In [1]:
```
1 import pandas as pd
```

In [2]:
```
1 df = pd.read_excel('./data.xlsx')
```

In [3]:
```
1 df.head()
```

Out[3]:

	Restaurant ID	Restaurant Name	Country Code	City	Address	Locality	Locality Verbose	Longitude	L
0	7402935	Skye	94	Jakarta	Menara BCA, Lantai 56, Jl. MH. Thamrin, Thamri...	Grand Indonesia Mall, Thamrin	Grand Indonesia Mall, Thamrin, Jakarta	106.821999	-6.
1	7410290	Satoo - Hotel Shangri-La	94	Jakarta	Hotel Shangri-La, Jl. Jend. Sudirman	Hotel Shangri-La, Sudirman	Hotel Shangri-La, Sudirman, Jakarta	106.818961	-6.
2	7420899	Sushi Masa	94	Jakarta	Jl. Tuna Raya No. 5, Penjaringan	Penjaringan	Penjaringan, Jakarta	106.800144	-6.
3	7421967	3 Wise Monkeys	94	Jakarta	Jl. Suryo No. 26, Senopati, Jakarta	Senopati	Senopati, Jakarta	106.813400	-6.
4	7422489	Avec Moi Restaurant and Bar	94	Jakarta	Gedung PIC, Jl. Teluk Betung 43, Thamrin, Jakarta	Thamrin	Thamrin, Jakarta	106.821023	-6.

In [4]:
```
1 # Get the dimensions of the DataFrame ( number of rows and columns)
2
3 num_rows, num_columns = df.shape
4 print(f"Number of rows: {num_rows}")
5 print(f"Number of columns: {num_columns}")
```

```
Number of rows: 9551
Number of columns: 19
```

In [5]:
```
1 # List all the columns in the DataFrame
2 print("Columns in the DataFrame:")
3 print(df.columns.tolist())
```

```
Columns in the DataFrame:
['Restaurant ID', 'Restaurant Name', 'Country Code', 'City', 'Address', 'Loca
lity', 'Locality Verbose', 'Longitude', 'Latitude', 'Cuisines', 'Average Cost
for two', 'Currency', 'Has Table booking', 'Has Online delivery', 'Price rang
e', 'Aggregate rating', 'Rating color', 'Rating text', 'Votes']
```

In [6]:
```
print("Data types of each column:")
print(df.dtypes)
```

```
Data types of each column:
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
Price range               int64
Aggregate rating        float64
Rating color             object
Rating text              object
Votes                     int64
dtype: object
```

In [7]:
```
# Missing values

missing_values = df.isnull().sum()
print("Missing values in each column:")
print(missing_values)
```

```
Missing values in each column:
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 [8]:
```
# Calculate the percentage of missing values.
percentage_missing = ( missing_values/ num_rows) * 100
print("Percentage of missing values in each column:")
print(percentage_missing)
```

```
Percentage of missing values in each column:
Restaurant ID           0.000000
Restaurant Name         0.010470
Country Code            0.000000
City                    0.000000
Address                 0.000000
Locality                0.000000
Locality Verbose        0.000000
Longitude               0.000000
Latitude                0.000000
Cuisines                0.094231
Average Cost for two    0.000000
Currency                0.000000
Has Table booking       0.000000
Has Online delivery     0.000000
Price range             0.000000
Aggregate rating        0.000000
Rating color            0.000000
Rating text             0.000000
Votes                   0.000000
dtype: float64
```

In [9]:
```
# Drop the missing values
df = df.dropna()
```

In [10]:
```
duplicate_rows = df[ df.duplicated() ]
print("Duplicated rows:")
print(duplicate_rows)
```

```
Duplicated rows:
Empty DataFrame
Columns: [Restaurant ID, Restaurant Name, Country Code, City, Address, Locali
ty, Locality Verbose, Longitude, Latitude, Cuisines, Average Cost for two, Cu
rrency, Has Table booking, Has Online delivery, Price range, Aggregate ratin
g, Rating color, Rating text, Votes]
Index: []
```

In [12]:
```
# Explore the geographical distribution of the restaurants and identify th
city_restaurant_counts = df['City'].value_counts()
```

In [13]:
```
1 city_restaurant_counts
```

Out[13]:
```
New Delhi         5473
Gurgaon           1118
Noida             1080
Faridabad          251
Ghaziabad           25
                  ...
Consort              1
Lincoln              1
Monroe               1
Potrero              1
Lakes Entrance       1
Name: City, Length: 140, dtype: int64
```

In [14]:
```
1 city_with_max_restaurants = city_restaurant_counts.idxmax()
2 max_restaurants_count = city_restaurant_counts.max()
```

In [15]:
```
1 city_with_min_restaurants = city_restaurant_counts.idxmin()
2 min_restaurants_count = city_restaurant_counts.min()
```

In [16]:
```
1 print(f"City with the maximum number of restaurants: {city_with_max_restau
2 print(f"Number of restaurants in the city with the maximum: {max_restauran
3 print(f"City with the minimum number of restaurants: {city_with_min_restau
4 print(f"Number of restaurants in the city with the minimum:{min_restaurant
```

```
City with the maximum number of restaurants: New Delhi
Number of restaurants in the city with the maximum: 5473
City with the minimum number of restaurants: Penola
Number of restaurants in the city with the minimum:1
```

In [17]:
```
1 franchise_presence = df.groupby(['Restaurant Name', 'Country Code']).size(
```

In [18]:
```
1 franchise_presence
```

Out[18]:

	Restaurant Name	Country Code	Presence Count
0	12212	1	1
1	Let's Burrrp	1	1
2	#45	1	1
3	#Dilliwaala6	1	1
4	#InstaFreeze	1	1
...	...	...	...
7458	t Lounge by Dilmah	1	1
7459	tashas	189	1
7460	wagamama	148	1
7461	{Niche} - Cafe & Bar	1	1
7462	Ìàukura€Ùa Sofras€±	208	1

7463 rows × 3 columns

In [19]:
```
1 max_presence_franchise = franchise_presence.loc[ franchise_presence['Prese
2
3 # Print the result
4 print("Franchise with the most national presence :")
5 print(max_presence_franchise)
```

```
Franchise with the most national presence :
Restaurant Name    Cafe Coffee Day
Country Code                     1
Presence Count                  83
Name: 1100, dtype: object
```

In [20]:
```
1 # Find out the ratio between restaurants that allow table booking vs that
2
3 table_booking_counts = df['Has Table booking'].value_counts()
```

In [21]:
```
1 ratio_booking = table_booking_counts['Yes'] / table_booking_counts['No']
```

In [22]:
```
1 print(f"Ratio of the restaurants that allow table booking vs that do not a
```

```
Ratio of the restaurants that allow table booking vs that do not allow table
booking : 0.13813670523678873
```

In [23]:
```
1 # Find the percentage of restaurants providing online booking.
2 online_delivery_counts = df['Has Online delivery'].value_counts()
```

In [24]:
```
percentage_online_delivery = ( online_delivery_counts['Yes'] / len(df) ) *

print(f"Percentage of restaurants providing online delivery: {percentage_o
```

```
Percentage of restaurants providing online delivery: 25.68913111833141%
```

In [25]:
```
# Calculate the difference in the number of votes between restaurants that

restaurants_that_deliver = df[ df['Has Online delivery'] == 'Yes']
restaurants_without_delivery = df [ df['Has Online delivery'] == 'No']

total_votes_with_delivery = restaurants_that_deliver['Votes'].sum()
total_votes_without_delivery = restaurants_without_delivery['Votes'].sum()
```

In [26]:
```
difference_in_votes = total_votes_with_delivery - total_votes_without_deli
```

In [27]:
```
print(f"Difference in the number of votes between restaurants that deliver
```

```
Difference in the number of votes between restaurants that deliver and restau
rants that do not deliver: -459322
```

In [28]:
```
df['Cuisine List'] = df['Cuisines'].str.split(',')
```

In [29]:
```
cuisine_df = df.explode('Cuisine List')
```

In [30]:
```
top_cuisines = cuisine_df['Cuisine List'].value_counts()
```

In [31]:
```
top_10_cuisines = top_cuisines.head(10)

print("Top 10 cuisines served across cities: ")
print(top_10_cuisines)
```

```
Top 10 cuisines served across cities:
North Indian     2991
 Chinese         1880
 Fast Food       1314
 North Indian     968
Chinese           855
 Mughlai          780
Fast Food         672
Bakery            621
Cafe              617
 Italian          529
Name: Cuisine List, dtype: int64
```

In [32]:
```
df['Number of Cuisines'] = df['Cuisines'].str.split(',').apply(len)

max_cuisines = df['Number of Cuisines'].max()
min_cuisines = df['Number of Cuisines'].min()

print(f"Maximum number of cuisines served by a restaurant : {max_cuisines}")
print(f"Minimum number of cuisines served by a restaruant : {min_cuisines}")
```

```
Maximum number of cuisines served by a restaurant : 8
Minimum number of cuisines served by a restaruant : 1
```

In [35]:
```
#df.explode('Cuisine List')
```

In [34]:
```
df.explode('Cuisine List').groupby(['City', 'Cuisine List']).size().reset
```

Out[34]:

	City	Cuisine List	Count
0	Abu Dhabi	Afghani	1
1	Abu Dhabi	Arabian	2
2	Abu Dhabi	Asian	1
3	Abu Dhabi	Biryani	1
4	Abu Dhabi	Burger	1
...	...	...	...
2395	€¡stanbul	Desserts	2
2396	€¡stanbul	Italian	1
2397	€¡stanbul	Restaurant Cafe	4
2398	€¡stanbul	Turkish	1
2399	€¡stanbul	World Cuisine	1

2400 rows × 3 columns

In [36]:
```
cuisine_counts = df.explode('Cuisine List').groupby(['City', 'Cuisine List
```

In [37]:
```
1 cuisine_counts
```

Out[37]:

	City	Cuisine List	Count
0	Abu Dhabi	Afghani	1
1	Abu Dhabi	Arabian	2
2	Abu Dhabi	Asian	1
3	Abu Dhabi	Biryani	1
4	Abu Dhabi	Burger	1
...	...	...	...
2395	€¡stanbul	Desserts	2
2396	€¡stanbul	Italian	1
2397	€¡stanbul	Restaurant Cafe	4
2398	€¡stanbul	Turkish	1
2399	€¡stanbul	World Cuisine	1

2400 rows × 3 columns

In [38]:
```
1 cuisine_counts.groupby('City').apply(lambda x : x [ x['Count'] == x['Count
```

Out[38]:

		City	Cuisine List	Count
City				
Abu Dhabi	**24**	Abu Dhabi	Indian	6
Agra	**45**	Agra	North Indian	12
Ahmedabad	**55**	Ahmedabad	Continental	8
	60	Ahmedabad	Italian	8
Albany	**91**	Albany	American	4
...	**...**	...	...	...
	2381	Winchester Bay	Seafood	1
Winchester Bay	**2382**	Winchester Bay	Steak	1
	2383	Winchester Bay	Burger	1
Yorkton	**2384**	Yorkton	Asian	1
€¡stanbul	**2397**	€¡stanbul	Restaurant Cafe	4

242 rows × 3 columns

In [39]:
```
1 most_served_cuisine_by_city = cuisine_counts.groupby('City').apply(lambda
```

In [40]:

```
most_served_cuisine_by_city = most_served_cuisine_by_city[['City', 'Cuisin
most_served_cuisine_by_city.columns = [ 'City', 'Most Served Cuisine']

print("Single most served cuisine for each city:")
print(most_served_cuisine_by_city)
```

```
Single most served cuisine for each city:
               City Most Served Cuisine
0         Abu Dhabi              Indian
1              Agra        North Indian
2         Ahmedabad         Continental
3         Ahmedabad             Italian
4            Albany            American
..              ...                 ...
237  Winchester Bay             Seafood
238  Winchester Bay               Steak
239  Winchester Bay              Burger
240         Yorkton               Asian
241       €¡stanbul     Restaurant Cafe

[242 rows x 2 columns]
```