

Que traducido sería algo así como **disparadores**, son oyentes, que **se mantienen a la escucha de los eventos** que puedan producirse en una tabla (**insert, update y delete**) y ejecutan un código de manera automática antes (before) o después (after) de que se produzca este evento.

Para referenciar las columnas de antes y después de que el evento se haya disparado, se usan las palabras clave **OLD** y **NEW**. Con la sentencia insert solo se permite NEW, con update se permiten ambas y con delete solo OLD.

Sintaxis necesaria para crear un trigger

```
CREATE TRIGGER <nombre>
  {BEFORE | AFTER}
  {INSERT | UPDATE | DELETE}
  ON
  <tablename>
  FOR EACH ROW
  BEGIN
    <sentenciasSQL>
  END;
```

Para poner nombre a los triggers **es conveniente seguir una convención** que hará más fácil identificar sobre que evento y tabla actúa el trigger. Esta sería una buena forma de nombrar nuestros triggers:

NombreTabla + “_” + abreviatura_tipo_trigger

Por lo tanto para nombrar un trigger que se ejecuta sobre la tabla T1 antes de un update lo haríamos de la siguiente forma si seguimos esta convención:

“T1_BU”

*Esto **puede ser útil, pero no es necesario**, puedes poner el nombre que quieras a tu trigger.*

1.- Cread la tabla NOTAS en la base de Datos P19.

CREATE TABLE Notas(nombre varchar(20), nota dec(3,1));

Inserta los valores:

("Arturo",4.2)

("Olivia",8.8)

("Braulio",10)

("Elena",1)

```
MariaDB [P19]> DROP DATABASE P19;
Query OK, 1 row affected (0.05 sec)

MariaDB [(none)]> CREATE DATABASE P19;
Query OK, 1 row affected (0.00 sec)

MariaDB [(none)]>
MariaDB [(none)]> USE P19;
Database changed
MariaDB [P19]>
MariaDB [P19]> CREATE TABLE Notas(nombre varchar(20), nota dec(3,1));
Query OK, 0 rows affected (0.04 sec)

MariaDB [P19]>
MariaDB [P19]> INSERT INTO Notas VALUES("Arturo",4.2),("Olivia",8.8),("Braulio",10),("Elena",1);
Query OK, 4 rows affected (0.00 sec)
Records: 4  Duplicates: 0  Warnings: 0
```

Cread el trigger:

CREATE TRIGGER NOTAS_BI **before** **INSERT ON** NOTAS
FOR each row
begin

```
IF NEW.nota < 0 THEN
    SET NEW.nota = 0;
ELSEIF NEW.nota > 10 THEN
    SET NEW.nota = 10;
END IF;
```

end//

```
MariaDB [P19]> DELIMITER //
MariaDB [P19]> CREATE TRIGGER NOTAS_BI BEFORE INSERT ON Notas FOR EACH ROW
-> BEGIN
->     IF NEW.nota < 0 THEN
->         SET NEW.nota = 0;
->     ELSEIF NEW.nota > 10 THEN
->         SET NEW.nota = 10;
->     END IF;
-> END//
Query OK, 0 rows affected (0.02 sec)
```

Inserta los valores en la tabla NOTAS:

("Amelia",7)

("Bernardo",18.6)

("Carlota",-5)

("Dario",2.5)

```
MariaDB [P19]> INSERT INTO Notas VALUES("Amelia",7),("Bernardo",18.6),("Carlota",-5),("Dario",2.5);
Query OK, 4 rows affected (0.01 sec)
Records: 4 Duplicates: 0 Warnings: 0
```

Di que contiene la tabla **NOTAS** y explica por qué.

Los valores insertados menores de 0 o mayores de 10 cambian a 0 o 10 respectivamente.

2.- Cread la tabla BAJAS

CREATE TABLE Bajas(nombre varchar(20), fecha date);

Corrige los errores de este Trigger y créalo:

CREATE TRIGGER NOTAS_BI **after DELETE ON** NOTAS

FOR each row

begin

Al ser solo una instrucción no hacen falta BEGIN y END

INSERT INTO Bajas VALUES(NEW.nombre,curdate());

end//

```
MariaDB [P19]> DELIMITER //
MariaDB [P19]> CREATE TRIGGER NOTAS_AD AFTER DELETE ON Notas FOR EACH ROW
-> BEGIN
->     INSERT INTO Bajas VALUES(OLD.nombre,CURDATE());
-> END//
Query OK, 0 rows affected (0.01 sec)

MariaDB [P19]> DELIMITER ;
```

Borra los registros de:

Elena, Amelia,Bruno y Dario.

```
MariaDB [P19]> DELETE FROM Notas WHERE nombre="Elena" OR nombre="Amelia" OR nombre="Bruno" OR nombre="Dario";
Query OK, 3 rows affected (0.01 sec)
```

Di que hay en la tabla BAJAS.

Están los nombres de los usuario eliminados y la fecha de eliminación.

3.- Crea el trigger:

```
mysql> delimiter //  
CREATE TRIGGER Notas_BU BEFORE UPDATE ON Notas  
FOR EACH ROW  
BEGIN  
    IF NEW.nota < OLD.nota THEN  
        SET NEW.nota = OLD.nota;  
    ELSE  
        SET NEW.nota = OLD.nota+2;  
    END IF;  
END;//  
mysql> delimiter ;
```

```
MariaDB [P19]> DELIMITER //  
MariaDB [P19]> CREATE TRIGGER Notas_BU BEFORE UPDATE ON Notas FOR EACH ROW  
-> BEGIN  
->     IF NEW.nota < OLD.nota THEN  
->         SET NEW.nota = OLD.nota;  
->     ELSE  
->         SET NEW.nota = OLD.nota+2;  
->     END IF;  
-> END//  
Query OK, 0 rows affected (0.01 sec)  
  
MariaDB [P19]> DELIMITER ;
```

Modifica las siguientes NOTAS:

("Arturo",4.2) → 9

("Olivia",8.8) → 3

("Braulio",10) → 4

```
MariaDB [P19]> UPDATE Notas SET nota=9 WHERE nombre="Arturo" AND nota=4.2;  
Query OK, 1 row affected (0.01 sec)  
Rows matched: 1  Changed: 1  Warnings: 0  
  
MariaDB [P19]> UPDATE Notas SET nota=3 WHERE nombre="Olivia" AND nota=8.8;  
Query OK, 0 rows affected (0.00 sec)  
Rows matched: 1  Changed: 0  Warnings: 0  
  
MariaDB [P19]> UPDATE Notas SET nota=4 WHERE nombre="Braulio" AND nota=10;  
Query OK, 0 rows affected (0.00 sec)  
Rows matched: 1  Changed: 0  Warnings: 0
```

Explica que sucede y que contiene la tabla NOTAS.

Suma a la nota dos puntos si la nueva nota es mayor a la vieja.

4.- Desde el PhpMyAdmin.**Creamos tres tablas.**

```
CREATE TABLE prueba1(a1 INT);
CREATE TABLE prueba2(a2 INT);
CREATE TABLE prueba3(a3 INT);
```

<input type="checkbox"/> prueba1	★	Examinar	Estructura	Buscar	Insertar	Vaciar	Eliminar	0	InnoDB	latin1_swedish_ci	16 KB
<input type="checkbox"/> prueba2	★	Examinar	Estructura	Buscar	Insertar	Vaciar	Eliminar	0	InnoDB	latin1_swedish_ci	16 KB
<input type="checkbox"/> prueba3	★	Examinar	Estructura	Buscar	Insertar	Vaciar	Eliminar	0	InnoDB	latin1_swedish_ci	16 KB

Y ahora creamos un Trigger.

✓ Disparador 'Disp_4' creado.

```
CREATE TRIGGER `Disp_4` AFTER INSERT ON `prueba1` FOR EACH ROW INSERT INTO prueba2 SET a2=NEW.a1;
```

5.- Hacemos uno de más de una línea.

Detalles

Nombre del disparador	Disp_5
Tabla	prueba1
Tiempo	AFTER
Evento	INSERT
Definición	<pre>1 BEGIN 2 INSERT INTO prueba2 SET a2=NEW.a1; 3 INSERT INTO prueba3 SET a3=NEW.a1; 4 END</pre>

No puede haber dos triggers en la misma tabla para el mismo evento.
Por eso da error.

MySQL ha dicho: #1235 - Esta versión de MariaDB no soporta todavía 'multiple triggers with the same action time and event for one table'

6.- Modifica el trigger anterior para que en la tabla prueba2 inserte un valor 10 unidades mayor que el insertado en prueba1 y en prueba3 el triple del valor insertado en prueba1.

✓ Se modificó el disparador 'Disp_4'.

```
DROP TRIGGER IF EXISTS `Disp_4`;CREATE DEFINER='root'@'localhost' TRIGGER `Disp_4` AFTER INSERT ON `prueba1` FOR EACH ROW BEGIN INSERT INTO prueba2 SET a2=NEW.a1*10; INSERT INTO prueba3 SET a3=NEW.a1*3; END
```