

Acesso a Registros

Adaptado dos Originais de:

Leandro C. Cintra
Maria Cristina F. de Oliveira

1

Chaves

- Uma **chave (key)** está associada a um registro e permite a sua recuperação
 - É uma ferramenta conceitual importante
 - Pode-se caracterizar basicamente em:
 - Chave Primária
 - Chaves Secundárias

2

Chaves Primária e Secundária

- Uma **chave primária** é, por definição, a chave utilizada para identificar unicamente um registro
 - Exemplos:
 - No. USP
 - CPF
 - RG
 - ...
 - Sobrenome, por outro lado, não é uma boa escolha ...

3

Chaves Primária e Secundária

- Uma **chave secundária**, não necessariamente identifica unicamente um registro
 - pode ser utilizada para buscas por vários registros
 - por exemplo: todos os "**Silvas**" que moram em **São Paulo**

4



Escolha da Chave Primária

- A chave primária deve ser "dataless"
 - não deve ter um significado associado
 - não deve **mudar nunca**
- Existência de significado poderia implicar mudança do valor da chave
 - invalidaria referências já existentes, baseadas na chave antiga

5



Forma Canônica da Chave

- **Formas canônicas** para as chaves:
 - uma representação padronizada da chave
 - única, conforme com uma regra
 - por exemplo, todos os caracteres maiúsculos
 - "Ana", "ana" e "ANA": forma canônica será **ANA**

6



Busca Seqüencial

- Busca pelo registro que tem uma determinada chave, em um arquivo
 - Lê o arquivo, registro a registro, em busca de um registro contendo um certo valor de chave
 - Se cada registro lido demanda 1 acesso ao disco, tem-se custo de **O(n) acessos**
 - n acessos para um arquivo com n registros

7



Busca Seqüencial

- O custo de buscar e ler um registro, depois buscar e ler outro, é geralmente maior que o custo de buscar e ler dois registros sucessivos de uma só vez
 - apenas 1 *seeking* se ambos estiverem no mesmo cluster
- **Pode-se melhorar o desempenho da busca seqüencial lendo um **bloco** de registros por vez, e então processar este bloco em RAM**

8



Busca Seqüencial

- Exemplo de Blocagem:
 - Arquivo com 4.000 registros
 - Busca seqüencial por um registro, sem blocagem, requer, em média, **2.000** acessos
 - Se um bloco é capaz de armazenar, em média, 16 registros, o número médio de acessos cai para **125**

9



Busca Seqüencial

- Blocagem:
 - Cada acesso gasta um pouco mais de tempo, mas o ganho é considerável
 - redução do número de *seekings*
 - melhora o desempenho, mas o custo continua diretamente proporcional ao tamanho do arquivo, ou seja, **O(n) acessos**
 - somente constante é reduzida

10



Busca Seqüencial

- Fácil de programar
- Requer estruturas de arquivos simples
- Aceitável ou Preferível:
 - Em arquivos com poucos registros
 - Em arquivos pouco pesquisados
 - Na busca por registros com um certo valor de chave secundária, para a qual se espera muitos registros (muitas ocorrências)
 - Na busca por todos os registros (e.g. mala direta)

11



Acesso Direto

- A alternativa mais radical ao acesso seqüencial é o **acesso direto** (ou **aleatório**)
- O acesso direto implica realizar um *seeking* lógico direto para o início do registro desejado e ler o registro imediatamente
 - em geral, um único acesso ao disco traz o registro
- É **O(1) acessos** se:
 - a posição lógica do início do registro for conhecida

12

Acesso Direto

- Para localizar a posição exata do início do registro no arquivo lógico, pode-se utilizar um arq. de índice separado
 - associação entre chaves e posições dos registros
 - pode ser estruturado para otimizar consultas
 - veremos posteriormente no curso ...
- Ou um mapeamento funcional entre chaves e posições
 - tabela hash externa ...
- Ou pode-se ter uma chave **RRN**
 - **RRN = Relative Record Number**
 - RRN = 0, 1, 2, 3, ... (posição relativa do registro dentro do arquivo)

13

Acesso Direto com RRN

- Acesso direto com **RRN** demanda o uso de registros de tamanho fixo T
 - Posição de início do registro (byte offset):
 - **Byte Offset = RRN * T**
 - Exemplo:
 - Registro na posição lógica 546 a partir do início
 - Tamanho de cada registro é 128
 - **Byte offset = 546 x 128 = 69.888 bytes**

14

Organização e Acesso de Arquivos

- **Organização de Arquivos**
 - campos de tamanho fixo ou variável
 - registros de tamanho fixo ou variável
 - técnicas de separação entre campos e registros
- **Acesso a arquivos**
 - acesso seqüencial
 - acesso direto

15

Organização e Acesso de Arquivos

- Considerações a respeito da organização:
 - arquivo pode ser dividido em campos ?
 - os campos podem ser agrupados em registros ?
 - campos e registros têm tamanho fixo ou variável ?
 - como separar os campos e registros ?
 - como identificar o espaço utilizado e o "lixo" ?
- Existem muitas respostas para estas questões...

16



Organização e Acesso de Arquivos

- Existem muitas respostas para estas questões ...
 - a escolha de uma organização em particular depende, entre outras coisas, do que se vai fazer com o arquivo
- Por exemplo, arquivos com regs. de tamanhos muito diferentes deveriam utilizar regs. de tamanho variável
 - mas não é possível acessá-los diretamente por RRN
 - requer ao menos verificar um arquivo de índice ...
 - Complexidade de acesso por RRN será de $O(1)$ acessos apenas se todo o arquivo de índice puder ser lido em memória de uma única vez ou se for constituído de índices com tamanho fixo em bytes

17



Organização e Acesso de Arquivos

- Pode ainda haver limitações de linguagem
- Exemplo:
 - C permite acesso a qualquer byte com **fseek**
 - permite implementar acesso direto a registros de tamanho variável uma vez que se conheça a posição de início do registro
 - Pascal permite *seeking* apenas para *records*
 - registros do mesmo tipo e tamanho
 - acesso direto a registros de tamanho variável não é viável

18



Registro Cabeçalho

- Em geral, é interessante manter algumas informações sobre o arquivo no seu início
 - cabeçalho no início do arquivo: **header record**
- Algumas informações típicas são:
 - datas de criação e atualização
 - número de registros
 - tamanho de cada registro (caso fixo)
 - campos de cada registro (caso fixo)
 - no de campos
 - tipo de cada campo – inteiro, string com delimitador, etc
 - byte offsets de cada registro... índice "externo" !

19



Observações

- Parte das informações atuais tratadas pelos computadores não se ajustam bem ao modelo de dados como seqüências de campos e registros
 - som, imagens, ...
- É mais fácil pensar em dados deste tipo como objetos que representam som, imagens, etc.
 - possuem sua própria maneira de serem manipulados

20



Observações

- O termo **modelo abstrato de dados** captura a noção de que o dado não precisa ser visto da forma como está armazenado e vice-versa
 - permite uma visão dos dados orientada à aplicação
- **Metadados** podem ser vistos como realizações desses modelos
 - são dados que descrevem os dados
 - arquivos com conteúdo auto-explicável (ex. PDF, PS, ...)
 - permite portabilidade e facilita conversões de padrões

21



Exercícios

- Capítulo 4 (Folk & Zoellick, 1987)
- Lista de Exercícios (CoTeia)

22



Bibliografia

- **M. J. Folk and B. Zoellick, *File Structures: A Conceptual Toolkit*, Addison Wesley, 1987.**

23