Serie de eBooks técnicos de SQL Server

Microsoft

Rendimiento de SQL Server: consultas más rápidas con SQL Server

Rendimiento de SQL Server: consultas más rápidas con SQL Server

Contenido

01
Introducción: El crecimiento más rápido de los datos exige un acceso más rápido

02
Consultas más rápidas con SQL Server

03
Rendimiento de las bases de datos

04
Rendimiento de las consultas

05
Herramientas y características adicionales para mejorar del rendimiento

06
Establecimiento de la norma de velocidad y rendimiento

¿Quiénes deben leer este eBook?

Este eBook está dirigido a arquitectos, administradores y desarrolladores de bases de datos que desean acelerar las funcionalidades de procesamiento de consultas para admitir las aplicaciones más exigentes controladas por datos. Al leer este eBook, aprenderá a aprovechar al máximo SQL Server, usando las funcionalidades avanzadas de procesamiento integradas entre las que se incluyen rendimiento in-memory, seguridad, análisis y flexibilidad. En este eBook se analizan herramientas y características como los índices de columnstore y el Procesamiento de consultas adaptable y se incluyen detalles técnicos sobre cómo poner en práctica estas funcionalidades.

La manera de administrar los datos puede marcar la diferencia para el negocio.

Ahora que los datos se factorizan en un número creciente de interacciones en todos lados, no solo resulta importante administrar ese enorme volumen de datos sino también aprovechar todo su potencial.

Esta necesidad brinda la oportunidad de modernizar las aplicaciones de su organización e impulsar la transformación digital con mejores análisis integrados. Al usar las funcionalidades de inteligencia empresarial más avanzadas, puede aprovechar al máximo la gran cantidad y variedad de datos que existen, acelerar la actividad empresarial gracias a un proceso de toma de decisiones más inteligente y una ejecución más rápida, y obtener una ventaja competitiva.

Microsoft SQL Server puede ayudarlo a lograr este objetivo a través de la mejor y más rápida plataforma disponible para sus aplicaciones y datos. SQL Server ofrece funcionalidades integradas críticas, incluidas:

- El mejor rendimiento in-memory del sector[1]
- Seguridad de confianza
- Análisis avanzado revolucionario en la base de datos
- Flexibilidad para ejecutar todos los recursos de datos en cualquier entorno con cualquier dato

[1] Gartner ha clasificado a Microsoft como un líder con la visión más completa y la mayor capacidad para ejecutar sistema de administración de bases de datos operativas durante tres años consecutivos. Blog de SQL Server, Three years in a row—Microsoft is a leader in the ODBMS Magic Quadrant (Tres años consecutivos: Microsoft es líder del Cuadrante mágico de ODBMS), 3 de noviembre de 2017.

SQL Server combina una velocidad mayor con más opciones para elegir. A llevar la potencia de SQL Server a Linux, contenedores basados en Linux y Windows, Microsoft le permite elegir los lenguajes de desarrollo, tipos de datos, entornos (locales o en la nube) y sistemas operativos que funcionan mejor en su situación específica.

IDC calcula que el volumen de la esfera de datos global sujeta a análisis de datos aumentará por un factor de

50 hasta llegar a 5,2 ZB en 2025.

Fuente: IDC, Data Age 2025, abril de 2017

SQL Server proporciona funcionalidades y características integradas que aceleran el rendimiento de análisis y el procesamiento de consultas para que su aplicación de base de datos funcione a máxima velocidad.

SQL Server posee varias comparativas de alto rendimiento para el procesamiento de transacciones con aplicaciones empresariales líderes:

Hewlett Packard Enterprise (HPE) anunció un nuevo récord mundial en el resultado de la comparativa TPC-H de 10 TB[1] con SQL Server 2017 y Windows Server 2016, lo que demuestra el liderazgo de SQL Server en cuanto a precio y rendimiento.

HPE también anunció el primer resultado de TPC-H de 3 TB[2] y demostró la potencia que SQL Server 2017 tiene para administrar cargas de trabajo de consultas de análisis, incluidos almacenamientos de datos.

SQL Server es líder indiscutible en las cargas de trabajo de procesamiento de transacciones en línea (OLTP). Lenovo anunció recientemente un nuevo récord mundial en el resultado de la comparativa TPC-E[3] con SQL Server 2017 y Windows Server 2016. En la actualidad, este es el mejor resultado de TPC-E tanto en términos de rendimiento como de precio/rendimiento.

SQL Server posee un récord mundial en el resultado de la comparativa TPC-H de 10 TB[4] (no agrupado) para SQL Server en Red Hat Enterprise Linux. ■

[1] Resultados no agrupados de TPC-H de 10 TB del 9 de noviembre de 2017.

[2] Resultados no agrupados de TPC-H de 3 TB del 9 de noviembre de 2017.

[3] Resultados de comparativa de TPC-E del 9 de noviembre de 2017.

[4] Resultados de comparativa de TPC-H en RHEL de abril de 2017.

Las bases de datos realizan tanto transacciones simples como complejas.

En función de los tipos y del nivel de complejidad que implica la realización de dichas transacciones, la cantidad de tiempo que una base de datos tarda en devolver resultados puede aumentar considerablemente. Cuando piensa en la optimización de una base de datos, es importante conocer los tipos de transacciones y el rendimiento que la base de datos tendrá que mantener para que se considere que tiene suficiente capacidad de respuesta.

SQL Server cuenta con varias características diseñadas para aumentar el rendimiento de las transacciones. El procesamiento de datos in-memory puede ahorrar una importante cantidad de tiempo con ciertos tipos de transacciones. Los índices de columnstore aprovechan el almacenamiento de datos basado en columnas para organizar grandes cantidades de información preparada para el análisis en un formato comprimido. Esto hace que las búsquedas sean extremadamente rápidas en comparación con las búsquedas de índices de árbol B que se realizan en el almacenamiento de datos basado en filas.

Datos in-memory

En términos generales, la tecnología de procesamiento de datos in-memory es más rápida porque ahorra los tiempos de lectura y búsqueda en el disco. El procesamiento de datos in-memory también elimina el tiempo de espera debido a que no hay bloqueo de simultaneidad, pero sin persistencia es vulnerable a la naturaleza transitoria de la RAM. Es decir, un apagón repentino puede provocar pérdida de datos.

SQL Server agregó funcionalidad para aprovechar lo mejor de la tecnología de procesamiento de datos in-memory a la vez que se mitigan los riesgos. Esto te ayuda a sacar el máximo partido de OLTP in-memory y a mejorar notablemente el rendimiento.

Algunos tipos de transacciones son candidatos ideales para OLTP in-memory. Cuando las transacciones son breves y abundantes se consigue el mayor aumento del rendimiento (especialmente si se procesa un porcentaje elevado de instrucciones INSERT). Esto incluye el registro de transacciones de ventas, informes de grandes cantidades de sensores remotos o de la Internet de las Cosas (IoT) o clics en anuncios, por nombrar algunos ejemplos.

Es posible combinar varias características de OLTP in-memory de SQL Server para optimizar el rendimiento según el tipo de datos que se esté procesando:

Tablas optimizadas para memoria. La clave para OLTP in-memory es el uso de tablas optimizadas para memoria, un tipo especial de estructura de datos de tabla que se crea en la memoria y no en disco. Se dice que las tablas optimizadas para memoria usan un enfoque optimista para el procesamiento de transacciones simultáneas porque se basan en el hecho de que las posibilidades de que dos operaciones UPDATE afecten a la misma fila de datos son muy escasas. Por tanto, una fila no se bloquea antes de su actualización. En su lugar, el sistema realiza una validación rápida en el momento de la confirmación y marca cualquier conflicto que pueda producirse, lo que ahorra unos preciosos milisegundos de tiempo de procesamiento.

Tablas no duraderas. En SQL Server, las tablas optimizadas para memoria son persistentes de manera predeterminada, aunque pueden configurarse para que sean no persistentes o no perdurables si el riesgo de pérdida de datos es aceptable. Se usan como almacenamiento temporal para los resultados de las consultas o para almacenamiento en caché.

Variables de tabla optimizadas para memoria. Esta característica es útil cuando una variable se declara como una tabla in-memory. Estas variables almacenan los resultados de las consultas de tal manera que es fácil pasarlos a otras instrucciones o procedimientos, como procedimientos almacenados compilados de forma nativa o interpretados (estos últimos se usan para las tablas basadas en disco).

Procedimientos almacenados compilados de manera nativa. Un procedimiento almacenado se compiló a código nativo y es capaz de tener acceso a tablas optimizadas para memoria. Los procedimientos almacenados se compilan durante la creación, lo que les da una ventaja sobre los procedimientos almacenados interpretados porque la detección de errores se realiza durante la creación, no en tiempo de ejecución. Se obtienen mayores aumentos de rendimiento cuanto más compleja es la lógica y más filas de datos procesa un procedimiento almacenado compilado de manera nativa. Obtenga más información sobre los procedimientos recomendados de uso de procedimientos almacenados compilados de forma nativa.

Funciones escalares definidas por el usuario compiladas de manera nativa. Estas funciones definidas por el usuario, también llamadas UDF, se compilaron a código nativo para conseguir una ejecución más rápida en las tablas optimizadas para memoria. Las UDF que se definen de esta manera solo se pueden ejecutar en tablas optimizadas para memoria y no en tablas tradicionales basadas en disco.

