SALES DATA PYTHON ANALYSIS PROJECT REPORT

INTRODUCTION

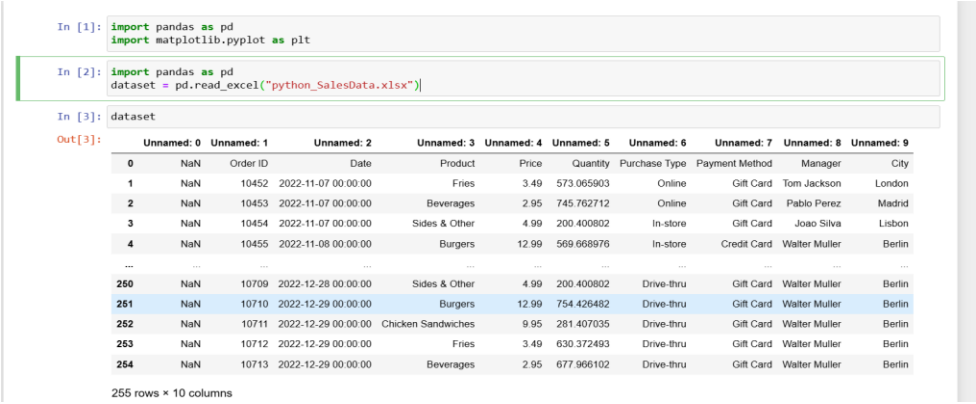
The objective of this project was to demonstrate Python data analysis skills using the Pandas and Matplotlib libraries. The dataset (Python_SalesData.xlsx) contained raw retail sales records, which were cleaned, explored, and analyzed to extract meaningful business insights. The analysis focused on data cleaning, descriptive statistics, trend analysis, and answering 10 business related questions.

Process Overview

1. Imported pandas as pd and matplotlib.pyplot as plt into the jupyter library then imported my dataset using

   import pandas as pd

   dataset = pd.read_excel("python_SalesData.xlsx")



```
In [1]: import pandas as pd
        import matplotlib.pyplot as plt

In [2]: import pandas as pd
        dataset = pd.read_excel("python_SalesData.xlsx")

In [3]: dataset

Out[3]:
```

	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	2022-11-07 00:00:00	Fries	3.49	573.065903	Online	Gift Card	Tom Jackson	London
2	NaN	10453	2022-11-07 00:00:00	Beverages	2.95	745.762712	Online	Gift Card	Pablo Perez	Madrid
3	NaN	10454	2022-11-07 00:00:00	Sides & Other	4.99	200.400802	In-store	Gift Card	Joao Silva	Lisbon
4	NaN	10455	2022-11-08 00:00:00	Burgers	12.99	569.668976	In-store	Credit Card	Walter Muller	Berlin
...	...	...	...	...	...	...	...	...	...	...
250	NaN	10709	2022-12-28 00:00:00	Sides & Other	4.99	200.400802	Drive-thru	Gift Card	Walter Muller	Berlin
251	NaN	10710	2022-12-29 00:00:00	Burgers	12.99	754.426482	Drive-thru	Gift Card	Walter Muller	Berlin
252	NaN	10711	2022-12-29 00:00:00	Chicken Sandwiches	9.95	281.407035	Drive-thru	Gift Card	Walter Muller	Berlin
253	NaN	10712	2022-12-29 00:00:00	Fries	3.49	630.372493	Drive-thru	Gift Card	Walter Muller	Berlin
254	NaN	10713	2022-12-29 00:00:00	Beverages	2.95	677.966102	Drive-thru	Gift Card	Walter Muller	Berlin

255 rows × 10 columns

2. Cleaned the dataset using the following steps
   - Fixed the header and removed unnamed columns

     import pandas as pd

     # Read the file and force the first row as header

     dataset = pd.read_excel("Python_SalesData.xlsx", header=1)
   - Dropped empty columns

     # Drop columns that are completely empty

     dataset = dataset.dropna(axis=1, how='all')
   - Fixed column names

     dataset.columns = ["OrderId", "Date", "Product", "Price", "Quantity",

"PurchaseType", "PaymentMethod", "Manager", "City"]

