TRIGGER

1. Create a Simple Trigger that does not allow Insert,Update and Delete Operations on the Table.

```
SQL> select * from item;


ITEM_ID ITEMNAME                 PRICE

---------- -------------------- ----------
      1234 Geera                 204.5
      1235 Colgate               200
      1236 lays                   50
      1237 Buiscuit              100


SQL> create trigger mytrig
  2  BEFORE INSERT OR UPDATE OR DELETE ON item FOR EACH ROW
  3  begin
  4  raise_application_error(-20010,'you are not permitted to do this operation');
  5  end;
  6  /


Trigger created.


SQL> insert into item values(1239,'Laptop',25000);

insert into item values(1239,'Laptop',25000)
            *
ERROR at line 1:
ORA-20010: you are not permitted to do this operation
ORA-06512: at "SYSTEM.MYTRIG", line 2
ORA-04088: error during execution of trigger 'SYSTEM.MYTRIG'


SQL>delete from item where itemname='lays';
```

```
delete from item where itemname='lays'
                    *
         ERROR at line 1:
         ORA-20010: you are not permitted to do this operation
         ORA-06512: at "SYSTEM.MYTRIG", line 2
        ORA-04088: error during execution of trigger 'SYSTEM.MYTRIG'
SQL>update item set price=100 where item_id=1237;
update item set price=100 where item_id=1237
                *
         ERROR at line 1:
         ORA-20010: you are not permitted to do this operation
         ORA-06512: at "SYSTEM.MYTRIG", line 2
        ORA-04088: error during execution of trigger 'SYSTEM.MYTRIG'
```

2. Create a trigger that displays a message after update, Delete, Insert operations on a table.

```
SQL> select * from emp1;

      ID NAME                 SALARY
---------- -------------------- ----------
       123 Amith               15000
       124 Amitha              18000
       125 Amritha             25000
       126 Amal                35000

SQL> create or replace trigger mytrig2
  2  after update or insert or delete on emp1
  3  for each row
  4  begin
  5  if updating then
  6  dbms_output.put_line('updated');
  7  elsif inserting then
  8  dbms_output.put_line('insertion done');
  9  elsif deleting then
 10  dbms_output.put_line('deleted');
 11  end if;
 12  end;
 13  /

Trigger created.


SQL> insert into emp1 values(127,'Ankitha',28000);
insertion done


1 row created.


SQL> delete from emp1 where id=123;
deleted


1 row deleted.
SQL> update emp1 set salary=23000 where id=2;
updated
```

3. Create a trigger that gets invoked before insert operation on a table. The trigger should convert employee name to uppercase before its stored in the table.

```
SQL> select * from emp41;

        ID NAME                 ADDRESS
---------- -------------------- ----------
         1 Sreelakshmi          Kottayam
         2 Sreehari             Palakkadu
         3 Vijay                Alappuzha



SQL> create or replace trigger trig1
  2  before insert on emp41
  3  for each row
  4  begin
  5  :new.name:=upper(:new.name);
  6  end;
  7  /

Trigger created.
SQL>insert into emp41 values(4,'surya','Idukki');
1 row created.

SQL> select * from emp41;

        ID NAME                 ADDRESS
```

```
---------- ------------------- ----------
       1 Sreelakshmi              Kottayam

       2 Sreehari                 Palakkadu

       3 Vijay                       Alappuzha

       4 SURYA                     Idukki
```

4. Create a row-level trigger for the customers table that would fire for UPDATE operations performed on the CUSTOMERS table. This trigger should display the salary difference between the old values and new values

```
SQL> SELECT * FROM CUSTOMERS41;

      ID NAME                    SALARY

--------------------------------------------------------------------------------
       1 Sreelakshmi                 21000


       2 Sreehari                 35000


       3 Vijay                 39000


       4 Surya                  56000




SQL> CREATE OR REPLACE TRIGGER TRG5
```

```
  2   BEFORE UPDATE ON CUSTOMERS41 FOR EACH ROW
  3   WHEN(NEW.ID > 0)
  4   DECLARE
  5   SAL_DIFFERENCE NUMBER;
  6   BEGIN
  7   SAL_DIFFERENCE:=:NEW.SALARY-:OLD.SALARY;
  8   DBMS_OUTPUT.PUT_LINE('Old Salary:'||:OLD.SALARY);
  9   DBMS_OUTPUT.PUT_LINE('NEW Salary:'||:NEW.SALARY);
 10   DBMS_OUTPUT.PUT_LINE('Salary Difference:'||SAL_DIFFERENCE);
 11   END;
 12   /

Trigger created.



SQL> UPDATE CUSTOMERS41 SET SALARY=67000 WHERE ID=1;
Old Salary: 21000
NEW Salary : 67000
Salary Difference: 46000
1 row updated.

SQL> SELECT * FROM CUSTOMERS41;

        ID NAME                    SALARY
------------------------------------------------------------------------------
         1 Sreelakshmi                67000
```

2	Sreehari	35000
3	Vijay	39000
4	Surya	56000