Unidad 1: Planificación del Almacenamiento e Indexación

Bases de Datos Avanzadas, Sesión 1:

Dispositivos y Estructuras de Almacenamiento

Iván González Diego

Dept. Ciencias de la Computación

Universidad de Alcalá

INDICE

- *Introducción.*
- *Visión Medios de Almacenamiento*
- *Gestor de Memoria Intermedia*
- *Estructuras de Almacenamiento:*
  - *Longitud Fija*
  - *Longitud Variable*

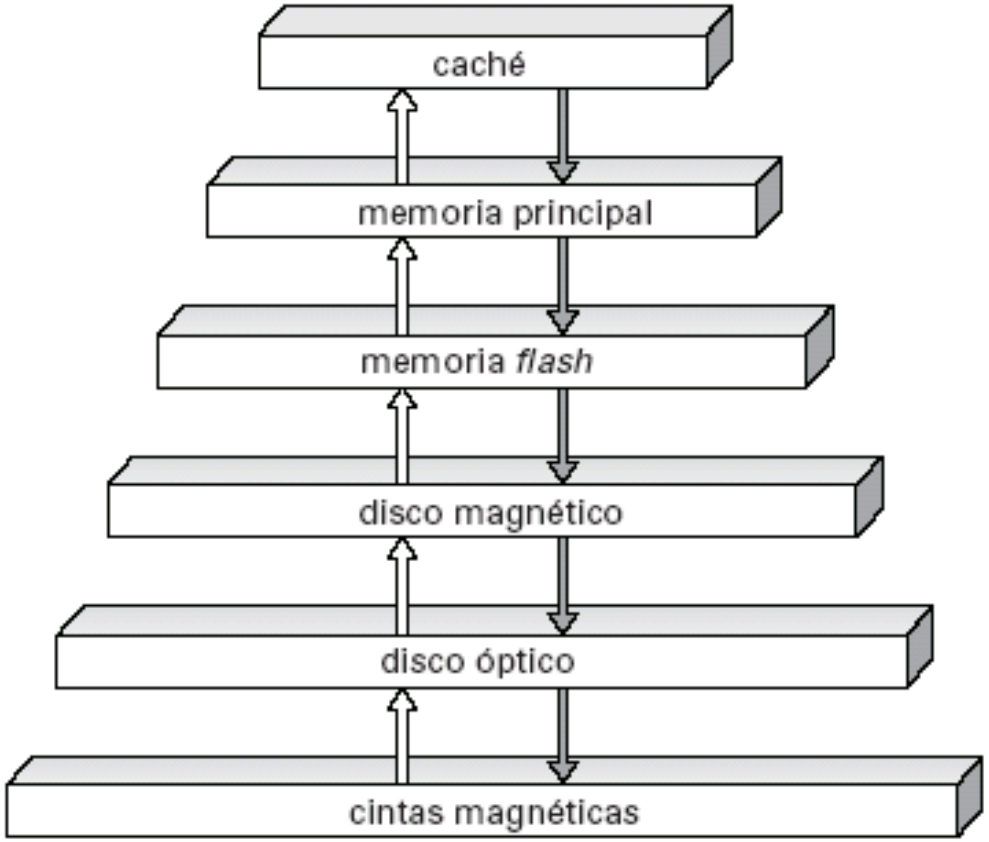
Referencias: Silberschatz 4ª Ed. Pp 249 - 315
Elmasri, 3ª Ed. Pp 105 - 181

Introducción

- *En este tema se va a tratar sobre el almacenamiento de los datos de usuario y la planificación del mismo según el sistema gestor de base de datos.*

- *De forma detallada se entrará en los modos de almacenamiento de los ficheros de datos y el funcionamiento de los índices de una base de datos.*

