Sección 1: Conocimiento teórico

**1. Explique las diferencias entre los sistemas OLTP y OLAP. Proporcione ejemplos de**

**escenarios de uso y cómo optimizar cada tipo de sistema:**

OLTP que significa Online Transactional Processing por sus siglas en ingles, son modelos de bases de datos orientados a almacenar grandes volúmenes de pequeñas registros transaccionales tales como los registros de un banco, esto quiere decir que se va almacenando la informacion de la transacciones fila por fila por un numero limitado de columnas no variables. Un ejemplo basico son la bases de datos como Postgres y Mysql, y para poder optimizar las consultar es necesario limitarlas mediante una instrucción WHERE que trae un pequeño conjunto de datos donde extraerá la fila completa, incluso si solo se selecciona una columna.

Por otro lado, el sistema OLAP que significa Online Analytical Processing, se basa principalmente en columnas, especialmente diseñado para el análisis de datos y es extremadamente rápido ya que al hacer una consulta no trae todos los registro de una fila si esta no se especifica, por lo que para mejorar su rendimiento al hacer una consulta, solo se deben traer la columnas que se necesitan.

**2. Describa el concepto de normalización de datos. Incluya ejemplos de problemas**

**que pueden surgir al aplicar diferentes niveles de normalización y cómo**

**resolverlos.**

La normalización de datos se utiliza para eliminar la redundancia de los datos que se pueden presentar en las diferentes tablas que existen en un modelo,

### 1. Explique las diferencias entre los sistemas OLTP y OLAP. Proporcione ejemplos de escenarios de uso y cómo optimizar cada tipo de sistema

Los sistemas OLTP (Online Transaction Processing) y OLAP (Online Analytical Processing) tienen propósitos distintos y se optimizan de maneras diferentes. OLTP se centra en el procesamiento de transacciones pequeñas y frecuentes, como la compra de un producto, donde la concurrencia es alta y la escritura de datos es prioritaria. Para optimizar OLTP, se utiliza la normalización para minimizar la redundancia de datos y técnicas como el particionamiento de tablas para mejorar la eficiencia en la escritura. Los índices se emplean para acelerar las consultas más comunes. En contraste, OLAP se orienta al análisis de grandes volúmenes de datos, típicamente en escenarios donde se realizan pocas transacciones pero con un gran número de registros. Estos sistemas sacrifican la normalización en favor de la desnormalización, lo que permite una mayor eficiencia en las consultas. Las optimizaciones incluyen la creación de cubos multidimensionales, esquemas estrella o copo de nieve, y vistas materializadas, que permiten un acceso rápido y eficiente a la información agregada y desnormalizada, ideal para informes y análisis complejos.

### 2. Describa el concepto de normalización de datos. Incluya ejemplos de problemas que pueden surgir al aplicar diferentes niveles de normalización y cómo resolverlos.

La normalización es un proceso en el diseño de bases de datos que organiza los datos para reducir la redundancia y mejorar la integridad de los datos. Se estructura en diversas formas normales (1FN, 2FN, 3FN), cada una con reglas que definen cómo deben organizarse los datos. La 1FN asegura que los atributos son atómicos y que cada registro es único; la 2FN elimina dependencias parciales entre atributos; y la 3FN se enfoca en eliminar dependencias transitivas. Sin embargo, la normalización puede generar modelos de datos complejos, que pueden afectar el rendimiento debido a las numerosas relaciones entre tablas. Para mitigar estos problemas, se pueden crear índices y vistas materializadas que simplifiquen las consultas. En términos de consistencia, es crucial manejar inserciones, actualizaciones y eliminaciones a través de procedimientos almacenados para evitar inconsistencias en los datos.

### 3. Describa el concepto de desnormalización de datos. Incluya ejemplos de problemas que pueden surgir al aplicar diferentes niveles de normalización y cómo resolverlos.

La desnormalización es el proceso opuesto a la normalización y se utiliza para simplificar el modelo de datos y mejorar el rendimiento de las consultas en sistemas como bodegas de datos y OLAP. Al desnormalizar, se aumenta la redundancia de datos, lo que facilita el acceso rápido a la información, pero también incrementa el espacio de almacenamiento y complica el mantenimiento de los datos, especialmente durante actualizaciones y eliminaciones. Para mitigar estos problemas, se pueden usar procedimientos almacenados o triggers que garanticen que los datos en las tablas desnormalizadas se actualicen de manera consistente. Además, la desnormalización puede impactar negativamente en el rendimiento de escritura, por lo que es recomendable aplicar particionamiento de tablas, programar las escrituras durante períodos de baja carga, o mejorar el hardware para manejar eficientemente la carga adicional.

### 4. ¿Qué es un esquema de estrellas y cómo se diferencia de un esquema de copo de nieve? Diseñe un ejemplo concreto y compare las implicaciones de rendimiento y mantenimiento

Los esquemas estrella y copo de nieve son modelos dimensionales utilizados en bodegas de datos para estructurar la información. En un esquema estrella, las tablas de dimensiones se conectan directamente a una tabla de hechos central, lo que simplifica el diseño y mejora el rendimiento de las consultas, a costa de redundancia y desnormalización. Este esquema es más fácil de entender y utilizar, pero puede llevar a un mayor almacenamiento de datos redundantes. El esquema de copo de nieve, por otro lado, normaliza las tablas de dimensiones creando subtablas adicionales, lo que reduce la redundancia y mejora la consistencia de los datos, pero a costa de un diseño más complejo y de un rendimiento ligeramente inferior en las consultas debido a la necesidad de más uniones entre tablas.

### 5. Explique el concepto de un almacén de datos y cómo se diferencia de una base de datos. Describa cómo manejaría una migración de una base de datos transaccional a un almacén de datos.

Un almacén de datos (data warehouse) es un repositorio centralizado que integra grandes volúmenes de datos provenientes de diversas fuentes, y está diseñado para soportar análisis complejos, a menudo utilizando datos desnormalizados para mejorar el rendimiento. A diferencia de una base de datos transaccional, que se centra en el procesamiento eficiente de transacciones individuales y mantiene un alto nivel de normalización para garantizar la consistencia, un almacén de datos está optimizado para consultas de lectura intensiva. Al migrar una base de datos transaccional a un almacén de datos, se debe analizar la estructura de la base de datos, identificar las tablas de hechos y dimensiones, y diseñar procesos de ETL (Extracción, Transformación, Carga) que preparen los datos para su inclusión en el almacén, asegurando que se ajusten a los requerimientos de análisis y a la estructura del almacén de datos.

### 6. Describa el teorema de CAP y sus implicaciones para las bases de datos distribuidas

El teorema de CAP establece que en un sistema distribuido solo es posible garantizar dos de las tres propiedades siguientes: consistencia, disponibilidad y tolerancia a particiones. Esto significa que, en la práctica, se deben hacer concesiones al diseñar sistemas distribuidos. Por ejemplo, un sistema que prioriza la consistencia y la disponibilidad no podrá tolerar particiones, lo que podría no ser adecuado para entornos con alta latencia de red o fallos frecuentes. Por otro lado, un sistema que favorece la disponibilidad y la tolerancia a particiones podría no ser siempre consistente. Entender las necesidades del sistema y los posibles puntos de falla es crucial para decidir qué propiedades priorizar, asegurando así que el diseño del sistema cumpla con los requisitos de negocio y operativos.

### 7. ¿Cuáles son las propiedades de ACID en un sistema de base de datos? Proporcione ejemplos de cómo estas propiedades se aplican y se garantizan en bases de datos distribuidas

Las propiedades ACID (Atomicidad, Consistencia, Aislamiento, Durabilidad) son fundamentales para garantizar la integridad de las transacciones en bases de datos. La atomicidad asegura que una transacción se complete en su totalidad o no se ejecute, lo que es crucial para evitar estados inconsistentes. La consistencia garantiza que cada transacción mantiene las reglas de integridad de la base de datos, dejando la base en un estado válido. El aislamiento asegura que las transacciones no interfieran entre sí, mientras que la durabilidad asegura que los cambios realizados por una transacción completada persistan incluso ante fallos del sistema. En bases de datos distribuidas, implementar ACID puede ser más complejo debido a la necesidad de coordinar entre múltiples nodos. Protocolos como el de consenso o el uso de logs de transacciones son esenciales para garantizar estas propiedades.

### 8. Explique el término “ETL” y “ELT”.

ETL (Extracción, Transformación, Carga) y ELT (Extracción, Carga, Transformación) son enfoques utilizados para mover y transformar datos entre sistemas, especialmente en procesos de integración de datos y carga en almacenes de datos. En ETL, los datos se extraen de su fuente, se transforman en un área temporal (staging) para garantizar su calidad y luego se cargan en el sistema destino. Este enfoque es ideal cuando se desea minimizar el impacto en los sistemas fuente y destino, utilizando recursos de procesamiento externos. En ELT, los datos se extraen y cargan directamente en el sistema destino, donde se realiza la transformación, aprovechando la capacidad de procesamiento del destino. ELT es útil cuando se dispone de potentes recursos de procesamiento en el destino o cuando no se cuenta con un área temporal. La elección entre ETL y ELT depende de los recursos disponibles y de los requisitos específicos del proyecto.

### 9. ¿Qué es un lago de datos y cómo se diferencia de un almacén de datos Proporcione ejemplos de casos de uso.

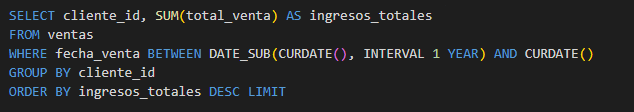
Un lago de datos (data lake) y un almacén de datos (data warehouse) son ambos repositorios para grandes volúmenes de datos, pero se diferencian en su estructura y propósito. Un lago de datos permite almacenar datos en su formato nativo, sean estructurados, semi-estructurados o no estructurados, lo que facilita su almacenamiento y procesamiento futuro para diferentes tipos de análisis. Un almacén de datos, en cambio, está más estructurado y optimizado para el análisis de datos a nivel empresarial, utilizando datos desnormalizados y agregados para mejorar el rendimiento de las consultas. Los casos de uso de un lago de datos incluyen almacenamiento de archivos multimedia, datos de sensores IoT, y registros de eventos, donde la flexibilidad del formato y la posibilidad de procesar datos en su estado bruto son críticas. Por otro lado, un almacén de datos es más adecuado para informes empresariales, análisis históricos y otras aplicaciones que requieren datos estructurados y bien organizados.

Sección 2: Habilidades prácticas

**A. Escritura de consultas SQL:**

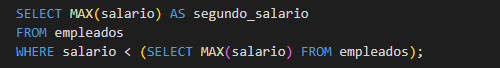
1. Escriba una consulta SQL para encontrar los 5 principales clientes por ingresos del

último año en la tabla de ventas.

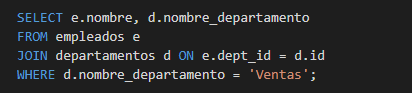


2. Escriba una consulta SQL para recuperar el segundo salario más alto de la tabla de

empleados.



3. Optimice una consulta SQL y explique las mejoras de rendimiento



Crear índices en las columnas dept\_id en empleados y id en departamentos mejorará el rendimiento.



La próxima vez en realizar la consulta, mejorará su rendimiento

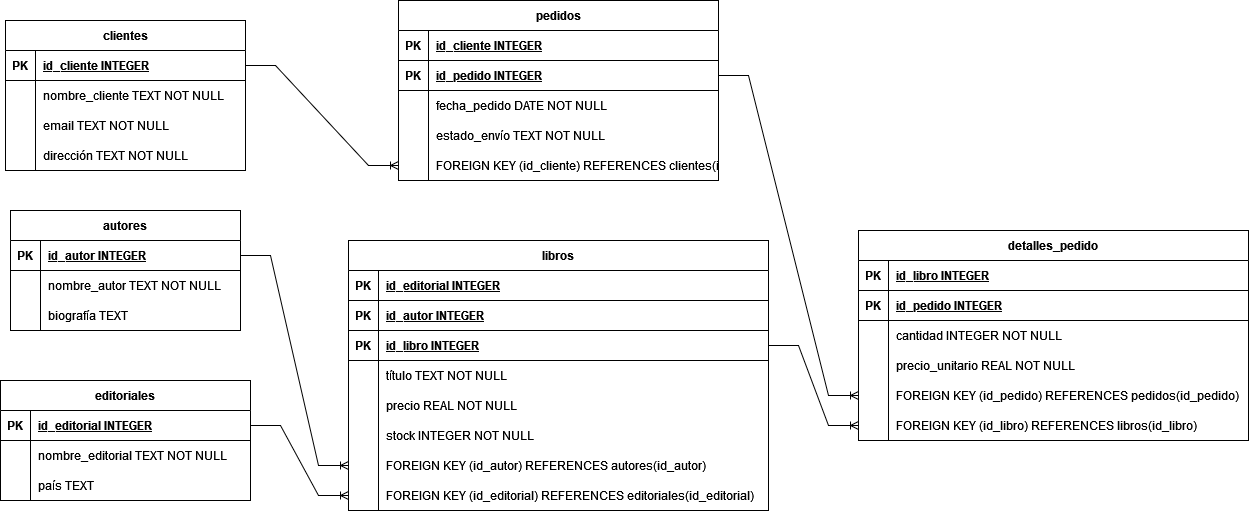
**B. Diseño de la base de datos:**

1. Diseñe un esquema de base de datos normalizado para una librería en línea. Incluya tablas para libros, autores, editores, clientes y pedidos. Nota: se aprecia en gran medida incluir requisitos funcionales adicionales como manejo de stock y seguimiento de envíos.

* libros: id\_libro (PK), título, id\_autor (FK), id\_editorial (FK), precio, stock.
* autores: id\_autor (PK), nombre\_autor, biografía.
* editoriales: id\_editorial (PK), nombre\_editorial, país.
* clientes: id\_cliente (PK), nombre\_cliente, email, dirección.
* pedidos: id\_pedido (PK), id\_cliente (FK), fecha\_pedido, estado\_envío.
* detalles\_pedido: id\_pedido (FK), id\_libro (FK), cantidad, precio\_unitario.

2. Explique sus decisiones de diseño y cree un diagrama de ER para el esquema diseñado.

Este diseño normalizado evita la redundancia y asegura la integridad de los datos. El uso de tablas intermedias como detalles\_pedido permite gestionar las relaciones muchos-a-muchos entre pedidos y libros.



**C. Modelado de datos:**

1. Desarrolle un esquema de estrellas para un almacén de datos de ventas al por menor. Identifique la tabla de hechos y las tablas de dimensiones, incluyendo múltiples tablas de hechos y dimensiones compartidas.

* Tabla de hechos:
  + ventas:id\_venta (PK),id\_tiempo (FK),id\_producto (FK),id\_cliente (FK),id\_tienda (FK),cantidad,total\_venta
  + devoluciones:id\_devolucion (PK),id\_tiempo (FK),id\_producto (FK),id\_cliente (FK),id\_tienda (FK),cantidad\_devuelta,monto\_devuelto
  + descuentos:id\_descuento (PK),id\_tiempo (FK),id\_producto (FK),id\_cliente (FK),id\_tienda (FK),porcentaje\_descuento,monto\_descuento
* Tablas de dimensiones:
  + tiempo:id\_tiempo (PK),fecha,mes,trimestre,año,día\_semana
  + productos:id\_producto (PK),nombre\_producto,categoría,precio,marca,id\_proveedor (FK)
  + clientes:id\_cliente (PK),nombre\_cliente,email,región,segmento
  + tiendas:id\_tienda (PK),nombre\_tienda,ubicación,tamaño,tipo\_tienda
  + proveedores:id\_proveedor (PK),nombre\_proveedor,país,tipo\_proveedor
  + empleados:id\_empleado (PK),nombre\_empleado,puesto,id\_tienda (FK),fecha\_contratación

Las tablas de hechos (ventas, devoluciones, descuentos) capturan eventos cuantificables como ventas realizadas, productos devueltos, y descuentos aplicados, respectivamente. Cada tabla se conecta con varias dimensiones para permitir análisis complejos. Las tablas de dimensiones (tiempo, productos, clientes, tiendas, proveedores, empleados) contienen datos descriptivos que permiten contextualizar los hechos. Estas dimensiones son compartidas entre las distintas tablas de hechos, lo que facilita análisis integrados.

**D. Transformación de datos:**

1. Proporcione un flujo de trabajo de muestra de ETL para extraer los datos de los clientes de una base de datos transaccional, transformarlos mediante la limpieza y la agregación, y cargarlos en un almacén de datos.

Ver el proceso ETL en la carpeta etl\_python/etl.py

Sección 3: Habilidades blandas y liderazgo

1. Describa un momento en el que tuvo que dirigir un equipo multifuncional para entregar un proyecto de datos. ¿A qué desafíos se enfrentó y cómo los superó?

En la anterior empresas donde trabajé, aunque no he tenido un perfil para dirigir grupos de trabajo, en una situación tuvimos que entregar un reporte que no se habían podido resolver, ya que el sistema que era legacy contenía bastante información y procesos, donde se perdia el origen de los datos donde teníamos que capturar la informacion. Para solucionarlo, me enfoqué en el trabajo en equipo y la experiencia de las personas que llevaban mas tiempo trabajando en el sistema ODI. Hicimos reuniones diarias para revisar los ETL y ajustar cualquier error. También me aseguré de que todos estuviéramos de acuerdo en la solución y entendiéramos el problema. Al final, logramos entregar el proyecto a tiempo, lo cual no hubiera sido posible sin ese esfuerzo de integración.

2. ¿Cómo se mantiene actualizado con las últimas tendencias y tecnologías en arquitectura de datos y gestión de datos? Explique cómo aplicó una tecnología o tendencia reciente en un proyecto real.

Pienso que la educación es uno de los factores importantes para estar en constante crecimiento profesional, por mi parte estudiar una maestría a complementado el desarrollo de software con los métodos de creación de modelos de inteligencia artificial mediante el machine learning y el análisis de datos, aunque también me gusta por medio de las redes sociales seguir a las personas influyentes que están a la vanguardia de la tecnología. Un ejemplo practico, fue la de aplicar un framework como Apache Beam en un proceso de conversaciones de un agente cognitivo con los clientes de claro para solucionar un problema de bloqueos de celulares, mediante computación en la nube, pudimos preprocesar muchísima información de las conversaciones en poco tiempo, lo que no se podía realizar con métodos convencionales.

3. Proporcione un ejemplo de un problema complejo que resolvió en su papel anterior como arquitecto de datos. Explique su enfoque y el resultado, incluyendo un problema técnico específico relacionado con arquitectura de datos.

Un reto memorable fue cuando tuve que optimizar el rendimiento de un sistema de procesamiento de datos en tiempo real. El sistema estaba sufriendo latencia debido al volumen de datos que procesaba. Mi enfoque fue dividir el flujo de datos en etapas, aplicando un enfoque de procesamiento en paralelo con Apache Beam. La idea era desacoplar los componentes y que cada uno manejara su parte del trabajo sin esperar a los otros. Tras implementar este cambio, la latencia se redujo significativamente, lo que permitió que el sistema procesara datos casi en tiempo real.

Sección 4: Codificación y secuencias de comandos

1. Escriba un script de Python para conectarse a una base de datos y recuperar datos de una tabla. Maneje errores y asegure conexiones. Incluir operaciones adicionales, como inserción y actualización de datos.

Ver el proceso ETL en la carpeta etl\_python/etl.py

2. Cree un script ETL simple utilizando un lenguaje de su elección (por ejemplo, Python, Java) para extraer datos de un archivo CSV, transformarlo eliminando duplicados y cargarlo en una base de datos. Incluya transformaciones más complejas, como mapeo de campos y agregaciones

Ver el proceso ETL en la carpeta etl\_python/etl.py