EDA ON ZOMATO DATASET

IMPORTING LIBRARIES

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

IMPORTING DATASET

In [10]:
```
df=pd.read_csv("E:\\DATA SCIENCE\\Project\\python\\zomato\\zomato.csv",encoding='latin-
```

READING THE 1ST 10 ROWS

In [11]:
```
df.head()
```

Out[11]:

	Restaurant ID	Restaurant Name	Country Code	City	Address	Locality	Locality Verbose	Longitude	
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	1
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	1
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	1

	Restaurant ID	Restaurant Name	Country Code	City	Address	Locality	Locality Verbose	Longitude	
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	1
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	1

5 rows × 21 columns

UNDERSTANDING THE DATASET

In [12]:
```
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
```

GETTING THE ROWS & COLUMN

In [14]:
```
df.shape
```

Out[14]:
```
(9551, 21)
```

GETTING NUMERICAL VALUES AND VARIABLES

In [17]:
```
df.describe()
```

Out[17]:

	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
mean	9.051128e+06	18.365616	64.126574	25.854381	1199.210763	1.804837	2.666370
std	8.791521e+06	56.750546	41.467058	11.007935	16121.183073	0.905609	1.516378
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
75%	1.835229e+07	1.000000	77.282006	28.642758	700.000000	2.000000	3.700000
max	1.850065e+07	216.000000	174.832089	55.976980	800000.000000	4.000000	4.900000

GETTING THE SUM OF NULL / MISSING VALUES

In [18]:
```
df.isnull().sum()
```

Out[18]:
```
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
```

SO THERE ARE 9 MISSING VALUES

FINDING OUT THE VALUE WHERE NULL VALUES > 0 BY USING LIST COMPREHENSIONS

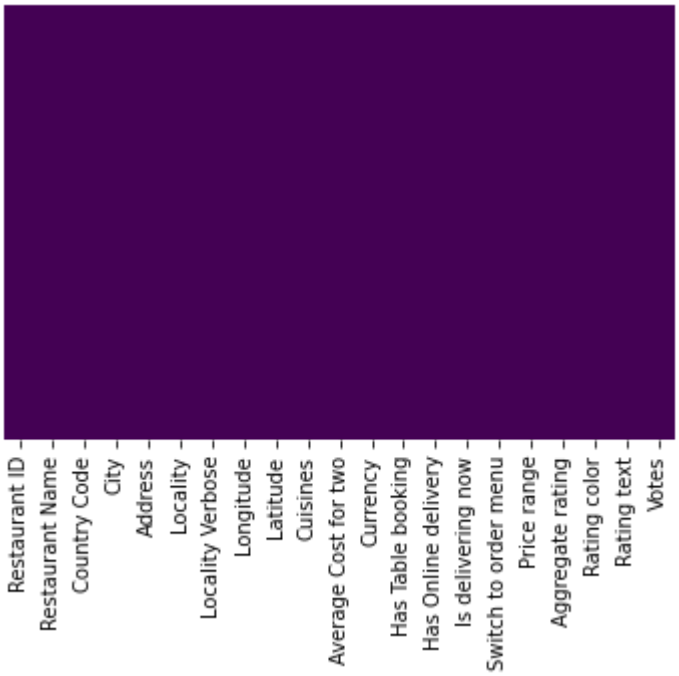
In [22]:
```
[i for i in df.columns if df[i].isnull().sum()>0]
```

Out[22]:
```
['Cuisines']
```

WE FOUND 'CUISINES' COLUMN HAVING NULL VALUE

In [28]:
```
sns.heatmap(df.isnull(),yticklabels=False,cbar=False,cmap='viridis')
```

Out[28]:
```
<AxesSubplot:>
```



WE DO NOT HAVE MANY NULL VALUES. SO, WE ARE NOT ABLE TO SEE ANY.

