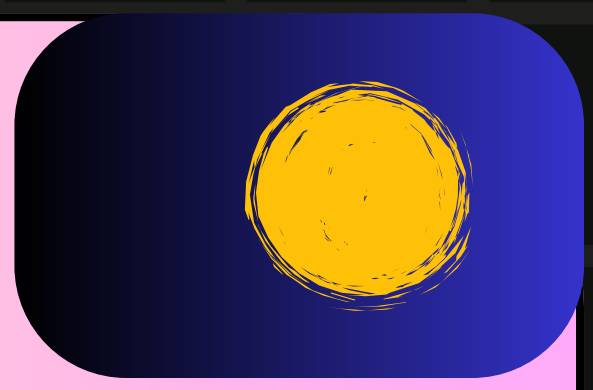
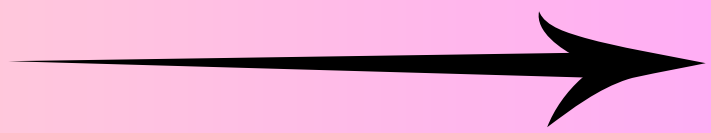


Exploring SQL Triggers!

@Vaishnavi Dauale



1

What are SQL Triggers?

- SQL triggers are automatic sets of SQL commands triggered by changes like INSERT, UPDATE, or DELETE in a table. They enforce rules, maintain data integrity, and streamline database operations.

- A trigger is called a special procedure because it cannot be called directly like a stored procedure.

2

Why Triggers Matter?

- By leveraging SQL triggers, you can automate critical tasks, ensure data accuracy, and maintain robust auditing mechanisms effortlessly. If you’re working with databases, understanding triggers is a must!

3

When to Use Triggers?

- Triggers automate actions in response to specific database events, perfect for scenarios where automatic handling of data changes is crucial. For instance, tracking changes in a dynamic table or enforcing complex business rules.

4

Basic Syntax of Trigger :

```
CREATE TRIGGER schema.trigger_name
ON table_name
AFTER {INSERT, UPDATE, DELETE}
AS
{SQL_Statements}
```

5

Steps to follow Create and Use Triggers in SQL :

1. Create table :

- First, you need to create two tables EmployeeDetail and employee.

```sql
CREATE TABLE EmployeeDetail (
    id INT AUTO_INCREMENT PRIMARY KEY,
    employee_Number INT NOT NULL,
    last_name VARCHAR(50) NOT NULL,
    change_date DATETIME DEFAULT NULL,
    action VARCHAR(50) DEFAULT NULL
);
```

6

Creating employee table :

```sql
CREATE TABLE employee (
    id INT AUTO_INCREMENT PRIMARY KEY,
    employee_Number INT,
    last_name VARCHAR(50),
    change_date DATETIME
);
```

7

2. Inserting Initial Data :

- Insert some initial data into the employee table.

```
INSERT INTO employee (employee_Number, last_name, change_date)
VALUES (101, 'singh', '2020-02-28');
```

3. Creating a Trigger :

- Create a trigger that is executed before an update on the employee table. This trigger will log the changes into the EmployeeDetail table.

```
CREATE TRIGGER before_on_employee_update
BEFORE UPDATE ON employee
FOR EACH ROW
INSERT INTO EmployeeDetail (action, employee_Number, last_name, change_date)
VALUES ('update', OLD.employee_Number, OLD.last_name, NOW());
```

4. Using the Trigger :

Now, let's update the employee table and see how the trigger works.

Update the employee table :

```
UPDATE employee
SET last_name = 'kumar'
WHERE employee_Number = 101;
```

10

5. Checking the Log Table :

Finally, check the EmployeeDetail table to see the log of changes.

```
SELECT * FROM EmployeeDetail;
```

11

Triggers V/S Store Procedure :

Stored Procedure	Triggers
Set of SQL Statements, which has to be explicitly called by user, application or trigger.	Set of SQL Statements, which has to be implicitly fired when a specific event occurs.
By using the exec command, we can execute stored procedure	Triggers cannot be executed directly by a user. Only when the corresponding events are fired, triggers are created
Parameter can be passed as input	Parameter cannot be passed as input
Call a stored procedure from another stored procedure	Cannot directly call another trigger within a trigger
Can return zero or n values	Cannot return values

Thank You

@Vaishnavi Dauale