**Data analytics with cognos Project**

**Phase 3 Submission**

**Product sales analysis**

**Innovation Phase**

**Team members:**

**Dhivyadharshini S B –** [**dhivyadharshu07@gmail.com**](mailto:dhivyadharshu07@gmail.com)

**Dharini A –** [**dhariniathithakumar@gmail.com**](mailto:dhariniathithakumar@gmail.com)

**Dileep-** [**dilleep18@gmail.com**](mailto:dilleep18@gmail.com)

**Dhavasri –** [**indradhavasri212@gmail.com**](mailto:indradhavasri212@gmail.com)

This includes the Development part 1

**Data Processing:**

Data processing is a critical step in product sales analysis. It involves cleaning, transforming, and preparing your data so that it's in a suitable format for analysis. Here are some key steps and techniques in data processing for product sales analysis:

**1. Data Cleaning:**

- Handling Missing Values: Identify and handle missing data, either by filling in missing values or removing incomplete records.

- Removing Duplicates: Check for and remove duplicate records from your dataset.

- Data Validation: Ensure data integrity by checking for outliers, anomalies, or incorrect entries.

**2. Data Transformation:**

**-** Data Normalization/Scaling: If necessary, scale your data to ensure that different features are on a similar scale. This is important when using certain algorithms like k-means clustering.

- Data Encoding: Convert categorical variables into numerical format, typically using techniques like one-hot encoding or label encoding.

- Date and Time Processing: Extract relevant information from date and time columns, such as day of the week, month, or year.

- Aggregation: Summarize or aggregate data to a level that is suitable for analysis. For sales data, this might involve aggregating daily sales into monthly or yearly totals.

- Feature Engineering: Create new features that can provide additional insights, such as calculating profit margins or customer lifetime value.

**3. Data Exploration:**

- Data Visualization: Create various charts and graphs to visualize the data, helping to identify patterns, trends, and outliers.

- Descriptive Statistics: Calculate summary statistics using methods like `data.describe()` as mentioned in the previous response. This can give you an initial understanding of your data's distribution.

**4. Data Filtering and Subsetting:**

- Depending on your analysis goals, you may need to filter or subset the data to focus on specific products, time periods, customer segments, or other relevantcriteria.

**5. Time Series Analysis:**

- If you're dealing with time series data, consider time series-specific techniques like forecasting, moving averages, and seasonality decomposition**.**

**6. Customer Segmentation:**

- Segment customers based on their purchase behavior or demographics. Customer segmentation can provide valuable insights for marketing and sales strategies.

**7. Statistical Analysis:**

**-** Conduct statistical tests to answer specific questions, such as whether there is a significant difference in sales before and after a promotional campaign.

**8. Machine Learning:**

**-** Implement machine learning models for predictive analysis, such as sales forecasting or product recommendations**.**

**9. Report Generation:**

**-** Summarize your findings and insights in a clear and understandable format, often through the creation of reports or dashboards using tools like Tableau, Power BI, or custom Python/R scripts**.**

**10. Data Storage and Backup:**

**-** Ensure that you have a well-organized data storage and backup strategy to preserve data integrity and availability.

# DATA CLEANING

1. **Understanding the Data:**

Begin by understanding the dataset's structure, column names, data types, and identifying any missing or duplicate values. This step helps in formulating a strategy for cleaning the data effectively.

# Handling Missing Values:

Decide how to handle missing values in the dataset. Options include removing rows with missing values or filling missing values with appropriate measures like the mean or median.

# Handling Duplicates:

Identify and remove duplicate rows to ensure that the analysis is performed on unique data points, preventing skewed results.

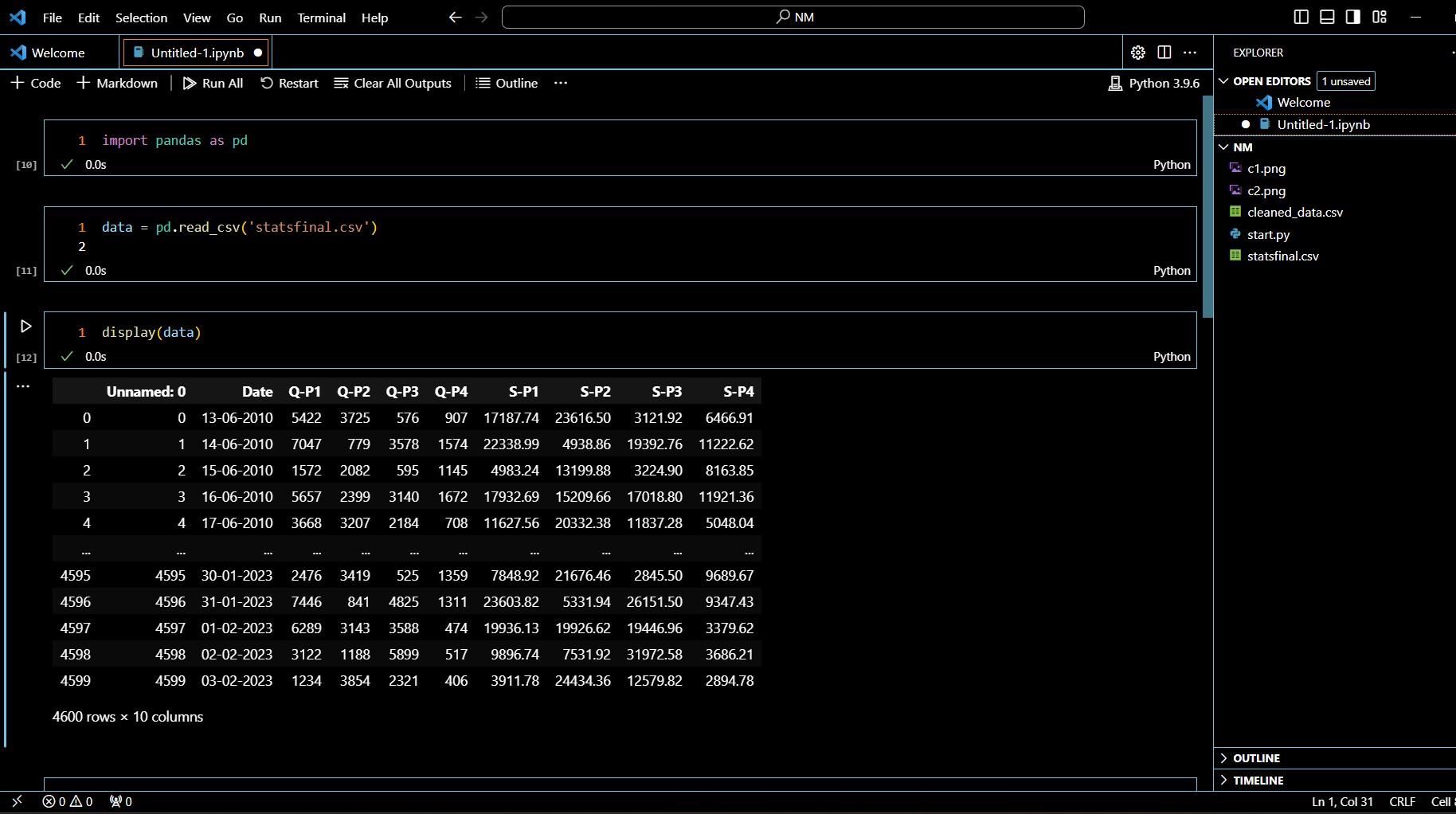
# Saving Cleaned Data:

