1. Sobre Dremio.

Los datos están almacenados en un data lake. Es muy lento consultar con eficacia temporal, por lo que la mayoría de las organizaciones utilizan una combinación de data warehouses, extractos, cubos, y ETL para hacer que los datos en los lagos de datos sean utilizables.

Esto crea un análisis de datos complejo y costoso de infraestructura. Lo cual es lamentable, porque

el almacenamiento de data lake en sí es económico, fácil de provisión y escala, y ya tiene un adjunto

ecosistema de herramientas y soluciones. En muchas organizaciones, también es el primer lugar donde los datos aterrizan después de ser generados. Si el almacenamiento del data lake se puede consultar directamente en un forma eficaz, muchas de las soluciones adicionales que se ponen en marcha para sortear estos problemas de rendimiento podrían simplificarse o eliminarse por completo.

El Dremio's data lake engine proporciona rendimiento, junto con una capa semántica de autoservicio. Y hace esto de una manera que mantiene la flexibilidad inherente a el almacenamiento del data lake; no tienes que subir tus datos a un tercero o ponerlo en un formato propietario. Dremio proporciona:

* consultas rápidas, directamente en el almacenamiento del data lake. Tecnologías de Dremio como Data Reflections, Columnar Cloud Cache (C3) y Predictive Pipelining junto con Apache Arrow para ejecutar consultas en el lago de datos almacenamiento a velocidad interactiva.
* Una capa semántica de autoservicio. Esta capa de abstracción permite que TI aplique seguridad y significado comercial, al tiempo que permite a los analistas y científicos de datos para explorar datos y derivar nuevos conjuntos de datos virtuales (tablas virtuales).
* Flexibilidad y apertura. Dremio deja a las organizaciones evitan el bloqueo de proveedores, consulta a través de las nubes y mantenga sus datos en el almacenamiento que tú controlas.
* Se une con cualquier cosa. Mientras que muchos las organizaciones tienen muchos datos en almacenamientos de data lake, los datos a menudo también se encuentran en otros lugares. Dremio hace que sea fácil unir el almacenamiento del data lake con otras fuentes de datos, sin usar ETL.

1. Un cluster de Dremio:

Dremio presenta una arquitectura escalable. Es diseñado para escalar de un servidor a miles de servidores en un solo clúster.

Profundizar más aquí.

1. Roles de los nodos:

En un cluster de Dremio hay dos roles principales. Cada rol se puede escalar de forma independiente:

* Coordinadores: Estos nodos son responsables para coordinar la ejecución de consultas, gestionar metadatos y al servicio de la interfaz de usuario. Aplicaciones de clientes, así como herramientas de BI, se conectan y comunican con otros nodos coordinadores. Los coordinadores están altamente disponibles y pueden ser ampliados para procesar más clientes concurrentes.
* Ejecutores. Estos nodos son responsables de la ejecución de la consulta. Las aplicaciones clientes no conectan a los ejecutores. Los ejecutores pueden ser ampliado para procesar volúmenes de datos más grandes y más consultas concurrentes. Debido a que los ejecutores no tienen estado, las implementaciones pueden tratar estos nodos como recursos elásticos y escalar el sistema de forma dinámica.

1. Aceleración

Dremio utiliza una representación física altamente optimizada de los datos de origen llamada Data Reflections. El almacenamiento de los Data Reflection puede estar en la nube como Amazon S3, Azure ADLS, HDFS o almacenamiento de conexión directa (DAS). El tamaño de los Data Reflection Store tienda puede exceder el tamaño físico de memoria. Esta arquitectura permite a Dremio acelerar más datos a un costo menor, lo que resulta en una proporción de aciertos de caché mucho mayor en comparación con arquitecturas tradicionales solo de memoria. Los Data Reflections son utilizadas automáticamente por el optimizador de Dremio basado en costos en el momento de la consulta.

Tenga en cuenta que los Data Reflections son invisibles para el final

usuarios. A diferencia de los cubos OLAP, las tablas de agregación, y BI extrae, el usuario no está explícitamente conectado a un Data Reflections. En cambio, los usuarios emiten consultas contra el modelo lógico de Dremio, y el optimizador de acelera automáticamente la consulta

aprovechando los Data Reflections que son adecuados para dicha consulta basada en el grafo de dependencia del optimizador y análisis de costos.

Cuando el optimizador no puede acelerar la consulta, Dremio utiliza su motor alto rendimiento de ejecución distribuido, aprovechando su procesamiento columnar en memoria (a través de Apache Arrow) y pushdown avanzados en las fuentes de datos subyacentes donde están disponibles:

FOTO

En los casos en que se pueda satisfacer toda la consulta por Data Reflections, sin consultar las fuentes de datos, el plan es reescrito por el optimizador para utilizar los Data Reflections:

Hay, por supuesto, situaciones donde el plan óptimo incluye una combinación de datos de los datos fuente y una o más reflexiones de datos.

1. Data Reflections:

Un Data Reflections se representa como un plan lógico y la materialización física correspondiente. El plan lógico puede basarse en uno o más conjuntos de datos físicos (p. ej., depósito S3, archivo de registro, registro) directorio, tabla de colmena, tabla de Oracle). A partir de un punto de vista de gestión, un Data Reflections es anclado a un único conjunto de datos físico o virtual, aunque en el momento de la consulta puede acelerar las consultas

en otros conjuntos de datos también.

La materialización física de Data Reflections está basado en Apache Parquet, con una variedad de optimizaciones circundantes como el nivel de columna Estadísticas. Los Data Reflections se pueden ordenar, particionado, agregado y distribuido por columnas específicas. Para comprender cómo funcionan los Data Reflections, nosotros debemos responder lo siguiente independiente

preguntas:

1. ¿Cómo se gestionan los Data Reflections? ¿Como son creados y actualizados?

2. ¿Cómo se usan los Data Reflections para acelerar consultas?

3. ¿Cómo se almacenan físicamente los Data Reflections? y optimizado?

1. Gestion de Data Reflections

Desde el punto de vista de la gestión, los Data Reflections siempre están anclados a un conjunto de datos específico físico o virtual en el sistema. Mientras las consultas SQL

en el conjunto de datos X a menudo se acelerará a través de Data Reflections ancladas al conjunto de datos X, el optimizador es libre de usar cualquier otro Data Reflections para

acelerar tales consultas. Además, los datos se pueden usar reflexiones ancladas al conjunto de datos X por el optimizador para acelerar las consultas en el conjunto de datos Y.

Hay dos tipos de reflexiones de datos:

• Raw Reflflectitions: una proyección de uno o más columnas del conjunto de datos de anclaje. Los datos pueden ser ordenado, particionado y distribuido por diferentes columnas del conjunto de datos de anclaje

• Aggregatition Reflflectitions: una agregación en una o más columnas del conjunto de datos de anclaje. El data reflection se define por dimensiones y medidas, y contiene nivel agregado datos para cada una de las medidas, como el recuento, suma, min y max. Los datos se pueden ordenar, particionado y distribuido por diferentes columnas del conjunto de datos de anclaje.

Puede haber circunstancias únicas en las que es deseable para crear un data reflection personalizado, similar a una vista materializada en una relación base de datos. En tales casos, simplemente cree un nuevo conjunto de datos virtual con la consulta SQL que define

la materialización deseada, y crear una sola Reflexión sin formato que incluye todas las columnas en el conjunto de datos virtual.

1. Dremio actualiza automáticamente los data Reflections para garantizar la actualización de los datos. Hay tres refrescos métodos:

• Actualización completa. Este método es el más adecuado para conjuntos de datos mutantes.

• Actualización incremental para fuentes basadas en archivos. Este método es más adecuado para grandes, conjuntos de datos de solo agregar que se representan como una colección de archivos (p. ej., S3). El sistema identifica automáticamente nuevos archivos en el directorio.

• Actualización incremental para fuentes basadas en tablas. Este método es más adecuado para grandes, conjuntos de datos de solo agregado representados como tablas o colecciones (por ejemplo, Oracle, Elasticsearch, MongoDB). Un aumento monotónico columna, como una marca de tiempo created\_at o Se requiere clave primaria secuencial.

Internamente, Dremio mantiene un grafo de dependencia (DAG) que define el orden en que los data reflections se actualizan. Las dependencias son calculadas en base al álgebra relacional, y el tiempo de inicio de actualización actual tiene en cuenta la cantidad de tiempo esperada requerida para completar todo el ciclo de actualización.

Tenga en cuenta que este enfoque basado en grafos reduce el tiempo de ciclo de extremo a extremo, así como el cálculo recursos necesarios para completar el ciclo. Además, al aprovechar un data reflection para actualizar otra reflexión de datos, el sistema puede evitar lecturas intensivas de recursos en funcionamiento bases de datos más de una vez.

1. Data reflection externos

Dremio admite la noción de Data reflection externos. Los usuarios pueden crear y mantener Data reflection utilizando un proceso externo como Apache Spark, luego registre los Data reflection externos en Dremio. Dremio considerará estos Data reflection en su análisis basado en costos para acelerar las consultas. Cualquier fuente de datos que Dremio soportes se pueden usar para Data reflection externos.

Los Data reflection externos son útiles en casos donde tiene sentido gestionar la creación fuera del proceso de Dremio. Por ejemplo, para procesamiento que se ejecuta durante muchas horas o días, o cuando ya existen procesos existentes que crea representaciones optimizadas de datos para patrones de consulta específicos. La ventaja de registrar estos recursos en Dremio es que simplifica la experiencia para los consumidores de datos, y proporciona capacidades adicionales para asegurar y que rige el acceso a los datos, así como linaje de datos de seguimiento.

1. Usando Data Reflection para acelerar consultas

Dremio incluye un optimizador basado en costos que no solo planifica y optimiza consultas, pero también explora oportunidades para utilizar Data Reflections para reducir el costo de la consulta (es decir, acelerar la consulta). Cuando llega una nueva consulta, el optimizador considera todas los Data Reflection y automáticamente reescribe el plan de consulta para utilizar Data Reflection cuando sea posible. El optimizador utiliza dos fases. algoritmo:

1. Poda de Data Reflection. El optimizador no tiene en cuenta los Data Reflection que son irrelevantes porque sus planes lógicos no tienen conjuntos de datos físicos en común con el plan lógico de las consultas.

2. Emparejamiento de subgrafos. El optimizador usa un innovador algoritmo de grafo jerárquico para hacer coincidir sub-grafos del plan lógico de la consulta con los planes lógicos de Data Reflections.

1. Vida de una consulta:

Las aplicaciones del cliente emiten consultas a Dremio sobre ODBC, JDBC o REST. Una consulta puede involucrar a uno o más conjuntos de datos, que potencialmente residen en diferentes

fuentes de datos. Por ejemplo, una consulta puede ser una unión entre S3, Elasticsearch y varias tablas de Oracle.

Dremio utiliza dos técnicas principales para reducir la cantidad de procesamiento requerida para una consulta:

• Push-down en las fuentes de datos subyacentes. El optimizador considerará las capacidades de las fuentes de datos subyacente y el relativo costo. Luego generará un plan que realiza etapas de la consulta en la fuente o en el entorno de ejecución distribuida de Dremio para lograr el más eficiente plan general posible.

• Aceleración vía Data Reflection:

El optimizador usará Data Reflections para porciones de la consulta cuando esto produce el plan general más eficiente. En muchos casos toda la consulta puede ser atendida desde Data Reflection ya que pueden ser órdenes de magnitud más eficiente que el procesamiento

consultas en la fuente de datos subyacente

1. Query Push Down

Dremio puede empujar el procesamiento hacia fuentes de datos tanto relacionales como no relacionales. Las fuentes de datos no relacionales generalmente no admiten SQL y tiene una ejecución limitada capacidades. Un sistema de archivos, por ejemplo, no puede aplicar predicados o agregaciones. El optimizador de Dremio comprende las capacidades de cada fuente de datos. Cuando sea más eficiente, Dremio empujará la mayor cantidad de consultas a la fuente subyacente como sea posible, y realiza el resto por su cuenta motor de ejecución distribuida.

1. Ejecución de una consulta:

La vida de una consulta incluye las siguientes fases:

1. El cliente envía una consulta al coordinador a través de ODBC / JDBC / REST

2. Planificación:

1. El coordinador analiza la consulta en modelo relacional universal de Dremio.

2. El coordinador considera las estadísticas disponibles. sobre fuentes de datos para desarrollar un plan de consulta, así como habilidades funcionales de la fuente.

3. El coordinador reescribe el plan de consulta para usar

(1) los data reflection disponibles, considerando el ordenado, particionado y distribución de los data reflections; y

(2) las capacidades disponibles de las fuentes de datos.

3. Ejecución:

1. Los ejecutores leen los datos en los buffers de Arrow de fuentes en paralelo. Los datos pueden provenir de los Data Reflections (archivos de Parquet) o las fuentes de datos subyacentes. Al leer desde una fuente de datos, el ejecutor envía las consultas nativas (por ejemplo, Elasticsearch Painless Script, Microsoft Transact-SQL) según lo determinado por optimizador en la fase de planificación.

2. Los ejecutores ejecutan el plan de las consultas reescrito.

3. Un ejecutor combina los resultados de uno o más ejecutores y transmite los resultados finales al coordinador

4. El cliente recibe los resultados de coordinador

Tenga en cuenta que todas las operaciones de datos se realizan en el nodo ejecutor, permitiendo que el sistema escale a muchos clientes concurrentes usando solo unos pocos nodos coordinadores.

1. Apache Arrow y Apache Parquet:

Dremio utiliza un almacenamiento columnar de alto rendimiento y ejecución, con tecnología Apache Arrow (columnar en memoria) y Apache Parquet (columnar en disco). De hecho, el equipo ingenieros de Dremio incluye a los cocreadores de Apache Arrow.

1. Apache Arrow:

Apache Arrow es un proyecto de código abierto que permite el procesamiento de datos en memoria en columna e intercambio. Dremio fue parte de la fundación equipo detrás de Arrow, que ahora incluye comprometidos de varias compañías incluyendo IBM, Cloudera, Databricks, Hortonworks, Intel, MapR y Two Sigma.

Dremio es el primer motor de ejecución construido a partir desde cero en Apache Arrow. Internamente, los datos en memoria se mantienen fuera del montón en el formato de Arrow, y pronto habrá una API RPC que devuelve los resultados de la consulta en buffers de Arrow en memoria.

Una variedad de otros proyectos ha adoptado Arrow también. Python (Pandas) y R están entre estos proyectos, permitiendo a los científicos de datos trabajar más eficientemente con datos. Por ejemplo, Wes McKinney, creador de la popular biblioteca Pandas, demostró recientemente cómo Arrow permite Usuarios de Python leer datos en Pandas a más de 10 GB / s.

1. Apache Parquet

Apache Parquet es un proyecto de código abierto que habilita el almacenamiento de datos en columnas. Ha surgido en últimos años como el formato columnar más común en los ecosistemas Hadoop y AWS.

A diferencia de Apache Arrow, que está optimizado para almacenamiento en memoria y procesamiento eficiente en el CPU, Parquet está optimizado para el almacenamiento en disco.

Por ejemplo, utiliza codificación y compresión de esquemas, como diccionario y longitud de ejecución codificación, para minimizar la huella general y el almacenamiento I\O.

Dremio incluye un ultra alto rendimiento, lector de parquet vectorizado que lee Datos con formato de parquet del disco en Datos con formato de Arrow en la memoria. El lector de Parquet también permite el procesamiento rápido de datos sin procesar como data reflection. Incluye lo último en tecnología, capacidades tales como:

• Versiones inteligentes y poda de página

• Operaciones en el lugarsin descomprimir datos

• Procesamiento vectorizado

• Cero copias de memoria

1. Seguridad y gobernanza de datos:

La seguridad y el gobierno de datos son críticos para cualquier empresa. Sin embargo, la complejidad creciente y la demanda de datos a menudo conduce a la expansión de datos,

resultando en un riesgo significativo. Dremio habilita analistas de negocios y científicos de datos para descubrir, seleccionar, acelerar y compartir datos sin tener para exportar copias de los datos a sistemas no gobernados, incluidas hojas de cálculo desconectadas, servidores de BI no gobernados y bases de datos privadas.

Esto reduce el riesgo de acceso no autorizado a los datos, así como el robo de datos. Dremio ofrece una variedad de seguridad y capacidades de gobernanza que incluyen:

• Control de acceso a filas y columnas en cualquier fuente

• Procedencia de datos y linaje de datos.

• Autenticación basada en LDAP / AD

• Control de acceso basado en roles.

• Auditoría

• Cifrado

1. Procedencia de los datos, linaje de datos.

El gráfico de datos de Dremio muestra la procedencia y linaje de cada conjunto de datos en el sistema, incluidos aquellos en el espacio personal de un usuario. TI puede fácilmente entender cómo se creó un conjunto de datos, transformado, unido y compartido, así como el linaje completo de estos pasos entre conjuntos de datos. Esta información es capturada y rastreada por Dremio automáticamente.

Data Reflection

Los Data Reflection son esencialmente estructuras de datos precomputadas y usadas durante la ejecución de la consulta. Estas estructuras de datos mantienen datos en formatos que son lógicamente y físicamente optimizados. Un Data Reflection está asociado a un conjunto de datos físicos o virtual. Debido a que un conjunto de datos virtual puede representar la combinación de múltiples conjuntos de datos, Data Reflection puede incluir datos de múltiples fuentes.

Los planificadores de consulta de Dremio es capaz de usar un Data Reflection para acelerar consultas en diferentes conjuntos de datos que pueden compartir unos o más ancestros comunes. Esta flexibilidad incrementa la probabilidad de que la consulta se pueda acelerar, y permite a los administradores mantener un número relativamente bajo de Data Reflections para acelerar las consultas.

A alto nivel un Data Reflections está definida por una expresión SQL que representa el árbol expandido de todos los conjuntos de datos virtuales ancestros hasta las fuentes de datos físicas. Considerar el siguiente ejemplo: