

# Fundamentos de Organización de Datos

Clase 6

# Agenda

## Arboles

- Binarios
- AVL
- Multicamino
- Balanceados

## Arboles Balanceados

- Características
- B, B\*, B+
- Operaciones
- Prefijos simples

# Arboles → introducción

## Problemas con los índices?

- La búsqueda binaria aun es costosa
- Mantener los índices ordenados es costoso
- Solución → RAM
- Objetivo → persistencia de datos

## Árboles

- Estructuras de datos que permiten localizar en forma más rápida información de un archivo, tienen intrínsecamente búsqueda binaria

# Arboles binarios

## Que es un árbol binario?

- Estructuras de datos donde cada nodo tiene a lo sumo dos sucesores, a izquierda y a derecha

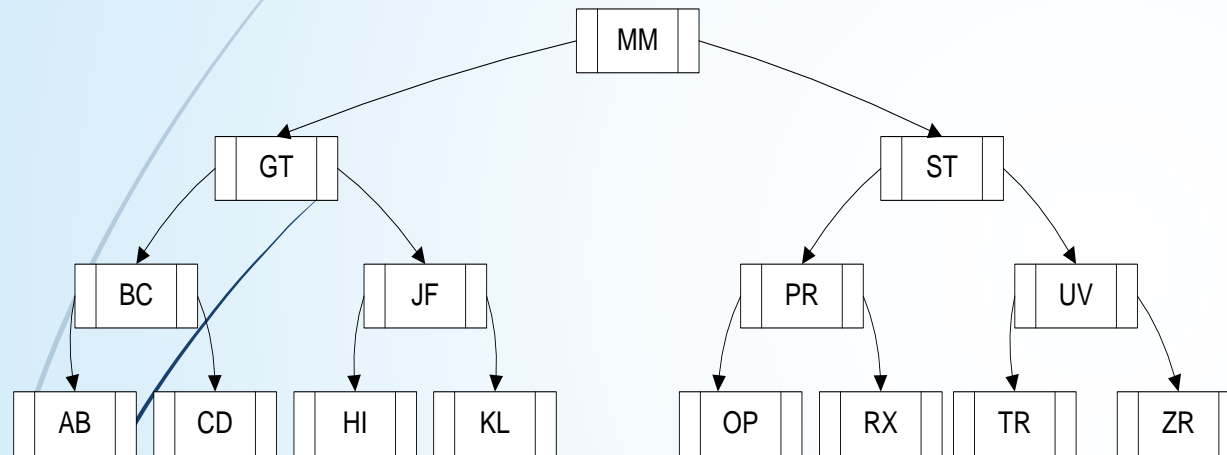
## Un árbol binario, puede implantarse en disco?

- Como lograr la persistencia?

## Ejemplo → supongamos estas claves

- MM ST GT PR JF BC UV CD HI AB KL TR OP RX ZR

# Arboles binarios



Raíz → 0

	Clave	Hijo izq	Hijo Der
0	MM	1	2
1	GT	3	4
2	ST	8	11
3	BC	5	6
4	JF	7	14
5	AB	-1	-1
6	CD	-1	-1
7	HI	-1	-1

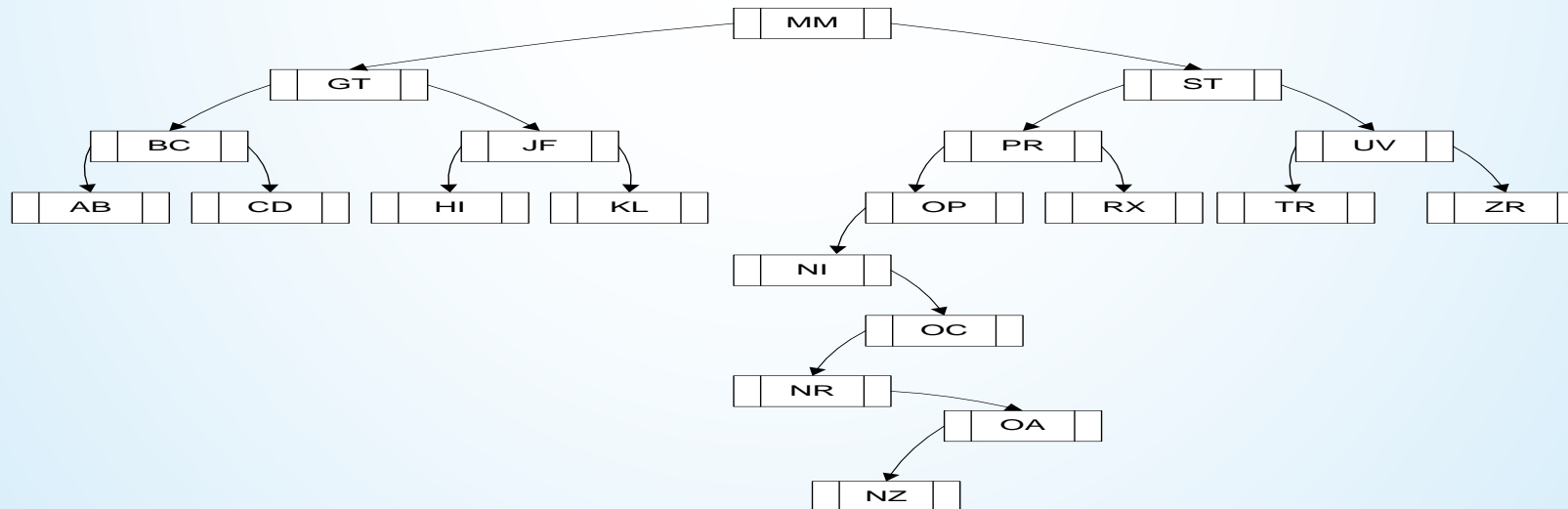
	Clave	Hijo izq	Hijo Der
8	PR	9	10
9	OP	-1	-1
10	RX	-1	-1
11	UV	12	13
12	TR	-1	-1
13	ZR	-1	-1
14	KL	-1	-1

# Arboles binarios

Árbol balanceado: un árbol está balanceado cuando la altura de la trayectoria más corta hacia una hoja no difiere de la altura de la trayectoria más grande.

Inconveniente de los binarios: se **desbalancean** fácilmente.

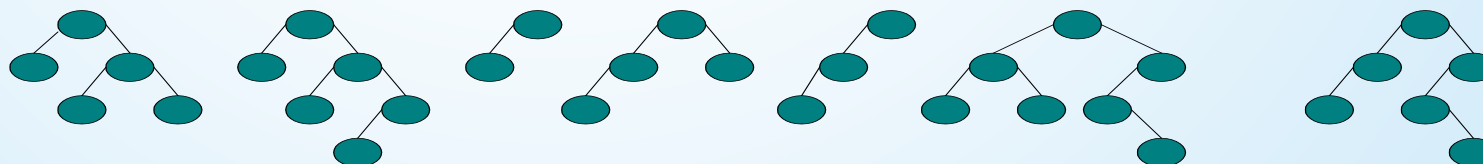
Supongamos que llegan las claves : NI OC NR OA NZ



# Árboles AVL

## Árboles AVL

- Árbol binario balanceado en altura (BA(1)) en el que las inserciones y eliminaciones se efectúan con un mínimo de accesos.
- Árbol balanceado en altura:
  - Para cada nodo existe un límite en la diferencia que se permite entre las alturas de cualquiera de los subárboles del nodo (BA(k)), donde k es el nivel de balance)
  - Ejemplos:



# Arboles AVL y Binarios

## Características/Conclusiones

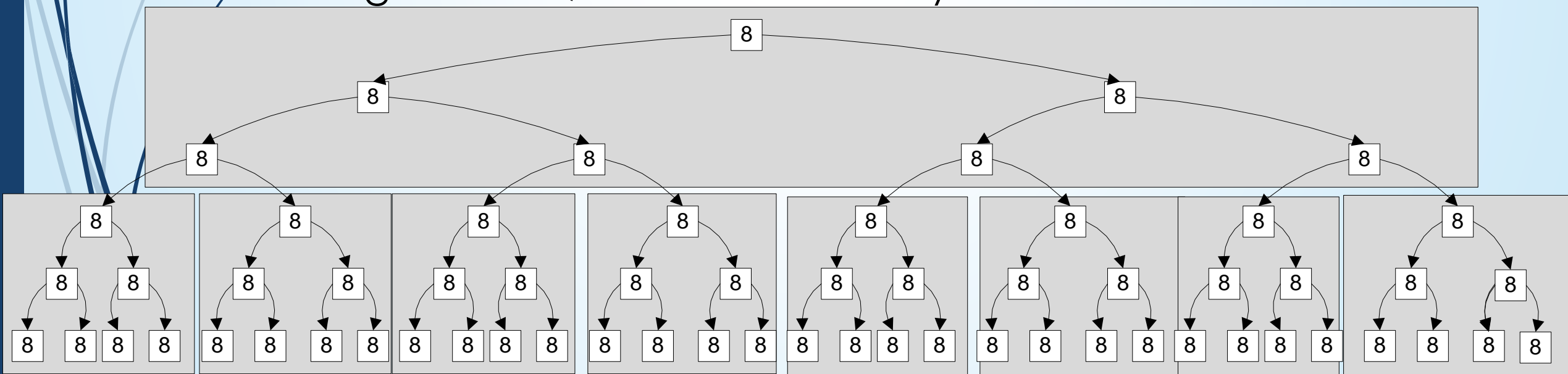
- Estructura que debe ser respetada
- Mantener árbol, rotaciones restringidas a un área local del árbol
  - Binario: → Búsqueda:  $\log_2(N+1)$
  - AVL: → Búsqueda:  $1.44 \log_2(N+2)$
  - Ambas performance por el peor caso posible



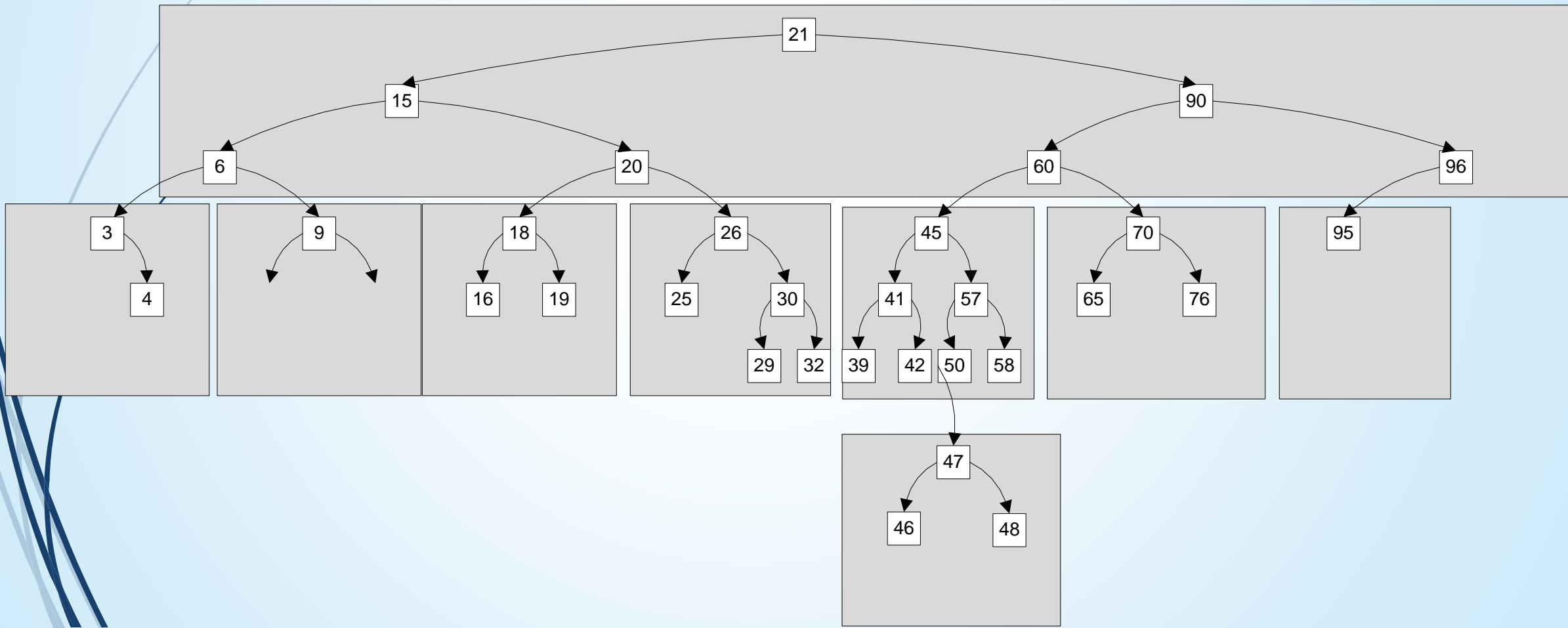
# Árboles Binarios Paginados

## Árboles binarios paginados

- Problemas de almacenamiento secundario, buffering, páginas de memoria, varios registros individuales, minimiza el número de accesos
- Problema: construcción descendente, como se elige la raíz?, cómo va construyendo balanceado?



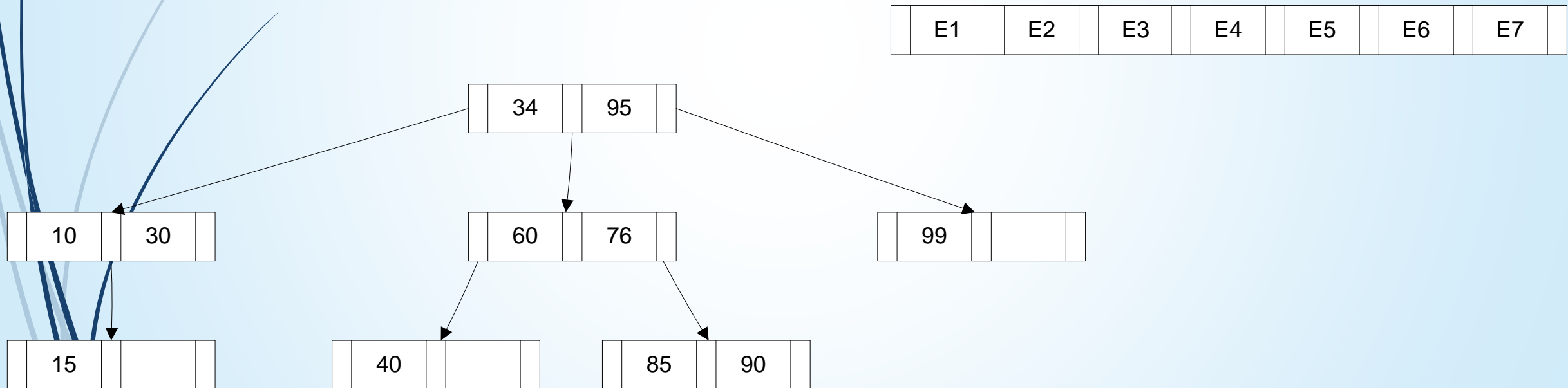
# Árboles Binarios Paginados



# Árboles multcamino

Generalización de árboles binarios, c/nodo tiene  $k$  punteros y  $k-1$  claves (o registros), disminuye la profundidad del árbol,

- Orden del árbol.



# Arboles balanceados

Son árboles multicamino con una construcción especial en forma ascendente que permite mantenerlo balanceado a bajo costo.

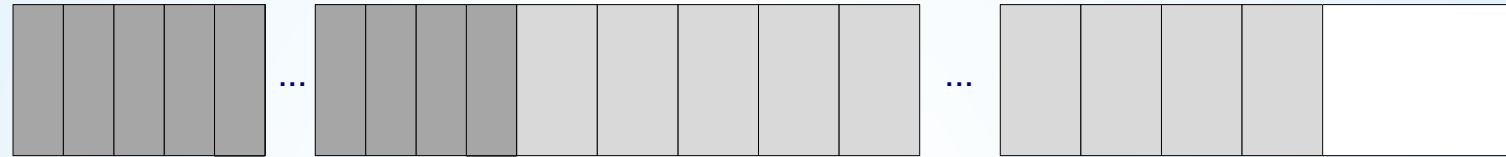
Propiedades de un árbol B de orden  $M$ :

- Ningún nodo tiene más de  $M$  hijos
- C/nodo (menos raíz y los terminales) tienen como mínimo  $\lceil M/2 \rceil$  hijos
- La raíz tiene como mínimo 2 hijos (o sino ninguno)
- Todos los nodos terminales a igual nivel
- Nodos no terminales con  $K$  hijos contienen  $K-1$  registros. Los nodos terminales tienen:
  - Mínimo  $\lceil M/2 \rceil - 1$  registros
  - Máximo  $M - 1$  registros

PO	R1	P1	R2	P2	R3	P3	R4	P4	R5	P5	Nro de registros
----	----	----	----	----	----	----	----	----	----	----	------------------

# Arboles Balanceados

## Formato del nodo

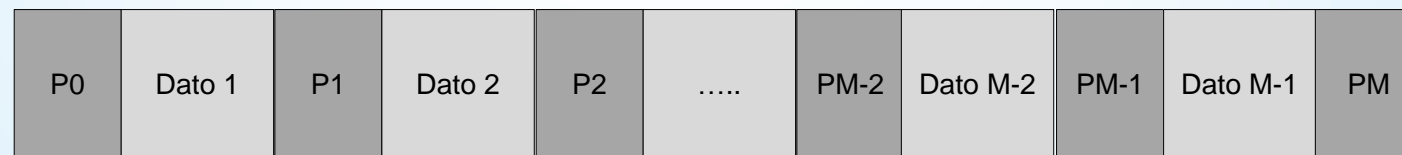


Hijos (M celdas)

Datos ( M -1 celdas)

Nro de Registros

## Formato del Nodo para archivo del índice arbol b



## Formato Gráfico del Nodo del índice arbol B

# Arboles balanceados

## Creacion:

- Dadas las claves: 43 2 53 88 75 80  
15 49 60 20 57 24
- Como se construye el árbol?
- Como se genera el archivo de datos que persiste el árbol?

