Capítulo 7. IAGP 2005/06. Organización de Ficheros y Métodos de Acceso

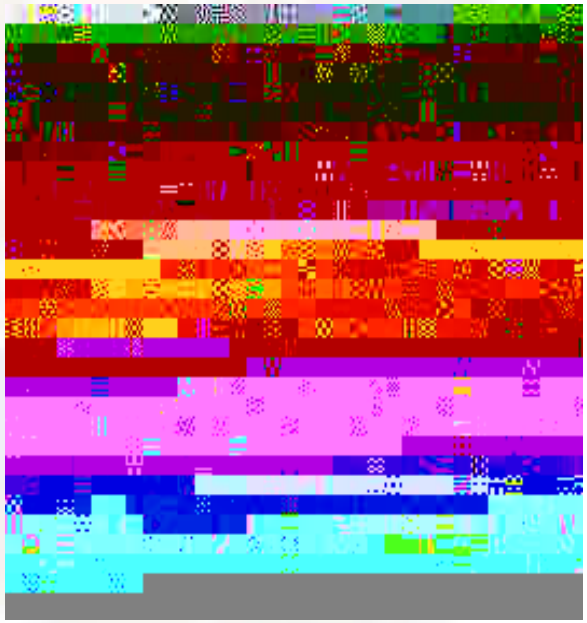
Actualizado 2006/06/17

Buscar en toda la web

1.- Soportes

Son los dispositivos que almacenan los datos, hay dos tipos:

- Soportes de *Acceso Directo* a los datos (Ej.: discos). Son los más empleados.
- Soportes de *Acceso Secuencial* (Ej.: cintas magnéticas). Se suelen usar en copias de seguridad.



Cilindro lógico: Está formado por todas las pistas que tengan el mismo número en todos los discos del disco duro. Tiene importancia cuando hablamos de la velocidad en la transferencia, la información que se quiere leer simultáneamente se almacena en el mismo cilindro lógico para evitar tener que desplazar a menudo el brazo de las cabezas lectoras.

2.- Procesamiento de Ficheros

Es la forma de solicitar la información al disco. Existen dos métodos para ello:

- Modo Secuencial: Se lee la información de un fichero de registro en registro teniendo que leer todos los que hay antes del que buscamos. Se emplea bien por deseo, bien por imposición del tipo de soporte que estamos usando. El acceso secuencial es recomendable cuando se quiere trabajar con muchos registros del fichero.
- Modo Directo: Se puede acceder a un registro si tener que leer todos los anteriores (basta con un pequeño número de lecturas). Hay dos maneras:
    - *Cálculo*: Cada región tiene una clave sobre la que se aplica un cálculo que indica el lugar de grabación (**Hashing**).
    - *índices*: existe un índice independiente o asociado al fichero en el cual se busca el registro y se nos indica donde está.

3.- Organización de Ficheros

Son los modos de disponer los registros del fichero en el soporte. Existen tres modos principales:

- Secuencial: Un registro a continuación de otro.
- Directo: Los registros binarios no se disponen en el soporte atendiendo a un algoritmo de cálculo.
- Indexado: Los registros generalmente se almacenan secuencialmente y van con un índice.

A continuación y durante el resto del presente tema, se profundizará sobre estos tres métodos de organización de ficheros.

4.- Ficheros Lineales (Modo Secuencial)

Los registros están dispuestos uno a continuación de otro. Existen dos formas de este tipo: *simple*

y encadenado.

Simple

La disposición de los ficheros (uno detrás de otro) se traduce en un almacenamiento sin huecos entre ellos.

Nota: un registro físico es el bloque fijo que se transfiere del disco a la memoria principal, y por tanto puede contener más de un registro lógico.

A continuación veremos las *principales características de los ficheros lineales simples*:

- A) Consulta muy rápida en procesamiento secuencial.

