

Banco de dados II

04 - Indexação Baseada em Hash

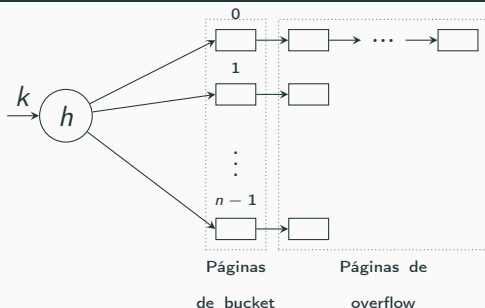
Marcos Roberto Ribeiro



Instituto Federal Minas Gerais - Campus Bambuí

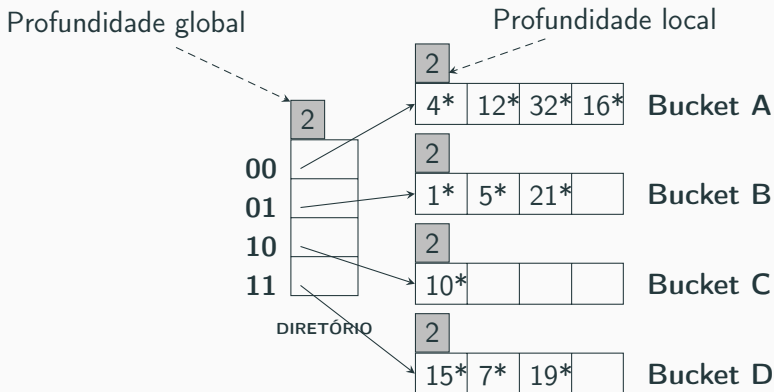
2018

Hash Estático



- $h(k) = k \bmod n$ onde n é o número de buckets (é o mesmo que pegar os $\log_2 n$ últimos bits de k)
- Páginas de overflow são criadas quando não há espaço na página de bucket
- O índice é estático porque o número de páginas de bucket não muda (podem ser criadas assim que o índice é criado)
- O problema é que podem aparecer longas cadeias de overflow causando lentidão nas pesquisas

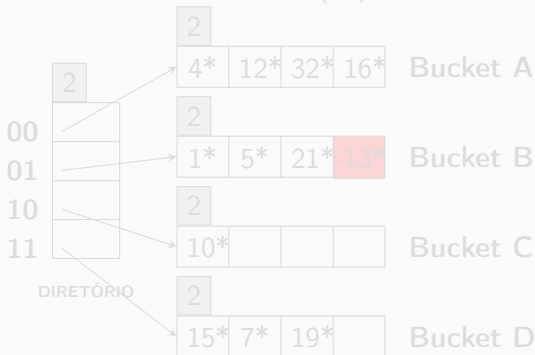
Hash Extensível



- O hash extensível utiliza um diretório de ponteiros para os buckets
- Este diretório é duplicado se ocorrer algum overflow

Inserções em Hash Extensível

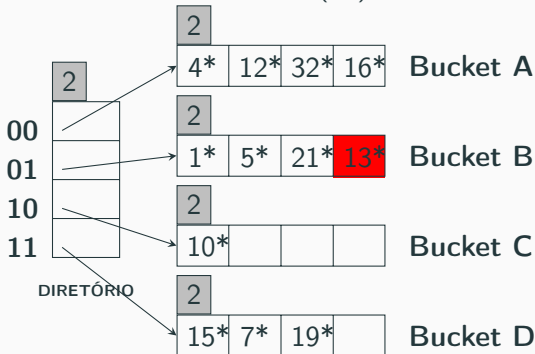
- Para inserir uma entrada calculamos a função hash sobre a chave considerando os últimos n bits
- Onde n é a profundidade local
- Exemplo: Inserir a entrada 13^* , temos $h(13) = 01$



- Neste caso a entrada cabe no bucket, então apenas gravamos a entrada na posição livre

Inserções em Hash Extensível

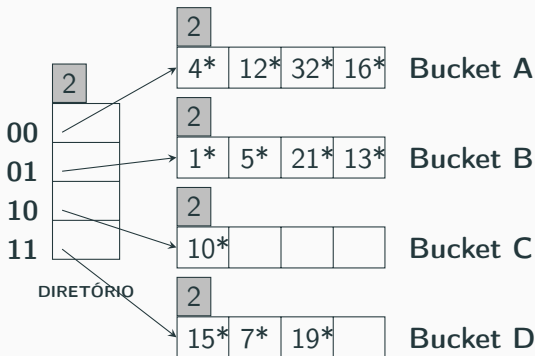
- Para inserir uma entrada calculamos a função hash sobre a chave considerando os últimos n bits
- Onde n é a profundidade local
- Exemplo: Inserir a entrada 13^* , temos $h(13) = 01$



- Neste caso a entrada cabe no bucket, então apenas gravamos a entrada na posição livre

Inserção com Overflow

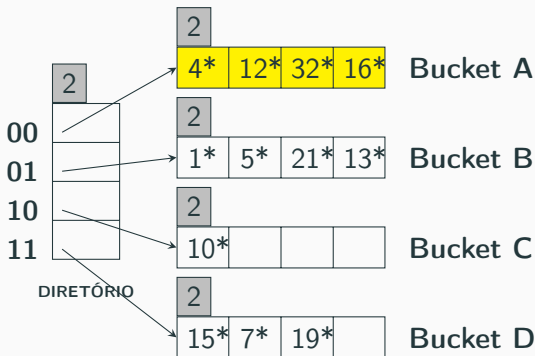
- Agora considere a inserção da entrada 20*
- $h(20) = 00$



- Porém, o bucket A está cheio

Inserção com Overflow

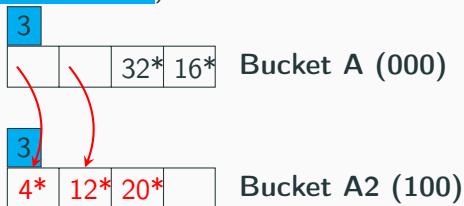
- Agora considere a inserção da entrada 20^*
- $h(20) = 00$



- Porém, o bucket A está cheio

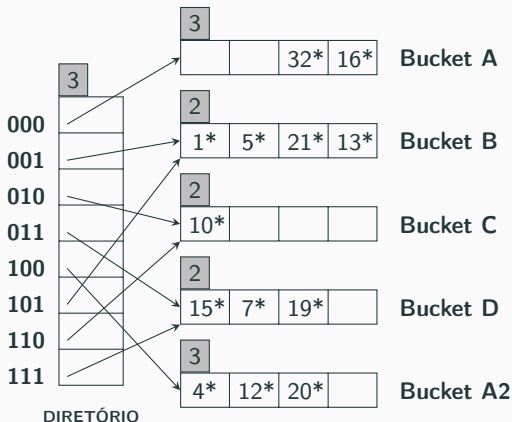
Inserção com Overflow - Novo Bucket

- Aumentamos a profundidade local do bucket A para 3
- Criamos o novo bucket A2 e redistribuímos os registros usando a nova profundidade (3 últimos bits)



Inserção com Overflow - Duplicação do Tamanho do Diretório

- Devemos duplicar o tamanho do diretório sempre que há uma profundidade local maior do que a profundidade global
- Os buckets que não foram divididos permanecem com a profundidade local inalterada e recebem dois ponteiros



Exclusões e Valores Duplicados



- Durante a exclusões poderíamos excluir buckets vazios
- Porém, na prática, isto não é viável porque os arquivos tendem a crescer
- Para lidar com valores duplicados é necessário ter páginas de overflow

Hashing Linear

- O hashing linear reduz as divisões e proporciona maior ocupação dos buckets do que o hashing extensível
- Este tipo de hashing utiliza uma família de funções hash h_0, h_1, h_2, \dots
- O intervalo de h_{i+1} é o dobro do intervalo de h_i
- Por exemplo, supondo 32 buckets iniciais ($N_0 = 32 = 2^5$):
 - $h_0(k) = k \bmod 32 \Rightarrow d_0 = 5$ (5 bits)
 - $h_1(k) = k \bmod 64 \Rightarrow d_1 = d_0 + 1 = 5 + 1 = 6$ (6 bits, $N_1 = 64$)
 - \vdots

Exemplo de Hash Linear

- Iniciamos com 4 buckets ($N_0 = 4$) e nível 0 (zero)
- $N = N_0 * 2^{\text{nível}} = 4 * 2^0 = 1$ (número de buckets atual)

	h_1	h_0	
000	00		32* 44* 36* 
001	01		9* 25* 5* 
010	10		14* 18* 10* 30*
011	11		31* 35* 7* 11*

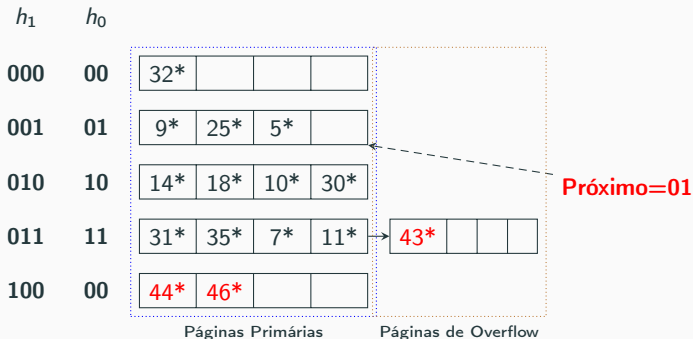
Páginas Primárias

Próximo=00

- Podem ocorrer páginas de overflow
- Dividimos o bucket apontando por **próximo** sempre que ocorre overflow
- Redistribuímos as entradas usando $h_{\text{nível}+1}$ e incrementamos o apontador **próximo**

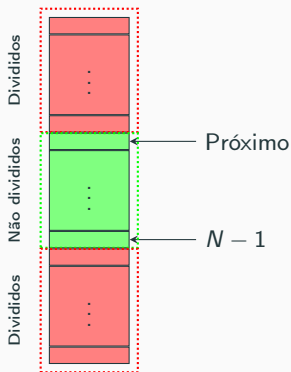
Inserção com Overflow

- Ao inserir 43^* , temos $h_0(43) = 11$
- Como o bucket **11** está cheio, criamos um página de overflow para incluir a entrada 43^*
- Dividimos o bucket **00** apontado por próximo e incrementamos próximo para **01**

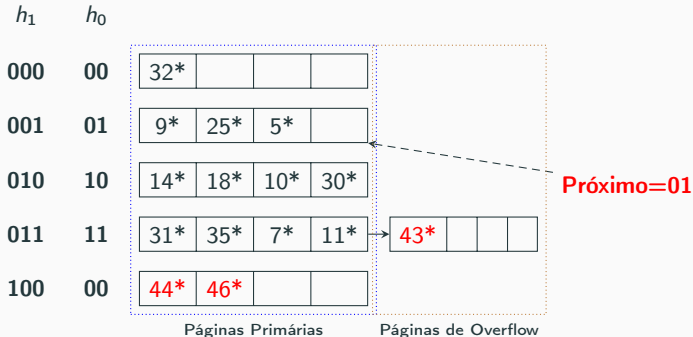


Rodadas de Divisão

- O apontador próximo circula pelos buckets até que todos sejam divididos
- Quando isto acontece, passamos a usar as funções hash do próximo nível



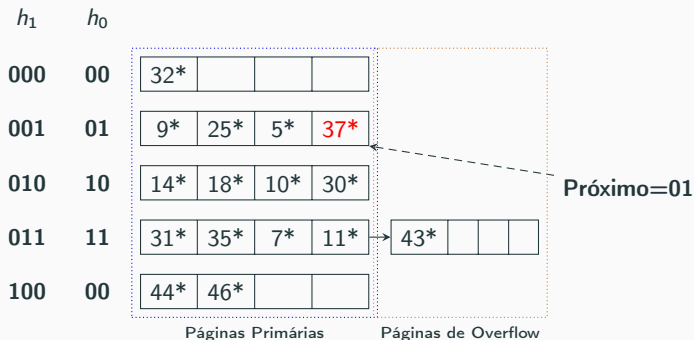
Pesquisa



- Primeiro usamos a função $h_{\text{nível}}$
- Se $\text{próximo} \leq h_{\text{nível}} < N_{\text{nível}}$, já encontramos o bucket. Exemplo: $h_0(18) = 10(2)$, $\text{próximo} \leq 10 < 100$, (bucket 10)
- Caso contrário, aplicamos $h_{\text{nível}+1}$. Exemplo:
 $h_0(32) = 00$, $00 < \text{próximo}$, $h_1(32) = 000$
 $h_0(44) = 00$, $00 < \text{próximo}$, $h_1(44) = 100$

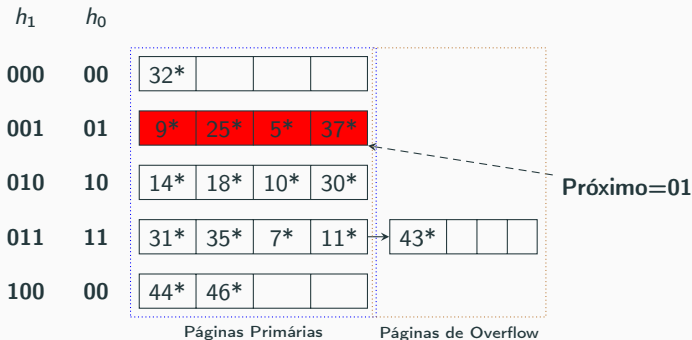
Inserção sem Overflow

- Nem sempre ocorre divisão, por exemplo, na inserção da entrada 37*



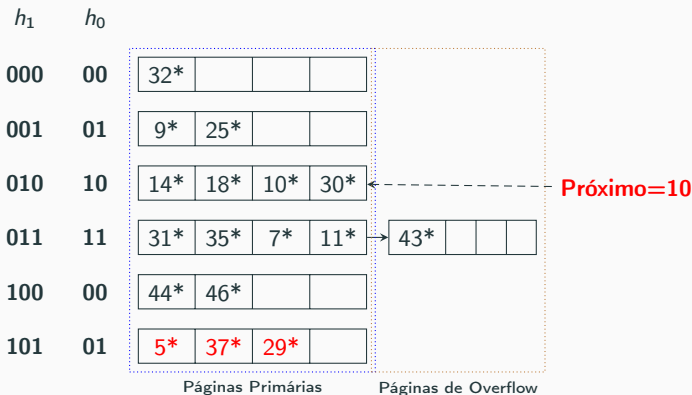
Inserção com Divisão e Sem Overflow

- Se o bucket que causar overflow for aquele apontado pelo **próximo**, dividimos o bucket e evitamos este overflow
- Exemplo: inserir a entrada 29^* , $h_0(29) = 01$ (está cheio e é o próximo)



Inserção com Divisão e Sem Overflow

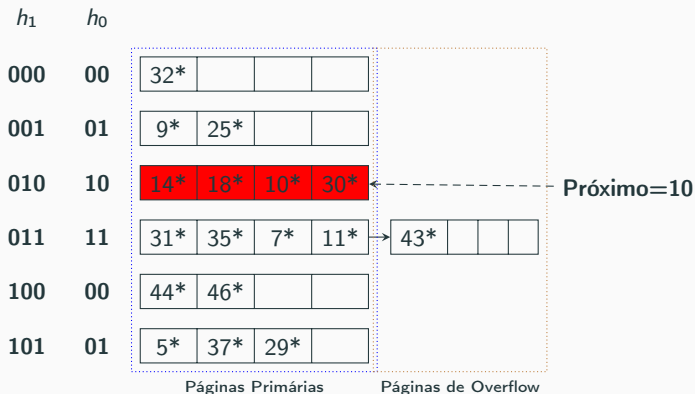
- Se o bucket que causar overflow for aquele apontado pelo **próximo**, dividimos o bucket e evitamos este overflow
- Exemplo: inserir a entrada 29^* , $h_0(29) = 01$ (está cheio e é o próximo)



- Inserir as entradas 22^* , 66^* e 34^*

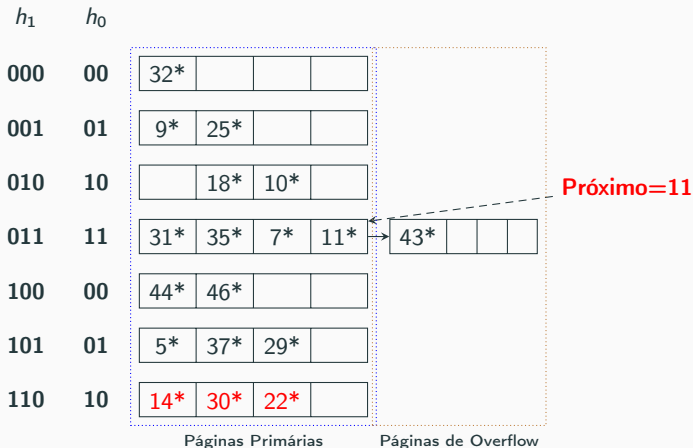
Inserção da Entrada 22*

- $h_0(22) = 10$ (está cheio e é o próximo)



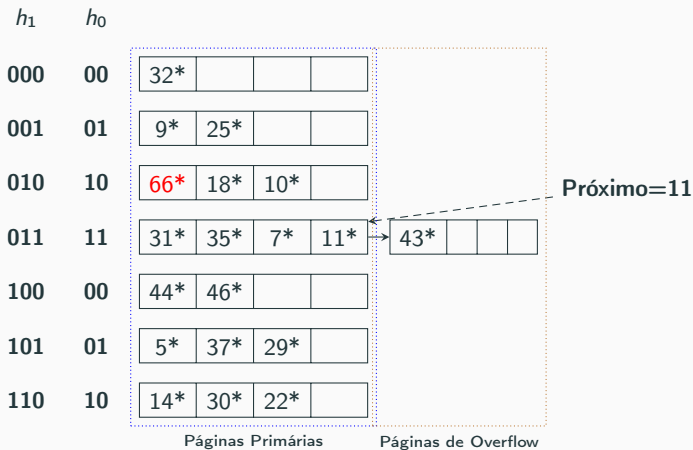
Inserção da Entrada 22*

- $h_0(22) = 10$ (está cheio e é o próximo)



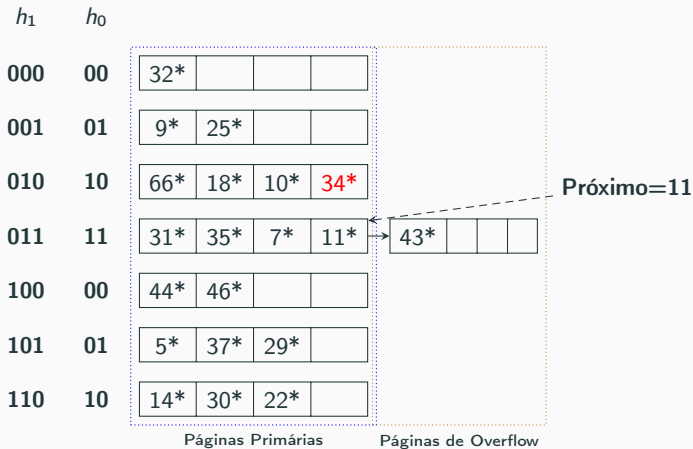
Inserção da Entrada 66*

- $h_0(66) = 10$, como $01 < \text{próximo}$, aplicamos h_1
- $h_1(66) = 010$



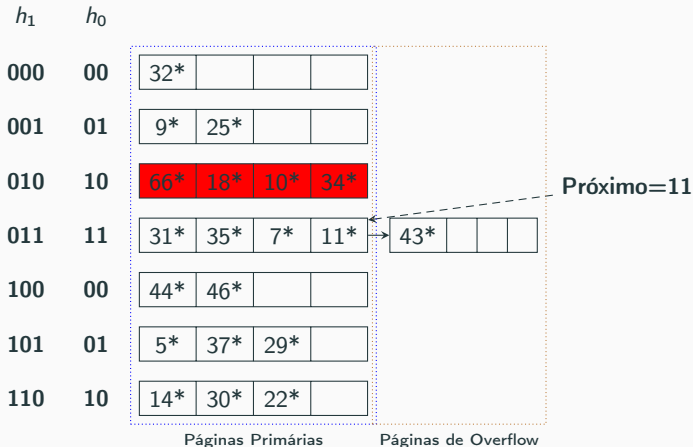
Inserção da Entrada 34*

- $h_0(34) = 10$, como $01 < \text{próximo}$, aplicamos h_1
- $h_1(34) = 010$



Início de Nova Rodada

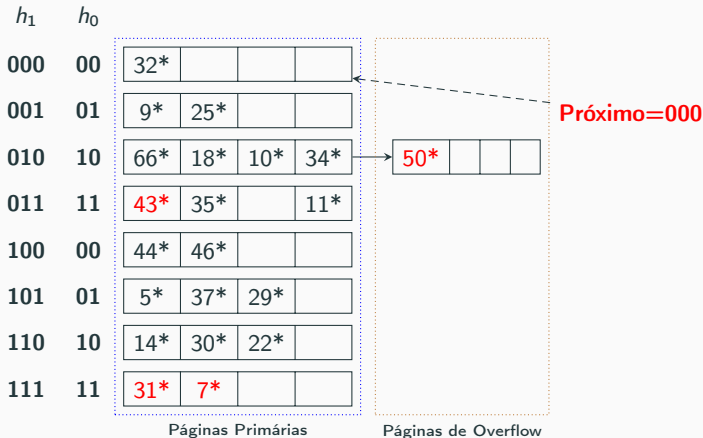
- Vamos inserir a entrada 50*, $h_0(50) = 10$, como $01 < \text{próximo}$, aplicamos h_1
- $h_1(50) = 010$, mas bucket 010 está cheio, precisamos de overflow



- O próximo é último bucket do nível 0, passamos então para o próximo nível e mudamos o próximo para o primeiro bucket novamente

Início de Nova Rodada

- Vamos inserir a entrada 50*, $h_0(50) = 10$, como $01 < \text{próximo}$, aplicamos h_1
- $h_1(50) = 010$, mas bucket 010 está cheio, precisamos de overflow



- O **próximo** é último bucket do nível 0, passamos então para o próximo nível e mudamos o **próximo** para o primeiro bucket novamente

Referências I



Date, C. J. (2004).

Introdução a sistemas de bancos de dados.

Elsevier, Rio de Janeiro.



Elmasri, R. and Navathe, S. B. (2011).

Sistemas de banco de dados.

Pearson Addison Wesley, São Paulo, 6 edition.



Ramakrishnan, R. and Gehrke, J. (2008).

Sistemas de gerenciamento de banco de dados.

McGrawHill, São Paulo, 3 edition.



Silberschatz, A., Korth, H. F., and Sudarshan, S. (2007).

Sistema de bancos de dados.