Zomato Dataset Exploratory Data Analysis

In [1]:
```
#Importing The necessary Python Libraray
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
```

In [2]:
```
df=pd.read_csv("C:/Users/HP/Downloads/zomato.csv",encoding="latin-1")
```

In [3]:
```
#To show the top 5 records
df.head()
```

Out[3]:

	Restaurant ID	Restaurant Name	Country Code	City	Address	Locality	Locality Verbose	Longitude	Latitud
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	14.5654
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	14.5537
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	14.5814
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	14.5853
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	14.5844

5 rows × 21 columns

In [4]:
```
#to see the shape (number of rows and number of columns
df.shape
```

Out[4]:
```
(9551, 21)
```

In [5]:
```
#To see all the columns of the data set
df.columns
```

Out[5]:
```
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')
```

In [6]:
```
#To know the information of the data set
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
```

Df.describe command will provide the statistical summary of all the columns of integer data types as staitistics measurements can not be calculated for the text data types.

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

Out[7]:

	Restaurant ID	Country Code	Longitude	Latitude	Average Cost for two	Price range	Aggregate rating	
count	9.551000e+03	9551.000000	9551.000000	9551.000000	9551.000000	9551.000000	9551.000000	955
mean	9.051128e+06	18.365616	64.126574	25.854381	1199.210763	1.804837	2.666370	15
std	8.791521e+06	56.750546	41.467058	11.007935	16121.183073	0.905609	1.516378	43
min	5.300000e+01	1.000000	-157.948486	-41.330428	0.000000	1.000000	0.000000	
25%	3.019625e+05	1.000000	77.081343	28.478713	250.000000	1.000000	2.500000	
50%	6.004089e+06	1.000000	77.191964	28.570469	400.000000	2.000000	3.200000	3
75%	1.835229e+07	1.000000	77.282006	28.642758	700.000000	2.000000	3.700000	13
max	1.850065e+07	216.000000	174.832089	55.976980	800000.000000	4.000000	4.900000	1093

EDA steps

1) Finding out the Missing values 2) Exploring the Numerical Variables 3) Exploring the Categorical Variables 4) Finding the relationship Between the features

In [8]:
```
#To find out the missing values over here
df.isnull().sum()
```

Out[8]:
```
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 [9]:
```
#Another way of finding out the columns having null values or missing values
[features for features in df.columns if df[features].isnull().sum() >0]
```

Out[9]:
```
['Cuisines']
```

In [10]:
```
#Importing another file
df_country = pd.read_excel("C:/Users/HP/Downloads/Country-Code.xlsx")
df_country.head()
```

Out[10]:

	Country Code	Country
0	1	India
1	14	Australia
2	30	Brazil
3	37	Canada
4	94	Indonesia

In [11]:
```
df_country.shape
```

Out[11]:
```
(15, 2)
```

In [12]:
```
a=df.columns
b=df_country.columns
print(a)
print(b)
```

```
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(['Country Code', 'Country'], dtype='object')
```

In [13]:
```
#We have country code column common in the both data set so we can merge both data se
final_df = pd.merge(df,df_country,on='Country Code',how='left')
```

In [14]:
```
final_df.head(5)
```

Out[14]:

	Restaurant ID	Restaurant Name	Country Code	City	Address	Locality	Locality Verbose	Longitude	Latitud
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	14.5654
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	14.5537
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	14.5814
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	14.5853
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	14.5844

5 rows × 22 columns

In [15]:
```
#another way of finding data types
final_df.dtypes
```

Out[15]:
```
Restaurant ID           int64
Restaurant Name        object
Country Code            int64
City                   object
Address                object
Locality               object
Locality Verbose       object
Longitude             float64
Latitude              float64
Cuisines               object
Average Cost for two    int64
Currency               object
Has Table booking      object
Has Online delivery    object
Is delivering now      object
Switch to order menu   object
Price range             int64
Aggregate rating      float64
Rating color           object
Rating text            object
Votes                   int64
```

