Laboratórios de Informática III

Frederico Cunha Afonso
a104001
LEI

Índice

Introdução

Este relatório objetiva apresentar uma análise detalhada sobre o projeto desenvolvido no âmbito da disciplina **Laboratórios de Informática III**.

Arquitetura

Antes de começar o desenvolvimento do programa, houve um pequeno período de tempo em que se discutiu as diversas estratégias que se podiam aplicar para ter o melhor rendimento possível em termos de tempo e memória gasta. Houve vários aspetos a ter em conta na implementação de certas estruturas de dados no projeto, tendo-se separado as estruturas de dados em dois tipos:

- Estruturas de dados que visavam o armazenamento de informação diretamente relacionada com a informação de utilizadores, voos, reservas e passageiros;
- Estruturas de dados genéricas que tinham como objetivo armazenar informação genérica (void *) em grandes quantidades (auxiliares);

Estruturas de Dados

Flight

Quando se trata de voos, é usada a estrutura "Flight", que contêm strings relativas ao seu identificador do voo, companhia aérea, modelo do avião, aeroporto de origem, aeroporto de destino, data e hora estimada e real de partida e data e hora estimada de chegada, tudo informação obtida do ficheiro "flights.csv", deixando a única informação que não se obtém diretamente desse ficheiro o número de passageiros válidos, sendo que esta é a única variável daqui que só é obtida através da leitura de um outro ficheiro.

Para determinar o número de passageiros válidos é necessário acessar ao ficheiro "passengers.csv" e assim contar todos os passageiros (que são utilizadores válidos) associados ao identificador do voo.

User

A estrutura "User" trata de guardar toda a informação relevante de um utilizador. Assim, esta contém o identificador, o nome, a data de nascimento, o género, o passaporte, o código do país de residência, a data de criação e estado da conta de um dado utilizador. Toda esta informação é obtida diretamente do ficheiro .csv relativo a utilizadores (users.csv).

É de notar que, de maneira a gastar o mínimo de memória possível, passei de usar uma string para guardar o género e estado de conta de um utilizador para usar um short, sendo que ambos só podem assumir dois estados (género feminino ou masculino e estado de conta ativo ou inativo);

Reservation

Como última estrutura *não auxiliar*, a "Reservation" contém o identificador da reserva, identificador do utilizador, identificador do hotel, nome do hotel, número de estrelas do hotel, percentagem do imposto da cidade (sobre o valor total), data de início e fim de estadia, preço por noite, inclusão de pequeno-almoço e classificação atribuída pelo utilizador de uma certa reserva. Tal como com a "User", toda a informação contida desta estrutura é obtida ao acessar o ficheiro que lhe é relativo (reservations.csv)

Também como com a "User", em vez de guardar toda a informação como strings, transformaram-se certos dados de string para outros tipos de forma a reduzir a quantidade de memória necessária para guardar os ditos dados.

Como um **char *** requer 8 bytes de memória, houve uma reestruturação desta estrutura de maneira a usar o mínimo número de **char *** possíveis.

A classificação de uma reserva e o número de estrelas só pode variar entre 1 e 5 (ou, no caso da classificação, pode incluir 0 caso seja nula) , por isso passei a usar **char** (1 byte).

A percentagem do imposto da cidade e o preço por noite de uma reserva tem que ser um inteiro superior a 0, por isso (esperando que não seja maior que 10000) passei a usar um **short** (2 bytes).

O identificador de um hotel tem o formato “HTL1102”, logo, em vez de guardar os primeiros caracteres que estão presentes em todos os outros identificadores, apenas guardo o número seguido pelo “HTL” (e.g. de “HTL1102” será apenas guardado “1102”) num **short** (2 bytes)

A inclusão de pequeno-almoço só pode ter dois estados (incluir ou não incluir), passei a usar um **char** (1 byte).

O identificador da reserva tem o formato “BookXXXXXXXXXX” (alterando os X’s por números), por isso seria uma boa ideia guardar apenas o número (**int**) do id em vez da string toda (e.g. de “Book0000000002”, guardar apenas o 2), mas infelizmente isto só seria possível para um quarto de todas as possíveis reservas a serem feitas, pois como um **int** apenas consegue armazenar **2,147,483,647** números positivos e é possível haver **9,999,999,999** reservas (Book9999999999), isto não é possível. Logo, de maneira a não usar um **char ***, passei a guardar os últimos 9 dígitos do identificador como um **int** (4 bytes) e o primeiro dígito como um **char** (1 byte), usando assim apenas 5 bytes (e.g. para o identificador “Book1234567890”, será guardado como primeiro dígito ‘1’ e será guardado como int “234567890”).

Passando assim de usar **88 bytes** por “Reservation” a usar **46 bytes** (o que se torna numa grande diferença quando estivermos a tratar de datasets de maior escala).

Auxiliares

De maneira a responder às queries da forma mais eficiente possível, foram usadas estruturas de dados capazes de receber informação genérica com diferentes níveis de eficiência de inserção e obtenção de dados nos catálogos.

BTree