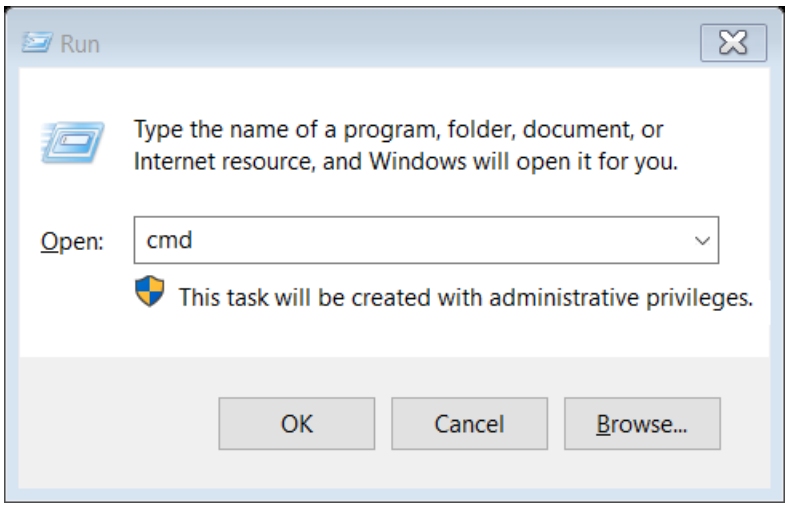
1. Open the command prompt Press WIN+R , type cmd



2. Create user with your id number and grant all privileges.

```
SQL> CREATE USER CSE_5B1 IDENTIFIED BY SUPRAJA;

User created.

SQL> GRANT ALL PRIVILEGES TO CSE_5B1;

Grant succeeded.
```

3. Now sign in with the new user.

s

```
SQL*Plus: Release 11.2.0.2.0 Production on Mon Jan 8 09:27:08 2024

Copyright (c) 1982, 2014, Oracle.  All rights reserved.

Enter user-name: CSE_5B1
Enter password:

Connected to:
Oracle Database 11g Express Edition Release 11.2.0.2.0 - 64bit Production
```

1. DDL COMMANDS

Write SQL queries to CREATE TABLES for various databases using DDL commands (i.e. CREATE, ALTER, DROP, TRUNCATE).

CREATE TABLE

Syntax:

```
CREATE TABLE tablename (
column1 data_ type [constraint]
[, column2 data_ type [constraint] ] [,
PRIMARY KEY (column1 [, column2]) ]
[, FOREIGN KEY (column1 [, column2]) REFERENCES tablename] [,CONSTRAINT constraint]);
```

Example:

```
224G1A05B1>CREATE TABLE Orders(
  2  OrderID int NOT NULL,
  3  OrderNumber int NOT NULL,
  4  PersonID int,
  5  PRIMARY KEY(OrderID)
  6  );

Table created.
```

ALTER TABLE

Syntax 1:
ALTER TABLE tablename
{ADD | MODIFY} (column_name data_type [{ADD|MODIFY}
Column_name data_type]);

Syntax 2;
ALTER TABLE tablename
ADD constraint [ADD constraint];

Syntax 3:
ALTER TABLE tablename
DROP {PRIMARY KEY | COLUMN column_name | CONSTRAINT constraint_name);

Syntax 4:
ALTER TABLE tablename
ENABLE CONSTRAINT constraint_name;

Example:

```
224G1A05B1>ALTER TABLE Orders
  2  ADD (Mail varchar(32));

Table altered.
```

DESC Orders:

```
224G1A05B1>DESC Orders
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 ORDERID                                   NOT NULL NUMBER(38)
 ORDERNUMBER                               NOT NULL NUMBER(38)
 PERSONID                                           NUMBER(38)
 MAIL                                               VARCHAR2(32)
```

DROP TABLE

Syntax:

DROP TABLE table_name;

```
224G1A05B1>DROP table Orders;

Table dropped.
```

Example:

TRUNCATE TABLE

Syntax:

TRUNCATE TABLE table_name;

Example:

```
224G1A05B1>TRUNCATE TABLE Orders;

Table truncated.
```

2.DML COMMANDS

Write SQL queries to MANIPULATE TABLES for various databases using DML commands (i.e. INSERT, SELECT, UPDATE, DELETE,)

```
224G1A05B1>CREATE TABLE Student(
  2  Roll_no INT NOT NULL PRIMARY KEY,
  3  Name VARCHAR(50) NOT NULL,
  4  Age INT NOT NULL,
  5  Address VARCHAR(255),
  6  Date_Of_Birth DATE
  7  );

Table created.
```

INSERT
Syntax:
INSERT INTO tablename
VALUES (value1,value2,...,valuen);

Syntax 2:
INSERT INTO tablename
(column1, column2,…,column) VALUES (value1, value2,...,valuen);

Example:

```
224G1A05B1>INSERT INTO Student (Roll_no,Name,Age)
  2  VALUES (3,'suppu',20);

1 row created.
```

```
224G1A05B1>INSERT INTO Student (Roll_no,Name,Age)
  2  VALUES (2,'lalli',25);

1 row created.
```

SELECT
Syntax:

```
SELECT *
DROM <table_name>;
```

Example:

```
224G1A05B1>SELECT * FROM Student;

   ROLL_NO NAME                                               AGE
---------- -------------------------------------------------- ----------
ADDRESS
--------------------------------------------------------------------------------
DATE_OF_B
---------
         3 suppu                                               20



         2 lalli                                               25



   ROLL_NO NAME                                               AGE
---------- -------------------------------------------------- ----------
ADDRESS
--------------------------------------------------------------------------------
DATE_OF_B
---------
```

UPDATE
Syntax:
UPDATE table_name SET [column_name1= value_1, column_name2= value_2,...]
WHERE CONDITION;

```
224G1A05B1>UPDATE Student
  2  SET Age= Age +1;

2 rows updated.
```

DELETE
Syntax:
DELETE FROM table_Name WHERE condition;
Example:

```
224G1A05B1>DELETE FROM Student
  2  WHERE Age<21;

0 rows deleted.
```

3.VIEWS

Write SQL queries to create VIEWS for various databases (i.e. CREATE VIEW, UPDATE VIEW, ALTER VIEW, and DELETE VIEW).

View syntax:

CREATE VIEW VIEW_NAME AS <QUERY EXPRESSION>

```
224G1A05B1>CREATE VIEW FACULTY AS
  2  SELECT ID,NAME,DEPT_NAME
  3  FROM INSTRUCTOR;

View created.
```

An equivalent relation of view without using view as original relation

```
224G1A05B1>CREATE VIEW PHYSICS_FALL_WATSON AS
  2  (
  3  SELECT COURSE_ID,BUIDIND
  4  FROM (SELECT COURSE_ID,BUIDIND
  5  FROM studentsCOURSES, SECTIONS
  6  WHERE studentsCOURSES.DEPT_NAME='csm'
  7  AND SECTIONS.SEMESTER = 'FALL')
  8  );

View created.
```

Commands to insert, Delete and update view

```
224G1A05B1>CREATE VIEW HISTORY_instructors As
  2  SELECT * FROM instructors
  3  WHERE DEPT_NAME='HISTORY';

View created.
```

```
224G1A05B1>SELECT * FROM instructors;

ID    NAME                 DEPT_NAME              SALARY
----- -------------------- -------------------- ----------
101   abhishek kumar       compsci                 65000
102   supraja              finance                 95000
103   kalyani              history                 68000
104   prabhas              finance                 70000
105   anushka              finance                 43000
106   sweety               history                 48000
107   santhosh             history                 90000

7 rows selected.

224G1A05B1>
```

DELETE VIEW:

```
224G1A05B1>DELETE VIEW HISTORY_instructors;

View deleted
```

DROP VIEW

```
224G1A05B1>DROP VIEW HISTORY_instructors;

View dropped.
```

UPDATE VIEW

```
224G1AO5b1>UPDATE VIEW HISTORY_instructors;

View updated
```

4.RELATIONAL SET OPERATIONS

Write SQL queries to perform RELATIONAL SET OPERATIONS (i.e. UNION, UNION ALL, INTERSECT, MINUS, CROSS JOIN, NATURAL JOIN)

```
224G1A05B1>CREATE TABLE rooms
  2  (BUILDING VARCHAR2(15),
  3  ROOM_NUMBER VARCHAR2(7),
  4  CAPACITY NUMERIC(4,0),
  5  PRIMARY KEY (BUILDING, ROOM_NUMBER)
  6  );

Table created.
```

```
224G1A05B1>INSERT INTO rooms VALUES ('Packard', '101', '500');

1 row created.

224G1A05B1>INSERT INTO rooms VALUES ('Painter', '514', '10');

1 row created.

224G1A05B1>INSERT INTO rooms VALUES ('Taylor', '3128', '70');

1 row created.

224G1A05B1>INSERT INTO rooms VALUES ('Watson', '100', '30');

1 row created.

224G1A05B1>INSERT INTO rooms VALUES ('Watson', '120', '50');

1 row created.
```

Union operation:

```
224G1A05B1>SELECT course_id
  2  FROM SECTIONS
  3  where semester = 'Fall' AND year= 2009
  4  UNION
  5  (SELECT course_id
  6  FROM SECTIONS
  7  WHERE semester = 'Spring' AND year= 2010);

no rows selected
```

Union all Operation:

```
224G1A05B1>select course_id
  2  from SECTIONS
  3  where semester = 'Fall' and year= 2009
  4  UNION ALL
  5  select course_id
  6  from SECTIONS
  7  where semester = 'Spring' and year= 2010;

no rows selected
```

Intersect Operation:

```
224G1A05B1>SELECT COURSE_ID
  2   FROM SECTION
  3   WHERE SEMESTER = 'FALL' AND YEAR=2006
  4   INTERSECT
  5   SELECT COURSE_ID
  6   FROM SECTION
  7   WHERE SEMESTER = 'SPRING' AND YEAR=2008;

no rows selected
```

