| Ex.No.11 .05.25 | TRIGGERS |
|---|---|

## AIM

To implement and demonstrate the use of database triggers to perform and control INSERT, UPDATE, and DELETE function.

## CREATE TABLE

SQL>CREATE TABLE stu (student_id NUMBER PRIMARY KEY, name VARCHAR2(50), department VARCHAR2(30));

Table created.

## INSERT VALUES TO TABLE

SQL> INSERT INTO stu (student_id, name, department) VALUES (1, 'Jai', 'IT');

1 row created.

SQL> INSERT INTO stu (student_id, name, department) VALUES (2, 'Kavin', 'AI');

1 row created.

SQL> INSERT INTO stu (student_id, name, department) VALUES (3, 'Jegan', 'EEE');

1 row created.

SQL> CREATE TABLE student_audit_log (student_id NUMBER, action_time DATE, action_type VARCHAR2(10));

Table created.

```
SQL> CREATE OR REPLACE TRIGGER trg_insert_student
AFTER INSERT ON stu
FOR EACH ROW
BEGIN
   INSERT INTO student_audit_log(student_id, action_time, action_type)
   VALUES(:NEW.student_id, SYSDATE, 'INSERT');
END;
```

```
/
Trigger created.

SQL> INSERT INTO stu (student_id, name, department) VALUES (4, 'kharthik', 'FT');
1 row created.

SQL> SELECT * FROM student_audit_log;

STUDENT_ID    ACTION_TI     ACTION_TYP
----------    ---------     ----------
         4    06-MAY-25     INSERT

SQL> CREATE OR REPLACE TRIGGER trg_update_student
AFTER UPDATE ON stu
FOR EACH ROW
BEGIN
   INSERT INTO student_audit_log(student_id, action_time, action_type)
   VALUES(:NEW.student_id, SYSDATE, 'UPDATE');
END;
/
Trigger created.

SQL> UPDATE stu SET department = 'ECE' WHERE student_id = 2;
1 row updated.

SQL> SELECT * FROM student_audit_log;

STUDENT_ID    ACTION_TI     ACTION_TYP
----------    ---------     ----------
         4    06-MAY-25     INSERT
```

```
      2                06-MAY-25        UPDATE
```

```
SQL> CREATE OR REPLACE TRIGGER trg_delete_student
AFTER DELETE ON stu
FOR EACH ROW
BEGIN
   INSERT INTO student_audit_log(student_id, action_time, action_type)
   VALUES(:OLD.student_id, SYSDATE, 'DELETE');
END;
/
Trigger created.
```

```
SQL> DELETE FROM stu WHERE student_id = 3;
1 row deleted.
```

```
SQL> SELECT * FROM student_audit_log;
```

```
STUDENT_ID ACTION_TI       ACTION_TYP
---------- ---------       ----------
         4 06-MAY-25       INSERT
         2 06-MAY-25       UPDATE
         3 06-MAY-25        DELETE
```

**EXAMPLE 1**

**INSERT, UPDATE, DELETE ON STU TABLE**

```
SQL> CREATE OR REPLACE TRIGGER trg_student_all_actions
AFTER INSERT OR UPDATE OR DELETE ON stu
FOR EACH ROW
```

```
BEGIN
   IF INSERTING THEN
      INSERT INTO student_audit_log(student_id, action_time, action_type)
      VALUES(:NEW.student_id, SYSDATE, 'INSERT');
   ELSIF UPDATING THEN
      INSERT INTO student_audit_log(student_id, action_time, action_type)
      VALUES(:NEW.student_id, SYSDATE, 'UPDATE');
   ELSIF DELETING THEN
      INSERT INTO student_audit_log(student_id, action_time, action_type)
      VALUES(:OLD.student_id, SYSDATE, 'DELETE');
   END IF;
END;
/
Trigger created.

SQL>INSERT INTO stu (student_id, name, department) VALUES (5, 'Kavin', 'ECE');
1 row created.
SQL>UPDATE stu SET department = 'EEE' WHERE student_id = 1;
1 row created.
SQL>DELETE FROM stu WHERE student_id = 2;
1 row deleted.


SQL> SELECT * FROM student_audit_log;

STUDENT_ID      ACTION_TI     ACTION_TYP
----------      ---------     ----------
         4      06-MAY-25     INSERT
         2      06-MAY-25     UPDATE
         3      06-MAY-25     DELETE
```

| | | |
|---|---|---|
| 5 | 06-MAY-25 | INSERT |
| 5 | 06-MAY-25 | INSERT |
| 1 | 06-MAY-25 | UPDATE |
| 1 | 06-MAY-25 | UPDATE |
| 2 | 06-MAY-25 | DELETE |
| 2 | 06-MAY-25 | DELETE |

9 rows selected.

**EXAMPLE 2**

**PREVENT NULL VALUE FOR DEPARTMENT**

```
SQL> CREATE OR REPLACE TRIGGER trg_prevent_null_dept
BEFORE UPDATE ON stu
FOR EACH ROW
BEGIN
   IF :NEW.department IS NULL THEN
      RAISE_APPLICATION_ERROR(-20002, 'Department cannot be set to NULL.');
   END IF;
END;
/
Trigger created

UPDATE stu SET department = NULL WHERE student_id = 1;
ERROR at line 1:
ORA-20002: Department cannot be set to NULL.
ORA-06512: at "SYSTEM.TRG_PREVENT_NULL_DEPT", line 3
ORA-04088: error during execution of trigger 'SYSTEM.TRG_PREVENT_NULL_DEPT'
```

| CONTENTS | MARKS ALLOTED | MARKS OBTAINED |
|---|---|---|
| Aim,Algorithm,SQL,PL/SQL | 30 | |
| Execution and Result | 20 | |
| Viva | 10 | |
| Total | 60 | |

## RESULT

The experiment effectively demonstrated the use of database triggers in enforcing business rules and automatically maintaining audit trails.