Level-1

Task 1:

Data Exploration and Preprocessing

In [1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
import warnings
warnings.filterwarnings('ignore')
```

In [2]:
```
file_path="C:\\Users\\prade\\Downloads\\Dataset .csv"
df=pd.read_csv(file_path)
df
```

Out[2]:

	Restaurant ID	Restaurant Name	Country Code	City	Address	Locality	
0	6317637	Le Petit Souffle	162	Makati City	Third Floor, Century City Mall, Kalayaan Avenu...	Century City Mall, Poblacion, Makati City	Mal
1	6304287	Izakaya Kikufuji	162	Makati City	Little Tokyo, 2277 Chino Roces Avenue, Legaspi...	Little Tokyo, Legaspi Village, Makati City	Leg Maka
2	6300002	Heat - Edsa Shangri-La	162	Mandaluyong City	Edsa Shangri-La, 1 Garden Way, Ortigas, Mandal...	Edsa Shangri-La, Ortigas, Mandaluyong City	Edsa M
3	6318506	Ooma	162	Mandaluyong City	Third Floor, Mega Fashion Hall, SM Megamall, O...	SM Megamall, Ortigas, Mandaluyong City	SM M Ci
4	6314302	Sambo Kojin	162	Mandaluyong City	Third Floor, Mega Atrium, SM Megamall, Ortigas...	SM Megamall, Ortigas, Mandaluyong City	SM M Ci
...	...	...	...	...	...	...	
9546	5915730	Naml\ Gurme	208	��stanbul	Kemanke�� Karamustafa Pa��a Mahallesi, R\ht\m ...	Karak�_y	
9547	5908749	Ceviz A��ac\	208	��stanbul	Ko��uyolu Mahallesi, Muhittin ��st�_nda�� Cadd...	Ko��uyolu	K
9548	5915807	Huqqa	208	��stanbul	Kuru�_e��me Mahallesi, Muallim Naci Caddesi, N...	Kuru�_e��me	Kuru
9549	5916112	A���k Kahve	208	��stanbul	Kuru�_e��me Mahallesi, Muallim Naci Caddesi, N...	Kuru�_e��me	Kuru
9550	5927402	Walter's Coffee Roastery	208	��stanbul	Cafea��a Mahallesi, Bademalt\ Sokak, No 21/B, ...	Moda	

9551 rows × 21 columns

In [3]:
```
df.head()
```

Out[3]:

	Restaurant ID	Restaurant Name	Country Code	City	Address	Locality	Locality Verbose	Lo
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.
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.
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.
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.
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.

5 rows × 21 columns

In [4]:
```
df.tail()
```

Out[4]:

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

5 rows × 21 columns

In [6]:
```
df.shape
```

Out[6]: (9551, 21)

In [7]:
```
df.size
```

Out[7]: 200571

In [8]:
```
print("Number of rows are: ",df.shape[0])
print("Number of columns are: ",df.shape[1])
```

```
Number of rows are:  9551
Number of columns are:  21
```

In [9]:
```
Duplicate=df.duplicated().sum()
Duplicate
```

Out[9]: 0

In [10]:
```
df.isnull()
```

Out[10]:

	Restaurant ID	Restaurant Name	Country Code	City	Address	Locality	Locality Verbose	Longitude	Latitu
0	False	False	False	False	False	False	False	False	Fal
1	False	False	False	False	False	False	False	False	Fal
2	False	False	False	False	False	False	False	False	Fal
3	False	False	False	False	False	False	False	False	Fal
4	False	False	False	False	False	False	False	False	Fal
...	...	...	...	...	...	...	...	...	
9546	False	False	False	False	False	False	False	False	Fal
9547	False	False	False	False	False	False	False	False	Fal
9548	False	False	False	False	False	False	False	False	Fal
9549	False	False	False	False	False	False	False	False	Fal
9550	False	False	False	False	False	False	False	False	Fal

9551 rows × 21 columns

In [11]:
```
df.isnull().sum()
```

Out[11]:
```
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
```

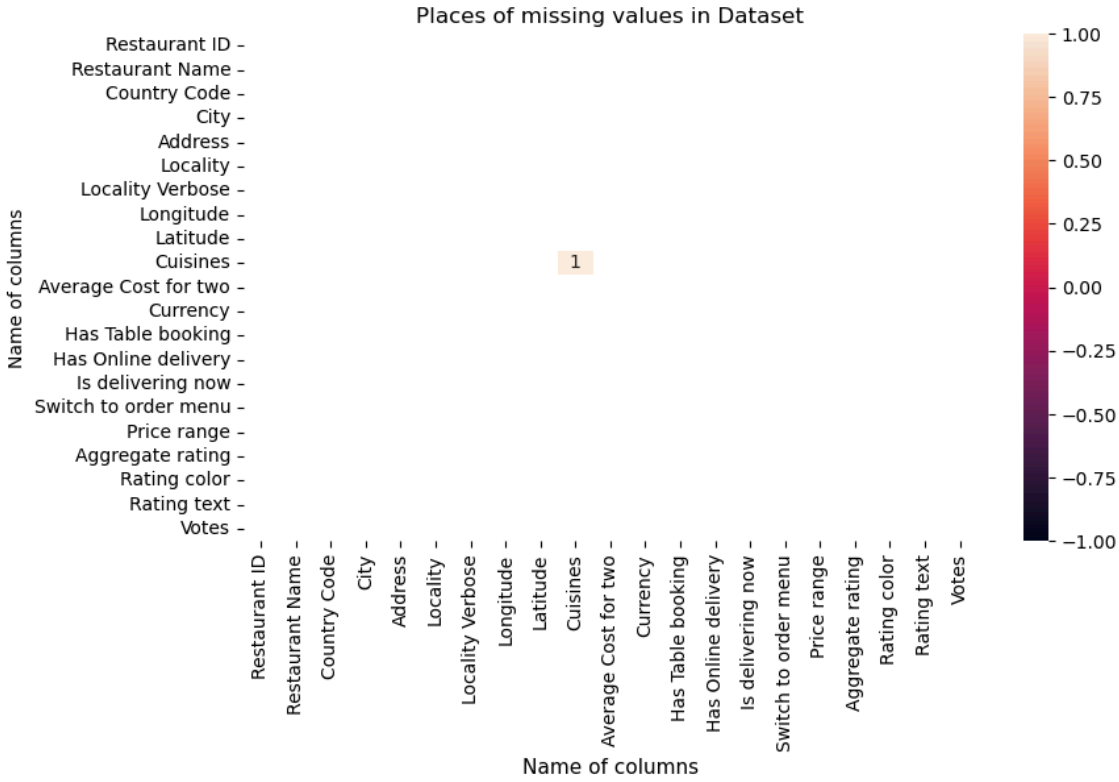
In [12]:
```
# Visualizing the missing values
# Checking Null Value by Plotting Heatmap

plt.figure(figsize = (9,5))

sns.heatmap(df.isnull().corr(), vmin=-1, annot= True)

plt.xlabel('Name of columns', fontsize=11)
plt.ylabel('Name of columns', fontsize=10)
plt.title('Places of missing values in Dataset', fontsize=12)


plt.show()
```



Handling Missing Values

In [13]:
```
dp=df.dropna(subset=['Cuisines'])
```

In [14]:
```
print("Missing values/null values count after handling:")
df.isna().sum()
```

```
Missing values/null values count after handling:
```

Out[14]:
```
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
```

Data Type Conversion

In [15]:
```
df.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              9542 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
```

Distribution of The Target Variable

In [16]:
```
target_counts = df['Aggregate rating'].value_counts()
print("Distribution of target variable:")
print(target_counts)
```

```
Distribution of target variable:
Aggregate rating
0.0    2148
3.2     522
3.1     519
3.4     498
3.3     483
3.5     480
3.0     468
3.6     458
3.7     427
3.8     400
2.9     381
3.9     335
2.8     315
4.1     274
4.0     266
2.7     250
4.2     221
2.6     191
4.3     174
4.4     144
2.5     110
4.5      95
2.4      87
4.6      78
4.9      61
2.3      47
4.7      42
2.2      27
4.8      25
2.1      15
2.0       7
1.9       2
1.8       1
Name: count, dtype: int64
```

Task-2:Descriptive Analysis

In [17]:
```
numeric_columns = df.select_dtypes(include=['int', 'float'])

# Calculate basic statistical measures using .describe()
summary_stats = numeric_columns.describe()
print(summary_stats)
```

```
       Restaurant ID  Country Code    Longitude     Latitude  \
count   9.551000e+03   9551.000000  9551.000000  9551.000000
mean    9.051128e+06     18.365616    64.126574    25.854381
std     8.791521e+06     56.750546    41.467058    11.007935
min     5.300000e+01      1.000000  -157.948486   -41.330428
25%     3.019625e+05      1.000000    77.081343    28.478713
50%     6.004089e+06      1.000000    77.191964    28.570469
75%     1.835229e+07      1.000000    77.282006    28.642758
max     1.850065e+07    216.000000   174.832089    55.976980

       Average Cost for two  Price range  Aggregate rating         Votes
count           9551.000000  9551.000000       9551.000000   9551.000000
mean            1199.210763     1.804837          2.666370    156.909748
std            16121.183073     0.905609          1.516378    430.169145
min                0.000000     1.000000          0.000000      0.000000
25%              250.000000     1.000000          2.500000      5.000000
50%              400.000000     2.000000          3.200000     31.000000
75%              700.000000     2.000000          3.700000    131.000000
max           800000.000000     4.000000          4.900000  10934.000000
```

In [20]:
```
median = numeric_columns.median()
print(f"\nMedian for numerical columns:\n{median}")
```

```
Median for numerical columns:
Restaurant ID           6.004089e+06
Country Code            1.000000e+00
Longitude               7.719196e+01
Latitude                2.857047e+01
Average Cost for two    4.000000e+02
Price range             2.000000e+00
Aggregate rating        3.200000e+00
Votes                   3.100000e+01
dtype: float64
```

In [21]:
```
# Calculate standard deviation for numerical columns
std_dev = numeric_columns.std()
print(f"\nStandard deviation for numerical columns:\n{std_dev}")
```

```
Standard deviation for numerical columns:
Restaurant ID           8.791521e+06
Country Code            5.675055e+01
Longitude               4.146706e+01
Latitude                1.100794e+01
Average Cost for two    1.612118e+04
Price range             9.056088e-01
Aggregate rating        1.516378e+00
Votes                   4.301691e+02
dtype: float64
```

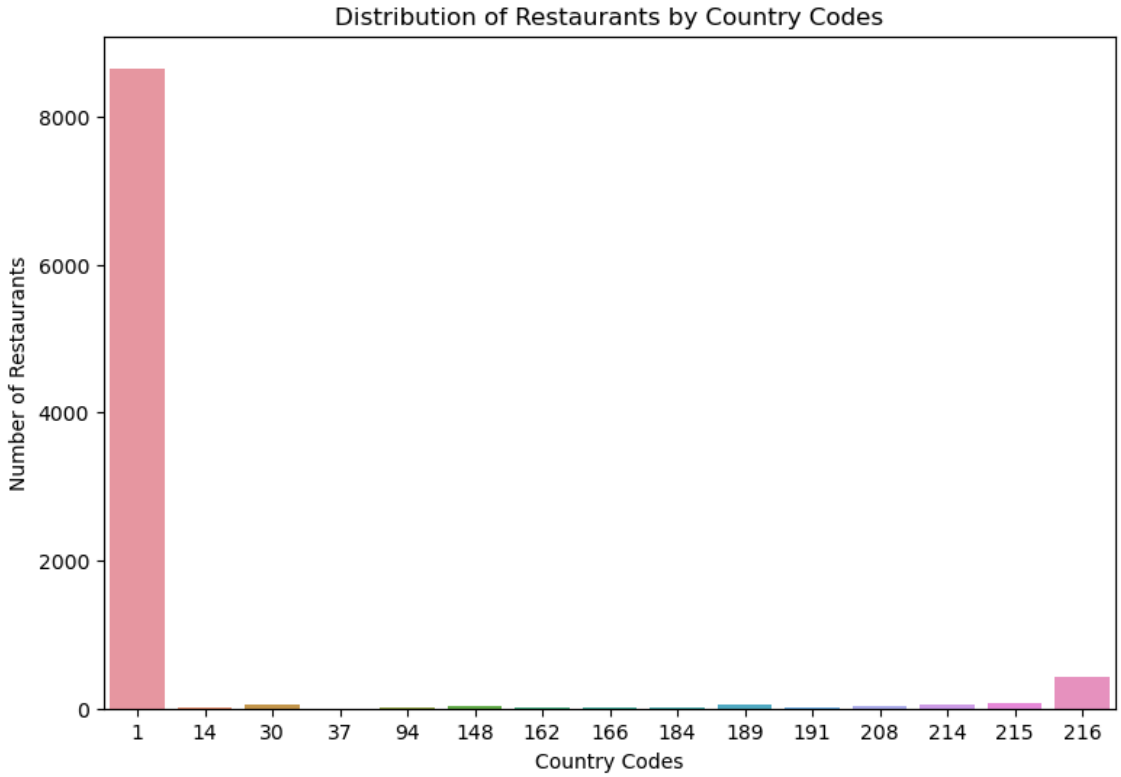
Distribution of Categorical Variables

In [22]:
```
plt.figure(figsize=(9, 6))


sns.countplot(x = df['Country Code'])


plt.xlabel('Country Codes')
plt.ylabel('Number of Restaurants')
plt.title('Distribution of Restaurants by Country Codes')


plt.show()
```

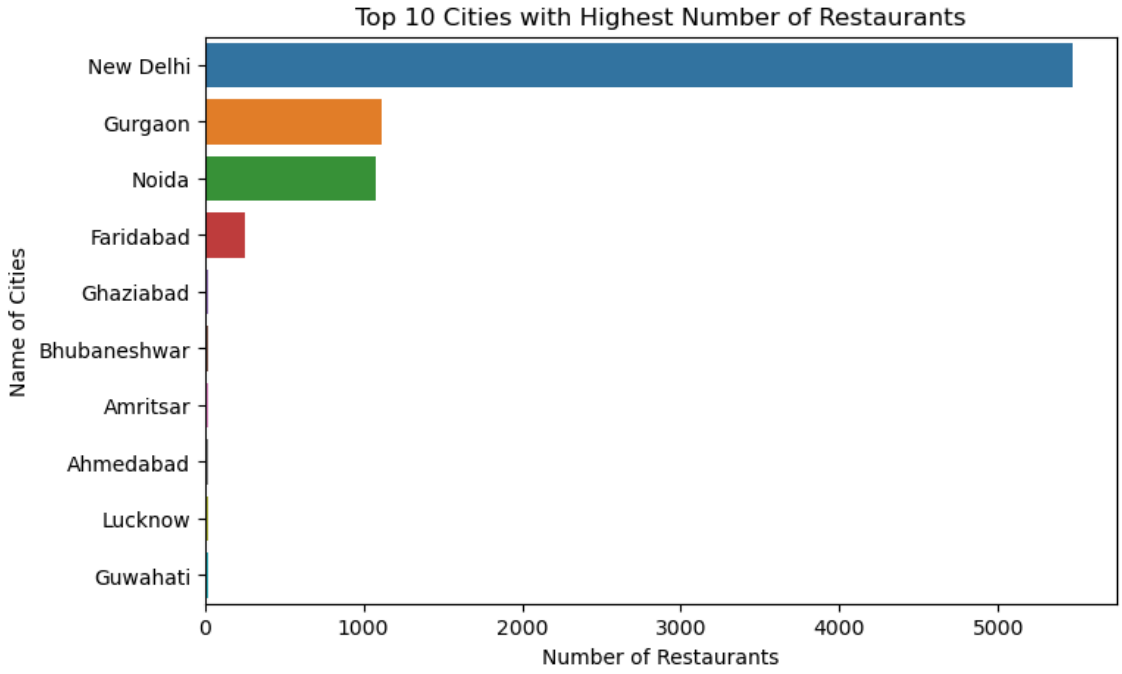


In [23]:
```
plt.figure(figsize=(8, 5))


sns.countplot(y = df['City'], order=df.City.value_counts().iloc[:10].index)


plt.xlabel('Number of Restaurants')
plt.ylabel('Name of Cities')
plt.title('Top 10 Cities with Highest Number of Restaurants')


plt.show()
```

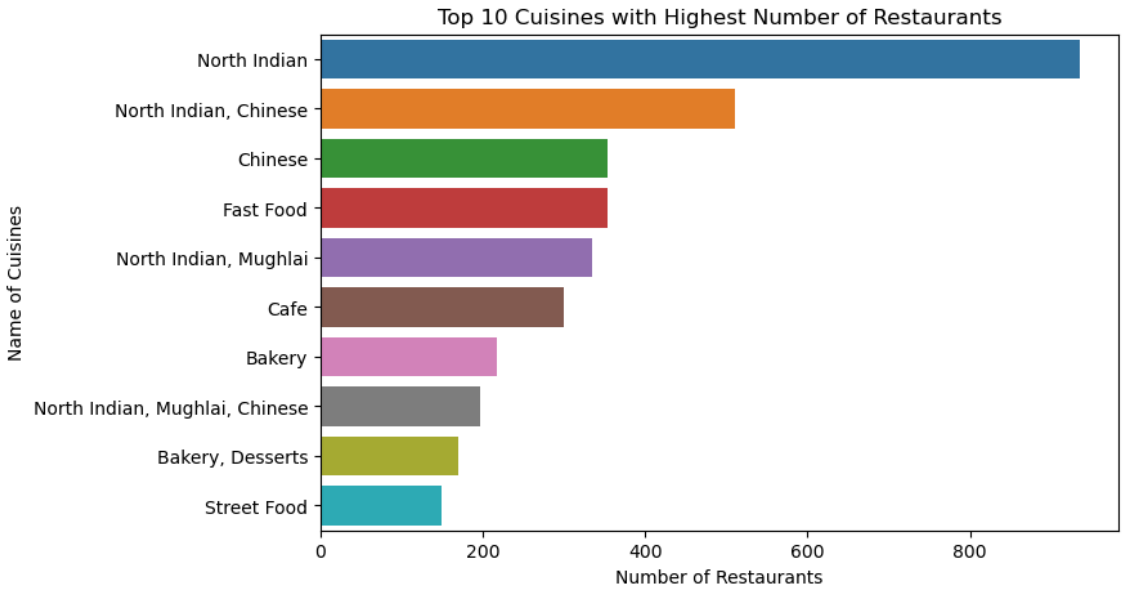


In [24]:
```
plt.figure(figsize=(8, 5))

# Create the figure object
# There are many cuisine names present in the data, so i select only the to
sns.countplot(y = df['Cuisines'], order=df.Cuisines.value_counts().iloc[:10

# Set Labels
plt.xlabel('Number of Restaurants')
plt.ylabel('Name of Cuisines')
plt.title('Top 10 Cuisines with Highest Number of Restaurants')

plt.show()
```



Top Cuisines and Cities

In [25]:
```
# Identify the top 10 cuisines
top_cuisines = df['Cuisines'].value_counts().head(10)

# Display the results
print("Top 10 Cuisines with Highest Number of Restaurants:")
print(top_cuisines)
```

```
Top 10 Cuisines with Highest Number of Restaurants:
Cuisines
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: count, dtype: int64
```

In [26]:
```
# Identify the top 10 cities
top_cities = df['City'].value_counts().head(10)

# Display the results
print("Top 10 Cities with Highest Number of Restaurants:")
print(top_cities)
```

```
Top 10 Cities with Highest Number of Restaurants:
City
New Delhi       5473
Gurgaon         1118
Noida           1080
Faridabad        251
Ghaziabad         25
Bhubaneshwar      21
Amritsar          21
Ahmedabad         21
Lucknow           21
Guwahati          21
Name: count, dtype: int64
```

Task-3: Geospatial Analysis

Visualize Locations of Restaurants

In [28]:
```
pip install shapely
```

```
Collecting shapelyNote: you may need to restart the kernel to use updated
packages.

  Obtaining dependency information for shapely from https://files.pythonh
osted.org/packages/29/cd/763817c27e6cb6d04ffd477a5dcdfdd71bc3fb640f5748c9
f2c1cd08ba52/shapely-2.0.3-cp311-cp311-win_amd64.whl.metadata (https://fi
les.pythonhosted.org/packages/29/cd/763817c27e6cb6d04ffd477a5dcdfdd71bc3f
b640f5748c9f2c1cd08ba52/shapely-2.0.3-cp311-cp311-win_amd64.whl.metadata)
  Downloading shapely-2.0.3-cp311-cp311-win_amd64.whl.metadata (7.2 kB)
Requirement already satisfied: numpy<2,>=1.14 in c:\users\prade\anaconda3
\lib\site-packages (from shapely) (1.24.3)
Downloading shapely-2.0.3-cp311-cp311-win_amd64.whl (1.4 MB)
   ---------------------------------------- 0.0/1.4 MB ? eta -:--:--
   ------------ ------------------------- 0.5/1.4 MB 10.5 MB/s eta 0:0
0:01
   ---------------------- ---------------- 0.8/1.4 MB 8.9 MB/s eta 0:00:
01
   --------------------------------- ------ 1.2/1.4 MB 8.5 MB/s eta 0:00:
01
   ---------------------------------------  1.4/1.4 MB 8.3 MB/s eta 0:00:
01
   ---------------------------------------- 1.4/1.4 MB 7.0 MB/s eta 0:00:
00
Installing collected packages: shapely
Successfully installed shapely-2.0.3
```

In [30]:
```
pip install geopandas
```

```
e\anaconda3\lib\site-packages (from pandas>=1.4.0->geopandas) (2.8.2)
Requirement already satisfied: pytz>=2020.1 in c:\users\prade\anaconda
3\lib\site-packages (from pandas>=1.4.0->geopandas) (2023.3.post1)
Requirement already satisfied: tzdata>=2022.1 in c:\users\prade\anacon
da3\lib\site-packages (from pandas>=1.4.0->geopandas) (2023.3)
Requirement already satisfied: numpy>=1.21.0 in c:\users\prade\anacond
a3\lib\site-packages (from pandas>=1.4.0->geopandas) (1.24.3)
Requirement already satisfied: colorama in c:\users\prade\anaconda3\li
b\site-packages (from click~=8.0->fiona>=1.8.21->geopandas) (0.4.6)
Downloading geopandas-0.14.3-py3-none-any.whl (1.1 MB)
   ---------------------------------------- 0.0/1.1 MB ? eta -:--:--
   -------------- ------------------------- 0.4/1.1 MB 8.7 MB/s eta 0:
00:01
   --------------------------- ------------ 0.7/1.1 MB 9.5 MB/s eta 0:
00:01
   ---------------------------------------  1.1/1.1 MB 7.8 MB/s eta 0:
00:01
   ---------------------------------------- 1.1/1.1 MB 7.0 MB/s eta 0:
00:00
Downloading fiona-1.9.5-cp311-cp311-win_amd64.whl (22.9 MB)
```

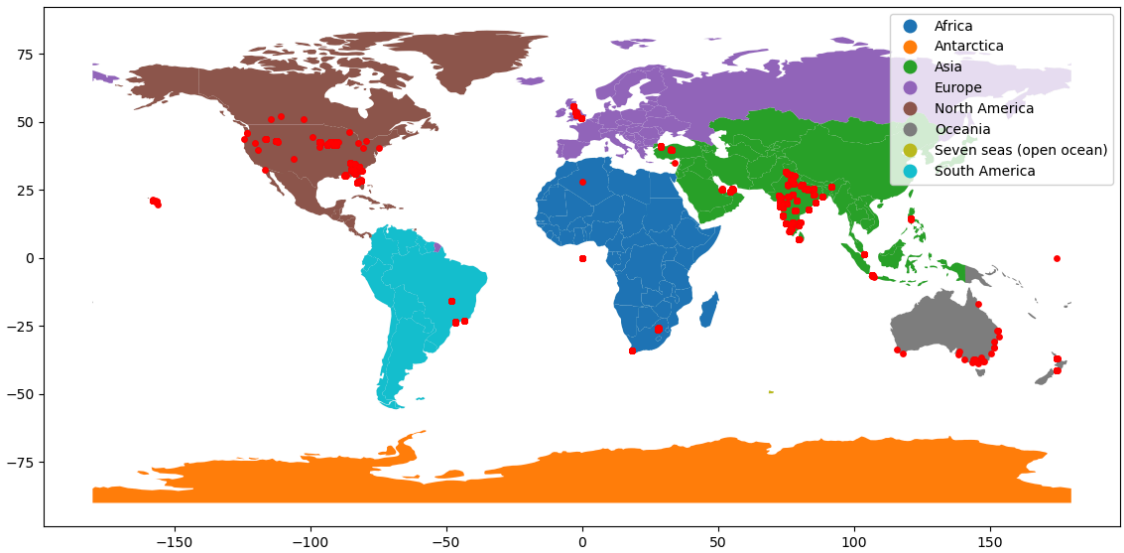
In [31]:
```
from shapely.geometry import Point
import geopandas as gpd
from geopandas import GeoDataFrame

