

L.a)

Raid: replicación o paridad, espacio o paridad distribuida

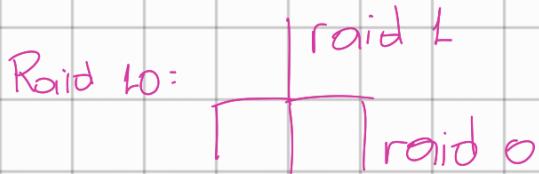
Raid 0: sin redundancia, gano velocidad

Raid 1: con redundancia, gano redundancia

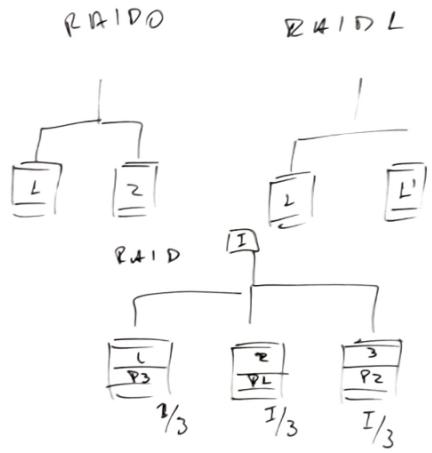
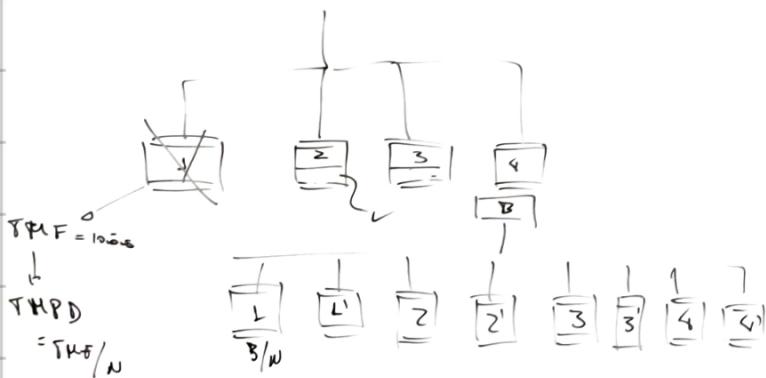
Raid 5: 

disco 1	disco 2	disco 3
paridad 3	paridad 1	paridad 2

 gano velocidad y redundancia y pierde espacio



RAID  
↓  
conjunto.  
redundancia



$$= \overline{TMPP} = \frac{TMF/N}{2} = 2 \times TMF/N$$

1-6)

¿Porque usamos un btree? Porque se ordena solo  
El btree es un indice ordenado porque está dentro de  
un ordeno ordenado (no necesariamente secencial)

## Ventajas de los btrees vs btree?

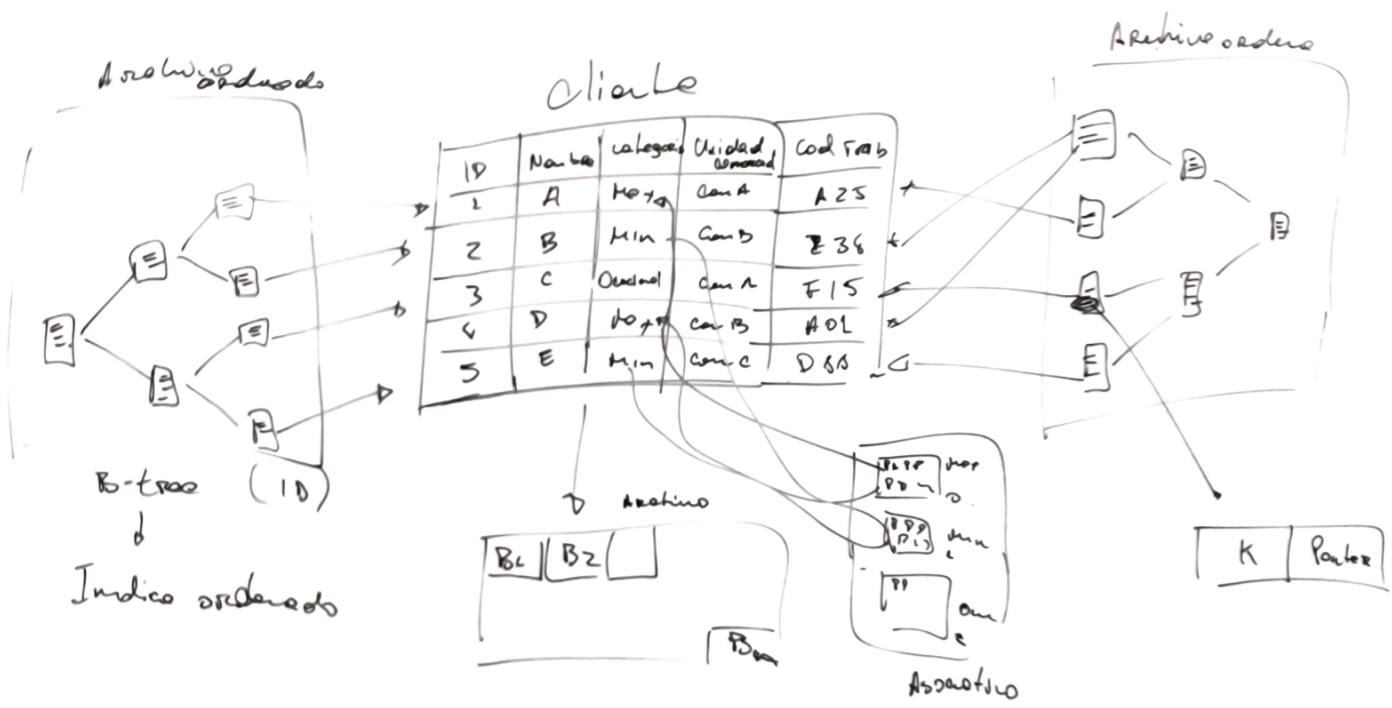
- Complejidad física menor
  - Búsqueda secundaria en las hojas
  - Búsqueda en rango

## Impl. física

## Ordenado General ( igualdad, rango )

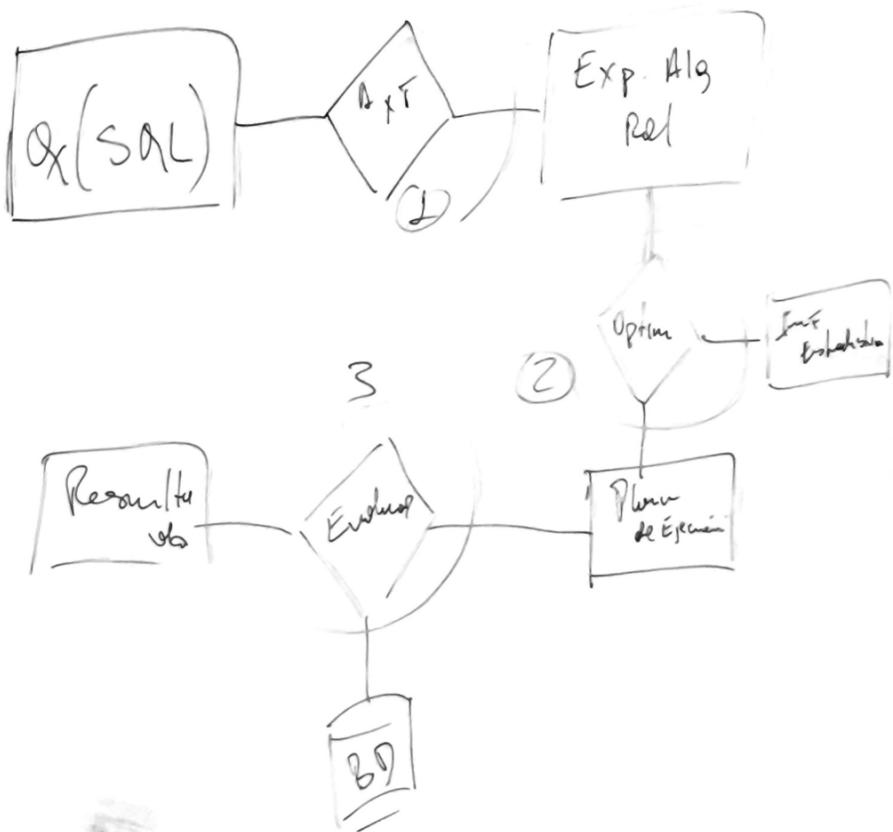
Asociativo (igualdad, no combiñ)

Bitmap (Igualdad, si cambia, multiples claves y multiples condiciones)



L.C)

Porque necesitamos consultar? para procesarla y luego interpretarla, ya que es un lenguaje de alto nivel



L.d)

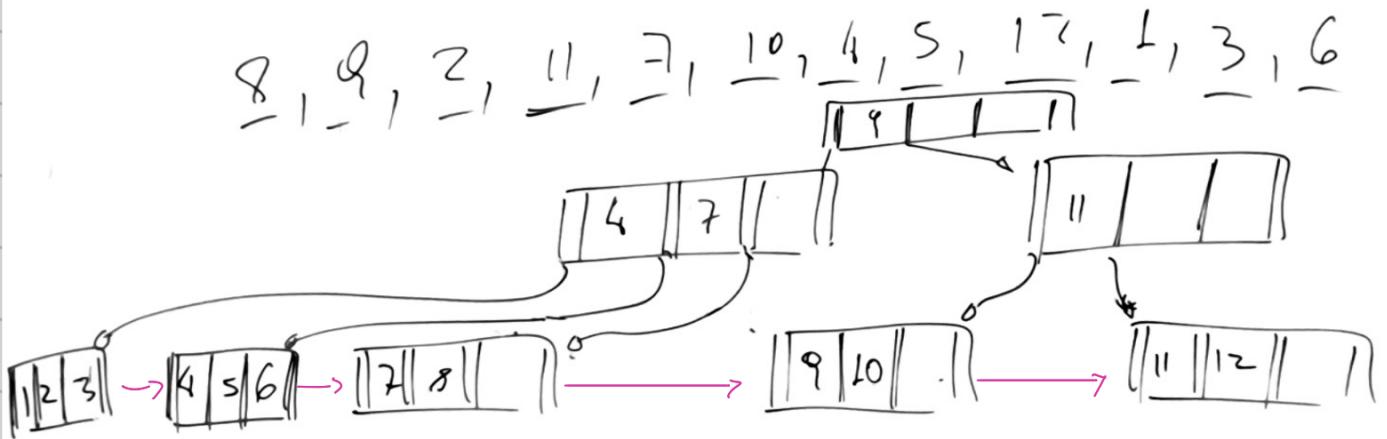
- Es como ordenar una fila: heap, agrupación
- registro de datos no es (clave, puntero)
- registro de datos no se implementan con listas libres, esos son archivos

L.e)

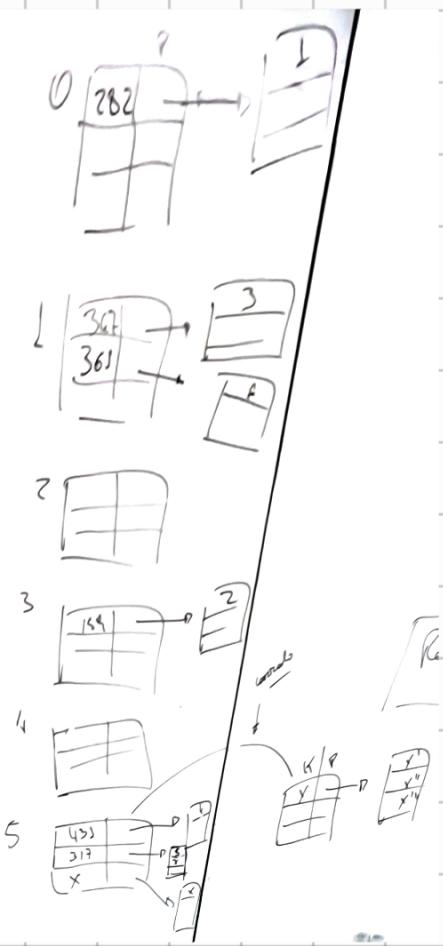
Se autoordenan, balanceadas, capacidad disponible, en bt toda la info está disponible a nivel hoja, búsqueda en rango, hacer join usando merge join, conteo de datos, join basado en hash

## Tema 2)

a) Dibujar cada vez que se rompe el arbol



b)



c)

primario: no necesita cajón de punteros

secundario: si necesita

denso: todos los valores estén indexados

disperso: no todos

Tema 3)

$r_L(A, B, \underline{C})$   $\nearrow FK$   
 $r_z(\underline{C}, D, E)$   $\nearrow PK$

$$R_L \rightarrow 20000 \text{ filas} \rightarrow 100 \text{ filas/Bloque} \rightarrow 200 \text{ Bloques}$$

$$R_z \rightarrow 60000 \text{ filas} \rightarrow 30 \text{ filas/Bloque} \rightarrow 2000 \text{ Bloques}$$

$$\begin{aligned} 40^0 &= 1 \\ 40^1 &= 40 \\ 40^2 &= 1600 \\ 40^3 &= 64000 \\ 40^4 &= - \end{aligned}$$

A) R.B.A.I. s/ Indice Btree en Rz con Modus de 80 puenteros

1.) Bucle Anidado

$$C = n_{R_L} \times B_{R_Z} + B_{R_L} = \\ 20000 \times 200 + 200 =$$

$$= 40000000 + 200 = 40.000.200 \checkmark$$

$$C = B_{R_L} \times B_{R_Z} + B_{R_L} = \\ 200 \times 2000 + 200 =$$

$$= 40000 + 200 = 40.200 \checkmark$$

3) Merge Join + ORcl ( $R_L, M=3$ )

~~$C = B_{R_L} + B_{R_Z} + ORcl(R_L, M=3) =$~~

~~$= B_{R_L} + B_{R_Z} + B_{R_Z} * \lceil \log_2 (B_{R_L}/M) \rceil + 1$~~

~~$= 200 + 2000 + 200 * (\lceil \log_2 (66,66) \rceil + 1)$~~

$$\begin{aligned} C &= B_{R_L} + n_{R_L} * \frac{C_i}{\lceil \log_2 (60000) \rceil} \\ &= 200 + 20000 * \frac{3}{\lceil \log_2 (60000) \rceil} \end{aligned}$$

5) Revisar q/ hash con  $M=15$

$$3 * (B_{R_L} + B_{R_Z}) =$$

$$3 * (200 + 2000) = \\ = 6600 \sim$$

depende si tiene  
o no particionamiento  
recursivo (formula en  
el libro)