**1. Creación de la Base de Datos**

Para la creación de la base de datos tienda\_libros, se utilizó la sentencia CREATE DATABASE IF NOT EXISTS para evitar errores en caso de que ya existiera. Esta práctica es recomendable en scripts que pueden ejecutarse múltiples veces. La selección de la base de datos se realizó con USE tienda\_libros para establecer el contexto de trabajo y asegurar que todas las operaciones posteriores se ejecuten en la base de datos correcta.

**2. Modelado Entidad-Relación**

El diseño del modelo ER se basó en un sistema típico de gestión de una tienda de libros, identificando cinco entidades principales: Autor, Libro, Cliente, Pedido y DetallePedido. Las relaciones establecidas siguen un patrón lógico donde un autor puede escribir múltiples libros (1:N), un cliente puede realizar múltiples pedidos (1:N), y cada pedido puede contener múltiples libros a través de la tabla de detalle que actúa como una relación muchos a muchos entre Pedido y Libro. Esta estructura normalizada evita la redundancia de datos y mantiene la integridad referencial.

**3. Definición de Tablas (DDL)**

Las tablas se diseñaron siguiendo las mejores prácticas de bases de datos relacionales. Se definieron tipos de datos apropiados para cada columna: INT AUTO\_INCREMENT para las claves primarias, VARCHAR con longitudes específicas para textos, DECIMAL para valores monetarios, DATE y YEAR para fechas, y ENUM para estados predefinidos. Las restricciones NOT NULL se aplicaron a campos esenciales, mientras que las claves foráneas incluyen ON DELETE CASCADE para mantener la consistencia cuando se eliminan registros padre. Se implementó una columna calculada subtotal en DetallePedido que se actualiza automáticamente usando GENERATED ALWAYS AS.

**4. Inserción de Datos (DML)**

Los datos de prueba se insertaron respetando estrictamente las relaciones definidas en el modelo. Se comenzó con las tablas padre (Autor y Cliente) antes de insertar en las tablas hijo (Libro, Pedido, DetallePedido). Los datos incluyen autores reconocidos de la literatura latinoamericana e internacional, libros con precios realistas en pesos colombianos, clientes con información completa, y pedidos que reflejan escenarios reales de compra. Se aseguró que todas las claves foráneas tuvieran valores válidos existentes en las tablas referenciadas.**5. Consultas Básicas**

Las consultas básicas implementadas demuestran el uso de filtros con WHERE, ordenamiento con ORDER BY, y agregaciones con GROUP BY y COUNT. La primera consulta filtra libros por año de publicación y los ordena cronológicamente, mientras que la segunda utiliza un LEFT JOIN para incluir autores que podrían no tener libros asociados, mostrando el uso de agregaciones para contar libros por autor. Estas consultas son fundamentales para reportes básicos del sistema.

**6. Consultas Avanzadas (JOIN)**

Las consultas con JOIN demuestran la capacidad de relacionar información de múltiples tablas. Se implementaron INNER JOIN para obtener solo registros que tienen correspondencia en

**7. Actualización y Eliminación**

Las operaciones de actualización y eliminación se implementaron con precisión. Para la actualización, se modificó el precio unitario de un detalle específico usando WHERE con la clave primaria para evitar cambios no deseados. Para la eliminación, se creó primero un registro temporal que luego se eliminó, demostrando el uso seguro de DELETE con condiciones específicas. Estas operaciones son críticas para el mantenimiento de datos y requieren precaución para evitar pérdida de información.

**8. Transacciones**

La implementación de transacciones demuestra el control de la consistencia de datos en operaciones múltiples. Se utilizó START TRANSACTION para iniciar el bloque, seguido de las operaciones de inserción en Pedido y DetallePedido, y finalmente COMMIT para confirmar los cambios. El uso de variables del sistema como LAST\_INSERT\_ID() permitió capturar el ID generado automáticamente para usarlo en operaciones relacionadas. Las transacciones son fundamentales cuando múltiples operaciones deben ejecutarse como una unidad atómica, garantizando que todas se completen exitosamente o ninguna se ejecute.

**9. Creación de Vistas**

La vista Vista\_Pedidos simplifica el acceso a información frecuentemente consultada, combinando datos de pedidos y clientes en una sola estructura virtual. Esta vista encapsula la lógica del JOIN, haciendo más fácil para los usuarios finales obtener información consolidada sin escribir consultas complejas. La consulta posterior sobre la vista demuestra cómo se pueden aplicar filtros adicionales sobre datos ya procesados, mejorando la eficiencia en consultas recurrentes y proporcionando una capa de abstracción que oculta la complejidad del modelo subyacente.

**Consideraciones Técnicas Adicionales**

El script incluye verificaciones y consultas adicionales que permiten validar la correcta implementación de todas las operaciones. Se agregaron consultas de resumen que muestran estadísticas útiles como ventas por autor, demostrando consultas más avanzadas que combinan múltiples tablas y funciones de agregación. El diseño permite escalabilidad futura y mantiene la flexibilidad para agregar nuevas funcionalidades sin comprometer la estructura existente.

**Conclusiones**

El taller se completó exitosamente cumpliendo todos los objetivos planteados. Se demostró competencia en DDL para definir estructuras de datos, DML para manipular información, consultas simples y complejas para extraer datos, transacciones para garantizar consistencia, y vistas para simplificar el acceso a datos. El modelo diseñado es robusto, normalizado y escalable, apropiado para un sistema real de gestión de tienda de libros