dmv1

November 3, 2024

1 Data Loading, Storage and File Formats Problem Statement: Analyzing Sales Data from Multiple. The goal is to load and analyze sales data from different file formats, including CSV, Excel, and JSON, and perform data cleaning, transformation, and analysis on the dataset.

[3]:
```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
```

[4]:
```
csv=pd.read_csv(r"C:\Users\dell\Desktop\DMV and ML\DMV Datasets\format1.csv")
excel=pd.read_excel(r"C:\Users\dell\Desktop\DMV and ML\DMV Datasets\format2.
↪xlsx")
json=pd.read_json(r"C:\Users\dell\Desktop\DMV and ML\DMV Datasets\format3.json")
```

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

[5]:

	Branch	City	Customer type	Gender	Product line	Unit price	Quantity	Tax 5%	Total	Date	Time	Payment	cogs	gross margin percentage	gross income	Rating
0	A	Yangon	Member	Female	Health and beauty	74.69	7	26.1415	548.9715	1/5/2019	13:08	Ewallet	522.83	4.761905	26.1415	9.1
1	C	Naypyitaw	Normal	Female	Electronic accessories	15.28	5	3.8200	80.2200	3/8/2019	10:29	Cash	76.40	4.761905	3.8200	9.6
2	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	16.2155	7.4
3	A	Yangon	Member	Male	Health and beauty	58.22	8	23.2880	489.0480	1/27/2019	20:33	Ewallet	465.76			
4	A	Yangon	Normal	Male	Sports and travel	86.31	7	30.2085	634.3785	2/8/2019	10:37	Ewallet	604.17			

```
3                4.761905       23.2880     8.4
4                4.761905       30.2085     5.3
```

[6]:
```
excel.head()
```

[6]:
```
  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
```

[7]:
```
json.head()
```

[7]:
```
     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
```

[8]:
```
# def merge(dataframes):
#     if dataframes:
#         return pd.concat(dataframes)
# df= merge([csv,excel,json])


df= pd.concat([csv,excel,json])
```

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

[9]:
```
  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
```

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

[10]:
```
        Unit price     Quantity       Tax 5%        Total        cogs  \
count  1000.000000  1000.000000  1000.000000  1000.000000  1000.00000
mean     55.672130     5.510000    15.379369   322.966749   307.58738
std      26.494628     2.923431    11.708825   245.885335   234.17651
min      10.080000     1.000000     0.508500    10.678500    10.17000
25%      32.875000     3.000000     5.924875   124.422375   118.49750
50%      55.230000     5.000000    12.088000   253.848000   241.76000
75%      77.935000     8.000000    22.445250   471.350250   448.90500
max      99.960000    10.000000    49.650000  1042.650000   993.00000

       gross margin percentage  gross income      Rating
count              1000.000000   1000.000000  1000.00000
```

```
mean                    4.761905     15.379369     6.97270
std                     0.000000     11.708825     1.71858
min                     4.761905      0.508500     4.00000
25%                     4.761905      5.924875     5.50000
50%                     4.761905     12.088000     7.00000
75%                     4.761905     22.445250     8.50000
max                     4.761905     49.650000    10.00000
```

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

```
<class 'pandas.core.frame.DataFrame'>
Index: 1000 entries, 0 to 999
Data columns (total 16 columns):
 #   Column                   Non-Null Count  Dtype
---  ------                   --------------  -----
 0   Branch                   1000 non-null   object
 1   City                     1000 non-null   object
 2   Customer type            1000 non-null   object
 3   Gender                   1000 non-null   object
 4   Product line             1000 non-null   object
 5   Unit price               1000 non-null   float64
 6   Quantity                 1000 non-null   int64
 7   Tax 5%                   1000 non-null   float64
 8   Total                    1000 non-null   float64
 9   Date                     1000 non-null   object
 10  Time                     1000 non-null   object
 11  Payment                  1000 non-null   object
 12  cogs                     1000 non-null   float64
 13  gross margin percentage  1000 non-null   float64
 14  gross income             1000 non-null   float64
 15  Rating                   1000 non-null   float64
dtypes: float64(7), int64(1), object(8)
memory usage: 132.8+ KB
```

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

```
C:\Users\dell\AppData\Local\Temp\ipykernel_12120\2104285443.py:2: UserWarning:
Could not infer format, so each element will be parsed individually, falling
back to `dateutil`. To ensure parsing is consistent and as-expected, please
specify a format.
  df.Time=pd.to_datetime(df.Time)
```

[13]:
```
df.head()
```

[13]:

	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  \
0         7  26.1415  548.9715 2019-01-05 2024-11-02 13:08:00      Ewallet
1         5   3.8200   80.2200 2019-03-08 2024-11-02 10:29:00         Cash
2         7  16.2155  340.5255 2019-03-03 2024-11-02 13:23:00  Credit card
3         8  23.2880  489.0480 2019-01-27 2024-11-02 20:33:00      Ewallet
4         7  30.2085  634.3785 2019-02-08 2024-11-02 10:37:00      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
```

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

[14]:
```
Branch                          object
City                            object
Customer type                   object
Gender                          object
Product line                    object
Unit price                     float64
Quantity                         int64
Tax 5%                         float64
Total                          float64
Date                    datetime64[ns]
Time                    datetime64[ns]
Payment                         object
cogs                           float64
gross margin percentage        float64
gross income                   float64
Rating                         float64
dtype: object
```

2 DATA CLEANING

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

[15]:
```
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
```

[16]:
```
df.duplicated().sum()
```

[16]:
```
0
```

3 DATA ANALYSIS

[17]:
```
total_sales = df['Total'].sum()
total_sales
```

[17]:
```
322966.749
```

[18]:
```
average_order_value = df.groupby('Gender')['Total'].mean()
average_order_value
```

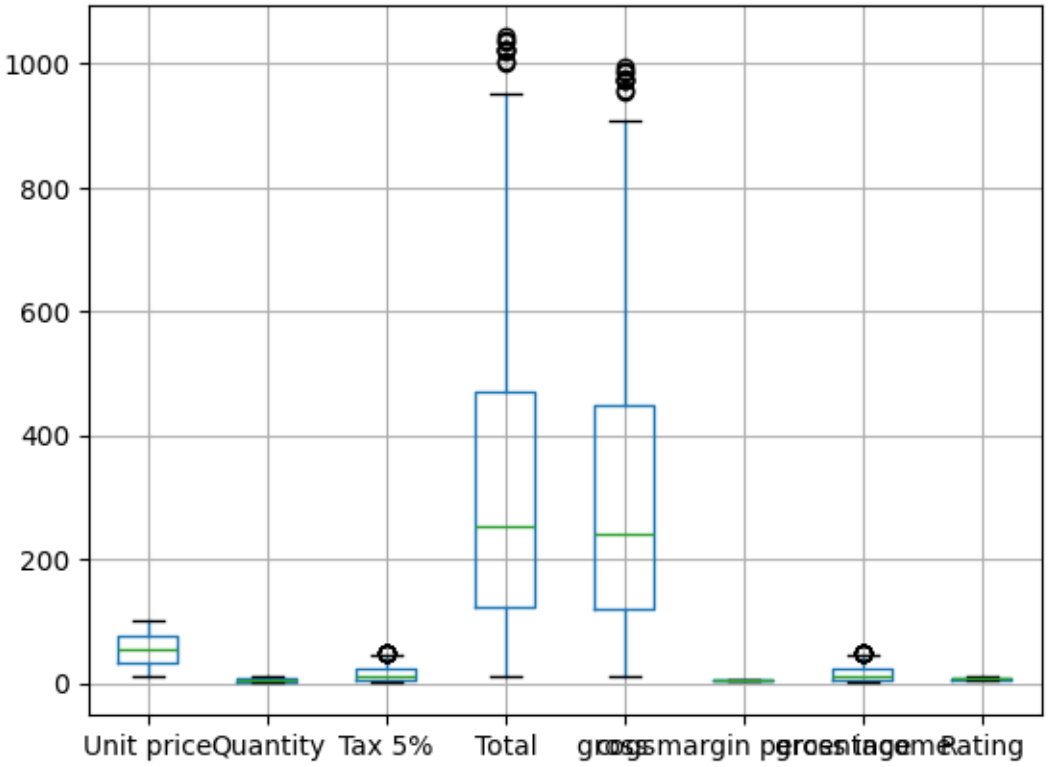
[18]:
```
Gender
Female    335.095659
Male      310.789226
Name: Total, dtype: float64
```

[19]:
```
product_category_distribution = df['Product line'].value_counts()
product_category_distribution
```

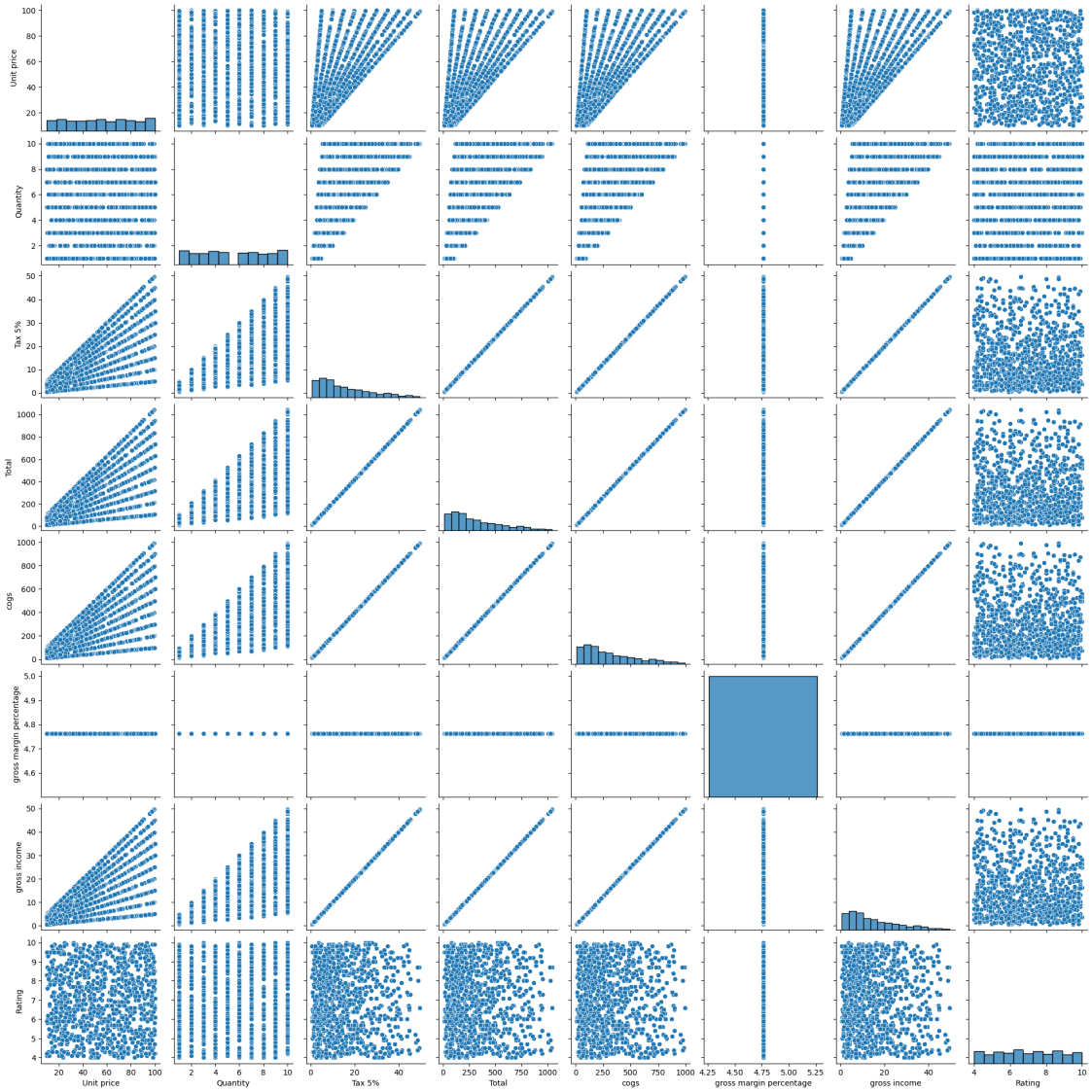
[19]:
```
Product line
Fashion accessories       178
Food and beverages        174
Electronic accessories    170
Sports and travel         166
Home and lifestyle        160
Health and beauty         152
Name: count, dtype: int64
```

4 DATA VISUALIZATION

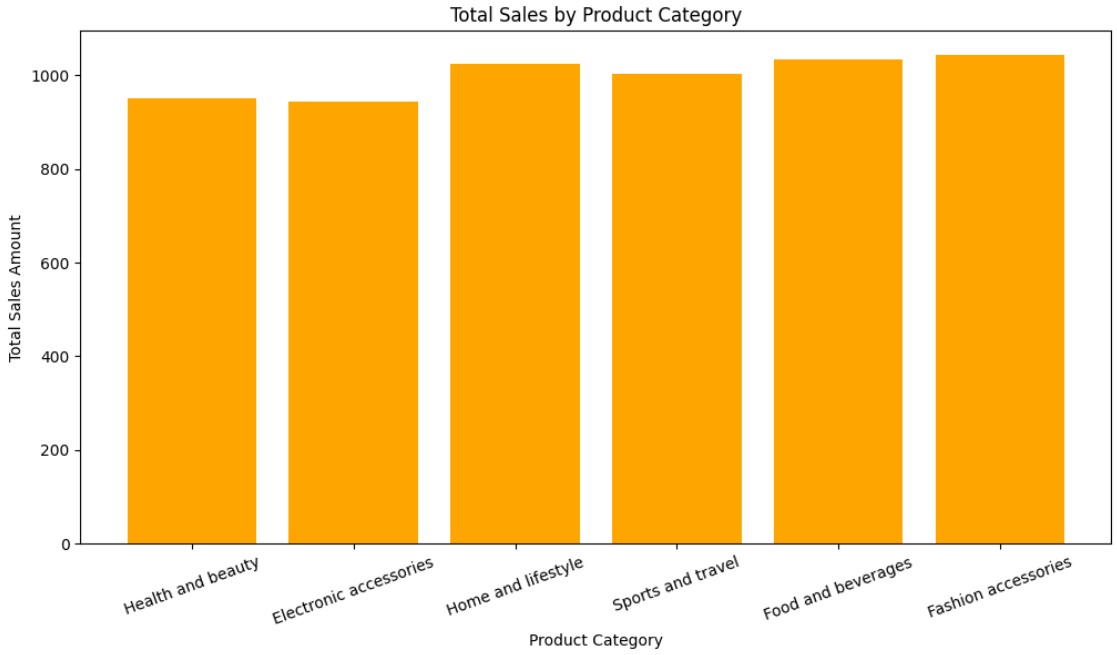
[20]:
```
df.boxplot()
plt.show()
```



[21]:
```
sns.pairplot(df)
plt.show()
```

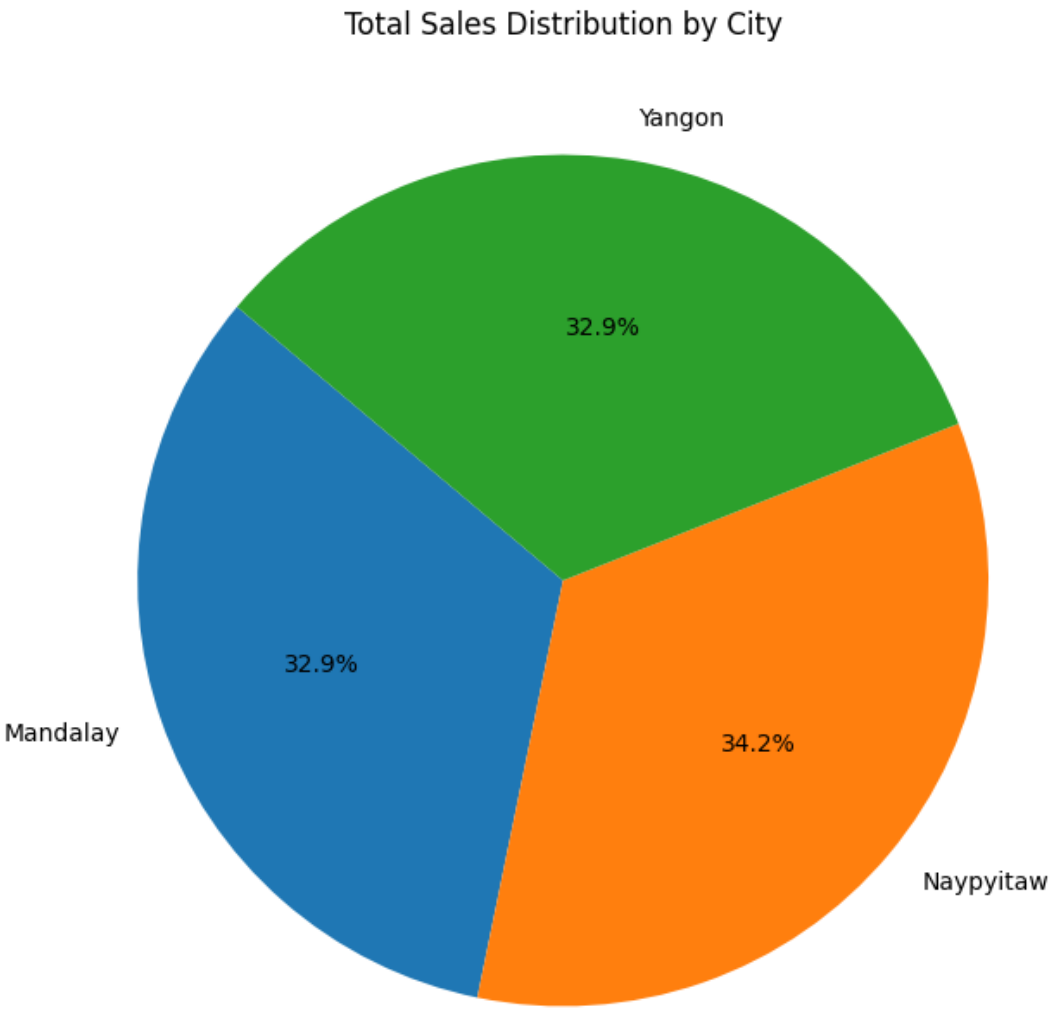


[22]:
```
plt.figure(figsize=(12, 6))
plt.bar(df['Product line'], df['Total'], color='orange')
plt.xlabel('Product Category')
plt.ylabel('Total Sales Amount')
plt.title('Total Sales by Product Category')
plt.xticks(rotation=20)
plt.show()
```



[23]:
```
sales_by_city = df.groupby('City')['Total'].sum()

plt.figure(figsize=(8, 8))
plt.pie(sales_by_city,labels=sales_by_city.index, autopct='%1.1f%%',␣
 ↪startangle=140)
plt.title('Total Sales Distribution by City')
plt.show()
```


Total Sales Distribution by City
Yangon
32.9%
32.9%
Mandalay
34.2%
Naypyitaw

[ ]: