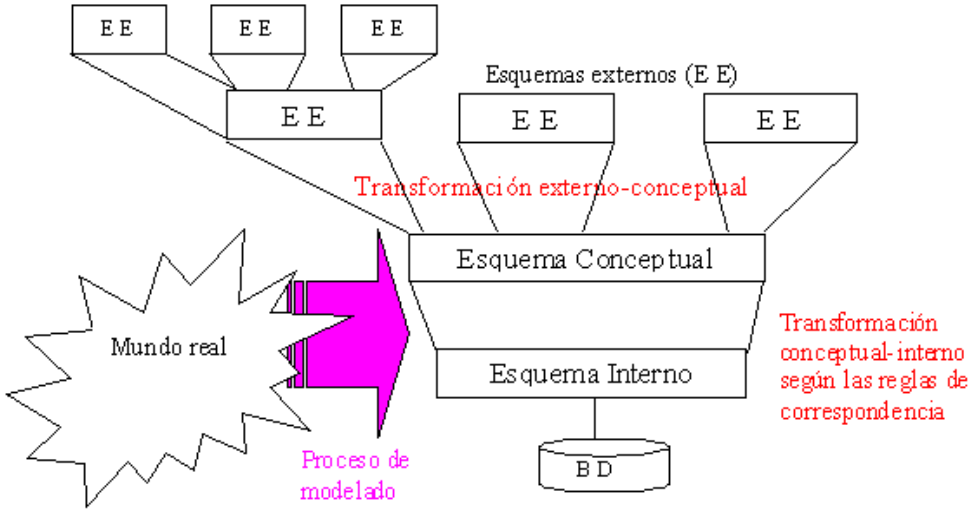


UNIVERSIDAD DE LOS ANDES

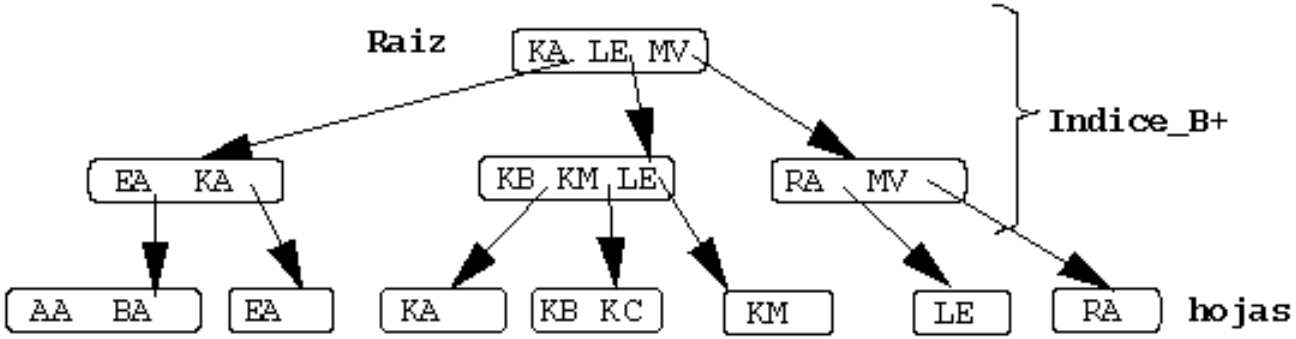
Bases de datos Unidad 1

Universidad de Los Andes
Escuela de Ingeniería de Sistemas
Departamento de Computación

Tema 1. Los sistemas de gestión de archivos

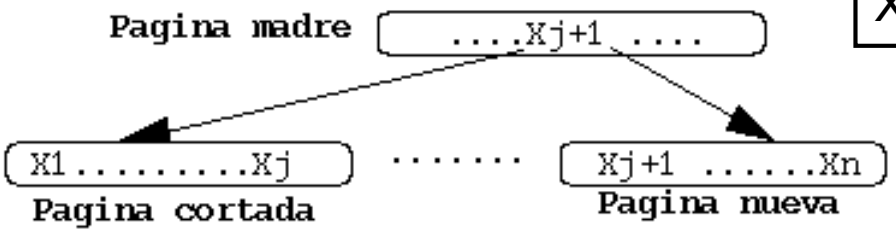
Árbol_B+

- **Mejora del árbol_B en lo que respecta al tratamiento secuencial de las claves en el índice y en la separación lógica del índice_B+ del archivo de datos que está indizando**



Operaciones:

- Consulta
- Inserción
- Eliminación



X0 P0	X1 P1	X2 P2	.....	Xn Pn

Especificación del TAD Arbol_B+ (h,k)

17/4/98

Especificación Arbol_B+

1 **Sintáctica:**

creaArbol_B+(Entero+) Arbol_B+,
insArbol_B+(Arbol_B+, Reg) Arbol_B+,
eliArbol_B+(Arbol_B+, TipoClave) Arbol_B+,
conArbol_B+(Arbol_B+, TipoClave) TipoResto,
rango(Arbol_B+, TipoClave, TipoClave) ListaEnl[Reg],
orden(Arbol_B+) Entero+,
altura(Arbol_B+) Entero+,
vacíoArbol_B+(Arbol_B+) Lógico,
destruyeArbol_B+(Arbol_B+) .

2 **Declaraciones:**

Reg: r
Entero+ : o
TipoClave: c
TipoResto: {TipoRestoNoDef}

3 **Semántica:**

vacíoArbol_B+(creaArbol_B+(o)) = Verdadero
vacíoArbol_B+(insArbol_B+(creaArbol_B+(o), r)) = Falso
conArbol_B+(creaArbol_B+(o), c) = {TipoRestoNoDef}
eliArbol_B+(creaArbol_B+(o)) = creaArbol_B+(o)
altura(creaArbol_B+(o)) = 0
orden(creaArbol_B+(o)) = o

-creaArbol_B+(): Crea un árbol_B+ vacío de orden m.

-insArbol_B+(): Ingresa un nuevo registro al árbol_B+ en la posición que le corresponde según el orden ascendente de su clave.

-eliArbol_B+(): Elimina el registro de clave dada en el árbol_B+, si existe. Si está vacío no hace ninguna eliminación.

-destruyeArbol_B+(): Destruye el árbol_B+.

-vacíoArbol_B+(): Regresa Verdadero si el árbol_B+ está vacío.

-conArbol_B+(): Devuelve el resto del registro si él está en el árbol_B+, de lo contrario regresa {TipoRestoNoDef}.

-rango(): Devuelve una lista de registros que están en el rango solicitado.

