VANNI, ALEJANDRO – AP3

**DIAGRAMAS DE SECUENCIA**

A continuación se presentan dos diagramas de secuencia UML desarrollados en sintaxis Mermaid. Estos representan interacciones clave dentro del sistema: el registro de un nuevo pedido y la actualización del stock de productos.

**📌 Diagrama de Secuencia 1: Registrar nuevo pedido**

```mermaid  
sequenceDiagram  
 participant Usuario  
 participant Sistema  
 participant BaseDatos  
  
 Usuario->>Sistema: Registrar nuevo pedido  
 Sistema->>BaseDatos: Verifica si el cliente existe  
 alt Cliente nuevo  
 Sistema->>BaseDatos: Insertar nuevo cliente  
 end  
 Sistema->>BaseDatos: Consultar productos  
 Usuario->>Sistema: Selecciona productos y cantidades  
 Sistema->>BaseDatos: Registrar pedido y detalle  
 Sistema-->>Usuario: Confirmación del pedido  
```

**📌 Diagrama de Secuencia 2: Actualizar stock de producto**

```mermaid  
sequenceDiagram  
 participant Usuario  
 participant Sistema  
 participant BaseDatos  
  
 Usuario->>Sistema: Actualizar stock  
 Sistema->>BaseDatos: Buscar producto  
 Sistema->>BaseDatos: Actualizar cantidad  
 Sistema-->>Usuario: Confirmación de actualización  
```

**IMPLEMENTACIÓN DE BASE DE DATOS EN MYSQL**

A continuación se presentan las sentencias SQL utilizadas para la creación de la base de datos del sistema. Las tablas creadas permiten gestionar clientes, productos, pedidos y sus detalles. Estas sentencias cumplen con las observaciones solicitadas, incluyendo claves primarias, foráneas, y tipos de datos apropiados.

**📌 Sentencias CREATE TABLE:**

-- Tabla de clientes  
CREATE TABLE Cliente (  
 id\_cliente INT AUTO\_INCREMENT PRIMARY KEY,  
 nombre VARCHAR(100) NOT NULL,  
 apellido VARCHAR(100) NOT NULL,  
 dni VARCHAR(20) NOT NULL UNIQUE,  
 direccion VARCHAR(255),  
 telefono VARCHAR(20),  
 email VARCHAR(100)  
);  
  
-- Tabla de productos  
CREATE TABLE Producto (  
 id\_producto INT AUTO\_INCREMENT PRIMARY KEY,  
 nombre VARCHAR(100) NOT NULL,  
 descripcion TEXT,  
 precio DECIMAL(10, 2) NOT NULL,  
 stock INT NOT NULL  
);  
  
-- Tabla de pedidos  
CREATE TABLE Pedido (  
 id\_pedido INT AUTO\_INCREMENT PRIMARY KEY,  
 fecha DATE NOT NULL,  
 id\_cliente INT,  
 total DECIMAL(10, 2),  
 FOREIGN KEY (id\_cliente) REFERENCES Cliente(id\_cliente)  
);  
  
-- Tabla de detalle de pedidos  
CREATE TABLE DetallePedido (  
 id\_detalle INT AUTO\_INCREMENT PRIMARY KEY,  
 id\_pedido INT,  
 id\_producto INT,  
 cantidad INT,  
 precio\_unitario DECIMAL(10, 2),  
 subtotal DECIMAL(10, 2),  
 FOREIGN KEY (id\_pedido) REFERENCES Pedido(id\_pedido),  
 FOREIGN KEY (id\_producto) REFERENCES Producto(id\_producto)  
);

**📌 Ejemplos de inserción, consulta y borrado:**

-- Insertar cliente  
INSERT INTO Cliente (nombre, apellido, dni, direccion, telefono, email)  
VALUES ('Juan', 'Pérez', '30123456', 'Av. Siempre Viva 123', '1123456789', 'juan.perez@example.com');  
  
-- Insertar producto  
INSERT INTO Producto (nombre, descripcion, precio, stock)  
VALUES ('Gin London Dry', 'Botella 750ml', 8500.00, 25);  
  
-- Consultar productos  
SELECT \* FROM Producto;  
  
-- Consultar pedidos de un cliente  
SELECT p.id\_pedido, p.fecha, d.cantidad, d.subtotal  
FROM Pedido p  
JOIN DetallePedido d ON p.id\_pedido = d.id\_pedido  
WHERE p.id\_cliente = 1;  
  
-- Eliminar producto  
DELETE FROM Producto WHERE id\_producto = 1;

**PLAN DE PRUEBAS Y CASOS DE PRUEBA**

A continuación, se presenta un plan de pruebas básico junto con casos de prueba diseñados para verificar el funcionamiento del prototipo desarrollado. Se enfocan en operaciones fundamentales del sistema como la gestión de pedidos y productos.

**📌 Caso de prueba 1: Registro de nuevo pedido**

- Objetivo: Verificar que el sistema registre correctamente un pedido con sus productos.  
- Datos de entrada: Cliente existente, productos con stock suficiente, cantidades válidas.  
- Resultado esperado: El pedido se registra en la base de datos con los detalles correspondientes.  
- Resultado obtenido: ✅ Pedido registrado correctamente.  
- Estado: Aprobado.

**📌 Caso de prueba 2: Actualización de stock**

- Objetivo: Verificar que el sistema permita actualizar el stock de un producto.  
- Datos de entrada: ID de producto existente, nueva cantidad.  
- Resultado esperado: El stock del producto se actualiza en la base de datos.  
- Resultado obtenido: ✅ Stock actualizado correctamente.  
- Estado: Aprobado.

**📌 Caso de prueba 3: Eliminación de producto**

- Objetivo: Verificar que el sistema permita eliminar un producto del catálogo.  
- Datos de entrada: ID de producto existente sin pedidos asociados.  
- Resultado esperado: El producto es eliminado de la base de datos.  
- Resultado obtenido: ✅ Producto eliminado correctamente.  
- Estado: Aprobado.