APS – Algoritmos e Estruturas de Dados II

Índices com Árvore-B

Um apreciador de cervejas, insatisfeito com o tempo gasto para realizar as operações de busca relacionadas às cervejas já degustadas, resolveu contratá-lo para criar uma versão mais eficiente do programa de gerenciamento das cervejas já experimentadas.

A APS SURGE

Desenvolva um programa que permita ao colecionador organizar seus filmes. O programa deverá permitir:
1. Inserir uma nova cerveja.
2. Modificar o campo nota de uma cerveja.
3. Buscar cervejas a partir de uma chave primária.
4. Listar todas as cervejas no catálogo ordenadas pela chave primária.

Para realizar essa tarefa será necessário organizar 2 arquivos distintos:
(a) um arquivo que conterá todos os registros e (b) um arquivo de índice primário.

Estrutura do arquivo de dados

O arquivo de dados deve ser ASCII (arquivo texto) e organizado em registros de tamanho fixo de 128 bytes. Os campos de nome, fabricante, país e estilo devem ser de tamanho variável. Os demais campos devem ser de tamanho fixo: teor alcoólico (4 bytes), nota de 0 a 5 (1 byte) e chave primária (5 bytes). A soma de bytes dos campos fornecidos (incluindo os delimitadores necessários) nunca irá ultrapassar 128 bytes. Os campos do registro devem separados pelo caractere delimitador @ (arroba). Cada registro terá 8 delimitadores, mais 10 espaços ocupados pelos campos de tamanho fixo. Você poderá fixar um tamanho máximo para cada campo: nome, fabricante, país e estilo de forma a garantir que ocupem, juntos, um máximo de 110 bytes.
Caso o registro tenha menos de 128 bytes, o espaço adicional deve ser marcado de alguma forma a completar os 128 bytes. Segue abaixo um exemplo com quatro registros. Os espaços adicionais foram preenchidos com o caractere #.

```
LEF01@Leffe Blonde@Leffe@Bélgica@Belgian Blond Ale@6,6%@5@##########
##################################################HEI01@Hein
eken@Heineken Brasil@Brasil@Lager@5,0%@4@##########################
##############################################WAL03@Session Citra@
Wals@Brasil@India PaleAle@3.9%@5@##################################
##################################CON02@Conti Zero Grau@Conti@Bra
sil@Lager-Pilsen@4,4%@0@############################################
##########################
```

Não há quebras de linhas no arquivo (elas foram inseridas aqui apenas para exemplificar). O arquivo de dados não deverá conter cabeçalho e deverá se chamar brejas.dat. Instruções para as operações com registros:

- Inserção: cada cerveja inserida no catálogo dever ser inserida no final do arquivo de dados e atualizado nos índices.
- Atualização: o registro deverá ser localizado acessando o índice primário. A nota deverá ser atualizada no registro na mesma posição em que está (não deve ser feita remoção seguida de inserção).

RRN (Relative Record Number ou Número Relativo do Registro): O endereço de um registro é o deslocamento em bytes (offset) desde o início do arquivo (ou do cabeçalho). No caso de registros de tamanho fixo, podemos armazenar no índice apenas o número de registros que antecedem cada registro (RRN). Neste caso, o endereço de um dado registro é obtido multiplicando-se o seu RRN pelo comprimento dos registros.

O Índice

Deverá ser criado um arquivo chamado ind.idx, contendo as chaves primárias e os RRNs dos respectivos registros (e a estrutura da árvore). A forma de organização do índice deve ser uma Árvore-B. Os "ponteiros" para páginas armazenadas no arquivo devem ser os RRNs para localização das páginas no próprio arquivo, e as RRNs para registros devem apontar para o arquivo de dados. A árvore-B deve seguir as seguintes especificações:
• Cada página da árvore deve comportar 4 entradas de índice.
• Os endereços (RRN) devem ser do tipo int de 4 bytes (tanto os RRNs dos descendentes de cada página, quanto os RRNs dos registros do arquivo de dados). RRNs nulos (não utilizados) devem ter valor "-1".
Delimitem as páginas e os registros (no exemplo utilizei @ para delimitar os registros)
Os índices devem ser armazenados utilizando o seguinte modo:

RRN_{pag1}<$Chave_1$,RRN_{dados1}>@RRN_{pag2}<$Chave_2$,RRN_{dados2}>@RRN_{pag3}<$Chave_3$,RRN_{dados3}>@RRN_{pag4}<$Chave_4$,RRN_{dados4}>@RRN_{pag5}

RRN_{pagi} é o RRN da página filha ***i*** *(no arquivo de índices)*, e <$\boldsymbol{Chave_j}$,$\boldsymbol{RRN_{dados\,j}}$> é um par: $Chave_j$ = chave primária do registro **j**, e $RRN_{dados\,j}$ = RRN do registro **j** (*no arquivo de dados)*. A linha acima representa uma única página da Árvore-B, a próxima página deve começar imediatamente após o término da primeira página, utilizando algum caracter delimitador.

O Usuário

O programa deve permitir interação com o usuário pelo shell. As seguintes operações devem estar disponíveis:

- Inserção de cerveja : O usuário deve ser capaz de inserir uma nova cerveja, com os seguintes campos: ID (chave primária), nome, fabricante, país, estilo, teor alcoólico e nota
- Busca de cerveja : O usuário deve poder buscar por uma cerveja pela chave primária. Caso a cerveja não exista, seu programa deve informar tal fato ao usuário. Caso a cerveja exista, todos os seus dados devem ser impressos na tela (bonitinho).
- Finalizar a execução : O usuário deve ser capaz de encerrar a execução do programa.

A Implementação

Implementar uma biblioteca de manipulação de arquivos para o seu programa, contendo as seguintes funcionalidades:

1. Uma estrutura de dados Árvore-B.
2. Verificar se o arquivo de dados existe
3. Verificar se o índice primário existe
4. Criar/recriar o índice primário: deve refazer o índice primário a partir do arquivo de dados e substituir, caso haja, um índice existente no disco.
5. Inserir um registro: modificando o arquivo de dados no disco, e o índice na memória principal.
6. Utilizar as linguagens C ou C++. Separar a interface da implementação