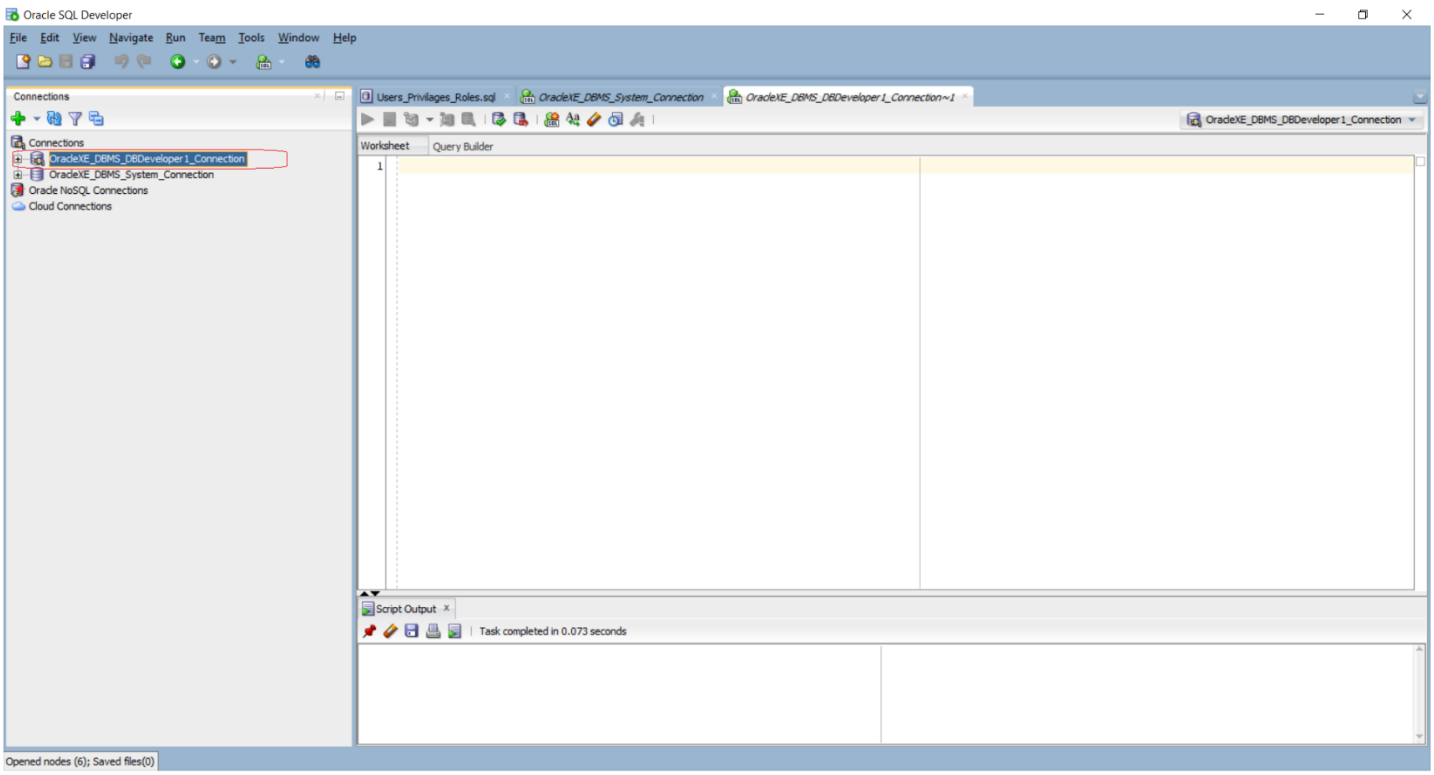
```
-- GRANT Privileges to more than one user in single GRANT statement
GRANT Privilege_Name TO user1, user2;
```

```
--Grant more than one system privileges to more than one user in a single GRANT statement
GRANT Privilege_Name1, Privilege_Name2, Privilege_Name3, Privilege_Name(n) TO user1, user2;
```

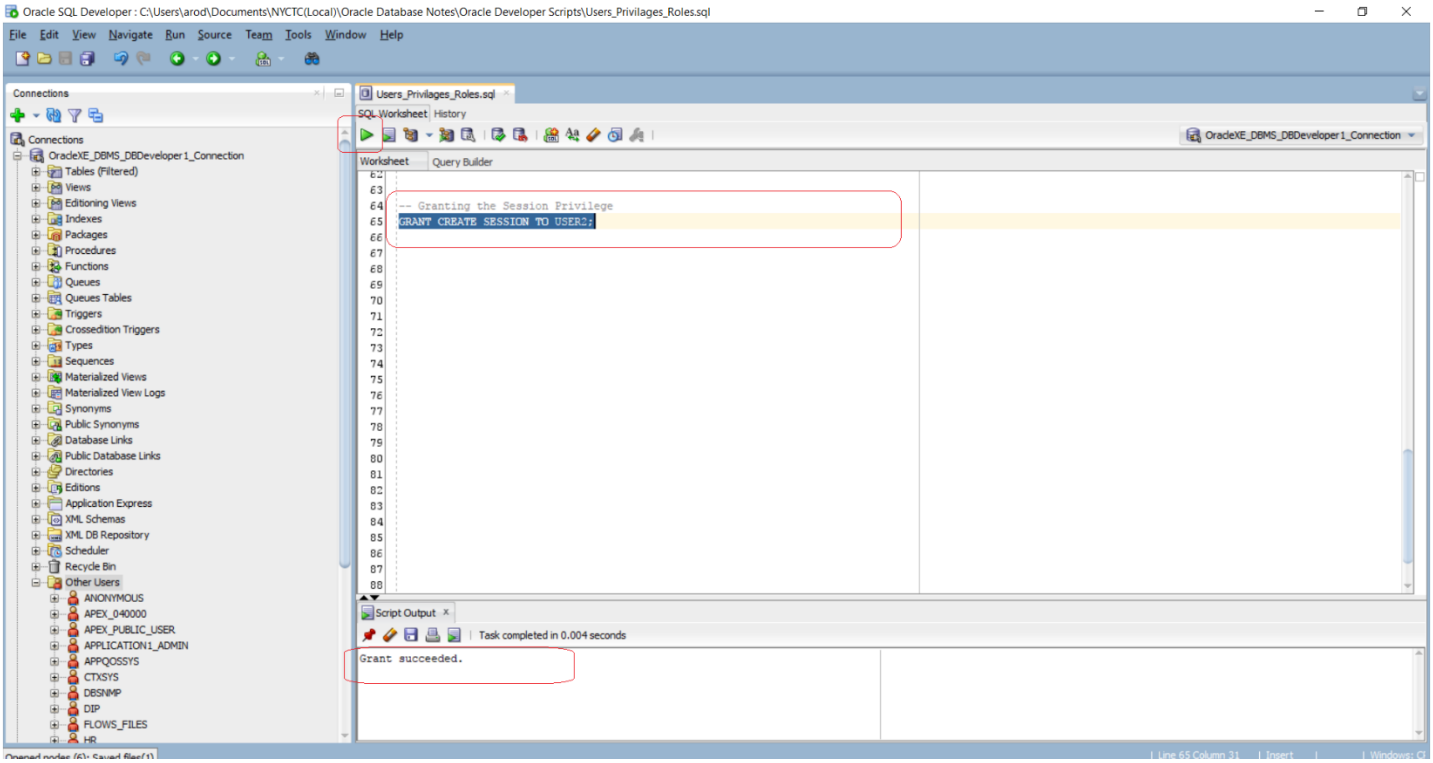
- For our example, we will GRANT the CREATE SESSION privilege to User 2 using the script command version in SQL Developer using the following command:

```
-- Grant CREATE SESSION privilege to User2
GRANT CREATE SESSION TO use2;
```

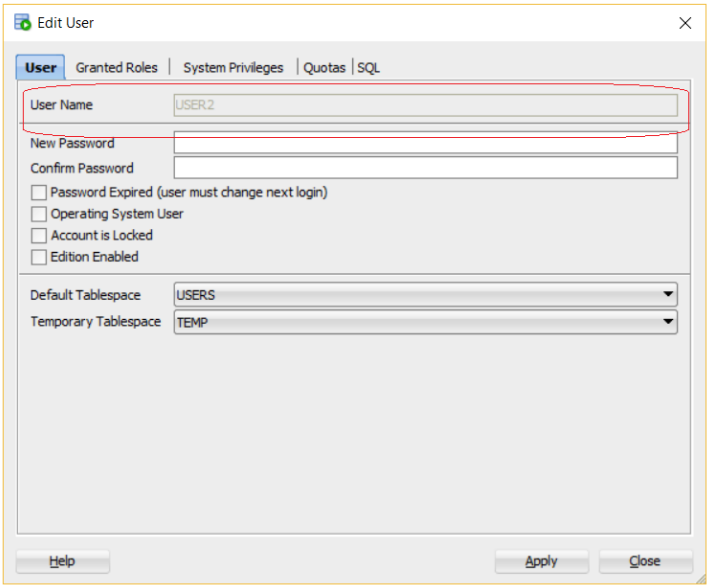
Step 1 - Log in to the schema (This case login into the DBDeveloper1 schema)

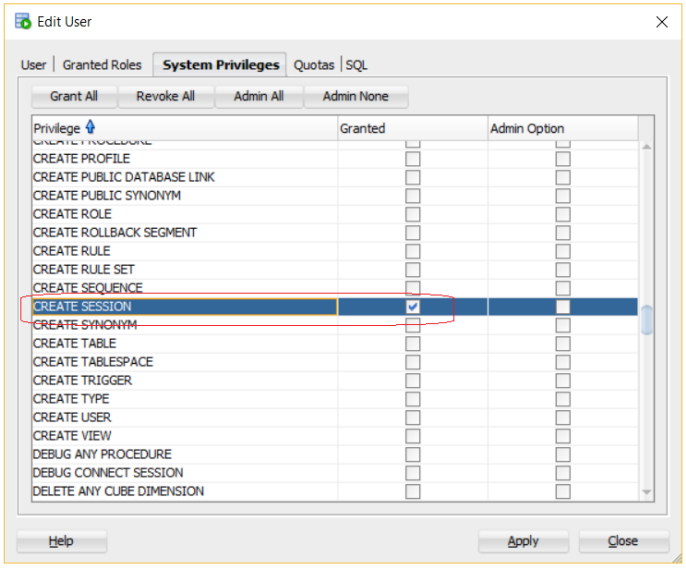


Step 2 – In the Script Windows Enter the Command & Execute



Step 4 – CONFIRM privilege was granted - *Right-click* USER2 select Edit User… & view the Edit User Window, then System Privileges and view the CREATE SESSION privilege and see that is CHECKED!





Option 3 – Granting User System Privileges Using Oracle SQLPlus

- We will grant **CREATE SESSION** privilege to **USER3 & USER4**.

Syntax for Granting User Privileges (Same as SQL Developer)

- The syntax or commands are the same in SQLPlus:

```
-- Grant one system privilege to single user
GRANT Privilege_Name TO user;
```

```
--Grant more than one system privileges to single user in a single GRANT statement
GRANT Privilege_Name1, Privilege_Name2, Privilege_Name3, Privilege_Name(n) TO user;
```

```
-- GRANT Privileges to more than one user in single GRANT statement
GRANT Privilege_Name TO user1, user2;
```

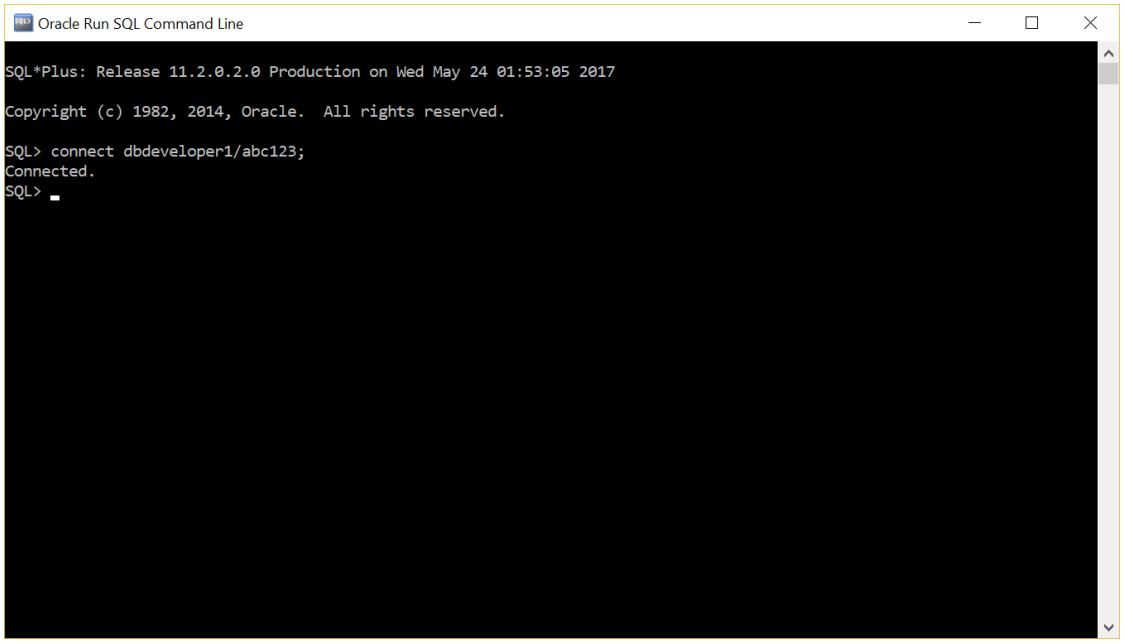
```
--Grant more than one system privileges to more than one user in a single GRANT statement
GRANT Privilege_Name1, Privilege_Name2, Privilege_Name3, Privilege_Name(n) TO user1, user2;
```

- For our example, we will GRANT the CREATE SESSION privilege to User 2 using the script command version in SQL Developer using the following command:

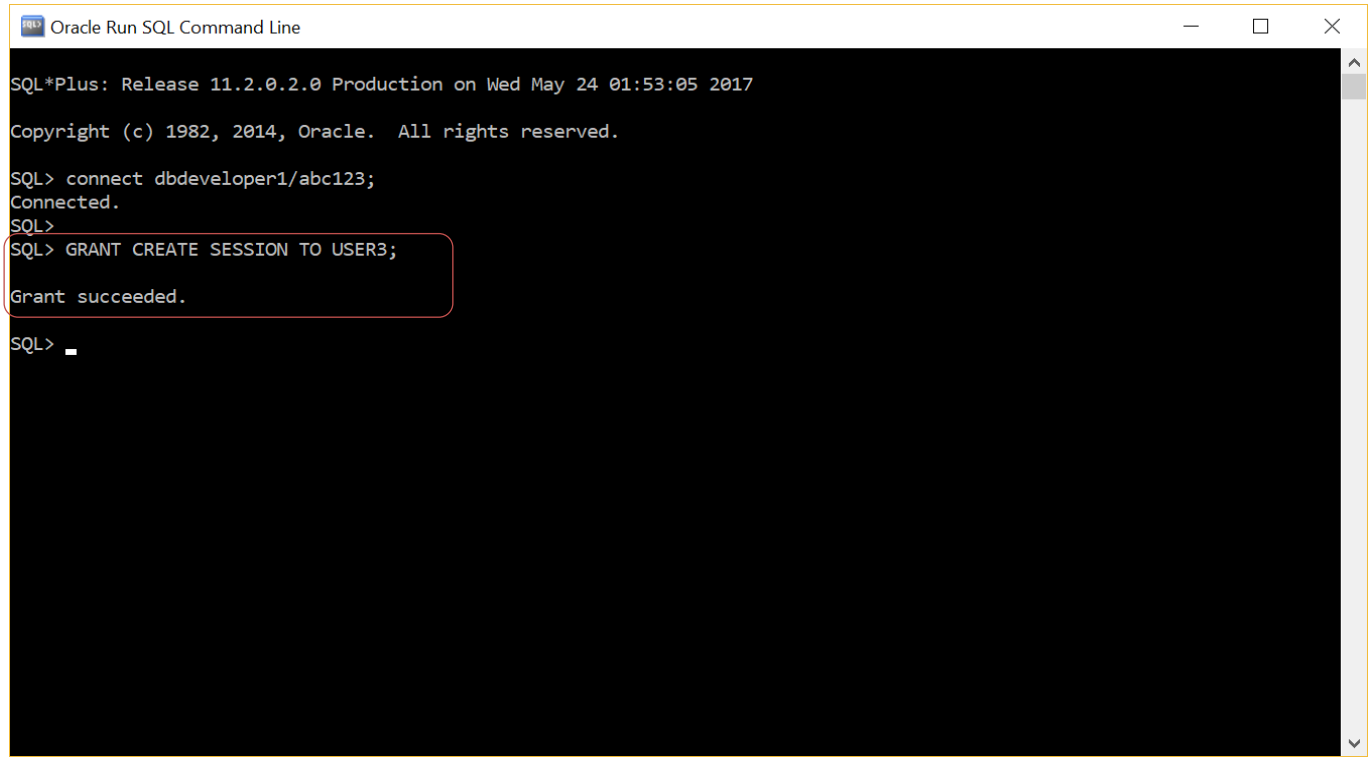
```
-- Grant CREATE SESSION privilege to User3
GRANT CREATE SESSION TO use3;
```

```
-- Grant CREATE SESSION privilege to User4
GRANT CREATE SESSION TO use4;
```

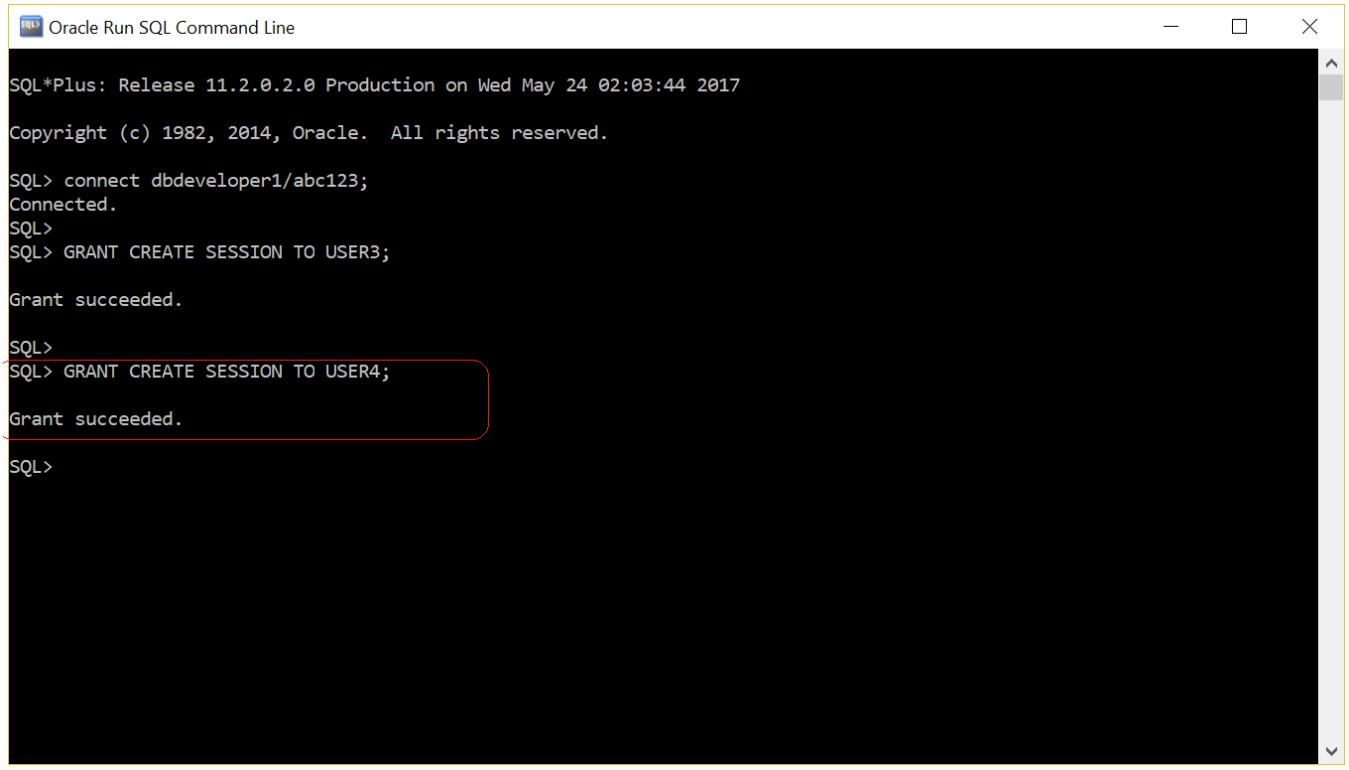
Step 1 – Connect as DBDeveloper1 Admin account to be able to GRANT privileges



Step 2 – In the Script Windows Enter the GRANT Command for USER3 & Enter



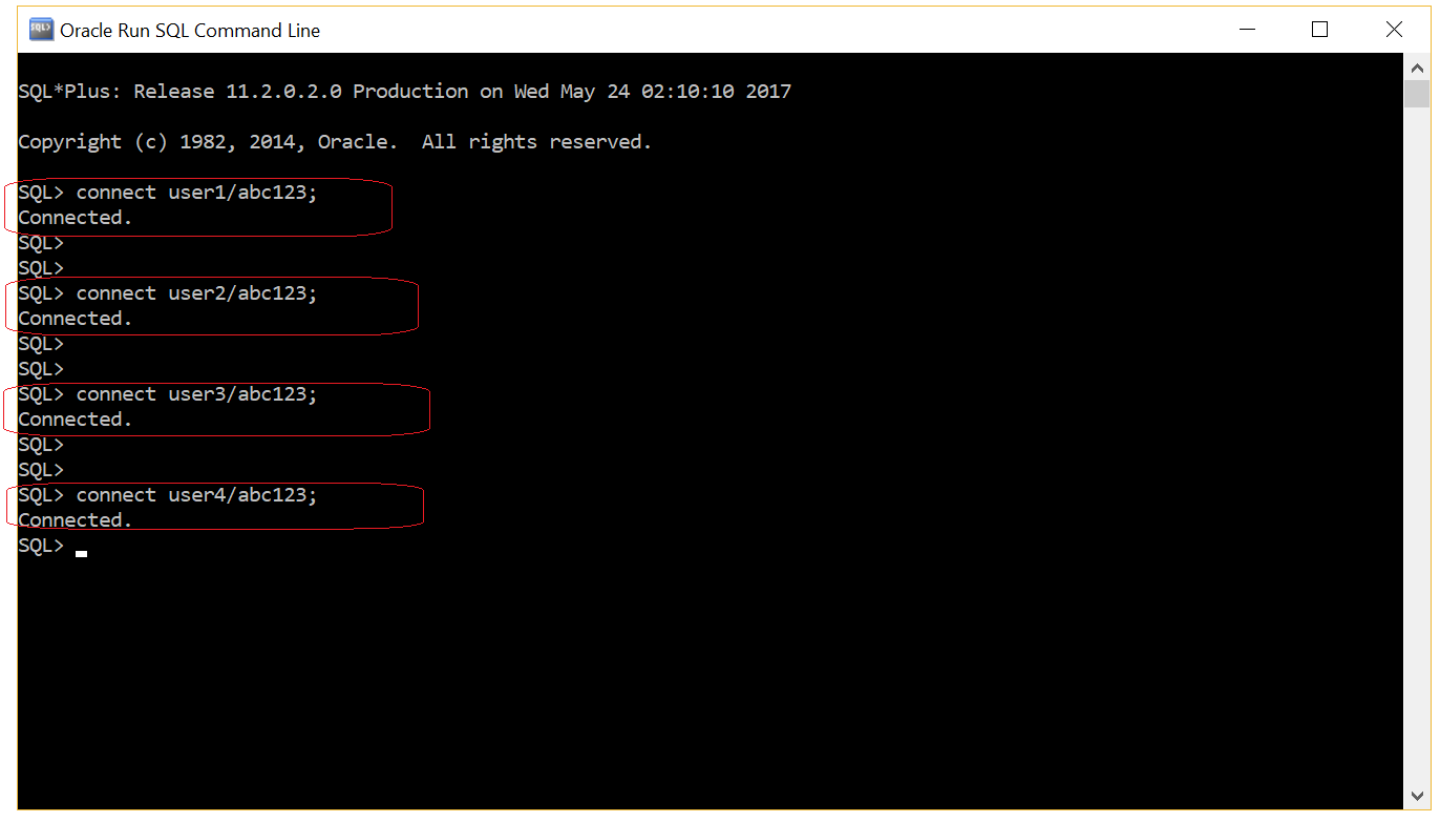
Step 3 – In the Script Windows Enter the Command for USER4 & Enter



(IMPORTANT) VERIFYING The new USERS CAN NOW CONNECT to the Database Because They have been GRANTED the privilege

- Now the users have been GRANTED the CREATE SESSION privilege so they can now connect to the database.
- **IMPORTANT:**
  - Nevertheless, although they can CONNECT, they cannot perform actions on the database objects BECAUSE THEY NEED ADDITIONAL PRIVILEGES TO BE ABLE TO PERFORM OPERATIONS ON THE DATABASE.
- We are now going to prove that we can CONNECT using all 4 accounts.

Step 1 – We now PROVE that we can successfully connect to connect to the database using USER1, USER2, USER3 & USER4



Conclusion

- **IMPORTANT:**
  - These users can now connect to the database but CANNOT PERFORM any operations such as run queries, view tables, create tables etc.
  - **BECAUSE THEY NEED ADDITIONAL PRIVILEGES TO BE ABLE TO PERFORM OPERATIONS ON THE DATABASE.**

(IMPORTANT) Attempting to Perform Actions on Database Objects with User1

- Again USER1, USER2, USER3 & USER4, can connect to the database after GRANTING the CREATE SESSION privilege BUT:
  - Cannot perform actions on the database objects BECAUSE THEY NEED ADDITIONAL PRIVILEGES TO BE ABLE TO PERFORM OPERATIONS ON THE DATABASE.

- We are now going to prove this using SQLPlus. SQLPlus is the most convenient method to test this as follows:

1. We are going to connect with USER1 to the database schema for DBDeveloper1
2. We are going to attempt to perform the following actions:
   - SELECT a record from the CUSTOMER table in DBDeveloper1 schema
   - DELETE a record from the CUSTOMER table in DBDeveloper1 schema
   - CREATE a table in DBDeveloper1 schema

Actions to test access to database objects for USER1

Test #1 – SELECT Query on CUSTOMER table

- The syntax or commands we will use is as follows:
  - **COMMAND –** Access the CUSTOMER table:

```
-- Grant OBJECT privilege to single user
SELECT * FROM DBDeveloper1.Customer WHERE customer_id ='3333';
```

Test #2 – DELETE Query on CUSTOMER table

- The syntax or commands we will use is as follows:
  - **COMMAND –** Delete the CUSTOMER table:

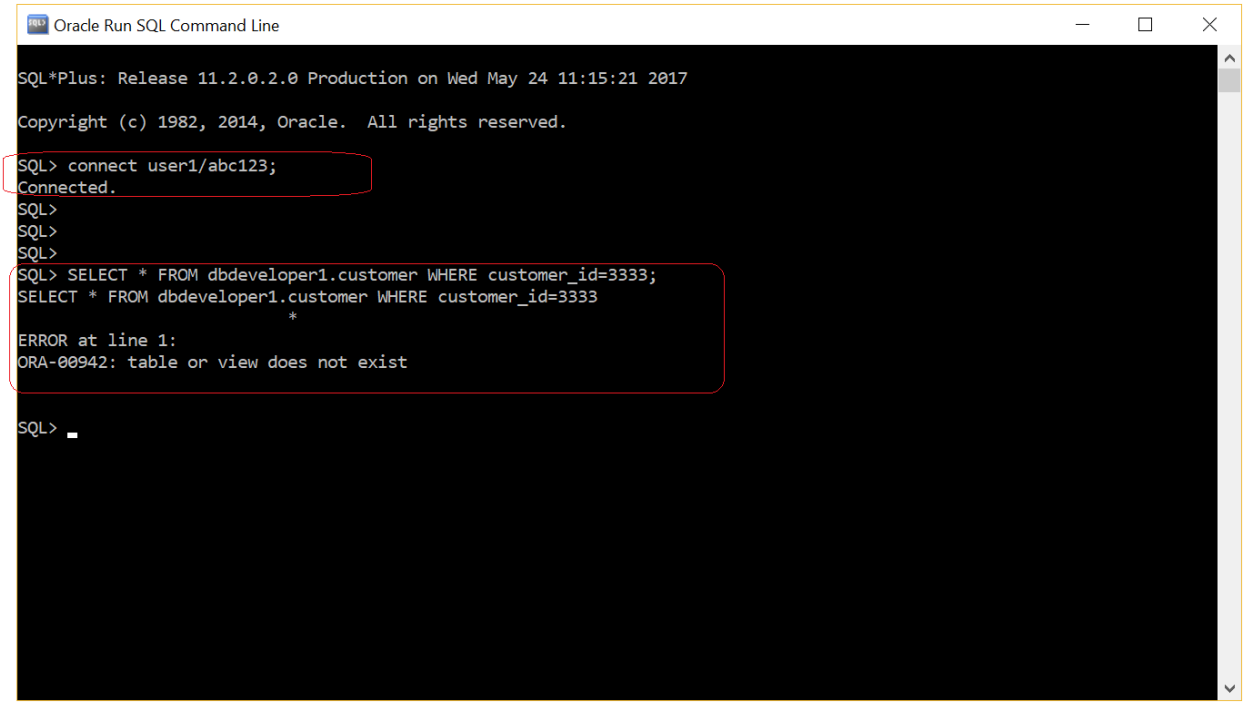
```
-- Grant OBJECT privilege to single user
DELETE FROM DBDeveloper1.Customer WHERE customer_id ='3333';
```

Test #3 – CREATE a TABLE

- The syntax or commands we will use is as follows:
  - **COMMAND –** Create a table in DBDeveloe1 schema:

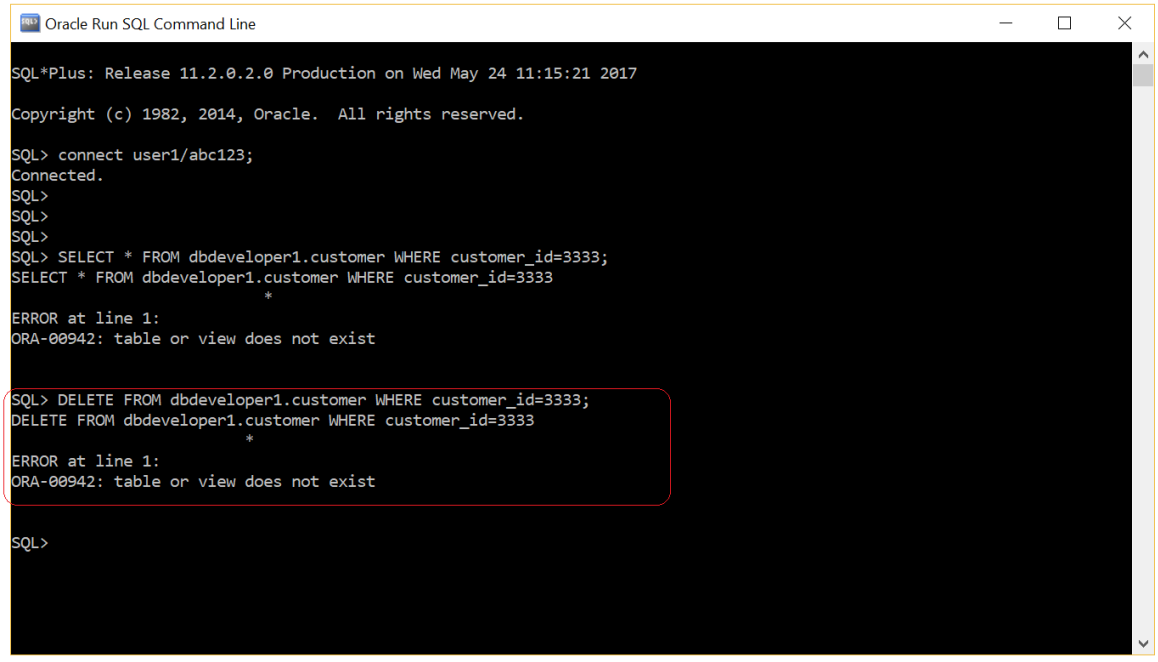
```
-- Grant OBJECT privilege to single user
CREATE TABLE DBDeveloper1.Department (Dept_ID NUMBER(4) PRIMARY KEY, Dept_Name VARCHAR2(20)
NOT NULL);
```

Step 1 – Connect as USER1 account to the database & EXECUTE a SELECT statement on the CUSTOMER TABLE



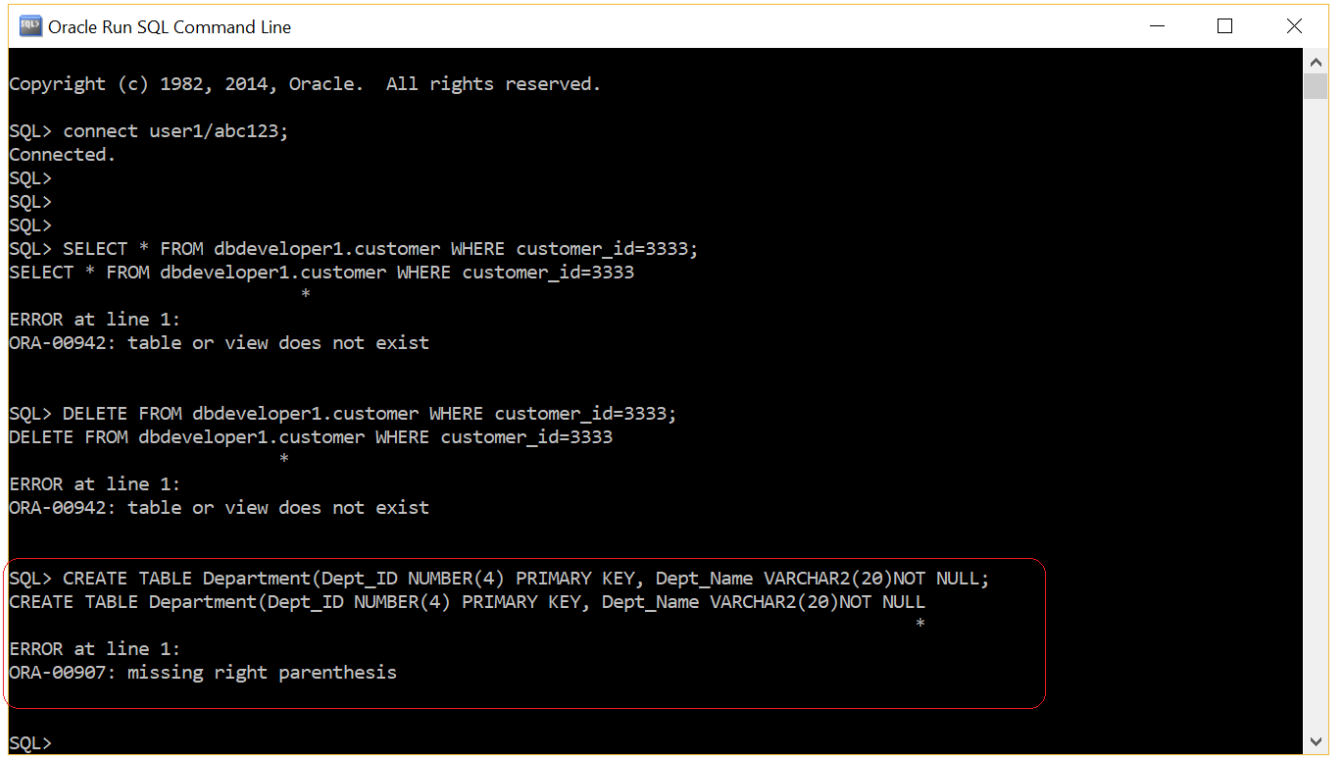
- Results:
  - **We could NOT execute the query**

Step 2 – As USER1 account, try to EXECUTE a DELETE statement on the CUSTOMER TABLE



- Results:
  - **We could NOT execute the query**

Step 3 – As USER1 account, try to EXECUTE a CREATE TABLE statement



- Results:
  - **We could NOT execute the query**

Conclusion

- **IMPORTANT:**
  - **USER 1 DOES NOT HAVE THE ADDITIONAL PRIVILEGES TO BE ABLE TO PERFORM OPERATIONS ON THE DATABASE.**

Granting ADDITIONAL Privileges to USER1 Using Oracle SQLPlus

- We will grant **CREATE SESSION** privilege to **USER3 & USER4**.
- Again USER1, USER2, USER3 & USER4, can connect to the database after GRANTING the CREATE SESSION privilege BUT:
  - Cannot perform actions on the database objects BECAUSE THEY NEED ADDITIONAL PRIVILEGES TO BE ABLE TO PERFORM OPERATIONS ON THE DATABASE.
- We are now going to GRANT the required privileges to USER1 using SQLPlus. For this you can use either ORACLE SQL DEVELOPER or SQLPLUS.
- **TO GRANT PRIVILEGES YOU MUST BE AN ADMIN so we will use DBDEVELOPER1 ACCOUNT**.
- We will grant the following privileges:

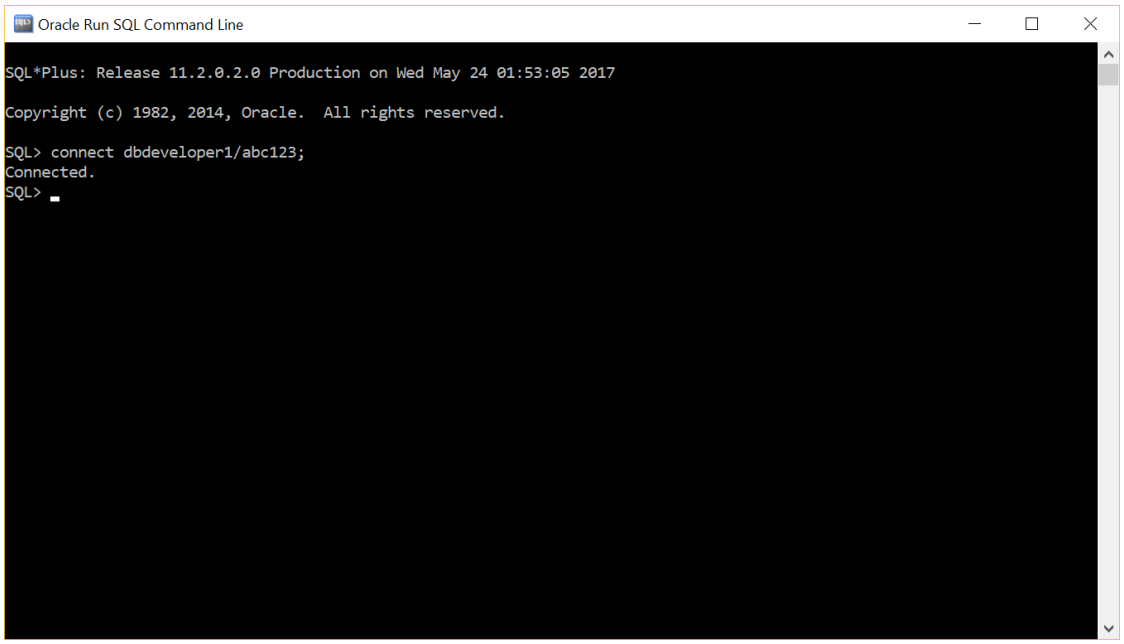
1. SELECT ON customer table
2. SELECT ANY TABLE
3. DELETE ON customer table
4. DELETE ANY TABLE
5. CREATE TABLE
6. CREATE ANY TABLE

- IMPORTANT additional steps:
  - We will have to go back and update the QUOTA to USER1 for the users tablespace.
  - I DID NOT do this when I created USER1, so I need to do now otherwise I cannot create a table since NO SPACE ALLOCATED.

Syntax for Actions to test access to database objects for USER1

- The syntax or commands we will use are as follows:

Step 1 – Connect as DBDeveloper1 Admin account to be able to GRANT privileges



GRANTING SELECT & SELECT ANY TABLE privileges

- **SYNTAX –** GRANT SELECT object privilege to access the CUSTOMER table for USER1:

```
-- Grant OBJECT privilege to single user
GRANT Object_Privilage_Name ON Owner.Object_Name TO user;
```

- **COMMAND EXAMPLE**:

```
GRANT SELECT ON DBDeveloper1.Customer TO USER1;  -- Object Privilege
```

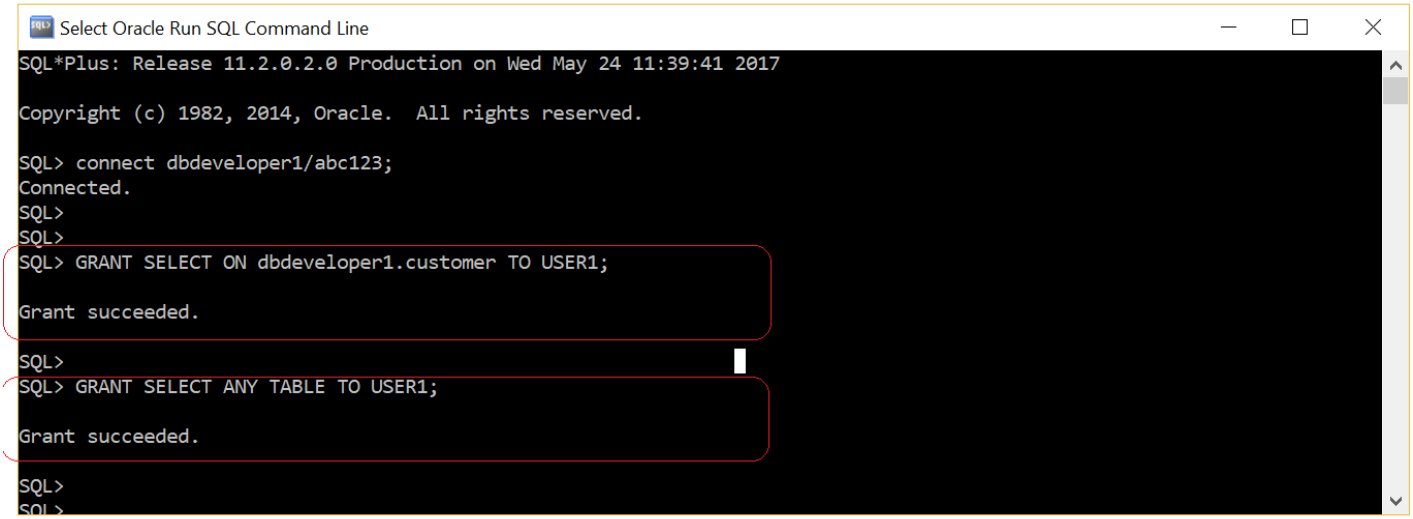
- **SYNTAX –** GRANT SELECT ANY TABLE system privilege to USER1:

```
-- Grant SYSTEM privilege to single user
GRANT System_Privilage_Name TO user;
```

- **COMMAND EXAMPLE**:

```
GRANT SELECT ANY TABLE TO USER1;  -- System Privilege
```

Step 2 – In the Script Windows Enter the GRANT Command for USER1 & Enter



GRANTING DELETE & DELETE ANY TABLE privileges

- **SYNTAX –** GRANT DELETE object privilege to delete records in the CUSTOMER table to USER1:

```
-- Grant OBJECT privilege to single user
GRANT Object_Privilage_Name ON Owner.Object_Name TO user;
```

- **COMMAND EXAMPLE**:

```
GRANT DELETE ON DBDeveloper1.Customer TO USER1;  -- Object Privilege
```

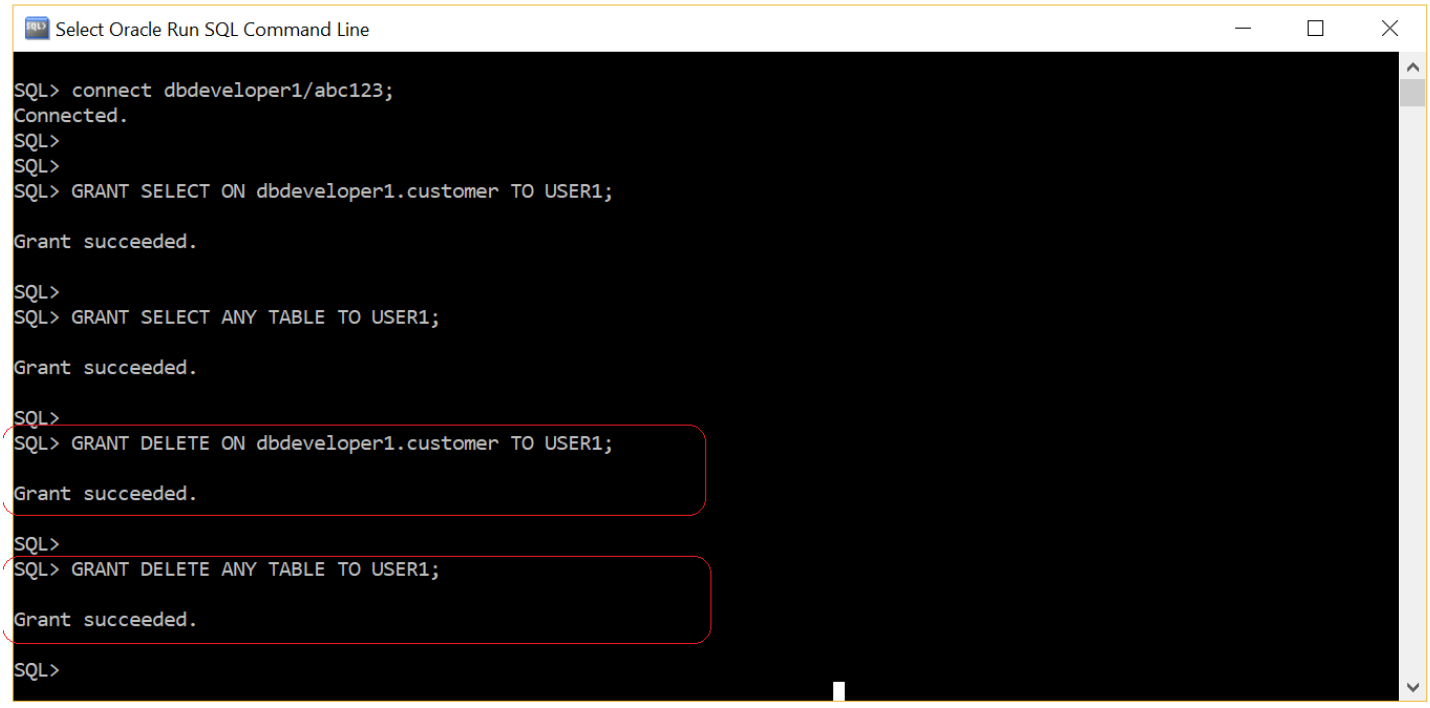
- **SYNTAX –** GRANT DELETE ANY TABLE system privilege to USER1:

```
-- Grant SYSTEM privilege to single user
GRANT System_Privilage_Name TO user;
```

- **COMMAND EXAMPLE**:

```
GRANT DELETE ANY TABLE TO USER1;  -- System Privilege
```

Step 3 – In the Script Windows Enter the NEXT GRANT Command for USER1 & Enter



GRANTING CREATE TABLE & CREATE ANY TABLE privileges

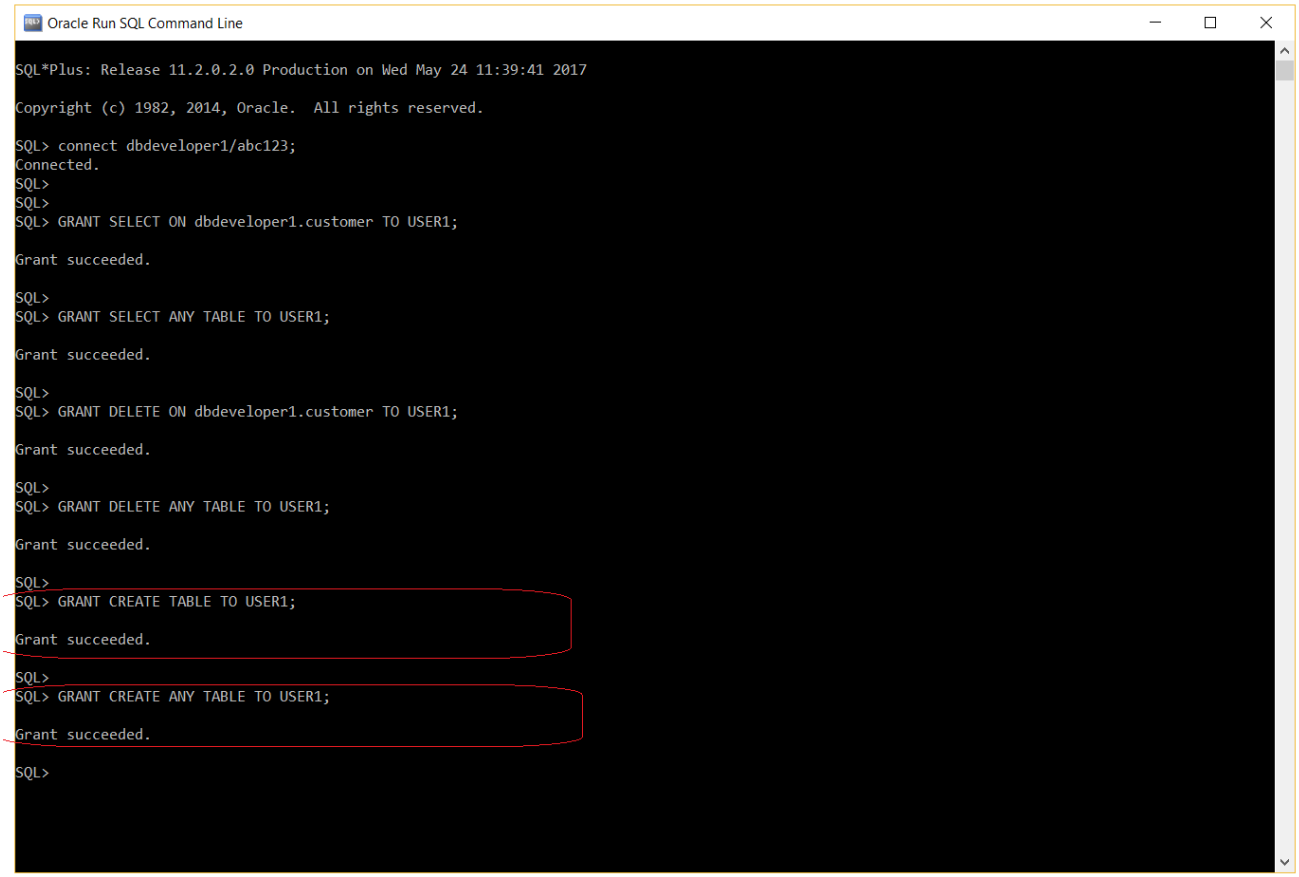
- **SYNTAX –** GRANT CREATE TABLE & CREATE ANY TABLE system privilege to USER1:

```
-- Grant SYSTEM privilege to single user
GRANT System_Privilage_Name TO user;
```

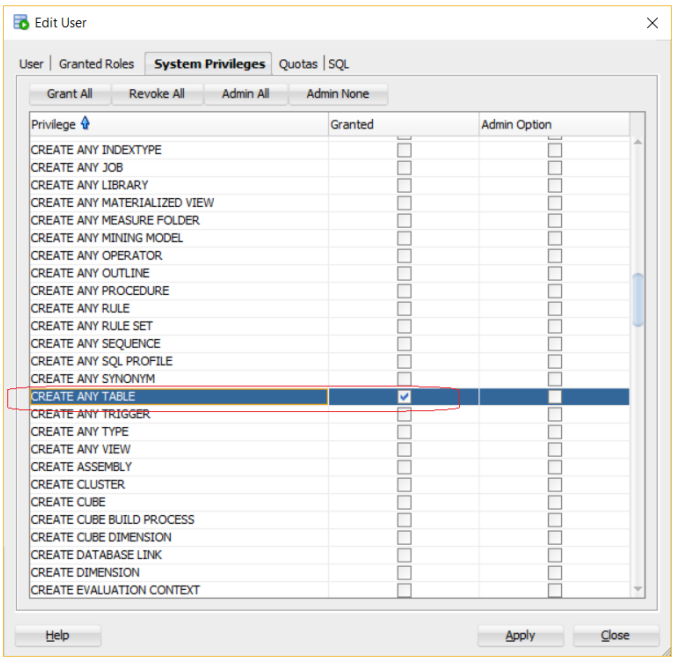
- **COMMAND EXAMPLE**:

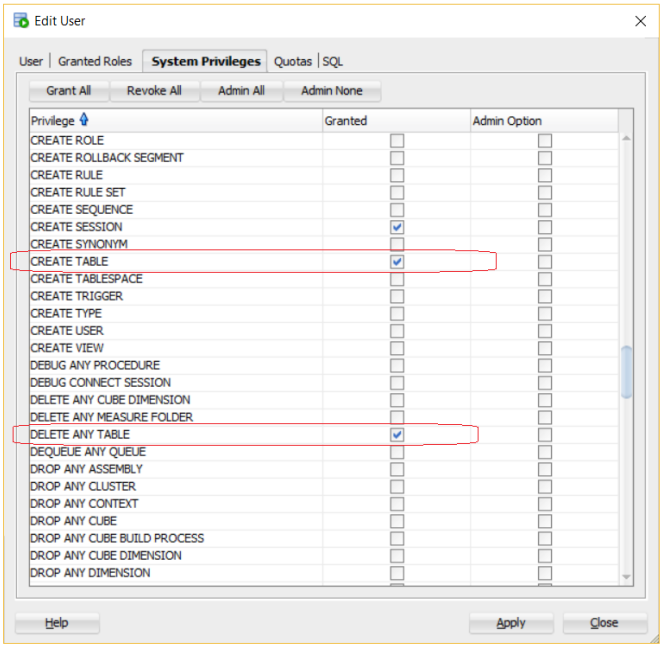
```
GRANT CREATE TABLE TO USER1;   -- System Privilege

GRANT CREATE ANY TABLE TO USER1;   -- System Privilege
```

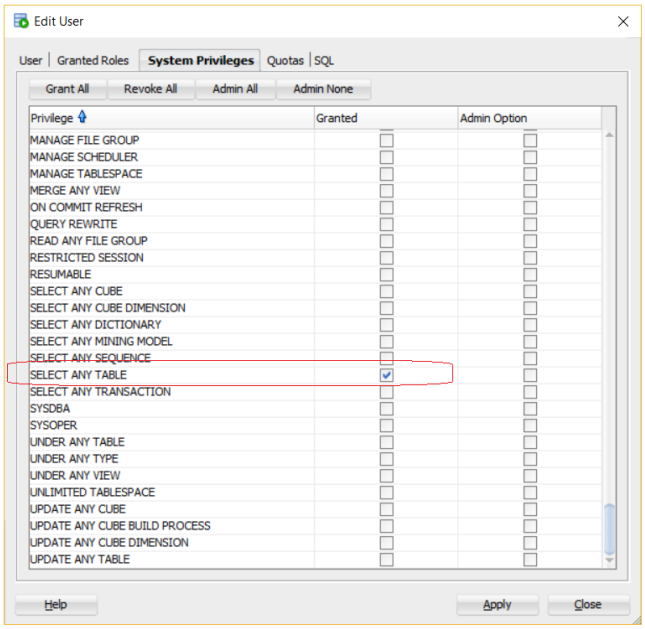
Step 4 – In the Script Windows Enter the NEXT GRANT Command for USER1 & Enter



Step 5 – Using SQL Developer verify permissions in the users edit window







(IMPORTANT) Attempting AGAIN to Perform Actions on Database Objects for User1, AFTER PRIVILEGES WERE GRANTED

- USER1, was granted the following privileges:

1. SELECT ON customer table
2. SELECT ANY TABLE
3. DELETE ON customer table
4. DELETE ANY TABLE
5. CREATE TABLE
6. CREATE ANY TABLE

- IMPORTANT additional steps we need to do before testing:

  - We will have to go back and update the QUOTA to USER1 for the users tablespace.
  - I DID NOT do this when I created USER1, so I need to do now otherwise I cannot create a table since NO SPACE ALLOCATED.

- We are now going to prove this using SQLPlus to perform the test. SQLPlus is the most convenient method to test this as follows:

1. We are going to connect with USER1 to the database schema for DBDeveloper1
2. We are going to attempt to perform the following actions:

   - SELECT a record from the CUSTOMER table in DBDeveloper1 schema
   - DELETE a record from the CUSTOMER table in DBDeveloper1 schema
   - CREATE a table in DBDeveloper1 schema

Actions to test access to database objects for USER1

Test #1 – SELECT Query on CUSTOMER table

- The syntax or commands we will use is as follows:

  - **COMMAND –** Access the CUSTOMER table:

```
-- Grant OBJECT privilege to single user
SELECT * FROM DBDeveloper1.Customer WHERE customer_id ='3333';
```

Test #2 – DELETE Query on CUSTOMER table

- The syntax or commands we will use is as follows:

  - **COMMAND –** Delete the CUSTOMER table:

```
-- Grant OBJECT privilege to single user
DELETE FROM DBDeveloper1.Customer WHERE customer_id ='3333';
```

Test #3 – CREATE a TABLE

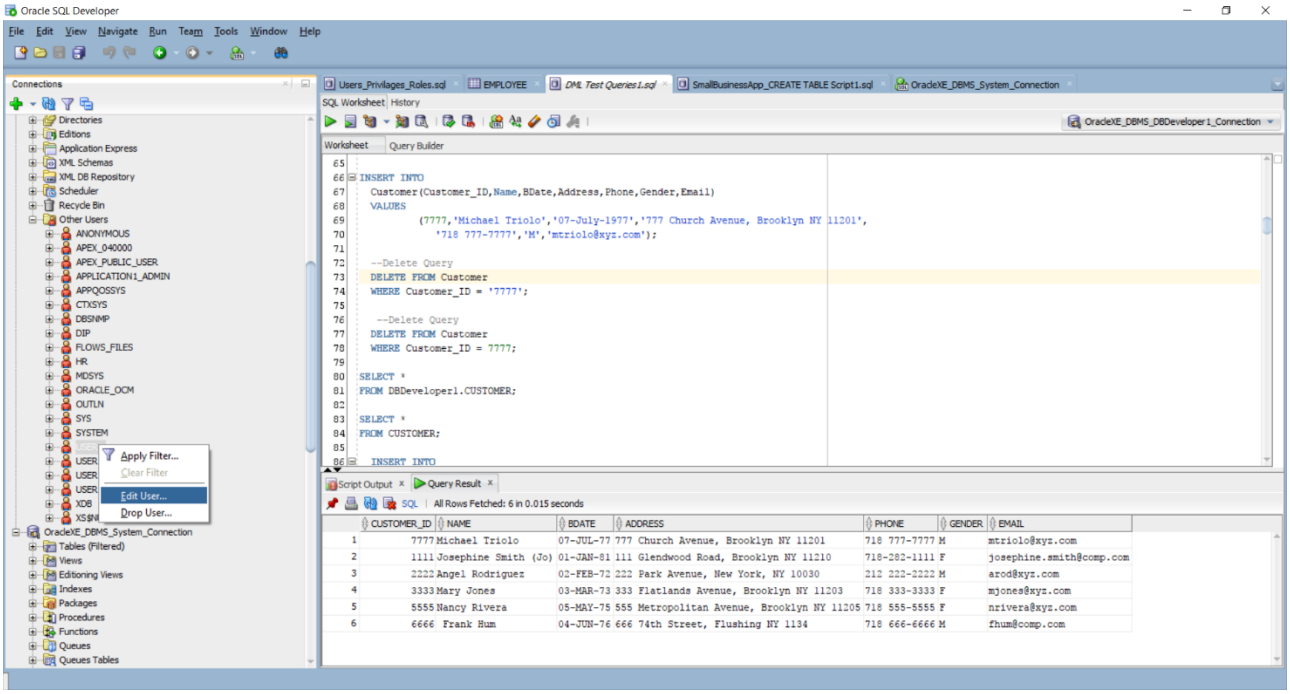
- The syntax or commands we will use is as follows:

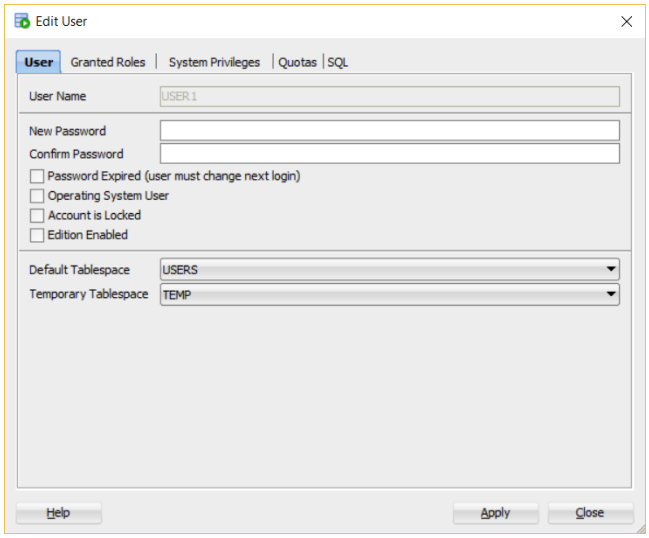
  - **COMMAND –** Create a table in DBDeveloe1 schema:

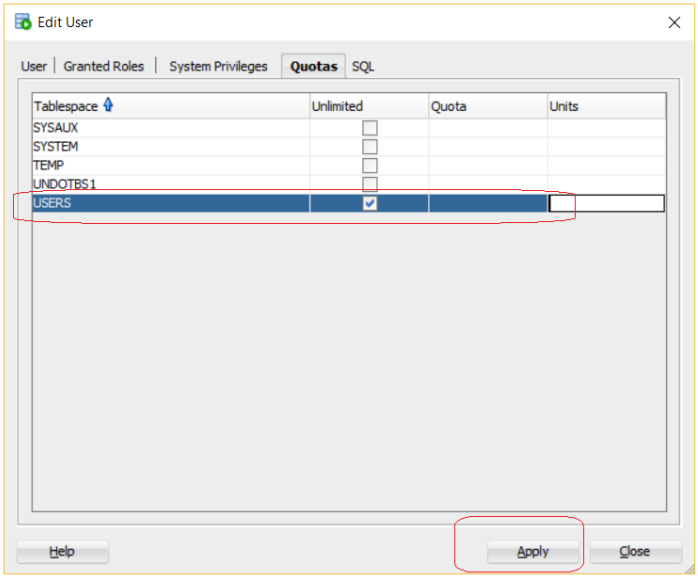
```
-- Grant OBJECT privilege to single user
CREATE TABLE DBDeveloper1.Department (Dept_ID NUMBER(4) PRIMARY KEY, Dept_Name VARCHAR2(20)
NOT NULL);
```

