Quiz #5 y #6

Fecha: 21/10/2022
Autor: Deyan Sanabria Fallas #2021046131

Explique en que consiste un clustered index y cuál es la diferencia entre este y un índice non-clustered que utiliza INCLUDE para agregar columnas al índice.

Clustered Index

Un clustered index es un indice creado en disco que vive junto a los datos, suele hacerse con un arbol enario y nos brinda la ventaja de no tener que ir a buscar en un puntero por los datos y tener que abrir otro archivo, con la desventaja que cualquier modificacion a la columna indexada va a provocar que se tenga que reordenar los datos, lo cual lo hace costoso. Util usarlo en situaciones donde las columnas no se modifiquen con regularidad.

Diferencia con un indice non-clustered que utiliza INCLUDE

Una gran diferencia con respecto a un non-clustered index con INCLUDE va a tener mucha repeticion de datos. Un non-Clustered esta montado sobre un heap que tiene punteros a la ubicacion dentro de un archivo donde se pueden encontrar los datos, pero adicionalmente, en las hojas se suele guardar el valor directo de la columna indexada.

Esto causa que tengamos datos sobre el heap y en el archivo a donde apunta el puntero, por lo que pierde bastante la gracia. Ademas, se recomienda el uso de un non-clustered index cuando los datos cambian de forma frecuente, pero con muchos datos en las hojas del heap, vamos a tener que modificar multiples lugares cuando ocurran modificaciones.

Explique el concepto de memory footprint y como afecta este la creación de índices. ¿Cuál es la relación entre un memory footprint alto y la paginación a disco?

Como su nombre lo indica, la *huella de memoria* es la huella que deja alguna pieza de software en la memoria durante su ejecucion, esto significa que entre mas "memory footprint" deje un programa, mas memoria va a utilizar.

La gracia de un indice es que sea rapido a la hora de realizar busquedas, el problema es que debe entrar el indice entero dentro de la memoria para la busqueda. Si este indice no cabe en la memoria, el sistema se vera en la obligacion de intercambiar datos entre memoria y disco conforme necesita mas datos del indice, lo que nos lleva al siguiente punto.

La relacion que tiene la paginacion con un alto memory footprint es ese proceso que ocurre cuando toda la memoria esta siendo ocupada, como por ejemplo, como se dijo anteriormente, un indice que no entra completo en la memoria, el sistema va a cargar todo lo que pueda en memoria y conforme se necesitan nuevos datos del indice que no se pudieron cargar a memoria, estos se obtendran de disco, este proceso se le llama paginacion, pues vamos intercambiando paginas entre el disco y la memoria para obtener los datos que no entraron anteriormente.

Dicho esto, la paginacion se da cuando tenemos un alto memory footprint, pues si llenamos la memoria por completo, el sistema esta obligado a paginar.

¿Que tipo de base de datos recomendaría a TooSlow para almacenar sus datos?

Contexto

> **FAST**antic Inc es una empresa especializada en optimización de búsquedas sobre datos, está a sido contratada por la empresa TooSlow para ayudarle a organizar 40 billones de registros, los registros tienen las siguientes columnas:
>
> - country: este es un código de país
> - city: está es una ciudad en un país específico.
> - date: está es la fecha en que el registro fue agregado a los datos.
> - payload: es un documento JSON que contiene el evento.
>
> **FAST**antic Inc debe optimizar la búsqueda sobre las columnas country, city y date. Explique la mejor forma de organizar los datos para incrementar la velocidad de búsqueda, actualmente se hace un scan sobre todos los datos. Asuma que no existe una base de datos mencione estructuras de datos que utilizará

Recomendacion

Para un caso como este, lo ideal es ver la cual seria el uso de la base de datos, si bien ese cierto que es una base de datos que busca ser optimizada para busquedas, hay que comprobar que tipo de busquedas se requieren realizar y cuales suelen ser las mas frecuentes.

Por ejemplo, se puede observar un espacio para almacenar fechas, se podria hacer un analisis extenso para ver datos de que fechas son mas frecuentados y montar una base de datos *time series* con diversos data tiers acorde a lo obtenido por el analisis. Si existen datos muy viejos que no se suelen consultar, buena idea seria un frozen data tier, asi bajamos costes.

Otra buena idea puede ser una base de datos *SQL* donde la columna del payload nos indique donde se guarda el documento JSON del espacio *payload*, pero dependera del acceso hacia tal documento y como se gustaria acceder.

Lo que es claro es que el cliente necesita velocidad sobre las columnas de country, city y date. En una base de datos SQL se podrian hacer ciertas optimizaciones para estas busquedas. Iniciando por indexacion, va a depender de las peticiones que se hagan sobre la tabla el como podemos realizar la indexacion. Si los datos no se modifican con frecuencia, se podria montar un *clustered index* sobre alguna de las columnas,

preferiblemente la mas frecuentada, ya que no podemos tener multiples *clustered index* y este tipo de indice es muy rapido, gracias a montarlo en un *arbol enario*, siempre procurando mantener la altura del arbol lo mas pequeña posible.

Ahora, se podria implementar varios *non-Clustered index*, pero igual, es necesario saber el tipo de busquedas que se realizaran. Por ejemplo, si se hacen busquedas por las columnas de country y city, hacer un indice acorde, y tambien dependera si se ordena de cierta manera, cual de las dos columnas debe quedar antes en la indexacion.

Lo anterior puede aplicarse sobre *bases de datos distribuidas*, de tal forma que tengamos un *hash table* que nos redireccione hacia ciertos servidores y en dichos servidores tener las indexaciones a como se plantearon anteriormente.

Hay que tener muy claro que hay que hacer lo justo y necesario, pasarse de indices o hace muy pocos nos va a penalizar en rendimiento, estas formas de indexacion pueden ser aplicadas como un *indice parcial* para intentar compensar. Tambien destacar que es bastante poco el contexto que existe, pues las recomendaciones planteadas no son mas que ideas que se pueden aplicar dependiendo del caso de uso que tenga la empresa, siempre hay que planear la base de datos alrededor de lo que le sirve al negocio.