Costos

Selección y Join

Selección en un bloque

σ Nombre LIKE ‘D%’ OR fecha_ingreso <= 2010-12-31

Selección en un bloque

σ Nombre LIKE ‘D%’ OR fecha_ingreso <= 2010-12-31

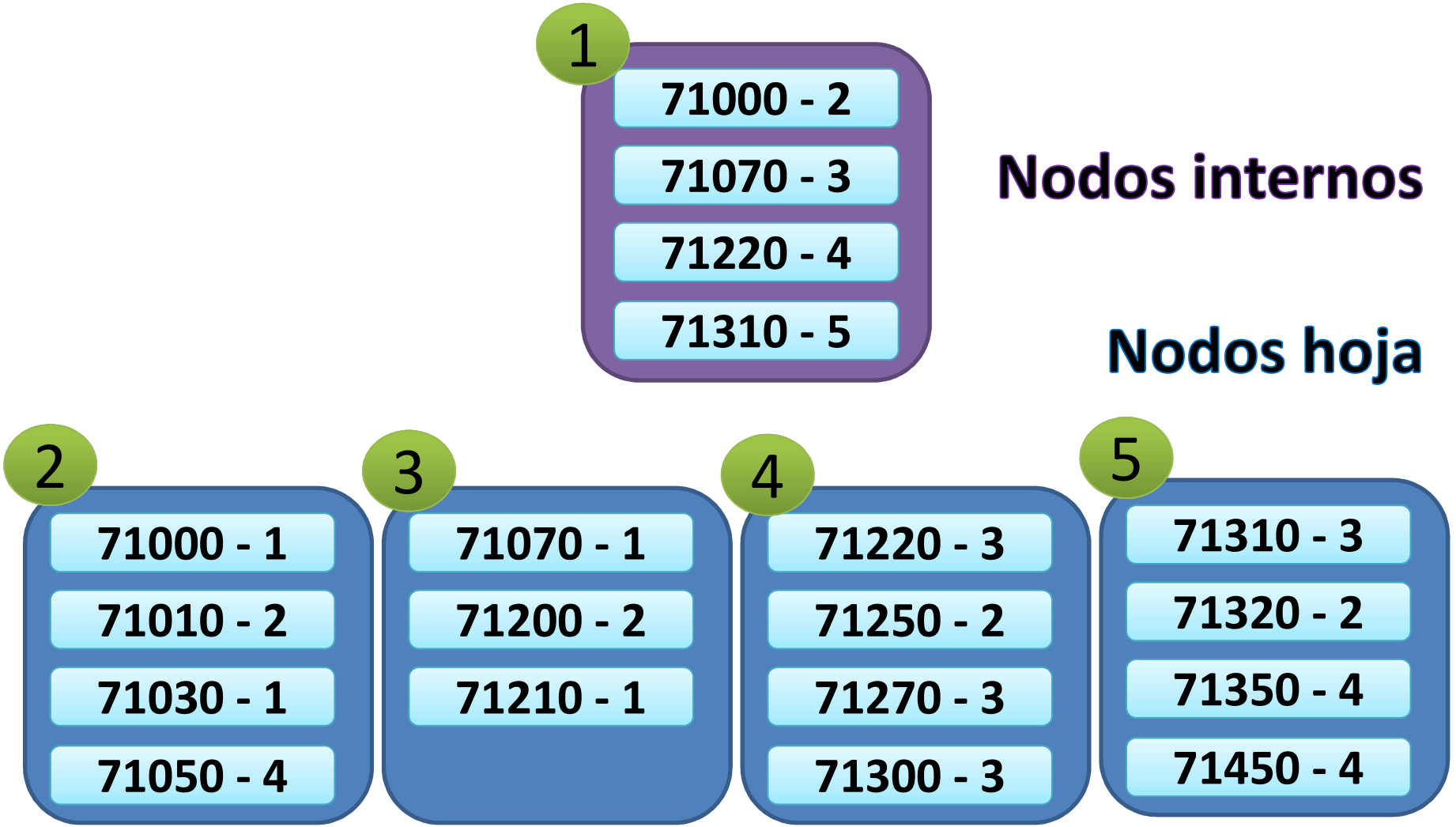
Selección en un bloque

σ Nombre LIKE ‘D%’ OR fecha_ingreso <= 2010-12-31

Costo file scan

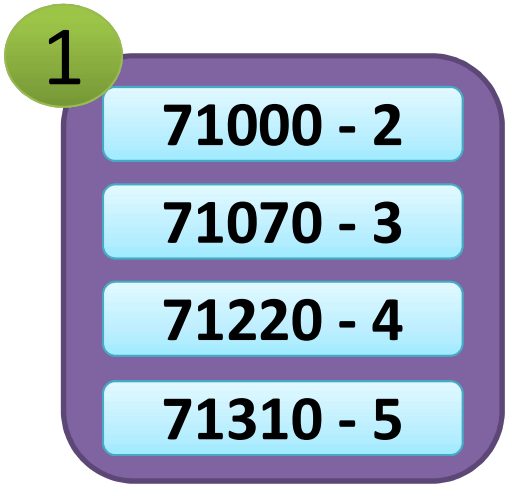
- B(R)

Índices Árbol B+



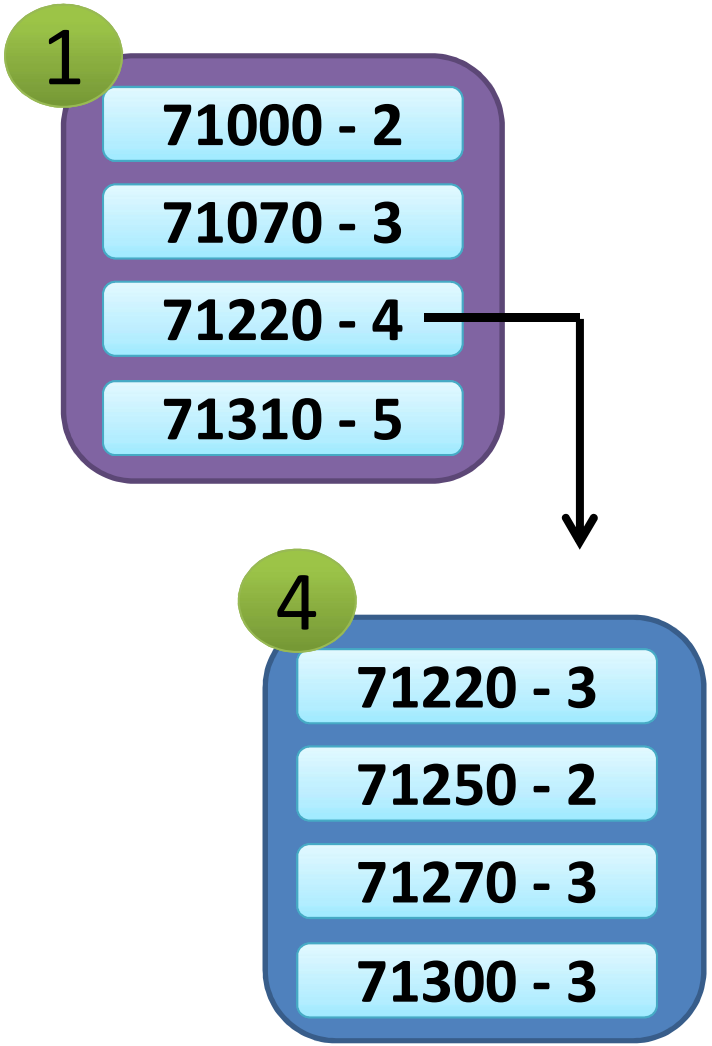
Árbol B+ búsqueda por igualdad

$\sigma_{padron = 71270}$



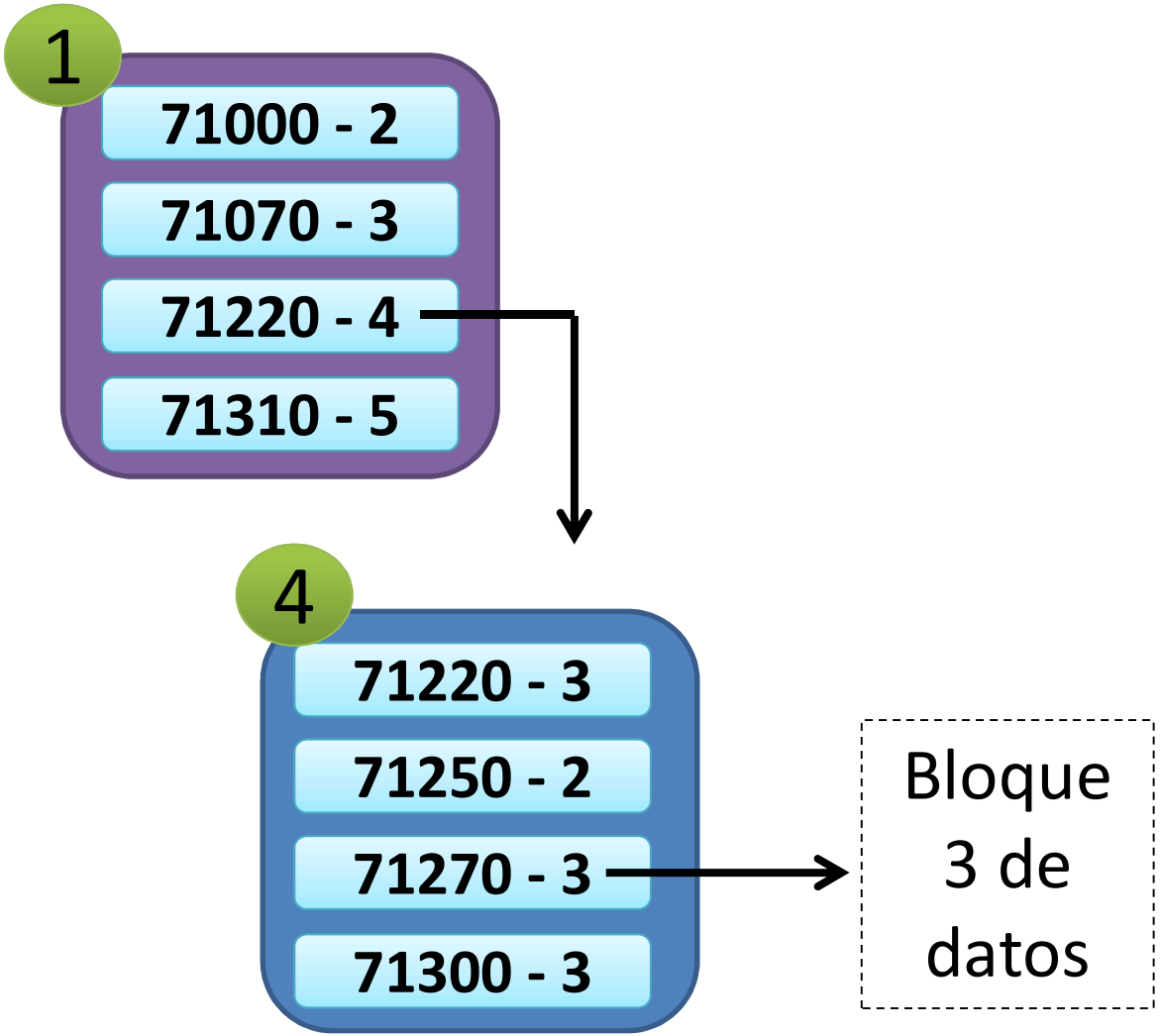
Árbol B+ búsqueda por igualdad

$\sigma_{padron = 71270}$



Árbol B+ búsqueda por igualdad

$\sigma_{padron = 71270}$



Costo de búsqueda por igualdad

- Altura de índice
- Acceso a los datos de la/s fila/s devueltas
  - Estimación de cantidad de filas
    - n(R) / V(Ai,R)
  - Si el archivo está ordenado por la clave de búsqueda, es menos de 1 acceso por fila

Costo index scan

- Índice primaro:

  Height(Idx) + 1

- Índice secundario:

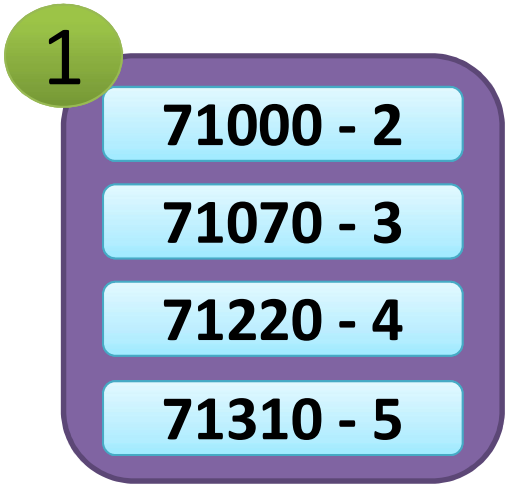
  Height(Idx) + ⌈ n(R) / V(Ai,R) ⌉

- Índice de clustering:

  Height(Idx) + ⌈ B(R) / V(Ai,R) ⌉

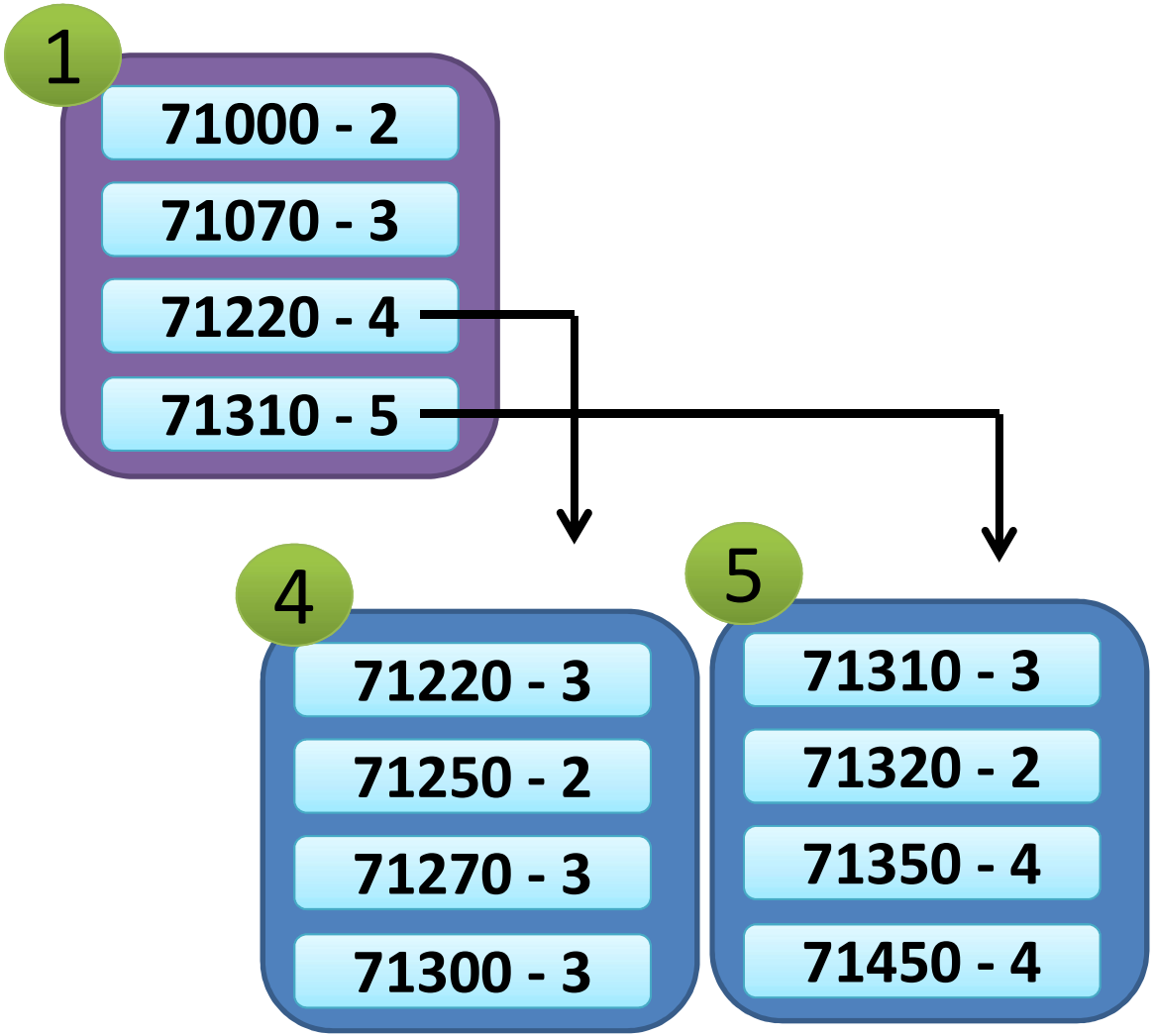
Árbol B+ búsqueda por rango

$\sigma_{\text{padron} >= 71270 \wedge \text{padron} <= 71320}$

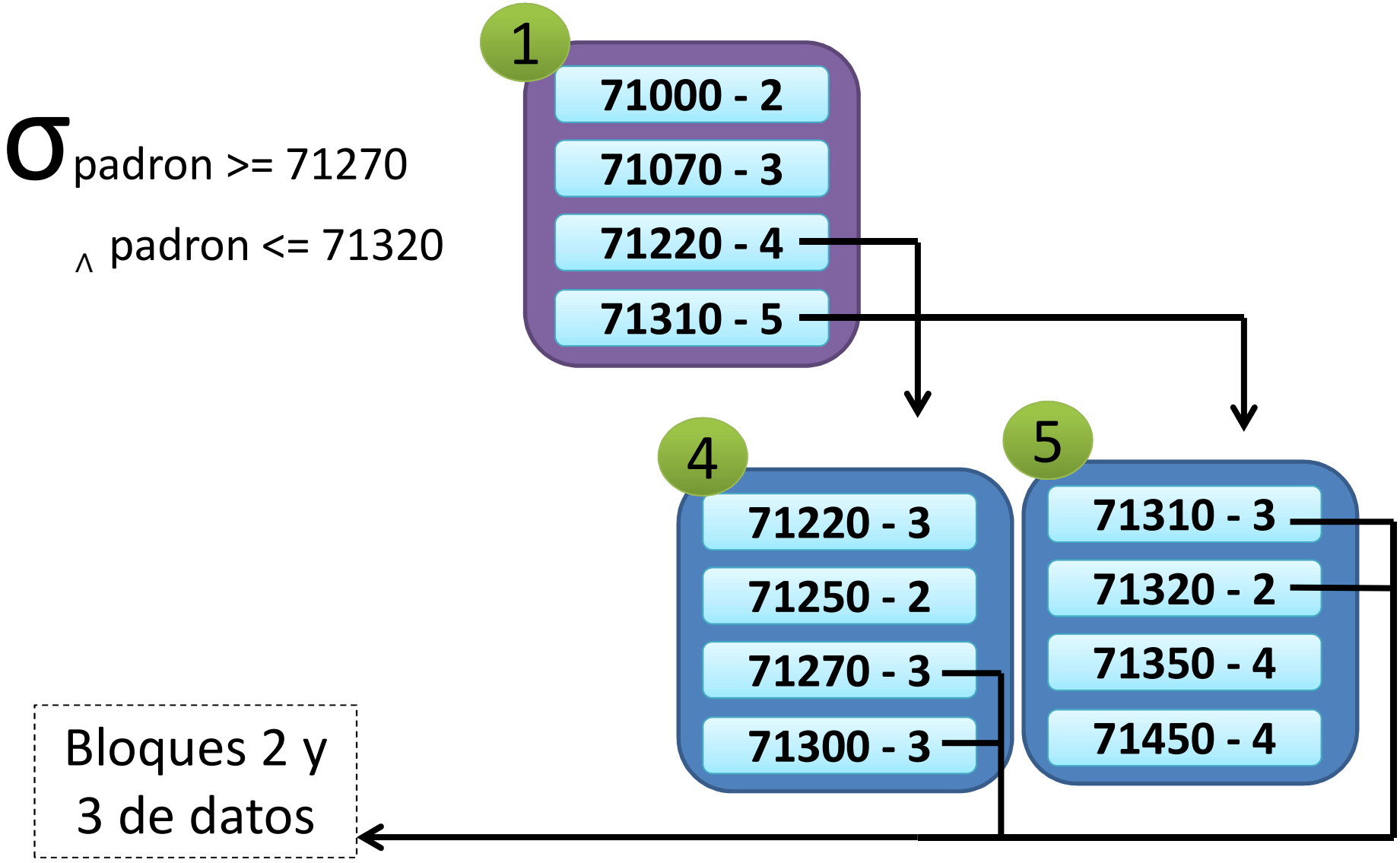


Árbol B+ búsqueda por rango

$\sigma_{\text{padron} >= 71270 \wedge \text{padron} <= 71320}$

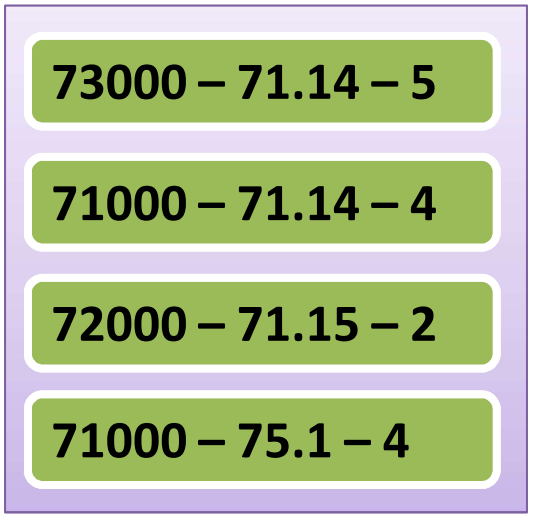


Árbol B+ búsqueda por rango



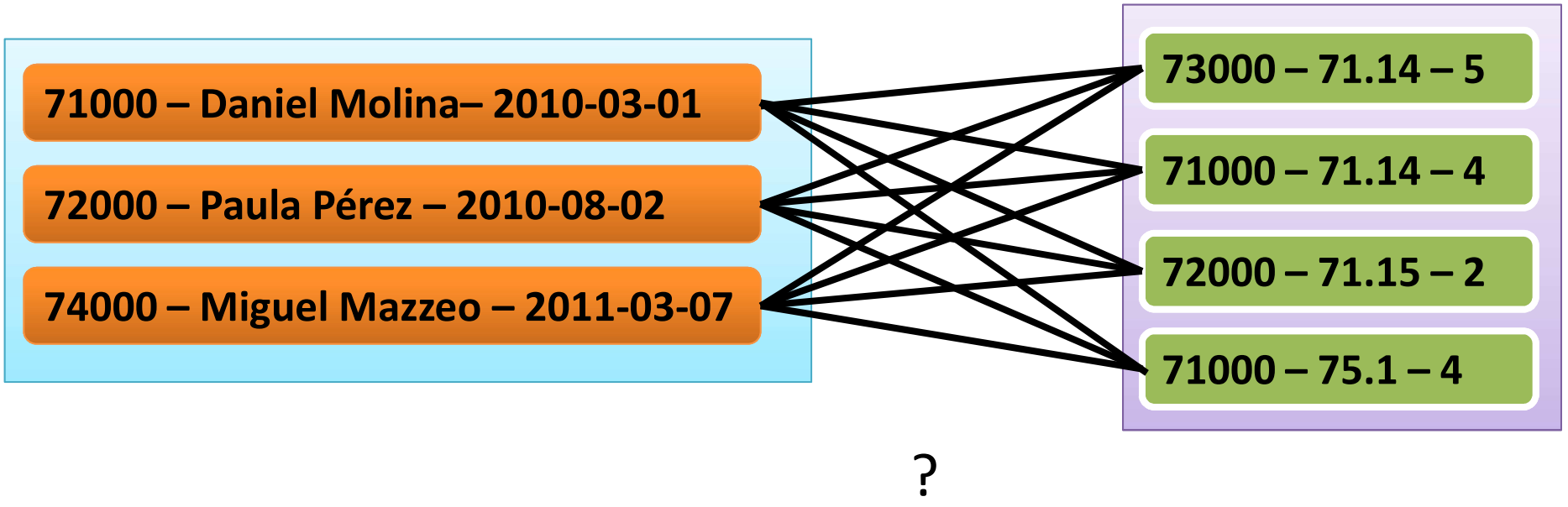
Join con bloques en memoria

$\bowtie$ alumnos.padron = notas.padron



Join con bloques en memoria

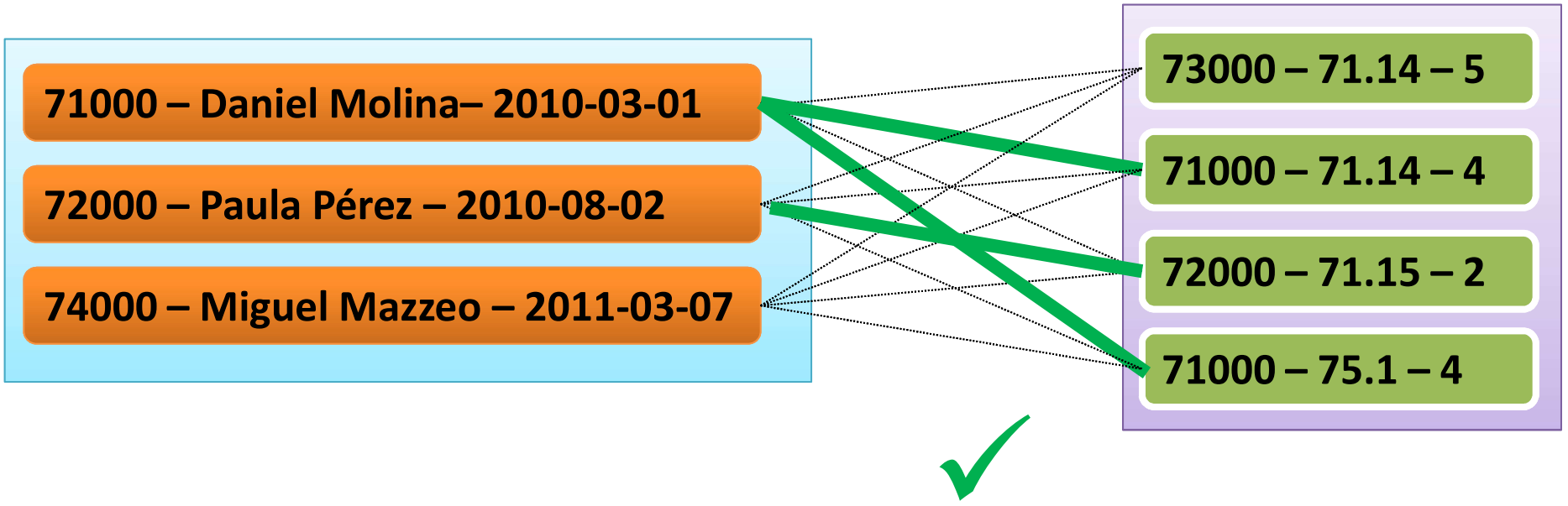
$\bowtie$ alumnos.padron = notas.padron



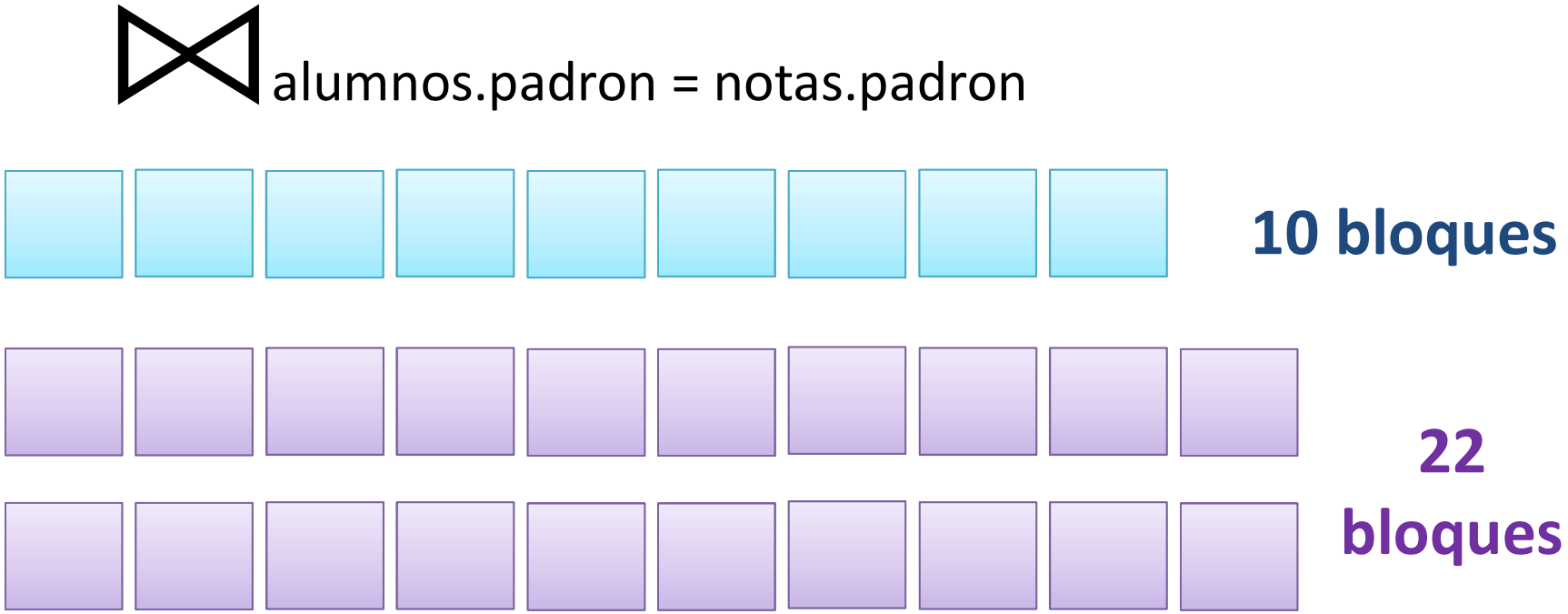
?

Join con bloques en memoria

$\bowtie$ alumnos.padron = notas.padron

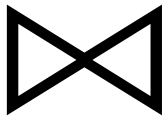


Join de varios bloques



- Con memoria suficiente
- Con 2 bloques para lectura y 1 para resultados
- Generalizar a M bloques (1 para resultados)

Join con Hash Grace


alumnos.padron = notas.padron

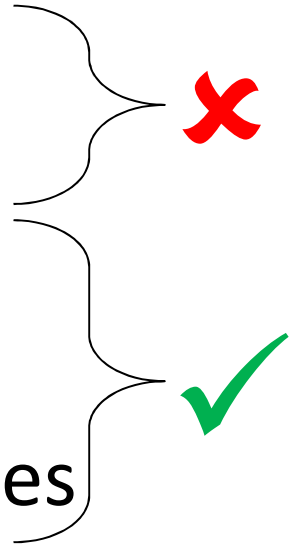
- Armar N particiones de las tablas Ri y Si
  - Se hashea por el atributo de junta
  - Filas de Ri únicamente se juntan con Si
  - Debe tenerse M >= N+1 de memoria para particionar
  - Las particiones de la menor tabla deben tener tamaño menor a M - 2

Join con Hash Grace

B(alumnos) = 10

B(notas) = 22

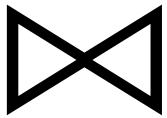
- M = 3: 2 particiones de 5 bloques
- M = 4: 3 particiones de 4 bloques
- M = 5: 4 particiones de 3 bloques
- M = 6: 5 particiones de 2 bloques
- M >=7: M-2 particiones de 1/2 bloques



Loops anidados vs Hash Grace

M	Loops anidados	Hash Grace
3	**230**	
4	**120**	
5	98	**96**
6	**76**	96
7	**54**	96
8	**54**	96
9	**54**	96
10	**54**	96
11	**54**	96
12	**32**	96
13	**32**	96
14	**32**	96
15	**32**	96

Join con índices


alumnos.padron = notas.padron

- Si hay índice por padron en alguna de las tablas
  - Se lee toda la otra tabla
  - Por cada fila leída, se hace index scan en la otra por el padrón leído
- Costo total B(R) + n(R) * Costo_index_scan(S)

Resumen costos por operación

σ

- File Scan: B(R)
- Index Scan: Height(Idx) + 1 *(idx primario)*
  Height(Idx) + ⌈ n(R) / V(Ai,R) ⌉ *(idx sec.)*
  Height(Idx) + ⌈ B(R) / V(Ai,R) ⌉ *(idx clust)*

⋈

- Loops anidados: B(R) + ⌈ B(R) / (M-2) ⌉ * B(S)
- Único loop: B(R) + n(R) * Costo_Index_Scan(S)
- Junta Hash: 3 * (B(R) + B(S))