In [2]:

```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
```

In [3]:

```
from google.colab import drive
```

In [4]:

```
drive.mount('/content/drive')
```

```
Mounted at /content/drive
```

In [5]:

```
df1=pd.read_excel('/content/drive/MyDrive/data.xlsx')
```

In [6]:

```
df2=pd.read_excel('/content/drive/MyDrive/variable description.xlsx')
```

In [7]:

```
df3=pd.read_excel('/content/drive/MyDrive/Country-Code.xlsx')
```

In [8]:

```
df1.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 9551 entries, 0 to 9550
Data columns (total 19 columns):
 #   Column                Non-Null Count  Dtype  
---  ------                --------------  -----  
 0   Restaurant ID         9551 non-null   int64  
 1   Restaurant Name       9550 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  Price range           9551 non-null   int64  
 15  Aggregate rating      9551 non-null   float64
 16  Rating color          9551 non-null   object 
 17  Rating text           9551 non-null   object 
 18  Votes                 9551 non-null   int64  
dtypes: float64(3), int64(5), object(11)
memory usage: 1.4+ MB
```

In [9]:

```
df2.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 19 entries, 0 to 18
Data columns (total 2 columns):
 #   Column       Non-Null Count  Dtype 
---  ------       --------------  ----- 
 0   Variable     19 non-null     object
 1   Description  19 non-null     object
dtypes: object(2)
memory usage: 432.0+ bytes
```

In [10]:

```
df2
```

Out[10]:

	Variable	Description
0	Restaurant ID	Identification Number
1	Restaurant Name	Name Of the Restaurant
2	Country Code	Country code
3	City	City Name of the Restaurant
4	Address	Detailed address of the restaurant
5	Locality	Shot Address Of the Restaurant
6	Locality Verbose	Long Address of the Restaurant
7	Longitude	Longitude
8	Latitude	Latitude
9	Cuisines	Types Of Cuisines Served
10	Average Cost for two	Average Cost if two people visit the Restaurant
11	Currency	Local currency
12	Has Table booking	Can we book tables in Restaurant? Yes/No
13	Has Online delivery	Can we have online delivery ? Yes/No
14	Price range	Categorized price between 1 -4
15	Aggregate rating	Categorizing ratings between 1-5
16	Rating color	Different colors representing Customer Rating
17	Rating text	Different Rating like Excellent, Very Good ,Go...
18	Votes	No.Of Votes received by restaurant from custom...

In [11]:

```
df3.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 15 entries, 0 to 14
Data columns (total 2 columns):
 #   Column        Non-Null Count  Dtype 
---  ------        --------------  ----- 
 0   Country Code  15 non-null     int64 
 1   Country       15 non-null     object
dtypes: int64(1), object(1)
memory usage: 368.0+ bytes
```

df2 and df3 is clean.df1 needs little cleaning

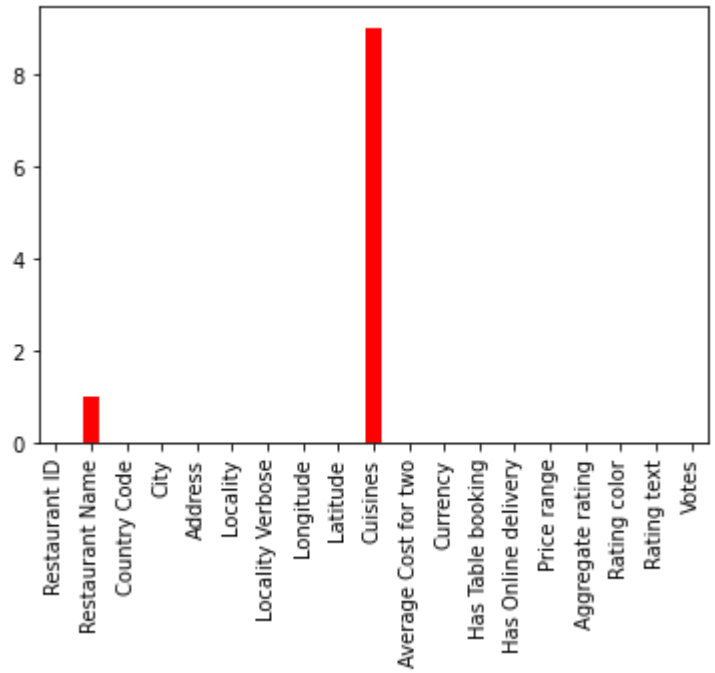
1. Cleaning

In [12]:

```
df1.isnull().sum().plot(kind='bar',color='red')
```

Out[12]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f0839d0e610>
```



In [13]:

```
df1.isnull().sum()
```

Out[13]:

```
Restaurant ID         0
Restaurant Name       1
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
Price range           0
Aggregate rating      0
Rating color          0
Rating text           0
Votes                 0
dtype: int64
```

In [14]:

```
df1["Restaurant Name"]=df1["Restaurant Name"].fillna(df1["Restaurant Name"].mode
()[0])
df1["Cuisines"]=df1["Cuisines"].fillna(df1["Cuisines"].mode()[0])
```

In [15]:

```
df1.isnull().sum()
```

Out[15]:

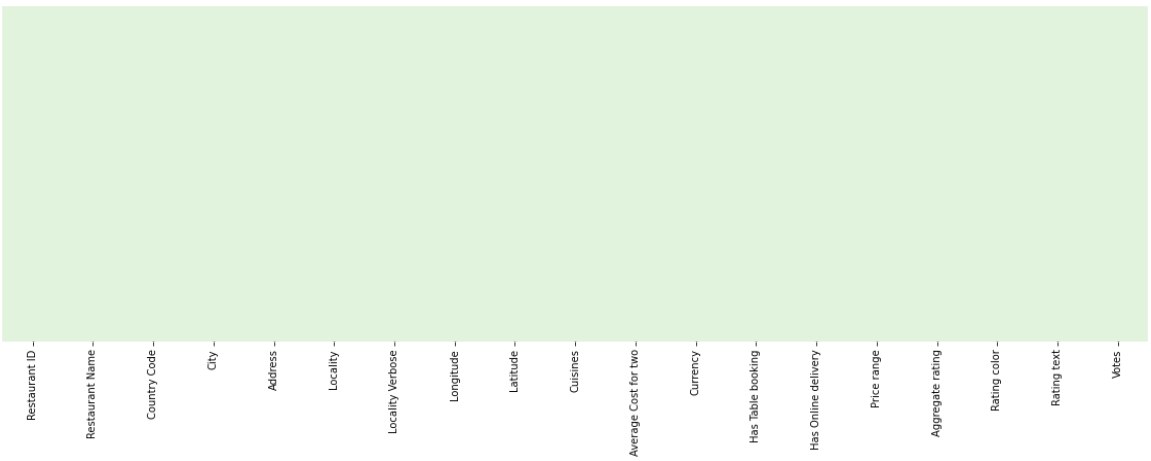
```
Restaurant ID           0
Restaurant Name         0
Country Code            0
City                    0
Address                 0
Locality                0
Locality Verbose        0
Longitude               0
Latitude                0
Cuisines                0
Average Cost for two    0
Currency                0
Has Table booking       0
Has Online delivery     0
Price range             0
Aggregate rating        0
Rating color            0
Rating text             0
Votes                   0
dtype: int64
```

In [16]:

```
colormap = sns.color_palette("Greens")
plt.rcParams['figure.figsize']=(20,6)
sns.heatmap(df1.isnull(),yticklabels=False,cbar=False,cmap=colormap)
```

Out[16]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f08399ffbd0>
```



Null Values from column "Restaurant Name" AND Cuisines has been replaced with the mode of the columns since both the columns have categorial variable

1.Explore the geographical distribution of the restaurants, finding out the cities with maximum / minimum number of restaurants.

In [17]:

```
df=df1.merge(df3,on='Country Code',how='left')
#merging with df3 to get the country name
```

In [18]:

```
b=df.groupby(["Country"]).agg({ "Restaurant ID" : "count"}).reset_index().rename
({"Restaurant ID" :"No. of Restaurant"},axis='columns').sort_values(by='No. of R
estaurant',ascending=False)
b=b.head(5)
```

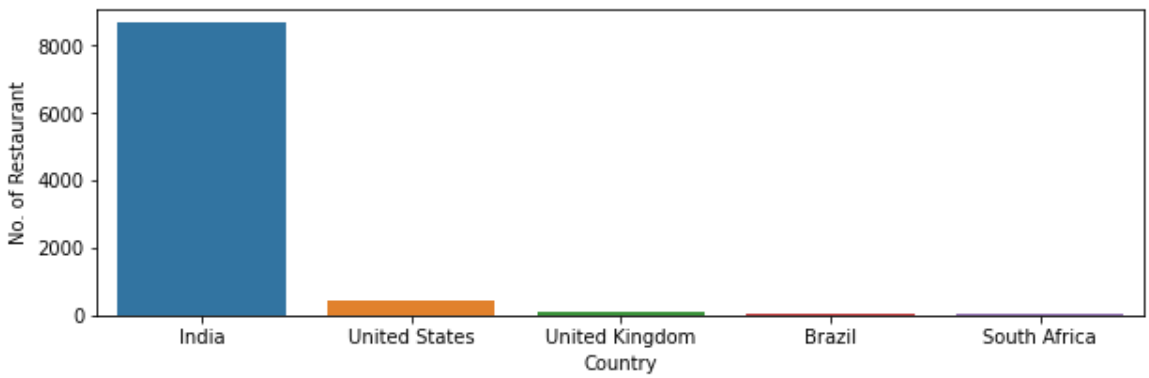
In [22]:

```
plt.rcParams['figure.figsize']=(10,3)
sns.barplot("Country","No. of Restaurant",data=b)
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: Fu
tureWarning: Pass the following variables as keyword args: x, y. Fro
m version 0.12, the only valid positional argument will be `data`, a
nd passing other arguments without an explicit keyword will result i
n an error or misinterpretation.
  FutureWarning
```

Out[22]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f0839a87f90>
```

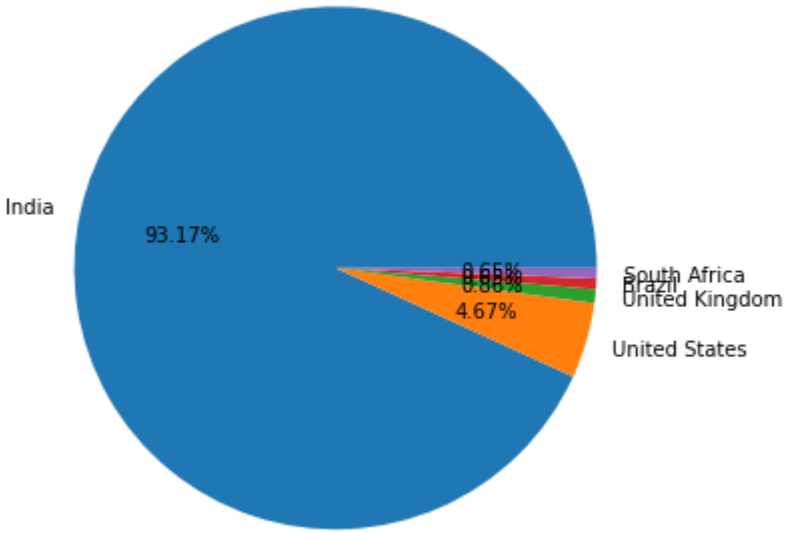


In [23]:

```
plt.rcParams['figure.figsize']=(10,3)
plt.pie(b["No. of Restaurant"],labels=b["Country"],autopct='%1.2f%%',radius=2)
```

Out[23]:

```
([<matplotlib.patches.Wedge at 0x7f0839c5b1d0>,
  <matplotlib.patches.Wedge at 0x7f0839c5b910>,
  <matplotlib.patches.Wedge at 0x7f0839c631d0>,
  <matplotlib.patches.Wedge at 0x7f0839c63ad0>,
  <matplotlib.patches.Wedge at 0x7f0839b2e650>],
 [Text(-2.14958641544394, 0.4682715478639219, 'India'),
  Text(2.113007155070432, -0.6125363357558145, 'United States'),
  Text(2.187120187817026, -0.237708401288674, 'United Kingdom'),
  Text(2.1959220340232783, -0.13388958320597136, 'Brazil'),
  Text(2.1995467728848292, -0.04465415873057457, 'South Africa')],
 [Text(-1.1725016811512399, 0.2554208442894119, '93.17%'),
  Text(1.1525493573111445, -0.3341107285940806, '4.67%'),
  Text(1.192974647900196, -0.12965912797564036, '0.86%'),
  Text(1.197775654921788, -0.07303068174871163, '0.65%'),
  Text(1.1997527852099066, -0.024356813853040674, '0.65%')])
```



In [24]:

```
#checking distribution at more granular level(City)
a=df.groupby(["City"]).agg({ "Restaurant ID" : "count"}).reset_index().rename({
"Restaurant ID" :"No. of Restaurant"},axis='columns').sort_values(by='No. of Res
taurant',ascending=False)
a
```

Out[24]:

	City	No. of Restaurant
88	New Delhi	5473
50	Gurgaon	1118
89	Noida	1080
43	Faridabad	251
48	Ghaziabad	25
...	...	...
37	Dicky Beach	1
68	Lorn	1
107	Quezon City	1
66	Lincoln	1
65	Lakeview	1

141 rows × 2 columns

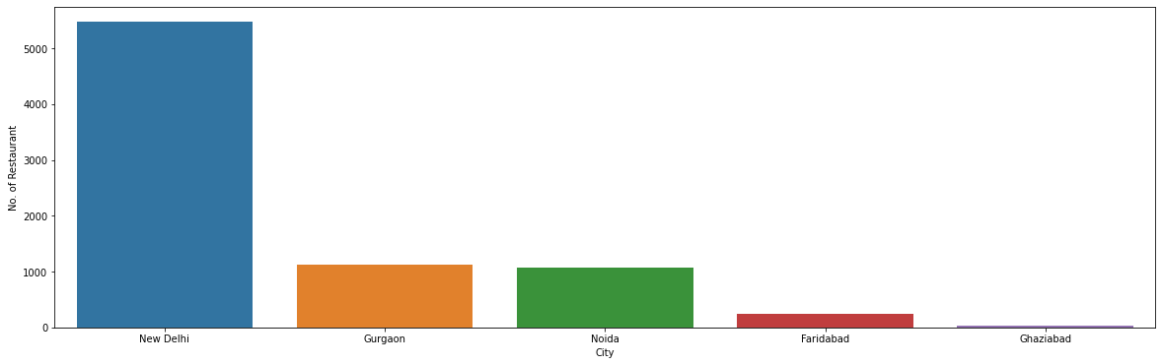
In [25]:

```
#Top 5 across cities
plt.rcParams['figure.figsize']=(20,6)
sns.barplot("City","No. of Restaurant",data=a.head(5))
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: Fu
tureWarning: Pass the following variables as keyword args: x, y. Fro
m version 0.12, the only valid positional argument will be `data`, a
nd passing other arguments without an explicit keyword will result i
n an error or misinterpretation.
  FutureWarning
```

Out[25]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f0839b4c390>
```

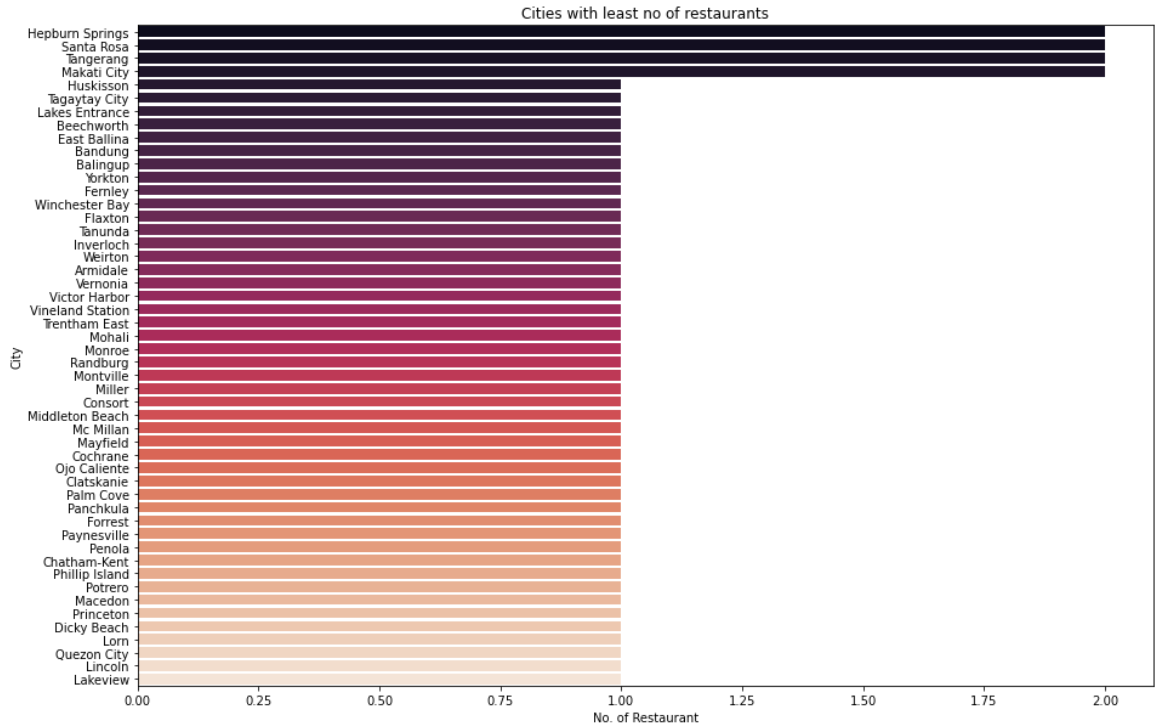


In [28]:

```
plt.rcParams['figure.figsize']=(15,10)
sns.barplot(x='No. of Restaurant',y='City',data=a.tail(50),palette='rocket')
plt.title("Cities with least no of restaurants")
```

Out[28]:

```
Text(0.5, 1.0, 'Cities with least no of restaurants')
```



Explore how ratings are distributed overall

In [104]:

```
z=df1.groupby(['Aggregate rating','Rating color','Rating text']).size().reset_in
dex().rename(columns={0:"rating ct"})
z
```

Out[104]:

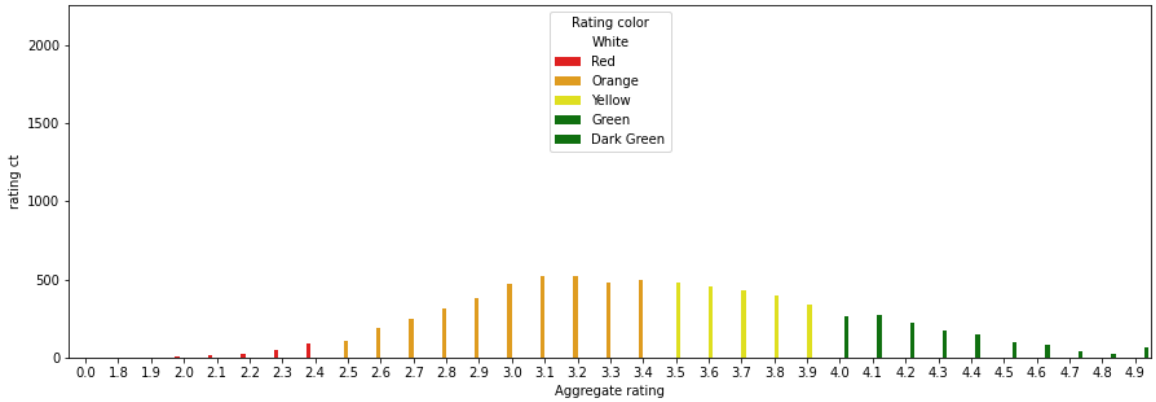
	Aggregate rating	Rating color	Rating text	rating ct
0	0.0	White	Not rated	2148
1	1.8	Red	Poor	1
2	1.9	Red	Poor	2
3	2.0	Red	Poor	7
4	2.1	Red	Poor	15
5	2.2	Red	Poor	27
6	2.3	Red	Poor	47
7	2.4	Red	Poor	87
8	2.5	Orange	Average	110
9	2.6	Orange	Average	191
10	2.7	Orange	Average	250
11	2.8	Orange	Average	315
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 [34]:

```
plt.rcParams['figure.figsize']=(15,5)
sns.barplot(x='Aggregate rating',y='rating ct',data=z,palette=["white","red","orange","yellow","green","green"],hue='Rating color')
```

Out[34]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f0835be0290>
```



Restaurant franchising is a thriving venture. So, it is very important to explore the franchise with most national presence

In [35]:

```
a=df.groupby(["Country","Restaurant Name"]).agg({ "Restaurant ID" : "count"}).reset_index().rename({"Restaurant ID" :"No. of Restaurant"},axis='columns').sort_values(by='No. of Restaurant',ascending=False)
a
```

Out[35]:

	Country	Restaurant Name	No. of Restaurant
1061	India	Cafe Coffee Day	84
1975	India	Domino's Pizza	79
5522	India	Subway	63
2486	India	Green Chick Chop	51
3689	India	McDonald's	48
...	...	...	...
2639	India	Hawai Adda	1
2637	India	Havemore	1
2636	India	Haveliram	1
2635	India	Hauz Khas Social	1
7471	United States	Zunzi's	1

7472 rows × 3 columns

In [36]:

```
a["Country"].unique()
```

Out[36]:

```
array(['India', 'United States', 'UAE', 'United Kingdom', 'Indonesi
a',
       'Turkey', 'Phillipines', 'Brazil', 'Australia', 'South Afric
a',
       'Sri Lanka', 'Singapore', 'Qatar', 'New Zealand', 'Canada'],
      dtype=object)
```

In [37]:

```
a["rank"]=a.groupby(['Country'])['No. of Restaurant'].rank(method='first',ascend
ing=False)
```

In [38]:

```
a=a[a["rank"]==1]
```

In [39]:

```
a["Country-Restaurant"]=a["Country"]+"-"+a["Restaurant Name"]
```

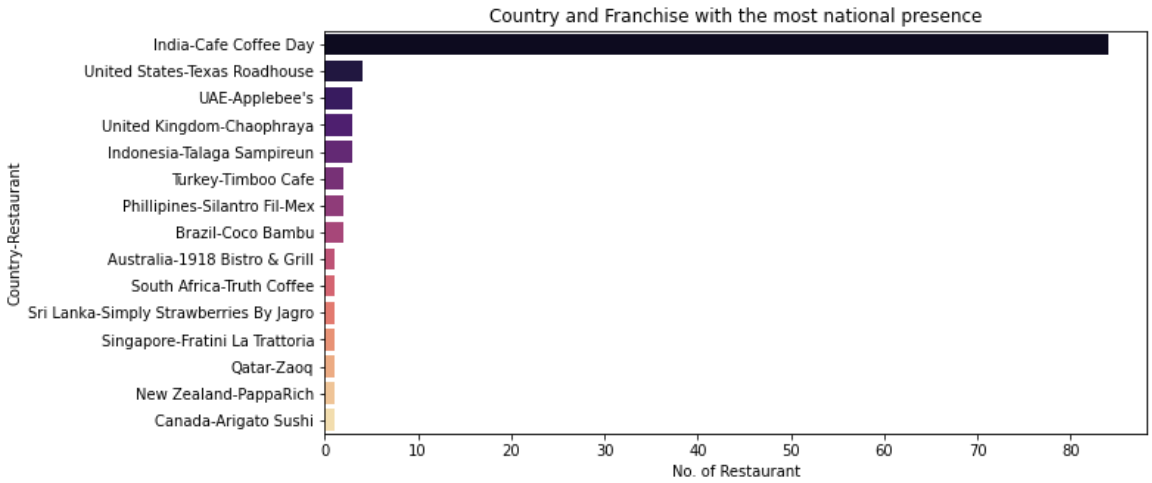
```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:1: Sett
ingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pand
as-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-co
py
  """Entry point for launching an IPython kernel.
```

In [42]:

```
plt.rcParams['figure.figsize']=(10,5)
sns.barplot(x='No. of Restaurant',y='Country-Restaurant',data=a,palette='magma')
plt.title("Country and Franchise with the most national presence")
```

Out[42]:

```
Text(0.5, 1.0, 'Country and Franchise with the most national presenc
e')
```



Find out the ratio between restaurants that allow table booking vs. those that do not allow table booking

In [43]:

```
ratio=df.groupby("Has Table booking").agg({"Restaurant ID": "count"}).reset_inde
x().rename({"Restaurant ID":"No. of Restaurant"},axis=1)
ratio
```

Out[43]:

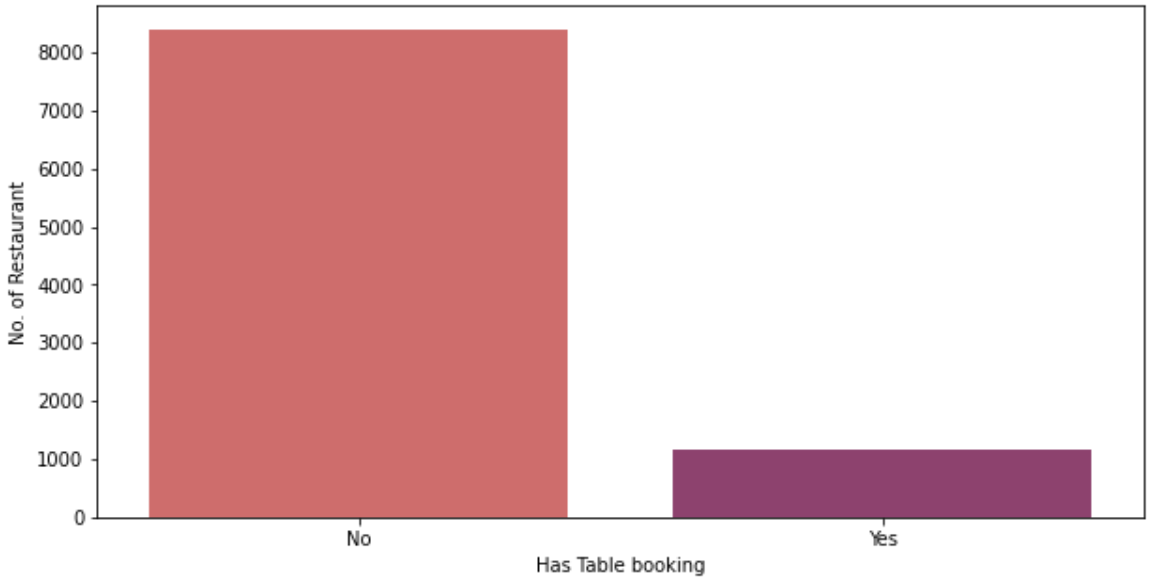
	Has Table booking	No. of Restaurant
0	No	8393
1	Yes	1158

In [44]:

```
sns.barplot(x='Has Table booking',y='No. of Restaurant',data=ratio,palette='flare')
```

Out[44]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f0834df23d0>
```



In [45]:

```
#Conclusion:  No of restaurants which provides only table booking very less as compared to the one which does not
#Ratio:0.13
```

In [46]:

```
int(ratio[ratio["Has Table booking"]=='Yes']["No. of Restaurant"])/int(ratio[ratio["Has Table booking"]=='No']["No. of Restaurant"])
```

Out[46]:

```
0.13797211962349576
```

In [47]:

```
x=pd.DataFrame()
```

Find out the percentage of restaurants providing online delivery

In [48]:

```
online=df.groupby("Has Online delivery").agg({"Restaurant ID": "count"}).reset_index().rename({"Restaurant ID":"No. of Restaurant"},axis=1)
online
```

Out[48]:

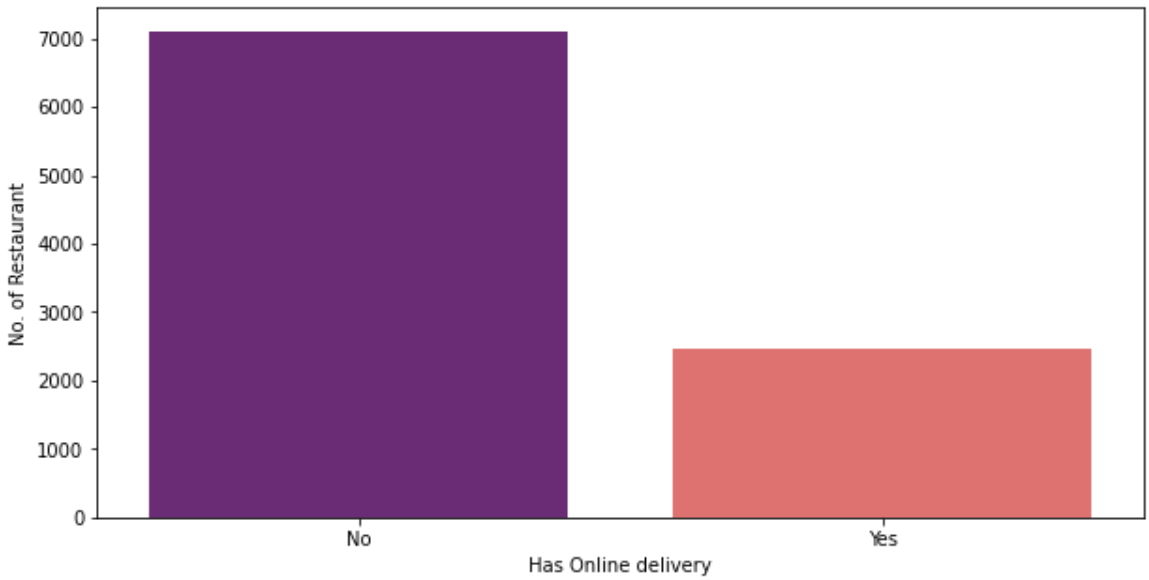
	Has Online delivery	No. of Restaurant
0	No	7100
1	Yes	2451

