

Soluciones Ejercicios Sesión 1

1º -

- a) El tamaño del registro será la suma de las longitudes de todos los campos que componen el registro:

$$L_{\text{registro}} = 30 + 9 + 9 + 40 + 9 + 8 + 1 + 4 + 4 + 1 = 115 \text{ bytes}$$

$$N \text{ registros/ Bloque} = \lfloor 512 / 115 \rfloor = \lfloor 4.452 \rfloor = 4 \text{ registros / bloque}$$

$$N \text{ bloques} = \lceil 30000 / 4 \rceil = 7500 \text{ bloques}$$

- Si el campo de búsqueda fuese clave, sólo un registro se tiene que recuperar. El mejor caso sería 1 bloque (está el registro en el primer bloque) y el peor caso sería 7500 bloques (el registro está en el último bloque). De media serían 7500/2 bloques. Si el registro no se encuentra en la tabla, serían 7500 bloques de lectura secuencial.
- Si el campo de búsqueda no es clave, hay valores duplicados, por tanto hay que leer secuencialmente el fichero con lo que son 7500 accesos a bloques

Si el fichero estuviese ordenado según un atributo, la búsqueda binaria de un registro sería de $\lceil \log_2 (7500) \rceil = 13$ accesos. Con la búsqueda binaria leemos ya un bloque con datos válidos.

- b) Se trata de un índice primario (fichero de datos ordenado) sobre un campo clave con lo que el índice puede ser denso o disperso.

El factor de bloques corresponde con el número de registros / bloque. Para el caso del índice primario con clave de búsqueda según el campo NSS, la longitud del registro índice es:

$$L_{\text{registro}} = L_{\text{NSS}} + P_{\text{Reg}} = 9 + 7 = 16 \text{ bytes}$$

$$N \text{ registros/ Bloque} = \lfloor 512 / 16 \rfloor = 32 \text{ registros / bloque}$$

- Si el índice primario fuese denso, habría $nR_i = V(\text{NSS}) = 30000$ entradas, luego el número de bloques del registro índice sería:

$$N \text{ bloques denso} = \lceil 30000 / 32 \rceil = 938 \text{ bloques}$$

$$\text{Haciendo una búsqueda binaria en el índice supondría (está ordenado el índice): } \lceil \log_2 (938) \rceil = 10 \text{ accesos}$$

- Si es un índice multinivel sobre el denso, los niveles superiores al 1 apuntan a bloques de índices, luego el factor de bloque para los índices de nivel 2 en adelante serían:

$$L_{\text{registro}} = L_{\text{NSS}} + P_{\text{Bloque}} = 9 + 6 = 15 \text{ bytes}$$

$$N \text{ registros/ Bloque} = \lfloor 512 / 15 \rfloor = 34 \text{ registros / bloque}$$

Entonces los bloques de promedio de cada nivel serían:

$$\text{Nivel interno: } \lceil 30000 / 32 \rceil = 938 \text{ bloques para apuntar a los 30000 registros de datos.}$$

$$\text{Nivel intermedio: } \lceil 938 / 34 \rceil = 28 \text{ bloques}$$

$$\text{Nivel externo: } \lceil 28 / 34 \rceil = 1 \text{ bloque}$$

$$\text{Número total de bloques del índice multinivel será: } 938 + 28 + 1 = 967 \text{ bloques}$$

Para el índice multinivel:

$$N \text{ accesos} = \text{Número de niveles} + 1 = 3 + 1 = 4 \text{ accesos}$$

- Si el índice primario fuese disperso, habría $\min\{7500, 30.000\}$ entradas (apuntamos al primer registro de cada bloque), luego el número de bloques del registro índice sería:

$N \text{ bloques} = \lceil 7500 / 32 \rceil = 235 \text{ bloques}$

Haciendo una búsqueda binaria en el índice supondría: $\lceil \log_2 (235) \rceil = 8 \text{ accesos}$

- Si es un índice multinivel sobre el disperso, los niveles superiores al 1 apuntan a bloques de índices, luego el factor de bloque para los índices de nivel 2 en adelante serían:

$L_{\text{registro}} = L_{\text{NSS}} + P_{\text{Bloque}} = 9 + 6 = 15 \text{ bytes}$

$N \text{ registros/ Bloque} = \lfloor 512 / 15 \rfloor = 34 \text{ registros / bloque}$

Entonces los bloques de promedio de cada nivel serían:

Nivel interno: $\lceil 7500 / 32 \rceil = 235 \text{ bloques}$ para apuntar a los 7500 bloques de datos.

Nivel intermedio: $\lceil 235 / 34 \rceil = 7 \text{ bloques}$

Nivel externo: $\lceil 7 / 34 \rceil = 1 \text{ bloque}$

Luego con 3 niveles sería el máximo nivel.

- Número total de bloques del índice multinivel será: $235 + 7 + 1 = 243 \text{ bloques}$
- El número de accesos para recuperar un alumno con un NSS dado sería: se debe de recuperar un registro ya que el campo es clave.

Si se efectúa una búsqueda binaria sobre el índice primario disperso de un nivel:

$N \text{ accesos} = \lceil \log_2 (235) \rceil + 1 = 8 + 1 = 9$ para acceder al dato darían 9 accesos a bloques para índice disperso.

Para el caso de índice denso:

$N \text{ accesos} = \lceil \log_2 (938) \rceil + 1 = 10 + 1 = 11 \text{ bloques}$

Para el índice multinivel:

$N \text{ accesos} = \text{Número de niveles} + 1 = 3 + 1 = 4 \text{ accesos}$

- En este caso, el fichero no se encuentra ordenado según el campo clave NSS, por lo que el número de entradas serán de $nR_i = V(\text{NSS}) = 30000$. Índice secundario sobre campo clave y denso.

La longitud del registro será: $9 + 7 = 16 \text{ bytes}$

Factor de bloques $= \lfloor 512 / 16 \rfloor = 32 \text{ registros / bloque}$

$N \text{ bloques registro índice secundario} = \lceil 30000 / 32 \rceil = 938 \text{ bloques}$

Se puede ver que coincide con el índice primario denso al ser un campo clave.

Para crear el índice multinivel, construimos índices dispersos sobre el índice anterior de la forma más eficiente, realizaríamos.

Nivel interno: $\lceil 30000 / 32 \rceil = 938 \text{ bloques}$ para apuntar a los 30000 registros de datos.

Nivel intermedio: $\lceil 938 / 34 \rceil = 28 \text{ bloques}$

Nivel externo: $\lceil 28 / 34 \rceil = 1 \text{ bloque}$

- Número total de bloques del índice multinivel será: $938 + 28 + 1 = 967 \text{ bloques}$
- El número de accesos para recuperar un NSS dado sería: hay que recuperar un registro ya que el campo es clave.

Si se efectúa una búsqueda binaria sobre el índice secundario de un nivel:

$N \text{ accesos} = \lceil \log_2 (938) \rceil + 1 = 10 + 1$ para acceder al dato darían 11 accesos a bloques.
 Para el índice multinivel:
 $N \text{ accesos} = t + 1 = 3 + 1 = 4$ accesos

- d) El índice es secundario sobre un campo clave, luego tiene que ser denso. En este caso, hay que determinar los valores n de los nodos intermedios y padre y n_h de los nodos hoja antes de proceder, teniendo como restricción que cada nodo debe de caber en un bloque de disco.

Para el nodo intermedio, n indica el número de punteros a bloque que caben en un nodo, luego se debe de cumplir que:

$$n * P_{\text{bloque}} + (n-1) * L_{\text{NSS}} \leq B_{\text{util}}$$

Los nodos se encuentran al 69% de capacidad, luego $B_{\text{util}} = B * 0.69 = 512 * 0.69 = 353,28 \rightarrow 353$ bytes

$$n * 6 + (n-1) * 9 \leq 353$$

$15 * n \leq 362 \Rightarrow n = 24$ punteros a bloque, ya que es el mayor valor entero que cumple la condición.

Para el nodo hoja, n_h indica el número de punteros a registro ó claves de búsqueda que caben en un nodo, luego se debe de cumplir que:

$$n_h * (P_{\text{registro}} + L_{\text{NSS}}) + P_{\text{bloque}} \leq B_{\text{util}}$$

$$n_h * (9 + 7) + 6 \leq 353$$

$n_h * 16 \leq 347 \Rightarrow n_h = 21$ punteros a registros ó claves de búsqueda.

Como hay 30000 registros de datos, habrá 30000 punteros a registros en los nodos hoja (índice denso):

Como es un índice multinivel especial, para determinar el número de bloques promedio por nivel, se parte desde los nodos hoja hacia el nodo raíz. Luego **el número de bloques en cada nivel** será:

Nivel hoja: $\lceil 30000 / 21 \rceil = 1429$ bloques para apuntar a los 30000 registros de datos.

Nivel intermedio 1: $\lceil 1429 / 24 \rceil = 60$ bloques

Nivel intermedio 2: $\lceil 60 / 24 \rceil = 3$ bloques

Nivel raíz: $\lceil 3 / 23 \rceil = 1$ bloque

El número de bloques totales será: $1429 + 60 + 3 + 1 = 1493$ bloques

Tiene altura 4 el árbol, por lo que recuperar un alumno con un NSS dado será $4 + 1 = 5$ bloques (hay que recuperar un registro porque el campo es clave).

2º-

- a) ¿Cuánto espacio se requiere para almacenar el archivo y sus índices?

$L_{\text{registro}} = 20 + 40 + 2 + 16 + 32 = 110$ bytes

$B_{\text{util}} = B - L_{\text{control}} = 512 - 0 = 512 \text{ bytes} * 0.65 = 332.8 = 332$ bytes

$N \text{ registros/ Bloque} = \lfloor 332 / 110 \rfloor = \lfloor 3.01 \rfloor = 3$ registros / bloque

$N^\circ \text{ total bloques} = \lceil 100000 / 3 \rceil = 33334$ bloques del archivo de datos

Primer índice B^+ sobre el atributo carnet, que consta de 20 bytes.

Es un índice secundario sobre campo clave, luego es denso. $V(\text{carnet}) = 100000$

- Nodo raíz, intermedio: hay n punteros a bloque $(n-1)$ Valores del campo

$$n * L_{\text{Pbloque}} + (n-1) * L_{\text{carnet}} \leq B_{\text{util}}$$

$B_{\text{util}} = B - L_{\text{control}} = 512 - 0 = 512 \text{ bytes} * 0.69 = 353.28 \text{ bytes} \rightarrow 353$ bytes

$$n \cdot 6 + (n-1) \cdot 20 \leq 353$$

$$26 \cdot n \leq 373 \Rightarrow n = 14 \text{ punteros a bloque.}$$

- Nodo hoja: hay n_h valores de la clave, n_h punteros a registro y 1 puntero a bloque

$$n_h \cdot (P_{\text{registro}} + L_{\text{carnet}}) + P_{\text{bloque}} \leq \text{Butil}$$

$$n_h \cdot (20 + 7) + 6 \leq 353$$

$$n_h \cdot 27 \leq 347 \Rightarrow n_h = 12 \text{ punteros a registros ó claves de búsqueda.}$$

El N° de bloques totales para este índice en promedio es:

Nivel hoja: $\lceil 100000 / 12 \rceil = 8334$ bloques para apuntar a los 30000 bloques de datos.

Nivel intermedio 1: $\lceil 8334 / 14 \rceil = 596$ bloques

Nivel intermedio 2: $\lceil 596 / 14 \rceil = 43$ bloques

Nivel intermedio 3: $\lceil 43 / 14 \rceil = 4$ bloques

Nivel raíz: $\lceil 4 / 14 \rceil = 1$ bloque

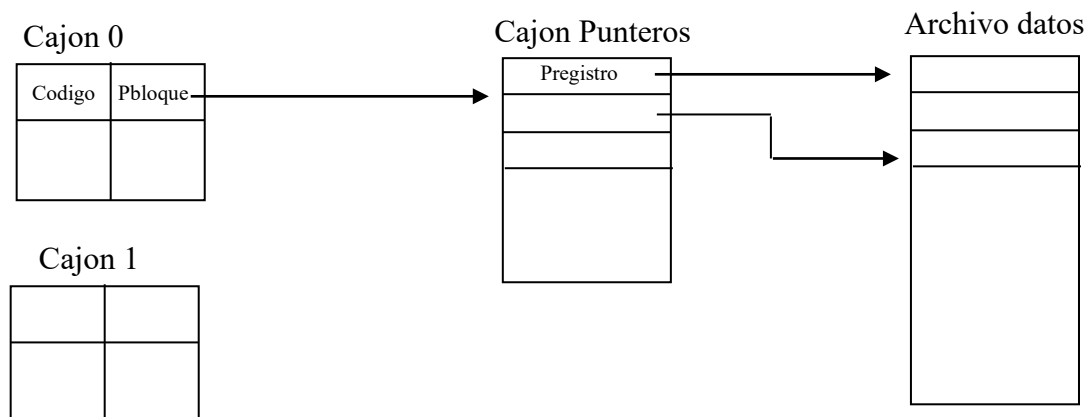
Número de bloques: $8334 + 596 + 43 + 4 + 1 = 8978$ bloques

Segundo índice asociativo hash sobre el atributo código_carrera (2 bytes)

Es un índice secundario sobre campo no clave, lo que implica denso y tiene cajones de punteros.

