LEVEL-1

Task-1: Data Exploration and Preprocessing

What is data exploration and Data Pre-Processing

-Data exploration refers to the initial step in data analysis in which data analysts use data visualization and statistical techniques to describe dataset characterizations, such as size, quantity, and accuracy, in order to better understand the nature of the data.

-Data Preprocessing-refers to the cleaning, transforming, and integrating of data in order to make it ready for analysis. The goal of data preprocessing is to improve the quality of the data

1-Reading CSV File.

In [1]:
```
import pandas as pd
csv_1=pd.read_csv("D:\Internship\dataset.csv")
csv_1
```

Out[1]:

	Restaurant ID	Restaurant Name	Country Code	City	Address	Locality	Locality Verbose	Longitude	Latitu
0	6317637	Le Petit Souffle	162	Makati City	Third Floor, Century City Mall, Kalayaan Avenu...	Century City Mall, Poblacion, Makati City	Century City Mall, Poblacion, Makati City, Mak...	121.027535	14.565
1	6304287	Izakaya Kikufuji	162	Makati City	Little Tokyo, 2277 Chino Roces Avenue, Legaspi...	Little Tokyo, Legaspi Village, Makati City	Little Tokyo, Legaspi Village, Makati City, Ma...	121.014101	14.553
2	6300002	Heat - Edsa Shangri-La	162	Mandaluyong City	Edsa Shangri-La, 1 Garden Way, Ortigas, Mandal...	Edsa Shangri-La, Ortigas, Mandaluyong City	Edsa Shangri-La, Ortigas, Mandaluyong City, Ma...	121.056831	14.581
3	6318506	Ooma	162	Mandaluyong City	Third Floor, Mega Fashion Hall, SM Megamall, O...	SM Megamall, Ortigas, Mandaluyong City	SM Megamall, Ortigas, Mandaluyong City, Mandal...	121.056475	14.585
4	6314302	Sambo Kojin	162	Mandaluyong City	Third Floor, Mega Atrium, SM Megamall, Ortigas...	SM Megamall, Ortigas, Mandaluyong City	SM Megamall, Ortigas, Mandaluyong City, Mandal...	121.057508	14.584
...	...	...	...	...	...	...	...	...	
9546	5915730	Naml\ Gurme	208	��stanbul	Kemanke�� Karamustafa Pa��a Mahallesi, R\ht\m ...	Karak�_y	Karak�_y, ��stanbul	28.977392	41.022
9547	5908749	Ceviz A��ac\	208	��stanbul	Ko��uyolu Mahallesi, Muhittin ��st�_nda�� Cadd...	Ko��uyolu	Ko��uyolu, ��stanbul	29.041297	41.009
9548	5915807	Huqqa	208	��stanbul	Kuru�_e��me Mahallesi, Muallim Naci Caddesi, N...	Kuru�_e��me	Kuru�_e��me, ��stanbul	29.034640	41.055
9549	5916112	A���k Kahve	208	��stanbul	Kuru�_e��me Mahallesi, Muallim Naci Caddesi, N...	Kuru�_e��me	Kuru�_e��me, ��stanbul	29.036019	41.057
9550	5927402	Walter's Coffee Roastery	208	��stanbul	Cafea��a Mahallesi, Bademalt\ Sokak, No 21/B, ...	Moda	Moda, ��stanbul	29.026016	40.984

9551 rows × 21 columns

-No. of rows =9551

-No of columns = 21

2- Checking no of Missing Values in each Column

In [2]:
```
missing_values = csv_1.isnull().sum()

# Display the result
print("Missing values in each column:")
print(missing_values)
```

```
Missing values in each column:
Restaurant ID           0
Restaurant Name         0
Country Code            0
City                    0
Address                 0
Locality                0
Locality Verbose        0
Longitude               0
Latitude                0
Cuisines                9
Average Cost for two    0
Currency                0
Has Table booking       0
Has Online delivery     0
Is delivering now       0
Switch to order menu    0
Price range             0
Aggregate rating        0
Rating color            0
Rating text             0
Votes                   0
dtype: int64
```

-Hence there is a missing value in column cuisines

Cousines - number of missing values =9

So we can ignore it or reolace it by not specified

In [3]:
```
csv_1['Cuisines'].fillna('Not Specified', inplace= True)
```

In [4]:
```
# lets check it again

csv_1.isnull().sum()
```

Out[4]:
```
Restaurant ID           0
Restaurant Name         0
Country Code            0
City                    0
Address                 0
Locality                0
Locality Verbose        0
Longitude               0
Latitude                0
Cuisines                0
Average Cost for two    0
Currency                0
Has Table booking       0
Has Online delivery     0
Is delivering now       0
Switch to order menu    0
Price range             0
Aggregate rating        0
Rating color            0
Rating text             0
Votes                   0
dtype: int64
```

Now there is no missing values in the data set

In [5]:
```
## check for duplicate

dup = csv_1.duplicated().sum()
print(f'Number of duplicate rows are {dup}')
```

```
Number of duplicate rows are 0
```

In [6]:
```
csv_1.head()
```

Out[6]:

	Restaurant ID	Restaurant Name	Country Code	City	Address	Locality	Locality Verbose	Longitude	Latitude	Cuisines
0	6317637	Le Petit Souffle	162	Makati City	Third Floor, Century City Mall, Kalayaan Avenu...	Century City Mall, Poblacion, Makati City	Century City Mall, Poblacion, Makati City, Mak...	121.027535	14.565443	French, Japanese, Desserts
1	6304287	Izakaya Kikufuji	162	Makati City	Little Tokyo, 2277 Chino Roces Avenue, Legaspi...	Little Tokyo, Legaspi Village, Makati City	Little Tokyo, Legaspi Village, Makati City, Ma...	121.014101	14.553708	Japanese
2	6300002	Heat - Edsa Shangri-La	162	Mandaluyong City	Edsa Shangri-La, 1 Garden Way, Ortigas, Mandal...	Edsa Shangri-La, Ortigas, Mandaluyong City	Edsa Shangri-La, Ortigas, Mandaluyong City, Ma...	121.056831	14.581404	Seafood, Asian, Filipino, Indian
3	6318506	Ooma	162	Mandaluyong City	Third Floor, Mega Fashion Hall, SM Megamall, O...	SM Megamall, Ortigas, Mandaluyong City	SM Megamall, Ortigas, Mandaluyong City, Mandal...	121.056475	14.585318	Japanese, Sushi
4	6314302	Sambo Kojin	162	Mandaluyong City	Third Floor, Mega Atrium, SM Megamall, Ortigas...	SM Megamall, Ortigas, Mandaluyong City	SM Megamall, Ortigas, Mandaluyong City, Mandal...	121.057508	14.584450	Japanese, Korean

5 rows × 21 columns

In [7]:
```
csv_1.tail()
```

Out[7]:

	Restaurant ID	Restaurant Name	Country Code	City	Address	Locality	Locality Verbose	Longitude	Latitude
9546	5915730	Naml\ Gurme	208	��stanbul	Kemanke�� Karamustafa Pa��a Mahallesi, R\ht\m ...	Karak�_y	Karak�_y, ��stanbul	28.977392	41.022793
9547	5908749	Ceviz A��ac\	208	��stanbul	Ko��uyolu Mahallesi, Muhittin ��st�_nda�� Cadd...	Ko��uyolu	Ko��uyolu, ��stanbul	29.041297	41.009847
9548	5915807	Huqqa	208	��stanbul	Kuru�_e��me Mahallesi, Muallim Naci Caddesi, N...	Kuru�_e��me	Kuru�_e��me, ��stanbul	29.034640	41.055817
9549	5916112	A���k Kahve	208	��stanbul	Kuru�_e��me Mahallesi, Muallim Naci Caddesi, N...	Kuru�_e��me	Kuru�_e��me, ��stanbul	29.036019	41.057979
9550	5927402	Walter's Coffee Roastery	208	��stanbul	Cafea��a Mahallesi, Bademalt\ Sokak, No 21/B, ...	Moda	Moda, ��stanbul	29.026016	40.984776

5 rows × 21 columns

To find no of rows and columns

In [8]:
```
csv_1.shape
```

Out[8]: (9551, 21)

3- Now we have to perform data conversion if needed. Analyze the distribution of target variable [,Aggregate Rating'] and identify any class imbalance

In [9]:
```
csv_1.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 9551 entries, 0 to 9550
Data columns (total 21 columns):
 #   Column                Non-Null Count  Dtype  
---  ------                --------------  -----  
 0   Restaurant ID         9551 non-null   int64  
 1   Restaurant Name       9551 non-null   object 
 2   Country Code          9551 non-null   int64  
 3   City                  9551 non-null   object 
 4   Address               9551 non-null   object 
 5   Locality              9551 non-null   object 
 6   Locality Verbose      9551 non-null   object 
 7   Longitude             9551 non-null   float64
 8   Latitude              9551 non-null   float64
 9   Cuisines              9551 non-null   object 
 10  Average Cost for two  9551 non-null   int64  
 11  Currency              9551 non-null   object 
 12  Has Table booking     9551 non-null   object 
 13  Has Online delivery   9551 non-null   object 
 14  Is delivering now     9551 non-null   object 
 15  Switch to order menu  9551 non-null   object 
 16  Price range           9551 non-null   int64  
 17  Aggregate rating      9551 non-null   float64
 18  Rating color          9551 non-null   object 
 19  Rating text           9551 non-null   object 
 20  Votes                 9551 non-null   int64  
dtypes: float64(3), int64(5), object(13)
memory usage: 1.5+ MB
```

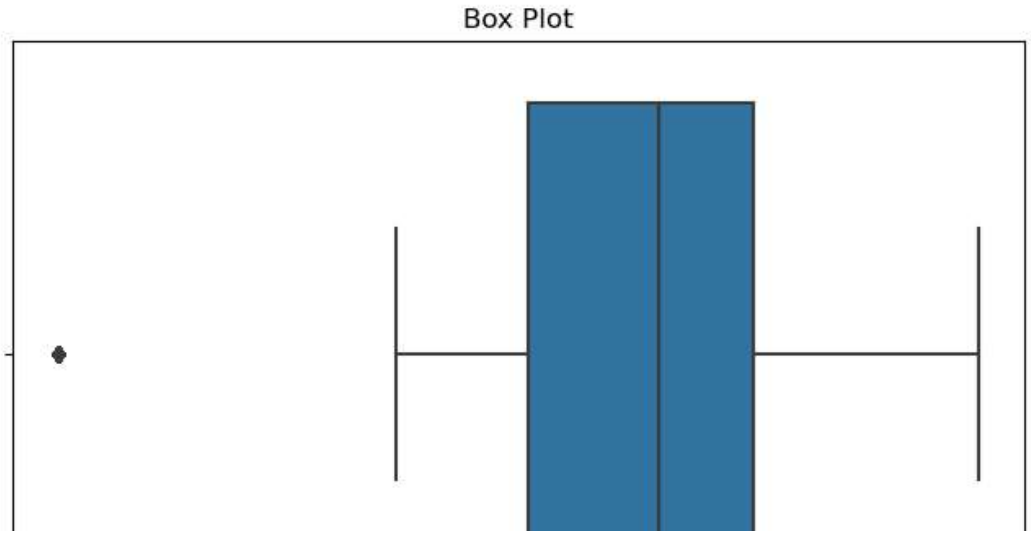
So , no need to any data conversion

In [10]:
```
# Targhet variable [" Aggregate Rating"]

target = "Aggregate rating"

# Descriptive statistics

print(csv_1[target].describe())
```

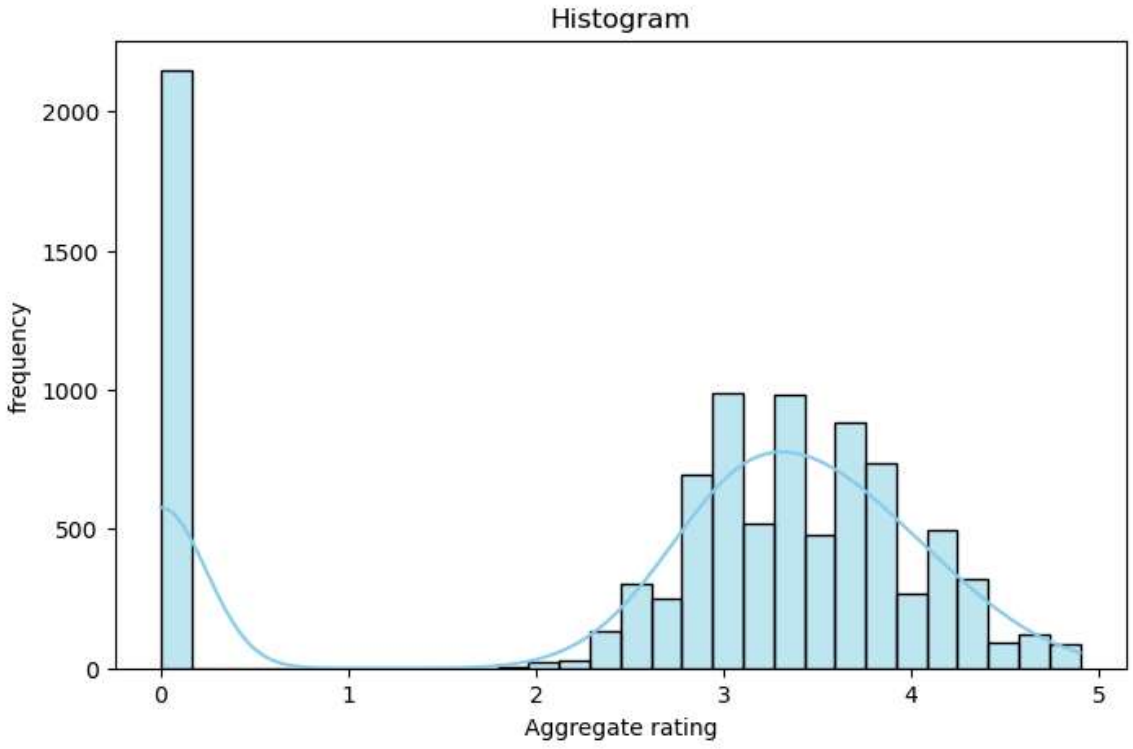
```
count    9551.000000
mean        2.666370
std         1.516378
min         0.000000
25%         2.500000
50%         3.200000
75%         3.700000
max         4.900000
Name: Aggregate rating, dtype: float64
```

In [11]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

In [12]:
```
plt.figure(figsize=(8, 5))
sns.boxplot(x=csv_1[target])
plt.title('Box Plot')
plt.xlabel('Aggregate rating')
plt.show()
```



In [13]:
```
# Histogram
plt.figure(figsize=(8, 5))
sns.histplot(x=csv_1[target],bins =30,kde =True , color ='Skyblue')
plt.title('Histogram')
plt.xlabel('Aggregate rating')
plt.ylabel('frequency')
plt.show()
```



No class imbalance

Level 1-Task 2

Task : Descriptive Analysis

Calculate basic statistical measure (mean ,median ,SD etc) for numerical columns

In [14]:
```
csv_1.describe()
```

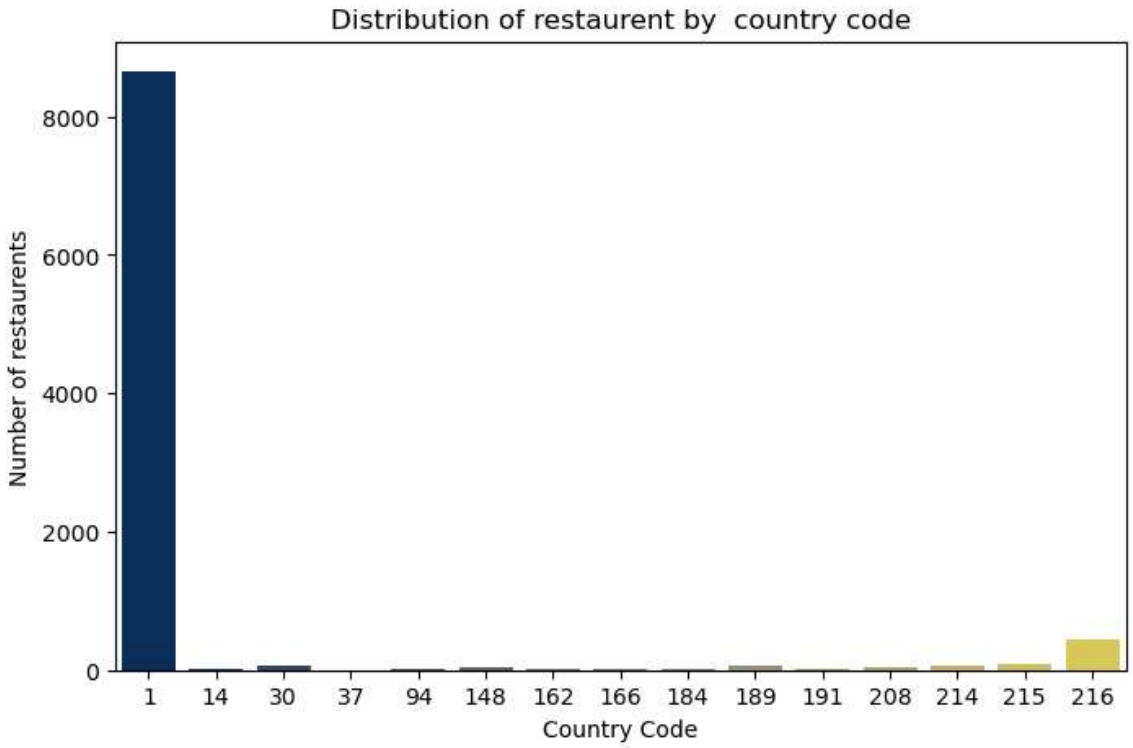
Out[14]:

	Restaurant ID	Country Code	Longitude	Latitude	Average Cost for two	Price range	Aggregate rating	Votes
count	9.551000e+03	9551.000000	9551.000000	9551.000000	9551.000000	9551.000000	9551.000000	9551.000000
mean	9.051128e+06	18.365616	64.126574	25.854381	1199.210763	1.804837	2.666370	156.909748
std	8.791521e+06	56.750546	41.467058	11.007935	16121.183073	0.905609	1.516378	430.169145
min	5.300000e+01	1.000000	-157.948486	-41.330428	0.000000	1.000000	0.000000	0.000000
25%	3.019625e+05	1.000000	77.081343	28.478713	250.000000	1.000000	2.500000	5.000000
50%	6.004089e+06	1.000000	77.191964	28.570469	400.000000	2.000000	3.200000	31.000000
75%	1.835229e+07	1.000000	77.282006	28.642758	700.000000	2.000000	3.700000	131.000000
max	1.850065e+07	216.000000	174.832089	55.976980	800000.000000	4.000000	4.900000	10934.000000

2-Explore the distribution of categorical variable like "Country Code", "City","Cuisines" . Identify the top city and Cuisines whit highest number of restaurents.

