Eventos en MySQL. Triggers Before Delete.

Introducción a los disparadores MySQL BEFORE DELETE

Los triggers de MySQL BEFORE DELETE se disparan automáticamente antes de que ocurra un evento de eliminación en una tabla.
Aquí está la sintaxis básica de crear un disparador MySQL BEFORE DELETE:

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

En esta sintaxis:
Primero, se especifica el nombre del disparador que desea crear, después de las palabras clave CREATE TRIGGER.
En segundo lugar, viene la cláusula BEFORE DELETE para especificar que el disparador se invoca justo antes de un evento de eliminación.
En tercer lugar, vemos el nombre de la tabla con la que está asociado el activador después de la palabra clave ON.
Finalmente, el cuerpo del disparador que consiste en una o más declaraciones que se ejecutan cuando se activa el disparador.
Tenga en cuenta que si el cuerpo del trigger tiene varias instrucciones, se debe usar el bloque BEGIN END para .[en]volver.[es]tas instrucciones, además de cambiar temporalmente el delimitador predeterminado de la siguiente manera:

```
DELIMITER $$

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

DELIMITER ;
```

En un disparador BEFORE DELETE, puede acceder a la fila ANTIGUA (OLD) pero no puede actualizarla. Además, no hay una fila NUEVA (NEW) en el disparador BEFORE DELETE.

Ejemplo de disparador MySQL BEFORE DELETE

Veamos el siguiente ejemplo de disparador BEFORE DELETE.

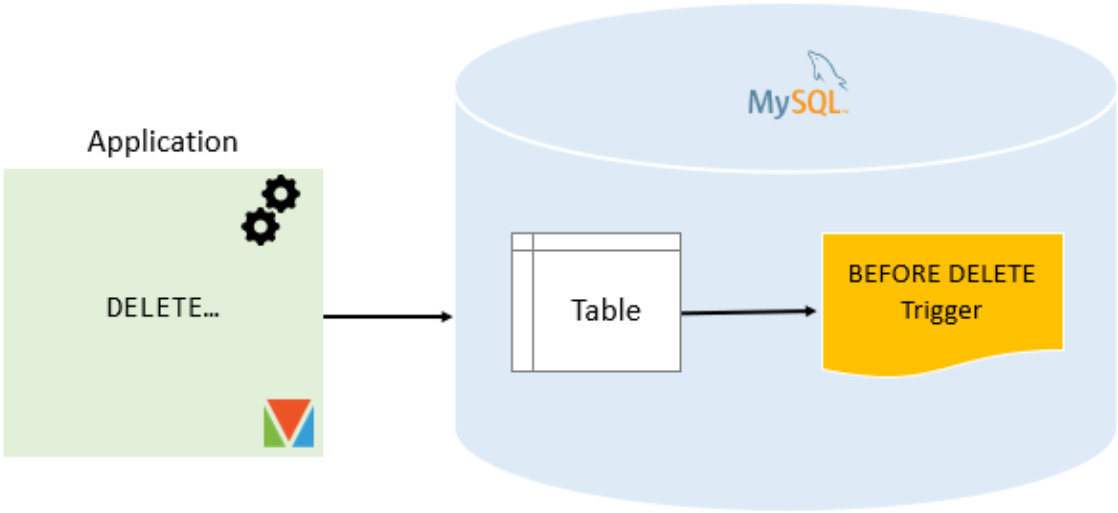

Figura 1: MySQL AFTER UPDATE Trigger

Configurar una tabla de muestra

Primero, creamos una nueva tabla llamada Salaries (Salarios) que almacene la información salarial de los empleados

```
DROP TABLE IF EXISTS Salaries;

CREATE TABLE Salaries (
    employeeNumber INT PRIMARY KEY,
    validFrom DATE NOT NULL,
    amount DEC(12 , 2 ) NOT NULL DEFAULT 0
);
```

En segundo lugar, inserte algunas filas en la tabla Salarios:

```
INSERT INTO salaries(employeeNumber,validFrom,amount)
VALUES
    (1002,'2000-01-01',50000),
    (1056,'2000-01-01',60000),
    (1076,'2000-01-01',70000);
```

Tercero, cree una tabla que almacene el salario eliminado:

```
DROP TABLE IF EXISTS SalaryArchives;

CREATE TABLE SalaryArchives (
    id INT PRIMARY KEY AUTO_INCREMENT,
    employeeNumber INT PRIMARY KEY,
    validFrom DATE NOT NULL,
    amount DEC(12 , 2 ) NOT NULL DEFAULT 0,
    deletedAt TIMESTAMP DEFAULT NOW()
);
```

Creación del trigger BEFORE DELETE de nuestro ejemplo

El siguiente disparador BEFORE DELETE inserta una nueva fila en la tabla SalarioArchivos antes de que se elimine una fila de la tabla Salarios.

```
DELIMITER $$

CREATE TRIGGER before_salaries_delete
BEFORE DELETE
ON salaries FOR EACH ROW
BEGIN
    INSERT INTO SalaryArchives(employeeNumber,validFrom,amount)
    VALUES(OLD.employeeNumber,OLD.validFrom,OLD.amount);
END$$

DELIMITER ;
```

En este disparador:
Primero, vemos que el nombre del disparador es before_salaries_delete especificado en la cláusula CREATE TRIGGER:

```
CREATE TRIGGER before_salaries_delete
```

En segundo lugar, el evento disparador es:

```
BEFORE DELETE
```

En tercer lugar, la tabla con la que está asociado el activador es la tabla Salarios:

```
ON Salaries FOR EACH ROW
```

Finalmente, dentro del cuerpo del disparador, insertamos la fila eliminada, de la tabla Salaries, en la tabla SalaryArchives.

Probando el dispardor MySQL BEFORE DELETE

Primero, eliminamos una fila de la tabla Salarios:

```
DELETE FROM salaries
WHERE employeeNumber = 1002;
```

En segundo lugar, consulta los datos de la tabla SalaryArchives:

```
SELECT * FROM SalaryArchives;
```

Se invocó el disparador y se insertó una nueva fila en la tabla SalaryArchives.
Tercero, elimine todas las filas de la tabla Salarios:

```
DELETE FROM salaries;
```

Finalmente, consulta los datos de la tabla SalaryArchives:

```
SELECT * FROM SalaryArchives;
```

El disparador se activó dos veces porque la instrucción DELETE eliminó dos filas de la tabla Salarios.