ALGORITMOS E ESTRUTURAS DE DADOS II

Índices (resumo+árvores B+lista invertidas)

Karina Valdivia Delgado

Índices

- O arquivo de índice é um arquivo ordenado que possui índices que permitem o acesso aleatório a um registro dado um valor de um campo.

Índices

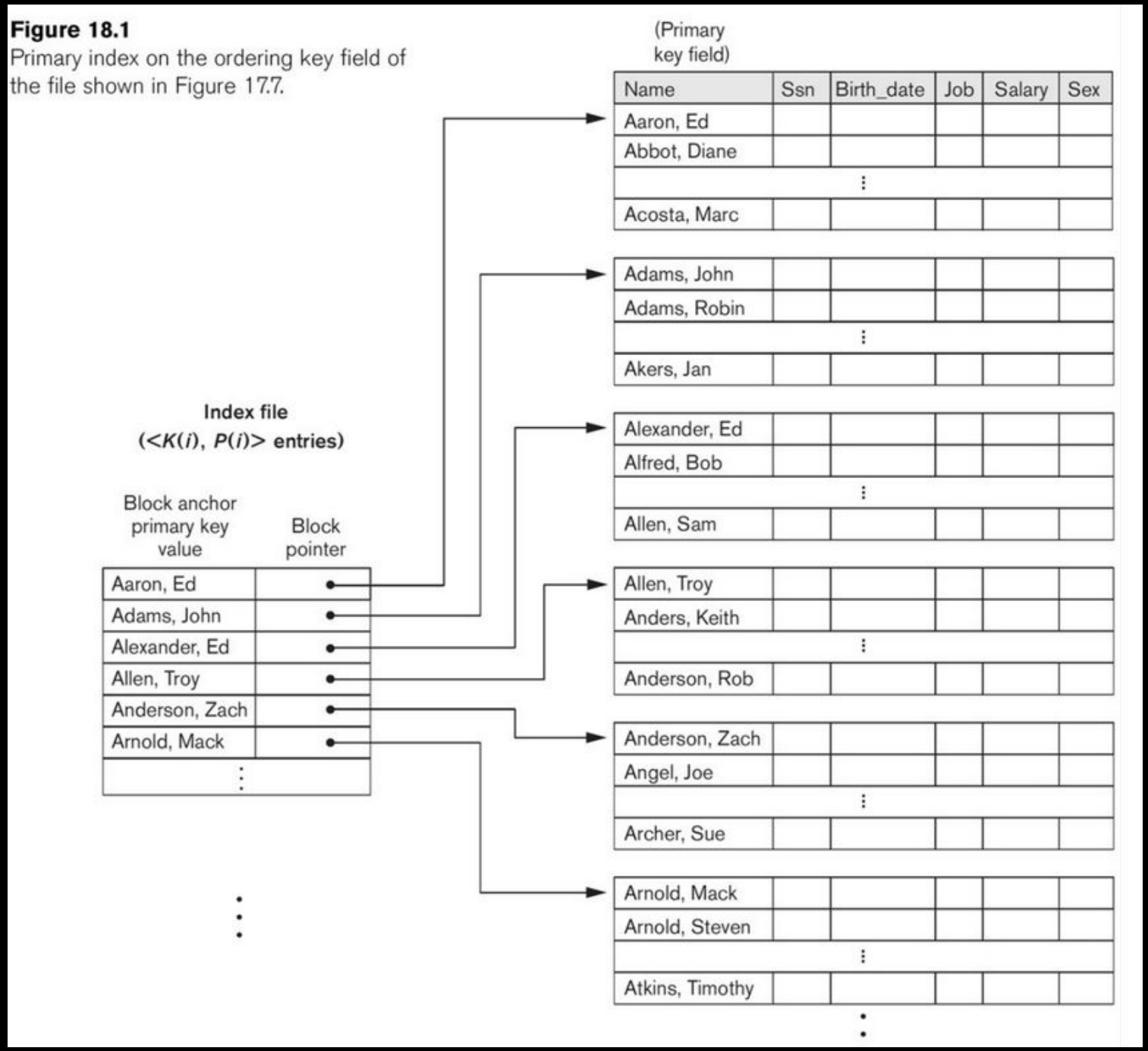
- O arquivo de índice segue a mesma ordem do arquivo de dados
  - **Primário:** entrada <k,b>, em que k:chave do primeiro registro do bloco e b: bloco
  - **De clustering:** entrada <c,b>, em que c:campo de classificação física, vários registros tem o mesmo valor e b:bloco
- O arquivo de índice não segue a mesma ordem do arquivo de dados
  - **Secundário:** <i,w>, em que i:campo de indexação que não ordena fisicamente e w: bloco ou registro

Índices

- **Denso:** possuem uma entrada para cada registro no conjunto de dados
- **Esparso:** possuem entradas para apenas alguns registros

Índice primário esparso

- entradas <k,b>
  k:chave do primeiro registro do bloco
  b: bloco
- Busca O(lg bi)
  bi: número de blocos de índice
  B: número de blocos de dados



Índice de clustering esparso

- entrada <c,b>
  c:campo de agrupamento físico, vários registros tem o mesmo valor
  b:bloco
- Busca O(lg bi)
  bi: número de blocos de índice
  B: número de blocos de dados

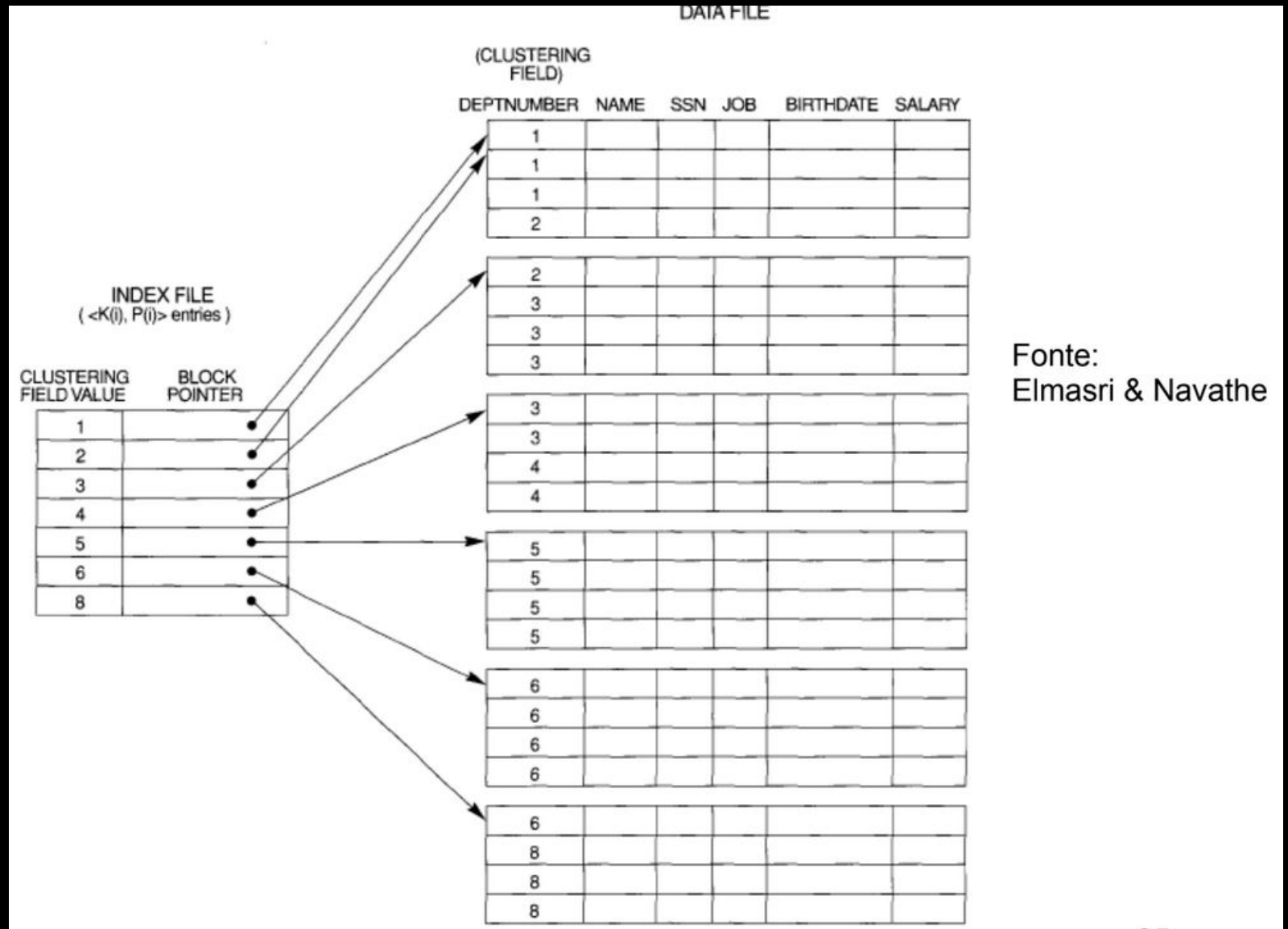


FIGURE 14.2 A clustering index on the DEPTNUMBER ordering nonkey field of an EMPLOYEE file.

Fonte:
Elmasri & Navathe

Índice secundário denso

- entrada <i,w>
  i:campo de indexação que não ordena fisicamente
  w: bloco ou registro
- Busca O(lg bi)
  bi: número de blocos de índice
  B: número de blocos de dados

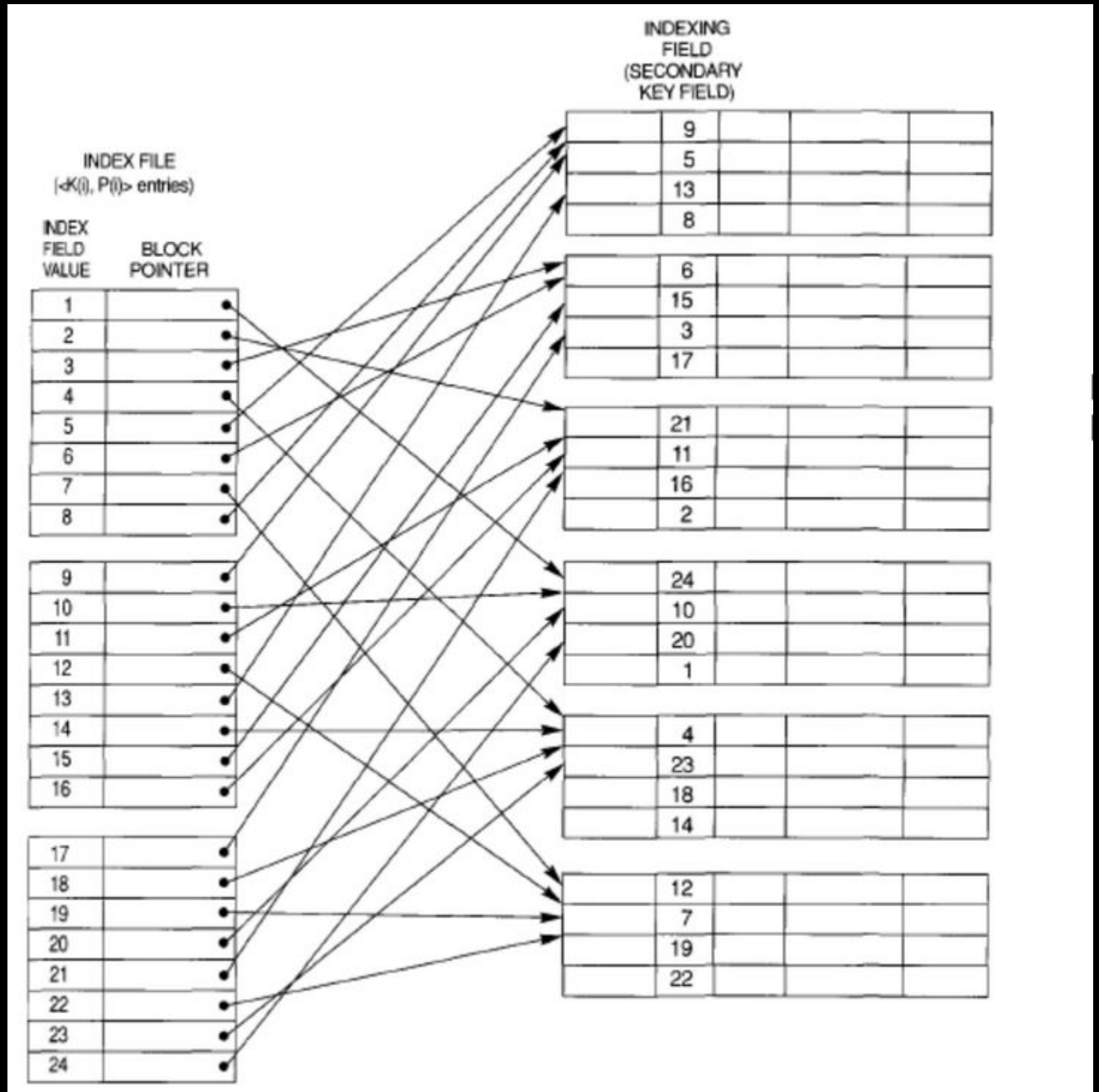


FIGURE 14.4 A dense secondary index (with block pointers) on a nonordering key field of a file.

Índices

- Inserção/remoção complicada O(bi+B)

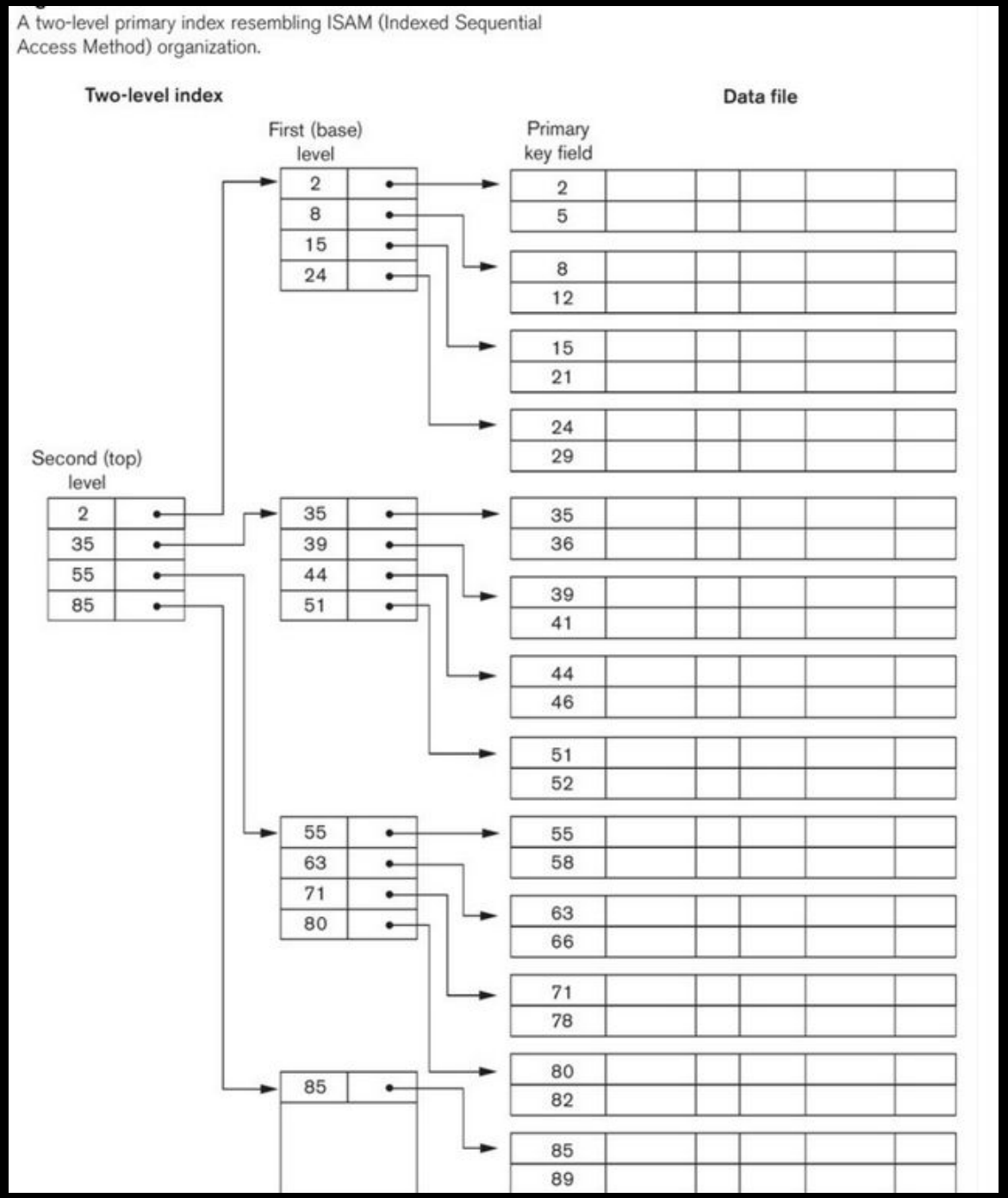
Índices

- O que fazer se o arquivo é muito grande e o próprio índice ficou grande (com muitos blocos)

  Índice do índice

Índice primário esparso multinível

- entradas <k,b>
  k:chave do primeiro registro do bloco
  b: bloco
- Busca O(t)
  t = ceil($\log_{fbi} r^1$) altura da árvore
  fbi = nr de registros que cabem em um bloco de índice (fator de blocagem dos blocos de índice)
  r^1 = nro total de registros de índice no nível 1
- Inserção/remoção cada vez mais complicada!



Como manter uma busca eficiente mas permitir uma inserção e remoção razoável?

Como manter uma busca eficiente mas permitir uma inserção e remoção razoável?

Está acontecendo algo similar ao que acontecia em memória principal:

- **Considerando um vetor ordenado, podemos usar busca binária, mas o problema era justamente na inserção/remoção**
- **Qual era a alternativa para manter uma busca eficiente e permitir ainda uma inserção e remoção razoável?**

Como manter uma busca eficiente mas permitir uma inserção e remoção razoável?

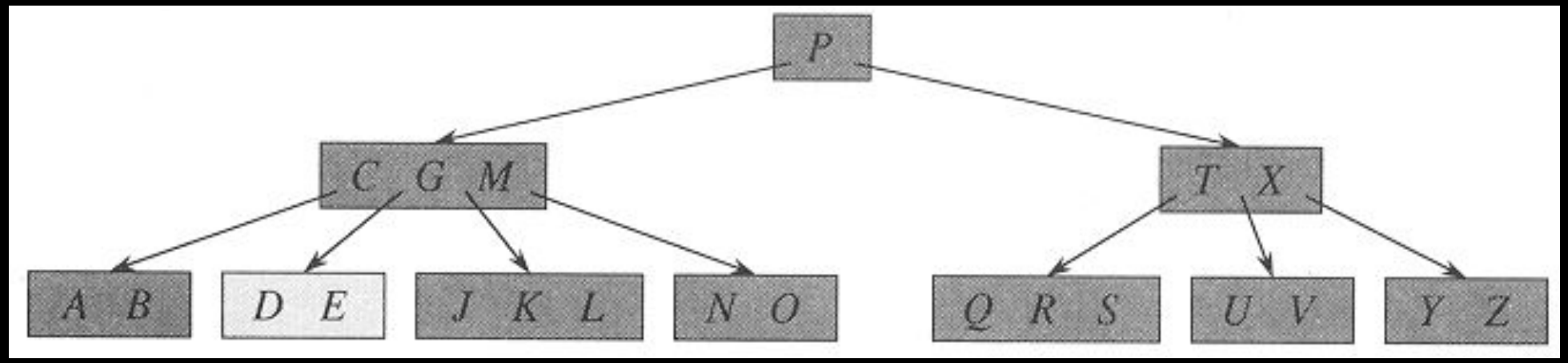
Está acontecendo algo similar ao que acontecia em memória principal:

- **Considerando um vetor ordenado, podemos usar busca binária, mas o problema era justamente na inserção/remoção**
- **Qual era a alternativa para manter uma busca eficiente e permitir ainda uma inserção e remoção razoável?**
    - **Árvores binárias de busca balanceadas**

Como manter uma busca eficiente mas permitir uma inserção e remoção razoável?

Como manter uma busca eficiente mas permitir uma inserção e remoção razoável?

Usar árvore B como índice



Índices e árvores B

A árvore B pode ser utilizada como arquivo de índices

- Cada nó da árvore tem as chaves e ponteiros para um outro arquivo, o arquivo de registros de dados.
- Vantagem: dados são independentes do arquivo de índices.

Índices e árvores B

- A aplicação de árvores B como índice é imediata, não preciso fazer mudanças nos algoritmos.
- A única mudança é que ao invés de ponteiros para áreas de memória, utiliza-se campos do tipo inteiro com endereços relativos ao início do arquivo dos registros

Árvore B, hash e B+ funcionam B para índices primários

Arquivo invertido funciona bem para índice secundário, bem como a árvore K-D

Curiosidades: Ger. Arq. X PostGreSQL

- **Índice padrão: Árvore-B**

```
CREATE INDEX name ON table USING column;
```

- **Índice alternativo: Hash**

```
CREATE INDEX name ON table USING hash(column);
```

- **Índice flexível: Árvore-B+ com extensões para conjuntos: GiST (*Generalized Search Tree*)**

```
CREATE INDEX name ON table USING gist(column);
```

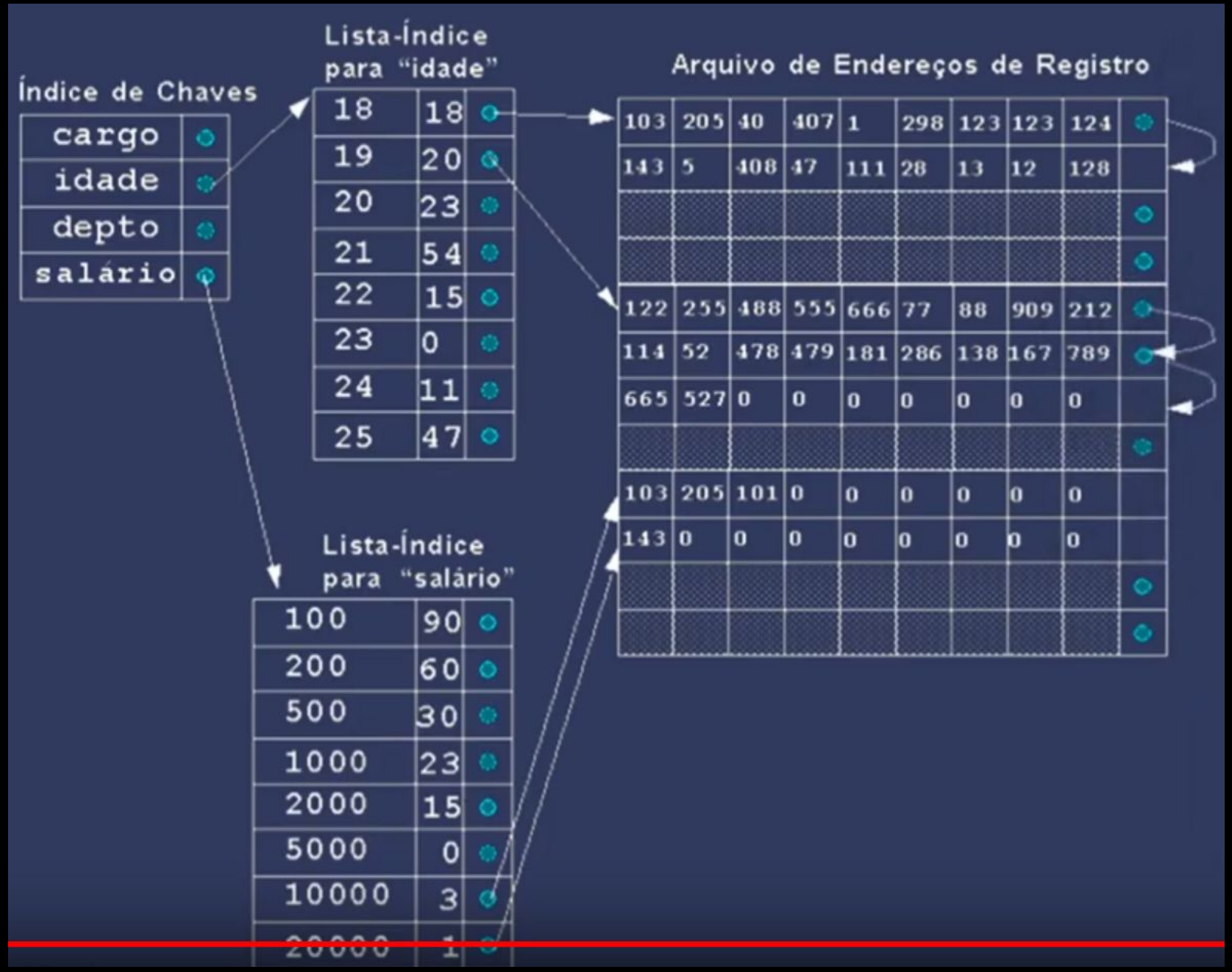
- **Índice para chave secundária: Arquivo Invertido: GIN (*Generalized Inverted Index*)**

```
CREATE INDEX name ON table USING gin(column);
```

Fonte:https://www.youtube.com/watch?v=iNG4l5mWzgU&list=PLmDIGfkfgKy1pouEOuiW1G681p7u_20O8&index=28

Lista invertida

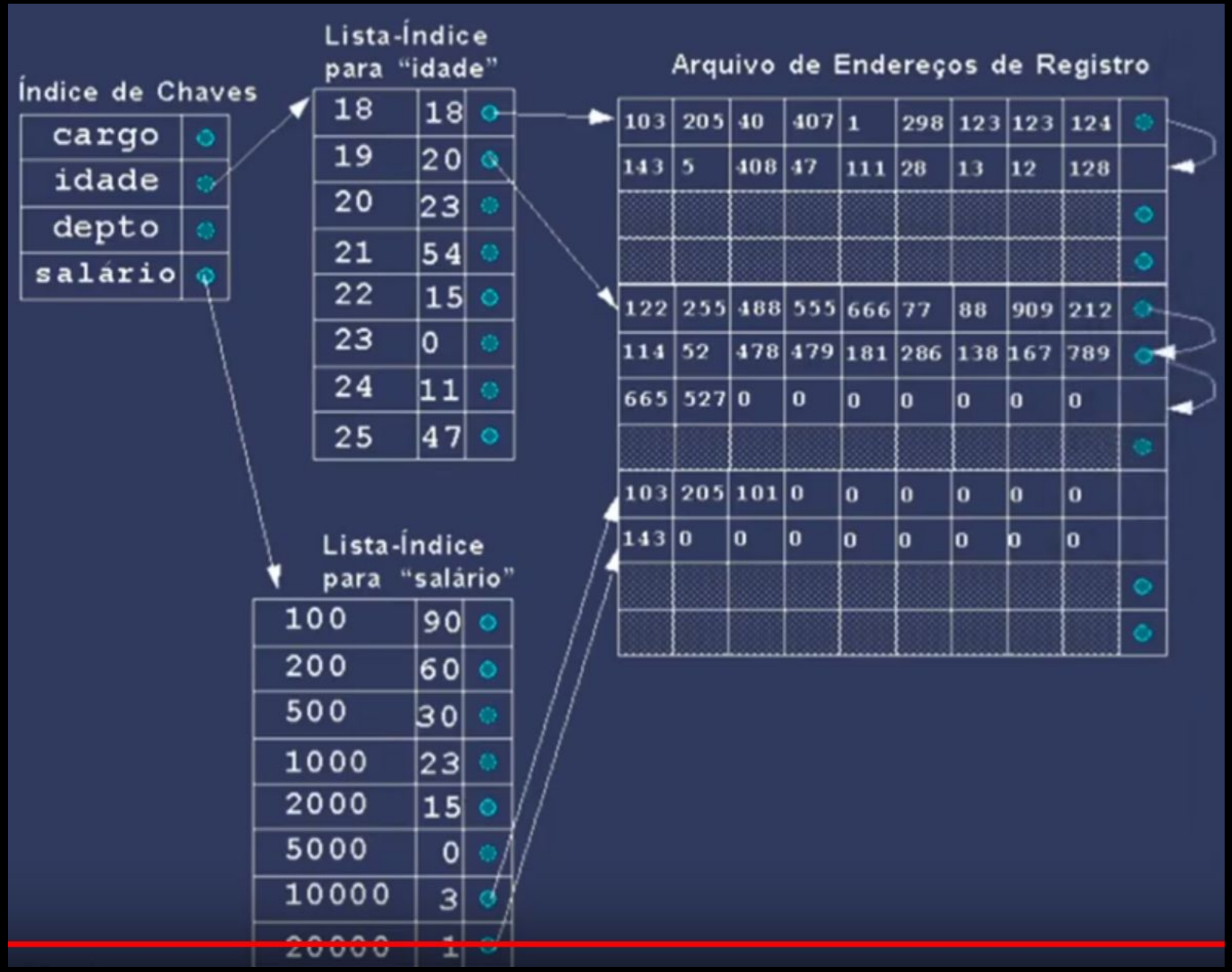
- Tem uma lista-índice para cada chave secundária, em que cada entrada é <i,q,w>
  i:campo de indexação que não ordena fisicamente
  q:quantidade de registros com o mesmo valor
  w: bloco
- Tem um arquivo de endereços de registros organizado em blocos
- Além do registro de dados em si



Fonte: https://www.youtube.com/watch?v=op7hJvKHVrA&list=PLmDIGfkfgKy1pouEOuiW1G681p7u_20O8&index=28

Lista invertida

O valor que está apontando para o endereço e não o endereço que está apontando para o valor, por isso é chamada de lista invertida.

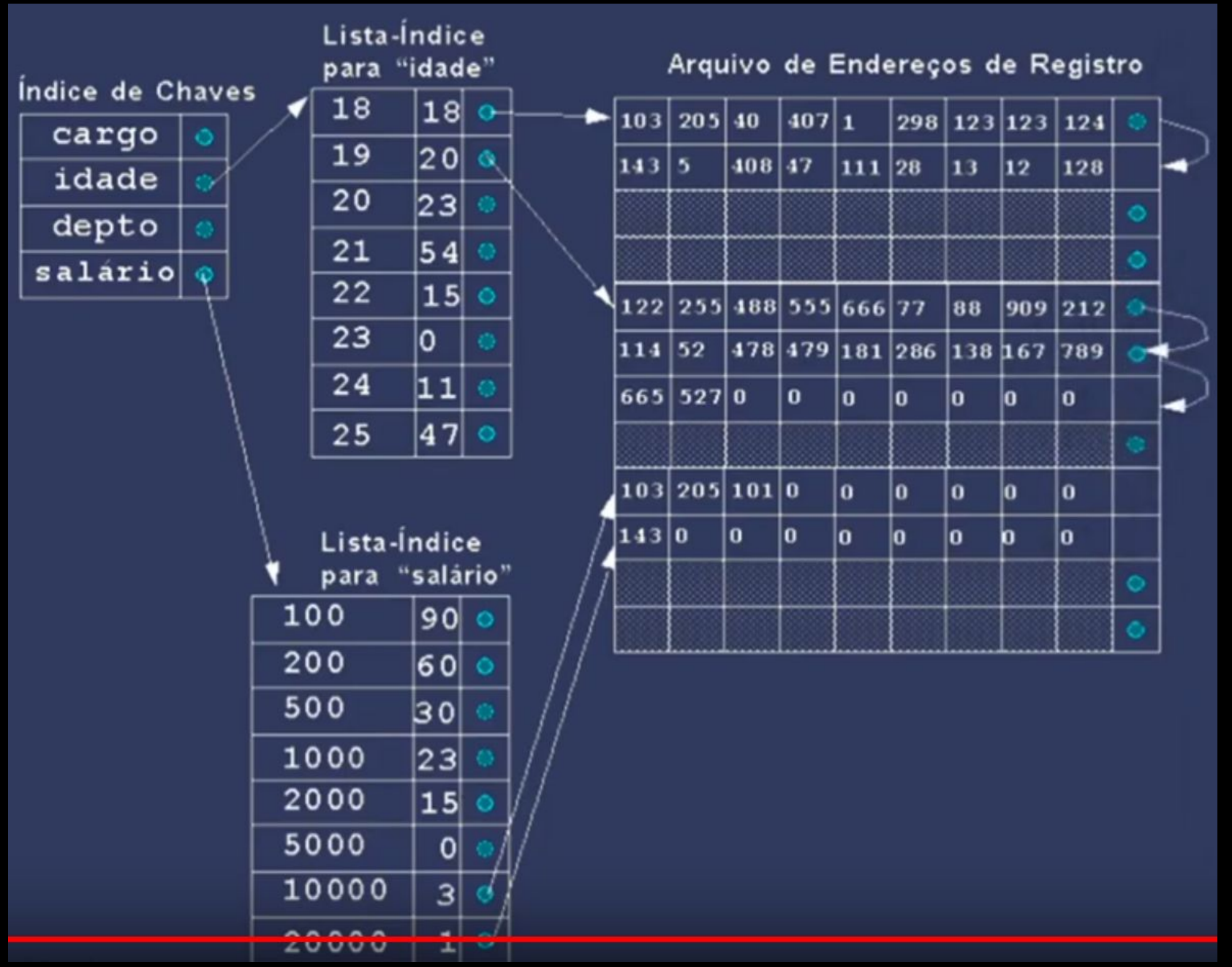


Fonte: https://www.youtube.com/watch?v=op7hJvKHVrA&list=PLmDIGfkfgKy1pouEOuiW1G681p7u_20O8&index=28

Lista invertida

Pesquisas de chaves em arquivos Lista invertida são realizadas através da geração de tabelas

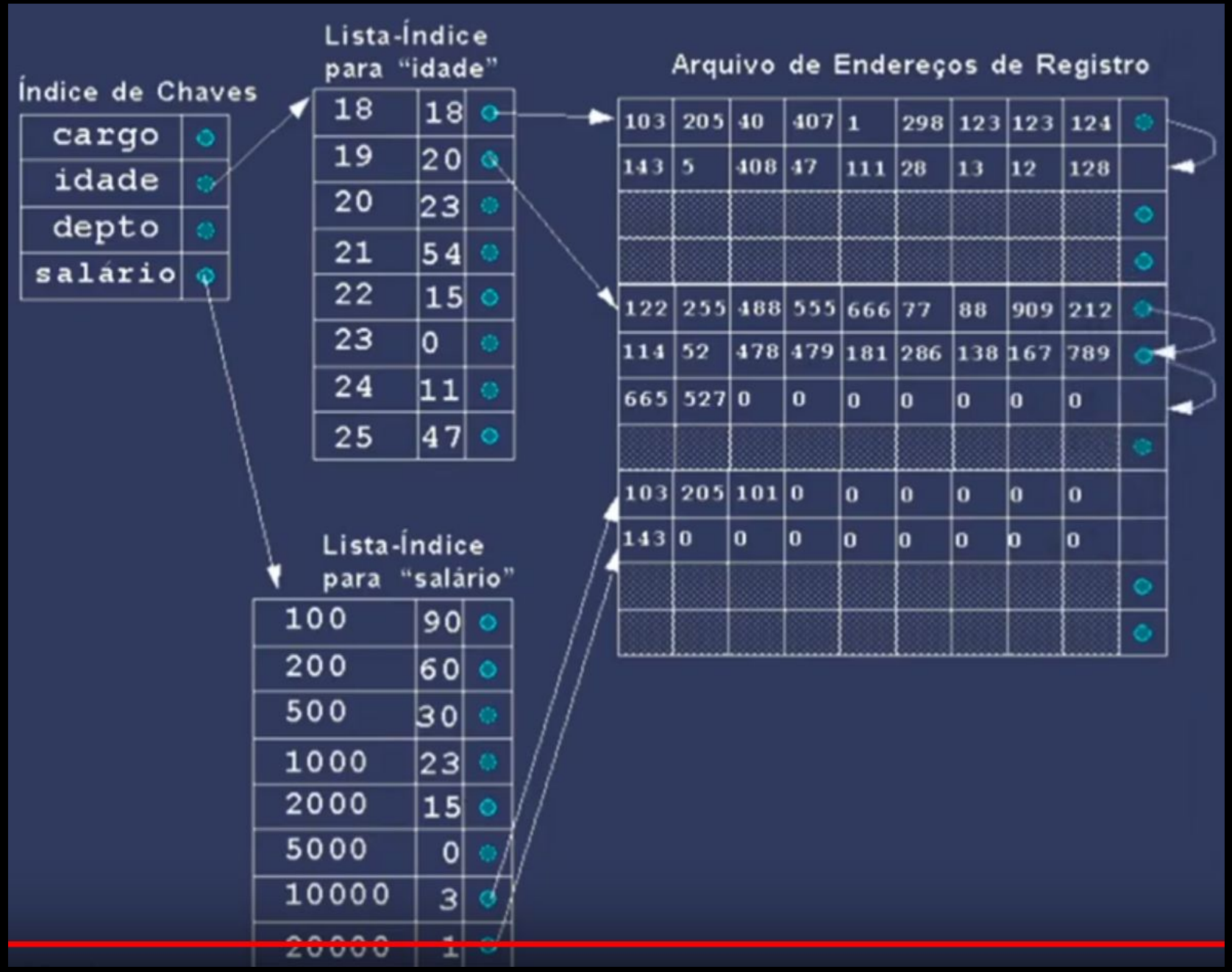
- **Pesquisas de chave simples** geram uma tabela com todos os registros com o valor da chave correspondente. Ex: registros com salário 10000 vai gerar um arquivo com 3 registros



Fonte: https://www.youtube.com/watch?v=op7hJvKHVrA&list=PLmDIGfkfgKy1pouEOuiW1G681p7u_20O8&index=28

Lista invertida

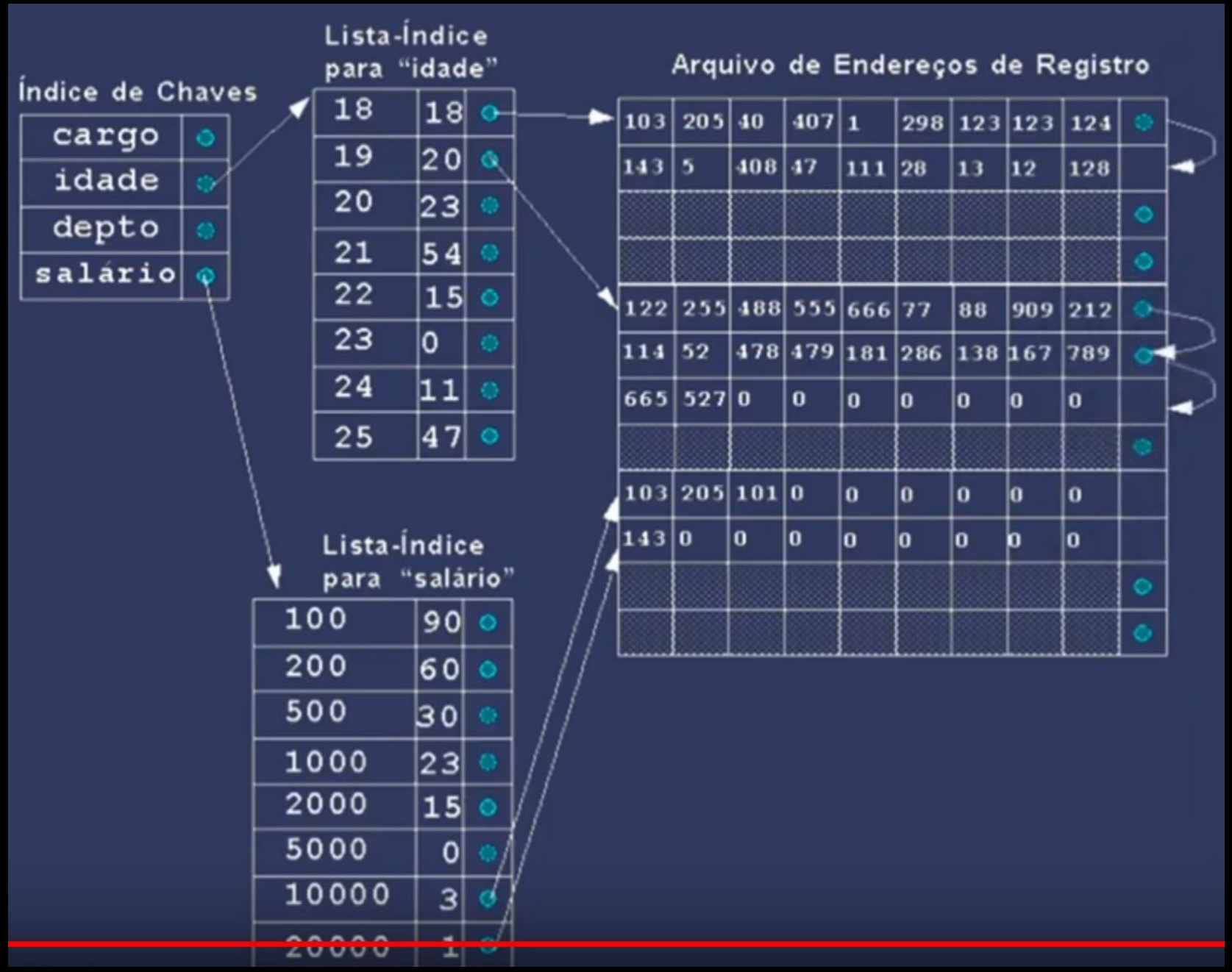
- **Pesquisas conjuntivas:** começo com o valor de chave com menor número de ocorrências e depois endereços não encontrados para as outras chaves são retirados.
  Ex: registros com salário 10000 e idade 18. Começo criando uma lista com os endereços dos registros com salário 10000 e o registro com endereço 101 é retirado, ficando no final com 2 registros.



Fonte: https://www.youtube.com/watch?v=op7hJvKHVrA&list=PLmDIGfkfgKy1pouEOuiW1G681p7u_20O8&index=28

Lista invertida

- **Pesquisas disjuntivas:** começo com o valor de chave com maior número de ocorrências e depois endereços não encontrados para as outras chaves são adicionados.
  Ex: registros com salário 10000 ou idade 18. Começo criando uma lista com os endereços dos registros com idade 18, e o registro com endereço 101 é adicionado, ficando no final com 18+1 registros.



Fonte: https://www.youtube.com/watch?v=op7hJvKHVrA&list=PLmDIGfkfgKy1pouEOuiW1G681p7u_20O8&index=28