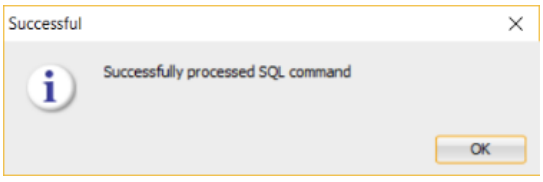
Update quota information for USER1

Step 1 – ADD QUOTA TO USER1 – You can use SQL Developer to do this



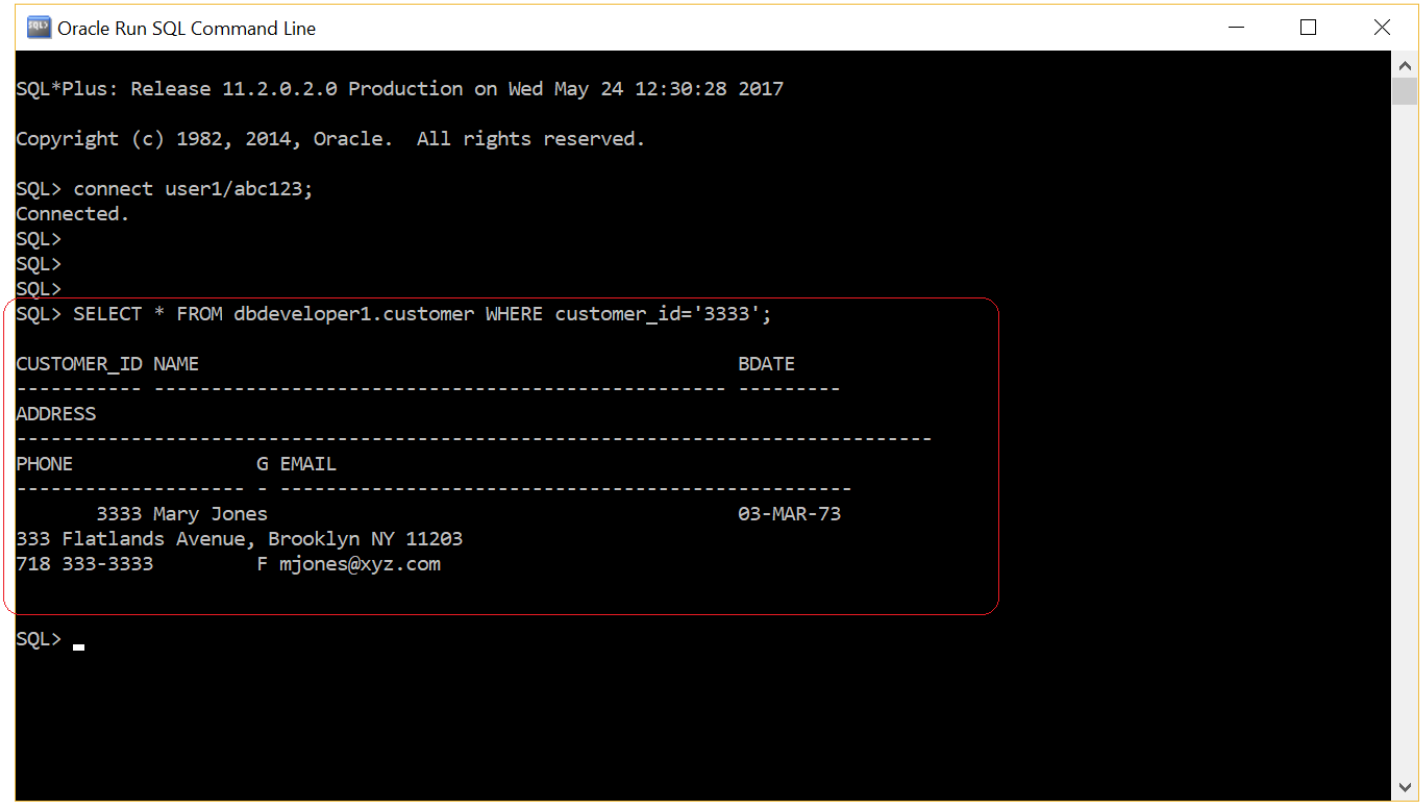






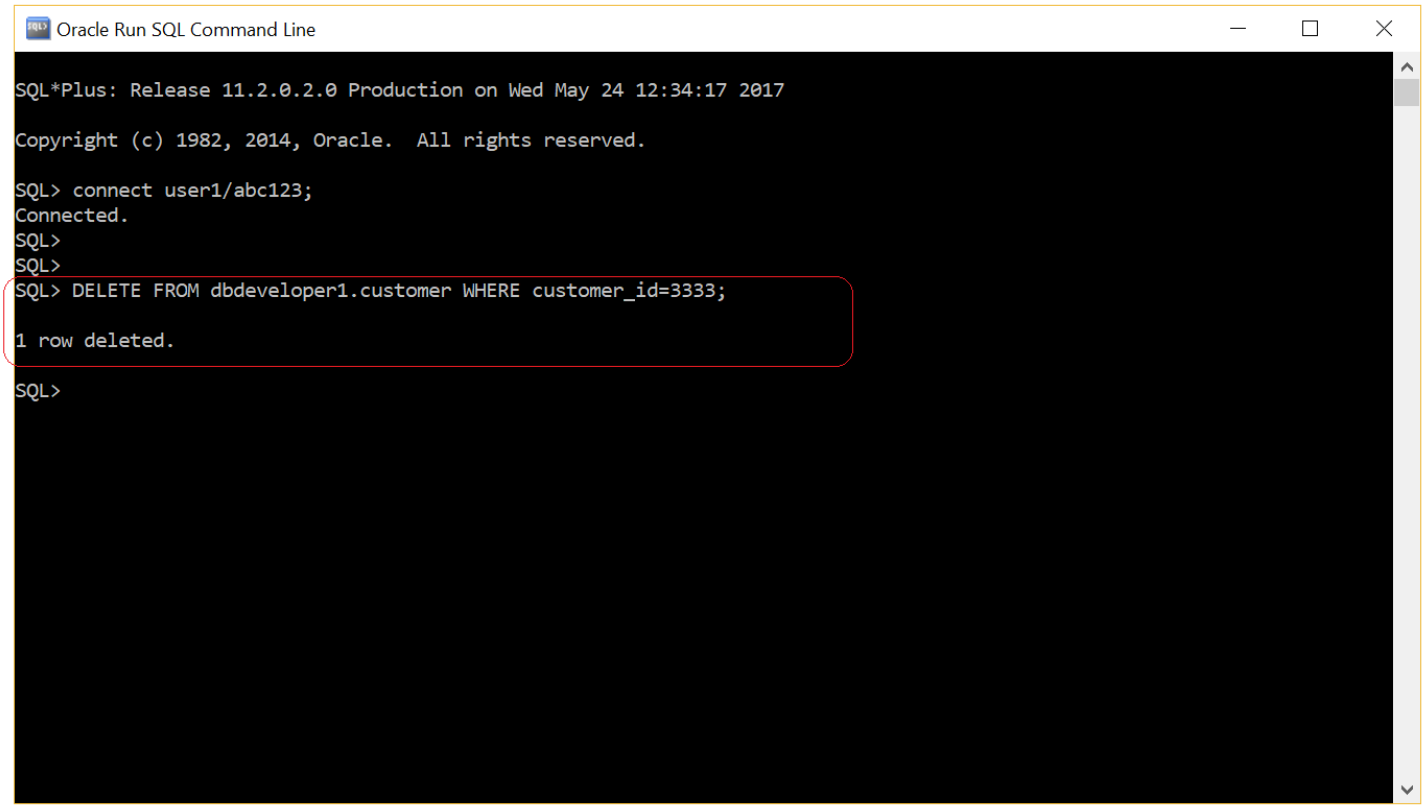
TEST THE PERMISSIONS

Step 1 – Connect as USER1 account to the database & EXECUTE a SELECT statement on the CUSTOMER TABLE



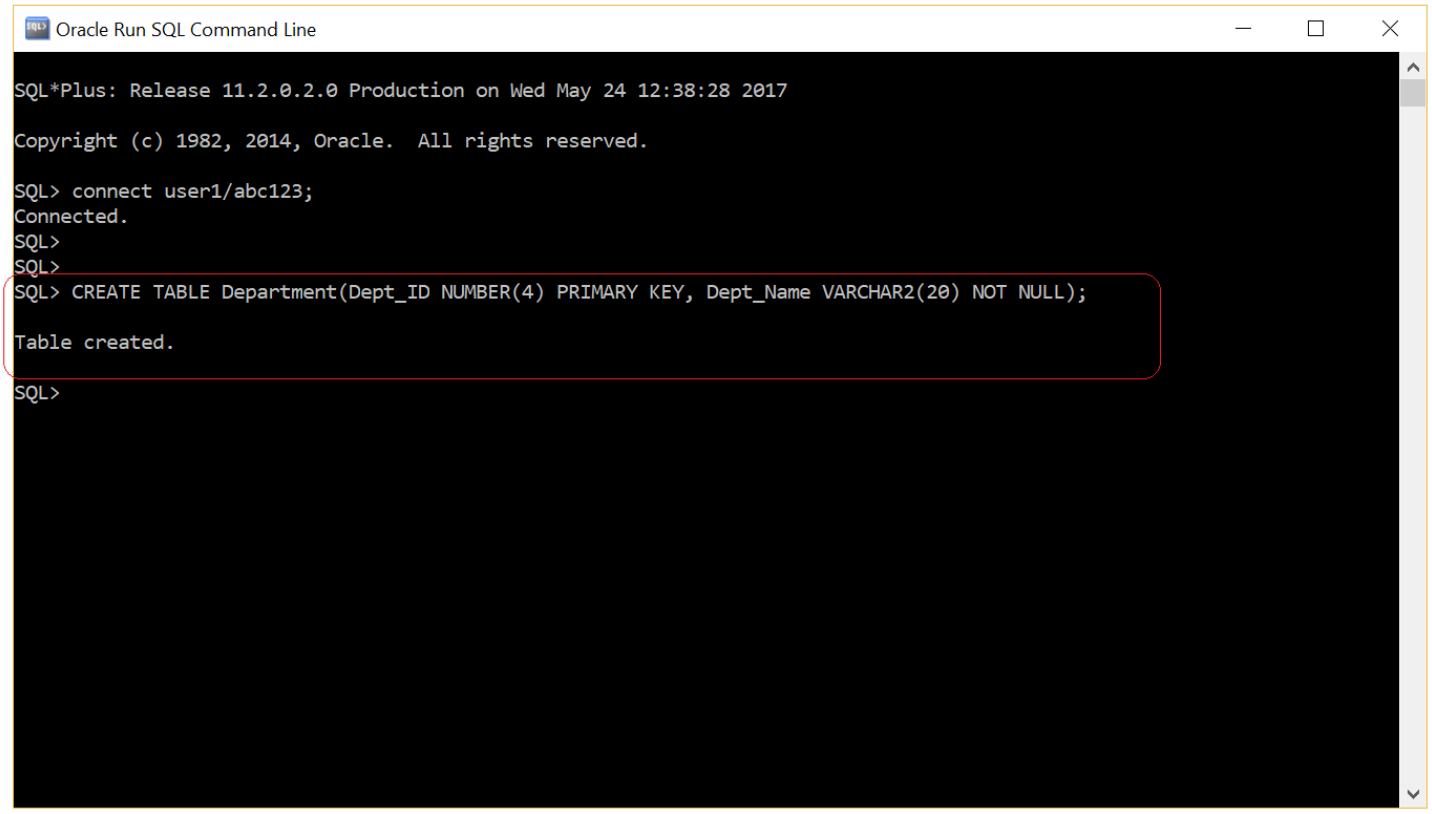
- Results:
  - **We were able to execute the query now**

Step 2 – As USER1 account, try to EXECUTE a DELETE statement on the CUSTOMER TABLE



- Results:
  - **We could execute the query**

Step 3 – As USER1 account, try to EXECUTE a CREATE TABLE statement

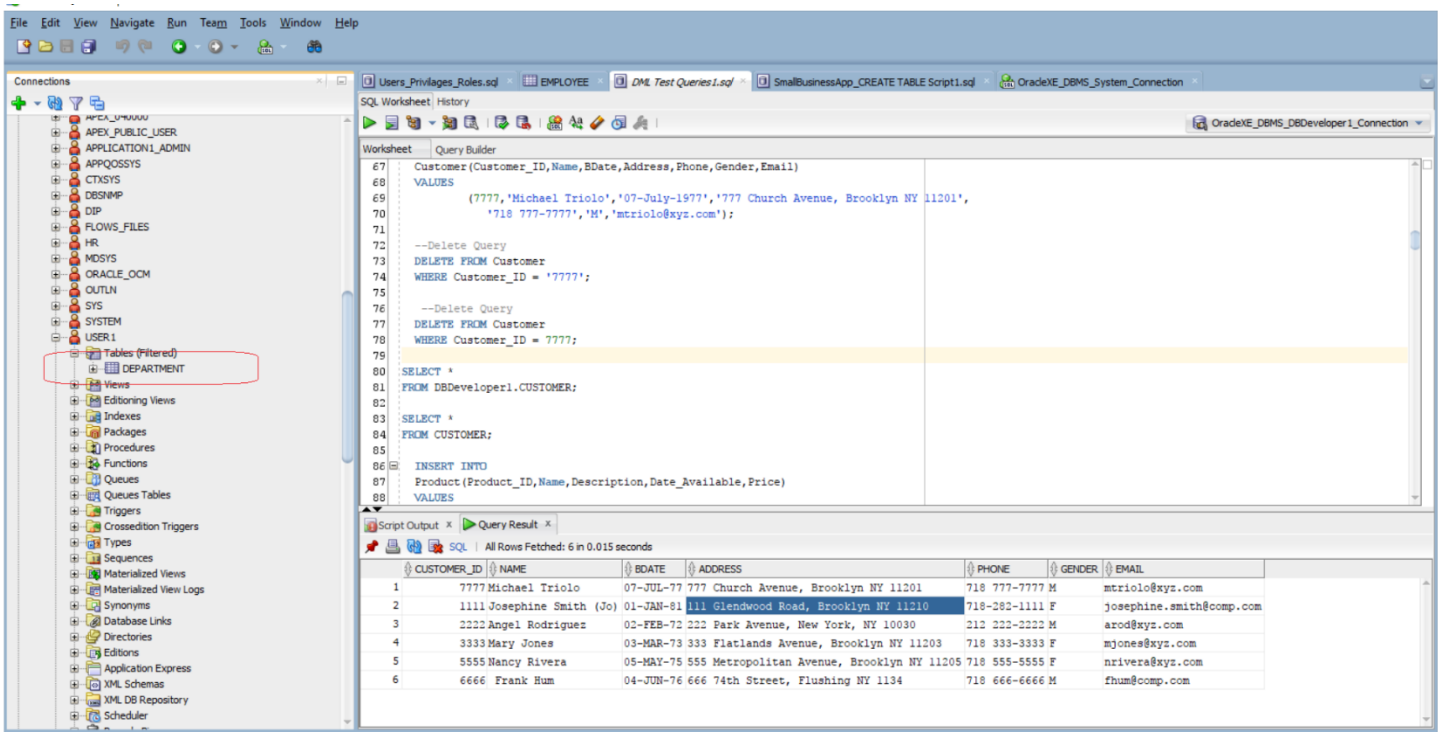


- Results:
  - **We could execute the query**

Conclusion

- **IMPORTANT:**
  - **USER 1 WAS GRANTED THE REQUIRED PRIVILEGES AND IS ABLE TO PERFORM OPERATIONS ON THE DATABASE.**

Step 4 – Verifying the TABLE was created for USER1



Roles

- Roles are a list of privileges grouped together and given a name:
  - Roles are useful to quickly and easily grant permissions to users.
  - You create roles and assign them to users.
  - A role can be combined with privileges and other existing roles.
  - There are **database-defined roles** or you can create your own **custom roles** which provide more flexibility.

- Guidelines for granting Roles:
  - **Use principle of least privileges for a Role**:
    - Create a Role with only the privilege that a user would need to perform their jobs. No more
    - If the user does not need all the privileges contained within an existing specific role, then choose another role that only has the privileges they need. Or create a new role.
    - Do not grant roles with powerful privileges such as CREATE DATABASE LINK to a regular user, only to a DBA user.

Pre-defined Roles Created in Oracle

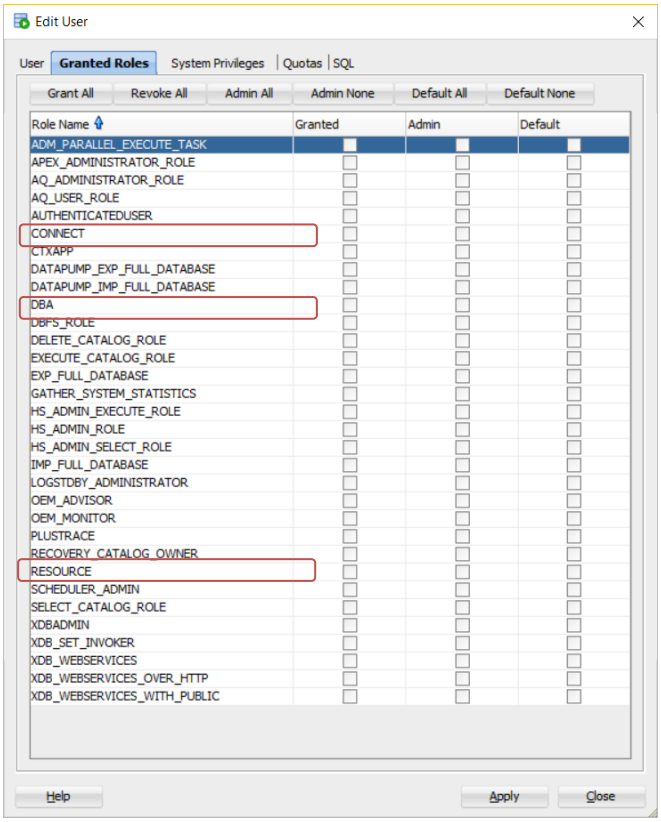
- Example of 3 roles automatically creates the following Roles by default:

Table 7–1 Oracle Database Predefined Roles

Role Name	Description
CONNECT	Enables a user to connect to the database. Grant this role to any user or application that needs database access. If you create a user using Database Control, then this role is automatically granted to the user.
RESOURCE	Enables a user to create, modify, and delete certain types of schema objects in the schema associated with that user. Grant this role only to developers and to other users that must create schema objects. This role grants a subset of the create object system privileges. For example, it grants the CREATE TABLE system privilege, but does not grant the CREATE VIEW system privilege. It grants only the following privileges: CREATE CLUSTER, CREATE INDEXTYPE, CREATE OPERATOR, CREATE PROCEDURE, CREATE SEQUENCE, CREATE TABLE, CREATE TRIGGER, CREATE TYPE.
DBA	Enables a user to perform most administrative functions, including creating users and granting privileges; creating and granting roles; creating, modifying, and deleting schema objects in any schema; and more. It grants all system privileges, but does not include the privileges to start or shut down the database instance. It is by default granted to users SYS and SYSTEM.

List of Oracle Built-in Roles

- List in Oracle SQL Developer of all the roles automatically created:



- Detailed list of roles with description:

Role Name	Description
AQ_ADMINISTRATOR_ROLE	Privilege to administer Advanced Queuing
AQ_USER_ROLE	**Deprecated**
AUTHENTICATEDUSER	DBUriServlet Security
CONNECT	Contains the create session privilege (only)
CSW_USR_ROLE	Provides user privileges to manage the Catalog Services for the Web (CSW) component of Oracle Spatial.
CTXAPP	Enables developers create Oracle Text indexes and index preferences, and to use PL/SQL packages.
CWM_USER	Provides privileges to manage Common Warehouse Metadata (CWM), which is a repository standard used by Oracle data warehousing and decision support.
DATAPUMP_EXP_FULL_DATABASE	The DATAPUMP_EXP_FULL_DATABASE role affects only Export operations. It allows users running these operations to do the following: • Perform the operation outside of the scope of their schema • Monitor jobs that were initiated by another user • Export objects (for example, TABLESPACE definitions) that unprivileged users cannot reference Although the SYS schema does not have the DATAPUMP_EXP_FULL_DATABASE role assigned to it, all security checks performed by Data Pump that require the DATAPUMP_EXP_FULL_DATABASE role will also grant access to the SYS schema.
DATAPUMP_IMP_FULL_DATABASE	This role affects only Import and SQL_FILE operations. It allows users running these operations to do the following: • Perform the operation outside of the scope of their schema • Monitor jobs that were initiated by another user • Import objects (for example, DIRECTORY definitions) that unprivileged users cannot create Although the SYS schema does not have the DATAPUMP_IMP_FULL_DATABASE role assigned to it, all security checks performed by Data Pump that require the DATAPUMP_IMP_FULL_DATABASE role will also grant access to the SYS schema.
DBA	Example Database Administrator role. Should not be used
DELETE_CATALOG_ROLE	Allow users to delete records from the system audit table (AUD$)
DMUSER_ROLE	Related to the Java API and Data Miner. In Release 1, a separate role called DMUSER_ROLE has to be created (using the script dm/admin/odmcrt.sql), and every user of the ODM Java API or Data Miner must be granted privileges on this role. This is no longer a requirement in Release 2.
DM_CATALOG_ROLE	Undocumented
EJBCLIENT	Provides privileges to connect to EJBs from a Java stored procedure.
EXECUTE_CATALOG_ROLE	Allow users EXECUTE privileges for packages and procedures in the data dictionary
EXP_FULL_DATABASE	Provides the privileges required to perform full and incremental database exports, and includes: SELECT ANY TABLE, BACKUP ANY TABLE, EXECUTE ANY PROCEDURE, EXECUTE ANY TYPE, ADMINISTER

	RESOURCE MANAGER, and INSERT, DELETE, and UPDATE on the tables SYS.INCVID, SYS.INCFIL, and SYS.INCEXP. Also the following roles: EXECUTE_CATALOG_ROLE and SELECT_CATALOG_ROLE.
GATHER_SYSTEM_STATISTICS	To update the dictionary system statistics a user must have DBA privileges or the GATHER_SYSTEM_STATISTICS role.
GLOBAL_AQ_USER_ROLE	Required to register through LDAP using JDBC connection parameters as this requires the ability to write access to the connection factory entries in the LDAP server (which requires the LDAP user to be either the database itself or be granted GLOBAL_AQ_USER_ROLE).
HS_ADMIN_ROLE	Provides privileges for DBAs who need to use the DBA role using Oracle Database Heterogeneous Services to access appropriate tables in the data dictionary. Used to protect access to the Heterogeneous Services (HS) data dictionary tables (grants SELECT) and packages (grants EXECUTE). It is granted to SELECT_CATALOG_ROLE and EXECUTE_CATALOG_ROLE such that users with generic data dictionary access also can access the HS data dictionary.
IMP_FULL_DATABASE	Provides the privileges required to perform full database imports. Includes an extensive list of system privileges (use view DBA_SYS_PRIVS to view privileges) and the following roles: EXECUTE_CATALOG_ROLE and SELECT_CATALOG_ROLE. This role is provided for convenience in using the export and import utilities.
JAVADEBUGPRIV	Grants permissions to run the Java debugger
JAVAIDPRIV	**Deprecated**
JAVASYSPRIV	Grants permissions for Java administrators including updating JVM-protected packages
JAVAUSERPRIV	Grants permissions for Java users such as examining properties
JAVA_ADMIN	Java administration privileges including permission to modify PolicyTable.
JAVA_DEPLOY	Provides privileges to deploy ncomp DLLs into the `javavm/admin` directory using the `ncomp` and `deployns` utilities. Without this role, the `javavm/deploy` and `javavm/admin` directories cannot be accessible.
JMXSERVER	Provides permissions to start and maintain a JMX agent in a session. The procedure dbms_java.start_jmx_agent starts the agent in a specific session that generally remains active for the duration of the session.
LOGSTDBY_ADMINISTRATOR	A prototype role created by default with RESOURCE, and EXECUTE on DBMS_LOGSTDBY privileges. **It is advisable to not use this role but rather to craft your own specific to your needs. Read Oracle's comments, in red with respect to RESOURCE. They apply here too.**
MGMT_USER	Provides adminstrative privileges to perform various activities with Oracle Enterprise Manager.
OEM_ADVISOR	Required to run the Segment Advisor manually with Enterprise Manager.
OEM_MONITOR	Provides privileges needed by the Management Agent component of Oracle Enterprise Manager to monitor and manage the database.
OLAPI_TRACE_USER	Provides privileges to perform OLAP API tracing. Contact Oracle Support for more information.
OLAP_DBA	To create dimensional objects in any schema
OLAP_USER	Create dimensional objects
OLAP_XS_ADMIN	Administer OLAP data security
ORDADMIN	After installing Oracle Multimedia DICOM, the ORDADMIN role is created, with the database system privileges required for administration of the DICOM data model repository.

	The ORDADMIN role must be assigned to the administrator of the DICOM data model repository.
OWB$CLIENT	Privileges granted to PUBLIC are available to all sessions.
OWB_DESIGNCENTER_VIEW	Provides privileges from the database level for any registered Oracle Warehouse Builder user to query the Warehouse Builder public views, such as ALL_IV_PROJECTS. A Warehouse Builder administrator can use the ACCESS_PUBLICVIEW_BROWSER system privilege from the Warehouse Builder security level to control an Warehouse Builder user's access to those public views.
OWB_USER	With Oracle Warehouse builder enables a remote Oracle WorkFlow instance to connect to the services provided by the Control Center.
PLUSTRACE	Traditionally required to use AUTOTRACE but in 11gR1 it seems to function without this role being required.
PUBLIC	-
RECOVERY_CATALOG_OWNER	Provides privileges for owner of the recovery catalog. Includes: CREATE SESSION, ALTER SESSION, CREATE SYNONYM, CREATE VIEW, CREATE DATABASE LINK, CREATE TABLE, CREATE CLUSTER, CREATE SEQUENCE, CREATE TRIGGER, and CREATE PROCEDURE
RESOURCE	Provides the following system privileges: CREATE CLUSTER, CREATE INDEXTYPE, CREATE OPERATOR, CREATE PROCEDURE, CREATE SEQUENCE, CREATE TABLE, CREATE TRIGGER, CREATE TYPE. This role is provided for compatibility with previous releases of Oracle Database. You can determine the privileges encompassed by this role by querying the DBA_SYS_PRIVS data dictionary view. **Note: Oracle recommends that you design your own roles for database security rather than relying on this role. This role may not be created automatically by future releases of Oracle Database.**
SCHEDULER_ADMIN	Allows the grantee to execute the procedures of the DBMS_SCHEDULER package. It includes all of the job scheduler system privileges and is included in the DBA role.
SELECT_CATALOG_ROLE	Provides SELECT privilege on objects in the data dictionary. Also provides the HS_ADMIN_ROLE privilege.
SPATIAL_CSW_ADMIN	Privileges granted the Catalog Services for the Web (CSW) account used by the Oracle Spatial CSW cache manager to load all record type metadata, and record instances from the database into the main memory for the record types that are cached.
SPATIAL_WFS_ADMIN	Privileges granted the Web Feature Service (WFS) account used by the Oracle Spatial WFS cache manager to load all feature type metadata, and feature instances from the database into main memory for the feature types that are cached.
WFS_USR_ROLE	Privileges granted a Web Feature Service (WFS) user
WKUSER	Privileges that must be granted to database users hosting new Oracle Ultra Search instances.
WM_ADMIN_ROLE	Contains all Workspace Manager privileges with the grant option. By default, the database administrator (DBA role) is granted the WM_ADMIN_ROLE role.
XDBADMIN	Allows the grantee to register an XML schema globally, as opposed to registering it for use or access only by its owner. It also lets the grantee bypass access control list (ACL) checks when accessing Oracle XML DB Repository.
XDB_SET_INVOKER	Allows the grantee to define invoker's rights handlers and to create or update the resource configuration for XML repository triggers. By default, Oracle Database grants this role to the DBA role but not to the XDBADMIN role.

XDB_WEBSERVICES	Allows the grantee to access Oracle Database Web services over HTTPS. However, it does not provide the user access to objects in the database that are public. To allow public access, you need to grant the user the XDB_WEBSERVICES_WITH_PUBLIC role. For a user to use these Web services, SYS must enable the Web service servlets.
XDB_WEBSERVICES_OVER_HTTP	Allows the grantee to access Oracle Database Web services over HTTP. However, it does not provide the user access to objects in the database that are public. To allow public access, you need to grant the user the XDB_WEBSERVICES_WITH_PUBLIC role.
XDB_WEBSERVICES_WITH_PUBLIC	Allows the grantee access to public objects through Oracle Database Web services.

To be continued