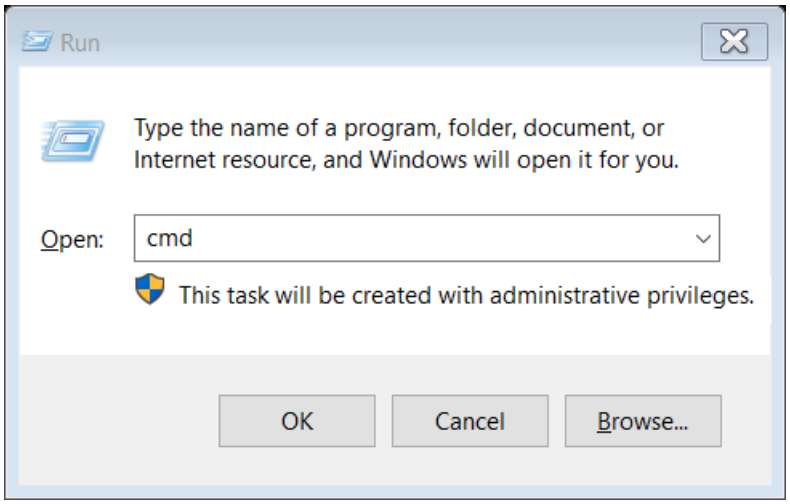
1. Open the command prompt Press WIN+R , type CMD



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

```
SQL> CREATE USER C#CSE_571 IDENTIFIED BY sweety;

User created.

SQL> grant all privileges to C#CSE_571;

Grant succeeded.
```

3. Now sign in with the new user.

```
SQL*Plus: Release 21.0.0.0.0 - Production on Mon Jan 8 10:18:22 2024
Version 21.3.0.0.0

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

Enter user-name: system
Enter password:
Last Successful login time: Sun Dec 31 2023 13:29:26 +05:30

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

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 table _ name (

column1 data_ type [constraint]

[, column2 data_ type [constraint]] [,

PRIMARY KEY (column1 [, column2])]

[, FOREIGN KEY (column1 [, column2]) REFERENCES table_ name] [,CONSTRAINT constraint]);

Example:

```
224G1A0571>create table orders(
  2  orderId int NOT NULL,
  3  orderNumber int,
  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 t_ name

ADD constraint [ADD constraint];

Syntax 3:

ALTER TABLE t_ name

DROP {PRIMARY KEY | COLUMN Colum_ name | CONSTRAINT Constrain_ name);

Syntax 4:

ALTER TABLE t_ name

ENABLE CONSTRAINT constrain_ name;

Example:

```
224G1A0571>ALTER TABLE Orders
  2  ADD(MAil varchar(32));

Table altered.
```

```
224G1A0571>ALTER TABLE Orders
  2  DROP COLUMN mail;

Table altered.
```

DESC Orders:

```
224G1A0571>DESC Orders
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 ORDERID                                   NOT NULL NUMBER(38)
 ORDERNUMBER                                        NUMBER(38)
 PERSONID                                           NUMBER(38)
```

DROP TABLE :

Syntax:

DROP TABLE t_ name;

Example :

```
224G1A0571>DROP TABLE EMP;

Table dropped.
```

TRUNCATE TABLE :

Syntax:

TRUNCATE TABLE table_name;

Example :

```
224G1A0571>TRUNCATE table DEPARTMENT;

Table truncated.
```

2.DML COMMANDS

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

```
224G1A0571>CREATE TABLE Student (
  2  Roll_no int,
  3  Name varchar(50),
  4  Age int,
  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:

```
224G1A0571>INSERT INTO Student(Roll_no,Name,Age)
  2  VALUES(3,'Charlie',20);

1 row created.

224G1A0571>INSERT INTO Student(Roll_no,Name,Age)
  2  VALUES(1,'sweety',30);

1 row created.
```

SELECT

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

Example :

```
224G1A0571>SELECT *FROM Student;

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



         1 sweety                                              30



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

UPDATE

Syntax:

UPDATE t_ name SET [column_name1= value_1, column_name2= value_2,...]

