**TRIGGERS**

Un **trigger** es un procedimiento almacenado en la base de datos que se ejecuta automáticamente cada vez que ocurre un evento especial en ella. Por ejemplo, cuando se inserta una fila en una tabla específica o cuando ciertas columnas de la tabla se actualizan.

Por lo general, estos eventos que desencadenan los triggers son cambios en las tablas mediante operaciones de inserción, eliminación y actualización de datos (**insert**, **update** y **delete**).

Algunos de sus usos principales son:

* **Automatización de Tareas**: Imagina que cada vez que se registra un nuevo usuario en tu sistema, también debes actualizar una tabla de auditoría. Un trigger se encargará de esto automáticamente después de cada inserción en la tabla de usuarios.
* **Integridad de Datos**: Los triggers pueden aplicar reglas que protegen la integridad de tus datos. Por ejemplo, podrían impedir que se elimine un registro que aún está siendo referenciado en otra tabla.
* **Auditoría y Registro**: Los triggers son útiles para mantener un registro histórico de los datos. Por ejemplo, podrías querer saber cuándo se modificó por última vez un campo específico.
* **Acciones en cascada**: Los triggers pueden ejecutar acciones en cascada. Si borras un registro en una tabla, un trigger podría eliminar automáticamente los registros relacionados en otras tablas para mantener la integridad referencial.

Hay 2 clases de triggers:

* **Triggers DDL (Data Definition Language):** Esta clase de Triggers se activa en eventos que modifican la estructura de la base de datos (como crear, modificar o eliminar una tabla) o en ciertos eventos relacionados con el servidor, como cambios de seguridad o actualización de eventos estadísticos.
* **Triggers DML (Data Modification Language):** Esta es la clase más común de Triggers. En este caso, el evento de disparo es una declaración de modificación de datos; podría ser una declaración de inserción, actualización o eliminación en una tabla o vista. Hay 2 tipos:
  + **FOR o AFTER**: Estos tipos de Triggers se ejecutan después de completar la instrucción de disparo (inserción, actualización o eliminación).
  + **INSTEAD OF**: estos triggers se ejecutan en lugar de la instrucción de disparo, la reemplazan. Son de gran utilidad en los casos en los que es necesario tener integridad referencial entre bases de datos.

Cuando se define un trigger:

* Se especifica qué la activará **(insert, update, delete)**
* Se define una función que se ejecutara como respuesta
* Configurar si se ejecutará antes o después del evento que lo activa (**BEFORE**, **AFTER**)
* Configurar si debería activarse una vez por fila afectada (**FOR EACH ROW**) o una vez por cada sentencia SQL (**FOR EACH STATEMENT**)

Sintaxis:

create trigger [trigger\_name]

{before | after}

{insert | update | delete}

on [table\_name]

{for each row | for each statement}

[trigger\_body]

Pasó a paso:

1. Definir el Evento (cuándo se realizará) y la Tabla del trigger

CREATE TABLE Empleados (ID INT PRIMARY KEY, Nombre VARCHAR(50), Salario INT );

1. Definir la Acción que realizará el trigger

CREATE TABLE Auditoria\_Empleados (AuditoriaID INT PRIMARY KEY, ID\_Empleado INT, Accion VARCHAR(50), Fecha\_Accion DATETIME );

1. Escribir el Código del Trigger

CREATE TRIGGER InsertarAuditoria AFTER INSERT ON Empleados FOR EACH ROW BEGIN INSERT INTO Auditoria\_Empleados (ID\_Empleado, Accion, Fecha\_Accion) VALUES (NEW.ID, 'INSERT', NOW()); END;

1. Probar el Trigger

INSERT INTO Empleados (ID, Nombre, Salario) VALUES (1, 'Juan',

50000);

SELECT \* FROM Auditoria\_Empleados;

**PERMISOS**

En SQL, los permisos se refieren a los derechos generales que se otorgan a los usuarios o roles para realizar acciones sobre objetos dentro de una base de datos. Estos permisos determinan qué operaciones pueden realizar los usuarios, como consultar datos, modificar tablas, crear vistas, ejecutar procedimientos almacenados, entre otras acciones.

Los permisos son esenciales para la seguridad y la gestión de la base de datos, ya que permiten controlar quién tiene acceso a qué datos y qué tipo de modificaciones pueden realizar. Los principales tipos de permisos incluyen:

**SELECT, INSERT, UPDATE, DELETE, CREATE, ALTER, DROP**

**EXECUTE**: Permite ejecutar procedimientos almacenados y funciones definidas por el usuario.

**GRANT**: Permite otorgar permisos a otros usuarios.

**REVOKE**: Permite revocar permisos que se han otorgado previamente.

**CREATE USER**: Permite crear nuevos usuarios en la base de datos.

**ALTER USER**: Permite modificar las propiedades de un usuario existente (como la contraseña o los privilegios).

**DROP USER**: Permite eliminar un usuario de la base de da

**PRIVILEGIOS**

Establecer los privilegios de bases de datos determina la autorización que se debe tener para crear o acceder a las tablas de los almacenes de datos para las bases de datos de SQL Server. En SQL, los privilegios se refieren a los derechos y permisos que un usuario tiene sobre objetos específicos de la base de datos, como tablas, vistas, procedimientos almacenados, etc. Estos privilegios controlan qué acciones pueden realizar los usuarios en estos objetos y son fundamentales para la seguridad y la gestión de la base de datos. Estos privilegios se asignan a roles o usuarios específicos y se gestionan mediante comandos SQL como GRANT y REVOKE. Es importante administrar cuidadosamente los privilegios para garantizar la integridad y la seguridad de los datos en la base de datos.

Algunos de los privilegios comunes en SQL son:

* **SELECT**
* **INSERT**
* **UPDATE**
* **DELETE**
* **CREATE**
* **DROP**

Los privilegios anteriores son otorgados usando:

* **GRANT**: Permite otorgar privilegios a otros usuarios.
* **REVOKE**: Permite revocar privilegios otorgados previamente.

Ejemplos:

Para conceder los privilegios de **select, insert, update y delete**:

**GRANT** SELECT, INSERT, UPDATE, DELETE ON mydatabase.mi\_tabla TO 'usuario1'@'localhost';

Para conceder todos los privilegios:

**GRANT** **ALL PRIVILEGES** ON mydatabase.\* TO 'usuario1'@'localhost';

Para revokar el privilegio de insert:

**REVOKE** INSERT ON mydatabase.mi\_tabla FROM 'usuario1'@'localhost';

**Fuentes**

<https://www.aluracursos.com/blog/que-es-y-como-trigger-en-sql#:~:text=Un%20trigger%20es%20un%20bloque,ocurre%20un%20evento%20de%20trigger>.

<https://www.dongee.com/tutoriales/como-funciona-un-trigger-en-sql-un-tutorial-paso-a-paso/>

<https://www.geeksforgeeks.org/sql-trigger-student-database/>

<https://learn.microsoft.com/es-es/sql/relational-databases/security/permissions-database-engine?view=sql-server-ver16>