Name : Rahul Rajput

Registration Number - 12107183

Roll Number - RK21UTB49 Set - 1 Question Number - 1

Q1 List the Pros & Cons of Joins & Subquery?

Aspect	Joins	Subqueries
Pros		
Performance	often more efficient for multiple tables.	may be less efficient for large dataset.
Readability	Can make queries more readable.	can simplfy complex queries.
Flexibility	Can Combine data in Various ways	Provides fine-grained Control
Scalability	Efficient for Complex large datasets	Simpler for Single - table operations
Portability	-	More portable across database Systems

Name : Rahul Rajput Registration Number-12107183

Roll Number- RK21UTB49 Set -1 Question Number - 1

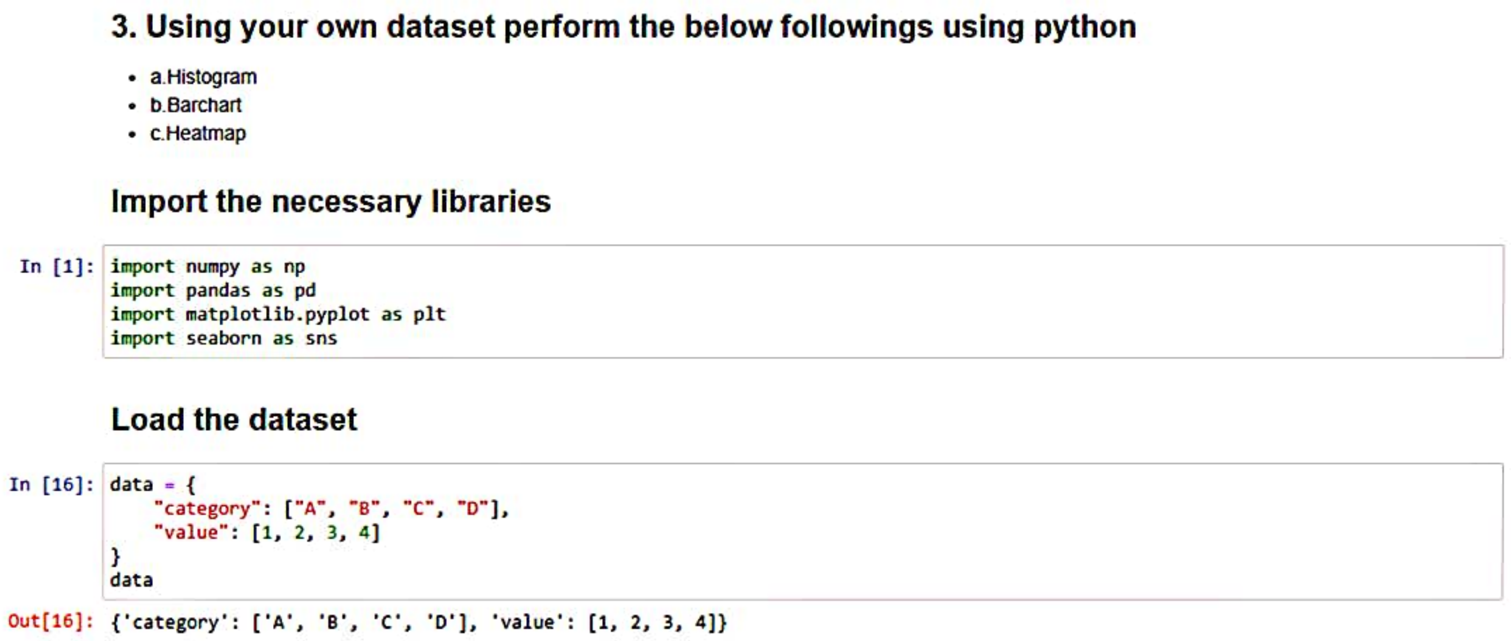
Aspect	Joins	Subqueries
Cons		
Complexity	Can become complex with many tables	Complex Subqueries Can be Challenging
Redundancy	May result in duplicated data	May lead to redundant or inefficient queries
Maintnance	Updates may require Changes to many queries	Easier to maintain and Understand.
Inpficient	-	May cause performance issues with multiple subqueries
Limited Capabilities	Suitable for Complex data retrieval tasks	May not handle all Complex operations

Name : Rahul Rajput Registration Number- 12107183

Roll Number- RK21UTB49 Set-1 Question Number- 2

Ans2

```
CREATE TABLE StudentMarks(
 StudentName VARCHAR(255),
 ClassName VARCHAR(255),
 Student Marks INT
 );

INSERT INTO StudentMarks(StudentName, ClassName,
Student Marks) VALUES
('Anni' ,'Class 7' , 92),
('Kaley', 'Class 9', 80),
(' Tom', 'Class 10', 85),
(' Johny', 'Class 4', 76),
(' Jimmy', 'Class 3', 95);

SELECT StudentName, DENSE-RANK()OVER(ORDER BY
 StudentMarks DESC) AS Student Rank
 FROM StudentMarks;
```



3. Using your own dataset perform the below followings using python

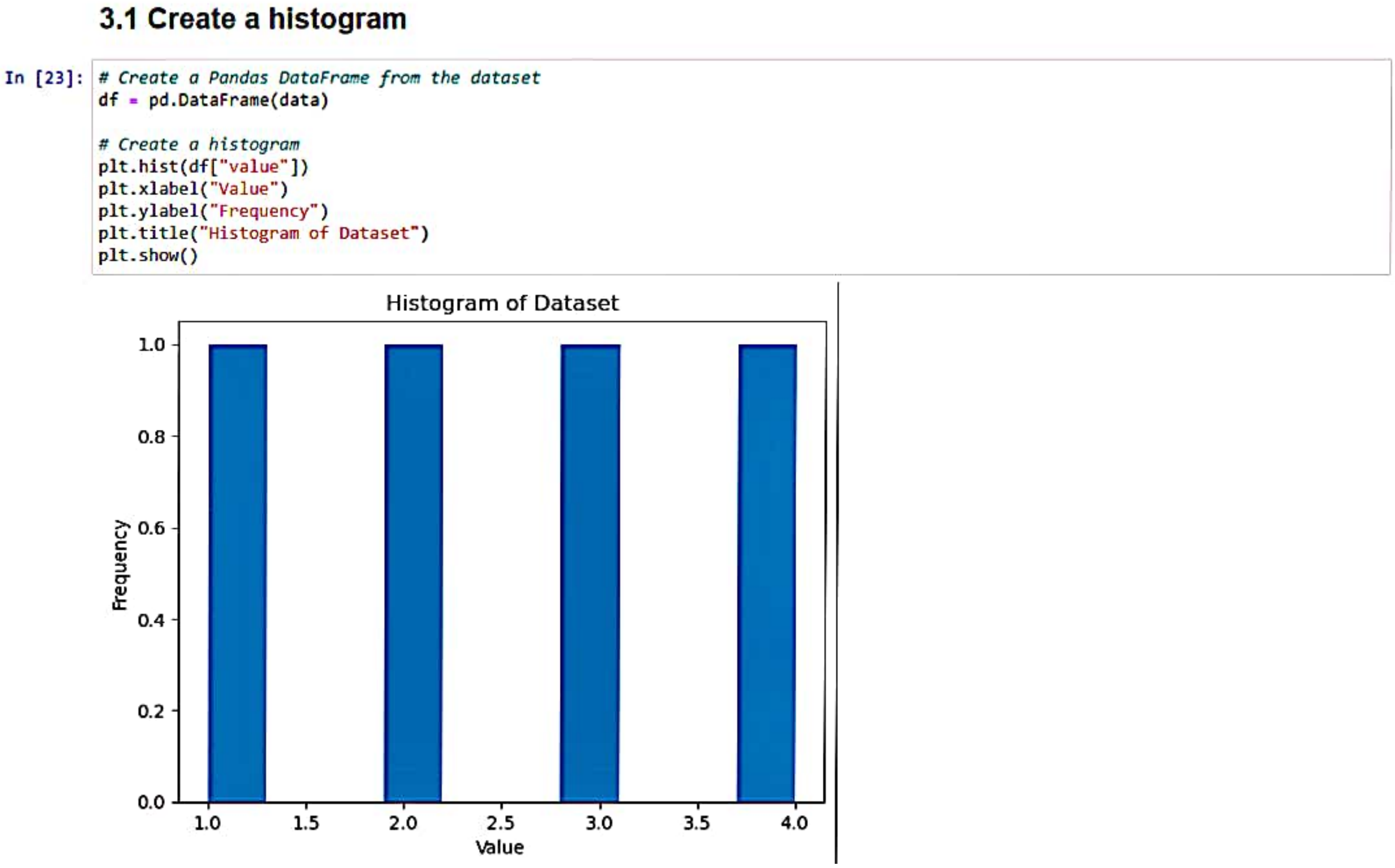
- a.Histogram
- b.Barchart
- c.Heatmap

Import the necessary libraries

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

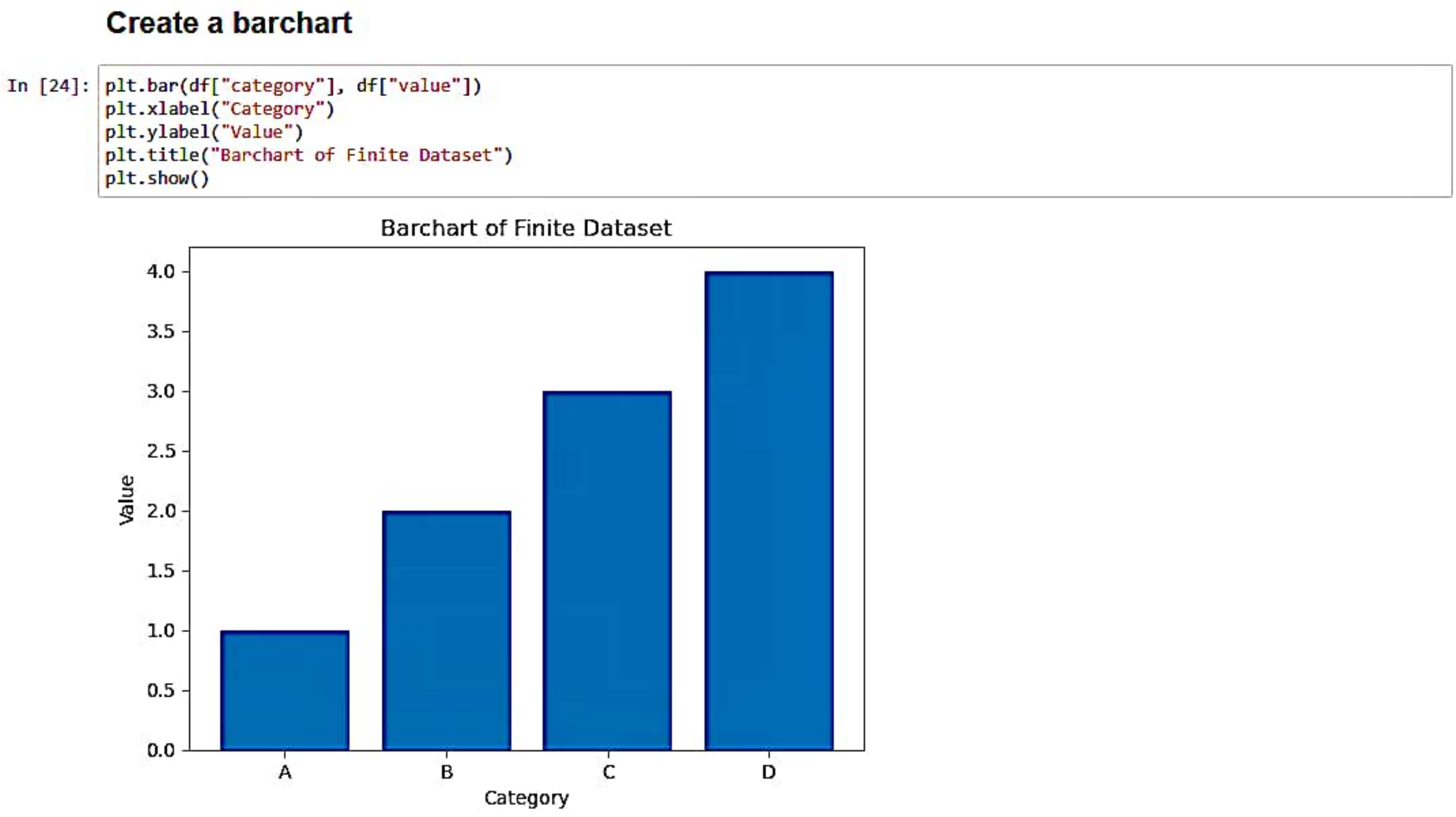
Load the dataset

```
In [16]: data = {
             "category": ["A", "B", "C", "D"],
             "value": [1, 2, 3, 4]
         }
         data

Out[16]: {'category': ['A', 'B', 'C', 'D'], 'value': [1, 2, 3, 4]}
```



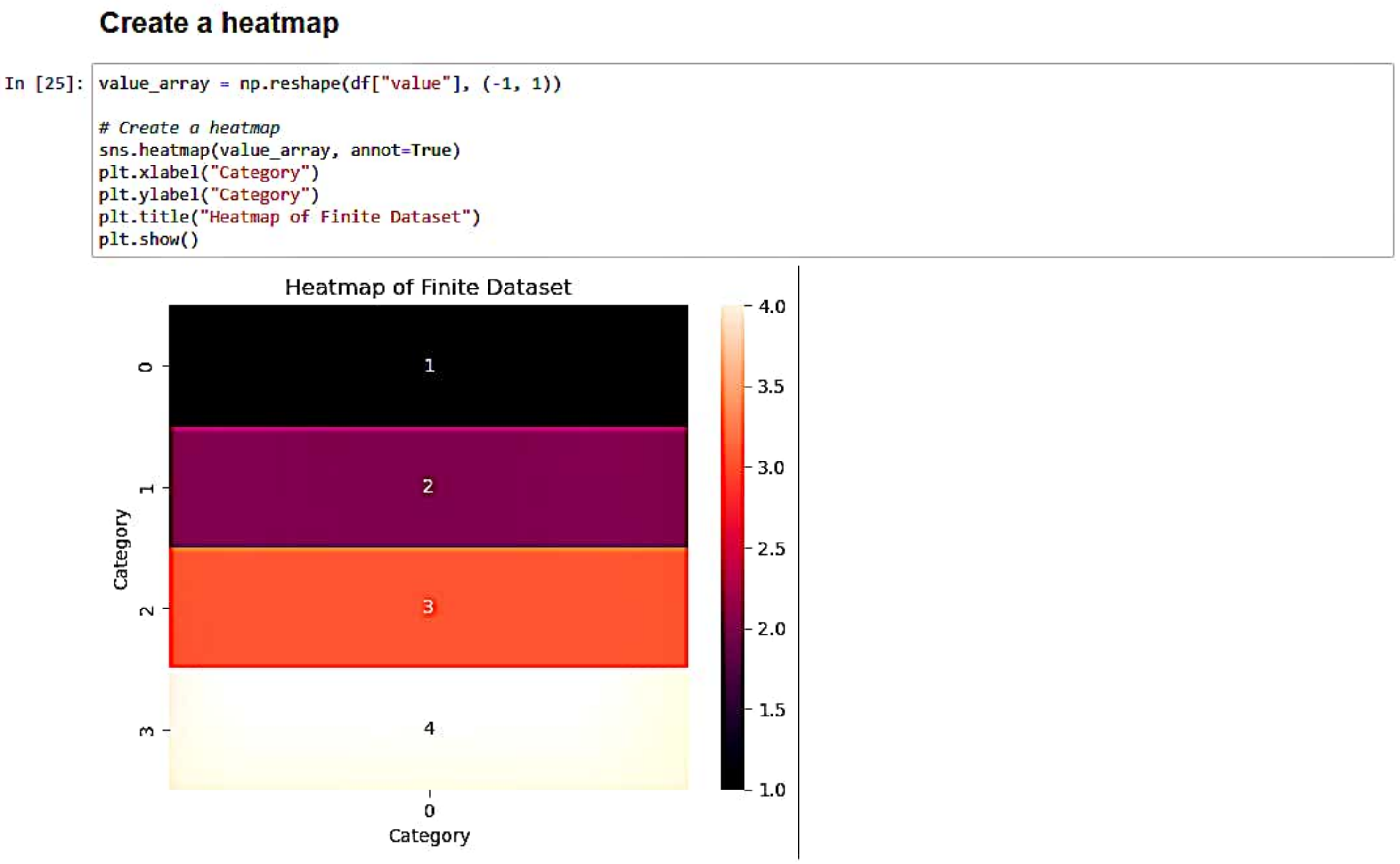
3.1 Create a histogram

```
In [23]: # Create a Pandas DataFrame from the dataset
         df = pd.DataFrame(data)

         # Create a histogram
         plt.hist(df["value"])
         plt.xlabel("Value")
         plt.ylabel("Frequency")
         plt.title("Histogram of Dataset")
         plt.show()
```



Create a barchart

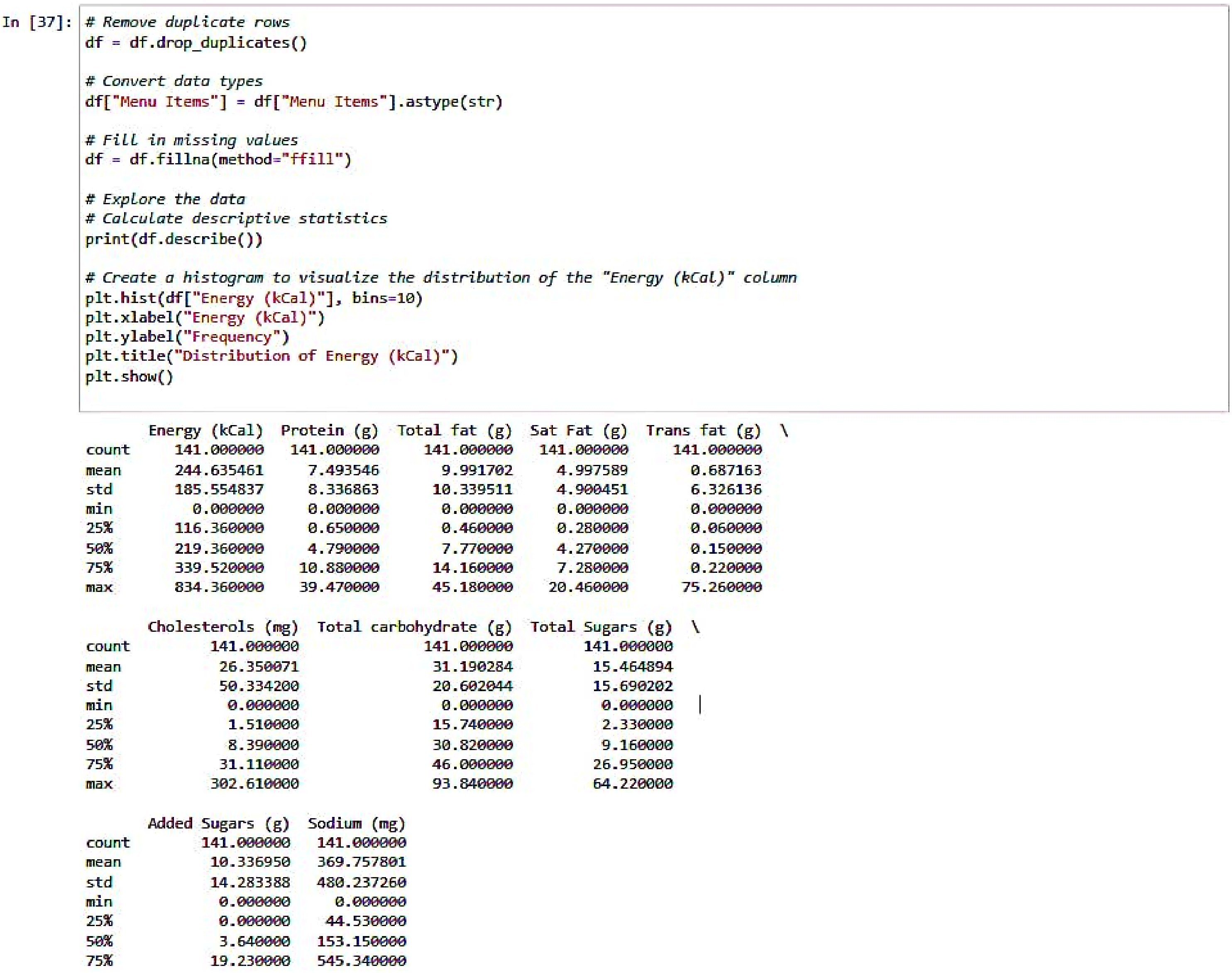
```
In [24]: plt.bar(df["category"], df["value"])
         plt.xlabel("Category")
         plt.ylabel("Value")
         plt.title("Barchart of Finite Dataset")
         plt.show()
```



Create a heatmap

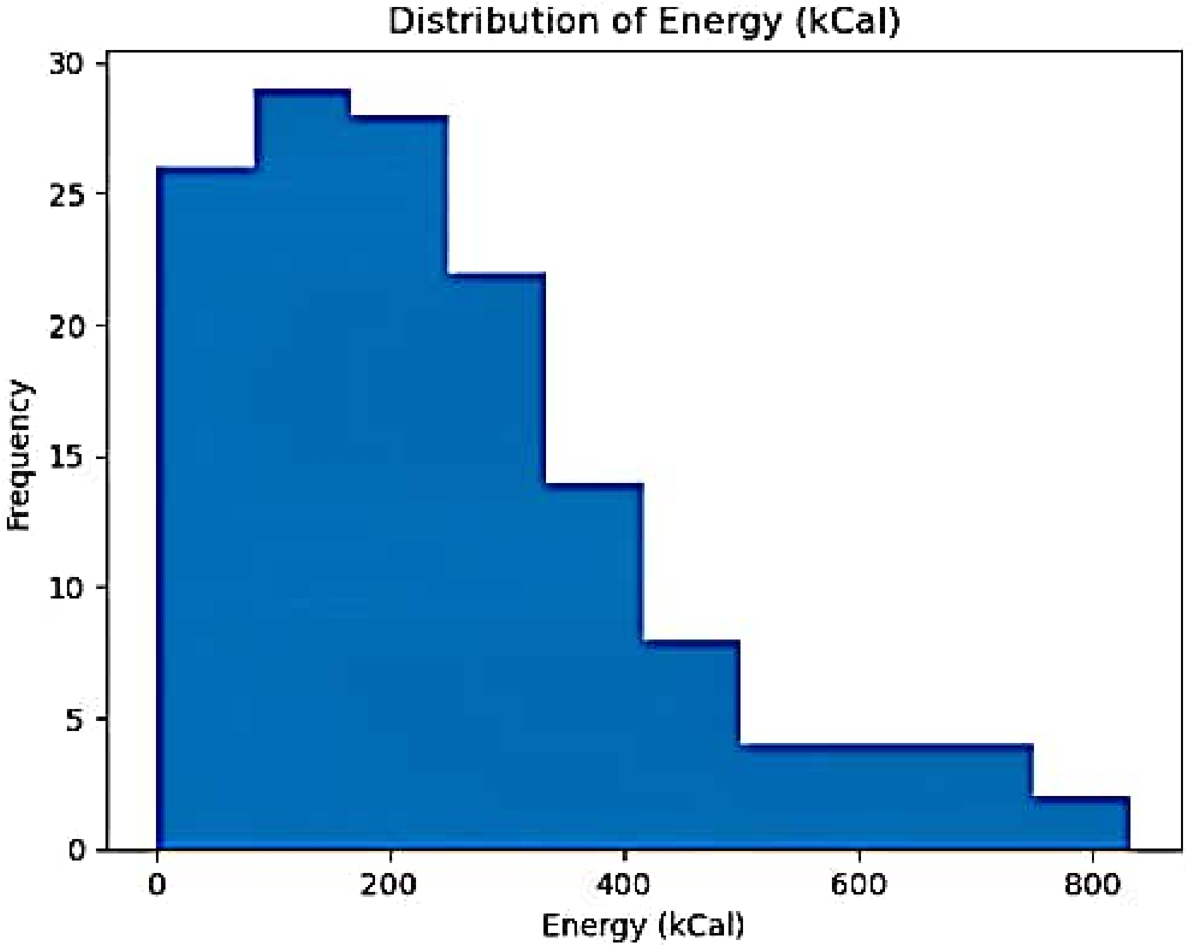
```
In [25]: value_array = np.reshape(df["value"], (-1, 1))

         # Create a heatmap
         sns.heatmap(value_array, annot=True)
         plt.xlabel("Category")
         plt.ylabel("Category")
         plt.title("Heatmap of Finite Dataset")
         plt.show()
```

4.2 Data Handling

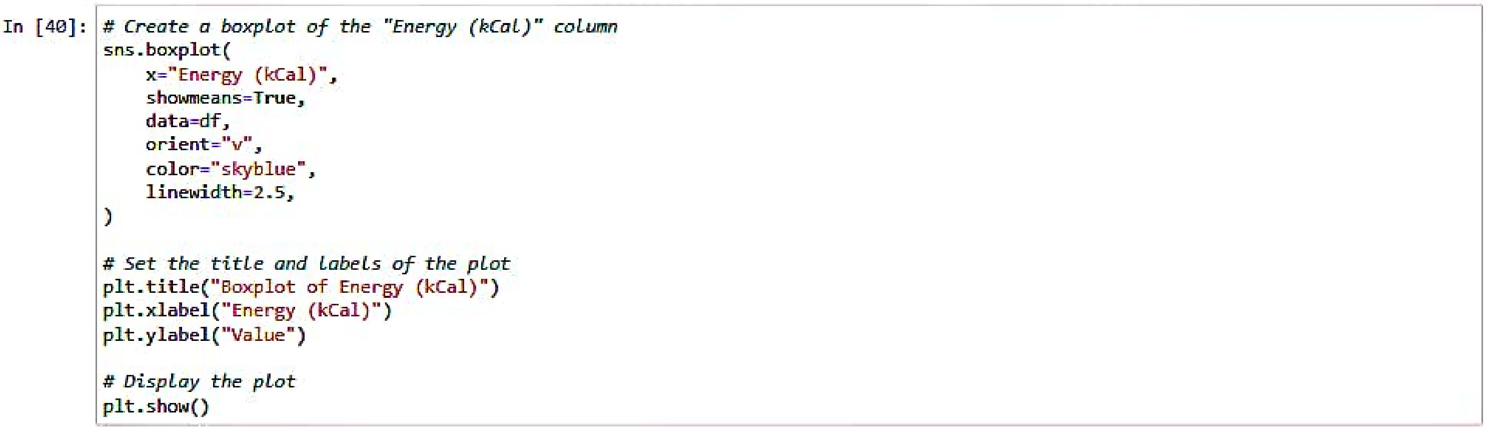


```
In [37]: # Remove duplicate rows
         df = df.drop_duplicates()

         # Convert data types
         df["Menu Items"] = df["Menu Items"].astype(str)

         # Fill in missing values
         df = df.fillna(method="ffill")

         # Explore the data
         # Calculate descriptive statistics
         print(df.describe())

         # Create a histogram to visualize the distribution of the "Energy (kCal)" column
         plt.hist(df["Energy (kCal)"], bins=10)
         plt.xlabel("Energy (kCal)")
         plt.ylabel("Frequency")
         plt.title("Distribution of Energy (kCal)")
         plt.show()
```

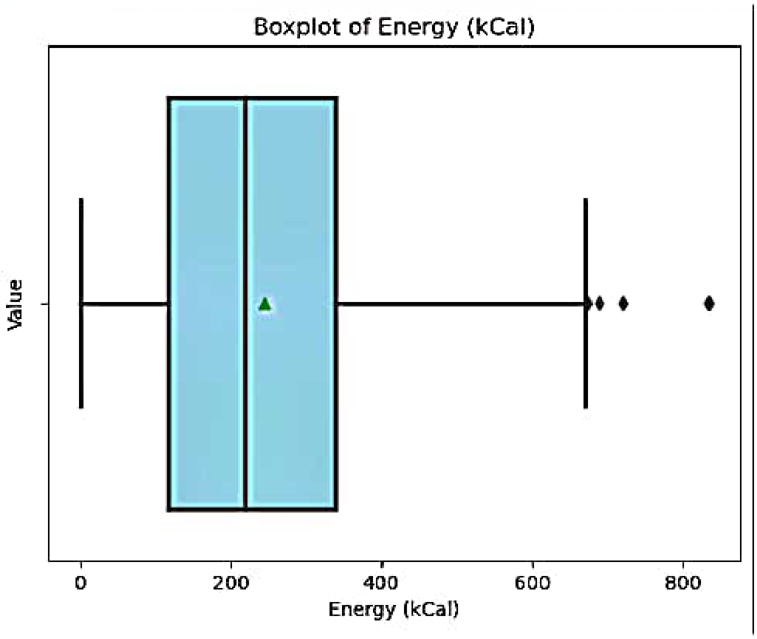
```
       Energy (kCal)  Protein (g)  Total fat (g)  Sat Fat (g)  Trans fat (g)  \
count     141.000000   141.000000     141.000000   141.000000     141.000000
mean      244.635461     7.493546       9.991702     4.997589       0.687163
std       185.554837     8.336863      10.339511     4.900451       6.326136
min         0.000000     0.000000       0.000000     0.000000       0.000000
25%       116.360000     0.650000       0.460000     0.280000       0.060000
50%       219.360000     4.790000       7.770000     4.270000       0.150000
75%       339.520000    10.880000      14.160000     7.280000       0.220000
max       834.360000    39.470000      45.180000    20.460000      75.260000

       Cholesterols (mg)  Total carbohydrate (g)  Total Sugars (g)  \
count         141.000000              141.000000        141.000000
mean           26.350071               31.190284         15.464894
std            50.334200               20.602044         15.690202
min             0.000000                0.000000          0.000000
25%             1.510000               15.740000          2.330000
50%             8.390000               30.820000          9.160000
75%            31.110000               46.000000         26.950000
max           302.610000               93.840000         64.220000

       Added Sugars (g)  Sodium (mg)
count        141.000000   141.000000
mean          10.336950   369.757801
std           14.283388   480.237260
min            0.000000     0.000000
25%            0.000000    44.530000
50%            3.640000   153.150000
75%           19.230000   545.340000
```


Distribution of Energy (kCal)
Frequency
0
5
10
15
20
25
30
0
200
400
600
800
Energy (kCal)

5.Perform boxplot for detecting outlier in python using your own dataset?



```
In [40]: # Create a boxplot of the "Energy (kCal)" column
         sns.boxplot(
             x="Energy (kCal)",
             showmeans=True,
             data=df,
             orient="v",
             color="skyblue",
             linewidth=2.5,
         )

         # Set the title and labels of the plot
         plt.title("Boxplot of Energy (kCal)")
         plt.xlabel("Energy (kCal)")
         plt.ylabel("Value")

         # Display the plot
         plt.show()
```



Name : Rahul Rajput Registration Number-12107183

Roll Number- RK21UTB49 Set-1 Question Number-6

Q6 Explain the necessity of data Visualization ?

1. Enhanced Comprehension : Complex datasets can be Challenging to Understand when presented in raw numerical form. Visualization simplifies data, making it more accessible and Comprehensible

2. Facilitates Decision-Making : Visualization helps decision-makers identify insights and trends more effectively. Whetor in business, science or policy data pointers enable faster, mor informed decisions.

3 Communication : Data Visualization is a Universal language that transcends barriers. Pointers provide a Common Visual refrence for teams, enabling efficient Communications and Collaboration

4. Detection of Anomalies : Visualization can highlight anomalies, outliers or unexpected patterns in data. This Can be Critical for identifying issues or opportunities that might otherwise go Unnoticed.

5. Monitors Performance: Dashboards and real time Visualizations help track key performance Indicators, enabling organizations to react swiftly to changing Conditions and make data backed improvements.

6. Aids Trend Analysis: Visualizations assists in tracking and Comparing data over time. This is crucial for recognizing long-term trends and planning for the future.

7. Enhances User Engagement: In applications, websites and reports, Visualization capativate users and enhance their engagement. Interactive Visualization can allow users to explore dat on their own.

8. Promotes Data Quality: Visualization can highlight data Quality issues, such as missing or inconsistent data prompting data Cleaning and Validation efforts.