Ensayo sobre Árboles B, B+, B-link

Jorge Luis Huanca Mamani

Febrero 2021

1 Introducción

Todos en algun momento nos hemos topado con bases de datos ya sea de manera directa o indirecta, con el pasar de los años las bases de datos han tomado suma relevancia a nivel personal, pero mas a nivel empresarial y esto se considera una de las mayores aportaciones que ah dado la computación a las personas, ¿pero que ocurre por detrás? ¿cómo es posible obtener un dato específico de un gran cúmulo? mejor aún ¿cómo lo hace tan rápido? son preguntas que a lo largo de este ensayo iremos respondiendo una a una.

Para nosotros poder almacenar información para luego acceder y/o manipularla es necesario pensar en algun tipo de estructura que nos permita hacerlo, si bien es cierto hay muchas estructuras que intentaron hacer esto, pero hay una en particular que no solo logró hacerlo de manera eficiente, si no que con el pasar de los años esta tuvo mejoras tras mejoras y es actualmente la mas utilizada en el ambito de bases de datos, estamos hablando de los Arboles B.

Todos los sistemas de bases de datos hacen uso de algún tipo de índice para acelerar la recuperacion de información, un índice es un mapeo de valores clave en una ubicación física.

Por ejemplo, en las bases de datos relacionales, un índice asigna valores de columnas a filas, para cuando un usuario envíe una consulta para recuperar filas de una tabla que tienen un valor determinado, el sistema de base de datos use dicho índice y acceda a las filas deseadas para poder entregar la informacion deseada, y no tener que leer todas las filas una a una.

2 Desarrollo

Los Árboles B son una generalización de los árboles balanceados, éstos representan básicamente un método para almacenar y recuperar información de medios externos.

Como ya se mencionó la estructura mas utilizada en sistemas de bases de datos es el Árbol B, un Árbol B consta de un conjunto de nodos la cual se denominan páginas y estas están organizadas como un ärbol, ahora si dichos nodos apuntan a otros nodos y estos otros nodos estan en un nivel inferior, se les denominan nodos internos, si los nodos no apuntan a ningun otro nodo en un nivel mas abajo estos se les denominan nodos hojas.

Si los nodos internos contienen valores clave y no los datos entonces se dice que es un Árbol B+, quizas una pregunta comun es ¿qué caso tienen los nodos internos?, siendo una respuesta inmediata, son nodos de referencia para hacer una busqueda eficaz sin pasar por otros nodos insulsamente.

Los Árboles B+, se han convertido en la técina mas utilizada para la organizacion de archivos indexados, y como ya se mencionó la principal caracteristica de este árbol es que toda la información se encuentra en las hojas, mientras que los nodos internos almacenan claves que se utilizan como índices y debido a esta característica en particular todos los caminos que van desde la raiz hasta el nodo hoja tienen la misma longitud.

Cabe mencionar que los Árboles b+ ocupan un poco mas de espacio que los árboles B y esto ocure por que existe la probabilidad de que alguna clave este duplicada.

Ahora bien, si se desea hacer una búsqueda de un valor "K" es necesario asegurarnos de que el arbol no esté vacío, luego empezar en el nodo mas significativo es decir la raíz y cuando estemos recorriendo cada espacio del bloque es necesario determinar a cual nodo saltar en caso no se encuentre en el nodo actual donde se esta haciendo el recorrido y asi hacerlo hasta llegar a un nodo hoja.

Para esto se sabe que el Árbol B+ ordena las claves de manera eficaz, por lo tanto, se puede obtener todas las claves en un rango dado haciendo una búsqueda de la clave en extremo inferior de dicho rango para luego escanear las hojas en orden hasta llegar al extremo superior del rango.

En caso de querer insertar un dato en un arbol B+ la dificultad se presenta cuando se desea insertar una clave en una página y ésta se encuentra llena, en este caso, la página afectada se divide en dos distribuyendose las claves de manera equitativa la mitada a la izquierda y la otra mitada a la derecha y la clave media sube al pagina antecesora convirtiendose en padre de las claves distribuidas en izquierda y derecha.

Ahora veamos el caso de eliminación de un nodo en un árbol b+, cabe mencionar que es mas sencillo de lo que se imagina, esto se debe a que las claves(datos) que se debe eliminar siempre se encuentran en las hojas y para ello se debe detectar algunos casos especiales.

1. Si al momento de eliminar una clave de un bloque "m" queda mayor o igual al orden del arbol "d" entonces estamos en el mejor de los caso termina la

operacion sin alterar algun otro nodo.

2. Si al momento de eliminar una clave de un bloque "m" queda menor al orden del arbol "d" entonces se debe realizar una redistribucion de claves, esto se debe hacer tanto en el indice como en las hojas. Cuando se cambia la estructura del arbol, se eliminan las claves que quedaron en los nodos interiores luego de que se haya eliminado el dato de la hoja.

Pongámonos en el caso de que estemos buscando una clave k en un árbol b+, lo correcto sería empezar con la lectura en la raíz y navegar hasta una hoja dependiendo el valor que se esté buscando, como se esta leyendo toda la raiz es imperativo establecer un estado de bloqueo de lectura en la raiz, y esto replicarlo en los demas nodos interiores hasta llegar a la hoja, entonces si varias transacciones activas están usando el arbol entonces grandes porciones del arbol se bloquean y esto puede bloquear varias transacciones de actualizacion,ahora bien esto es un problema común denominado el cuello de botella

Este cuello de botella de bloqueo se puede evitar aprovechando el hecho de que todas las transacciones atraviesan el árbol B de raız a hoja. Considere un árbol simple que consta de una página P (el padre), un hijo C de P y un hijo G de C (G es el nieto de P). En lugar de mantener bloqueos de lectura en todas las páginas que toca, en realidad es seguro que una transacción T_i libere su bloqueo de lectura en P después de haber establecido un bloqueo de lectura C, donde C cubre el rango de claves de interés.

El punto importante es que T_i adquirió su bloqueo en C antes de soltar su bloqueo en P. Desciende a través del árbol como si bajara una escalera, colocando un pie firmemente en el siguiente peldaño inferior antes de levantar el otro pie del peldaño superior. Esto se llama acoplamiento de bloqueo, o cangrejo (refernte a la forma en que camina un cangrejo). El efecto es que ninguna transacción que obtenga bloqueos en conflicto puede pasar a T_i en el camino hacia abajo, porque T_i siempre mantiene un candado en alguna página en el camino hacia su destino final.

La solución mas óptima es buscar en el árbol utilizando solo bloqueos de lectura, realizando un seguimiento de las páginas que están llenas. Si la hoja deseada resulta estar llena, suelte su bloqueo y comience a bajar desde la raíz nuevamente. Esta vez, el procedimiento de inserción mantiene bloqueos de escritura en todas las páginas que deben actualizarse, que incluyen la hoja L y su padre P, más el padre de P si P está lleno, más el abuelo de P si el padre de P está lleno, y así sucesivamente.

Después de este pequeño bombardeo de información sobre los arboles B y B+ ¿se puede mejorar aún mas?, bueno si resumimos todo lo anterior sabemos que si nosotros accedemos a un dato en especifico y este dato se encuentra en las hojas, que pasa si una vez encontrado dicho dato se nos da por ¿buscar otro

dato? y que dicho dato no este en la hoja en la que te encuentras, segun la estructura del árbol b+ deberíamos iniciar otra vez en el nodo raiz y recorrerlo hasta la hoja y esto no es óptimo es muy costozo, para esto nace el concepto de un árbol B-link que es basicamente tener un puntero en una hoja hacia el nodo vecino(derecho) que es otra hoja y evitarnos la lectura de la raiz hacia una hoja, con esta mejora evitamos lecturas insulsas.

3 Conclusión

Sin duda alguna los árboles B y sus diferentes mejoras han tomado protagonismo en el ámbito de las bases de datos, logran hacer lecturas mínimas al disco y a la vez son capaces de trabajar con grandes cantidades de información haciendo uso de índices.