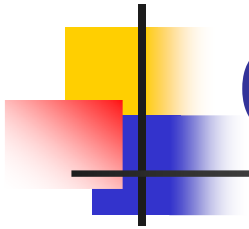
Busca indexada em arquivos: Complexidade e aplicações

Curso: Sistemas de Informação

USP LESTE

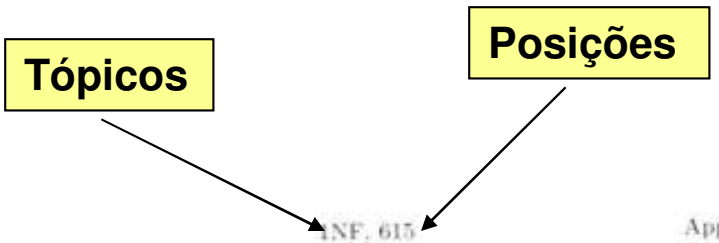
Prof. José de Jesús Pérez Alcázar



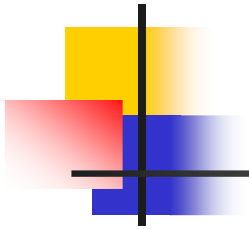
Objetivo

- Apresentar as técnicas e conceitos de indexação, os quais, têm ampla aplicação no desenvolvimento eficiente de sistemas de bancos de dados ou sistemas de arquivos.

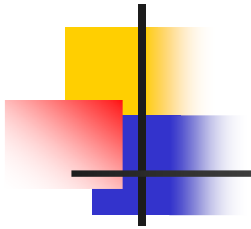
O que é um índice?



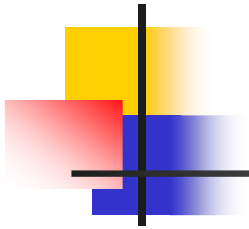
SUBJECT INDEX



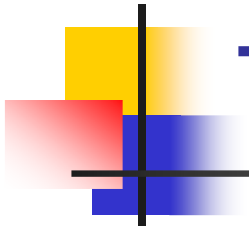
- Livros: índice forma de achar tópicos rapidamente.
- Mecanismos de indexação usados para melhorar o acesso aos dados desejados.
- Índice: Estrutura auxiliar que torna mais eficiente a pesquisa baseada em alguns campos → campos de indexação



- A indexação independe da organização dos registros no arquivo de dados
- São armazenados em arquivos auxiliares chamados **arquivos de índice**
- Os arquivos de índice:
  - Formados por registros (entradas do índice) → campo indexação + apontador (valor, posição)
  - Menores que o arquivo original ⇨

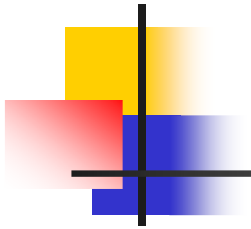


- Classificação de arquivos índice:
  - Baseados em arquivos ordenados (índices de um só nível)
  - Baseados em vários níveis (estruturas de dados de árvore)
  - Outras estruturas.



Tipos de índices ordenados

- Índice Primário → quando o campo de indexação é uma chave e o arquivo de dados é ordenado por esse campo.
- Índice de agrupamento (“cluster”) → quando o campo de indexação não é chave mas o arquivo de dados é ordenado por esse campo.
- Índice secundário → campo de indexação não é o de ordenação.

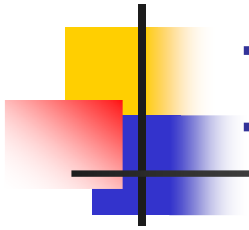


- Quantos índices primários podem existir por arquivo?
- Quantos índices de “cluster” podem existir por arquivo?
- Quantos índices secundários podem existir por arquivo?

Índices primários

- Registros são da forma $< K_i, P_i >$, onde K_i é o campo de indexação e P_i é o endereço a um bloco de dados.
- Arquivo de registros de tamanho fixo ordena
- Existe ro de índice para cada bloco ; K_i é a chave do registro âncora (índice esparso)

