



UA

Unidad 1: Planificación del Almacenamiento e Indexación

*Bases de Datos Avanzadas, Sesión 2:
Planificación del Almacenamiento de
Registros*

*Iván González Diego
Dept. Ciencias de la Computación
Universidad de Alcalá*



INDICE

- *Organización de los archivos.*
- *Estimación tamaño de un archivo*
- *Secuenciales o Cluster Index*
- *Arboles B^+*
- *Hash*
- *Agrupaciones*
- *Estimación de número de registros y coste*
- *Ejemplo*

Referencias: Silberschatz 4ª Ed. Pp 249 - 315
Elmasri, 3ª Ed. Pp 105 - 181



Organización de los archivos

- *Vista lógico \Rightarrow archivos son secuencias de registros que se deberían de corresponder con bloques de disco.*
- *Rara vez un registro va a ocupar exactamente un bloque de disco*
- *Registros \Rightarrow Tamaño variable (cadenas texto, arrays, etc).*
- *En bases de datos se manejan dos supuestos:*
 - *Registros de tamaño fijo \Rightarrow gran velocidad de acceso.*
 - *Registros de tamaño variable \Rightarrow gran eficiencia en el espacio*



Estimación tamaño de un archivo

- ❑ *Secuencia lógica de bloques que se almacenan en disco.*
- ❑ $B \rightarrow$ tamaño del bloque (bytes)
- $n_R \rightarrow$ número de registros del archivo
- ❑ $L_R = \sum L_{C_i} \rightarrow$ longitud del registro (suma de bytes de cada campo C_i)
- ❑ $f_R \rightarrow$ factor de bloque (número de registros enteros que caben en 1 bloque)

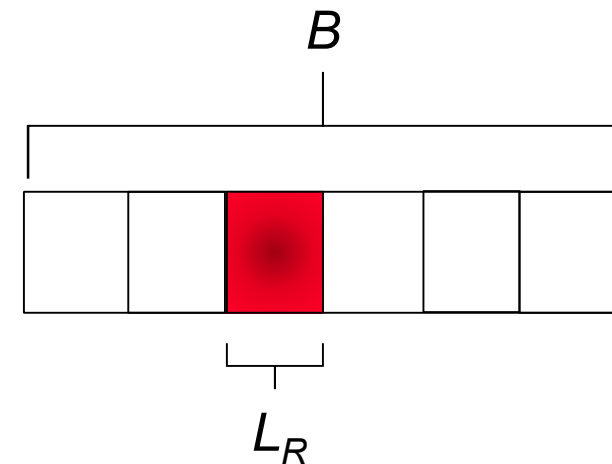
$$f_R = \lfloor B/L_R \rfloor$$

- ❑ $b_R \rightarrow$ Número de bloques del archivo

$$b_R = \lceil n_R / f_R \rceil$$

- ❑ *Tamaño en disco (bytes) $\rightarrow b_R * B$*
- ❑ *Puede haber información de control L_c*

$$f_R = \lfloor (B-L_c)/L_R \rfloor$$





Organización de los registros en los archivos

Varios tipos:

- *Montículo: No hay ordenación. Se colocan los registros donde hay espacio libre.*
- *Archivos secuenciales: Se colocan ordenados a partir de una clave de búsqueda*
- *Organización por árboles B^+*
- *Organización asociativa hash:*
 - *Función hash que relaciona algún atributo de cada registro.*
 - *Dependiendo del resultado, se sitúa el registro en un bloque.*
- *Agrupaciones:*
 - *Múltiples relaciones (tablas) en el mismo archivo.*
 - *Registros de distintas relaciones se guardan en el mismo bloque.*



