

SUBMITTED BY SUBMITTED TO

Ms. Swati Sharma

**EXPERIMENT – 1**

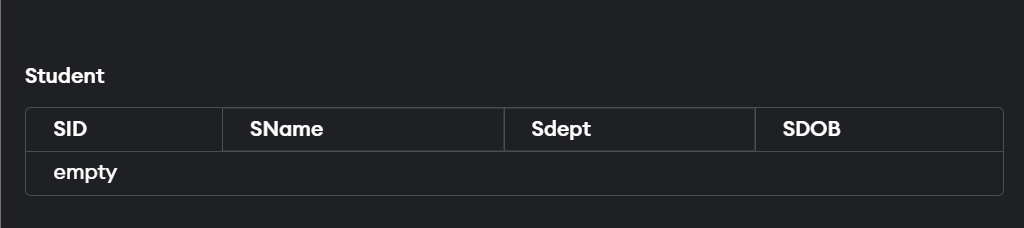
Title-implement data definition language

Objective-study of data definition language commands

:create table , alter table , drop table , rename table .

Code-

Create table Student (SID int ,SName char , Sdept char , SDOB date );



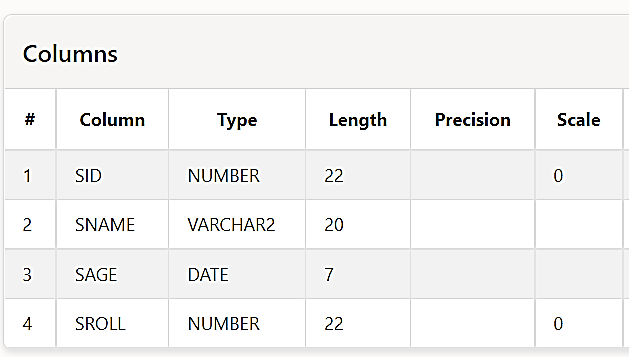
Alter table –

ALTER TABLE Student ADD Sroll int;

ALTER TABLE Student MODIFY SName varchar (20);

ALTER TABLE Student DROP COLUMN Sdept;

ALTER TABLE Student RENAME COLUMN SDOB TO Sage;



**EXPERIMENT - 2**

**Title** Data Manipulation Language Statements

**Objective** Study of Data Manipulation Statements.

**Code –**

Create Table Emp1(EID int, EName varchar(20), Edept varchar(20),

Salary int);

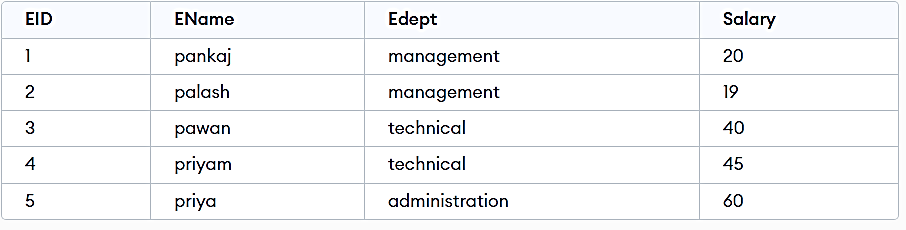
INSERT INTO Emp1 VALUES(1,'pankaj','management',20);

INSERT INTO Emp1 values(2,'palash','management',19);

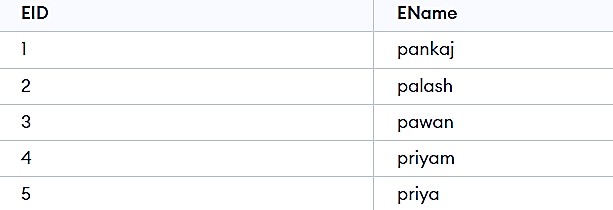
INSERT INTO Emp1 values(3,'pawan','technical',40);

INSERT INTO Emp1 values(4,'priyam','technical',45);

