Universidade Do Minho

Mestrado Integrado em Engenharia Informática

Laboratórios de Informática III

Docentes:

Vitor Fonte
João Marco Silva
João Carlos Pereira

Discentes:

Tiago Baptista 75328
Lucas Pereira 68547

Junho 2018

Índice de Conteúdos

Resumo **2**

Introdução **4**

Classes/Estruturas **5**

Interrogações **6**

Interrogação 1 6

Interrogação 2 6

Interrogação 3 6

Interrogação 4 6

Interrogação 5 6

Interrogação 6 6

Interrogação 7 6

Interrogação 8 7

Interrogação 9 7

Interrogação 10 7

Interrogação 11 7

Conclusão **8**

Resumo

Atendendo ao facto de que esta é a segunda de duas partes que constituíram o projeto relativo à unidade curricular de Laboratórios de Informática III, a familiarização e entendimento do problema base foram bastante mais rápidos do que na primeira fase. Assim sendo, e uma vez que o projeto proposto pelos docentes seria a execução de um programa semelhante ao da primeira fase, mas na linguagem de programação JAVA, algumas bases tiveram de ser desenvolvidas para que as *queries* fossem executadas com sucesso e eficácia.

Como objetivo o trabalho enumera-se o desenvolvimento de um sistema que seja capaz de processar ficheiros do tipo XML que por sua vez armazenam dados provenientes da plataforma *StackOverflow.* Após o processamento, é pretendido que o sistema seja capaz de resolver algumas interrogações à custa da implementação de estruturas de dados auxiliares.

Introdução

Em 2008, os criadores Jeff Atwood e Joel Spolsky criaram o *StackOverflow*. Este site foi criado como uma alternativa mais aberta a sites mais antigos de pergunta e resposta. O nome assim como o logotipo foram obtidos por meio de votação dos participantes do site quando este ainda permanecia fechado. Assim sendo, o intuito é responder a questões dos seus utilizadores sobre programação em vários tipos de linguagens. O site tem uma dinâmica bastante interessante pois os utilizadores obtêm 10 pontos de reputação por cada voto favorável das duas respostas.

Assim sendo, e tal como previamente resumindo, o objetivo do trabalho seria utilizar ficheiros *XML* que contém informação do site anteriormente descrito para assim responder a algumas interrogações do cariz estatístico.

Classes/Estruturas

Para este projeto, foi tomada a decisão de criação das seguintes classes referentes ao package *engine*:

- **Classe User**;
- **Classe Post**;

São referentes aos objetos que são necessários criar para inserção nas estruturas que são utilizadas para armazenamento de informação, com os respetivos e habituais *getters*, *setters*, etc.

- **Classe *Parser***;

Classe para parsing dos ficheiros que são recebidos pelo caminho relativo aos dumps.
Esta classe não tem variáveis privadas e só estão codificados métodos(*static*) para fazer parsing de acordo com o *XML* respetivo. A razão pela qual foi optado criar a sua classe própria de parsing é a questão da modularização do código.
Ainda nesta classe temos uma função auxiliar para alteração para LocalDate.
O método para parsing usado foi o *STAX* visto ser mais eficiente do que a *DOM* em termos de memória porque faz uma única travessia ao ficheiro *XML* e só guarda em memória pedaços do mesmo.

- **Classe *TCDCommunity***;

Classe que serve como base de dados para o projeto e implementação da *TADCommunity,* devidamente encapsulada, com "getters" para ser possível obter referências das estruturas e sem "setters" clássicos para as variáveis privadas(estruturas). Apenas existem funções para adicionar, um a um, objetos do tipo User ou Post à estrutura desejada.
Ainda nesta classe, como foi dito anteriormente, existe a implementação da *TADCommunity*, mas, mais uma vez, devido a questões de modularização, foi decidido criar outra classe funcional(*QueryEngine*) para fazer a respetiva computação das queries.
Como estruturas da nossa "base de dados" optamos por:

->HashMap<Long,User> para os Users;

->HashMap<Long,Post> para os Posts;

->HashMap<LocalDate,TreeMap<Long,Post>> para todas os Posts que são pergunta;

->HashMap<LocalDate,TreeMap<Long,Post>> para todas os Posts que são resposta;

- **Classe *QueryEngine***

Classe que foi criada com o objetivo de modularização da *TCDCommunity*. Esta classe é responsável pela computação dos dados de maneira a responder às queries.

Interrogações

Interrogação 1

Na primeira Interrogação é dado o *id* de um *post* e caso o mesmo seja uma pergunta, é pedido um par constituído pelo título do mesmo post e o nome do seu autor, caso seja uma resposta é pedido como retorno do título e nome do utilizador da pergunta à qual o post responde.

De maneira a resolver este problema começa-se por procurar o *Post* na *HashMap* (geral) dos posts pelo seu *id*.Após isso, é verificado de que tipo de Post está a ser tratado,através do método *get_post_type()* .Caso seja uma pergunta, é guardado o seu título ,procurado e guardado o nome do user que o criou e são retornados na forma de um par.

Interrogação 2

A segunda interrogação pretende obter o top N utilizadores com maior número de posts de sempre.Considerando tanto perguntas quanto respostas dadas pelo respectivo utilizador.

De maneira a resolver este problema é criado um ArrayList de Users, para a qual é aplicado um *sort*, que permite ordenar o *array* pelo número de posts de um *user*.Após isto é aplicada uma *stream* que "corta" a lista pelo N e devolve a tal lista.

Interrogação 3

Com a terceira interrogação é pretendido que, dado um intervalo de tempo arbitrário se obtenha o número total de posts. Deve ainda identificar-se as perguntas e respostas separadamente, no mesmo período. Para tal, criou-se um método que dada

uma *flag* (tipo de *post* : pergunta ou resposta), realiza uma travessia na respetiva *HashMap* devolvendo o tamanho de cada *HashTree* presente no *value* da mesma.

Interrogação 4

Na quarta interrogação é proposto a execução de um método que dado um intervalo de tempo arbitrário, retorne todas as perguntas que contenham uma determinada *tag*. Deve notar-se que o retorno deve ser uma lista de ID's das perguntas, ordenadas por cronologia inversa.

Para tal, foi realizada uma travessia à *HashMap* das perguntas verificando se a *tag* querida se encontra em cada post. Caso isso se verifique, essa *tag* é inserida num *ArrayList.* Após este procedimento, ordena-se o *ArrayList* através do método *sort* e efetua-se uma *String* que devolta todos os *id*'s dos *posts*.

Interrogação 5

Na quinta interrogação pretende-se devolver a informação do perfil de um determinado utilizador e os *ID*'s dos seus últimos 10 posts, ordenando-os por cronologia inversa. Para tal, realizou-se uma procura na *HashMap* dos users pelo ID pretendido. Em seguida, criou-se um *ArrayList* com todos os posts realizados por esse utilizador, ordenando-se estes através de um método *sort*. Para concluir, foi efetuada uma stream que permitiu devolver o *ID* dos posts e truncar o arraylist para o tamanho pretendido,10.

Interrogação 6

Na interrogação 6 , é pretendido que se devolva o *ID* das n respostas com mais votos. Esta devolução deve ser feita por ordem decrescente, tendo em conta o seu score. Deve notar-se que o *score* é definido pela diferença entre os *up votes e down votes*. Para que isto fosse possível começou-se por realizar uma travessia à HashMap das respostas, guardando todos os posts que se encontre entre as datas pretendidas para um *ArrayList*.

Em seguida, realizou-se um método *sort* dando como parametro o score da resposta. Por último utilizou-se o stream para obter o *ID* dos respectivos *posts*, truncando a lista pelo n pretendido.

Interrogação 7

Na sétima interrogação, pretende-se obter as n perguntas com mais respostas num dado intervalo de tempo, devolvendo por ordem decrescente do número de respostas.

Para realizar esta tarefa, iniciou-se uma travessia à *HashMap* das perguntas, guardando todos os *posts* que se encontre entre as datas pretendidas para um *ArrayList*. Em seguida, realizou-se um método *sort* dando como parâmetro o número de respostas. Para terminar utilizou-se o *stream* para obter o *ID* dos respectivos *posts*, truncando a lista pelo n pretendido.

Interrogação 8

Na presente interrogação, é pretendido que dada uma determinada palavra se devolva uma lista com os *ID's* de n perguntas cujos seus títulos a contenham. Esta devolução deve ser feita pela ordem inversa da cronologia.

Para realizar esta tarefa, iniciou-se uma travessia à *HashMap* das perguntas, guardando todos os posts que se encontre entre as datas pretendidas e que satisfaçam a condição de conterem a palavra dada no seu título,para um *ArrayList*. Em seguida, realizou-se um método *sort* dando como parâmetro a data de criação do *post*. Para terminar utilizou-se o stream para obter o *ID* dos respectivos posts, truncando a lista pelo n pretendido.

Interrogação 9

Nesta interrogação dados os ids de dois utilizadores pretende-se obter as últimas N perguntas em que participaram, dois utilizadores.De notar que os mesmos podem ter participado via pergunta ou resposta.

Contudo, o desenvolvimento desta interrogação não foi possível devido a complicações no que diz respeito ao desenvolvimento do código necessário para obtermos uma solução válida e elegante.

Interrogação 10

Nesta interrogação pretende-se que dado um id de uma pergunta se retorna a melhor resposta à mesma.Para isso é efetuado uma travessia à *HashMap* que possui todos os *posts* inseridos por *id* e para cada um deles é aplicado a seguinte cálculo : *0.45*p.get_score()+0.25*u.get_reputation()+0.2*p.get_score()+0.1*p.get_comment_count()*, vai sendo guardando o resultado numa variável auxiliar e comparando com os seguintes de maneira a obter o maior.Retornando o *id* da pergunta com melhor resultado.

Interrogação 11

Nesta interrogação pretende-se obter o identificador das N *tags* mais usadas mais usadas pelos N utilizadores com melhor reputação

Contudo, o desenvolvimento desta interrogação não foi possível devido a complicações no que diz respeito ao desenvolvimento do código necessário para obtermos uma solução válida e elegante.

Conclusão

Em suma, considera-se que o projeto geral, incluindo as duas fases, foi uma mais valia em termos de aprendizagem, uma vez que fez com que os discentes desenvolvessem competências nas duas linguagens de programação o que se tornará muito útil para trabalhos futuros. Após a conclusão da fase dois é possível fazer uma pequena comparação entre as duas fases. Assim sendo, conclui-se que a implementação na linguagem JAVA foi mais acessível, primeiro porque a estrutura base já estava pensada e executado e depois porque esta linguagem tem funcionalidades que permitem a execução mais rápida das queries pretendidas. Apesar de tudo isto, a otimização em C é mais rápida embora a memória sejam mais otimizada em JAVA, o que não causa problemas na gestão da mesma.