MySQL stored function

A stored function is a special kind stored program that returns a single value. You use stored functions to encapsulate common formulas or business rules that are reusable among SQL statements or stored programs.

## MySQL stored function syntax

The following illustrates the simplest syntax for creating a new stored function:

CREATE FUNCTION function\_name(param1,param2,…)

    RETURNS datatype

   [NOT] DETERMINISTIC

statements

**Now, we can call the stored function() in an SQL SELECT statement as follows:**

SELECT

    customerName, CustomerLevel(creditLimit)

FROM

    customers

ORDER BY customerName;

## Introduction to MySQL triggers

 A trigger can be defined to be invoked either before or after the data is changed by [INSERT](http://www.mysqltutorial.org/mysql-insert-statement.aspx), [UPDATE](http://www.mysqltutorial.org/mysql-update-data.aspx)or [DELETE](http://www.mysqltutorial.org/mysql-delete-statement.aspx)statement.

* BEFORE INSERT – activated before data is inserted into the table.
* AFTER INSERT – activated after data is inserted into the table.
* BEFORE UPDATE – activated before data in the table is updated.
* AFTER UPDATE – activated after data in the table is updated.
* BEFORE DELETE – activated before data is removed from the table.
* AFTER DELETE – activated after data is removed from the table.

## MySQL trigger syntax

In order to create a new trigger, you use the CREATE TRIGGER statement. The following illustrates the syntax of the  CREATE TRIGGER statement:

CREATE TRIGGER trigger\_name trigger\_time trigger\_event

ON table\_name

FOR EACH ROW

BEGIN

...

END;

* You put the **trigger name** after the CREATE TRIGGER statement. The trigger name should follow the naming convention [trigger time]\_[table name]\_[trigger event], for example before\_employees\_update*.*
* Trigger activation time can be BEFORE or AFTER. You must specify the **activation time** when you define a trigger. You use the BEFORE keyword if you want to process action prior to the change is made on the table and AFTER if you need to process action after the change is made.
* The **trigger event** can be **INSERT, UPDATE or  DELETE**. This event causes the trigger to be invoked. A trigger only can be invoked by one event. To define a trigger that is invoked by multiple events, you have to define multiple triggers, one for each event.
* A trigger must be associated with a specific table. Without a table trigger would not exist therefore you have to specify the table name after the ON keyword.
* You place the SQL statements between BEGIN and END block. This is where you define the logic for the trigger.

**OLD or NEW**

In an INSERT trigger, only NEW.col\_name can be used; there is no old row. In a DELETE trigger, only OLD.col\_name can be used; there is no new row. In an UPDATE trigger, you can use OLD.col\_name to refer to the columns of a row before it is updated and NEW.col\_name to refer to the columns of the row after it is updated.