- To ensure accurate analysis, I converted key columns into appropriate data types: The date column was converted into a datetime format to allow grouping and time-series analysis (e.g., monthly trends). The quantity and price columns were converted into numeric types to enable calculations such as revenue, averages, variance, and standard deviation. Any invalid or non-convertible entries were coerced into missing values (NaT or NaN) for proper handling during data cleaning.

dataset["Date"] = pd.to_datetime(dataset["Date"], errors="coerce")
dataset["Price"] = pd.to_numeric(dataset["Price"], errors="coerce")
dataset["Quantity"] = pd.to_numeric(dataset["Quantity"], errors="coerce")

- Created a revenue column

*dataset["Revenue"] = dataset["Price"] * dataset["Quantity"]*

```
In [4]: import pandas as pd

        # Read the file and force the first row as header
        dataset = pd.read_excel("Python_SalesData.xlsx", header=1)
```

```
In [5]: # Drop columns that are completely empty
        dataset = dataset.dropna(axis=1, how='all')
```

```
In [6]: dataset.columns = ["OrderId", "Date", "Product", "Price", "Quantity",
                           "PurchaseType", "PaymentMethod", "Manager", "City"]
```

```
In [7]: dataset["Date"] = pd.to_datetime(dataset["Date"], errors="coerce")
        dataset["Price"] = pd.to_numeric(dataset["Price"], errors="coerce")
        dataset["Quantity"] = pd.to_numeric(dataset["Quantity"], errors="coerce")
```

```
In [8]: dataset["Revenue"] = dataset["Price"] * dataset["Quantity"]
```

- Loaded my cleaned dataset for analysis.

```
In [10]: dataset
```

Out[10]:

	OrderId	Date	Product	Price	Quantity	PurchaseType	PaymentMethod	Manager	City	Revenue
0	10452	2022-11-07	Fries	3.49	573.065903	Online	Gift Card	Tom Jackson	London	2000.0
1	10453	2022-11-07	Beverages	2.95	745.762712	Online	Gift Card	Pablo Perez	Madrid	2200.0
2	10454	2022-11-07	Sides & Other	4.99	200.400802	In-store	Gift Card	Joao Silva	Lisbon	1000.0
3	10455	2022-11-08	Burgers	12.99	569.668976	In-store	Credit Card	Walter Muller	Berlin	7400.0
4	10456	2022-11-08	Chicken Sandwiches	9.95	201.005025	In-store	Credit Card	Walter Muller	Berlin	2000.0
...	...	...	...	...	...	...	...	...	...	...
249	10709	2022-12-28	Sides & Other	4.99	200.400802	Drive-thru	Gift Card	Walter Muller	Berlin	1000.0
250	10710	2022-12-29	Burgers	12.99	754.426482	Drive-thru	Gift Card	Walter Muller	Berlin	9800.0
251	10711	2022-12-29	Chicken Sandwiches	9.95	281.407035	Drive-thru	Gift Card	Walter Muller	Berlin	2800.0
252	10712	2022-12-29	Fries	3.49	630.372493	Drive-thru	Gift Card	Walter Muller	Berlin	2200.0
253	10713	2022-12-29	Beverages	2.95	677.966102	Drive-thru	Gift Card	Walter Muller	Berlin	2000.0

254 rows × 10 columns

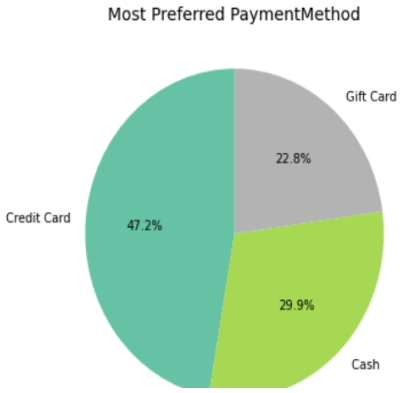
BUSINESS QUESTIONS & ANSWERS

The following project questions were answered using Pandas aggregation and statistical functions:

Q1. What was the Most Preferred Payment Method?

In [25]:
```
# Q1. Most Preferred Payment Method
most_preferred_payment = dataset['PaymentMethod'].value_counts().idxmax()
print("Most Preferred PaymentMethod:", most_preferred_payment)

# Q1: Payment Method Pie Chart
plt.figure(figsize=(6,6))
dataset['PaymentMethod'].value_counts().plot(
    kind='pie', autopct='%1.1f%%', startangle=90, cmap='Set2')
plt.title("Most Preferred PaymentMethod", fontsize=14)
plt.ylabel("")
plt.show()
```

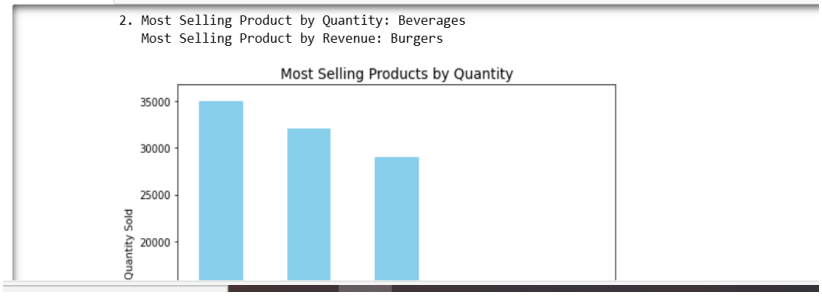
```
Most Preferred PaymentMethod:  Credit Card
```



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

In [28]:
```
# --- Q2. Most Selling Product by Quantity and by Revenue ---
top_product_quantity = dataset.groupby('Product')['Quantity'].sum().idxmax()
top_product_revenue = dataset.groupby('Product')['Revenue'].sum().idxmax()
print("2. Most Selling Product by Quantity:", top_product_quantity)
print("   Most Selling Product by Revenue:", top_product_revenue)

# Q2. Most Selling Products by Quantity
product_quantity = dataset.groupby('Product')['Quantity'].sum().sort_values(ascending=False)

plt.figure(figsize=(8,6))
product_quantity.plot(kind='bar', color='skyblue')
plt.title("Most Selling Products by Quantity", fontsize=14)
plt.ylabel("Total Quantity Sold")
plt.xlabel("Product")
plt.xticks(rotation=45)
plt.show()
```

```
2. Most Selling Product by Quantity: Beverages
   Most Selling Product by Revenue: Burgers
```



Q3. Which City had the maximum Revenue, and which Manager earned the Maximum Revenue?

In [14]:
```
# --- Q3. City & Manager with Maximum Revenue ---
top_city = dataset.groupby('City')['Revenue'].sum().idxmax()
top_manager = dataset.groupby('Manager')['Revenue'].sum().idxmax()
print("3. City with Maximum Revenue:", top_city)
print("   Manager with Maximum Revenue:", top_manager)
```

```
3. City with Maximum Revenue: Lisbon
   Manager with Maximum Revenue: Joao Silva
```

Q4. What was the Average Revenue?

In [34]:
```
# --- Q4. Average Revenue ---
avg_revenue = dataset['Revenue'].mean()
print("4. Average Revenue:", round(avg_revenue, 2))
```

```
4. Average Revenue: 3029.58
```

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

n [16]:
```
# --- Q5. Average Revenue in November & December ---
nov_dec = dataset[dataset['Date'].dt.month.isin([11,12])]
avg_rev_nov_dec = nov_dec['Revenue'].mean()
print("5. Average Revenue (Nov & Dec):", round(avg_rev_nov_dec, 2))
```

```
5. Average Revenue (Nov & Dec): 3029.58
```

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

In [17]:
```
# --- Q6. Standard Deviation of Revenue & Quantity ---
std_revenue = dataset['Revenue'].std()
std_quantity = dataset['Quantity'].std()
print("6. Std Dev Revenue:", round(std_revenue, 2))
print("   Std Dev Quantity:", round(std_quantity, 2))
```

```
6. Std Dev Revenue: 2420.12
   Std Dev Quantity: 214.89
```

Q7. What was the Variance of revenue and Quantity?

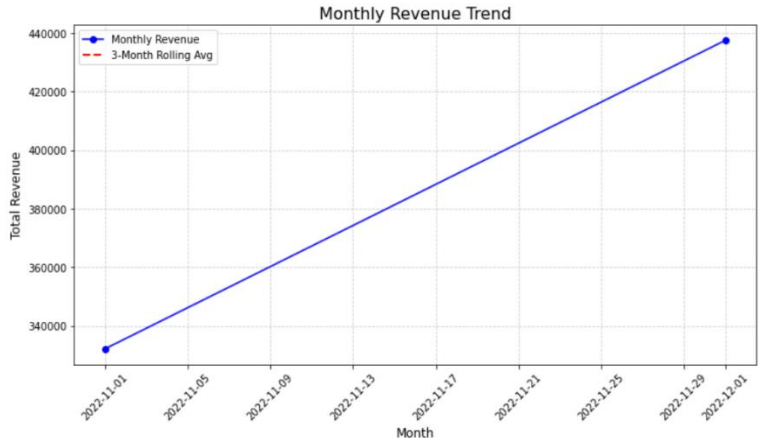
In [18]:
```
# --- Q7. Variance of Revenue & Quantity ---
var_revenue = dataset['Revenue'].var()
var_quantity = dataset['Quantity'].var()
print("7. Variance Revenue:", round(var_revenue, 2))
print("   Variance Quantity:", round(var_quantity, 2))
```

```
7. Variance Revenue: 5857001.11
   Variance Quantity: 46177.52
```

Q8. Was the revenue increasing or decreasing over the time?

```
In [19]: import matplotlib.pyplot as plt

# --- Q8. Revenue Trend (Increasing/Decreasing) ---
revenue_trend = dataset.groupby(dataset['Date'].dt.to_period('M'))['Revenue'].sum()

# Convert PeriodIndex to datetime for plotting
revenue_trend.index = revenue_trend.index.to_timestamp()

# Decide if increasing or decreasing
trend_direction = "Increasing" if revenue_trend.iloc[-1] > revenue_trend.iloc[0] else "Decreasing"
print("8. Revenue Trend Over Time:", trend_direction)

# --- Plot the Revenue Trend ---
plt.figure(figsize=(12,6))

# Original monthly revenue
plt.plot(revenue_trend.index, revenue_trend.values, marker='o', label="Monthly Revenue", color="blue")

# Rolling average (3-month window for smoothing)
rolling_avg = revenue_trend.rolling(window=3).mean()
plt.plot(rolling_avg.index, rolling_avg.values, color="red", linewidth=2, linestyle="--", label="3-Month Rolling Avg")

plt.title("Monthly Revenue Trend", fontsize=16)
plt.xlabel("Month", fontsize=12)
plt.ylabel("Total Revenue", fontsize=12)
plt.xticks(rotation=45)
plt.grid(True, linestyle="--", alpha=0.6)
plt.legend()

plt.show()
```

```
8. Revenue Trend Over Time: Increasing
```



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

```
In [20]: # --- Q9. Average Quantity & Revenue per Product ---
avg_stats = dataset.groupby('Product').agg(
    avg_quantity=('Quantity','mean'),
    avg_revenue=('Revenue','mean')
).round(2)
print("9. Average Quantity & Revenue per Product:\n", avg_stats)
```

```
9. Average Quantity & Revenue per Product:
                    avg_quantity  avg_revenue
Product
Beverages                 699.66      2064.00
Burgers                   558.12      7250.00
Chicken Sandwiches        214.15      2204.60
Fries                     628.13      2464.21
Sides & Other             200.40      1000.00
```

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

```
n [21]: # --- Q10. Total Number of Orders ---
total_orders = dataset['OrderId'].nunique()
print("10. Total Orders:", total_orders)
```

```
10. Total Orders: 254
```

CONCLUSION

- Credit Card payments dominate sales transactions.
- Beverages are most sold by quantity, while Burgers generate the most revenue
- Lisbon city and Manager Joao Silva outperform others in revenue contribution.
- Revenue steadily increases, with a strong peak in November & December (holiday effect).

RECOMMENDATIONS

1. Expand credit card payment support to improve customer convenience.
2. Promote Burgers more aggressively since it yields higher revenue.
3. Replicate strategies used in Lisbon to boost other cities.
4. Reward and support high-performing managers like Joao Silva.

HERE IS A LINK TO THE DATASET:

https://colab.research.google.com/drive/1tiT1-PiGXzYZ1AndCL1G5WoMkEm_cmp4?usp=sharing