



Web Scrapping e Análise de Dados UEFA Champions League

Projeto Integrador

Gabriel de Jesus Nunes da Costa

Centro Universitário IESB – Brasília – DF – Brazil

Abstract. *The project presented aims to capture data on the UEFA Champions League football competition in Season 2020/2021 through the Python programming language, and realize a more precise study/analysis of the data obtained, seeking to understand and unravel the proposed data about the games played and about the players of each position and their areas, like goalkeepers, defenders, left-back, right-back, midfielders and forwards.*

Resumo. *O projeto apresentado tem como objetivo capturar os dados sobre a competição de futebol UEFA Champions League na Temporada 2020/2021 através da linguagem de programação Python, e realizar um estudo/análise mais precisa sobre os dados obtidos, procurando entender e desvendar os dados propostos sobre as partidas jogadas e sobre os jogadores de cada posição e suas áreas, sendo: goleiros, zagueiros, laterais, meio-campos e atacantes.*

Conteúdo

1	Concepção do Projeto	4
1.1	O que é a UEFA Champions League	4
2	Projeto de Pesquisa	5
3	Montagem do Projeto	6
3.1	Descrição das Atividades	7
3.1.1	Objetivo / Entendimento do Negócio	7
3.1.2	Como foi realizado ?	8
3.1.3	Coleta de Dados	9
3.1.4	Programação	11
3.1.5	Banco de Dados	19
4	Análise dos Resultados	24
4.1	Análises - Gerais	24
4.1.1	Time Campeão	24
4.1.2	Times que mais marcaram gols - Mandante e Visitante	24
4.1.3	Artilheiros do Campeonato	25
4.1.4	Jogadores com mais minutos em campo	26
4.1.5	Jogadores com mais divididas ganhas	26
4.1.6	Jogadores com mais interceptações	26
4.1.7	Jogadores com mais bloqueios	27
4.1.8	Jogadores com mais assistências	27
4.1.9	Análise Minuciosa - Goleiros	28
4.2	Time da Competição	31
4.3	Curiosidades	32

5	Conclusões	33
5.1	Relato da Experiência	33
6	Referências	34

1. Concepção do Projeto

1.1. O que é a UEFA Champions League

A UEFA Champions League é o campeonato mais importante disputados por clubes de futebol no território europeu, realizado pela UEFA (União das Federações Europeias de Futebol). É o equivalente à CONMEBOL Libertadores da América, disputada pelos clubes da América do Sul, e inclusive também dá, ao seu campeão, o acesso para disputar o campeonato mundial de clubes da FIFA.

Desde que o futebol moderno foi criado na Inglaterra, por volta de 1863, as disputas para saber quem era o melhor guiavam as partidas, que foram ganhando cada vez mais praticantes e expandindo suas fronteiras. Com isso, passadas algumas décadas, os europeus pensavam em como criar um torneio de proporções continentais para descobrir o melhor time daquele conglomerado de países.

A UEFA Champions League é um dos torneios mais prestigiados do mundo e a competição de clubes mais prestigiada no futebol europeu, disputada pelas equipas mais bem classificadas nos respectivos campeonato nacionais na época anterior, sendo o número de vagas atribuídos consoante o ranking da UEFA (União das Federações Europeias de Futebol).

Embora a maioria dos campeonatos nacionais da Europa apenas o campeão nacional possa participar, os campeonatos nacionais mais fortes oferecem quatro vagas para a competição e fornece até cinco vagas desde a temporada 2015–16. Os clubes que não se qualificam para a Liga dos Campeões, podem ser elegíveis para a Liga Europa da UEFA.

No seu formato atual, a Liga dos Campeões começa em meados de julho com três rodadas de qualificação e uma rodada de play-offs. As dez equipes sobreviventes dos play-offs entram na fase de grupos, juntando outras 22 equipes previamente qualificadas. As 32 equipes são colocadas em oito grupos de quatro equipes e jogam em um sistema ida e volta. As oito equipes que ficarem na primeira colocação em cada grupo e as oito que ficarem na segunda colocação de cada grupo, avançam para a fase eliminatória que culmina com a partida final em maio. O vencedor da Liga dos Campeões se qualifica para a Supercopa da UEFA e para a Copa do Mundo de Clubes da FIFA. Consequentemente a sua popularidade mundial, a final da Liga dos Campeões da UEFA é o evento esportivo anual mais visto em todo o mundo.



Figura 1. Logo UEFA Champions League

2. Projeto de Pesquisa

O projeto de pesquisa tem como base a realização de uma coleta de dados através da técnica de Web Scrapping e realizar uma análise de validação para entender as relações dos dados dos jogadores e dados das partidas ocorridas na competição UEFA Champions League na temporada de 2020-2021, onde no qual será acompanhado a inserção de dados do início das Oitavas de final da competição até o seu término e a inserção dos dados dos jogadores durante a competição inteira.

Como dito anteriormente a técnica de coleta de dados utilizada é o Web Scrapping, porém o que é Web Scrapping ?

É uma técnica de “garimpo” da internet que envolve extrair informações relevantes de determinado site para depois serem analisadas. Esses dados serão usados para aprimorar a tomada de decisões com maior chance de acerto e sucesso.

É possível fazer o mesmo processo manualmente, mas quando se fala de Web Scrapping a ideia é automatizar o trabalho usando bots. Assim, é possível coletar um número muito maior de dados em uma curta fração do tempo.

Para a fundamentação dos dados é preciso realizar uma coleta de dados de forma eficiente e correta, pois as informações obtidas através dos dados entregam e agregam mais conhecimento para uma utilização e uma análise mais precisa. Antes da coleta de dados foi realizado um **Roadmap** para entender o objetivo do projeto e projetar passo a passo do projeto e foi realizado um estudo para entender como o projeto seria desenvolvido.

Foram utilizados dois sites para a pesquisa e coleta dos dados, sendo o primeiro site o Gazzeta Esportiva para a coleta de dados referentes as informações de partidas, resultados, local horário e etc, já a coleta de dados referentes aos jogadores e suas informações em campo, foi utilizado o site de estatísticas FBref, local onde contém as estatísticas de diversos campeonatos.

A pesquisa do projeto teve início no dia 10 de Março de 2021, dia onde estava sendo realizado a segunda etapa dos jogos das Oitavas de Finais. Através desse fato pôde-se avaliar os dados sendo coletados desde o início do campeonato e analisar o preenchimento da base de dados a cada fase que o campeonato acontecia.

3. Montagem do Projeto

Como citado acima, o primeiro passo para a realização do projeto foi realizar um **Roadmap**, para ter um entendimento sobre cada passo que iria ser realizado.



Figura 2. Roadmap

3.1. Descrição das Atividades

3.1.1. Objetivo / Entendimento do Negócio

A parte primária para dar início ao projeto foi entender o objetivo do projeto, por isso algumas perguntas foram colocadas em pauta para o projeto ser guiado de forma correta, onde as principais perguntas são:

- Porque coletar os dados da UEFA Champions League ?
- Qual tipo de análise vai ser realizada ?

Logo, o objetivo do projeto é:

"Extrair os dados utilizando a técnica de Web Scrapping e realizar uma análise de relação dos dados pós competição e comparar com o desempenho dos times e jogadores"

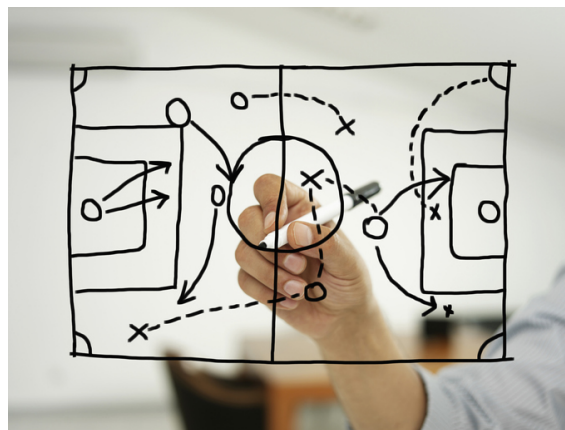


Figura 3. Ilustração: Dados e Futebol

3.1.2. Como foi realizado ?

Para realizar todo o projeto, a linguagem de programação utilizada foi o Python para fazer o Web Scrapping , pois é uma linguagem mais apropriada para manipulações de dados. Segundamente foi estudado sobre o que é, e como funciona o Web Scrapping, que é a coleta de dados web, ou raspagem web, onde é uma forma de mineração que permite a extração de dados de sites da web convertendo-os em informação estruturada para posterior análise. Logo depois foi realizado diversas pesquisas e estudos para aprender a como se fazer um Web Scrapping e como utilizar os dados extraídos das páginas web.

Softwares utilizados no projeto:

- **Python:** Linguagem de programação de alto nível, interpretada de script e orientada a objetos.
- **Visual Studio Code:** é um editor de código-fonte desenvolvido pela Microsoft para Windows, Linux e macOS.
- **Docker:** Tecnologia de containerização, que pode ser utilizada para criar containers Linux. Cada container funciona como uma máquina virtual modular.
- **PostgreSQL:** Sistema gerenciador de banco de dados.
- **DBeaver:** Ferramenta de administração de banco de dados
- **Anaconda - Jupyter Notebook:** software para realizar a análise de dados.

Para realizar a programação e o Web Scrapping foram utilizadas três bibliotecas, sendo:

- **Requests:** biblioteca que permite que seja enviada solicitações HTTP em Python.
- **Beautiful Soup:** biblioteca Python de extração de dados de arquivos HTML e XML.
- **Psycopg:** biblioteca para realizar conexão com o banco de dados PostgreSQL

Foram utilizados como base 2 sites para realizar a extração dos dados sobre a UEFA Champions League 2021/2021, sendo eles:

- **Gazzeta Esportiva** - "www.gazetaesportiva.com"
- **FBref** - "www.fbref.com"

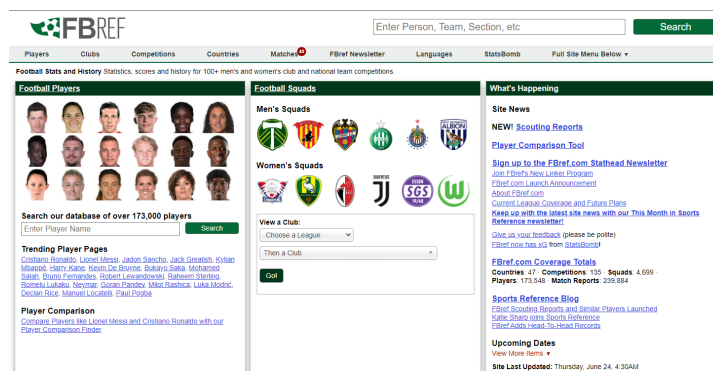


Figura 4. Página Inicial FBref

3.1.3. Coleta de Dados

Os dados foram coletados como dito anteriormente pela técnica de Web Scrapping. Através do Web Scrapping foi possível obter o HTML completo das páginas, e através desse fato foi possível pesquisar as informações através do próprio código fonte, buscando internamente dentro da estrutura do HTML, como: **Head, Title, Body, Divs e etc**, assim facilitando para obter os textos presentes nas páginas.

Para facilitar na programação e no Web Scrapping foi decidido que os dados seriam coletados de acordo com os dados das partidas e a posição do jogador, sendo no total divididos em 6 programas, onde possuem os dados dos :

- **Matches_info_UCL** - Jogos
- **Stats_Strikers** - Artilheiros
- **Stats_Assists** - Meios campistas
- **Stats_Defensor_2** - Volantes
- **Stats_Defensor** - Defensores
- **Stats_GK** - Goleiros

Abaixo segue os códigos realizados e exemplo de código fonte HTML, onde foi coletado os dados.







Nome	Data de modificação	Tipo	Tamanho
 Matches_Infos_UCL	23/06/2021 10:09	Arquivo PY	8 KB
 Stats.Defensor	23/06/2021 09:21	Arquivo PY	12 KB
 Stats.Defensor_2	23/06/2021 09:19	Arquivo PY	11 KB
 Stats_Assists	23/06/2021 14:22	Arquivo PY	7 KB
 Stats_GK	22/06/2021 20:42	Arquivo PY	8 KB
 Stats_Strikers	23/06/2021 09:27	Arquivo PY	8 KB

Figura 5. Códigos Realizados

```

</select>
</div></div>
<div class=" to hidden"><div class="formfield label"><span class="to to hidden">a</span>
</div></div>
<div class=" to hidden"><div class="formfield dropdown"><select id="plyrto" name="plyrto" class="no_chosen to hidden">
  <option value="" selected>Carreira completa</option>

  <option value="1" disabled>Club Seasons</option>
  <option value="2017">2017 (idade 16, Molde)</option>
  <option value="2018">2018 (idade 17, Molde)</option>
  <option value="2018-2019">2018-2019 (idade 18, RB Salzburg)</option>
  <option value="2019-2020">2019-2020 (idade 19, 2 clubes)</option>
  <option value="2020-2021">2020-2021 (idade 20, Dortmund)</option>
  <option value="1" disabled>National Team Seasons</option>
  <option value="2020-2021">2020-2021 (idade 20, Norway)</option>
  <option value="2021">2021 (idade 20, Norway)</option>
  <option value="2022">2022 (idade 21, Norway)</option>
</select>
</div></div>

```

Figura 6. Exemplo: Código Fonte HTML

Na área estatística relacionado ao futebol existem diversos campos de dados que podem ser utilizados, porém o foco foi dividido de acordo com a posição do jogador e as suas estatísticas principais. Segue abaixo um dicionário das colunas coletadas e seu respectivo significado:

Colunas sobre os dados da partida - Tabela **Matches_Infos_UCL**

- **Data_Jogo** : Data do jogo;
- **Dia_Semana** : Dia da semana que ocorreu o jogo;
- **Estadio_Jogo** : Estádio onde ocorreu o jogo;
- **Etapa** : Qual etapa se referia o jogo (Oitavas, Quartas, Semi, Final);
- **Horario_Jogo** : Horário do jogo;
- **Placar_Mandante** : Placar do time mandante;
- **Placar_Visitante** : Placar do time visitante;
- **Time_Mandante** : Nome do time mandante;
- **Time_Visitante** : Nome do time visitante;

Colunas sobre os jogadores - Tabelas **Stats**

- **Assists** : Assistências realizadas;
- **Assists_per90** : Assistências dividido por 90;
- **Blocks** : Número de vezes que a bola foi bloqueada ao ficar no caminho;
- **Blocks_Pass** : Número de vezes que a bola foi bloqueada na hora do passe;
- **Blocks_Shot** : Número de vezes que a bola foi bloqueada na hora do chute adversário;
- **Cleareance** : Cortes / Chutões;
- **Games** : Jogos realizados;
- **Games_Start** : Jogos como titular(começa jogando);
- **Goals** : Gols marcados;
- **Goals_Against** : Gols sofridos;
- **Goals_Against_per90** : Gols sofridos dividido por 90;
- **Goals_per90** : Gols dividido por 90;
- **Interception** : Quantidade de interceptações;
- **Minutes** : Minutos jogados;
- **Minutes_90s** : Minutos jogados dividido por 90;
- **Player** : Nome do Jogador;
- **Position_Player** : Posição;
- **Saves** : Defesas realizadas;
- **Saves_Percent** : Percentual de Defesas;
- **Shots_on_Target** : Quantidade de chutes sofridos;
- **Tackle** : Divididas/Combates;
- **Tackle_ATK** : Divididas ganhas no ataque;
- **Tackle_DEF** : Divididas ganhas na defesa;
- **Tackle_MID** : Divididas ganhas no meio campo;
- **Tackle_Won** : Divididas que o time ganhou a posse de bola;
- **Team** : Time;

3.1.4. Programação

Todos códigos seguiram o mesmo padrão, utilizando as bibliotecas **Requests** e **BeautifulSoup**. A estrutura dos dados também foram formatadas para uma melhor inserção no banco de dado e para uma utilização melhor dos dados, para futuras análises. De acordo com os Web Scrapings realizados foi definido que os dados seriam inseridos dentro de estruturas de listas para facilitar a inserção no banco de dados PostgreSQL, logo os dados coletados seguem o mesmo padrão de utilização em lista.

Abaixo segue 2 exemplos dos 6 códigos realizados:

Código Web Scraping para coletar dados sobre os jogos, horários e times.

```
1 # BIBLIOTECAS
2
3 import requests
4 import psycopg2
5 from bs4 import BeautifulSoup
6
7 # BANCO DE DADOS
8 conn = psycopg2.connect(database = 'projeto_integrado' , user =
    'postgres' , password = '123' )
9 cur = conn.cursor()
10
11 # REQUESTS / BEAUTIFULSOUP
12
13 headers = {'User-Agent': 'Mozilla/5.0 (Windows NT 10.0; Win64;
    x64) AppleWebKit/537.36 (KHTML, like Gecko)
    Chrome/90.0.4430.85 Safari/537.36'}
14 response =
    requests.get('https://www.gazetaesportiva.com/campeonatos/
15 uefa-champions-league-2020/', headers = headers)
16
17 print(response.status_code)
18
19 content = response.content
20
21 site = BeautifulSoup(content, "html.parser")
22
23 # FORMATAÇÃO DOS DADOS
24
25 html_UCL = site.find_all('a', attrs={'class' : 'team-link'})
26
27 times_UCL = []
28
29 for conteudo in html_UCL:
30     if conteudo.get_text() != '':
31         times_UCL.append(conteudo.get_text())
```

```
1 oitavas_de_final = times_UCL[0:32]
2 quartas_de_final = times_UCL[32:48]
3 semi_final = times_UCL[48:56]
4 final = times_UCL[56:]
5
6 times_oitavas_final = []
7 for i in range(0, len(oitavas_de_final), 2):
8     times_oitavas_final.append([oitavas_de_final[i],
9     oitavas_de_final[i+1]])
10
11 times_quartas_final = []
12 for i in range(0, len(quartas_de_final), 2):
13     times_quartas_final.append([quartas_de_final[i],
14     quartas_de_final[i+1]])
15
16 times_semi_final = []
17 for i in range(0, len(semi_final), 2):
18     times_semi_final.append([semi_final[i], semi_final[i+1]])
19
20 times_final=[]
21 for i in range(0, len(final), 2):
22     times_final.append([final[i], final[i+1]])
23
24 placares_UCL = site.find_all('span', attrs={'class' : 'score'})

```

```
1 resultados_jogos_UCL = []
2
3 for resultado in placares_UCL:
4
5     placar_texto = resultado.get_text()
6
7     placar_texto_format = []
8
9     for conteudo_placar in placar_texto:
10
11         if conteudo_placar != '\n':
12             placar_texto_format.append(int(conteudo_placar))
13
14     resultados_jogos_UCL.append(placar_texto_format)
15
16 # PLACARES DOS JOGOS DE ACORDO COM A FASE
17
18 resultados_UCL_oitavas = resultados_jogos_UCL[0:16]
19 resultados_UCL_quartas = resultados_jogos_UCL[16:24]
20 resultados_UCL_semis = resultados_jogos_UCL[24:28]
21 resultados_UCL_final = resultados_jogos_UCL[28:]
22
23 # DIA, DATA, HORA, ESTADIO =
```

```
24
25 info_site_UCL = site.find_all('span', attrs={'class' : 'date'})

1 info_format_UCL = []
2
3 for info in info_site_UCL:
4
5     info_texto = info.get_text()
6
7     informacoes_gerais = info_texto.split('-')
8
9     informacoes_gerais[1] = info_texto[6:11]
10
11     informacoes_gerais.append(info_texto[12:17])
12
13     informacoes_gerais.append(info_texto[20:])
14
15     info_format_UCL.append(informacoes_gerais)
16
17 # Formatação das informações gerais
18
19 informacoes_oitavas = info_format_UCL[0:16]
20 informacoes_quartas = info_format_UCL[16:24]
21 informacoes_semis = info_format_UCL[24:28]
22 informacoes_final = info_format_UCL[28:]
23
24 # INSERÇÃO NO BANCO DE DADOS
25
26 for i in range(len(times_oitavas_final)):
27     inserir_oitavas = "Insert into Matches_infos
28         (time_mandante, time_visitante, placar_mandante,
29         placar_visitante, estadio_jogo, dia_semana, data_jogo,
30         horario_jogo, etapa) values( %s, %s, %s, %s, %s, %s, %s, %s,
31         %s);"
32     cur.execute(inserir_oitavas, (times_oitavas_final[i][0],
33     times_oitavas_final[i][1],
34     str(resultados_UCL_oitavas[i][0]),
35     str(resultados_UCL_oitavas[i][1]),
36     informacoes_oitavas[i][3], informacoes_oitavas[i][0],
37     informacoes_oitavas[i][1], informacoes_oitavas[i][2],
38     'Oitavas de final'))
39
40 for i in range(len(times_quartas_final)):
41     inserir_quartas = "Insert into Matches_infos
42         (time_mandante, time_visitante, placar_mandante,
43         placar_visitante, estadio_jogo, dia_semana, data_jogo,
44         horario_jogo, etapa) values( %s, %s, %s, %s, %s, %s, %s, %s,
45         %s);"
46     cur.execute(inserir_quartas, (times_quartas_final[i][0],
```

```
times_quartas_final[i][1],
str(resultados_UCL_quartas[i][0]),
str(resultados_UCL_quartas[i][1]),
informacoes_quartas[i][3], informacoes_quartas[i][0],
informacoes_quartas[i][1], informacoes_quartas[i][2],
'Quartas de finais'))

33
34 for i in range(len(times_semi_final)):
35     inserir_semi = "Insert into Matches_infos (time_mandante,
time_visitante, placar_mandante, placar_visitante,
estadio_jogo, dia_semana, data_jogo, horario_jogo, etapa)
values( %s, %s, %s, %s, %s, %s, %s, %s, %s);"

36
37     if len(resultados_UCL_semis[i]) != 0:
38         cur.execute(inserir_semi, (times_semi_final[i][0],
times_semi_final[i][1], str(resultados_UCL_semis[i][0]),
str(resultados_UCL_semis[i][1]), informacoes_semis[i][3],
informacoes_semis[i][0], informacoes_semis[i][1],
informacoes_semis[i][2], 'Semifinal'))

39
40 for i in range (len(times_final)):
41     inserir_final = "Insert into Matches_infos (time_mandante,
time_visitante, placar_mandante, placar_visitante,
estadio_jogo, dia_semana, data_jogo, horario_jogo, etapa)
values( %s, %s, %s, %s, %s, %s, %s, %s, %s);"

42
43     if len(resultados_UCL_final[i]) != 0:
44         cur.execute(inserir_final, (times_final[i][0],
times_final[i][1], str(resultados_UCL_final[i][0]),
str(resultados_UCL_final[i][1]), informacoes_final[i][3],
informacoes_final[i][0], informacoes_final[i][1],
informacoes_final[i][2], 'Final'))

45
46 conn.commit()
```

Código Web Scrapping para coletar dados sobre os artilheiros

```
1 ## BIBLIOTECAS
2
3 import requests
4 import psycopg2
5 from bs4 import BeautifulSoup as soup
6
7 ## BANCO DE DADOS
8
9 conn = psycopg2.connect(database = 'projeto_integrado' , user =
'postgres' , password = '123' )
10 cur = conn.cursor()
11
```

```

12 ## FUNÇÕES
13 def Data_Format (lista):
14
15     del(lista [0])
16     del(lista [6:11])
17     del(lista [8:])
18
19 def Replace_Float(lista, lista_nova):
20     for i in lista:
21         z = i.replace(",",".")
22         lista_nova.append(float(z))
23
24 # REQUESTS / BEAUTIFULSOUP
25 url =
26     'https://fbref.com/pt/stathead/player_comparison.cgi?request
27 =1&sum=0&comp_type=spec&dom_lg=1&spec_comps=8&player_id1=1f44ac
28 21&plyrfrom=2020-2021&player_id2=42fd9c7f&p2yrfrom=2020-2021&
29 player_id3=ald5bd30&p3yrfrom=2020-2021&player_id4=69384e5d&p4yr
30 from=2020-2021&player_id5=04e17fd5&p5yrfrom=2020-2021&player_id
31 6=e342ad68&p6yrfrom=2020-2021&player_id7=129af0db&p7yrfrom
32 =2020-2021'
33 r = requests.get(url)
34
35 soup = soup(r.text, 'html.parser')
36
37 league_table = soup.find('table', class_ = 'min_width
38     per_g_toggle sortable stats_table min_width shade_zero')
39
40 # LISTAS DE MANIPULAÇÃO
41 Headers = [] # Titulos da coluna
42 Jogadores_Artilheiros = [] # Lista com nome dos artilheiros
43 Teams_Strikers = [] # Times dos artilheiros
44 Data_Strikers = [] # Dados dos artilheiros
45 Position_Players = [] # Posição dos jogadores
46 # ENCONTRANDO OS ELEMENTOS
47
48 # LISTA DE JOGADORES
49 for content in league_table.find_all('tbody'):
50     access_tr_1 = content.find_all('tr')
51     for value in access_tr_1:
52         players_list = value.find('th', class_ = 'left').text
53         Jogadores_Artilheiros.append(players_list)
54
55 # DADOS
56 for content in league_table.find_all('tbody'): # Encontrar o
57     tbody (estrutura todo conteúdo da tabela 'corpo')
58     access_tr_2 = content.find_all('tr') # Encontrar todos os
59     tr
  
```

```
11     for content_ in access_tr_2:
12         access_td = content_.find_all('td', class_ = 'right') #
Especificar div e a classe
13         for value in access_td:
14             header_table = ((value.get('data-stat')) # header
da tabela
15             value_table = (value.get_text()) # valor
16             Data_Strikers.append(value_table)
17             Headers.append(header_table)
18 # EQUIPE
19 for content in league_table.find_all('tbody'): # Encontrar o
tbody (estrutura todo conteúdo da tabela 'corpo')
20     access_tr_3 = content.find_all('tr') # Encontrar todos os
tr
21     for content_ in access_tr_3:
22         access_td = content_.find_all('td', class_ = 'left') #
Especificar div e a classe
23         for value in access_td:
24             y = (value.get_text()) # valor
25             Teams_Strikers.append(y)
26
27 Teams_Strikers = Teams_Strikers[1::2]
28
29 Teams_Strikers[0] = 'Borussia Dortmund'
30 Teams_Strikers[1] = 'Paris Saint Germain'
31 Teams_Strikers[3] = 'Paris Saint Germain'
32
33 # POSIÇÃO
34 for content in league_table.find_all('tbody'): # Encontrar o
tbody (estrutura todo conteúdo da tabela 'corpo')
35     access_tr_4 = content.find_all('tr') # Encontrar todos os
tr
36     for content_ in access_tr_4:
37         access_td = content_.find_all('td', class_ = 'center')
# Especificar div e a classe
38         for value in access_td:
39             p = (value.get_text()) # valor
40             Position_Players.append(p)
41
42 # DELIMITANDO DADOS NAS LISTAS
43 Data_Striker_1 = Data_Strikers[0:26]
44 Data_Striker_2 = Data_Strikers[26:52]
45 Data_Striker_3 = Data_Strikers[52:78]
46 Data_Striker_4 = Data_Strikers[78:104]
47 Data_Striker_5 = Data_Strikers[104:130]
48 Data_Striker_6 = Data_Strikers[130:156]
49
50 # Objetivos : games - games_start - minutes - minutes_90s -
goals - assists - goals_90 - assists_90
```



```
51 Headers = Headers[:26]
52 # UTILIZAÇÃO DA FUNÇÃO DE FORMATAÇÃO DE LISTAS
53 Data_Format(Headers)
54 Data_Format(Data_Striker_1)
55 Data_Format(Data_Striker_2)
56 Data_Format(Data_Striker_3)
57 Data_Format(Data_Striker_4)
58 Data_Format(Data_Striker_5)
59 Data_Format(Data_Striker_6)
60
61 # DADOS FORMATADOS DE CADA ARTILHEIRO
62 Data_Striker_Format_1 = []
63 Data_Striker_Format_2 = []
64 Data_Striker_Format_3 = []
65 Data_Striker_Format_4 = []
66 Data_Striker_Format_5 = []
67 Data_Striker_Format_6 = []
68
69 Replace_Float(Data_Striker_1, Data_Striker_Format_1)
70 Replace_Float(Data_Striker_2, Data_Striker_Format_2)
71 Replace_Float(Data_Striker_3, Data_Striker_Format_3)
72 Replace_Float(Data_Striker_4, Data_Striker_Format_4)
73 Replace_Float(Data_Striker_5, Data_Striker_Format_5)
74 Replace_Float(Data_Striker_6, Data_Striker_Format_6)

1 # INSERÇÃO ARTILHEIROS
2 cur.execute("""INSERT INTO Stats_UCL_ATK_MID (player, team,
    position_player, games, games_starts, minutes, minutes_90s,
    goals,assists, goals_per90, assists_per90) VALUES(%s,%s,%s,
    %s, %s, %s, %s, %s, %s, %s, %s, %s)""",
3 (Jogadores_Artilheiros[0],Teams_Strikers[0],Position_Players[0],
    Data_Striker_Format_1[0], Data_Striker_Format_1[1],
    Data_Striker_Format_1[2], Data_Striker_Format_1[3],
    Data_Striker_Format_1[4],
    Data_Striker_Format_1[5],Data_Striker_Format_1[6],
    Data_Striker_Format_1[7]))
4
5 cur.execute("""INSERT INTO Stats_UCL_ATK_MID (player, team,
    position_player, games, games_starts, minutes, minutes_90s,
    goals,assists, goals_per90, assists_per90) VALUES(%s,%s,%s,
    %s, %s, %s, %s, %s, %s, %s, %s)""",
6 (Jogadores_Artilheiros[1],Teams_Strikers[1],Position_Players[1],
    Data_Striker_Format_2[0], Data_Striker_Format_2[1],
    Data_Striker_Format_2[2], Data_Striker_Format_2[3],
    Data_Striker_Format_2[4],
    Data_Striker_Format_2[5],Data_Striker_Format_2[6],
    Data_Striker_Format_2[7]))
7
8 cur.execute("""INSERT INTO Stats_UCL_ATK_MID (player, team,
```

```
position_player, games, games_starts, minutes, minutes_90s,
goals,assists, goals_per90, assists_per90) VALUES(%s,%s,%s,
%s, %s, %s, %s, %s, %s, %s, %s)""",
9 (Jogadores_Artilheiros[2],Teams_Strikers[2],Position_Players[2],
Data_Striker_Format_3[0], Data_Striker_Format_3[1],
Data_Striker_Format_3[2], Data_Striker_Format_3[3],
Data_Striker_Format_3[4],
Data_Striker_Format_3[5],Data_Striker_Format_3[6],
Data_Striker_Format_3[7]))
10
11 cur.execute("""INSERT INTO Stats_UCL_ATK_MID (player, team,
position_player, games, games_starts, minutes, minutes_90s,
goals,assists, goals_per90, assists_per90) VALUES(%s,%s,%s,
%s, %s, %s, %s, %s, %s, %s, %s)""",
12 (Jogadores_Artilheiros[3],Teams_Strikers[3],Position_Players[3],
Data_Striker_Format_4[0], Data_Striker_Format_4[1],
Data_Striker_Format_4[2], Data_Striker_Format_4[3],
Data_Striker_Format_4[4],
Data_Striker_Format_4[5],Data_Striker_Format_4[6],
Data_Striker_Format_4[7]))
13
14 cur.execute("""INSERT INTO Stats_UCL_ATK_MID (player, team,
position_player, games, games_starts, minutes, minutes_90s,
goals,assists, goals_per90, assists_per90) VALUES(%s,%s,%s,
%s, %s, %s, %s, %s, %s, %s, %s)""",
15 (Jogadores_Artilheiros[4],Teams_Strikers[4],Position_Players[4],
Data_Striker_Format_5[0], Data_Striker_Format_5[1],
Data_Striker_Format_5[2], Data_Striker_Format_5[3],
Data_Striker_Format_5[4],
Data_Striker_Format_5[5],Data_Striker_Format_5[6],
Data_Striker_Format_5[7]))
16
17 cur.execute("""INSERT INTO Stats_UCL_ATK_MID (player, team,
position_player, games, games_starts, minutes, minutes_90s,
goals,assists, goals_per90, assists_per90) VALUES(%s,%s,%s,
%s, %s, %s, %s, %s, %s, %s, %s)""",
18 (Jogadores_Artilheiros[5],Teams_Strikers[5],Position_Players[5],
Data_Striker_Format_6[0], Data_Striker_Format_6[1],
Data_Striker_Format_6[2], Data_Striker_Format_6[3],
Data_Striker_Format_6[4],
Data_Striker_Format_6[5],Data_Striker_Format_6[6],
Data_Striker_Format_6[7]))
19
20 conn.commit()
```

3.1.5. Banco de Dados

Logo após toda coleta de dados e toda estrutura de formatação dos dados alinhada o próximo passo foi criar um container no DOCKER e instalar o PostGreSQL para inserir todos os dados de programação no banco de dados. Antes da inserção foi realizado alguns scripts para a criação e configuração do banco de dados, logo segue abaixo todo passo a passo pra inserção dos dados banco de dados:

1. Configurar o sistema no DBeaver e criar o banco de dados chamado **Projeto Integrado**;
2. Criar as tabelas e definir os tipos das colunas;
3. Realizar a conexão do código Python com o banco de dados, através da biblioteca Psycopg2;
4. Inserir as listas com os dados para as tabelas;

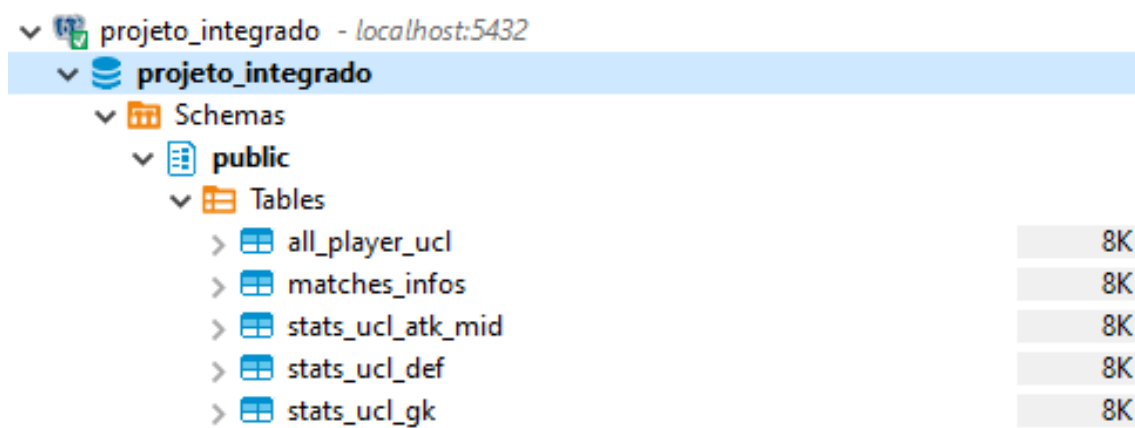


Figura 7. Banco de Dados e Tabelas

```

/* UEFA CHAMPIONS LEAGUE TABLES*/

-- TABELA DE DADOS DOS JOGOS
create table Matches_infos (
  id_jogo serial,
  time_mandante varchar(50),
  time_visitante varchar(50),
  placar_mandante int,
  placar_visitante int,
  estadio_jogo varchar(50),
  dia_semana varchar(3),
  data_jogo varchar(6),
  horario_jogo varchar(6),
  etapa varchar(50)
)

```

Figura 8. Criação de Tabela dos Dados dos jogos

Para a parte de comunicação Python e Banco de Dados PostGreSQL, foi utilizado a biblioteca Psycopg2. Abaixo segue a programação com a conexão e as tabelas implantadas no banco de dados:

Primeira parte de conexão Python - PostGreSQL

```
1 import psycopg2
2
3 ## BANCO DE DADOS
4
5 conn = psycopg2.connect(database = 'projeto_integrado' , user =
    'postgres' , password = '123' )
6 cur = conn.cursor()
```

Inserção dos dados nas tabelas

```
1 # INSERÇÃO ARTILHEIROS
2 cur.execute("""INSERT INTO Stats_UCL_ATK_MID (player, team,
    position_player, games, games_starts, minutes, minutes_90s,
    goals,assists, goals_per90, assists_per90) VALUES(%s,%s,%s,
    %s, %s, %s, %s, %s, %s, %s)""",
3 (Jogadores_Artilheiros[0],Teams_Strikers[0],Position_Players[0],
    Data_Striker_Format_1[0], Data_Striker_Format_1[1],
    Data_Striker_Format_1[2], Data_Striker_Format_1[3],
    Data_Striker_Format_1[4],
    Data_Striker_Format_1[5],Data_Striker_Format_1[6],
    Data_Striker_Format_1[7]))
4
5 cur.execute("""INSERT INTO Stats_UCL_ATK_MID (player, team,
    position_player, games, games_starts, minutes, minutes_90s,
    goals,assists, goals_per90, assists_per90) VALUES(%s,%s,%s,
    %s, %s, %s, %s, %s, %s, %s)""",
6 (Jogadores_Artilheiros[1],Teams_Strikers[1],Position_Players[1],
    Data_Striker_Format_2[0], Data_Striker_Format_2[1],
    Data_Striker_Format_2[2], Data_Striker_Format_2[3],
    Data_Striker_Format_2[4],
    Data_Striker_Format_2[5],Data_Striker_Format_2[6],
    Data_Striker_Format_2[7]))
7
8 cur.execute("""INSERT INTO Stats_UCL_ATK_MID (player, team,
    position_player, games, games_starts, minutes, minutes_90s,
    goals,assists, goals_per90, assists_per90) VALUES(%s,%s,%s,
    %s, %s, %s, %s, %s, %s, %s)""",
9 (Jogadores_Artilheiros[2],Teams_Strikers[2],Position_Players[2],
    Data_Striker_Format_3[0], Data_Striker_Format_3[1],
    Data_Striker_Format_3[2], Data_Striker_Format_3[3],
    Data_Striker_Format_3[4],
    Data_Striker_Format_3[5],Data_Striker_Format_3[6],
    Data_Striker_Format_3[7]))
```

```
10
11 cur.execute("""INSERT INTO Stats_UCL_ATK_MID (player, team,
    position_player, games, games_starts, minutes, minutes_90s,
    goals,assists, goals_per90, assists_per90) VALUES(%s,%s,%s,
    %s, %s, %s, %s, %s, %s, %s, %s)""",
12 (Jogadores_Artilheiros[3],Teams_Strikers[3],Position_Players[3],
    Data_Striker_Format_4[0], Data_Striker_Format_4[1],
    Data_Striker_Format_4[2], Data_Striker_Format_4[3],
    Data_Striker_Format_4[4],
    Data_Striker_Format_4[5],Data_Striker_Format_4[6],
    Data_Striker_Format_4[7]))
13
14 cur.execute("""INSERT INTO Stats_UCL_ATK_MID (player, team,
    position_player, games, games_starts, minutes, minutes_90s,
    goals,assists, goals_per90, assists_per90) VALUES(%s,%s,%s,
    %s, %s, %s, %s, %s, %s, %s, %s)""",
15 (Jogadores_Artilheiros[4],Teams_Strikers[4],Position_Players[4],
    Data_Striker_Format_5[0], Data_Striker_Format_5[1],
    Data_Striker_Format_5[2], Data_Striker_Format_5[3],
    Data_Striker_Format_5[4],
    Data_Striker_Format_5[5],Data_Striker_Format_5[6],
    Data_Striker_Format_5[7]))
16
17 cur.execute("""INSERT INTO Stats_UCL_ATK_MID (player, team,
    position_player, games, games_starts, minutes, minutes_90s,
    goals,assists, goals_per90, assists_per90) VALUES(%s,%s,%s,
    %s, %s, %s, %s, %s, %s, %s, %s)""",
18 (Jogadores_Artilheiros[5],Teams_Strikers[5],Position_Players[5],
    Data_Striker_Format_6[0], Data_Striker_Format_6[1],
    Data_Striker_Format_6[2], Data_Striker_Format_6[3],
    Data_Striker_Format_6[4],
    Data_Striker_Format_6[5],Data_Striker_Format_6[6],
    Data_Striker_Format_6[7]))
19
20 conn.commit()
```

Tabelas preenchidas com os dados

	123 id_jogo	nsc time_mandante	nsc time_visitante	123 placar_mandante	123 placar_visitante	nsc estadio_jogo	nsc dia_semana	nsc data_jogo	nsc horario_jogo	nsc etapa
1	1	RB Leipzig	Liverpool	0	2	Puskas Stadium	ter	16/02	17:00	Oitavas de final
2	2	Liverpool	RB Leipzig	2	0	Puskas Stadium	qua	16/03	17:00	Oitavas de final
3	3	Barcelona	Paris Saint Germain	1	4	Camp Nou	ter	16/02	17:00	Oitavas de final
4	4	Paris Saint Germain	Barcelona	1	1	Parc des Princes	qua	16/03	17:00	Oitavas de final
5	5	Porto	Juventus	2	1	Estádio do Dragão	qua	17/02	17:00	Oitavas de final
6	6	Juventus	Porto	3	2	Juventus Stadium	ter	09/03	17:00	Oitavas de final
7	7	Sevilla	Borussia Dortmund	2	3	Ramón Sánchez Pizjuán	qua	17/02	17:00	Oitavas de final
8	8	Borussia Dortmund	Sevilla	2	2	Signal Iduna Park	ter	09/03	17:00	Oitavas de final
9	9	Lazio	Bayern	1	4	Stadio Olimpico	ter	23/02	17:00	Oitavas de final
10	10	Bayern	Lazio	2	1	Allianz Arena	qua	17/03	17:00	Oitavas de final
11	11	Atlético de Madrid	Chelsea	0	1	Arena Nacionala	ter	23/02	17:00	Oitavas de final
12	12	Chelsea	Atlético de Madrid	2	0	Stamford Bridge	qua	17/03	17:00	Oitavas de final
13	13	Borussia Mönchengladbach	Manchester City	0	2	Puskas Stadium	qua	24/02	17:00	Oitavas de final
14	14	Manchester City	Borussia Mönchengladbach	2	0	Puskas Stadium	ter	16/03	17:00	Oitavas de final
15	15	Atalanta	Real Madrid	0	1	Atleti Azzurri d'Italia	qua	24/02	17:00	Oitavas de final
16	16	Real Madrid	Atalanta	3	1	Alfredo Di Stefano	ter	16/03	17:00	Oitavas de final
17	17	Manchester City	Borussia Dortmund	2	1	Etihad Stadium	ter	06/04	16:00	Quartas de finais
18	18	Borussia Dortmund	Manchester City	1	2	Signal Iduna Park	qua	14/04	16:00	Quartas de finais
19	19	Real Madrid	Liverpool	3	1	Alfredo Di Stefano	ter	06/04	16:00	Quartas de finais
20	20	Liverpool	Real Madrid	0	0	Anfield Road	qua	14/04	16:00	Quartas de finais
21	21	Porto	Chelsea	0	2	Ramón Sánchez Pizjuán	qua	07/04	16:00	Quartas de finais
22	22	Chelsea	Porto	0	1	Ramón Sánchez Pizjuán	ter	13/04	16:00	Quartas de finais
23	23	Bayern	Paris Saint Germain	2	3	Allianz Arena	qua	07/04	16:00	Quartas de finais
24	24	Paris Saint Germain	Bayern	0	1	Parc des Princes	ter	13/04	16:00	Quartas de finais
25	25	Real Madrid	Chelsea	1	1	Alfredo Di Stefano	ter	27/04	16:00	Semifinal
26	26	Chelsea	Real Madrid	2	0	Stamford Bridge	qua	05/05	16:00	Semifinal
27	27	Paris Saint Germain	Manchester City	1	2	Parc des Princes	qua	28/04	16:00	Semifinal
28	28	Manchester City	Paris Saint Germain	2	0	Etihad Stadium	ter	04/05	16:00	Semifinal
29	29	Manchester City	Chelsea	0	1	Estádio do Dragão	sáb	29/05	16:00	Final

Figura 9. Tabela dos Dados dos jogos

	nsc player	nsc team	nsc position_player	123 games	123 games_starts	123 minutes	123 minutes_90s	123 goals_against	123 goals_against_per90	123 shots_on_target	123 saves
1	Keylor Navas	Paris Saint Germain	GK	12	12	1,080	12	1.25	1.25	65	52
2	Manuel Neuer	Bayern Munich	GK	8	8	720	8	8	1	43	35
3	Edouard Mendy	Chelsea	GK	12	12	1,080	12	3	0.25	33	30
4	Agustin Marchesin	Porto	GK	9	9	840	9.3	9	0.96	35	26
5	Thibaut Courtois	Real Madrid	GK	12	12	1,080	12	14	1.17	39	26
6	José Sá	Olympiacos	GK	6	6	540	6	10	1.67	31	24

Figura 10. Tabela dos Dados dos Goleiros

Os dados coletados sobre os jogadores da defesa no código **Stats_Defensor** e **Stats_Defensor_2** foram colocados na mesma tabela por possuírem as mesma colunas e pela análise serem executadas da mesma forma.

	nsc player	nsc team	nsc position_player	123 games	123 games_starts	123 minutes	123 minutes_90s	123 tackle	123 tackle_won	123 tackle_def	123 tackle_mid
1	Lucas Hernández	Bayern Munich	DF	10	7	696	7.7	29	17	20	7
2	Fabinho	Liverpool	DF,MF	8	7	554	6.2	28	16	18	9
3	Cristian Romero	Atalanta	DF	7	7	625	6.9	28	15	18	10
4	Casemiro	Real Madrid	MF	10	9	828	9.2	28	17	13	13
5	Kyle Walker	Manchester City	DF	11	11	981	10.9	28	20	12	12
6	Stefan Lainer	Borussia Mönchengladbach	DF	8	8	683	7.6	25	14	12	9
7	Marquinhos	Paris Saint Germain	DF,MF	10	10	815	9.1	18	11	10	7
8	André Herrera	Paris Saint Germain	MF	10	5	539	6	18	12	6	9
9	Ben Chilwell	Chelsea	DF	10	9	773	8.6	19	13	10	6
10	Jorginho	Chelsea	MF	12	12	1,028	11.4	25	18	10	11
11	Jude Bellingham	Borussia Dortmund	MF	10	8	687	7.6	25	15	10	9

Figura 11. Tabela dos Dados dos Zagueiros e Volantes

Os dados coletados referentes aos códigos **Stats_Strikers** e **Stats_Assists** foram implementados na mesma tabela pois a estrutura de análise realizada será a mesma, além de possuírem as mesmas colunas.

	ABC player	ABC team	ABC position_player	123 games	123 games_starts	123 minutes	123 minutes_90s	123 goals	123 assists	123 goals_per90	123 assists_per90
1	Erling Haaland	Borussia Dortmund	FW	8	8	703	7.8	10	2	1.28	0.26
2	Kylian Mbappé	Paris Saint Germain	FW	10	10	899	10	8	3	0.8	0.3
3	Marcus Rashford	Manchester Utd	FW	6	5	414	4.6	6	0	1.3	0
4	Neymar	Paris Saint Germain	FW,MF	9	9	743	8.3	6	2	0.73	0.24
5	Youssef En-Nesyri	Sevilla	FW	8	4	388	4.3	6	0	1.39	0
6	Mohamed Salah	Liverpool	FW	10	9	778	8.6	6	1	0.69	0.12
7	Juan Cuadrado	Juventus	DF	6	6	549	6.1	0	6	0	0.98
8	Ángel Di María	Paris Saint Germain	FW,MF	10	8	691	7.7	1	4	0.13	0.52
9	Kevin De Bruyne	Manchester City	MF,FW	8	8	665	7.4	3	4	0.41	0.54
10	Joshua Kimmich	Bayern Munich	MF	7	7	616	6.8	1	4	0.15	0.58
11	Jadon Sancho	Borussia Dortmund	FW,MF	6	4	387	4.3	2	3	0.47	0.7
12	Dušan Tadić	Ajax	FW,MF	6	6	540	6	2	3	0.33	0.5

Figura 12. Tabela dos Dados dos Meios Campistas e Atacantes

ABC player	ABC team	ABC position_player	123 games	123 games_starts	123 minutes	123 minutes_90s	123 goals	123 assists	123 goals_per90	123 assists_per90	123 tackle
Erling Haaland	Borussia Dortmund	FW	8	8	703	7.8	10	2	1.28	0.26	[NULL]
Kylian Mbappé	Paris Saint Germain	FW	10	10	899	10	8	3	0.8	0.3	[NULL]
Marcus Rashford	Manchester Utd	FW	6	5	414	4.6	6	0	1.3	0	[NULL]
Neymar	Paris Saint Germain	FW,MF	9	9	743	8.3	6	2	0.73	0.24	[NULL]
Youssef En-Nesyri	Sevilla	FW	8	4	388	4.3	6	0	1.39	0	[NULL]
Mohamed Salah	Liverpool	FW	10	9	778	8.6	6	1	0.69	0.12	[NULL]
Juan Cuadrado	Juventus	DF	6	6	549	6.1	0	6	0	0.98	[NULL]
Ángel Di María	Paris Saint Germain	FW,MF	10	8	691	7.7	1	4	0.13	0.52	[NULL]
Kevin De Bruyne	Manchester City	MF,FW	8	8	665	7.4	3	4	0.41	0.54	[NULL]
Joshua Kimmich	Bayern Munich	MF	7	7	616	6.8	1	4	0.15	0.58	[NULL]
Jadon Sancho	Borussia Dortmund	FW,MF	6	4	387	4.3	2	3	0.47	0.7	[NULL]
Dušan Tadić	Ajax	FW,MF	6	6	540	6	2	3	0.33	0.5	[NULL]
Lucas Hernández	Bayern Munich	DF	10	7	696	7.7	[NULL]	[NULL]	[NULL]	[NULL]	29
Fabinho	Liverpool	DF,MF	8	7	554	6.2	[NULL]	[NULL]	[NULL]	[NULL]	28
Cristian Romero	Atalanta	DF	7	7	625	6.9	[NULL]	[NULL]	[NULL]	[NULL]	28
Casemiro	Real Madrid	MF	10	9	828	9.2	[NULL]	[NULL]	[NULL]	[NULL]	28
Kyle Walker	Manchester City	DF	11	11	981	10.9	[NULL]	[NULL]	[NULL]	[NULL]	28
Stefan Lainer	Borussia Monchengladbach	DF	8	8	683	7.6	[NULL]	[NULL]	[NULL]	[NULL]	25
Marquinhos	Paris Saint Germain	DF,MF	10	10	815	9.1	[NULL]	[NULL]	[NULL]	[NULL]	18
Ander Herrera	Paris Saint Germain	MF	10	5	539	6	[NULL]	[NULL]	[NULL]	[NULL]	18
Ben Chilwell	Chelsea	DF	10	9	773	8.6	[NULL]	[NULL]	[NULL]	[NULL]	19
Jorginho	Chelsea	MF	12	12	1,028	11.4	[NULL]	[NULL]	[NULL]	[NULL]	25
Jude Bellingham	Borussia Dortmund	MF	10	8	687	7.6	[NULL]	[NULL]	[NULL]	[NULL]	25
Keylor Navas	Paris Saint Germain	GK	12	12	1,080	12	[NULL]	[NULL]	[NULL]	[NULL]	[NULL]
Manuel Neuer	Bayern Munich	GK	8	8	720	8	[NULL]	[NULL]	[NULL]	[NULL]	[NULL]
Edouard Mendy	Chelsea	GK	12	12	1,080	12	[NULL]	[NULL]	[NULL]	[NULL]	[NULL]
Agustín Marchesin	Porto	GK	9	9	840	9.3	[NULL]	[NULL]	[NULL]	[NULL]	[NULL]
Thibaut Courtois	Real Madrid	GK	12	12	1,080	12	[NULL]	[NULL]	[NULL]	[NULL]	[NULL]
José Sá	Olympicos	GK	6	6	540	6	[NULL]	[NULL]	[NULL]	[NULL]	[NULL]

Figura 13. Tabela com todos Jogadores

4. Análise dos Resultados

4.1. Análises - Gerais

4.1.1. Time Campeão

Como apresentado na imagem abaixo o Chelsea foi campeão do UEFA Champions League 2020/2021, ganhando na final do time do Manchester City por 1 a 0.

```
info_matches.tail(1)
```

	id_jogo	time_mandante	time_visitante	placar_mandante	placar_visitante	estadio_jogo	dia_semana	data_jogo	horario_jogo	etapa
28	29	Manchester City	Chelsea	0	1	Estádio do Dragão	sáb	29/05	16:00	Final

Figura 14. Dados da Final

4.1.2. Times que mais marcaram gols - Mandante e Visitante

Através dos dados abaixo percebe-se que o Manchester City foi o time com mais gols marcados. E percebe-se que o campeão da edição (Chelsea) marcou apenas 9 gols.

```
info_matches.groupby(['time_mandante'])['placar_mandante'].sum().nlargest(16)
```

```
time_mandante
Real Madrid      7
Manchester City  6
Bayern           4
Chelsea          4
Borussia Dortmund  3
Juventus         3
Liverpool        2
Paris Saint Germain  2
Porto            2
Sevilla          2
Barcelona        1
Lazio           1
Atalanta       0
Atlético de Madrid  0
Borussia Mönchengladbach  0
RB Leipzig       0
Name: placar_mandante, dtype: int64
```

Figura 15. Times Mandantes e Gols marcados


```
info_matches.groupby(['time_visitante'])['placar_visitante'].sum().nlargest(16)
```

```
time_visitante
Paris Saint Germain      7
Manchester City          6
Bayern                   5
Chelsea                  5
Borussia Dortmund       4
Liverpool                3
Porto                    3
Sevilla                  2
Atalanta                1
Barcelona                 1
Juventus                  1
Lazio                     1
Real Madrid               1
Atlético de Madrid       0
Borussia Mönchengladbach  0
RB Leipzig                0
Name: placar_visitante, dtype: int64
```

Figura 16. Times Visitantes e Gols marcados

4.1.3. Artilheiros do Campeonato

Os artilheiros são os caras do campeonato. Aqueles que balançam as redes. Como na resolução dos dados abaixo, Haaland(Dortmund) foi o artilheiro do campeonato com 10 gols.

```
atk_mid.groupby(['player', 'position_player'])['goals'].sum().nlargest(6)
```

```
player      position_player  goals
Erling Haaland      FW         10.0
Kylian Mbappé      FW          8.0
Marcus Rashford     FW          6.0
Mohamed Salah       FW          6.0
Neymar              FW,MF        6.0
Youssef En-Nesyri   FW          6.0
Name: goals, dtype: float64
```

Figura 17. Artilheiros

4.1.4. Jogadores com mais minutos em campo

```
all_players.groupby(['player', 'position_player'])['minutes'].sum().nlargest(5)
```

player	position_player	minutes
Edouard Mendy	GK	1080.0
Keylor Navas	GK	1080.0
Thibaut Courtois	GK	1080.0
Jorginho	MF	1028.0
Kyle Walker	DF	981.0

Name: minutes, dtype: float64

Figura 18. Jogares com mais minutagens

4.1.5. Jogadores com mais divididas ganhas

```
In [170]: df.groupby(['player', 'position_player'])['tackle_won'].sum().nlargest(5)
```

player	position_player	tackle_won
Kyle Walker	DF	20.0
Jorginho	MF	18.0
Casemiro	MF	17.0
Lucas Hernández	DF	17.0
Fabinho	DF, MF	16.0

Name: tackle_won, dtype: float64

Figura 19. Jogares com mais divididas ganhas

4.1.6. Jogadores com mais intercepções

```
df.groupby(['player', 'position_player'])['interception'].sum().nlargest(5)
```

player	position_player	interception
Jorginho	MF	21.0
Fabinho	DF, MF	17.0
Casemiro	MF	16.0
Cristian Romero	DF	16.0
Ben Chilwell	DF	14.0

Name: interception, dtype: float64

Figura 20. Jogares com mais intercepções

4.1.7. Jogadores com mais bloqueios

```
df.groupby(['player', 'position_player'])['blocks'].sum().nlargest(5)
```

player	position_player	blocks
Ander Herrera	MF	27.0
Marquinhos	DF,MF	27.0
Ben Chilwell	DF	26.0
Jorginho	MF	26.0
Casemiro	MF	25.0

Name: blocks, dtype: float64

Figura 21. Jogares com mais bloqueios

4.1.8. Jogadores com mais assistências

```
atk_mid.groupby(['player', 'position_player'])['assists'].sum().nlargest(6)
```

player	position_player	assists
Juan Cuadrado	DF	6.0
Joshua Kimmich	MF	4.0
Kevin De Bruyne	MF,FW	4.0
Ángel Di María	FW,MF	4.0
Dušan Tadić	FW,MF	3.0
Jadon Sancho	FW,MF	3.0

Name: assists, dtype: float64

Figura 22. Jogares com mais assistências

4.1.9. Análise Minuciosa - Goleiros

Através dos dados coletados pôde-se realizar diversas análises e a partir dessas foi possível entender algumas dinâmicas com os times e jogadores da UEFA Champions League na Temporada de 2020/2021. Abaixo segue algumas análises realizadas:

O goleiro é uma das peças mais fundamentais, uma peça única em campo, onde faz toda diferença em um jogo de futebol. Os goleiros costumam impactar as partidas, sejam por defesas espetaculares ou falhas, impactando diretamente nos resultados dos jogos. Conforme os dados coletados dos 6 melhores goleiros da competição foi realizada uma análise, onde foi descoberto informações bem importantes.

O primeiro passo foi identificar qual goleiro sofreu mais gols, e a partir do gráfico abaixo percebe-se que o goleiro que mais levou gol foi o Keylor Navas(PSG) com 15 gols sofridos e o que menos sofreu gol foi o Mendy(Chelsea) com 3, ou seja, Navas tomou gol 5 vezes a mais que Mendy.

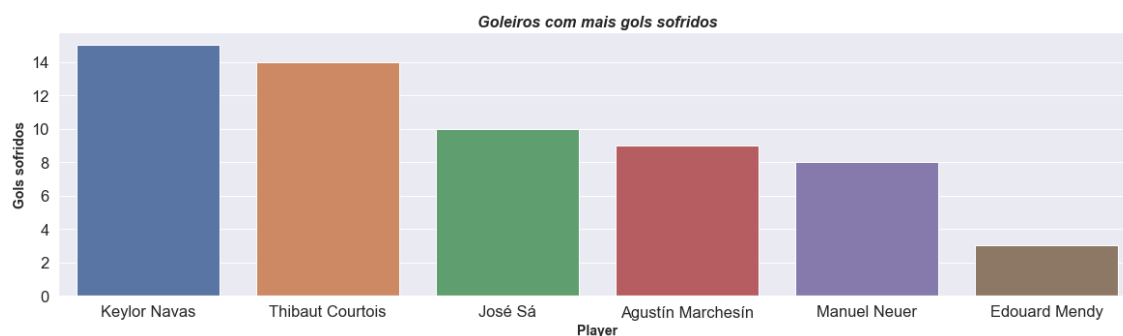


Figura 23. Tabela com todos Jogadores

Em seguida foi analisado a quantidade de jogos de ambos jogadores para entender a diferença de valores em relação ao gráfico anterior e percebe-se que ambos possuem o mesmo número de partidas, 12.

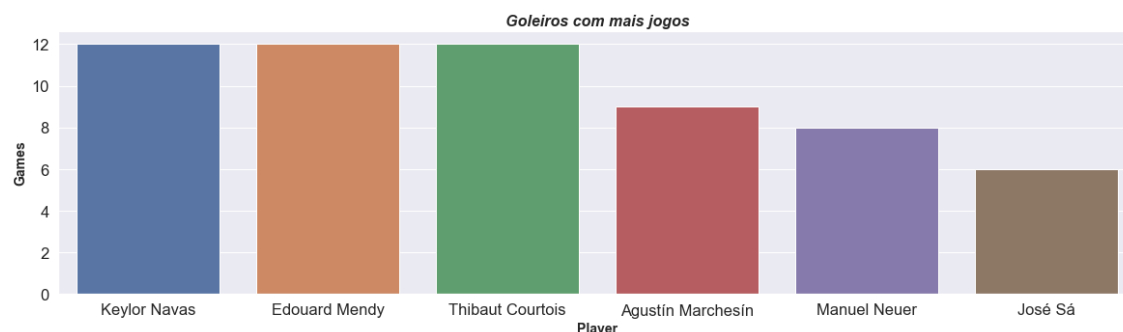


Figura 24. Tabela com quantidade de jogos

Ainda em busca do porquê em relação as discrepância de gols sofridos, foi analisado a quantidade de chutes que cada goleiro sofreu, sendo Keylor Navas (PSG) recebendo 65 chutes no gol e Mendy(Chelsea) recebendo 33 chutes no gol, ou seja Navas recebeu praticamente o dobro de chutes que Mendy.

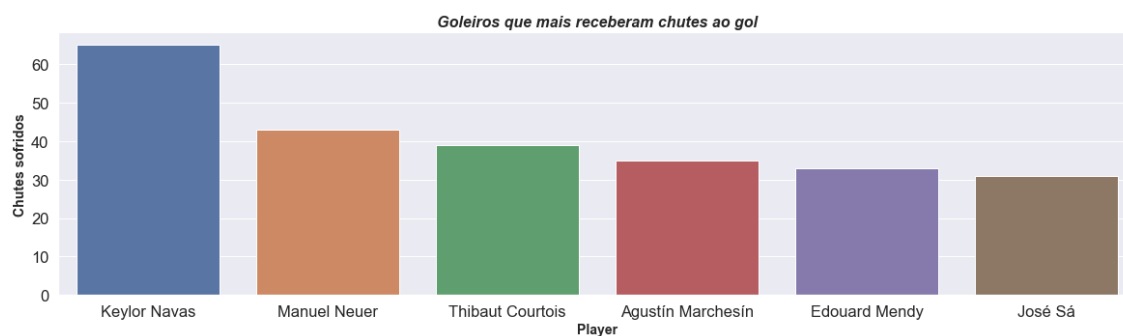


Figura 25. Quantidade de chute recebidos

Através de entender mais os dados apresentados foi realizado um gráfico para ver quais goleiros defendem mais e o percentual de defesas realizadas e percebe-se que Navas é o que possui mais defesas e Mendy possui o maior percentual de defesas realizadas, onde o percentual é dado por **(chutes no gol - gols sofridos) / chutes no gol**

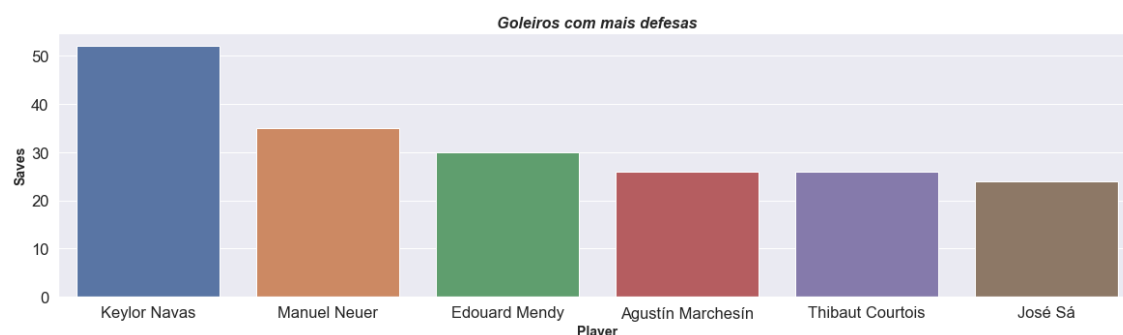


Figura 26. Gráfico quantidade de defesas

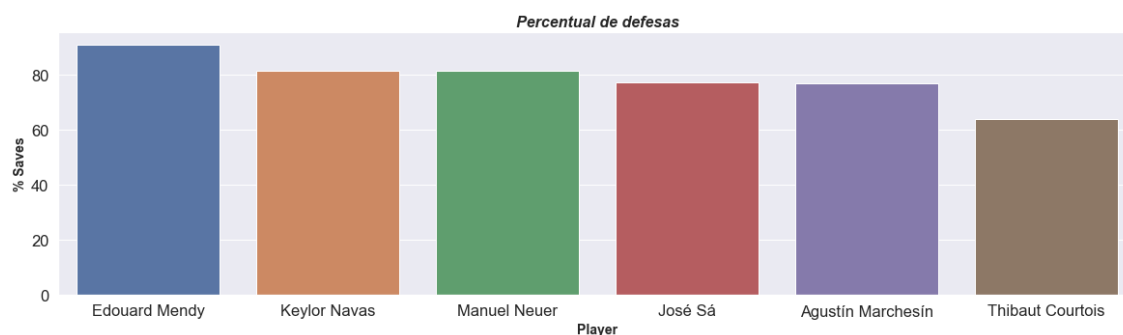


Figura 27. Gráfico percentual de defesas

No gráfico de percentual de defesas percebe-se uma pequena diferença entre Mendy e Navas, mesmo com Navas sofrendo 5 vezes mais gols que Mendy e recebendo 2 vez mais a quantidade de chutes, acredito que Navas tenha tido um melhor desempenho considerando que ele teve que realizar mais defesas e ainda sim tendo uma diferença de porcentagem mínima em relação ao percentual de defesas. E sobre Mendy pode-se afirmar que é um goleiro que quase não recebe chutes ao gol e quando é acionado responde bem defendendo aos chutes.

Considerando também os times enfrentados por ambos durante a Champions League pode-se, afirmar que os times enfrentados pelo PSG de Navas são times que atacam mais que os times que enfrentaram o Chelsea de Mendy.



Figura 28. Mendy e Navas

4.2. Time da Competição

Considerando cada característica a cada posição e a cada jogador, foi possível montar o time da competição, onde possui os melhores jogadores de acordo com os dados descritivos. As características levadas em conta para cada posição foram:

- **Goleiro** : O que mais realizou defesas
- **Zagueiros/Laterais/Volantes**: O que mais realizou interceptações, bloqueios, divididas e assistências.
- **Meio-Campos**: Quem mais realizou assistências e gols.
- **Atacantes**: Os artilheiros da competição.

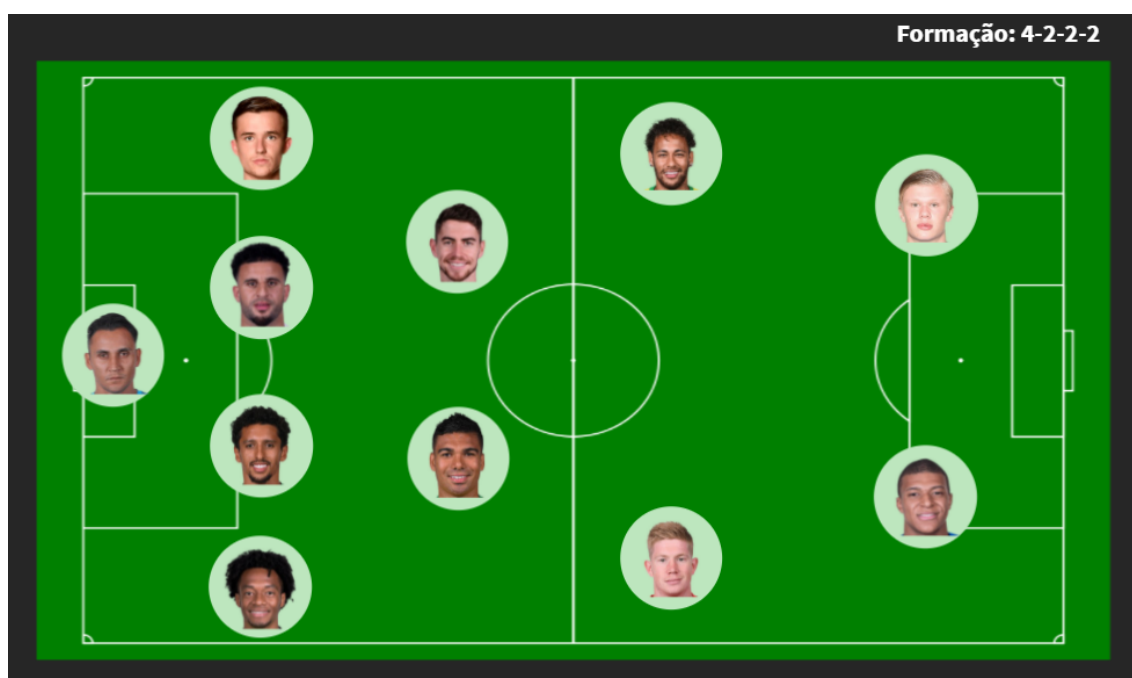


Figura 29. Time da Competição

Segue abaixo o nome dos jogadores em cada posição:

- **Goleiro** : Keylor Navas
- **Lateral Esquedo** : Ben Chilwell
- **Zagueiro Esquerdo**: Kyle Walker e Marquinho
- **Lateral Direito** : Cuadrado
- **Volantes**: Casemiro e Jorginho
- **Meio-Campos**: Neymar e Kevin De Bruyne
- **Atacantes**: Mbappe e Haaland

4.3. Curiosidades

- Neymar é o único meio campo na lista dos artilheiros da UEFA Champions League 2020/2021
- Na competição total o PSG marcou 22 gols, só Neymar e Mbappé marcaram 14
- A questão em relação ao goleiro Mendy do Chelsea receber menos chutes é por causa da formação dada pelo time, onde o time possui 3 zagueiros, diferente do PSG de Navas que conta apenas com 2 zagueiros como percebe-se na ilustração abaixo.

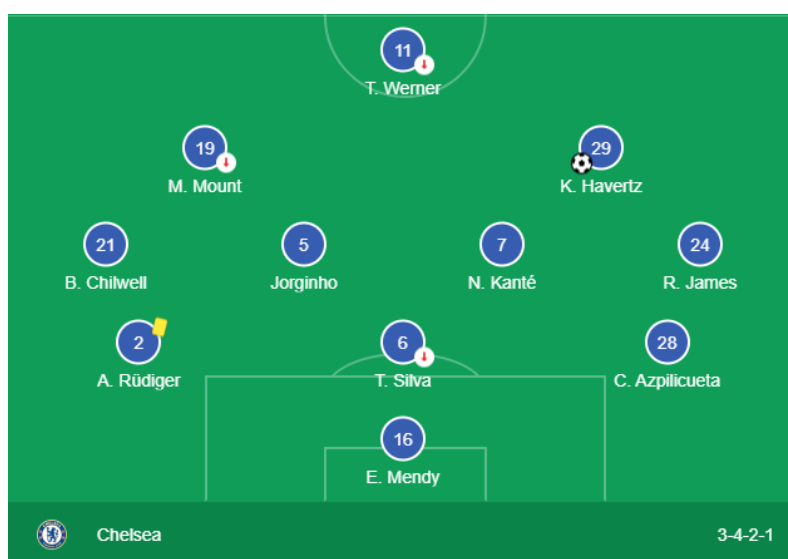


Figura 30. Formação Chelsea

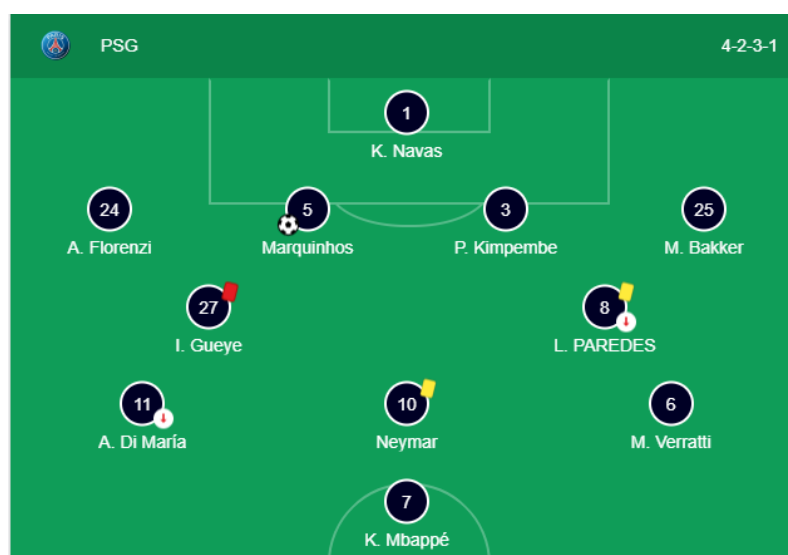


Figura 31. Formação PSG

5. Conclusões

A partir das análises realizadas percebe-se que há uma discrepância em ser o melhor jogador e o jogador com as melhores estatísticas, pois percebemos que os jogadores do time campeão (Chelsea) quase não aparecem nas principais estatísticas, logo mais evidenciando a coletividade de um time vencedor, onde as estatísticas estão distribuídas pela equipe inteira. Através das análises realizadas acima pode-se ver a dimensão e o impacto que jogadores de alto nível apresentam na maior competição Europeia, e percebe-se o qual grande é a importância da UEFA Champions League.



Figura 32. Logo UEFA Champions League

5.1. Relato da Experiência

Através do projeto foi possível colocar em prática diversos usos de conhecimentos necessário para um cientista de dado, como: programação em linguagem Python, realizar Web Scrapping, formatar os dados, fazer a ligação da linguagem Python com banco de dado, realizar a estruturação de um banco de dados e por fim realizar uma análise com os dados obtidos. Como eu não possuía experiência em Web Scrapping foi muito interessante ver o desenvolvimento do projeto, desde o básico até a coleta completa dos dados. A partir do projeto pôde-se ter uma ideia maior do conhecimento sobre coleta de dados com Web Scrapping e sua utilização para projetos futuros.

6. Referências

NETSHOES

Você Sabe O Que É A UEFA Champions League? Netshoes.com.br.

Disponível em: <<https://www.netshoes.com.br/blog/esportes/post/voce-sabe-o-que-e-a-uefa-champions-league>>.

Acesso em: 24 Jun. 2021.

Gazeta Esportiva

Tabela da Liga dos Campeões 2020-2021 - Gazeta Esportiva.

Disponível em: <<https://www.gazetaesportiva.com/campeonatos/uefa-champions-league-2020/>>.

Acesso em: 24 Jun. 2021.

FBref.com

FBref.com

Disponível em: <<https://fbref.com/en/>>.

Acesso em: 24 Jun. 2021.

HERBERT, Anthony.

Como começar a usar a biblioteca Requests em Python. DigitalOcean.

Disponível em: <<https://www.digitalocean.com/community/tutorials/how-to-get-started-with-the-requests-library-in-python-pt>>.

Acesso em: 24 Jun. 2021.

TECLA SAP.

Posições dos jogadores de futebol (em inglês e português). Tecla SAP.

Disponível em: <<https://www.teclasap.com.br/posicoes-dos-jogadores-de-futebol/>>.

Acesso em: 24 Jun. 2021.

UEFA.

Liga dos Campeões.UEFA

Disponível em: <https://pt.uefa.com/uefachampionsleague/>

Acesso em: 24 Jun. 2021.

LUIS FELIPE BUENO

Como fazer Web Scrapping em Python - Data Hackers - Medium. Medium.

Disponível em: <<https://medium.com/data-hackers/como-fazer-web-scrapping-em-python-23c9d465a37f>>.

Acesso em: 24 Jun. 2021.

HIEMATH, Omkar S.

Web Scrapping With Python - A Beginner's Guide | Edureka. Edureka.

Disponível em: <<https://www.edureka.co/blog/web-scrapping-with-python/>>.

Acesso em: 24 Jun. 2021.