**TITULO: TALLER DE PW - SQL**

**TEMA: MANEJO DE PROCEDIMIENTOS ALMACENADOS Y TRIGGERS EN SQL**

**OBJETIVO:** El estudiante crea procedimientos almacenados y triggers que actúan sobre la BBDD actual

**Nombre estudiante \_\_\_\_\_\_ Curso PW\_\_\_\_\_\_\_ Fecha Entrega \_\_\_\_\_\_\_\_\_\_\_\_**

TRIGGERS

Es un objeto que se crea con la sentencia CREATE TRIGGER y tiene que estar asociado a una tabla. Mediante esta regla, se ejecutan una serie de instrucciones cuando se producen ciertos eventos sobre una tabla, un trigger se activa cuando ocurre un evento de inserción, actualización o borrado, sobre la tabla a la que está asociado.

Con la ayuda de los triggers en MySQL podemos conseguir automatizar acciones sin necesidad de desarrollar lógica en otros lenguajes.

Los tiempos del trigger son:

**BEFORE**: Antes de un evento.

**AFTER**: Después de un evento.

Los eventos que pueden ocurrir sobre la tabla son:

**INSERT**: El trigger se activa cuando se inserta una nueva fila sobre la tabla asociada.

**UPDATE**: El trigger se activa cuando se actualiza una fila sobre la tabla asociada.

**DELETE**: El trigger se activa cuando se elimina una fila sobre la tabla asociada.

Diagrama

Descripción generada automáticamente

**SINTAXIS:**

DROP TRIGGER IF EXISTE NOMBRE\_DISPARADOR

DELIMITER $$

CREATE TRIGGER NOMBR E\_DISPARADOR

[TIEMPO] [EVENTO] \*\*\*\* MOMENTO REAL Y TIPO DE EVENTO

ON NOMBRE\_TABLA ON -- EN LA TABLA

FOR EACH ROW FOR EACH ROW -- POR CADA FILA

BEGIN BEGIN -- INICIO DE OPERACIONES

OPERACIONES QUE REALIZAR O AFECTA A LA TABLA

END $$

DELIMITER ;

**VARIABLES NEW Y OLD**

Para poder manejar estas instrucciones condicionales tenemos las variables OLD y NEW.

**Variable NEW**

NEW almacena el valor que aporta la consulta a la base de datos. Con esta variable podemos acceder a los datos introducidos. Con NEW.nombre\_columna se almacenará la información con el nuevo valor que tendrá ese registro modificado (desde un UPDATE o INSERT) en la tabla. Los trigger relacionados con DELETE no tendrán disponible la variable NEW.

**EJEMPLO VARIABLE NEW**

En un disparador AFTER INSERT, puede acceder a los valores NEW. Además, no puede acceder a los OLD valores porque no hay OLD en activadores INSERT.

DELIMITER $$

CREATE TRIGGER trigger\_producto\_historico

**AFTER INSERT** ON producto

FOR EACH ROW

BEGIN

INSERT INTO producto\_historico(id, descripcion, precio)

VALUES (NEW.id, NEW.descripcion, NEW.precio, CURDATE());

END$$

DELIMITER ;

**Variable OLD**

OLD a diferencia de NEW, almacena el valor de las columnas que van a ser borradas o eliminadas. Al igual que pasa con NEW, OLD no está disponible en todas las instrucciones, por lo general estará presenta en los tiempos BEFORE UPDATE o BEFORE DELETE.

**EJEMPLO VARIABLE OLD**

En un disparador BEFORE DELETE, puede acceder a la fila OLD. Además, no hay ninguna fila NEW en BEFORE DELETE.

DELIMITER $$

CREATE TRIGGER reversar\_ventas\_inv

**BEFORE DELETE**

ON ventas FOR EACH ROW

BEGIN

UPDATE productos SET saldo\_pro =productos.saldo\_pro + OLD.cantidad

WHERE productos.cod\_pro=OLD.cod\_pro;

END$$

DELIMITER ;

**EJEMPLO VARIABLE OLD**

En un disparador BEFORE UPDATE, puede acceder a los valores OLD, puede actualizar los valores NEW pero no puede actualizarlos OLD.

DELIMITER $$

CREATE TRIGGER clave\_usuarios

**BEFORE UPDATE**

ON usuarios FOR EACH ROW

BEGIN

INSERT INTO claves\_old (nombre, clave) VALUES (OLD.nombre,

OLD.clave);

END$$

DELIMITER ;

**A TENER EN CUENTA**

Tenga en cuenta que en un disparador BEFORE INSERT, puede acceder y cambiar los valores NEW. Sin embargo, no puede acceder a los valores OLD porque obviamente los valores OLD no existen.

DELIMITER $$

CREATE TRIGGER check\_nota

**BEFORE INSERT**

ON alumnos FOR EACH ROW

BEGIN

IF NEW.nota <0 THEN

SET NEW.nota=0;

ELSEIF NEW.nota>10 THEN

SET NEW.nota=10;

END IF;

END$$

DELIMITER ;

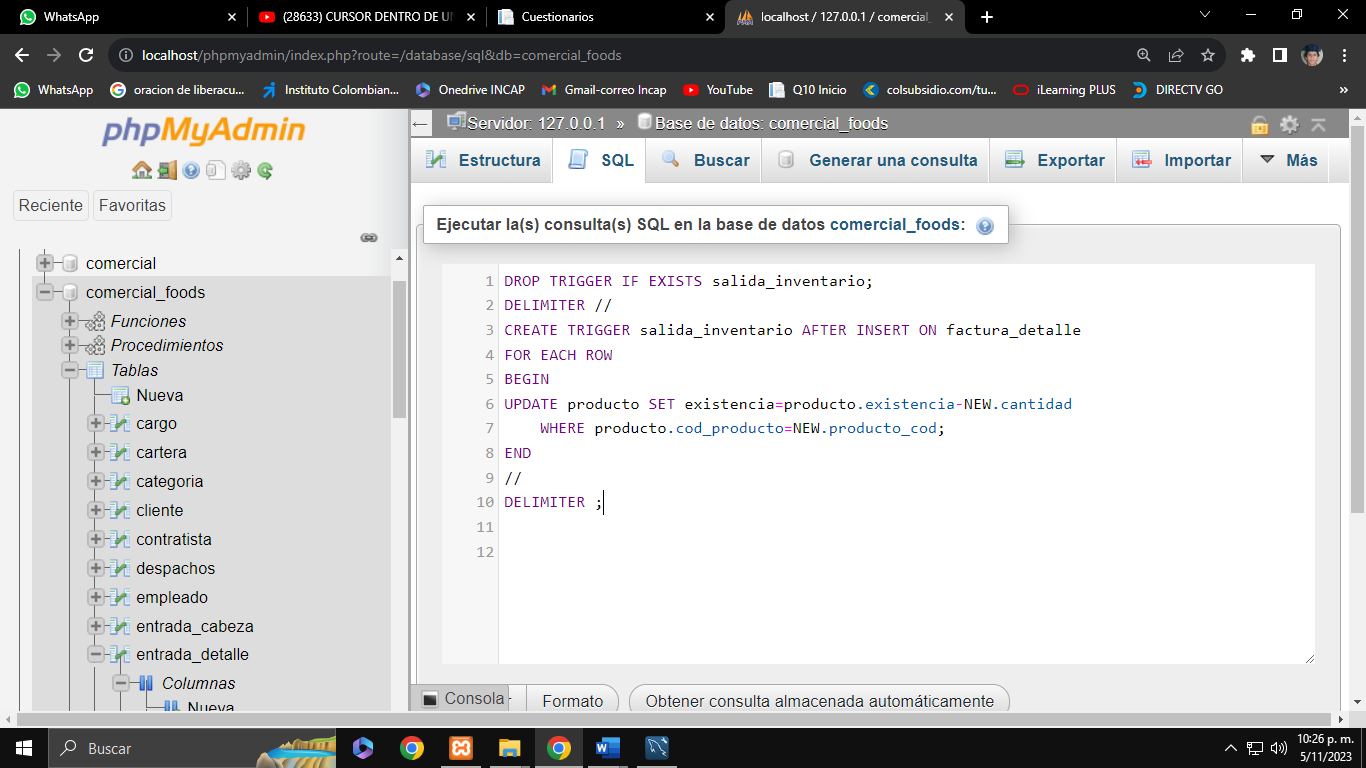
Tenga en cuenta que, en los disparadores BEFORE puede acceder a los valores antiguos y nuevos de las columnas mediante los modificadores OLD y NEW.

En un disparador AFTER DELETE, puede acceder a la fila OLD.

Tenga en cuenta que no hay ninguna fila NEW en el AFTER DELETE.

**EJEMPLO**

Vamos a crear un triggers para actualizar las salidas de inventarios en cantidades al momento instantáneo que se efectúa una venta o salida varia



ATENTOS: Dependiendo de la cantidad de códigos de factura registrados en la tabla Factura\_cabeza y factura\_detalle colocamos esta instrucción Insert into a continuación:

**TABLA FACTURA CABEZA**

INSERT INTO factura\_cabeza (fecha\_factura, cliente\_cod, empleado\_cod, forma\_pago, tipomov\_cod) VALUES

(‘2023-10-25’, 8, 5,’efectivo’,1);

**TABLA FACTURA DETALLE**

INSERT INTO factura\_detalle (factura\_cod, producto\_cod, cantidad, valor\_venta, descuento) VALUES

(42, 1, 50, 3000, 0.1);

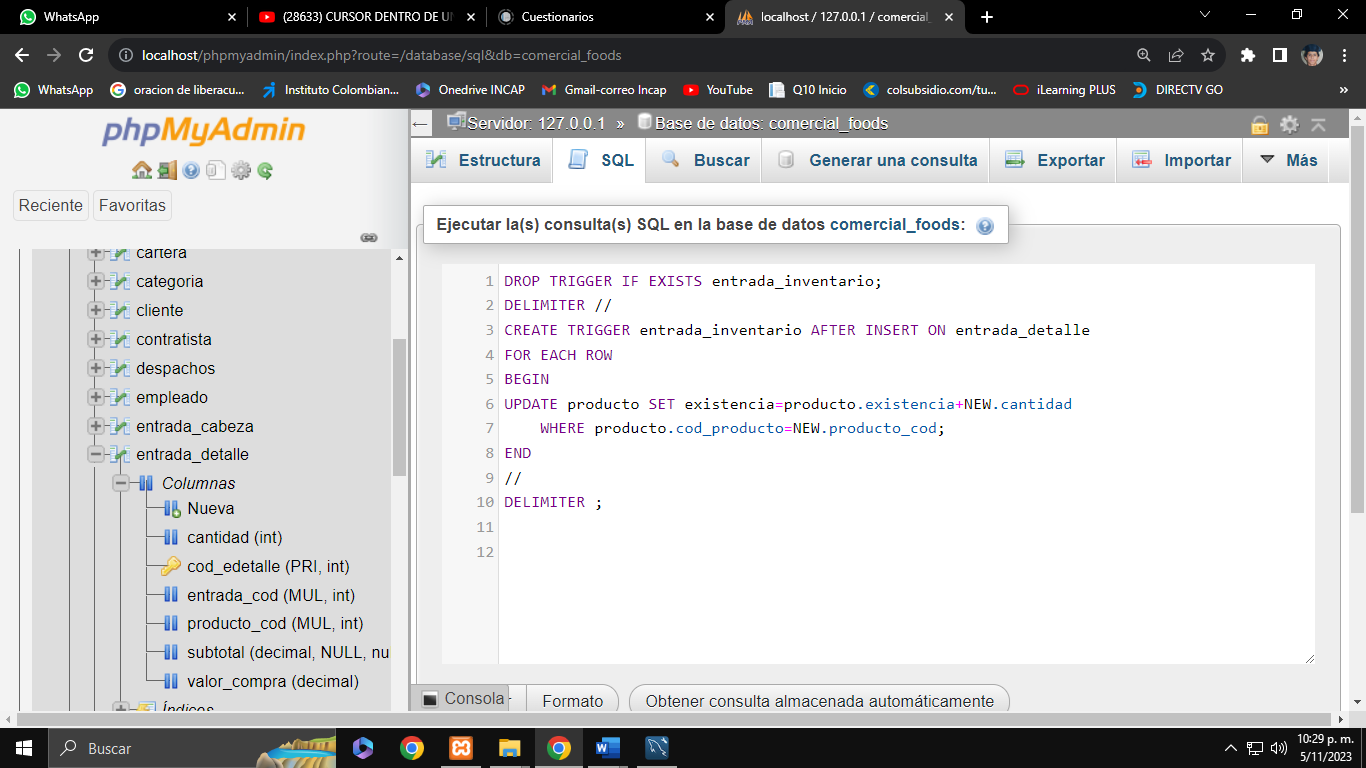
(42, 2, 50, 8000, 0.1);

(42, 3, 50, 3800, 0.1);

(42, 4, 50, 3000, 0.1);

**EJEMPLO**

Vamos a crear un triggers para actualizar las entradas de inventarios en cantidades al momento instantáneo que se efectúa una compra o devolución de venta.



ATENTOS: Dependiendo de la cantidad de códigos de factura registrados en la tabla Entrada\_cabeza y Entrada\_detalle colocamos esta instrucción Insert into a continuación:

**TABLA ENTRADA CABEZA**

INSERT INTO entrada\_cabeza (fecha\_entrada, proveedor\_cod, empleado\_cod, forma\_pago, tipomov\_cod) VALUES

('2023-08-12', 2, 1,'efectivo',2);

**TABLA ENTRADA DETALLE**

INSERT INTO entrada\_detalle (entrada\_cod, producto\_cod, cantidad, valor\_compra) VALUES

(21, 1, 100, 2000),

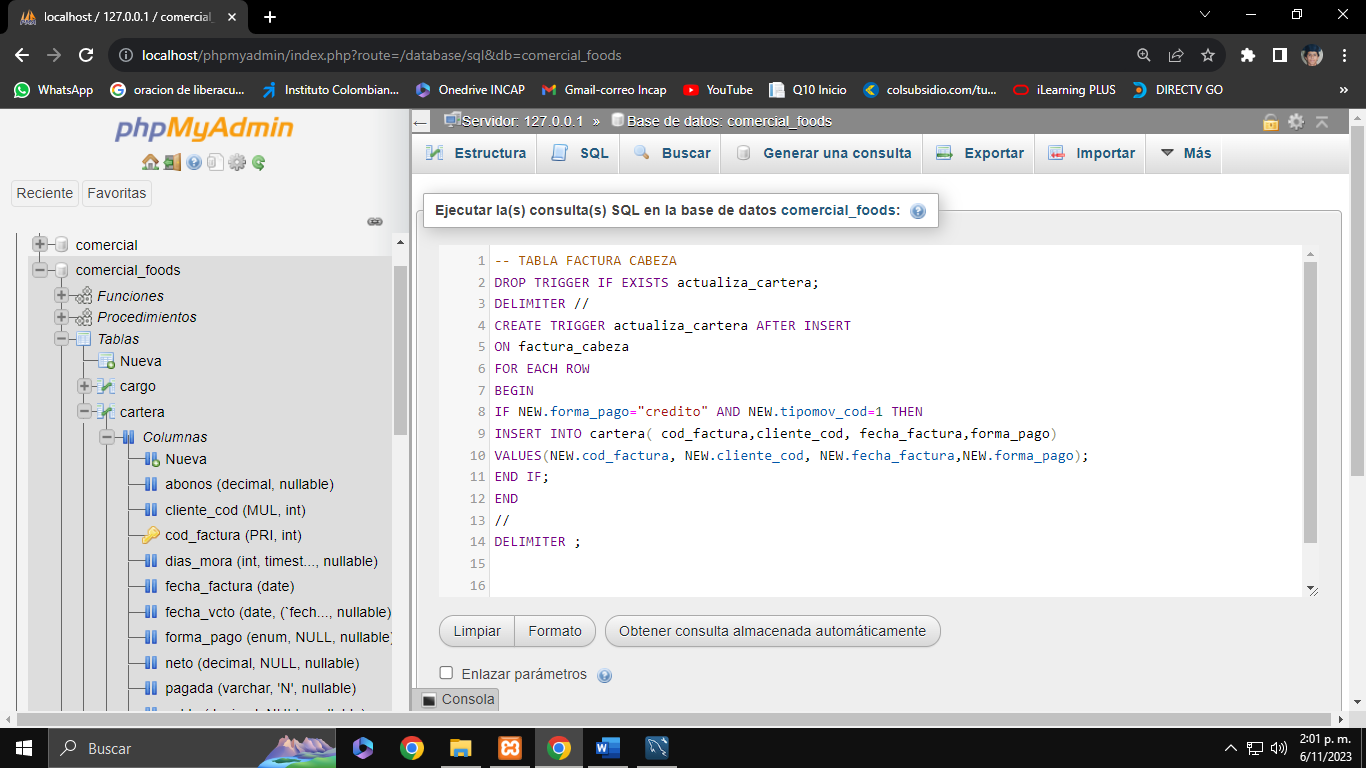
(21, 2, 100, 7000),

(21, 14, 100, 9000),

(21, 16, 100, 8000);

**EJEMPLO**

Vamos a crear un triggers para actualizar los registros de ingresos de las facturas cuya forma de pago es CREDITO al momento instantáneo que se efectúa una venta.



**-- TABLA FACTURA CABEZA**

INSERT INTO factura\_cabeza (fecha\_factura, cliente\_cod, empleado\_cod, forma\_pago, tipomov\_cod) VALUES

('2023-10-25', 9, 6,'credito',1);

**ATENTO ESTUDIANTE:**

Como podemos observar después de aplicar este trigger actualiza\_cartera nuestra columna de NETO no se ha actualizado por pertenecer a otra tabla factura\_detalle, se debe de crear un procedimiento almacenado con cursores además se debe hacer una consulta de resumen para ello.

**TAREA**: Haz el procedimiento almacenado faltante que permita actualizar el valor neto de las facturas de crédito recientemente ingresadas en esta tabla. (Vistas en la sesión de clase de la semana pasada)

**A TENER EN CUENTA:**

Para que los procedimientos sean automatizados en la BBDD se deben de implementar los objetos EVENTOS en SQL

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente



Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

Captura de pantalla de computadora

Descripción generada automáticamente

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

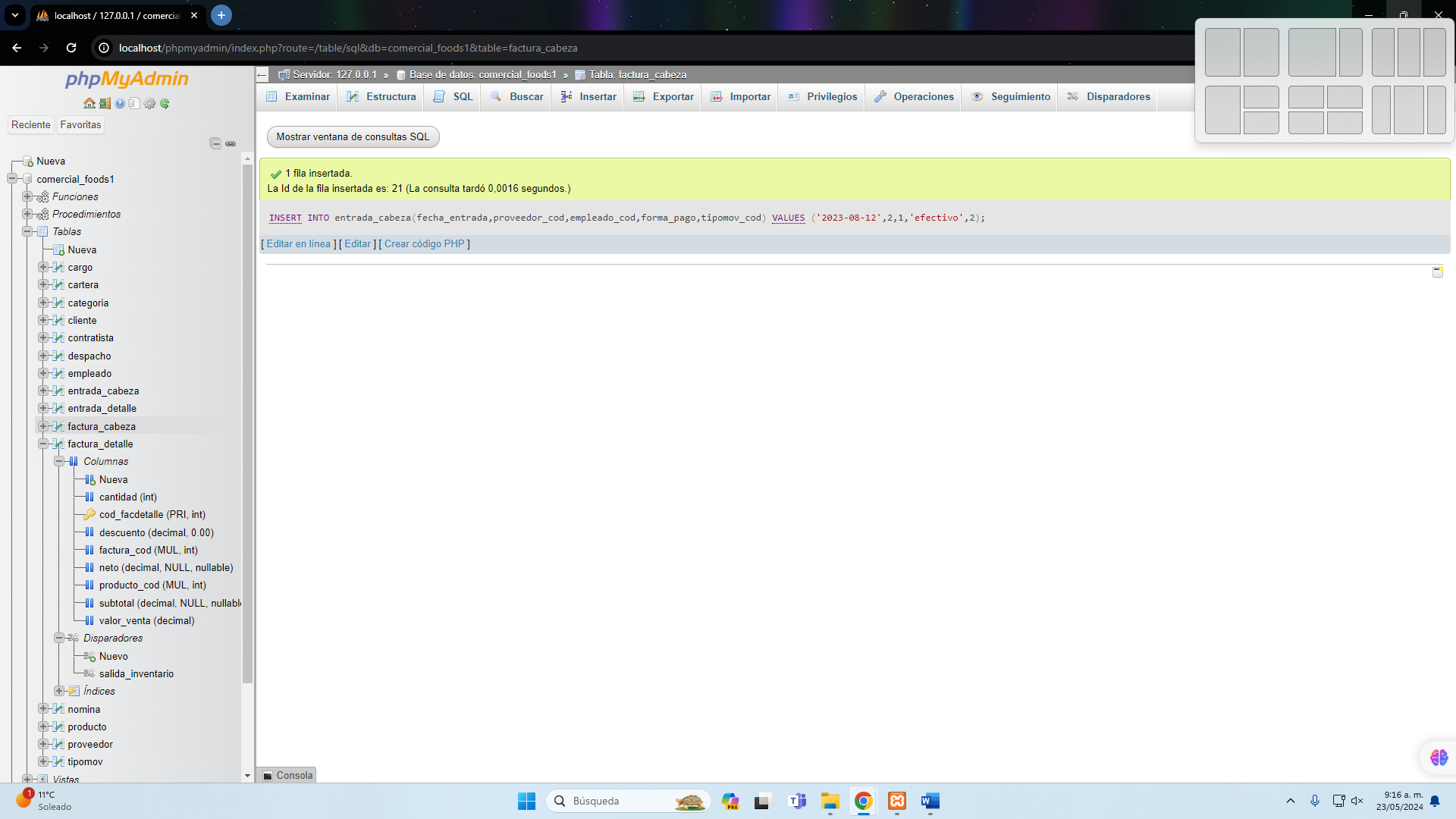
Descripción generada automáticamente

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

Una captura de pantalla de una computadora

Descripción generada automáticamente



Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente