

Manual sobre el uso de disparadores

El trigger o disparador MySQL es un objeto de la base de datos que está asociado con tablas. El trigger se puede ejecutar cuando se ejecuta una de las siguientes sentencias MySQL en la tabla: INSERT, UPDATE y DELETE. Se puede invocar antes o después del evento.

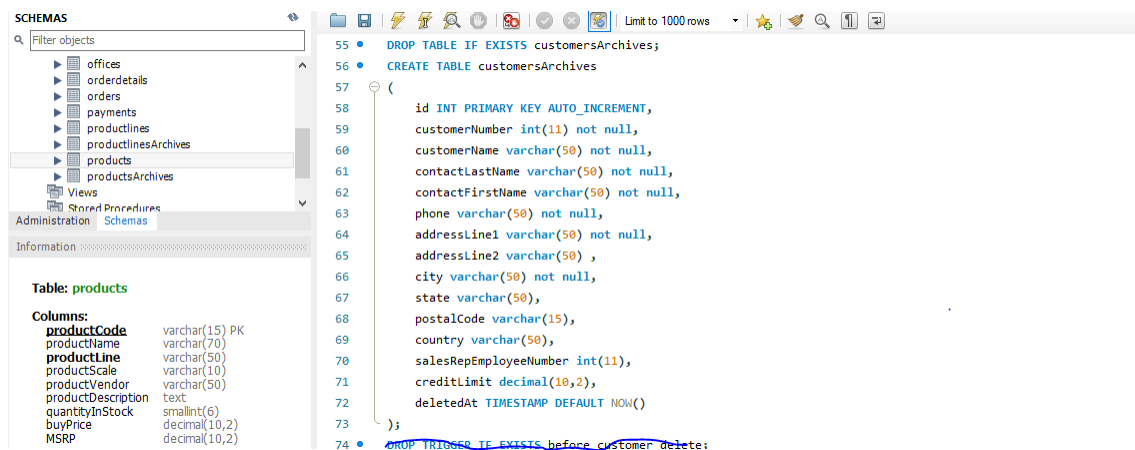
Paso 1: Antes que nada vamos a modificar la base de datos classicmodels le agregamos la sentencia ON UPDATE CASCADE-ON DELETE CASCADE en cada una de las tablas que tengan referencias esto nos ayudara a sincronizar las tablas para que si hacemos un cambio o si eliminamos un dato en una tabla este también se modifique en todas las tablas que se relacionan y que contienen el dato en todas las foreign key:

```
CREATE TABLE `employees` (
  `employeeNumber` int(11) NOT NULL,
  `lastName` varchar(50) NOT NULL,
  `firstName` varchar(50) NOT NULL,
  `extension` varchar(10) NOT NULL,
  `email` varchar(100) NOT NULL,
  `officeCode` varchar(10) NOT NULL,
  `reportsTo` int(11) DEFAULT NULL,
  `jobTitle` varchar(50) NOT NULL,
  PRIMARY KEY (`employeeNumber`),
  KEY `reportsTo` (`reportsTo`),
  KEY `officeCode` (`officeCode`),
  CONSTRAINT `employees_ibfk_2` FOREIGN KEY (`officeCode`) REFERENCES `offices`
  (`officeCode`)
  ON UPDATE CASCADE
  ON DELETE CASCADE,
  CONSTRAINT `employees_ibfk_1` FOREIGN KEY (`reportsTo`) REFERENCES `employees`
  (`employeeNumber`)
  ON UPDATE CASCADE
  ON DELETE CASCADE
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
```

Paso 2: ejemplo 1

Generar un registro en caso de que se elimina algo de las siguientes tablas: productlines, product, customers:

Después de crear el trigger debemos crear una tabla de auditoria por cada una de las tablas que vamos a modificar esta tabla de auditoria debe contener los mismos campos que la tabla principal con el tipo de dato y el valor similar por ejemplo:



The screenshot shows the MySQL Workbench interface. On the left, the 'SCHEMAS' pane shows a list of tables including 'customersArchives'. The main editor displays the following SQL code:

```
55 • DROP TABLE IF EXISTS customersArchives;
56 • CREATE TABLE customersArchives
57 • (
58 •   id INT PRIMARY KEY AUTO_INCREMENT,
59 •   customerNumber int(11) not null,
60 •   customerName varchar(50) not null,
61 •   contactLastName varchar(50) not null,
62 •   contactFirstName varchar(50) not null,
63 •   phone varchar(50) not null,
64 •   addressLine1 varchar(50) not null,
65 •   addressLine2 varchar(50) ,
66 •   city varchar(50) not null,
67 •   state varchar(50),
68 •   postalCode varchar(15),
69 •   country varchar(50),
70 •   salesRepEmployeeNumber int(11),
71 •   creditLimit decimal(10,2),
72 •   deletedAt TIMESTAMP DEFAULT NOW()
73 • );
74 • DROP TRIGGER IF EXISTS before_customer_delete;
```

Below the SQL editor, the 'Table: products' structure is shown:

Columns:	
productCode	varchar(15) PK
productName	varchar(70)
productLine	varchar(50)
productScale	varchar(10)
productVendor	varchar(50)
productDescription	text
quantityInStock	smallint(6)
buyPrice	decimal(10,2)
MSRP	decimal(10,2)

```

DROP TRIGGER IF EXISTS before_customer_delete;
DELIMITER $$
CREATE TRIGGER before_customer_delete
    BEFORE DELETE
    ON customers
    FOR EACH ROW
> BEGIN
>     INSERT INTO customersArchives( customerNumber, customerName, contactLastName, contactFirstName, phone,
- addressLine1, addressLine2, city, state, postalCode, country, salesRepEmployeeNumber, creditLimit)
- VALUES (OLD.customerNumber, OLD.customerName, OLD.contactLastName, OLD.contactFirstName, OLD.phone,
- OLD.addressLine1, OLD.addressLine2, OLD.city, OLD.state, OLD.postalCode, OLD.country, OLD.salesRepEmployeeNumber, OLD.creditLimit);
- END$$
DELIMITER ;products

```

Antes

Ahora

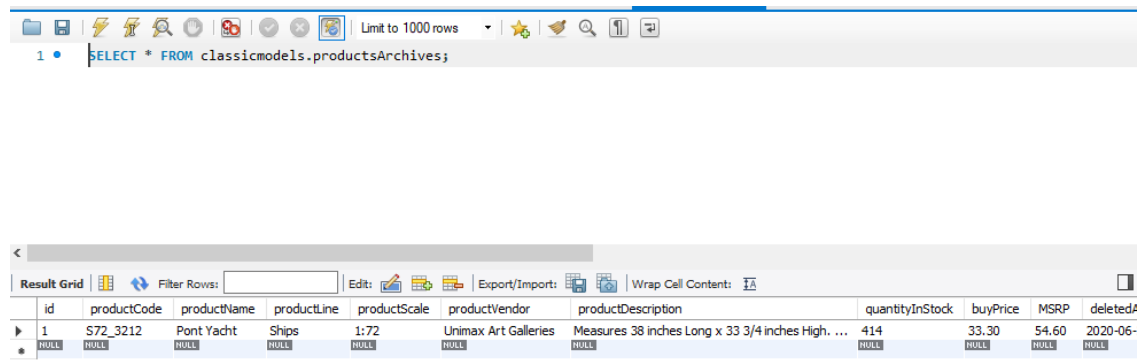
Limit to 1000 rows

1 • `SELECT * FROM classicmodels.products;`

Result Grid Filter Rows: Edit: Export/Import: Wrap Cell Content: `Ctrl+L`

productCode	productName	productLine	productScale	productVendor	productDescription	quantityInStock
5700_1938	The Mayflower	Ships	1:700	Studio M Art Models	Measures 31 1/2 inches Long x 25 1/2 inches High...	737
5700_2047	HMS Bounty	Ships	1:700	Unimax Art Galleries	Measures 30 inches Long x 27 1/2 inches High x...	3501
5700_2466	America West Airlines B757-200	Planes	1:700	Motor City Art Classics	Official logos and insignias. Working steering sy...	9653
5700_2610	The USS Constitution Ship	Ships	1:700	Red Start Diecast	All wood with canvas sails. Measures 31 1/2" Le...	7083
5700_2824	1982 Camaro Z28	Classic Cars	1:18	Carousel DieCast Legends	Features include opening and closing doors. Col...	6934
5700_2834	ATA: B757-300	Planes	1:700	Highway 66 Mini Classics	Exact replica with official logos and insignias and ...	7106
5700_3167	F/A 18 Hornet 1/72	Planes	1:72	Motor City Art Classics	10" Wingspan with retractable landing gears.Co...	551
5700_3505	The Titanic	Ships	1:700	Completed model measures 19 1/2 inches long, ...	Completed model measures 19 1/2 inches long, ...	1956
5700_3962	The Queen Mary	Ships	1:700	Welly Diecast Productions	Exact replica. Wood and Metal. Many extras inc...	5088
5700_4002	American Airlines: MD-11S	Planes	1:700	Second Gear Diecast	Polished finish. Exact replica with official logos an...	8820
572_1253	Boeing X-32A JSF	Planes	1:72	Motor City Art Classics	10" Wingspan with retractable landing gears.Co...	4857

El reporte ya esta en la tabla de auditoria de la tabla producto:



The screenshot shows a SQL query window with the following query:

```
1 • SELECT * FROM classicmodels.productsArchives;
```

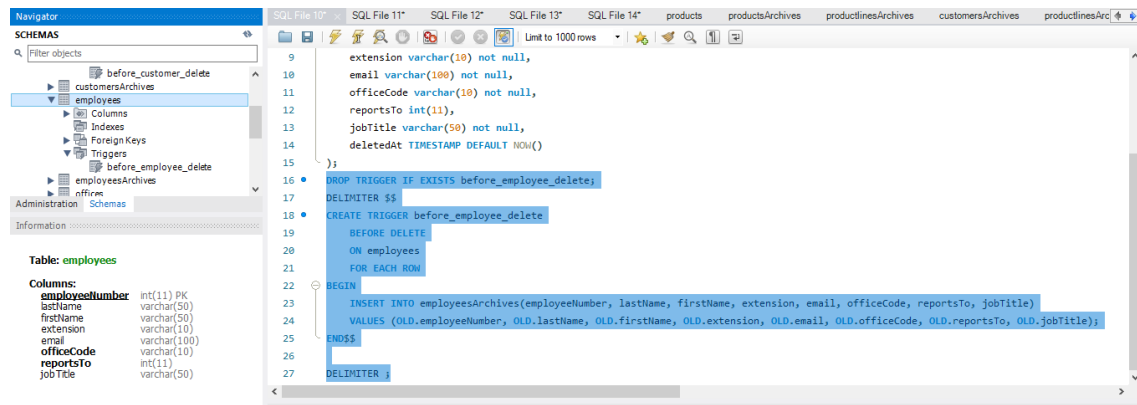
The results grid displays the following data:

id	productCode	productName	productLine	productScale	productVendor	productDescription	quantityInStock	buyPrice	MSRP	deletedAt
1	S72_3212	Pont Yacht	Ships	1:72	Unimax Art Galleries	Measures 38 inches Long x 33 3/4 inches High. ...	414	33.30	54.60	2020-06-2

Y eso seria en resumen el primer literal teniendo en cuenta que son tres tablas a las que aremos un reporte.

Paso 3: ejemplo 2

Registros que se elimina en la tabla employees tiene que pasar a una tabla exEmployees al borrarlos:



The screenshot shows a SQL script window with the following code:

```
9 extension varchar(10) not null,  
10 email varchar(100) not null,  
11 officeCode varchar(10) not null,  
12 reportsTo int(11),  
13 jobTitle varchar(50) not null,  
14 deletedAt TIMESTAMP DEFAULT NOW()  
15 }  
16 DROP TRIGGER IF EXISTS before_employee_delete;  
17 DELIMITER $$  
18 CREATE TRIGGER before_employee_delete  
19 BEFORE DELETE  
20 ON employees  
21 FOR EACH ROW  
22 BEGIN  
23 INSERT INTO employeesArchives(employeeNumber, lastName, firstName, extension, email, officeCode, reportsTo, jobTitle)  
24 VALUES (OLD.employeeNumber, OLD.lastName, OLD.firstName, OLD.extension, OLD.email, OLD.officeCode, OLD.reportsTo, OLD.jobTitle);  
25 END$$  
26  
27 DELIMITER ;
```

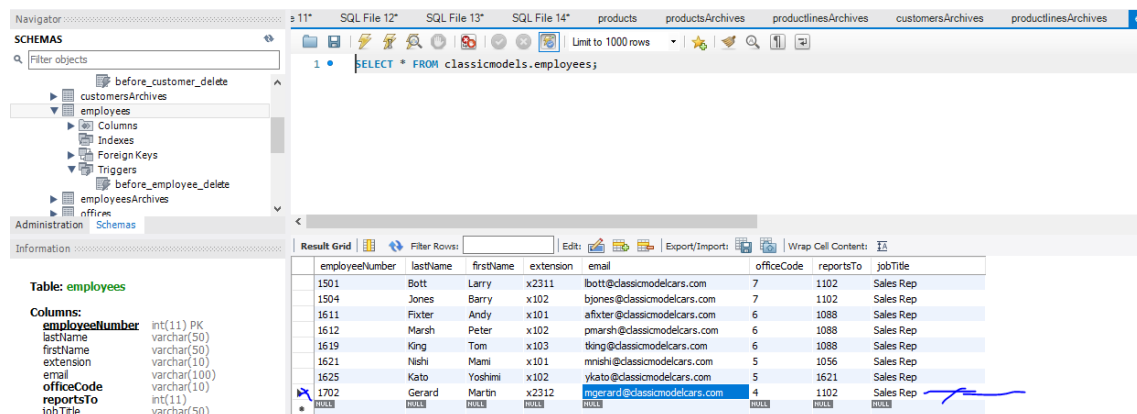
The left pane shows the schema for the 'employees' table:

Table: employees

Columns:

- employeeNumber int(11) PK
- lastName varchar(50)
- firstName varchar(50)
- extension varchar(10)
- email varchar(100)
- officeCode varchar(10)
- reportsTo int(11)
- jobTitle varchar(50)

Como se opserva en la imagen lo primero que hacemos es crear la tabla de auditoria donde se genera el reporte de cualquier empleado eliminado luego creamos el disparador para ese proceso y el resultado debe ser este:



The screenshot shows a SQL query window with the following query:

```
1 • SELECT * FROM classicmodels.employees;
```

The results grid displays the following data:

employeeNumber	lastName	firstName	extension	email	officeCode	reportsTo	jobTitle
1501	Bott	Larry	x2311	lbott@classicmodelcars.com	7	1102	Sales Rep
1504	Jones	Barry	x102	bjones@classicmodelcars.com	7	1102	Sales Rep
1611	Fixter	Andy	x101	afixter@classicmodelcars.com	6	1088	Sales Rep
1612	Marsh	Peter	x102	pmarsh@classicmodelcars.com	6	1088	Sales Rep
1619	King	Tom	x103	tking@classicmodelcars.com	6	1088	Sales Rep
1621	Nishi	Mami	x101	mnishi@classicmodelcars.com	5	1056	Sales Rep
1625	Kato	Yoshihito	x102	ykato@classicmodelcars.com	5	1621	Sales Rep
1702	Gerard	Martin	x2312	mgerard@classicmodelcars.com	4	1102	Sales Rep

Después:

```

1 • SELECT * FROM classicmodels.employees;
2
3 • DELETE FROM employees
4   WHERE employeeNumber='1702';

```

employeeNumber	lastName	firstName	extension	email	officeCode	reportsTo	jobTitle
1401	Castillo	Pamela	x2759	pcastillo@classicmodelcars.com	4	1102	Sales Rep
1501	Bott	Larry	x2311	lbott@classicmodelcars.com	7	1102	Sales Rep
1504	Jones	Barry	x102	bjones@classicmodelcars.com	7	1102	Sales Rep
1611	Fixter	Andy	x101	afixter@classicmodelcars.com	6	1088	Sales Rep
1612	Marsh	Peter	x102	pmarsh@classicmodelcars.com	6	1088	Sales Rep
1619	King	Tom	x103	tking@classicmodelcars.com	6	1088	Sales Rep
1621	Nishi	Mami	x101	mnishi@classicmodelcars.com	5	1056	Sales Rep
1625	Kato	Yoshimi	x102	ykato@classicmodelcars.com	5	1621	Sales Rep
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

En la tabla exmpleado tenemos ya un empleado que fue eliminado:

```

1 • SELECT * FROM classicmodels.employeesArchives;

```

	id	employeeNumber	lastName	firstName	extension	email	officeCode	reportsTo	jobTitle	deletedAt
▶	1	1702	Gerard	Martin	x2312	mgerard@classicmodelcars.com	4	1102	Sales Rep	2020-06-21 07:27:07
*		NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

Paso 4: ejemplo 3

Impedir que se puede borrar algo de las tablas orders y payments:

```

1 • use classicmodels;
2 • DROP TRIGGER IF EXISTS No_delete_order;
3   DELIMITER $$
4 • CREATE TRIGGER No_delete_order
5     BEFORE DELETE ON orders
6     FOR EACH ROW
7     BEGIN
8       If Old.orderNumber is not null then
9         SIGNAL SQLSTATE '45000'
10        SET MESSAGE_TEXT = 'Este registro no lo puedes eliminar';
11      End If;
12    END$$
13   DELIMITER ;

```

Resultado cuando intentamos borrar sale error y el mensaje que le mandamos por defecto.

```
15 • delete from orders
16 where orders.orderNumber=10104;
17
```

Output

#	Time	Action	Message
81	02:52:42	DROP TRIGGER IF EXISTS No_delete_payments	0 row(s) affected, 1 warning(s): 1360 Trigger does not exist
82	02:52:54	CREATE TRIGGER No_delete_payments BEFORE DELETE ON payments ...	0 row(s) affected
83	02:55:48	SELECT * FROM classicmodels.orders LIMIT 0, 1000	314 row(s) returned
84	02:56:34	Delete from orders where orders.orderNumber=10104	Error Code: 1644. No puede eliminar este registro de la tabla orders
85	03:00:37	Delete from orders where orders.orderNumber=10104	Error Code: 1644. No puede eliminar este registro de la tabla orders

```
DROP TRIGGER IF EXISTS No_delete_payments;
DELIMITER $$
CREATE TRIGGER No_delete_payments
    BEFORE DELETE ON payments
FOR EACH ROW
BEGIN
    If Old.checkNumber is not null then
        SIGNAL SQLSTATE '45000'
        SET MESSAGE_TEXT = 'No puede eliminar este registro de la tabla payments';
    End If;
END$$
DELIMITER ;
```

Resultado:

```
delete from payments
where payments.customerNumber=112;
```

Output

Time	Action	Message
03:03:27	DROP TRIGGER IF EXISTS No_delete_payments	0 row(s) affected
03:03:34	CREATE TRIGGER No_delete_payments BEFORE DELETE ON payments ...	0 row(s) affected
03:04:39	SELECT * FROM classicmodels.payments LIMIT 0, 1000	261 row(s) returned
03:05:05	Delete from payments where payments.customerNumber=112	Error Code: 1644. No puede eliminar este registro de la tabla payments
03:05:47	Delete from payments where payments.customerNumber=112	Error Code: 1644. No puede eliminar este registro de la tabla payments

Paso 4: ejemplo 3

Actualizaciones o inserts a la tabla products tiene que registrarse en una tabla logProducto:

The screenshot shows the SQL Server Enterprise Manager interface. On the left, the 'products' table structure is displayed with columns: productCode (varchar(15) PK), productName (varchar(70)), productLine (varchar(50)), productScale (varchar(10)), productVendor (varchar(50)), productDescription (text), quantityInStock (smallint(6)), buyPrice (decimal(10,2)), and MSRP (decimal(10,2)). The main window shows a SQL script with the following steps:

```
1 drop table if exists logProducts;
2 create table logProducts(
3     id int auto_increment,
4     productCode VARCHAR(15),
5     dateLog TIMESTAMP DEFAULT NOW(),
6     descripcion VARCHAR(255) NOT NULL,
7     PRIMARY KEY (id, productCode));
8
9 DROP TRIGGER IF EXISTS after_inserts_products;
10 DELIMITER $$
11 create trigger after_inserts_products
12 after insert on products
13 for each row
14 Begin
15     INSERT INTO logProducts(productCode, descripcion)
16     VALUES (new.productCode, CONCAT('Se ha insertado un nuevo producto: ', NEW.productName));
17 End$$
18 DELIMITER ;
19
20 INSERT INTO products(productCode, productName, productLine, productScale, productVendor, productDescription, quantityInStock, buyPrice, MSRP,
```

Para insertar los datos del nuevo producto:

The screenshot shows the SQL script execution results. The first part of the script is:

```
21 INSERT INTO products(productCode, productName, productLine, productScale, productVendor, productDescription, quantityInStock, buyPrice, MSRP)
22 VALUES('S72_3213', 'Nuevo producto', 'Planes', '1:72', 'Artefacta', 'Nuevo producto', 4554, 15.51, 51.51);
23 SELECT * FROM classicmodels.logProducts;
```

The second part of the screenshot shows the 'Result Grid' for the SELECT statement. It displays one row of data from the logProducts table:

id	productCode	dateLog	descripcion
1	S72_3213	2020-06-21 07:44:13	Se ha insertado un nuevo producto: Nuevo pro...

Y podemos observar en la tabla producto que tenemos en nuevo producto:

The screenshot shows the 'Result Grid' for the SELECT statement 'SELECT * FROM classicmodels.products;'. It displays a list of products, with the new product 'Nuevo producto' highlighted by a blue arrow. The product details are:

productCode	productName	productLine	productScale	productVendor	productDescription	quantityInStock	buyPrice	MSRP
S700_2834	ATA: B757-300	Planes	1:700	Highway 66 Mini Classics	Exact replica with official logos and insignias and ...	7106	59.33	118.65
S700_3167	F/A 18 Hornet 1/72	Planes	1:72	Motor City Art Classics	10" Wingspan with retractable landing gears.Co...	551	54.40	80.00
S700_3505	The Titanic	Ships	1:700	Carousal DieCast Legends	Completed model measures 19 1/2 inches long, ...	1956	51.09	100.17
S700_3962	The Queen Mary	Ships	1:700	Welly Diecast Productions	Exact replica. Wood and Metal. Many extras inc...	5088	53.63	99.31
S700_4002	American Airlines: MD-11S	Planes	1:700	Second Gear Diecast	Polished finish. Exact replica with official logos an...	8620	36.27	74.03
S72_1253	Boeing X-32A JSF	Planes	1:72	Motor City Art Classics	10" Wingspan with retractable landing gears.Co...	4857	32.77	49.66
S72_3213	Nuevo producto	Planes	1:72	Artefacta	Nuevo producto	4554	15.51	51.51