Tema 4.4

Diseño físico

Los accesos a la base de datos se hacen: **Consulta SQL** del usuario, **consulta** del **esquema físico** por el SGBG, una vez sabe donde están las cosas (sabe dónde están los ficheros) se los **pide al SO** para que se los traiga en un buffer. Una vez los tiene el **SGBD puede realizar algunas operaciones** sobre los datos como filtrar filas o juntar tablas. Por último, se le **proporciona el resultado al usuario** a través de la aplicación de la base de datos.

ESTRUCTURA FÍSICA DE LA BASE DE DATOS

Los datos se almacenan en memoria secundaria como estructuras de datos "Ficheros", los cuales están compuestos por bloques. Formato muy útil para manejar información para los sistemas operativos.

Ficheros

Si tuviéramos una base de datos de libros, todas las filas de los libros estarían en un fichero con todos su atributos.

Operaciones con ficheros: El cómo se hagan es crucial en la manipulación de los datos. Hay de 2 tipos:

- **Operaciones de recuperación**: Leer el fichero buscando cosas según cierto criterio.
  - **Buscar registro**: Localizar el primer registro que cumpla cierta condición "apellido = nsq".
  - **Leer**: Obtener datos del registro identificado
  - **Buscar siguiente**: Tras encontrar un registro que cumpla la condición, pues otro.
  - **Buscar todos**: Buscas el siguiente hasta llegar al finar, todos los que cumplan la condición.
  - **Buscar ordenados**: Combina la búsqueda con el ordenamiento. Devolver todos pero ordenados

Bloques

Son la unidad mínima de almacenamiento que se puede leer o escribir desde o hacia un medio de almacenamiento físico. Tienen datos de cada fila con sus atributos y también punteros a otros registros.

Tamaño: El tamaño de los bloques depende del sistema operativo o del SGBD. Hay que ir cuidado xq si es mu pequeño puede tener mucha sobrecarga y si es muy grande puede desperdiciar mucha memoria o producirse fragmentación en los datos y que se tengan que hacer muchos accesos solo para leer una cosa.

Es **importante para el rendimiento de una BDD** esta organización xq **optimiza el acceso a datos** (no hace falta leer todo un fichero si puedes leer solo el bloque que quieres) **evitando** también contra la **fragmentación**, la **escritura** de datos se debe de hacer **optimizando el espacio en memoria** y la **transferencia de datos**, según el tamaño y como estén distribuidos los bloques, al **pasarlos a los buffers** cambia mucho si se hace más rápido o más lento.

ORGANIZACIONES DE FICHEROS

Hay varias formas y cada una tiene sus ventajas, desventajas y limitaciones y se ha de elegir según luego se vayan a usar los datos: Fichero Desordenado (Heap), Fichero Ordenado y Fichero Disperso (Hash).

Fichero Desordenado (Heap)

Los registros se insertan en el próximo espacio disponible, generalmente al final del fichero.

Ventaja: **Ideal** para **operaciones de inserción rápidas**, ya que no hay necesidad de reordenar los registros existentes

Desventaja: **Las búsquedas** pueden ser lentas si no hay un índice, ya que podemos tener que revisar todo el fichero para encontrar un registro específico

Fichero Ordenado

Mantiene los registros secuenciados según un campo específico, conocido como campo de ordenación.

Ventaja: Esto es especialmente útil para **consultas con condiciones de rango**, ya que los registros ya están en un orden que permite omitir datos irrelevantes rápidamente.

Búsquedas por nombres que empiecen por una letra so easy. Va desde el primero que cumpla hasta el último.

Fichero Disperso (Hash)

La **posición de cada registro** en el fichero depende de los valores de un campo específico y una función de hash.

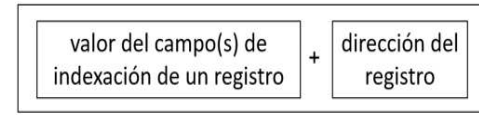
Ventaja: Permite una recuperación muy rápida para consultas de igualdad porque se puede calcular la ubicación exacta del registro sin necesidad de recorrer otros registros. Y el SO va directamente al punto donde se encuentra.

ÍNDICES

Un índice en una base de datos es una **estructura de datos** que **mejora la velocidad de las operaciones de recuperación de datos** a **cambio de operaciones** de escritura adicionales y el uso de más espacio de almacenamiento

actúan como una especie de "tabla de contenidos" para la base de datos, permitiendo que el SGBD localice rápida y eficientemente los datos sin tener que buscar a través de cada registro de forma secuencial.

Partes: Tiene claves de acceso generadas a partir de uno o varios campos de la tabla (columnas de índice). Se organizan para que el SGBD encuentre la dirección física de esos registros de manera eficiente.



Estructuras de Índices en Árbol B+

Son **estructuras en forma de Árbol equilibrado** que permiten **búsquedas**, **inserciones** y **eliminaciones** **eficientes**. Todas sus hojas están a la misma profundidad con cada nodo teniendo un número de claves (*con lo que puedes acceder a valor que toca*) dentro de un rango. *Útiles para recuperaciones por rango y devolver varios registros*.

Características

- **Equilibrio**: Al estar equilibrado todos los caminos son igual de largos y se pueden estimar los costes.
- **División/Fusión de nodos**: Al insertar/eliminar claves los nodos se dividen o se fusiona para equilibrarse.
- **Datos es en las Hojas**: Los datos (*punteros a los datos*) SOLO están en los nodos hojas.
- **Encadenamiento**: Las hojas están enlazadas, un apunta a la siguiente: fácil recorrido secuencial.

Creación: Se inserta un dato, este va hasta el nodo hoja que le correspondería siguiendo los índices. Cuando llega se añade un puntero en el nodo hoja que apunta al nuevo dato.
Si no hay espacio en el nodo hoja, este se parte en dos y crea un nodo padre. Los datos se dividen: Cada nodo hoja tiene la mitad de los datos ordenados por rango (*uno los altos y el otros los bajos*) y el padre se queda con el puntero del dato que está justo en la mitad.
Cada vez que se parte un nodo se actualizan los punteros a los nodos para que esté todo ordenado y los nodos hoja apunten al siguiente para asegurar el encadenamiento.

TOMA DE DECISIONES

Principal objetivo es asegurar que la base de datos **cumpla con los requisitos de rendimiento especificados**, lo que incluye velocidades de consulta rápidas y tiempos de actualización aceptables, incluso bajo carga pesada.

Técnicas de Rendimiento y Características del SGBD: Hay que **saber como afectan las configuraciones, técnicas de rendimiento** y otras cosas al rendimiento del sistema. También saber **que nos ofrece el SGBD elegido** (xq pot cambiar de uno a otro).

Entradas de Información para el Diseño Físico: Tienen **problemas sobre todo a la hora de comprobar restricciones**. Por ejemplo, las de unicidad (como las CP) pueden hacer que vaya mu lento xq tiene que comprobarlas todas, pues el uso de índices va muy bien. También para claves ajenas cosas que mejores los JOINs van bien.

Modelo de Procesamiento del Sistema: Es el como se Acceden, borran, modifican y leen los datos:

- **Inserción**: Conocer el volumen y la frecuencia de las inserciones: *Decidir, si un fichero desordenado sería apropiado por su rapidez en la adición de nuevos registros…*
- **Modificación**: Analizar qué columnas son modificadas regularmente: *Particionar una tabla para minimizar las actualizaciones costosas en bloques grandes de datos.*
- **Borrado**: Afecta a cómo se **recuperarán los datos** y a los **índices**: Pot requerir la reorganización de índices.
- **Recuperación**: Saber qué columnas se utilizan más para búsquedas para poder beneficiarse los índices.

IMPLEMENTACIÓN DE BASES DE DATOS RELACIONALES

Durante la implementación, se toman decisiones concretas sobre cómo se almacenarán los datos.

Consideraciones diseño físico: Se detallan las estructuras de almacenamiento que representarán lógicamente las tablas, relaciones y otros objetos de la base de datos: *Almacenamiento a nivel de ficheros y cómo estos se mapean a estructuras lógicas como tablas, vistas, índices, etc.*

Por ejemplo, cuando hacemos una tabla, como esta se debe almacenar: Si en un ficehro desordenado para insertar rápido o en uno ordenado para buscar más fácilmente.

Ajuste de la Base de Datos (*Tuning*): **Después de implementar la base de datos** según el diseño físico se debe de ajustar: Monitorizar el rendimiento al usarlo, realizar cambios, eliminar cuellos de botella y +↑ la eficiencia.

Consideraciones de los tipos de ficheros en implementación física

- **Fichero Desordenado (Heap)**: Es la estructura simple donde los registros se añaden en el orden en que llegan, lo que permite inserciones rápidas. ***Implementación típica de tablas de datos***.
- **Fichero organizado como índice**: Los registros se almacenan de manera que cada fichero es esencialmente un índice, proporcionando ***accesos rápidos basados en múltiples campos***.
- **Fichero Disperso (Con Hashing)**: Los registros se almacenan en bloques basados en una función hash aplicada a un campo específico (*campo de hash*), ***lo cual acelera las búsquedas por ese campo***.