- B) Modificaciones del fichero:

    - Si el **soporte** es **secuencial** la modificación obliga a hacer una copia del fichero. Al realizar una inserción hay que desplazar hacia atrás todos los que siguen. Al efectuar un borrado hay que desplazar hacia delante todos los registros que seguían al registro borrado, y por último para modificar un registro también hay que hacer una copia ya que para modificarlo hay que leerlo entero, con lo cual, una vez leído, la cabeza ya ha pasado por él y habría que volverla hacia atrás (cosa que no podemos hacer).

    - Si el **soporte** es **directo** es posible hacer modificaciones sencillas, pero la inserción y el borrado requieren una copia del fichero. Para hacer dicha copia se emplea el *Algoritmo de la Línea de Balance* que consiste en tener un ***fichero de movimientos*** que almacena los registros que van a sufrir modificación. Este fichero y aquel del que proceden los datos

deben tener la misma clave, se procesa el primer registro de ambos y se graba en otro fichero la modificación (si procede) de ese registro, o bien si en el fichero de movimientos se indica el borrado no se copia.

- C) Proceso lento para consultas puntuales

- D) Aprovechan mucho el espacio de almacenamiento (sólo se precisa el justo para los datos)

- E) Posibilidad de usar cualquier tipo de soporte.

- F) Problema para procesar un fichero por más de una clave (campo de registro), ya que si un registro está ordenado en función de una clave no puede estarlo por otra. Las soluciones a este problema son: o bien se tienen dos ficheros iguales o más (tantos como clasificaciones diferentes haya) cada uno ordenado con respecto a una clave, o bien se clasifica el fichero cada vez que se

quiera acceder (lo cual es muy lento).

Encadenado

Los ficheros lineales encadenados mejoran a los simples. Los registros se procesan en el orden lógico (uno detrás de otro), pero este no tiene porque coincidir con el orden físico (los registros se enlazan por punteros). Es imprescindible un soporte de acceso directo.

Los registros deben contener un campo extra para almacenar el puntero (que puede dar la dirección exacta del siguiente registro o bien ser una dirección relativa respecto del comienzo del fichero). Se crea para evitar las copias implicadas en el proceso de inserción y borrado; estos procesos sólo conllevan un reajuste de punteros.

Los punteros son entre registros físicos, y recordemos que en un registro físico cabe más de un registro lógico.

Este tipo de organización se usa mucho con diferentes estructuras:

A) Listas Simples. Son de acceso o procesamiento secuencial y suelen ser pilas o colas. Son las más sencillas y responden a la descripción general que se ha hecho para los ficheros secuenciales encadenados.

B) Listas Múltiples.

- Son también de acceso secuencial, es decir, que para llegar a un registro lógico, hay que pasar previamente por todos los anteriores a él. En este tipo de listas cada registro lleva más de un puntero.
- Permiten tener clasificados los registros por más de una clave, teniendo varios campos de puntero.

- Suele haber un registro índice que es cabeza de todas las listas, o sea, es un registro de punteros que apuntan al principio de la lista correspondiente a la ordenación que deseemos.

- Como los registros no se almacenan secuencialmente, y sin embargo si se accede secuencialmente, este acceso es más lento porque la cabeza tiene que *ir dando saltos*.

- Regularmente se deben reorganizar los datos para acelerar el acceso a través de la clave más habitual.

C) Anillos. Se emplean como estructura de muchos de los modelos de bases de datos.

D) Árboles. Tienen dos funciones principales: la construcción de índices y de ficheros.

El tipo de árbol que se emplea generalmente es el binario, en su variante de **árbol binario de búsqueda**, se usa porque permite que se procesen los registros de forma directa y porque es sencillo hacer un recorrido secuencial en ellos, al procesar el árbol en in-orden.

Los árboles binarios no de búsqueda sirven para desarrollar cualquier tipo de estructura jerárquica siguiendo la técnica del enlace al sucesor - enlace al gemelo. En esta técnica el hijo izquierdo de cada nodo es un sucesor, y el hijo derecho un gemelo. Veamos como se aplicaría esta técnica al siguiente árbol:

En los árboles la consulta y la inserción son sencillas. Las supresiones se pueden hacer o bien por marca (no se precisa reorganizar), o bien supresiones reales que tienen la ventaja de que no dejan huecos en la estructura. Las organizaciones encadenadas se prestan bien a compartir espacio en el soporte con otras organizaciones encadenadas que haya. La estructura de árbol tiene muy pocas desventajas.

Sin embargo uno de los principales problemas de las demás estructuras encadenadas, es que si se hacen muchas supresiones, quedan excesivos huecos, con lo que el fichero se desaprovecha excesivamente. Para evitar esto existen dos técnicas: la recuperación de huecos y la gestión dinámica del espacio libre.

La ***recuperación de huecos*** consiste en lo siguiente: al crear el fichero se reserva espacio y se encadenan los huecos por medio de punteros (por tanto los registros deben ser de longitud fija). Siempre se tiene un puntero señalando a la primera posición libre del fichero (que es el hueco al que se acude a la hora de realizar una nueva inserción). Si fuese necesario hacer una supresión, el puntero de inserción pasaría a apuntar al registro borrado, y dicho registro apuntaría a donde estaba apuntando el puntero de inserción antes de realizar el borrado. Veamos un ejemplo:

En un principio tenemos el fichero distribuido de esta manera donde cada cuadro representa un registro (los sombreados son registros ocupados). El puntero marcado es el de inserción.

Si borramos uno de los registros ocupados, por ejemplo el de la segunda fila y la segunda columna, el fichero quedaría como sigue:

Como vemos el puntero de inserción apunta ahora al registro que acaba de ser borrado,

por tanto cuando hagamos la próxima inserción se realizará en dicho registro.

La ***gestión dinámica del espacio libre*** permite:

- Tener registros de longitud variable.
- Que los nuevos registros se inserten lo más cerca posible de los anteriores.
- Reorganizar los huecos para que queden juntos, esto quiere decir que al hacer supresiones habrá movimiento de registros a nivel físico.

Este método se podría representar gráficamente de la siguiente forma:

R1 R2 R3 R4 Hueco Hueco

Si borramos R2 quedaría:

R1 R3 R4 Hueco Hueco Hueco

5.- Ficheros con índices (Modo Indexado)

En este modo de organización, al fichero le acompaña un fichero de índice que tiene la función de permitir el acceso directo a los registros del fichero de datos.

El índice se puede organizar de diversas formas, las más típicas son: secuencial,

multinivel y árbol.

A través del índice podremos procesar un fichero de forma secuencial o de forma directa según la clave de indexación, y esto independientemente de como esté organizado el fichero por sí mismo.

El índice debe estar organizado en función de alguno de los campos de los registros de datos. Se pueden tener tantos índices como se quiera variando la clave (o campo) que se emplee. El índice está formado por registros (entradas) que contienen:

- Clave de organización.
- Puntero(s) al fichero de datos, en concreto al registro que corresponda.

Los índices se pueden clasificar en dos tipos, según cada entrada señale a la dirección de un registro del fichero de datos (índice total o denso), o bien apunte a un grupo de registros del fichero de datos que debe estar ordenado (índice escaso o no denso). En el caso de índices totales, el fichero puede estar desordenado.

Con el segundo tipo se podría procesar directamente el fichero de datos de forma secuencial.

Los índices totales o densos no suelen utilizarse de forma simple, sino combinados con índices escasos más cortos, de esta manera pueden almacenarse en memoria principal obteniendo así un acceso más rápido.

A continuación veremos las tres formas principales de organizar los índices:

Secuencial

En esta forma de organización se usan cadenas de punteros. Dentro de ella podemos distinguir dos tipos:

A. ***Simple*** Casi no se utiliza, en ella las inserciones y supresiones son realizadas por copia.

B. ***Encadenada:*** Se emplea más que la anterior, aún así tiene el problema de que si crece mucho el fichero de datos, crece también el número de entradas, provocando que la búsqueda, al ser secuencial no se óptima para un acceso directo.

Multinivel o Jerarquizada

Consiste en varios índices secuenciales encadenados. Tendremos un índice a los registros de datos y otros índices que apuntarán a un índice de nivel menor.

Este método surge para mejorar la organización secuencial encadenada. Los índices de

nivel alto suelen ser escasos y los de primer nivel densos.

Al igual que en la organización secuencial, si aumenta el número de registros sigue aumentando el número de entradas.

Árbol

Viene a mejorar el problema del crecimiento de entradas en un nivel. Se pretende que el número de entradas en cada nivel sea fijo, y lo que crezca sea el número de niveles.

Se usan diferentes tipos de árboles, binarios (de búsqueda y AVL), multirrrama y B^+

6.- Índices Secundarios

Hasta ahora se ha hablado de índices primarios (aquellos que emplean claves primarias, es decir una clave, un registro).

Puede ser que nos interese tener un índice para claves que no sean primarias, o sea una clave para más de un registro, éstos son los llamados índices secundarios. Su principal característica es que, al contrario que en los primarios donde el direccionamiento pudiera ser real (posición Exacta en el disco) o relativo (en función de la posición del fichero), en los secundarios se emplea el direccionamiento simbólico (la clave proporciona la clave primaria del registro, y no su dirección ni física, ni relativa, y el sistema emplea la clave primaria para localizar ese registro), en definitiva, emplea punteros indirectos.

La ventaja de este direccionamiento es que podemos hacer muchos índices secundarios y a la hora de modificar los ficheros, las direcciones físicas cambian, con lo que se deben cambiar también los índices primarios (actualizarlos); esta operación puede llevar mucho tiempo, sin embargo, al usar direccionamiento simbólico no es necesario modificar los

índices, puesto que no tienen punteros a ningún sitio.

Veamos, a continuación, algunas generalidades de los ficheros en los que se usa este tipo de direccionamiento:

- Reciben el nombre de ficheros invertidos.
- A partir de un dato (clave secundaria), se obtiene una clave primaria que lleva a más datos.
- Ejisten dos tipos de ficheros invertidos: Losficheros totalmente invertidos y los ficheros parcialmente invertidos
- En los **Ficheros Totalmente Invertidos** de una clave secundaria se obtienen todas las primarias relacionadas. Ej.:

Este tipo de ficheros cuentan con una ventaja añadida, que es la de poder responder a

ciertas preguntas sin tener que usar el fichero de datos (¿Número de alumnos en Almería?)

- Los **Ficheros Parcialmente Invertidos** se utilizan para evitar que el índice crezca mucho. Cuando buscamos por medio de una clave secundaria, no aparecen todas las claves primarias relacionadas a ella, sino que sólo aparece la primera clave primaria, y dentro del fichero de datos Ejisten punteros a los registros de igual clave secundaria. Estos ficheros también son llamados Ficheros Multilista. La técnica más usada es la de usar "*punteros empotrados*" en el fichero de datos. Los ficheros de este tipo son los más utilizados, empleando para implementar el índice un árbol tipo B.

Índices múltiples.

Son índices formados por más de un atributo (campo). Se suelen emplear estructuras de array (rejillas) n-dimensionales. En las celdas de las rejillas en las que hubiera concordancia habría un puntero simbólico al registro que correspondiera. Son más rápidos pero ocupan mucho más espacio.

B)**índices de árboles B** (*Generalmente B^+*).

1. índices Secuenciales Multinivel.

I. ISAM (*Métodos de Acceso Secuencial Indexado*).

Este método usa un fichero de datos secuencial y un índice secuencial.

Divide el espacio del soporte en tres zonas: *área de Datos, área de índices* y *área de Desborde*, las cuales se subdividen en otras según la estructura de los soportes. Los datos se organizan en pistas (que es la unidad de transferencia con la memoria principal) y éstas en cilindros lógicos.

Cilindros del área de Datos

La pista 0 de todos los cilindros se reserva para crear los llamados índices de pistas y alguna más para los excedentes del cilindro (al final).

Cuando se llene una pista se pasa a la siguiente pista libre de ese mismo cilindro

(se va rellenando cilindro a cilindro). Al rellenar una pista se crea en el índice de pista una entrada con la clave de mayor orden de esa pista y un puntero a esa pista.

Al llenar un cilindro, en el área de índices se crea una entrada en el índice de cilindros con la clave de mayor orden y un puntero al cilindro.

Puede existir un tercer índice, el índice maestro, muy pequeño que apunta al índice del cilindro.

La mejora que obtenemos con este método es que al poder llevar una pista entera a memoria principal se trabaja más rápido; si al hacer una inserción excedo el tamaño de la pista el/los registro/s excedente/s va/n a las pistas del área de excedentes del cilindro.

Tratamiento de los registros excedentes.

Pueden **almacenarse** en una zona (un cilindro o más) exclusiva para ellos. Otra forma sería reservar pistas para los registros excedentes al final de cada cilindro. Por último una tercera forma consiste en una mezcla de las dos anteriores, es decir tener pistas al final de los cilindros y una zona exclusiva.

Esta 3ª forma es la más utilizada, ya que la 1ª presenta el inconveniente de tener que hacer movimientos de las cabezas del disco para acceder a los excedentes, y la 2ª, aunque no tiene este problema tiene otros dos inconvenientes: que se puede agotar el espacio reservado o bien que por miedo a que esto ocurra se desaproveche mucho espacio en el soporte.

Para **localizarlos** según que técnica empleemos tardaremos mucho (búsqueda secuencial, índice de pistas para los Ejcedentes). La técnica más empleada consiste en que en el índice de pistas cada entrada sean en realidad dos

entradas, una para los registros almacenados normalmente, y otra para los Ejcedentarios. Por tanto cada entrada estará compuesta por una entrada N que será un puntero a la pista y como clave la mayor de la pista, y una entrada O que tiene un puntero a la menor entrada correspondiente a esa pista que esté en el área de Excedentes y como clave la mayor de dichos Excedentes. Veamos un esquema que nos aclare esto un poco:

***Nota**:* En caso de no haber excedentes la entrada O es igual que la N.

A continuación veremos un ejemplo de inserción que provocará la aparición de un registro Ejcedente: Supongamos que tenemos la distribución que se muestra en la siguiente figura, y queremos añadir un registro con la clave 15:

Una vez introducido el 15, las pistas quedarían así:

Como vemos ahora la entrada O tiene como clave la del mayor excedente, que en este caso es el único. Veamos ahora que ocurriría si hubiese más de un registro excedente, para ellos supongamos que introducimos la clave 16.

Ahora, la entrada O nos indica cual es el mayor excedente, y su puntero señala al menor excedente. Los registros excedentes de la misma pista se encadenan formando una lista desde el menor al mayor.

La entrada O también es útil para saber a donde debe ir un registro (a qué pista), en nuestro caso por ejemplo, si introducimos el 49, ya sabríamos que pertenece a la pista 1 y que iría a la zona de excedentes de dicha pista.

Para no tener muchos excedentarios se suelen crear registros falsos hacia el final de las pistas. Tienen dos funciones: si llega un registro con la misma clave que uno falso simplemente sobreescribimos el falso y si no el que saldría sería uno falso que no amplia la zona de xjcedentes. Esta es la única forma de dejar huecos que se puede emplear en ISAM.

II.- VSAM (Método de Acceso a Memoria Virtual).

Son tres modos de organización:

Uno para ficheros secuenciales (**ESDS**)
Otro para ficheros de acceso directo o registros de dirección calculada (**RRDS**)

Otro para ficheros secuenciales indexados (**KSDS**) **ESDS** = Conjunto de datos en secuencia de entrada.

KSDS = Conjunto de datos en secuencia de clave.

A) KSDS.

Los tres modos se diferencian de ISAM en que son independientes del hardware o soporte. VSAM independiza las unidades de transferencia del soporte. Su unidad de transferencia son los intervalos de control, los ficheros son mucho más transportables que los de ISAM.

Los intervalos se agrupan en áreas de control (puede ser o no un cilindro).

Dentro de los intervalos de control se pueden dejar espacios libres al final de los mismos y en un área pueden haber intervalos completamente vacíos.

KSDS permite que el índice esté organizado como un árbol B^+.

El tamaño de un área de control suele estar definido por el sistema, lo que se permite es definir el número de intervalos que se quiere que estén vacíos. Podemos definir la longitud de los intervalos de control.

área de Datos + área de índices = Cluster

En el área de índices se tendrá un árbol B^+. Cada nodo del árbol será un intervalo de control. En las hojas se encuentran todas las claves y los nodos de las hojas están enlazados por punteros. Las hojas forman lo que se llama conjunto de secuencias. Los elementos de dicho conjunto son los nodos con entradas que serán: como clave la mayor contenida en un intervalo de control del área de datos, y un puntero a ese intervalo del área de datos.

Cuando hay varios intervalos vacíos habrán nodos que lo indiquen y un puntero a esos intervalos.

El acceso directo se hace con la búsqueda en el árbol. El acceso secuencial se hace empleando los punteros horizontales que enlazan las hojas del índice.

Veamos un esquema que representa como sería el árbol del área de índices, y como estaría unido al área de datos.

Los registros de datos pueden ser de longitud fija o variable, y al principio de cada intervalo hay unos caracteres de control que indican el nivel de ocupación (interesa al hacer un recorrido secuencial).

Al eliminar un registro los que estén a su derecha se moverán a la izquierda dejando siempre los espacios libres al final del intervalo. Y si algún intervalo quedara vacío aparecerá en el conjunto de secuencias como una entrada de

vacío (esto se llama reclamación dinámica de espacio libre).

Este proceso (junto a otros que veremos) evitan la necesidad de tener zonas de excedentes.

Procesos de Partición de Intervalos y de áreas.

En una inserción si hay espacio al final no hay problema. Si el intervalo estuviera completo lo que se hace es partir en dos el intervalo de control, pasando a ocupar una de las mitades alguno de los intervalos libres que queden. A continuación podemos ver un esquema que nos sirve de ejemplo de una situación en la que esto ocurriría:
en un área de control los intervalos

Esto provoca que sea posible que en un área de control los intervalos no estén

en orden de clave, pero lo que siempre estará ordenado es el conjunto de secuencias, por eso en el recorrido secuencial se emplean las hojas.

Lo que queremos decir con esto queda Expresado en el siguiente esquema.

Si se nos acaban los intervalos libres en un área y hay que insertar, lo que se

El inconveniente que tiene con respecto a ISAM es que hay que dejar más espacios libres, a cambio el localizar los registros es mucho más rápido.

B) **RRDS**. Ficheros con registros de dirección calculada o de organización directa

La dirección de los registros en el soporte viene dada por un cálculo sobre la clave primaria de los registros de tal forma que si aplicamos siempre el mismo cálculo sobre una misma clave se obtiene el mismo resultado (que permite llegar hasta el registro).

Puede haber dirección real (se especifica exactamente donde está) o dirección relativa (posición respecto al fichero). Para localizar los registros se aplica el cálculo a la clave del registro buscado.

El cálculo se llama algoritmo de direccionamiento y será definido por el usuario.

