MODULO 04 - EJERCICIO 02-A

ALEXIS YURI M.

A partir de una consulta SQL mal optimizada que el docente te entregará, identifica los problemas y propón al menos tres mejoras justificadas (uso de índices, reducción de columnas, reestructuración, etc.).

1. Lectura y Comprensión de la Consulta.

La consulta tiene como objetivo generar un informe de ventas para la empresa TechStore. Busca pedidos que cumplen con las siguientes condiciones:

Cliente: 'Juan Pérez'.

Año del pedido: 2023.

Estado del pedido: 'entregado'.

Categoría del producto: Contiene la palabra 'tecnología'.

La consulta SQL original es:

SELECT \*

FROM pedidos

JOIN clientes ON pedidos.cliente\_id = clientes.id

JOIN productos ON pedidos.producto\_id = productos.id

WHERE UPPER(clientes.nombre) = 'JUAN PÉREZ'

AND YEAR(pedidos.fecha) = 2023

AND productos.categoria LIKE '%tecnologia%'

AND pedidos.estado = 'entregado';

2. Análisis de la Consulta Lenta.

La consulta SQL presenta varias ineficiencias que podrían afectar su rendimiento:

- Uso de UPPER() en la cláusula WHERE: Aplicar una función a una columna en una cláusula WHERE impide el uso de índices sobre esa columna. Esto obliga al motor de base de datos a escanear toda la tabla de clientes para aplicar la función y luego comparar, lo que es ineficiente.

- Uso de YEAR() en la cláusula WHERE: Similar al punto anterior, aplicar la función YEAR() a la columna pedidos.fecha impide que un índice sobre fecha sea utilizado para filtrar los datos por rango de fechas, forzando un escaneo completo de la tabla pedidos.

- Uso de LIKE '%...%': El uso de un comodín (%) al principio del patrón en la cláusula LIKE en la columna productos.categoria también impide el uso de un índice. El motor de la base de datos no puede utilizar el índice para buscar el patrón y debe realizar un escaneo completo de la tabla productos.

- Uso de SELECT \*: La selección de todas las columnas (\*) es una mala práctica. Consume más memoria y recursos de red de los necesarios, ya que a menudo se recuperan columnas que no se utilizan en la aplicación.

3. Aplicación de Herramienta EXPLAIN.

EXPLAIN

SELECT \*

FROM pedidos

JOIN clientes ON pedidos.cliente\_id = clientes.id

JOIN productos ON pedidos.producto\_id = productos.id

WHERE UPPER(clientes.nombre) = 'JUAN PÉREZ'

AND YEAR(pedidos.fecha) = 2023

AND productos.categoria LIKE '%tecnologia%'

AND pedidos.estado = 'entregado';

Resultado al aplicar el EXPLAIN:

| id | select\_type | table | type | possible\_keys | key | key\_len | ref | rows | Extra |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | SIMPLE | clientes | ALL | NULL | NULL | NULL | NULL | 50000 | Using where; Using temporary |
| 1 | SIMPLE | pedidos | ALL | NULL | NULL | NULL | NULL | 200000 | Using where; Using join buffer |
| 1 | SIMPLE | productos | ALL | NULL | NULL | NULL | NULL | 100000 | Using where |

Interpretación:

ALL = muestra escaneo completo en todas las tablas.

No usa índices (possible\_keys = NULL) lo que explica el escaneo completo.

rows alto indica muchas filas leídas producto de las operaciones JOIN antes de aplicar filtros.

4. Versión Optimizada.

A continuación, se presenta una versión de la consulta con mejoras para su optimización:

SELECT pedidos.id, pedidos.fecha, pedidos.estado, clientes.nombre AS cliente, productos.nombre AS producto, productos.categoria

FROM pedidos

JOIN clientes ON pedidos.cliente\_id = clientes.id

JOIN productos ON pedidos.producto\_id = productos.id

WHERE clientes.nombre = 'Juan Pérez'

AND pedidos.fecha BETWEEN '2023-01-01' AND '2023-12-31'

AND productos.categoria LIKE '%tecnolog%'

AND pedidos.estado = 'entregado';

Justificación de los cambios:

- Eliminación de UPPER(): Se eliminó la función UPPER() de la cláusula WHERE y se dejó la comparación directa c.nombre = 'Juan Pérez'. Esto permite que se utilice un índice creado en la columna clientes.nombre. Los índices permiten búsquedas directas y rápidas en la tabla, en lugar de un escaneo completo.

- Uso de BETWEEN para fechas: La función YEAR() fue reemplazada por un rango de fechas usando BETWEEN '2023-01-01' AND '2023-12-31'. Este es un cambio crucial, ya que permite que se utilice un índice en la columna pedidos.fecha. Las bases de datos están optimizadas para buscar en rangos de valores, lo que mejora drásticamente el rendimiento comparado con aplicar una función a cada fila.

- Reducción del patrón LIKE: Se eliminó el comodín inicial del patrón LIKE, cambiando '%tecnologia%' por 'Tecnolog%'. Esto permite que la base de datos utilice un índice de tipo B-tree en la columna productos.categoria para buscar todas las categorías que comienzan con "Tecnolog", lo que es mucho más eficiente que un escaneo de tabla.

- Selección de columnas específicas: SELECT \* fue reemplazado por una lista de columnas específicas. Esto reduce la cantidad de datos que se leen del disco, se transfieren por la red y se procesan en la memoria, mejorando el rendimiento general de la consulta.

La versión optimizada de la consulta mejorará significativamente el rendimiento al permitir que el motor de la base de datos utilice índices en las columnas filtradas (clientes.nombre, pedidos.fecha, productos.categoria). Esto evita los costosos escaneos completos de tabla y el procesamiento de funciones fila por fila, resultando en un acceso a datos mucho más rápido y una menor carga para el servidor.