Relatório da Fase 1 – Grupo 25

Hugo António Gomes Ramos (A100644)
Martim Quintelas Pinto Félix (A100647)
João Silva Loureiro (A100832)

18 de Novembro de 2023

1. Introdução

O presente relatório tem como objetivo abordar a primeira fase do projeto da UC Laboratórios de Informática III do ano letivo 2023/2024. Nesta fase, como objetivos o trabalho necessitava de fazer o parsing dos dados de entrada, funcionar através do modo de operação *batch,* realizar 6 das 10 queries propostas e validar o dataset fornecido pelos docentes. Para além disto, temos de ter em conta, durante todo o projeto, o uso de encapsulamento e modularidade.
Este relatório irá abranger as decisões tomadas pelo grupo, como os métodos de raciocínio para o desenvolvimento do mesmo.
Dito isso, é importante destacar que nas estruturas que exemplificaremos posteriormente, consideramos a natureza dos dados envolvidos e as questões que precisavam de ser resolvidas dentro de um tempo útil. Portanto, se considerássemos outros fatores, a abordagem adotada certamente seria diferente.

2. Catálogos

Primeiramente, a fase inicial do projeto aborda o armazenamento dos dados, ou seja, a necessidade de preservar todas as informações relevantes numa estrutura específica. Como reparamos que existem quatro entidades fundamentais (*Users*, *Flights*, *Reservations*, *Passengers*), optamos por criar uma estrutura de dados para cada uma dessas entidades.
(Podemos mencionar a alteração que o João fez de meter nas structs dos Users todas as informações importantes para as queries que não são necessariamente deles.)

No desenvolvimento do nosso projeto, como é uma possibilidade utilizar a biblioteca *GLIB* adotamos uma estratégia consistente ao utilizar HashTables para as quatro entidades fundamentais. A implementação das HashTables foram, a nosso ver, uma das melhores formas de realizar o projeto já que contribuem para a eficiência e facilidade de gestão dos dados.

Vantagens de utilizar HashTables:

- Acesso Rápido por *Key*:
As HashTables proporcionam acesso direto aos dados através de uma chave única associada a cada entidade. Este método de acesso constante é especialmente valioso ao lidar com grandes conjuntos de dados como é o caso, pois não depende do tamanho total do ficheiro.

- Eficiência em Termos de Tempo:
As operações de inserção, pesquisa e remoção em uma HashTables têm uma complexidade média de tempo O(1) (constante) para operações bem-sucedidas. Isso resulta em tempos de resposta rápidos, essenciais para otimizar o desempenho em cenários onde a velocidade de acesso aos dados é crucial.

- Identificação Única por Chave (ID):
A natureza única das chaves nas HashTables corresponde perfeitamente à necessidade de identificadores distintos (ID) para cada entidade (*Users*, *Flights*, *Reservations*, *Passengers*). Isto garante a unicidade em todo o sistema.

- Facilidade de Implementação:

Apesar de as HashTables já serem uma estrutura simples, o facto de podermos usar a biblioteca GLIB facilita ainda mais o processo já que nos fornece uma implementação eficiente de tabelas Hash através da estrutura de dados das HashTables. A estrutura das HashTables facilita a implementação e gestão do código. A utilização de funções Hash permite a distribuição eficiente dos dados, mantendo ao mesmo tempo a acessibilidade.

Desvantagens de utilizar HashTables:

- Colisões:

Uma desvantagem potencial das HashTables é a possibilidade de colisões, onde duas chaves diferentes são mapeadas para a mesma posição na tabela. Existem métodos de resolução de colisões, mas a escolha do método certo é crucial para evitar degradações no desempenho.

- Consumo de Memória:

HashTables podem consumir mais memória em comparação com estruturas de dados lineares, principalmente devido à necessidade de manter uma tabela Hash adicional.

- Dificuldade na Ordenação:

As HashTables não mantêm uma ordem específica nos elementos, o que pode ser uma desvantagem se a ordenação dos dados for uma consideração importante no contexto do projeto. Visto que este é um ponto importante no trabalho tivemos de arranjar uma solução. Solução esta que iremos explicar mais à frente quando chegarmos à parte do relatório referente à resolução das queries.

Em suma, a decisão de utilizar HashTables para todas as entidades foi baseada na busca por um equilíbrio entre eficiência operacional e facilidade de implementação. Ao reconhecer as vantagens e desvantagens associadas às HashTables, conseguimos criar uma estrutura de dados coesa para atender às necessidades específicas do nosso projeto, onde a identificação única por *Key* desempenha um papel central.

3. Queries

Para a primeira fase do trabalho decidimos implementar as bases do projeto, como o parser e definir as estruturas dos dados que utilizaremos ao longo do projeto (Como o catálogo para guardar os dados e os *Users*, *Flights*, *Passengers* e *Reservations*). Foram implementadas as queries 1, 2, 3, 4, 5, 6. (EXEMPLOS)

Query 1
(Texto)

Query 2
(Texto)

Query 3
(Texto)

Query 4
(Texto)

Query 5

(Texto)

Query 6
(Texto)

4. Análise de Desempenho

5. Possíveis otimizações e futuras limitações

6. Conclusão

Para concluir, acreditamos que ,apesar de tudo, o desenvolvimento da primeira fase do projeto de Laboratórios de Informática III foi muito satisfatória. Consolidamos os nossos conhecimentos relativamente à linguagem C e aprimoramos conhecimentos adquirimos nos anos anteriores.
O uso de HashTable foi o mais complicado, considerando a inexperiência no que toca ao uso desta estrutura em específico, no entanto ficamos agradados com o trabalho elaborado.
Como em tudo, ainda existe espaço para melhorias e pretendemos desenvolvê-las para a próxima fase do trabalho.

TUDO O QUE ESTÁ SUBLINHADO, É PORQUE PRECISA DE REVISÕES OU ALTERAÇÕES!