MALAD KANDIVALI EDUCATION SOCIETY'S

NAGINDAS KHANDWALA COLLEGE OF COMMERCE, ARTS & MANAGEMENT STUDIES & SHANTABEN NAGINDAS KHANDWALA COLLEGE OF SCIENCE

MALAD [W], MUMBAI – 64

AUTONOMOUS INSTITUTION

(Affiliated To University Of Mumbai)

Reaccredited 'A' Grade by NAAC | ISO 9001:2015 Certified

CERTIFICATE

Name: Aniket Ramesh Prajapati

Roll No: 351 Programme: BSc - CS Semester: III

This is certified to be a bonafide record of practical works done by the above student in the college laboratory for the course **DATABASE MANAGEMENT SYSTEM – II** for the partial fulfilment of Third Semester of BSc IT/CS during the academic year 2021-22.

The journal work is the original study work that has been duly approved in the year 2021-22 by the undersigned.

External Examiner

Ms. Sweety Garg
(Subject-In-Charge)

Date of Examination: (College Stamp)

INDEX

DBMS PRACTICAL NO: -1

1.Write a PL/SQL block to display the message “hello world”.

```
Run SQL Command Line

SQL*Plus: Release 11.2.0.2.0 Production on Thu Jul 15 16:11:11 2021

Copyright (c) 1982, 2014, Oracle.  All rights reserved.

SQL> connect system
Enter password:
Connected.
SQL> set  serveroutput on
SQL> begin
  2  dbms_output.put_line('hello world');
  3  end;
  4  /
hello world
```

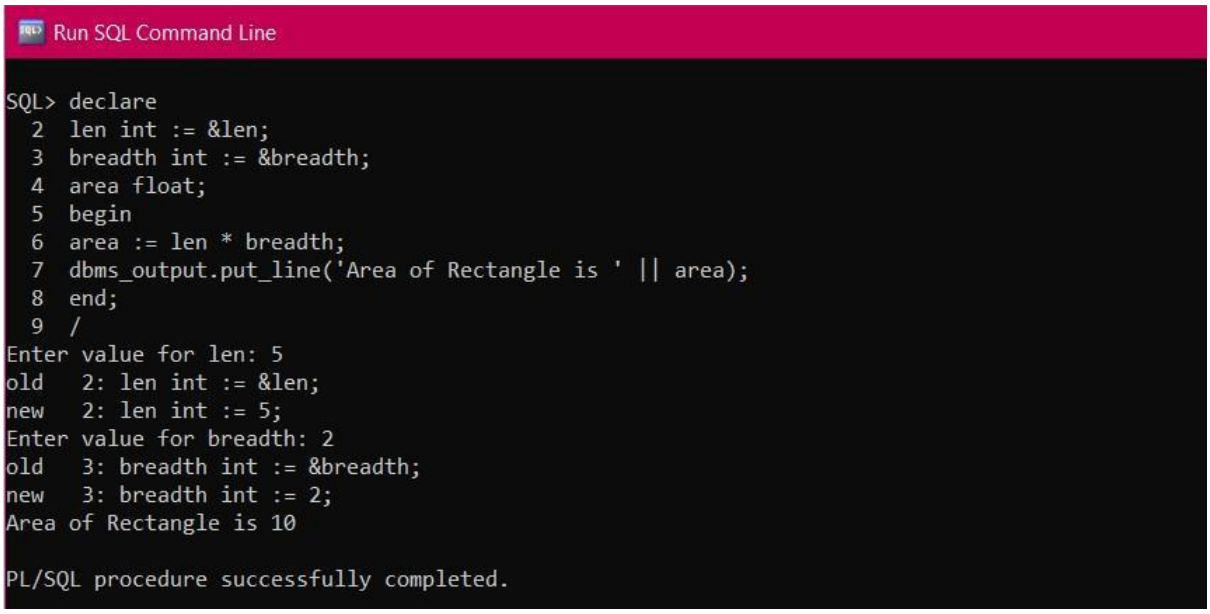
2. Write a PL/SQL block which will read a number from the user and display it on the screen.

```
SQL> declare
  2  m int := &m;
  3  begin
  4  dbms_output.put_line(m);
  5  end;
  6  /
Enter value for m: 25
old   2: m int := &m;
new   2: m int := 25;
25

PL/SQL procedure successfully completed.
```

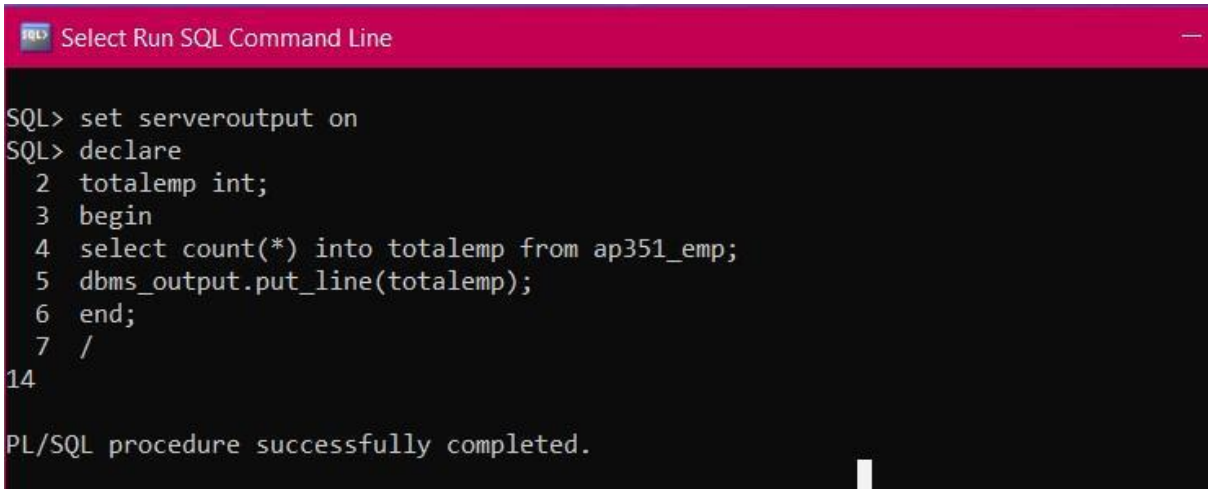
3. Write a PL/SQL block to read a message from user and display it.

```
Run SQL Command Line

SQL> declare
  2  a varchar(20) := '&a';
  3  begin
  4  dbms_output.put_line(a);
  5  end;
  6  /
Enter value for a: Hey Aniket
old   2: a varchar(20) := '&a';
new   2: a varchar(20) := 'Hey Aniket';
Hey Aniket

PL/SQL procedure successfully completed.
```

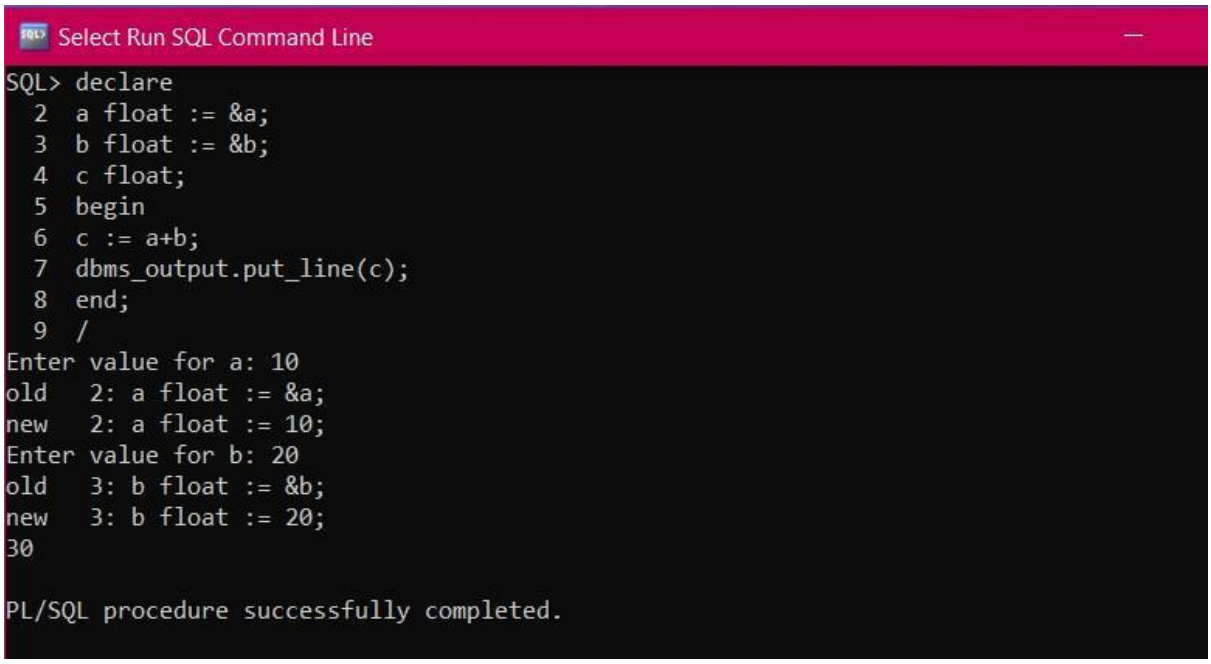
4. Write a PL/SQL block to display the area of a rectangle when length and breadth are accepted by the user.



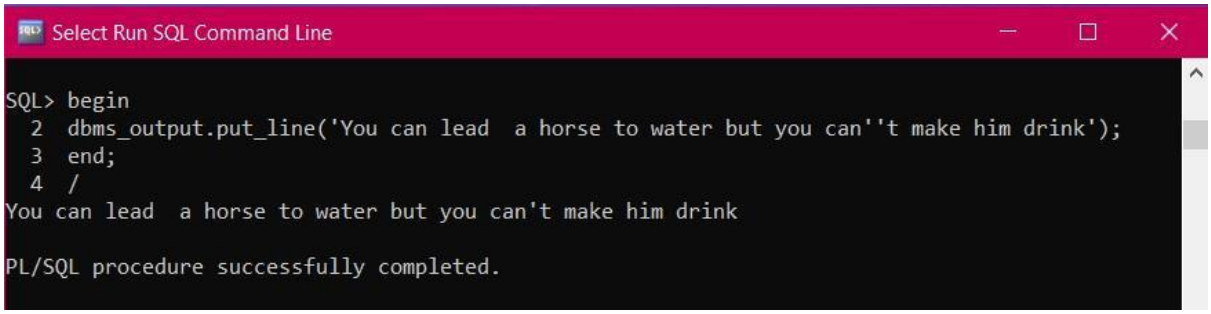
5. Write a PL/SQL block to display the total number of employees.



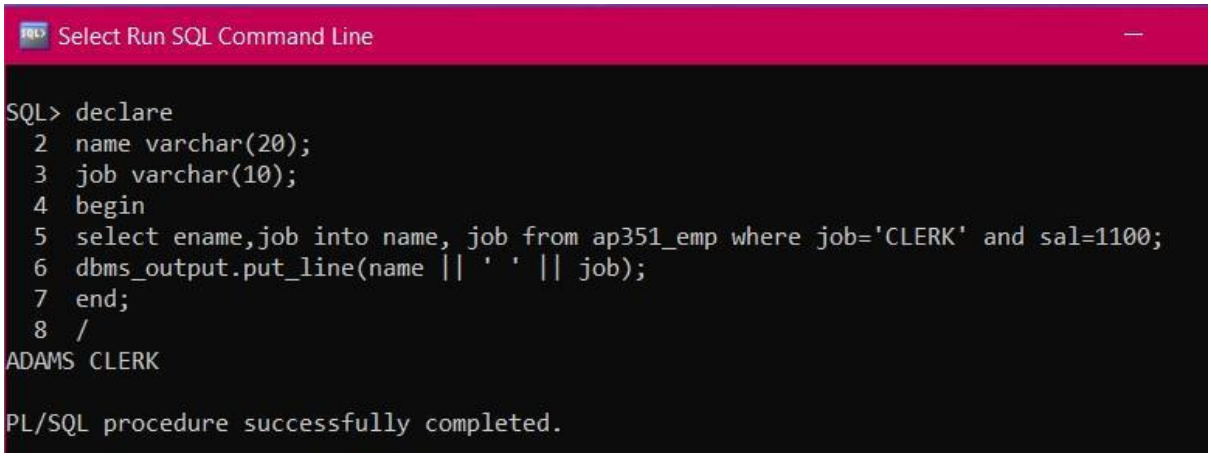
6. Write a PL/SQL block to print the sum of two numbers accepted by user.



7. Write a PL/SQL block to print the message ‘You can lead a horse to water but you can't make him drink’.



8. Write a PL/SQL block to print the name and job of an employee who is working as CLERK earning salary of Rs 1100/-.



9. Write a PL/SQL block to calculate Simple Interest where principle, rate and time are accepted by the user.

Select Run SQL Command Line

```
SQL> declare
  2  a float;
  3  principal float := &principal;
  4  rate float := &rate;
  5  time float := &time;
  6  begin
  7  a := (principal*rate*time)/100;
  8  dbms_output.put_line('simple interest =' || a);
  9  end;
 10  /
Enter value for principal: 5000
old   3: principal float := &principal;
new   3: principal float := 5000;
Enter value for rate: 5
old   4: rate float := &rate;
new   4: rate float := 5;
Enter value for time: 2
old   5: time float := &time;
new   5: time float := 2;
simple interest =500

PL/SQL procedure successfully completed.
```

10. Write a PL/SQL block to calculate the area of the circle and store the radius and area in a table AOC (radius, area).

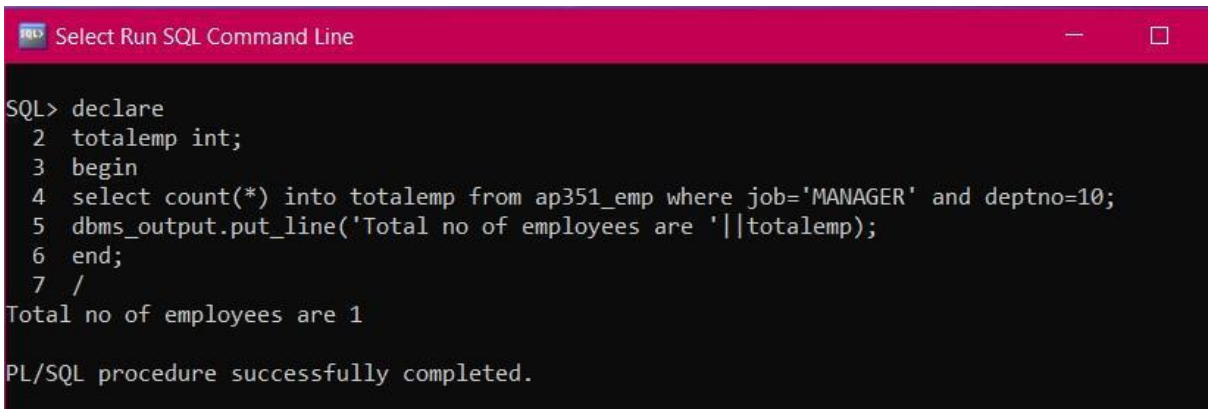
Select Run SQL Command Line

```
SQL> declare
  2  r int := &r;
  3  a float;
  4  begin
  5  a := 3.14*r*r;
  6  insert into AOC values(r,a);
  7  end;
  8  /
Enter value for r: 5
old   2: r int := &r;
new   2: r int := 5;
insert into AOC values(r,a);
```

11. Write a PL/SQL block to print the salary and job of the employee whose employee number is entered by the user using percent type attribute.

```
SQL> declare
  2  eno EMP.EMPNO%type :=&eno;
  3  salary EMP.SAL%type;
  4  job EMP.JOB%type;
  5  begin
  6  select sal into salary from EMP where empno = eno;
  7  select job into job from EMP where empno = eno;
  8  dbms_output.put_line('JOB:' || job);
  9  dbms_output.put_line('SALARY:' || salary);
 10  end;
 11  /
Enter value for eno: 5
old   2: eno EMP.EMPNO%type :=&eno;
new   2: eno EMP.EMPNO%type :=5;
```

12. Write a PL/SQL block to print the total number of employees working as Manager in deptno 10.



13. Write a PL/SQL block to print the total salary of the employees from the employee table

```
Select Run SQL Command Line
SQL> declare
  2  totalsal int;
  3  begin
  4  select sum(sal) into totalsal from ap351_emp;
  5  dbms_output.put_line('Total no of employees is '||totalsal);
  6  end;
  7  /
Total no of employees is 29025

PL/SQL procedure successfully completed.
```

14. Write a PL/SQL block to find the cube of a number.

```
Select Run SQL Command Line

SQL> declare
  2  cube float;
  3  a int :=5;
  4  begin
  5  cube := a*a*a;
  6  dbms_output.put_line('cube is = '||cube);
  7  end;
  8  /
cube is = 125

PL/SQL procedure successfully completed.
```

15. Write a block to print the message "I'm a user".

```
Select Run SQL Command Line

SQL> begin
  2  dbms_output.put_line('i''m a user');
  3  end;
  4  /
i'm a user

PL/SQL procedure successfully completed.

SQL>
```

PRACTICAL NO: -2

1. Write a PL/SQL program to display the sum of first 10 integers.

```
Run SQL Command Line

SQL*Plus: Release 11.2.0.2.0 Production on Mon Aug 9 17:24:11 2021

Copyright (c) 1982, 2014, Oracle.  All rights reserved.

SQL> connect system
Enter password:
Connected.
SQL> declare
  2  a int;
  3  b int :=0;
  4  begin
  5  for a in 1..10 loop
  6  b :=b+a;
  7  end loop;
  8  dbms_output.put_line(b);
  9  end;
 10  /

PL/SQL procedure successfully completed.
```

2. Write a PL/SQL program to display the sum of first 10 odd numbers.

```
Run SQL Command Line

SQL> declare
  2  a number :=0;
  3  b number :=1;
  4  c number :=0;
  5  begin
  6  while a<10 loop
  7  dbms_output.put_line(b);
  8  c :=b+c;
  9  b :=b+2;
 10  a :=a+1;
 11  end loop;
 12  dbms_output.put_line(c);
 13  end;
 14  /
1
3
5
7
9
11
13
15
17
19
100

PL/SQL procedure successfully completed.
```

3. Write a PL/SQL program to calculate the area of a circle and insert the area and radius in a table aoc (sno,radius,area) till radius is less than 10.

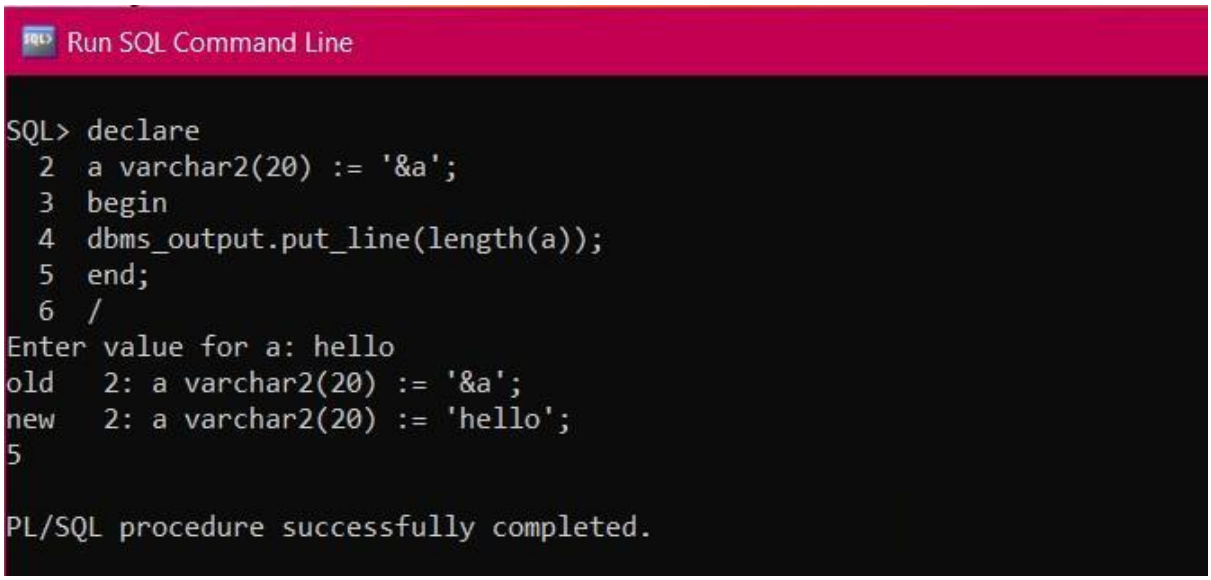
```
SQL> declare
  2  a number;
  3  radius number :=1;
  4  area number;
  5  pi number :=3.14;
  6  begin
  7  for a in 1..9 loop
  8  radius := a;
  9  area := radius * radius * pi;
 10  insert into aco values(a,radius,area);
 11  end loop;
 12  end;
 13  /

PL/SQL procedure successfully completed.
```

4. Write a PL/SQL program to reverse the number (234 as 432).

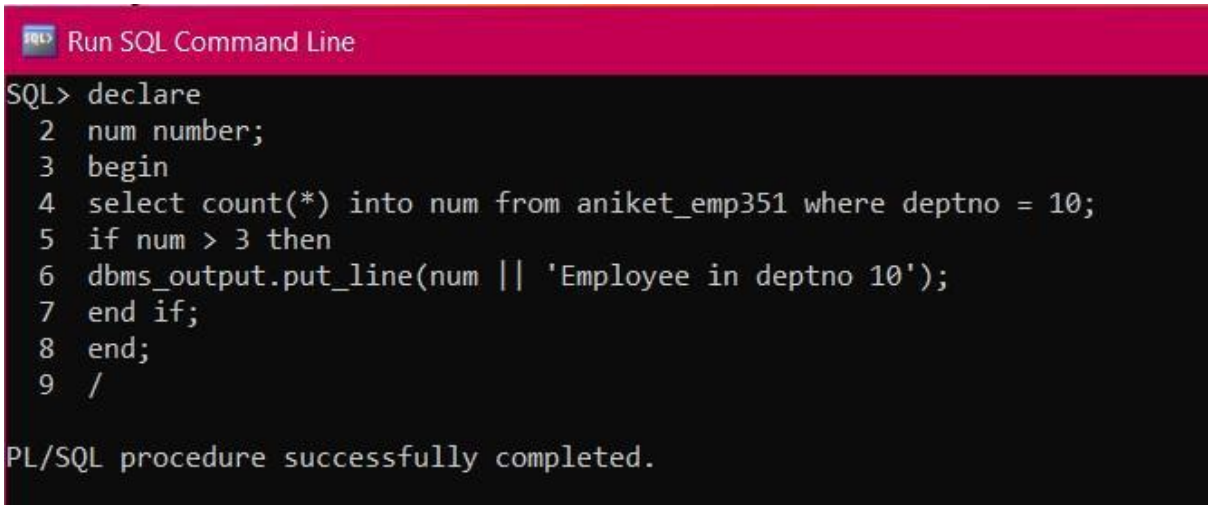
Run SQL Command Line

```
SQL> declare
  2  a number :=234;
  3  b number;
  4  begin
  5  b :=0;
  6  while a>0 loop
  7  b :=(b * 10) + mod(a,10);
  8  a :=floor(a/10);
  9  end loop;
 10  dbms_output.put_line(b);
 11  end;
 12  /
432

PL/SQL procedure successfully completed.
```

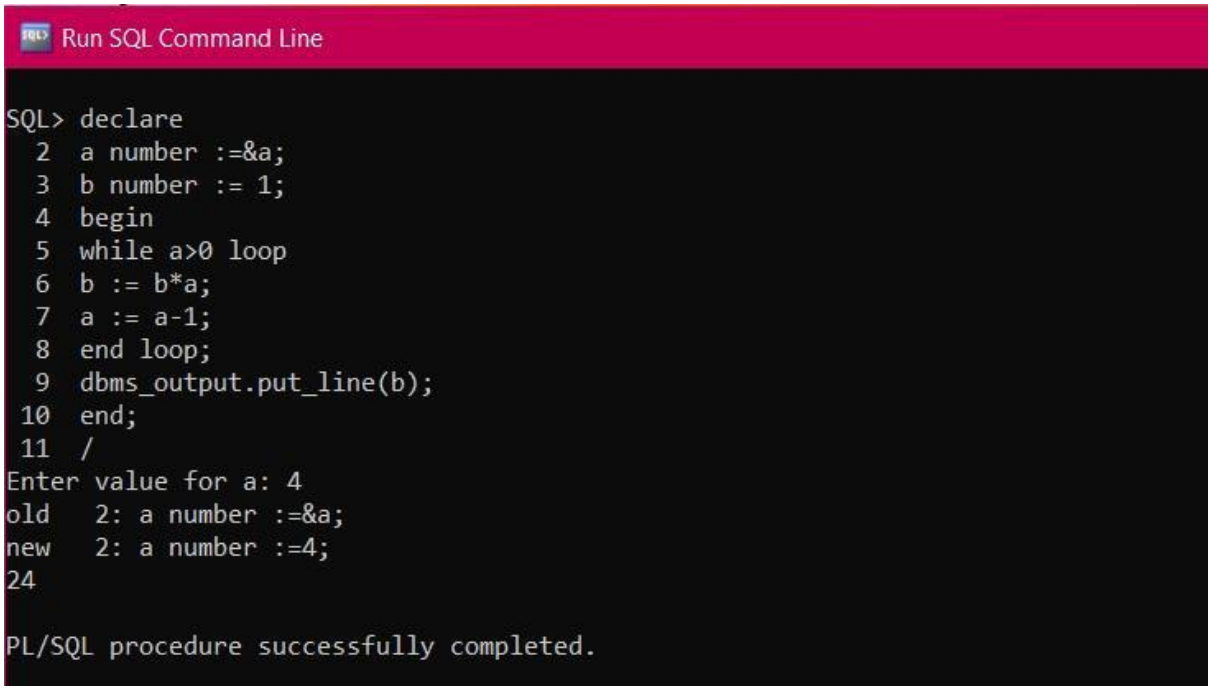
5. Write a PL/SQL program to print the length of entered string.



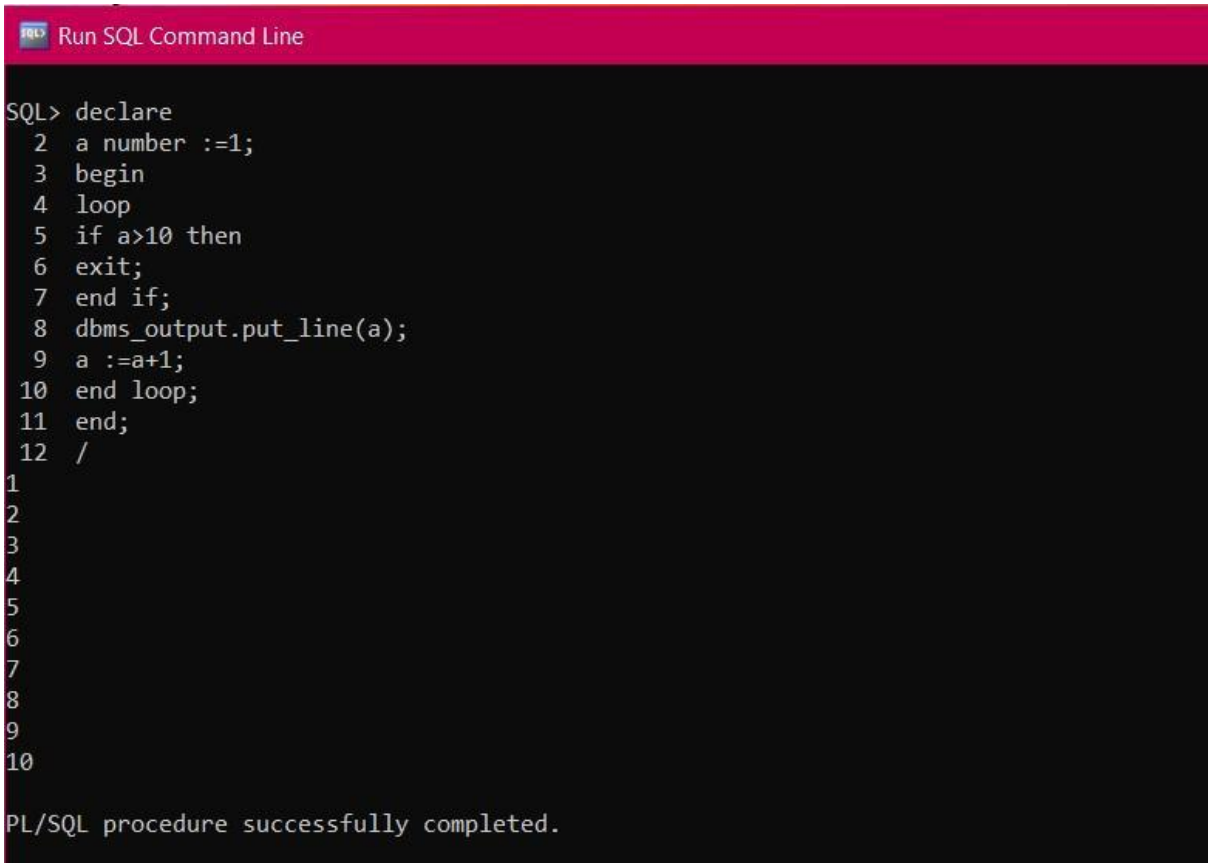
6. Write a PL/SQL program to count number of employees in dept 10, and if they are greater than 3 print the count otherwise do nothing.



7. Write a PL/SQL block to find the factorial of a number.



8. Write a block to display the number from 1 to 10 using unconstraint loop.

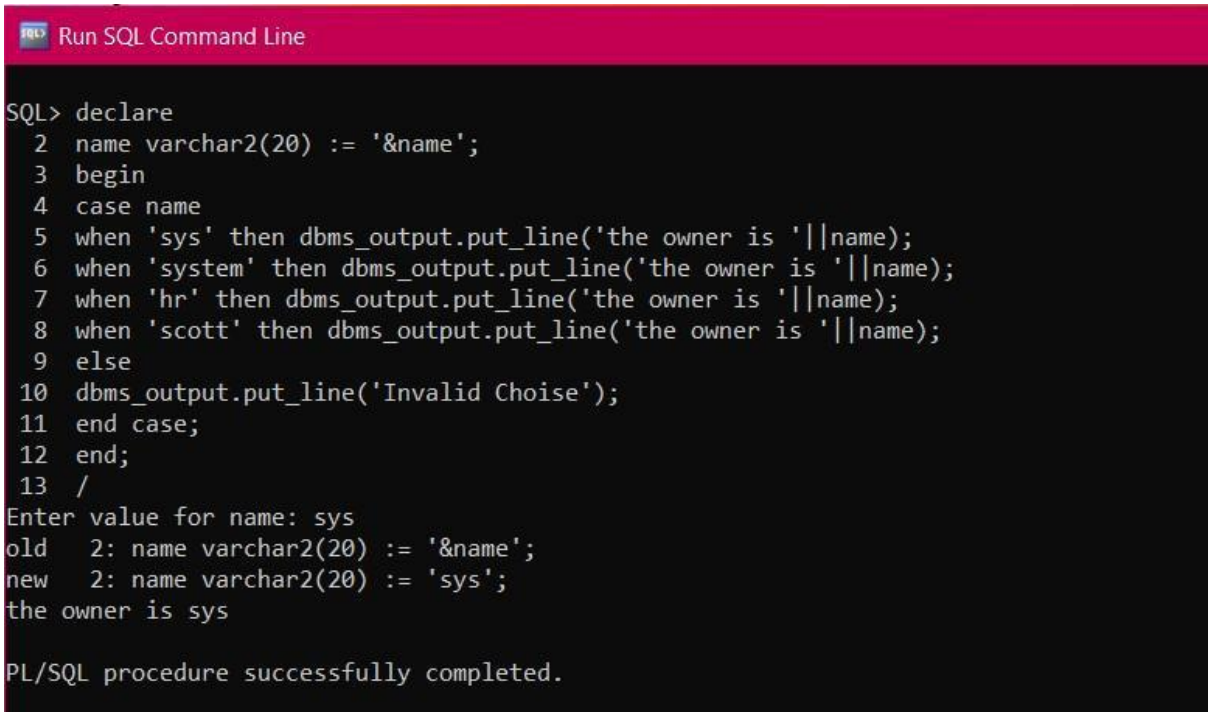


9. Write a PL/SQL block using CASE statement to accept the owner name from the user.

The user name can be SYS, SYSTEM, HR or SCOTT.

If the owner name is SYS then print the result is ‘The Owner is SYS’. If the owner name is SYSTEM then print the result is ‘The Owner is SYSTEM’.

If the owner name is HR then print the result is ‘The Owner is HR’. If the owner name is SCOTT then print the result is ‘The Owner is SCOTT’. Otherwise print ‘Invalid Choice’



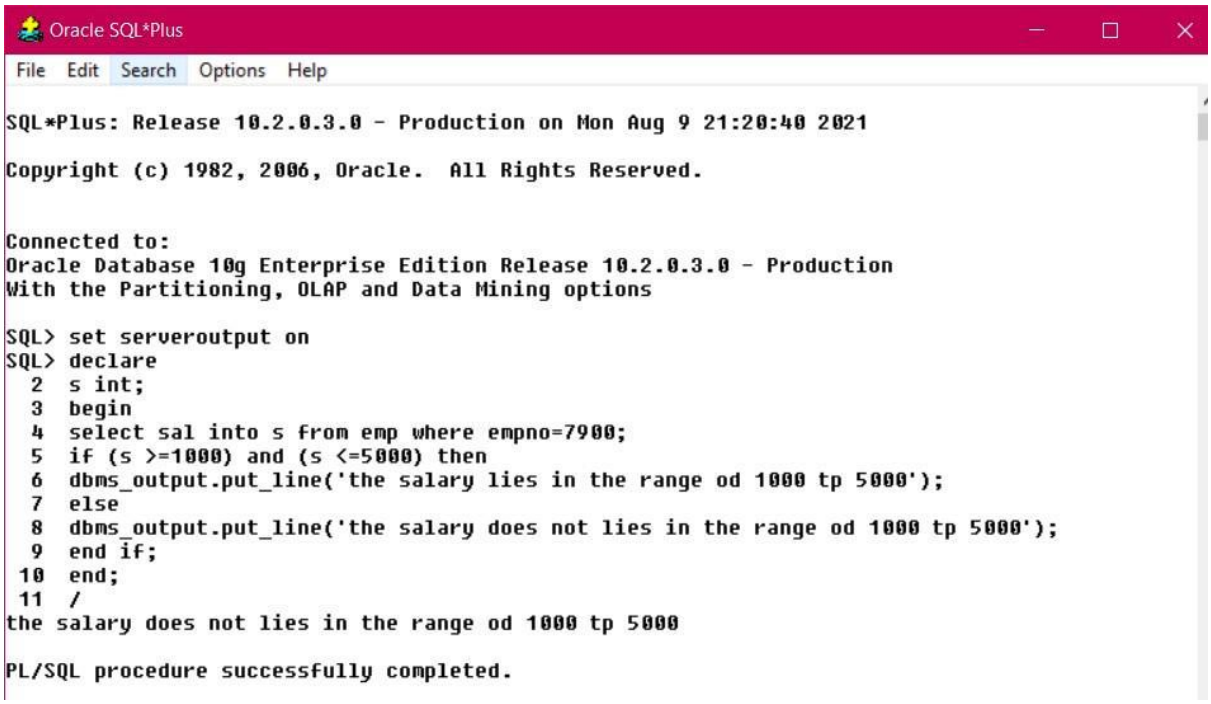
10. Write a PL/SQL block to find factorial of a number which is accepted by the user and store it under the table fac(num,fact).

```
Run SQL Command Line

PL/SQL procedure successfully completed.

SQL> create  table fac(a number,b number);

Table created.

SQL> declare
  2  a number := &a;
  3  b number := 1;
  4  c number;
  5  begin
  6  c := a;
  7  while c>0 loop
  8  b := b*c;
  9  c := c-1;
 10  end loop;
 11  insert into fac values(a,b);
 12  end;
 13  /
Enter value for a: 5
old   2: a number := &a;
new   2: a number := 5;

PL/SQL procedure successfully completed.
```

