**PROCESO DE GESTIÓN DE FORMACIÓN PROFESIONAL INTEGRAL**

**ACTIVIDAD BESE DE DATOS**

**ANALISIS Y DESARROLLO DE SOFTWARE 228118**

**JERSEY OTAVO**

**FICHA 2672883**

**MONICA PENAGOS MARTINEZ**

**BASE DE DATOS**

**BOGOTA 2023**

**MODULO COMPRAS**

1.Este TRIGGER se ejecutará automáticamente después de que se inserte una nueva compra en la tabla "compras" y

extraerá información sobre la compra, como el nombre del proveedor, la fecha de compra y el total de la compra.

Luego, creará un mensaje de notificación y lo almacenará en la tabla "notificaciones" junto con la fecha de creación.

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

DELIMITER //

CREATE TRIGGER AfterInsertCompra

AFTER INSERT ON compras FOR EACH ROW

BEGIN

DECLARE proveedor\_nombre VARCHAR(255);

DECLARE fecha\_compra DATE;

DECLARE total\_compra DECIMAL(10, 2);

DECLARE mensaje\_notificacion TEXT;

SELECT nombre INTO proveedor\_nombre FROM proveedores WHERE id\_proveedor = NEW.proveedor;

SELECT fecha\_entrada, total INTO fecha\_compra, total\_compra FROM compras WHERE id\_compra = NEW.id\_compra;

SET mensaje\_notificacion = CONCAT('Nueva compra realizada a proveedor: ', proveedor\_nombre,

'. Fecha de compra: ', fecha\_compra, '. Total de la compra: $', total\_compra);

INSERT INTO notificaciones (mensaje, fecha\_creacion) VALUES (mensaje\_notificacion, NOW());

END;

//

DELIMITER ;

SELECT \* FROM notificaciones;

2.El procedimiento almacenado "ProcesarCompras" inicia declarando variables para rastrear el ID de la última compra

y el total acumulado. Luego, utiliza un bucle WHILE para recorrer todas las compras, obteniendo el total de cada compra

y registrando tareas en la tabla "notificaciones" según el total. Finalmente, suma todos los totales de compra y

registra la suma total en la tabla "notificaciones". Este procedimiento proporciona una manera automatizada

de procesar compras y generar notificaciones basadas en criterios de total de compra.

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

DELIMITER //

CREATE PROCEDURE ProcesarCompras()

BEGIN

DECLARE last\_purchase\_id INT;

DECLARE total\_sum DECIMAL(10, 2);

DECLARE current\_purchase\_id INT;

DECLARE current\_total DECIMAL(10, 2);

SELECT MAX(id\_compra) INTO last\_purchase\_id FROM compras;

SET total\_sum = 0;

SET current\_purchase\_id = 1; -- Supongo que los IDs de compra comienzan desde 1

WHILE current\_purchase\_id <= last\_purchase\_id DO

SELECT total INTO current\_total FROM compras WHERE id\_compra = current\_purchase\_id;

IF current\_total > 1000 THEN

INSERT INTO notificaciones (mensaje) VALUES ('Compra con total mayor a 1000 registrada');

ELSE

INSERT INTO notificaciones (mensaje) VALUES ('Compra con total menor o igual a 1000 registrada');

END IF;

SET total\_sum = total\_sum + current\_total;

SET current\_purchase\_id = current\_purchase\_id + 1;

END WHILE;

INSERT INTO notificaciones (mensaje) VALUES (CONCAT('La suma total de todas las compras es ', total\_sum));

END //

DELIMITER ;

CALL ProcesarCompras();

3.En este procedimiento almacenado, se toma el ID de la compra y el total de la compra como parámetros de entrada.

Luego, se utiliza una expresión CASE para determinar la acción a realizar según el total de la compra.

La acción se almacena en la variable "accion", y luego se registra en la tabla "notificaciones" junto con la fecha

actual.

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

DELIMITER //

CREATE PROCEDURE ProCompras(IN compra\_id INT, IN total\_compra DECIMAL(10, 2))

BEGIN

DECLARE accion VARCHAR(255);

CASE

WHEN total\_compra > 1000 THEN

SET accion = 'Acción especial';

WHEN total\_compra BETWEEN 500 AND 1000 THEN

SET accion = 'Acción para compra mediana';

ELSE

SET accion = 'Acción estándar para compra pequeña';

END CASE;

INSERT INTO notificaciones (fecha\_notificacion, mensaje)

VALUES (NOW(), accion);

END//

DELIMITER ;

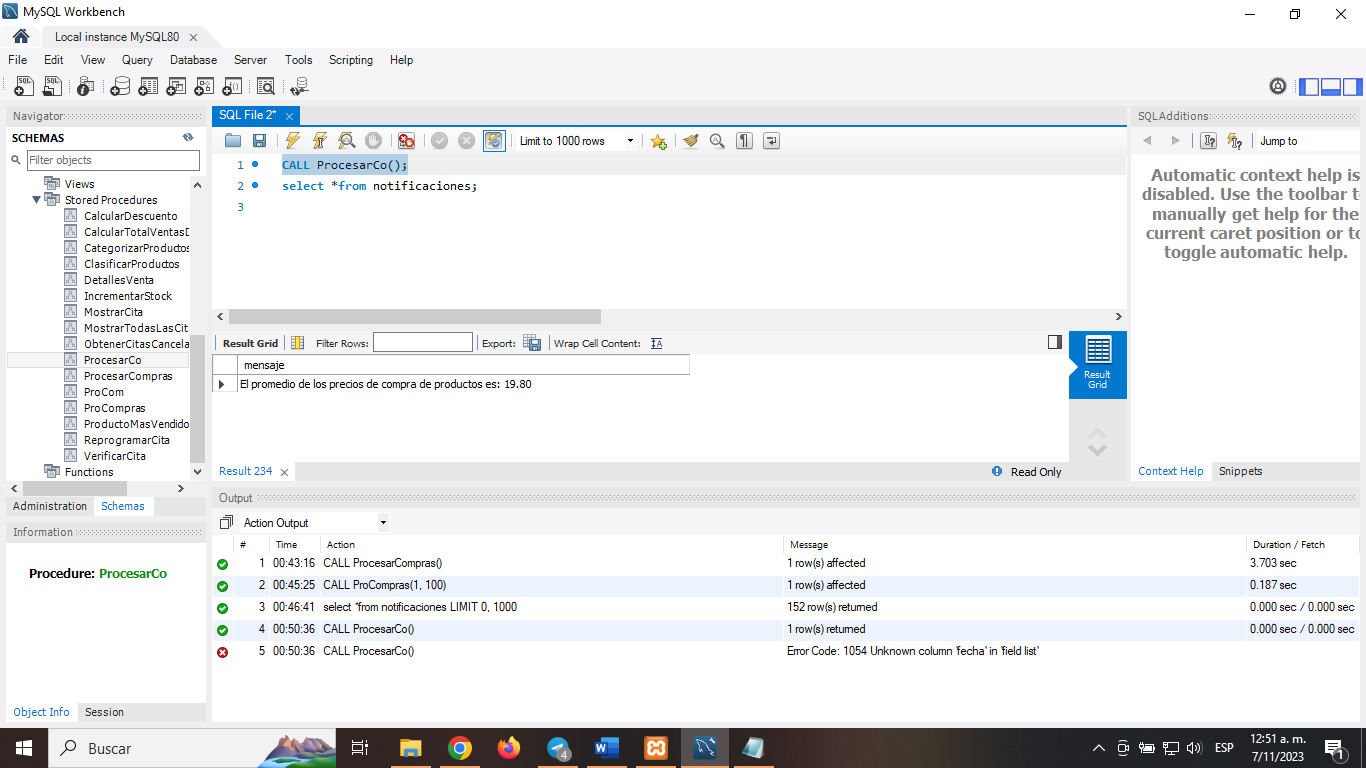
CALL ProCompras(compra\_id, total\_compra);remplazar compra id por id de la compra y el total por el valor

4.l procedimiento almacenado "ProcesarCo" calcula el promedio de los precios de compra de productos en la tabla

"productos". Luego, imprime este promedio en la consola y registra un mensaje junto con la fecha actual en la tabla

"notificaciones". Al llamar al procedimiento, se ejecutan estas acciones de manera automatizada, proporcionando

información útil sobre los precios de compra de productos



DELIMITER //

CREATE PROCEDURE ProcesarCo()

BEGIN

DECLARE promedio DECIMAL(10, 2);

SELECT AVG(precio\_compra) INTO promedio FROM productos;

SELECT CONCAT('El promedio de los precios de compra de productos es: ', promedio) AS mensaje;

INSERT INTO notificaciones (mensaje, fecha)

VALUES (CONCAT('El promedio de los precios de compra de productos es: ', promedio), NOW());

END//

DELIMITER ;

CALL ProcesarCo();

5. Este procedimiento almacenado calcula la fecha del último mes, crea una tabla temporal llamada TempResult para almacenar los resultados y luego actualiza esta tabla con los datos de las compras realizadas en el último mes. Finalmente, devuelve la información de proveedores, productos y precios correspondientes y elimina la tabla temporal.Una captura de pantalla de una red social

Descripción generada automáticamente

DELIMITER //

CREATE PROCEDURE ActualizarComprasUltimoMes()

BEGIN

DECLARE fecha\_mes\_anterior DATE;

SET fecha\_mes\_anterior = DATE\_SUB(NOW(), INTERVAL 1 MONTH);

-- Crear una tabla temporal para almacenar los resultados

CREATE TEMPORARY TABLE TempResult (

proveedor\_nombre VARCHAR(255),

producto\_nombre VARCHAR(255),

precio DECIMAL(10, 2)

);

-- Actualizar la tabla temporal con los datos de las compras del último mes

INSERT INTO TempResult (proveedor\_nombre, producto\_nombre, precio)

SELECT p.nombre AS proveedor\_nombre, pr.nombre AS producto\_nombre, dc.precio\_compra

FROM compras c

INNER JOIN proveedores p ON c.proveedor = p.id\_proveedor

INNER JOIN detalle\_compras dc ON c.id\_compra = dc.compra

INNER JOIN productos pr ON dc.producto = pr.id\_producto

WHERE c.fecha\_entrada >= fecha\_mes\_anterior;

-- Devolver los resultados

SELECT \* FROM TempResult;

-- Eliminar la tabla temporal

DROP TEMPORARY TABLE TempResult;

END //

DELIMITER ;

CALL ActualizarComprasUltimoMes();