# Create Point geometry from latitude and longitude using Shapely
gdf = gpd.GeoDataFrame(
    df,
    geometry=gpd.points_from_xy(df.Longitude, df.Latitude)
)

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

# Create a map that fits the screen and plots the restaurant locations
# The "continent" column is used for coloring and a legend is displayed
gdf.plot(ax=world.plot("continent", legend = True, figsize=(14, 12)), marke

# Show the map
plt.show()
```



Correlation Between the Restaurant's Location and its Rating

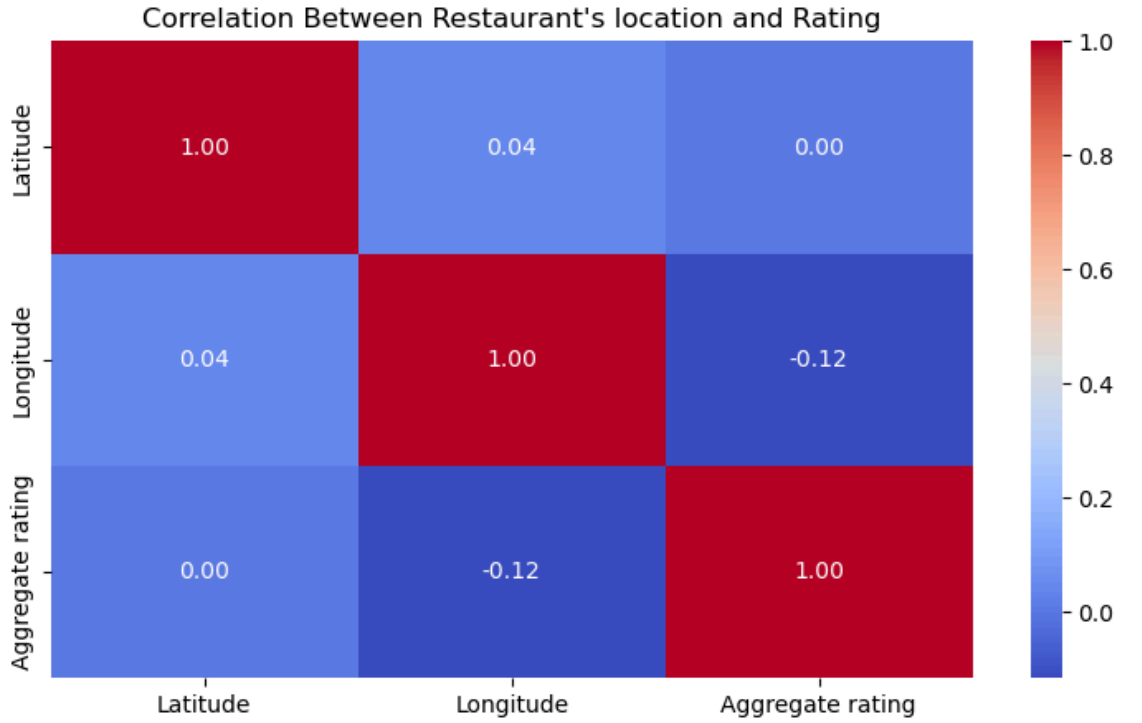
In [32]:
```
plt.figure(figsize=(9, 5))

# Calculate the correlation between latitude, longitude, and ratings
correlation_matrix = df[['Latitude', 'Longitude', 'Aggregate rating']].corr

# Create a heatmap to visualize the correlation
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm', fmt=".2f")

# Set Title
plt.title("Correlation Between Restaurant's location and Rating")

# Display Chart
plt.show()
```



In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]: