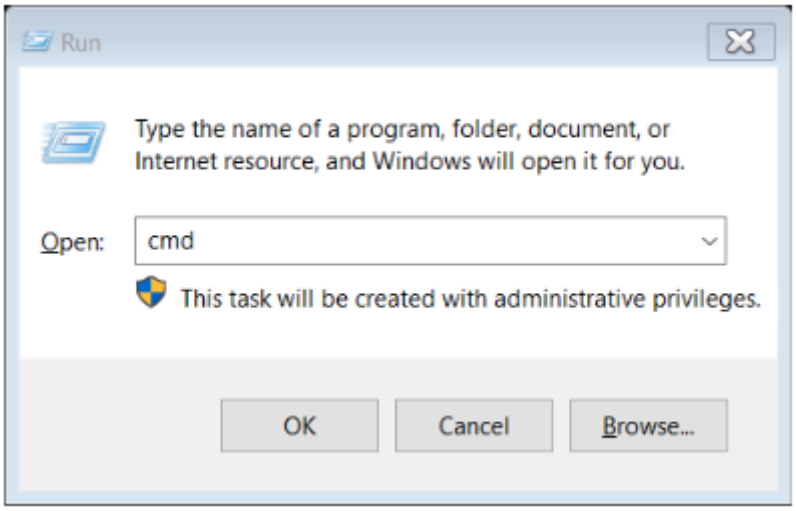
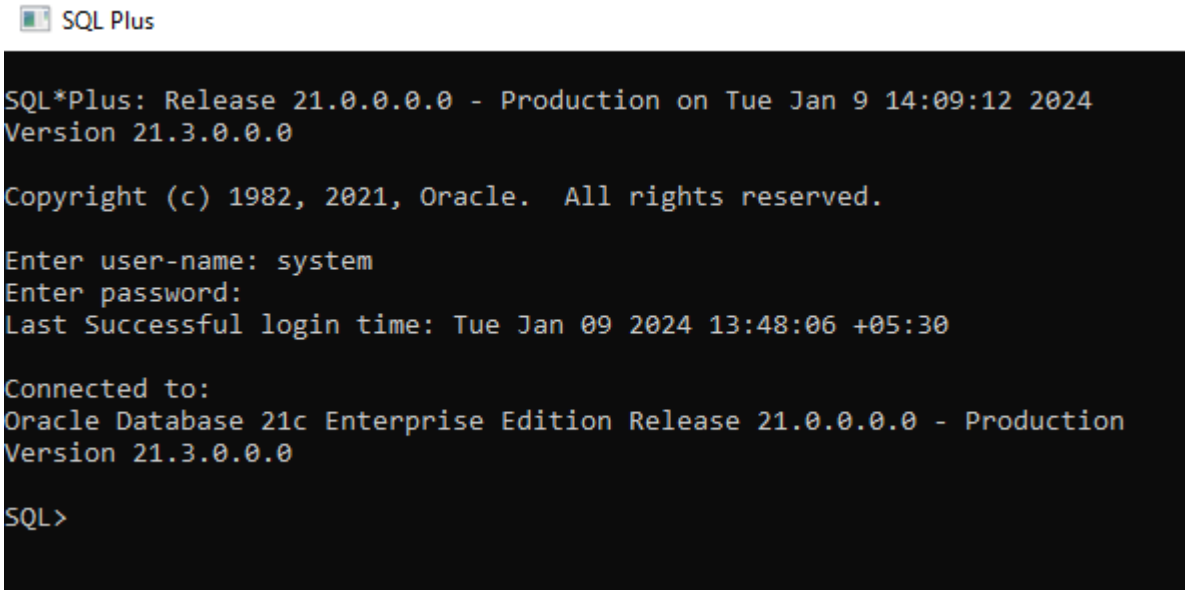
M YAMUNASREE **14-Sep-23**

Design of Databases using DDL Language

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



2. Once sqlplus prompt open then enter username and password it will connected



EXPERIMENT-1

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

Create table

```
SQL> CREATE TABLE persons(
  2  person_id NUMBER GENERATED BY DEFAULT AS IDENTITY,
  3  first_name VARCHAR2(50) NOT NULL,
  4  last_name VARCHAR2(50) NOT NULL,
  5  PRIMARY KEY(person_id)
  6  );

Table created.
```

Desc table

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

```
SQL> CREATE TABLE purchase_order_item(
  2  po_nr NUMBER NOT NULL,
  3  item_nr NUMBER NOT NULL,
  4  product_id NUMBER NOT NULL,
  5  quantity NUMBER NOT NULL,
  6  purchase_unit NUMBER NOT NULL,
  7  buy_price NUMBER(9,2) NOT NULL,
  8  delivery_date DATE,
  9  PRIMARY KEY(po_nr,item_nr)
 10  );

Table created.
```

Alter table

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

Table altered.
```

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

```
SQL> ALTER TABLE persons
  2  ADD(
  3  phone VARCHAR(20),
  4  email VARCHAR(100)
  5  );

Table altered.
```

```
SQL> DESC persons;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 PERSON_ID                                 NOT NULL NUMBER
 FIRST_NAME                                NOT NULL VARCHAR2(50)
 LAST_NAME                                 NOT NULL VARCHAR2(50)
 BIRTHDATE                                 NOT NULL DATE
 PHONE                                              VARCHAR2(20)
 EMAIL                                              VARCHAR2(100)
```

```
SQL> CREATE TABLE persons(
  2  person_id NUMBER GENERATED BY DEFAULT AS IDENTITY,
  3  first_name VARCHAR2(50) NOT NULL,
  4  last_name VARCHAR2(50) NOT NULL,
  5  PRIMARY KEY(person_id)
  6  );

Table created.
```

Drop table

```
SQL> DROP TABLE persons;

Table dropped.

SQL> DESC persons;
ERROR:
ORA-04043: object persons does not exist
```

```
SQL> INSERT INTO person(person_id,person_name)
  2  VALUES(1,'mani');

1 row created.

SQL> INSERT INTO person(person_id,person_name)
  2  VALUES(2,'chandu');

1 row created.
```

```
SQL> CREATE TABLE person(
  2  person_id NUMBER,
  3  person_name VARCHAR2(50) NOT NULL
  4  );

Table created.
```

```
SQL> DESC person;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 PERSON_ID                                          NUMBER
 PERSON_NAME                               NOT NULL VARCHAR2(50)
```

Truncate table

```
SQL> TRUNCATE TABLE person;

Table truncated.
```

```
SQL> SELECT * FROM person;

no rows selected
```

EXPERIMENT-2

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

```
SQL> CREATE TABLE employee(
  2  employee_id NUMBER NOT NULL,
  3  name VARCHAR(50) NOT NULL,
  4  dept_name VARCHAR(50) NOT NULL
  5  );

Table created.
```

SELECT COMMAND

```
SQL> SELECT employee_id,name FROM employee;

EMPLOYEE_ID NAME
----------- --------------------------------------------------
          1 mani
          2 deepu
          3 chandu
          4 gnani
```

INSERT COMMAND

```
SQL> INSERT INTO employee(employee_id,name,dept_name)
  2  VALUES(1,'mani','cse');

1 row created.

SQL> INSERT INTO employee(employee_id,name,dept_name)
  2  VALUES(2,'deepu','eee');

1 row created.

SQL> INSERT INTO employee(employee_id,name,dept_name)
  2  VALUES(3,'chandu','ece');

1 row created.

SQL> INSERT INTO employee(employee_id,name,dept_name)
  2  VALUES(4,'gnani','mech');

1 row created.
```

DELETE COMMAND

```
SQL> DELETE FROM employee WHERE dept_name='cse';

1 row deleted.

SQL> SELECT * FROM employee;

EMPLOYEE_ID NAME
----------- --------------------------------------------------
DEPT_NAME
--------------------------------------------------
          2 deepu
eee

          3 chandu
ece

          4 gnani
mech
```

UPDATE COMMAND

```
SQL> UPDATE employee
  2  SET name='gnanadeep'
  3  WHERE employee_id='4';

1 row updated.

SQL> SELECT * FROM employee;

EMPLOYEE_ID NAME
----------- --------------------------------------------------
DEPT_NAME
--------------------------------------------------
          1 mani
cse

          2 deepu
eee

          3 chandu
ece


EMPLOYEE_ID NAME
----------- --------------------------------------------------
DEPT_NAME
--------------------------------------------------
          4 gnanadeep
mech
```

EXPERIMENT-3

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

```
SQL> CREATE TABLE students (
  2  student_id INT PRIMARY KEY,
  3  first_name VARCHAR(50),
  4  last_name VARCHAR(50),
  5  age INT,
  6  grade VARCHAR(10)
  7  );

Table created.
```

```
SQL> INSERT INTO students(student_id,first_name,last_name,age,grade)
  2  VALUES(1,'mani','chandrika',20,'A');

1 row created.
```

```
SQL> INSERT INTO students(student_id,first_name,last_name,age,grade)
  2  VALUES(2,'gnana','deep',21,'B');

1 row created.
```

```
SQL> INSERT INTO students(student_id,first_name,last_name,age,grade)
  2  VALUES(3,'madhu','sudhan',22,'A');

1 row created.
```

CREATE VIEW

```
SQL> CREATE VIEW view_high_rankers AS
  2  SELECT *
  3  FROM students
  4  WHERE grade='A';

View created.
```

```
SQL> SELECT * FROM students;

STUDENT_ID FIRST_NAME
---------- --------------------------------------------------
LAST_NAME                                            AGE GRADE
-------------------------------------------------- ---------- ----------
         1 mani
chandrika                                             20 A

         2 gnana
deep                                                  21 B

         3 madhu
sudhan                                                22 A
```

```
SQL> SELECT student_id,first_name FROM view_high_rankers;

STUDENT_ID FIRST_NAME
---------- --------------------------------------------------
         1 mani
         3 madhu
```

```
SQL> CREATE VIEW view_young_students AS
  2  SELECT *
  3  FROM students
  4  WHERE age < 21;

View created.
```

```
SQL> SELECT student_id,first_name,age FROM view_young_students;

STUDENT_ID FIRST_NAME                                          AGE
---------- -------------------------------------------------- ----------
         1 mani                                                20
```

```
SQL> CREATE VIEW view_s_lastname_students AS
  2  SELECT *
  3  FROM students
  4  WHERE last_name LIKE 'm%';

View created.
```

```
SQL> CREATE VIEW view_m_firstname_students AS
  2  SELECT *
  3  FROM students
  4  WHERE first_name LIKE 'm%';

View created.
```

```
SQL> SELECT student_id,first_name,age FROM view_m_firstname_students;

STUDENT_ID FIRST_NAME                                          AGE
---------- -------------------------------------------------- ----------
         1 mani                                                20
         3 madhu                                               22
```

INSERT VIEW

```
SQL> INSERT INTO view_m_firstname_students VALUES(5,'mani','deep',25,'E');

1 row created.

SQL> SELECT student_id,first_name,age FROM view_m_firstname_students;

STUDENT_ID FIRST_NAME                                          AGE
---------- -------------------------------------------------- ----------
         1 mani                                                26
         3 madhu                                               22
         5 mani                                                25
```

UPDATE VIEW

```
SQL> UPDATE view_m_firstname_students
  2  SET age=26
  3  WHERE student_id=1;

1 row updated.

SQL> SELECT student_id,first_name,age FROM view_m_firstname_students;

STUDENT_ID FIRST_NAME                                          AGE
---------- -------------------------------------------------- ----------
         1 mani                                                26
         3 madhu                                               22
```

DELETE VIEW

```
SQL> DELETE FROM view_m_firstname_students WHERE student_id=3;

1 row deleted.

SQL> SELECT student_id,first_name,age FROM view_m_firstname_students;

STUDENT_ID FIRST_NAME                                          AGE
---------- -------------------------------------------------- ----------
         1 mani                                                26
         5 mani                                                25
```

EXPERIMENT-4

4. Write SQL queries to perform RELATIONAL SET OPERATIONS (i.e. UNION, UNION ALL,

INTERSECT, MINUS, CROSS JOIN, NATURAL JOIN).

```
SQL> CREATE TABLE emp1(
  2  eid int,
  3  ename VARCHAR(20),
  4  eplace VARCHAR(20)
  5  );

Table created.
```

```
SQL> INSERT INTO emp1
  2  VALUES('11','mani','DMM');

1 row created.
```

```
SQL> INSERT INTO emp1
  2  VALUES('12','jyothi','PKD');

1 row created.
```

```
SQL> INSERT INTO emp1
  2  VALUES('13','ayesha','ATP');

1 row created.
```

```
SQL> CREATE TABLE emp2(
  2  eid int,
  3  ename VARCHAR(20),
  4  eplace VARCHAR(20)
  5  );

Table created.
```

```
SQL> INSERT INTO emp2
  2  VALUES('24','gowri','bngr');

1 row created.
```

```
SQL> INSERT INTO emp2
  2  VALUES('25','chandu','chennai');

1 row created.
```

```
SQL> INSERT INTO emp2
  2  VALUES('26','deep','tirupati');

1 row created.
```

```
SQL> INSERT INTO emp2
  2  VALUES('27','gnana','pune');

1 row created.
```

```
SQL> SELECT * FROM emp1;

       EID ENAME                EPLACE
---------- -------------------- --------------------
        11 mani                 DMM
        12 jyothi               PKD
        13 ayesha               ATP
```

```
SQL> SELECT * FROM emp2;

       EID ENAME                EPLACE
---------- -------------------- --------------------
        24 gowri                bngr
        25 chandu               chennai
        26 deep                 tirupati
        27 gnana                pune
```

UNION

```
SQL> SELECT * FROM emp1 UNION SELECT * FROM emp2;

       EID ENAME                EPLACE
---------- -------------------- --------------------
        11 mani                 DMM
        12 jyothi               PKD
        13 ayesha               ATP
        24 gowri                bngr
        25 chandu               chennai
        26 deep                 tirupati
        27 gnana                pune

7 rows selected.
```

```
SQL> SELECT * FROM emp1 MINUS SELECT * FROM emp2;

       EID ENAME                EPLACE
---------- -------------------- --------------------
        11 mani                 DMM
        12 jyothi               PKD
        13 ayesha               ATP
```

MINUS

UNION ALL

```
SQL> SELECT * FROM emp1 UNION ALL SELECT * FROM emp2;

       EID ENAME                EPLACE
---------- -------------------- --------------------
        11 mani                 DMM
        12 jyothi               PKD
        13 ayesha               ATP
        24 gowri                bngr
        25 chandu               chennai
        26 deep                 tirupati
        27 gnana                pune

7 rows selected.
```

NATURAL JOIN

```
SQL> SELECT * FROM emp1 NATURAL JOIN emp2;

no rows selected
```

INTERSECT

```
SQL> SELECT * FROM emp1 INTERSECT SELECT * FROM emp2;

no rows selected
```

CROSS JOIN

```
SQL> SELECT * FROM emp1 CROSS JOIN emp2;

       EID ENAME                EPLACE                    EID
---------- -------------------- -------------------- ----------
ENAME                EPLACE
-------------------- --------------------
        11 mani                 DMM                        24
gowri                bngr

        11 mani                 DMM                        25
chandu               chennai

        11 mani                 DMM                        26
deep                 tirupati


       EID ENAME                EPLACE                    EID
---------- -------------------- -------------------- ----------
ENAME                EPLACE
-------------------- --------------------
        11 mani                 DMM                        27
gnana                pune

        12 jyothi               PKD                        24
gowri                bngr

        12 jyothi               PKD                        25
chandu               chennai


       EID ENAME                EPLACE                    EID
---------- -------------------- -------------------- ----------
ENAME                EPLACE
-------------------- --------------------
        12 jyothi               PKD                        26
deep                 tirupati

        12 jyothi               PKD                        27
```

```
SQL> SELECT * FROM emp1 CROSS JOIN emp2;

       EID ENAME                EPLACE                    EID
---------- -------------------- -------------------- ----------
ENAME                EPLACE
-------------------- --------------------
        11 mani                 DMM                        24
gowri                bngr

        11 mani                 DMM                        25
chandu               chennai

        11 mani                 DMM                        26
deep                 tirupati


       EID ENAME                EPLACE                    EID
---------- -------------------- -------------------- ----------
ENAME                EPLACE
-------------------- --------------------
        11 mani                 DMM                        27
gnana                pune

        12 jyothi               PKD                        24
gowri                bngr

        12 jyothi               PKD                        25
chandu               chennai


       EID ENAME                EPLACE                    EID
---------- -------------------- -------------------- ----------
ENAME                EPLACE
-------------------- --------------------
        12 jyothi               PKD                        26
deep                 tirupati

        12 jyothi               PKD                        27
gnana                pune

        13 ayesha               ATP                        24
gowri                bngr


       EID ENAME                EPLACE                    EID
---------- -------------------- -------------------- ----------
ENAME                EPLACE
-------------------- --------------------
        13 ayesha               ATP                        25
chandu               chennai

        13 ayesha               ATP                        26
deep                 tirupati

        13 ayesha               ATP                        27
gnana                pune

12 rows selected.
```

```
SQL> SELECT * FROM emp1 NATURAL JOIN emp2;

no rows selected
```

EXPERIMENT-5

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

MAX).

```
SQL> CREATE TABLE instructor (
  2  ID NUMBER(10) PRIMARY KEY,
  3  Name VARCHAR2(25) NOT NULL,
  4  dept_name VARCHAR2(10) NOT NULL,
  5  salary NUMBER(10,0)
  6  );

Table created.
```

```
SQL> INSERT INTO instructor VALUES(1,'Mani','CSE',30000);

1 row created.

SQL> INSERT INTO instructor VALUES(2,'Deep','CSE',40000);

1 row created.

SQL> INSERT INTO instructor VALUES(3,'Mouni','CSM',50000);

1 row created.

SQL> INSERT INTO instructor VALUES(4,'Dhanu','CSM',60000);

1 row created.

SQL> INSERT INTO instructor VALUES(5,'Madhu','CSD',70000);

1 row created.
```

```
SQL> SELECT * FROM instructor;

        ID NAME                      DEPT_NAME      SALARY
---------- ------------------------- ---------- ----------
         1 Mani                      CSE             30000
         2 Deep                      CSE             40000
         3 Mouni                     CSM             50000
         4 Dhanu                     CSM             60000
         5 Madhu                     CSD             70000
```

```
SQL> SELECT dept_name,avg(salary)
  2  FROM instructor
  3  GROUP BY dept_name;

DEPT_NAME  AVG(SALARY)
---------- -----------
CSE              35000
CSM              55000
CSD              70000
```

SUM

```
SQL> SELECT SUM(salary)
  2  FROM instructor
  3  WHERE dept_name = 'CSE';

SUM(SALARY)
-----------
      70000
```

AVG

```
SQL> SELECT Avg(salary)
  2  FROM instructor
  3  WHERE dept_name = 'CSE';

AVG(SALARY)
-----------
      35000
```

COUNT

```
SQL> SELECT count(salary)
  2  FROM instructor
  3  WHERE dept_name = 'CSE';

COUNT(SALARY)
-------------
            2
```

MIN

```
SQL> SELECT Min(salary)
  2  FROM instructor
  3  WHERE dept_name = 'CSE';

MIN(SALARY)
-----------
      30000
```

EXPERIMENT-6

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

```
SQL> CREATE TABLE library (
  2  Rollno NUMBER,
  3  Book VARCHAR(10)
  4  );

Table created.
```

```
SQL> INSERT INTO library VALUES(11,'DBMS');

1 row created.

SQL> INSERT INTO library VALUES(12,'JAVA');

1 row created.

SQL> INSERT INTO library VALUES(13,'SE');

1 row created.

SQL> INSERT INTO library VALUES(14,'PYTHON');

1 row created.

SQL> INSERT INTO library VALUES(15,'ORACLE');

1 row created.
```

```
SQL> SELECT * FROM library;

    ROLLNO BOOK
---------- ----------
        11 DBMS
        12 JAVA
        13 SE
        14 PYTHON
        15 ORACLE
```

INNER JOIN

```
SQL> SELECT *
  2  FROM student
  3  INNER JOIN library ON student.branch = 'CSE'
  4  AND library.title = 'DBMS';

    ROLLNO NAME
---------- --------------------------------------------------
BRANCH                                                ROLLNO
-------------------------------------------------- ----------
TITLE
--------------------------------------------------
         1 John
CSE                                                        1
DBMS
```

```
SQL> SELECT *
  2  FROM student
  3  INNER JOIN library ON student.Rollno = library.Rollno;

    ROLLNO NAME
---------- --------------------------------------------------
BRANCH                                                ROLLNO
-------------------------------------------------- ----------
TITLE
--------------------------------------------------
         1 John
CSE                                                        1
DBMS

         2 Jane
EEE                                                        2
JAVA
```

LEFT OUTER JOIN

```
SQL> SELECT *
  2  FROM student
  3  LEFT OUTER JOIN library ON student.Rollno = library.Rollno;

    ROLLNO NAME
---------- --------------------------------------------------
BRANCH                                                 ROLLNO
-------------------------------------------------- ----------
TITLE
--------------------------------------------------

         1 John
CSE                                                         1
DBMS

         2 Jane
EEE                                                         2
JAVA

    ROLLNO NAME
---------- --------------------------------------------------
BRANCH                                                 ROLLNO
-------------------------------------------------- ----------
TITLE
--------------------------------------------------

         3 Bob
ECE
```

RIGHT OUTER JOIN

```
SQL> SELECT *
  2  FROM student
  3  RIGHT OUTER JOIN library ON student.Rollno = library.Rollno;

    ROLLNO NAME
---------- --------------------------------------------------
BRANCH                                                 ROLLNO
-------------------------------------------------- ----------
TITLE
--------------------------------------------------
         1 John
CSE                                                         1
DBMS

         2 Jane
EEE                                                         2
JAVA

    ROLLNO NAME
---------- --------------------------------------------------
BRANCH                                                 ROLLNO
-------------------------------------------------- ----------
TITLE
--------------------------------------------------



                                                            4
SE
```

FULL OUTER JOIN

```
SQL> SELECT *
  2  FROM student
  3  FULL OUTER JOIN library ON student.Rollno = library.Rollno;

    ROLLNO NAME
---------- --------------------------------------------------
BRANCH                                                ROLLNO
-------------------------------------------------- ----------
TITLE
--------------------------------------------------
         1 John
CSE                                                        1
DBMS

         2 Jane
EEE                                                        2
JAVA

    ROLLNO NAME
---------- --------------------------------------------------
BRANCH                                                ROLLNO
-------------------------------------------------- ----------
TITLE
--------------------------------------------------



                                                           4
SE

         3 Bob
ECE
```

NATURAL JOIN

```
SQL> SELECT *
  2  FROM student
  3  NATURAL JOIN library;

    ROLLNO NAME
---------- --------------------------------------------------
BRANCH
--------------------------------------------------
TITLE
--------------------------------------------------
         1 John
CSE
DBMS

         2 Jane
EEE
JAVA
```

EXPERIMENT-7

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

```
SQL> connect sys as sysdba
Enter password:
Connected.
SQL> CREATE TABLE student (
  2  StudentID INT PRIMARY KEY,
  3  FirstName VARCHAR(50),
  4  LastName VARCHAR(50),
  5  Age INT,
  6  Grade FLOAT
  7  );

Table created.
```

```
SQL> INSERT INTO Student VALUES(1,'Mani','Chandrika',20,50);

1 row created.

SQL> INSERT INTO Student VALUES(2,'Gnana','Deep',21,60);

1 row created.

SQL> INSERT INTO Student VALUES(3,'Madhu','Sudhan',22,70);

1 row created.

SQL> INSERT INTO Student VALUES(4,'Chandra','Kala',18,NULL);

1 row created.
```

```
SQL> SELECT * FROM Student WHERE Age BETWEEN 20 AND 25;

 STUDENTID FIRSTNAME
---------- --------------------------------------------------
LASTNAME                                                AGE      GRADE
-------------------------------------------------- ---------- ----------
         1 Mani
Chandrika                                               20         50

         2 Gnana
Deep                                                    21         60

         3 Madhu
Sudhan                                                  22         70
```

BETWEEN

IS NULL

```
SQL> SELECT * FROM Student WHERE Grade is NULL;

 STUDENTID FIRSTNAME
---------- --------------------------------------------------
LASTNAME                                                 AGE      GRADE
-------------------------------------------------- ---------- ----------
         4 Chandra
Kala                                                      18
```

EXISTS

```
SQL> SELECT * FROM student
  2  WHERE EXISTS
  3  (SELECT StudentID, FirstName FROM student WHERE Grade IS NULL);
```

```
 STUDENTID FIRSTNAME
---------- --------------------------------------------------
LASTNAME                                                 AGE      GRADE
-------------------------------------------------- ---------- ----------
         4 Chandra
Kala                                                      18
```

LIKE

```
SQL> SELECT StudentID, FirstName FROM Student WHERE FirstName LIKE 'M%';

 STUDENTID FIRSTNAME
---------- --------------------------------------------------
         1 Mani
         3 Madhu
```

IN

```
SQL> SELECT StudentID, FirstName, Age FROM Student WHERE Age IN(20,21,22);

 STUDENTID FIRSTNAME                                              AGE
---------- -------------------------------------------------- ----------
         1 Mani                                                       20
         2 Gnana                                                      21
         3 Madhu                                                      22
```

EXPERIMENT-8

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

UPPER

```
SQL> SELECT UPPER('hello world') FROM dual;

UPPER('HELL
-----------
HELLO WORLD
```

LOWER

```
SQL> SELECT LOWER('HELLO WORLD') FROM dual;

LOWER('HELL
-----------
hello world
```

INITCAP

```
SQL> SELECT INITCAP('hello world') FROM dual;

INITCAP('HE
-----------
Hello World
```

LENGT

```
SQL> SELECT LENGTH('MANI CHANRIKA') FROM dual;

LENGTH('MANICHANRIKA')
----------------------
                    13
```

SUBSTR

```
SQL> SELECT SUBSTR('MANI CHANDRIKA',3,7) FROM dual;

SUBSTR(
-------
NI CHAN
```

REPLACE

```
SQL> SELECT REPLACE('MANI CHANDRIKA','CHANDRIKA','CHANDANA') FROM dual;

REPLACE('MANI
-------------
MANI CHANDANA
```

INSTR

```
SQL> SELECT INSTR('MANI CHANDRIKA','CHANDRIKA') FROM dual;

INSTR('MANICHANDRIKA','CHANDRIKA')
----------------------------------
                                 6
```

LPAD

```
SQL> SELECT LPAD('MANI CHANDRIKA',20,'*') FROM dual;

LPAD('MANICHANDRIKA'
--------------------
******MANI CHANDRIKA
```

RPAD

```
SQL> SELECT RPAD('MANI CHANDRIKA',20,'*') FROM dual;

RPAD('MANICHANDRIKA'
--------------------
MANI CHANDRIKA******
```

TRIM

```
SQL> SELECT TRIM('         MANI CHANDRIKA       ') FROM dual;

TRIM('MANICHAN
--------------
MANI CHANDRIKA
```

LTRIM

```
SQL> SELECT LTRIM('         MANI CHANDRIKA       ') FROM dual;

LTRIM('MANICHANDRIKA'
---------------------
MANI CHANDRIKA
```

RTRIM

```
SQL> SELECT RTRIM('         MANI CHANDRIKA       ') FROM dual;

RTRIM('MANICHANDRIKA')
----------------------
         MANI CHANDRIKA
```

ROUND

```
SQL> SELECT ROUND(15.789,1) FROM DUAL;

ROUND(15.789,1)
---------------
           15.8
```

MOD

```
SQL> SELECT MOD(1600,100) FROM DUAL;

MOD(1600,100)
-------------
            0
```

TRUNC

```
SQL> SELECT TRUNC(45.426,1) FROM DUAL;

TRUNC(45.426,1)
---------------
           45.4
```

DATE

```
SQL> SELECT SYSDATE FROM DUAL;

SYSDATE
---------
10-DEC-23
```

```
SQL> SELECT MONTHS_BETWEEN(SYSDATE,'10-DEC-23') FROM DUAL;

MONTHS_BETWEEN(SYSDATE,'10-DEC-23')
-----------------------------------
                                  0
```

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

ADD_MONTH
---------
10-FEB-24
```

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

NEXT_DAY(
---------
14-DEC-23
```

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

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

```
SQL> SELECT TRUNC('25-JUL-03','YEAR')
  2  FROM DUAL;
```

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

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

EXPERIMENT-9

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

PRIMARY KEY

```
SQL> CREATE TABLE students(
  2  StudentID INT PRIMARY KEY,
  3  FirstName VARCHAR(50),
  4  LastName VARCHAR(50)
  5  );

Table created.
```

```
SQL> CREATE TABLE Courses (
  2  CourseID INT PRIMARY KEY,
  3  CourseName VARCHAR(50)
  4  );

Table created.
```

FOREIGN KEY

```
SQL> CREATE TABLE Enrollments (
  2  EnrollmentID INT PRIMARY KEY,
  3  StudentID INT,
  4  CourseID INT,
  5  FOREIGN KEY(StudentID) REFERENCES Students(StudentID),
  6  FOREIGN KEY(CourseID) REFERENCES Courses(CourseID)
  7  );

Table created.
```

UNIQUE KEY

```
SQL> CREATE TABLE Employees (
  2  EmployeeID INT PRIMARY  KEY,
  3  EmployeeName VARCHAR(50),
  4  Email VARCHAR(50) UNIQUE
  5  );

Table created.
```

NOT NULL KEY

```
SQL> CREATE TABLE prders (
  2  orderID INT PRIMARY KEY,
  3  productName VARCHAR(50) NOT NULL,
  4  quantity INT
  5  );

Table created.
```

CHECK

```
SQL> CREATE TABLE products (
  2  productID INT PRIMARY KEY,
  3  productName VARCHAR(50),
  4  price DECIMAL(10,2) CHECK(price>0)
  5  );

Table created.
```

DEFAULT

```
SQL> CREATE TABLE customers (
  2  customerID INT PRIMARY KEY,
  3  customerName VARCHAR(50),
  4  country VARCHAR(50) DEFAULT 'US'
  5  );

Table created.
```

EXPERIMENT-10

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

```
SQL> 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.

SQL> SET SERVEROUT ON
SQL> /
3628800

PL/SQL procedure successfully completed.
```

EXPERIMENT-11

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

```
SQL> 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.

```
SQL> 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.

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

Procedure created.
```

```
SQL> DECLARE
  2   CNT NUMBER;
  3   BEGIN
  4  INSERTUSER(101,'NARASIMHA');
  5   SELECT COUNT(*) INTO CNT FROM SAILOR;
  6   DBMS_OUTPUT.PUT_LINE(CNT||' RECORD IS INSERTED SUCCESSFULLY');
  7   END;
  8  /
RECORD INSERTED SUCCESSFULLY
3 RECORD IS INSERTED SUCCESSFULLY

PL/SQL procedure successfully completed.
```

```
SQL> DROP PROCEDURE insertuser;

Procedure dropped.
```

EXPERIMENT-14

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

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

Function created.

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

PL/SQL procedure successfully completed.
```

Recursive Fuction

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

Function created.

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

PL/SQL procedure successfully completed.

SQL> DROP FUNCTION fact;

Function dropped.

SQL>
```

EXPERIMENT-15

15. Write PL/SQL program to implement Trigger on table.

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

Table created.

SQL> 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  FOREIGN KEY (DEPT_NAME) REFERENCES DEPARTMENT(DEPT_NAME) ON DELETE SET NULL
  8  );

Table created.
```

```
SQL> 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.

SQL> 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  /

PL/SQL procedure successfully completed.
```

EXPERIMENT-16

16. Write PL/SQL program to implement Cursor on table.

1. implicit cursor

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

Table created.
```

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

1 row created.

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

1 row created.

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

1 row created.

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

1 row created.

SQL> INSERT INTO customers VALUES(4, 'chandhan',25,'Noida',31000);
INSERT INTO customers VALUES(4, 'chandhan',25,'Noida',31000)
*
ERROR at line 1:
ORA-00001: unique constraint (SYSTEM.SYS_C008321) violated


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

1 row created.

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

1 row created.
```

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

EXPLICIT CURSOR

```
PL/SQL procedure successfully completed.

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

PL/SQL procedure successfully completed.
```