NORMALIZACION Y OPTIMIZACION DEL DISEÑO

Ejemplo: Una base de datos de facturación que normaliza información de clientes y productos, pero mantiene desnormalizada una tabla de ventas para mejorar consultas frecuentes.

Ejercicios propuestos

1. Normalizar la base de datos de un sistema de inventario hasta 3FN.
2. Crear índices para mejorar consultas en una base de datos de ventas.
3. Decidir qué tablas conviene desnormalizar en un sistema de reportes.

DESARROLLO DEL EJERCICIO 3

Criterios para decidir qué tablas desnormalizar

1. **Tablas consultadas con alta frecuencia**
   - Si una tabla es usada repetidamente en reportes y requiere múltiples uniones (*joins*) con otras tablas, conviene integrar ciertos datos directamente.
   - ✅ *Ejemplo:* unir información de ventas con clientes y productos en una sola tabla para agilizar la generación de reportes.
2. **Tablas que requieren agregaciones complejas**
   - Cuando se calculan valores como sumas, promedios, conteos, etc., en cada consulta, puede ser más eficiente **almacenar esos valores precalculados** en columnas adicionales.
   - ✅ *Ejemplo:* guardar el total de ventas por mes directamente, en vez de calcularlo cada vez.
3. **Datos históricos que no cambian**
   - Si los datos no se modifican con frecuencia, desnormalizar no genera inconsistencias.
   - ✅ *Ejemplo:* registros de ventas pasadas o reportes anuales.
4. **Consultas de lectura intensiva y pocas escrituras**
   - En entornos donde se hacen muchas consultas y pocas actualizaciones, la desnormalización mejora mucho la rapidez sin causar problemas de actualización de datos.
   - ✅ *Ejemplo:* sistemas de inteligencia de negocios (BI) o *data warehouses*.

📊 3. Ejemplo Práctico – Sistema de Reportes de Ventas

Supongamos que tenemos estas tablas normalizadas:

- **Cliente** (ID_cliente, Nombre, Dirección, Teléfono)
- **Producto** (ID_producto, Nombre, Precio, Categoría)
- **Venta**(ID_venta, ID_cliente, ID_producto, Fecha, Cantidad, Total)

4. Ventajas de la Desnormalización

- ✅ **Mejor rendimiento en consultas de lectura** y generación de reportes.
- ✅ **Menor complejidad** en las consultas SQL.
- ✅ **Reducción de la carga** en el servidor de base de datos en operaciones frecuentes.
- ✅ **Aceleración de procesos analíticos**, ideal para dashboards en tiempo real.

⚠ 5. Desventajas y Precauciones

- ❌ **Redundancia de datos**: la misma información puede almacenarse en varias tablas.
- ❌ **Posibles inconsistencias** si los datos no se actualizan correctamente.
- ❌ **Mayor espacio de almacenamiento**.
- ❌ **Mantenimiento más complejo**: se deben actualizar múltiples campos cuando cambia un dato base.