

14/10/25

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 trg - parent - parent delete
BEFORE DELETE ON department
FOR EACH ROW
DECLARE
    v-count NUMBER;
BEGIN
    SELECT COUNT (*) INTO v-count FROM employee WHERE
    dept_id = :old.dept_id;
    IF v-count > 0 THEN
        RAISE -APPLICATION_ERROR (-20001, 'cannot delete
        parent record. child records exist in EMPLOYEE table!');
    END IF;
END
```

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 - email
BEFORE INSERT OR UPDATE ON students
FOR EACH ROW
DECLARE
v_count number;
BEGIN
SELECT COUNT(*) INTO v_count FROM students WHERE
email = :new.email;
IF v_count > 0 THEN
RAISE_APPLICATION_ERROR(-2002, duplicate email
detected. Each mail must be unique!);
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 trg_limit_total_salary
BEFORE INSERT ON employee
FOR EACH ROW
DECLARE
    v_total NUMBER;
    v_threshold CONSTANT NUMBER := 100000;
BEGIN
    SELECT SUM(salary), 0 INTO v_total FROM
    IF (v_total + :NEW.salary) > v_threshold THEN
        RAISE_APPLICATION_ERROR (-20003, 'Total salary exceed');
    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 table employee_audit (  
  emp_id number;  
  old_salary number;  
  new_salary number;  
  change_date date;  
  changed_by VARCHAR2(30));
```

create or replace trigger try_audit
after update of salary on employee for each row

```
begin  
  insert into employee_audit (emp_id, old_salary,  
    new_salary, change_date) values (old emp_id,  
    old salary, new_salary, sysdate, user);  
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 table activity_log (  
    table_name Varchar2 (50),  
    operation_type VARCHAR2 (20);  
    user_name VARCHAR2 (30),  
    activity_date DATE);
```

Create or Replace Trigger ~~any user~~ - activity
after insert or update or delete on employee

Begin

```
INSERT INTO activity_log (table_name, operation_type,  
    user_name, activity_date) Values (:Employee,  
    ora_system, User, Sysdate);  
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 table sales (

sale_id number,
amount number,
running_total number);

Create or replace trigger trg_update_running_total
after insert on sales;

update sales set running_total = v_total where
sale_id = :NEW.sale_id;

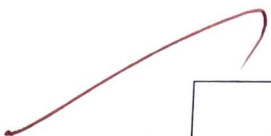
END;


/


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 try-check-stock availability
Before insert on orders
for Each row
Declare
v-stock number;
Begin
  select quantity_in_stock INTO v-stock FROM
inventory WHERE item_id = NEW
item_id;
  IF v-stock < NEW.order_quantity THEN
    RAISE_APPLICATION_ERROR (-20004, 'Insufficient
stock available for the requested item :');
  END IF;
END;
/
```



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



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