Organización de los registros en los archivos: Archivos secuenciales

- Están diseñados para maximizar la capacidad de procesamiento relacionada con un atributo determinado de los registros, que suele ser la clave de búsqueda
- Los registros se vinculan con punteros.
- Permite la lectura de registros en orden \Rightarrow útil en ciertos algoritmos \rightarrow Búsqueda binaria
- Difícil mantener el orden físico cuando se insertan o borran registros.
- Ocupa:

$$f_R = \lfloor B/L_R \rfloor$$

$$b_R = \lceil n_R / f_R \rceil$$

C-215	Becerril	700	
C-101	Centro	500	
C-110	Centro	600	
C-305	Collado Mediano	350	
C-217	Galapagar	750	
C-222	Moralzarzal	700	
C-102	Navacerrada	400	
C-201	Navacerrada	900	
C-218	Navacerrada	700	



Organización de los registros en los archivos: Archivos secuenciales

□ Inserción:

- Localizar el registro que precede al que se va a insertar según el orden de la clave.
- Si existe un hueco libre **en el bloque** de ese registro \Rightarrow se insertará ahí el nuevo registro
- Si no hay hueco \Rightarrow se inserta el registro en un **bloque de desbordamiento**
- Actualización de los punteros necesarios:
 - Puntero del nuevo registro.
 - Puntero del registro anterior.



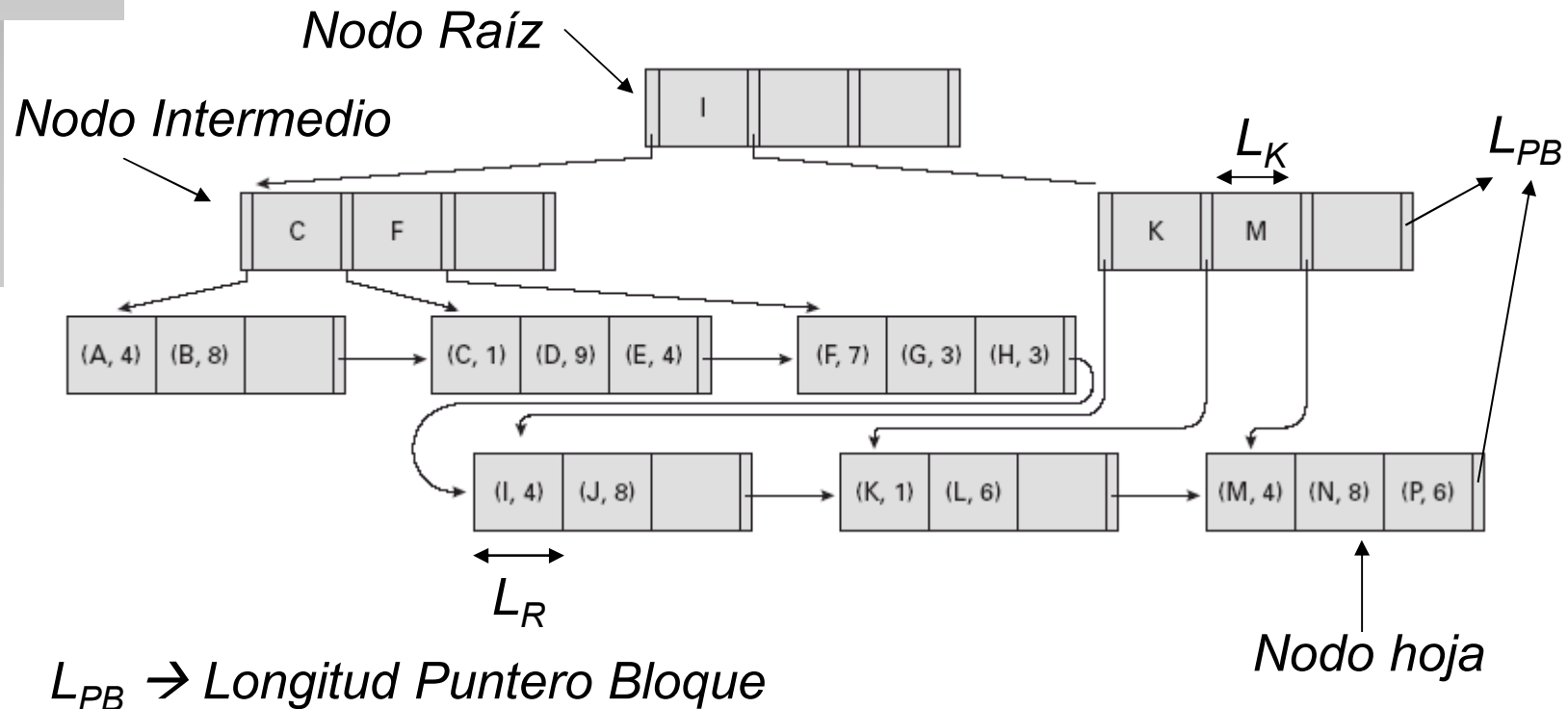
□ Problema:

- Puede perderse el orden por completo \Rightarrow Reordenar el fichero.



Organización de Archivos con Árbol B^+

- También es posible utilizar árboles B^+ para organizar los registros de los archivos.
- Parecido al árbol binario pero con más valores en cada nodo.
- Los nodos hojas almacenan registros.





Organización de Archivos con Árbol B^+

Nodo Raíz / Intermedio: $n * L_{PB} + (n-1) * L_K \leq B$

Nodo Hoja: $n_h * L_R + L_{PB} \leq B$

□ Número de boques del archivo:

- Número de nodos hoja: $N_{hoja} = \lceil n_R / n_h \rceil$
- Número bloques intermedio 1: $N_{Int1} = \lceil N_{hoja} / n \rceil$
- Número bloques intermedio 2: $N_{Int2} = \lceil N_{Int1} / n \rceil$
- Número bloques intermedio 3: $N_{Int3} = \lceil N_{Int2} / n \rceil$
-
- Número de bloques raíz: 1



Asociación Estática

- *Es una técnica basada en la utilización de funciones de asociación.*
- *Sea K el conjunto de los valores de clave de búsqueda.*
- *Se B el conjunto de todas las direcciones de los cajones.*
- *Una función h es una función de K a B tal que:*

$$h(k)=B$$

- *Número de cajones $N_c \rightarrow$ Número de cajones: Número de valores diferentes de la función de asociación.*
- *Ejemplo: $h(x)=x \bmod 10 \rightarrow 0,1,2,3,4,5,6,7,8,9 \rightarrow 10$ cajones*
 $h(x)=\lfloor x/100 \rfloor$, sabiendo que $\min(x)=0$, $\max(x)=399$
 $0,1,2,3 \rightarrow 4$ cajones



Funciones de Asociación

☐ *Distribución Uniforme:*

- *Cada cajón tiene asignado el mismo número de valores de la clave de búsqueda dentro del conjunto de todos los valores posibles.*

☐ *Distribución Aleatoria:*

- *En el caso medio, cada cajón tendrá casi el mismo número de valores asignados a él, sin tener en cuenta la distribución actual de los valores de la clave de búsqueda.*
- *El valor de asociación no será correlativo a ninguna orden exterior visible en los valores de la clave de búsqueda.*



Organización de archivos por asociación

Cajón 0

--	--	--

Cajón 1

--	--	--

Cajón 2

--	--	--

Cajón 3

C-217	Barcelona	750
C-101	Daimiel	500

Cajón 4

C-222	Reus	700

Cajón 5

C-102	Pamplona	400
C-201	Pamplona	900
C-218	Pamplona	700

Cajón 6

--	--	--

Cajón 7

C-215	Madrid	700

Cajón 8

C-305	Ronda	350
C-110	Daimiel	600

Cajón 9

--	--	--

B_C

L_R

n_{RC}

Si equiprobabilidad:

$$n_{RC} = n_R / N_C$$

$$n_{RC} = \text{nº registros / cajón}$$

$$f_{rc} = \lfloor B / L_R \rfloor$$

$$B_C = \lceil n_{RC} / f_r \rceil$$

$$\text{Número bloques: } N_C * B_C$$

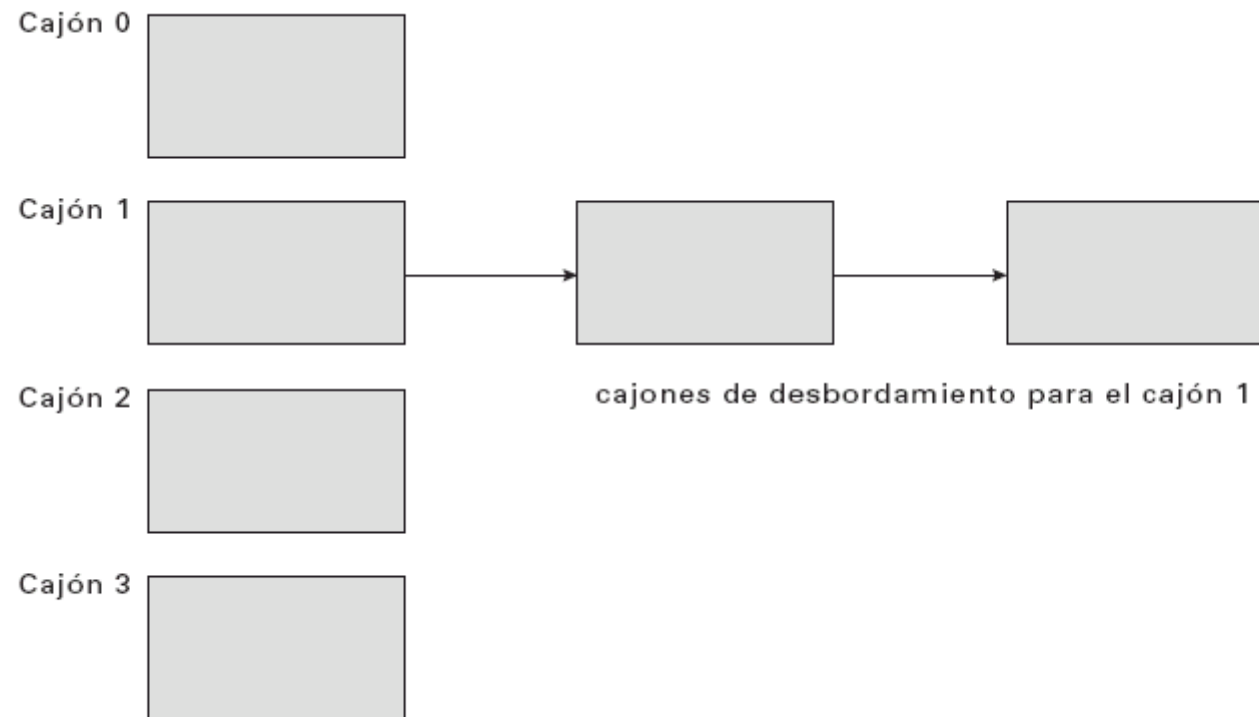


Gestión del desbordamiento de cajones

- *Cajones insuficientes*
- *El número de cajones debe ser escogido tal que $n_c > n_r / f_r$*
- *Atasco*
 - *Algunos cajones tienen asignados más registros que otros.*
 - *Puede ocurrir:*
 - *Varios registros podrían tener la misma clave de búsqueda.*
 - *La función de asociación elegida podría producir una distribución irregular de las claves de búsqueda.*
- *En general $n_c = (n_r / n_f) * (1 + d)$ donde d suele ser un factor de 0.2 aproximadamente.*
- *Cajones de desbordamiento + lista enlazada*