IMPORTING ANOTHER DATASET i.e. COUNTRY CODE

In [30]:
```
df_Country=pd.read_excel('E:\\DATA SCIENCE\\Project\\python\\zomato\\Country-Code.xlsx'
```

GETTING THE TOP ROWS

In [31]:
```
df_Country.head()
```

Out[31]:

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

COMBINING THE TWO DATAFRAME & DISPLAYING TOP 5 ROWS

In [33]:
```
df_final=pd.merge(df,df_Country,on='Country Code',how='left')
df_final.head(5)
#on: The common table
#how: type of join
```

Out[33]:

	Restaurant ID	Restaurant Name	Country Code	City	Address	Locality	Locality Verbose	Longitude	
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	1
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	1
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	1

	Restaurant ID	Restaurant Name	Country Code	City	Address	Locality	Locality Verbose	Longitude	
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	1
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	1

5 rows × 22 columns

To check Datatypes

In [35]:
```
df_final.dtypes
```

Out[35]:
```
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
```

HOWMANY DIFFERENT COUNTRIES ARE THERE AND DISPLAY THEIR RESPECTIVE

RECORDS

In [56]:
```
df_final.Country.value_counts()
```

Out[56]:
```
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
```

DISPLAYING ONLY THE COUNTRY NAME

In [68]:
```
country_names=df_final.Country.value_counts().index
```

VALUES OF EACH COUNTRY

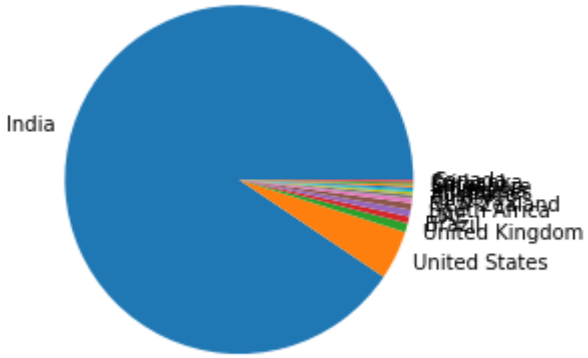
In [72]:
```
country_value=df_final.Country.value_counts().values
```

WHICH COUNTRY PROVIDING MAXIMUM ORDERS [PIE CHART]

In [73]:
```
plt.pie(country_value,labels=country_names)
```

Out[73]:
```
([<matplotlib.patches.Wedge at 0x17454135dc0>,
  <matplotlib.patches.Wedge at 0x174541356d0>,
  <matplotlib.patches.Wedge at 0x174541b1c70>,
  <matplotlib.patches.Wedge at 0x174541b1fd0>,
  <matplotlib.patches.Wedge at 0x174541c7550>,
  <matplotlib.patches.Wedge at 0x174541c7a30>,
  <matplotlib.patches.Wedge at 0x174541c7f10>,
  <matplotlib.patches.Wedge at 0x174541d4430>,
  <matplotlib.patches.Wedge at 0x174541d4910>,
  <matplotlib.patches.Wedge at 0x174541d4df0>,
  <matplotlib.patches.Wedge at 0x1745419c910>,
  <matplotlib.patches.Wedge at 0x174541e07c0>,
  <matplotlib.patches.Wedge at 0x174541e0ca0>,
  <matplotlib.patches.Wedge at 0x174541f01c0>,
  <matplotlib.patches.Wedge at 0x174541f06a0>],
```

```
[Text(-1.052256163793291, 0.3205572737577906, 'India'),
 Text(0.9911329812843455, -0.477132490415823, 'United States'),
 Text(1.0572858296119743, -0.3035567072257165, 'United Kingdom'),
 Text(1.070138816916019, -0.2545641619112621, 'Brazil'),
 Text(1.0793506814479759, -0.21213699926648824, 'UAE'),
 Text(1.086881147244973, -0.16937937230799818, 'South Africa'),
 Text(1.0918635911832035, -0.1335436192729486, 'New Zealand'),
 Text(1.0947903814016446, -0.10692998078388304, 'Turkey'),
 Text(1.096631023945382, -0.08602556201794338, 'Australia'),
 Text(1.0978070729776455, -0.06942355882735218, 'Phillipines'),
 Text(1.0986791544015209, -0.05388984768543213, 'Indonesia'),
 Text(1.0993059848742366, -0.039068550263413035, 'Singapore'),
 Text(1.0997248508282123, -0.02460187941736628, 'Qatar'),
 Text(1.0999533462179636, -0.010130949802716446, 'Sri Lanka'),
 Text(1.0999990477553414, -0.0014473898376707638, 'Canada')])
```

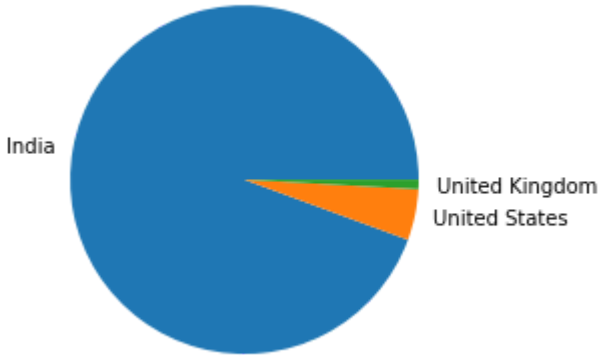


In [ ]:
```
#INDIA IS GIVING MAXIMUM ORDERS
```

TOP3 COUNTRIES PROVIDING MAXIMUM ORDERS

In [74]:
```
plt.pie(country_value[:3],labels=country_names[:3])
```

Out[74]:
```
([<matplotlib.patches.Wedge at 0x17454239400>,
  <matplotlib.patches.Wedge at 0x17454239940>,
  <matplotlib.patches.Wedge at 0x17454239e20>],
 [Text(-1.0829742700952103, 0.19278674827836725, 'India'),
  Text(1.077281715838356, -0.22240527134123297, 'United States'),
  Text(1.0995865153823035, -0.03015783794312073, 'United Kingdom')])
```

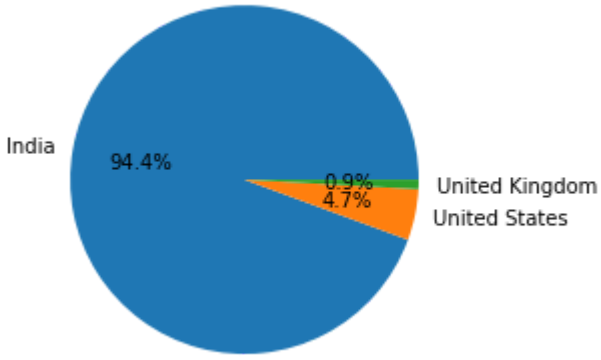


In [ ]:
```
#TOP3 : INDIA > USA > UK
```

PERCENTAGE OF ORDER THAT EACH COUNTRY CONTRIBUTES

In [82]:
```
plt.pie(country_value[:3],labels=country_names[:3],autopct='%1.1f%%')
```

Out[82]:
```
([<matplotlib.patches.Wedge at 0x174515aba90>,
  <matplotlib.patches.Wedge at 0x174515ba250>,
  <matplotlib.patches.Wedge at 0x174515ba970>],
 [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, '94.4%'),
  Text(0.5876082086391032, -0.12131196618612707, '4.7%'),
  Text(0.5997744629358018, -0.01644972978715676, '0.9%')])
```



In [ ]:
```
#Observation:
#Zomato's Maximum Transaction Records are from India,
#After India USA then UK comes.
```

reading the columns that are available

In [83]:
```
df_final.columns
```

Out[83]:
```
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')
```

Let us know the Rating, color and respective categories.

In [85]:
```
df_final.groupby(['Aggregate rating', 'Rating color','Rating text']).size()
```

Out[85]:

Aggregate rating	Rating color	Rating text	
0.0	White	Not rated	2148
1.8	Red	Poor	1
1.9	Red	Poor	2
2.0	Red	Poor	7
2.1	Red	Poor	15
2.2	Red	Poor	27
2.3	Red	Poor	47
2.4	Red	Poor	87
2.5	Orange	Average	110
2.6	Orange	Average	191
2.7	Orange	Average	250
2.8	Orange	Average	315
2.9	Orange	Average	381
3.0	Orange	Average	468
3.1	Orange	Average	519
3.2	Orange	Average	522
3.3	Orange	Average	483
3.4	Orange	Average	498
3.5	Yellow	Good	480
3.6	Yellow	Good	458
3.7	Yellow	Good	427
3.8	Yellow	Good	400
3.9	Yellow	Good	335
4.0	Green	Very Good	266
4.1	Green	Very Good	274
4.2	Green	Very Good	221
4.3	Green	Very Good	174
4.4	Green	Very Good	144
4.5	Dark Green	Excellent	95
4.6	Dark Green	Excellent	78
4.7	Dark Green	Excellent	42
4.8	Dark Green	Excellent	25
4.9	Dark Green	Excellent	61

dtype: int64

In [88]:
```
df_final.groupby(['Aggregate rating', 'Rating color','Rating text']).size().reset_index
```

Out[88]:

	Aggregate rating	Rating color	Rating text	0

	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

NOW LET'S REPLACE THIS 0 INDEX WITH 'RATING COUNT'

In [92]:
```
ratings=df_final.groupby(['Aggregate rating', 'Rating color','Rating text']).size().res
ratings.head(10)
```

Out[92]:

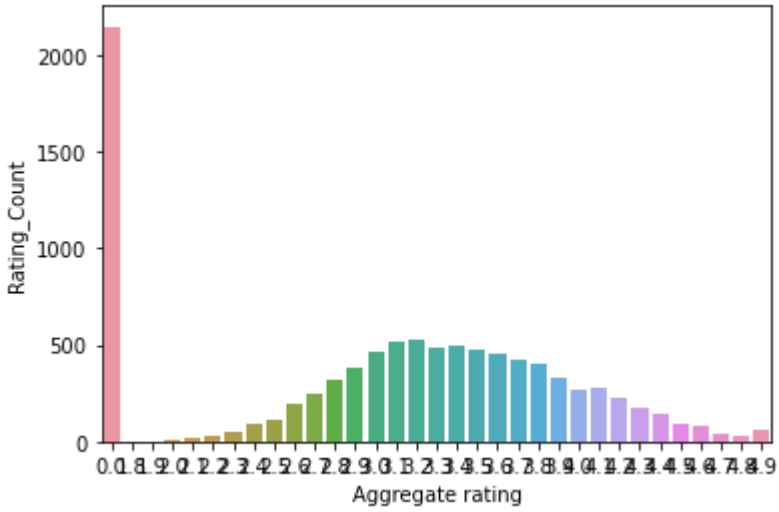
	Aggregate rating	Rating color	Rating text	Rating_Count
0	0.0	White	Not rated	2148
1	1.8	Red	Poor	1
2	1.9	Red	Poor	2
3	2.0	Red	Poor	7
4	2.1	Red	Poor	15
5	2.2	Red	Poor	27
6	2.3	Red	Poor	47
7	2.4	Red	Poor	87
8	2.5	Orange	Average	110
9	2.6	Orange	Average	191

OBSERVATION

1. RATING BETWEEN 4.5 TO 4.9 => EXCELLENT
2. RATING BETWEEN 4.0 TO 3.4 => VERY GOOD
3. RATING BETWEEN 3.5 TO 3.9 => GOOD
4. RATING BETWEEN 3.0 TO 3.4 => AVERAGE
5. RATING BETWEEN 2.5 TO 2.9 => AVERAGE
6. RATING BETWEEN 2.0 TO 2.4 => POOR
7. 2148 CUSTOMER HAS NOT GIVEN ANY RATING.