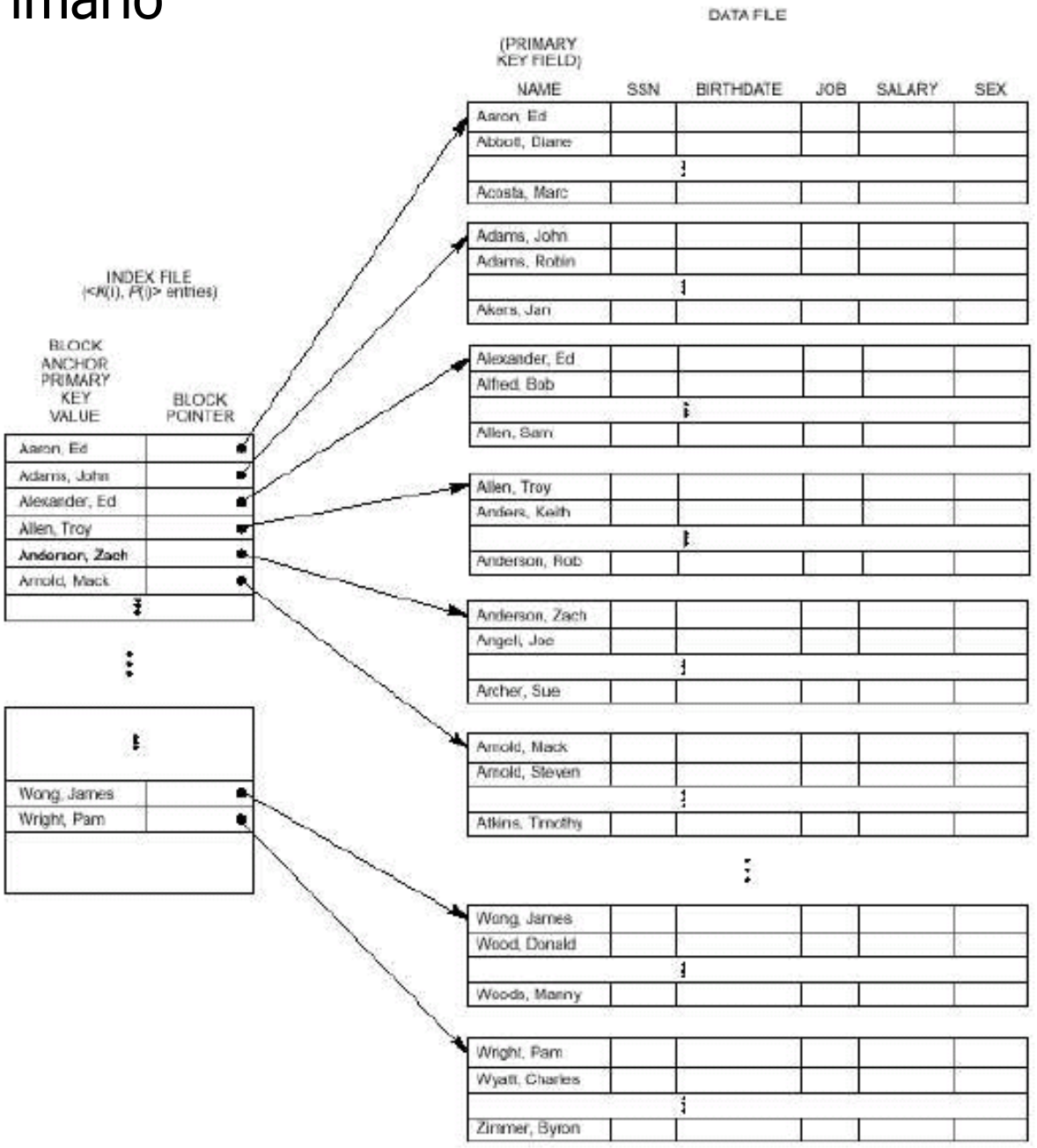
Primeiro registro de cada bloco → registro âncora ou âncora do bloco



Índices primários

- Registros menores e menos registros que o arquivo de dados. Usa menos blocos. Boa parte pode estar em memória principal. Veja Fig.
- Operação de busca envolve:
  - Uma busca no índice (pode ser binária)
  - Carga do bloco do arquivo de dados
  - Busca no bloco (pode ser binária)

Índice primário



Índices primários – Exemplo 1

- Parâmetros gerais:
  - Tamanho do bloco 1024 bytes
  - Arquivos de tamanho fixo
  - Alocação não espalhada
- Parâmetros do arquivos de dados:
  - Número de registros: r = 30.000 registros
  - Tamanho de registro: R = 100 bytes
  - Fator de bloco: bfr = $\lfloor B/R \rfloor$ = 10 registros por bloco
  - Número de blocos: b = $\lceil (r/bfr) \rceil$ = 3000 blocos

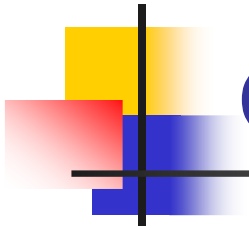
Índices primários – Exemplo 1

- Parâmetros do arquivo de índices
  - Chave primária tem 9 bytes
  - Apontador tem 6 bytes
  - Tamanho do registro R_i = 15 bytes
  - Fator de bloco $bfr_i = \lfloor B / R_i \rfloor$ = 68 registros por bloco
  - Número de registros r_i = 3000
  - Número de blocos $b_i = \lceil (r_i / bfr_i) \rceil$ = 45 blocos

Índices primários – Exemplo 1

- Custo da busca:
  - Sem índice: uma busca binária pode ser feita com $\lceil \log_2 b \rceil = \lceil \log_2 3000 \rceil = 12$ acessos a blocos
  - Com índice: uma busca binária pode ser feita com $\lceil \log_2 b_i \rceil = \lceil \log_2 45 \rceil = 6$ acessos a blocos de índice + 1 no arquivo de dados

Índices primários - comentários

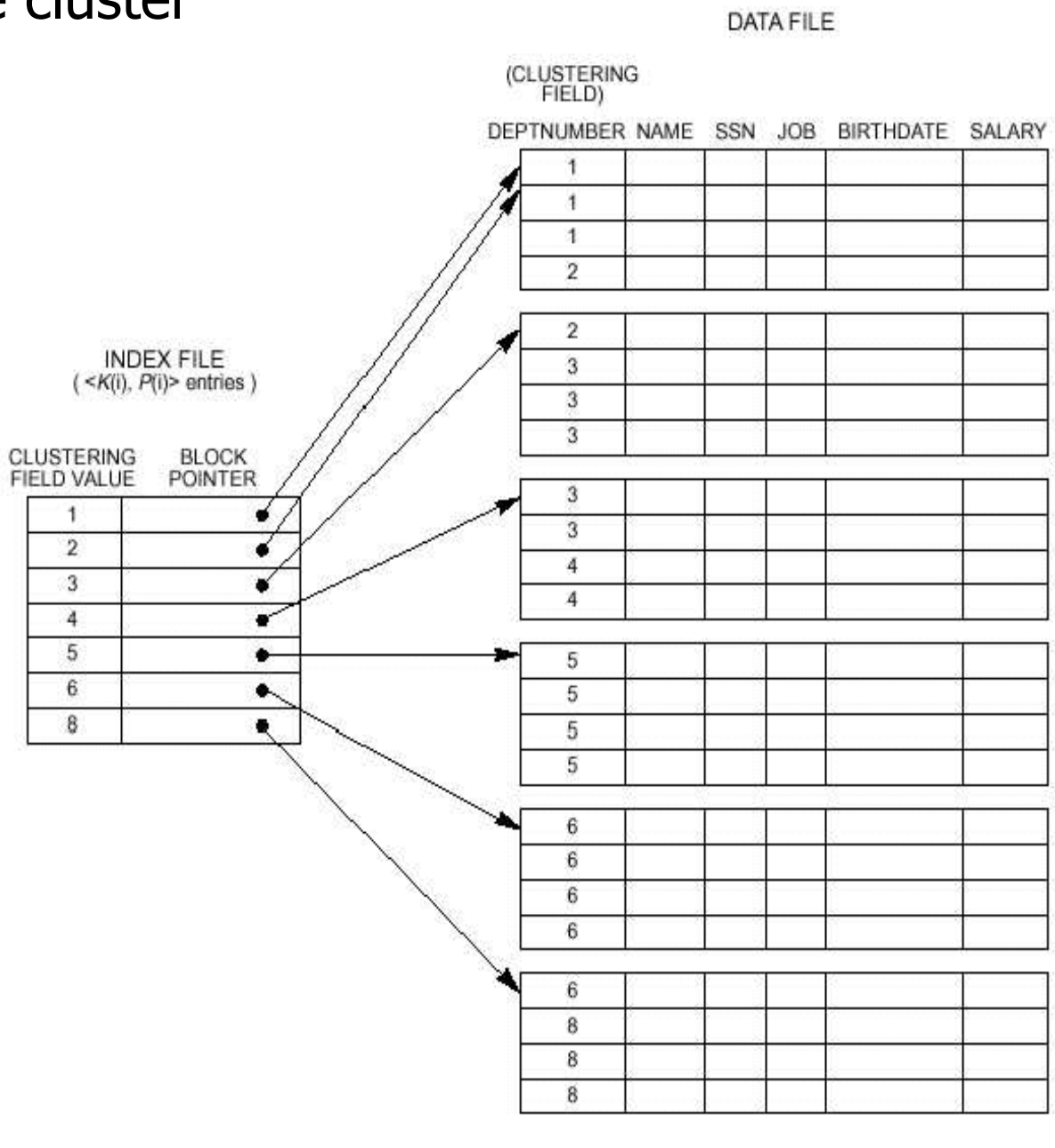


- Maior problema (similar arquivos ordenados) é a inserção e remoção de registros.
- Problema composto com índices primários → inserir um registro na posição certa pode mudar também registros âncoras dos blocos.
- Remoção de registros tratados usando marcadores

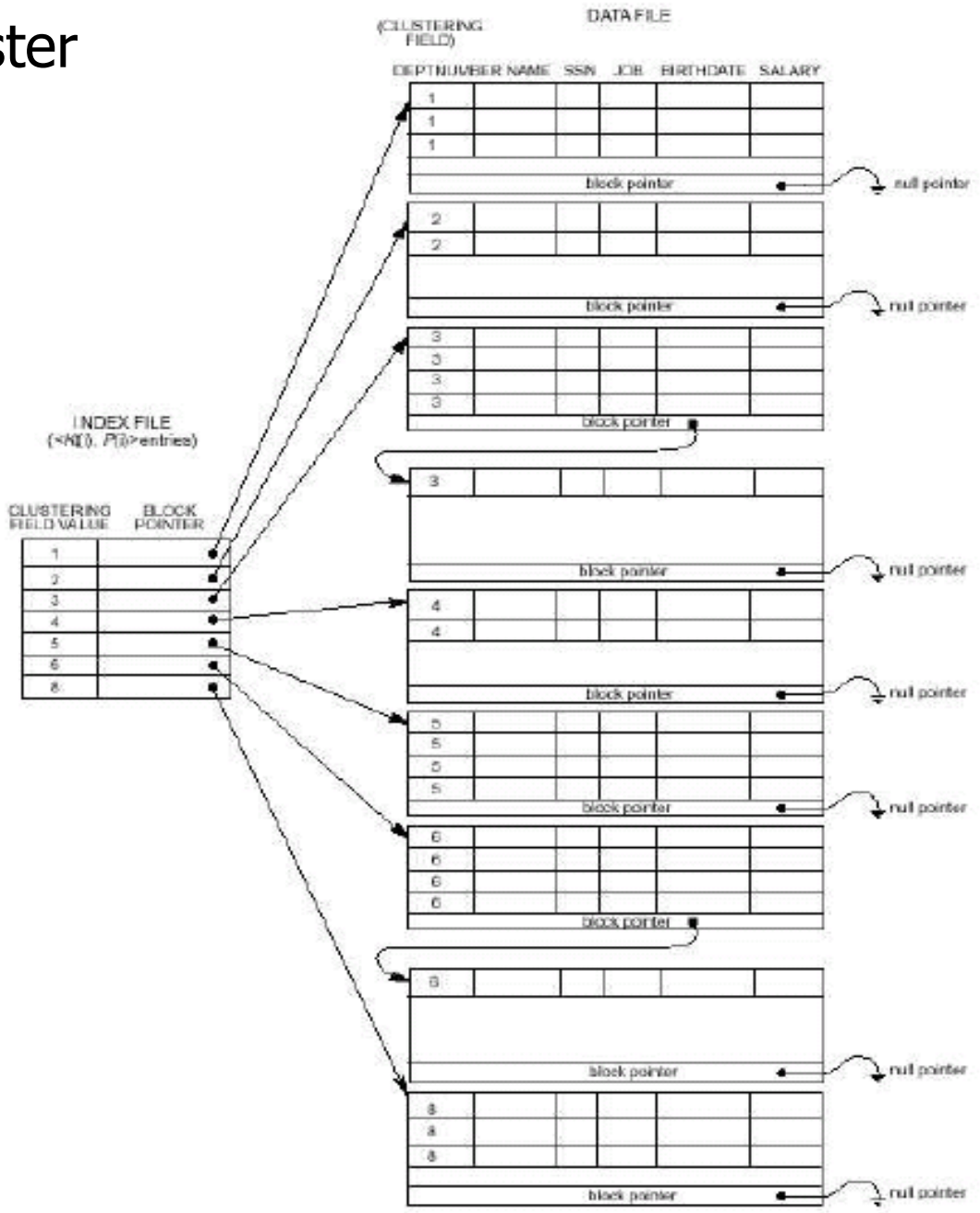
Índice de "Cluster"

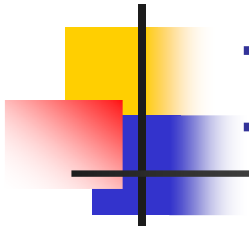
- Campo de cluster: ordena o arquivo, mas podem haver dois registros diferentes com o mesmo valor.
- Existe um registro de índice para cada valor diferente do campo de cluster no arquivo de dados (Índice esparso) 
- Para facilitar remoções e inserções pode ser usado um bloco para cada valor diferente do campo de cluster → blocos adicionais são encadeados se for necessário 

Índice de cluster



Índice de cluster

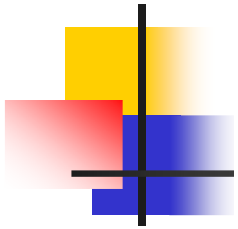


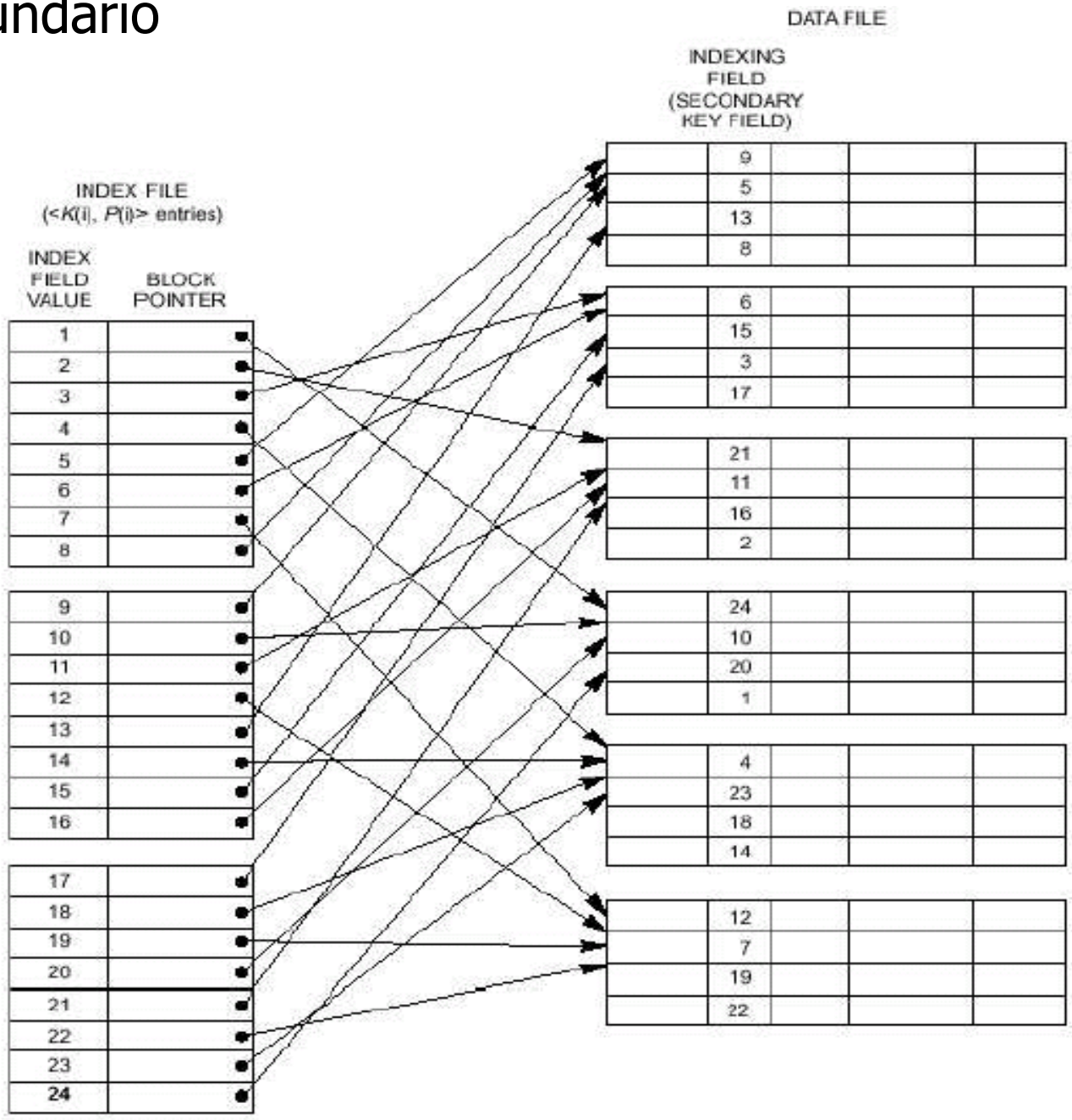


Índice secundário

- Campo de indexação:
  - Não ordena o arquivo de dados
  - Pode ou não ser uma chave no arquivo de dados
- No caso de ser chave:
  - Existe uma entrada de índice para cada registro no arquivo de dados (índice denso)
  - Provê maior ganho com relação a uma busca sem índice que o índice primário → busca seqüencial no arquivo de dados
  - Precisa de maior espaço de armazenamento e o tempo de busca é maior que o índice primário

Índice secundário





Índice secundário – Exemplo 2

r = 30000 registros

R = 100 bytes

bfr = 10 regs.x bloco

b = 3000 blocos

- Parâmetros gerais e parâm[illegible]vo de dados: os mesmos do e[illegible]r
- Parâmetros do arquivo de índices
  - Chave primária tem 9 bytes
  - Apontador tem 6 bytes
  - Tamanho do registro R_i = 15 bytes
  - Fator de bloco $bfr_i = \lfloor B / R_i \rfloor$ = 68 registros por bloco
  - Número de registros r_i = 30000
  - Número de blocos $b_i = \lceil (r_i / bfr_i) \rceil$ = 442 blocos

Índice secundário – Exemplo 2

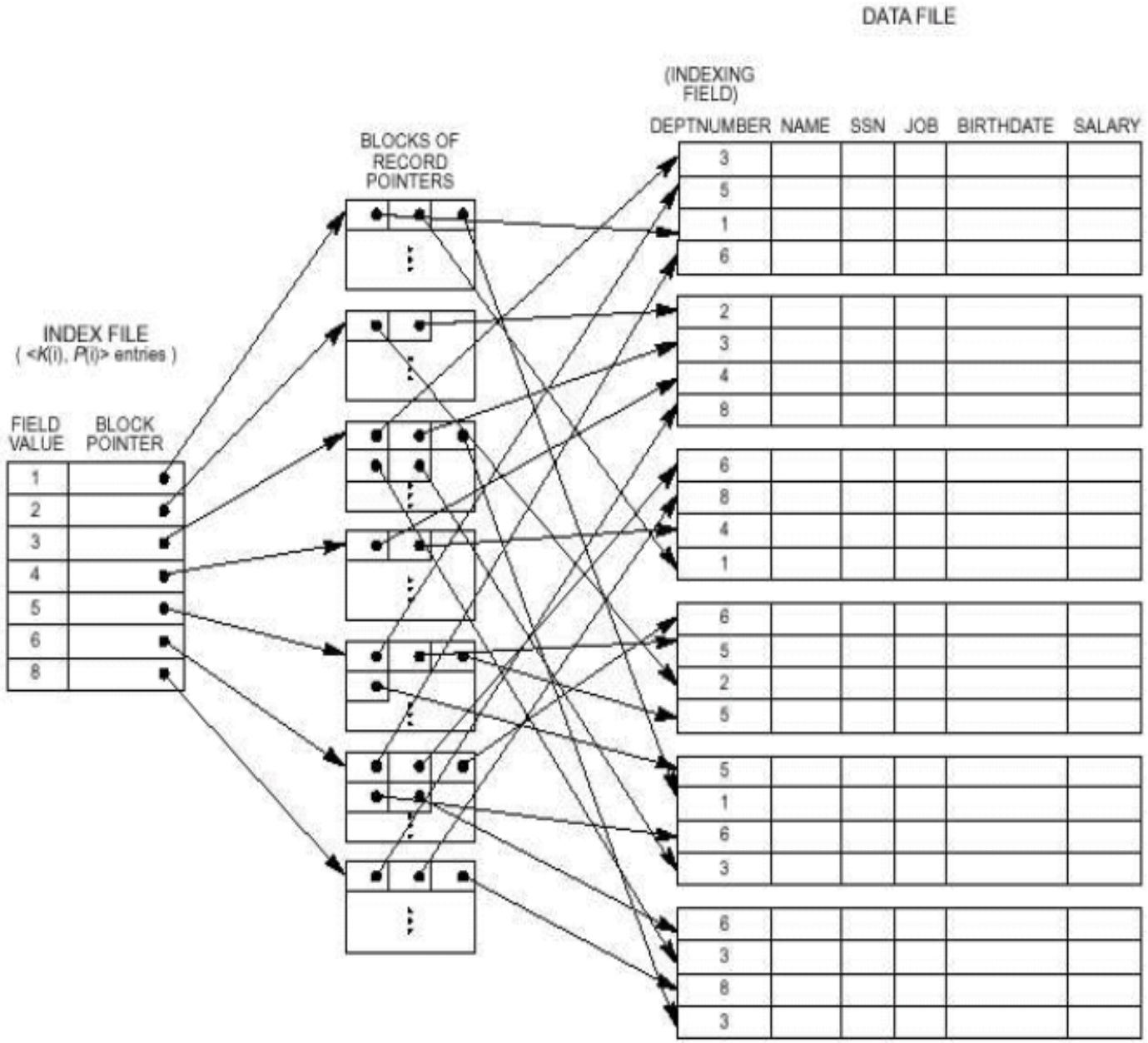
- Custo da busca:
  - Sem índice: Uma busca seqüencial pode ser feita em média com $\lceil b/2 \rceil$ = 1500 acessos a blocos
  - Com índice: Uma busca binária pode ser feita com $\lceil \log_2 b_i \rceil$ = 9 acessos a blocos de índice + 1 no arquivo de dados

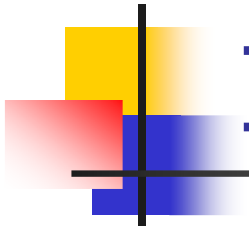
Índice secundário

- No caso de ser um campo não chave:
  - Opção 1: uma entrada de índice para cada registro do arquivo de dados
  - Opção 2: Vários apontadores em cada entrada do índice; uma para cada posição do arq[illegible] dados onde o valor do campo de indexação ocorre
  - Opção 3: Cada entrada do índice aponta para um bloco de registros que contém apontadores a blocos do arquivo de dados (Veja Fig.)
- Provê uma ordenação lógica dos registros pelo campo de indexação.

$[K_i,<P_{i1},..,P_{in}>]$

Índice secundário





Índices Multinível

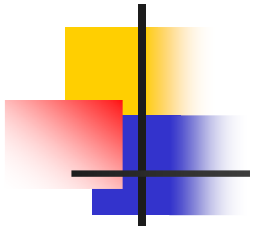
- Motivação → Reduzir o número de acesso a blocos do índice de $\log_2 b_i$ para $\log_{fo} b_i$
- Fan-out: fator de bloco do arquivo de índice; fo > 2

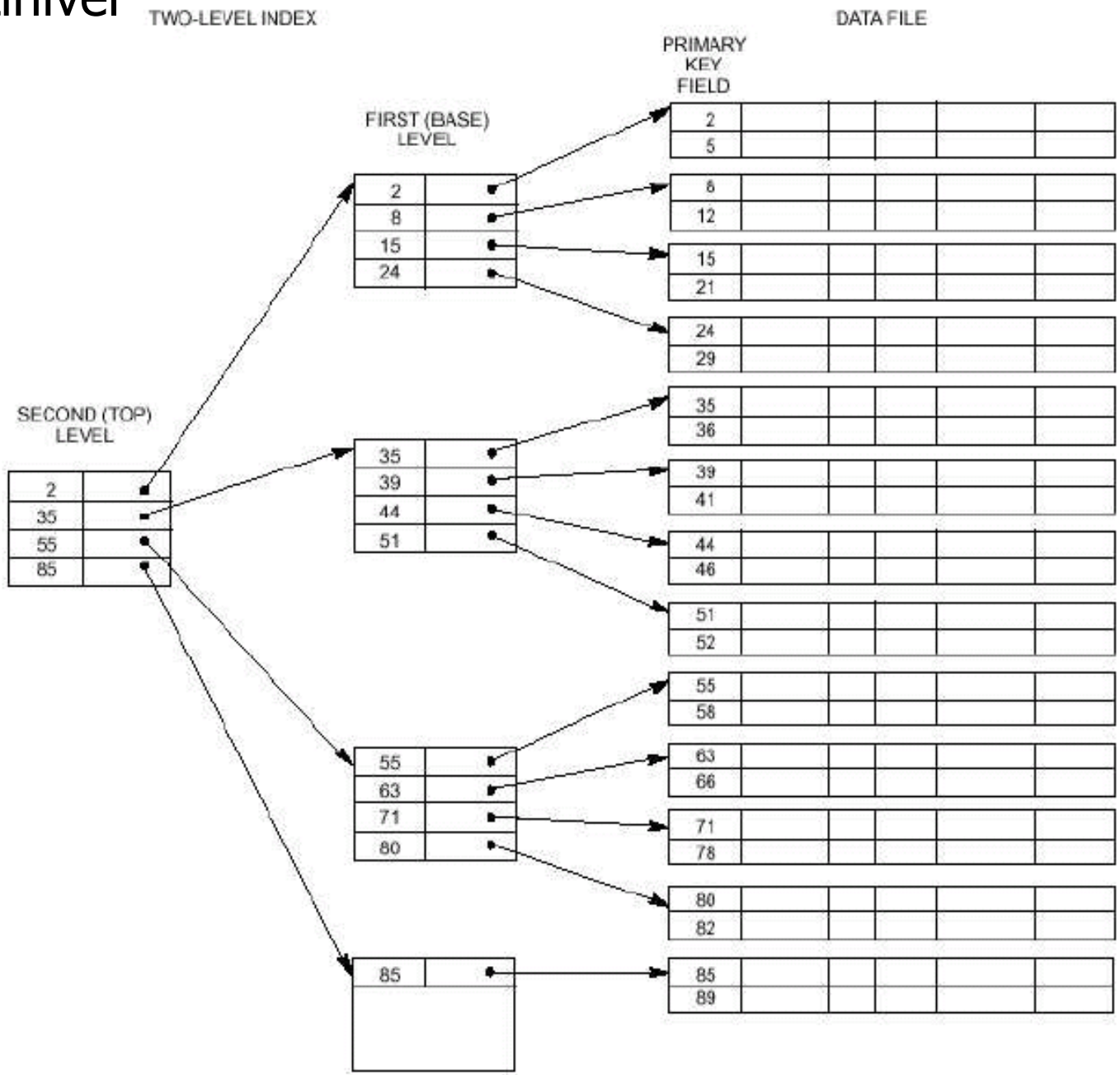
Nível	Nr. de Registros	Fan-Out	Nr. de Blocos
1	r_1	fo	$\lceil r_1/fo \rceil$
2	$r_2 = \lceil r_1/fo \rceil$	fo	$\lceil r_1/fo^2 \rceil$
3	$r_3 = \lceil r_1/fo^2 \rceil$	fo	$\lceil r_1/fo^3 \rceil$
$\vdots$	$\vdots$	$\vdots$	$\vdots$
t	$r_t = \lceil r_1/fo^{t-1} \rceil$	fo	$\lceil r_1/fo^t \rceil$

$\therefore$ Se $1 \leq r_1/(f_0)^t$, t = $\left| \log_{fo}(r_1) \right|$

Pode ser construido sobre qualquer tipo: primário, secundário ou de cluster (Veja Fig.)

Índice multinível





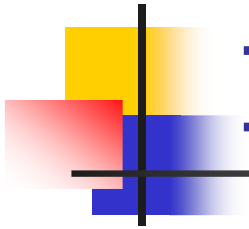
Índices multinível – Exemplo 3

B = 1024 bytes

R_i = 15 bytes

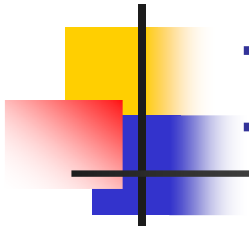
b_i = 442 blocos

- Considerando o ex
  - Fan-out (entradas de índices por bloco) = bfr = B/R_i = 68
  - Estrutura dos índices:
    - Nível 1: b_1 = 442 blocos
    - Nível 2: $b_2 = \lceil (b_1 / fo) \rceil$ = 7 blocos
    - Nível 3: $b_3 = \lceil (b_2 / fo) \rceil$ = 1 bloco
  - Número de níveis t = $\lceil \log_{68} 30000 \rceil$ = 3
  - Número de acessos t + 1 = 4



Índices multinível

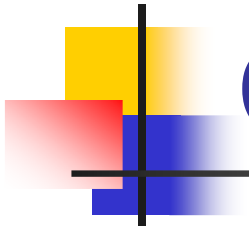
- Se o índice for esparso é necessário acessar o bloco do arquivo de dados para determinar a existência de um registro.
- Este tipo de índices foi muito usado nos primeiros sistemas IBM → ISAM



Índices multinível

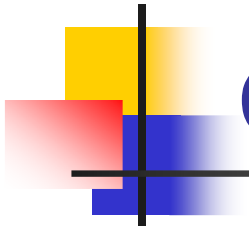
- Algoritmo (Busca em um índice não denso com t níveis)
- p ← endereço do nível topo do índice;
- para j ← t passo – 1 até 1 faça
- inicio
- leia o bloco índice (no nível j) cujo endereço é p;
- busca no bloco p pela entrada i tal que Kj(i) ≤ K < Kj (i+1) ;
- (Se Kj(i) é a última entrada no bloco então a condição é Kj(i) ≤ K)
- p ← Pj(i);
- fim
- leia o bloco do arquivo de dados cujo endereço é p;
- busca no bloco p pelo registro com chave K;

Índices multinível - Comentários



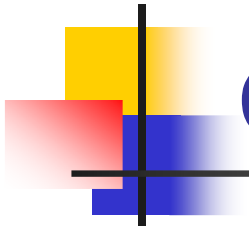
- Os índices multinível reduzem o número de blocos acessados na busca de um registro.
- Segue o problema das inserções e remoções (ordenamento físico dos índices)
- Solução : Deixar espaço livre para inserções com expansão dinâmica dos arquivos. Índice dinâmico multinível → árvores B e B^+
- Base dos SGBDs.

Comentários sobre atualização de índices - Remoção



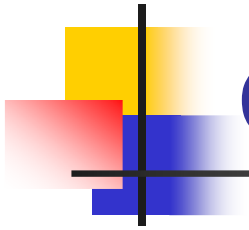
- Se o registro for o único registro no arquivo com um valor específico do campo de indexação, o valor deve ser removido do índice.
- Remoção de índices de um só nível
  - Densos: remoção do registro índice é similar a do registro do arquivo
  - Esparsos: Procura-se valor seguinte do campo índice no arquivo e substitua entrada no índice. Caso contrário ela é removida. ⇦

Comentários sobre atualização de índices - Inserção

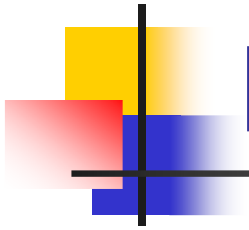


- Índices de um só nível:
  - Busca o lugar de inserção
  - Índices densos: senão aparece no índice insira-o.
  - Índices esparso: índice tem uma entrada para cada bloco, precisa ser mudado só no caso da criação de um novo bloco.
- Multinível: extensões dos algoritmos de um só nível.

Comentários sobre atualização de índices

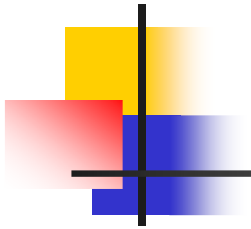


- Em geral, quando um arquivo é modificado, todo índice do arquivo deve ser atualizado. Isto impõe um "overhead" na atualização de arquivos.



Bibliografia

- Folk, M.; Zoellick, B. “File Structures” 2nd edition. Addison-Wesley Pubs. 1992.
- Elmasri, R.; Navathe, S. “Fundamentals of Database Systems” third edition. Addison-Wesley Pubs. 2001.
- Silberschatz, A.; Korth, H.F.; Sudarshan, S. “Database system concepts” 4th edition. McGraw-Hill, 2001.



- Índices podem ser esparsos ou densos.
- Densos: uma entrada para cada valor do campo de indexação
- Dispersos ou esparsos: uma entrada para alguns dos valores.