La desventaja de este método es que cuando el rango de claves es superior al de registros se desperdicia espacio en el soporte.

El algoritmo que se suele emplear es "aleatorizado", es decir que obtiene números aleatorios pero siempre dentro del rango de claves (Método Hashing).

Puede ocurrir que para claves diferentes se obtenga la misma dirección. A los registros que les ocurre esto se llaman *sinónimos* y deberían estar en la misma posición, lo cual es imposible, por tanto tendremos excedentes.

Generalmente la dirección tendrá espacio suficiente para almacenar más de un registro. Este espacio suele ser de una página. A estos espacios se les llama cubos. Los cubos permiten emplear registros de tamaño variable aunque se estén empleando direcciones relativas (los cubos son de tamaño fijo y en ellos se

busca secuencialmente).

Algoritmos aleatorios (Hashing) más usuales.

(1) **Truncamiento.** (75527 -> 75 / 527 -> 527)

Reduce el rango en función del número de registros que realmente existe, truncando la clave, o sea quedándote con sólo una parte de la clave (los menos significativos, los más importantes, etc..)

(2) **Extracción.** (75527 -> 552)

Nos quedamos con las cifras centrales.

(3) **Selección.**(7 5 5 2 7 -> 5 7 5)

Tomamos determinadas posiciones de la clave y las colocamos en el orden que nosotros queramos. Generalmente se emplea en compañía de otros.

(4) **Multiplicación.**

Multiplicar una parte de la clave por otra parte de la misma (y luego por ejemplo truncar el resultado).

(5) **Cambio De Base.**

Suponer la base en una base diferente a la suya y pasarla a la base en que está.

(6) **División Por Número Primo.**

Se divide la clave por un número primFONTo y como resultado tomamos el resto. Tiene la ventaja de no precisar cálculos posteriores ya que al ser el

número primo más o menos igual al número de registros posibles en el fichero, el resto siempre estará en el rango permitido.

(7) **Clave No Numérica.**

Obtiene direcciones relativas. Se transforma la clave no numérica en numérica (por ejemplo obtener el equivalente binario del carácter).

El método más utilizado es el 6.

¿Cómo tratar los sinónimos?

Existen dos métodos para el tratamiento de sinónimos:

El primero buscar una nueva posición en el espacio reservado al fichero y que esté libre, y el segundo método emplear la zona de excedentes.

*El primer método presenta dos posibilidades:

a) *Sondeo lineal o asignación consecutiva.*

Cuando un registro produce una dirección ocupada, se busca en la siguiente dirección, a continuación de esa, si está ocupada se pasa a la siguiente, y así sucesivamente hasta hallar un espacio libre.

El inconveniente es que tiende a acumular a los registros en zonas de soporte, lo que amplia la posibilidad de que hayan sinónimos.

b) *Doble Hashing.*

Si aplicando el cálculo se obtiene una dirección ocupada, lo que hacemos es a esa dirección obtenida aplicar o bien el mismo cálculo, o bien otro distinto

consiguiéndose así una mayor dispersión de los registros.

$$f(k) \rightarrow D$$

$$f(D) \rightarrow D'$$

Si D' está ocupado se suelen usar zonas de excedentes.

* En cuanto al segundo método existen dos formas de organizar la zona de excedentes:

a) *Secuencial Simple*.

Los sinónimos se colocan secuencialmente en el orden de llegada. Este método se usa cuando se presuponen pocos sinónimos.

b) *Secuencial Encadenado de árboles B (Generalmente B^+)*.

a.

El sinónimo va a la primera posición libre de la zona de Ejcedentes y se encadena con un puntero a la posición original que le correspondería al registro en cuestión. Si viene un segundo sinónimo se encadena al primero que llegó.

Ventajas e inconvenientes de los ficheros con registros de dirección calculada.

- Necesita soportes de acceso directo (Inconveniente)

- Es mejor que los índices cuando se requiere acceso a registros