CK0117 - Sistemas de Bancos de Dados - 2019-1

Javam Machado; Daniel Praciano

TRABALHO II - Operadores Algébricos

1 Aspectos Gerais

O trabalho consiste em implementar as operações básicas existentes em um SGBD relacional: **seleção**, **projeção** e **junção** (descritos na seção 2). Na implementação, desconsidere o uso de SGBDs. Use um índice *Hash* nas operações em que a sua utilização for necessária, procure, na linguagem escolhida, uma lib que implementa tal índice, caso não encontre, a equipe deve implementar o hash estático (sem *overflow*).

Para executar os operadores implementados, vamos utilizar o esquema apresentado na figura 1. Isto é, todos os operadores receberão como entrada uma (ou mais) tabela presente nesse esquema, dependendo se o operador é unário ou binário. Mais detalhes de implementação serão descritos na seção 3.

Resumidamente, observa-se, a partir do esquema apresentado, que existe uma tabela $Vinho$, que possui como chave primária (PK) *vinho_id* e duas chaves estrangeiras (FK): *uva_id* que referencia o atributo de mesmo nome da tabela Uva e *pais_producao_id* que referencia *pais_id* da tabela $Pais$. Ademais, essa mesma situação ocorre entre as tabelas Uva e $Pais$.

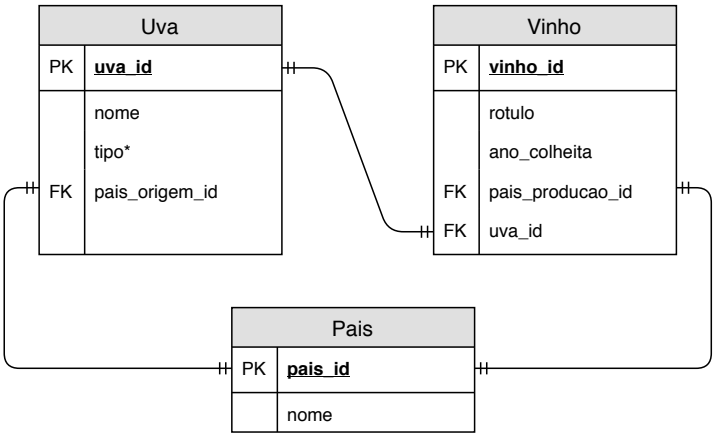

Figure 1: Modelagem E-R do esquema a ser utilizado.

2 Lista de Operadores

Cada equipe de, *no máximo*, dois alunos implementará **um e somente um** dos seguintes operadores de maneira **exclusiva**. Além disso, o resultado do operador deve conter as **tuplas retornadas da operação**, mas as tabelas originais **não devem ser modificadas**.

Index Selection (IS): Seleção sobre uma tabela utilizando um índice. Para tal considere um predicado de seleção com igualdade.

Distinct Sort Projection (DSP): Projeção com eliminação de duplicatas realizando a ordenação. Para tal você precisa realizar a ordenação externa da relação.

Distinct Hash Projection (DHP): Projeção com eliminação de duplicatas construindo as partições (*buckets*). Você deve utilizar uma implementação de índice *hash* no trabalho, como definido na seção 1.

Nested Loop Block Join (NLBJ): Junção com laço aninhado orientado a bloco - permitir a variação do número de frames de memória para a tabela *outer* a fim de testar o programa.

Nested Loop Index Join (NLIJ): Junção com laço aninhado utilizando índice. Você deve utilizar uma implementação de índice *hash* no trabalho, como definido na seção 1.

Sort Merge Join (SMJ): Junção utilizando a ordenação externa das tabelas. Grave as tabelas ordenadas no espaço de memória que simula um disco antes de fazer a etapa de comparação

Hash Join (HJ): Junção utilizando a criação de partições (*buckets*) das tabelas. Grave as tabelas particionadas no espaço de memória que simula um disco antes de fazer a partição.

3 Implementação

Nesse trabalho, um banco de dados será mapeado na **memória** de modo que cada tabela terá um conjunto de páginas sem quantidade definida, sendo que cada página terá um conjunto de 12 tuplas e cada tupla terá como base uma estrutura adjacente baseada no esquema da tabela. Uma possível maneira de implementar essas abstrações é utilizar as classes propostas na figura abaixo.

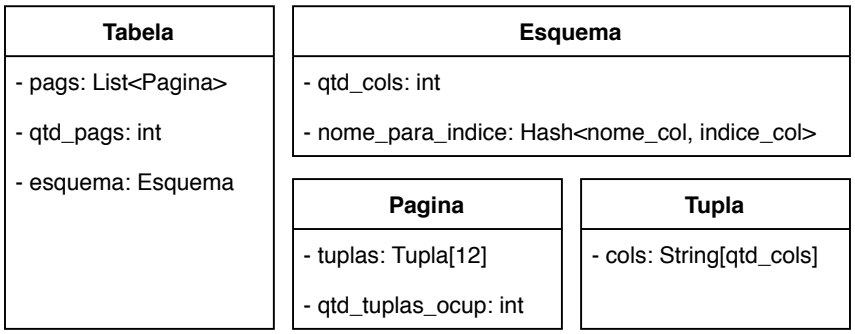

Figure 2: Sugestão de implementação por meio do diagrama de classes.

Essa é uma sugestão que poderá facilitar a implementação, mas não é obrigatório seguir essa organização. Por outro lado, será necessário utilizar o arquivo **Main** da respectiva linguagem na sua implementação. Esse arquivo contém uma interface que deverá ser implementada, juntamente com alguns casos de testes para validar a implementação.

4 Entrega

Data da entrega: Segunda-feira - 22 de abril de 2019 com apresentação e arguição no LEC/DC. O código do trabalho deve ser enviado por email para daniel.praciano@lsbd.ufc.br até o final do horário da entrega, i.e meio dia da mesma segunda-feira. Envios posteriores não serão aceitos.