



UA

# *Unidad 1: Planificación del Almacenamiento e Indexación*

*Bases de Datos Avanzadas, Sesión 1:  
Planificación del Almacenamiento de  
Registros*

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



# INDICE

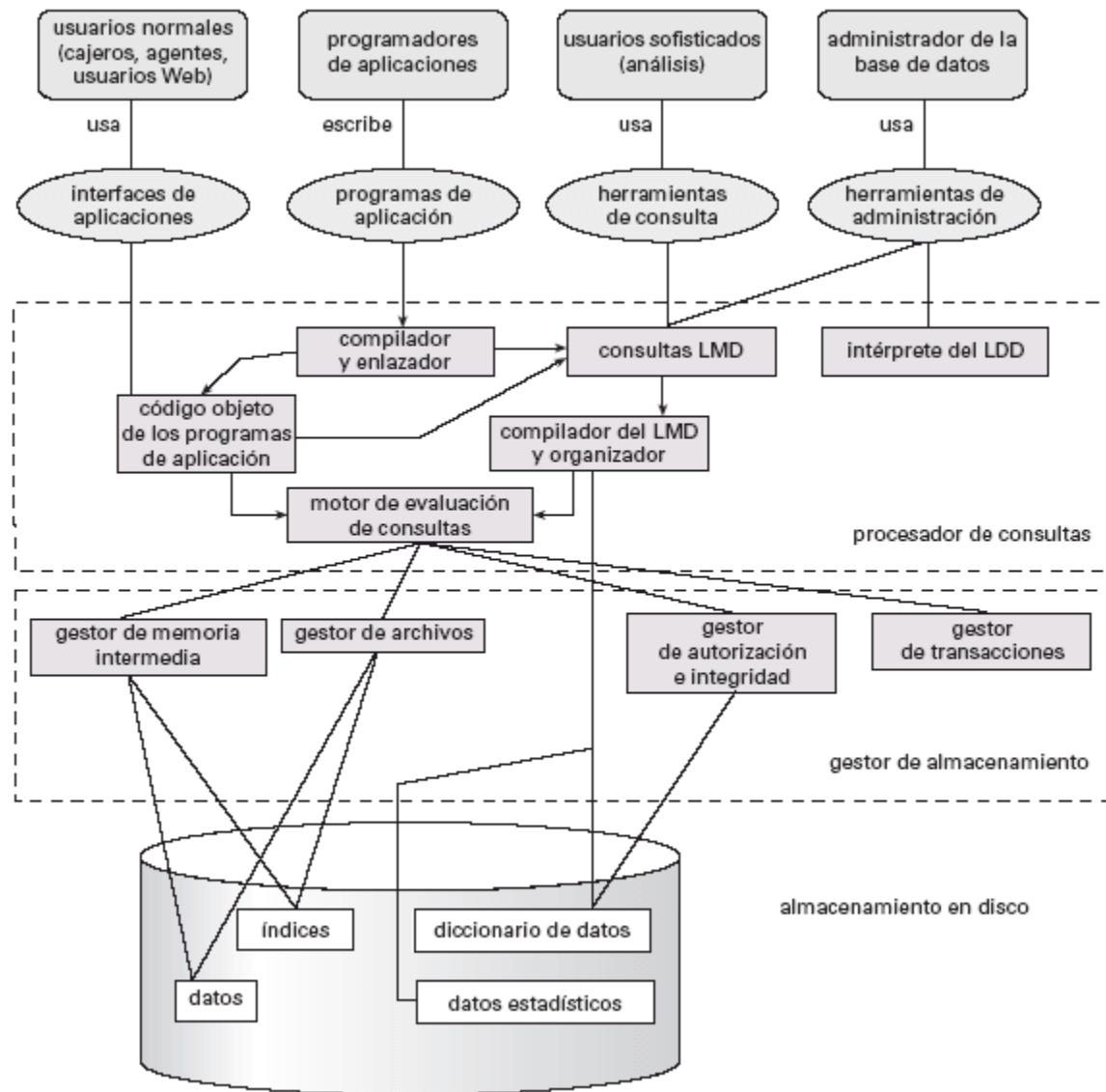
- *Estructura de un Gestor de Bases de Datos*
- *Organización de los archivos.*
- *Estimación del tamaño de un archivo*
- *Secuenciales o Cluster Index*
- *Arboles B<sup>+</sup>*
- *Hash*
- *Agrupaciones*
- *Estimación de número de registros y coste*
- *Ejemplo*

