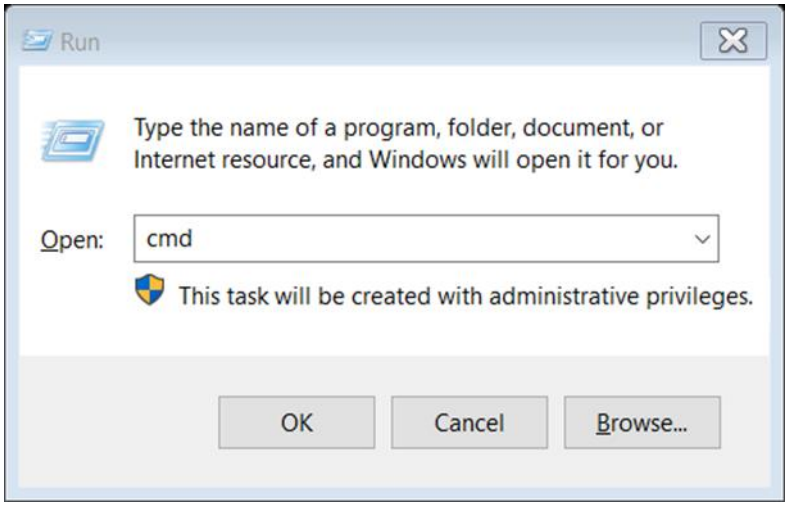
1.Open the command prompt Press WIN+R , type cmd



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

```
SQL> create user c##573 identified by praneesha;

User created.
```

```
SQL> GRANT ALL PRIVILEGES TO c##573;

Grant succeeded.
```

3. Now sign in with the new user.

```
C:\Users\prane>sqlplus

SQL*Plus: Release 21.0.0.0.0 - Production on Mon Jan 8 11:49:56 2024
Version 21.3.0.0.0

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

Enter user-name: c##573
Enter password:

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

LIST OF EXPERIMENTS

1. Write SQL queries to CREATE TABLES for various databases using DDL commands (i.e. CREATE, ALTER, DROP, TRUNCATE).
2. Write SQL queries to MANIPULATE TABLES for various databases using DML commands (i.e. INSERT, SELECT, UPDATE, DELETE,).
3. Write SQL queries to create VIEWS for various databases (i.e. CREATE VIEW, UPDATE VIEW, ALTER VIEW, and DELETE VIEW).
4. Write SQL queries to perform RELATIONAL SET OPERATIONS (i.e. UNION, UNION ALL, INTERSECT, MINUS, CROSS JOIN, NATURAL JOIN).
5. Write SQL queries to perform SPECIAL OPERATIONS (i.e. ISNULL, BETWEEN, LIKE, IN, EXISTS).
6. Write SQL queries to perform JOIN OPERATIONS (i.e. CONDITIONAL JOIN, EQUI JOIN, LEFT OUTER JOIN, RIGHT OUTER JOIN, FULL OUTER JOIN)
7. Write SQL queries to perform AGGREGATE OPERATIONS (i.e. SUM, COUNT, AVG, MIN, MAX).
8. Write SQL queries to perform ORACLE BUILT-IN FUNCTIONS (i.e. DATE, TIME).
9. Write SQL queries to perform KEY CONSTRAINTS (i.e. PRIMARY KEY, FOREIGN KEY, UNIQUE NOT NULL, CHECK, DEFAULT).
10. Write a PL/SQL program for calculating the factorial of a given number.
11. Write a PL/SQL program for finding the given number is prime number or not.
12. Write a PL/SQL program for displaying the Fibonacci series up to an integer.
13. Write PL/SQL program to implement Stored Procedure on table.
14. Write PL/SQL program to implement Stored Function on table.
15. Write PL/SQL program to implement Trigger on table.
16. Write PL/SQL program to implement Cursor on table.

EXPERIMENT-1

Write SQL queries to create tables for various databases using DDL commands(CREATE,ALTER,DROP,TRUNCATE)

CREATE TABLE:

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]);

