Optimización de consultas a través de índices

Introducción:

En la gestión de bases de datos (SGBD), un índice es una estructura de datos diseñada para mejorar la velocidad de las operaciones de recuperación de datos en una tabla. La analogía más común es el índice de un libro: en lugar de leer un libro completo de principio a fin para encontrar un tema específico, se consulta el índice, que proporciona una lista ordenada de temas y los números de página exactos donde se encuentran.

Los índices representan una balanza entre costos y beneficios.[4]

Beneficio (Operaciones de Lectura): Como demostrará este análisis, los índices aceleran drásticamente las operaciones SELECT. Permiten al software gestor de bases de datos localizar un pequeño subconjunto de filas sin necesidad de examinar la tabla completa.

Costo (Operaciones de Escritura): Cada vez que se modifica un dato en una tabla mediante una operación INSERT, UPDATE o DELETE, el motor de la base de datos trabajar más para actualizar todas las estructuras de índice asociadas a esa tabla. Esto mantiene los índices sincronizados y ordenados, pero introduce una sobrecarga que puede ralentizar las operaciones de escritura, además, Los índices son estructuras físicas en disco y, por lo tanto, consumen espacio de almacenamiento adicional.

Aplicación práctica a nuestro proyecto de estudio “StudIA”

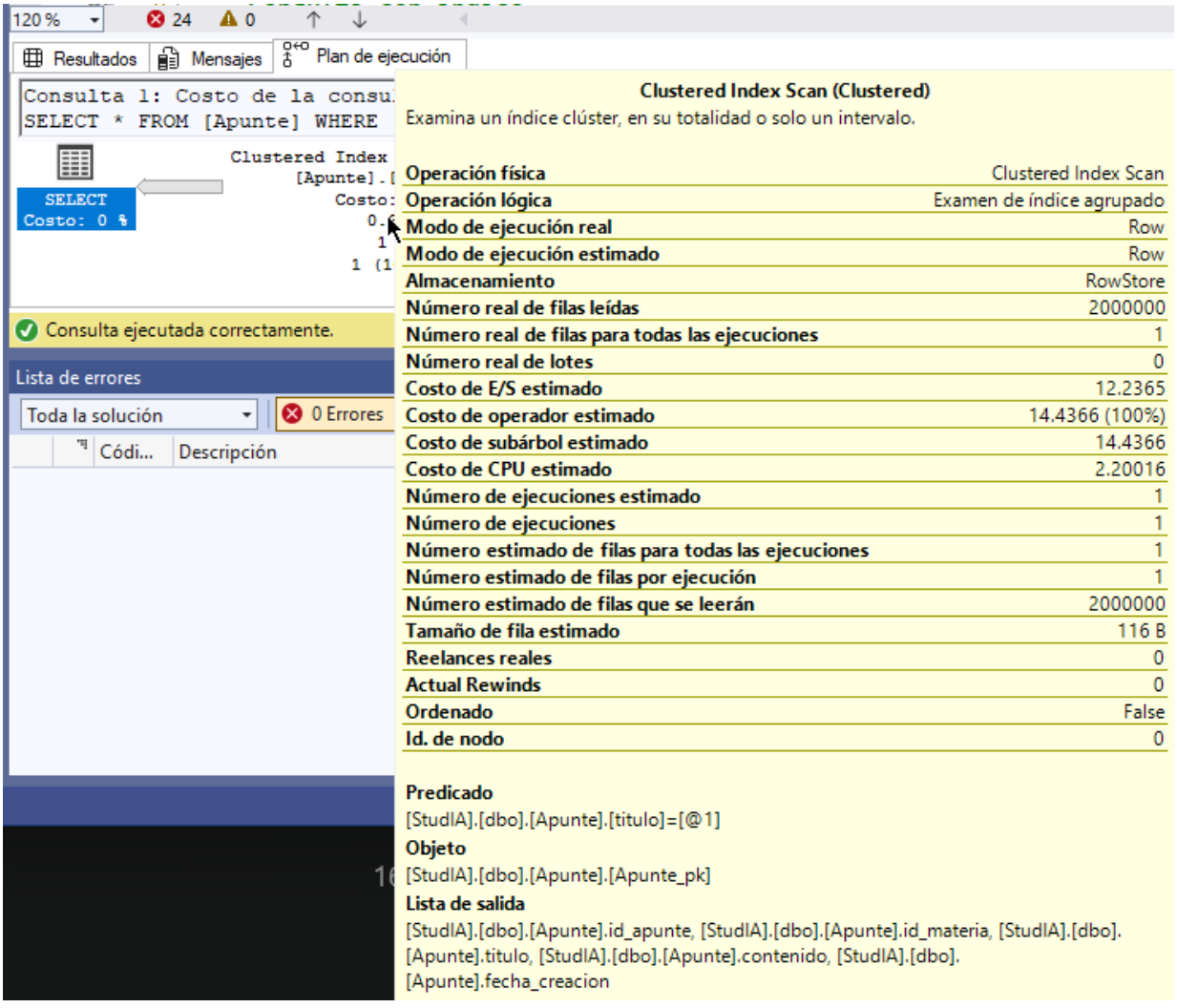
Se realizará un experimento practico en un caso de estudio ideal para observar la balanza entre costo-beneficio de la aplicación de índices.

Para ello, se poblaron las tablas Usuario, Materias y Apuntes, con un gran volumen de datos, para para simular un escenario de uso real.

El objetivo del experimento fue comparar el rendimiento de una consulta de búsqueda sobre la columna título en dos escenarios distintos:

1. **Escenario 1 (Sin Índice):** Una ejecución de consulta sin un índice específico que apoye la búsqueda por título.
2. **Escenario 2 (Con Índice):** La misma consulta lógica ejecutada después de crear un índice optimizado en la columna titulo.

Análisis del primer escenario. Búsqueda Ineficiente.



La primera ejecución representa el escenario "antes" de la optimización. El plan de ejecución muestra el intento del motor de base de datos de encontrar un registro específico basándose en la columna título sin un índice de ayuda.

El plan de ejecución muestra que el optimizador de consultas eligió la operación física Clustered Index Scan. El costo estimado de esta única operación es de 14.4366, este valor representa el 100% del costo total de la consulta. Este valor sirve como nuestro punto de referencia de ineficiencia.

índice agrupado

- A diferencia de otros tipos de índices, un índice agrupado define el orden físico en que las filas de datos de la tabla se almacenan en el disco.
- Debido a que los datos solo pueden estar ordenados físicamente de una manera, solo puede existir un índice agrupado por tabla.
- En SQL Server, cuando se define una Clave Primaria (Primary Key), esta crea por defecto un índice agrupado en esa(s) columna(s).

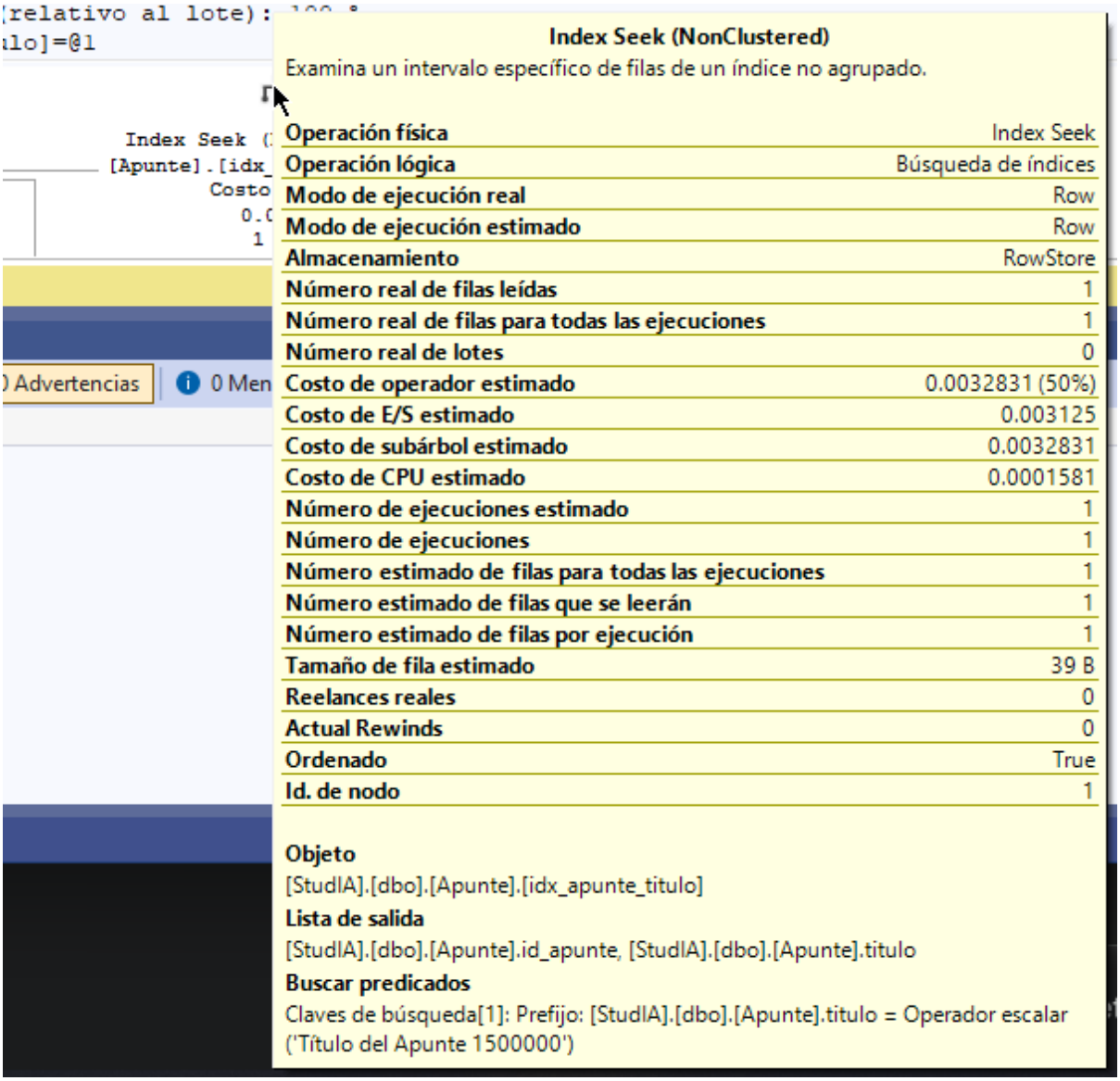
En la Imagen, se ve que se está examinando [Apunte].[Apunte_pk]. Esto confirma que el motor está utilizando la clave primaria, que es el índice agrupado de la tabla. Entonces, las filas de la tabla Apunte están físicamente ordenadas en el disco por su id_apunte, no por su título.

La operación Scan lee todas las filas de una estructura de datos de principio a fin. Si combinamos los conceptos de Índice agrupado y Scan obtenemos un Clustered Index Scan, que funcionalmente es un **Table Scan. El motor lee la tabla entera siguiendo el orden físico.**

Los datos del plan de ejecución confirman la ineficiencia:

Número real de filas leídas: 2,000,000. Este dato nos dice que para encontrar el apunte con un titulo específico, el motor tuvo que leer los dos millones de filas que contiene la tabla.

Análisis del Segundo escenario. Búsqueda Optimizada.



La segunda ejecución muestra el plan de ejecución después de haber creado un índice en la columna titulo (identificado como [Apunte].[idx_apunte_titulo]).

El plan de ejecución ha cambiado radicalmente. La operación ahora es un Index Seek (NonClustered). El costo estimado de esta operación es de 0.0032831, Comparando con el del escenario anterior (14.4366), la nueva operación es aproximadamente **4,400 veces más eficiente**.

Índice No Agrupado (Non-Clustered Index)

A diferencia del índice agrupado, un índice no agrupado es una **estructura separada** de los datos de la tabla.[6] Esta estructura contiene dos componentes clave:

1. Los valores de la(s) columna(s) indexada(s) (en este caso, titulo), almacenados en un árbol.
2. Un puntero que indica la ubicación de la fila de datos real a la que pertenece ese valor.

Seek

Una operación Seek es fundamentalmente diferente de un Scan. En lugar de leer toda la estructura, el motor utiliza la estructura de Árbol del índice no agrupado para navegar directamente a la(s) fila(s) que cumplen la condición.

El costo de la operación no depende del tamaño total de la tabla, sino de la "profundidad" del árbol de índice, que crece muy lentamente, por lo tanto, permite al motor encontrar cualquier valor con una cantidad mínima de lecturas.

Los datos de este plan validan la eficiencia:

- **Número real de filas leídas: 1.** Este es el resultado de la optimización. El gestor de BD utilizó la estructura ordenada del índice idx_apunte_titulo para encontrar instantáneamente la entrada 'Titulo del Apunte 1500000' y leyó solo esa entrada del índice.

El motor de SQL Server utilizó el índice idx_apunte_titulo para encontrar instantáneamente la ubicación del valor buscado. Esta operación fue casi instantánea y requirió leer solo una fila del índice.

Conclusiones

Los datos demuestran que la creación de un índice no agrupado en la columna titulo transformó la naturaleza de la consulta. Se pasó de una operación de fuerza bruta que lee toda la tabla, a una operación altamente eficiente, que lee solo los datos que necesita.

La investigación del Escenario muestra que, sin un índice de apoyo en titulo, el optimizador de SQL Server no tuvo más opción que recurrir a un costoso Clustered Index Scan, leyendo 2 millones de filas para encontrar un solo registro, un método insostenible en una aplicación real.

La investigación del Escenario 2 (Imagen 2) muestra que, con la adición de un índice no agrupado, el optimizador cambió su estrategia a un Index Seek, utilizando el índice como un mapa para localizar y leer una sola fila.

Para el proyecto "StudiA", esto implica que un diseño de base de datos exitoso no solo consiste en modelar correctamente las entidades y relaciones, sino también en analizar los patrones de consulta esperados, por ejemplo: ¿los usuarios buscarán apuntes por título?, ¿filtrarán por materia? Esta anticipación permite una estrategia de indexación inteligente que garantiza que la aplicación sea rápida, escalable y ofrezca una experiencia de usuario eficiente.

Fuentes consultadas

- IBM. (s.f.). Índices. Documentación de IBM Db2. https://www.ibm.com/docs/es/db2/11.5.x?topic=objects-indexes

- Microsoft Tech Community. (2006). *Scans vs. Seeks*. https://techcommunity.microsoft.com/blog/sqlserver/scans-vs-seeks/383115

- Microsoft Learn. (2025). Clustered and nonclustered indexes described. https://learn.microsoft.com/es-es/sql/relational-databases/indexes/clustered-and-nonclustered-indexes-described?view=sql-server-ver17

- SQL Shack. (s.f.). *¿Cuál es la diferencia entre Índices Agrupados y No Agrupados en SQL Server?*.
- https://www.sqlshack.com/es/cual-es-la-diferencia-entre-indices-agrupados-y-no-agrupados-en-sql-server/