Bases II GR 1 - Resumen 9 - 31/05/2023

Gerald Núñez Chavarría - 2021023226

Graph Databases for Beginners

1. Inverted Indexes

Un índice invertido es un método de almacenamiento de documentos en el que, en lugar de enumerar las palabras de cada documento, se almacena la lista de documentos en los que aparece cada palabra. Esto se hace para permitir una búsqueda y recuperación eficaces de los documentos a partir de palabras clave. Además de almacenar las listas de documentos, los índices invertidos también pueden almacenar las posiciones de las palabras dentro de los documentos para soportar consultas de frases y proximidad. El nivel de granularidad del almacenamiento de posiciones puede variar, pero normalmente se hace a nivel de palabra.

Algunas implementaciones de índices invertidos almacenan dos listas separadas: una para listas de documentos (y a menudo frecuencias de palabras) y otra para listas completas de posiciones de palabras. Esto permite recuperar más rápidamente los resultados de consultas sencillas consultando las listas de documentos más cortas. Algunas implementaciones también incluyen metainformación sobre la posición de cada palabra, como el tamaño de la fuente o el tipo de texto, que puede utilizarse para mejorar la clasificación de los resultados de búsqueda.

Existen variaciones en el almacenamiento del léxico, que se refiere a todos los tokens indexados en la colección. El léxico puede almacenar información estadística para cada token, como el número de documentos en los que aparece. El léxico puede ser útil para diversos fines.

El espacio utilizado por el índice invertido varía entre el 5 y el 100% del tamaño total de los documentos indexados. Este enorme rango existe porque las implementaciones del índice invertido presentan muchas variaciones diferentes. Hay algunas formas de almacenar la lista invertida comprimida, como: Variable length integers, Elias gamma, Golomb-Rice, Delta coding...

Construir un índice invertido puede ser todo un reto, ya que requiere gestionar la memoria, el espacio en disco y el tiempo de CPU de forma eficiente. Aunque los avances en hardware informático no suponen una ayuda significativa debido al rápido crecimiento de las colecciones indexadas, para colecciones pequeñas, la inversión en memoria es el enfoque más rápido y sencillo. Sin embargo, para colecciones grandes, los resultados temporales deben almacenarse en disco, y una estrategia eficaz consiste en construir inversiones en memoria de tamaño limitado, guardarlas en disco y fusionarlas para formar el índice invertido final.

2. Técnicas de índice invertido

A continuación se comentan técnicas utilizadas para implementar un motor de búsqueda utilizando un índice invertido.

Document preprocessing: Los documentos no se idexan directamente, si no que normalmente sufren una etapa de preprocesamiento que tiene las siguientes fases.

- Lexing: Es el proceso de convertir un documento de caracteres a tokens alfanuméricos. Se realiza una conversión a minúsculas y se separan los caracteres por espacios o puntuación. Los tokens con muchos números se eliminan. Los lenguajes ideográficos requieren tecnologías de búsqueda especiales.

- Stemming: El stemming es el proceso de transformar palabras a su raíz morfológica para indexarlas. Aunque tiene ventajas, como reducir el tamaño del índice, también presenta limitaciones y errores. Actualmente, muchos motores de búsqueda, como Google, evitan utilizar stemming debido a los problemas que genera. En su lugar, se utilizan consultas con comodines o la expansión de consultas.
- Stop words: Las palabras de parada son términos comunes como "a", "the", "of" y "to" que se ignoran al indexar documentos. Aunque ahorran espacio en el índice, pueden dificultar la búsqueda de información específica en ciertos casos. Aunque las palabras de parada pueden comprimirse mejor, los motores de búsqueda modernos, como Google, no suelen utilizarlas para no limitar la búsqueda de combinaciones de palabras.

Query types: Hay muchas maneras diferentes de buscar la información, a continuación algunas.

- Normal: Una consulta normal es aquella que no es especializada y busca documentos que contengan un término específico. Para consultas con múltiples palabras, hay diferentes enfoques: algunos usan operadores booleanos implícitos, mientras que otros permiten que falten algunos términos y clasifican los resultados en función de cuántos términos se encuentren. La segunda opción es más flexible pero más costosa de evaluar. La mayoría de los motores de búsqueda en Internet no la utilizan.
- Boolean: Las consultas booleanas son aquellas en las que los términos de búsqueda se conectan mediante los operadores lógicos AND, OR y NOT. Se implementan utilizando un índice invertido. El operador NOT debe combinarse con otros términos usando AND y luego eliminar los documentos que contienen el término NOT. El operador OR combina las listas de documentos de cada término mediante una unión, y el operador AND las combina mediante una intersección. El operador AND reduce el conjunto de resultados, mientras que el operador OR lo expande. Se pueden aplicar optimizaciones como recuperar primero los términos menos comunes y leer las listas de documentos en paralelo desde el disco. La elección de optimizaciones depende de las decisiones de implementación en el sistema de búsqueda.
- Phrase: Las consultas de frase se utilizan para encontrar documentos que contienen las palabras dadas en el orden especificado. Son útiles para encontrar documentos con palabras comunes usadas de manera específica. Las consultas de frase son más costosas porque además de las listas de documentos, también deben realizar un seguimiento de las posiciones de las palabras en cada documento que podrían ser la posición inicial de la frase buscada. Se han investigado índices auxiliares para mejorar la velocidad de las búsquedas de frases.
- Proximity: Las consultas de proximidad son de la forma "término1 CERCA (n) término2" y coinciden con documentos donde el término1 ocurre dentro de n palabras del término2. Son útiles en casos donde no se sabe cómo se listarán los nombres, por ejemplo. Las consultas de proximidad se implementan de manera similar a las consultas de frase, pero permiten una diferencia máxima de n en las posiciones de las palabras.
- Wildcard: Las consultas con comodines son una forma de coincidencia difusa o inexacta. Hay dos variantes principales: comodines para palabras completas y comodines dentro de palabras. Los comodines para palabras completas permiten dejar palabras completas sin especificar, mientras que los comodines dentro de palabras permiten dejar partes de una sola palabra sin especificar. Estas consultas pueden implementarse eficientemente como variantes de consultas de frase o mediante la expansión de los comodines y la ejecución de la consulta modificada. Sin embargo, las consultas con comodines dentro de palabras requieren un léxico explícito y son más costosas en términos de tiempo y posiblemente espacio, por lo que muchas implementaciones optan por no admitirlas.