Reflexión Individual – Adriana

1. ¿Qué criterios usaron para decidir qué entidades y relaciones debían formar parte del modelo?
   - Lo primero fue pensar cómo funciona una tienda de ropa en la vida real y qué información necesita guardar para trabajar bien. Tomamos de referencia marcas conocidas como Zara para guiarnos mejor. A partir de ahí, identificamos lo más importante: las prendas, los clientes, las ventas, el inventario y cómo se relacionan entre sí. Cada entidad se incluyó porque cumple un papel clave en el manejo del negocio, por ejemplo, saber cuántas unidades hay de una prenda, su categoría o si es para niño o adulto. También consideramos qué datos podían cambiar con el tiempo, como el precio o el stock, y por eso agregamos tablas para llevar esos historiales.

2. ¿Qué tan adecuadas fueron las claves primarias y foráneas que definieron en su diseño?

   - Todas las tablas tienen su clave primaria, y usamos claves foráneas para mantener bien conectada la información entre tablas. Desde el inicio tuvimos cuidado de no dejar nada sin relación. En algunos casos, las consultas terminaron siendo algo largas por la cantidad de joins que se necesitaban, pero era parte del diseño que queríamos para tener una base sólida. Lo bueno de esto es que evitamos errores, datos repetidos o que no tuvieran sentido.

3. ¿En qué medida aplicaron la normalización?

   - La normalización fue una prioridad. Creamos tablas separadas para datos que podían repetirse o tener múltiples valores, como los correos, direcciones y teléfonos de los clientes. También nos aseguramos de evitar dependencias innecesarias entre los atributos. Gracias a eso, el modelo se volvió más claro y escalable, a pesar de que aumentó el número de relaciones

4. ¿Qué restricciones y reglas del negocio implementaron directamente en la base de datos y por qué?

   - Se usaron varias restricciones como not null, check para asegurar que montos y cantidades fueran válidos, y unique para evitar datos duplicados, por ejemplo, en marcas o categorías. También se agregaron timestamps automáticos para registrar cuándo se insertaba cada dato. Los triggers jugaron un papel importante: uno registraba cambios de precio, otro calculaba el monto total en los detalles de ventas, y otro ajustaba el stock automáticamente. Todo esto se hizo para asegurar que la lógica del negocio no dependiera solo del código, sino también de la base de datos.

5. ¿Qué ventajas o desventajas identificas del modelo que construyeron al momento de hacer consultas complejas?

    - Una ventaja grande es la flexibilidad del modelo, ya que nos permitió hacer distintos reportes y aplicar filtros desde el frontend sin problema. Sin embargo, una desventaja es que algunas consultas resultaron bastante extensas y con muchos joins, lo cual complicaba el uso de group by y algunas condiciones. A pesar de eso, se logró que los endpoints respondieran bien y mostraran los datos correctamente.

6. ¿qué cambiarían en el diseño de la base de datos si tuvieran que escalar este sistema a un entorno de producción?
    - Una mejora sería llevar un control más claro de quién hace cada cambio, como guardar el usuario que modificó precios o stock. También sería útil usar vistas para simplificar las consultas más largas y procedimientos almacenados para operaciones comunes. Así se haría más fácil el mantenimiento y sería más seguro trabajar con más datos.