ÍNDICES EM BANCOS DE DADOS

Sérgio Mergen

Introdução

- Na aula passada, vimos uma implementação básica de um índice
  - A implementação consiste no uso de dois níveis de RecordManagers
    - Um para o índice
    - Um para os dados
  - A implementação é simples
    - Utiliza apenas uma camada de índice

Introdução

- Na prática, usa-se árvores B+ para realizar a manipulação das páginas de registros e de índices multinível

- Essa estrutura facilita
  - Inserção /remoção
  - Criação de páginas de estouro
  - Reorganização das páginas
  - Distribuição dos registros entre páginas vizinhas
  - Gerenciamentos dos múltiplos níveis do índice

Sumário

- Árvores B+
- Tipos de Árvores B+
- Índices no DBest

Árvores B+

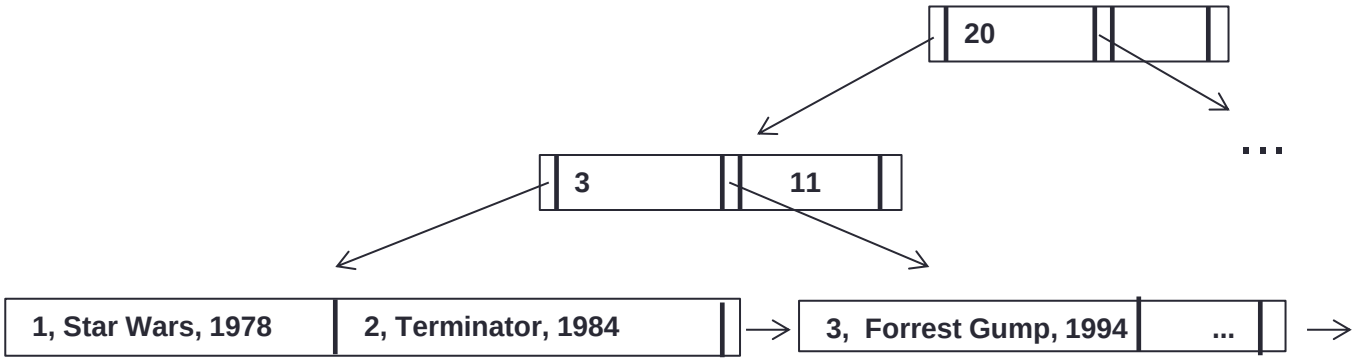
- Características:
  - Armazena pares do tipo <chave, valor>
  - Usada para
    - Organizar os valores por ordem de chave
    - Localizar os valores com base em filtros sobre a chave

- Nas árvores B, nem todas chaves estão no nível folha

- Nas árvores B+, todas chaves estão no nível folha
  - O que permite acessar todas chaves percorrendo apenas os nós no último nível

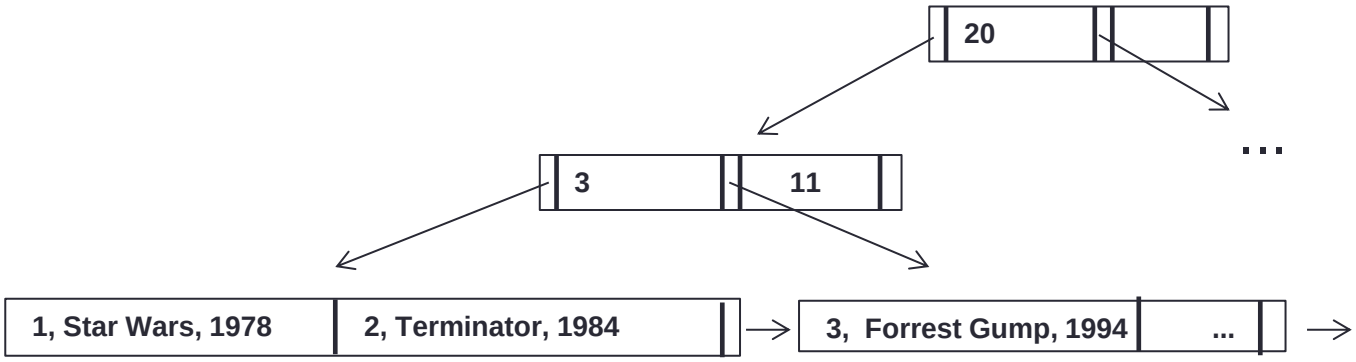
Árvores B+

- Pares armazenados na árvore
  - Chave: idM
  - Valor: registro de filme



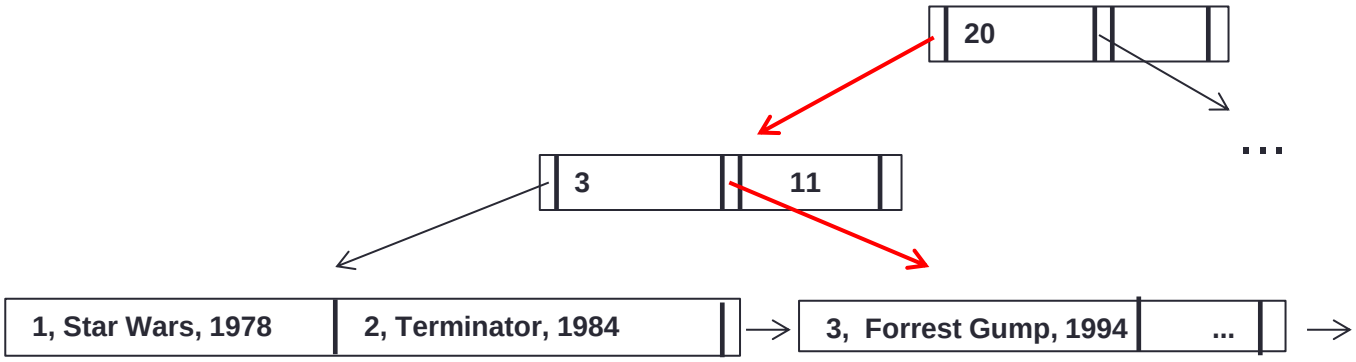
Árvores B+

- Os nós correspondem à páginas
  - Nós internos possuem roteamentos com base na chave (idM)
  - O nível folha possui todos os pares, ordenados pelo idM



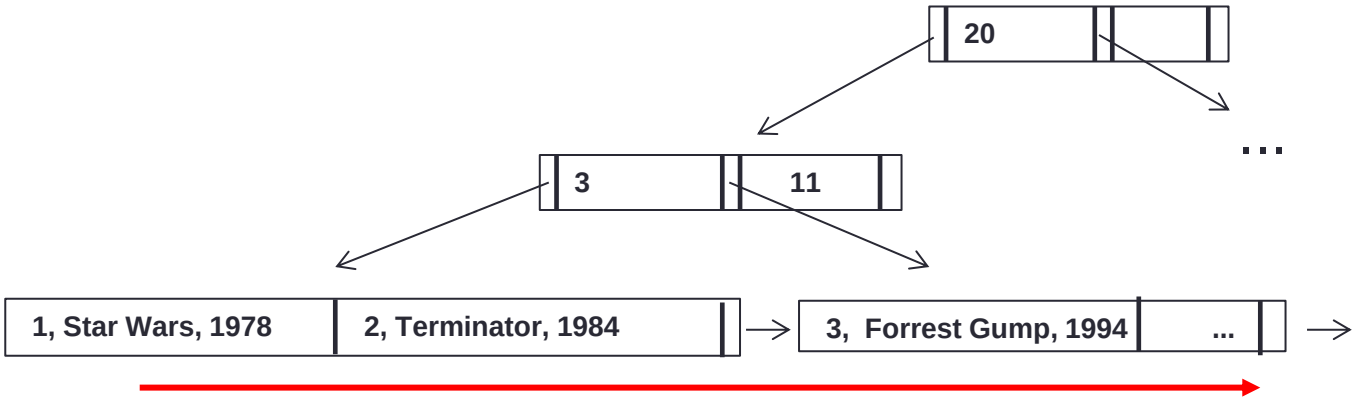
Árvores B+

- Uma árvore B+ pode ser usada para buscar registros com base em algum critério de filtragem
  - Para isso, deve-se navegar desde o nó raiz até o nó folha correspondente
    - Ex. acessar o registro que tenha idM = 3



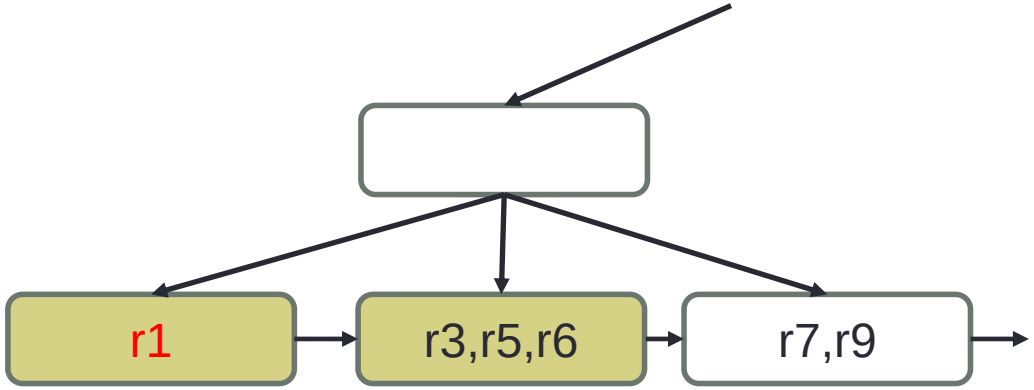
Árvores B+

- As inserções e remoções em uma árvore B+ usam métodos que asseguram que os registros no nível folha estejam ordenados pela chave de busca
  - Assim, pode-se ler os registros em ordem da chave de busca acessando o nível folha da esquerda para a direita

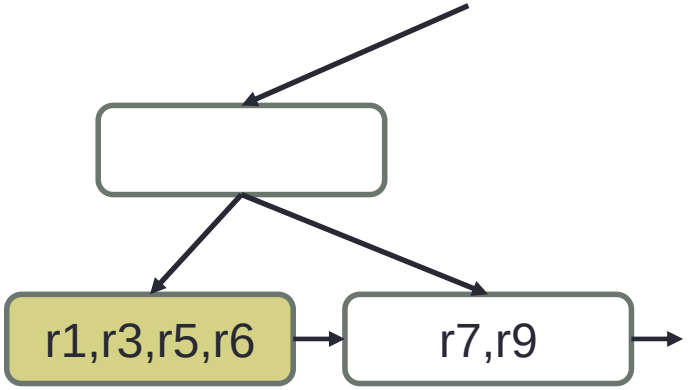


Árvores B+

- **Remoção:** deixa espaço livre na página. Caso o nó fique com poucos registros, ele sofre merge (mescla com uma página vizinha)
- Ex. a página com apenas um registro será mesclada



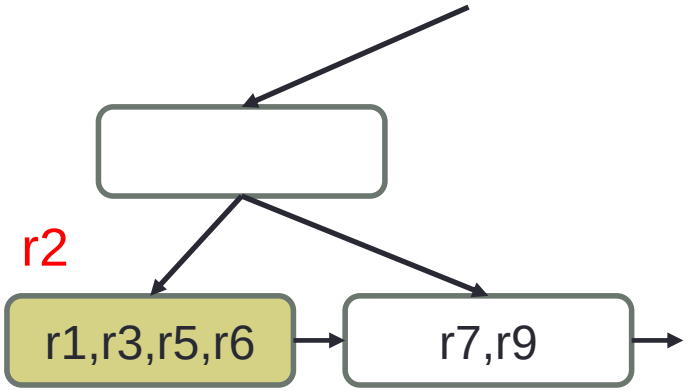
antes



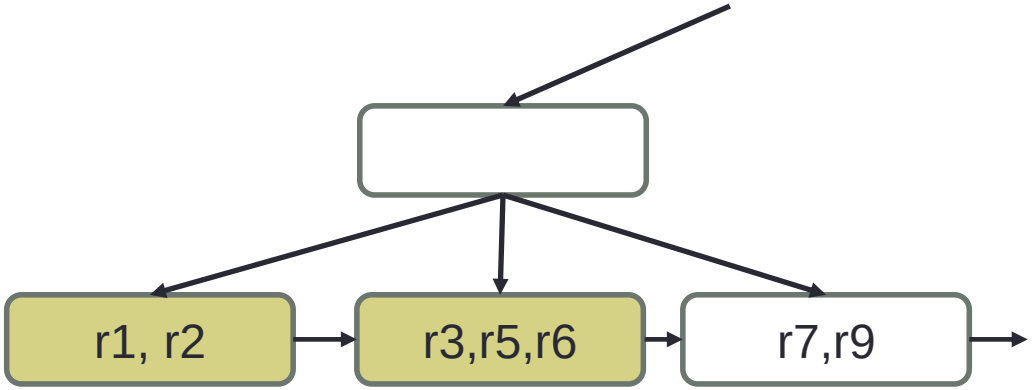
depois

Árvores B+

- **Inserção e Update:** O tamanho máximo de registro corresponde a pouco menos da metade do tamanho da página
  - Caso não haja espaço, a página sofre split (é dividida em duas)
- Ex. a inserção do registro r2 faria a primeira página estourar a sua capacidade



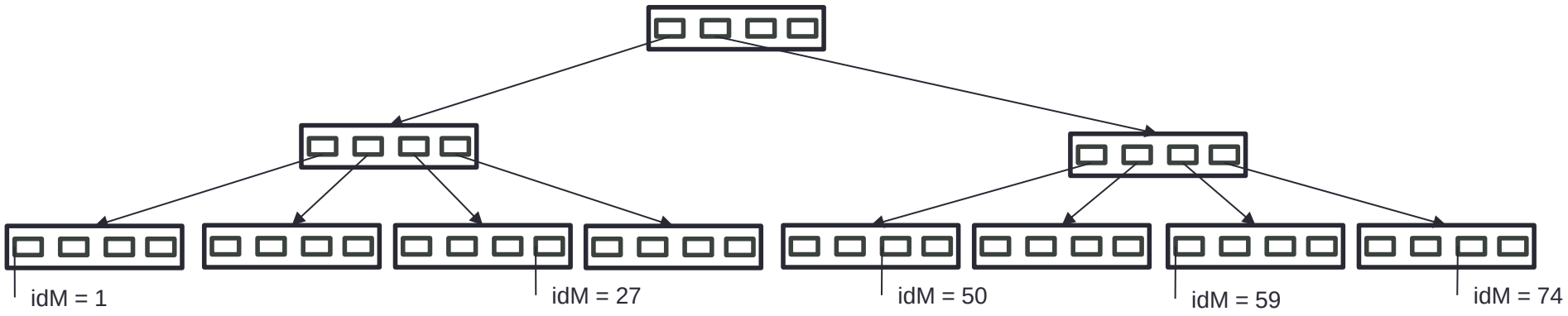
Antes do split



Depois do split

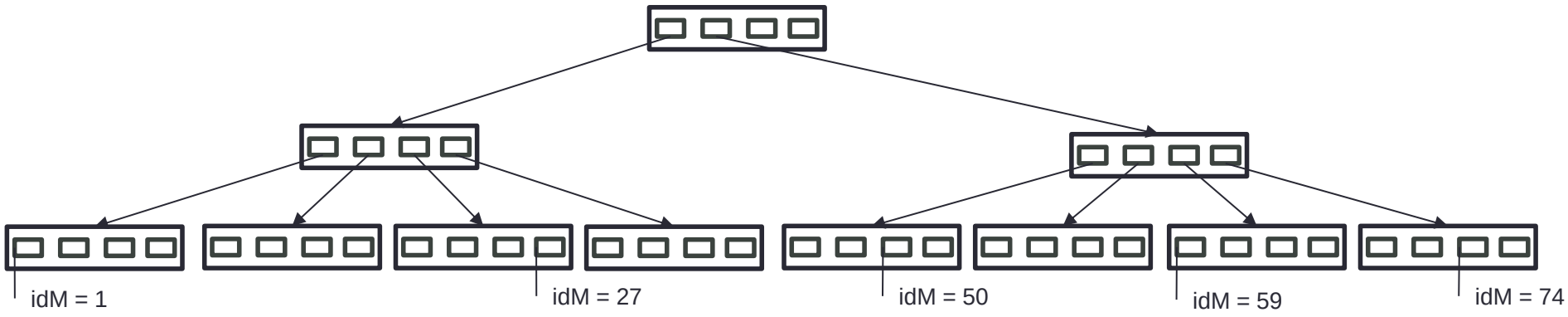
Árvores B+

- Exemplo maior
  - Cada nó possui múltiplas entradas
    - Uma entrada possui uma chave de busca e um ponteiro para um nó do nível inferior
    - No nível folha, em vez de ponteiro, é armazenado o valor que corresponde à chave de busca
    - O nível folha possui todas as chaves de busca, ordenadas por idM



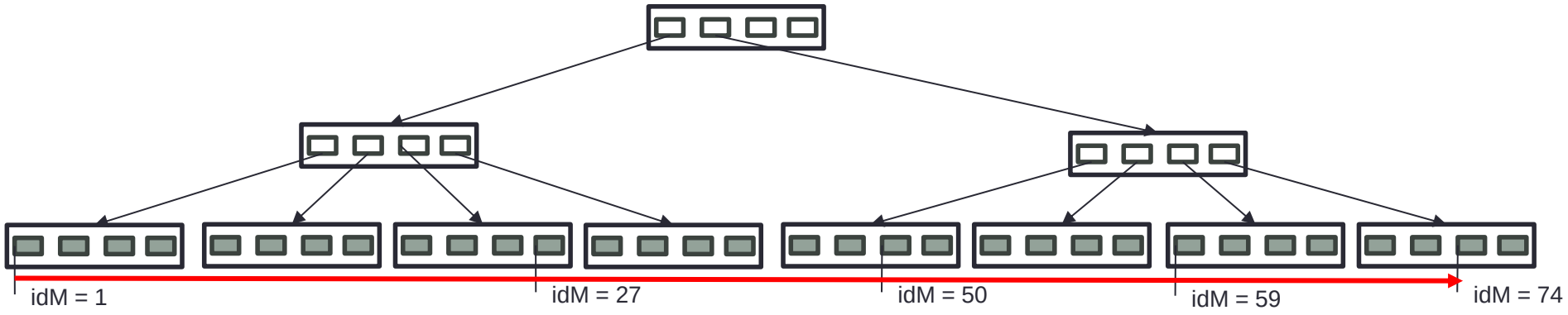
Árvores B+

- A seguir veremos exemplos de buscas que podem ser feitas sobre uma árvore B+



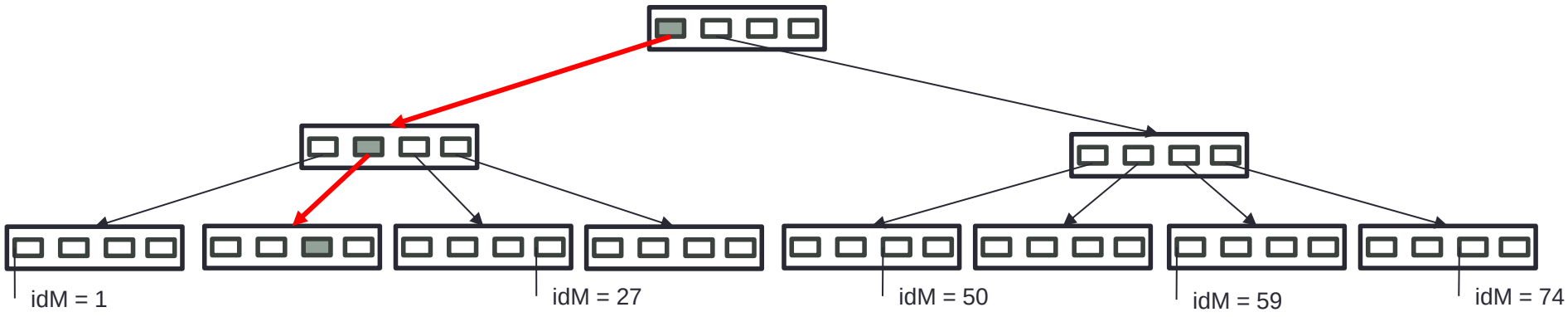
Árvores B+

- Para ler todos os registros
  - A busca localiza o nó folha mais à esquerda
  - Todos os nós seguintes são lidos



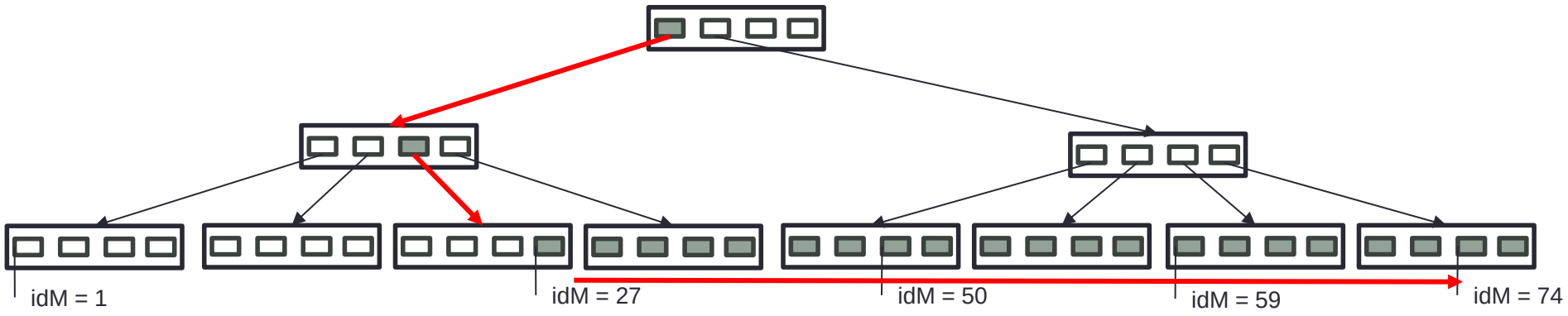
Árvores B+

- Filtro: idM = 9
  - A busca leva ao nó folha que possui idM=9, caso exista



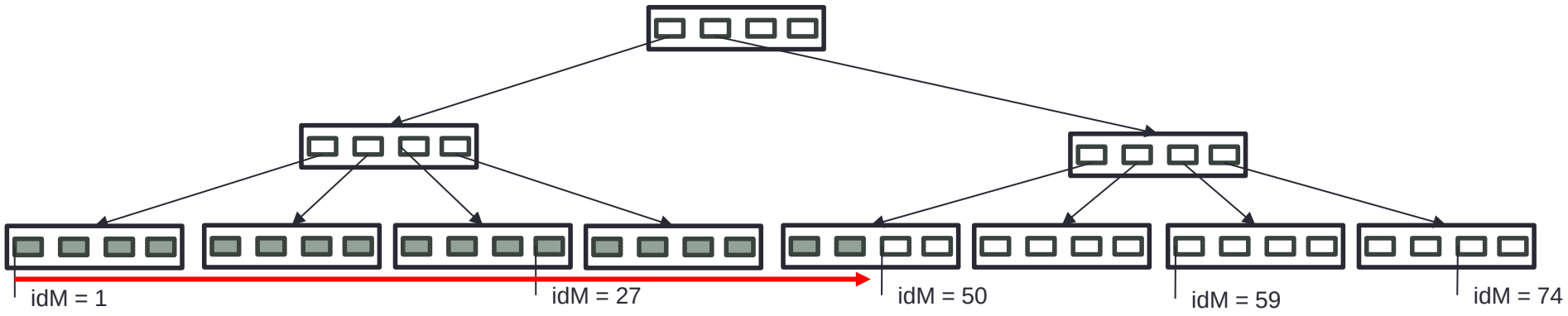
Árvores B+

- Filtro: idM >= 27
  - A busca leva ao primeiro nó folha que possua idM >=27.
  - Todas as entradas a partir desse ponto são recuperadas



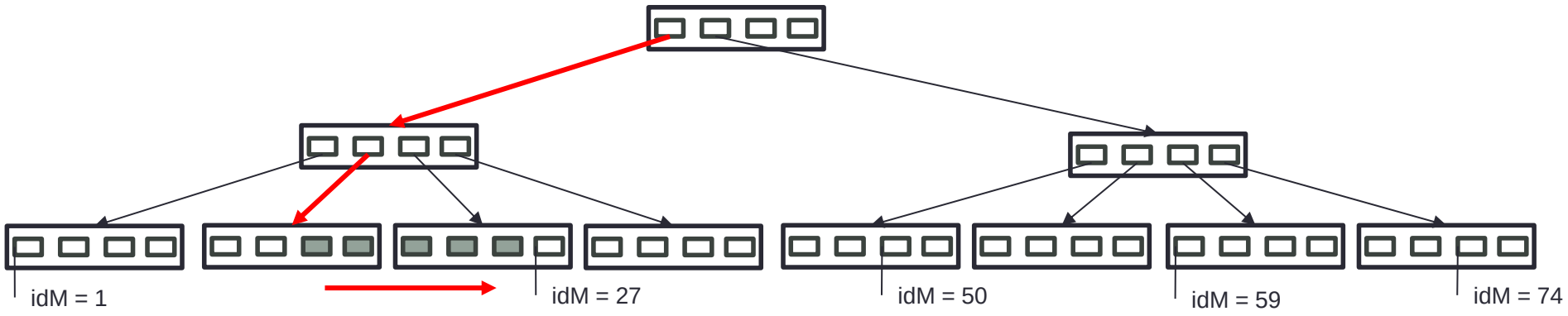
Árvores B+

- Filtro: idM < 50
  - A busca localiza o nó folha mais à esquerda
  - Os nós seguintes são lidos até que se encontre uma entrada com idM>=50



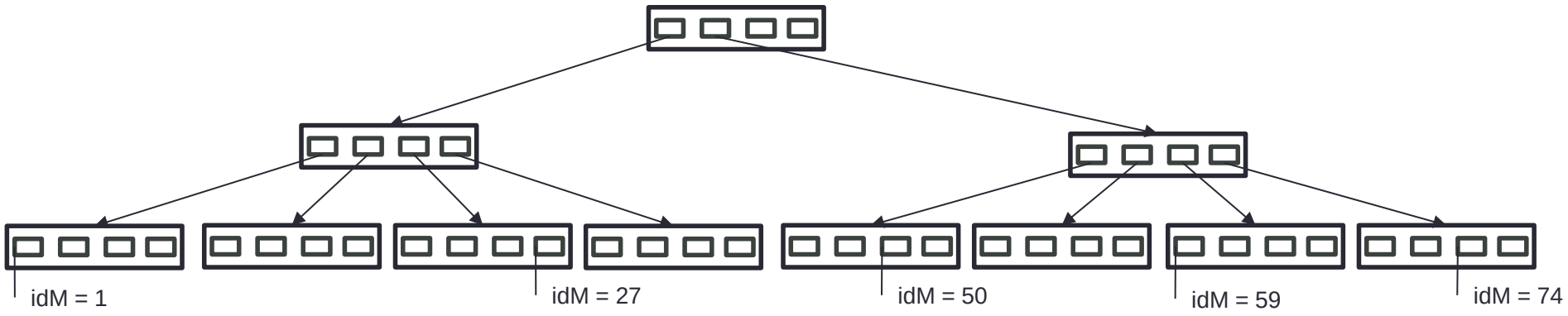
Árvores B+

- Filtro: idM >= 9 and idM < 27
  - A busca leva ao primeiro nó folha que possua idM >=9.
  - Os nós seguintes são lidos até que se encontre uma entrada com idM>=27



Árvores B+

- Como vimos, uma árvore B+ pode ser usada não apenas para organização física dos registros
  - Mas também como mecanismo eficiente de busca
  - Por esse motivo, as árvores B+ são as estruturas de dados preferidas para a criação de índices



Árvores B+

- O custo da busca é proporcional à altura da árvore
- Se o arquivo possuir K valores, a altura da árvore não ultrapassa $\lceil \log_{\lceil n/2 \rceil}(K) \rceil$.
- Supondo
  - Nó de 4kb
  - 40 bytes por entrada
  - n será igual a 100.
- Com 1 milhão de valores e n = 100
  - No máximo log50(1,000,000) = 4 nós são acessados em uma busca.
    - no máximo 4 acessos à páginas

Sumário

- Árvores B+
- Tipos de Árvores B+
- Índices no DBest

Árvores B+

- Dois tipos de índices baseados em árvores B+
  - Índice **clusterizado**
  - Índice **não clusterizado**

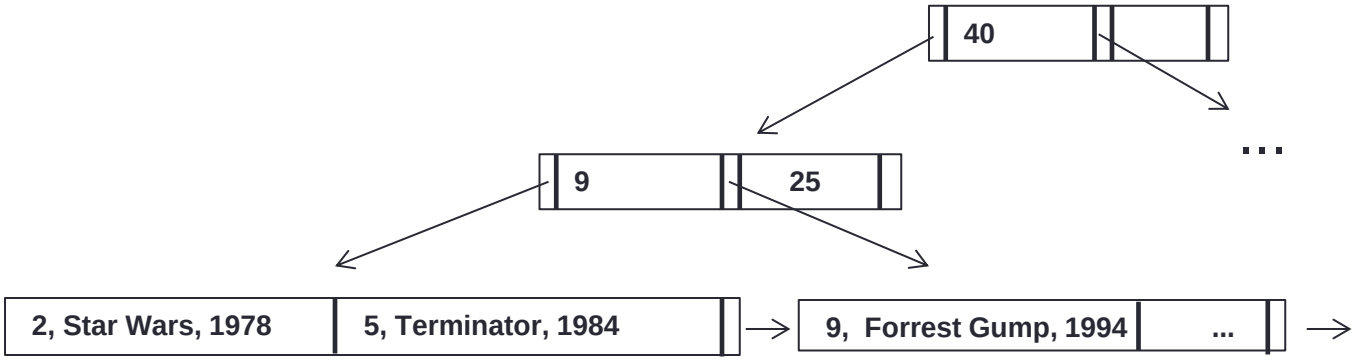
Árvores B+

- Índice **clusterizado**
  - é a própria árvore B+ usada para organizar os registros
    - O nível folha guarda todos os registros
    - Formato das entradas: <chave de busca, registro completo>

- Cada tabela só pode ter um índice clusterizado
  - Criado sobre sua chave primária

Árvores B+

- Páginas no nível folha de um índice clusterizado
  - Contém os registros
  - Organizadas por ordem de chave primária dos registros
  - Remanejadas conforma registros são inseridos/removidos



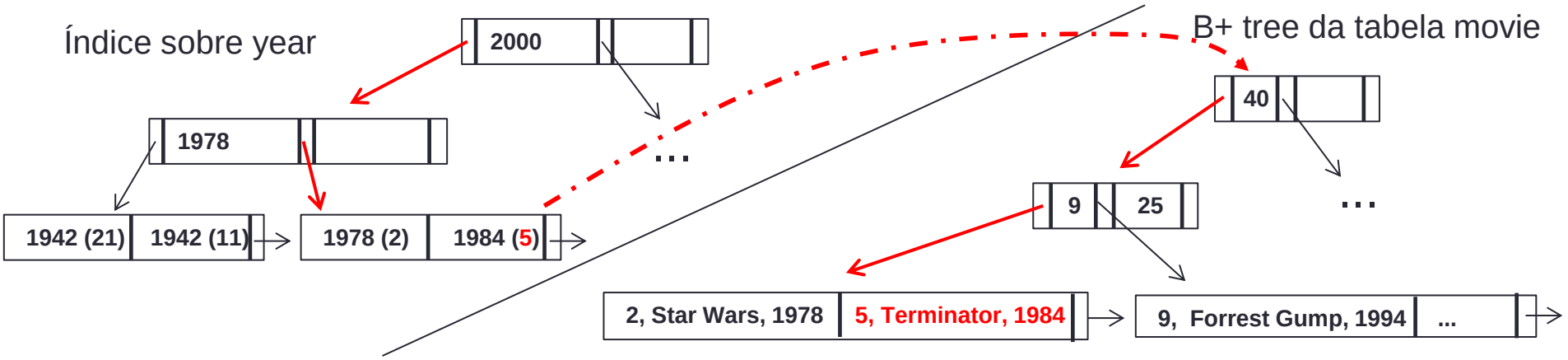
Árvores B+

- Índice **não clusterizado**
  - É uma árvore separada, usada apenas para facilitar o acesso
    - o nível folha guarda um ponteiro para o registro
  - O formato do ponteiro depende de como as tabelas são organizadas
    - Como uma B+tree: o ponteiro é a chave primária do registro
    - Como uma heap: o ponteiro é o RID do registro

RID (Record ID): local físico onde o registro foi armazenado

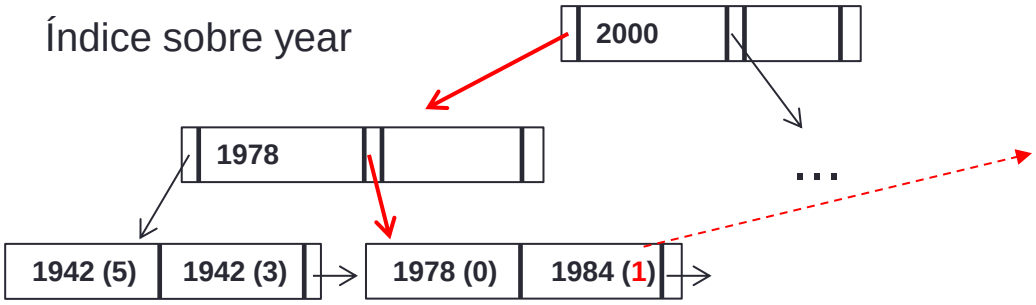
Árvores B+

- Tabela organizada como uma B+ tree
  - Nível folha de um índice secundário guarda a chave primária
  - Para acessar o registro, deve-se fazer uma busca na B+ tree que indexa essa chave
- Ex.
  - o filme com ano=1984 tem como ponteiro a chave primária 5



Árvores B+

- Tabela organizada como uma heap
  - Nível folha de um índice secundário guarda o RID do registro
  - Para recuperar o registro, deve-se acessar o arquivo na posição RID
- Ex.
  - o filme com ano=1984 tem como ponteiro o RID 1



heap da tabela movie

rid			
0	2	Star Wars	1978
1	5	Terminator	1984
2	9	Forrest Gump	1994
3	11	Bambi	1942
4	20	Joker	2019
5	21	Casablanca	1942

Sumário

- Árvores B+
- Tipos de Árvores B+
- Índices no DBest

Índices no DBest

- Nos próximos slides, veremos como o DBest organiza seus índices primários e secundários

Esquema usado como exemplo

Movie (idM, title, year)

Person (idP, p_name)

Movie_cast (idM, idP, c_name, ...)
  idM referencia movie
  idP referencia person

Índices no DBest

O índice primário de cada tabela é organizado por uma B+tree clusterizada.

Como o nível folha possui todos os registros, os dados da tabela são acessados por meio desse índice

Tabela movie

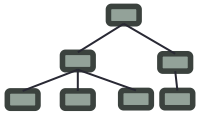


Tabela movie_cast

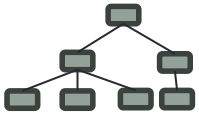
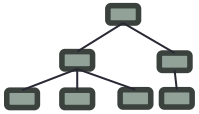


Tabela person



Esquema

Movie (idM, title, year)

Person (idP, p_name)

Movie_cast (idM, idP, c_name, ...)
  idM referencia movie
  idP referencia person

Estruturas de dados

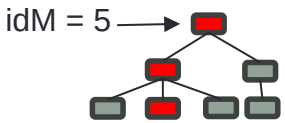
nome	Tipo	tipo	chave	valor
movie	tabela	B+tree clust.	idM	idM, title, year
movie_cast	tabela	B+tree clust.	idM, idP	idM, idP, c_name, ...
person	tabela	B+tree clust.	idP	idP, p_name
...	...	...	...	...

Índices no DBest

Consultas sobre a chave primária podem ser respondidas usando o índice primário

O nível folha dá acesso ao registro completo

Tabela movie



Esquema

Movie (idM, title, year)

Person (idP, p_name)

Movie_cast (idM, idP, c_name, ...)
  idM referencia movie
  idP referencia person

Estruturas de dados

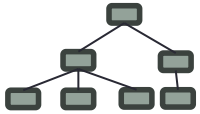
nome	Tipo	tipo	chave	valor
movie	tabela	B+tree clust.	idM	idM, title, year
movie_cast	tabela	B+tree clust.	idM, idP	idM, idP, c_name, ...
person	tabela	B+tree clust.	idP	idP, p_name
...	...	...	...	...

Índices no DBest

Índices secundários são B+trees não clusterizadas

O valor é a chave primária do registro referenciado

idx_year



Esquema

Movie (idM, title, year)

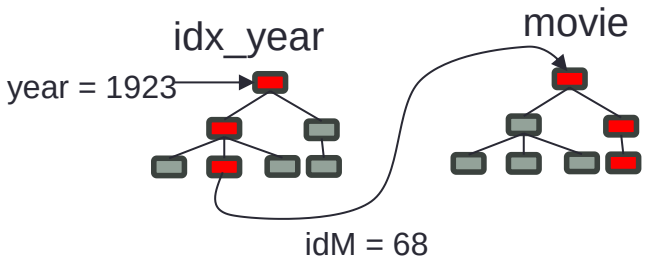
Estruturas de dados

nome	Tipo	tipo	chave	valor
movie	tabela	B+tree clust.	idM	idM, title, year
Idx_year	Índice	B+tree ñ clust.	year	idM
...	...	...	...	...

Índices no DBest

Uma busca sobre uma coluna indexada pode ser respondida usando o índice secundário

O índice primário correspondente deve ser acessado para recuperar o registro completo



Esquema

Movie (idM, title, year)

Estruturas de dados

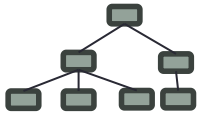
nome	Tipo	tipo	chave	valor
movie	tabela	B+tree clust.	idM	idM, title, year
Idx_year	Índice	B+tree ñ clust.	year	idM
...	...	...	...	...

Índices no DBest

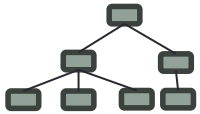
Chaves estrangeiras também são Índices secundários

O valor é a chave primária do registro referenciado

fk_mc_m



fk_mc_p



Esquema

Movie_cast (idM, idP, c_name, ...)
  idM referencia movie
  idP referencia person

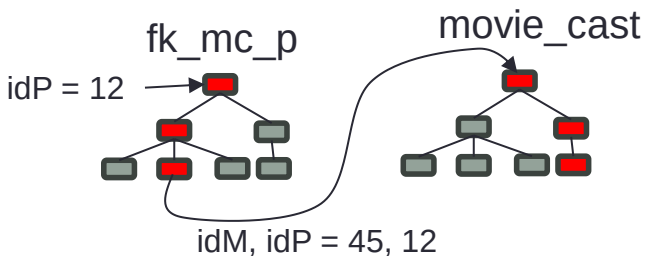
Estruturas de dados

nome	Tipo	tipo	chave	valor
movie_cast	tabela	B+tree clust.	idM, idP	idM, idP, c_name, ...
Fk_mc_p	Índice	B+tree ñ clust.	idP	idM, idP
Fk_mc_m	índice	B+tree ñ clust.	idM	idM, idP
...	...	...	...	...

Índices no DBest

Uma busca sobre uma chave estrangeira pode ser respondida usando esse índice secundário

O índice primário correspondente deve ser acessado para recuperar o registro completo



Esquema

Movie_cast (idM, idP, c_name, ...)
  idM referencia movie
  idP referencia person

Estruturas de dados

nome	Tipo	tipo	chave	valor
movie_cast	tabela	B+tree clust.	idM, idP	idM, idP, c_name, ...
Fk_mc_p	Índice	B+tree ñ clust.	idP	idM, idP
Fk_mc_m	índice	B+tree ñ clust.	idM	idM, idP
...	...	...	...	...

Encerramento

- Índices oferecem benefícios substanciais na procura por registros
  - **mas** as atualizações em índices impõem uma sobrecarga
  - quando o arquivo é modificado
    - os índices sobre o arquivo também precisam ser

- A varredura sequencial em um índice primário é eficiente
  - **mas** a varredura sequencial em um índice secundário é cara
  - Cada acesso a registro pode resultar na transferência de uma página do disco

Atividade Individual

- O pacote ibd.index.btree contém uma implementação de uma árvore B+ (classe BPlusTreeFile)
  - Boa parte das principais funções de busca estão implementadas

    - search(Key key)
      - Retorna o registro cuja chave de busca k seja igual a key (v = key)

    - searchLarger(Key lowerBound)
      - Retorna todos os registros cuja chave de busca k seja maior do que o lowerBound (k > lowerBound)

    - searchSmaller(Key upperBound)
      - Retorna todos os registros cuja chave de busca k seja menor do que o upperBound (k < upperBound)

Atividade Individual

- A função abaixo ainda não foi implementada
  - searchRange(Key lowerBound, Key upperBound)
    - Retorna todos os registros cuja chave de busca k seja
      - maior do que lowerBound (k> lowerBound) E
      - menor do que o upperBound (k< upperBound)

- O objetivo da atividade é implementar e testar essa função

- A função pode ser testada usando a classe ibd.index.btree.generic.Main

- Dica: analise as outras funções implementadas para estruturar a solução