Referencias: *Silberschatz 4<sup>a</sup> Ed. Pp 249 - 315*  
*Elmasri, 3<sup>a</sup> Ed. Pp 105 - 181*



UA

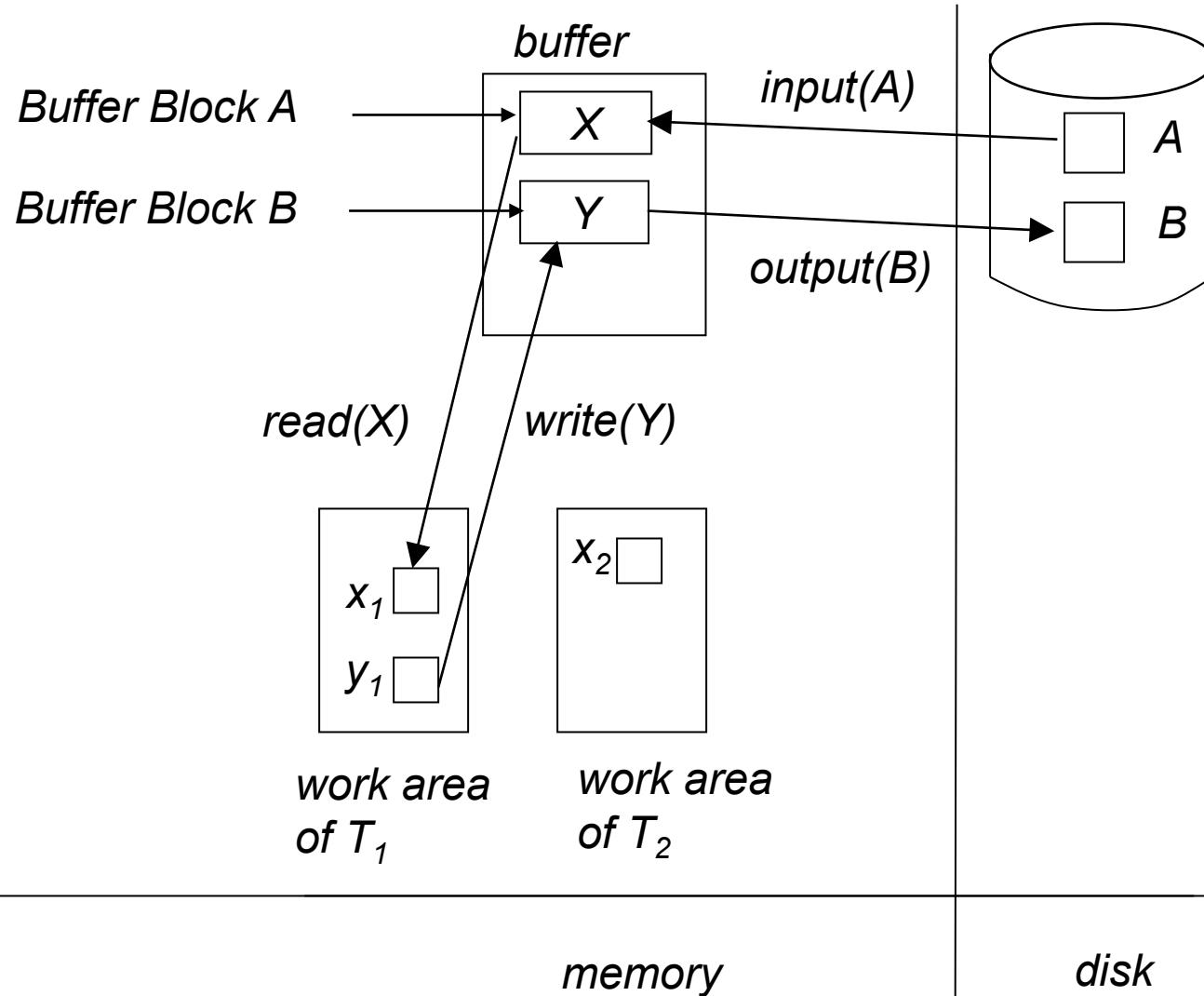
# Estructura Gestor Bases de Datos





UA

# Estructura Gestor Bases de Datos

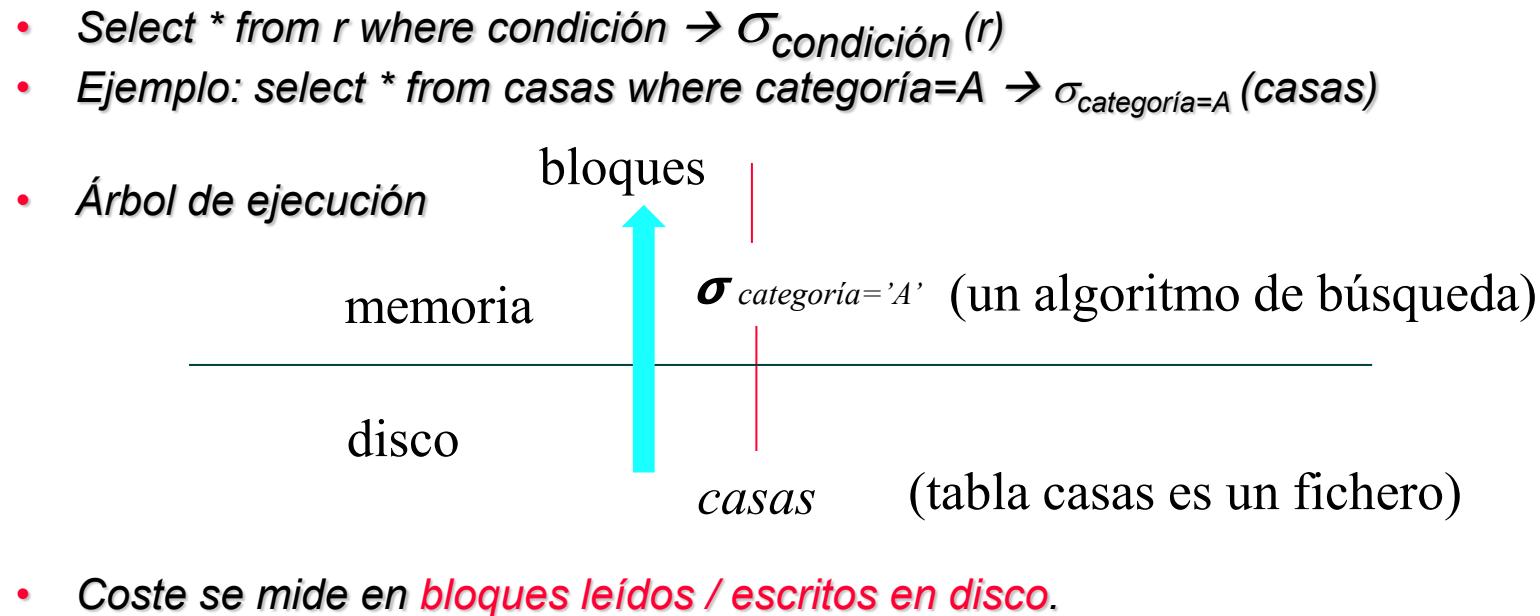




UA

# Objetivo de esta Unidad

- Saber **cuento cuesta** buscar en un archivo los datos que cumplen una condición determinada:



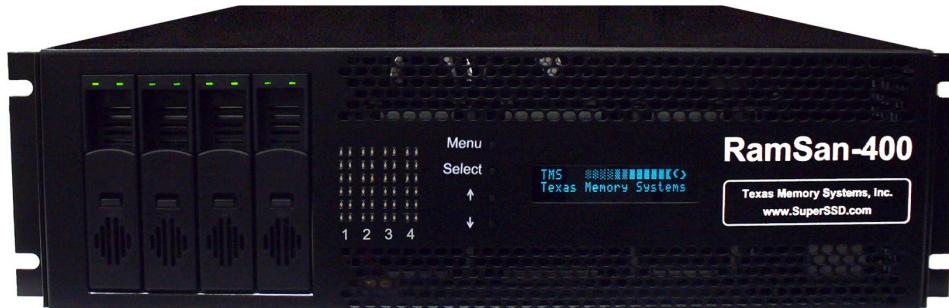
- Estimar cuantos datos se recuperarán con la consulta anterior
  - Número de registros a recuperar →  $n_{rc}$
  - Información estadística  $V(\text{campo})$  → número de valores diferentes



