Performance de vendas de 3 filiais de um supermercado

In [ ]:
```
Toda organização necessita constantemente analisar a performance da empresa para identificar gargalos ou até mesmo realizar melhor
A análise preditiva a seguir foi extraída através do dataset https://www.kaggle.com/datasets/aungpyaeap/supermarket-sales publicad
Nele contém os dados históricos de 3 filiais de uma rede de supermercado pelo período de 3 meses.
A partir desses dados, fiz uma análise exploratória para identificar insights que servirão de base para tomada de decisão.
```

In [1]:
```
#carregando a biblioteca pandas
import pandas as pd
```

In [2]:
```
#lendo o repositório
df = pd.read_csv('supermarket_sales - Sheet1.csv')
```

In [3]:
```
#verificando os primeiros dados do dataset para ter uma melhor visão do cenário.
df.head(20)
```

Out[3]:

	Invoice ID	Branch	City	Customer type	Gender	Product line	Unit price	Quantity	Tax 5%	Total	Date	Time	Payment	cogs	gross margin percentage	gros incom
0	750-67-8428	A	Yangon	Member	Female	Health and beauty	74.69	7	26.1415	548.9715	1/5/2019	13:08	Ewallet	522.83	4.761905	26.141
1	226-31-3081	C	Naypyitaw	Normal	Female	Electronic accessories	15.28	5	3.8200	80.2200	3/8/2019	10:29	Cash	76.40	4.761905	3.820
2	631-41-3108	A	Yangon	Normal	Male	Home and lifestyle	46.33	7	16.2155	340.5255	3/3/2019	13:23	Credit card	324.31	4.761905	16.215
3	123-19-1176	A	Yangon	Member	Male	Health and beauty	58.22	8	23.2880	489.0480	1/27/2019	20:33	Ewallet	465.76	4.761905	23.288

	Invoice ID	Branch	City	Customer type	Gender	Product line	Unit price	Quantity	Tax 5%	Total	Date	Time	Payment	cogs	gross margin percentage	gros incom
4	373-73-7910	A	Yangon	Normal	Male	Sports and travel	86.31	7	30.2085	634.3785	2/8/2019	10:37	Ewallet	604.17	4.761905	30.208
5	699-14-3026	C	Naypyitaw	Normal	Male	Electronic accessories	85.39	7	29.8865	627.6165	3/25/2019	18:30	Ewallet	597.73	4.761905	29.886
6	355-53-5943	A	Yangon	Member	Female	Electronic accessories	68.84	6	20.6520	433.6920	2/25/2019	14:36	Ewallet	413.04	4.761905	20.652
7	315-22-5665	C	Naypyitaw	Normal	Female	Home and lifestyle	73.56	10	36.7800	772.3800	2/24/2019	11:38	Ewallet	735.60	4.761905	36.780
8	665-32-9167	A	Yangon	Member	Female	Health and beauty	36.26	2	3.6260	76.1460	1/10/2019	17:15	Credit card	72.52	4.761905	3.626
9	692-92-5582	B	Mandalay	Member	Female	Food and beverages	54.84	3	8.2260	172.7460	2/20/2019	13:27	Credit card	164.52	4.761905	8.226
10	351-62-0822	B	Mandalay	Member	Female	Fashion accessories	14.48	4	2.8960	60.8160	2/6/2019	18:07	Ewallet	57.92	4.761905	2.896
11	529-56-3974	B	Mandalay	Member	Male	Electronic accessories	25.51	4	5.1020	107.1420	3/9/2019	17:03	Cash	102.04	4.761905	5.102
12	365-64-0515	A	Yangon	Normal	Female	Electronic accessories	46.95	5	11.7375	246.4875	2/12/2019	10:25	Ewallet	234.75	4.761905	11.737
13	252-56-2699	A	Yangon	Normal	Male	Food and beverages	43.19	10	21.5950	453.4950	2/7/2019	16:48	Ewallet	431.90	4.761905	21.595

	Invoice ID	Branch	City	Customer type	Gender	Product line	Unit price	Quantity	Tax 5%	Total	Date	Time	Payment	cogs	gross margin percentage	gros incom
14	829-34-3910	A	Yangon	Normal	Female	Health and beauty	71.38	10	35.6900	749.4900	3/29/2019	19:21	Cash	713.80	4.761905	35.69
15	299-46-1805	B	Mandalay	Member	Female	Sports and travel	93.72	6	28.1160	590.4360	1/15/2019	16:19	Cash	562.32	4.761905	28.11
16	656-95-9349	A	Yangon	Member	Female	Health and beauty	68.93	7	24.1255	506.6355	3/11/2019	11:03	Credit card	482.51	4.761905	24.12
17	765-26-6951	A	Yangon	Normal	Male	Sports and travel	72.61	6	21.7830	457.4430	1/1/2019	10:39	Credit card	435.66	4.761905	21.78
18	329-62-1586	A	Yangon	Normal	Male	Food and beverages	54.67	3	8.2005	172.2105	1/21/2019	18:00	Credit card	164.01	4.761905	8.20
19	319-50-3348	B	Mandalay	Normal	Female	Home and lifestyle	40.30	2	4.0300	84.6300	3/11/2019	15:30	Ewallet	80.60	4.761905	4.03

In [4]:
```python
# Identificando as colunas.
df.columns
```

Out[4]:
```
Index(['Invoice ID', 'Branch', 'City', 'Customer type', 'Gender',
       'Product line', 'Unit price', 'Quantity', 'Tax 5%', 'Total', 'Date',
       'Time', 'Payment', 'cogs', 'gross margin percentage', 'gross income',
       'Rating'],
      dtype='object')
```

In [5]:
```python
# consultando informações das dimensões do dataset
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
```

In [ ]:

```
Nesse caso terá que alterar o tipo do Dtype da coluna data, pois posteriormente irei trabalhar com essa coluna.
Caso não alterado o codigo pode apresentar erros.
```

In [6]:

```
df["Date"] = pd.to_datetime(df["Date"])
```

In [7]:

```
# Pronto o formato da coluna data foi alterado para o tipo "datetime"
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   datetime64[ns]
 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: datetime64[ns](1), float64(7), int64(1), object(8)
memory usage: 132.9+ KB
```

In [8]:

```
#exibindo algumas informações estatisticas relevantes
df.describe()
```

Out[8]:

	Unit price	Quantity	Tax 5%	Total	cogs	gross margin percentage	gross income	Rating
count	1000.000000	1000.000000	1000.000000	1000.000000	1000.00000	1.000000e+03	1000.000000	1000.00000
mean	55.672130	5.510000	15.379369	322.966749	307.58738	4.761905e+00	15.379369	6.97270
std	26.494628	2.923431	11.708825	245.885335	234.17651	6.131498e-14	11.708825	1.71858
min	10.080000	1.000000	0.508500	10.678500	10.17000	4.761905e+00	0.508500	4.00000
25%	32.875000	3.000000	5.924875	124.422375	118.49750	4.761905e+00	5.924875	5.50000
50%	55.230000	5.000000	12.088000	253.848000	241.76000	4.761905e+00	12.088000	7.00000
75%	77.935000	8.000000	22.445250	471.350250	448.90500	4.761905e+00	22.445250	8.50000
max	99.960000	10.000000	49.650000	1042.650000	993.00000	4.761905e+00	49.650000	10.00000

In [9]:

```
# identificando as marcas mais vendidos em todas as lojas
df.groupby('Branch')['Quantity'].sum().sort_values(ascending=False)
```

Out[9]:

```
Branch
A    1859
C    1831
```

```
B    1820
Name: Quantity, dtype: int64
```

In [10]:

```
# Identificando a categoria mais vendida.
df.groupby('Product line')['Quantity'].sum().sort_values(ascending=False)
```

Out[10]:

```
Product line
Electronic accessories    971
Food and beverages        952
Sports and travel         920
Home and lifestyle        911
Fashion accessories       902
Health and beauty         854
Name: Quantity, dtype: int64
```

In [11]:

```
# Identificando a loja que mais vendeu.
df.groupby('City')['Quantity'].sum().sort_values(ascending=False)
```

Out[11]:

```
City
Yangon       1859
Naypyitaw    1831
Mandalay     1820
Name: Quantity, dtype: int64
```

In [12]:

```
# Identificando genero que mais compram
df.groupby('Gender')['Quantity'].sum().sort_values(ascending=False)
```

Out[12]:

```
Gender
Female    2869
Male      2641
Name: Quantity, dtype: int64
```

In [13]:

```
# identificando o Mês que teve mais vendas
df.groupby(df["Date"].dt.month)["Unit price"].sum()
```

Out[13]:

```
Date
1    19753.89
2    17159.52
3    18758.72
Name: Unit price, dtype: float64
```

In [14]:
```
# Analisando as datas onde mais teve vendas
df.groupby('Date')['Quantity'].sum().sort_values(ascending=False)
```

Out[14]:
```
Date
2019-02-07    128
2019-03-14    117
2019-02-15    106
2019-03-05    103
2019-03-09     99
             ... 
2019-02-13     31
2019-02-28     30
2019-03-18     27
2019-02-18     24
2019-02-21     18
Name: Quantity, Length: 89, dtype: int64
```

In [15]:
```
# forma de Pagamentos mais utilizada.
df.groupby('Payment').count()
```

Out[15]:

	Invoice ID	Branch	City	Customer type	Gender	Product line	Unit price	Quantity	Tax 5%	Total	Date	Time	cogs	gross margin percentage	gross income	Rating
Payment																
Cash	344	344	344	344	344	344	344	344	344	344	344	344	344	344	344	344
Credit card	311	311	311	311	311	311	311	311	311	311	311	311	311	311	311	311
Ewallet	345	345	345	345	345	345	345	345	345	345	345	345	345	345	345	345

In [16]:
```
# analisando o nivel de satisfação dos cliente com a compra efetuada.
df.groupby('Rating').count()
```

Out[16]:

	Invoice ID	Branch	City	Customer type	Gender	Product line	Unit price	Quantity	Tax 5%	Total	Date	Time	Payment	cogs	gross margin percentage	gross income
Rating																
4.0	11	11	11	11	11	11	11	11	11	11	11	11	11	11	11	11

	Invoice ID	Branch	City	Customer type	Gender	Product line	Unit price	Quantity	Tax 5%	Total	Date	Time	Payment	cogs	gross margin percentage	gross income
Rating																
4.1	17	17	17	17	17	17	17	17	17	17	17	17	17	17	17	17
4.2	22	22	22	22	22	22	22	22	22	22	22	22	22	22	22	22
4.3	18	18	18	18	18	18	18	18	18	18	18	18	18	18	18	18
4.4	17	17	17	17	17	17	17	17	17	17	17	17	17	17	17	17
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
9.6	17	17	17	17	17	17	17	17	17	17	17	17	17	17	17	17
9.7	14	14	14	14	14	14	14	14	14	14	14	14	14	14	14	14
9.8	19	19	19	19	19	19	19	19	19	19	19	19	19	19	19	19
9.9	16	16	16	16	16	16	16	16	16	16	16	16	16	16	16	16
10.0	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5

61 rows × 16 columns

In [17]:

```
# identificando o valor do produto mais caro
df[['Unit price']].idxmax()
```

Out[17]:

```
Unit price    122
dtype: int64
```

In [18]:

```
# identificando o valor do produto mais barato
df[['Unit price']].min()
```

Out[18]:

```
Unit price    10.08
dtype: float64
```

Conclusão

In [ ]:

```
Após fazer uma análise dos dados obtivemos os seguintes resultados; as marcas A e C foram as marcas mais vendidas, as categorias A
```

```
bebidas, Esportes e viagens tiveram melhores desempenho.
Yangon e Naypyitaw foram as cidades que mais obtiverão vendas. Os principais compradores foram do sexo feminino.
Janeiro e fevereiro foram os meses que mais venderam.
 Os períodos onde houve mais compras foram do 5° ao 15º dia dos meses de fevereiro e março. Pagamentos por Ewallet e Dinheiro fora
entre os consumidores.
Medindo o nível de satisfação, o maior número de pessoas avaliou com a nota 4,2 e só apenas 5 entrevistados deu nota 5,0.
O Produto mais caro registrado foi 122.00 e o mais barato foi 10.08.
```