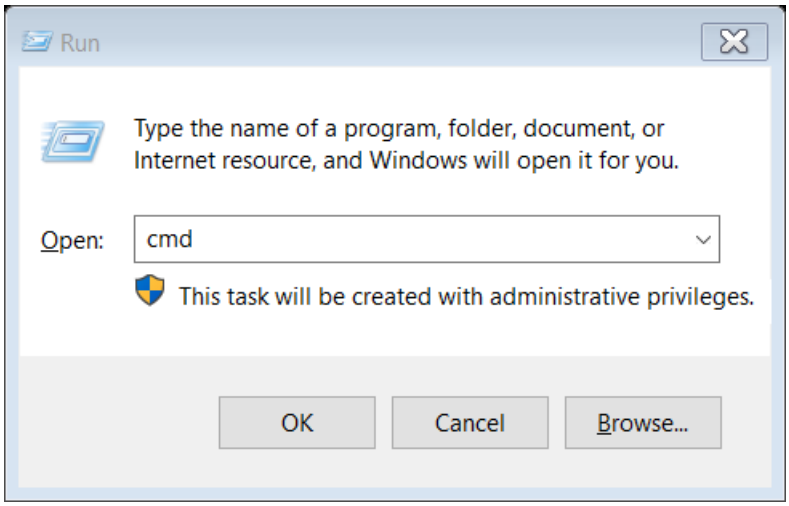
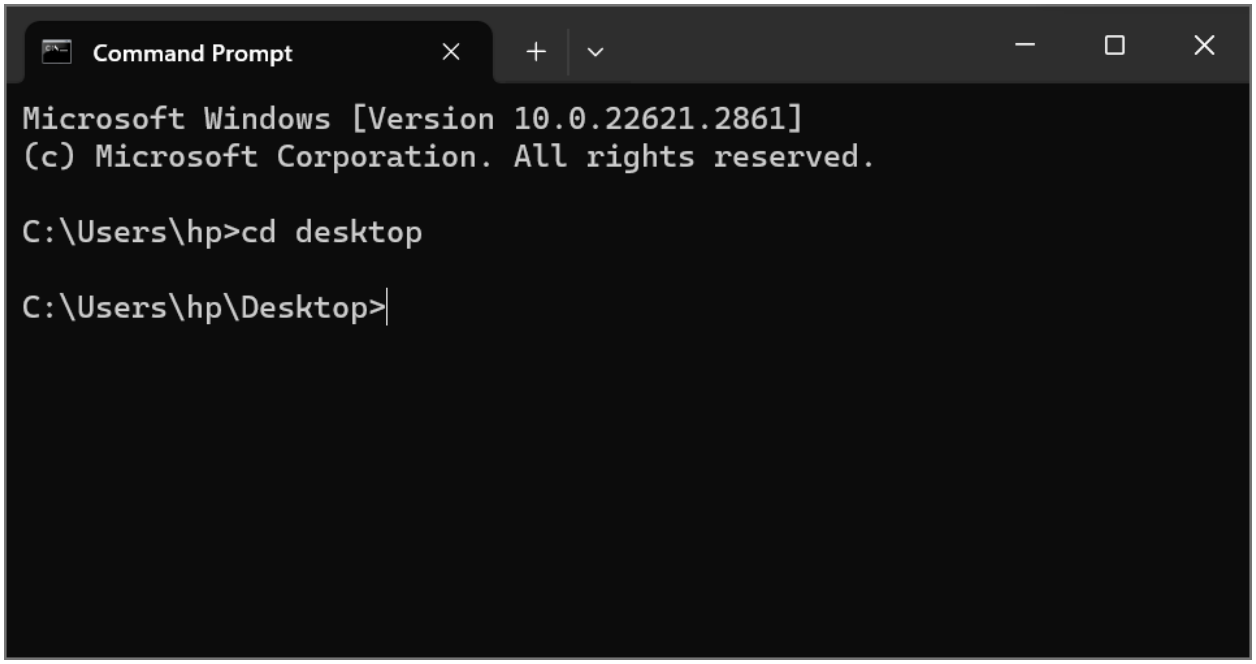
Design of Databases using DDL Commands

1. Open the command prompt Press WIN+R , type cmd

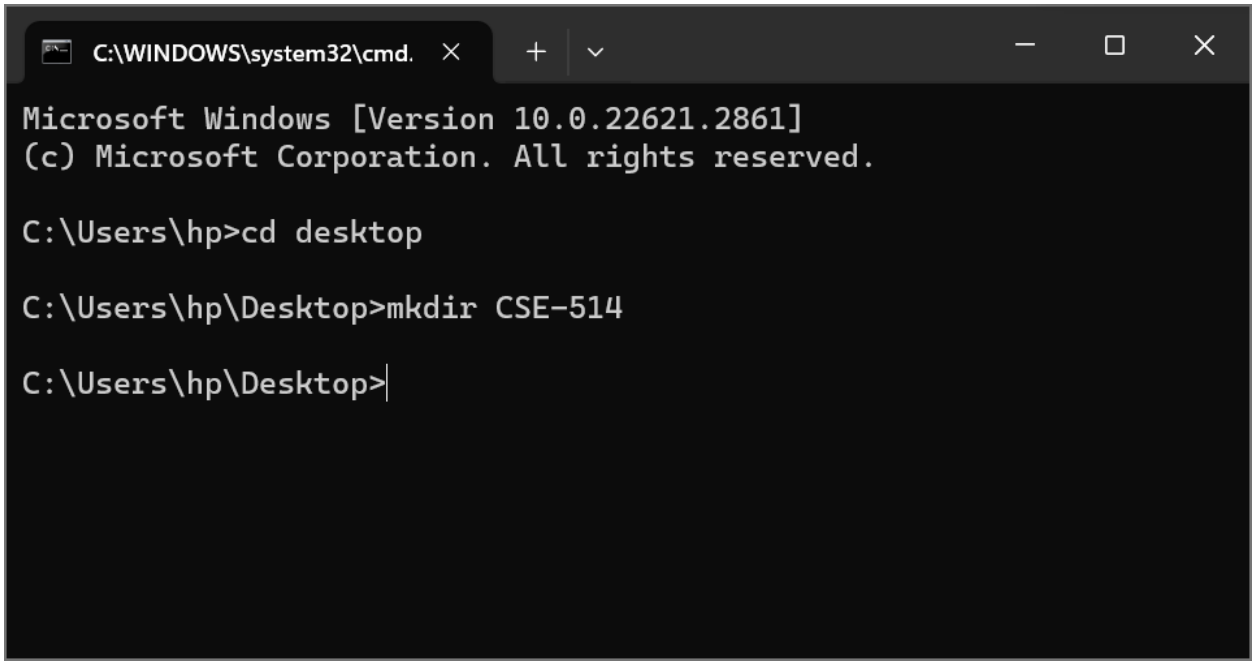


2. Once cmd prompt open go to DESKTOP using cd Desktop

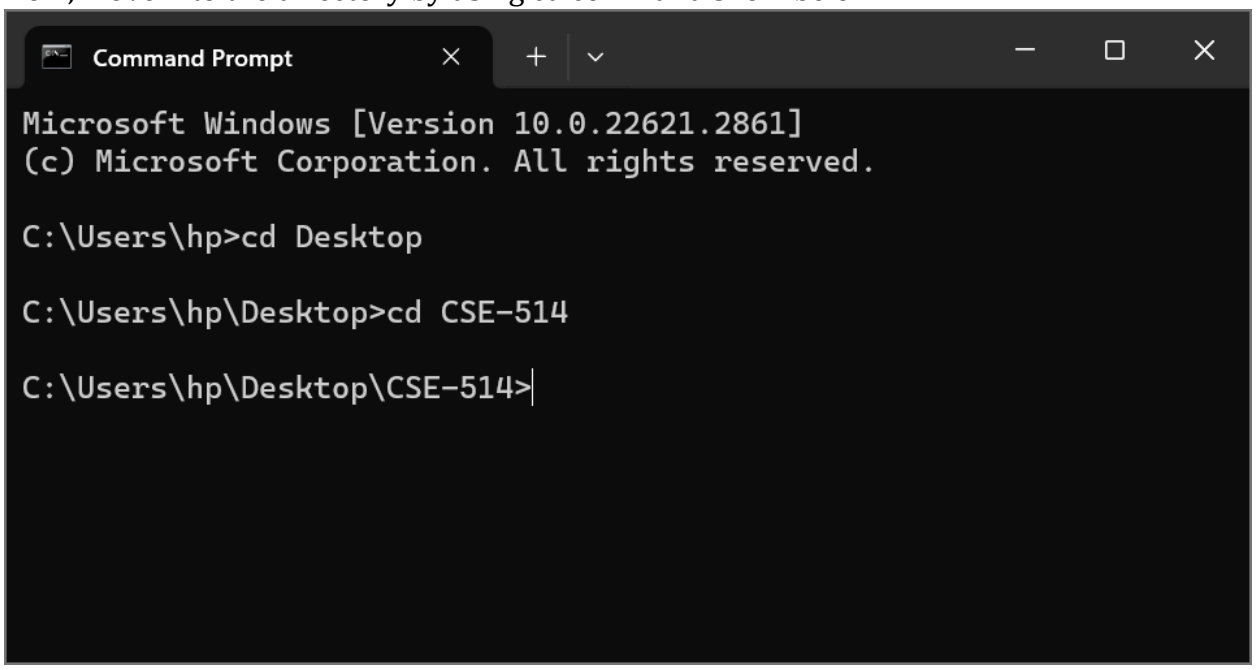


3. Now create a Directory using mkdir or md command using your branch abbrevation and last 3 digit hall ticket number like md CSE-514.

Design of Databases using DDL Commands



4. Now, move into the directory by using cd command show below.



5. To Login, , Type sqlplus command enter username and password when system is prompted.

Design of Databases using DDL Commands

6. **CREATE TABLE:**

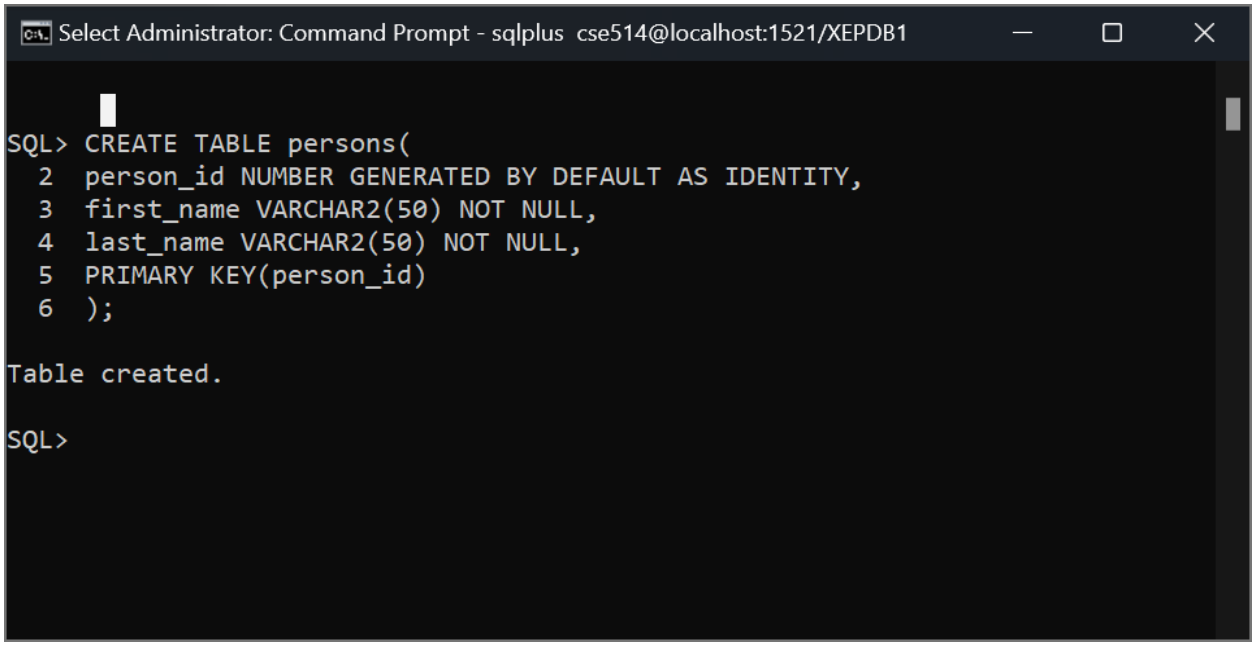
   To create a new table in oracle Database, you use CREATE TABLE statement

   Syntax:

```
CREATE TABLE schema_name.table_name (
    column_1 data_type column_constraint,
    column_2 data_type column_constraint,
    ...
    table_constraint
 );
```

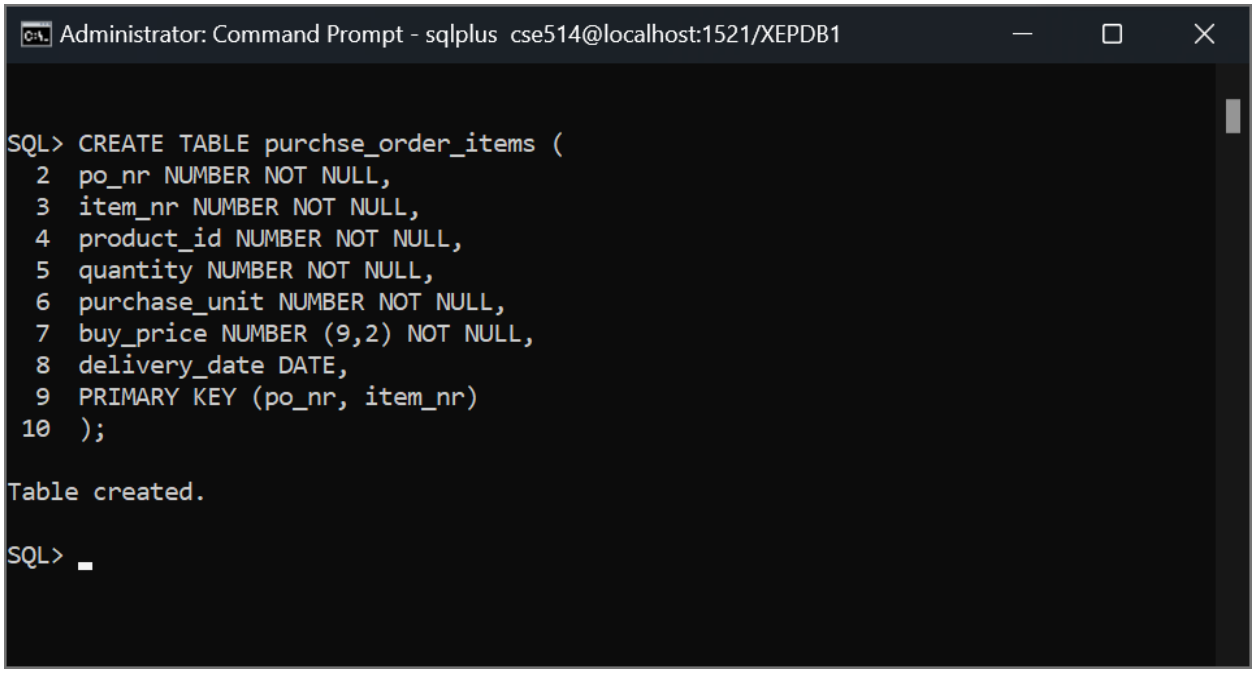
7. Now Oracle Create Table statement of one table example:

   create table for persons



8. Oracle Create Table of multiple columns example

Design of Databases using DDL Commands



ALTER TABLE:

To modify the structure of an existing table, you use the ALTER TABLE statement

Syntax :

```
ALTER TABLE table_name action;
```

9. Oracle Alter Table ADD column

Syntax:

```
ALTER TABLE table_name
ADD column_name type constraint;
```

Design of Databases using DDL Commands

```
SQL> ALTER TABLE persons
  2  ADD birthdate DATE NOT NULL;

Table altered.

SQL> DESC persons;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 PERSON_ID                                 NOT NULL NUMBER
 FIRST_NAME                                NOT NULL VARCHAR2(50)
 LAST_NAME                                 NOT NULL VARCHAR2(50)
 BIRTHDATE                                 NOT NULL DATE
```

10. Oracle ALTER TABLE for multiple columns

Syntax:

```
ALTER TABLE table_name
 MODIFY ( column_1 type constraint,
          column_1 type constraint,
         ...);
```

```
SQL> ALTER TABLE persons
  2  ADD(
  3  phone VARCHAR(20),
  4  email VARCHAR(20)
  5  );

Table altered.

SQL> DESC persons
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 PERSON_ID                                 NOT NULL NUMBER
 FIRST_NAME                                NOT NULL VARCHAR2(50)
 LAST_NAME                                 NOT NULL VARCHAR2(50)
 BIRTHDATE                                 NOT NULL DATE
 PHONE                                              VARCHAR2(20)
 EMAIL                                              VARCHAR2(20)
```

11. **DROP TABLE**

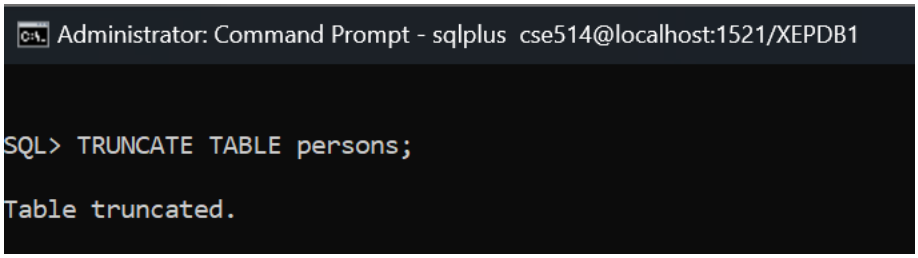
To move a table to recycle bin or remove it entirely from database, you use DROP TABLE statement.

Design of Databases using DDL Commands

```
SQL> DROP TABLE persons;

Table dropped.
```

12. **TRUNCATE TABLE**

Oracle introduced the TRUNCATE TABLE statement that allows you to delete all rows from big table.

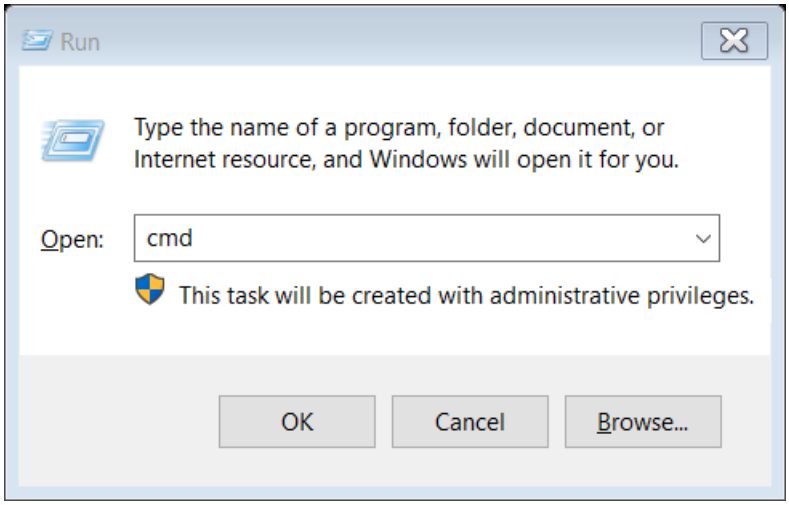


13.Summary of the Lab Report

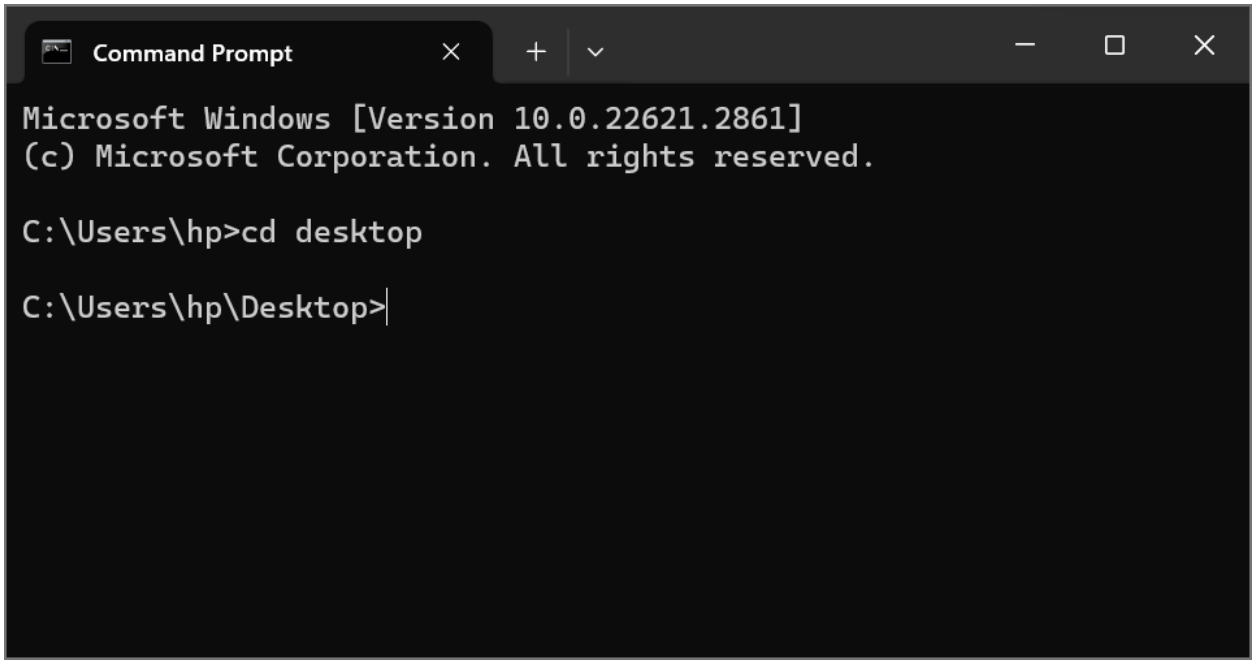
Design of Databases using DDL Commands

1.	Number of Screen Shorts taken from Step 5	10
2.	Number of tables creation specified in observation	20
3.	Number of tables you created in the lab	
4.	Number of Select Statements specified in the observation	20
5.	Number of Select statements you practised in lab	19
6	Number of Insert Statements specified in observation	20
7	Number of Insert Statements you practiced in Lab	15
8.	Number of Alter Statements specified in observation	10
9	Number of Alter Statements practiced in lab	11
10	Number of Drop Statements specified in Observation	5
11	Number of Drop Statements Practiced in Lab	6
12	Number of truncate Statements Specified in Observation	
13	Number of Truncate statements Practiced in Lab	
14	Total number of Statements specified in lab	100
15	Total number of statements practiced in lab	80
16	Number of any addition statements practiced by you.	13
17	Number of Screenshots available in the your Document, Removing First 7 Screenshots	110
18	Status of the lab in percentage	80%

Design of Databases using DDL Commands

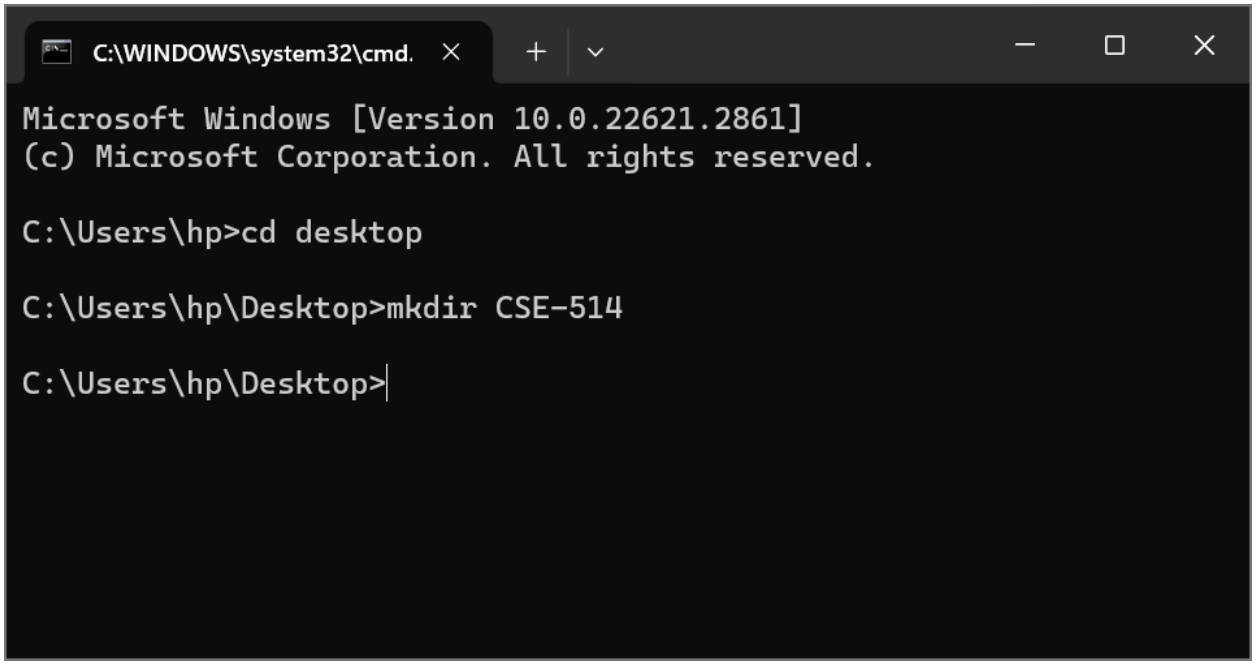


14. Once cmd prompt open go to DESKTOP using cd Desktop

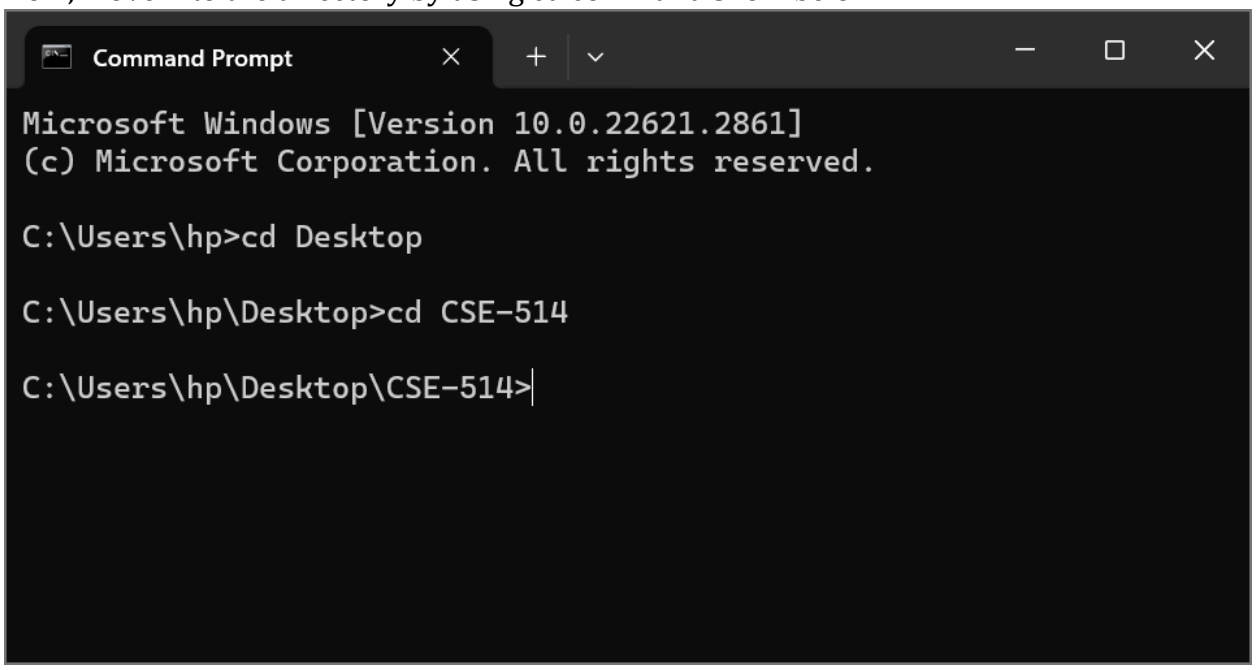


15. Now create a Directory using mkdir or md command using your branch abbrevation and last 3 digit hall ticket number like md CSE-514.