Visión general de los medios de almacenamiento

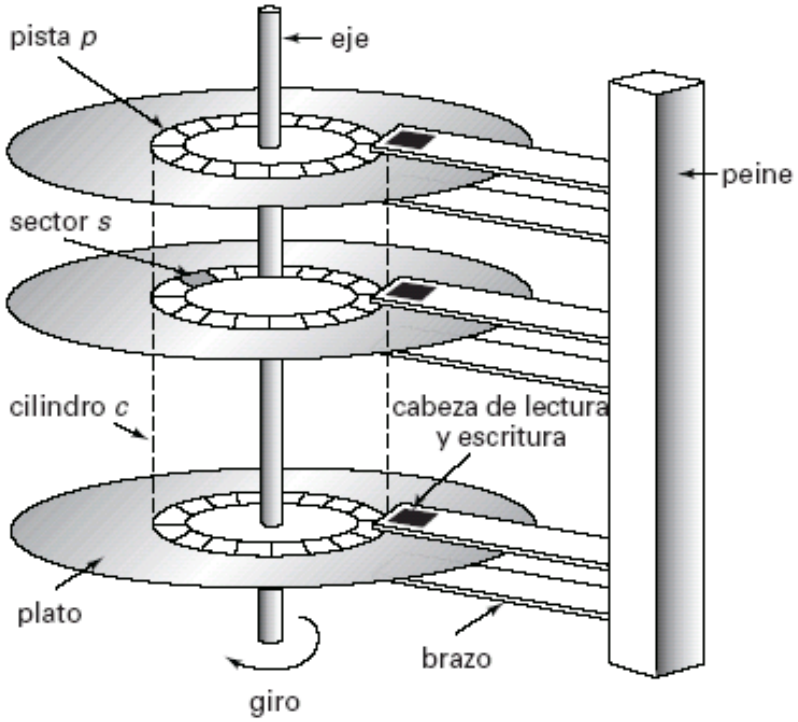


Visión general de los medios de almacenamiento

- *Distintos tipos de memoria:*
  - *Caché: La más rápida junto con los registros. Pequeño tamaño y control hardware.*
  - *Memoria principal: medio de almacenamiento donde se ejecutan los programas. No suele ser lo suficientemente grande para guardar toda una base de datos.*
  - *Memoria flash: Memoria de lectura y escritura bastante lenta.*
  - *Discos magnéticos: Principal medio de almacenamiento. Hace las veces de disco swaping con la memoria principal.*
  - *Almacenamiento Optico:*
    - *CD-ROM: ⇒ sirven como copia de respaldo,*
    - *DVD: (4-17 GB)*
  - *Almacenamiento en cinta: cintas de gran capacidad (80 GB – 1 TB). Copias de respaldo. DLT*

UA

Discos de almacenamiento

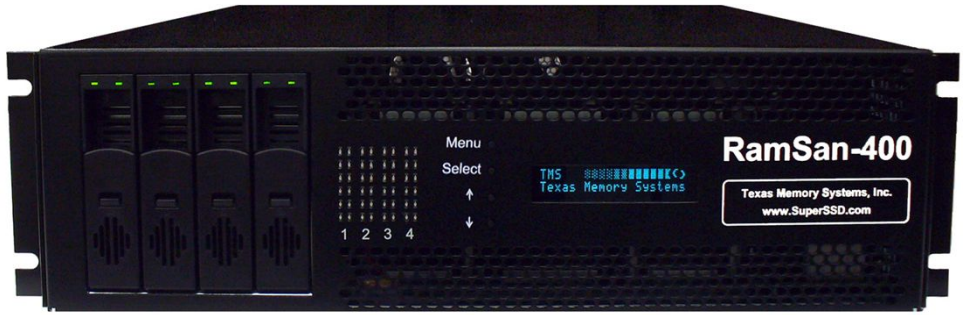


Discos de almacenamiento

- *Tiempo de acceso: tiempo transcurrido entre la solicitud de lectura y el comienzo de la transferencia de datos*
- *Tiempo medio de Búsqueda : tiempo para ubicar el brazo en una pista en media.*
- *Tiempo de latencia rotacional: tiempo de espera una vez movido el brazo para que el sector pase por debajo de la cabeza*
- *Tiempo medio de latencia: en media la mitad de rotación del disco*
- *Velocidad de transferencia de datos: velocidad a la que se pueden recuperar o guardar datos.*
- *Tiempo medio entre fallos: medida de fiabilidad.*

UA

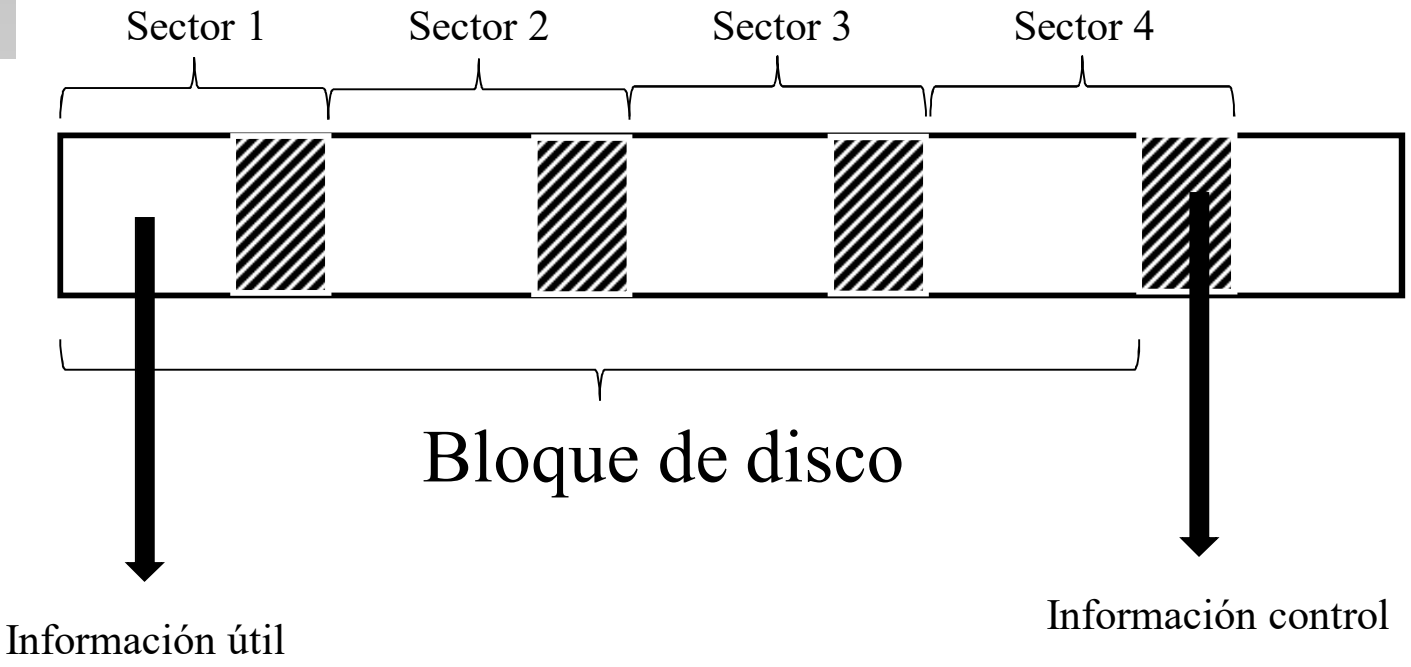
Discos de Estado Sólido (SSD)



SD basado en DDR SDRAM. Máx. 128 GB y 3072 MB/s.

Acceso a los datos: bloques de disco

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

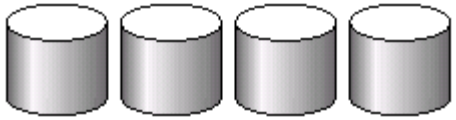


Sistemas de acceso a disco: Sistemas RAID

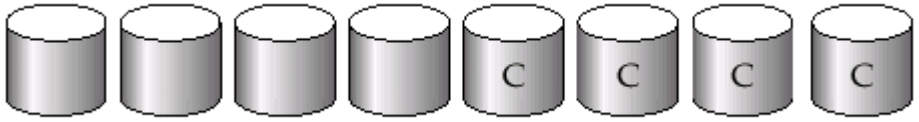
- *RAID: Redundant Array of Independent Disks*
- *Compuestos de discos pequeños y de bajo coste:*
  - *Gran rendimiento*
  - *Gran fiabilidad al montarlos en este tipo de sistemas.*
- *Múltiples modos de funcionamiento:*
  - *Con redundancia.*
  - *Paridad de datos*
  - *Paralelismo, etc.*
- *Distribución de datos a nivel de bloque (1 bloque en cada disco) o a nivel de bit (1 bit en cada disco)*
- *Son utilizados normalmente en cualquier sistema de almacenamiento masivo de datos, utilizándose de forma normal el RAID 1 o el 5.*

Niveles RAID

- *0: Disposición de disco con distribución a nivel de bloque y sin redundancia*
  - *Aplicaciones de Alto Rendimiento donde la pérdida de datos no es crítica*



- *1: Creación de imágenes de disco con distribución de bloque*



- *5: Distribución datos nivel bloque, pero distribuyendo la paridad entre todos los discos, para cada bloque un disco guarda paridad, y el resto datos.*

Acceso al almacenamiento: Gestor de memoria intermedia

- *Son parecidos a los gestores de memoria virtual aunque más complejos.*
- *Se encargan de manipular los bloques de la memoria al disco o del disco a la memoria*
- *Objetivo ⇒ minimización de los accesos al disco.*
- *Los sistemas operativos utilizan esquemas de gestión de memoria estáticos, en base a previsiones estadísticas.*
- *Los SBGD por el contrario son capaces de planificar los accesos a datos que realizarán, el orden y la estrategia a seguir ⇒ Gestión de memoria mucho más precisa que un S.O.*

Acceso al almacenamiento: Gestor de memoria intermedia

- *Los esquemas de gestión de la memoria intermedia son:*
  - *Estrategias de sustitución:*
    - *MRU: Más Recientemente Utilizado fuera.*
    - *LRU: Menos Recientemente Utilizado fuera.*
  - *Bloques clavados: Limita el número de veces que se puede escribir un bloque en disco.*
  - *Salida forzada de bloques: Se obliga a la escritura de un bloque en disco, y se elimina de la memoria principal.*

Acceso al almacenamiento: Gestor de memoria intermedia

- *Ejemplo: Reunión natural entre prestatario y cliente.*

```
For each tupla p de prestatario do
    For each tupla c de cliente do
      if p[nombre-cliente]=c[nombre-cliente] then
      begin
        sea x una tupla de reunión natural
        x[nombre_cliente]:=p[nombre_cliente]
        x[numero_prestamo]:=p[numero_prestamo]
        x[calle_cliente]:=p[calle_cliente]
        x[ciudad_cliente]:=p[ciudad_cliente]
      End
    End
  End
```

- *P: LRU o extracción inmediata*
- *C: MRU ⇒ elimina el último utilizado. No se utilizará hasta la próxima iteración del bucle.*

Organización de los archivos

- *Vista lógico ⇒ 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*
- *Regla ⇒ Un registro en un Bloque de Datos ⇒ un acceso disco*
- *Registros ⇒ Tamaño variable (cadenas texto, arrays, etc).*
- *En bases de datos se manejan dos supuestos:*
  - *Registros de tamaño fijo ⇒ gran velocidad de acceso.*
  - *Registros de tamaño variable ⇒ gran eficiencia en el espacio*

Organización de los archivos: Registros de longitud fija

- *Ejemplo:*

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

- *Suponiendo que un real ocupa 8 bytes, los caracteres son ASCII de 1 byte, el registro tendrá 40 bytes de longitud.*

Organización de los archivos: Registros de longitud 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

- *Problemas de la estructura:*
  - *Dificultad en los borrados ⇒ Creación de huecos, dificultad de contabilización de los espacios en esquemas simples.*
  - *Dificultad en almacenamiento ⇒ Puede haber registros que estén almacenados en 2 bloques ⇒ dos accesos a disco para leer un registro.*

Organización de los archivos: Registros de longitud fija

UA

- *Estrategias de actualización de las tablas:*
  - *Desplazamiento en borrado.*
    - *Desplazamiento de los siguientes registros: Gran sobrecarga de disco. No se utiliza.*

registro 0	C-102	Navacerrada	400
registro 1	C-305	Collado Mediano	350
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

    - *Desplazamiento del último registro: Menos sobrecarga. Tampoco se suele utilizar*

registro 0	C-102	Navacerrada	400
registro 1	C-305	Collado Mediano	350
registro 8	C-218	Navacerrada	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

Organización de los archivos: Registros de longitud fija

- *Seguimiento del borrado.*
  - *Se basa en que es más probable que se inserten datos en vez de borrarlos ⇒ los huecos dejados al borrar se pueden utilizar posteriormente en las nuevas inserciones*
  - *Estructura de cabecera con seguimiento del primer registro libre y lista enlazada a partir de este para acceder al resto de registros borrados. Se denomina lista libre*
  - *Cuando se añade un nuevo registro ⇒ se pone en el primer hueco libre y se actualiza la cabecera.*

Organización de los archivos: Registros de longitud variable

- *Se deben a:*
  - *Almacenamiento de varios tipos de registros sobre el mismo archivo*
  - *Tipos de registros que permiten longitudes variables de sus campos*
  - *Tipos de registros con campos repetidos*

- *Ejemplo:*

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

Organización de los archivos: Registros de longitud variable

- *Representación de cadenas de bytes:*
  - *Método muy sencillo.*
  - *Existe una marca de "fin de registro".*

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	⊥				

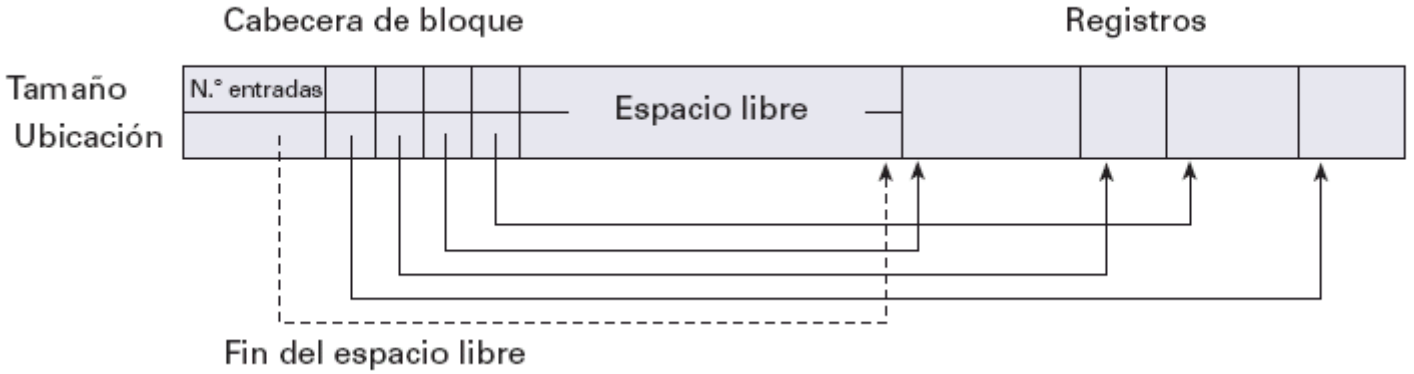
- *Esquema Alternativo ⇒ cabecera con longitud del registro*

Organización de los archivos: Registros de longitud variable

- *Problemas*
  - *Problemas para reutilizar el espacio borrado, al tener distintas longitudes ⇒ Necesidad de compactación de datos.*
  - *No deja espacio para el aumento del tamaño de los registros ya incluidos.*
    - *Si un registro aumenta ⇒ desplazarlo*
    - *Provoca el mismo efecto que un borrado.*
  - *No se suele utilizar.*
- *Solución:*
  - *Estructura de páginas con ranuras.*

Organización de los archivos: Registros de longitud variable

- *Estructura de páginas con ranuras:*
  - *Número de elementos del registro de cabecera*
  - *Final del espacio vacío del bloque*
  - *Array con la ubicación y el tamaño de cada registro.*
- *Comportamiento*
  - *Los registros reales ⇒ seguidos dentro del bloque, pero empezando desde el final*
  - *El número de entradas y el array de "punteros" se sitúan al principio del bloque.*
  - *El espacio libre se sitúa en el medio de los Registros-cabecera*



Organización de los archivos: Registros de longitud variable

- *Comportamiento (continuación)*
  - *Si se añade un registro:*
    - *añadir una entrada al array*
    - *El registro se inserta al final del espacio libre.*
    - *Posteriormente se actualiza el espacio libre.*
  - *Si se borra un registro:*
    - *Su entrada se le da un tamaño de –1*
    - *Se desplazan los que había antes que él, actualizando el resto de la cabecera y el nuevo punto del espacio libre.*

- *Dado que la escritura se realiza a través de bloques (2 Kb-4 Kb) ⇒ Coste de mover la información dentro del bloque no es alto ⇒ se reduce la fragmentación del espacio utilizado.*

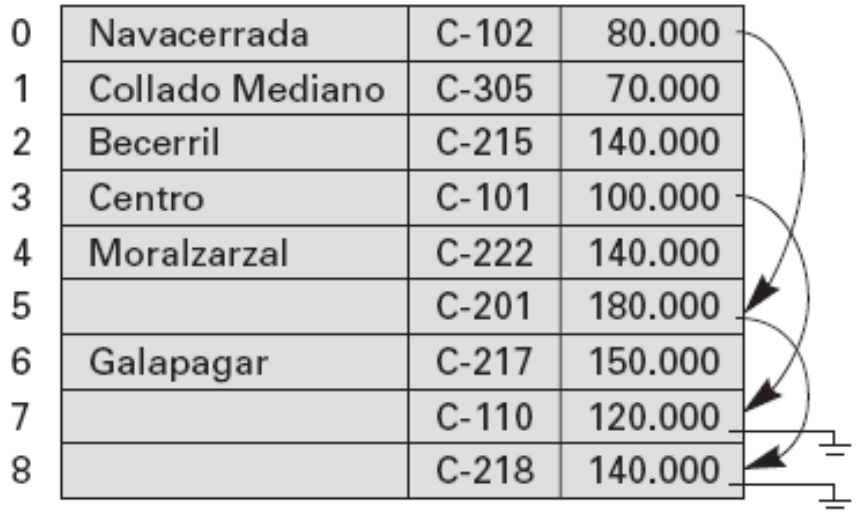
Organización de los archivos: Registros de longitud variable

- *Representación de longitud fija*
  - *Mediante **espacio reservado**.*
    - *Si nunca se supera la longitud máxima asignada, se puede utilizar.*
    - *Se utiliza una marca para representar la falta de información.*
    - *Es poco útil con información de distinto tamaño*
    - *Pero si la información tiene un tamaño cercano al máximo, merece la pena implementar este tipo de esquemas.*

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	⊥	⊥	⊥	⊥

Organización de los archivos: Registros de longitud variable

- *Representación de longitud fija con punteros*
  - *Similar al utilizado en el seguimiento del espacio libre en los esquemas de registros de longitud fija*
  - *Mantiene la información de los registros ocupados por la misma entidad superior*



Organización de los archivos: Registros de longitud variable

- *Se suele utilizar una variante de dos tablas:*
  - *Una tabla de **bloque ancla** ⇒ mantiene la información del primer registro.*
  - ***Bloque de desbordamiento** ⇒ mantiene la información de los siguientes registros.*