```
224G1A0573>CREATE TABLE persons(
  2      person_id NUMBER,
  3      first_name VARCHAR2(50) NOT NULL,
  4      last_name VARCHAR2(50) NOT NULL,
  5      PRIMARY KEY(person_id)
  6  );

Table created.
```

ALTER TABLE:

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;

```
224G1A0573>Alter table persons
  2  add( mail varchar2
  3  (15));

Table altered.
```

DROP TABLE:

DROP TABLE

Syntax:

DROP TABLE table_name;

Example:

```
224G1A0573>desc persons;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------
 PERSON_ID                                 NOT NULL NUMBER
 FIRST_NAME                                NOT NULL VARCHAR2(50)
 LAST_NAME                                 NOT NULL VARCHAR2(50)
 MAIL                                               VARCHAR2(15)
```

The person table contains the following columns , when we want to delete the specific columns we use drop command. The drop command is used as follows:

```
224G1A0573>alter table persons
  2  drop column mail;

Table altered.
```

After the usage of drop command the table columns are as follows:

```
224G1A0573>desc persons;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------
 PERSON_ID                                 NOT NULL NUMBER
 FIRST_NAME                                NOT NULL VARCHAR2(50)
 LAST_NAME                                 NOT NULL VARCHAR2(50)
```

TRUNCATE TABLE:

TRUNCATE TABLE

Syntax:

TRUNCATE TABLE table_name;

Example:

To delete all the rows from the existing table we use the truncate table. Initially surcharges table is like

```
224G1A0573>desc surcharges;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------
 SURCHARGE_ID                              NOT NULL NUMBER
 SURCHARGE_NAME                            NOT NULL VARCHAR2(255)
 AMOUNT                                             NUMBER(9,2)
```

After applying the truncate command

```
224G1A0573>truncate table surcharges;

Table truncated.

224G1A0573>select * from surcharges;

no rows selected
```

EXPERIMENT-2

Write SQL queries to MANIPULATE TABLES for various databases using DML commands

(INSERT , SELECT, UPDATE, DELETE,).

INSERT

Syntax:

INSERT INTO tablename

VALUES (value1,value2,...,valuen);

Syntax 2:

INSERT INTO tablename

(column1, column2,...,column) VALUES (value1, value2,...,valuen);

TABLE CREATION:

```
224G1A0573>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.
```

INSERT COMMAND:

To insert a new row into a table, you use the oracle INSERT statment

```
224G1A0573>INSERT INTO classroom VALUES ('Packard', '101', '500');

1 row created.

224G1A0573>INSERT INTO classroom VALUES ('Painter', '514', '10');

1 row created.

224G1A0573>INSERT INTO classroom VALUES ('Taylor', '3128', '70');

1 row created.

224G1A0573>INSERT INTO classroom VALUES ('Watson', '100', '30');

1 row created.

224G1A0573>INSERT INTO classroom VALUES ('Watson', '120', '50');

1 row created.
```

SELECT

Syntax:

SELECT *

FROM <table_name>;

```
224G1A0573>Select * from classroom;

BUILDING        ROOM_NU   CAPACITY
--------------- ------- ----------
Packard         101             500
Painter         514              10
Taylor          3128             70
Watson          100              30
Watson          120              50
```

UPDATE

Syntax:

UPDATE table_name SET [column_name1= value_1, column_name2= value_2,...]

WHERE CONDITION;

```
224G1A0573>update classroom
  2  set capacity = capacity+10;

5 rows updated.
```

DELETE

Syntax:

DELETE FROM table_Name WHERE condition;

Example:

```
224G1A0573>Delete from classroom
  2  where capacity<10;

0 rows deleted.
```

EXPERIMENT-3

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>

```
224G1A0573>CREATE VIEW FACULTY AS
  2  SELECT ID,NAME,DEPT_NAME
  3  FROM INSTRUCTOR;

View created.
```

```
224g1a0573>SELECT * FROM INSTRUCTORS;

ID    NAME                 DEPT_NAME                SALARY
----- -------------------- -------------------- ----------
10101 Srinivasan           Comp. Sci.                65000
12121 Wu                   Finance                   90000
15151 Mozart               Music                     40000
22222 Einstein             Physics                   95000
32343 El Said              History                   60000
33456 Gold                 Physics                   87000
45565 Katz                 Comp. Sci.                75000
58583 Califieri            History                   62000
76543 Singh                Finance                   80000
76766 Crick                Biology                   72000
83821 Brandt               Comp. Sci.                92000

ID    NAME                 DEPT_NAME                SALARY
----- -------------------- -------------------- ----------
98345 Kim                  Elec. Eng.                80000

12 rows selected.
```

An equivalent relation of view without using view as original relation:

```
224g1a0573>RUN
  1  CREATE VIEW PHYSICS_FALL_2009 AS
  2  SELECT COURSE.COURSE_ID,  BUILDING
  3  FROM COURSE, SECTION
  4  WHERE COURSE.COURSE_ID = SECTION.COURSE_ID
  5  AND COURSE.DEPT_NAME = 'PHYSICS'
  6  AND SECTION.SEMESTER = 'FALL'
  7* AND SECTION.YEAR = '2009'

View created.
```

```
224g1a0573>RUN
  1  CREATE VIEW PHYSICS_FALL_2009_WATSON AS
  2  (SELECT COURSE_ID
  3  FROM (SELECT COURSE.COURSE_ID, BUILDING
  4  FROM COURSE, SECTION
  5  WHERE COURSE.COURSE_ID = SECTION.COURSE_ID
  6  AND COURSE.DEPT_NAME = 'PHYSICS'
  7  AND SECTION.SEMESTER = 'FALL'
  8  AND SECTION.YEAR = '2009')
  9* WHERE BUILDING= 'WATSON')

View created.
```

```
224g1a0573>SELECT COURSE_ID
  2  FROM PHYSICS_FALL_2009
  3  WHERE BUILDING= 'WATSON';

no rows selected
```

DROP VIEW :

```
224g1a0573>DROP VIEW FACULTY;

View dropped.
```

EXPERIMENT-4

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

CREATING CLASSROOM TABLE AND INSERTING VALUES:

```
224g1a0573>run
  1  CREATE TABLE CLASS
  2  (BUILDING VARCHAR2(15),
  3  ROOM_NUMBER VARCHAR2(7),
  4  CAPACITY NUMERIC(4,0),
  5  PRIMARY KEY (BUILDING, ROOM_NUMBER)
  6* )

Table created.
```

```
224g1a0573>INSERT INTO class VALUES ('Packard', '101', '500');

1 row created.

224g1a0573>INSERT INTO class VALUES ('Painter', '514', '10');

1 row created.

224g1a0573>INSERT INTO class VALUES ('Taylor', '3128', '70');

1 row created.

224g1a0573>INSERT INTO class VALUES ('Watson', '100', '30');

1 row created.

224g1a0573>INSERT INTO class VALUES ('Watson', '120', '50');

1 row created.
```

CREATING SECTION TABLE AND INSERTING VALUES:

```
224g1a0573>run
  1  CREATE TABLE SECTIONS
  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  PRIMARY KEY (COURSE_ID, SEC_ID, SEMESTER, YEAR),
 10  FOREIGN KEY (BUILDING, ROOM_NUMBER) REFERENCES CLASSROOM(BUILDING,
 11  ROOM_NUMBER)
 12  ON DELETE SET NULL
 13* )

Table created.
```

UNION:

```
224g1a0573>run
  1  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:

```
224g1a0573>run
  1  select course_id
  2  from section
  3  where semester = 'Fall' and year= 2009
  4  UNION ALL
  5  select course_id
  6  from section
  7* where semester = 'Spring' and year= 2010

no rows selected
```

INTERSECT:

```
224g1a0573>run
  1  (select course_id
  2  from section
  3  where semester = 'Fall' and year= 2009)
  4  INTERSECT
  5  (select course_id
  6  from section
  7* where semester = 'Spring' and year= 2010)

no rows selected
```

INTERSECT ALL:

```
224g1a0573>RUN
  1  (select course_id
  2  from section
  3  where semester = 'Fall' and year= 2009)
  4  INTERSECT ALL
  5  (select course_id
  6  from section
  7* where semester = 'Spring' and year= 2010)

no rows selected
```

EXPECT:

```
224g1a0573>RUN
  1  (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:

```
224g1a0573>RUN
  1  (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
```

EXPERIMENT-5

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

CREATING TABLE:

```
224G1A0573>run
  1  CREATE TABLE INSTRUCT
  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.
```

INSERTING VALUES INTO INSTRUCT TABLE

```
224G1A0573>insert into instruct values ('10101', 'Srinivasan', 'Comp. Sci.', '65000');

1 row created.

224G1A0573>insert into instruct values ('12121', 'Wu', 'Finance', '90000');

1 row created.

224G1A0573>insert into instruct values ('15151', 'Mozart', 'Music', '40000');

1 row created.

224G1A0573>insert into instruct values ('22222', 'Einstein', 'Physics', '95000');

1 row created.

224G1A0573>insert into instruct values ('32343', 'El Said', 'History', '60000');

1 row created.

224G1A0573>insert into instruct values ('33456', 'Gold', 'Physics', '87000');

1 row created.

224G1A0573>insert into instruct values ('45565', 'Katz', 'Comp. Sci.', '75000');

1 row created.

224G1A0573>insert into instruct values ('58583', 'Califieri', 'History', '62000');

1 row created.

224G1A0573>insert into instruct values ('76543', 'Singh', 'Finance', '80000');

1 row created.

224G1A0573>insert into instruct values ('76766', 'Crick', 'Biology', '72000');

1 row created.

224G1A0573>insert into instruct values ('83821', 'Brandt', 'Comp. Sci.', '92000');

1 row created.

224G1A0573>insert into instruct values ('98345', 'Kim', 'Elec. Eng.', '80000');

1 row created.
```

ISNULL:

```
224G1A0573>select name
  2  from instruct
  3  where salary is null;

no rows selected
```

```
224G1A0573>select name
  2  from instruct
  3  where salary is not null;

NAME
--------------------
Srinivasan
Wu
Mozart
Einstein
El Said
Gold
Katz
Califieri
Singh
Crick
Brandt

NAME
--------------------
Kim

12 rows selected.
```

BETWEEN:

```
224G1A0573>select name
  2  from instruct
  3  where salary between 90000 and 100000;

NAME
--------------------
Wu
Einstein
Brandt
```

LIKE:

```
224G1A0573>select dept_name
  2  from department
  3  where building like '%Watson%';

no rows selected
```

IN:

```
224g1a0573>SELECT * FROM EMPLOYEE WHERE EMP_ID IN (102,104,105);

    EMP_ID EMP_NAME                                            EMP_SALARY
---------- -------------------------------------------------- ----------
EMP_DEPTID EMP_DEPTNAME
---------- --------------------------------------------------
       102 BOB                                                    450000
         2 MARKETING

       104 DAVID                                                  400000
         4 HUMAN RESOURCES

       105 EMILY                                                  500000
         5 FINANCE
```

```
224g1a0573>SELECT * FROM EMPLOYEE WHERE EMP_NAME IN ('ALICE','BOB','CHARLIE');

    EMP_ID EMP_NAME                                            EMP_SALARY
---------- -------------------------------------------------- ----------
EMP_DEPTID EMP_DEPTNAME
---------- --------------------------------------------------
       101 ALICE                                                  500000
         1 ENGINEERING

       102 BOB                                                    450000
         2 MARKETING

       103 CHARLIE                                                600000
         3 SALES
```

EXPERIMENT-6

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

CREATE TABLE:

```
224g1a0573>RUN
  1  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.
```

```
224g1a0573>CREATE TABLE DEPT(
  2  DEPT_ID INT NOT NULL PRIMARY KEY,
  3  DEPT_NAME VARCHAR(50) NOT NULL
  4  );

Table created.
```

INSERTING VALUES:

```
224g1a0573>INSERT INTO DEPT VALUES (1,'ENGINEERING');

1 row created.

224g1a0573>INSERT INTO DEPT VALUES (2,'MARKETING');

1 row created.

224g1a0573>INSERT INTO DEPT VALUES (3,'SALES');

1 row created.

224g1a0573>INSERT INTO DEPT VALUES (4,'HUMAN RESOURCES');

1 row created.

224g1a0573>INSERT INTO DEPT VALUES (5,'FINANCE');

1 row created.
```

```
224g1a0573>INSERT INTO EMPLOYEE VALUES(101,'ALICE',500000.00,1,'ENGINEERING');

1 row created.

224g1a0573>INSERT INTO EMPLOYEE VALUES(102,'BOB',450000.00,2,'MARKETING');

1 row created.

224g1a0573>INSERT INTO EMPLOYEE VALUES(103,'CHARLIE',600000.00,3,'SALES');

1 row created.

224g1a0573>INSERT INTO EMPLOYEE VALUES(104,'DAVID',400000.00,4,'HUMAN RESOURCES');


1 row created.

224g1a0573>INSERT INTO EMPLOYEE VALUES(105,'EMILY',500000.00,5,'FINANCE');

1 row created.
```

Natural JOIN

```
224g1a0573>SELECT * FROM EMPLOYEE INNER JOIN DEPT
  2  ON EMPLOYEE.EMP_DEPTID = DEPT.DEPT_ID;

    EMP_ID EMP_NAME                                            EMP_SALARY
---------- -------------------------------------------------- ----------
EMP_DEPTID EMP_DEPTNAME                                          DEPT_ID
---------- -------------------------------------------------- ----------
DEPT_NAME
--------------------------------------------------
       101 ALICE                                                 500000
         1 ENGINEERING                                              1
ENGINEERING

       102 BOB                                                   450000
         2 MARKETING                                                2
MARKETING

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

       103 CHARLIE                                               600000
         3 SALES                                                    3
SALES

       104 DAVID                                                 400000
         4 HUMAN RESOURCES                                          4

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

       105 EMILY                                                 500000
         5 FINANCE                                                  5
FINANCE
```

CONDITIONAL JOIN

```
224g1a0573>SELECT EMPLOYEE.EMP_NAME,EMPLOYEE.EMP_SALARY,DEPT.DEPT_NAME
  2  FROM EMPLOYEE
  3  INNER JOIN DEPT ON EMPLOYEE.EMP_DEPTID = DEPT.DEPT_ID
  4  WHERE EMPLOYEE.EMP_SALARY > 50000;

EMP_NAME                                           EMP_SALARY
-------------------------------------------------- ----------
DEPT_NAME
--------------------------------------------------

ALICE                                                  500000
ENGINEERING

BOB                                                    450000
MARKETING

CHARLIE                                                600000
SALES


EMP_NAME                                           EMP_SALARY
-------------------------------------------------- ----------
DEPT_NAME
--------------------------------------------------

DAVID                                                  400000
HUMAN RESOURCES

EMILY                                                  500000
FINANCE
```

RIGHT OUTER JOIN

```
224g1a0573>SELECT * FROM EMPLOYEE RIGHT JOIN DEPT
  2  ON EMPLOYEE.EMP_DEPTID = DEPT.DEPT_ID;

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

       101 ALICE                                                 500000
         1 ENGINEERING                                              1
ENGINEERING

       102 BOB                                                   450000
         2 MARKETING                                                2
MARKETING

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

       103 CHARLIE                                               600000
         3 SALES                                                    3
SALES

       104 DAVID                                                 400000
         4 HUMAN RESOURCES                                          4

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

       105 EMILY                                                 500000
         5 FINANCE                                                  5
FINANCE
```

LEFT OUTER JOIN

```
224g1a0573>SELECT * FROM EMPLOYEE LEFT JOIN DEPT
  2  ON EMPLOYEE.EMP_DEPTID = DEPT.DEPT_ID;

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

       101 ALICE                                                   500000
         1 ENGINEERING                                                1
ENGINEERING

       102 BOB                                                     450000
         2 MARKETING                                                  2
MARKETING

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

       103 CHARLIE                                                 600000
         3 SALES                                                      3
SALES

       104 DAVID                                                   400000
         4 HUMAN RESOURCES                                            4

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

       105 EMILY                                                   500000
         5 FINANCE                                                    5
FINANCE
```

FULL OUTER JOIN

```
224g1a0573>SELECT * FROM EMPLOYEE FULL OUTER JOIN DEPT
  2  ON EMPLOYEE.EMP_DEPTID = DEPT.DEPT_ID;

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

       101 ALICE                                              500000
         1 ENGINEERING                                             1
ENGINEERING

       102 BOB                                                450000
         2 MARKETING                                               2
MARKETING

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

       103 CHARLIE                                            600000
         3 SALES                                                   3
SALES

       104 DAVID                                              400000
         4 HUMAN RESOURCES                                         4

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

       105 EMILY                                              500000
         5 FINANCE                                                 5
FINANCE
```

EXPERIMENT 7

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

CREATING INSTRUCTOR TABLE:

```
224G1A0573>run
  1  CREATE TABLE INSTRUCT
  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.
```

INSERTING VALUES INTO INSTRUCTOR TABLE

```
224G1A0573>insert into instruct values ('10101', 'Srinivasan', 'Comp. Sci.', '65000');

1 row created.

224G1A0573>insert into instruct values ('12121', 'Wu', 'Finance', '90000');

1 row created.

224G1A0573>insert into instruct values ('15151', 'Mozart', 'Music', '40000');

1 row created.

224G1A0573>insert into instruct values ('22222', 'Einstein', 'Physics', '95000');

1 row created.

224G1A0573>insert into instruct values ('32343', 'El Said', 'History', '60000');

1 row created.

224G1A0573>insert into instruct values ('33456', 'Gold', 'Physics', '87000');

1 row created.

224G1A0573>insert into instruct values ('45565', 'Katz', 'Comp. Sci.', '75000');

1 row created.

224G1A0573>insert into instruct values ('58583', 'Califieri', 'History', '62000');

1 row created.

224G1A0573>insert into instruct values ('76543', 'Singh', 'Finance', '80000');

1 row created.

224G1A0573>insert into instruct values ('76766', 'Crick', 'Biology', '72000');

1 row created.

224G1A0573>insert into instruct values ('83821', 'Brandt', 'Comp. Sci.', '92000');

1 row created.

224G1A0573>insert into instruct values ('98345', 'Kim', 'Elec. Eng.', '80000');

1 row created.
```

AVERAGE:

TO FIND AVERAGE SALARY OF COMPUTER SCIENCE DEPARTMENT:

```
224G1A0573>select avg(salary) as avg_salary
  2  from instruct
  3  where dept_name = 'Comp. Sci.';

AVG_SALARY
----------
77333.3333
```

COUNT:

TO FIND THE COUNT OF INSTRUCTORS:

```
224G1A0573>select count (*)
  2  from instruct;

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

SUM:

TO FIND THE SUM OF THE SALARY OF HISTORY DEPARTMENT:

```
224G1A0573>Select sum (salary)
  2  from instruct
  3  where dept_name = 'History';

SUM(SALARY)
-----------
     122000
```

MAX:

TO FIND THE MAXIMUM SALARY FROM INSTRUCTORS:

```
224G1A0573>select max(salary)
  2  from instruct;

MAX(SALARY)
-----------
      95000
```

MIN:

TO FIND THE MINMUM SALARY FROM INSTRUCTORS:

```
224G1A0573>select min(salary)
  2  from instruct;

MIN(SALARY)
-----------
      40000
```

EXPERIMENT 8

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

DATE FUNCTIONS:

```
SQL> SELECT SYSDATE
  2  FROM DUAL;

SYSDATE
---------
14-DEC-23
```

```
224G1A0573>SELECT SYSDATE
  2  FROM DUAL;

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

TO KNOW THE NUMBER OF MONTHS PRESENT BETWEEN TWO SPECIFIED DATES:

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

MONTHS_BETWEEN(SYSDATE,'15-FEB-20')
-----------------------------------
                          45.9866237
```

```
224G1A0573>SELECT MONTHS_BETWEEN(SYSDATE,'15-FEB-20')
  2  FROM DUAL;

MONTHS_BETWEEN(SYSDATE,'15-FEB-20')
-----------------------------------
                         47.5114139
```

TO ADD MONTHS:

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

ADD_MONTH
---------
14-FEB-24
```

```
224G1A0573>SELECT ADD_MONTHS(SYSDATE,2)
  2  FROM DUAL;

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

TO KNOW WHEN THE SPECIFIC DAY OCCURS:

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

NEXT_DAY(
---------
21-DEC-23
```

```
224G1A0573>SELECT NEXT_DAY(SYSDATE,'THURSDAY')
  2  FROM DUAL;

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

TO KNOW THE LAST DAY:

```
SQL> SELECT LAST_DAY(SYSDATE)
  2  FROM DUAL;

LAST_DAY(
---------
31-DEC-23
```

```
224g1a0573>SELECT ROUND(45.626,2)
  2  FROM DUAL;

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

224g1a0573>SELECT ROUND(45.626,0)
  2  FROM DUAL;

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

224g1a0573>SELECT ROUND(45.626,-1)
  2  FROM DUAL;

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

224g1a0573>SELECT ROUND(45.626,-2)
  2  FROM DUAL;

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

```
224G1A0573>SELECT LAST_DAY(SYSDATE)
  2  FROM DUAL;

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

```
224g1a0573>SELECT TRUNC(45.626, 2)
  2  FROM DUAL;

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

TIME FUNCTIONS:

```
SQL> SELECT SYSDATE AS CURRENT_DATE_TIME, EXTRACT(YEAR FROM SYSDATE) AS ONLY_CURRENT_YEAR
  2  FROM DUAL;

CURRENT_D ONLY_CURRENT_YEAR
--------- -----------------
14-DEC-23              2023
```

```
224G1A0573>SELECT SYSDATE AS CURRENT_DATE, EXTRACT(YEAR FROM SYSDATE) AS ONLY_CURRENT_YEAR
  2  FROM DUAL;

CURRENT_D ONLY_CURRENT_YEAR
--------- -----------------
30-JAN-24              2024
```

EXPERIMENT-9

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

NOT NULL COnstraint Example:

```
224g1a0573>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.
```

```
224g1a0573>ALTER TABLE students
  2  DROP CONSTRAINT UC_Person;

Table altered.
```

```
224g1a0573>ALTER TABLE student
  2  MODIFY Age int NOT NULL;

Table altered.
```

UNIQUE CONSTRAINT Example

```
224g1a0573>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.
```

```
224g1a0573>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:

```
224g1a0573>run
  1  CREATE TABLE Personed (
  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.
```

```
224g1a0573>desc personed;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 ID                                        NOT NULL NUMBER(38)
 LASTNAME                                  NOT NULL VARCHAR2(255)
 FIRSTNAME                                          VARCHAR2(255)
 AGE                                                NUMBER(38)
```

CHECK CONSTRAINT:

```
224g1a0573>RUN
  1  CREATE TABLE Pers (
  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.
```

DEFAULT CONSTRAINTS:

```
224g1a0573>ALTER TABLE PERS
  2  MODIFY CITY DEFAULT 'SADNESS';

Table altered.
```

EXPERIMENT -10

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

```
224G1A0573>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.
```

EXPERIMENT-11

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

```
224G1A0573>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

PL/SQL procedure successfully completed.
```

EXPERIMENT 12

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

```
224G1A0573>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.
```

EXPERIMENT-13

13. 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];

```
224G1A0573>CREATE TABLE SAILOR(ID NUMBER(10) PRIMARY KEY,NAME VARCHAR2(100));

Table created.

224G1A0573>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.
```

```
224G1A0573>INSERT INTO SAILOR VALUES(101,'Anu');

1 row created.

224G1A0573>INSERT INTO SAILOR VALUES(102,'Pranee');

1 row created.

224G1A0573>INSERT INTO SAILOR VALUES(103,'Nithya');

1 row created.
```

```
224G1A0573>INSERT INTO SAILOR VALUES(104,'Usha');

1 row created.

224G1A0573>INSERT INTO SAILOR VALUES(105,'Kavitha');

1 row created.

224G1A0573>INSERT INTO SAILOR VALUES(106,'Sangeetha');

1 row created.
```

```
224G1A0573>DECLARE
  2  CNT NUMBER;
  3  BEGIN
  4  SELECT COUNT(*) INTO CNT FROM SAILOR;
  5  DBMS_OUTPUT.PUT_LINE(CNT||' RECORD IS INSERTED SUCCESSFULLY');
  6  END;
  7  /
6 RECORD IS INSERTED SUCCESSFULLY

PL/SQL procedure successfully completed.
```

```
224G1A0573>DROP procedure insertuser;

Procedure dropped.
```

EXPERIMENT 14

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];

```
224G1A0573>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.
```

```
224G1A0573>DECLARE
  2   N3 NUMBER(2);
  3  BEGIN
  4   N3 := ADDER(11,22);
  5   DBMS_OUTPUT.PUT_LINE('ADDITION IS: ' || N3);
  6   END;
  7   /
ADDITION IS: 33

PL/SQL procedure successfully completed.
```

```
224G1A0573>DROP function adder;

Function dropped.
```

EXPERIMENT 15

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;

```
224G1A0573>run
  1  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* )

Table created.
```

```
224G1A0573>insert into instructors values ('10101', 'Srinivasan', 'Comp. Sci.', '65000');

1 row created.

224G1A0573>insert into instructors values ('12121', 'Wu', 'Finance', '90000');

1 row created.

224G1A0573>insert into instructors values ('15151', 'Mozart', 'Music', '40000');

1 row created.

224G1A0573>insert into instructors values ('22222', 'Einstein', 'Physics', '95000');

1 row created.

224G1A0573>insert into instructors values ('32343', 'El Said', 'History', '60000');

1 row created.

224G1A0573>insert into instructors values ('33456', 'Gold', 'Physics', '87000');

1 row created.

224G1A0573>insert into instructors values ('45565', 'Katz', 'Comp. Sci.', '75000');

1 row created.

224G1A0573>insert into instructors values ('58583', 'Califieri', 'History', '62000');

1 row created.

224G1A0573>insert into instructors values ('76543', 'Singh', 'Finance', '80000');

1 row created.

224G1A0573>insert into instructors values ('76766', 'Crick', 'Biology', '72000');

1 row created.

224G1A0573>insert into instructors values ('83821', 'Brandt', 'Comp. Sci.', '92000');

1 row created.

224G1A0573>insert into instructors values ('98345', 'Kim', 'Elec. Eng.', '80000');

1 row created.
```

```
224G1A0573>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.
```

```
224G1A0573>insert into department values ('Biology', 'Watson', '90000');

1 row created.

224G1A0573>insert into department values ('Comp. Sci.', 'Taylor', '100000');

1 row created.

224G1A0573>insert into department values ('Elec. Eng.', 'Taylor', '85000');

1 row created.

224G1A0573>insert into department values ('Finance', 'Painter', '120000');

1 row created.

224G1A0573>insert into department values ('History', 'Painter', '50000');

1 row created.

224G1A0573>insert into department values ('Music', 'Packard', '80000');

1 row created.

224G1A0573>insert into department values ('Physics', 'Watson', '70000');

1 row created.
```

EXPERIMENT -16

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;

CREATE TABLE:

```
224g1a0573>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.
```

INSERTNG VALUES

```
224g1a0573>INSERT INTO customers VALUES(1,'Ramesh',23,'Allabad',25000);

1 row created.

224g1a0573>INSERT INTO customers VALUES(2, 'Suresh',22,'Kanpur',27000);

1 row created.

224g1a0573>INSERT INTO customers VALUES(3, 'Mahesh',24,'Ghaziabad',29000);

1 row created.

224g1a0573>INSERT INTO customers VALUES(4, 'chandhan',25,'Noida',31000);

1 row created.

224g1a0573>INSERT INTO customers VALUES(5, 'Alex', 21, 'paris',33000);

1 row created.

224g1a0573>INSERT INTO customers VALUES(6, 'Sunita',20,'delhi',35000);

1 row created.
```

```
224g1a0573> 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.
```