```
Country                 object
dtype: object
```

In [16]:
```
final_df.columns
```

Out[16]:
```
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', 'Country'],
      dtype='object')
```

In [17]:
```
final_df.Country.value_counts()
```

Out[17]:
```
India             8652
United States      434
United Kingdom      80
Brazil              60
UAE                 60
South Africa        60
New Zealand         40
Turkey              34
Australia           24
Phillipines         22
Indonesia           21
Singapore           20
Qatar               20
Sri Lanka           20
Canada               4
Name: Country, dtype: int64
```

In [18]:
```
country_names=final_df.Country.value_counts().index
country_names
```

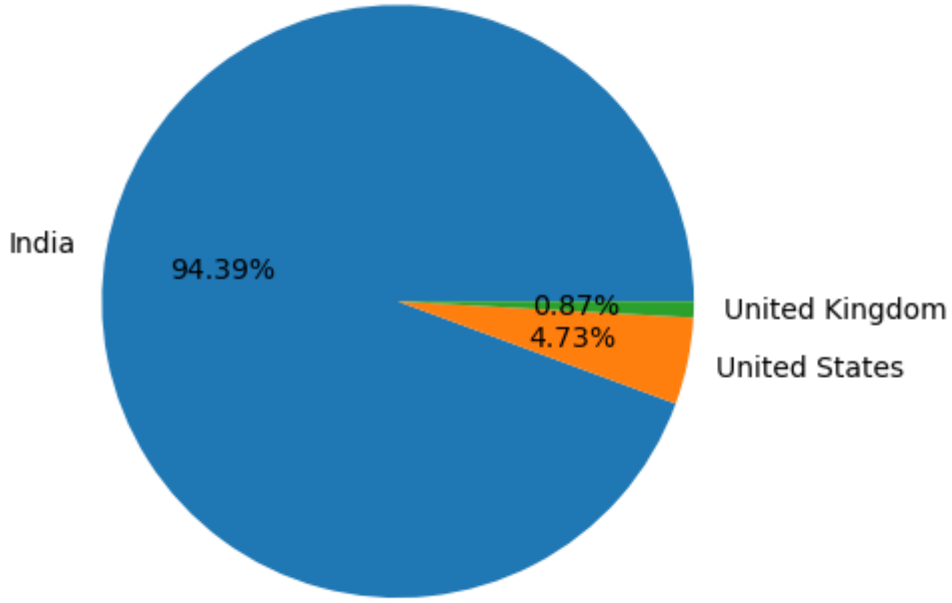
Out[18]:
```
Index(['India', 'United States', 'United Kingdom', 'Brazil', 'UAE',
       'South Africa', 'New Zealand', 'Turkey', 'Australia', 'Phillipines',
       'Indonesia', 'Singapore', 'Qatar', 'Sri Lanka', 'Canada'],
      dtype='object')
```

In [19]:
```
country_val=final_df.Country.value_counts().values
print(country_val)
```

```
[8652  434   80   60   60   60   40   34   24   22   21   20   20   20
    4]
```

In [20]:
```
#Pie charts - Top 3 countries using Zomato based on the transactions
plt.pie(country_val[:3],labels=country_names[:3],autopct="%1.2f%%")
```

Out[20]:
```
([<matplotlib.patches.Wedge at 0x1e0157a6d30>,
  <matplotlib.patches.Wedge at 0x1e0157bc4c0>,
  <matplotlib.patches.Wedge at 0x1e0157bcbe0>],
 [Text(-1.0829742700952103, 0.19278674827836725, 'India'),
  Text(1.077281715838356, -0.22240527134123297, 'United States'),
  Text(1.0995865153823035, -0.03015783794312073, 'United Kingdom')],
 [Text(-0.590713238233751, 0.10515640815183668, '94.39%'),
  Text(0.5876082086391032, -0.12131196618612707, '4.73%'),
  Text(0.5997744629358018, -0.01644972978715676, '0.87%')])
```



Maximum Transactions are from India (94.39%) and then from Unites states and then United Kingdom

In [36]:
```
final_df.columns
```

Out[36]:
```
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', 'Country'],
      dtype='object')
```

In [37]:
```
final_df.groupby(['Aggregate rating', 'Rating color', 'Rating text']).size().reset_inc
```

Out[37]:

	Aggregate rating	Rating color	Rating text	0
0	0.0	White	Not rated	2148
1	1.8	Red	Poor	1
2	1.9	Red	Poor	2
3	2.0	Red	Poor	7
4	2.1	Red	Poor	15
5	2.2	Red	Poor	27
6	2.3	Red	Poor	47
7	2.4	Red	Poor	87
8	2.5	Orange	Average	110
9	2.6	Orange	Average	191
10	2.7	Orange	Average	250
11	2.8	Orange	Average	315
12	2.9	Orange	Average	381
13	3.0	Orange	Average	468
14	3.1	Orange	Average	519
15	3.2	Orange	Average	522
16	3.3	Orange	Average	483

17	3.4	Orange	Average	498
18	3.5	Yellow	Good	480
19	3.6	Yellow	Good	458
20	3.7	Yellow	Good	427
21	3.8	Yellow	Good	400
22	3.9	Yellow	Good	335
23	4.0	Green	Very Good	266
24	4.1	Green	Very Good	274
25	4.2	Green	Very Good	221
26	4.3	Green	Very Good	174
27	4.4	Green	Very Good	144
28	4.5	Dark Green	Excellent	95
29	4.6	Dark Green	Excellent	78
30	4.7	Dark Green	Excellent	42
31	4.8	Dark Green	Excellent	25
32	4.9	Dark Green	Excellent	61

In [23]:
```
ratings=final_df.groupby(['Aggregate rating', 'Rating color', 'Rating text']).size().
```

In [38]:
```
print(ratings)
```

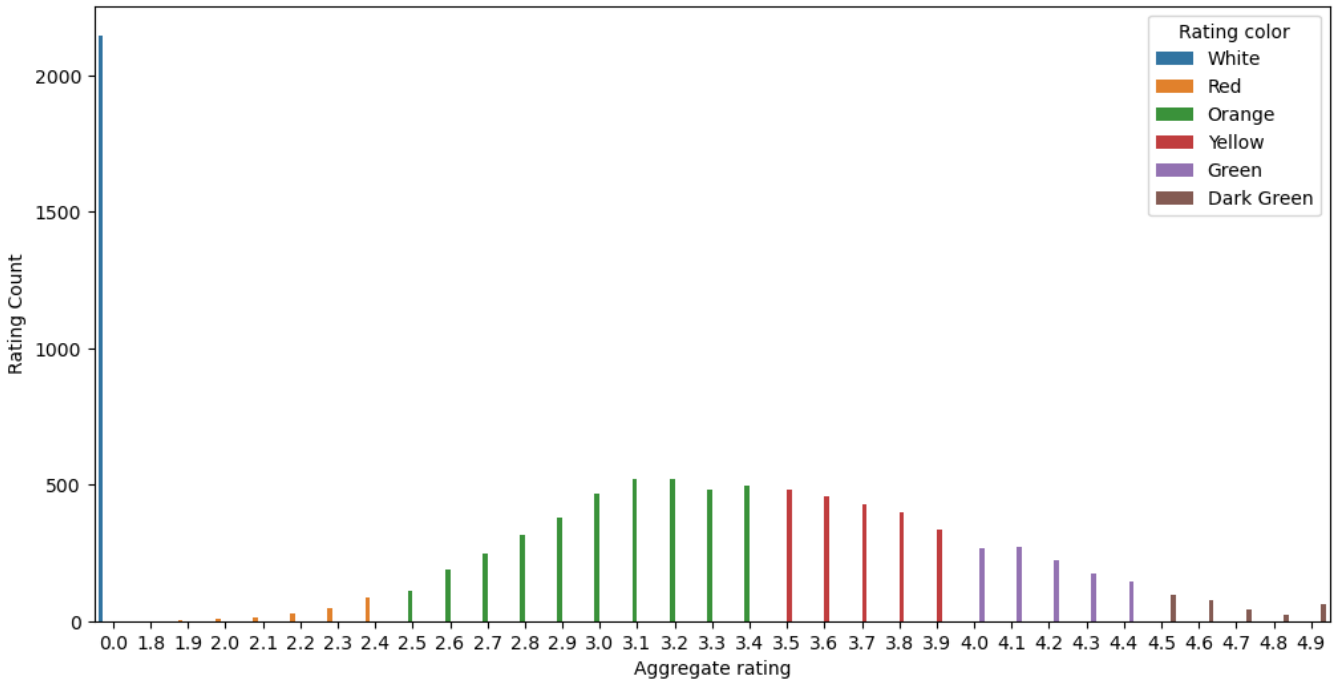
```
    Aggregate rating Rating color Rating text  Rating Count
0                0.0        White   Not rated          2148
1                1.8          Red        Poor             1
2                1.9          Red        Poor             2
3                2.0          Red        Poor             7
4                2.1          Red        Poor            15
5                2.2          Red        Poor            27
6                2.3          Red        Poor            47
7                2.4          Red        Poor            87
8                2.5       Orange     Average           110
9                2.6       Orange     Average           191
10               2.7       Orange     Average           250
11               2.8       Orange     Average           315
12               2.9       Orange     Average           381
13               3.0       Orange     Average           468
14               3.1       Orange     Average           519
15               3.2       Orange     Average           522
16               3.3       Orange     Average           483
17               3.4       Orange     Average           498
18               3.5       Yellow        Good           480
19               3.6       Yellow        Good           458
20               3.7       Yellow        Good           427
21               3.8       Yellow        Good           400
22               3.9       Yellow        Good           335
23               4.0        Green   Very Good           266
24               4.1        Green   Very Good           274
25               4.2        Green   Very Good           221
26               4.3        Green   Very Good           174
27               4.4        Green   Very Good           144
28               4.5   Dark Green   Excellent            95
29               4.6   Dark Green   Excellent            78
30               4.7   Dark Green   Excellent            42
31               4.8   Dark Green   Excellent            25
32               4.9   Dark Green   Excellent            61
```

Observations

when rating is between 4.5 to 4.9 it is excellent when rating is between 4.0 to 4.4 it is very good when rating is between 3.5 to 3.9 it is Good when rating is between 2.5 to 3.4 it is average when rating is between 1.8 to 2.4 it is Poor

In [39]:
```
import matplotlib
matplotlib.rcParams['figure.figsize'] = (12,6)
sns.barplot(x='Aggregate rating',y='Rating Count',hue='Rating color',data=ratings)
```

Out[39]:
```
<AxesSubplot:xlabel='Aggregate rating', ylabel='Rating Count'>
```

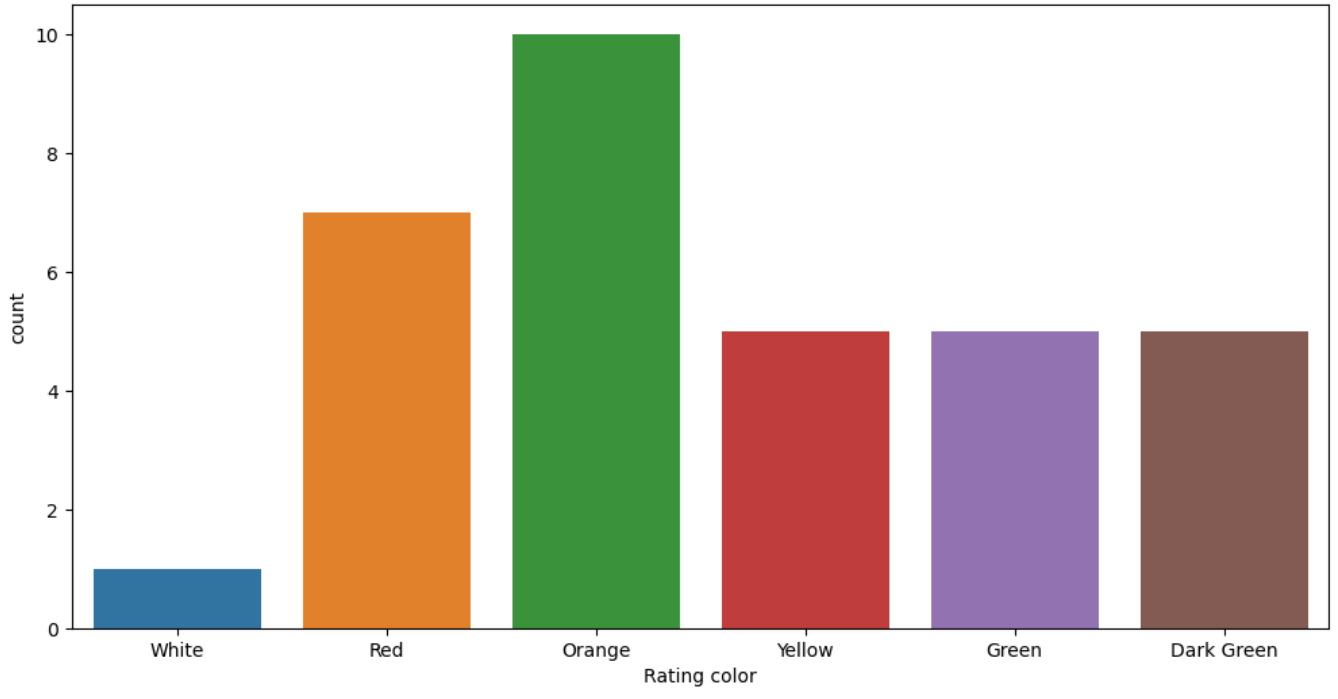


observations

1)Not rated Count is very High 2)Maximum Number of ratings between 2.5 to 3.9

In [40]:
```
## Count Plot
sns.countplot(x='Rating color',data=ratings)
```

Out[40]:
```
<AxesSubplot:xlabel='Rating color', ylabel='count'>
```



In [41]:
```
# Countries name that has given zero ratings
final_df[final_df['Rating color']=='White'].groupby('Country').size().reset_index()
```

Out[41]:

	Country	0
0	Brazil	5
1	India	2139
2	United Kingdom	1
3	United States	3

Observation

max number of zero ratings are from Indian Customers

In [42]:
```
#  Which Currency are used by which country
final_df[['Country','Currency']].groupby(['Country','Currency']).size().reset_index()
```

Out[42]:

	Country	Currency	0
0	Australia	Dollar($)	24
1	Brazil	Brazilian Real(R$)	60
2	Canada	Dollar($)	4
3	India	Indian Rupees(Rs.)	8652
4	Indonesia	Indonesian Rupiah(IDR)	21
5	New Zealand	NewZealand($)	40
6	Phillipines	Botswana Pula(P)	22
7	Qatar	Qatari Rial(QR)	20
8	Singapore	Dollar($)	20
9	South Africa	Rand(R)	60
10	Sri Lanka	Sri Lankan Rupee(LKR)	20
11	Turkey	Turkish Lira(TL)	34
12	UAE	Emirati Diram(AED)	60
13	United Kingdom	Pounds(□£)	80
14	United States	Dollar($)	434

In [43]:
```
#Which Countries has Online delievery options
final_df[final_df['Has Online delivery']=="Yes"].Country.value_counts()
```

Out[43]:
```
India    2423
UAE        28
Name: Country, dtype: int64
```

Observations

Online delievery are available in India and UAE

In [44]:
```
final_df.columns
```

Out[44]:
```
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', 'Country'],
      dtype='object')
```

In [45]:
```
final_df.columns
```

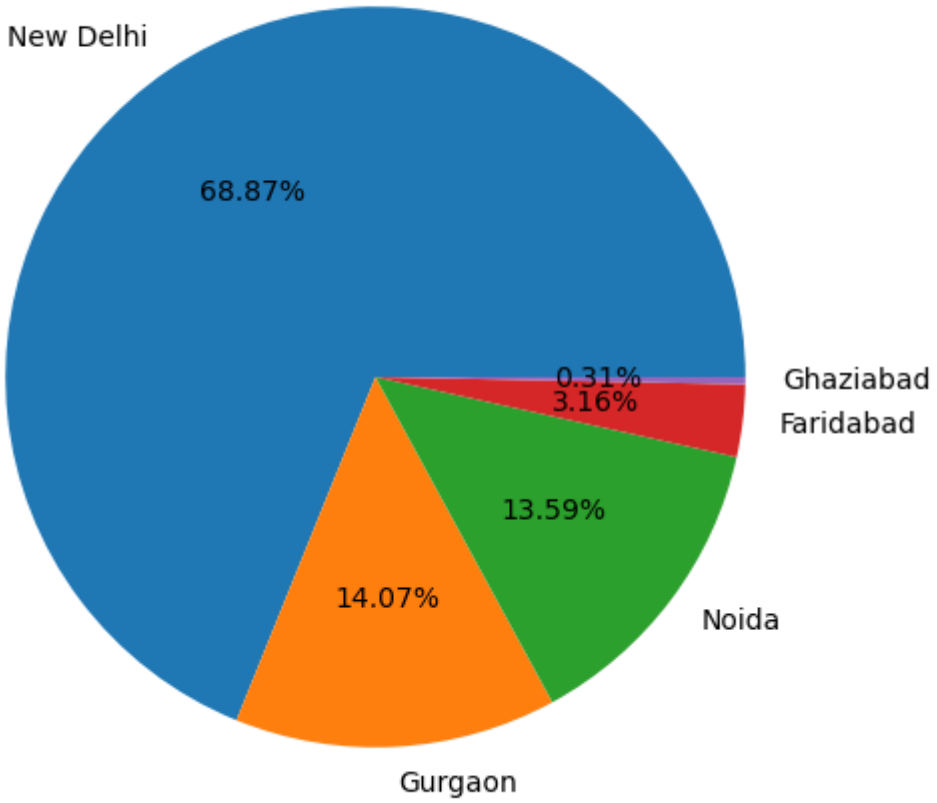
Out[45]:
```
Index(['Restaurant ID', 'Restaurant Name', 'Country Code', 'City', 'Address',
       'Locality', 'Locality Verbose', 'Longitude', 'Latitude', 'Cuisines',
       'Average Cost for two', 'Currency', 'Has Table booking',
       'Has Online delivery', 'Is delivering now', 'Switch to order menu',
       'Price range', 'Aggregate rating', 'Rating color', 'Rating text',
       'Votes', 'Country'],
      dtype='object')
```

In [32]:
```
final_df.City.value_counts().index
```

Out[32]:
```
Index(['New Delhi', 'Gurgaon', 'Noida', 'Faridabad', 'Ghaziabad',
       'Bhubaneshwar', 'Amritsar', 'Ahmedabad', 'Lucknow', 'Guwahati',
       ...
       'Ojo Caliente', 'Montville', 'Monroe', 'Miller', 'Middleton Beach',
       'Panchkula', 'Mc Millan', 'Mayfield', 'Macedon', 'Vineland Station'],
      dtype='object', length=141)
```

In [33]:
```
# Top 5 citites doing highest transactions from India
city_values = final_df.City.value_counts().values
city_labels = final_df.City.value_counts().index
plt.pie(city_values[:5],labels=city_labels[:5],autopct='%1.2f%%')
```

Out[33]:
```
([<matplotlib.patches.Wedge at 0x1e0178fceb0>,
  <matplotlib.patches.Wedge at 0x1e0178f5e50>,
  <matplotlib.patches.Wedge at 0x1e017908c70>,
  <matplotlib.patches.Wedge at 0x1e017914400>,
  <matplotlib.patches.Wedge at 0x1e017914b20>],
 [Text(-0.6145352824185932, 0.9123301960708633, 'New Delhi'),
  Text(0.0623675251198054, -1.0982305276263407, 'Gurgaon'),
  Text(0.8789045225625368, -0.6614581167535246, 'Noida'),
  Text(1.0922218418223437, -0.13058119407559224, 'Faridabad'),
  Text(1.099946280005612, -0.010871113182029924, 'Ghaziabad')],
 [Text(-0.3352010631374145, 0.497634652402289, '68.87%'),
  Text(0.0340186500653484, -0.5990348332507311, '14.07%'),
  Text(0.47940246685229276, -0.36079533641101336, '13.59%'),
  Text(0.5957573682667329, -0.07122610585941394, '3.16%'),
  Text(0.5999706981848791, -0.005929698099289049, '0.31%')])
```



"""""#Obsercations

Top 5 Cities from India Doing Highest Number of the Transactions

1)New Delhi 2)Gurgaon 3)Noida 4)Faridabad 5)Gaziabad

In [ ]:
```
# Find the Top 10 Cuisines
final_df
```

In [ ]:
```
final_df[['Cuisines']].groupby(['Cuisines']).value_counts().reset_index()
```

In [ ]:
```
# Afgani cuisines is the most sold cuisine
```