In [15]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

In [16]:
```
# explore the distribution of "country code"

plt.figure(figsize=(8, 5))
sns.countplot(x= 'Country Code', data=csv_1,palette ='cividis')

plt.title('Distribution of restaurent by  country code')
plt.xlabel('Country Code')
plt.ylabel('Number of restaurents')
plt.show()
```

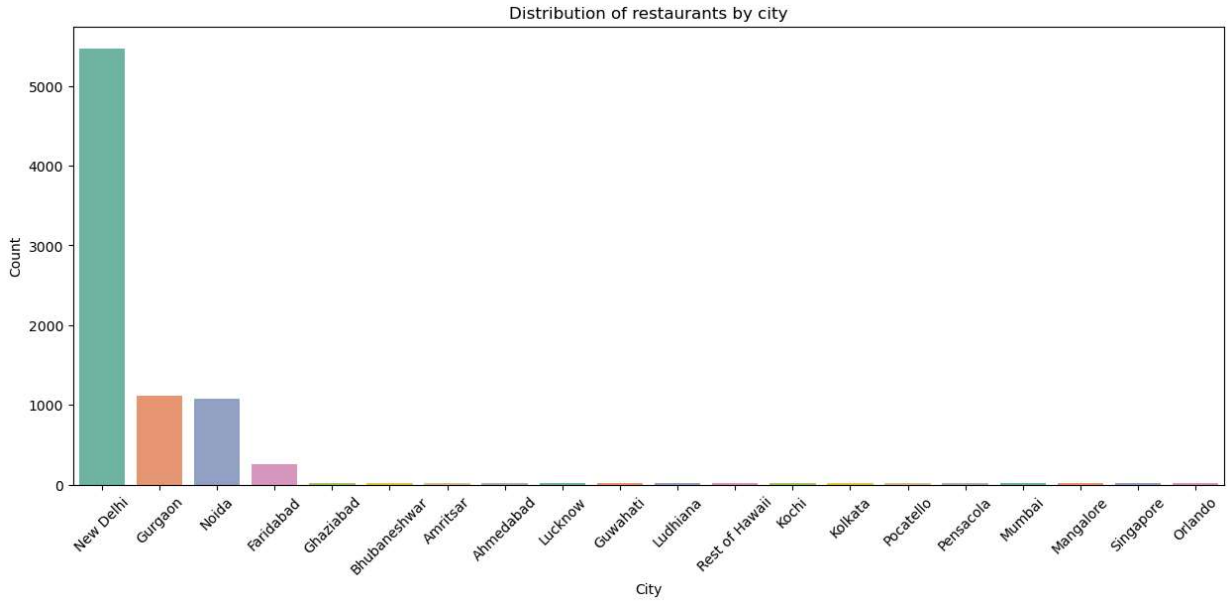


The majority of restaurents are located om Country Code 1 followed by second highest concentration in country code 216

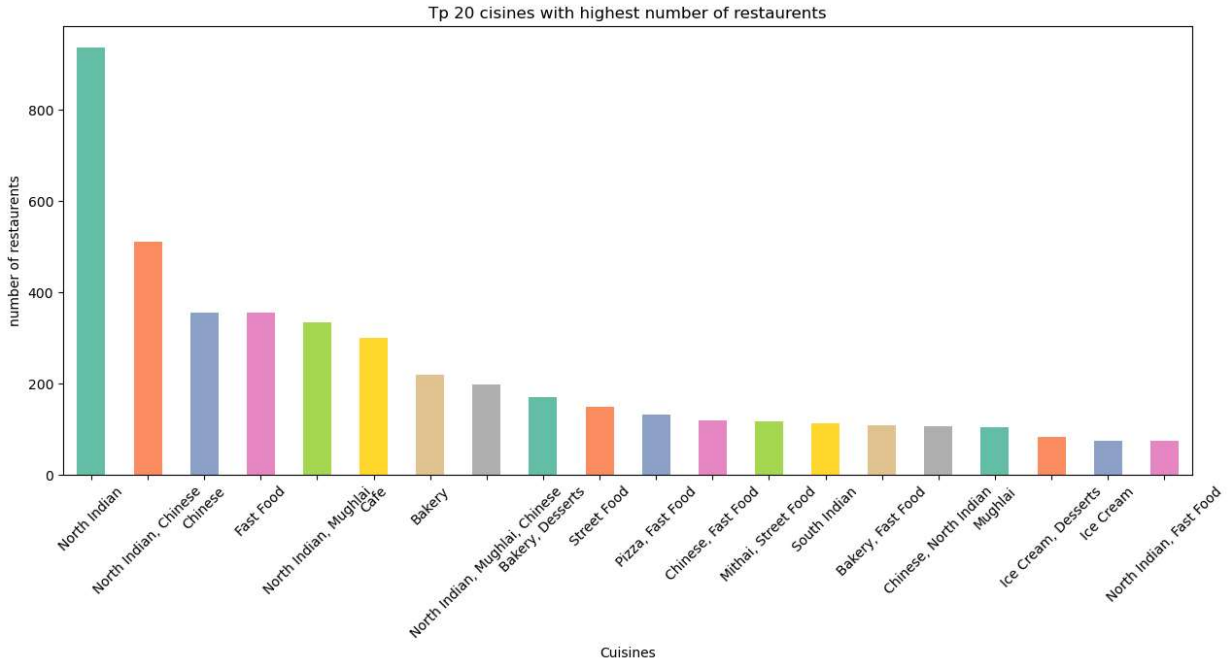
In [17]:
```
top_countries = csv_1["Country Code"].value_counts().head()
print('Top 5 countries with the highest number of restaurants: ')
print(top_countries)
```

```
Top 5 countries with the highest number of restaurants:
1      8652
216     434
215      80
30       60
214      60
Name: Country Code, dtype: int64
```

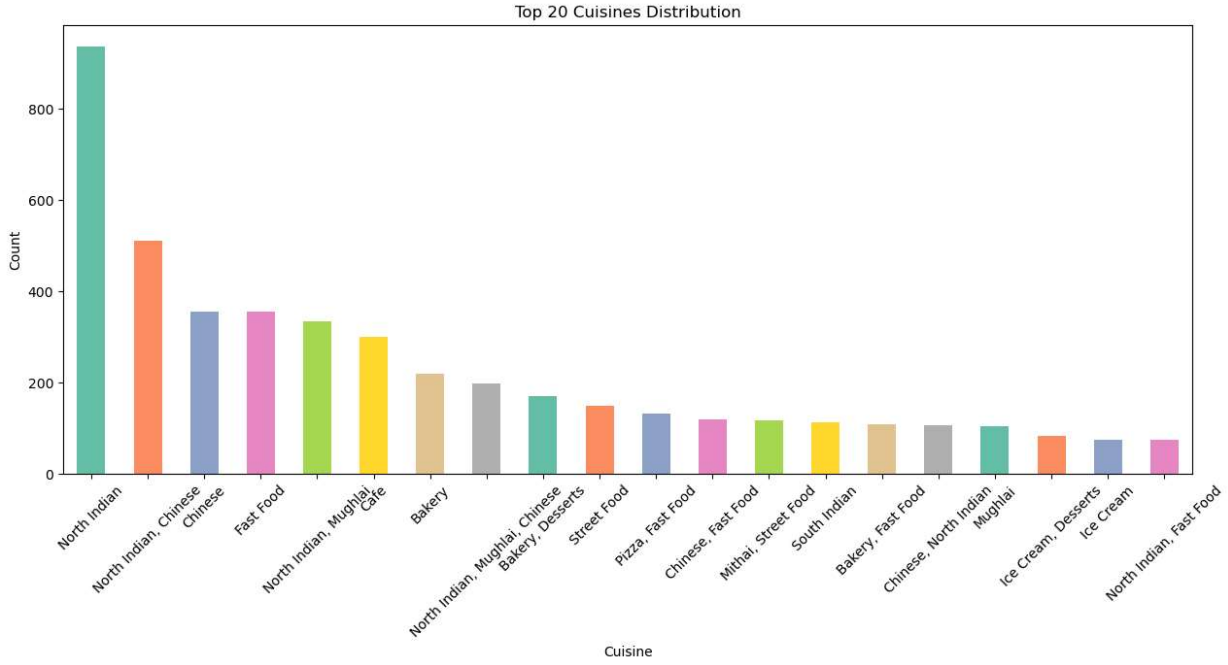
In [18]:
```
plt.figure(figsize=(15, 6))

# Correct method name value_counts and correct variable naming for order
sns.countplot(x='City', data=csv_1, order=csv_1['City'].value_counts().head(20).index, palette='Set

plt.title('Distribution of restaurants by city')
plt.xlabel('City')
plt.ylabel('Count')
plt.xticks(rotation=45)
plt.show()
```



In [19]:
```
plt.figure(figsize=(15, 6))

# Correct method name value_counts and correct variable naming for order
Cuisines_count = csv_1['Cuisines'].value_counts()
Cuisines_count.head(20).plot(kind='bar', color=sns.color_palette("Set2"))

plt.title('Tp 20 cisines with highest number of restaurents')
plt.xlabel('Cuisines')
plt.ylabel('number of restaurents')
plt.xticks(rotation=45)
plt.show()
```



In [20]:
```
import matplotlib.pyplot as plt
import seaborn as sns

# Assuming cuisines_count is a Series with the count of cuisines
cuisines_count = csv_1['Cuisines'].value_counts()  # Replace 'Cuisine' with the actual column name

plt.figure(figsize=(15, 6))
cuisines_count.head(20).plot(kind='bar', color=sns.color_palette('Set2'))

plt.title('Top 20 Cuisines Distribution')
plt.xlabel('Cuisine')
plt.ylabel('Count')
plt.xticks(rotation=45)
plt.show()
```



Top cuisines and Cities

In [21]:
```
top_cities = csv_1['City'].value_counts().head(10)
print('topo 10  cities with highest number of restaurents :')
print(top_cities)
```

```
topo 10  cities with highest number of restaurents :
New Delhi       5473
Gurgaon         1118
Noida           1080
Faridabad        251
Ghaziabad         25
Bhubaneshwar      21
Amritsar          21
Ahmedabad         21
Lucknow           21
Guwahati          21
Name: City, dtype: int64
```

In [22]:
```
top_cuisines = cuisines_count.head(10)
print('top 10 cuisines with highest number of restaurents:')
print(top_cuisines)
```

```
top 10 cuisines with highest number of restaurents:
North Indian                      936
North Indian, Chinese             511
Chinese                           354
Fast Food                         354
North Indian, Mughlai             334
Cafe                              299
Bakery                            218
North Indian, Mughlai, Chinese    197
Bakery, Desserts                  170
Street Food                       149
Name: Cuisines, dtype: int64
```

Task-3

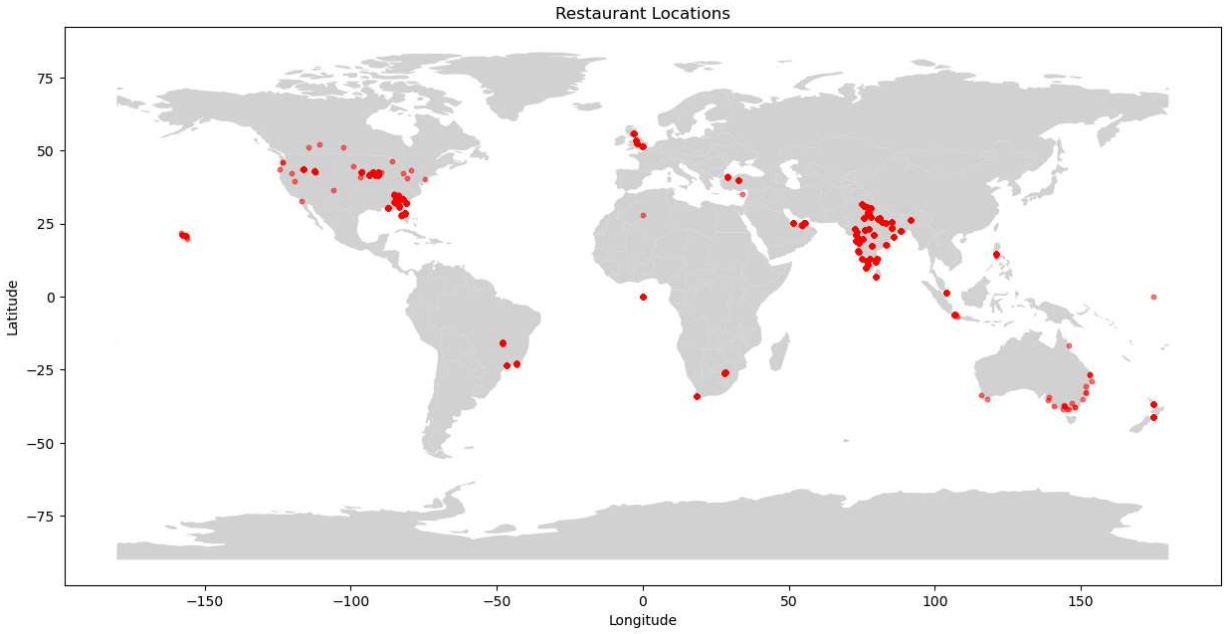
Geo-Spatial Analysis

1-Visualise the locations of the restaurents on map using latitude and longitude information.

In [23]:
```
# Location of restaurents on a map using latitude and longitudesinformation
#import tyhe necessary libraries

from shapely.geometry import point
import geopandas as gpd
from geopandas import GeoDataFrame
```

In [24]:
```
from shapely.geometry import Point
import geopandas as gpd
import matplotlib.pyplot as plt

# Assuming csv_1 is your DataFrame with 'Latitude' and 'Longitude' columns
# Replace 'Latitude' and 'Longitude' with the actual column names if different

# Create a GeoDataFrame
gdf = gpd.GeoDataFrame(
    csv_1, geometry=gpd.points_from_xy(csv_1['Longitude'], csv_1['Latitude'])
)

# Plot the locations of restaurants
world = gpd.read_file(gpd.datasets.get_path('naturalearth_lowres'))

fig, ax = plt.subplots(figsize=(15, 10))
world.plot(ax=ax, color='lightgrey')

gdf.plot(ax=ax, markersize=10, color='red', alpha=0.5)
plt.title('Restaurant Locations')
plt.xlabel('Longitude')
plt.ylabel('Latitude')
plt.show()
```

```
C:\Users\anami\AppData\Local\Temp\ipykernel_13416\3698913475.py:14: FutureWarning: The geopandas.dataset module is deprecated and will be removed in GeoPandas 1.0. You can get the original 'naturalearth_lowres' data from https://www.naturalearthdata.com/downloads/110m-cultural-vectors/. (https://www.naturalearthdata.com/downloads/110m-cultural-vectors/.)
  world = gpd.read_file(gpd.datasets.get_path('naturalearth_lowres'))
```



In [25]:
```
pip install shapely
```

```
Requirement already satisfied: shapely in c:\users\anami\anaconda3\lib\site-packages (2.0.4)
Requirement already satisfied: numpy<3,>=1.14 in c:\users\anami\anaconda3\lib\site-packages (from shapely) (1.24.3)
Note: you may need to restart the kernel to use updated packages.
```

In [26]:
```
pip install geopandas
```

```
Requirement already satisfied: geopandas in c:\users\anami\anaconda3\lib\site-packages (0.14.4)
Requirement already satisfied: fiona>=1.8.21 in c:\users\anami\anaconda3\lib\site-packages (from g
eopandas) (1.9.6)
Requirement already satisfied: numpy>=1.22 in c:\users\anami\anaconda3\lib\site-packages (from geo
pandas) (1.24.3)
Requirement already satisfied: packaging in c:\users\anami\anaconda3\lib\site-packages (from geopa
ndas) (23.0)
Requirement already satisfied: pandas>=1.4.0 in c:\users\anami\anaconda3\lib\site-packages (from g
eopandas) (1.5.3)
Requirement already satisfied: pyproj>=3.3.0 in c:\users\anami\anaconda3\lib\site-packages (from g
eopandas) (3.6.1)
Requirement already satisfied: shapely>=1.8.0 in c:\users\anami\anaconda3\lib\site-packages (from
geopandas) (2.0.4)
Requirement already satisfied: attrs>=19.2.0 in c:\users\anami\anaconda3\lib\site-packages (from f
iona>=1.8.21->geopandas) (22.1.0)
Requirement already satisfied: certifi in c:\users\anami\anaconda3\lib\site-packages (from fiona>=
1.8.21->geopandas) (2023.5.7)
Requirement already satisfied: click~=8.0 in c:\users\anami\anaconda3\lib\site-packages (from fion
a>=1.8.21->geopandas) (8.0.4)
Requirement already satisfied: click-plugins>=1.0 in c:\users\anami\anaconda3\lib\site-packages (f
rom fiona>=1.8.21->geopandas) (1.1.1)
Requirement already satisfied: cligj>=0.5 in c:\users\anami\anaconda3\lib\site-packages (from fion
a>=1.8.21->geopandas) (0.7.2)
Requirement already satisfied: six in c:\users\anami\anaconda3\lib\site-packages (from fiona>=1.8.
21->geopandas) (1.16.0)
Requirement already satisfied: python-dateutil>=2.8.1 in c:\users\anami\anaconda3\lib\site-package
s (from pandas>=1.4.0->geopandas) (2.8.2)
Requirement already satisfied: pytz>=2020.1 in c:\users\anami\anaconda3\lib\site-packages (from pa
ndas>=1.4.0->geopandas) (2022.7)
Requirement already satisfied: colorama in c:\users\anami\anaconda3\lib\site-packages (from click~
=8.0->fiona>=1.8.21->geopandas) (0.4.6)
Note: you may need to restart the kernel to use updated packages.
```

In [27]:
```
# Create Point geomtry from latitude and longitude using Shapely
gdf = gpd.GeoDataFrame(csv_1, geometry=gpd.points_from_xy(csv_1.Longitude, csv_1.Latitude))

# Create a base map of the world using Geopandas
world = gpd.read_file(gpd.datasets.get_path('naturalearth_lowers'))

# Create a map that fits the screen and plots the restaurant locations
gdf.plot(ax=world.plot("continent",legend = True, figsize=(18, 15)), marker= '0', color'red', marke

plt.show()
```

```
  Cell In[27], line 8
    gdf.plot(ax=world.plot("continent",legend = True, figsize=(18, 15)), marker= '0', color'red',
markersize= 15)

    ^
SyntaxError: positional argument follows keyword argument
```

In [ ]:
```
plt.show
```

2-Analyze thedistribution of restaurents across different cities or countries. determine if there is any correlation between the restaurents location and its rating

In [ ]:
```
plt.figure(figsize=(8, 5))
sns.countplot(y = csv_1['City'],order = csv_1.City.value_counts().head(10).index,palette='Set2')


plt.xlabel('No of restaurents')
plt.ylabel('No of cities')
plt.title('Distribution of restaurents across cities')
plt.show()
```

In [ ]:
```
import seaborn as sns
import matplotlib.pyplot as plt

# Assuming 'csv_1' is your DataFrame
sns.countplot(y=csv_1['City'], order=csv_1['City'].value_counts().head(10).index, palette='Set2')

correlation_matrix = csv_1[['Latitude', 'Longitude', 'Aggregate rating']].corr()
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm', fmt=".2f")

plt.title('Correlation between restaurants located at rating')
plt.show()
```

OBSERVATION:-

-The Restaurents dataset has various attributes such as Restaurents Ids,name,city, country,types of cuisines.

-There are 9551 rows and 21 columns

-In this dataset9 missing values from "Cuisines"column.so it can be replaced by non specified.

-In dataset no duplicates are present.

-Top cuisines are "North india","Chinese","Fast Food"

-USA and India has most number of restsaurents

Level 2: Task-1

Task-1: Table Booking and Online Delivery

-Determine the percentage of restaurants that offer table booking and online delivery.

-Compare the average ratings of restaurants with table booking and those without.

-Analyze the availability of online delivery among restaurants with different price ranges

Task-2: Price Range Analysis

-Determine the most common price range among all the restaurants.

-Calculate the average rating for each price range.

-Identify the color that represents the highest average rating among different price ranges.

Task-3: Feature Engineering

-Extract additional features from the existing columns, such as the length of the restaurant name or address.

-Create new features like "Has Table Booking" or "Has Online Delivery" by encoding categorical variables

In [28]:
```
# Importing the warnings

import warnings
warnings.filterwarnings("ignore")
```

In [29]:
```
# Importing libraries which we are going to use in EDA.

import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns;sns.set(color_codes=True)
%matplotlib inline
```

Level 2-Task 1:-

Task: Table Booking and Online Delivery

Determine the percentage of restaurents that offer table booking and online delivery.

In [30]:
```
# Print column names to check for discrepancies
print(csv_1.columns)

# Strip any leading/trailing spaces from column names
csv_1.columns = csv_1.columns.str.strip()

# Convert column names to lowercase for consistency (optional)
csv_1.columns = csv_1.columns.str.lower()

# Use the correct column name (adjust based on your actual column names)
try:
    print(csv_1["has table booking"].value_counts())
except KeyError:
    print("The column 'Has Table Booking' does not exist in the DataFrame.")
```

```
Index(['Restaurant ID', 'Restaurant Name', 'Country Code', 'City', 'Address',
       'Locality', 'Locality Verbose', 'Longitude', 'Latitude', 'Cuisines',
       'Average Cost for two', 'Currency', 'Has Table booking',
       'Has Online delivery', 'Is delivering now', 'Switch to order menu',
       'Price range', 'Aggregate rating', 'Rating color', 'Rating text',
       'Votes'],
      dtype='object')
No     8393
Yes    1158
Name: has table booking, dtype: int64
```

In [31]:
```
# Print the list of columns to debug
print(csv_1.columns)

# Clean the column names
csv_1.columns = csv_1.columns.str.strip()  # Strip leading/trailing spaces
csv_1.columns = csv_1.columns.str.lower()  # Convert to lowercase for uniformity

# Print the cleaned column names to verify
print(csv_1.columns)

# Use the correct column name based on cleaned columns
try:
    print(csv_1["has online delivery"].value_counts())  # Adjust based on actual column name
except KeyError:
    print("The column 'Has Online delivery' does not exist in the DataFrame.")
```

```
Index(['restaurant id', 'restaurant name', 'country code', 'city', 'address',
       'locality', 'locality verbose', 'longitude', 'latitude', 'cuisines',
       'average cost for two', 'currency', 'has table booking',
       'has online delivery', 'is delivering now', 'switch to order menu',
       'price range', 'aggregate rating', 'rating color', 'rating text',
       'votes'],
      dtype='object')
Index(['restaurant id', 'restaurant name', 'country code', 'city', 'address',
       'locality', 'locality verbose', 'longitude', 'latitude', 'cuisines',
       'average cost for two', 'currency', 'has table booking',
       'has online delivery', 'is delivering now', 'switch to order menu',
       'price range', 'aggregate rating', 'rating color', 'rating text',
       'votes'],
      dtype='object')
No     7100
Yes    2451
Name: has online delivery, dtype: int64
```

In [32]:
```
print("Table Booking : ",round((1158/(8393+1158)) *100, 2),"%")
print("Online Delivery : ",round((2451/(7100+2451)) *100, 2),"%")
```

```
Table Booking :  12.12 %
Online Delivery :  25.66 %
```

2- Compare the avarage reting of restaurentswith table booking and those without

In [34]:
```
# Step 1: Print the column names to see if 'Has Table Booking' exists and is correctly named
print(csv_1.columns)

# Step 2: Strip any leading or trailing whitespace from the column names
csv_1.columns = csv_1.columns.str.strip()

# Step 3: Try to access the column again
print(csv_1.columns)  # Print again to verify the changes

# Now filter the DataFrame
if 'Has Table Booking' in csv_1.columns:
    csv_with_table_booking = csv_1[csv_1['Has Table Booking'] == 'Yes']
    csv_without_table_booking = csv_1[csv_1['Has Table Booking'] == 'No']
else:
    print("The column 'Has Table Booking' does not exist in the DataFrame.")
```

```
Index(['restaurant id', 'restaurant name', 'country code', 'city', 'address',
       'locality', 'locality verbose', 'longitude', 'latitude', 'cuisines',
       'average cost for two', 'currency', 'has table booking',
       'has online delivery', 'is delivering now', 'switch to order menu',
       'price range', 'aggregate rating', 'rating color', 'rating text',
       'votes'],
      dtype='object')
Index(['restaurant id', 'restaurant name', 'country code', 'city', 'address',
       'locality', 'locality verbose', 'longitude', 'latitude', 'cuisines',
       'average cost for two', 'currency', 'has table booking',
       'has online delivery', 'is delivering now', 'switch to order menu',
       'price range', 'aggregate rating', 'rating color', 'rating text',
       'votes'],
      dtype='object')
The column 'Has Table Booking' does not exist in the DataFrame.
```

In [36]:
```
csv_1.columns = csv_1.columns.str.strip()

# Print the column names to verify
print(csv_1.columns)

# Check if 'Has Table Booking' column exists
if 'Has Table Booking' in csv_1.columns:
    # Filter the DataFrame for rows with 'Yes' and 'No' in the 'Has Table Booking' column
    csv_with_table_booking = csv_1[csv_1['Has Table Booking'] == 'Yes']
    csv_without_table_booking = csv_1[csv_1['Has Table Booking'] == 'No']

    # Check if 'Aggregate rating' column exists
    if 'Aggregate rating' in csv_1.columns:
        # Calculate and print the average ratings
        print("Average Ratings : ")
        print("with table booking : ", round(csv_with_table_booking["Aggregate rating"].mean(), 2))
        print("without table booking : ", round(csv_without_table_booking["Aggregate rating"].mean(
    else:
        print("The column 'Aggregate rating' does not exist in the DataFrame.")
else:
    print("The column 'Has Table Booking' does not exist in the DataFrame.")
```

```
Index(['restaurant id', 'restaurant name', 'country code', 'city', 'address',
       'locality', 'locality verbose', 'longitude', 'latitude', 'cuisines',
       'average cost for two', 'currency', 'has table booking',
       'has online delivery', 'is delivering now', 'switch to order menu',
       'price range', 'aggregate rating', 'rating color', 'rating text',
       'votes'],
      dtype='object')
The column 'Has Table Booking' does not exist in the DataFrame.
```

3-Analyze the availability of online delivery amongewataurents with different price range

In [37]:
```
import matplotlib.pyplot as plt
```

In [41]:
```
print(csv_1.columns)

# Step 2: Strip any leading or trailing whitespace from the column names
csv_1.columns = csv_1.columns.str.strip()

# Step 3: Check if 'Price range' and 'Has Online delivery' columns exist
if 'Price range' in csv_1.columns and 'Has Online delivery' in csv_1.columns:
    # Perform the grouping and plotting
    Online_Delivery_by_price_range = csv_1.groupby('Price range')['Has Online delivery'].value_coun
    Online_Delivery_by_price_range.plot(kind='bar', stacked=True, colormap='viridis', figsize=(10,

    plt.title('Online Delivery Availability by Price Range')
    plt.xlabel('Price Range')
    plt.ylabel('Percentage')
    plt.legend(bbox_to_anchor=(1.05, 1), title='Online Delivery')
    plt.show()
else:
    print("The necessary columns do not exist in the DataFrame. Please check the column names.")
```

```
Index(['restaurant id', 'restaurant name', 'country code', 'city', 'address',
       'locality', 'locality verbose', 'longitude', 'latitude', 'cuisines',
       'average cost for two', 'currency', 'has table booking',
       'has online delivery', 'is delivering now', 'switch to order menu',
       'price range', 'aggregate rating', 'rating color', 'rating text',
       'votes'],
      dtype='object')
The necessary columns do not exist in the DataFrame. Please check the column names.
```

In [42]:
```
# Taking only those restaurents with online delivery availible

Online_delivery_Yes = csv_1[csv_1['Has Online Delivery'] == 'Yes']

# Group by price renge and calculate the percentage of restaurents wth online delivery
Online_delivery_counts = Online_delivery_Yes.groupby(['Price Range' , 'Has Online Delivery']).size(
Online_delivery_counts.plot(kind='bar',stacked=True,colormap='cividis',figsize=(10,6))

plt.title('Online Delivery Availability by price range')
plt.xlabel('Price Range')
plt.ylabel('no of restaurents ')
plt.xticks(rotation = 0)
plt.legend(title='Online Delivery',bbox_to_anchor(1.05,1))
#plt.legend(title='Online Delivery', bbox_to_anchor=(1.05, 1),loc='upper left')
plt.show()
```

```
  Cell In[42], line 13
    plt.legend(title='Online Delivery',bbox_to_anchor(1.05,1))
                                                     ^
SyntaxError: positional argument follows keyword argument
```

In [44]:
```
import pandas as pd
import matplotlib.pyplot as plt

# Assuming csv_1 is your DataFrame

# Step 1: Print the column names to verify
print(csv_1.columns)

# Step 2: Strip any leading or trailing whitespace from the column names
csv_1.columns = csv_1.columns.str.strip()

# Step 3: Check if 'Price range' and 'Has Online delivery' columns exist
if 'Price range' in csv_1.columns and 'Has Online delivery' in csv_1.columns:
    # Perform the grouping and plotting
    Online_Delivery_by_price_range = csv_1.groupby('Price range')['Has Online delivery'].value_coun
    Online_Delivery_by_price_range.plot(kind='bar', stacked=True, colormap='viridis', figsize=(10,

    plt.title('Online Delivery Availability by Price Range')
    plt.xlabel('Price Range')
    plt.ylabel('Percentage')
    plt.legend(bbox_to_anchor=(1.05, 1), title='Online Delivery')
    plt.show()
else:
    print("The necessary columns do not exist in the DataFrame. Please check the column names.")
```

```
Index(['restaurant id', 'restaurant name', 'country code', 'city', 'address',
       'locality', 'locality verbose', 'longitude', 'latitude', 'cuisines',
       'average cost for two', 'currency', 'has table booking',
       'has online delivery', 'is delivering now', 'switch to order menu',
       'price range', 'aggregate rating', 'rating color', 'rating text',
       'votes'],
      dtype='object')
The necessary columns do not exist in the DataFrame. Please check the column names.
```

-From the above 1st graph we can see that most of the restaurent do not have theonline delivery services.In price renge 1 less then 20% are available in price renge2 around 40% arw available,In price range 3 it look like 30% are available .and in price range of 4 only 10%are available.

From the above 2nd graph we analyze the people used to by from the price range2 and verey less number of people buy food from price range 4 may be because of its costliest in price compare to others

Level 2=Task 2

Task: Price Range Analysis

1-Determine the most common price range among all the restaurants.

In [46]:
```
import pandas as pd
csv_1=pd.read_csv("D:\Internship\dataset.csv")
csv_1
```

Out[46]:

	Restaurant ID	Restaurant Name	Country Code	City	Address	Locality	Locality Verbose	Longitude	Latitu
0	6317637	Le Petit Souffle	162	Makati City	Third Floor, Century City Mall, Kalayaan Avenu...	Century City Mall, Poblacion, Makati City	Century City Mall, Poblacion, Makati City, Mak...	121.027535	14.565
1	6304287	Izakaya Kikufuji	162	Makati City	Little Tokyo, 2277 Chino Roces Avenue, Legaspi...	Little Tokyo, Legaspi Village, Makati City	Little Tokyo, Legaspi Village, Makati City, Ma...	121.014101	14.553
2	6300002	Heat - Edsa Shangri-La	162	Mandaluyong City	Edsa Shangri-La, 1 Garden Way, Ortigas, Mandal...	Edsa Shangri-La, Ortigas, Mandaluyong City	Edsa Shangri-La, Ortigas, Mandaluyong City, Ma...	121.056831	14.581
3	6318506	Ooma	162	Mandaluyong City	Third Floor, Mega Fashion Hall, SM Megamall, O...	SM Megamall, Ortigas, Mandaluyong City	SM Megamall, Ortigas, Mandaluyong City, Mandal...	121.056475	14.585
4	6314302	Sambo Kojin	162	Mandaluyong City	Third Floor, Mega Atrium, SM Megamall, Ortigas...	SM Megamall, Ortigas, Mandaluyong City	SM Megamall, Ortigas, Mandaluyong City, Mandal...	121.057508	14.584
...	...	...	...	...	...	...	...	...	
9546	5915730	Naml\ Gurme	208	��stanbul	Kemanke�� Karamustafa Pa��a Mahallesi, R\ht\m ...	Karak�_y	Karak�_y, ��stanbul	28.977392	41.022
9547	5908749	Ceviz A��ac\	208	��stanbul	Ko��uyolu Mahallesi, Muhittin ��st�_nda�� Cadd...	Ko��uyolu	Ko��uyolu, ��stanbul	29.041297	41.009
9548	5915807	Huqqa	208	��stanbul	Kuru�_e��me Mahallesi, Muallim Naci Caddesi, N...	Kuru�_e��me	Kuru�_e��me, ��stanbul	29.034640	41.055
9549	5916112	A���k Kahve	208	��stanbul	Kuru�_e��me Mahallesi, Muallim Naci Caddesi, N...	Kuru�_e��me	Kuru�_e��me, ��stanbul	29.036019	41.057
9550	5927402	Walter's Coffee Roastery	208	��stanbul	Cafea��a Mahallesi, Bademalt\ Sokak, No 21/B, ...	Moda	Moda, ��stanbul	29.026016	40.984

9551 rows × 21 columns

In [47]:
```
csv_1["Price range"].value_counts()
```

Out[47]:
```
1    4444
2    3113
3    1408
4     586
Name: Price range, dtype: int64
```

In [48]:
```
most_common = csv_1["Price range"].mode()[0]
print("Most common Price among all restaurents : ", most_common )
```

```
Most common Price among all restaurents :  1
```

2- Calculate the avarage rating of each price range,

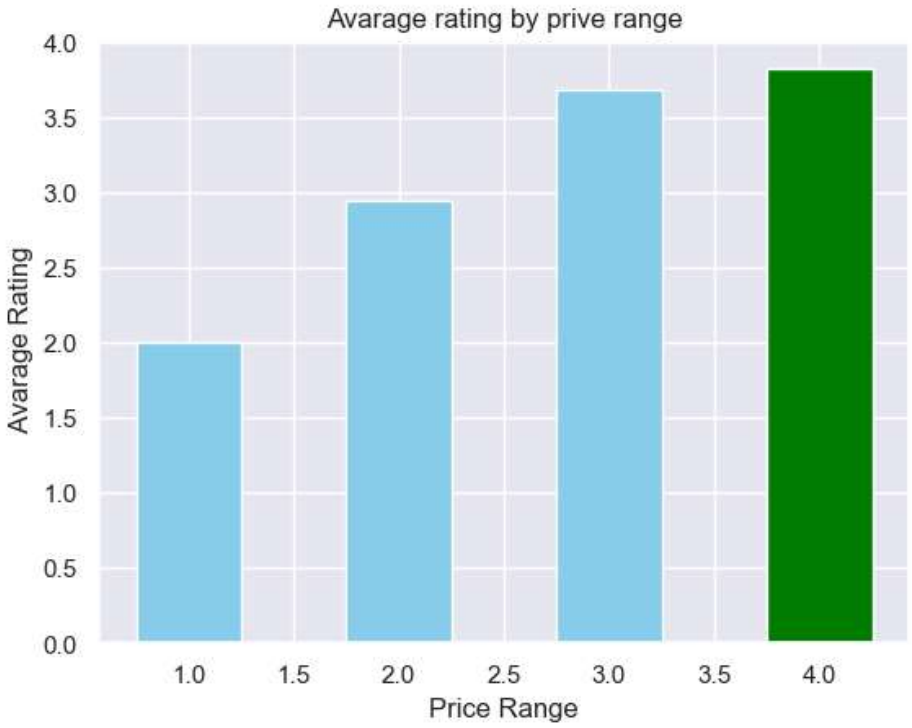
& identify the color that represent then highest avarage ratingamong different price range.

In [49]:
```
Avarage_Rating_by_price_range = csv_1.groupby('Price range')['Aggregate rating'].mean().round(2)

print("Avarage Rating for each  price range :")
print(Avarage_Rating_by_price_range)
```

```
Avarage Rating for each  price range :
Price range
1    2.00
2    2.94
3    3.68
4    3.82
Name: Aggregate rating, dtype: float64
```

In [50]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns;sns.set(color_codes=True)
%matplotlib inline
```

In [53]:
```
# find the price range with highest avarage rating
highest_avg_rating_color = Avarage_Rating_by_price_range.idxmax()
plt.bar(Avarage_Rating_by_price_range.index, Avarage_Rating_by_price_range,color='skyblue' ,width=0
plt.bar(highest_avg_rating_color , Avarage_Rating_by_price_range[highest_avg_rating_color] , color=

plt.xlabel('Price Range')
plt.ylabel(' Avarage Rating')
plt.title('Avarage rating by prive range')

plt.show()
```



Price range 4 get the highest avarage rating,which is 3.82 followed by price range 3,2,1.

Level 2-Task 3:-

1. Extract additional features from the existing columns, such as the length of the restaurant name or address.

In [54]:
```
csv_1['Restaurant Name Length'] = csv_1['Restaurant Name'].apply(lambda x: len(str(x)))
csv_1['Address Length'] = csv_1['Address'].apply(lambda x: len(str(x)))
```

In [55]:
```
csv_1[['Restaurant Name' ,'Restaurant Name Length','Address' , 'Address Length']]
```

Out[55]:

	Restaurant Name	Restaurant Name Length	Address	Address Length
0	Le Petit Souffle	16	Third Floor, Century City Mall, Kalayaan Avenu...	71
1	Izakaya Kikufuji	16	Little Tokyo, 2277 Chino Roces Avenue, Legaspi...	67
2	Heat - Edsa Shangri-La	22	Edsa Shangri-La, 1 Garden Way, Ortigas, Mandal...	56
3	Ooma	4	Third Floor, Mega Fashion Hall, SM Megamall, O...	70
4	Sambo Kojin	11	Third Floor, Mega Atrium, SM Megamall, Ortigas...	64
...	...	...	...	...
9546	Naml\ Gurme	11	Kemanke�� Karamustafa Pa��a Mahallesi, R\ht\m ...	103
9547	Ceviz A��ac\	12	Ko��uyolu Mahallesi, Muhittin ��st�_nda�� Cadd...	77
9548	Huqqa	5	Kuru�_e��me Mahallesi, Muallim Naci Caddesi, N...	73
9549	A���k Kahve	11	Kuru�_e��me Mahallesi, Muallim Naci Caddesi, N...	75
9550	Walter's Coffee Roastery	24	Cafea��a Mahallesi, Bademalt\ Sokak, No 21/B, ...	65

9551 rows × 4 columns

2-Create new features like "Has Table Booking" or "Has Online Delivery" by encoding categorical variables

In [56]:
```
csv_1['Has Table Booking'] = csv_1['Has Table booking'].apply(lambda x:  1 if x == 'Yes' else 0)
csv_1['Has Online Delivery'] = csv_1['Has Online delivery'].apply(lambda x: 1 if x == 'Yes' else 0)
```

In [57]:
```
csv_1[['Has Table Booking' ,'Has Table booking','Has Online Delivery', 'Has Online delivery']]
```

Out[57]:

	Has Table Booking	Has Table booking	Has Online Delivery	Has Online delivery
0	1	Yes	0	No
1	1	Yes	0	No
2	1	Yes	0	No
3	0	No	0	No
4	1	Yes	0	No
...	...	...	...	...
9546	0	No	0	No
9547	0	No	0	No
9548	0	No	0	No
9549	0	No	0	No
9550	0	No	0	No

9551 rows × 4 columns

-Two new column added 'Restaurant Name Length' and 'Adress Length' from the length of n the restaurant name or adress.

-And also two new binary column added by encoding categorical variable 'Has Table Booking' and 'Has Online Delivery'.

Observation:-

-Percentage of restaurants offer table booking is 12.12% and percentage of restaurants offer online delivery is 25.66%.

-Avarage rating with table booking is 3.44 and without table booking is 2.56.

-Most of the restaurants have do not online delivery service. In price range1 less then 20% are available in price range 2 around 40% are available. in price range 3 it looklike 30% are available in price range 4 only 10% are available.

-People mostly buy from price range 2 and very less number of people buy food from price range 4 may be because of its costliest in price compare to others.

- Most common price range among all restaurants are1.

- Price range 4 get the highest avarage rating which is 3.83 followed by price range 3,2,1.

Level-3

Task 1: Predictive Modeling

Build a regression model to predict the aggregate rating of a restaurant based on available features.

Split the dataset into training and testing sets and evaluate the model's performance using appropriate metrics.

Experiment with different algorithms (e.g., linear regression, decision trees, random forest) and compare their performance

Task 2 : Customer Preference Analysis

Analyze the relationship between the type of cuisine and the restaurant's rating.

Identify the most popular cuisines among customers based on the number of votes.

Determine if there are any specific cuisines that tend to receive higher ratings.

Task 3: Data Visualization

Create visualizations to represent the distribution of ratings using different charts (histogram, bar plot, etc.).

Compare the average ratings of different cuisines or cities using appropriate visualizations.

Visualize the relationship between various features and the target variable to gain insights.

In [58]:
```
# Importing the warnings

