INTERNATIONAL INSTITUTE OF PROFESSIONAL STUDY

NAME – Palak Ved

CLASS –M.tech (IT)7th sem

ROLL NO. – IT-2K20-39

SUBJECT –PLSQL LAB ASSIGNMENT

ENROLLMENT NO.- DE2002042

Date – 7/11/2023

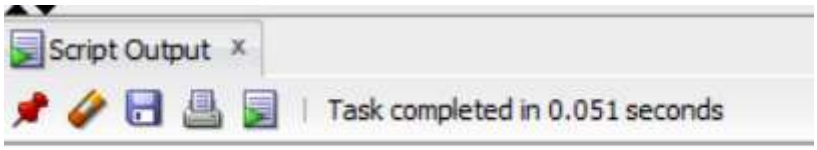
Signature

BASIC PLSQL PROGRAMS

1. Hello World

```
/*simple hello world program*/
declare
msg varchar2(20) := 'hello world' ;
BEGIN
dbms_output.put_line(msg);
END;
/
```

OUTPUT



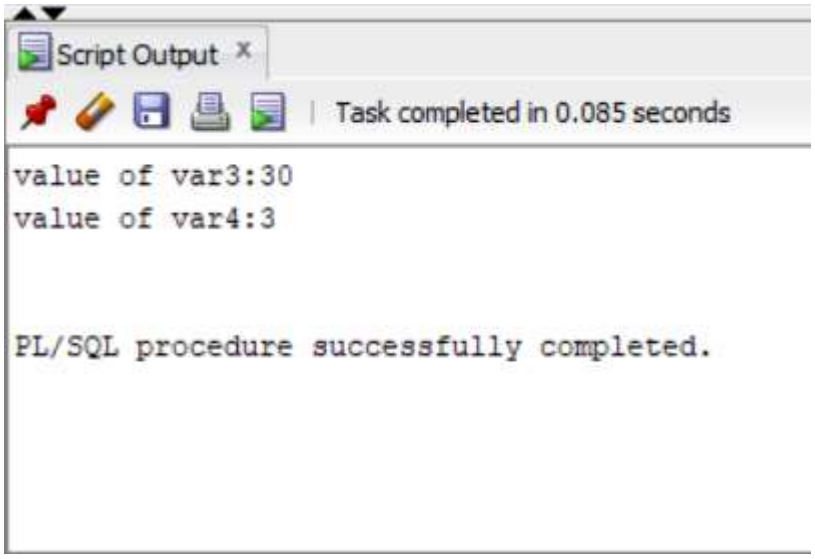

```
PL/SQL procedure successfully completed.

hello world


PL/SQL procedure successfully completed.
```

2. Adding two integer

```
Declare
var1 integer :=10;
var2 integer :=20;
var3 integer ;
var4 real ;
BEGIN
var3 :=var1 +var2;
dbms_output.put_line('value of var3:' || var3);
var4 :=var3/10;
dbms_output.put_line('value of var4:' || var4);
end;
/
```

OUTPUT

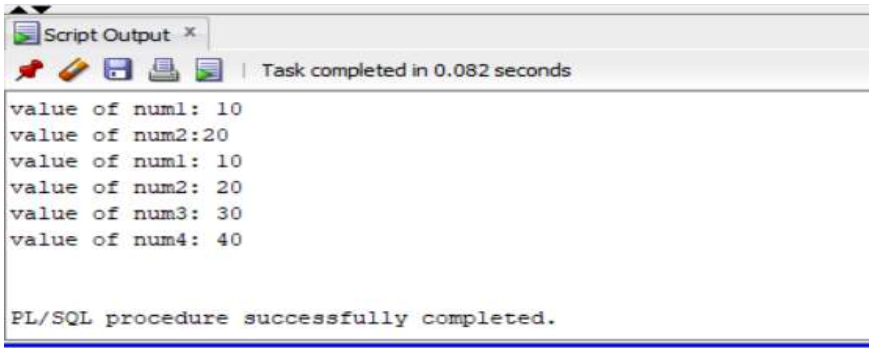
v



3.Global &Local Variable

```
DECLARE ----Global program
num1 number :=10;
num2 number :=20;
BEGIN
dbms_output.put_line('value of num1: '| | num1);
dbms_output.put_line('value of num2:' ||num2);
DECLARE ----local variable
num3 number :=30;
num4 number :=40;
BEGIN
dbms_output.put_line('value of num1: '| | num1);
dbms_output.put_line('value of num2: '| | num2);
dbms_output.put_line('value of num3: '| | num3);
dbms_output.put_line('value of num4: '| | num4);
END;
END;
```

OUTPUT



4.Constant Variable

```
set serveroutput on
DECLARE
---- constant declaration
pi constant number := 3.14;
---- other declaration
radius number;
diameter number;
circumferrence number;
area number;
BEGIN
radius :=10.5;
diameter :=radius*2;
dbms_output.put_line('diameter := ' || diameter);
circumferrence :=2*pi*radius;
dbms_output.put_line('circumferrence := ' || circumferrence);
area :=pi*radius*radius;
dbms_output.put_line('area of circle := ' || area);
END;
```

OUTPUT

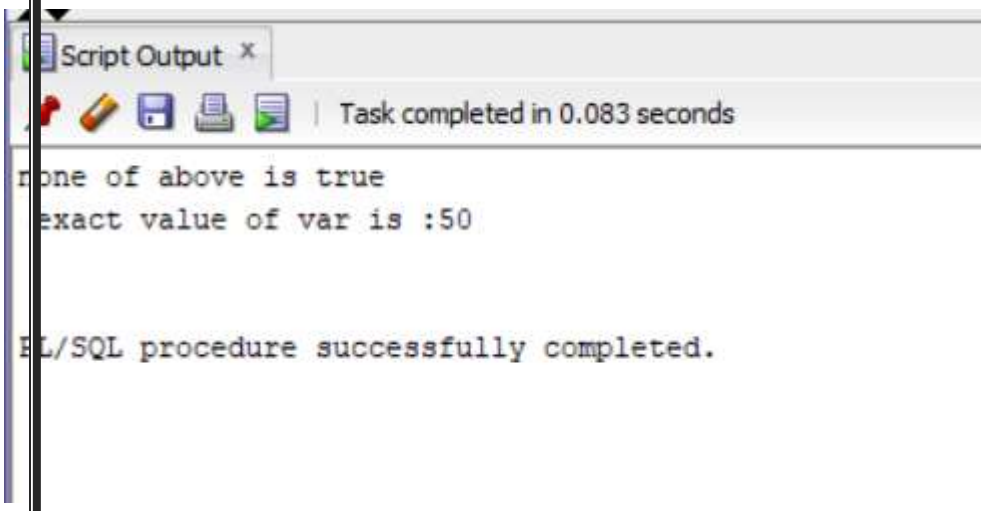
```
Script Output
Task completed in 0.108 seconds
diameter := 21
circumferrence := 65.94
area of circle := 346.185


PL/SQL procedure successfully completed.
```

5.IF – Else

```
DECLARE
var number(3) :=50;
BEGIN
if(var =10)
then
dbms_output.put_line('value of var is : 10');
elsif(var=20) then
dbms_output.put_line('value of var is : 20');
elsif(var=30) then
dbms_output.put_line('value of var is : 30');
else
    dbms_output.put_line('none of above is true');
END if;
dbms_output.put_line(' exact value of var is :' || var);
```

END;

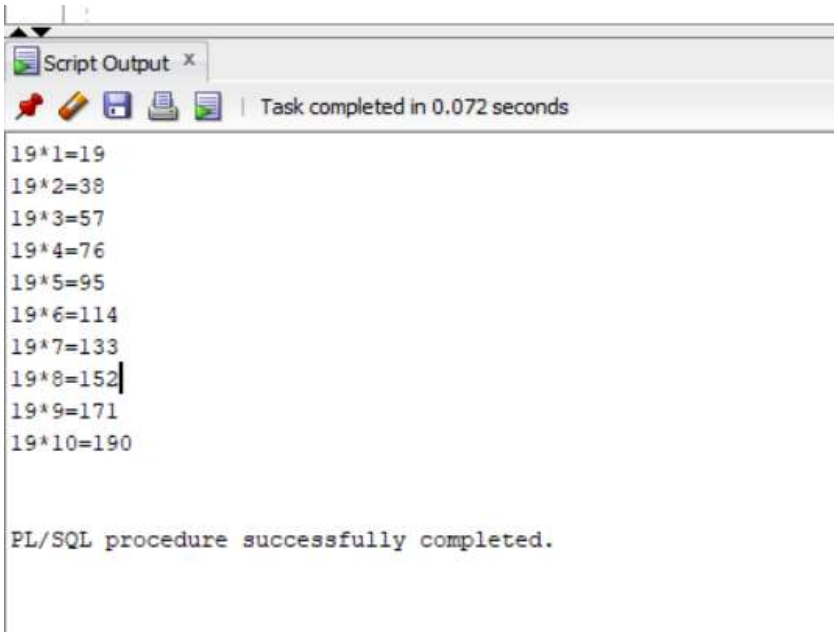
OUTPUT


Script Output
Task completed in 0.083 seconds
none of above is true
exact value of var is :50
PL/SQL procedure successfully completed.

6. While Loop

```
DECLARE

counter number(20) := 1;

a_result number;

Begin

while(counter<=10)loop

a_result := 19*counter;

dbms_output.put_line('19' || '*' || counter || '=' || a_result);

counter := counter+1;

END loop;

END;
```

OUTPUT



7. For Loop

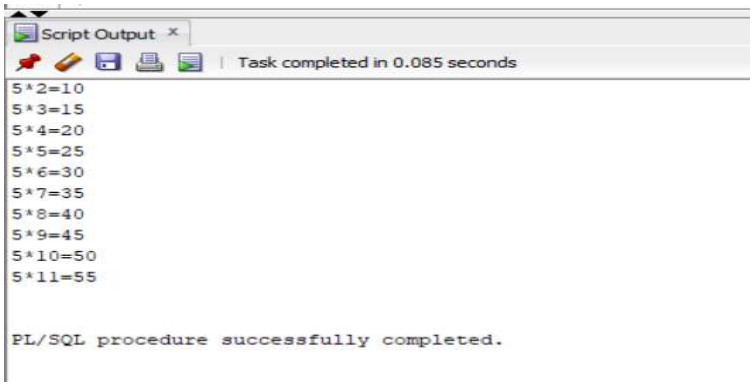
```
set serveroutput on
begin
for i  in reverse 1..10 loop
dbms_output.put_line(i);
end loop;
end;
```

OUTPUT

8. Simple Loop

```
---- simple for loop

set serveroutput on

declare

res number;

counter number :=0;

begin

loop counter :=counter+1;

res := 5*counter;

dbms_output.put_line(5 || '*' || counter || '=' || res);

if(counter>10) then

exit;

end if;

end loop;

end;

/
```

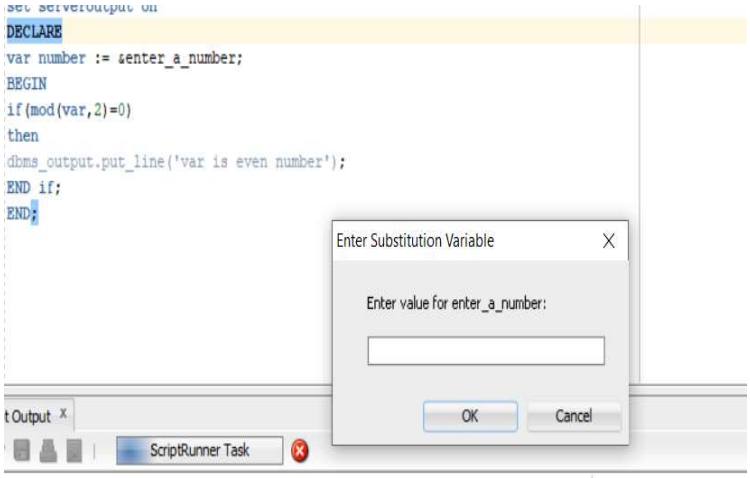
OUTPUT



9. User Input

```
set serveroutput on

DECLARE

var number := &enter_a_number;

BEGIN

if(mod(var,2)=0)

then

dbms_output.put_line('var is even number');

END if;

END;
```

OUTPUT

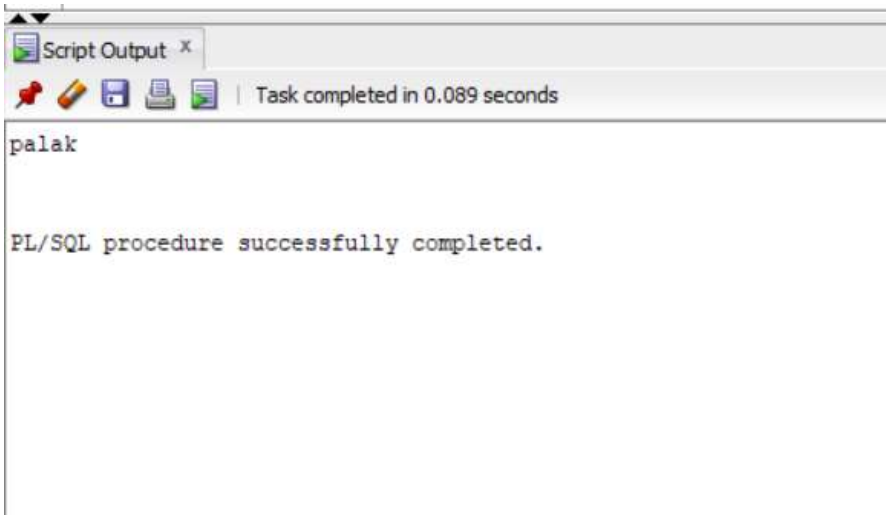


10.Bind Variable

---- Bind variable (we can define anywhere in program for this we have to use variable keyword and use : before variable name when intializing

```
variable a varchar2(10);

 ---exec : a := 'hello';


begin

:a :='palak';

dbms_output.put_line( :a);

end;

/
```

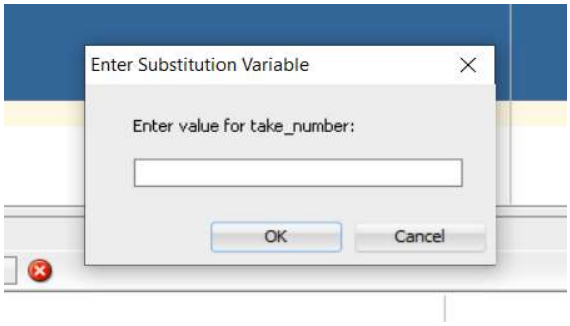
OUTPUT

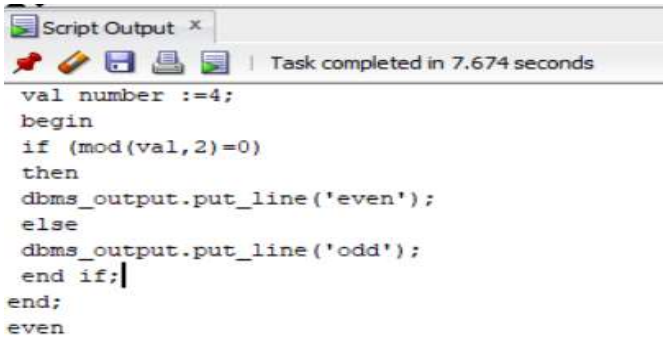


II. Even /Odd using Plsql

```
declare
 val number :=&take_number;
 begin
 if (mod(val,2)=0)
 then
 dbms_output.put_line('even');
 else
 dbms_output.put_line('odd');
 end if;
end;
/
```

