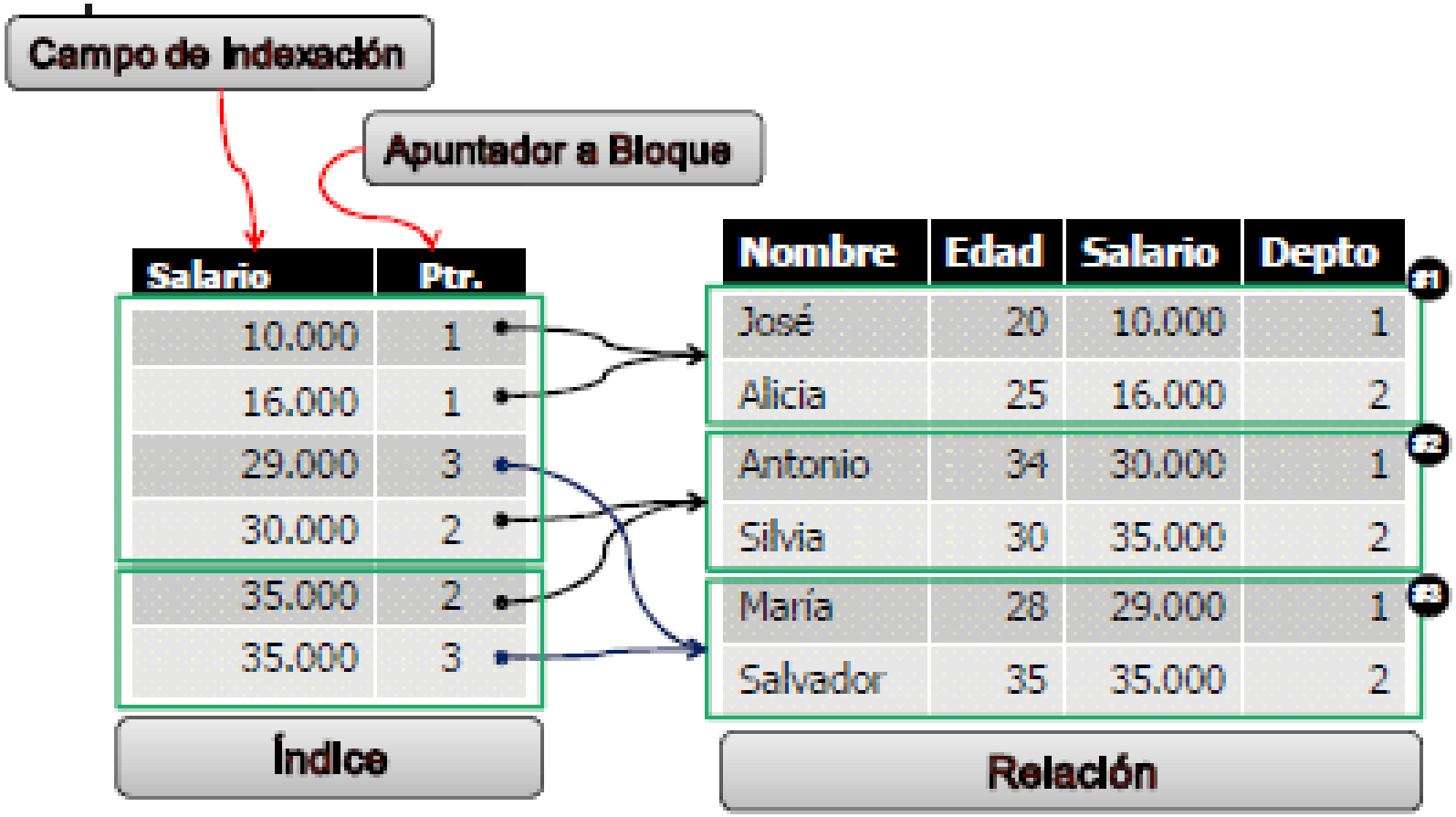
ÍNDICES

PROCESAMIENTO Y OPTIMIZACIÓN DE CONSULTAS

ÍNDICES

- Archivo ordenado que contiene una copia de uno o más atributos de una tabla.
- Para cada valor del atributo se tiene un puntero a donde se almacena el registro correspondiente.
- Objetivo: acelerar la lectura de datos de tablas
- **Clave**: la clave del incide se forma por uno o mas columnas de una tabla.
- **Contenido**: Valores de la clave de indexación; junto con los punteros a todos los bloques que contienen registros con esos valores.

ÍNDICES: EJEMPLO

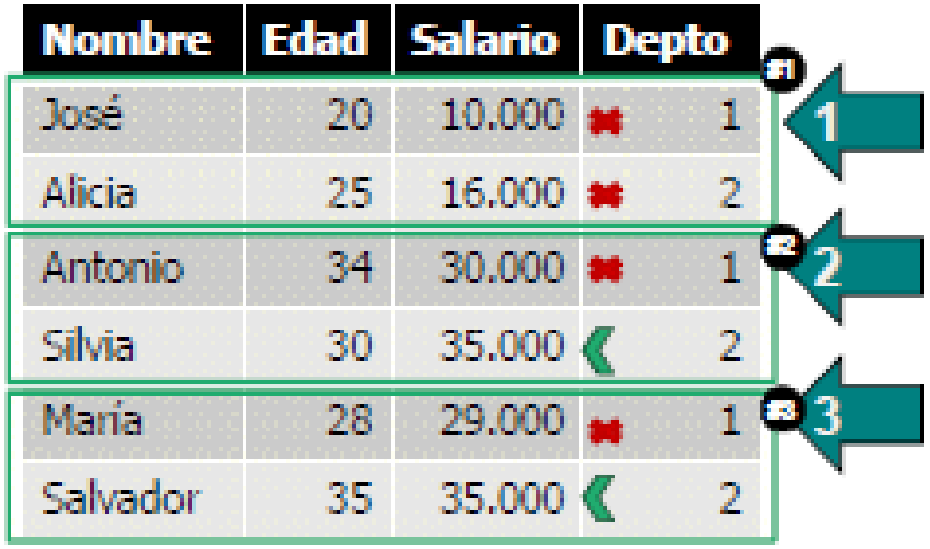


PROCESAMIENTO Y OPTIMIZACIÓN DE CONSULTAS PROCESAMIENTO SIN ÍNDICES

Archivo Secuencial

- Se lee uno a uno los registros de Empleados
- Ello implica leer todos los bloques de disco
- Para cada registro se evalúa la condición de búsqueda
- Tengo que leer el 100% de los registros para obtener el resultado

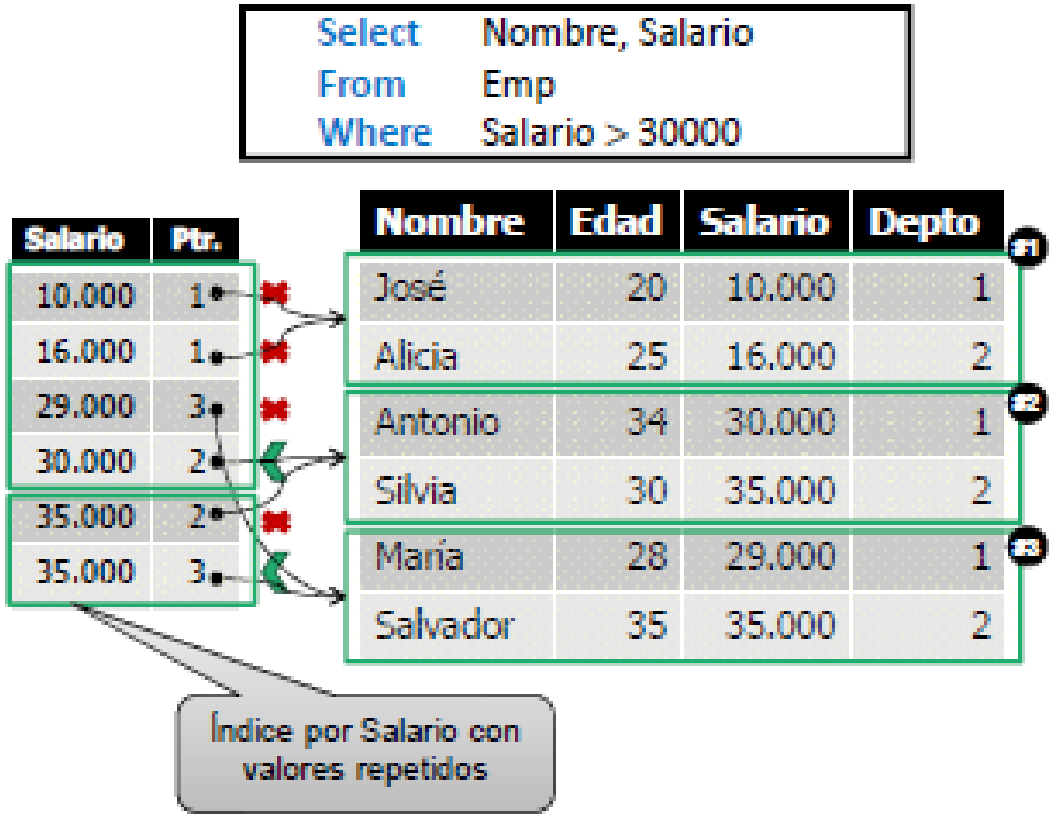
```
Select   Nombre, Salario
From     Emp
Where    Salario > 30000
```



PROCESAMIENTO Y OPTIMIZACIÓN DE CONSULTAS PROCESAMIENTO CON ÍNDICES

Índice

- Se realiza la búsqueda en el índice
- Ello implica leer todos los bloques del índice del disco
- Para cada clave del índice se evalúa la condición de búsqueda
- Luego se leen los datos de la tabla desde el disco



TIPOS DE ÍNDICES

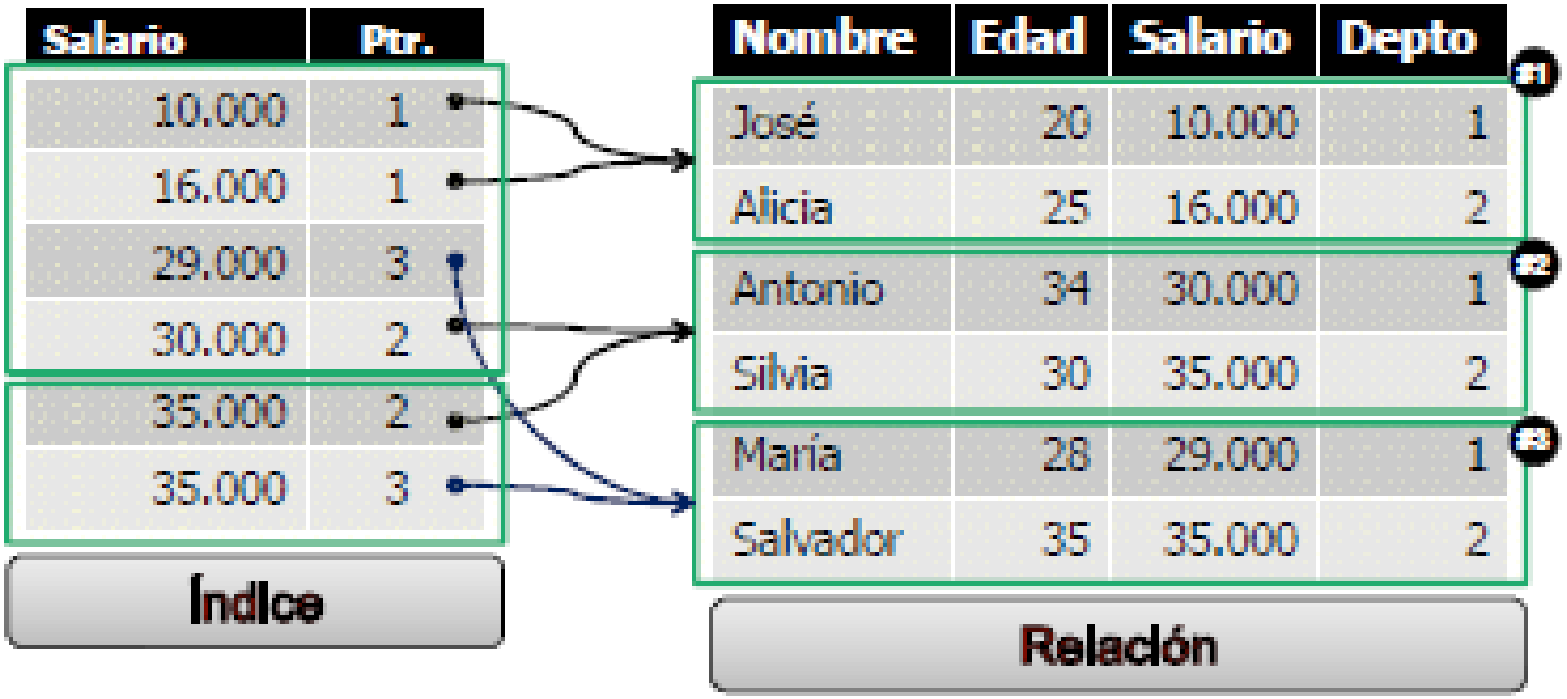
- Índice secundario
- Índice de agrupamiento
- Índice primario

ÍNDICE SECUNDARIO

ÍNDICE SECUNDARIO

- Archivo ordenado que se construye sobre **uno o más atributos de una relación.**
- Puede haber varios índices secundarios sobre un mismo archivo físico de datos.
- Nunca puede haber más de un índice sobre los mismos atributos de indexación
- Los índices secundarios pueden construirse tanto sobre atributos **claves o sobre** atributos **no clave**

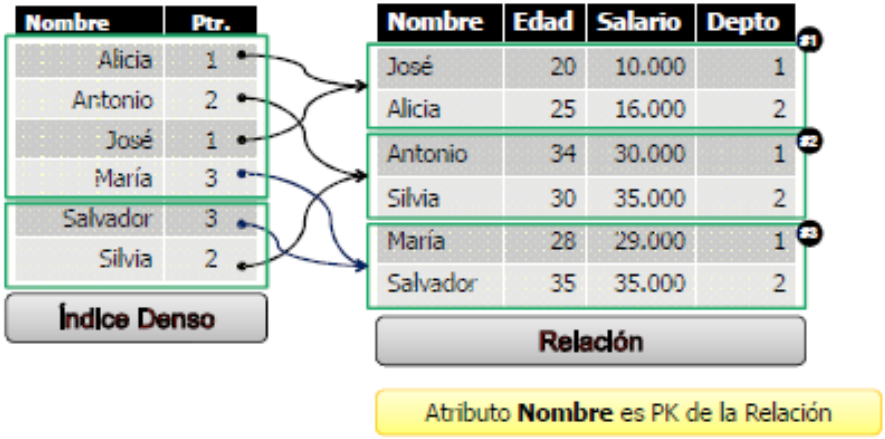
EJEMPLO - Índice Secundario



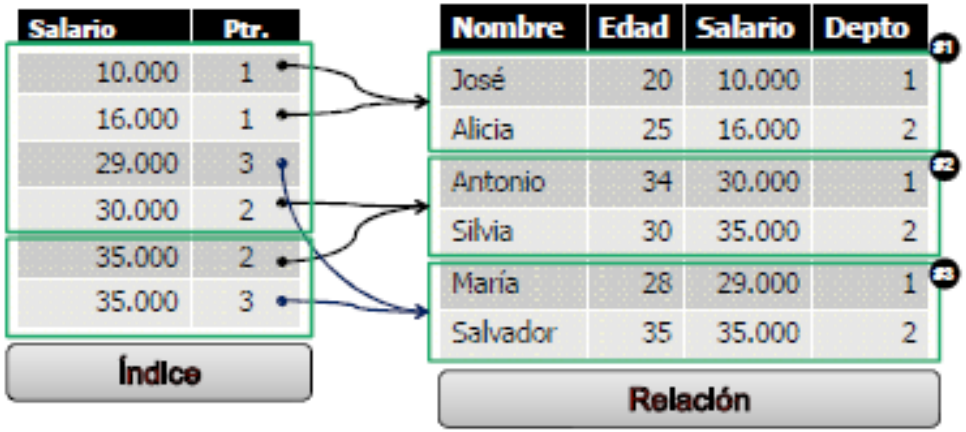
CLASIFICACIÓN - Índice Secundario

- Los índices secundarios sobre **atributos claves son índices densos**
- Los índices secundarios sobre **atributos no claves pueden ser densos o no densos**

EJEMPLO - Índice Secundario :
ÍNDICE **DENSO SOBRE ATRIBUTO CLAVE**



EJEMPLO - Índice Secundario :
ÍNDICE **DENSO SOBRE ATRIBUTO NO CLAVE**

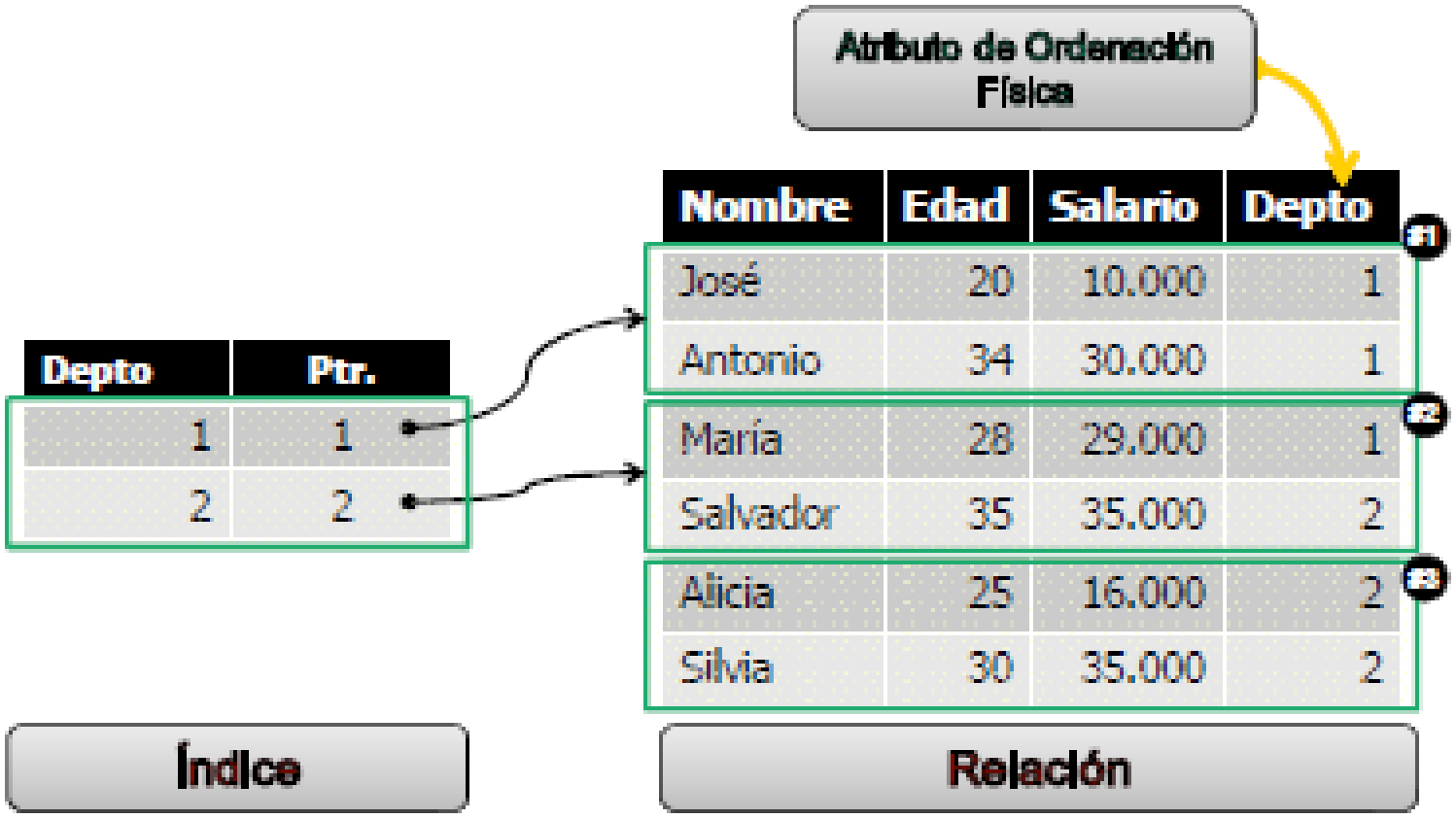


ÍNDICE DE AGRUPAMIENTO

ÍNDICE DE AGRUPAMIENTO

- Archivo ordenado que se construye **según el atributo de ordenamiento físico del** archivo de datos
- El atributo de ordenación del archivo puede tener valores repetidos

EJEMPLO - Índice de Agrupamiento



Índice de Agrupamiento

Son índices no densos

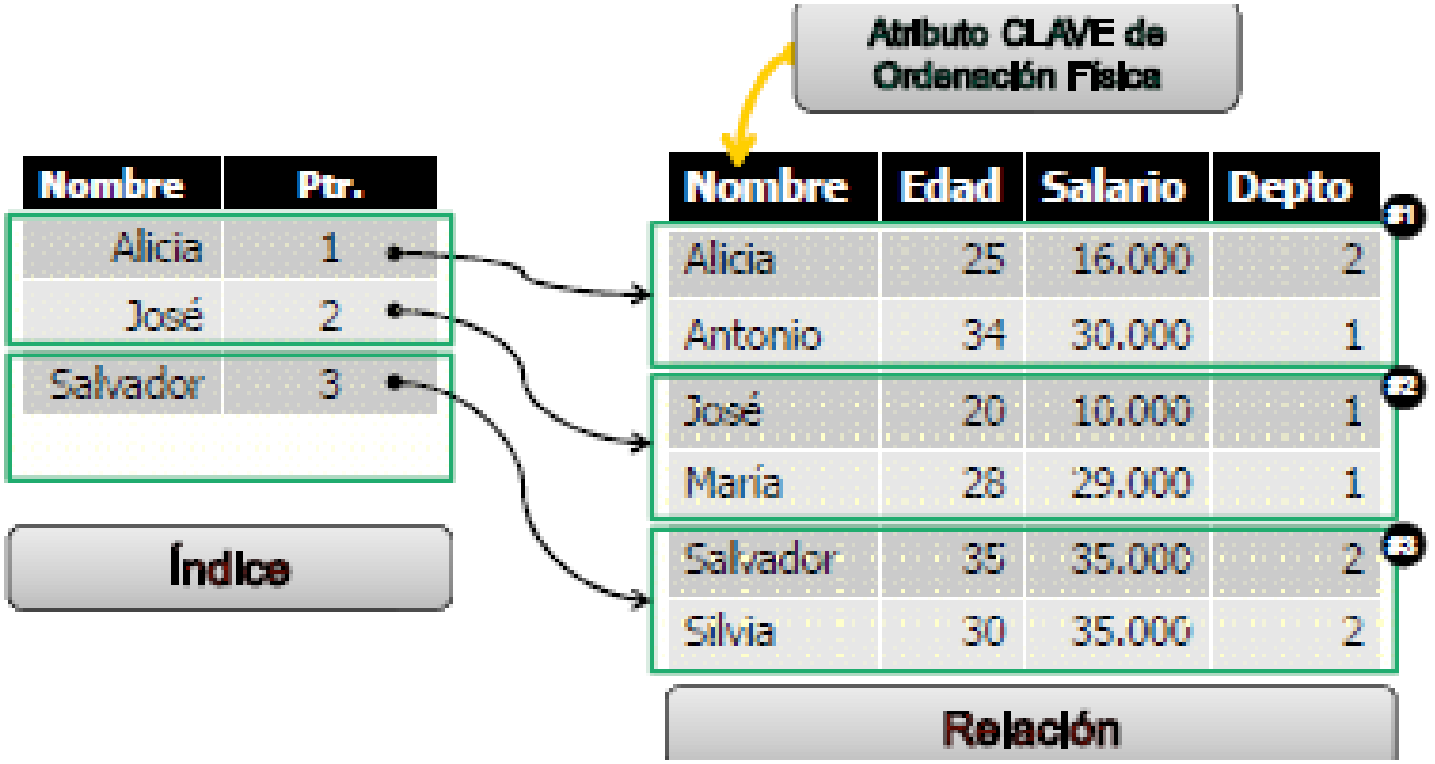
- Sólo se crea **un registro en el índice por cada valor distinto del atributo de agrupamiento**
- Se **referencia el primer bloque del archivo de datos que** contiene **la primer tupla con ese valor**
- Para encontrar el resto de las tuplas se lee secuencialmente el archivo hasta que se encuentre un registro que no cumpla la condición
- n índice = V (Atributo Indexación, Relación)

ÍNDICE PRIMARIO

ÍNDICE PRIMARIO

- Archivo ordenado que se construye **según el atributo de ordenamiento físico del** archivo de datos
- El atributo de ordenación del archivo **debe ser clave**

EJEMPLO - Índice Primario



Índice Primario

- Son índices no densos
- Sólo se crea **un registro en el índice por cada bloque de disco del archivo de datos**
- El número total de entradas del índice será igual al número de bloques de disco del archivo de datos
- n índice = b relación

TIPOS DE ÍNDICE - RESUMEN

Dadas las condiciones de los atributos de indexación, los índices más eficientes según el caso son:

Clave de Indexación	De Ordenación Física	NO Ordenación Física
Clave	Primario	Secundario
NO clave	Agrupamiento	Secundario

RESUMEN DE PROPIEDADES

TIPOS DE ÍNDICE

Tipo	Cantidad de entradas del índice	Denso o No
Primario	Cantidad de bloques del archivo de datos $n_{índice} = b_{relación}$	No denso
Agrupamiento	Cantidad de valores distintos del atributo de agrupamiento $n_{índice} = V(A, R)$	No denso
Secundario (por clave)	Cantidad de registros del archivo de datos $n_{índice} = n_{relación}$	Denso
Secundario (por No clave)	Cantidad de registros o cantidad de valores distintos del atributo de indexación $n_{índice} = n_{relación}$ **ó** $n_{índice} = V(A, R)$	Denso o No denso

ESTRUCTURAS DE ÍNDICES

ÍNDICES ESTRUCTURAS TÍPICAS

Índice Multi-nivel

- Mejora el rendimiento de los índices de un solo nivel particionando las búsquedas al generar “índices” sobre los índices en cada nivel.

Arboles B y B+

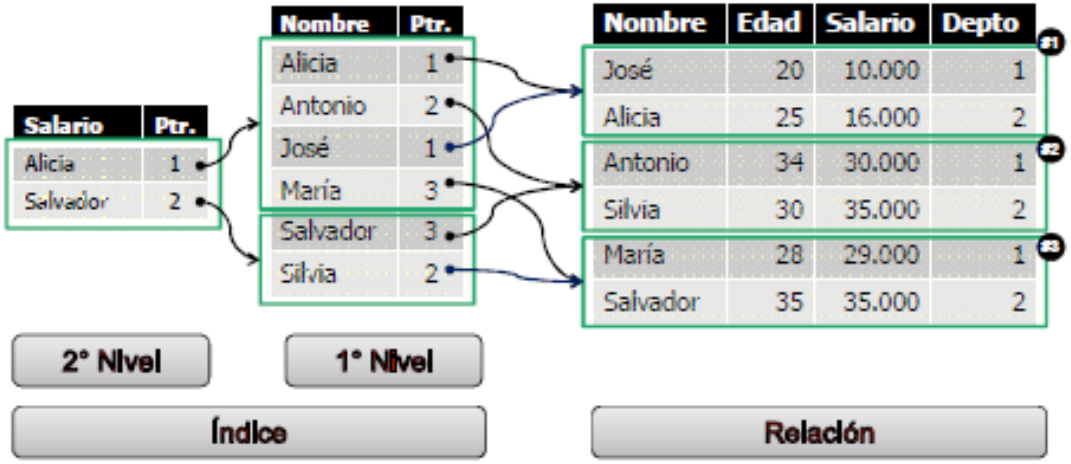
- Buen comportamiento en recuperación tanto por condiciones de igualdad como de orden. Buen comportamiento en la inserción
- Ocupa más disco.