# Arboles Balanceados

Nodo Raiz: 7								
	Punteros				Datos			Nro Datos
0	-1	-1	-1		2	15		2
1	-1	-1	-1	-1	57	60	75	3
2	0	5	3		20	43		2
3	-1	-1			49			1
4	-1	-1			88			1
5	-1	-1			24			1
6	1	4			80			1
7	2	6			53			1

**HEA!** Herramienta de Software para enseñanza de árboles B

Tipo de Árbol

ARBOL B

Orden

4

CREAR

Nº a insertar

+

24

>>

Nº a eliminar

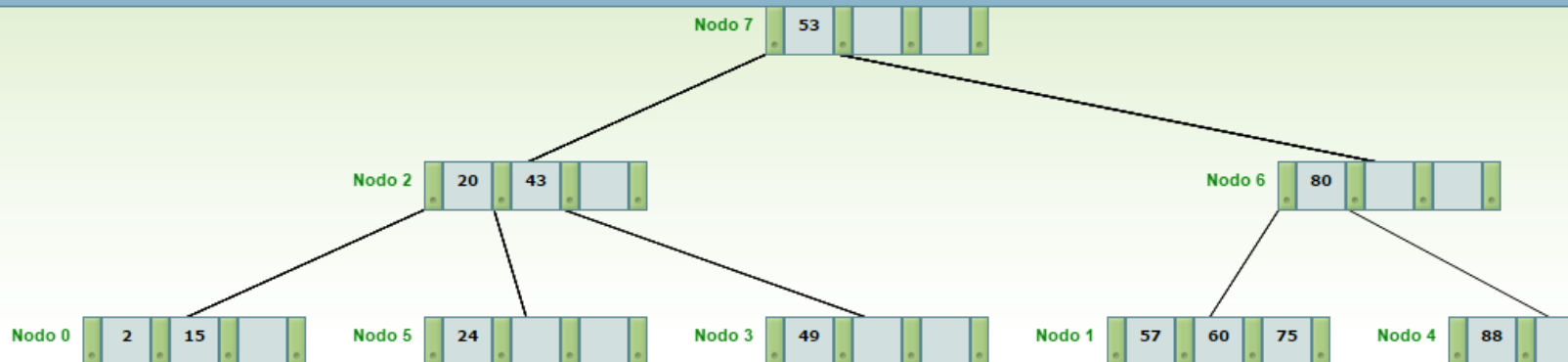
-

>>

Nº a buscar

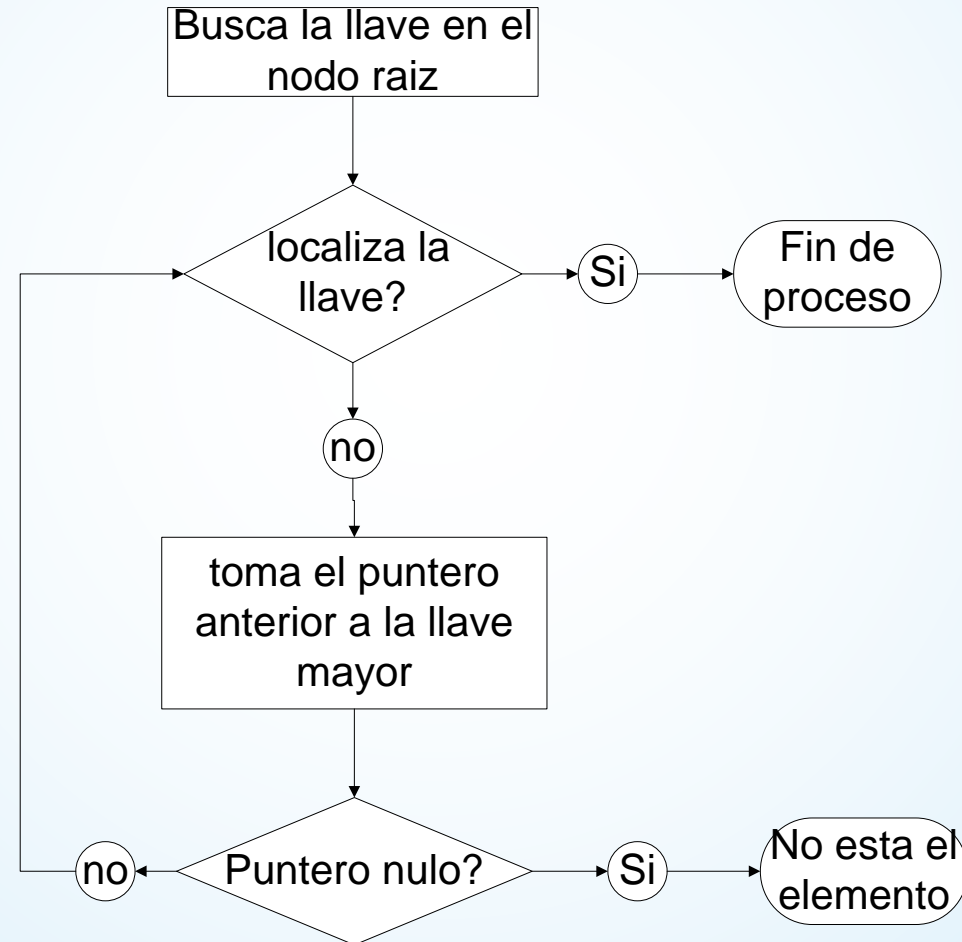
BUSCAR

LIMPIAR



# Árboles Balanceados

- Búsqueda de información:





# Arboles Balanceados

## Performance de búsqueda

- Mejor caso: 1 lectura
- Pero caso:  $h$  lecturas (con  $h$  altura del árbol)
- Cual es el valor de  $h$ ?
  - Axioma: árbol balanceado de Orden  $M$ , si el número de elementos del árbol es  $N \rightarrow$  hay  $N+1$  punteros nulos en nodos terminales.

# Árboles Balanceados

Cota  
para  
h

Nivel	# mínimo de descendientes
1	2
2	$2 * [M/2]$
3	$2 * [M/2] * [M/2]$
.....	
h	$2 * [M/2]^{h-1}$

**Relación entre h y # de nodos**

$$N+1 \geq 2 * [M/2]^{h-1}$$

$$h \leq [ 1 + \log_{[M/2]} ((N+1)/2) ]$$

**Si  $M = 512$  y  $N = 1000000 \rightarrow h \leq 3.37$  (4 lecturas encuentra un registro)**

# Árboles Balanceados

## Inserción (creación)

- Los registros se insertan en un nodo Terminal
- Casos posibles
  - El registro tiene lugar en el nodo Terminal (no se produce overflow): solo se hacen reacomodamientos internos en el nodo
  - El registro no tiene lugar en el nodo Terminal (se produce overflow): el nodo se divide y los elementos se reparten entre los nodos, hay una promoción al nivel superior, y esta puede propagarse y generar una nueva raíz.

# Árboles Balanceados

## Performance de la inserción

- Mejor caso (sin overflow)
  - $H$  lecturas
  - 1 escritura
- Peor caso (overflow hasta la raíz, aumenta en uno el nivel del árbol)
  - $H$  lecturas
  - $2h+1$  escrituras (dos por nivel más la raíz)
- Estudios realizados
  - $M = 10$     25% divisiones
  - $M = 100$     2% divisiones

# Árboles Balanceados

## Eliminación

- Siempre eliminar de nodos terminales (trabajamos con árboles)
- Si se va a eliminar un elemento que no está en nodo terminal → llevarlo primero a nodo terminal
- Posibilidades ante eliminación
  - Mejor caso: borra un elemento del nodo y no produce underflow, solo reacomodos ( # elementos  $\geq [M/2]-1$ )
  - Peor caso: se produce underflow, #elementos  $< [M/2] - 1$
- Dos soluciones
  - Redistribuir
  - concatenar

# Árboles Balanceados

## Definición: nodo adyacente hermano

- Dos nodos son adyacentes hermanos si tienen el mismo padre y son apuntados por punteros adyacentes en el padre.

## Redistribuir

- Cuando un nodo tiene underflow puede trasladarse llaves de un nodo adyacente hermano (en caso que este tenga suficientes elementos)

## Concatenación:

- Si un nodo adyacente hermano está al mínimo (no le sobra ningún elemento) no se puede redistribuir, se concatena con un nodo adyacente disminuyendo el # de nodos (y en algunos casos la altura del árbol)

# Árboles Balanceados

## Performance de la eliminación

- Mejor caso (borra de un nodo Terminal)
  - $H$  lecturas
  - 1 escritura
- Peor caso (concatenación lleva a decrementar el nivel del árbol en 1)
  - $2h - 1$  lecturas
  - $H + 1$  escrituras

# Árboles Balanceados $\rightarrow B^*$

## Eliminación

- Redistribución
- Concatenación

## Inserción

- ???????
- División



## Árboles Balanceados $\rightarrow B^*$

La redistribución podría posponer la creación de páginas nuevas

Se pueden generar árboles B más eficientes en términos de utilización de espacio

# Árboles Balanceados $\rightarrow B^*$

---

Árbol B especial en que cada nodo está lleno por lo menos en  $2/3$  partes

---

## Propiedades (orden $M$ )

Cada página tiene máximo  $M$  descendientes

Cada página, menos la raíz y las hojas, tienen al menos  $\lceil (2M - 1) / 3 \rceil$  descendientes

La raíz tiene al menos dos descendientes (o ninguno)

Todas las hojas aparecen en igual nivel

Una página que no sea hoja si tiene  $K$  descendientes contiene  $K-1$  llaves

Una página hoja contiene por lo menos  $\lceil (2M - 1) / 3 \rceil - 1$  llaves, y no más de  $M-1$ .

---

# Árboles Balanceados $\rightarrow B^*$

## Operaciones de Búsqueda

- Igual que el árbol B común

## Operaciones de Inserción

- Tres casos posible
  - **Derecha:** redistribuir con nodo adyacente hermano de la derecha (o izq. si es el último)
  - **Izquierda:** redistribuir con nodo adyacente hermano de la izquierda (o der. si es el último)
  - **Izquierda o derecha:** si el nodo de la derecha está lleno se busca redistribuir con la izquierda ,y viceversa.
  - **Izquierda y derecha:** busca llenar los tres nodos, estos tendrán un  $\frac{3}{4}$  parte llena.

# Árboles Balanceados $\rightarrow B^*$

## Costo de la redistribución

	Mejor	Peor
Derecha	RRWW	RRWWWW
Izq o der	RRWW	RRRWWWW (divido solo dos)
Izq y der	RRWW	RRRWWWWW

# Árboles Balanceados

---

Técnicas de paginado    estrategias de reemplazo: LRU (last recently used)

---

Análisis numérico	# llaves = 2400    # páginas = 140    Altura = 3 niveles			
	1	5	10	20
	3.00	1.71	1.42	0.97

---

# Árboles Balanceados

Archivos  
secuenciales  
indizados

Permiten una mejor  
recorrida por algún  
tipo de orden

Indizado (ordenado por una llave)

Secuencial (acceder por orden físico,  
devolviendo el registro en orden de llave)

Hasta ahora métodos  
disjuntos, se opta:

rápida recuperación (Árbol)

Recuperación ordenada (secuencial)

Debemos encontrar  
una solución que  
agrupe ambos casos

# Árboles Balanceados

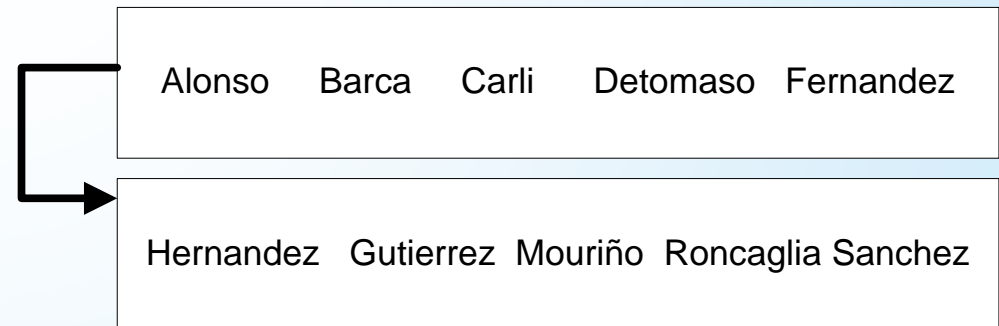
## Conjunto de secuencias

- Conjunto de registros que mantienen un orden físico por llave mientras que se agregan o quitan datos, si podemos mantenerlo podemos indizarlos

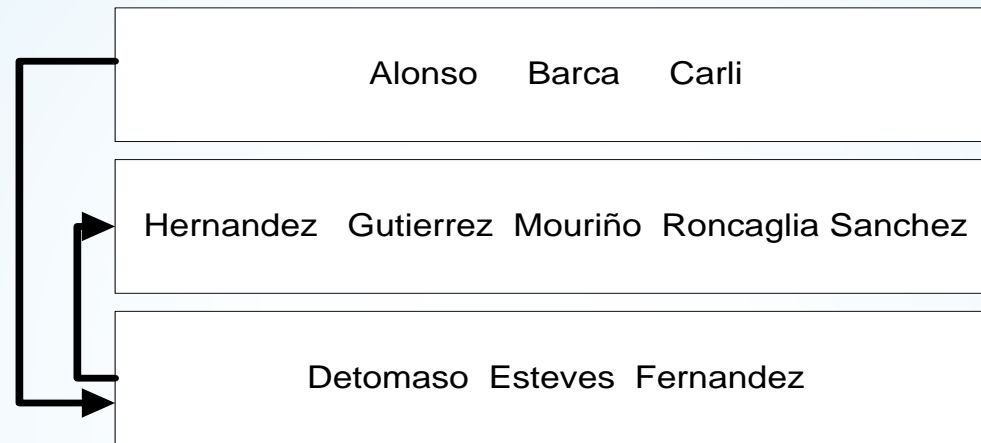
## Posible solución

- Mantener bloques de datos
- Cada bloque con registros y puntero al siguiente

Alonso   Barca   Carli   Detomaso   Fernandez



# Árboles Balanceados



## Costo

- Aumenta el tamaño del archivo (fragmentación interna)
- No hay orden físico salvo dentro del un bloque.
- Tamaño del bloque
  - Debe permitir almacenar varios bloques en RAM (redistribución)
  - Las E/S deben ser rápidas y sin necesidad de desplazamientos
- Como logramos ahora una rápida búsqueda?



# Árboles Balanceados → B+

Consiste en un conjunto de grupos de registros ordenados por clave en forma secuencial, junto con un conjunto de índices, que proporciona acceso rápido a los registros.

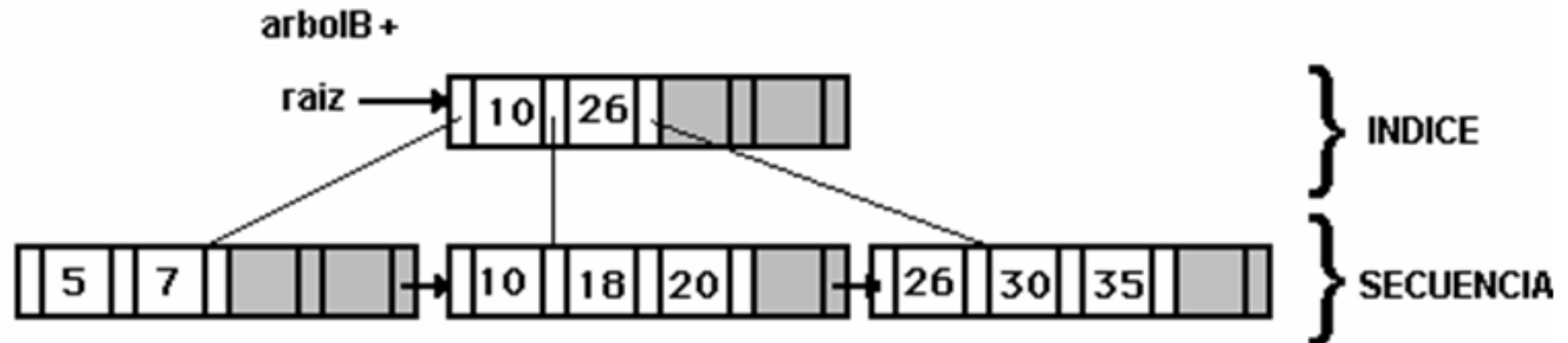
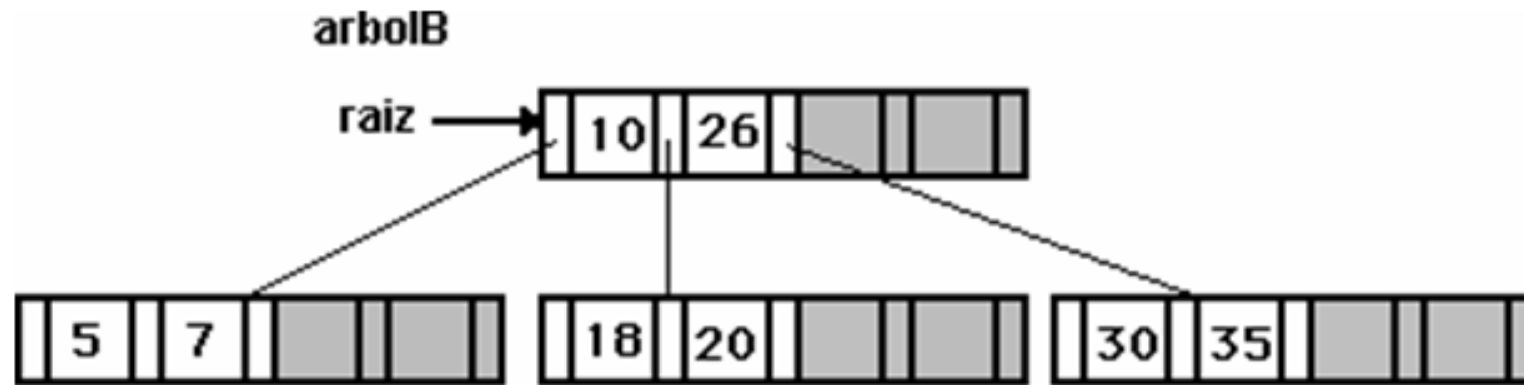
## Propiedades

- Cada página tiene máximo  $M$  descendientes
- Cada página, menos la raíz y las hojas, tienen entre  $[M/2]$  y  $M$  hijos
- La raíz tiene al menos dos descendientes (o ninguno)
- Todas las hojas aparecen en igual nivel
- Una página que no sea hoja si tiene  $K$  descendientes contiene  $K-1$  llaves
- Los nodos terminales representan un conjunto de datos y son linkeados juntos.

Los nodos no terminales no tienen datos sino punteros a los datos.

# Árboles B+

## ■ Ejemplo

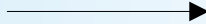


# Árboles Balanceados → B+

Nodo Raíz



Nodo Inicial



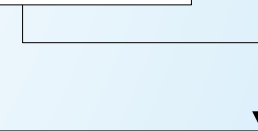
Alfa Beta Kappa Phi

Se inserta Gamma

Nodo Raíz



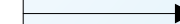
Gamma



Nodo Inicial



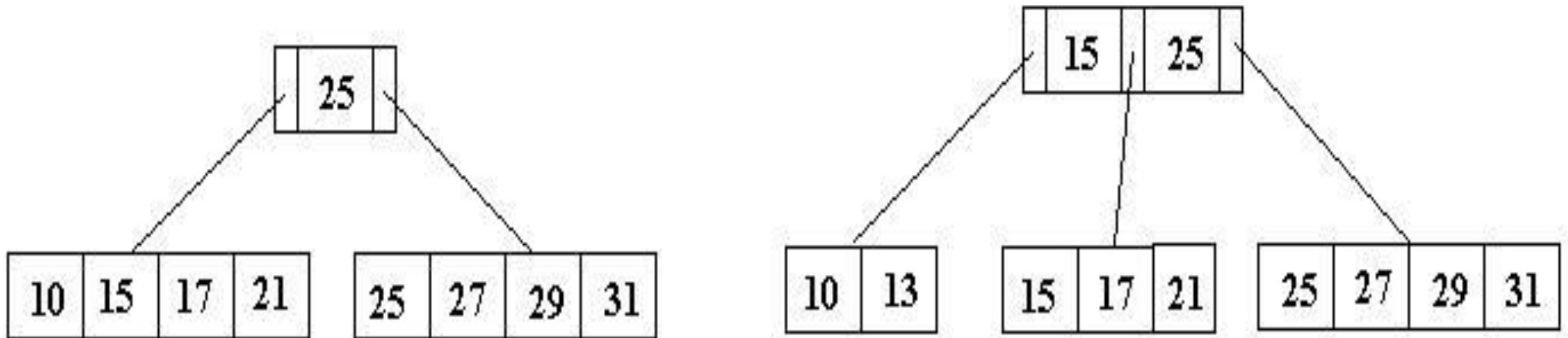
Alfa Beta



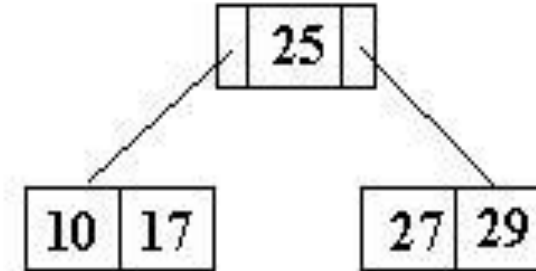
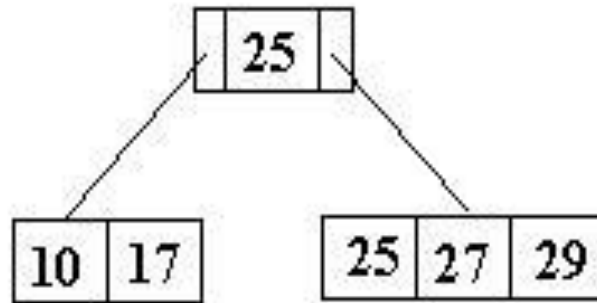
Gamma Kappa Phi

# Arboles B+

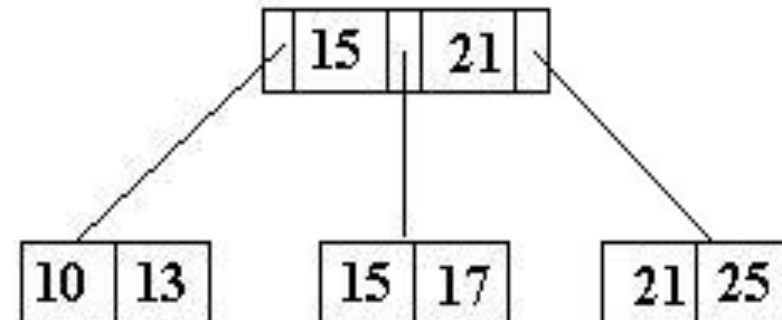
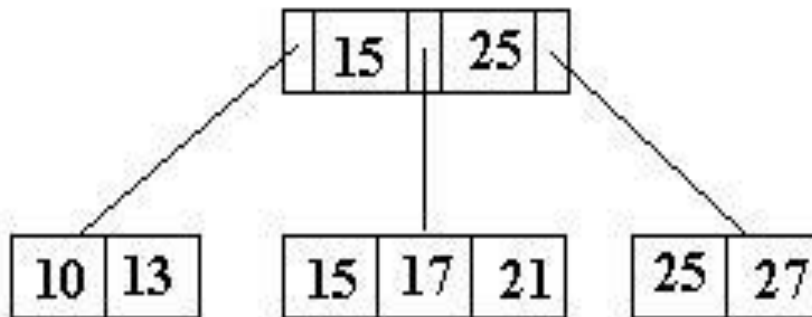
- ➔ Inserción clave 13 ( $M=5$ )



# Arboles B+

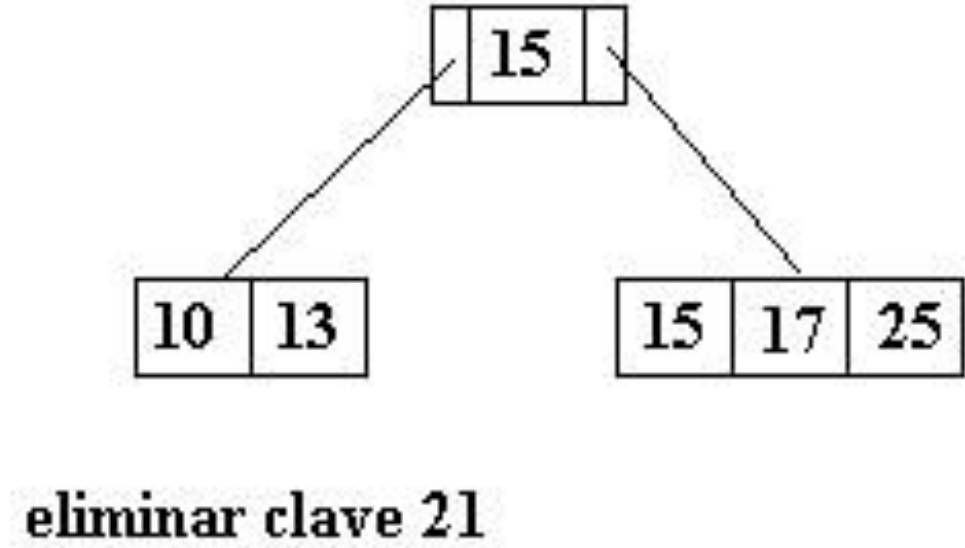
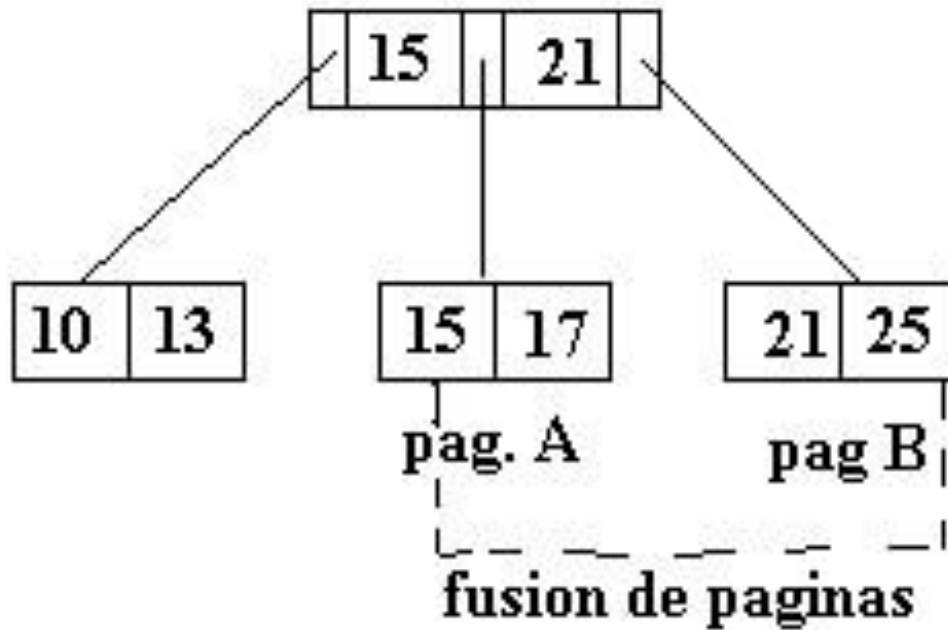


eliminar clave 25



eliminar clave 27

# Arboles B+



# Árboles Balanceados $\rightarrow$ B+

## Separadores

- Derivados de las llaves de los registros que limitan un bloque en el conjunto de secuencia
- Separadores más cortos, ocupan espacio mínimo

## Árbol B+ de prefijos simples

- Árbol B+ en el cual el conjunto índice está constituido por separadores más cortos

# Árboles Balanceados → B+

Nodo Raíz

G

Alfa Beta

Gamma Kappa Phi

Nodo Inicial

Nodo Raíz

Gon

Garcia Gomez

Gonzalez Gutierrez Hernandez

Nodo Inicial



# Árboles Balanceados → conclusiones

	Árbol B	Árbol B+
Ubicación de datos	Nodos (cualquiera)	Nodo Terminal
Tiempo de búsqueda	=	=
Procesamiento secuencial	Lento (complejo)	Rápido (con punteros)
Inserción eliminación	Ya discutida	Puede requerir + tiempo