ASSIGNMENT-9

TRIGGERS AND EXPLICITS

Name: Shangirne Kharbanda

Registration Number: 20BAI1154

Slot : L47 + L48

1. Create a trigger named display_salary_changes. The trigger should fire whenever there is a delete or insert or update on customers table. The difference in salary should be computed and displayed. Assume that the table customers contains the fields id, name, age, address, salary.

```
CREATE OR REPLACE TRIGGER display_salary_change

  BEFORE DELETE OR INSERT OR UPDATE ON CUSTOMER20BAI1154

   FOR EACH ROW

   WHEN (NEW.ID > 0)

DECLARE

     sal_diff number

BEGIN

     sal_diff := :NEW.salary  - :OLD.salary;

   dbms_output.put_line('Old salary: ' || :OLD.salary);
dbms_output.put_line('New salary: ' || :NEW.salary);
dbms_output.put_line('Salary difference: ' || sal_diff);

  END;

 /

UPDATE CUSTOMER20BAI1154 SET SALARY=40000  WHERE ID=1;
```

Output:-

```
1 row(s) updated.

Old salary: 20000

New salary: 40000

Salary difference: 20000
```

2. Create a trigger named display_semester_changes. The trigger should fire whenever a student semester value is changed in student table. Assume that the student table contains the fieldsregno, name, age, dept, semester. Display the old and new value in the command line.

```
Create table STUDENT_20BAI1154(

REGNO INT,

NAME VARCHAR2(20),

AGE INT,

DEPT VARCHAR2(20),

SEMESTER INT);


Insert  into STUDENT_20BAI1154(REGNO,NAME,AGE,DEPT,SEMESTER)
VALUES(1,’Mark’,18,’CSE’, 2);

Insert  into STUDENT_20BAI1154(REGNO,NAME,AGE,DEPT,SEMESTER)
VALUES(2,’Shawna’,19,’CSE’, 3);

Insert  into STUDENT_20BAI1154(REGNO,NAME,AGE,DEPT,SEMESTER)
VALUES(3,’Rachel’,19,’CSE’, 3);

COMMIT;


CREATE OR REPLACE TRIGGER display_sem_changes

  BEFORE UPDATE ON STUDENT_20BAI1154
```

```
   FOR EACH ROW

   WHEN (NEW.regno > 0)

 BEGIN

      dbms_output.put_line('Old semester: ' || :OLD.semester);
dbms_output.put_line('New semester: ' || :NEW.semester);

   END;

   /

update  STUDENT_20BAI1154  set SEMESTER = 4 where REGNO= 3;
```

Output:-

```
1 row(s) updated.

Old semester: 3

New semester: 4
```

3. Demonstrate an example for implicit cursor – ROWCOUNT.

```
SQL> DECLARE

2                         total_rows number(2);

3                         BEGIN

4                         UPDATE CUSTOMER20BAI1154

5                         SET salary = salary + 500;

6                         IF sql%notfound THEN

7                         dbms_output.put_line('no customers selected');

8                         ELSIF sql%found THEN

9                         total_rows := sql%rowcount;

10                        dbms_output.put_line( total_rows || ' customers selected ');

11                        END IF;
```

```
12                          END;

13                          /


PL/SQL Procedure successfully completed.
```

4. Create an explicit cursor named c_customers and fetch the id, name and address of all customers in the customer table using the cursor.

```
   c_id CUSTOMER20BAI1154 id%type;

   c_name CUSTOMER20BAI1154.name%type;
c_addrCUSTOMER20BAI1154.address%type;

     CURSOR c_customers is

       SELECT id, name, address FROM CUSTOMER20BAI1154;

   BEGIN

     OPEN c_customers;

     LOOP

    FETCH c_customers into c_id, c_name, c_addr

EXIT WHEN  c_customers%notfound;

       dbms_output.put_line(c_id || ' ' || c_name || ' ' || c_addr);

    END LOOP;

    CLOSE c_customers;

  END;

  /
```

Output:-

```
 Statement processed.

1 Rachel London

2 Dolores Westworld

3 Marcus Detroit

4 Clarke  Michigan
```

5. Create an explicit cursor named c_customers and fetch the details of all customers in the customer table whose age is greater than 50 using the cursor.

```
DECLARE

     c_id CUSTOMER20BAI1154.id%type;

  c_name CUSTOMER20BAI1154.name%type;

   c_addr CUSTOMER20BAI1154.address%type;

     CURSOR c_customers is

    SELECT id, name, address FROM CUSTOMER20BAI1154  where age>50;

    BEGIN

    OPEN c_customers;

    LOOP

   FETCH c_customers into c_id, c_name, c_addr;

 EXIT WHEN c_customers%notfound;

   dbms_output.put_line(c_id || ' ' || c_name || ' ' || c_addr);

   END LOOP;

   CLOSE c_customers;
 END;
 /
```

Output:-

```
Statement processed.
3 Marcus Detroit
```

6. Create an explicit cursor named c_customers and fetch the details of all customers who are minors.

```
DECLARE
 c_id CUSTOMER20BAI1154.id%type;
  c_name CUSTOMER20BAI1154 name%type;
 c_addr CUSTOMER20BAI1154.address%type;
    CURSOR c_customers is
   SELECT id, name, address FROM CUSTOMERBAI1154 where age<18;
   BEGIN
   OPEN c_customers;
   LOOP
  FETCH c_customers into c_id, c_name, c_addr;
 EXIT WHEN c_customers%notfound;
  dbms_output.put_line(c_id || ' ' || c_name || ' ' || c_addr);
  END LOOP;
  CLOSE c_customers;
  END;
  /

Statement processed.

2 Dolores Westworld
```