In [49]:

```
sns.barplot(x='Has Online delivery',y='No. of Restaurant',data=online,palette='magma')
```

Out[49]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f0834dddd10>
```



In [50]:

```
int(online[online["Has Online delivery"]=='Yes']["No. of Restaurant"])/int(online[online["Has Online delivery"]=='No']["No. of Restaurant"])
```

Out[50]:

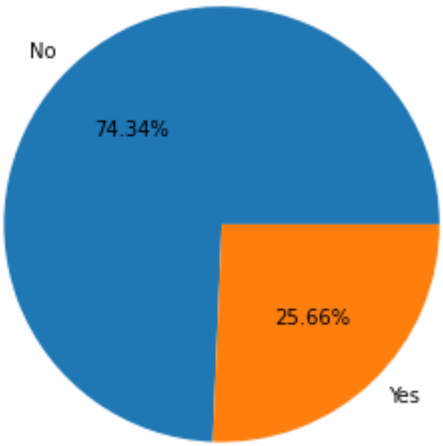
```
0.3452112676056338
```

In [51]:

```
plt.pie(online["No. of Restaurant"],labels=online["Has Online delivery"],autopct
='%1.2f%%')
```

Out[51]:

```
([<matplotlib.patches.Wedge at 0x7f0834d05350>,
  <matplotlib.patches.Wedge at 0x7f0834d05ad0>],
 [Text(-0.7614681082348079, 0.7938301582462732, 'No'),
  Text(0.7614681825585476, -0.7938300869524803, 'Yes')],
 [Text(-0.4153462408553497, 0.4329982681343308, '74.34%'),
  Text(0.41534628139557145, -0.43299822924680736, '25.66%')])
```



In [52]:

```
#Conclusion:  74.34% restaurants  doesnt have online delivery
```

Calculate the difference in number of votes for the restaurants that deliver and the restaurants that do not deliver

In [53]:

```
diff=df.groupby("Has Online delivery").agg({"Votes" :"sum"}).reset_index()
diff
```

Out[53]:

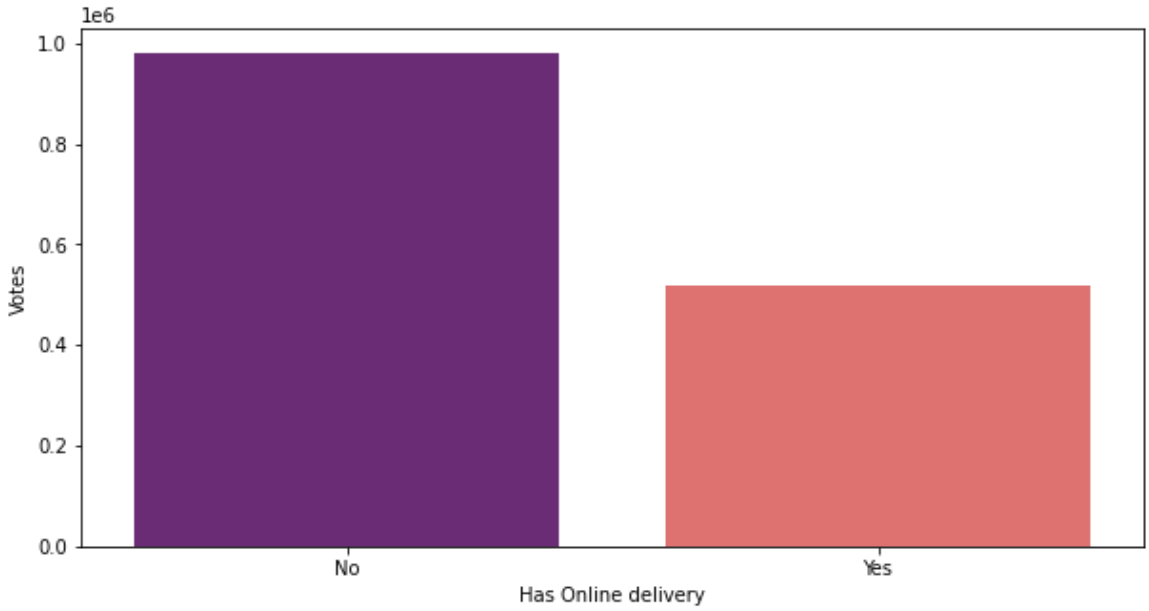
	Has Online delivery	Votes
0	No	980731
1	Yes	517914

In [54]:

```
sns.barplot(x='Has Online delivery',y='Votes',data=diff,palette='magma')
```

Out[54]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f0834ca5fd0>
```



In [55]:

```
int(diff[diff["Has Online delivery"]=='Yes']["Votes"])-int(diff[diff["Has Online delivery"]=='No']["Votes"])
```

Out[55]:

```
-462817
```

In [56]:

```
#there is difference of "462817" Votes between no of votes for the restaurant th
at delivers and the restaurant that do not
#Surpsisingly restaurant that do not deliver has more votes
```

week2:**Task**

What are the top 10 cuisines served across cities?

In [57]:

```
c=df["Cuisines"].value_counts().reset_index().head(10).rename({"index":"Cuisine
s","Cuisines" :"Count. of Cuisines"},axis='columns')
c
```

Out[57]:

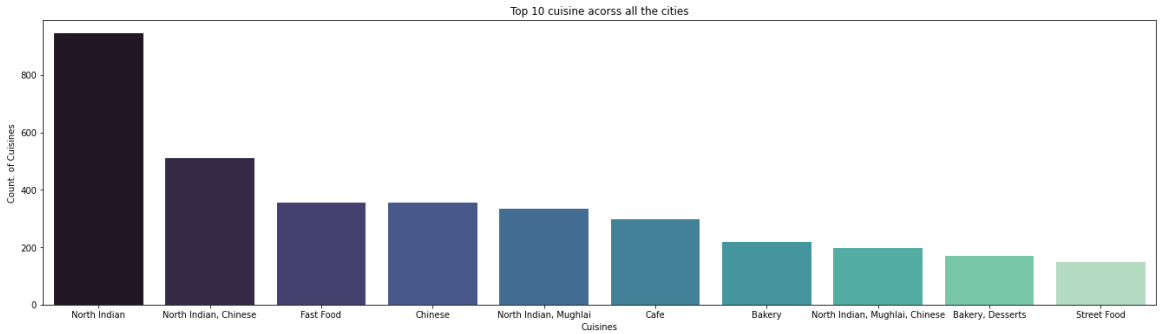
	Cuisines	Count. of Cuisines
0	North Indian	945
1	North Indian, Chinese	511
2	Fast Food	354
3	Chinese	354
4	North Indian, Mughlai	334
5	Cafe	299
6	Bakery	218
7	North Indian, Mughlai, Chinese	197
8	Bakery, Desserts	170
9	Street Food	149

