Trabajo Práctico 4.1

Normalización de Bases de Datos Relacionales

Denormalización

- **Alumno**: Sussini Guanziroli, Patricio
- **Materia**: Bases de datos I
- **Tutora**: Constanza Uño

ACTIVIDAD II

➢ Análisis crítico y aplicación práctica de estructuras normalizadas

En esta actividad vas a evaluar estructuras de bases de datos que ya se encuentran normalizadas para detectar si realmente cumplen con las tres primeras formas normales (1FN, 2FN y 3FN) y para explorar casos donde la desnormalización puede ser una decisión válida.

➢ Parte 1 – Evaluación de diseño normalizado

Se entregará a los estudiantes una estructura de base de datos que aparenta estar en 3FN (con varias tablas y relaciones).

Tareas:

- Analizar las claves primarias y foráneas.
- Identificar posibles violaciones a la 2FN o 3FN, justificando cada hallazgo.
- En caso de detectar errores, proponer la estructura corregida.

Sugerencia de estructura para analizar:

- ❖ CLIENTES(id_cliente, nombre, email, id_localidad)
- ❖ VENTAS(id_venta, fecha, id_cliente, total)
- ❖ DETALLES_VENTA(id_venta, id_producto, cantidad, precio_unitario)
- ❖ PRODUCTOS(id_producto, descripcion, precio)
- ❖ LOCALIDADES(id_localidad, nombre_provincia)

Evaluación del Diseño Normalizado:

Auditamos el esquema de datos propuesto para validar su correcta normalización hasta 3FN.

1. **Análisis de la estructura:**
   - ➔ **Primera Forma Normal:** Todas las tablas cumplen la 1FN. Cada columna contiene valores atómicos y cada registro es único.
   - ➔ **Segunda Forma Normal:** La única tabla con clave primaria compuesta es DETALLES_VENTA con id_venta y id_producto como

PKs. Los atributos no clave como cantidad y precio unitario son dependientes, su funcionamiento de la clave completa, no solo parcialmente. Por lo tanto y consecuentemente el modelo cumple con la 2FN.

2. **Diagnóstico de Violaciones de la 3FN:**

   Buscamos eliminar dependencias transitivas. Encontramos una irregularidad en ventas.

   - ➔ **Hallazgo:** El campo **total** en la tabla **VENTAS** viola los principios de la 3FN.
   - ➔ Este campo es un dato derivado y calculado. Su valor se puede obtener en cualquier momento sumando el resultado de **cantidad * precio_unitario** para todos los productos asociados al mismo id_venta en la tabla DETALLES_VENTA.
   - ➔ Almacenar este dato crea redundancia y un riesgo de anomalías de actualización y problemas de escalabilidad y consistencia.

3. **Propuesta de mejora:**

   Para que el modelo se adhiera a la 3era forma normal, se debe eliminar el dato previamente detallado que es calculado y almacenado.

   Estructura Corregida:

   - ❖ CLIENTES(id_cliente, nombre, email, id_localidad)
   - ❖ VENTAS(id_venta, fecha, id_cliente) ⇒ *Eliminamos el campo total.*
   - ❖ DETALLES_VENTA(id_venta, id_producto, cantidad, precio_unitario)
   - ❖ PRODUCTOS(id_producto, descripcion, precio)
   - ❖ LOCALIDADES(id_localidad, nombre_provincia)

➢ Parte 2 – Normalización inversa guiada

Se solicita al estudiante que seleccione una porción de la base ya normalizada y la denormalice parcialmente para cumplir uno de los siguientes objetivos:

- Reducir cantidad de joins en reportes frecuentes.
- Aumentar la velocidad de consultas en dashboards o informes.
- Simplificar el modelo en contextos de baja escala.

➢ Deben:

- Explicar qué tablas deciden fusionar o simplificar.
- Justificar la decisión desde el punto de vista funcional o de rendimiento.
- Señalar las ventajas y desventajas de este cambio.

Propuesta de Denormalización Estratégica:

Procedemos a desnormalizar para obtener un rendimiento.

1. **Objetivo y Escenario de Desnormalización:**
   - ➔ **Objetivo:** Aumentar la velocidad de consulta para un reporte en dashboard.
   - ➔ **Escenario:** El reporte más utilizado por el área comercial es “Ventas Detalladas”, que debe mostrar por cada línea de venta el nombre del producto y el nombre de la categoría a la que pertenece.
   - ➔ **Problema de Rendimiento:** En el modelo normalizado para generar este reporte es necesario consultar la tabla DETALLES_VENTA y hacer dos JOINs, uno con PRODUCTOS y otro con la tabla CATEGORIAS. Estas uniones ralentizar el tiempo del reporte.

2. **Técnica y Modelo Desnormalizado Propuesto:**
    - ➔ **Técnica aplicada:** Duplicación de atributos. Se copian las columnas de una tabla a la otra para evitar los JOIN.
    - ➔ **Modelo Denormalizado:** Se modifica la tabla DETALLES_VENTA para incluir la descripción del producto y el nombre de su categoría.
        - ◆ **Antes:** DETALLES_VENTA(id_venta, id_producto, cantidad, precio_unitario)
        - ◆ **Después:** DETALLES_VENTA(id_venta, id_producto, cantidad, precio_unitario, nombre_producto, categoria_producto)

3. **Justificación:**

    La decisión estratégica tomada y descrita previamente, intercambia la consistencia extrema del modelo por mayor rendimiento, buscando un balance funcional orientado a las necesidades prácticas.

    - ➔ **Justificación de Rendimiento: (Instrucciones generadas por IA)**
        - ◆ Consulta normalizada es **lenta.**

          SELECT d.id_venta, p.descripcion, c.nombre, d.cantidad
          FROM DETALLES_VENTA d
          JOIN PRODUCTOS p ON d.id_producto = p.id_producto
          JOIN CATEGORIAS c ON p.id_categoria = c.id;
        - ◆ La desnormalizada es **rápida.**

          SELECT id_venta, nombre_producto, categoria_producto, cantidad
          FROM DETALLES_VENTA;

- ➔ **Discusión Ventajas vs. Desventajas:**
    - ◆ **Ventajas:**
        - **Mayor Rendimiento de Lectura:** Acceso directo a los datos.
        - **Simplificación de consultas:** Sentencias SQL más sencillas.

- **Desventajas:**
    - **Redundancia y mayor uso de almacenamiento:** Nombre de producto categoría se repiten en cada registro de venta, aumentando el tamaño de la base de datos.
    - **Riesgo de inconsistencia:** Si el nombre de un producto cambia, este cambio no se va a ver reflejado en los registros históricos de ventas. Si así se necesitase se requeriría una actualización masiva, compleja y prácticamente imposible.
    - **Complejidad de Mantenimiento:** La lógica para insertar nuevos datos en DETALLES_VENTA ahora debe incluir la búsqueda y copia de nombre_producto y categoría_producto lo cual añade complejidad al sistema.

Conclusión final:

La normalización es un pilar fundamental para diseñar bases de datos eficientes y consistentes, sin embargo la desnormalización es necesaria en el mundo de la inteligencia de los negocios y los sistemas de análisis donde la velocidad de consultas es máxima prioridad por cuestiones empresariales, administrativas y económicas.

La decisión de desnormalización expresada y diseñada en este trabajo se justifica si es el sistema es predominantemente de lectura y la velocidad es máxima prioridad. También considerando que los datos duplicados son de baja volatilidad, es decir que son siempre los mismos. La clave está en aplicar esta técnica de forma medida y consciente, documentando las dependencias y evaluando el balance de rendimiento vs. costos, siempre en positivo.