La función es $\text{codigo_carrera} \bmod 8$, por lo que la función de asociación sólo puede direccionar 8 cajones.

Como hay 16 carreras diferentes $V(\text{código_carrera}, r) = 16$, y hay 8 cajones, habrá 2 códigos de carrera en cada cajón de media. El esquema será el siguiente:



La longitud del registro índice será: $L_{\text{registro}} = 6 + 2 = 8$ bytes

$\text{Butil} = B - L_{\text{control}} = 512 \text{ bytes} \cdot 0.69 = 353 \text{ bytes}$

$\text{Factor Bloque} = \lfloor 353 / 8 \rfloor = 44 \text{ registros / bloque}$

Como hay 2 registros por cajón, entonces cada cajón ocupará 1 bloque.

Como las carreras se encuentran distribuidas uniformemente entre los alumnos, habrá $100000/16=6250$ alumnos por cada carrera. Cada cajón de punteros deberá de contener 6250 punteros a registro

El factor de bloque del cajón de punteros será: $\lfloor 353 / 7 \rfloor = 50$ punteros al 69%

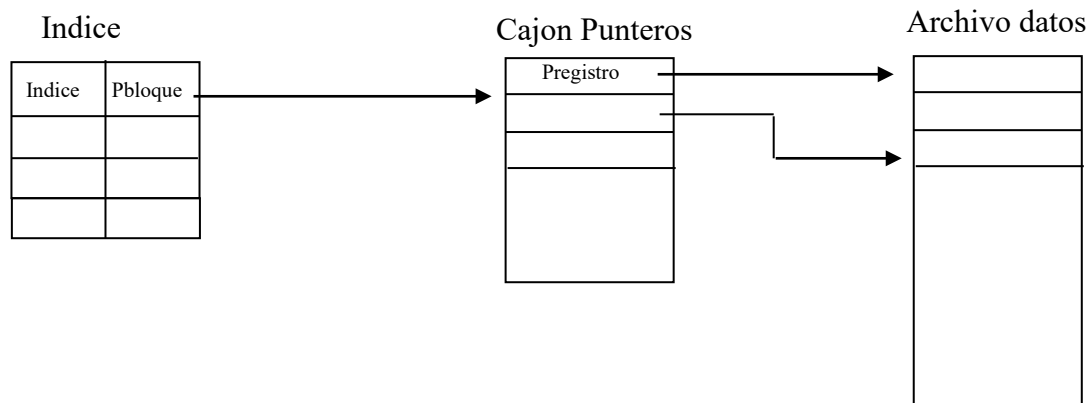
El número de bloques necesarios para almacenar cada cajón de punteros será: $\lceil 6250 / 50 \rceil = 125$ bloques.

El tamaño de este índice será: $8 \text{ cajones} \cdot 1 \text{ bloque} + 16 \text{ cajones de punteros} \cdot 125 \text{ bloques} = 8 \cdot 1 + 125 \cdot 16 = 2008 \text{ bloques}$

Tercer índice secundario sobre atributo índice_académico (32 bytes)

Este índice es secundario sobre un campo no clave, lo que implica que es denso y necesita cajones de punteros.

Este índice se debe de construir sobre el índice académico que tiene 8 valores diferentes ($V(\text{índice_academico})=8$), luego el esquema del índice será:



La longitud de registro del índice será: $L_{\text{registro}} = L_{\text{índice}} + P_{\text{bloque}} = 32 + 6 = 38$ bytes

Factor Bloque de índice = $\lfloor 353 / 38 \rfloor = 9$ registros / bloque al 69%

Luego con 1 bloque sería suficiente para guardar el índice.

Si el índice académico está distribuido uniformemente, cada cajón de punteros debería de tener $100000/8=12500$ registros

El factor de bloque del cajón de punteros sería: $\lfloor 353 / 7 \rfloor = 50$ registros al 69%,

Con lo que cada cajón de punteros a registro tendría:

$$\lceil 12500 / 50 \rceil = 250 \text{ bloques.}$$

La estructura del índice ocuparía: $1 + 250 \cdot 8 = 2001$ bloques.

- b) Si se desea insertar un registro nuevo de un estudiante. ¿Cuál es el coste de realizar dicha operación?

