Date: 8/10/25

EXERCISE-17

TRIGGER

DEFINITION

AIM: To learn about triggers in PL/SQL.

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

These two variables retain the new and old values of the column updated in the database. The values in these variables can be used in the database triggers for data manipulation

SYNTAX

create or replace trigger triggername [before/after] {DML statements}
on [tablename] [for each row/statement]
begin

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 try_prevent_dept_del
BEFORE DELETE ON DEPARTMENTS FOR EACH ROW
DECLARE
    V_employee_count NUMBER;
BEGIN
    Select count(*) INTO V_employee_count
    FROM EMPLOYEES where dept_id =: OLD.dept_id;
    IF V_employee_count > 0 THEN
        RAISE_APPLICATION_ERROR(-20001, 'Cannot
delete department ID' || :OLD.dept_id || '. There are' || V_employee_count
|| ' employees still assigned to this department.');
    END IF;
END try_prevent_dept_del;
```

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 trg_check_duplicate_product
BEFORE INSERT OR UPDATE OF product_name ON
PRODUCTS FOR EACH ROW DECLARE v_count NUMBER;
BEGIN
  select COUNT(*) into v_count FROM products where
product_name = :NEW.product_name AND (:NEW.product_id IS
NULL OR product_id != :NEW.product_id);
  IF v_count > 0 THEN
    RAISE_APPLICATION_ERROR(-20002, 'Duplicate
Error: The product name '||:NEW.product_name||' already
exists.' || 'The column must contain unique values.'); END IF;
END trg_check_duplicate_product;
```

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 try_budget_threshold
BEFORE INSERT ON Projects
DECLARE
    C_budget_limit CONSTANT NUMBER := 1000000;
  V_current_total_budget NUMBER;
  BEGIN
    Select SUM(budget) INTO V_current_total_budget
    FROM Projects;
  IF(V_current_total_budget + :NEW.budget) > C_budget_limit THEN
  RAISE_APPLICATION_ERROR(-20003, 'Insertion rejected. Total
Project budget of $' || TO_CHAR(C_budget_limit, 'FM9,999,999.
00') || 'Cannot be exceeded!');
      END IF;
END try_budget_threshold;
```

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_audit_salary_changes
BEFORE UPDATE OF salary ON employees
FOR EACH ROW
BEGIN
  IF :OLD.salary != :NEW.salary THEN
      INSERT INTO employee_audit (audit_id, employee_id,
  old_salary, new_salary, change_date) VALUES (employee_
audit_seq.NEXTVAL, :OLD.employee_id, :OLD.salary, :NEW.
salary, SYSDATE); END IF; END trg_audit_salary_changes;
```

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_employee_activity_log
AFTER INSERT OR UPDATE OR DELETE ON
employees FOR EACH ROW
DECLARE
BEGIN v_dml_type VARCHAR2(10);
    IF INSERTING THEN
        v_dml_type := 'INSERT';
    ELSIF UPDATING THEN
        v_dml_type := 'UPDATE';
    ELSIF DELETING THEN
        v_dml_type := 'DELETE';
    END IF;

INSERT INTO audit_log (log_id, table_name, dml_type,
change_user, change_date) VALUES (audit_log_seq.NEXTVAL,
'EMPLOYEES', v_dml_type, USER, SYSDATE);
END trg_employee_activity_log;
```

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_update_total_sales
AFTER INSERT ON inventory_sales FOR EACH ROW
BEGIN
    UPDATE inventory_totals SET total_sales = total_sales + :NEW.
    sale_amount;
IF SQL%ROWCOUNT=0 THEN
        INSERT INTO inventory_totals (total_sales)
    VALUES (:NEW.sale_amount);
END IF;

END trg_update_total_sales;
```

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_validate_item_availability
BEFORE INSERT ON orders FOR EACH ROW
DECLARE
    v_available_stock NUMBER;
    v_pending_qty NUMBER;
BEGIN
    Select stock_level
    INTO v_available_stock
    FROM inventory
    where item_id = :NEW.item_id;
    Select NVL (SUM(quantity),0) INTO v_pending_qty
    FROM orders where item_id = :NEW.item_id AND
    status = 'PENDING';
IF :NEW.quantity > (v_available_stock - v_pending_qty) THEN
    RAISE_APPLICATION_ERROR (-20003, 'order rejected: Insufficient
Stock for Item ID' || :NEW.item_id || '. Available stock (net of pending
is only' ||(v_available_stock - v_pending_qty) || ".");
    :NEW.status := 'PENDING';
EXCEPTION
    WHEN NO_DATA_FOUND THEN
RAISE_APPLICATION_ERROR (-20004, 'order rejected:
Item ID' || :NEW.item_id || ' does not exist in inventory.');
```

END trg_validate_item_availability;

RESULT:

Thus the concept of triggers in PL/SQL is studied.

Evaluation Procedure	Marks awarded
PL/SQL Procedure(5)	5
Program/Execution (5)	5
Viva(5)	3
Total (15)	13
Faculty Signature	