Trabalho final de INF01124 – 2021/1
Professor João Comba
FIFA21 - Players

Neste trabalho aplicamos diversas técnicas vistas em aula para explorar o dataset **FIFA21 - Players**. Estes dados foram disponibilizados no Kaggle[1] e a partir dele foram gerados os conjuntos de dados disponíveis para este trabalho. O enunciado do trabalho inicia com a descrição dos dados, seguida das tarefas solicitadas.

1. Dados:

Os dados são compostos dos arquivos **`players.csv`**, **`rating.csv`** e **`tags.csv`**, contendo respectivamente informações sobre jogadores, avaliações de usuários e anotações em texto-livre (t*ags*). O arquivo **`players.csv`** contém informações de 18.944 jogadores, composto de um id no FIFA, seu nome e uma lista de posições:

sofifa_id	name	player_positions
158023	Lionel Andrés Messi Cuccittini	RW, ST, CF
20801	Cristiano Ronaldo dos Santos Aveiro	ST, LW
200389	Jan Oblak	GK
188545	Robert Lewandowski	ST
190871	Neymar da Silva Santos Júnior	LW, CAM
192985	Kevin De Bruyne	CAM, CM
231747	Kylian Mbappé Lottin	ST, LW, RW
192448	Marc-André ter Stegen	GK

O arquivo **`rating.csv`** contém 24,188,078 de avaliações (notas entre 1 e 5) de usuários para jogadores. Esses dados foram geradores simulando avaliações de usuários para cada jogador disponível no dataset:

[1] https://www.kaggle.com/stefanoleone992/fifa-21-complete-player-dataset

	user_id	sofifa_id	rating
0	106180	158023	2.5
1	98236	158023	2.5
2	28038	158023	3.5
3	92821	158023	5.0
4	95674	158023	5.0
...	...	...	...
20683995	124317	257936	2.5
20683996	74541	257936	2.5
20683997	126198	257936	1.0
20683998	77921	257936	2.5
20683999	115420	257936	1.0

O arquivo **`tag.csv`** contém 364,950 anotações de texto livre (tags) (ex.: Brazil, FK Specialist, Speedster, Playmaker, Paris Saint-Germain) para 18,944 jogadores.

	user_id	sofifa_id	tag
0	17800	158023	Clinical Finisher
1	17800	158023	Complete Forward
2	17800	158023	Dribbler
3	17800	158023	Distance Shooter
4	17800	158023	FK Specialist
...	...	...	...
364945	28151	257936	Tianjin TEDA FC
364946	28151	257936	Chinese Super League
364947	110052	257936	China PR
364948	110052	257936	Tianjin TEDA FC
364949	110052	257936	Chinese Super League

Também disponibilizamos um arquivo com 10,000 avaliações ao invés das 20 milhões para ajudar nos testes (**`minirating.csv`** - disponível para download apenas no Moodle**).** A leitura dos dados a partir do CSV pode ser lenta, em especial para o arquivo `rating.csv` que possui

mais de 400MB de dados. É permitido usar código externo para leitura eficiente de arquivos CSV. Um exemplo de biblioteca é https://github.com/AriaFallah/csv-parser.

2. Pesquisas

O objetivo do trabalho é implementar estruturas de dados e algoritmos que suportam as seguintes pesquisas sobre os dados:

2.1 Pesquisas sobre os nomes de jogadores

Esta pesquisa tem por objetivo retornar a lista de jogadores cujo nome começa com um texto que pode ser o prefixo ou texto completo do nome de um jogador. Todos os jogadores que satisfizerem o texto de consulta devem ser retornados, junto com a lista de posições dos jogadores, avaliação média e número de avaliações. A sintaxe dessa consulta é:

```
player <name or prefix>
```

Um exemplo da consulta é dado abaixo:

```
$ player Fer
```

sofifa_id	name	player_positions	rating	count
135507	Fernando Luiz Rosa	CB, CDM	3.720244	6402
228618	Ferland Mendy	LB	3.186525	1410
184134	Fernando Francisco Reges	CDM, CM	3.421439	4977
209960	Fernando Pacheco Flores	GK	3.061224	343
241461	Ferran Torres García	RM, LM, CF	3.190110	455
202642	Fernando Lucas Martins	CDM, CM	2.981132	265
207707	Fernando Marçal de Oliveira	LB, CB	3.214713	4513
240289	Fernando Calero Villa	CB	2.920752	1224
240471	Fernando Gorriarán Fontes	CM, CDM	2.837413	572
162131	Fernando Javier Llorente Torres	ST	3.501560	9935

A consulta pode ser feita diretamente pelo console, e o resultado também pode ser impresso no console. Como no exemplo acima, tente deixar o texto a ser impresso compacto.

Uma primeira observação é que os dados de jogadores não contêm algumas das informações a serem retornadas. Por exemplo, deve-se primeiramente guardar em uma **tabela hash** as

médias de avaliações e total de avaliações para cada jogador. Essas informações devem ser calculadas e armazenadas em uma etapa de pré-processamento. Para responder esta pesquisa, deve-se implementar uma **árvore trie** que busca todos os identificadores de jogadores que correspondem ao nome ou prefixo dado, e com essa lista de identificadores, buscar na tabela hash as informações complementares dos jogadores.

2.2 Pesquisas sobre jogadores revisados por usuários

Esta pesquisa deve retornar a lista com no máximo 20 jogadores revisados pelo usuário e para cada jogador mostrar a nota dada pelo usuário, a média global e a contagem de avaliações. A lista deve ser ordenada pela nota dada pelo usuário de maneira decrescente (maiores notas primeiro). A sintaxe dessa consulta é:

`user <userID>`

Um exemplo da consulta é dado abaixo:

```
$ user 4
```

sofifa_id	name	global_rating	count	rating
158963	Lucas Rodrigo Biglia	3.529665	6034	5.0
41	Andrés Iniesta Luján	3.866787	9680	5.0
176733	Marcus Berg	3.506993	9367	5.0
185221	Luiz Gustavo Dias	3.612475	8633	5.0
199206	Domagoj Vida	3.577764	9098	5.0
210513	Juan Fernando Quintero Paniagua	3.307442	3682	4.5
202053	R. Miyaichi	2.950679	5525	4.5
168542	David Josué Jiménez Silva	3.988069	8717	4.5
198554	Anthony Ujah	3.016280	4484	4.5
208830	Jamie Vardy	3.769690	6945	4.5
204078	Rafał Wolski	2.557216	1372	4.0
254428	Joel David Ezequiel Fernández	2.604532	684	4.0
183285	Mamadou Sakho	3.532185	8622	4.0
192667	Allan Romeo Nyom	3.156884	2760	4.0
201938	Anthony Limbombe Ekango	2.788538	1518	4.0
25798	Roque Luis Santa Cruz Cantero	3.422341	9722	4.0
206254	Miroslav Stevanović	3.067771	4375	4.0
150106	Avraam Papadopoulos	2.938465	5826	4.0
120312	Michael Dawson	2.949701	2674	4.0
241569	Héctor David Martínez	2.833938	1379	4.0

Observem que os dados de entrada não possuem diretamente as informações de usuários prontas para o retorno dessa pesquisa. Estas informações estão colocadas no arquivo **`rating.csv`**. Deve-se implementar uma estrutura de dados (livre) para calcular e agregar em pré-processamento as informações por usuário. Com essa estrutura construída, para responder essa pesquisa basta consultar as entradas pelo identificador do usuário e retornar a lista de jogadores que esse usuário revisou e suas informações adicionais.

2.3 Pesquisas sobre os melhores jogadores de uma determinada posição

Esta pesquisa tem por objetivo retornar a lista de jogadores com melhores notas **de uma dada posição**. Para evitar que um jogador seja retornando com uma boa média mas com poucas avaliações, esta consulta somente deve retornar os melhores jogadores **com no mínimo 1000 avaliações**. Para gerenciar o número de jogadores a serem retornados, a consulta deve receber como parâmetro um número N que corresponde ao número máximo de jogadores a serem retornados. O resultado da consulta deve estar ordenado da maior para a menor avaliação. A sintaxe dessa consulta é:

`top<N> ‘<position>’`

Um exemplo da consulta é dado abaixo:

```
$ top10 ‘ST’
```

sofifa_id	name	player_positions	rating	count
158023	Lionel Andrés Messi Cuccittini	RW, ST, CF	4.256382	12887
20801	Cristiano Ronaldo dos Santos Aveiro	ST, LW	4.247577	11144
176580	Luis Alberto Suárez Díaz	ST	4.174944	9403
41236	Zlatan Ibrahimović	ST	4.105496	9389
188545	Robert Lewandowski	ST	4.081675	7989
153079	Sergio Leonel Agüero del Castillo	ST	4.052894	11249
165153	Karim Benzema	CF, ST	4.036720	10471
183277	Eden Hazard	LW, ST	4.014172	9314
194765	Antoine Griezmann	ST, CF, LW	4.006960	13937
167664	Gonzalo Gerardo Higuaín	ST	3.918668	7285

2.4 Pesquisas sobre ‘tags’ de jogadores

Esta pesquisa tem por objetivo explorar a lista de tags adicionadas por cada usuário em cada revisão. Para uma lista de tags dada como entrada, a pesquisa deve retornar a lista de jogadores que estão associados a esse conjunto de tags. A sintaxe dessa consulta é:

```
tags <list of tags>
```

Um exemplo da consulta é dado abaixo:

```
$ tags ‘Brazil’ ‘Dribbler’
```

sofifa_id	name	player_positions	rating	count
230658	Arthur Henrique Ramos de Oliveira Melo	CM	3.536732	2491
188803	Taison Barcellos Freda	LM, CAM	3.334329	4518
176676	Marcelo Vieira da Silva Júnior	LB	3.929943	11562
238794	Vinícius José Paixão de Oliveira Júnior	LW	3.547206	4385
230475	Adilson Patrick Edrada Pereira	CAM	2.888048	661
230666	Gabriel Fernando de Jesus	ST	3.476959	4340
201995	Felipe Anderson Pereira Gomes	LM, CAM	3.665974	5290
201400	Rafael Alcântara do Nascimento	CM, CAM, RM	3.292032	3652
210411	Otávio Edmilson da Silva Monteiro	RM, CAM, CM	2.922907	227
190483	Douglas Costa de Souza	LM, RW, LW	3.704957	9663
180403	Willian Borges da Silva	RW, LW, RM	3.678571	4004
200949	Lucas Rodrigues Moura da Silva	RM, CF	3.721487	4761
201942	Roberto Firmino Barbosa de Oliveira	CF	3.802751	5488
190871	Neymar da Silva Santos Júnior	LW, CAM	4.242804	17370
236632	David Neres Campos	LW, RW, CAM	3.706804	7246
189242	Philippe Coutinho Correia	CAM, LW, CM	3.928441	7106
222716	Everton Sousa Soares	LM	3.114625	253

Estas informações não estão diretamente nos arquivos de dados. Portanto, deve-se projetar uma estrutura de dados para responder essa pesquisa. Como as tags podem ser termos com espaço (ex.: Solid Player, French Ligue 1, Manchester United), a tag passada na consulta deve ser escrita entre apóstrofes.

3. Implementação

Os usuários devem construir uma aplicação que funciona em duas fases. A fase 1 corresponde a construção e inicialização das estruturas de dados necessárias para suportar as consultas. Ao executar a fase de construção, esta não deve demorar mais de 3 minutos. Quem conseguir fazer esta etapa em menos de 1 minuto ganha um bônus de 10%.

Depois dessas estruturas serem construídas, a aplicação entra na fase 2, que corresponde ao modo console. Nesta fase será possível fazer as pesquisas listadas na seção anterior, bem como a exibição dos resultados.

O arquivo `players_21.csv` está incluso entre os conjuntos de dados e trata-se do conjunto original. Neste temos diversas informações que podem ser utilizadas para melhorar as listagens como, por exemplo, clube, idade e nacionalidade dos jogadores.

Interfaces gráficas e novas consultas serão recompensadas com até 20 por cento de nota extra.

É possível fazer o trabalho em C, C++, Python e Java. Não é permitido usar bibliotecas ou mecanismos da linguagem de alto nível, nem implementações prontas para lidar, buscar ou armazenar os dados (ex.: pandas, numpy, dicionários, maps, bancos de dados). Todas as estruturas citadas anteriormente, buscas e ordenações devem ser implementadas pelo aluno. Não é permitido abrir os arquivos após a fase de construção e inicialização das estruturas.

4. Grupos e Apresentação

Os trabalhos podem ser feitos de grupos de até 2 pessoas. A definição dos componentes do grupo deve ser comunicada ao professor até a aula do dia **28/10**.

Os trabalhos serão apresentados na aula do dia **23/11**. Antes desta data será disponibilizado um link para os horários das apresentações dos respectivos grupos. Cada grupo terá aproximadamente 5 minutos para apresentar o trabalho. As seguintes instruções devem ser seguidas:

- cada grupo deve estar disponível 10 minutos antes do horário da apresentação no Teams da disciplina
- o trabalho deve ser mostrado com modo de compartilhamento de tela
- a aplicação deve já ter executado para construir as estruturas de dados de suporte, estar pronta para responder decisões de implementação, e executar elas

- o grupo deve relatar brevemente as seguintes informações no começo da apresentação: tempo de construção das estruturas de dados, e explicação das estruturas de dados usadas para cada uma das quatro consultas acima

- cada integrante deve estar apto para demonstrar como resolveu cada tarefa (explicar decisões de implementação), integrante não presente recebe nota 0

- enviar código fonte em um arquivo .zip no moodle