Al insertar un nuevo alumno hay que insertar el registro en el archivo de datos y además hay que modificar cada una de las estructuras índice:

1º - Índice primario: habría que recorrer el árbol B^+ hasta localizar la posición donde debería de estar, ordenar el nodo hoja, localizar el bloque de datos donde tiene que ir el nuevo registro (leerlo, modificarlo y escribirlo) y además insertar un puntero a registro en el nodo hoja. Esta operación lleva: 5 bloques para localizar + 1 para insertar el puntero a registro + 1 leer el bloque de datos donde se va a insertar + 1 para escribir el bloque de datos con el registro = 8 bloques. Tener en cuenta que los bloques de datos y de los índices no están llenos y por lo tanto no hace falta crear nuevos bloques.

2º - Insertar puntero con valor de registro para código de carrera en el cajón de punteros correcto: 1 bloque para acceder al cajón + 1 bloque leer el primer bloque de punteros + 1 bloque de escribir el bloque actualizado = 3 bloques.

3º - Recorrer el índice de índice académico para localizar en qué cajón hay que insertar el puntero a registro nuevo = 3 bloques.

Total = $8+3+3=14$ bloques.

- c) Si se desea buscar un estudiante por su carnet, ¿Cuánto cuesta esta operación?
Hay que recuperar sólo un registro ya que el campo carnet es clave.

Nivel del árbol +1 = $5+1 = 6$

- d) ¿Cuál es el coste de listar a todos los estudiantes que cursan una carrera dada?. Asumir el peor caso.

Si se quiere recuperar una carrera dada, el número de registros esperados será $100000/V(\text{código_carrera})=100000/16=6250$ alumnos

Hay que procesar la función hash, leer el cajón adecuado, leer todos los punteros a registros y leer del archivo de datos: 1 bloque + 125 bloques + 6250 bloques (peor caso, todos estén en un bloque diferente) = 6376

- e) ¿Cuál es el coste de listar a todos los estudiantes que tienen un índice académico mayor que 3.0?. Asumir el peor caso.

Recuperar los alumnos de un índice académico supone obtener $100000/V(\text{índice_academico},r)=100000/8=12500$ alumnos

Recorrer el índice encontrando los valores mayores que 3, leer todos los punteros a registros y leer el archivo de datos: 1 bloque + $250*4$ (4 valores que cumplen la condición) + $12500*4$ (leer cada bloque donde se encuentra el dato) = 51001

- f) ¿Cómo se podrían mejorar los procesos de lectura anteriores?

Los procesos de lectura anteriores se podrían mejorar leyendo los punteros a registros, obteniendo el número de bloque donde se encuentran los datos y ordenando secuencialmente por el número de bloque a leer, para así realizar menos accesos a disco, debido a que puede haber varios registros que se deban de leer y caigan en el mismo bloque.

3º -

- a) Organización asociativa con 1000 cajones.

Darse cuenta de que en este caso no es un índice + archivo de datos, es sólo un archivo de datos que se organiza de manera asociativa.

Butil=B-Lcontrol = $1000-0=1000$ bytes al 100% de capacidad.

N registros/ Bloque = $\lfloor 1000 / 200 \rfloor = 5$ registros / bloque

Si hay 1.000.000 registros, cada cajón contendrá $1.000.000 / 1000 = 1000$ registros

Nº bloques por cajón = $\lceil 1000 / 5 \rceil = 200$ bloques por cajón.

Luego el tamaño del fichero será: $1000*200=200.000$ bloques.

Luego el número de accesos en promedio para localizar un dato en un cajón sería: $200/2 = 100$ bloques. Es un campo clave, mejor caso 1 bloque, peor caso 200 bloques, la media 100 bloques.

b) Para un índice primario de un nivel

Es un índice primario sobre campo clave y además denso.

$$L_{\text{registro}} = L_{\text{NSS}} + P_{\text{Reg}} = 10 + 5 = 15 \text{ bytes}$$

$$N \text{ registros/ Bloque} = \lfloor 1000 / 15 \rfloor = 66 \text{ registros / bloque}$$

$$N \text{ bloques} = \lceil 1000000 / 66 \rceil = 15152 \text{ bloques}$$

$$\text{Haciendo un búsqueda binaria en el índice supondría: } = C_{\text{índice}} + C_{\text{datos}} = \lceil \log_2 (15152) \rceil + 1 = 15 \text{ accesos}$$

c) Para el árbol B⁺

Es un índice primario sobre campo clave y además denso.

Para el nodo intermedio, n indica el número de punteros a bloque que caben en un nodo, luego se debe de cumplir que:

$$n * P_{\text{bloque}} + (n-1) * L_K \leq \text{Butil}$$

$$n * 5 + (n-1) * 10 \leq 1000$$

$$15 * n \leq 1010 \Rightarrow n = 67 \text{ punteros a bloque}$$

Para el nodo hoja, en este caso al ser el puntero de registro igual al puntero de bloque, el nodo intermedio es igual al nodo hoja, por lo que en promedio hay 66 valores del campo

Suponiendo que el árbol tendría todos su nodos al 100% de capacidad, el número de bloques que podría almacenar en promedio cada nivel sería:

$$\text{Nivel hoja: } \lceil 1000000 / 66 \rceil = 15152 \text{ bloques para apuntar a los 30000 registros de datos.}$$

$$\text{Nivel intermedio 1: } \lceil 15152 / 67 \rceil = 227 \text{ bloques}$$

$$\text{Nivel intermedio 2: } \lceil 227 / 67 \rceil = 4 \text{ bloques}$$

$$\text{Nivel raíz: } \lceil 4 / 67 \rceil = 1 \text{ bloque}$$

$$\text{El número total de bloques del índice será: } 15152 + 227 + 4 + 1 = 15384 \text{ bloques}$$

Por lo que el árbol tendrá una altura de 4 niveles y por lo tanto el número de accesos para localizar un registro será de 5 (4 +1 de recuperar un registro al ser clave).

En este caso se puede ver que todos los nodos de este árbol son iguales, ya que el puntero a registro es igual al puntero a bloque.

4º -

Se trata de organización de archivo, no hay índice. El problema se puede esquematizar en lo siguiente:

Nodos intermedios /Raíz

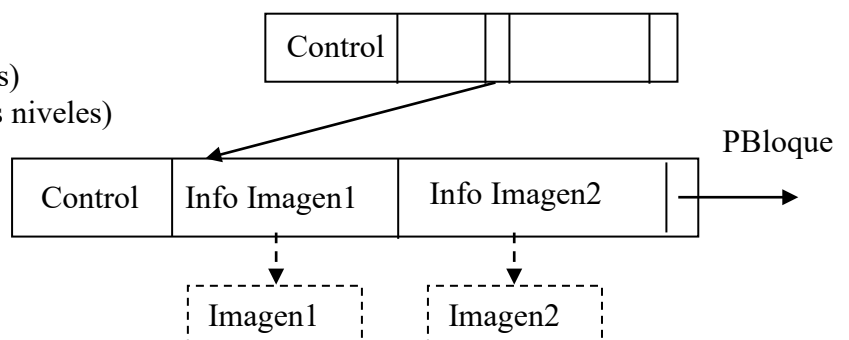
(nodos del árbol B⁺, páginas de índices)

con su información de control (2 o más niveles)

Nodos hoja (Páginas de datos)

(información a

organizar) con su información de control



(Ficheros de imágenes)

a) Los nodos hoja que contienen la información a organizar tienen una información de control de 196 bytes. La longitud del registro es de $L_R = 128$ bytes que es la información a organizar.

Nodos hoja

$$\text{Butil} = B - L_{\text{control}} = 4096 - 196 = 3900 \text{ bytes} * 0.8 = 3120 \text{ bytes}$$

$$nh * L_{\text{registro}} + L_{\text{bloque}} \leq \text{Butil}$$

$$nh * 128 + 4 \leq 3120$$

$$nh \leq 3116/128 = 24 \text{ registros por cada nodo hoja}$$

Nodos intermedios/Raíz

El nodo del árbol B^+ (nodos intermedio y raíz) deberá de tener una información de control de 12 bytes, los valores de la clave de búsqueda y los punteros a bloque:

$$\text{Butil} = B - L_{\text{control}} = 4096 - 12 = 4084 \text{ bytes} * 0.8 = 3267.2 \text{ bytes} \rightarrow 3267 \text{ bytes}$$

$$n * L_{\text{puntero_bloque}} + (n - 1) * L_{\text{clave}} \leq B$$

$$n * 4 + (n - 1) * 12 \leq 3267$$

$$16 * n \leq 3.279 \Rightarrow n = 204 \text{ punteros ó 203 valores de la clave de búsqueda}$$

Hay 100.000 imágenes que hay que organizar, luego primero se calcula el número de nodos hoja para esas 100.000 imágenes, y luego los nodos intermedio más raíz:

$$\text{Nivel hoja: } \lceil 100000 / 24 \rceil = 4167 \text{ bloques}$$

$$\text{Nivel intermedio 1: } \lceil 4167 / 204 \rceil = 21 \text{ bloques}$$

$$\text{Nivel intermedio 2: } \lceil 21 / 204 \rceil = 1 \text{ bloques}$$

Número medio de nodos para 100.000 imágenes será: $1 + 21 + 4167 = 4188$ bloques

La capacidad máxima del árbol sería para tres niveles al 80%:

Nivel	Nodos	Puneros	Imágenes organizadas
Nivel 1 (Raíz)	1	205	
Nivel 2	205	$205 * 205 = 42.025$	
Nivel hoja	42.025		$42.025 * 24 = 1.008.600$

Luego entonces hay 2 niveles de bloques índices del árbol B^+ más 1 nivel de las páginas de datos. En total 3 niveles.

