ACTIVIDAD 1

**Descripción del problema**

Examine el siguiente caso de estudio:

Caso de Estudio: **Optimización del Rendimiento Financiero**

Recientemente eres contratado como Administrador de Base de Datos (DBA) en "Finanzas Rápidas". Esta empresa maneja registros financieros de clientes, transacciones y balances. La base de datos utilizada para almacenar esta información ha estado experimentando ciertos problemas de rendimiento, y tu misión es mejorar la eficiencia para garantizar un acceso rápido a los datos.

Los empleados de "Finanzas Rápidas" se han quejado de que las consultas a la base de datos están tardando demasiado, afectando la productividad diaria. Además, han notado que algunas búsquedas no están devolviendo los resultados esperados, lo que genera confusiones en la interpretación de los registros financieros.

Realizaste un análisis inicial y descubriste que las consultas más lentas están relacionadas con la búsqueda de transacciones específicas y la generación de informes financieros.

Aplicaste el conocimiento de SELECT y WHERE para optimizar las consultas. Ajustaste las consultas para seleccionar solo la información necesaria y filtrar los resultados utilizando condiciones adecuadas.

***SELECT fecha, monto, concepto***

***FROM transacciones***

***WHERE cliente\_id = '123' AND tipo = 'Ingreso'***

***ORDER BY fecha DESC***

***LIMIT 10;***

Notaste que algunos registros tenían valores nulos, lo que afectaba las consultas. Utilizaste IS NOT NULL para excluir estos registros y garantizar la integridad de los resultados.

***SELECT \****

***FROM clientes***

***WHERE email IS NOT NULL;***

Encontraste que las búsquedas por nombre de cliente eran problemáticas. Implementaste el operador LIKE para realizar búsquedas más flexibles y evitar errores tipográficos.

***SELECT \****

***FROM clientes***

***WHERE nombre LIKE 'Juan%';***

Después de aplicar estas soluciones, observaste mejoras significativas en la velocidad de las consultas y la precisión de los resultados. Sin embargo, para garantizar un rendimiento continuo, decidiste explorar funciones de agregación para optimizar la generación de informes financieros.

Implementaste funciones de agregación como SUM y AVG para simplificar la generación de informes financieros que requerían cálculos sobre grandes conjuntos de datos.

***SELECT cliente\_id, SUM(monto) AS total\_ingresos***

***FROM transacciones***

***WHERE tipo = 'Ingreso'***

***GROUP BY cliente\_id;***

A través de la aplicación de conceptos básicos de base de datos y consultas SQL, lograste mejorar significativamente el rendimiento de la base de datos de "Finanzas Rápidas". Los tiempos de respuesta de las consultas se redujeron, y las búsquedas proporcionan resultados más precisos. Además, al incorporar funciones de agregación, optimizaste la generación de informes financieros, mejorando la eficiencia en el manejo de datos financieros de la empresa.

En resumen, este caso de estudio destaca la importancia de comprender y aplicar conceptos básicos de base de datos para resolver problemas prácticos en el entorno laboral. La optimización continua y la exploración de funciones más avanzadas aseguran que la base de datos pueda mantener un rendimiento eficiente a medida que evolucionan las necesidades de la empresa.

**Qué debo hacer**

Para desarrollar esta actividad, usted debe responder las preguntas de estudio con base en lo descrito en el caso y en las unidades de trabajo previas.

1)Después de recibir quejas sobre la lentitud de ciertas consultas en la base de datos, decides utilizar SELECT y WHERE para mejorar la eficiencia. Aplicas estas cláusulas para seleccionar información específica de transacciones, enfocándose en los ingresos de un cliente con el ID '123'. ¿Cuál es la mejor planificación para mejorar la eficiencia en la búsqueda de transacciones de ingresos para un cliente específico?  
  
**-Utilizar SELECT \* FROM transacciones WHERE cliente\_id = '123' AND tipo = 'Ingreso';  
  
-Aplicar SELECT fecha, monto, concepto FROM transacciones WHERE cliente\_id = '123' ORDER BY fecha DESC LIMIT 10;  
  
-Ejecutar SELECT \* FROM transacciones WHERE tipo = 'Ingreso' AND cliente\_id = '123';  
  
-Seleccionar todas las transacciones y filtrar en la aplicación.**

Aplicar SELECT fecha, monto, concepto FROM transacciones WHERE cliente\_id = '123' ORDER BY fecha DESC LIMIT 10; selecciona específicamente las columnas necesarias (fecha, monto, concepto) y filtra por el cliente '123'. Además, utiliza ORDER BY para ordenar los resultados por fecha de manera descendente y LIMIT 10 para limitar el número de resultados. Esto mejora significativamente la eficiencia de la consulta al traer solo la información relevante y reducir el volumen de datos procesados.

2) Durante tu análisis, identificas que algunos registros tienen valores nulos, afectando las consultas. Implementas IS NOT NULL para excluir estos registros y garantizar la integridad de los resultados, especialmente al consultar la información de los clientes.  ¿Cuál es la medida más efectiva para garantizar que las consultas excluyan registros con valores nulos en el campo de correo electrónico de los clientes?  
  
-**Ejecutar SELECT \* FROM clientes WHERE email IS NULL;**

**-Utilizar SELECT \* FROM clientes WHERE email != NULL;**

**-Aplicar SELECT \* FROM clientes WHERE email IS NOT NULL;**

**-Seleccionar todos los clientes y filtrar en la aplicación.**

La opción “Aplicar SELECT \* FROM clientes WHERE email IS NOT NULL” es la más adecuada ya que utiliza la cláusula IS NOT NULL para excluir registros con valores nulos en el campo de correo electrónico. Excluir registros con valores nulos garantiza la integridad de los resultados y evita posibles errores al consultar información de clientes. Esto contribuye a mantener la consistencia y confiabilidad de la base de datos.

3) Para mejorar las búsquedas por nombre de cliente, implementas el operador LIKE para realizar búsquedas más flexibles y evitar errores tipográficos. ¿Cuál es la mejor técnica para buscar clientes cuyo nombre comienza con "Juan"?

**-Utilizar SELECT \* FROM clientes WHERE nombre LIKE '%Juan%';**

**-Ejecutar SELECT \* FROM clientes WHERE nombre = 'Juan%';**

**-Aplicar SELECT \* FROM clientes WHERE nombre LIKE 'Juan%';**

**-Seleccionar todos los clientes y buscar en la aplicación.**

La opción “Utilizar SELECT \* FROM clientes WHERE nombre LIKE '%Juan%';” es la mejor técnica, ya que utiliza LIKE '%Juan%' para buscar cualquier nombre que contenga la cadena "Juan" en cualquier posición. Esto proporciona flexibilidad en la búsqueda y evita errores tipográficos al permitir que el nombre de Juan aparezca en cualquier parte del campo. Es una solución práctica y eficaz para mejorar la flexibilidad en las búsquedas de nombres.

4) Después de observar mejoras significativas, decides explorar funciones de agregación para optimizar la generación de informes financieros. Implementas funciones como SUM y AVG para simplificar cálculos sobre grandes conjuntos de datos. ¿Cuál es la mejor opción para calcular el total de ingresos por cliente a partir de transacciones de tipo "Ingreso"?

**-Ejecutar SELECT cliente\_id, monto FROM transacciones WHERE tipo = 'Ingreso';**

**-Utilizar SELECT cliente\_id, AVG(monto) AS total\_ingresos FROM transacciones WHERE tipo = 'Ingreso' GROUP BY cliente\_id;**

**-Aplicar SELECT cliente\_id, SUM(monto) AS total\_ingresos FROM transacciones WHERE tipo = 'Ingreso' GROUP BY cliente\_id;**

**-Seleccionar todas las transacciones y calcular en la aplicación.**

La opción “Aplicar SELECT cliente\_id, SUM(monto) AS total\_ingresos FROM transacciones WHERE tipo = 'Ingreso' GROUP BY cliente\_id;” es la estrategia más efectiva, ya que utiliza la función de agregación SUM para calcular el total de ingresos por cliente a partir de transacciones de tipo "Ingreso". Además, GROUP BY cliente\_id agrupa los resultados por cliente, permitiendo una presentación clara y eficiente de la información agregada. Esta técnica simplifica el proceso de generación de informes financieros y mejora la eficiencia en el manejo de grandes conjuntos de datos.