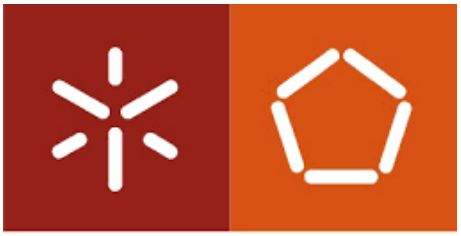
Relatório Projeto C - LI3

Grupo 63:
António Luís de Macedo Fernandes (a93312)
José Diogo Martins Vieira (a93251)
João Silva Torres a93231

May 5, 2021

Universidade do Minho

Contents

1 Introdução

No âmbito do desenvolvimento do projeto da unidade curricular Laboratórios de Informática III foi-nos proposto desenvolver, em lingagem C, um programa que realize a gestão de reviews. Este tem de ler três ficheiro de grandes dimensões: users.cvs, business.cvs e reviews.cvs.

Ora, para que tal seja possível tivemos de introduzir princípios de programação como: a modularidade e encapsulamento; a criação de código reutilizável; a escolha otimizada das estruturas de dados e reutilização; e testes de performance e profiling.

Um dos principais objetivos foi arranjar um processo que permitisse aceder a todos os dados. Optamos pelo o uso das Hash Tables. Estas associam a cada key uma determinada informação de um referente de dados. Com efeito, basta passar o id que nos dá logo a informação correspondente o que nos permite aceder a todos os dados de forma simples, rápida e eficaz.

Em seguida, abordaremos de forma sintetizada os diferentes módulos do nosso trabalho.

Por fim, na última secção deste relatório ilustraremos exemplos de execução das diferentes queries do nosso trabalho e seu respetivo tempo de execução.

2 Módulos do Trabalho

2.1 user.c

Neste módulo encontram-se definidas as funções relativas ao tratamento de dados do tipo users (utilizadores).

Inicialmente, começamos por definir a struct user que é contida pelo user_id, pelo nome, por um array de apontadores dos ids dos amigos e o respetivo número total dos amigos. Para além disso encontram-se definidos os construtores , funçoes de acesso de variáveis do tipo user (getters) e a função strtoUser que transforma uma string (linha do ficheiro) num user.

2.2 business.c

Neste módulo encontram-se definidas as funções relativas ao tratamento de dados do tipo business (negócios).

A struct business ficou definida por várias strings (char *) referentes ao id do negócio, nome do negócio, cidade e estado. Depois utilizamos um array de pointers para as diferentes categorias e uma variavél que armazena o número total de categorias.

Para além disso encontram-se definidas funções de comparação entre dois apontadores de categorias, função de impressão , funções de alteração de variáveis do tipo business, os habituais getters e a função que converte uma linha do ficheiro num negócio.

2.3 review.c

Neste módulo encontram-se definidas as funções relativas ao tratamento de dados do tipo reviews.

A struct review ficou definida por várias strings (char *) referentes ao id da review, id do negócio, id do user, a data da review e o corpo do texto da mesma. Também definimos um float para o número de estrelas, e 3 inteiros que guardam o quão engraçada, util e fixe a review foi.

Para além disso encontram-se definidos os construtores , funçoes de acesso de variáveis do tipo user (getters) e a função strToReview que transforma uma string (linha do ficheiro) numa review.

2.4 UserCatalog.c

Este módulo é responsável pelo armazenamento dos users.

Para tal proposito, utilizamos uma hashtable em que fazemos associar a cada key (user_id) a informação respetiva do utilizador.

Como habitual, também foram definidas várias funções com o objetivo de manipular este tipo de catalogo.

2.5 ReviewCatalog.c

Este módulo é responsável pelo armazenamento das reviews.

Para tal proposito, utilizamos uma hashtable em que fazemos associar a cada key (review_id) a informação respetiva da sua review.

Como habitual, também foram definidas várias funções com o objetivo de manipular este tipo de catalogo.

2.6 BusinessCatalog.c

Este módulo é responsável pelo armazenamento dos negócios.

Para tal proposito, utilizamos uma hashtable em que fizemos a cada key (business_id) a respetiva informação do negócio.

Como habitual, também foram definidas várias funções com o objetivo de manipular este tipo de catalogo.

Para além disso definimos uma função que devolve um array de apontadores com os negócios começados por uma certa letra que será depois utilizada na query 2.

2.7 table.c

Neste módulo definimos o tipo TABLE que é returnado nas diferentes queries e depois analisado pelo interpretador. A struct TABLE ficou definida por um array de apontadores , uma variável que armazena o tamanho da table e ainda uma variavel que armazena o número de colunas. Este array será depois preenchido conforme o tipo de dados que queremos retornar nas várias querys.

Para além disso, definimos funções de inicialização, de acesso á Table (getters) e de inserção.

2.8 stats.c

Este módulo é responsável por fazer a conexão entre os vários tipos de dados. Para tal efeito definimos 4 Hashtables:

Na GHashTable UserStats fazemos corresponder a cada user_id o tipo userStat que é definido o número de reviews e uma hashtable com os id dos negócios.

Na GHashTable BusStats usámos como key o bussines_id ao qual fazemos corresponder a struct busStat que apresenta o numero total de estrelas , o numero de reviews e uma hashtable com os id das reviews.

Na GHashTable CityStats as keys são as várias cidades e fazemos associar uma hashtable com o id dos negocios

Por fim, na GHashTable CategoryStats as keys são as várias categorias e fazemos associar uma hashtable com o id dos negócios.

Para cada uma destas structs, definimos os usuais metodos de acesso(getters), construtores, inicialização e inserção de diferentes dados.

Todas estas HashTables foram definidas de modo a facilitar a busca entre os diferentes tipos de dados que nos é solicitado nas querys. As restantes funções deste modulo são usadas para esse fim.

2.9 sgr.c

Neste módulo começamos por definir a struct sgr onde armazenamos os catalogos dos diferentes tipos de dados (users, reviews, business) com o respetivo "path" para o ficheiro correspondente. seguida dos usuais metodos de inicialização e os getters. Para além disso, definimos funçoes de leitura dos respetivos ficheiros e invocamos funções dos modulos falados ateriormente.

Posteriormente neste ficheiro encontram-se definidas as varias queries que serão devidamente abordadas nos sub tópicos seguintes.

2.9.1 query 1

Nesta query realiza-se o load dos diferentes ficheiros. Recebe como argumento os respetivos caminhos para os ficheiros.

2.9.2 query 2

Nesta query a Table que vamos retornar vai apresentar uma coluna com os nomes do negócios começados por uma certa letra mais o total desses negócios. Para tal fim, definimos uma função auxiliar (startedbyLetter) encontrada no businesscatalog.

2.9.3 query 3

Nesta query a Table que vamos retornar vai apresentar a informação dos negócio (nome, cidade, estado, stars, número total de Reviews) conforme o id lido

Para tal fim utilizamos as funções auxiliares definidas no modulo stats.c, nomeadamente a getMediaStars e getTotalReviews para calcularmos respetivamente o número médio de estrelas e o total de reviews.

2.9.4 query 4

Nesta query a Table que vamos retornar vai apresentar um coluna para o id e outra para o nome dos negócios aos quais o utilizador já fez review.

Para esse fim, utilizamos a função userReviews definida no módulo stats.

2.9.5 query 5

Nesta query a Table que vamos retornar vai apresentar um coluna para o id e outra para o nome dos negócios com mais ou iguais estrelas dadas numa dada cidade.

Para esse fim, utilizamos o CityStats para irmos buscar todos os negócios da cidade dada. De seguida, comparámos o número de estrelas de cada negócio com o número de estrelas recebido como argumento.

2.9.6 query 6

Nesta query a Table que vamos retornar vai apresentar 3 colunas com os top n negócios (tendo em conta o numero medio de estrelas) em cada cidade, uma para o id outra para o nome e ainda as estrelas desse negócio.

Para este fim, percorremos todas as cidades do CityStats, organizamos por número de estrelas e recolhemos os top n negócios.

2.9.7 query 7

Nesta query a Table que vamos retornar vai apresentar a lista dos ids dos utilizadores e o número total de utilizadores que tenham feito reviews de negócios em diferentes estados.

Para esse fim, percorremos todos os do UserStats e verificamos se realizou negócios em mais do que um estado.

2.9.8 query 8

Nesta query a Table que vamos retornar vai apresentar três colunas uma para o id outra para o nome e a terceira para o número de estrelas dos top n negócios que pertecem a uma determinada categoria.

Para este fim, utilizamos o CategoryStats para irmos buscar todos os negócios da categoria dada, organizamos por número de estrelas e recolhemos os top n negócios.

2.9.9 query 9

Nesta query a Table que vamos retornar vai apresentar uma coluna com os ids das reviews com a dada palavra.

Para este fim, percorremos o ReviewCatalog e verificamos se a palavra se encontra no campo text das diferentes review.

2.10 view.c

Este módulo é referente a parte mais estética do trabalho e da apresentação do programa. Sendo assim definidas várias funções de impressão no ecrãn e funções auxiliares que representam o display das diferentes querys.

2.11 Interpretador

Neste módulo encontra-se o nosso interpretador e este é responsável pela interação do utilizador com o nosso programa.

Encontram-se também as invocações das diferentes queries e a possibilidade de executar os diferentes comandos pedidos (Com exceção do comando fromCSV e da indexação).

3 Resultados

Nesta secção encontram-se os diferentes tempos de execução das queries do nosso programa. Usamos como input files os seguintes files:

-business_full.csv

-users_full.csv

-reviews_1M.csv

Query	Tempo de Execução	Inputs
1	9.337631	-
2	0.042924	Letra = "a"
3	0.000012	User Id = hrvQyPfec2YqNk9ZknlYig
4	0.000029	User Id = NaYKSt-NJnNfQJkK6milfA
5	0.006485	Cidade = Atlanta e Estrelas = 3
6	0.254212	Numero de negócios = 50
7	0.694328	-
8	0.133509	Categoria = Restaurants e Numero de negócios = 50
9	0.915961	Palavra = "fun"

Nota : Estes testes foram realizados em ambiente Linux, 8 GB RAM, Ryzen 7 3700U (Processador) e apresentámos o resultado mais satisfatório de uma série de 6 testes.

4 Conclusão

Nesta fase final do projeto constatamos que, apesar de termos cumprido o que nos foi pedido, não gerimos o tempo da melhor forma.

Dito isto, dedicamos demasiado tempo no desenvolvimento da apresentação. Temos um resultado final bastante apelativo do ponto de vista estético, mas concluimos que podiamos ter dedicado parte deste tempo para uma melhor otimização do interpretador.

Fomos capazes de implementar todas as funcionalidades pedidas à exceção de dois comandos do interpretador - csv e indexação.

Por outro lado, obtivemos tempos de execução bastante satisfatórios, o que nos leva a crer que, de uma forma geral, tivemos um bom aproveitamento.

Concluindo, este trabalho ajudou-nos a consolidar e desenvolver novas aptidões em programação, nomeadamente em linguagem C.