Índices

Prof. Dr. Lucas C. Ribas

Disciplina: Estrutura de Dados II

Departamento de Ciências de Computação e Estatística

unesp
UNIVERSIDADE ESTADUAL PAULISTA
"JÚLIO DE MESQUITA FILHO"

IBILCE / UNESP - CÂMPUS DE SÃO JOSÉ DO RIO PRETO

Organização x Acesso

- Organização Física:
  - registros de tamanho variável
  - registros de tamanho fixo
- Acesso ao Arquivo:
  - Sequencial
  - Direto
- O que influencia:
  - o uso que se fará do arquivo
  - facilidades da linguagem de programação usada

unesp

Na aula de hoje: Índices

unesp

Índice

- Em geral, um índice fornece mecanismos para localizar informações
  - Índice de um livro ou catálogo de uma biblioteca
    - Facilitam muito o trabalho de busca!
- Em arquivos
  - Permite localizar registros rapidamente
  - Não é necessário ordenar arquivo de dados, nem quando novos registros são adicionados

unesp

Índice

- Índices Simples
  - Representados em vetores, cuja estrutura contém as chaves e os campos de referência
- Outros Esquemas de Indexação
  - Usam estruturas de dados mais complexas (árvores)

Índice simples

- Exemplo: uma enorme coleção de CDs
- Registros de tamanho variável
  - **ID Number:** Número de identificação
  - **Title**: Título
  - **Composer**: Compositor(es)
  - **Artist**: Artista(s)
  - **Label**: Rótulo (código da gravadora)

unesp

Índice simples

#bytes do registro

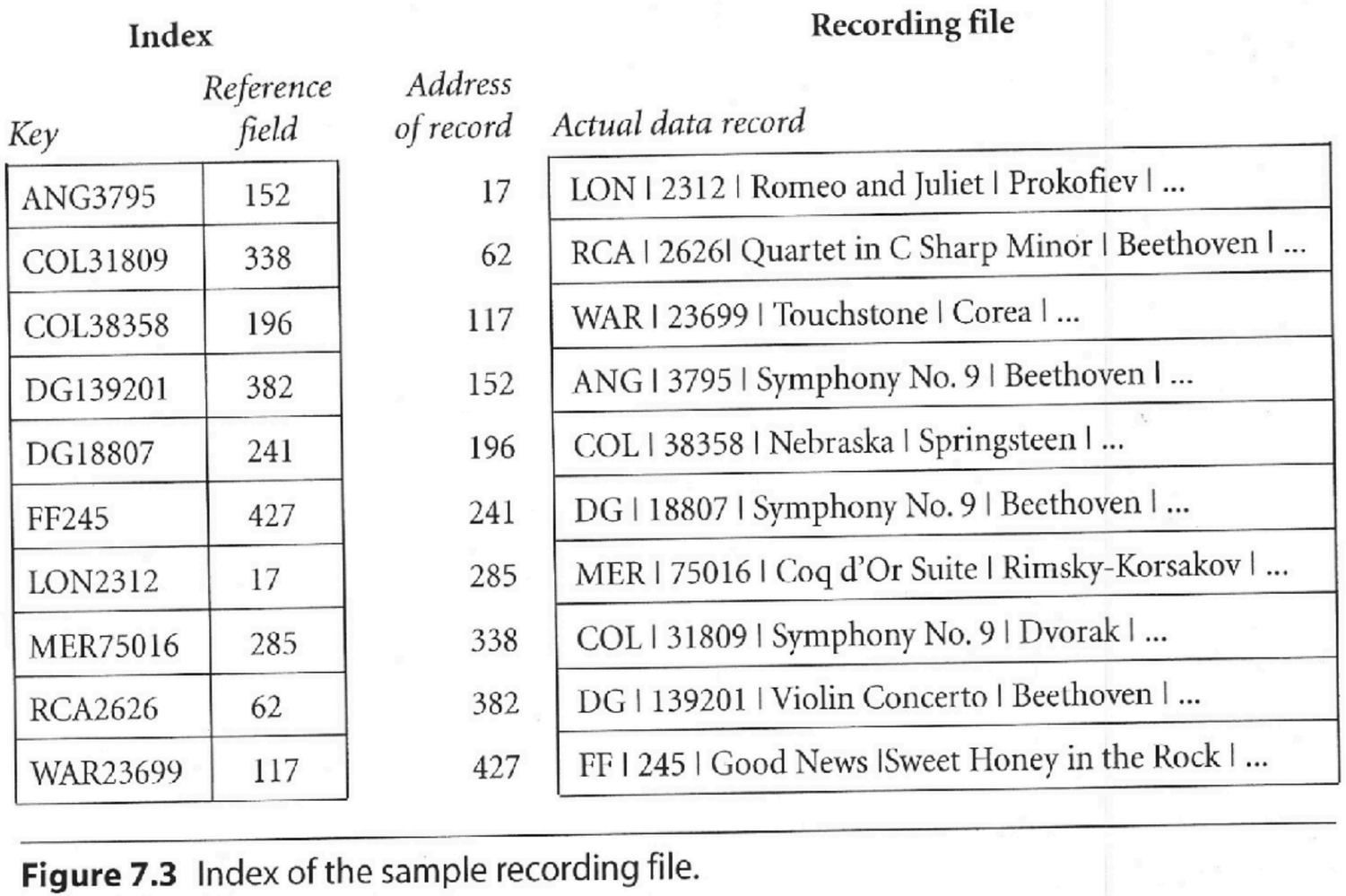
Record address	Label	ID number	Title	Composer(s)	Artist(s)
17	LON	2312	Romeo and Juliet	Prokofiev	Maazel
62	RCA	2626	Quartet in C Sharp Minor	Beethoven	Julliard
117	WAR	23699	Touchstone	Corea	Corea
152	ANG	3795	Symphony No. 9	Beethoven	Giulini
196	COL	38358	Nebraska	Springsteen	Springsteen
241	DG	18807	Symphony No. 9	Beethoven	Karajan
285	MER	75016	Coq d'Or Suite	Rimsky-Korsakov	Leinsdorf
338	COL	31809	Symphony No. 9	Dvorak	Bernstein
382	DG	139201	Violin Concerto	Beethoven	Ferras
427	FF	245	Good News	Sweet Honey in the Rock	Sweet Honey in the Rock

Figure 7.2 Contents of sample recording file.

Chave primária: combinação de **Label + ID Number**

Poderia ser qualquer outro campo ou combinação de campos que fosse único para cada registro

Índice simples



Index

Key	*Reference field*
ANG3795	152
COL31809	338
COL38358	196
DG139201	382
DG18807	241
FF245	427
LON2312	17
MER75016	285
RCA2626	62
WAR23699	117

Recording file

Address of record	*Actual data record*
17	LON | 2312 | Romeo and Juliet | Prokofiev | ...
62	RCA | 2626| Quartet in C Sharp Minor | Beethoven | ...
117	WAR | 23699 | Touchstone | Corea | ...
152	ANG | 3795 | Symphony No. 9 | Beethoven | ...
196	COL | 38358 | Nebraska | Springsteen | ...
241	DG | 18807 | Symphony No. 9 | Beethoven | ...
285	MER | 75016 | Coq d'Or Suite | Rimsky-Korsakov | ...
338	COL | 31809 | Symphony No. 9 | Dvorak | ...
382	DG | 139201 | Violin Concerto | Beethoven | ...
427	FF | 245 | Good News |Sweet Honey in the Rock | ...

Figure 7.3 Index of the sample recording file.

Índice simples

- O índice consiste, em geral, em um outro arquivo com registros de tamanho fixo
  - Mesmo que o arquivo principal com os dados não tenha registros de tamanho fixo
- Cada registro do índice contém pelo menos 2 campos de tamanho fixo
  - Chave
  - Posição inicial (*byte offset*) ou RRN do registro no arquivo de dados

unesp

Índice simples

- A cada registro do arquivo de dados corresponde um registro no índice
- O índice está ordenado, apesar do arquivo de dados não estar
  - Em geral, o arquivo de dados está organizado segundo a ordem de entrada dos registros (*entry sequenced file*)
- Vantagens do arquivo de índice sobre o de dados
  - Mais fácil de trabalhar, pois usa registros de tamanho fixo
  - Pode ser pesquisado com busca binária (em memória principal, inclusive, se valer a pena carregá-lo)
  - É muito menor do que o arquivo de dados

unesp

Índice simples

- Registros de tamanho fixo no arquivo índice impõem um limite ao tamanho da chave primária
- Os registros do índice poderiam conter outros campos além da chave/offset (por exemplo, o tamanho do registro)
- Como são feitas as operações básicas num arquivo indexado e “entry-sequenced”?
  - Inserção de registros
  - Remoção de registros
  - Atualização de registros
  - Busca de registros

unesp

Índice simples

- Como são feitas as operações básicas num arquivo indexado e “entry-sequenced”?
  - Inserção de registros
    - No arquivo de dados: no final;
    - Inserção de novo registro de chave no índice
  - Remoção de registros
    - Busca do registro usando o índice -> 1 seek + remoção
    - Remoção de registro de chave no índice
  - Atualização de registros
    - Busca no índice + seek
    - Se mudar chave, alterar tb o índice
  - Busca de registros
    - Busca no índice + seek

unesp

Índice simples

- A inclusão de registros será muito mais rápida se o índice pode ser mantido em memória interna e o arquivo de dados é *entry sequenced*
- Dados: a chave e o *offset*, um único *seek* é necessário no arquivo de dados para recuperar o registro correspondente

unesp

Operações básicas no índice

- Para índices que cabem em memória
  - Criar arquivos índice e de dados
  - Carregar índice para memória
  - Inserir registro
    - Inserção deve ser feita no arquivo de dados...
    - e também no índice, que eventualmente será reorganizado
  - Eliminar registro
    - Remove do arquivo de dados (aula anterior)
    - Remove também do índice
      - Índice pode ser reorganizado ou pode-se apenas marcar os registros excluídos

unesp

Operações básicas no índice

- Para índices que cabem em memória
  - Atualizar registro - duas categorias
    - Muda o valor da chave
    - Muda o conteúdo do registro
  - Atualizar índice no disco: caso sua cópia em memória tenha sido alterada
    - É imperativo que o programa se proteja contra índices desatualizados
      - Possíveis estratégias?

unesp

Como evitar índices desatualizados

- Deve haver um mecanismo que permita saber se o índice está atualizado em relação ao arquivo de dados
  - Possibilidade: um status flag é setado no arquivo índice mantido em disco assim que a sua cópia na memória é alterada
  - Esse flag pode ser mantido no registro header do arquivo índice, e atualizado sempre que o índice é reescrito no disco
    - Se um programa detecta que o índice está desatualizado, uma função deve ser ativada para reconstruir o índice a partir do arquivo de dados

unesp

Como evitar índices desatualizados

- Implementar em C uma sub-rotina que construa um índice em arquivo a partir de um arquivo de dados de alunos entry sequenced, com registros de tamanho variável

```
struct aluno {
   char *nome;
   int nro_Unesp;
}
fscanf(fp,”%d”,size);
fgets (str, tamanho, fp);

17 Fernando|18009|12 Maria|19001...
```

unesp

Índices muito grandes

- Se o índice não cabe inteiro na memória, o acesso e manutenção precisam ser feitos em memória secundária
- Não é mais aconselhável usar índices simples, uma vez que
  - A busca binária pode exigir vários acessos a disco
  - A necessidade de deslocar registros nas inserções e remoções de registros tornaria a manutenção do índice excessivamente cara

unesp

Índices muito grandes

- Utilizam-se outras organizações
  - Hashing, caso a velocidade de acesso seja a prioridade máxima
    - Acesso direto apenas
  - Árvores-B, caso se deseje combinar acesso por chaves e acesso sequencial eficientemente

unesp

Índices muito grandes

- Ainda assim, o uso de índice simples que não cabe em RAM ainda é mais vantajoso do que manter um arquivo de dados ordenado e não indexado
  - Tem registros de tamanho fixo, portanto, é possível fazer busca binária;
  - É bem menor do que arquivo de dados, facilitando a manutenção;
  - Índices fornecem múltiplas visões de um conjunto de dados

unesp

Acesso por múltiplas chaves

- Como decidimos qual será a chave primária dos registros?
  - Identifica unicamente, mas só tem valor “organizacional”
- Normalmente, o acesso a registros não se faz por chave primária, e sim por chaves secundárias
  - Quando se procura a busca por um livro em um biblioteca, começa-se pelo seu número ou pelo título/autor?
- Solução: cria-se um índice que relaciona uma chave secundária à chave primária (e não diretamente ao registro)
  - Índice secundário

unesp

Acesso por múltiplas chaves

- Índices permitem muito mais do que simplesmente melhorar o tempo de busca por um registro

- **Múltiplos índices secundários**

  - Permitem manter diferentes visões dos registros em um arquivo de dados

    - Pode-se criar vários indices

  - Permitem combinar chaves associadas e, deste modo, fazer buscas que combinam visões particulares

    - Buscas que sejam a união das respostas de índices secundários.

    - Buscas que sejam a interseção das respostas de índices secundários

unesp

Acesso por múltiplas chaves

54	5
143	4
210	3
323	1
329	2
400	0

INDEX

0	400 | Minas | Milton Nascimento | Emi-Odeon | 1975
1	323 | Falso brilhante | Elis Regina | Philips | 1976
2	329 | A Arte de | Chico Buarque | PolyGram | 1982
3	210 | Chico Canta | Chico Buarque | Philips | 1985
4	143 | A Arte de | Milton nascimento | Universal | 1988
5	54 | Geraes | Milton Nascimento | Emi-Odeon | 1976

Chave primária

Index para Artista

CHICO BUARQUE	210
CHICO BUARQUE	329
ELIS REGINA	323
MILTON NASCIMEN	54
MILTON NASCIMEN	143
MILTON NASCIMEN	400

Índice secundário

- Campos chave na forma canônica → no formato que será usado na busca.

Neste caso:

- letras maiúsculas
- Tamanho max=15 catracteres

- Para exibir o nome no formato normal deve-se buscar o artista no arquivo de dados.

Alterações nas operações básicas

Inserir registro

- Quando um novo registro é inserido no arquivo, devem ser inseridas as entradas correspondentes no índice primário e nos índices secundários
  - melhor se índices (primário e secundários) couberem na RAM
- Campos do índice secundário são de tamanho fixo, embora no arquivo de dados esses valores possam estar em campos de tamanho variável (ex. nome do compositor ou da música)
  - O valor então pode ser truncado
  - Levar isso em conta quando estabelecer o tamanho do campo
- Diferença importante entre os índices primário e os secundários: nesses últimos pode ocorrer duplicação de chaves
  - **Chaves duplicadas** devem ser mantidas **agrupadas e ordenadas**

unesp

Alterações nas operações básicas

Eliminar registro

- Implica em remover o registro do arquivo de dados e de todos os índices (primário e secundários)
- Se índices mantidos ordenados, **rearranjo dos registros remanescentes** para não deixar "espaços vagos“
- **Alternativa:** atualizar apenas o índice primário, sem eliminar a entrada correspondente ao registro no índice secundário - busca retorna valor inválido
  - Como o índice secundário referencia o índice primário (e não o registro físico no arquivo de dados), se for feita uma busca por um registro já removido essa condição será acusada na busca pela chave primária feita no índice primário, e a não remoção da entrada do índice secundário.

unesp

Alterações nas operações básicas

- **Vantagem**: economia de tempo substancial quando vários índices secundários estão associados ao arquivo primário, principalmente se esses índices são mantidos em disco

- **Custo:**
  - espaço ocupado por registros inválidos. Para reduzir:
    - Pode-se fazer "coletas de lixo" periódicas nos índices secundários
    - Ainda será um problema se o arquivo for muito volátil
      - Outra solução: índice em árvore-B
  - previsão no algoritmo de busca: embora presente no índice secundário, não haverá registro correspondente no primário

unesp

Alterações nas operações básicas

- **Atualizar registro** - 3 situações
  - Alterou uma chave secundária: o índice secundário para esta chave, se houver, precisa ser reordenado
  - Alterou a chave primária: reordenar o índice primário e corrigir os campos de referência dos índices secundários
    - Atualização dos índices secundários não requer reorganização
  - Alterou outros campos: não afeta nenhum dos índices
    - E se o tamanho do registro mudar?

unesp

Busca usando múltiplas chaves

- Uma das aplicações mais importantes das chaves secundárias é localizar conjuntos de registros do arquivo de dados usando uma ou mais chaves
- Pode-se fazer uma busca em vários índices e combinar (AND,OR,NOT) os resultados
- Exemplo: encontre todos os registros de dados tal que
  - composer = "BEETHOVEN" AND title = "SYMPHONY NO. 9"

Busca usando múltiplas chaves

→ **Exemplo: arquivo de CDs, e dois arquivos secundários para ele, com chaves Artista e Rótulo**

0	400	Minas	Milton Nascimento	Emi-Odeon	1975
1	323	Falso brilhante	Elis Regina	Philips	1976
2	329	A Arte de	Chico Buarque	PolyGram	1982
3	210	Chico Canta	Chico Buarque	Philips	1985
4	143	A Arte de	Milton nascimento	Universal	1988
5	54	Geraes	Milton Nascimento	Emi-Odeon	1976

54	5
143	4
210	3
323	1
329	2
400	0

Índice primário

CHICO BUARQUE	210
CHICO BUARQUE	329
ELIS REGINA	323
MILTON NASCIMEN	54
MILTON NASCIMEN	143
MILTON NASCIMEN	400

EMI-ODEON	54
EMI-ODEON	400
PHILIPS	210
PHILIPS	323
POLYGRAM	329
UNIVERSAL	143

Índices secundários

Busca usando múltiplas chaves

0	400	Minas	Milton Nascimento	Emi-Odeon	1975
1	323	Falso brilhante	Elis Regina	Philips	1976
2	329	A Arte de	Chico Buarque	PolyGram	1982
3	210	Chico Canta	Chico Buarque	Philips	1985
4	143	A Arte de	Milton nascimento	Universal	1988
5	54	Geraes	Milton Nascimento	Emi-Odeon	1976

54	5
143	4
210	3
323	1
329	2
400	0

Resultado da consulta 4

Encontre CD's com Artista="Chico Buarque"e Rótulo="Philips"

a. encontre CD's com Artista= "Chico Buarque"

CHICO BUARQUE	210
CHICO BUARQUE	329

b. encontre todos os CD's com Rótulo= "Philips"

PHILIPS	210
PHILIPS	323

EMI-ODEON	54
EMI-ODEON	400
PHILIPS	210
PHILIPS	323
POLYGRAM	329
UNIVERSAL	143

c. Faça AND dos resultados (referência à chaves primárias)

210

d. Buscar registro com Artista="Chico Buarque"e Rótulo="Philips"

210	Chico Canta	Chico Buarque	Philips	1985

Melhoria de índices secundários

- Dois problemas na estrutura de índices secundários até agora
  - Repetição das chaves secundárias (arquivos maiores…)
  - Necessidade de reordenar os índices sempre que um novo registro é inserido no arquivo, mesmo que esse registro tenha um valor de chave secundária já existente no arquivo

Melhoria de índices secundários

- **Solução 1:** associar uma lista de tamanho fixo a cada chave secundária
  - Não seria mais necessário reordenar o índice a cada inserção de registro
  - Porém:
    - Limitado a um número fixo de repetições
    - Ocorre enorme fragmentação interna no índice - que talvez não compense a eliminação da duplicação de chaves

CHICO BUARQUE	210
CHICO BUARQUE	329
ELIS REGINA	323
MILTON NASCIMEN	54
MILTON NASCIMEN	143
MILTON NASCIMEN	400

CHICO BUARQUE	210	329			
ELIS REGINA	323				
MILTON NASCIMEN	54	143	400		

Melhoria de índices secundários

- **Solução 2:** manter uma lista de referências - listas invertidas
  - Pode-se associar cada chave secundária a uma lista encadeada das chaves primárias referenciadas
  - Índice secundário passa a ser composto por registros com 2 campos: campo chave e campo com o RRN/byte offset do primeiro registro com essa chave na lista invertida
  - As referências às chaves primárias associadas a cada chave secundária são mantidas em um arquivo seqüencial separado, organizado segundo a entrada dos registros.
  - O índice de chaves secundárias funciona da seguinte forma:
    - cada chave secundária leva a uma ou mais chaves primárias (daí o termo "invertida“ a chave secundária leva à lista de chaves primárias, a qual por sua vez leva aos registros...)

unesp

Melhoria de índices secundários

Solução 2: Ligar à lista de referências - Listas invertidas

- **Exemplo:Considerando o arquivo de dados de Cds.**

400	Minas	Milton Nascimento	Emi-Odeon	1975
323	Falso brilhante	Elis Regina	Philips	1976
329	A Arte de	Chico Buarque	PolyGram	1982
210	Chico Canta	Chico Buarque	Philips	1985
143	A Arte de	Milton nascimento	Universal	1988
54	Geraes	Milton Nascimento	Emi-Odeon	1976

54	5
143	4
210	3
323	1
329	2
400	0

Índice primário

CHICO BUARQUE	210
CHICO BUARQUE	329
ELIS REGINA	323
MILTON NASCIMEN	54
MILTON NASCIMEN	143
MILTON NASCIMEN	400

Índice secundário

≅

CHICO BUARQUE	3
ELIS REGINA	1
MILTON NASCIMEN	5

+

0	400	-1
1	323	-1
2	329	-1
3	210	2
4	143	0
5	54	4

Índice secundário com lista invertida

Melhoria de índices secundários

- **Listas invertidas. Vantagens desta estratégia?**
  - Índice secundário só é alterado quando é inserido um registro com chave secundária inexistente
    - Operações de eliminação, inserção ou alteração de registros já existentes implicam apenas em alterar arquivo das listas encadeadas
    - Ordenação do arquivo de índice secundário é mais rápida: menos registros - e registros menores
  - Arquivo com listas encadeadas nunca precisa ser ordenado, pois é *entry sequenced*
  - É fácil reutilizar o espaço liberado pelos registros eliminados do arquivo de listas encadeadas, já que os registros são de tamanho fixo.

unesp

Melhoria de índices secundários

- **Listas invertidas. Problemas desta estratégia?**
  - Registros associados a um chave secundária não estão necessariamente adjacentes no disco: podem ser necessários vários seeks para recuperar os registros de uma lista encadeada.
  - O ideal seria manter o índice e a lista encadeada na memória

unesp

Associação do índice ao endereço físico do registro

- **Em índices primários** -> a associação (*binding)* ocorre no momento em que o arquivo é criado
  - fornece acesso direto e, portanto, mais rápido, a um registro, dada a sua chave
- **Em índices secundários** -> associadas a um endereço ocorre apenas no momento em que são de fato usadas
  - Isso é possível visto que os índices não se referem diretamente ao endereço físico dos registros, mas ao índice primário.

unesp

Associação do índice ao endereço físico do registro

- As chaves secundárias são associadas a um endereço apenas no momento em que são de fato usadas (*late binding*)
  - mínima quantidade de reorganização quando os registros são adicionados ou excluídos
  - Alterações importantes são feitas em um lugar em vez de em muitos lugares
  - Desvantagem: Isso implica em um acesso mais lento (principalmente se indices estiverem em disco), ***busca binária em indices secundários + primários***
- O *early binding* só é aconselhável se o arquivo de dados é (quase) estático, e o acesso rápido a registros é a maior prioridade

Exercícios

1- Escreva um algoritmo para uma função de busca de um registro em arquivos de índice simples.

2- Por que, para a eliminação de um registro do arquivo de dados, é possível eliminar o registro apenas do índice primário, e não do secundário?

3- Explique a importância do flag "out-of-date" no header de um arquivo de índices utilizado num ambiente de multiprogramação.

4- O que é uma lista invertida de índices? Quais são suas vantagens?

Exercícios - Solução

1- Se for de chaves primárias usar pesquisa binária.

Senão

- fazer busca binária no índice secundário
- obter todas as chaves encontradas que sejam iguais e
- selecionar a chave primária do registro procurado.

Exercícios - Solução

2- Porque o índice secundário não aponta para um registro físico e sim para o índice primário, assim quando a chave primária não existir (embora a chave secundária exista), saberemos que o registro não existe.

3- Esse *flag* é importante para garantir a consistência do arquivo de dados com o arquivo de índices, com vários programas que os acessam, pois indica que o arquivo de índices está atualizado de acordo com o arquivo de dados. Por exemplo, se houver uma queda do sistema, o flag indicará inconsistência caso o arquivo de índices não tenha sido atualizado (antes da queda). Neste caso, o arquivo de índices deve ser reconstruído, para garantir a consistência dos índices dos programas.

Exercícios - Solução

4- É uma lista de chaves primárias associado a uma chave secundária.

Suas vantagens são:

- eliminar a repetição das chaves secundárias (uma chave secundária aponta para uma lista de chaves primárias). Isso permite economizar espaço.
- inserção ou eliminação de um registro não é necessário alterar a tabela de índices secundários (movimentar os elementos ou reordená-los). Somente a lista invertida onde a chave primária será eliminada ou inserida, será atualizada.

Leitura recomenda

- Leitura recomendada: FOLK, M.J. File Structures, Addison-Wesley, 1992.

  Capítulos 6.

unesp

Referências

- FOLK, M.J. File Structures, Addison-Wesley, 1992.
- File Structures: Theory and Pratice”, P. E. Livadas, Prentice-Hall, 1990;
- Contém material extraído e adaptado das notas de aula dos professores Moacir Ponti, Thiago Pardo, Leandro Cintra e Maria Cristina de Oliveira.