import warnings
warnings.filterwarnings("ignore")
```

In [59]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns;sns.set(color_codes=True)
%matplotlib inline
```

Level 3- Task 1: Predictive Modeling

1. Build a regression model to predict the aggregate rating of a restaurant based on available features.

&Split the dataset into training and testing sets and evaluate the model's performance using appropriate metrics.¶

In [60]:
```
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, r2_score
```

In [61]:
```
# Convert categorical variable to numeric

# csv_1 = pd.get_dummies(csv_1, columns=['Has Table booking' , 'Has Online delivery'], drop_first=Tr

csv_1 = pd.get_dummies(csv_1, columns=['Has Table booking', 'Has Online delivery'], drop_first=True
```

In [62]:
```
# # Select features and target variable
x = csv_1[['Average Cost for two', 'Votes', 'Price range', 'Has Table booking_Yes', 'Has Online del
y = csv_1['Aggregate rating']
```

In [63]:
```
# Split the dataset into training and test sets( 80% training and 20% testing)

x_train,x_test,y_train,y_test = train_test_split(x,y,test_size=0.2, random_state=42)
```

In [64]:
```
# Initialize and train the  Linear regression Model
model = LinearRegression()
model.fit(x_train, y_train)

# Predict ratings on the testing set

y_pred = model.predict(x_test)

# Evaluate the models performance
mse = mean_squared_error(y_test,y_pred)
r2 = r2_score(y_test,y_pred)

print("Model: Linear Regression")
print("Mean Squared Error (MSE):",mse)
print("R-squared (R2) Score:",r2)
```

```
Model: Linear Regression
Mean Squared Error (MSE): 1.6764802747031446
R-squared (R2) Score: 0.26344464090219477
```

2-Experiment with different algorithms (e.g., linear regression, decision trees, random forest) and compare their performance

In [65]:
```
from sklearn.tree import DecisionTreeRegressor
from sklearn.ensemble import RandomForestRegressor
```

In [66]:
```
# Initialize and train  different regression model

models = {
    'Linear REgression': LinearRegression(),

    'Decision Tree':DecisionTreeRegressor(random_state=42),
    'Random Forest' : RandomForestRegressor(random_state=42)

}

# Evaluate models

results = {}
for name, model in models.items():
    model.fit(x_train,y_train)
    y_pred = model.predict(x_test)
    mse = mean_squared_error(y_test,y_pred)
    r2 = r2_score(y_test,y_pred)
    results[name] = {'MSE':mse, 'R2 Score' : r2}

        # Diaplay results
#result_csv_1 = csv_1.DataFrame(results)
results_csv_1 = pd.DataFrame(results)
print(results_csv_1)
```

```
          Linear REgression  Decision Tree  Random Forest
MSE                1.676480       0.203498       0.133938
R2 Score           0.263445       0.910594       0.941155
```

-Linear Regression produced an MSE of 1.6765 and an R-squared value is 0.2634.

-Decision tree yielded an MSE of 0.2074 and an R-squared value of 0.9089

- Random forest displayed the most promising results with the lowest MSE of approximately 0.1337 and the highest R squred of about 0.9413

Level -3:Task 2

Task: Customer Preference Analysis

1. Analyze the relationship between the type of cuisine and the restaurant's rating.

In [67]:
```
cuisines = csv_1['Cuisines']
```

In [68]:
```
cuisines.value_counts().head(10)
```

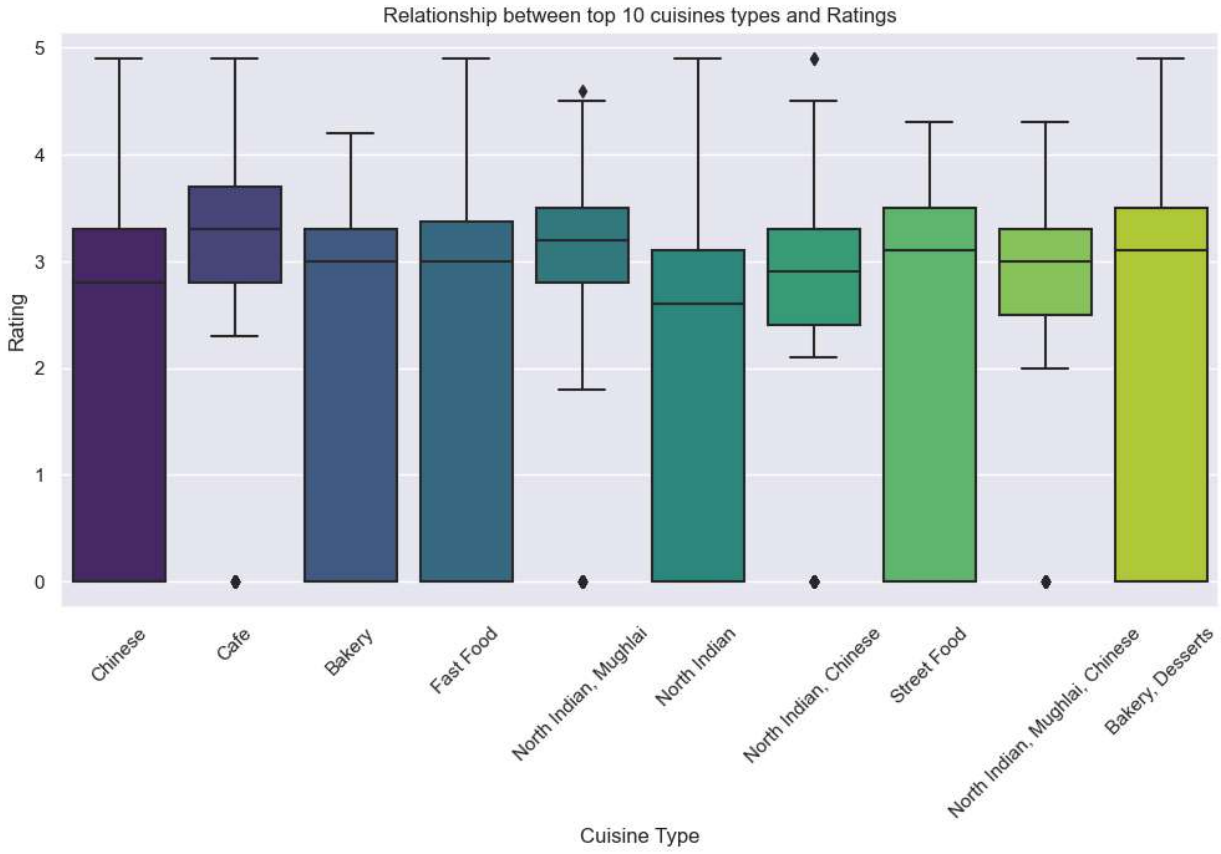
Out[68]:
```
North Indian                      936
North Indian, Chinese             511
Chinese                           354
Fast Food                         354
North Indian, Mughlai             334
Cafe                              299
Bakery                            218
North Indian, Mughlai, Chinese    197
Bakery, Desserts                  170
Street Food                       149
Name: Cuisines, dtype: int64
```

In [69]:
```
# Get the top 10 most common cuisines

top_10_cuisines = cuisines.value_counts().head(10).index
```

In [70]:
```
# Create a dataframe with cuisines ypes and currosponding Rating

cuisine_rating = pd.DataFrame({'Cuisine':cuisines,'Rating': csv_1['Aggregate rating']})
```

In [71]:
```
# Filter cuisines rating dataframe to include only  the top 10 cuisines
cuisine_ratings_top_10 = cuisine_rating[cuisine_rating['Cuisine'].isin(top_10_cuisines)]
```

In [72]:
```
# Plot the relationship between top 20 cusines types nd rating

plt.figure(figsize=(12,6))
sns.boxplot(x='Cuisine', y='Rating',data=cuisine_ratings_top_10,palette='viridis')
plt.title('Relationship between top 10 cuisines types and Ratings')
plt.xlabel('Cuisine Type')
plt.ylabel('Rating')
plt.xticks(rotation=45)
plt.show()
```



2- Identify the most popular cuisines among customers based on the number of votes.

In [73]:
```
# Create  a dataframe with the cuisine types and currospondng votes
cuisine_votes = pd.DataFrame({'Cuisine':cuisines,'Votes': csv_1['Votes']})
```

In [74]:
```
# Groupby cuisind and sum the votes for each cuisines
cusine_votes_sum = cuisine_votes.groupby('Cuisine')['Votes'].sum()
```

In [75]:
```
import pandas as pd

# Example: Replace this with your actual data loading code
# csv_1 = pd.read_csv('your_data.csv')

# Check column names to ensure they are correct
print(csv_1.columns)

# Remove any leading/trailing whitespace from column names
csv_1.columns = csv_1.columns.str.strip()

# Step 1: Group by Cuisines and sum the votes
cuisine_votes_sum = csv_1.groupby('Cuisines')['Votes'].sum().reset_index()
cuisine_votes_sum.columns = ['Cuisine', 'Total Votes']

# Step 2: Sort cuisines based on the total votes in descending order
popular_cuisines = cuisine_votes_sum.sort_values(by='Total Votes', ascending=False)

# Step 3: Display the sorted DataFrame
print(popular_cuisines)
```

```
Index(['Restaurant ID', 'Restaurant Name', 'Country Code', 'City', 'Address',
       'Locality', 'Locality Verbose', 'Longitude', 'Latitude', 'Cuisines',
       'Average Cost for two', 'Currency', 'Is delivering now',
       'Switch to order menu', 'Price range', 'Aggregate rating',
       'Rating color', 'Rating text', 'Votes', 'Restaurant Name Length',
       'Address Length', 'Has Table Booking', 'Has Online Delivery',
       'Has Table booking_Yes', 'Has Online delivery_Yes'],
      dtype='object')
                                            Cuisine  Total Votes
1514                            North Indian, Mughlai        53747
1306                                    North Indian        46241
1329                            North Indian, Chinese        42012
331                                            Cafe         30657
497                                         Chinese         21925
...                                             ...           ...
1252  Mithai, North Indian, South Indian, Chinese, S...            0
489                                  Cafe, Tibetan              0
524                       Chinese, Fast Food, Pizza              0
527                               Chinese, Italian               0
1687                Seafood, Mughlai, North Indian               0

[1825 rows x 2 columns]
```

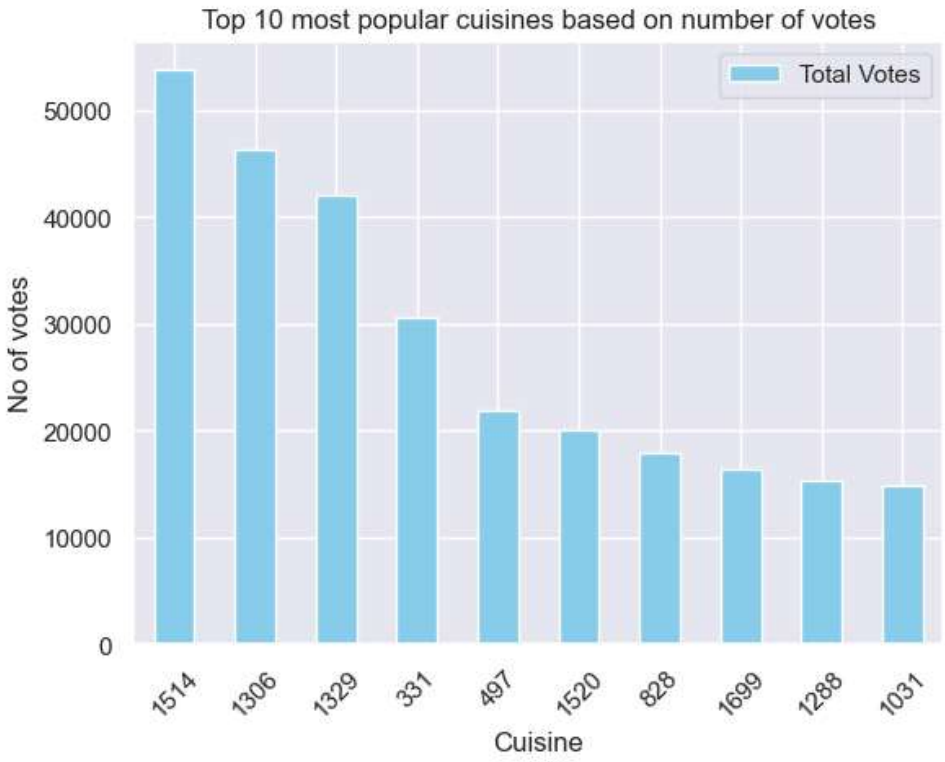
In [76]:
```
poular_cuisines = cuisine_votes_sum.sort_values(by='Total Votes', ascending=False)
```

In [77]:
```
print(popular_cuisines.head(10))
```

```
                           Cuisine  Total Votes
1514          North Indian, Mughlai        53747
1306                  North Indian         46241
1329          North Indian, Chinese        42012
331                          Cafe          30657
497                       Chinese         21925
1520  North Indian, Mughlai, Chinese        20115
828                     Fast Food         17852
1699                  South Indian         16433
1288          Mughlai, North Indian        15275
1031                       Italian         14799
```

In [78]:
```
# Plotting the bar plot
plt.figure(figsize=(10,6))

popular_cuisines.head(10).plot(kind='bar',color='skyblue')

plt.title('Top 10 most popular cuisines based on number of votes')
plt.xlabel('Cuisine')
plt.ylabel('No of votes')
plt.xticks(rotation=45)
plt.show()
```

```
<Figure size 1000x600 with 0 Axes>
```



3-Determine if there are any specific cuisines that tend to receive higher ratings

In [79]:
```
#Create a DataFrame with cuisine types and currosponding ratings

cuisine_rating = pd.DataFrame({'Cuisine' : cuisines, 'Rating' : csv_1['Aggregate rating']})
```

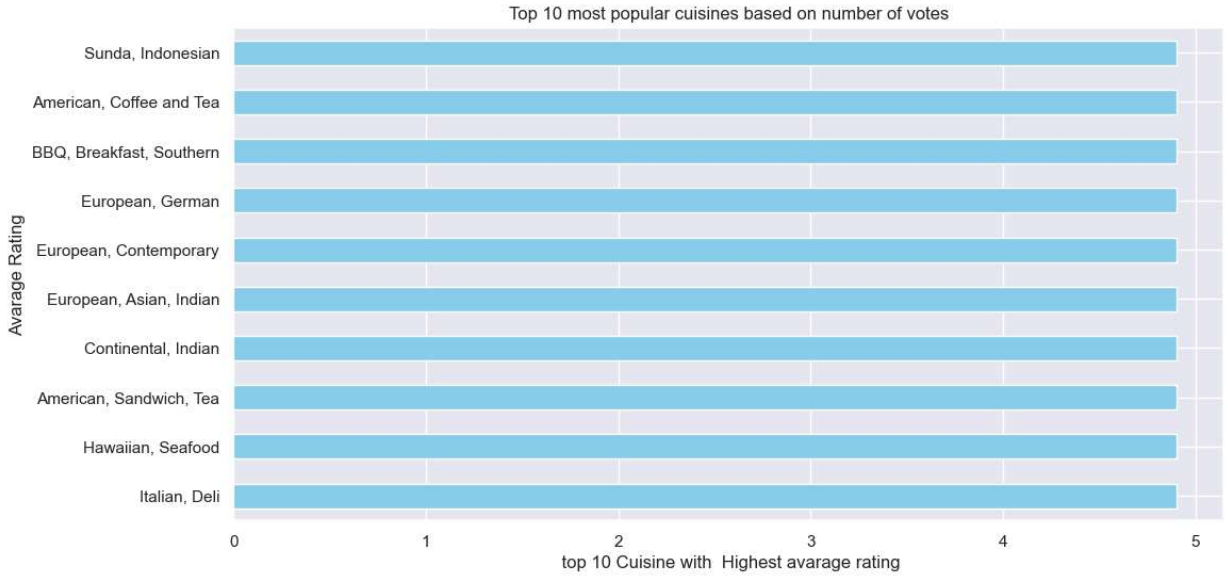
In [80]:
```
# CAlcutlate thge avarage rating for each cuisines

avarage_rating_by_cuisine  = cuisine_rating.groupby('Cuisine')['Rating'].mean()
```

In [81]:
```
# Sort cuisine based on the avarage rating  in descending order
sorted_cuisines_by_rating = avarage_rating_by_cuisine.sort_values(ascending=False)
```

In [82]:
```
# Display the top 10 cuisine with highest avarage rating

print("Top 10 cuisine with highest avarage rating :")
print(sorted_cuisines_by_rating.head(10))
```

```
Top 10 cuisine with highest avarage rating :
Cuisine
Italian, Deli               4.9
Hawaiian, Seafood           4.9
American, Sandwich, Tea     4.9
Continental, Indian         4.9
European, Asian, Indian     4.9
European, Contemporary      4.9
European, German            4.9
BBQ, Breakfast, Southern    4.9
American, Coffee and Tea    4.9
Sunda, Indonesian           4.9
Name: Rating, dtype: float64
```

In [83]:
```
# Plot the graph

plt.figure(figsize=(12,6))

sorted_cuisines_by_rating.head(10).plot(kind='barh',color='skyblue')

plt.title('Top 10 most popular cuisines based on number of votes')
plt.xlabel(' top 10 Cuisine with  Highest avarage rating')
plt.ylabel('Avarage Rating')

plt.show()
```



In [ ]:
```
```