

Documentação TP2 Banco de Dados

Franciane Alves de Almeida

Lorena Anunciação de Jesus

Vinicius Feitosa Monteiro

1. Estrutura dos arquivos de dados e índices

- **Record**

Registro contendo os campos passados na especificação do trabalho.

- **Block**

Corresponde a um bloco de registro na memória, ele possui dois registro e o resto do seu espaço é ocupado com fillers para poder ocupar o tamanho de um bloco.

- **Parser**

Percorre o arquivo de entrada, gera os registro e insere no arquivo hash.

- **HashFile**

Cria o arquivo hash com os registros do arquivo de entrada.

2. Quais fontes formam cada programa

- **upload**

- support/parser.hpp
- support/parser.cpp
- support/hashFile.hpp
- support/hashFile.cpp
- structures/record.hpp
- structures/block.hpp

3. Funções de que cada fonte contém

- **scr** - Contém todos os código fonte
- **scr/support** - Contém os códigos necessários para a leitura e interpretação do arquivo de entrada e criação do arquivo hash
- **scr/structures** - Contém as estruturas usadas para a criação do projeto

4. Quem desenvolveu cada fonte/função

A Lorena ficou encarregada da parte do parser, o Vinicius e a Franciane ficaram encarregados da parte do bloco e do hash, e todos da equipe estiveram presentes na criação da classe do registro e na parte de containerização.

5. Qual o papel de cada função

- **Parser**

- **copyToString**

Recebe uma string lida pelo parser (que pode estar com ou sem falhas de formatação) e copia seu conteúdo para um ponteiro char que representa um atributo.

- **setRecord**

Recebe um vetor de strings que são os atributos que compõem um registro. Transforma cada elemento do vetor no seu respectivo tipo do atributo (int, string, etc) e depois gera um registro com cada um desses atributos.

- **readFile**

Lê e percorre todo o arquivo linha por linha, separando os atributos e tratando as exceções contidas no arquivo. Ao ler uma linha, coleta os atributos usando como separador `","`.

- **Record**

- **record**

Constrói o registro com base nos atributos recebidos como parâmetro.

- **print**

Mostra o conteúdo de cada um dos atributos.

- **Block**

- **block**

Constrói o bloco com base nos atributos recebidos como parâmetro.

- **addRecord**

Recebe um registro e insere-o no vetor de registros que gera um bloco. Checando assim se existe posição disponível no vetor para a inserção do registro.

- **full**

Checa se o bloco está cheio, ou seja, já possui registros salvos.

- **print**

Mostra o conteúdo do bloco.

- **HashFile**

- **calculateld**

Função hash que calcula o valor da chave hash com base no id do registro

- **initFile**

Gera arquivo hash vazio

- **insertRecord**

Insere o registro no bloco que é inserido no arquivo hash.

6. Como rodar

- **Upload**

Compilar

```
> g++ upload.cpp support/parser.cpp support/hashFile.cpp -o main
```

Rodar

Passar como parâmetro o diretório onde se encontra o arquivo de entrada, coloquei no exemplo o diretório em que o arquivo está salvo no projeto.

```
> ./main ../arquivo.csv
```