

**UNIVERSIDAD TECNOLÓGICA DE**

**SAN LUIS RIO COLORADO**

**Triggers**

**MTRA. ARIADNA ALEJANDRA DERBEZ**

**ALUMNO: VICTOR MANUEL GALVAN COVARRUBIAS**

San Luis Rio Colorado, Sonora Septiembre, 2020

A **trigger** is a set of actions that are run automatically when a specified change operation (SQL INSERT, UPDATE, or DELETE statement) is performed on a specified table. Triggers are useful for tasks such as enforcing business rules, validating input data, and keeping an audit trail

**Data Manipulation Language Triggers or DML triggers.**

As the name suggests these are the triggers which depend on DML statements such as Update, Insert or Delete. They get fired either before or after them. Using DML trigger you can control the behavior of your DML statements. You can audit, check, replace or save values before they are changed. Automatic Increment of your Numeric primary key is one of the most frequent tasks of these types of triggers.

**Data Definition Language Triggers or DDL triggers.**

Again as the name suggests these are the type of triggers which are created over DDL statements such as CREATE or ALTER. They get fired either before or after the execution of your DDL statements. Using this type of trigger, you can monitor the behavior and force rules on your DDL statements.

**System or Database Event triggers.**

Third type of triggers is system or database triggers. These are the type of triggers which come into action when some system event occurs such as database log on or log off. You can use these triggers for auditing purposes. For example, keeping an eye on information of system access like say who connects with your database and when. Most of the time System or Database Event triggers work as Swiss Knife for DBAs and help them in increasing the security of the data.

**Instead-of Trigger.**

This is a type of trigger which enables you to stop and redirect the performance of a DML statement. Often this type of trigger helps you in managing the way you write to non-updatable views. You can also see the application of business rules by INSTEAD OF triggers where they insert, update or delete rows directly in tables that are defining updatable views.  Alternatively, sometimes the INSTEAD OF triggers are also seen inserting, updating or deleting rows in designated tables that are otherwise unrelated to the view.

**Compound triggers.**

These are multi-tasking triggers that act as both statement as well as row-level triggers when the data is inserted, updated or deleted from a table. You can capture information at four timing points using this trigger:

* before the firing statement;
* prior to the change of each row from the firing statement;
* post each row changes from the firing statement;
* after the firing statement.

All these types of triggers can be used to audit, check, save and replace the values. Even before they are changed right when there is a need to take action at the statement as well as at row event levels.

**CREATE** /\*OR REPLACE\*/ **TRIGGER** trigger\_name

{**BEFORE** /\***OR AFTER**\*/ Triggering\_event ON table\_name

[FOR EACH ROW]

[FOLLOWS another\_trigger\_name]

[ENABLE/DISABLE]

[WHEN condition]

**DECLARE**

  declaration statements

**BEGIN**

  executable statements

**EXCEPTION**

  exception-handling statements

**END**;

**ALTER TRIGGER** trigger\_name

**ON** /\*Table name or view name\*/

[ **WITH** <Options> ]

{ **FOR** | **AFTER** | **INSTEAD OF** }

{ [**INSERT**], [**UPDATE**] , [**DELETE**] }

**AS**

/\*Your code goes here\*/

**SELECT**

name,

is\_instead\_of\_trigger

**FROM**

sys.triggers

**WHERE**

type = 'TR';

**DROP TRIGGER** /\* IF EXISTS \*/ trigger\_name;