Project Foundations for Data Science: FoodHub Data Analysis

Marks: 40 points

Context

The number of restaurants in New York is increasing day by day. Lots of students and busy professionals rely on those restaurants due to their hectic lifestyles. Online food delivery service is a great option for them. It provides them with good food from their favorite restaurants. A food aggregator company FoodHub offers access to multiple restaurants through a single smartphone app.

The app allows the restaurants to receive a direct online order from a customer. The app assigns a delivery person from the company to pick up the order after it is confirmed by the restaurant. The delivery person then uses the map to reach the restaurant and waits for the food package. Once the food package is handed over to the delivery person, he/she confirms the pick-up in the app and travels to the customer's location to deliver the food. The delivery person confirms the drop-off in the app after delivering the food package to the customer. The customer can rate the order in the app. The food aggregator earns money by collecting a fixed margin of the delivery order from the restaurants.

Objective

The food aggregator company has stored the data of the different orders made by the registered customers in their online portal. They want to analyze the data to get a fair idea about the demand of different restaurants which will help them in enhancing their customer experience. Suppose you are a Data Scientist at Foodhub and the Data Science team has shared some of the key questions that need to be answered. Perform the data analysis to find answers to these questions that will help the company to improve the business.

Data Description

The data contains the different data related to a food order. The detailed data dictionary is given below.

Data Dictionary

- order_id: Unique ID of the order
- customer_id: ID of the customer who ordered the food
- restaurant_name: Name of the restaurant
- cuisine_type: Cuisine ordered by the customer
- cost_of_the_order: Cost of the order
- day_of_the_week: Indicates whether the order is placed on a weekday or weekend (The weekday is from Monday to Friday and the weekend is Saturday and Sunday)
- rating: Rating given by the customer out of 5

- food_preparation_time: Time (in minutes) taken by the restaurant to prepare the food. This is calculated by taking the difference between the timestamps of the restaurant's order confirmation and the delivery person's pick-up confirmation.
- delivery_time: Time (in minutes) taken by the delivery person to deliver the food package. This is calculated by taking the difference between the timestamps of the delivery person's pick-up confirmation and drop-off information

Please read the instructions carefully before starting the project.

This is a commented Jupyter IPython Notebook file in which all the instructions and tasks to be performed are mentioned. Read along carefully to complete the project.

- Blanks '_____' are provided in the notebook that needs to be filled with an appropriate code to get the correct result. Please replace the blank with the right code snippet. With every '_____' blank, there is a comment that briefly describes what needs to be filled in the blank space.
- Identify the task to be performed correctly, and only then proceed to write the required code.
- Fill the code wherever asked by the commented lines like "# write your code here" or "# complete the code". Running incomplete code may throw an error.
- Please run the codes in a sequential manner from the beginning to avoid any unnecessary errors.
- You can the results/observations derived from the analysis here and use them to create your final presentation.

Let us start by importing the required libraries

```
# Import libraries for data manipulation
import numpy as np
import pandas as pd

# Import libraries for data visualization
import matplotlib.pyplot as plt
import seaborn as sns
```

Understanding the structure of the data

```
# uncomment and run the following lines for Google Colab
from google.colab import drive
drive.mount('/content/drive')
```

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

```
# Read the data
path = '/content/drive/MyDrive/Python course/foodhub_order.csv'
df = pd.read_csv(path)
# Returns the first 5 rows
df.head()
```

```
{"summary":"{\n  \"name\": \"df\",\n  \"rows\": 1898,\n  \"fields\":
[\n    {\n      \"column\": \"order_id\",\n      \"properties\": {\n
\"dtype\": \"number\",\n        \"std\": 548,\n        \"min\":
1476547,\n        \"max\": 1478444,\n        \"num_unique_values\":
1898,\n        \"samples\": [\n          1477722,\n          1478319,\
n          1477650\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"customer_id\",\n      \"properties\": {\n        \"dtype\":
\"number\",\n        \"std\": 113698,\n        \"min\": 1311,\n
\"max\": 405334,\n        \"num_unique_values\": 1200,\n
\"samples\": [\n          351329,\n          49987,\n          345899\
n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"restaurant_name\",\n      \"properties\": {\n        \"dtype\":
\"category\",\n        \"num_unique_values\": 178,\n
\"samples\": [\n          \"Tortaria\",\n          \"Osteria
Morini\",\n          \"Philippe Chow\"\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"cuisine_type\",\n
\"properties\": {\n        \"dtype\": \"category\",\n
\"num_unique_values\": 14,\n        \"samples\": [\n
\"Thai\",\n          \"French\",\n          \"Korean\"\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"cost_of_the_order\",\n
\"properties\": {\n        \"dtype\": \"number\",\n        \"std\":
7.48381211004957,\n        \"min\": 4.47,\n        \"max\": 35.41,\n
\"num_unique_values\": 312,\n        \"samples\": [\n          21.29,\
n          7.18,\n          13.34\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"day_of_the_week\",\n
\"properties\": {\n        \"dtype\": \"category\",\n
\"num_unique_values\": 2,\n        \"samples\": [\n
\"Weekday\",\n          \"Weekend\"\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"rating\",\n      \"properties\":
{\n        \"dtype\": \"category\",\n        \"num_unique_values\":
4,\n        \"samples\": [\n          \"5\",\n          \"4\"\n
],\n        \"semantic_type\": \"\",\n        \"description\": \"\"\n
}\n    },\n    {\n      \"column\": \"food_preparation_time\",\n
\"properties\": {\n        \"dtype\": \"number\",\n        \"std\":
4,\n        \"min\": 20,\n        \"max\": 35,\n
\"num_unique_values\": 16,\n        \"samples\": [\n          25,\n
23\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"delivery_time\",\n      \"properties\": {\n        \"dtype\":
\"number\",\n        \"std\": 4,\n        \"min\": 15,\n
\"max\": 33,\n        \"num_unique_values\": 19,\n        \"samples\":
[\n          20,\n          21\n        ],\n        \"semantic_type\":
```

\"\",\n   \"description\": \"\"\n   }\n   }\n]\
n}","type":"dataframe","variable_name":"df"}

```
df.tail()
```

{"summary":"{\n \"name\": \"df\",\n \"rows\": 5,\n \"fields\": [\n
{\n   \"column\": \"order_id\",\n   \"properties\": {\n
\"dtype\": \"number\",\n   \"std\": 513,\n   \"min\":
1476701,\n   \"max\": 1478056,\n   \"num_unique_values\":
5,\n   \"samples\": [\n   1477421,\n   1478056,\n
1477819\n   ],\n   \"semantic_type\": \"\",\n
\"description\": \"\"\n   }\n   },\n   {\n   \"column\":
\"customer_id\",\n   \"properties\": {\n   \"dtype\":
\"number\",\n   \"std\": 156441,\n   \"min\": 35309,\n
\"max\": 397537,\n   \"num_unique_values\": 5,\n
\"samples\": [\n   397537,\n   120353,\n   35309\
n   ],\n   \"semantic_type\": \"\",\n
\"description\": \"\"\n   }\n   },\n   {\n   \"column\":
\"restaurant_name\",\n   \"properties\": {\n   \"dtype\":
\"string\",\n   \"num_unique_values\": 4,\n   \"samples\":
[\n   \"The Smile\",\n   \"Jack's Wife Freda\",\n
\"Chipotle Mexican Grill $1.99 Delivery\"\n   ],\n
\"semantic_type\": \"\",\n   \"description\": \"\"\n   }\
n   },\n   {\n   \"column\": \"cuisine_type\",\n
\"properties\": {\n   \"dtype\": \"string\",\n
\"num_unique_values\": 4,\n   \"samples\": [\n
\"American\",\n   \"Mediterranean\",\n   \"Mexican\"\n
],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n
}\n   },\n   {\n   \"column\": \"cost_of_the_order\",\n
\"properties\": {\n   \"dtype\": \"number\",\n   \"std\":
5.9201494913557715,\n   \"min\": 12.18,\n   \"max\": 25.22,\
n   \"num_unique_values\": 4,\n   \"samples\": [\n
12.18,\n   19.45,\n   22.31\n   ],\n
\"semantic_type\": \"\",\n   \"description\": \"\"\n   }\
n   },\n   {\n   \"column\": \"day_of_the_week\",\n
\"properties\": {\n   \"dtype\": \"category\",\n
\"num_unique_values\": 2,\n   \"samples\": [\n
\"Weekday\",\n   \"Weekend\"\n   ],\n
\"semantic_type\": \"\",\n   \"description\": \"\"\n   }\
n   },\n   {\n   \"column\": \"rating\",\n   \"properties\":
{\n   \"dtype\": \"category\",\n   \"num_unique_values\":
2,\n   \"samples\": [\n   \"Not given\",\n   \"5\"\
n   ],\n   \"semantic_type\": \"\",\n
\"description\": \"\"\n   }\n   },\n   {\n   \"column\":
\"food_preparation_time\",\n   \"properties\": {\n
\"dtype\": \"number\",\n   \"std\": 3,\n   \"min\": 23,\n
\"max\": 31,\n   \"num_unique_values\": 3,\n   \"samples\":
[\n   31,\n   23\n   ],\n   \"semantic_type\":
\"\",\n   \"description\": \"\"\n   }\n   },\n   {\n
\"column\": \"delivery_time\",\n   \"properties\": {\n

```
\"dtype\": \"number\",\n        \"std\": 5,\n        \"min\": 17,\n
\"max\": 31,\n        \"num_unique_values\": 4,\n        \"samples\":
[\n          19,\n          31\n        ],\n        \"semantic_type\":
\"\",\n        \"description\": \"\"\n      }\n    }\n  ]\
n}","type":"dataframe"}
```


Question 1: How many rows and columns are present in the data?

```
# Check the shape of the dataset
df.shape ## Fill in the blank
```

```
(1898, 9)
```

Question 2: What are the datatypes of the different columns in the dataset?

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1898 entries, 0 to 1897
Data columns (total 9 columns):
 #   Column                 Non-Null Count  Dtype
---  ------                 --------------  -----
 0   order_id               1898 non-null   int64
 1   customer_id            1898 non-null   int64
 2   restaurant_name        1898 non-null   object
 3   cuisine_type           1898 non-null   object
 4   cost_of_the_order      1898 non-null   float64
 5   day_of_the_week        1898 non-null   object
 6   rating                 1898 non-null   object
 7   food_preparation_time  1898 non-null   int64
 8   delivery_time          1898 non-null   int64
dtypes: float64(1), int64(4), object(4)
memory usage: 133.6+ KB
```

Question 3: Are there any missing values in the data? If yes, treat them using an appropriate method.

```
# Checking for missing values in the data
df.isnull().sum()  #Write the appropriate function to print the sum of
null values for each column
```

```
order_id                0
customer_id             0
restaurant_name         0
cuisine_type            0
cost_of_the_order       0
day_of_the_week         0
rating                  0
```

```
food_preparation_time    0
delivery_time            0
dtype: int64
```

Question 4: Check the statistical summary of the data. What is the minimum, average, and maximum time it takes for food to be prepared once an order is placed?

```
# Get the summary statistics of the numerical data
df.describe() ## Write the appropriate function to print the
statitical summary of the data (Hint - you have seen this in the case
studies before)
```


```
{"summary":"{\n  \"name\": \"df\",\n  \"rows\": 8,\n  \"fields\": [\n
{\n      \"column\": \"order_id\",\n      \"properties\": {\n
\"dtype\": \"number\",\n        \"std\": 683381.6954349227,\n
\"min\": 548.0497240214614,\n        \"max\": 1478444.0,\n
\"num_unique_values\": 7,\n        \"samples\": [\n          1898.0,\n
1477495.5,\n          1477969.75\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"customer_id\",\n
\"properties\": {\n        \"dtype\": \"number\",\n        \"std\":
136848.58768663486,\n        \"min\": 1311.0,\n        \"max\":
405334.0,\n        \"num_unique_values\": 8,\n        \"samples\": [\n
171168.478398314,\n          128600.0,\n          1898.0\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"cost_of_the_order\",\n
\"properties\": {\n        \"dtype\": \"number\",\n        \"std\":
665.43708115231,\n        \"min\": 4.47,\n        \"max\": 1898.0,\n
\"num_unique_values\": 8,\n        \"samples\": [\n
16.498851422550054,\n          14.14,\n          1898.0\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"food_preparation_time\",\n
\"properties\": {\n        \"dtype\": \"number\",\n        \"std\":
662.6216207031504,\n        \"min\": 4.6324807759288555,\n
\"max\": 1898.0,\n        \"num_unique_values\": 8,\n
\"samples\": [\n          27.371970495258168,\n          27.0,\n
1898.0\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"delivery_time\",\n      \"properties\": {\n        \"dtype\":
\"number\",\n        \"std\": 663.516466506826,\n        \"min\":
4.972636933991106,\n        \"max\": 1898.0,\n
\"num_unique_values\": 8,\n        \"samples\": [\n
24.161749209694417,\n          25.0,\n          1898.0\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    }\n  ]\n}","type":"dataframe"}
```

Question 5: How many orders are not rated?

```
df['rating'].value_counts() ## Complete the code
```

```
rating
Not given    736
5            588
4            386
3            188
Name: count, dtype: int64
```

Exploratory Data Analysis (EDA)

Univariate Analysis

Question 6: Explore all the variables and provide observations on their distributions. (Generally, histograms, boxplots, countplots, etc. are used for univariate exploration)

Order ID

```
# check unique order ID
df['order_id'].nunique()
```

```
1898
```

Customer ID

```
# check unique customer ID
df['customer_id'].nunique()  ## Complete the code to find out number
of unique Customer ID
```

```
1200
```

Restaurant name

```
# check unique Restaurant Name
df['restaurant_name'].nunique()  ## Complete the code to find out
number of unique Restaurant Name
```

```
178
```

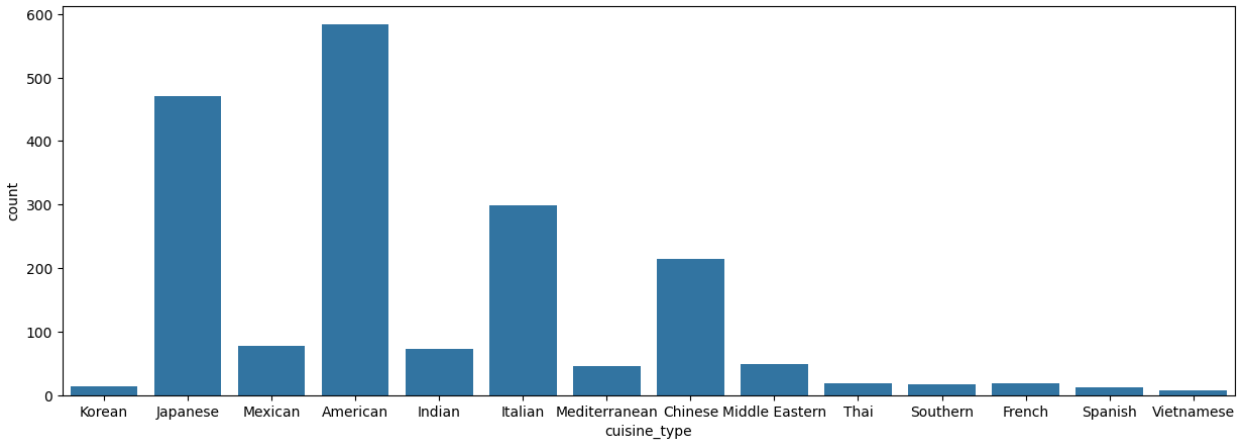
Cuisine type

```
# Check unique cuisine type
df['cuisine_type'].nunique() ## Complete the code to find out  number
of  unique cuisine type
```

```
14
```

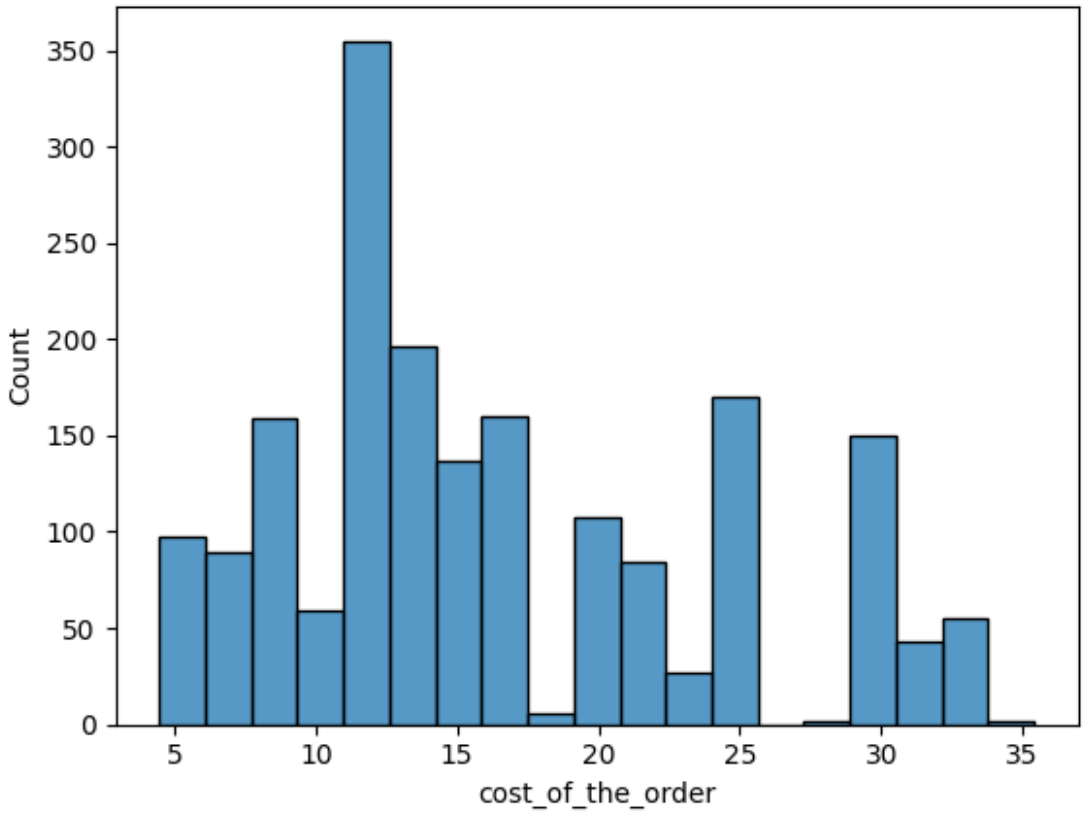
```
plt.figure(figsize = (15,5))
sns.countplot(data = df, x = 'cuisine_type') ## Create a countplot for cuisine type.
```

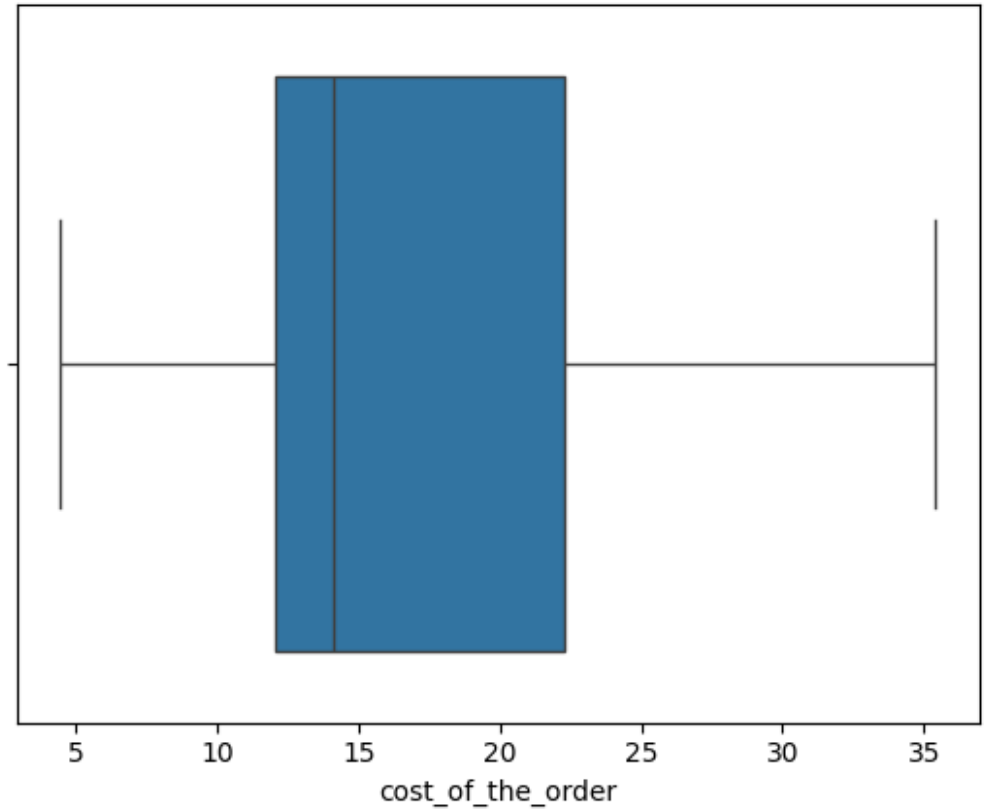
```
<Axes: xlabel='cuisine_type', ylabel='count'>
```



Cost of the order

```
sns.histplot(data=df,x='cost_of_the_order') ## Histogram for the cost of order
plt.show()
sns.boxplot(data=df,x='cost_of_the_order') ## Boxplot for the cost of order
plt.show()
```


Count
350
300
250
200
150
100
50
0
5
10
15
20
25
30
35
cost_of_the_order


5
10
15
20
25
30
35
cost_of_the_order

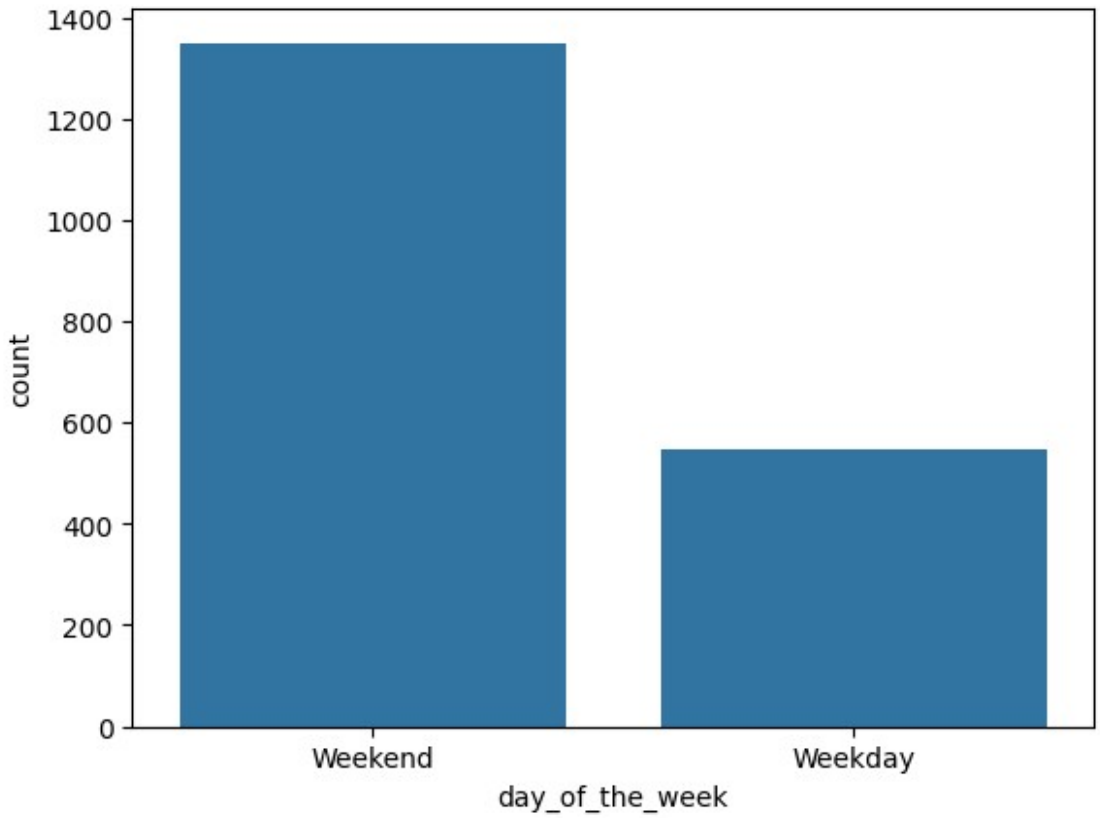
Day of the week

```
# # Check the unique values
df['day_of_the_week'].unique() ## Complete the code to check unique values for the 'day_of_the_week' column
```

```
array(['Weekend', 'Weekday'], dtype=object)
```

```
sns.countplot(data = df, x ='day_of_the_week') ## Complete the code to plot a bar graph for 'day_of_the_week' column
```

```
<Axes: xlabel='day_of_the_week', ylabel='count'>
```



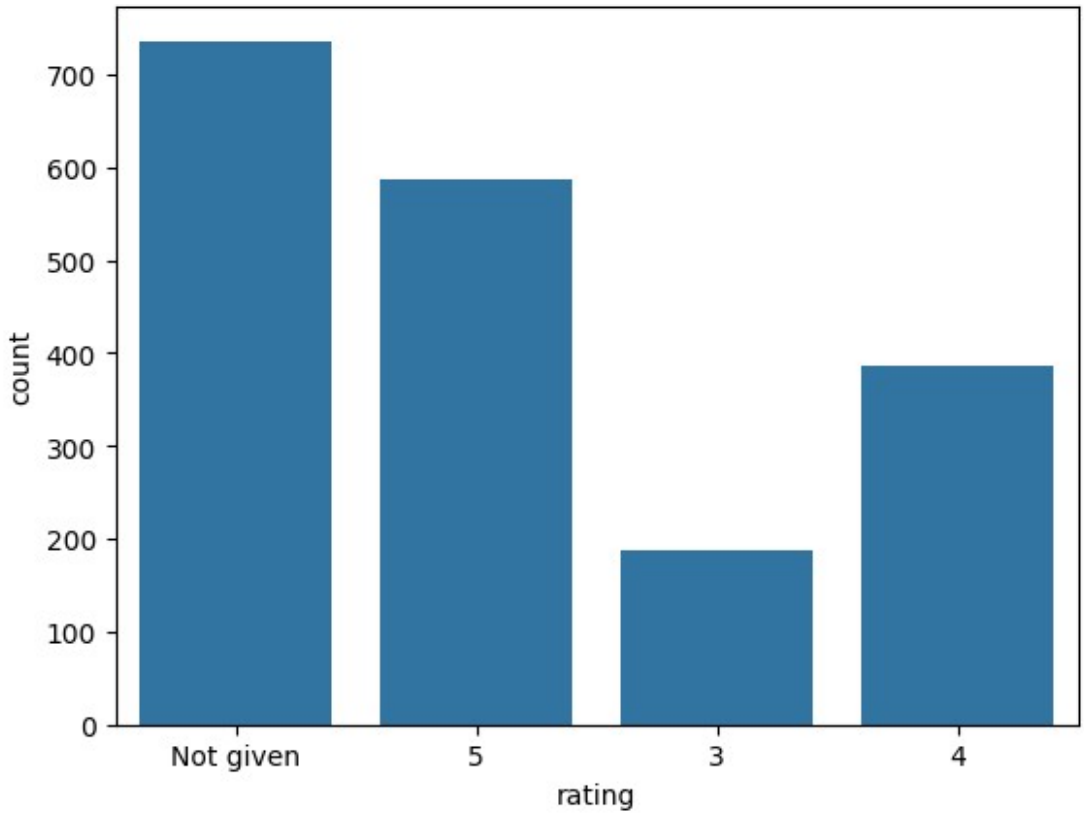
Rating

```
# Check the unique values
df['rating'].unique() ## Complete the code to check unique values for the 'rating' column
```

```
array(['Not given', '5', '3', '4'], dtype=object)
```

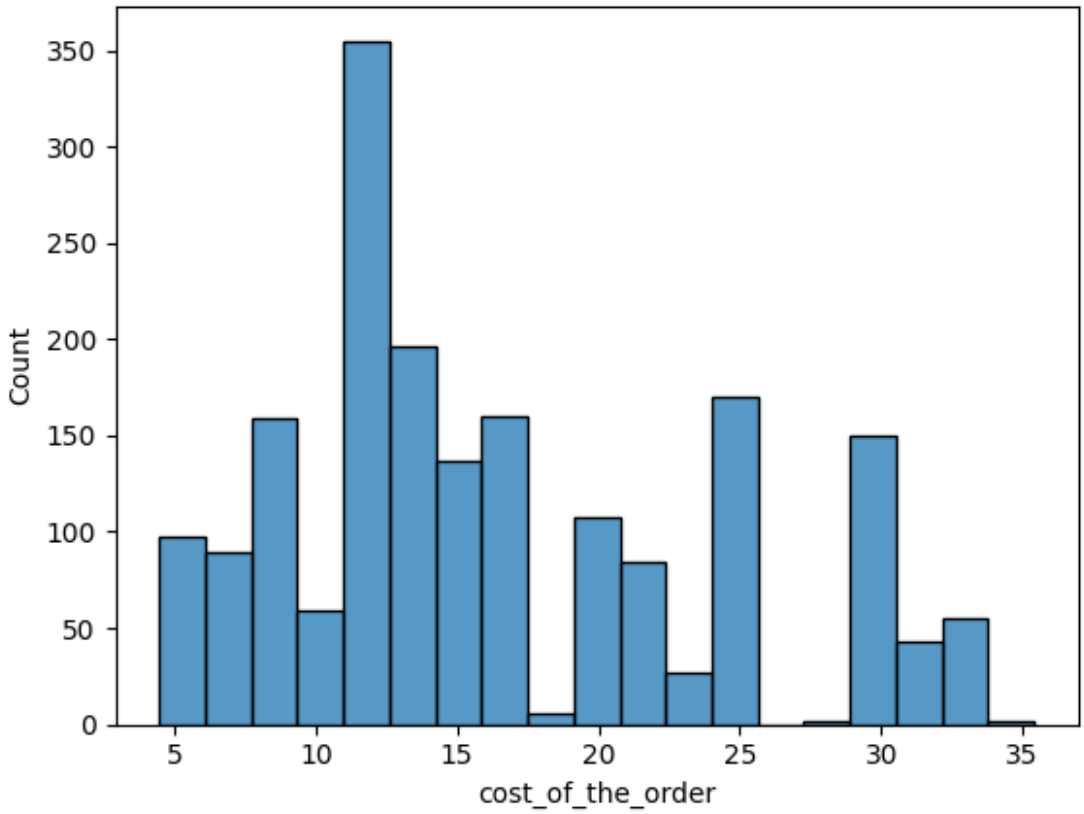
```
sns.countplot(data = df, x ='rating') ## Complete the code to plot bar graph for 'rating' column
```

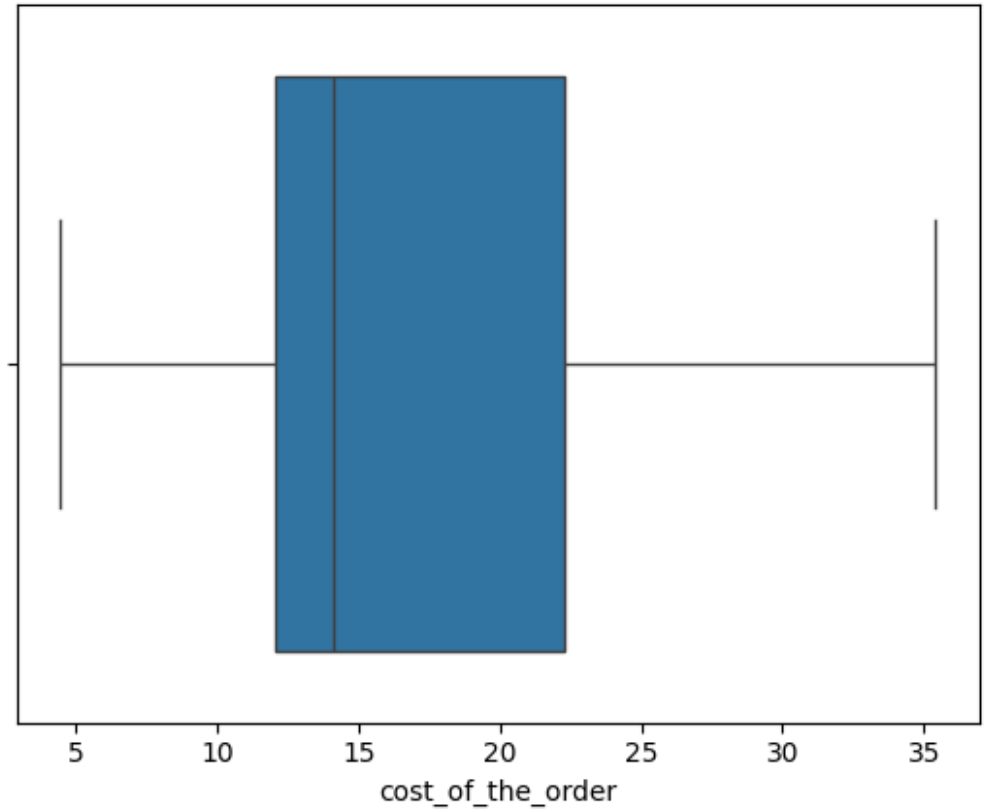
```
<Axes: xlabel='rating', ylabel='count'>
```



Food Preparation time

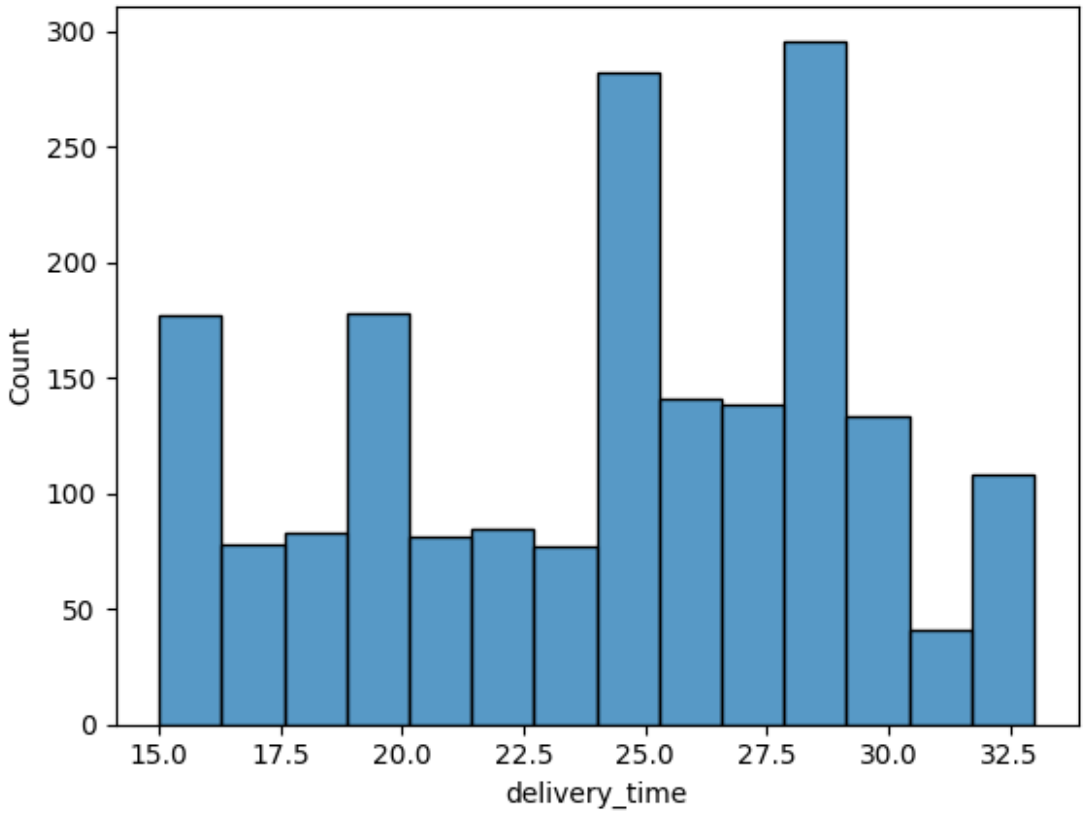
```
sns.histplot(data=df,x='cost_of_the_order') ## Complete the code to
plot the histogram for the cost of order
plt.show()
sns.boxplot(data=df,x='cost_of_the_order') ## Complete the code to
plot the boxplot for the cost of order
plt.show()
```

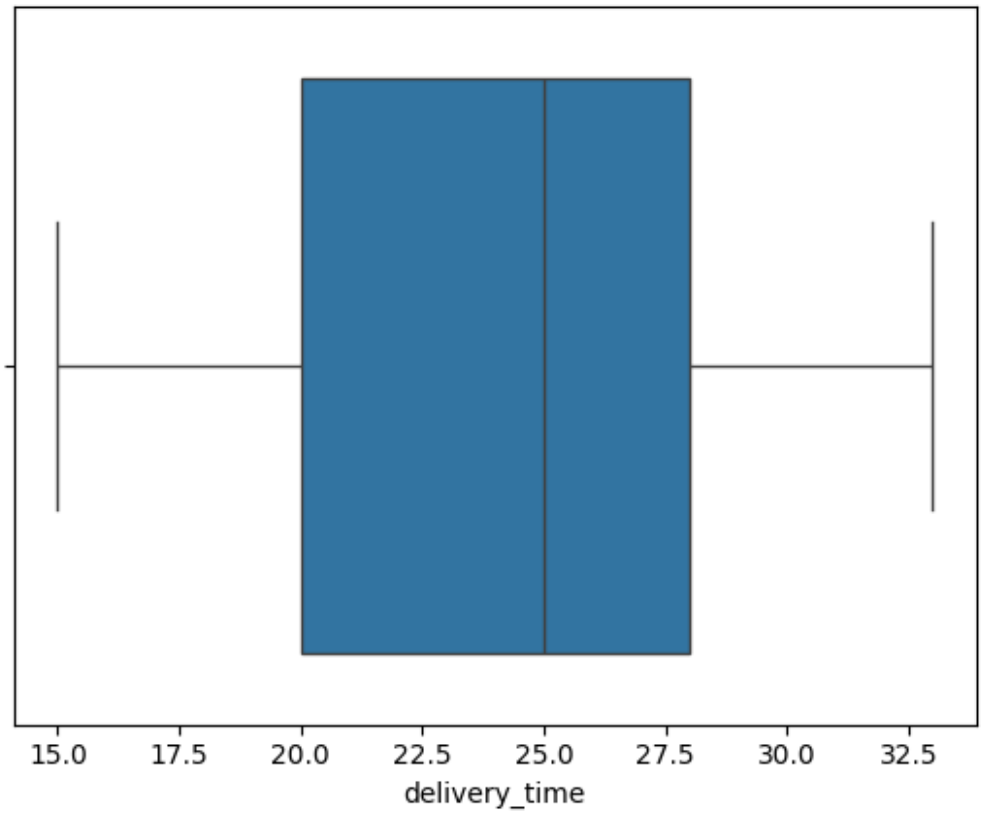

Count
350
300
250
200
150
100
50
0
5
10
15
20
25
30
35
cost_of_the_order


5
10
15
20
25
30
35
cost_of_the_order

Delivery time