In [65]:

```
plt.rcParams['figure.figsize']=(23,6)
sns.barplot(x='Cuisines',y='Count. of Cuisines',data=c,palette='mako')
plt.title("Top 10 cuisine acorss all the cities")
```

Out[65]:

```
Text(0.5, 1.0, 'Top 10 cuisine acorss all the cities')
```

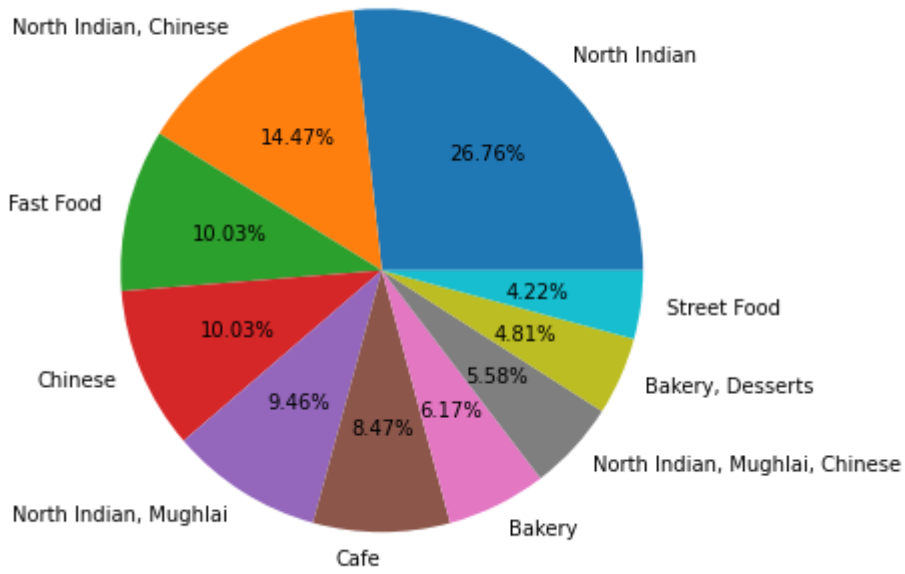


In [66]:

```
plt.pie(c["Count. of Cuisines"],labels=c["Cuisines"],autopct='%1.2f%%')
```

Out[66]:

```
([<matplotlib.patches.Wedge at 0x7f0834751d10>,
  <matplotlib.patches.Wedge at 0x7f08346df4d0>,
  <matplotlib.patches.Wedge at 0x7f08346dfd90>,
  <matplotlib.patches.Wedge at 0x7f08346e9550>,
  <matplotlib.patches.Wedge at 0x7f08346e9f90>,
  <matplotlib.patches.Wedge at 0x7f08346f5850>,
  <matplotlib.patches.Wedge at 0x7f08346f5e50>,
  <matplotlib.patches.Wedge at 0x7f08346ff650>,
  <matplotlib.patches.Wedge at 0x7f08346e9d90>,
  <matplotlib.patches.Wedge at 0x7f08346f5650>],
 [Text(0.7335674636606228, 0.819682119030616, 'North Indian'),
  Text(-0.5893433631658553, 0.9288026702655193, 'North Indian, Chine
se'),
  Text(-1.0695669205605947, 0.25695642129070567, 'Fast Food'),
  Text(-1.0156578084359724, -0.42242066256640176, 'Chinese'),
  Text(-0.5885167416932667, -0.9293266620229621, 'North Indian, Mugh
lai'),
  Text(-0.001468010939087877, -1.0999990204286014, 'Cafe'),
  Text(0.4870112613457864, -0.9863163951402139, 'Bakery'),
  Text(0.8101505161499852, -0.7440807356610658, 'North Indian, Mughl
ai, Chinese'),
  Text(1.0060114877036386, -0.44490548053301426, 'Bakery, Dessert
s'),
  Text(1.0903483121505928, -0.14539793048855149, 'Street Food')],
 [Text(0.4001277074512487, 0.44709933765306326, '26.76%'),
  Text(-0.32146001627228465, 0.5066196383266469, '14.47%'),
  Text(-0.5834001384875971, 0.14015804797674855, '10.03%'),
  Text(-0.553995168237803, -0.23041127049076454, '10.03%'),
  Text(-0.3210091318326909, -0.5069054520125248, '9.46%'),
  Text(-0.000800732395024783, -0.599999465688328, '8.47%'),
  Text(0.26564250618861074, -0.5379907609855711, '6.17%'),
  Text(0.4419002815363555, -0.40586221945149037, '5.58%'),
  Text(0.5487335387474391, -0.24267571665437138, '4.81%'),
  Text(0.5947354429912324, -0.07930796208466444, '4.22%')])
```



What is the maximum and minimum number of cuisines that a restaurant serves? Also, which is the most served cuisine across the restaurant for each city?

In [67]:

```
p=df.groupby(["City","Cuisines"]).agg({"Cuisines" :"count"}).rename({"Cuisines"
:"No. of Cuisines"},axis='columns').reset_index()
p
```

Out[67]:

	City	Cuisines	No. of Cuisines
0	Abu Dhabi	American	2
1	Abu Dhabi	American, Desserts	1
2	Abu Dhabi	American, Mexican, Seafood	1
3	Abu Dhabi	Asian	1
4	Abu Dhabi	Chinese	1
...	...	...	...
3026	€¡stanbul	Restaurant Cafe	2
3027	€¡stanbul	Restaurant Cafe, Desserts	1
3028	€¡stanbul	Restaurant Cafe, Turkish, Desserts	1
3029	€¡stanbul	Turkish	1
3030	€¡stanbul	World Cuisine, Patisserie, Cafe	1

3031 rows × 3 columns

In [68]:

```
p["rank"]=p.groupby(['City'])['No. of Cuisines'].rank(method='first',ascending=False)
p=p[p["rank"]==1]
p
```

Out[68]:

	City	Cuisines	No. of Cuisines	rank
0	Abu Dhabi	American	2	1.0
28	Agra	North Indian, Mughlai	5	1.0
32	Ahmedabad	Cafe, American, Continental, Armenian, Fast Food	1	1.0
66	Albany	North Indian	3	1.0
79	Allahabad	North Indian, Chinese	3	1.0
...	...	...	...	...
3000	Weirton	Burger, Greek, Sandwich	1	1.0
3003	Wellington City	Cafe	3	1.0
3018	Winchester Bay	Burger, Seafood, Steak	1	1.0
3019	Yorkton	Asian	1	1.0
3022	€¡stanbul	Cafe	3	1.0

141 rows × 4 columns

What is the distribution cost across the restaurants?

In [69]:

```
df_cost=df.groupby("Restaurant Name").agg({"Average Cost for two" :"mean"}).reset_index().sort_values(by='Average Cost for two',ascending=True)
```

In [70]:

```
df_cost
```

Out[70]:

	Restaurant Name	Average Cost for two
1746	Cookie Shoppe	0.0
486	Atmosphere Grill Cafe Sheesha	0.0
2808	HI Lite Bar & Lounge	0.0
2852	Happy Joe's Pizza & Ice Cream	0.0
511	Azteca	0.0
...	...	...
505	Avec Moi Restaurant and Bar	350000.0
41	3 Wise Monkeys	450000.0
6170	Sushi Masa	500000.0
5897	Skye	800000.0
5594	Satoo - Hotel Shangri-La	800000.0

7445 rows × 2 columns

How ratings are distributed among the various factors?

In [71]:

```
df_rating=df[df["Aggregate rating"]==0]
```

In [72]:

```
df_rating["Country"].value_counts()
```

Out[72]:

```
India             2139
Brazil               5
United States        3
United Kingdom       1
Name: Country, dtype: int64
```

In [73]:

```
#most zero ratings is from India might be casue of the busines also reside in in
dia acc to this data
```

In [74]:

```
df_rating["City"].value_counts()
```

Out[74]:

```
New Delhi          1425
Noida               384
Gurgaon             228
Faridabad           100
SÌ£o Paulo             3
Ghaziabad              2
Davenport              2
Rio de Janeiro         1
BrasÌ_lia              1
Pocatello              1
Birmingham             1
Name: City, dtype: int64
```

In [75]:

```
#at city granularity max zero rating in from New Delhi
```

In [76]:

```
df_rating["Restaurant Name"].value_counts()
```

Out[76]:

```
Cafe Coffee Day            16
Baskin Robbins             12
Aggarwal Sweets            10
Bikaner Sweets              7
Green Chick Chop            7
                           ..
Gurgaon Mughlai Chicken     1
Shree Shyam Bhojnalaya      1
Best Pizza Hut              1
The Burger Chef             1
Damascena Coffee House      1
Name: Restaurant Name, Length: 1947, dtype: int64
```

In [77]:

```
# max zero rating is for CCD
```

Explain the factors in the data that may have an effect on ratings. For example, number of cuisines, cost, delivery option, etc.

In [78]:

```
df["Rating text"].unique()
```

Out[78]:

```
array(['Very Good', 'Excellent', 'Good', 'Average', 'Not rated', 'Po
or'],
      dtype=object)
```

In [79]:

```
price=df.groupby(["Price range","Rating text"]).agg({"Rating text" :"count"}).re
name({"Rating text":"ct of text"},axis='columns').reset_index()
price
```

Out[79]:

	Price range	Rating text	ct of text
0	1	Average	1898
1	1	Excellent	32
2	1	Good	608
3	1	Not rated	1700
4	1	Poor	62
5	1	Very Good	144
6	2	Average	1425
7	2	Excellent	69
8	2	Good	794
9	2	Not rated	402
10	2	Poor	98
11	2	Very Good	325
12	3	Average	313
13	3	Excellent	126
14	3	Good	498
15	3	Not rated	35
16	3	Poor	20
17	3	Very Good	416
18	4	Average	101
19	4	Excellent	74
20	4	Good	200
21	4	Not rated	11
22	4	Poor	6
23	4	Very Good	194

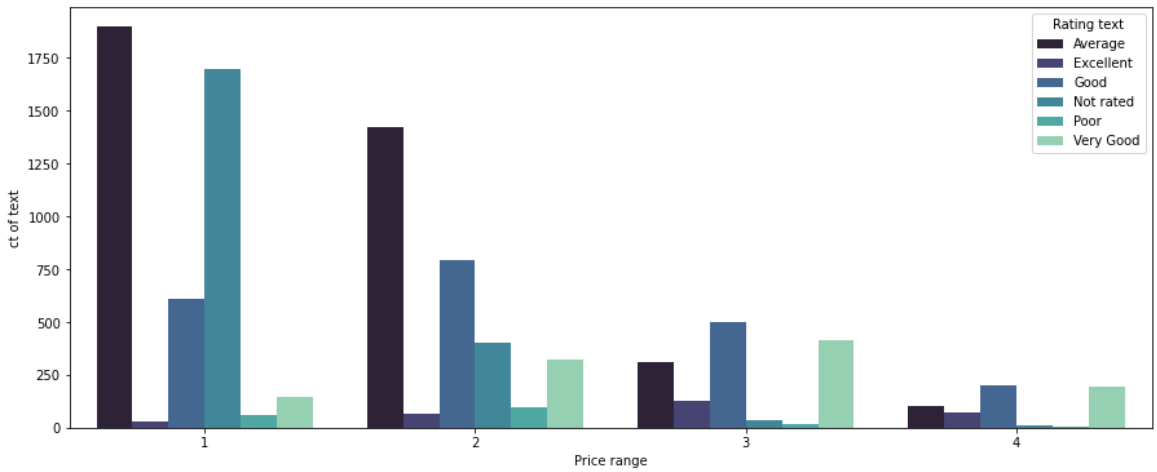
In [88]:

```
plt.rcParams['figure.figsize']=(15,6)
sns.barplot("Price range","ct of text",hue='Rating text',data=price,palette='mako')
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: FutureWarning: Pass the following variables as keyword args: x, y. From version 0.12, the only valid positional argument will be `data`, and passing other arguments without an explicit keyword will result in an error or misinterpretation.
  FutureWarning
```

Out[88]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f083434db10>
```



In [89]:

```
#Conclusion:Bad rating is max for cheap/low range restaurants
```

In [90]:

```
delivery=df.groupby(["Has Online delivery","Rating text"]).agg({"Rating text" :
"count"}).rename({"Rating text":"ct of text"},axis='columns').reset_index()
delivery
```

Out[90]:

	Has Online delivery	Rating text	ct of text
0	No	Average	2632
1	No	Excellent	262
2	No	Good	1282
3	No	Not rated	2052
4	No	Poor	70
5	No	Very Good	802
6	Yes	Average	1105
7	Yes	Excellent	39
8	Yes	Good	818
9	Yes	Not rated	96
10	Yes	Poor	116
11	Yes	Very Good	277

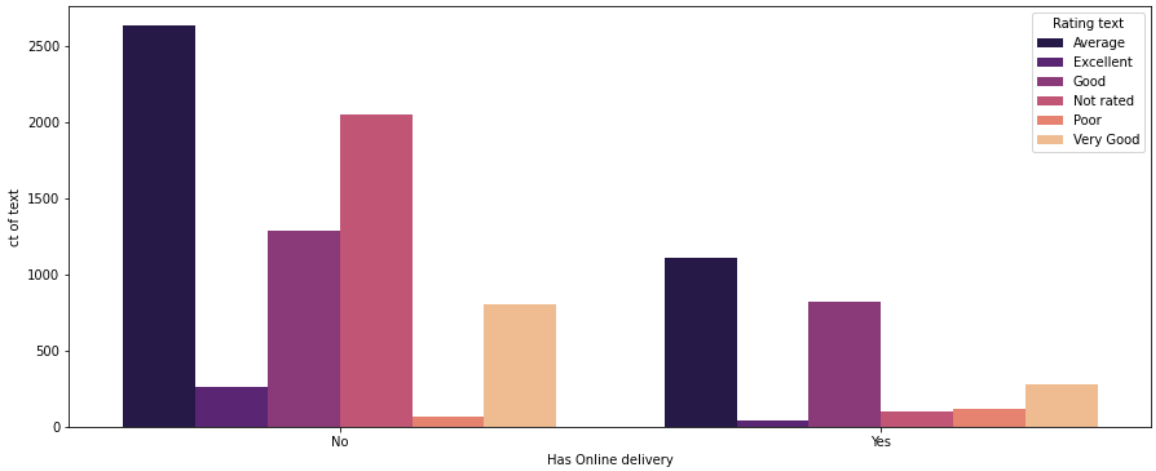
In [91]:

```
sns.barplot("Has Online delivery","ct of text",hue='Rating text',data=delivery,palette='magma')
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: FutureWarning: Pass the following variables as keyword args: x, y. From version 0.12, the only valid positional argument will be `data`, and passing other arguments without an explicit keyword will result in an error or misinterpretation.
  FutureWarning
```

Out[91]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f083420c810>
```



In [92]:

```
#Restaurants which didnt have only deliver has bad /poor ratings
```

In [93]:

```
table=df.groupby(["Has Table booking","Rating text"]).agg({"Rating text" :"coun
t"}).rename({"Rating text":"ct of text"},axis='columns').reset_index()
table
```

Out[93]:

	Has Table booking	Rating text	ct of text
0	No	Average	3343
1	No	Excellent	256
2	No	Good	1694
3	No	Not rated	2101
4	No	Poor	162
5	No	Very Good	837
6	Yes	Average	394
7	Yes	Excellent	45
8	Yes	Good	406
9	Yes	Not rated	47
10	Yes	Poor	24
11	Yes	Very Good	242

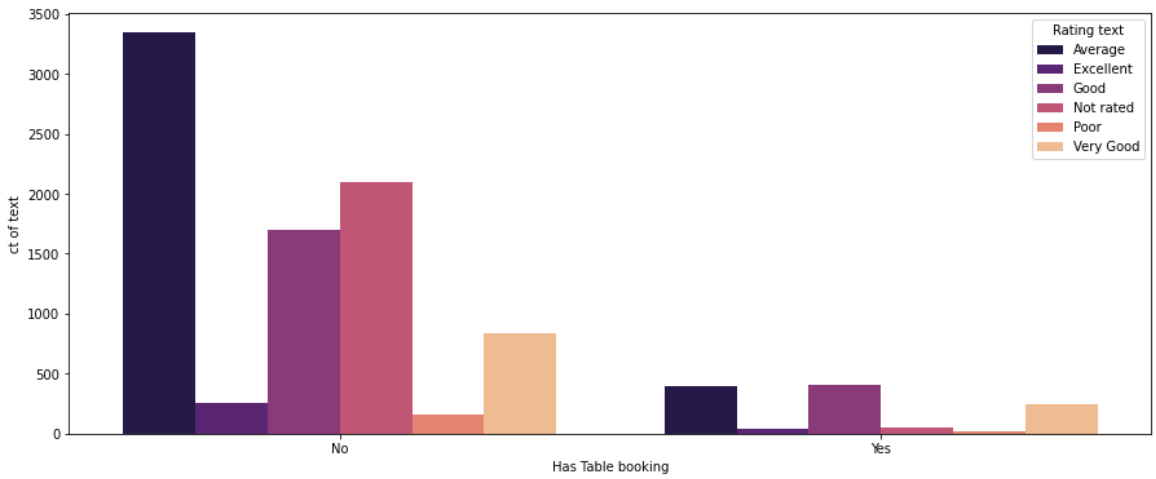
In [94]:

```
sns.barplot("Has Table booking","ct of text",hue='Rating text',data=table,palette='magma')
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: FutureWarning: Pass the following variables as keyword args: x, y. From version 0.12, the only valid positional argument will be `data`, and passing other arguments without an explicit keyword will result in an error or misinterpretation.
  FutureWarning
```

Out[94]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f0834122d90>
```



In [95]:

```
#Conclusion:Since percentage of restaurants which do not have table booking option hence ratings are also biased but only within those also average rating is quite high for those
```

In [96]:

```
df.to_csv('restaurant.csv')
```

In [106]:

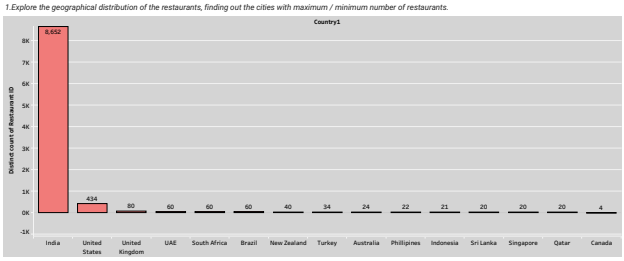
```
!jupyter nbconvert --to html /content/Restaurant_Recommendation_.ipynb
```

```
[NbConvertApp] Converting notebook /content/Restaurant_Recommendatio
n_.ipynb to html
[NbConvertApp] Writing 967889 bytes to /content/Restaurant_Recommend
ation_.html
```

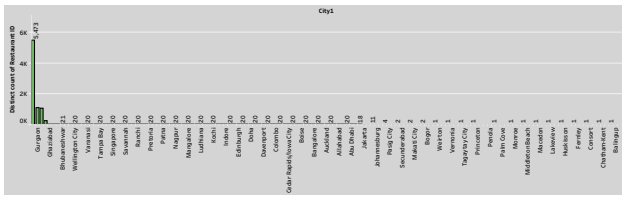
In [ ]:

```

```

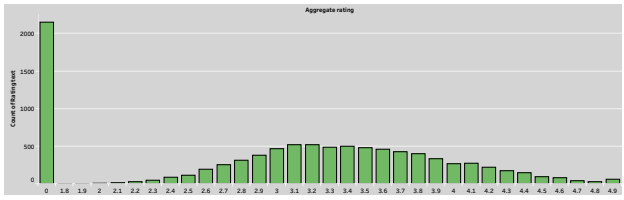
1.Explore the geographical distribution of the restaurants, finding out the cities with maximum / minimum number of restaurants.



No. of restaurant across cities

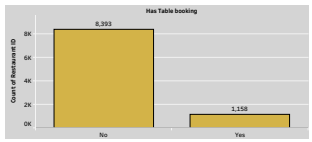


Rating distribution

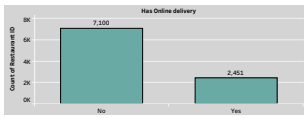


The franchise with most national presence

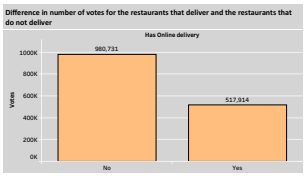
ratio between restaurants that allow table booking vs. those that do not allow table booking



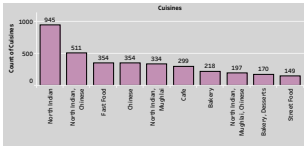
Find out the percentage of restaurants providing online delivery



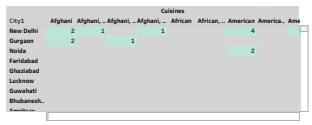
Difference in number of votes for the restaurants that deliver and the restaurants that do not deliver



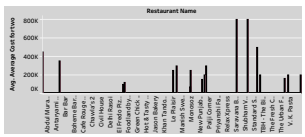
Top 10 cuisines across all the cities



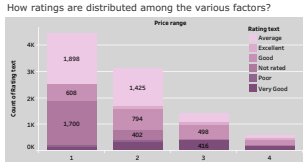
most served cuisine across the restaurant for each city



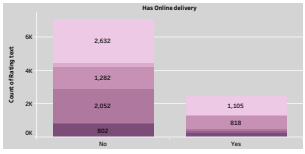
distribution cost across the restaurants



How ratings are distributed among the various factors?



factors in the data that may have an effect on ratings



Factors in the data that may have an effect on ratings