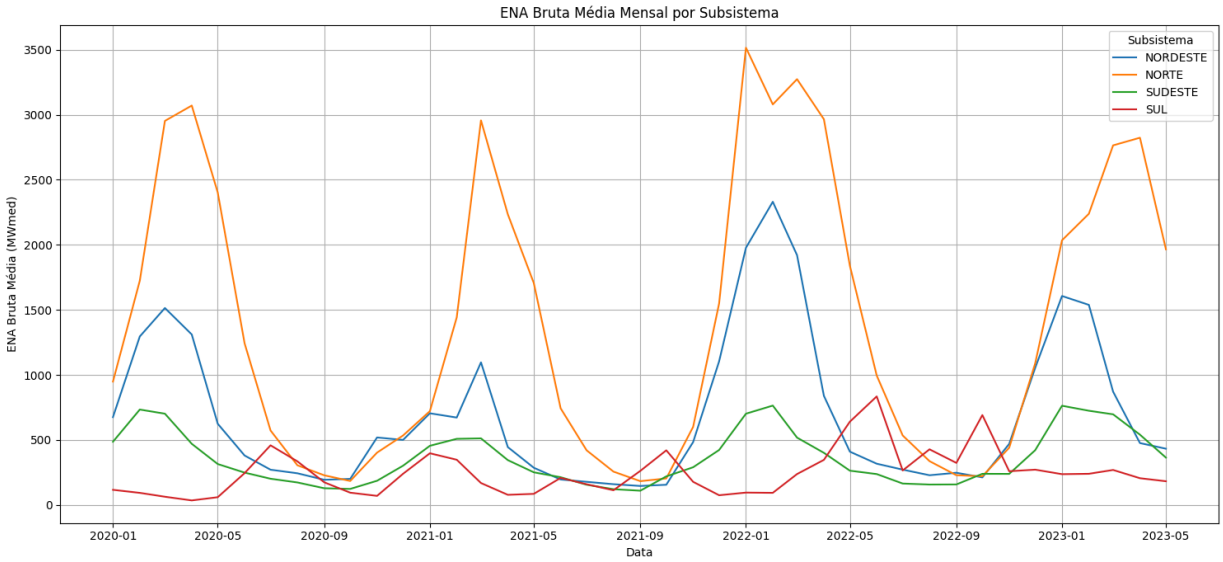
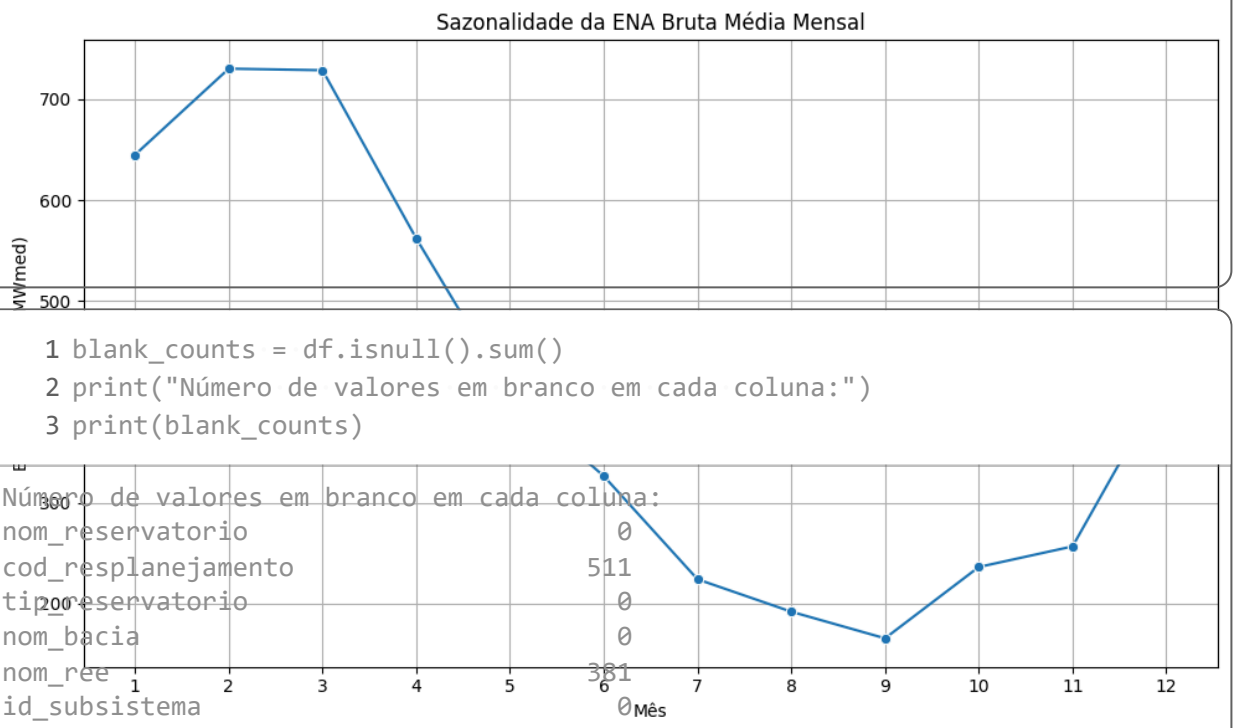
```python
# Tendência Mensal da ENA Bruta por Subsistema
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

# le o CSV unificado
df = pd.read_csv("/content/ENA_DIARIO_RESERVATORIOS_2020_2025.csv", sep=";",

# Converte datas
df["ena_data"] = pd.to_datetime(df["ena_data"], errors="coerce")

# Tendencia Mensal por Subsistema
df["mes_ano"] = df["ena_data"].dt.to_period("M")
monthly_ena = df.groupby(["mes_ano", "nom_subsistema"])["ena_bruta_res_mwmed
monthly_ena["mes_ano"] = monthly_ena["mes_ano"].dt.to_timestamp()

plt.figure(figsize=(15, 7))
sns.lineplot(data=monthly_ena, x="mes_ano", y="ena_bruta_res_mwmed", hue="no
plt.title("ENA Bruta Média Mensal por Subsistema")
plt.xlabel("Data")
plt.ylabel("ENA Bruta Média (MWmed)")
plt.legend(title="Subsistema")
plt.grid(True)
plt.tight_layout()
plt.show()

# Sazonalidade (médias por mês)
df["mes"] = df["ena_data"].dt.month
monthly_avg_ena = df.groupby("mes")["ena_bruta_res_mwmed"].mean().reset_inde

plt.figure(figsize=(10, 6))
sns.lineplot(data=monthly_avg_ena, x="mes", y="ena_bruta_res_mwmed", marker=
plt.title("Sazonalidade da ENA Bruta Média Mensal")
plt.xlabel("Mês")
plt.ylabel("ENA Bruta Média (MWmed)")
plt.xticks(range(1, 13))
plt.grid(True)
plt.tight_layout()
plt.show()

```






```
1 blank_counts = df.isnull().sum()
2 print("Número de valores em branco em cada coluna:")
3 print(blank_counts)
```

```
Número de valores em branco em cada coluna:
nom_reservatorio                    0
cod_resplanejamento               511
tip_reservatorio                    0
nom_bacia                           0
nom_ree                           381
id_subsistema                       0
nom_subsistema                      0
ena_data                            0
ena_bruta_res_mwmed              1480
ena_bruta_res_percentualmlt      1480
ena_armazenavel_res_mwmed        1480
ena_armazenavel_res_percentualmlt 1480
ena_queda_bruta                  1480
mlt_ena                          1481
fator_armazenamento              2454
desvio_mlt                       1481
eficiencia_queda                 2454
mes_ano                             0
mes                                 0
dtype: int64
```

```
1 df = df.drop('id_subsistema', axis=1)
2 print("Coluna 'id_subsistema' removida.")
```

```
Coluna 'id_subsistema' removida.
```

```
1 print("Columns in the DataFrame:")
```

```
print(df.columns)
for col in df.columns:
    print(col)
```

```
Columns in the DataFrame:
nom_reservatorio
cod_resplanejamento
tip_reservatorio
nom_bacia
nom_ree
nom_subsistema
ena_data
ena_bruta_res_mwmed
ena_bruta_res_percentualmlt
ena_armazenavel_res_mwmed
ena_armazenavel_res_percentualmlt
ena_queda_bruta
mlt_ena
fator_armazenamento
desvio_mlt
eficiencia_queda
mes_ano
mes
```

```
print("Quantidade de valores nulos por coluna:")
print(blank_counts)
```

```
Quantidade de valores nulos por coluna:
nom_reservatorio                      0
cod_resplanejamento                   0
tip_reservatorio                      0
nom_bacia                             0
nom_ree                               0
id_subsistema                         0
nom_subsistema                        0
ena_data                              0
ena_bruta_res_mwmed                 631
ena_bruta_res_percentualmlt         631
ena_armazenavel_res_mwmed           631
ena_armazenavel_res_percentualmlt   632
ena_queda_bruta                     632
mlt_ena                             632
fator_armazenamento                 857
desvio_mlt                          632
eficiencia_queda                    857
mes_ano                               0
mes                                   0
dtype: int64
```

```
print("Descrição dos dados:")
display(df.describe(include='all'))
```

Descrição dos dados:

	nom_reservatorio	cod_resplanejamento	tip_reservatorio	nom_bacia	no
count	94608	94608.000000	94608	94608	
unique	150	NaN	2	22	
top	14 DE JULHO	NaN	Fio dagua	PARANAIBA	PA
freq	631	NaN	56135	11985	
mean	NaN	133.657957	NaN	NaN	
min	NaN	1.000000	NaN	NaN	
25%	NaN	48.000000	NaN	NaN	
50%	NaN	119.000000	NaN	NaN	
75%	NaN	204.000000	NaN	NaN	

```
# Colocar no campo de valores null a mediana referente aos campos existente
numeric_cols_with_nulls = df.select_dtypes(include=['number']).columns[df.s

for col in numeric_cols_with_nulls:
  median_val = df[col].median()
  df[col] = df[col].fillna(median_val)

print("Valores nulos preenchidos com a mediana para colunas numéricas.")
```

```
Valores nulos preenchidos com a mediana para colunas numéricas.
```

```
mode_nom_ree = df['nom_ree'].mode()[0]
df['nom_ree'] = df['nom_ree'].fillna(mode_nom_ree)

print("Valores nulos na coluna 'nom_ree' preenchidos com a moda.")
```

```
Valores nulos na coluna 'nom_ree' preenchidos com a moda.
```

```
print("Número de valores nulos por coluna após preenchimento de 'nom_ree':")
print(df.isnull().sum())
```

```
Número de valores nulos por coluna após preenchimento de 'nom_ree':
nom_reservatorio                0
cod_resplanejamento             0
tip_reservatorio                0
nom_bacia                       0
nom_ree                         0
nom_subsistema                  0
ena_data                        0
ena_bruta_res_mwmed             0
ena_bruta_res_percentualmlt     0
ena_armazenavel_res_mwmed       0
ena_armazenavel_res_percentualmlt    0
```

```
ena_queda_bruta                    0
mlt_ena                            0
fator_armazenamento                0
desvio_mlt                         0
eficiencia_queda                   0
mes_ano                            0
mes                                0
dtype: int64
```

```
print("Quantidade de valores não nulos por coluna:")
print(df.count())
```

```
Quantidade de valores não nulos por coluna:
nom_reservatorio                   94608
cod_resplanejamento                94608
tip_reservatorio                   94608
nom_bacia                          94608
nom_ree                            94608
nom_subsistema                     94608
ena_data                           94608
ena_bruta_res_mwmed                94608
ena_bruta_res_percentualmlt        94608
ena_armazenavel_res_mwmed          94608
ena_armazenavel_res_percentualmlt  94608
ena_queda_bruta                    94608
mlt_ena                            94608
fator_armazenamento                94608
desvio_mlt                         94608
eficiencia_queda                   94608
mes_ano                            94608
mes                                94608
dtype: int64
```

```
!pip install xlwt
```

```
Collecting xlwt
  Downloading xlwt-1.3.0-py2.py3-none-any.whl.metadata (3.5 kB)
Downloading xlwt-1.3.0-py2.py3-none-any.whl (99 kB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 100.0/100.0 kB 4.6 MB/s eta 0:00:00
Installing collected packages: xlwt
Successfully installed xlwt-1.3.0
```

```
import numpy as np
import pandas as pd

max_rows_per_sheet = 1048575  # um a menos que o limite real
num_sheets = int(np.ceil(len(df) / max_rows_per_sheet))

with pd.ExcelWriter("ena_trusted_cleaned_split.xlsx", engine="openpyxl") as
    for i in range(num_sheets):
        start_row = i * max_rows_per_sheet
        end_row = min((i + 1) * max_rows_per_sheet, len(df))
        df_sheet = df.iloc[start_row:end_row].reset_index(drop=True)
        sheet_name = f"Sheet_{i+1}"
        df_sheet.to_excel(writer, sheet_name=sheet_name, index=False)

```

```
print(f"DataFrame salvo em 'ena_trusted_cleaned_split.xlsx' em {num_sheets}
```

```
DataFrame salvo em 'ena_trusted_cleaned_split.xlsx' em 1 sheets.
```

```
import pandas as pd

# Lista de arquivos de 2020 a 2025
arquivos = [
    "ENA_DIARIO_RESERVATORIOS_2020.csv",
    "ENA_DIARIO_RESERVATORIOS_2021.csv",
    "ENA_DIARIO_RESERVATORIOS_2022.csv",
    "ENA_DIARIO_RESERVATORIOS_2023.csv",
    "ENA_DIARIO_RESERVATORIOS_2024.csv",
    "ENA_DIARIO_RESERVATORIOS_2025.csv"
]

# Ler todos os CSVs e empilhar em um único DataFrame
dfs = [pd.read_csv(arq, sep=";", encoding="utf-8") for arq in arquivos]
df = pd.concat(dfs, ignore_index=True)

# Padronizar nome das colunas (boa prática)
df.columns = df.columns.str.strip().str.lower().str.replace(" ", "_")

# Converter a coluna de data
df["ena_data"] = pd.to_datetime(df["ena_data"], errors="coerce")

# Salvar em CSV unificado
df.to_csv("ENA_DIARIO_RESERVATORIOSs_2020_2025.csv", sep=";", encoding="utf

print("arquivo unificado gerado: ENA_DIARIO_RESERVATORIOSs_2020_2025.csv")

```

```
✅ Arquivo unificado gerado: ENA_DIARIO_RESERVATORIOSs_2020_2025.csv
```

```
print("Número de campos null por coluna:")
print(df.isnull().sum())
```

```
Número de campos null por coluna:
nom_reservatorio                     0
cod_resplanejamento                1678
tip_reservatorio                     0
nom_bacia                            0
nom_ree                           1362
id_subsistema                        0
nom_subsistema                       0
ena_data                             0
ena_bruta_res_mwmed                3490
ena_bruta_res_percentualmlt        3490
ena_armazenavel_res_mwmed          3490
ena_armazenavel_res_percentualmlt  3490
ena_queda_bruta                    3490
mlt_ena                            3490
dtype: int64
```

```
print("Número total de linhas na tabela:")
print(len(df))
```

```
Número total de linhas na tabela:
318478
```

```
numeric_cols_with_nulls = df.select_dtypes(include=['number']).columns[df.s

for col in numeric_cols_with_nulls:
  mean_val = df[col].mean()
  df[col] = df[col].fillna(mean_val)

print("Valores nulos preenchidos com a média para colunas numéricas.")
```

```
Valores nulos preenchidos com a média para colunas numéricas.
```

```
print("Número de campos null por coluna:")
print(df.isnull().sum())
```

```
Número de campos null por coluna:
nom_reservatorio                       0
cod_resplanejamento                    0
tip_reservatorio                       0
nom_bacia                              0
nom_ree                             1362
id_subsistema                          0
nom_subsistema                         0
ena_data                               0
ena_bruta_res_mwmed                    0
ena_bruta_res_percentualmlt            0
ena_armazenavel_res_mwmed              0
ena_armazenavel_res_percentualmlt      0
ena_queda_bruta                        0
mlt_ena                                0
dtype: int64
```

```
print("Valores únicos na coluna 'nom_ree' e suas contagens:")
display(df['nom_ree'].value_counts())
```

```
Valores únicos na coluna 'nom_ree' e suas contagens:
```

nom_ree	count
PARANA	92004
SUDESTE	77968
SUL	44157
PARANAPANEMA	21175
NORDESTE	18819
BELO MONTE	6273
ITAIPU	2091

```
dtype: int64
```

```
Comece a programar ou gere código com IA.
```

```
mode_nom_ree = df['nom_ree'].mode()[0]
df['nom_ree'] = df['nom_ree'].fillna(mode_nom_ree)

print("Valores nulos na coluna 'nom_ree' preenchidos com a moda.")
```

```
Valores nulos na coluna 'nom_ree' preenchidos com a moda.
```

```
print(df.isnull().sum())
```

```
nom_reservatorio                  0
cod_resplanejamento               0
cod_reservatorio                  0
nom_bacia                         0
nom_ree                           0
id_subsistema                     0
nom_subsistema                    0
ena_data                          0
ena_bruta_res_mwmed               0
ena_bruta_res_percentualmlt       0
ena_armazenavel_res_mwmed         0
ena_armazenavel_res_percentualmlt 0
ena_queda_bruta                   0
mlt_ena                           0
dtype: int64
```

```
df.to_csv('ena_trusted_cleaned_final.csv', index=False)
print("Tabela salva como 'ena_trusted_cleaned_final.csv'. Você pode baixá-l
```

```
Tabela salva como 'ena_trusted_cleaned_final.csv'. Você pode baixá-la agora.
```

```
Comece a programar ou gere código com IA.
```

```
df['ena_data'] = pd.to_datetime(df['ena_data'], errors='coerce')
df['ano'] = df['ena_data'].dt.year
```

Calcular máximos anuais por bacia

```
1 max_ena_anual_por_bacia = df.groupby(['nom_bacia', 'ano'])['ena_bruta_res_m
```

```
1 mean_max_ena_por_bacia = max_ena_anual_por_bacia.groupby('nom_bacia')['ena_b
2 display(mean_max_ena_por_bacia)
```

	nom_bacia	ena_bruta_res_mwmed
0	AMAZONAS	11270.815195
1	ARAGUARI	515.378498
2	CAPIVARI	911.659522
3	DOCE	1294.839234
4	GRANDE	2921.155505
5	IGUACU	3109.404037
6	ITABAPOANA	508.390494
7	ITAJAI	871.933756
8	JACUI	1998.066498
9	JEQUITINHONHA	2134.974760
10	MUCURI	205.556496
11	PARAGUACU	666.958746
12	PARAGUAI	459.257997
13	PARAIBA DO SUL	553.268253
14	PARANA	19718.521523
15	PARANAIBA	3977.160519
16	PARANAPANEMA	1174.070755
17	PARNAIBA	587.473003
18	SANTA MARIA VIT	NaN
19	SAO FRANCISCO	6555.348193
20	TIETE	1966.129738
21	TOCANTINS	18668.575379
22	URUGUAI	5350.419803

Próximas etapas: Gerar código com mean_max_ena_por_bacia | New interactive sheet

Exibir resultados

Apresentar as bacias com suas médias de máximos anuais de ENA bruta.

```
display(mean_max_ena_por_bacia)
```

	nom_bacia	ena_bruta_res_mwmed
0	AMAZONAS	11270.815195
1	ARAGUARI	515.378498
2	CAPIVARI	911.659522
3	DOCE	1294.839234
4	GRANDE	2921.155505
5	IGUACU	3109.404037
6	ITABAPOANA	508.390494
7	ITAJAI	871.933756
8	JACUI	1998.066498
9	JEQUITINHONHA	2134.974760
10	MUCURI	205.556496
11	PARAGUACU	666.958746
12	PARAGUAI	459.257997
13	PARAIBA DO SUL	553.268253
14	PARANA	19718.521523
15	PARANAIBA	3977.160519
16	PARANAPANEMA	1174.070755
17	PARNAIBA	587.473003
18	SANTA MARIA VIT	NaN
19	SAO FRANCISCO	6555.348193
20	TIETE	1966.129738
21	TOCANTINS	18668.575379
22	URUGUAI	5350.419803

Próximas etapas: Gerar código com `mean_max_ena_por_bacia` New interactive sheet

Preparar dados de data

```
df['ena_data'] = pd.to_datetime(df['ena_data'], errors='coerce')
df['ano'] = df['ena_data'].dt.year
```

Calcular máximos anuais por bacia

```
1 max_ena_anual_por_bacia = df.groupby(['nom_bacia', 'ano'])[['ena_bruta_res_
```

Para cada bacia, calcular a média dos valores máximos anuais de ENA bruta e ENA armazenável encontrados no passo anterior.

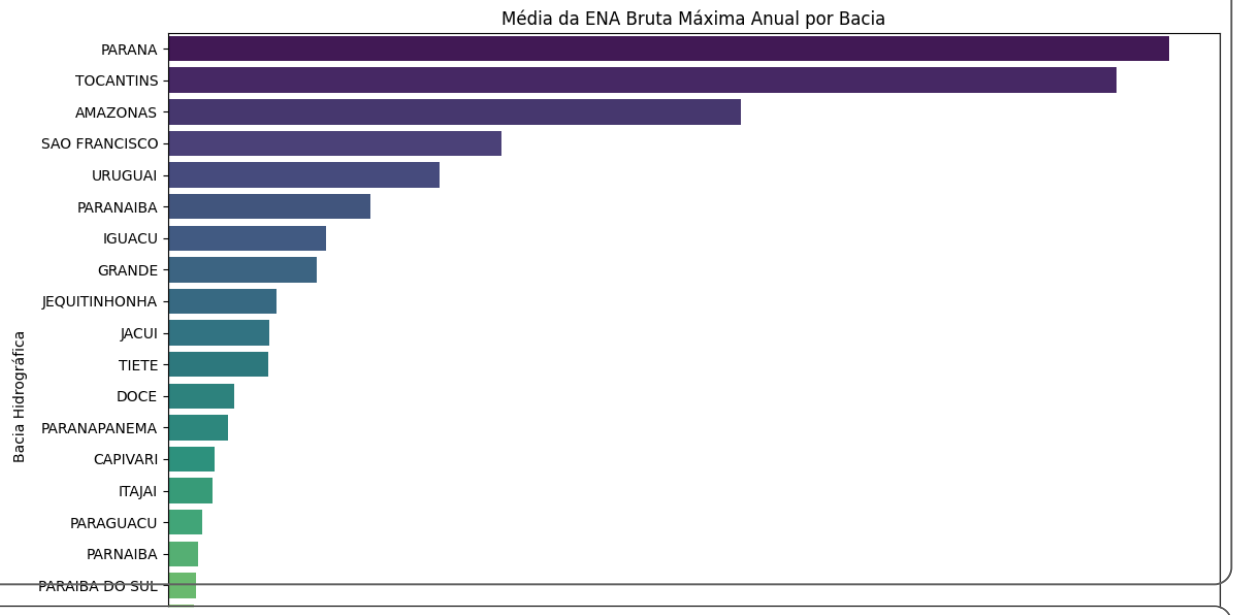
```
1 mean_max_ena_por_bacia = max_ena_anual_por_bacia.groupby('nom_bacia')[['ena
2 display(mean_max_ena_por_bacia)
```

	nom_bacia	ena_bruta_res_mwmed	ena_armazenavel_res_mwmed
0	AMAZONAS	11270.815195	11270.815195
1	ARAGUARI	515.378498	271.955253
2	CAPIVARI	911.659522	816.425719
3	DOCE	1294.839234	1122.902253
4	GRANDE	2921.155505	2869.129992
5	IGUACU	3109.404037	2446.498733
6	ITABAPOANA	508.390494	110.749999
7	ITAJAI	871.933756	321.325000
8	JACUI	1998.066498	1263.899248
9	JEQUITINHONHA	2134.974760	1612.961513
10	MUCURI	205.556496	89.837000
11	PARAGUACU	666.958746	549.724993
12	PARAGUAI	459.257997	459.257997
13	PARAIBA DO SUL	553.268253	405.426998
14	PARANA	19718.521523	18858.140664
15	PARANAIBA	3977.160519	3977.160519
16	PARANAPANEMA	1174.070755	1012.018265
17	PARNAIBA	587.473003	463.152500
18	SANTA MARIA VIT	NaN	NaN
19	SAO FRANCISCO	6555.348193	5879.794727
20	TIETE	1966.129738	1966.129738
21	TOCANTINS	18668.575379	12955.074578
22	URUGUAI	5350.419803	4658.466697

Próximas etapas: Gerar código com mean_max_ena_por_bacia | New interactive sheet

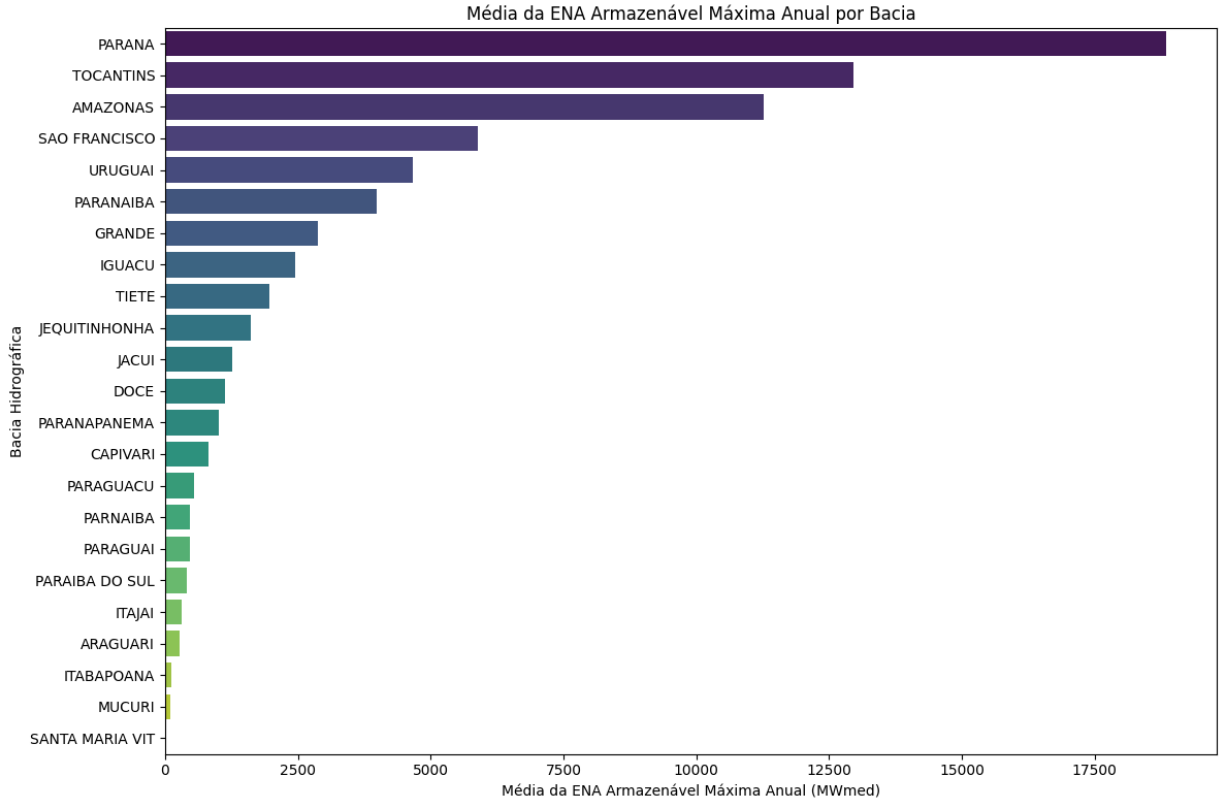
```
plt.figure(figsize=(12, 8))
sns.barplot(data=mean_max_ena_por_bacia.sort_values('ena_bruta_res_mwmed', a
            x='ena_bruta_res_mwmed', y='nom_bacia', palette='viridis')
plt.title('Média da ENA Bruta Máxima Anual por Bacia')
plt.xlabel('Média da ENA Bruta Máxima Anual (MWmed)')
plt.ylabel('Bacia Hidrográfica')
plt.tight_layout()
plt.show()

plt.figure(figsize=(12, 8))
sns.barplot(data=mean_max_ena_por_bacia.sort_values('ena_armazenavel_res_mwm
            x='ena_armazenavel_res_mwmed', y='nom_bacia', palette='viridis')
plt.title('Média da ENA Armazenável Máxima Anual por Bacia')
plt.xlabel('Média da ENA Armazenável Máxima Anual (MWmed)')
plt.ylabel('Bacia Hidrográfica')
plt.tight_layout()
plt.show()
```

```
/tmp/ipython-input-4083449121.py:6: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v

  sns.barplot(data=mean_max_ena_por_bacia.sort_values('ena_bruta_res_mwmed', asc
```




```
1 Comece a programar ou gere código com IA.
```

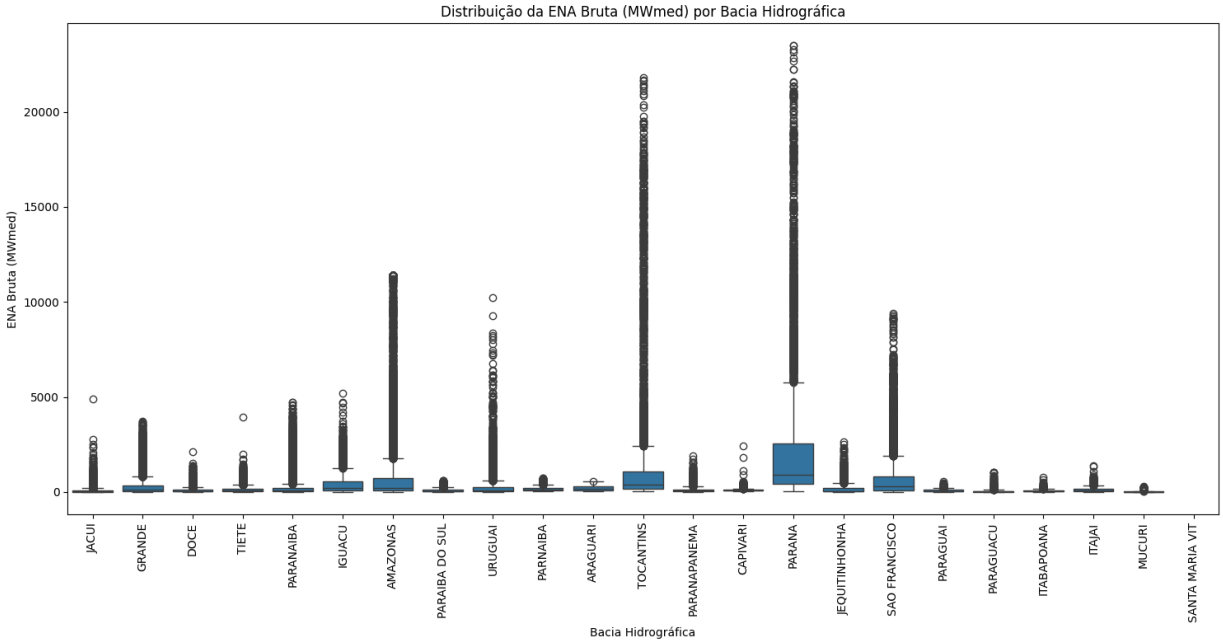
```
1 fig = px.line(df, x='ena_data', y='ena_bruta_res_mwmed', color='nom_bacia',
2               title='ENA Bruta (MWmed) ao Longo do Tempo por Bacia Hidrográf
3               labels={'ena_data': 'Data', 'ena_bruta_res_mwmed': 'ENA Bruta
4 fig.update_layout(xaxis_title='Data', yaxis_title='ENA Bruta (MWmed)')
5 fig.show()

sns.barplot(data=mean_max_ena_por_bacia.sort_values('ena_armazenavel_res_mwmed
```



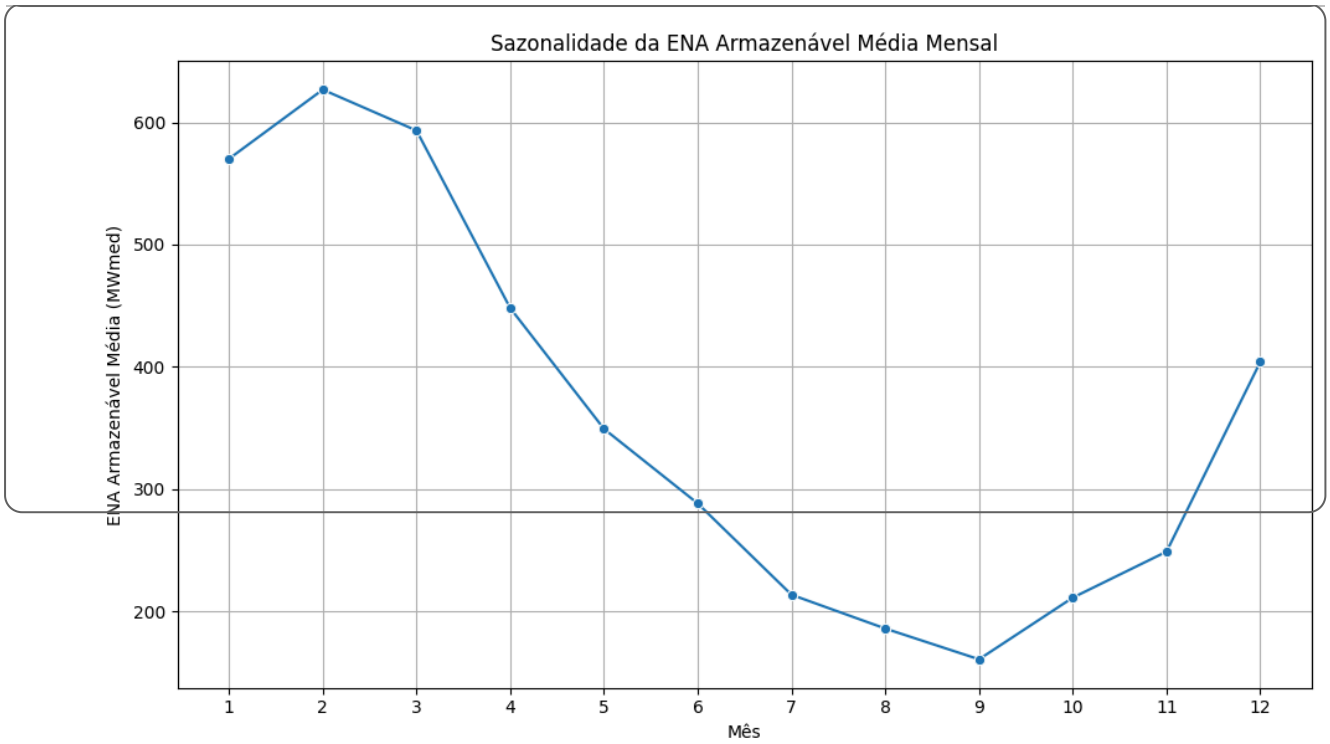
ENA Bruta (MWmed) ao Longo do Tempo por Bacia Hidrográfica

```
Comece a programar ou gere código com IA.
```

```
plt.figure(figsize=(15, 8))
sns.boxplot(data=df, x='nom_bacia', y='ena_bruta_res_mwmed')
plt.title('Distribuição da ENA Bruta (MWmed) por Bacia Hidrográfica')
plt.xlabel('Bacia Hidrográfica')
plt.ylabel('ENA Bruta (MWmed)')
plt.xticks(rotation=90)
plt.tight_layout()
plt.show()
```




```
df['mes'] = df['ena_data'].dt.month
monthly_avg_ena_armazenavel = df.groupby('mes')['ena_armazenavel_res_mwmed']

plt.figure(figsize=(10, 6))
sns.lineplot(data=monthly_avg_ena_armazenavel, x='mes', y='ena_armazenavel_r
plt.title('Sazonalidade da ENA Armazenável Média Mensal')
plt.xlabel('Mês')
plt.ylabel('ENA Armazenável Média (MWmed)')
plt.xticks(range(1, 13))
plt.grid(True)
plt.tight_layout()
plt.show()
```


Sazonalidade da ENA Armazenável Média Mensal
ENA Armazenável Média (MWmed)
600
500
400
300
200
1
2
3
4
5
6
7
8
9
10
11
12
Mês