Output:

Table (1): The Instructor table

	ID	name	dept_name	salary
1	1234	Einstein	ICE	95000
2	12121	Watson	CSE	90000
3	32343	Warner	EEE	60000
4	45565	Messi	CE	75000
5	45995	Neymar	CE	75000

Table (2): Table After Inserting new values.

	ID	name	dept_name	salary
1	1234	Einstein	ICE	95000
2	12121	Watson	CSE	90000
3	45565	Messi	CE	75000
4	45995	Neymar	CE	75000
5	45862	Sayeeda	CE	75000
6	58642	Khan	EEE	80000
7	89652	Sayeeda Khan	ICE	90000

Table (3): Trigger table after inserting new values.

	ID	name	dept_name	salary
1	45862	Sayeeda	CE	75000
2	58642	Khan	EEE	80000
3	89652	Sayeeda Khan	ICE	90000

Result and Discussion: We can see that first we don't have any trigger that's why we just inserted in table and delete from the table but now after triggering when we insert some values into the table the new values also added into a second table and when we remove some values from our main table the deleted values also stored in a backup table.

Experiment Number: 07

Name of the Experiment: To Study and Implementation of Triggering System on Database Table Using SQL Commands with Example.

Objectives: To understand the concept of Triggering System on Database Table Using SQL Commands.

Theory: Database triggers are indispensable mechanisms in the realm of database management. They are essentially predefined pieces of code or actions that respond automatically to specific events or modifications in a database table. These events encompass a wide array of database activities, including data insertion, updates, or deletions. Triggers are multifunctional tools that ensure data integrity by validating changes, facilitate auditing processes by tracking alterations, maintain historical records by logging data modifications, and automate various tasks, such as sending notifications or updating related data points. By acting as a built-in alert system, triggers lessen the need for manual intervention, bolster data accuracy, fortify security measures, and streamline database operations. These features make triggers indispensable for ensuring data consistency, integrity, and the efficient management.

Experimental Requirements:

- Laptop/ Desktop Computer.
- Microsoft SQL Server application.
- Microsoft SQL Server Management Studio application.

Source Code:

```
USE master;
CREATE DATABASE Triggered1;
USE Triggered1;

-- To see all the table names within the database
SELECT table_name
FROM information_schema.tables
WHERE table_type = 'Base table';

DROP TABLE IF EXISTS backup_ins;
DROP TABLE IF EXISTS backup_del;
DROP TABLE IF EXISTS instructor;

CREATE TABLE instructor (
   ID INT,
   name NVARCHAR(50),
   dept_name NVARCHAR(50),
   salary INT
);
```

```
INSERT INTO instructor VALUES (1234, 'Einstein', 'ICE', 95000);
INSERT INTO instructor VALUES (12121, 'Watson', 'CSE', 90000);
INSERT INTO instructor VALUES (32343, 'Warner', 'EEE', 60000);
INSERT INTO instructor VALUES (45565, 'Messi', 'CE', 75000);
INSERT INTO instructor VALUES (45995, 'Neymar', 'CE', 75000);

SELECT * FROM instructor;

-- Create another table for inserted value keeping
CREATE TABLE backup_ins (
   ID INT,
   name NVARCHAR(50),
   dept_name NVARCHAR(50),
   salary INT
);

-- Create another table for deleted value keeping
CREATE TABLE backup_del (
   ID INT,
   name NVARCHAR(50),
   dept_name NVARCHAR(50),
   salary INT
);

-- Creating trigger for insert
CREATE TRIGGER ins_trigger
ON instructor
AFTER INSERT
AS
BEGIN
   INSERT INTO backup_ins SELECT ID, name, dept_name, salary FROM inserted;
   PRINT 'The trigger inserted successfully';
END;

INSERT INTO instructor VALUES (45862, 'Shihab', 'CE', 75000);
INSERT INTO instructor VALUES (58642, 'Ibne', 'EEE', 80000);
INSERT INTO instructor VALUES (89652, 'Shihab', 'ICE', 90000);

SELECT * FROM instructor;
SELECT * FROM backup_ins;

-- Delete trigger
CREATE TRIGGER del_trigger
ON instructor
AFTER DELETE
AS
BEGIN
   INSERT INTO backup_del SELECT ID, name, dept_name, salary FROM deleted;
END;
```

```
DELETE FROM instructor WHERE ID = 32343;

SELECT * FROM instructor;
SELECT * FROM backup_del;
```

Output:

Table (1): The Instructor table

	ID	name	dept_name	salary
1	1234	Doland Trump	ICE	95000
2	12121	Vladimir Putin	CSE	90000
3	32343	Kim-jon-un	EEE	60000
4	45565	John Cina	CE	75000
5	45995	Virat Kohli	CE	75000

Table (2): Table After Inserting new values.

	ID	name	dept_name	salary
1	1234	Doland Trump	ICE	95000
2	12121	Vladimir Putin	CSE	90000
3	45565	John Cina	CE	75000
4	45995	Virat Kohli	CE	75000
5	45862	Shihab	CE	75000
6	58642	Ibne	EEE	80000
7	89652	Shihab	ICE	90000

Table (3): Trigger table after inserting new values.

	ID	name	dept_name	salary
1	45862	Shihab	CE	75000
2	58642	Ibne	EEE	80000
3	89652	Shihab	ICE	90000

Table (4): Trigger table after deleted one value.

	ID	name	dept_name	salary
1	32343	Kim-jon-un	EEE	60000

Result and Discussion: We can see that first we don't have any trigger that's why we just inserted in table and delete from the table but now after triggering when we insert some values into the table the new values also added into a second table and when we remove some values from our main table the deleted values also stored in a backup table.