Abbas kashim S

DOCUMENTATION OF FLIGHTS TICKETS

REQUIREMENTS:

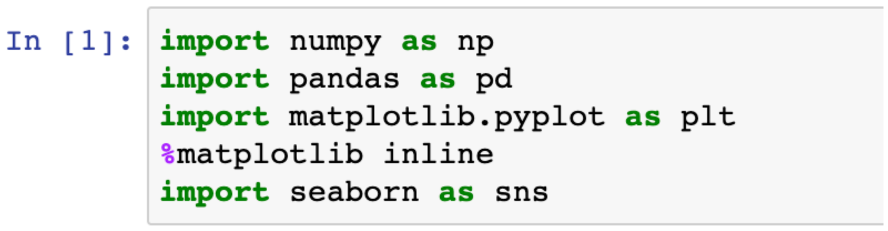
Need To Find The Prices Of Fligts And Which Flight has the top bookings

INDEX:

STEP 1:

This code snippet imports necessary Python libraries for data analysis and visualization.

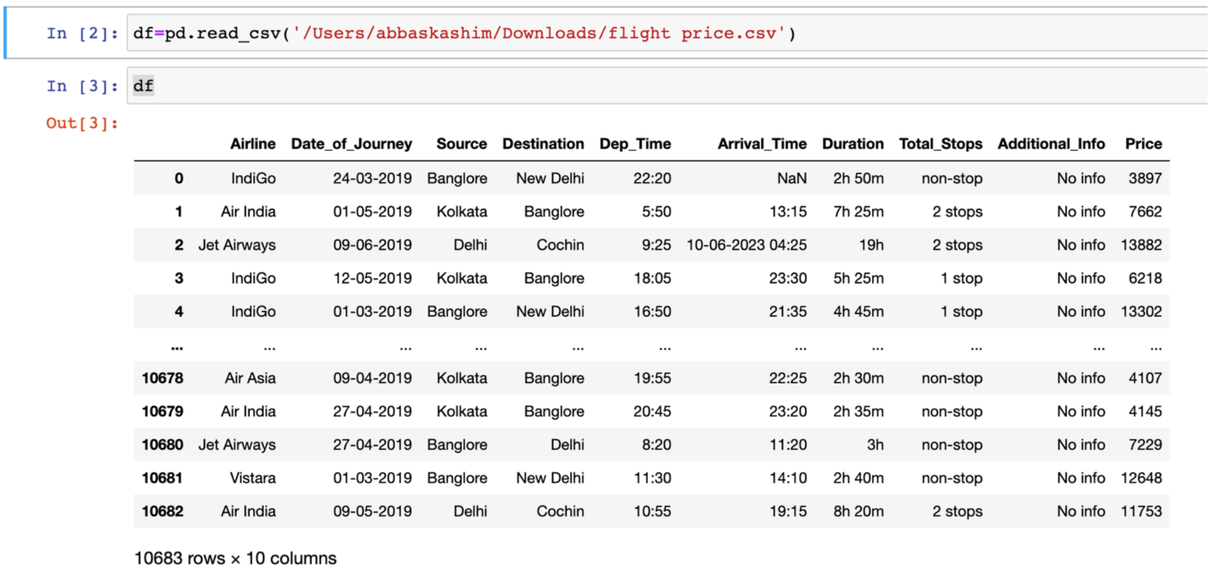
- **import numpy as np:** Imports the NumPy library, which provides support for working with arrays and matrices of numerical data.
- **import pandas as pd:** Imports the Pandas library, used for data manipulation and analysis, particularly with data structures like DataFrames.
- **import matplotlib.pyplot as plt:** Imports the Matplotlib library's pyplot module, which is widely used for creating visualizations like charts, graphs, and plots.
- **%matplotlib inline:** This is a magic command used in Jupyter Notebook environments to display Matplotlib plots directly in the notebook interface.
- **import seaborn as sns:** Imports the Seaborn library, which is built on top of Matplotlib and provides additional functionality for creating aesthetically pleasing statistical visualizations.



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

STEP 2:

This command reads a CSV file named 'flight price.csv' located at the specified file path ('/Users/abbaskashim/Downloads/') using the Pandas library. The data from the CSV file is loaded into a DataFrame, which is a tabular data structure commonly used in data analysis. The variable name 'df' is assigned to this DataFrame. Lastly, by typing 'df', the code is likely intended to display the contents of the DataFrame in the interactive environment, showing the data and its structure to the user.



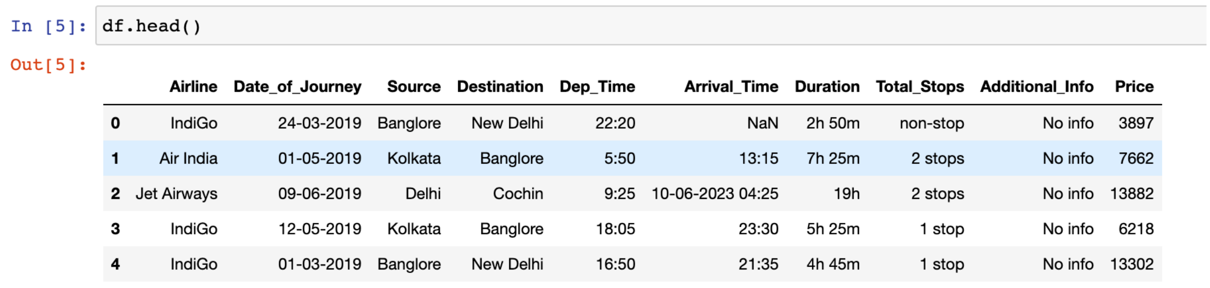
```
In [2]: df=pd.read_csv('/Users/abbaskashim/Downloads/flight price.csv')

In [3]: df
Out[3]:
```

	Airline	Date_of_Journey	Source	Destination	Dep_Time	Arrival_Time	Duration	Total_Stops	Additional_Info	Price
0	IndiGo	24-03-2019	Banglore	New Delhi	22:20	NaN	2h 50m	non-stop	No info	3897
1	Air India	01-05-2019	Kolkata	Banglore	5:50	13:15	7h 25m	2 stops	No info	7662
2	Jet Airways	09-06-2019	Delhi	Cochin	9:25	10-06-2023 04:25	19h	2 stops	No info	13882
3	IndiGo	12-05-2019	Kolkata	Banglore	18:05	23:30	5h 25m	1 stop	No info	6218
4	IndiGo	01-03-2019	Banglore	New Delhi	16:50	21:35	4h 45m	1 stop	No info	13302
...	...	...	...	...	...	...	...	...	...	...
10678	Air Asia	09-04-2019	Kolkata	Banglore	19:55	22:25	2h 30m	non-stop	No info	4107
10679	Air India	27-04-2019	Kolkata	Banglore	20:45	23:20	2h 35m	non-stop	No info	4145
10680	Jet Airways	27-04-2019	Banglore	Delhi	8:20	11:20	3h	non-stop	No info	7229
10681	Vistara	01-03-2019	Banglore	New Delhi	11:30	14:10	2h 40m	non-stop	No info	12648
10682	Air India	09-05-2019	Delhi	Cochin	10:55	19:15	8h 20m	2 stops	No info	11753

10683 rows × 10 columns

STEP 3:

The command "df.head()" is likely used in a programming context, specifically in Python with the Pandas library. It displays the first few rows of a DataFrame (tabular data structure) to provide a quick preview of its contents, helping to understand the data's structure and initial values.

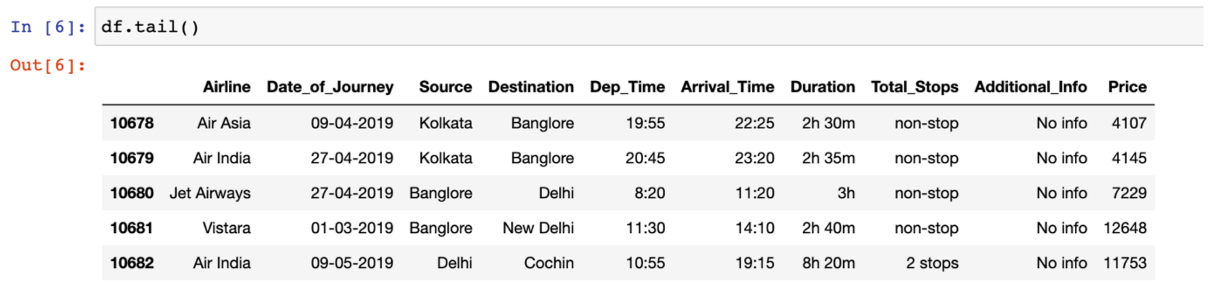


```
In [5]: df.head()
Out[5]:
```

	Airline	Date_of_Journey	Source	Destination	Dep_Time	Arrival_Time	Duration	Total_Stops	Additional_Info	Price
0	IndiGo	24-03-2019	Banglore	New Delhi	22:20	NaN	2h 50m	non-stop	No info	3897
1	Air India	01-05-2019	Kolkata	Banglore	5:50	13:15	7h 25m	2 stops	No info	7662
2	Jet Airways	09-06-2019	Delhi	Cochin	9:25	10-06-2023 04:25	19h	2 stops	No info	13882
3	IndiGo	12-05-2019	Kolkata	Banglore	18:05	23:30	5h 25m	1 stop	No info	6218
4	IndiGo	01-03-2019	Banglore	New Delhi	16:50	21:35	4h 45m	1 stop	No info	13302

STEP 4:

The command "df.tail()" is used in Python with the Pandas library to display the last few rows of a DataFrame, allowing you to observe the end of the dataset and understand its final values and structure.



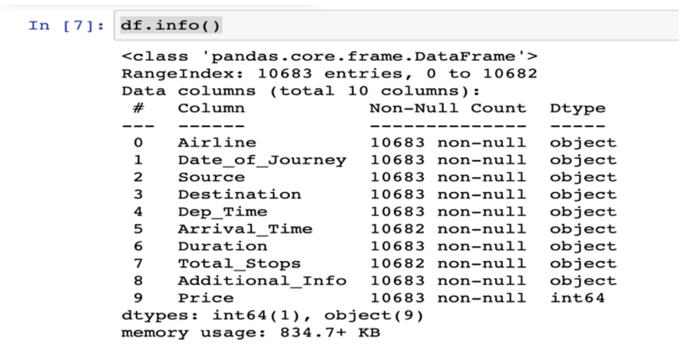
```
In [6]: df.tail()
```

Out[6]:

	Airline	Date_of_Journey	Source	Destination	Dep_Time	Arrival_Time	Duration	Total_Stops	Additional_Info	Price
10678	Air Asia	09-04-2019	Kolkata	Banglore	19:55	22:25	2h 30m	non-stop	No info	4107
10679	Air India	27-04-2019	Kolkata	Banglore	20:45	23:20	2h 35m	non-stop	No info	4145
10680	Jet Airways	27-04-2019	Banglore	Delhi	8:20	11:20	3h	non-stop	No info	7229
10681	Vistara	01-03-2019	Banglore	New Delhi	11:30	14:10	2h 40m	non-stop	No info	12648
10682	Air India	09-05-2019	Delhi	Cochin	10:55	19:15	8h 20m	2 stops	No info	11753

STEP 5:

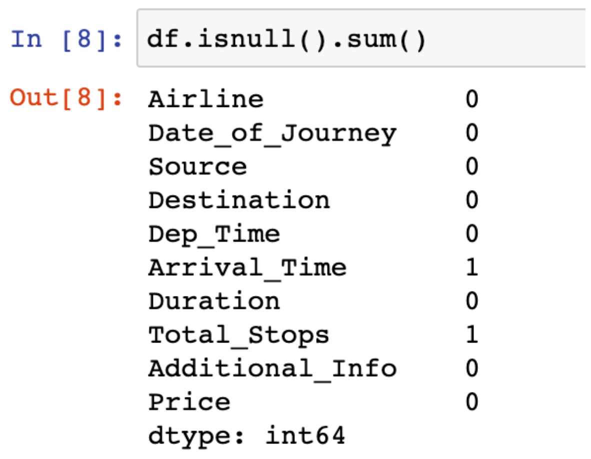
The command "df.info()" is utilized in Python with the Pandas library to obtain a concise summary of a DataFrame's structure. It provides information about the column data types, non-null values, and memory usage, aiding in data exploration and quality assessment.



```
In [7]: df.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10683 entries, 0 to 10682
Data columns (total 10 columns):
 #   Column           Non-Null Count  Dtype
---  ------           --------------  -----
 0   Airline          10683 non-null  object
 1   Date_of_Journey  10683 non-null  object
 2   Source           10683 non-null  object
 3   Destination      10683 non-null  object
 4   Dep_Time         10683 non-null  object
 5   Arrival_Time     10682 non-null  object
 6   Duration         10683 non-null  object
 7   Total_Stops      10682 non-null  object
 8   Additional_Info  10683 non-null  object
 9   Price            10683 non-null  int64
dtypes: int64(1), object(9)
memory usage: 834.7+ KB
```

STEP 5:

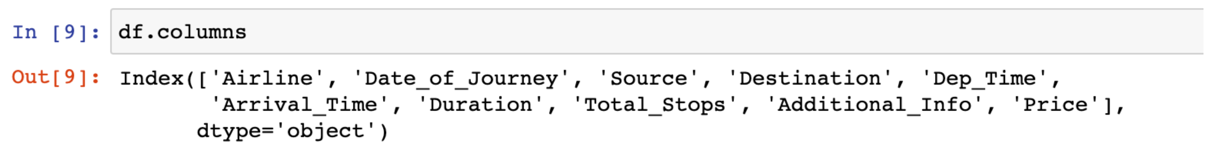
The command "df.isnull().sum()" is used in Python with the Pandas library to calculate and display the number of missing (null) values in each column of a DataFrame. It helps to identify how much data is missing in each column, which is crucial for data preprocessing and analysis.



```
In [8]: df.isnull().sum()

Out[8]: Airline            0
        Date_of_Journey    0
        Source             0
        Destination        0
        Dep_Time           0
        Arrival_Time       1
        Duration           0
        Total_Stops        1
        Additional_Info    0
        Price              0
        dtype: int64
```

STEP 6:

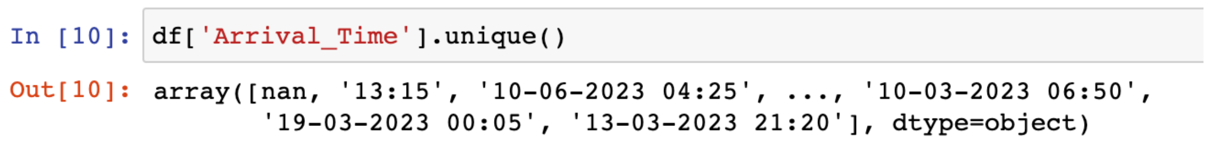
In Python with the Pandas library, the command "df.columns" is used to retrieve and display the list of column names present in a DataFrame named "df." This provides an overview of the available columns and is helpful for referencing specific columns during data manipulation and analysis.



```
In [9]: df.columns

Out[9]: Index(['Airline', 'Date_of_Journey', 'Source', 'Destination', 'Dep_Time',
               'Arrival_Time', 'Duration', 'Total_Stops', 'Additional_Info', 'Price'],
              dtype='object')
```

STEP 7:

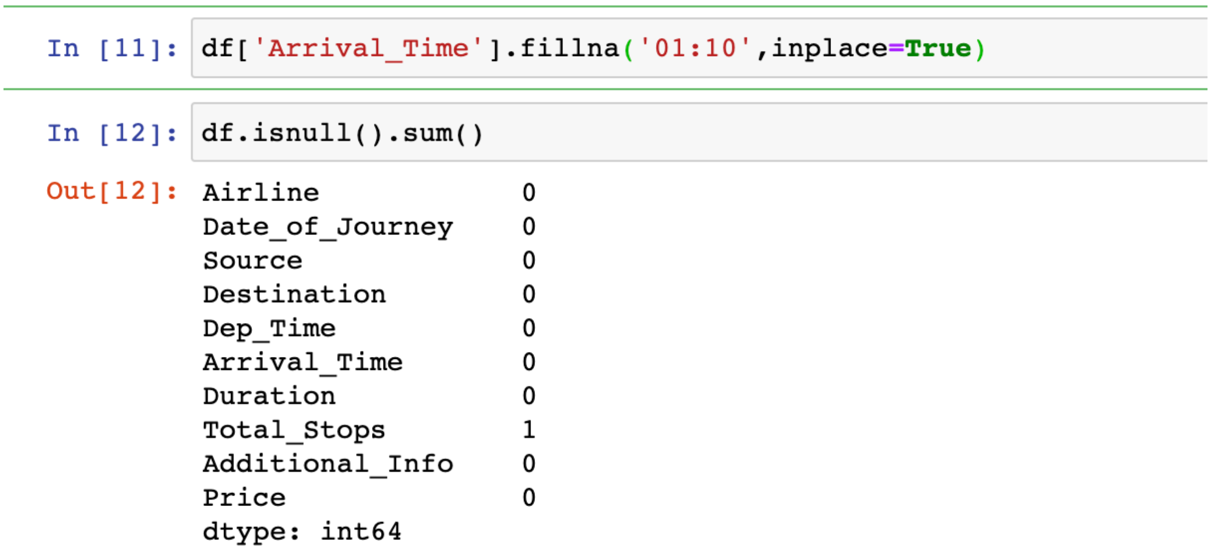
The command "df['Arrival_Time'].unique()" is used to retrieve the unique values present in the "Arrival_Time" column of a DataFrame named "df" in Python with the Pandas library. It provides a list of distinct arrival times within the specified column, useful for understanding the variety of values in that column.



```
In [10]: df['Arrival_Time'].unique()

Out[10]: array([nan, '13:15', '10-06-2023 04:25', ..., '10-03-2023 06:50',
                '19-03-2023 00:05', '13-03-2023 21:20'], dtype=object)
```

STEP 8:

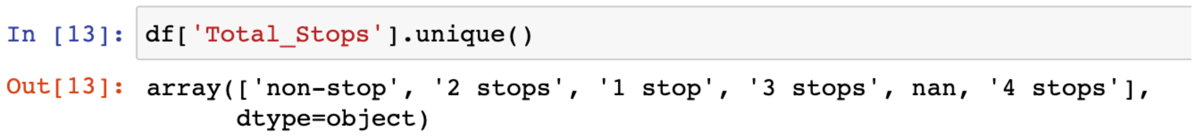
The command "df['Arrival_Time'].fillna('01:10', inplace=True)" is used in Python with the Pandas library to fill any missing (NaN) values in the "Arrival_Time" column of the DataFrame "df" with the value '01:10'. The inplace=True argument means that the changes are applied directly to the DataFrame without the need to assign the result back to the DataFrame variable.



```
In [11]: df['Arrival_Time'].fillna('01:10',inplace=True)

In [12]: df.isnull().sum()

Out[12]: Airline            0
         Date_of_Journey    0
         Source             0
         Destination        0
         Dep_Time           0
         Arrival_Time       0
         Duration           0
         Total_Stops        1
         Additional_Info    0
         Price              0
         dtype: int64
```

STEP 9:

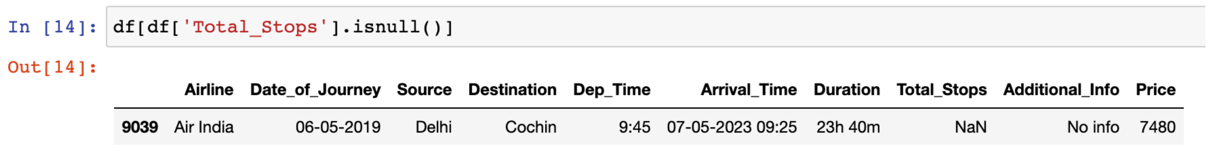
The command "df['Total_Stops'].unique()" is used to retrieve the unique values present in the "Total_Stops" column of a DataFrame named "df" in Python with the Pandas library. It provides a list of distinct values indicating the different numbers of stops for flights, offering insights into the variety of stop options available in the dataset.



```
In [13]: df['Total_Stops'].unique()

Out[13]: array(['non-stop', '2 stops', '1 stop', '3 stops', nan, '4 stops'],
               dtype=object)
```

STEP 10:

The command "df[df['Total_Stops'].isnull()]" is used to filter the DataFrame "df" in Python with the Pandas library, showing rows where the "Total_Stops" column has missing (NaN) values. This helps in identifying and examining the rows with incomplete information regarding the number of stops for flights.

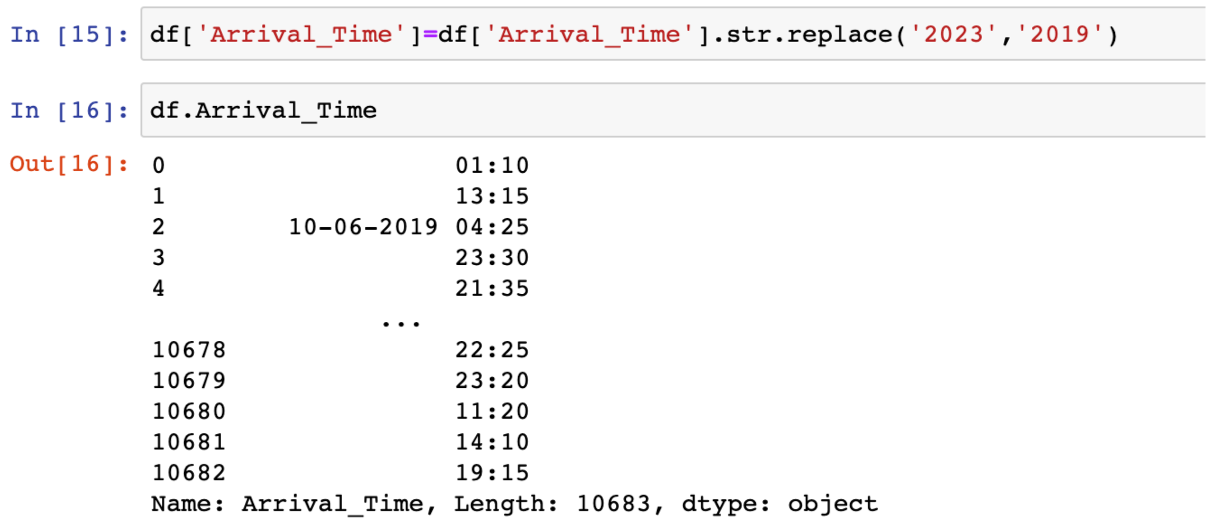


```
In [14]: df[df['Total_Stops'].isnull()]
Out[14]:
```

	Airline	Date_of_Journey	Source	Destination	Dep_Time	Arrival_Time	Duration	Total_Stops	Additional_Info	Price
9039	Air India	06-05-2019	Delhi	Cochin	9:45	07-05-2023 09:25	23h 40m	NaN	No info	7480

STEP 11:

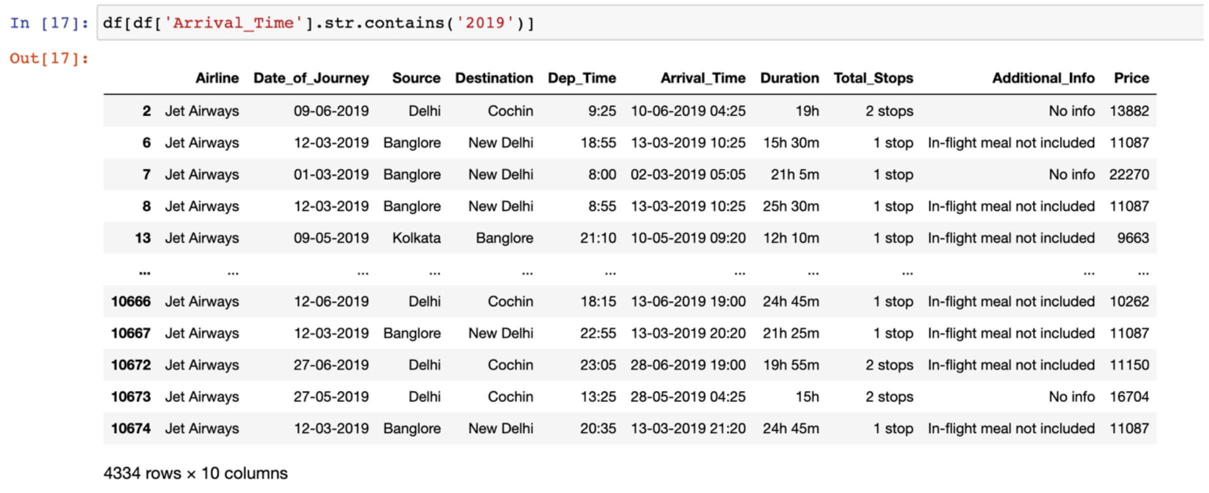
The code you've provided is used to replace the year "2023" with "2019" in the "Arrival_Time" column of the DataFrame "df" using the Pandas library in Python. This is assuming that the "Arrival_Time" column contains strings representing time values with a year component. After executing this code, the "Arrival_Time" column will have the "2023" year replaced by "2019" in all its values. The resulting modified "Arrival_Time" column will be displayed.



```
In [15]: df['Arrival_Time']=df['Arrival_Time'].str.replace('2023','2019')

In [16]: df.Arrival_Time

Out[16]: 0                         01:10
         1                         13:15
         2              10-06-2019 04:25
         3                         23:30
         4                         21:35
                              ...
         10678                     22:25
         10679                     23:20
         10680                     11:20
         10681                     14:10
         10682                     19:15
         Name: Arrival_Time, Length: 10683, dtype: object
```

STEP 12:

The code "df[df['Arrival_Time'].str.contains('2019')]" is used to filter the DataFrame "df" in Python with the Pandas library, showing only the rows where the "Arrival_Time" column contains the substring "2019". This code helps you isolate and examine the rows where the arrival time values contain the year "2019".



```
In [17]: df[df['Arrival_Time'].str.contains('2019')]
```

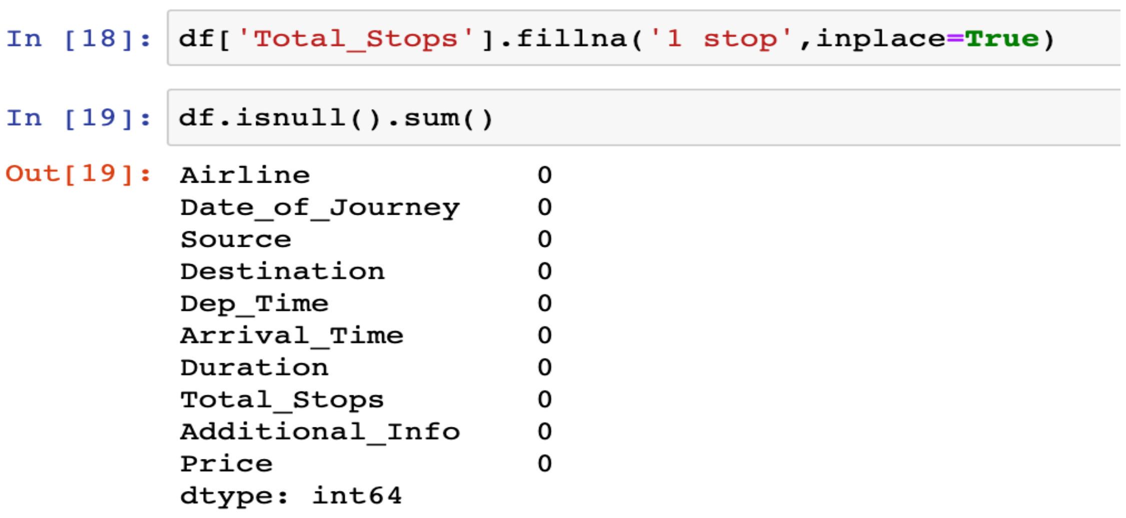
Out[17]:

	Airline	Date_of_Journey	Source	Destination	Dep_Time	Arrival_Time	Duration	Total_Stops	Additional_Info	Price
2	Jet Airways	09-06-2019	Delhi	Cochin	9:25	10-06-2019 04:25	19h	2 stops	No info	13882
6	Jet Airways	12-03-2019	Banglore	New Delhi	18:55	13-03-2019 10:25	15h 30m	1 stop	In-flight meal not included	11087
7	Jet Airways	01-03-2019	Banglore	New Delhi	8:00	02-03-2019 05:05	21h 5m	1 stop	No info	22270
8	Jet Airways	12-03-2019	Banglore	New Delhi	8:55	13-03-2019 10:25	25h 30m	1 stop	In-flight meal not included	11087
13	Jet Airways	09-05-2019	Kolkata	Banglore	21:10	10-05-2019 09:20	12h 10m	1 stop	In-flight meal not included	9663
...	...	...	...	...	...	...	...	...	...	...
10666	Jet Airways	12-06-2019	Delhi	Cochin	18:15	13-06-2019 19:00	24h 45m	1 stop	In-flight meal not included	10262
10667	Jet Airways	12-03-2019	Banglore	New Delhi	22:55	13-03-2019 20:20	21h 25m	1 stop	In-flight meal not included	11087
10672	Jet Airways	27-06-2019	Delhi	Cochin	23:05	28-06-2019 19:00	19h 55m	2 stops	In-flight meal not included	11150
10673	Jet Airways	27-05-2019	Delhi	Cochin	13:25	28-05-2019 04:25	15h	2 stops	No info	16704
10674	Jet Airways	12-03-2019	Banglore	New Delhi	20:35	13-03-2019 21:20	24h 45m	1 stop	In-flight meal not included	11087

4334 rows × 10 columns

STEP 13:

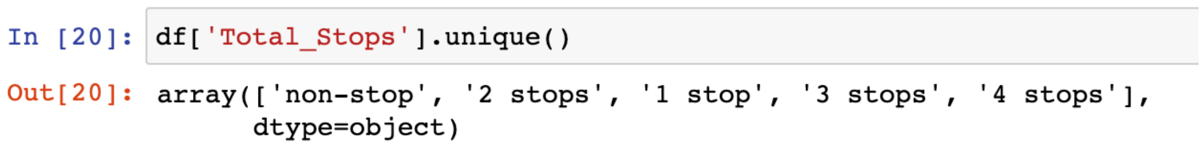
- df['Total_Stops'].fillna('1 stop', inplace=True): This code fills any missing (NaN) values in the "Total_Stops" column of the DataFrame "df" with the value '1 stop'. The inplace=True argument ensures that the changes are applied directly to the DataFrame.
- df.isnull().sum(): This code calculates and displays the sum of missing values in each column of the DataFrame "df". It provides a count of how many missing values exist in each column after the previous filling operation.



```
In [18]: df['Total_Stops'].fillna('1 stop',inplace=True)

In [19]: df.isnull().sum()

Out[19]: Airline            0
         Date_of_Journey    0
         Source             0
         Destination        0
         Dep_Time           0
         Arrival_Time       0
         Duration           0
         Total_Stops        0
         Additional_Info    0
         Price              0
         dtype: int64
```

STEP 14:

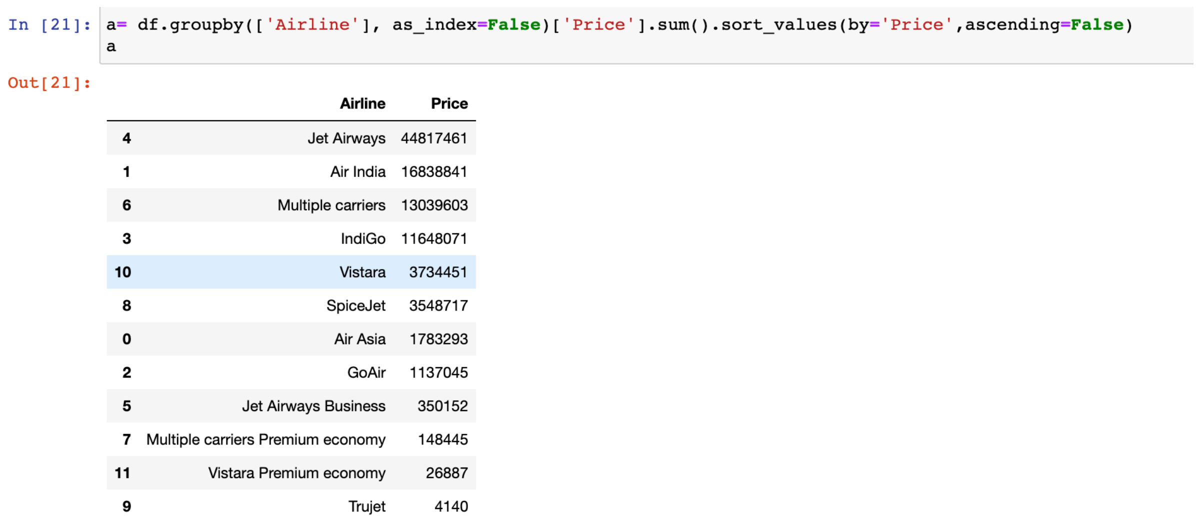
After filling missing values with '1 stop' using the previous code, the command df['Total_Stops'].unique() will display the unique values present in the "Total_Stops" column of the DataFrame "df." This will include the original unique values along with '1 stop' as the new value used to fill the missing values.



```
In [20]: df['Total_Stops'].unique()

Out[20]: array(['non-stop', '2 stops', '1 stop', '3 stops', '4 stops'],
               dtype=object)
```

STEP 15:

- a = df.groupby(['Airline'], as_index=False)['Price'].sum(): This code groups the DataFrame "df" by the "Airline" column and calculates the sum of the "Price" column for each airline. The as_index=False parameter ensures that the "Airline" column is not set as the index of the resulting DataFrame "a."
- .sort_values(by='Price', ascending=False): After calculating the sum of prices for each airline, this code sorts the resulting DataFrame "a" in descending order based on the "Price" column.



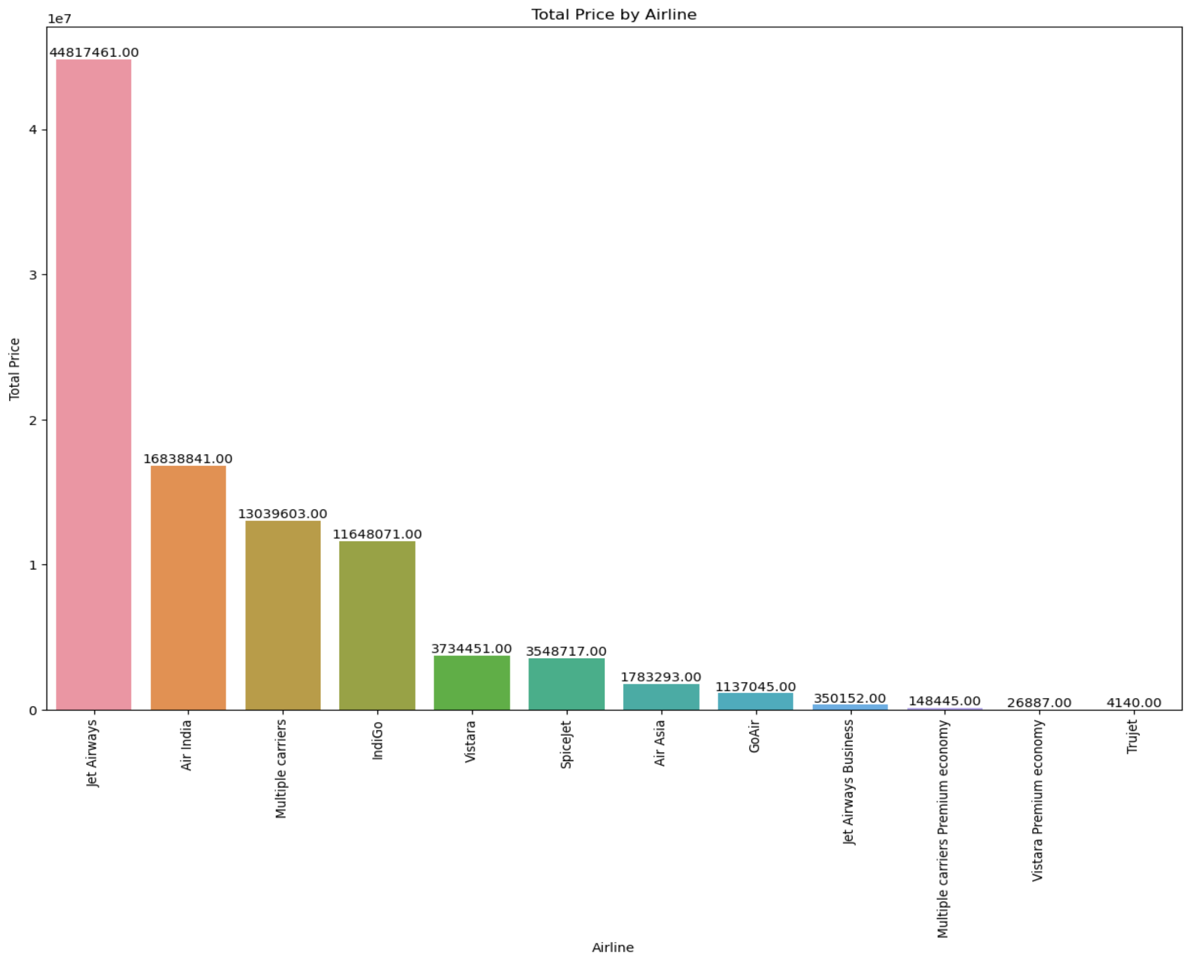
```
In [21]: a= df.groupby(['Airline'], as_index=False)['Price'].sum().sort_values(by='Price',ascending=False)
         a

Out[21]:
```

	Airline	Price
4	Jet Airways	44817461
1	Air India	16838841
6	Multiple carriers	13039603
3	IndiGo	11648071
10	Vistara	3734451
8	SpiceJet	3548717
0	Air Asia	1783293
2	GoAir	1137045
5	Jet Airways Business	350152
7	Multiple carriers Premium economy	148445
11	Vistara Premium economy	26887
9	Trujet	4140

STEP 16:

The code you provided utilizes the Seaborn and Matplotlib libraries to create a bar plot displaying the total prices for each airline. Here's a breakdown of the code:

- plt.figure(figsize=(15, 10)): Sets the figure size for the plot.
- x = sns.barplot(x='Airline', y='Price', data=a): Creates a bar plot using Seaborn's barplot function, where the x-axis represents airlines and the y-axis represents the total prices. The data for the plot is taken from the DataFrame "a."
- for bars in x.containers: and x.bar_label(bars, fmt='%.2f'): These lines add labels to the bars in the bar plot, displaying the total prices with two decimal places.
- plt.title('Total Price by Airline'): Sets the title of the plot.
- plt.xlabel('Airline'): Sets the label for the x-axis.
- plt.ylabel('Total Price'): Sets the label for the y-axis.
- plt.xticks(rotation=90): Rotates the x-axis labels by 90 degrees for better readability.
- plt.show(): Displays the plot.


Total Price by Airline
1e7
Total Price
Airline
Jet Airways
Air India
Multiple carriers
IndiGo
Vistara
SpiceJet
Air Asia
GoAir
Jet Airways Business
Multiple carriers Premium economy
Vistara Premium economy
Trujet