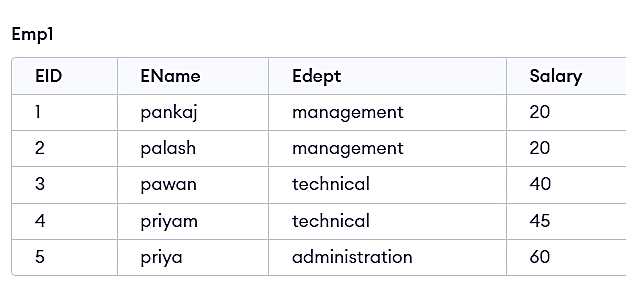
INSERT INTO Emp1 values(5,'priya','administration',60)



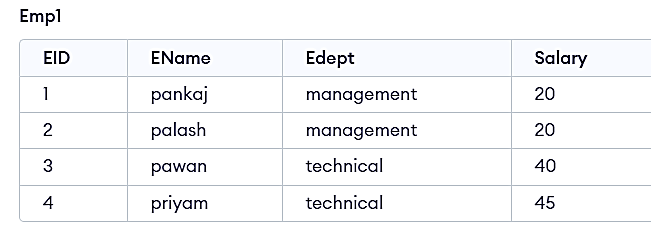
select "EID", "ENAME" from "EMP1";



UPDATE Emp1 SET Salary = 20 WHERE Edept='management' ;



DELETE FROM Emp1 WHERE Edept='administration'



**EXPERIMENT - 3**

Title- Implement various type of Integrity Constraints on database.

Objective- Study of various type of Integrity Constraints.

**Code-**

CREATE TABLE Persons

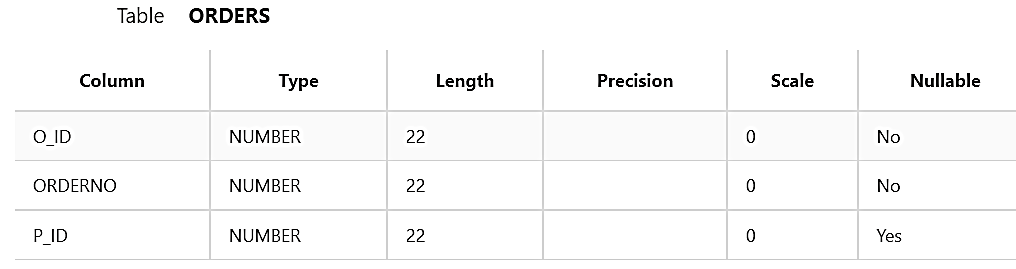
(P\_Id int NOT NULL,LastName varchar(255) NOT NULL,

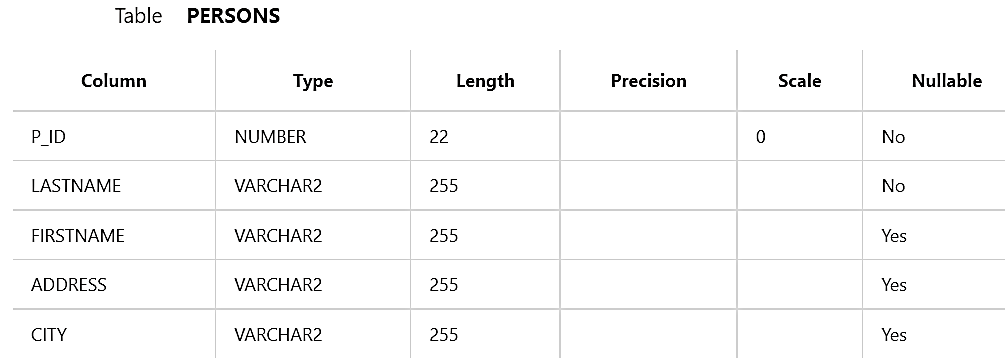
FirstName varchar(255),Address varchar(255),City varchar(255),PRIMARY KEY (P\_Id))

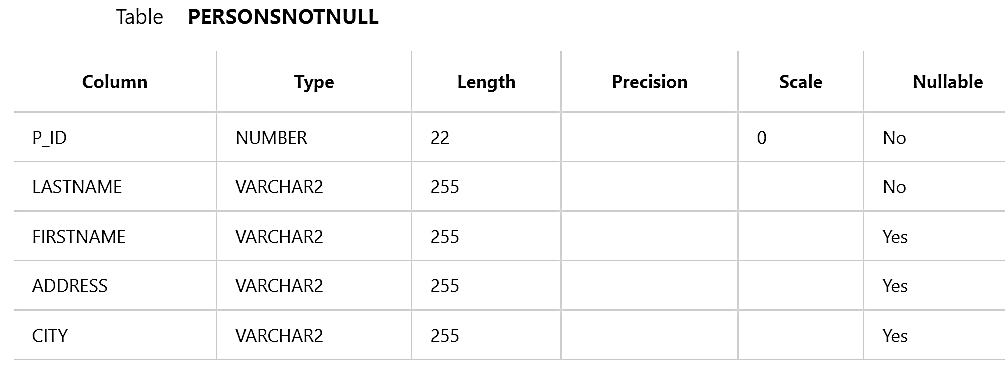
CREATE TABLE Orders

(O\_Id int NOT NULL,OrderNo int NOT NULL,P\_Id int,

PRIMARY KEY (O\_Id),FOREIGN KEY (P\_Id) REFERENCES Persons(P\_Id))







**EXPERIMENT - 3**

Title : SELECT Command

Objective : Study of SELECT command with different clauses.

Code :

Create Table Emp1(EID int, EName varchar(20), Edept varchar(20),

Salary int);

INSERT INTO Emp1 VALUES(1,'pankaj','management',20);

INSERT INTO Emp1 values(2,'palash','management',19);

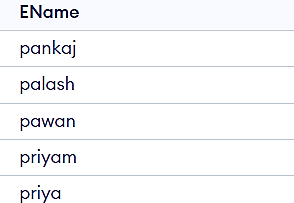
INSERT INTO Emp1 values(3,'pawan','technical',40);

INSERT INTO Emp1 values(4,'priyam','technical',45);

SELECT \*FROM Emp1 WHERE Salary=20



SELECT EName FROM Emp1



SELECT \* FROM Emp1 ORDER BY Salary desc



**EXPERIMENT - 4**

Title: SINGLE ROW functions and Group functions

Objective: Study of SINGLE ROW functions (Character, Numeric, Date functions) and GROUP functions (avg, count, max, min, sum).



**EXPERIMENT – 5**

**Title :-Implement various type of SET OPERATORS (Union, Intersect, Minus) and JOINS**.  
 Objective- Study of various type of SET OPERATORS (Union, Intersect, Minus) and Various type of JOINS.

Code-

create table t1(

id int , name varchar(255), department varchar(255) , salary int , year\_of\_experience int);

insert into t1 (id,name,department,salary,year\_of\_experience)

values

(1, 'Akash' , 'development' , 7200, 2), (2, 'abhishek', 'production' , 4500, 1), (3, 'pranav' , 'hr' , 5900, 3), (4, 'shubham' , 'accounts' , 5700, 2),( 5, 'sunil', 'development' , 8700, 3);

create table t2(

id int , name varchar(255), department varchar(255) , salary int , year\_of\_experience int);

insert into t2(id,name,department,salary,year\_of\_experience)

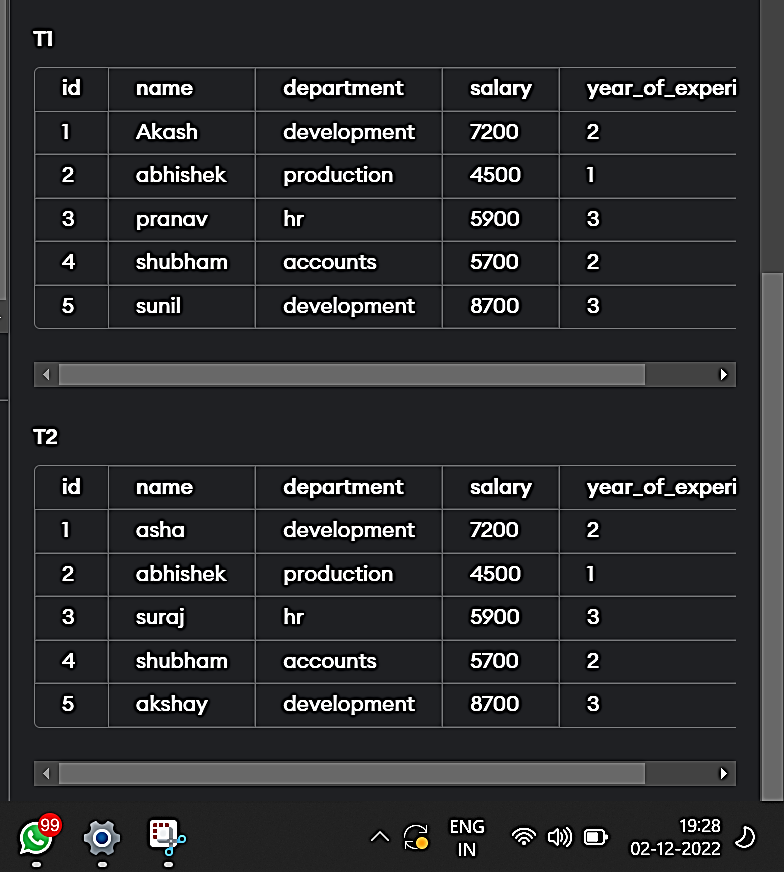
values

(1, 'asha' , 'development' , 7200, 2), (2, 'abhishek', 'production' , 4500, 1), (3, 'suraj' ,' hr' , 5900, 3), (4, 'shubham' , 'accounts' , 5700, 2),( 5, 'akshay', 'development' , 8700, 3);

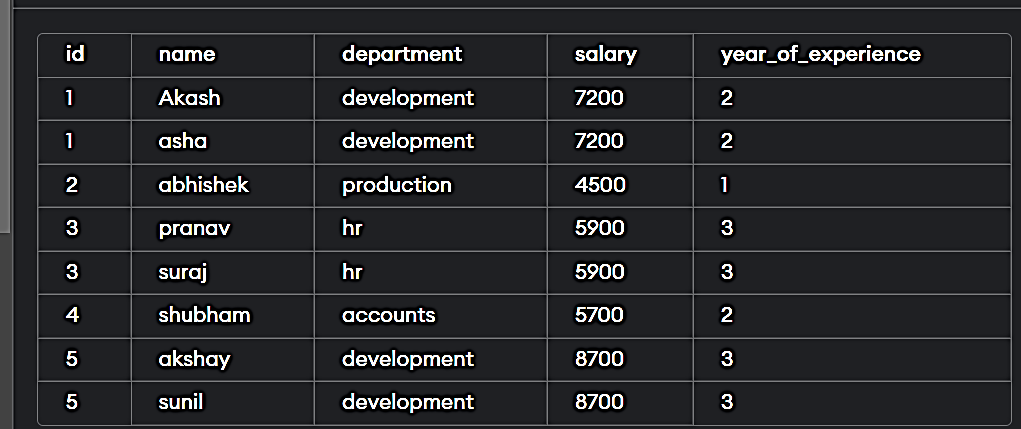
SELECT\*FROM t1 UNION SELECT\*FROM t2; // for union

SELECT \*FROM t1 INTERSECT SELECT \*FROM t2; // for intersection

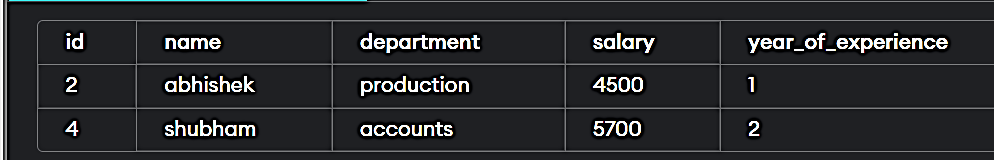
SELECT \*FROM t1 MINUS SELECT \*FROM t2; // for minus

Table creation-

For union-



For intersection-



**EXPERIMENT -6**

**Title** : Subqueries

**Objective**: Study and implement the concept of sub queries.

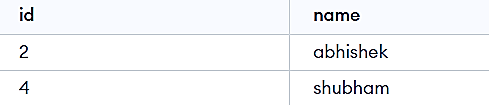
**UNION CLAUSE**

SELECT id,name FROM t1 UNION SELECT id,name FROM t2;



**INTERSECTION CLAUSE**

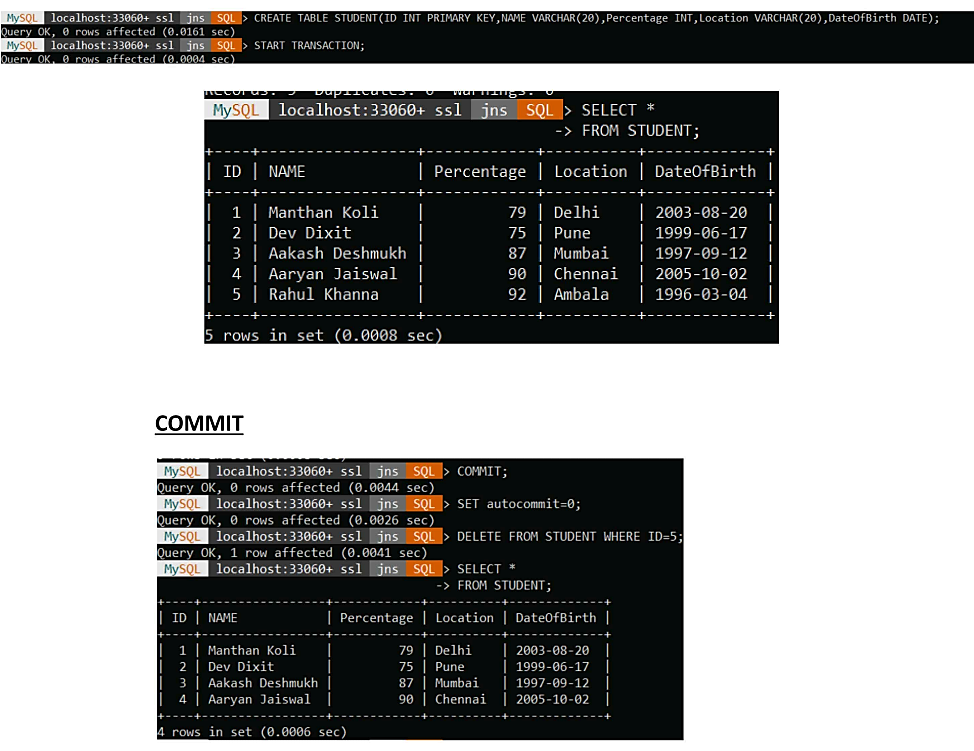
SELECT id,name FROM t1 INTERSECT SELECT id,name FROM t2;



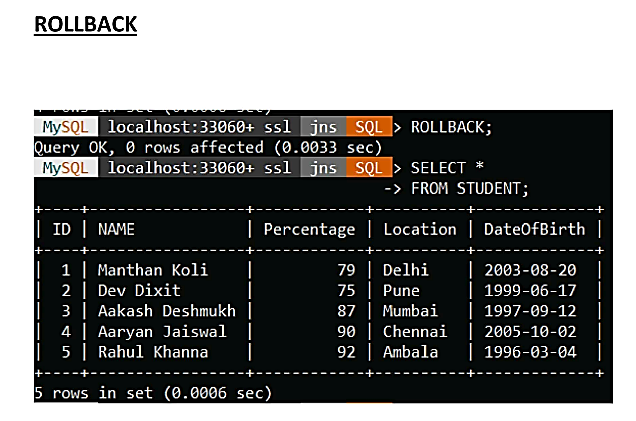
**EXPERIMENT -7**

**Title**:Control languages

**Objective:** Study and implement the concept of Data Control Language (DCL), Transaction Control Language (TCL).



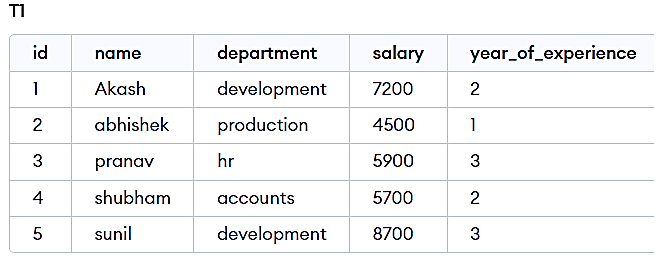
**ROLLBACK:-**



**EXPERIMENT -8**

**Title** :Views

**Objective**: Study of Simple and Complex View.



**CREATE VIEW**

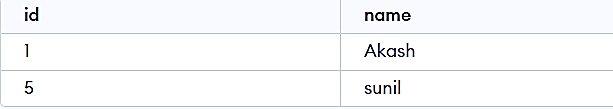
SELECT \* FROM T1;

CREATE VIEW [salary above 5900] AS

SELECT id,name FROM T1

WHERE salary>5900;

SELECT \* FROM [salary above 5900];



**DROP VIEW**

DROP VIEW[salary above 5900]



**EXPERIMENT -10**

Title –write a PL/SQL block to satisfy some conditions by accepting input from user .

Code-

create table GFG (id number(4), author varchar2(50),

likes number(4))

insert into GFG values(1, 'sam', 10);

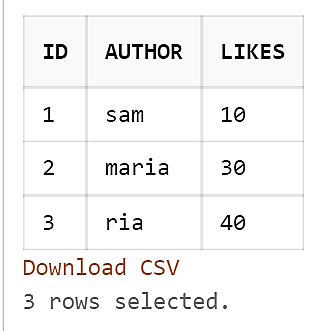
insert into GFG values(2, 'maria', 30);

insert into GFG values(3, 'ria', 40);

select \* from GFG;

select \* from GFG where id=&id;

output-



**EXPERIMENT -11**

**Title-** PL/SQL block for greatest of three numbers using IF AND ELSEIF

**Objective -**PL/SQL Control Structure provides conditional tests.

**Code:**

DECLARE

a NUMBER := 46;

b NUMBER := 67;

c NUMBER := 21;

BEGIN

IF a > b

AND a > c THEN

dbms\_output.Put\_line('Greatest number is '

||a);

ELSIF b > a

AND b > c THEN

dbms\_output.Put\_line('Greatest number is '

||b);

ELSE

--if c is greater then print c

dbms\_output.Put\_line('Greatest number is '

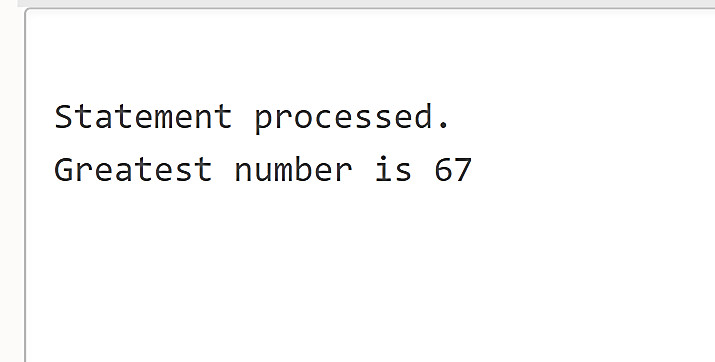
||c);

END IF;

--end if condition

END;

Output-



**EXPERIMENT -12**

**Title**- PL/SQL block for summation of odd numbers using for LOOP

**Objective**- PL/SQL Control Structure provides conditional tests, loops, flow control and branches that let to produce well-structured programs.

**Code**-

DECLARE

-- declare variable num

num NUMBER(3) := 1;

sum1 NUMBER(4) := 0;

BEGIN

WHILE num <= 100 LOOP

-- display odd number

dbms\_output.Put\_line(num);

-- the sum of all odd numbers

sum1 := sum1 + num;

--next odd number

num := num + 2;

-- end loop ,

END LOOP;

dbms\_output.Put\_line('Sum of all odd numbers is '|| sum1);

END;

**Output-**



**EXPERIMENT -13**

**Title** -write a PL/SQL command for GCD numbers.

**Objective-**PL/SQL control structure provides conditional tests .

**Code-** DECLARE

-- declare variable num1, num2 and t

-- and these three variables datatype are integer

num1 INTEGER;

num2 INTEGER;

t INTEGER;

BEGIN

num1 := 8;

num2 := 48;

WHILE MOD(num2, num1) != 0 LOOP

t := MOD(num2, num1);

num2 := num1;

num1 := t;

END LOOP;

dbms\_output.Put\_line('GCD of '

||num1

||' and '

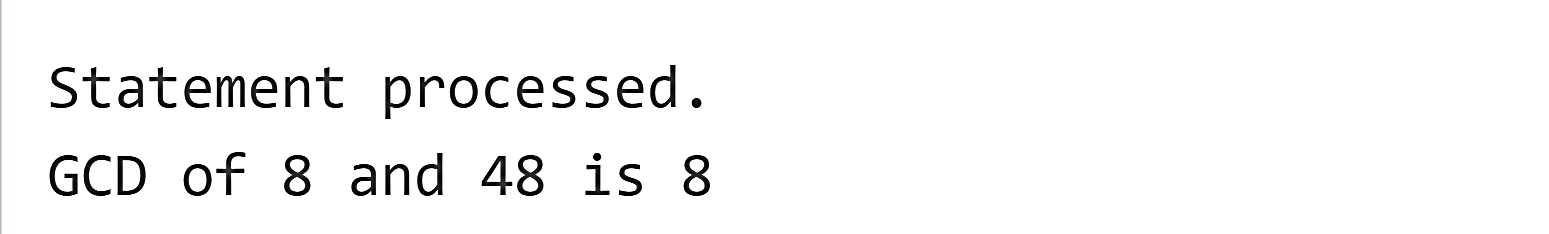
||num2

||' is '

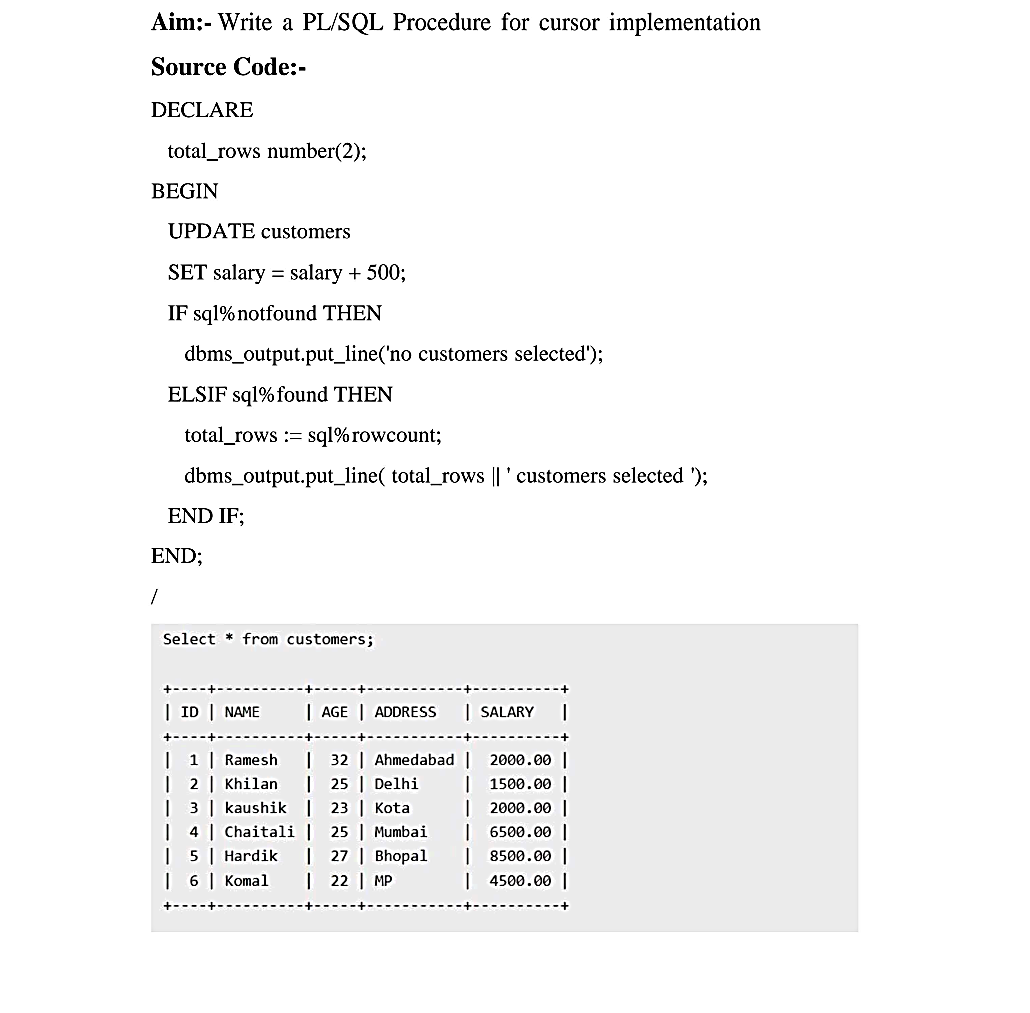
||num1);

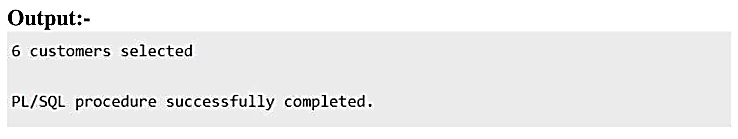
END;

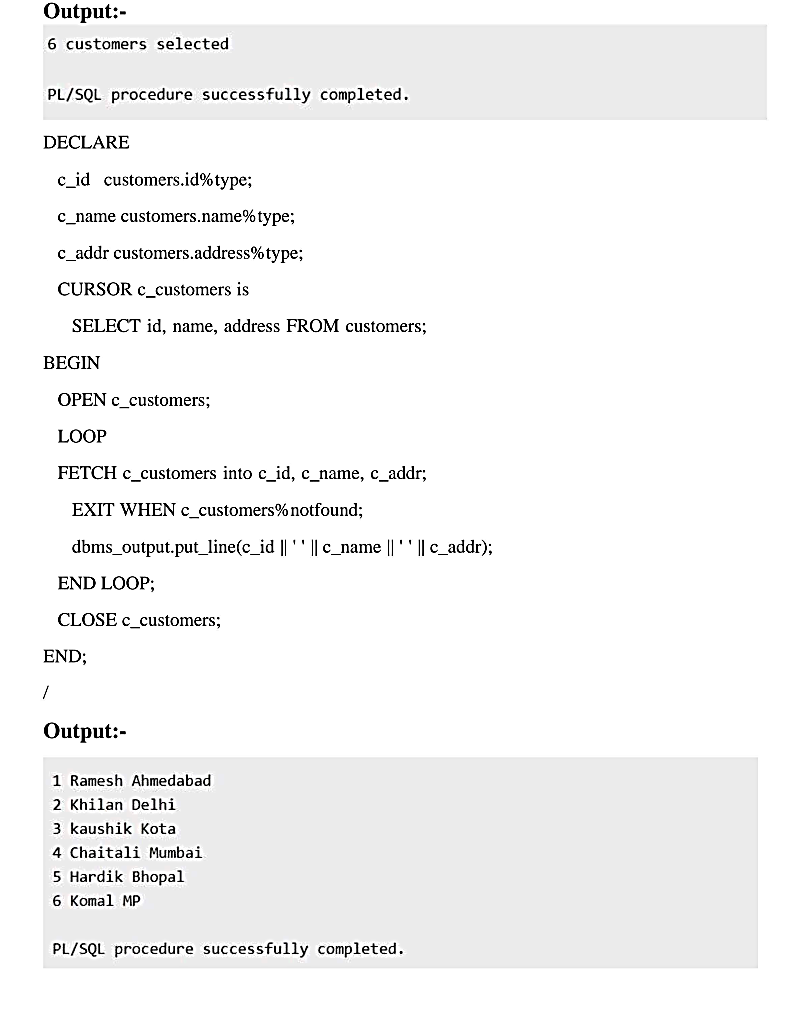
Output-



**EXPERIMENT -14**







**EXPERIMENT -15**

