SISTEMA DE GESTIÓN DE VENTAS POR PRODUCTOS:

Objetivo:
El objetivo principal de este proyecto es diseñar, desarrollar y optimizar una base de datos para un Sistema de Gestión de Ventas utilizando SQL Server o el gestor de base de datos relacional de su preferencia, aplicando los conceptos fundamentales de las unidades 1 a 4 del curso.

Este sistema debe permitir almacenar, gestionar y optimizar la información de ventas, clientes, productos y transacciones, garantizando un acceso eficiente a los datos, una ejecución rápida de consultas, un manejo adecuado de concurrencia y transacciones, y un diseño seguro y escalable.

Requerimientos Generales:

1. Consultas SQL optimizadas.
2. Implementación de transacciones con control de concurrencia para garantizar la integridad de los datos y evitar conflictos.
3. Arquitectura eficiente, normalizada y optimizada para el uso de stored procedures, vistas , triggers y seguridad de usuarios.

Requerimientos Específicos:

1. Crear, Eliminar y Aumentar Productos.
2. Actualizar Stock o Inventario al realizar Transacciones o Eliminar Datos.
3. Consultar Ventas realizadas por Fechas
4. Creación y Eliminación de Clientes.
5. Registrar Ventas y Generación de Factura.
6. Aplicar Descuentos.
7. Ver Reportes.
8. Creación de Usuarios para mayor Seguridad.
9. Creación de Índices

.
Diseño de Base de Datos:

Desarrollo:

Implementación de Índices:

- Manga:
  Se crearon 3 índices no clusterizados en las columnas Manga_name, Author_name y Genre debido a que consideramos que serán los principales motores de búsqueda en la tabla (Nivel de eficacia es baja debido a que la tabla no contiene tantos datos, mejora imperceptible).
- Volumen:
  Se crearon 2 índices no clusterizados en las columnas Id_Manga y Volume_nro debido a que consideramos que serán los principales motores de búsqueda en la tabla (Nivel de eficacia es alta debido a que la tabla contiene bastantes datos, mejora entre 1 segundo).
- Ventas:

Se creó un índice no clusterizado en las columna Id_Volume debido a que consideramos que será el pilar de la tabla (Nivel de eficacia es media a alta debido a que dependera si esta con un join o no, mejora 1 segundo).

- Detalles:
  Se creó tes índices no clusterizados en las columnas Sales_date, Id_Customer y Id_Employee debido a que consideramos que serán las principales columnas para hacer join y se usarán en vistas (Nivel de eficacia es alta debido a que esta tabla es un conector entre las tablas, mejora 1-2 segundos).
- Empleado:
  Se creó tes índices no clusterizados en las columnas Employee_Name, Email y Phone_number debido a que consideramos que serán las principales columnas para buscar información del Empleado (Nivel de eficacia es baja debido a que esta tabla no contiene tantos datos, mejora imperceptible).
- Cliente:
  Se creó tes índices no clusterizados en las columnas Customer_Name, Email y NIT debido a que consideramos que serán las principales columnas para buscar información del Cliente (Nivel de eficacia es alta debido a que esta tabla puede llegar a contener una gran cantidad de datos, mejora escalable).

Vistas:
Este caso se usa para evitar que Gerente pueda acceder a la información, hay 2 tipos de reportes.

Transacciones:
Para un mejor manejo de transacciones se implementó triggers que permiten un mejor flujo de la información para evitar que choque o cause un error, además de manejar la atomicidad y evitar bloqueos, también los índices son utilizados.

Stored Procesures y triggers: En este caso se crearon diferentes triggers para diferentes funciones:

Triggers:

- Cliente(DELETE): En caso de eliminar un cliente, esté en Detalles debe ser reemplazado por un cliente predeterminado.
- Empleado(DELETE): En caso de eliminar un empleado, esté en Detalles debe ser reemplazado por un cliente predeterminado.
- Venta y Venta Detalle(DELETE): En este caso son inseparables las tablas, si un dato se elimina en una, repercute en la otra.
- Volumen(DELETE): En este caso debe reemplazar el volumen determinado por uno predeterminado en las tabla Ventas.
- Manga(DELETE): En este caso debe reemplazar el manga por uno predeterminado en las tabla Volume.
- Venta y Venta Detalle(INSERT): En este caso al ingresar, primero debe existir detalle venta que comprueba que existe una compra y luego ventas que actualiza a detalle venta.

Stored Procesures:

- Aumentar Stock: Aumenta la cantidad de stock en volumenes

Seguridad: En este caso se crearán 3 roles para evitar crear choques, los usuarios son Vendedor (Puede modificar o agregar ciertas tablas), Gerente (Solo puede ver reportes) y DBA (Dueño Total)

https://github.com/dAHTvind888/DB2-TiendaManga