-orden(): Regresa el orden del árbol_B+.

-altura(): Regresa la altura actual.

Implementación del TAD Árbol_B+ (h,k)

	Arbol_B+ Clases: Entero+, Lógico, ApuntadorA, NodoB, TipoClave, TipoResto, Reg, ElePila, PilaEnl[ElePila]	
1	**Estructura**: privado: h : Entero+ = 0 nc : Entero+ = 0 m : Entero+ raiz : ApuntadorA NodoB = Nulo	**-h.** Altura actual del árbol_B+ -**nc**. Número actual de claves en el árbol. **-m.** Orden del árbol_B+ -**raiz**: Apuntador al nodo raíz. -**Arbol_B+()**. *Constructor*. Crea un árbol vacío de orden m.
2	**Operaciones:** público: Arbol_B+(Entero+: orden) ~Arbol_B+() insArbol_B+(Reg: r) eliArbol_B+(TipoClave: cl) conArbol_B+(TipoClave: cl): TipoResto rango(TipoClave: vi, vf): ListaEnl[Reg] vacíoArbol_B+(): Lógico numClaves(): Entero+ altura(): Entero+ orden(): Entero+ despliegue() **privado:** limpiarArbol_B+(ApuntadorA NodoB: pa) recorrer(ApuntadorA NodoB: pa, PilaEnl[ElePila]: p, TipoClave: cl): Lógico DividirNodo(ApuntadorA NodoB: pa, ph, TipoClave: clp): Lógico propagarFision(ApuntadorA NodoB: pa, ph, TipoClave: clp, PilaEnl[ElePila]: p) compactarNodo(ApuntadorA NodoB: pa, PilaEnl[ElePila]: p) Redistribucion(ApuntadorA NodoB: pa, pp, phi, phd, Entero+: p,c)	**-~Arbol_B+().** *Destructor.* Destruye el árbol . -**insArbol_B+()**. *Transformador*. Ingresa una nueva entrada en la posición que le corresponde según el orden ascendente de sus claves. -**eliArbol_B+()**. *Transformador*. Elimina una entrada de clave = cl, si el árbol no está vacío. -**conArbol_B+()**. *Observador*. Regresa el resto de la entrada si la clave existe. **-rango().** *Observador.* Regresa la lista de las entradas $\in$[vi, vf] -**vacíoArbol_B+()**. *Observador*. Regresa Verdadero si el árbol está vacío. -**numClaves()**. *Observador*. Regresa el número actual de claves. **-altura()**. *Observador*. Regresa la altura actual de la estructura. **-orden()**. *Observador*. Regresa el orden del árbol_B+. **-despliegue().** *Observador*. Despliega el contenido del árbol. **-limpiarArbol_B+().** *Transformador.* Devuelve todos los nodos. **-recorrer().** *Observador.* Desciende en el árbol hasta encontrar la hoja donde debe ser insertado r, regresando el camino de descenso en la pila y la dirección del nodo hoja en pa. **-dividirNodo().** *Observador.* Divide un nodo en dos nodos hermanos, devolviendo el nodo fisionado en pa, su hermano en ph y la clave promocionada en clp. **-propagarFision().** *Observador.* Inserta la clave promocionada en el nodo padre, si éste desborda llama a dividirNodo() hasta que no haya desborde, regresando el nodo actual en pa. **-compactarNodo().** *Observador.* Compacta un nodo semi-vacío. **-redistribucion().** *Observador.* Redistribuye las entradas luego de una eliminación.

UNIVERSIDAD DE LOS ANDES

Árbol_B+

- **Ventajas:**
  - Facilidad para la construcción y mantenimiento del árbol
  - Se mantiene un árbol siempre equilibrado
  - La altura del árbol es menor que en el B, ya que la información asociada a la clave R aparece solamente en el archivo de datos haciendo que las entradas del árbol sean más pequeñas y la ramificación sea mayor
  - Los costos de tratamiento siguen siendo logarítmicos
  - Permite el acceso secuencial ordenado por la clave al archivo y a su vez el acceso selectivo a un registro con clave dada
- **Desventaja:**
  - Es necesario llegar siempre hasta el último nivel (hojas) para determinar si una clave solicitada está o no en el índice y por ende en el archivo

Técnicas de compactación en árboles_B

- **Una de estas técnicas toma en cuenta los prefijos comunes que puedan existir entre las claves, ya que ellas normalmente se construyen con campos que identifican unívocamente los registros y que tienen un significado en la realidad (árbol_B con prefijos)**
  - Por ejemplo, si se indexa en archivo de publicaciones en una biblioteca, cada libro presente está identificado por una cota cuyos primeros caracteres identifican el área de la que trata el libro, así:

QA76.9D3M3 TK51055T35

QA76.73C25L5 TK51055P76

Prefijo: QA76.

Prefijo: TK51055

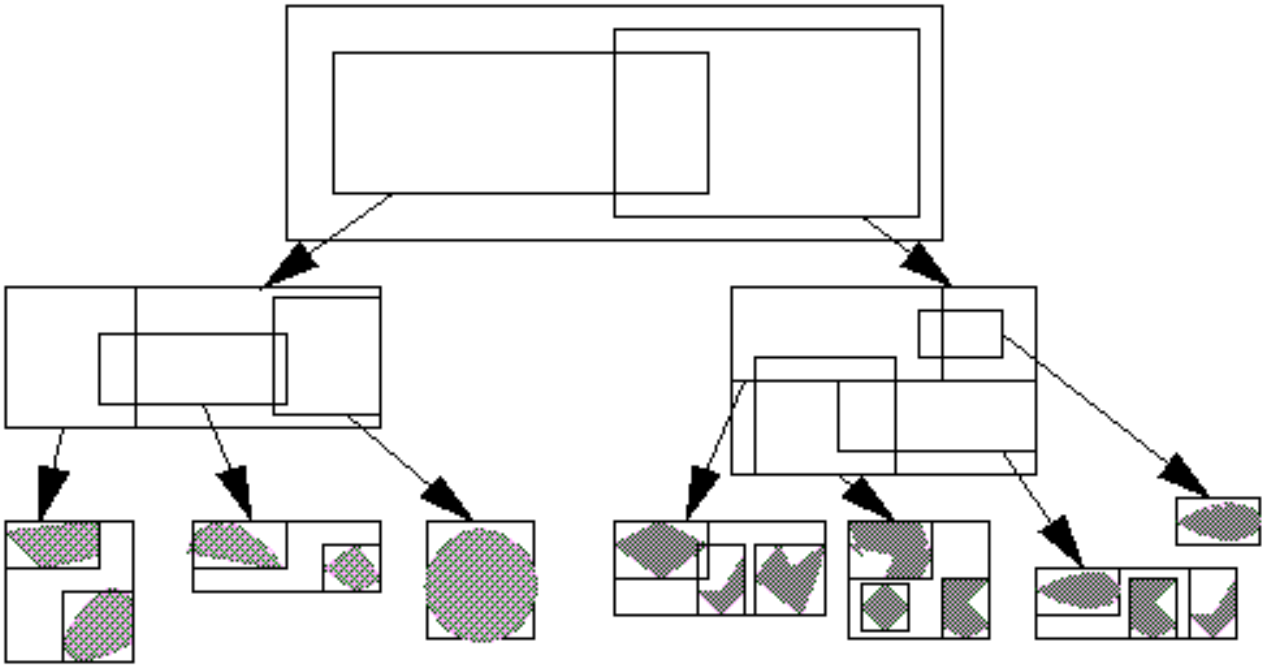
UNIVERSIDAD DE LOS ANDES

Árbol_R

- **Propiedades similares al árbol_B: dinámico, balanceado, cada entrada tiene un único identificador y crece solamente hacia arriba (Guttman, 1984)**
- **Índice para la recuperación de objetos según su localización o posición en el espacio.**
- **Cada nodo hoja contiene entradas de la forma (I,p)**
  - donde I es el mínimo rectángulo que cubre al objeto espacial indizado, I=(I1,...,Id) donde d es el número de dimensiones e Ij es un intervalo [a,b] que describe la extensión del objeto en la dimensión j
  - p es el apuntador al registro donde se encuentran los datos de ese objeto.
- **Los nodos índice tienen básicamente la misma forma pero el apuntador p contendrá la dirección de un nodo hijo e I será el mínimo rectángulo que cubre todos los rectángulos que están en el nodo hijo**

Árbol_R

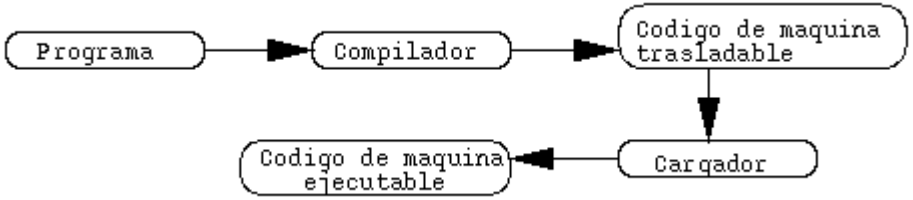
- **Operaciones:**
  - Consulta
  - Inserción-expansión
  - Eliminación-condensación



Objetivos de los SGA

1. **Posibilidad de automatizar la gestión de datos de una empresa**
2. **Utilización de los discos: permitir el manejo de la MS**
3. **Independencia entre los programas de aplicación y las memorias secundarias**
4. **Uso de lenguajes anfitriones**

lenguaje de programación que permite la inclusión en su código de las sentencias de manipulación de archivos y las de definición de los datos que ellos contienen



5. **Posibilidades de acceso secuencial y selectivo**
6. **Posibilidad de múltiples usuarios**

Objetivos del SGA

7. **Seguridad y protección de los archivos**

Debe proteger los archivos contra accesos malintencionados o no autorizados y garantizar la conservación de los archivos en caso de fallas de algún equipo o de algún programa

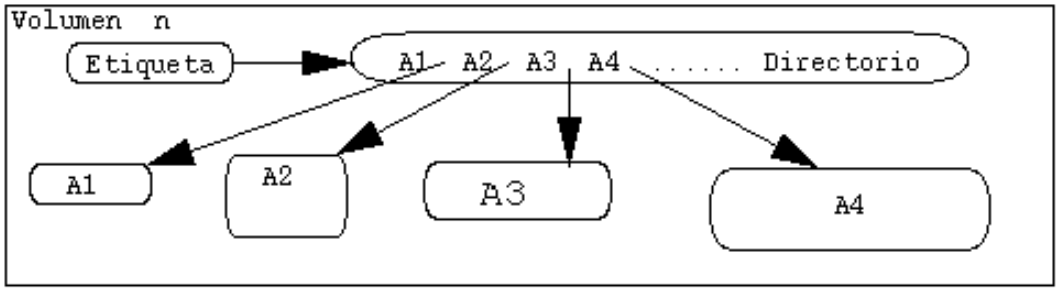
Técnicas de control de accesos: utilización de derechos de acceso y las claves de protección

Protección contra las fallas: Procesos de recuperación que dependen del tipo de falla

- Falla simple (pérdida de parte o todo el contenido de la memoria central) se utiliza la recuperación en caliente
- Falla catastrófica (se pierde parte o toda la memoria secundaria (falla de disco)) se usa la recuperación en frío, pues se deben usar los respaldos más recientes

Funciones del SGA

1. **Manipulación de archivos: Operaciones básicas**
2. **Direccionamiento relativo: útil en caso de copia del archivo y control del mismo**
3. **Localización de espacio en la memoria secundaria: archivos de tamaño fijo o variable, necesitan reservar zonas de MS continuas para el almacenamiento del mismo**
4. **Localización de los archivos en los volúmenes: identificación del volumen (nombre o número) que le sirve al operador al montarlo o desmontarlo en la unidad**



Funciones del SGA

5. **Control de los archivos: El núcleo del SGA incluye las funciones de:**
   - compartir archivos
   - resistencia a las fallas
   - seguridad y confiabilidad de los datos

Un método de acceso es selectivo si él **no** es secuencial

- **Organización y métodos de acceso selectivo**
  - Acceso selectivo a un registro implica contar con un identificador del registro o del conjunto de registros que se desean obtener luego de la búsqueda
  - Este identificador se llama comúnmente la **clave de búsqueda o de recuperación**

identificador asociado a cada registro físico que puede o no estar incluido en el registro lógico

UNIVERSIDAD DE LOS ANDES

Organización relativa

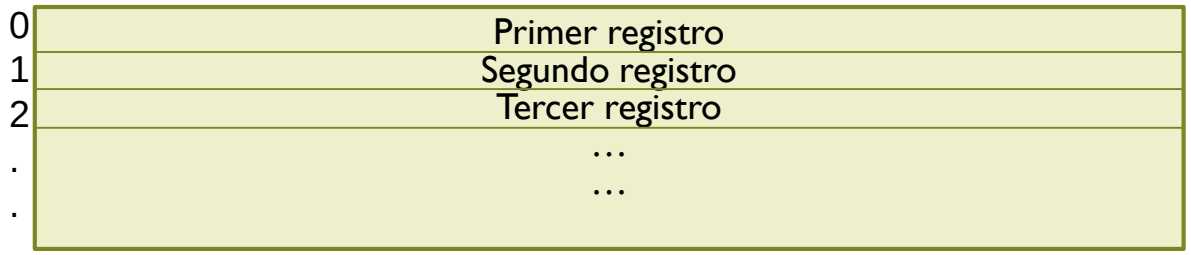
- **Clave: número entero que determina la dirección relativa del registro en el archivo**
- **Archivo directo o relativo: archivo de registros de longitud fija donde cada registro tiene asociado un número de orden en el archivo siendo éste su clave única y numérica**
  - Cálculo de la dirección de inicio del registro buscado i (DR_i)

$$DR_i = D + L \times i$$

  - donde **D** es la dirección de inicio del archivo, **L** es la longitud de los registros e **i** es la clave
  - Si la longitud de los registros es variable se selecciona la longitud máxima, es decir **L** = Máx(L_o, L_1, ... , L_n) siendo n el número de registros en el archivo menos 1
- **Organización válida para archivos pequeños cuyos registros sean de tamaño mas o menos fijo y colocados continuamente, sin huecos intermedios**

Organización relativa

- **Ventajas:**
  - Son muy simples de usar y programar
  - Se necesita un **único acceso** para lectura o para escritura
- **Desventajas:**
  - Los registros deben ser de longitud fija
  - Acceso y tratamiento debe hacerse por la clave o en forma secuencial
  - Si se hacen eliminaciones se tendrán huecos intermedios
    - Se marcan los registros eliminados o
    - Se compacta el archivo eliminando la brecha



UNIVERSIDAD DE LOS ANDES

Organización aleatoria

- **Clave: campo del registro con el que se calcula la dirección relativa del paquete donde se colocará el registro**
- **Archivo aleatorio: los registros son "empotrados" en P paquetes de tamaño fijo l, cuya dirección se calcula por medio de una función (*hash*) aplicada a la clave**
  - generalmente se tiene un acceso a disco para la L/E de un paquete
  - dirección relativa DR_i del paquete i donde se encuentra el registro de clave buscada se calcula con la misma fórmula del anterior

$$DR_i = D + LP \times i$$

  - donde, **D** es la dirección de inicio del archivo, LP el tamaño de los paquetes e **i** es el número del paquete
  - dentro del paquete los registros se colocan en orden de llegada

UNIVERSIDAD DE LOS ANDES

Organización aleatoria

- **Funciones de transformación clave-dirección o *hash*:**
  - **Método del centro de los cuadrados, análisis de dígitos, plegado, desplazamiento, división (el más popular), división polinómica, método de Lin, etc**
- **archivo con huecos intermedios, dada la distribución de los valores de la clave**
- **porcentaje de ocupación del 80% al 90% es aceptable**
- **Inserción en paquete lleno produce un desbordamiento o colisión, que se resuelve con los métodos de manejo de colisiones**
  - El más simple de todos es prohibir la inserción en paquete lleno
  - Direccionamiento abierto: búsqueda secuencial de un espacio vacío
  - Encadenamiento: formando una lista enlazada de paquetes de desborde
  - Aplicar una segunda función de empotramiento

UNIVERSIDAD DE LOS ANDES

Organización aleatoria

- **Ventajas:**
  - Se adapta bien a cualquier tipo de clave y es simple
  - Da un buen rendimiento si no hay desborde
  - Lectura de un paquete en un acceso
  - Escritura en dos accesos, si no hay desborde
- **Desventajas:**
  - Se degrada con los desbordes
  - La tasa de ocupación es menor que uno (TC < 1)
  - El tamaño del archivo debe fijarse a priori si no se usa la solución virtual o dinámica

UNIVERSIDAD DE LOS ANDES

Organización indizada

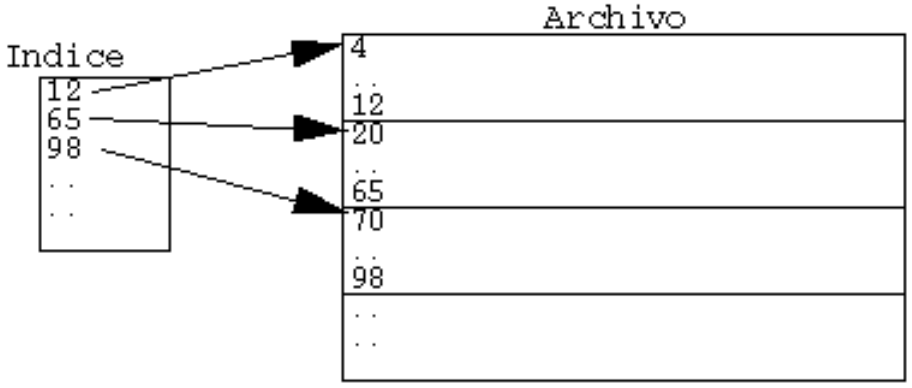
- **Clave: campo(s) del registro con el que se calcula su dirección relativa**
- **Archivo asociado con un índice de acceso**
- **Índice: estructura que permite asociar a una clave (concatenada o no) la dirección relativa de ese registro dentro del archivo**
- **Acceso a un registro en el archivo:**
  - Se accede al índice
  - Se busca dentro del índice la dirección relativa del registro que contiene la clave
  - Se convierte la dirección relativa a absoluta
  - Se accede al registro del archivo
  - Se transfiere el registro al usuario que lo solicitó

Organización indizada

- **Densidad de un índice: den = NC / NR**

  donde NC: número de claves en el índice y NR: número de registros en el archivo
- **Índice denso: den = 1**
- **Índice no denso: den < 1**
- **El archivo se divide en bloques de tamaño fijo y el índice se conforma con las tuplas de la forma:**

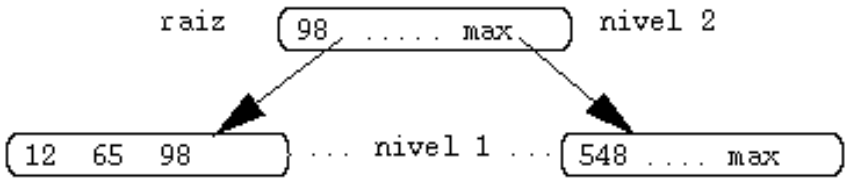
  < clave mayor del bloque , dirección relativa del bloque >



Organización indizada

- **Índice jerárquico: índice con n niveles en forma de árbol**
- **IS3: tipo indizado del sistema 3**
- **ISAM: secuencial indizado**
- **VSAM: secuencial indizado regular IBM**

		Archivo ordenado	Archivo desordenado
Índice denso	Ordenado	posible	IS3
	Desordenado	Imposible	posible
Índice no denso	Ordenado	ISAM, VSAM	Imposible
	desordenado	Imposible	Imposible



ISAM *(Indexed Sequential Access Method)*

- **Primer método de acceso secuencial indizado que aparece en las referencias bibliográficas**
- **Primera implantación: IBM en sus sistemas 360/370 bajo el sistema operativo DOS y OS/VS**
- **Ventajas:**
  - archivo siempre ordenado
  - tiempo de acceso bueno si no hay desbordes (3 E/S para leer un registro)
- **Desventajas:**
  - gestión de los desbordes compleja y degrada el sistema
  - necesita reorganizaciones periódicas
  - dependiente de las características de los discos

UNIVERSIDAD DE LOS ANDES

ISAM

- **Características:**
  - Apegado a las características de la MS (discos)
  - Usa tres zonas lógicas:
    - **primaria**, donde van los registros
      - comprendida por n pistas de los cilindros sucesivos asignados al archivo
      - registros almacenados por orden creciente de sus claves
    - **desborde**, donde van los registros que no caben en la primaria
      - Compuesta por algunas pistas de los cilindros sucesivos asignados al archivo y por un cilindro independiente denominado cilindro de desborde donde se colocan los que no caben en la zona de desborde de los cilindros
      - registros enlazados (lista) siguiendo el orden creciente de sus claves
    - **índice**, donde van las entradas del índice
      - compuesta por las primeras pistas de cada cilindro donde se encuentran los últimos niveles del índice jerárquico denominado índice de pistas, un índice de cilindros y si es necesario un índice maestro

ISAM

- el índice de pistas y el de cilindros son obligatorios y el maestro es opcional
- El índice de pistas está presente en cada cilindro, bien sea en la primera pista o en la pista de la mitad del cilindro, dependiendo de las velocidades de acceso a dichas pistas.

- **Índice de pistas se compone de:**
  - un campo que contiene el apuntador al último registro en la zona de desborde, denominado control de desborde (CD)
  - las entradas del índice compuestas por
    - la clave mayor en la pista (**CMP**)
    - el apuntador a esa pista
    - la clave mayor de la pista en la zona de desborde (**CMD**)
    - el apuntador al inicio de la lista enlazada de registros en la zona de desborde

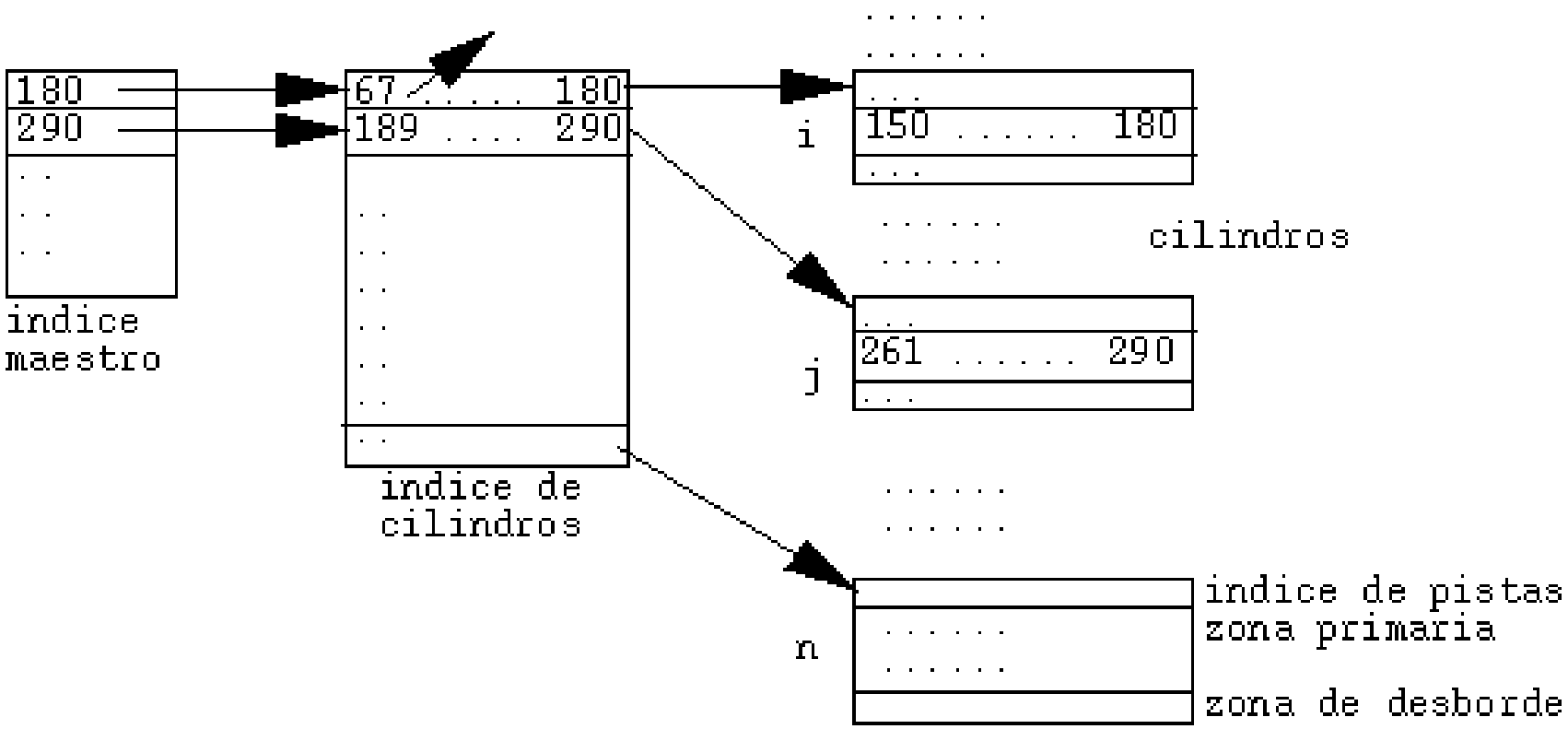
CD | CMP1 | A1 | CMD1 | AL1 | ... | CMPn | An | CMDn | ALn

UNIVERSIDAD DE LOS ANDES

ISAM

- **El índice de cilindros es uno solo por archivo y está almacenado en una zona particular del volumen**
- **Sus entradas están compuestas por:**
  - la clave mayor en el cilindro i (CMCi) y
  - el apuntador al índice de pistas en ese cilindro
- **El índice maestro se usa única y exclusivamente cuando el índice de cilindros es muy grande, es decir desborda la zona donde está almacenado**
- **Sus entradas se componen de:**
  - la clave mayor del paquete j en el índice de cilindros (CMPj) y
  - el apuntador a dicho paquete

ISAM

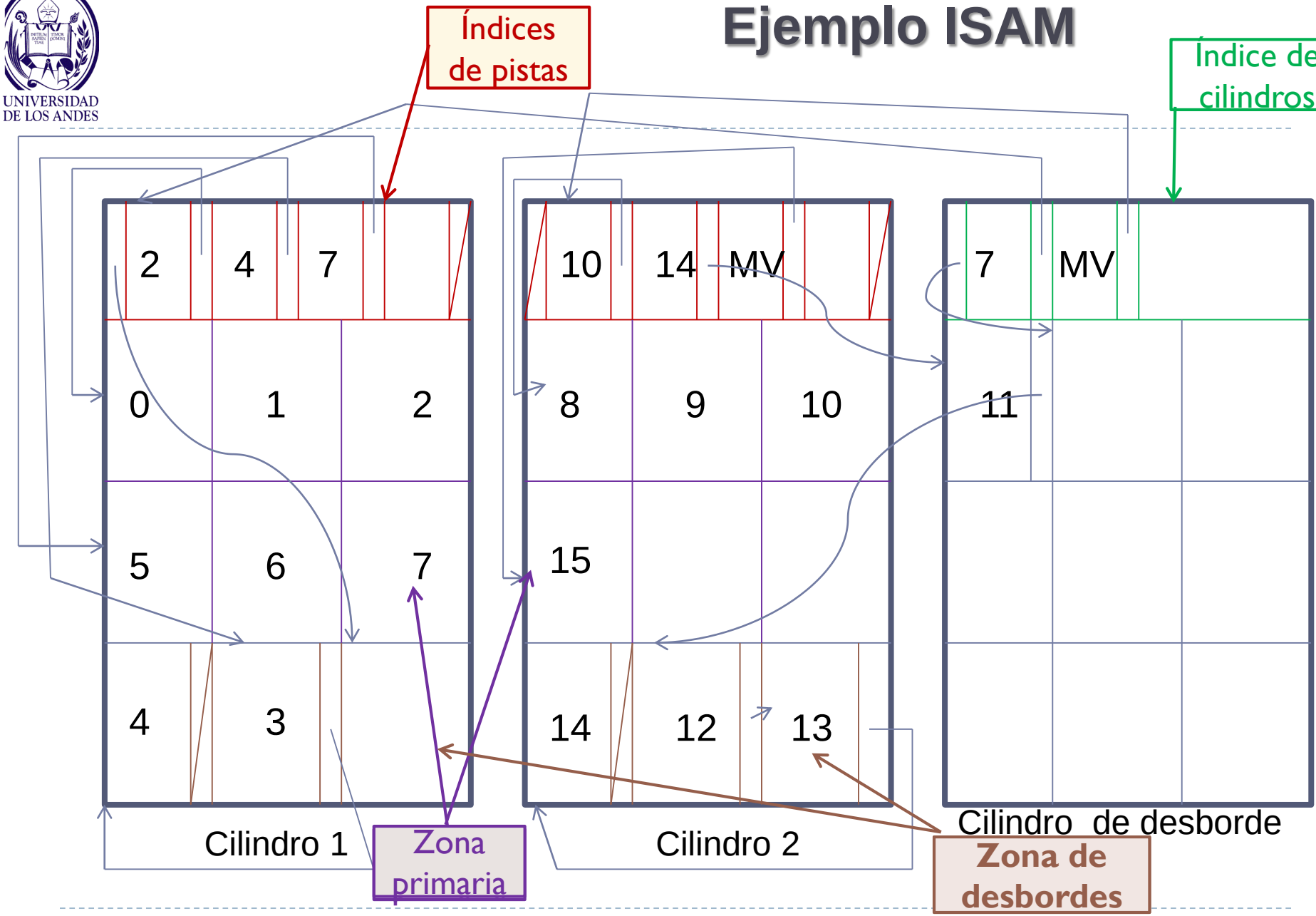

180
290
indice
maestro
67 180
189 290
indice de
cilindros
i
150 180
cilindros
j
261 290
n
indice de pistas
zona primaria
zona de desborde

UNIVERSIDAD DE LOS ANDES

Ejemplo ISAM

- **La secuencia inicial de carga está dada por los registros ordenados por su clave numérica: 0, 1, 4, 5, 6, 7, 10, 12, 14 y 15**
- **2 pistas en la zona primaria y 3 registros por pista**
- **Carga inicial:**
  - se hace en las zonas primarias de los cilindros asignados, sin colocar ningún registro en la zona de desborde, por lo cual los campos CMD en cada índice de pista contendrán el valor nulo
  - Se utiliza un valor de clave MV seleccionado para que no exista ninguna clave de registro mayor que MV
- **Luego de unas nuevas inserciones correspondientes a los registros con claves: 2, 3, 11, 8, 13 y 9**
- **La estructura obtenida se muestra en la próxima diapositiva**

Ejemplo ISAM



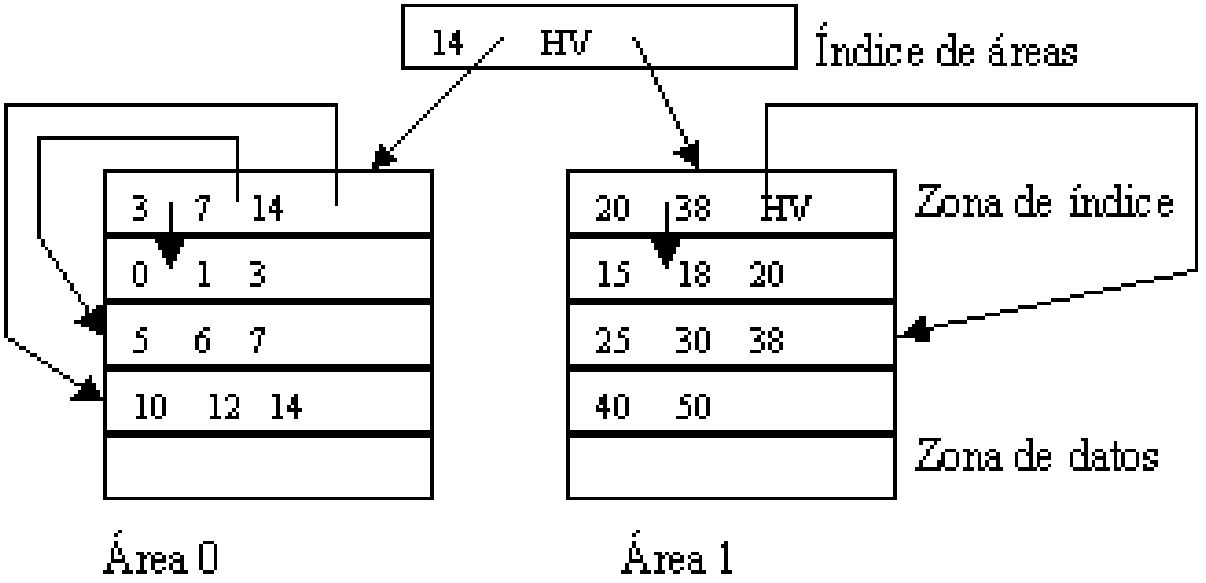
VSAM (*Virtual Sequential Access Method*)

- **Archivo secuencial indizado regular: archivo ordenado por la clave con un índice no denso ordenado, donde el archivo y el índice están organizados bajo la forma de un árbol_B+**
- **Surge como una mejora de IBM al anterior ISAM**
- **Hace que el almacenamiento de los archivos sea más independiente de la MS**
- **Características:**
  - Archivo dividido en áreas (conjunto de pistas de un cilindro o de cilindros contiguo)
  - Cada área se divide en intervalos, los cuales se componen de una parte de una pista o de varias pistas consecutivas accedidas en una E/S
  - Hay división de intervalos y áreas para producir el crecimiento del archivo, por lo cual no hay áreas de desborde pues las políticas de fisión las absorben

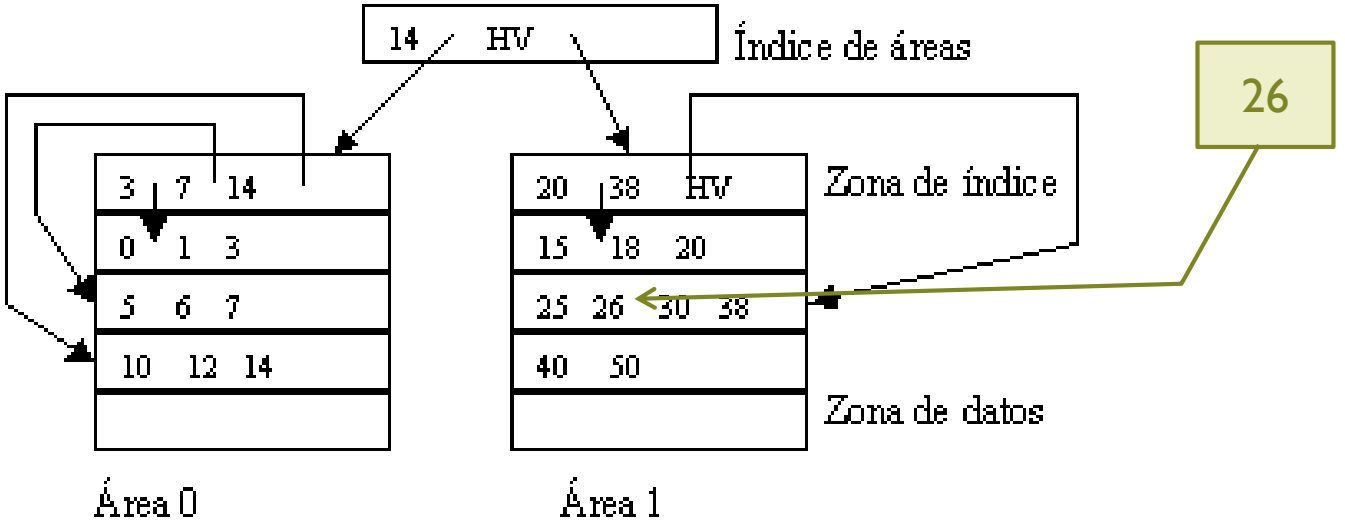
VSAM

- **Zona de datos: contiene los registros**
  - Está dividida en intervalos y áreas
  - Registros almacenados por orden ascendente de la clave
  - Las actualizaciones son previstas dejando intervalos vacíos por áreas y registros vacíos por intervalos
- **Ejemplo:**

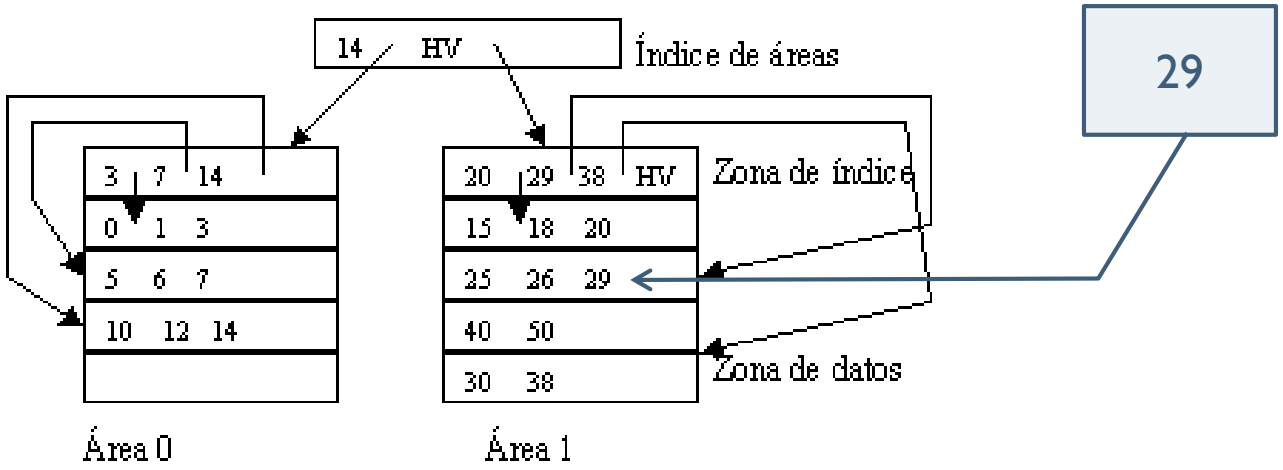
- Carga inicial: 0, 1, 3, 5, 6, 7, 10, 12, 14, 15, 18, 20, 25, 30, 38, 40 y 50
- áreas con 4 intervalos
- 4 registros por intervalo



- **Casos de inserción:**
  - Primero: inserción sobre un intervalo no lleno
    - Como el registro de clave 26 va en el segundo intervalo, este último es leído y el registro es insertado dentro de él siguiendo el orden creciente de sus claves

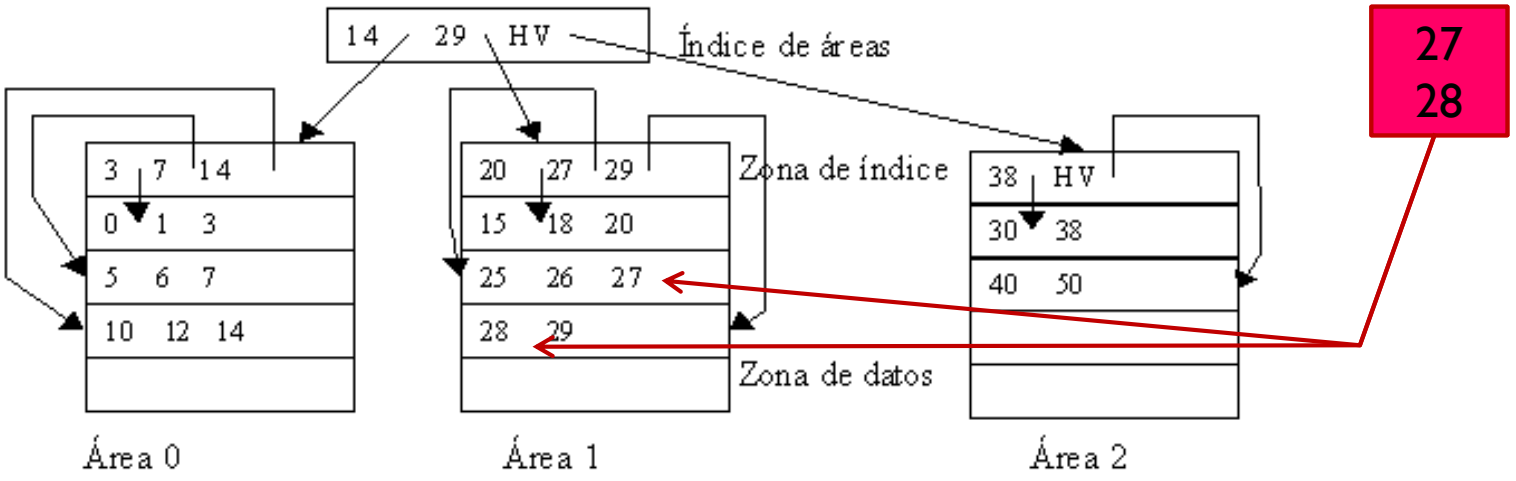


- **Casos de inserción:**
  - Segundo: inserción sobre un intervalo lleno
    - Como el intervalo está lleno y la inserción debe hacerse allí, se dice que dicho intervalo desbordó por lo cual debe ser dividido en dos intervalos semi-llenos
    - Esta fisión puede presentar dos casos suplementarios, ellos son:
      - Existe un intervalo vacío en el área hacia el cual se dirigirán los registros en desborde: inserción del registro con clave 29



Casos de inserción:

- No existe un intervalo vacío en el área:
  - Al no haber un intervalo libre en el área, el intervalo que desborde tendrá que ser dividido y dicha fisión hará que el área desborde a su vez, teniéndose que fisionar el área también. La fisión de un área se realiza de igual manera que la de los intervalos. La inserción de los registros de claves 27 y 28 consecutivamente hacen que el área 1 desborde, creándose una nueva área la número 2



UNIVERSIDAD DE LOS ANDES

VSAM

- **Zona de índice: pueden existir 2 ó más niveles de índice**
  - Primer nivel: índice de intervalos y hay uno por cada área
    - Cada entrada contiene la clave mayor del intervalo y su dirección
  - Segundo nivel: índice de áreas y hay uno por archivo
    - Cada entrada de éste contiene la clave mayor del área y su dirección
  - En caso que el índice de área sea muy grande y desborde se tiene un tercer nivel denominado índice maestro
- **Ventajas:**
  - El archivo está siempre ordenado y agrupado según la clave
  - El tiempo de acceso en lectura es siempre tres E/S
- **Desventajas:**
  - El tiempo de acceso en escritura puede ser grande si hay fisión de intervalo y de área

UNIVERSIDAD DE LOS ANDES

Autoevaluación

1. ¿Qué es un árbol_B+ y cuáles son sus diferencias con el árbol_B?
2. ¿Cuál es el formato de un nodo del árbol_B+?
3. ¿Qué es un árbol_R y cuáles son sus diferencias con el índice malla y sus similitudes con el árbol_B?
4. ¿Cuáles son los objetivos y las funciones del SGA?
5. ¿Cuáles son las diferencias entre las organizaciones de archivos directa, aleatoria e indizada?
6. ¿Cuáles son las características de los métodos de acceso indizados ISAM y de VSAM?
7. ¿Cuál es la relación entre el árbol_B+ y VSAM?

UNIVERSIDAD DE LOS ANDES

Ejercicios

Dibuje lo solicitado a continuación para las claves: AB, 2, 34, 56, 1, 45, 44, 89, 78, 64, 77, 92, 43, 55, 12, 9, 8, 10, 32, 16, 4 y 18

1. Un árbol_B+ con k = 3
2. La carga inicial de ISAM con: 3 cilindros (incluido el de desborde), 3 pistas por zona primaria en cada cilindro y 3 registros por pista
3. Luego de la carga inicial, inserte las claves: 48, 0 y 118
4. La estructura VSAM con: 3 intervalos por área y 3 registros por intervalo
5. Elimine los registros de clave: 78, 56 y 77 de cada una de las estructuras (árbol_B+, ISAM y VSAM)