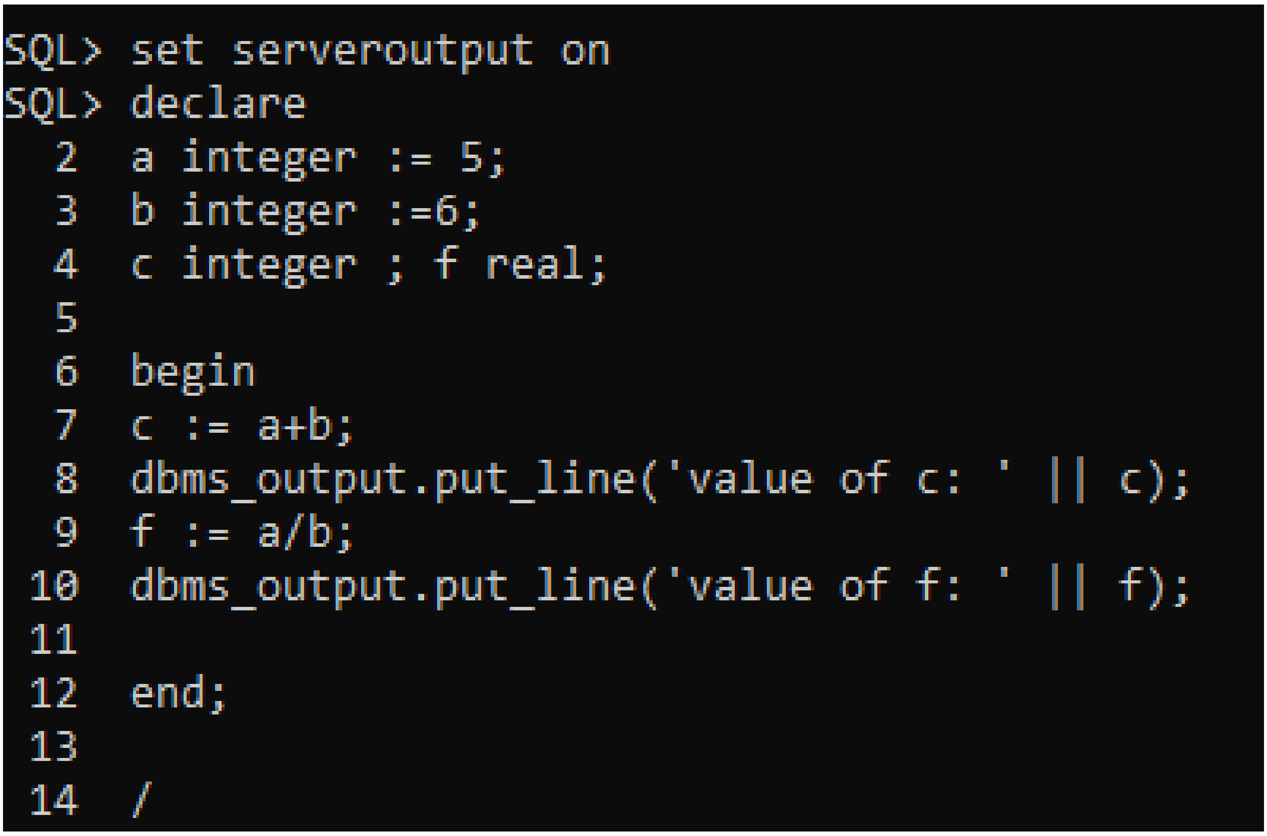
practical no.1

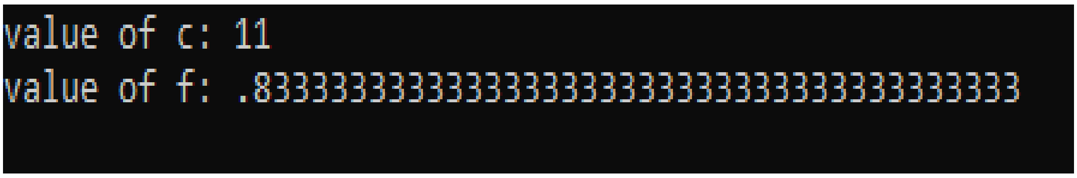
Basic of plsql

code:



```
SQL> set serveroutput on
SQL> declare
a integer := 5;
b integer :=6;
c integer ; f real;

begin
c := a+b;
dbms_output.put_line('value of c: ' || c);
f := a/b;
dbms_output.put_line('value of f: ' || f);

end;

/
```

Output:

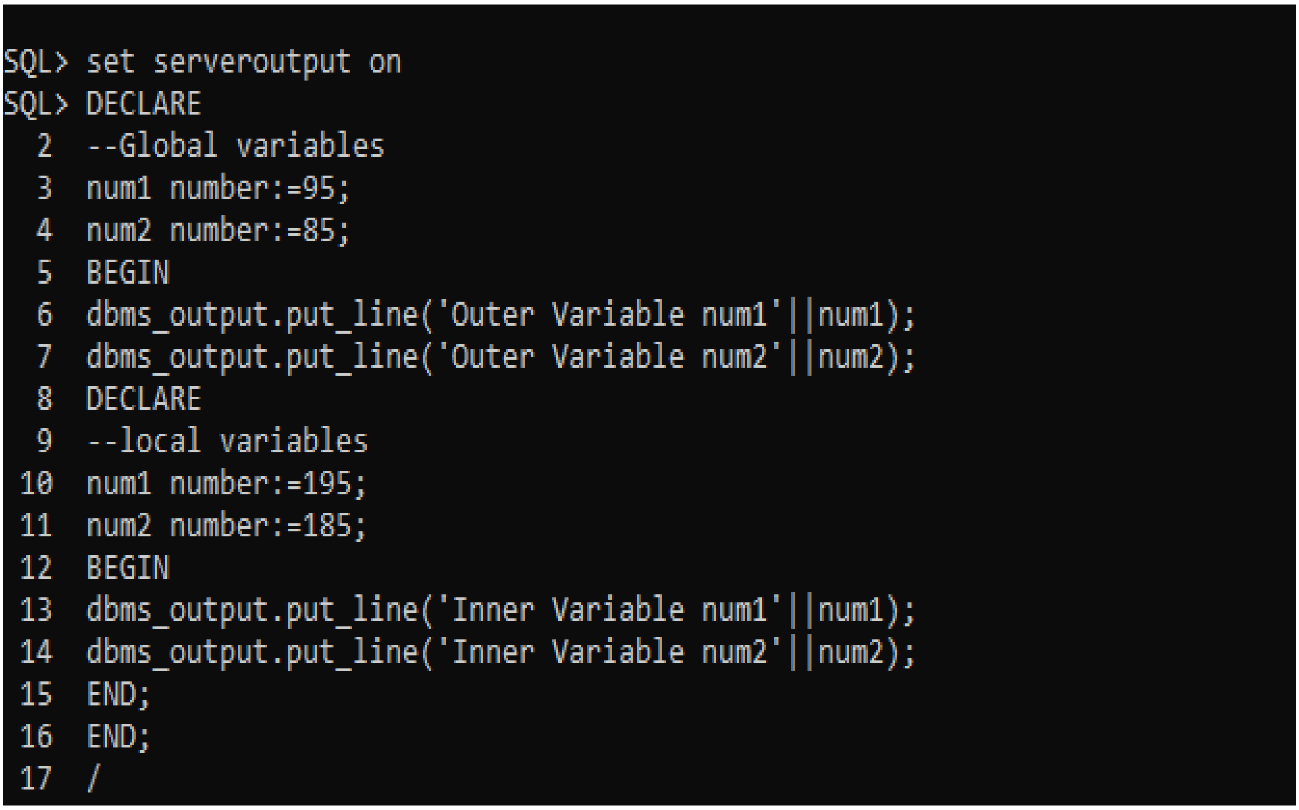


```
value of c: 11
value of f: .8333333333333333333333333333333333333333
```

Practical no.2

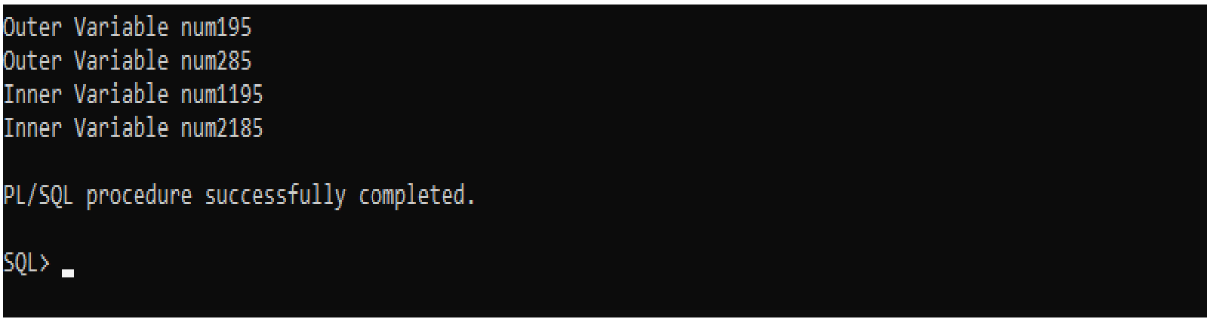
Declaring variables(local and global)

Code:



```
SQL> set serveroutput on
SQL> DECLARE
--Global variables
num1 number:=95;
num2 number:=85;
BEGIN
dbms_output.put_line('Outer Variable num1'||num1);
dbms_output.put_line('Outer Variable num2'||num2);
DECLARE
--local variables
num1 number:=195;
num2 number:=185;
BEGIN
dbms_output.put_line('Inner Variable num1'||num1);
dbms_output.put_line('Inner Variable num2'||num2);
END;
END;
/
```

Output:



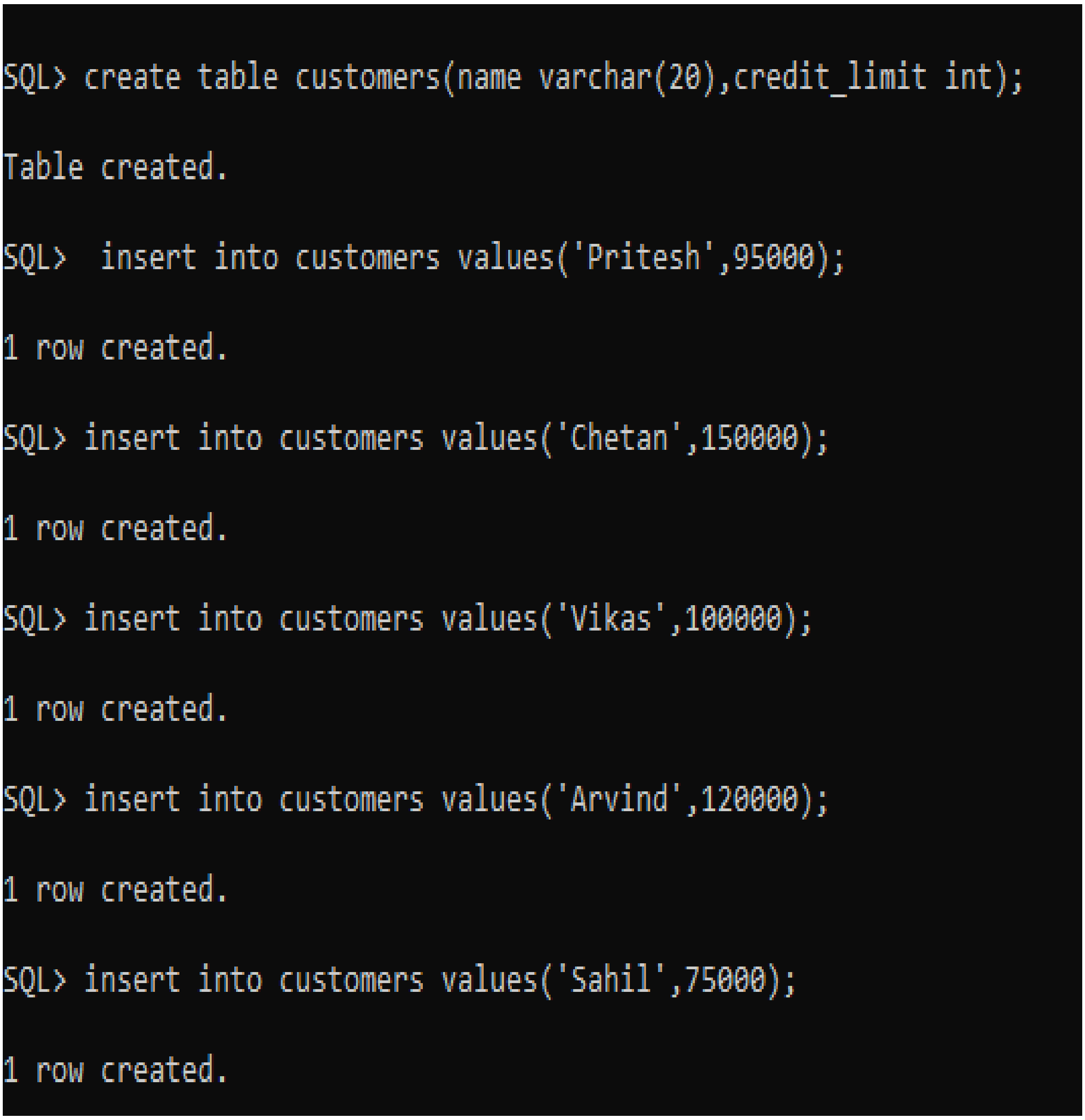
```
Outer Variable num195
Outer Variable num285
Inner Variable num1195
Inner Variable num2185

PL/SQL procedure successfully completed.

SQL>
```

Practical No. 3

varry

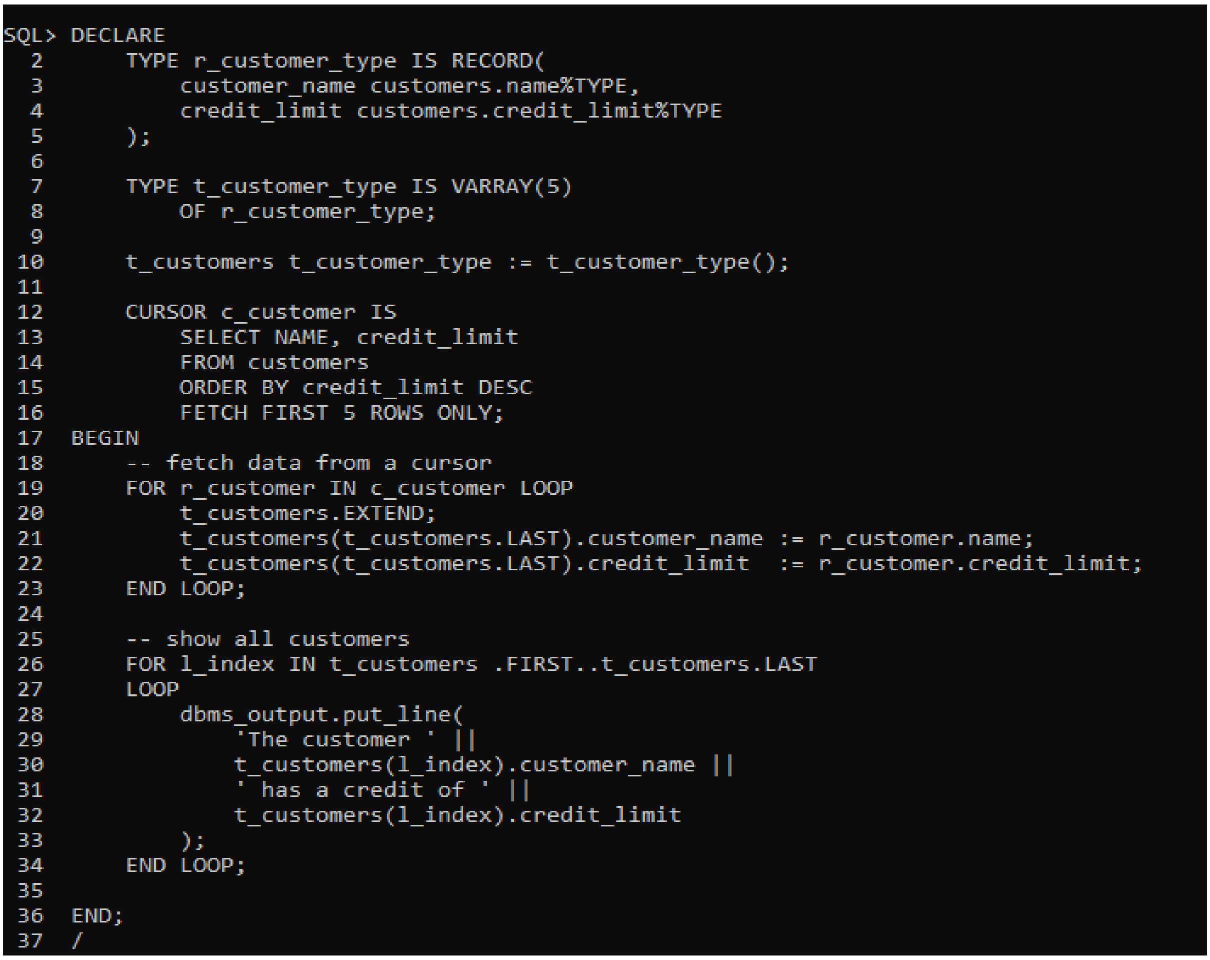
Code:

#creating a table



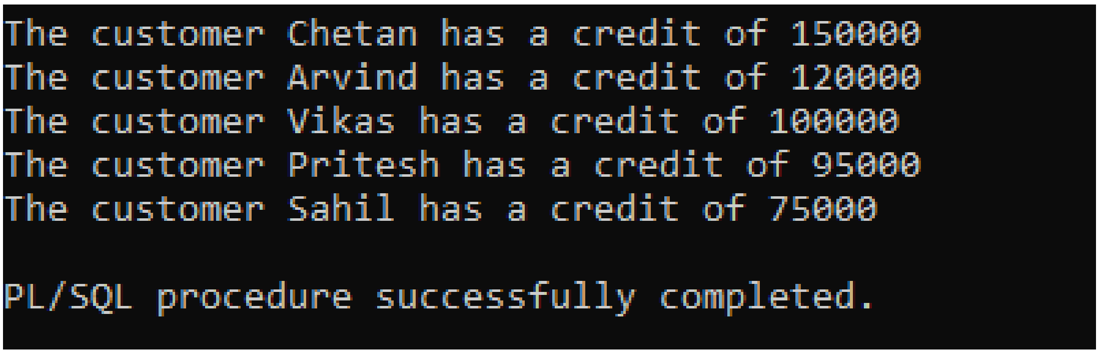
```
SQL> create table customers(name varchar(20),credit_limit int);

Table created.

SQL>  insert into customers values('Pritesh',95000);

1 row created.

SQL> insert into customers values('Chetan',150000);

1 row created.

SQL> insert into customers values('Vikas',100000);

1 row created.

SQL> insert into customers values('Arvind',120000);

1 row created.

SQL> insert into customers values('Sahil',75000);

1 row created.
```

VARRY



```
SQL> DECLARE
    TYPE r_customer_type IS RECORD(
        customer_name customers.name%TYPE,
        credit_limit customers.credit_limit%TYPE
    );

    TYPE t_customer_type IS VARRAY(5)
        OF r_customer_type;

    t_customers t_customer_type := t_customer_type();

    CURSOR c_customer IS
        SELECT NAME, credit_limit
        FROM customers
        ORDER BY credit_limit DESC
        FETCH FIRST 5 ROWS ONLY;
BEGIN
    -- fetch data from a cursor
    FOR r_customer IN c_customer LOOP
        t_customers.EXTEND;
        t_customers(t_customers.LAST).customer_name := r_customer.name;
        t_customers(t_customers.LAST).credit_limit  := r_customer.credit_limit;
    END LOOP;

    -- show all customers
    FOR l_index IN t_customers .FIRST..t_customers.LAST
    LOOP
        dbms_output.put_line(
            'The customer ' ||
            t_customers(l_index).customer_name ||
            ' has a credit of ' ||
            t_customers(l_index).credit_limit
        );
    END LOOP;

END;
/
```

Output:

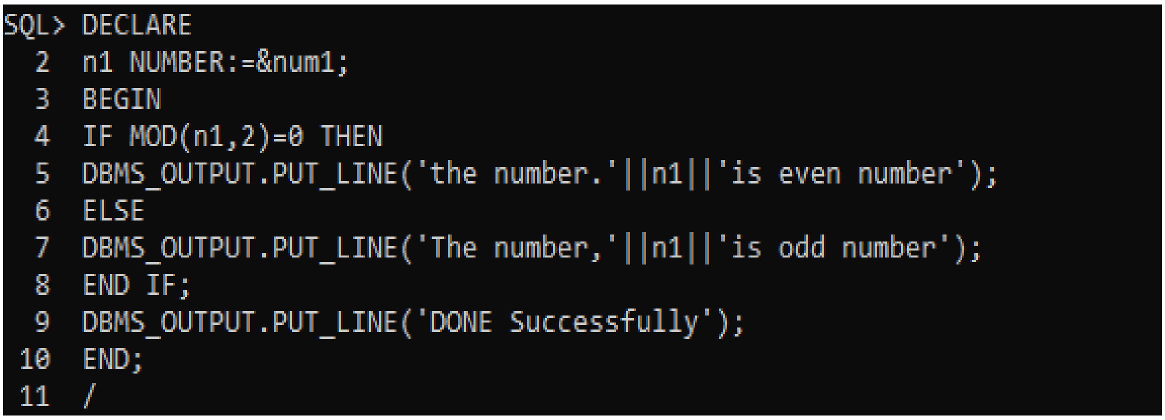


```
The customer Chetan has a credit of 150000
The customer Arvind has a credit of 120000
The customer Vikas has a credit of 100000
The customer Pritesh has a credit of 95000
The customer Sahil has a credit of 75000

PL/SQL procedure successfully completed.
```

Practical No.4

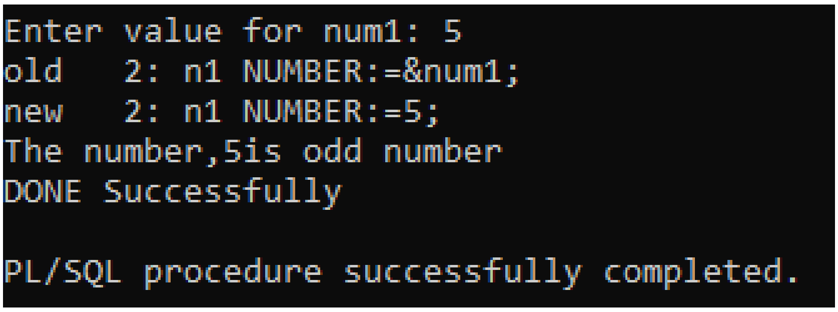
Conditional and looping statements

Code: conditional statements



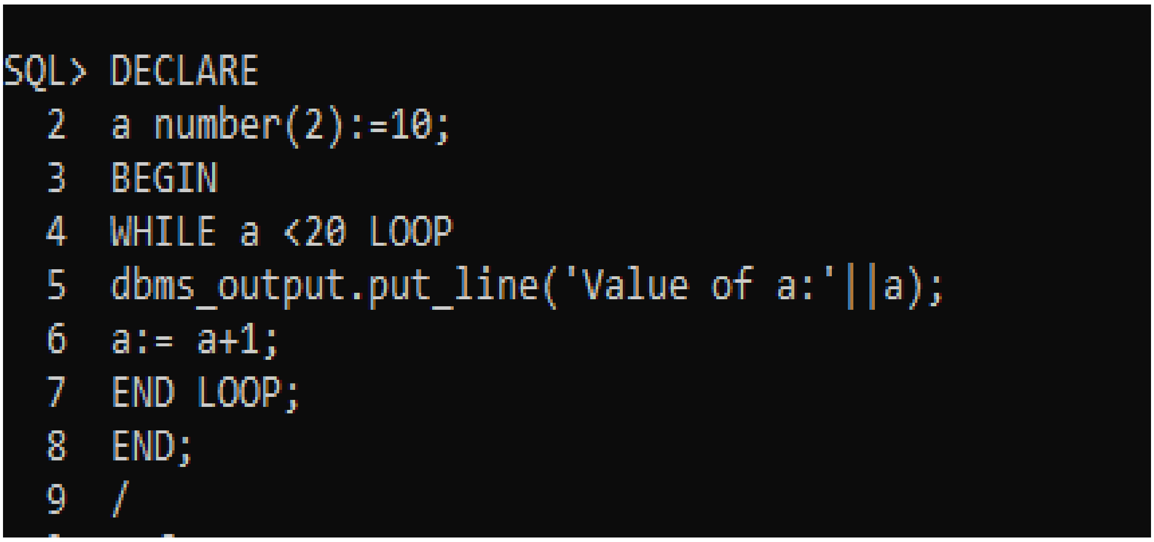
```
SQL> DECLARE
n1 NUMBER:=&num1;
BEGIN
IF MOD(n1,2)=0 THEN
DBMS_OUTPUT.PUT_LINE('the number.'||n1||'is even number');
ELSE
DBMS_OUTPUT.PUT_LINE('The number,'||n1||'is odd number');
END IF;
DBMS_OUTPUT.PUT_LINE('DONE Successfully');
END;
/
```

output:



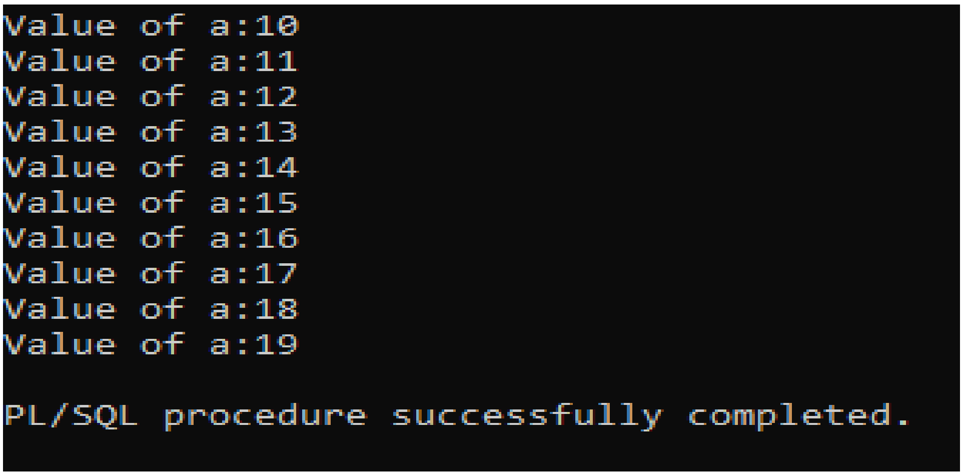
```
Enter value for num1: 5
old   2: n1 NUMBER:=&num1;
new   2: n1 NUMBER:=5;
The number,5is odd number
DONE Successfully

PL/SQL procedure successfully completed.
```

code: while loop



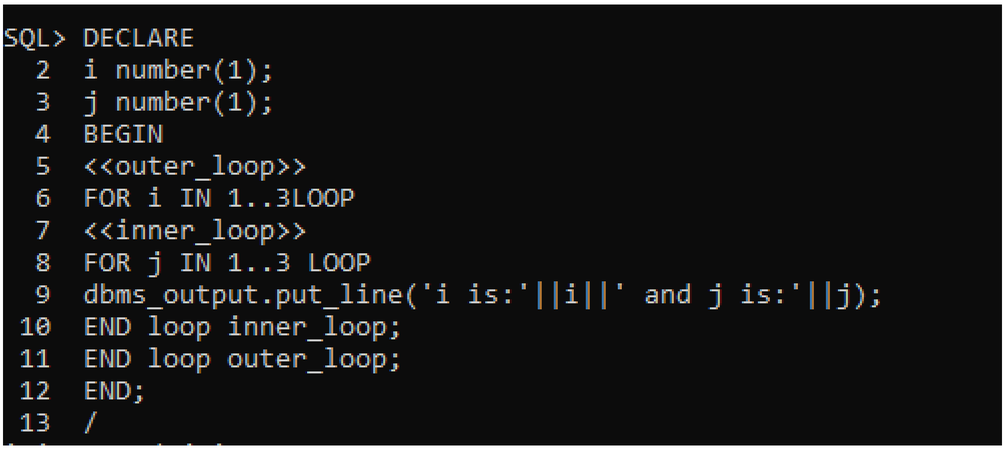
```
SQL> DECLARE
a number(2):=10;
BEGIN
WHILE a <20 LOOP
dbms_output.put_line('Value of a:'||a);
a:= a+1;
END LOOP;
END;
/
```

Output:



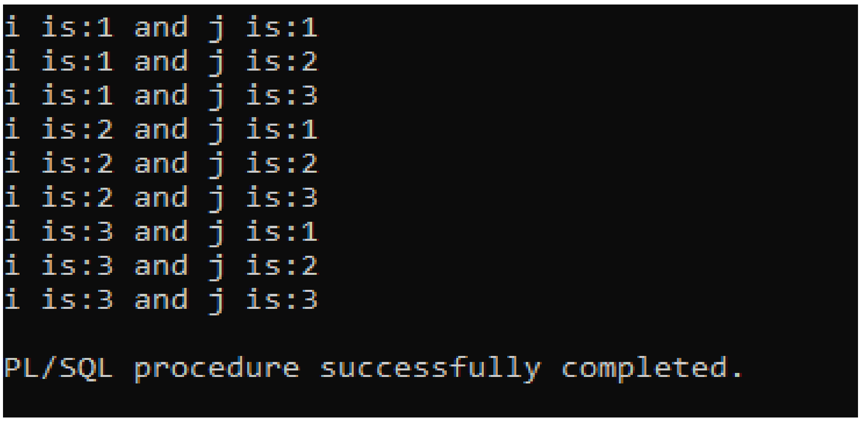
```
Value of a:10
Value of a:11
Value of a:12
Value of a:13
Value of a:14
Value of a:15
Value of a:16
Value of a:17
Value of a:18
Value of a:19

PL/SQL procedure successfully completed.
```

Code: for loop



```
SQL> DECLARE
     i number(1);
     j number(1);
     BEGIN
     <<outer_loop>>
     FOR i IN 1..3LOOP
     <<inner_loop>>
     FOR j IN 1..3 LOOP
     dbms_output.put_line('i is:'||i||' and j is:'||j);
     END loop inner_loop;
     END loop outer_loop;
     END;
     /
```

Output:

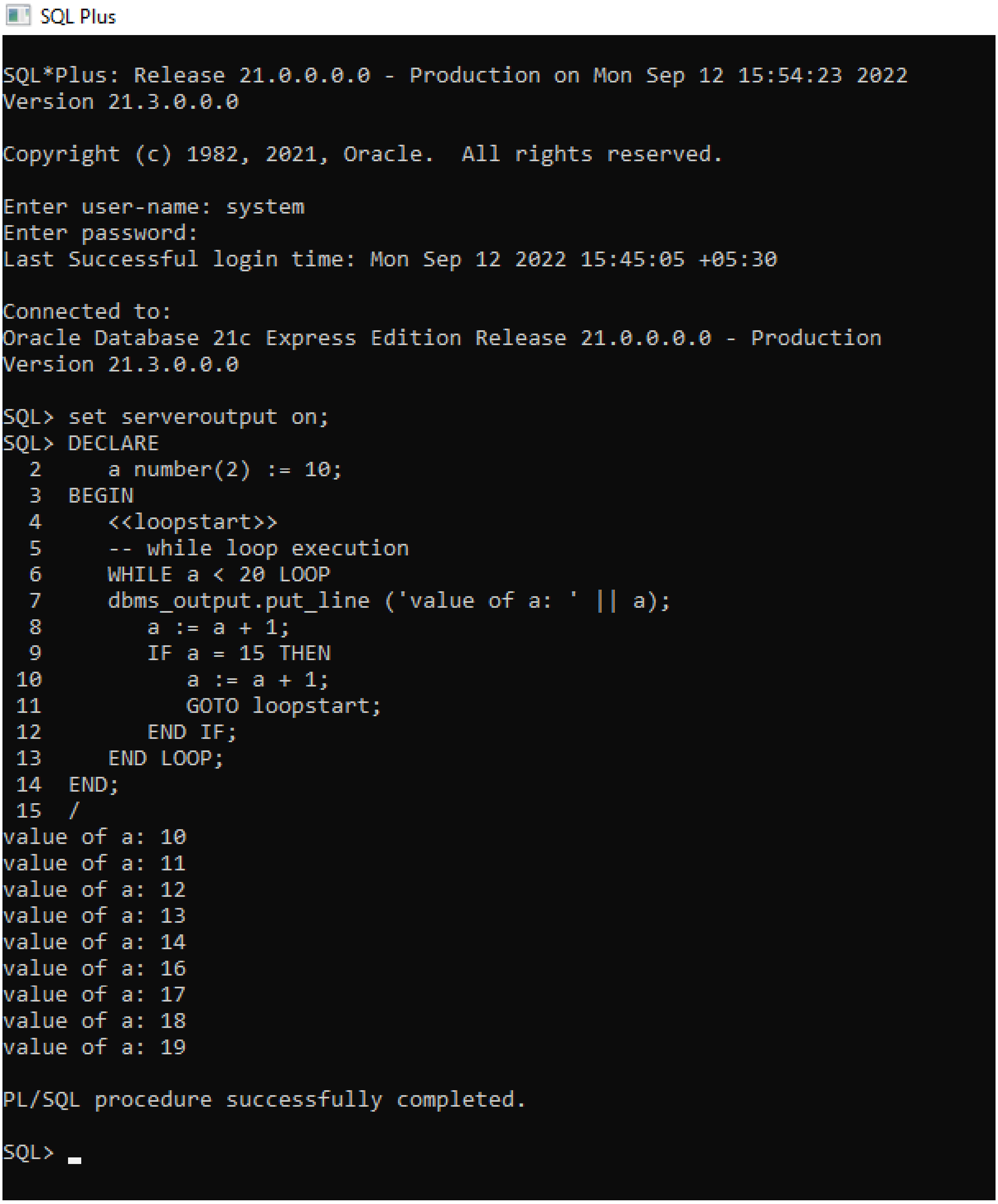


```
i is:1 and j is:1
i is:1 and j is:2
i is:1 and j is:3
i is:2 and j is:1
i is:2 and j is:2
i is:2 and j is:3
i is:3 and j is:1
i is:3 and j is:2
i is:3 and j is:3

PL/SQL procedure successfully completed.
```

Practical no. 5

Plsql block with basic programming construct: GOTO

Code:

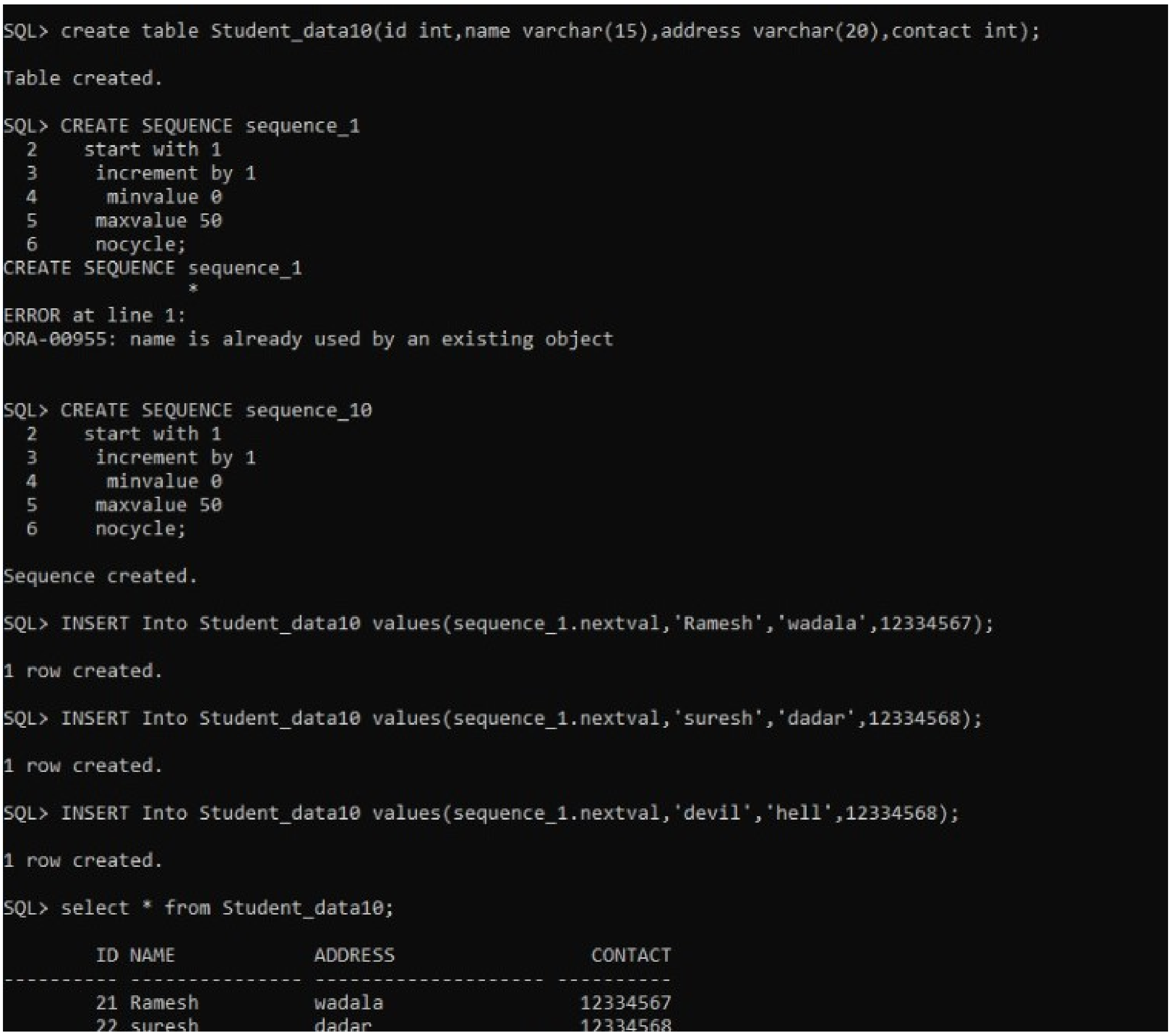


```
SQL*Plus: Release 21.0.0.0.0 - Production on Mon Sep 12 15:54:23 2022
Version 21.3.0.0.0

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

Enter user-name: system
Enter password:
Last Successful login time: Mon Sep 12 2022 15:45:05 +05:30

Connected to:
Oracle Database 21c Express Edition Release 21.0.0.0.0 - Production
Version 21.3.0.0.0

SQL> set serveroutput on;
SQL> DECLARE
        a number(2) := 10;
     BEGIN
        <<loopstart>>
        -- while loop execution
        WHILE a < 20 LOOP
        dbms_output.put_line ('value of a: ' || a);
           a := a + 1;
           IF a = 15 THEN
              a := a + 1;
              GOTO loopstart;
           END IF;
        END LOOP;
     END;
     /
value of a: 10
value of a: 11
value of a: 12
value of a: 13
value of a: 14
value of a: 16
value of a: 17
value of a: 18
value of a: 19

PL/SQL procedure successfully completed.

SQL>
```

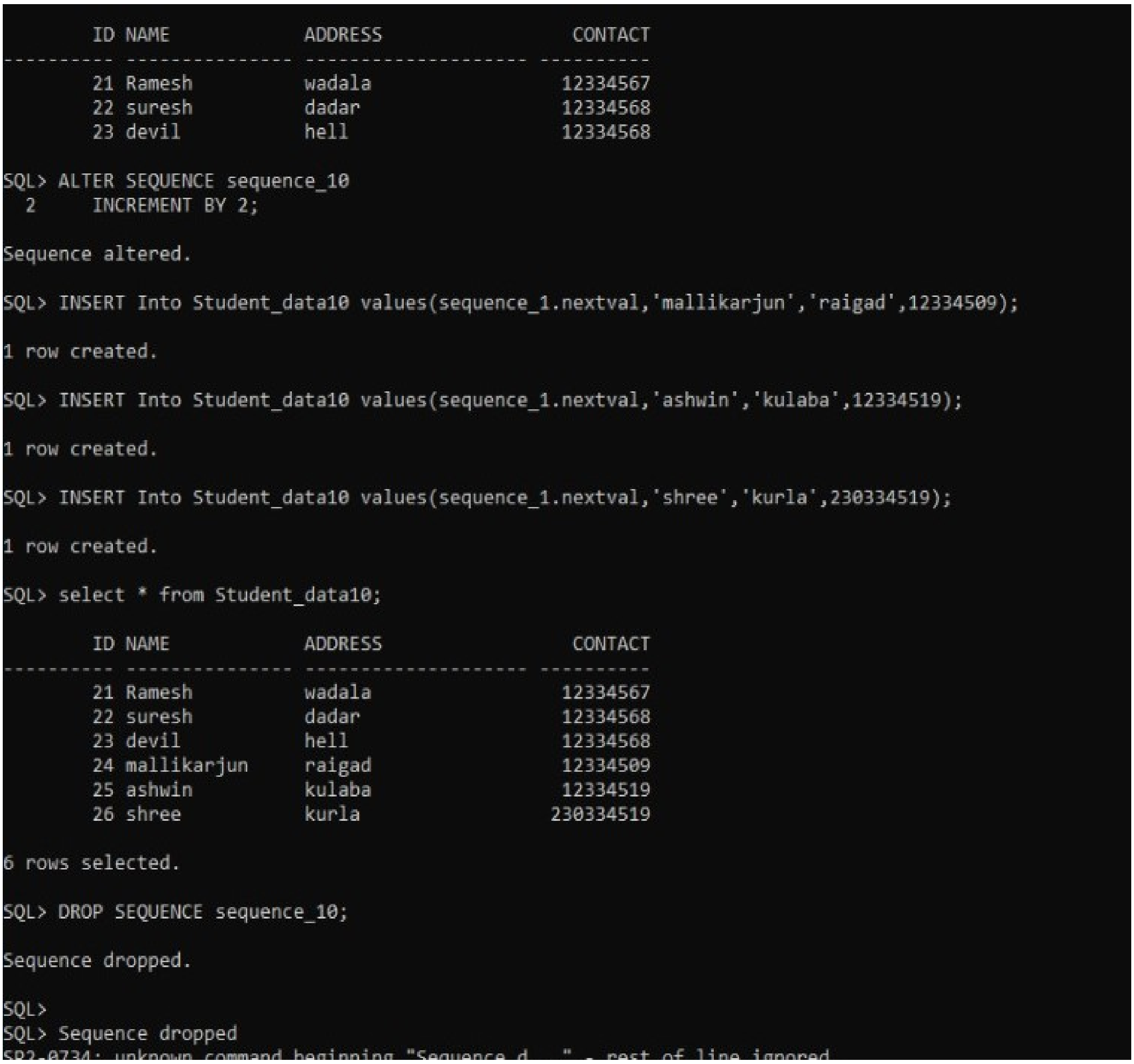
Practical no. 6

SEQUENCE

Code:



```
SQL> create table Student_data10(id int,name varchar(15),address varchar(20),contact int);

Table created.

SQL> CREATE SEQUENCE sequence_1
  2    start with 1
  3     increment by 1
  4      minvalue 0
  5     maxvalue 50
  6     nocycle;
CREATE SEQUENCE sequence_1
                *
ERROR at line 1:
ORA-00955: name is already used by an existing object


SQL> CREATE SEQUENCE sequence_10
  2    start with 1
  3     increment by 1
  4      minvalue 0
  5     maxvalue 50
  6     nocycle;

Sequence created.

SQL> INSERT Into Student_data10 values(sequence_1.nextval,'Ramesh','wadala',12334567);

1 row created.

SQL> INSERT Into Student_data10 values(sequence_1.nextval,'suresh','dadar',12334568);

1 row created.

SQL> INSERT Into Student_data10 values(sequence_1.nextval,'devil','hell',12334568);

1 row created.

SQL> select * from Student_data10;

        ID NAME            ADDRESS                 CONTACT
---------- --------------- -------------------- ----------
        21 Ramesh          wadala                 12334567
        22 suresh          dadar                  12334568
```

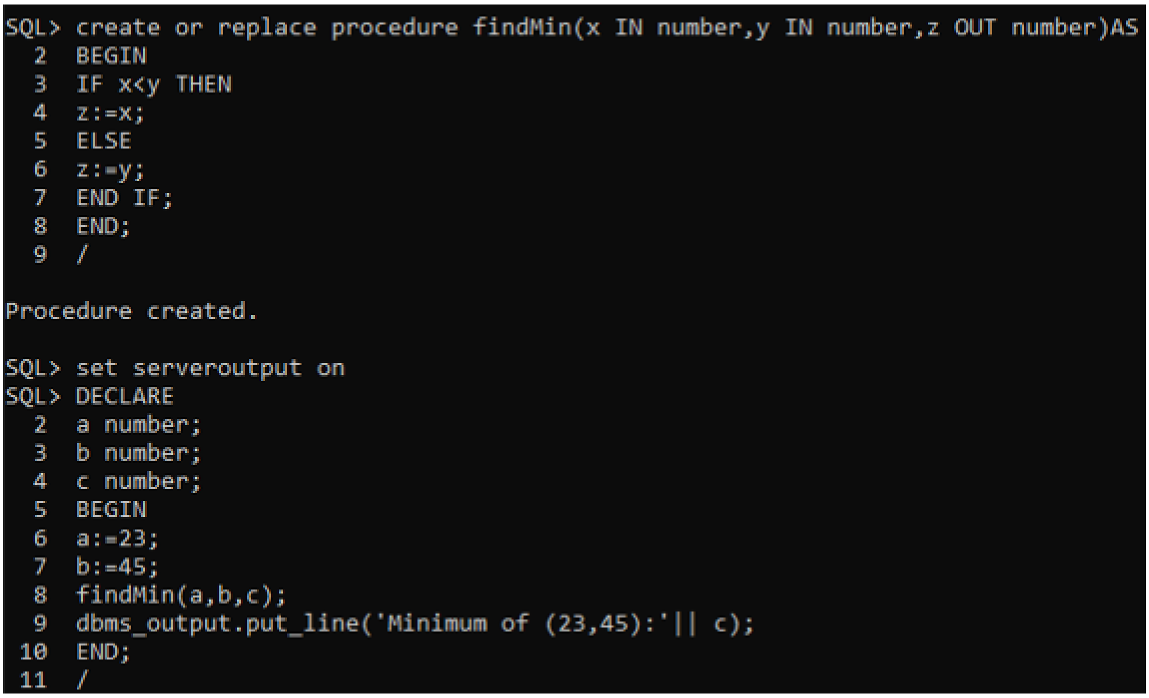


```
        ID NAME           ADDRESS                 CONTACT
---------- -------------- -------------------- ----------
        21 Ramesh         wadala                 12334567
        22 suresh         dadar                  12334568
        23 devil          hell                   12334568

SQL> ALTER SEQUENCE sequence_10
  2     INCREMENT BY 2;

Sequence altered.

SQL> INSERT Into Student_data10 values(sequence_1.nextval,'mallikarjun','raigad',12334509);

1 row created.

SQL> INSERT Into Student_data10 values(sequence_1.nextval,'ashwin','kulaba',12334519);

1 row created.

SQL> INSERT Into Student_data10 values(sequence_1.nextval,'shree','kurla',230334519);

1 row created.

SQL> select * from Student_data10;

        ID NAME           ADDRESS                 CONTACT
---------- -------------- -------------------- ----------
        21 Ramesh         wadala                 12334567
        22 suresh         dadar                  12334568
        23 devil          hell                   12334568
        24 mallikarjun    raigad                 12334509
        25 ashwin         kulaba                 12334519
        26 shree          kurla                 230334519

6 rows selected.

SQL> DROP SEQUENCE sequence_10;

Sequence dropped.

SQL>
SQL> Sequence dropped
SP2-0734: unknown command beginning "Sequence d..." - rest of line ignored.
```

Practical No. 7

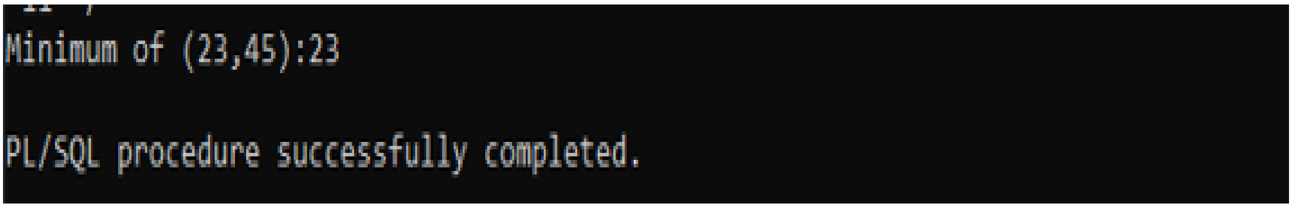
PROCEDURES and FUNCTIONS

Code: procedure



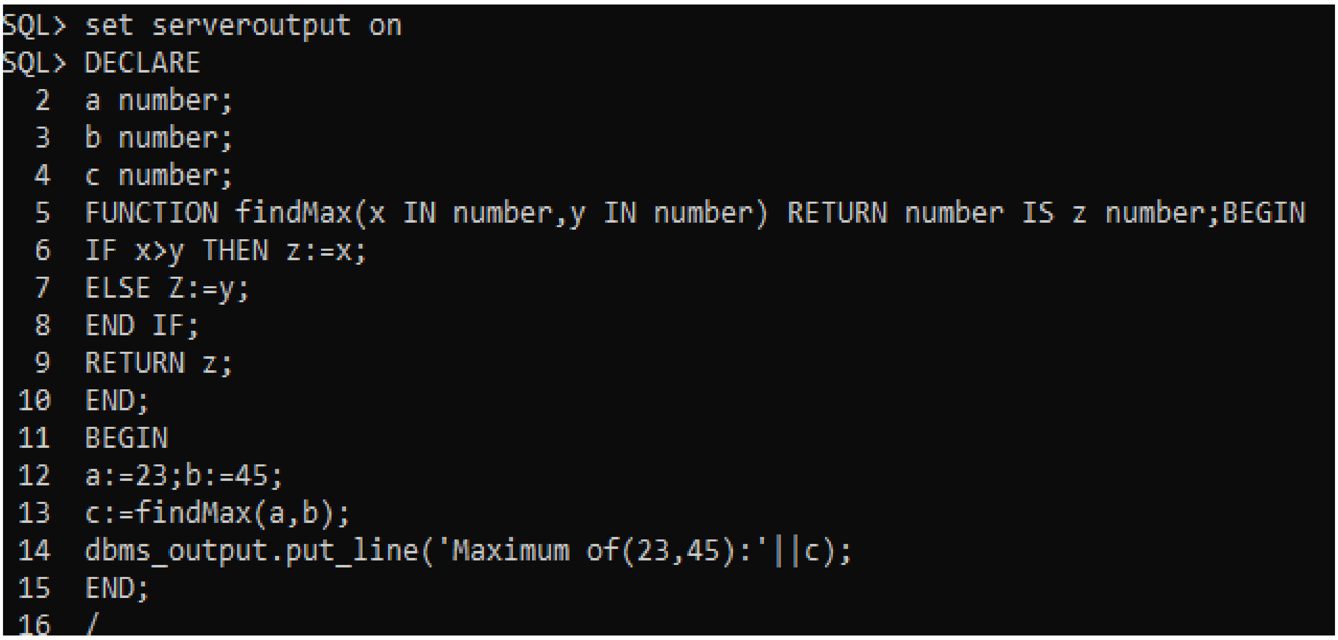
```
SQL> create or replace procedure findMin(x IN number,y IN number,z OUT number)AS
     BEGIN
     IF x<y THEN
     z:=x;
     ELSE
     z:=y;
     END IF;
     END;
     /

Procedure created.

SQL> set serveroutput on
SQL> DECLARE
     a number;
     b number;
     c number;
     BEGIN
     a:=23;
     b:=45;
     findMin(a,b,c);
     dbms_output.put_line('Minimum of (23,45):'|| c);
     END;
     /
```

Output:



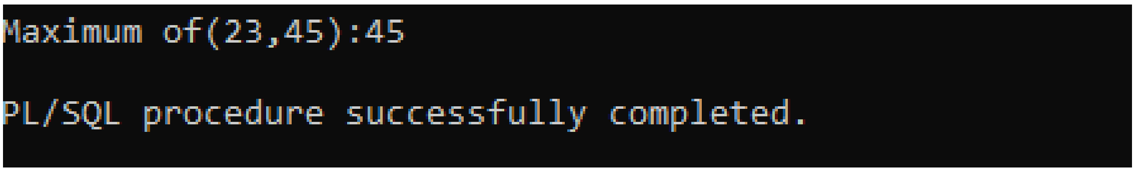
```
Minimum of (23,45):23

PL/SQL procedure successfully completed.
```

Code: Function



```
SQL> set serveroutput on
SQL> DECLARE
a number;
b number;
c number;
FUNCTION findMax(x IN number,y IN number) RETURN number IS z number;BEGIN
IF x>y THEN z:=x;
ELSE Z:=y;
END IF;
RETURN z;
END;
BEGIN
a:=23;b:=45;
c:=findMax(a,b);
dbms_output.put_line('Maximum of(23,45):'||c);
END;
/
```

Output:

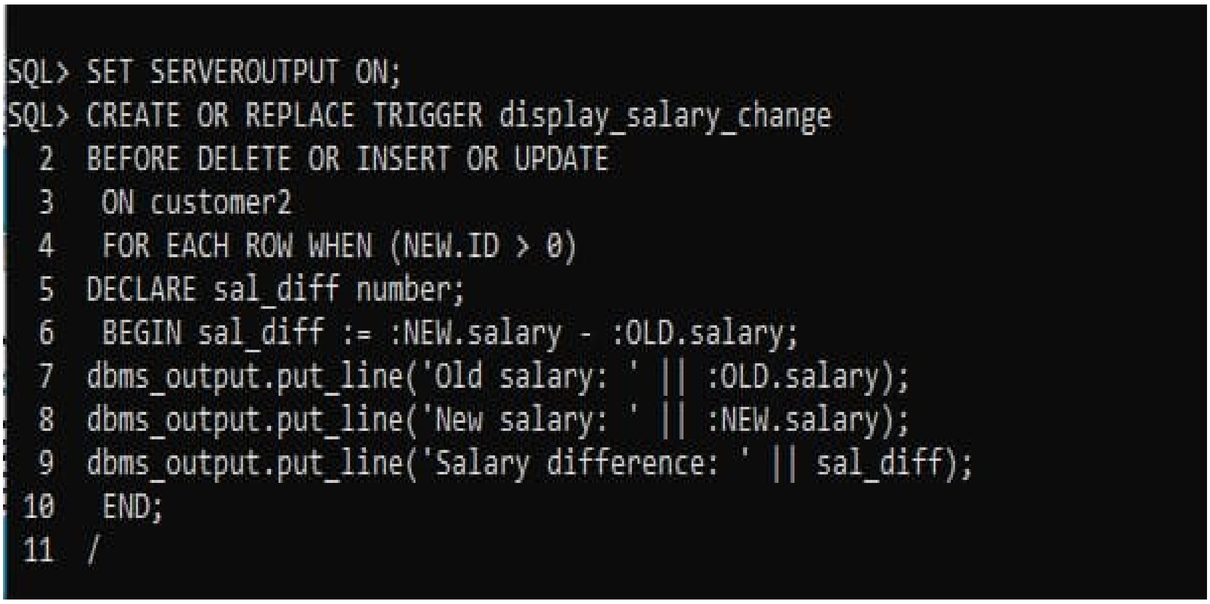


```
Maximum of(23,45):45

PL/SQL procedure successfully completed.
```

Practical no. 8

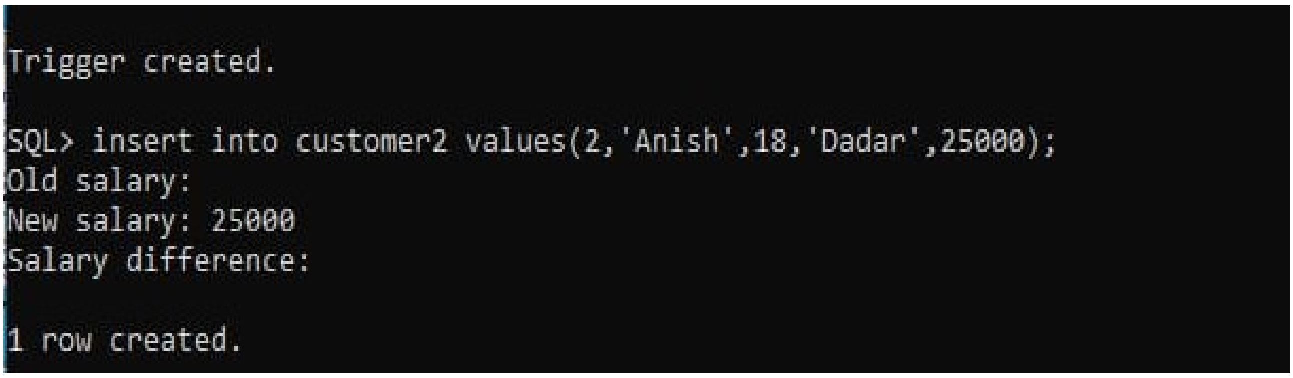
TRIGGERS

Code:



```
SQL> SET SERVEROUTPUT ON;
SQL> CREATE OR REPLACE TRIGGER display_salary_change
     BEFORE DELETE OR INSERT OR UPDATE
      ON customer2
      FOR EACH ROW WHEN (NEW.ID > 0)
     DECLARE sal_diff number;
      BEGIN sal_diff := :NEW.salary - :OLD.salary;
     dbms_output.put_line('Old salary: ' || :OLD.salary);
     dbms_output.put_line('New salary: ' || :NEW.salary);
     dbms_output.put_line('Salary difference: ' || sal_diff);
      END;
     /
```

Output:

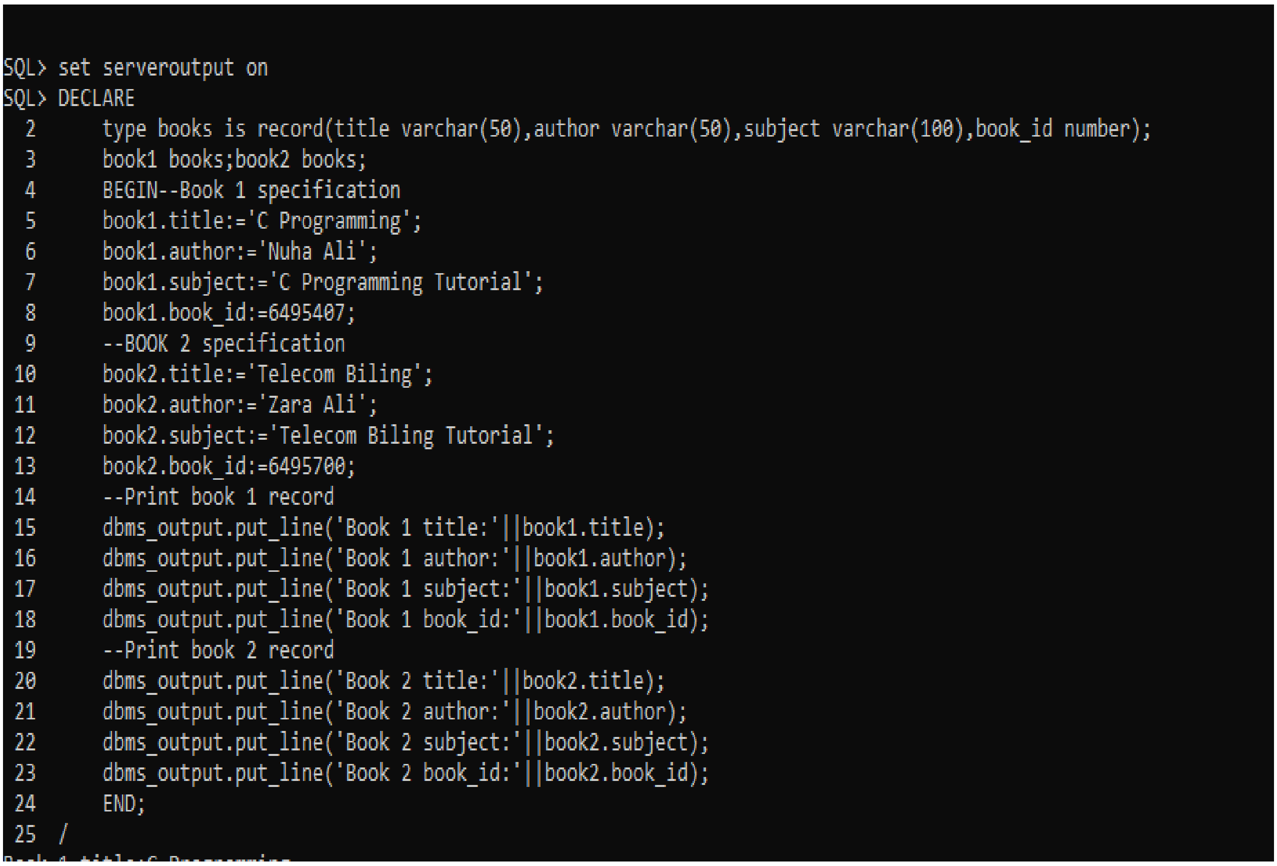


```
Trigger created.

SQL> insert into customer2 values(2,'Anish',18,'Dadar',25000);
Old salary:
New salary: 25000
Salary difference:

1 row created.
```

Practical No: 9

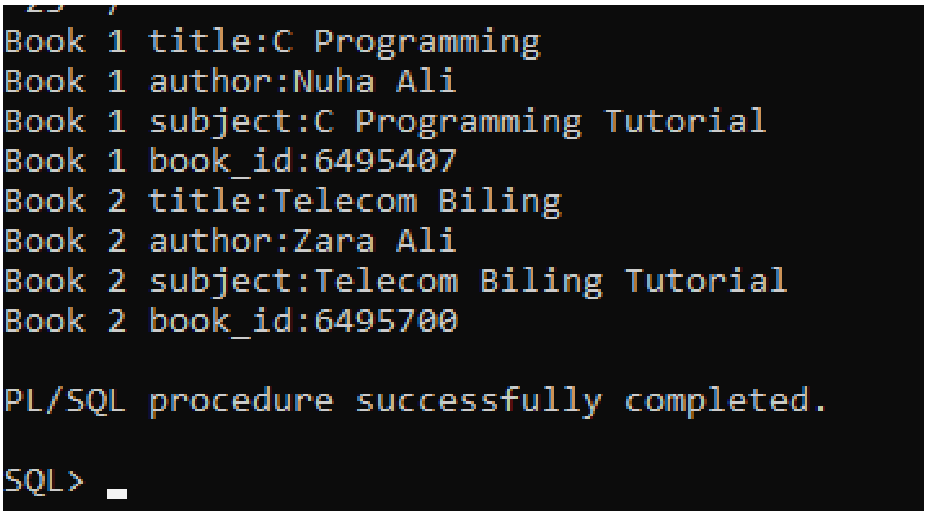
CURSOR

Code:



```
SQL> set serveroutput on
SQL> DECLARE
     type books is record(title varchar(50),author varchar(50),subject varchar(100),book_id number);
     book1 books;book2 books;
     BEGIN--Book 1 specification
     book1.title:='C Programming';
     book1.author:='Nuha Ali';
     book1.subject:='C Programming Tutorial';
     book1.book_id:=6495407;
     --BOOK 2 specification
     book2.title:='Telecom Biling';
     book2.author:='Zara Ali';
     book2.subject:='Telecom Biling Tutorial';
     book2.book_id:=6495700;
     --Print book 1 record
     dbms_output.put_line('Book 1 title:'||book1.title);
     dbms_output.put_line('Book 1 author:'||book1.author);
     dbms_output.put_line('Book 1 subject:'||book1.subject);
     dbms_output.put_line('Book 1 book_id:'||book1.book_id);
     --Print book 2 record
     dbms_output.put_line('Book 2 title:'||book2.title);
     dbms_output.put_line('Book 2 author:'||book2.author);
     dbms_output.put_line('Book 2 subject:'||book2.subject);
     dbms_output.put_line('Book 2 book_id:'||book2.book_id);
     END;
 /
Book 1 title:C Programming
```

Output:

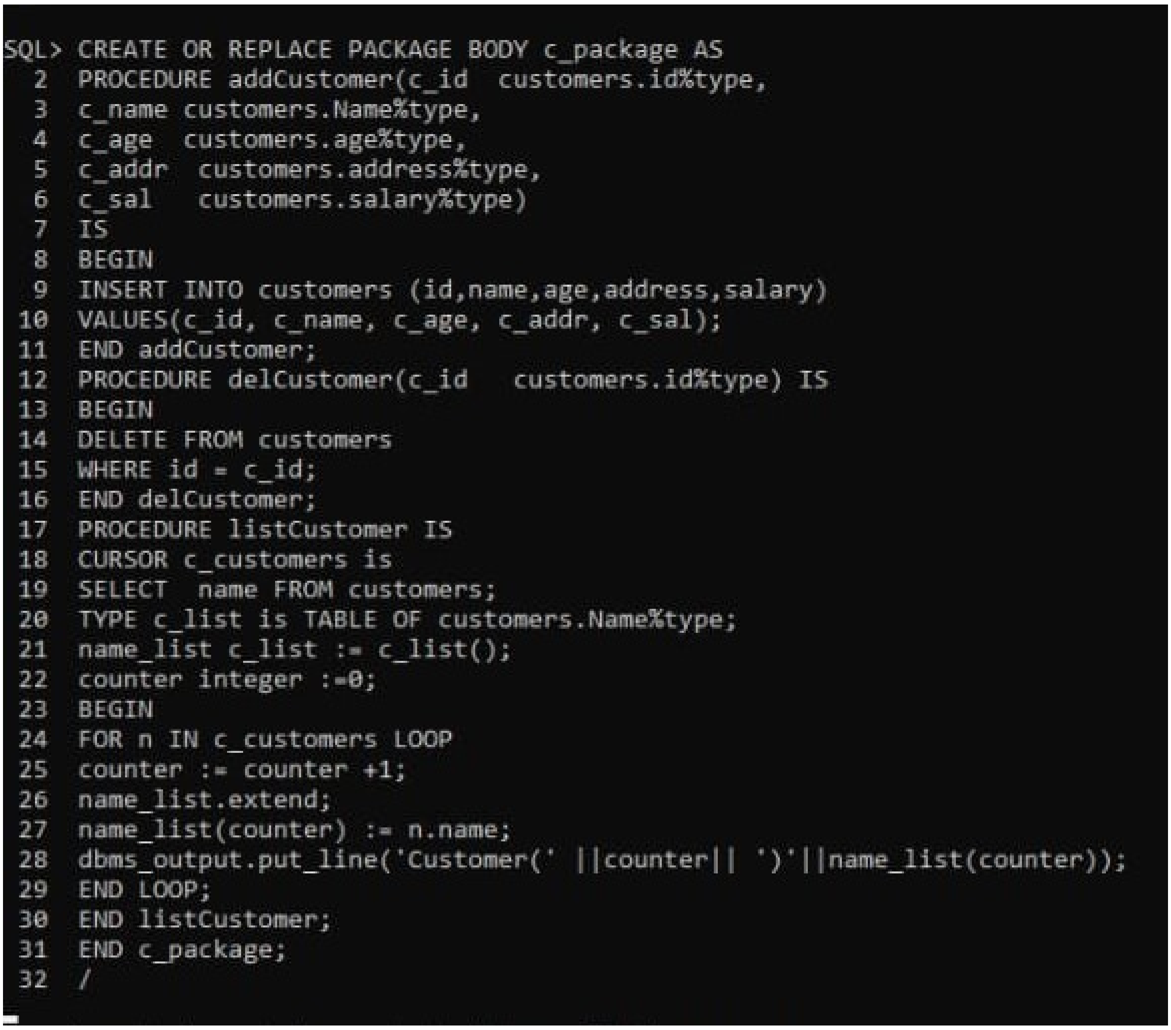


```
Book 1 title:C Programming
Book 1 author:Nuha Ali
Book 1 subject:C Programming Tutorial
Book 1 book_id:6495407
Book 2 title:Telecom Biling
Book 2 author:Zara Ali
Book 2 subject:Telecom Biling Tutorial
Book 2 book_id:6495700

PL/SQL procedure successfully completed.

SQL>
```

Practical no.10

PACKAGES

Code:

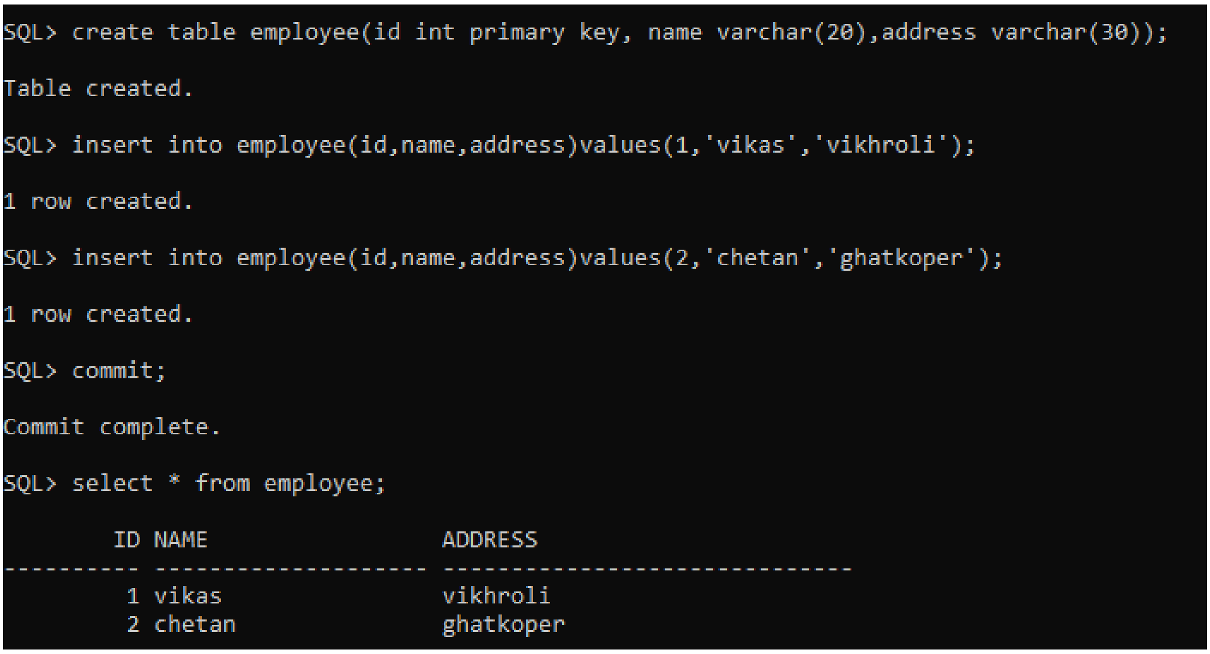


```
SQL> CREATE OR REPLACE PACKAGE BODY c_package AS
PROCEDURE addCustomer(c_id  customers.id%type,
c_name customers.Name%type,
c_age  customers.age%type,
c_addr  customers.address%type,
c_sal   customers.salary%type)
IS
BEGIN
INSERT INTO customers (id,name,age,address,salary)
VALUES(c_id, c_name, c_age, c_addr, c_sal);
END addCustomer;
PROCEDURE delCustomer(c_id   customers.id%type) IS
BEGIN
DELETE FROM customers
WHERE id = c_id;
END delCustomer;
PROCEDURE listCustomer IS
CURSOR c_customers is
SELECT  name FROM customers;
TYPE c_list is TABLE OF customers.Name%type;
name_list c_list := c_list();
counter integer :=0;
BEGIN
FOR n IN c_customers LOOP
counter := counter +1;
name_list.extend;
name_list(counter) := n.name;
dbms_output.put_line('Customer(' ||counter|| ')'||name_list(counter));
END LOOP;
END listCustomer;
END c_package;
/
```

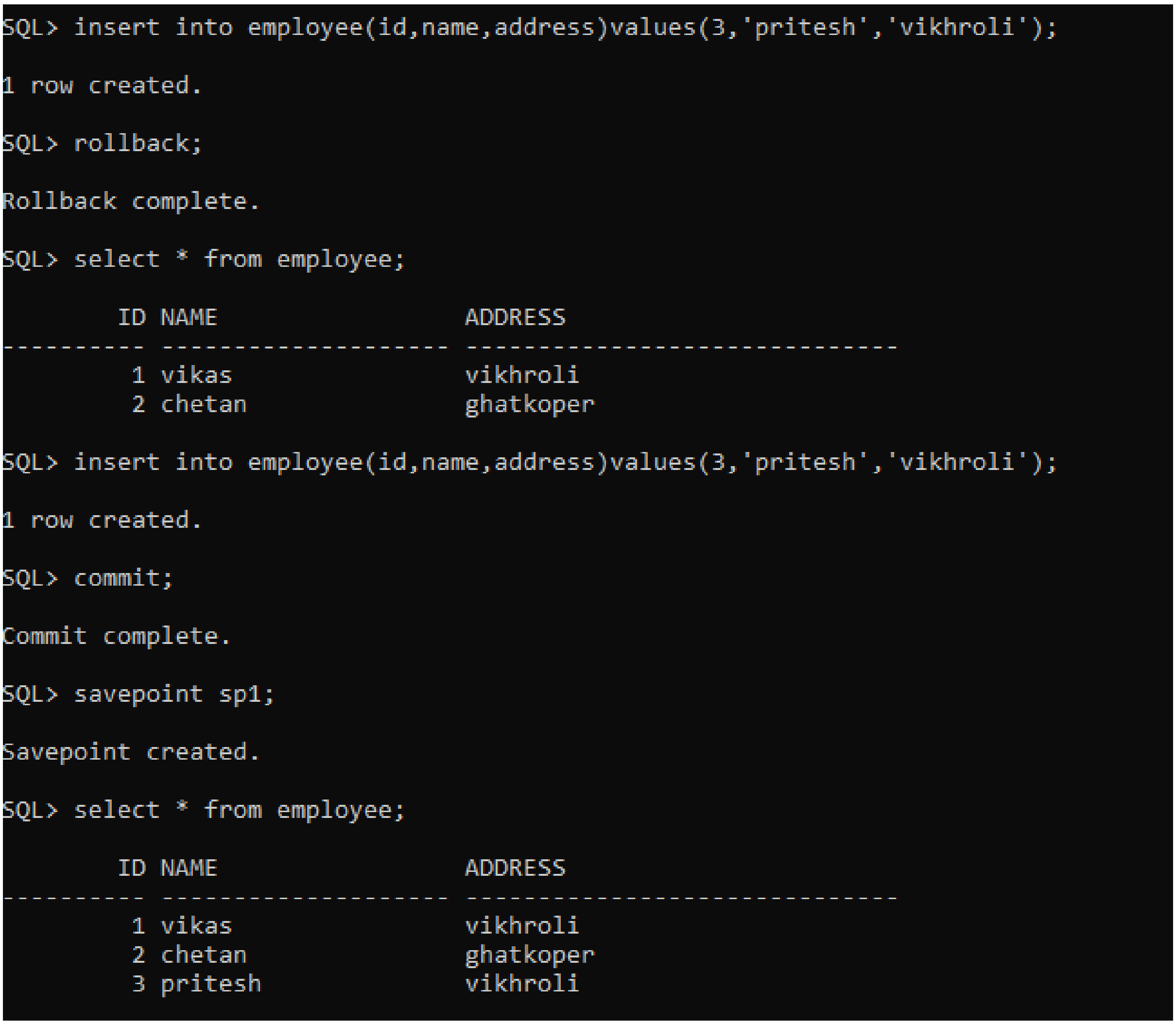
Practical No. 11

TRANSACTION MANAGEMENT

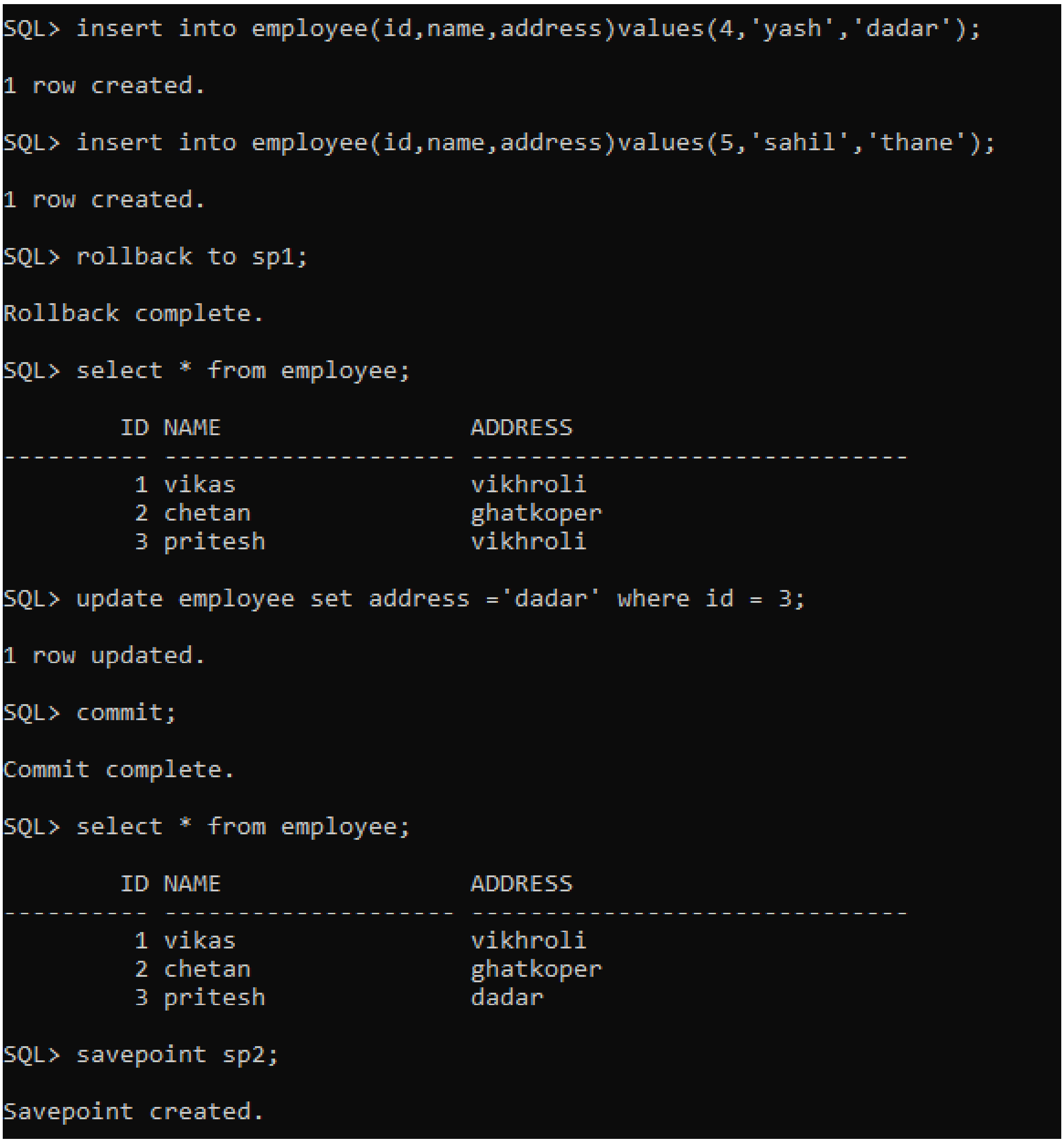
Code



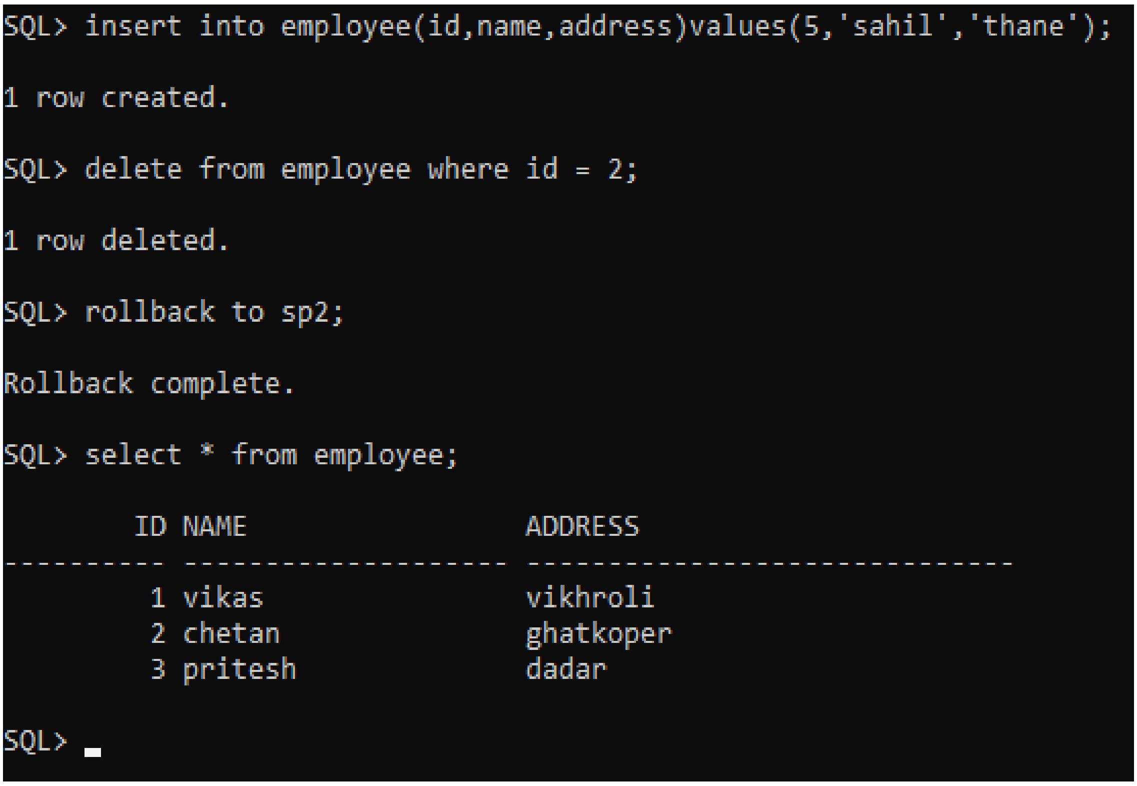
```
SQL> create table employee(id int primary key, name varchar(20),address varchar(30));

Table created.

SQL> insert into employee(id,name,address)values(1,'vikas','vikhroli');

1 row created.

SQL> insert into employee(id,name,address)values(2,'chetan','ghatkoper');

1 row created.

SQL> commit;

Commit complete.

SQL> select * from employee;

        ID NAME                 ADDRESS
---------- -------------------- ------------------------------
         1 vikas                vikhroli
         2 chetan               ghatkoper
```



```
SQL> insert into employee(id,name,address)values(3,'pritesh','vikhroli');

1 row created.

SQL> rollback;

Rollback complete.

SQL> select * from employee;

        ID NAME                 ADDRESS
---------- -------------------- ------------------------------
         1 vikas                vikhroli
         2 chetan               ghatkoper

SQL> insert into employee(id,name,address)values(3,'pritesh','vikhroli');

1 row created.

SQL> commit;

Commit complete.

SQL> savepoint sp1;

Savepoint created.

SQL> select * from employee;

        ID NAME                 ADDRESS
---------- -------------------- ------------------------------
         1 vikas                vikhroli
         2 chetan               ghatkoper
         3 pritesh              vikhroli
```



```
SQL> insert into employee(id,name,address)values(4,'yash','dadar');

1 row created.

SQL> insert into employee(id,name,address)values(5,'sahil','thane');

1 row created.

SQL> rollback to sp1;

Rollback complete.

SQL> select * from employee;

        ID NAME                 ADDRESS
---------- -------------------- ------------------------------
         1 vikas                vikhroli
         2 chetan               ghatkoper
         3 pritesh              vikhroli

SQL> update employee set address ='dadar' where id = 3;

1 row updated.

SQL> commit;

Commit complete.

SQL> select * from employee;

        ID NAME                 ADDRESS
---------- -------------------- ------------------------------
         1 vikas                vikhroli
         2 chetan               ghatkoper
         3 pritesh              dadar

SQL> savepoint sp2;

Savepoint created.
```



```
SQL> insert into employee(id,name,address)values(5,'sahil','thane');

1 row created.

SQL> delete from employee where id = 2;

1 row deleted.

SQL> rollback to sp2;

Rollback complete.

SQL> select * from employee;

        ID NAME                 ADDRESS
---------- -------------------- ------------------------------
         1 vikas                vikhroli
         2 chetan               ghatkoper
         3 pritesh              dadar

SQL> _
```