18CSC303J

Database Management Systems

Record

Register no:	RA1911003010569
Name of the student:	Yogesh
Semester:	6th
Department:	CSE

SRM INSTITUTE OF SCIENCE & TECHNOLOGY
(Deemed to be University u/s 3 of UGC Act, 1956)

SRM INSTITUTE OF SCIENCE AND TECHNOLOGY
S.R.M. NAGAR, KATTANKULATHUR - 603 203 KANCHEEPURAM DISTRICT

BONAFIDE CERTIFICATE

Register No:
RA1911003010569

Certified to be the bonafide record of work done by Yogesh RA1911003010569 *of* CSE *B.Tech Degree course in the Practical* 18CSC303J – Database Management Systems *in* SRM INSTITUTE OF SCIENCE AND TECHNOLOGY, KATTANKULATHUR *during the academic year* 2021-2022*.*

Lab Incharge

Date:

Year Coordinator

Submitted for the University Examination held in the Database Management Systems Lab, SRM INSTITUTE OF SCIENCE AND TECHNOLOGY, Kattankulathur.

Date: Examiner-1
Examiner-2

INDEX SHEET

NAME:- YOGESH   DATE:- 07 January 2022

REGISTER NUMBER:- RA1911003010569

Ex. No: 1   SQL DLL COMMANDS

Date: 06/01/2022

AIM:

To write SQL queries to execute different DLL commands.

Algorithm:

a. Open SQL client server
b. Write the SQL queries for the given task
c. The SQL queries are executed and output is recorded

DML Commands:

CREATE

```
SQL> CREATE TABLE studentinfo (ID VARCHAR(20), NAME VARCHAR(20), DEPARTMENT VARCHAR(20), MARK1 INT, MARK2 INT);

Table created.

SQL> INSERT INTO studentinfo VALUES('RA1911003010569', 'YOGESH', 'CSE', 97, 98);

1 row created.

SQL> INSERT INTO studentinfo VALUES('RA1911003010582', 'DEVESH', 'CSE', 95, 100);

 1 row created.
```

```
SQL> INSERT INTO studentinfo VALUES('RA1911003010625', 'VAIBHAV', 'CSE',
92, 90);

1 row created.

SQL> SELECT * FROM studentinfo;

ID                   NAME                 DEPARTMENT           MARK1
-------------------- -------------------- -------------------- ----------
     MARK2
----------
RA1911003010569      YOGESH               CSE                         100
        98

RA1911003010582      DEVESH               CSE                          95
       100

RA1911003010625      VAIBHAV              CSE                          92
        90

ALTER ADD
SQL>
SQL>
SQL>
SQL> ALTER TABLE studentinfo ADD EMAIL VARCHAR(255);

Table altered.

SQL> SELECT * FROM studentinfo;

ID                   NAME                 DEPARTMENT           MARK1
-------------------- -------------------- -------------------- ----------
     MARK2
----------
```

```
EMAIL
--------------------------------------------------------------------------------
RA1911003010569      YOGESH               CSE                        100
        98



RA1911003010582      DEVESH               CSE                         95
       100



ID                   NAME                 DEPARTMENT                MARK1
-------------------- -------------------- -------------------- ----------
     MARK2
----------
EMAIL
--------------------------------------------------------------------------------


RA1911003010625      VAIBHAV              CSE                         92
        90



ALTER DROP
SQL> ALTER TABLE studentinfo DROP COLUMN EMAIL;

Table altered.

SQL> SELECT * FROM studentinfo;

ID                   NAME                 DEPARTMENT                MARK1
-------------------- -------------------- -------------------- ----------
     MARK2
----------
RA1911003010569      YOGESH               CSE                        100
        98
```

```
RA1911003010582      DEVESH              CSE                     95
       100

RA1911003010625      VAIBHAV             CSE                     92
        90



SQL> ALTER TABLE studentinfo MODIFY COLUMN DEPARTMENT VARCHAR(50);
ALTER TABLE studentinfo MODIFY COLUMN DEPARTMENT VARCHAR(50)
                                     *
ERROR at line 1:
ORA-00905: missing keyword

ALTER MODIFY
SQL> ALTER TABLE studentinfo MODIFY DEPARTMENT varchar(30);

Table altered.

SQL> SELECT * FROM studentinfo;

ID                   NAME                 DEPARTMENT
-------------------- -------------------- ------------------------------
     MARK1      MARK2
---------- ----------
RA1911003010569      YOGESH              CSE
       100         98

RA1911003010582      DEVESH              CSE
        95        100

RA1911003010625      VAIBHAV             CSE
```

```
       92        90
```

ALTER RENAME

```
SQL> ALTER TABLE studentinfo RENAME COLUMN DEPARTMENT TO BRANCH;

Table altered.

SQL> SELECT * FROM studentinfo;

ID                   NAME                 BRANCH
-------------------- -------------------- ------------------------------
     MARK1      MARK2
---------- ----------
RA1911003010569      YOGESH               CSE
       100         98

RA1911003010582      DEVESH               CSE
        95        100

RA1911003010625      VAIBHAV              CSE
        92         90
```

DROP TABLE

```
SQL> DROP TABLE studentinfo;

Table dropped.

SQL> SELECT * FROM studentinfo;
SELECT * FROM studentinfo
              *
ERROR at line 1:
ORA-00942: table or view does not exist
```

```
SQL> spool off
```

Result:

Thus the DLL commands are used to modify or manipulate data records present in the studentinfo database tables.

NAME: Yogesh

REGISTER NUMBER: - RA1911003010569

Date : 25 January 2022

EXP 2 - A : DML COMMANDS

Aim: To execute DML commands in SQL

Algorithm:

d. Open SQL client server
e. Write the SQL queries for the given task
f. The SQL queries are executed and output is recorded

1. List the distinct salary records in the company table.
2. List the records in the company table with the minimum salary.
3. List the records in the company table with the maximum salary.
4. List the top 4 records in the company table.
5. Count the number of records in the company table.
6. Find the average salary from the company table.
7. Find the sum of salary from the company table.
8. List the records from the company table where age ranges between 25 to 27.
9. List the records from the company table where age not ranges between 25 to 27.
10. List the names of the employees from the company where name starts with ‘M’.
11. List the names of the employees f

```
SQL*Plus: Release 21.0.0.0.0 - Production on Tue Jan 25 09:47:54 2022
Version 21.3.0.0.0

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

Enter user-name:
RA1911003010569/RA1911003010569@sowmiya-a2.c6hfisyr3ugy.us-east-1.rds.amazonaw
s.com:1521/a2
```

```
Connected to:
Oracle Database 19c Enterprise Edition Release 19.0.0.0.0 - Production
Version 19.13.0.0.0

SQL> CREATE TABLE employee(name varchar(40), age int, salary int);

Table created.

SQL> ALTER TABLE employee
  2  ADD id int;

Table altered.

SQL> SELECT * FROM employee;

no rows selected

SQL> INSERT INTO employee VALUES('yogesh', 25 , 100000, 569);

1 row created.

SQL> SELECT * FROM employee;

NAME                                            AGE     SALARY         ID
---------------------------------------- ---------- ---------- ----------
yogesh                                           25     100000        569

SQL> INSERT INTO employee VALUES('ABHIGYAN', 27 , 110000, 607);

1 row created.
```

```
SQL> INSERT INTO employee VALUES('VAIBHAV', 26 , 140000, 625);

1 row created.

SQL> INSERT INTO employee VALUES('DIXIT', 36 , 101000, 626);

1 row created.

SQL> INSERT INTO employee VALUES('TUSHAR', 28 , 121000, 702);

1 row created.

SQL> select distinct salary from employee;;
select distinct salary from employee;
                                    *
ERROR at line 1:
ORA-00933: SQL command not properly ended



SQL> select distinct salary from employee;

    SALARY
----------
    100000
    101000
    110000
    140000
    121000

SQL> select min(salary) from employee;

MIN(SALARY)
```

```
-----------
    100000

SQL> select max(salary) from employee;

MAX(SALARY)
-----------
    140000

SQL> select * from employee fetch first 4 rows only;

NAME                                            AGE     SALARY         ID
-------------------------------------------------- ---------- ---------- ----------
yogesh                                            25     100000        569
ABHIGYAN                                          27     110000        607
VAIBHAV                                           26     140000        625
DIXIT                                             36     101000        626

SQL> INSERT INTO employee VALUES('ANURAG', 29 , 101000, 788);

1 row created.

SQL> select * from employee fetch first 4 rows only;

NAME                                            AGE     SALARY         ID
-------------------------------------------------- ---------- ---------- ----------
yogesh                                            25     100000        569
ABHIGYAN                                          27     110000        607
VAIBHAV                                           26     140000        625
DIXIT                                             36     101000        626

SQL> select count(*) from employee;
```

```
  COUNT(*)
----------
         6

SQL> select avg(salary) from employee;

AVG(SALARY)
-----------
 112166.667

SQL> select sum(salary) from employee;

SUM(SALARY)
-----------
     673000

SQL> select * from employee where age >=27 and age<=35;

NAME                                            AGE     SALARY         ID
-------------------------------------------- ---------- ---------- ----------
ABHIGYAN                                         27     110000        607
TUSHAR                                           28     121000        702
ANURAG                                           29     101000        788

SQL> select * from employee where age >=25 and age<=27;

NAME                                            AGE     SALARY         ID
-------------------------------------------- ---------- ---------- ----------
yogesh                                           25     100000        569
ABHIGYAN                                         27     110000        607
VAIBHAV                                          26     140000        625
```

```
SQL> INSERT INTO employee VALUES('MANOJ', 23 , 101200, 988);

1 row created.

SQL> INSERT INTO employee VALUES('DEVD', 26 , 101200, 698);

1 row created.

SQL> select * from employee where name like 'M%';

NAME                                            AGE     SALARY        ID
-------------------------------------- ---------- ---------- ----------
MANOJ                                            23     101200       988

SQL> select * from employee where name like '%D';

NAME                                            AGE     SALARY        ID
-------------------------------------- ---------- ---------- ----------
DEVD                                             26     101200       698
```

Result: Basic SQL DML commands were executed on table “employee

NAME :- Yogesh

REGISTER NUMBER :- RA1911003010569

Date : 25 January 2022

EXP 2 - B :- DDL COMMAND

Aim: To execute DDL commands in SQL

Algorithm:

a. Open SQL client server
b. Write the SQL queries for the given task
c. The SQL queries are executed and output is recorded

```
SQL> CREATE TABLE frgroup(name varchar(40), pntno int, hno varchar(40));

Table created.

SQL>  INSERT INTO frgroup VALUES('Ayush',3685 , 'C-155');

1 row created.

SQL>  INSERT INTO frgroup VALUES('yogesh',3620 , 'D-65');

1 row created.

SQL>  INSERT INTO frgroup VALUES('Nitin',4890 , 'C-71');

1 row created.

SQL>  INSERT INTO frgroup VALUES('Deepanker',4870 , 'D-81');

1 row created.
```

```
SQL>  SELECT * FROM frgroup;

NAME                                      PNTNO
---------------------------------------- ----------
HNO
----------------------------------------
Ayush                                     3685
C-155

yogesh                                     3620
D-65

Nitin                                     4890
C-71


NAME                                      PNTNO
---------------------------------------- ----------
HNO
----------------------------------------
Deepanker                                  4870
D-81



SQL>  ALTER TABLE employee
  2
SQL>  ALTER TABLE frgroup
  2  add id int
  3  ;
```

```
Table altered.

SQL>  INSERT INTO frgroup VALUES('Pratesh',4110 , 'B-81', 88);

1 row created.

SQL>  SELECT * FROM frgroup;

NAME                                   PNTNO
-------------------------------------- ----------
HNO                                     ID
-------------------------------------- ----------
Ayush                                  3685
C-155

yogesh                                 3620
D-65

Nitin                               4890
C-71


NAME                                   PNTNO
-------------------------------------- ----------
HNO                                     ID
-------------------------------------- ----------
Deepanker                                4870
D-81

Pratesh                                4110
B-81                                   88
```

```
SQL> alter table frgroup
  2  drop column hno;

Table altered.

SQL> alter table frgroup
  2  rename to township;

Table altered.

SQL>  SELECT * FROM township;

NAME                                  PNTNO        ID
------------------------------------- ---------- ----------
Ayush                                3685
yogesh                                3620
Nitin                              4890
Deepanker                               4870
Pratesh                               4110        88



SQL> alter table township modify name varchar(30);

Table altered.
```

Result: Basic SQL DDL commands were executed on table “employee”

Name :- yogesh

Register No :- RA1911003010569

Date : 02 February 2022

EXP - 3: DCL and TCL

Aim: To execute DCL and TCL commands in SQL

Algorithm:

a. Open SQL client server
b. Write the SQL queries for the given task
c. The SQL queries are executed and output is recorded

```
SQL*Plus: Release 21.0.0.0.0 - Production on Wed Feb 2 09:52:03 2022

Version 21.3.0.0.0


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


Enter user-name:
RA1911003010569/RA1911003010569@sowmiya-a2.c6hfisyr3ugy.us-east-1.rds.amazonaw
s.com:1521/a2

Last Successful login time: Sat Jan 29 2022 10:34:25 +05:30


Connected to:

Oracle Database 19c Enterprise Edition Release 19.0.0.0.0 - Production

Version 19.13.0.0.0


SQL> create table friends(

  2  name char(20),

  3  age int,

  4  pnumber int);


Table created.
```

```
SQL> insert into friends values('Kusum', 21, 9874566578);

1 row created.

SQL> insert into friends values('Abhigyan', 20, 8884566578);

1 row created.

SQL> insert into friends values('Vaibhav', 19, 9999978510);

1 row created.

SQL> insert into friends values('Yogesh', 22, 9999900000);

1 row created.

SQL> select * from friends;

NAME                        AGE    PNUMBER
-------------------- ---------- ----------
Kusum                        21 9874566578
Abhigyan                     20 8884566578
Vaibhav                      19 9999978510
Yogesh                       22 9999900000

SQL> savepoint s1;

Savepoint created.

SQL> update set name = 'yogesh' where age = 19;
update set name = 'yogesh' where age = 19
```

```
      *
ERROR at line 1:
ORA-00903: invalid table name



SQL> update friends set name = 'yogesh' where age = 19;

1 row updated.

SQL> select * from friends;

NAME                        AGE    PNUMBER
-------------------- ---------- ----------
Kusum                        21 9874566578
Abhigyan                     20 8884566578
yogesh                       19 9999978510
Yogesh                       22 9999900000

SQL> rollback to s1;

Rollback complete.

SQL> select * from friends;

NAME                        AGE    PNUMBER
-------------------- ---------- ----------
Kusum                        21 9874566578
Abhigyan                     20 8884566578
Vaibhav                      19 9999978510
Yogesh                       22 9999900000

SQL> delete from friends where age <21;
```

```
2 rows deleted.

SQL> savepoint s2;

Savepoint created.

SQL> select * from friends;

NAME                        AGE    PNUMBER
-------------------- ---------- ----------
Kusum                        21 9874566578
Yogesh                       22 9999900000

SQL> rollback to s1;

Rollback complete.

SQL> select * from friends;

NAME                        AGE    PNUMBER
-------------------- ---------- ----------
Kusum                        21 9874566578
Abhigyan                     20 8884566578
Vaibhav                      19 9999978510
Yogesh                       22 9999900000

SQL> commit;

Commit complete.

SQL> set transaction read only;
```

```
Transaction set.

SQL> grant select on users to'Udit'@'localhost';
grant select on users to'Udit'@'localhost'
                         *
ERROR at line 1:
ORA-00987: missing or invalid username(s)



SQL> revoke select on users from 'Udit'@'localhost';
revoke select on users from 'Udit'@'localhost'
                             *
ERROR at line 1:
ORA-00987: missing or invalid username(s)
```

Result: Basic SQL DCL and TCL commands were executed on table “players”

NAME : - Yogesh

REGISTER NUMBER : - RA1911003010569

Date: 09 February 2022

<u>Database Management Systems</u>

<u>EXP- 4</u>

Aim: To execute the inbuilt functions in SQL

Algorithm:

a. Open SQL client server
b. Write the SQL queries for the given inbuilt function
c. The SQL queries are executed and output is recorded

<u>CODE</u>

```
SQL> create table exam(id int, name varchar(20), email varchar(20), phonemo varchar(10));

Table created.

SQL> desc exam;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 ID                                                 NUMBER(38)
 NAME                                               VARCHAR2(20)
 EMAIL                                              VARCHAR2(20)
 PHONEMO                                            VARCHAR2(10)

SQL> alter exam teacher add marks int;
alter exam teacher add marks int
      *
ERROR at line 1:
```

```
ORA-00940: invalid ALTER command



SQL> alter table exam teacher add marks int;
alter table exam teacher add marks int
                 *
ERROR at line 1:
ORA-01735: invalid ALTER TABLE option



SQL> alter table exam add marks int;

Table altered.

SQL> insert into exam values(1,'yogesh', 'adi1111@gmail.com', 9874563210, 89);

1 row created.

SQL> insert into exam values(2,'Abhigyan', 'abhi01@gmail.com', 9874532160, 93);

1 row created.

SQL> insert into exam values(3,'Vaibhav', 'vj66@gmail.com', 7878532160, 95);

1 row created.

SQL> insert into exam values(4,'Kusum', 'ks123@gmail.com', 9007132160, 86);

1 row created.

SQL> insert into exam values(5,'Advait', 'av@gmail.com', 7897132160, 92);
```

```
1 row created.

SQL> select * from exam;

        ID NAME                 EMAIL                PHONEMO         MARKS
---------- -------------------- -------------------- ---------- ----------
         1 yogesh               adi1111@gmail.com    9874563210         89
         2 Abhigyan             abhi01@gmail.com     9874532160         93
         3 Vaibhav              vj66@gmail.com       7878532160         95
         4 Kusum                ks123@gmail.com      9007132160         86
         5 Advait               av@gmail.com         7897132160         92

SQL> select name ASCII('a') as NumCode from exam;
select name ASCII('a') as NumCode from exam
            *
ERROR at line 1:
ORA-00923: FROM keyword not found where expected



SQL> select name, ASCII('a') as NumCode from exam;

NAME                    NUMCODE
-------------------- ----------
yogesh                       97
Abhigyan                     97
Vaibhav                      97
Kusum                        97
Advait                       97

SQL> select ASCII(name) as ASCIIfirstchar from exam;

ASCIIFIRSTCHAR
```

```
--------------
          65
          65
          86
          75
          65

SQL> select name, ASCII(name) as ASCIIfirstchar from exam;

NAME                 ASCIIFIRSTCHAR
-------------------- --------------
yogesh                           65
Abhigyan                         65
Vaibhav                          86
Kusum                            75
Advait                           65

SQL> select name,concat(name,"yadav") as con from exam where name = 'yogesh';
select name,concat(name,"yadav") as con from exam where name = 'yogesh'
                        *
ERROR at line 1:
ORA-00904: "yadav": invalid identifier



SQL> select name,concat(name,'yadav') as con from exam where name = 'yogesh';

NAME                 CON
-------------------- --------------------------
yogesh               yogeshyadav

SQL> select * from exam;
```

```
        ID        NAME               EMAIL                  PHONEMO       MARKS
      ---------- ------------------- -------------------         -----------------      -----------
        1         yogesh             adi1111@gmail.com    9874563210        89
        2         Abhigyan         abhi01@gmail.com     9874532160        93
        3         Vaibhav          vj66@gmail.com        7878532160        95
        4         Kusum            ks123@gmail.com      9007132160        86
        5         Advait            av@gmail.com          7897132160        92

SQL> insert into exam values(6,'suraj', 'sk789@gmail.com', 9098972160, 86);

1 row created.

SQL> insert into exam values(7,'tushar', 'tt666@gmail.com', 9999972160, 92);

1 row created.

SQL> select * from exam;

      ID   NAME               EMAIL             PHONEMO       MARKS
---------- ------------------- ------------------- ---------- ----------
     1 yogesh             adi1111@gmail.com    9874563210        89
       2 Abhigyan           abhi01@gmail.com     9874532160        93
       3 Vaibhav            vj66@gmail.com       7878532160        95
       4 Kusum              ks123@gmail.com      9007132160        86
       5 Advait             av@gmail.com         7897132160        92
       6 suraj              sk789@gmail.com      9098972160        86
       7 tushar             tt666@gmail.com      9999972160        92

7 rows selected.

SQL> select name, initcap(name) as capital from exam;
```

```
NAME                CAPITAL
-------------------- --------------------
yogesh              yogesh
Abhigyan            Abhigyan
Vaibhav             Vaibhav
Kusum               Kusum
Advait              Advait
suraj               Suraj
tushar              Tushar

7 rows selected.

SQL> select name, lower(name) as lo from exam;

NAME                LO
-------------------- --------------------
yogesh              yogesh
Abhigyan            abhigyan
Vaibhav             vaibhav
Kusum               kusum
Advait              advait
suraj               suraj
tushar              tushar

7 rows selected.

SQL> select ltrim('    exam') as lefttrim from exam;

LEFT
----
exam
exam
```

```
exam

exam

exam

exam

exam


7 rows selected.


SQL> select rtrim('exam     ') as righttrim from exam;


RIGH

----

exam

exam

exam

exam

exam

exam

exam


7 rows selected.


SQL> select name, substr(name, 1, 3) as substring from exam;


NAME                 SUBSTRING

-------------------- ------------

yogesh               Adi

Abhigyan             Abh

Vaibhav              Vai

Kusum                Kus

Advait               Adv

suraj                sur
```

```
tushar          tus

7 rows selected.

SQL> alter table exam add scgpa float;

Table altered.

SQL> update exam set sgpa = 9.228 where name='yogesh';
update exam set sgpa = 9.228 where name='yogesh'
                *
ERROR at line 1:
ORA-00904: "SGPA": invalid identifier


SQL> update exam set scgpa = 9.228 where name='yogesh';

1 row updated.

SQL> update exam set scgpa = 9.67 where name='Abhigyan';

1 row updated.

SQL> update exam set scgpa = 9.77 where name='Vaibhav';

1 row updated.

SQL> update exam set scgpa = 8.6 where name='Kusum';

1 row updated.

SQL> update exam set scgpa = 9.4 where name='Advait';
```

```
1 row updated.

SQL> update exam set scgpa = 8.22 where name='suraj';

1 row updated.

SQL> update exam set scgpa = 9.55 where name='tushar';

1 row updated.

SQL> select * from exam;

        ID NAME                 EMAIL                PHONEMO         MARKS
---------- -------------------- -------------------- ---------- ----------
     SCGPA
----------
         1 yogesh               adi1111@gmail.com    9874563210         89
     9.228

         2 Abhigyan             abhi01@gmail.com     9874532160         93
      9.67

         3 Vaibhav              vj66@gmail.com       7878532160         95
      9.77


        ID NAME                 EMAIL                PHONEMO         MARKS
---------- -------------------- -------------------- ---------- ----------
     SCGPA
----------
         4 Kusum                ks123@gmail.com      9007132160         86
```

```
       8.6

         5 Advait               av@gmail.com         7897132160         92
       9.4

         6 suraj                sk789@gmail.com      9098972160         86
      8.22


        ID NAME                 EMAIL                PHONEMO        MARKS
---------- -------------------- -------------------- ---------- ----------
     SCGPA
----------
         7 tushar               tt666@gmail.com      9999972160         92
      9.55


7 rows selected.

SQL> select name, round(scgpa) as scgpa from exam;

NAME                      SCGPA
-------------------- ----------
yogesh                        9
Abhigyan                     10
Vaibhav                      10
Kusum                         9
Advait                        9
suraj                         8
tushar                       10

7 rows selected.
```

```
SQL> select name, replace(name, 'a', 'e') as replacename from exam where name='tushar';

NAME                 REPLACENAME
-------------------- --------------------
tushar               tusher

SQL> select scgpa, abs(scgpa) as absrating from exam;

     SCGPA  ABSRATING
---------- ----------
     9.228      9.228
      9.67       9.67
      9.77       9.77
       8.6        8.6
       9.4        9.4
      8.22       8.22
      9.55       9.55

7 rows selected.

SQL> select name, power(scgpa,2) as poweroftwo from exam;

NAME                 POWEROFTWO
-------------------- ----------
yogesh                85.155984
Abhigyan                93.5089
Vaibhav                 95.4529
Kusum                     73.96
Advait                    88.36
suraj                   67.5684
tushar                  91.2025
```

```
7 rows selected.

SQL> select scgpa, log(scgpa,10) as log from exam;

     SCGPA        LOG
---------- ----------
     9.228 1.03615391
      9.67 1.01478905
      9.77  1.0102086
       8.6 1.07009273
       9.4  1.0276142
      8.22  1.0930493
      9.55 1.02040465

7 rows selected.

SQL> select count(name) from exam;

COUNT(NAME)
-----------
          7

SQL> select avg(scgpa) from exam;

AVG(SCGPA)
----------
9.20542857

SQL> select sum(scgpa) from exam;

SUM(SCGPA)
```

```
----------
    64.438

SQL> select cos(scgpa) from exam;

COS(SCGPA)
----------
-.98070161
-.97008344
-.94100034
-.67872005
-.99969304
-.35790039
-.99216996

7 rows selected.

SQL> select sin(scgpa) from exam;

SIN(SCGPA)
----------
.195510492
-.24277173
-.33840562
.734397098
.024775425
.933759772
-.12489504

7 rows selected.

SQL> select scgpa , ceil(scgpa) from exam;
```

```
     SCGPA CEIL(SCGPA)
---------- -----------
     9.228          10
      9.67          10
      9.77          10
       8.6           9
       9.4          10
      8.22           9
      9.55          10

7 rows selected.

SQL> select scgpa, atan(scgpa) from exam;

     SCGPA ATAN(SCGPA)
---------- -----------
     9.228  1.46285171
      9.67     1.46775
      9.77  1.46879738
       8.6  1.45503711
       9.4  1.46481197
      8.22  1.44973671
      9.55   1.4664645

7 rows selected.

SQL> select scgpa , floor(scgpa) from exam;

     SCGPA FLOOR(SCGPA)
---------- ------------
     9.228            9
```

```
      9.67           9
      9.77           9
       8.6          8
       9.4          9
      8.22           8
      9.55           9

7 rows selected.

SQL> select max(scgpa) from exam;

MAX(SCGPA)
----------
      9.77

SQL> select min(scgpa) from exam;

MIN(SCGPA)
----------
      8.22

SQL> select variance(scgpa) from exam;

VARIANCE(SCGPA)
---------------
     .338212952

SQL> select name, lpad(name, '6', '--> ') from exam;

NAME                 LPAD(NAME,'6','-->')
-------------------- ------------------------
yogesh               yogesh
```

```
Abhigyan          Abhigy
Vaibhav           Vaibha
Kusum             -Kusum
Advait            Advait
suraj             -suraj
tushar            tushar

7 rows selected.

SQL> select name, lpad(name, '15', '*') from exam;

NAME
--------------------
LPAD(NAME,'15','*')
------------------------------------------------------------
yogesh
*********yogesh

Abhigyan
*******Abhigyan

Vaibhav
********Vaibhav


NAME
--------------------
LPAD(NAME,'15','*')
------------------------------------------------------------
Kusum
**********Kusum
```

```
Advait

*********Advait


suraj

**********suraj



NAME

--------------------

LPAD(NAME,'15','*')

------------------------------------------------------------

tushar

*********tushar



7 rows selected.


SQL> select rtrim(name) as name from exam;


NAME

--------------------

Yogesh

Abhigyan

Vaibhav

Kusum

Advait

suraj

tushar


7 rows selected.


SQL> select ltrim(name) as name from exam;
```

```
NAME
--------------------
Yogesh
Abhigyan
Vaibhav
Kusum
Advait
suraj
tushar
7 rows selected.
```

<u>RESULT</u>

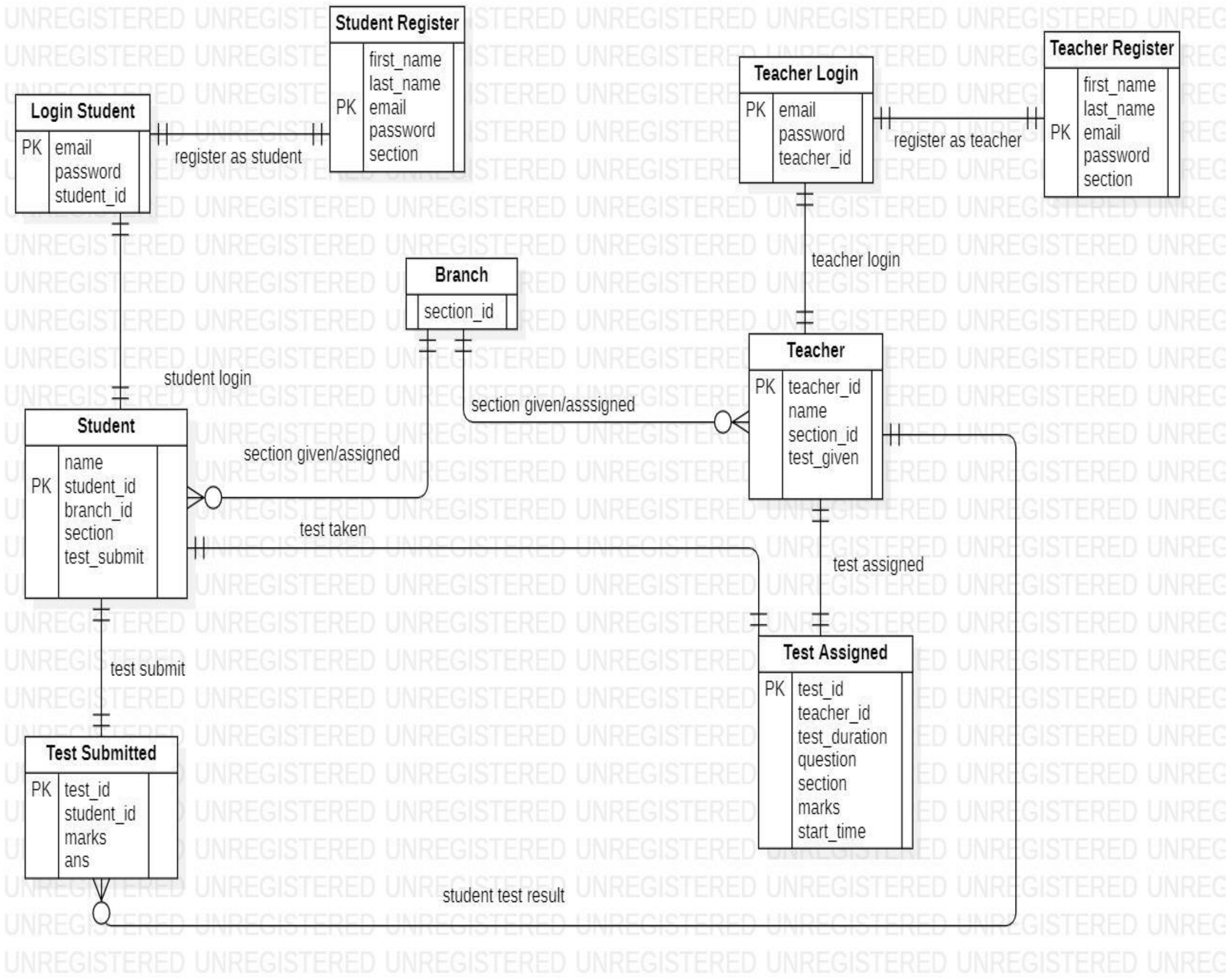
The aim of the experiment was to test the inbuilt functions of SQL, and it was successfully performed on the SQL online editor.

