System Catalog

---------------------

En líneas generales, los DBMS llevan estadísticas y hacen un tracking de información respecto a la estructura de la base de datos. Esto con el objetivo de desarrollar otras tareas de forma eficiente.

Esta información relevante para el DBMS se encuentra en el System Catalog o Catalogo, que puede simplificarse como un conjunto de tablas que el DBMS mantiene para su propósito propio. La información del catálogo está asociada a describir las tablas existentes, las vistas, las columnas, los privilegios de distintos tipos de usuarios y otras estructuras de la base de de datos. En resumen, es información que refleja la estructura de la base de datos en cuestión. Y esta información está contenida en tablas que son generadas automáticamente por el DBMS al momento de creación de la base de datos.

Si bien estas tablas solo pueden ser modificadas, creadas o eliminadas por el propio DBMS como un mecanismo para evitar errores destructivos por parte de los usuarios, la información contenida en el mismo suele ser accesible por medio de ciertas vistas provistas por el propio DBMS.

Contenido del Catálogo:

Cada tabla contenida en el catálogo contiene información de un tipo concreto de estructura presente en la base de datos:

* Tablas: Describe cada tabla por medio de su nombre, propietario, columnas, tipo de datos, etc.
* Columnas: Cada columna se describe mediante su nombre, tabla de pertenencia , tipo de dato, tamaño, comportamiento ante NULL, etc.
* Usuarios: Describe a los usuarios autorizados, su nombre de usuario, su clave encriptada, etc.
* Vistas
* Privilegios

Information Schema

El standard SQL no especifica qué tablas debe soportar el DBMS sino que indica qué vistas debe hacer accesibles al usuario para hacer visible la estructura de la base de datos. Este conjunto de vistas se denomina INFORMATION\_SCHEMA

Estadísticas y su interpretación

Las estadísticas de una base de datos juegan un rol central al momento en que un DBMS requiere definir o construir un plan de ejecución (para una dada consulta).

Su influencia es incluso más relevante que la de los propios indices.

Es importante entender que todos los índices tienen una estadística asociada pero no todas las estadísiticas tienen un ínidice asociado. Las estadísticas nos permiten determinar, entre otras cosas, qué indices son fructiferos para reducir el coste de una o varias consultas.

Las estadísticas se componen de tres items. Y para cada estadística se debe indicar el índice o tabla sobre la cuál se quiere consultar y el nombre de la estadística:

* Encabezado o header:
  + Nombre
  + Updated: Fecha de la última actualización de la estadísitca mostrada
  + Filas: Son las filas totales de la estadística (ya sea de un índice o tabla) al momento en que fue generada. Una forma sencilla de analizar si una estadística está actualizada es comparar las filas actuales de la tabla con las utilizadas para computar la estadística. Aunque hay que tener presente si la misma se hizo sobre una versión filtrada.
  + Filas muestreadas: nro de filas utilizadas para generar la estadística
  + Steps: Nro de intervalos en el histograma
  + Average Key lenght: Longitud promedio de la columna
* Density vector: Es una medida de la distribución de valores en la tabla. Tiene una entrada por nro de columnas en la tabla
  + Densidad: Nro entre 1 y 0 que representa la cantidad de valores distinguibles (1/n). cuanto más cerca de 0 más único es el valor del campo. Las columnas que son clave candidata son las que tienen el valor más bajo
  + Average Length:
  + Columns: columnas que constituyen el prefijo.
* Histogram: Es una representación de la distribución de valores en una tabla. Esta representación se realiza definiendo un nro dado de intervalos
  + RANGE\_HI\_KEY: Valor que define el extremo superior del rango
  + RANGE\_ROWS: Nro de filas promedio que se contienen en cada rango o barra, sin contemplar aquellas que son iguales al valor extremo
  + EQ\_ROWS: Nro estimado de filas que son iguales al valor del extremo superior
  + DISTINCT\_RANGE\_ROWS: Nro estimado de valores distinguibles en cada rango, sin considerar el valor extremo
  + AVG\_RANGE\_ROWS: RANGE\_ROWS/DISTINCT\_RANGE\_ROWS

Clustered tables/Sorted Tables y Heap Tables

En el contexto de SQL, Clustered tables y sorted tables son términos equivalentes, ya que refieren a una tabla que está ordenada en base a un dado atributo/columna. Por otra parte, Heap tables son tablas que no están ordenadas en función de ninguna columna.

Discutimos a continuación las implicaciones a nivel práctico de una y otra implementación:

* Heap tables:
  + Información ordenada sin un orden aparente
  + No podemos recuperar la información con rapidez, a menos que contemos con otros indices (de tipo non-clustered) asociados a la tabla
  + Las páginas donde se almacenan los datos de la tabla no están vinculadas entre si, es decir que, finalizada una página debemos analizar el IAM(index allocation map) para determinar la siguiente página a escanear.
  + No requerimos tiempo y procesamiento adicional para mantener el índice ya que este no existe
  + No requerimos espacio para almacenar el árbol de indice
* Clustered/sorted files
  + La información (entradas de la tabla) están almacenados en el orden definido por la clave del ínidce clustered
  + Podemos recuperar rápidamente información de la tabla, si la consulta se basa o utiliza el atributo de indexación
  + Las páginas asociadas a la tabla están vinculadas entre si para un rápido acceso.
  + Se requiere tiempo adicional para mantener actualizado el index ante operaciones de UPDATE, DELETE, INSERT
  + Se requiere espacio adicional para almacenar el árbol del índice

Fragmentación: Es un problema que aparece en las tablas frente a las operaciones de DELETE, INSERT y UPDATES.

La Fragmentación conlleva un uso ineficiente del espacio y del tiempo necesario para recorrer las tablas en las consultas.

* Heap files:
  + Si solo se realizan INSERTS, la tabla no se fragmenta ya que solo se escribe nueva información al final de la misma.
  + Si existen varias operaciones, no necesariamente los espacios liberados serán utilizados para almacenar los nuevos datos
* Sorted files:
  + Si solo se realizan INSERTS, la tabla no se fragmenta si el atributo de indexación es secuencial.
  + Pueden ser necesario dividir páginas en múltiples páginas para almacenar la información en el orden especificado por el atributo de indexación.

En principìo, se estima que una tabla sorted/clustered será menos fragmentada que una heap.

Soluciones para la fragmentación

* Clustered: Reconstruir y reorganizar la tabla en base al índice
* Heap:
  + Crear un índice clustered
  + Ordenar creando una nueva tabla en base a la ya existente

Clustered y Non-clustered index

Los índices clustered son ínidices que almacenan la información, directamente. Así operan en SQL. En cambio, el índice nonclustered almacena un puntero a la información. Por esta razon, una tabla solo puede tener un índice clustered. Y, a su vez, una tabla que no cuenta con un índice clustered es una tabla heap ( De acuerdo a lo explicado antes)

La utilidad de los ínidices es acceder a la información o recuperar la información solicitada por una consulta con rapidez. De no contar con un índice, la alternativa es realizar un Table scan, lo cual conlleva un gasto de procesamiento considerablemente mayor (en la mayoría de los casos)

Generalmente, el costo computacional de realizar una búsqueda mediante un índice non-clustered es superior al de un índice clustered. Ya que en uno contamos con la información en el mismo índice mientras que en el otro optenemos el puntero a RID donde iremos a buscar la información. El único caso en que esto no es así es cuando contamos con la información a recuperar en el propio nodo hoja del árbol del índice non-clustered, es decir, solo el valor del atributo en base al cual fue creado el índice.

Cómo leer las métricas de un plan de ejecución

Tanto las operaciones como las flechas que llevan información entre las operaciones incluyen información sobre el plan de ejecución:

* Physical Operation: La operación ya esa Join, Seek, Scan, etc
* Logical Operation: Es la operación implementada, es decir, ya optimizada
* Estimated I/O cost: No funciona como un parámetro absoluto, sino relativo para comparación entre distintos planes
* Estimated CPU cost: idem
* Estimated Operator cost: suma de los anteriores dos.
* Estimated Subtree cost: Costo acumulado del arbol hasta la operación actual
* Estimated number of rows: Se basa en las estabísticas disponibles
* Estimated row size: idem
* Ordered: Valor boolean que indica si el output está ordenado en cierta forma
* NodeID
* Actual Nomber of Rows:
* Actual Rebinds / Rewinds: No vamos a abordarlos. Pero básicamante, Rebinds está asociado a reordenar los parámetros a utilizar en un join u operación similar y reevaluar el costo de la operación. Mientras que Rewind refiere a si manteniendo una forma menos eficiente de la operación actual puedo volver más eficiente una operación posterior (como por ejemplo, un ordenamiento util o interesante)
* Cached plan size: Refiere al estpacio en memoria que el plan de ejecución va a ocupar en el cache de Store procedures.

Predicado: Término que es representado en el nodo ya sea un filtrado, descripción o comparación de datos.

Object: Elemento utilizado para realizar la operación. Por ejemplo, la clave primaria, un índice, etc

Output List: Son los atributos que se obtienen a la salida

Bibliografía:

1. James Groff and Paul Weinberg. 2009. SQL The Complete Reference, 3rd Edition (3rd. ed.). McGraw-Hill, Inc., USA.