K. J. Somaiya College of Engineering, Mumbai-77
(Autonomous College Affiliated to University of Mumbai)

Batch: A3	**Roll No.: 16010121045**
Experiment / assignment / tutorial No. 6	
Grade: AA / AB / BB / BC / CC / CD /DD	

Title: Queries based on Triggers

Objective: To be able to use trigger on table.

Expected Outcome of Experiment:

CO 3 : Use SQL for Relational database creation, maintenance and query processing

Books/ Journals/ Websites referred:

1. Dr. P.S. Deshpande, SQL and PL/SQL for Oracle 10g.Black book, Dreamtech Press
2. www.db-book.com
3. Korth, Slberchatz, Sudarshan : "Database Systems Concept", 5th Edition , McGraw Hill
4. Elmasri and Navathe,"Fundamentals of database Systems", 4th Edition,PEARSON Education.

Resources used: Postgresql

Theory

Triggers are database call-back functions, which are automatically performed/invoked when a specified database event occurs.

Triggers can be specified to fire

- Before the operation is attempted on a row (before constraints are checked and the INSERT, UPDATE or DELETE is attempted)
- After the operation has completed (after constraints are checked and the INSERT, UPDATE, or DELETE has completed)
- Instead of the operation (in the case of inserts, updates or deletes on a view)

The basic syntax of creating a trigger is as follows −

CREATE TRIGGER trigger_name [BEFORE|AFTER|INSTEAD OF] event_name

```
ON table_name
[
 -- Trigger logic goes here....
];
```

event_name could be INSERT, DELETE, UPDATE, and TRUNCATE database operation on the mentioned table table_name. You can optionally specify FOR EACH ROW after table name.

The following is the syntax of creating a trigger on an UPDATE operation on one or more specified columns of a table as follows −

```
CREATE  TRIGGER trigger_name [BEFORE|AFTER] UPDATE OF column_name
ON table_name
[
 -- Trigger logic goes here....
];
```

Implementation Screenshots (Problem Statement, Query and Screenshots of Results):

Created a table COMPANY

```
CREATE TABLE COMPANY(
   ID INT PRIMARY KEY     NOT NULL,
   NAME           TEXT    NOT NULL,
   AGE            INT     NOT NULL,
   ADDRESS        CHAR(50),
   SALARY         REAL
);
```

Data Output | Explain | Messages | Notifications

id [PK] integer	name text	age integer	address character (50)	salary real

Created table audit :

```
CREATE TABLE AUDIT(
   EMP_ID INT NOT NULL,
   ENTRY_DATE TEXT NOT NULL
);
```

Data Output | Explain | Messages | Notifications

emp_id integer	entry_date text

Created a trigger on COMPANY table

```
CREATE TRIGGER example_trigger AFTER INSERT ON COMPANY
FOR EACH ROW EXECUTE PROCEDURE auditlogfunc();

CREATE OR REPLACE FUNCTION auditlogfunc() RETURNS TRIGGER AS $example_table$
   BEGIN
      INSERT INTO AUDIT(EMP_ID, ENTRY_DATE) VALUES (new.ID, current_timestamp);
      RETURN NEW;
   END;
$example_table$ LANGUAGE plpgsql;
```

Added values to COMPANY

Query Editor Query History

```
INSERT INTO COMPANY (ID,NAME,AGE,ADDRESS,SALARY)
VALUES (1, 'Paul', 32, 'California', 20000.00 );
```

Data Output Explain Messages Notifications

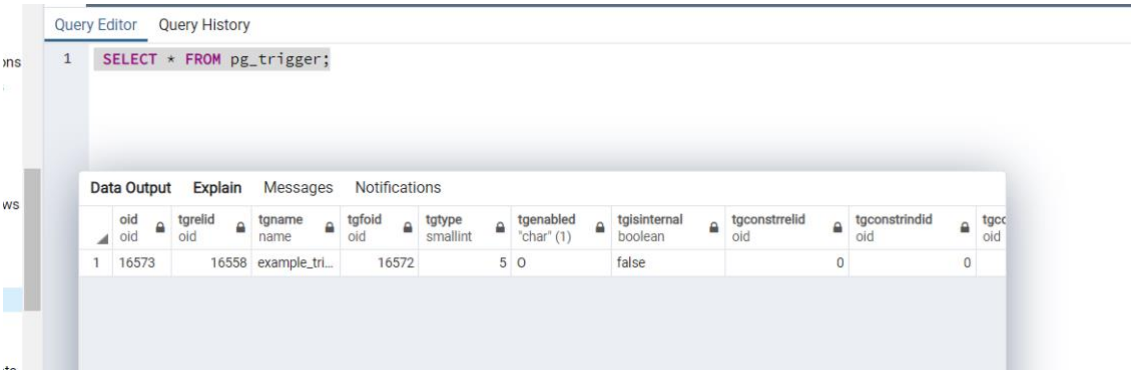
	id [PK] integer	name text	age integer	address character (50)	salary real
1	1	Paul	32	California	20000

The audit table gets updated :

Data Output Explain Messages Notifications

	emp_id integer	entry_date text
1	1	2023-04-03 11:...

Listing all triggers :



Conclusion:

The given exp on triggers were learned and implemented successfully.

Post Lab Questions:

1. **Write a trigger to count number of new tuples inserted using each insert statement.**

   CREATE TRIGGER count_inserts AFTER INSERT ON table_name

   FOR EACH ROW

   BEGIN

   SET @num_inserts = @num_inserts + 1;

   END;

2. **Trigger is special type of _____ procedure.**

   a) Stored
   b) Function
   c) View
   d) Table

3. **Triggers can be enabled or disabled with the _____ statement.**

   a) **ALTER TABLE statement**
   b) DROP TABLE statement
   c) DELETE TABLE statement
   d) None of the mentioned