SQL Server mejora el rendimiento de las cargas de trabajo de OLTP in-memory al eliminar muchas de las limitaciones que presentaban las tablas y los procedimientos almacenados de las versiones anteriores del producto. Las características incluidas en la versión 2017 facilitan la migración de aplicaciones y aprovechan los beneficios de OLTP in-memory. Además, las tablas optimizadas para memoria ahora admiten cargas de trabajo de OLTP más rápidas y ofrecen mejor rendimiento como consecuencia de las operaciones paralelizadas.

SQL Server también ofrece estos beneficios:

Se eliminó la limitación de ocho índices en las tablas optimizadas para memoria. Ahora puedes crear tantos índices en las tablas optimizadas para memoria como en las tablas basadas en disco. Cualquier tabla basada en disco de la base de datos que antes no se podía migrar debido a esta limitación puede ahora estar optimizada para memoria.

La repetición del registro de transacciones de las tablas optimizadas para memoria ahora se hace en paralelo. Esto agiliza los tiempos de recuperación y aumenta considerablemente el rendimiento sostenido de la configuración de los grupos de disponibilidad AlwaysOn.

El rendimiento de la reconstrucción de índices (no agrupados) de árbol Bw para las tablas `MEMORY_OPTIMIZED` durante la recuperación de la base de datos se optimizó considerablemente. Esta mejora reduce de manera sustancial el tiempo de recuperación de la base de datos si se usan índices no en clúster.

`sp_spaceused` se admite ahora en tablas optimizadas para memoria. Muestra el número de filas, el espacio en disco reservado y el espacio en disco usado por una tabla, una vista indexada o una cola de Service Broker en la base de datos actual. También puede mostrar el espacio reservado y usado por toda la base de datos.

`sp_rename` se admite ahora en tablas optimizadas para memoria y módulos T-SQL compilados de manera nativa.

Los archivos de grupo de archivos optimizados para memoria ahora se pueden almacenar en Azure Storage. Se admite la copia de seguridad y restauración de archivos optimizados para memoria en Azure Storage.

Las tablas optimizadas para memoria admiten ahora columnas calculadas. El área expuesta de consultas en los módulos nativos se mejoró para incluir compatibilidad total con las funciones JSON. Ahora hay disponible compatibilidad nativa adicional para construcciones de consultas como `CROSS APPLY`, `CASE` y `TOP (N) WITH TIES`.

Más información: OLTP in-memory

Para que una aplicación pueda usar OLTP in-memory, se puede crear una tabla optimizada para memoria:

```
 CREATE TABLE SupportEvent
(
    SupportEventId   int NOT NULL
        PRIMARY KEY NONCLUSTERED,
    ...
)      WITH (
        MEMORY_OPTIMIZED = ON,
        DURABILITY = SCHEMA_AND_DATA);
```

La cláusula `MEMORY_OPTIMIZED = ON` identifica una tabla como optimizada para memoria y `SCHEMA_AND_DATA` especifica que se registrarán todos los cambios en la tabla y que los datos de la tabla se almacenan in-memory. Cada tabla optimizada para memoria debe contener al menos un índice.

➔ **Para obtener información detallada** sobre las tablas optimizadas para memoria, consulte Tablas con optimización para memoria.

También puede crear procedimientos almacenados compilados de manera nativa para tener acceso a datos de tablas optimizadas para memoria. A continuación se muestra una sintaxis de ejemplo:

```
CREATE PROCEDURE dbo.usp_add_kitchen @dept_id int, @kitchen_
count int NOT NULL
WITH EXECUTE AS OWNER, SCHEMABINDING, NATIVE_COMPILATION
AS
BEGIN ATOMIC WITH (
TRANSACTION ISOLATION LEVEL = SNAPSHOT, LANGUAGE = N'us_
english')

  UPDATE dbo.Departments
  SET kitchen_count = ISNULL(kitchen_count, 0) +
@kitchen_count
  WHERE id = @dept_id
END;
GO
```

Un procedimiento creado sin `NATIVE_COMPILATION` no se puede modificar para convertirlo en un procedimiento almacenado compilado de manera nativa.

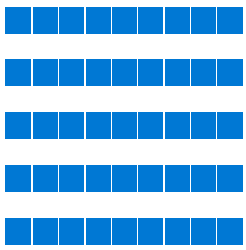
➔ **Para ver una explicación** de la programación en procedimientos almacenados compilados de manera nativa, el área expuesta de consulta admitida y los operadores, consulte Características admitidas en los módulos T-SQL compilados de manera nativa.

Índices de columnstore

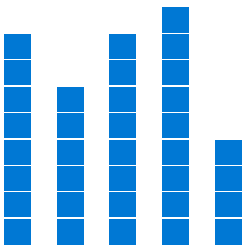
Un índice de columnstore es una tecnología para almacenar y consultar grandes cantidades de datos que tienen formato de columnas. Esta es una de las características más eficaces de SQL Server para las consultas de análisis de alta velocidad y bases de datos de gran tamaño. Los índices de columnstore mejoran el rendimiento al comprimir los datos en columnas para reducir el uso de memoria y disco, filtrar automáticamente los escaneos mediante la eliminación del grupo de filas y procesar las consultas por lotes. Con los índices de columnstore de SQL Server, puede habilitar el análisis operativo; es decir, la capacidad de ejecutar análisis en tiempo real con una carga de trabajo transaccional.