Intersect All

```
224G1A05B1>SELECT COURSE_ID
  2   FROM SECTION
  3   WHERE SEMESTER = ‘FALL’ AND YEAR=2006
  4   INTERSECT ALL
  5   SELECT COURSE_ID
  6   FROM SECTION
  7   WHERE SEMESTER = ‘SPRING’ AND YEAR=2008;

no rows selected
```

except all or minus all operations:

```
224G1A05B1>(select course_id
   2  from section
   3  where semester = ‘Fall’ and year=2009)
   4  EXCEPT ALL
   5  (select course_id
   6  from section where semester = ‘Spring’ and year=2010);

   no rows selected
```

5.SPECIAL OPERATIONS

Write SQL queries to perform SPECIAL OPERATIONS (i.e. ISNULL, BETWEEN, LIKE, IN, EXISTS).

```
224G1A05B1>CREATE TABLE Dept(
  2  Dept_id INT NOT NULL PRIMARY KEY,
  3  Dept_name VARCHAR(50) NOT NULL
  4  );

Table created.
```

```
224G1A05B1>CREATE TABLE Employee(
  2  Emp_id INT NOT NULL PRIMARY KEY,
  3  Emp_name VARCHAR(50) NOT NULL,
  4  Emp_salary DECIMAL(10,2) NOT NULL,
  5  Emp_deptId INT NOT NULL,
  6  Emp_deptName VARCHAR(50) NOt NULL,
  7  CONSTRAINT fk_Emp_deptId FOREIGN KEY (Emp_deptId) REFERENCES Dept(Dept_id)
  8  );

Table created.
```

```
224G1A05B1>INSERT INTO Dept (Dept_id,Dept_name)
  2  VALUES(1,'Engineering');

1 row created.

224G1A05B1>INSERT INTO Dept (Dept_id,Dept_name)
  2  VALUES(2,'Marketing');

1 row created.

224G1A05B1>INSERT INTO Dept (Dept_id,Dept_name)
  2  VALUES(3,'sales');

1 row created.
```

```
224G1A05B1>INSERT INTO Dept (Dept_id,Dept_name)
  2  VALUES(4,'Human Resources');

1 row created.

224G1A05B1>INSERT INTO Dept (Dept_id,Dept_name)
  2  VALUES(5,'Finance');

1 row created.
```

```
224G1A05B1>INSERT INTO Employee (Emp_id,Emp_name,Emp_salary,Emp_deptId,Emp_deptName)
  2  VALUES(101,'Alice',50000.00,1,'Engineering');

1 row created.

224G1A05B1>INSERT INTO Employee (Emp_id,Emp_name,Emp_salary,Emp_deptId,Emp_deptName)
  2  VALUES(102,'Bob',45000.00,2,'Marketing');

1 row created.

224G1A05B1>INSERT INTO Employee (Emp_id,Emp_name,Emp_salary,Emp_deptId,Emp_deptName)
  2  VALUES(103,'Charlie',60000.00,3,'sales');

1 row created.

224G1A05B1>INSERT INTO Employee (Emp_id,Emp_name,Emp_salary,Emp_deptId,Emp_deptName)
  2  VALUES(104,'David',40000.00,4,'Human Resorces');

1 row created.

224G1A05B1>INSERT INTO Employee (Emp_id,Emp_name,Emp_salary,Emp_deptId,Emp_deptName)
  2  VALUES(105,'Emily',55000.00,5,'Finance');

1 row created.
```

IS NULL

```
224G1A05B1>SELECT * FROM Dept WHERE Dept_name IS NULL;

no rows selected
```

BETWEEN

```
224G1A05B1>SELECT * FROM Employee WHERE Emp_salary BETWEEN 45000 AND 60000;

    EMP_ID EMP_NAME                                            EMP_SALARY
---------- -------------------------------------------------- ----------
EMP_DEPTID EMP_DEPTNAME
---------- --------------------------------------------------
       101 Alice                                                   50000
         1 Engineering

       102 Bob                                                     45000
         2 Marketing

       103 Charlie                                                 60000
         3 sales
```

```
         3 sales


    EMP_ID EMP_NAME                                            EMP_SALARY
---------- -------------------------------------------------- ----------
EMP_DEPTID EMP_DEPTNAME
---------- --------------------------------------------------
       105 Emily                                                   55000
         5 Finance
```

LIKE

```
         5 Finance


224G1A05B1>SELECT * FROM Employee WHERE Emp_name LIKE 'A%';

    EMP_ID EMP_NAME                                            EMP_SALARY
---------- -------------------------------------------------- ----------
EMP_DEPTID EMP_DEPTNAME
---------- --------------------------------------------------
       101 Alice                                                   50000
         1 Engineering
```

```
224G1A05B1>SELECT * FROM Dept WHERE Dept_name LIKE '%ing';

   DEPT_ID DEPT_NAME
---------- --------------------------------------------------
         1 Engineering
         2 Marketing

224G1A05B1>SELECT * FROM Employee WHERE Emp_name LIKE '_____';

    EMP_ID EMP_NAME                                            EMP_SALARY
---------- -------------------------------------------------- ----------
EMP_DEPTID EMP_DEPTNAME
---------- --------------------------------------------------
       101 Alice                                                   50000
         1 Engineering

       104 David                                                   40000
         4 Human Resorces

       105 Emily                                                   55000
         5 Finance
```

IN

```
224G1A05B1>SELECT * FROM Employee WHERE Emp_id IN (102,104,105);

    EMP_ID EMP_NAME                                           EMP_SALARY
---------- -------------------------------------------------- ----------
EMP_DEPTID EMP_DEPTNAME
---------- --------------------------------------------------
       102 Bob                                                     45000
         2 Marketing

       104 David                                                   40000
         4 Human Resorces

       105 Emily                                                   55000
         5 Finance
```

```
224G1A05B1>SELECT * FROM Employee WHERE Emp_name In ('Alice','Bob','Charlie');

    EMP_ID EMP_NAME                                           EMP_SALARY
---------- -------------------------------------------------- ----------
EMP_DEPTID EMP_DEPTNAME
---------- --------------------------------------------------
       101 Alice                                                   50000
         1 Engineering

       102 Bob                                                     45000
         2 Marketing

       103 Charlie                                                 60000
         3 sales
```

EXISTS

Syntax

SELECT <COLUMNS>FROM<table>
WHERE EXISTS(<subquery);

```
224G1A05B1>SELECT * FROM Employee
  2  WHERE EXISTS  (SELECT 1 FROM Dept
  3  WHERE Employee.Emp_deptId=Dept.Dept_id
  4  HAVING COUNT(*)>2);

no rows selected
```

6.JOIN OPERATIONS

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

Natural JOIN

```
224G1A05B1>SELECT * FROM EMPLOYEE INNER JOIN DEPART
  2  ON EMPLOYEE.EMP_DEPTID = DEPART.DEPT_ID;

    EMP_ID EMP_NAME                                            EMP_SALARY
---------- -------------------------------------------------- ----------
EMP_DEPTID EMP_DEPTNAME                                          DEPT_ID
---------- -------------------------------------------------- ----------
DEPT_NAME
--------------------------------------------------
       101 Alice                                                   50000
         1 Engineering                                                1
Engineering

       102 Bob                                                     45000
         2 Marketing                                                  2
Marketing

    EMP_ID EMP_NAME                                            EMP_SALARY
---------- -------------------------------------------------- ----------
EMP_DEPTID EMP_DEPTNAME                                          DEPT_ID
---------- -------------------------------------------------- ----------
DEPT_NAME
--------------------------------------------------

       103 Charlie                                                 60000
         3 Sales                                                      3
sales

       104 David                                                   40000
         4 HumanResources                                             4


    EMP_ID EMP_NAME                                            EMP_SALARY
---------- -------------------------------------------------- ----------
EMP_DEPTID EMP_DEPTNAME                                          DEPT_ID
---------- -------------------------------------------------- ----------
DEPT_NAME
--------------------------------------------------
HumanResources

       105 Emily                                                   50000
         5 Finance                                                    5
Finance
```

CONDITIONAL JOIN

```
224G1A05B1>SELECT EMPLOYEE.EMP_NAME, EMPLOYEE.EMP_SALARY, DEPART.DEPT_NAME
  2  FROM EMPLOYEE
  3  INNER JOIN DEPART ON EMPLOYEE.EMP_DEPTID = DEPART.DEPT_ID
  4  WHERE EMPLOYEE.EMP_SALARY > 50000;

EMP_NAME                                           EMP_SALARY
-------------------------------------------------- ----------
DEPT_NAME
--------------------------------------------------
Charlie                                                 60000
sales
```

RIGHT OUTER JOIN

```
224G1A05B1>SELECT * FROM EMPLOYEE RIGHT JOIN DEPART
  2  ON EMPLOYEE.EMP_DEPTID = DEPART.DEPT_ID;

    EMP_ID EMP_NAME                                            EMP_SALARY
---------- -------------------------------------------------- ----------
EMP_DEPTID EMP_DEPTNAME                                          DEPT_ID
---------- -------------------------------------------------- ----------
DEPT_NAME
--------------------------------------------------
       101 Alice                                                   50000
         1 Engineering                                                1
Engineering

       102 Bob                                                     45000
         2 Marketing                                                  2
Marketing

    EMP_ID EMP_NAME                                            EMP_SALARY
---------- -------------------------------------------------- ----------
EMP_DEPTID EMP_DEPTNAME                                          DEPT_ID
---------- -------------------------------------------------- ----------
DEPT_NAME
--------------------------------------------------

       103 Charlie                                                 60000
         3 Sales                                                      3
sales

       104 David                                                   40000
         4 HumanResources                                             4
```

```
    EMP_ID EMP_NAME                                            EMP_SALARY
---------- -------------------------------------------------- ----------
EMP_DEPTID EMP_DEPTNAME                                          DEPT_ID
---------- -------------------------------------------------- ----------
DEPT_NAME
--------------------------------------------------
HumanResources

       105 Emily                                                   50000
         5 Finance                                                     5
Finance
```

LEFT OUTER JOIN

```
224G1A05B1>SELECT * FROM EMPLOYEE LEFT JOIN DEPART
  2  ON EMPLOYEE.EMP_DEPTID = DEPART.DEPT_ID;

    EMP_ID EMP_NAME                                            EMP_SALARY
---------- -------------------------------------------------- ----------
EMP_DEPTID EMP_DEPTNAME                                          DEPT_ID
---------- -------------------------------------------------- ----------
DEPT_NAME
--------------------------------------------------
       101 Alice                                                   50000
         1 Engineering                                                1
Engineering

       102 Bob                                                     45000
         2 Marketing                                                  2
Marketing

    EMP_ID EMP_NAME                                            EMP_SALARY
---------- -------------------------------------------------- ----------
EMP_DEPTID EMP_DEPTNAME                                          DEPT_ID
---------- -------------------------------------------------- ----------
DEPT_NAME
--------------------------------------------------

       103 Charlie                                                 60000
         3 Sales                                                      3
sales

       104 David                                                   40000
         4 HumanResources                                             4

    EMP_ID EMP_NAME                                            EMP_SALARY
---------- -------------------------------------------------- ----------
EMP_DEPTID EMP_DEPTNAME                                          DEPT_ID
---------- -------------------------------------------------- ----------
DEPT_NAME
--------------------------------------------------
HumanResources

       105 Emily                                                   50000
         5 Finance                                                     5
Finance
```

FULL OUTER JOIN

```
224G1A05B1>SELECT * FROM EMPLOYEE FULL OUTER JOIN DEPART
  2  ON EMPLOYEE.EMP_DEPTID = DEPART.DEPT_ID;

    EMP_ID EMP_NAME                                            EMP_SALARY
---------- -------------------------------------------------- ----------
EMP_DEPTID EMP_DEPTNAME                                          DEPT_ID
---------- -------------------------------------------------- ----------
DEPT_NAME
--------------------------------------------------
       101 Alice                                                   50000
         1 Engineering                                                1
Engineering

       102 Bob                                                     45000
         2 Marketing                                                  2
Marketing

    EMP_ID EMP_NAME                                            EMP_SALARY
---------- -------------------------------------------------- ----------
EMP_DEPTID EMP_DEPTNAME                                          DEPT_ID
---------- -------------------------------------------------- ----------
DEPT_NAME
--------------------------------------------------

       103 Charlie                                                 60000
         3 Sales                                                      3
sales

       104 David                                                   40000
         4 HumanResources                                             4


    EMP_ID EMP_NAME                                            EMP_SALARY
---------- -------------------------------------------------- ----------
EMP_DEPTID EMP_DEPTNAME                                          DEPT_ID
---------- -------------------------------------------------- ----------
DEPT_NAME
--------------------------------------------------
HumanResources

       105 Emily                                                   50000
         5 Finance                                                    5
Finance
```

7. AGGREGATE OPERATIONS

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

```
224G1A05B1>CREATE TABLE DEPARTMENTS
  2  (DEPT_NAME VARCHAR2(15),
  3  BUILDING VARCHAR2(15),
  4  BUDGET NUMERIC(12,2) CHECK (BUDGET>0),
  5  PRIMARY KEY(DEPT_NAME)
  6  );

Table created.
```

```
224G1A05B1>CREATE TABLE INSTRUCTORS
  2  (ID VARCHAR2(5),
  3  NAME VARCHAR2(20) NOT NULL,
  4  DEPT_NAME VARCHAR2(20),
  5  SALARY NUMERIC(8,2) CHECK (SALARY > 29000),
  6  PRIMARY KEY (ID),
  7  FOREIGN KEY (DEPT_NAME) REFERENCES DEPARTMENT(DEPT_NAME)
  8  ON DELETE SET NULL
  9  );

Table created.
```

```
224G1A05B1>INSERT into DEPARTMENTS values('Comp.sci','Taylor','100000');
1 row created.
224G1A05B1>INSERT into DEPARTMENTS values('Elec.Eng','Taylor','85000');
1 row created.
224G1A05B1>INSERT into DEPARTMENTS values('Finance','painter','12000');
1 row created.
224G1A05B1>INSERT into DEPARTMENTS values('History','painter','50000');
1 row created.
224G1A05B1>INSERT into DEPARTMENTS values('Music','packard','80000');
1 row created.
224G1A05B1>INSERT into DEPARTMENTS values('physics','watson','70000');
1 row created.
```

‘

yg

```
224G1A05B1>INSERT INTO INSTRUCTORS VALUES ('10101', 'Srinivasan', 'Comp. Sci.', '65000');
1 row created.
224G1A05B1>INSERT INTO INSTRUCTORS VALUES ('12121', 'Wu', 'Finance', '90000');
1 row created.
224G1A05B1>INSERT INTO INSTRUCTORS VALUES ('15151', 'Mozart', 'Music', '40000');
1 row created.
224G1A05B1>INSERT INTO INSTRUCTORS VALUES ('22222', 'Einstein', 'Physics', '95000');
1 row created.
224G1A05B1>INSERT INTO INSTRUCTORS VALUES ('32343', 'El Said', 'History', '60000');
1 row created.
224G1A05B1>INSERT INTO INSTRUCTORS VALUES ('33456', 'Gold', 'Physics', '87000');
1 row created.
224G1A05B1>INSERT INTO INSTRUCTORS VALUES ('45565', 'Katz', 'Comp. Sci.', '75000');
1 row created.
```

COUNT

```
224G1A05B1>SELECT COUNT(*) AS TotalEmployees FROM INSTRUCTORS;

TOTALEMPLOYEES
--------------
            12
```

SUM

```
224G1A05B1>SELECT SUM(SALARY) FROM INSTRUCTORS;

SUM(SALARY)
-----------
     898000
```

MIN

```
224G1A05B1>SELECT MIN(SALARY) AS MinSalary FROM INSTRUCTORS;

 MINSALARY
----------
     40000
```

MAX

```
224G1A05B1>SELECT MAX(SALARY) AS MaxSalary FROM INSTRUCTORS;

 MAXSALARY
----------
     95000
```

AVG

```
224G1A05B1>SELECT AVG(SALARY) AS MaxSalary FROM INSTRUCTORS;

 MAXSALARY
----------
74833.3333
```

8. ORACLE BUILT-IN FUNCTIONS

Write SQL queries to perform ORACLE BUILT-IN FUNCTIONS (i.e. DATE, TIME)

case-conversion functions:

```
224G1A05B1>SELECT LOWER('SQL COURSE')
  2  FROM DUAL;

LOWER('SQL
----------
sql course

224G1A05B1>SELECT UPPER('SQL COURSE')
  2  FROM DUAL;

UPPER('SQL
----------
SQL COURSE

224G1A05B1>SELECT INITCAP('SQL course')
  2  FROM DUAL;

INITCAP('S
----------
Sql Course
```

character manipulation functions:

```
224G1A05B1>SELECT CONCAT('HELLO','WORLD')
  2  FROM DUAL;

CONCAT('HE
----------
HELLOWORLD

224G1A05B1>SELECT SUBSTR('HELLO WORLD',1,5)
  2  FROM DUAL;

SUBST
-----
HELLO
```

```
224G1A05B1>SELECT LPAD(SALARY, 10, '*')
  2  FROM INSTRUCTOR;

no rows selected

224G1A05B1>SELECT RPAD(SALARY,10,'*')
  2  FROM INSTRUCTOR;

no rows selected

224G1A05B1>SELECT REPLACE('jack and jue','j','BL')
  2  FROM DUAL;

REPLACE('JACKA
--------------
BLack and BLue
```

Number Functions:

```
224G1A05B1>SELECT ROUND(45.626,2)
  2  FROM DUAL;

ROUND(45.626,2)
---------------
          45.63

224G1A05B1>SELECT ROUND(45.626,0)
  2  FROM DUAL;

ROUND(45.626,0)
---------------
              46
```

```
224G1A05B1>SELECT ROUND(45.626,-1)
  2  FROM DUAL;

ROUND(45.626,-1)
----------------
              50

224G1A05B1>SELECT ROUND(45.626,-2)
  2  FROM DUAL;

ROUND(45.626,-2)
----------------
               0
```

```
224G1A05B1>SELECT TRUNC(45.626,2)
  2  FROM DUAL;

TRUNC(45.626,2)
---------------
          45.62
```

```
224G1A05B1>SELECT TRUNC(45.626,0)
  2  FROM DUAL;

TRUNC(45.626,0)
---------------
             45
```

```
224G1A05B1>SELECT TRUNC(45.626,-1)
  2  FROM DUAL;

TRUNC(45.626,-1)
----------------
              40
```

```
224G1A05B1>SELECT TRUNC(45.626,-2)
  2  FROM DUAL;

TRUNC(45.626,-2)
----------------
               0
```

Date functions:

```
224G1A05B1>SELECT SYSDATE
  2  FROM DUAL;

SYSDATE
---------
30-JAN-24
```

```
224G1A05B1>SELECT MONTHS_BETWEEN(SYSDATE,'15-FEB-20')
  2  FROM DUAL;

MONTHS_BETWEEN(SYSDATE,'15-FEB-20')
-----------------------------------
                          47.5122827
```

```
224G1A05B1>SELECT ADD_MONTHS(SYSDATE,2)
  2  FROM DUAL;

ADD_MONTH
---------
30-MAR-24
```

```
224G1A05B1>SELECT NEXT_DAY(SYSDATE,'THURSDAY')
  2  FROM DUAL;

NEXT_DAY(
---------
01-FEB-24
```

```
224G1A05B1>SELECT LAST_DAY(SYSDATE)
  2  FROM DUAL;

LAST_DAY(
---------
31-JAN-24
```

9.KEY CONSTRAINTS

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

NOT NULL COnstraint Example

```
224G1A05B1>CREATE TABLE STUDENT (
  2  ID int NOT NULL,
  3  LastName varchar(255) NOT NULL,
  4  FirstName varchar(255) NOT NULL,
  5  Age int
  6  );

Table created.
```

```
224G1A05B1>ALTER TABLE STUDENT
  2  MODIFY Age int NOT NULL;

Table altered.
```

UNIQUE CONSTRAINT Example

```
224G1A05B1>CREATE TABLE STUDENTS (
  2  ID int NOT NULL,
  3  LastName varchar(255) NOT NULL,
  4  FirstName varchar(255),
  5  Age int,
  6  CONSTRAINT UC_Person UNIQUE (ID,LastName)
  7  );

Table created.
```

```
224G1A05B1>ALTER TABLE STUDENTS
  2  DROP CONSTRAINT UC_Person;

Table altered.
```

```
224G1A05B1>DESC STUDENTS;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 ID                                        NOT NULL NUMBER(38)
 LASTNAME                                  NOT NULL VARCHAR2(255)
 FIRSTNAME                                          VARCHAR2(255)
 AGE                                                NUMBER(38)
```