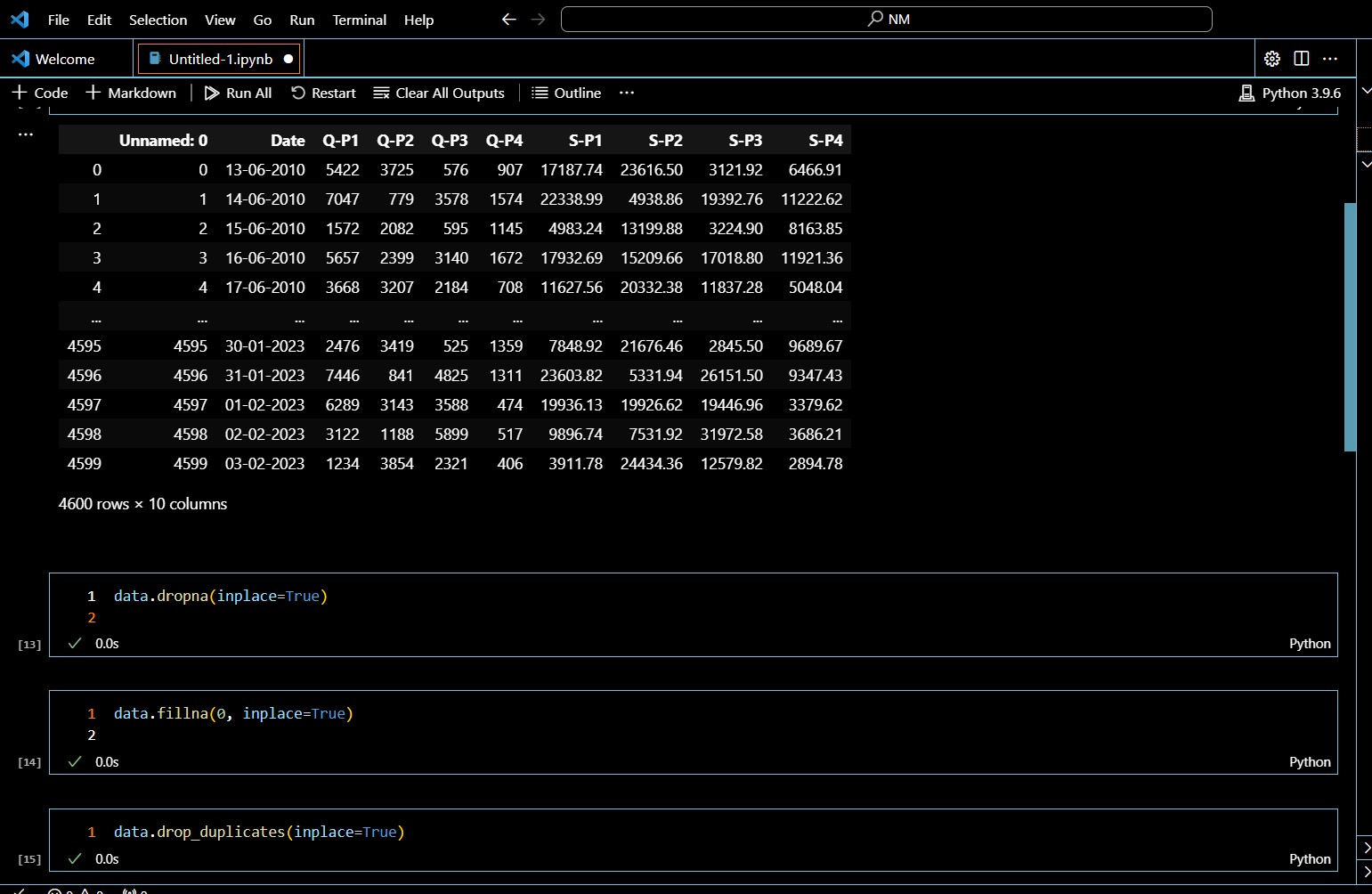
After completing the cleaning and preprocessing steps, save the cleaned dataset. This ensures that the cleaned data is available for further analysis and modeling without the need to repeat the cleaning process every time the analysis is performed.

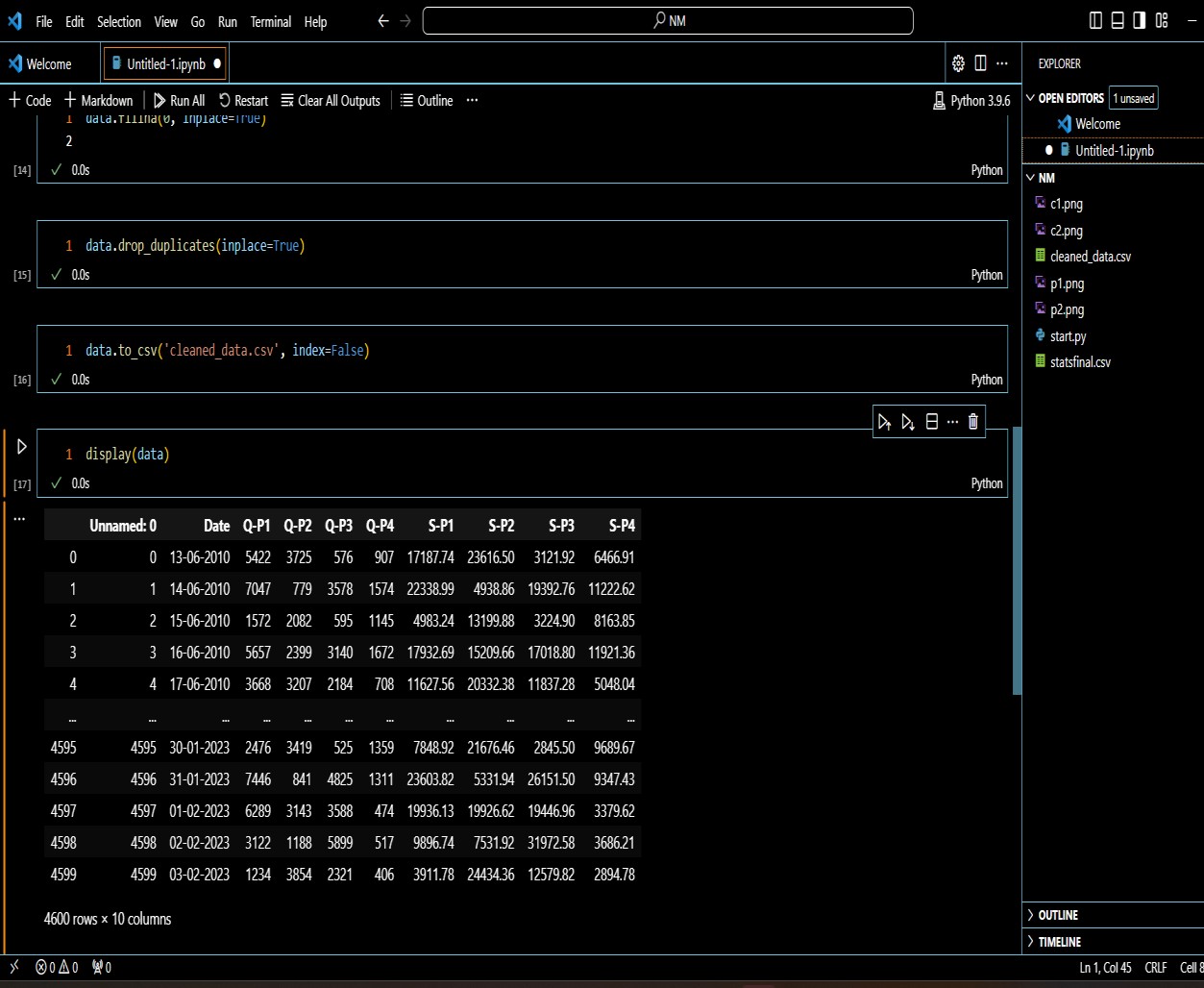
About Dataset

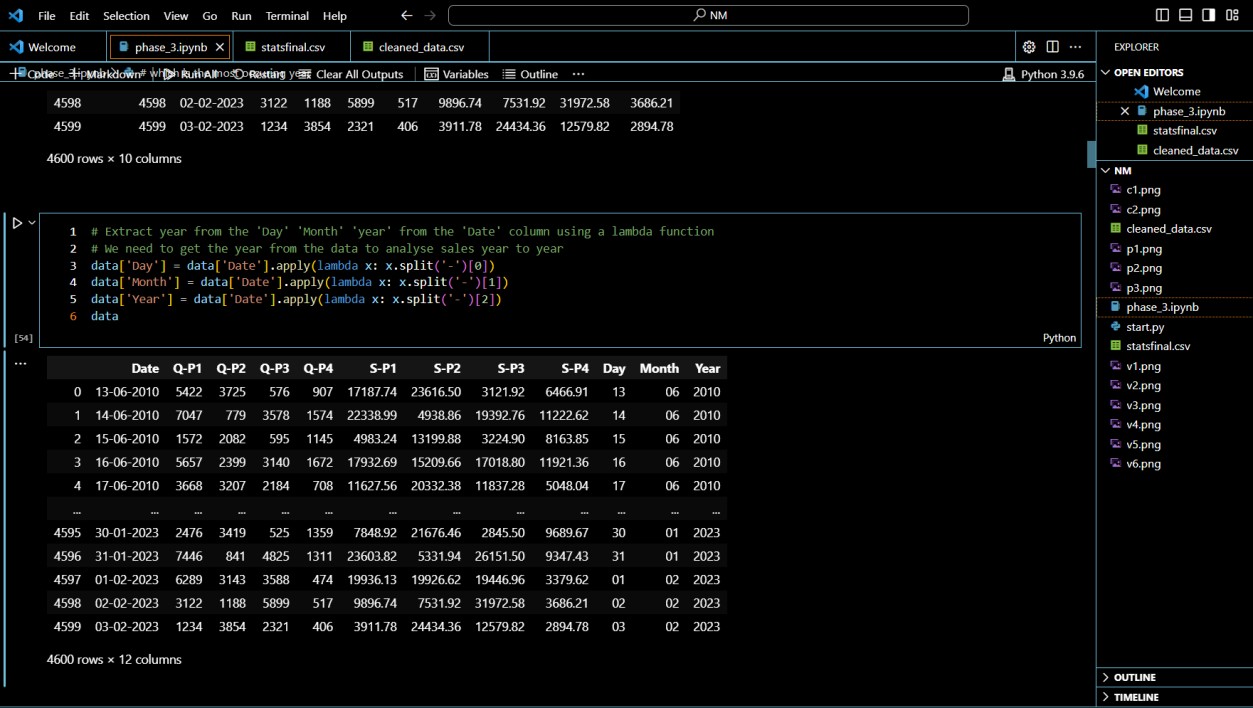
* + Q1- Total unit sales of product 1
  + Q2- Total unit sales of product 2
  + Q3- Total unit sales of product 3
  + Q4- Total unit sales of product 4
  + S1- Total revenue from product 1
  + S2- Total revenue from product 2
  + S3- Total revenue from product 3
  + S4- Total revenue from product 4

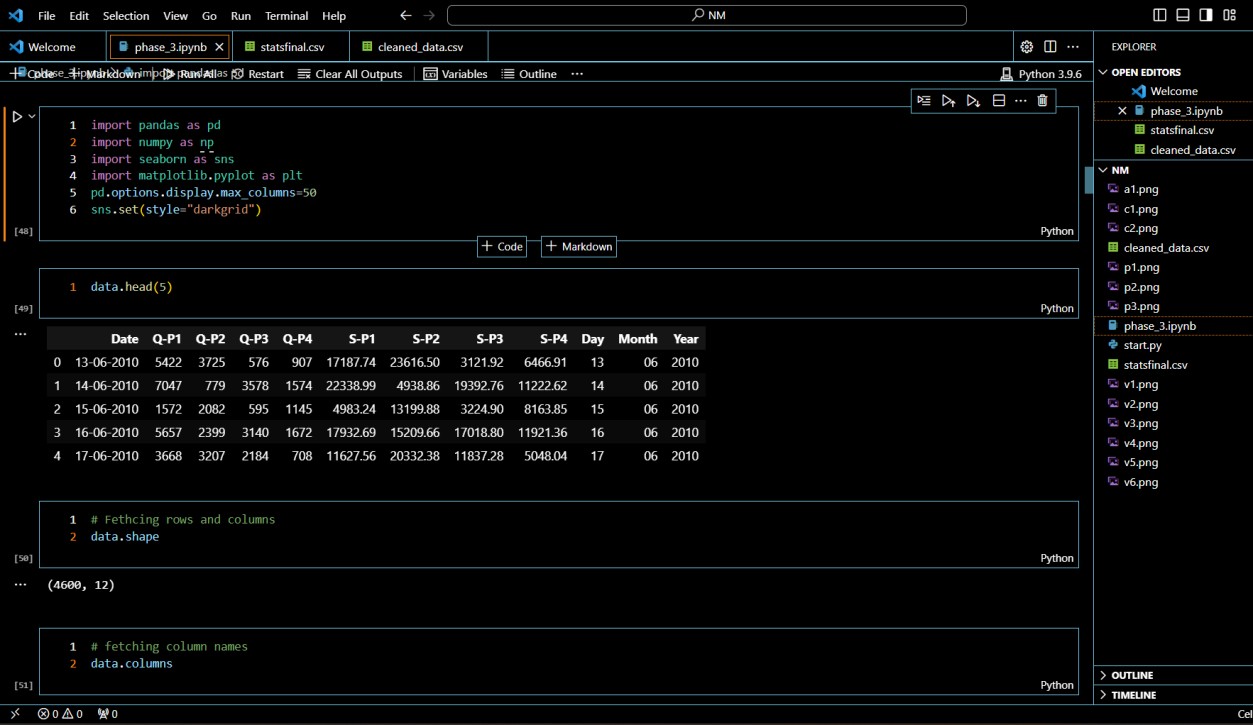
Data Cleaning

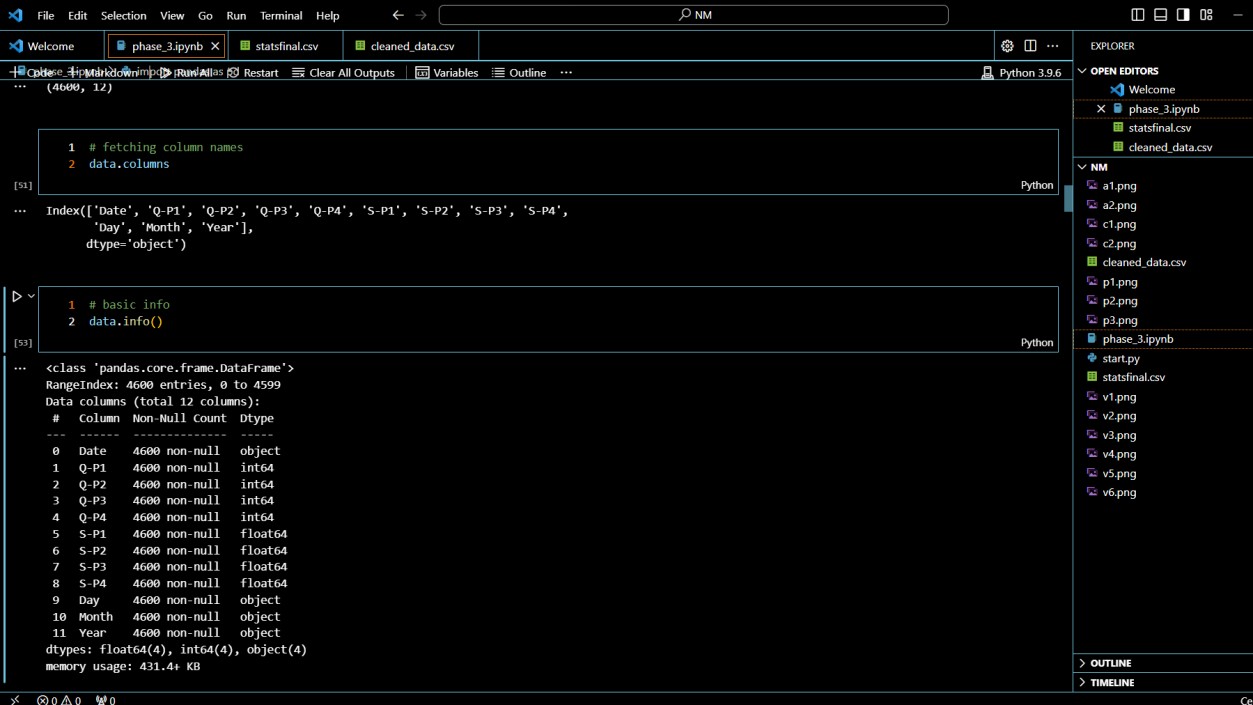


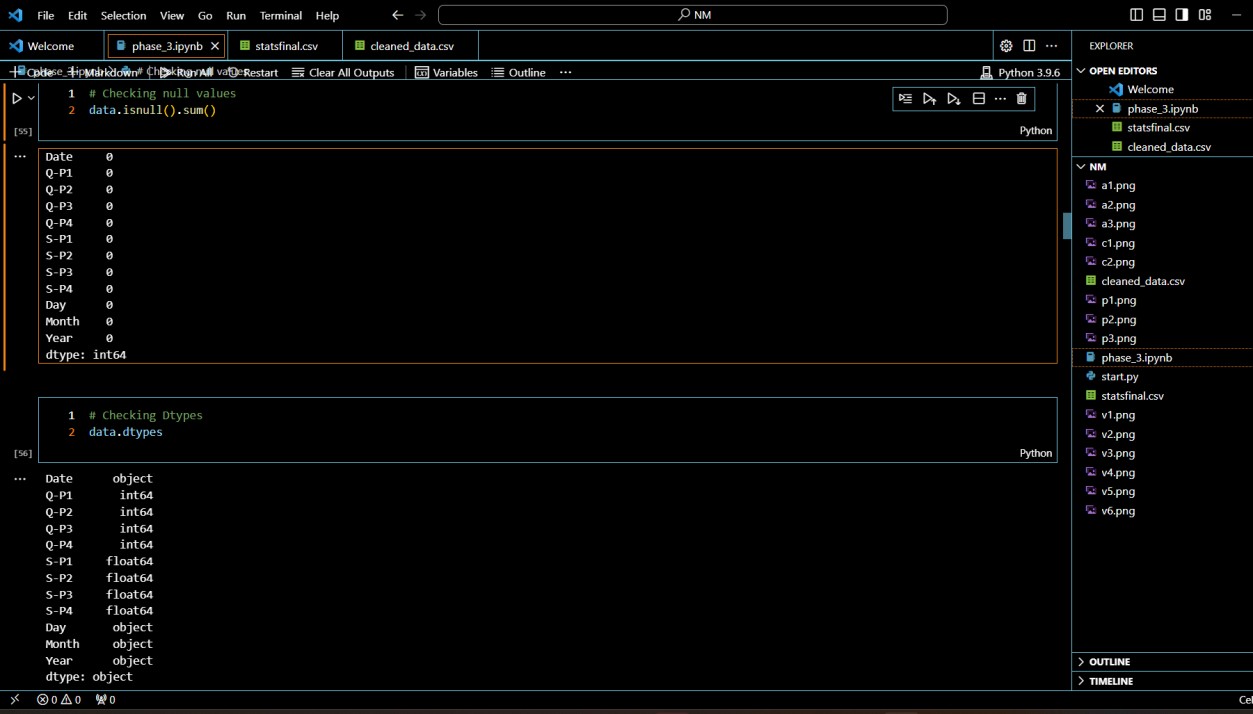




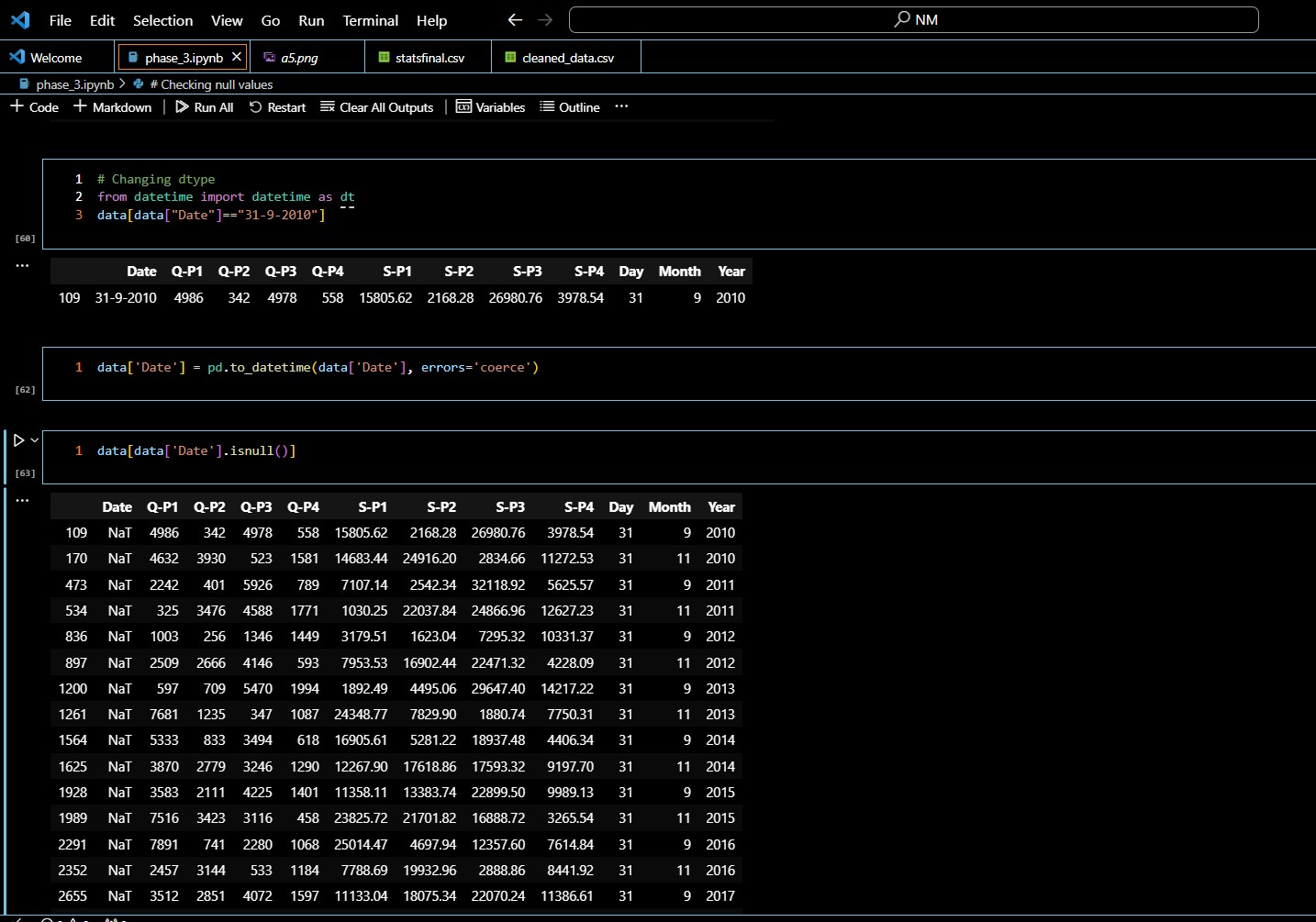


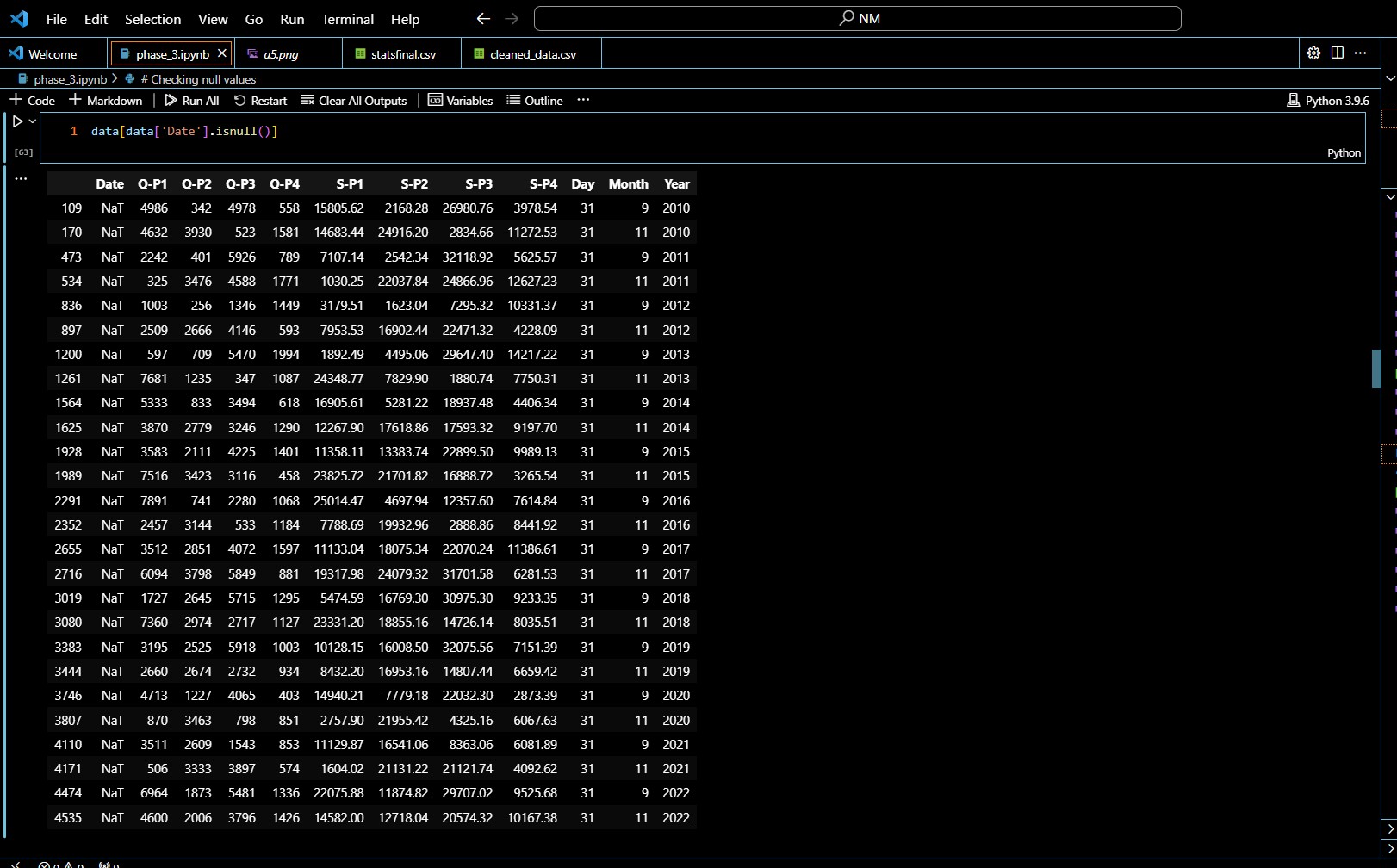




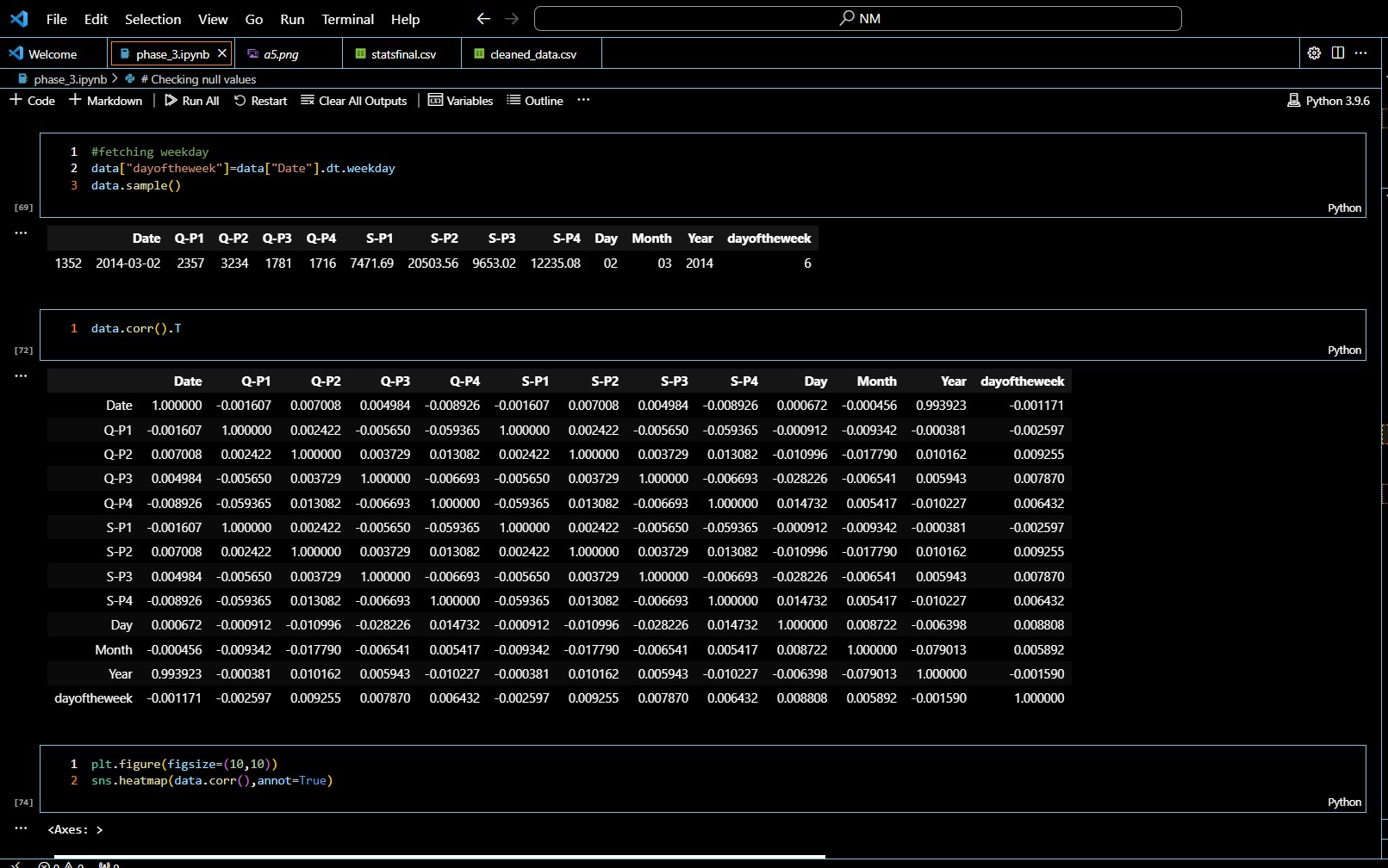


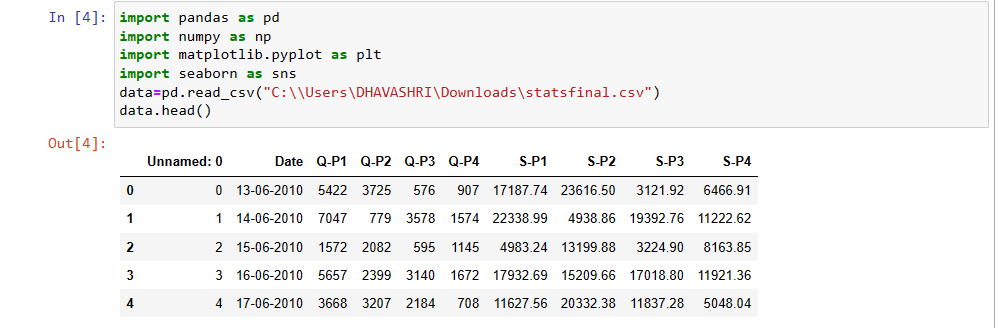












**1. Data Exploration**:

- Check the data's structure: `data.info()`

- Summary statistics: `data.describe()`

- Check for missing values: `data.isnull().sum()`

**2. Data Visualization:**

- Plot histograms or bar charts to visualize distributions.

- Create scatter plots for relationships between variables.

- Use Seaborn for more advanced visualization.

**3. Data Preprocessing**:

- Handle missing values (e.g., impute or remove rows/columns).

- Convert data types if necessary.

- Encode categorical variables.

**4. Analysis:**

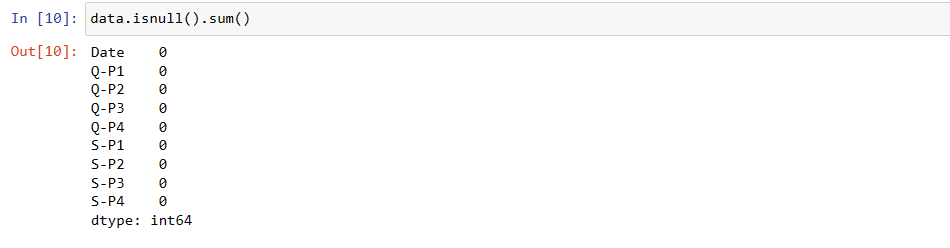
- Calculate total sales, profit, or any other relevant metrics.

- Perform time series analysis if your dataset includes timestamps.

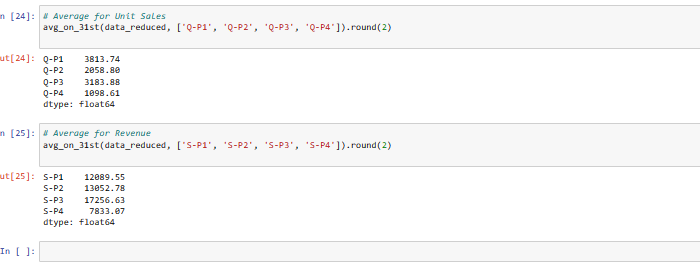
- Conduct hypothesis tests or statistical analysis if needed.

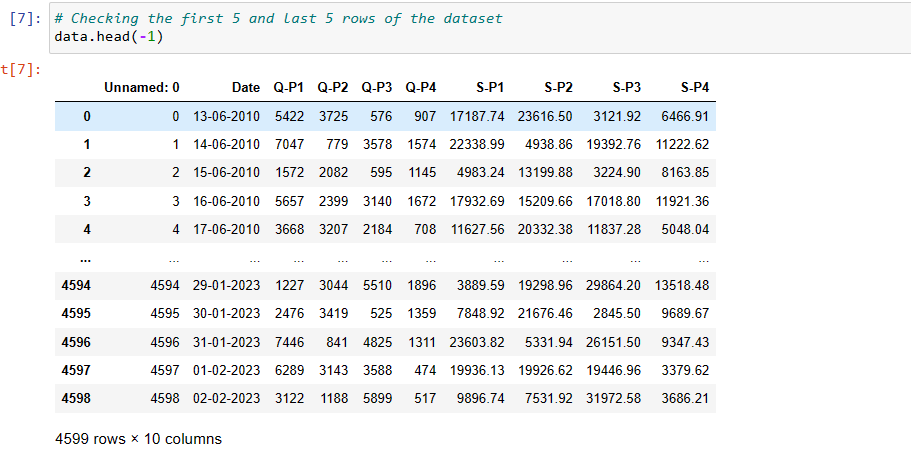
**5. Machine Learning (if applicable):**

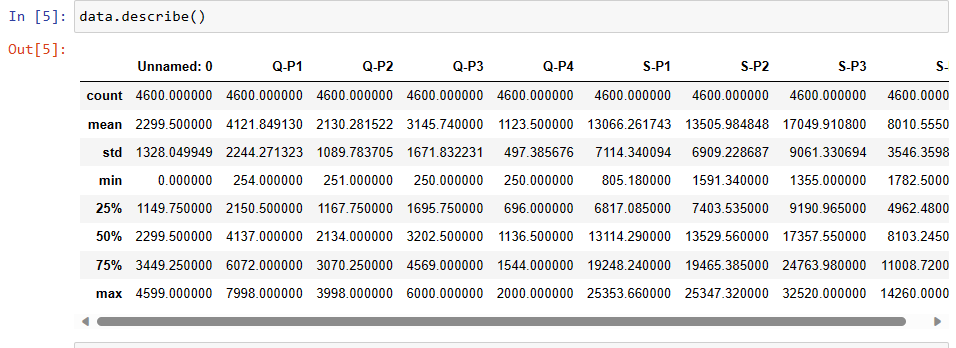
- Build predictive models for sales forecasting.



1. **data.isnull()**: This part of the code generates a DataFrame of the same shape as **data**, where each element is **True** if the corresponding element in **data** is null (missing), and **False** otherwise.
2. **.sum()**: This part of the code calculates the sum for each column in the DataFrame obtained in step 1. Since **True** is treated as 1 and **False** as 0 when you sum a boolean DataFrame, this effectively counts the number of missing values in each column.







1. Count: It counts the number of non-null (non-missing) values in each numeric column. This can give you an idea of whether there are any missing values in your dataset.

2. Mean: It calculates the arithmetic mean (average) for each numeric column, which gives you a measure of central tendency.

3. Standard Deviation (std): It measures the dispersion or spread of the data. A higher standard deviation indicates more variation in the data.

4. Minimum: It gives the minimum (lowest) value in each column, which can be useful for understanding the range of values.

5. 25th Percentile (25%): This represents the value below which 25% of the data falls. It's also called the first quartile.

6. 50th Percentile (50%): This is the median or the value below which 50% of the data falls. It's the second quartile.

7. 75th Percentile (75%): This represents the value below which 75%<class 'pandas.core.frame.DataFrame'>: This line tells you that you're working with a pandas DataFrame.

8. RangeIndex: 1000 entries, 0 to 999: It provides information about the index, including the number of entries and the range.

9. Data columns (total 5 columns): This line tells you that there are 5 columns in the DataFrame.

10. Column1 1000 non-null int64: This is information about the first column. It tells you the name of the column, the number of non-null (non-missing) values, and the data type (in this case, int64).

11. Column2 1000 non-null float64: Similar to the first column, this is information about the second column, which is a float.

12. Column3 1000 non-null object: Information about the third column, which is of type object (typically strings).

13. Column4 1000 non-null int64: Information about the fourth column, another integer column.

14. Column5 1000 non-null datetime64[ns]: Information about the fifth column, which is of type datetime.

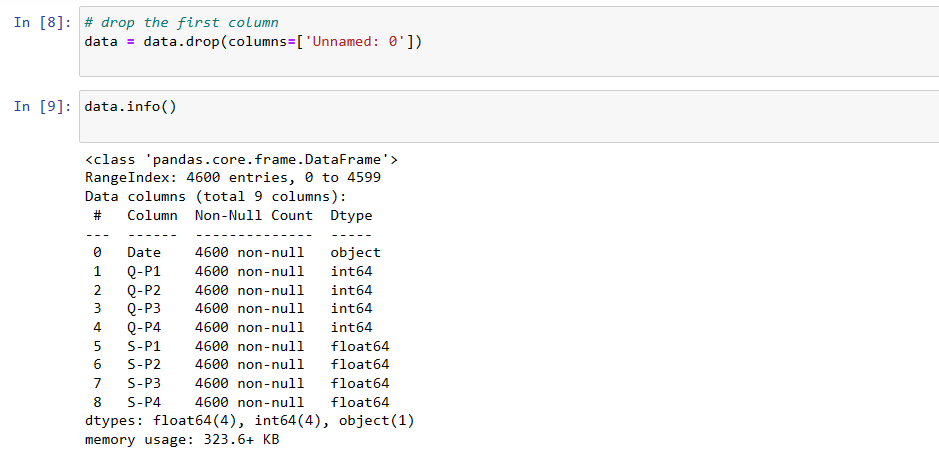
15. dtypes: This section lists the data types of all columns.

16. memory usage: It shows the memory usage of the DataFrame.

17. This information is very useful when you're cleaning, transforming, or analyzing a dataset because it gives you an initial understanding of the data's structure and helps identify potential issues with missing values or incorrect data types.

19. of the data falls. It's the third quartile.

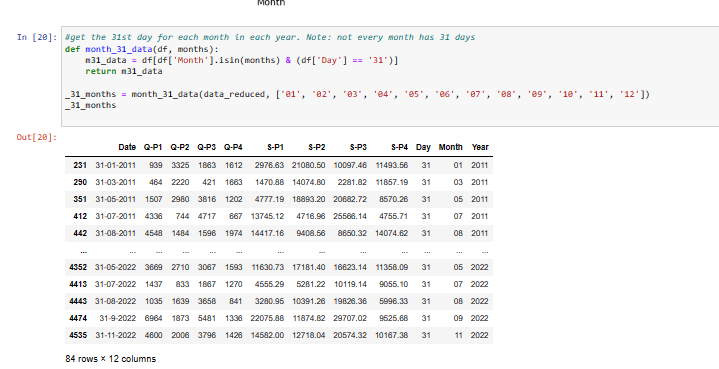
20. Maximum: It gives the maximum (highest) value in each column, which helps you understand the range of values.

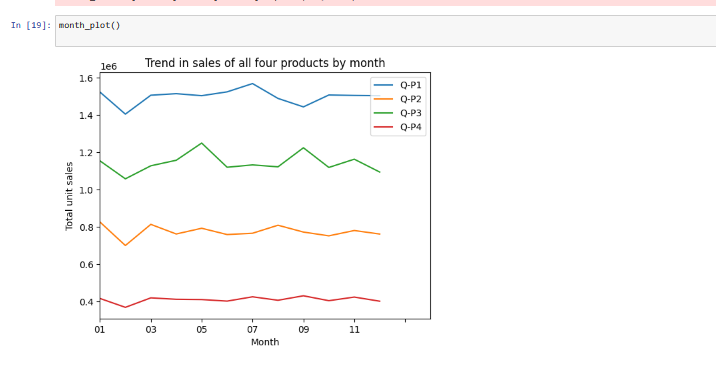


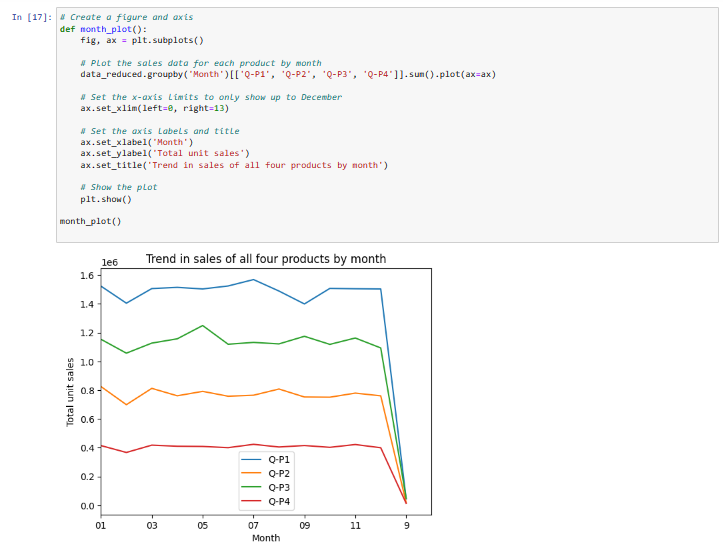
* **<class 'pandas.core.frame.DataFrame'>**: This line tells you that you're working with a pandas DataFrame.
* **RangeIndex: 1000 entries, 0 to 999**: It provides information about the index, including the number of entries and the range.
* **Data columns (total 5 columns)**: This line tells you that there are 5 columns in the DataFrame.
* **Column1 1000 non-null int64**: This is information about the first column. It tells you the name of the column, the number of non-null (non-missing) values, and the data type (in this case, int64).
* **Column2 1000 non-null float64**: Similar to the first column, this is information about the second column, which is a float.
* **Column3 1000 non-null object**: Information about the third column, which is of type object (typically strings).
* **Column4 1000 non-null int64**: Information about the fourth column, another integer column.
* **Column5 1000 non-null datetime64[ns]**: Information about the fifth column, which is of type datetime.
* **dtypes**: This section lists the data types of all columns.
* **memory usage**: It shows the memory usage of the DataFrame.

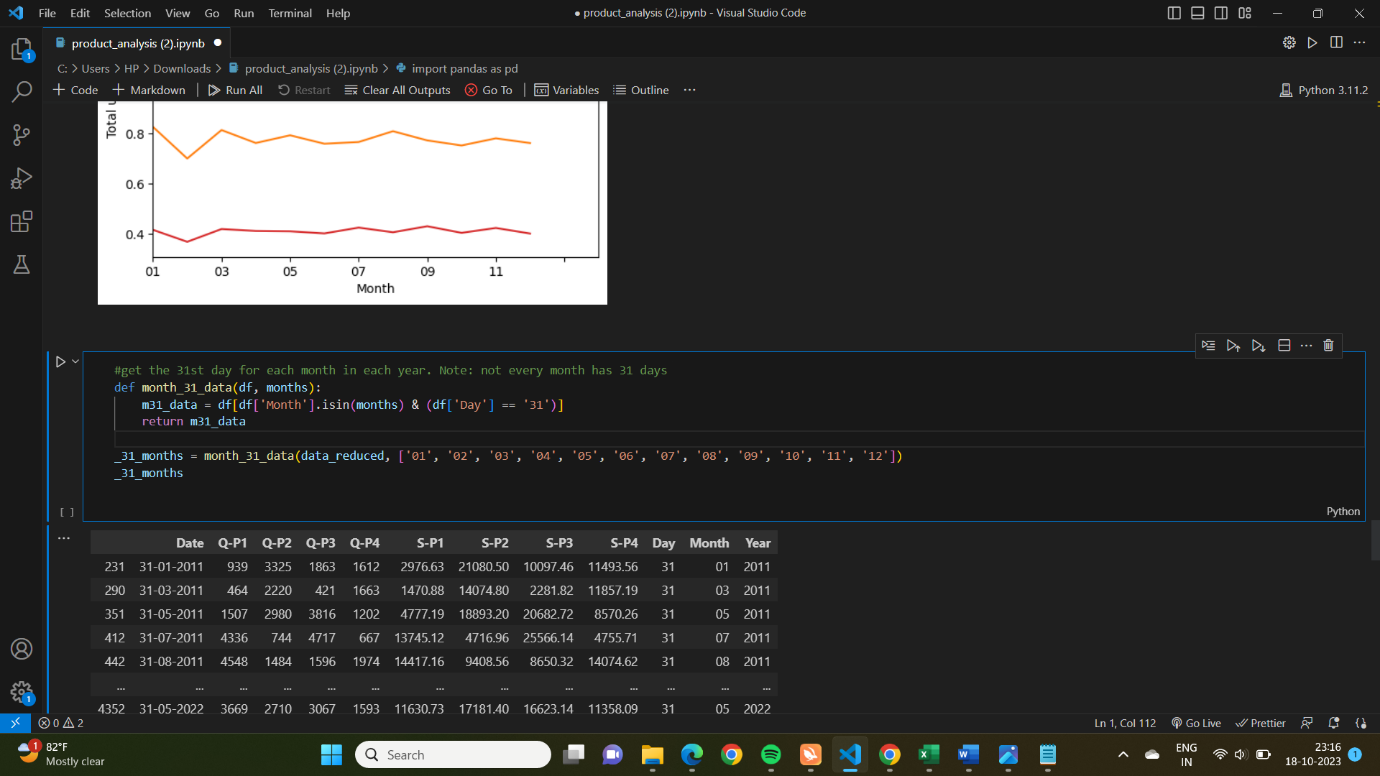
This information is very useful when you're cleaning, transforming, or analyzing a dataset because it gives you an initial understanding of the data's structure and helps identify potential issues with missing values or incorrect data types.

Top of Form









1. **df** is the DataFrame containing the dataset.
2. **months** is a list of months (in the format '01', '02', ..., '12') for which you want to retrieve data for the 31st day.
3. The function first filters the DataFrame to include rows where the 'Month' column matches one of the specified months and the 'Day' column is '31'.

