# Sistemas de Banco de Dados

## Fundamentos em Bancos de Dados Relacionais

Wladmir Cardoso Brandão
www.wladmirbrandao.com

# INDEXAÇÃO MULTINÍVEL

# Indexação

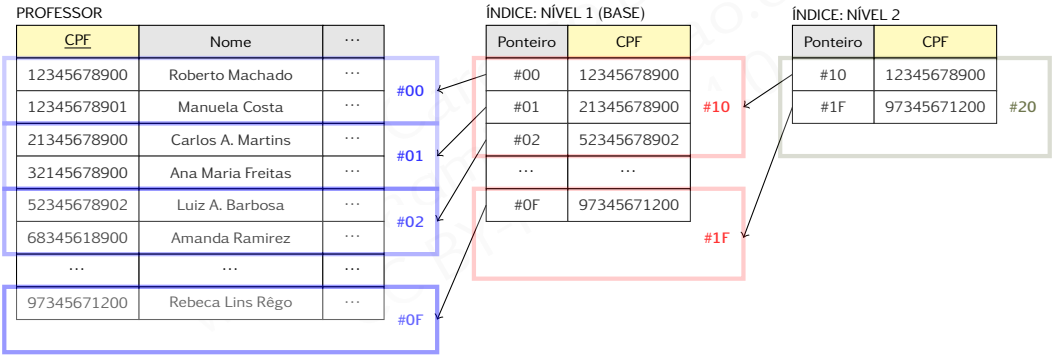
Índice multinível → diferentes níveis de índices são construídos, reduzindo o espaço de pesquisa

- **Estático** → compacto, sem espaço extra em blocos de índice para acomodação de novos registros
  - Tipicamente implementado como arquivo de índice, em que se contrói índices sobre índices
- **Dinâmico** → flexível, com espaço para alocação dinâmica de registros, tornando operações de alteração de dados mais eficientes
  - Tipicamente implementado como árvore B (B Tree) ou B+ (B+ Tree), estruturas baseadas em árvores de pesquisa de múltiplos caminhos, com restrições

# Índice Multinível Estático

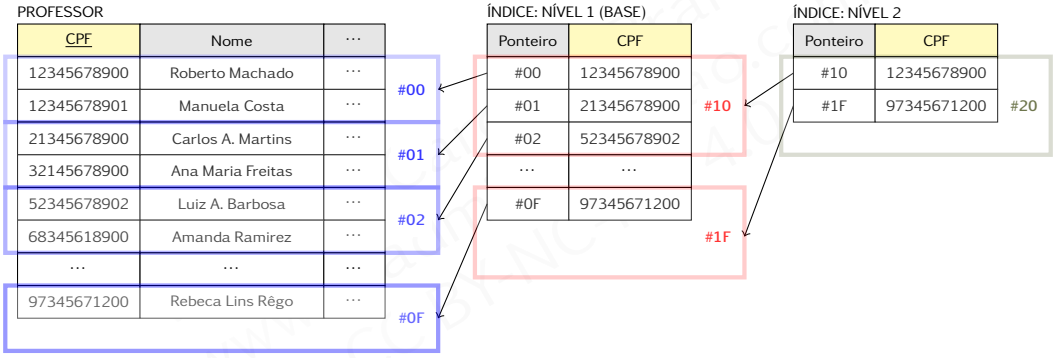
**Nível Base** → índice com valores distintos no campo de ordenação

**Nível Subsequente** → índice primário sobre nível adjacente anterior



# Índice Multinível Estático

Derivam-se níves até que o *n*-ésimo nível seja armazenado em apenas um bloco



# Índice Multinível Estático

Pesquisa → complexidade logarítmica com base > 2, apenas um bloco em cada nível precisa ser acessado

- Base logarítmica → fator de bloco do índice, ou **fan-out** ($fo$)
- Níveis → $h \approx \lceil log_{fo} r_1 \rceil$, onde $r_1$ é o número de registros no nível base
- Acessos a blocos → $A = h$

Alteração → níveis ordenados, custo alto de operações de alteração do campo de indexação

Arquivo **ISAM** (Indexed Sequential Access Method) → arquivo sequencial indexado com índice multinível na chave primária

# Índice Multinível Estático

Para um arquivo indexado de Professor, com 200.000 registros de tamanho fixo de 185B, ordenado pela chave primária CPF e armazenado em um disco com blocos de 4KB, teremos:

- Fator de Bloco $\rightarrow F = \left\lfloor \frac{4KB}{185B} \right\rfloor = \left\lfloor \frac{4 \times 1.024B}{185B} \right\rfloor \approx \lfloor 22,14 \rfloor = 22$
- # Blocos $\rightarrow B = \left\lceil \frac{200.000}{22} \right\rceil \approx \lceil 9.090,91 \rceil = 9.091$
- Espaço $\rightarrow S = 9.091 \times 4KB = 36.364KB \approx 35,51MB$

Pesquisas nesse arquivo demandarão acessos a blocos de disco:

- Pela chave primária $\rightarrow A = \lceil \log_2 9.091 \rceil \approx \lceil 13,15 \rceil = 14$
- Por outro campo $\rightarrow A = 9.091$

# Índice Multinível Estático

Para Professor sendo arquivo ISAM, com CPF de 11B, e ponteiro de bloco ocupando 16B, teremos:

- Fator de Bloco $\rightarrow F_M = \left\lfloor \frac{4KB}{11B+16B} \right\rfloor = \left\lfloor \frac{4 \times 1.024B}{27B} \right\rfloor \approx \lfloor 151,70 \rfloor = 151$
- # Blocos $\rightarrow B_{M1} = \left\lceil \frac{9.091}{151} \right\rceil \approx \lceil 60,20 \rceil = 61$, $B_{M2} = \left\lceil \frac{61}{151} \right\rceil \approx \lceil 0,40 \rceil = 1$
- Espaço $\rightarrow S_M = (61+1) \times 4KB = 248KB$

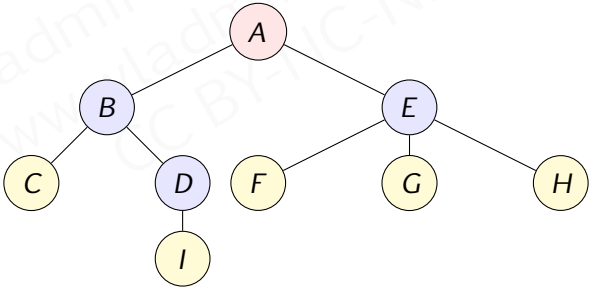
Pesquisas nesse arquivo só podem ser realizadas pelo campo de indexação e demandarão acessos a blocos de disco:

- $A_M = h \approx \lceil \log_{151} 9.091 \rceil \approx \lceil 1,81 \rceil = 2$
- $+1$ acesso para recuperar o registro no arquivo indexado

# Índice Multinível Dinâmico

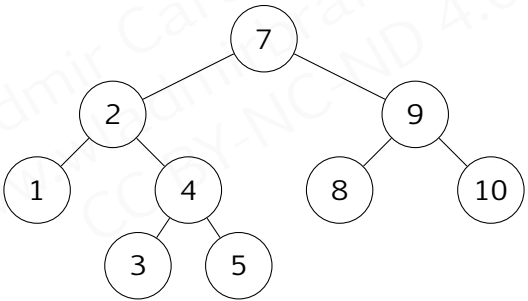
**Árvore (*Tree*)** → estrutura hierárquica de nós (elementos) conectados

- Raiz → nó sem pai, de nível zero
- Folha → nó sem filhos
- Interno → nó não folha e não raiz
- O nível de um nó na árvore é o nível do seu pai mais um
- Subárvore → árvore formada por um nó e todos os seus descendentes



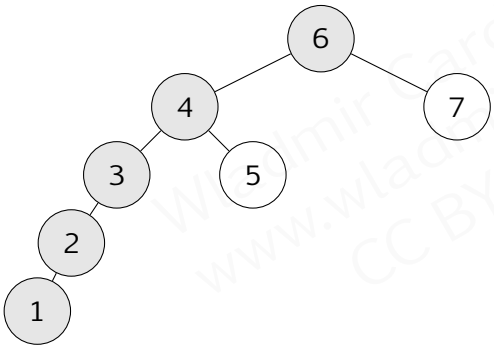
**Árvore de Busca (*Search Tree*)** → nós com restrições para eficiência em busca

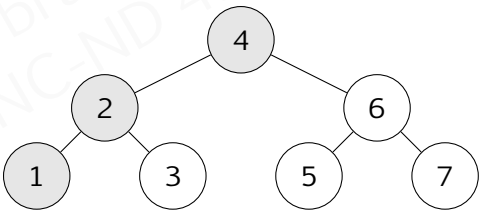
- Binária → nó tem no máximo dois filhos
  - Chave não pode ser menor que qualquer outra em subárvores da esquerda
  - Chave não pode ser maior que qualquer outra em subárvores da direita



**Árvore de busca binária (BST)** → balanceamento fundamental para eficiência

- Balanceamento → altura de árvore $\approx \lceil \log_2 n \rceil$, onde $n$ = número de nós
  - Subárvores esquerda e direita com ≈ mesmo número de nós

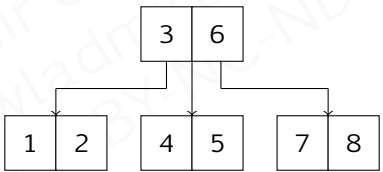


# Índice Multinível Dinâmico

**Árvore de busca de múltiplos caminhos (*m-way*)** → generalização de BST

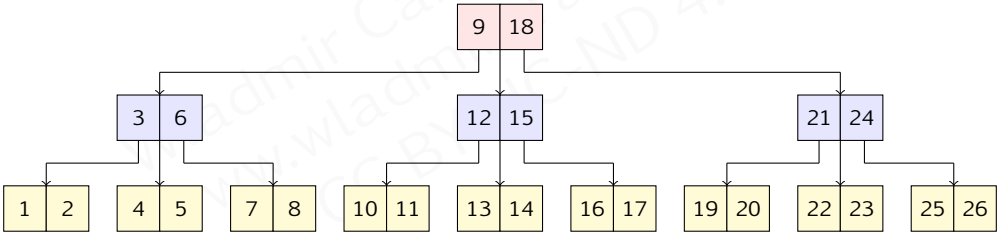
- Cada nó tem $m$ filhos
- Multiplicidade → cada nó contém $m - 1$ elementos
  - $h \leq n \leq m^h - 1$, onde $h =$ altura e $n =$ número de nós
- Balanceamento → $h \approx \lceil \log_m n \rceil$



# Índice Multinível Dinâmico

B TREE → *m-way* com restrições que tornam busca e atualização muito eficientes

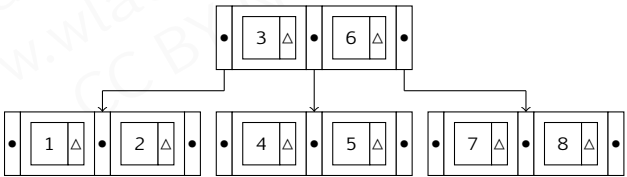
- Nó raiz, não folha, tem ao menos dois filhos
- Nó interno tem ao menos $\lceil m/2 \rceil$ filhos
- Nós folha estão no mesmo nível



# Índice Multinível Dinâmico

B Tree → nós ao menos meio cheios, ≈ 69% quando árvore estabiliza

- Pesquisa → eficiente, poucos níveis e perfeitamente balanceada
- Alteração → eficiente, espaço para acomodar novos registros
- Ideal para armazenamento em memória secundária
  - Tipicamente configurada para que um nó ocupe um bloco em disco
  - Elemento → campo de indexação + ponteiro de bloco (△)
  - Nó com *m* ponteiros de nó (•), um para cada filho



# Índice Multinível Dinâmico

Arquivo de PROFESSOR, com índice B TREE em DEPARTAMENTO de 8B, com um nó ocupando um bloco de disco, ponteiro de nó de 12B, ponteiro de bloco de 16B

- Tamanho do elemento $\rightarrow (8B + 16B) = 24B$
- Nó ocupa 1 bloco $\rightarrow 4KB \geq ((m-1) \times (24B + 12B)) + 12B$
  - # Elementos por nó $\rightarrow (m-1) = \left\lfloor \frac{4KB - 12B}{24B + 12B} \right\rfloor = \left\lfloor \frac{4.084B}{36B} \right\rfloor \approx \lfloor 113,44 \rfloor = 113$
  - Ordem da árvore $\rightarrow m = 113 + 1 = 114$
- Altura da árvore $\rightarrow h \approx \lceil \log_{114} 200.000 \rceil \approx \lceil 2,57 \rceil = 3$

| Nível | # Nós | # Registros | # Ponteiros de Nó |
|---|---|---|---|
| 0 | 1 | 113 | 114 |
| 1 | 114 | 12.882 | 12.996 |
| 2 | 12.996 | 1.468.584 | - |

# Índice Multinível Dinâmico

Considerando uma ocupação de nós em 69%:

- Fator de Bloco $\rightarrow F_B = \lceil 113 \times 0.69 \rceil \approx \lceil 77,97 \rceil = 78$
- # Blocos $\rightarrow B_B = \left\lceil \frac{200.000}{78} \right\rceil \approx \lceil 2.564,10 \rceil = 2.565$
- Espaço $\rightarrow S_B = 2.565 \times 4KB = 10.260KB \approx 10,01MB$

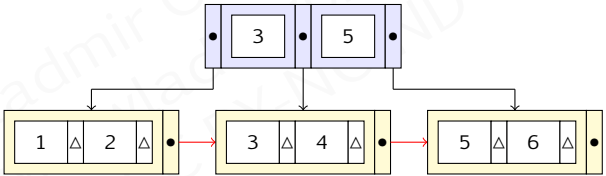
Pesquisas nesse índice só podem ser realizadas pelo campo de indexação e demandarão acessos a blocos de disco:

- $A_B = h = 3$
- $+1$ acesso para recuperar o registro no arquivo indexado

# Índice Multinível Dinâmico

**B+ Tree** → extensão B Tree com restrições que tornam ainda mais eficientes a busca e a remoção

- Nó Índice → raiz ou interno que armazena exclusivamente chave
- Nó Registro → folha que armazena registro de índice
- Nós folha em lista encadeada ordenada



# Índice Multinível Dinâmico

Arquivo de Professor, com índice B+ Tree em Departamento de 8B, com um nó ocupando um bloco de disco, ponteiro de nó de 12B, ponteiro de bloco de 16B

- Nó Índice
  - Tamanho do elemento $\rightarrow 8B$
  - Nó ocupa 1 bloco $\rightarrow 4KB \geq ((m-1) \times (8B + 12B)) + 12B$
    - Elementos por nó $\rightarrow (m-1) = \left\lfloor \frac{4KB - 12B}{8B + 12B} \right\rfloor = \left\lfloor \frac{4.084B}{20B} \right\rfloor \approx \lfloor 204,2 \rfloor = 204$
    - Ordem da árvore $\rightarrow m = 204 + 1 = 205$
  - Altura da árvore $\rightarrow h \approx \lceil log_{205} 200.000 \rceil \approx \lceil 2,29 \rceil = 3$
  - Altura de nós índice $\rightarrow h_i = h - 1 = 2$

# Índice Multinível Dinâmico

| Nível | # Nós | # Registros | # Ponteiros de Nó |
|---|---|---|---|
| 0 | 1 | 204 | 205 |
| 1 | 205 | 41.820 | 42.025 |
| 2 | 42.025 | 8.573.100 | - |

- Nó Registro
  - Tamanho do elemento $\rightarrow (8B + 16B) = 24B$
  - Nó ocupa 1 bloco $\rightarrow 4KB \geq ((m-1) \times (24B)) + 12B$
    - Elementos por nó $\rightarrow (m-1) = \left\lfloor \frac{4KB - 12B}{24B} \right\rfloor = \left\lfloor \frac{4.084B}{24B} \right\rfloor \approx \lfloor 170,16 \rfloor = 170$

# Índice Multinível Dinâmico

Considerando uma ocupação de nós em 69%:

- Fator de Bloco $\rightarrow F_{B+} = \lceil 170 \times 0.69 \rceil \approx \lceil 117,30 \rceil = 118$
- # Blocos $\rightarrow B_{B+} = \left\lceil \frac{200.000}{118} \right\rceil \approx \lceil 1.694,91 \rceil = 1.695$
- Nível 0 a $h_i - 1 \rightarrow 1.695$ ponteiros
  - # Nós = # Blocos = $\rightarrow \left\lceil \frac{1.695}{\lceil (204 \times 0.69) \rceil + h_i} \right\rceil \approx \left\lceil \frac{1.695}{143} \right\rceil \approx \lceil 11,85 \rceil = 12$
- Espaço $\rightarrow S_{B+} = (1.695 + 12) \times 4KB = 6.828KB \approx 6,67MB$

Pesquisas pelo campo de indexação demandam acessos a blocos:

- $A_{B+} = h = 3$
- $+1$ acesso para recuperar o registro no arquivo indexado

# Referências Bibliográficas


[1] Elmasri, Ramez; Navathe, Sham. *Fundamentals of Database Systems*. 7ed. Pearson, 2016.

[2] Silberschatz, Abraham; Korth, Henry F.; Sudarshan, S. *Database System Concepts*. 6ed. McGraw-Hill, 2011.

[3] Date, Christopher J. *An Introduction to Database Systems*. 8ed. Pearson, 2004.