SALES DATA ANALYSIS USING PYTHON

2025

OUTLINE

Introduction

This project is part of the Axia Africa Python training program and is designed to showcase practical data analysis skills using Python libraries such as **Pandas** and **Matplotlib**. The project simulates a real-world data analyst role, where raw sales data must be cleaned, structured, and analyzed to generate meaningful insights that support business decision-making.

Through this exercise, I will gain hands-on experience in data wrangling, performing exploratory data analysis (EDA), applying descriptive statistics, and visualizing results. The goal is to transform messy data into clear, actionable findings that would be valuable to business stakeholders.

Purpose of the Project

The purpose of this project is to:

i. Demonstrate the ability to import, clean, and preprocess raw sales data using **Pandas**.
ii. Explore the dataset to understand business patterns and trends through descriptive statistics and aggregation.
iii. Use **Matplotlib** for effective data visualization that highlights sales performance across different products, cities, managers, and payment methods.
iv. Provide answers to specific business-related questions

Dataset Description

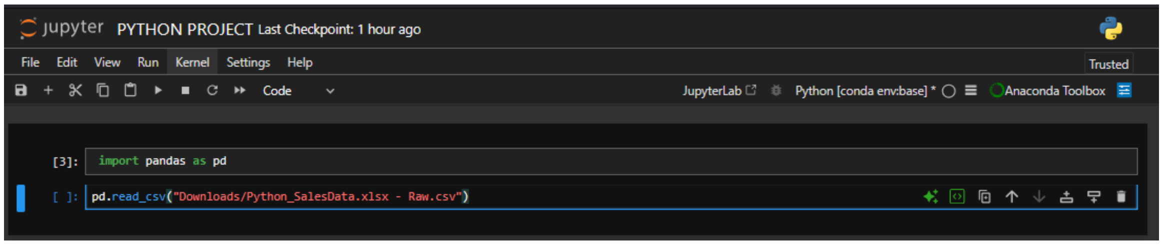
The dataset consists of **255 rows and 10 columns**, representing raw sales transactions from a retail food business. After cleaning, the dataset will include the following key fields:

- **Order ID**: Unique identifier for each transaction.
- **Date**: The date when the sales transaction occurred.
- **Product**: Category of product sold (e.g., Burgers, Beverages, Fries).
- **Price**: Unit price of the product sold.
- **Quantity**: Quantity sold in that order.
- **Purchase Type**: Mode of purchase (e.g., Online, In-store).

- **Payment Method**: Method of payment used (e.g., Credit Card, Gift Card).
- **Manager**: The sales manager associated with the order.
- **City**: The city where the transaction occurred.

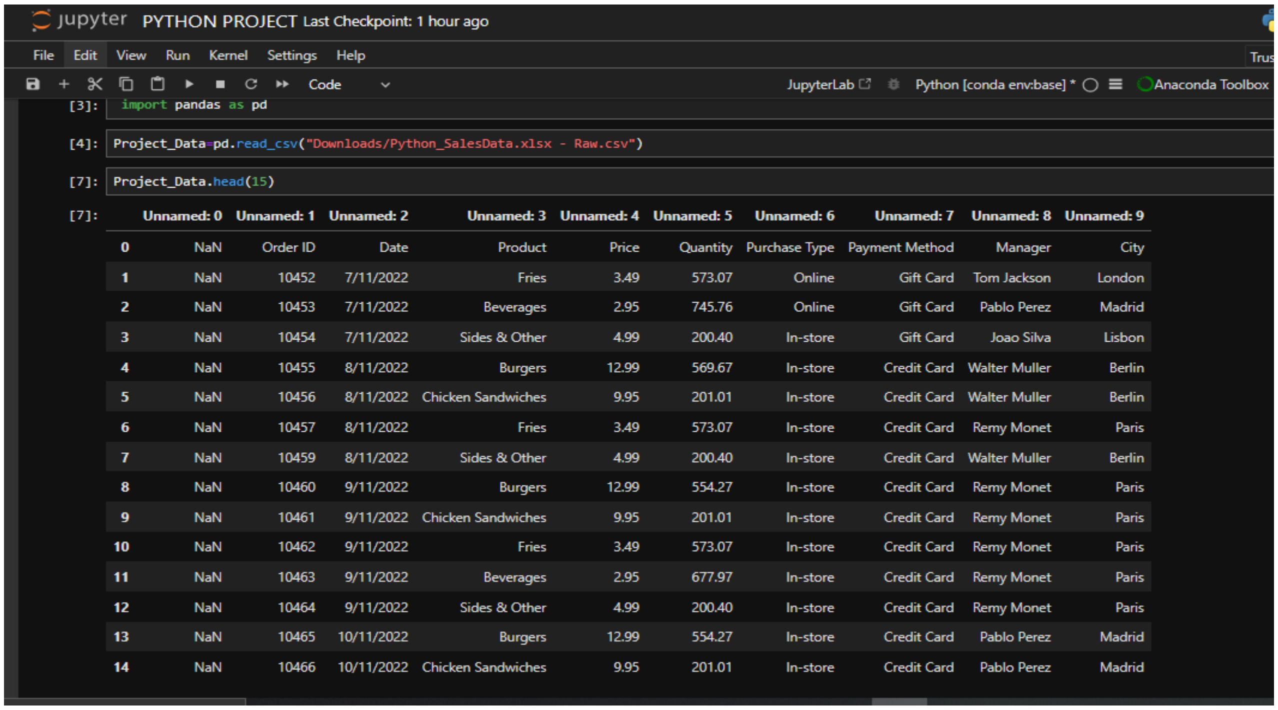
The dataset contains some issues such as missing headers, inconsistent formatting, and potential duplicate values. These will be addressed during the **data cleaning phase** before any analysis is performed.

2.0 Step Wise Objectives


jupyter PYTHON PROJECT Last Checkpoint: 1 hour ago
File Edit View Run Kernel Settings Help
Trusted
Code
JupyterLab
Python [conda env:base] *
Anaconda Toolbox

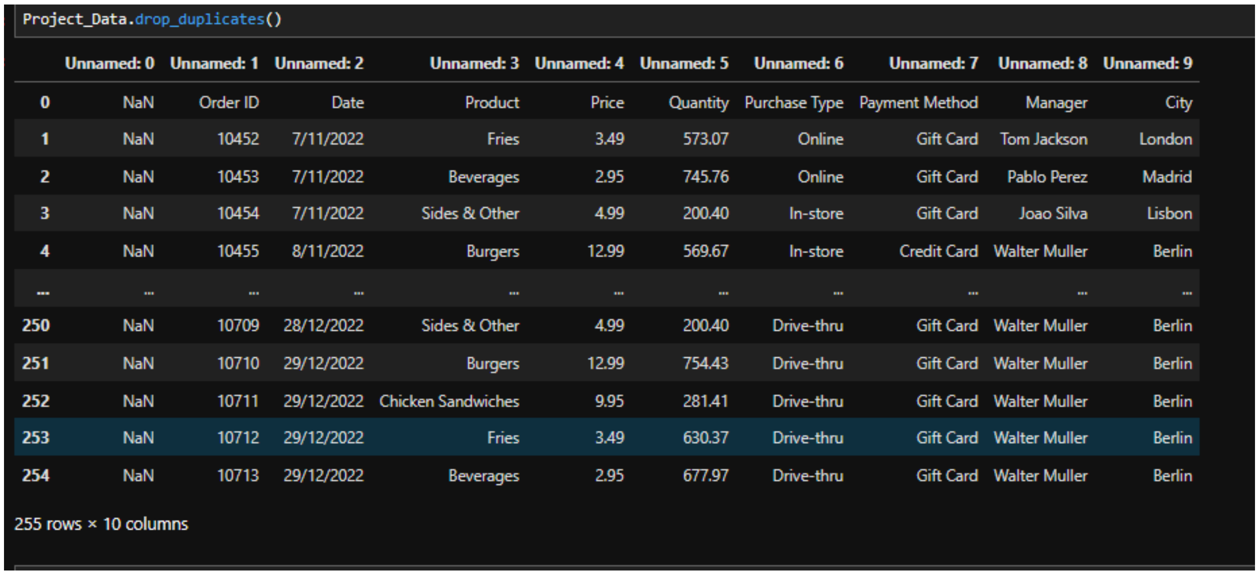
```
[3]: import pandas as pd
[ ]: pd.read_csv("Downloads/Python_SalesData.xlsx - Raw.csv")
```

Setting Up: Importing Pandas


jupyter PYTHON PROJECT Last Checkpoint: 1 hour ago
File Edit View Run Kernel Settings Help
Code
JupyterLab
Python [conda env:base] *
Anaconda Toolbox

```
[3]: import pandas as pd

[4]: Project_Data=pd.read_csv("Downloads/Python_SalesData.xlsx - Raw.csv")

[7]: Project_Data.head(15)

[7]:     Unnamed: 0  Unnamed: 1  Unnamed: 2          Unnamed: 3  Unnamed: 4  Unnamed: 5     Unnamed: 6      Unnamed: 7     Unnamed: 8  Unnamed: 9
      0         NaN    Order ID        Date             Product       Price    Quantity  Purchase Type  Payment Method        Manager        City
      1         NaN       10452   7/11/2022               Fries        3.49      573.07         Online       Gift Card    Tom Jackson      London
      2         NaN       10453   7/11/2022           Beverages        2.95      745.76         Online       Gift Card    Pablo Perez      Madrid
      3         NaN       10454   7/11/2022       Sides & Other        4.99      200.40       In-store       Gift Card     Joao Silva      Lisbon
      4         NaN       10455   8/11/2022             Burgers       12.99      569.67       In-store     Credit Card   Walter Muller      Berlin
      5         NaN       10456   8/11/2022  Chicken Sandwiches        9.95      201.01       In-store     Credit Card   Walter Muller      Berlin
      6         NaN       10457   8/11/2022               Fries        3.49      573.07       In-store     Credit Card    Remy Monet        Paris
      7         NaN       10459   8/11/2022       Sides & Other        4.99      200.40       In-store     Credit Card   Walter Muller      Berlin
      8         NaN       10460   9/11/2022             Burgers       12.99      554.27       In-store     Credit Card    Remy Monet        Paris
      9         NaN       10461   9/11/2022  Chicken Sandwiches        9.95      201.01       In-store     Credit Card    Remy Monet        Paris
     10         NaN       10462   9/11/2022               Fries        3.49      573.07       In-store     Credit Card    Remy Monet        Paris
     11         NaN       10463   9/11/2022           Beverages        2.95      677.97       In-store     Credit Card    Remy Monet        Paris
     12         NaN       10464   9/11/2022       Sides & Other        4.99      200.40       In-store     Credit Card    Remy Monet        Paris
     13         NaN       10465  10/11/2022             Burgers       12.99      554.27       In-store     Credit Card    Pablo Perez      Madrid
     14         NaN       10466  10/11/2022  Chicken Sandwiches        9.95      201.01       In-store     Credit Card    Pablo Perez      Madrid
```

Setting Up: Previewing the Data

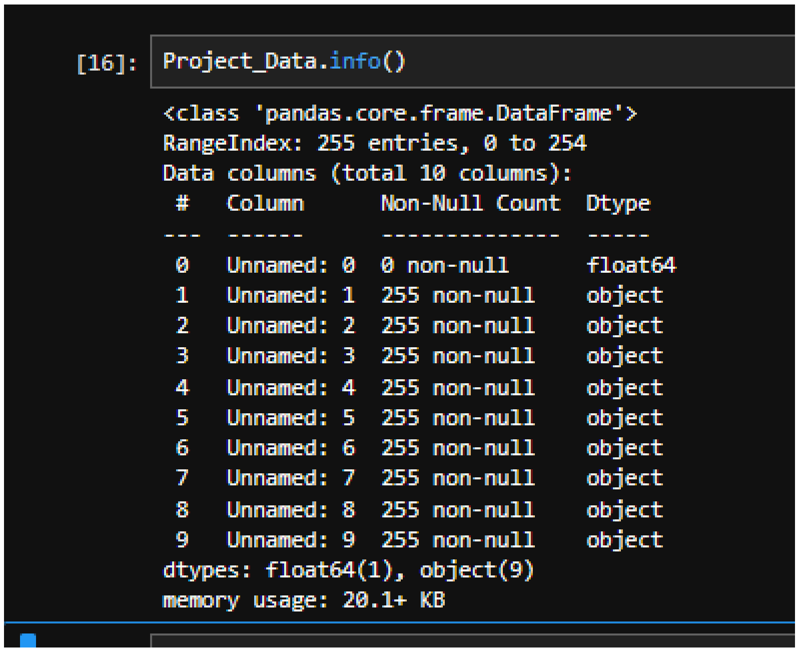


```
Project_Data.drop_duplicates()
```

	Unnamed: 0	Unnamed: 1	Unnamed: 2	Unnamed: 3	Unnamed: 4	Unnamed: 5	Unnamed: 6	Unnamed: 7	Unnamed: 8	Unnamed: 9
0	NaN	Order ID	Date	Product	Price	Quantity	Purchase Type	Payment Method	Manager	City
1	NaN	10452	7/11/2022	Fries	3.49	573.07	Online	Gift Card	Tom Jackson	London
2	NaN	10453	7/11/2022	Beverages	2.95	745.76	Online	Gift Card	Pablo Perez	Madrid
3	NaN	10454	7/11/2022	Sides & Other	4.99	200.40	In-store	Gift Card	Joao Silva	Lisbon
4	NaN	10455	8/11/2022	Burgers	12.99	569.67	In-store	Credit Card	Walter Muller	Berlin
...	...	...	...	...	...	...	...	...	...	...
250	NaN	10709	28/12/2022	Sides & Other	4.99	200.40	Drive-thru	Gift Card	Walter Muller	Berlin
251	NaN	10710	29/12/2022	Burgers	12.99	754.43	Drive-thru	Gift Card	Walter Muller	Berlin
252	NaN	10711	29/12/2022	Chicken Sandwiches	9.95	281.41	Drive-thru	Gift Card	Walter Muller	Berlin
253	NaN	10712	29/12/2022	Fries	3.49	630.37	Drive-thru	Gift Card	Walter Muller	Berlin
254	NaN	10713	29/12/2022	Beverages	2.95	677.97	Drive-thru	Gift Card	Walter Muller	Berlin

255 rows × 10 columns

Removing duplicates

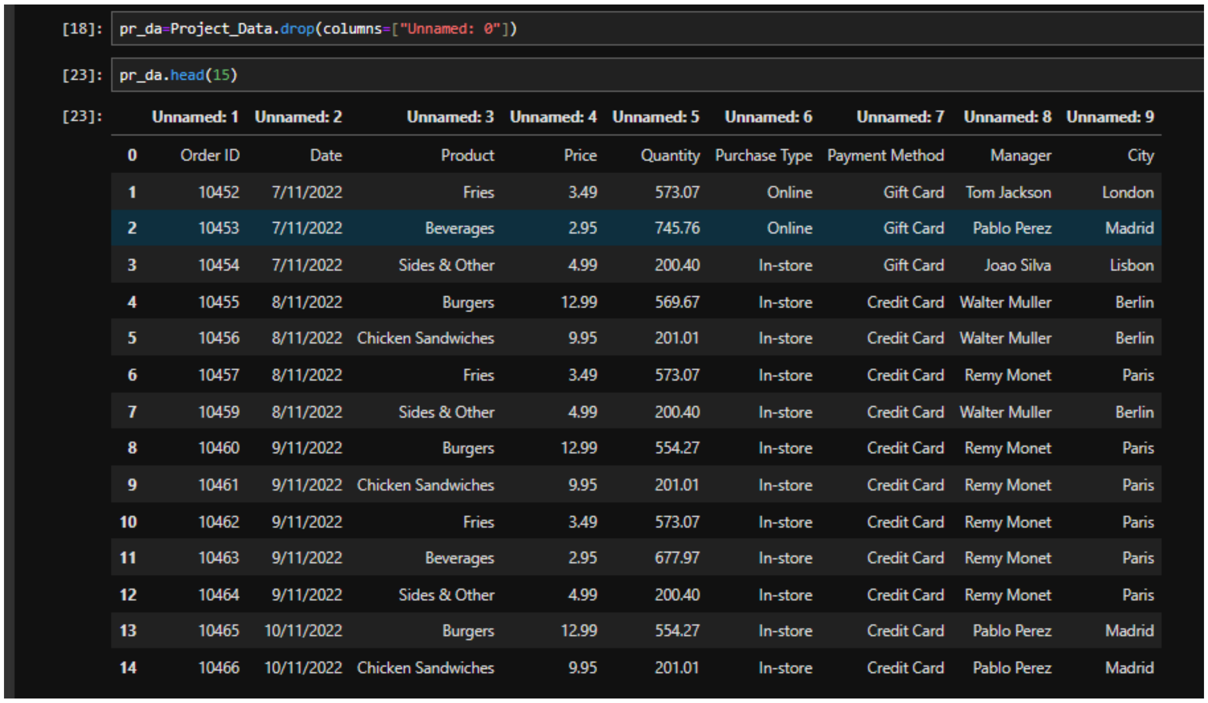


```
[16]: Project_Data.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 255 entries, 0 to 254
Data columns (total 10 columns):
 #   Column      Non-Null Count  Dtype
---  ------      --------------  -----
 0   Unnamed: 0  0 non-null      float64
 1   Unnamed: 1  255 non-null    object
 2   Unnamed: 2  255 non-null    object
 3   Unnamed: 3  255 non-null    object
 4   Unnamed: 4  255 non-null    object
 5   Unnamed: 5  255 non-null    object
 6   Unnamed: 6  255 non-null    object
 7   Unnamed: 7  255 non-null    object
 8   Unnamed: 8  255 non-null    object
 9   Unnamed: 9  255 non-null    object
dtypes: float64(1), object(9)
memory usage: 20.1+ KB
```

Checking the data

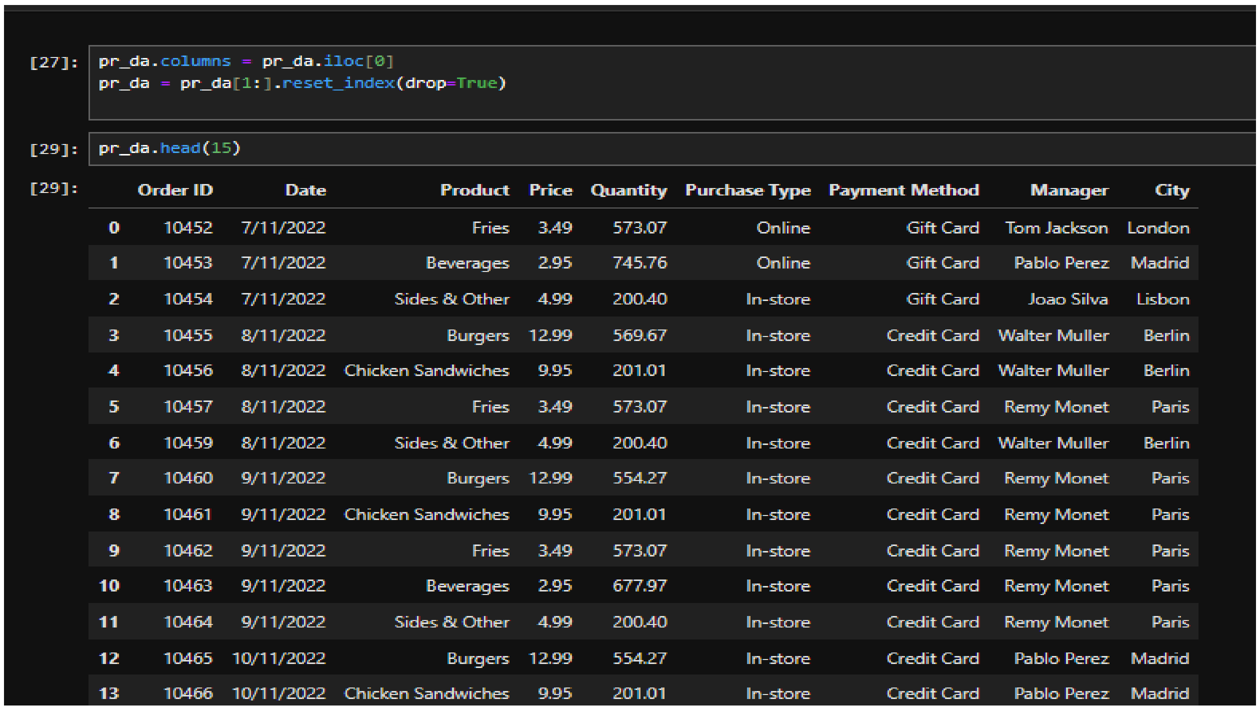
No duplicates were found
The Data consists of A total of 225 Rows and 10 Columns.
The Data type of The data is incorrect It also has A Null column and the header is also incorrect

2.1 Removing Null Columns and Rows



```
[18]: pr_da=Project_Data.drop(columns=["Unnamed: 0"])

[23]: pr_da.head(15)

[23]:     Unnamed: 1  Unnamed: 2          Unnamed: 3  Unnamed: 4  Unnamed: 5  Unnamed: 6      Unnamed: 7      Unnamed: 8    Unnamed: 9
       0    Order ID        Date             Product       Price    Quantity  Purchase Type  Payment Method    Manager          City
       1       10452   7/11/2022               Fries        3.49      573.07        Online       Gift Card    Tom Jackson     London
       2       10453   7/11/2022           Beverages        2.95      745.76        Online       Gift Card    Pablo Perez     Madrid
       3       10454   7/11/2022       Sides & Other        4.99      200.40      In-store       Gift Card     Joao Silva     Lisbon
       4       10455   8/11/2022             Burgers       12.99      569.67      In-store     Credit Card  Walter Muller     Berlin
       5       10456   8/11/2022  Chicken Sandwiches        9.95      201.01      In-store     Credit Card  Walter Muller     Berlin
       6       10457   8/11/2022               Fries        3.49      573.07      In-store     Credit Card    Remy Monet       Paris
       7       10459   8/11/2022       Sides & Other        4.99      200.40      In-store     Credit Card  Walter Muller     Berlin
       8       10460   9/11/2022             Burgers       12.99      554.27      In-store     Credit Card    Remy Monet       Paris
       9       10461   9/11/2022  Chicken Sandwiches        9.95      201.01      In-store     Credit Card    Remy Monet       Paris
      10       10462   9/11/2022               Fries        3.49      573.07      In-store     Credit Card    Remy Monet       Paris
      11       10463   9/11/2022           Beverages        2.95      677.97      In-store     Credit Card    Remy Monet       Paris
      12       10464   9/11/2022       Sides & Other        4.99      200.40      In-store     Credit Card    Remy Monet       Paris
      13       10465  10/11/2022             Burgers       12.99      554.27      In-store     Credit Card    Pablo Perez     Madrid
      14       10466  10/11/2022  Chicken Sandwiches        9.95      201.01      In-store     Credit Card    Pablo Perez     Madrid
```

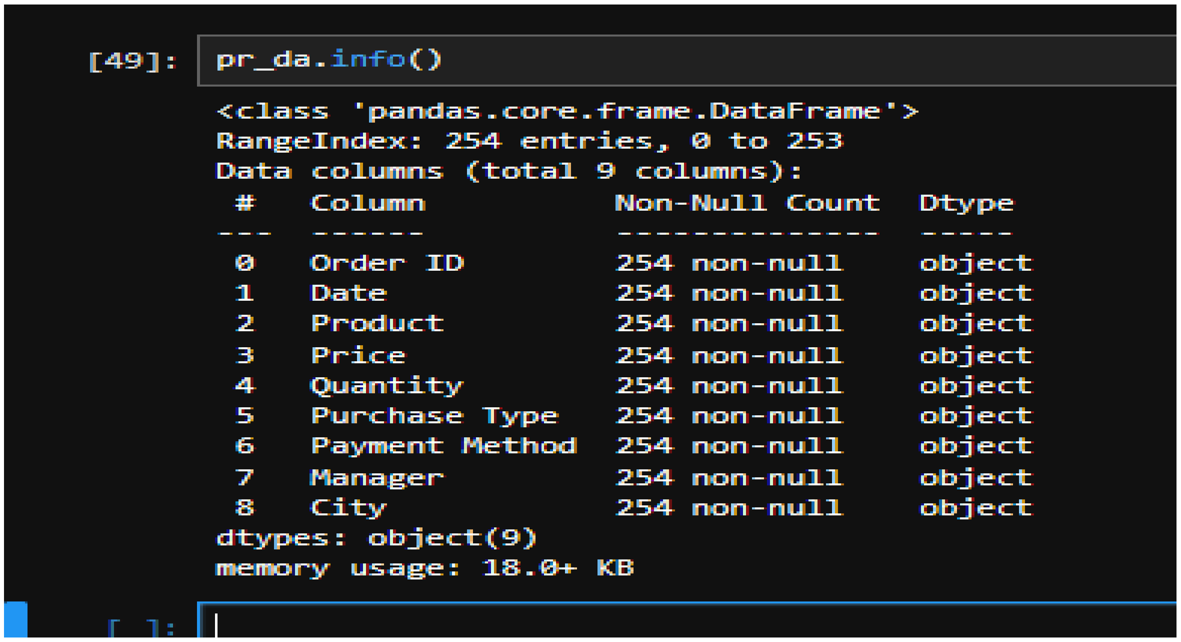
Removing null column



```
[27]: pr_da.columns = pr_da.iloc[0]
      pr_da = pr_da[1:].reset_index(drop=True)

[29]: pr_da.head(15)

[29]:    Order ID        Date             Product  Price  Quantity  Purchase Type  Payment Method        Manager    City
      0     10452   7/11/2022               Fries   3.49    573.07         Online       Gift Card    Tom Jackson  London
      1     10453   7/11/2022           Beverages   2.95    745.76         Online       Gift Card    Pablo Perez  Madrid
      2     10454   7/11/2022       Sides & Other   4.99    200.40       In-store       Gift Card     Joao Silva  Lisbon
      3     10455   8/11/2022             Burgers  12.99    569.67       In-store     Credit Card  Walter Muller  Berlin
      4     10456   8/11/2022  Chicken Sandwiches   9.95    201.01       In-store     Credit Card  Walter Muller  Berlin
      5     10457   8/11/2022               Fries   3.49    573.07       In-store     Credit Card    Remy Monet    Paris
      6     10459   8/11/2022       Sides & Other   4.99    200.40       In-store     Credit Card  Walter Muller  Berlin
      7     10460   9/11/2022             Burgers  12.99    554.27       In-store     Credit Card    Remy Monet    Paris
      8     10461   9/11/2022  Chicken Sandwiches   9.95    201.01       In-store     Credit Card    Remy Monet    Paris
      9     10462   9/11/2022               Fries   3.49    573.07       In-store     Credit Card    Remy Monet    Paris
     10     10463   9/11/2022           Beverages   2.95    677.97       In-store     Credit Card    Remy Monet    Paris
     11     10464   9/11/2022       Sides & Other   4.99    200.40       In-store     Credit Card    Remy Monet    Paris
     12     10465  10/11/2022             Burgers  12.99    554.27       In-store     Credit Card    Pablo Perez  Madrid
     13     10466  10/11/2022  Chicken Sandwiches   9.95    201.01       In-store     Credit Card    Pablo Perez  Madrid
```

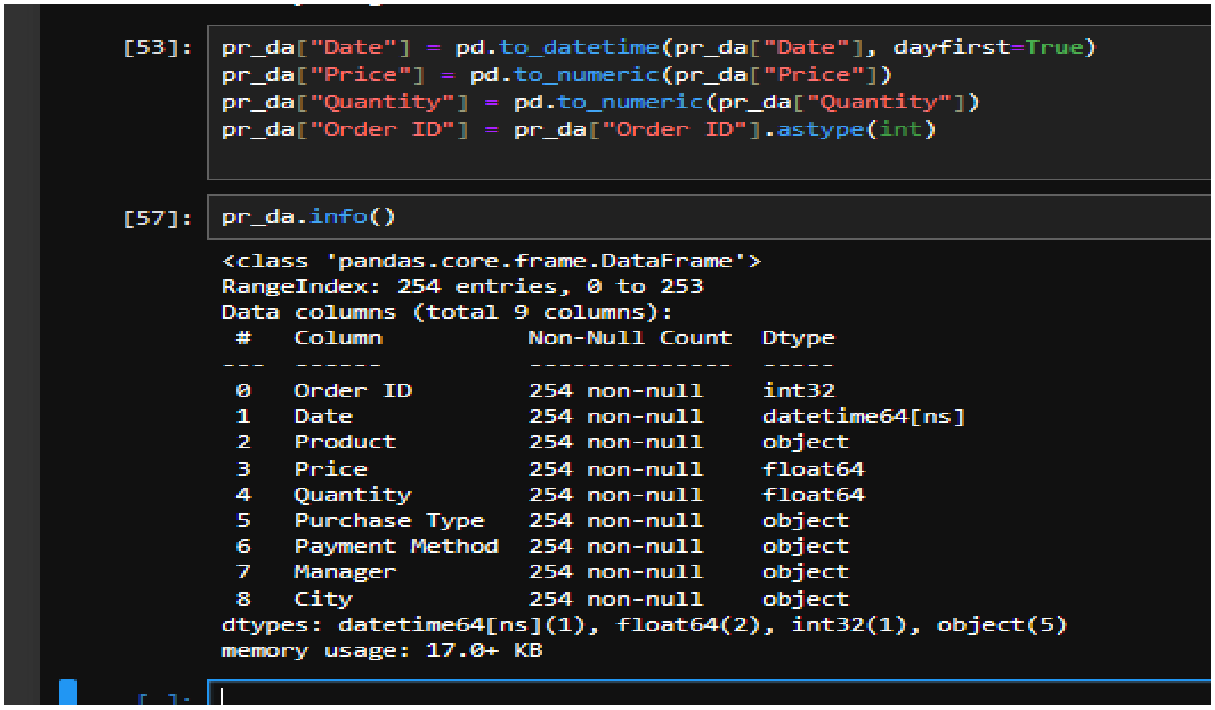
Removing the first null row

2.2 Appropriating the Correct Data types



```
[49]: pr_da.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 254 entries, 0 to 253
Data columns (total 9 columns):
 #   Column          Non-Null Count  Dtype 
---  ------          --------------  ----- 
 0   Order ID        254 non-null    object
 1   Date            254 non-null    object
 2   Product         254 non-null    object
 3   Price           254 non-null    object
 4   Quantity        254 non-null    object
 5   Purchase Type   254 non-null    object
 6   Payment Method  254 non-null    object
 7   Manager         254 non-null    object
 8   City            254 non-null    object
dtypes: object(9)
memory usage: 18.0+ KB

[ ]:
```

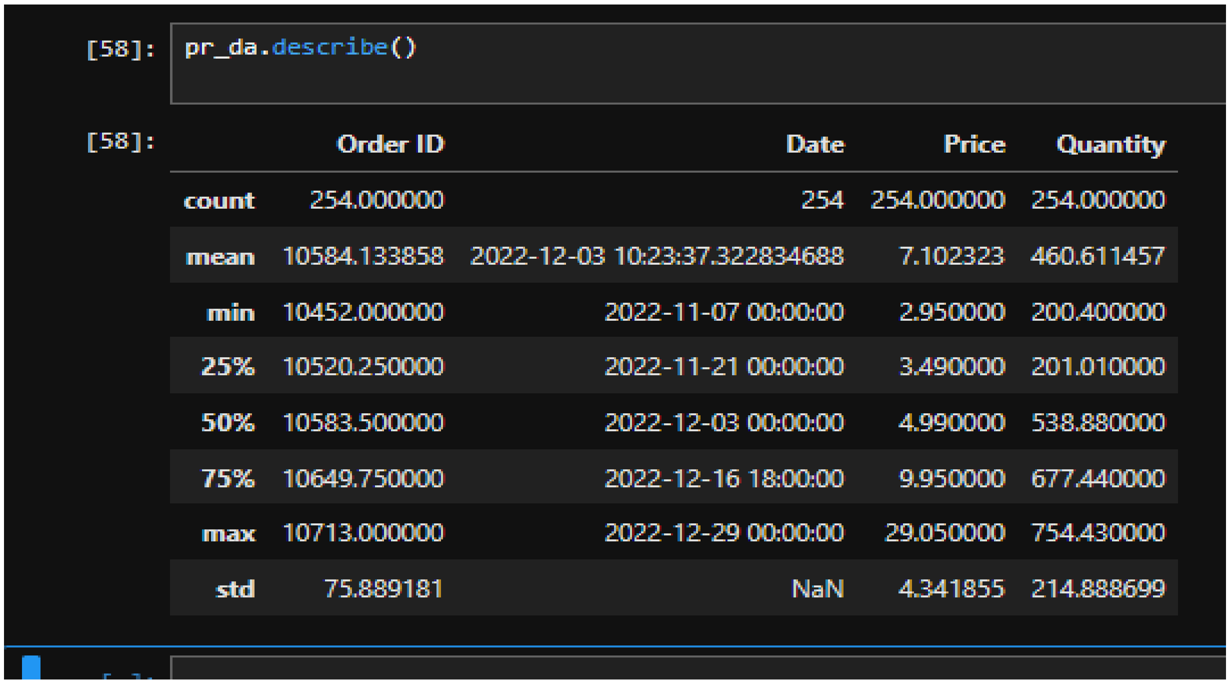
Checking the Data types of the columns



```
[53]: pr_da["Date"] = pd.to_datetime(pr_da["Date"], dayfirst=True)
      pr_da["Price"] = pd.to_numeric(pr_da["Price"])
      pr_da["Quantity"] = pd.to_numeric(pr_da["Quantity"])
      pr_da["Order ID"] = pr_da["Order ID"].astype(int)

[57]: pr_da.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 254 entries, 0 to 253
Data columns (total 9 columns):
 #   Column          Non-Null Count  Dtype         
---  ------          --------------  -----         
 0   Order ID        254 non-null    int32         
 1   Date            254 non-null    datetime64[ns]
 2   Product         254 non-null    object        
 3   Price           254 non-null    float64       
 4   Quantity        254 non-null    float64       
 5   Purchase Type   254 non-null    object        
 6   Payment Method  254 non-null    object        
 7   Manager         254 non-null    object        
 8   City            254 non-null    object        
dtypes: datetime64[ns](1), float64(2), int32(1), object(5)
memory usage: 17.0+ KB

[ ]:
```

Corrected the Data types of the columns

2.3 Understanding the data by using descriptive statistics

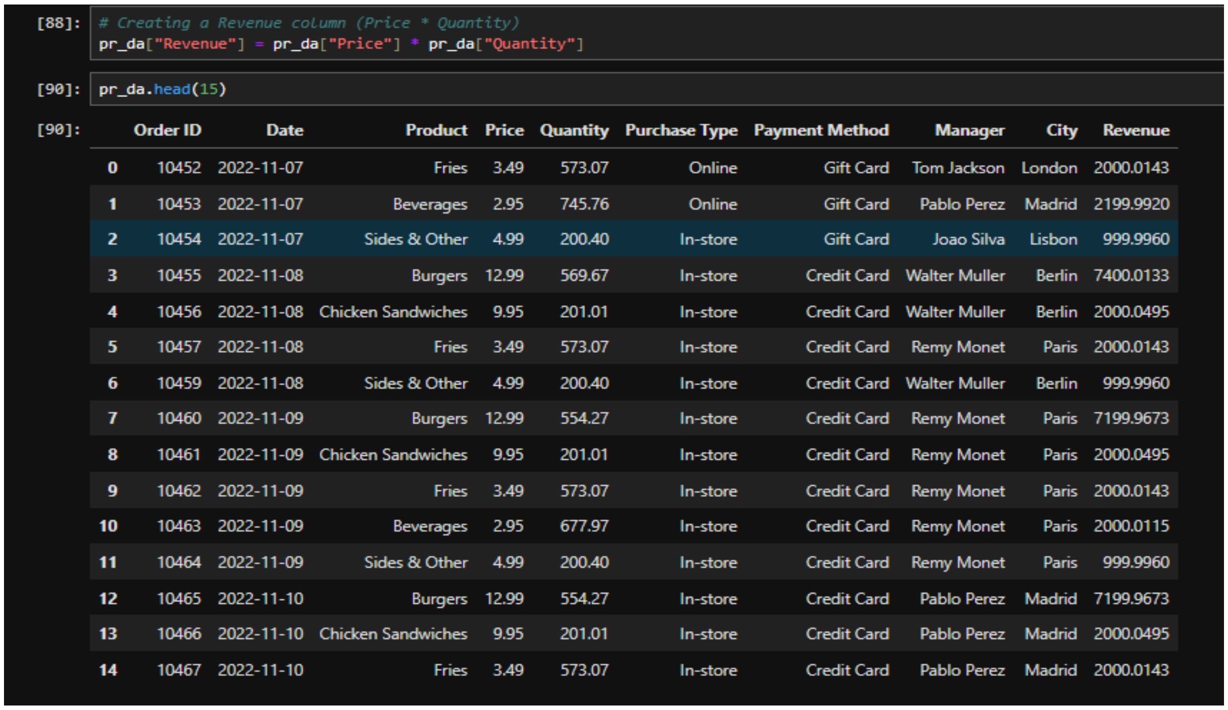


```
[58]: pr_da.describe()
```

```
[58]:
           Order ID                           Date       Price    Quantity
count    254.000000                            254  254.000000  254.000000
mean   10584.133858  2022-12-03 10:23:37.322834688    7.102323  460.611457
min    10452.000000            2022-11-07 00:00:00    2.950000  200.400000
25%    10520.250000            2022-11-21 00:00:00    3.490000  201.010000
50%    10583.500000            2022-12-03 00:00:00    4.990000  538.880000
75%    10649.750000            2022-12-16 18:00:00    9.950000  677.440000
max    10713.000000            2022-12-29 00:00:00   29.050000  754.430000
std       75.889181                            NaN    4.341855  214.888699
```

Using the Describe function to understand the data

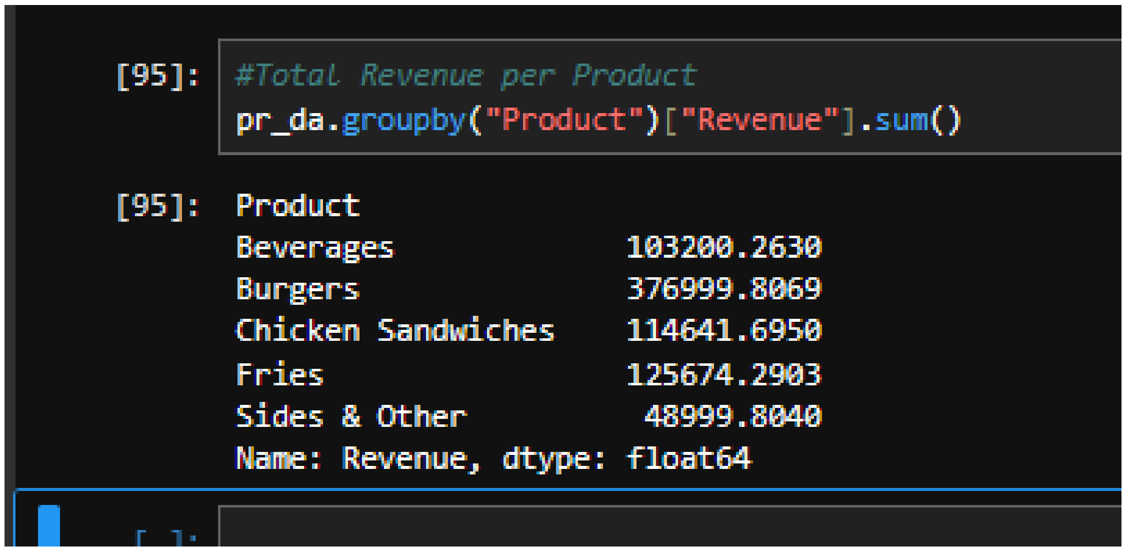
2.4 Using aggregation techniques to explore trends, distributions, and group-wise summaries.



```
[88]: # Creating a Revenue column (Price * Quantity)
      pr_da["Revenue"] = pr_da["Price"] * pr_da["Quantity"]

[90]: pr_da.head(15)
```

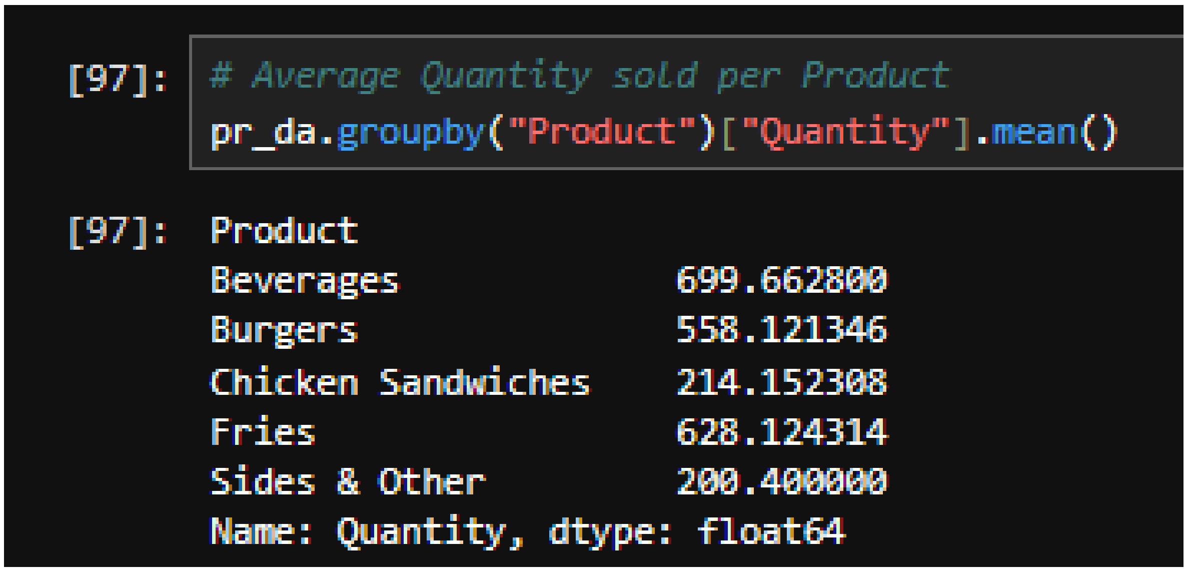
```
[90]:
    Order ID        Date             Product  Price  Quantity  Purchase Type  Payment Method      Manager    City    Revenue
0      10452  2022-11-07               Fries   3.49    573.07         Online       Gift Card  Tom Jackson  London  2000.0143
1      10453  2022-11-07           Beverages   2.95    745.76         Online       Gift Card  Pablo Perez  Madrid  2199.9920
2      10454  2022-11-07       Sides & Other   4.99    200.40       In-store       Gift Card   Joao Silva  Lisbon   999.9960
3      10455  2022-11-08             Burgers  12.99    569.67       In-store     Credit Card Walter Muller  Berlin  7400.0133
4      10456  2022-11-08  Chicken Sandwiches   9.95    201.01       In-store     Credit Card Walter Muller  Berlin  2000.0495
5      10457  2022-11-08               Fries   3.49    573.07       In-store     Credit Card   Remy Monet   Paris  2000.0143
6      10459  2022-11-08       Sides & Other   4.99    200.40       In-store     Credit Card Walter Muller  Berlin   999.9960
7      10460  2022-11-09             Burgers  12.99    554.27       In-store     Credit Card   Remy Monet   Paris  7199.9673
8      10461  2022-11-09  Chicken Sandwiches   9.95    201.01       In-store     Credit Card   Remy Monet   Paris  2000.0495
9      10462  2022-11-09               Fries   3.49    573.07       In-store     Credit Card   Remy Monet   Paris  2000.0143
10     10463  2022-11-09           Beverages   2.95    677.97       In-store     Credit Card   Remy Monet   Paris  2000.0115
11     10464  2022-11-09       Sides & Other   4.99    200.40       In-store     Credit Card   Remy Monet   Paris   999.9960
12     10465  2022-11-10             Burgers  12.99    554.27       In-store     Credit Card  Pablo Perez  Madrid  7199.9673
13     10466  2022-11-10  Chicken Sandwiches   9.95    201.01       In-store     Credit Card  Pablo Perez  Madrid  2000.0495
14     10467  2022-11-10               Fries   3.49    573.07       In-store     Credit Card  Pablo Perez  Madrid  2000.0143
```

Creating a Revenue column



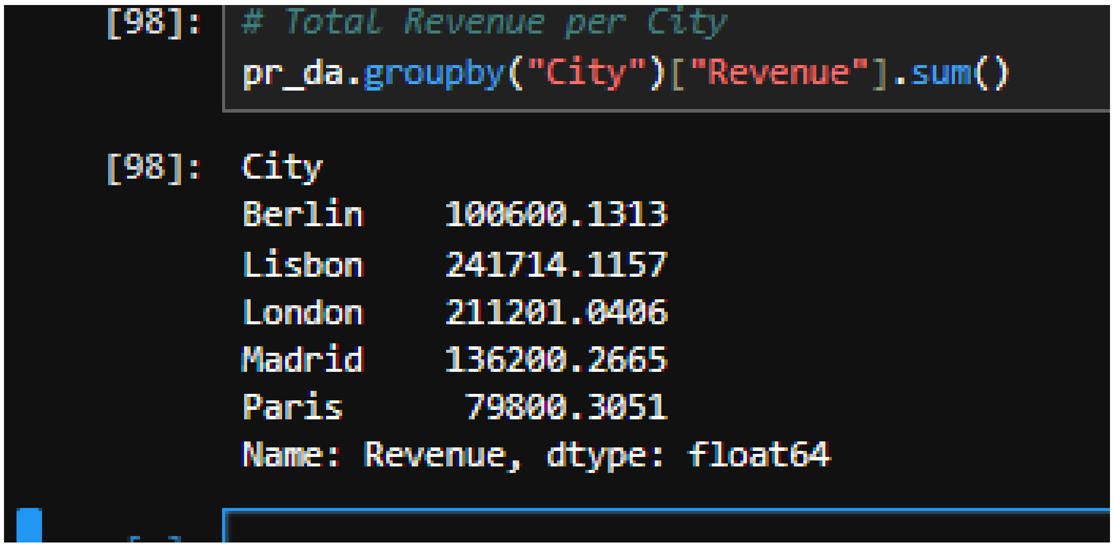
```
[95]: #Total Revenue per Product
      pr_da.groupby("Product")["Revenue"].sum()

[95]: Product
      Beverages             103200.2630
      Burgers               376999.8069
      Chicken Sandwiches    114641.6950
      Fries                 125674.2903
      Sides & Other          48999.8040
      Name: Revenue, dtype: float64
```

Total Revenue per Product



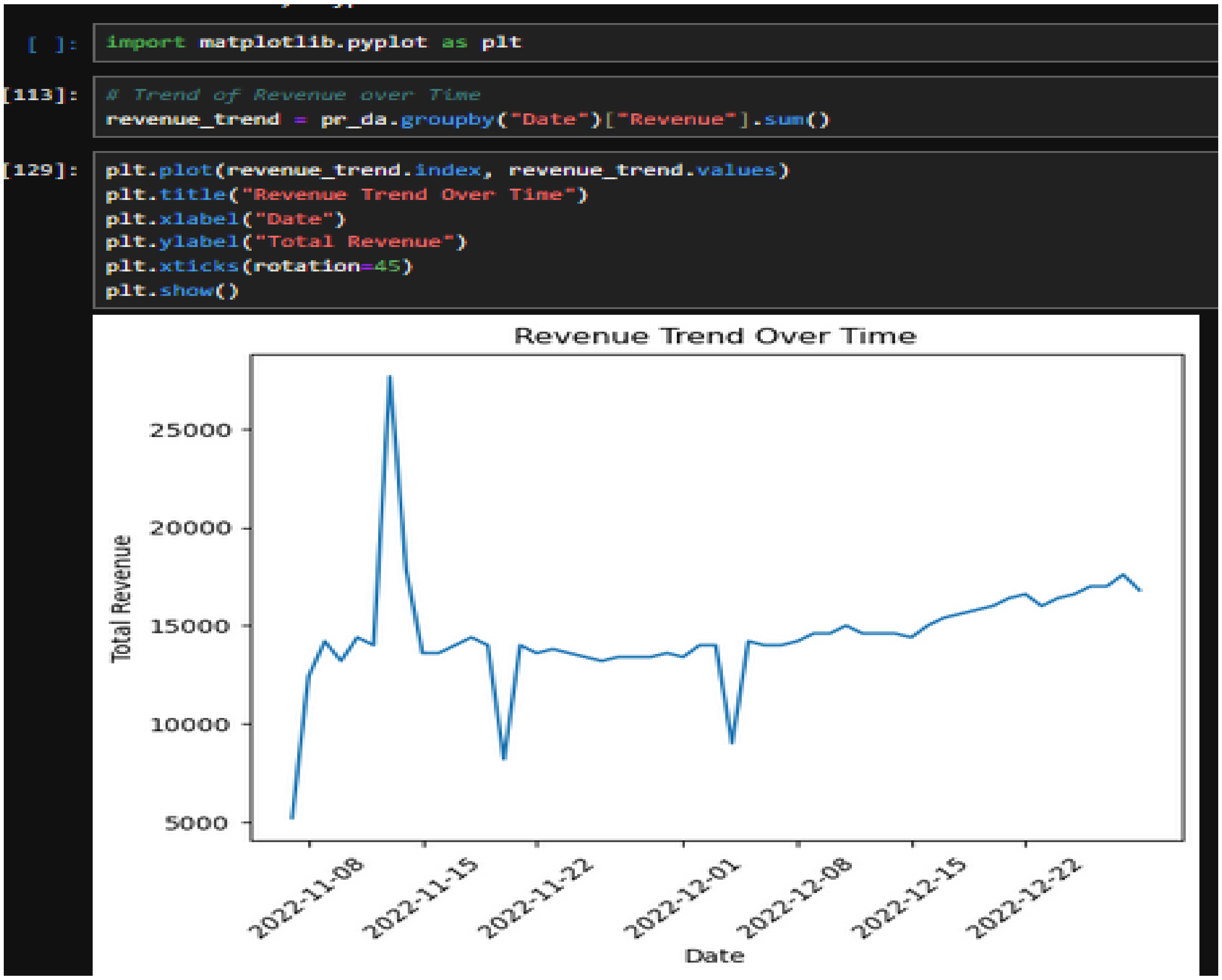
```
[97]: # Average Quantity sold per Product
      pr_da.groupby("Product")["Quantity"].mean()

[97]: Product
      Beverages             699.662800
      Burgers               558.121346
      Chicken Sandwiches    214.152308
      Fries                 628.124314
      Sides & Other         200.400000
      Name: Quantity, dtype: float64
```

Average Quantity sold per Product



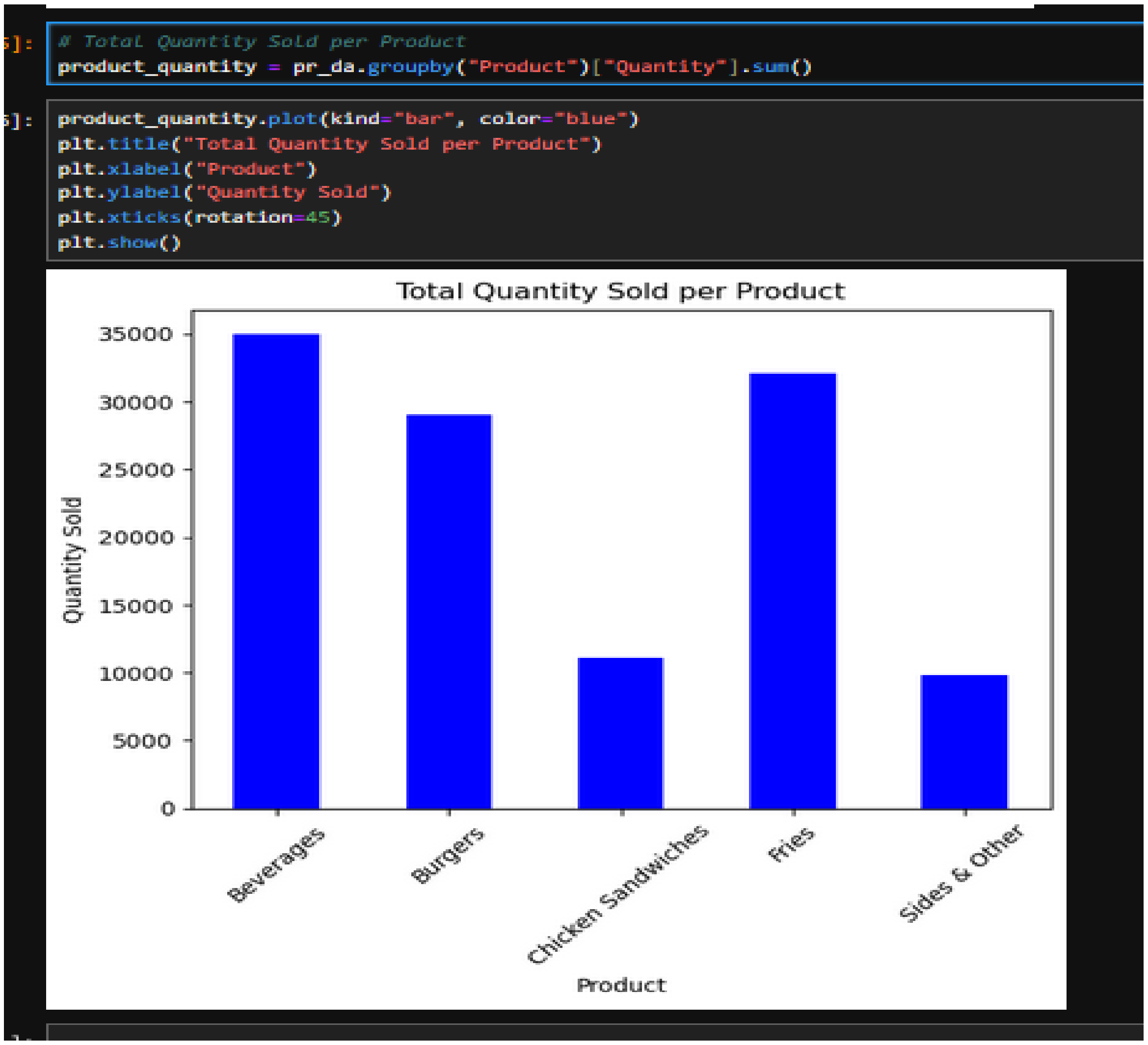
```
[98]: # Total Revenue per City
      pr_da.groupby("City")["Revenue"].sum()

[98]: City
      Berlin     100600.1313
      Lisbon     241714.1157
      London     211201.0406
      Madrid     136200.2665
      Paris       79800.3051
      Name: Revenue, dtype: float64
```

Total Revenue per City



```
[ ]: import matplotlib.pyplot as plt

[113]: # Trend of Revenue over Time
       revenue_trend = pr_da.groupby("Date")["Revenue"].sum()

[129]: plt.plot(revenue_trend.index, revenue_trend.values)
       plt.title("Revenue Trend Over Time")
       plt.xlabel("Date")
       plt.ylabel("Total Revenue")
       plt.xticks(rotation=45)
       plt.show()
```

Trend of Revenue over Time



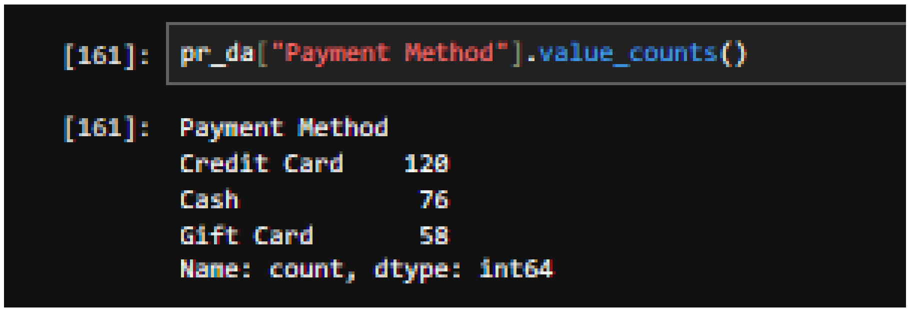
```
# Total Quantity Sold per Product
product_quantity = pr_da.groupby("Product")["Quantity"].sum()
```

```
product_quantity.plot(kind="bar", color="blue")
plt.title("Total Quantity Sold per Product")
plt.xlabel("Product")
plt.ylabel("Quantity Sold")
plt.xticks(rotation=45)
plt.show()
```

Total Quantity Sold per Product

3.0 Analysis of Key Business Questions

Q1. What was the Most Preferred Payment Method

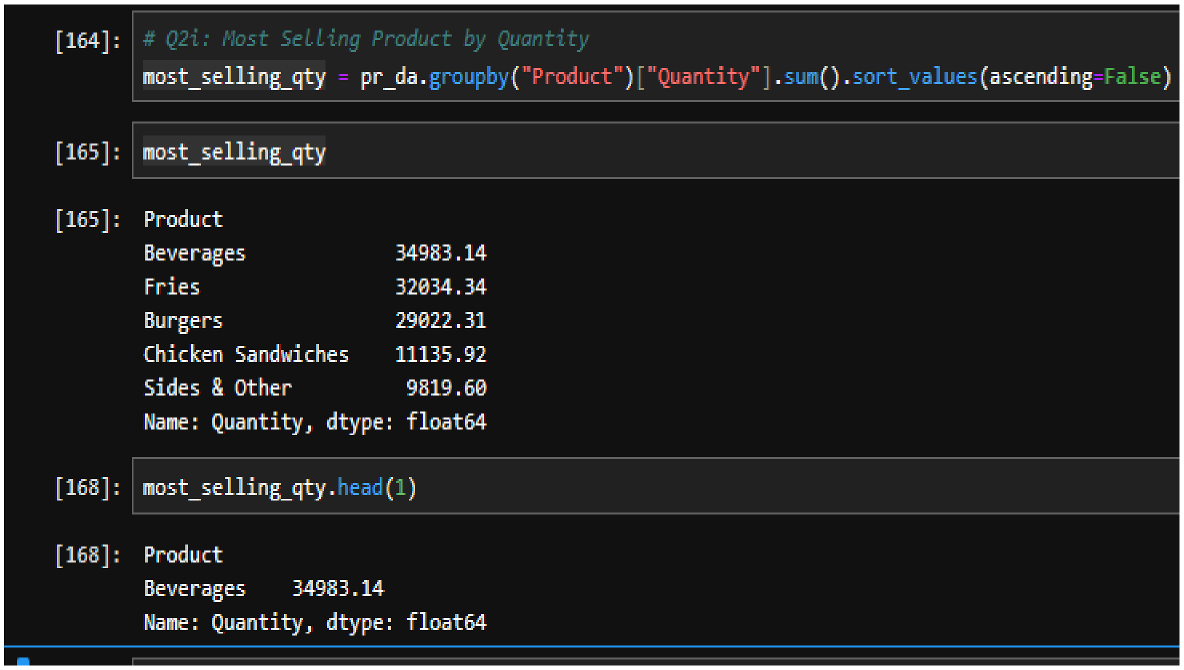


```
[161]:  pr_da["Payment Method"].value_counts()

[161]:  Payment Method
        Credit Card    120
        Cash            76
        Gift Card       58
        Name: count, dtype: int64
```

After analyzing the dataset, the distribution of payment methods shows that Credit Card was the most preferred method of payment. It accounted for the highest number of transactions compared to Cash and Gift Card.

This indicates that most customers prefer cashless transactions, which could suggest higher trust in electronic payment systems and also convenience for larger purchases. Businesses can focus on improving digital payment channels and offering incentives for card payments since that’s the dominant choice.

Q2. Which one was the Most Selling Product by Quantity and by Revenue?



```
[164]: # Q2i: Most Selling Product by Quantity
       most_selling_qty = pr_da.groupby("Product")["Quantity"].sum().sort_values(ascending=False)

[165]: most_selling_qty

[165]: Product
       Beverages           34983.14
       Fries               32034.34
       Burgers             29022.31
       Chicken Sandwiches  11135.92
       Sides & Other        9819.60
       Name: Quantity, dtype: float64

[168]: most_selling_qty.head(1)

[168]: Product
       Beverages    34983.14
       Name: Quantity, dtype: float64
```

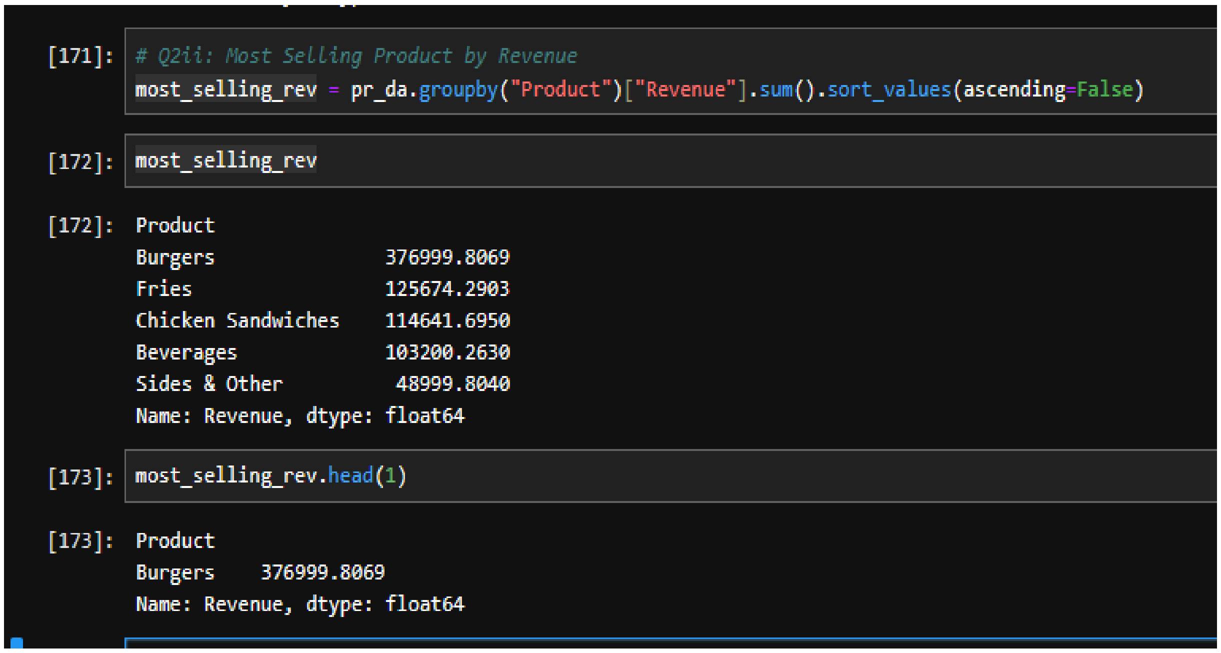
Most Selling Product by Quantity

The analysis shows that Beverages had the highest quantity sold (34,983.14 units), followed by Fries (32,034.34 units) and Burgers (29,022.31 units). Other product categories such as Chicken Sandwiches and Sides & Other had significantly lower sales volumes.

Insight from This Result:
This result highlights that Beverages dominate customer purchases in terms of volume, which suggests a consistent demand for drinks alongside meals. The close competition between Fries and Burgers also indicates that these are staple items frequently chosen by customers.

From a business perspective, this insight suggests:

- Inventory Management: Beverages, Fries, and Burgers should be prioritized in stock planning to meet demand consistently.
- Cross-Selling Opportunities: Since beverages are highly purchased, bundling them with other items (e.g., burgers or sandwiches) could further boost sales.
- Promotional Strategy: Focused promotions on high-volume items could reinforce customer loyalty while introducing offers on lower-performing products (e.g., Chicken Sandwiches, Sides) to balance sales.



```
[171]: # Q2ii: Most Selling Product by Revenue
       most_selling_rev = pr_da.groupby("Product")["Revenue"].sum().sort_values(ascending=False)

[172]: most_selling_rev

[172]: Product
       Burgers               376999.8069
       Fries                 125674.2903
       Chicken Sandwiches    114641.6950
       Beverages             103200.2630
       Sides & Other          48999.8040
       Name: Revenue, dtype: float64

[173]: most_selling_rev.head(1)

[173]: Product
       Burgers    376999.8069
       Name: Revenue, dtype: float64
```

Most Selling Product by Revenue

The analysis reveals that Burgers generated the highest revenue at approximately 376,999.81, followed by Fries (125,674.29) and Chicken Sandwiches (114,641.70). Despite being the most purchased item by quantity, Beverages only generated about 103,200.26 in revenue, ranking below the main food categories. Sides & Other contributed the least, at around 48,999.80.

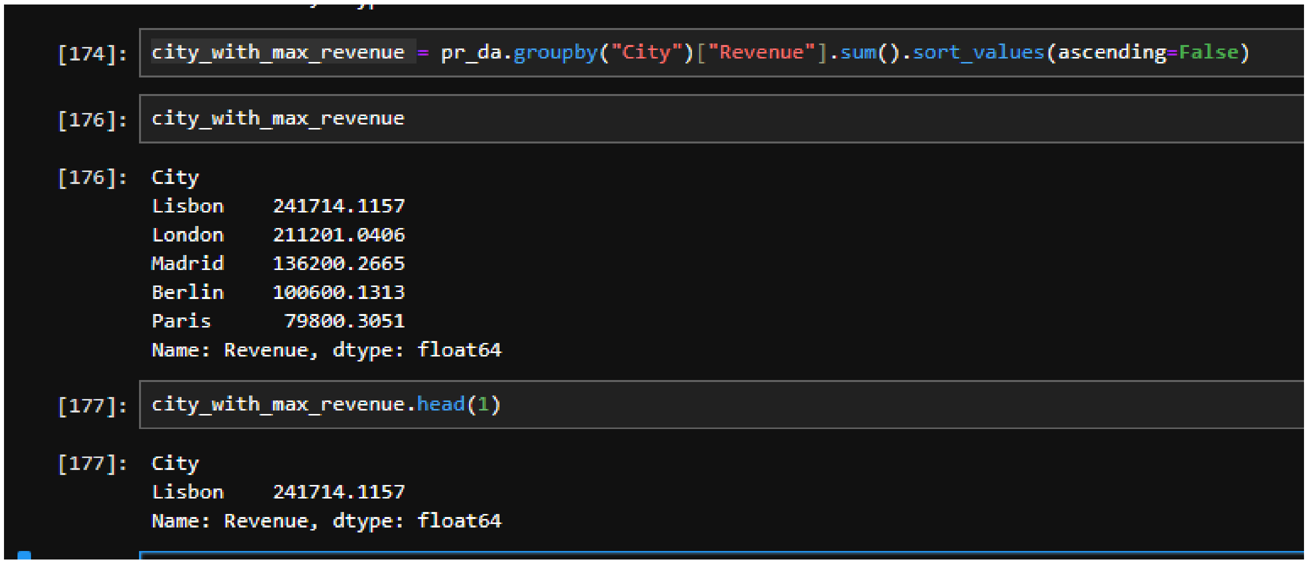
Insights from the result:

- Burgers are the top revenue driver, showing their strong pricing power and profitability compared to other products.
- The difference between quantity and revenue patterns indicates that Beverages sell the most units but at lower prices, while Burgers, though fewer in units than beverages, contribute far more financially.
- This highlights the importance of product pricing in overall revenue contribution.

Business Implications:

- Profit Maximization: Since Burgers are the main revenue driver, they should remain the focus of marketing campaigns, premium combos, and upselling opportunities.
- Value Bundling: Pairing Beverages (high volume) with Burgers (high revenue) can maximize both unit sales and profitability.
- Growth Potential: Sides & Other could be strategically promoted or rebranded to increase their contribution to total revenue.