Name:- Yogesh

Register No:- RA1911003010569

Date:16 February 2022

DBMS EXP 5 :- ER DIAGRAM



RESULT:

Hence the ER diagram for the Exam Management system is made using the above-mentioned tool StarUml.

NAME:-yogesh

REGISTER NO: RA1911003010569

Date: 23 February 2022

EX 6: Subqueries

Aim: To execute certain subqueries in the given scenarios

Algorithm:

a. Open SQL client server
b. Write the SQL queries for the given inbuilt function
c. The SQL queries are executed and output is recorded

I. Suppliers(sid:integer, sname:string, city:string, street:string)
Parts(pid:integer, pname:string, color:string)
Catalog(sid:integer, pid:integer, cost:real)

SQL QUERIES:--

```
SQL> create table suppliers (sid int PRIMARY KEY, sname varchar(25), city
varchar(25), street varchar(25));

Table created.

SQL> create table parts (pid int PRIMARY KEY, pname varchar(25), color
varchar(25));

Table created.

SQL> create table catalog( sid int,  FOREIGN KEY (sid) REFERENCES
suppliers(sid), pid int,FOREIGN KEY (pid) REFERENCES parts(pid), cost real);

Table created.

SQL> insert into suppliers(1, 'yogesh', 'bareilly', 's1');
insert into suppliers(1, 'yogesh', 'bareilly', 's1')
                     *
ERROR at line 1:
ORA-00928: missing SELECT keyword


SQL> insert into suppliers values(1, 'yogesh', 'bareilly', 's1');
```

```
1 row created.

SQL> insert into suppliers values(2 , 'abhigyan', 'mumbai', 's23');

1 row created.

SQL> insert into suppliers values(3 , 'vaibhav', 'chennai', 's33');

1 row created.

SQL> insert into suppliers values(4 , 'kusum', 'banglore', 's40');

1 row created.

SQL> insert into suppliers values(5 , 'suraj', 'delhi', 's55');

1 row created.

SQL> insert into parts values(1, 'hammer', 'black');

1 row created.

SQL> insert into parts values(2, 'bolts', 'blue');

1 row created.

SQL> insert into parts values(3, 'nuts', 'red');

1 row created.

SQL> insert into parts values(4, 'spanner', 'grey');

1 row created.

SQL> insert into parts values(5, 'lock', 'silver');

1 row created.

SQL> insert into catalog values(1, 2, 3.14);

1 row created.

SQL> insert into catalog values(2, 3, 7.84);

1 row created.

SQL> insert into catalog values(3, 4, 8.94);

1 row created.
```

```
SQL> insert into catalog values(4, 5, 9.84);

1 row created.

SQL> insert into catalog values(5, 1, 10.41);

1 row created.

SQL> select *from suppliers;

       SID SNAME                    CITY
---------- ------------------------ ------------------------
STREET
------------------------
         1 yogesh                   bareilly
s1

         2 abhigyan                 mumbai
s23

         3 vaibhav                  chennai
s33


       SID SNAME                    CITY
---------- ------------------------ ------------------------
STREET
------------------------
         4 kusum                    banglore
s40

         5 suraj                    delhi
s55


SQL> select *from parts;

       PID PNAME                    COLOR
---------- ------------------------ ------------------------
         1 hammer                   black
         2 bolts                    blue
         3 nuts                     red
         4 spanner                  grey
         5 lock                     silver

SQL> select *from catalog;

       SID        PID       COST
---------- ---------- ----------
         1          2       3.14
```

```
    2        3      7.84
    3        4      8.94
    4        5      9.84
    5        1     10.41
```

```
SQL> select s.sname from suppliers s where s.sid not in (select c.sid from catalog c
where c.pid not in(select p.pid from parts p where p.color<>'blue'));

SNAME
-------------------------
abhigyan
vaibhav
kusum
suraj
```

- Write a query retrieves the name (sname) of suppliers, who have supplied a non-blue part.

II. Consider the following relations:

Student	
Roll_No	Student_Name
1	Raj
2	Rohit
3	Raj

Performance		
Roll_No	Course	Marks
1	Math	80
1	English	70
2	Math	75
3	English	80
2	Physics	65
3	Math	80

- Write a query to find the sum of marks for each student.

SQL QUERIES:-

```
SQL>
SQL>
SQL> create table Student( Roll_No int PRIMARY KEY, Student_Name
varchar(15));

Table created.

SQL> create table Performance(Roll_No int, FOREIGN KEY (Roll_No)
REFERENCES Student(Roll_No), Course varchar(15), Marks int);

Table created.
```

```
SQL> insert into Student values(1, 'Raj');

1 row created.

SQL> insert into Student values(2, 'Rohit');

1 row created.

SQL> insert into Student values(3, 'Raj');

1 row created.

SQL> insert into Performance values(1, 'Math', 80);

1 row created.

SQL> insert into Performance values(1, 'English', 70);

1 row created.

SQL> insert into Performance values(2, 'Math', 75);

1 row created.

SQL> insert into Performance values(3, 'English', 80);

1 row created.

SQL> insert into Performance values(2, 'Physics', 65);

1 row created.

SQL> insert into Performance values(3, 'Math', 80);

1 row created.

SQL>
SQL> 1 row created
SQL>
SQL> select * from Student;

   ROLL_NO STUDENT_NAME
---------- ---------------
         1 Raj
         2 Rohit
         3 Raj
```

```
SQL> select * from Performance;

  ROLL_NO COURSE              MARKS
---------- --------------- ----------
        1 Math                 80
        1 English               70
        2 Math                 75
        3 English               80
        2 Physics               65
        3 Math                 80
```

6 rows selected.

```
SQL> SELECT S.Roll_No, Sum(P.Marks) FROM Student S, Performance P
WHERE S.Roll_No= P.Roll_No GROUP BY S.Roll_No;

  ROLL_NO SUM(P.MARKS)
---------- ------------
        1         150
        2         140
         3         160
```

III. Passenger (pid, pname, age)
Reservation (pid, class, tid)

Table: Passenger

pid	pname	age
0	Sachin	65
1	Rahul	66
2	Sourav	67
3	Anil	69

Table : Reservation

pid	class	tid
0	AC	8200
1	AC	8201
2	SC	8201
5	AC	8203
1	SC	8204
3	AC	8202

- Write a query to find the passengers who have done registration and also who have age greater than 65 who are travelling in “AC” class.

SQL QUERIES:-

SQL>

```
SQL> create table Passenger(pid int, pname char(15),age int);

Table created.

SQL> create table Reservation(pid int, class char(2),tid int);

Table created.

SQL> insert into passenger values(0,'Sachin', 65);

1 row created.

SQL> insert into passenger values(1,'Rahul', 66);

1 row created.

SQL> insert into passenger values(2,'Sourav', 67);

1 row created.

SQL> insert into passenger values(3,'Anil', 69);

1 row created.

SQL> insert into reservation values(0,'AC', 8200);

1 row created.

SQL> insert into reservation values(1,'AC', 8201);
```

```
1 row created.

SQL> insert into reservation values(2,'SC', 8201);

1 row created.

SQL> insert into reservation values(5,'AC', 8203);

1 row created.

SQL> insert into reservation values(1,'SC', 8204);

1 row created.

SQL> insert into reservation values(3,'AC', 8202);

1 row created.

SQL> select * from passenger;

       PID PNAME                  AGE
---------- --------------- ----------
         0 Sachin                  65
         1 Rahul                   66
         2 Sourav                  67
         3 Anil                    69

SQL> select * from reservation;

       PID CL        TID
```

```
---------- -- ----------
         0 AC       8200
         1 AC       8201
         2 SC       8201
         5 AC       8203
         1 SC       8204
         3 AC       8202

6 rows selected.

SQL> select R.pid from reservation R where R.class = 'AC' AND EXISTS (select *
from passenger P where P.age > 65 and P.pid = R.pid);

       PID
----------
         1
         3
```

Result: Subqueries were executed successfully for the given scenarios

NAME : - Yogesh

Register Number : -RA1911003010569

Date : 09 March 2022

Database Management Systems-18CSC303J

EXPERIMENT- 7

Set Operators and Views

Aim: To execute Set Operators and View queries

Algorithm:

a. Open SQL client server
b. Write the SQL queries for the given set operators and views in the given scenarios
c. The SQL queries are executed and output is recorded

1. Write the query to demonstrate the various set operators (UNION, UNION ALL, MINUS, INTERSECT)

Test Table

Roll_No	Name	Status
12	Nick	Pass
13	Paul	Pass
11	Ricky	Fail
14	Smith	Fail
15	Tim	Pass

Retest table

Roll_No	Name
11	Ricky
15	Smith

```
SQL> create table Test_table(Roll_No int, Name char(15), Status char(10));

Table created.

SQL> insert into Test_Table values(12, 'Nick' , Pass);
insert into Test_Table values(12, 'Nick' , Pass)
                                         *
ERROR at line 1:
ORA-00984: column not allowed here
```

```
SQL> insert into Test_Table values(12, 'Nick' , 'Pass');

1 row created.

SQL> insert into Test_Table values(13, 'Paul' , 'Pass');

1 row created.

SQL> insert into Test_Table values(11, 'Ricky' , 'Fail');

1 row created.

SQL> insert into Test_Table values(14, 'Smith' , 'Fail');

1 row created.

SQL> insert into Test_Table values(15, 'Tim' , 'Pass');

1 row created.

SQL> select * from Test_Table;

  ROLL_NO NAME            STATUS
---------- --------------- ----------
        12 Nick            Pass
        13 Paul            Pass
        11 Ricky           Fail
        14 Smith           Fail
        15 Tim             Pass

SQL> create table Retest_Table(Roll_No int, Name char(15));

Table created.

SQL> insert into Retest_Table values(11, 'Ricky');

1 row created.

SQL> insert into Retest_Table values(15, 'Smith');

1 row created.

SQL> select * from Retest_Table;

  ROLL_NO NAME
---------- ---------------
        11 Ricky
        15 Smith

SQL>  SELECT Roll_No, Name FROM Test_Table
  2  UNION ALL
  3
SQL> SELECT Roll_No, Name FROM Test_Table UNION ALL Select Roll_No from
Rtest_Table;
SELECT Roll_No, Name FROM Test_Table UNION ALL Select Roll_No from Rtest_Table
                                                                *
ERROR at line 1:
ORA-00942: table or view does not exist


SQL> SELECT Roll_No, Name FROM Test_Table UNION ALL Select Roll_No from
Retest_Table;
SELECT Roll_No, Name FROM Test_Table UNION ALL Select Roll_No from Retest_Table
*
ERROR at line 1:
ORA-01789: query block has incorrect number of result columns


SQL> SELECT Roll_No, Name FROM Test_Table UNION ALL Select Roll_No,Name from
Retest_Table;

   ROLL_NO NAME
---------- ---------------
        12 Nick
        13 Paul
        11 Ricky
        14 Smith
        15 Tim
        11 Ricky
        15 Smith

7 rows selected.

SQL> SELECT Roll_No, Name FROM Test_Table UNION Select Roll_No,Name from
Retest_Table;

   ROLL_NO NAME
---------- ---------------
        11 Ricky
        12 Nick
        13 Paul
        14 Smith
        15 Smith
        15 Tim

6 rows selected.

SQL> SELECT Roll_No, Name FROM Test_Table MINUS Select Roll_No,Name from
Retest_Table;

   ROLL_NO NAME
---------- ---------------
        12 Nick
        13 Paul
        14 Smith
        15 Tim

SQL> SELECT Roll_No, Name FROM Test_Table INTERSECT Select Roll_No,Name from
Retest_Table;

   ROLL_NO NAME
---------- ---------------
        11 Ricky

SQL>
SQL>
```

2. Write a query using INTERSECT set operator to list the student id and residence location of the students.

   Student table

Student_id	Student_name	City	Age
1	Raj	Chennai	25
2	yogesh	Vizag	24
3	Ram	Pune	26
4	Sam	Delhi	28

   Student personal table

Student_id	Department	College	City	Rank
1	Science	DCE	Chennai	4
2	Arts	ABC	Vizag	1
3	Commerce	KEC	Delhi	2
4	Science	SIT	Pune	3
5	Electronics	KLN	Pune	5

```
SQL>
SQL> CREATE TABLE Student_Table(Student_id int,Student_name varchar(10),City
varchar(20),Age int);

Table created.

SQL>  CREATE TABLE Student_personal(Student_id int,Department varchar(20),College
varchar(20),City varchar(20),Rank int);

Table created.

SQL> INSERT INTO Student_Table VALUES(1,'Raj','Chennai',25)
```

```
  2  ;

1 row created.

SQL> INSERT INTO Student_Table VALUES(2,'yogesh','Vizag',24);

1 row created.

SQL> INSERT INTO Student_Table VALUES(3,'Ram','Pune',26);

1 row created.

SQL>  INSERT INTO Student_Table VALUES(4,'Sam','Delhi',28);

1 row created.

SQL>  INSERT INTO Student_personal VALUES(1,'Science','DCE','Chennai',4);

1 row created.

SQL> INSERT INTO Student_personal VALUES(2,'Arts','ABC','Vizag',1);

1 row created.

SQL> INSERT INTO Student_personal VALUES(3,'Commerce','KEC','Delhi',2);

1 row created.

SQL> INSERT INTO Student_personal VALUES(4,'Science','SIT','Pune',3);

1 row created.

SQL> INSERT INTO Student_personal VALUES(5,'Electronics','KLN','Pune',5);
SP2-0734: unknown command beginning "INSERT INTO..." - rest of line ignored.
SQL>
SQL> INSERT INTO Student_personal VALUES(5,'Electronics','KLN','Pune',5);

1 row created.

SQL> SELECT * FROM Student_Table;

STUDENT_ID STUDENT_NA CITY                        AGE
---------- ---------- -------------------- ----------
         1 Raj        Chennai                      25
         2 yogesh     Vizag                        24
         3 Ram        Pune                         26
         4 Sam        Delhi                        28

SQL> SELECT * FROM Student_personal;

STUDENT_ID DEPARTMENT           COLLEGE              CITY
---------- -------------------- -------------------- --------------------
      RANK
```

```
----------
         1 Science             DCE                 Chennai
         4

         2 Arts               ABC                 Vizag
         1

         3 Commerce            KEC                 Delhi
         2


STUDENT_ID DEPARTMENT          COLLEGE             CITY
---------- -------------------- -------------------- --------------------
      RANK
----------
         4 Science             SIT                 Pune
         3

         5 Electronics         KLN                 Pune
         5


SQL> SELECT Student_id , City FROM Student_Table  INTERSECT SELECT Student_id ,
City FROM Student_personal;

STUDENT_ID CITY
---------- --------------------
         1 Chennai
         2 Vizag
```

3. Write a query using UNION & UNION ALL set operators to list the student id and residence location of the students using the student and student personal table given above.

```
SQL> SELECT Student_id , City FROM Student_Table UNION  SELECT Student_id ,
City FROM Student_personal;

STUDENT_ID CITY
---------- --------------------
         1 Chennai
         2 Vizag
         3 Delhi
         3 Pune
         4 Delhi
         4 Pune
         5 Pune

7 rows selected.

SQL> SELECT Student_id , City FROM Student_Table UNION ALL  SELECT Student_id
, City FROM Student_personal;

STUDENT_ID CITY
---------- --------------------
```

```
        1 Chennai
        2 Vizag
        3 Pune
        4 Delhi
        1 Chennai
        2 Vizag
        3 Delhi
        4 Pune
        5 Pune

9 rows selected.
```

4. Write a query using MINUS set operators to list the student id and residence location of the students using the student and student personal table given above.

```
SQL> SELECT Student_id , City FROM Student_Table MINUS SELECT Student_id ,
City FROM Student_personal;

STUDENT_ID CITY
---------- --------------------
        3 Pune
        4 Delhi

SQL>
```

5. Employee(Business_Id, login_Id, Organization_Name, Organizational_level, Job_title, Gender, Martial_status, BirthDate); (Minimum 10 records need to be created)

   - Write a query for SQL view (view name: Employee_Records) to fetch columns of the table and filter the results using where clause with the martial_status ‘M’.
   - Write a query to update, delete and insert from SQL view (view name: Employee_Records) table.