WHERE CONDITION;

```
224G1A0571>UPDATE Student
  2  SET Age = Age + 1;

2 rows updated.
```

DELETE

Syntax: DELETE FROM t_ Name WHERE condition;

```
224G1A0571>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>

View syntax:

CREATE VIEW VIEW_NAME AS Example-1:

Create a view for clerk to check instructor information with out salary visibility :

CREATE VIEW FACULTY AS SELECT ID,NAME,DEPT_NAME FROM INSTRUCTOR;

```
224G1A0571>CREATE VIEW FACULTY AS
  2  SELECT ID,NAME,branch,salary
  3  FROM Department;

View created.
```

```
224G1A0571>CREATE VIEW STUDENT AS
  2  SELECT ID,NAME,DEPT_NAME
  3  FROM INSTRUCTOR;

View created.
```

Create a view that able to satify the above constraints to do insertion, deletion, and Update.

```
224G1A0571>CREATE VIEW DEPARTMENTS_TOTAL_SALARY(DEPT_NAME, TOTAL_SALARY) AS
  2  SELECT DEPT_NAME, SUM(SALARY)
  3  FROM INSTRUCTOR
  4  GROUP BY DEPT_NAME;

View created.
```

Create a view that able to satify the above constraints to do insertion, deletion, and Update.

```
224G1A0571>CREATE VIEW HISTORY_INSTRUCTORS AS
  2  SELECT *
  3  FROM INSTRUCTOR
  4  WHERE DEPT_NAME= 'HISTORY';

View created.
```

To verify the output of view use the following statement.

Commands to insert, Delete and update view :

```
224G1A0571>INSERT INTO History_instructors VALUES('58584','Dacid Coy','History',34000);

1 row created.

224G1A0571>Update History_instructors SET name='james robert' where Id='58584';

0 rows updated.

224G1A0571>Delete FROM History_instructors where id='58584';

0 rows deleted.

224G1A0571>SELECT * FROM instructor where dept_name='history';

no rows selected
```

To verify the output of view use the following statement.

SELECT * FROM history_instructors ;

```
224G1A0571>SELECT * FROM history_instructors;

no rows selected
```

Select * from instructors;

```
224G1A0571>SELECT * From instructor;

ID    NAME                 DEPT_NAME              SALARY
----- -------------------- -------------------- ----------
10101 Srinivasan           Comp. Sci.              75000
12121 Wu                   Finance                100000
15151 Mozart               Music                   50000
22222 Einstein             Physics                105000
32343 El Said              History                 70000
33456 Gold                 Physics                 97000
45565 Katz                 Comp. Sci.              85000
58583 Califieri            History                 72000
76543 Singh                Finance                 90000
76766 Crick                Biology                 82000
83821 Brandt               Comp. Sci.             102000

ID    NAME                 DEPT_NAME              SALARY
----- -------------------- -------------------- ----------
98345 Kim                  Elec. Eng.              90000
58584 Dacid Coy            History                 34000

13 rows selected.
```

An equivalent relation of view without using view as original relation

Commands to insert, Delete and update view

```
224G1A0571>CREATE VIEW HISTORY_Department AS
  2  SELECT *
  3  FROM Department
  4  WHERE Name='HISTORY';

View created.
```

```
224G1A0571>SELECT * FROM Department;

NAME                            ID BRANCH                    SALARY
-------------------- ---------- -------------------- ----------
sweety                         571 cse                       20000
pranathi                       572 csd                       30000
saniya                         594 csd                       40000
Athika                         500 csm                       30000
praneesha                      573 cse                       20000
supraja                        502 csm                       60000
varsha                         540 csd                       30000
Thanu                          592 cse                       50000
Ithika                         570 csd                       40000

9 rows selected.
```

DELETE VIEW:

DROP VIEW view_name;

```
224G1A0571>DROP view History_instructors;

View dropped.
```

4. RELATIONAL SET OPERATIONS

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

Classroom table :

```
224G1A0571>CREATE TABLE CLASSROOM
  2  (BUILDING VARCHAR2(15),
  3  ROOM_NUMBER VARCHAR2(7),
  4  CAPACITY NUMERIC(4,0),
  5  PRIMARY KEY (BUILDING, ROOM_NUMBER)
  6  );

Table created.
```

Section table :

```
224G1A0571>CREATE TABLE SECTION
  2  (COURSE_ID VARCHAR2(8), SEC_ID VARCHAR2(8),
  3  SEMESTER VARCHAR2(6) CHECK (SEMESTER IN ('FALL', 'WINTER',
  4  'SPRING', 'SUMMER')),
  5  YEAR NUMERIC(4,0) CHECK (YEAR > 1701 AND YEAR < 2100),
  6  BUILDING VARCHAR2(15),
  7  ROOM_NUMBER VARCHAR2(7),
  8  TIME_SLOT_ID VARCHAR2(4),
  9  FOREIGN KEY (BUILDING, ROOM_NUMBER) REFERENCES CLASSROOM(BUILDING,
 10  ROOM_NUMBER)
 11  ON DELETE SET NULL
 12  );

Table created.
```

Instances of classroom table :

```
224G1A0571>INSERT INTO classroom VALUES ('Packard', '101', '500');

1 row created.

224G1A0571>INSERT INTO classroom VALUES ('Painter', '514', '10');

1 row created.

224G1A0571>INSERT INTO classroom VALUES ('Taylor', '3128', '70');

1 row created.

224G1A0571>INSERT INTO classroom VALUES ('Watson', '100', '30');

1 row created.

224G1A0571>INSERT INTO classroom VALUES ('Watson', '120', '50');

1 row created.
```

Union operation:

```
224G1A0571>SELECT course_id
  2  FROM section
  3  where semester = 'Fall' AND year= 2009
  4  UNION
  5  (SELECT course_id
  6  FROM section
  7  WHERE semester = 'Spring' AND year= 2010);

no rows selected
```

Union all Operation:

```
224G1A0571>(select SEC_ID
  2  from section
  3  where semester = 'Fall' and year= 2009)
  4  UNION ALL
  5  (select SEC_ID
  6  from section
  7  where semester = 'Spring' and year= 2010);

no rows selected
```

Intersect Operation:

```
224G1A0571>(select SEC_ID
  2  from section
  3  where semester = 'Fall' and year= 2009)
  4  INTERSECT
  5  (select SEC_ID
  6  from section
  7  where semester = 'Spring' and year= 2010);

no rows selected
```

Intersect all operation:

```
224G1A0571>(select SEC_ID
  2  from section
  3  where semester = 'fall' and year= 2009)
  4  INTERSECT ALL
  5  (select SEC_ID
  6  from section
  7  where semester = 'spring' and year= 2010);

no rows selected
```

except or minus operation:

```
224G1A0571>SELECT COURSE_ID
  2  from section
  3  where semester = 'Fall' and year= 2009
  4  EXCEPT
  5  (select COURSE_ID
  6  from section
  7  where semester = 'Spring' and year= 2010);

no rows selected
```

except all or minus all operations

```
224G1A0571>(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).

Emp table :

```
224G1A0571>CREATE TABLE Emp(
  2  ID VARCHAR2(5),
  3  NAME VARCHAR2(20) NOT NULL,
  4  DEPT_NAME VARCHAR2(20),
  5  SALARY NUMERIC(8,2) CHECK (SALARY > 29000)
  6  );

Table created.
```

```
224G1A0571>CREATE table Department (
  2  Name varchar(20),
  3  Id int,
  4  branch varchar(20),
  5  salary int
  6  );

Table created.
```

```
224G1A0571>INSERT INTO Department VALUES('&Name','&ID','&branch','&salary');
Enter value for name: Romio
Enter value for id: 202
Enter value for branch: cse
Enter value for salary: 20000
old   1: INSERT INTO Department VALUES('&Name','&ID','&branch','&salary')
new   1: INSERT INTO Department VALUES('Romio','202','cse','20000')

1 row created.

224G1A0571>/
Enter value for name: juliet
Enter value for id: 203
Enter value for branch: csd
Enter value for salary: 30000
old   1: INSERT INTO Department VALUES('&Name','&ID','&branch','&salary')
new   1: INSERT INTO Department VALUES('juliet','203','csd','30000')

1 row created.
```

```
224G1A0571>INSERT INTO Student(Roll_no,Name,Age)
  2  VALUES(3,'Charlie',20);

1 row created.

224G1A0571>INSERT INTO Student(Roll_no,Name,Age)
  2  VALUES(1,'sweety',30);

1 row created.
```

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)

```
224g1a0571>CREATE TABLE COURSE1
  2  (COURSE_ID VARCHAR2(8),
  3  TITLE VARCHAR2(50),
  4  DEPT_NAME VARCHAR2(20),
  5  CREDITS NUMERIC(2,0) CHECK (CREDITS > 0),
  6  PRIMARY KEY (COURSE_ID)
  7  );

Table created.
```

```
224g1a0571>insert into course1 values ('BIO-101', 'Intro. to Biology', 'Biology', '4');

1 row created.

224g1a0571>insert into course1 values ('BIO-301', 'Genetics', 'Biology', '4');

1 row created.

224g1a0571>insert into course1 values ('BIO-399', 'Computational Biology', 'Biology', '3');

1 row created.

224g1a0571>insert into course1 values ('CS-101', 'Intro. to Computer Science', 'Comp. Sci.', '4');

1 row created.

224g1a0571>insert into course1 values ('CS-190', 'Game Design', 'Comp. Sci.', '4');

1 row created.

224g1a0571>insert into course1 values ('CS-315', 'Robotics', 'Comp. Sci.', '3');

1 row created.

224g1a0571>insert into course1 values ('CS-319', 'Image Processing', 'Comp. Sci.', '3');

1 row created.
```

```
224g1a0571>CREATE TABLE TEACHES1
  2  (ID VARCHAR2(5),
  3  COURSE_ID VARCHAR2(8),
  4  SEC_ID VARCHAR2(8),
  5  SEMESTER VARCHAR2(6),
  6  YEAR NUMERIC(4,0),
  7  PRIMARY KEY (ID, COURSE_ID, SEC_ID, SEMESTER, YEAR)
  8  );

Table created.
```

```
224g1a0571>insert into teaches1 values ('10101', 'CS-101', '1', 'Fall', '2009');

1 row created.

224g1a0571>insert into teaches1 values ('10101', 'CS-315', '1', 'Spring', '2010');

1 row created.

224g1a0571>insert into teaches1 values ('10101', 'CS-347', '1', 'Fall', '2009');

1 row created.

224g1a0571>insert into teaches1 values ('12121', 'FIN-201', '1', 'Spring', '2010');

1 row created.

224g1a0571>insert into teaches1 values ('15151', 'MU-199', '1', 'Spring', '2010');

1 row created.

224g1a0571>insert into teaches1 values ('22222', 'PHY-101', '1', 'Fall', '2009');

1 row created.

224g1a0571>insert into teaches1 values ('32343', 'HIS-351', '1', 'Spring', '2010');

1 row created.

224g1a0571>insert into teaches1 values ('45565', 'CS-101', '1', 'Spring', '2010');

1 row created.
```

Natural JOIN

```
224g1a0571>select * from COURSE1 INNER JOIN TEACHES1
  2  ON COURSE1.DEPT_NAME = TEACHES1.SEC_ID;

no rows selected
```

OUTER JOINS

```
224g1a0571>select * from COURSE1 FULL OUTER JOIN TEACHES1
  2  ON COURSE1.DEPT_NAME = TEACHES1.SEC_ID;

COURSE_I TITLE                                              DEPT_NAME
-------- -------------------------------------------------- --------------------
   CREDITS ID    COURSE_I SEC_ID   SEMEST       YEAR
---------- ----- -------- -------- ------ ----------

          10101 CS-101   1        Fall         2009


          10101 CS-315   1        Spring       2010


          10101 CS-347   1        Fall         2009


COURSE_I TITLE                                              DEPT_NAME
-------- -------------------------------------------------- --------------------
   CREDITS ID    COURSE_I SEC_ID   SEMEST       YEAR
---------- ----- -------- -------- ------ ----------

          12121 FIN-201  1        Spring       2010


          15151 MU-199   1        Spring       2010


          22222 PHY-101  1        Fall         2009


COURSE_I TITLE                                              DEPT_NAME
-------- -------------------------------------------------- --------------------
   CREDITS ID    COURSE_I SEC_ID   SEMEST       YEAR
---------- ----- -------- -------- ------ ----------

          32343 HIS-351  1        Spring       2010


          45565 CS-101   1        Spring       2010

CS-101   Intro. to Computer Science                         Comp. Sci.
         4
```

```
COURSE_I TITLE                                              DEPT_NAME
-------- -------------------------------------------------- --------------------
   CREDITS ID    COURSE_I SEC_ID   SEMEST       YEAR
---------- ----- -------- -------- ------ ----------
CS-190   Game Design                                        Comp. Sci.
         4

CS-315   Robotics                                           Comp. Sci.
         3

CS-319   Image Processing                                   Comp. Sci.
         3


COURSE_I TITLE                                              DEPT_NAME
-------- -------------------------------------------------- --------------------
   CREDITS ID    COURSE_I SEC_ID   SEMEST       YEAR
---------- ----- -------- -------- ------ ----------
BIO-101  Intro. to Biology                                  Biology
         4

BIO-301  Genetics                                           Biology
         4

BIO-399  Computational Biology                              Biology
         3


15 rows selected.
```

left outer join operation.

```
224g1a0571>select * from COURSE1 LEFT JOIN TEACHES1
  2  ON COURSE1.DEPT_NAME = TEACHES1.SEC_ID;

COURSE_I TITLE                                              DEPT_NAME
-------- -------------------------------------------------- --------------------
   CREDITS ID     COURSE_I SEC_ID   SEMEST       YEAR
---------- ----- -------- -------- ------ ----------
CS-101   Intro. to Computer Science                         Comp. Sci.
         4

CS-190   Game Design                                        Comp. Sci.
         4

CS-315   Robotics                                           Comp. Sci.
         3


COURSE_I TITLE                                              DEPT_NAME
-------- -------------------------------------------------- --------------------
   CREDITS ID     COURSE_I SEC_ID   SEMEST       YEAR
---------- ----- -------- -------- ------ ----------
CS-319   Image Processing                                   Comp. Sci.
         3

BIO-101  Intro. to Biology                                  Biology
         4

BIO-301  Genetics                                           Biology
         4


COURSE_I TITLE                                              DEPT_NAME
-------- -------------------------------------------------- --------------------
   CREDITS ID     COURSE_I SEC_ID   SEMEST       YEAR
---------- ----- -------- -------- ------ ----------
BIO-399  Computational Biology                              Biology
         3


7 rows selected.
```

RIGHT OUTER JOIN

```
224g1a0571>select * from COURSE1 RIGHT JOIN TEACHES1
  2  ON COURSE1.DEPT_NAME = TEACHES1.SEC_ID;

COURSE_I TITLE                                              DEPT_NAME
-------- -------------------------------------------------- --------------------
   CREDITS ID    COURSE_I SEC_ID   SEMEST       YEAR
---------- ----- -------- -------- ------ ----------

          10101 CS-101   1        Fall         2009


          10101 CS-315   1        Spring       2010


          10101 CS-347   1        Fall         2009


COURSE_I TITLE                                              DEPT_NAME
-------- -------------------------------------------------- --------------------
   CREDITS ID    COURSE_I SEC_ID   SEMEST       YEAR
---------- ----- -------- -------- ------ ----------

          12121 FIN-201  1        Spring       2010


          15151 MU-199   1        Spring       2010


          22222 PHY-101  1        Fall         2009


COURSE_I TITLE                                              DEPT_NAME
-------- -------------------------------------------------- --------------------
   CREDITS ID    COURSE_I SEC_ID   SEMEST       YEAR
---------- ----- -------- -------- ------ ----------

          32343 HIS-351  1        Spring       2010


          45565 CS-101   1        Spring       2010


8 rows selected.
```

7. AGGREGATE OPERATIONS

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

Instructor table :

```
224G1A0571>CREATE TABLE INSTRUCTOR(
  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  );

Table created.
```

Instance of instructor values :

```
224G1A0571>insert into instructor values ('10101', 'Srinivasan', 'Comp. Sci.', '65000');

1 row created.

224G1A0571>insert into instructor values ('12121', 'Wu', 'Finance', '90000');

1 row created.

224G1A0571>insert into instructor values ('15151', 'Mozart', 'Music', '40000');

1 row created.

224G1A0571>insert into instructor values ('22222', 'Einstein', 'Physics', '95000');

1 row created.

224G1A0571>insert into instructor values ('32343', 'El Said', 'History', '60000');

1 row created.

224G1A0571>insert into instructor values ('33456', 'Gold', 'Physics', '87000');

1 row created.
```

Department table :

```
224G1A0571>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.
```

Instances of department table :

```
224G1A0571>insert into department values ('Comp. Sci.', 'Taylor', '100000');

1 row created.

224G1A0571>insert into department values ('Elec. Eng.', 'Taylor', '85000');

1 row created.

224G1A0571>insert into department values ('Finance', 'Painter', '120000');

1 row created.

224G1A0571>insert into department values ('History', 'Painter', '50000');

1 row created.

224G1A0571>insert into department values ('Music', 'Packard', '80000');

1 row created.

224G1A0571>insert into department values ('Physics', 'Watson', '70000');

1 row created.
```

Course table :

```
224G1A0571>CREATE TABLE COURSE
  2  (COURSE_ID VARCHAR2(8),
  3  TITLE VARCHAR2(50),
  4  DEPT_NAME VARCHAR2(20),
  5  CREDITS NUMERIC(2,0) CHECK (CREDITS > 0),
  6  PRIMARY KEY (COURSE_ID),
  7  FOREIGN KEY (DEPT_NAME) REFERENCES DEPARTMENT(DEPT_NAME)
  8  ON DELETE SET NULL
  9  );

Table created.
```

Instances of course table :

```
224G1A0571>insert into course values ('CS-101', 'Intro. to Computer Science', 'Comp. Sci.', '4');

1 row created.

224G1A0571>insert into course values ('CS-190', 'Game Design', 'Comp. Sci.', '4');

1 row created.

224G1A0571>insert into course values ('CS-315', 'Robotics', 'Comp. Sci.', '3');

1 row created.

224G1A0571>insert into course values ('CS-319', 'Image Processing', 'Comp. Sci.', '3');

1 row created.

224G1A0571>insert into course values ('CS-347', 'Database System Concepts', 'Comp. Sci.', '3');

1 row created.

224G1A0571>insert into course values ('EE-181', 'Intro. to Digital Systems', 'Elec. Eng.', '3');

1 row created.
```

Teaches table

```
224G1A0571>CREATE TABLE TEACHES
  2  (ID VARCHAR2(5),
  3  COURSE_ID VARCHAR2(8),
  4  SEC_ID VARCHAR2(8),
  5  SEMESTER VARCHAR2(6),
  6  YEAR NUMERIC(4,0),
  7  PRIMARY KEY (ID, COURSE_ID, SEC_ID, SEMESTER, YEAR)
  8  );

Table created.
```

Instance of teaches table :

```
224G1A0571>insert into teaches values ('10101', 'CS-101', '1', 'Fall', '2009');

1 row created.

224G1A0571>insert into teaches values ('10101', 'CS-315', '1', 'Spring', '2010');

1 row created.

224G1A0571>insert into teaches values ('10101', 'CS-347', '1', 'Fall', '2009');

1 row created.

224G1A0571>insert into teaches values ('12121', 'FIN-201', '1', 'Spring', '2010');

1 row created.

224G1A0571>insert into teaches values ('15151', 'MU-199', '1', 'Spring', '2010');

1 row created.

224G1A0571>insert into teaches values ('22222', 'PHY-101', '1', 'Fall', '2009');

1 row created.

224G1A0571>insert into teaches values ('32343', 'HIS-351', '1', 'Spring', '2010');

1 row created.
```

To count instructors who teaches in the year 2010 having Spring semester :

```
224G1A0571>select count (distinct ID )
  2  from teaches
  3  where semester = 'Spring' and year = 2010;

COUNT(DISTINCTID)
-----------------
                4

224G1A0571>select count (*)
  2  from course;

  COUNT(*)
----------
        12
```

COUNT :

```
224g1a0571>SELECT COUNT(*) AS TotalEmployees FROM INSTRUCTOR;

TOTALEMPLOYEES
--------------
            13
```

SUM :

```
224G1A0571>SELECT SUM(SALARY) FROM DEPARTMENT;

SUM(SALARY)
-----------
     410000
```

MIN :

```
224G1A0571>SELECT MIN(SALARY) AS MinSalary FROM DEPARTMENT;

 MINSALARY
----------
     20000
```

MAX :

```
224G1A0571>SELECT MAX(SALARY) AS MaxSalary FROM DEPARTMENT;

 MAXSALARY
----------
     60000
```

AVG:

```
224G1A0571>SELECT AVG(SALARY) AS AvgSalary FROM DEPARTMENT;

 AVGSALARY
----------
34166.6667
```

8. ORACLE BUILT-IN FUNCTIONS

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

case-conversion functions:

```
224G1A0571>SELECT LOWER('SQL Course')
  2  FROM DUAL;

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

224G1A0571>SELECT UPPER('SQL Course')
  2  FROM DUAL;

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

224G1A0571>SELECT INITCAP('SQL course')
  2  FROM DUAL;

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

character manipulation functions:

```
224G1A0571>SELECT CONCAT('HELLO', 'WORLD')
  2  FROM DUAL;

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

224G1A0571>SELECT SUBSTR('HELLO WORLD',1,5)
  2  FROM DUAL;

SUBST
-----
HELLO
```

```
224G1A0571>SELECT LPAD(SALARY, 10, '*')
  2  FROM INSTRUCTOR;

LPAD(SALARY,10,'*')
----------------------------------------
*****65000
*****90000
*****40000
*****95000
*****60000
*****87000
*****75000
*****62000
*****80000
*****72000
*****92000

LPAD(SALARY,10,'*')
----------------------------------------
*****80000

12 rows selected.
```

Number Functions:

```
224G1A0571>SELECT ROUND(45.626,2)
  2  FROM DUAL;

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

224G1A0571>SELECT ROUND(45.626,0)
  2  FROM DUAL;

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

224G1A0571>SELECT ROUND(45.626,-1)
  2  FROM DUAL;

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

224G1A0571>SELECT ROUND(45.626,-2)
  2  FROM DUAL;
```

```
224G1A0571>SELECT TRUNC(45.626, 2)
  2  FROM DUAL;

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

224G1A0571>SELECT TRUNC(45.626, 0)
  2  FROM DUAL;

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

224G1A0571>SELECT TRUNC(45.626, -1)
  2  FROM DUAL;

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

224G1A0571>SELECT TRUNC(45.626, -2)
  2  FROM DUAL;

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

Date functions:

```
224G1A0571>SELECT SYSDATE
  2  FROM DUAL;

SYSDATE
---------
11-JAN-24

224G1A0571>SELECT MONTHS_BETWEEN(SYSDATE,'15-FEB-20')
  2  FROM DUAL;

MONTHS_BETWEEN(SYSDATE,'15-FEB-20')
-----------------------------------
                          46.8986705

224G1A0571>SELECT ADD_MONTHS(SYSDATE, 2)
  2  FROM DUAL;

ADD_MONTH
---------
11-MAR-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

```
224G1A0571>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.
```

```
224G1A0571>ALTER TABLE students
  2  DROP CONSTRAINT UC_Person;

Table altered.
```

UNIQUE CONSTRAINT Example

```
224G1A0571>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.
```

```
224G1A0571>ALTER TABLE students
  2  DROP CONSTRAINT UC_Person;

Table altered.
```

PRIMARY KEY CONSTRAINT Example:

```
224G1A0571>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.
```

```
224G1A0571>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


224G1A0571>ALTER TABLE Persons
  2  DROP CONSTRAINT PK_Person;

Table altered.
```

```
224G1A0571>DESC persons;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 ID                                        NOT NULL NUMBER(38)
 LASTNAME                                  NOT NULL VARCHAR2(255)
 FIRSTNAME                                          VARCHAR2(255)
 AGE                                                NUMBER(38)
```

CHECK CONSTRAINTS Example:

```
224G1A0571>CREATE TABLE Persons1(
  2  ID int NOT NULL,
  3  LastName varchar(255) NOT NULL,
  4  FirstName varchar(255),
  5  Age int
  6  );

Table created.
```

DEFAULT CONSTRAINTS Example:

```
224G1A0571>ALTER TABLE Persons MODIFY city DEFAULT NULL;
ALTER TABLE Persons MODIFY city DEFAULT NULL
                             *
ERROR at line 1:
ORA-00904: "CITY": invalid identifier
```

```
224G1A0571>CREATE TABLE Orders1(
  2   ID int NOT NULL,
  3   OrderNumber int NOT NULL
  4  );

Table created.
```

```
224G1A0571>ALTER TABLE Persons MODIFY city DEFAULT NULL;
ALTER TABLE Persons MODIFY city DEFAULT NULL
                             *
ERROR at line 1:
ORA-00904: "CITY": invalid identifier
```

10. FACTORIAL

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

```
SQL> SET SQLPROMPT "224G1A0571>"
224G1A0571>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  /

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.

```
224G1A0571>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||' is a prime number');
 20  ELSE
 21  DBMS_OUTPUT.PUT_LINE(n||' is not a prime number');
 22  END IF;
 23  END;
 24  /
13 is a prime number
```

12.FIBONACCI

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

```
224G1A0571>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:

```
224G1A0571>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:

```
224G1A0571>CREATE FUNCTION factorial(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.
```

```
224G1A0571>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: 5
old   4: BEGIN num:= &n;
new   4: BEGIN num:= 5;

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;

```
224G1A0571>CREATE TABLE INSTRUCTOR
  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  );

Table created.
```

```
224G1A0571>insert into instructor values ('10101', 'Srinivasan', 'Comp. Sci.', '65000');

1 row created.

224G1A0571>insert into instructor values ('12121', 'Wu', 'Finance', '90000');

1 row created.

224G1A0571>insert into instructor values ('15151', 'Mozart', 'Music', '40000');

1 row created.

224G1A0571>insert into instructor values ('22222', 'Einstein', 'Physics', '95000');

1 row created.

224G1A0571>insert into instructor values ('32343', 'El Said', 'History', '60000');

1 row created.

224G1A0571>insert into instructor values ('33456', 'Gold', 'Physics', '87000');

1 row created.

224G1A0571>insert into instructor values ('45565', 'Katz', 'Comp. Sci.', '75000');

1 row created.

224G1A0571>insert into instructor values ('58583', 'Califieri', 'History', '62000');

1 row created.

224G1A0571>insert into instructor values ('76543', 'Singh', 'Finance', '80000');

1 row created.

224G1A0571>insert into instructor values ('76766', 'Crick', 'Biology', '72000');

1 row created.

224G1A0571>insert into instructor values ('83821', 'Brandt', 'Comp. Sci.', '92000');

1 row created.
```

```
224G1A0571>CREATE OR REPLACE TRIGGER display_salary_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.
```

```
224G1A0571>DECLARE
  2   total_rows number(2);
  3   BEGIN
  4   UPDATE instructor
  5   SET salary = salary + 5000;
  6   IF sql%notfound THEN
  7   dbms_output.put_line('no instructors 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  /
12 instructors 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;

```
224G1A0571>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.
```

```
224G1A0571>INSERT INTO customers VALUES(3, 'Mahesh',24,'Ghaziabad',29000);

1 row created.

224G1A0571>INSERT INTO customers VALUES(4, 'chandhan',25,'Noida',31000);

1 row created.

224G1A0571>INSERT INTO customers VALUES(5, 'Alex', 21, 'paris',33000);

1 row created.

224G1A0571>INSERT INTO customers VALUES(6, 'Sunita',20,'delhi',35000);

1 row created.
```

```
224G1A0571>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  /
3 Mahesh Ghaziabad
4 chandhan Noida
5 Alex paris
6 Sunita delhi

PL/SQL procedure successfully completed.
```