PRIMARY KEY CONSTRAINT Example:

```
224G1A05B1>CREATE TABLE Persons(
  2  ID int NOT NULL,
  3  LastName varchar(255) NOT NULL,
  4  FirstName varchar(255),
  5  Age int,
  6  CONSTRAINT PK_Person PRIMARY KEY (ID,LastName)
  7  );

Table created.
```

```
224G1A05B1>ALTER TABLE Persons
  2  ADD CONSTRAINT PK_Person PRIMARY KEY (ID,LastName);
ADD CONSTRAINT PK_Person PRIMARY KEY (ID,LastName)
                         *
ERROR at line 2:
ORA-02260: table can have only one primary key


224G1A05B1>ALTER TABLE Persons
  2  DROP CONSTRAINT PK_Person;

Table altered.
```

```
224G1A05B1>DESC Persons;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 ID                                        NOT NULL NUMBER(38)
 LASTNAME                                  NOT NULL VARCHAR2(255)
 FIRSTNAME                                          VARCHAR2(255)
 AGE                                                NUMBER(38)
```

CHECK CONSTRAINTS Example:

```
224G1A05B1>CREATE TABLE People(
  2  ID int NOT NULL,
  3  LastName varchar(255) NOT NULL,
  4  FirstName varchar(255),
  5  Age int,
  6  City varchar(255),
  7  CONSTRAINT CHK_Person CHECK (Age>=18 AND City='Sandnes')
  8  );

Table created.
```

ALTER

```
224G1A05B1>ALTER TABLE People
  2  ADD CONSTRAINT CHK_PeopleAge CHECK (Age>=18 AND City='Sandness');

Table altered.

224G1A05B1>ALTER TABLE People
  2  DROP CONSTRAINT chk_PeopleAge;

Table altered.
```

DEFAULT CONSTRAINTS Example:

```
224G1A05B1>ALTER TABLE People
  2  MODIFY City DEFAULT 'Sandness';

Table altered.
```

```
224G1A05B1>CREATE TABLE Orders(
  2  ID int NOT NULL,
  3  OrderNumber int NOT NULL,
  4  OrderDate date DEFAULT GETDATE()
  5  );
OrderDate date DEFAULT GETDATE()
                       *
ERROR at line 4:
ORA-04044: procedure, function, package, or type is not allowed here
```

```
224G1A05B1>ALTER TABLE People MODIFY city DEFAULT NULL;

Table altered.
```

10. FACTORIAL

Write a PL/SQL program for calculating the factorial of a given number

```
SQL> SET SQLPROMPT "224G1A05B1>"
224G1A05B1>Set serveroutput on
224G1A05B1>DECLARE
  2  fac NUMBER :=1;
  3  n NUMBER :=10;
  4  BEGIN
  5  WHILE n>0 LOOP
  6  fac:=n*fac;
  7  n:=n-1;
  8  END LOOP;
  9  DBMS_OUTPUT.PUT_LINE(FAC);
 10  END;
 11  /
3628800

PL/SQL procedure successfully completed.
```

```
224G1A05B1>declare
  2  n number;
  3  fact number:=1;
  4  i number :=1;
  5  c number:=0;
  6  begin
  7  n:=&n;
  8  if(n<0)
  9  then
 10  dbms_output.put_line('factorial of negative number does not exist');
 11  end if;
 12  while(i<=n)
 13  loop
 14  fact :=fact*i;
 15  i :=i + 1;
 16  end loop;
 17  dbms_output.put_line('factorial of '||n||' is '||fact);
 18  end;
 19  /
Enter value for n: 4
old   7: n:=&n;
new   7: n:=4;
factorial of 4 is 24

PL/SQL procedure successfully completed.
```

11.PRIME NUMBER OR NOT

Write a PL/SQL program for finding the given number is prime number or not.

```
224G1A05B1>DECLARE
  2  n NUMBER;
  3  i NUMBER;
  4  temp NUMBER;
  5  BEGIN
  6  n := 13;
  7  i :=2;
  8  temp :=1;
  9  FOR i IN 2..n/2
 10  LOOP
 11  IF MOD(n,i)=0
 12  THEN
 13  temp :=0;
 14  EXIT;
 15  END IF;
 16  END LOOP;
 17  IF temp = 1
 18  THEN
 19  DBMS_OUTPUT.PUT_LINE(n||'n is a prime number');
 20  ELSE
 21  DBMS_OUTPUT.PUT_LINE(n||'n is not a prime number');
 22  END IF;
 23  END;
 24  /
13n is a prime number

PL/SQL procedure successfully completed.
```

```
224G1A05B1>declare
  2  n number;
  3  i number;
  4  flag number;
  5
  6  begin
  7  i:=2;
  8  flag:=1;
  9  n:=&n;
 10
 11  for i in 2..n/2
 12  loop
 13  if mod(n,i)=0
 14  then
 15  flag:=0;
 16  exit;
 17  end if;
 18  end loop;
 19
 20  if flag=1
 21  then
 22  dbms_output.put_line('prime');
 23  else
 24  dbms_output.put_line('not prime');
 25  end if;
 26  end;
 27  /
Enter value for n: 7
old   9: n:=&n;
new   9: n:=7;
prime

PL/SQL procedure successfully completed.
```

12.FIBONACCI

Write a PL/SQL program for displaying the Fibonacci series up to an integer

```
224G1A05B1>DECLARE
  2  FIRST NUMBER:=0;
  3  SECOND NUMBER :=1;
  4  TEMP NUMBER;
  5  N NUMBER :=5;
  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  /
SERIES:
0
1
1
2
3
5

PL/SQL procedure successfully completed.
```

13.STORED PROCEDURE

Write PL/SQL program to implement Stored Procedure on table.

SYNTAX:
CREATE [OR REPLACE] PROCEDURE procedure_name
[(parameter [,parameter])]
(IS | AS)
[declaration_section]
BEGIN
executable_section
[EXCEPTION exception_section]
END [procedure_name];

Example:

```
224G1A05B1>DECLARE
  2  a number;
  3  b number;
  4  c number;
  5  PROCEDURE findMin(x IN number, y IN number, z OUT number) IS
  6  BEGIN
  7  IF x<y THEN
  8    z:=x;
  9  ELSE
 10  z:=y;
 11  END IF;
 12  END;
 13  BEGIN
 14    a:=23;
 15    b:=45;
 16  findMin(a,b,c);
 17  dbms_output.put_line('Minimum of (23,45) : ' || c);
 18  END;
 19  /
Minimum of (23,45) : 23

PL/SQL procedure successfully completed.
```

14.STORED FUNCTION

Write PL/SQL program to implement Stored Function on table.

SYNTAX:
CREATE [OR REPLACE] FUNCTION function_name
[(parameter [,parameter])]
RETURN return_datatype
(IS | AS)
[declaration_section]
BEGIN executable_section
[EXCEPTION exception_section]
END [procedure_name];

Example:

```
224G1A05B1>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.

224G1A05B1>DECLARE
  2  num number;
  3  factorial number;
  4  BEGIN num:=&n;
  5  factorial :=fact(num);
  6  dbms_output.put_line(' Factorial ' || num || ' is ' || factorial);
  7  END;
  8  /
Enter value for n: 8
old   4: BEGIN num:=&n;
new   4: BEGIN num:=8;
Factorial 8 is 40320

PL/SQL procedure successfully completed.
```

15.IMPLEMENT TRIGGER

Write PL/SQL program to implement Trigger on table

Syntax:
CREATE [OR REPLACE] TRIGGER TRIGGER_NAME
{BEFORE | AFTER | INSTEAD OF }
{INSERT [OR] | UPDATE [OR] | DELETE}
[OF COL_NAME]
ON TABLE_NAME
[REFERENCING OLD AS O NEW AS N]
[FOR EACH ROW]
WHEN (CONDITION)
DECLARE
DECLARATION-STATEMENTS
BEGIN
EXECUTABLE-STATEMENTS
EXCEPTION
EXCEPTION-HANDLING-STATEMENTS
END;

```
224G1A05B1>CREATE TABLE Departments
  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.
```

```
224G1A05B1>CREATE TABLE Instructors
  2  (ID VARCHAR2(5),
  3  NAME VARCHAR2(20) NOT NULL,
  4  DEPT_NAME VARCHAR2(20),
  5  SALARY NUMERIC(8,2) CHECK (SALARY>29000),
  6  PRIMARY KEY(ID),
  7  FOREIGN KEY (DEPT_NAME) REFERENCES Departments(DEPT_NAME)
  8  );

Table created.
```

```
224G1A05B1>CREATE OR REPLACE TRIGGER display_SALARY_changes
  2  BEFORE UPDATE ON Instructors
  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.
```

```
224G1A05B1>DECLARE
  2  total_rows number(2);
  3  BEGIN
  4  UPDATE Instructors
  5  SET SALARY = SALARY+5000;
  6  IF sql%notfound THEN
  7  dbms_output.put_line('no Instructorsn updated');
  8  ELSIF sql%found THEN
  9  total_rows :=sql%rowcount;
 10  dbms_output.put_line(total_rows || ' Instructors updated ');
 11  END IF;
 12  END;
 13  /
no Instructorsn updated

PL/SQL procedure successfully completed.
```

16.IMPLEMENT CURSOR

Write PL/SQL program to implement Cursor on table

Declare the cursor:
SYNTAX:
CURSOR cursor_name IS select_statement;

Open the cursor
SYNTAX:
OPEN cursor_name;

Fetch the cursor
SYNTAX:
FETCH cursor_name INTO variable_list;

Close the cursor:
SYNTAX:
Close cursor_name;

```
224G1A05B1>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));

Table created.
```

```
224G1A05B1>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  /
101 ABHISHEK KUMAR atp
102 SUPRAJA gooty
103 dinesh hyderabad

PL/SQL procedure successfully completed.
```