Práctica 18 – Bases de datos

En esta práctica utilizaremos operaciones básicas CRUD y algunas operaciones DDL para trabajar con bases de datos relacionales en nuestras aplicaciones.

Práctica

Objetivo: Desarrollar un programa en Java que gestione una base de datos MySQL para una tienda online, implementando las siguientes funcionalidades mediante un menú interactivo.

Requisitos del programa:

Menú Principal

1. Crear tablas:
    a. Crear tabla clientes.
    b. Crear tabla población
    c. Crear tabla provincias
    d. Crear tabla productos
    e. Crear tabla categorías
2. Rellenar datos aleatorios
3. Insertar nuevo cliente / Población / Provincia / Producto / Categoría
4. Mostrar registros:
    a. clientes.
    b. población
    c. provincias
    d. productos
    e. categorías
5. Eliminar registro:
    a. clientes.
    b. población
    c. provincias
    d. productos
    e. Categorías
6. Actualizar precios de productos (en base a un precio facilitado por el usuario)
7. Salir

Funcionalidades obligatorias:

A) Crear tablas:
   - Permitir elegir qué tablas crear (clientes, provincias, poblaciones, productos, categorías).
   - Usar sentencias SQL para crear las tablas con la estructura definida el anexo.
   - Verificar si las tablas ya existen antes de crearlas.

B) Rellenado automático de datos mediante sentencia SQL.

C) Insertar cliente manual:
   - Solicitar al usuario datos (nombre, apellido, email, teléfono, ...)
   - [opcional] Validar que el email no exista previamente en la base de datos.
   - Mostrar mensaje de éxito/error tras la inserción.

D) Eliminar registro:
   - Permitir eliminar un cliente por su ID.
   - [Opcional] Mostrar lista de clientes antes de la eliminación.
   - Usar `commit` para garantizar integridad.

E) Mostrar registros:
   - Opción 1: Mostrar clientes con su población y provincia.
   - Opción 2: Mostrar productos con su categoría.
   - [Opcional] Formatear la salida en consola como tabla (tablas).

F) Actualizar precios:
   - Solicitar al usuario un precio límite (ej: 50€).
   - Actualizar todos los productos con precio inferior al límite, incrementándolos un 10%.
   - Mostrar el número de productos afectados.

Requisitos técnicos:

- Usar JDBC para la conexión a MySQL.
- Utilizar `PreparedStatement` para evitar inyecciones SQL.
- Gestionar errores con bloques `try-catch` y mensajes descriptivos.
- Cerrar recursos (`Connection`, `Statement`) en bloques `finally`.

ANEXO:

Datos de conexión:

Bases de datos:	Mysql 5.7.44
Hostname:	51.210.126.152
Nombre base de datos:	practicas
Usuario:	practicas
Contraseña:	OjsWSm8lQ5OYY26R6dIG

TABLAS:

#Clientes

Nombre campo	Tipo datos	Requisitos
ClienteID	INT	Clave primaria, único, no nulo, autoincrementado.
Nombre	VARCHAR(100)	No nulo, para almacenar el nombre del cliente.
Apellido	VARCHAR(100)	No nulo, para almacenar el apellido del cliente.
Email	VARCHAR(255)	Único, no nulo, almacena el correo electrónico.
Telefono	VARCHAR(15)	Opcional, almacena el número de teléfono.
Direccion	TEXT	Opcional, almacena la dirección completa del cliente.
PoblacionID	INT	Clave foránea, no nulo, referencia a la tabla Poblaciones (Población).
FechaNacimient	DATE	Opcional, almacena la fecha de nacimiento del cliente.
Activo	BOOLEAN	No nulo, valor por defecto TRUE, indica si el cliente está activo o inactivo.
FechaCreacion	TIMESTAMP	No nulo, valor por defecto= fecha hoy.

#Poblaciones

Nombre campo	Tipo datos	Requisitos
PoblacionID	INT	Clave primaria, único, no nulo, autoincrementado.
NombrePoblacion	VARCHAR(100)	No nulo, para almacenar el nombre de la población.
ProvinciaID	INT	Clave foránea, no nulo, referencia a la tabla Provincias(ProvinciaID).

#Provincias

Nombre campo	Tipo datos	Requisitos
ProvinciaID	INT	Clave primaria, único, no nulo, autoincrementado.
NombreProvincia	VARCHAR(100)	No nulo, para almacenar el nombre de la provincia.

#Productos

Nombre campo	Tipo datos	Requisitos
ProductoID	INT	Clave primaria, único, no nulo, autoincrementado.
NombreProducto	VARCHAR(150)	No nulo, almacena el nombre del producto.
DescripcionProducto	TEXT	Opcional. Descripción detallada del producto.
PrecioUnitario	DECIMAL(10,2	No nulo. Precio unitario del producto (2 decimales).
StockDisponible	INT	No nulo, valor por defecto 0. Cantidad disponible en inventario.
CategoriaID	INT	Clave foránea, no nulo, referencia a la tabla Categorias(CategoriaID).
FechaCreacion	TIMESTAMP	No nulo, valor por defecto = fecha hoy.

#Categorías

Nombre campo	Tipo datos	Requisitos
CategoriaID	INT	Clave primaria, único, no nulo, autoincrementado.
NombreCategoria	VARCHAR(100)	No nulo, almacena el nombre de la categoría.
DescripcionCategoria	TEXT	Opcional, almacena descripción detallada de la categoría.

Relaciones entre tablas:

1. La tabla **Clientes** tiene una relación con la tabla **Poblaciones**, mediante el campo foráneo (`PoblacionID → Poblaciones.PoblacionID`). Esto permite asociar cada cliente a una población específica.
2. La tabla **Poblaciones** tiene una relación con la tabla **Provincias**, mediante el campo foráneo (`ProvinciaID → Provincias.ProvinciaID`). Esto permite asociar cada población a una provincia.
3. La tabla **Productos** tiene una relación con la tabla **Categorías**, mediante el campo foráneo (`CategoriaID → Categorias.CategoriaID`). Esto permite clasificar los productos dentro de categorías específicas.