Design of Databases using DDL Commands

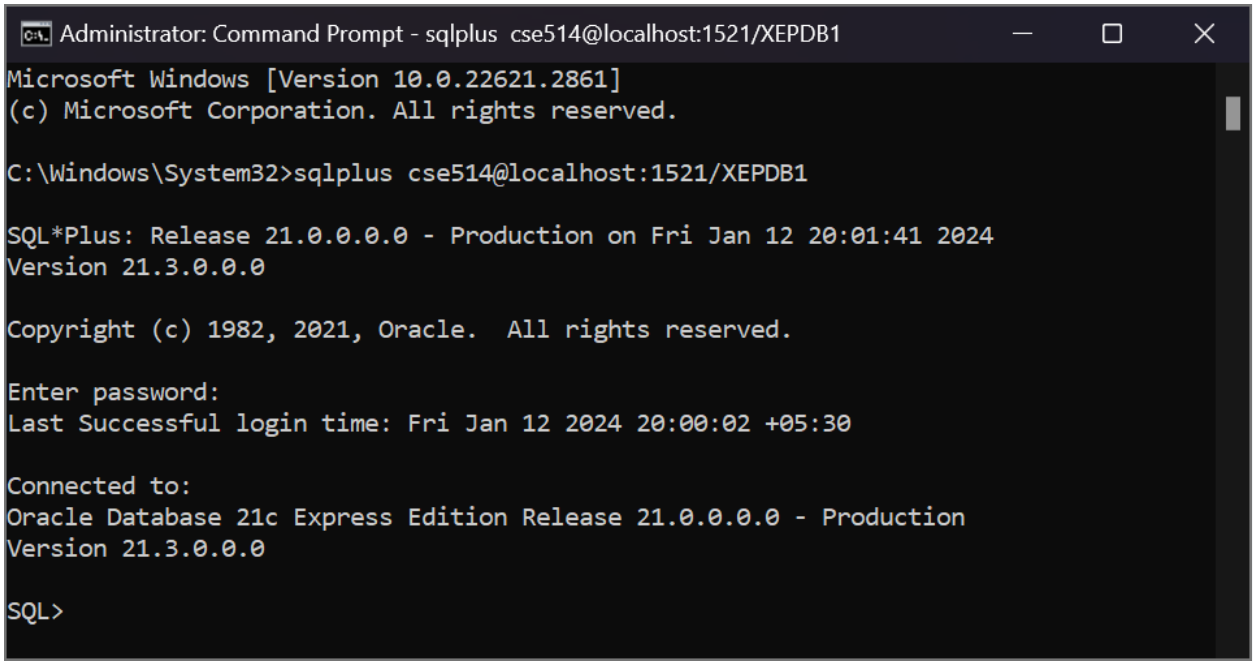


16. Now, move into the directory by using cd command show below.



17. To Login, , Type sqlplus command enter username and password when system is prompted.

Design of Databases using DDL Commands



18. Now you have to create a file using spool command, the file name must contain experiment no, branch, hall ticket number and date. For example, exp1_cse_501_14_sep_2023.txt

19. Check you login into correct user by using show user command and also set the prompt

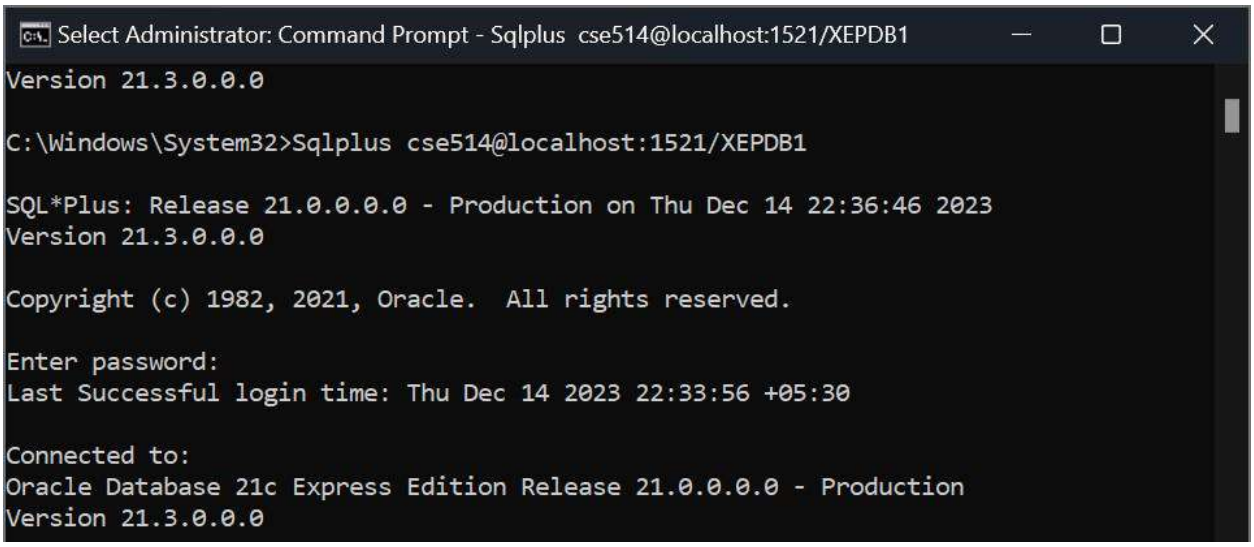
20. Now, execute all the commands that are Discussed in your observation and also some addition commands which are practiced in the lab. Once executed paste the Screen Shots below

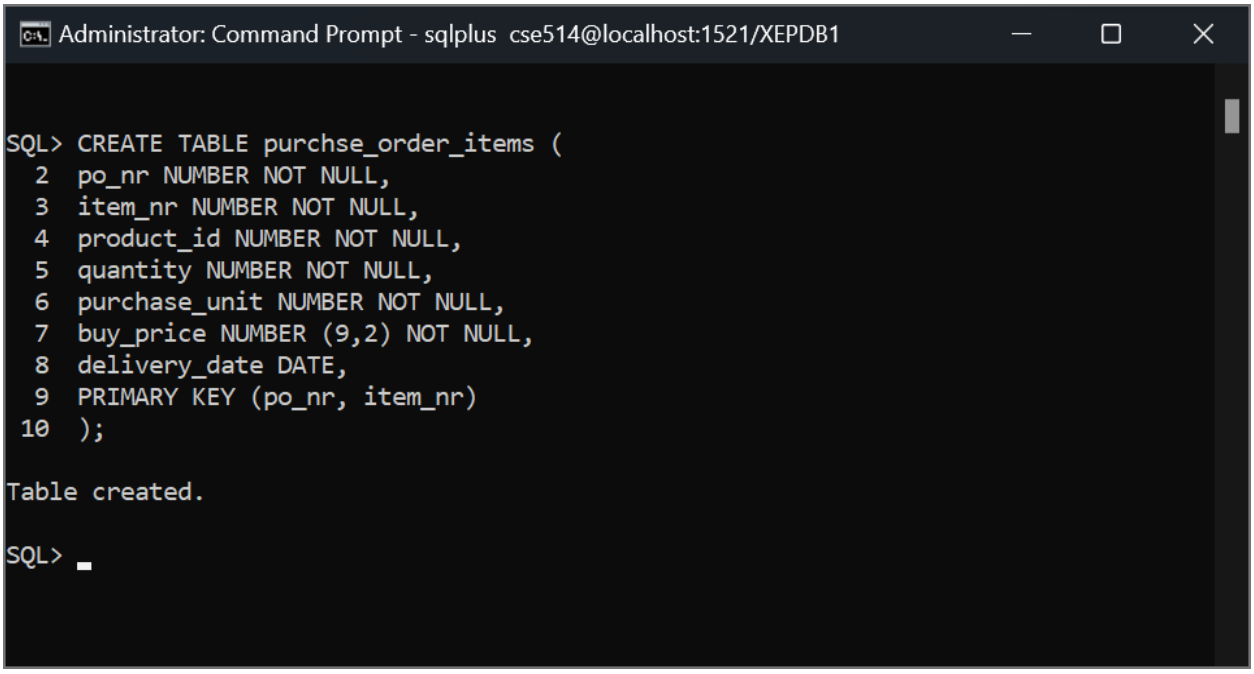
21. Summary of the Lab Report

Design of Databases using DDL Commands

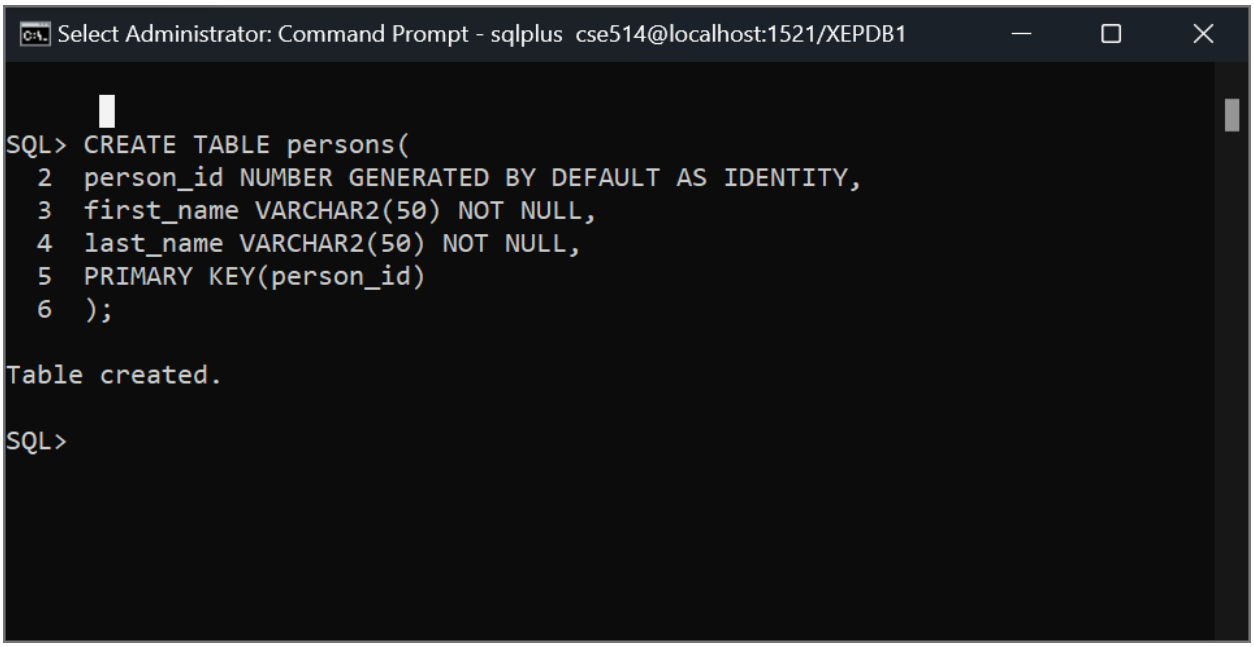
1.	Number of Screen Shorts taken from Step 8	100
2.	Number of tables creation specified in observation	20
3.	Number of tables you created in the lab	19
4.	Number of Select Statements specified in the observation	20
5.	Number of Select statements you practised in lab	19
6	Number of Insert Statements specified in observation	20
7	Number of Insert Statements you practiced in Lab	15
8.	Number of Alter Statements specified in observation	10
9	Number of Alter Statements practiced in lab	11
10	Number of Drop Statements specified in Observation	5
11	Number of Drop Statements Practiced in Lab	6
12	Number of truncate Statements Specified in Observation	
13	Number of Truncate statements Practiced in Lab	
14	Total number of Statements specified in lab	100
15	Total number of statements practiced in lab	80
16	Number of any addition statements practiced by you.	13
17	Number of Screenshots available in the your Document, Removing First 7 Screenshots	110
18	Status of the lab in percentage	80%

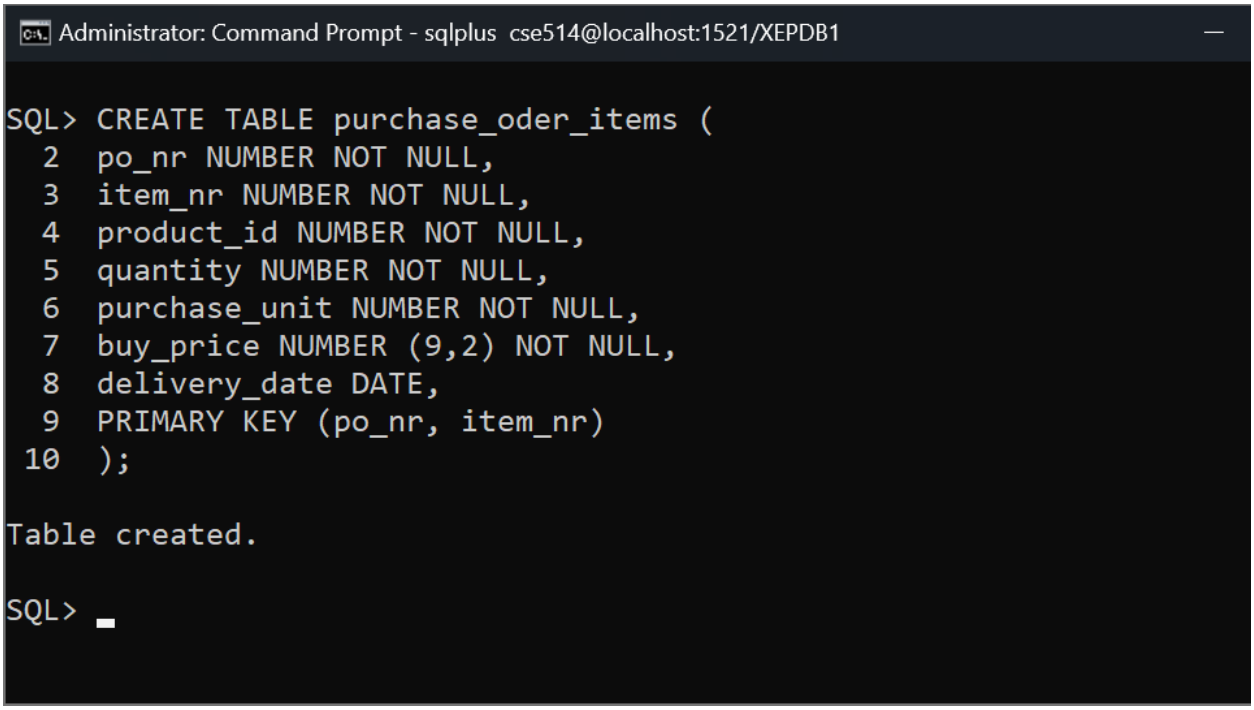
Design of Databases using DDL Commands





Design of Databases using DDL Commands





Design of Databases using DDL Commands

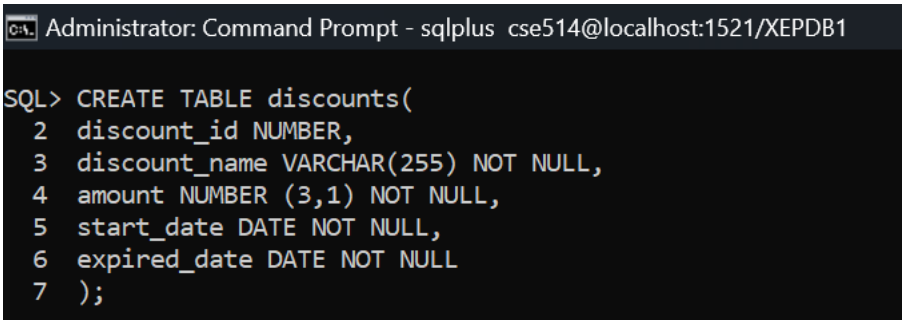
```
SQL> ALTER TABLE persons
  2  ADD birthdate DATE NOT NULL;

Table altered.

SQL> DESC persons;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 PERSON_ID                                 NOT NULL NUMBER
 FIRST_NAME                                NOT NULL VARCHAR2(50)
 LAST_NAME                                 NOT NULL VARCHAR2(50)
 BIRTHDATE                                 NOT NULL DATE
```

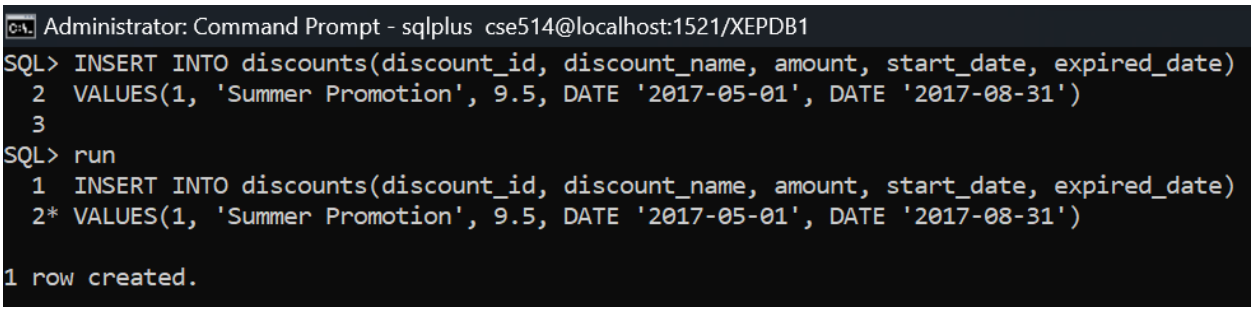
```
SQL> ALTER TABLE persons
  2  ADD(
  3  phone VARCHAR(20),
  4  email VARCHAR(20)
  5  );

Table altered.

SQL> DESC persons
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 PERSON_ID                                 NOT NULL NUMBER
 FIRST_NAME                                NOT NULL VARCHAR2(50)
 LAST_NAME                                 NOT NULL VARCHAR2(50)
 BIRTHDATE                                 NOT NULL DATE
 PHONE                                              VARCHAR2(20)
 EMAIL                                              VARCHAR2(20)
```

```
SQL> DROP TABLE persons;

Table dropped.
```

```
Administrator: Command Prompt - sqlplus  cse514@localhost:1521/XEPDB1


SQL> TRUNCATE TABLE persons;

Table truncated.
```

Design of Databases using DDL Commands





Design of Databases using DDL Commands

```
SQL> declare
  2  num number;
  3  i number:=1;
  4  c number:=0;
  5  begin
  6  num:=&num;
  7  for i in 1..num
  8  loop
  9  if((mod(num,i))=0)
 10  then
 11  c:=c+1;
 12  end if;
 13  end loop;
 14  if(c>2)
 15  then
 16  dbms_output.put_line(num || ' not a prime');
 17  else
 18  dbms_output.put_line(num || ' is prime');
 19  end if;
 20  end;
 21  /
Enter value for num: 3
3 is prime

PL/SQL procedure successfully completed.
```

Design of Databases using DDL Commands

```
SQL> DECLARE
  2  FIRST NUMBER:=0;
  3  SECOND NUMBER:=1;
  4  TEMP NUMBER;
  5  N NUMBER:=&num;
  6  I NUMBER;
  7  BEGIN
  8  DBMS_OUTPUT.PUT_LINE('SERIES:');
  9  DBMS_OUTPUT.PUT_LINE(FIRST);
 10  DBMS_OUTPUT.PUT_LINE(SECOND);
 11  FOR I IN 2..N
 12  LOOP
 13  TEMP:=FIRST+SECOND;
 14  FIRST:=SECOND;
 15  SECOND:=TEMP;
 16  DBMS_OUTPUT.PUT_LINE(TEMP);
 17  END LOOP;
 18  END;
 19  /
Enter value for num: 5
SERIES:
0
1
1
2
3
5

PL/SQL procedure successfully completed.
```

Design of Databases using DDL Commands

```
PL/SQL procedure successfully completed.

SQL> DECLARE
  2  fac NUMBER := 1;
  3  n NUMBER := &NUM;
  4  BEGIN
  5  WHILE n > 0
  6  LOOP
  7  fac:=n*fac;
  8  n:=n-1;
  9  END LOOP;
 10  DBMS_OUTPUT.PUT_LINE(FAC);
 11  END;
 12  /
Enter value for num: 7
5040

PL/SQL procedure successfully completed.

SQL> SET SERVEROUT ON
SQL> /
Enter value for num: 3
6

PL/SQL procedure successfully completed.
```

Design of Databases using DDL Commands

```
SQL> DECLARE
  2  Fact NUMBER := 1;
  3  n NUMBER;
  4  n1 NUMBER;
  5  BEGIN
  6  n:=&n;
  7  n1:=n;
  8  WHILE n>0 LOOP
  9  Fact := n*fact;
 10  n:=n-1;
 11  END LOOP;
 12  DBMS_OUTPUT.PUT_LINE('Factorial of '|| n1 || ' is :' ||Fact
);
 13  END;
 14  /
Enter value for n: 4
Factorial of 4 is :24

PL/SQL procedure successfully completed.

SQL> SET SERVEROUT ON
SQL> /
Enter value for n: 5
Factorial of 5 is :120

PL/SQL procedure successfully completed.
```

Design of Databases using DDL Commands

```
SQL> DECLARE
  2  n NUMBER;
  3  i NUMBER;
  4  temp NUMBER;
  5  n1 NUMBER;
  6  BEGIN
  7  n := &n;
  8  n1 := n;
  9  i := 2;
 10  temp := 1;
 11  FOR i IN 2..n/2
 12  LOOP
 13  IF MOD(n, i) = 0
 14  THEN
 15  temp := 0;
 16  EXIT;
 17  END IF;
 18  END LOOP;
 19  IF temp = 1
 20  THEN
 21  DBMS_OUTPUT.PUT_LINE(n||' is a prime number');
 22  ELSE
 23  DBMS_OUTPUT.PUT_LINE(n||' is not a prime number');
 24  END IF;
 25  END;
 26  /
Enter value for n: 5
5 is a prime number

PL/SQL procedure successfully completed.
```

```
SQL> CREATE TABLE SAILOR(
  2  ID NUMBER (10) PRIMARY KEY,
  3  NAME VARCHAR2(100) );

Table created.
```

Design of Databases using DDL Commands

```
SQL> CREATE OR REPLACE PROCEDURE INSERTUSER (
  2      ID   IN NUMBER,
  3      NAME IN VARCHAR2
  4  ) IS
  5  BEGIN
  6      INSERT INTO SAILOR VALUES (ID, NAME);
  7      DBMS_OUTPUT.PUT_LINE('RECORD INSERTED SUCCESSFULLY');
  8  END;
  9  /

Procedure created.
```

```
SQL> EXEC INSERTUSER(102, 'CGKJ')

PL/SQL procedure successfully completed.
```

```
SQL> SET SERVEROUT ON
SQL> /

Procedure created.

SQL> EXEC INSERTUSER(&id, '&name');
Enter value for id: 12
Enter value for name: edrf
RECORD INSERTED SUCCESSFULLY

PL/SQL procedure successfully completed.
```

Design of Databases using DDL Commands

```
SQL> CREATE OR REPLACE FUNCTION ADDER(N1 IN NUMBER, N2 IN NUMBER
)
  2  RETURN NUMBER
  3  IS
  4  N3 NUMBER(8);
  5  BEGIN
  6  N3 :=N1+N2;
  7  RETURN N3;
  8  END;
  9  /

Function created.

SQL> SELECT ADDER(9, 6) FROM DUAL;

ADDER(9,6)
----------
        15
```

```
SQL> DROP FUNCTION Adder;

Function dropped.
```

```
SQL> CREATE FUNCTION fact(x number)
  2  RETURN number
  3  IS
  4   f number;
  5  BEGIN
  6   IF x=0 THEN
  7   f := 1;
  8   ELSE
  9   f := x * fact(x-1);
 10   END IF;
 11  RETURN f;
 12  END;
 13  /
```

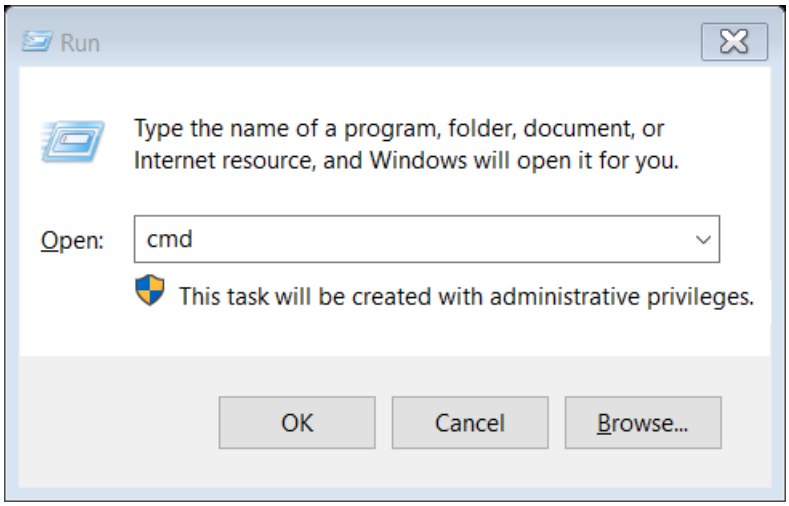
Design of Databases using DDL Commands

```
SQL> DECLARE
  2  num number;
  3  factorial number;
  4  BEGIN
  5   num:= 6;
  6   factorial := fact(num);
  7   dbms_output.put_line(' Factorial '|| num || ' is ' || fact
orial);
  8  END;
  9  /

PL/SQL procedure successfully completed.

SQL> set serverout on
SQL> /
Factorial 6 is 720

PL/SQL procedure successfully completed.
```

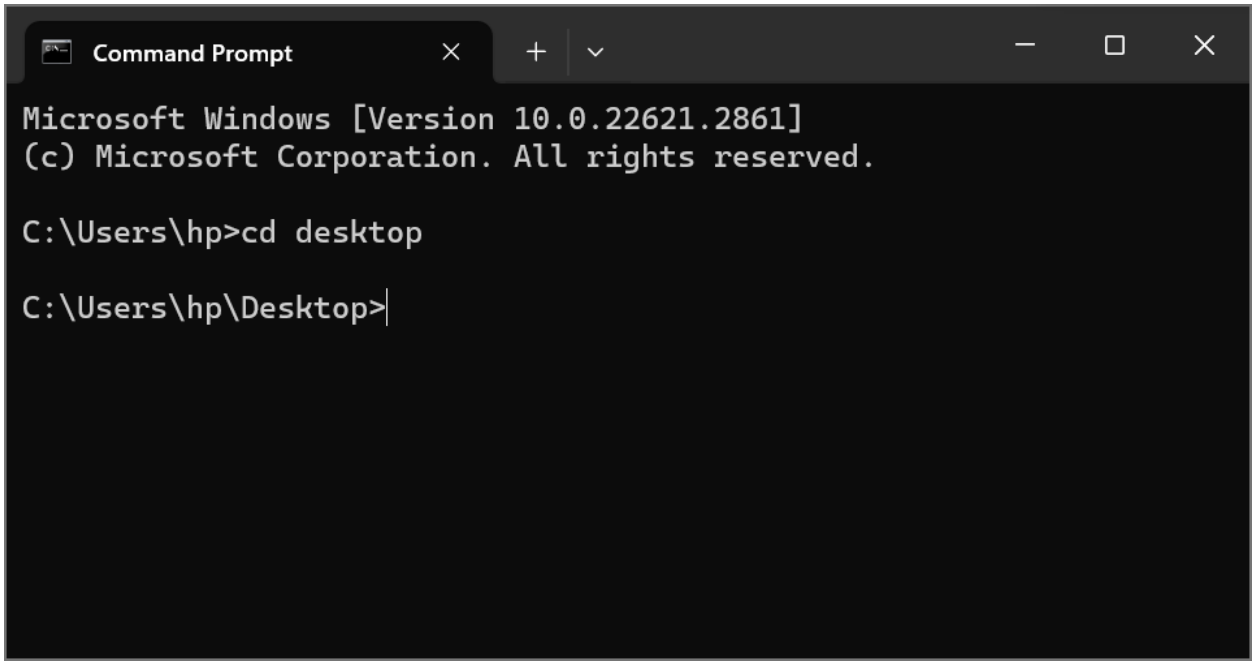
Name : S.M.Chaithra Experiment - 2 Date: 19-10-2023

Manipulating and Querying of Database Using DML Commands

1. Open the command prompt Press WIN+R , type cmd

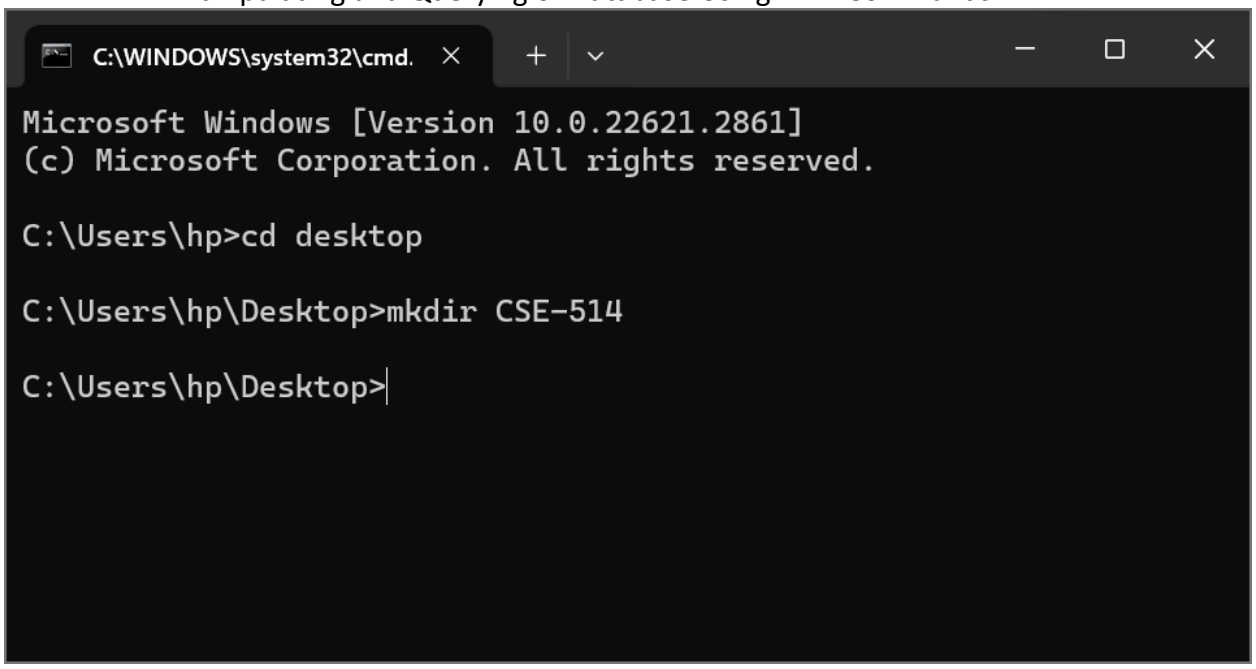


2. Once cmd prompt open go to DESKTOP using cd Desktop

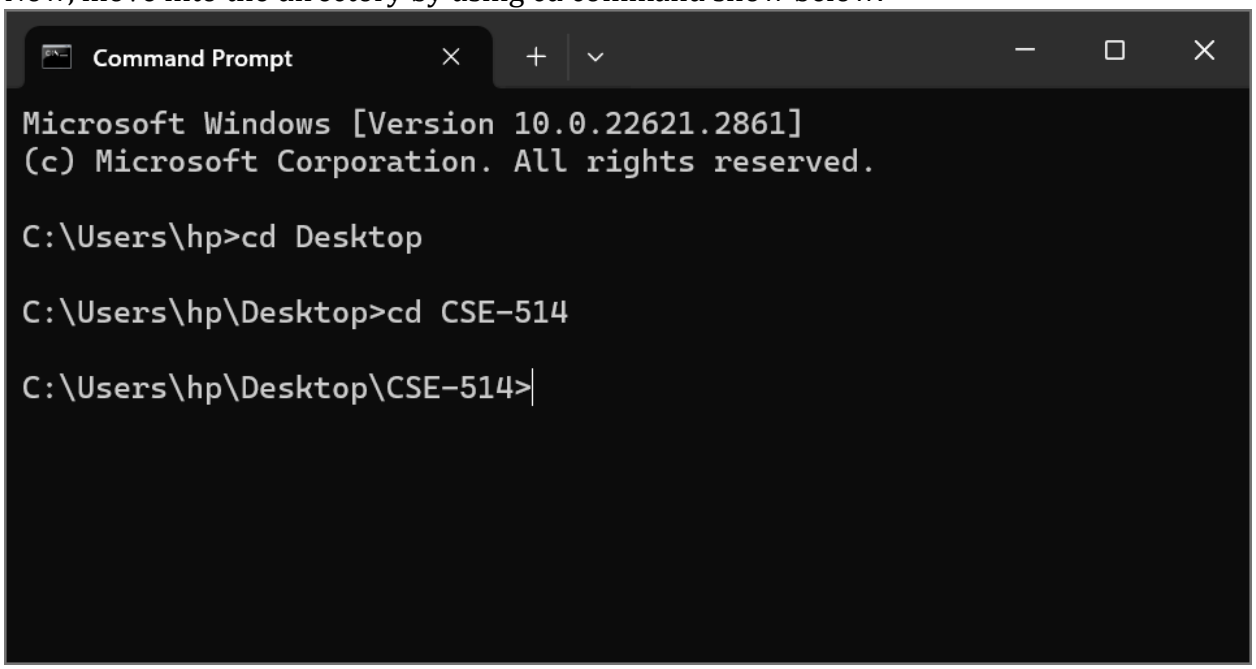


3. Now create a Directory using mkdir or md command using your branch abbrevation and last 3 digit hall ticket number like md CSE-514.

Manipulating and Querying of Database Using DML Commands



4. Now, move into the directory by using cd command show below.



5. To Login, , Type sqlplus command enter username and password when system is prompted.

Manipulating and Querying of Database Using DML Commands

6. DML Commands are used to manipulate and query databases using DML Commands like INSERT, SELECT, UPDATE, and delete.

7. **Introduction to Oracle INSERT statement:**
   To insert a new row into a table, you use the Oracle INSERT statement as follows:

```
INSERT INTO table_name (column_list)
VALUES( value_list);
```

   If the value list has the same order as the table columns, you can skip the column list although this is not considered as a good practice:

```
INSERT INTO table_name
VALUES (value_list);
```

EX:

```
SQL> CREATE TABLE parts (
  2  part_id NUMBER,
  3  part_name VARCHAR(50) NOT NULL,
  4  lead_time NUMBER(2, 0) NOT NULL,
  5  cost NUMBER(9,2) NOT NULL,
  6  status NUMBER(1,0) NOT NULL,
  7  PRIMARY KEY(part_id)
  8  );

Table created.
```

Manipulating and Querying of Database Using DML Commands

```
SQL> INSERT INTO parts(part_id,part_name,lead_time,cost,status)
  2  VALUES(1,'sed dictum',5,134,0);

1 row created.

SQL> INSERT INTO parts(part_id,part_name,lead_time,cost,status)
  2  VALUES(2,'tristique neque',3,62,1);

1 row created.
```

```
SQL> INSERT INTO parts(part_id,part_name,lead_time,cost,status)
  2  VALUES(3,'dolor quam',16,82,1);

1 row created.

SQL> INSERT INTO parts(part_id,part_name,lead_time,cost,status)
  2  VALUES(4,'nec, diam.',41,10,1);

1 row created.

SQL> INSERT INTO parts(part_id,part_name,lead_time,cost,status)
  2  VALUES(5,'vitae erat',22,116,0);

1 row created.
```

8. Oracle UPDATE – update multiple columns of a single row

```
UPDATE
    table_name
SET
    column1 = value1,
    column2 = value2,
    column3 = value3,
    ...
WHERE
    condition;
```

EX:

Manipulating and Querying of Database Using DML Commands

```
SQL> UPDATE parts
  2  SET cost = 130
  3  WHERE part_id = 1;

1 row updated.
```

Oracle UPDATE – update multiple rows example

EX:

```
SQL> UPDATE parts
  2  SET cost = cost * 1.05;

5 rows updated.
```

9. SELECT COMMAND:

The SELECT command used to list the contents of a table.

```
INSERT INTO target_table (col1, col2, col3)
SELECT col1,
       col2,
       col3
FROM source_table
WHERE condition;
```

EX:

```
SQL> SELECT * FROM parts
  2  WHERE part_id = 1;

   PART_ID PART_NAME                                          LEAD_TIME
---------- -------------------------------------------------- ----------
      COST     STATUS
---------- ----------
         1 sed dictum                                                  5
     136.5          0
```

Manipulating and Querying of Database Using DML Commands

```
SQL> SELECT * FROM parts;

   PART_ID PART_NAME                                          LEAD_TIME
---------- -------------------------------------------------- ----------
      COST     STATUS
---------- ----------
         1 sed dictum                                                  5
     136.5          0

         2 tristique neque                                             3
      65.1          1

         3 dolor quam                                                 16
      86.1          1


   PART_ID PART_NAME                                          LEAD_TIME
---------- -------------------------------------------------- ----------
      COST     STATUS
---------- ----------
         4 nec, diam.                                                 41
      10.5          1

         5 vitae erat                                                 22
     121.8          0
```

10.DELETE COMMAND

To delete all rows or specified rows in a table.

```
DELETE
FROM
    table_name
WHERE
    condition;
```

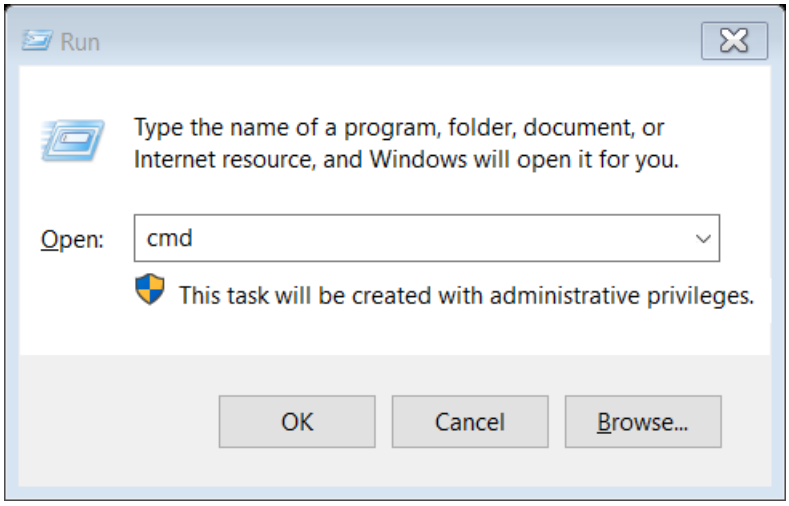
```
SQL> DELETE FROM parts;

5 rows deleted.
```

```
SQL> SELECT * FROM parts;

no rows selected
```

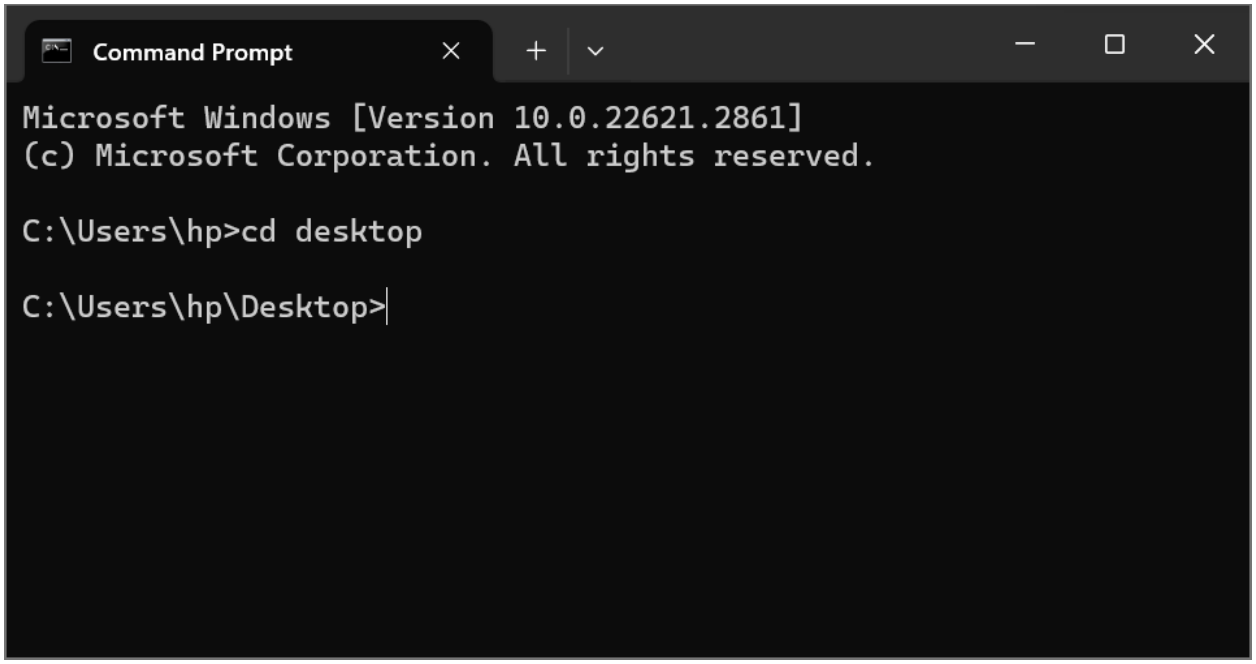
Manipulating and Querying of Database Using DML Commands

SQL queries to implement VIEWS for various database

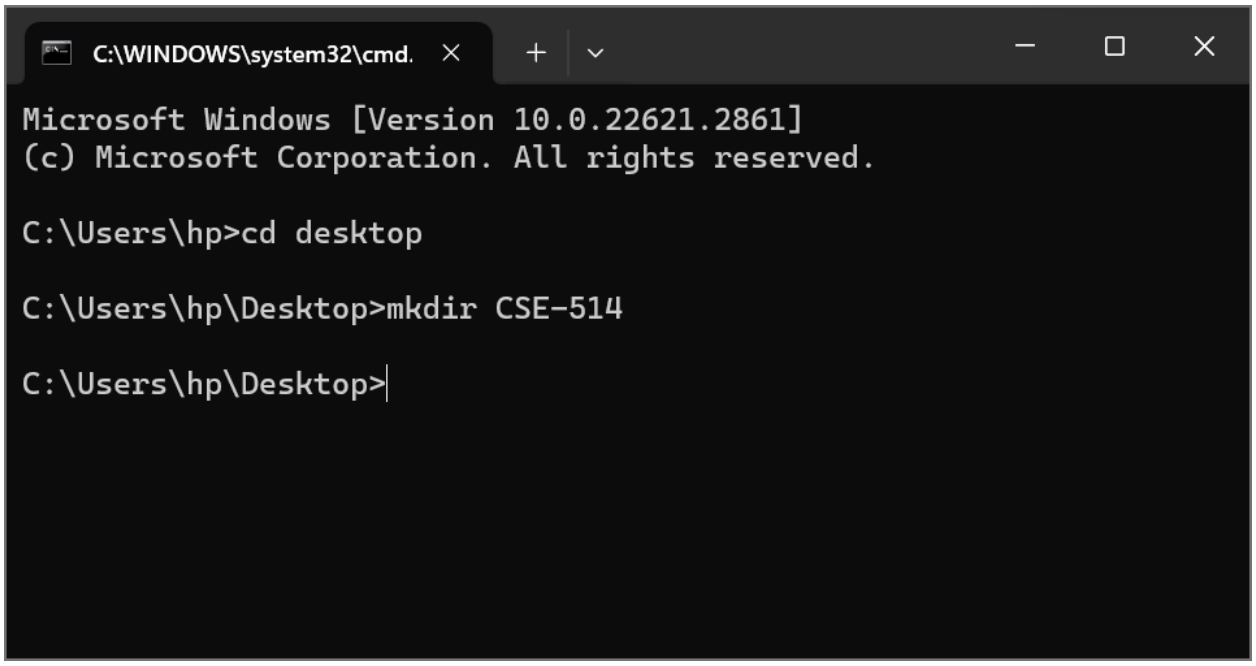
1. Open the command prompt Press WIN+R , type cmd



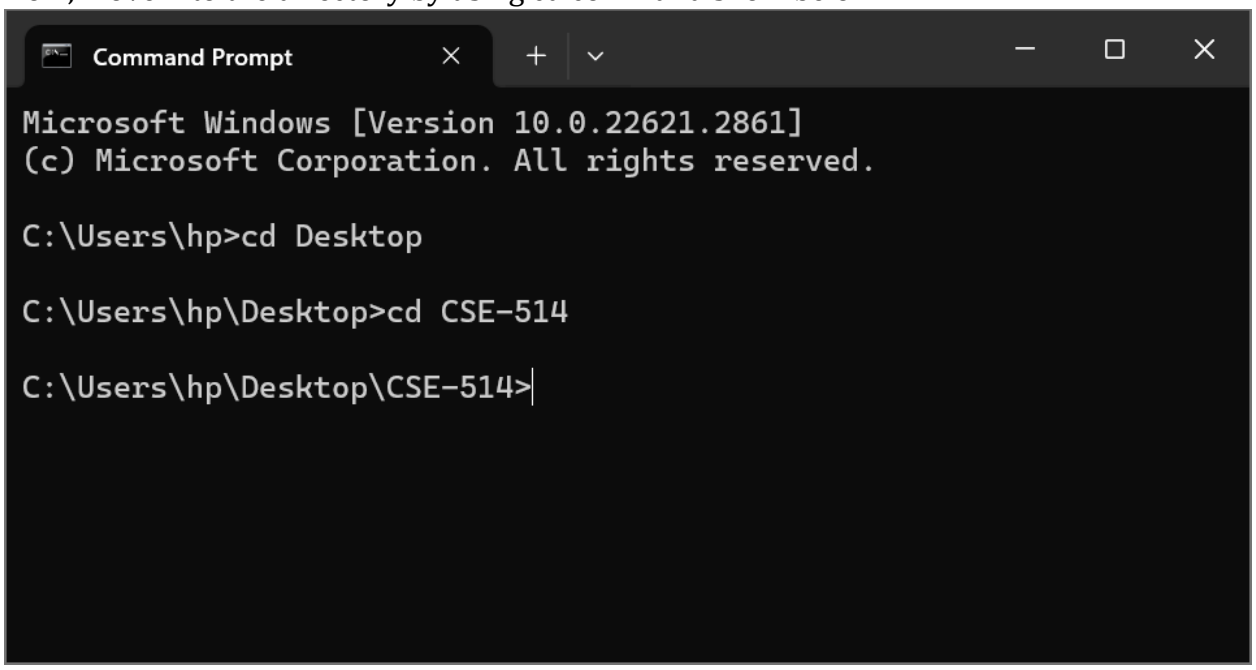
2. Once cmd prompt open go to DESKTOP using cd Desktop



3. Now create a Directory using mkdir or md command using your branch abbrevation and last 3 digit hall ticket number like md CSE-514.



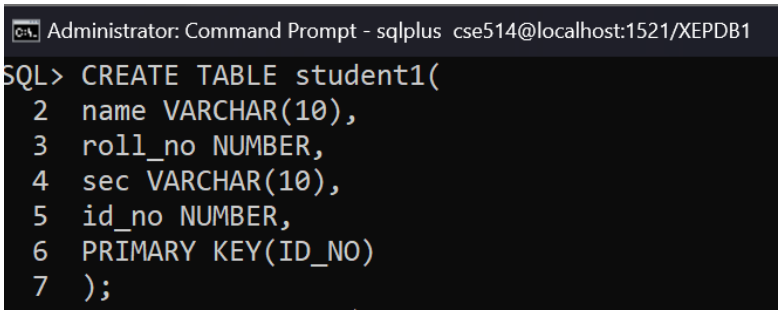
4. Now, move into the directory by using cd command show below.



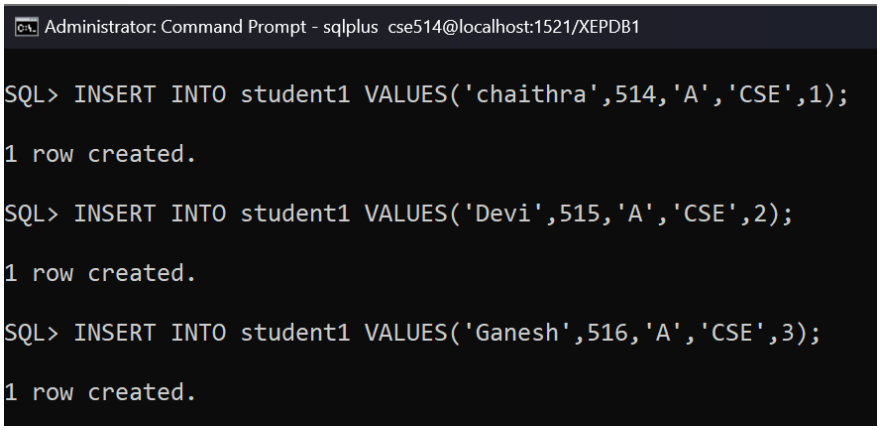
5. To Login, , Type sqlplus command enter username and password when system is prompted.

To implement a view level design using CREATE VIEW,ALTER VIEW and DELETE VIEW DDL commands:

CREATE student1 TABLE



INSERTING student1 VALUES



Creating view councellor1:

Inserting values into councellor1:

Selecting specific row:

```
Administrator: Command Prompt - sqlplus  cse514@localhost:1521/XEPDB1

SQL> INSERT INTO councellor1 VALUES('Ravi',516,7);

1 row created.

SQL> INSERT INTO councellor1 VALUES('Rajesh',509,8);

1 row created.

SQL> INSERT INTO councellor1 VALUES('Rakul',520,9);

1 row created.
```

Selecting specific row:

```
Administrator: Command Prompt - sqlplus  cse514@localhost:1521/XEPDB1

SQL> SELECT * FROM councellor1 WHERE id_no = 7;

NAME           ROLL_NO      ID_NO
---------- ---------- ----------
Ravi              516          7
```

Update:

```
Administrator: Command Prompt - sqlplus  cse514@localhost:1521/XEPDB1

SQL> UPDATE councellor1 SET name = 'Shruthi' WHERE id_no = 1;

1 row updated.

SQL> SELECT * FROM councellor1;

NAME           ROLL_NO      ID_NO
---------- ---------- ----------
Shruthi           514          1
Devi              515          2
Ganesh            516          3
Ravi              516          7
Rajesh            509          8
Rakul             520          9

6 rows selected.
```

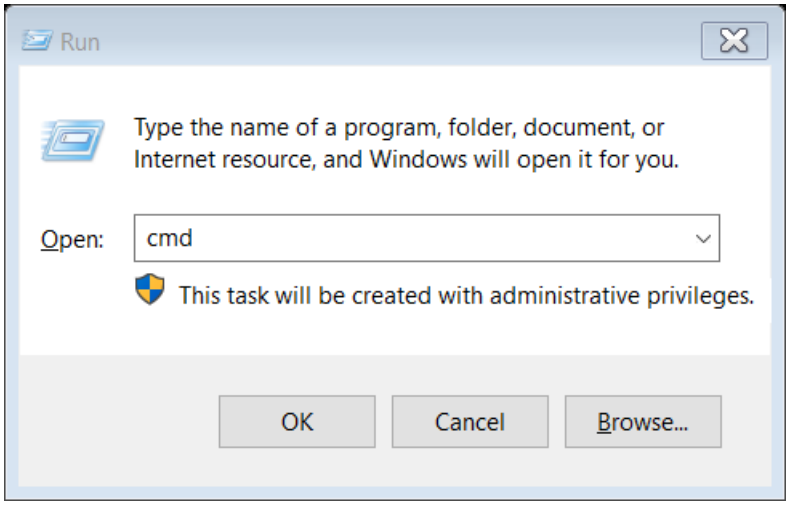
truncate or drop view

```
Administrator: Command Prompt - sqlplus  cse514@localhost:1521/XEPDB1

SQL> DROP VIEW councellor1;

View dropped.
```

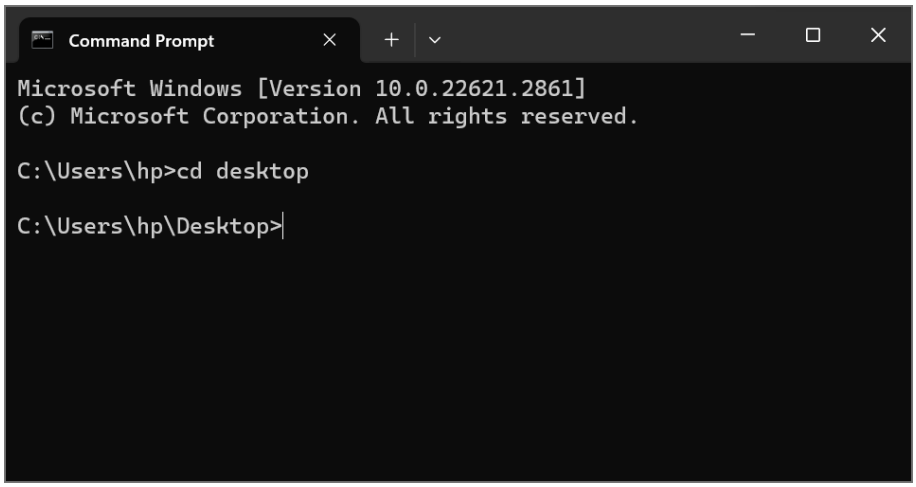
SQL queries to implement VIEWS for various database

SQL queries to perform RELATIONAL SET OPERATIONS

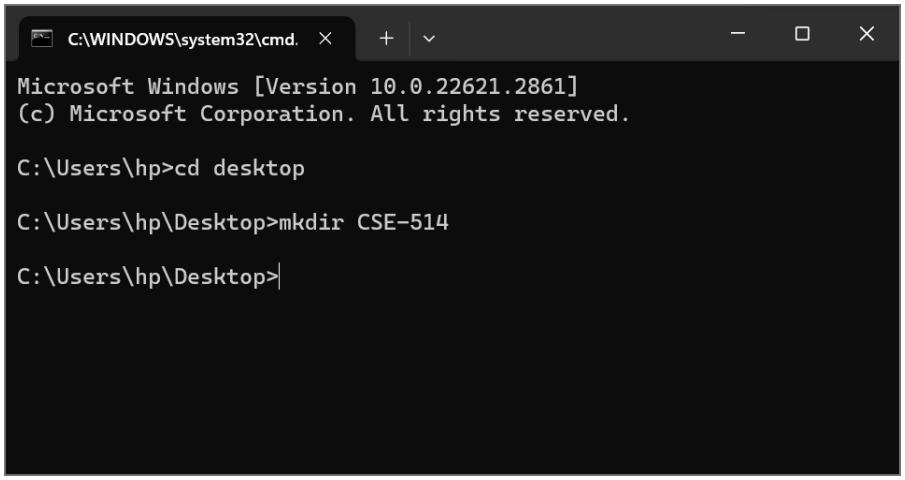
1. Open the command prompt Press WIN+R , type cmd



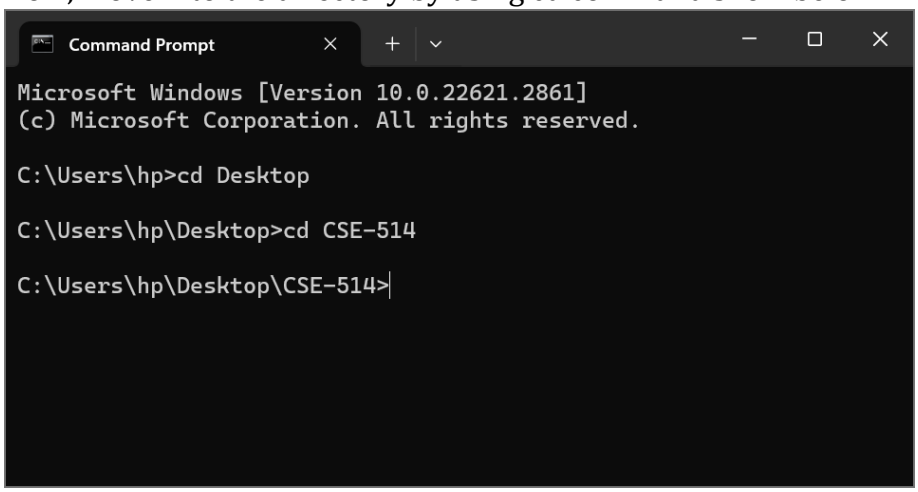
2. Once cmd prompt open go to DESKTOP using cd Desktop



3. Now create a Directory using mkdir or md command using your branch abbrevation and last 3 digit hall ticket number like md CSE-514.



4. Now, move into the directory by using cd command show below.



5. To Login, , Type sqlplus command enter username and password when system is prompted.

6. To perform RELATIONAL SET OPERATIONS (i.e. UNION, UNION ALL, INTERSECT, MINUS, CROSS JOIN, NATURAL JOIN).

CREATING STUDENT Table

```
SQL> CREATE TABLE student(
  2  roll_no int PRIMARY KEY,
  3  name VARCHAR2(20)
  4  );

Table created.
```

Inserting values into student table

```
Administrator: Command Prompt - sqlplus  cse514@localhost:1521/XEPDB1


SQL> INSERT INTO student VALUES(501, 'Abhi');

1 row created.

SQL> INSERT INTO student VALUES(502, 'Akhila');

1 row created.

SQL> INSERT INTO student VALUES(503, 'Bhavana');

1 row created.

SQL>
```

```
Administrator: Command Prompt - sqlplu
SQL> SELECT * FROM student
  2  ;

   ROLL_NO NAME
---------- --------------------
       501 Abhi
       502 Akhila
       503 Bhavana
```

CREATE EMPLOYEE TABLE

```
Administrator: Command Prompt - sqlplus  cse514@localhost:1521/XEPDB1
       502 Akhila
       503 Bhavana

QL> CREATE TABLE emplyee(
  2  emp_no int PRIMARY KEY,
  3  name VARCHAR2(20)
  4  );

able created.
```

Inserting values into employee table

```
Administrator: Command Prompt - sqlplus  cse514@localhost:1521/XEPDB1

SQL> INSERT INTO employee VALUES(504, 'Ahishek');

1 row created.

SQL> INSERT INTO employee VALUES(505, 'Arpita');

1 row created.
```

```
Administrator: Command Prompt - sqlplus  cse514@localhost:1521/XEPDB1
SQL> SELECT * FROM employee;

    EMP_NO NAME
---------- --------------------
       504 Ahishek
       505 Arpita
```

7. UNION OPERATION

UNION is used to combine the results of two or more SELECT statements.

Syntax:

SELECT * FROM table1
UNION
SELECT * FROM table2;

EX:

```
Administrator: Command Prompt - sqlplus  cse514@localhost:1521/XEPDB1
SQL> SELECT * FROM student
  2  UNION
  3  SELECT * FROM employee;

   ROLL_NO NAME
---------- --------------------
       501 Abhi
       502 Akhila
       503 Bhavana
       504 Ahishek
       505 Arpita
```

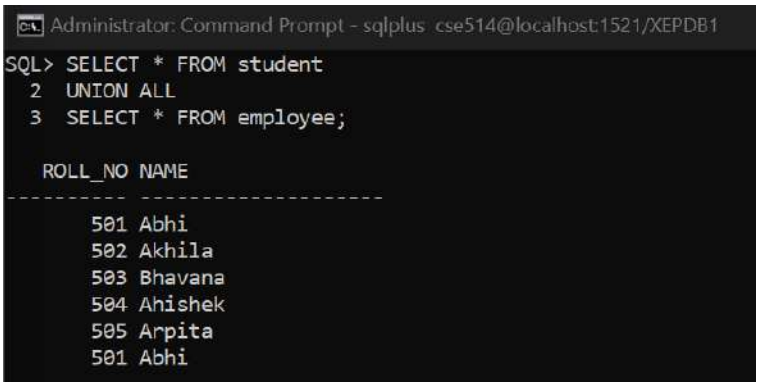
8. UNION ALL OPERATION

This operation is similar to UNION, but is also shows the duplicate rows.

Syntax:

SELECT * FROM table1
UNION ALL
SELECT * FROM table2;

EX:



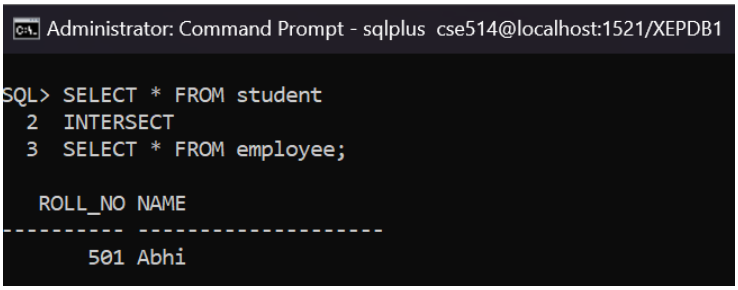
9. **INTERSECT OPERATIONS**

Intersect operation is used to combine two SELECT statements but it only returns the records which one common from both SELECT statements.

Syntax:

SELECT * FROM table1
INTERSECT
SELECT * FROM table2;

EX:



10. MINUS OPERATION

The MINUS operation combines results of two SELECT statements and returns only those in final result which belongs to first set of the result.

Syntax:

SELECT * FROM table1
MINUS
SELECT * FROM table2;

EX:

```
Administrator: Command Prompt - sqlplus  cse514@localhost:1521/XEPDB1
SQL> SELECT * FROM student
  2  MINUS
  3  SELECT * FROM employee;

   ROLL_NO NAME
---------- --------------------
       502 Akhila
       503 Bhavana
```

11. NATURAL JOIN OPERATION

It joins two tables based on same attribute name and data type. The resulting table will contain common column. To perform this operation there must be common attribute between two tables.

Syntax:

SELECT * FROM table1
NATURAL JOIN
table;

```
Administrator: Command Prompt - sqlplus  cse514@localhost:1521/XEPDB1
SQL> CREATE TABLE marks (
  2      roll_no INT,
  3      marks NUMBER(3),
  4      FOREIGN KEY (roll_no) REFERENCES students(roll_no)
  5  );

Table created.
```

EX:

```
Administrator: Command Prompt - sqlplus  cse514@localhost:1521/XEPDB1
SQL> SELECT * FROM student
  2  NATURAL JOIN
  3  marks;

   ROLL_NO NAME                      MARKS
---------- -------------------- ----------
        60 Abhi                        501
        70 Akhila                      502
```

12. CROSS JOIN OPERATION

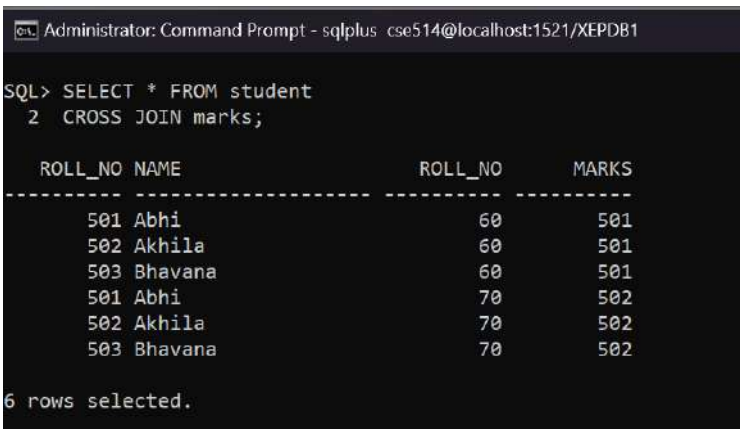
It will produce cross or cartesian product of two tables if there is no conditions specifies. The resulting table will contain all the attributes of both the tables including duplicate or common columns also.

Syntax:

SELECT * FROM table1
CROSS JOIN table2;

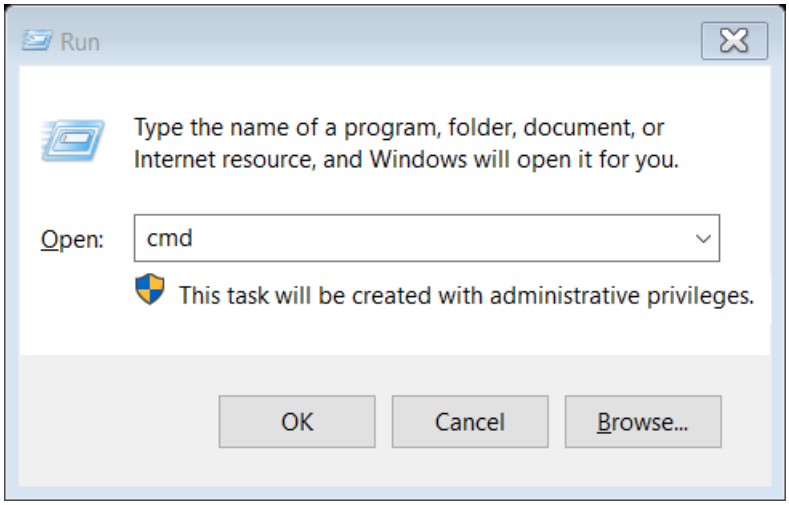
SQL queries to perform RELATIONAL SET OPERATIONS

EX:

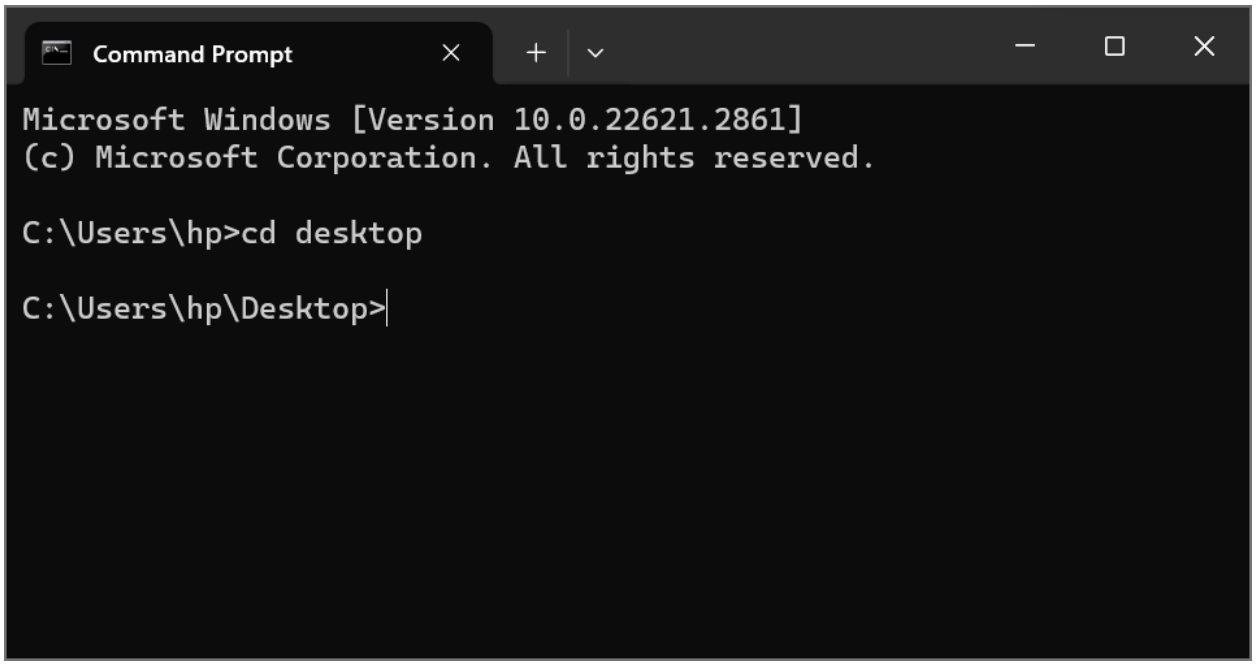


SQL queries to perform SPECIAL OPERATIONS

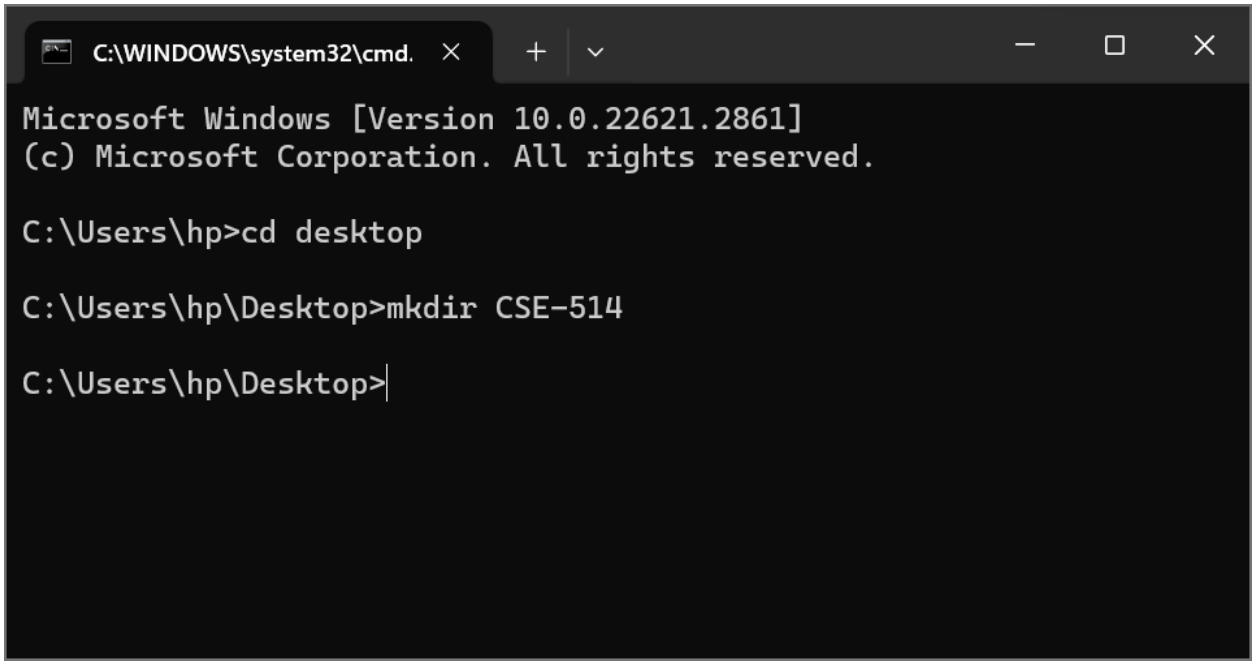
1. Open the command prompt Press WIN+R , type cmd



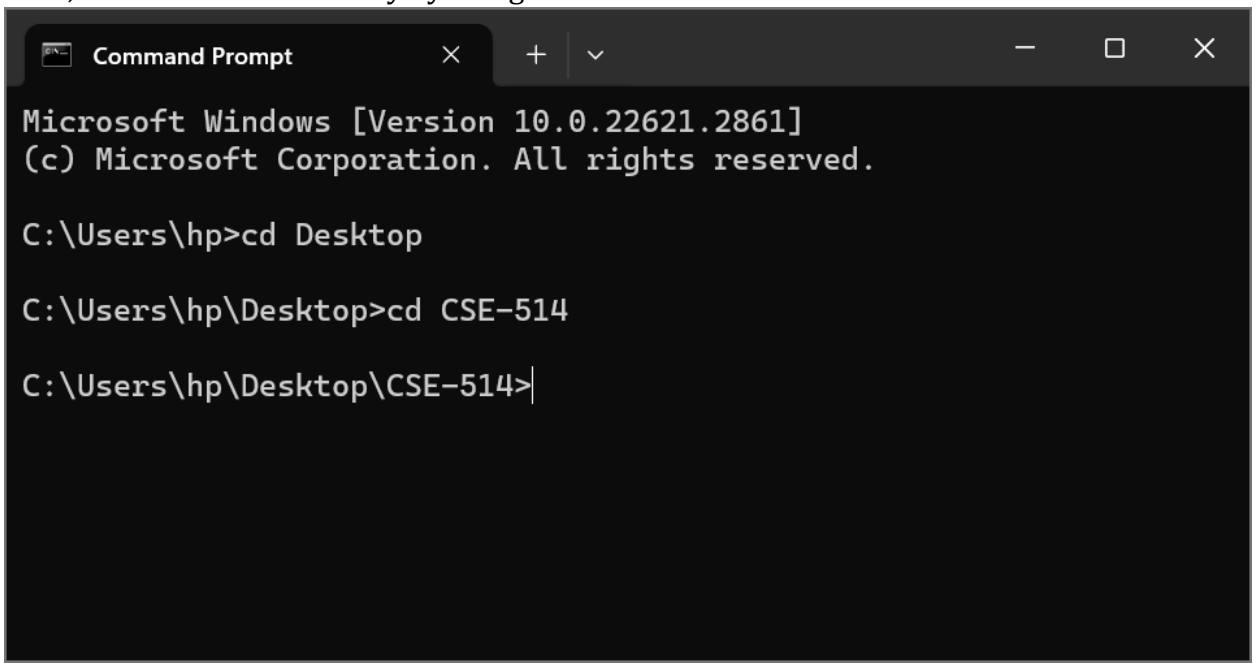
2. Once cmd prompt open go to DESKTOP using cd Desktop



3. Now create a Directory using mkdir or md command using your branch abbrevation and last 3 digit hall ticket number like md CSE-514.



4. Now, move into the directory by using cd command show below.



5. To Login, , Type sqlplus command enter username and password when system is prompted.

6. **CRAETE FACULTY TABLE**

```
Administrator: Command Prompt - sqlplus  cse514@localhost:1521/XEPDB1
SQL> CREATE TABLE faculty(
  2  id int PRIMARY KEY,
  3  name VARCHAR2(20) NOT NULL,
  4  age int,
  5  Salary NUMBER(6)
  6  );

Table created.
```

INSERTING VALUES TO FACULTY TABLE

```
Administrator: Command Prompt - sqlplus  cse514@localhost:1521/XEPDB1

SQL> INSERT INTO faculty VALUES(1,'Ravi',18,8000);

1 row created.

SQL> INSERT INTO faculty VALUES(2,'Raghu',19,3000);

1 row created.

SQL> INSERT INTO faculty VALUES(3,'Srujani',21,6000);

1 row created.

SQL> INSERT INTO faculty VALUES(4,'Dhoni',25,4500);

1 row created.

SQL> INSERT INTO faculty VALUES(5,'Sachin',28,7300);

1 row created.
```

```
SQL> SELECT * FROM faculty;

        ID NAME                          AGE     SALARY
---------- -------------------- ---------- ----------
         1 Ravi                         18       8000
         2 Raghu                        19       3000
         3 Srujani                      21       6000
         4 Dhoni                        25       4500
         5 Sachin                       28       7300
```

7. **IS NULL**

IS NULL operator is used to check the presence or absence of null values in a column.

Syntax:

SELECT column_name
FROM table_name
WHERE column_name IS NULL;

EX:

```
SQL> SELECT *
  2  FROM faculty
  3  WHERE SALARY IS NULL;

no rows selected
```

8. BETWEEN OPERATOR

BETWEEN operator returns information within given range of value.

Syntax:

SELECT * FROM table_name
WHERE column_name BETWEEN VALUE1 AND VALUE2;
EX:

```
Administrator: Command Prompt - sqlplus  cse514@localhost:1521/XEPDB1

SQL> SELECT *
  2  FROM faculty
  3  WHERE AGE between 20 AND 30;

        ID NAME                          AGE     SALARY
---------- -------------------- ---------- ----------
         3 Srujani                      21       6000
         4 Dhoni                        25       4500
         5 Sachin                       28       7300
```

9. LIKE OPERATOR

LIKE operator is used in a 'WHERE' clause to search for specified pattern in a column it is often used with your wild character.

- % represents zero or more character.
- - represents single character.

Syntax

SELECT * FROM table_name
WHERE column_name LIKE pattern;
Ex:

```
Administrator: Command Prompt - sqlplus  cse514@localhost:1521/XEPDB1

SQL> SELECT *
  2  FROM faculty
  3  WHERE name LIKE 'S%';

        ID NAME                          AGE     SALARY
---------- -------------------- ---------- ----------
         3 Srujani                      21       6000
         5 Sachin                       28       7300
```

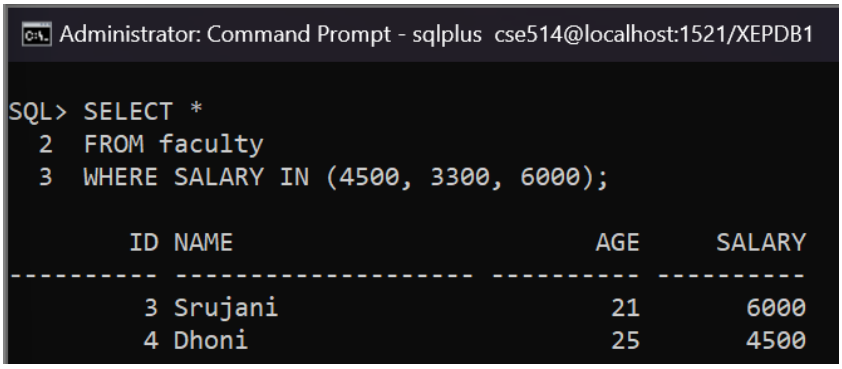
10.IN OPERATOR

The IN operator allows us to specify multiple values in a WHERE clause.

Syntax:

SELECT *
FROM table_name
WHERE column_name IN(VALUE 1, VALUE 2….);

EX:



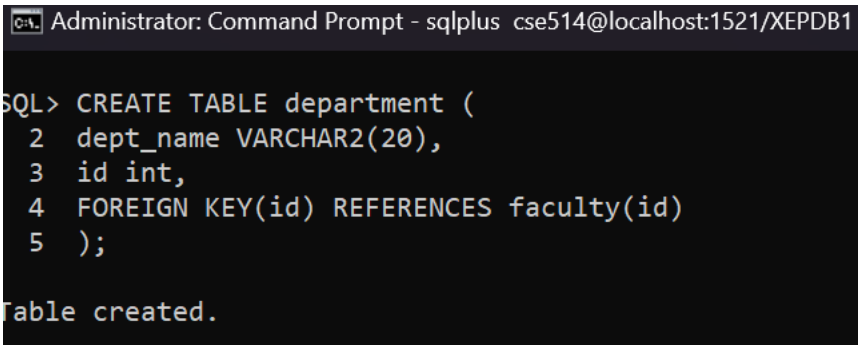
11. EXIST OPERATOR

EXIST operator is used to test for existence of any record in sub query.

Syntax:

SELECT * columm_name(S)
FROM table_name
WHERE EXISTS (SELECT column_name(S) FROM table_name WHERE CONDITION);

CREATING DEPARTMENT TABLE



INSERTING VALUES

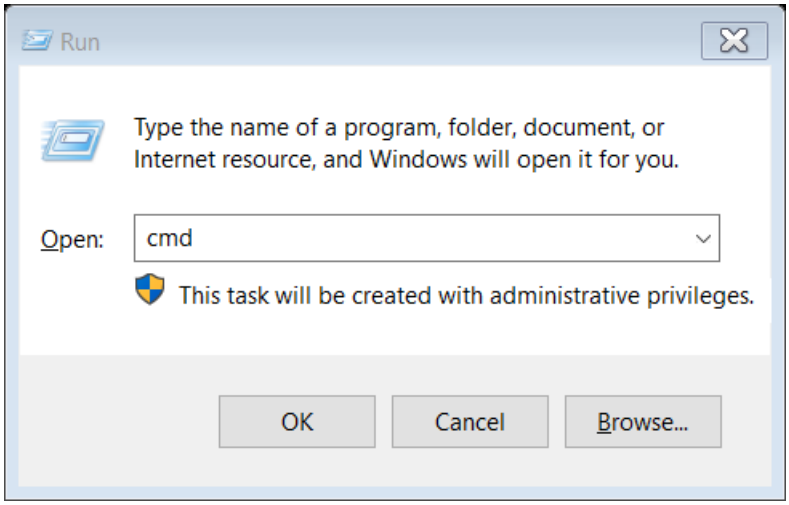
```
Administrator: Command Prompt - sqlplus  cse514@localhost:1521/XEPDB1
QL> INSERT INTO department values('cse',1);

 row created.

QL> INSERT INTO department values('csd',2);

 row created.

QL> INSERT INTO department values('csm',3);

 row created.
```

EX:

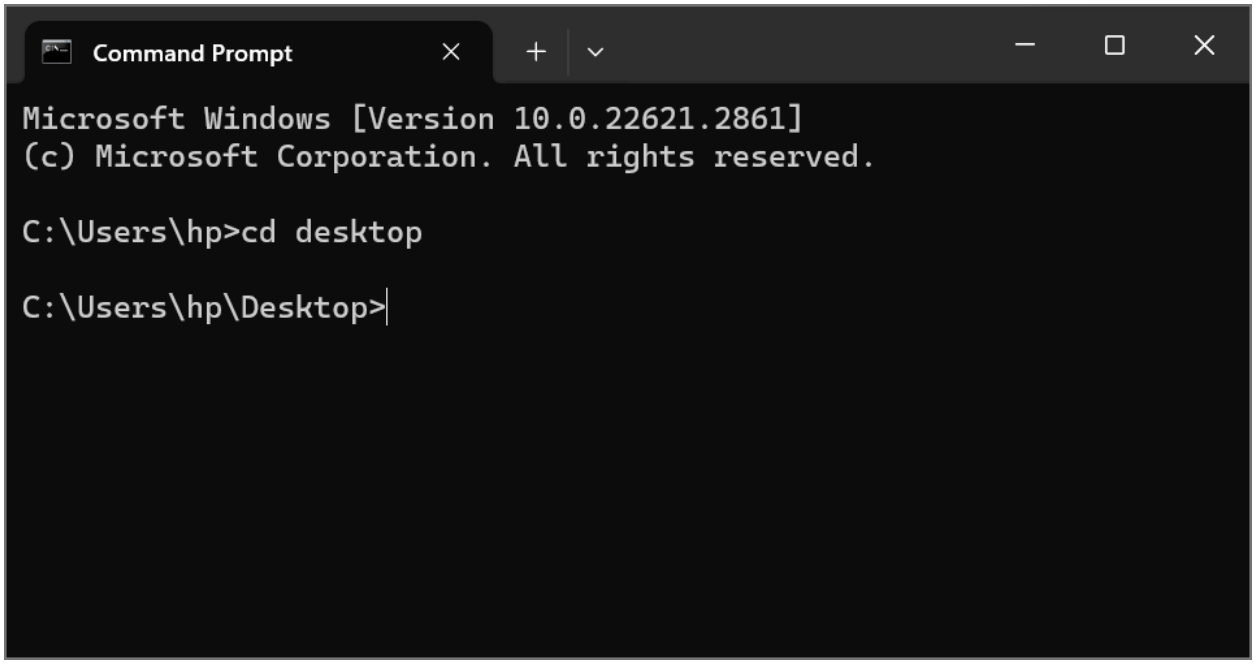
```
Administrator: Command Prompt - sqlplus  cse514@localhost:1521/XEPDB1


SQL> SELECT *
  2  FROM faculty
  3  WHERE EXISTS (SELECT 1 FROM department WHERE faculty.id = department.id);

        ID NAME                          AGE     SALARY
---------- -------------------- ---------- ----------
         1 Ravi                         18       8000
         2 Raghu                        19       3000
         3 Srujani                      21       6000
```

SQL queries to perform JOIN Operation

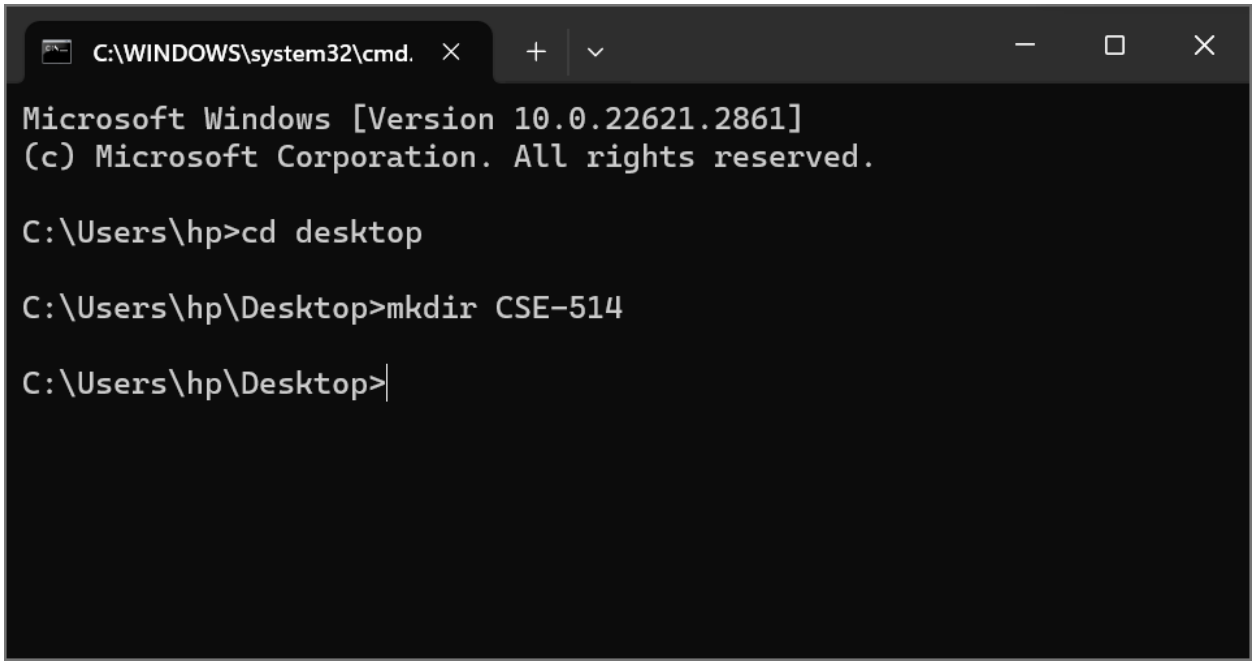
1. Open the command prompt Press WIN+R , type cmd



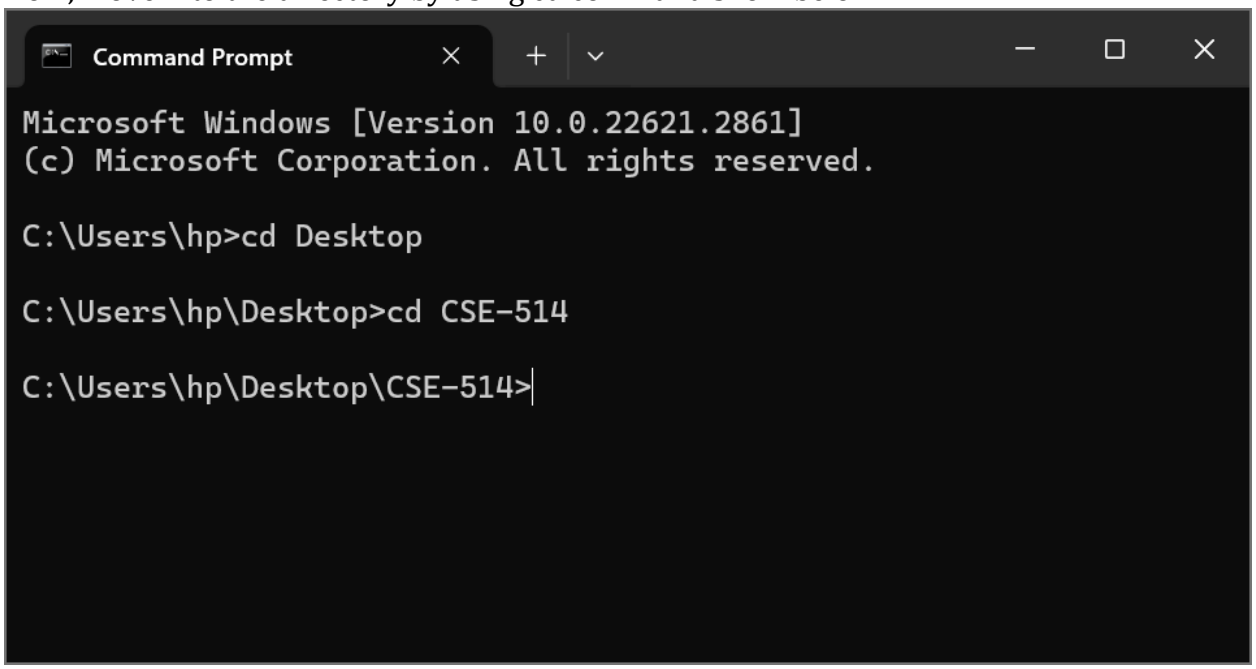
2. Once cmd prompt open go to DESKTOP using cd Desktop



3. Now create a Directory using mkdir or md command using your branch abbrevation and last 3 digit hall ticket number like md CSE-514.



4. Now, move into the directory by using cd command show below.



5. To Login, , Type sqlplus command enter username and password when system is prompted.

```
Administrator: Command Prompt - sqlplus  cse514@localhost:1521/XEPDB1
Microsoft Windows [Version 10.0.22621.2861]
(c) Microsoft Corporation. All rights reserved.

C:\Windows\System32>sqlplus cse514@localhost:1521/XEPDB1

SQL*Plus: Release 21.0.0.0.0 - Production on Fri Jan 12 20:01:41 2024
Version 21.3.0.0.0

Copyright (c) 1982, 2021, Oracle.  All rights reserved.

Enter password:
Last Successful login time: Fri Jan 12 2024 20:00:02 +05:30

Connected to:
Oracle Database 21c Express Edition Release 21.0.0.0.0 - Production
Version 21.3.0.0.0

SQL>
```

6. To perform SQL queries JOIN OPERATIONS (i.e. CONDITIONAL JOIN, EQUI JOIN, LEFT OUTER JOIN, RIGHT OUTER JOIN, FULL OUTER JOIN).

CREATE TABLE SAIL

```
Administrator: Command Prompt -
SQL> CREATE TABLE sail(
  2  sid NUMBER,
  3  sname VARCHAR2(5)
  4  );

Table created.
```

INSERTING VALUES

```
Administrator: Command Prompt - sqlplus  cse514@localhost:1521/XEPDB1

SQL> INSERT INTO sail VALues(1, 'aa');

1 row created.

SQL> INSERT INTO sail VALues(2, 'ab');

1 row created.

SQL> INSERT INTO sail VALues(3, 'ac');

1 row created.
```

```
Administrator: Command Prompt - sqlplus  cse514@localhost:1

SQL> SELECT* FROM sail;

       SID SNAME
---------- -----
         1 aa
         2 ab
         3 ac
```

CREATE BOAT TABLE

```
Administrator: Command Prompt - sqlplus  cse514@localhost:1521/
SQL> CREATE TABLE boat(
  2  sid NUMBER,
  3  bid VARCHAR2(2)
  4  );

Table created.
```

INSERTING VALUES

```
SQL> INSERT INTO boat VALUES(3,'b1');

1 row created.

SQL> INSERT INTO boat VALUES(4,'b2');

1 row created.

SQL> INSERT INTO boat VALUES(5,'b3');

1 row created.
```

```
SQL> SELECT * FROM boat;

       SID BI
---------- --
         3 b1
         4 b2
         5 b3
```

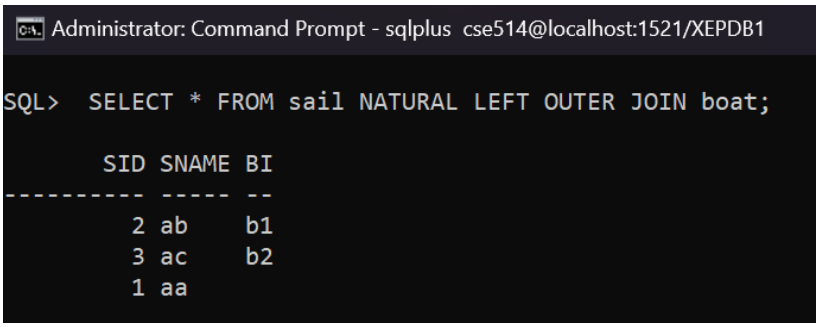
7. LEFT OUTER JOIN

It is a method of joining.

Syntax:

SELECT column_name FROM table_name
LEFT OUTER JOIN table_name;

EX:



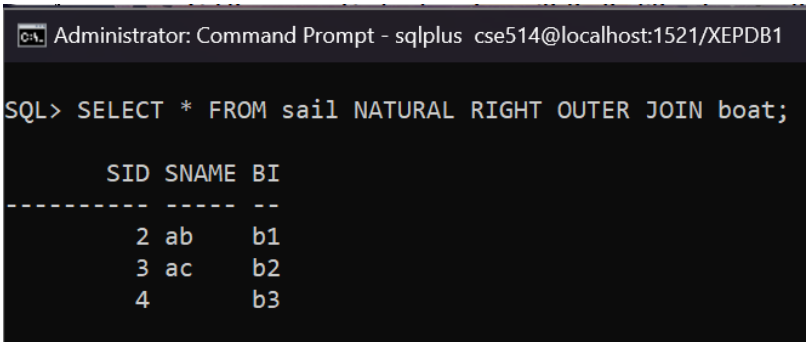
8. RIGHT OUTER JOIN

It is used to join tables.

Syntax:

SELECT column_name FROM table_name
RIGHT OUTER JOIN table_name;

EX:



9. FULL OUTER JOIN

It is used to join or combine tables.

Syntax

SELECT column_name FROM table_name
FULL OUTER JOIN table_name;

EX:

```
Administrator: Command Prompt - sqlplus  cse514@localhost:1521/XEPDB1

SQL> SELECT * FROM sail NATURAL FULL OUTER JOIN boat;

       SID SNAME BI
---------- ----- --
         2 ab    b1
         3 ac    b2
         4       b3
         1 aa
```

10.CONDITIONAL JOIN

It allows user to join tables when specified column values meet certain criteria.

Syntax:

SELECT column_name FROM table_name
WHERE (condition);

Ex:

```
Administrator: Command Prompt - sqlplus  cse514@localhost:1521/XEPDB1

SQL> SELECT *
  2  FROM sail
  3  JOIN boat ON sail.sid > boat.sid;

       SID SNAME        SID BI
---------- ----- ---------- --
         3 ac             2 b1
```

11.EQUI JOIN

It joins the columns whose values are matching

Syntax

SELECT column_name FROM table_name
JOIN table_name USING (column_name)

EX:

```
Administrator: Command Prompt - sqlplus  cse514@localhost:1521/XEPDB1

SQL> SELECT * FROM sail
  2  JOIN boat USING (sid);

       SID SNAME BI
---------- ----- --
         2 ab    b1
         3 ac    b2
```

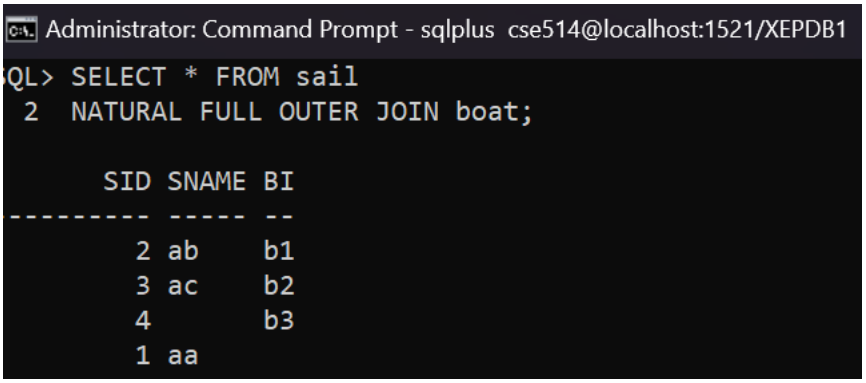
12. FULL OUTER JOIN

A full outer join is a type of relational database join that combines the results of both left outer join and right outer join.

Syntax

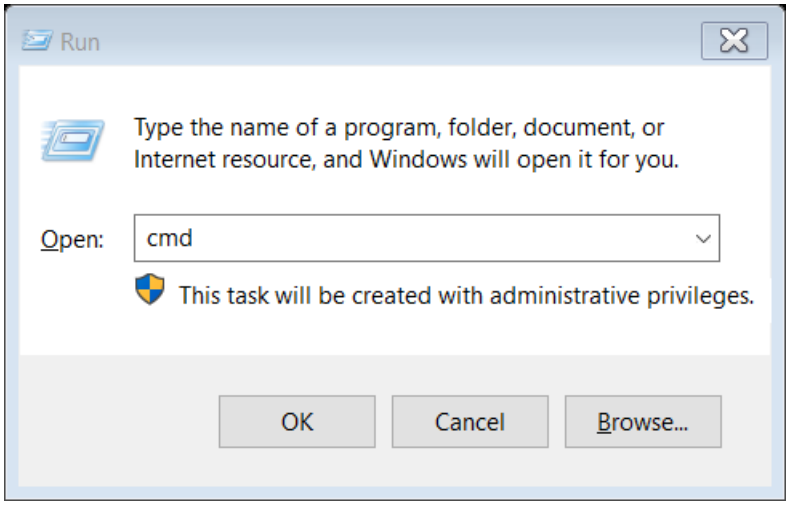
```
SELECT *
FROM table1
FULL OUTER JOIN table2
ON table1.column = table2.column;
```

EX:

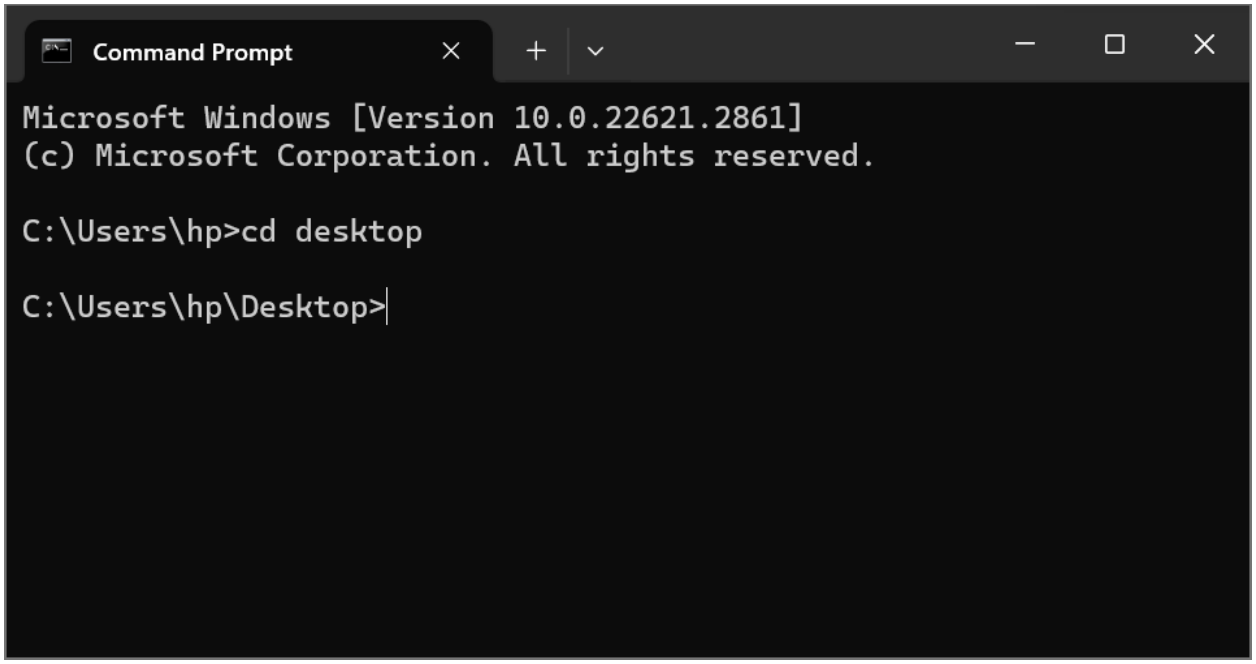


SQL queries to perform AGGREGATE operation

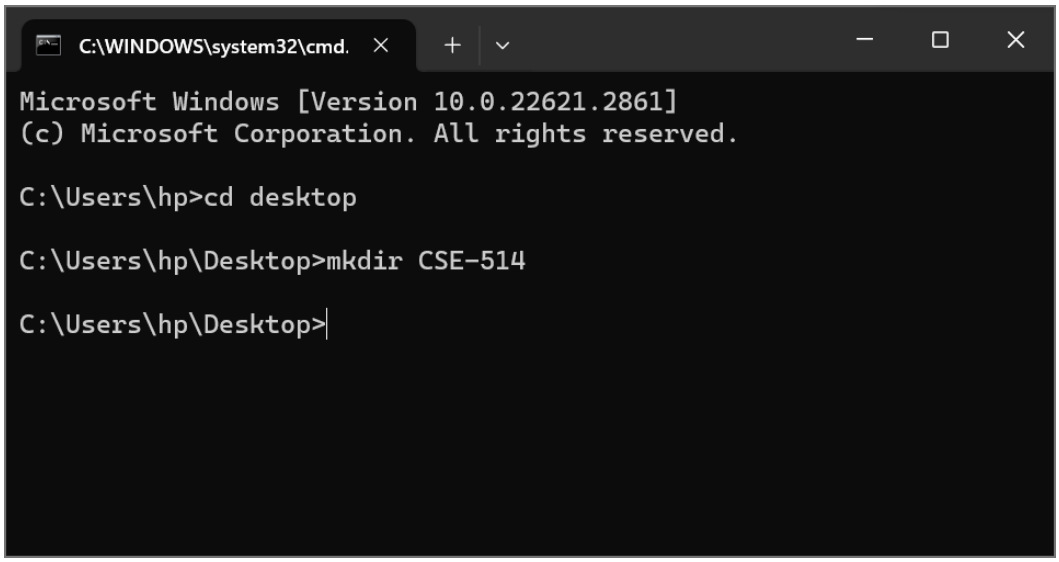
1. Open the command prompt Press WIN+R , type cmd



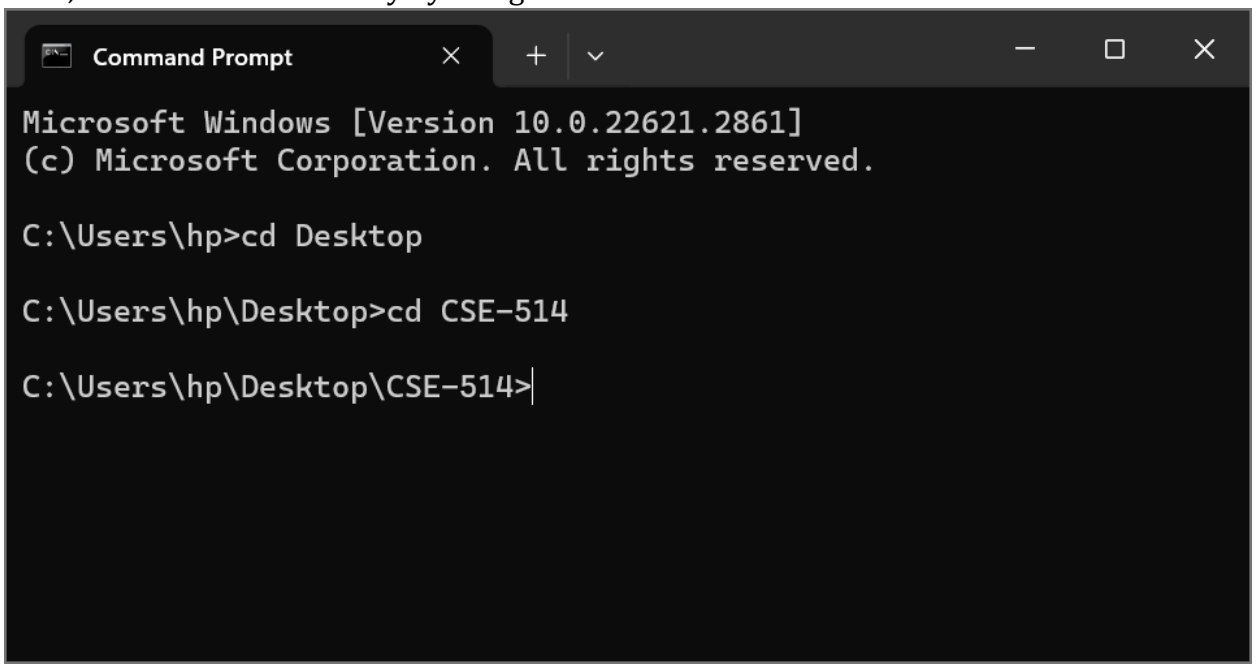
2. Once cmd prompt open go to DESKTOP using cd Desktop



3. Now create a Directory using mkdir or md command using your branch abbrevation and last 3 digit hall ticket number like md CSE-514.

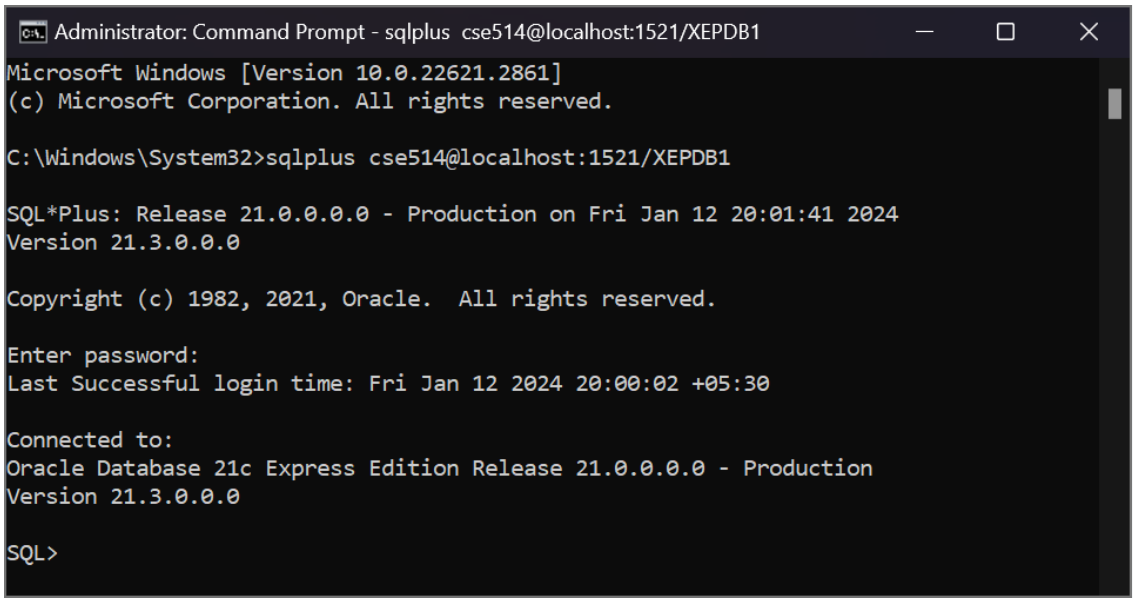


4. Now, move into the directory by using cd command show below.



5. To Login, , Type sqlplus command enter username and password when system is prompted.

SQL queries to perform AGGREGATE operation



TO implement SQL queries to perform AGGREGATE OPERATIONS (i.e. SUM, COUNT, AVG, MIN,MAX).

CREATE Department TABLE :

```
SQL> CREATE TABLE DEPARTMENT
  2  (DEPT_NAME VARCHAR2(20),
  3  BUILDING VARCHAR2(15),
  4  BUDGET NUMERIC(12,2) CHECK (BUDGET > 0),
  5  PRIMARY KEY (DEPT_NAME)
  6  );

Table created.
```

```
SQL> INSERT INTO DEPARTMENT(DEPT_NAME,BUILDING,BUDGET)
  2  VALUES('CSE','Watson',29000);

1 row created.
```

CREATE Instructor TABLE :

SQL queries to perform AGGREGATE operation

```
SQL> CREATE TABLE Instructor
  2  (
  3  ID VARCHAR2(20) NOT NULL,
  4  Name VARCHAR2(15),
  5  dept_name VARCHAR2(25),
  6  Salary NUMERIC(5,2) CHECK(Salary>29000),
  7  PRIMARY KEY(ID),
  8  FOREIGN KEY(dept_name) REFERENCES Department(dept_name) ON DELETE SET NULL
  9  );

Table created.
```

AVERAGE : The AVG() function returns the average value of a numeric column.

AVERAGE SYNTAX :

SELECT AVG(*column_name*)
FROM *table_name*
WHERE *condition;*

```
SQL> SELECT avg(budget)
  2  FROM Department
  3  WHERE Budget>0;

AVG(BUDGET)
-----------
      29000

SQL>
```

SUM : The SUM() function returns the total sum of a numeric column.

SUM SYNTAX :

SELECT SUM(*column_name*)
FROM *table_name*
WHERE *condition*;

SUM EXAMPLE :

```
SQL> SELECT avg(budget)
  2  FROM Department
  3  WHERE Budget>0;

AVG(BUDGET)
-----------
      29000
```

MAXIMUM : The MAX() function returns the largest value of the selected column.

MAX SYNTAX :

```
SELECT MAX(column_name)
FROM table_name
WHERE condition;
```

MAX EXAMPLE :

```
SQL> SELECT MAX(budget)
  2  FROM Department
  3  WHERE Budget>0;

MAX(BUDGET)
-----------
      29000
```

MINIMUM : The MIN() function returns the smallest value of the selected column.

MIN SYNTAX :

```
SELECT MIN(column_name)
FROM table_name
WHERE condition;
```

MIN EXAMPLE:

```
SQL> SELECT MIN(budget)
  2  FROM Department
  3  WHERE Budget>0;

MIN(BUDGET)
-----------
      29000
```

COUNT : The COUNT() function returns the number of rows that matches a specified criterion.

COUNT SYNTAX :

```
SELECT COUNT(column_name)
FROM table_name
WHERE condition;
```

COUNT EXAMPLE :

```
SQL> SELECT count(budget)
  2  FROM Department
  3  WHERE Budget>0;

COUNT(BUDGET)
-------------
            1
```

GROUP BY : The GROUP BY statement groups rows that have the same values into summary rows, like "find the number of customers in each country".

The GROUP BY statement is often used with aggregate functions (COUNT(), MAX(), MIN(), SUM(), AVG()) to group the result-set by one or more columns.

GROUP BY SYNTAX :

SELECT *column_name(s)*
FROM *table_name*
WHERE *condition*
GROUP BY *column_name(s)*
ORDER BY *column_name(s);*

GROUP BY EXAMPLE :

```
SQL> SELECT budget
  2  FROM Department
  3  WHERE budget>0
  4  GROUP BY budget;

    BUDGET
----------
     29000
```

HAVING : The HAVING clause was added to SQL because the WHERE keyword cannot be used with aggregate functions.

HAVING SYNTAX :

SELECT *column_name(s)*
FROM *table_name*

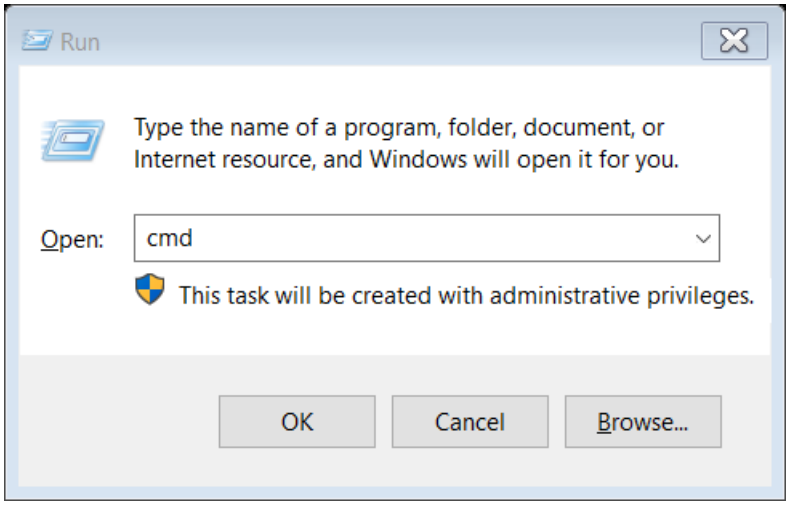
GROUP BY *column_name(s)*
HAVING *condition;*

HAVING EXAMPLE :

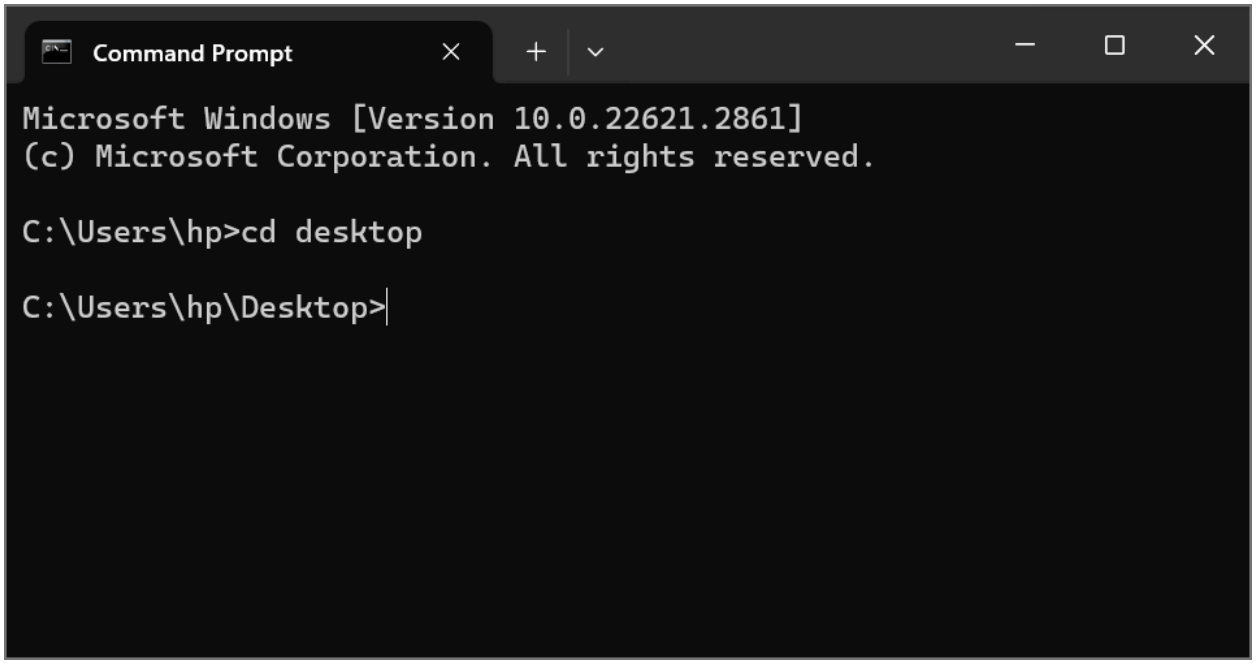
```
SQL> SELECT budget
  2  FROM Department
  3  GROUP BY budget
  4  HAVING budget>0;

    BUDGET
----------
     29000
```

ORACLE BUILT-IN function

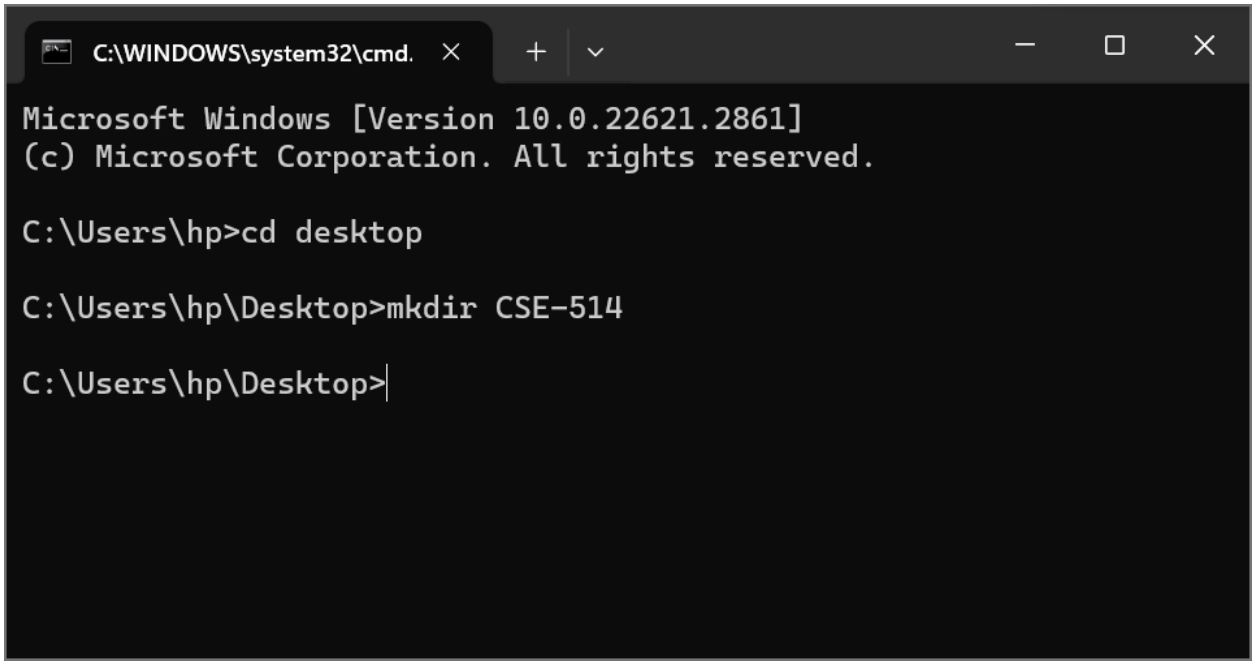
1. Open the command prompt Press WIN+R , type cmd



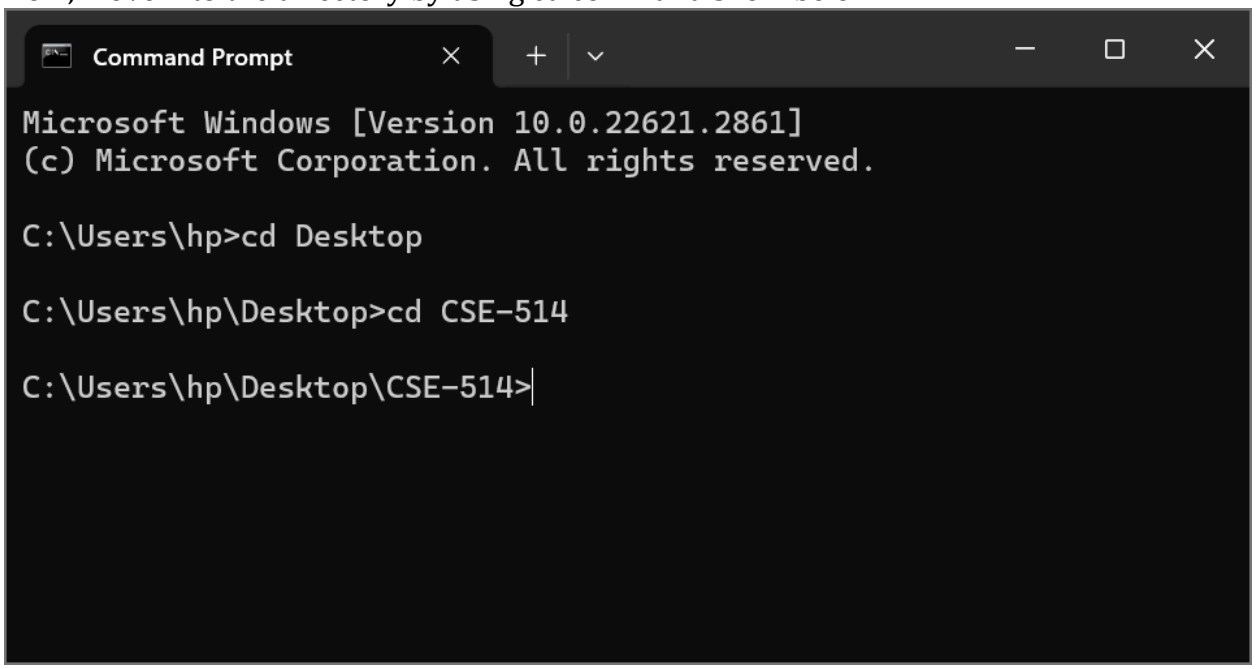
2. Once cmd prompt open go to DESKTOP using cd Desktop



3. Now create a Directory using mkdir or md command using your branch abbrevation and last 3 digit hall ticket number like md CSE-514.



4. Now, move into the directory by using cd command show below.



5. To Login, , Type sqlplus command enter username and password when system is prompted.

```
Administrator: Command Prompt - sqlplus  cse514@localhost:1521/XEPDB1
Microsoft Windows [Version 10.0.22621.2861]
(c) Microsoft Corporation. All rights reserved.

C:\Windows\System32>sqlplus cse514@localhost:1521/XEPDB1

SQL*Plus: Release 21.0.0.0.0 - Production on Fri Jan 12 20:01:41 2024
Version 21.3.0.0.0

Copyright (c) 1982, 2021, Oracle.  All rights reserved.

Enter password:
Last Successful login time: Fri Jan 12 2024 20:00:02 +05:30

Connected to:
Oracle Database 21c Express Edition Release 21.0.0.0.0 - Production
Version 21.3.0.0.0

SQL>
```

6. To perform SQL queries to perform ORACLE BUILT-IN FUNCTIONS (DATE, TIME).

CREATE INSTRUCTOR TABLE

```
SQL> CREATE TABLE INSTRUCTOR (
  2       ID VARCHAR2(5),
  3       NAME VARCHAR2(50) NOT NULL,
  4       DEPT_NAME VARCHAR2(50),
  5       SALARY NUMBER(8,2) CHECK (SALARY > 29000),
  6       PRIMARY KEY (ID)
  7  );

Table created.
```

INSERTING VALUES

```
SQL> INSERT INTO INSTRUCTOR VALUES('501','Abhi','cse',65000);

1 row created.

SQL> INSERT INTO INSTRUCTOR VALUES('502','Bhavana','csd',72000);

1 row created.

SQL> INSERT INTO INSTRUCTOR VALUES('503','Chai','csm',30000);

1 row created.

SQL> INSERT INTO INSTRUCTOR VALUES('504','Dev','mec',80000);

1 row created.
```

```
SQL> SELECT * FROM INSTRUCTOR;

ID    NAME
----- --------------------------------------------------
DEPT_NAME                                           SALARY
-------------------------------------------------- ----------
501   Abhi
cse                                                 65000

502   Bhavana
csd                                                 72000

503   Chai
csm                                                 30000
```

UPPERCASE

Syntax

SELECT UPPER(COL_name) FROM table_name;

EX:

```
SQL> SELECT UPPER(Name) from INSTRUCTOR;

UPPER(NAME)
----------------------------------------
ABHI
BHAVANA
CHAI
DEV
```

LOWER CASE

Syntax

SELECT LOWER(COL name) FROM table_name;

EX:

```
SQL> SELECT LOWER(Name) from INSTRUCTOR;

LOWER(NAME)
----------------------------------------
abhi
bhavana
chai
dev
```

INIT CAP

Syntax

SELECT Upper('String'), Lower('String'),
INITCAP ('String') FROM table name;

EX:

```
SQL> SELECT Upper ('hi friends'), Lower('HI FRIENDS'),
  2  INITCAP ('hi friends') FROM dual;

UPPER('HIF LOWER('HIF INITCAP('H
---------- ---------- ----------
HI FRIENDS hi friends Hi Friends
```

LENGTH

Syntax:

SELECT LENGTH('String') FROM table name;

EX:

```
SQL> SELECT LENGTH('HELLO WORLD') from dual;

LENGTH('HELLOWORLD')
--------------------
                  11
```

SUBSTR

Syntax

SELECT SUBSTR('String', index 1, index2) from table name;

EX:

```
SQL> SELECT SUBSTR('Hello world',3,7) from dual;

SUBSTR(
-------
llo wor
```

REPLACE

Syntax

SELECT REPLACE ('String1', 'Sub String', 'String 2') from table name;

EX:

```
SQL> SELECT REPLACE('Hello world','world','India') from dual;

REPLACE('HE
-----------
Hello India
```

INSTR

LPAD
Syntax
SELECT LPAD ('String', 20, '*') from dual;
EX:

```
SQL> SELECT LPAD ('Hello world', 20, '*') from dual;

LPAD('HELLOWORLD',20
--------------------
*********Hello world
```

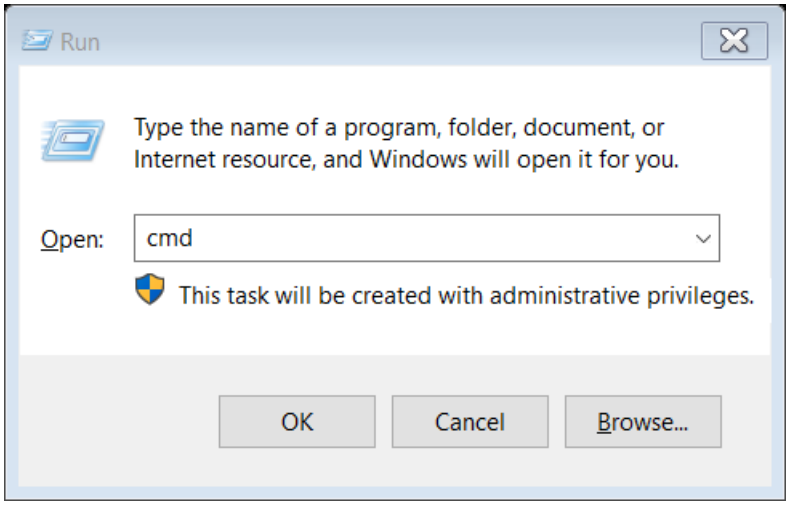
RPAD
Syntax
SELECT RPAD ('String', 20, '*') from dual;
EX:

```
SQL> SELECT RPAD ('Hello world', 20, '*') from dual;

RPAD('HELLOWORLD',20
--------------------
Hello world*********
```

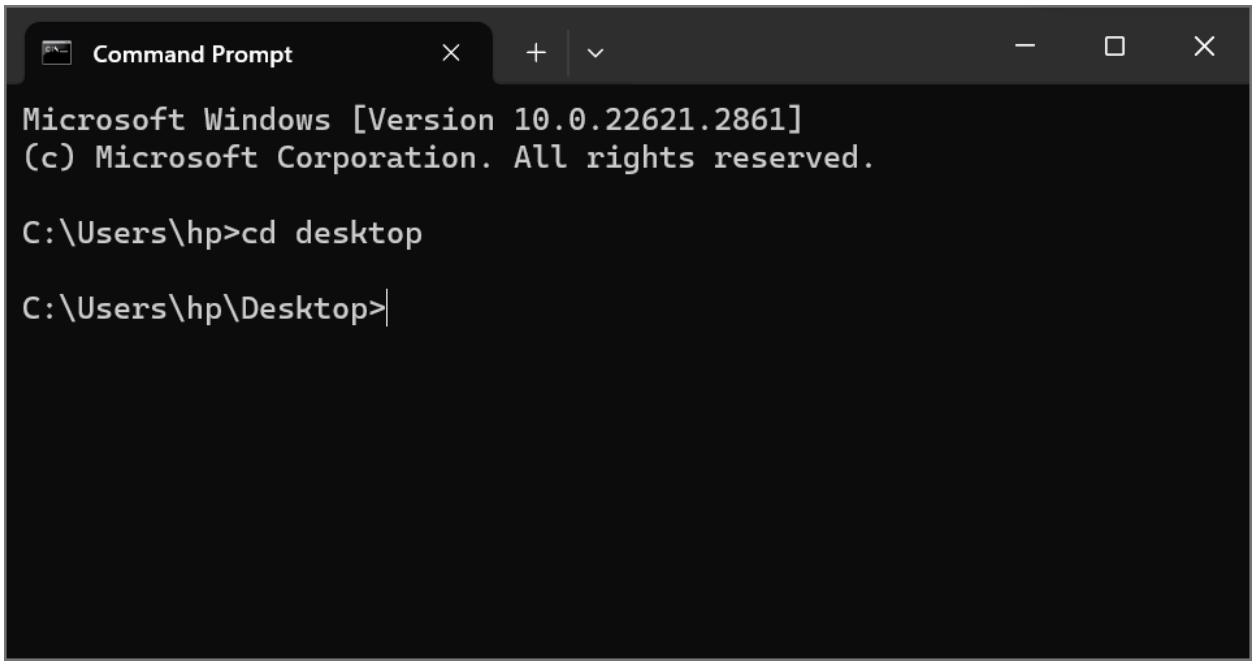
CONCAT
Syntax
SELECT CONCAT(column 1, column2) from table name;

SQL queries to perform key constraints

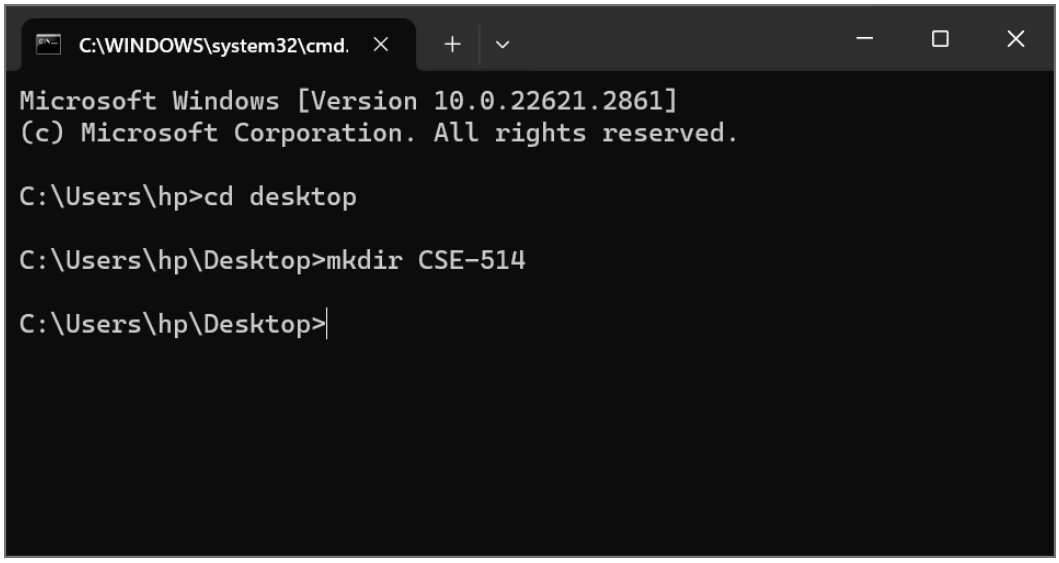
1. Open the command prompt Press WIN+R , type cmd



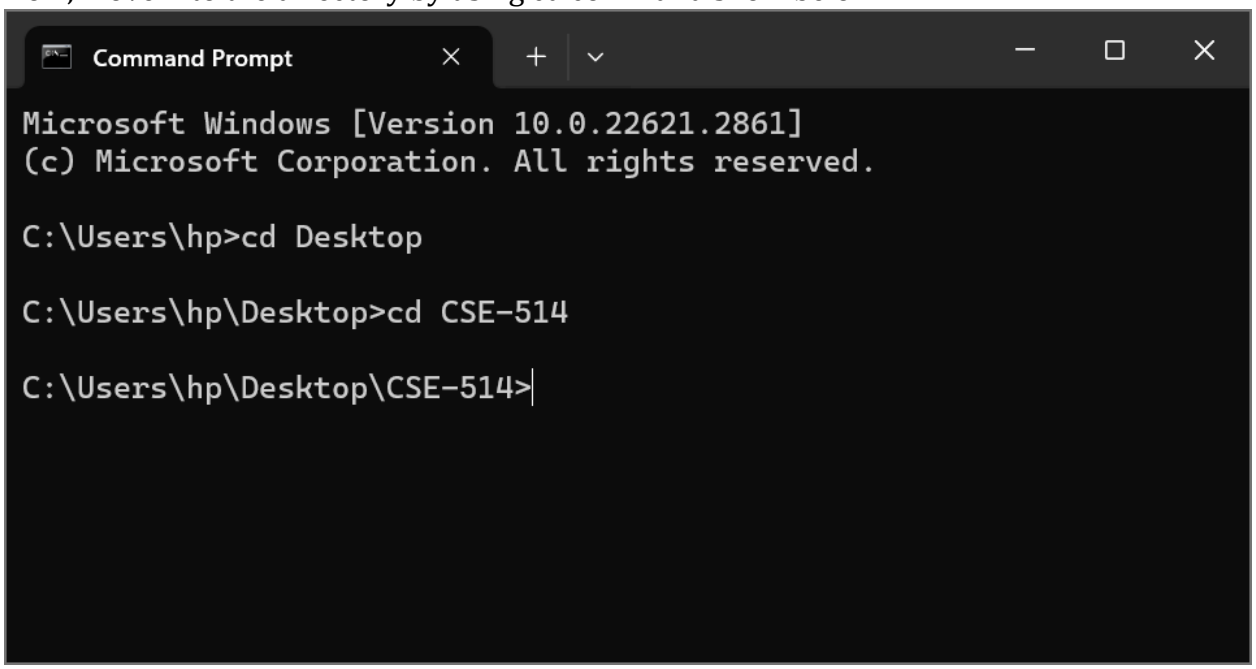
2. Once cmd prompt open go to DESKTOP using cd Desktop



3. Now create a Directory using mkdir or md command using your branch abbrevation and last 3 digit hall ticket number like md CSE-514.

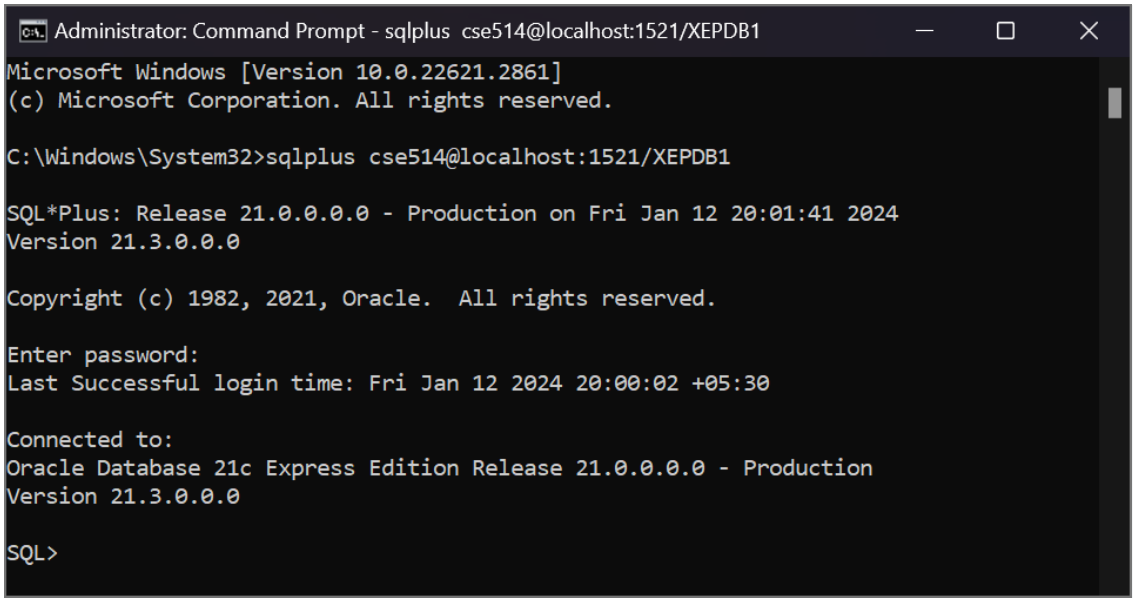


4. Now, move into the directory by using cd command show below.



5. To Login, , Type sqlplus command enter username and password when system is prompted.

SQL queries to perform key constraints



To Implement SQL queries to perform KEY CONSTRAINTS (i.e. PRIMARY KEY, FOREIGN KEY, UNIQUE NOT NULL, CHECK, DEFAULT).

Types of SQL Constraints.

1. NOT NULL - Ensures that a column cannot have a NULL value

2. UNIQUE - Ensures that all values in a column are different

3. PRIMARY KEY - A combination of a NOT NULL and UNIQUE. Uniquely I Identifies each row in a table

4. FOREIGN KEY - Uniquely identifies a row/record in another table

5. CHECK - Ensures that all values in a column satisfies a specific condition

6. DEFAULT - Sets TO a default value for a column when no value is specified

PRIMARY KEY: A primary key is a field which can uniquely identify each row in table and this constraint is used to specify a field as primary key.

```
SQL> CREATE TABLE student5(
  2  ID NUMBER,
  3  NAME VARCHAR2(20),
  4  ADDRESS VARCHAR2(20)
  5  );

Table created.

SQL>
```

FOREIGN KEY: A foreign key is a field which can uniquely each row in another table.

```
SQL> CREATE TABLE orders5(
  2  o_id NUMBER NOT NULL,
  3  c_id NUMBER,
  4  PRIMARY KEY(o_id),
  5  FOREIGN KEY(c_id)REFERENCES customer(c_id)
  6  );
```

```
Table created.

SQL>
```

UNIQUE: This constraint when specified with a column, tells that the values in the column must be unique i.e., the values in any row of a column must not be repeated.

```
SQL> CREATE TABLE student3(
  2  id NUMBER UNIQUE,
  3  name VARCHAR2(20),
  4  address VARCHAR2(20)
  5  );

Table created.

SQL>
```

NOT NULL: This constraint tells that we cannot store a null value in a column.

```
SQL> CREATE TABLE student3(
  2  ID NUMBER,
  3  NAME VARCHAR2(20) NOT NULL,
  4  ADDRESS VARCHAR2(20)
  5  );

Table created.

SQL>
```

DEFAULT: This constraint specifies a default value for the column when no value is specified by the user.

```
SQL> CREATE TABLE student6(
  2  ID NUMBER,
  3  NAME VARCHAR2(20) NOT NULL,
  4  AGE NUMBER DEFAULT 18
  5  );

Table created.

SQL>
```

CHECK: This constraint helps to validate the value for the column to meet a particular condition i.e. it helps to ensure that the value stored in a column meets a specific condition.

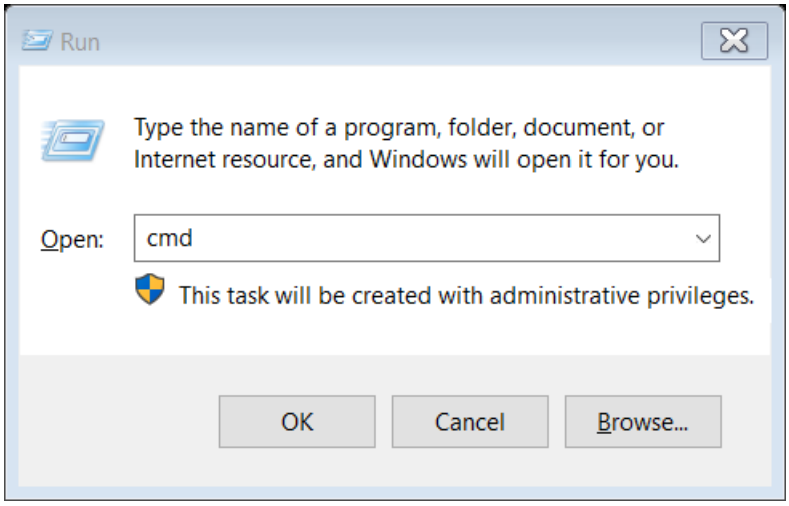
SQL queries to perform key constraints

```
SQL> CREATE TABLE student8(
  2  id NUMBER NOT NULL,
  3  NAME VARCHAR2(20) NOT NULL,
  4  AGE NUMBER NOT NULL CHECK(AGE>=18)
  5  );

Table created.

SQL>
```

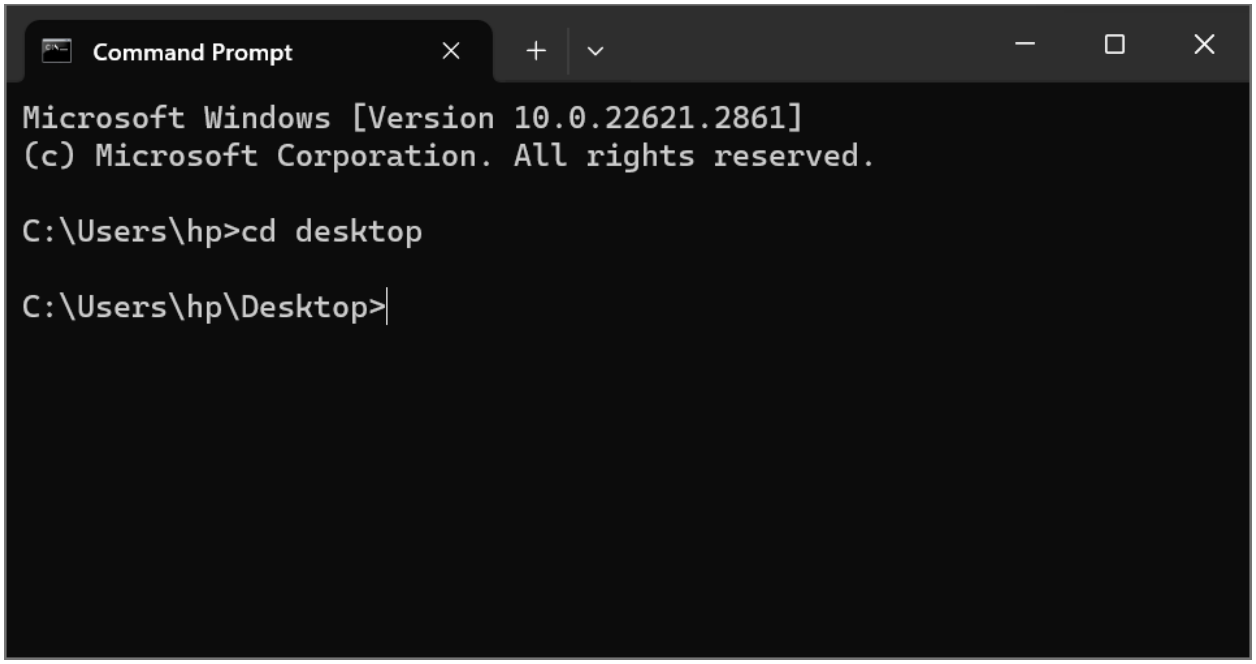
Conclusion: In this lab, we have practiced KEY CONSTRAINTS PRIMARY KEY, FOREIGN KEY, UNIQUE, NOT NULL, CHECK, DEFAULT on user created tables.

PL/SQL program to perform factorial numbers

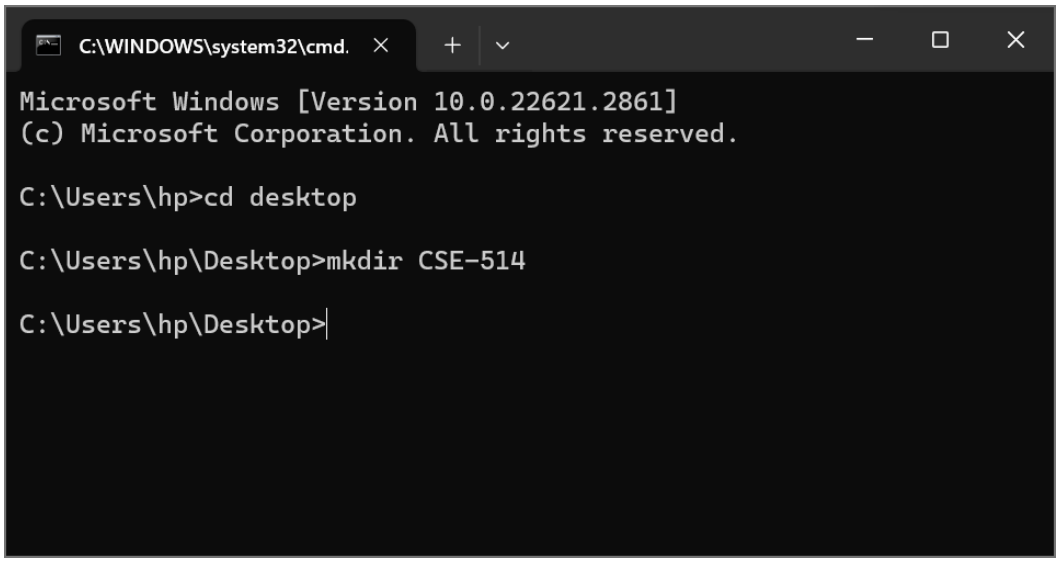
1. Open the command prompt Press WIN+R , type cmd



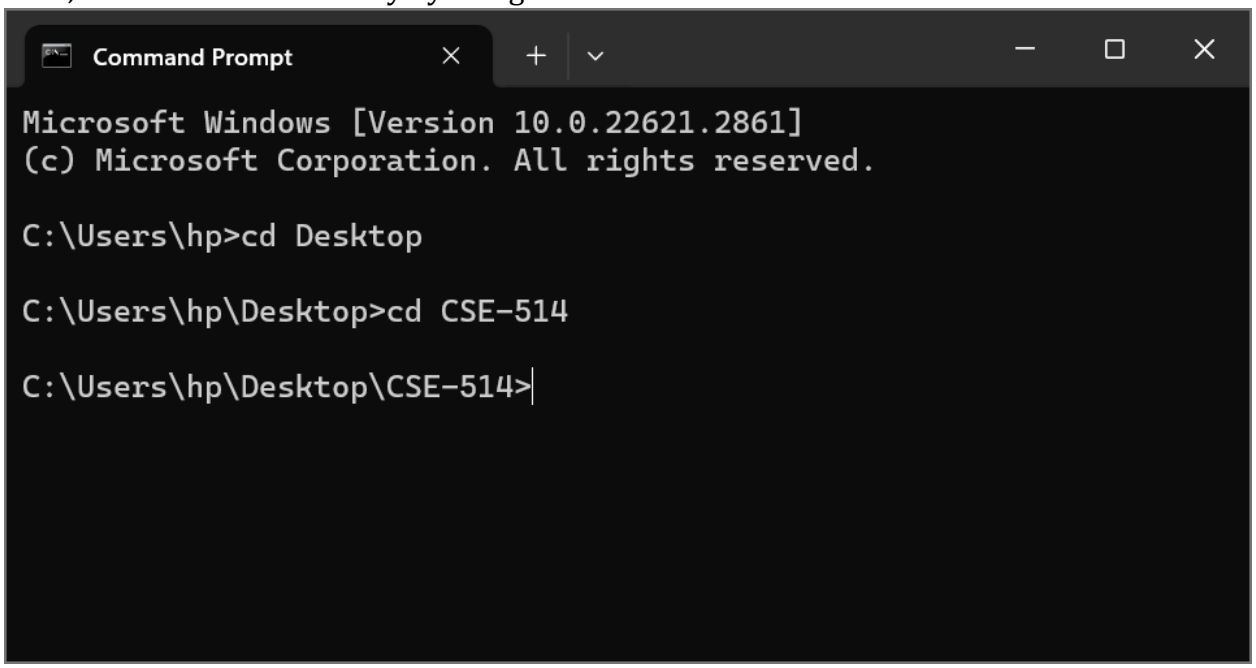
2. Once cmd prompt open go to DESKTOP using cd Desktop



3. Now create a Directory using mkdir or md command using your branch abbrevation and last 3 digit hall ticket number like md CSE-514.

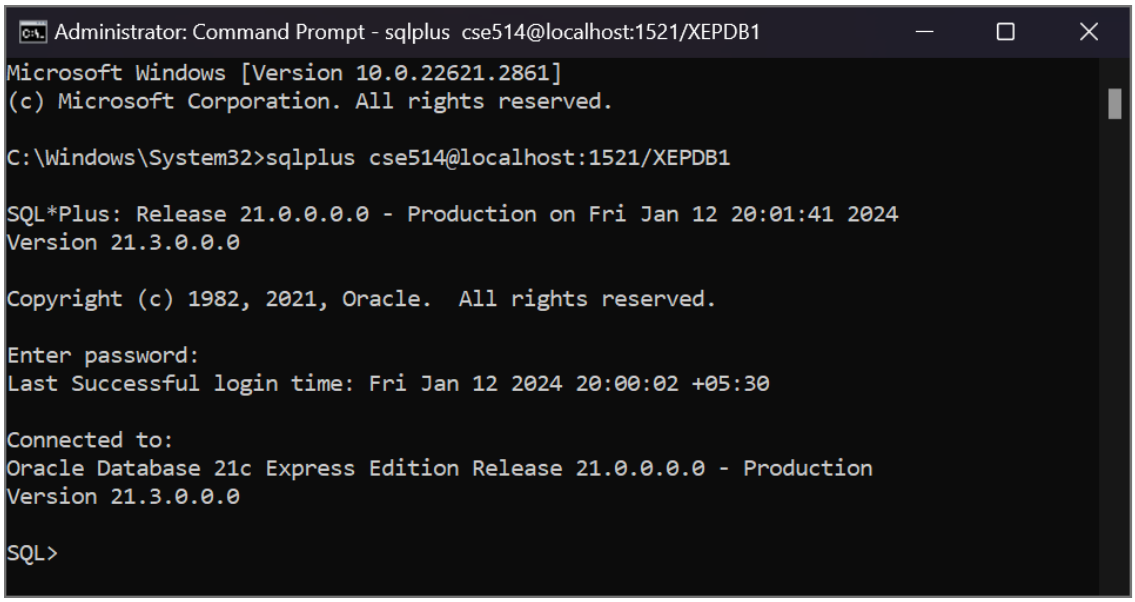


4. Now, move into the directory by using cd command show below.



5. To Login, , Type sqlplus command enter username and password when system is prompted.

PL/SQL program to perform factorial numbers



To write a PL/SQL program for calculating the factorial of a given number.

```
SQL> DECLARE
  2  FACT NUMBER:=1;
  3  N NUMBER;
  4  N1 NUMBER;
  5  BEGIN
  6  N:=&N;
  7  N1:=N;
  8  WHILE N>0 LOOP
  9  FACT:=N*FACT;
 10  N:=N-1;
 11  END LOOP;
 12  DBMS_OUTPUT.PUT_LINE('The Factorial of '||n1||' is '||FACT);
 13  END;
 14  /
Enter value for n: 5
old   6: N:=&N;
new   6: N:=5;
The Factorial of 5 is 120

PL/SQL procedure successfully completed.

SQL> SET VERIFY OFF
SQL> /
Enter value for n: 4
The Factorial of 4 is 24

PL/SQL procedure successfully completed.
```

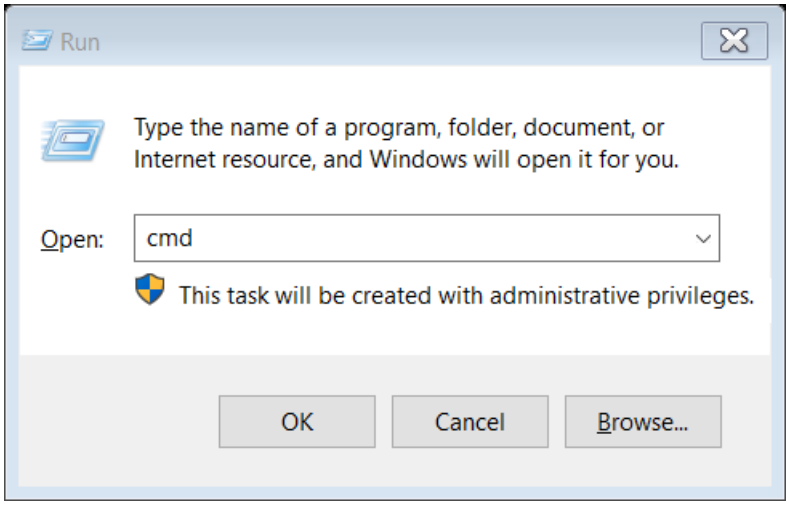
PL/SQL program to perform factorial numbers

- To run the program '/' is used.
- To display the output, we use "SET SERVEROUT ON".
- To eliminate debugging message "SET VERIFY OFF" should be used.

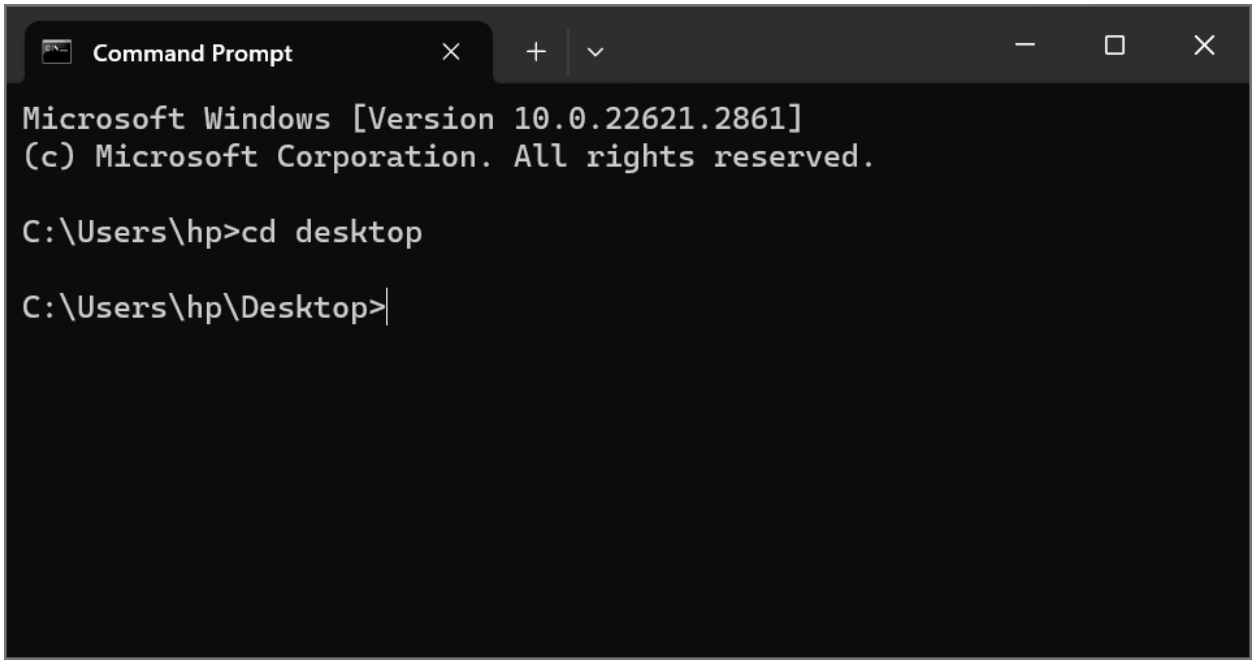
PL/SQL programs to find given number is prime or not

PL/SQL programs to find given number is prime or not

1. Open the command prompt Press WIN+R , type cmd

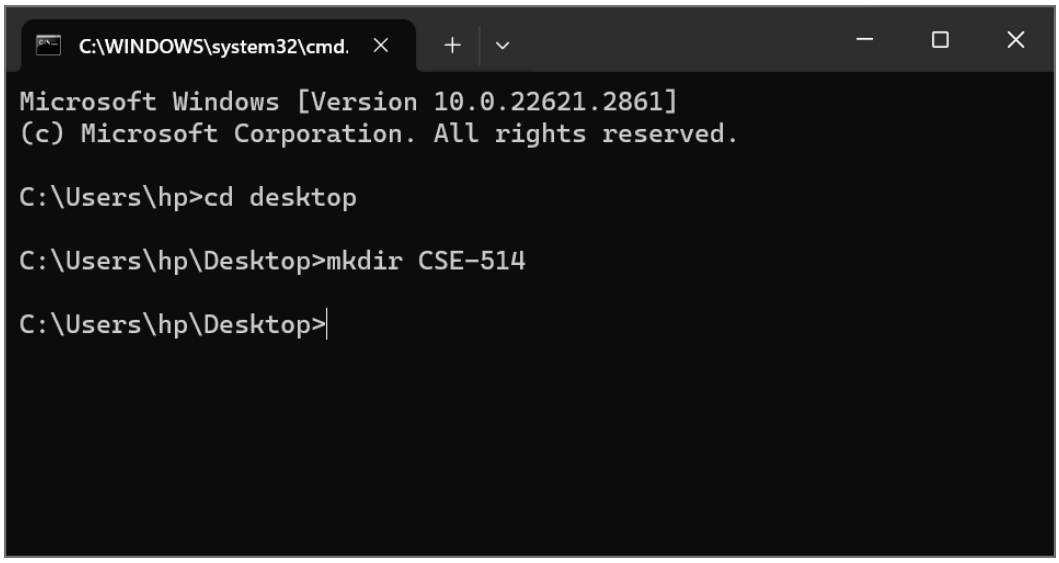


2. Once cmd prompt open go to DESKTOP using cd Desktop

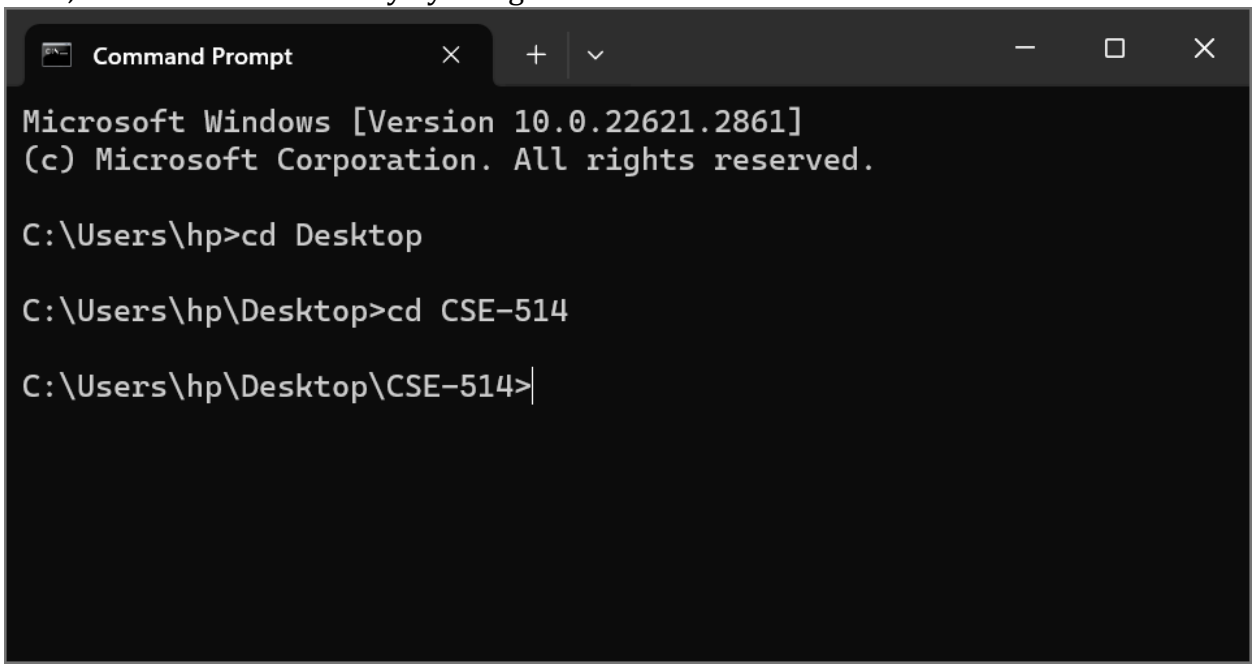


3. Now create a Directory using mkdir or md command using your branch abbrevation and last 3 digit hall ticket number like md CSE-514.

PL/SQL programs to find given number is prime or not

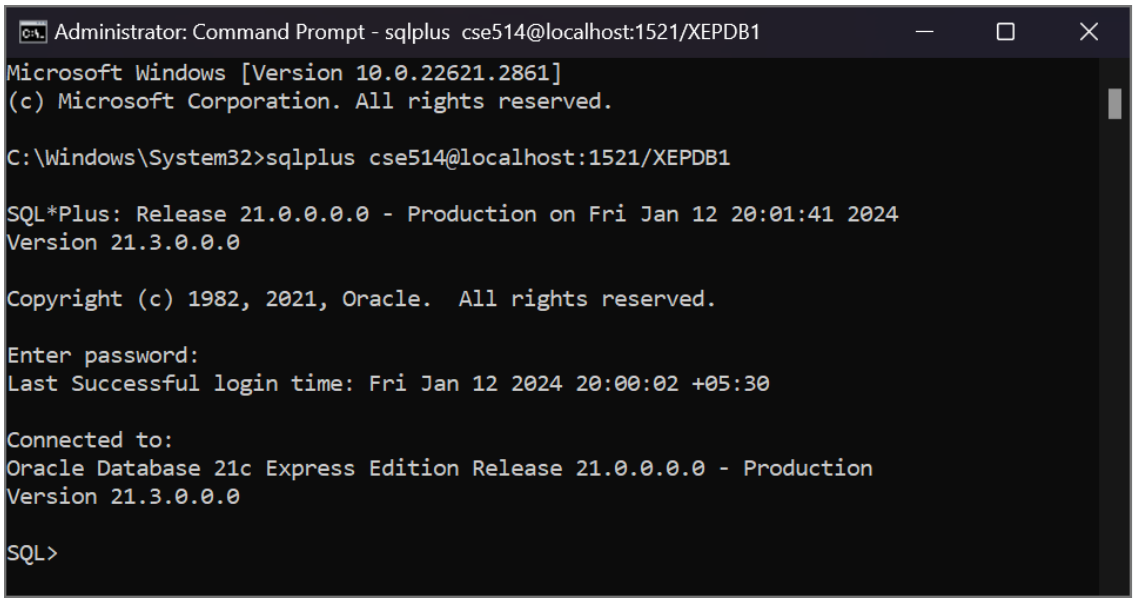


4. Now, move into the directory by using cd command show below.



5. To Login, , Type sqlplus command enter username and password when system is prompted.

PL/SQL programs to find given number is prime or not



To write a PL/SQL program for calculating the factorial of a given number.

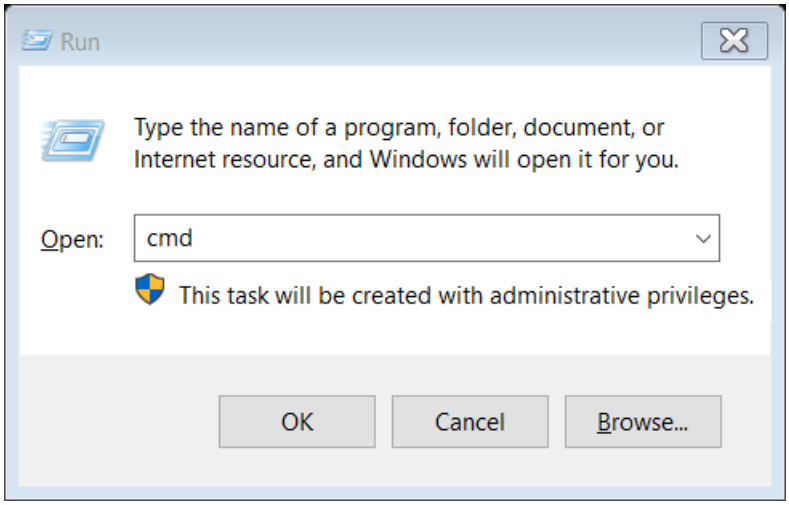
```
SQL> DECLARE
  2  N NUMBER;
  3  N1 NUMBER;
  4  I NUMBER;
  5  TEMP NUMBER;
  6  BEGIN
  7  N:=&N;
  8  N1:=N;
  9  I:=2;
 10  TEMP:=1;
 11  FOR I IN 2..N/2
 12  LOOP
 13  IF MOD(N,I)=0
 14  THEN
 15  TEMP:=0;
 16  EXIT;
 17  END IF;
 18  END LOOP;
 19  IF TEMP=1
 20  THEN
 21  DBMS_OUTPUT.PUT_LINE(N||' is a prime number');
 22  ELSE
 23  DBMS_OUTPUT.PUT_LINE(N||' is not a prime number');
 24  END IF;
 25  END;
 26  /
Enter value for n: 8
8 is not a prime number

PL/SQL procedure successfully completed.

SQL> /
Enter value for n: 11
11 is a prime number

PL/SQL procedure successfully completed.
```

PL/SQL programs to find given number is prime or not

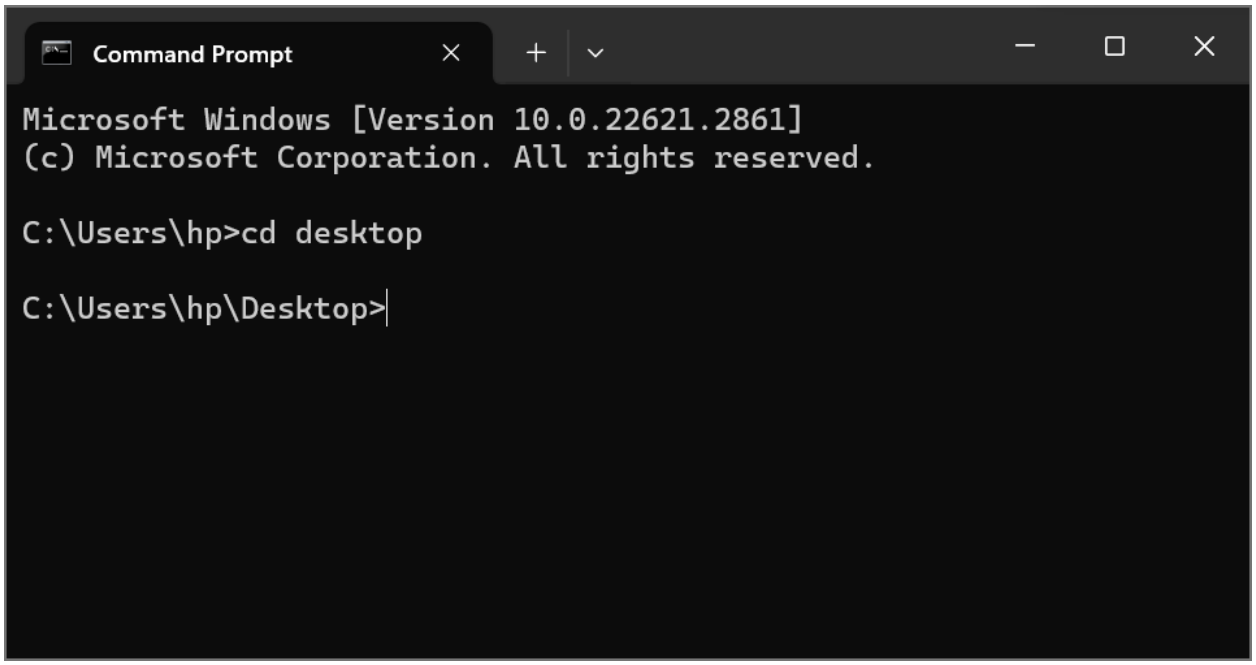
- To run the program '/' is used.
- To display the output, we use "SET SERVEROUT ON".
- To eliminate debugging message "SET VERIFY OFF" should be used.

PL/SQL program of Fibonacci series

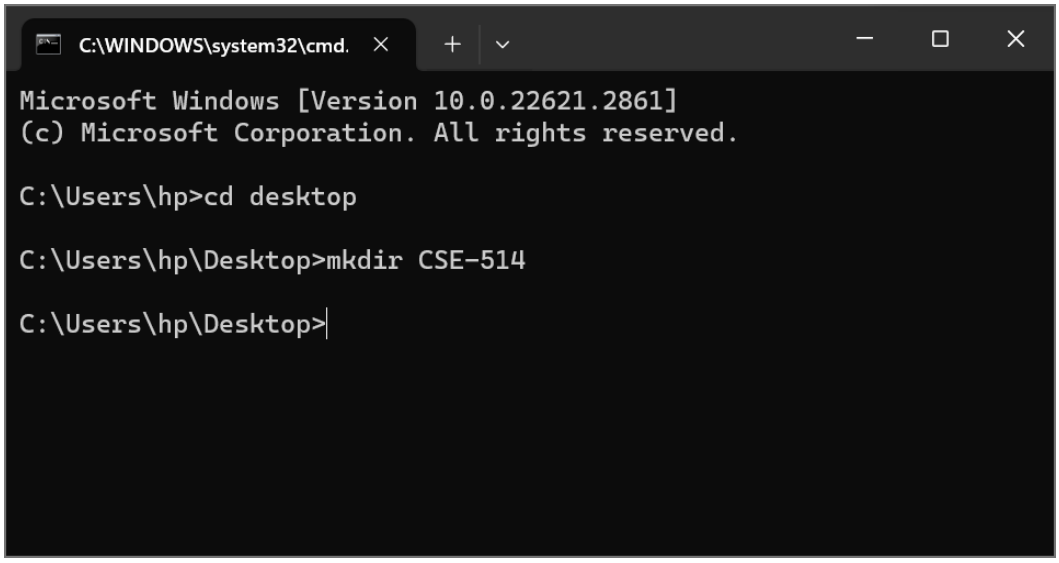
1. Open the command prompt Press WIN+R , type cmd



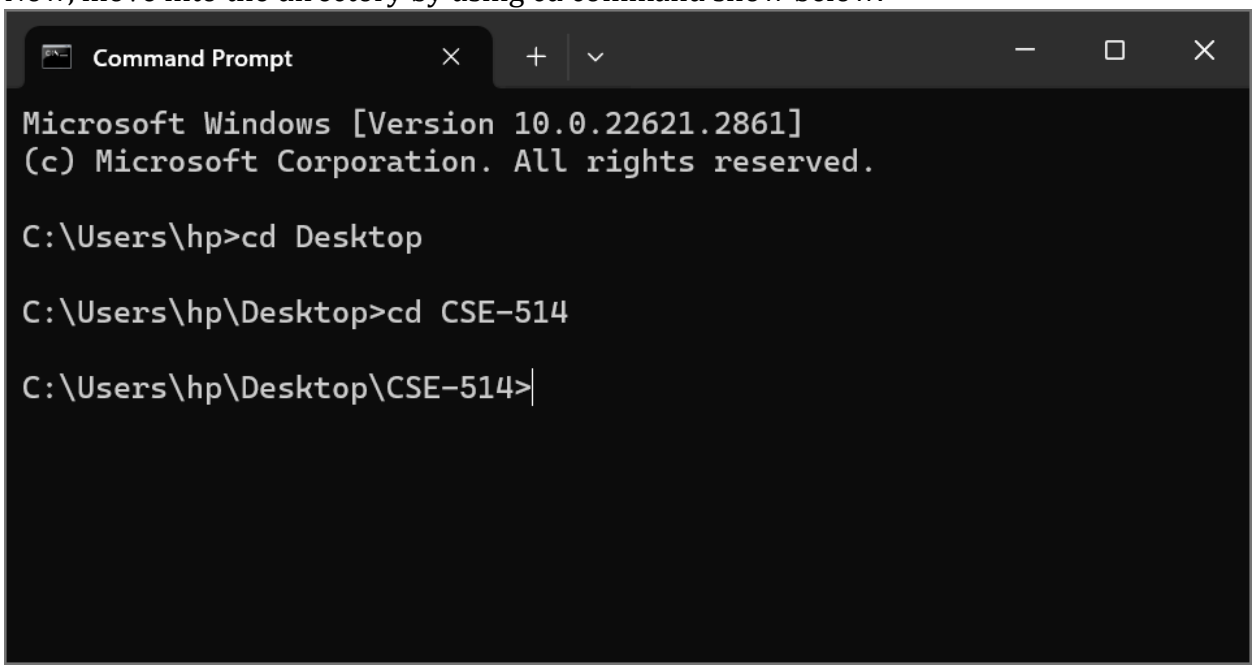
2. Once cmd prompt open go to DESKTOP using cd Desktop



3. Now create a Directory using mkdir or md command using your branch abbrevation and last 3 digit hall ticket number like md CSE-514.

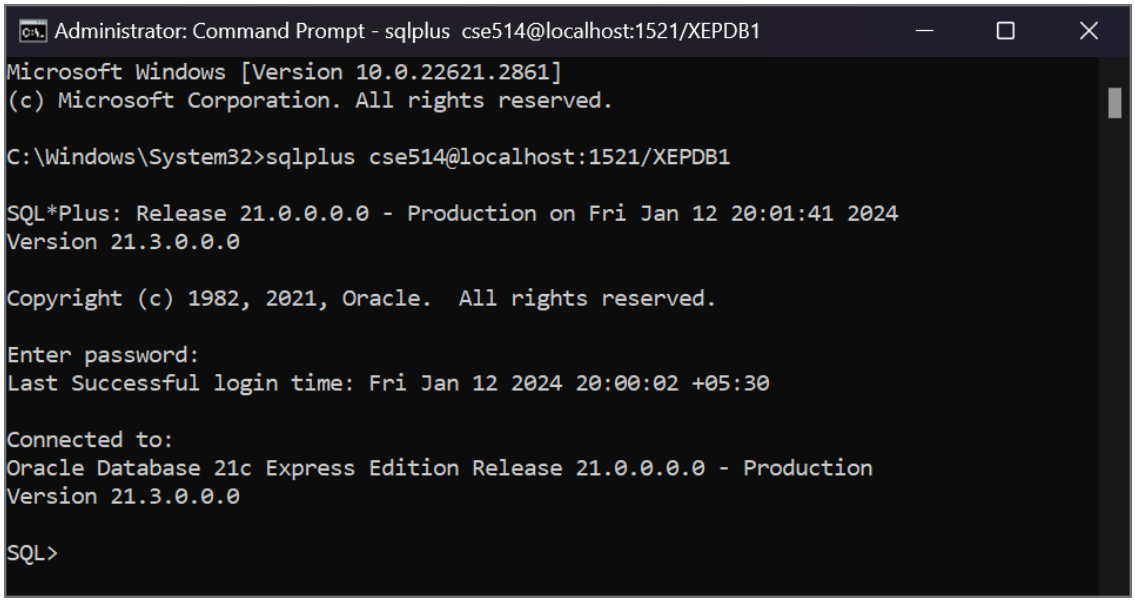


4. Now, move into the directory by using cd command show below.



5. To Login, , Type sqlplus command enter username and password when system is prompted.

PL/SQL program of Fibonacci series



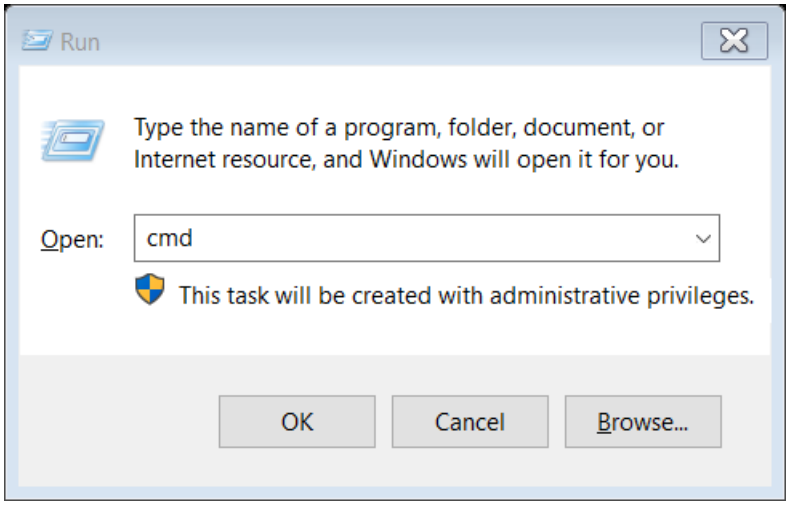
To write a PL/SQL program for displaying the Fibonacci series up to an integer.

```
SQL> DECLARE
  2  FIRST NUMBER:=0;
  3  SECOND NUMBER:=1;
  4  TEMP NUMBER;
  5  N NUMBER;
  6  N1 NUMBER;
  7  I NUMBER;
  8  BEGIN
  9  N:=&N;
 10  N1:=N;
 11  DBMS_OUTPUT.PUT_LINE('SERIES:');
 12  DBMS_OUTPUT.PUT_LINE(FIRST);
 13  DBMS_OUTPUT.PUT_LINE(SECOND);
 14  FOR I IN 2..N
 15  LOOP
 16  TEMP:=FIRST+SECOND;
 17  FIRST:=SECOND;
 18  SECOND:=TEMP;
 19  DBMS_OUTPUT.PUT_LINE(TEMP);
 20  END LOOP;
 21  END;
 22  /
Enter value for n: 6
SERIES:
0
1
1
2
3
5
8

PL/SQL procedure successfully completed.
```

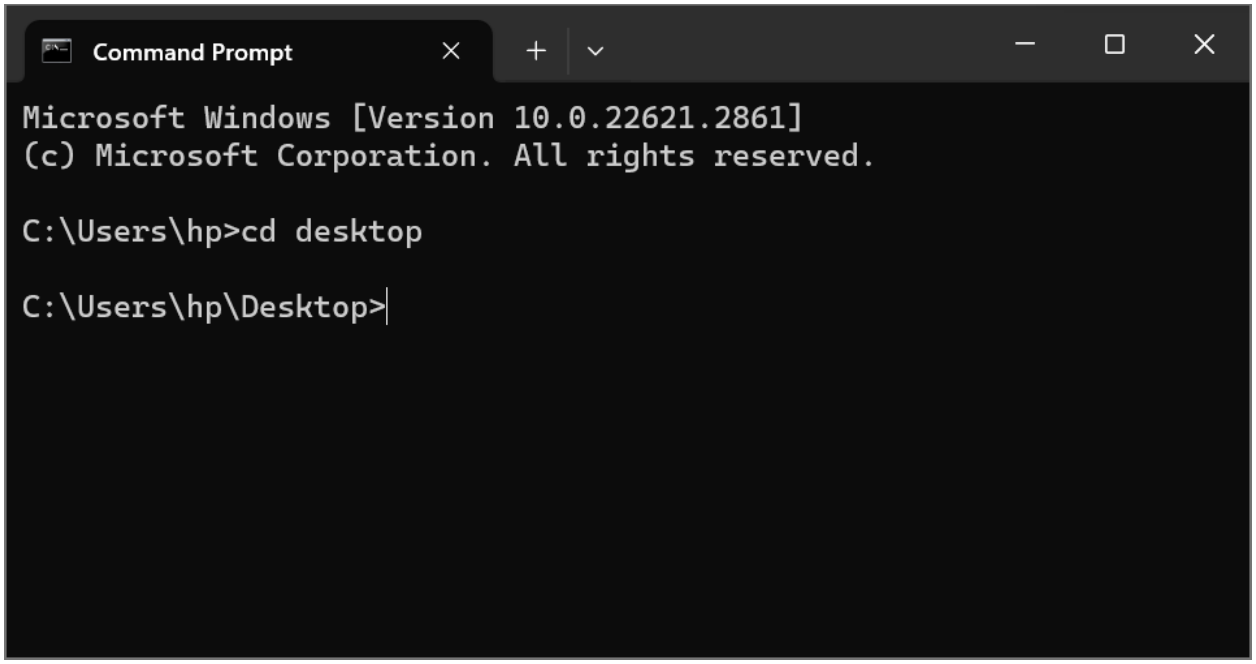
- To run the program '/' is used.
- To display the output, we use "SET SERVEROUT ON".
- To eliminate debugging message "SET VERIFY OFF" should be used.

PL/SQL to implement stored procedure on table

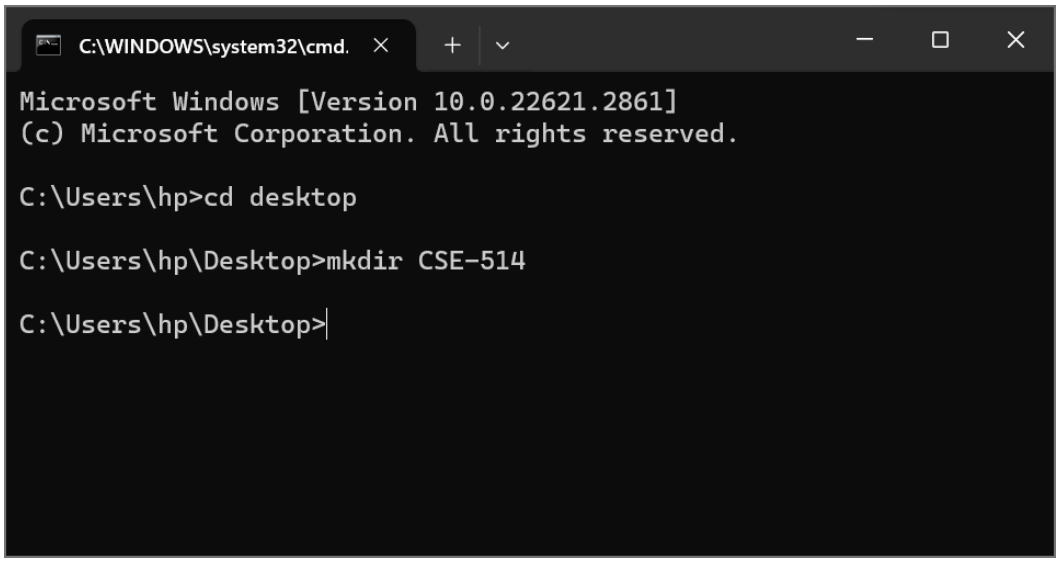
1. Open the command prompt Press WIN+R , type cmd



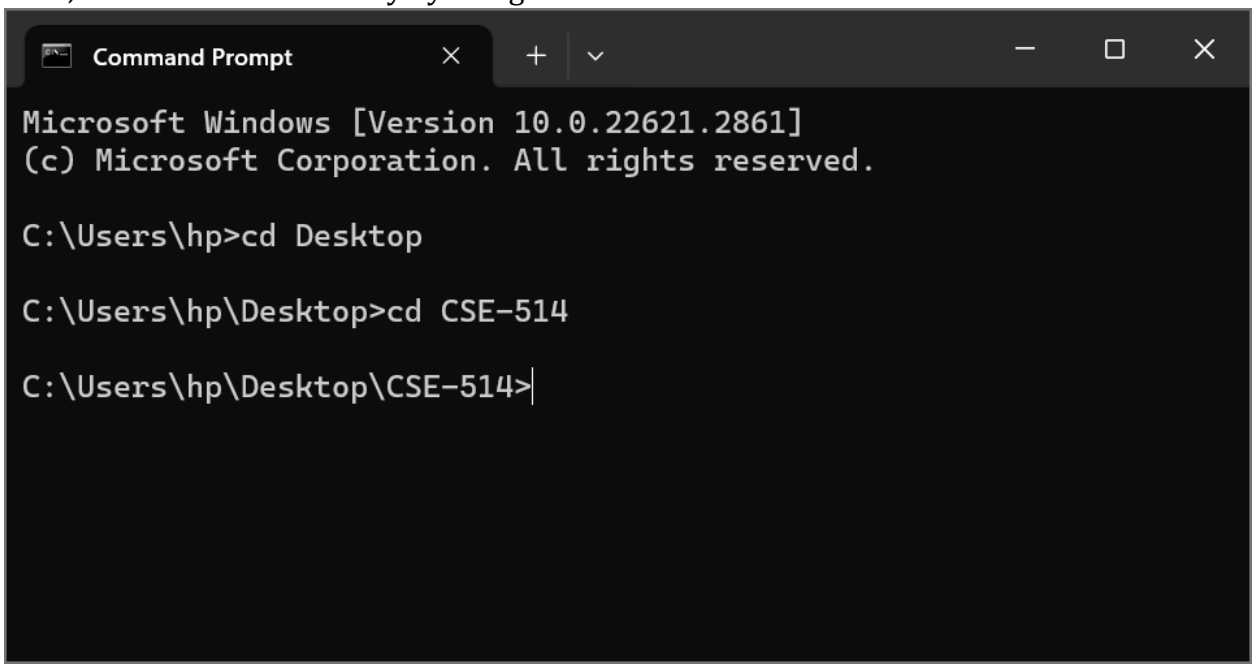
2. Once cmd prompt open go to DESKTOP using cd Desktop



3. Now create a Directory using mkdir or md command using your branch abbrevation and last 3 digit hall ticket number like md CSE-514.



4. Now, move into the directory by using cd command show below.



5. To Login, , Type sqlplus command enter username and password when system is prompted.

   To write a PL/SQL program to implement Stored Procedure on table.

   Example 1

PL/SQL to implement stored procedure on table

```
SQL> CREATE TABLE SAILOR(ID NUMBER(10) PRIMARY KEY,NAME VARCHAR2(100));

Table created.

SQL> CREATE OR REPLACE PROCEDURE INSERTUSER
  2  (ID IN NUMBER,
  3  NAME IN VARCHAR2)
  4  IS
  5  BEGIN
  6  INSERT INTO SAILOR VALUES(ID,NAME);
  7  DBMS_OUTPUT.PUT_LINE('RECORD INSERTED SUCCESSFULLY');
  8  END;
  9  /

Procedure created.
```

EXECUTION PROCEDURE:

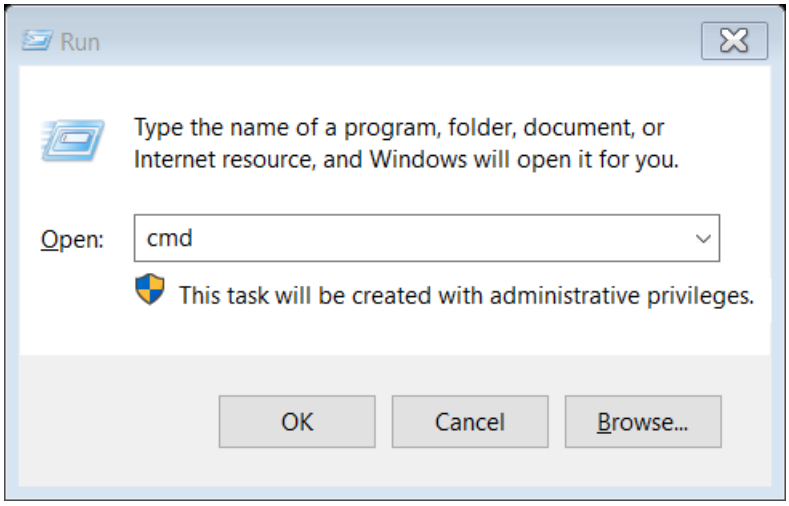
```
SQL> DECLARE
  2  CNT NUMBER;
  3  BEGIN
  4  INSERTUSER(202,'CHINNU');
  5  SELECT COUNT(*) INTO CNT FROM SAILOR;
  6  DBMS_OUTPUT.PUT_LINE(CNT||' RECORD IS INSERTED SUCCESSFULLY');
  7  END;
  8  /

PL/SQL procedure successfully completed.
```

DROP PROCEDURE:

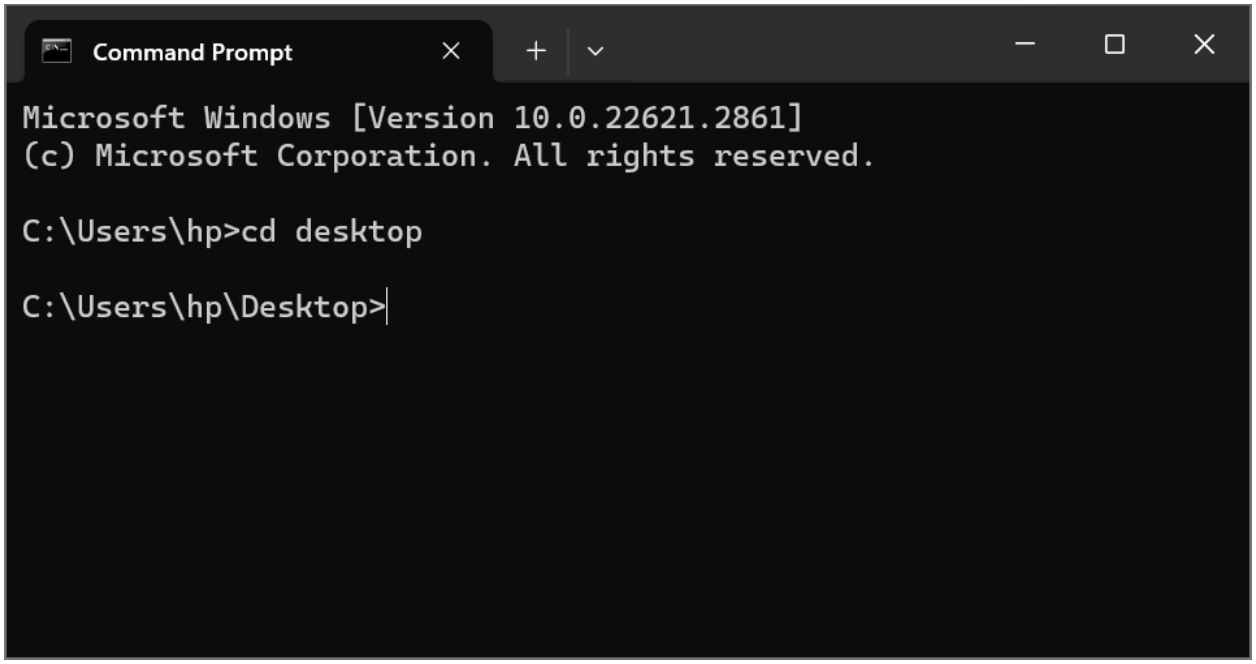
```
SQL> DROP PROCEDURE insertuser;

Procedure dropped.
```

PL/SQL program to implement stored function on table

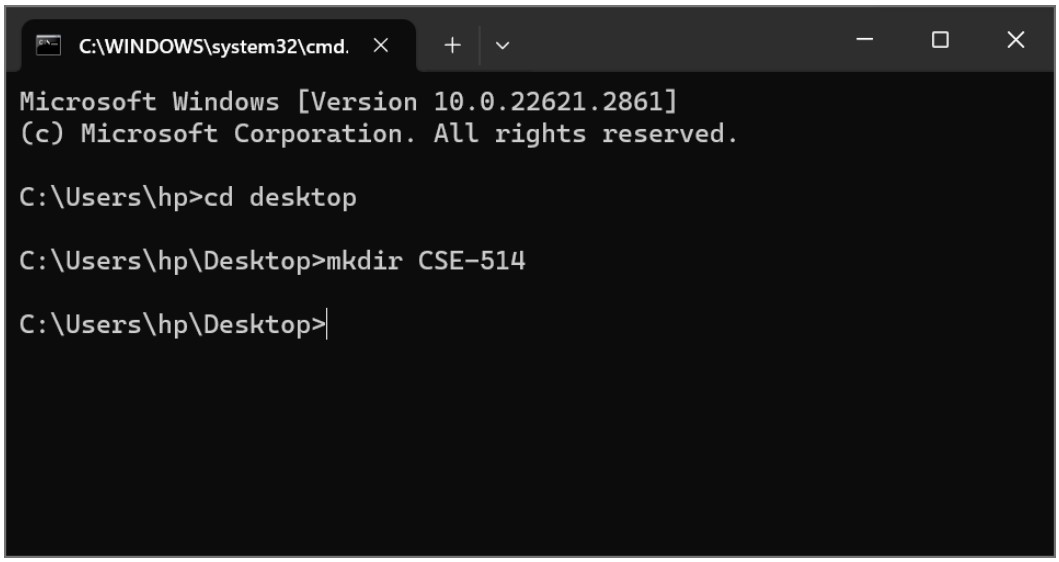
1. Open the command prompt Press WIN+R , type cmd



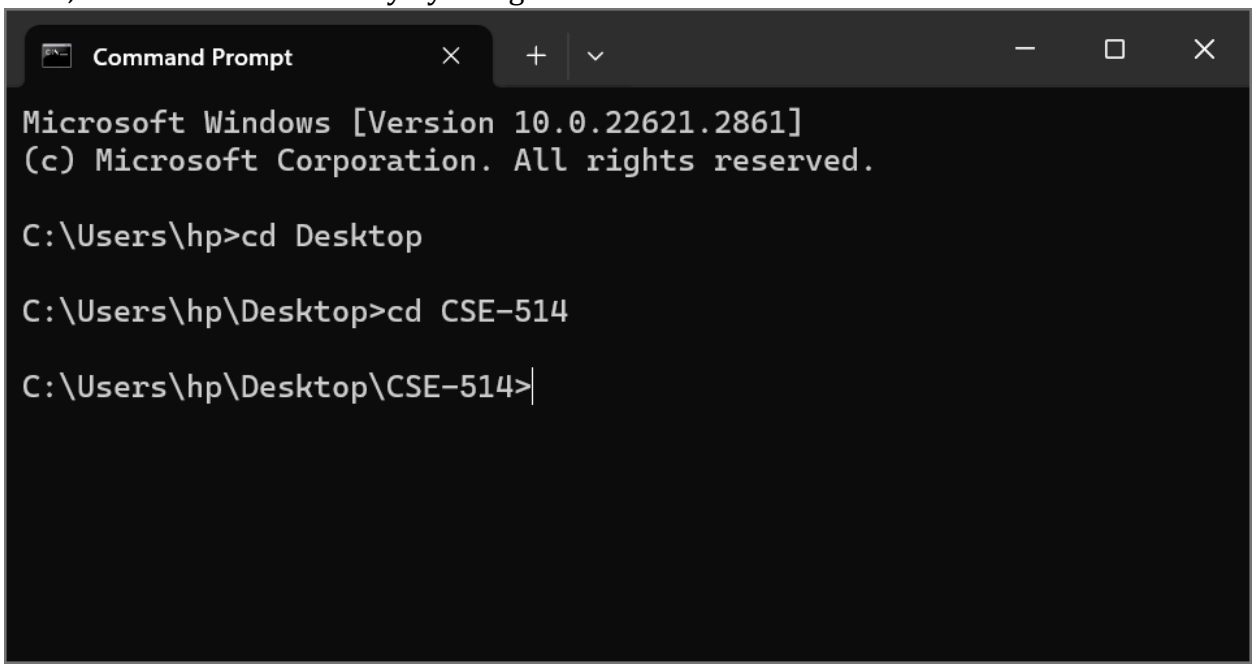
2. Once cmd prompt open go to DESKTOP using cd Desktop



3. Now create a Directory using mkdir or md command using your branch abbrevation and last 3 digit hall ticket number like md CSE-514.



4. Now, move into the directory by using cd command show below.



5. To Login, , Type sqlplus command enter username and password when system is prompted.

   To write a PL/SQL program to implement Stored Function on table.
   EXAMPLE-1:

PL/SQL program to implement stored function on table

```
SQL> CREATE OR REPLACE FUNCTION ADDER(N1 IN NUMBER, N2 IN NUMBER)
  2  RETURN NUMBER
  3  IS
  4  N3 NUMBER(8);
  5  BEGIN
  6  N3:=N1+N2;
  7  RETURN N3;
  8  END;
  9  /

Function created.
```

EXECUTION PROCEDURE:

```
SQL> DECLARE
  2  N3 NUMBER(2);
  3  BEGIN
  4  N3:=ADDER(22,44);
  5  DBMS_OUTPUT.PUT_LINE('ADDITION IS: '||N3);
  6  END;
  7  /

PL/SQL procedure successfully completed.

SQL> SET SERVEROUT ON
SQL> /
ADDITION IS: 66

PL/SQL procedure successfully completed.
```

```
SQL> DROP FUNCTION ADDER;

Function dropped.

SQL>
```

EXAMPLE-2

```
SQL> CREATE FUNCTION FACT(X NUMBER)
  2  RETURN NUMBER
  3  IS
  4  F NUMBER;
  5  BEGIN
  6  IF X=0 THEN
  7  F:=1;
  8  ELSE
  9  F:=X*FACT(X-1);
 10  END IF;
 11  RETURN F;
 12  END;
 13  /

Function created.
```

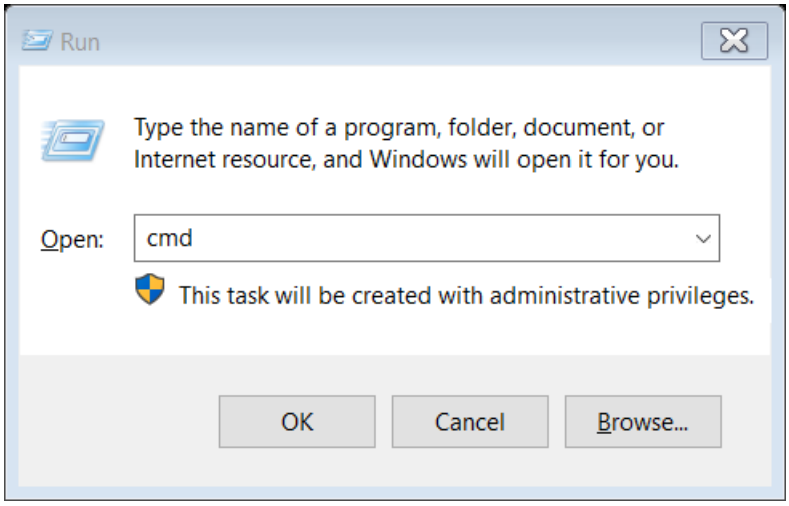
EXECUTION PROCEDURE:

PL/SQL program to implement stored function on table

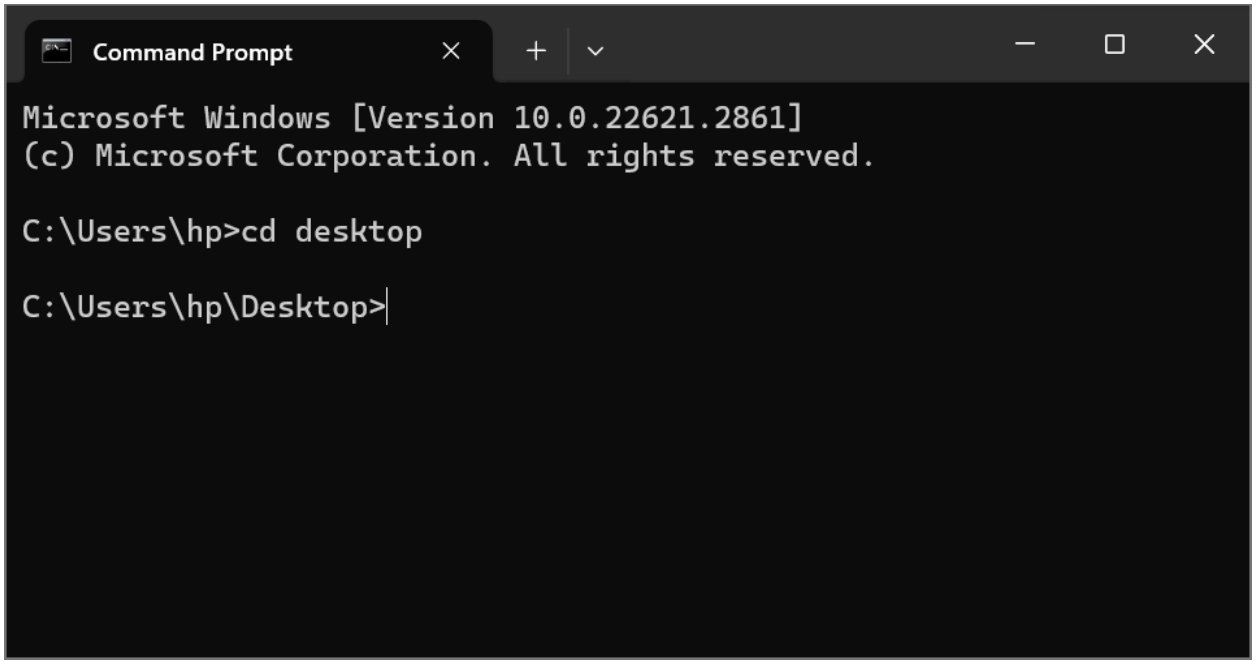
```
SQL> DECLARE
  2  NUM NUMBER;
  3  FACTORIAL NUMBER;
  4  BEGIN
  5  NUM:=4;
  6  FACTORIAL:=FACT(NUM);
  7  DBMS_OUTPUT.PUT_LINE(' FACTORIAL '||NUM||' IS '|| FACTORIAL);
  8  END;
  9  /
FACTORIAL 4 IS 24

PL/SQL procedure successfully completed.
```

```
SQL> DROP FUNCTION FACT;

Function dropped.
```

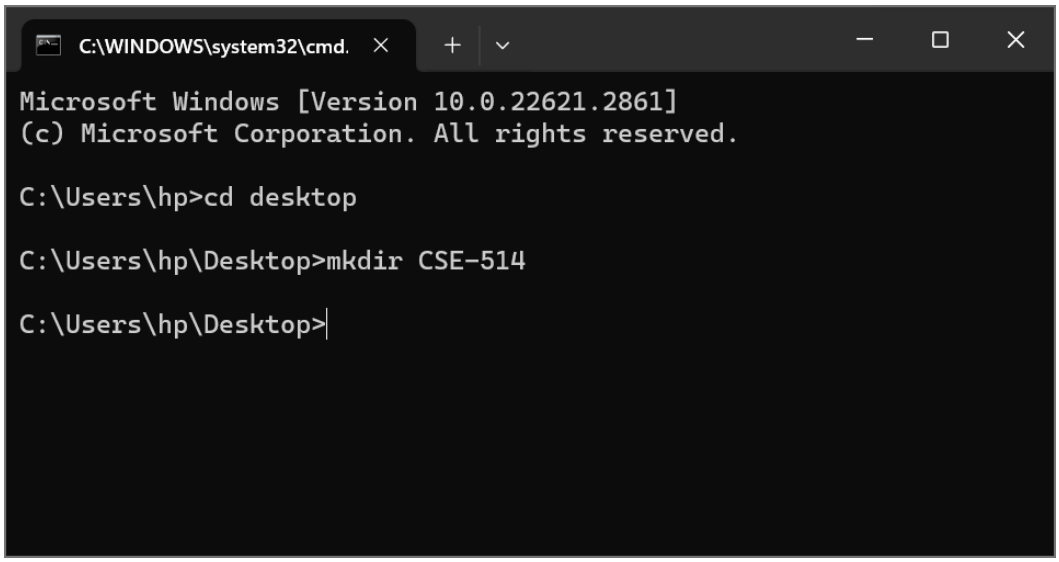
1. Open the command prompt Press WIN+R , type cmd



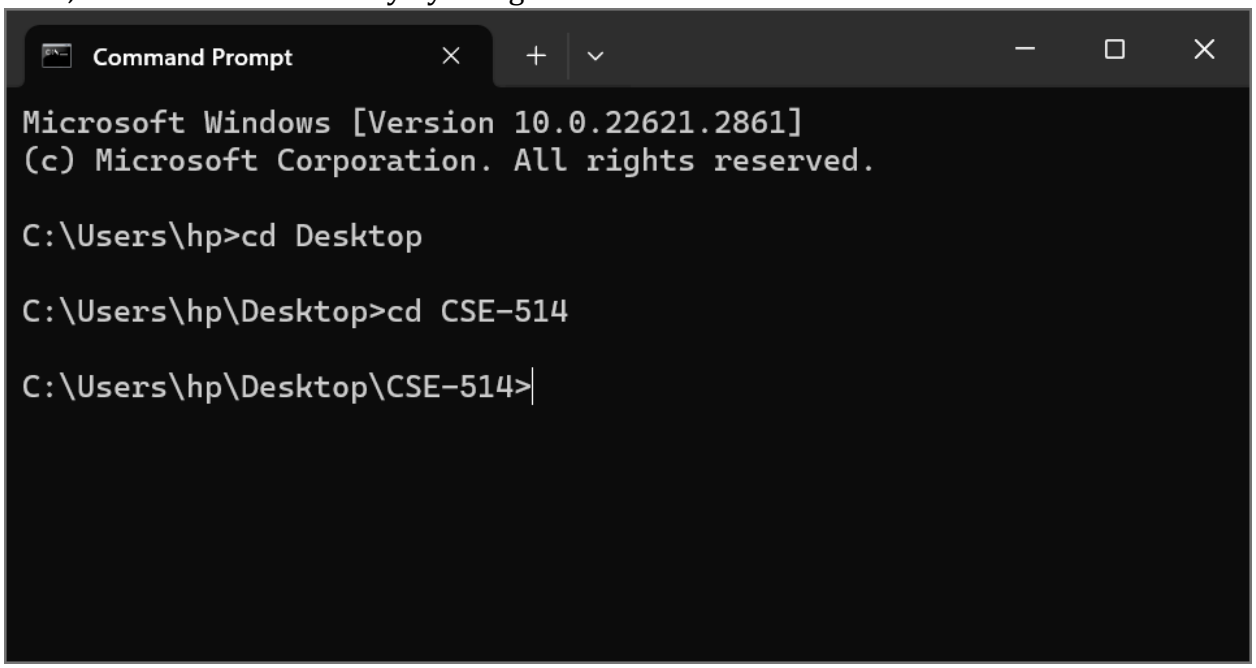
2. Once cmd prompt open go to DESKTOP using cd Desktop



3. Now create a Directory using mkdir or md command using your branch abbrevation and last 3 digit hall ticket number like md CSE-514.



4. Now, move into the directory by using cd command show below.



5. To Login, , Type sqlplus command enter username and password when system is prompted.

   To write PL/SQL program to implement Trigger on table.

   CREATING TABLE

PL/SQL to implement trigger on table

```
SQL> CREATE TABLE INSTRUCTOR(
  2  ID NUMBER PRIMARY KEY,
  3  NAME VARCHAR2(50) NOT NULL,
  4  DEPT_NAME VARCHAR2(20) NOT NULL,
  5  SALARY NUMBER(10,2) CHECK (SALARY>30000)
  6  );

Table created.
```

```
SQL> INSERT INTO INSTRUCTOR VALUES(30,'AMMU','CSE',50000);

1 row created.

SQL> INSERT INTO INSTRUCTOR VALUES(40,'ANI','CSM',57000);

1 row created.

SQL> INSERT INTO INSTRUCTOR VALUES(50,'ARUSH','CSD',40000);

1 row created.
```

```
SQL> SELECT * FROM INSTRUCTOR;

        ID NAME
---------- ----------------------------
DEPT_NAME               SALARY
-------------------- ----------
        30 AMMU
CSE                      50000

        40 ANI
CSM                      57000

        50 ARUSH
CSD                      40000
```

AN EXAMPLE TO CREATE TRIGGER:

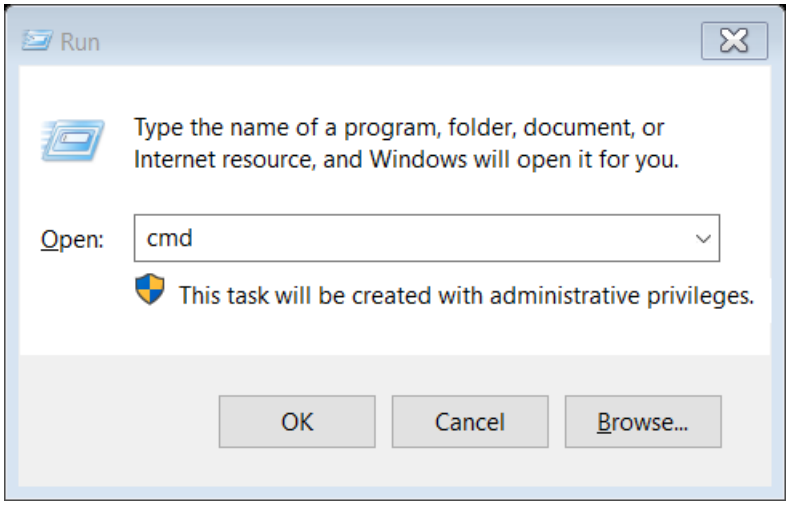
```
SQL> CREATE OR REPLACE TRIGGER display_changes
  2  BEFORE UPDATE ON instructor
  3  FOR EACH ROW
  4  WHEN (NEW.ID=OLD.ID)
  5  DECLARE
  6  sal_diff NUMBER;
  7  BEGIN
  8  sal_diff:=:NEW.SALARY- :OLD.SALARY;
  9  DBMS_OUTPUT.PUT_LINE('OLD SALARY: '|| :OLD.SALARY);
 10  DBMS_OUTPUT.PUT_LINE('NEW SALARY: '|| :NEW.SALARY);
 11  DBMS_OUTPUT.PUT_LINE('SALARY DIFFERENCE: '|| sal_diff);
 12  END;
 13  /

Trigger created.
```

A PL/SQL Procedure to execute a trigger:

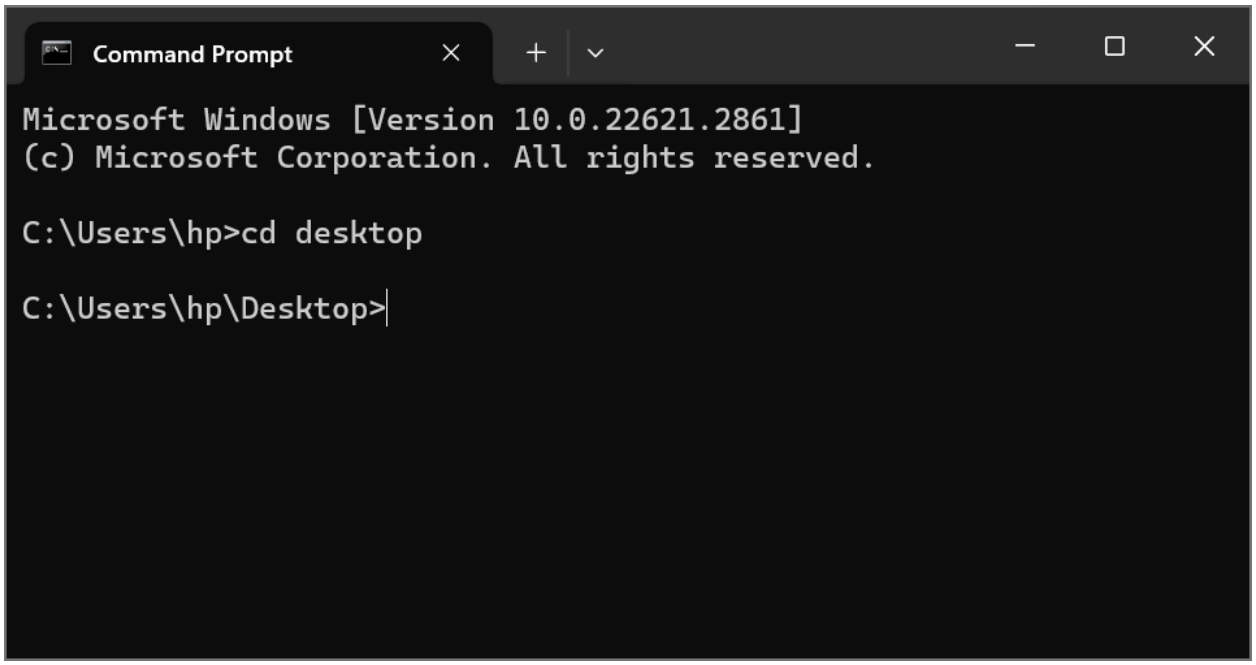
```
SQL> DECLARE
  2  tot_rows NUMBER;
  3  BEGIN
  4  UPDATE instructor
  5  SET SALARY=SALARY*1.5;
  6  IF sql%notfound THEN
  7  DBMS_OUTPUT.PUT_LINE(' NO INSTRUCTORS UPDATED');
  8  ELSIF sql%found THEN
  9  tot_rows:=sql%rowcount;
 10  DBMS_OUTPUT.PUT_LINE(tot_rows||' INSTRUCTORS UPDATED');
 11  END IF;
 12  END;
 13  /
OLD SALARY: 55000
NEW SALARY: 82500
SALARY DIFFERENCE: 27500
OLD SALARY: 50000
NEW SALARY: 75000
SALARY DIFFERENCE: 25000
OLD SALARY: 60000
NEW SALARY: 90000
SALARY DIFFERENCE: 30000
3 INSTRUCTORS UPDATED

PL/SQL procedure successfully completed.
```

PL/SQL programs to implement cursor on table

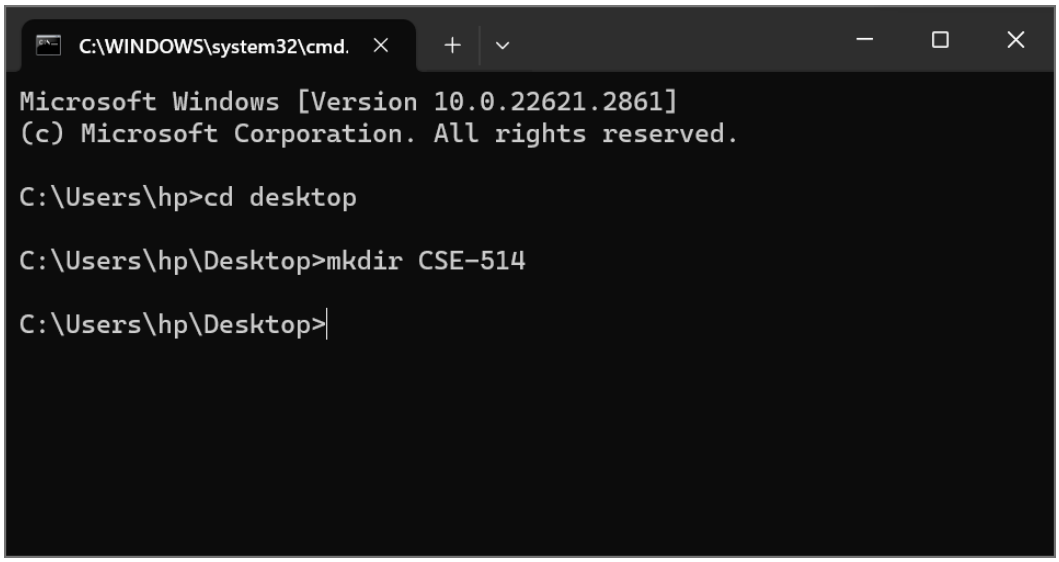
1. Open the command prompt Press WIN+R , type cmd



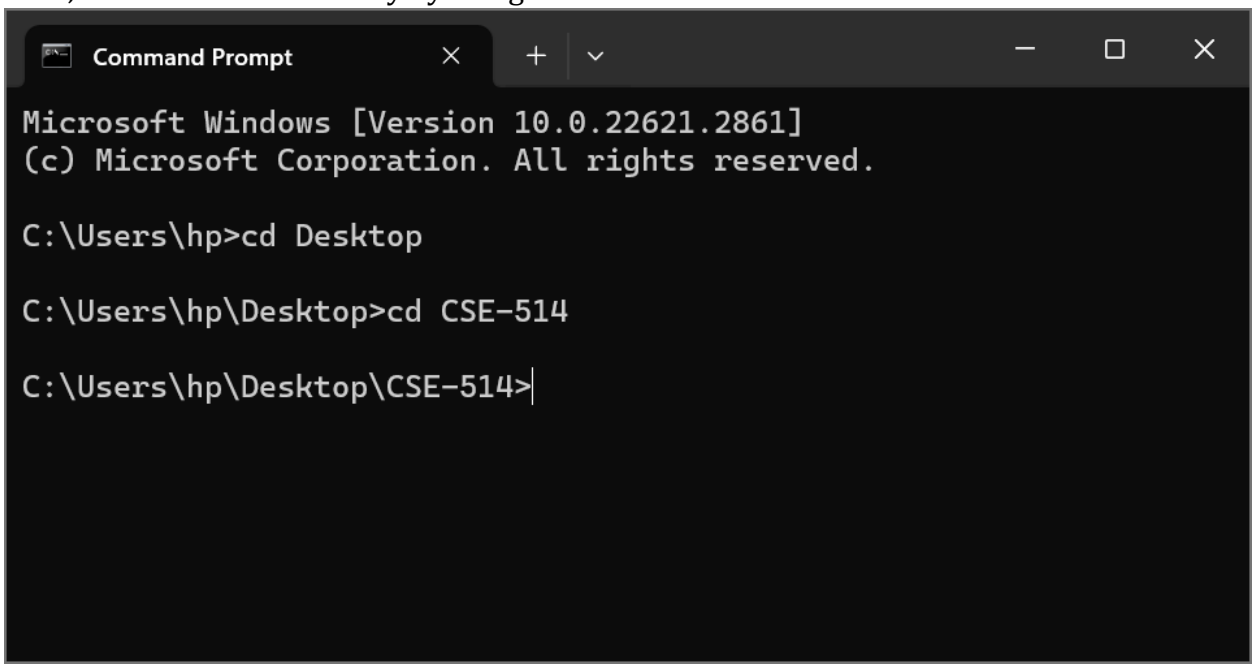
2. Once cmd prompt open go to DESKTOP using cd Desktop



3. Now create a Directory using mkdir or md command using your branch abbrevation and last 3 digit hall ticket number like md CSE-514.



4. Now, move into the directory by using cd command show below.



5. To Login, , Type sqlplus command enter username and password when system is prompted.
   To write a PL/SQL program to implement Cursor on table.
   CREATING A TABLE:

PL/SQL programs to implement cursor on table

```
SQL> CREATE TABLE customers(
  2  ID NUMBER PRIMARY KEY,
  3  NAME VARCHAR2(20) NOT NULL,
  4  AGE NUMBER,
  5  ADDRESS VARCHAR2(20),
  6  SALARY NUMERIC(20,2)
  7  );

Table created.
```

INSERTING VALUES INTO TABLE:

```
SQL> INSERT INTO customers VALUES(501, 'Siri','19','Delhi', '270000');

1 row created.

SQL> INSERT INTO customers VALUES(502, 'Smith','21','Agra', '290000');

1 row created.

SQL> INSERT INTO customers VALUES(503, 'Suresh','23','Noida', '320000');

1 row created.
```

```
SQL> SELECT * FROM customers;

        ID NAME                           AGE ADDRESS
   SALARY
---------- -------------------- ---------- --------------------
       501 Siri                          19 Delhi
   270000
       502 Smith                         21 Agra
   290000
       503 Suresh                        23 Noida
   320000
```

PL/SQL Program using Explicit Cursors:

PL/SQL programs to implement cursor on table

```
SQL> DECLARE
  2  c_id customers.id%type;
  3  c_name customers.name%type;
  4  c_addr customers.address%type;
  5  CURSOR c_customers IS
  6  SELECT id,name,address FROM customers;
  7  BEGIN
  8  OPEN c_customers;
  9  LOOP
 10  FETCH c_customers INTO c_id,c_name,c_addr;
 11  EXIT WHEN c_customers%notfound;
 12  DBMS_OUTPUT.PUT_LINE(c_id||' '||c_name||' '||c_addr);
 13  END LOOP;
 14  CLOSE c_customers;
 15  END;
 16  /

PL/SQL procedure successfully completed.

SQL> SET SERVEROUT ON
SQL> /
501 Siri Delhi
502 Smith Agra
503 Suresh Noida

PL/SQL procedure successfully completed.
```