Proyecto Final – Librería Online

Autora: Noelia Rossi
Curso: SQL – Coderhouse
Repositorio (scripts SQL): https://github.com/noeliaximenarossi-ctrl/SQL-Coder

Sumario

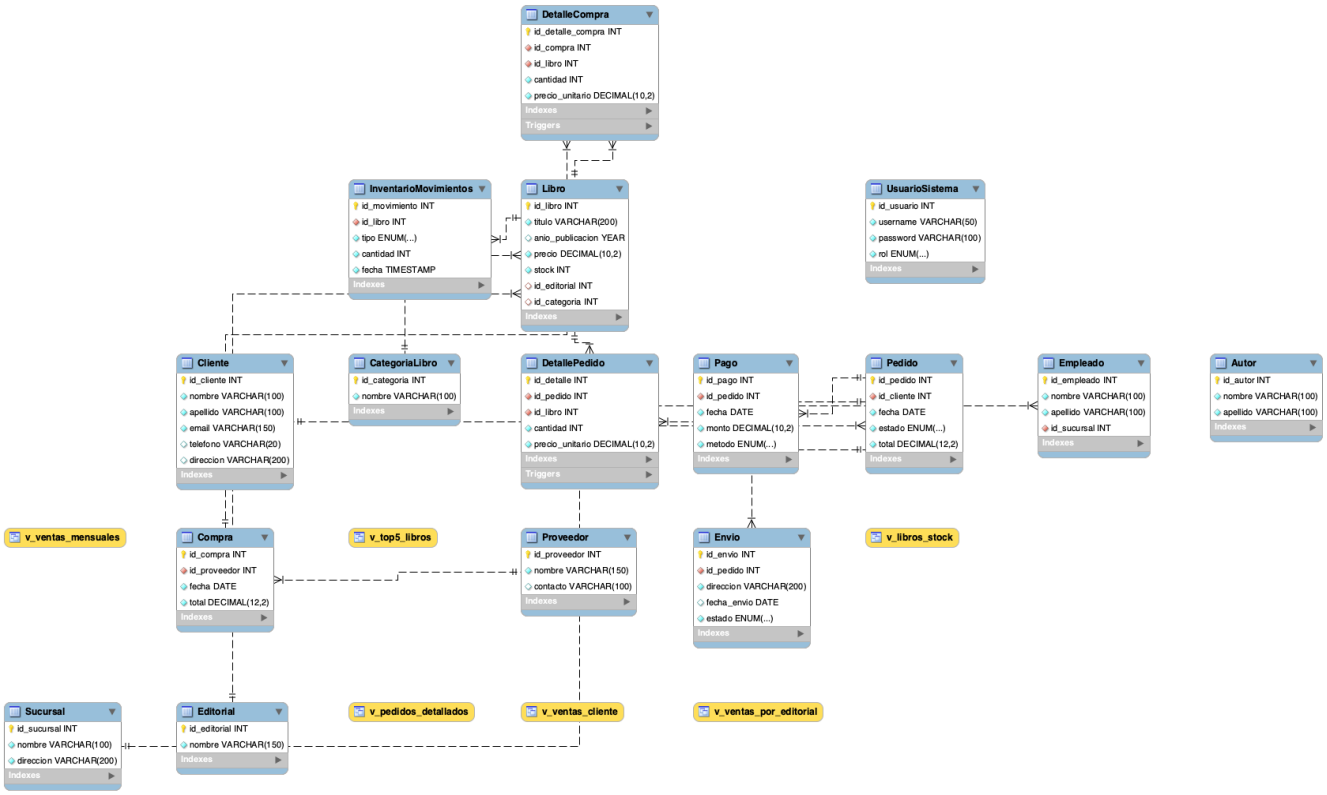
1. Introducción y Objetivo

Se implementa una base relacional para una Librería Online con foco en operaciones de compras, ventas, pagos, envíos e inventario, garantizando integridad referencial y automatización mediante objetos SQL. El objetivo es disponer de un backend de datos capaz de soportar informes analíticos y dashboards.

2. Situación Problemática y Modelo de Negocio

Se reemplaza una gestión manual en planillas por un esquema normalizado. El modelo contempla clientes, libros (editorial/categoría), proveedores, compras y detalle, pedidos de clientes y detalle, pagos, envíos, sucursales/empleados y bitácora de inventario.

3. Diagrama Entidad■Relación (ER)



4. Paso a Paso de Ejecución

- Abrir MySQL Workbench (MySQL 8.x).
- Ejecutar en orden:
- a) libreriaonlineRossi_creacion.sql → crea base y tablas (+ FKs).
- b) libreriaonlineRossi_objetos.sql → crea vistas, funciones, SPs, triggers (con DROP IF EXISTS).
- c) libreriaonlineRossi_inserts_correct_v2.sql → inserta datos de prueba (coherentes con triggers).
- d) libreriaonlineRossi_tests_PASS_FAIL_v4.sql → valida integridad, triggers y SPs.
- Si hay datos previos: ejecutar libreriaonlineRossi_drop_all.sql o DROP DATABASE libreria_online;
- Confirmar base activa con USE libreria_online;

5. Checklist de Cumplimiento (rúbrica)

Criterio	Cumplimiento
Tablas (15+)	OK (16 tablas)
Vistas (5+)	OK (6 vistas)
Stored Procedures (2+)	OK (4 SPs)
Funciones (2+)	OK (3 funciones)
Triggers (2+)	OK (6 triggers)

Script creación + script inserción	OK (se proveen)
Script objetos (vistas/SPs/funciones/triggers)	OK
Script tests PASS/FAIL	OK
ER en PDF + explicación	OK
Datasets para informes	OK (consultas incluidas)

6. Estructura de Tablas (detalle por columna y claves)

Cliente

Información de clientes.

Columna	Tipo/Clave	Descripción
id_cliente	INT PK	Identificador
nombre	VARCHAR(100)	Nombre
apellido	VARCHAR(100)	Apellido
email	VARCHAR(150) UNIQUE	Correo
telefono	VARCHAR(20)	Teléfono
direccion	VARCHAR(200)	Domicilio

Autor

Autores (referencial).

Columna	Tipo/Clave	Descripción
id_autor	INT PK	Id
nombre	VARCHAR(100)	Nombre
apellido	VARCHAR(100)	Apellido

Editorial

Editoriales.

Columna	Tipo/Clave	Descripción
id_editorial	INT PK	Id
nombre	VARCHAR(150) UNIQUE	Nombre

CategoriaLibro

Categorías de libros.

Columna	Tipo/Clave	Descripción
id_categoria	INT PK	Id
nombre	VARCHAR(100) UNIQUE	Nombre

Libro

Catálogo de libros.

Columna	Tipo/Clave	Descripción
id_libro	INT PK	Id
titulo	VARCHAR(200)	Título
anio_publicacion	YEAR	Año
precio	DECIMAL(10,2)	Precio
stock	INT	Stock
id_editorial	INT FK	-> Editorial
id_categoria	INT FK	-> CategoriaLibro

Proveedor

Proveedores.

Columna	Tipo/Clave	Descripción
id_proveedor	INT PK	Id
nombre	VARCHAR(150)	Nombre
contacto	VARCHAR(100)	Contacto

Compra

Compras a proveedores.

Columna	Tipo/Clave	Descripción
id_compra	INT PK	Id
id_proveedor	INT FK	-> Proveedor
fecha	DATE	Fecha
total	DECIMAL(12,2)	Total por trigger

DetalleCompra

Ítems de compra.

Columna	Tipo/Clave	Descripción
id_detalle_compra	INT PK	Id
id_compra	INT FK	-> Compra
id_libro	INT FK	-> Libro
cantidad	INT	Cant.
precio_unitario	DECIMAL(10,2)	Precio

Pedido

Pedidos de clientes.

Columna	Tipo/Clave	Descripción
id_pedido	INT PK	Id
id_cliente	INT FK	-> Cliente
fecha	DATE	Fecha
estado	ENUM('pendiente','enviado','entregado')	Estado
total	DECIMAL(12,2)	Total por trigger

DetallePedido

Ítems del pedido.

Columna	Tipo/Clave	Descripción
id_detalle	INT PK	Id
id_pedido	INT FK	-> Pedido
id_libro	INT FK	-> Libro
cantidad	INT	Cant.
precio_unitario	DECIMAL(10,2)	Precio

Pago

Pagos de pedidos.

Columna	Tipo/Clave	Descripción
id_pago	INT PK	Id
id_pedido	INT FK	-> Pedido
fecha	DATE	Fecha
monto	DECIMAL(10,2)	Monto
metodo	ENUM('tarjeta','transferencia','efectivo')	Método

Sucursal

Sucursales.

Columna	Tipo/Clave	Descripción
id_sucursal	INT PK	Id
nombre	VARCHAR(100)	Nombre
direccion	VARCHAR(200)	Dirección

Empleado

Empleados por sucursal.

Columna	Tipo/Clave	Descripción
id_empleado	INT PK	Id
nombre	VARCHAR(100)	Nombre
apellido	VARCHAR(100)	Apellido
id_sucursal	INT FK	-> Sucursal

UsuarioSistema

Usuarios del sistema.

Columna	Tipo/Clave	Descripción
id_usuario	INT PK	Id
username	VARCHAR(50) UNIQUE	Usuario
password	VARCHAR(100)	Clave/Hash
rol	ENUM('admin','vendedor','cliente')	Rol

Envio

Envíos de pedidos.

Columna	Tipo/Clave	Descripción
id_envio	INT PK	Id
id_pedido	INT FK	-> Pedido
direccion	VARCHAR(200)	Dirección
fecha_envio	DATE	Fecha
estado	ENUM('preparando','enviado','entregado')	Estado

InventarioMovimientos

Bitácora de stock.

Columna	Tipo/Clave	Descripción
id_movimiento	INT PK	Id
id_libro	INT FK	-> Libro
tipo	ENUM('entrada','salida')	Tipo
cantidad	INT	Cant.
fecha	TIMESTAMP DEFAULT CURRENT_TIMESTAMP	Fecha

7. Objetos de Base (definición y propósito)

7.1 Vistas

- **v_ventas_cliente**: Pedidos y total comprado por cliente (para ranking). Archivo: libreriaonlineRossi_objetos.sql
- **v_libros_stock**: Catálogo con stock y atributos (para control). Archivo: libreriaonlineRossi_objetos.sql
- **v_pedidos_detallados**: Detalle de pedidos con subtotales (para auditoría). Archivo: libreriaonlineRossi_objetos.sql
- **v_ventas_mensuales**: Agregación de ventas por mes (para series). Archivo: libreriaonlineRossi_objetos.sql
- **v_top5_libros**: Top 5 títulos por unidades (para best-sellers). Archivo: libreriaonlineRossi_objetos.sql
- **v_ventas_por_editorial**: Ventas agregadas por editorial (mix de oferta). Archivo: libreriaonlineRossi_objetos.sql

7.2 Funciones

- **fn_total_pedidos_cliente(p_id)**: Devuelve SUM(total) de los pedidos del cliente.
- **fn_total_pagado_pedido(p_pedido)**: Devuelve SUM(monto) de pagos vinculados al pedido.
- **fn_saldo_pedido(p_pedido)**: Calcula saldo = total - pagos.

7.3 Stored Procedures

- **sp_crear_pedido(p_cliente,p_fecha,p_libro,p_cantidad,p_precio)**: Crea pedido + 1er ítem; triggers recalculan total/stock.
- **sp_agregar_detalle(p_pedido,p_libro,p_cantidad,p_precio)**: Agrega ítem; triggers actualizan total/stock.
- **sp_registrar_pago(p_pedido,p_monto,p_metodo)**: Valida método ENUM y registra pago con fecha actual.
- **sp_actualizar_estado_pedido(p_pedido,p_estado)**: Valida estado ENUM y actualiza estado del pedido.

7.4 Triggers

- **trg_detallepedido_ai/au/ad**: Mantienen total de pedido y stock ante INSERT/UPDATE/DELETE (salidas).
- **trg_detallecompra_ai/au/ad**: Mantienen total de compra y stock ante operaciones en DetalleCompra (entradas).

8. Inserción de Datos y Volumen

El script de inserciones genera ~100 clientes, 120 libros, 80 pedidos con 200 líneas, 120 pagos, 40 compras con 120 líneas, 15 proveedores, 4 sucursales y 20 empleados. Los triggers ajustan totales y stock automáticamente.

9. Consultas de Informes (datasets para dashboards)

Ventas mensuales

```
SELECT DATE_FORMAT(p.fecha,'%Y-%m') AS periodo, COUNT(DISTINCT p.id_pedido) AS pedidos,
SUM(p.total) AS total_ventas FROM Pedido p GROUP BY DATE_FORMAT(p.fecha,'%Y-%m') ORDER BY
periodo;
```

Top 10 libros

```
SELECT l.titulo, SUM(d.cantidad) AS unidades FROM DetallePedido d JOIN Libro l ON
l.id_libro=d.id_libro GROUP BY l.titulo ORDER BY unidades DESC LIMIT 10;
```

Ventas por editorial

```
SELECT e.nombre AS editorial, SUM(d.cantidad*d.precio_unitario) AS total_ventas FROM
DetallePedido d JOIN Libro l ON l.id_libro=d.id_libro JOIN Editorial e ON
e.id_editorial=l.id_editorial GROUP BY e.nombre ORDER BY total_ventas DESC;
```

Top clientes

```
SELECT CONCAT(c.nombre,' ',c.apellido) AS cliente, COALESCE(SUM(p.total),0) AS total_comprado
FROM Cliente c LEFT JOIN Pedido p ON p.id_cliente=c.id_cliente GROUP BY c.id_cliente, cliente
ORDER BY total_comprado DESC LIMIT 10;
```

Estado de pedidos

```
SELECT p.estado, COUNT(*) AS cantidad FROM Pedido p GROUP BY p.estado;
```

Stock bajo

```
SELECT l.titulo, l.stock FROM Libro l WHERE l.stock<=10 ORDER BY l.stock ASC LIMIT 10;
```

10. Validaciones / Tests PASS■FAIL

El archivo 'libreriaonlineRossi_tests_PASS_FAIL_v4.sql' realiza: sanidad de tablas, verificación de FKs (sin huérfanos), existencia de objetos, prueba controlada de triggers (totales y stock) y flujo completo de SPs.

11. Herramientas y Tecnologías

MySQL 8.0, MySQL Workbench (Reverse Engineer para ER), GitHub, (opcional) Python + Matplotlib/ReportLab para generar gráficos y PDFs.