

# FICHEROS Y BASES DE DATOS (E44)

## 3º INGENIERÍA EN INFORMÁTICA

### Tema 3.

#### Estructuras de Almacenamiento

#### Básicas. Definición y Manejo.

- 1.- Búsqueda de Información. Definición de Clave.
- 2.- Definición y Manejo de Índices.
- 3.- índices Multinivel. Árboles B y B+.

(Capítulos 6, 7, 9 y 10 del Folk)

(Capítulo 3 del Date)

(Capítulo 5 del Elmasri)

# BÚSQUEDA DE INFORMACIÓN

## Definición de Clave

- En el tema anterior, se ha supuesto que el usuario conocía de antemano la posición del registro a acceder dentro del fichero.
- Esta circunstancia no suele ser habitual, sino que normalmente se desea acceder a los registros que contienen cierta información en un campo.
- Este método de búsqueda de información se denomina Acceso por Clave.
- Los valores en este campo deben de cumplir una serie de normas de almacenamiento que aseguren su localización, tales como,
  - Las cadenas de caracteres deben de estar en mayúsculas, y sólo debe de aparecer un espacio en blanco entre palabras.
  - No deben aparecer espacios en blanco ni en los campos numéricos ni en los de fecha.
- En estos casos se dice que la clave aparece en Forma Canónica.
- Normalmente se utiliza el valor de una clave para identificar cada uno de los registros del fichero, en cuyo caso se denomina Clave Primaria del Fichero.
- Las operaciones de inserción deben de incluir controles que aseguren que no exista duplicidad en los valores de la clave primaria.
- El resto de clave son Claves Secundarias.

# BÚSQUEDA DE INFORMACIÓN

## Búsqueda Secuencial y Búsqueda Binaria

- El método de acceso más sencillo es la búsqueda secuencial, pero su coste es  $(n)$ .
- Un método alternativo es la Búsqueda Binaria, cuyo coste es  $(\log n)$ .
- Este método requiere que los registros sean de tamaño fijo y que estén ordenados por el valor del campo clave asociado al proceso de búsqueda.

## Ordenación de un Fichero

- Los métodos clásicos para la ordenación de un conjunto de valores en memoria RAM, no suelen ser aplicables para la ordenación de un fichero.
- Estos métodos requieren un alto número de comparación y de desplazamiento de valores, que en el caso de un fichero requerirían gran cantidad de desplazamientos del cabezal.
- Por esta razón, es necesario definir alternativas que no presenten estas dificultades.
- La idea principal de todas ellas va a ser leer completa, o parcialmente, todos los registros, y posteriormente realizar la ordenación en memoria RAM.
- De este modo el coste de la lectura de los datos es mucho menor, ya que se reduce el número de desplazamientos al mínimo.

# BÚSQUEDA DE INFORMACIÓN

## Ordenación en Memoria

- La primera idea es leer todos los registros en un vector de cadena de caracteres, que se puede ordenar en memoria.
- Pero el orden de este vector de cadenas puede ser diferente al definido por la clave.
- Para evitar este problema, se crea un segundo vector de caracteres, el Índice, en el que aparece la clave de los registros y un puntero a la posición del registro en el anterior vector.
- La ordenación de este nuevo vector, obtiene el orden buscado con un menor coste.
- Todavía es posible reducir el coste, si se define un tercer vector que contiene únicamente los punteros antes mencionados, evitando el movimiento de las claves dentro del vector.

## Ordenación por Clave

- Cuando el fichero es demasiado grande para almacenarse en memoria, se debe considerar almacenar únicamente el índice.
- En este caso el puntero es el número relativo de registro o el primer byte del registro.
- El método es similar al realizado en memoria.
- La escritura requeriría una segunda lectura del fichero, que además sería aleatoria.
- Esta lectura tiene un coste demasiado alto, lo que aconseja la utilización de otras técnicas.

# BÚSQUEDA DE INFORMACIÓN

## Conclusiones

- La utilización de la búsqueda binaria reduce el coste de acceso, pero todavía es alto.
- Si la clave de búsqueda no coincide con la clave de ordenación, se debe reordenar el fichero o bien utilizar la búsqueda secuencial.
- Además, el uso de la búsqueda binaria se restringe a registros de tamaño fijo.
- Mantener un fichero ordenado puede ser muy costoso cuando el número de actualizaciones es demasiado alto.
- Este hecho reduce de modo significativo la ventaja obtenida en el uso de la búsqueda binaria.
- La ordenación en memoria sólo es útil cuando el fichero es lo suficientemente pequeño como para poder almacenarse en memoria.
- Por su parte, la ordenación por clave requiere una lectura aleatoria adicional con un coste demasiado alto.
- Todos estos hechos aconsejan definir una técnica que,
  - Reduzca el coste de acceso, a uno o dos accesos a disco, para cualquier clave.
  - La búsqueda de información no requiera ordenar los registros del fichero.
  - Las operaciones de actualización se realicen de modo eficiente.

# DEFINICIÓN Y MANEJO DE ÍNDICES

## Concepto de Fichero Índice

- La lectura aleatoria de la ordenación por clave se puede eliminar si se almacena en memoria secundaria el índice ordenado.
- Éste es el origen de los Ficheros Índices, que se define como un fichero cuyos registros son de tamaño fijo, en el que se almacena la clave del registro y un puntero a este registro.
- En el fichero índice pueden aparecer otras informaciones como el tamaño del registro.
- Su utilización presenta una serie de ventajas,
  - Permite el uso de la búsqueda binaria tanto para ficheros de registros de tamaño fijo como para los de tamaño variable.
  - El proceso de ordenación se restringe al fichero índice, que en muchos casos puede realizarse en memoria.
  - Es posible definir diferentes ficheros índice, cada uno de los cuales se asocia a una clave de búsqueda.
- Pero también presenta inconvenientes,
  - Las operaciones de actualización requieren la actualización de más de un fichero.
  - El orden establecido en los ficheros índices debe de mantenerse.
  - Si el fichero índice es demasiado grande para almacenarse en memoria, estas operaciones pueden ser muy costosas.

# DEFINICIÓN Y MANEJO DE ÍNDICES

## Tipos de Índices

- Una primera clasificación es la siguiente,
  - Índice Denso, si contiene una referencia a todos los registros del fichero.
  - Índice no Denso, si sólo contiene referencia a un subconjunto de registros.
- Los segundos se suelen definir cuando el orden del fichero y del índice coincide.
- Otra posible clasificación es la siguiente,
  - Índice Primario, si contiene la clave primaria del fichero.
  - Índice Secundario, si contiene otra clave.
- En los primeros se debe asegurar que no exista ninguna clave repetida, por lo que el tamaño de su registro debe de ser suficiente.
- En los segundos, no existe esta condición, por lo que es posible que aparezcan claves que estén asociadas a más de un registro, lo que puede provocar infrautilización del espacio.
- Una opción es incluir más de un puntero en el registro de estos índices, aunque el problema es determinar este número, de modo que no se aumente la infrautilización.
- Otra opción es la definición de un índice sobre el índice secundario, que almacene el puntero al registro del fichero.
- Así, cada valor de clave tiene asociado un conjunto de punteros, que pueden aparecer como una lista enlazada.

# DEFINICIÓN Y MANEJO DE ÍNDICES

## Operaciones sobre Índices

- Alguna consulta de datos puede realizarse sin necesidad de acceder a los datos, mediante la comparación de los punteros de los índices.
- Entre éstas cabe comentar la existencia de valores y la combinación de condiciones.
- Una inserción requiere la inserción de una nueva entrada en todos los ficheros índices, lo que puede producir su reordenación.
- Una eliminación debe propagarse en todos los ficheros índice, que puede implementarse asignando un valor concreto en el puntero.
- Cuando se manejan índices secundarios, este proceso puede ser muy costoso por lo que se aconseja modificar la filosofía de estos índices,
  - El puntero no hace referencia al fichero original, sino que contiene la clave primaria.
  - Sólo se borra sobre el índice primario.
  - La búsqueda sobre el índice secundario de un registro borrado se detecta al acceder al índice primario.
- La actualización depende del campo objeto de la operación,
  - Si se modifica la clave primaria se debe reordenar el índice asociado, y los punteros de los índices secundarios.
  - La modificación sobre una clave secundaria, requiere la reordenación de su índice.

# DEFINICIÓN Y MANEJO DE ÍNDICES

## Control contra Fallos del Sistema

- El estado de un índice debe de reflejar en todo momento el contenido del fichero al que referencia.
- El funcionamiento de las operaciones sobre índices aseguran que éstos se actualizan de modo adecuado en memoria.
- Estas actualizaciones también se deben de reflejar en memoria secundaria, cuando el fichero se cierra.
- Si esta actualización no se produjera, por un fallo del sistema, el fichero índice no reflejaría el estado real del fichero y por tanto debería de reestablecerse.
- Para detectar esta posibilidad, en el registro de cabecera del fichero índice se introduce una bandera que indique esta situación.
  - La lectura del fichero debería marcar el fichero como leído y no escrito.
  - Por su parte, la escritura debería modificar el valor de esta bandera.
- La lectura debería controlar el estado de esta bandera, reconstruyendo el fichero índice si ésta indicara que ha sido leído y no reescrito.

# ÍNDICES MULTINIVEL. ÁRBOLES B Y B+

## Índices Multinivel

- Cuando un fichero índice es demasiado grande para almacenarse en memoria RAM, su gestión puede resultar muy costosa,
  - La búsqueda binaria requiere la realización de muchos desplazamientos.
  - Mantener la clasificación del índice puede requerir un movimiento de datos importante.
- Estos problemas también surgen si no es posible almacenar en memoria todos los ficheros índice de un fichero.
- La solución a estos problemas es la definición de un índice no denso sobre el fichero índice en cuestión.
- Este proceso puede repetirse hasta llegar a un nivel en el que todo el fichero índice quepa en memoria, y más concretamente, en un buffer, o página, de memoria.
- De este modo se define un Índice Multinivel, compuesto por un conjunto de índices no densos que actúan sobre un índice que puede ser denso o no denso.
- El coste de acceso puede reducirse de modo significativo, pero en cambio las inserciones y los borrados pueden ser muy costosas.
- Para reducir este coste, es necesario definir una estructura de ficheros que permita reducir el coste de todas las operaciones.

# ÍNDICES MULTINIVEL. ÁRBOLES B Y B+

## Árboles Binarios

- El árbol binario es la estructura de datos que mejor se acomoda al funcionamiento de la búsqueda binaria, y además tiene un coste de inserción y borrado moderado.
- Por esta razón, fue la elegida para la definición de los Índices Multinivel Dinámicos.
- La idea es construir un árbol binario con las claves del índice, que posteriormente debe de ser almacenado en memoria secundaria.
- Los árboles binarios se almacenan como árboles, donde cada hoja contiene un subarbol del árbol binario original.
- El número de hojas en cada subarbol suele ser constante, y se relaciona con la capacidad de una página de memoria.
- Mediante la utilización de esta estructura de ficheros, se consigue reducir el coste de la búsqueda binaria.
- Siendo N el número de hojas en un árbol perfectamente balanceado y completo, el número de desplazamientos máximo para encontrar un elemento es  $\log_2(N + 1)$ .
- Si en una página caben hasta k hojas del árbol binario, esta nueva alternativa permite reducir el número de desplazamientos hasta un máximo de  $\log_{k+1}(N + 1)$ .

# ÍNDICES MULTINIVEL. ÁRBOLES B Y B+

## Árboles Balanceados

- La cuestión se presenta en las propiedades de árbol binario, ya que resulta fundamental que la altura del árbol sea mínima.
- Por tanto resulta necesario obtener árboles que se encuentren Balanceados.
- Inicialmente se consideró una alternativa de construcción descendente que dió lugar a los árboles AVL.
- La idea es ir insertando las claves en el árbol binario, manteniendo los subárboles en cada página lo más equilibrados posible.
- Esta filosofía falla si la secuencia de claves no es adecuada, como por ejemplo que lleguen ordenadas.
- Un análisis del problema concluyó que la causa era la mala elección de la raíz del árbol.
- Para evitar estos problemas se optó por la construcción ascendente, los Árboles B.
- En estos árboles, las claves se insertan en una página, y cuando ésta se completa se elige la mejor raíz de las que aparezcan en la página.
- A diferencia de los árboles AVL, la raíz no es el primer elemento de secuencia, sino que se elige de entre un conjunto de claves.
- Además, el método de construcción del árbol asegura que el árbol mantendrá una altura mínima.

# ÍNDICES MULTINIVEL. ÁRBOLES B Y B+

## Propiedades de los Árboles B

- Las propiedades de los árboles B parten de dos conceptos básicos,
  - Se define el Orden de un Árbol B como el número máximo de hijos de una página.
  - Por su parte la Hoja de un Árbol B como el nivel más bajo de la estructura.
- Así, un árbol B de orden  $m$  cumple que,
  - Una página puede referenciar  $m$  páginas hijo como máximo.
  - Toda página, excepto la raíz y las hojas, tienen al menos  $m/2$  hijos.
  - La raíz tiene por lo menos dos páginas hijo.
  - Todas las hojas aparecen en el mismo nivel.
  - Una página, que no es hoja y que tiene  $k$  hijos, contiene  $k-1$  claves.
  - Una página hoja contiene al menos  $m/2 - 1$  claves y no más de  $m-1$  claves.
- Las operaciones de actualización deben de respetar estas reglas, por tanto,
  - Si una inserción debe insertar una clave en una página hoja completa, debe dividirla en dos páginas.
  - La eliminación que produzca una página con un número menor al mínimo definido, debe de modificar su estructura.

# ÍNDICES MULTINIVEL. ÁRBOLES B Y B+

## Operaciones sobre Árboles B

- La inserción en un árbol B se fundamenta en el proceso de División y Promoción.
  - Una nueva clave siempre se intenta insertar en una página hoja.
  - Si ésta se encuentra completa, se elige una clave que divida el nuevo conjunto de claves en dos subconjuntos equilibrados.
  - Los dos subconjuntos se almacenan en dos páginas, mientras que la clave elegida se inserta en el nivel superior, en donde se puede repetir el proceso.
- Por su parte, el borrado se fundamenta en la Eliminación, Redistribución y Concatenación.
  - Siempre se deben eliminar claves situadas en una página hoja.
  - Si se desea eliminar una clave que no esté en una página hoja, se intercambia con la clave siguiente, que está en una hoja, y luego se elimina.
  - Cuando una página hoja quede con un número de claves menor que el mínimo, hay que examinar sus páginas hermana,
    - Si una página hermana tiene más del número mínimo de claves, se deben Redistribuir sus claves con la página actual.
    - En caso contrario, se Concatena la página actual, una página hermana y la clave que las separa, repitiéndose el proceso.

# ÍNDICES MULTINIVEL. ÁRBOLES B Y B+

## Tipos de Árboles B

- Un Árbol B\* intenta mejorar el llenado de las páginas del árbol,
  - La redistribución se incluye en la inserción, tal que si es posible se distribuyen las claves entre páginas hermanas y si no se divide.
  - Así, la división involucra dos páginas llenas, dando lugar a tres nuevas páginas.
    - Una página tiene al menos  $(2m-1)/3$  hijos.
    - Una página hoja tiene al menos  $(2m-1)/3$  claves.
- Por su parte el Árbol B+ pretende el manejo de los datos de modo secuencial y directo,
  - Los datos aparecen en bloques enlazados, denominados Conjunto Secuencia.
  - La inserción utiliza la división de bloques, mientras que el borrado se fundamenta en la redistribución.
  - Sobre éste se construye un índice en forma de árbol B, denominado Conjunto Índice.
  - Las hojas del índice forman un índice no denso sobre los bloques, que incluyen la clave del último registro del bloque.
  - Su objeto es orientar la búsqueda, por lo que sería suficiente que los nodos del árbol sólo contuvieran un separador entre las claves de los registros de los bloques, formando los Árboles B+ con Prefijos Simples.

# ÍNDICES MULTINIVEL. ÁRBOLES B Y B+

## Tipos de Árboles B

