Universidade do Minho
Licenciatura em Engenharia Informática

Fase 1

Laboratórios de informática III 2022/23

Grupo 1:

Afonso Martins (A94881)
André Alves (a95033)
Gonçalo Freitas (A96136)



22 de novembro de 2022

Conteúdo

Lista de Figuras

1 Introdução

Ao longo deste relatório é abordado o desenvolvimento do trabalho, centrando a atenção na resolução das etapas propostas pelos docentes nesta primeira fase assim como nas estratégias e métodos utilizados.

2 Desenvolvimento do trabalho

Nesta primeria fase do trabalho de Laboratórios de Informática III deparamos agora com a necessidade de criar uma arquitetura para uma API. Onde nos era aconselhado a separação em 'Leitura' e 'Módulos de Dados', como podemos observar na Figura 2.1 (Figura abaixo).

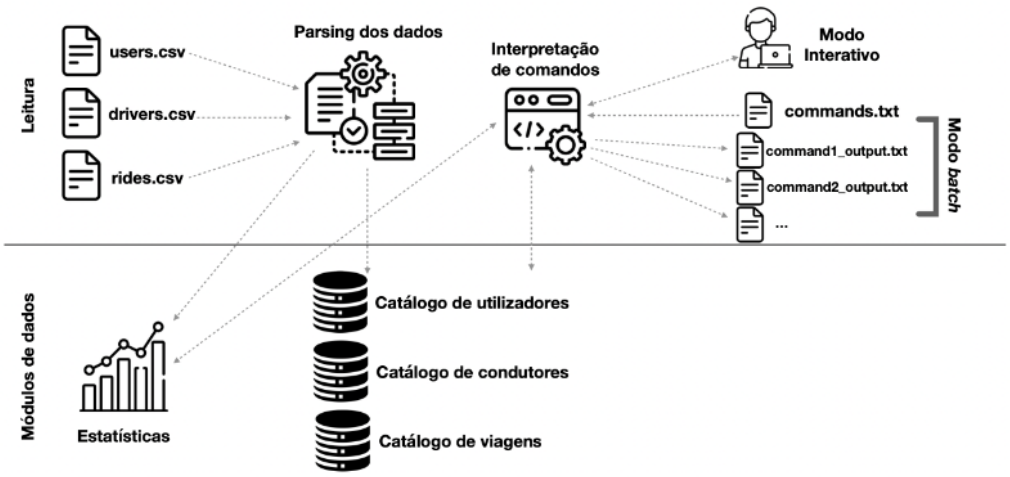

Figura 1: Arquitetura de referência

Após a leitura dos ficheiros '*.csv' é necessário guardar os dados importantes para a realização das queries. Após a leitura necessitamos de armazenar os dados, na qual situação optamos por utilizar vários tipos de hashtable, como explicado.Para alguns valores necessários às queries, iniciaremos a sua realização durante a execução da criação dos catálogos, tais que serão mencionados posteriormente, pois desta forma conseguimos otimizar o programa fazendo com que não seja necessária a travessia dos dados na sua totalidade múltiplas vezes. Finalizando assim com a interpretação do ficheiro de comandos que resulta no output de vários ficheiros consoante o número de queries solicitadas, consoante o input.

2.1 Hashtable vs Outros métodos

Após o teste de outras estruturas de dados como árvores binárias e listas ligadas, o tempo de inserção pós ordem teria um custo muito mais elevado do que numa hashtable. Como o número de input de comandos não é assim tão elevado, não compensa a criação de uma árvore binária devido à sua organização demorada.

2.2 Encapsulamento e Modularidade

O encapsulamento e a modularidade apenas nos permite aceder aos módulos, 'user.c', 'users.c', 'driver.c', 'drivers.c', 'ride.c', 'rides.c', 'hashtable.c', 'queries.c' e 'parser.c' através da sua API, promovendo o encapsulamento.

2.3 Catálogos

Optando assim pela utilização como armazenamento de dados de uma hashtable, criamos hashtables para os ficheiro dos 'users', 'drivers' e 'rides'. A criação destes por este método torna o seu tempo de execução linear.

Aqui, cada linha do ficheiro é diferenciada pelo seu respetivo id e é guardado na tabela consoante a função hash da key(id). Ficheiros em que as keys são diferentes e as hash tem o mesmo valor, é solucionado como se fosse uma espécie de lista ligada inserida na hash.

3 Queries

As queries são comandos executados através de uma constante, id da querie, e algumas por uma constante e certos argumentos, tais argumentos variando de query para query.Nestas queries existem parâmetros que são facilmente calculados na criação dos catálogos respetivos a cada ficheiro, logo como mencionado atrás estes valores serão logo criados juntamente com os catálogos globais, por exemplo as tabelas $\mathrm{ht}_u ser_r ideeht_d river_r ide, quesousadasparaaprimeiraquery, sogerad$

3.1 Query 1

Nesta query nós consoante o id que recebermos, caso seja um número é um driver e caso seja uma string é um user, iremos retornar das seguintes formas respetivamente:

1. User: nome;genero;idade;avaliacao_media;numero_viagens;total_gasto
2. Driver: nome;genero;idade;avaliacao_media;numero_viagens;total_auferido

Os parâmetros acima serão, para além dos obviamente detetáveis, a avalicao media que o user dá ou que o driver recebe consoante todas as viagens que fizeram, o número de viagens que o user ou que o driver realizaram e o total que o user gastou ou que o driver recebeu por todas as suas viagens. Caso um user ou driver esteja ou inativo ou até que não exista nos nossos catálogos a query não irá retornar nada.

Para a realização desta query ao invés de percorrermos o catálogo necessário uma vez para cada parâmetro,sendo que isso ia ter um custo demasiado elevado, ao realizarmos o cálculo do parâmetro da avaliacao_media cálculamos juntamente o número de viagens que ele realizou e o total_gasto ou total_auferido, dependendo do id. Desta forma ao invés de percorrermos 3 vezes a tabela percorremos a percorremos uma vez.

Tempo de execução – N - Linear.

3.2 Query 2

Nesta query, consoante o n que recebermos, devolvemos os n condutores com maior avalição média. Em caso de empate, o resultado deverá ser ordenado de forma a que os condutores com a viagem mais recente surjam primeiro. Caso haja novo empate, é usado o id do condutor, por ordem crescente, para desempatar. Retorna os dados no seguinte formato:

1. id;nome;avaliacao_media

Os parâmetros acima serão, o id do *driver*, o nome do *driver* e a avaliação média que o driver recebe consoante todas as viagens que fez.

Para a realização desta query percorremos o catálogo dos drivers e, de cada driver, com o auxílio da função $\mathbf{get}_a valiacao_m edia_d river, obtamosovalordaavaliaomdiadocondutoremquesto.$

3.3 Query 3

4 Conclusão

Achamos que este foi o melhor método para a resolução das queries pois o custo de execução de cada query, provinha do esforço necessário para a criação das nossas estruturas de dados (hashtables) que era relativamente reduzido comparando com o esforço necessário caso fosse preciso usar algoritmos de ordenação, ou árvores binárias.

Estatisticamente o tempo de criação de 3 tabelas com 1 milhões de linhas era entre 12 a 15 segundos mais 1 a 2 segundos para as libertar. Isto num computador razoável. A seguir dizemos algumas vantagens e desvantagens deste trabalho.

1. **Desvantagens – Maior tempo de execução caso o número de comandos seja elevado.**
2. **Vantagens – Menor risco de o programa "arrebentar" por falta de memória, caso o número de linhas dos ficheiros seja muito, mas muito elevado; Tempo de execução normal caso os comandos sejam todos diferentes.Relativamente a problemas de memoria libertamos a memória alocada não necessária sempre que possível.**

No geral, consideramos que este trabalho foi predominantemente criado pelo elemento André Alves onde teve alguma ajuda do elemento Gonçalo Freitas, tendo o último elemento nunca aparecido a reuniões de grupo e nunca demonstrou interesse na criação do trabalho. Achamos bastante simples a implementação e bastante interessante e útil a inserção dos módulos e das cápsulas, como mencionado anteriormente.