In [94]:
```
sns.barplot(x='Aggregate rating',y='Rating_Count',data=ratings)
```

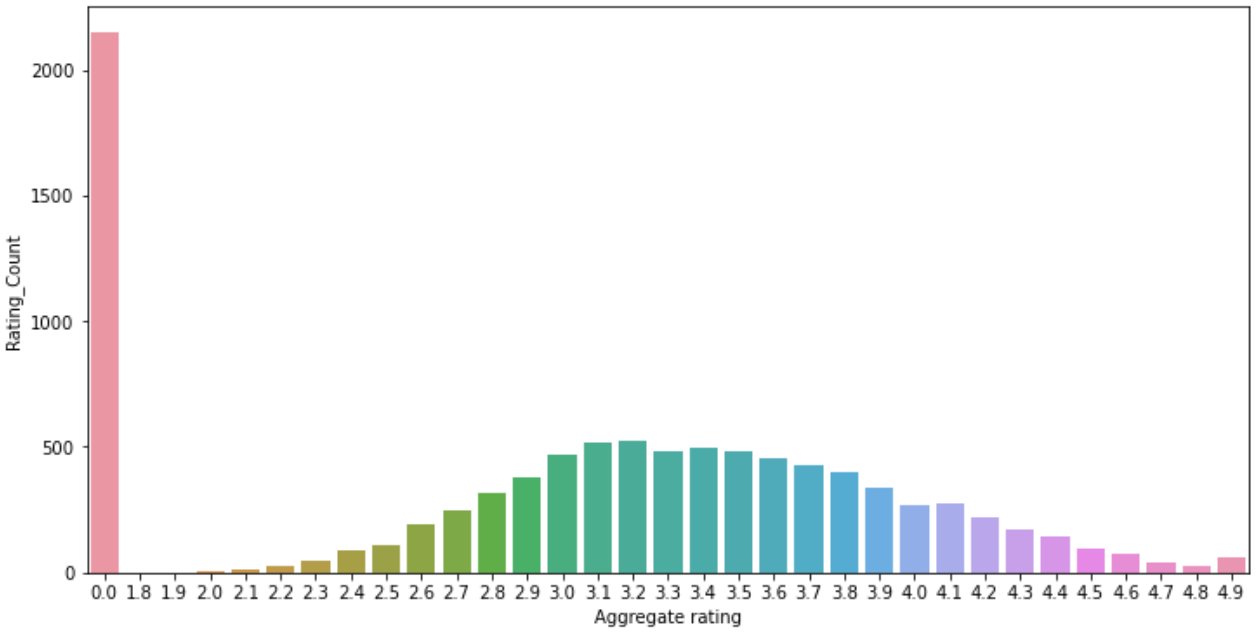
Out[94]:
```
<AxesSubplot:xlabel='Aggregate rating', ylabel='Rating_Count'>
```



Let's make this dig Bigger.

In [96]:
```
plt.rcParams['figure.figsize']=(12,6)
#rcparams:used to change any parameters if we want.
sns.barplot(x='Aggregate rating',y='Rating_Count',data=ratings)
```

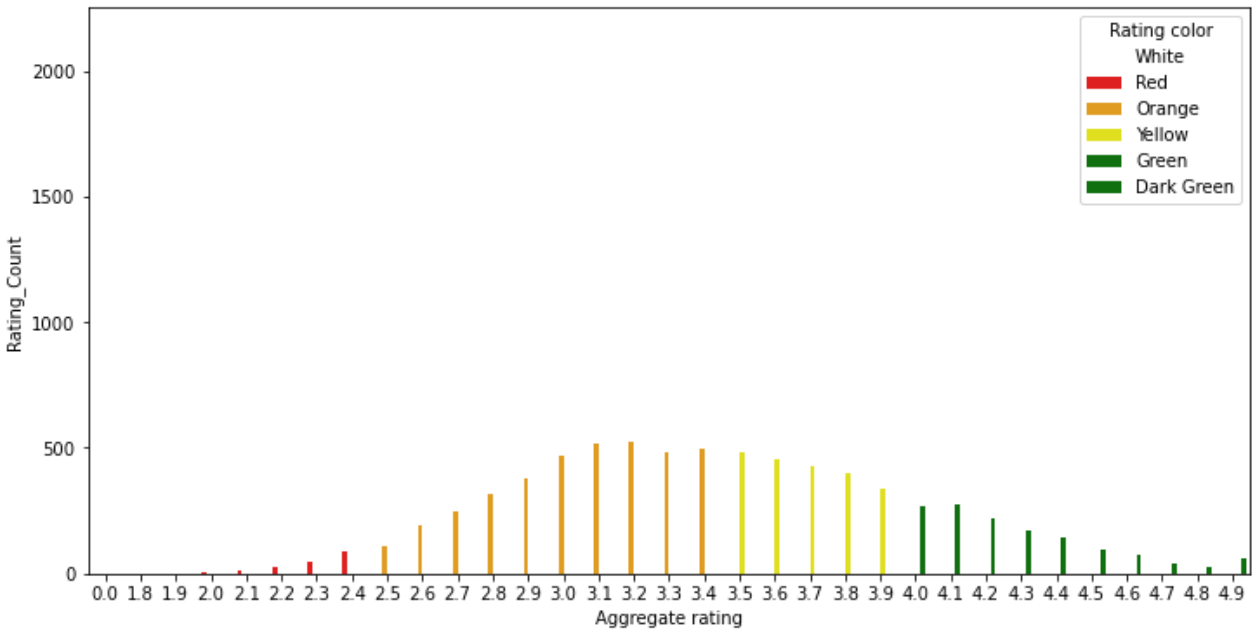
Out[96]:
```
<AxesSubplot:xlabel='Aggregate rating', ylabel='Rating_Count'>
```



Let's assign the color to tha ratings as per our dataset.

In [102…
```
plt.rcParams['figure.figsize']=(12,6)
#rcparams:used to change any parameters if we want.
sns.barplot(x='Aggregate rating',y='Rating_Count',hue='Rating color',data=ratings,palet
#hue : to get the label of colors.
```

Out[102… <AxesSubplot:xlabel='Aggregate rating', ylabel='Rating_Count'>

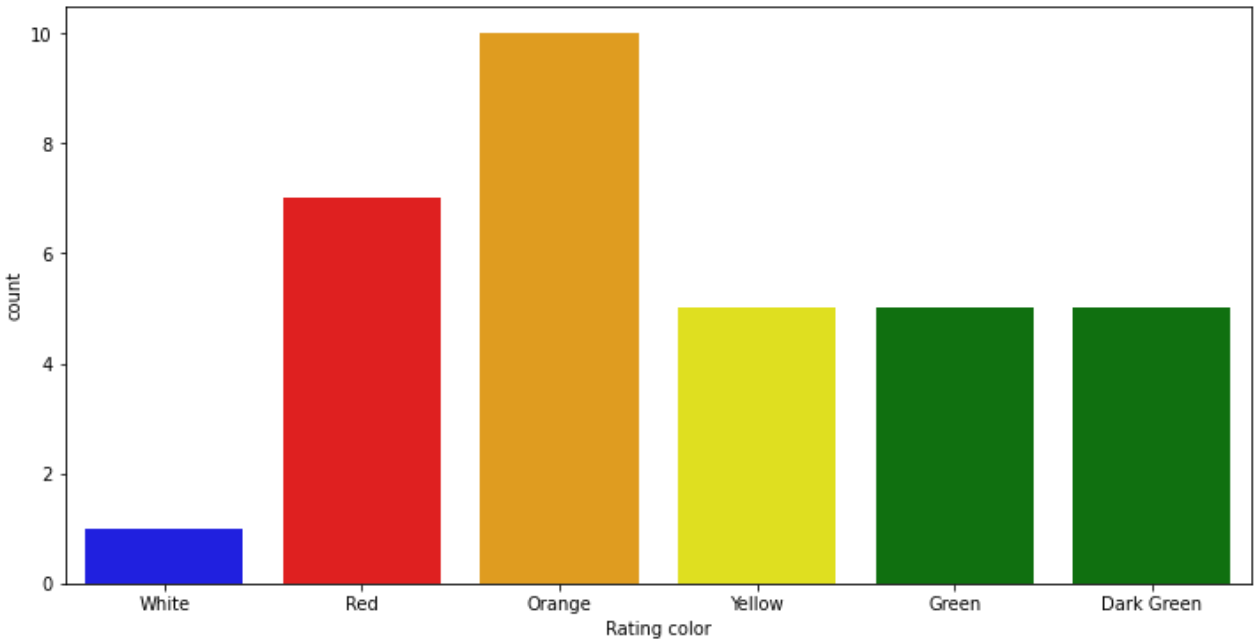


OBSERVATION:

1. Count of 'NOT RATED' is very high.
2. Maximum number of rating is between 3.0 to 3.4

In [105…
```
#count plot
sns.countplot(x='Rating color',data=ratings, palette=['blue','red','orange','yellow','g
```

Out[105… <AxesSubplot:xlabel='Rating color', ylabel='count'>



In [108…
```
ratings.head(10)
```

Out[108…

	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

FIND THE COUNTRIES THAT HAS GIVEN 0 RATINGS

In [117…

```
df_final.columns
```

Out[117…

```
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 [128…

```
df_final.columns
```

Out[128…

```
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 [130…

```
df_final.groupby(['Aggregate rating','Country']).size().reset_index().head(5)
```

Out[130…

	Aggregate rating	Country	0
0	0.0	Brazil	5
1	0.0	India	2139
2	0.0	United Kingdom	1
3	0.0	United States	3

	Aggregate rating	Country	0
4	1.8	India	1

OBSERVATION: INDIAN CUSTOMERS HAS GIVEN ZERO RATING.

WHICH CURRENCY IS USED BY WHICH COUNTRY

In [137…
```
df_final[['Country','Currency']].groupby(['Country','Currency']).size().reset_index()
```

Out[137…

	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

which country do have online delivery

In [138…
```
df_final[df_final['Has Online delivery']=='Yes'].Country.value_counts()
```

Out[138…
```
India    2423
UAE        28
Name: Country, dtype: int64
```

OBSERVATION:

1. Online delivery is available in UAE& INDIA.

CREATING PIE CHART FOR TOP 5 CITIES DISTRIBUTION

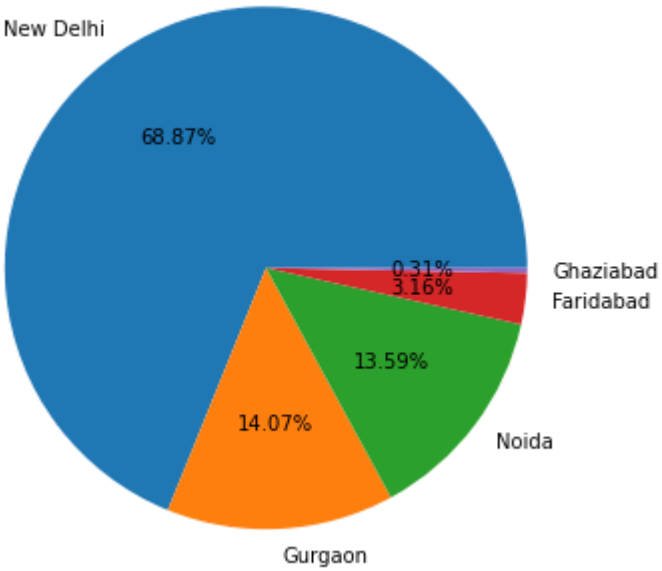
In [139…
```
df_final.City.value_counts().index
```

Out[139…
```
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 [147…
```
city_values=df_final.City.value_counts().values
city_labels=df_final.City.value_counts().index
```

In [150…
```
plt.pie(city_values[:5],labels=city_labels[:5],autopct='%1.2f%%')
```

Out[150…
```
([<matplotlib.patches.Wedge at 0x174552bb460>,
  <matplotlib.patches.Wedge at 0x174552bbbe0>,
  <matplotlib.patches.Wedge at 0x174553e9340>,
  <matplotlib.patches.Wedge at 0x174553e9a60>,
  <matplotlib.patches.Wedge at 0x174553f71c0>],
 [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%')])
```



TOP10 CUISINES

In [157…
```
df_final.head(2)
```

Out[157…

	Restaurant ID	Restaurant Name	Country Code	City	Address	Locality	Locality Verbose	Longitude	Latitude	Cui
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	Fr Japa Des
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	Japa

2 rows × 22 columns

In [194…
```
df_final.groupby(['Cuisines']).size().reset_index().head(10)
```

Out[194…

	Cuisines	0
0	Afghani	4
1	Afghani, Mughlai, Chinese	1
2	Afghani, North Indian	1
3	Afghani, North Indian, Pakistani, Arabian	1
4	African	1
5	African, Portuguese	1
6	American	31
7	American, Asian, Burger	1
8	American, Asian, European, Seafood	1
9	American, Asian, Italian, Seafood	1

In [203…
```
df_TOP10=df_final.groupby(['Cuisines']).size().reset_index().rename(columns={0:'TOP10'}
df_TOP10.head(10)
```

Out[203…

	Cuisines	TOP10

	Cuisines	TOP10
0	Afghani	4
1	Afghani, Mughlai, Chinese	1
2	Afghani, North Indian	1
3	Afghani, North Indian, Pakistani, Arabian	1
4	African	1
5	African, Portuguese	1
6	American	31
7	American, Asian, Burger	1
8	American, Asian, European, Seafood	1
9	American, Asian, Italian, Seafood	1

In [208…

```
df_TOP10.sort_values(by=['TOP10'],ascending=False).head(10)
```

Out[208…

	Cuisines	TOP10
1306	North Indian	936
1329	North Indian, Chinese	511
497	Chinese	354
828	Fast Food	354
1514	North Indian, Mughlai	334
331	Cafe	299
177	Bakery	218
1520	North Indian, Mughlai, Chinese	197
186	Bakery, Desserts	170
1749	Street Food	149