Archivos - Indices

Bases de Datos

Tecnicatura Universitaria en Procesamiento y Explotación de Datos

2023

Archivo Secuencial Desordenado

- Es la forma básica más simple.
- Los nuevos registros se ingresan al final del archivo.
- Ingresar un nuevo registro es eficiente: se lee el último bloque del archivo en la memoria buffer se agrega el registro y se vuelve a grabar el bloque en el dispositivo.
- La dirección del último bloque se guarda en la cabecera del archivo.
- La búsqueda de un nuevo registro implica la búsqueda lineal leyendo bloque por bloque. En promedio se leen b/2 bloques.

b= cantidad de bloques del archivo

Archivo Secuencial Desordenado

- Eliminar un registro implica leer el bloque que tiene el registro en memoria, eliminar (o marcar) el registro y regrabar el bloque en disco.
- El archivo puede requerir una reorganización periódica para recuperar espacio y que los bloques ocupen su máxima capacidad.
- Los registros que se empleen en estos archivos pueden ser de longitud fija o variable, con división de un registro en diferentes bloques o no.
- Para leer los registros en orden según un campo se debe crear una copia clasificada de estos registros. La clasificación de un archivo voluminoso puede ser costosa. Se emplea clasificación externa.
- Para un archivo con registros de longitud fija, sin división de un registro en diferentes bloques, y que emplea almacenamiento contiguo, se puede acceder fácilmente al registro n-ésimo por medio de:

Bloque: $\lfloor n/b_{fr} \rfloor$ *registro en el bloque* = n mod b_{fr}

Archivo Secuencial Ordenado

- Los registros se almacenan físicamente ordenados en el dispositivo de acuerdo con un "campo" de ordenamiento. Si el campo de ordenamiento es un campo clave ➔ clave de ordenamiento.
- Leer los registros en orden es eficiente ya que no se requiere ordenar el archivo. Leer el próximo registro en orden no requiere de accesos adicionales de bloques.
- La lectura al azar de un registro en un archivo ordenado por clave es más rápida ya que se puede emplear la búsqueda binaria (si las direcciones de los bloques se encuentran en la cabecera del archivo). Esta búsqueda asegura encontrar el registro (o no) con un esfuerzo de $log_2 b$
- Las búsquedas que involucran criterios tales como $<, \leq, >, \geq$ es eficiente porque se conoce el orden de los registros

Archivo Secuencial Ordenado

- Los inserciones y eliminaciones de registros son operaciones costosas porque los registros deben permanecer en orden.
- Inserción significa encontrar el bloque, leerlo, grabar el nuevo registro, grabar el bloque, por lo general se deben regrabar en el disco en promedio la mitad de los bloques del archivo. Se puede dejar un espacio para preverlo, pero una vez que se llena el bloque…
- Un método que se emplea frecuentemente es: crear un archivo de transacciones y contar con un archivo maestro ordenado. Periódicamente el archivo de transacciones se ordena y es mezclado con el archivo maestro para generar un nuevo maestro ordenado.
- Los archivos ordenados rara vez se emplean en aplicaciones de bases de datos, a no ser que se le especifique un índice primario.

Archivos hash (acceso directo)

- Provee un acceso muy rápido a los registros bajo ciertas condiciones.
- Archivos "hash" o de acceso directo.
- Se emplea un campo simple (K) del registro denominado campo hash, si es clave se denomina clave hash.
- Se aplica una función *h* sobre el campo hash que da como resultado la dirección del bloque en disco donde se va a almacenar el registro.
- La búsqueda de un registro particular se lleva a cabo en la memoria principal una vez que el bloque se ha instalado en ella.
- La función que se emplea con mayor frecuencia es la del resto de la división: $h(K)=k \bmod M$ (arreglo de 0 a M-1)

Archivos hash (acceso directo)

- Si el campo has no es numérico se lo debe transformar. Hay otras funciones hash que aplican otras técnicas.
- *Colisión:* ocurre cuando el valor de un registro con clave hash diferente da la misma dirección que otro/s registros y debe ser ubicado en un lugar diferente.
- Hallar otro lugar para el nuevo registro cuando se produce una colisión se llama "resolución de colisiones":
  - Direccionamiento abierto: A partir de la dirección que da la transformación de la clave hash, el programa busca en las direcciones subsecuentes hasta que encuentra la primera posición vacía.
  - Encadenamiento: En este caso se tienen varias direcciones (bloques) aparte donde se guardarán las colisiones. En la primera posición vacía de los bloques se guarda la colisión, y se asigna un puntero hacia esta posición en el bloque que debería haber ocupado. Se mantiene una lista vinculada de colisiones.
  - Hash múltiple: se aplica una segunda conversión clave-dirección, si da una nueva colisión se palica una tercera ó alguno de los métodos para colisiones descriptos anteriormente.

Archivos hash (acceso directo)

- En los archivos se habla de "bucket" (cubo) , que lo constituye un bloque o varios consecutivos. Cada cubo constituye una dirección posible del archivo hash. Puede ser una dirección absoluta (bloque) o relativa (número de bloque).
- Cada "bucket" puede contener más de un registro, es decir que cada dirección de mi archivos pueda contener un conjunto de registros.
- Un cubo que contenga más de un registro maneja mejor las colisiones.
- Este archivo es apto para el acceso de registros al azar, no así para el acceso de registros en orden de acuerdo con la clave.
- Existen funciones hash que "preservan el orden" de las claves
- Otra parámetro es la "densidad de empaquetamiento":

  número de registros a almacenar / capacidad total del archivo

  Valores óptimos de esta relación oscilan alrededor de 0,8

Hash dinámico

- Direccionamiento calculado extensible
- Direccionamiento lineal

Archivos indexados

Archivo con una organización primaria secuencial desordenada, ordenada ó hash.

Índice : estructura de acceso auxiliar para aumentar la velocidad de acceso a los registros en respuesta a ciertas condiciones de búsqueda

Los índices por lo general presentan caminos de acceso alternativos

Básicamente cualquier campo del registro que compone el archivo puede emplearse para construir un índice
➔ Un archivo puede tener varios índice definidos sobre él

Para acceder a los datos asociados con algún valor definido en el índice Primero se accede al índice que apunta al bloque que contiene los datos Asociados con esa clave

Indices de un solo nivel

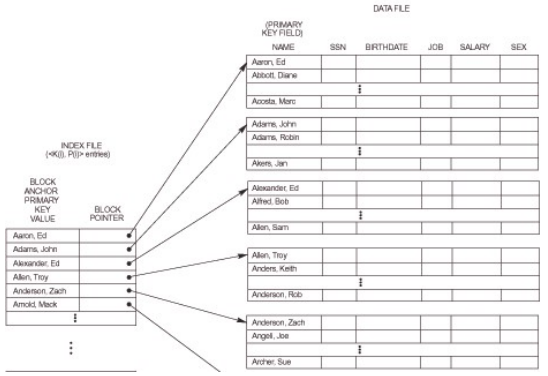
- **La estructura del índice se define sobre un campo del archivo (campo de indexación)**
- **El índice almacena todos los valores del campo de indexación + un puntero al bloque que contiene el registro asociado con ese valor de campo**
- **Los valores en el índice están ordenados para que se puedan realizar búsquedas binarias (basado en el hecho que el archivo de índice es más pequeño que el que contiene los registros de datos)**

Tipos
- **Primario: basado en el campo clave de ordenación física del archivo**
- **Agrupación: basado en el campo de ordenación que NO es clave**
- **Secundario: basado sobre cualquier campo pero que NO es por el cual se ordenó el archivo**

Indices de un solo nivel: Indices primarios

- **Una entrada en el índice por cada bloque del archivo primario: debido a que el índice se construye con la clave de ordenación**
- **Cada registro de índice contiene la clave primaria del primer registro del bloque y un puntero a ese bloque**
- **El índice tendrá tantas entradas (registros de índice) como bloques tenga el archivo**
- **Esto constituye un índice disperso (no denso) porque tiene entradas en el índice sólo para algunos valores de búsqueda y no para todos**
- **Un índice denso por el contrario se da cuando se construye un índice por el campo clave pero que los registros no estan ordenados➔ por lo tanto requerirá una entrada en el índice por cada valor de clave**
- **Este archivo presenta problemas en el momento de la inserción, modificación y eliminación de registros**

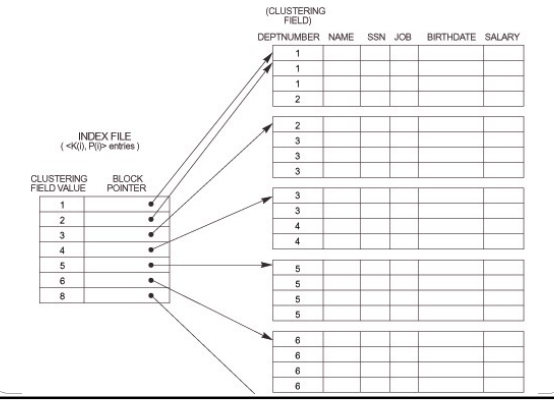
Indice Primario



Indices de un solo nivel: Indices de agrupación

- **Registros del archivo ordenados por algún campo NO clave (no tiene un valor distinto para cada registro).**
- **El registro de índice posee dos campos, uno para el valor del campo clave y el otro un puntero al bloque**
- **Una entrada por cada bloque que apunta al primero bloque del área de datos que tiene ese valor.**
- **También presenta problemas en las inserciones y eliminaciones de registros.**
- **Es un índice no-denso**

