INDEXACIÓN

Cada tipo de base de datos tiene su propio formato para almacenar datos. Están ajustados y optimizados para casos de uso específicos. En el siguiente ejemplo, tenemos una base de datos con información sobre automóviles:

ID	Brand	Model	Color	Price
1	Ford	Focus	Grey	42000
2	Toyota	Prius	White	40500
3	BMW	M5	Red	60000
4	Audi	A3	Black	38000
5	Toyota	Camry	White	51500
6	VW	Golf	Grey	32000

Almacenamiento interno

Cada base de datos se almacena internamente en un archivo con una cierta codificación y formato aplicados. Imaginemos que una base de datos está representada por un archivo CSV.

```
ID,Brand,Model,Color,Price
1,Ford,Focus,Grey,42000
2,Toyota,Prius,White,40500
3,BMW,M5,Red,60000
4,Audi,A3,Black,38000
5,Toyota,Camry,White,51500
6,VW,Golf,Grey,32000
```

- Hacer una búsqueda con solo seis entradas no es un problema.
- Si este tuviera 100.000 entradas su búsqueda seria diferente.
- El tiempo de consulta aumenta proporcionalmente al tamaño del archivo.
- Como sabemos que la base de datos crecerá con el tiempo, necesitamos encontrar una solución.

Índice de base de datos

Un índice de base de datos es una estructura de datos que se utiliza para mejorar los tiempos de búsqueda de datos.

Si necesitamos recuperar un automóvil de nuestra tabla por ID=6, sería mucho más rápido saltar a la fila respectiva de inmediato sin recorrer el resto. Esta es la idea principal de la indexación. También necesitamos guardar el desplazamiento que apunta a la entrada respectiva.

La forma más sencilla de lograrlo es mantener el desplazamiento de cada entrada en un hash. La clave es el valor de la columna que queremos indexar (en este ejemplo, es la columna de ID).
El valor hash es el desplazamiento en el archivo de base de datos. Para ID = 1, el desplazamiento es 0. Para ID = 2, el desplazamiento es 24.

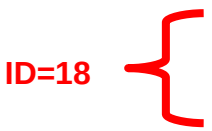
```
Hash Index {1:0, 2:24, 3:51, 4:70, 5:92, 6:118}
1 => {0}        --------->   1,Ford,Focus,Grey,42000
2 => {24}       --------->   2,Toyota,Prius,White,40500
3 => {51}       --------->   3,BMW,M5,Red,60000
4 => {70}       --------->   4,Audi,A3,Black,38000
5 => {92}       --------->   5,Toyota,Camry,White,51500
6 => {118}      --------->   6,VW,Golf,Grey,32000
```

La búsqueda se hace sobre el índice hash y toma el desplazamiento del ID respectivo. Luego comienza a leer el archivo de la base de datos desde el desplazamiento exactamente en la entrada necesaria.

Tabla no indexada

ID	UNIDAD	COSTO_UNITARIO
10	12	1.15
12	12	1.05
14	18	1.31
18	18	1.34
11	24	1.15
dieciséis	12	1.31
10	12	1.15
12	24	1.3
18	6	1.34
18	12	1.35
14	12	1,95
21	18	1.36
12	12	1.05
20	6	1.31
18	18	1.34
11	24	1.15
14	24	1.05

Tabla indexada por el campo ID



ID	UNIDAD	COSTO_UNITARIO
10	12	1.15
10	12	1.15
11	24	1.15
11	24	1.15
12	12	1.05
12	24	1.3
12	12	1.05
14	18	1.31
14	12	1,95
14	24	1.05
18	18	1.34
18	6	1.34
18	12	1.35
18	18	1.34
20	6	1.31
21	18	1.36
dieciséis	12	1.31

Si realizamos una búsqueda por ID = 18, las filas que tengan el ID=18 aparecen en filas contiguas. La búsqueda termina cuando el ID sea diferente de 18.

Tipos de estructuras que utiliza la indexación

Índice del árbol B

El B-Tree o índice de árbol equilibrado es una de las estructuras de indexación más utilizadas. Es una estructura de datos basada en árbol que mantiene un orden de datos, lo que permite operaciones eficientes de búsqueda, inserción y eliminación. Los índices B-Tree son especialmente adecuados para manejar grandes cantidades de datos y son el tipo de índice predeterminado para muchas bases de datos relacionales como Postgresql , MySQL y Oracle. Las ventajas de los índices B-Tree incluyen:

- Compatible con la mayoría de los sistemas de gestión de bases de datos .
- Puede manejar diversas operaciones de consulta, como coincidencias exactas, consultas de rango y clasificación.
- Fácilmente adaptable a diferentes tipos y tamaños de datos.

Índice hash

Un índice Hash utiliza una función hash para asignar los datos indexados a ubicaciones específicas en la estructura del índice. Este tipo de índice se utiliza principalmente para consultas de coincidencia exacta, donde la base de datos busca registros con un valor específico en la columna indexada. Los índices hash se adaptan a escenarios en los que los datos se distribuyen uniformemente y las consultas implican búsquedas exactas de valores-clave. Ventajas de los índices Hash:

- Rendimiento rápido de consultas para consultas de coincidencia exacta.
- Puede manejar datos de alta cardinalidad.
- Gastos generales de mantenimiento bajos.

Limitaciones de los índices Hash:

- No apto para consultas de rango u operaciones de clasificación.
- Sensibilidad a la selección de funciones hash y distribución de datos.

Cuándo usar indexación

Los índices están diseñados para aumentar el rendimiento de la base de datos; por lo tanto, la indexación se puede usar siempre que necesitemos mejorar significativamente el rendimiento de la base de datos. Cuanto más se expanda su base de datos, más probable es que la indexación lo beneficie.

Sin embargo, lo primero y más importante que debe recordar es que el índice ocupa espacio adicional; por lo tanto, cuanto mayor sea la tabla, mayor será el índice. Cada vez que realice una operación de agregar, eliminar o actualizar, también deberá ejecutar la misma operación en el índice.

Cuándo no usar indexacion

Cuando se escriben datos en la base de datos, primero se actualiza la tabla original, seguida de otros índices basados en esa tabla. Cuando se realiza una escritura en la base de datos, los índices se vuelven inoperables hasta que se actualizan.

Los índices nunca serán funcionales si la base de datos recibe escrituras continuamente.

Esta es la razón por la que los índices a menudo se aplican a bases de datos en almacenes de datos que obtienen nuevos datos de forma planificada (durante las horas de menor actividad) en lugar de bases de datos de producción que pueden recibir nuevas escrituras todo el tiempo.

Conclusiones

- La indexación de bases de datos ayuda a mejorar los tiempos de búsqueda
- La indexación incluye una estructura de datos con columnas para los criterios de búsqueda, así como un puntero.
- El puntero es la dirección en el disco de memoria de la fila que contiene la información restante.
- Orden de los tipos de estructuras de datos utilizadas en la indexación para mejorar el rendimiento de las consultas son árbol B, árbol R, tabla hash o mapa de bits.