Gestión del desbordamiento de cajones





Organización de los registros en los archivos: Agrupaciones

- ❑ *Utilizado para bases de datos de tamaño relativamente grande en el que se utiliza un único archivo.*
- ❑ *La gestión del archivo y su estructura pasa del S.O. al SGBD.*
- ❑ *Se optimiza su estructura para la ejecución de consultas.*
- ❑ *Su estructura está diseñada para poder mezclar los datos de distintas relaciones dentro del mismo bloque de disco \Rightarrow puede recuperar datos de varias relaciones en una lectura \Rightarrow mejora la ejecución de ciertas consultas.*
- ❑ *Sistema inteligente de bases de datos \Rightarrow capaz de deducir posibles mezclas de datos de forma automática.*
- ❑ *Mezcla de tuplas de distintas relaciones \Rightarrow punteros de situación \Rightarrow permitiendo acceder a las tuplas de las relaciones de forma independiente cuando es necesario.*



Organización de los registros en los archivos: Agrupaciones

nombre-cliente	número-cuenta
López	C-102
López	C-220
López	C-503
Abril	C-305

nombre-cliente	calle-cliente	ciudad-cliente
López	Principal	Arganzuela
Abril	Preciados	Valsaín

```
select número-cuenta, nombre-cliente, calle-cliente,  
       ciudad-cliente  
from impositor, cliente  
where impositor.nombre-cliente = cliente.nombre-  
      cliente
```

```
select *  
from cliente
```

López	Mayor	Arganzuela
López	C-102	
López	C-220	
López	C-503	
Abril	Preciados	Valsaín
Abril	C-305	

López	Mayor	Arganzuela	
López	C-102		
López	C-220		
López	C-503		
Abril	Preciados	Valsaín	
Abril	C-305		



Estimación registros/bloques a recuperar del archivo de datos

- $V(A) \rightarrow$ Número de valores diferentes campo A en archivo / tabla R
- Número registros a recuperar $\rightarrow n_{rc}$, ejemplo $\sigma_{A=1}(R)$?
- Si el campo es clave $\rightarrow n_{rc}=1$,
 - Archivo no ordenado por A , mejor caso 1 bloque, peor b_R , media $b_R / 2$
 - Archivo ordenado por A , búsqueda binaria : $\lceil \log_2 (b_R) \rceil$
- Si el campo no es clave $\rightarrow n_{rc}=n_R / V(A)$
 - Si el archivo no ordenado por A , \rightarrow leer b_R bloques (peor caso)
 - Si el archivo datos ordenado \rightarrow leer : $\lceil \log_2 (b_R) \rceil + \lceil n_{rc} / f_R \rceil - 1$ bloques.
- Ejemplo: $f_R=2$ (registros /bloque), $n_R=6$, $b_R=3$, $\sigma_{valor=A}(R)$?

A
A
A
A
B
B

Ordenado

$n_{rc}=6 / 2=3$ reg.

Leer = $\lceil \log_2 (3) \rceil + \lceil 3/2 \rceil - 1 = 3$ bl.

A
B
A
A
B
A

No Ordenado

$n_{rc}=6 / 2=3$ reg.

Leer = 3 bl.



Ejemplo

- Considerar un archivo de datos que contiene información sobre estudiantes, donde se almacena:
 - Número de carnet: 20 bytes, es la PK
 - Nombre del alumno: 40 bytes.
 - Código de carrera: 2 bytes, in (0,1,2,3,4,5,6,7,8,9)
 - Edad: 16 bytes.
 - Índice académico: 32 bytes.
- Si hay 100.000 registros y cada bloque es de 512 bytes. ¿Cuál es el espacio necesario para almacenar el archivo?
- Si el bloque está ocupado al 65%, ¿Cuál será el espacio del disco ahora?. Redondear al entero más cercano.
- Coste y registros para $\sigma_{\text{carnet}=2345}(r)$?
- Coste y registros para $\sigma_{\text{codigo_carrera}=2}(r)$?
- Si está ordenado el archivo por carnet, $\sigma_{\text{carnet}=2345}(r)$?
- Si está ordenado el archivo por código carrera, $\sigma_{\text{codigo_carrera}=2}(r)$? 18



