

Tema 4

El Nivel Interno

Apartado 1

Introducción

Contenidos

- Introducción
- Método de Acceso a la BD almacenada
- Representación de la BD en el nivel interno

Contenidos

- Introducción
- Método de Acceso a la BD almacenada
- Representación de la BD en el nivel interno

Introducción

Importancia de la organización de los datos

Una BD sirve para ...

almacenar de forma permanente grandes cantidades de datos.

Con el propósito principal de ...

gestionar de forma eficiente los datos y su almacenamiento para poderlos recuperar/consultar cuando sea necesario.

Esto tiene sus consecuencias tanto en la organización lógica de los datos, como en su organización física.



Introducción

Nivel Interno

Nivel Interno

- Expresa en última instancia, las operaciones sobre los datos (creación, alteración y recuperación) en términos de actuación sobre unidades mínimas de almacenamiento denominadas páginas o bloques de base de datos.
- Provee al administrador de mecanismos para optimizar el almacenamiento y el acceso a los datos.
- Se encuentra implementado en el SGBD.



Introducción

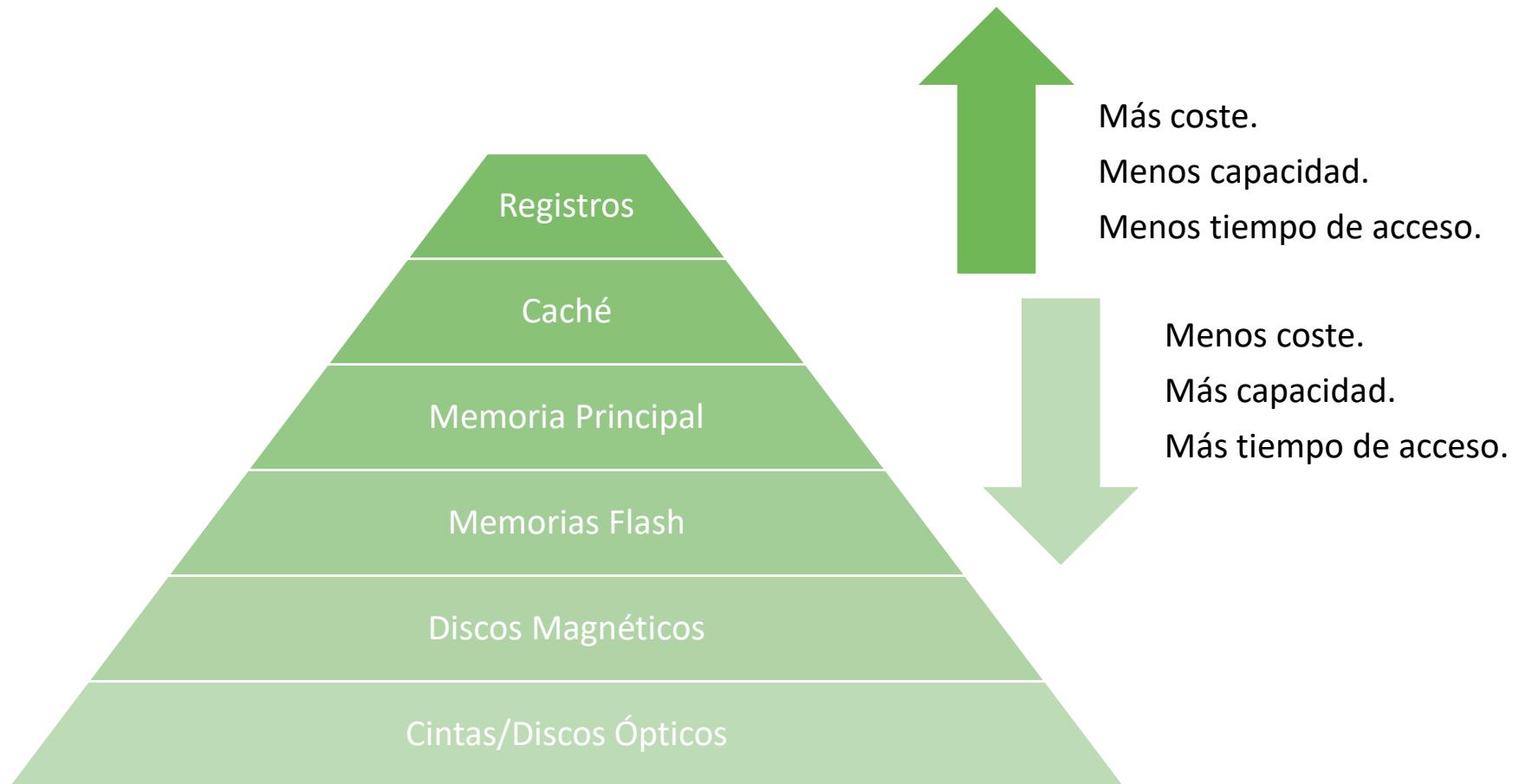
Nivel Físico

Nivel Físico

- Se encuentra implementado en el Sistema Operativo.
- Proporciona al SGBD una capa de abstracción sobre el hardware.
- Realiza el acceso a los medios de almacenamiento mediante llamadas a los servicios del sistema de archivos proporcionado por el SO.

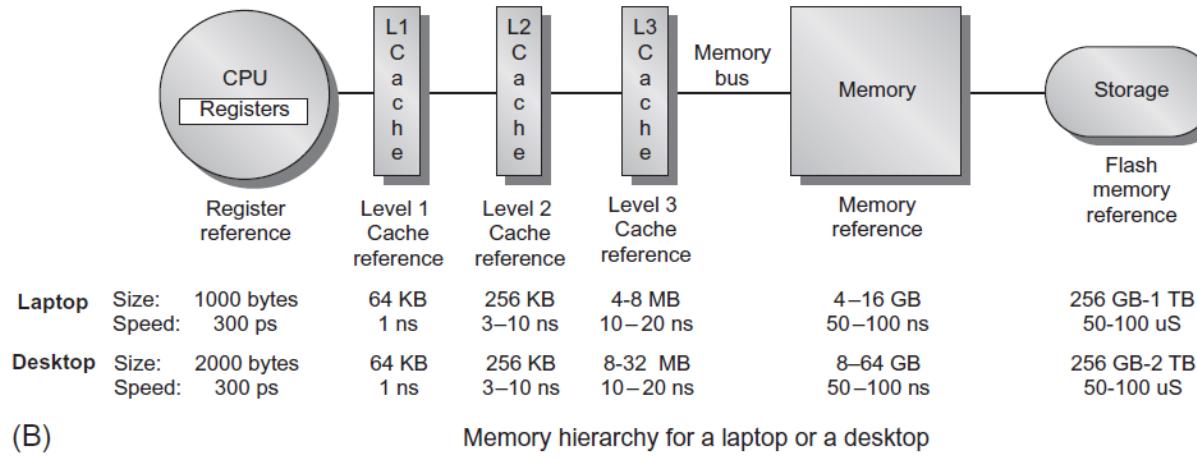
Introducción

Jerarquía de memoria



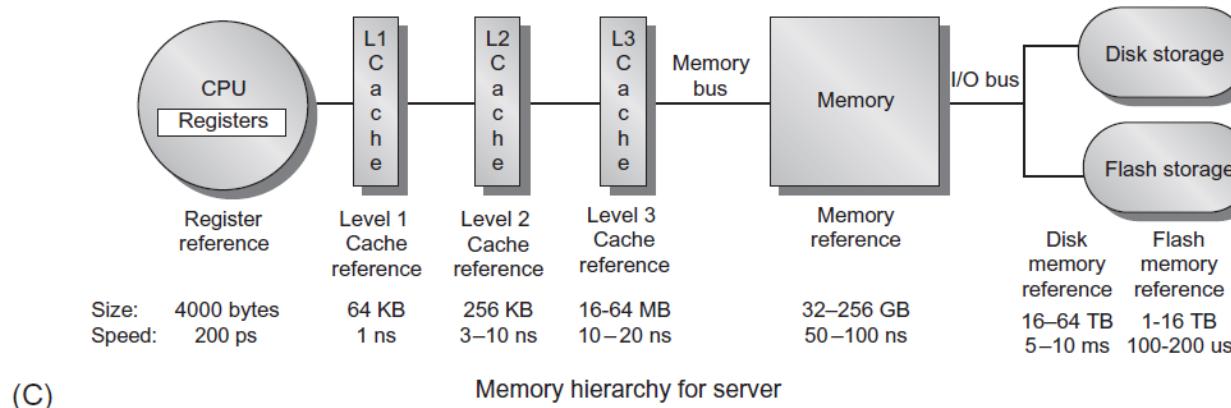
Introducción

Jerarquía de memoria



(B)

Memory hierarchy for a laptop or a desktop



(C)

Memory hierarchy for server

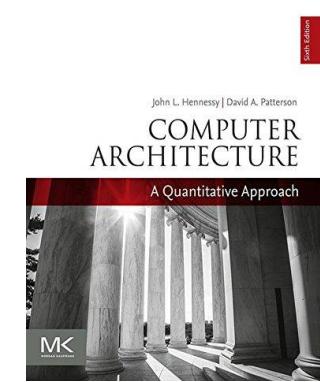


Figura tomada de John. L. Hennessy, David A. Patterson: Computer Architecture. A Quantitative Approach, 6th Edition, Morgan Kaufmann, 2017.

Introducción

Memoria principal

- Es el dispositivo de almacenamiento primario de los ordenadores.
 - Hace trabajos de caché de la porción de la BD de uso más reciente.
 - Elemento de almacenamiento intermedio que ubica de forma temporal los datos afectados por las operaciones.
- El nivel interno debe optimizar su uso:
 - Es rápida: acelera el procesamiento
 - Es cara: hay que optimizar su uso
- Es volátil, su información se pierde cuando se apaga el sistema o se cae. El nivel interno debe garantizar que la información contenida tenga un respaldo en almacenamiento secundario para que no se pierda.
- Se utilizan distintos niveles de caché para acelerar el acceso a los datos.

Introducción

Discos duros



- Dispositivo de almacenamiento más usado en BD.

- Conjunto de discos magnéticos con dos caras.
 - Cada cara tiene un conjunto de pistas concéntricas (cilindro: la misma pista de todas las caras).
 - Cada pista se divide en sectores con la misma capacidad de almacenamiento (bloque).
-
- Localización de un bloque:
 - cilindro
 - superficie de disco
 - sector

- Parámetros:

- Capacidad
- Tiempo medio de acceso
- RPM
- Velocidad sostenida de lectura/escritura.

Imágenes tomadas de <https://pixabay.com/>

Introducción

- Medidas de rendimiento:
 - Tiempo medio de acceso (ta): tiempo medio transcurrido entre una instrucción y la obtención de la información.
 - Tiempo medio de búsqueda (tb): tiempo medio de posicionamiento en pista.
 - Tiempo de latencia rotacional (tl): tiempo medio de posicionamiento en sector.

$$ta=tb+tl$$

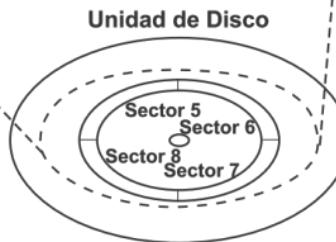
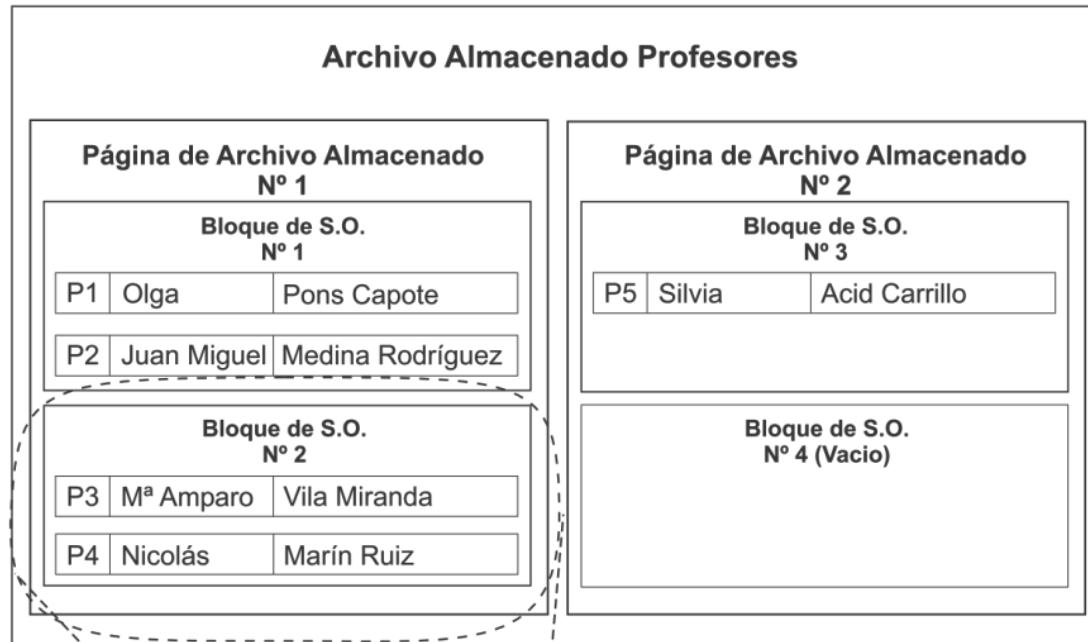
- Tiempo medio entre fallos (MTBF)



Contenidos

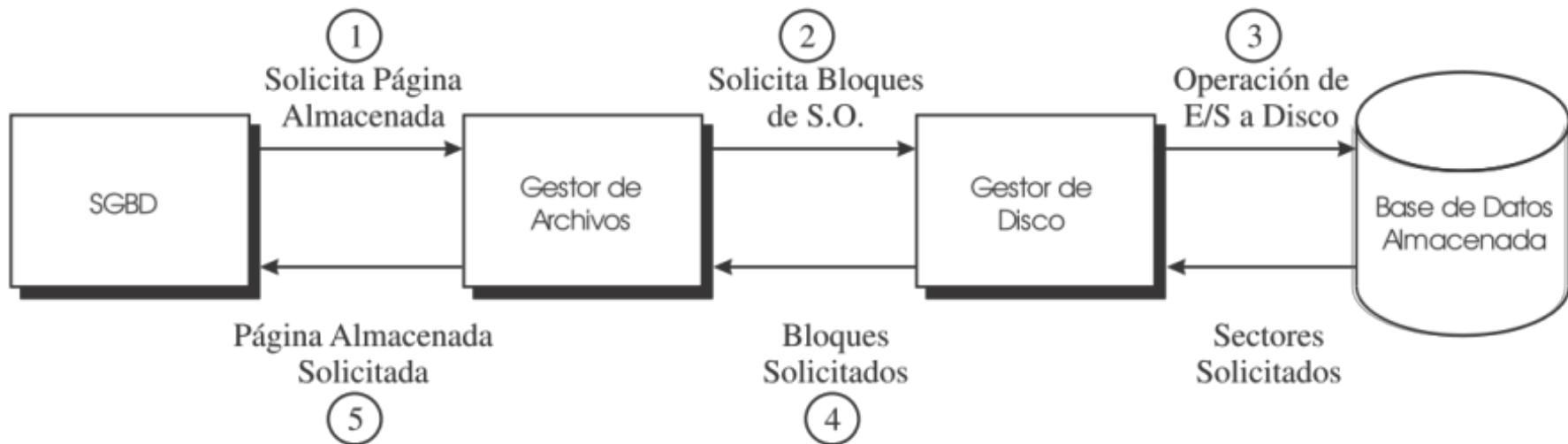
- Introducción
- Método de Acceso a la BD almacenada
- Representación de la BD en el nivel interno

Método de acceso a la BD



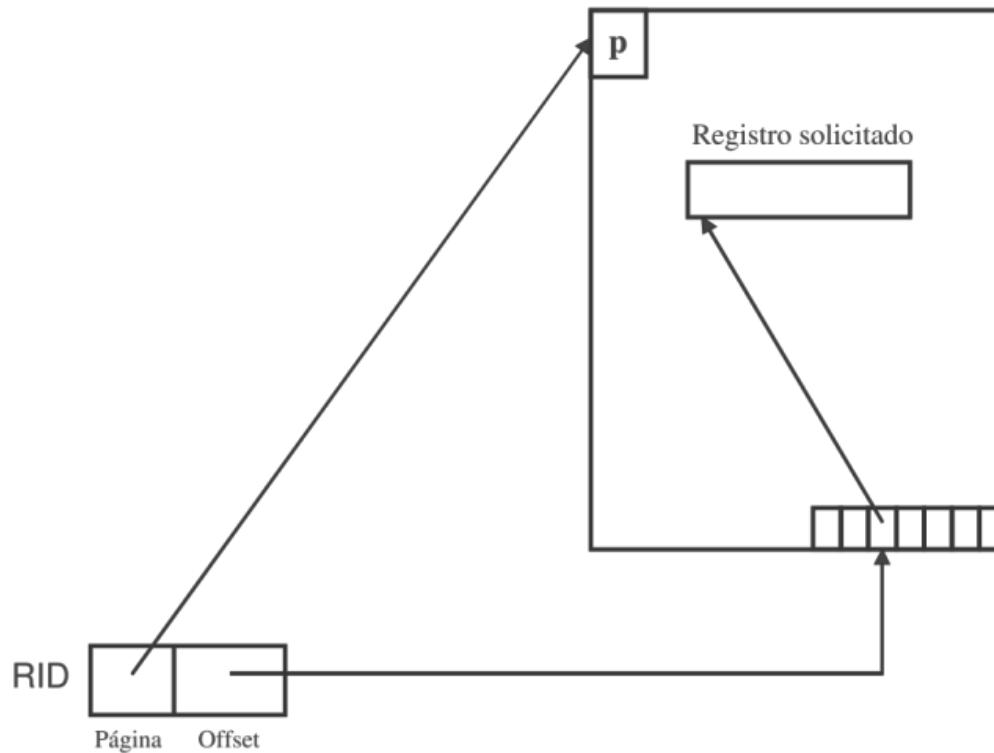
Método de acceso a la BD

¿Cómo se transforma un registro almacenado en una representación física en el almacenamiento secundario?



Método de acceso a la BD

- Para que el gestor de almacenamiento pueda localizar un registro almacenado, se utiliza el RID (Record IDentifier):



Método de acceso a la BD

- Cada registro almacenado:
 - Cabecera: Número y tipo de columnas que lo integran.
 - Datos: Contenido de las columnas.
- Las páginas o bloques de la BD deben tener un tamaño múltiplo de los bloques del sistema operativo (mínima unidad de E/S).
- Para recuperar un registro almacenado hay que determinar la página de BD que lo contiene y entonces recuperar los bloques de disco que la integran.
- Hay que organizar la estructura de almacenamiento y los métodos de acceso, de forma que se optimice la interacción con los dispositivos de almacenamiento secundario.
- **Deben minimizarse las operaciones de E/S al almacenamiento secundario.**

Método de acceso a la BD

- El Gestor de Disco del SO

- Normalmente el SGBD interactúa con la BD almacenada en el sistema de almacenamiento secundario a través del gestor de disco del SO.
- El gestor de disco organiza los datos en archivos (conjuntos de bloques) de SO.
- Una BD puede utilizar uno o varios de estos archivos para almacenar su contenido.
- También se encarga de gestionar el espacio libre en el disco.



Método de acceso a la BD

- Funciones elementales:

- Crear un nuevo archivo de sistema operativo.
- Eliminar un archivo de sistema operativo existente.
- Añadir un bloque nuevo al conjunto de bloques c.
- Eliminar el bloque b del conjunto de bloques c.
- Devolver el bloque b del conjunto de bloques c.
- Reemplazar el bloque b dentro del conjunto de bloques c.



Método de acceso a la BD

- El Gestor de Archivos del SGBD
 - Componente del SGBD que se encarga de:
 - Hacer la transformación entre:
 - Campos, registros y archivos almacenados
 - y
 - bloques y conjuntos de bloques (que pueda entender el gestor de disco).
 - Organizar los datos de manera que se minimice el tiempo de recuperación:
 - Minimizar las E/S a disco.



Método de acceso a la BD

- Funciones básicas:
 - Crear un nuevo archivo almacenado.
 - Asociar al archivo un conjunto de páginas o bloques de la BD.
 - Eliminar un archivo almacenado.
 - Recuperar el registro almacenado r del archivo almacenado a.
 - Normalmente, el SGBD proporciona el RID.
 - Sólo hay que obtener en memoria la página que contiene el registro para extraerlo.
 - Añadir un nuevo registro almacenado al archivo almacenado a. Hay que localizar la página de BD más apropiada de las pertenecientes al archivo almacenado.
 - Si no se pudiera, se solicita una nueva página.
 - Se devuelve al SGBD el RID nuevo.



Método de acceso a la BD

- Funciones básicas (continuación):

- Eliminar el registro r del archivo almacenado a.
 - Hay que recuperar la página de BD que contiene dicho registro y marcar el espacio ocupado por el registro en dicha página como disponible.
- Actualizar el registro r en el archivo almacenado a.
 - Recupera la página de la BD que contiene el registro que se desea actualizar.
 - Trata de sustituir la información. Si no puede, se intenta ubicar en otra página.



Contenidos

- Introducción
- Método de Acceso a la BD almacenada
- Representación de la BD en el nivel interno

Representación de la BD en el nivel interno

- La BD se representa de diferentes formas en los diferentes niveles de la arquitectura del SGBD.
 - Su representación en el nivel interno no tiene por qué coincidir con su representación en el nivel conceptual.
 - Cada conjunto de registros del mismo tipo no tiene por qué ser un fichero.
- El nivel interno debe traducir las estructuras del nivel conceptual a otras estructuras intermedias más cercanas al almacenamiento real de los datos (nivel físico).



Representación de la BD en el nivel interno

Agrupamiento

La BD en el nivel interno

Conjunto de páginas en las que se van ubicando los registros.

Agrupamiento Intra-Archivo

Ubicar en una página registros del mismo tipo.

Es el más frecuente.

Agrupamiento Inter-Archivo

Ubicar en una página registros de distinto tipo.

Ha de existir relación (por ejemplo entidades fuerte-débil)

Representación de la BD en el nivel interno

- Tenemos tres archivos almacenados:
Asignaturas,
Profesores e Imparte.
Con una secuencia de páginas asociada a cada archivo.
- En cada página se almacena un solo registro.
- La página 0 es el directorio de archivos almacenados.
- También tenemos el conjunto de páginas libres.

Ejemplo

0	x	1	2	2	3	3	4	4	5	5	x
			A₁		A₂		A₃		A₄		A₅
6	7	7	8	8	9	9	10	10	11	11	x
	P₁		P₂		P₃		P₄		P₅		P₆
12	13	13	14	14	15	15	16	16	17	17	x
	A₁/P₁		A₁/P₂		A₁/P₃		A₁/P₄		A₁/P₅		A₁/P₆
18	19	19	20	20	21	21	22	22	23	23	x
	A₃/P₁		A₃/P₂		A₃/P₆		A₄/P₂		A₄/P₄		A₄/P₅
24	25	25	26	26	27	27	28	28	29	29	x



Representación de la BD en el nivel interno

Índice en la página 0

0		X
Conjunto	Dirección a Primera Pág	
Pág. Libres	24	
Asignaturas	1	
Profesores	6	
Imparte	12	



Representación de la BD en el nivel interno

Ejemplo

- Se inserta una nueva asignatura con código A6.

- Se localiza la primera página libre (la 24).
- Se inserta el registro correspondiente.
- Se añade esta página al conjunto de páginas de asignaturas.

0	1	2	3	4	5	24
	A ₁	A ₂	A ₃	A ₄	A ₅	
6	7	8	9	10	11	X
	P ₁	P ₂	P ₃	P ₄	P ₅	P ₆
12	13	14	15	16	17	
	A ₁ /P ₁	A ₁ /P ₂	A ₁ /P ₃	A ₁ /P ₄	A ₁ /P ₅	A ₁ /P ₆
18	19	20	21	22	23	X
	A ₃ /P ₁	A ₃ /P ₂	A ₃ /P ₆	A ₄ /P ₂	A ₄ /P ₄	A ₄ /P ₅
24	X	25	26	27	28	29
	A ₆					X



Representación de la BD en el nivel interno

Ejemplo

- ➡ Se borra la asignatura con código A2.
 - La página que contiene a esta asignatura pasa al conjunto de páginas libres.
 - Se reorganiza la lista correspondiente a Asignaturas.

0	1	3	2	25	3	4	5	24
		A₁			A₃	A₄	A₅	
6	7	8		9	10	11	X	
		P₂	P₃	P₄	P₅	P₆		
12	13	14		15	16	17		
	A₁/P₁	A₁/P₂	A₁/P₃	A₁/P₄	A₁/P₅	A₁/P₆		
18	19	20		21	22	23	X	
	A₃/P₁	A₃/P₂	A₃/P₆	A₄/P₂	A₄/P₄	A₄/P₅		
24	X	25		26	27	28	29	X
	A₆							



Representación de la BD en el nivel interno

Ejemplo

- Se introduce un nuevo profesor con código P7.

- Se ubica en la primera página libre disponible (la segunda).
- Se ajustan las cadenas de punteros.

0	1	3	2	2X	3	4	5	24
	A1	P7		A3		A4		A5
6	7	8		9	10	11	X	
	P1	P2	P3	P4	P5	P6		
12	13	14		15	16	17		
	A1/P1	A1/P2	A1/P3	A1/P4	A1/P5	A1/P6		
18	19	20		21	22	23	X	
	A3/P1	A3/P2	A3/P6	A4/P2	A4/P4	A4/P5		
24	X	25		26	27	28	29	X
	A6							

Representación de la BD en el nivel interno

Ejemplo

Se borra A4.

- Su página pasa al conjunto de páginas libres.
- Se reorganiza la cadena de punteros de las Asignaturas.

0	1	3	2	X	3	5	4	25	5	24
		A ₁	P ₇		A ₃				A ₅	
6	7	8		9	10			11	2	
	P ₁	P ₂	P ₃	P ₄	P ₅	P ₆				
12	13	14		15	16	17				
A _{1/P₁}	A _{1/P₂}	A _{1/P₃}	A _{1/P₄}	A _{1/P₅}	A _{1/P₆}					
18	19	20		21	22	23	X			
A _{3/P₁}	A _{3/P₂}	A _{3/P₆}	A _{4/P₂}	A _{4/P₄}	A _{4/P₅}					
24	X	25		26	27	28		29	X	
	A ₆									

Representación de la BD en el nivel interno

Ejemplo Punteros para el recorrido secuencial lógico

0	X	1	3	2	X	3	5	4	25	5	24		
			A ₁		P ₇		A ₃				A ₅		
6		7	7		8	8	9	9	10	10	11	11	2
		P ₁		P ₂		P ₃		P ₄		P ₅		P ₆	
12		13	13		14	14	15	15	16	16	17	17	18
		A _{1/P₁}		A _{1/P₂}		A _{1/P₃}		A _{1/P₄}		A _{1/P₅}		A _{1/P₆}	
18		19	19		20	20	21	21	22	22	23	23	X
		A _{3/P₁}		A _{3/P₂}		A _{3/P₆}		A _{4/P₂}		A _{4/P₄}		A _{4/P₅}	
24	X	25		26	26		27	27	28	28	29	29	30
		A ₆											



Representación de la BD en el nivel interno

Índice en la página 0

0		X
Conjunto	Dirección a Primera Pág	
Pág. Libres	24	
Asignaturas	1	
Profesores	6	
Imparte	12	



Representación de la BD en el nivel interno

Factor de bloqueo N

p	Información de cabecera	
A1	BASES DE DATOS	A2 ALGEBRA
A3	COMPUTABILIDAD	A4 METODOLOGÍA
A5	PROGRAMACION DE BD	
<i>Espacio libre</i>		

En realidad, las páginas contienen más de un registro

Representación de la BD en el nivel interno

Ejemplo

p	Información de cabecera	
A1	BASES DE DATOS	A3 COMPUTABILIDAD
A4	METODOLOGÍA	A5 PROGRAMACION DE BD
A7	MATEMÁTICA DISCRETA	A9 SISTEMAS DE BD
<i>Espacio libre</i>		

Inserción de A9, borrado de A2 e inserción de A7

Representación de la BD en el NI

Consideraciones finales

La organización descrita es un ejemplo general. Cada SGBD comercial utiliza su variante concreta, aunque la idea subyacente es la misma.

No existe una relación directa fichero-almacenado/fichero-físico, ya que todos los conjuntos de páginas irán almacenados, con toda probabilidad, en uno o varios ficheros físicos.



Contenidos

- Introducción
- Método de Acceso a la BD almacenada
- Representación de la BD en el nivel interno

Imágenes

- Imágenes tomadas de [Pixabay](#)
 - Portada
 - Imagen de [Manfred Steger](#)
 - Cabecera
 - Imágenes de [Gerd Altmann](#)
 - T11
 - Disco duro de [OpenClipart-Vectors](#)
 - Lector de [Jörn Heller](#)



Apartado 2

Métodos de organización y acceso a los datos: Índices

Contenidos

- Organización y métodos de acceso
- Organización secuencial
- Indexación
- Ficheros indexados
- Índices no densos
- Índices jerárquicos
- Árboles B+
- Árboles B
- Árboles B+ en BD
- Índices clave invertida
- Índices BITMAP

Contenidos

- Organización y métodos de acceso
- Organización secuencial
- Indexación
- Ficheros indexados
- Índices no densos
- Índices jerárquicos
- Árboles B+
- Árboles B
- Árboles B+ en BD
- Índices clave invertida
- Índices BITMAP

Organización y Métodos de Acceso

- **OBJETIVO:** Minimizar el número de accesos a disco → minimizar la cantidad de páginas de BD involucradas en una operación de BD.
 - Ninguna de las organizaciones presentadas es mejor en términos absolutos.
 - Criterios básicos para medir la “calidad” de una organización son:
 - Tiempo de acceso a los datos requeridos.
 - Porcentaje de memoria ocupada por los datos requeridos con respecto a las páginas de BD que los contienen.
 - Trabajaremos a dos niveles:
 - Organización de registros de datos a nivel de almacenamiento.
 - Adición de estructuras complementarias para acelerar el acceso a dichos registros.



Contenidos

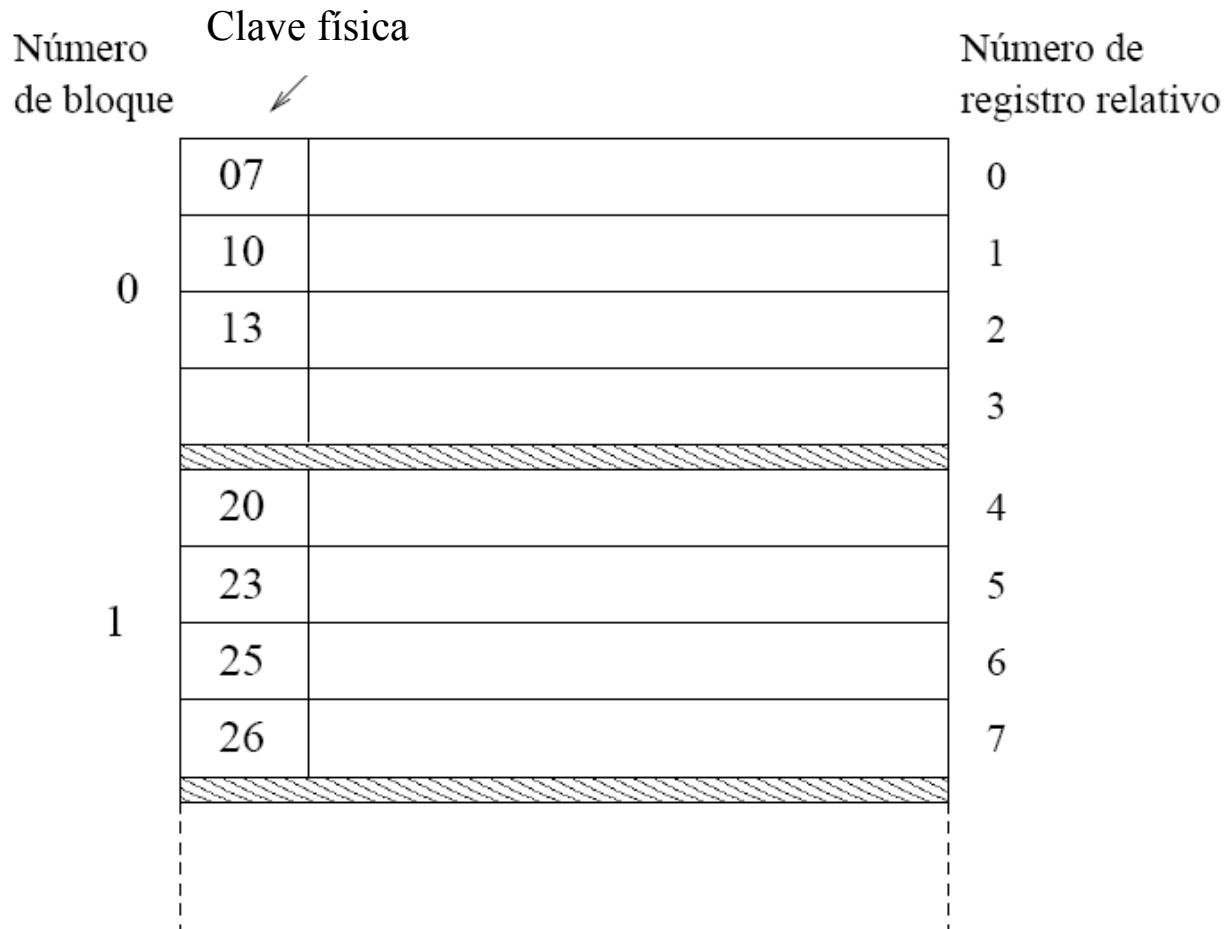
- Organización y métodos de acceso
- Organización secuencial
- Indexación
- Ficheros indexados
- Índices no densos
- Índices jerárquicos
- Árboles B+
- Árboles B
- Árboles B+ en BD
- Índices clave invertida
- Índices BITMAP

Organización Secuencial

- Fichero de acceso secuencial:
 - Aquel donde los registros están almacenados consecutivamente.
 - Para acceder a un registro determinado debemos pasar obligatoriamente por los registros que le preceden.
 - Los registros suelen estar ordenados por una clave (clave física).



Organización Secuencial



Organización Secuencial

- Ejemplo:

- Mostrar la relación completa de departamentos.
- La consulta se resolvería rápidamente si los departamentos están almacenados conjuntamente en bloques contiguos de un fichero.
- Sin embargo:
 - ¿Qué pasa si queremos plantear consultas por valor de clave o por rango de valores?



Organización Secuencial

- El primer caso implica:
 - Recorrer uno tras otro cada uno de los registros.
 - En el caso peor (no encontrarse dicho departamento – si el fichero no está ordenado por el valor de la clave, o ser el último de la lista) supone recorrer de forma completa el fichero.
 - Búsqueda es $O(N)$
- El segundo caso tiene un tratamiento muy parecido:
 - Se realiza la búsqueda por valor de clave de la cota inferior del intervalo.
 - Se continúa hasta alcanzar la cota superior, si están ordenados por el valor de la clave.



Organización Secuencial

- Inserción de un nuevo registro:
 - Buscar el bloque que le corresponde.
 - Si hay sitio, se inserta el nuevo registro.
 - En caso contrario, o bien se opta por crear un nuevo bloque o bien se crea un bloque de desbordamiento.
 - Es recomendable dejar espacio vacío en los bloques para evitar los problemas de reorganización.
- Borrado de un registro:
 - Buscar el registro.
 - Puede implicar una reorganización local de los registros de un bloque.



Organización Secuencial

- En resumen, las dos operaciones pueden suponer:
 - Localización del registro o del lugar que debe ocupar.
 - Escritura del bloque del registro que se inserta o borra.
 - Creación o liberación de bloques de datos en el fichero secuencial.
 - Creación o liberación de bloques de desbordamiento.
 - Reorganización de registros entre bloques contiguos, lo que implica la escritura de los bloques implicados en el desplazamiento.

Organización Secuencial

- Como puede verse, esta forma de organizar los registros no está exenta de grandes inconvenientes.
- Pueden subsanarse, al menos en parte, mediante el uso de estructuras adicionales que nos permitan:
 - Acelerar la localización de los datos.
 - Disminuir el número de bloques de disco transferidos.
- Entre las técnicas más populares se encuentran:
 - Índices
 - Acceso directo.



Contenidos

- Organización y métodos de acceso
- Organización secuencial
- Indexación
- Ficheros indexados
- Índices no densos
- Índices jerárquicos
- Árboles B+
- Árboles B
- Árboles B+ en BD
- Índices clave invertida
- Índices BITMAP

Indexación

- Tiene por objeto disminuir el tiempo de acceso a los datos por una clave de búsqueda.
- Similar a la idea de un índice en un libro.
- Existen muchas formas de llevar a cabo esta idea.



Contenidos

- Organización y métodos de acceso
- Organización secuencial
- Indexación
- Ficheros indexados
- Índices no densos
- Índices jerárquicos
- Árboles B+
- Árboles B
- Árboles B+ en BD
- Índices clave invertida
- Índices BITMAP

Ficheros indexados

- Partimos de un fichero secuencial sobre el que disponemos una estructura adicional: fichero índice
 - Sus registros poseen:
 - Campo clave (la clave de búsqueda)
 - Campo de referencia que contiene RIDs de registros.
 - Son más pequeños que los del fichero de datos, aunque el número de ellos es el mismo en ambos ficheros.

Ficheros indexados

Índice primario

Número
de bloque

RID

07	
10	
13	
15	
20	
23	
25	
26	

Número
de bloque

07		...
10		
13		
15		
20		
23		
25		
26		

Número de
registro relativo

- 0
- 1
- 2
- 3
- 4
- 5
- 6
- 7

Fichero secuencial



Ficheros indexados

- Índice primario:
 - La clave de búsqueda es el mismo campo por el que está ordenado el fichero secuencial de datos (clave física).
- Índices Secundarios:
 - Construidos sobre otros campos que no sean la clave física del fichero de datos.

Ficheros indexados

Indice primario

RID

07	
10	
13	
15	
20	
23	
25	
26	

Número
de bloque

07		21
10		81
13		60
15		10
20		21
23		95
25		73
26		94

Fichero secuencial

Indice secundario

RID

10	
21	
21	
60	
73	
81	
94	
95	



Ficheros indexados

- Proceso de consulta
 - Consulta por un valor de la clave
 - Sobre el índice localizamos la clave (recorrido secuencial).
 - Obtenemos el RID del registro requerido.
 - Vamos a disco para recuperar el bloque de datos donde se encuentra el registro señalado por el RID.
 - La búsqueda en el índice es más rápida.
 - Consulta por rango de valores:
 - Búsqueda en el índice por valor de clave de la cota inferior.
 - Recorrido de las entradas del índice que están en el intervalo, recuperando los registros correspondientes gracias a su RID.



Ficheros indexados

- Inserción y borrado de registros:
 - La idea es similar a la operación en el fichero secuencial, aunque determinados procesos de búsqueda pueden ayudarse del índice.
 - Hay que actualizar también el índice (inserción o borrado en un fichero secuencial).



Ficheros indexados

- Ejemplo: Borrado del registro de clave 13

Índice denso

Número
de bloque

Número de bloque	RID
07	
10	
15	13
20	15
23	20
25	
26	

Fichero secuencial

Número de
registro relativo

0	07	
1	10	
2	13	15
3		espacio libre
4	20	
5	23	
6	25	
7	26	



Ficheros indexados

- Se puede montar un índice sobre más de un campo de un registro.
 - Clave: puede verse como la concatenación de los campos indicados.
- Hay que tener cuidado:
 - Un índice sobre nombre-alumno y DNI.
 - Útil para consultas que involucran:
 - Nombre
 - Nombre y DNI
 - No es útil para consultas sobre el DNI.



Ficheros indexados

- Conclusiones:

- Los índices:

- Aceleran determinados procesos de búsqueda.
 - Suponen un coste adicional en términos de espacio.
 - Hay que mantenerlos en operaciones de inserción, actualización y borrado.

- Por tanto:

- Hay que considerar la conveniencia de crear cada índice.
 - Disponibilidad de espacio de almacenamiento.
 - Frecuencia de las consultas.
 - Frecuencia de las operaciones de mantenimiento de los datos.



Contenidos

- Organización y métodos de acceso
- Organización secuencial
- Indexación
- Ficheros indexados
- Índices no densos
- Índices jerárquicos
- Árboles B+
- Árboles B
- Árboles B+ en BD
- Índices clave invertida
- Índices BITMAP



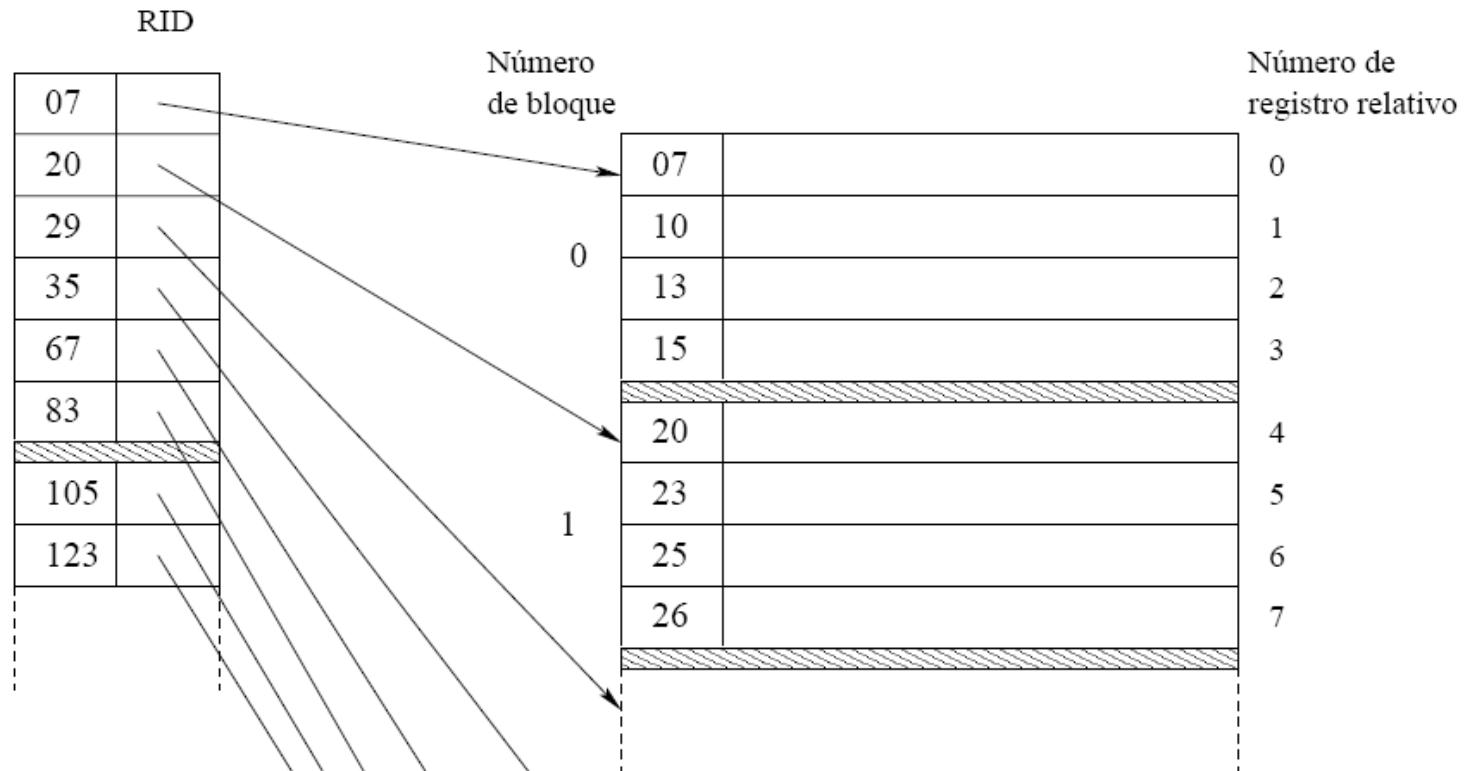
Índices no densos

- Ideal: Mantener el índice en memoria principal.
- Realidad: los índices siguen siendo muy grandes, porque contienen todos los registros del fichero que indexan.
 - Son densos.
- Para reducir el tamaño aparecen los índices no densos:
 - Registros compuestos por:
 - La clave de búsqueda
 - La dirección de comienzo del bloque donde puede encontrarse el registro deseado
 - El número de registros se reduce al número de bloques del fichero de datos
 - El acceso secuencial al índice no denso se acelera.



Índices no densos

Indice no denso



Índices no densos

- Diferencias en el proceso de búsqueda:
 - Una vez encontrado el bloque donde podría encontrarse el registro:
 - Hay que cargarlo en memoria.
 - Hay que hacer una búsqueda secuencial.
 - No tiene costes en términos de acceso a disco.
 - No se tiene garantía alguna de encontrar el registro deseado hasta consultar el bloque de datos leído.



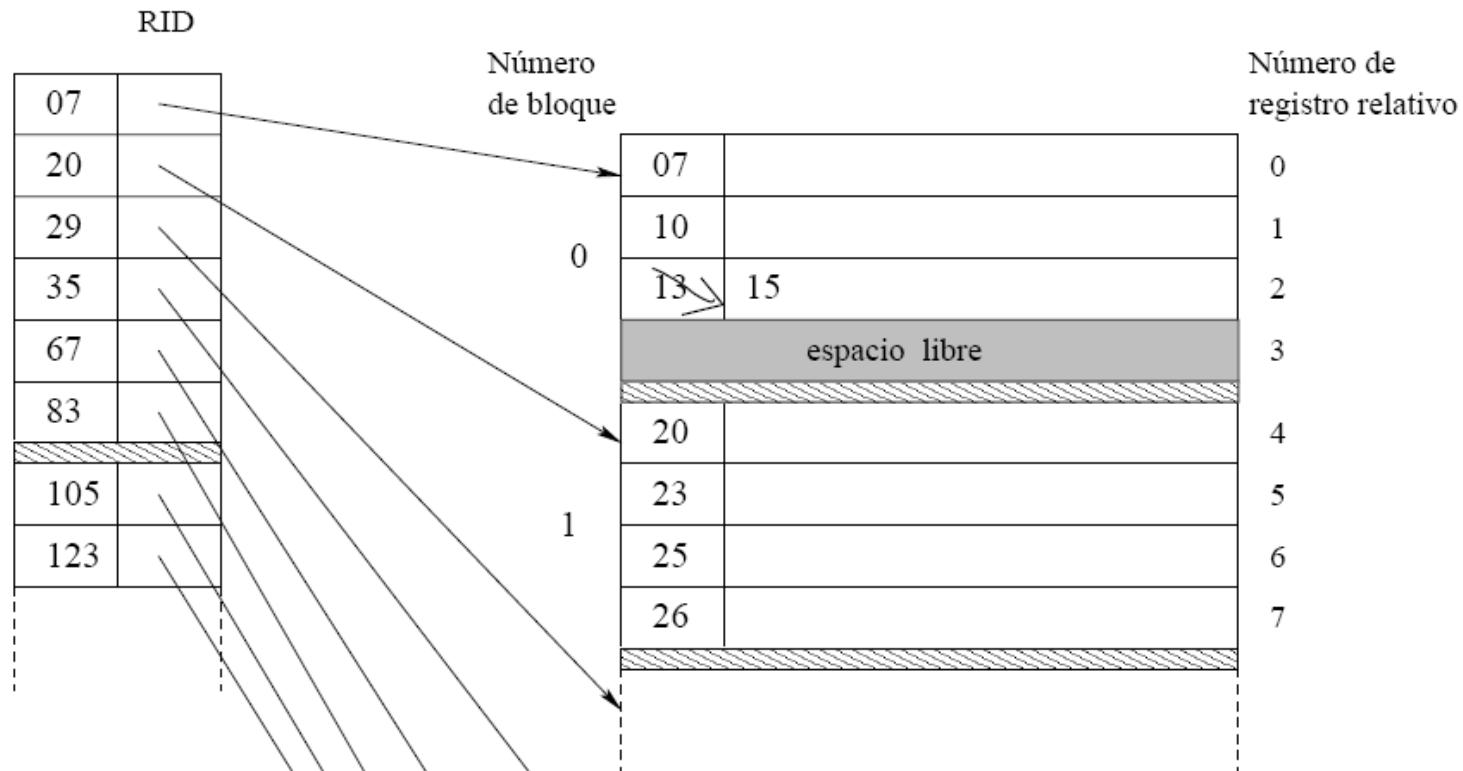
Índices no densos

- Los índices no densos solo se pueden definir sobre la clave física.
- El mantenimiento de un índice no denso es menos costoso:
 - Inserción y borrado menos frecuentes.
 - Ocurren cuando la operación afecta al valor representativo del bloque.



Índices no densos

Indice no denso



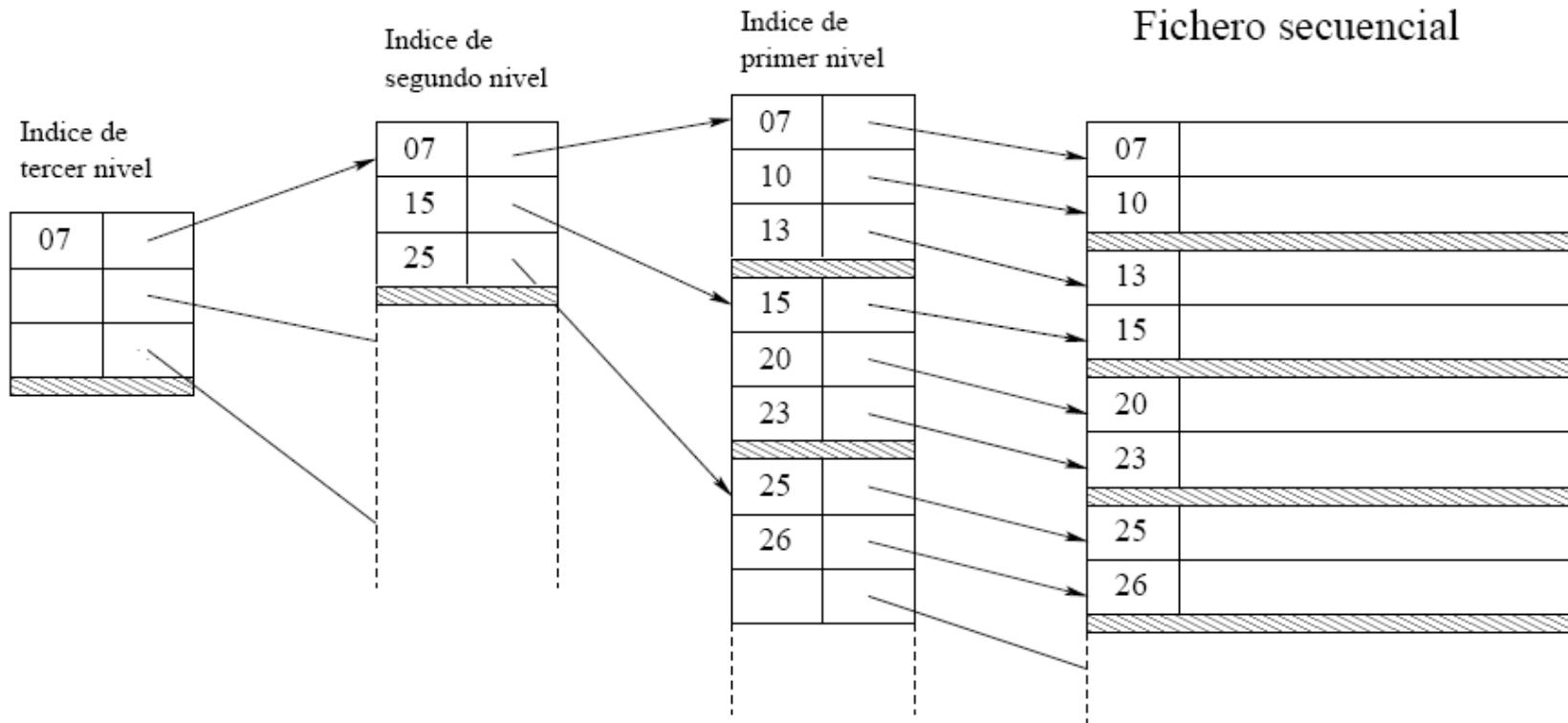
Contenidos

- Organización y métodos de acceso
- Organización secuencial
- Indexación
- Ficheros indexados
- Índices no densos
- Índices jerárquicos
- Árboles B+
- Árboles B
- Árboles B+ en BD
- Índices clave invertida
- Índices BITMAP

Índices jerárquicos

- Volvemos al objetivo de disminuir el tiempo necesario para recorrer el índice en busca de un registro:
 - Idea: crear índices sobre índices.
 - Varios niveles en el acceso a los datos.
- Un índice jerárquico o multinivel está formado:
 - Un índice de primer nivel sobre el fichero de datos.
 - Puede ser denso o no dependiendo de la clave.
 - Otros índices, no densos, construidos sucesivamente unos sobre otros.
- El tamaño de los bloques se establece con la idea de optimizar cada una de las operaciones de acceso al disco físico.
- Se reduce el número de accesos a disco para localizar un registro:
 - Viene determinado por el número de niveles.
 - Ideal: como mucho, un acceso a disco por nivel, más el acceso para recuperar el bloque del fichero de datos.
- Se complica el mantenimiento del índice.

Indices jerárquicos

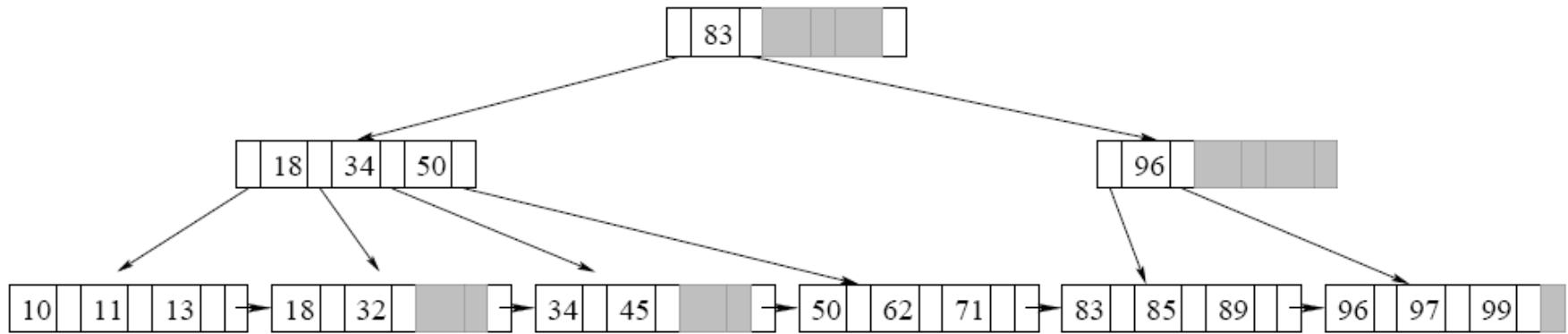


Contenidos

- Organización y métodos de acceso
- Organización secuencial
- Indexación
- Ficheros indexados
- Índices no densos
- Índices jerárquicos
- Árboles B+
- Árboles B
- Árboles B+ en BD
- Índices clave invertida
- Índices BITMAP

- Propuestos en 1972 por Bayer y McCreight, son una generalización de los árboles binarios balanceados en la que los nodos pueden tener más de dos hijos.
- Todos los valores de la clave se encuentran almacenados en los nodos hoja.
- Un Árbol B+ de orden M (el máximo número de hijos que puede tener cada nodo) es un árbol con la siguiente estructura:
 - Nodo de nivel superior: raíz del árbol
 - Nodos del nivel inferior: hojas.
 - Cada nodo distinto de las hojas tiene como máximo M hijos.
 - Todos los nodos hoja aparecen al mismo nivel.
 - Las claves contenidas en cada nodo nos guiarán hasta el siguiente nodo del nivel inmediatamente inferior.
 - Un nodo no hoja con n hijos contiene:
 - n-1 valores de clave almacenados.
 - n punteros P_i que apuntan a un nodo hijo

Árboles B+



Herramienta interactiva:

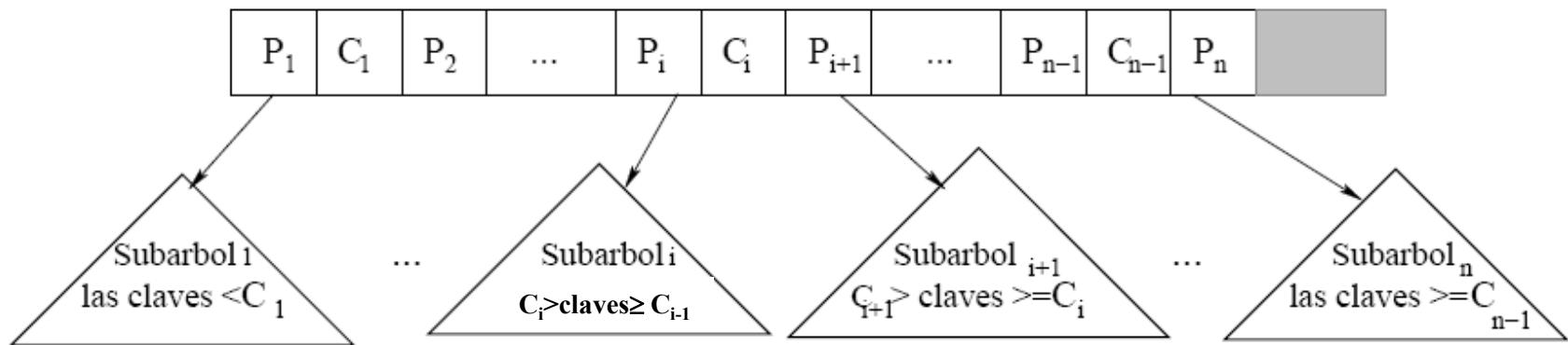
<https://www.cs.usfca.edu/~galles/visualization/BPlusTree.html>

Árboles B+

- Restricciones dentro de los nodos:

- Los valores de clave C_i están ordenados dentro del nodo.
- Los valores x del subárbol apuntado por P_i cumplen:

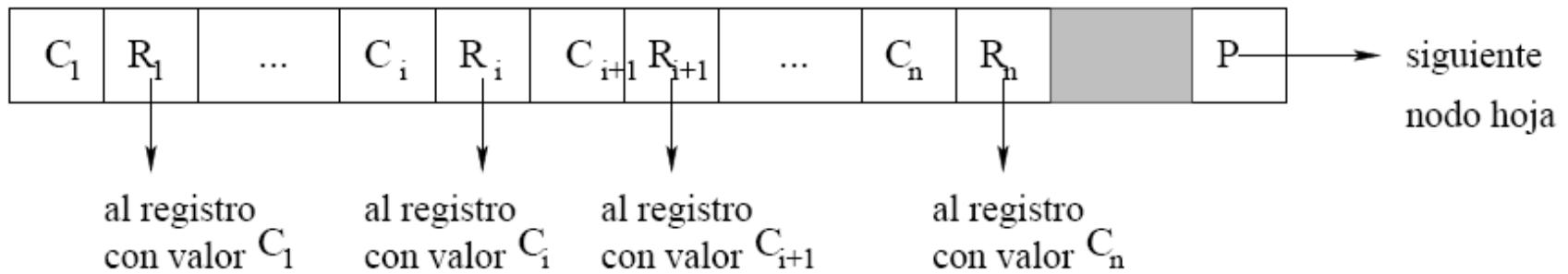
- $C_{i-1} \leq x < C_i$
- Excepto para:
 - $i = 1$, donde $x < C_1$
 - $i = n$, donde $x \geq C_{n-1}$
- Salvo que se repitan valores; entonces hay que adaptar lo anterior.



Árboles B+

- Nodos hoja:

- Tienen una estructura diferente
 - Parejas clave – RID
 - Punteros al siguiente nodo hoja
 - Algunas variantes también tienen punteros al nodo hoja anterior.
- La lista concatenada de nodos hoja (conjunto secuencia) tiene gran utilidad a la hora de hacer consultas por intervalos.



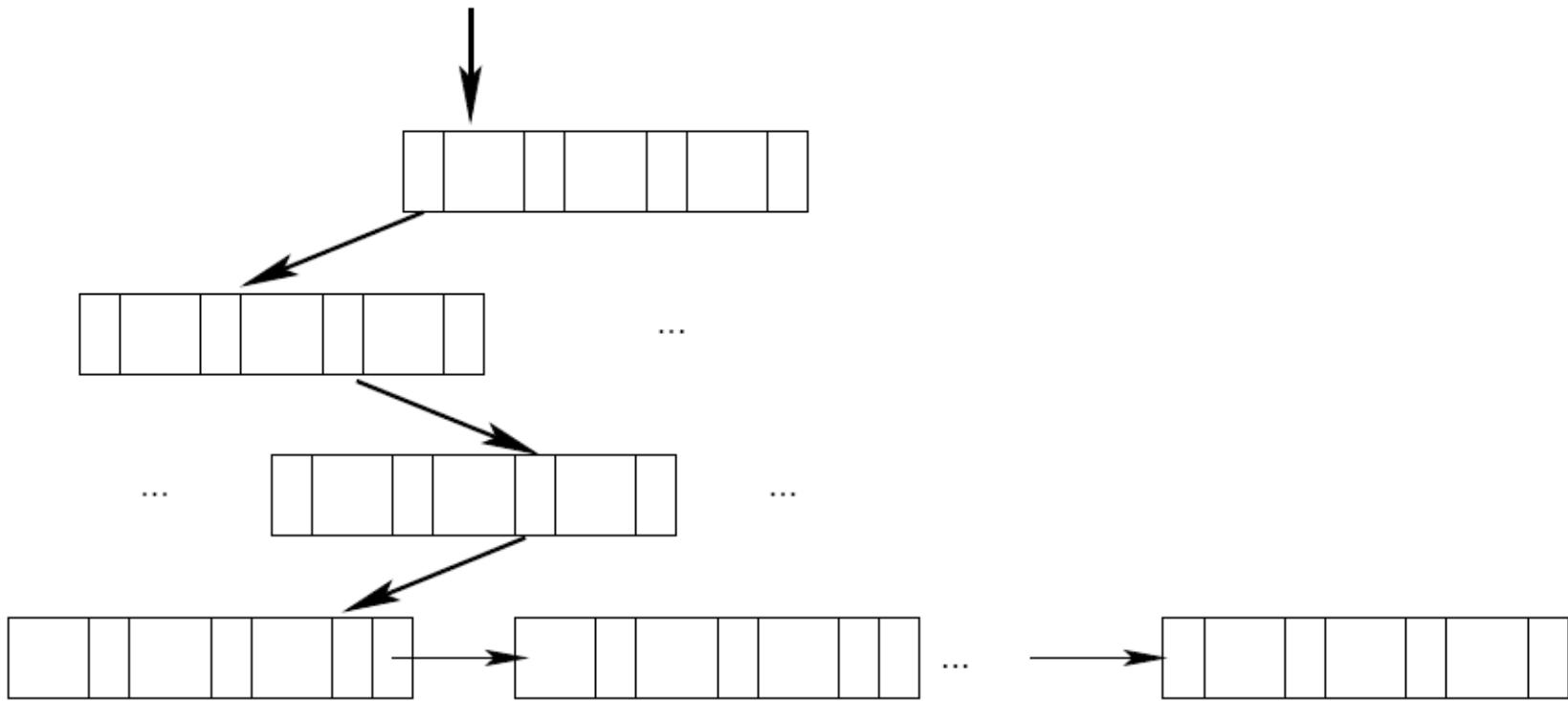
- Restricciones en nodos hoja:
 - Las claves aparecen ordenadas en cada nodo y entre nodos.
 - Todos los nodos hoja se encuentran en el mismo nivel.
 - Árbol equilibrado.
 - Todos los caminos desde la raíz a un nodo hoja tienen la misma longitud.

Árboles B+

Nodo	Restricción	Min	Max	Ejemplo $M=5$
Raíz (cuando no es nodo único)	Número de hijos	2	M	2-5
Interno	Número de hijos	$\lceil M/2 \rceil$	M	3-5
Hoja	Número de claves	$\lfloor M/2 \rfloor$	M-1	2-4

- Proceso de consulta
 - Localización de un registro:
 - Navegamos desde la raíz, bajando niveles.
 - Buscamos el registro en el nodo hoja y, en su caso, recuperamos el registro del fichero de datos gracias al RID.
 - Consultas por rango:
 - Se localiza el nodo hoja que contiene el valor inferior.
 - Se recorren los nodos hoja hasta alcanzar el superior, recuperando los registros pertinentes del fichero de datos.
- Inserción y borrado
 - Se utilizan algoritmos que garantizan que el árbol resultante sea equilibrado.

Árboles B+



Contenidos

- Organización y métodos de acceso
- Organización secuencial
- Indexación
- Ficheros indexados
- Índices no densos
- Índices jerárquicos
- Árboles B+
- Árboles B
- Árboles B+ en BD
- Índices clave invertida
- Índices BITMAP

- Los Árboles-B (B-Tree) son una variante de B+Tree en la que no se almacenan todos los valores de la clave en los nodos hoja, sino que algunos valores se van almacenando en los nodos intermedios conforme se crea el árbol.

Herramienta interactiva:

<https://www.cs.usfca.edu/~galles/visualization/BTree.html>



Contenidos

- Organización y métodos de acceso
- Organización secuencial
- Indexación
- Ficheros indexados
- Índices no densos
- Índices jerárquicos
- Árboles B+
- Árboles B
- Árboles B+ en BD
- Índices clave invertida
- Índices BITMAP

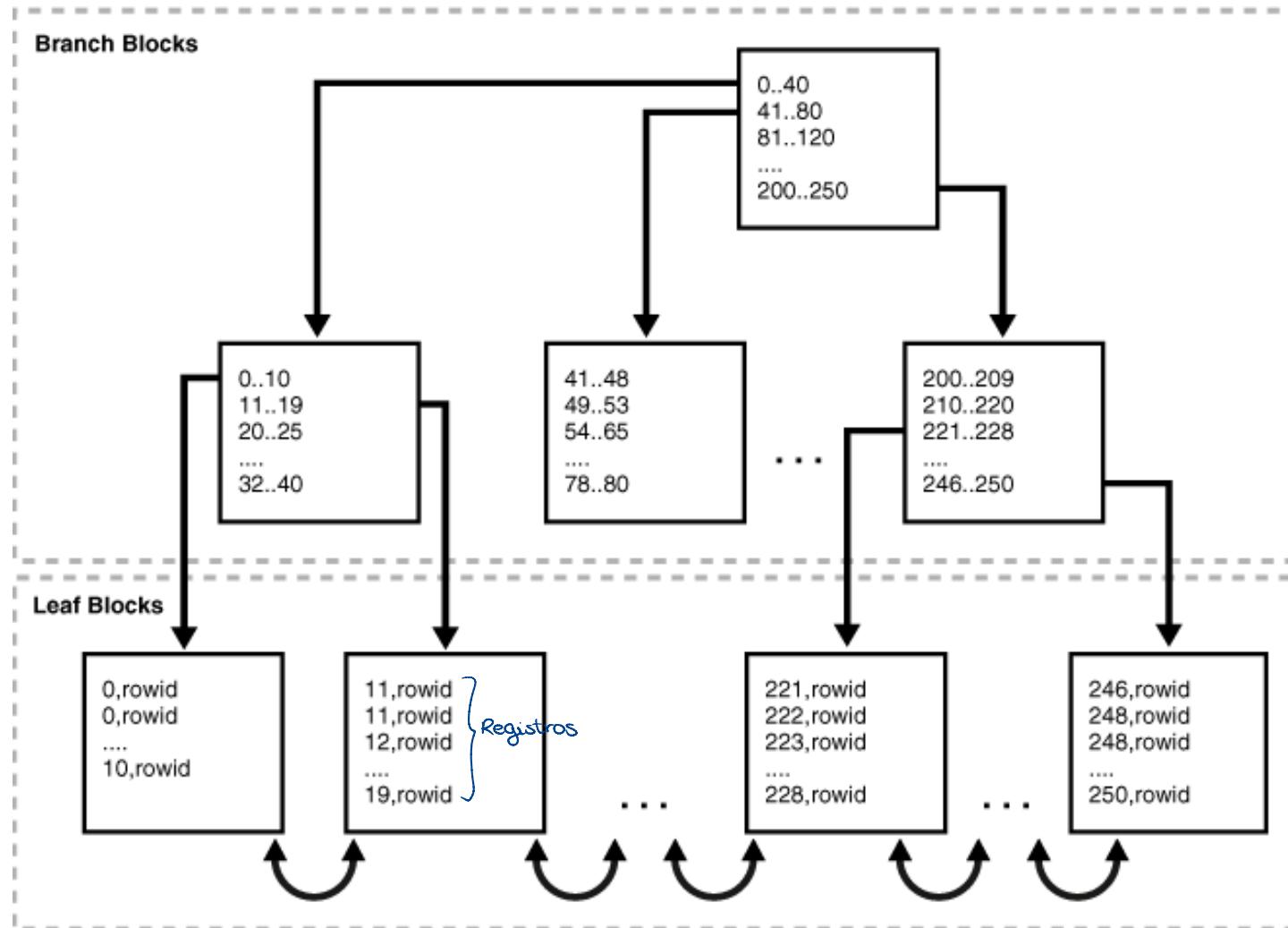


Árboles B+ en Bases de Datos

- Son variaciones del Árbol B+, de orden elevado; se procura que cada nodo tenga una capacidad de almacenamiento similar al tamaño de un bloque de datos.
- Esto reduce los accesos a disco que suelen ser los que determinan el rendimiento de las búsquedas en BD.
- En los nodos intermedios solo están los rangos de los valores de la clave y los punteros a los nodos hijo correspondientes.
- En los nodos hoja se encuentran todos los valores de la clave ordenados junto con los RIDs(rowid) que apuntan a las tuplas que contienen ese valor de la clave.
- Los nodos hoja, que forman el conjunto secuencia, se encuentran enlazados para poder recuperar por búsquedas secuenciales, a veces se encuentran doblemente enlazados, para facilitar búsquedas ascendentes y descendentes por el valor de la clave.

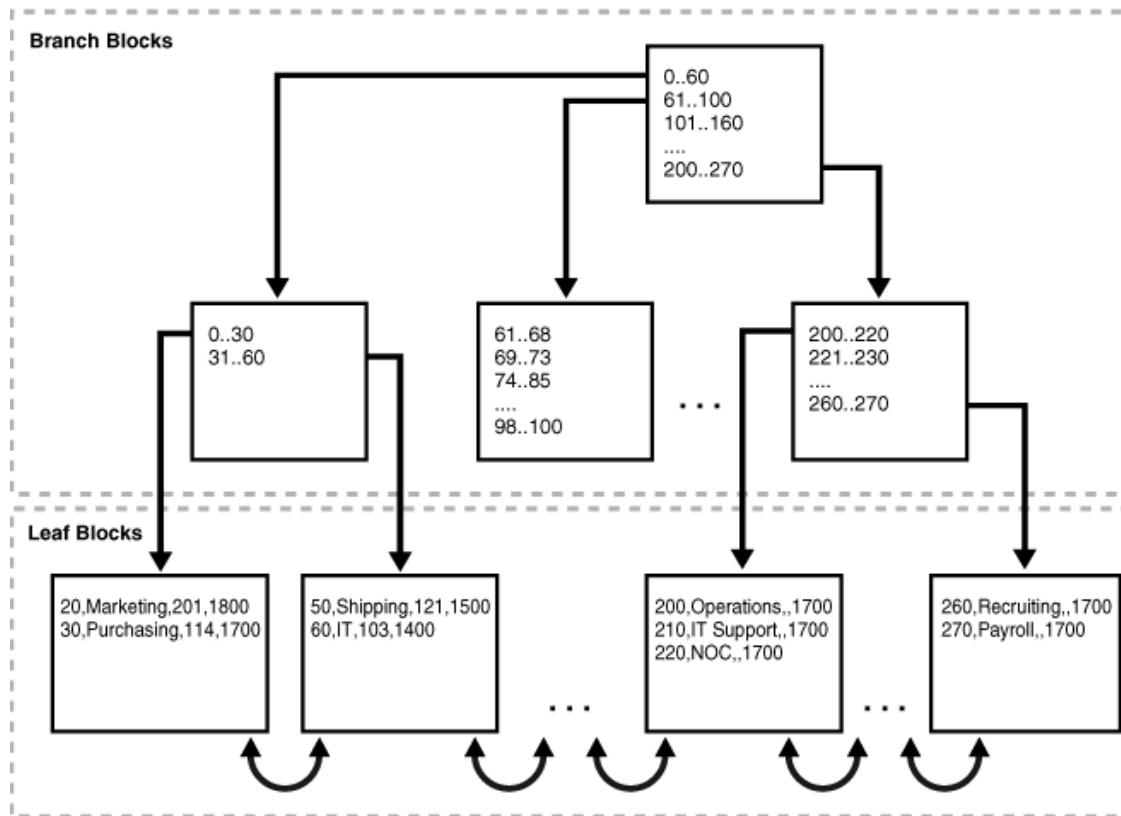


Árboles B+ en Bases de Datos



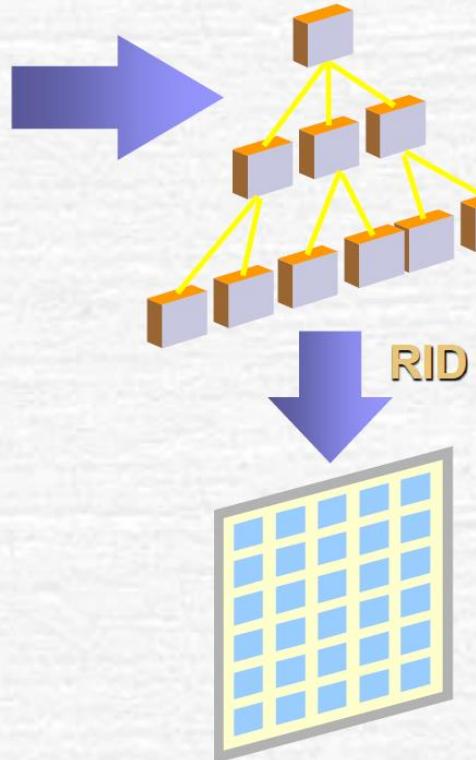
Árboles B+ en Bases de Datos

- Tablas Organizadas por Índice (IOT). Las hojas contienen las tuplas en lugar del RID. Una IOT solo puede estar organizada de esta forma mediante una clave (normalmente la clave primaria), aunque se pueden definir índices adicionales basados en otras claves.

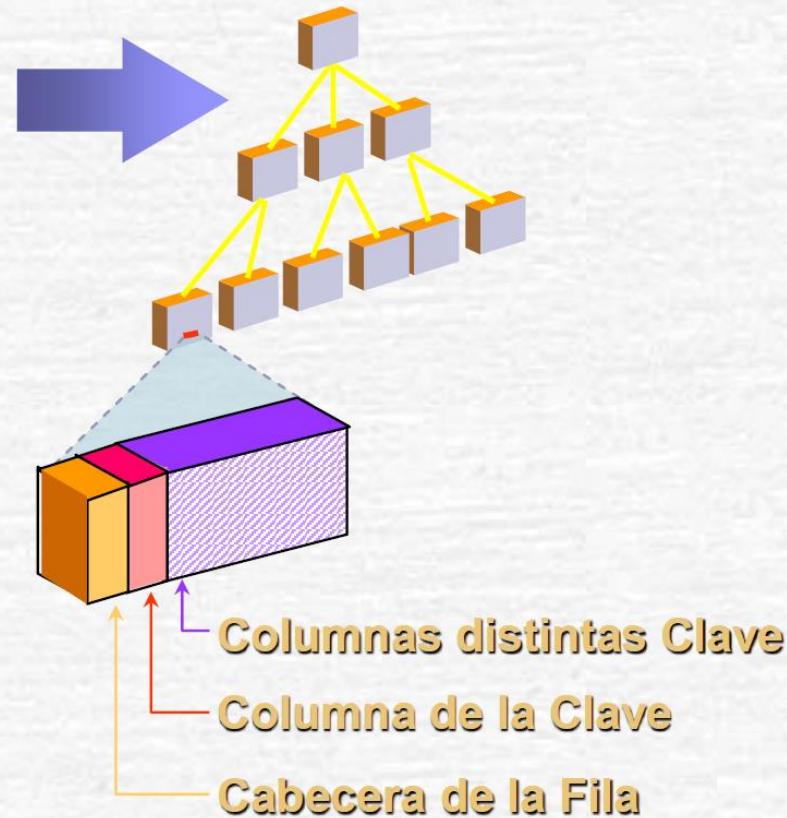


Árboles B+ en Bases de Datos

Acceso indizado a la tabla



Acceso a una IOT



Contenidos

- Organización y métodos de acceso
- Organización secuencial
- Indexación
- Ficheros indexados
- Índices no densos
- Índices jerárquicos
- Árboles B+
- Árboles B
- Árboles B+ en BD
- Índices clave invertida
- Índices BITMAP

Índices por Clave Invertida

- Índice por Clave Invertida (reverse index). Invierte los datos del valor de la clave.
 - Para el empleado 7698 almacena 8967.
 - Este índice es adecuado para búsquedas basadas en predicados `=`, pero no por intervalos.
 - Con este índice se reducen los embotellamientos (retención de bloques de BD) en el índice cuando se manipulan datos consecutivos de forma concurrente.

Así se mejora el acceso concurrente
a instancias de una entidad con
claves consecutivas



Índices Clave Invertida

Índice Invertido sobre EMP(EMPNO)

KEY	ROWID
EMPNO	(BLOCK# ROW# FILE#)
1257	0000000F.0002.0001
2877	0000000F.0006.0001
4567	0000000F.0004.0001
6657	0000000F.0003.0001
8967	0000000F.0005.0001
9637	0000000F.0001.0001
9947	0000000F.0000.0001
...	...
...	...

Tabla EMP

EMPNO	ENAME	JOB	...
7499	ALLEN	SALESMAN	
7369	SMITH	CLERK	
7521	WARD	SALESMAN	...
7566	JONES	MANAGER	
7654	MARTIN	SALESMAN	
7698	BLAKE	MANAGER	
7782	CLARK	MANAGER	
...
...



Contenidos

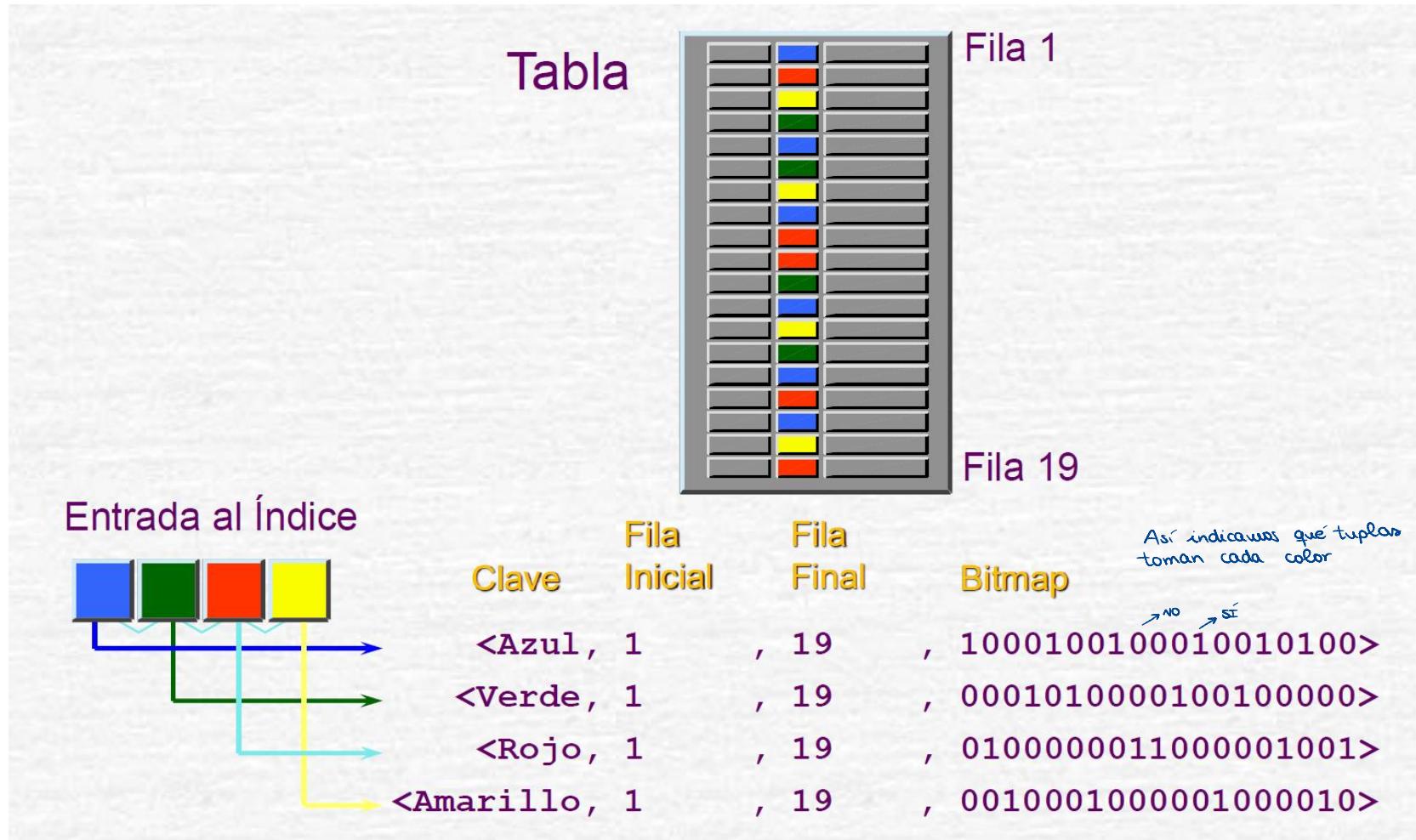
- Organización y métodos de acceso
- Organización secuencial
- Indexación
- Ficheros indexados
- Índices no densos
- Índices jerárquicos
- Árboles B+
- Árboles B
- Árboles B+ en BD
- Índices clave invertida
- Índices BITMAP

Índices BITMAP

- Indices BITMAP

- Para cada valor que toma la clave almacena una secuencia de bits (tantos como tuplas contenga la tabla):
 - El bit 1 indica que ese valor está presente en la tupla.
 - El 0 que no lo está.

Indices BITMAP



Índices BITMAP

B-tree	Bitmap
Adecuado para columnas que tomen muchos valores	Adecuado para columnas que tomen pocos valores
Actualizaciones sobre las claves no muy costosa	Actualizaciones sobre las claves muy costosa
Ineficiente para consultas usando predicados OR	Eficiente para consultas usando predicados OR



Contenidos

- Organización y métodos de acceso
- Organización secuencial
- Indexación
- Ficheros indexados
- Índices no densos
- Índices jerárquicos
- Árboles B+
- Árboles B
- Árboles B+ en BD
- Índices clave invertida
- Índices BITMAP

Imágenes

- Imágenes tomadas de [Pixabay](#)
 - Portada
 - Imagen de [Manfred Steger](#)
 - Cabecera
 - Imágenes de [Gerd Altmann](#)

Tema 4

El Nivel Interno

Apartado 3

Métodos de organización y acceso a los datos: Acceso Directo

Contenidos

- Acceso directo
- Hashing básico
- Hashing dinámico

Contenidos

- Acceso directo
- Hashing básico
- Hashing dinámico

Acceso directo

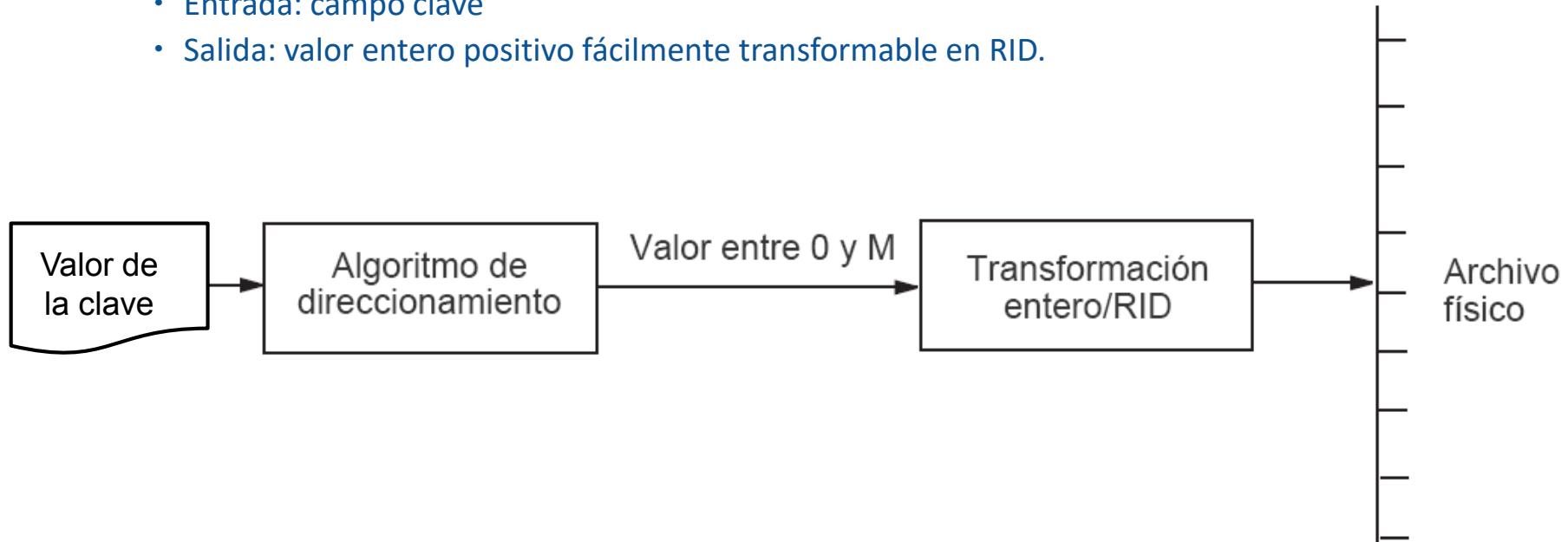
- Otra forma de acceder a un registro almacenado
 - No hay una estructura adicional.
 - Se usa un algoritmo que nos indique directamente la posición del registro deseado.
- Acceso directo:
 - Calcular directamente la dirección de un registro mediante la aplicación de algún algoritmo o función sobre un campo determinado del mismo.
 - El campo debe identificar únicamente al registro.



Acceso directo

- Funcionamiento:

- Normalmente no es posible establecer una clave que sea totalmente correlativa y única para cada registro.
- Hay que buscar un algoritmo que transforme los valores de un cierto campo en una dirección.
 - Entrada: campo clave
 - Salida: valor entero positivo fácilmente transformable en RID.



Acceso directo

- Los algoritmos de direccionamiento no suelen mantener el orden de la clave.
 - Los registros no están almacenados según el orden de la clave.
 - Problemas con la recuperación por intervalos.

Acceso directo

- Hay una gran variedad de algoritmos:
 - Dependen del tipo de clave:
 - Si la clave es alfanumérica, hay que transformarla a un valor numérico.
 - Suelen estar basados en un mecanismo de generación de números pseudoaleatorios:
 - *Cuadrados centrales*:
 - Se eleva la clave al cuadrado y se eligen tantos dígitos centrales como sea necesario.
 - *Congruencias*:
 - Se divide la clave por M y se toma el resto.
 - *Desplazamiento*:
 - Se superponen adecuadamente los dígitos binarios de la clave y luego se suman.
 - *Conversión de base*:
 - Se cambia la base de numeración y se suprimen algunos dígitos resultantes.



- Problemas

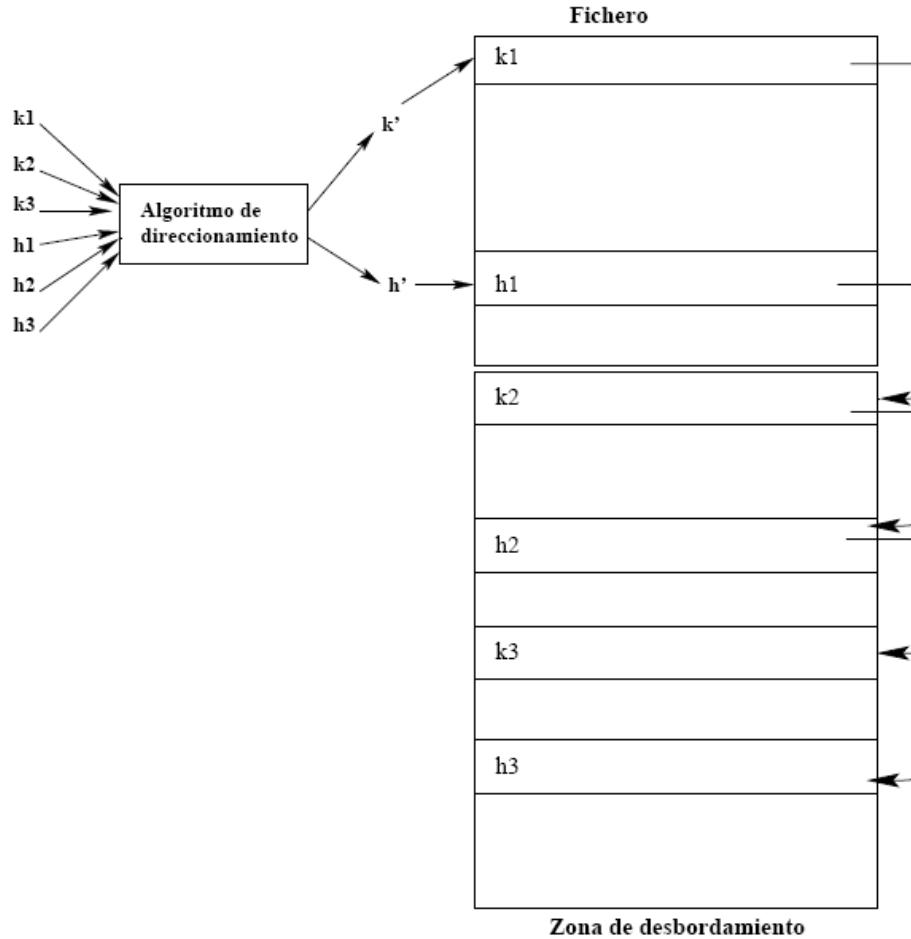
- Salvo que el campo clave se diseñe para ello, es prácticamente imposible encontrar una transformación que dé un valor entero positivo en un rango de valores limitado tal que no haya dos valores distintos de clave que den lugar al mismo número.
 - Por lo que se producen colisiones.
- Los algoritmos también producen huecos:
 - Zonas vacías del rango de salida, no asignadas por el algoritmo.
 - Se traducen en huecos en el fichero de datos.

Acceso directo

- Para gestionar colisiones y huecos:
 - Combinar el acceso directo con una gestión mediante listas de colisión.
 - Zona de desbordamiento.
 - Al producirse una colisión:
 - El registro problemático se almacena en la zona de desbordamiento.
 - Los sinónimos (registros con claves que producen colisión) se conectan mediante una lista.



Acceso directo



Acceso directo

- Si crecen las listas de sinónimos:
 - El acceso directo puro no resulta adecuado.
 - Mantener listas.
 - Zona de desbordamiento casi como el fichero original.
- Han aparecido técnicas más sofisticadas:
 - Hashing



Contenidos

- Acceso directo
- Hashing básico
- Hashing dinámico

Hashing básico

- Si el problema principal es que los valores de las claves no están uniformemente distribuidos en el intervalo $[0, M]$:
 - Se acumulan en una parte de este intervalo.
 - Solución:
 - Asignar más espacio a esa parte del intervalo.
- Técnica:
 - Se divide el espacio del fichero en “cubos” (buckets).
 - El algoritmo de direccionamiento asigna cubos, no direcciones concretas.
 - En cada “cubo” puede haber más de un registro.
 - Ciertos rangos de valores tienen asignados más cubos que otros.
 - Se complementa con el uso de “cubos de desbordamiento”.



Hashing básico

- Parámetros:

- Número de cubos.
- Tamaño de los cubos (relación con bloques físicos)
 - “slots” por cubo.
- La transformación clave/dirección, que debe tener en cuenta la distribución de la clave según rangos para evitar que unos cubos no se llenen mucho y otros se queden muy vacíos.



Hashing básico

- Para insertar un registro:
 - Transformar la clave.
 - Localizar el cubo correspondiente.
 - Si hay sitio se inserta el registro y hemos terminado.
 - Si no hay sitio, se sitúa el registro en un cubo de desbordamiento conectándolo con el cubo que realmente le corresponde mediante punteros.



Hashing básico

- El proceso de búsqueda:
 - Transformar la clave.
 - Localizar el cubo correspondiente.
 - Realizar una búsqueda secuencial dentro del cubo.
 - Si hemos encontrado el registro, el proceso termina.
 - En caso contrario, se impone “un barrido por punteros” a través de los cubos de desbordamiento.

Contenidos

- Acceso directo
- Hashing básico
- Hashing dinámico

Hashing dinámico

- El hashing básico sigue teniendo problemas:
 - Es necesario conocer la distribución previa de las claves para asignar adecuadamente los cubos.
 - En otro caso siguen apareciendo huecos/colisiones.
 - Al aumentar el número de registros, aumentan los registros en páginas de desbordamiento.
 - Se hacen necesarias las reorganizaciones.



Hashing dinámico

- Solución:
 - Trabajar de forma dinámica.
- Se parte de una configuración uniforme y de pocos cubos.
- Los restantes se van generando conforme se necesiten.
 - Se asignan a los rangos conforme la afluencia de registros lo demanda.
 - Hashing dinámico o extensible



Hashing dinámico

- Técnica:
 - El valor transformado del campo clave nos lleva a la entrada de una tabla índice que se almacena en memoria.
 - Allí está la dirección del cubo donde se encuentran los registros que tienen asociado este valor transformado.
 - Puede ocurrir que varias entradas de la tabla conduzcan al mismo cubo.
- Proceso:
 - Se toma un conjunto inicial de cubos, direccionados por la tabla índice.
 - A medida que vamos insertando registros, se van generando nuevos cubos y cambiando las salidas de la tabla índice.



Hashing dinámico

- Algoritmo de Hashing Dinámico

- Datos de partida:

- k = clave para direccionar
 - $k' = h(k)$ valor entero entre 0 y M
 - n = número de bits que tiene k' en binario
 - $d \leq n$, los d primeros* dígitos de k' seleccionan el cubo donde está el registro y se llaman pseudollave.
 - $b < d \leq n$, inicialmente el archivo tiene 2^b cubos distintos, como máximo tendrá 2^d . Si son necesarios más, hay que aumentar d (si se puede).



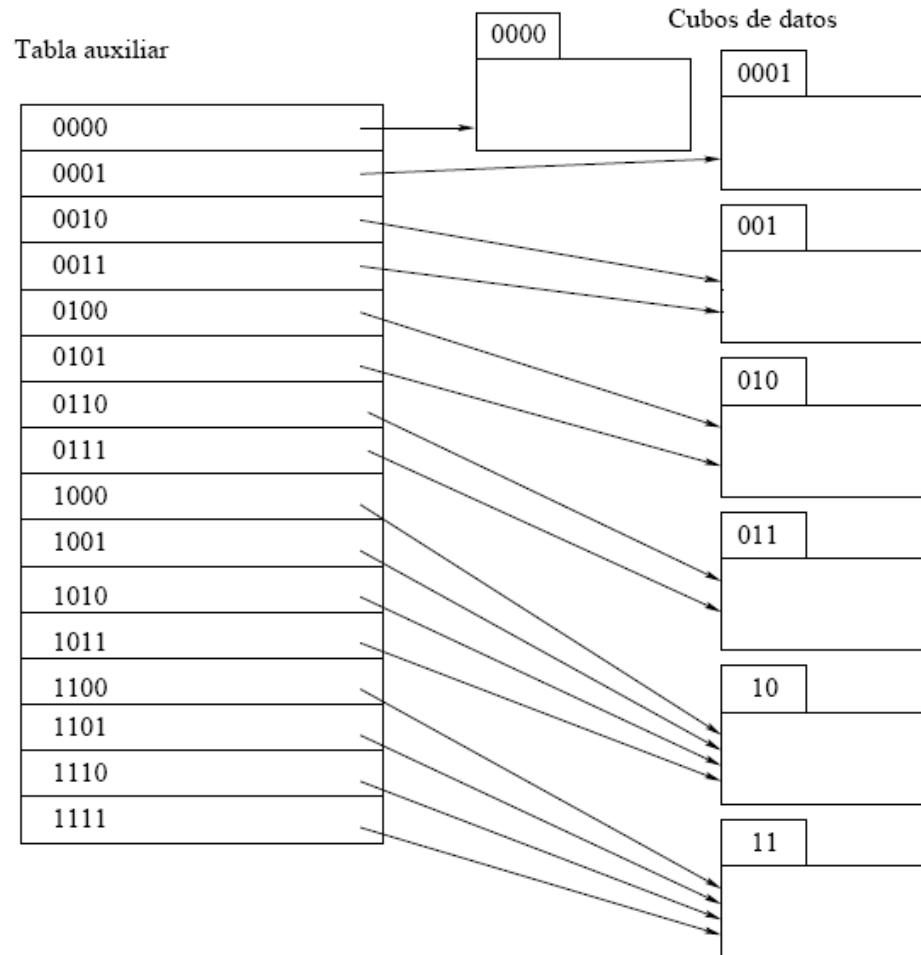
Hashing dinámico

- Algoritmo

- Se considera una tabla índice en memoria con 2^d filas.
- En la primera columna de esta tabla se sitúan todas las posibles sucesiones de d dígitos binarios :
 - d es la “profundidad global” de la tabla
- Todos los cubos tienen en principio “profundidad local” igual a b :
 - En principio, todas las entradas cuyos b primeros* dígitos son iguales apuntan al mismo cubo.
 - Allí se almacenan los registros cuyo valor de k' tiene esos b primeros dígitos.
- Cuando se llena un cubo se divide en 2, poniendo en uno de ellos los registros con el dígito $b+1$ de k' a 0 y en otro los que lo tienen igual a 1.
 - Este proceso aumenta la profundidad local de los cubos generados, que está limitada por la profundidad global de la tabla. Si se alcanza el límite, antes de dividir habría que aumentar la profundidad global (si es posible).



Hashing dinámico

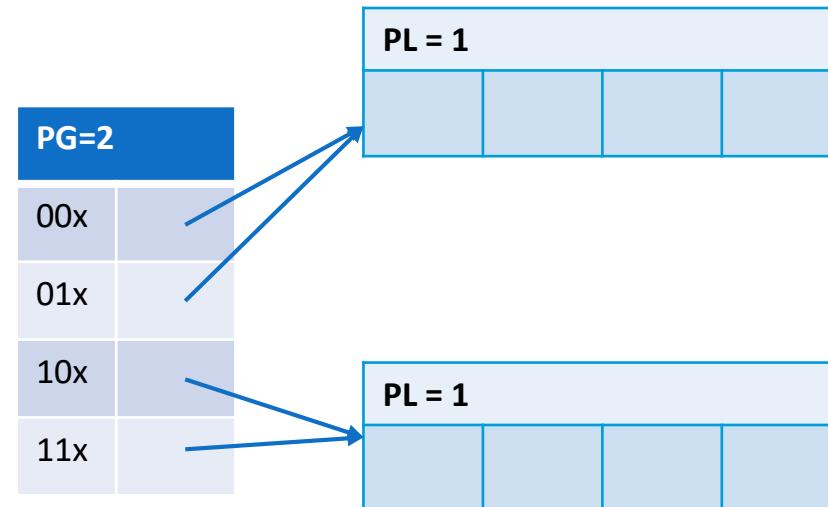


Hashing dinámico

- Ejemplo

- Inicialmente:

- Tabla auxiliar de 2 bits de profundidad global (PG=d).
 - $b=1$; Por tanto 2 cubos con profundidad local (PL) 1.
 - 4 registros por cubo.
 - $h(k)=k \bmod 8$
 - Tomamos los d primeros bits para direccionar.

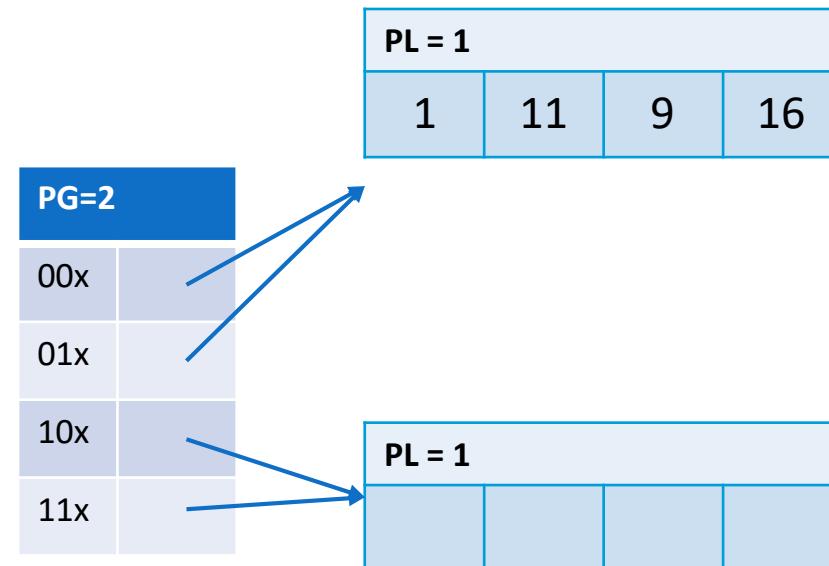


Hashing dinámico

- Insertamos: 1, 11, 9, 16, 3, 32, 15, 4, 2, 19, 7, 14, 17, 27



clave	H(k)	Binario
1	1	001
11	3	011
9	1	001
16	0	000
3	3	011
32	0	000
15	7	111
4	4	100
2	2	010
19	3	011
7	7	111
14	6	110
17	1	001
27	3	011

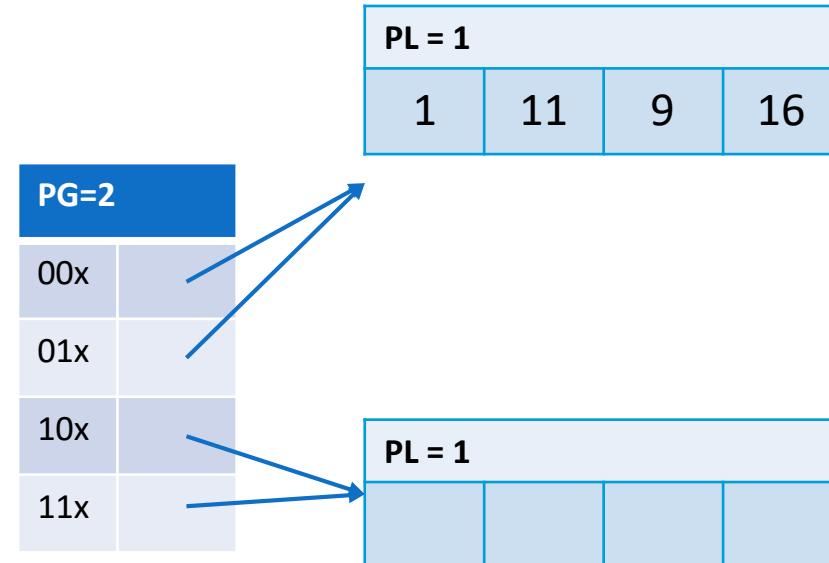


Hashing dinámico

- Insertamos: 1, 11, 9, 16, 3, 32, 15, 4, 2, 19, 7, 14, 17, 27



clave	H(k)	Binario
1	1	001
11	3	011
9	1	001
16	0	000
3	3	011
32	0	000
15	7	111
4	4	100
2	2	010
19	3	011
7	7	111
14	6	110
17	1	001
27	3	011

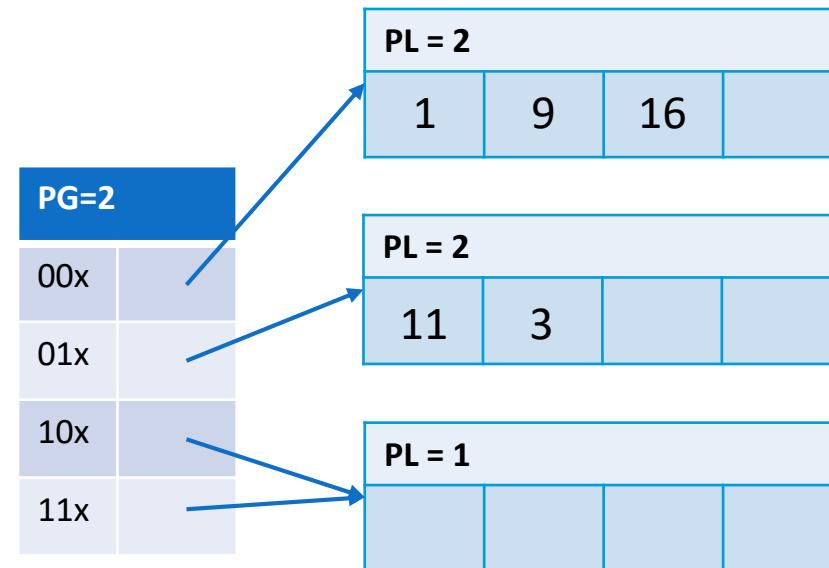


Hashing dinámico

- Insertamos: 1, 11, 9, 16, 3, 32, 15, 4, 2, 19, 7, 14, 17, 27



clave	H(k)	Binario
1	1	001
11	3	011
9	1	001
16	0	000
3	3	011
32	0	000
15	7	111
4	4	100
2	2	010
19	3	011
7	7	111
14	6	110
17	1	001
27	3	011

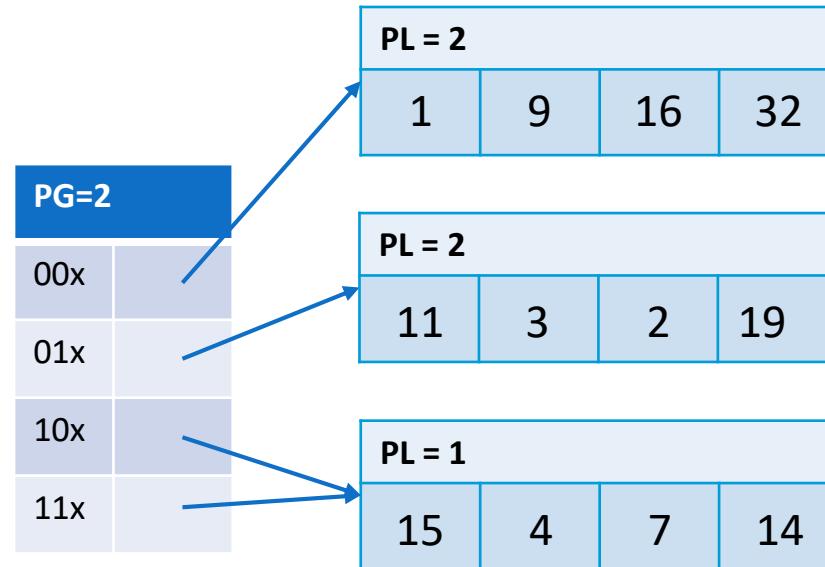


Hashing dinámico

- Insertamos: 1, 11, 9, 16, 3, 32, 15, 4, 2, 19, 7, 14, 17, 27



clave	H(k)	Binario
1	1	001
11	3	011
9	1	001
16	0	000
3	3	011
32	0	000
15	7	111
4	4	100
2	2	010
19	3	011
7	7	111
14	6	110
17	1	001
27	3	011

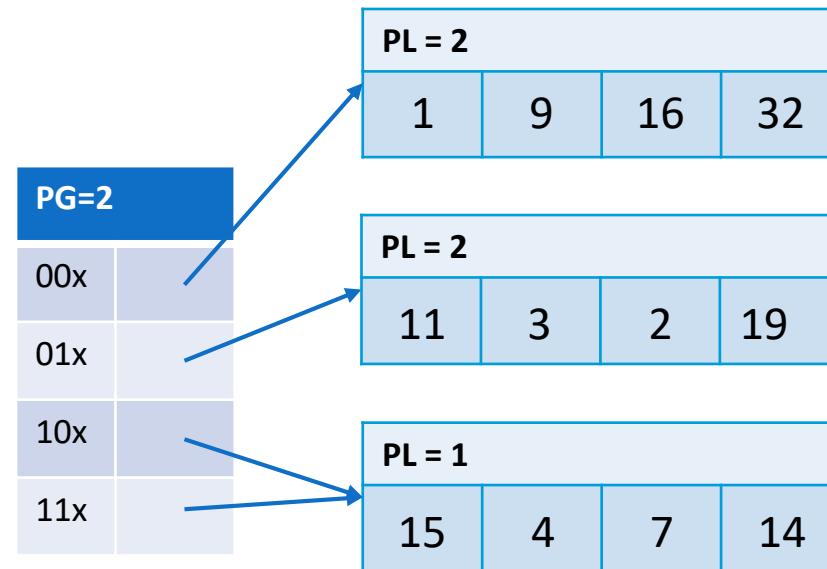


Hashing dinámico

- Insertamos: 1, 11, 9, 16, 3, 32, 15, 4, 2, 19, 7, 14, 17, 27



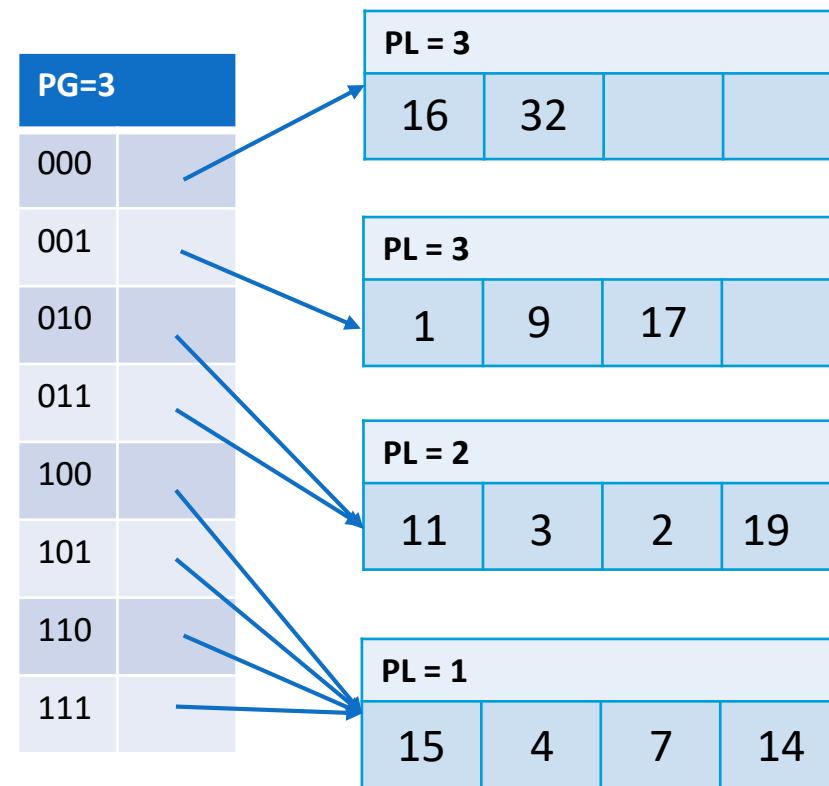
clave	H(k)	Binario
1	1	001
11	3	011
9	1	001
16	0	000
3	3	011
32	0	000
15	7	111
4	4	100
2	2	010
19	3	011
7	7	111
14	6	110
17	1	001
27	3	011



Hashing dinámico

- Insertamos: 1, 11, 9, 16, 3, 32, 15, 4, 2, 19, 7, 14, 17, 27

clave	H(k)	Binario
1	1	001
11	3	011
9	1	001
16	0	000
3	3	011
32	0	000
15	7	111
4	4	100
2	2	010
19	3	011
7	7	111
14	6	110
17	1	001
27	3	011

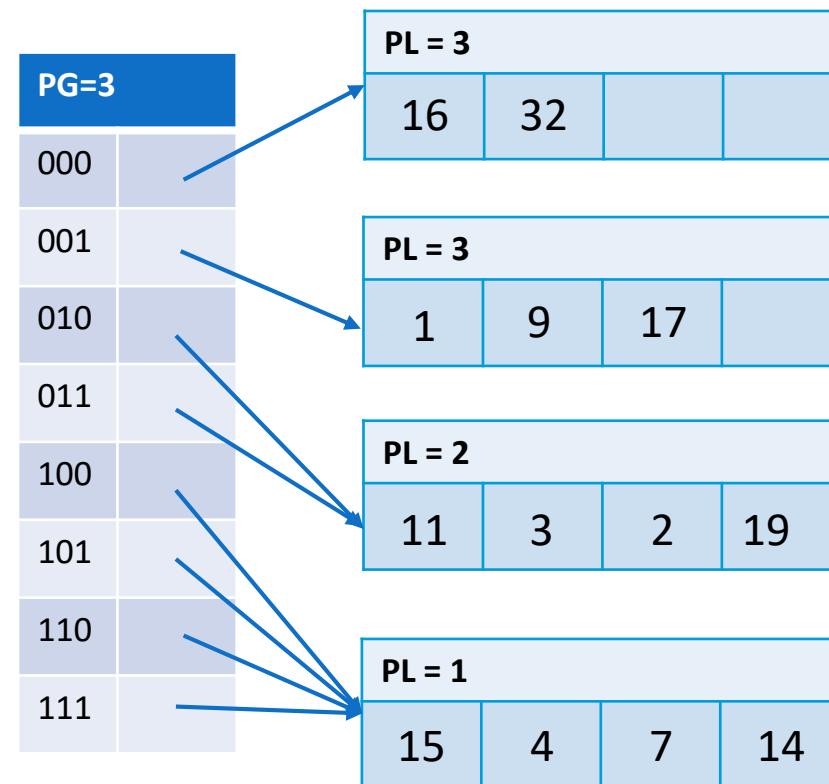


Hashing dinámico

- Insertamos: 1, 11, 9, 16, 3, 32, 15, 4, 2, 19, 7, 14, 17, 27



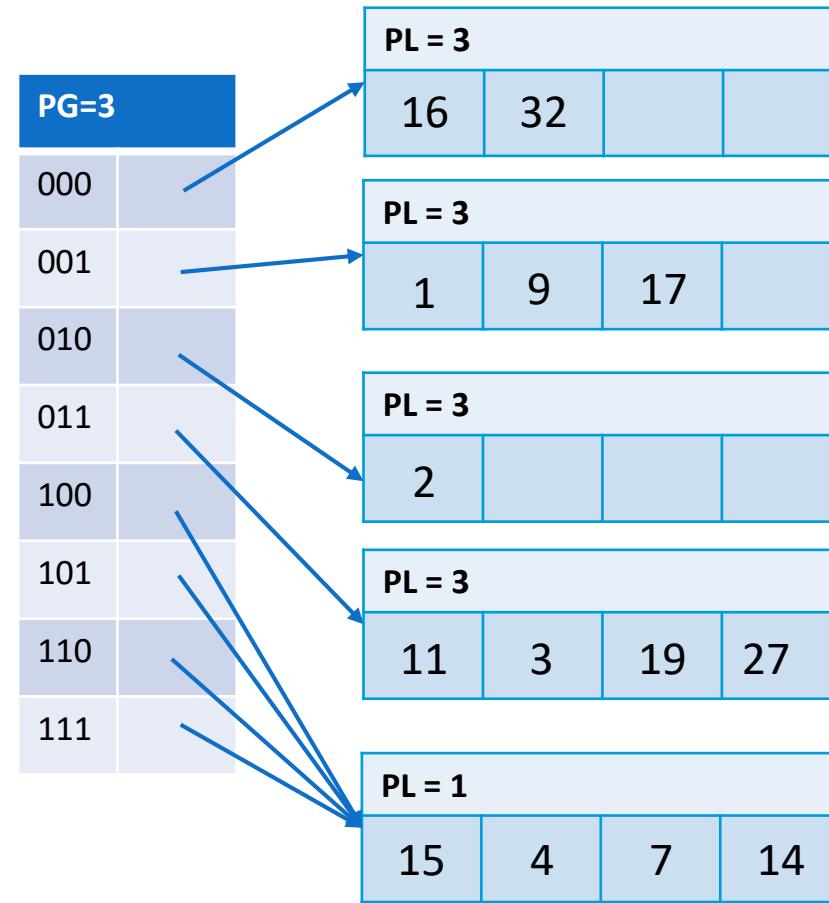
clave	H(k)	Binario
1	1	001
11	3	011
9	1	001
16	0	000
3	3	011
32	0	000
15	7	111
4	4	100
2	2	010
19	3	011
7	7	111
14	6	110
17	1	001
27	3	011



Hashing dinámico

- Insertamos: 1, 11, 9, 16, 3, 32, 15, 4, 2, 19, 7, 14, 17, 27

clave	H(k)	Binario
1	1	001
11	3	011
9	1	001
16	0	000
3	3	011
32	0	000
15	7	111
4	4	100
2	2	010
19	3	011
7	7	111
14	6	110
17	1	001
27	3	011



Hashing dinámico

- El hashing dinámico supera los problemas clásicos del acceso directo.
- También tiene sus inconvenientes:
 - Utilizar una tabla índice adicional (nuevos accesos a disco si no cabe en memoria).
 - El tamaño de la tabla depende de “d”.

Contenidos

- Acceso directo
- Hashing básico
- Hashing dinámico

Imágenes

- Imágenes tomadas de [Pixabay](#)
 - Portada
 - Imagen de [Manfred Steger](#)
 - Cabecera
 - Imágenes de [Gerd Altmann](#)