El número máximo de nodos será de: $1 + 205 + 42.025 = 42.231$ nodos

Cada nivel indexa como máximo al 80% (es decir, si el árbol fuese de 1, 2 o 3 niveles):

24 para árbol de nivel 1

4.920 para árbol de nivel 2

1.008.600 para las páginas de datos para un árbol de 3 niveles

Número máximo de páginas de datos: 42.025

Número máximo de registros que se pueden almacenar: 1.008.600

El espacio que ocupa cada nivel como máximo será:

1 nodo para nivel 1 * 4Kbytes = 4 Kbytes.

205 nodos para nivel 2 * 4 Kbytes = 820 Kbytes

42.025 nodos de datos * 4 Kbytes = 168.100 Kbytes

Luego en total son 168.924 Kbytes \Rightarrow 164.1 MBytes. Como máximo

b) En 824 KB de memoria caben la raíz y el nivel 1 por completo del árbol B⁺ (4 KB + 820 KB), luego para localizar una imagen se necesitan:

2 accesos a **memoria** (raíz y nivel 1), que no suponen coste efectivo alguno (no es disco)

+ 1 acceso a **disco** para el nodo de datos (nodo hoja)

+ 1 acceso a **disco** para la página de la imagen

= 2 accesos a **disco**.

Recuperar 100.000 imágenes por la clave, estando las páginas de datos al 80% supondría que es necesario "navegar" o recorrer en secuencia únicamente todos nodos de datos. El número promedio de éstos será de $\lceil 100.000 / 24 \rceil = 4.167$ páginas de datos (o accesos), más recuperar 100.000 imágenes en fichero. En total supondría 104.167 accesos al disco.

5º -

(a) **Número de I/Os: 1** (Cada rango posee 3.000 registros, que caben en 1 bloque cada uno)

(b) Construyamos una tabla simplemente con los datos del enunciado:

Rango	Prob	# Registros	# Bloques
[1]	1/15	1.000	1
[2]	2/15	2.000	1
[3]	3/15	3.000	1
[4]	4/15	4.000	2
[5]	5/15	5.000	2

Usando la tabla anterior, se puede ver que:

$$\text{Número de I/Os: } \frac{1 \times (1000 + 2000 + 3000) + 2 \times (4000 + 5000)}{1000 + 2000 + 3000 + 4000 + 5000} = 1,6$$

(c) **Número de I/Os: 5** (El rango [1] contiene 15.000 registros, que caben en 5 bloques, mientras que los demás contienen 0 registros y usan 0 bloques)

(d) **Número de I/Os: 2** (Un árbol B⁺ con 15.000 registros y 3.300 punteros tiene 2 niveles)

(e) La estructura en retícula es mejor, siempre que los datos no estén *extremadamente* sesgados, como ocurren en el caso (c).

Un índice en retícula proporciona acceso directo a los bloques que contienen los datos en un rango dado. Un árbol B⁺ requiere atravesar primero desde la raíz a la primera hoja.

(f) Retícula.

Las tablas con estructura asociativa (*hash*) no preservan ninguna clase de orden, por lo que no son buenas en absoluto para consultas de rangos.

6º -

1. Calcule el número N de cajones distintos necesarios. *Justifíquelo*. [2 p]

El número de cajones es, por definición, simplemente el número de valores distintos de la función *hash*. Para la función dada y el rango de valores de entrada dado, obtenemos que h es un entero en el rango $0 \leq h \leq 4$, por lo que:

$$N = 5 \text{ cajones}$$

2. Suponga que los valores de A están *uniformemente distribuidos*, esto es, un registro cualquiera puede estar en cualquiera de los N cajones con igual probabilidad, y una clave de búsqueda puede estar en cualquier rango con igual probabilidad. ¿Cuál es el número de operaciones de I/O esperado para buscar un registro? *Justifíquelo*. [3 p]

Cada cajón posee $16.000 / 5 = 3.200$ registros (< 4.000), que caben en 1 bloque cada uno.

$$\text{Número de I/Os} = 1$$

3. Ahora suponga que existe un *cierto sesgo* en los datos. En particular, la probabilidad de que un valor de A (tanto en el registro como en la búsqueda) esté en el cajón i es $2^i / N(N+1)$ (tenga en cuenta que la suma de estas probabilidades, desde $i=1$ hasta N , vale 1). ¿Cuál es el número de operaciones de I/O esperado para buscar un registro? *Justifíquelo*. [3 p]

Para $N = 5$, la expresión $2^i / N(N+1)$ queda simplemente $i / 15$

Cajón	Prob.	# Registros	# Bloques
[1]	1/15	1.067	1
[2]	2/15	2.133	1
[3]	3/15	3.200	1
[4]	4/15	4.267	2
[5]	5/15	5.333	2

Usando la tabla anterior, se puede ver que:

$$\text{Número de I/Os} = 1 \times \frac{1}{15} + 1 \times \frac{2}{15} + 1 \times \frac{3}{15} + 2 \times \frac{4}{15} + 2 \times \frac{5}{15} = \frac{24}{15} = 1,6$$

4. Ahora considere un caso de *sesgo extremo*. La probabilidad de que un valor de A (tanto en el registro como en la búsqueda) esté en el cajón 1 ó 2 es de $1/2$ y, por lo tanto, la probabilidad es 0 para los otros rangos. ¿Cuál es el número de operaciones de I/O esperado para buscar un registro? *Justifíquelo*. [3 p]

Cajón	Prob.	# Registros	# Bloques
[1]	1/2	8.000	2
[2]	1/2	8.000	2
[3]	0	0	0
[4]	0	0	0
[5]	0	0	0

Usando la tabla anterior, se puede ver que:

$$\text{Número de I/Os} = 2 \times \frac{1}{2} + 2 \times \frac{1}{2} = 2$$

5. Si se usa un árbol B+ para la misma aplicación, ¿Cuál es el número de operaciones de I/O esperado? Suponga que los nodos de un árbol B+ contienen también 4.000 punteros y claves. *Justifique* porqué. [2 p]

Un índice con estructura de árbol B+ con 4.000 claves en cada nodo, tiene **2 niveles** para indexar 16.000 registros. Por lo tanto:

Número de I/Os = 2

6. *Justifique* qué índice, el asociativo o un árbol B+, es mejor para consultas de rangos en dos situaciones: la actual y una general. [2 p]

En el presente problema, en **todas** las situaciones, incluso en la de sesgo extremo, el índice asociativo es mejor (o como mucho) igual de costoso que un índice B+.

Sin embargo, los índices asociativos no se construyen generalmente usando funciones similares a la presente (p.e. la función *h* usada no posee algunas de las propiedades deseadas en una función *hash*, como la propiedad de dispersión, etc.), por lo que los índices asociativos son extremadamente ineficientes para búsquedas de rangos (se debería buscar valor a valor), por lo que un índice de árbol B+ resulta más conveniente.

7º -

1. $\sigma_A < 5,000$ (R)

$\sigma_A < 50.000$ (R)

Tanto el acceso sobre los datos directamente, como el uso de un índice B+, operan sobre estructuras (datos o índices según el caso) ordenadas. Para esta selección, la elección de acceder directamente al fichero secuencial ordenado es ligeramente mejor en coste a usar un índice con estructura de árbol B+, simplemente por la sobrecarga requerida para navegar por el árbol y buscar cada bloque con el puntero a los datos.

2. $\sigma_A = 50.000$ (R)

$\sigma_A = 50.000$ (R)

Un índice asociativo será más eficiente aquí, dado que sólo habrá que leer el bloque del cajón de índices correspondiente al valor buscado.

3. $\sigma_A > 50,000 \wedge A < 50,010$ (R)

$\sigma_A > 50.000 \wedge A < 50.010$ (R)

Un índice con estructura de árbol B+ será más eficiente aquí, dado que para encontrar el primer valor sólo habrá que recorrer el árbol en profundidad y a partir de ese punto son accesos consecutivos a los bloques (ordenados) del árbol. Leer el fichero de datos ordenado hasta encontrar el punto de comienzo del intervalo implica una búsqueda binaria que es mayor que el coste de atravesar un árbol equivalente.

4. $\sigma_A \neq 50,000$ (R)

$\sigma_A \neq 50,000$ (R)

Puesto que esta selección requerirá analizar (leer) todas las entradas disponibles, y en el caso del árbol B+ vamos a empezar desde el principio del índice ordenado, acceder directamente al fichero de datos ordenado será más eficiente, nuevamente debido al sobrecoste de la búsqueda en el índice.