SQL Server ofrece varias funcionalidades para los índices de columnstore, incluidas:

- Recompilación en línea de índices de columnstore no en clúster (NCCI).
- Compatibilidad con objetos grandes (LOB) para los índices de columnstore.
- Columnas calculadas para el índice de columnstore (CCI).
- Características de optimizador de consultas, como Machine Learning Services.



Datos que se almacenan como filas



Datos que se almacenan como columnas

Más información: índices de columnstore

Un índice de columnstore es clúster o no clúster. Un índice de columnstore en clúster (CCI) es el almacenamiento físico de toda la tabla. Use un CCI para almacenar tablas de hechos y tablas de grandes dimensiones en cargas de trabajo de almacenamiento de datos. Un índice de columnstore no en clúster (NCCI) es un índice secundario que se crea en una tabla de almacenamiento de filas. Use un NCCI para realizar análisis en tiempo real en una carga de trabajo OLTP. Un índice de columnstore en clúster puede tener índices de árbol B no en clúster.

➔ **Para obtener más información** sobre los CCI y NCCI, visite Introducción a los índices de columnstore.

Más información: índices de columnstore para el almacenamiento de datos

Los CCI son los más adecuados para las consultas de análisis, porque en este tipo de consultas se suelen realizar operaciones sobre grandes intervalos de valores en lugar de buscar valores específicos. Al crear una tabla con la instrucción `CREATE TABLE`, puede designar la tabla como un CCI si se crea una opción `CLUSTERED COLUMNSTORE INDEX`:

```
--Create the table
CREATE TABLE t_account (
    AccountKey int NOT NULL,
    AccountDescription nvarchar (50),
    AccountType nvarchar(50),
    UnitSold int
);
GO
--Store the table as a columnstore.
CREATE CLUSTERED COLUMNSTORE INDEX taccount_cci ON t_account;
GO
```

Puede crear índices de árbol B no en clúster como índices secundarios en un CCI. Para optimizar las búsquedas de tablas en un almacenamiento de datos, puede crear un NCCI diseñado para ejecutar consultas que funcionan mejor con estas búsquedas:

```
CREATE NONCLUSTERED COLUMNSTORE INDEX taccount_nc1 ON t_
account (AccountKey);
```

Más información: columnstore para procesamiento híbrido de transacciones y análisis (HTAP)

El procesamiento híbrido de transacciones y análisis (HTAP) usa un índice de columnstore en una tabla de almacenamiento de filas que se puede actualizar. El índice de columnstore mantiene una copia de los datos, por lo que las cargas de trabajo de OLTP y análisis se ejecutan en copias aisladas de los datos. De esta forma se puede realizar procesamiento de análisis en tiempo real sobre cargas de trabajo de datos transaccionales sin que el rendimiento disminuya. Para cada tabla, quite todos los índices de árbol B que estén diseñados principalmente para acelerar el análisis existente en la carga de trabajo OLTP. Reemplázalos por un solo índice de columnstore. Consulte el ejemplo siguiente para crear un índice de columnstore no en clúster en la tabla OLTP con una condición de filtrado:

```
CREATE TABLE t_account (
    accountkey int PRIMARY KEY,
    accountdescription nvarchar (50),
    accounttype nvarchar(50),
    unitsold int
);
--Create the columnstore index with a filtered condition
CREATE NONCLUSTERED COLUMNSTORE INDEX account_NCCI
ON t_account (accountkey, accountdescription, unitsold)   ;
```

El índice de columnstore en una tabla in-memory te permite usar análisis operativo al integrar las tecnologías OLTP in-memory y columnstore in-memory; de esta forma, se consigue un rendimiento elevado para ambas cargas de trabajo. El índice de columnstore de una tabla in-memory debe incluir todas las columnas. En este ejemplo, se crea una tabla optimizada para memoria con un índice de columnstore:

```
CREATE TABLE t_account (
    accountkey int NOT NULL PRIMARY KEY NONCLUSTERED,
    Accountdescription nvarchar (50),
    accounttype nvarchar(50),
    unitsold int,
    INDEX t_account_cci CLUSTERED COLUMNSTORE
)
    WITH (MEMORY_OPTIMIZED =
ON );
GO
```

Con la creación de este índice, puedes implementar HTAP sin necesidad de hacer cambios en la aplicación. Las consultas de análisis se ejecutarán en el índice de columnstore y las operaciones OLTP seguirán ejecutándose en tus índices de árbol B de OLTP.

Recompilación en línea de índices de columnstore no en clúster

Los índices de columnstore son un componente importante para que el rendimiento de las consultas siga siendo eficiente. Para mantener el rendimiento, hay que reconstruir estos índices periódicamente; en el caso de índices muy grandes, esta operación puede tardar mucho tiempo. Para cualquier negocio en línea, el hecho de que una aplicación no esté operativa durante una hora significa una pérdida real de dinero y daños posiblemente aún más graves. Para que tu aplicación funcione sin interrupciones, SQL Server admite copias de seguridad en línea, comprobaciones de coherencia y reconstrucciones de índices.

En SQL Server, puedes pausar la construcción de un índice y reanudarla en cualquier momento, incluso después de que se produzca un error. Puede recompilar índices mientras están en uso y también puedes pausar y reanudar esas recompilaciones, siguiendo exactamente en el lugar donde se quedaron. Puede disfrutar de los beneficios que supone usar menos espacio de registro que en las operaciones de reconstrucción de índices de las versiones anteriores. ■

Las consultas mal escritas pueden degradar el rendimiento de la aplicación y limitar la disponibilidad de información crítica para el negocio.

También puede llevar a un uso ineficiente de recursos como la CPU, la memoria y la red. Las regresiones en los planes de ejecución de consultas también pueden afectar el rendimiento de manera considerable. Estas regresiones pueden producirse si ha habido cambios en la aplicación, las estadísticas de base de datos están obsoletas o las estimaciones de recuento de filas no son exactas. Aunque el hardware del servidor de bases de datos sea el más potente que existe, su rendimiento puede verse afectado negativamente por unas cuantas consultas que se comportan mal. De hecho, incluso una mala consulta puede causar problemas de rendimiento graves en la base de datos.

El rendimiento de las consultas depende de muchos factores, uno de los cuales es el plan de consulta. Cuando se ajuste y optimice una consulta que presenta un rendimiento bajo, un DBA suele empezar examinando el plan de ejecución de esa consulta y evaluándolo para determinar cuál es el mejor y más eficaz plan según la estimación de datos. Para ayudar en este proceso, SQL Server proporciona características de rendimiento y procesamiento de consultas que cambian el funcionamiento de los planes de consulta:

Las mejoras realizadas en **Query Store** le permiten hacer seguimiento de la información de resumen de las estadísticas de espera, lo que ayuda a reducir el tiempo dedicado a la solución de problemas.

Procesamiento de consultas adaptable (AQP) es una manera de optimizar el plan de ejecución de SQL Server mediante la mitigación de los errores del plan de consulta y la adaptación del plan de ejecución en función de los resultados de la ejecución.

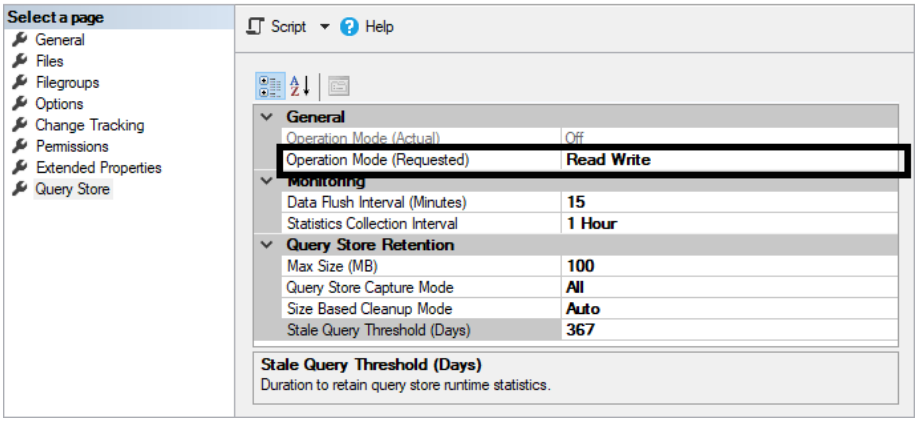
Query Store

Query Store recopila telemetría sobre las estadísticas de tiempo de compilación y tiempo de ejecución. También captura el historial de consultas y planes para su revisión. Los datos están separados por ventanas de tiempo, para que pueda ver los patrones de uso de la base de datos y entender cuándo se produjeron cambios en el plan de consulta en el servidor.

Las estadísticas de espera son otra fuente de información que ayuda a solucionar problemas de rendimiento en SQL Server. Antes, las estadísticas de espera solo estaban disponibles en el nivel de instancia, por lo que resultaba complicado retroceder en los detalles hasta la consulta real. Query Store lo ayuda a hacer un seguimiento más eficaz de las estadísticas de espera porque proporciona información resumida. Las estadísticas de espera están vinculadas a un plan de consulta y se toman a lo largo del tiempo, como las estadísticas de tiempo de ejecución. De este modo se obtienen más conocimientos sobre el rendimiento y los cuellos de botella de las cargas de trabajo, a la vez que se conservan las principales ventajas de Query Store.

Más información: uso de SQL Server Management Studio o la sintaxis de Transact-SQL para Query Store

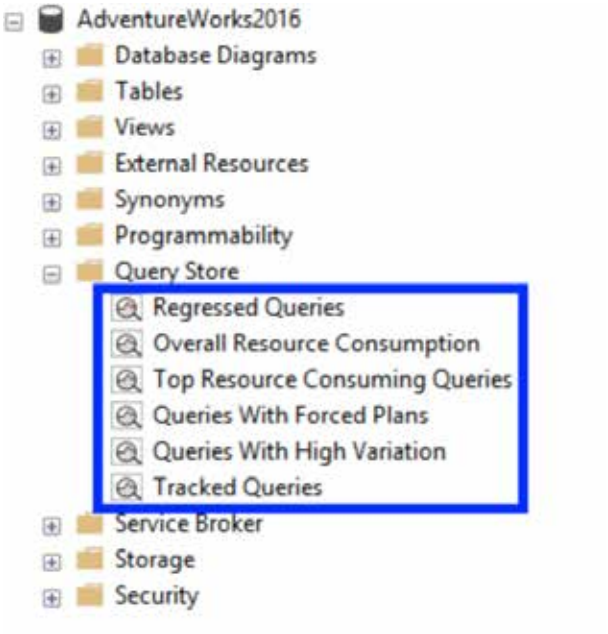
SQL Server Management Studio (SSMS) hospeda una serie de interfaces de usuario diseñadas para configurar Query Store, así como para consumir datos recopilados sobre las cargas de trabajo. En el Explorador de objetos de SSMS, puede habilitar Query Store si selecciona el cuadro **Modo de operación (solicitado)**:



También puede usar la instrucción `ALTER DATABASE` para implementar Query Store. Por ejemplo:

```
ALTER DATABASE AdventureWorks2012 SET QUERY_STORE = ON;
```

Una vez que seleccione una de estas opciones, actualice la parte de base de datos del panel del Explorador de objetos para agregar la sección **Query Store**. Podrá ver los informes de Query Store:



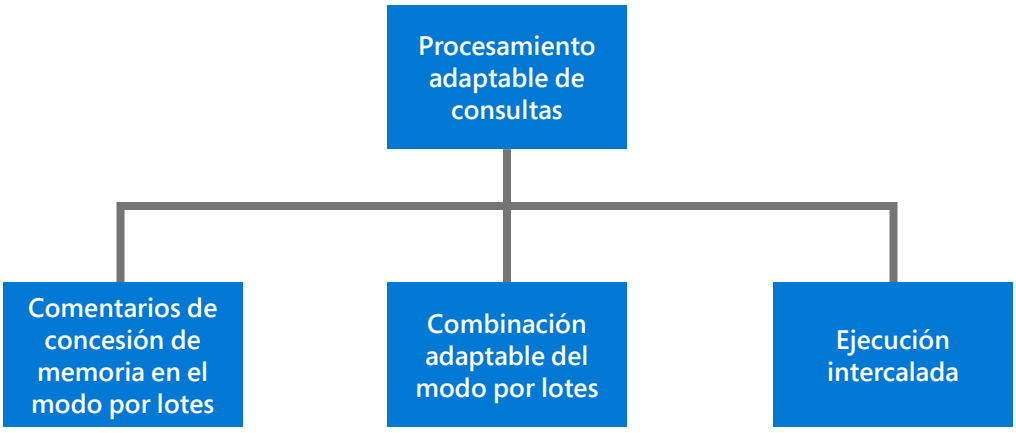
➔ **Para obtener más información** sobre cómo supervisar el rendimiento mediante Query Store, consulte la documentación de Microsoft.

Procesamiento adaptable de consultas

Durante la optimización y el procesamiento de las consultas, el proceso de estimación de cardinalidad (EC) es responsable de calcular el número aproximado de filas procesadas en cada paso del plan de ejecución. Unas estimaciones incorrectas pueden dar como resultado un tiempo de respuesta a la consulta más lento, una utilización innecesaria de recursos (memoria, CPU e I/O) y una reducción del rendimiento y la simultaneidad.

Para mejorar el proceso de CE, SQL Server ofrece una familia de características denominada Procesamiento de consultas adaptable (AQP). Gracias a AQP, SQL Server procesa mucho más rápido las cargas de trabajo porque permite que el procesador de consultas ajuste las opciones del plan de consulta en función de las características de tiempo de ejecución. AQP rompe la barrera entre el plan de consulta y la ejecución real. La optimización se puede hacer mientras la consulta está en ejecución o incluso una vez que se completa, lo que beneficia a las ejecuciones de consultas posteriores. AQP ofrece tres técnicas para adaptarse a las características de la carga de trabajo de la aplicación:

- Comentarios de concesión de memoria en el modo por lotes.
- Combinaciones adaptables del modo por lotes.
- Ejecución intercalada de funciones con valores de tabla de varias instrucciones.



Comentarios de concesión de memoria en el modo por lotes

Para el plan posterior a la ejecución de una consulta, SQL Server realiza una estimación de cardinalidad de un determinado lote de T-SQL y calcula la concesión de memoria mínima necesaria para la ejecución, así como la concesión de memoria ideal necesaria para mantener todas las filas del lote in-memory. Si hay problemas con la CE, el rendimiento se resiente y la memoria disponible se ve limitada. Las concesiones de memoria excesivas hacen que se desperdicie memoria y se reduzca la simultaneidad. Las concesiones de memoria insuficientes provocan desbordamientos costosos en el disco.

Con los comentarios de concesión de memoria en el modo por lotes, SQL Server vuelve a calcular la memoria real necesaria para una consulta y, luego, actualiza el valor de la concesión para el plan almacenado en caché. Cuando se ejecuta una instrucción de consulta idéntica, la consulta usa el nuevo tamaño de la concesión de memoria. El rendimiento mejora porque hay menos escrituras en tempdb; además, como las concesiones de memoria a los lotes son más precisas, se puede proporcionar memoria adicional a los lotes que más la necesitan.

En el caso de concesiones **excesivas**, si la memoria concedida es mayor que el doble del tamaño de la memoria usada, los comentarios de concesión de memoria se recalculan y se actualiza el plan almacenado en caché. En los planes cuyas concesiones de memoria son inferiores a 1 MB no se recalculan los excesos. En las concesiones de memoria **insuficientes** que provocan desbordamientos en disco para los operadores en el modo por lotes, los comentarios de concesión de memoria activarán un nuevo cálculo. Los eventos de escritura se notifican a los comentarios de concesión de memoria. Este evento devuelve el identificador de nodo del plan y el tamaño de los datos escritos de ese nodo.

Combinaciones adaptables del modo por lotes

Normalmente, SQL Server elige entre tres tipos de operadores de combinación física: combinaciones de bucles anidados, combinaciones de mezcla y combinaciones hash. Cada tipo de combinación tiene sus puntos fuertes y débiles, en función de las características de los patrones de consulta y datos. El algoritmo más adecuado en cada consulta depende de las estimaciones de cardinalidad de las entradas de combinación. Unas CE de entrada incorrectas pueden provocar la selección de un algoritmo de combinación inadecuado.

Con la característica de combinaciones adaptables en el modo por lotes, SQL Server permite aplazar la selección de un método de combinación hash o de combinación de bucles anidados hasta **después** de explorar la primera entrada. El operador de combinación adaptable define un umbral que se usa para decidir cuándo se debe cambiar a un plan de bucles anidados. Por tanto, un plan puede cambiarse dinámicamente a una estrategia de combinación mejor durante la ejecución.

Ejecución intercalada de funciones con valores de tabla de varias instrucciones

Las funciones con valores de tabla de múltiples instrucciones (MSTVF) son populares entre los desarrolladores, aunque su ejecución inicial puede causar disminuciones del rendimiento. Con la ayuda de AQP, SQL Server resuelve este problema mediante la ejecución intercalada de la característica MSTVF. Esta característica cambia el límite unidireccional entre las fases de optimización y ejecución para realizar una ejecución de una sola consulta y permite adaptar los planes en función de las estimaciones de cardinalidad revisadas. Con la ejecución intercalada, los recuentos de fila reales de las MSTVF se usan para hacer optimizaciones del plan en niveles inferiores desde las referencias de las MSTVF. El resultado es un plan mejor fundamentado basado en las características reales de la carga de trabajo y, en definitiva, un mejor rendimiento de las consultas.

Cuando se encuentra una MSTVF, el optimizador de consultas realiza las siguientes acciones:

- Pausar la optimización.
- Ejecutar el subárbol de MSTVF para obtener una CE precisa.
- Seguir procesando las operaciones posteriores con una serie de supuestos exactos.

En función de los resultados de la ejecución (número estimado de filas), el optimizador de consultas puede considerar que hay un plan mejor y ejecutar la consulta con el plan modificado.

En general, cuanto mayor sea la desviación entre el número de filas real y estimado, junto con el número de operaciones del plan de bajada, mayor será el impacto sobre el rendimiento. **La ejecución intercalada es beneficiosa para las consultas cuando las dos afirmaciones siguientes son verdaderas:**

- Hay una gran desviación entre el número estimado y real de filas del conjunto de resultados intermedio (en este caso, la MSTVF).
- La consulta en general es sensible a un cambio en el tamaño de los resultados intermedios. Esto suele ocurrir cuando hay un árbol complejo sobre el subárbol en el plan de consulta. Una simple instrucción "`SELECT *`" de una MSTVF no se verá beneficiada por la ejecución intercalada.

Más información: habilitación de AQP

Puedes hacer que las cargas de trabajo sean aptas automáticamente para AQP si habilita el nivel de compatibilidad 140 para la base de datos. Este es un ejemplo de cómo se puede establecer mediante T-SQL:

```
ALTER DATABASE [YourDatabaseName]
SET COMPATIBILITY_LEVEL = 140;
```

➔ **Para obtener más información** sobre cómo usar estas características, consulte Procesamiento adaptable de consultas en bases de datos SQL. ■

SQL Server también ofrece y admite otras herramientas y características para supervisar y optimizar el rendimiento, incluidas opciones de configuración de las características y características de supervisión y optimización.

Opciones de configuración de características para el rendimiento

SQL Server ofrece varias opciones de configuración de disco, servidor, tabla y consulta en el nivel de motor de base de datos para mejorar aún más el rendimiento.

Configuración de discos. Puede definir los niveles 0, 1 y 5 de matriz redundante de discos independientes (RAID) con SQL Server. Con SQL Server puede configurar el nivel 0 de RAID para el seccionamiento de disco, el nivel 1 para el reflejo de disco y el nivel 5 para el seccionamiento con paridad.

Configuración de tempdb. Para optimizar el rendimiento de tempdb, puede usar opciones como la inicialización instantánea de archivos de base de datos, el crecimiento automático y el seccionamiento de disco para mantener tempdb en la unidad local en lugar de tenerlo en la unidad de red compartida. Para obtener más información, consulte Optimizar el rendimiento de tempdb en SQL Server.

Configuración del servidor. Puede configurar opciones del procesador, la memoria, el índice, la copia de seguridad y la consulta para mejorar el rendimiento del servidor con SQL Server. Estas configuraciones incluyen opciones para obtener el grado máximo de paralelismo como MAXDOP, memoria de servidor máxima, optimizar para cargas de trabajo ad hoc y desencadenadores anidados. Para obtener más información, consulte Opciones de configuración de rendimiento.

Configuración de la base de datos. Establezca la compresión de fila y página mediante la función de compresión de datos para índices y tablas de almacenamiento de filas y de columnstore a fin de optimizar el rendimiento de la base de datos. También puede cambiar el nivel de compatibilidad de una base de datos según sus necesidades.

Configuración de tablas. Puede usar índices y tablas con particiones para mejorar el rendimiento de las tablas.

Opciones de rendimiento de consultas. Para mejorar el rendimiento en el nivel de consulta, puede usar índices, particiones, procedimientos almacenados, UDF y estadísticas. Use características descritas previamente como las tablas optimizadas para memoria y los procedimientos almacenados compilados de manera nativa para mejorar el rendimiento de OLTP in-memory. Visite Opciones de rendimiento de consultas para obtener más información.

Características de supervisión y optimización del rendimiento

Herramientas como Query Store, los planes de ejecución, las estadísticas de consultas dinámicas y el Asistente para la optimización de motor de base de datos pueden ayudarlo a supervisar los eventos de SQL Server. También puede usar varios comandos de T-SQL como sp_trace_setfilter para hacer un seguimiento de los eventos de proceso del motor o DBCC TRACEON, un comando para habilitar las marcas de seguimiento. Además, también puede establecer una base de rendimiento si usa sp_configure para determinar si el sistema SQL Server tiene un rendimiento óptimo o no. Para obtener más información, consulte Herramientas de supervisión y optimización del rendimiento.

Resource Governor

Resource Governor es una herramienta de SQL Server que lo ayuda a administrar y especificar límites en los consumos de recursos del sistema. Puede definir simplemente el límite de recursos de CPU, E/S física y memoria que las solicitudes de aplicación entrantes pueden usar. Con Resource Governor, puedes observar continuamente los patrones de uso de los recursos y ajustar la configuración del sistema en consecuencia para maximiza la eficacia. Para obtener más información sobre su uso y funcionamiento, consulte la documentación de Resource Governor. ■

SQL Server 2017 ofrece funcionalidades de procesamiento más rápidas para las aplicaciones más exigentes controladas por datos. Incluye un conjunto único de características basadas en funcionalidades de rendimiento avanzadas líder del sector.

OLTP in-memory proporciona cargas de trabajo de procesamiento de transacciones en línea más rápidas y mejor rendimiento. Los índices de columnstore mejoran el rendimiento de la base de datos y permiten realizar análisis de alta velocidad. Además, hay características como Procesamiento de consultas adaptable que permiten que los DBA optimicen aún más sus funcionalidades de procesamiento de consultas. SQL Server 2017 ofrece la velocidad, las características y la escalabilidad que las organizaciones necesitan para seguir funcionando al ritmo que sus usuarios exigen. ■

Descubra cómo ejecutar SQL Server en la plataforma de su preferencia.

Más información sobre las características más recientes de SQL Server.

Descubra el rendimiento y las comparativas del sector de SQL.

Descargue SQL Server.