OUTPUT

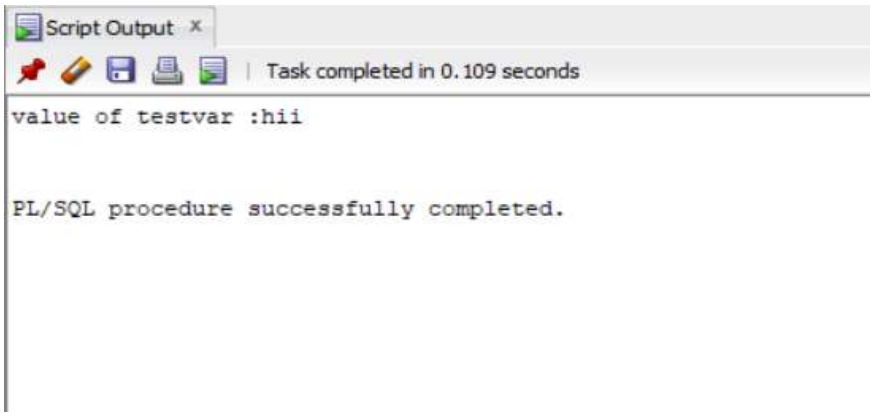




12. Varchar in plsql

```
set serveroutput on
DECLARE
/*cant start with number, special char, _*/
/* we can only use # and $  special char at end of varname*/
testvar varchar(20);
BEGIN
testvar := 'hii';
dbms_output.put_line('value of testvar :' || testvar);
END;
```

OUTPUT



TRIGGERS

/* USES OF TRIGGER */

------- Gain strong

---Impliment complex business rule

--- impliment complex validation

--- Automatically generate valurres

---for auditing

1. DDL TRIGGER

---using ddl trigger we can track changes to data base

#MAIN TABLE

```
create table S_schema_audit2(

ddl_date date,

ddl_user varchar(20),

obj_created varchar2(30),

obj_name varchar(20),

operation varchar2(20) );

create or replace trigger S_trigger_schema

after ddl on schema

begin

insert into S_schema_audit2
values(SYSDATE,SYS_CONTEXT('userrenv','current_user'),ora_dict_obj_type,ora_dict_
obj_name,ora_sysevent);

end;
```

umns | Data | Model | Constraints | Grants | Statistics | Triggers | Flashback | Dependencies | Details | Partitions | Indexes | SQL

Sort.. | Filter:

	DDL_DATE	DDL_USER	OBJ_CREATED	OBJ_NAME	OPERATION
1	09-OCT-23	(null)	TABLE	S_SCHEMA_AUDIT	DROP
2	09-OCT-23	(null)	TABLE	DUMMY	CREATE
3	09-OCT-23	(null)	TABLE	S_DUMMY	CREATE
4	09-OCT-23	(null)	TABLE	S_DUMMY	TRUNCATE
5	09-OCT-23	(null)	TABLE	S_DUMMY	DROP
6	09-OCT-23	(null)	TABLE	S_EMP3	CREATE
7	09-OCT-23	(null)	TABLE	S_EMP3	DROP
8	09-OCT-23	(null)	TABLE	S_EMP3	CREATE
9	11-OCT-23	(null)	PROCEDURE	S_PRO_NAME	CREATE
10	11-OCT-23	(null)	PROCEDURE	S_PRO_NAME	CREATE
11	11-OCT-23	(null)	PROCEDURE	S_PRO_NAME	CREATE
12	11-OCT-23	(null)	PROCEDURE	S_PRO_NAME	CREATE
13	11-OCT-23	(null)	PROCEDURE	S_PRO_NAME	CREATE
14	11-OCT-23	(null)	PROCEDURE	S_PRO_NAME	CREATE

2.DML TRIGGER

```
create table S_dept(
d_id varchar2(5) not null,
d_name varchar2(30),
d_salary number,

CONSTRAINT d_pk2 primary key(d_id)
);
insert into S_dept values(1001,'Raj',240000);
insert into S_dept values(1002,'Kunal',17000);
insert into S_dept values(1003,'Mukesh',17000);
insert into S_dept values(1004,'Anushka',9000);

create or replace trigger trigger_dept
after insert on S_dept
begin
dbms_output.put_line('record inserted');
end;
/insert into S_dept values(1009,'rohit',20000);
drop table S_dept;
create or replace trigger trigger10
before delete on S_dept
begin
dbms_output.put_line('drop');
end;
```

OUTPUT

Columns | Data | Model | Constraints | Grants

Sort.. | Filter

	D_ID	D_NAME	D_SALARY
1	1001	Raj	240000
2	1002	Kunal	17000
3	1003	Mukesh	17000
4	1004	Anushka	9000

3.DELETE DML TRIGGER

#MAIN TABLE

```
create table S_employee(
e_id varchar2(5) not null,
e_name varchar2(30),
e_salary number,


CONSTRAINT e_pk primary key(e_id)
);
insert into S_employee values(1001,'palak',20000);
insert into S_employee values(1002,'riya',30000);
insert into S_employee values(1003,'pooja',40000);
insert into S_employee values(1005,'po0nam',50000);
```

Columns | Data | Model | Constraints | Grants | Statistics | Trigg

Sort.. | Filter:

	E_ID	E_NAME	E_SALARY
1	1001	Ruhi	240000
2	1002	Riya	17000
3	1003	Pooja	17000
4	1004	Poonam	9000
5	1005	Priya	6000
6	1006	Princy	4800
7	1007	Prem	4800
8	1008	Pallavi	4200

BACKUP TABLE

```
create table S_emp_backup(
e_name varchar2(30),
e_salary number
);
```

TRIGGER

```
delete from S_EMPLOYEE;

create or replace trigger S_trigger1

before delete on S_EMPLOYEE

for each row

begin

insert into S_EMP_BACKUP values(:old.e_name , :old.e_salary);

end;
```

OUTPUT

Columns | Data | Model | Constraints | Grants | Statistics | Triggers

Sort.. | Filter:

	E_NAME	E_SALARY
1	palak	20000
2	riya	30000
3	pooja	40000
4	po0nam	50000

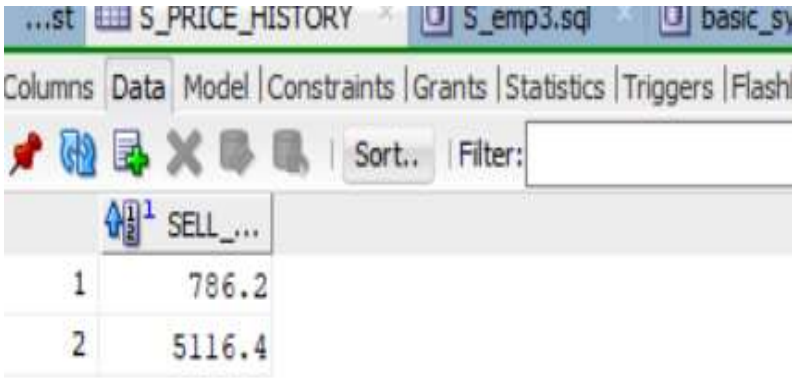
4.Insert Trigger

#product table

```
create table S_product(

p_id varchar2(5) not null,

p_name varchar2(30),

m_name varchar(50),

p_rate float,

sell_price float,

p_desp varchar2(20),

cat varchar2(20),

 CONSTRAINT pro_pk primary key(p_id)

);

insert into S_product values('p001','lux_soap','lux_international',15.67,20.43,'soap','grosery');


insert into S_product values('p002','copy','times_copy_org',14.8,16.4,'copy','book');

insert into S_product values('p003','marry_biscuit','marigold_org',2218.8,2220.4,'biscuit','grosery');

insert into S_product
values('p004','mitsubisi_pen','pen_international',2015.67,2120.43,'pen','stationary');

insert into S_product values('p005','learn_books','book_world',5104.8,5116.4,'book','book');

insert into S_product values('p006','tata_steel','tata_org',5118.8,5120.4,'tata','Electronic');

insert into S_product values('p007','seagate hdd','seaget_world',5104.8,5116.4,'HDD1034','Electronic');
```

```
# product backup

create table S_price_history(

price float

);

# TRIGGER

delete from S_product2;

create or replace trigger S_trigger2

before delete on S_product2

for each row

begin

insert into S_price_history_2 values(:old.sell_price);

end;
```

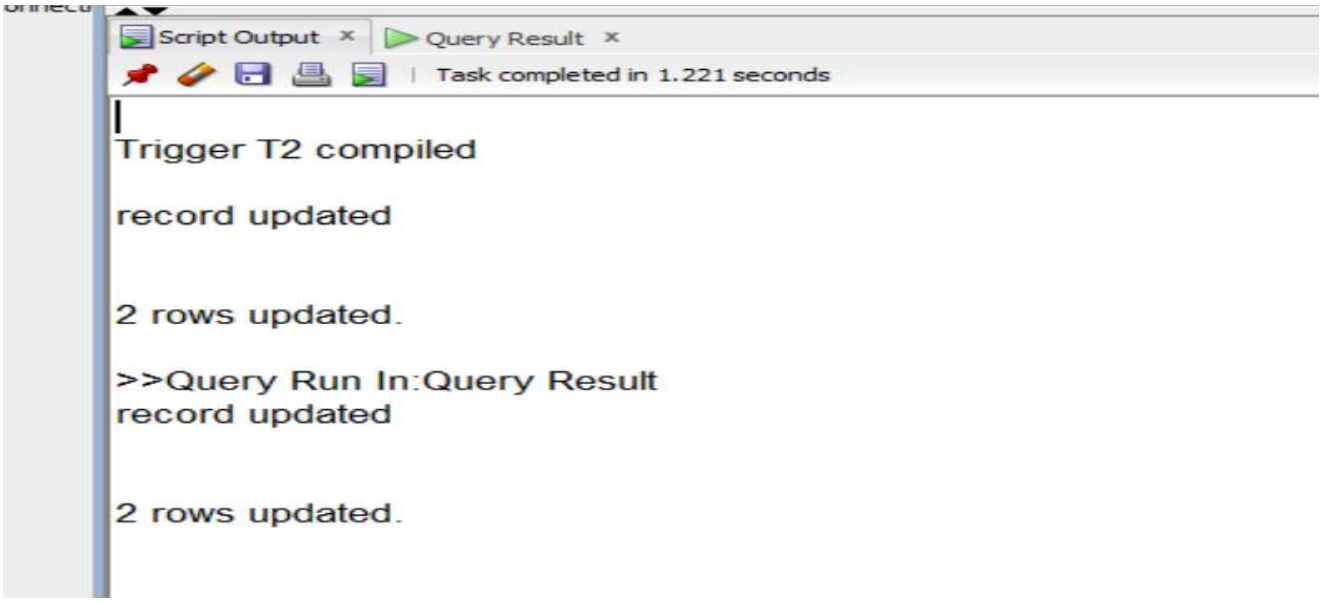
OUTPUT



6. BEFORE UPDATE TRIGGER

```
set serveroutput on;
create table emp(
e_id number,
e_name varchar(20),
salary number
);
insert into emp values(001,'abc',24000);
insert into emp values(002,'pqr',17000);
insert into emp values(003,'xyz',17000);
insert into emp values(004,'rst',9000);
insert into emp values(005,'jkl',6000);
insert into emp values(006,'rst',4200);
insert into emp values(007,'ijk',24000);
create or replace trigger t2
before update on emp
 begin
   dbms_output.put_line('record updated');
   end;
/update emp set salary=salary*1.2 where salary<10000;
select * from emp;
update emp set salary=salary*1.2 where e_id=007;
```

OUTPUT


Script Output
Query Result
Task completed in 1.221 seconds
Trigger T2 compiled
record updated
2 rows updated.
>>Query Run In:Query Result
record updated
2 rows updated.

ANCHORED TAG

```
create table S_emp3(
emp_id number,
first_name varchar2(20),
last_name varchar2(20),
email varchar2(30),
phone number,
salary number

);



insert into S_emp3 values(01, 'Neha' , 'khan' , 'neha@gmail.com' ,9930267 , 2000);
insert into S_emp3 values(02, 'Anjali' , 'gwala' , 'ag@gmail.com' ,12345678 , 5000);
insert into S_emp3 values(03, 'Palak' , 'ved' , 'palak@gmail.com' ,1234575 , 6000);
insert into S_emp3 values(04, 'Sonu' , 'sethi' , 'sonu@gmail.com' ,5742368 , 3000);

/* use of anchored type*/

declare
f_name
S_emp3.first_name%type;
```

```
begin
select first_name into f_name
from S_emp3 where emp_id = 2;
dbms_output.put_line(f_name);
end;/
```

OUTPUT

```
1 row inserted.

1 row inserted.

PL/SQL procedure successfully completed.

Palak
```

PROCEDURE IN PLSQL

1. Procedure to find max

```
create table S_emp_pro(
d_id number,
salary number
);
insert into S_emp_pro values(01 , 20000);
insert into S_emp_pro values(02 , 30000);
insert into S_emp_pro values(03 , 40000);
insert into S_emp_pro values(04 , 50000);
---
create or replace procedure S_e_proc(id_d number , s number)
IS
begin
update S_emp_pro
set salary = salary*s
where d_id = id_d;
end;
/
exec S_e_proc(2,200);
create or replace procedure S_e_proc(id_d number , s number)
IS
begin
update S_emp_pro
set salary = salary*s
where d_id = id_d;
end;
/
exec S_e_proc(2,200);
```

Columns | Data | Model | Constraints | Grants | Statistics | Triggers | Flashback | Dependencies | Details | Partiti

Sort.. | Filter:

	D_ID	SALARY
1	1	20000
2	2	240000000000
3	3	40000
4	4	50000
5	1	20000
6	2	1200000000
7	3	40000
8	4	50000
9	1	20000
10	2	6000000
11	3	40000
12	4	50000

```
/*out type*/
create procedure S_max_pro(x int, y int , z out int)
IS
begin
if x>y
then
z :=x ;
else
z :=y;
end if;
end S_max_pro;
/
declare
a int;
begin
S_max_pro(10,5,a);
dbms_output.put_line(a);
end;
```

```
*Action:
10


PL/SQL procedure successfully completed.
```

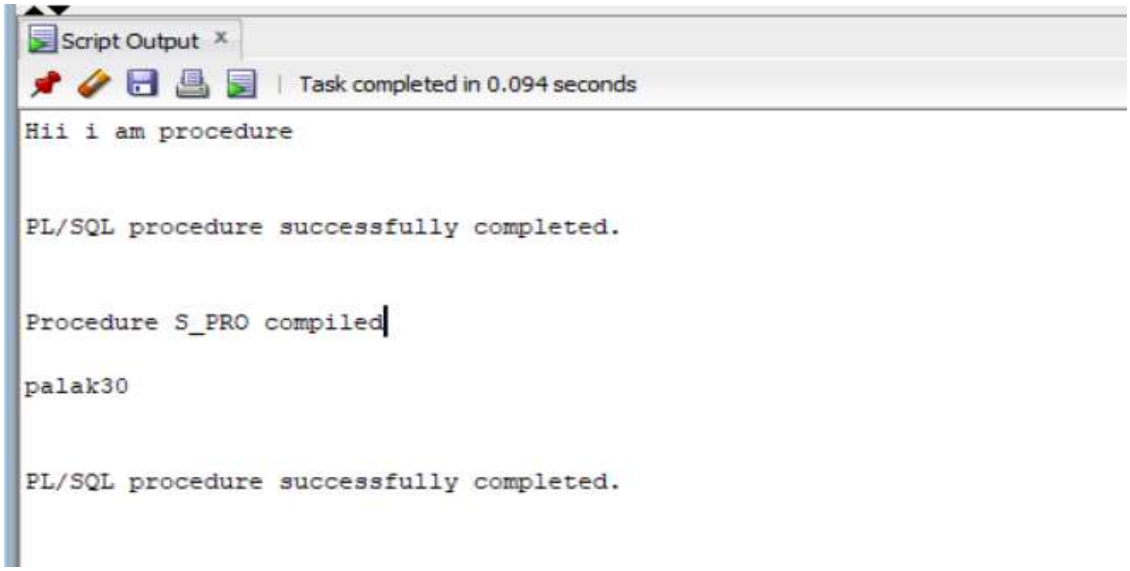
/* In out type*/

```
create or replace procedure S_square(z in out int)
Is
begin
z :=z*z;
end S_square;
/
declare
a int:=3;
begin
S_square(a);
dbms_output.put_line(a);
end;
/
```

```
Procedure S_SQUARE compiled

9


PL/SQL procedure successfully completed.
```

BASICS OF PROCEDURE

```
set serveroutput on;
Create or replace procedure S_pro_name
Is
begin
dbms_output.put_line('Hii i am procedure');
end S_pro_name;
/
/*three ways to call procedure*/
/* 1. begin
S_pro_name;
end;*/
/*  2. exec  S_pro_name;*/
--3.
execute S_pro_name;
create or replace procedure S_pro
IS
name varchar2(50) :='palak';
age number :=30;
begin
dbms_output.put_line(name||age);
end S_pro;
/
exec S_pro;
```



GREATER NUM PROCEDURE

```
create or replace procedure  S_great(numx in int , numy in int, z in out int)
IS
num1 number := numx ;
num2 number := numy ;
begin
if(num1 > num2) then
z := num1;
dbms_output.put_line(z);
else
z := num2;
dbms_output.put_line(z);
end if;
end S_great ;/
declare
a number :=3;
begin
S_great(30,40,a);
end;
```

```
Procedure S_GREAT compiled

40


PL/SQL procedure successfully completed.
```

FUNCTION S IN PLSQL

I Function to add nums

```
create or replace function S_addd(num1 in number , num2 in number)return number
is
num3 number;
begin
num3 := num1 +num2;
return num3;
end S_addd ;
/
begin
dbms_output.put_line(S_addd(3,20));
end;
```

OUTPUT


```
Function S_ADDD compiled

23


PL/SQL procedure successfully completed.
```

2. UPDATE SALARY USING FUNCTION

```
create table S_empfunc(
e_id number,
e_name varchar(30),
e_salary number
);
insert into S_empfunc values(1001, 'palak' , 20000);
insert into S_empfunc values(1002, 'Riya' , 30000);
insert into S_empfunc values(1003, 'Anjali' , 50000);
insert into S_empfunc values(1004, 'Chana' , 20000);
insert into S_empfunc values(1005, 'Golu' , 70000);
insert into S_empfunc values(1006, 'Saloni' , 10000);
insert into S_empfunc values(1007, 'johm' ,70000);

create or replace function S_uptsalary(id in number, amt in number)return number
is
d number;
begin
update S_empfunc set e_salary = e_salary+amt
where e_id = id;
select e_salary into d from  S_empfunc
where e_id =id;
return d;
end S_uptsalary;
/
declare
d number;
begin
d := S_uptsalary(1001,2000);
end;
```

OUTPUT

...st | S_EMPFUNC | S_emp3.sql | basic_syntax_trigger.sql | employee_table.

Columns | Data | Model | Constraints | Grants | Statistics | Triggers | Flashback | Dependencies | Details | Parti

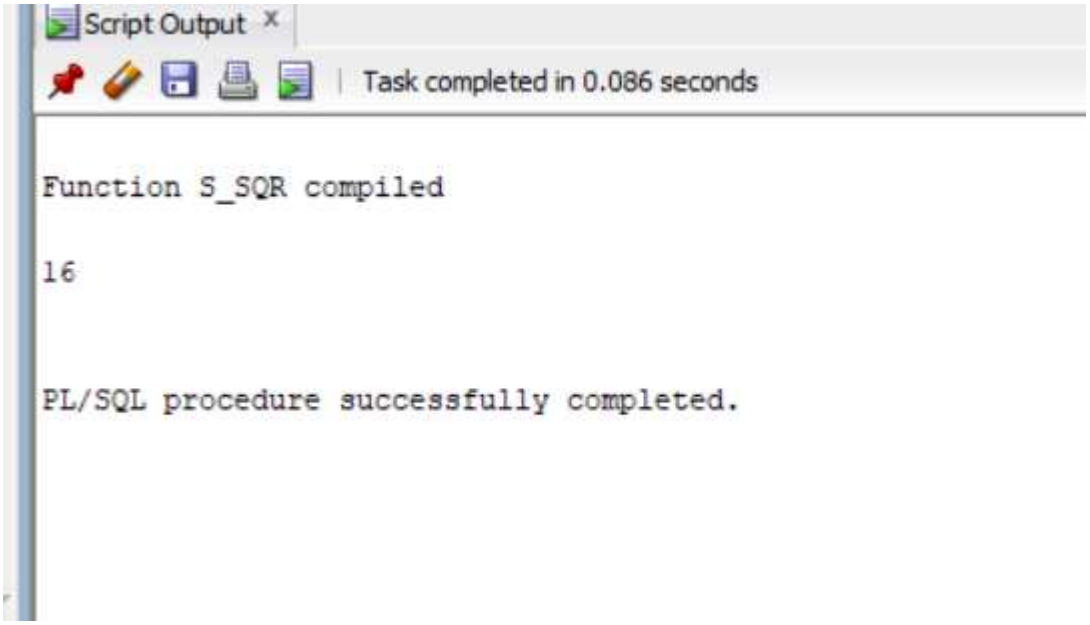
Sort.. | Filter:

	E_ID	E_NAME	E_SALARY
1	1001	palak	22000
2	1002	Riya	30000
3	1003	Anjali	50000
4	1004	Chana	20000
5	1005	Golu	70000
6	1006	Saloni	10000
7	1007	johm	70000
8	1001	palak	20000
9	1002	Riya	30000
10	1003	Anjali	50000
11	1004	Chana	20000
12	1005	Golu	70000
13	1006	Saloni	10000
14	1007	johm	70000

4.FUNCTION TO FIND SQUARE

```
create or replace function  S_sqr(a number)return number
is
sqr number;
begin
sqr :=a*a;
return sqr;
end S_sqr;
/
begin
dbms_output.put_line(S_sqr(4));
end;
```

OUTPUT

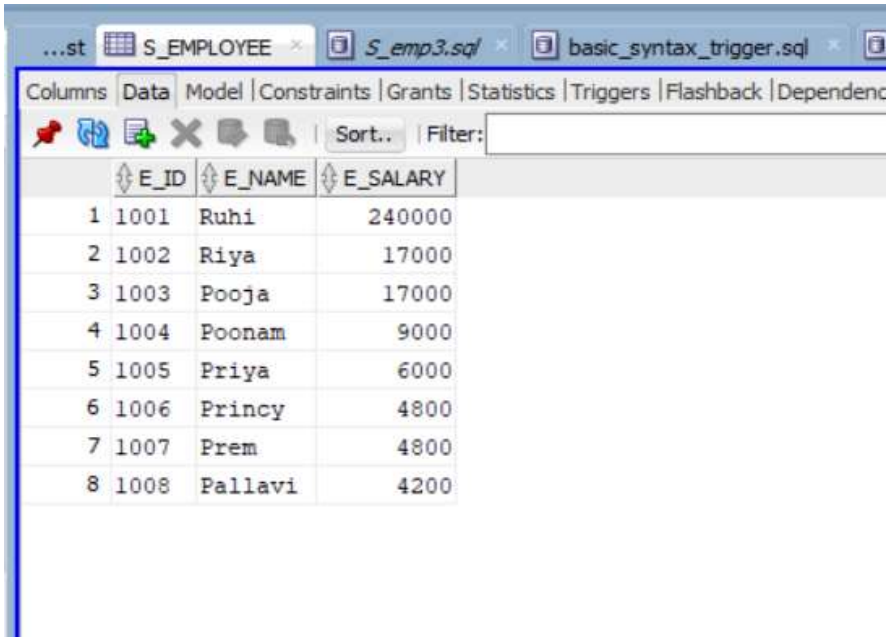


CURSOR IN PLSQL

1.

 # Employee table

```
create table S_employee(
e_id varchar2(5) not null,
e_name varchar2(30),
e_salary number,

CONSTRAINT e_pk primary key(e_id)
);
insert into S_employee values(1001,'palak',20000);
insert into S_employee values(1002,'riya',30000);
insert into S_employee values(1003,'pooja',40000);
insert into S_employee values(1005,'po0nam',50000);
```



	E_ID	E_NAME	E_SALARY
1	1001	Ruhi	240000
2	1002	Riya	17000
3	1003	Pooja	17000
4	1004	Poonam	9000
5	1005	Priya	6000
6	1006	Princy	4800
7	1007	Prem	4800
8	1008	Pallavi	4200

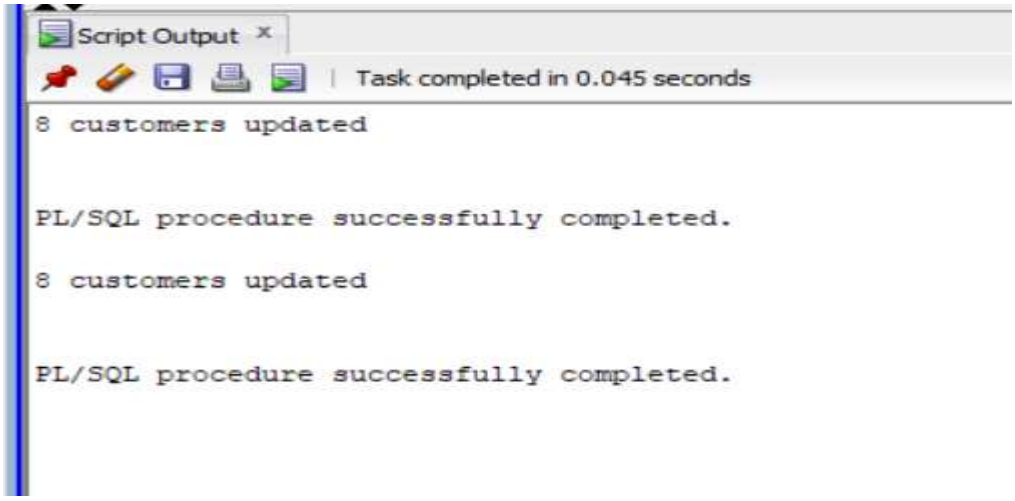
/*EXPLICIT CURSOR*/

```
set serveroutput on;
declare
c_id customer.id%type;
c_name customer.name%type;
c_address customer.address%type;
cursor c1 is
select id,name,address from customer;
begin
open c1;
loop fetch c1 into c_id,c_name,c_address;
exit when c1%notfound;
dbms_output.put_line(c_id||c_name||c_address);
end loop;
close c1;
end;
/
select * from customer;
```

 # CURSOR

```
DECLARE
   total_rows number(2);
BEGIN
```

```
  UPDATE  S_EMPLOYEE
  SET E_salary = E_salary + 5000;
  IF sql%notfound THEN
    dbms_output.put_line('no customers updated');
  ELSIF sql%found THEN
    total_rows := sql%rowcount;
    dbms_output.put_line( total_rows || ' customers updated ');
  END IF;
END;
/
```



...st | S_EMPLOYEE | S_emp3.sql | basic_syntax_trigger.sql

Columns | Data | Model | Constraints | Grants | Statistics | Triggers | Flashback | Deper

Sort.. | Filter:

	E_ID	E_NAME	E_SALARY
1	1001	Ruhi	262000
2	1002	Riya	39000
3	1003	Pooja	39000
4	1004	Poonam	31000
5	1005	Priya	28000
6	1006	Princy	26800
7	1007	Prem	26800
8	1008	Pallavi	26200