Hash

- Muy buen comportamiento en la inserción y en la recuperación por condiciones de igualdad.
- No funciona bien para condiciones con relaciones de orden

ÍNDICES DE MÚLTIPLES NIVELES

EJEMPLO: 2 NIVELES



ÍNDICES DE MÚLTIPLES NIVELES

EJEMPLO: 3 NIVELES

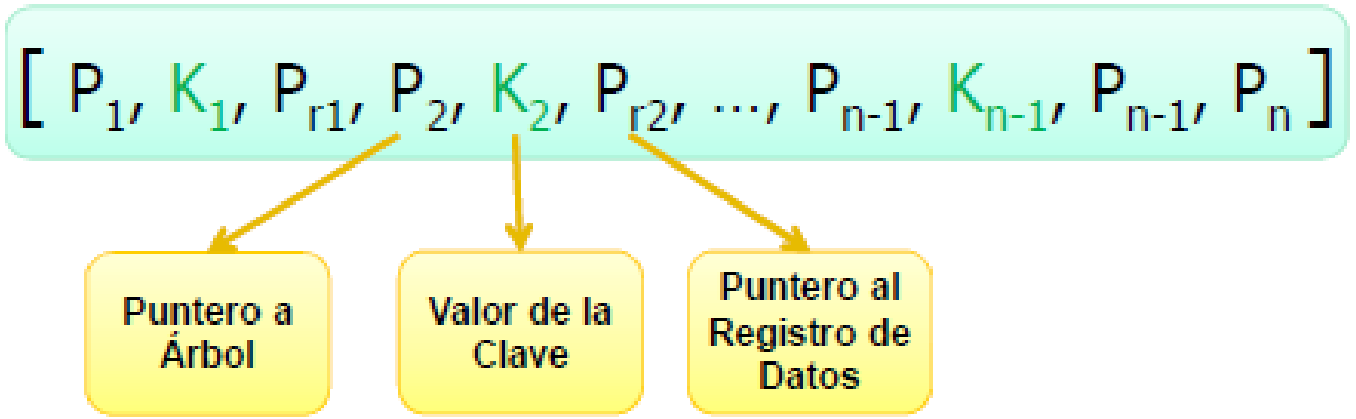
ÁRBOLES B

Un árbol B es un árbol de búsqueda con restricciones que garantizan que:

- El árbol este equilibrado
- El espacio desperdiciado por la eliminación no sea excesivo.

ESTRUCTURA

- Cada nodo interno del árbol B tiene la forma



ÁRBOLES B+

- Un árbol B+ es una variación de la estructura de datos del árbol B, en donde los punteros de datos sólo se almacenan en los nodos hojas.

Esto permite lograr:

- Índices con menos niveles
- Índices con mayor capacidad

NODO INTERNO ESTRUCTURA

El nodo interno de un árbol B^+ de **orden p** se tiene la siguiente forma:

$$[\ P_1, K_1, P_2, K_2, \ldots P_{n-1}, K_{n-1}, P_n\ ]$$

- $P_1, P_2, \ldots P_{n-1}, P_n$: Puntero a Árbol
- $K_1, K_2, \ldots K_{n-1}$: Valor de la Clave

Con $n \leq p$

NODO HOJA ESTRUCTURA

Los nodos hojas de un árbol B^+ de **orden p** tiene la siguiente forma:

$$[K_1, Pr_1, K_2, Pr_2, \ldots K_{n-1}, Pr_{n-1}, P_{sig\_hoja}]$$

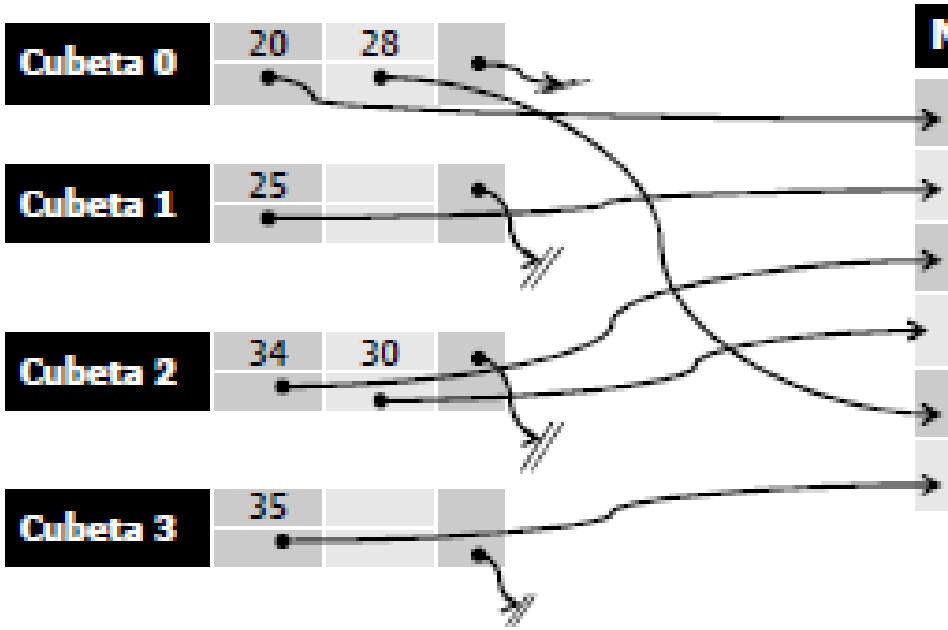
- $K_1, K_2, \ldots K_{n-1}$: Valor de la Clave
- $Pr_1, Pr_2, \ldots Pr_{n-1}$: Puntero al Registro de Datos
- P_{sig_hoja}: Puntero al siguiente Nodo Hoja

Con $n \leq p$

HASH

- Dividen el índice en cubetas según la clave de indexación
- Se usa una f(x) para determinar en qué cubeta se coloca una clave
- Las cubetas crecen mediante desbordamiento de bloques.

EJEMPLO – HASH

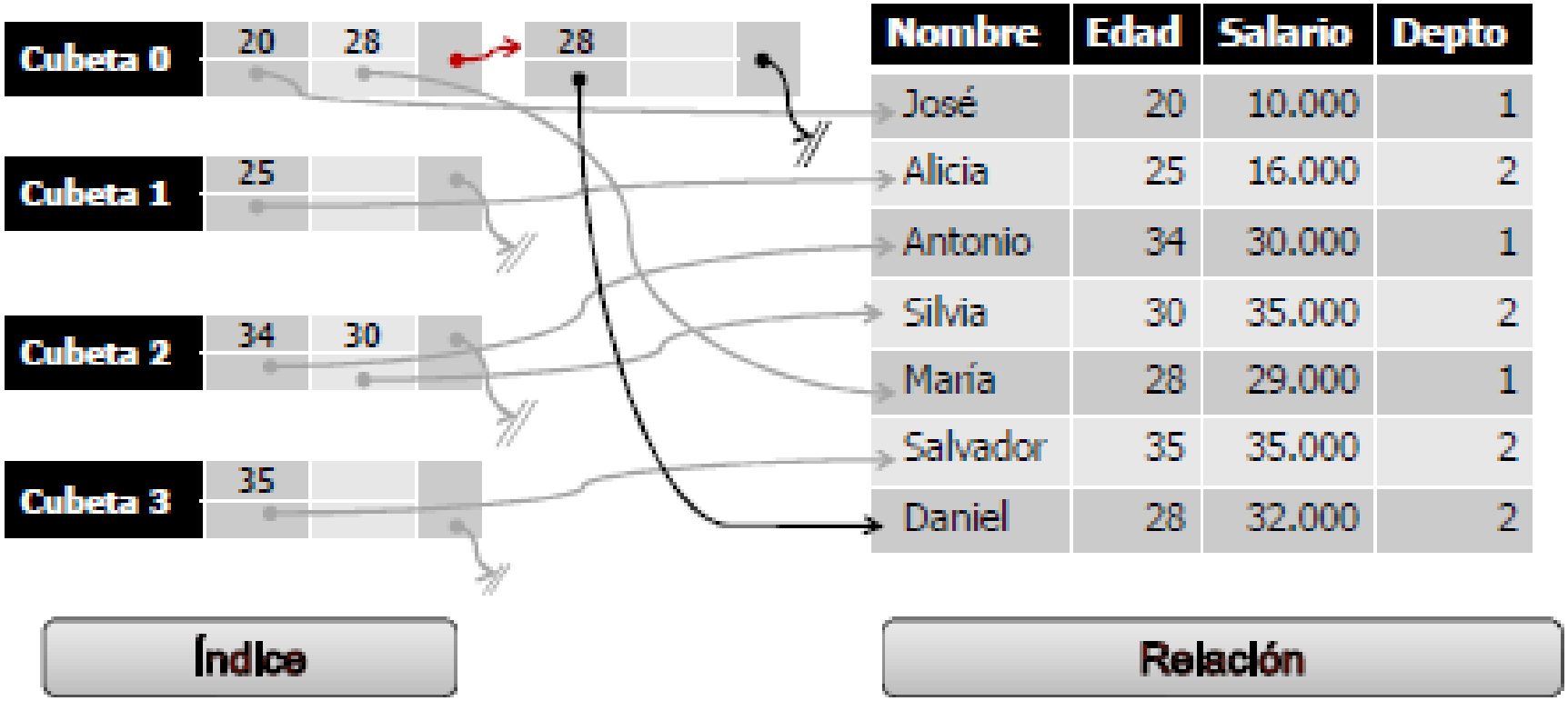


Nombre	Edad	Salario	Depto
José	20	10.000	1
Alicia	25	16.000	2
Antonio	34	30.000	1
Silvia	30	35.000	2
María	28	29.000	1
Salvador	35	35.000	2

Índice

Relación

EJEMPLO – HASH (NUEVA TUPLA)



Nombre	Edad	Salario	Depto
José	20	10.000	1
Alicia	25	16.000	2
Antonio	34	30.000	1
Silvia	30	35.000	2
María	28	29.000	1
Salvador	35	35.000	2
Daniel	28	32.000	2

RESUMEN

- Los DMBS's implementan diferentes estrategias para organizar los registros de una tabla:
  - Registros desordenados con acceso secuencial.
  - Registros ordenados con acceso secuencial.
  - Registros Indexados por la PK con hash.
  - Registros Indexados por la PK con árbol B+.
  - Registros Indexados por otro atributo con índices.
  - Índices con Duplicados o con Valores Únicos