Ejemplo

- ❑ *La longitud del registro será:*
 $LR = 20 + 40 + 2 + 16 + 32 = 110 \text{ bytes.}$
- ❑ *Número de registros / bloque = Factor de bloque = $\lfloor 512 / 110 \rfloor = \lfloor 4.65 \rfloor = 4$*
- ❑ *Número de bloques = $\lceil 100.000 / 4 \rceil = 25.000$ bloques.*
- ❑ *Si cada bloque ocupa 512 bytes = $25.000 * 512 = 12.20 \text{ MB}$*



Ejemplo

- Si el bloque está al 65%, entonces el número de registros / bloque ahora será : $4 * 0.65 = 2.63 \Rightarrow 3$ registros
- Número de bloques = $\lceil 100.000 / 3 \rceil = 33.334$ bloques.
- Si cada bloque ocupa 512 bytes = $33.334 * 512 = 16.27$ MB
- Si ahora se utiliza la lista enlazada entre registros para seguir un orden lógico del fichero y cada puntero ocupa 12 bytes, ¿Cuál es el tamaño ocupado por el fichero ?



Ejemplo

- *La longitud del registro será:*
 $LR = 20 + 40 + 2 + 16 + 32 + 12 = 122 \text{ bytes.}$
- *Número de registros / bloque = Factor de bloque = $\lfloor 512 / 122 \rfloor = \lfloor 4.19 \rfloor = 4$*
- *Número de bloques = $\lceil 100.000 / 4 \rceil = 25.000$ bloques.*
- *Si cada bloque ocupa 512 bytes = $25.000 * 512 = 12.20 \text{ MB}$*



Ejemplo

□ $\sigma_{\text{carnet}=2345}(r)$? No hay ordenación

Número de valores diferentes $V(\text{carnet})=100.000$, es PK

Número de registros a recuperar:

$$n_{rc} = n_R / V(\text{carnet}) = 100000/100000 = 1$$

Número de bloques a leer: No hay ordenación: Búsqueda secuencial, peor caso, leer todos los bloques : $B_R = 25.000$ bloq.

□ $\sigma_{\text{codigo_carrera}=2}(r)$? No hay ordenación

Número de valores diferentes $V(\text{carnet})=10$, no es PK

Número de registros a recuperar:

$$n_{rc} = n_R / V(\text{código_carrera}) = 100000/10 = 10.000$$

Número de bloques a leer: No hay ordenación: Búsqueda secuencial, peor caso, leer todos los bloques : $B_R = 25.000$ bloq.



Ejemplo

□ $\sigma_{\text{carnet}=2345}(r)$? Hay ordenación

Número de valores diferentes $V(\text{carnet})=100.000$, es PK

Número de registros a recuperar:

$$n_{rc} = n_R / V(\text{carnet}) = 100000/100000 = 1$$

Número de bloques a leer: hay ordenación: Búsqueda binaria, $\lceil \log_2 (b_R) \rceil + \lceil n_{rc} / f_R \rceil - 1 = \lceil \log_2 (25000) \rceil + \lceil 1 / 4 \rceil - 1 = 15 \text{ bl.}$

$\sigma_{\text{codigo_carrera}=2}(r)$? Hay ordenación

Número de valores diferentes $V(\text{carnet})=10$, no es PK

Número de registros a recuperar:

$$n_{rc} = n_R / V(\text{codigo_carrera}) = 100000/10 = 10.000$$

Número de bloques a leer: hay ordenación: Búsqueda binaria, $\lceil \log_2 (b_R) \rceil + \lceil n_{rc} / f_R \rceil - 1 = \lceil \log_2 (25000) \rceil + \lceil 10.000 / 4 \rceil - 1 = 2514 \text{ bloq.}$