Indice Agrupación



Indices de un solo nivel: Indices secundarios

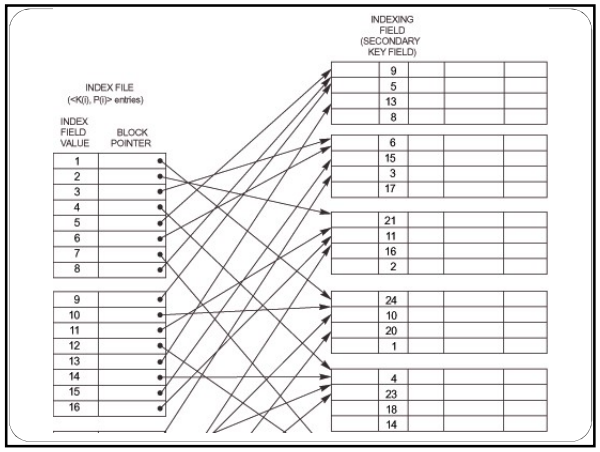
- **Se construye con un campo que NO es la clave de ordenación del archivo.**
- **El campo que se emplea para indexar puede tener un valor distinto (clave secundaria) ➔ Una entrada en el índice por cada valor diferente de clave➔ Indice denso**
- **Se aplica la búsqueda binaria para acelerar la búsqueda**

Indices de un solo nivel: Indices secundarios

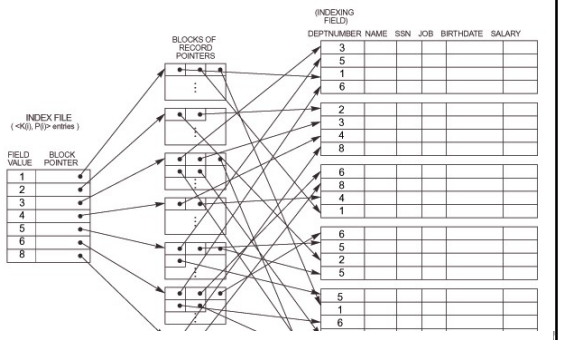
- **Si el campo es NO clave (no tiene un valor distinto para c/registro):**
  1. **Varias entradas en el índice para el mismo valor de clave (denso)**
  2. **Emplear registros de longitud variable con un campo repetitivo para el puntero solamente (no para la clave)**
  3. **Entradas de índice de longitud fija y crear un nivel más entre el índice y el área de datos para manejar los punteros múltiples. Indice no-denso. Se puede emplear un bloque ó una lista enlazada de bloques (esto depende del tamaño).**

  La búsqueda binaria para 1 y 2 se debe modificar.

  En el caso de 3, se requieren mas accesos por el nivel extra de indirección, pero los algoritmos son más simples



Indice secundario
Campo no-clave



Indices multinivel

- Dado que los registros del índice del primer nivel se encuentran en bloques y están ordenados, se puede crear para estos registros un nuevo índice primario para este. A este nuevo nivel lo denominamos segundo nivel de índice.
- El factor de bloqueo (fo=fbli) del 2do nivel es igual al del 1er. nivel, por lo tanto todas las entradas de índice tienen el mismo tamaño → valor del campo y dirección de bloque:
- Cantidad de bloques 1er. Nivel: $\lceil r1/fo \rceil = b1$;
  - r1 = valores de indexación diferentes
- Cantidad de bloques 2do. Nivel: $\lceil b1/fo \rceil = b2$;
- Cantidad de bloques del nivel n: $\lceil$ (nro. de bloques del nivel n-1)/fo $\rceil$
- Cada nivel reduce el número de entradas en un factor de fo
- Por lo general se busca que el nivel + externo del índice contenga un sólo bloque

Indices multinivel

- Un índice multinivel que tenga r1 entradas en el primer nivel, tendrá t niveles de acuerdo con: $t = \lceil \log_{fo}(r1) \rceil$
- Acceder a los datos asociados con un valor de indexación tendrá tantos accesos como niveles tenga el índice + 1.
- El esquema descripto se puede usar para cualquier tipo de índice: primario, secundario o de agrupación, siempre que el primer nivel tenga valores distintos y tamaño fijo.
- Un índice primario multinivel se denomina archivo secuencial-indexado.
- Se emplea un algortimo de búsqueda que es similar ya sea que el índice sea denso como no denso.