

zomatosales

August 10, 2024

1 ZOMATO SALES ANALYSIS

```
[1]: import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
plt.style.use('dark_background')
```

2 Reading CSV

```
[2]: df=pd.read_csv(r'C:\Users\Windows\Desktop\projects\zomato.csv')
df.head()
```

```
[2]:
```

	url	address	name
0	https://www.zomato.com/bangalore/jalsa-banasha...	942, 21st Main Road, 2nd Stage, Banashankari, ...	Jalsa
1	https://www.zomato.com/bangalore/spice-elephan...	2nd Floor, 80 Feet Road, Near Big Bazaar, 6th ...	Spice Elephant
2	https://www.zomato.com/SanchurroBangalore?cont...	1112, Next to KIMS Medical College, 17th Cross...	San Churro Cafe
3	https://www.zomato.com/bangalore/addhuri-udupi...	1st Floor, Annakuteera, 3rd Stage, Banashankar...	Addhuri Udupi Bhojana
4	https://www.zomato.com/bangalore/grand-village...	10, 3rd Floor, Lakshmi Associates, Gandhi Baza...	Grand Village

	online_order	book_table	rate	votes	phone
0	Yes	Yes	4.1/5	775	080 42297555\r\n+91 9743772233
1	Yes	No	4.1/5	787	080 41714161
2	Yes	No	3.8/5	918	+91 9663487993
3	No	No	3.7/5	88	+91 9620009302
4	No	No	3.8/5	166	+91 8026612447\r\n+91 9901210005

	location	rest_type
0	Banashankari	Casual Dining

```

1 Banashankari          Casual Dining
2 Banashankari Cafe, Casual Dining
3 Banashankari          Quick Bites
4 Basavanagudi          Casual Dining

```

```

                                dish_liked \
0 Pasta, Lunch Buffet, Masala Papad, Paneer Laja...
1 Momos, Lunch Buffet, Chocolate Nirvana, Thai G...
2 Churros, Cannelloni, Minestrone Soup, Hot Choc...
3                                     Masala Dosa
4                               Panipuri, Gol Gappe

```

```

                                cuisines approx_cost(for two people) \
0 North Indian, Mughlai, Chinese                        800
1      Chinese, North Indian, Thai                      800
2      Cafe, Mexican, Italian                          800
3      South Indian, North Indian                      300
4      North Indian, Rajasthani                        600

```

```

                                reviews_list menu_item \
0 [('Rated 4.0', 'RATED\n A beautiful place to ...      []
1 [('Rated 4.0', 'RATED\n Had been here for din...      []
2 [('Rated 3.0', 'RATED\n Ambience is not that ...      []
3 [('Rated 4.0', 'RATED\n Great food and proper...      []
4 [('Rated 4.0', 'RATED\n Very good restaurant ...      []

```

```

listed_in(type) listed_in(city)
0      Buffet    Banashankari
1      Buffet    Banashankari
2      Buffet    Banashankari
3      Buffet    Banashankari
4      Buffet    Banashankari

```

```
[3]: df.shape
```

```
[3]: (51717, 17)
```

```
[4]: df.columns
```

```
[4]: Index(['url', 'address', 'name', 'online_order', 'book_table', 'rate', 'votes',
          'phone', 'location', 'rest_type', 'dish_liked', 'cuisines',
          'approx_cost(for two people)', 'reviews_list', 'menu_item',
          'listed_in(type)', 'listed_in(city)'],
          dtype='object')
```

```
[5]: df=df.
      ↪drop(['url', 'address', 'phone', 'menu_item', 'dish_liked', 'reviews_list'], axis=1)
```

```
df.head()
```

```
[5]:
```

	name	online_order	book_table	rate	votes	location \
0	Jalsa	Yes	Yes	4.1/5	775	Banashankari
1	Spice Elephant	Yes	No	4.1/5	787	Banashankari
2	San Churro Cafe	Yes	No	3.8/5	918	Banashankari
3	Addhuri Udupi Bhojana	No	No	3.7/5	88	Banashankari
4	Grand Village	No	No	3.8/5	166	Basavanagudi

	rest_type	cuisines \
0	Casual Dining	North Indian, Mughlai, Chinese
1	Casual Dining	Chinese, North Indian, Thai
2	Cafe, Casual Dining	Cafe, Mexican, Italian
3	Quick Bites	South Indian, North Indian
4	Casual Dining	North Indian, Rajasthani

	approx_cost(for two people)	listed_in(type)	listed_in(city)
0	800	Buffet	Banashankari
1	800	Buffet	Banashankari
2	800	Buffet	Banashankari
3	300	Buffet	Banashankari
4	600	Buffet	Banashankari

```
[6]: df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 51717 entries, 0 to 51716
Data columns (total 11 columns):
#   Column                                Non-Null Count  Dtype
---  -
0   name                                  51717 non-null  object
1   online_order                         51717 non-null  object
2   book_table                           51717 non-null  object
3   rate                                 43942 non-null  object
4   votes                                51717 non-null  int64
5   location                             51696 non-null  object
6   rest_type                            51490 non-null  object
7   cuisines                             51672 non-null  object
8   approx_cost(for two people)          51371 non-null  object
9   listed_in(type)                      51717 non-null  object
10  listed_in(city)                      51717 non-null  object
dtypes: int64(1), object(10)
memory usage: 4.3+ MB
```

3 Dropping Duplicates

```
[7]: df.drop_duplicates(inplace=True)
df.shape
```

```
[7]: (51609, 11)
```

4 Cleaning Rate column

```
[8]: df['rate'].unique()
```

```
[8]: array(['4.1/5', '3.8/5', '3.7/5', '3.6/5', '4.6/5', '4.0/5', '4.2/5',
        '3.9/5', '3.1/5', '3.0/5', '3.2/5', '3.3/5', '2.8/5', '4.4/5',
        '4.3/5', 'NEW', '2.9/5', '3.5/5', nan, '2.6/5', '3.8 /5', '3.4/5',
        '4.5/5', '2.5/5', '2.7/5', '4.7/5', '2.4/5', '2.2/5', '2.3/5',
        '3.4 /5', '-', '3.6 /5', '4.8/5', '3.9 /5', '4.2 /5', '4.0 /5',
        '4.1 /5', '3.7 /5', '3.1 /5', '2.9 /5', '3.3 /5', '2.8 /5',
        '3.5 /5', '2.7 /5', '2.5 /5', '3.2 /5', '2.6 /5', '4.5 /5',
        '4.3 /5', '4.4 /5', '4.9/5', '2.1/5', '2.0/5', '1.8/5', '4.6 /5',
        '4.9 /5', '3.0 /5', '4.8 /5', '2.3 /5', '4.7 /5', '2.4 /5',
        '2.1 /5', '2.2 /5', '2.0 /5', '1.8 /5'], dtype=object)
```

5 Removing “NEW”, “-” and “/5” from Rate Column

```
[9]: def handlerate(value):
    if(value=='NEW' or value=='-'):
        return np.nan
    else:
        value=str(value).split('/')
        value=value[0]
        return float(value)
df['rate']=df['rate'].apply(handlerate)
df['rate'].head()
```

```
[9]: 0    4.1
1    4.1
2    3.8
3    3.7
4    3.8
Name: rate, dtype: float64
```

6 Filling Null Values in Rate Column with Mean

```
[10]: df['rate'] = df['rate'].fillna(df['rate'].mean())
      df['rate'].isnull().sum()
```

```
[10]: np.int64(0)
```

```
[11]: df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 51609 entries, 0 to 51716
Data columns (total 11 columns):
 #   Column                                Non-Null Count  Dtype
---  -
 0   name                                51609 non-null  object
 1   online_order                        51609 non-null  object
 2   book_table                          51609 non-null  object
 3   rate                                51609 non-null  float64
 4   votes                              51609 non-null  int64
 5   location                            51588 non-null  object
 6   rest_type                          51382 non-null  object
 7   cuisines                           51564 non-null  object
 8   approx_cost(for two people)         51265 non-null  object
 9   listed_in(type)                    51609 non-null  object
10   listed_in(city)                    51609 non-null  object
dtypes: float64(1), int64(1), object(9)
memory usage: 4.7+ MB
```

7 Dropping Null Values

```
[12]: df.dropna(inplace=True)
      df.head()
```

```
[12]:
```

	name	online_order	book_table	rate	votes	location	\
0	Jalsa	Yes	Yes	4.1	775	Banashankari	
1	Spice Elephant	Yes	No	4.1	787	Banashankari	
2	San Churro Cafe	Yes	No	3.8	918	Banashankari	
3	Addhuri Udupi Bhojana	No	No	3.7	88	Banashankari	
4	Grand Village	No	No	3.8	166	Basavanagudi	

	rest_type	cuisines	\
0	Casual Dining	North Indian, Mughlai, Chinese	
1	Casual Dining	Chinese, North Indian, Thai	
2	Cafe, Casual Dining	Cafe, Mexican, Italian	
3	Quick Bites	South Indian, North Indian	
4	Casual Dining	North Indian, Rajasthani	

```

    approx_cost(for two people) listed_in(type) listed_in(city)
0                800          Buffet Banashankari
1                800          Buffet Banashankari
2                800          Buffet Banashankari
3                300          Buffet Banashankari
4                600          Buffet Banashankari

```

```

[13]: df.rename(columns = {'approx_cost(for two people)': 'Cost2plates',
    ↪ 'listed_in(type)': 'Type'}, inplace = True)
df.head()

```

```

[13]:          name online_order book_table rate votes location \
0          Jalsa          Yes         Yes  4.1   775 Banashankari
1  Spice Elephant          Yes          No  4.1   787 Banashankari
2  San Churro Cafe          Yes          No  3.8   918 Banashankari
3 Addhuri Udupi Bhojana          No          No  3.7    88 Banashankari
4   Grand Village          No          No  3.8   166 Basavanagudi

```

```

          rest_type          cuisines Cost2plates Type \
0  Casual Dining North Indian, Mughlai, Chinese      800 Buffet
1  Casual Dining   Chinese, North Indian, Thai      800 Buffet
2  Cafe, Casual Dining   Cafe, Mexican, Italian      800 Buffet
3   Quick Bites   South Indian, North Indian      300 Buffet
4  Casual Dining   North Indian, Rajasthani      600 Buffet

```

```

    listed_in(city)
0  Banashankari
1  Banashankari
2  Banashankari
3  Banashankari
4  Banashankari

```

```

[14]: df['location'].unique()

```

```

[14]: array(['Banashankari', 'Basavanagudi', 'Mysore Road', 'Jayanagar',
    'Kumaraswamy Layout', 'Rajarajeshwari Nagar', 'Vijay Nagar',
    'Uttarahalli', 'JP Nagar', 'South Bangalore', 'City Market',
    'Nagarbhavi', 'Bannerghatta Road', 'BTM', 'Kanakapura Road',
    'Bommanahalli', 'CV Raman Nagar', 'Electronic City', 'HSR',
    'Marathahalli', 'Wilson Garden', 'Shanti Nagar',
    'Koramangala 5th Block', 'Koramangala 8th Block', 'Richmond Road',
    'Koramangala 7th Block', 'Jalahalli', 'Koramangala 4th Block',
    'Bellandur', 'Sarjapur Road', 'Whitefield', 'East Bangalore',
    'Old Airport Road', 'Indiranagar', 'Koramangala 1st Block',
    'Frazer Town', 'RT Nagar', 'MG Road', 'Brigade Road',
    'Lavelle Road', 'Church Street', 'Ulsoor', 'Residency Road',

```

```
'Shivajinagar', 'Infantry Road', 'St. Marks Road',
'Cunningham Road', 'Race Course Road', 'Commercial Street',
'Vasanth Nagar', 'HBR Layout', 'Domlur', 'Ejipura',
'Jeevan Bhima Nagar', 'Old Madras Road', 'Malleshwaram',
'Seshadripuram', 'Kammanahalli', 'Koramangala 6th Block',
'Majestic', 'Langford Town', 'Central Bangalore', 'Sanjay Nagar',
'Brookefield', 'ITPL Main Road, Whitefield',
'Varthur Main Road, Whitefield', 'KR Puram',
'Koramangala 2nd Block', 'Koramangala 3rd Block', 'Koramangala',
'Hosur Road', 'Rajajinagar', 'Banaswadi', 'North Bangalore',
'Nagawara', 'Hennur', 'Kalyan Nagar', 'New BEL Road', 'Jakkur',
'Rammurthy Nagar', 'Thippasandra', 'Kaggadasapura', 'Hebbal',
'Kengeri', 'Sankey Road', 'Sadashiv Nagar', 'Basaveshwara Nagar',
'Yeshwantpur', 'West Bangalore', 'Magadi Road', 'Yelahanka',
'Sahakara Nagar', 'Peenya'], dtype=object)
```

```
[15]: df['listed_in(city)'].unique()
```

```
[15]: array(['Banashankari', 'Bannerghatta Road', 'Basavanagudi', 'Bellandur',
'Brigade Road', 'Brookefield', 'BTM', 'Church Street',
'Electronic City', 'Frazer Town', 'HSR', 'Indiranagar',
'Jayanagar', 'JP Nagar', 'Kalyan Nagar', 'Kammanahalli',
'Koramangala 4th Block', 'Koramangala 5th Block',
'Koramangala 6th Block', 'Koramangala 7th Block', 'Lavelle Road',
'Malleshwaram', 'Marathahalli', 'MG Road', 'New BEL Road',
'Old Airport Road', 'Rajajinagar', 'Residency Road',
'Sarjapur Road', 'Whitefield'], dtype=object)
```

8 Listed in(city) and location, both are there, lets keep only one

```
[16]: df.drop(['listed_in(city)'],axis=1,inplace=True)
```

```
[17]: df['Cost2plates'].unique()
```

```
[17]: array(['800', '300', '600', '700', '550', '500', '450', '650', '400',
'900', '200', '750', '150', '850', '100', '1,200', '350', '250',
'950', '1,000', '1,500', '1,300', '199', '80', '1,100', '160',
'1,600', '230', '130', '50', '190', '1,700', '1,400', '180',
'1,350', '2,200', '2,000', '1,800', '1,900', '330', '2,500',
'2,100', '3,000', '2,800', '3,400', '40', '1,250', '3,500',
'4,000', '2,400', '2,600', '120', '1,450', '469', '70', '3,200',
'60', '560', '240', '360', '6,000', '1,050', '2,300', '4,100',
'5,000', '3,700', '1,650', '2,700', '4,500', '140'], dtype=object)
```

```
[18]: def handlecost(value):
value=str(value)
```

```

    if ',' in value:
        value=value.replace(',','')
        return float(value)
    else:
        return float(value)
df['Cost2plates']=df['Cost2plates'].apply(handlecost)
df['Cost2plates'].unique()

```

```

[18]: array([ 800., 300., 600., 700., 550., 500., 450., 650., 400.,
           900., 200., 750., 150., 850., 100., 1200., 350., 250.,
           950., 1000., 1500., 1300., 199., 80., 1100., 160., 1600.,
           230., 130., 50., 190., 1700., 1400., 180., 1350., 2200.,
          2000., 1800., 1900., 330., 2500., 2100., 3000., 2800., 3400.,
           40., 1250., 3500., 4000., 2400., 2600., 120., 1450., 469.,
           70., 3200., 60., 560., 240., 360., 6000., 1050., 2300.,
          4100., 5000., 3700., 1650., 2700., 4500., 140.])

```

```

[19]: df['Cost2plates'].dtype

```

```

[19]: dtype('float64')

```

9 Cleaning Rest Type Column

```

[20]: rest_types = df['rest_type'].value_counts(ascending =False)
rest_types

```

```

[20]: rest_type
Quick Bites                19010
Casual Dining              10253
Cafe                       3682
Delivery                   2574
Dessert Parlor             2242
...
Dessert Parlor, Kiosk        2
Dessert Parlor, Food Court   2
Food Court, Beverage Shop    2
Sweet Shop, Dessert Parlor    1
Quick Bites, Kiosk           1
Name: count, Length: 93, dtype: int64

```

```

[21]: rest_typelessthan1000=rest_types[rest_types<1000]
rest_typelessthan1000

```

```

[21]: rest_type
Beverage Shop              863
Bar                        686

```


Food Court	616
Sweet Shop	468
Bar, Casual Dining	411
...	
Dessert Parlor, Kiosk	2
Dessert Parlor, Food Court	2
Food Court, Beverage Shop	2
Sweet Shop, Dessert Parlor	1
Quick Bites, Kiosk	1

Name: count, Length: 85, dtype: int64

10 Making Rest Types less than 1000 in frequency as others

```
[22]: def handleresttype(value):
        if(value in rest_typelessthan1000):
            return 'others'
        else:
            return value
df['rest_type'] = df['rest_type'].apply(handleresttype)
df['rest_type'].value_counts()
```

```
[22]: rest_type
Quick Bites          19010
Casual Dining        10253
others               9003
Cafe                 3682
Delivery             2574
Dessert Parlor       2242
Takeaway, Delivery   2008
Bakery               1140
Casual Dining, Bar    1130
Name: count, dtype: int64
```

11 Cleaning Location Column

```
[23]: location= df['location'].value_counts(ascending=False)
locationlessthan300=location[location<300]

def handle_location(value):
    if(value in locationlessthan300):
        return "others"
    else:
        return value
df['location']=df['location'].apply(handle_location)
df['location'].value_counts()
```

```

[23]: location
      BTM                    5056
      others                 4954
      HSR                    2494
      Koramangala 5th Block  2479
      JP Nagar               2218
      Whitefield             2105
      Indiranagar            2026
      Jayanagar              1916
      Marathahalli           1805
      Bannerghatta Road      1609
      Bellandur              1268
      Electronic City        1246
      Koramangala 1st Block  1236
      Brigade Road           1210
      Koramangala 7th Block  1174
      Koramangala 6th Block  1127
      Sarjapur Road          1047
      Koramangala 4th Block  1017
      Ulsoor                 1011
      Banashankari           902
      MG Road                 893
      Kalyan Nagar           841
      Richmond Road          803
      Malleshwaram           721
      Frazer Town             714
      Basavanagudi           684
      Residency Road         671
      Brookefield            656
      New BEL Road           644
      Banaswadi              640
      Kammanahalli           639
      Rajajinagar            591
      Church Street          566
      Lavelle Road           518
      Shanti Nagar           508
      Shivajinagar           498
      Cunningham Road        490
      Domlur                  482
      Old Airport Road        437
      Ejipura                 433
      Commercial Street       370
      St. Marks Road          343
      Name: count, dtype: int64

```

12 Cleaning Cuisines Column

```
[24]: cuisines=df['cuisines'].value_counts(ascending =False)
      cuisineslessthan100=cuisines[cuisines<100]

      def handle_cuisines(value):
          if(value in cuisineslessthan100):
              return 'others'
          else:
              return value
      df['cuisines']=df['cuisines'].apply(handle_cuisines)
      df['cuisines'].value_counts( ascending=False)
```

```
[24]: cuisines
      others                26159
      North Indian           2852
      North Indian, Chinese   2351
      South Indian           1820
      Biryani                 903
      ...
      South Indian, Chinese, North Indian  105
      North Indian, Mughlai, Chinese       104
      South Indian, Fast Food              104
      Italian, Pizza                      102
      North Indian, Chinese, Seafood       102
      Name: count, Length: 70, dtype: int64
```

```
[25]: df.head()
```

```
[25]:
```

	name	online_order	book_table	rate	votes	location \
0	Jalsa	Yes	Yes	4.1	775	Banashankari
1	Spice Elephant	Yes	No	4.1	787	Banashankari
2	San Churro Cafe	Yes	No	3.8	918	Banashankari
3	Addhuri Udupi Bhojana	No	No	3.7	88	Banashankari
4	Grand Village	No	No	3.8	166	Basavanagudi

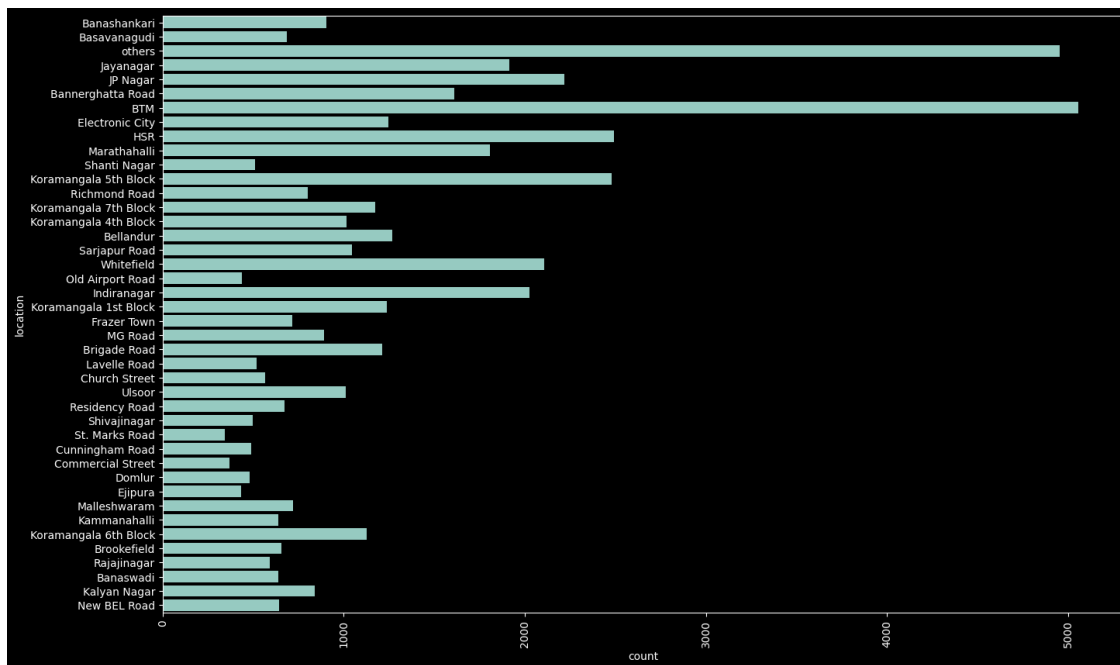
	rest_type	cuisines	Cost2plates	Type
0	Casual Dining	North Indian, Mughlai, Chinese	800.0	Buffet
1	Casual Dining	others	800.0	Buffet
2	others	others	800.0	Buffet
3	Quick Bites	South Indian, North Indian	300.0	Buffet
4	Casual Dining	others	600.0	Buffet

13 Data is Clean, Lets jump to Visualization

14 Count Plot of Various Locations

```
[26]: plt.figure(figsize = (16,10))
      ax = sns.countplot(df['location'])
      plt.xticks(rotation=90)
```

```
[26]: (array([ 0., 1000., 2000., 3000., 4000., 5000., 6000.]),
      [Text(0.0, 0, '0'),
       Text(1000.0, 0, '1000'),
       Text(2000.0, 0, '2000'),
       Text(3000.0, 0, '3000'),
       Text(4000.0, 0, '4000'),
       Text(5000.0, 0, '5000'),
       Text(6000.0, 0, '6000')])
```



15 Visualizing Online Order

```
[27]: plt.figure(figsize=(6,6))
      sns.countplot(x=df['online_order'], palette='inferno')
      plt.xlabel('Online Order')
      plt.ylabel('Count')
      plt.title('Count of Online Orders')
      plt.show()
```

```
C:\Users\Windows\AppData\Local\Temp\ipykernel_10424\1264392201.py:2:  
FutureWarning:
```

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `legend=False` for the same effect.

```
sns.countplot(x=df['online_order'], palette='inferno')
```



16 Visualizing Book Table

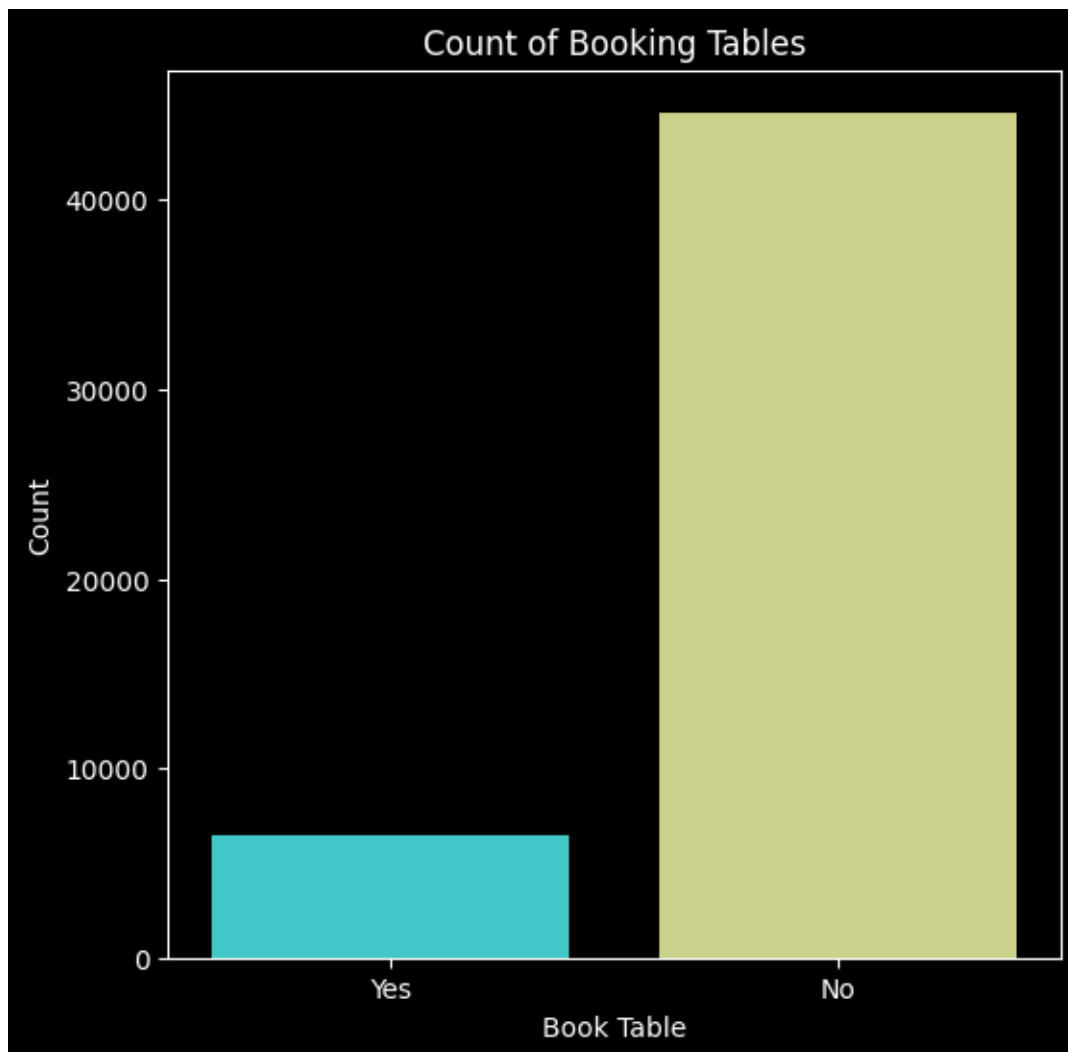
```
[28]: plt.figure(figsize = (6,6))
sns.countplot(x=df['book_table'], palette = 'rainbow')
plt.xlabel('Book Table')
plt.ylabel('Count')
plt.title('Count of Booking Tables')
plt.show()
```

C:\Users\Windows\AppData\Local\Temp\ipykernel_10424\2907597021.py:2:

FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `legend=False` for the same effect.

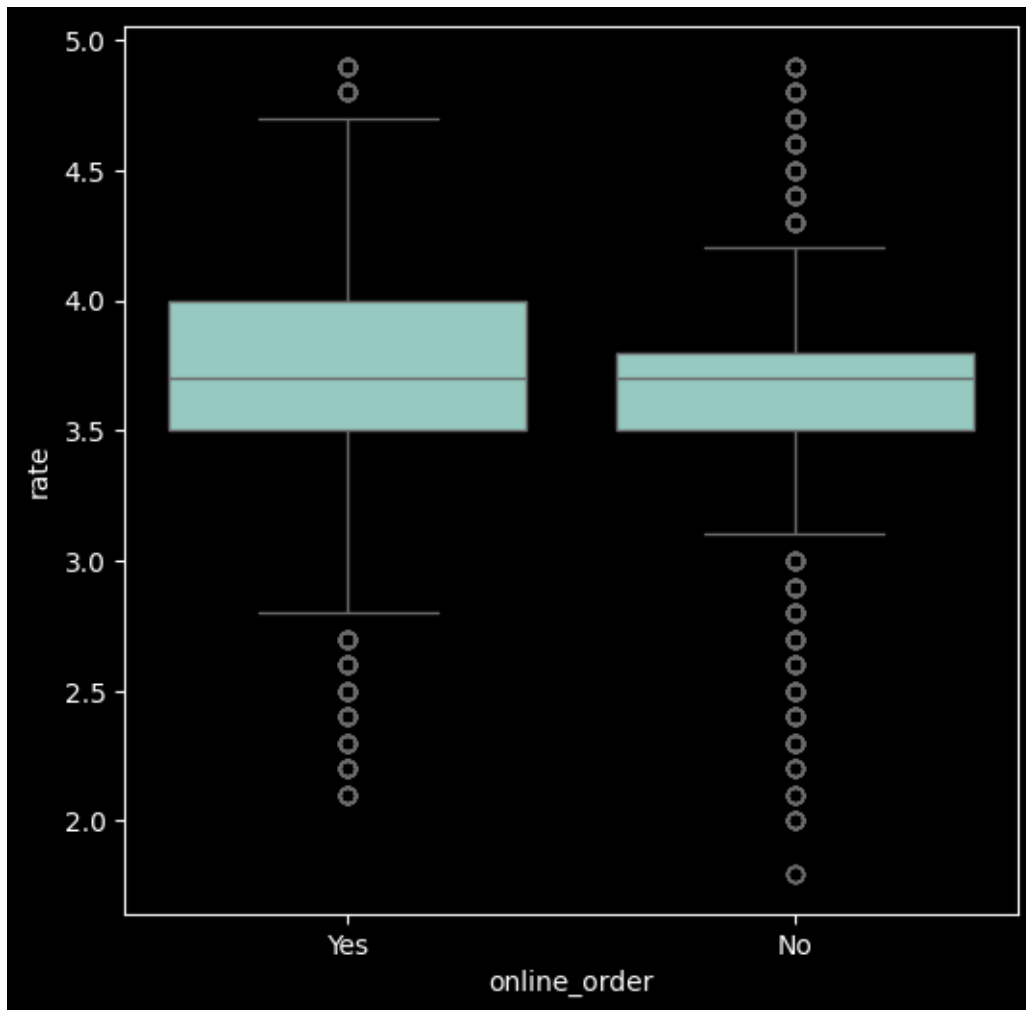
```
sns.countplot(x=df['book_table'], palette = 'rainbow')
```



17 Visualizing Online Order vs Rate

```
[29]: plt.figure(figsize = (6,6))  
      sns.boxplot(x = 'online_order', y = 'rate', data = df)
```

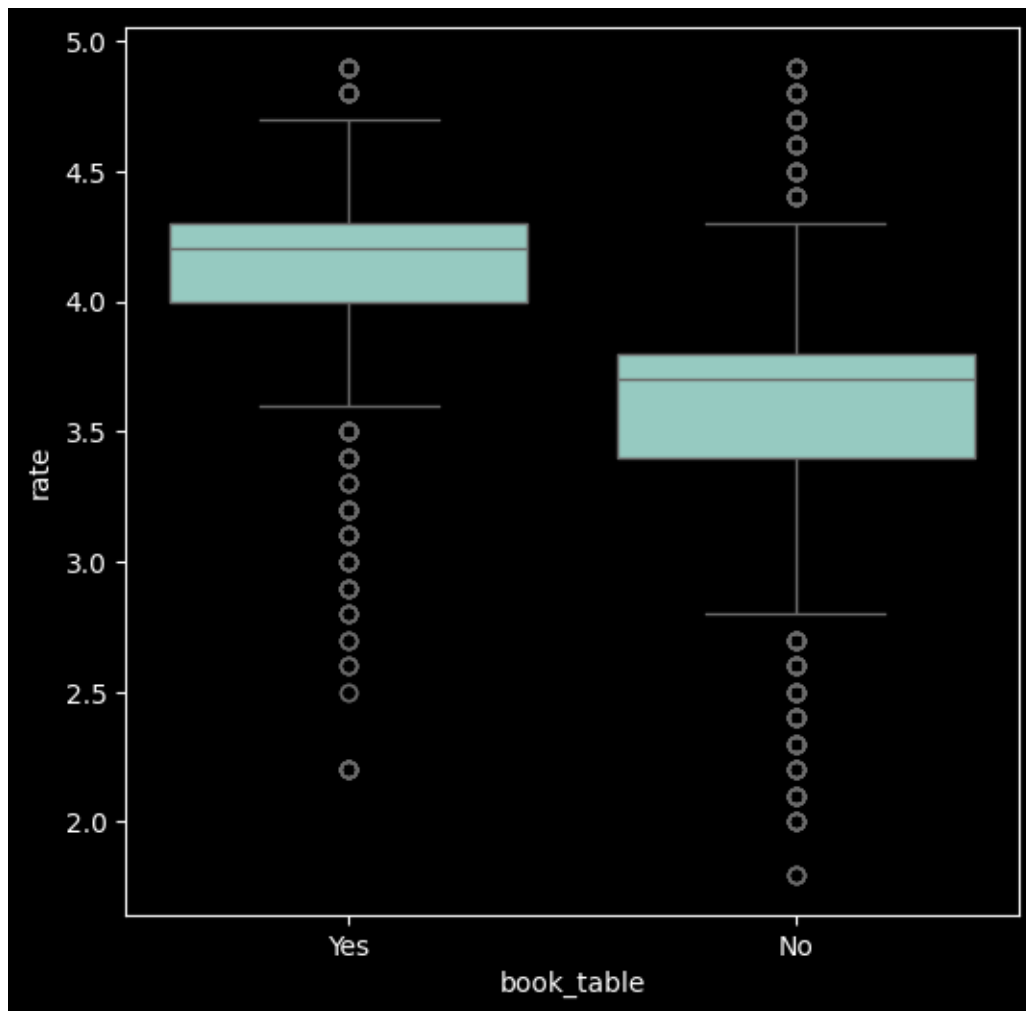
```
[29]: <Axes: xlabel='online_order', ylabel='rate'>
```



18 Visualizing Book Table vs Rate

```
[30]: plt.figure(figsize=(6,6))
sns.boxplot(x='book_table',y='rate',data =df)
```

```
[30]: <Axes: xlabel='book_table', ylabel='rate'>
```



19 Visualizing Online Order Facility, Location Wise

```
[31]: df1 = df.groupby(['location','online_order'])['name'].count()
df1.to_csv('location_online.csv')
df1 = pd.read_csv('location_online.csv')
df1 = pd.pivot_table(df1, values=None, index=['location'],
    columns=['online_order'], fill_value=0, aggfunc=np.sum)
df1
```


C:\Users\Windows\AppData\Local\Temp\ipykernel_10424\2546502282.py:4:
FutureWarning: The provided callable <function sum at 0x0000023DC1C60EA0> is
currently using DataFrameGroupBy.sum. In a future version of pandas, the
provided callable will be used directly. To keep current behavior pass the
string "sum" instead.

```
df1 = pd.pivot_table(df1, values=None, index=['location'],
columns=['online_order'], fill_value=0, aggfunc=np.sum)
```

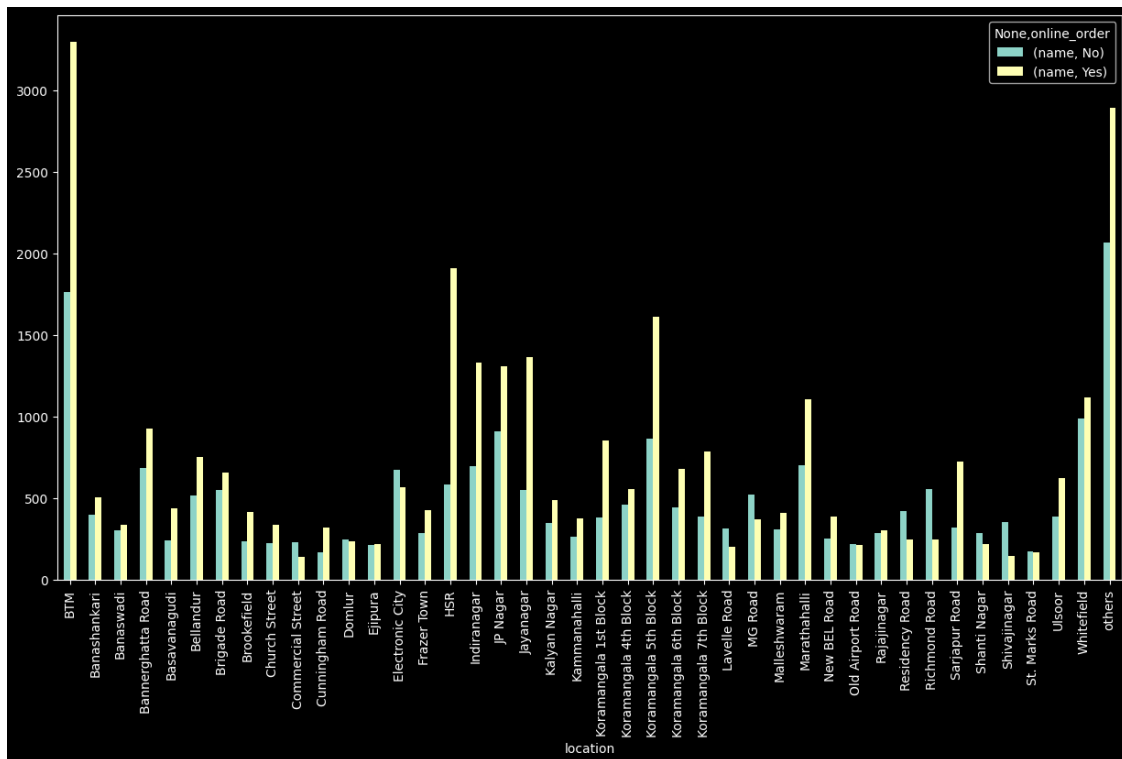
```
[31]:
```

	name	
online_order	No	Yes
location		
BTM	1763	3293
Banashankari	397	505
Banaswadi	302	338
Bannerghatta Road	685	924
Basavanagudi	243	441
Bellandur	517	751
Brigade Road	552	658
Brookefield	239	417
Church Street	226	340
Commercial Street	228	142
Cunningham Road	168	322
Domlur	247	235
Ejipura	214	219
Electronic City	676	570
Frazer Town	287	427
HSR	584	1910
Indiranagar	697	1329
JP Nagar	911	1307
Jayanagar	552	1364
Kalyan Nagar	350	491
Kammanahalli	264	375
Koramangala 1st Block	384	852
Koramangala 4th Block	459	558
Koramangala 5th Block	866	1613
Koramangala 6th Block	445	682
Koramangala 7th Block	389	785
Lavelle Road	315	203
MG Road	520	373
Malleshwaram	309	412
Marathahalli	701	1104
New BEL Road	255	389
Old Airport Road	221	216
Rajajinagar	286	305
Residency Road	424	247
Richmond Road	557	246
Sarjapur Road	323	724

Shanti Nagar	289	219
Shivajinagar	354	144
St. Marks Road	176	167
Ulsoor	389	622
Whitefield	986	1119
others	2064	2890

```
[32]: df1.plot(kind = 'bar', figsize = (15,8))
```

```
[32]: <Axes: xlabel='location'>
```



20 Visualizing Types of Restaurants vs Rate

```
[33]: plt.figure(figsize = (14, 8))
sns.boxplot(x = 'Type', y = 'rate', data = df, palette = 'inferno')
```

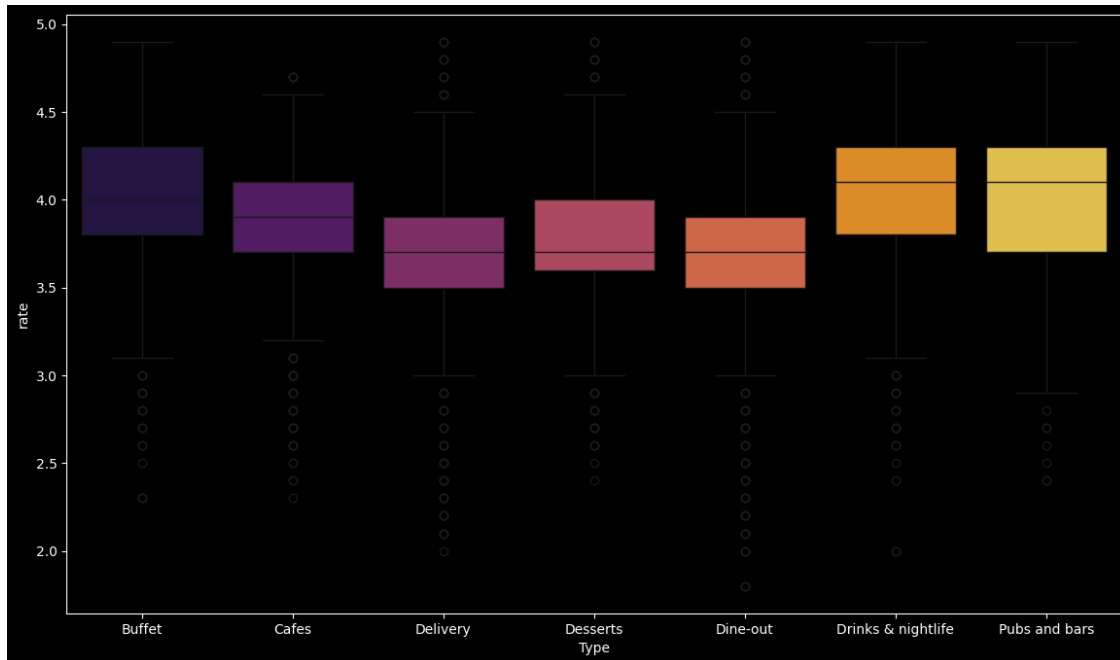
C:\Users\Windows\AppData\Local\Temp\ipykernel_10424\2234948669.py:2:

FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `legend=False` for the same effect.

```
sns.boxplot(x = 'Type', y = 'rate', data = df, palette = 'inferno')
```

```
[33]: <Axes: xlabel='Type', ylabel='rate'>
```



21 Grouping Types of Restaurants, location wise

```
[34]: df3 = df.groupby(['location', 'Type'])['name'].count()
df3.to_csv('location_Type.csv')
df3 = pd.read_csv('location_Type.csv')
df3 = pd.pivot_table(df3, values=None, index=['location'], columns=['Type'],
    ↪ fill_value=0, aggfunc=np.sum)
df3
```

```
C:\Users\Windows\AppData\Local\Temp\ipykernel_10424\1140243432.py:4:
FutureWarning: The provided callable <function sum at 0x0000023DC1C60EA0> is
currently using DataFrameGroupBy.sum. In a future version of pandas, the
provided callable will be used directly. To keep current behavior pass the
string "sum" instead.
df3 = pd.pivot_table(df3, values=None, index=['location'], columns=['Type'],
fill_value=0, aggfunc=np.sum)
```

```
[34]:
```

	name	
Type	Buffet Cafes Delivery Desserts Dine-out	
location		

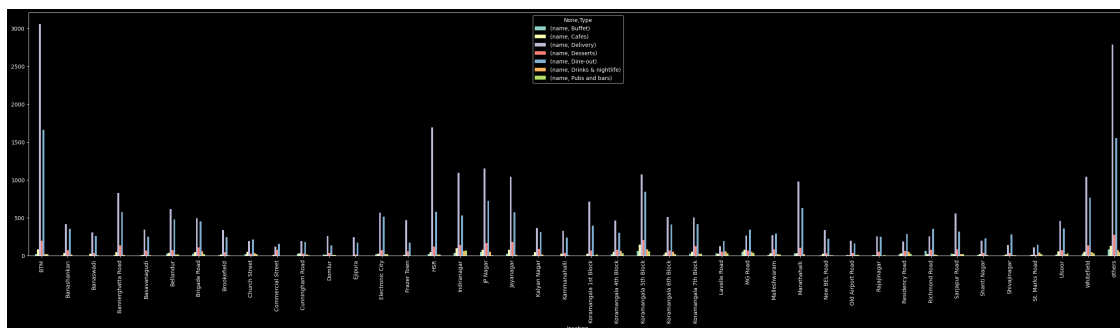
BTM	21	83	3053	198	1660
Banashankari	7	36	418	71	356
Banaswadi	0	24	310	37	262
Bannerghatta Road	9	46	828	137	578
Basavanagudi	7	11	344	66	251
Bellandur	28	36	617	75	479
Brigade Road	25	46	497	108	455
Brookefield	6	17	339	45	245
Church Street	19	51	193	29	215
Commercial Street	0	13	121	77	159
Cunningham Road	29	34	194	26	184
Domlur	15	13	261	35	135
Ejipura	0	0	245	16	172
Electronic City	23	24	570	71	516
Frazer Town	1	11	470	56	172
HSR	19	49	1694	120	580
Indiranagar	38	97	1091	140	529
JP Nagar	45	76	1151	166	722
Jayanagar	27	77	1043	182	575
Kalyan Nagar	9	45	366	88	315
Kammanahalli	2	27	329	35	240
Koramangala 1st Block	3	26	716	70	398
Koramangala 4th Block	21	53	464	81	302
Koramangala 5th Block	65	146	1075	209	842
Koramangala 6th Block	18	43	511	70	411
Koramangala 7th Block	25	52	503	127	417
Lavelle Road	30	27	127	50	191
MG Road	51	76	266	68	343
Malleshwaram	11	31	269	85	291
Marathahalli	34	32	980	105	630
New BEL Road	4	29	338	33	224
Old Airport Road	12	5	200	35	164
Rajajinagar	10	4	258	55	251
Residency Road	20	31	187	63	289
Richmond Road	63	21	257	78	356
Sarjapur Road	25	22	558	82	319
Shanti Nagar	9	22	198	39	229
Shivajinagar	6	17	143	37	280
St. Marks Road	5	10	111	10	145
Ulsoor	16	56	456	71	359
Whitefield	28	51	1041	137	768
others	83	133	2787	276	1553

Type	Drinks & nightlife	Pubs and bars
location		
BTM	22	19

Banashankari	14	0
Banaswadi	6	1
Bannerghatta Road	9	2
Basavanagudi	5	0
Bellandur	17	16
Brigade Road	57	22
Brookefield	4	0
Church Street	36	23
Commercial Street	0	0
Cunningham Road	16	7
Domlur	12	11
Ejipura	0	0
Electronic City	21	21
Frazer Town	2	2
HSR	14	18
Indiranagar	65	66
JP Nagar	51	7
Jayanagar	12	0
Kalyan Nagar	18	0
Kammanahalli	6	0
Koramangala 1st Block	7	16
Koramangala 4th Block	62	34
Koramangala 5th Block	84	58
Koramangala 6th Block	51	23
Koramangala 7th Block	25	25
Lavelle Road	59	34
MG Road	53	36
Malleshwaram	20	14
Marathahalli	22	2
New BEL Road	8	8
Old Airport Road	12	9
Rajajinagar	3	10
Residency Road	55	26
Richmond Road	16	12
Sarjapur Road	19	22
Shanti Nagar	9	2
Shivajinagar	7	8
St. Marks Road	40	22
Ulsoor	23	30
Whitefield	47	33
others	75	47

```
[35]: df3.plot(kind = 'bar', figsize = (36,8))
```

```
[35]: <Axes: xlabel='location'>
```



22 No. of Votes, Location Wise

```
[36]: df4 = df[['location', 'votes']]
df4.drop_duplicates()
df5 = df4.groupby(['location'])['votes'].sum()
df5 = df5.to_frame()
df5 = df5.sort_values('votes', ascending=False)
df5.head()
```

```
[36]:          votes
location
Koramangala 5th Block  2214083
Indiranagar          1165909
Koramangala 4th Block   685156
Church Street         590306
JP Nagar             586522
```

23 Visualizing Top Cuisines

```
[37]: df6 = df[['cuisines', 'votes']]
df6.drop_duplicates()
df7 = df6.groupby(['cuisines'])['votes'].sum()
df7 = df7.to_frame()
df7 = df7.sort_values('votes', ascending=False)
df7.head()
```

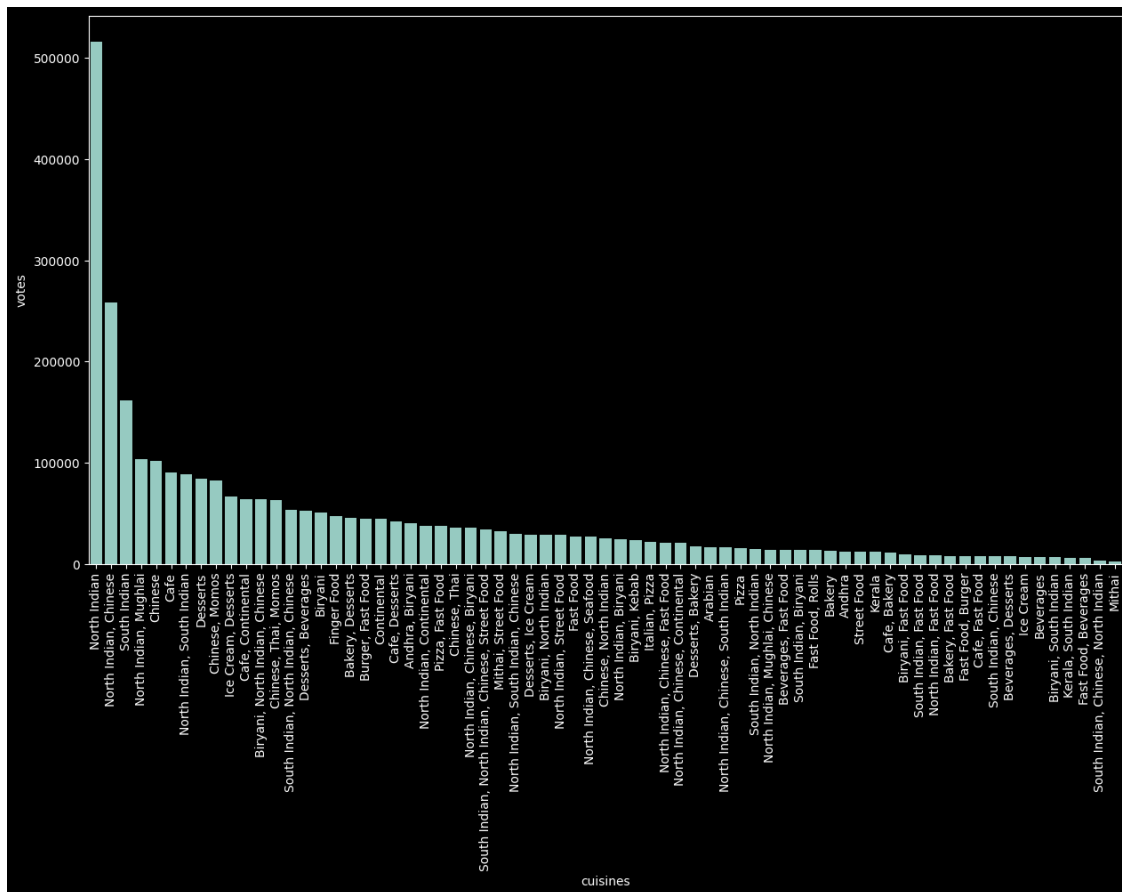
```
[37]:          votes
cuisines
others          11542182
North Indian     516310
North Indian, Chinese  258225
South Indian     161975
North Indian, Mughlai  103706
```

```
[38]: df7 = df7.iloc[1:, :]  
df7.head()
```

```
[38]:
```

cuisines	votes
North Indian	516310
North Indian, Chinese	258225
South Indian	161975
North Indian, Mughlai	103706
Chinese	101728

```
[39]: plt.figure(figsize=(15, 8))  
sns.barplot(x=df7.index, y=df7['votes'])  
plt.xticks(rotation=90)  
plt.show()
```



24 THANK YOU !!!!

[]: