Ejercicio Numpy y Problemas Estadísticos

Importar la librería NumPy

In [ ]:
```
import pandas as pd
import numpy as np
import os

os.chdir('E:\WORK IN PROGRESS\Data Analytics course\parte 2 python\week 23')
```

Importar el archivo “supermarket_sales.csv” (https://www.kaggle.com/datasets/aungpyaeap/supermarket-sales)

In [ ]:
```
df=pd.read_csv('supermarket_sales - Sheet1.csv')
```

In [ ]:
```
df.sample(5)
```

Out[ ]:

	Invoice ID	Branch	City	Customer type	Gender	Product line	Unit price	Quantity	Tax 5%	Tot
726	442-44-6497	C	Naypyitaw	Member	Male	Home and lifestyle	55.57	3	8.3355	175.045
415	268-03-6164	B	Mandalay	Normal	Male	Health and beauty	96.11	1	4.8055	100.915
117	659-65-8956	B	Mandalay	Member	Male	Fashion accessories	51.36	1	2.5680	53.928
896	781-84-8059	C	Naypyitaw	Normal	Male	Fashion accessories	60.74	7	21.2590	446.439
42	354-25-5821	B	Mandalay	Member	Female	Sports and travel	69.12	6	20.7360	435.456

In [ ]:
```
df.shape
```

Out[ ]:
```
(1000, 17)
```

In [ ]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1000 entries, 0 to 999
Data columns (total 17 columns):
 #   Column                   Non-Null Count  Dtype  
---  ------                   --------------  -----  
 0   Invoice ID               1000 non-null   object 
 1   Branch                   1000 non-null   object 
 2   City                     1000 non-null   object 
 3   Customer type            1000 non-null   object 
 4   Gender                   1000 non-null   object 
 5   Product line             1000 non-null   object 
 6   Unit price               1000 non-null   float64
 7   Quantity                 1000 non-null   int64  
 8   Tax 5%                   1000 non-null   float64
 9   Total                    1000 non-null   float64
 10  Date                     1000 non-null   object 
 11  Time                     1000 non-null   object 
 12  Payment                  1000 non-null   object 
 13  cogs                     1000 non-null   float64
 14  gross margin percentage  1000 non-null   float64
 15  gross income             1000 non-null   float64
 16  Rating                   1000 non-null   float64
dtypes: float64(7), int64(1), object(9)
memory usage: 132.9+ KB
```

Generar estadística descriptiva básica en las columnas unit_price y quantity:

Cálculo de la media, mediana, moda

In [ ]:

```
# Para hacer un análisis univariado más completo utilizo numpy
unit_price_np = df[['Unit price']].to_numpy()
quantity_np = df[['Quantity']].to_numpy()
```

estadística descriptiva unit_price

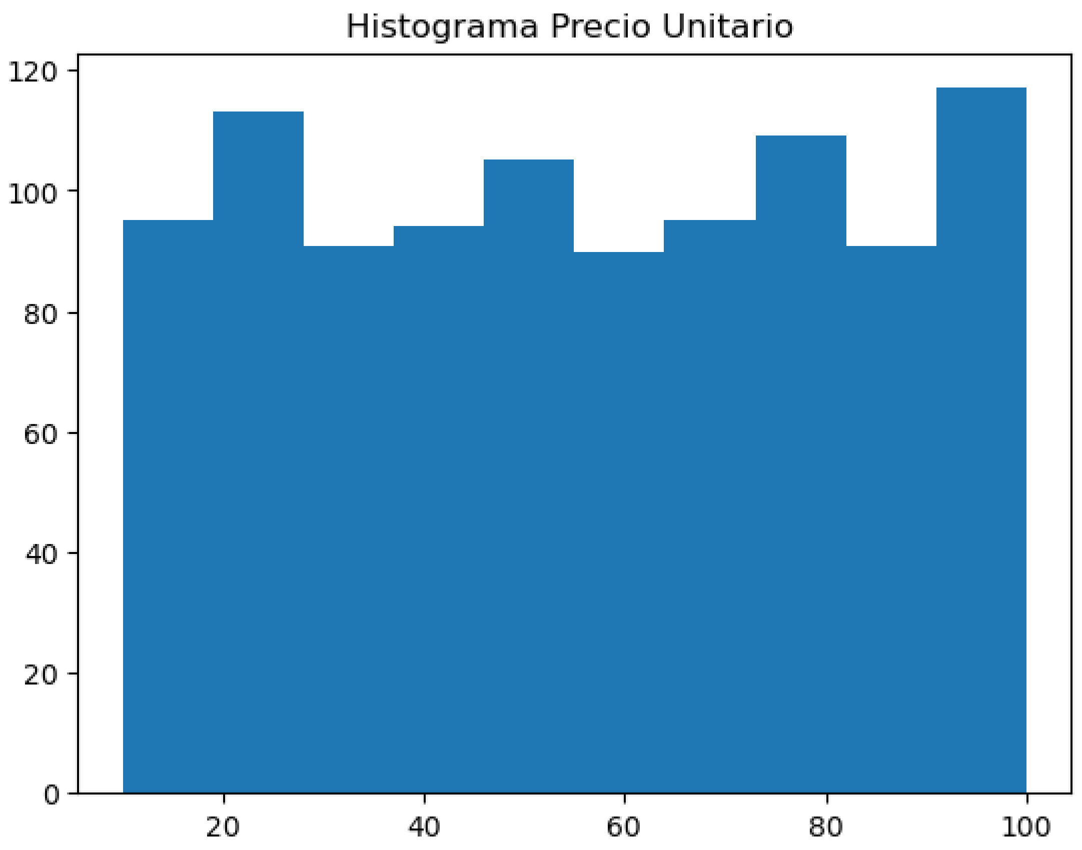
In [ ]:

```
media =unit_price_np.mean()
format_media="{:.2f}".format(media)
print('La media del precio unitario es = $',format_media)

mediana = np.median(unit_price_np)
format_mediana="{:.2f}".format(mediana)
print('La mediana del precio unitario es = $',format_mediana)

vals, counts=np.unique(unit_price_np,return_counts=True)
index=np.argmax(counts)
moda=vals[index]
print(f'la moda del precio unitario es = ${moda:.2f}')
```

```
La media del precio unitario es = $ 55.67
La mediana del precio unitario es = $ 55.23
la moda del precio unitario es = $83.77
```

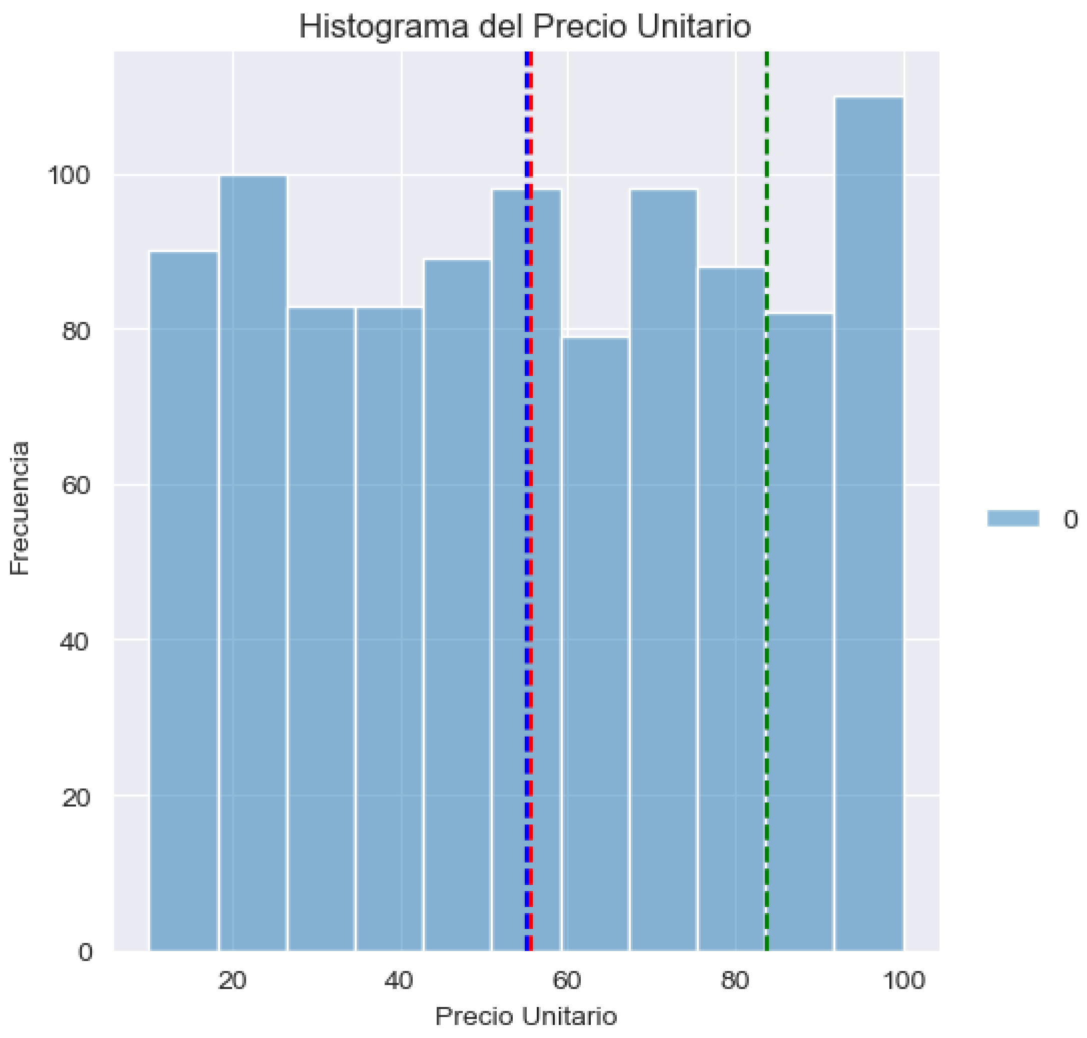
In [ ]:

```
import matplotlib.pyplot as plt
plt.hist(unit_price_np)
plt.title('Histograma Precio Unitario ')
plt.show()
```



In [ ]:
```
import seaborn as sns

sns.set_style('darkgrid')
sns.displot(unit_price_np)
plt.xlabel('Precio Unitario')
plt.ylabel('Frecuencia')
plt.title('Histograma del Precio Unitario')
plt.axvline(x=unit_price_np.mean(),color='red', ls='--')
plt.axvline(x=mediana,color='blue',ls='--')
plt.axvline(x=moda,color='green',ls='--')
```

Out[ ]:
```
<matplotlib.lines.Line2D at 0x1e94d13d6a0>
```

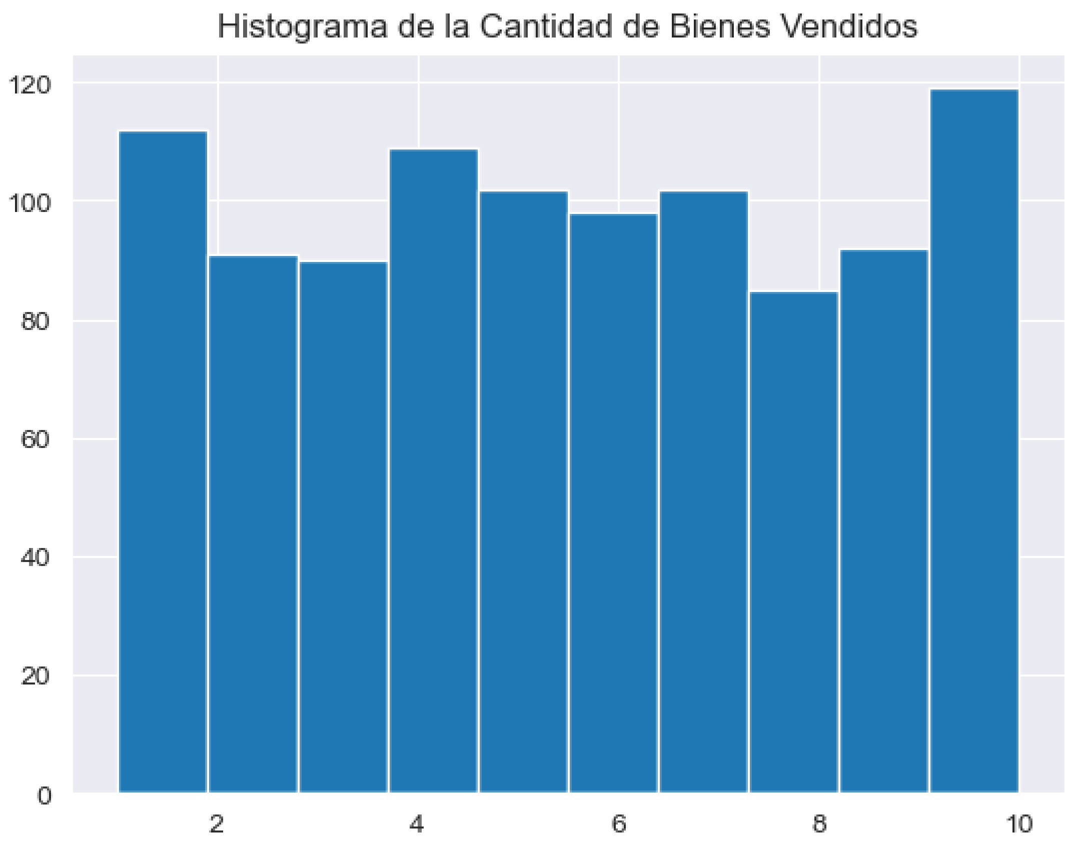


estadística descriptiva quantity

In [ ]:
```
media =quantity_np.mean()
format_media="{:.2f}".format(media)
print('La media de la cantidad de bienes vendidos es =',format_media)

mediana = np.median(quantity_np)
format_mediana="{:.2f}".format(mediana)
print('La mediana de la cantidad de bienes vendidos es =',format_mediana)

vals, counts=np.unique(quantity_np,return_counts=True)
index=np.argmax(counts)
moda=vals[index]
print(f'la moda de la cantidad de bienes vendidos es = {moda:.2f}')
```

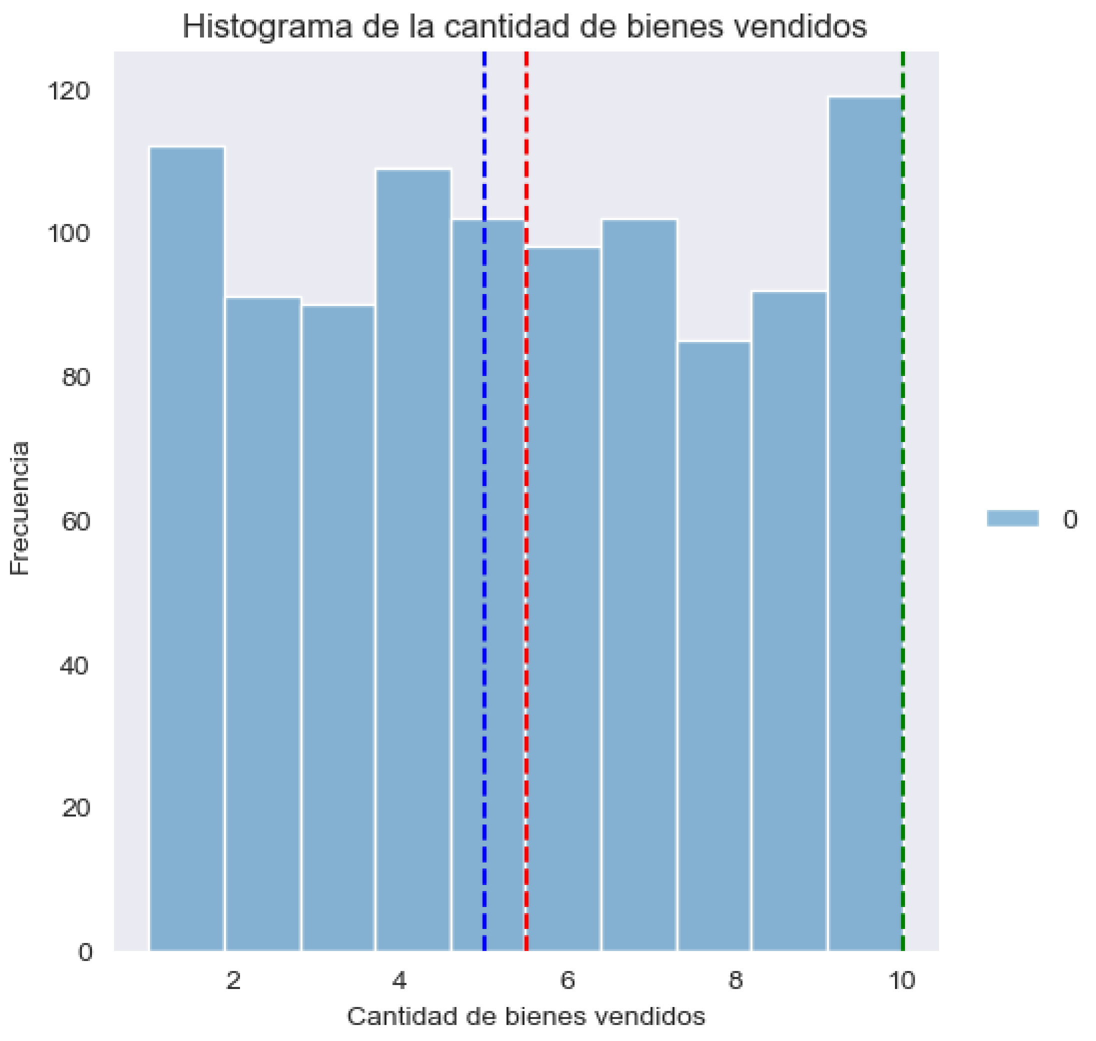
```
La media de la cantidad de bienes vendidos es = 5.51
La mediana de la cantidad de bienes vendidos es = 5.00
la moda de la cantidad de bienes vendidos es = 10.00
```

In [ ]:
```
plt.hist(quantity_np)
plt.title('Histograma de la Cantidad de Bienes Vendidos')
plt.show()
```



In [ ]:
```
sns.set_style('dark')
sns.displot(quantity_np,bins=10)
plt.xlabel('Cantidad de bienes vendidos')
plt.ylabel('Frecuencia')
plt.title('Histograma de la cantidad de bienes vendidos')
plt.axvline(x=quantity_np.mean(),color='red', ls='--')
plt.axvline(x=mediana,color='blue',ls='--')
plt.axvline(x=moda,color='green',ls='--')
```

Out[ ]:
```
<matplotlib.lines.Line2D at 0x1e94d27a040>
```



Obtener el total promedio:

1) por ciudad

```
In [ ]: df_city_mean=df.groupby('City')['Total'].mean().rename('Average of total sales').s
        df_city_mean['Average of total sales']=df_city_mean['Average of total sales'].appl
        df_city_mean
```

Out[ ]:

	City	Average of total sales
0	Naypyitaw	337.10
1	Mandalay	319.87
2	Yangon	312.35

```
In [ ]: # Para no tener que cambiar el formato de cada línea de código,
        # utilizo la siguiente opción:
        pd.options.display.float_format = '{:.2f}'.format
```

2) por product line y ciudad

```
In [ ]: df_city_productline_mean = pd.pivot_table(df, values='Total', index=['City'], colu
        df_city_productline_mean.style.background_gradient(cmap='Greens')
```

Out[ ]:

Product line	Electronic accessories	Fashion accessories	Food and beverages	Health and beauty	Home and lifestyle	Sports and travel	Average of total sales
City							
Mandalay	310.026245	264.730911	304.297770	376.993585	350.983290	322.390306	319.872506
Naypyitaw	344.890445	331.693385	360.103864	319.525500	308.790067	350.265067	337.099715
Yangon	305.285225	320.245265	295.915526	268.037298	344.879931	328.350839	312.354031
Average of total sales	319.632538	305.089298	322.671517	323.643020	336.636956	332.065220	322.966749

3) por género

In [ ]:
```
df_gender_mean=df.groupby('Gender')['Total'].mean().rename('Average of total sales
df_gender_mean
```

Out[ ]:

	Gender	Average of total sales
0	Female	335.10
1	Male	310.79

4) por género y product line

In [ ]:
```
df_gender_productline_mean = pd.pivot_table(df, values='Total', index=['Gender'],
df_gender_productline_mean.style.background_gradient(cmap='Greens')
```

Out[ ]:

Product line	Electronic accessories	Fashion accessories	Food and beverages	Health and beauty	Home and lifestyle	Sports and travel	Average of total sales
Gender							
Female	322.643125	317.056250	368.565750	290.015414	380.213639	324.712739	335.095659
Male	316.691965	291.079207	273.499125	348.099460	294.136241	340.360327	310.789226
Average of total sales	319.632538	305.089298	322.671517	323.643020	336.636956	332.065220	322.966749

Usar el método rank para generar top 5 de:

In [ ]:
```
# Se añadirá una columna al final del dataframe con el número del Ranking.
```

1) ventas por ciudad

In [ ]:
```
city_list=list(df['City'].unique())
city_list
```

Out[ ]:
```
['Yangon', 'Naypyitaw', 'Mandalay']
```

```
In [ ]: city_list=list(df['City'].unique())

        df_cities=pd.DataFrame()

        for city in city_list:
            df_helper= df[df['City']==city]
            df_helper['Rank']=df_helper['Total'].rank(ascending=False)
            df_helper.sort_values(by='Rank',inplace=True)
            df_result=df_helper.head(5)
            df_cities=pd.concat([df_cities,df_result])

        df_cities
```

```
C:\Users\oscah\AppData\Local\Temp\ipykernel_18952\1332603943.py:7: SettingWithCopy
Warning:
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/stabl
e/user_guide/indexing.html#returning-a-view-versus-a-copy
  df_helper['Rank']=df_helper['Total'].rank(ascending=False)
C:\Users\oscah\AppData\Local\Temp\ipykernel_18952\1332603943.py:8: SettingWithCopy
Warning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stabl
e/user_guide/indexing.html#returning-a-view-versus-a-copy
  df_helper.sort_values(by='Rank',inplace=True)
C:\Users\oscah\AppData\Local\Temp\ipykernel_18952\1332603943.py:7: SettingWithCopy
Warning:
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/stabl
e/user_guide/indexing.html#returning-a-view-versus-a-copy
  df_helper['Rank']=df_helper['Total'].rank(ascending=False)
C:\Users\oscah\AppData\Local\Temp\ipykernel_18952\1332603943.py:8: SettingWithCopy
Warning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stabl
e/user_guide/indexing.html#returning-a-view-versus-a-copy
  df_helper.sort_values(by='Rank',inplace=True)
C:\Users\oscah\AppData\Local\Temp\ipykernel_18952\1332603943.py:7: SettingWithCopy
Warning:
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/stabl
e/user_guide/indexing.html#returning-a-view-versus-a-copy
  df_helper['Rank']=df_helper['Total'].rank(ascending=False)
C:\Users\oscah\AppData\Local\Temp\ipykernel_18952\1332603943.py:8: SettingWithCopy
Warning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stabl
e/user_guide/indexing.html#returning-a-view-versus-a-copy
  df_helper.sort_values(by='Rank',inplace=True)
```

Out[ ]:

	Invoice ID	Branch	City	Customer type	Gender	Product line	Unit price	Quantity	Tax 5%	Total
167	687-47-8271	A	Yangon	Normal	Male	Fashion accessories	98.98	10	49.49	1039.29
429	325-77-6186	A	Yangon	Member	Female	Home and lifestyle	90.65	10	45.33	951.83
959	384-59-6655	A	Yangon	Member	Female	Food and beverages	98.66	9	44.40	932.34
105	704-48-3927	A	Yangon	Member	Male	Electronic accessories	88.67	10	44.34	931.03
529	827-77-7633	A	Yangon	Normal	Male	Sports and travel	98.09	9	44.14	926.95
350	860-79-0874	C	Naypyitaw	Member	Female	Fashion accessories	99.30	10	49.65	1042.65
557	283-26-5248	C	Naypyitaw	Member	Female	Food and beverages	98.52	10	49.26	1034.46
699	751-41-9720	C	Naypyitaw	Normal	Male	Home and lifestyle	97.50	10	48.75	1023.75
422	271-88-8734	C	Naypyitaw	Member	Female	Fashion accessories	97.21	10	48.60	1020.71
166	234-65-2137	C	Naypyitaw	Normal	Male	Home and lifestyle	95.58	10	47.79	1003.59
996	303-96-2227	B	Mandalay	Normal	Female	Home and lifestyle	97.38	10	48.69	1022.49
792	744-16-7898	B	Mandalay	Normal	Female	Home and lifestyle	97.37	10	48.69	1022.38
122	219-22-9386	B	Mandalay	Member	Male	Sports and travel	99.96	9	44.98	944.62
209	817-69-8206	B	Mandalay	Normal	Female	Electronic accessories	99.73	9	44.88	942.45
96	766-85-7061	B	Mandalay	Normal	Male	Health and beauty	87.87	10	43.94	922.63

2) ventas por member

In [ ]:
```
customer_type_list=list(df['Customer type'].unique())
customer_type_list
```

Out[ ]:
```
['Member', 'Normal']
```

In [ ]:
```
customer_type_list=list(df['Customer type'].unique())

df_customer_type=pd.DataFrame()

for customer_type in customer_type_list:
    df_helper= df[df['Customer type']==customer_type]
    df_helper['Rank']=df_helper['Total'].rank(ascending=False)
    df_helper.sort_values(by='Rank',inplace=True)
    df_result=df_helper.head(5)
    df_customer_type=pd.concat([df_customer_type,df_result])

df_customer_type
```

```
C:\Users\oscah\AppData\Local\Temp\ipykernel_18952\1120970538.py:7: SettingWithCopy
Warning:
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/stabl
e/user_guide/indexing.html#returning-a-view-versus-a-copy
  df_helper['Rank']=df_helper['Total'].rank(ascending=False)
C:\Users\oscah\AppData\Local\Temp\ipykernel_18952\1120970538.py:8: SettingWithCopy
Warning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stabl
e/user_guide/indexing.html#returning-a-view-versus-a-copy
  df_helper.sort_values(by='Rank',inplace=True)
C:\Users\oscah\AppData\Local\Temp\ipykernel_18952\1120970538.py:7: SettingWithCopy
Warning:
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/stabl
e/user_guide/indexing.html#returning-a-view-versus-a-copy
  df_helper['Rank']=df_helper['Total'].rank(ascending=False)
C:\Users\oscah\AppData\Local\Temp\ipykernel_18952\1120970538.py:8: SettingWithCopy
Warning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stabl
e/user_guide/indexing.html#returning-a-view-versus-a-copy
  df_helper.sort_values(by='Rank',inplace=True)
```

Out[ ]:

	Invoice ID	Branch	City	Customer type	Gender	Product line	Unit price	Quantity	Tax 5%	Total
350	860-79-0874	C	Naypyitaw	Member	Female	Fashion accessories	99.30	10	49.65	1042.65
557	283-26-5248	C	Naypyitaw	Member	Female	Food and beverages	98.52	10	49.26	1034.46
422	271-88-8734	C	Naypyitaw	Member	Female	Fashion accessories	97.21	10	48.60	1020.71
429	325-77-6186	A	Yangon	Member	Female	Home and lifestyle	90.65	10	45.33	951.83
141	280-17-4359	C	Naypyitaw	Member	Male	Health and beauty	90.50	10	45.25	950.25
167	687-47-8271	A	Yangon	Normal	Male	Fashion accessories	98.98	10	49.49	1039.29
699	751-41-9720	C	Naypyitaw	Normal	Male	Home and lifestyle	97.50	10	48.75	1023.75
996	303-96-2227	B	Mandalay	Normal	Female	Home and lifestyle	97.38	10	48.69	1022.49
792	744-16-7898	B	Mandalay	Normal	Female	Home and lifestyle	97.37	10	48.69	1022.38
166	234-65-2137	C	Naypyitaw	Normal	Male	Home and lifestyle	95.58	10	47.79	1003.59

3) ventas por payment

In [ ]:
```python
payment_list=list(df['Payment'].unique())
payment_list
```

Out[ ]:
```
['Ewallet', 'Cash', 'Credit card']
```

In [ ]:
```python
payment_list=list(df['Payment'].unique())

df_payment=pd.DataFrame()


for payment in payment_list:
    df_helper= df[df['Payment']==payment]
    df_helper['Rank']=df_helper['Total'].rank(ascending=False)
    df_helper.sort_values(by='Rank',inplace=True)
    df_result=df_helper.head(5)
    df_payment=pd.concat([df_payment,df_result])

df_payment
```

```
C:\Users\oscah\AppData\Local\Temp\ipykernel_18952\3926951295.py:8: SettingWithCopy
Warning:
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/stabl
e/user_guide/indexing.html#returning-a-view-versus-a-copy
  df_helper['Rank']=df_helper['Total'].rank(ascending=False)
C:\Users\oscah\AppData\Local\Temp\ipykernel_18952\3926951295.py:9: SettingWithCopy
Warning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stabl
e/user_guide/indexing.html#returning-a-view-versus-a-copy
  df_helper.sort_values(by='Rank',inplace=True)
C:\Users\oscah\AppData\Local\Temp\ipykernel_18952\3926951295.py:8: SettingWithCopy
Warning:
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/stabl
e/user_guide/indexing.html#returning-a-view-versus-a-copy
  df_helper['Rank']=df_helper['Total'].rank(ascending=False)
C:\Users\oscah\AppData\Local\Temp\ipykernel_18952\3926951295.py:9: SettingWithCopy
Warning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stabl
e/user_guide/indexing.html#returning-a-view-versus-a-copy
  df_helper.sort_values(by='Rank',inplace=True)
C:\Users\oscah\AppData\Local\Temp\ipykernel_18952\3926951295.py:8: SettingWithCopy
Warning:
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/stabl
e/user_guide/indexing.html#returning-a-view-versus-a-copy
  df_helper['Rank']=df_helper['Total'].rank(ascending=False)
C:\Users\oscah\AppData\Local\Temp\ipykernel_18952\3926951295.py:9: SettingWithCopy
Warning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stabl
e/user_guide/indexing.html#returning-a-view-versus-a-copy
  df_helper.sort_values(by='Rank',inplace=True)
```

Out[ ]:

	Invoice ID	Branch	City	Customer type	Gender	Product line	Unit price	Quantity	Tax 5%	Total
557	283-26-5248	C	Naypyitaw	Member	Female	Food and beverages	98.52	10	49.26	1034.46
699	751-41-9720	C	Naypyitaw	Normal	Male	Home and lifestyle	97.50	10	48.75	1023.75
996	303-96-2227	B	Mandalay	Normal	Female	Home and lifestyle	97.38	10	48.69	1022.49
429	325-77-6186	A	Yangon	Member	Female	Home and lifestyle	90.65	10	45.33	951.83
435	751-69-0068	C	Naypyitaw	Normal	Male	Sports and travel	99.24	9	44.66	937.82
166	234-65-2137	C	Naypyitaw	Normal	Male	Home and lifestyle	95.58	10	47.79	1003.59
357	554-42-2417	C	Naypyitaw	Normal	Female	Sports and travel	95.44	10	47.72	1002.12
141	280-17-4359	C	Naypyitaw	Member	Male	Health and beauty	90.50	10	45.25	950.25
941	702-83-5291	C	Naypyitaw	Member	Male	Fashion accessories	99.82	9	44.92	943.30
611	277-35-5865	C	Naypyitaw	Member	Female	Food and beverages	98.97	9	44.54	935.27
350	860-79-0874	C	Naypyitaw	Member	Female	Fashion accessories	99.30	10	49.65	1042.65
167	687-47-8271	A	Yangon	Normal	Male	Fashion accessories	98.98	10	49.49	1039.29
792	744-16-7898	B	Mandalay	Normal	Female	Home and lifestyle	97.37	10	48.69	1022.38
422	271-88-8734	C	Naypyitaw	Member	Female	Fashion accessories	97.21	10	48.60	1020.71
122	219-22-9386	B	Mandalay	Member	Male	Sports and travel	99.96	9	44.98	944.62

Obtener además el % de aporte de cada categoría.

por branch

In [ ]:
```
df_branchpct=df.groupby('Branch')['Total'].sum().reset_index()
df_branchpct['Percent']=(df_branchpct['Total']/df_branchpct['Total'].sum())*100
df_branchpct
```

Out[ ]:

	Branch	Total	Percent
0	A	106200.37	32.88
1	B	106197.67	32.88
2	C	110568.71	34.24

In [ ]:
```
import plotly.express as px
fig = px.pie(df,names='Branch',values='Total', title='Porcentaje de cada branch co
fig.show()
```

por ciudad

In [ ]:
```
df_citypct=df.groupby('City')['Total'].sum().reset_index()
df_citypct['Percent']=(df_citypct['Total']/df_citypct['Total'].sum())*100
df_citypct
```

Out[ ]:

	City	Total	Percent
0	Mandalay	106197.67	32.88
1	Naypyitaw	110568.71	34.24
2	Yangon	106200.37	32.88

In [ ]:
```
fig = px.pie(df,names='City',values='Total', title='Porcentaje de cada ciudad con
fig.show()
```

por tipo de cliente

In [ ]:
```
df_customer_typepct=df.groupby('Customer type')['Total'].sum().reset_index()
df_customer_typepct['Percent']=(df_customer_typepct['Total']/df_customer_typepct['
df_customer_typepct
```

Out[ ]:

	Customer type	Total	Percent
0	Member	164223.44	50.85
1	Normal	158743.30	49.15

In [ ]:
```
fig = px.pie(df,names='Customer type',values='Total', title='Porcentaje del tipo d
fig.show()
```

por género

In [ ]:
```
df_genderpct=df.groupby('Gender')['Total'].sum().reset_index()
df_genderpct['Percent']=(df_genderpct['Total']/df_genderpct['Total'].sum())*100
df_genderpct
```

Out[ ]:

	Gender	Total	Percent
0	Female	167882.92	51.98
1	Male	155083.82	48.02

In [ ]:
```
fig = px.pie(df,names='Gender',values='Total', title='Porcentaje por genero con re
fig.show()
```

por línea de productos

In [ ]:
```
df_product_linepct=df.groupby('Product line')['Total'].sum().reset_index()
df_product_linepct['Percent']=(df_product_linepct['Total']/df_product_linepct['Tot
df_product_linepct
```

Out[ ]:

	Product line	Total	Percent
0	Electronic accessories	54337.53	16.82
1	Fashion accessories	54305.89	16.81
2	Food and beverages	56144.84	17.38
3	Health and beauty	49193.74	15.23
4	Home and lifestyle	53861.91	16.68
5	Sports and travel	55122.83	17.07

In [ ]:
```
fig = px.pie(df,names='Product line',values='Total', title='Porcentaje de cada lin
fig.show()
```

por metodo de pago

In [ ]:
```
df_paymentpct=df.groupby('Payment')['Total'].sum().reset_index()
df_paymentpct['Percent']=(df_paymentpct['Total']/df_paymentpct['Total'].sum())*100
df_paymentpct
```

Out[ ]:

	Payment	Total	Percent
0	Cash	112206.57	34.74
1	Credit card	100767.07	31.20
2	Ewallet	109993.11	34.06

In [ ]:
```
fig = px.pie(df,names='Payment',values='Total', title='Porcentaje por forma de pag
fig.show()
```

correlación entre la hora (sin minutos) y el total

In [ ]:
```
df[['Hours','Minutes']]=df.Time.str.split(':',expand=True)
df.sample(5)
```

Out[ ]:

	Invoice ID	Branch	City	Customer type	Gender	Product line	Unit price	Quantity	Tax 5%	Total	
488	556-72-8512	C	Naypyitaw	Normal	Male	Home and lifestyle	22.96	1	1.15	24.11	1
415	268-03-6164	B	Mandalay	Normal	Male	Health and beauty	96.11	1	4.81	100.92	1
96	766-85-7061	B	Mandalay	Normal	Male	Health and beauty	87.87	10	43.94	922.63	3
515	413-20-6708	C	Naypyitaw	Member	Female	Fashion accessories	51.47	1	2.57	54.04	3
915	717-96-4189	C	Naypyitaw	Normal	Female	Electronic accessories	35.49	6	10.65	223.59	

In [ ]:
```
df['Hours']=df['Hours'].astype('int64')
```

In [ ]:
```
df[['Hours','Total']].corr()

# No hay evidencias para justificar una correlación entre la hora y el total de ve
```

Out[ ]:

	Hours	Total
Hours	1.00	-0.00
Total	-0.00	1.00

correlación entre unit_price y el rating de la transacción

In [ ]:
```
df[['Unit price','Rating']].corr()

# No hay evidencias que justifiquen una posible correlación.
# entre el precio unitario y el rating de la transacción.
```

Out[ ]:

	Unit price	Rating
Unit price	1.00	-0.01
Rating	-0.01	1.00