Universidade do Minho

# Laboratórios de Informática III

Trabalho Prático

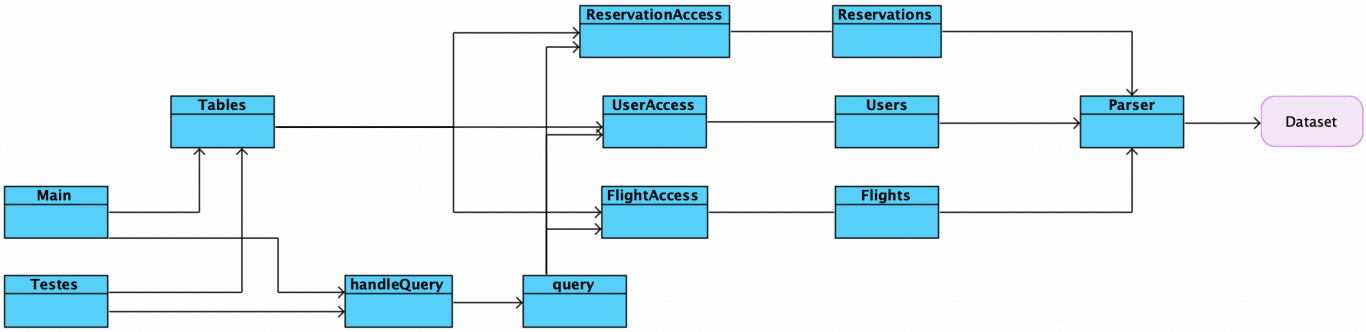
Grupo 53

Luiz Rodrigues A100700
Michelle de Morais A98576
Carolina Pereira A100836

# Índice

# 1. Introdução.

Este relatório tem como objetivo demonstrar a versão final do projeto, comparando as diferenças entre a versão anterior e a atual, os objetivos que foram atingidos, a performance do programa, as estratégias utilizadas para otimização e as melhorias que podem ser feitas.




Representação gráfica do programa

# 2. Alterações.

A maior diferença que tivemos foi a remoção das estruturas de Passengers, foram removidos por não serem necessários, as informações presentes nos ficheiros são atribuídas diretamente aos utilizadores e aos voos.

Também foram alteradas as estruturas utilizadas para armazenar as informações das queries, que antes eram em sua maioria listas ligadas e passaram para ser árvores binárias, uma vez que a sua ordenação é mais rápida.

Tendo a modularidade em mente foi criada uma função de print geral em que o resultado de uma query é passado para uma string e depois enviado para o módulo que trata dos prints.

## 3. Estatísticas.

O módulo de estatísticas foi adicionado para armazenar informações relevantes para as queries, no momento a única que foi necessária para um funcionamento mais rápido foi a query 7, em que na sua primeira execução demora por volta dos 6 segundos para o dataset grande porém para execuções subsequentes as suas execução já está nos campos do milisegundos .

Para as queries em que era aplicável foram criadas tabelas que guardam os voos associados a um aeroporto (origem) e as reservas associadas a um hotel específico, isso permite que não seja necessário percorrer todas as milhares de entradas para a execução.

## 4. Módulos novos

O módulo Interativo foi criado, esse que permite o utilizador escrever as queries que quer executar e receber a resposta, como foi recomendado caso o utilizador não insira um caminho para o dataset utilizamos um como padrão (pequeno).

O módulo de testes foi implementado para comparar os resultados das queries, e permite o utilizador ver os resultados de duas formas diferentes, pode escrever os valores em ficheiro para uma melhor percepção de cada resultado.

Foram criados módulos de acesso para users, flights e reservations, sendo estes os únicos que podem obter as informações de cada tabela. Foi mantido a estrutura Tables para conseguirmos passar as referências destes módulos de controle de uma forma mais simples, para evitar ter de passar cada módulo que trata disso.

Seguem alguns exemplos da execução no modo interativo:

```
Insert path to dataset:
dataset/data
```

Dataset indicado pelo utilizador

```
Insert query number and arguments:
10
```

Query e argumentos indicados pelo utilizador

```
page 1 of 2
2010;905;0;0;0;0
2011;857;0;0;0;0
2012;878;0;0;0;0
2013;833;0;0;0;0
2014;851;0;0;0;0
2015;859;0;0;0;0
2016;864;0;0;0;0
2017;843;0;0;0;0
2018;852;0;0;0;0
2019;839;0;0;0;0

Press Enter to continue
```

Resultado de uma query

```
page 2 of 2
2020;883;0;0;0;0
2021;0;350;26642;8887;13982
2022;0;355;27510;8957;14472
2023;0;245;19175;8266;9904

Press Enter to continue
```

Segunda página do resultado

```
Insert query number and arguments:
10F
```

Query com formatação

```
page 1 of 14
--- 1 ---
year: 2010
users: 905
flights: 0
passengers: 0
unique_passengers: 0
reservations: 0

Press Enter to continue
```

Resultado formatado de uma query

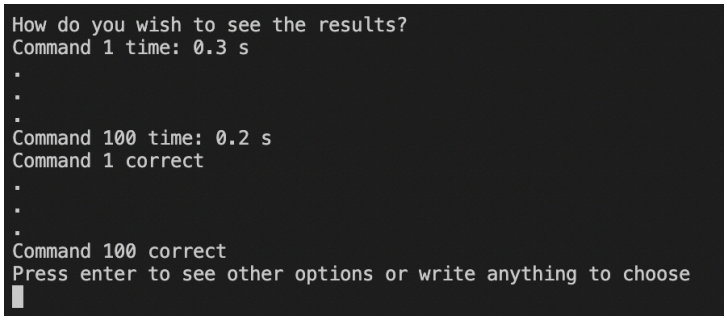
```
page 14 of 14
--- 14 ---
year: 2023
users: 0
flights: 245
passengers: 19175
unique_passengers: 8266
reservations: 9904

Press Enter to continue
```

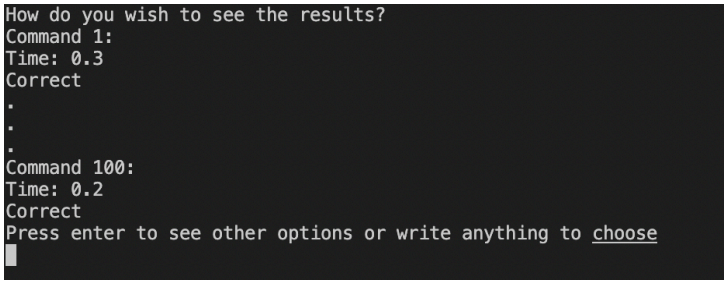
Última página do resultado

# 5. Testes de funcionalidade.

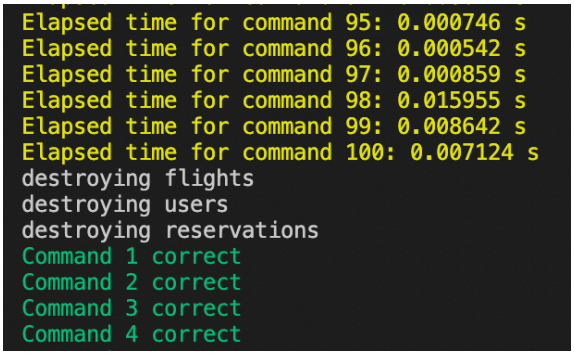
Ao utilizar o programa de testes é possível ver se os resultados estão corretos e também ver os tempos de execução e a memória utilizada.



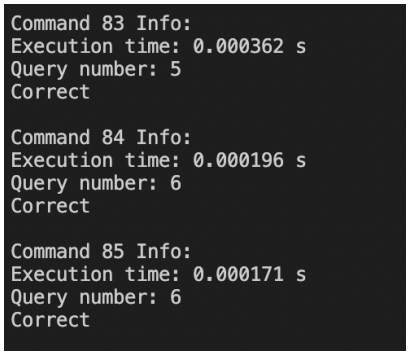

Opção 1 dos testes




Opção 2 dos testes




Resultados opção 1

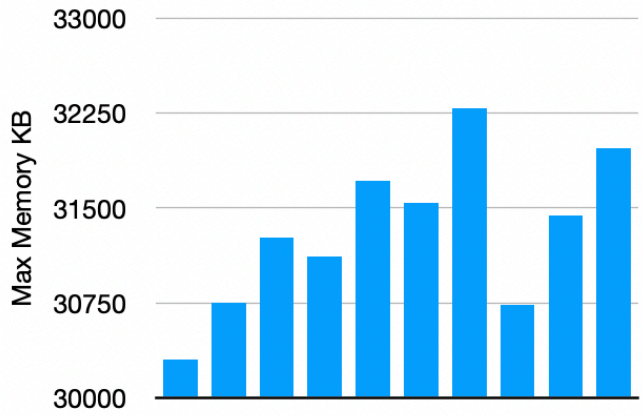


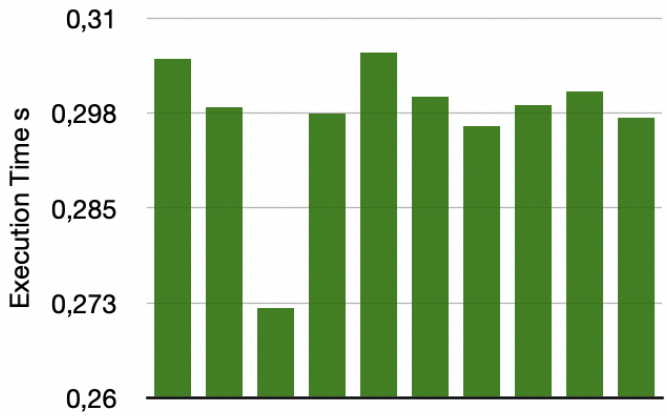

Resultados opção 2

Com a execução do programa de testes conseguimos fazer uma análise mais profunda da performance do programa. Para isso vamos fazer 10 execuções para cada dataset iniciando nas máquinas que temos disponíveis.

1 º teste MacBook air m1:

- Dataset pequeno



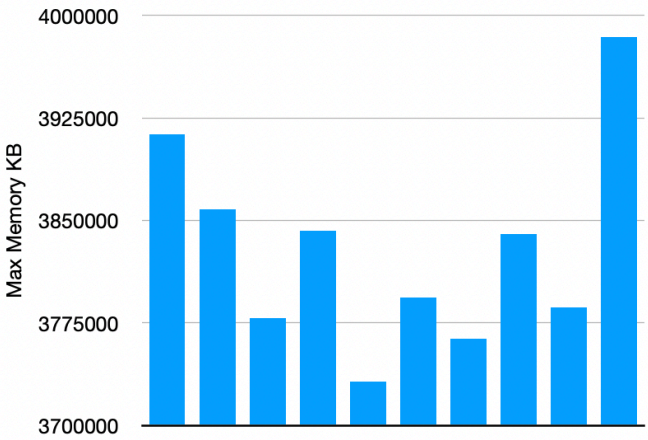



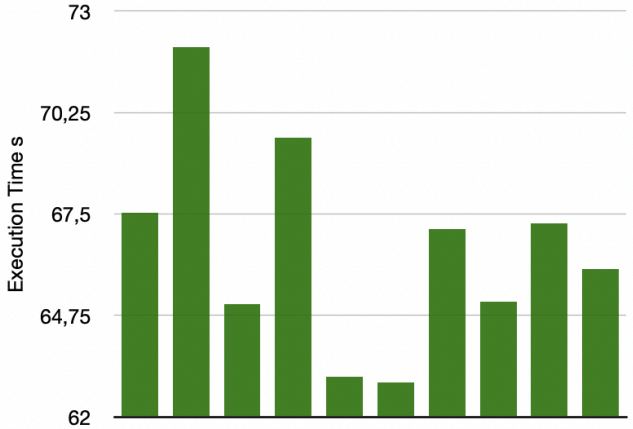

Pelos gráficos é possível ver que o tempo de execução é extremamente consistente em comparação com a memória máxima utilizada, com um caso em que a execução correu um pouco mais rápido. Com essas execuções também conseguimos obter algumas estatísticas sobre a execução:

Em relação ao tempo de execução temos: média = 0.296855 s; mediana = 0.29839 s; mínimo = 0.27183 s; máximo = 0.305457 s; desvio padrão = 0.09.

Em relação à memória máxima utilizada temos: média = 31312 KB; mediana = 31352 KB; mínimo = 30304; máximo = 32288 KB; desvio padrão = 606.454.

- Dataset grande



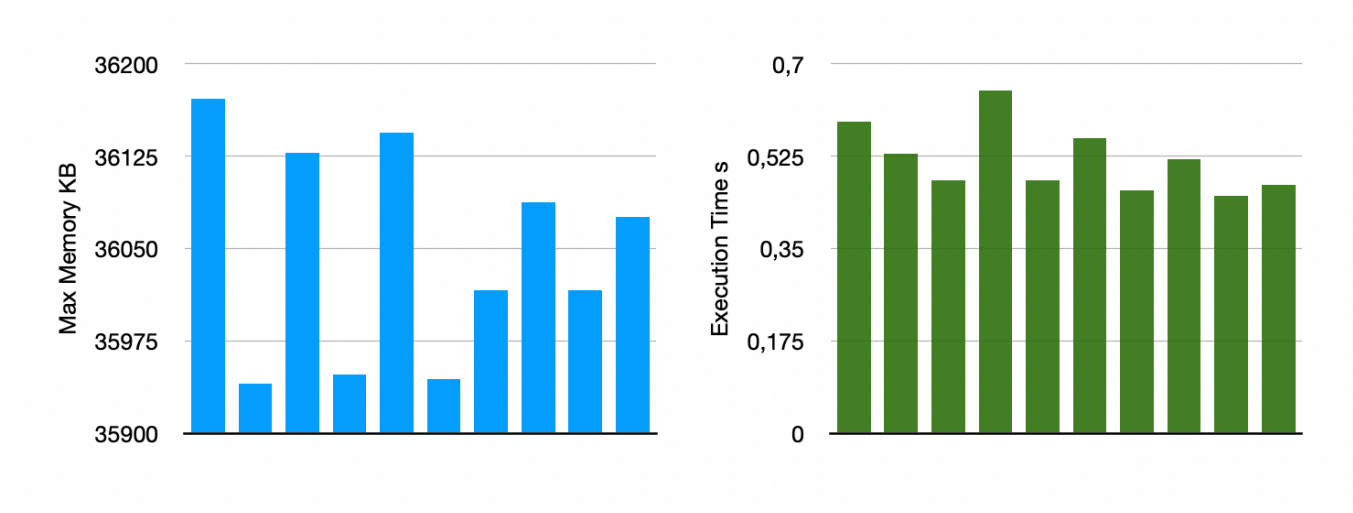
Com o dataset grande é possível ver que há muito mais variação no tempo e na memória, isso indica que há áreas no código em que pequenas mudanças podem desencadear grandes alterações no resultado, logo é algo que deve ser observado com mais cuidado.

Em relação ao tempo de execução temos: média = 66.568878 s; mediana = 66.548365 s; mínimo = 62.930699 s; máximo = 72.019125 s; desvio padrão = 2.796.

Em relação à memória máxima utilizada temos: média = 3829282 KB; mediana = 3816864 KB; mínimo = 3732016; máximo = 3984352 KB; desvio padrão = 75751.942.

Ao executarmos o programa nas nossas máquinas não vimos grandes diferenças por isso decidimos comparar com os valores presentes no site de testes automáticos. Porém não temos certeza das máquinas que o executam,

- Dataset pequeno

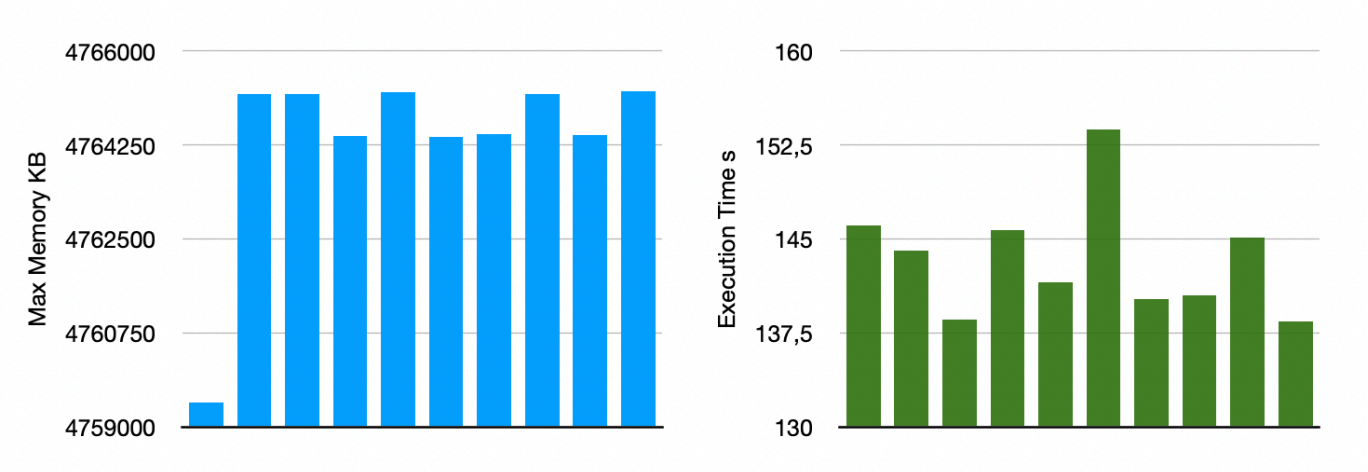



Pelos gráficos é possível observar que os tempos são muito mais consistentes que a memória máxima.

Em relação ao tempo de execução temos: média = 0.519 s; mediana = 0.5 s; mínimo = 0.45 s; máximo = 0.65 s; desvio padrão = 0.064.

Em relação à memória máxima utilizada temos: média = 36047,2 KB; mediana = 36046 KB; mínimo = 35940 KB; máximo 36172 KB; desvio padrão = 86.957.

- Dataset grande




Nesse caso a memória é muito mais consistente com apenas um caso que pode ser considerado um outlier, o tempo de execução também é bem consistente com um exemplo em que demorou mais.

Em relação ao tempo de execução temos: média 143.41 s; mediana = 142.815 s; mínimo = 138.4 s; máximo = 153.77 s; desvio padrão = 4.636.

Em relação à memória máxima utilizada temos: média = 4764324.4 KB; mediana = 4764824 KB; mínimo = 4759452 KB; máximo = 4765256 KB; desvio padrão = 1756.323.

O que podemos concluir destes testes é que nas nossas máquinas é que a execução do dataset grande foi muito mais rápida, e utilizou aproximadamente 1 GB a menos de memória, isto pode ser dado por vários motivos, entre eles temos as representações dos valores, tamanho dos inteiros e de doubles, também deve-se a otimização do próprio sistema, em

que pode ser armazenado de uma melhor forma. Já os tempos de execução sendo consistentes nas nossas máquinas pode vir do facto de termos corrido o programa sequencialmente, o que permite que os valores armazenados na cache sejam acessados novamente antes do valor ser alterado. A consistência do site de testes pode ser dado pela forma que é executada, principalmente se houver restrições na execução, o que faz com que o programa corra sempre de uma forma parecida.

## 6. Conclusão e trabalho futuro.

Devido ao limite de memória que nos foi imposto, não foi possível criar estatísticas para ajudar em algumas queries, o que poderíamos fazer é não guardar em memória informações desnecessárias, como as notas de um voo, o número de telefone de um utilizador ou os comentários e detalhes de uma reserva. estas informações poderiam ser guardadas em ficheiros ou, numa implementação mais avançada, numa base de dados, mas isso deveria ser estudado mais à fundo caso fosse necessário criar mais queries em que essas informações fossem necessárias, isso diminuiria a memória mínima necessária e permitiria novas estruturas.

Também é possível alterar os campos das estruturas para utilizar variáveis que ocupem menos espaço, por exemplo usar short ao invés de int onde é aplicável.

Também poderia ser criado uma estrutura para armazenar as datas, isso não foi feito devido à complexidade que o projeto já estava antes de pensarmos em implementar isso, logo se tivéssemos que iniciar novamente o projeto faríamos uma estrutura para guardar as datas.