**PROCESO DE GESTIÓN DE FORMACIÓN PROFESIONAL INTEGRAL**

**ACTIVIDAD BESE DE DATOS**

**ANALISIS Y DESARROLLO DE SOFTWARE 228118**

**MAURICIO RIVERA**

**FICHA 2672883**

**MONICA PENAGOS MARTINEZ**

**BASE DE DATOS**

**BOGOTA 2023**

**MODULO INVENTARIO**

1.crea un disparador llamado "NotificarInventario" que se ejecuta después de la inserción de un nuevo registro en la

tabla "inventario." El disparador busca información sobre el producto insertado y compone un mensaje de notificación.

Luego, inserta ese mensaje en una tabla de notificaciones.

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

DELIMITER //

CREATE TRIGGER NotificarInventario

AFTER INSERT ON inventario

FOR EACH ROW

BEGIN

DECLARE producto\_nombre VARCHAR(255);

DECLARE producto\_cantidad INT;

DECLARE mensaje\_notificacion TEXT;

SELECT nombre, cantidad INTO producto\_nombre, producto\_cantidad

FROM productos

WHERE id\_producto = NEW.producto\_id;

SET mensaje\_notificacion = CONCAT(

'Se ha insertado un nuevo producto en el inventario:',

'\nNombre del producto: ', producto\_nombre,

'\nCantidad: ', producto\_cantidad

);

INSERT INTO notificaciones (mensaje) VALUES (mensaje\_notificacion);

END;

//

DELIMITER ;

SELECT \* FROM notificaciones;

2. este procedimiento almacenado incrementará el campo "cantidad" en la tabla "productos" por el valor proporcionado en el parámetro "incremento" para todos los productos.

Una captura de pantalla de una red social

Descripción generada automáticamente

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

DELIMITER //

CREATE PROCEDURE IncrementarStock(

IN incremento INT

)

BEGIN

DECLARE producto\_id INT;

DECLARE stock\_actual INT;

DECLARE done INT DEFAULT FALSE;

-- Obtener la lista de productos

DECLARE cur CURSOR FOR SELECT id\_producto, cantidad FROM productos;

-- Manejador para cuando no haya más filas

DECLARE CONTINUE HANDLER FOR NOT FOUND SET done = TRUE;

-- Abrir el cursor

OPEN cur;

-- Iniciar el bucle

read\_loop: LOOP

FETCH cur INTO producto\_id, stock\_actual;

IF done THEN

LEAVE read\_loop;

END IF;

-- Incrementar el stock del producto

UPDATE productos

SET cantidad = stock\_actual + incremento

WHERE id\_producto = producto\_id;

END LOOP;

-- Cerrar el cursor

CLOSE cur;

END;

//

DELIMITER ;

CALL IncrementarStock(10);

3. Este procedimiento almacenado categoriza los productos en función de su stock y muestra los resultados al final.

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

DELIMITER //

CREATE PROCEDURE CategorizarProductos()

BEGIN

DECLARE producto\_id INT;

DECLARE stock INT;

DECLARE done INT DEFAULT FALSE;

-- Obtener la lista de productos

DECLARE cur CURSOR FOR SELECT id\_producto, cantidad FROM productos;

-- Manejador para cuando no haya más filas

DECLARE CONTINUE HANDLER FOR NOT FOUND SET done = TRUE;

-- Crear una tabla temporal para almacenar los resultados

CREATE TEMPORARY TABLE TempResult (

ID\_PRODUCTO INT,

NOMBRE VARCHAR(255),

STOCK INT,

Categoria VARCHAR(255)

);

-- Abrir el cursor

OPEN cur;

-- Iniciar el bucle

read\_loop: LOOP

FETCH cur INTO producto\_id, stock;

IF done THEN

LEAVE read\_loop;

END IF;

-- Categorizar el producto en función de su stock

INSERT INTO TempResult (ID\_PRODUCTO, NOMBRE, STOCK, Categoria)

VALUES (producto\_id, (SELECT nombre FROM productos WHERE id\_producto = producto\_id), stock,

CASE

WHEN stock < 10 THEN 'Agotado'

WHEN stock >= 10 THEN 'Producto con buen stock'

ELSE 'Sin Categoría'

END);

END LOOP;

-- Cerrar el cursor

CLOSE cur;

-- Mostrar los resultados

SELECT \* FROM TempResult;

-- Eliminar la tabla temporal

DROP TEMPORARY TABLE IF EXISTS TempResult;

END;

//

DELIMITER ;

CALL CategorizarProductos();

4. Este procedimiento almacenado determina el producto más vendido en el inventario calculando la suma de la cantidad para cada producto en la tabla "inventario" donde la fecha de salida no es nula. Luego, se ordena por la cantidad total en orden descendente y se toma el primer resultado. Si se encuentra un producto más vendido, se muestra su nombre. En caso contrario, se muestra un mensaje.

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

DELIMITER //

CREATE PROCEDURE ProductoMasVendido()

BEGIN

DECLARE producto\_mas\_vendido INT;

DECLARE cantidad\_mas\_vendida INT;

SELECT producto\_id, SUM(cantidad) AS total\_cantidad

INTO producto\_mas\_vendido, cantidad\_mas\_vendida

FROM inventario

WHERE fecha\_salida IS NOT NULL

GROUP BY producto\_id

ORDER BY total\_cantidad DESC

LIMIT 1;

IF producto\_mas\_vendido IS NOT NULL THEN

SELECT nombre

FROM productos

WHERE id\_producto = producto\_mas\_vendido;

ELSE

SELECT 'No hay un producto más vendido en el inventario.';

END IF;

END //

DELIMITER ;

call ProductoMasVendido();

5. Este procedimiento almacenado que calcula y encuentra al proveedor que ha recibido el mayor número de compras en la base de datos. Utiliza un bucle WHILE para iterar a través de los proveedores, sumando sus compras y rastreando al proveedor con el total más alto. Al final, muestra el nombre del proveedor con la mayor cantidad de compras

Una captura de pantalla de una red social

Descripción generada automáticamente

DELIMITER //

CREATE PROCEDURE EncontrarProveedorPrincipal()

BEGIN

DECLARE done INT DEFAULT FALSE;

DECLARE proveedor\_id INT;

DECLARE max\_total\_compras DECIMAL(10, 2);

DECLARE proveedor\_con\_mas\_compras INT;

-- Variables temporales para almacenar el resultado

DECLARE temp\_total DECIMAL(10, 2);

-- Cursor para obtener la lista de proveedores

DECLARE cur\_proveedores CURSOR FOR

SELECT id\_proveedor FROM proveedores;

-- Manejo de errores

DECLARE CONTINUE HANDLER FOR NOT FOUND SET done = TRUE;

OPEN cur\_proveedores;

-- Inicializamos el máximo a cero

SET max\_total\_compras = 0;

read\_loop: LOOP

FETCH cur\_proveedores INTO proveedor\_id;

IF done THEN

LEAVE read\_loop;

END IF;

-- Cálculo del total de compras para el proveedor actual

SELECT SUM(c.total) INTO temp\_total

FROM compras c

WHERE c.proveedor = proveedor\_id;

-- Si encontramos un nuevo máximo, lo actualizamos

IF temp\_total > max\_total\_compras THEN

SET max\_total\_compras = temp\_total;

SET proveedor\_con\_mas\_compras = proveedor\_id;

END IF;

END LOOP;

CLOSE cur\_proveedores;

-- Finalmente, seleccionamos el nombre del proveedor con más compras

SELECT nombre FROM proveedores

WHERE id\_proveedor = proveedor\_con\_mas\_compras;

END;

//

DELIMITER ;

CALL EncontrarProveedorPrincipal();