UNIVERSITY OF WESTMINSTER

INFORMATICS INSTITUTE OF TECHNOLOGY

5DATA005W DATA ENGINEERING

COURSEWORK

Lecturer Name: MS. Yaalini Balathasan

Tutorial Group B

Name: M S Noor

UOW ID: w2052142

IIT No: 20231154

Table of Contents

Data Extraction

Sample CSV files which shows the customer,sales and products transactions of a electronic store was downloaded from. https://mavenanalytics.io/data-playground as per requirement of the coursework specifications.

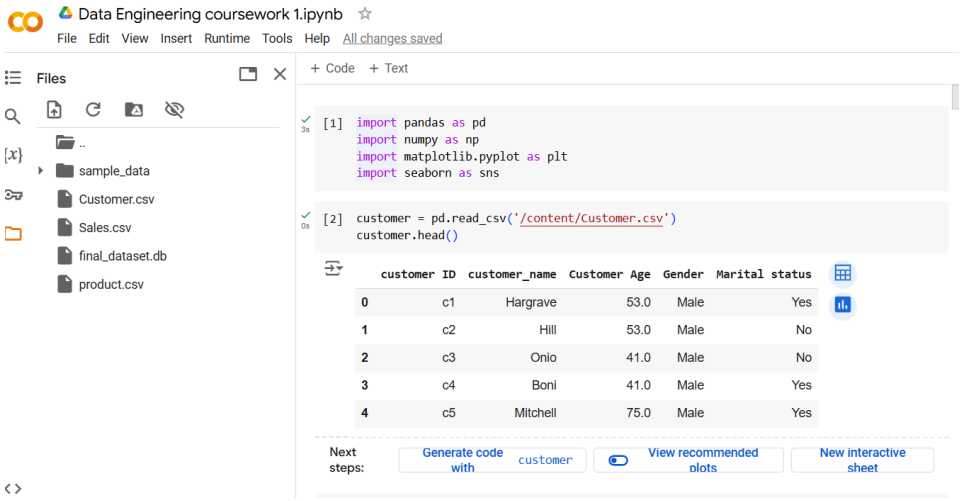
Below are the sources which I found related attributes related to dataset files:

https://maven-datasets.s3amazonaws.com/Bank+Customer+Churn/Bank+Customer+Churn.zip

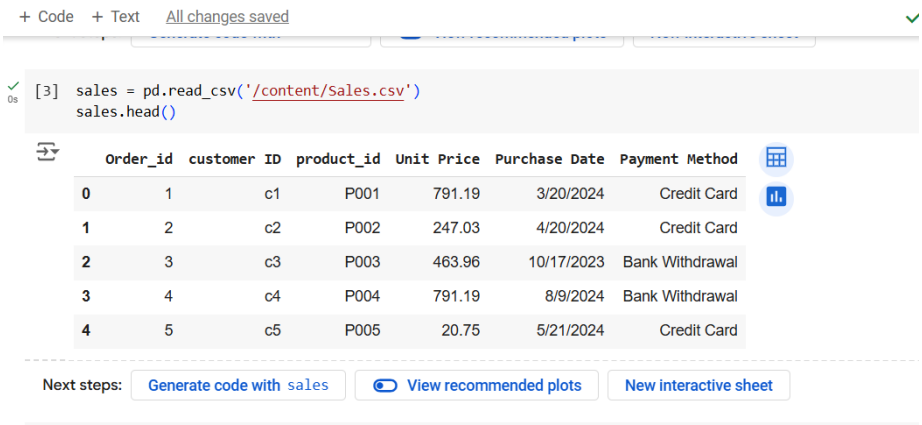
https://www.kaggle.com/datasets/abhishekrp1517/online-retail-transactions-dataset

https://maven-datasets.s3.amazonaws.com/Coffee+Shop+Sales/Coffee+Shop+Sales.zip

I found different types of attributes related to my dataset for customer,sales and product from different websites and I created my dataset.

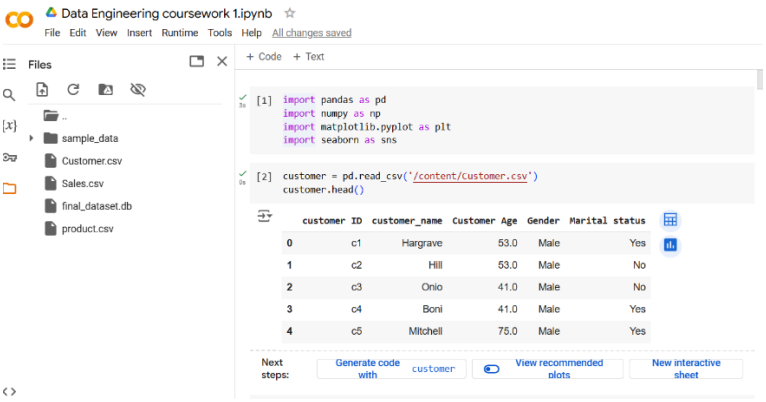


I saved my dataset in google drive and it icludes all information about customer transactions, sales transactions and product transactions. Customer_id, product_id and order_id are some unique attributes (primary keys) which were used to record data. There minimum of of five attributes to each dataset. The sales dataset has more characteristics than the other two datasets.



```
[3] sales = pd.read_csv('/content/Sales.csv')
    sales.head()
```

	Order_id	customer ID	product_id	Unit Price	Purchase Date	Payment Method
0	1	c1	P001	791.19	3/20/2024	Credit Card
1	2	c2	P002	247.03	4/20/2024	Credit Card
2	3	c3	P003	463.96	10/17/2023	Bank Withdrawal
3	4	c4	P004	791.19	8/9/2024	Bank Withdrawal
4	5	c5	P005	20.75	5/21/2024	Credit Card



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

```
[2] customer = pd.read_csv('/content/Customer.csv')
    customer.head()
```

	customer ID	customer_name	Customer Age	Gender	Marital status
0	c1	Hargrave	53.0	Male	Yes
1	c2	Hill	53.0	Male	No
2	c3	Onio	41.0	Male	No
3	c4	Boni	41.0	Male	Yes
4	c5	Mitchell	75.0	Male	Yes

From the above diagram I have imported pandas library for data manipulation .”pd.read_csv” function from pandas reads the all three csv files which is located in the directory. “head()” function shows the first 5 rows of the dataset which read in the csv file. In this case, it extract data from a csv file into pandas dataframe and display a preview data. This is the first step many data analysis works.

Task 01

1.1 Data Exploration

Sample csv files containing customer purchase data such as customer, purchase and sales data files was reviewed to have an idea of its structure and content.

1.(a) No. of data points

1a.

```
[5] #Number of Data points for Customer
    datapointsC = len(customer)
    datapointsC
```

```
599
```

```
#Number of Data points for sales
datapointsS = len(sales)
datapointsS
```

```
599
```

```
[7] #Number of data points for products
    datapointsP = len(products)
    datapointsP
```

```
599
```

These lines of codes shows the number of rows in each data file. Each csv file (customer, sales, product) contains 599 data points by using “len” function from pandas.

1.(b) Name of Attributes

1b.

```
[8] #name of attributes for customer information
    customer_attribute = customer.columns
    print(customer_attribute)
```

```
Index(['customer ID', 'customer_name', ' Customer Age', 'Gender',
       'Marital status'],
      dtype='object')
```

```
[9] #name of attributes for sales transactions
    sales_attribute = sales.columns
    print(sales_attribute)
```

```
Index(['Order_id', 'customer ID', 'product_id', 'Unit Price', 'Purchase Date',
       'Payment Method'],
      dtype='object')
```

```
#name of attributes for inventory management data
products_attribute = products.columns
print(products_attribute)
```

```
Index(['product_id', 'Product_name', ' Product Quantity', 'Product country',
       'Shipping Type'],
      dtype='object')
```

The above figure shows the names of the attributes in each dataset.By using ".columns" of pandas dataframe to access the column names (attribute) of a dataframe.

1.(c) Type of Attribute

1C.

```
[11] #customer type of attributes
     customer.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 599 entries, 0 to 598
Data columns (total 5 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   customer ID     599 non-null    object
 1   customer_name   599 non-null    object
 2    Customer Age   582 non-null    float64
 3   Gender          599 non-null    object
 4   Marital status  599 non-null    object
dtypes: float64(1), object(4)
memory usage: 23.5+ KB
```

```
[12] #sales type of attributes
     sales.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 599 entries, 0 to 598
Data columns (total 6 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   Order_id        599 non-null    int64
 1   customer ID     599 non-null    object
 2   product_id      599 non-null    object
 3   Unit Price      584 non-null    float64
 4   Purchase Date   599 non-null    object
 5   Payment Method  582 non-null    object
dtypes: float64(1), int64(1), object(4)
memory usage: 28.2+ KB
```

```
#products type of attributes
products.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 599 entries, 0 to 598
Data columns (total 5 columns):
 #   Column             Non-Null Count  Dtype
---  ------             --------------  -----
 0   product_id         599 non-null    object
 1   Product_name       599 non-null    object
 2    Product Quantity  580 non-null    float64
 3   Product country    586 non-null    object
 4   Shipping Type      599 non-null    object
dtypes: float64(1), object(4)
memory usage: 23.5+ KB
```

The purpose of the command “info()” is to display the datatype of each column and summary of the dataset.From the above three figures it indicates the datatypes of each dataset.

1.(d) No. of missing values for each attribute

1D.

```
#number of missing values for customer attributes
customer_missingvalues = customer.isnull().sum()
customer_missingvalues
```

	0
customer ID	0
customer_name	0
Customer Age	17
Gender	0
Marital status	0

dtype: int64

```
[15] #number of missing values for sales attributes
     sales_missingvalues = sales.isnull().sum()
     sales_missingvalues
```

	0
Order_id	0
customer ID	0
product_id	0
Unit Price	15
Purchase Date	0
Payment Method	17

dtype: int64

```
#number of missing values for products attributes
products_missingvalues = products.isnull().sum()
products_missingvalues
```

	0
product_id	0
Product_name	0
Product Quantity	19
Product country	13
Shipping Type	0

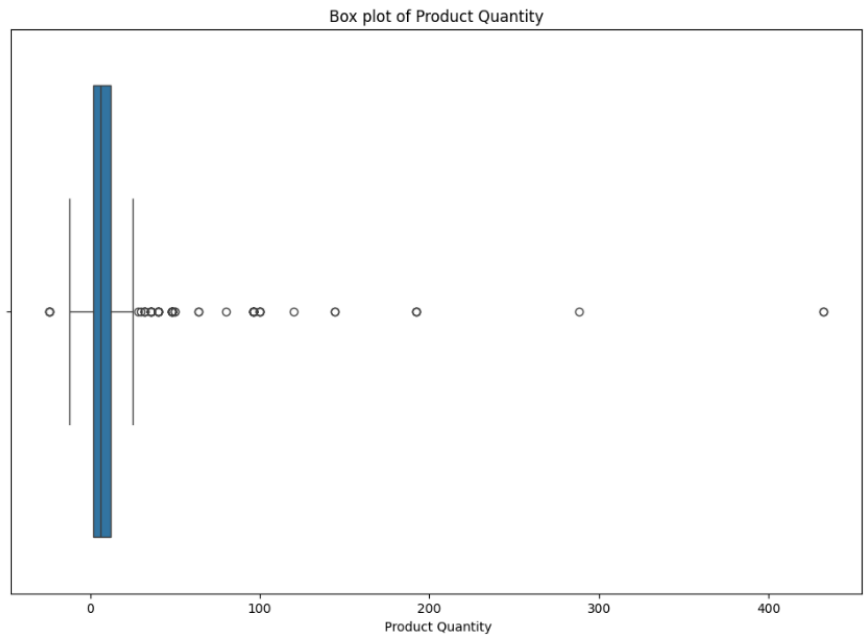
dtype: int64

From the above figures it shows the exact number of values missing values of the 3 datasets.In customer there's 17 missing values which is in "customer age" attribute. Both "sales" and "Products" csv dataset has 32 missing values in respectively.The total number of missing values are calculated by using ".sum()" in pandas. "isnull()" command used to find each individual cell of missing values in dataset.

1.(e) Entry Errors for each Attribute

```
PQ_dataset = final_dataset[' Product Quantity']

plt.figure(figsize=(12,8))
sns.boxplot(x=PQ_dataset)
plt.title('Box plot of Product Quantity')
plt.xlabel(' Product Quantity')
plt.show()
```



The above boxplot shows the outliers for " Product Quantity" attribute in product dataset which has entry errors. Entry errors may have negative consequences on the reliability and quality of the data which makes difficult to analyze. Matplotlib and Seaborn libraries were used to plot the boxplot diagram.

1E.

```
#Entry errors for numerical values in customer file
customer[' Customer Age'].describe()
```

	Customer Age
count	582.000000
mean	49.115120
std	18.407478
min	18.000000
25%	33.000000
50%	50.000000
75%	64.000000
max	80.000000

dtype: float64

```
#Entry errors for product numerical  files
products[' Product Quantity'].describe()
```

	Product Quantity
count	580.000000
mean	14.191379
std	35.868060
min	-24.000000
25%	2.000000
50%	6.000000
75%	12.000000
max	432.000000

dtype: float64

```
[17] #Entry errors for categorical values in customer file
     print(customer['Gender'].unique())
```

```
['Male' 'Female']
```

```
[18] print(customer['Marital status'].unique())
```

```
['Yes' 'No']
```

```
#Entry errors for categorical values in product file
print(products['Product_name'].unique())
```

```
['Smartphone' 'Tablet' 'Laptop' 'Smartwatch']
```

```
[20] print(products['Product country'].unique())
```

```
['France' 'Spain' 'Germany' nan]
```

```
#Entry errors for numerical values in sales file
sales['Unit Price'].describe()
```

	Unit Price
count	584.000000
mean	481.825068
std	316.269875
min	20.750000
25%	247.030000
50%	463.960000
75%	791.190000
max	844.830000

dtype: float64

```
#Entry errors for categorical values in sales file
Saleslist = sales['Purchase Date'].unique().tolist()
print(Saleslist)
```

```
['3/20/2024', '4/20/2024', '10/17/2023', '8/9/2024', '5/21/2024', '5/26/2024', '1/30/2024', '6/24/2024', '10/3/2023', '1/1/2024', '6/25/2024',
```

```
[23] print(sales['Payment Method'].unique())
```

```
['Credit Card' 'Bank Withdrawal' nan 'Mailed Check']
```

None of the other attributes in the dataset has entry errors.

1.(f) Heatmap to check missing values

1F.

```
[24] #Heatmap to check missing values for customer file
     sns.heatmap(customer.isnull(),cbar=False,cmap='viridis')
     plt.title('Customer Missing Values')
```

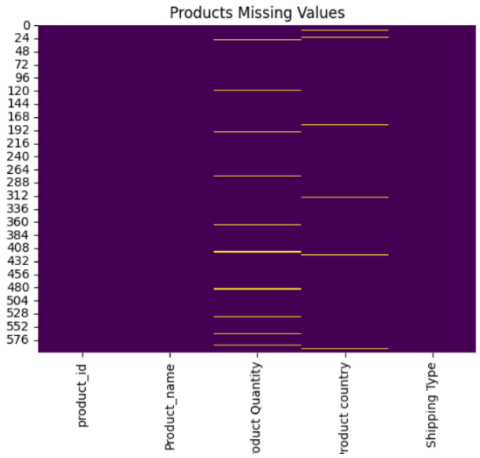
```
Text(0.5, 1.0, 'Customer Missing Values')
```


```
#Heatmap to check missing values for sales file
sns.heatmap(sales.isnull(),cbar=False,cmap='viridis')
plt.title('Sales Missing Values')
```

```
Text(0.5, 1.0, 'Sales Missing Values')
```

```
[26] #Heatmap to check missing values for products file
     sns.heatmap(products.isnull(),cbar=False,cmap='viridis')
     plt.title('Products Missing Values')
```

```
Text(0.5, 1.0, 'Products Missing Values')
```



The above heatmaps indicates the customer, product and sales dataset visual representation of missing values in their relative datasets. The yellow lines shows the missing values in columns of the datasets.Missing values can significantly impact the analysis and decision making.

Task 02.

2.1 Data Transformation

Task 02

2.1

```
#merging
S_customer = pd.merge(sales,customer,on='customer ID',how ='outer')
final_dataset = pd.merge(S_customer,products,on='product_id',how='outer')
final_dataset
```

r_id	customer ID	product_id	Unit Price	Purchase Date	Payment Method	customer_name	Customer Age	Gender	Marital status	Product_name	Product Quantity	Product country	Shipping Type
1	c1	P001	791.19	3/20/2024	Credit Card	Hargrave	53.0	Male	Yes	Smartphone	6.0	France	Standard
2	c2	P002	247.03	4/20/2024	Credit Card	Hill	53.0	Male	No	Tablet	6.0	Spain	Overnight
3	c3	P003	463.96	10/17/2023	Bank Withdrawal	Onio	41.0	Male	No	Laptop	8.0	France	Express
4	c4	P004	791.19	8/9/2024	Bank Withdrawal	Boni	41.0	Male	Yes	Smartphone	6.0	France	Overnight
5	c5	P005	20.75	5/21/2024	Credit Card	Mitchell	75.0	Male	Yes	Smartphone	6.0	Spain	Express
...	...	...	...	...	...	...	...	...	...	...	...	...	...
595	c595	P595	247.03	3/8/2024	Credit Card	Greco	22.0	Female	Yes	Tablet	1.0	Spain	Overnight
596	c596	P596	463.96	12/8/2023	Credit Card	Lombardi	75.0	Female	Yes	Laptop	1.0	Germany	Overnight

✓ 0s completed at 1:52 PM

Combine multiple csv files data with their attributes into a single dataset to make it easy to analyze and make decisions. To merge two dataframes we use the pandas function “pd.merge”. “on=” indicates the joining column for this merge. “outer” join combines to keep all rows from both datasets. This code merges and combine customer, product and sales datasets with a new dataframe “final_dataset”.

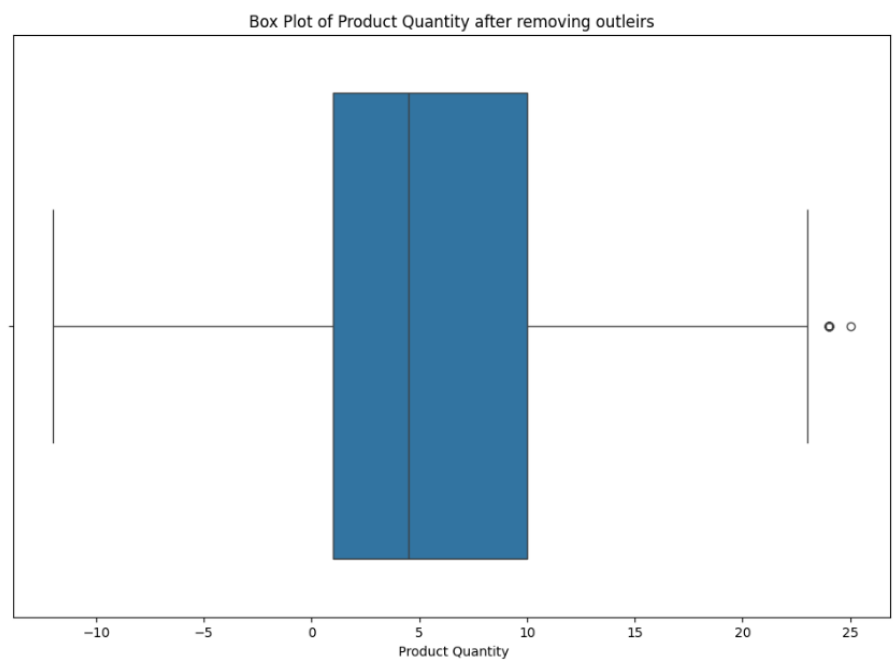
2.2 Data cleaning process

```
Q1 = final_dataset[' Product Quantity'].quantile(0.25)
Q3 = final_dataset[' Product Quantity'].quantile(0.75)
IQR = Q3 - Q1

lower_bound = Q1 - 1.5 * IQR
upper_bound = Q3 + 1.5 * IQR

# Filter data to exclude outliers
filtered_PQ_dataset = final_dataset[(final_dataset[' Product Quantity'] >= lower_bound
```

```
[78] plt.figure(figsize=(12, 8))  # Adjust figure size if needed
     sns.boxplot(x=filtered_PQ_dataset[' Product Quantity'])
     plt.title('Box Plot of Product Quantity after removing outleirs')
     plt.xlabel(' Product Quantity')
     plt.show()
```



The above diagram and codes show a data cleaning procedure for the product dataset of the "Product Quantity" attribute after removing outliers. By Removing the outliers from the boxplot diagram, it provides a more representative view of the "Product Quantity" dataset. The lines inside the box indicate the median. Data points beyond the whiskers are outliers.

```
column_rename =  {'Shipping Type': ' Delivery Method'}
final_dataset = final_dataset.rename(columns=column_rename)
final_dataset
```

r_id	customer ID	product_id	Unit Price	Purchase Date	Payment Method	customer_name	Customer Age	Gender	Marital status	Product_name	Product Quantity	Product country	Delivery Method
1	c1	P001	791.19	3/20/2024	Credit Card	Hargrave	53	Male	Yes	Smartphone	6	France	Standard
2	c2	P002	247.03	4/20/2024	Credit Card	Hill	53	Male	No	Tablet	6	Spain	Overnight
3	c3	P003	463.96	10/17/2023	Bank Withdrawal	Onio	41	Male	No	Laptop	8	France	Express
4	c4	P004	791.19	8/9/2024	Bank Withdrawal	Boni	41	Male	Yes	Smartphone	6	France	Overnight
5	c5	P005	20.75	5/21/2024	Credit Card	Mitchell	75	Male	Yes	Smartphone	6	Spain	Express
...	...	...	...	...	...	...	...	...	...	...	...	...	...
595	c595	P595	247.03	3/8/2024	Credit Card	Greco	22	Female	Yes	Tablet	1	Spain	Overnight
596	c596	P596	463.96	12/8/2023	Credit Card	Lombardi	75	Female	Yes	Laptop	1	Germany	Overnight
597	c597	P597	844.83	6/1/2024	Bank Withdrawal	Uchenna	45	Male	No	Smartwatch	3	Spain	Overnight
598	c598	P598	791.19	6/18/2024	Bank Withdrawal	Coffman	73	Male	No	Smartphone	2	Germany	Standard

```
column_rename =  {'Purchase Date': ' Aqcuisition Date'}
final_dataset = final_dataset.rename(columns=column_rename)
final_dataset
```

	Order_id	customer ID	product_id	Unit Price	Aqcuisition Date	Payment Method	customer_name	Customer Age	Gender	Marital status	Product_name	Product Quantity	Product country	De
0	1	c1	P001	791.19	3/20/2024	Credit Card	Hargrave	53	Male	Yes	Smartphone	6	France	S
1	2	c2	P002	247.03	4/20/2024	Credit Card	Hill	53	Male	No	Tablet	6	Spain	O
2	3	c3	P003	463.96	10/17/2023	Bank Withdrawal	Onio	41	Male	No	Laptop	8	France	E
3	4	c4	P004	791.19	8/9/2024	Bank Withdrawal	Boni	41	Male	Yes	Smartphone	6	France	O
4	5	c5	P005	20.75	5/21/2024	Credit Card	Mitchell	75	Male	Yes	Smartphone	6	Spain	E
...	...	...	...	...	...	...	...	...	...	...	...	...	...	
594	595	c595	P595	247.03	3/8/2024	Credit Card	Greco	22	Female	Yes	Tablet	1	Spain	O
595	596	c596	P596	463.96	12/8/2023	Credit Card	Lombardi	75	Female	Yes	Laptop	1	Germany	O
596	597	c597	P597	844.83	6/1/2024	Bank Withdrawal	Uchenna	45	Male	No	Smartwatch	3	Spain	O
597	598	c598	P598	791.19	6/18/2024	Bank Withdrawal	Coffman	73	Male	No	Smartphone	2	Germany	S

0s completed at 1:52 PM

The above two diagrams shows another python script for data cleaning to rename columns in the “final_dataset” dataset. The “column_rename” line creates a dictionary while “columns=column_rename” is an argument which shows to find the new names through dictionary. This code renames “shipping type” and “Purchase Date” in the merged dataset

```
[32] customer[' Customer Age'] = customer[' Customer Age'].fillna(0).astype(int)
```

```
[33] final_dataset[' Customer Age'] = final_dataset[' Customer Age'].fillna(0).astype(int)
```

```
final_dataset[' Product Quantity'] = final_dataset[' Product Quantity'].fillna(0).astype(int)
```

```
final_dataset
```

r_id	customer ID	product_id	Unit Price	Purchase Date	Payment Method	customer_name	Customer Age	Gender	Marital status	Product_name	Product Quantity	Product country	Shipping Type
1	c1	P001	791.19	3/20/2024	Credit Card	Hargrave	53	Male	Yes	Smartphone	6	France	Standard
2	c2	P002	247.03	4/20/2024	Credit Card	Hill	53	Male	No	Tablet	6	Spain	Overnight
3	c3	P003	463.96	10/17/2023	Bank Withdrawal	Onio	41	Male	No	Laptop	8	France	Express
4	c4	P004	791.19	8/9/2024	Bank Withdrawal	Boni	41	Male	Yes	Smartphone	6	France	Overnight
5	c5	P005	20.75	5/21/2024	Credit Card	Mitchell	75	Male	Yes	Smartphone	6	Spain	Express
...	...	...	...	...	...	...	...	...	...	...	...	...	...
595	c595	P595	247.03	3/8/2024	Credit Card	Greco	22	Female	Yes	Tablet	1	Spain	Overnight
596	c596	P596	463.96	12/8/2023	Credit Card	Lombardi	75	Female	Yes	Laptop	1	Germany	Overnight
597	c597	P597	844.83	6/1/2024	Bank Withdrawal	Uchenna	45	Male	No	Smartwatch	3	Spain	Overnight
598	c598	P598	791.19	6/18/2024	Bank Withdrawal	Coffman	73	Male	No	Smartphone	2	Germany	Standard
599	c599	P599	20.75	7/9/2024	Credit Card	Alexandrova	73	Male	No	Smartphone	1	Germany	Overnight

✓ 0s completed at 1:52 PM

This 3rd data cleaning procedure indicates how to change datatype to an integer.In the above diagrams we can see the both " customer age" and "Product Quantity" attributes datatype has been changed into integers. ".astype(int)" is the argument which changes the datatype to an integer.

2.3 Feature Engineering

Adapting new features for customer age

In this feature a new column is getting added to the merged dataset.The raw age data changed into meaningful datatypes as young, adult, late adulthood according to the customers age under a new coulumn named "age range". If the age is less than 30 the function string returns young age group, if the age is above 30 and less than 60 its return to adult category and if the

age is greater than 60 the function returns as late adulthood.This categorize the customer age data into groups.

2.3

```
#feature engineering scripts
#Adapting new features for customer age
def age_range(age):
    if age < 30:
        return 'Young'
    elif age < 60:
        return 'Adult'
    else:
        return 'Late Adulthood'

final_dataset['Age Range'] = final_dataset[' Customer Age'].apply(age_range)
final_dataset
```

er ID	product_id	Unit Price	Aqcuisition Date	Payment Method	customer_name	Customer Age	Gender	Marital status	Product_name	Product Quantity	Product country	Delivery Method	Age Range
1	P001	791.19	3/20/2024	Credit Card	Hargrave	53	Male	Yes	Smartphone	6	France	Standard	Adult
2	P002	247.03	4/20/2024	Credit Card	Hill	53	Male	No	Tablet	6	Spain	Overnight	Adult
3	P003	463.96	10/17/2023	Bank Withdrawal	Onio	41	Male	No	Laptop	8	France	Express	Adult
4	P004	791.19	8/9/2024	Bank Withdrawal	Boni	41	Male	Yes	Smartphone	6	France	Overnight	Adult
5	P005	20.75	5/21/2024	Credit Card	Mitchell	75	Male	Yes	Smartphone	6	Spain	Express	Late Adulthood
...	...	...	...	...	...	...	...	...	...	...	...	...	...

✓ 0s completed at 1:52 PM

Calculating total purchasing value

```
#calculating total purchasing value
final_dataset['Total Aqcuisition Value'] = final_dataset['Unit Price'] * final_dataset[' Product Quantity']
final_dataset
```

d	Unit Price	Aqcuisition Date	Payment Method	customer_name	Customer Age	Gender	Marital status	Product_name	Product Quantity	Product country	Delivery Method	Age Range	Total Aqcuisition Value
1	791.19	3/20/2024	Credit Card	Hargrave	53	Male	Yes	Smartphone	6	France	Standard	Adult	4747.14
2	247.03	4/20/2024	Credit Card	Hill	53	Male	No	Tablet	6	Spain	Overnight	Adult	1482.18
3	463.96	10/17/2023	Bank Withdrawal	Onio	41	Male	No	Laptop	8	France	Express	Adult	3711.68
4	791.19	8/9/2024	Bank Withdrawal	Boni	41	Male	Yes	Smartphone	6	France	Overnight	Adult	4747.14
5	20.75	5/21/2024	Credit Card	Mitchell	75	Male	Yes	Smartphone	6	Spain	Express	Late Adulthood	124.50
.	...	...	...	...	...	...	...	...	...	...	...	...	...
5	247.03	3/8/2024	Credit Card	Greco	22	Female	Yes	Tablet	1	Spain	Overnight	Young	247.03
6	463.96	12/8/2023	Credit Card	Lombardi	75	Female	Yes	Laptop	1	Germany	Overnight	Late Adulthood	463.96
7	844.83	6/1/2024	Bank Withdrawal	Uchenna	45	Male	No	Smartwatch	3	Spain	Overnight	Adult	2534.49
8	791.19	6/18/2024	Bank Withdrawal	Coffman	73	Male	No	Smartphone	2	Germany	Standard	Late Adulthood	1582.38

✓ 0s completed at 1:52 PM

The above diagram shows the line of codes which creates a new coulumn in the merged dataset named" Total acquisition Value" .In this case it multiplies the Unit price of each product by Product Quantity to get the total purchase value.

```
[57] total_customer_sales = final_dataset.groupby('customer ID')['Total Aqcuisition Value'].sum()
     total_customer_sales
```

customer ID	Total Aqcuisition Value
c1	4747.14
c10	14846.72
c100	0.00
c101	0.00
c102	0.00
...	...
c95	4747.14
c96	124.50
c97	2490.00
c98	11135.04
c99	5928.72

599 rows × 1 columns

dtype: float64

This diagram show the same thing as the total acquisition values. In this code it shows the total acquisition value related to each customer.

Handling the Missing Values

```
mean_unit_price = final_dataset['Unit Price'].mean()
final_dataset['Unit Price'].fillna(mean_unit_price, inplace=True)
final_dataset['Unit Price']
```

```
<ipython-input-64-88e1590de146>:2: FutureWarning: A value is trying to be set on a copy of a DataFrame or Series through chained assignment usin
The behavior will change in pandas 3.0. This inplace method will never work because the intermediate object on which we are setting values alway

For example, when doing 'df[col].method(value, inplace=True)', try using 'df.method({col: value}, inplace=True)' or df[col] = df[col].method(val
```

```
final_dataset['Unit Price'].fillna(mean_unit_price, inplace=True)
```

	Unit Price
0	791.19
1	247.03
2	463.96
3	791.19
4	20.75
...	...
594	247.03
595	463.96
596	844.83
597	791.19
598	20.75

```
mean_product_Q = final_dataset[' Product Quantity'].mean()
final_dataset[' Product Quantity'].fillna(mean_product_Q, inplace=True)
final_dataset[' Product Quantity']
```

```
<ipython-input-65-a402f3bd86f4>:2: FutureWarning: A value is trying to be set on a copy of a DataFrame or Series through chained assignment usin
The behavior will change in pandas 3.0. This inplace method will never work because the intermediate object on which we are setting values alway

For example, when doing 'df[col].method(value, inplace=True)', try using 'df.method({col: value}, inplace=True)' or df[col] = df[col].method(val
```

```
final_dataset[' Product Quantity'].fillna(mean_product_Q, inplace=True)
```

	Product Quantity
0	6
1	6
2	8
3	6
4	6
...	...
594	1
595	1
596	3
597	2

```python
mean_payment_method = final_dataset['Payment Method'].mode()[0]
final_dataset['Payment Method'].fillna
final_dataset['Payment Method'].fillna(mean_payment_method, inplace=True)
final_dataset['Payment Method']
```

```
<ipython-input-66-22e8001e3521>:3: FutureWarning: A value is trying to be set on a copy of a DataFrame or Series through chained assignment using
The behavior will change in pandas 3.0. This inplace method will never work because the intermediate object on which we are setting values always

For example, when doing 'df[col].method(value, inplace=True)', try using 'df.method({col: value}, inplace=True)' or df[col] = df[col].method(valu
```

```
final_dataset['Payment Method'].fillna(mean_payment_method, inplace=True)
```

	Payment Method
0	Credit Card
1	Credit Card
2	Bank Withdrawal
3	Bank Withdrawal
4	Credit Card
...	...
594	Credit Card
595	Credit Card
596	Bank Withdrawal

```python
mean_product_country = final_dataset['Product country'].mode()[0]
final_dataset['Product country'].fillna
final_dataset['Product country'].fillna(mean_product_country , inplace=True)
final_dataset['Product country']
```

```
<ipython-input-67-715f713d8170>:3: FutureWarning: A value is trying to be set on a copy of a DataFrame or Series through chained assignment using
The behavior will change in pandas 3.0. This inplace method will never work because the intermediate object on which we are setting values always

For example, when doing 'df[col].method(value, inplace=True)', try using 'df.method({col: value}, inplace=True)' or df[col] = df[col].method(valu
```

```
final_dataset['Product country'].fillna(mean_product_country , inplace=True)
```

	Product country
0	France
1	Spain
2	France
3	France
4	Spain
...	...
594	Spain
595	Germany
596	Spain

```python
mean_total_aqcuisition_value = final_dataset['Total Aqcuisition Value'].mean()
final_dataset['Total Aqcuisition Value'].fillna(mean_total_aqcuisition_value , inplace=True)
final_dataset['Total Aqcuisition Value']
```

	Total Aqcuisition Value
0	4747.14
1	1482.18
2	3711.68
3	4747.14
4	124.50
...	...
594	247.03
595	463.96
596	2534.49
597	1582.38
598	20.75

599 rows × 1 columns

dtype: float64

#After handling all the missing values

```
[69] products_new = final_dataset.isnull().sum()
     products_new
```

	0
Order_id	0
customer ID	0
product_id	0
Unit Price	0
Aqcuisition Date	0
Payment Method	0
customer_name	0
Customer Age	0
Gender	0
Marital status	0
Product_name	0
Product Quantity	0
Product country	0
Delivery Method	0
Age Range	0

In this case, all the above diagram the merged final dataset missing values were handled. Numeric values used “mean” to fill out missing numeric values and “mode[0]” to fill categorical missing values. “fillna()” to fill the missing values.

2.4 Metafile

```
#statisctics for numerical variables
N_satistics = final_dataset[['Unit Price',' Customer Age',' Product Quantity','Total Aqcuisition Value']].describe()
N_satistics
```

	Unit Price	Customer Age	Product Quantity	Total Aqcuisition Value
count	599.000000	599.000000	599.000000	599.000000
mean	481.825068	47.721202	13.741235	6539.568225
std	312.278089	19.895565	35.381317	22429.999496
min	20.750000	0.000000	-24.000000	-18988.560000
25%	247.030000	31.000000	2.000000	463.960000
50%	463.960000	50.000000	6.000000	1582.380000
75%	791.190000	64.000000	12.000000	5068.980000
max	844.830000	80.000000	432.000000	364966.560000

Next steps: Generate code with N_satistics | View recommended plots | New interactive sheet

```
[65] #statistic for categorical variables
     C_satistics = final_dataset.describe(include='object')
     C_satistics
```

	customer ID	product_id	Aqcuisition Date	Payment Method	customer_name	Gender	Marital status	Product_name	Product country	Delivery Method	Age Range
count	599	599	599	599	599	599	599	599	599	599	599
unique	599	599	304	3	485	2	2	4	3	3	3
top	c1	P001	6/13/2024	Bank Withdrawal	Shih	Male	No	Smartphone	France	Overnight	Adult
freq	1	1	6	376	5	315	304	250	282	213	261

Next steps: Generate code with C_satistics | View recommended plots | New interactive sheet

```
[66] final_dataset_dp = len(final_dataset)
     final_dataset_attributes = list(final_dataset)
     final_dataset_data_type = final_dataset.dtypes
```

```
metadataf = pd.DataFrame({
    'Property': ['Number of DP - final_dataset','types of attributes - final_dataset', 'data type - final_dataset','Numerical variables - final_
    'Value': [final_dataset_dp, final_dataset_attributes, final_dataset_data_type, N_satistics, C_satistics]
})
```

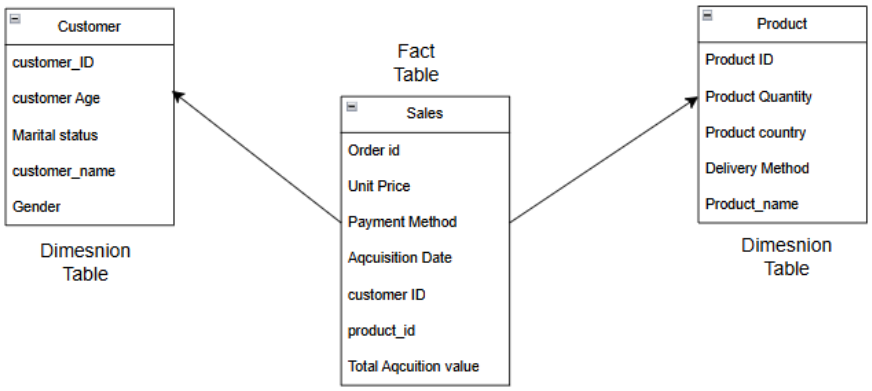
```
[68] metadataf.to_csv('metadataf.csv', index=False)
```

The above screenshots indicates the statistics of numerical and categorical value data in the merged final dataset.

Task 03

Dataloading

Simple Dimensional Diagram



Task 03

```
[69] #Data loading SQL
     import sqlite3 as sql
```

```
[70] conn = sql.connect('final_dataset.db')
     cur = conn.cursor()
```

```
[71] final_dataset.to_sql('final_dataset',conn,if_exists='replace',index=False)
```

```
599
```

```
[72] conn.commit()
     conn.close()
```

On task 3 I have imported the sqlite3 with sqlite database. This setup a "conn" connection to and SQLite database file named final_dataset.db. In this case, I have created a new database to store the data. "conn.close()" closes the database connection.

This task involved transferring the cleaned final dataset into a relational database to analysis. Setting up a schema setup is crucial in the data loading process when working with my sql database. Primary keys, foreign keys, constraints are some of the main components of schema setup in SQL database.A well-designed schema setup ensures correctly loaded data.

Task 05.

Self Reflection

The most challenging task was to find a suitable CSV file dataset that can use according to the coursework specifications. It also involved overcoming challenges like data interpret, cleaning, transforming and coding.

By working through these I learned so many things and had valuable improvement of practical skills. This contribution gave a deeper understanding of data management and analysis for real world data industry.

Data integration and combining three datasets into one merged file was challenging as always. When cleaning the data there was so many data quality issues like missing values and oultiers in the product quantity column. Also, this includes removing duplicates in the dataset as well.

The coursework provided me with invaluable learning experiences in various data areas.

Storing data into a SQLite database initiate the concept of data management which will set the footing for more data pipelines.

This coursework was a deep dive into the practical aspect of data engineering.

Overall, this coursework has been an outstanding experience, which improved my skills in coding and changed my view and importance of handling data.