In [1]:

```
!pip install panda
```

```
Requirement already satisfied: panda in c:\users\batyrzhan\anaconda3\lib\site-pac
kages (0.3.1)
Requirement already satisfied: setuptools in c:\users\batyrzhan\anaconda3\lib\sit
e-packages (from panda) (80.9.0)
Requirement already satisfied: requests in c:\users\batyrzhan\anaconda3\lib\site-
packages (from panda) (2.32.5)
Requirement already satisfied: charset_normalizer<4,>=2 in c:\users\batyrzhan\ana
conda3\lib\site-packages (from requests->panda) (3.4.4)
Requirement already satisfied: idna<4,>=2.5 in c:\users\batyrzhan\anaconda3\lib\s
ite-packages (from requests->panda) (3.11)
Requirement already satisfied: urllib3<3,>=1.21.1 in c:\users\batyrzhan\anaconda3
\lib\site-packages (from requests->panda) (2.5.0)
Requirement already satisfied: certifi>=2017.4.17 in c:\users\batyrzhan\anaconda3
\lib\site-packages (from requests->panda) (2026.1.4)
```

In [2]:

```
import pandas as pd
```

In [8]:

```
df = pd.read_csv(r'C:\Users\BATYRZHAN\Desktop\ml\SuperMarket Analysis.csv')
```

Q1. Load the (SuperMarket Analysis.csv) dataset and display the first 5 rows.

In [9]:

```
df.head(5)
```

Out[9]:

	Invoice ID	Branch	City	Customer type	Gender	Product line	Unit price	Quantity	Tax 5%
0	750-67-8428	Alex	Yangon	Member	Female	Health and beauty	74.69	7	26.1415
1	226-31-3081	Giza	Naypyitaw	Normal	Female	Electronic accessories	15.28	5	3.8200
2	631-41-3108	Alex	Yangon	Normal	Female	Home and lifestyle	46.33	7	16.2155
3	123-19-1176	Alex	Yangon	Member	Female	Health and beauty	58.22	8	23.2880
4	373-73-7910	Alex	Yangon	Member	Female	Sports and travel	86.31	7	30.2085

Q2. Display the dataset shape (rows and columns).

```
In [10]: df.shape
```

```
Out[10]: (1000, 17)
```

Q3. List all column names.

```
In [11]: df.columns
```

```
Out[11]: Index(['Invoice ID', 'Branch', 'City', 'Customer type', 'Gender',
                'Product line', 'Unit price', 'Quantity', 'Tax 5%', 'Sales', 'Date',
                'Time', 'Payment', 'cogs', 'gross margin percentage', 'gross income',
                'Rating'],
               dtype='object')
```

Q4. Identify categorical and numerical columns.

```
In [12]: categorical = df.select_dtypes(include='object').columns
         numerical = df.select_dtypes(include='number').columns
         print(categorical)
         print(numerical)
```

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

Q5. Check for missing values in each column.

```
In [13]: df.isna().sum()
```

```
Out[13]: Invoice ID                 0
         Branch                     0
         City                       0
         Customer type              0
         Gender                     0
         Product line               0
         Unit price                 0
         Quantity                   0
         Tax 5%                     0
         Sales                      0
         Date                       0
         Time                       0
         Payment                    0
         cogs                       0
         gross margin percentage    0
         gross income               0
         Rating                     0
         dtype: int64
```

Q6. Display the data types of each column.

```
In [14]: df.dtypes
```

```
Out[14]: Invoice ID                  object
         Branch                      object
         City                        object
         Customer type               object
         Gender                      object
         Product line                object
         Unit price                 float64
         Quantity                     int64
         Tax 5%                     float64
         Sales                      float64
         Date                        object
         Time                        object
         Payment                     object
         cogs                       float64
         gross margin percentage    float64
         gross income               float64
         Rating                     float64
         dtype: object
```

Q7. Show summary statistics for numerical columns (use pandas method).

```
In [15]: df.describe()
```

Out[15]:

	Unit price	Quantity	Tax 5%	Sales	cogs	gross margin percentage	
count	1000.000000	1000.000000	1000.000000	1000.000000	1000.00000	1.000000e+03	1
mean	55.672130	5.510000	15.379369	322.966749	307.58738	4.761905e+00	
std	26.494628	2.923431	11.708825	245.885335	234.17651	6.131498e-14	
min	10.080000	1.000000	0.508500	10.678500	10.17000	4.761905e+00	
25%	32.875000	3.000000	5.924875	124.422375	118.49750	4.761905e+00	
50%	55.230000	5.000000	12.088000	253.848000	241.76000	4.761905e+00	
75%	77.935000	8.000000	22.445250	471.350250	448.90500	4.761905e+00	
max	99.960000	10.000000	49.650000	1042.650000	993.00000	4.761905e+00	

Q8. Filter rows where Sales > 500.

In [16]:

```
df[df['Sales'] > 500]
```

Out[16]:

	Invoice ID	Branch	City	Customer type	Gender	Product line	Unit price	Quantity	Tax 5
0	750-67-8428	Alex	Yangon	Member	Female	Health and beauty	74.69	7	26.14
4	373-73-7910	Alex	Yangon	Member	Female	Sports and travel	86.31	7	30.20
5	699-14-3026	Giza	Naypyitaw	Member	Female	Electronic accessories	85.39	7	29.88
7	315-22-5665	Giza	Naypyitaw	Member	Female	Home and lifestyle	73.56	10	36.78
14	829-34-3910	Alex	Yangon	Member	Female	Health and beauty	71.38	10	35.69
...	...	...	...	...	...	...	...	...	
988	267-62-7380	Giza	Naypyitaw	Member	Male	Electronic accessories	82.34	10	41.17
989	430-53-4718	Cairo	Mandalay	Member	Male	Health and beauty	75.37	8	30.14
991	602-16-6955	Cairo	Mandalay	Normal	Female	Sports and travel	76.60	10	38.30
996	303-96-2227	Cairo	Mandalay	Normal	Female	Home and lifestyle	97.38	10	48.69
999	849-09-3807	Alex	Yangon	Member	Female	Fashion accessories	88.34	7	30.91

227 rows × 17 columns

Q9. Filter sales in City = “Yangon” and Sales > 200.

In [17]:

```
df[(df['City'] == 'Yangon') & (df['Sales'] > 200)]
```

Out[17]:

	Invoice ID	Branch	City	Customer type	Gender	Product line	Unit price	Quantity	Tax 5%
0	750-67-8428	Alex	Yangon	Member	Female	Health and beauty	74.69	7	26.1415
2	631-41-3108	Alex	Yangon	Normal	Female	Home and lifestyle	46.33	7	16.2155
3	123-19-1176	Alex	Yangon	Member	Female	Health and beauty	58.22	8	23.2880
4	373-73-7910	Alex	Yangon	Member	Female	Sports and travel	86.31	7	30.2085
6	355-53-5943	Alex	Yangon	Member	Female	Electronic accessories	68.84	6	20.6520
...	...	...	...	...	...	...	...	...	...
976	221-25-5073	Alex	Yangon	Normal	Female	Food and beverages	74.66	4	14.9320
981	809-46-1866	Alex	Yangon	Normal	Male	Health and beauty	58.15	4	11.6300
982	139-32-4183	Alex	Yangon	Member	Female	Sports and travel	97.48	9	43.8660
990	886-18-2897	Alex	Yangon	Normal	Female	Food and beverages	56.56	5	14.1400
999	849-09-3807	Alex	Yangon	Member	Female	Fashion accessories	88.34	7	30.9190

204 rows × 17 columns

Q10. Sort all orders by Sales in descending order.

In [18]:

```
df.sort_values(by='Sales', ascending=False)
```

Out[18]:

	Invoice ID	Branch	City	Customer type	Gender	Product line	Unit price	Quantity	Tax 5
350	860-79-0874	Giza	Naypyitaw	Member	Female	Fashion accessories	99.30	10	49.65
167	687-47-8271	Alex	Yangon	Normal	Male	Fashion accessories	98.98	10	49.49
557	283-26-5248	Giza	Naypyitaw	Member	Female	Food and beverages	98.52	10	49.26
699	751-41-9720	Giza	Naypyitaw	Normal	Male	Home and lifestyle	97.50	10	48.75
996	303-96-2227	Cairo	Mandalay	Normal	Female	Home and lifestyle	97.38	10	48.69
...	...	...	...	...	...	...	...	...	
402	236-86-3015	Giza	Naypyitaw	Member	Male	Home and lifestyle	13.98	1	0.69
443	192-98-7397	Giza	Naypyitaw	Normal	Male	Fashion accessories	12.78	1	0.63
223	279-62-1445	Giza	Naypyitaw	Member	Female	Fashion accessories	12.54	1	0.62
629	308-39-1707	Alex	Yangon	Normal	Female	Fashion accessories	12.09	1	0.60
822	784-21-9238	Giza	Naypyitaw	Member	Male	Sports and travel	10.17	1	0.50

1000 rows × 17 columns

Q11. Sort by Date (ascending) and then Time (ascending).

In [19]:

```
df.sort_values(by=['Date', 'Time'], ascending=[True, True])
```

Out[19]:

	Invoice ID	Branch	City	Customer type	Gender	Product line	Unit price	Quantity	Ta 5
17	765-26-6951	Alex	Yangon	Member	Female	Sports and travel	72.61	6	21.78
970	746-04-1077	Cairo	Mandalay	Member	Female	Food and beverages	84.63	10	42.31
839	271-77-8740	Giza	Naypyitaw	Member	Female	Sports and travel	29.22	6	8.76
523	133-14-7229	Giza	Naypyitaw	Normal	Male	Health and beauty	62.87	2	6.28
567	651-88-7328	Alex	Yangon	Normal	Female	Fashion accessories	65.74	9	29.58
...	...	...	...	...	...	...	...	...	.
122	219-22-9386	Cairo	Mandalay	Member	Female	Sports and travel	99.96	9	44.98
45	132-32-9879	Cairo	Mandalay	Member	Female	Electronic accessories	93.96	4	18.79
73	841-35-6630	Giza	Naypyitaw	Member	Female	Electronic accessories	75.91	6	22.77
234	157-13-5295	Alex	Yangon	Member	Male	Health and beauty	51.94	10	25.97
326	815-11-1168	Alex	Yangon	Member	Male	Food and beverages	99.78	5	24.94

1000 rows × 17 columns

Q12. Sort by Unit price and Quantity.

In [20]:

```
df.sort_values(['Unit price', 'Quantity']).head()
```

Out[20]:

	Invoice ID	Branch	City	Customer type	Gender	Product line	Unit price	Quantity	Tax 5%
944	333-23-2632	Alex	Yangon	Member	Male	Health and beauty	10.08	7	3.528
572	239-48-4278	Alex	Yangon	Member	Male	Food and beverages	10.13	7	3.545
784	516-77-6464	Giza	Naypyitaw	Member	Female	Health and beauty	10.16	5	2.540
822	784-21-9238	Giza	Naypyitaw	Member	Male	Sports and travel	10.17	1	0.508
881	115-38-7388	Giza	Naypyitaw	Member	Female	Fashion accessories	10.18	8	4.072

Q13. Calculate the total sales in “Sales” per Branch.

In [21]:

```
df.groupby('Branch')['Sales'].sum()
```

Out[21]:

```
Branch
Alex     106200.3705
Cairo    106197.6720
Giza     110568.7065
Name: Sales, dtype: float64
```

Q14. Calculate average Sales per City

In [22]:

```
df.groupby('City')['Sales'].mean()
```

Out[22]:

```
City
Mandalay     319.872506
Naypyitaw    337.099715
Yangon       312.354031
Name: Sales, dtype: float64
```

Q15. Find the quantities sold per product line.

In [23]:

```
df.groupby('Product line')['Quantity'].sum()
```

Out[23]:

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

Q16. Calculate average gross income per Gender

In [24]:

```
df.groupby('Gender')['gross income'].mean()
```

Out[24]:

```
Gender
Female    16.234829
Male      14.240749
Name: gross income, dtype: float64
```

Q17. Count number of sales per Payment method.

In [25]:

```
df['Payment'].value_counts()
```

Out[25]:

```
Payment
Ewallet        345
Cash           344
Credit card    311
Name: count, dtype: int64
```

Q18. Find maximum Sales per Branch.

In [26]:

```
df.groupby('Branch')['Sales'].max()
```

Out[26]:

```
Branch
Alex     1039.29
Cairo    1022.49
Giza     1042.65
Name: Sales, dtype: float64
```

In [27]:

```
df.sort_values(by='Sales', ascending=False)['Branch'].iloc[0]
```

Out[27]:

```
'Giza'
```

Q19. Find minimum Unit price per Product line.

In [28]:

```
df.groupby('Product line')['Unit price'].min()
```

```
Out[28]: Product line
         Electronic accessories    10.56
         Fashion accessories       10.18
         Food and beverages        10.13
         Health and beauty         10.08
         Home and lifestyle        10.53
         Sports and travel         10.17
         Name: Unit price, dtype: float64
```

Q20. Find the sum of gross income per Product line and Branch.

```
In [29]: df.groupby(['Product line', 'Branch'])['gross income'].sum()
```

```
Out[29]: Product line            Branch
         Electronic accessories  Alex       872.2435
                                 Cairo      811.9735
                                 Giza       903.2845
         Fashion accessories     Alex       777.7385
                                 Cairo      781.5865
                                 Giza      1026.6700
         Food and beverages      Alex       817.2905
                                 Cairo      724.5185
                                 Giza      1131.7550
         Health and beauty       Alex       599.8930
                                 Cairo      951.4600
                                 Giza       791.2060
         Home and lifestyle      Alex      1067.4855
                                 Cairo      835.6745
                                 Giza       661.6930
         Sports and travel       Alex       922.5095
                                 Cairo      951.8190
                                 Giza       750.5680
         Name: gross income, dtype: float64
```

21. What is the total quantities sold in Product line: “Electronic accessories”?

```
In [31]: total = df[df['Product line'] == 'Electronic accessories']['Quantity'].sum()
         print(total)
```

```
971
```

22. What is the average Sales for female customers?

```
In [32]: df[df['Gender'] == 'Female']['Sales'].mean()
```

```
Out[32]: np.float64(340.9314141856392)
```

23. What is the most expensive Unit price among Customer type members only?

In [33]:
```
df[df['Customer type'] == 'Member']['Unit price'].max()
```

Out[33]:
```
99.96
```

24. How many orders with Rating >= 9 ?

In [35]:
```
(df['Rating'] >= 9).sum()
```

Out[35]:
```
np.int64(166)
```

25. What is the total Sales for Payment “Credit card” in Branch C?

In [36]:
```
df[(df['Payment'] == 'Credit card') & (df['Branch'] == 'C')]['Sales'].sum()
```

Out[36]:
```
np.float64(0.0)
```