METODOLOGIA

13 DE MAIO DE 2024

FURIA – PROCESSO SELETIVO

FERRAMENTA UTILIZADA:

O Jupyter Notebook foi a ferramenta utilizada para se fazer a análise da base de dados disponibilizada pela Furia em seu processo seletivo.

Essa escolha foi realizada devido à proximidade com o Python, mais de 5 anos de experiência com ela. Além de sua versatilidade, devido as inúmeras bibliotecas internas e as criadas pelos próprios usuários, ela é uma ferramenta que consegue destrinchar uma base de dados com facilidade e depois fazer uma limpeza bem feita.

PRÉ-ANÁLISE:

Transformação do arquivo:
Para se utilizar a biblioteca pandas no Python, foi mais fácil abrir um arquivo do Excel (.xlsx) em branco, depois clicar em “Dados” => “Obter Dados” => “De Arquivo” => “De Texto/CSV”. Depois o arquivo foi salvo como .xlsx.

VERIFICAR BIBLIOTECAS:

Bibliotecas utilizadas:
Como o Python tem uma comunidade Open Source muito forte, existem várias bibliotecas utilizadas pelos usuários que não vem junto do software. No caso dessa análise, é necessário baixar as bibliotecas: pandas, missingno e matplolib.

Imports:

In [1]:
```
import pandas as pd
import missingno as msno
import numpy as np
import matplotlib.pyplot as plt
import random
```

INSERIR BASE DE DADOS:

Inserir base de dados no Jupyter Notebook:

```
dataFrame = pd.read_excel("Furia_dataBase.xlsx")
```

OLHADA NA BASE:

```
1 dataFrame.columns
```

```
Index(['Entity', 'Username', 'Medium', 'Medium ID', 'Media Type', 'Date',
       'Variety', 'Engagement', 'Likes, Reactions, +1's', 'Comments, Replies',
       'Shares', 'Reposts, Retweets', 'Reach', 'Impressions',
       'Estimated Impressions', 'Paid Impressions', 'Paid Video Views',
       'Views', 'Avg. Time Watched (sec)', 'Total Time Watched (sec)',
       'Post Valuation in USD', 'Follower Count'],
      dtype='object')
```

PRÉ-ANÁLISE:

```
1 dataFrame.describe()
```

	Engagement	Likes, Reactions, +1's	Comments, Replies	Shares	Reposts, Retweets	Reach	Impressions	Estimated Impressions	Paid Impressions	Paid Video Views	
count	3.461700e+04	3.461700e+04	34617.000000	34617.000000	34617.000000	3.461700e+04	3.461700e+04	3.461700e+04	34617.000000	34617.000000	3.4
mean	2.620143e+03	2.301438e+03	34.358870	12.660571	88.007251	1.353425e+03	3.698535e+04	6.322528e+04	31.148713	7.338360	1.6
std	4.313073e+04	3.896178e+04	684.402212	1068.775191	1639.834883	2.419332e+04	2.186291e+05	4.387296e+05	2130.121762	765.606734	1.9
min	0.000000e+00	0.000000e+00	0.000000	0.000000	0.000000	0.000000e+00	0.000000e+00	0.000000e+00	0.000000	0.000000	0.0
25%	4.000000e+00	3.000000e+00	0.000000	0.000000	0.000000	0.000000e+00	0.000000e+00	4.960000e+02	0.000000	0.000000	0.0
50%	7.700000e+01	7.000000e+01	1.000000	0.000000	0.000000	0.000000e+00	0.000000e+00	9.667000e+03	0.000000	0.000000	0.0
75%	7.080000e+02	5.940000e+02	9.000000	0.000000	12.000000	0.000000e+00	2.341700e+04	4.416800e+04	0.000000	0.000000	0.0
max	6.387589e+06	6.369418e+06	116872.000000	182168.000000	182168.000000	3.586367e+06	2.314775e+07	5.097296e+07	287562.000000	133547.000000	2.3

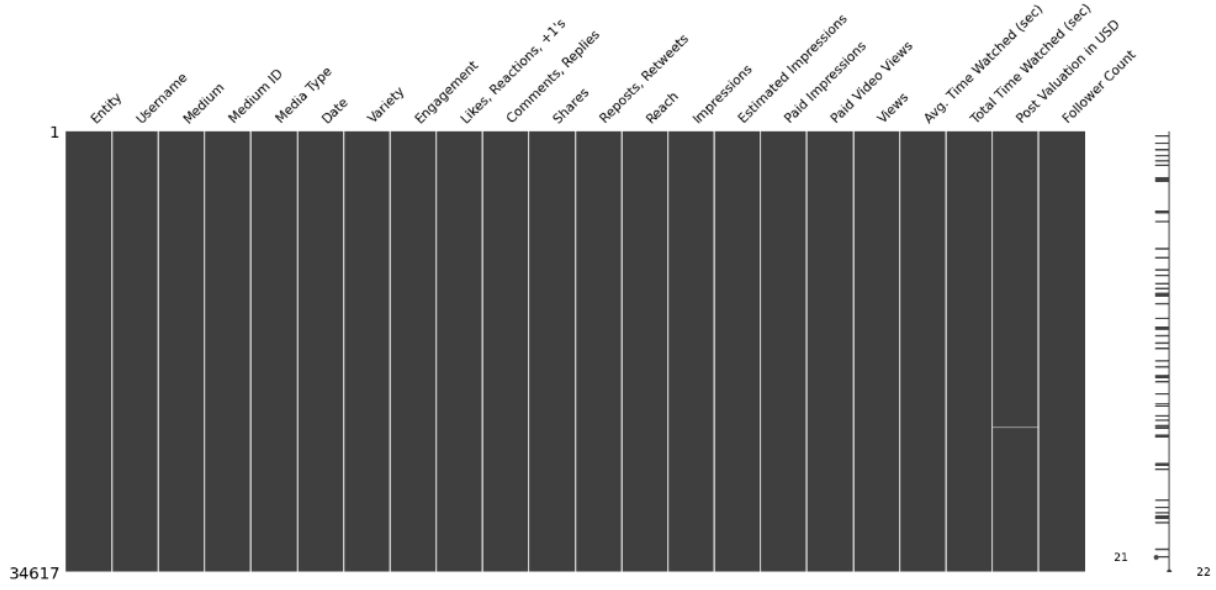
Primeiras impressões:
Alguns números são bem grandes, aparência de bastante notação científica. Grande presença de zeros na base de dados.

LIMPEZA DA BASE DE DADOS:

Bibliotecas utilizadas:
Utilizando a biblioteca missingno é possível tornar visível a busca por valores nulos ou NaN's, note que na matriz abaixo a linha em branco na coluna "Post Valuation in USD" demonstra a presença de NaN's.

```
1 msno.matrix(dataFrame)
```

```
<AxesSubplot: >
```



PRÉ-ANÁLISE:

```
In [8]: 1 dataFrame.isna().sum()

Out[8]: Entity                      0
        Username                    0
        Medium                      0
        Medium ID                   0
        Media Type                  0
        Date                        0
        Variety                     0
        Engagement                  0
        Likes, Reactions, +1's      0
        Comments, Replies           0
        Shares                      0
        Reposts, Retweets           0
        Reach                       0
        Impressions                 0
        Estimated Impressions       0
        Paid Impressions            0
        Paid Video Views            0
        Views                       0
        Avg. Time Watched (sec)     0
        Total Time Watched (sec)    0
        Post Valuation in USD      62
        Follower Count              0
        dtype: int64
```

Conclusão da análise:
Sim, existem valores nulos na Base de Dados, e eles são 62 no total. Todos estão presentes na coluna “Post Valuation in USD”.

Retirar os NaN

```
1 dataFrame = dataFrame.dropna()
2 # Verificando se não existe mais NaN
3 dataFrame.isna().sum()
```

LIMPEZA DA BASE DE DADOS:

Código para remoção de valores nulos:
Com a função .dropna() é possível retirar todos os valores nulos dentro de uma Base de Dados. Note que é necessário atualizar o dataFrame quando essa operação é realizada.

PRÉ-ANÁLISE:

Verificar se tem duplicatas

```
1 print(f"Existem duplicatas: {dataFrame.duplicated().any()}")
2 print(f"O número total de duplicatas o dataFrame é igual a: {dataFrame.duplicated().sum()}")
```

```
Existem duplicatas: True
O número total de duplicatas o dataFrame é igual a: 182
```

Verificação de Duplicatas:
É muito comum existirem valores repetidos dentro de uma base de dados, com a função .duplicated().any() é possível verificar se existe valores duplicados dentro da base de dados. Com o atributo .sum() é possível somar todos as duplicatas.

```
dataFrame = dataFrame.drop_duplicates()
```

Verificação de Duplicatas:
Para se remover os valores duplicados dentro de uma base de dados, é possível utilizar a função .drop_duplicates() presente no pandas.DataFrame.

INÍCIO DA ANÁLISE DE DADOS:

Quais colunas estão presentas na base de dados:

```
1 dataFrame.columns
```

```
Index(['Entity', 'Username', 'Medium', 'Medium ID', 'Media Type', 'Date',
       'Variety', 'Engagement', 'Likes, Reactions, +1's', 'Comments, Replies',
       'Shares', 'Reposts, Retweets', 'Reach', 'Impressions',
       'Estimated Impressions', 'Paid Impressions', 'Paid Video Views',
       'Views', 'Avg. Time Watched (sec)', 'Total Time Watched (sec)',
       'Post Valuation in USD', 'Follower Count'],
      dtype='object')
```

Qual o tipo de cada coluna:

```
1 dataFrame.dtypes
```

```
Entity                        object
Username                      object
Medium                        object
Medium ID                     object
Media Type                    object
Date                          object
Variety                       object
Engagement                     int64
Likes, Reactions, +1's         int64
Comments, Replies              int64
Shares                         int64
Reposts, Retweets              int64
Reach                          int64
Impressions                    int64
Estimated Impressions          int64
Paid Impressions               int64
Paid Video Views               int64
Views                          int64
Avg. Time Watched (sec)        int64
Total Time Watched (sec)       int64
Post Valuation in USD        float64
Follower Count                 int64
dtype: object
```

INÍCIO DA ANÁLISE DE DADOS:

Transformar Date em Datetime:
Para se fazer análises temporais é necessários transformar o tipo da coluna “Date” para que ele se torne um padrão internacional reconhecido de data.

```
1 dataFrame['Date'] = pd.to_datetime(dataFrame['Date'], dayfirst=True)
```

Transformar Date em Datetime:
Nomes grandes e complexos como nome de colunas geralmente causam erros na hora das análises serem realizadas. Por precaução, mudou-se o nome da coluna para “Likes”.

Mudar nome gigante do Likes + Reactions para somente Likes:

- facilitar o processo de análise de dados

```
1 dataFrame = dataFrame.rename(columns={"Likes, Reactions, +1's": "Likes"})
```

Trocar código de usuários por usuários genéricos:
Os nomes de usuários eram muito grandes e complexos, ter que ficando digitando os diferentes nomes seria uma grande perda de tempo. Por exemplo, o usuário `'caa58707a4d577a912067bc202c2f48d3a7883688d74d4d548d5b9d9d153a7ef'` se tornou “Usuario1”. Note que não colocar acentos nesse caso é algo proposital, devido o Python ser em Inglês.

Dicionário com o nome antigo associado ao nome novo:
Foi criado um dicionário (estrutura de armazenamento de dados no Python) para armazenar o usuário antigo e a sua nova nomenclatura. Assim, caso necessário o nome pode ser revertido sem quaisquer problemas.

O mesmo processo foi repetido com as entidades.
Tanto a substituição de nomes quanto o armazenamento em um dicionário também foi realizado com as entidades.

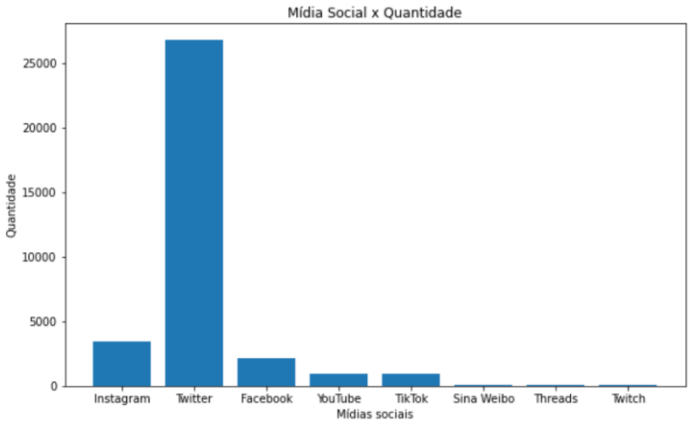
ANÁLISE DE DADOS:

Agora que a base de dados foi preparada, não será mais necessário a explicação de códigos.

Análise de quantas postagens foram feitas em cada Rede Social:

Twitter fica em primeiro lugar (26784), Instagram em segundo (3410) e Facebook em terceiro.

```
O meio Instagram tem 3410 postagens
O meio Twitter tem 26784 postagens
O meio Facebook tem 2135 postagens
O meio YouTube tem 947 postagens
O meio TikTok tem 953 postagens
O meio Sina Weibo tem 55 postagens
O meio Threads tem 60 postagens
O meio Twitch tem 29 postagens
```



Mídia Social:

É esperado que o Twitter tenha um número de textos muito maior do eu imagens ou vídeos.

```
A mídia social analizada é: Twitter
            Medium
Media Type
image         9878
mixed            2
photo            2
text         12447
video         4455
```

No Instagram não há como postar mídias em texto, e é esperado um equilíbrio entre imagens e vídeos.

```
A mídia social analizada é: Instagram
            Medium
Media Type
image         1607
mixed           46
video         1757
```

FUNÇÕES: FACILITAR O MANUSEIO:

Analisar Entidades por postagem em mídias sociais

In [46]:

```python
def comparar_dois_a_dois(dataFrame, coluna1, coluna2):

    dataFrame_teste = dataFrame

    analise_para_dropar = list(dataFrame_teste.columns)

    realmente_dropar = []

    for strings in analise_para_dropar:
        if strings == coluna1 or strings == coluna2:
            pass
        else:
            realmente_dropar.append(strings)

    dataFrame_teste = dataFrame_teste.drop(realmente_dropar,axis=1)

    for midias_sociais in list(dataFrame_teste[coluna1].unique()):
        print(f"A mídia social analizada é: {midias_sociais}")
        print(dataFrame_teste[dataFrame_teste[coluna1]==midias_sociais].groupby(coluna2).count())
```

Análise separada por Entidades

```python
def org_x_usuario_metrica_geral(dataFrame, nomeEntidade, redeSocial):

    dataFrame_entidade = dataFrame[dataFrame['Entity']==nomeEntidade]

    dataFrame_entidade_midiaSocial = dataFrame_entidade[dataFrame['Medium']==redeSocial]
    dataFrame_entidade_midiaSocial = dataFrame_entidade_midiaSocial.drop(["Entity","Medium","Medium ID",
                                                                          "Media Type","Date","Variety"],axis=1)

    dataFrame_entidade_midiaSocial = dataFrame_entidade_midiaSocial.groupby("Username").sum()

    return dataFrame_entidade_midiaSocial
```

FUNÇÕES: FACILITAR O MANUSEIO:

```python
def entidade_engajamento_por_usuario(dataFrame, nomeEntidade):

    import matplotlib.pyplot as plt

    plt.figure(figsize=(25,6))

    dataFrame = dataFrame[dataFrame['Entity']==nomeEntidade]
    dataFrame = dataFrame[["Username","Engagement"]]
    dataFrame = dataFrame.groupby("Username").sum().sort_values(by="Engagement",ascending=True)

    plt.title(f"{nomeEntidade}: Username x Engagement")
    plt.xlabel("Usuarios")
    plt.ylabel("Quantidade")
    plt.bar(dataFrame.index, dataFrame['Engagement'])
    plt.show()
```

```python
def usuario_x_tempo_engajamento(dataFrame, nomeUsuario):

    dataFrame = dataFrame[dataFrame["Username"]==nomeUsuario]
    dataFrame = dataFrame[["Username","Date","Engagement"]].drop("Username",axis=1)
    dataFrame = dataFrame.set_index("Date").sort_values(by="Date",ascending=True)

    delta_Engagement = [0]
    valores_Engajamento_sequencia = list(dataFrame['Engagement'])

    contador = 1
    while contador < len(valores_Engajamento_sequencia):

        dif = valores_Engajamento_sequencia[contador] - valores_Engajamento_sequencia[contador-1]
        delta_Engagement.append(dif)

        contador = contador + 1

    dataFrame['delta_Engagement'] = delta_Engagement

    return dataFrame
```

FUNÇÕES: FACILITAR O MANUSEIO:

```python
def entidade_x_tempo_engajamento(dataFrame, nomeEntidade):

    dataFrame = dataFrame[dataFrame["Entity"]==nomeEntidade]
    dataFrame = dataFrame[["Entity","Date","Engagement"]].drop("Entity",axis=1)
    dataFrame = dataFrame.set_index("Date").sort_values(by="Date",ascending=True)

    delta_Engagement = [0]
    valores_Engajamento_sequencia = list(dataFrame['Engagement'])

    contador = 1
    while contador < len(valores_Engajamento_sequencia):

        dif = valores_Engajamento_sequencia[contador] - valores_Engajamento_sequencia[contador-1]
        delta_Engagement.append(dif)

        contador = contador + 1

    dataFrame['delta_Engagement'] = delta_Engagement

    return dataFrame
```

```python
def dataFrame_followerCount_usuario_grafico(dataFrame, nomeUsuario):

    dataFrame_FollowerCount = dataFrame[["Entity","Username","Follower Count","Date","Medium"]].sort_values("Date").set_inde

    dataFrame_FollowerCount = dataFrame_FollowerCount[dataFrame_FollowerCount["Username"]==nomeUsuario]


    print(f"As mídias sociais disponíveis para análise são: {list(dataFrame_FollowerCount['Medium'].unique()) }")
    midiaSelecionada = input("Qual mídia social deverá ser analisada? ")

    dataFrame_FollowerCount = dataFrame_FollowerCount[dataFrame_FollowerCount["Medium"]==midiaSelecionada]

    plt.figure(figsize=(16,9))

    plt.plot(dataFrame_FollowerCount.index, dataFrame_FollowerCount['Follower Count'])
    plt.xlabel("Tempo")
    plt.ylabel(f"{nomeUsuario}")
    plt.title(f"{nomeUsuario} x Tempo")

    plt.show()
```

FUNÇÕES: FACILITAR O MANUSEIO:

```
def dataFrame_followerCount_entidade_grafico(dataFrame, nomeEntidade):

    dataFrame_FollowerCount = dataFrame[["Entity","Username","Follower Count","Date","Medium"]].sort_values("Date").set_inde

    dataFrame_FollowerCount = dataFrame_FollowerCount[dataFrame_FollowerCount["Entity"]==nomeEntidade]


    print(f"Os usuários disponíveis para consulta são: {list(dataFrame_FollowerCount['Username'].unique())}")
    nomeUsuario = input("Qual usuário você quer consultar? ")

    dataFrame_FollowerCount = dataFrame_FollowerCount[dataFrame_FollowerCount['Username']==nomeUsuario]

    print(f"As mídias sociais disponíveis para análise são: {list(dataFrame_FollowerCount['Medium'].unique()) }")
    midiaSelecionada = input("Qual mídia social deverá ser analisada? ")


    dataFrame_FollowerCount = dataFrame_FollowerCount[dataFrame_FollowerCount["Medium"]==midiaSelecionada]


    plt.figure(figsize=(16,9))

    plt.plot(dataFrame_FollowerCount.index, dataFrame_FollowerCount['Follower Count'])
    plt.xlabel("Tempo")
    plt.ylabel(f"{nomeUsuario}")
    plt.title(f"({nomeEntidade} + {midiaSelecionada}) = {nomeUsuario} x Tempo")

    plt.show()
```

```
def engagementMax_usuario_redeSocial(dataFrame, redeSocial):

    dataMedia = dataFrame
    dataMedia = dataMedia[dataMedia["Medium"]==redeSocial]
    dataMedia = dataMedia[["Date","Username","Medium","Engagement"]].set_index("Date")
    maximo_engagement = dataMedia["Engagement"].max()
    dataMedia = dataMedia[dataMedia["Engagement"]==maximo_engagement]

    return dataMedia
```

```
def likesMax_mediumType_redeSocial(dataFrame, redeSocial):

    dataMedia = dataFrame
    dataMedia = dataMedia[dataMedia["Medium"]==redeSocial]
    dataMedia = dataMedia[["Date","Username","Medium","Medium ID","Likes"]].set_index("Date")
    maximo_engagement = dataMedia["Likes"].max()
    dataMedia = dataMedia[dataMedia["Likes"]==maximo_engagement]

    return dataMedia
```

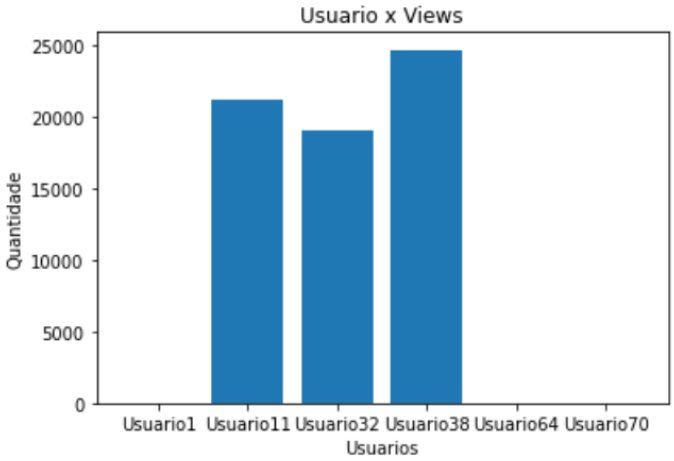
FUNÇÕES: FACILITAR O MANUSEIO:

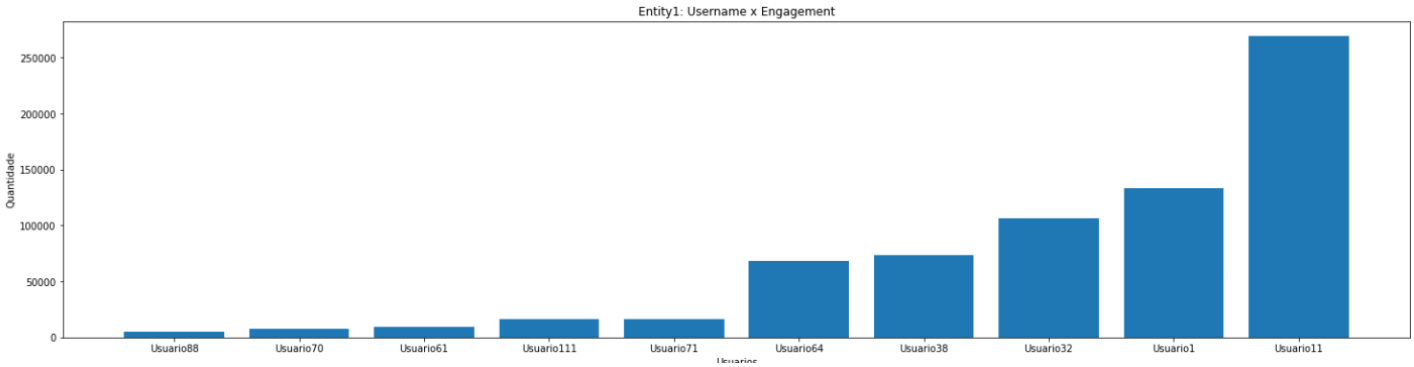
```python
def metrica_minMax_redeSocial(dataFrame, redeSocial, metrica, maxMin):

    dataMedia = dataFrame
    dataMedia = dataMedia[dataMedia["Medium"]==redeSocial]
    dataMedia = dataMedia.set_index("Date")

    if maxMin.lower() == "max":
        maximo_metrica = dataMedia[metrica].max()
        print(f"Máximo {metrica} = {maximo_metrica}")

        dataMedia = dataMedia[dataMedia[metrica]==maximo_metrica]

        dataMedia = dataMedia[["Entity","Username",metrica,"Medium ID"]]

    elif maxMin.lower() == "min":
        min_metrica = dataMedia[metrica].min()
        print(f"Mínimo {metrica} = {min_metrica}")

        dataMedia = dataMedia[dataMedia[metrica]==min_metrica]

        dataMedia = dataMedia[["Entity","Username",metrica,"Medium ID"]]

    else:
        dataMedia = "Escreva min ou max da próxima vez!"


    return dataMedia
```

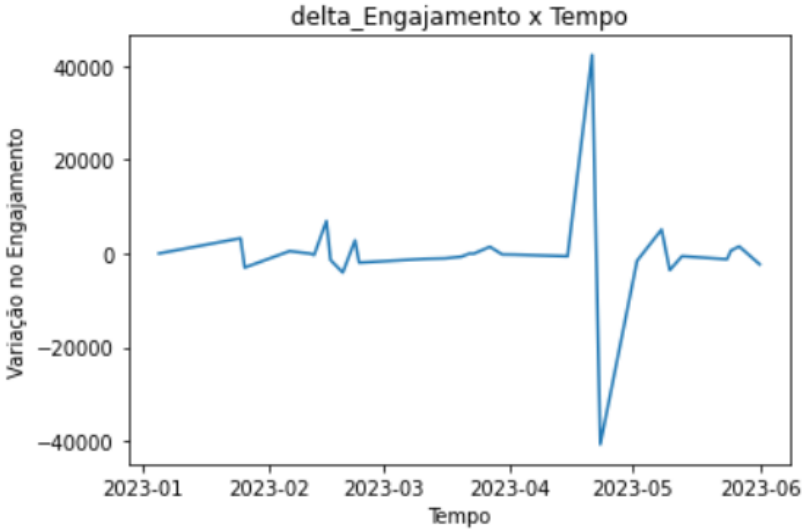
```python
def entidade_metrica_redeSocial(dataFrame, redeSocial, metrica):

    dataMedia = dataFrame
    dataMedia = dataMedia[dataMedia["Medium"]==redeSocial]

    dataMedia = dataMedia.set_index("Date")
    dataMedia = dataMedia.groupby(by="Entity").sum().sort_values(by=metrica,ascending=True)

    plt.figure(figsize=(16,9))

    plt.xlabel("Entidades")
    plt.ylabel(f"{metrica}")
    plt.title(f"Entidades x {metrica}")

    plt.scatter(dataMedia.index, dataMedia[metrica])

    plt.show()
```

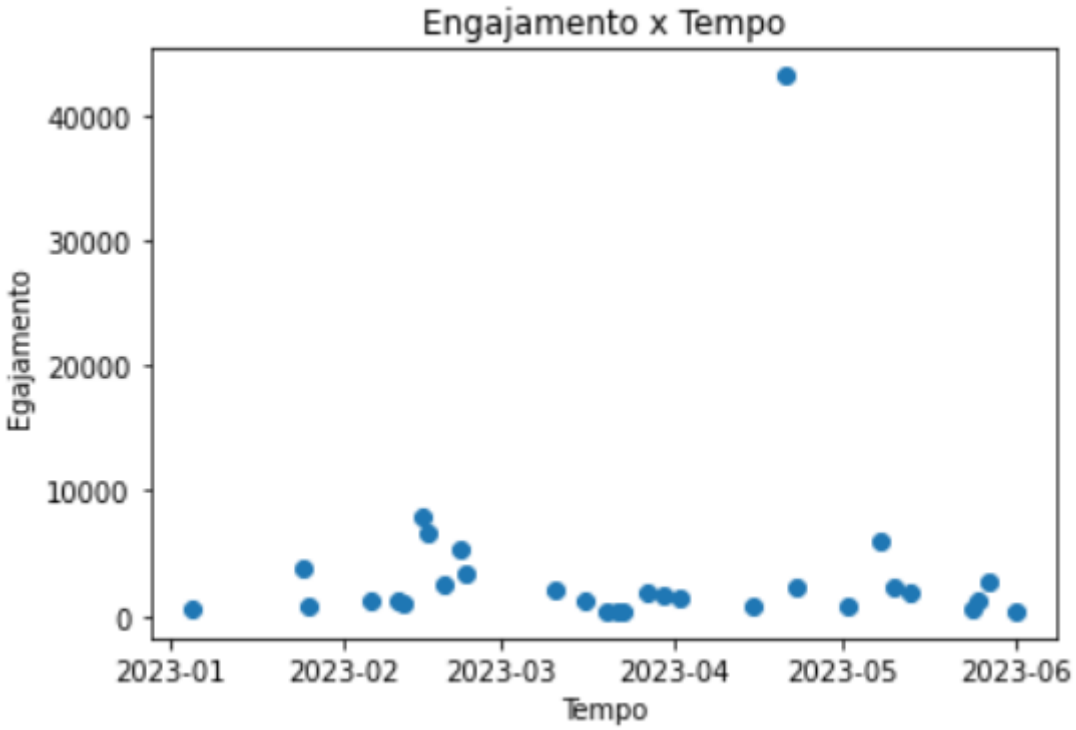
CONCLUSÕES:

```
dataFrame_teste = tipoPostagem_x_entidade_x_usuario(dataFrame,"Entity1","image")
plt.bar(dataFrame_teste.index, dataFrame_teste['Views'])
plt.title("Usuario x Views")
plt.xlabel("Usuarios")
plt.ylabel("Quantidade")
plt.show()
```



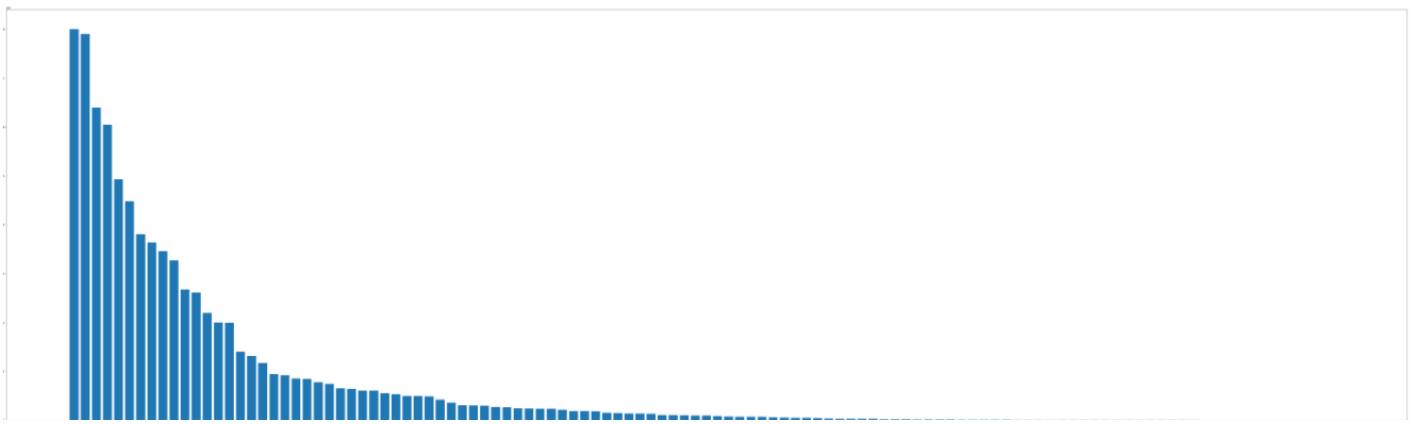







```
1 plt.figure(figsize=(100,30))
2 plt.bar(dataFrame[['Username','Engagement']].groupby('Username').sum().sort_values(by='Engagement',ascending=False).index, d
```

```
<BarContainer object of 115 artists>
```



CONCLUSÕES:

Qual entidade tem maior engajamento?

Entity	Engagement
Entity11	16588154

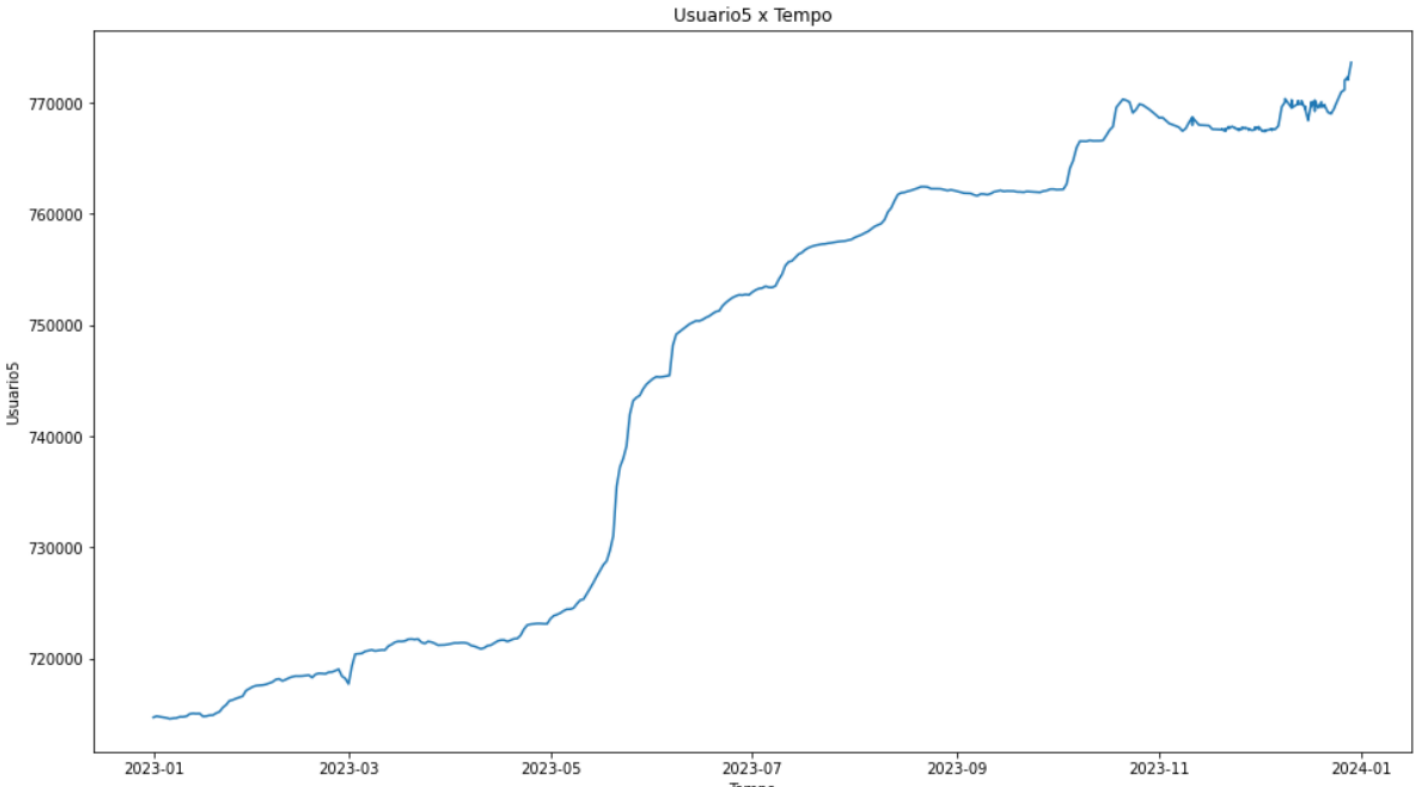
Qual entidade tem menor engajamento?

Entity	Engagement
Entity1	705811

Possibilidade de traçar uma análise temporal por usuário x métrica:

```
1 dataFrame_followerCount_usuario_grafico(dataFrame, "Usuario5")
```

```
As mídias sociais disponíveis para análise são: ['Twitter']
Qual mídia social deverá ser analisada? Twitter
```



CONCLUSÕES:

Qual usuário teve maior número de Followers no Twitter?

Date	Username	Medium	Follower Count
2023-01-22	Usuario22	Twitter	5793314

Qual usuário teve maior engajamento no Twitter?

Date	Username	Medium	Engagement
2023-06-07	Usuario5	Twitter	115910

Qual post teve maior número de likes no Twitter?

Date	Username	Medium	Medium ID	Likes
2023-06-07	Usuario5	Twitter	47c0eda6a885c1086fd9d64e4ac483d5b3fb616b8052e0...	100873

Qual entidade teve o maior número de Shares no Instagram?

```
Máximo Shares = 182168
```

Date	Entity	Username	Shares	Medium ID
2023-10-05	Entity10	Usuario30	182168	4271f6d48d8dcae90a55444334d41f94ec852f6e3a4933...

CONCLUSÕES:

Qual entidade teve o maior Engagement no Twitter?

```
1 entidade_metrica_redeSocial(dataFrame, "Twitter","Engagement")
```