# Organización de los archivos



*Vista lógico ⇒ archivos son secuencias de registros que se deberían de corresponder con bloques de disco.*

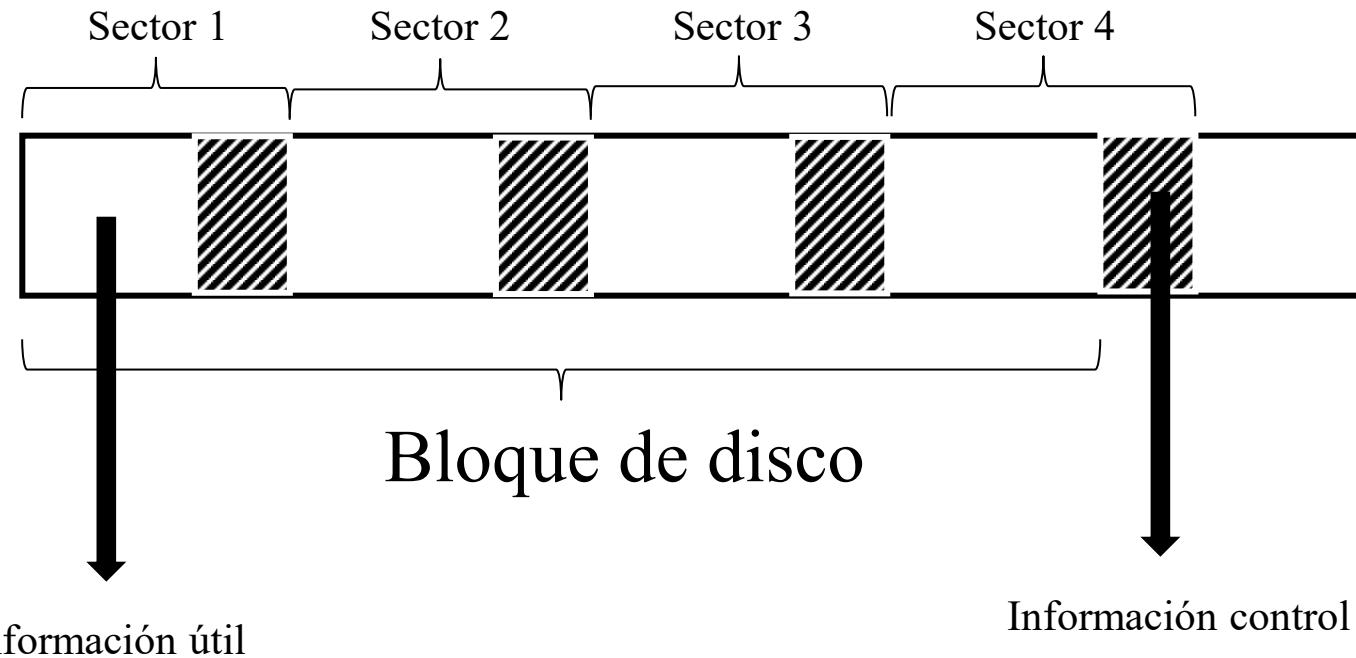




UA

# Acceso a los datos: bloques de disco

- ❑ Operación I/O  $\Rightarrow$  dirección disco  $\Rightarrow$  numero bloque
- ❑ Bloque  $\Rightarrow$  secuencia continua de sectores de una pista
- ❑ Datos se transfieren en bloques
- ❑ Ejemplo: SO 4 sectores de disco / bloque



Información útil

Información control



UA

# Organización de los archivos

- *Regla ⇒ Un registro en un Bloque de Datos ⇒ un acceso disco*
- *Rara vez un registro va a ocupar exactamente un bloque de disco*
- *Registros ⇒ Tamaño fijo o variable (cadenas texto, arrays, etc).*

```
Type deposito = record
    nombre_sucursal: char(22);
    numero_cuenta: char(10);
    saldo: real;
end
```

```
type Lista_cuentas = record
    nombre_sucursal: char (22);
    información_cuenta: array [1..∞] of record
        numero_cuenta: char(10);
        saldo: real;
    end
end
```



UA

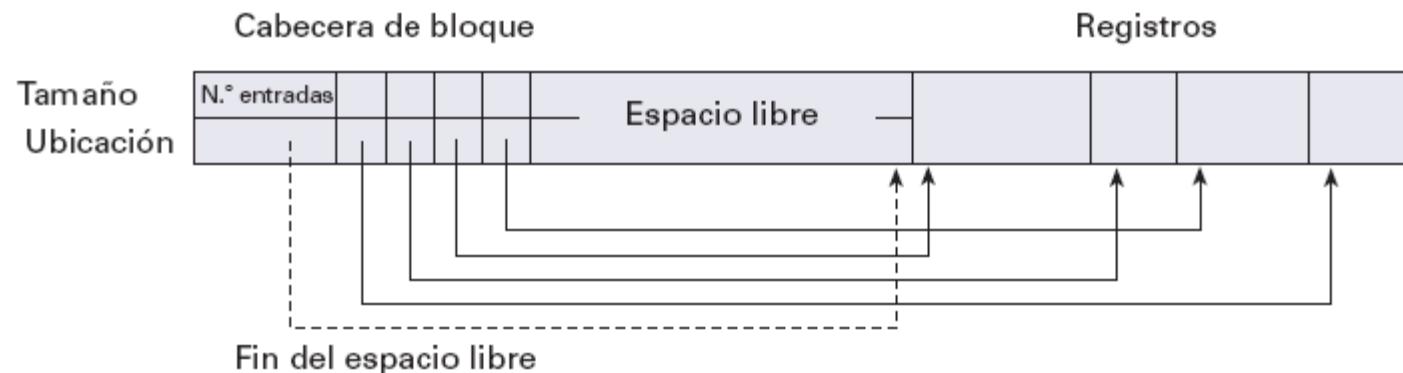
# Organización de los archivos

- *Estructura fija*

|            |       |                 |     |
|------------|-------|-----------------|-----|
| registro 0 | C-102 | Navacerrada     | 400 |
| registro 1 | C-305 | Collado Mediano | 350 |
| registro 2 | C-215 | Becerril        | 700 |
| registro 3 | C-101 | Centro          | 500 |
| registro 4 | C-222 | Moralzarzal     | 700 |
| registro 5 | C-201 | Navacerrada     | 900 |
| registro 6 | C-217 | Galapagar       | 750 |
| registro 7 | C-110 | Centro          | 600 |
| registro 8 | C-218 | Navacerrada     | 700 |

|   |                 |       |     |       |     |       |     |
|---|-----------------|-------|-----|-------|-----|-------|-----|
| 0 | Navacerrada     | C-102 | 400 | C-201 | 900 | C-218 | 700 |
| 1 | Collado Mediano | C-305 | 350 | ⊥     | ⊥   | ⊥     | ⊥   |
| 2 | Becerril        | C-215 | 700 | ⊥     | ⊥   | ⊥     | ⊥   |
| 3 | Centro          | C-101 | 500 | C-110 | 600 | ⊥     | ⊥   |
| 4 | Moralzarzal     | C-222 | 700 | ⊥     | ⊥   | ⊥     | ⊥   |
| 5 | Galapagar       | C-217 | 750 | ⊥     | ⊥   | ⊥     | ⊥   |

- *Estructura página de ranuras*





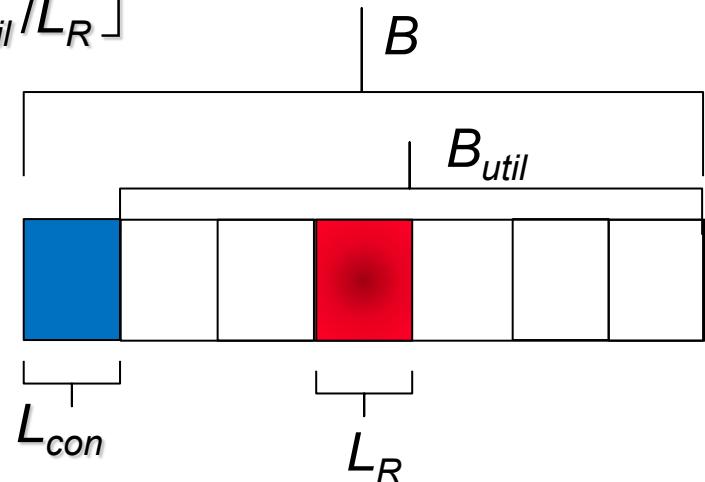
UA

# Estimación tamaño de un archivo

- Secuencia lógica de bloques que se almacenan en disco.
- $B \rightarrow$  tamaño total del bloque (bytes)
- $B_{util} \rightarrow$  Tamaño útil del bloque para almacenar registros (bytes)
- $n_R \rightarrow$  número de registros del archivo
- $L_R = \sum L_{Ci} \rightarrow$  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)  
$$f_R = \lfloor B_{util} / L_R \rfloor$$
- $b_R \rightarrow$  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



$$B_{util} = B - L_{con} \text{ se puede aplicar un porcentaje de llenado (FILL FACTOR)}$$



# *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<sup>+</sup>*
- *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 ⇒ útil en ciertos algoritmos → Búsqueda binaria
- Difícil mantener el orden físico cuando se insertan o borran registros.
- Ocupa:

$$f_R = \lfloor B_{util} / L_R \rfloor$$

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

$\xleftarrow{\hspace{1cm}} L_R \xrightarrow{\hspace{1cm}}$

|       |                 |     |  |   |
|-------|-----------------|-----|--|---|
| 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.

|       |                 |     |  |
|-------|-----------------|-----|--|
| 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 |  |



## Problema:

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

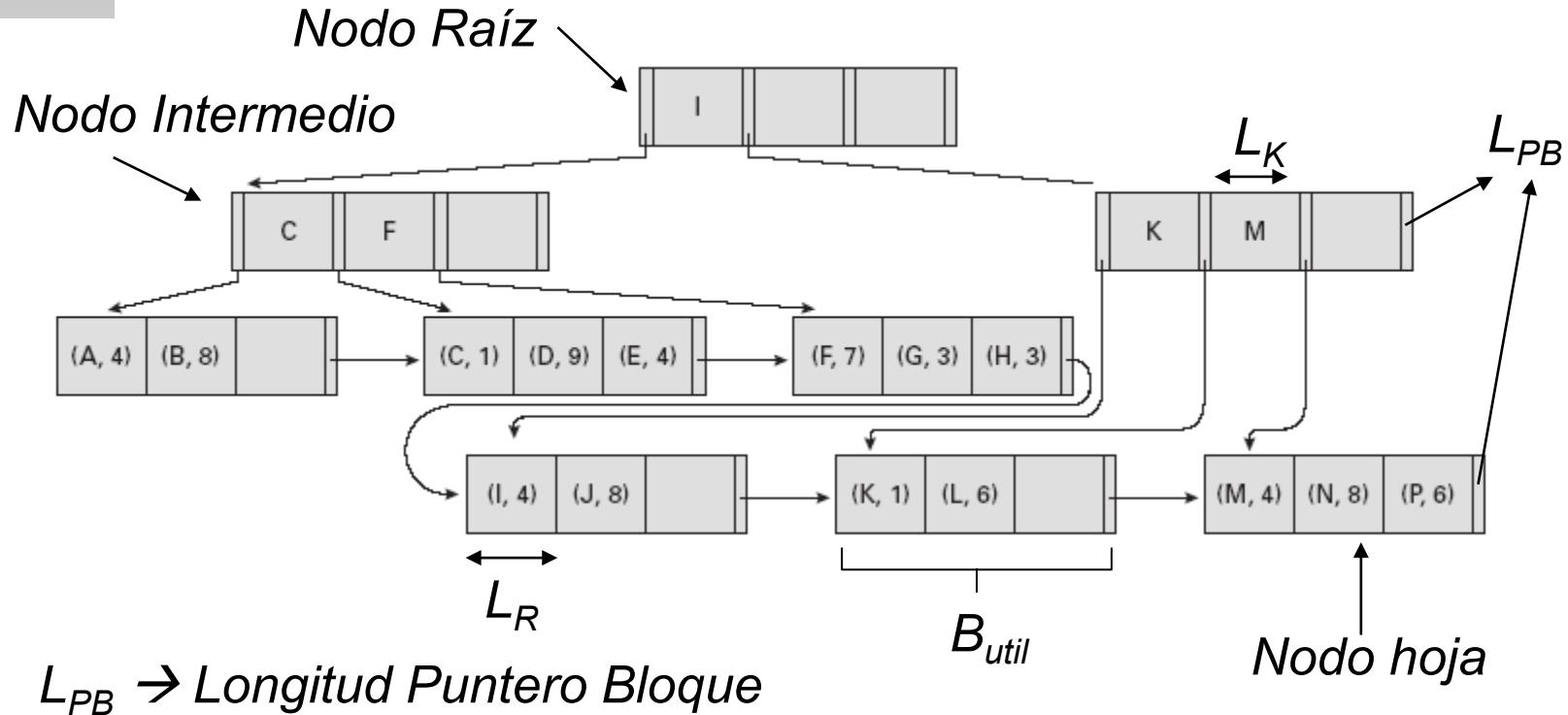
|       |         |     |  |
|-------|---------|-----|--|
| C-888 | Leganés | 800 |  |
|-------|---------|-----|--|



# Organización de Archivos con Árbol $B^+$

UA

- 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. (1 nodo = 1 bloque)





UA

# Organización de Archivos con Árbol B<sup>+</sup>

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

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

## □ 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

UA

- Es una técnica basada en la utilización de funciones de asociación.
- Sea  $K$  el conjunto de los valores de clave de búsqueda.
- Se  $N_i$  una dirección de un cajón (Bucket).
- Una función  $h$  es una función de  $K$  a  $N_i$  tal que:

$$h(k)=N_i$$

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



# Funciones de Asociación

UA



## Distribución Uniforme:

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



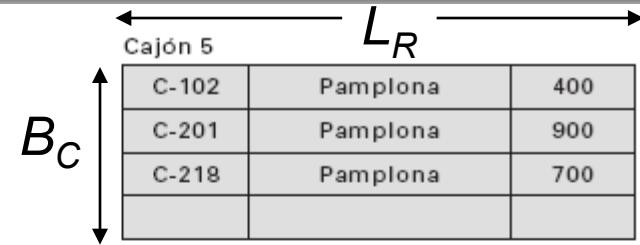
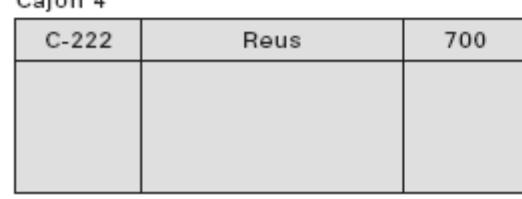
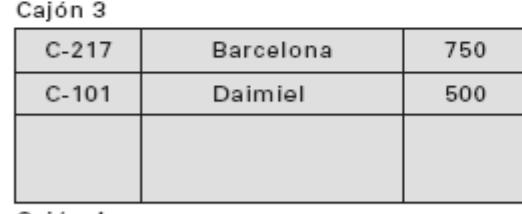
## Distribución Aleatoria:

- *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.*



UA

# Organización de archivos por asociación



$n_C$

Si equiprobabilidad:

$$n_C = n_R/N$$

$n_C$  = nº registros /cajón

$$f_{rc} = \lfloor B_{\text{util}}/L_R \rfloor$$

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

Número bloques:  $N * B_c$



UA

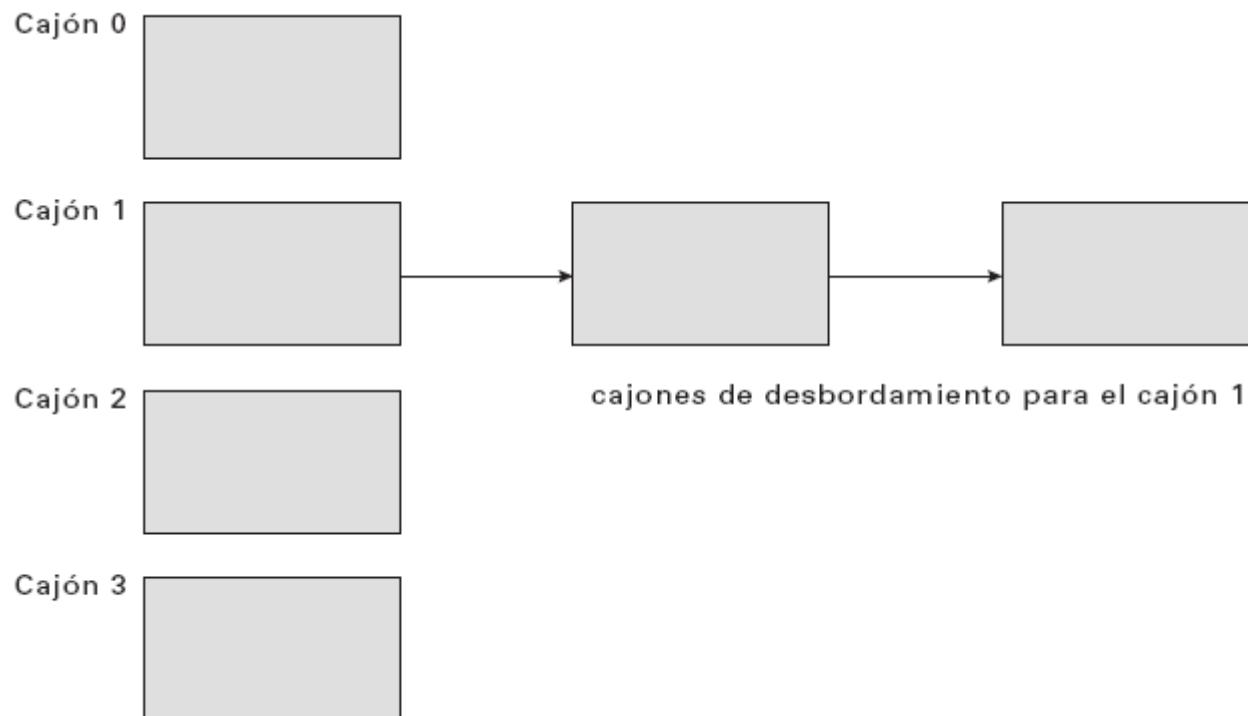
# Gestión del desbordamiento de cajones

- *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*



UA

# 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 ⇒ puede recuperar datos de varias relaciones en una lectura ⇒ mejora la ejecución de ciertas consultas.
- Sistema inteligente de bases de datos ⇒ capaz de deducir posibles mezclas de datos de forma automática.
- Mezcla de tuplas de distintas relaciones ⇒ punteros de situación ⇒ permitiendo acceder a las tuplas de las relaciones de forma independiente cuando es necesario.



# *Organización de los registros en los archivos: Agrupaciones*

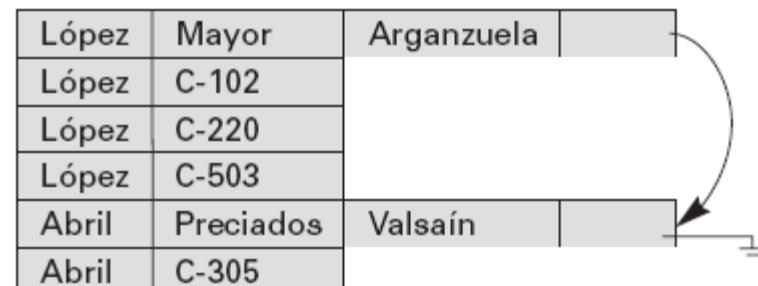
| nombre-cliente | número-cuenta |
|----------------|---------------|
| López          | C-102         |
| López          | C-220         |
| López          | C-503         |
| Abril          | C-305         |

| 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     |            |





# 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  $\sigma_{A=1}(R)$  ? (Igualdad)
- 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.



# Estimación registros/bloques a recuperar del archivo de datos

- $\sigma_{A>=1}(R)$  ó  $\sigma_{A<=1}(R)$  ó  $\sigma_{A>1 \wedge A<3}(R)$  ? (Comparación)  
 $n_{rc} = n_v * n_r / V(A, R)$  ( $Nv \rightarrow$  Número de valores diferentes que cumplen la condición)
- $\sigma_{A<>1}(R)$  ? (Distinto)  $\rightarrow n_{rc} = n_r - n_r(\sigma_{A=1}(R))$
- $\sigma_{A=1 \wedge B=3}(R)$  ?  $\rightarrow n_{rc} = n_r / (V(A, R) * V(B, R))$
- $\sigma_{A=1 \vee B=3}(R)$  ?  $\rightarrow n_{rc} = n_r / V(A, R) + n_r / V(B, R)$  (Aproximada)
- Coste?  $\rightarrow$  Depende si hay ordenación o no.