Prácticas recomendadas para las consultas de Amazon Redshift

# AWS Orientación prescriptiva

# AWS Orientación prescriptiva: Prácticas recomendadas para las consultas de Amazon Redshift

# Table of Contents

# Prácticas recomendadas para las consultas de Amazon Redshift

Ethan Stark, Amazon Web Services (AWS)

junio de 2024 (historial de documentos)

## Descripción general

En esta guía se proporcionan recomendaciones y prácticas recomendadas para optimizar el rendimiento de las consultas y tablas en Amazon Redshift. Puede usar Amazon Redshift para consultar petabytes de datos estructurados y semiestructurados en su almacén de datos y su lago de datos mediante SQL estándar. En esta guía también se proporciona una descripción general de los componentes básicos de la arquitectura de un almacén de datos de Amazon Redshift. Este conocimiento, junto con la comprensión de los factores de rendimiento de las consultas (como las propiedades de las tablas, la configuración del clúster y la estructura de las consultas), puede ayudarlo a diseñar tablas y consultas eficientes y eficaces para su almacén de datos de Amazon Redshift.

## Destinatarios previstos

Esta guía está prevista para ingenieros de datos, arquitectos de datos y analistas de datos que diseñan o utilizan tablas y consultas en Amazon Redshift.

## Objetivos

Esta guía puede serle útil a usted y a su organización para lograr los objetivos siguientes:

- Diseñar tablas para realizar operaciones óptimas de almacenamiento y recuperación de datos
- Diseñar consultas para obtener un rendimiento y un ahorro de costos óptimos
- Optimizar el rendimiento de Amazon Redshift Spectrum para consultar datos directamente desde los archivos de Amazon Simple Storage Service (Amazon S3)

# Componentes de arquitectura de un almacén de datos de Amazon Redshift

Le recomendamos que tenga un conocimiento básico de los principales componentes de la arquitectura de un almacén de datos de Amazon Redshift. Este conocimiento puede ayudarlo a comprender mejor cómo diseñar las consultas y las tablas para obtener un rendimiento óptimo.

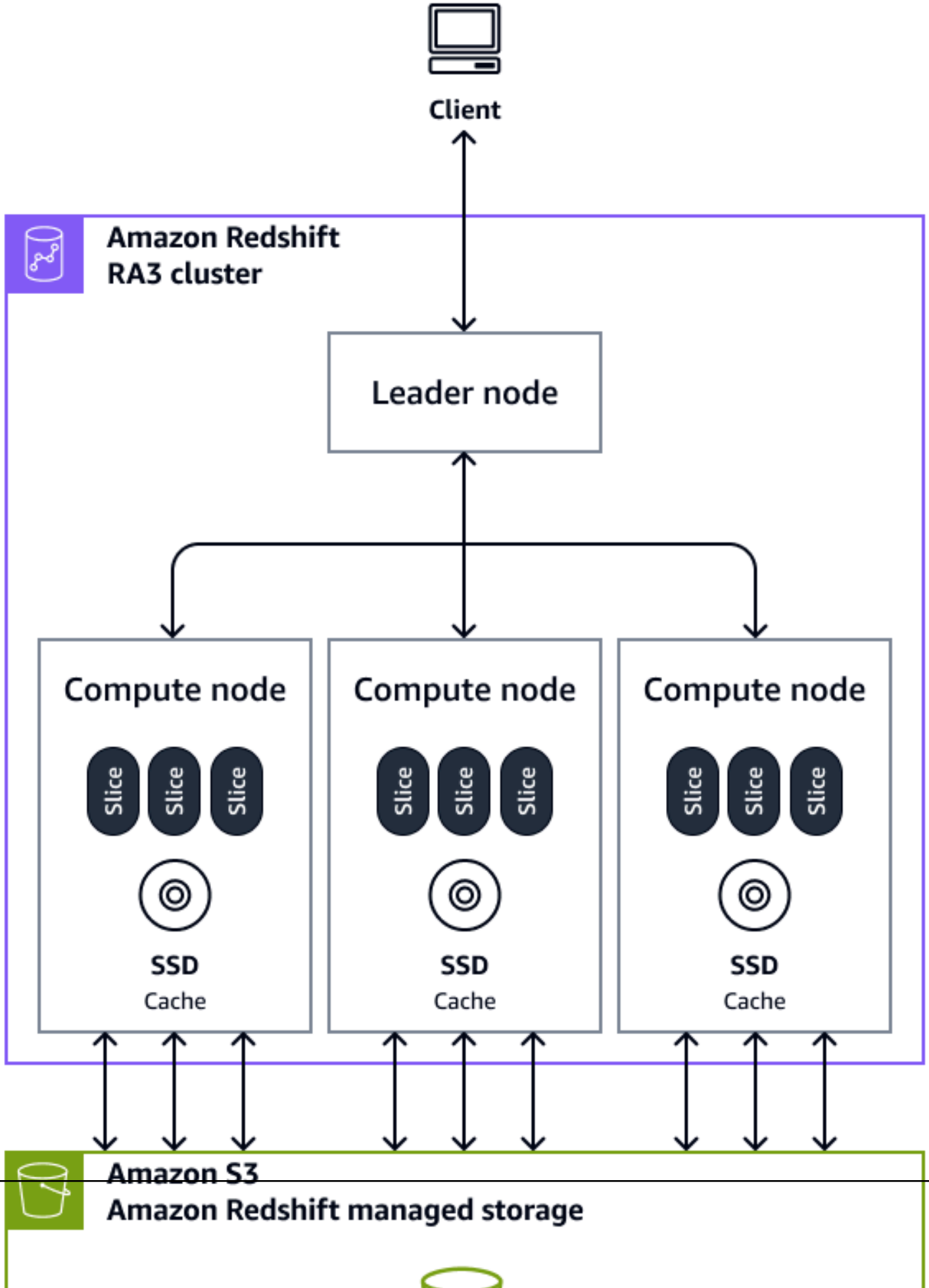
Un almacén de datos en Amazon Redshift consta de los siguientes componentes de arquitectura principales:

- Clústeres: un clúster, compuesto por uno o más nodos de computación, es el componente de infraestructura principal de un almacén de datos de Amazon Redshift. Los nodos de computación son transparentes para las aplicaciones externas, pero la aplicación cliente solo interactúa directamente con el nodo principal. Los clústeres habituales tienen dos o más nodos de computación. Los nodos de computación se coordinan a través del nodo principal.
- Nodo principal: los nodos principales administran las comunicaciones de los programas cliente y todos los nodos de computación. Los nodos principales también preparan los planes para ejecutar una consulta cada vez que se envía una consulta a un clúster. Cuando los planes están listos, el nodo principal compila el código, lo distribuye a los nodos de computación y les asigna sectores de los datos a cada uno para procesar los resultados de la consulta.
- Nodo de computación: los nodos de computación ejecutan las consultas. El nodo principal compila un código para los elementos individuales del plan a fin de ejecutar la consulta y lo asigna a los nodos de computación individuales. Los nodos de computación ejecutan el código compilado y envían resultados intermedios de vuelta al nodo principal para su agregación final. Cada nodo de computación tiene su propia CPU dedicada, memoria y almacenamiento en disco integrado. A medida que la carga de trabajo crece, puede aumentar la capacidad de computación y almacenamiento de un clúster aumentando el número de nodos, actualizando el tipo de nodo o ambas.
- Sector de nodo: los nodos de computación están divididos en sectores. A cada sector de los nodos de computación se le asigna una parte de la memoria y del espacio en disco del nodo, donde se procesa una parte de la carga de trabajo asignada al nodo. A continuación, los sectores funcionan en paralelo para completar la operación. Los datos se distribuyen entre los sectores en función del estilo de distribución y la clave de distribución de una tabla en particular. Una distribución uniforme de los datos hace posible que Amazon Redshift asigne las cargas de trabajo de manera uniforme a los sectores y maximiza las ventajas del procesamiento en paralelo. El número de sectores

por nodo de computación se decide en función del tipo de nodo. Para obtener más información, consulte Clústeres y nodos de Amazon Redshift en la documentación de Amazon Redshift.

- Procesamiento en paralelo masivo (MPP): Amazon Redshift utiliza la arquitectura MPP para procesar datos de forma rápida, incluso las consultas complejas y con grandes cantidades de datos. Varios nodos de computación ejecutan el mismo código de consulta en partes de los datos para maximizar el procesamiento en paralelo.
- Aplicación cliente: Amazon Redshift se integra a distintas herramientas de carga de datos, extracción, transformación y carga (ETL), inteligencia empresarial (BI), generación de informes, minería de datos y análisis. Todas las aplicaciones cliente se comunican con el clúster solamente mediante el nodo principal.

En el siguiente diagrama se muestra cómo los componentes de la arquitectura de un almacén de datos de Amazon Redshift trabajan juntos para acelerar las consultas.


Client
Amazon Redshift
RA3 cluster
Leader node
Compute node
Slice
Slice
Slice
SSD
Cache
Compute node
Slice
Slice
Slice
SSD
Cache
Compute node
Slice
Slice
Slice
SSD
Cache
Amazon S3
Amazon Redshift managed storage

El ciclo de vida de la consulta consta de siete etapas:

1. Recepción y análisis de consultas:
   - El nodo principal recibe la consulta y analiza el SQL.
   - El analizador genera un árbol de consultas inicial que representa la estructura lógica de la consulta original.
   - Amazon Redshift introduce este árbol de consultas en el optimizador de consultas.
2. Optimización de consultas:
   - El optimizador evalúa la consulta y, de ser necesario, la reescribe para maximizar su eficacia.
   - Este proceso de optimización puede implicar la creación de varias consultas relacionadas que reemplazan una misma consulta.
3. Generación de planes de consultas:
   - El optimizador genera un plan de consultas (o varios planes, si es necesario) para su ejecución.
   - El plan de consultas especifica las opciones de ejecución, como los tipos de combinaciones, el orden de las combinaciones, los métodos de agregación y los requisitos de distribución de datos.
4. Traducción del motor de ejecución:
   - El motor de ejecución traduce el plan de consultas en pasos, segmentos y flujos separados:
     - Paso: representa una operación individual requerida durante la ejecución de la consulta. Los pasos se pueden combinar para permitir que los nodos de computación realicen consultas, combinaciones y otras operaciones en la base de datos.
     - Segmento: combina varios pasos que puede ejecutar un solo proceso. También es la unidad de compilación más pequeña que puede ejecutar un segmento de nodos de computación. (Un sector es la unidad de procesamiento en paralelo de Amazon Redshift).
     - Flujo: conjunto de segmentos distribuidos en los sectores de nodos de computación disponibles.
   - El motor de ejecución genera un código compilado en función de estos pasos, segmentos y flujos. El código compilado se ejecuta más rápido que el código interpretado y consume menor capacidad de computación.
   - El nodo principal transmite el código compilado a los nodos de computación.
5. Ejecución en paralelo:
   - Este paso se produce una vez para cada flujo.
   - Los sectores de los nodos de computación ejecutan los segmentos de la consulta en paralelo.

- Durante este proceso, Amazon Redshift optimiza la comunicación de red, el uso de la memoria y la administración de discos para transmitir los resultados intermedios de un paso del plan de consulta al siguiente.
- Esta optimización contribuye a una ejecución más rápida de las consultas.

6. Procesamiento de flujos:
   - Este paso se produce una vez para cada flujo.
   - El motor crea segmentos ejecutables para cada flujo para lograr un procesamiento en paralelo eficiente.
7. Clasificación y agregación finales:
   - El nodo principal gestiona cualquier clasificación o agregación final que necesite la consulta.
   - Cuando ha finalizado, el nodo principal devuelve los resultados al cliente.

Para obtener más información sobre los componentes de la arquitectura, consulte Arquitectura del sistema de almacenamiento de datos en la documentación de Amazon Redshift.

# Factores de rendimiento de las consultas para Amazon Redshift

Hay una serie de factores que pueden afectar al rendimiento de las consultas. Los siguientes aspectos de sus datos, su clúster y de las operaciones de la base de datos influyen en el tiempo de procesamiento de sus consultas:

- Propiedades de la tabla
  - Claves de clasificación (Amazon Redshift Advisor)
  - Compresión de datos (automatizado)
  - Distribución de datos (automatizado)
  - Información de tablas (automatizado)
- Configuración del clúster
  - Tipo de nodo
  - Tamaño y cantidad de los nodos y sectores
  - Administración de la carga de trabajo (automatizado)
  - Aceleración de consultas cortas (automatizado)
- Consulta SQL
  - Estructura de consulta
  - Compilación de código

## Propiedades de la tabla

Las tablas de Amazon Redshift son las unidades fundamentales para almacenar datos en Amazon Redshift y cada tabla tiene un conjunto de propiedades que determinan su comportamiento y accesibilidad. Algunas de estas propiedades son la clasificación, el estilo de distribución, la codificación por compresión y muchas otras. Entender estas propiedades es fundamental para optimizar el rendimiento, la seguridad y la rentabilidad de las tablas de Amazon Redshift.

### Claves de clasificación

Amazon Redshift almacena los datos en disco en un determinado orden en función de las clave de clasificación de una tabla. El optimizador y el procesador de consultas utilizan la información

de ubicación de los datos en el nodo de computación para reducir la cantidad de bloques que se deben examinar. Esto mejora considerablemente la velocidad de las consultas al reducir la cantidad de datos que se tienen que procesar. Le recomendamos que utilice claves de clasificación para facilitar los filtros de la cláusula `WHERE`. Para obtener más información, consulte Uso de las claves de clasificación en la documentación de Amazon Redshift.

## Compresión de datos

La compresión de datos reduce los requisitos de almacenamiento, lo que reduce el disco I/O y mejora el rendimiento de las consultas. Cuando se ejecuta una consulta, los datos comprimidos se leen en la memoria y luego se descomprimen cuando se ejecuta la consulta. Al cargar menos datos en la memoria, Amazon Redshift puede asignar más memoria al analizar los datos. Como el almacenamiento en columnas guarda los datos similares de forma secuencial, Amazon Redshift puede aplicar codificaciones de compresión adaptables que estén asociadas específicamente a los tipos de datos en columnas. La mejor forma de habilitar la compresión de datos en las columnas de las tablas es mediante la opción `AUTO` de Amazon Redshift de aplicar las codificaciones de compresión óptimas cuando se carga la tabla con los datos. Para obtener más información sobre el uso de la compresión automática de datos, consulte Carga de tablas con compresión automática en la documentación de Amazon Redshift.

## Distribución de datos

Amazon Redshift almacena los datos en los nodos de computación en función del estilo de distribución de la tabla. Cuando ejecuta una consulta, el optimizador de consultas redistribuye los datos a los nodos informáticos según se necesite para realizar uniones y agregaciones. Elegir el estilo de distribución correcto para una tabla ayuda a reducir el impacto del paso de redistribución al localizar los datos en el lugar que deben estar, antes de que se realicen las combinaciones. Le recomendamos que utilice claves de distribución para facilitar las combinaciones más comunes. Para obtener más información, consulte Uso de los estilos de distribución de datos en la documentación de Amazon Redshift.

## Información de tablas

Aunque, desde el primer momento, Amazon Redshift proporciona un rendimiento líder en el sector para la mayoría de las cargas de trabajo, mantener los clústeres de Amazon Redshift ejecutándose de forma correcta requiere tareas de mantenimiento. Al actualizar y eliminar datos, se crean filas inactivas que hay que eliminar, e incluso las tablas que solo se adjuntan se tienen que volver a ordenar si el orden de incorporación no es coherente con la clave de clasificación.

## Vacuum

El proceso de succión en Amazon Redshift es esencial para el buen estado y el mantenimiento de su clúster de Amazon Redshift. También afecta al rendimiento de las consultas. Como las eliminaciones y las actualizaciones marcan los datos antiguos pero en realidad no los eliminan, debe utilizar la succión para recuperar el espacio en disco que ocupaban las filas de la tabla que las operaciones `UPDATE` y `DELETE` anteriores marcaron para su eliminación. Amazon Redshift puede ordenar y realizar una operación `VACUUM DELETE` de forma automática en las tablas en segundo plano.

Para limpiar las tablas tras una carga o una serie de actualizaciones incrementales, también puede ejecutar el comando `VACUUM`, ya sea en la base de datos completa o en tablas individuales. Si las tablas tienen claves de clasificación y las cargas de las tablas no se han optimizado para ordenarlas a medida que se insertan, debe utilizar la succión para volver a ordenar los datos (lo que puede ser crucial para el rendimiento). Para obtener más información, consulte Limpieza de tablas en la documentación de Amazon Redshift.

## Análisis

La operación `ANALYZE` actualiza los metadatos estadísticos de las tablas de una base de datos de Amazon Redshift. Mantener las estadísticas actualizadas mejora el rendimiento de las consultas, ya que permite que el planificador de consultas elija los planes óptimos. Amazon Redshift monitorea constantemente la base de datos y, de forma automática, realiza operaciones de análisis en segundo plano. Para minimizar el impacto en el rendimiento del sistema, la operación `ANALYZE` se ejecuta automáticamente durante periodos en los que hay poca carga de trabajo. Si decide ejecutar explícitamente el comando `ANALYZE`, haga lo siguiente:

- Ejecute el comando `ANALYZE` antes de ejecutar cualquier consulta.
- Ejecute el comando `ANALYZE` en la base de datos de manera habitual al final de cada ciclo de carga o actualización regular.
- Ejecute el comando `ANALYZE` en las tablas nuevas que cree y en las tablas o columnas existentes que sufran cambios significativos.
- Considere ejecutar operaciones `ANALYZE` en programas diferentes para diferentes tipos de tablas y columnas, según su uso en consultas y su tendencia al cambio.
- Para ahorrar tiempo y recursos del clúster, al ejecutar el comando `ANALYZE` use la cláusula `PREDICATE COLUMNS`.

# Configuración del clúster

Un clúster es una colección de nodos que realizan el almacenamiento y el procesamiento reales de los datos. Es fundamental configurar el clúster de Amazon Redshift correctamente si quiere lograr lo siguiente:

- Alta escalabilidad y simultaneidad
- Uso eficiente de Amazon Redshift
- Mejor rendimiento
- Menor costo

## Tipo de nodo

Un clúster de Amazon Redshift puede utilizar uno de varios tipos de nodos (RA3 DC2, y DS2). Cada tipo de nodo ofrece diferentes tamaños y límites para ayudarlo a escalar su clúster de manera adecuada. El tamaño del nodo determina la capacidad de almacenamiento, la memoria, la CPU y el precio de cada nodo del clúster. La optimización del costo y el rendimiento comienza con la elección del tipo y tamaño de nodo correctos. Para obtener más información sobre los tipos de nodos, consulte Información general de los clústeres de Amazon Redshift en la documentación de Amazon Redshift.

## Tamaño y cantidad de los nodos y sectores

Un nodo de computación está particionado en sectores. Si hay más nodos, hay más procesadores y sectores lo que permite procesar sus consultas con mayor rapidez al ejecutar porciones de una consulta de manera simultánea en todos los sectores. Sin embargo, cuantos más nodos, mayor será el gasto. Esto significa que debe encontrar el equilibrio adecuado entre costo y rendimiento para su sistema. Para obtener más información sobre la arquitectura de clústeres de Amazon Redshift, consulte Arquitectura del sistema de almacenamiento de datos en la documentación de Amazon Redshift.

## Administración de la carga de trabajo

La administración de la carga de trabajo (WLM) de Amazon Redshift permite a los usuarios administrar las colas de cargas de trabajo con prioridades de manera flexible, por lo que las consultas cortas y de ejecución rápida no quedarán bloqueadas en las colas detrás de las consultas

de ejecución prolongada. La WLM automática utiliza algoritmos de machine learning (ML) para perfilar las consultas y colocarlas en la cola adecuada con los recursos adecuados, a la vez que administra la simultaneidad de las consultas y la asignación de memoria. Para obtener más información sobre la WLM, consulte Implementación de la administración de la carga de trabajo en la documentación de Amazon Redshift.

### Aceleración de consultas cortas

La aceleración de consultas cortas (SQA) da prioridad a una serie de consultas que se ejecutan rápidamente frente a consultas que tardan más en ejecutarse. SQA ejecuta las consultas en un espacio dedicado, de forma que estas consultas no tienen que esperar en las colas detrás de otras consultas más largas. SQA solamente da prioridad a las consultas de corta ejecución y a las consultas que están en una cola definida por el usuario. Si usa SQA, las consultas cortas se ejecutan con mayor rapidez y tardará menos en ver los resultados. Si habilita SQA, podrá reducir o eliminar las colas de WLM dedicadas a las consultas cortas. Además, las consultas largas no necesitan competir por los espacios de una cola de WLM. Esto significa que puede configurar las colas de WLM para que utilicen menos espacios de consulta. Si se utiliza una simultaneidad más baja, el rendimiento de las consultas aumenta y el rendimiento de todo el sistema mejora con la mayoría de las cargas de trabajo. Para obtener más información sobre SQA, consulte Uso de la aceleración de consultas cortas en la documentación de Amazon Redshift.

## Consulta SQL

Una consulta de base de datos es una solicitud de los datos de una base de datos. La solicitud debe proceder en un clúster de Amazon Redshift mediante SQL. Amazon Redshift admite herramientas de cliente SQL que se conectan a través de Java Database Connectivity (JDBC) y Open Database Connectivity (ODBC). Puede utilizar la mayoría de las herramientas de cliente SQL compatibles con los controladores JDBC u ODBC.

### Estructura de consulta

La forma en la que está escrita la consulta afecta el rendimiento. Le recomendamos que escriba consultas de forma que se tenga que procesar y devolver la menor cantidad de datos que sea necesaria para satisfacer sus necesidades. Para obtener más información sobre cómo estructurar las consultas, consulte la sección Prácticas recomendadas para diseñar consultas de Amazon Redshift de esta guía.

# Compilación de código

Amazon Redshift genera y compila código optimizado para cada plan de ejecución de consultas. El código compilado se ejecuta más rápido porque elimina los gastos generales de tener que recurrir a un intérprete. Para minimizar la latencia de las consultas nuevas y, al mismo tiempo, conservar las ventajas de rendimiento del código compilado, Amazon Redshift utiliza una técnica denominada composición. Composition genera una estructura ligera de lógica preexistente para procesar las nuevas consultas de forma inmediata y, al mismo tiempo, compila código altamente optimizado y específico para las consultas en segundo plano. Esto elimina la compilación de la ruta crítica de ejecución de las consultas. Esto significa que las consultas nuevas se inician más rápido y ofrecen un rendimiento coherente con las ejecuciones posteriores.

Amazon Redshift también utiliza un servicio de compilación sin servidor para escalar las compilaciones de consultas más allá de los recursos informáticos de un clúster de Amazon Redshift. Los segmentos de código compilados se almacenan en caché tanto localmente en el clúster como en una caché remota prácticamente ilimitada que persiste después de reiniciarse el clúster. Las ejecuciones posteriores de la misma consulta se realizan en menos tiempo, ya que se puede omitir la fase de compilación. Al utilizar un servicio de compilación escalable, Amazon Redshift compila el código en paralelo para ofrecer un rendimiento rápido y uniforme.

# Prácticas recomendadas para el diseño de tablas de Amazon Redshift

En esta sección se brinda información general sobre las prácticas recomendadas para diseñar tablas de bases de datos. Le recomendamos que siga estas prácticas recomendadas para lograr un rendimiento y una eficacia óptimos en las consultas.

## Descripción del funcionamiento de las claves de clasificación

Amazon Redshift almacena los datos en el disco en un determinado orden en función de la clave de ordenación. El optimizador de consultas de Amazon Redshift utiliza la ordenación cuando determina cuáles son los planes óptimos de consulta. Para usar las claves de clasificación de forma eficaz, le recomendamos que haga lo siguiente:

- Mantenga la tabla lo más ordenada posible.
- Utilice la clasificación `VACUUM` para restablecer un rendimiento óptimo.
- Evite comprimir la columna de claves de clasificación.
- Si la clave de clasificación está comprimida y la relación `sortkey1_skew` es significativamente alta, vuelva a crear la tabla sin habilitar la compresión de la clave de clasificación.
- Evite aplicar una función a las columnas de la clave de clasificación. Por ejemplo, en la siguiente consulta, la columna de claves de clasificación `trans_dt : TIMESTAMPTZ` no se usa si convierte su tipo a `DATE`:

```
select order_id, order_amt
from sales
where trans_dt::date = '2021-01-08'::date
```

- Realice las operaciones `INSERT` en el orden de las claves de clasificación.
- Utilice las claves de clasificación en la cláusula `GROUP BY` siempre que sea posible.

## Consejos de ajuste de consultas

Le recomendamos que siga estos pasos para ajustar las consultas:

- Ordene siempre las claves de clasificación compuestas desde la cardinalidad más baja hasta la cardinalidad más alta para lograr una eficacia óptima.

- Si la clave principal de una clave de clasificación compuesta es relativamente única (es decir, tiene una cardinalidad alta), evite agregar más columnas a la clave de clasificación. Agregar más columnas tiene poco impacto en el rendimiento de las consultas, pero incrementa los costos de mantenimiento.

## Evaluación de la eficacia de las claves de clasificación

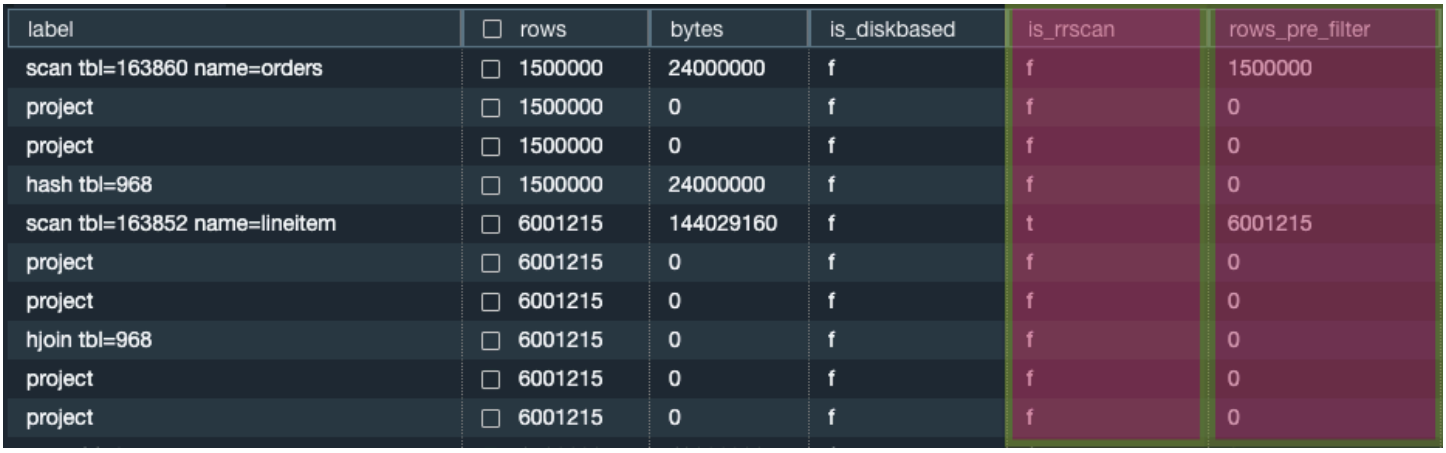
Para optimizar las consultas, debe poder evaluar su eficacia. Le recomendamos que utilice la vista SVL_QUERY_SUMMARY para encontrar información general acerca de la ejecución de una consulta. En esta vista, puede utilizar el atributo `IS_RRSCAN` para determinar si un paso del plan `EXPLAIN` usa un análisis de rango restringido. También puede utilizar el atributo `rows_pre_filter` para determinar la selectividad de una clave de clasificación.

También puedes usar una vista de administración GitHub llamada v_my_last_query_summary. En la vista se muestra información sobre la última consulta que se ejecutó.

La siguiente instrucción muestra cómo obtener información general acerca de la ejecución de una consulta.

```
select lpad(' ',stm+seg+step) || label as label,
       rows,
       bytes,
       is_diskbased,
       is_rrscan,
       rows_pre_filter
from svl_query_summary
where query = pg_last_query_id()
order by stm, seg, step;
```

La consulta anterior devuelve el siguiente ejemplo de salida.



| label | rows | bytes | is_diskbased | is_rrscan | rows_pre_filter |
|---|---|---|---|---|---|
| scan tbl=163860 name=orders | 1500000 | 24000000 | f | f | 1500000 |
| project | 1500000 | 0 | f | f | 0 |
| project | 1500000 | 0 | f | f | 0 |
| hash tbl=968 | 1500000 | 24000000 | f | f | 0 |
| scan tbl=163852 name=lineitem | 6001215 | 144029160 | f | t | 6001215 |
| project | 6001215 | 0 | f | f | 0 |
| project | 6001215 | 0 | f | f | 0 |
| hjoin tbl=968 | 6001215 | 0 | f | f | 0 |
| project | 6001215 | 0 | f | f | 0 |
| project | 6001215 | 0 | f | f | 0 |

## Información sobre su tabla

Es importante entender las propiedades fundamentales de la tabla. Para obtener más información acerca de la tabla, siga estos pasos:

- Use PG_TABLE_DEF para ver información acerca de las columnas de la tabla.
- Use SVV_TABLE_INFO para consultar información más exhaustiva acerca de una tabla, incluidos el sesgo de distribución de datos, el sesgo de distribución de claves, el tamaño de tabla y las estadísticas.

## Selección del estilo de distribución de tablas adecuado

Cuando ejecuta una consulta, el optimizador de consultas redistribuye las filas a los nodos de computación según se necesite para realizar combinaciones y agregaciones. El objetivo al seleccionar un estilo de distribución de tablas es reducir el impacto del paso de redistribución al localizar los datos en el lugar que deben estar antes de que ejecute la consulta.

Recomendamos el siguiente enfoque para elegir el estilo de distribución de tablas adecuado:

- Para evitar la difusión y la redistribución en un plan de ejecución de consultas, coloque las filas dentro del mismo nodo. Por ejemplo, si selecciona `DISTKEY`, puede distribuir la tabla de hechos y la tabla unidimensional en sus columnas comunes. Seleccione la mayor dimensión según el tamaño del conjunto de datos filtrado. Solo se deben distribuir las filas que se usan en la combinación; por lo tanto, considere el tamaño del conjunto de datos después del filtrado, no el tamaño de la tabla.

- Asegúrese de que no haya asimetría en la columna en la que se cree la clave de distribución. De ser así, un nodo de computación podría realizar más tareas que otros. Si observa que hay asimetría, considere la posibilidad de cambiar la columna de claves de distribución. Se puede considerar que una columna es candidata a contener claves de distribución si sus valores están distribuidos uniformemente o tienen valores cardinales altos.
- Si la tabla utilizada en la condición de combinación es pequeña (menos de 1 GB), considere el estilo de distribución `ALL`.
- Puede comprimir la clave de distribución, pero debe evitar comprimir la columna de las claves de clasificación (especialmente la primera columna de las claves de clasificación).

> **Note**
>
> Si utiliza la optimización automática de tablas, no necesita elegir el estilo de distribución de la tabla. Para obtener más información, consulte Uso de la optimización de tablas automática en la documentación de Amazon Redshift. Para que Amazon Redshift elija el estilo de distribución adecuado, especifique `AUTO` en el estilo de distribución.

# Prácticas recomendadas para diseñar consultas de Amazon Redshift

En esta sección se brinda información general sobre las prácticas recomendadas para diseñar consultas. Le recomendamos que siga las prácticas recomendadas de esta sección para lograr un rendimiento y una eficacia óptimos en las consultas.

## Recomendación de evitar usar la instrucción SELECT * FROM

Le recomendamos que evite el uso de la instrucción `SELECT * FROM`. En su lugar, enumere siempre las columnas para su análisis. Esto reduce el tiempo de ejecución de las consultas y los costos de los análisis de las consultas de Amazon Redshift Spectrum.

Ejemplo de lo que se debe evitar

```
select *
from sales;
```

Ejemplo de práctica recomendada

```
select sales_date, sales_amt
from sales;
```

## Identificación de los problemas con las consultas

Le recomendamos que consulte la vista STL_ALERT_EVENT_LOG para identificar y corregir posibles problemas con la consulta.

## Obtención de información resumida sobre la consulta

Le recomendamos que use las vistas SVL_QUERY_SUMMARY y SVL_QUERY_REPORT para obtener información resumida sobre las consultas. Puede utilizar esta información para optimizar sus consultas.

# Recomendación de evitar las combinaciones cruzadas

Le recomendamos que evite usar las combinaciones cruzadas a menos que sea absolutamente necesario. Si una condición de combinación, las combinaciones cruzadas son un producto cartesiano de dos tablas. Por lo general, las combinaciones cruzadas se ejecutan como combinaciones de bucles anidados (que son los tipos de combinación más lentos).

Ejemplo de lo que se debe evitar

```
select c.c_name,
          n.n_name
from tpch.customer c,
        tpch.nation n;
```

Ejemplo de práctica recomendada

```
select c.c_name,
           n.n_name
from tpch.customer c,
join tpch.nation n
  on n.n_nationkey = c.c_nationkey;
```

# Recomendación de evitar las funciones en los predicados de las consultas

Le recomendamos que evite el uso de las funciones en los predicados de las consultas. El uso de funciones en los predicados de las consultas puede afectar negativamente al rendimiento, ya que las funciones suelen agregar una sobrecarga de procesamiento adicional a cada fila y suelen ralentizar la ejecución general de la consulta.

Ejemplo de lo que se debe evitar

```
select sum(o_totalprice)
from tpch.orders
where datepart(year, o_orderdate) = 1992;
```

Ejemplo de práctica recomendada

```
select sum(o_totalprice)
```

```
from tpch.orders
where o_orderdate between '1992-01-01' and '1992-12-31';
```

## Recomendación de evitar las conversiones de tipo innecesarias

Le recomendamos que evite usar las conversiones de tipo innecesarias en las consultas, ya que la conversión de tipos de datos requiere tiempo y recursos y ralentiza la ejecución de las consultas.

Ejemplo de lo que se debe evitar

```
select sum(o_totalprice)
from tpch.orders
where o_ordertime::date = '1992-01-01';
```

Ejemplo de práctica recomendada

```
select sum(o_totalprice)
from tpch.orders
where o_ordertime between '1992-01-01 00:00:00' and '1992-12-31 23:59:59';
```

## Uso de expresiones CASE en las agregaciones complejas

Le recomendamos que utilice una expresión CASE para realizar agregaciones complejas en lugar de seleccionar de la misma tabla varias veces.

Ejemplo de lo que se debe evitar

```
select sum(sales_amt) as us_sales
from sales
where country = 'US';

select sum(sales_amt) as ca_sales
from sales
where country = 'CA';
```

Ejemplo de práctica recomendada

```
select sum(case when country = 'US' then sales_amt end) as us_sales,
```

```
           sum(case when country = 'CA' then sales_amt end) as ca_sales
from sales;
```

## Uso de subconsultas

Le recomendamos que utilice subconsultas en los casos donde una tabla en la consulta se utilice solamente para condiciones de predicado y la subconsulta devuelva una cantidad pequeña de filas (menos de 200).

Ejemplo de lo que se debe evitar

Si una subconsulta devuelve menos de 200 filas:

```
select sum(order_amt) as total_sales
from sales
where region_key IN
        (select region_key
         from regions
         where state = 'CA');
```

Ejemplo de práctica recomendada

Si una subconsulta devuelve 200 filas o más:

```
select sum(o.order_amt) as total_sales
from sales o
join regions r
  on r.region_key = o.region_key
  and r.state = 'CA';
```

## Uso de predicados

Le recomendamos que utilice predicados para restringir el conjunto de datos tanto como sea posible. En SQL, los predicados se utilizan para filtrar y restringir los datos que se devuelven en una consulta. Al especificar las condiciones en un predicado, puede especificar qué filas deben incluirse en los resultados de la consulta en función de las condiciones especificadas. Esto le permite recuperar solo los datos que le interesan y mejora la eficacia y precisión de las consultas. Para obtener más información, consulte Condiciones en la documentación de Amazon Redshift.

## Adición de predicados para filtrar tablas con combinaciones

Le recomendamos que agregue predicados para filtrar tablas que participen en combinaciones, aun cuando se apliquen los mismos filtros. El uso de predicados para filtrar tablas con combinaciones en SQL puede mejorar el rendimiento de las consultas, ya que reduce la cantidad de datos que se deben procesar y el tamaño del conjunto de resultados intermedios. Al especificar las condiciones de la operación de combinación en la cláusula `WHERE`, el motor de ejecución de consultas puede eliminar las filas que no coincidan con las condiciones antes de combinarlas. En consecuencia, se reduce el tamaño del conjunto de resultados y se incrementa la velocidad de las consultas.

Ejemplo de lo que se debe evitar

```
select p.product_name, sum(o.order_amt)
from sales o
join product p
   on r.product_key = o.product_key
where o.order_date > '2022-01-01';
```

Ejemplo de práctica recomendada

```
select p.product_name, sum(o.order_amt)
from sales o
join product p
  on p.product_key = o.product_key
  and p.added_date > '2022-01-01'
where o.order_date > '2022-01-01';
```

## Uso de los operadores menos costosos para los predicados

En el predicado, utilice los operadores menos costosos que pueda. Los operadores de condiciones de comparación son preferibles a los operadores LIKE. Los operadores `LIKE` siguen siendo preferibles a los operadores SIMILAR TO o POSIX.

## Uso de las claves de clasificación en las cláusulas GROUP BY

Utilice las claves de clasificación en la cláusula `GROUP BY` para que el planificador de consultas pueda utilizar la agrupación de manera más eficiente. Una consulta puede considerarse para la agregación en una fase cuando la lista `GROUP BY` tiene solamente columnas de clave de

clasificación, una de las cuales es también la clave de distribución. Las columnas con clave de clasificación en la lista GROUP BY deben incluir la primera clave de clasificación y, luego, las demás claves de clasificación que desee usar en el orden de la clave de clasificación.

## Recomendación de aprovechar las vistas materializadas

Si es posible, reescriba la consulta sustituyendo el código complejo por una vista materializada, lo que mejorará considerablemente el rendimiento de la consulta. Para obtener más información, consulte Creación de vistas materializadas en Amazon Redshift en la documentación de Amazon Redshift.

## Precaución con las columnas de las cláusulas GROUP BY y ORDER BY

Si utiliza las cláusulas GROUP BY y ORDER BY, asegúrese de colocar las columnas en el mismo orden en ambas cláusulas GROUP BY y ORDER BY. GROUP BY requiere implícitamente que los datos estén ordenados. Si la cláusula ORDER BY es diferente, los datos se deben ordenar dos veces.

Ejemplo de lo que se debe evitar

```
select a, b, c, sum(d)
from a_table
group by b, c, a
order by a, b, c
```

Ejemplo de práctica recomendada

```
select a, b, c, sum(d)
from a_table
group by a, b, c
order by a, b, c
```

# Prácticas recomendadas para usar Amazon Redshift Spectrum

En esta sección se proporciona información general sobre las prácticas recomendadas para utilizar Amazon Redshift Spectrum. Le recomendamos que siga estas prácticas recomendadas para lograr un rendimiento óptimo al utilizar Redshift Spectrum:

- Tenga en cuenta que los tipos de archivos tienen una influencia importante en el rendimiento de las consultas de Redshift Spectrum. Para mejorar el rendimiento, utilice archivos codificados en columnas como ORC o Parquet y utilice el formato CSV únicamente para las tablas de dimensiones muy pequeñas.
- Utilice las particiones basadas en prefijos para aprovechar la poda de particiones. Esto significa que debe usar filtros que estén vinculados mediante claves a las particiones del lago de datos.
- Redshift Spectrum se escala automáticamente para procesar solicitudes grandes, así que debería hacer todo lo posible en Redshift Spectrum (por ejemplo, inserir predicados).
- Preste atención a los archivos de partición en las columnas que se filtran frecuentemente. Si los datos se dividen en una o más columnas filtradas, Redshift Spectrum puede aprovechar la poda de particiones y omitir los análisis en busca de particiones y archivos innecesarios. Una práctica habitual es particionar datos según el tiempo.
- Puede comprobar la eficacia de sus particiones y la eficiencia de su consulta de Redshift Spectrum mediante la siguiente consulta.

```
Select query,
           segment,
           max(assigned_partitions) as total_partitions,
           max(qualified_partitions) as qualified_partitions
From svl_s3partition
Where query=pg_last_query_id()
Group by 1,2;
```

En la consulta anterior se muestra lo siguiente:

- total_partitions — El número de particiones reconocidas por AWS Glue Data Catalog
- qualified_partitions: número de prefijos en Amazon Simple Storage Service (Amazon S3) a los que se accede para la consulta de Redshift Spectrum

- También puede comprobar la tabla del sistema `SVL_S3QUERY_SUMMARY` para obtener información sobre la eficacia de sus particiones y la eficiencia de la consulta de Redshift Spectrum. Para ello, utilice la siguiente instrucción.

```
Select *
From svl_s3query_summary
Where query=pg_last_query_id();
```

  En la consulta anterior se devuelve incluso más información, incluidos los valores `is_partitioned`, `s3_scanned_rows/bytes` y `s3_returned_rows/bytes`, además de archivos que indican la eficacia de la poda de particiones.

## Inserción de predicados en Redshift Spectrum

El uso de la inserción de predicados evita consumir recursos del clúster de Amazon Redshift. Puede inserir muchas operaciones de SQL a la capa de Redshift Spectrum. Recomendamos que aproveche esta opción siempre que sea posible.

Tenga en cuenta lo siguiente:

- Dentro de la capa de Redshift Spectrum puede evaluar algunos tipos de operaciones de SQL completamente, incluidas las siguientes:
  - Cláusulas `GROUP BY`
  - Condiciones de comparación y búsqueda de coincidencias de patrones (por ejemplo, `LIKE`)
  - Funciones de agrupación (por ejemplo, `COUNT`, `SUM`, `AVG`, `MIN` y `MAX`)
  - `regex_replace`, `to_upper`, `date_trunc` y otras funciones
- Algunas operaciones, como `DISTINCT` y `ORDER BY`, no se pueden enviar a la capa de Redshift Spectrum. Realice `ORDER BY` solo en el nivel superior de la consulta si es posible, ya que la clasificación se realiza en el nodo principal.
- Examine el plan `EXPLAIN` de la consulta para comprobar si la inserción de predicados es efectiva. Para encontrar partes de Redshift Spectrum en un comando `EXPLAIN`, busque estos pasos:
  - S3 Seq Scan
  - S3 HashAggregate
  - S3 Query Scan
  - Escaneo marino PartitionInfo

  - Partition Loop
- Utilice el menor número de columnas en la consulta. Redshift Spectrum puede eliminar las columnas para analizar si los datos están en formato Parquet u ORC.
- Haga un uso extensivo de las particiones para el procesamiento en paralelo y la eliminación de particiones, e intente que el tamaño de los archivos sea de al menos 64 MB si es posible.
- Establezca `TABLE PROPERTIES 'numRows'='nnn'` si usa `CREATE EXTERNAL TABLE` o `ALTER TABLE`. Amazon Redshift no analiza las tablas externas para generar las estadísticas de las tablas que el optimizador de consultas emplea a la hora de crear un plan de consulta. Si no se establecen estas estadísticas, Amazon Redshift supone que las tablas externas son las más grandes, mientras que las tablas locales son las más pequeñas.

## Consejos de ajuste de consultas para Redshift Spectrum

Le recomendamos que tenga en cuenta lo siguiente para ajustar las consultas:

- El número de nodos de Redshift Spectrum que el clúster de Amazon Redshift puede utilizar para una consulta depende del número de sectores del clúster.
- Aumentar el tamaño el clúster puede beneficiar a los perfiles de computación locales, los perfiles de almacenamiento y las capacidades de consulta de la consulta del lago de datos de Amazon S3.
- El planificador de consultas de Amazon Redshift envía predicados y agrupaciones a la capa de consultas de Redshift Spectrum siempre que sea posible.
- Cuando se devuelven grandes cantidades de datos de Amazon S3, el procesamiento se ve limitado por los recursos del clúster.
- Como Redshift Spectrum se escala automáticamente para procesar solicitudes de gran tamaño, su rendimiento general mejora siempre que pueda transferir el procesamiento a la capa de Redshift Spectrum.

# Recursos

- Prácticas recomendadas de Amazon Redshift (documentación de Amazon Redshift)
- Prácticas recomendadas de Amazon Redshift para el diseño de consultas (documentación de Amazon Redshift)
- Ajuste del rendimiento de las consultas (documentación de Amazon Redshift)
- Plan de consultas (documentación de Amazon Redshift)
- Mejora del rendimiento de las consultas de Amazon Redshift Spectrum (documentación de Amazon Redshift)
- Descripción del ciclo de vida de las consultas en Amazon Redshift (Recomendaciones de AWS)

# Historial de documentos

En la siguiente tabla, se describen cambios significativos de esta guía. Si quiere recibir notificaciones de futuras actualizaciones, puede suscribirse a las notificaciones RSS.

| Cambio | Descripción | Fecha |
| --- | --- | --- |
| Se eliminó AQUA | Eliminamos la informaci ón sobre Advanced Query Accelerator (AQUA). | 14 de junio de 2024 |
| Publicación inicial | — | 3 de febrero de 2023 |

Las traducciones son generadas a través de traducción automática. En caso de conflicto entre la traducción y la version original de inglés, prevalecerá la version en inglés.