Eventos en MySQL. Triggers Before Update.

Introducción a los triggers MySQL BEFORE UPDATE

Los disparadores MySQL BEFORE UPDATE se invocan automáticamente antes de que ocurra un evento de actualización en la tabla asociada con los disparadores.
Aquí está la sintaxis para crear un disparador MySQL BEFORE UPDATE:

```
CREATE TRIGGER trigger_name
BEFORE UPDATE
ON table_name FOR EACH ROW
trigger_body
```

En la sintaxis podemos apreciar que:
Primero, se especifia el nombre del disparador que desea crear después de las palabras clave CREATE TRIGGER.
En segundo lugar, use la cláusula BEFORE UPDATE para especificar el momento en el que invocar el trigger.
En tercer lugar, el nombre de la tabla a la que pertenece el disparador, después de la palabra clave ON.
Finalmente, el cuerpo del disparador que contiene una o más declaraciones.
Si tiene más de una instrucción en el trigger_body, debe usar el bloque BEGIN END. Además, debe cambiar el delimitador predeterminado de la siguiente manera:

```
DELIMITER $$

CREATE TRIGGER trigger_name
    BEFORE UPDATE
    ON table_name FOR EACH ROW
BEGIN
    -- statements
END$$

DELIMITER ;
```

En un disparador BEFORE UPDATE, puede actualizar los valores NEW pero no puede actualizar los valores OLD.

Ejemplo de trigger MySQL BEFORE UPDATE

Veamos un ejemplo de uso de un disparador BEFORE UPDATE.

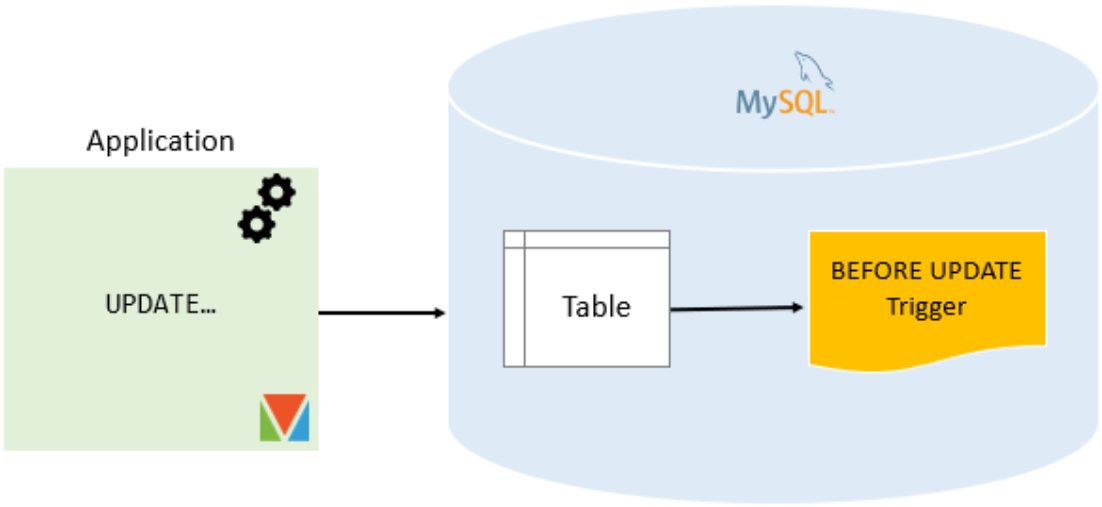

Figura 1: MySQL BEFORE UPDATE Trigger

Configurar una tabla de muestra

Primero, cree una nueva tabla llamada ventas (sales) para almacenar volúmenes de ventas:

```
DROP TABLE IF EXISTS sales;

CREATE TABLE sales (
    id INT AUTO_INCREMENT,
    product VARCHAR(100) NOT NULL,
    quantity INT NOT NULL DEFAULT 0,
    fiscalYear SMALLINT NOT NULL,
    fiscalMonth TINYINT NOT NULL,
    CHECK(fiscalMonth >= 1 AND fiscalMonth <= 12),
    CHECK(fiscalYear BETWEEN 2000 and 2050),
    CHECK (quantity >=0),
    UNIQUE(product, fiscalYear, fiscalMonth),
    PRIMARY KEY(id)
);
```

En segundo lugar, inserte algunas filas en la tabla **sales**:

```
INSERT INTO sales(product, quantity, fiscalYear, fiscalMonth)
VALUES
    ('2003 Harley-Davidson Eagle Drag Bike',120, 2020,1),
    ('1969 Corvair Monza', 150,2020,1),
    ('1970 Plymouth Hemi Cuda', 200,2020,1);
```

Tercero, consulte los datos de la tabla de ventas para verificar la inserción:

```
SELECT * FROM sales;
```

Creación de un ejemplo de trigger BEFORE UPDATE:

La siguiente instrucción crea un disparador BEFORE UPDATE en la tabla de ventas.

```
DELIMITER $$

CREATE TRIGGER before_sales_update
BEFORE UPDATE
ON sales FOR EACH ROW
BEGIN
    DECLARE errorMessage VARCHAR(255);
    SET errorMessage = CONCAT('The new quantity ',
                        NEW.quantity,
                        ' cannot be 3 times greater than the current quantity ',
                        OLD.quantity);

    IF new.quantity > old.quantity * 3 THEN
        SIGNAL SQLSTATE '45000'
            SET MESSAGE_TEXT = errorMessage;
    END IF;
END $$

DELIMITER ;
```

El disparador se desencadena automáticamente antes de que ocurra un evento de actualización para cada fila en la tabla de ventas.
Si actualiza el valor en la columna de cantidad a un nuevo valor que es 3 veces mayor que el valor actual, el disparador genera un error y detiene la actualización.
Examinemos el disparador en detalle:
Primero, el nombre del disparador es before_sales_update especificado en la cláusula CREATE TRIGGER:

```
CREATE TRIGGER before_sales_update
```

En segundo lugar, el evento disparador es:

```
BEFORE UPDATE
```

Cuarto, declare una variable y establezca su valor a un mensaje de error. Tenga en cuenta que, en un disparador BEFORE, puede acceder a los valores antiguos y nuevos de las columnas a través de los modificadores **OLD** y **NEW**.

```
DECLARE errorMessage VARCHAR(255);
SET errorMessage = CONCAT('The new quantity ',
                        NEW.quantity,
                        ' cannot be 3 times greater than the current quantity ',
                        OLD.quantity);
```

Tenga en cuenta que utilizamos la función CONCAT() para formar el mensaje de error.
Finalmente, usamos la instrucción IF-THEN para verificar si el nuevo valor es 3 veces mayor que el valor anterior, en ese caso se genera un error utilizando la instrucción SIGNAL:

```
IF new.quantity > old.quantity * 3 THEN
        SIGNAL SQLSTATE '45000'
            SET MESSAGE_TEXT = errorMessage;
END IF;
```

Probando el disparador MySQL BEFORE UPDATE

Primero, actualizamos la cantidad (quantity) de la fila con id 1 a 150:

```
UPDATE sales
SET quantity = 150
WHERE id = 1;
```

Funciona porque la nueva cantidad no viola la regla.
Tercero, actualice la cantidad (quantity) de la fila con id 1 a 500:

```
UPDATE sales
SET quantity = 500
WHERE id = 1;
```

MySQL emite este error:

```
Error Code: 1644. The new quantity 500 cannot be 3 times greater than
the current quantity 150
```

En este caso, el disparador encontró que la nueva cantidad causa una violación a la regla y provoca un error.
Finalmente, use SHOW ERRORS para mostrar el error:

```
SHOW ERRORS;
```