



```
In [130]: 1 url_resumo_mundial = 'https://raw.githubusercontent.com/alura-cursos/IA-produtividade-DS/main/Dados/Res'
2 url_producao = 'https://github.com/alura-cursos/IA-produtividade-DS/raw/main/Dados/Producao_total.csv'

In [131]: 1 import pandas as pd
2
3 # Importar o dataset de resumo mundial
4 resumo_mundial = pd.read_csv(url_resumo_mundial, parse_dates=['data'])
5
6 # Importar o dataset de produção
7 producao = pd.read_csv(url_producao)

In [132]: 1 # Exibir as primeiras 5 Linhas do dataframe
2 resumo_mundial.head()

data producao_total_de_combustiveis_fosseis producao_de_energia_nuclear producao_total_de_energia_renovavel produc
0 1973-01-31 4.932632 0.068103 0.403981 5.404715
1 1973-02-28 4.729582 0.064634 0.360900 5.155115
2 1973-03-31 4.946902 0.072494 0.400161 5.419556
3 1973-04-30 4.716271 0.064070 0.380470 5.160812
4 1973-05-31 4.956995 0.062111 0.392141 5.411246

In [133]: 1 # Exibir as últimas 5 linhas do dataframe
2 resumo_mundial.tail()

data producao_total_de_combustiveis_fosseis producao_de_energia_nuclear producao_total_de_energia_renovavel produc
594 2022-07-31 6.921426 0.718109 1.132400 8.7715
595 2022-08-31 7.047525 0.718526 1.044026 8.8100
596 2022-09-30 6.915201 0.664673 0.978647 8.5585
597 2022-10-31 7.126618 0.614741 1.019209 8.7605
598 2022-11-30 6.875368 0.647029 1.097519 8.6195

In [134]: 1 # Exibir as estatísticas descritivas do dataframe
2 resumo_mundial.describe()

data producao_total_de_combustiveis_fosseis producao_de_energia_nuclear producao_total_de_energia_renovavel produc
count 599 599.000000 599.000000 599.000000
mean 1997-12-30 5.034634 0.519567 0.593709
min 1973-01-31 3.676065 0.062111 0.304328
25% 1985-07-15 4.683559 0.328635 0.467414
50% 1997-12-31 4.831601 0.594293 0.527479
75% 2010-06-15 5.087384 0.681056 0.685252
max 2022-11-30 7.126618 0.780456 1.218790
std NaN 0.610126 0.202697 0.193351

In [135]: 1 # Exibir o tipo de dados de cada coluna do dataframe
2 resumo_mundial.dtypes

data producao_total_de_combustiveis_fosseis datetime64[ns]
producao_de_energia_nuclear float64
producao_total_de_energia_renovavel float64
producao_total_de_energia_primaria float64
importacoes_de_energia_primaria float64
exportacoes_de_energia_primaria float64
importacoes_liquidas_de_energia_primaria float64
variacao_nas_reservas_de_energia_primaria_e_outros float64
consumo_total_de_combustiveis_fosseis float64
consumo_de_energia_nuclear float64
consumo_total_de_energia_renovavel float64
consumo_total_de_energia_primaria float64
dtype: object

In [136]: 1 # Exibir o número de linhas e colunas do dataframe
2 resumo_mundial.shape

(599, 13)

In [137]: 1 # Exibir os nomes das colunas do dataframe
2 resumo_mundial.columns

Index(['data', 'producao_total_de_combustiveis_fosseis',
       'producao_de_energia_nuclear', 'producao_total_de_energia_renovavel',
       'producao_total_de_energia_primaria', 'importacoes_de_energia_primaria',
       'exportacoes_de_energia_primaria',
       'importacoes_liquidas_de_energia_primaria',
       'variacao_nas_reservas_de_energia_primaria_e_outros',
       'consumo_total_de_combustiveis_fosseis', 'consumo_de_energia_nuclear',
       'consumo_total_de_energia_renovavel',
       'consumo_total_de_energia_primaria'],
      dtype='object')
```

```
In [138]: 1 # Exibir os valores únicos de cada coluna do dataframe
2 resumo_mundial.unique()

data producao_total_de_combustiveis_fosseis
producao_de_energia_nuclear
producao_total_de_energia_renovavel
producao_total_de_energia_primaria
importacoes_de_energia_primaria
exportacoes_de_energia_primaria
importacoes_liquidas_de_energia_primaria
variacao_nas_reservas_de_energia_primaria_e_outros
consumo_total_de_combustiveis_fosseis
consumo_de_energia_nuclear
consumo_total_de_energia_renovavel
consumo_total_de_energia_primaria
dtype: int64
```

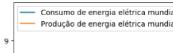
```
In [139]: 1 # Exibir a correlação entre as colunas do dataframe
2 resumo_mundial.corr()
```

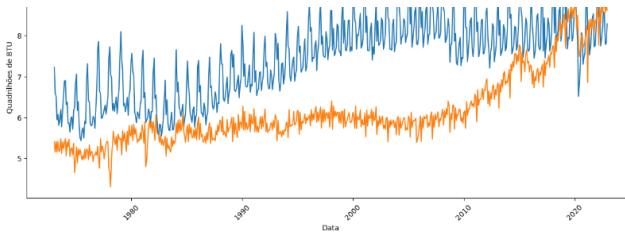
	data	producao_total_de_combustiveis_fosseis	producao_de_energia_nuclear
data	1.000000	0.660553	0.901802
producao_total_de_combustiveis_fosseis	0.660553	1.000000	0.411684
producao_de_energia_nuclear	0.901802	0.411684	1.000000
producao_total_de_energia_renovavel	0.843404	0.851062	0.699987
producao_total_de_energia_primaria	0.836674	0.958719	0.638849
importacoes_de_energia_primaria	0.647269	0.051540	0.779024
exportacoes_de_energia_primaria	0.782521	0.925480	0.515903
importacoes_liquidas_de_energia_primaria	-0.086538	-0.655291	0.215655
variacao_nas_reservas_de_energia_primaria_e_outros	0.083126	-0.005999	0.165886
consumo_total_de_combustiveis_fosseis	0.537073	0.189429	0.886630
consumo_de_energia_nuclear	0.901802	0.411684	1.000000
consumo_total_de_energia_renovavel	0.845541	0.847192	0.613568
consumo_total_de_energia_primaria	0.766414	0.401573	0.852501

```
In [140]: 1 import seaborn as sns
2
3 # Obter o dataframe de correlação
4 correlacao = resumo_mundial.corr()
5
6 # Visualizar a correlação em uma figura
7 sns.heatmap(correlacao, annot=True, cmap="RdYlGn")
```

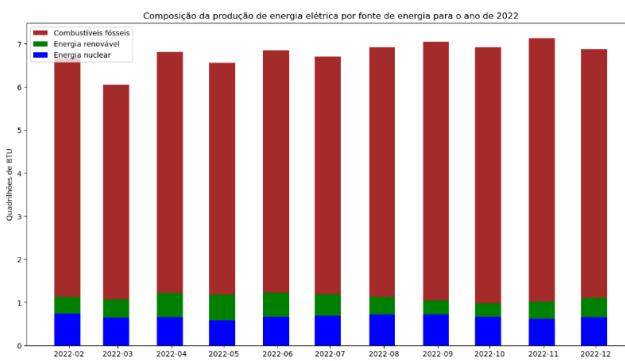
<Axes: >

Produção e consumo de energia elétrica mundial





```
In [142]: 1 # Criar um novo dataframe com apenas os dados do ano de 2022
2 resumo_2022 = resumo_mundial[resumo_mundial["data"].dt.year == 2022]
3
4 # Criar um gráfico de barras empilhadas
5 plt.figure(figsize=(15, 8))
6 plt.bar(resumo_2022["data"], resumo_2022["producao_total_de_combustiveis_fosseis"], color="brown", label="Combustíveis fósseis")
7 plt.bar(resumo_2022["data"], resumo_2022["producao_total_de_energia_renovavel"], color="green", label="Energia renovável")
8 plt.bar(resumo_2022["data"], resumo_2022["producao_de_energia_nuclear"], color="blue", label="Energia nuclear")
9
10 # Adicionar uma Legenda
11 plt.legend(fontsize="small")
12
13 # Adicionar um título e labels para eixo X e Y
14 plt.title("Composição da produção de energia elétrica por fonte de energia para o ano de 2022")
15 plt.xlabel("Data")
16 plt.ylabel("Quadrilhões de BTU")
17
18 # Adicionar um fundo na caixa de Legenda
19 plt.legend(frameon=True)
20
21 # Exibir o gráfico
22 plt.show()
```



```
In [143]: 1 # Exibir as primeiras 5 Linhas do dataframe
2 producao.head()
```

	continente	país	1980	1981	1982	1983	1984	1985	1986	1987	...	2013	2014	2015
0	África	Argélia	2.803017	3.037537	3.224934	3.606400	3.859176	3.907466	3.968325	4.218088	...	6.510418	6.561750	6.696193
1	África	Angola	0.335098	0.291294	0.276262	0.395400	0.462550	0.511269	0.621433	0.789429	...	3.932204	3.802528	3.949947
2	África	Benin	0.000000	0.000000	0.000000	0.008960	0.015730	0.017930	0.017930	0.015690	...	0.000019	0.000019	0.000056
3	África	Botswana	0.008262	0.008485	0.009242	0.008797	0.008752	0.009732	0.010912	0.010845	...	0.033335	0.038241	0.046562
4	África	Burkina Faso	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	...	0.001107	0.000981	0.000988

5 rows × 45 columns

```
In [144]: 1 # Exibir as últimas 5 linhas do dataframe
2 producao.tail()
```

	continente	país	1980	1981	1982	1983	1984	1985	1986	1987	...	2013	2014	2015
224	América Central e do Sul	Trinidad e Tobago	0.550035	0.612443	0.502478	0.484613	0.575515	0.610607	0.528431	0.494887	...	1.783162	1.754012	1.64
225	América Central e do Sul	Ilhas Turks e Caicos	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	...	0.000000	0.000000	0.00
226	América Central e do Sul	Ilhas Virgens Americanas	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	...	0.000012	0.000068	0.00
227	América Central e do Sul	Uruguai	0.023363	0.026342	0.025309	0.074850	0.073425	0.066694	0.075389	0.074871	...	0.094341	0.118105	0.12
228	América Central e do Sul	Venezuela	5.766784	5.667510	5.250659	5.032015	5.056074	4.817756	5.214788	5.171222	...	7.539512	7.347768	7.41

5 rows × 45 columns

```
In [145]: 1 # Exibir as estatísticas descritivas do dataframe
2 producao.describe()
```

	1980	1981	1982	1983	1984	1985	1986	1987	1988	1989	...	
count	229.000000	229.000000	229.000000	229.000000	229.000000	229.000000	229.000000	229.000000	229.000000	229.000000	...	229.0
mean	1.294072	1.272025	1.266935	1.279684	1.349113	1.382063	1.426948	1.460257	1.517144	1.546011	...	2.418
std	6.397011	6.389251	6.392847	6.352637	6.716241	6.796403	6.924452	7.056841	7.279238	7.307238	...	10.35
min	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	...	0.000
25%	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	...	0.000
50%	0.001787	0.001589	0.002154	0.002632	0.002975	0.003040	0.003489	0.003310	0.003428	0.003463	...	0.049
75%	0.244687	0.286900	0.276262	0.314713	0.321407	0.378639	0.364744	0.344538	0.380542	0.421517	...	0.722
max	67.146595	66.909508	66.526732	65.401473	68.799586	70.310670	72.985060	75.034245	77.680706	77.119582	...	113.4

8 rows × 43 columns

```
In [146]: 1 # Exibir o tipo de dados de cada coluna do dataframe
2 producao.dtypes

continentente    object
pais            object
1980           float64
1981           float64
1982           float64
1983           float64
1984           float64
1985           float64
1986           float64
1987           float64
1988           float64
1989           float64
1990           float64
1991           float64
1992           float64
1993           float64
1994           float64
1995           float64
1996           float64
1997           float64
1998           float64
1999           float64
2000           float64
2001           float64
2002           float64
2003           float64
2004           float64
2005           float64
2006           float64
2007           float64
2008           float64
2009           float64
2010           float64
2011           float64
2012           float64
2013           float64
2014           float64
2015           float64
2016           float64
2017           float64
2018           float64
2019           float64
2020           float64
2021           float64
producao_total   float64
dtype: object
```

```
In [147]: 1 # Exibir o número de linhas e colunas do dataframe
2 producao.shape
```

(229, 45)

```
In [148]: 1 # Exibir os nomes das colunas do dataframe
2 producao.columns
```

```
Index(['continente', 'pais', '1980', '1981', '1982', '1983', '1984', '1985',
       '1986', '1987', '1988', '1989', '1990', '1991', '1992', '1993', '1994',
       '1995', '1996', '1997', '1998', '1999', '2000', '2001', '2002', '2003',
       '2004', '2005', '2006', '2007', '2008', '2009', '2010', '2011', '2012',
       '2013', '2014', '2015', '2016', '2017', '2018', '2019', '2020', '2021',
       'producao_total'],
      dtype='object')
```

```
In [149]: 1 # Exibir os valores únicos de cada coluna do dataframe
2 producao.unique()
```

```
continente    7
pais          229
1980          135
1981          136
1982          137
1983          139
1984          140
1985          142
1986          141
1987          144
1988          144
1989          144
1990          146
1991          143
1992          162
1993          166
1994          167
1995          167
1996          167
1997          169
1998          169
1999          172
2000          172
2001          174
2002          172
2003          176
2004          176
2005          180
2006          181
2007          183
2008          184
2009          185
2010          189
2011          193
2012          193
2013          192
2014          195
2015          197
2016          198
2017          201
2018          202
2019          203
2020          203
2021          203
producao_total 219
dtype: int64
```

```
In [152]: 1 # Excluir colunas categóricas (no caso, a coluna 'País')
2 producao_numeric = producao.select_dtypes(include=[float, int])
3
4 # Calcular a correlação
5 correlacao = producao_numeric.corr()
6
7 # Exibir a matriz de correlação
8 print(correlacao)
```

	1980	1981	1982	1983	1984	1985
1980	1.000000	0.950842	0.955162	0.950594	0.989672	0.983878
1981	0.998942	1.000000	0.996445	0.991951	0.990836	0.985825
1982	0.995162	0.996445	1.000000	0.990794	0.998259	0.995239
1983	0.999584	0.991951	0.998794	1.000000	0.999536	0.998516
1984	0.989672	0.990836	0.990259	0.999536	1.000000	0.999008
1985	0.983878	0.995825	0.995239	0.998516	0.999008	1.000000
1986	0.986419	0.987466	0.994747	0.998444	0.998448	0.999156

```

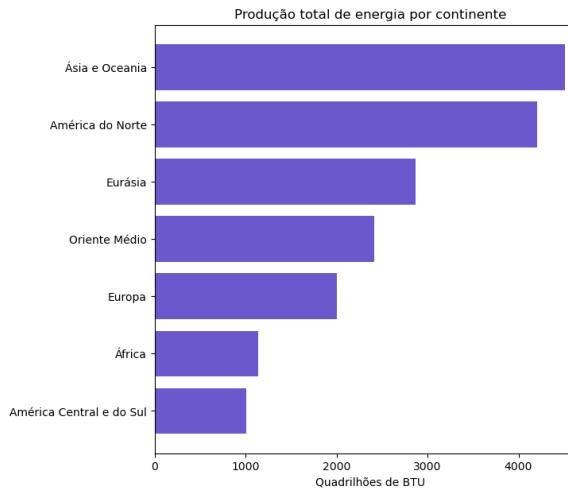
1987      0.983724  0.984424  0.993694  0.997575  0.997600  0.999015
1988      0.984873  0.985145  0.993317  0.996903  0.996852  0.998057
1989      0.983883  0.983708  0.991985  0.995761  0.996071  0.997424
1990      0.987122  0.987607  0.993563  0.995939  0.996115  0.996387
1991      0.983252  0.984913  0.987669  0.988258  0.988151  0.988935
1992      0.637319  0.635673  0.623068  0.609061  0.615399  0.605159
1993      0.645267  0.643288  0.638745  0.617270  0.623739  0.613951
1994      0.659897  0.657931  0.644709  0.631176  0.630805  0.628465
1995      0.662854  0.659986  0.647823  0.634888  0.641562  0.632457
1996      0.667908  0.665916  0.659116  0.641521  0.642872  0.639552
1997      0.671663  0.669262  0.657057  0.644224  0.651008  0.642047
1998      0.672572  0.669536  0.657224  0.644288  0.651095  0.642109
1999      0.666834  0.662655  0.651133  0.639988  0.645912  0.637521
2000      0.666459  0.657157  0.645229  0.633191  0.640125  0.632011
2001      0.654899  0.651006  0.649974  0.629747  0.636868  0.629775

```

```

In [153]: 
1 # Agrupar os dados por continente e calcular a soma da coluna "producao_total"
2 producao_total = producao.groupby("continente")["producao_total"].sum()
3
4 # Ordenar os dados do menor para o maior
5 producao_total = producao_total.sort_values()
6
7 # Criar um novo dataframe com os dados agrupados e ordenados
8 producao_total = pd.DataFrame(producao_total)
9
10 # Criar um gráfico de barras horizontal
11 plt.figure(figsize=(7, 7))
12 plt.barh(producao_total.index, producao_total["producao_total"], color="#6A58CC")
13
14 # Adicionar um título e labels para o eixo X
15 plt.title("Produção total de energia por continente")
16 plt.xlabel("Quadrilhões de BTU")
17
18 # Exibir o gráfico
19 plt.show()

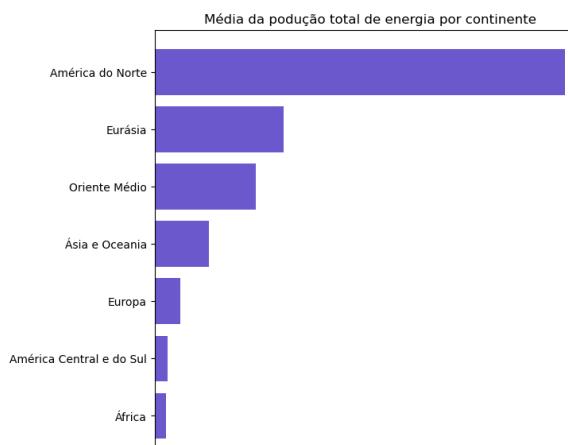
```



```

In [154]: 
1 # Agrupar os dados por continente e calcular a soma da coluna "producao_total"
2 producao_total = producao.groupby("continente")["producao_total"].mean()
3
4 # Ordenar os dados do menor para o maior
5 producao_total = producao_total.sort_values()
6
7 # Criar um novo dataframe com os dados agrupados e ordenados
8 producao_total = pd.DataFrame(producao_total)
9
10 # Criar um gráfico de barras horizontal
11 plt.figure(figsize=(7, 7))
12 plt.barh(producao_total.index, producao_total["producao_total"], color="#6A58CC")
13
14 # Adicionar um título e labels para o eixo X
15 plt.title("Média da produção total de energia por continente")
16 plt.xlabel("Quadrilhões de BTU")
17
18 # Exibir o gráfico
19 plt.show()

```





```
In [155]: 1 # Criar um novo dataframe com apenas os anos de "2007", "2008", "2009", "2010", "2011", "2012", "2013",
2 producao_anos = producao[["pais", "2007", "2008", "2009", "2010", "2011", "2012", "2013", "2014",
3
4 # Somar o total de energia produzida em cada país
5 producao_anos["producao_total"] = producao_anos[["2007", "2008", "2009", "2010", "2011", "2012", "2013"]
6
7 # Ordenar os dados pelo valor da coluna "producao_total" em ordem decrescente
8 producao_anos = producao_anos.sort_values("producao_total", ascending=False)
9
10 # Selecionar as primeiras 10 linhas do dataframe
11 producao_anos = producao_anos.head(10).round(2)
12
13 # Exibir o dataframe
14 producao_anos[['pais','producao_total']]
```

C:\Users\Public\Documents\Wondershare\CreatorTemp\ipykernel_17884\86943806.py:5: SettingWithCopyWarning:

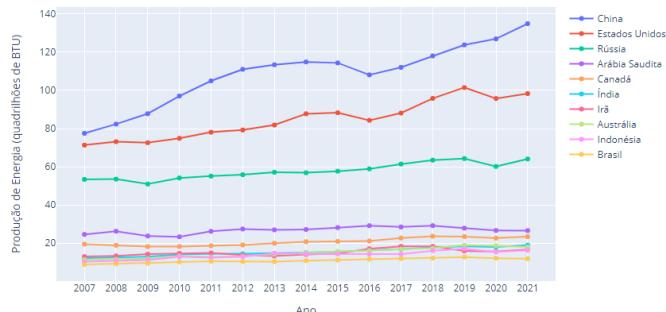
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy

país	producao_total
123 China	1627.55
183 Estados Unidos	1271.14
65 Rússia	867.13
174 Arábia Saudita	401.80
179 Canadá	311.06
130 Índia	231.95
165 Irã	227.89
117 Austrália	224.64
131 Indonésia	207.31
191 Brasil	163.53

```
In [156]: 1 import plotly.graph_objects as go
2
3 # 1. Criar um novo DataFrame com as colunas de interesse
4 colunas_anos = ["2007", "2008", "2009", "2010", "2011", "2012", "2013", "2014", "2015", "2016", "2017",
5 producao_pais = producao_anos[colunas_anos + ["pais"]]
6
7 # 2. Criar a plotagem usando plotly.graph_objects
8 fig = go.Figure()
9
10 for pais in producao_pais["pais"]:
11     dados_pais = producao_pais[producao_pais["pais"] == pais]
12     fig.add_trace(go.Scatter(
13         x=colunas_anos,
14         y=dados_pais[colunas_anos].values.tolist()[0],
15         mode="lines+markers",
16         name=pais
17     ))
18
19 fig.update_layout(
20     title="Histórico de Produção de Energia por País (2007-2021)",
21     xaxis_title="Ano",
22     yaxis_title="Produção de Energia (quadrilhões de BTU)",
23 )
24
25 fig.show()
26
```

Histórico de Produção de Energia por País (2007-2021)



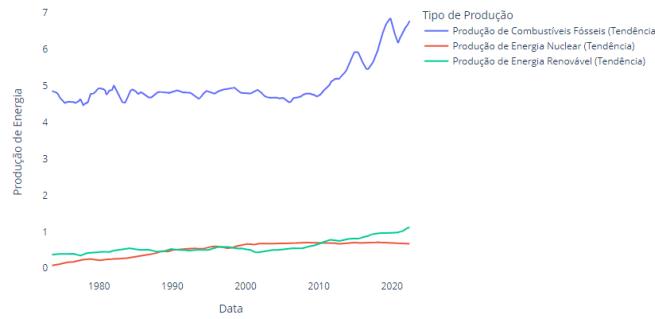
```
In [157]: 1 import statsmodels.api as sm
2
3 # 2. Criar um novo DataFrame com as colunas de interesse
4 producao_tipos = resumo_mundial[["data", "producao_total_de_combustiveis_fosseis", "producao_de_energia_nuclear",
5
6 # 3. Utilizar seasonal_decompose para decompor as séries temporais
7 producao_tipos.set_index("data", inplace=True)
8 decomposed_combustiveis = sm.tsa.seasonal_decompose(producao_tipos["producao_total_de_combustiveis_fosseis"])
9 decomposed_nuclear = sm.tsa.seasonal_decompose(producao_tipos["producao_de_energia_nuclear"], model='add')
10 decomposed_renovavel = sm.tsa.seasonal_decompose(producao_tipos["producao_total_de_energia_renovavel"],
11
12 # 4. Criar a plotagem usando plotly
13 fig = go.Figure()
14
```

```

15 fig.add_trace(go.Scatter(
16     x=producao_tipos.index,
17     y=decomposed_combustiveis.trend,
18     mode='lines',
19     name='Produção de Combustíveis Fósseis (Tendência)'
20 ))
21
22
23 fig.add_trace(go.Scatter(
24     x=producao_tipos.index,
25     y=decomposed_nuclear.trend,
26     mode='lines',
27     name='Produção de Energia Nuclear (Tendência)'
28 ))
29
30
31 fig.add_trace(go.Scatter(
32     x=producao_tipos.index,
33     y=decomposed_renovavel.trend,
34     mode='lines',
35     name='Produção de Energia Renovável (Tendência)'
36 ))
37
38 fig.update_layout(
39     title='Tendência de Produção de Energia por Tipo',
40     xaxis_title='Data',
41     yaxis_title='Produção de Energia',
42     legend_title='Tipo de Produção',
43     plot_bgcolor='white' # Definir o fundo do gráfico como branco
44 )
45
46 fig.show()

```

Tendência de Produção de Energia por Tipo



In [1]: 1