In [1]:

```
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('zomato.csv',encoding='latin-1')
df.head()
```

Out[2]:

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

5 rows × 21 columns

In [3]:

```
df.columns
```

Out[3]:

```
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 [4]:
```
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
```

In [5]:
```
df.describe()
```

Out[5]:

	Restaurant ID	Country Code	Longitude	Latitude	Average Cost for two	Price range	Aggregate rating	Vot
count	9.551000e+03	9551.000000	9551.000000	9551.000000	9551.000000	9551.000000	9551.000000	9551.0000
mean	9.051128e+06	18.365616	64.126574	25.854381	1199.210763	1.804837	2.666370	156.9097
std	8.791521e+06	56.750546	41.467058	11.007935	16121.183073	0.905609	1.516378	430.1691
min	5.300000e+01	1.000000	-157.948486	-41.330428	0.000000	1.000000	0.000000	0.0000
25%	3.019625e+05	1.000000	77.081343	28.478713	250.000000	1.000000	2.500000	5.0000
50%	6.004089e+06	1.000000	77.191964	28.570469	400.000000	2.000000	3.200000	31.0000
75%	1.835229e+07	1.000000	77.282006	28.642758	700.000000	2.000000	3.700000	131.0000
max	1.850065e+07	216.000000	174.832089	55.976980	800000.000000	4.000000	4.900000	10934.0000

In Data Analysis what all things we do

1.missing values 2.Explore Above the Numerical Variables 3.Explore about categorical Variables 4.Finding Relationship between features

In [6]:
```
df.isnull().sum()
```

Out[6]:
```
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 [7]:
```
[features for features in df.columns if df[features].isnull().sum()>0]
```

Out[7]:
```
['Cuisines']
```

In [8]:
```
df_country=pd.read_excel('Country-Code.xlsx')
df_country.head()
```

Out[8]:

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

In [9]:
```
df.columns
```

Out[9]:
```
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 [10]:
```
df_final=pd.merge(df,df_country,on='Country Code',how='left')
```

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

Out[11]:

	Restaurant ID	Restaurant Name	Country Code	City	Address	Locality	Locality Verbose	Longitude	Latitude
0	6317637	Le Petit Souffle	162	Makati City	Third Floor, Century	Century City Mall,	Century City Mall, Poblacion,	121.027535	14.565443

	Restaurant ID	Restaurant Name	Country Code	City	Address	Locality	Locality Verbose	Longitude	Latitude
					City Mall, Kalayaan Avenu...	Poblacion, Makati City	Makati City, Mak...		
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
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
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
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

5 rows × 22 columns

In [12]:
```
df_final.dtypes
```

Out[12]:
```
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 [13]:
```
df_final.columns
```

Out[13]:
```
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 [14]:
```
country_names=df_final.Country.value_counts().index
```

In [15]:
```
country_names
```

Out[15]:
```
Index(['India', 'United States', 'United Kingdom', 'Brazil', 'UAE',
       'South Africa', 'New Zealand', 'Turkey', 'Australia', 'Phillipines',
       'Indonesia', 'Singapore', 'Qatar', 'Sri Lanka', 'Canada'],
      dtype='object')
```

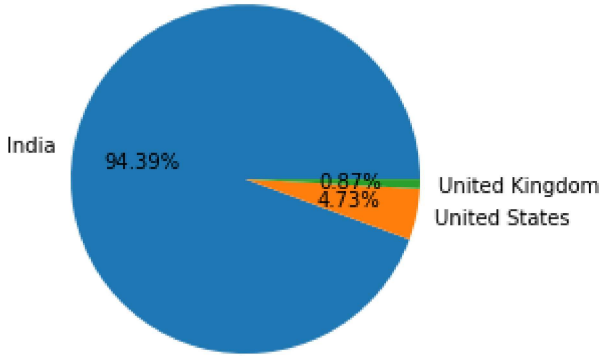
In [16]:
```
country_val=df_final.Country.value_counts().values
```

In [17]:
```
country_val
```

Out[17]:
```
array([8652,  434,   80,   60,   60,   60,   40,   34,   24,   22,   21,
         20,   20,   20,    4], dtype=int64)
```

In [18]:
```
## pie chart top 5 country
plt.pie(country_val[:3],labels=country_names[:3],autopct='%1.2f%%')
```

Out[18]:
```
([<matplotlib.patches.Wedge at 0x1cb1b61ddf0>,
  <matplotlib.patches.Wedge at 0x1cb1bcbd5b0>,
  <matplotlib.patches.Wedge at 0x1cb1bcbdcd0>],
 [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%')])
```



Observation:Zomato maximumum records or transaction is from india after that USA and then UK

In [19]:
```
ratings=df_final.groupby(['Aggregate rating','Rating color','Rating text']).size().reset_index(
```

In [20]:
```
ratings
```

Out[20]:

	Aggregate rating	Rating color	Rating text	rating Count
0	0.0	White	Not rated	2148

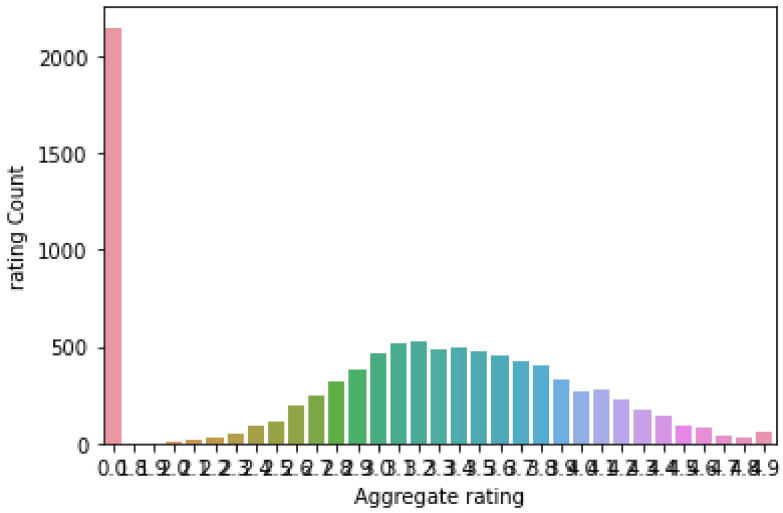
	Aggregate rating	Rating color	Rating text	rating Count
1	1.8	Red	Poor	1
2	1.9	Red	Poor	2
3	2.0	Red	Poor	7
4	2.1	Red	Poor	15
5	2.2	Red	Poor	27
6	2.3	Red	Poor	47
7	2.4	Red	Poor	87
8	2.5	Orange	Average	110
9	2.6	Orange	Average	191
10	2.7	Orange	Average	250
11	2.8	Orange	Average	315
12	2.9	Orange	Average	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

conclusion:

1.whenever the rating is between 4.5 to 4.9 ---> Excellent 2.whenever the rating is between 4.0 to 3.4 ---> very good 3.whenever the rating is between 3.5 to 3.9 ---> good 4.whenever the rating is between 3.0 to 3.4 ---> Average 5.whenever the rating is between 2.5 to 2.9 ---> Average 6.whenever the rating is between 2.0 to 2.4 ---> poor

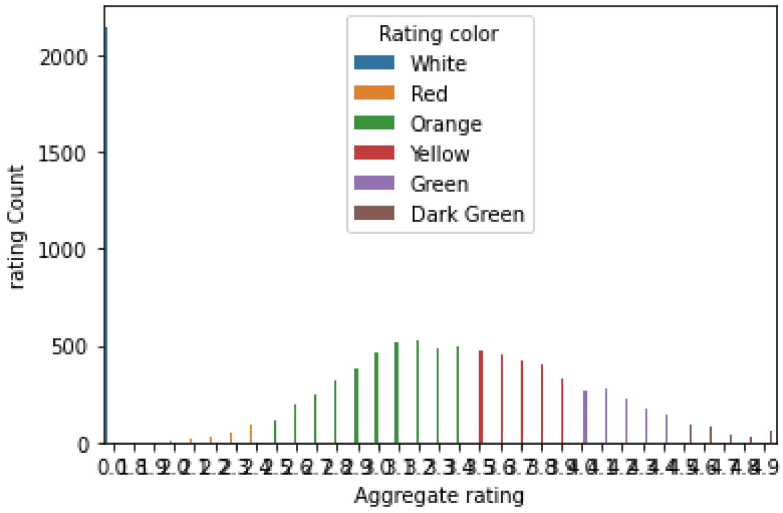
In [21]:
```
sns.barplot(x='Aggregate rating',y='rating Count',data=ratings)
```

Out[21]:
```
<AxesSubplot:xlabel='Aggregate rating', ylabel='rating Count'>
```



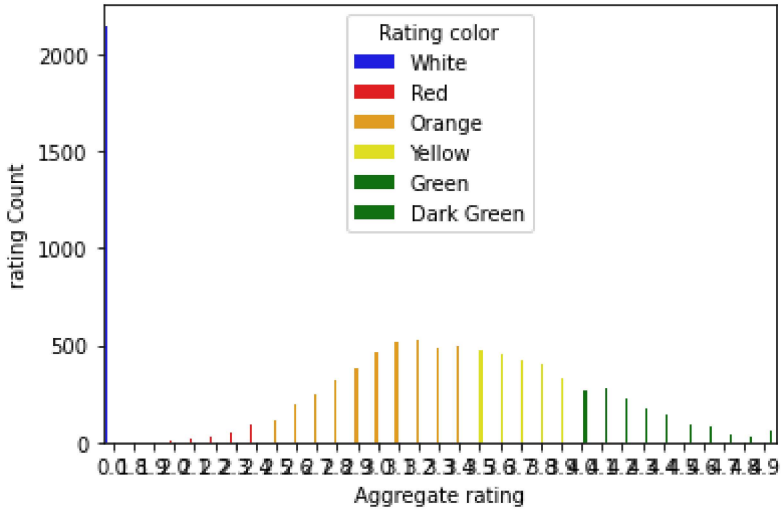
In [22]:
```
sns.barplot(x='Aggregate rating',y='rating Count',hue='Rating color',data=ratings)
```

Out[22]:
```
<AxesSubplot:xlabel='Aggregate rating', ylabel='rating Count'>
```



In [23]:
```
##Arranging the color with respect to their name
sns.barplot(x='Aggregate rating',y='rating Count',hue='Rating color',data=ratings,palette=['blu
```

Out[23]:
```
<AxesSubplot:xlabel='Aggregate rating', ylabel='rating Count'>
```

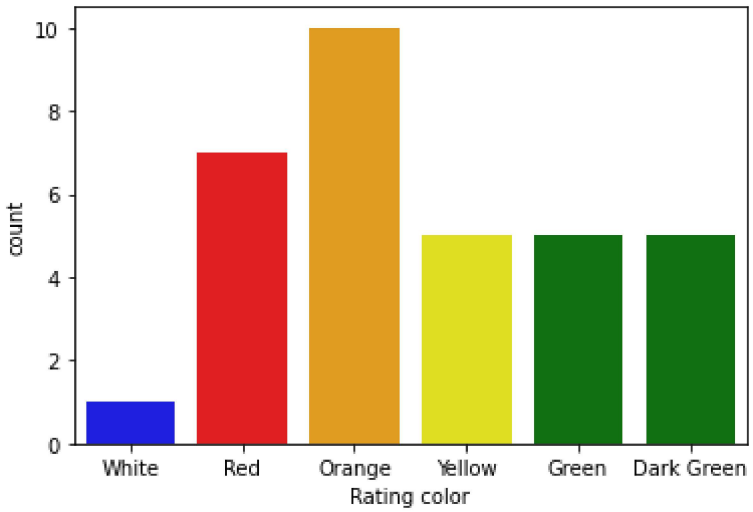


observation

1.Not Ratted is very high 2.maximum number of ratting is between 2.5 to 3.4

In [24]:
```
## count plot
sns.countplot(x='Rating color',data=ratings,palette=['blue','red','orange','yellow','green','gr
```

Out[24]:
```
<AxesSubplot:xlabel='Rating color', ylabel='count'>
```



In [25]:
```
### find the countries names that has been given zero rating
df_final[df_final['Rating color']=='White'].groupby(['Aggregate rating','Country']).size().rese
```

Out[25]:

	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

Obsevation: Maximum number of 0 ratings froms indian customers

In [26]:
```
##find out which currency is used by which country
df_final.columns
```

Out[26]:
```
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 [27]:
```
df_final[['Country','Currency']].groupby(['Country','Currency']).size().reset_index()
```

Out[27]:

	Country	Currency	0
0	Australia	Dollar($)	24
1	Brazil	Brazilian Real(R$)	60

	Country	Currency	0
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 [28]:
```
##find out which countries do Have Online delivery
df_final[df_final['Has Online delivery']=='YES'].Country.value_counts()
```

Out[28]:
```
Series([], Name: Country, dtype: int64)
```

In [29]:
```
df_final[['Has Online delivery','Country']].groupby(['Has Online delivery','Country']).size().r
```

Out[29]:

	Has Online delivery	Country	0
0	No	Australia	24
1	No	Brazil	60
2	No	Canada	4
3	No	India	6229
4	No	Indonesia	21
5	No	New Zealand	40
6	No	Phillipines	22
7	No	Qatar	20
8	No	Singapore	20
9	No	South Africa	60
10	No	Sri Lanka	20
11	No	Turkey	34
12	No	UAE	32
13	No	United Kingdom	80
14	No	United States	434
15	Yes	India	2423
16	Yes	UAE	28

Observation :

1.Online deliveries are available in india and UAE

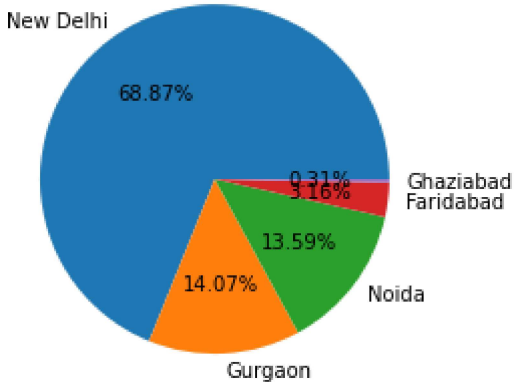
In [30]:
```
##create a pie chart for city distribution
```

In [31]:
```
city_label=df_final.City.value_counts().index
```

In [32]:
```
city_val=df_final.City.value_counts().values
```

In [33]:
```
plt.pie(city_val[:5],labels=city_label[:5],autopct='%1.2f%%')
```

Out[33]:
```
([<matplotlib.patches.Wedge at 0x1cb1d8b1e50>,
  <matplotlib.patches.Wedge at 0x1cb1d8c0640>,
  <matplotlib.patches.Wedge at 0x1cb1d8c0d60>,
  <matplotlib.patches.Wedge at 0x1cb1d8cc4f0>,
  <matplotlib.patches.Wedge at 0x1cb1d8ccc10>],
 [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%')])
```

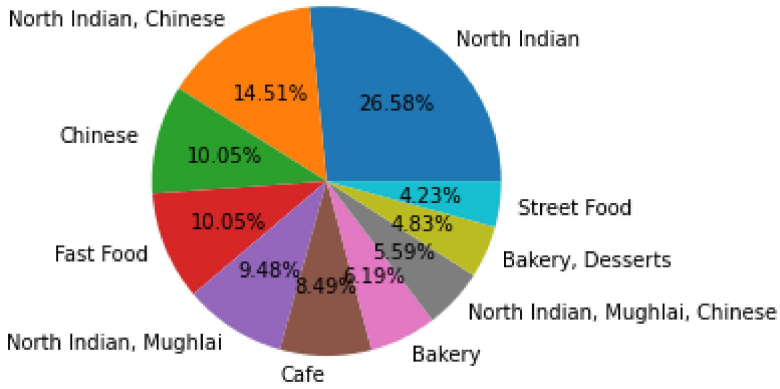


In [34]:
```
## find the top 10 cuisines
cuisines_label=df_final.Cuisines.value_counts().index
cuisines_val=df_final.Cuisines.value_counts().values
```

In [35]:
```
plt.pie(cuisines_val[:10],labels=cuisines_label[:10],autopct='%1.2f%%')
```

Out[35]:
```
([<matplotlib.patches.Wedge at 0x1cb1d90ff70>,
  <matplotlib.patches.Wedge at 0x1cb1d91a730>,
  <matplotlib.patches.Wedge at 0x1cb1d91ae50>,
  <matplotlib.patches.Wedge at 0x1cb1d9295b0>,
  <matplotlib.patches.Wedge at 0x1cb1d929cd0>,
  <matplotlib.patches.Wedge at 0x1cb1d936430>,
  <matplotlib.patches.Wedge at 0x1cb1d936b50>,
  <matplotlib.patches.Wedge at 0x1cb1d9412b0>,
  <matplotlib.patches.Wedge at 0x1cb1d9419d0>,
  <matplotlib.patches.Wedge at 0x1cb1d94f130>],
 [Text(0.7383739846958008, 0.8153550507137645, 'North Indian'),
  Text(-0.5794679314239953, 0.9349956772366362, 'North Indian, Chinese'),
```

```
 Text(-1.067309479615702, 0.26617752482593154, 'Chinese'),
 Text(-1.0185984499802057, -0.4152796620326146, 'Fast Food'),
 Text(-0.5935788454809928, -0.9261015895664211, 'North Indian, Mughlai'),
 Text(-0.005887079599915552, -1.0999842463843672, 'Cafe'),
 Text(0.4842062514572988, -0.9876964645323336, 'Bakery'),
 Text(0.808736477166136, -0.7456174022251013, 'North Indian, Mughlai, Chinese'),
 Text(1.0055375294202338, -0.44597564611473206, 'Bakery, Desserts'),
 Text(1.090298995560443, -0.14576728123927227, 'Street Food')],
[Text(0.4027494461977095, 0.4447391185711442, '26.58%'),
 Text(-0.316073417140361, 0.5099976421290743, '14.51%'),
 Text(-0.5821688070631101, 0.14518774081414446, '10.05%'),
 Text(-0.5555991545346576, -0.22651617929051704, '10.05%'),
 Text(-0.32377027935326874, -0.5051463215816842, '9.48%'),
 Text(-0.003211134327226664, -0.5999914071187457, '8.49%'),
 Text(0.26411250079489024, -0.5387435261085456, '6.19%'),
 Text(0.441128987545165, -0.40670040121369155, '5.59%'),
 Text(0.5484750160474001, -0.24325944333530836, '4.83%'),
 Text(0.5947085430329688, -0.07950942613051214, '4.23%')])
```



In [45]:

```
df_final.Cuisines.value_counts()
#.sort_values(ascending=False).head(10)
```

Out[45]:

```
North Indian                                              936
North Indian, Chinese                                     511
Chinese                                                   354
Fast Food                                                 354
North Indian, Mughlai                                     334
                                                         ...
Bengali, Fast Food                                          1
North Indian, Rajasthani, Asian                             1
Chinese, Thai, Malaysian, Indonesian                        1
Bakery, Desserts, North Indian, Bengali, South Indian       1
Italian, World Cuisine                                      1
Name: Cuisines, Length: 1825, dtype: int64
```

In [ ]:

In [ ]: