DMV1-Data Loading Storage and File Formats

October 26, 2023

[1]:
```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import numpy as np
import os
import warnings
warnings.filterwarnings('ignore')
```

```
C:\Users\hp\anaconda3\lib\site-packages\scipy\__init__.py:146: UserWarning: A
NumPy version >=1.16.5 and <1.23.0 is required for this version of SciPy
(detected version 1.25.2
  warnings.warn(f"A NumPy version >={np_minversion} and <{np_maxversion}"
```

[2]:
```
csv_data = pd.read_csv("C:/Users/hp/Downloads/Practical_Data/format1.csv")
excel_data = pd.read_excel("C:/Users/hp/Downloads/Practical_Data/format2.xlsx")
json_data = pd.read_json("C:/Users/hp/Downloads/Practical_Data/format3.json")
```

[3]:
```
csv_data.head()
```

[3]:

	Branch	City	Customer type	Gender	Product line	Unit price	\
0	A	Yangon	Member	Female	Health and beauty	74.69	
1	C	Naypyitaw	Normal	Female	Electronic accessories	15.28	
2	A	Yangon	Normal	Male	Home and lifestyle	46.33	
3	A	Yangon	Member	Male	Health and beauty	58.22	
4	A	Yangon	Normal	Male	Sports and travel	86.31	

	Quantity	Tax 5%	Total	Date	Time	Payment	cogs	\
0	7	26.1415	548.9715	1/5/2019	13:08	Ewallet	522.83	
1	5	3.8200	80.2200	3/8/2019	10:29	Cash	76.40	
2	7	16.2155	340.5255	3/3/2019	13:23	Credit card	324.31	
3	8	23.2880	489.0480	1/27/2019	20:33	Ewallet	465.76	
4	7	30.2085	634.3785	2/8/2019	10:37	Ewallet	604.17	

	gross margin percentage	gross income	Rating
0	4.761905	26.1415	9.1
1	4.761905	3.8200	9.6
2	4.761905	16.2155	7.4
3	4.761905	23.2880	8.4

```
4                4.761905       30.2085     5.3
```

[4]:
```
excel_data.head()
```

[4]:
```
  Branch       City Customer type  Gender            Product line  Unit price  \
0      A     Yangon        Normal    Male  Electronic accessories       51.69
1      B   Mandalay        Member  Female     Fashion accessories       54.73
2      B   Mandalay        Member    Male      Home and lifestyle       27.00
3      C  Naypyitaw        Normal  Female  Electronic accessories       30.24
4      B   Mandalay        Member  Female      Food and beverages       89.14

   Quantity   Tax 5%     Total       Date   Time      Payment    cogs  \
0         7  18.0915  379.9215  1/26/2019  18:22         Cash  361.83
1         7  19.1555  402.2655  3/14/2019  19:02  Credit card  383.11
2         9  12.1500  255.1500   3/2/2019  14:16         Cash  243.00
3         1   1.5120   31.7520   3/4/2019  15:44         Cash   30.24
4         4  17.8280  374.3880   1/7/2019  12:20  Credit card  356.56

   gross margin percentage  gross income  Rating
0                 4.761905       18.0915     5.5
1                 4.761905       19.1555     8.5
2                 4.761905       12.1500     4.8
3                 4.761905        1.5120     8.4
4                 4.761905       17.8280     7.8
```

[5]:
```
json_data.head()
```

[5]:
```
     Branch       City Customer type  Gender          Product line  Unit price  \
701      B   Mandalay        Normal    Male    Food and beverages       32.32
702      B   Mandalay        Member  Female   Fashion accessories       19.77
703      B   Mandalay        Member    Male     Health and beauty       80.47
704      B   Mandalay        Member  Female    Home and lifestyle       88.39
705      B   Mandalay        Normal    Male     Health and beauty       71.77

     Quantity   Tax 5%     Total       Date   Time      Payment    cogs  \
701         3   4.8480  101.8080 2019-03-27  19:11  Credit card   96.96
702        10   9.8850  207.5850 2019-02-27  18:57  Credit card  197.70
703         9  36.2115  760.4415 2019-01-06  11:18         Cash  724.23
704         9  39.7755  835.2855 2019-03-02  12:40         Cash  795.51
705         7  25.1195  527.5095 2019-03-29  14:06         Cash  502.39

     gross margin percentage  gross income  Rating
701                 4.761905        4.8480     4.3
702                 4.761905        9.8850     5.0
703                 4.761905       36.2115     9.2
704                 4.761905       39.7755     6.3
705                 4.761905       25.1195     8.9
```

[6]:
```
def merge(dataframes):
    if dataframes:
        return pd.concat(dataframes)
```

[7]:
```
df = merge([csv_data,excel_data,json_data])
```

[8]:
```
df.reset_index(inplace = True)
```

[9]:
```
df.head()
```

[9]:

	index	Branch	City	Customer type	Gender	Product line	\
0	0	A	Yangon	Member	Female	Health and beauty	
1	1	C	Naypyitaw	Normal	Female	Electronic accessories	
2	2	A	Yangon	Normal	Male	Home and lifestyle	
3	3	A	Yangon	Member	Male	Health and beauty	
4	4	A	Yangon	Normal	Male	Sports and travel	

	Unit price	Quantity	Tax 5%	Total	Date	Time	Payment	\
0	74.69	7	26.1415	548.9715	1/5/2019	13:08	Ewallet	
1	15.28	5	3.8200	80.2200	3/8/2019	10:29	Cash	
2	46.33	7	16.2155	340.5255	3/3/2019	13:23	Credit card	
3	58.22	8	23.2880	489.0480	1/27/2019	20:33	Ewallet	
4	86.31	7	30.2085	634.3785	2/8/2019	10:37	Ewallet	

	cogs	gross margin percentage	gross income	Rating
0	522.83	4.761905	26.1415	9.1
1	76.40	4.761905	3.8200	9.6
2	324.31	4.761905	16.2155	7.4
3	465.76	4.761905	23.2880	8.4
4	604.17	4.761905	30.2085	5.3

[10]:
```
df.describe()
```

[10]:

	index	Unit price	Quantity	Tax 5%	Total	\
count	1000.000000	1000.000000	1000.000000	1000.000000	1000.000000	
mean	376.650000	55.672130	5.510000	15.379369	322.966749	
std	324.157918	26.494628	2.923431	11.708825	245.885335	
min	0.000000	10.080000	1.000000	0.508500	10.678500	
25%	124.750000	32.875000	3.000000	5.924875	124.422375	
50%	249.500000	55.230000	5.000000	12.088000	253.848000	
75%	749.250000	77.935000	8.000000	22.445250	471.350250	
max	999.000000	99.960000	10.000000	49.650000	1042.650000	

	cogs	gross margin percentage	gross income	Rating
count	1000.00000	1.000000e+03	1000.000000	1000.00000
mean	307.58738	4.761905e+00	15.379369	6.97270
std	234.17651	6.131498e-14	11.708825	1.71858

```
min      10.17000             4.761905e+00      0.508500     4.00000
25%     118.49750             4.761905e+00      5.924875     5.50000
50%     241.76000             4.761905e+00     12.088000     7.00000
75%     448.90500             4.761905e+00     22.445250     8.50000
max     993.00000             4.761905e+00     49.650000    10.00000
```

[11]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1000 entries, 0 to 999
Data columns (total 17 columns):
 #   Column                   Non-Null Count  Dtype
---  ------                   --------------  -----
 0   index                    1000 non-null   int64
 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(2), object(8)
memory usage: 132.9+ KB
```

[12]:
```
#df.Date = pd.to_datetime(df.Date)
#df.Time = pd.to_datetime(df.Time)
```

[13]:
```
df.dtypes
```

[13]:
```
index                      int64
Branch                    object
City                      object
Customer type             object
Gender                    object
Product line              object
Unit price               float64
Quantity                   int64
Tax 5%                   float64
Total                    float64
```

```
Date                         object
Time                         object
Payment                      object
cogs                        float64
gross margin percentage     float64
gross income                float64
Rating                      float64
dtype: object
```

[14]:
```
df.isna().sum()
```

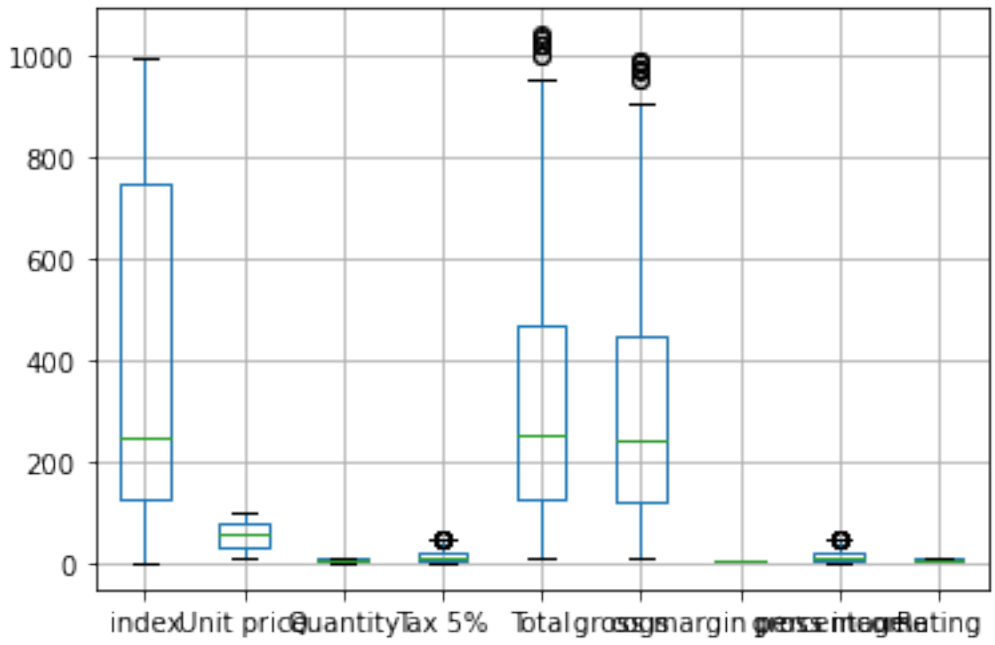
[14]:
```
index                       0
Branch                      0
City                        0
Customer type               0
Gender                      0
Product line                0
Unit price                  0
Quantity                    0
Tax 5%                      0
Total                       0
Date                        0
Time                        0
Payment                     0
cogs                        0
gross margin percentage     0
gross income                0
Rating                      0
dtype: int64
```

[15]:
```
df.duplicated().sum()
```

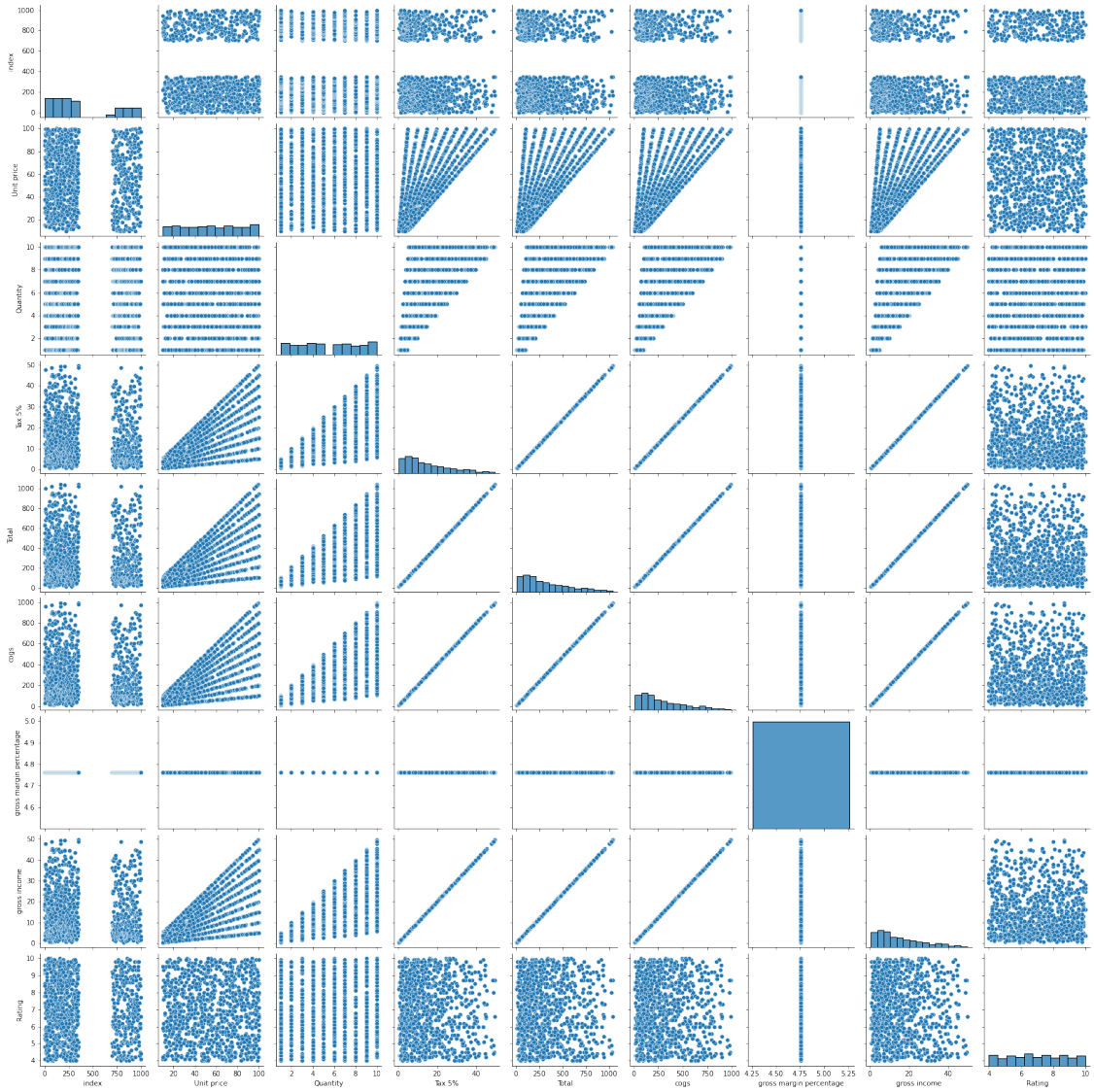
[15]:
```
0
```

[16]:
```
df.boxplot()
```

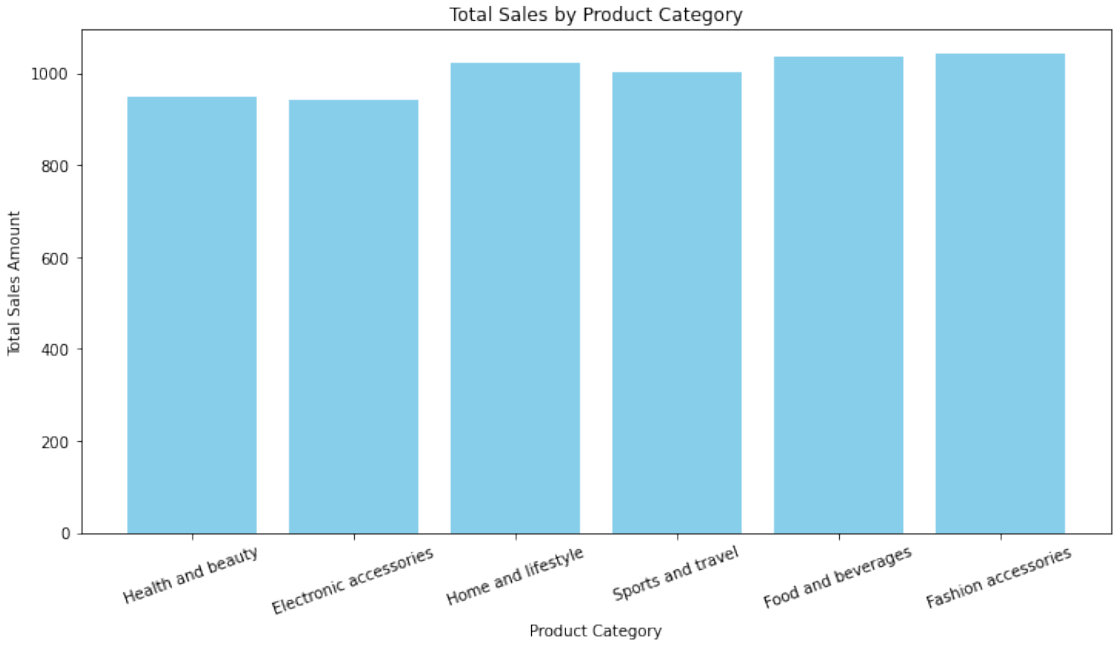
[16]:
```
<AxesSubplot:>
```



[17]:
```
sns.pairplot(df)
plt.show()
```



[18]:
```
plt.figure(figsize=(12,6))
plt.bar(df['Product line'], df['Total'], color ='skyblue')
plt.xlabel('Product Category')
plt.ylabel('Total Sales Amount')
plt.title('Total Sales by Product Category')
plt.xticks(rotation =20)
plt.show()
```


Total Sales by Product Category
Total Sales Amount
0
200
400
600
800
1000
Health and beauty
Electronic accessories
Home and lifestyle
Sports and travel
Food and beverages
Fashion accessories
Product Category

[ ]:

[ ]: