CK0117 - Sistemas de Bancos de Dados - 2022-1

Javam Machado; Daniel Praciano; Edvar Filho

TRABALHO I - ÍNDICES

1 Aspectos Gerais

Um banco de dados armazenado em disco possui uma relação que salva as informações relativas aos vinhos produzidos em uma vinícola. A Figura 1 apresenta o esquema da relação **Vinho**. Normalmente, a carga de trabalho que envolve essa relação realiza uma consulta considerando o ano no qual a uva utilizada para produzir o vinho foi colhida. Assim, o trabalho consiste em implementar um índice para essa relação, sendo que a chave de busca é o atributo **ano_colheita**, que é um número inteiro.

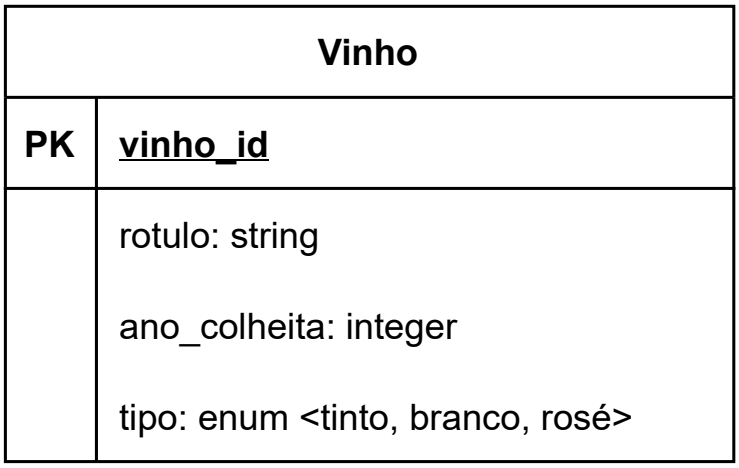

Figure 1: Relação a ser indexada.

Para esse trabalho, deverão ser utilizados os dados presentes na instância da relação **Vinho** que está no arquivo **vinho.csv** que será publicado junto a esse documento.

2 Tipos de Índice

Cada equipe de, *no máximo*, dois alunos implementará **um e somente um** dos seguintes tipos de índice. A escolha do tipo de índice de cada equipe será por meio de sorteio e, independente da escolha, os dois índices devem considerar que a relação **Vinho** está armazenada em um arquivo do disco no qual cada linha (registro) desse arquivo é uma página do banco de dados.

A seguir, apresentamos a especificação da interface que cada índice deve implementar:

Árvore B+: Implemente as operações de busca (por igualdade e também por intervalo), inserção e remoção de entradas de dados.

Hash Extensível: Implemente as operações de busca (por igualdade), inserção e remoção de entradas de dados.

3 Implementação

Nesse trabalho, um banco de dados será mapeado em **disco** de modo que dois ou mais arquivos o representem: um arquivo de dados condizente à relação, contendo um registro por linha, em texto e um ou mais arquivos de índice, contendo as informações do índice.

Para o índice em árvore B+, pode-se utilizar uma linha do arquivo como armazenamento do nó, com os ponteiros contendo o nome dos arquivos seguintes. Para o índice em hash, pode-se utilizar os nomes dos arquivos como diretórios. Essa é uma sugestão que poderá facilitar a implementação, mas não é obrigatório seguir essa organização.

Por outro lado, serão obrigatórios os seguintes pontos para ambos os tipos de índices:

- A implementação deverá ser feita somente nas linguagens **C**, **C++**, **Java** ou **Python**. Nada além disso!
- Somente uma única página de dados deve estar em memória por vez. Ou seja, não é permitido manter todos os dados em memória simultaneamente. Únicas exceções para essa restrição: raiz da árvore B+ e diretório do hash.
- Deverá existir um arquivo **Main** que será responsável por ler um arquivo de texto com as informações de entrada, bem como gerar um arquivo de texto com as informações de saída. Mais informações a seguir.

O arquivo **Main** será o responsável por realizar chamadas as funcionalidades implementadas de cada índice. Para tanto, um único arquivo de entrada **in.txt** será fornecido. Nesse arquivo de entrada, estarão as operações de índice que devem ser realizadas, sendo que cada linha desse arquivo é composta por uma das alternativas seguintes:

```
INC:x
REM:x
BUS=:x
BUS>:x
BUS<:x
```

De maneira intuitiva, INC, REM, BUSC=, BUS>, BUS< , representam, respectivamente, a operação, no índice, de inclusão, remoção, busca por igualdade, busca por maior que e, finalmente, busca por menor que. Ademais, em todas as operações, **x** representa o inteiro a ser usado nas respectivas operações. **Importante**: se uma operação não faz sentido para o seu tipo de índice, por exemplo *BUS>:x* para hash, somente a ignore.

A primeira linha do arquivo de entrada, antes das operações em si, irá indicar a quantidade de filhos de um nó para árvore B+ ou a profundidade global para hash extensível. A sintaxe dessa linha é a seguinte:

- Para árvore B+:

  ```
  FLH/<quantidade de filhos que um nó pode ter>
  ```

- Para hash extensível:

  ```
  PG/<profundidade global inicial>
  ```

Por fim, um arquivo de saída nomeado de **out.txt** deve ser gerado. A linha inicial do arquivo deve ser igual à primeira linha do arquivo de entrada e, após cada operação do arquivo de entrada ser realizada, deve ser incluída uma linha no arquivo de saída da seguinte forma:

- Para árvores B+:

```
INC:x/<quantidade de tuplas inseridas>
REM:x/<quantidade de tuplas removidas>
BUS=:x/<quantidade de tuplas selecionadas>
BUS>:x/<lista de todas as chaves maiores que x separadas por vírgula>
BUS<:x/<lista de todas as chaves menores que x separadas por vírgula>
```

- Para hash extensível:

```
INC:x/<qtd de tuplas inseridas>,<profundidade global>,<profundidade local>
REM:x/<qtd de tuplas removidas>,<profundidade global>,<profundidade local>
BUS:x/<quantidade de tuplas selecionadas>
```

Por fim, a última linha do arquivo de saída do índice em árvore B+ deve ser H/<altura da árvore>. Por outro lado, no arquivo do índice em hash extensível, deve-se adicionar a seguinte linha sempre logo após a inclusão de um elemento que duplica o diretório: DUP_DIR:/<profundidade global>,<profundidade local>. Além disso, a última linha do arquivo de saída deve ser P:/<profundidade global final>.

4 Entrega

Data da entrega: Quinta-feira - 28 de abril de 2022 até 23h59m com apresentação e arguição no LEC/DC no dia seguinte, 29 de abril, no horário da aula. O código do trabalho deve ser enviado no **classroom** até o final do horário da entrega. Envios posteriores serão penalizados. Quaisquer dúvidas podem ser enviadas aos monitores: Daniel Praciano (daniel.praciano@lsbd.ufc.br) ou Edvar Filho (edvar.filho@lsbd.ufc.br).