What is a Trigger in MySQL?

Answer:
A **trigger** in MySQL is a **special database object** that **automatically executes (fires)** when a specific event occurs on a table.
These events are:

- **INSERT**
- **UPDATE**
- **DELETE**

Triggers help enforce business rules, maintain logs, and ensure data consistency **without manual execution**.

□ Key Points About Triggers

- Automatically executed (no need to call them)
- Associated with a **table**
- Executes **BEFORE** or **AFTER** an event
- Cannot be called directly
- Useful for auditing, validation, and maintaining related data

---

□ Trigger Timing

1. **BEFORE Trigger**
   Executes **before** the actual database operation
   □ Used for **validation or modification** of data
2. **AFTER Trigger**
   Executes **after** the operation is completed
   □ Used for **logging or auditing**

□ Trigger Syntax (Basic Structure)

```
CREATE TRIGGER trigger_name
BEFORE | AFTER INSERT | UPDATE | DELETE
ON table_name
FOR EACH ROW
BEGIN
   -- SQL statements
END;
```

□ Example 1: BEFORE INSERT Trigger

□ Scenario:

Ensure salary is **not negative** before inserting data.

Table:

```
CREATE TABLE employees (
    id INT,
    name VARCHAR(50),
    salary INT
);
```

Trigger:

```
CREATE TRIGGER check_salary
BEFORE INSERT ON employees
FOR EACH ROW
BEGIN
    IF NEW.salary < 0 THEN
        SET NEW.salary = 0;
    END IF;
END;
```

□ Explanation:

- `NEW.salary` refers to the **new value being inserted**
- If salary is negative, it is automatically set to 0
- Trigger runs **before INSERT**

□ Example 2: AFTER INSERT Trigger (Audit Log)

□ Scenario:

Store a log whenever a new employee is added.

Log Table:

```
CREATE TABLE employee_log (
    emp_id INT,
    action VARCHAR(20),
    action_time TIMESTAMP
);
```

Trigger:

```
CREATE TRIGGER after_employee_insert
AFTER INSERT ON employees
FOR EACH ROW
BEGIN
    INSERT INTO employee_log
    VALUES (NEW.id, 'INSERT', NOW());
END;
```

□ Explanation:

- Runs **after a row is inserted**
- Records employee ID, action type, and time
- Used for **auditing purposes**

□ Example 3: AFTER DELETE Trigger

```
CREATE TRIGGER after_employee_delete
AFTER DELETE ON employees
FOR EACH ROW
BEGIN
    INSERT INTO employee_log
    VALUES (OLD.id, 'DELETE', NOW());
END;
```

□ Explanation:

- `OLD.id` refers to **deleted data**
- Used to track deleted records

□ OLD vs NEW Keywords

Keyword	Used In	Meaning
NEW	INSERT, UPDATE	New row data
OLD	UPDATE, DELETE	Old row data

□ Advantages of Triggers

✔ Automatic execution
✔ Improves data integrity

✔ Reduces application-side logic
✔ Useful for auditing and logging

□ Disadvantages of Triggers

□ Hard to debug
□ Can affect performance
□ Hidden logic (not easily visible)

□ Simple Interview Answer (Short)

A trigger is a database object that automatically executes before or after INSERT, UPDATE, or DELETE operations on a table to enforce rules or maintain logs.