DATA ANALYSIS LEVEL-1

step-1 : Import necessary python libraries

In [328…

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

step-2 : Load the dataset into dataframe

In [6]:

```
# read the csv file using pandas
restaurant_df=pd.read_csv(r"C:\Users\ubade\Downloads\Dataset .csv")
restaurant_df
```

Out[6]:

	Restaurant ID	Restaurant Name	Country Code	City	Address	Locality	Locality Verbose	Longitude	Latitude	Cuisines
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	French, Japanese, Desserts
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	Japanese
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	Seafood, Asian, Filipino, Indian
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	Japanese, Sushi
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	Japanese, Korean
...	...	...	...	...	...	...	...	...	...	...
9546	5915730	Naml\ Gurme	208	��stanbul	Kemanke�� Karamustafa Pa��a Mahallesi, R\ht\m ...	Karak�_y	Karak�_y, ��stanbul	28.977392	41.022793	Turkish
9547	5908749	Ceviz A��ac\	208	��stanbul	Ko��uyolu Mahallesi, Muhittin	Ko��uyolu	Ko��uyolu, ��stanbul	29.041297	41.009847	World Cuisine, Patisserie, Cafe

	Restaurant ID	Restaurant Name	Country Code	City	Address	Locality	Locality Verbose	Longitude	Latitude	Cuisines
					��st�_nda�� Cadd...					
9548	5915807	Huqqa	208	��stanbul	Kuru�_e��me Mahallesi, Muallim Naci Caddesi, N...	Kuru�_e��me	Kuru�_e��me, ��stanbul	29.034640	41.055817	Italian, World Cuisine
9549	5916112	A���k Kahve	208	��stanbul	Kuru�_e��me Mahallesi, Muallim Naci Caddesi, N...	Kuru�_e��me	Kuru�_e��me, ��stanbul	29.036019	41.057979	Restaurant Cafe
9550	5927402	Walter's Coffee Roastery	208	��stanbul	Cafea��a Mahallesi, Bademalt\ Sokak, No 21/B, ...	Moda	Moda, ��stanbul	29.026016	40.984776	Cafe

9551 rows × 21 columns

check allover data information

In [9]:
```
restaurant_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
```

checking for missing values

In [16]:
```
restaurant_df.isnull().sum()
```

Out[16]:
```
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
```

basic statistical summary

In [21]:
```
restaurant_df.describe()
```

Out[21]:

	Restaurant ID	Country Code	Longitude	Latitude	Average Cost for two	Price range	Aggregate rating	Votes
count	9.551000e+03	9551.000000	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	156.909748
std	8.791521e+06	56.750546	41.467058	11.007935	16121.183073	0.905609	1.516378	430.169145
min	5.300000e+01	1.000000	-157.948486	-41.330428	0.000000	1.000000	0.000000	0.000000
25%	3.019625e+05	1.000000	77.081343	28.478713	250.000000	1.000000	2.500000	5.000000
50%	6.004089e+06	1.000000	77.191964	28.570469	400.000000	2.000000	3.200000	31.000000
75%	1.835229e+07	1.000000	77.282006	28.642758	700.000000	2.000000	3.700000	131.000000
max	1.850065e+07	216.000000	174.832089	55.976980	800000.000000	4.000000	4.900000	10934.000000

checking unique values

In [26]:

```
restaurant_df.nunique()
```

Out[26]:
```
Restaurant ID           9551
Restaurant Name         7446
Country Code              15
City                     141
Address                 8918
Locality                1208
Locality Verbose        1265
Longitude               8120
Latitude                8677
Cuisines                1825
Average Cost for two     140
Currency                  12
Has Table booking          2
Has Online delivery        2
Is delivering now          2
Switch to order menu       1
Price range                4
Aggregate rating          33
Rating color               6
Rating text                6
Votes                   1012
dtype: int64
```

Task 1: Top Cuisines

Determine the top three most common cuisines in the dataset.

Calculate the percentage of restaurants that serve each of the top cuisines.

In [40]:
```
restaurant_df.columns
```

Out[40]:
```
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 [55]:
```
restaurant_df.head()
```

Out[55]:

	Restaurant ID	Restaurant Name	Country Code	City	Address	Locality	Locality Verbose	Longitude	Latitude	Cuisines	...	Currency
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	French, Japanese, Desserts	...	Botswana Pula(P)
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	Japanese	...	Botswana Pula(P)
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	Seafood, Asian, Filipino, Indian	...	Botswana Pula(P)
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	Japanese, Sushi	...	Botswana Pula(P)
4	6314302	Sambo Kojin	162	Mandaluyong City	Third Floor, Mega Atrium, SM	SM Megamall, Ortigas, Mandaluyong City	SM Megamall, Ortigas, Mandaluyong	121.057508	14.584450	Japanese, Korean	...	Botswana Pula(P)

Restaurant ID	Restaurant Name	Country Code	City	Address	Locality	Locality Verbose	Longitude	Latitude	Cuisines	...	Currency
				Megamall, Ortigas...		City, Mandal...					

5 rows × 21 columns

In [57]:
```python
restaurant_df['Cuisines']
```

Out[57]:
```
0               French, Japanese, Desserts
1                                 Japanese
2         Seafood, Asian, Filipino, Indian
3                          Japanese, Sushi
4                         Japanese, Korean
                      ...
9546                               Turkish
9547       World Cuisine, Patisserie, Cafe
9548                Italian, World Cuisine
9549                       Restaurant Cafe
9550                                  Cafe
Name: Cuisines, Length: 9551, dtype: object
```

In [59]:
```python
# count the occurrence of each cuisines : value_count()
restaurant_df['Cuisines'].value_counts().reset_index
```

Out[59]:
```
<bound method Series.reset_index of Cuisines
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: count, Length: 1825, dtype: int64>
```

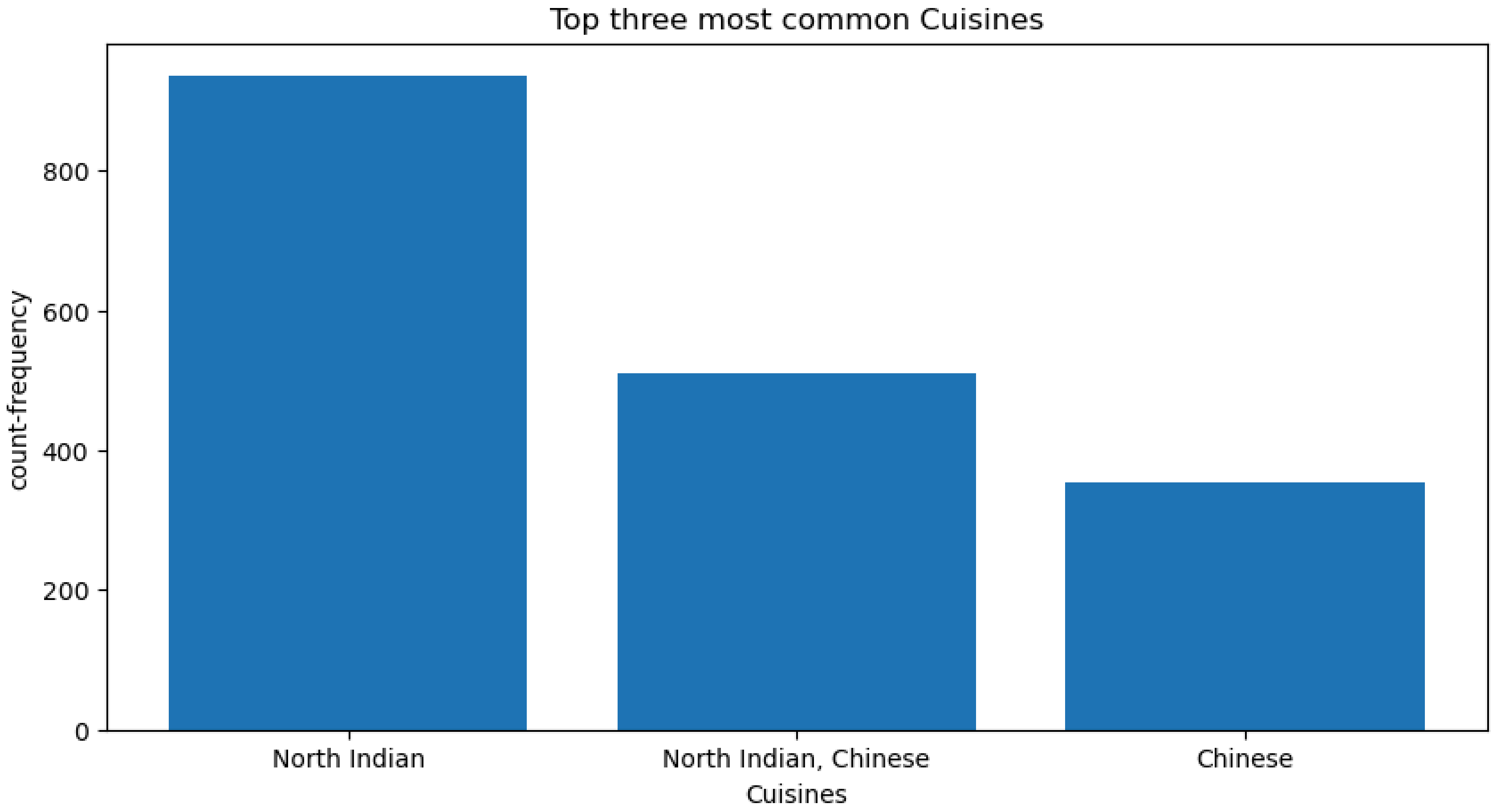
top three most common cuisines in the dataset.

In [94]:
```
top_cuisines = restaurant_df['Cuisines'].value_counts().reset_index().head(3)
top_cuisines
```

Out[94]:

	Cuisines	count
0	North Indian	936
1	North Indian, Chinese	511
2	Chinese	354

In [104…
```
plt.figure(figsize=(10,5))
values=top_cuisines['Cuisines']
lables=top_cuisines['count']
plt.bar(values,lables)
plt.title('Top three most common Cuisines')
plt.xlabel('Cuisines')
plt.ylabel('count-frequency')
plt.show()
```



Calculate the percentage of restaurants that serve each of the top cuisines.

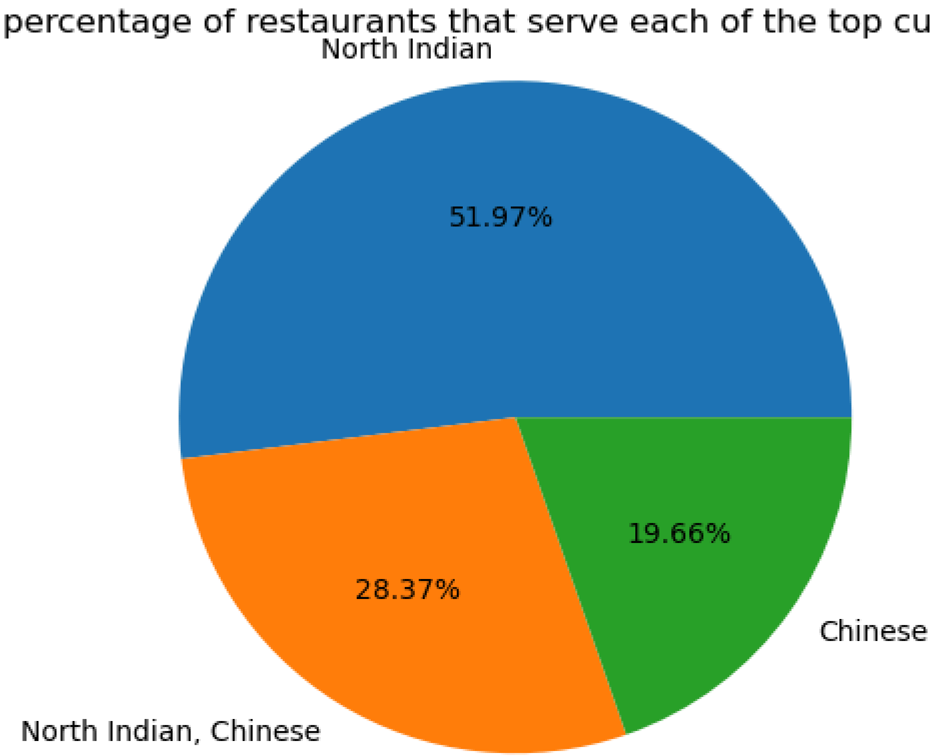
In [111…

```
# Count the occurrences of each cuisine
cuisine_counts = restaurant_df['Cuisines'].value_counts()

# Calculate the percentage of each cuisine
cuisine_percentage = (cuisine_counts / len(restaurant_df)) * 100

# Get the top 3 cuisines and their percentages
top_cuisines_percentage = cuisine_percentage.head(3)
print(round(top_cuisines_percentage,2))
```

```
Cuisines
North Indian           9.80
North Indian, Chinese  5.35
Chinese                3.71
Name: count, dtype: float64
```

In [142…

```python
plt.title('percentage of restaurants that serve each of the top cuisines')
plt.pie(top_cuisines_percentage,labels=top_cuisines_percentage.index, autopct='%0.2f%%')
plt.axis('equal')  # Equal aspect ratio ensures the pie chart is circular.
plt.show()
```



Task 2: City Analysis

Identify the city with the highest number of restaurants in the dataset.

Calculate the average rating for restaurants in each city.

Determine the city with the highest average rating.

- 1.city with the highest number of restaurants

In [159...

```
# Count the occurrences of each city
restaurant_city_counts = restaurant_df['City'].value_counts()

# Identify the city with the highest number of restaurants
highest_city = restaurant_city_counts.idxmax() #max index
highest_count = restaurant_city_counts.max()
print(f"The city with the highest number of restaurants is {highest_city} with {highest_count} restaurants.")
```

```
The city with the highest number of restaurants is New Delhi with 5473 restaurants.
```

- 2.the average rating for restaurants in each city.

In [185...

```
average_rating_by_city = restaurant_df.groupby('City')['Aggregate rating'].mean().reset_index()
print(average_rating_by_city)
```

```
                City  Aggregate rating
0          Abu Dhabi          4.300000
1               Agra          3.965000
2          Ahmedabad          4.161905
3             Albany          3.555000
4          Allahabad          3.395000
..               ...               ...
136          Weirton          3.900000
137  Wellington City          4.250000
138   Winchester Bay          3.200000
139          Yorkton          3.300000
140        ��stanbul          4.292857

[141 rows x 2 columns]
```

- 3.Determine the city with the highest average rating.

In [197…

```
# Calculate the average rating for each city
average_rating_by_city = restaurant_df.groupby('City')['Aggregate rating'].mean()

# Identify the city with the highest average rating
highest_avg_city = average_rating_by_city.idxmax()
highest_avg_rating = average_rating_by_city.max()
print(f"The city with the highest average rating is {highest_avg_city} with an average rating of {highest_avg_rating:.2f}.")
```

```
The city with the highest average rating is Inner City with an average rating of 4.90.
```

observations:

- city with the highest average rating
- inner city with 4.90 as avg rating

Task 3: Price Range Distribution

- **Create a histogram or bar chart to

visualize the distribution of price ranges among the restaurants.**

- **Calculate the percentage of restaurants

in each price range category.**

- 1.Create a bar chart to visualize the distribution of price ranges:

In [212…

```
# Count occurrences of each price range
price_range_counts = restaurant_df['Price range'].value_counts()

# Create a bar chart
plt.figure(figsize=(10,5))
price_range_counts.plot(kind='bar', color='green')
plt.title('Distribution of Price Ranges Among Restaurants')
```

```
plt.xlabel('Price Range')
plt.ylabel('Number of Restaurants')
plt.show()
```

Key Points:

- This code counts how many restaurants fall into each price range
- and visualizes that with a bar chart.

- 2.the percentage of restaurants in each price range category.

In [237…

```
# Calculate percentages
price_range_percentage = round((price_range_counts / len(restaurant_df)) * 100,2)

print("Percentage of restaurants in each price range category:")
pd.DataFrame(zip(price_range_percentage.index,price_range_counts,price_range_percentage.values),columns=['price range','count'
```

```
Percentage of restaurants in each price range category:
```

Out[237…

	price range	count	percentage
0	1	4444	46.53
1	2	3113	32.59
2	3	1408	14.74
3	4	586	6.14

Key Points:

- The value_counts() function counts the number of occurrences of each price range.
- Dividing by the total number of restaurants (len(restaurant_df))
- and multiplying by 100 gives the percentage for each category.

observations:

- percentage of restaurant in each price range category.
  - price range : 1 - percentage:46.53
  - price range : 2 - percentage:32.59
  - price range : 3 - percentage:14.74
  - price range : 4 - percentage:6.14

Task 4: Online Delivery

- **Determine the percentage of restaurants

that offer online delivery.**

- **Compare the average ratings of restaurants

with and without online delivery.**

- Determine the percentage of restaurants that offer online delivery:

In [251…
```
restaurant_df.columns
```

Out[251…
```
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 [253…
```
restaurant_df['Has Online delivery']
```

Out[253…
```
0       No
1       No
2       No
3       No
4       No
        ..
9546    No
9547    No
9548    No
9549    No
9550    No
Name: Has Online delivery, Length: 9551, dtype: object
```

In [303…
```
online_delivery_count = restaurant_df['Has Online delivery'].value_counts()
online_delivery_count
```

Out[303…

```
Has Online delivery
No     7100
Yes    2451
Name: count, dtype: int64
```

In [307…

```
total_restaurant_count=restaurant_df.shape[0]
online_restaurant_count=restaurant_df[restaurant_df['Has Online delivery']=='Yes']
online_restaurant_count=online_restaurant_count.shape[0]

online_delivery_percentage=round((online_restaurant_count/total_restaurant_count)*100,3)
print(f'percentage of online delivery taken by restaurant are:{online_delivery_percentage}')
```

```
percentage of online delivery taken by restaurant are:25.662
```

- Compare the average ratings of restaurants with and without online delivery.

In [321…

```
print('avg rating of restaurant with and without online delivery')
restaurant_df.groupby('Has Online delivery')['Aggregate rating'].mean().reset_index()
```

```
avg rating of restaurant with and without online delivery
```

Out[321…

	Has Online delivery	Aggregate rating
0	No	2.465296
1	Yes	3.248837

observation:

- assuming you have a DataFrame named restaurant_df with a column named 'Has Online delivery' indicating online delivery status and a column named 'Aggregate rating'.
- **GroupBy**: The groupby method groups the DataFrame by the 'Has Online delivery' column.
- **Mean Calculation**: It calculates the mean of the 'Aggregate rating' for each group.
- **Reset Index**: The reset_index() method converts the resulting Series back into a DataFrame.

- **Output**: The final DataFrame avg_ratings contains the average ratings for each delivery status.

In [ ]:

```

```

In [ ]:

```

```