
# Análise de dados - Financeiro

http://www.linkedin.com/in/rosileneljusto (http://www.linkedin.com/in/rosileneljusto)

**Nesse projeto temos uma empresa que está interessada em saber o cenário atual. Vai ser verificado alguns insights dos dados.**

In [1]:
```
# Importando bibliotecas

import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import matplotlib.patches as mpatches

# Desenhar os gráficos na página
%matplotlib inline
```

In [2]:
```
#importando os dados
#criar o dataframe com base no arquivo excel
#precisamos pular as 2 primeiras linhas (skiprows)
#precisamos pular as linhas finais (skipfooter)
df = pd.read_excel('bd_financeiro_.xlsx' ) # ex. ,skiprows=2, skipfooter=16
# skiprows  Este parâmetro é usado para pular linhas passadas em novo quadro de dados
# skipfooter    Este parâmetro é usado para pular o número de linhas na parte inferior do arquivo
```

In [3]:
```
# Verificando os dados
df.head()
```

Out[3]:

| | GRUPO_DESPESA | DESCRICAO | DT_DESPESA | VALOR_DESPESA | DT_VENDA | Classificacao | Valor | Date | mes | mes_nome | ano | dia_semana |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **0** | Custos Financeiros | Cartão de Crédito | 2017-01-02 | 12.1201 | 2018-02-24 | Turno 2 | 0.0 | 2017-01-01 | 1.0 | jan | 2017.0 | dom |
| **1** | Custos Financeiros | Cartão de Crédito | 2017-01-03 | 14.1480 | 2019-04-22 | Turno 2 | 0.0 | 2017-01-02 | 1.0 | jan | 2017.0 | seg |
| **2** | Custos Financeiros | Cartão de Crédito | 2017-01-04 | 17.8524 | 2019-08-30 | Turno 2 | 0.0 | 2017-01-03 | 1.0 | jan | 2017.0 | ter |
| **3** | Custos Financeiros | Cartão de Crédito | 2017-01-05 | 15.5943 | 2018-09-30 | Turno 2 | 0.0 | 2017-01-04 | 1.0 | jan | 2017.0 | qua |
| **4** | Custos Financeiros | Cartão de Crédito | 2017-01-06 | 11.4765 | 2019-12-24 | Turno 2 | 0.0 | 2017-01-05 | 1.0 | jan | 2017.0 | qui |

In [4]:
```
# descrever os dados
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 15569 entries, 0 to 15568
Data columns (total 13 columns):
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   GRUPO_DESPESA  15569 non-null  object
 1   DESCRICAO      15569 non-null  object
 2   DT_DESPESA     15569 non-null  datetime64[ns]
 3   VALOR_DESPESA  15569 non-null  float64
 4   DT_VENDA       2172 non-null   datetime64[ns]
 5   Classificacao  2172 non-null   object
 6   Valor          2172 non-null   float64
 7   Date           1095 non-null   datetime64[ns]
 8   mes            1095 non-null   float64
 9   mes_nome       1095 non-null   object
 10  ano            1095 non-null   float64
 11  dia_semana     1095 non-null   object
 12  dia_num        1095 non-null   float64
dtypes: datetime64[ns](3), float64(5), object(5)
memory usage: 1.5+ MB
```

In [5]:
```
# Exibir as 5 ultimas linhas
df.tail(20)
```

Out[5]:

| | GRUPO_DESPESA | DESCRICAO | DT_DESPESA | VALOR_DESPESA | DT_VENDA | Classificacao | Valor | Date | mes | mes_nome | ano | dia_seman |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **15549** | Tributos | Outros Tributos | 2017-10-28 | 307.2600 | NaT | NaN | NaN | NaT | NaN | NaN | NaN | Na |
| **15550** | Tributos | Outros Tributos | 2018-10-20 | 144.8000 | NaT | NaN | NaN | NaT | NaN | NaN | NaN | Na |
| **15551** | Tributos | Simples | 2019-10-20 | 1528.2200 | NaT | NaN | NaN | NaT | NaN | NaN | NaN | Na |
| **15552** | Tributos | Simples | 2019-10-31 | 801.1300 | NaT | NaN | NaN | NaT | NaN | NaN | NaN | Na |
| **15553** | Tributos | Outros Tributos | 2019-10-20 | 144.8000 | NaT | NaN | NaN | NaT | NaN | NaN | NaN | Na |
| **15554** | Tributos | Outros Tributos | 2019-10-31 | 157.9900 | NaT | NaN | NaN | NaT | NaN | NaN | NaN | Na |
| **15555** | Tributos | Simples | 2019-11-20 | 5152.2598 | NaT | NaN | NaN | NaT | NaN | NaN | NaN | Na |
| **15556** | Tributos | Outros Tributos | 2017-11-01 | 21.0000 | NaT | NaN | NaN | NaT | NaN | NaN | NaN | Na |
| **15557** | Tributos | Outros Tributos | 2017-11-02 | 188.0000 | NaT | NaN | NaN | NaT | NaN | NaN | NaN | Na |
| **15558** | Tributos | Simples | 2017-11-20 | 4985.7598 | NaT | NaN | NaN | NaT | NaN | NaN | NaN | Na |
| **15559** | Tributos | Outros Tributos | 2017-11-17 | 144.8000 | NaT | NaN | NaN | NaT | NaN | NaN | NaN | Na |
| **15560** | Tributos | Outros Tributos | 2017-11-28 | 309.2000 | NaT | NaN | NaN | NaT | NaN | NaN | NaN | Na |
| **15561** | Tributos | Outros Tributos | 2018-11-20 | 144.8000 | NaT | NaN | NaN | NaT | NaN | NaN | NaN | Na |
| **15562** | Tributos | Simples | 2019-12-20 | 5500.2598 | NaT | NaN | NaN | NaT | NaN | NaN | NaN | Na |
| **15563** | Tributos | Simples | 2017-12-20 | 5960.6699 | NaT | NaN | NaN | NaT | NaN | NaN | NaN | Na |
| **15564** | Tributos | Outros Tributos | 2017-12-17 | 144.8000 | NaT | NaN | NaN | NaT | NaN | NaN | NaN | Na |
| **15565** | Tributos | Outros Tributos | 2017-12-28 | 310.9400 | NaT | NaN | NaN | NaT | NaN | NaN | NaN | Na |

| | GRUPO_DESPESA | DESCRICAO | DT_DESPESA | VALOR_DESPESA | DT_VENDA | Classificacao | Valor | Date | mes | mes_nome | ano | dia_seman |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **15566** | Tributos | Simples | 2018-12-28 | 336.2300 | NaT | NaN | NaN | NaT | NaN | NaN | NaN | Na |
| **15567** | Tributos | Outros Tributos | 2018-12-20 | 144.8000 | NaT | NaN | NaN | NaT | NaN | NaN | NaN | Na |
| **15568** | Tributos | Simples | 2017-07-20 | 5744.8398 | NaT | NaN | NaN | NaT | NaN | NaN | NaN | Na |

In [6]:
```
# Tipos dos dados
df.dtypes
```

Out[6]:
```
GRUPO_DESPESA            object
DESCRICAO                object
DT_DESPESA       datetime64[ns]
VALOR_DESPESA           float64
DT_VENDA         datetime64[ns]
Classificacao            object
Valor                   float64
Date             datetime64[ns]
mes                     float64
mes_nome                 object
ano                     float64
dia_semana               object
dia_num                 float64
dtype: object
```

In [7]:
```
# Contando o dataframe
df.count
```

Out[7]:
```
<bound method DataFrame.count of            GRUPO_DESPESA          DESCRICAO DT_DESPESA  VALOR_DESPESA  \
0      Custos Financeiros  Cartão de Crédito 2017-01-02        12.1201
1      Custos Financeiros  Cartão de Crédito 2017-01-03        14.1480
2      Custos Financeiros  Cartão de Crédito 2017-01-04        17.8524
3      Custos Financeiros  Cartão de Crédito 2017-01-05        15.5943
4      Custos Financeiros  Cartão de Crédito 2017-01-06        11.4765
...                   ...                ...        ...           ...
15564            Tributos    Outros Tributos 2017-12-17       144.8000
15565            Tributos    Outros Tributos 2017-12-28       310.9400
15566            Tributos            Simples 2018-12-28       336.2300
15567            Tributos    Outros Tributos 2018-12-20       144.8000
15568            Tributos            Simples 2017-07-20      5744.8398

        DT_VENDA Classificacao  Valor       Date  mes mes_nome     ano  \
0     2018-02-24       Turno 2    0.0 2017-01-01  1.0      jan  2017.0
1     2019-04-22       Turno 2    0.0 2017-01-02  1.0      jan  2017.0
2     2019-08-30       Turno 2    0.0 2017-01-03  1.0      jan  2017.0
3     2018-09-30       Turno 2    0.0 2017-01-04  1.0      jan  2017.0
4     2019-12-24       Turno 2    0.0 2017-01-05  1.0      jan  2017.0
...          ...           ...    ...        ...  ...      ...     ...
15564        NaT           NaN    NaN        NaT  NaN      NaN     NaN
15565        NaT           NaN    NaN        NaT  NaN      NaN     NaN
15566        NaT           NaN    NaN        NaT  NaN      NaN     NaN
15567        NaT           NaN    NaN        NaT  NaN      NaN     NaN
15568        NaT           NaN    NaN        NaT  NaN      NaN     NaN

      dia_semana  dia_num
0            dom      7.0
1            seg      1.0
2            ter      2.0
3            qua      3.0
4            qui      4.0
...          ...      ...
15564        NaN      NaN
15565        NaN      NaN
15566        NaN      NaN
15567        NaN      NaN
15568        NaN      NaN
```

```
[15569 rows x 13 columns]>
```

In [ ]:
```
# Descreve os dados
# df.describe
```

In [8]:
```
# Renomeando algumas colunas
df.columns = ['Grupo-despesa', 'Descricao', 'Dt_Despesa', 'Valor_Despesa', 'Dt_Venda', 'classificacao', 'Valor', 'Date',
df.head()
```

Out[8]:

| | Grupo-despesa | Descricao | Dt_Despesa | Valor_Despesa | Dt_Venda | classificacao | Valor | Date | Mes | Mes_nome | Ano | Dia_Semana | Dia_Num |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **0** | Custos Financeiros | Cartão de Crédito | 2017-01-02 | 12.1201 | 2018-02-24 | Turno 2 | 0.0 | 2017-01-01 | 1.0 | jan | 2017.0 | dom | 7.0 |
| **1** | Custos Financeiros | Cartão de Crédito | 2017-01-03 | 14.1480 | 2019-04-22 | Turno 2 | 0.0 | 2017-01-02 | 1.0 | jan | 2017.0 | seg | 1.0 |
| **2** | Custos Financeiros | Cartão de Crédito | 2017-01-04 | 17.8524 | 2019-08-30 | Turno 2 | 0.0 | 2017-01-03 | 1.0 | jan | 2017.0 | ter | 2.0 |
| **3** | Custos Financeiros | Cartão de Crédito | 2017-01-05 | 15.5943 | 2018-09-30 | Turno 2 | 0.0 | 2017-01-04 | 1.0 | jan | 2017.0 | qua | 3.0 |
| **4** | Custos Financeiros | Cartão de Crédito | 2017-01-06 | 11.4765 | 2019-12-24 | Turno 2 | 0.0 | 2017-01-05 | 1.0 | jan | 2017.0 | qui | 4.0 |

In [9]:
```
# Selecionando as colunas númericas, object, float e int.
df.select_dtypes(include='number').head
```

Out[9]:
```
<bound method NDFrame.head of        Valor_Despesa  Valor  Mes     Ano  Dia_Num
0             12.1201    0.0  1.0  2017.0      7.0
1             14.1480    0.0  1.0  2017.0      1.0
2             17.8524    0.0  1.0  2017.0      2.0
3             15.5943    0.0  1.0  2017.0      3.0
4             11.4765    0.0  1.0  2017.0      4.0
...               ...    ...  ...     ...      ...
15564        144.8000    NaN  NaN     NaN      NaN
15565        310.9400    NaN  NaN     NaN      NaN
15566        336.2300    NaN  NaN     NaN      NaN
15567        144.8000    NaN  NaN     NaN      NaN
15568       5744.8398    NaN  NaN     NaN      NaN

[15569 rows x 5 columns]>
```

In [10]:
```
# Seleciona múltiplos tipos de dados
df.select_dtypes(include=['int', 'datetime', 'object']).head
```

Out[10]:
```
<bound method NDFrame.head of            Grupo-despesa          Descricao Dt_Despesa   Dt_Venda  \
0      Custos Financeiros  Cartão de Crédito 2017-01-02 2018-02-24
1      Custos Financeiros  Cartão de Crédito 2017-01-03 2019-04-22
2      Custos Financeiros  Cartão de Crédito 2017-01-04 2019-08-30
3      Custos Financeiros  Cartão de Crédito 2017-01-05 2018-09-30
4      Custos Financeiros  Cartão de Crédito 2017-01-06 2019-12-24
...                   ...                ...        ...        ...
15564            Tributos    Outros Tributos 2017-12-17        NaT
15565            Tributos    Outros Tributos 2017-12-28        NaT
15566            Tributos            Simples 2018-12-28        NaT
15567            Tributos    Outros Tributos 2018-12-20        NaT
15568            Tributos            Simples 2017-07-20        NaT

      classificacao       Date Mes_nome Dia_Semana
0           Turno 2 2017-01-01      jan        dom
1           Turno 2 2017-01-02      jan        seg
2           Turno 2 2017-01-03      jan        ter
3           Turno 2 2017-01-04      jan        qua
4           Turno 2 2017-01-05      jan        qui
...             ...        ...      ...        ...
15564           NaN        NaT      NaN        NaN
15565           NaN        NaT      NaN        NaN
15566           NaN        NaT      NaN        NaN
15567           NaN        NaT      NaN        NaN
15568           NaN        NaT      NaN        NaN

[15569 rows x 8 columns]>
```

In [ ]:
```
# Exclui certos tipos de dados
# df.select_dtypes(exclude='object')
```

In [11]:
```
# Transformando Tipos
df['Mes']= df['Mes'].astype(object).head
```

In [12]:
```
# Mostrando o tipo das colunas
df.dtypes
```

Out[12]:
```
Grupo-despesa            object
Descricao                object
Dt_Despesa       datetime64[ns]
Valor_Despesa           float64
Dt_Venda         datetime64[ns]
classificacao            object
Valor                   float64
Date             datetime64[ns]
Mes                      object
Mes_nome                 object
Ano                     float64
Dia_Semana               object
Dia_Num                 float64
dtype: object
```

In [13]:
```
# Remove linhas c/ indíce
# df.drop()
# Remove colunas com atribuição
# df = df.drop(columns=['Mes'])
# del df['Mes']
# df.pop(Mes)
# df.pop(Mes)
# df.show()
# df.printSchema()
# axis=0 ------coluna
# axis=1-------linha

# Removendo várias colunas
df = df.drop(['Mes', 'Mes_nome', 'Ano', 'Dia_Semana', 'Dia_Num'], axis=1)
df
```

Out[13]:

| | Grupo-despesa | Descricao | Dt_Despesa | Valor_Despesa | Dt_Venda | classificacao | Valor | Date |
|---|---|---|---|---|---|---|---|---|
| **0** | Custos Financeiros | Cartão de Crédito | 2017-01-02 | 12.1201 | 2018-02-24 | Turno 2 | 0.0 | 2017-01-01 |
| **1** | Custos Financeiros | Cartão de Crédito | 2017-01-03 | 14.1480 | 2019-04-22 | Turno 2 | 0.0 | 2017-01-02 |
| **2** | Custos Financeiros | Cartão de Crédito | 2017-01-04 | 17.8524 | 2019-08-30 | Turno 2 | 0.0 | 2017-01-03 |
| **3** | Custos Financeiros | Cartão de Crédito | 2017-01-05 | 15.5943 | 2018-09-30 | Turno 2 | 0.0 | 2017-01-04 |
| **4** | Custos Financeiros | Cartão de Crédito | 2017-01-06 | 11.4765 | 2019-12-24 | Turno 2 | 0.0 | 2017-01-05 |
| **...** | ... | ... | ... | ... | ... | ... | ... | ... |
| **15564** | Tributos | Outros Tributos | 2017-12-17 | 144.8000 | NaT | NaN | NaN | NaT |
| **15565** | Tributos | Outros Tributos | 2017-12-28 | 310.9400 | NaT | NaN | NaN | NaT |
| **15566** | Tributos | Simples | 2018-12-28 | 336.2300 | NaT | NaN | NaN | NaT |
| **15567** | Tributos | Outros Tributos | 2018-12-20 | 144.8000 | NaT | NaN | NaN | NaT |
| **15568** | Tributos | Simples | 2017-07-20 | 5744.8398 | NaT | NaN | NaN | NaT |

15569 rows × 8 columns

In [14]:
```
#df.isnull().sum().sum() retorna o total de valores ausentes

# Contando os números ausentes para cada coluna
df.isnull().sum()
```

Out[14]:
```
Grupo-despesa        0
Descricao            0
Dt_Despesa           0
Valor_Despesa        0
Dt_Venda         13397
classificacao    13397
Valor            13397
Date             14474
dtype: int64
```

In [15]:
```
# Porcentagem de valores ausentes
df.isna().mean()
```

Out[15]:
```
Grupo-despesa    0.000000
Descricao        0.000000
Dt_Despesa       0.000000
Valor_Despesa    0.000000
Dt_Venda         0.860492
classificacao    0.860492
Valor            0.860492
Date             0.929668
dtype: float64
```

In [16]:
```
# Encontrando valores ausentes
ausentes = np.nan
df
```

Out[16]:

| | Grupo-despesa | Descricao | Dt_Despesa | Valor_Despesa | Dt_Venda | classificacao | Valor | Date |
|---|---|---|---|---|---|---|---|---|
| **0** | Custos Financeiros | Cartão de Crédito | 2017-01-02 | 12.1201 | 2018-02-24 | Turno 2 | 0.0 | 2017-01-01 |
| **1** | Custos Financeiros | Cartão de Crédito | 2017-01-03 | 14.1480 | 2019-04-22 | Turno 2 | 0.0 | 2017-01-02 |
| **2** | Custos Financeiros | Cartão de Crédito | 2017-01-04 | 17.8524 | 2019-08-30 | Turno 2 | 0.0 | 2017-01-03 |
| **3** | Custos Financeiros | Cartão de Crédito | 2017-01-05 | 15.5943 | 2018-09-30 | Turno 2 | 0.0 | 2017-01-04 |
| **4** | Custos Financeiros | Cartão de Crédito | 2017-01-06 | 11.4765 | 2019-12-24 | Turno 2 | 0.0 | 2017-01-05 |
| **...** | ... | ... | ... | ... | ... | ... | ... | ... |
| **15564** | Tributos | Outros Tributos | 2017-12-17 | 144.8000 | NaT | NaN | NaN | NaT |
| **15565** | Tributos | Outros Tributos | 2017-12-28 | 310.9400 | NaT | NaN | NaN | NaT |
| **15566** | Tributos | Simples | 2018-12-28 | 336.2300 | NaT | NaN | NaN | NaT |
| **15567** | Tributos | Outros Tributos | 2018-12-20 | 144.8000 | NaT | NaN | NaN | NaT |
| **15568** | Tributos | Simples | 2017-07-20 | 5744.8398 | NaT | NaN | NaN | NaT |

15569 rows × 8 columns

In [17]:
```
#  Listando as colunas Horizontal
print(list(df.columns))
```

```
['Grupo-despesa', 'Descricao', 'Dt_Despesa', 'Valor_Despesa', 'Dt_Venda', 'classificacao', 'Valor', 'Date']
```

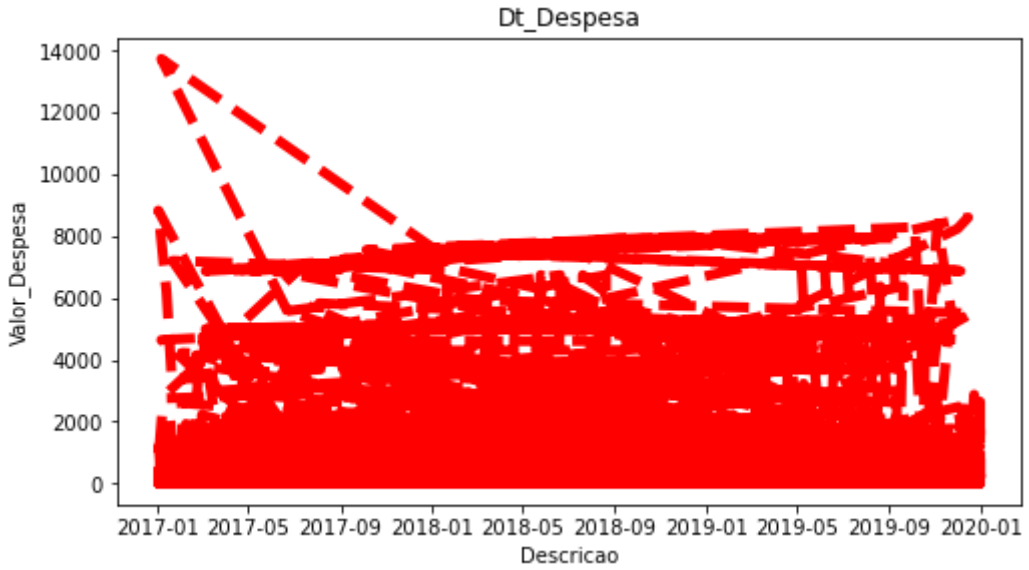
In [18]:
```
# listar linhas aleatoria do dataframe
df.sample()
```

Out[18]:

| | Grupo-despesa | Descricao | Dt_Despesa | Valor_Despesa | Dt_Venda | classificacao | Valor | Date |
|---|---|---|---|---|---|---|---|---|
| **10672** | Custo Mercadoria | Diversos | 2019-07-08 | 692.05 | NaT | NaN | NaN | NaT |

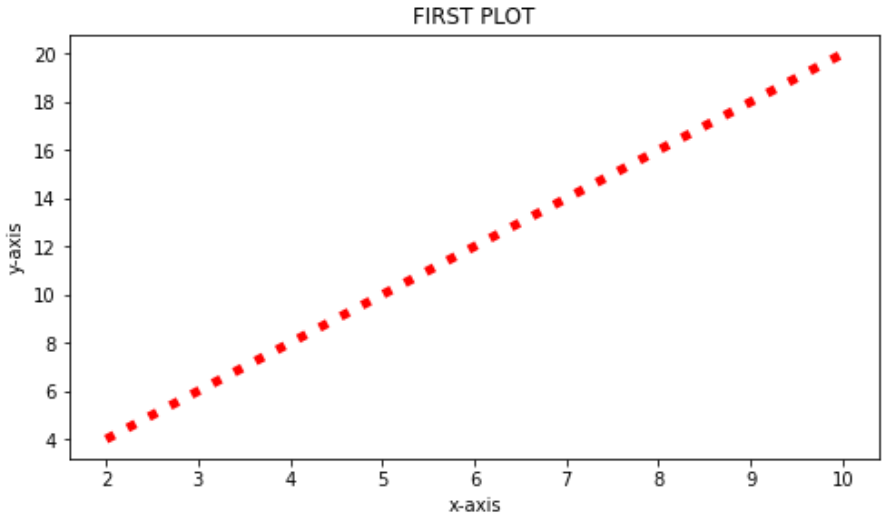
In [19]:
```
# Exibindo todas as colunas vertical
# list(df)
```

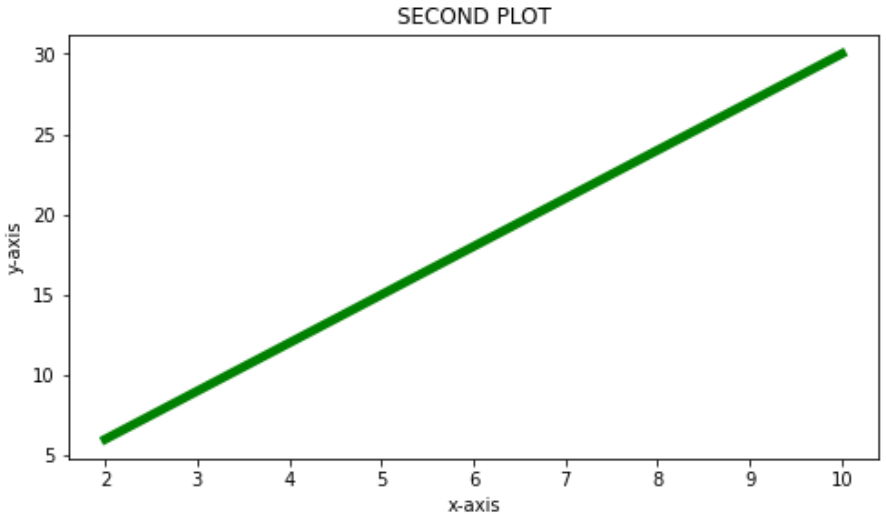
In [20]:
```
# Na variável, contar e plota no grafico de barra horizontal
#df['Grupo-despesa'].value_counts(10).plot.barh(title = 'Grupo-Despesa')

#x2=df[('Grupo-despesa')]
x = df[('Dt_Despesa')]
y = df[('Valor_Despesa')]
y_1 = y
     # primeiro gráfico
plt.subplots(figsize=(15,5))
plt.subplot(1, 2, 1)
plt.plot(x, y_1, 'r', linewidth=5, linestyle='--')
plt.title('Dt_Despesa')
plt.xlabel('Descricao')
plt.ylabel('Valor_Despesa')
plt.tight_layout(4)
plt.show()
# segundo gráfico
x = df[('Grupo-despesa')]
y = df[('Valor')]
y_2 = y
plt.subplots(figsize=(15,5))
plt.subplot(1, 2, 2)
plt.plot(x, y_2, 'r', linewidth=5)
plt.title('Grupo-despesa')
plt.xlabel('Grupo-despesa')
plt.ylabel('Valor')
plt.tight_layout(4)
plt.show()
```


Dt_Despesa
14000
12000
10000
8000
6000
4000
2000
0
Valor_Despesa
2017-01
2017-05
2017-09
2018-01
2018-05
2018-09
2019-01
2019-05
2019-09
2020-01
Descricao


Grupo-despesa
6000
5000
4000
3000
2000
1000
0
Valor
Custos Financeiros
Grupo-despesa

In [21]:
```
# plotar o gráfico por sub grupo procedimento
# df['Descricao'].value_counts().plot.bar()
x = np.array([2, 4, 6, 8, 10,])
y_1 = 2*x
y_2 = 3*x
plt.subplots(figsize=(15,5))
plt.subplot(1, 2, 1)
plt.plot(x, y_1, 'r', linewidth=5, linestyle=':')
plt.title('FIRST PLOT')
plt.xlabel('x-axis')
plt.ylabel('y-axis')
plt.subplot(1, 2, 2)
plt.plot(x, y_2, 'g', linewidth=5)
plt.title('SECOND PLOT')
plt.xlabel('x-axis')
plt.ylabel('y-axis')
plt.tight_layout(4)
plt.show()
plt.show()
```



In [22]:
```
# Mostrar os maiores valores
df.nlargest(20, "Valor")
```

Out[22]:

| | Grupo-despesa | Descricao | Dt_Despesa | Valor_Despesa | Dt_Venda | classificacao | Valor | Date |
|---|---|---|---|---|---|---|---|---|
| **1007** | Custos Financeiros | Cartão de Crédito | 2017-12-22 | 19.8882 | 2017-12-09 | Turno 2 | 6325.4199 | 2019-10-05 |
| **1941** | Custos Financeiros | Cartão de Crédito | 2019-11-02 | 10.8948 | 2019-10-12 | Turno 1 | 5538.1802 | NaT |
| **860** | Custos Financeiros | Cartão de Crédito | 2019-10-25 | 12.0166 | 2019-10-12 | Turno 2 | 5484.8799 | 2019-05-11 |
| **1659** | Custos Financeiros | Cartão de Crédito | 2019-07-25 | 10.5102 | 2019-07-06 | Turno 1 | 5234.6401 | NaT |
| **986** | Custos Financeiros | Cartão de Crédito | 2017-12-01 | 15.8739 | 2019-12-17 | Turno 2 | 5109.5200 | 2019-09-14 |
| **1014** | Custos Financeiros | Cartão de Crédito | 2017-12-30 | 33.9445 | 2017-12-16 | Turno 2 | 5096.3101 | 2019-10-12 |
| **1837** | Custos Financeiros | Cartão de Crédito | 2019-09-21 | 19.4433 | 2018-09-30 | Turno 1 | 5027.0898 | NaT |
| **1026** | Custos Financeiros | Cartão de Crédito | 2018-12-11 | 7.6447 | 2017-12-30 | Turno 2 | 4952.4502 | 2019-10-24 |
| **929** | Custos Financeiros | Cartão de Crédito | 2018-11-04 | 16.6494 | 2017-11-20 | Turno 2 | 4814.9302 | 2019-07-19 |
| **987** | Custos Financeiros | Cartão de Crédito | 2017-12-02 | 32.4913 | 2019-12-18 | Turno 2 | 4691.0098 | 2019-09-15 |
| **1048** | Custos Financeiros | Cartão de Crédito | 2017-07-03 | 6.6845 | 2018-12-22 | Turno 2 | 4605.3599 | 2019-11-15 |
| **1015** | Custos Financeiros | Cartão de Crédito | 2017-12-31 | 10.0993 | 2017-12-17 | Turno 2 | 4595.6499 | 2019-10-13 |
| **979** | Custos Financeiros | Cartão de Crédito | 2019-12-24 | 6.0314 | 2019-12-10 | Turno 2 | 4474.8999 | 2019-09-07 |
| **1572** | Custos Financeiros | Cartão de Crédito | 2018-06-26 | 6.3052 | 2018-06-10 | Turno 1 | 4450.8599 | NaT |
| **989** | Custos Financeiros | Cartão de Crédito | 2017-12-04 | 7.2888 | 2019-12-20 | Turno 2 | 4438.2002 | 2019-09-17 |
| **190** | Custos Financeiros | Cartão de Crédito | 2017-03-20 | 5.8218 | 2017-03-11 | Turno 2 | 4435.2202 | 2017-07-10 |
| **1600** | Custos Financeiros | Cartão de Crédito | 2019-06-24 | 10.3361 | 2019-06-08 | Turno 1 | 4417.5098 | NaT |
| **433** | Custos Financeiros | Cartão de Crédito | 2019-05-22 | 10.7688 | 2019-05-11 | Turno 2 | 4338.2402 | 2018-03-10 |
| **1021** | Custos Financeiros | Cartão de Crédito | 2018-12-06 | 8.5557 | 2017-12-23 | Turno 2 | 4281.1299 | 2019-10-19 |
| **1035** | Custos Financeiros | Cartão de Crédito | 2018-12-20 | 23.1049 | 2018-12-09 | Turno 2 | 4209.5400 | 2019-11-02 |

In [23]:
```
# Mostrar os menores valores
# df.nsmallest(nº número, 'nome da coluna')
df.nsmallest(20, 'Valor_Despesa')
```

Out[23]:

| | Grupo-despesa | Descricao | Dt_Despesa | Valor_Despesa | Dt_Venda | classificacao | Valor | Date |
|---|---|---|---|---|---|---|---|---|
| **8032** | Custos Financeiros | Cartão de Crédito | 2019-08-01 | 1.0005 | NaT | NaN | NaN | NaT |
| **8100** | Custos Financeiros | Cartão de Crédito | 2019-09-19 | 1.0019 | NaT | NaN | NaN | NaT |
| **5173** | Custos Financeiros | Cartão de Crédito | 2017-11-11 | 1.0023 | NaT | NaN | NaN | NaT |
| **7764** | Custos Financeiros | Cartão de Crédito | 2019-02-11 | 1.0024 | NaT | NaN | NaN | NaT |
| **7477** | Custos Financeiros | Cartão de Crédito | 2018-02-14 | 1.0063 | NaT | NaN | NaN | NaT |
| **7793** | Custos Financeiros | Cartão de Crédito | 2019-03-19 | 1.0080 | NaT | NaN | NaN | NaT |
| **8105** | Custos Financeiros | Cartão de Crédito | 2019-09-24 | 1.0081 | NaT | NaN | NaN | NaT |
| **2870** | Custos Financeiros | Cartão de Crédito | 2018-09-18 | 1.0150 | NaT | NaN | NaN | NaT |
| **8018** | Custos Financeiros | Cartão de Crédito | 2018-08-15 | 1.0150 | NaT | NaN | NaN | NaT |
| **8175** | Custos Financeiros | Cartão de Crédito | 2018-11-09 | 1.0178 | NaT | NaN | NaN | NaT |
| **8022** | Custos Financeiros | Cartão de Crédito | 2018-08-20 | 1.0192 | NaT | NaN | NaN | NaT |
| **5335** | Custos Financeiros | Cartão de Crédito | 2017-07-24 | 1.0206 | NaT | NaN | NaN | NaT |
| **3244** | Custos Financeiros | Cartão de Crédito | 2017-01-31 | 1.0266 | NaT | NaN | NaN | NaT |
| **7722** | Custos Financeiros | Cartão de Crédito | 2017-07-07 | 1.0295 | NaT | NaN | NaN | NaT |
| **2994** | Custos Financeiros | Cartão de Crédito | 2019-10-21 | 1.0309 | NaT | NaN | NaN | NaT |
| **7494** | Custos Financeiros | Cartão de Crédito | 2017-03-13 | 1.0324 | NaT | NaN | NaN | NaT |
| **5148** | Custos Financeiros | Cartão de Crédito | 2019-11-14 | 1.0338 | NaT | NaN | NaN | NaT |
| **8016** | Custos Financeiros | Cartão de Crédito | 2018-08-13 | 1.0346 | NaT | NaN | NaN | NaT |
| **8078** | Custos Financeiros | Cartão de Crédito | 2018-09-22 | 1.0374 | NaT | NaN | NaN | NaT |
| **7539** | Custos Financeiros | Cartão de Crédito | 2017-05-07 | 1.0382 | NaT | NaN | NaN | NaT |

In [24]:
```
vetor1 = np.array([0, 1, 2, 3, 4, 5, 6, 7, 8])
df
```

Out[24]:

| | Grupo-despesa | Descricao | Dt_Despesa | Valor_Despesa | Dt_Venda | classificacao | Valor | Date |
|---|---|---|---|---|---|---|---|---|
| **0** | Custos Financeiros | Cartão de Crédito | 2017-01-02 | 12.1201 | 2018-02-24 | Turno 2 | 0.0 | 2017-01-01 |
| **1** | Custos Financeiros | Cartão de Crédito | 2017-01-03 | 14.1480 | 2019-04-22 | Turno 2 | 0.0 | 2017-01-02 |
| **2** | Custos Financeiros | Cartão de Crédito | 2017-01-04 | 17.8524 | 2019-08-30 | Turno 2 | 0.0 | 2017-01-03 |
| **3** | Custos Financeiros | Cartão de Crédito | 2017-01-05 | 15.5943 | 2018-09-30 | Turno 2 | 0.0 | 2017-01-04 |
| **4** | Custos Financeiros | Cartão de Crédito | 2017-01-06 | 11.4765 | 2019-12-24 | Turno 2 | 0.0 | 2017-01-05 |
| **...** | ... | ... | ... | ... | ... | ... | ... | ... |
| **15564** | Tributos | Outros Tributos | 2017-12-17 | 144.8000 | NaT | NaN | NaN | NaT |
| **15565** | Tributos | Outros Tributos | 2017-12-28 | 310.9400 | NaT | NaN | NaN | NaT |
| **15566** | Tributos | Simples | 2018-12-28 | 336.2300 | NaT | NaN | NaN | NaT |
| **15567** | Tributos | Outros Tributos | 2018-12-20 | 144.8000 | NaT | NaN | NaN | NaT |
| **15568** | Tributos | Simples | 2017-07-20 | 5744.8398 | NaT | NaN | NaN | NaT |

15569 rows × 8 columns

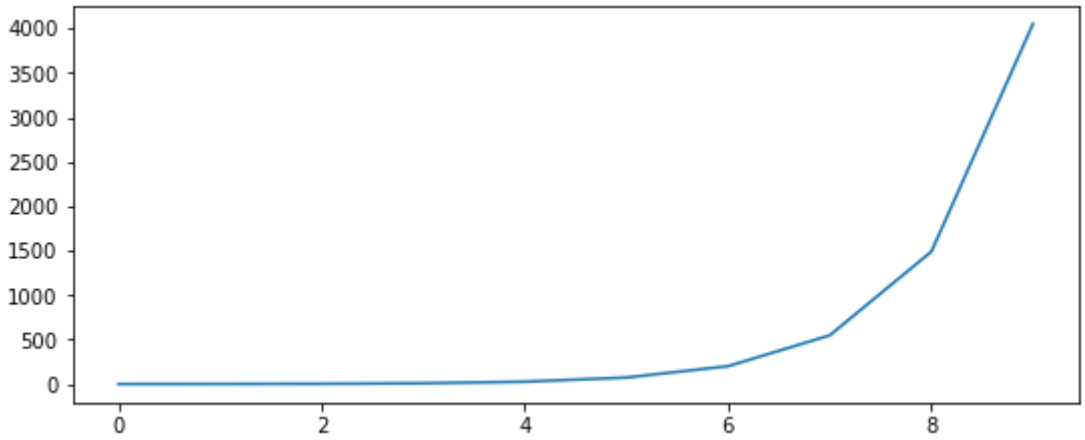
In [25]:
```
# Usando métodos do array NumPy
vetor1.cumsum()
```

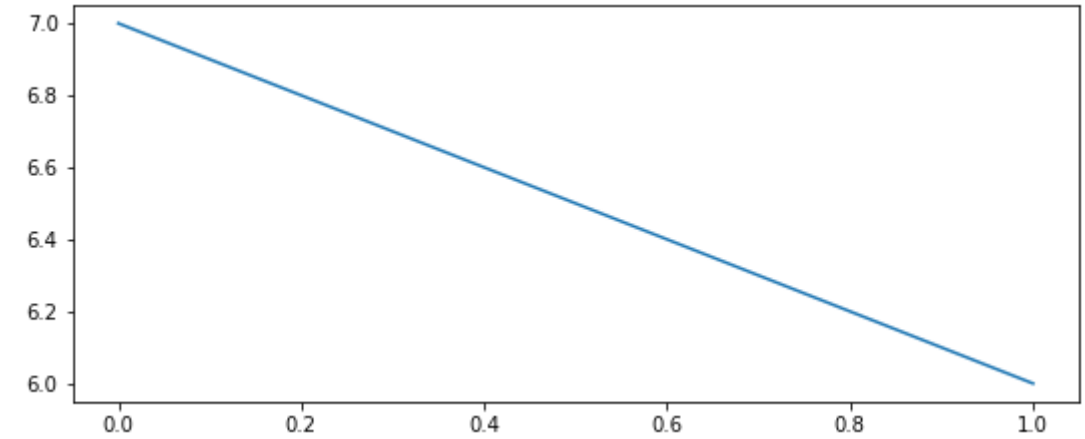
Out[25]:
```
array([ 0,  1,  3,  6, 10, 15, 21, 28, 36], dtype=int32)
```

In [26]:
```
# Um objeto do tipo ndarray é um recipiente multidimensional de itens do mesmo tipo e tamanho.
type(vetor1)
```

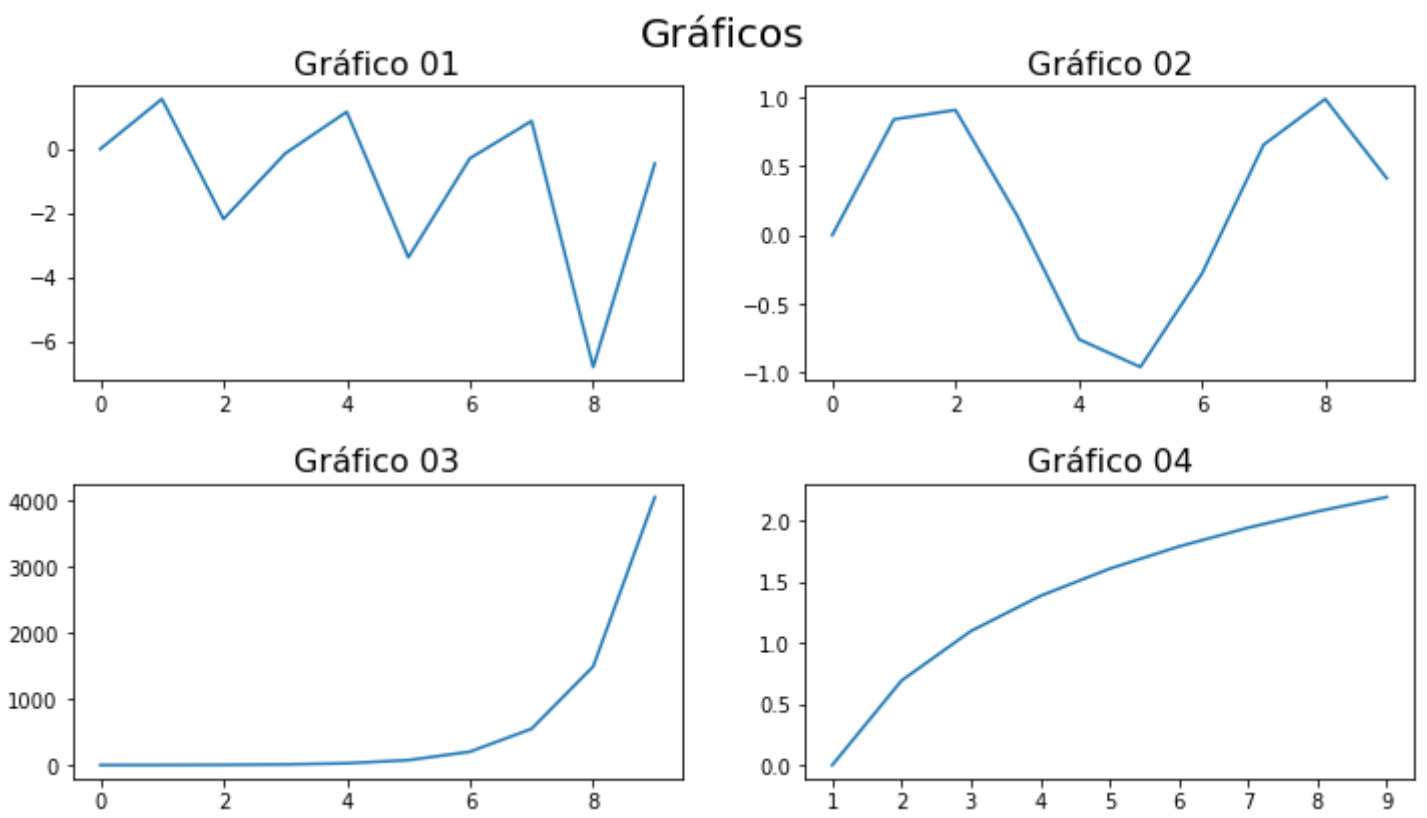
Out[26]:
```
numpy.ndarray
```

In [27]:
```
plt.figure(figsize = ((9, 8)))
#Gráfico 1:
x = np.arange(0, 10)
y = np.cosh(x)
plt.subplot(2, 1, 1)
plt.plot(x,y)
#Gráfico 2:
x = np.array([0, 1])
y = np.array([7, 6])
plt.subplot(2, 1, 2)
plt.plot(x,y)

plt.show()
```


4000
3500
3000
2500
2000
1500
1000
500
0
0
2
4
6
8


7.0
6.8
6.6
6.4
6.2
6.0
0.0
0.2
0.4
0.6
0.8
1.0

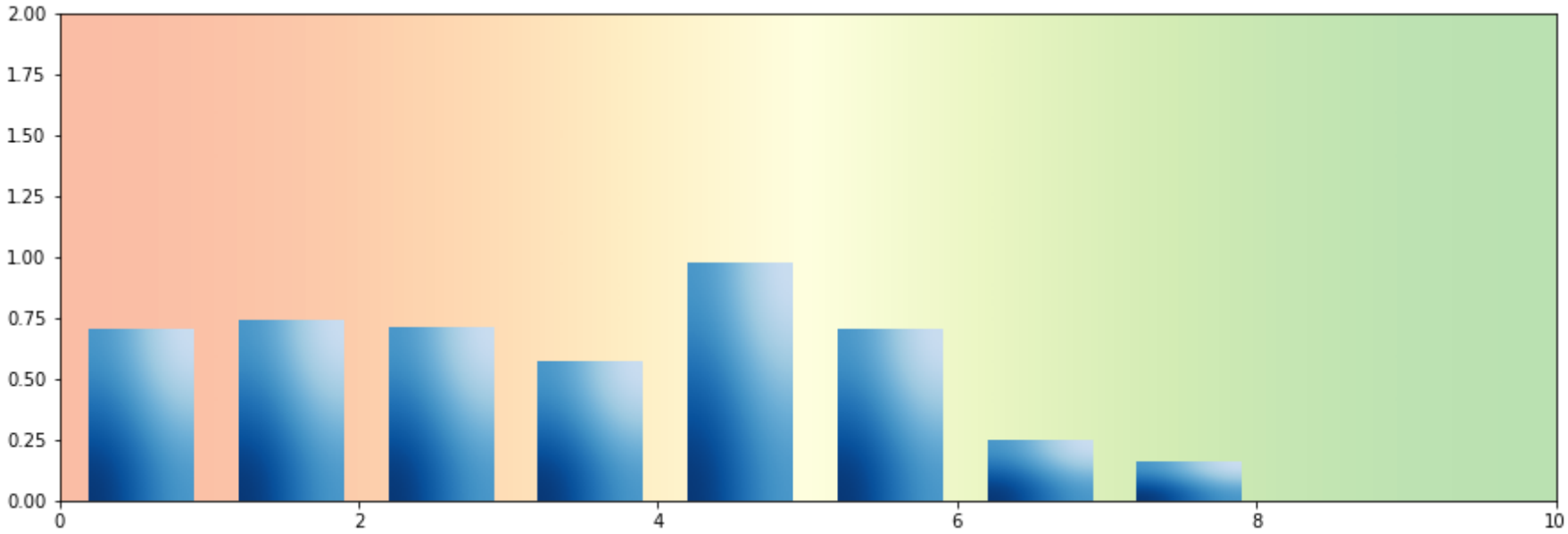
In [28]:
```
plt.figure(figsize = ((12, 6)))
x = np.arange(0, 10)
y = np.tan(x)
plt.subplot(2, 2, 1)
plt.plot(x,y)
plt.title("Gráfico 01", fontsize = 16)
x = np.arange(0, 10)
y = np.sin(x)
plt.subplot(2, 2, 2)
plt.plot(x,y)
plt.title("Gráfico 02", fontsize = 16)
x = np.arange(0, 10)
y = np.sinh(x)
plt.subplot(2, 2, 3)
plt.plot(x,y)
plt.title("Gráfico 03", fontsize = 16)
x = np.arange(1, 10)
y = np.log(x)
plt.subplot(2, 2, 4)
plt.plot(x,y)
plt.title("Gráfico 04", fontsize = 16)
plt.suptitle("Gráficos", fontsize = 20)
plt.subplots_adjust(left=0.125, bottom=0.1, right=0.9, top=0.9, wspace=0.2, hspace=0.35)


plt.show()
```


Gráficos
Gráfico 01
Gráfico 02
Gráfico 03
Gráfico 04

In [29]:

```
np.random.seed(19680801)
def gradient_image(ax, extent, direction=0.5, cmap_range=(0, 3), **kwargs):
    """
    Desenhe uma imagem gradiente com base em um mapa de cores.

    Parâmetros
    ----------
    machado: eixos
        Os eixos para desenhar.
    extensão
        A extensão da imagem como (xmin, xmax, ymin, ymax).
        Por padrão, isso está em coordenadas de eixos, mas pode ser
        alterado usando o argumento de palavra-chave *transform*.
    direção: flutuar
        A direção do gradiente. Este é um número em
        faixa de 0 (=vertical) a 1 (=horizontal).
    cmap_range : float, float
        A fração (cmin, cmax) do mapa de cores que deve ser
        usado para o gradiente, onde o mapa de cores completo é (0, 1).
    **kwargs
        Outros parâmetros são passados ••para `.Axes.imshow()`.
        Particularmente útil é *cmap*.
    """
    phi = direction * np.pi / 2
    v = np.array([np.cos(phi), np.sin(phi)])
    X = np.array([[v @ [1, 0], v @ [1, 1]],
                  [v @ [0, 0], v @ [0, 1]]])

    a, b = cmap_range
    X = a + (b - a) / X.max() * X
    im = ax.imshow(X, extent=extent, interpolation='bicubic',
                   vmin=0, vmax=1, **kwargs)
    return im
def gradient_barh(ax, x, y, width=0.5, bottom=0):
    for left, top in zip(x, y):
        right = left + width
        gradient_image(ax, extent=(left, right, bottom, top),
                       cmap=plt.cm.Blues_r, cmap_range=(0, 0.8))

xmin, xmax = xlim = 0, 10
```

```
ymin, ymax = ylim = 0, 2


fig, ax = plt.subplots(figsize=(15,5))
ax.set(xlim=xlim, ylim=ylim, autoscale_on=False)

# background image
gradient_image(ax, direction=1, extent=(0, 1, 0, 1), transform=ax.transAxes,
               cmap=plt.cm.RdYlGn, cmap_range=(0.2, 0.8), alpha=0.5)

N = 8
x = np.arange(N) + 0.20
y = np.random.rand(N)
gradient_barh(ax, x, y, width=0.7)
ax.set_aspect('auto')
plt.show()
```



In [ ]:
```
#import shutil
#src= r 'C:\Users\rosil\Projetos_portfolios_RosyDS\Analise de dados_Python - Financeiro.ipynb'
#des= r 'C:\Users\rosil\Projeto_portfolio_GitHub\Analise de dados_Python - Financeiro.pdf'
#shutil.copy2(src, des)
```

In [ ]:
```

```

In [ ]:
```

```