   ```
   SQL> create table Q5EMPLOYEE(Business_Id int, login_Id int, Organization_Name
   varchar(10), Organizational_level varchar(10), Job_title varchar(10), Gender varchar(1),
   Martial_status varchar(10), BirthDate DATE);

   Table created.

   SQL> insert into EMPLOYEE VALUES
   (1,8080,'yogesh','ENTRY','INTERN','M','SINGLE',TO_DATE('01-01-2000','dd-mm-yyyy'));

   1 row created.

   SQL> insert into EMPLOYEE VALUES
   (2,5050,'Suraj','ENTRY','INTERN','M','SINGLE',TO_DATE('27-01-2000','dd-mm-yyyy'));

   1 row created.

   SQL> insert into EMPLOYEE VALUES
   (3,2850,'Abhigyan','ENTRY','SDE','M','SINGLE',TO_DATE('27-01-2010','dd-mm-yyyy'));

   1 row created.
   ```

```
SQL> insert into EMPLOYEE VALUES
(4,9850,'Advait','ENTRY','SDE2','M','MARRIED',TO_DATE('15-03-2022','dd-mm-yyyy'));

1 row created.

SQL> insert into EMPLOYEE VALUES
(5,9890,'Advait','MID','SDE2','M','MARRIED',TO_DATE('25-03-2022','dd-mm-yyyy'));

1 row created.

SQL> insert into EMPLOYEE VALUES
(6,8780,'VAIBHAV','TOP','SDE3','M','SINGLE',TO_DATE('15-12-2002','dd-mm-yyyy'));

1 row created.

SQL> insert into EMPLOYEE VALUES (7,9790,'Yogesh','TOP','TEAM
LEAD','M','SINGLE',TO_DATE('15-12-2012','dd-mm-yyyy'));

1 row created.

SQL> insert into EMPLOYEE VALUES
(8,9990,'DIXIT','TOP','MANAGER','M','SINGLE',TO_DATE('15-12-2002','dd-mm-yyyy'));

1 row created.

SQL> insert into EMPLOYEE VALUES
(9,9990,'AYUSH','TOP','VP','M','MARRIED',TO_DATE('15-12-2002','dd-mm-yyyy'));

1 row created.

SQL> insert into EMPLOYEE VALUES
(10,1190,'SHRISHTHI','TOP','VP','F','MARRIED',TO_DATE('15-12-2002','dd-mm-yyyy'));

1 row created.



SQL> create or replace view Employee_Records as select * from EMPLOYEE where
MARTIAL_ST like 'SINGLE';
create or replace view Employee_Records as select * from EMPLOYEE where MARTIAL_ST
like 'SINGLE'
                                                              *
ERROR at line 1:
ORA-00904: "MARTIAL_ST": invalid identifier


SQL> create or replace view Employee_Records as select * from EMPLOYEE where
Martial_status like 'SINGLE';

View created.

SQL> select * from Q5EMPLOYEE;

BUSINESS_ID   LOGIN_ID ORGANIZATION ORGANIZATI JOB_TITLE  G MARTIAL_ST
BIRTHDATE
----------- ---------- ---------- ---------- ---------- - ---------- ---------
```

```
         1       8080 yogesh     ENTRY      INTERN     M SINGLE     01-JAN-00
         2       5050 Suraj      ENTRY      INTERN     M SINGLE     27-JAN-00
         3       2850 Abhigyan   ENTRY      SDE        M SINGLE     27-JAN-10
         4       9850 Advait     ENTRY      SDE2       M MARRIED    15-MAR-22
         5       9890 Advait     MID        SDE2       M MARRIED    25-MAR-22
         6       8780 VAIBHAV    TOP        SDE3       M SINGLE     15-DEC-02
         7       9790 Yogesh     TOP        TEAM LEAD  M SINGLE     15-DEC-12
         8       9990 DIXIT      TOP        MANAGER    M SINGLE     15-DEC-02
         9       9990 AYUSH      TOP        VP         M MARRIED    15-DEC-02
        10       1190 SHRISHTHI  TOP        VP         F MARRIED    15-DEC-02

10 rows selected.

SQL> drop view Employee_Records;

View dropped.


SQL> as select * from EMPLOYEE where Gender = 'M';
SP2-0734: unknown command beginning "as select ..." - rest of line ignored.
SQL> create view Employee_Records
  2  as select * from EMPLOYEE where Gender = 'M';

View created.

SQL> INSERT INTO EMPLOYEE
VALUES(0012,5578,'CVB','SOP','ANALYIST','M','SINGLE',TO_DATE('02-07-1989','dd-mm-yy
yy'));

1 row created.

SQL> INSERT INTO Employee_records
VALUES(0012,5578,'CVB','SOP','ANALYIST','M','SINGLE',TO_DATE('02-07-1989','dd-mm-yy
yy'));

1 row created.

SQL> select * from  Employee_records;

BUSINESS_ID   LOGIN_ID ORGANIZATI ORGANIZATI JOB_TITLE  G MARTIAL_ST
BIRTHDATE
----------- ---------- ---------- ---------- ---------- - ---------- ---------
         1       8080 yogesh     ENTRY      INTERN     M SINGLE     01-JAN-00
         2       5050 Suraj      ENTRY      INTERN     M SINGLE     27-JAN-00
         3       2850 Abhigyan   ENTRY      SDE        M SINGLE     27-JAN-10
         4       9850 Advait     ENTRY      SDE2       M MARRIED    15-MAR-22
         5       9890 Advait     MID        SDE2       M MARRIED    25-MAR-22
         6       8780 VAIBHAV    TOP        SDE3       M SINGLE     15-DEC-02
         7       9790 Yogesh     TOP        TEAM LEAD  M SINGLE     15-DEC-12
         8       9990 DIXIT      TOP        MANAGER    M SINGLE     15-DEC-02
         9       9990 AYUSH      TOP        VP         M MARRIED    15-DEC-02
        12       5578 CVB        SOP        ANALYIST   M SINGLE     02-JUL-89
        12       5578 CVB        SOP        ANALYIST   M SINGLE     02-JUL-89
```

6. Store_Contacts(Business_Id, Store_Name, Contact type, First_Name, Last_Name);

```
SQL>

SQL>

SQL> create table store_contacts(bussiness_id int,store_name varchar(10),contact_type int,firstname
varchar(10),lastname varchar(10));


Table created.


SQL> insert into store_contacts VALUES(1,'ABC',91,'DIXIT','FADADU');


1 row created.


SQL> insert into store_contacts VALUES(2,'CDE',92,'yogesh','yadav');


1 row created.


SQL> insert into store_contacts VALUES(3,'DEF',93,'ABHIGYAN','SINGH');


1 row created.


SQL> insert into store_contacts VALUES(4,'EFG',94,'YOGESH','YADAV');


1 row created.


SQL> insert into store_contacts VALUES(5,'FGH',95,'VAIBHAV','JOSHI');


1 row created.


SQL> insert into store_contacts VALUES(6,'GHI',96,'SURAJ','YADAV');


1 row created.


SQL> CREATE VIEW STORE_VIEW AS SELECT * FROM store_contacts;


View created.
```

```
SQL> SELECT * FROM STORE_VIEW;

BUSSINESS_ID STORE_NAME CONTACT_TYPE FIRSTNAME  LASTNAME
------------ ---------- ------------ ---------- ----------
           1 ABC                  91 DIXIT      FADADU
           2 CDE                  92 yogesh     yadav
           3 DEF                  93 ABHIGYAN   SINGH
           4 EFG                  94 YOGESH     YADAV
           5 FGH                  95 VAIBHAV    JOSHI
           6 GHI                  96 SURAJ      YADAV

6 rows selected.

SQL> SELECT * FROM store_contacts;

BUSSINESS_ID STORE_NAME CONTACT_TYPE FIRSTNAME  LASTNAME
------------ ---------- ------------ ---------- ----------
           1 ABC                  91 DIXIT      FADADU
           2 CDE                  92 yogesh     yadav
           3 DEF                  93 ABHIGYAN   SINGH
           4 EFG                  94 YOGESH     YADAV
           5 FGH                  95 VAIBHAV    JOSHI
           6 GHI                  96 SURAJ      YADAV

6 rows selected.

SQL>
```

<u>Result:</u> Set Operators and View Queries were executed successfully for the given scenarios

NAME : Yogesh

REGISTER NUMBER :- RA1911003010569

Date: 23 March 2022

DBMS EXP – 8

Aim: To execute PL/SQL queries

Algorithm:

a. Open SQL client server
b. Write the PL/SQL queries for the given scenarios/questions
c. The PL/SQL queries are executed and output is recorded

1. Write a PL/SQL program which processes a bank transaction. Before allowing you to withdraw
   $500 from account 3, it makes sure the account has sufficient funds to cover the withdrawal. If the funds are available, the program debits the account. Otherwise, the program prints a message “insufficient funds”.

ACCOUNTS TABLE

ACCOUNT ID	ACCOUNT TYPE	ACC BALANCE	CC_HOLDER NAME
1	SAVINGS	1500	JAMES
2	CURRENT	300	JOHN
3	SAVINGS	3000	SMITH
4	SAVINGS	4000	ADAMS
5	CURRENT	5000	FORD

```
SQL> set serveroutput on

SQL> DECLARE
```

```
bal number;
BEGIN
INSERT INTO Accounts_Table values(1,'Savings',1500,'James');
INSERT INTO Accounts_Table values(2,'Current',300,'JOHN');
INSERT INTO Accounts_Table values(3,'Savings',3000,'SMITH');
INSERT INTO Accounts_Table values(4,'Savings',4000,'ADAMS');
INSERT INTO Accounts_Table values(5,'Current',5000,'FORD');
COMMIT;
 dbms_output.put_line('Values Inserted');
 Select bal into bal from Accounts_Table where account_id = 3;
 IF bal > 500 then
   dbms_output.put_line('New Balance'||(bal-500));
 ELSE
   dbms_output.put_line('Insufficient Funds');
 END IF;
 END;
 /
```

2. Write a PL/SQL program for finding the area of square, circle, and rectangle using a switch case.

```
SQL> set serveroutput on

SQL> DECLARE
  r integer := 10;
  s integer := 4;
  l integer := 1;
  5  b integer := 1;
  6  choice int;
  7  area long;
  8  begin
  9  choice := &choice;
 10  case choice
 11  when '0' then
 12  s := &s;
 13  dbms_output.put_line('Square: '||s*s);
 14  when '1' then
 15  r := &r;
 16  dbms_output.put_line('Circle: '||3.14*r*r);
 17  when '2' then
 18  l := &l;
 19  b := &b;
 20  dbms_output.put_line('Rectangle: '||l*b);
 21  else dbms_output.put_line('No such case!!');
 22  end case;
 23  end;
 24  /
Enter value for choice: 0
old   9: choice := &choice;
new   9: choice := 0;
Enter value for s: 10
```

```
old  12: s := &s;

new  12: s := 10;

Enter value for r: 100

old  15: r := &r;

new  15: r := 100;

Enter value for l: 100

old  18: l := &l;

new  18: l := 100;

Enter value for b: 50

old  19: b := &b;

new  19: b := 50;

Square: 100


PL/SQL procedure successfully completed.


SQL>
```

3. Write a PL/SQL program for finding the square roots of 1 to 25 using for loop.

```
SQL> set serveroutput on
SQL> DECLARE
      n number := &first_n_number;
     BEGIN
      DBMS_OUTPUT.PUT_LINE ('The square roots of '||n||' numbers are: ');
         for i in 1..n loop
```

```
  6          dbms_output.put(SQRT(i)||'  ');
  7        END LOOP;
  8          dbms_output.new_line;
  9     END;
 10     /
Enter value for first_n_number: 25
old   2:      n number := &first_n_number;
new   2:      n number := 25;
The square roots of 25 numbers are:
1  1.41421356237309504880168872420969807857
1.73205080756887729352744634150587236694  2
2.23606797749978969640917366873127623544
2.44948974278317809819728407470589139197
2.64575131106459059050161575363926042571
2.82842712474619009760337744841939615714  3
3.16227766016837933199889354443271853372
3.31662479035539984911493273667068668393
3.46410161513775458705489268301174473389
3.60555127546398929311922126747049594625
3.74165738677394138558374873231654930176
3.87298334620741688517926539978239961083  4
4.12310562561766054982140985597407702515
4.24264068711928514640506617262909423571
4.35889894354067355223698198385961565914
4.47213595499957939281834733746255247088
4.58257569495584000658804719372800848898
4.69041575982342955456563011354446628059
4.79583152331271954159743806416269392
4.89897948556635619639456814941178278393  5


PL/SQL procedure successfully completed.
```

<u>Result:</u> PL/SQL Queries were executed successfully for the given scenarios

Name : - yogesh   Date : 30 March 2022

Register Number :- RA1911003010569

Ex.no 9

Aim: -

Algorithm:

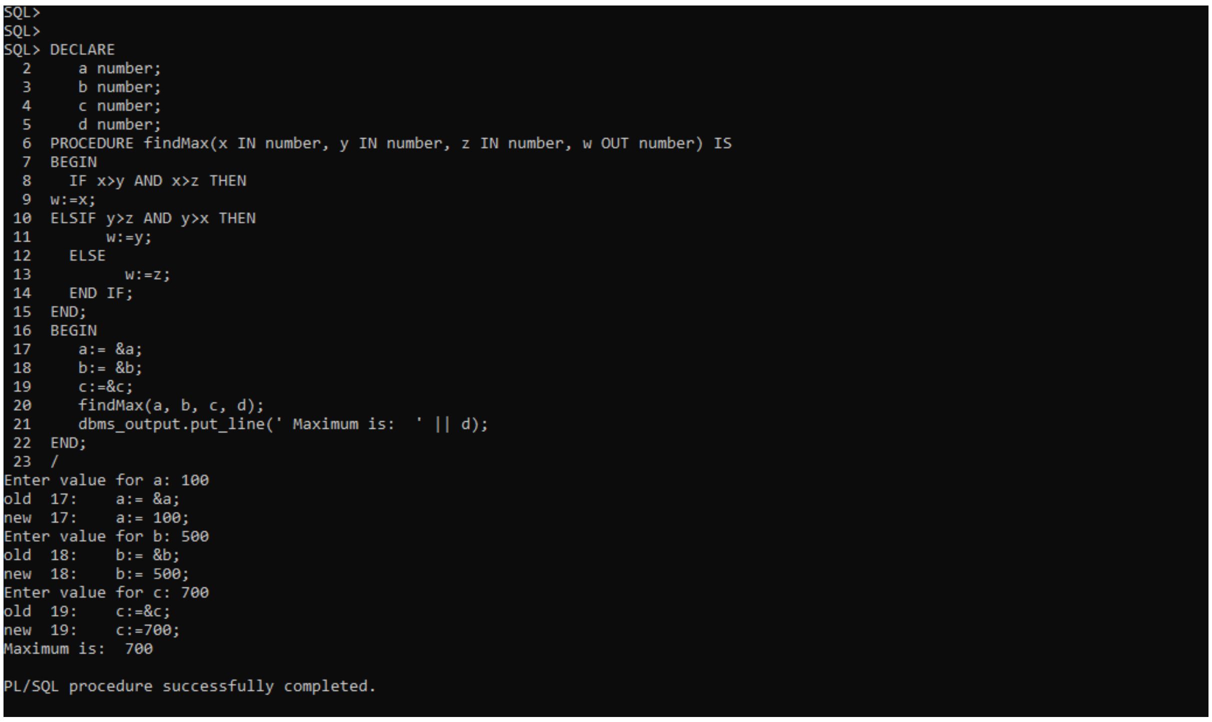
a. Open SQL client server
b. Write the PL/SQL code for the given scenarios/questions with procedures
c. The PL/SQL procedures are executed and output is recorded

Pl/sql procedure

1. Write a sql procedure program to find the largest of given three numbers.(Hint: A,B,C as IN parameter and Large as OUT parameter)

```
2.  SQL>
3.  SQL>
4.  SQL> DECLARE
5.    2    a number;
6.    3    b number;
7.    4    c number;
8.    5    d number;
9.    6  PROCEDURE findMax(x IN number, y IN number, z IN number, w OUT number)
    IS
10.   7  BEGIN
11.   8    IF x>y AND x>z THEN
12.   9  w:=x;
13.  10  ELSIF y>z AND y>x THEN
14.  11       w:=y;
15.  12    ELSE
16.  13         w:=z;
17.  14    END IF;
18.  15  END;
19.  16  BEGIN
20.  17    a:= &a;
21.  18    b:= &b;
22.  19    c:=&c;
23.  20    findMax(a, b, c, d);
24.  21    dbms_output.put_line(' Maximum is:  ' || d);
25.  22  END;
26.  23  /
27. Enter value for a: 100
28. old  17:    a:= &a;
29. new  17:    a:= 100;
30. Enter value for b: 500
31. old  18:    b:= &b;
32. new  18:    b:= 500;
33. Enter value for c: 700
34. old  19:    c:=&c;
```

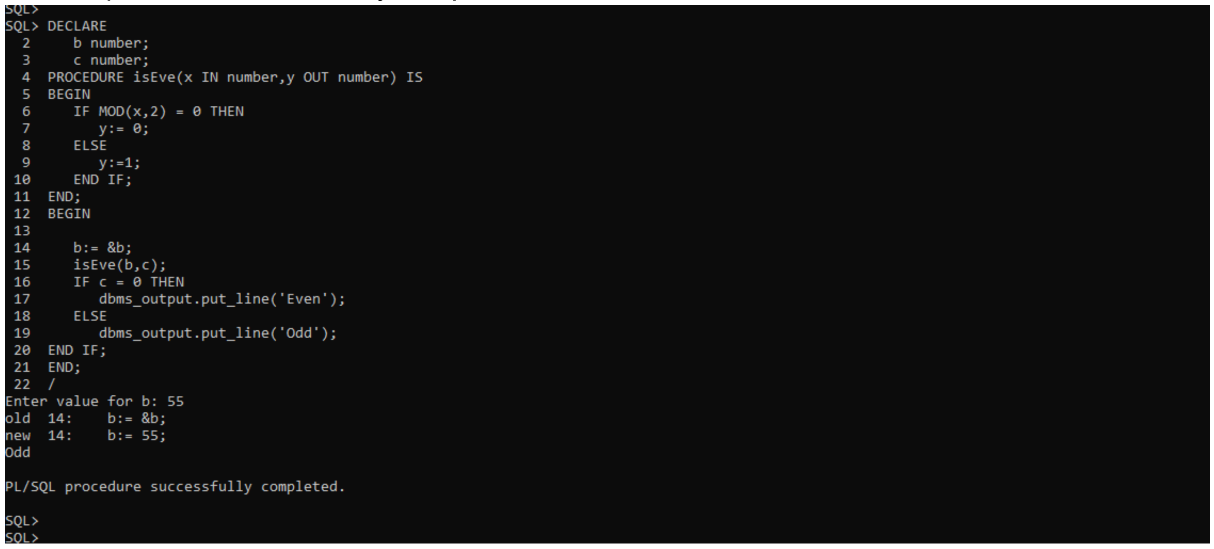
```
new  19:    c:=700;
Maximum is:  700

PL/SQL procedure successfully completed.

SQL>
SQL>
```



```
SQL>
SQL>
SQL> DECLARE
  2     a number;
  3     b number;
  4     c number;
  5     d number;
  6  PROCEDURE findMax(x IN number, y IN number, z IN number, w OUT number) IS
  7  BEGIN
  8    IF x>y AND x>z THEN
  9  w:=x;
 10  ELSIF y>z AND y>x THEN
 11        w:=y;
 12    ELSE
 13          w:=z;
 14    END IF;
 15  END;
 16  BEGIN
 17     a:= &a;
 18     b:= &b;
 19     c:=&c;
 20     findMax(a, b, c, d);
 21     dbms_output.put_line(' Maximum is:  ' || d);
 22  END;
 23  /
Enter value for a: 100
old  17:    a:= &a;
new  17:    a:= 100;
Enter value for b: 500
old  18:    b:= &b;
new  18:    b:= 500;
Enter value for c: 700
old  19:    c:=&c;
new  19:    c:=700;
Maximum is:  700

PL/SQL procedure successfully completed.
```

2.Write a sql procedure program to find the even or odd of a given number(Hint: Use A as IN OUT parameter)

```
SQL>
SQL>
SQL> DECLARE
  2    b number;
  3    c number;
  4  PROCEDURE isEve(x IN number,y OUT number) IS
  5  BEGIN
  6    IF MOD(x,2) = 0 THEN
  7      y:= 0;
  8    ELSE
  9      y:=1;
 10     END IF;
 11  END;
 12  BEGIN
 13
 14     b:= &b;
 15     isEve(b,c);
 16     IF c = 0 THEN
 17        dbms_output.put_line('Even');
 18     ELSE
 19        dbms_output.put_line('Odd');
 20  END IF;
 21  END;
 22  /
Enter value for b: 55
old  14:    b:= &b;
new  14:    b:= 55;
Odd

PL/SQL procedure successfully completed
```



```
SQL>
SQL> DECLARE
  2     b number;
  3     c number;
  4  PROCEDURE isEve(x IN number,y OUT number) IS
  5  BEGIN
  6     IF MOD(x,2) = 0 THEN
  7        y:= 0;
  8     ELSE
  9        y:=1;
 10     END IF;
 11  END;
 12  BEGIN
 13
 14     b:= &b;
 15     isEve(b,c);
 16     IF c = 0 THEN
 17        dbms_output.put_line('Even');
 18     ELSE
 19        dbms_output.put_line('Odd');
 20  END IF;
 21  END;
 22  /
Enter value for b: 55
old  14:    b:= &b;
new  14:    b:= 55;
Odd

PL/SQL procedure successfully completed.

SQL>
SQL>
```

Result: Procedures were executed successfully for the given scenarios

NAME : Yogesh

REGISTER NUMBER : RA1911003010569

Date : 30 March 2022

Ex.no 10

Aim: To execute PL/SQL Cursors

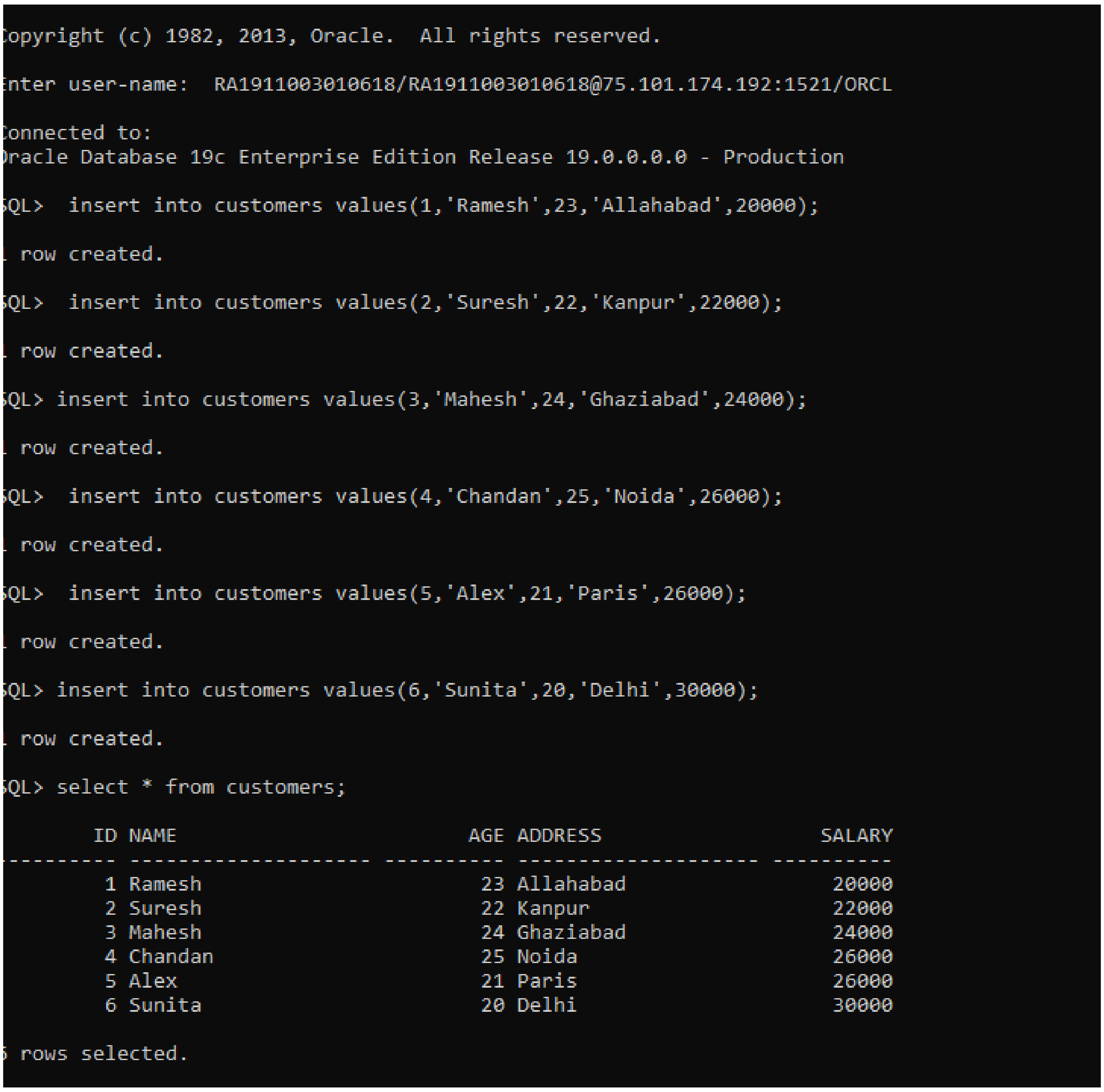
Algorithm:

a. Open SQL client server
b. Write the PL/SQL code for the given scenarios/questions with cursors
c. The PL/SQL codes are executed and output is recorded

Pl/sql cursor

1. Write a program to find the age of employees who are <=22 and increase the salary by 8000. Use sql%rowcount attribute to find the rows that got updated after execution.(Hint: implicit cursor)
2. Write a program to retrieve the employee name and address.(Hint: Explicit cursor)

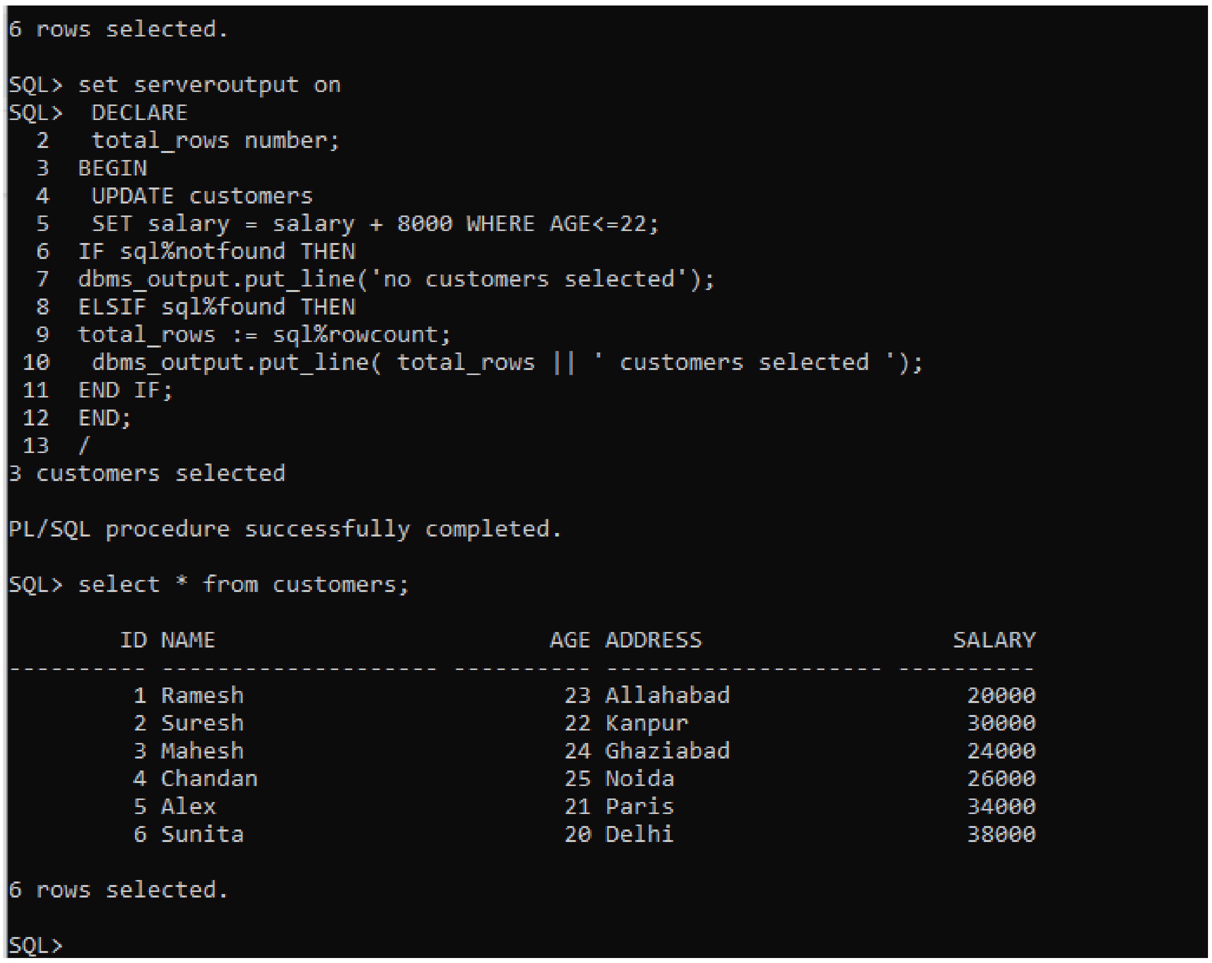
ID	NAME	AGE	ADDRESS	SALARY
1	Ramesh	23	Allahabad	20000
2	Suresh	22	Kanpur	22000
3	Mahesh	24	Ghaziabad	24000
4	Chandan	25	Noida	26000
5	Alex	21	Paris	28000
6	Sunita	20	Delhi	30000



```
opyright (c) 1982, 2013, Oracle.  All rights reserved.

nter user-name:  RA1911003010618/RA1911003010618@75.101.174.192:1521/ORCL

onnected to:
racle Database 19c Enterprise Edition Release 19.0.0.0.0 - Production

QL>  insert into customers values(1,'Ramesh',23,'Allahabad',20000);

 row created.

QL>  insert into customers values(2,'Suresh',22,'Kanpur',22000);

 row created.

QL> insert into customers values(3,'Mahesh',24,'Ghaziabad',24000);

 row created.

QL>  insert into customers values(4,'Chandan',25,'Noida',26000);

 row created.

QL>  insert into customers values(5,'Alex',21,'Paris',26000);

 row created.

QL> insert into customers values(6,'Sunita',20,'Delhi',30000);

 row created.

QL> select * from customers;

       ID NAME                                  AGE ADDRESS                  SALARY
---------- -------------------- ---------- -------------------- ----------
         1 Ramesh                                23 Allahabad                 20000
         2 Suresh                                22 Kanpur                    22000
         3 Mahesh                                24 Ghaziabad                 24000
         4 Chandan                               25 Noida                     26000
         5 Alex                                  21 Paris                     26000
         6 Sunita                                20 Delhi                     30000

 rows selected.
```

```
PL/SQL procedure successfully completed.

SQL> drop table customers;

Table dropped.

SQL>  Create table customers(id int, name varchar(20),age int,address varchar(20), salary int);

Table created.

SQL> insert into customers values(1,'Ramesh',23,'Allahabad',20000);

1 row created.

SQL>  insert into customers values(2,'Suresh',22,'Kanpur',22000);

1 row created.

SQL>  insert into customers values(3,'Mahesh',24,'Ghaziabad',24000);
```

```
1 row created.

SQL>  insert into customers values(4,'Chandan',25,'Noida',26000);

1 row created.

SQL>  insert into customers values(5,'Alex',21,'Paris',26000);

1 row created.

SQL>  insert into customers values(6,'Sunita',20,'Delhi',30000);

1 row created.


SQL> set serveroutput on
SQL>  DECLARE
c_name customers.name%type;
c_addr customers.address%type;
CURSOR c_customers is
        SELECT  name, address FROM customers;
    BEGIN
       OPEN c_customers;
       LOOP
      FETCH c_customers into c_name, c_addr;
          EXIT WHEN c_customers%notfound;
          dbms_output.put_line(c_name || ' ' || c_addr);
      END LOOP;
      CLOSE c_customers;
   END;
   /
   3 customers selected
```

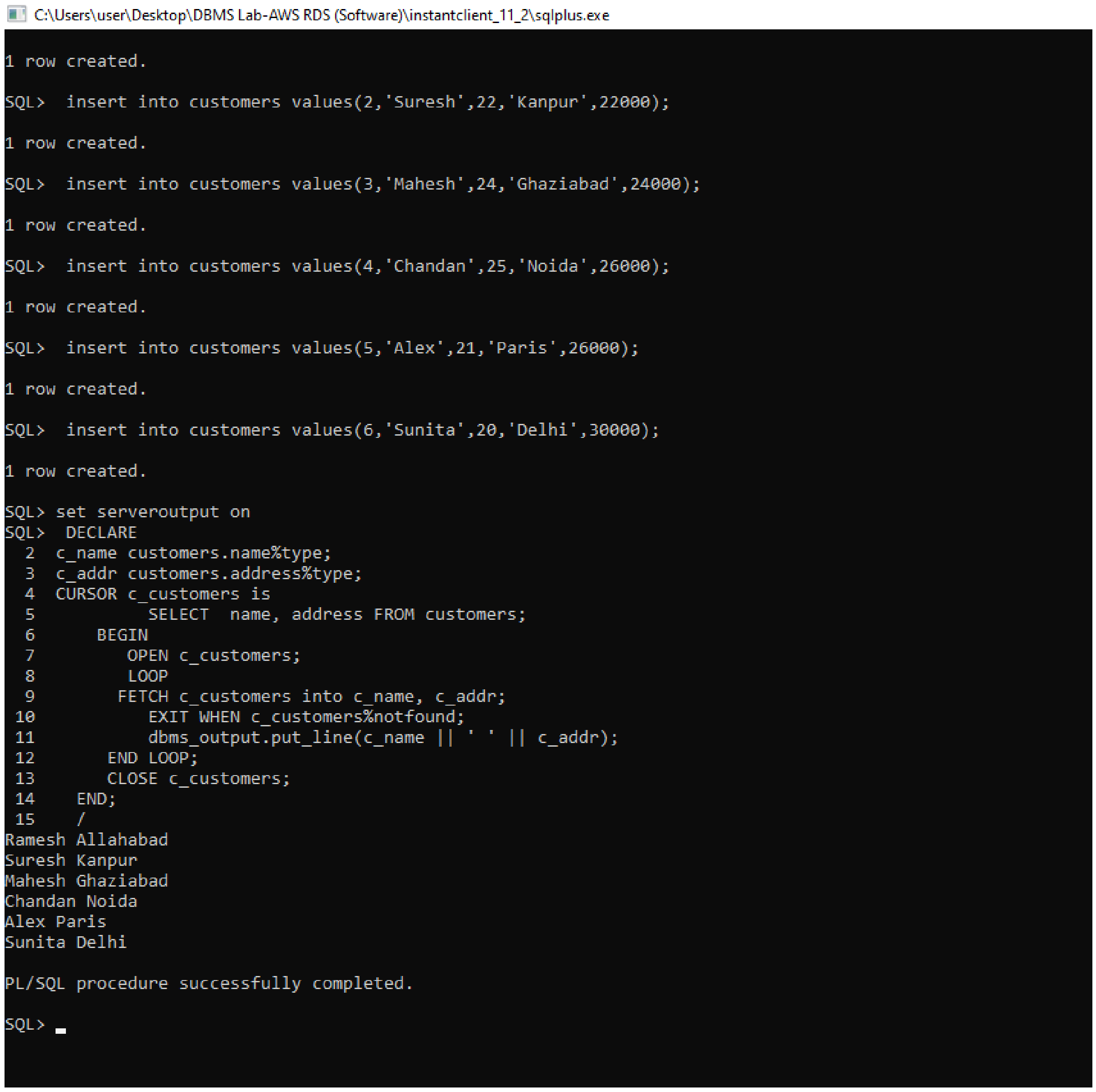


```
6 rows selected.

SQL> set serveroutput on
SQL>  DECLARE
  2   total_rows number;
  3  BEGIN
  4   UPDATE customers
  5   SET salary = salary + 8000 WHERE AGE<=22;
  6  IF sql%notfound THEN
  7  dbms_output.put_line('no customers selected');
  8  ELSIF sql%found THEN
  9  total_rows := sql%rowcount;
 10   dbms_output.put_line( total_rows || ' customers selected ');
 11  END IF;
 12  END;
 13  /
3 customers selected

PL/SQL procedure successfully completed.

SQL> select * from customers;

        ID NAME                        AGE ADDRESS                  SALARY
---------- -------------------- ---------- -------------------- ----------
         1 Ramesh                       23 Allahabad                 20000
         2 Suresh                       22 Kanpur                    30000
         3 Mahesh                       24 Ghaziabad                 24000
         4 Chandan                      25 Noida                     26000
         5 Alex                         21 Paris                     34000
         6 Sunita                       20 Delhi                     38000

6 rows selected.

SQL>
```

```
SQL> set serveroutput on
SQL>  DECLARE
  2  c_name customers.name%type;
  3  c_addr customers.address%type;
  4  CURSOR c_customers is
  5          SELECT  name, address FROM customers;
  6     BEGIN
  7        OPEN c_customers;
  8        LOOP
  9       FETCH c_customers into c_name, c_addr;
 10           EXIT WHEN c_customers%notfound;
 11           dbms_output.put_line(c_name || ' ' || c_addr);
 12       END LOOP;
 13       CLOSE c_customers;
 14    END;
 15    /
Ramesh Allahabad
Suresh Kanpur
Mahesh Ghaziabad
Chandan Noida
Alex Paris
Sunita Delhi
```

PL/SQL procedure successfully completed.

SQL>


C:\Users\user\Desktop\DBMS Lab-AWS RDS (Software)\instantclient_11_2\sqlplus.exe

```
1 row created.

SQL>  insert into customers values(2,'Suresh',22,'Kanpur',22000);

1 row created.

SQL>  insert into customers values(3,'Mahesh',24,'Ghaziabad',24000);

1 row created.

SQL>  insert into customers values(4,'Chandan',25,'Noida',26000);

1 row created.

SQL>  insert into customers values(5,'Alex',21,'Paris',26000);

1 row created.

SQL>  insert into customers values(6,'Sunita',20,'Delhi',30000);

1 row created.

SQL> set serveroutput on
SQL>  DECLARE
  2  c_name customers.name%type;
  3  c_addr customers.address%type;
  4  CURSOR c_customers is
  5           SELECT  name, address FROM customers;
  6      BEGIN
  7         OPEN c_customers;
  8         LOOP
  9        FETCH c_customers into c_name, c_addr;
 10           EXIT WHEN c_customers%notfound;
 11           dbms_output.put_line(c_name || ' ' || c_addr);
 12       END LOOP;
 13       CLOSE c_customers;
 14    END;
 15    /
Ramesh Allahabad
Suresh Kanpur
Mahesh Ghaziabad
Chandan Noida
Alex Paris
Sunita Delhi

PL/SQL procedure successfully completed.

SQL> _
```

NAME: yogesh Date : 6 April 2022

REGISTER NUMBER: RA1911003010569

Ex.no 11 PL/SQL Functions

Aim: To execute PL/SQL Functions

Algorithm:

a. Open SQL client server
b. Write the PL/SQL code for the given scenarios/questions with functions
c. The PL/SQL codes are executed and output is recorded

ID	NAME	AGE	ADDRESS	SALARY
1	Ramesh	32	Ahmedabad	2000.00
2	Khilan	25	Delhi	1500.00
3	kaushik	23	Kota	2000.00
4	Chaitali	25	Mumbai	6500.00
5	Hardik	27	Bhopal	8500.00
6	Komal	22	MP	4500.00

1. Write a pl/sql function program that finds the total sum of salary of customers whose salary is

greater than 4000.00

```
create table emp(ID int,NAME varchar(20),AGE int,ADDRESS VARCHAR(20),SALARY int);


Table created.
SQL> insert into emp values(1,'Ramesh',32,'Ahmedabad',2000.00);


1 row created.


SQL> insert into emp values(2,'Khilan',25,'Delhi',1500.00);
```

```
1 row created.

SQL>  insert into emp values(3,'Kaushik',23,'Kota',2000.00);

1 row created.

SQL> insert into emp values(4,'Chaitali',25,'Mumbai',6500.00);

1 row created.

SQL> insert into employees values(5,'Hardik',27,'Bhopal',8500.00);

1 row created.

SQL> insert into emp values(5,'Hardik',27,'Bhopal',8500.00);

1 row created.

SQL> insert into emp values(6,'Komal',22,'MP',4500.00);

1 row created.

SQL> select * from emp;

        ID NAME                        AGE ADDRESS                  SALARY
---------- -------------------- ---------- -------------------- ----------
         1 Ramesh                       32 Ahmedabad                  2000
         2 Khilan                    25 Delhi                  1500
         3 Kaushik                    23 Kota                   2000
         4 Chaitali                  25 Mumbai                   6500
         5 Hardik                    27 Bhopal                  8500
         6 Komal                      22 MP                     4500
```

```
6 rows selected.

SQL> set serveroutput on
SQL> CREATE OR REPLACE FUNCTION sumSal
  2     RETURN number is
  3     total number:=0;
  4     begin
  5        select sum(salary) into total from emp where salary>4000;
  6          return total;
  7     end;
  8     /

Function created.

SQL> declare
  2     c number;
  3     begin
  4     c:=sumSal();
  5     dbms_output.put_line(c);
  6     end;
  7     /
19500
```

2. Write a pl/sql function program to calculate the sum of first natural numbers

```
SQL> set serveroutput on
SQL> DECLARE
  2  X NUMBER;
  3  N NUMBER;
  4  FUNCTION FINDMAX(N IN NUMBER)
  5    RETURN NUMBER
  6    IS
```

```
 7    Z NUMBER;
 8  BEGIN
 9    Z:=(N*(N+1))/2;
10      RETURN Z;
11           END;
12           BEGIN
13          N:= &N;
14           X:=FINDMAX(N);
15           dbms_output.Put_line('Sum:' || X);
16           END;
17       /
Enter value for n: 10
old  13:         N:= &N;
new  13:         N:= 10;
Sum:55

PL/SQL procedure successfully completed.
```

NAME: Yogesh Date : 6 April 2022

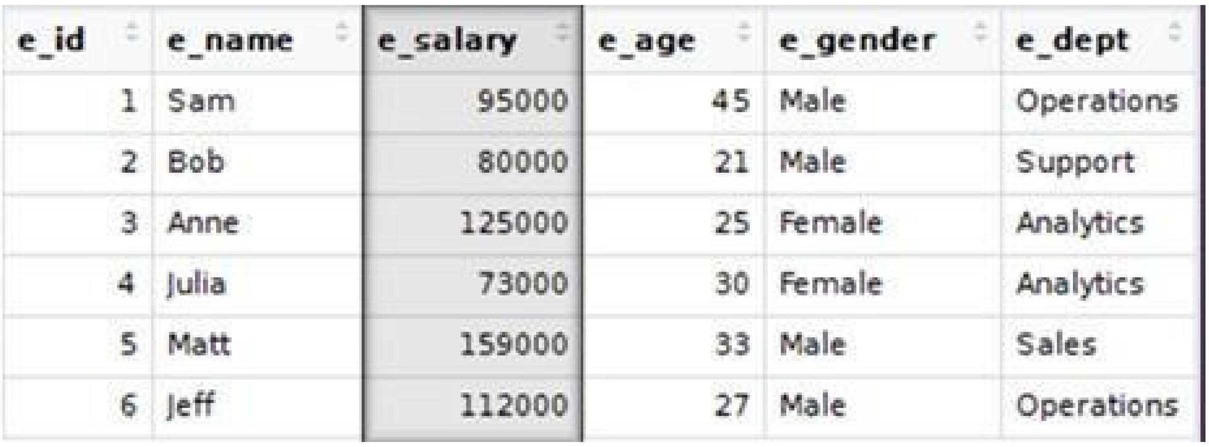
REGISTER NUMBER: RA1911003010569

Ex.no 12PL/SQL Triggers

Aim: To execute PL/SQL Functions

Algorithm:

a. Open SQL client server
b. Write the PL/SQL code for the given scenarios/questions with functions
c. The PL/SQL codes are executed and output is recorded



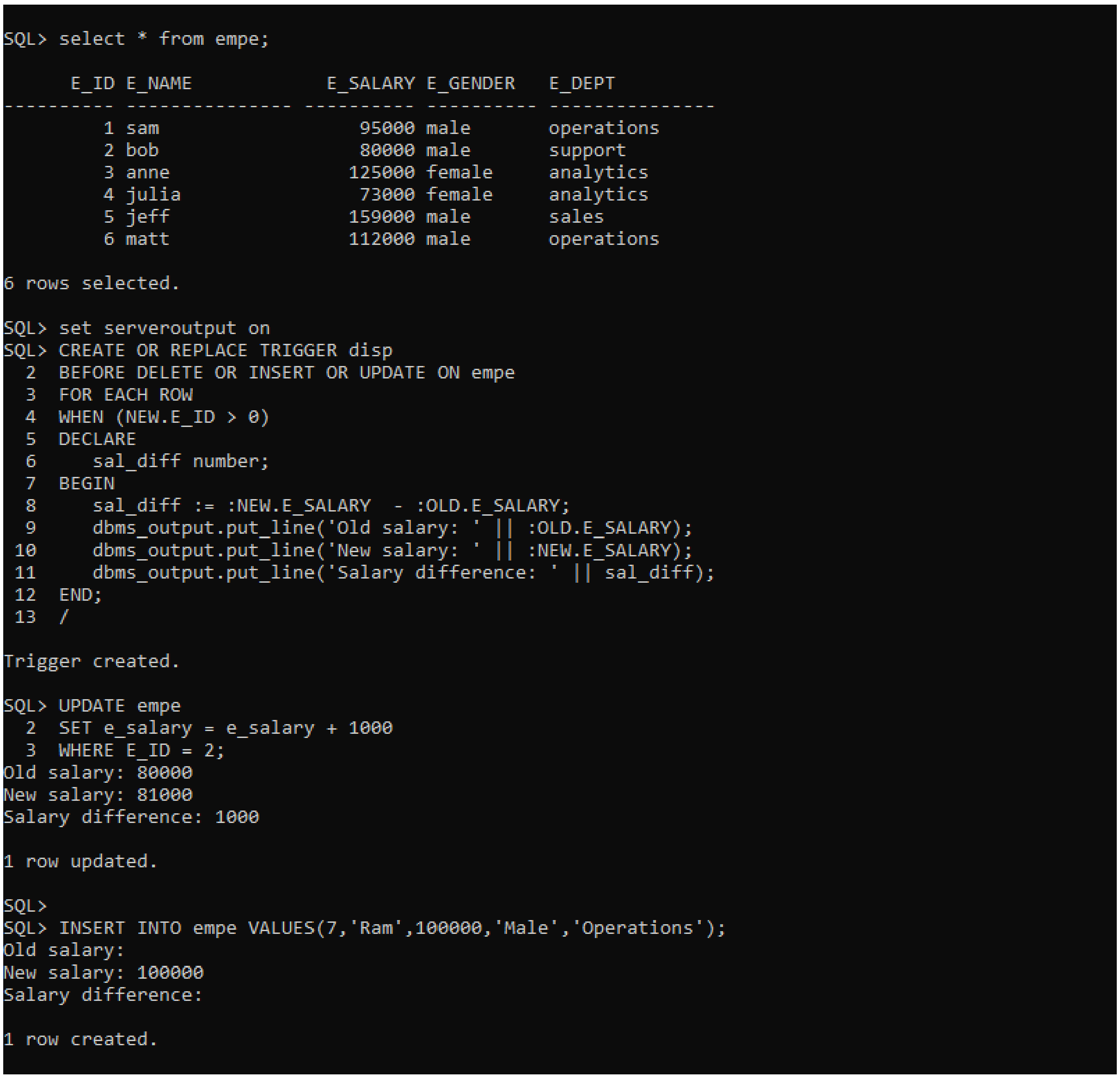
e_id	e_name	e_salary	e_age	e_gender	e_dept
1	Sam	95000	45	Male	Operations
2	Bob	80000	21	Male	Support
3	Anne	125000	25	Female	Analytics
4	Julia	73000	30	Female	Analytics
5	Matt	159000	33	Male	Sales
6	Jeff	112000	27	Male	Operations

1. Create a row-level trigger for the EMPLOYEE table that would get executed by the DML statement like UPDATE OR INSERT on that table.
2. The trigger will compute and show the SALARY difference between current and previous values.(Hint: previous salary:_______, current salary:________, salary difference:_)

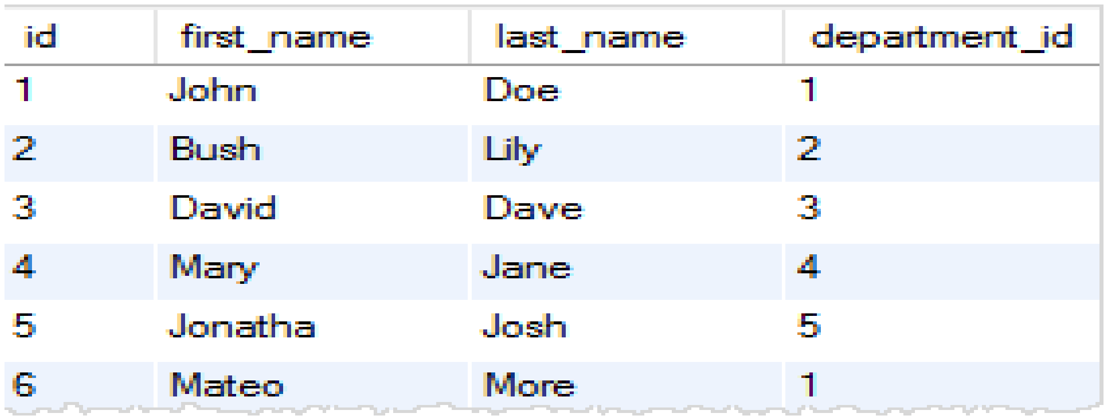
```
Enter user-name:
RA1911003010569/RA1911003010569@75.101.174.192:1521/ORCL

Connected to:

Oracle Database 19c Enterprise Edition Release 19.0.0.0.0 - Production
```

```
SQL> select * from empe;


     E_ID E_NAME            E_SALARY E_GENDER   E_DEPT
---------- --------------- ---------- ---------- ---------------
        1 sam                  95000 male       operations
        2 bob                  80000 male       support
        3 anne                125000 female     analytics
        4 julia                73000 female     analytics
        5 jeff                159000 male       sales
        6 matt                112000 male       operations


6 rows selected.


SQL> set serveroutput on
SQL> CREATE OR REPLACE TRIGGER disp
  2  BEFORE DELETE OR INSERT OR UPDATE ON empe
  3  FOR EACH ROW
  4  WHEN (NEW.E_ID > 0)
  5  DECLARE
  6    sal_diff number;
  7  BEGIN
  8    sal_diff := :NEW.E_SALARY  - :OLD.E_SALARY;
  9    dbms_output.put_line('Old salary: ' || :OLD.E_SALARY);
 10    dbms_output.put_line('New salary: ' || :NEW.E_SALARY);
 11    dbms_output.put_line('Salary difference: ' || sal_diff);
 12  END;
 13  /
```

```
Trigger created.


SQL> UPDATE empe

  2  SET e_salary = e_salary + 1000

  3  WHERE E_ID = 2;

Old salary: 80000

New salary: 81000

Salary difference: 1000


1 row updated.


SQL>

SQL> INSERT INTO empe VALUES(7,'Ram',100000,'Male','Operations');

Old salary:

New salary: 100000

Salary difference:


1 row created.
```



```
SQL> select * from empe;

      E_ID E_NAME            E_SALARY E_GENDER   E_DEPT
---------- --------------- ---------- ---------- ---------------
         1 sam                  95000 male       operations
         2 bob                  80000 male       support
         3 anne                125000 female     analytics
         4 julia                73000 female     analytics
         5 jeff                159000 male       sales
         6 matt                112000 male       operations

6 rows selected.

SQL> set serveroutput on
SQL> CREATE OR REPLACE TRIGGER disp
  2  BEFORE DELETE OR INSERT OR UPDATE ON empe
  3  FOR EACH ROW
  4  WHEN (NEW.E_ID > 0)
  5  DECLARE
  6     sal_diff number;
  7  BEGIN
  8     sal_diff := :NEW.E_SALARY  - :OLD.E_SALARY;
  9     dbms_output.put_line('Old salary: ' || :OLD.E_SALARY);
 10     dbms_output.put_line('New salary: ' || :NEW.E_SALARY);
 11     dbms_output.put_line('Salary difference: ' || sal_diff);
 12  END;
 13  /

Trigger created.

SQL> UPDATE empe
  2  SET e_salary = e_salary + 1000
  3  WHERE E_ID = 2;
Old salary: 80000
New salary: 81000
Salary difference: 1000

1 row updated.

SQL>
SQL> INSERT INTO empe VALUES(7,'Ram',100000,'Male','Operations');
Old salary:
New salary: 100000
Salary difference:

1 row created.
```



id	first_name	last_name	department_id
1	John	Doe	1
2	Bush	Lily	2
3	David	Dave	3
4	Mary	Jane	4
5	Jonatha	Josh	5
6	Mateo	More	1

1. Create a trigger for the STUDENT table that would get executed by

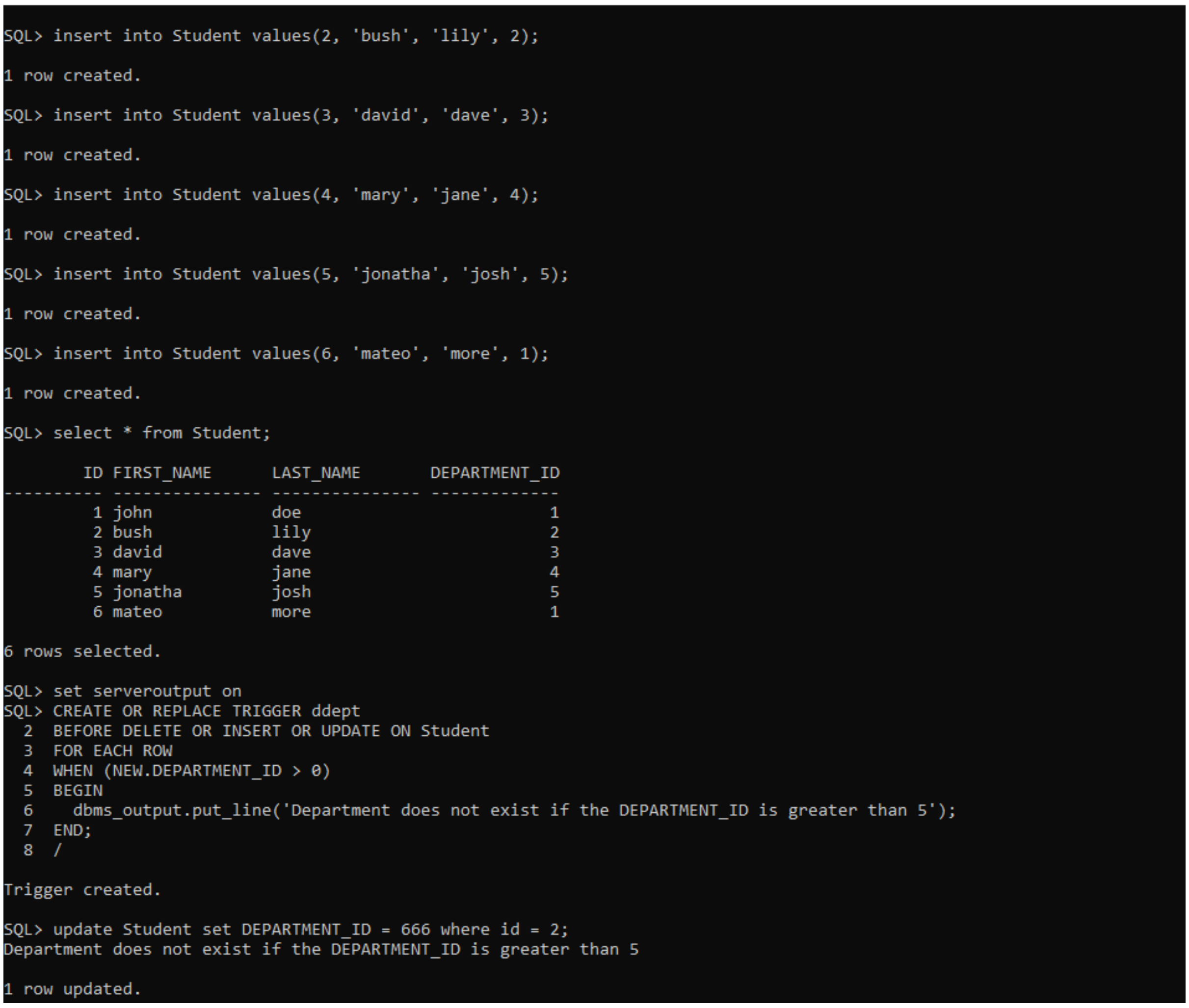
the DML statement like UPDATE OR INSERT on that table.

2. The trigger will compute and show the message “Department does not exist if the department_ id is greater than 5”.

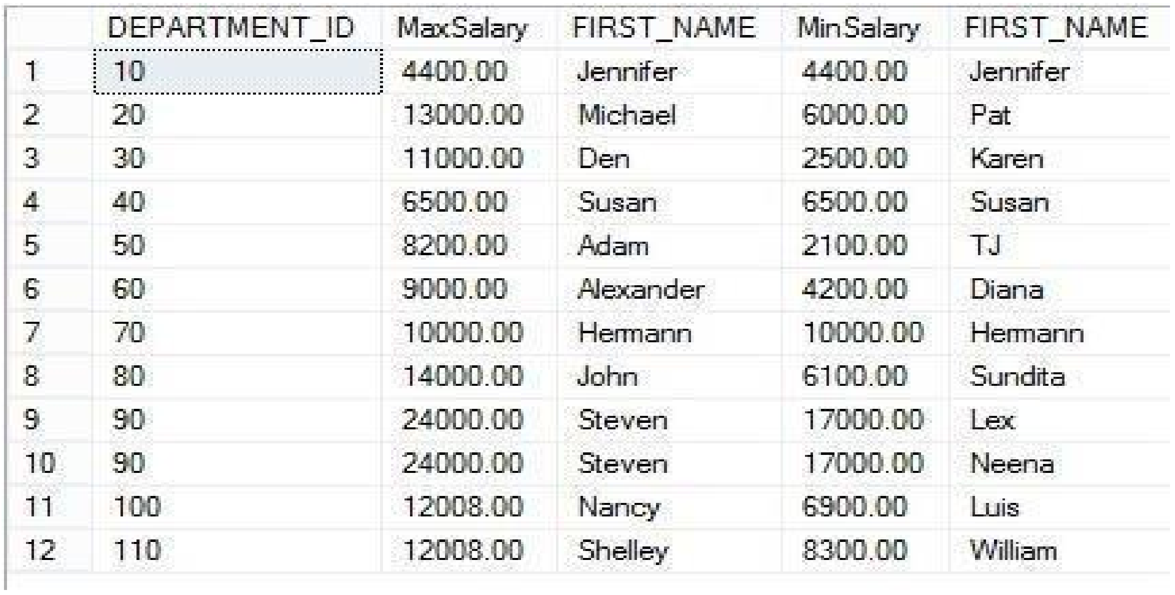
```
SQL>

SQL> create table Student(id int, First_Name char(15), Last_Name char(15),
Department_id int);


Table created.


SQL> insert into Student values(1, 'john', 'doe', 1);


1 row created.


SQL> insert into Student values(2, 'bush', 'lily', 2);


1 row created.


SQL> insert into Student values(3, 'david', 'dave', 3);


1 row created.


SQL> insert into Student values(4, 'mary', 'jane', 4);


1 row created.


SQL> insert into Student values(5, 'jonatha', 'josh', 5);
```

```
1 row created.


SQL> insert into Student values(6, 'mateo', 'more', 1);


1 row created.


SQL> select * from Student;


        ID FIRST_NAME      LAST_NAME       DEPARTMENT_ID
---------- --------------- --------------- -------------
         1 john            doe                         1
         2 bush            lily                        2
         3 david           dave                        3
         4 mary            jane                        4
         5 jonatha         josh                        5
         6 mateo           more                        1


6 rows selected.


SQL> set serveroutput on
SQL> CREATE OR REPLACE TRIGGER ddept
  2  BEFORE DELETE OR INSERT OR UPDATE ON Student
  3  FOR EACH ROW
  4  WHEN (NEW.DEPARTMENT_ID > 0)
  5  BEGIN
  6    dbms_output.put_line('Department does not exist if the DEPARTMENT_ID is
greater than 5');
```

```
  7  END;

  8  /


Trigger created.


SQL> update Student set DEPARTMENT_ID = 666 where id = 2;

Department does not exist if the DEPARTMENT_ID is greater than 5


1 row updated.
```



```
SQL> insert into Student values(2, 'bush', 'lily', 2);

1 row created.

SQL> insert into Student values(3, 'david', 'dave', 3);

1 row created.

SQL> insert into Student values(4, 'mary', 'jane', 4);

1 row created.

SQL> insert into Student values(5, 'jonatha', 'josh', 5);

1 row created.

SQL> insert into Student values(6, 'mateo', 'more', 1);

1 row created.

SQL> select * from Student;

        ID FIRST_NAME      LAST_NAME       DEPARTMENT_ID
---------- --------------- --------------- -------------
         1 john            doe                         1
         2 bush            lily                        2
         3 david           dave                        3
         4 mary            jane                        4
         5 jonatha         josh                        5
         6 mateo           more                        1

6 rows selected.

SQL> set serveroutput on
SQL> CREATE OR REPLACE TRIGGER ddept
  2  BEFORE DELETE OR INSERT OR UPDATE ON Student
  3  FOR EACH ROW
  4  WHEN (NEW.DEPARTMENT_ID > 0)
  5  BEGIN
  6    dbms_output.put_line('Department does not exist if the DEPARTMENT_ID is greater than 5');
  7  END;
  8  /

Trigger created.

SQL> update Student set DEPARTMENT_ID = 666 where id = 2;
Department does not exist if the DEPARTMENT_ID is greater than 5

1 row updated.
```



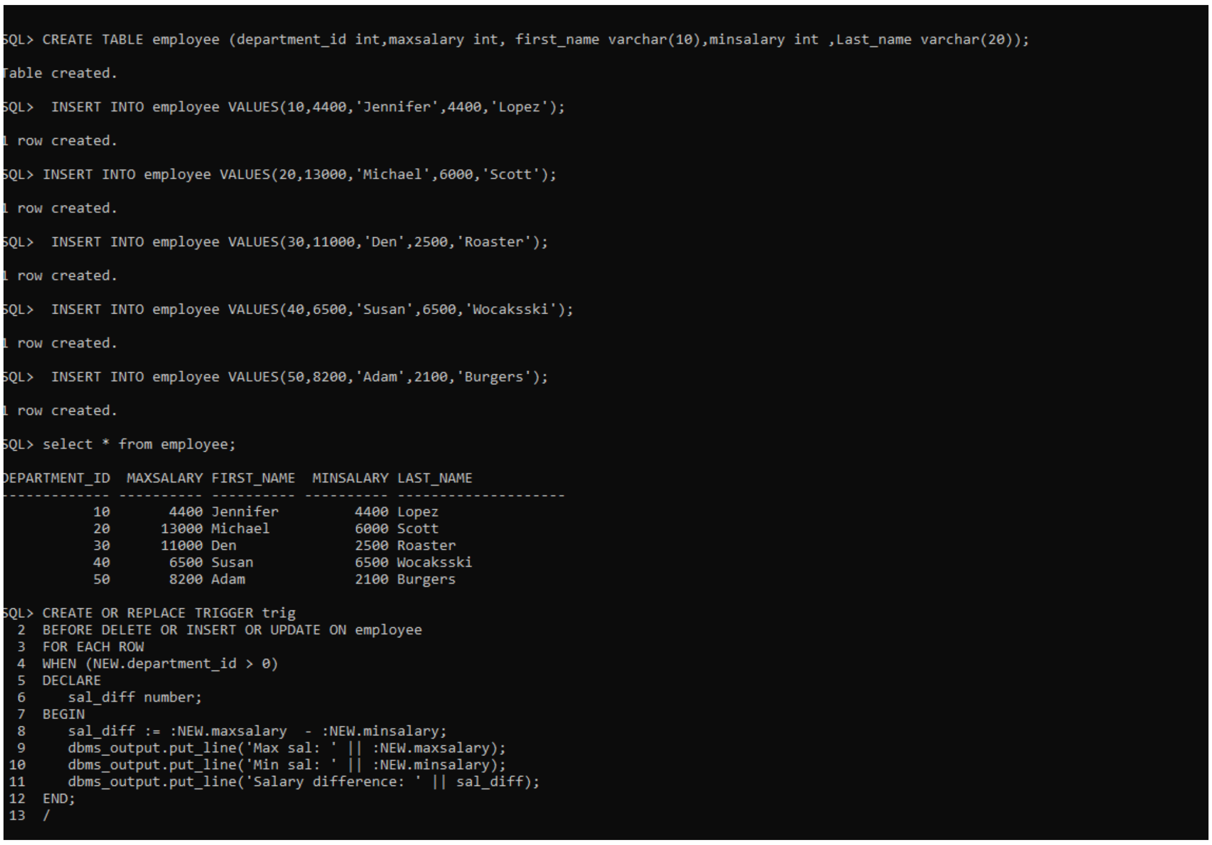
	DEPARTMENT_ID	MaxSalary	FIRST_NAME	MinSalary	FIRST_NAME
1	10	4400.00	Jennifer	4400.00	Jennifer
2	20	13000.00	Michael	6000.00	Pat
3	30	11000.00	Den	2500.00	Karen
4	40	6500.00	Susan	6500.00	Susan
5	50	8200.00	Adam	2100.00	TJ
6	60	9000.00	Alexander	4200.00	Diana
7	70	10000.00	Hermann	10000.00	Hermann
8	80	14000.00	John	6100.00	Sundita
9	90	24000.00	Steven	17000.00	Lex
10	90	24000.00	Steven	17000.00	Neena
11	100	12008.00	Nancy	6900.00	Luis
12	110	12008.00	Shelley	8300.00	William

1. Create a trigger for the EMPLOYEE table that would get executed by the DML statement like UPDATE OR INSERT on that table.
2. The trigger will compute and show the difference between the min salary and max salary.

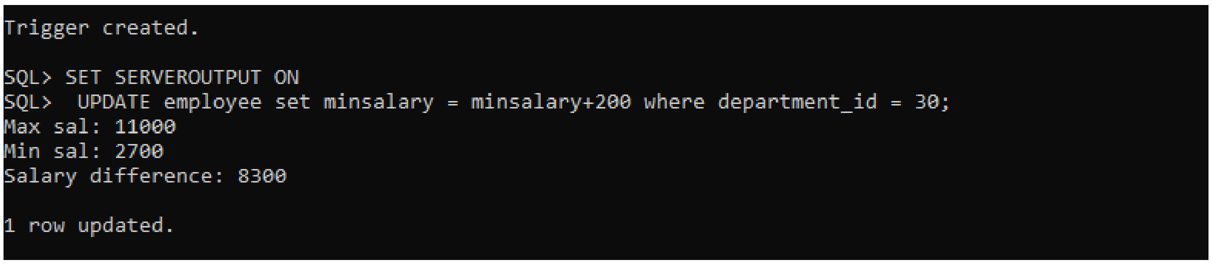
```
SQL> CREATE TABLE employee (department_id int,maxsalary int, first_name varchar(10),minsalary
int ,Last_name varchar(20));

Table created.

SQL>  INSERT INTO employee VALUES(10,4400,'Jennifer',4400,'Lopez');

1 row created.

SQL> INSERT INTO employee VALUES(20,13000,'Michael',6000,'Scott');

1 row created.

SQL>  INSERT INTO employee VALUES(30,11000,'Den',2500,'Roaster');
```

```
1 row created.

SQL>  INSERT INTO employee VALUES(40,6500,'Susan',6500,'Wocaksski');

1 row created.

SQL>  INSERT INTO employee VALUES(50,8200,'Adam',2100,'Burgers');

1 row created.

SQL> select * from employee;

DEPARTMENT_ID  MAXSALARY FIRST_NAME  MINSALARY LAST_NAME
------------- ---------- ---------- ---------- --------------------
           10       4400 Jennifer         4400 Lopez
           20      13000 Michael          6000 Scott
           30      11000 Den              2500 Roaster
           40       6500 Susan            6500 Wocaksski
           50       8200 Adam             2100 Burgers


SQL> CREATE OR REPLACE TRIGGER trig
  2  BEFORE DELETE OR INSERT OR UPDATE ON employee
  3  FOR EACH ROW
  4  WHEN (NEW.department_id > 0)
  5  DECLARE
  6     sal_diff number;
  7  BEGIN
  8     sal_diff := :NEW.maxsalary  - :NEW.minsalary;
  9     dbms_output.put_line('Max sal: ' || :NEW.maxsalary);
 10     dbms_output.put_line('Min sal: ' || :NEW.minsalary);
 11     dbms_output.put_line('Salary difference: ' || sal_diff);
 12  END;
 13  /
```

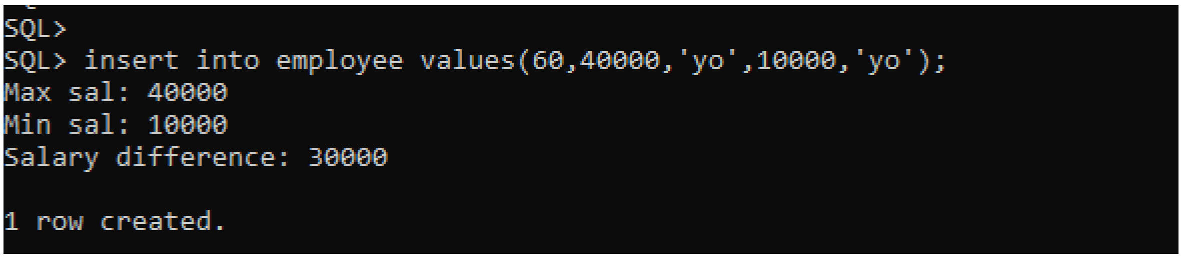
```
Trigger created.

SQL> SET SERVEROUTPUT ON
SQL>  UPDATE employee set minsalary = minsalary+200 where department_id = 30;
Max sal: 11000
Min sal: 2700
Salary difference: 8300

1 row updated.


SQL>
SQL> insert into employee values(60,40000,'yo',10000,'yo');
Max sal: 40000
Min sal: 10000
Salary difference: 30000

1 row created.
```



```
SQL> CREATE TABLE employee (department_id int,maxsalary int, first_name varchar(10),minsalary int ,Last_name varchar(20));

Table created.

SQL>  INSERT INTO employee VALUES(10,4400,'Jennifer',4400,'Lopez');

1 row created.

SQL> INSERT INTO employee VALUES(20,13000,'Michael',6000,'Scott');

1 row created.

SQL>  INSERT INTO employee VALUES(30,11000,'Den',2500,'Roaster');

1 row created.

SQL>  INSERT INTO employee VALUES(40,6500,'Susan',6500,'Wocaksski');

1 row created.

SQL>  INSERT INTO employee VALUES(50,8200,'Adam',2100,'Burgers');

1 row created.

SQL> select * from employee;

DEPARTMENT_ID  MAXSALARY FIRST_NAME  MINSALARY LAST_NAME
------------- ---------- ---------- ---------- --------------------
           10       4400 Jennifer         4400 Lopez
           20      13000 Michael          6000 Scott
           30      11000 Den              2500 Roaster
           40       6500 Susan            6500 Wocaksski
           50       8200 Adam             2100 Burgers

SQL> CREATE OR REPLACE TRIGGER trig
BEFORE DELETE OR INSERT OR UPDATE ON employee
FOR EACH ROW
WHEN (NEW.department_id > 0)
DECLARE
   sal_diff number;
BEGIN
   sal_diff := :NEW.maxsalary  - :NEW.minsalary;
   dbms_output.put_line('Max sal: ' || :NEW.maxsalary);
   dbms_output.put_line('Min sal: ' || :NEW.minsalary);
   dbms_output.put_line('Salary difference: ' || sal_diff);
END;
/
```



```
Trigger created.

SQL> SET SERVEROUTPUT ON
SQL>  UPDATE employee set minsalary = minsalary+200 where department_id = 30;
Max sal: 11000
Min sal: 2700
Salary difference: 8300

1 row updated.
```



```
SQL>
SQL> insert into employee values(60,40000,'yo',10000,'yo');
Max sal: 40000
Min sal: 10000
Salary difference: 30000

1 row created.
```

Result: PL/SQL Triggers were executed successfully for the given scenarios