Q3. Which City had maximum revenue, and Which Manager earned maximum revenue?



```
[174]: city_with_max_revenue = pr_da.groupby("City")["Revenue"].sum().sort_values(ascending=False)

[176]: city_with_max_revenue

[176]: City
       Lisbon    241714.1157
       London    211201.0406
       Madrid    136200.2665
       Berlin    100600.1313
       Paris      79800.3051
       Name: Revenue, dtype: float64

[177]: city_with_max_revenue.head(1)

[177]: City
       Lisbon    241714.1157
       Name: Revenue, dtype: float64
```

City with maximum revenue

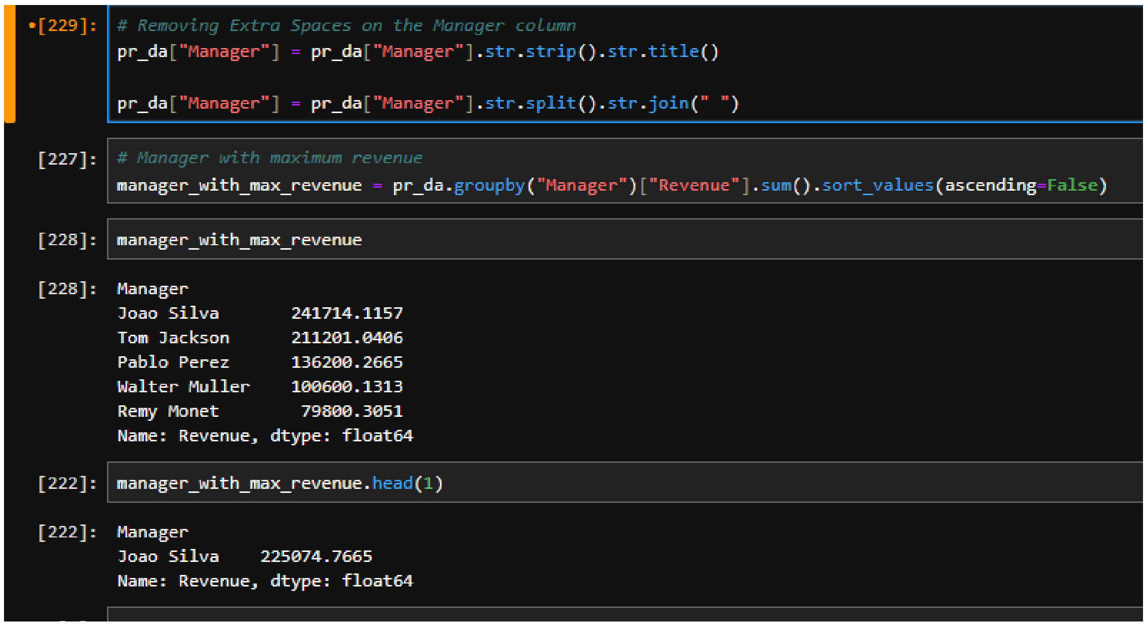
The analysis shows that Lisbon generated the highest revenue at approximately 241,714.12, followed by London (211,201.04) and Madrid (136,200.27). Meanwhile, Paris (79,800.31) recorded the lowest revenue among the listed cities.

Insight from the result:

- Lisbon stands out as the top-performing city, contributing significantly more revenue than the other locations.
- London, while second, still trails Lisbon by a notable margin of about 30,500.
- Paris shows the weakest performance, which may suggest either a smaller market size, weaker sales strategy, or lower customer demand.

Business Implications:

- Resource Allocation: Management should continue to strengthen operations in Lisbon while investigating strategies to replicate its success in other cities.
- Market Development: Paris presents an opportunity for improvement through targeted marketing campaigns, customer engagement strategies, or adjusted pricing models.
- Benchmarking: Sales practices in Lisbon could be studied and applied to underperforming cities to raise overall revenue performance.



```
•[229]: # Removing Extra Spaces on the Manager column
        pr_da["Manager"] = pr_da["Manager"].str.strip().str.title()

        pr_da["Manager"] = pr_da["Manager"].str.split().str.join(" ")

 [227]: # Manager with maximum revenue
        manager_with_max_revenue = pr_da.groupby("Manager")["Revenue"].sum().sort_values(ascending=False)

 [228]: manager_with_max_revenue

 [228]: Manager
        Joao Silva       241714.1157
        Tom Jackson      211201.0406
        Pablo Perez      136200.2665
        Walter Muller    100600.1313
        Remy Monet        79800.3051
        Name: Revenue, dtype: float64

 [222]: manager_with_max_revenue.head(1)

 [222]: Manager
        Joao Silva    225074.7665
        Name: Revenue, dtype: float64
```

Manager with maximum revenue

The Extra Spaces in the Manager's column was noticed and removed first, before continuing.

The analysis reveals that Joao Silva generated the highest revenue, contributing approximately 241,714.12, followed by Tom Jackson (211,201.04) and Pablo Perez (136,200.27). On the lower end, Remy Monet recorded about 79,800.31, making him the least-performing manager in terms of sales revenue.

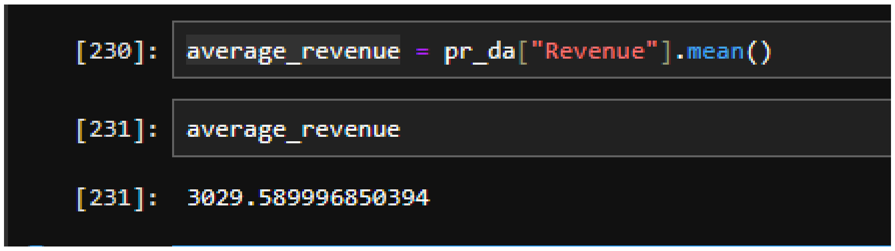
Insight From the Result:

- Joao Silva emerges as the top-performing manager, indicating strong sales execution or customer relationships in his region.
- Tom Jackson's performance is competitive and close behind, suggesting that with additional support, he could potentially rival Joao Silva.
- Managers like Remy Monet lag significantly, and this gap highlights performance disparity within the team.

Business Implications:

- Best Practices Sharing: Strategies and techniques employed by Joao Silva could be studied and replicated across the team.
- Training/Support: Lower-performing managers should receive targeted training or support to boost performance.
- Balanced Incentives: Management could design incentive programs to encourage more consistent revenue generation across managers.

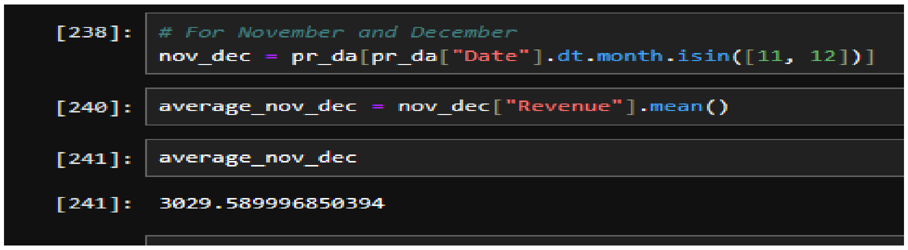
Q4. What was the Average Revenue?



```
[230]: average_revenue = pr_da["Revenue"].mean()

[231]: average_revenue

[231]: 3029.589996850394
```

Average Revenue

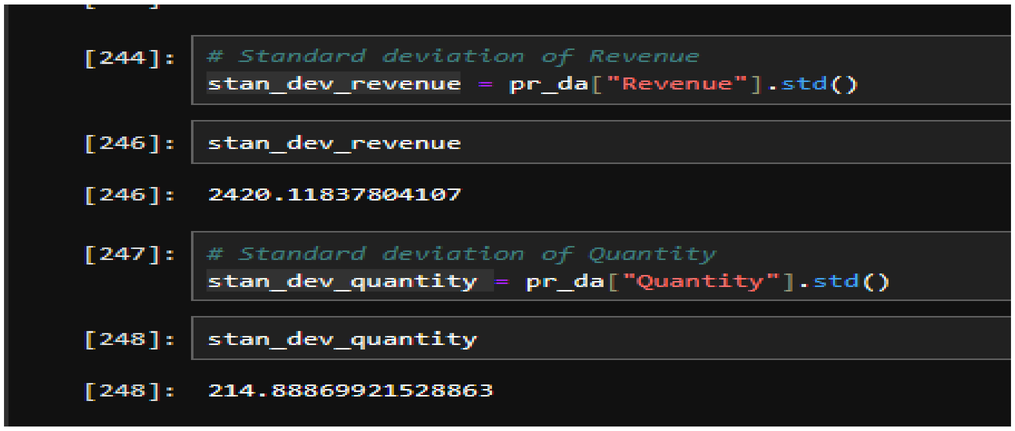
Q5. What was the Average Revenue of November & December?



```
[238]: # For November and December
       nov_dec = pr_da[pr_da["Date"].dt.month.isin([11, 12])]

[240]: average_nov_dec = nov_dec["Revenue"].mean()

[241]: average_nov_dec

[241]: 3029.589996850394
```

Average Revenue of November & December

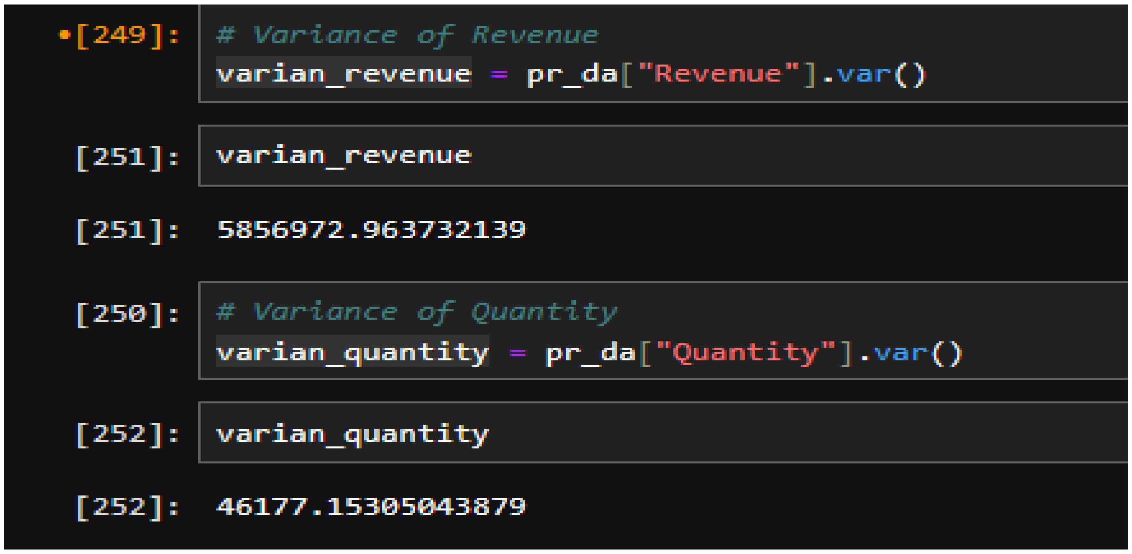
Q6. What was the Standard Deviation of Revenue and Quantity?



```
[244]: # Standard deviation of Revenue
       stan_dev_revenue = pr_da["Revenue"].std()

[246]: stan_dev_revenue

[246]: 2420.11837804107

[247]: # Standard deviation of Quantity
       stan_dev_quantity = pr_da["Quantity"].std()

[248]: stan_dev_quantity

[248]: 214.88869921528863
```

Standard Deviation of Revenue and Quantity

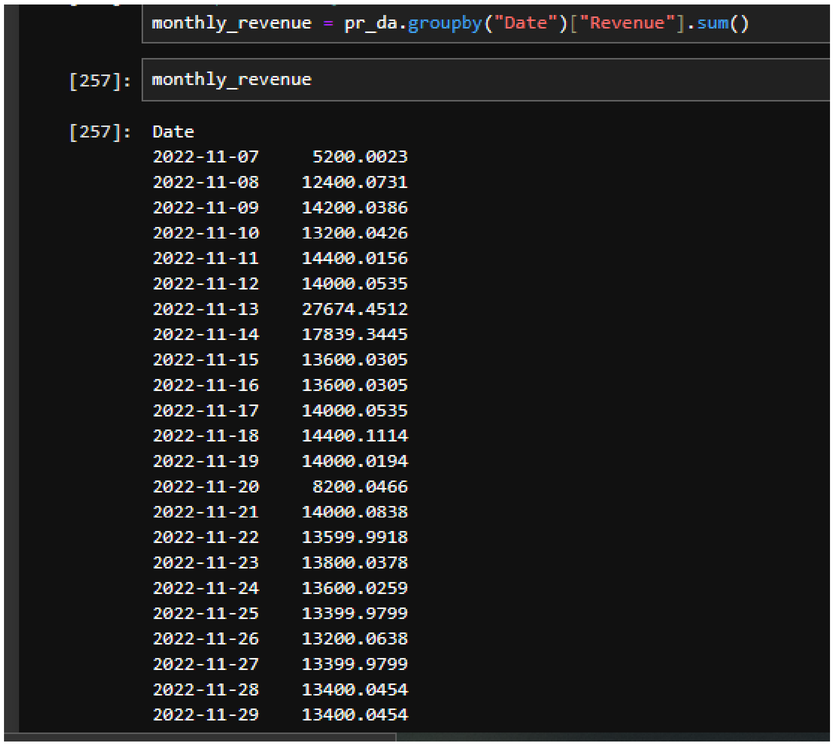
7. What was the Variance of Revenue and Quantity?



```
•[249]: # Variance of Revenue
        varian_revenue = pr_da["Revenue"].var()

 [251]: varian_revenue

 [251]: 5856972.963732139

 [250]: # Variance of Quantity
        varian_quantity = pr_da["Quantity"].var()

 [252]: varian_quantity

 [252]: 46177.15305043879
```

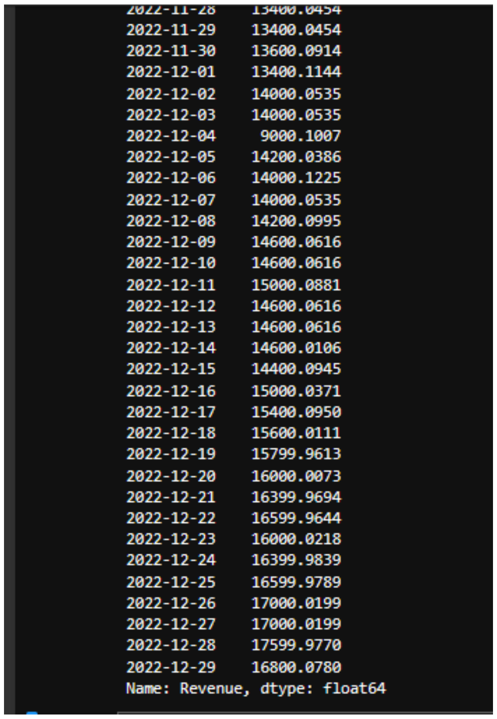
Variance of Revenue and Quantity

8. Was the revenue increasing or decreasing over the time?



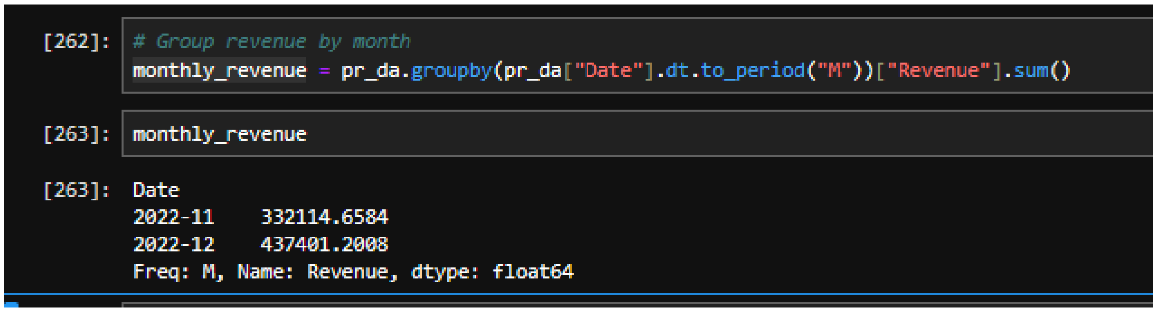
```
        monthly_revenue = pr_da.groupby("Date")["Revenue"].sum()

 [257]: monthly_revenue

 [257]: Date
        2022-11-07     5200.0023
        2022-11-08    12400.0731
        2022-11-09    14200.0386
        2022-11-10    13200.0426
        2022-11-11    14400.0156
        2022-11-12    14000.0535
        2022-11-13    27674.4512
        2022-11-14    17839.3445
        2022-11-15    13600.0305
        2022-11-16    13600.0305
        2022-11-17    14000.0535
        2022-11-18    14400.1114
        2022-11-19    14000.0194
        2022-11-20     8200.0466
        2022-11-21    14000.0838
        2022-11-22    13599.9918
        2022-11-23    13800.0378
        2022-11-24    13600.0259
        2022-11-25    13399.9799
        2022-11-26    13200.0638
        2022-11-27    13399.9799
        2022-11-28    13400.0454
        2022-11-29    13400.0454
```

Trend of revenue over time



```
2022-11-28    13400.0454
2022-11-29    13400.0454
2022-11-30    13600.0914
2022-12-01    13400.1144
2022-12-02    14000.0535
2022-12-03    14000.0535
2022-12-04     9000.1007
2022-12-05    14200.0386
2022-12-06    14000.1225
2022-12-07    14000.0535
2022-12-08    14200.0995
2022-12-09    14600.0616
2022-12-10    14600.0616
2022-12-11    15000.0881
2022-12-12    14600.0616
2022-12-13    14600.0616
2022-12-14    14600.0106
2022-12-15    14400.0945
2022-12-16    15000.0371
2022-12-17    15400.0950
2022-12-18    15600.0111
2022-12-19    15799.9613
2022-12-20    16000.0073
2022-12-21    16399.9694
2022-12-22    16599.9644
2022-12-23    16000.0218
2022-12-24    16399.9839
2022-12-25    16599.9789
2022-12-26    17000.0199
2022-12-27    17000.0199
2022-12-28    17599.9770
2022-12-29    16800.0780
Name: Revenue, dtype: float64
```

Trend of revenue over time



```
[262]: # Group revenue by month
       monthly_revenue = pr_da.groupby(pr_da["Date"].dt.to_period("M"))["Revenue"].sum()

[263]: monthly_revenue

[263]: Date
       2022-11    332114.6584
       2022-12    437401.2008
       Freq: M, Name: Revenue, dtype: float64
```

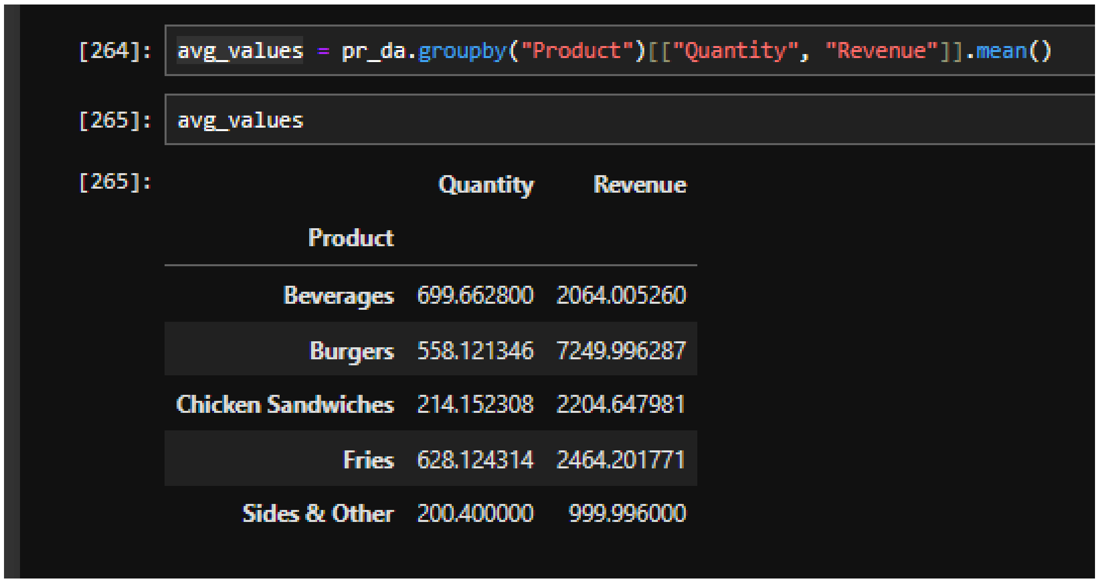
Grouping revenue by month

Revenue increased significantly from November to December (an increase of 105,286.54).

This represents roughly a 31.7% growth month-over-month.

Such a sharp increase is likely influenced by seasonal shopping trends, especially the holiday season in December where demand typically spikes.

Q9. What was the Average 'Quantity Sold' & 'Average Revenue' for each product?



```
[264]: avg_values = pr_da.groupby("Product")[["Quantity", "Revenue"]].mean()

[265]: avg_values

[265]:
                      Quantity      Revenue
            Product
          Beverages 699.662800  2064.005260
            Burgers 558.121346  7249.996287
 Chicken Sandwiches 214.152308  2204.647981
              Fries 628.124314  2464.201771
      Sides & Other 200.400000   999.996000
```

Average Quantity Sold & Average Revenue for each product

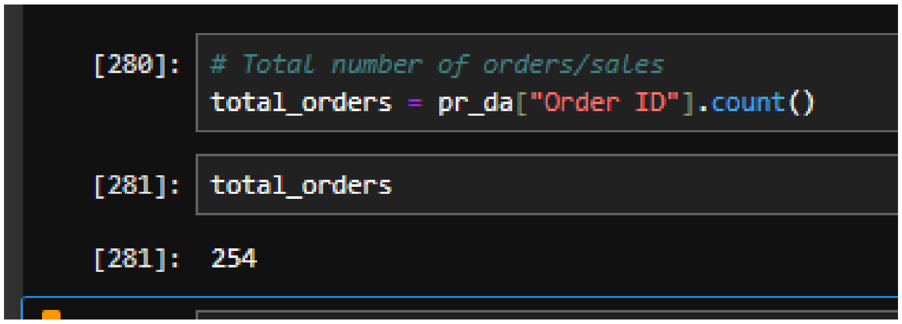
Beverages: Lead in average quantity sold (699.66 units per order), showing they are the highest-volume driver. Customers frequently add drinks to their purchases.

Burgers: Stand out as the highest average revenue per order (7,249.99), making them the most profitable product line despite not having the largest quantity.

Fries: Also move in large volumes (628.12), reinforcing their role as a popular side item often bundled with main meals.

Chicken Sandwiches & Sides/Other: Record relatively lower averages both in quantity and revenue, indicating they are niche or supplementary items.

Q10. What was the total number of orders or sales made?



```
[280]: # Total number of orders/sales
       total_orders = pr_da["Order ID"].count()

[281]: total_orders

[281]: 254
```

total number of orders or sales made

Conclusion

This project successfully demonstrated the end-to-end process of cleaning, analyzing, and extracting insights from raw sales data using **Python, Pandas, and Matplotlib**. Starting with a messy dataset containing missing values, duplicates, inconsistent formats, and extra spaces, the data was transformed into a well-structured and reliable form suitable for analysis.

Through descriptive statistics, aggregation, and visualization, answering several key business questions, such as identifying the most preferred payment method, the top-selling products by both quantity and revenue, the city and manager generating the highest revenue, and the overall sales trends across November and December. We also calculated measures of central tendency and dispersion, such as averages, variance, and standard deviation, to better understand the performance of sales.

The analysis revealed valuable insights, including:

- **Credit Card** as the most preferred payment method.
- **Beverages** driving the highest sales volume, while **Burgers** generated the most revenue.
- **Lisbon** contributing the maximum revenue among cities, and **Joao Silva** emerging as the manager with the highest earnings.
- Sales showed an **increasing trend from November to December**, indicating growing demand.

Overall, this project highlighted the importance of **data cleaning** as a foundation for accurate analysis, and how **Python-based tools** can turn raw sales data into actionable business insights. These findings can help guide future decision-making on product focus, resource allocation, and marketing strategies.