Practical 2-Introduction to Oracle XE using SqlPlus

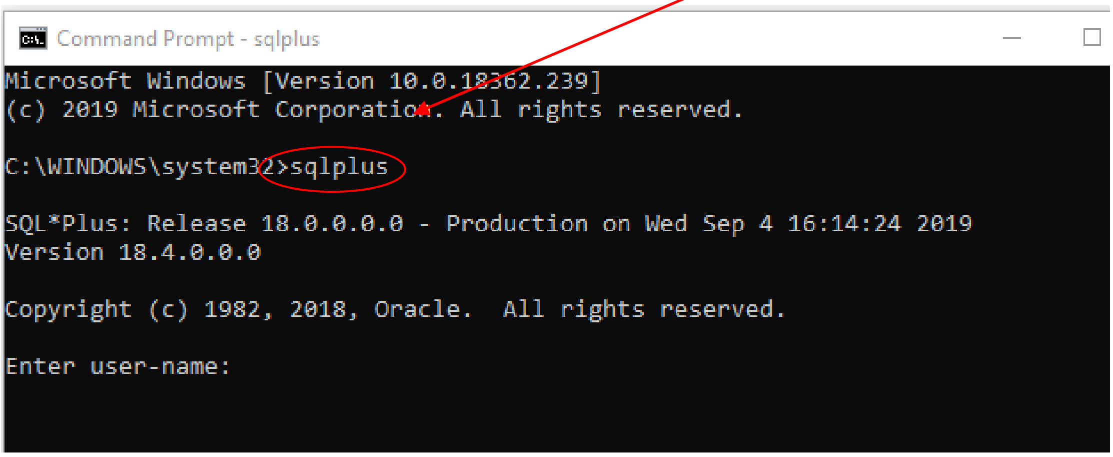
Oracle Database XE 18c (or 21c, most recent version)

Oracle Database XE is a cut down version of Oracle Enterprise Edition. You can download it on your home PC for practice. There are two methods available to you for using OracleXE, SQL*PLUS and SQLDeveloper. You can have BOTH open at once if you wish, for entering SQL and PL/SQL commands and running scripts.

Lets start with SQL*PLUS. During the semester you will be introduced to SQL Developer too and will be required to use it as the client interface to the database. We will toggle between SQLPLUS and Sql developer for various practicals.

SQLplus is a command-line program that you use to submit SQL and PL/SQL statements to an Oracle database. You can submit statements interactively or as SQLPlus scripts. SQL*Plus is installed with the database.

You can start SQL*Plus from the command line by typing CMD and then sqlplus, or on Microsoft Windows, from the Start menu by typing SQLplus.



```
Microsoft Windows [Version 10.0.18362.239]
(c) 2019 Microsoft Corporation. All rights reserved.

C:\WINDOWS\system32>sqlplus

SQL*Plus: Release 18.0.0.0.0 - Production on Wed Sep 4 16:14:24 2019
Version 18.4.0.0.0

Copyright (c) 1982, 2018, Oracle.  All rights reserved.

Enter user-name:
```

1. *Getting Started*

When prompted, enter the username and password:

Enter system for the username and system for the password when prompted.

You can also include the username and password when you start SQL via the Command Line.

sqlplus system/system

Alternatively enter from sql prompt that looks like this

SQL> connect system/system

- **To enter and execute SQL,** enter the statements at the SQL prompt, you can enter a single statement on multiple lines. At the end of a SQL statement, put a semi-colon (;) and then press the Enter key to execute the statement. For example:

  Check user by entering:

  SQL> SHOW USER; -- you are currently connected as system

  If the statement does not fit on one line, enter the first line and press the Enter key. Continue entering lines, and terminate the last line with a semi-colon (;).

  For example,

  SQL> SELECT username

  *SQL>*FROM all_users;

  If you wish to retrieve an already typed command use the up-arrow key.

- **Oracle table "dual"**

  All Oracle accounts have access to a table called **dual**. You can query against this table to get the current account, system date/time, and execute mathematical functions. For example run the following commands:

  SELECT user FROM dual; (same as SHOW USER);

  SELECT sysdate FROM dual;

  SELECT power(4,3) FROM dual;

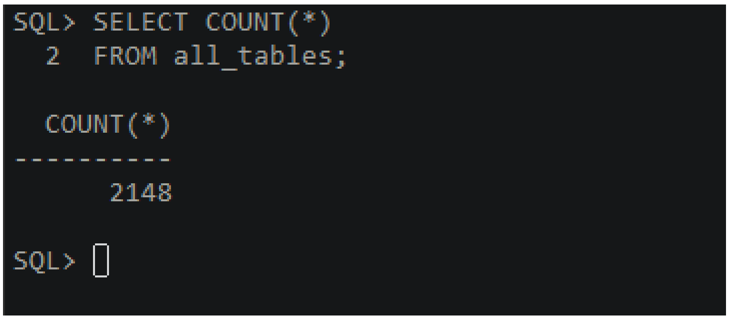
2. Open a Text Editor

Before you start writing your statements, **open Notepad, scITE(text editor, Notepad++ is good).** Keep a record of your work by copying and pasting from notepad to SQLPLUS as you progress. SQLplus is not very user-friendly and we will use SQL developer from next week, but it crucial that you obtain knowledge of it and that you become somewhat comfortable with it too!

In your chosen text editor type:

```
SELECT count(*)

FROM all_tables;
```

Now copy this statement from the text editor into SQL*Plus:



```
SQL> SELECT COUNT(*)
  2  FROM all_tables;

  COUNT(*)
----------
      2148

SQL> 
```

Save the text file(known as a script file) as Practical2.sql into your week2 practical folder.

It is common practice to use comments in SQL, use commenting dashes or blocks:

```
-- this is a comment

/*
This is a
multiline comment
*/
```

3. Spooling Output to a File

Apart from saving your work in Notepad it is also necessary to SPOOL your work. Spooling records every keystroke you make in your session in the lab, which is quite useful to look back on so that you can see all the errors you made and how you corrected them. (In the real world log files as created all the time!)

To **spool your work** you will type the following to a suitable location c:\, x:\:

SPOOL x:\<path>\<filename>

e.g. SPOOL X:\OraclePracticals\week2\pract2_log.log

Note that if you have any spaces in your path name, you must surround it in double quotes, for example: **SPOOL "X:\Oracle Practicals\week 2\pract2_log.log"**

This is ALSO NECESSARY for class assignments that ask for a log of your session, or in automated self generating scripts to check the output.

To **turn the spooling off**, enter the command:

SPOOL OFF

If you are executing a script file and wish to capture the commands from that file, use the SET ECHO command:

set echo {on|off}

Type into your .sql file and using the SQL prompt complete the following:

Output your resultset for each statement to a file on your X:\ by doing the following:

SQL>set echo on

SQL> spool x:\OraclePracticals\week2\pract1.log

--log files should always contain date and time at start and end

SQL>select to_char(sysdate, 'dd-mm-yyyy hh24:mi:ss') "Log start" from dual;

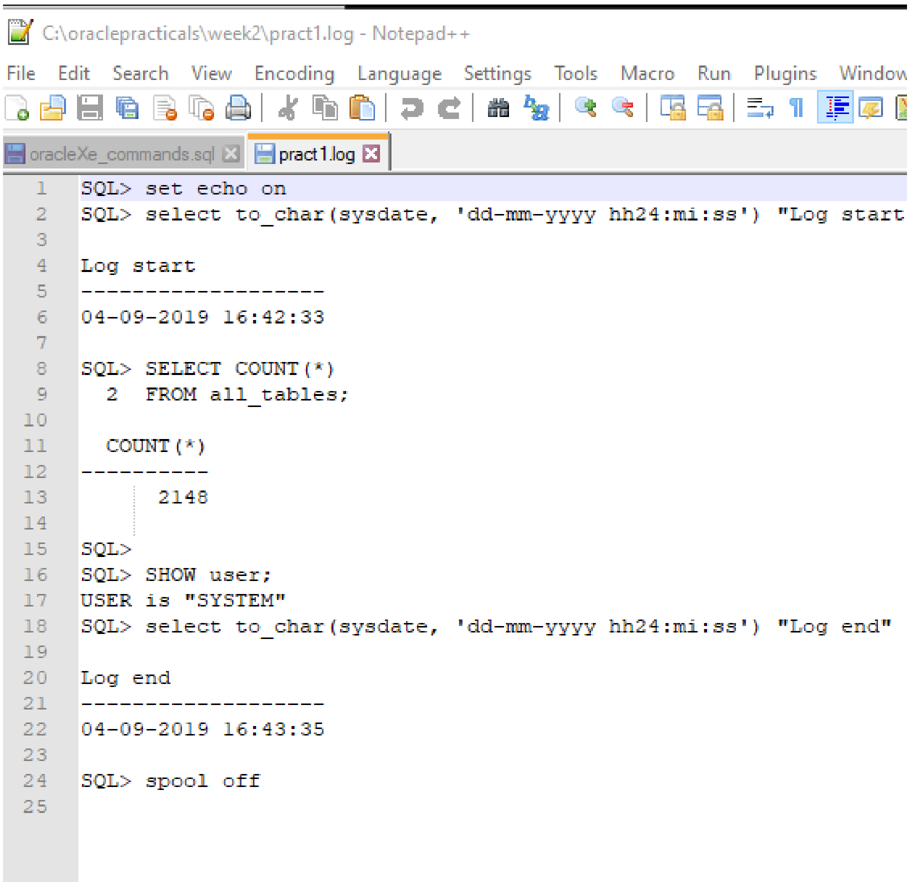
    --run script commands, for now type

SQL> show user;

SQL>select to_char(sysdate, 'dd-mm-yyyy hh24:mi:ss') "Log end" from dual;

SQL> Spool off

Check the log file in a text editor (notepad++) to observe its contents as follows…



```
SQL> set echo on
SQL> select to_char(sysdate, 'dd-mm-yyyy hh24:mi:ss') "Log start

Log start
-------------------
04-09-2019 16:42:33

SQL> SELECT COUNT(*)
  2  FROM all_tables;

  COUNT(*)
----------
      2148

SQL>
SQL> SHOW user;
USER is "SYSTEM"
SQL> select to_char(sysdate, 'dd-mm-yyyy hh24:mi:ss') "Log end"

Log end
-------------------
04-09-2019 16:43:35

SQL> spool off

```

You can also run the complete .sql file directly in SQLPlus.

Perform the following:

Spool again: **SPOOL X:\OraclePracticals\week2\pract2_log.log** (ensure that you have *your* correct path). Notice in your folder that your original log file has been overwritten!

Save your .sql file again.

Now run the script on sqlplus by executing the following command:

@X:\OraclePracticals\week2\practical2.sql

Note that if you have any spaces in your path name, you must surround it in double quotes, for example: **@ "X:\Oracle Practicals\week 2\practical2.sql"**

Spool off and observe the output in the log file.

4. Container Database (CDB)

Remember from the lecture that Oracle is a multitenant architecture that is a container database.

Every CDB has

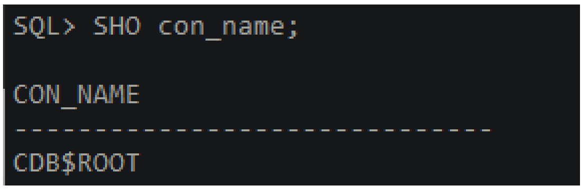
- one root with common users(Rem a common user is a database user known in every container e.g. system) The root container is called CDB$ROOT.
- one seed PDB. A template for creating new PDB's. It is called PDB$SEED
- zero or more PDB's. It is a user created pluggable database that contains the data and code for a specific application or group of users. Oracle XE comes with one PDB. In an enterprise environment no PDB's would exist initially, organisations would create many PDB's based on their business requirements.

4.1. Query the Containers

To query the containers, type the following queries into notepad(to save your work), then copy and paste to SQLplus. Observe the output!

4.1.1. Type the following show the current container you are connected to:

SHO con_name;



```
SQL> SHO con_name;

CON_NAME
------------------------------
CDB$ROOT
```

4.1.2.To query views about all containers installed.

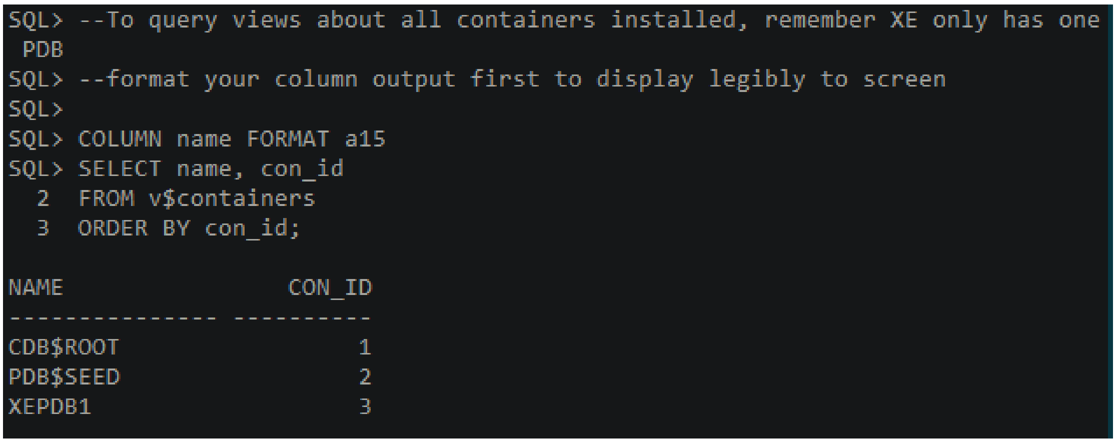
Remember XE only has one PDB and the seed. Format your column output first to display legibly to screen.

COLUMN name FORMAT a15

SELECT name, con_id

FROM v$containers

ORDER BY con_id;



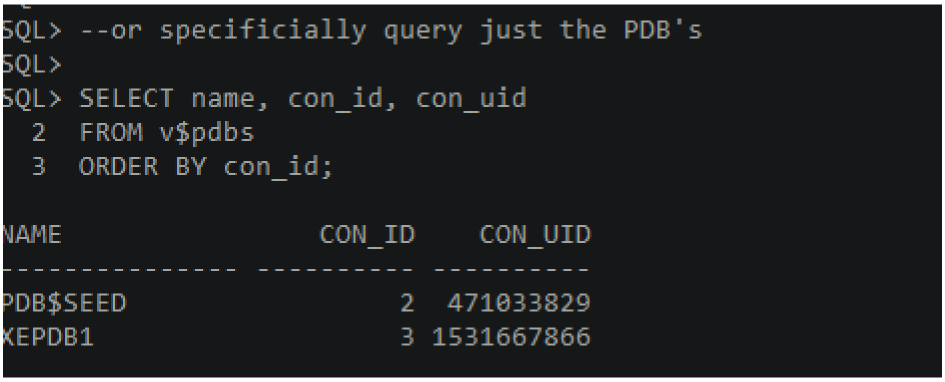
```
SQL> --To query views about all containers installed, remember XE only has one
 PDB
SQL> --format your column output first to display legibly to screen
SQL>
SQL> COLUMN name FORMAT a15
SQL> SELECT name, con_id
  2  FROM v$containers
  3  ORDER BY con_id;

NAME                CON_ID
--------------- ----------
CDB$ROOT                 1
PDB$SEED                 2
XEPDB1                   3
```

--or specificially query just the PDB's

SELECT name, con_id, con_uid

FROM v$pdbs

ORDER BY con_id;



```
SQL> --or specificially query just the PDB's
SQL>
SQL> SELECT name, con_id, con_uid
  2  FROM v$pdbs
  3  ORDER BY con_id;

NAME                CON_ID    CON_UID
--------------- ---------- ----------
PDB$SEED                 2  471033829
XEPDB1                   3 1531667866
```

4.1.3.The Table DBA_PDBS also contains information on PDB's

COLUMN pdb_name FORMAT a15 -- again format the output

SELECT pdb_id, pdb_name

FROM dba_pdbs

ORDER BY pdb_id;

4.1.4.Improve the output

The output is poor. There are commands you can run to improve the output.

You can use SET LINESIZE AND WRAP OFF,

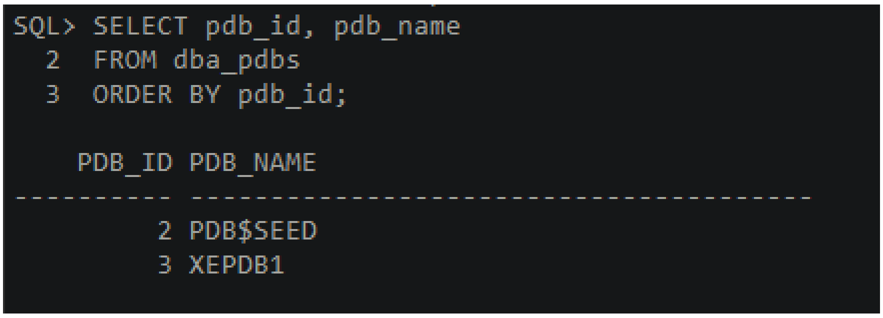
This command will make your row long enough to hold columns.

You need to increase or decrease number according your needs.

SET LINESIZE 50

SET WRAP OFF

-- run the statement again. It is better!



```
SQL> SELECT pdb_id, pdb_name
  2  FROM dba_pdbs
  3  ORDER BY pdb_id;

    PDB_ID PDB_NAME
---------- ----------------------------------------
         2 PDB$SEED
         3 XEPDB1
```

4.2. Etiquette

You may have noticed that any statements that I have written so far! I have laid them out legibly and used upper and lowercase to distinguish between keywords and column and table names. SQL is not case sensitive, but:

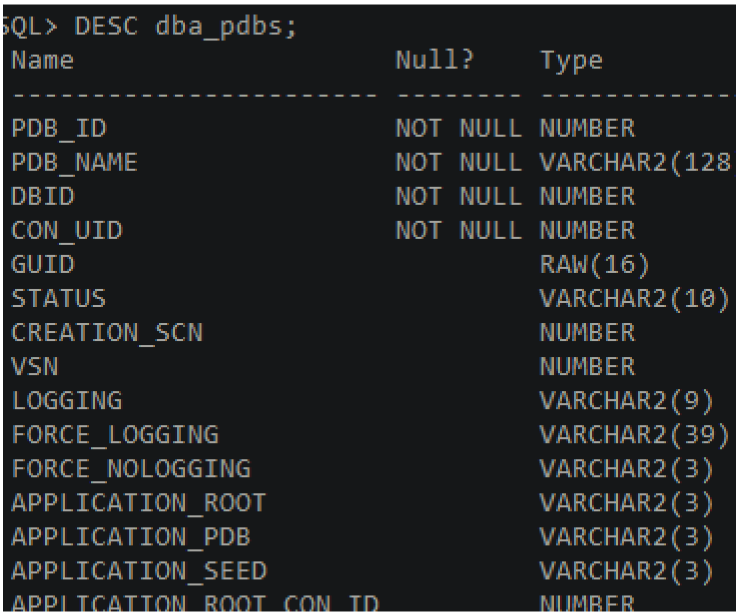
- Many organisations will have a standard in which they keep key words in capital letters.
- Anything surrounded by quotes is case sensitive. The actual data stored in the database is case sensitive e.g. ‘test’ is not the same as ‘TEST’ or ‘TesT’.
- Etiquette!: write keywords in uppercase and layout your statements legibly .

4.3. The Data Dictionary

You may wonder how you are to know all the data dictionary (DD) views and column names. As you learnt in the lecture the Oracle documentation is very good and a quick query will help you identify the view to find out the information you require. Again using a visual client tool makes development and administration easier, e.g. SQL developer, TOAD, Oracle Enterprise manager and we will use SQLDeveloper next week. However today we are learning about SQLPlus and we must query the DD views to find the information we are looking for.

I have placed on BB a table of views taken from Oracle documentation that lists views to query oracle roles and privileges.

In the view DBA_PDBS queried above, it contains the columns pdb_id and pdb_name. If I wanted to know the columns in DBA_PDBS to select. I would type: DESC dba_pdbs. We selected the pdb_id and pdb_name in the query in section 4.1.4.



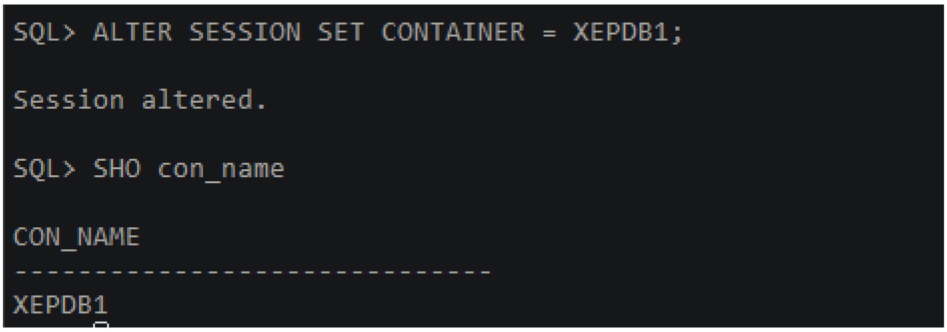
```
SQL> DESC dba_pdbs;
 Name                    Null?    Type
 ----------------------- -------- ------------
 PDB_ID                  NOT NULL NUMBER
 PDB_NAME                NOT NULL VARCHAR2(128
 DBID                    NOT NULL NUMBER
 CON_UID                 NOT NULL NUMBER
 GUID                             RAW(16)
 STATUS                           VARCHAR2(10)
 CREATION_SCN                     NUMBER
 VSN                              NUMBER
 LOGGING                          VARCHAR2(9)
 FORCE_LOGGING                    VARCHAR2(39)
 FORCE_NOLOGGING                  VARCHAR2(3)
 APPLICATION_ROOT                 VARCHAR2(3)
 APPLICATION_PDB                  VARCHAR2(3)
 APPLICATION_SEED                 VARCHAR2(3)
 APPLICATION_ROOT_CON_ID          NUMBER
```

5. *Switch Container*

In XE database, you will normally perform all operations inside the only PDB that XE supports.

To connect to a given database alter the session using the following command:

ALTER SESSION SET CONTAINER = XEPDB1;

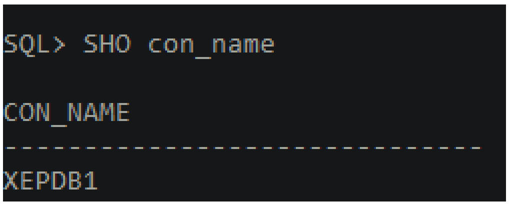


```
SQL> ALTER SESSION SET CONTAINER = XEPDB1;

Session altered.

SQL> SHO con_name

CON_NAME
------------------------------
XEPDB1
```

System is a common user that has privilege in this pluggable database too. Switch back to the root by typing:

ALTER SESSION SET CONTAINER = CDB$ROOT;

NOW Switch to the PDB again and type SHO con_name



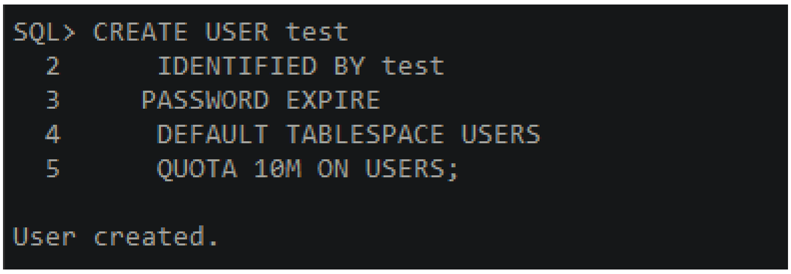
```
SQL> SHO con_name

CON_NAME
------------------------------
XEPDB1
```

You are now going to create a new user in XEPDB1…

6. Create a new User

Logged in as system in the PDB. Create a user called test. Allow the password to expire on first log on so that the user can reset it. Password expiry is important for DBA’s . It is called plausible deniability. It is important in this era that nobody including DBA's know a user’s password. You can set the default tablespace too, in this case its users. Set the quota, 10 megabytes on users tablespace.

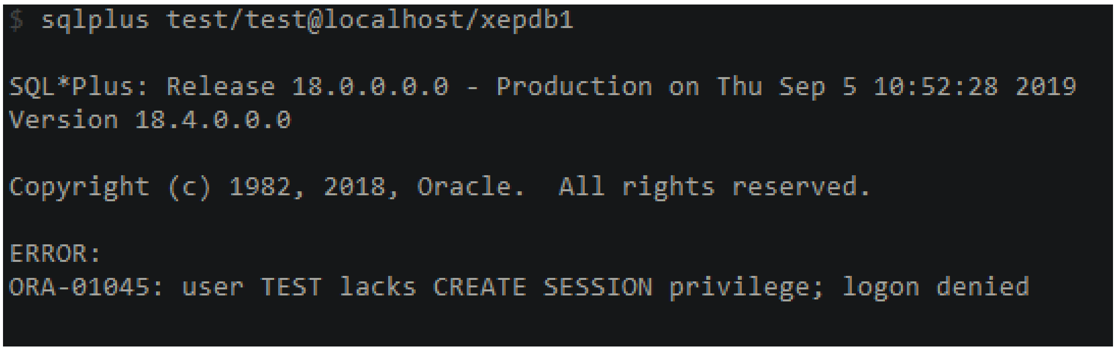
```
CREATE USER test
  IDENTIFIED BY test
  PASSWORD EXPIRE
  DEFAULT TABLESPACE USERS
  QUOTA 10M ON USERS;
```



```
SQL> CREATE USER test
  2      IDENTIFIED BY test
  3     PASSWORD EXPIRE
  4      DEFAULT TABLESPACE USERS
  5      QUOTA 10M ON USERS;

User created.
```

7. Connect as test

In another SQL pane, connect as test. Your must give the service name XEPDB1 and it will ask for test's password.

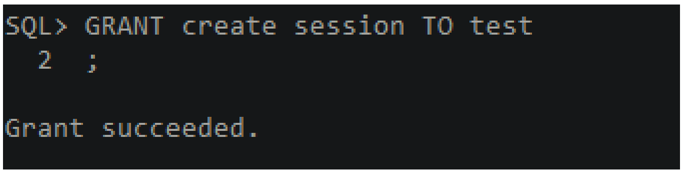


```
$ sqlplus test/test@localhost/xepdb1

SQL*Plus: Release 18.0.0.0.0 - Production on Thu Sep 5 10:52:28 2019
Version 18.4.0.0.0

Copyright (c) 1982, 2018, Oracle.  All rights reserved.

ERROR:
ORA-01045: user TEST lacks CREATE SESSION privilege; logon denied
```

What to note here is that test does not have CREATE SESSION privilege.

8. Grant Create session to test

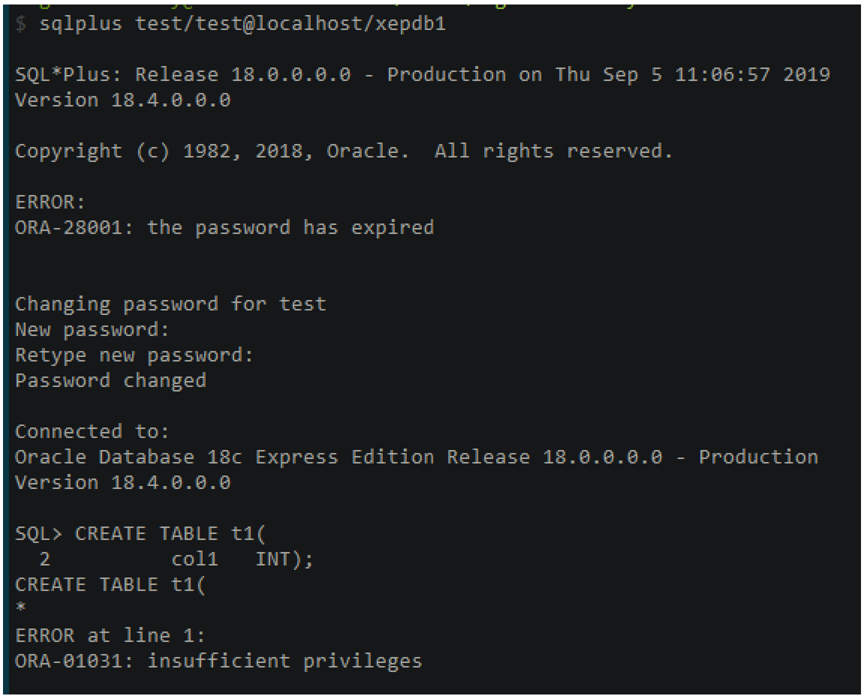
In your system connection GRANT create session TO test;



```
SQL> GRANT create session TO test
  2  ;

Grant succeeded.
```

When test connects it is asked to change the password. Change it to a suitable password.

Now test can connect but What can test do!

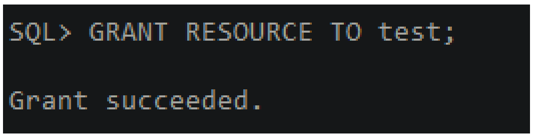


```
$ sqlplus test/test@localhost/xepdb1

SQL*Plus: Release 18.0.0.0.0 - Production on Thu Sep 5 11:06:57 2019
Version 18.4.0.0.0

Copyright (c) 1982, 2018, Oracle.  All rights reserved.

ERROR:
ORA-28001: the password has expired


Changing password for test
New password:
Retype new password:
Password changed

Connected to:
Oracle Database 18c Express Edition Release 18.0.0.0.0 - Production
Version 18.4.0.0.0

SQL> CREATE TABLE t1(
  2           col1   INT);
CREATE TABLE t1(
*
ERROR at line 1:
ORA-01031: insufficient privileges
```

The simple answer is very little. The only privilege test has is to connect. It cannot query or create objects etc.

9. Grant resource to test

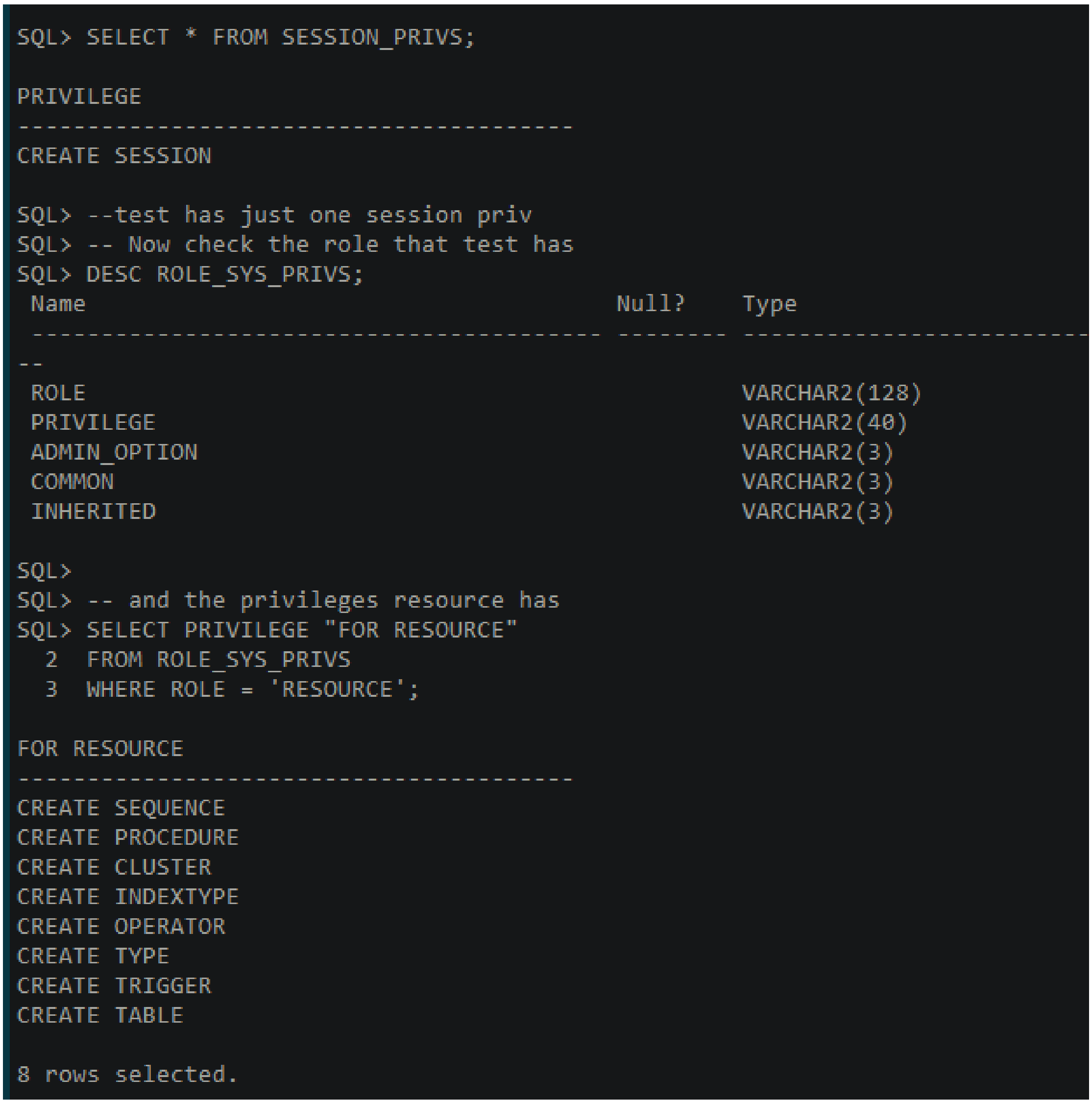
Back into your system connection pane to grant resource to test.



```
SQL> GRANT RESOURCE TO test;

Grant succeeded.
```

Now back to your test connection:

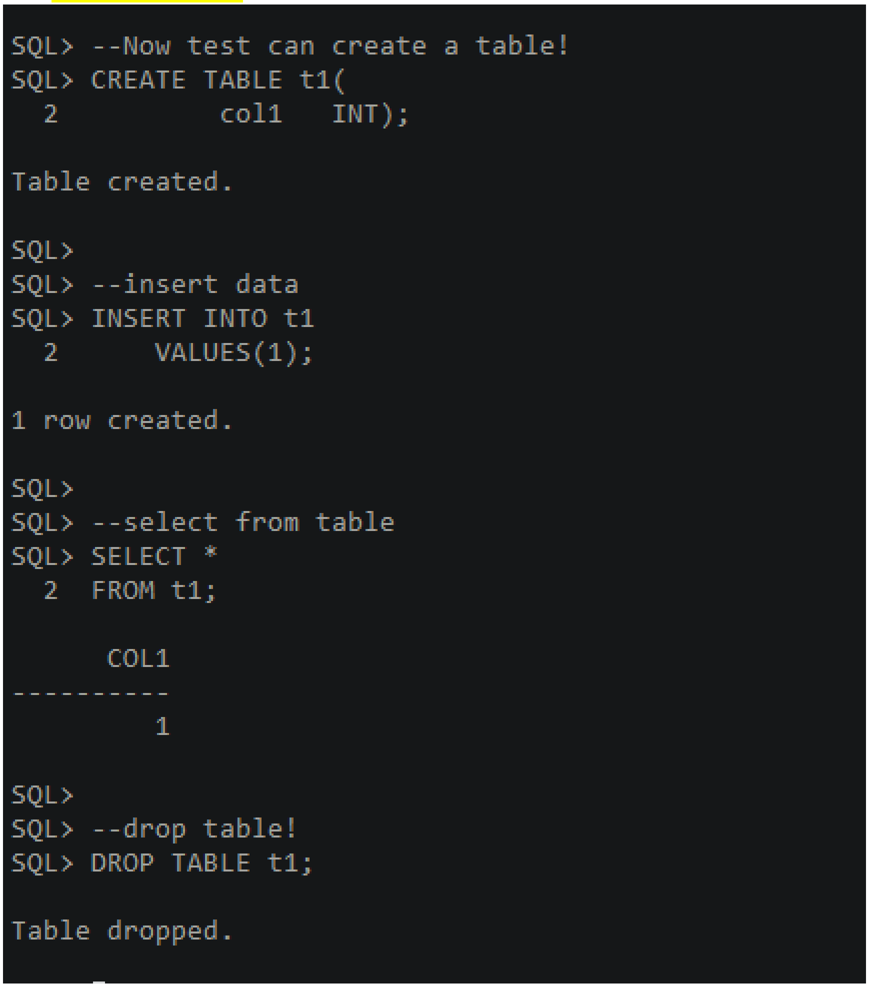
To Query the privileges that test has use the following DD views. Remember a list of these view in available in the week 2 folder on BB and in the oracle documentation. See screendump below:



```
SQL> SELECT * FROM SESSION_PRIVS;

PRIVILEGE
----------------------------------------
CREATE SESSION

SQL> --test has just one session priv
SQL> -- Now check the role that test has
SQL> DESC ROLE_SYS_PRIVS;
 Name                                      Null?    Type
 ----------------------------------------- -------- -------------------------
--
 ROLE                                               VARCHAR2(128)
 PRIVILEGE                                          VARCHAR2(40)
 ADMIN_OPTION                                       VARCHAR2(3)
 COMMON                                             VARCHAR2(3)
 INHERITED                                          VARCHAR2(3)

SQL>
SQL> -- and the privileges resource has
SQL> SELECT PRIVILEGE "FOR RESOURCE"
  2  FROM ROLE_SYS_PRIVS
  3  WHERE ROLE = 'RESOURCE';

FOR RESOURCE
----------------------------------------
CREATE SEQUENCE
CREATE PROCEDURE
CREATE CLUSTER
CREATE INDEXTYPE
CREATE OPERATOR
CREATE TYPE
CREATE TRIGGER
CREATE TABLE

8 rows selected.
```

If you run the same commands connected as system you will see that system has a lot more privileges. We will use the DD views regularly over the semester.

10. *Test can now create tables, insert data and drop the table and lots other stuff!*
    Perform the following to create a table, insert a record into the table and select from the table:

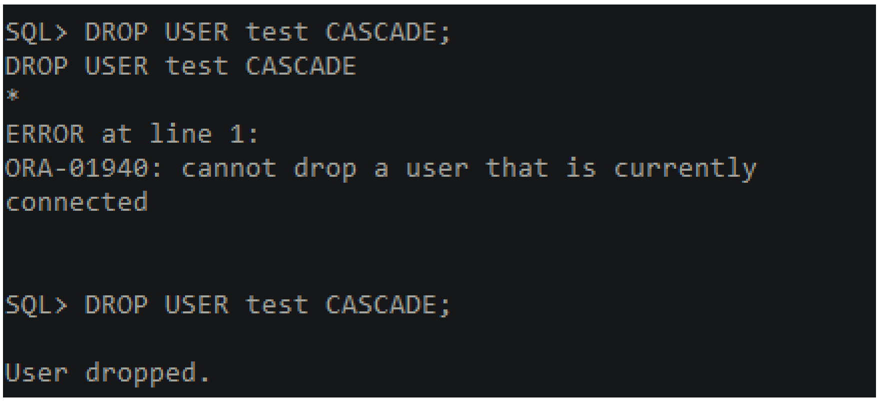


```
SQL> --Now test can create a table!
SQL> CREATE TABLE t1(
  2          col1   INT);

Table created.

SQL>
SQL> --insert data
SQL> INSERT INTO t1
  2       VALUES(1);

1 row created.

SQL>
SQL> --select from table
SQL> SELECT *
  2   FROM t1;

      COL1
----------
         1

SQL>
SQL> --drop table!
SQL> DROP TABLE t1;

Table dropped.
```

Can you Delete the record from the table before dropping the table!

11. Drop the test user

Connected as system drop the user called test. Test must be disconnected! And Because test owns objects you must also use the keyword cascade!



```
SQL> DROP USER test CASCADE;
DROP USER test CASCADE
*
ERROR at line 1:
ORA-01940: cannot drop a user that is currently
connected


SQL> DROP USER test CASCADE;

User dropped.
```

12. Task!

Create a new user called company on XEPDB1 with a suitable password, provide tablespace and quota.

Grant necessary privileges to company.

Create and populate a table called employee with 2 columns and 2 rows.

Select from employee.

Delete one record from employee.

Drop the company schema.

End of Practical