	Bansilal Ramnath Agarwal Charitable Trust's Vishwakarma Institute of Information Technology **Department of** **Artificial Intelligence and Data Science**	
Name: Siddhesh Dilip Khairnar		
Class: SY	Division: B	Roll No: 272028
Semester: III	Academic Year: 2022-2023	
Subject Name & Code: Database Management System: ADUA21204		
Title of Assignment: Write a database trigger on Employee table. The System should keep track of the records that are being updated or deleted. The old value of updated or deleted records should be added into a new table when the Employee table is updated. Employee (employee no, employee name, join date, designation, salary).		

Assignment No.- 06

PAGE NO.:
DATE: / /

Assignment no 6

Name: Siddhesh Dilip Khairnar
PRN no: 22110398 ~~PRN~~
Roll no: 272028

Aim: Database Trigger (Row level and statement level triggers, Before and After Triggers): Write a database trigger on Employee table. The system should keep track of the record that are being updated or deleted The old value of updated or deleted record should be added into a new table when the employee table is updated. Employee (Employee no, employee name, join_date, designation, salary)

1. Brief Trigger:

→ A trigger is a stored procedure in database which automatically invokes whenever a special event in the database occurs. For example, a trigger can be invoked when a row is inserted into a specified table or when certain table columns are being updated.

Syntax: →

```
create trigger [trigger_name]
[before | after]
{insert | update | delete}
on [table_name]
[for each row]
[trigger_body]
```

2. Brief about Row level and statement level trigger.

→ Row level trigger: Row level trigger executes once for each and every ~~every~~ row in the ~~fraction~~ transaction. ~~specifically~~ specifically used for data auditing purpose. "for ~~EA~~ Each row" clause is present in CREATE TRIGGER command.

PAGE NO.:
DATE: / /

Statement level triggers: statement level triggers execute only once for each single transaction. used for enforcing additional security on the transaction performed on table. "for each row" clause is omitted in CREATE TRIGGER command.

3. different form of trigger:

→ A single SQL statement can potentially fire upto four types of triggers:

(I) BEFORE row trigger.
(II) BEFORE statement triggers.
(III) AFTER row triggers.
(IV) AFTER statement triggers.

Implementation:

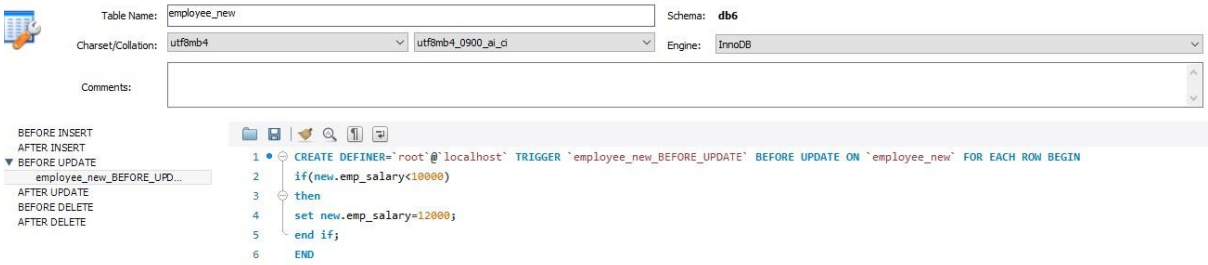
Old employee table

```
insert into employee_old values
(1,"AAA",20200101,"Intern",6000),(2,"BBB",20200101,"Intern",7000),
(3,"CCC",20180101,"Manager",35000),
(4,"DDD",20190101,"Employee",15000),(5,"EEE",20190101,"Employee",18000),
(6,"FFF",20200101,"Intern",9000);
```

Result Grid | Filter Rows: | Edit:

emp_id	emp_name	emp_join	designation	emp_salary
1	AAA	2020-01-01	Intern	6000
2	BBB	2020-01-01	Intern	7000
3	CCC	2018-01-01	Manager	35000
4	DDD	2019-01-01	Employee	15000
5	EEE	2019-01-01	Employee	18000
6	FFF	2020-01-01	Intern	9000
NULL	NULL	NULL	NULL	NULL

Before update trigger command



Before update trigger and generated table

```
update employee_new set designation="Employee" where emp_salary<15000;
```

Result Grid | Filter Rows: | Edit:

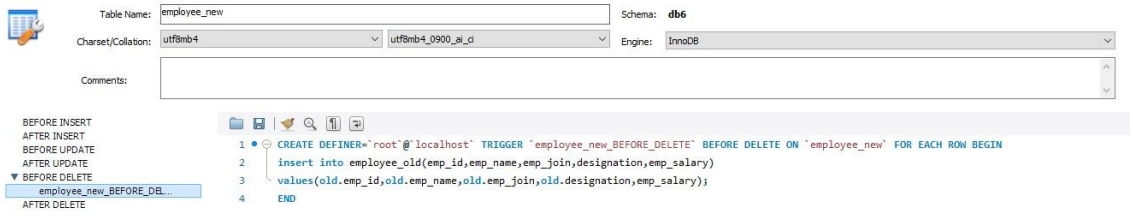
emp_id	emp_name	emp_join	designation	emp_salary
1	AAA	2020-01-01	Employee	12000
2	BBB	2020-01-01	Employee	12000
3	CCC	2018-01-01	Manager	35000
4	DDD	2019-01-01	Employee	15000
5	EEE	2019-01-01	Employee	18000
6	FFF	2020-01-01	Employee	12000
NULL	NULL	NULL	NULL	NULL

New Employee table

Result Grid | Filter Rows: | Edit:

emp_id	emp_name	emp_join	designation	emp_salary
1	AAA	2020-01-01	Employee	12000
2	BBB	2020-01-01	Employee	12000
3	CCC	2018-01-01	Manager	35000
4	DDD	2019-01-01	Employee	15000
5	EEE	2019-01-01	Employee	18000
6	FFF	2020-01-01	Employee	12000
NULL	NULL	NULL	NULL	NULL

Before delete trigger command



Old Employee table containing deleted values

```
1 •  SELECT * FROM db6.employee_old;
```

Result Grid | Filter Rows: | Edit:

emp_id	emp_name	emp_join	designation	emp_salary
1	AAA	2020-01-01	Employee	0
2	BBB	2020-01-01	Employee	0
6	FFF	2020-01-01	Employee	0
NULL	NULL	NULL	NULL	NULL

New Employee table

```
1     delete from employee_new where emp_salary=12000;
2
3 •   select * from employee_new;
```

Result Grid | Filter Rows: | Edit:

emp_id	emp_name	emp_join	designation	emp_salary
3	CCC	2018-01-01	Manager	35000
4	DDD	2019-01-01	Employee	15000
5	EEE	2019-01-01	Employee	18000
NULL	NULL	NULL	NULL	NULL