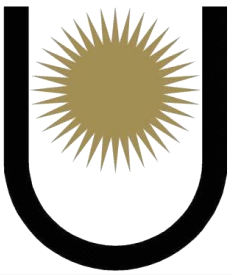
Optimización de consultas
a través de índices

2023



UNIVERSIDAD NACIONAL DEL NORDESTE

Profesor:
Villegas, Darío Oscar

Integrantes:
Acevedo, Joaquín Ignacio
Castillo, Zaira Salome
Leiva Falcón, Matías Gabriel
Medina, Andrea Marisol

Tabla de contenido

CAPÍTULO I: INTRODUCCIÓN

La *optimización de consultas a través de índices* se refiere a un proceso que busca mejorar el rendimiento de las consultas que se realizan en una base de datos. Las consultas son solicitudes de información que se hacen, y a medida que una base de datos crece en tamaño, la eficiencia en la recuperación de datos se convierte en un desafío importante.

Las *consultas* pueden ser simples o complejas y pueden involucrar la selección, filtrado, ordenamiento y agrupación de datos. La optimización de consultas se refiere a la mejora de la velocidad con la que estas consultas se ejecutan. Esto es esencial para garantizar que las aplicaciones y sistemas que dependen de la base de datos sean eficientes y respondan rápidamente a las solicitudes de los usuarios.

Los *índices* son estructuras de datos que se utilizan para acelerar la búsqueda y recuperación de información.

- **Objetivos Generales:**

  El objetivo general de este trabajo práctico es mostrar una forma de mejorar el rendimiento y la eficiencia de una base de datos mediante la optimización de consultas a través de índices. El resultado general esperado es lograr un sistema de gestión de bases de datos que ofrezca tiempos de respuesta más rápidos y un mejor rendimiento en la recuperación de datos, lo cual contribuirá a una experiencia de usuario más satisfactoria y a la mejora del funcionamiento de las aplicaciones que dependen de la base de datos.

- **Objetivos Específicos:**

  Diseñar e implementar índices adecuados: poder identificar las columnas clave en la base de datos y diseñar índices que aceleren la búsqueda y recuperación de datos relacionados con esas columnas.

CAPITULO II: MARCO CONCEPTUAL O REFERENCIAL

El marco conceptual o referencial de este trabajo se fundamenta en una serie de conceptos clave que son fundamentales para comprender el problema de la optimización de consultas a través de índices en bases de datos. A continuación, se presentan los conceptos y términos relevantes que sirven como base para este estudio:

Bases de Datos: Las bases de datos son sistemas de almacenamiento de información estructurada que juegan un papel fundamental en la gestión y acceso a datos en diversas aplicaciones y organizaciones.

Optimización de Consultas: La optimización de consultas se refiere al proceso de mejorar el rendimiento de las consultas a bases de datos, con el objetivo de acelerar la recuperación de datos y reducir el tiempo de respuesta.

Índices: Los índices son estructuras de datos que se utilizan para acelerar la búsqueda y recuperación de registros en una base de datos. Estos facilitan el acceso rápido a los datos al mantener una referencia ordenada de los valores de las columnas clave.

Rendimiento de Bases de Datos: El rendimiento de bases de datos se refiere a la capacidad de una base de datos para manejar consultas y transacciones de manera eficiente, minimizando los tiempos de respuesta y maximizando la capacidad de procesamiento.

Diseño de Índices: El diseño de índices implica la identificación de las columnas clave en una base de datos y la creación de índices adecuados para acelerar la recuperación de datos en función de los patrones de consulta.

Estructuras de Datos: Las estructuras de datos se refieren a las técnicas y algoritmos utilizados para organizar y acceder a los datos en una base de datos de manera eficiente, como árboles B, tablas hash, etc.

CAPÍTULO III: METODOLOGÍA SEGUIDA

En este capítulo se presenta el plan seguido o las acciones llevadas a cabo para realizar el trabajo, las dificultades encontradas, y cualquier otra información que proporcione la idea de cómo se realizó el trabajo.

- **Descripción de cómo se realizó el Trabajo Práctico.**

  La realización de este trabajo practico comenzó indagando recursos en línea, como ser artículos relacionados con el tema a tratar, las buenas prácticas, videos explicando cómo hacer uso e implementarlo a través del lenguaje SQL. Luego realizamos las consultas y creación de índices, nos basamos en el plan de ejecución para monitorear los tiempos de respuestas.

- **Herramientas (Instrumentos y procedimientos)**

  Sistema de Gestión de Bases de Datos (DBMS): Se utilizó un DBMS específico para el almacenamiento y gestión de datos, lo que incluyó la creación de tablas utilizando el modelo subido previamente al aula virtual, índices y consultas.

  Lenguaje SQL: Se utilizaron consultas SQL para diseñar índices, modificar consultas y realizar operaciones en la base de datos.

  Plan de ejecución: Monitoreo de rendimiento que permitió recopilar datos sobre el uso del sistema, los tiempos de respuesta de las consultas y las métricas de rendimiento.

CAPÍTULO IV: DESARROLLO DEL TEMA / PRESENTACIÓN DE RESULTADOS

Estructura Árbol B en la Implementación de Índices:

- Los índices en SQL Server (como los índices no clúster) se implementan utilizando la estructura de árbol B para permitir una búsqueda rápida y eficiente de datos.
- Cada nodo del árbol B almacena claves ordenadas y punteros a otras páginas o nodos hoja, lo que facilita la navegación hacia los datos buscados.

Los árboles B son cruciales en la implementación de índices en SQL Server, ya que permiten una búsqueda rápida y eficiente de datos a través de su estructura jerárquica y balanceada. Esta estructura es fundamental para mejorar el rendimiento de las consultas al reducir el número de comparaciones necesarias durante una búsqueda.

Tipos de índices

Montones o Heap:

Un montón es una tabla que no tiene un índice clúster. En las tablas almacenadas, se pueden crear uno o varios índices no clúster como un montón. Los datos se almacenan en el montón sin especificar un orden. Normalmente, en un principio los datos se almacenan en el orden en el que las filas se insertan en la tabla, pero el motor de base de datos puede mover los datos en el montón para almacenar las filas de forma eficaz, de modo que no se puede predecir el orden de los datos.

Características de los Montones o Heaps:

a) **Sin Orden Físico Definido**: En una tabla sin un índice agrupado, las filas de datos se almacenan en el orden en el que se insertan, sin un orden específico impuesto por un índice.
b) **Sin Clave de Ordenación Predeterminada:** La tabla sin índice agrupado no tiene una clave de ordenación predeterminada. Las consultas que no utilizan índices no tienen un orden garantizado para los resultados.
c) **Operaciones más Rápidas de Inserción:** Las operaciones de inserción en una tabla sin índice agrupado pueden ser más rápidas, ya que no hay necesidad de mantener un orden específico de las filas.

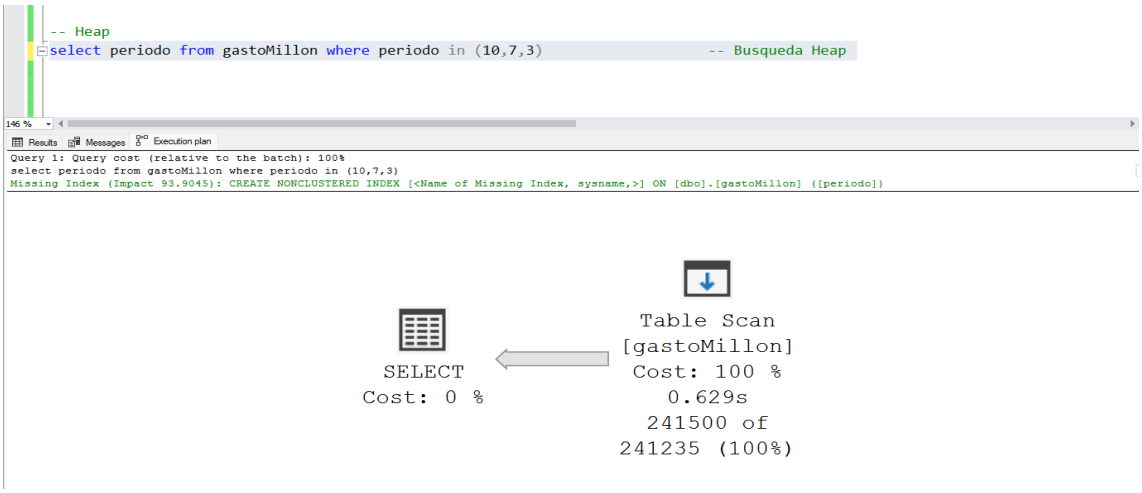
Ventajas y Desventajas de los Montones o Heaps:

Ventajas:

- Mejora el rendimiento de las operaciones de inserción
- Reducción de sobrecarga de mantenimiento de índices, ya que no hay un índice que requiera reorganización.

Desventajas:

- Las operaciones de búsqueda y consulta pueden ser más lentas, ya que no hay un orden garantizado para los datos.



-costo/tiempo de búsqueda: 0.629 segundos

Agrupado o Clustered:

Un índice clúster ordena y almacena las filas de datos de la tabla o vista por orden en función de la clave del índice clúster. El índice clúster se implementa como una estructura de árbol b que admite la recuperación rápida de las filas a partir de los valores de las claves del índice clúster.

Características de los Índices Agrupado o Clustered:

a) **Ordenación Física de los Datos:** El índice agrupado determina el orden físico de almacenamiento de los datos en la tabla. Las filas de datos se organizan directamente por la clave del índice.
b) **Estructura de la Tabla:** Una tabla puede tener solo un índice agrupado, ya que este define el orden físico de las filas. Por lo tanto, la elección de la clave del índice agrupado es crítica para el rendimiento.
c) **Búsqueda Rápida:** Dado que las filas de datos se almacenan en el orden del índice, las consultas que utilizan la clave del índice agrupado pueden ser más rápidas, ya que se reduce la necesidad de buscar en otros índices o la tabla en sí.

Ventajas y Desventajas de los Índices Agrupados:

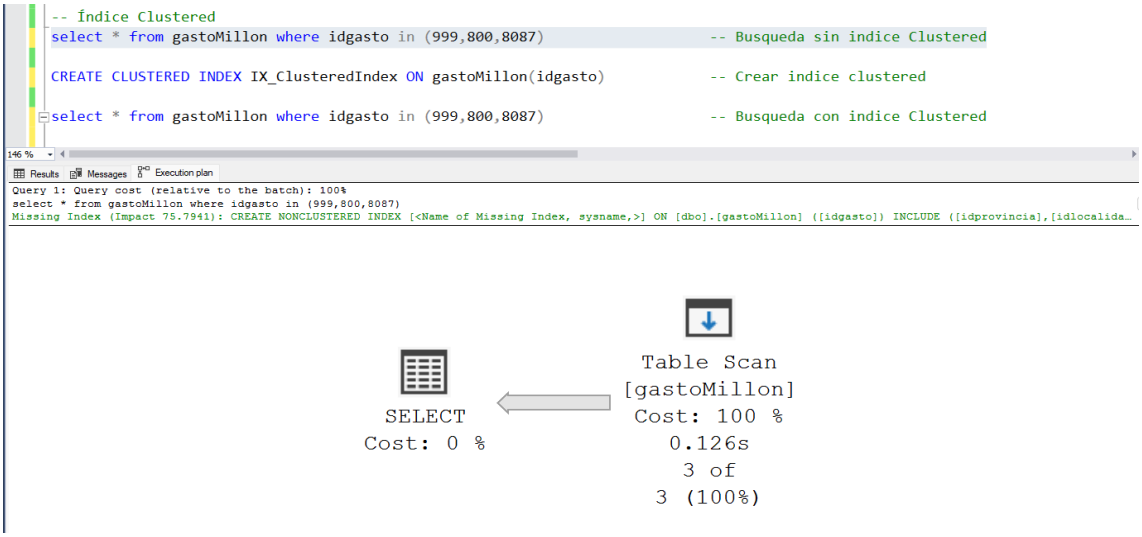
Ventajas:

- Mejora el rendimiento de consultas que utilizan la clave del índice agrupado.
- No se necesita una búsqueda adicional para obtener datos una vez que se ha localizado la fila a través del índice.
- Acelera la recuperación de rangos de valores basados en la clave de índice agrupado.

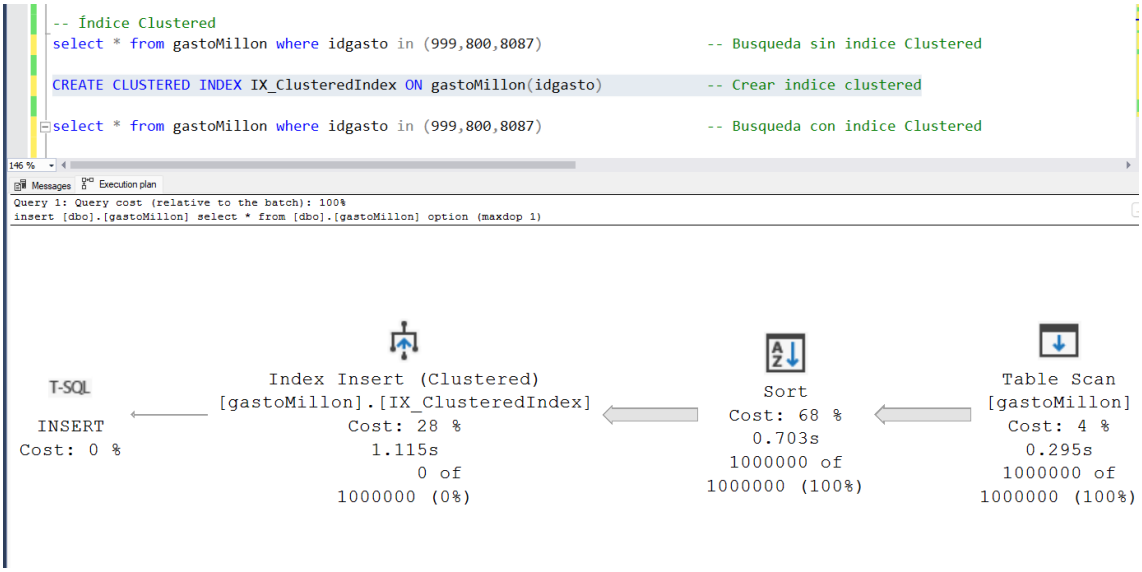
Desventajas:

- Las operaciones de inserción, actualización y eliminación pueden ser más lentas, ya que el motor de SQL Server debe reorganizar físicamente los datos para mantener el orden del índice agrupado.

- Puede haber fragmentación si hay inserciones, eliminaciones o actualizaciones frecuentes.



-costo/tiempo de búsqueda sin índice: 0.126 segundos

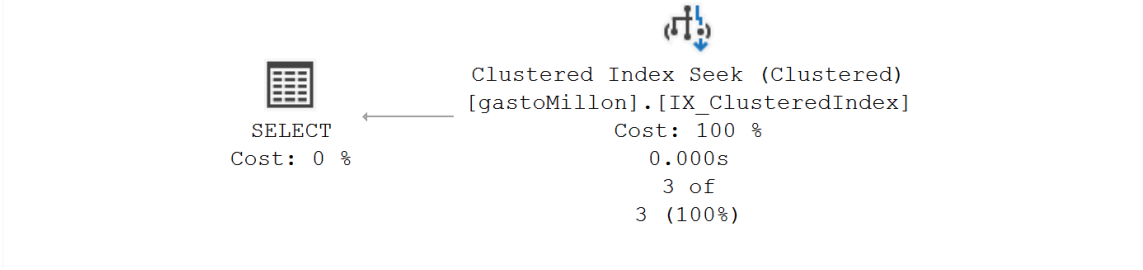


-costo/tiempo de creación de índice: 2.113 segundos

```
-- Índice Clustered
select * from gastoMillon where idgasto in (999,800,8087)                -- Busqueda sin indice Clustered

CREATE CLUSTERED INDEX IX_ClusteredIndex ON gastoMillon(idgasto)         -- Crear indice clustered

select * from gastoMillon where idgasto in (999,800,8087)                -- Busqueda con indice Clustered
```

146 % Results Messages Execution plan

```
Query 1: Query cost (relative to the batch): 100%
select * from gastoMillon where idgasto in (999,800,8087)
```



-costo/tiempo de búsqueda con índice: menos de 0.000 segundos

No agrupado/Non-Clustered:

Los índices no clúster se pueden definir en una tabla o vista con un índice clúster o en un montón. Cada fila del índice no clúster contiene un valor de clave no agrupada y un localizador de fila. Este localizador apunta a la fila de datos del índice clúster o el montón que contiene el valor de clave. Las filas del índice se almacenan en el mismo orden que los valores de la clave del índice, pero no se garantiza que las filas de datos estén en un determinado orden a menos que se cree un índice clúster en la tabla.

Características de los Índices No Agrupados:

a) **No Afecta el Orden Físico:** A diferencia de los índices agrupados, los índices no agrupados no organizan físicamente las filas de la tabla. En cambio, contienen una estructura de índice separada que apunta a las filas de datos.
b) **Estructura de Búsqueda Rápida:** Cada entrada en el índice no agrupado contiene la clave del índice y un puntero a la fila correspondiente en la tabla, lo que permite búsquedas rápidas.
c) **Múltiples Índices no Agrupados:** Una tabla puede tener múltiples índices no agrupados, lo que permite indexar diferentes columnas para mejorar el rendimiento de consultas específicas.

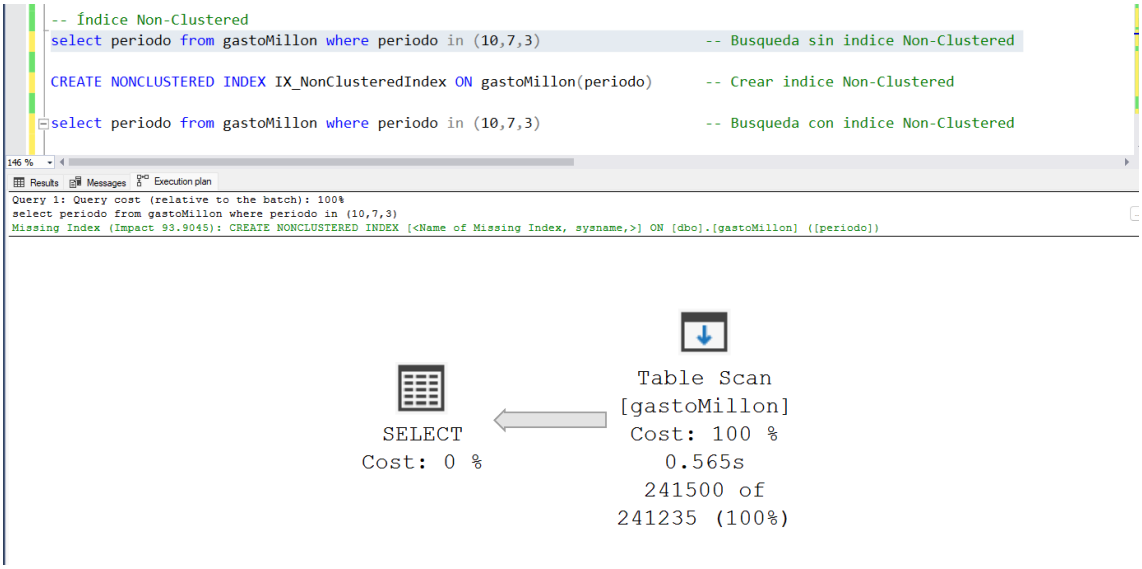
Ventajas y Desventajas de los Índices No Agrupados:

Ventajas:

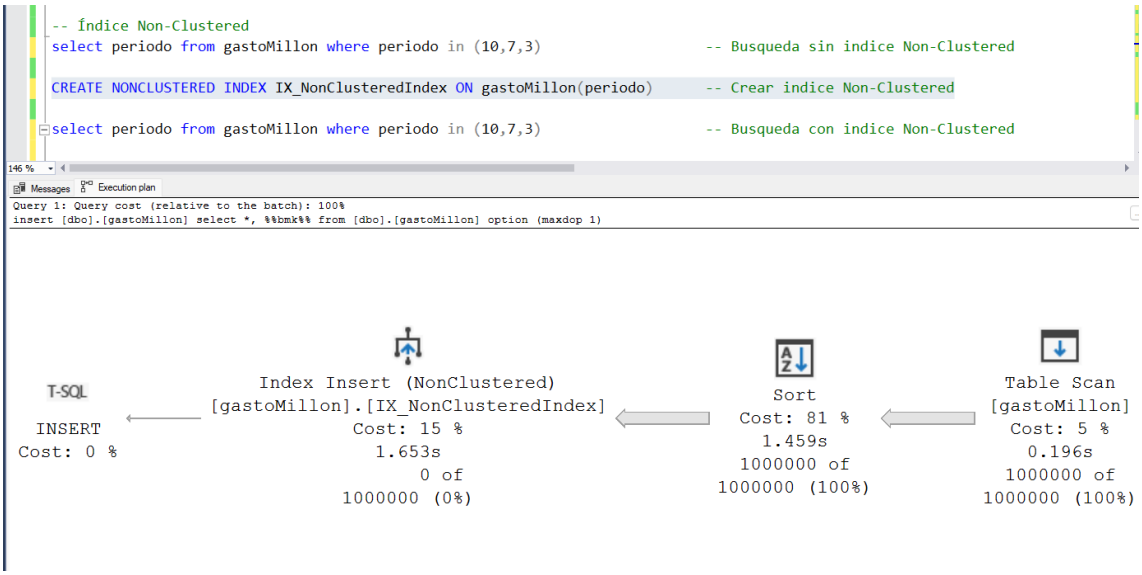
- Mejora el rendimiento de las consultas que utilizan columnas indexadas, ya que evita la necesidad de recorrer la tabla completa.
- No afecta el orden físico de los datos, por lo que las operaciones de inserción, actualización y eliminación pueden ser más rápidas que con los índices agrupados.

Desventajas:

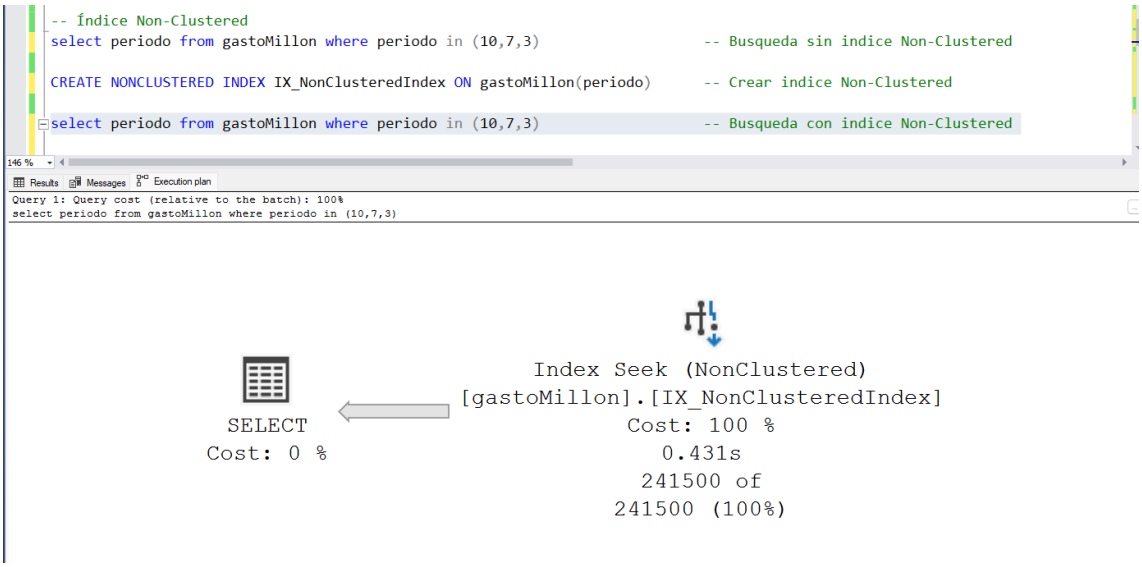
- Requiere una búsqueda adicional en la tabla después de la búsqueda en el índice para obtener los datos correspondientes.
- Puede ocupar más espacio en disco al almacenar la estructura del índice separada.



-costo/tiempo de búsqueda sin índice: 0.565 segundos



-costo/tiempo de creación de índice: 3.308 segundos



-costo/tiempo de búsqueda con índice: 0.431 segundos

Único/Unique:

Un índice único se asegura de que la clave de índice no contenga valores duplicados y, por tanto, cada fila de la tabla o vista sea en cierta forma única. La unicidad puede ser una propiedad tanto de índices clúster como de índices no clúster.

Características de los Índices Únicos:

a) **Unicidad de Valores:** Los índices únicos aseguran que los valores en la columna (o conjunto de columnas) indexada sean únicos, no permitiendo valores duplicados en esas columnas.
b) **Estructura Similar a Índices no Agrupados**: Los índices únicos se asemejan en estructura a los índices no agrupados, pero con la diferencia clave de garantizar la unicidad.
c) **Múltiples Índices Únicos:** Se pueden tener múltiples índices únicos en una tabla, permitiendo la imposición de restricciones de unicidad en varias columnas.

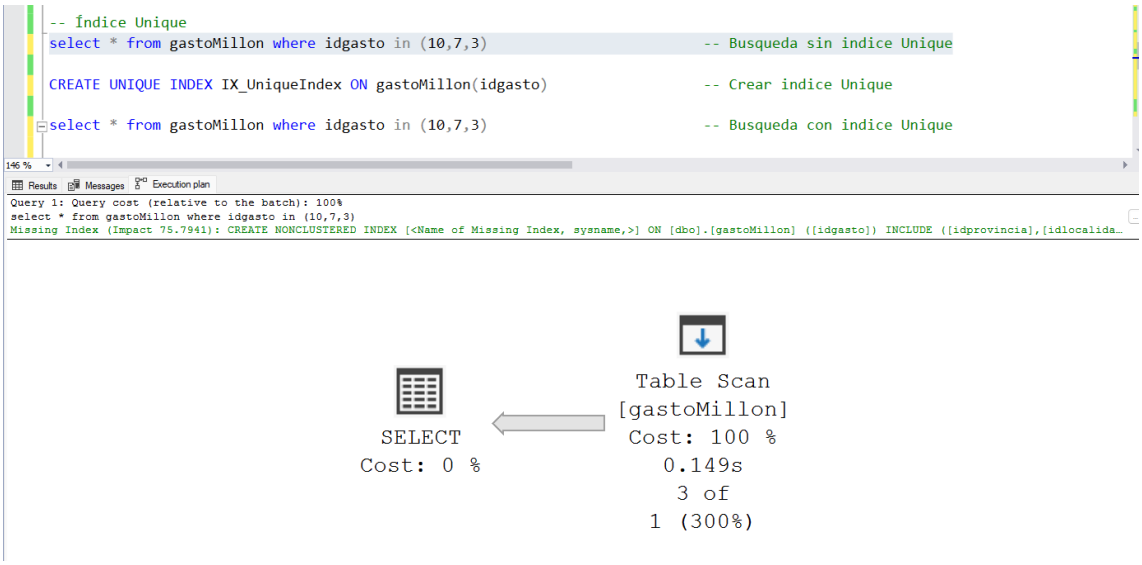
Ventajas y Desventajas de los Índices Únicos:

Ventajas:

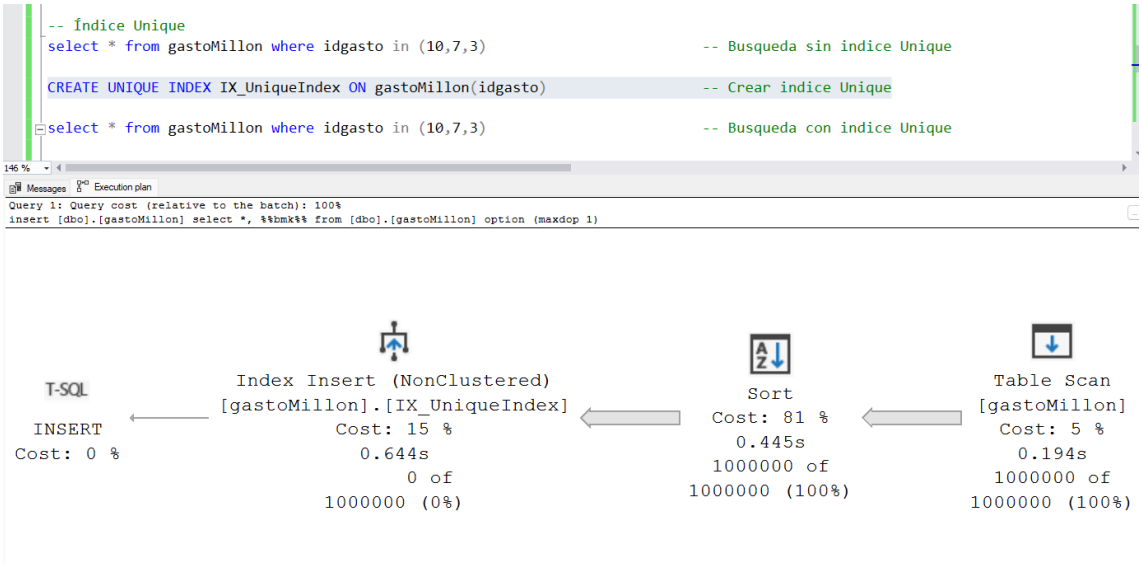
- Garantiza que los valores en la columna (o conjunto de columnas) indexadas sean únicos, lo que evita datos duplicados.
- Mejora la integridad de los datos y facilita la aplicación de reglas de negocio que requieren valores únicos.

Desventajas:

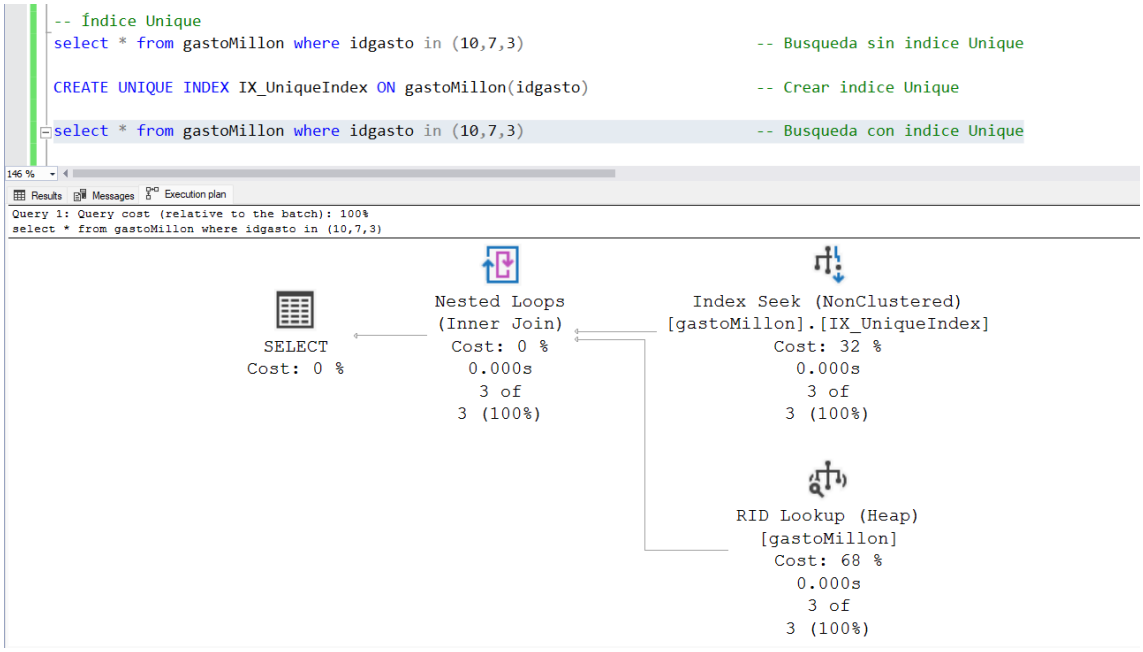
- Puede ralentizar la velocidad de inserción o actualización, ya que el motor de base de datos verifica la unicidad en el momento de la operación.
- Ocupa espacio adicional en disco debido a la estructura del índice.



-costo/tiempo de búsqueda sin índice: 0.149 segundos



-costo/tiempo de creación de índice: 1.283 segundos



-costo/tiempo de búsqueda con índice: menos de 0.000 segundos

<u>Índice con columnas incluidas/Index with included columns:</u>

Son una extensión de los índices no agrupados que permiten la inclusión de columnas adicionales no clave dentro de la estructura del índice. Estas columnas no clave incluidas son parte del índice, pero no se utilizan para ordenar o limitar la unicidad en el índice. En cambio, están disponibles para mejorar el rendimiento de consultas específicas. Las mejoras en el rendimiento se consiguen porque el optimizador de consultas puede localizar todos los valores de las columnas del índice, sin tener acceso a los datos de la tabla o del índice clúster, lo que da como resultado menos operaciones de E/S de disco.

Características de los Índices con Columnas Incluidas:

a) **Columnas Adicionales en el Índice:** Además de las columnas clave del índice, las columnas no clave pueden incluirse en la estructura del índice para mejorar el rendimiento de consultas.
b) **No se Utilizan para Ordenar o Filtrar:** A diferencia de las columnas clave, las columnas incluidas no se utilizan para ordenar datos ni para la unicidad del índice, pero están disponibles para mejorar la eficiencia de las consultas.
c) **Mejora del Rendimiento de Consultas Específicas:** Las columnas incluidas permiten la cobertura de consultas, lo que significa que los datos necesarios para la consulta están presentes en el índice, evitando búsquedas adicionales en la tabla de datos.

Ventajas y Desventajas de los Índices con Columnas Incluidas:

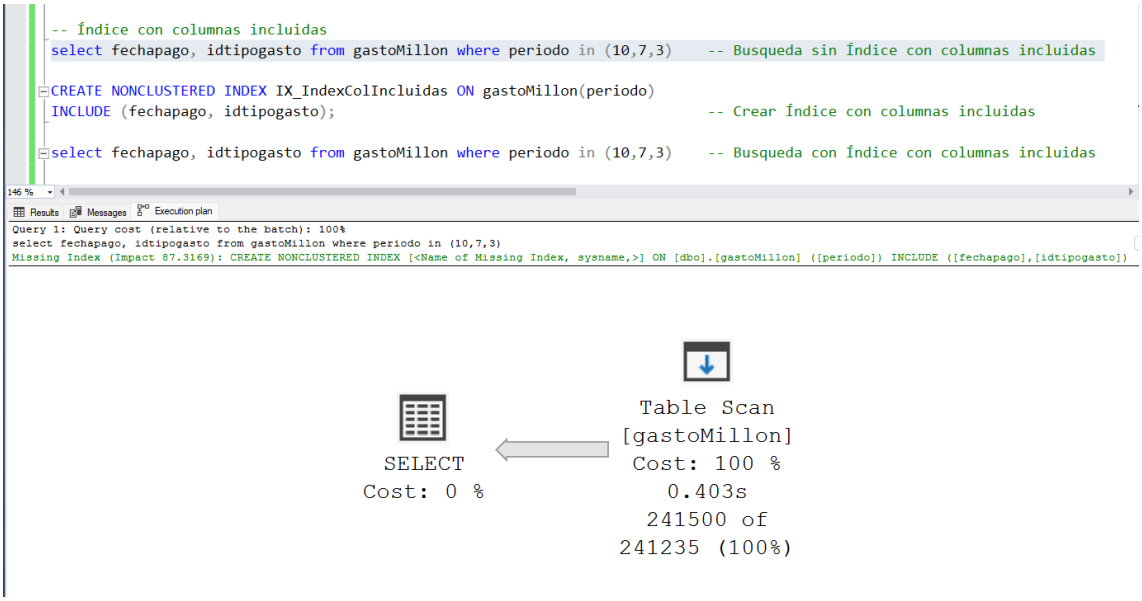
Ventajas:

- Mejora el rendimiento de consultas al proporcionar cobertura y evitar búsquedas adicionales en la tabla de datos.

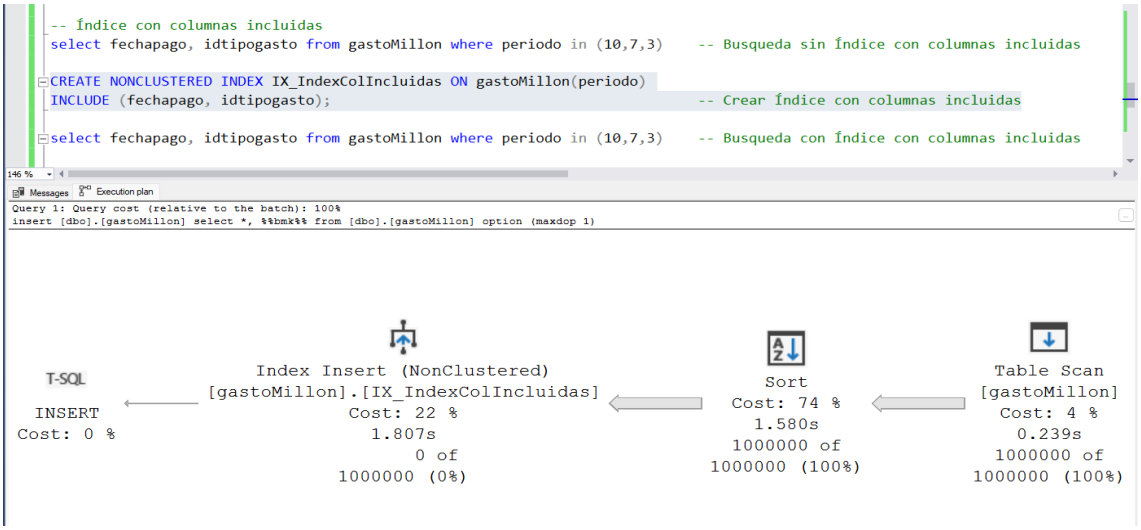
- Permite optimizar el rendimiento sin aumentar el número de índices, ya que las columnas incluidas son parte del índice no agrupado.

Desventajas:

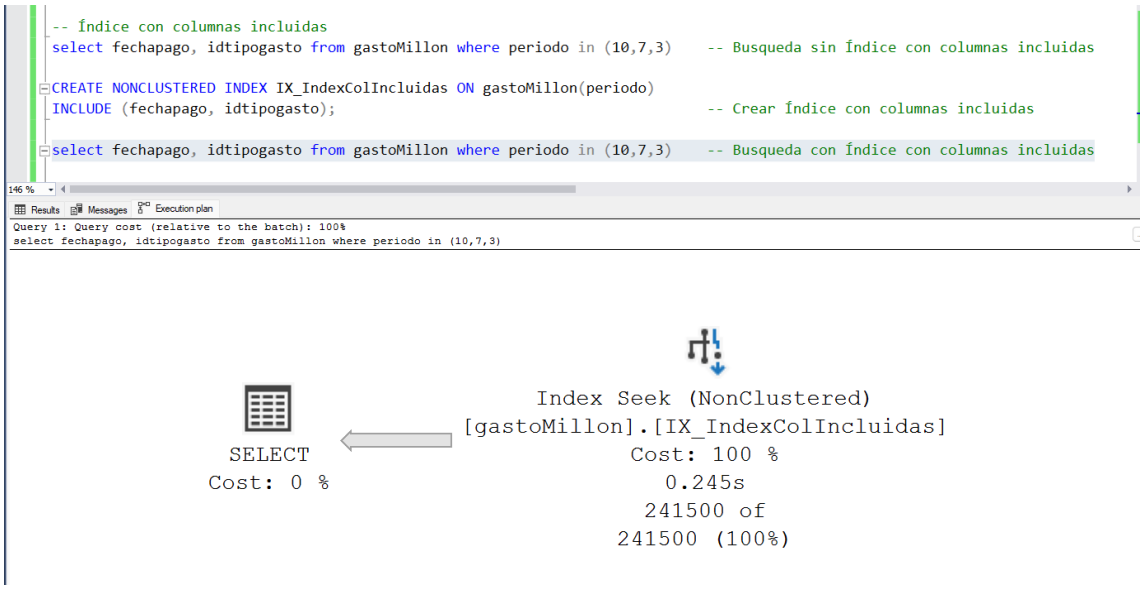
- Aumenta el tamaño del índice y, por ende, el uso de espacio en disco.
- No son adecuadas para todas las consultas, por lo que su uso debe ser estratégico.



-costo/tiempo de búsqueda sin índice: 0.403 segundos



-costo/tiempo de creación de índice: 3.626 segundos



-costo/tiempo de búsqueda con índice: 0.245 segundos

Filtrado/Filtered:

Índice no clúster optimizado, especialmente indicado para cubrir consultas que seleccionan de un subconjunto bien definido de datos. Este tipo de índice se crea utilizando una cláusula WHERE en su definición para filtrar qué filas de la tabla deben incluirse en el índice. Un índice filtrado bien diseñado puede mejorar el rendimiento de las consultas y reducir los costos de almacenamiento del índice en relación con los índices de tabla completa, así como los costos de mantenimiento.

Características de los Índices Filtrados:

a) **Subconjunto de Filas:** Los índices filtrados permiten seleccionar un subconjunto de filas basado en una condición WHERE, lo que significa que no todas las filas de la tabla estarán incluidas en el índice.
b) **Rendimiento Mejorado:** Al crear un índice filtrado, se pueden mejorar significativamente el rendimiento de consultas que usan las columnas especificadas en la cláusula WHERE del índice.
c) **Reducción del Tamaño del Índice:** Al indexar un subconjunto de filas, los índices filtrados pueden ser más pequeños y ocupar menos espacio en comparación con índices no filtrados.

Ventajas y Desventajas de los Índices Filtrados:

Ventajas:

- Mejora el rendimiento de consultas al indexar sólo el subconjunto necesario de filas.
- Reducción del tamaño del índice y uso de espacio en disco al almacenar menos datos.

Desventajas:

- Los índices filtrados pueden aumentar la complejidad del mantenimiento del esquema,

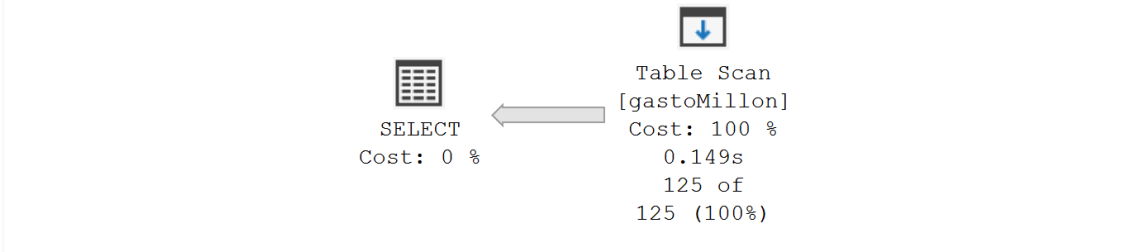
ya que se deben mantener y actualizar correctamente.

```
-- Índice Filtrado
select * from gastoMillon where importe = 79395.75                    -- Busqueda sin Índice Filtrado

CREATE NONCLUSTERED INDEX IX_FilteredIndex ON gastoMillon(importe)
WHERE importe > 50000;                                                -- Crear Índice Filtrado

select * from gastoMillon where importe = 79395.75                    -- Busqueda con Índice Filtrado
```

146 %

Results | Messages | Execution plan

```
Query 1: Query cost (relative to the batch): 100%
SELECT * FROM [gastoMillon] WHERE [importe]=@1
Missing Index (Impact 99.9344): CREATE NONCLUSTERED INDEX [<Name of Missing Index, sysname,>] ON [dbo].[gastoMillon] ([importe]) INCLUDE ([idgasto],[idprovincia],[…
```



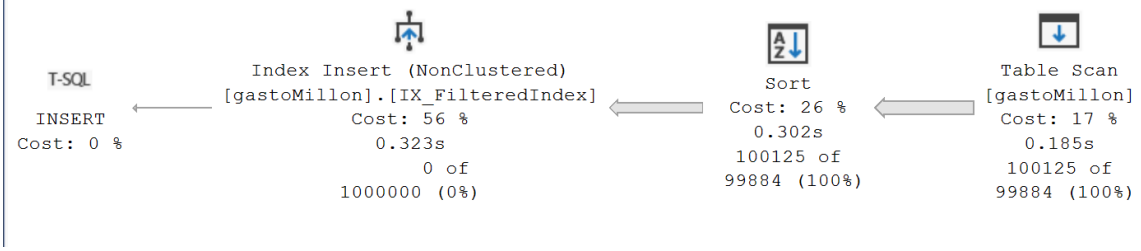
-costo/tiempo de búsqueda sin índice: 0.149 segundos

```
-- Índice Filtrado
select * from gastoMillon where importe = 79395.75                    -- Busqueda sin Índice Filtrado

CREATE NONCLUSTERED INDEX IX_FilteredIndex ON gastoMillon(importe)
WHERE importe > 50000;                                                -- Crear Índice Filtrado

select * from gastoMillon where importe = 79395.75                    -- Busqueda con Índice Filtrado
```

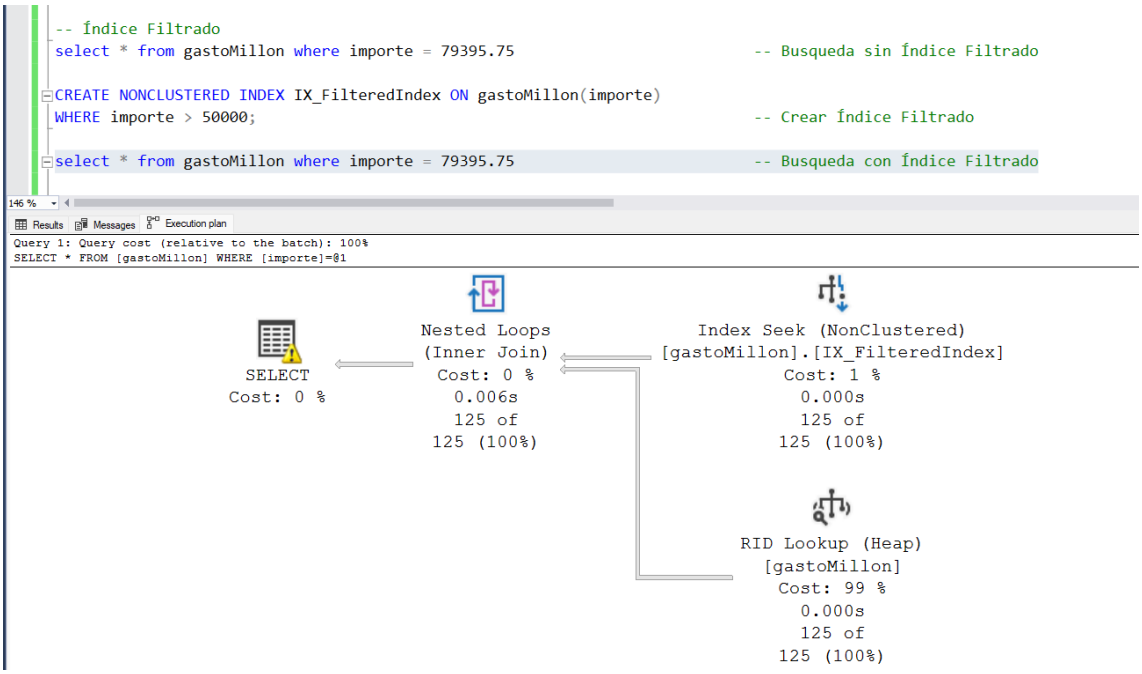
146 %

Messages | Execution plan

```
Query 1: Query cost (relative to the batch): 100%
insert [dbo].[gastoMillon] select *, %%bmk%% from [dbo].[gastoMillon] option (maxdop 1)
Missing Index (Impact 17.5792): CREATE NONCLUSTERED INDEX [<Name of Missing Index, sysname,>] ON [dbo].[gastoMillon] ([importe])
```



-costo/tiempo de creación de índice: 0.81 segundos



-costo/tiempo de búsqueda con índice: 0.006 segundos

CAPÍTULO V: CONCLUSIONES

Se puede observar que ciertos índices tienen tiempos y costos que a lo mejor su impacto en el rendimiento sea imperceptible o hasta incluso negativo, esto se debe a que la densidad y la selectividad de un índice en SQL Server son aspectos fundamentales que influyen significativamente en el rendimiento de las consultas. En resumen, la densidad y la selectividad impactan en la eficiencia de las búsquedas y en las decisiones del optimizador de consultas al elegir y utilizar los índices.

Densidad del Índice:

Una densidad alta, representada por una mayor proporción de valores únicos en el índice, tiende a mejorar el rendimiento. Permite una mayor selectividad en la búsqueda de datos, reduciendo la cantidad de filas que deben ser examinadas durante una consulta. Esto conduce a planes de consulta más eficientes y tiempos de respuesta más rápidos.

Por otro lado, una densidad baja (menor cantidad de valores únicos) puede resultar en un menor rendimiento, ya que las búsquedas necesitan examinar más filas, lo que puede llevar a planes de consulta menos eficientes y tiempos de respuesta más lentos.

Selectividad del Índice:

La selectividad está directamente relacionada con la densidad. Índices más selectivos (mayor densidad) tienden a ser más eficaces, permitiendo al optimizador de consultas seleccionar mejor qué índice utilizar y cómo ejecutar una consulta de manera eficiente.

En conclusión, la densidad y la selectividad del índice son fundamentales para la optimización del rendimiento en SQL Server. Un índice con una densidad y selectividad más alta permite planes de consulta más eficientes y tiempos de respuesta más rápidos, mientras que una densidad y selectividad más bajas pueden resultar en planes de consulta menos eficientes y tiempos de respuesta más lentos. Por lo tanto, es crucial diseñar e implementar índices con alta densidad y selectividad para mejorar el rendimiento de las consultas en la base de datos.

VI. BIBLIOGRAFÍA.

Microsoft.com
sqlshack.com