MySQL Triggers – Log of Records (LAB Lecture)

Department of Information Technology
Database Systems II (IT216)
Spring 2025-2026
Week 13 – April 28, 2026
Lecturer: Soma Soleimanzadeh

1

Contents

- **Triggers** in MySQL
- Syntax of Creating a **Trigger** in MySQL
- **Triggers** for
  1. Log of Record (The type of triggers in this lecture)
  2. Validating Input Data

2

Why We Need Triggers in Database?

- Triggers are useful in many situations. Some of the main reasons for using triggers are:
  - Keeping a **Log of Records**
  - Validating Input Data
  - Enforcing Business Rules

3

Syntax of Creating TRIGGER in MySQL

```
DELIMITER //
CREATE TRIGGER trigger_name
(BEFORE | AFTER) (INSERT | UPDATE | DELETE) ON table_name
FOR EACH ROW
BEGIN
    <Trigger Statements>
END//
DELIMITER ;
```

4

Log of Records Scenario

Employee Table

EID	Ename	Age

Employee Table

EID	Ename	Age
1	Hasan	44
2	Lana	~~36~~ 37

insert into Employee(Ename, Age) **values** ('Hasan', 44), ('Lana', 36);

update Employee **set** Age = 37 **where** EID = 2;

Emp_Log Table

id	ActionName	OldAge	NewAge	ByUser	ActionDate	ActionTime
1	Insert	Null	44	root	26 April 2026	06:12:45 PM
2	Insert	Null	36	root	26 April 2026	06:12:45 PM
3	Update	36	37	root	27 April 2026	09:00:31 AM

Log of Records Scenario

- Suppose there is a table to store employees' data in our database.
- We want to make a log of records that let us know about any modifications that happen on the table (**Employee** table).
- We will create an empty table (for example, the **Emp_Log** table) to store the log of records. For example:
  - Which user did which type of modification at which date and time on the **Employee** table?

6

Log of Records Scenario

- We need **3 triggers** that make a log of records when any modification (INSERT, UPDATE or DELETE) happens on **Employee** table.

  - First trigger is activated when **INSERT** happens.
  - Second trigger is activated when **DELETE** happens.
  - Third trigger is activated when **UPDATE** happens.

7

Let's Create Database and Tables

- Create **Company** database, and activate it.
- Create both **Employee** and **Emp_Log** tables.

Employee Table

EID	Ename	Age

Emp_Log Table

id	ActionName	OldAge	NewAge	ByUser	ActionDate	ActionTime

8

Let's Create Database and Tables

```
create database company;
use company;

create table employee
(EID int auto_increment primary key,
Ename varchar(50),
age int);
```

```
create table emp_log
(id int auto_increment primary key,
actionName varchar(10),
oldAge int,
newAge int,
byUser varchar(20),
actionDate date,
actionTime time);
```

9

INSERT Trigger on Employee

```
DELIMITER //
CREATE TRIGGER insert_tg_emp
AFTER INSERT ON Employee
FOR EACH ROW
BEGIN
        insert into Emp_Log
        set
            actionName = 'Insert',
            newAge = new.Age,
            byUser = user(),
            ActionDate = current_date(),
            ActionTime = current_time();
END//
DELIMITER ;
```

10

DELETE Trigger on Employee

```
DELIMITER $$
CREATE TRIGGER delete_tg_emp
AFTER DELETE ON Employee
FOR EACH ROW
BEGIN
        insert into Emp_Log
        set
            actionName = 'Delete',
            oldAge = old.Age,
            byUser = user(),
            ActionDate = current_date(),
            ActionTime = current_time();
END$$
DELIMITER ;
```

11

UPDATE Trigger on Employee

```
DELIMITER //
CREATE TRIGGER update_tg_emp
AFTER UPDATE ON Employee
FOR EACH ROW
BEGIN
        insert into Emp_Log
        set
            actionName = 'Update',
            newAge = new.Age,
            oldAge = old.Age,
            byUser = user(),
            ActionDate = current_date(),
            ActionTime = current_time();
END//
DELIMITER ;
```

12