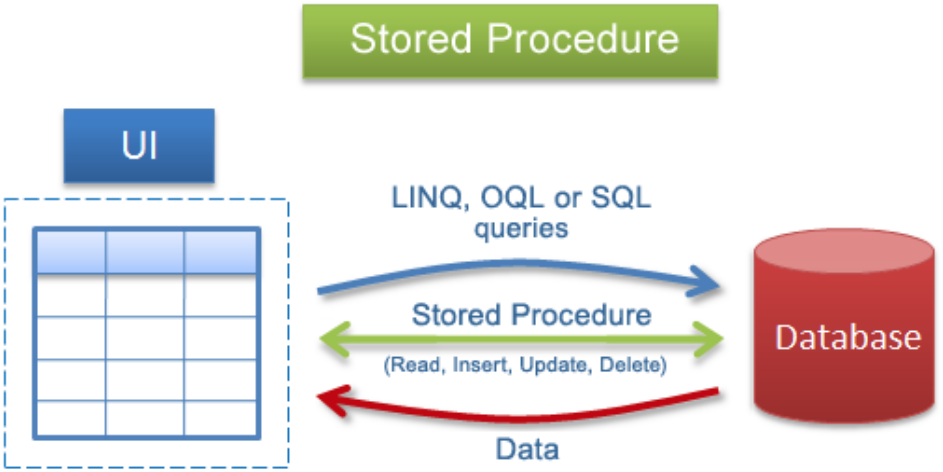

Unidad 6

Programación de Bases de Datos

Disparadores: Triggers

TRIGGERS -DISPARADORES

- Un ***trigger*** o disparador es un objeto con nombre en una base de datos que se asocia con una tabla, y se activa cuando ocurre un evento en particular para esa tabla.
- Un ***trigger*** es un tipo especial de procedimiento almacenado que se activa o ejecuta automáticamente cuando en una tabla ocurre un evento de tipo ***INSERT***, ***DELETE*** o ***UPDATE***. Es decir, los disparadores implementan una funcionalidad asociada a cualquier cambio en una tabla.
- Esto significa que invocaremos nuestros **Triggers** para ejecutar un **bloque de instrucciones** que proteja, restrinja o preparen la información de nuestras tablas, al momento de manipular nuestra información. Para crear triggers en MySQL necesitas los privilegios **SUPER** y **TRIGGER**.

Uso de disparadores

- Aunque el uso de los triggers es muy variado y depende mucho del tipo de aplicación o base de datos, podemos hacer la siguiente clasificación :
  - **Control de sesiones**: recoger en variables de sesión creadas por el usuario (@varible) un resumen del trabajo realizado durante esa sesión.
  - **Control de valores de entrada:** es muy recomendable controlar los valores insertados o actualizados en las tablas.
  - **Mantenimiento de campos derivados:** también llamados campos calculados. Estos campos son redundantes, pero en algunos casos aparecen en el diseño de una BD y debemos realizar un mantenimiento de los mismos (totalVentas, añosServicio, …)
  - **Estadísticas**. Inserciones sobre una tabla, …
  - **Registro y auditoría**. Podemos registrar quién, cuándo y qué operaciones realizan los diferentes usuarios sobre una BD, para llevar un mayor control en el sistema.

Sintaxis para la creación de Triggers

- **Creación de un Trigger**

```
CREATE [DEFINER={usuario|CURRENT_USER}]
TRIGGER nombre_del_trigger {BEFORE|AFTER} {UPDATE|INSERT|DELETE}
ON nombre_de_la_tabla
FOR EACH ROW
<bloque_de_instrucciones>
```

- **DEFINER**={usuario|CURRENT_USER}: Indica al **gestor de bases de datos** qué usuario tiene privilegios en su cuenta, para la invocación de los triggers cuando surjan los eventos **DML**. Por defecto esta característica tiene el valor CURRENT_USER que hace referencia al usuario actual que esta creando el Trigger.
- nombre_del_trigger: Indica el nombre de nuestro trigger. Existe una **nomenclatura** muy práctica para nombrar un trigger, la cual nos da mejor legibilidad en la administración de la base de datos. Primero ponemos el nombre de tabla, luego especificamos con la inicial de la operación DML y seguido usamos la inicial del momento de ejecución(AFTER o BEFORE).

Sintaxis para la creación de Triggers

- **Creación de un Trigger**

- Por ejemplo

```
-- BEFORE INSERT
clientes_BI_TRIGGER
```

- **BEFORE|AFTER**: Especifica si el Trigger se ejecuta antes o después del evento DML.
- **UPDATE|INSERT|DELETE:** sentencia que hace que se ejecute el Trigger.
- **ON** nombre_de_la_tabla: nombre de la tabla asociada.
- **FOR EACH ROW**: Establece que el Trigger se ejecute por cada fila en la tabla asociada.
- <bloque_de_instrucciones>: Define el bloque de sentencias que el Trigger ejecutará al ser invocado.

Sintaxis para la creación de Triggers

- **Creación de un Trigger**

- Por ejemplo

```
Delimiter $$
CREATE TRIGGER Empleado_BU_Trigger
BEFORE UPDATE ON Empleado
FOR EACH ROW
BEGIN
 if NEW.salario > 2000 then
   set NEW.salario=2000;
 end if;
END $$
Delimiter ;
```

Sintaxis para la creación de Triggers

IDENTIFICADORES NEW Y OLD EN TRIGGERS

- No puede haber dos disparadores en una misma tabla que correspondan al mismo momento y evento. Por ejemplo, no se pueden tener dos disparadores BEFORE UPDATE.
- El disparador no puede referirse a tablas directamente por su nombre, incluyendo la misma tabla a la que está asociado. Si queremos relacionar el trigger con columnas especificas de una tabla debemos usar los identificadores OLD y NEW.
- OLD indica el valor antiguo de la columna y NEW el valor nuevo que pudiese tomar. Por ejemplo: **OLD**.Salario ó **NEW**.Salario.
  - ***OLD.nombre_col*** hace referencia a una columna de una fila existente, antes de ser actualizada o borrada. No se puede utilizar en un disparador para INSERT, ya que no existe un valor anterior para la columna.
  - ***NEW.nombre_col*** hace referencia a una columna en una nueva fila a punto de ser insertada, o en una fila existente luego de que fue actualizada. No puede utilizarse en un disparador para DELETE.
- Si usamos la sentencia UPDATE podremos referirnos a un valor OLD y NEW, ya que modificaremos registros existentes por nuevos valores. En cambio si usamos INSERT solo usaremos NEW, ya que su naturaleza es únicamente de insertar nuevos valores a las columnas. Y si usamos DELETE usaremos OLD debido a que borraremos valores que existen con anterioridad.

Sintaxis para la creación de Triggers

TRIGGERS BEFORE Y AFTER

- Estas clausulas indican si el Trigger se ejecuta **antes** o **después** del evento DML. Hay ciertos eventos que NO son compatibles con estas sentencias.
- Por ejemplo, si tuvieras un **Trigger AFTER** que se ejecuta en una sentencia UPDATE, sería ilógico editar valores nuevos NEW, sabiendo que el evento ya ocurrió. Igual sucedería con la sentencia INSERT, el Trigger tampoco podría referenciar valores NEW, ya que los valores que en algún momento fueron NEW, han pasado a ser OLD.

```
1 delimiter $$
2 CREATE TRIGGER Empleado_BU_Trigger
3 AFTER UPDATE ON empleado
4 FOR EACH ROW
5 BEGIN
6  if NEW.salario > 2000 then
7    set NEW.salario=2000;
8  end if;
```

!	Desciption
!	Updating of NEW row is not allowed in after trigger

Eliminar y consultar los triggers

- **Eliminar un trigger**.

  DROP TRIGGER *[IF EXISTS]* *nombre_trigger;*

- **Consulta de triggers**.

  SHOW TRIGGERS *[FROM | IN base_datos]*

  [LIKE 'expres' | WHERE expres];

Esta instrucción muestra los triggers creados y cuya información está almacenada en el catálogo del sistema en una tabla llamada TRIGGERS.

Ejemplos de triggers

- Crear un trigger que compruebe que cualquier actualización del salario de un empleado, siempre que sea superior a 2000 €, pondrá esa cantidad como sueldo máximo.
- Crear un trigger que almacene en una tabla la información referente al usuario y fecha en que se realiza una inserción en la tabla de empleados.
- Crear un trigger que no permite que el salario de un empleado sea incrementado más de un 20 % de su salario actual.
- Crear una tabla, llamada despidos, con los campos Cod, nombre, empleo, Fecha, usuario. Crear un trigger que inserte en la tabla anterior los datos Nemp, nombre, puesto, cada vez que se borre un empleado de la Base de Datos. En los campos fecha y usuario se almacenará la fecha de la baja y el usuario que eliminó el registro.