Ex.No.: 13		WORKING WITH TRIGGER TRIGGER
Date:		

DEFINITION

A trigger is a statement that is executed automatically by the system as a side effect of a modification to the database. The parts of a trigger are,

- **Trigger statement**: Specifies the DML statements and fires the trigger body. It also specifies the table to which the trigger is associated.
- **Trigger body or trigger action**: It is a PL/SQL block that is executed when the triggering statement is used.
- **Trigger restriction**: Restrictions on the trigger can be achieved

The different uses of triggers are as follows,

- *To generate data automatically*
- *To enforce complex integrity constraints*
- *To customize complex securing authorizations*
- *To maintain the replicate table*
- To audit data modifications

TYPES OF TRIGGERS

The various types of triggers are as follows,

- **Before**: It fires the trigger before executing the trigger statement.
- **After**: It fires the trigger after executing the trigger statement
- .
- **For each row**: It specifies that the trigger fires once per row
- .
- **For each statement**: This is the default trigger that is invoked. It specifies that the trigger fires once per statement.

VARIABLES USED IN TRIGGERS

- :new
- :old

Program 1

Write a code in PL/SQL to develop a trigger that enforces referential integrity by preventing the deletion of a parent record if child records exist.

```
create or replace TRIGGER P_del_par Before DELETE on dept
  for each row
Declare
        C_count number ;

BEGIN
    Select Count (*) into C_count from emp where dept_id
= : old dept_id
    If count >0 Then
        raise_application error (-20001, 'cannot delete dept') ;
  END IF ;
```

Program 2

Write a code in PL/SQL to create a trigger that checks for duplicate values in a specific column and raises an exception if found.

```
Create or Replace TRIGGER check_dup_name Before INSERT
or update on users
for Each ROW
DECLARE
        v_count Number ;
BEGIN
        Select Count (*) into v_count from users where
    username = new_username ;
If v_count >0 Then
        Raise_application_error (-20001, duplicate username found
```

Program 3

Write a code in PL/SQL to create a trigger that restricts the insertion of new rows if the total of a column's values exceeds a certain threshold.

```
create or replace trigger c_tot_amt Before Insert on
sales for each row
Declare tot_amt number;
        threshold constant number := 10000;
Begin Select SUM(amnOt) INTO tot_amt from Sales
If total_amount + amount > threshold then
```

Program 4

Write a code in PL/SQL to design a trigger that captures changes made to specific columns and logs them in an audit table.

```
create or replace TRIGGER trg_emp_employees
After update of sal, dep_id on emp
for each row
BEGIN
  Insert into emp_id(emp_id, changed id, old values,
  newValues (select :OLD emp_id; 'salary', To CHAR
  (:OLD.salary) To char (NEW :Salary)
Select :OLD_emp_id, 'dept_id', to char(:OLD.deptid
```

Program 5

Write a code in PL/SQL to implement a trigger that records user activity (inserts, updates, deletes) in an audit log for a given set of tables.

```
create or Replace Trigger trg audit.emp
After insert or delete or update on emp
for each row
BEGIN
  Insert into audit log (action type table_name old val,
new val changed by (values)
Can when inserting the insert when updating then
update
END;
```

Program 7

Write a code in PL/SQL to implement a trigger that automatically calculates and updates a running total column for a table whenever new rows are inserted.

```
create or Replace Trigger trg
After insert on sales
for each row
Declare total Number;
Begin Select Set running back total = total where
id = new id;
END;
```

Program 8

Write a code in PL/SQL to create a trigger that validates the availability of items before allowing an order to be placed, considering stock levels and pending orders.

```
create or replace Trigger trg
Before Insert on orders
for each row
Declare v_stock_level Number
Begin Select stock level into v_stock_level from inventory
where item id = : new item id ;
If v_stock_level < : new. Quantity then
    RAISE_APPLICATION_ERROR (-20001, Insufficient stock);
END If ;
END ;
```

Evaluation Procedure	Marks awarded
PL/SQL Procedure(5)	
Program/Execution (5)	
Viva(5)	
Total (15)	
Faculty Signature	