Triggers

Topic 4 Lesson 5
Creating active databases

Trigger DB Object

Trigger: subroutine that starts automatically if specified change occurs to the DB

A trigger has three parts:

Event

Change to the database that activates the trigger

Condition

Query or test that is run when the trigger is activated

Action

Procedure that is executed when the trigger is activated, and its ***Condition*** is true

Trigger options

Event: can be INSERT, DELETE, or UPDATE of a DB table

Condition:

- Condition can be a true/false statement
  - All employee salaries are less than $100K
- Condition can be a query
  - Interpreted as true if and only if answer set is not empty

Action: can perform DB queries and updates that depend on:

- Answers to query in condition part
- Old and new values of tuples modified by the statement that activated the trigger
  - Old.field1 or New.field1
- Action can also contain data-definition commands, e.g., create new tables

When to fire a trigger

- Triggers can be a row-level or a statement-level trigger
  - Row-level trigger: trigger executed once per modified record
  - Statement level trigger: executed once per activating statement (not found in MySQL)

- Triggers can be executed before or after the activating SQL statement
  - Consider triggers on insertions
    - Trigger that initializes a variable for counting how many new tuples are inserted: execute **trigger before insertion**
    - Trigger that updates this count variable for each inserted tuple: **execute after each tuple is inserted** (might need to examine values of tuple to determine action)
    - Trigger can also be run **in place of the action**

Trigger example

CREATE TRIGGER <trigger-name> Trigger_time Trigger_event
ON table_name
FOR EACH ROW
BEGIN
END

Syntax

- Trigger_time is [BEFORE | AFTER]
- Trigger_event [INSERT | UPDATE | DELETE]
- Other key words – OLD AND NEW
- Naming convention for a trigger trigger_time_tablename_trigger_event
- Found in the directory associated with the database
  - File tablename.tdg – maps the trigger to the corresponding table
  - Triggername.trn contains the trigger definition

Trigger syntax

```
CREATE TRIGGER
  trigger_name
   {BEFORE|AFTER}
  {INSERT| UPDATE|
DELETE} ON
table_name
   FOR EACH ROW
trigger_body
```

```
DELIMITER //

CREATE TRIGGER
  vendors_before_update
   BEFORE UPDATE ON
  vendors
   FOR EACH ROW
BEGIN
   SET NEW.vendor_state =
  UPPER(NEW.vendor_state)
  ;
END//
```

Trigger example

```
CREATE TRIGGER trigger_after_sailor_insert
      AFTER INSERT ON SAILORS
      FOR EACH ROW
          BEGIN
             INSERT  INTO YoungSailors
                                (sid, name, age, rating)
             SELECT sid, name, age, rating
                    FROM  New.Sailors N
                                WHERE New.age <= 18;
           END;
```

Trigger has access to **NEW** and **OLD** field values

Trigger example (2)

Triggers that insert rows into the table

```
DELIMITER //

CREATE TRIGGER invoices_after_insert
  AFTER INSERT ON invoices
  FOR EACH ROW
BEGIN
    INSERT INTO invoices_audit VALUES
    (NEW.vendor_id, NEW.invoice_number,
     NEW.invoice_total, 'INSERTED', NOW());
END//

CREATE TRIGGER invoices_after_delete
  AFTER DELETE ON invoices
  FOR EACH ROW
BEGIN
    INSERT INTO invoices_audit VALUES
    (OLD.vendor_id, OLD.invoice_number,
     OLD.invoice_total, 'DELETED', NOW());
END//
```

Reviewing your trigger

Go to the trigger directory and read the file (.trg)
Program Data\MySQL\MySQL8.0\data\<db-name>*.trg

Use the DBMS to locate the trigger for you

Triggers in current schema

SHOW TRIGGERS;

ALL Triggers in DBMS using the System Catalog

```
SELECT * FROM Information_Schema.Triggers
WHERE Trigger_schema = 'database_name' AND
   Trigger_name = 'trigger_name';

SELECT trigger_schema, trigger_name, action_statement FROM
information_schema.triggers;
```

Trouble with triggers

- One DB Action can trigger multiple triggers
  - Execution of the order of the triggers used to be arbitrary, now is determined by trigger create time. The trigger can also specify if it should precede or follow a specific trigger (FOLLOWS or PRECEDES are the keywords)
- Challenge: Trigger action can fire other triggers
  - Very difficult to reason about what exactly will happen
    - Trigger can fire “itself” again
  - Unintended effects possible
- Introducing Triggers leads you to deductive databases
  - Need rule analysis tools that allow you to deduce truths about the data

MySQL limits the use of triggers

- Triggers not introduced until MySQL 5.0
- Triggers cannot be activated for foreign key actions
- No triggers on the MySQL system catalog database
- Active triggers are not notified when the meta data of the table is changed while it is running
- **No recursive triggers**
- Triggers cannot modify/alter the table that is already being used
  - For example the table that triggered it

Modifying a trigger

There is no edit operation defined for a trigger

```
CREATE TRIGGER …
DROP TRIGGER <TRIGGERNAME>;
CREATE TRIGGER …
```

Summary

Triggers allow you to create active databases, databases that perform an action when a specific modification operation is performed on a table.