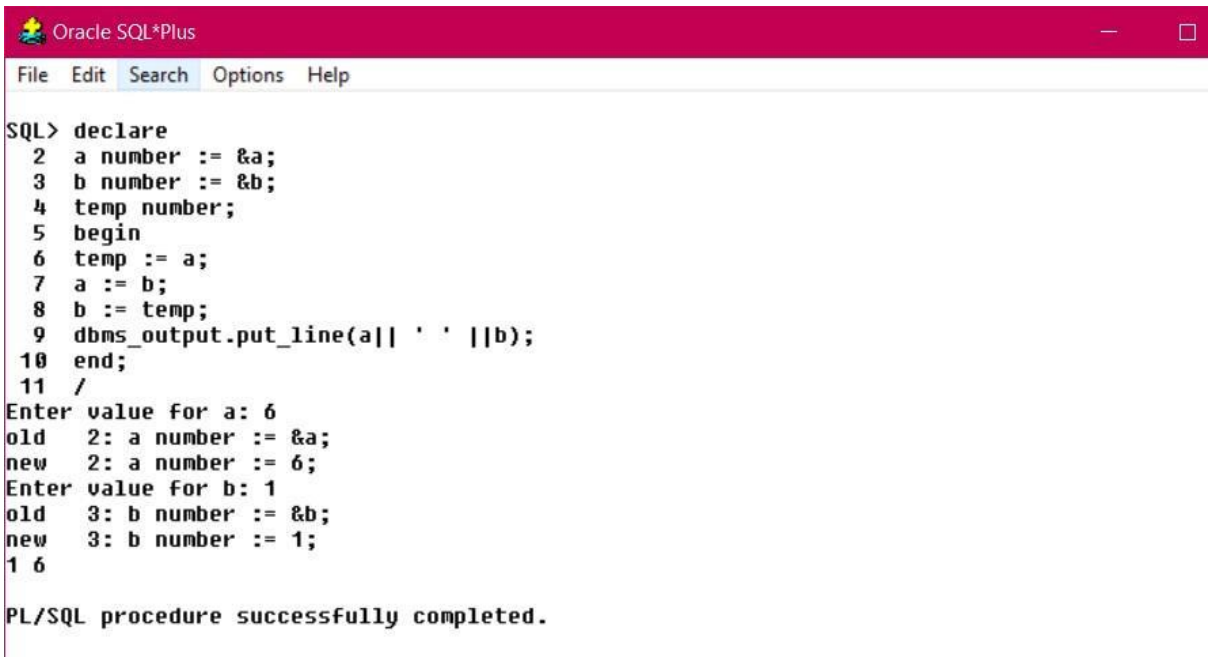
11. Write a PL/SQL to read a number and check whether it is greater than 100 or not and print appropriate message.

```
Run SQL Command Line

SQL> declare
  2  a number :=&a;
  3  begin
  4  if a>100 then
  5  dbms_output.put_line('the given number is greater than 100');
  6  else
  7  dbms_output.put_line('the given number is smaller than 100');
  8  end if;
  9  end;
 10  /
Enter value for a: 52
old   2: a number :=&a;
new   2: a number :=52;
the given number is smaller than 100

PL/SQL procedure successfully completed.
```

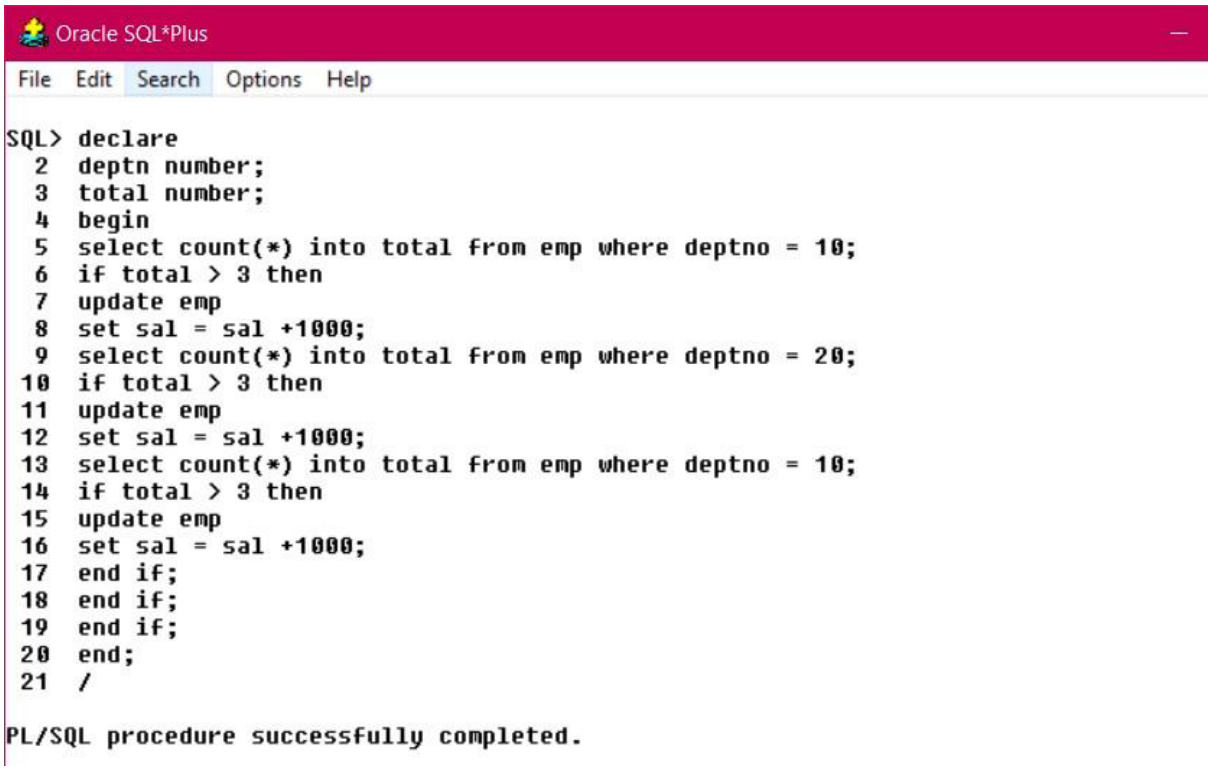
12. Write a PL/SQL to read the salary of an employee 7900 and display the appropriate message if it lies in the range of 1000 and 5000.



13. Write a PL/SQL to swap two numbers and display the swapped numbers.

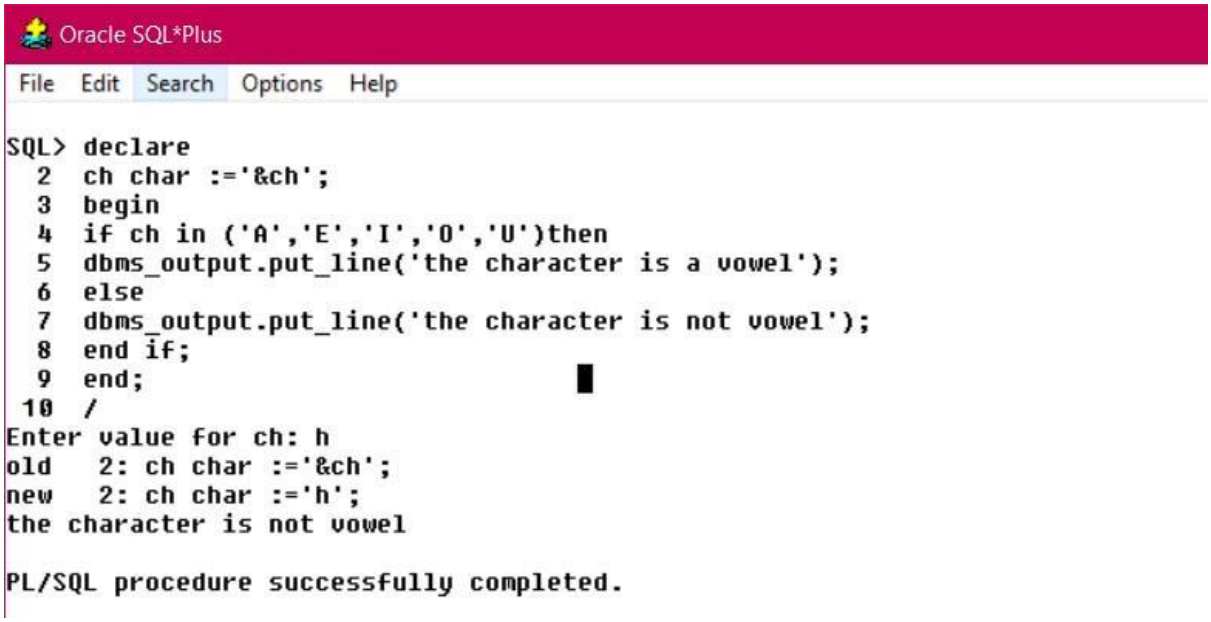


14. Write a PL/SQL block to update the salary of the employee with 1000 when total number of employees in a particular department is greater than 3.

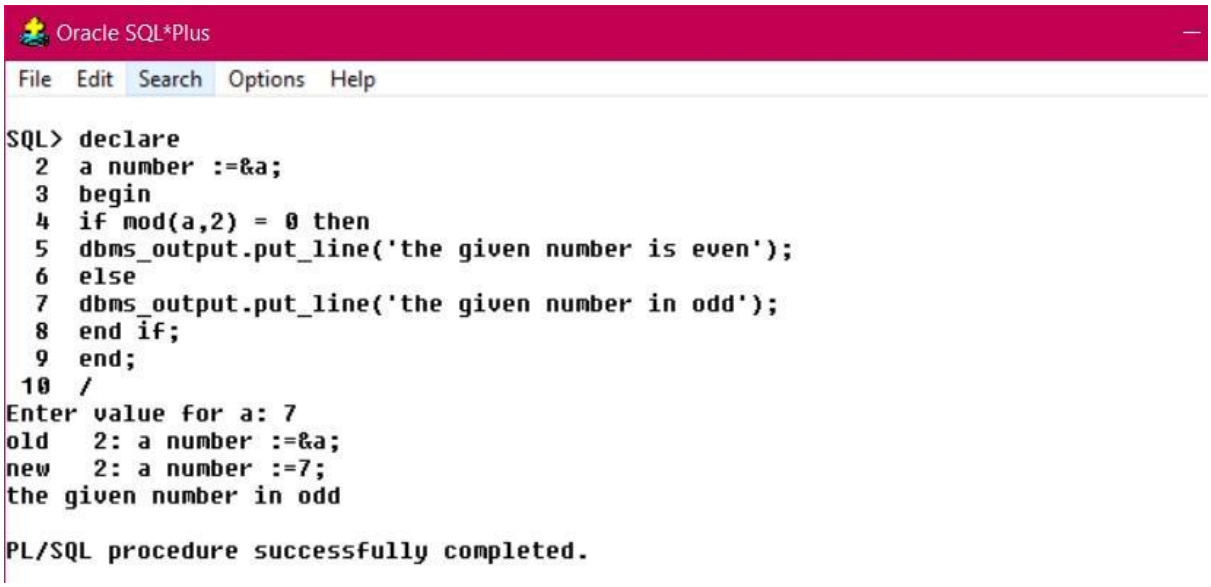


15. Write a PL/SQL block to delete the records of the table employee by accepting the table name from the user.

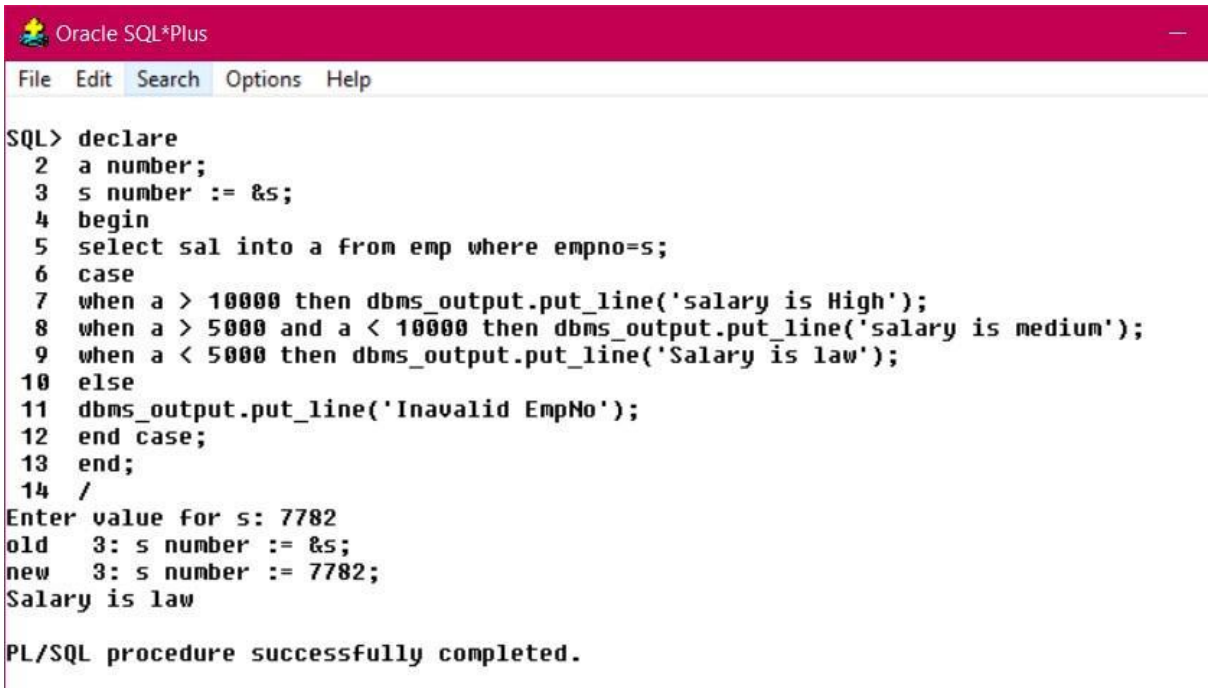
16. Write a PL/SQL to check whether the character entered is a vowel or not.



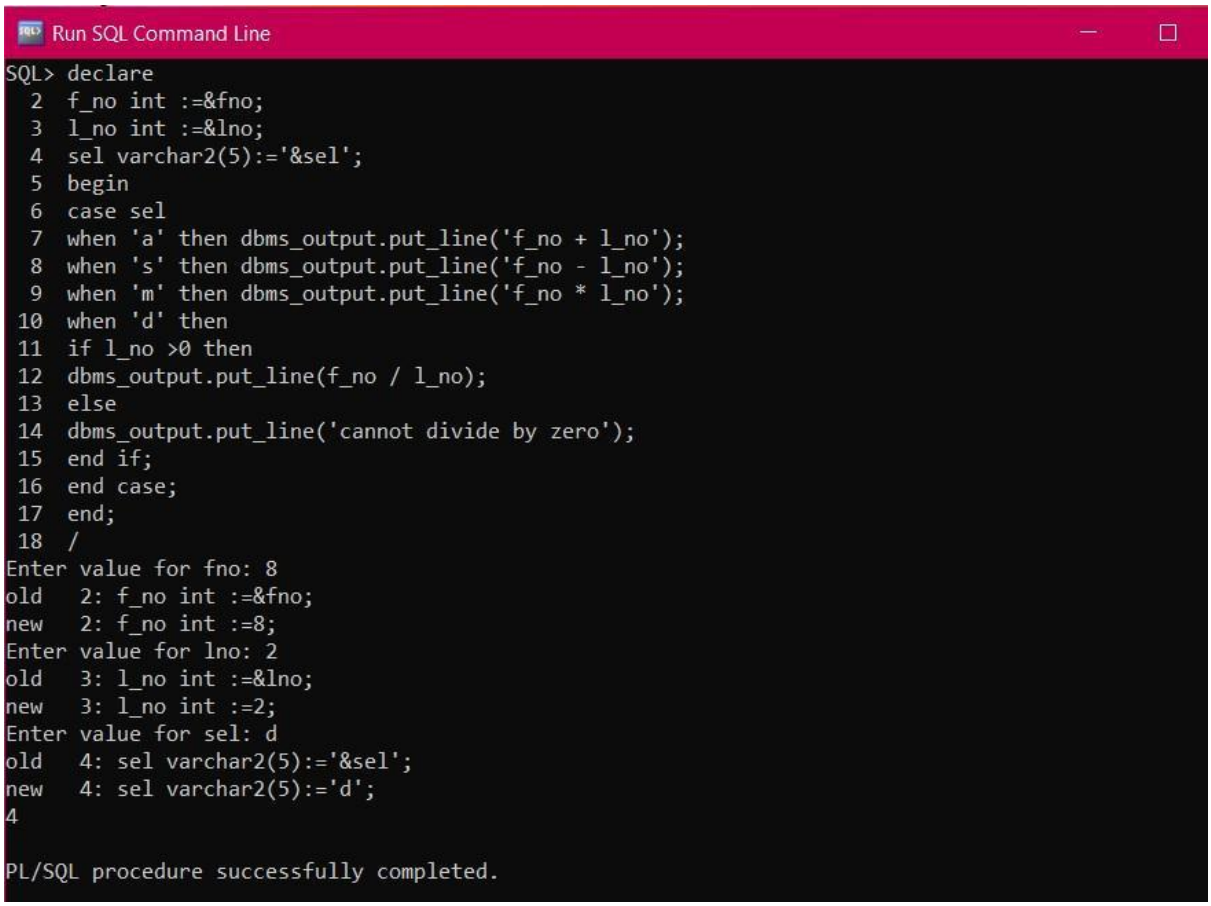
17. Write a PL/SQL to check whether a number is even or odd.



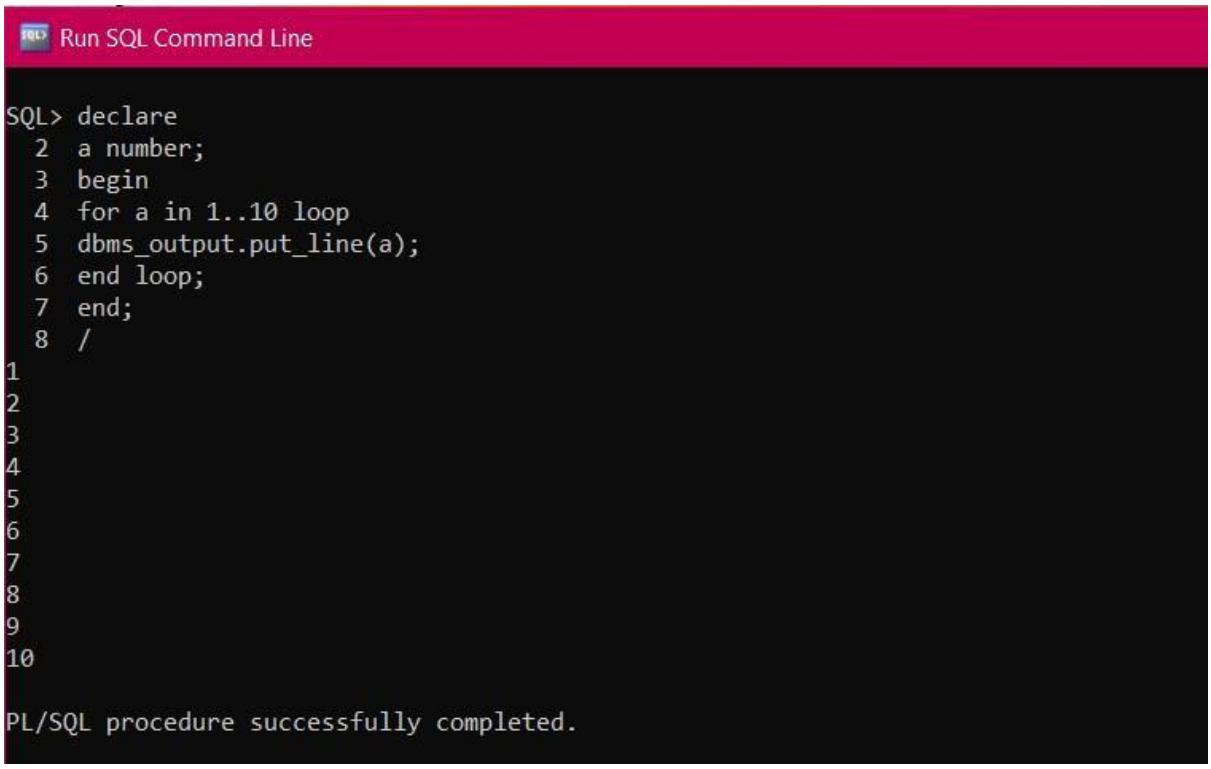
18. Write a PL/SQL block using case statement to print the salary as high if it is greater then 10000, moderate if it is between 5000 and 10000 and low if it is less than 5000.The salary has been taken as an input of a specific employee whose empid is accepted by the user.



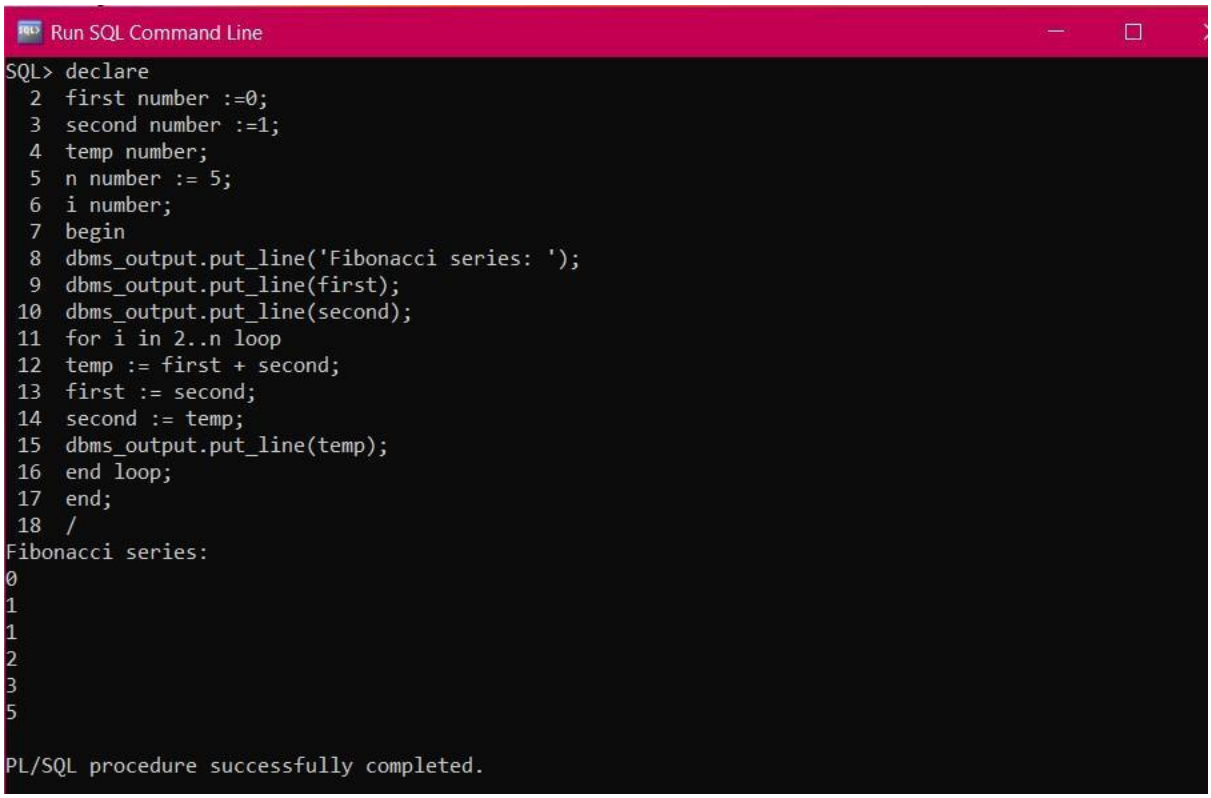
19. Write a PL/SQL block using case statement to perform addition, subtraction, multiplication and division for the individual choices a ,s ,m, d.The division can only take place if the divisor is greater than 0 else error message should be printed.



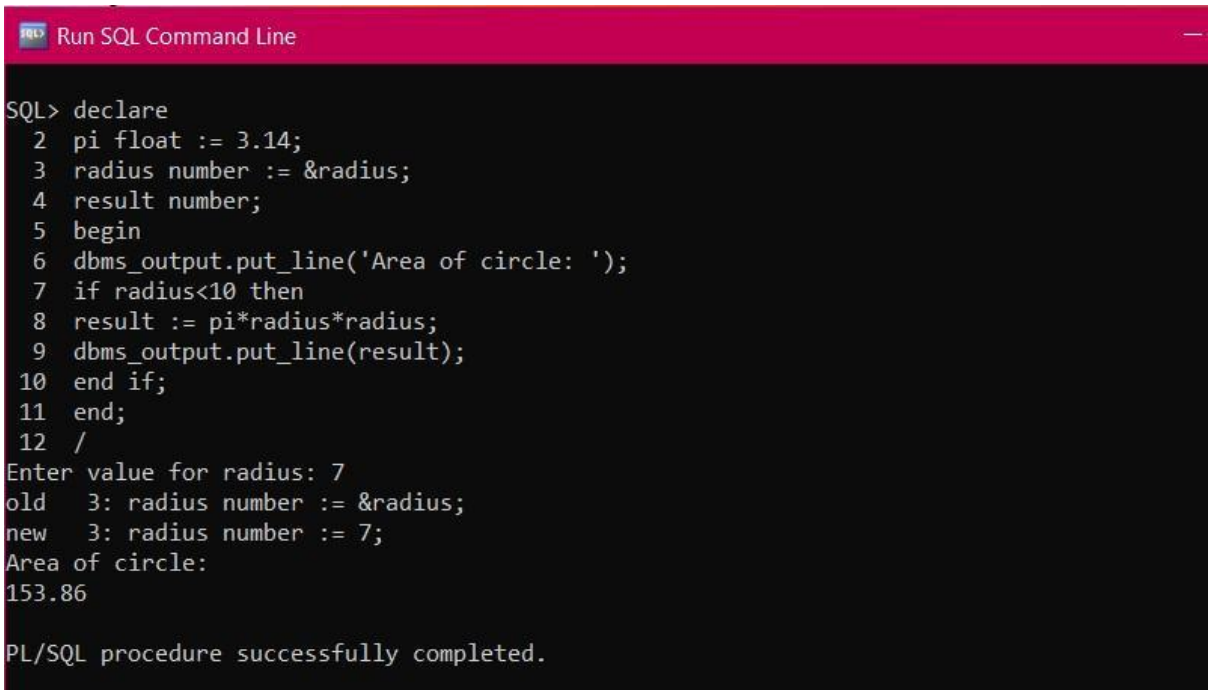
20. Write a PL/SQL block to print the numbers from 1 to 10 using While and For Loop.



21. Write a PL/SQL block to print the Fibonacci series up to 10.

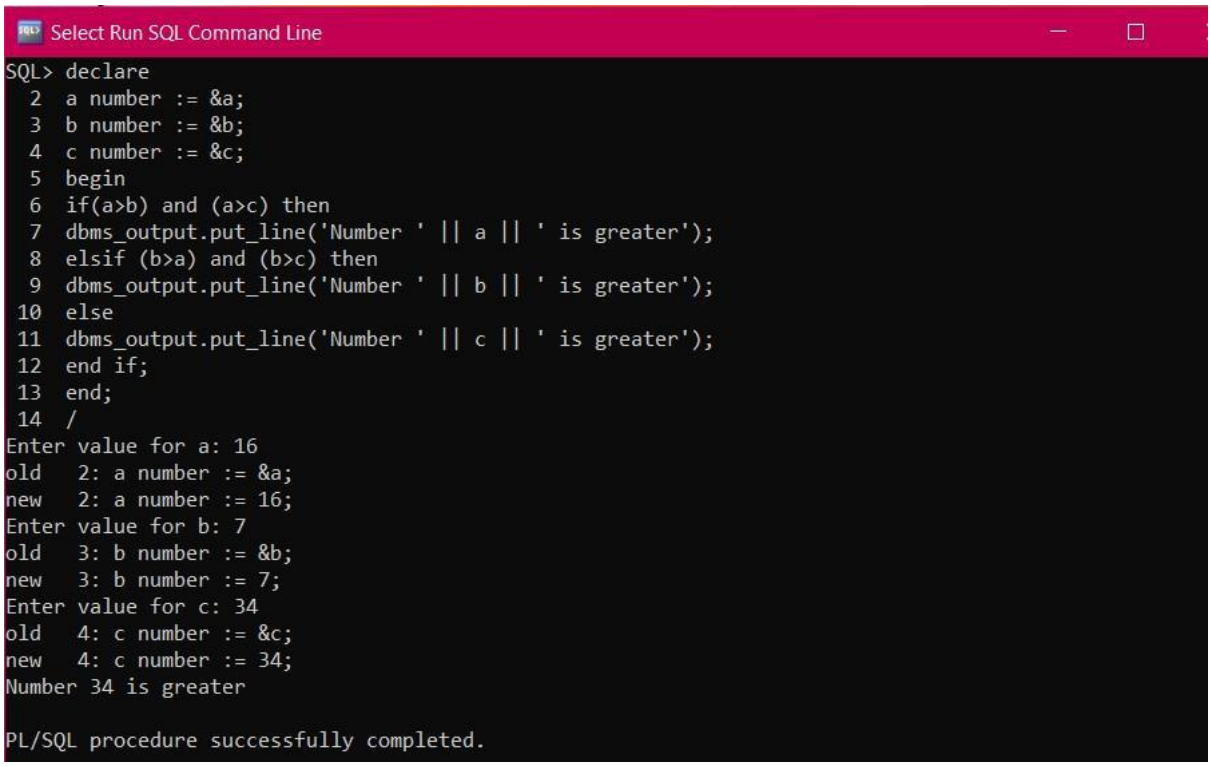


22. Write a PL/SQL block to calculate the area of a circle till radius less than 10. 23.Write a PL/SQL block to display the number of employees when the deptno is inputted by the user.

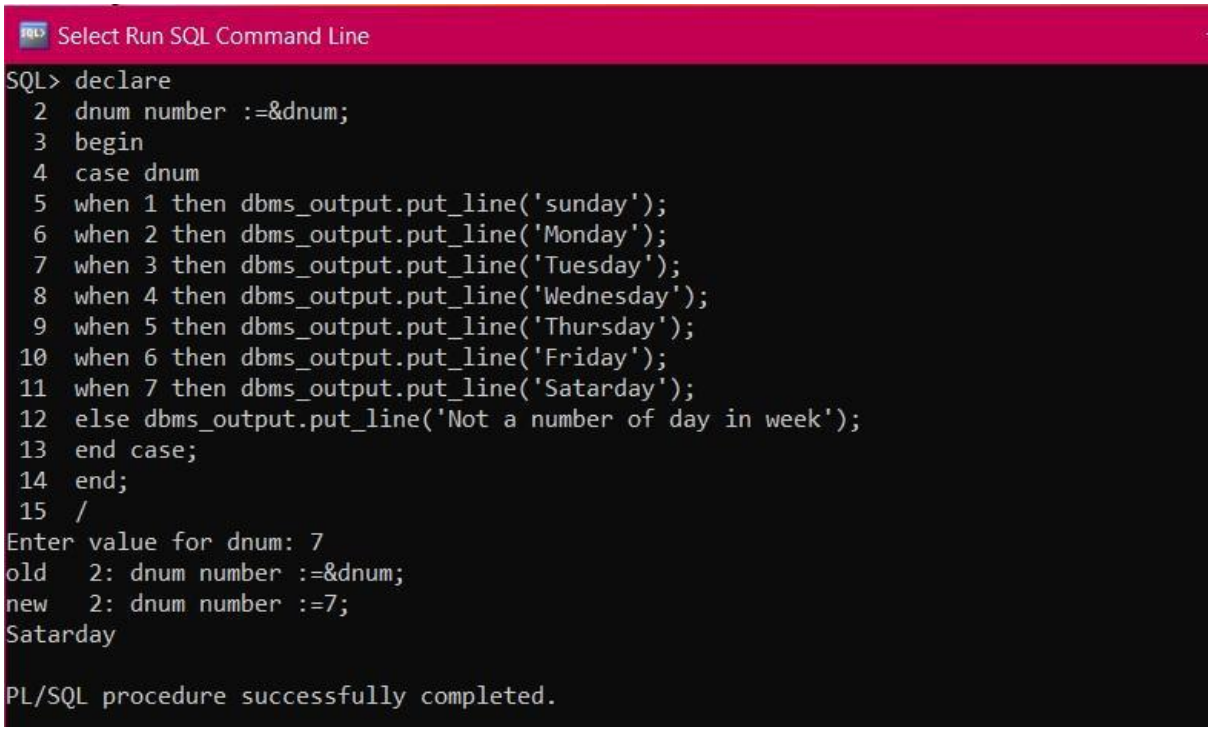


23. Write a PL/SQL block to display the number of employees when the deptno is inputted by the user.

24. Write a PL/SQL block to print greatest among three numbers.



25. Write a PL/SQL block to display the appropriate day of the week according to the choice made by the user.



26. Create a PL/SQL block that has four sections. Each section should output a statement. Use labels and the Goto command to output the section messages in the following order:

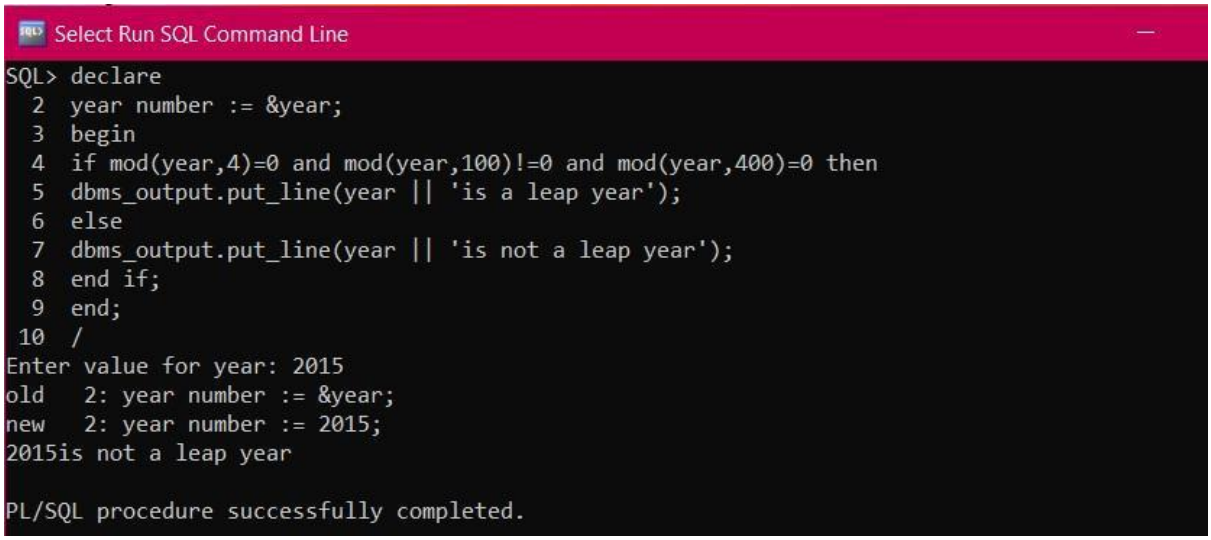
Section 3

Section 2

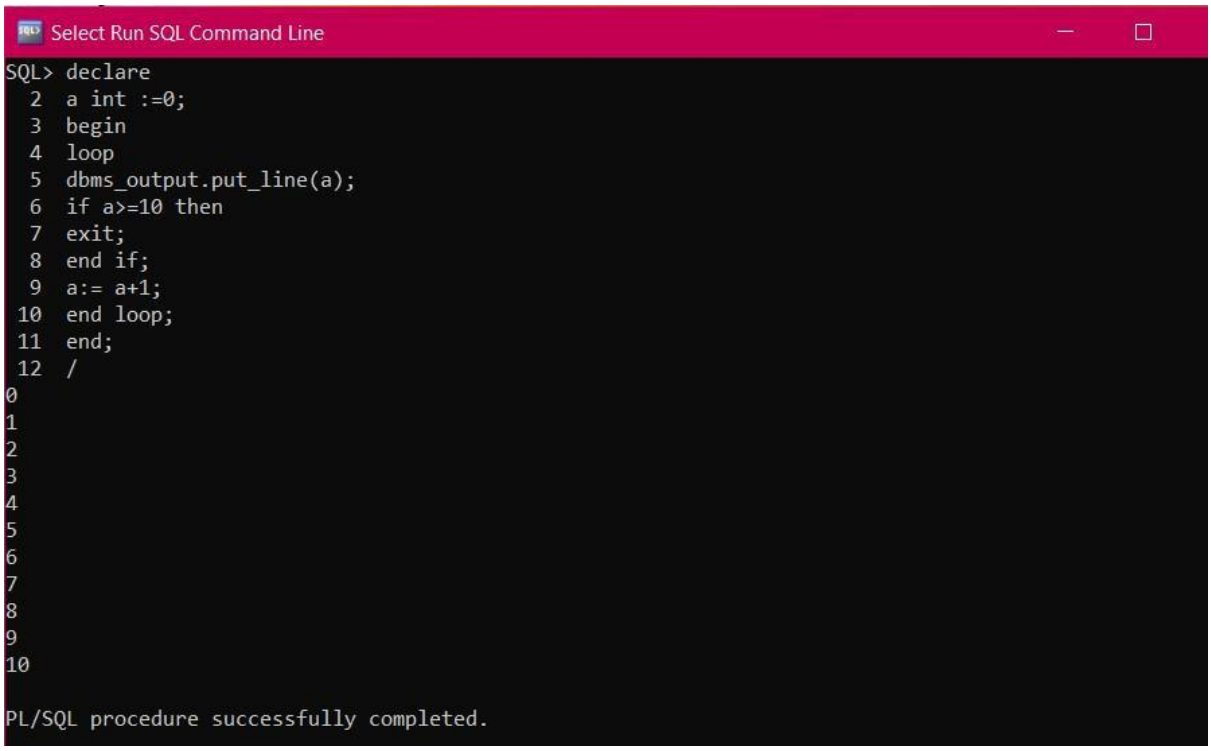
Section 1

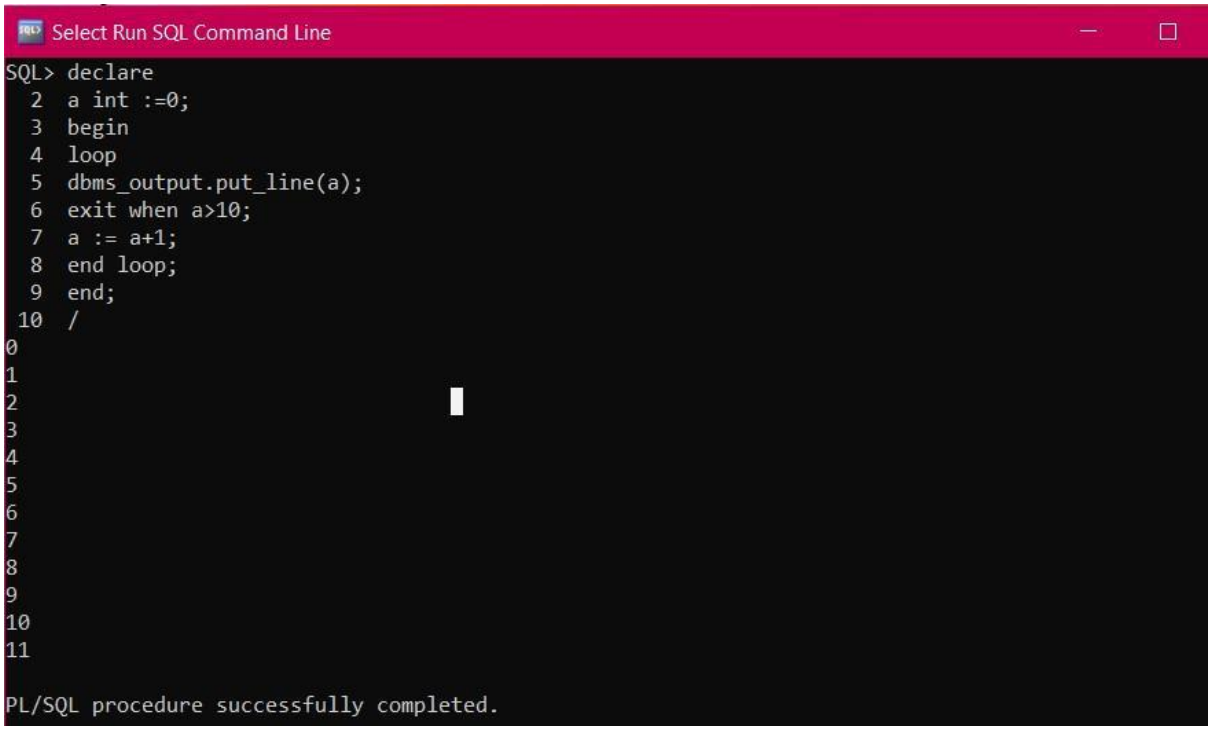
Section 4

27. Write a PL/SQL block to check whether the entered year is a leap year or not.



28. Write a PL/SQL block to display the numbers from 1 to 10 using EXIT and EXIT WHEN statement.





29. Write a PL/SQL block to accept job from EMP table. Give the following raise in the salary: -

By 9% if job is clerk. By 8% if job is manager.

By 7% if job is salesman.

Update the salary of the EMP table.

```
SQL> declare
  2  a varchar2(10):='&a';
  3  begin
  4  case a
  5  when 'CLERK'then
  6  update emp
  7  set sal=+(sal*(9/100)) where job=a;
  8  dbms_output.put_line('salary is updated');
  9  when 'MANAGER' then
 10  update emp
 11  set sal=sal+(sal*(8/100)) where job=a;
 12  dbms_output.put_line('salary is updated');
 13  when 'SALESMAN' then
 14  update emp
 15  set sal=sal+(sal*(7/100)) where job=a;
 16  dbms_output.put_line('salary is updated');
 17  end case;
 18  end;
 19  /
Enter value for a: MANAGER
old   2: a varchar2(10):='&a';
new   2: a varchar2(10):='MANAGER';

PL/SQL procedure successfully completed.

SQL>
```

30. Write a PL/SQL block to get the details of marks (rollno, marks1, marks2, grade) out of 100 for marks1 and marks2 respectively.

Display the grade in table marks using if statement as specified below If stud_percent > 70 then grade is 'A'

If stud_percent > 60 and <70 then grade is 'B' else give grade 'C'.

```
SQL> declare
  2  roll number := &roll;
  3  m1 number := &m1;
  4  m2 number := &m2;
  5  grade char;
  6  stud_percent number;
  7  begin
  8  stud_percent := ((m1+m2)/200) * 100;
  9  if stud_percent > 70 then
 10  insert into mark values(roll,m1,m2,'A');
 11  elsif stud_percent >60 and stud_percent > 70 then
 12  insert into mark values(roll,m1,m2,'B');
 13  else
 14  insert into mark values(roll,m1,m2,'C');
 15  end if;
 16  end;
 17  /
Enter value for roll: 7
old   2: roll number := &roll;
new   2: roll number := 7;
Enter value for m1: 80
old   3: m1 number := &m1;
new   3: m1 number := 80;
Enter value for m2: 90
old   4: m2 number := &m2;
new   4: m2 number := 90;

PL/SQL procedure successfully completed.
```

31. Write a PL/SQL block to book a ticket for a movie. The tickets are of two type's deluxe rows (D) and Ordinary rows (O).

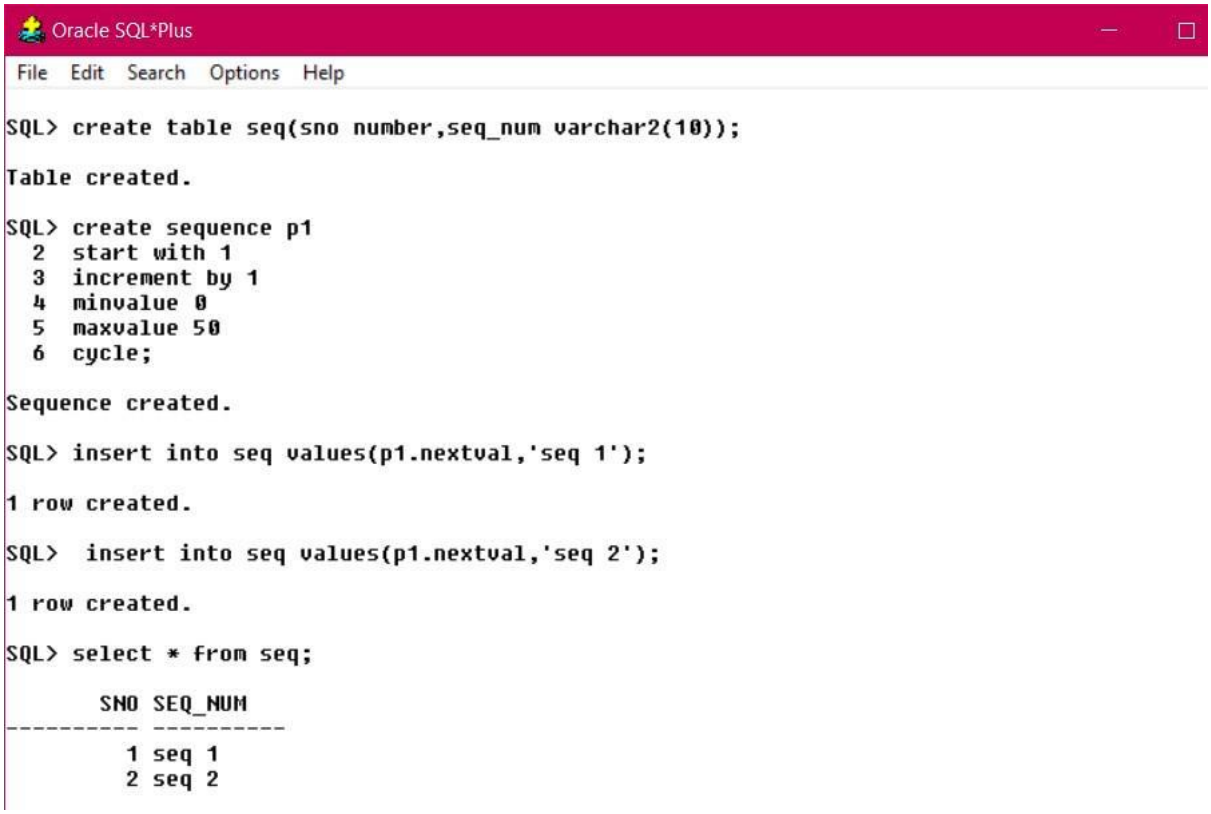
While booking the ticket the customer may ask 'D' or 'O' and number of ticket. For deluxe the rate is 350 and for ordinary 200.

Find the total amount that the customer will pay and number of tickets (using case statement).

```
SQL> declare
  2  t_type char(1) := '&t_type';
  3  no_tickets number := '&no_tickets';
  4  begin
  5  case t_type
  6  when 'd' then dbms_output.put_line('Total price: ' || no_tickets*350);
  7  when 'o' then dbms_output.put_line('Total price: ' || no_tickets*200);
  8  else
  9  dbms_output.put_line('Not a valid ticket type');
 10  end case;
 11  end;
 12  /
Enter value for t_type: o
old   2: t_type char(1) := '&t_type';
new   2: t_type char(1) := 'o';
Enter value for no_tickets: 2
old   3: no_tickets number := '&no_tickets';
new   3: no_tickets number := '2';

PL/SQL procedure successfully completed.
```

PRACTICAL NO: -3

1.Write a PL/SQL block to create a sequence by using cycle and insert the values in a table, altering sequences.



2. Write a sequence as 10, 20, 30 ….. 100 and bind it with the table product (product no, product name).

Oracle SQL*Plus

File Edit Search Options Help

```
SQL> create table products(pno number, pname varchar(20));

Table created.

SQL> create sequence pno
  2  start with 10
  3  increment by 10
  4  minvalue 0
  5  maxvalue 100
  6  nocycle;

Sequence created.

SQL> insert into products values(pno.nextval, 'product' ||pno.nextval);

1 row created.

SQL> insert into products values(pno.nextval, 'product' ||pno.nextval);

1 row created.

SQL> insert into products values(pno.nextval, 'product' ||pno.nextval);

1 row created.

SQL> select * from products;

       PNO PNAME
---------- --------------------
        10 product10
        20 product20
        30 product30
```

3. Write a sequence who's maximum value is 40 and is incremented by 4, starts with 1 and forming a cycle.

```
SQL>  create sequence proll
  2   start with 1
  3   increment by 4
  4   maxvalue 40
  5  cycle
  6  cache 5;

Sequence created.

SQL> create table prolls(no number);

Table created.

SQL> insert into prolls values(proll.nextval);

1 row created.

SQL> select * from prolls;

        NO
----------
         1

SQL> insert into prolls values(proll.nextval);

1 row created.

SQL> insert into prolls values(proll.nextval);

1 row created.

SQL> select * from prolls;

        NO
----------
         1
         5
         9

SQL>
```

PRACTICAL NO: -4

1.Create and replace an empty procedure and call it.

```
SQL> set serveroutput on
SQL> create procedure aniket as
  2  begin
  3  null;
  4  end;
  5  /

Procedure created.

SQL> execute aniket;

PL/SQL procedure successfully completed.

SQL> create or replace procedure aniket as
  2  begin
  3  dbms_output.put_line('Hello Aniket');
  4  end;
  5  /

Procedure created.

SQL> execute aniket;
Hello Aniket
```

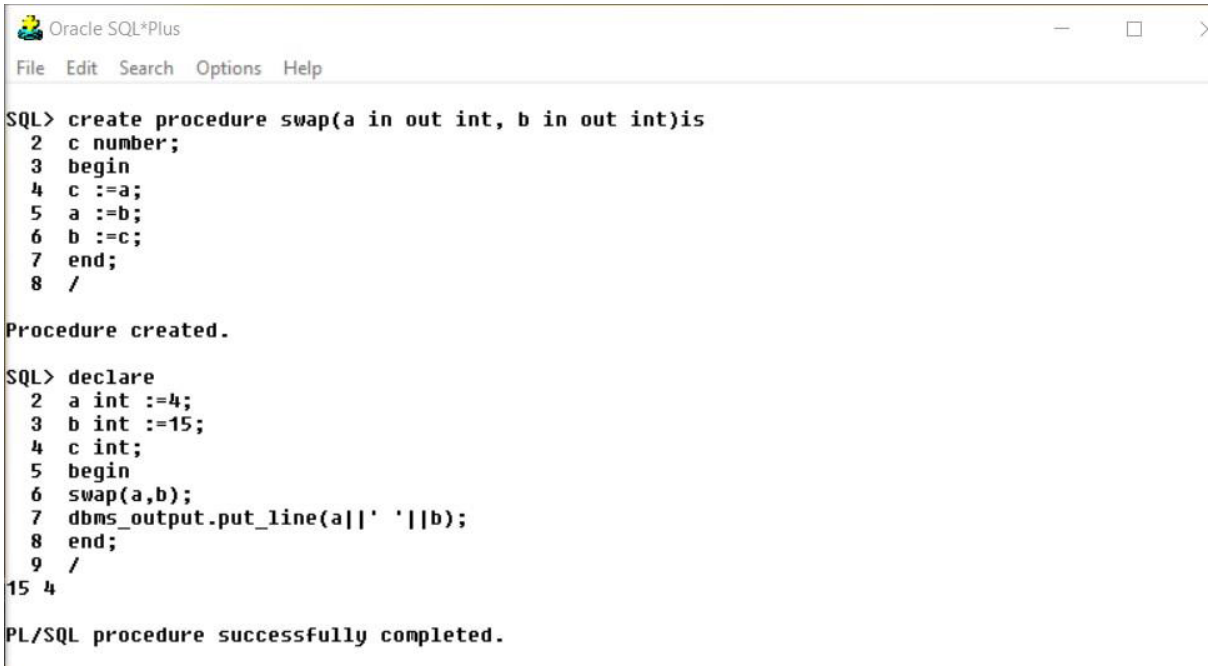
2. Create procedure and function to display square of number.

Oracle SQL*Plus

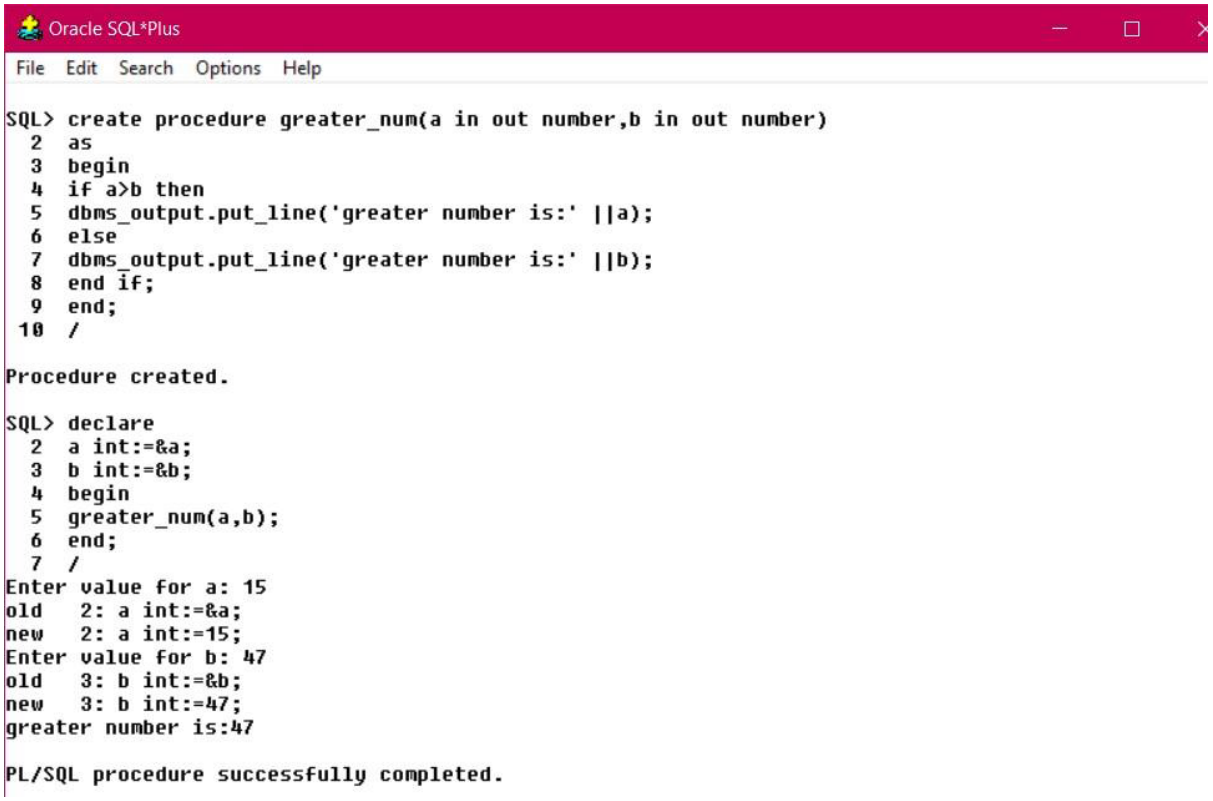
File Edit Search Options Help

```
SQL> create procedure square(n in int, s out int)
  2  as
  3  begin
  4  s:=n*n;
  5  end square;
  6  /

Procedure created.

SQL> declare
  2  n int :=&n;
  3  s int;
  4  begin
  5  square(n,s);
  6  dbms_output.put_line('square of the number is:'||s);
  7  end square;
  8  /
Enter value for n: 5
old   2: n int :=&n;
new   2: n int :=5;
square of the number is:25

PL/SQL procedure successfully completed.
```

3. Create a procedure and a function to swap two numbers.



4. Create a procedure and a function to display greatest among two.



Oracle SQL*Plus

File Edit Search Options Help

```
SQL> create or replace function greater(a in int, b in int)
  2  return int
  3  as
  4  begin
  5  if a>b then
  6  return a;
  7  else
  8  return b;
  9  end if;
 10  end;
 11  /

Function created.

SQL> select greater(50,47) from dual;

GREATER(50,47)
--------------
            50
```

5. Create a procedure and a function to display the employee name whose employee no is accepted by user.

Oracle SQL*Plus

File Edit Search Options Help

```
SQL> create procedure display_name(eno in int)
  2  as
  3  name varchar(20);
  4  begin
  5  select ename into name from emp where empno=eno;
  6  dbms_output.put_line(name);
  7  end;
  8  /

Procedure created.

SQL> execute display_name(7839);
KING

PL/SQL procedure successfully completed.
```

6. Create procedure and a function to display the sum of salary of the employees whose job is accepted by the user.

Oracle SQL*Plus

File Edit Search Options Help

```
SQL> create procedure salary(j in varchar)
  2  as
  3  sal int;
  4  begin
  5  select sum(sal) into sal from emp where j=job;
  6  dbms_output.put_line(sal);
  7  end;
  8  /

Procedure created.

SQL> execute salary('MANAGER');
8275

PL/SQL procedure successfully completed.
```

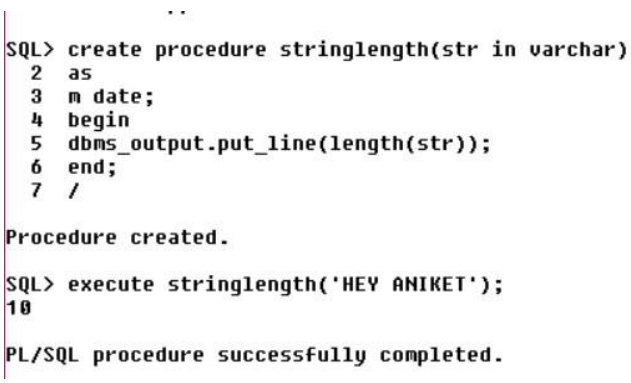
7. Create a procedure to display today's date.

```
Oracle SQL*Plus
File  Edit  Search  Options  Help

SQL> create procedure today_date
  2  as
  3  m date;
  4  begin
  5  select sysdate into m from dual;
  6  dbms_output.put_line(m);
  7  end;
  8  /

Procedure created.

SQL> execute today_date;
13-AUG-21

PL/SQL procedure successfully completed.
```

8. Create a procedure to fine the factorial of a number.

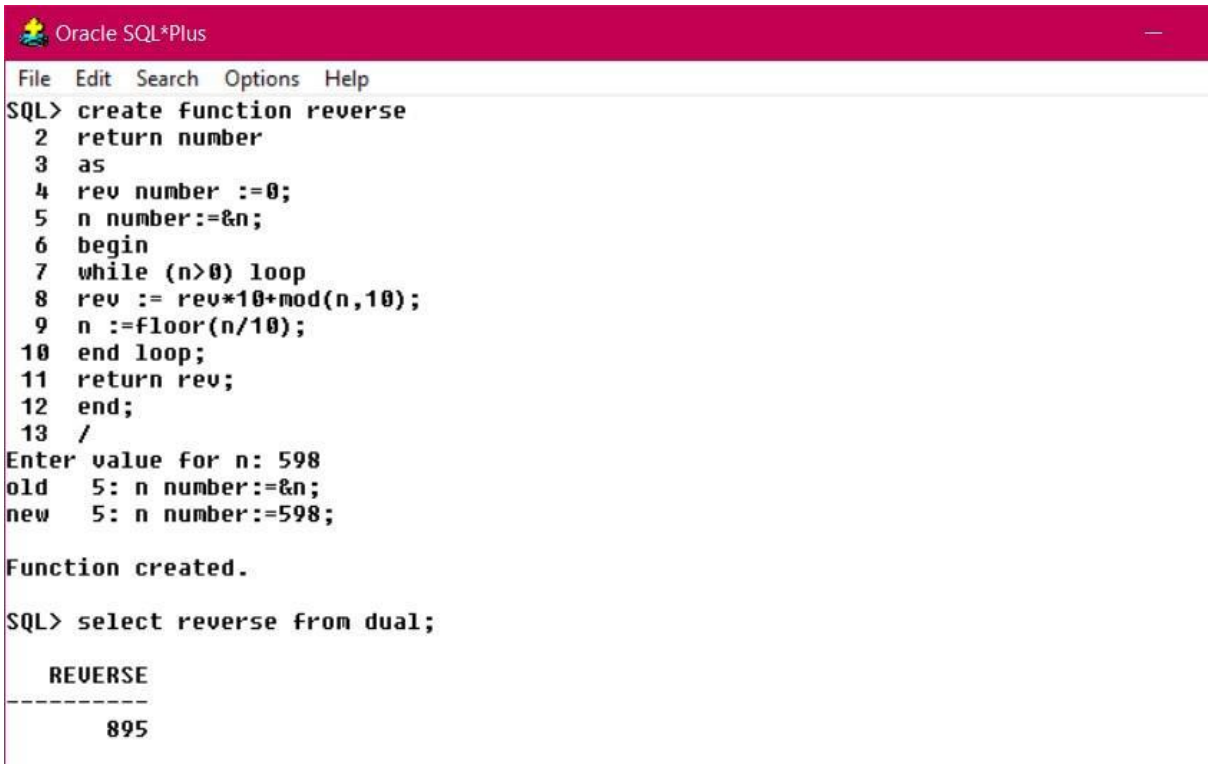
```
Oracle SQL*Plus
File  Edit  Search  Options  Help

SQL> create procedure factorial(num in int)
  2  as
  3  fact int :=1;
  4  begin
  5  for i in 1..num loop
  6  fact := fact * i;
  7  end loop;
  8  dbms_output.put_line(fact);
  9  end;
 10  /

Procedure created.

SQL> execute factorial(6);
720

PL/SQL procedure successfully completed.
```

9. Create a procedure to display length of a number.

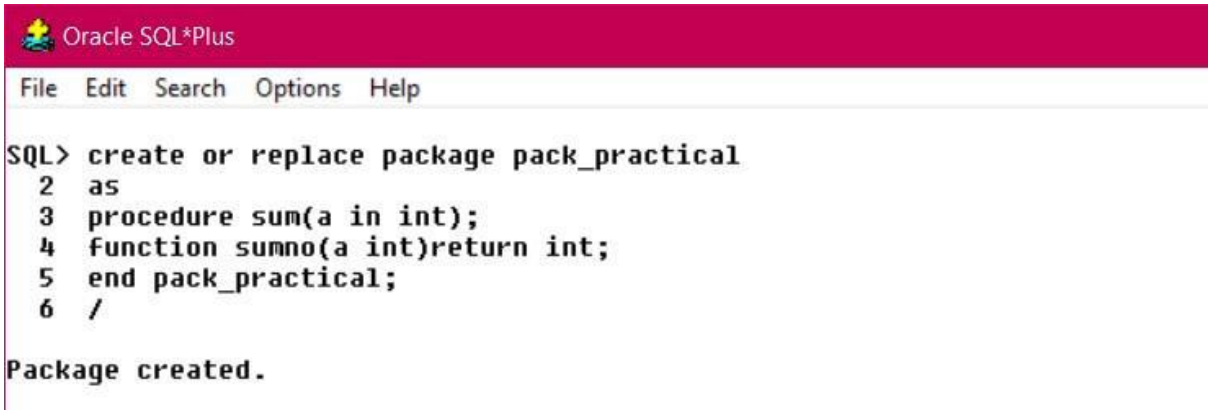


```
SQL> create procedure stringlength(str in varchar)
  2  as
  3  m date;
  4  begin
  5  dbms_output.put_line(length(str));
  6  end;
  7  /

Procedure created.

SQL> execute stringlength('HEY ANIKET');
10

PL/SQL procedure successfully completed.
```

10. Create a function to print the reverse of a string.



11. Create a package with function and procedure to fine the sum of 10 natural numbers.



Oracle SQL*Plus

File Edit Search Options Help

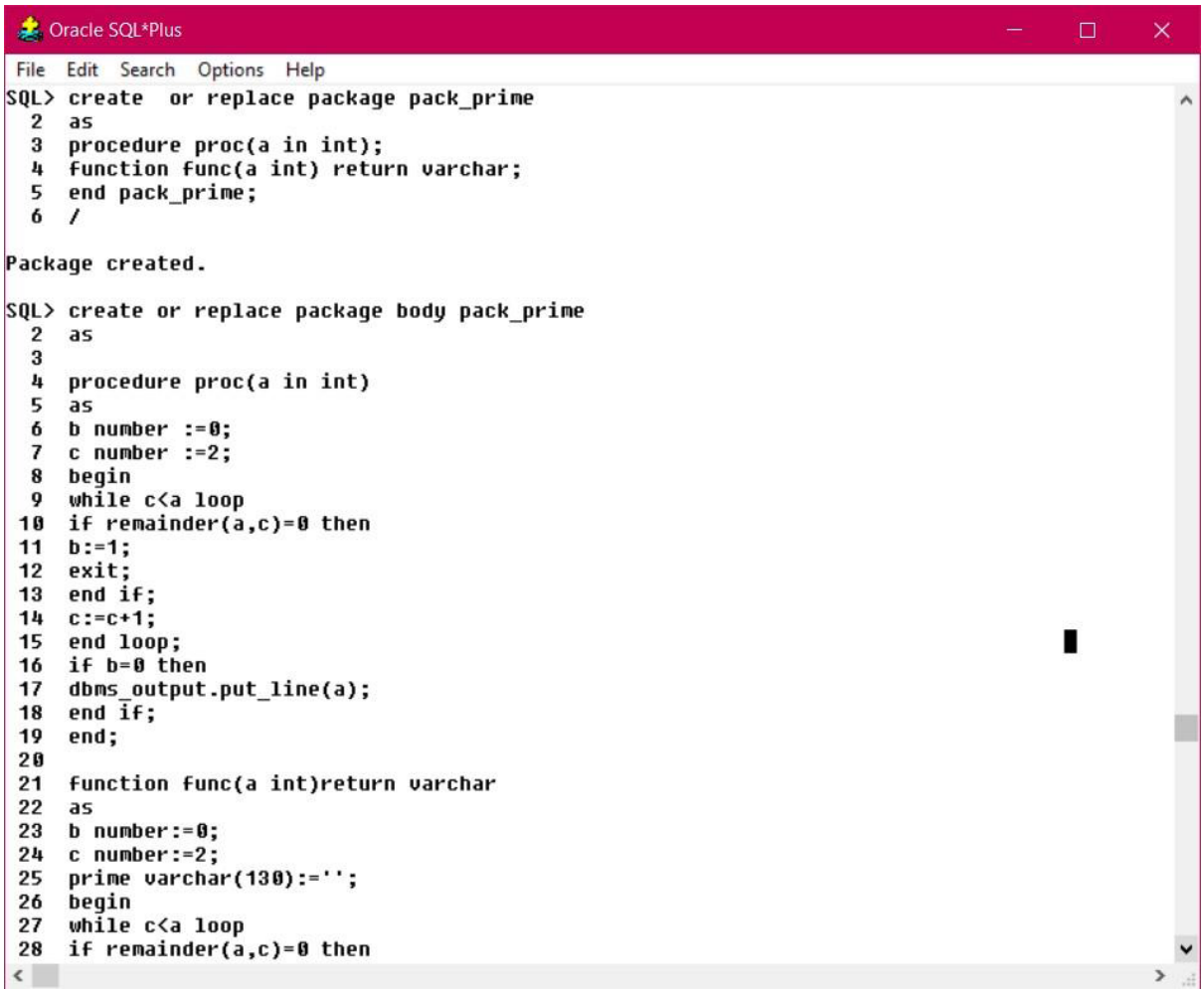
```
SQL> create or replace package body pack_practical
  2  as
  3  procedure sum(a in int)
  4  as
  5  begin
  6  dbms_output.put_line((a*(a+1))/2);
  7  end;
  8
  9   function sumno(a int) return int
 10   as
 11   begin
 12   return(a*(a+1))/2;
 13   end;
 14   end pack_practical;
 15   /

Package body created.
```

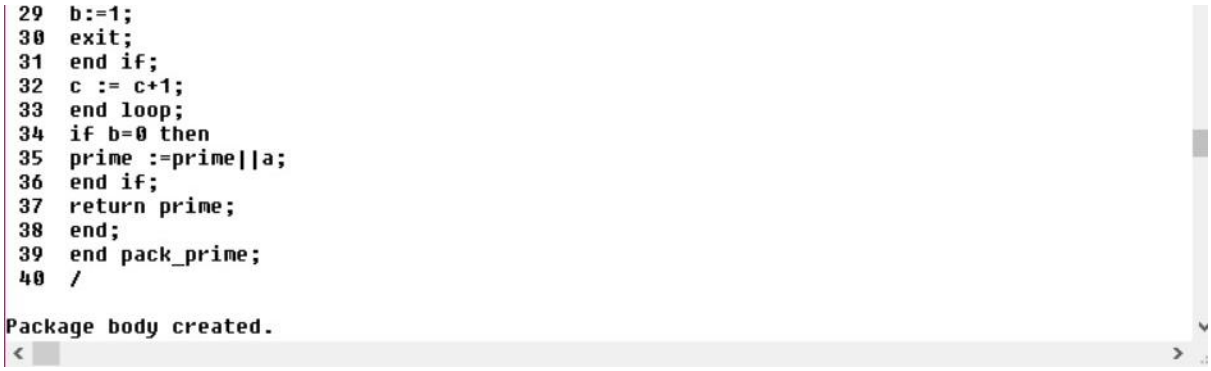
Oracle SQL*Plus

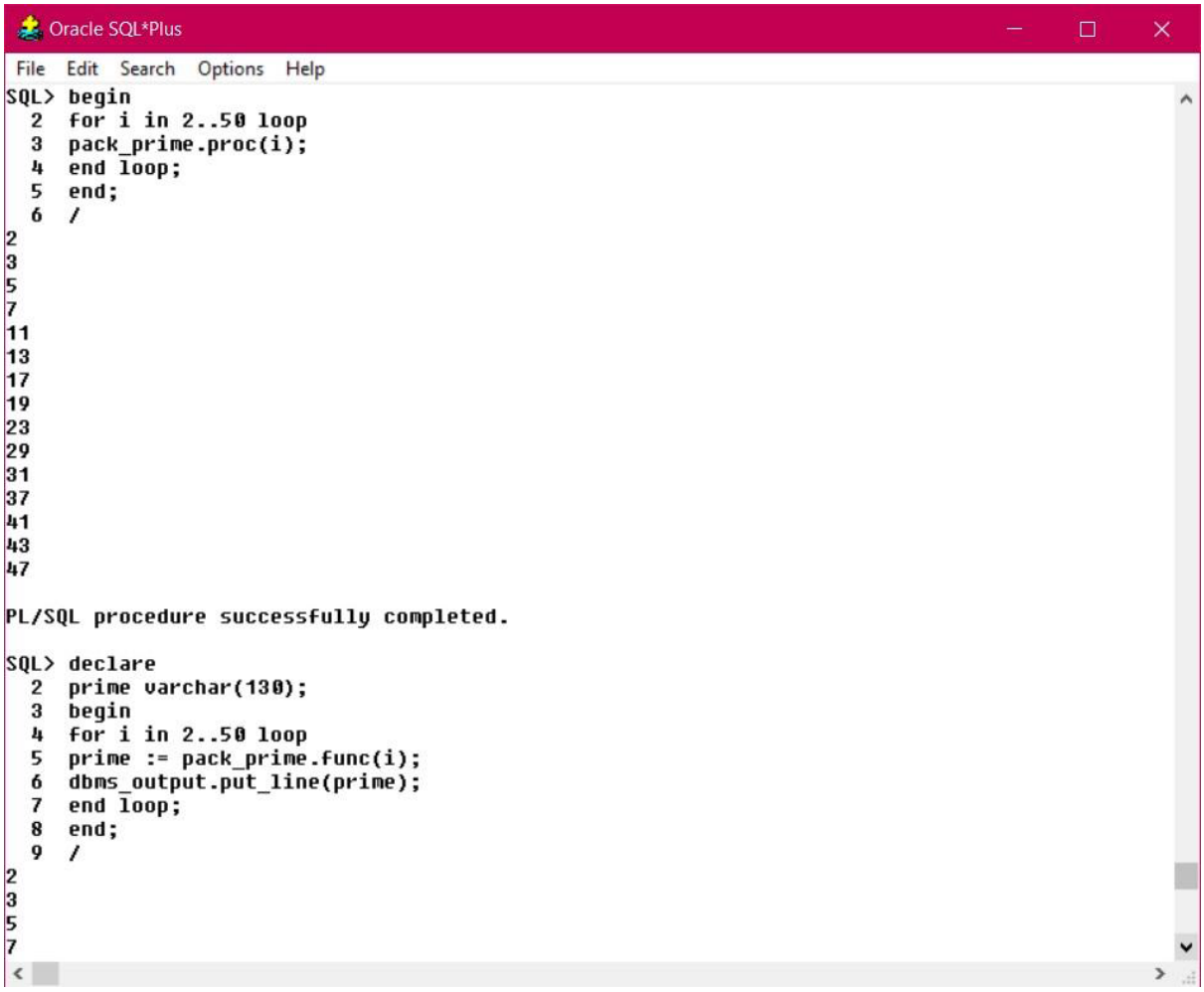
File Edit Search Options Help

```
SQL> execute pack_practical.sum(10);
55

PL/SQL procedure successfully completed.

SQL> select pack_practical.sumno(10) from dual;

PACK_PRACTICAL.SUMNO(10)
------------------------
                      55
```

12. Create a package with a function and procedure to print the prime numbers between 1 to 50.








```
11
13
17
19
23
29
31
37
41
43
47

PL/SQL procedure successfully completed.

SQL>
```

PRACTICAL NO: -5

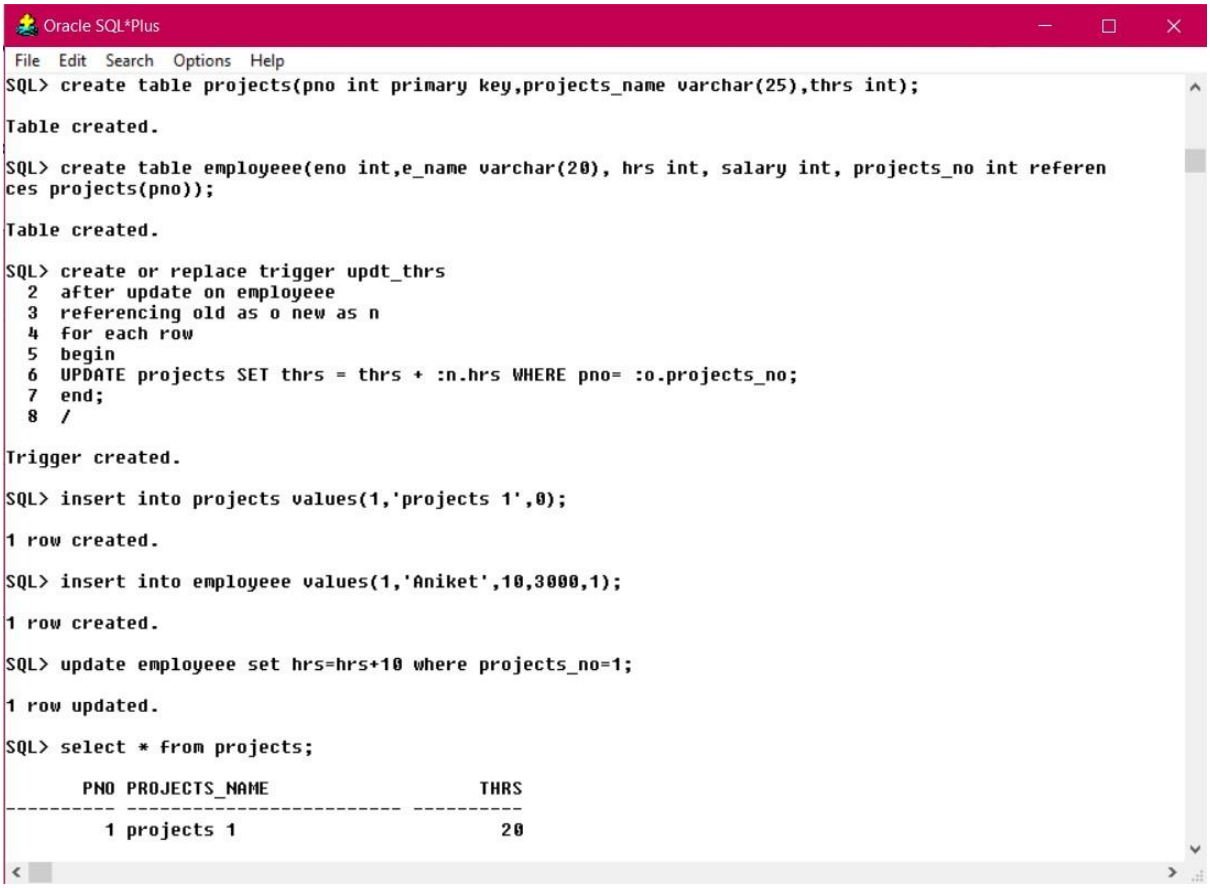
Create the following schemas:

Employee (eno, name, hrs, salary, project_no)

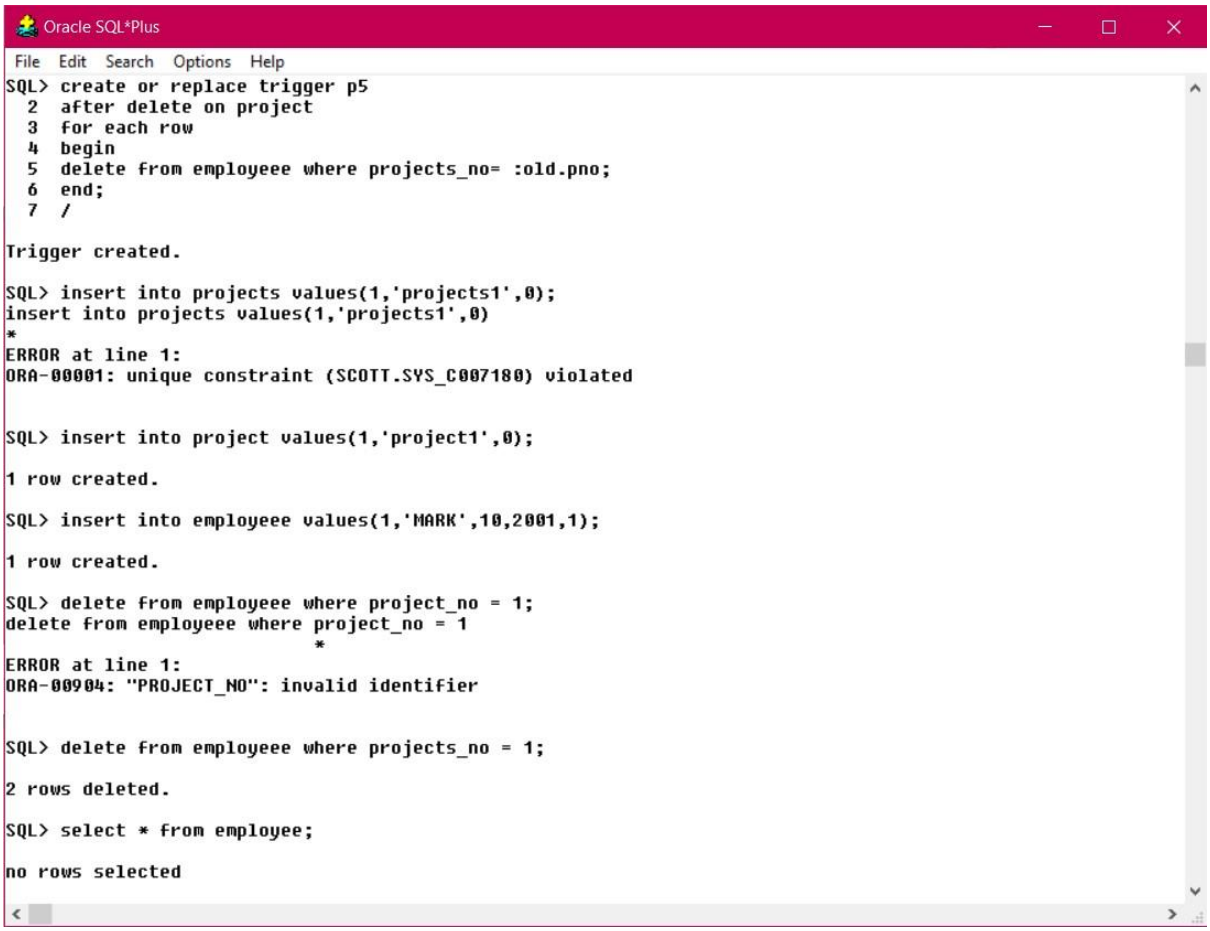
Project (pno, project name, thrs)

where thrs is the total hours and is the derived attribute. Its value is the sum of hrs of all employees working on that project. Also, in the given schemas eno and pno are primary keys. (for eg, if E1 is working on Project_no 10 for 12 hr and E2 is also working on Project_no 10 for 20 hr, than thrs will be updated automatically to 32 hrs)

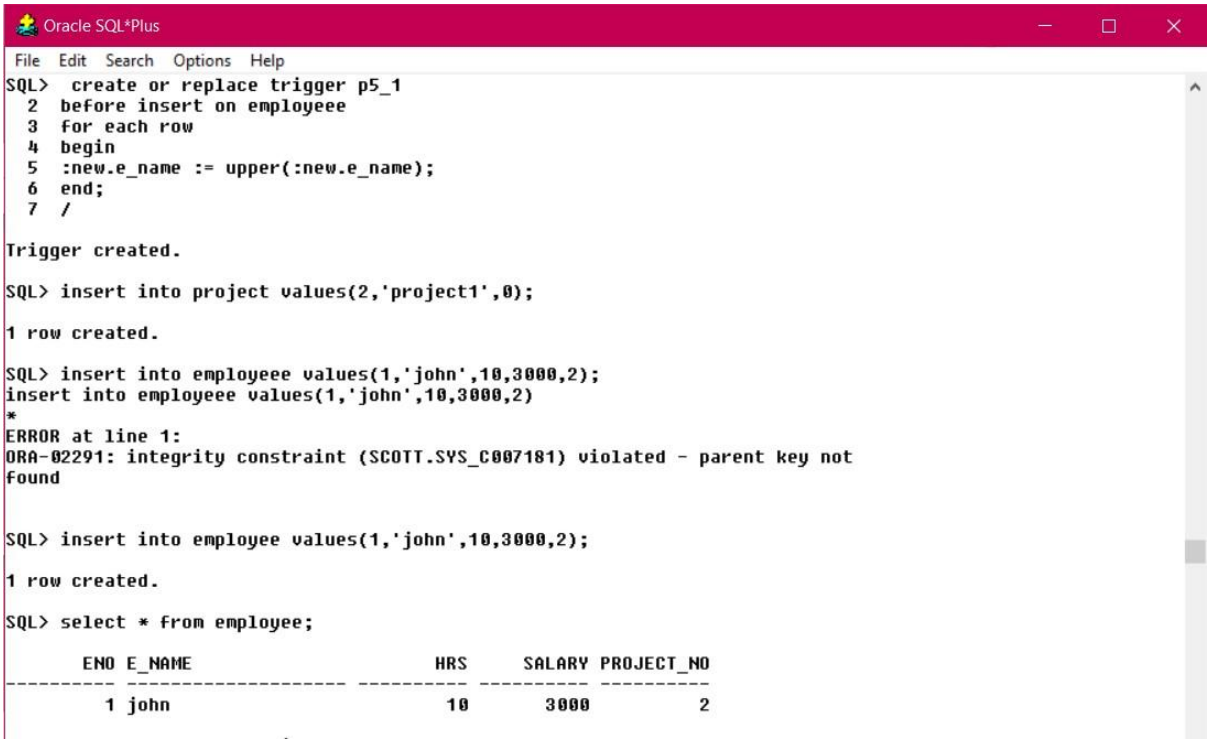
1. To update the total hours of project when hours of employee has been updated for that particular project.



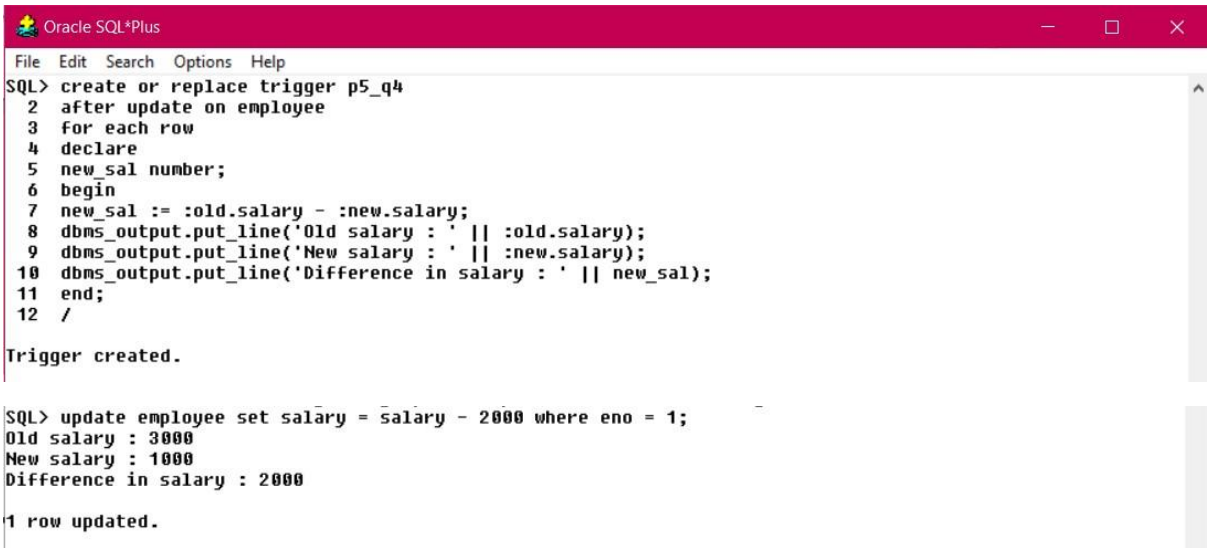
2. Creating a trigger to delete the employees wherever corresponding project is deleted from project table.



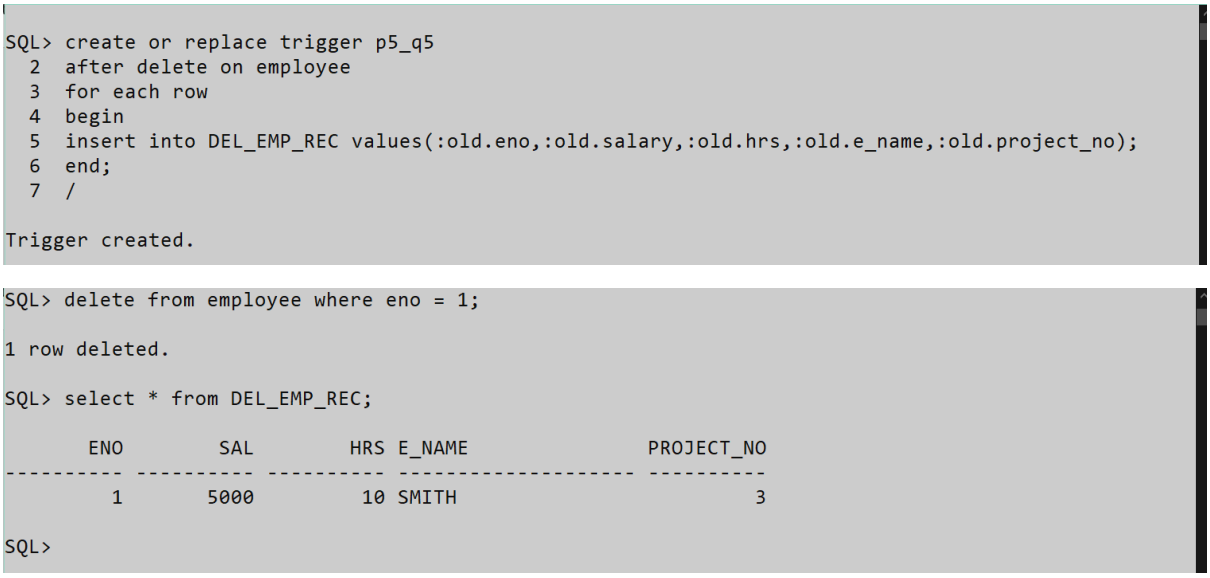
3. Create a trigger to insert the name of the employee in UPPERCASE in a table.



4. Create a trigger to update the salary of employees and to show the old salary, new salary and the difference.



5. Create a trigger to delete the record of an employee and insert the deleted record in DEL_EMP_REC table. In this case, DEL_EMP_REC is table which will hold the deleted records of an employee.



PRACTICAL NO: -6

1. Create a nested table with 5 integers and print them using collection method first and last. Also print total integers in a table.

```
SQL> declare
  2  type numerical is table of int;
  3  n numerical;
  4  begin
  5  n := numerical(1,2,3,4,5);
  6  dbms_output.put_line('Total Integers in list::'||n.count);
  7  for i in n.first..n.last loop
  8  dbms_output.put_line('Numbers: '||n(i));
  9  end loop;
 10  end;
 11  /
Total Integers in list::5
Numbers: 1
Numbers: 2
Numbers: 3
Numbers: 4
Numbers: 5

PL/SQL procedure successfully completed.
```

2. Using Varray, create a table of 5 elements and print them.

Oracle SQL*Plus

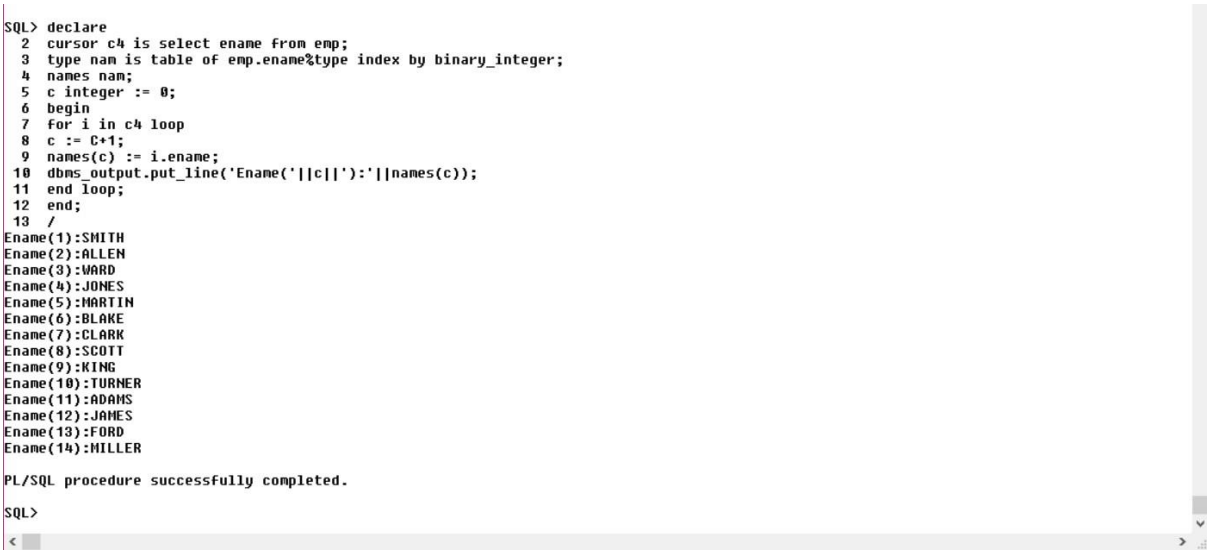
File Edit Search Options Help

```
SQL> declare
  2  type cars is varray(5) of varchar(20);
  3  c cars;
  4  begin
  5  c:=cars('Audi','BMW','JAGUAR','Rolls Royle');
  6  for i in c.first..c.last loop
  7  dbms_output.put_line('Cars: '||c(i));
  8  end loop;
  9  end;
 10  /
Cars: Audi
Cars: BMW
Cars: JAGUAR
Cars: Rolls Royle

PL/SQL procedure successfully completed.
```

3. Create a table of 10 colors, print them. Delete the color at location 5 and print the remaining elements.

```
Oracle SQL*Plus
File  Edit  Search  Options  Help
SQL> declare
  2  type color is table of varchar(20);
  3  c color;
  4  begin
  5  c:=color('red','green','blue','black','white','pink','brown','violet','biege','yellow');
  6  for i in c.first..c.last loop
  7  dbms_output.put_line('Color: '||c(i));
  8  end loop;
  9  dbms_output.put_line('------------------------');
 10  c.delete(5);
 11  for i in 1..4 loop
 12  dbms_output.put_line('color: '||c(i));
 13  end loop;
 14  for i in 6..10 loop
 15  dbms_output.put_line('color: '||c(i));
 16  end loop;
 17  end;
 18  /
Color: red
Color: green
Color: blue
Color: black
Color: white
Color: pink
Color: brown
Color: violet
Color: biege
Color: yellow
------------------------
color: red
color: green
color: blue
color: black
color: pink
color: brown
color: violet
color: biege
color: yellow

PL/SQL procedure successfully completed.

SQL>
```

4. Using Type table, print the name of the employees.



```
SQL> declare
  2  cursor c4 is select ename from emp;
  3  type nam is table of emp.ename%type index by binary_integer;
  4  names nam;
  5  c integer := 0;
  6  begin
  7  for i in c4 loop
  8  c := C+1;
  9  names(c) := i.ename;
 10  dbms_output.put_line('Ename('||c||'):'||names(c));
 11  end loop;
 12  end;
 13  /
Ename(1):SMITH
Ename(2):ALLEN
Ename(3):WARD
Ename(4):JONES
Ename(5):MARTIN
Ename(6):BLAKE
Ename(7):CLARK
Ename(8):SCOTT
Ename(9):KING
Ename(10):TURNER
Ename(11):ADAMS
Ename(12):JAMES
Ename(13):FORD
Ename(14):MILLER

PL/SQL procedure successfully completed.

SQL>
```

5. Using Index by table type, create a type of numbers indexed by binary integer to store the values from 1 to 10 at continuous index position. Delete the value at position 6 and display all the values.

```
SQL> declare
  2  type indextype is table of Number index by binary_integer;
  3  v1 indextype;
  4  begin
  5  v1(1):=1;
  6  v1(2):=2;
  7  v1(3):=3;
  8  v1(4):=4;
  9  v1(5):=5;
 10  v1(6):=6;
 11  v1(7):=7;
 12  v1(8):=8;
 13  v1(9):=9;
 14  v1(10):=10;
 15  for i in v1.first..v1.last loop
 16  dbms_output.put_line(v1(i));
 17  end loop;
 18  v1.delete(6);
 19  dbms_output.put_line('-------------------------');
 20  for i in 1..5 loop
 21  dbms_output.put_line(v1(i));
 22  end loop;
 23  for i in 7..10 loop
 24  dbms_output.put_line(v1(i));
 25  end loop;
 26  end;
 27  /
1
2
3
4
5
6
7
8
9
10
-------------------------
1
2
3
4
5
7
8
9
10

PL/SQL procedure successfully completed.

SQL>
```

PRACTICAL NO: -7

1. Using rowtype variable, display the records of the employee whose employee no is given by the user.

Oracle SQL*Plus

File Edit Search Options Help

```
SQL> declare
  2  erc emp%rowtype;
  3  begin
  4  select * into erc from emp where empno=&empno;
  5  dbms_output.put_line(erc.ename||'  '||erc.sal||'  '||erc.job||'  '||erc.deptno);
  6  end;
  7  /
Enter value for empno: 7839
old   4: select * into erc from emp where empno=&empno;
new   4: select * into erc from emp where empno=7839;
KING  5000  PRESIDENT  10

PL/SQL procedure successfully completed.
```

2. Using cursor type variable, display the records of the employee.

Oracle SQL*Plus

File Edit Search Options Help

```
SQL> declare
  2  cursor c351 is select * from emp;
  3  erc c351%rowtype;
  4  begin
  5  open c351;
  6  loop
  7  fetch c351 into erc;
  8  exit when c351%notfound;
  9  dbms_output.put_line(erc.empno||'  ' ||erc.ename);
 10  end loop;
 11  end;
 12  /
7369  SMITH
7499  ALLEN
7521  WARD
7566  JONES
7654  MARTIN
7698  BLAKE
7782  CLARK
7788  SCOTT
7839  KING
7844  TURNER
7876  ADAMS
7900  JAMES
7902  FORD
7934  MILLER

PL/SQL procedure successfully completed.
```

Oracle SQL*Plus

File Edit Search Options Help

```
SQL> declare
  2  cursor c351 is select * from emp;
  3  erc c351%rowtype;
  4  begin
  5  open c351;
  6  fetch c351 into erc;
  7  while c351%found loop
  8   dbms_output.put_line(erc.empno||'  ' ||erc.ename||'  '||erc.deptno);
  9  fetch c351 into erc;
 10  end loop;
 11  close c351;
 12  end;
 13  /
7369  SMITH  20
7499  ALLEN  30
7521  WARD  30
7566  JONES  20
7654  MARTIN  30
7698  BLAKE  30
7782  CLARK  10
7788  SCOTT  20
7839  KING  10
7844  TURNER  30
7876  ADAMS  20
7900  JAMES  30
7902  FORD  20
7934  MILLER  10

PL/SQL procedure successfully completed.
```

Oracle SQL*Plus

File Edit Search Options Help

```
SQL> declare
  2  cursor c351 is select * from emp;
  3  begin
  4  for erc in c351 loop
  5   dbms_output.put_line(erc.empno||'  ' ||erc.ename||'  '||erc.deptno);
  6  end loop;
  7  end;
  8  /
7369  SMITH  20
7499  ALLEN  30
7521  WARD  30
7566  JONES  20
7654  MARTIN  30
7698  BLAKE  30
7782  CLARK  10
7788  SCOTT  20
7839  KING  10
7844  TURNER  30
7876  ADAMS  20
7900  JAMES  30
7902  FORD  20
7934  MILLER  10

PL/SQL procedure successfully completed.
```

3. Using User based record, create a book type , read and display the values for specific

```
SQL> declare
  2  type book is record
  3  (bid int,bname varchar(25),bprice int);
  4  b1 book;
  5  b2 book;
  6  begin
  7  b1.bid :=351;
  8  b1.bname :='DBMS2';
  9  b1.bprice :=3994;
 10  -- print the details of book1
 11  dbms_output.put_line(b1.bid||'  '||b1.bname||'  '||b1.bprice);
 12  end;
 13  /
351  DBMS2  3994

PL/SQL procedure successfully completed.

SQL>
```

PRACTICAL NO: -8

1. Implement System Defined exception.

```
SQL> declare
  2  erec emp%rowtype;
  3  begin
  4
  5   select * into erec from emp where sal>1000;
  6
  7   dbms_output.put_line(erec.sal);
  8  exception
  9  when TOO_MANY_ROWS then
 10  dbms_output.put_line('Select statement is fetching than one row');
 11  end;
 12  /
Select statement is fetching than one row

PL/SQL procedure successfully completed.
```

2. Write a PL/SQL block to raise an exception when an employee name does not start with 'A'.

Oracle SQL*Plus

File Edit Search Options Help

```
SQL> --351--Aniket
SQL> declare
  2  name varchar(20) :='&name';
  3  name_val varchar(20);
  4  no_match_found exception;
  5  begin
  6  select ename into name_val from emp where ename=name;
  7  if name_val not like '%A' then
  8  raise no_match_found;
  9  else
 10  dbms_output.put_line(name_val);
 11  end if;
 12  exception
 13  when no_match_found then
 14  dbms_output.put_line('name does''t match given condition');
 15  end;
 16  /
Enter value for name: KING
old   2: name varchar(20) :='&name';
new   2: name varchar(20) :='KING';
name does't match given condition

PL/SQL procedure successfully completed.
```

3.Write a PL/SQL block to implement raise_application_error() when the age of a student is entered less than 0.

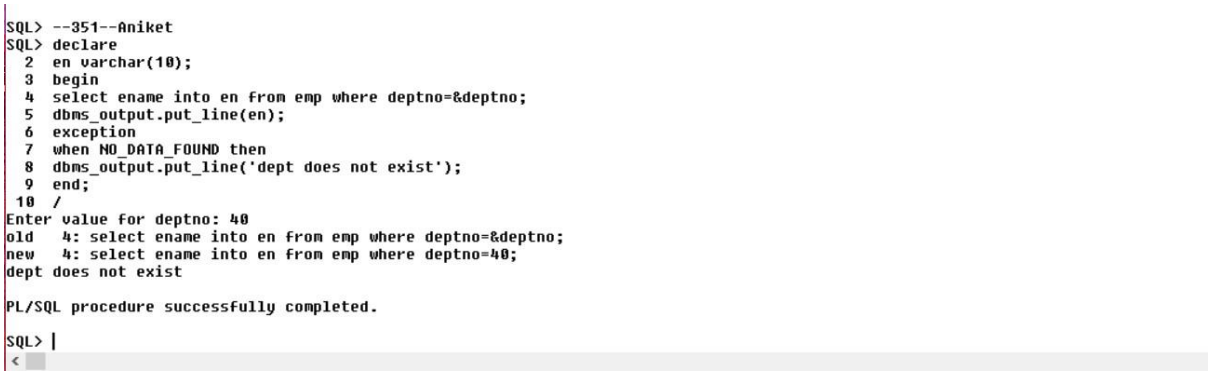
Oracle SQL*Plus

File Edit Search Options Help

```
SQL> --351--Aniket
SQL> declare
  2  a int :=&a;
  3  age exception;
  4  begin
  5  if (a<=0) then
  6  raise_application_error(-20000,'Age should be greater than 0');
  7  else
  8  dbms_output.put_line(a);
  9  end if;
 10  end;
 11  /
Enter value for a: 12
old   2: a int :=&a;
new   2: a int :=12;
12

PL/SQL procedure successfully completed.
```

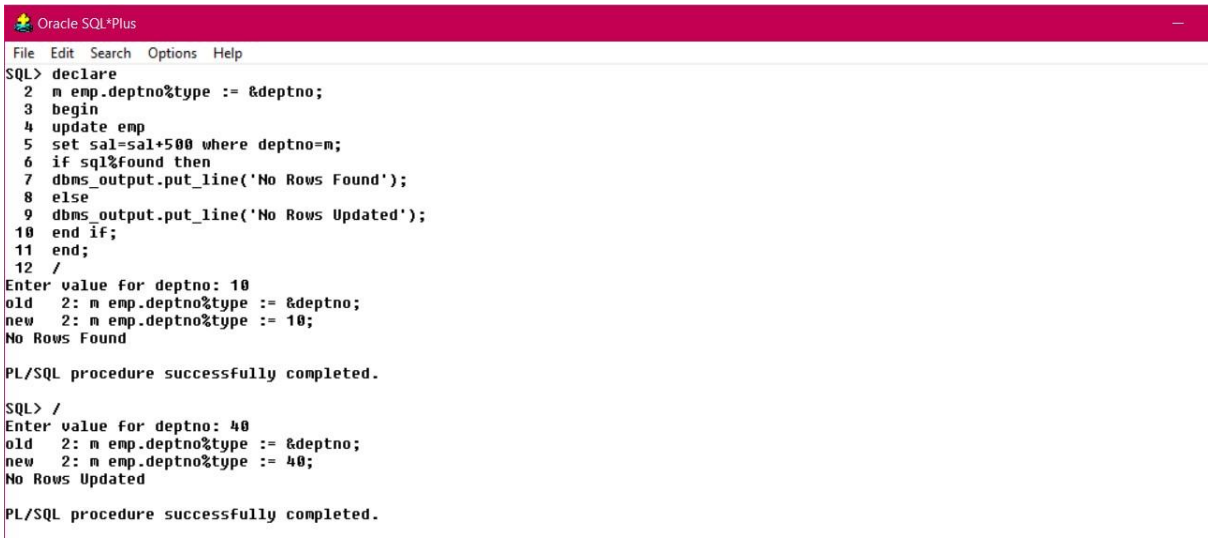
4.Write a PL/SQL block to raise an exception when divisor is 0.

5. Write a PL/SQL block to raise an exception when data record is not found in the table.

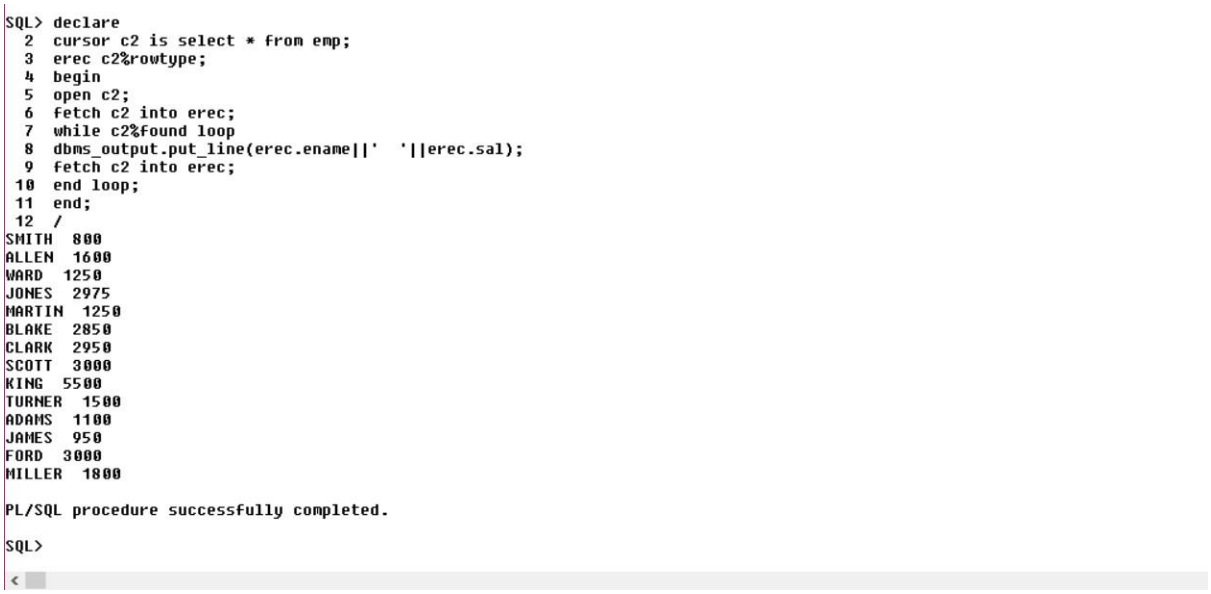


PRACTICAL NO: - 9

1. Write a PL/SQL block to UPDATE the salary of the employee by 500 whose department number is given by the user and display the number of rows if UPDATED. Otherwise DISPLAY the message 'No Records Updated' by using IMPLICIT CURSOR.



2. Write a Pl/SQL block using cursor to display the employee names and their salaries till the record is found using while loop.



3. Write a PL/SQL block to DISPLAY the employee name along with their jobs using BASIC LOOP.

```
SQL> declare
  2  erec emp%rowtype;
  3  cursor c3 is
  4  select * from emp;
  5  begin
  6  open c3;
  7  loop
  8  fetch c3 into erec;
  9  dbms_output.put_line(erec.ename||'  '||erec.job);
 10  exit when c3%notfound;
 11  end loop;
 12  end;
 13  /
SMITH  CLERK
ALLEN  SALESMAN
WARD  SALESMAN
JONES  MANAGER
MARTIN  SALESMAN
BLAKE  MANAGER
CLARK  MANAGER
SCOTT  ANALYST
KING  PRESIDENT
TURNER  SALESMAN
ADAMS  CLERK
JAMES  CLERK
FORD  ANALYST
MILLER  CLERK
MILLER  CLERK

PL/SQL procedure successfully completed.

SQL>
```

4. Write a PL/SQL block to display the employee names and their jobs using for loop.

```
SQL> declare
  2  cursor c4 is
  3  select * from emp;
  4  begin
  5  for erec in c4 loop
  6  dbms_output.put_line(erec.ename||'  '||erec.job);
  7  end loop;
  8  end;
  9  /
SMITH  CLERK
ALLEN  SALESMAN
WARD  SALESMAN
JONES  MANAGER
MARTIN  SALESMAN
BLAKE  MANAGER
CLARK  MANAGER
SCOTT  ANALYST
KING  PRESIDENT
TURNER  SALESMAN
ADAMS  CLERK
JAMES  CLERK
FORD  ANALYST
MILLER  CLERK

PL/SQL procedure successfully completed.

SQL>
```

5. Write a PL/SQL block using cursor to display the name and job of employee who are working as ‘MANAGER’. The value of the job as ‘MANAGER’ should be read by the user.

```
SQL> declare
  2  cursor c5(j varchar) is select * from emp where job=j;
  3  j varchar(25);
  4  elec c5%rowtype;
  5  begin
  6  j :='MANAGER';
  7  for elec in c5(j) loop
  8  dbms_output.put_line(elec.ename||'  '||elec.job);
  9  end loop;
 10  end;
 11  /
JONES  MANAGER
BLAKE  MANAGER
CLARK  MANAGER

PL/SQL procedure successfully completed.

SQL>
```

6. Write a PL/SQL block to display the details of the employee who are working in department 30 with the help of REF CURSOR.

```
SQL>  declare
  2   type employeee is ref cursor return emp%rowtype;
  3  c6 employeee;
  4  erec c6%rowtype;
  5  begin
  6  open c6 for select * from emp where deptno=30;
  7  loop
  8  fetch c6 into erec;
  9  exit when c6%notfound;
 10  dbms_output.put_line(erec.ename||'  '||erec.deptno||'  '||erec.job);
 11  end loop;
 12  end;
 13  /
ALLEN  30  SALESMAN
WARD  30  SALESMAN
MARTIN  30  SALESMAN
BLAKE  30  MANAGER
TURNER  30  SALESMAN
JAMES  30  CLERK

PL/SQL procedure successfully completed.

SQL> |
```

7. Write a PL/SQL block to check whether the CURSOR is OPEN or not and to DISPLAY the appropriate message using EXPLICIT CURSOR.

```
SQL> declare
  2  cursor c7 is select * from emp;
  3  erec emp%rowtype;
  4  begin
  5  open c7;
  6  if not c7%isopen then
  7  dbms_output.put_line('Cursor is not open.');
  8  end if;
  9  if c7%isopen then
 10  dbms_output.put_line('Cursor is open.');
 11  end if;
 12  end;
 13  /
Cursor is open.

PL/SQL procedure successfully completed.

SQL> |
```

8. Write a PL/SQL block to DISPLAY the employee name and their hire date using FOR LOOP.

```
SQL> declare
  2  cursor c8 is
  3  select * from emp;
  4  begin
  5  for erec in c8 loop
  6  dbms_output.put_line(erec.ename||'  '||erec.hiredate);
  7  end loop;
  8  end;
  9  /
SMITH  17-DEC-80
ALLEN  20-FEB-81
WARD  22-FEB-81
JONES  02-APR-81
MARTIN  28-SEP-81
BLAKE  01-MAY-81
CLARK  09-JUN-81
SCOTT  19-APR-87
KING  17-NOV-81
TURNER  08-SEP-81
ADAMS  23-MAY-87
JAMES  03-DEC-81
FORD  03-DEC-81
MILLER  23-JAN-82

PL/SQL procedure successfully completed.

SQL>
```

9. Write a PL/SQL block to DISPLAY the name of the employee whose name starts with ‘A’ and department is 20, where department number is passed by the user.

```
SQL> declare
  2  cursor c9(d int) is select * from emp where ename like 'A%' and deptno=d;
  3  d int;
  4  erec emp%rowtype;
  5  begin
  6  d :=20;
  7  for erec in c9(d) loop
  8  dbms_output.put_line('Name' ||'  '||'DeptNo');
  9  dbms_output.put_line(erec.ename||'  '||erec.deptno);
 10  exit when c9%notfound;
 11  end loop;
 12  end;
 13  /
Name  DeptNo
ADAMS  20

PL/SQL procedure successfully completed.

SQL>
```

10. Write a PL/SQL block to DISPLAY the employee number and employee name who are working as ‘MANAGER’ and earning salary greater than 1000.

```
SQL> declare
  2  cursor c10 is select * from emp where job='MANAGER' and sal>1000;
  3  erec emp%rowtype;
  4  begin
  5  open c10;
  6  loop
  7  fetch c10 into erec;
  8  dbms_output.put_line(erec.empno||'  '||erec.ename);
  9  exit when c10%notfound;
 10  end loop;
 11  end;
 12  /
7566  JONES
7698  BLAKE
7782  CLARK
7782  CLARK

PL/SQL procedure successfully completed.

SQL>
```