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_cout from emp where dept_id := OLD.dept_id;
   IF C_Count >0 THEN
        RAISE_APPLICATION_ERROR(-20001,' Cannot delete dept');
   END IF ;
END p_del_par
```

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');
   END IF;
END;
```

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;
    threshdd CONSTANT NUMBER := 10000;
BEGIN
    Selcts SUM(amnt) INTO tot_amt from sales;
    IF total_amount + NEW.amnt > threshdd THEN
        RAISE_APPLICATION_ERROR(-20001, 'Cannot insert');
    END IF;
END;
```

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_changes
AFTER UPDATE OF Sal, dept_id ON emp
FOR EACH ROW
BEGIN
    INSERT INTO emp_audit (emp_id, changed_col, old_value, new_value)
    (SELECT :OLD.emp_id, 'Salary', TO_CHAR(:OLD.Salary), TO_CHAR(NEW:Salary)
    from dual where :OLD.Salary != NEW.salary) UNION ALL
    (SELECT: OLD.emp_id, 'dept_id', TO_CHAR(:OLD.dept_id), TO_CHAR(:new.dept_id
    FROM dual where :OLD.dept_id != NEW.dept_id;

END;
```

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 (
  (CASE WHEN INSERTING THEN 'INSERT' WHEN UPDATING THEN 'UPDATE'
   ELSE 'DELETE' END, 'employees', CASE WHEN UPDATING OR DELETING THEN :OLD.* END,
   CASE WHEN INSERTING OR UPDATING THEN :NEW.* END, USER );
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 NVL(MAX(running_total),0) + :NEW.amount
    INTO total FROM sales

    UPDATE sales SET running_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.itemid;

   If v_stock_level < :NEW.quantity THEN
        RAISE_APPLICATION_ERROR(-20001, 'Insufficient Stock');
   END IF;
END;
```

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