```
sns.histplot(data=df,x='delivery_time') ## Complete the code to plot
the histogram for the delivery time
plt.show()
sns.boxplot(data=df,x='delivery_time') ## Complete the code to plot
the boxplot for the delivery time
plt.show()
```





Question 7: Which are the top 5 restaurants in terms of the number of orders received?

```
# Get top 5 restaurants with highest number of orders
df['restaurant_name'].sort_values().head(5) ## Complete the code
```

```
1877     'wichcraft
1583      12 Chairs
1457      12 Chairs
910       12 Chairs
925       12 Chairs
Name: restaurant_name, dtype: object
```

Question 8: Which is the most popular cuisine on weekends?

```
# Get most popular cuisine on weekends
df_weekend = df[df['day_of_the_week'] == 'Weekend']
df_weekend['cuisine_type'].unique() ## Complete the code to check
unique values for the cuisine type on weekend
```

```
array(['Korean', 'Japanese', 'American', 'Italian', 'Mexican',
       'Mediterranean', 'Chinese', 'Indian', 'Thai', 'Southern',
'French',
       'Spanish', 'Middle Eastern', 'Vietnamese'], dtype=object)
```

Question 9: What percentage of the orders cost more than 20 dollars?

```
# Get orders that cost above 20 dollars
df_greater_than_20 = df[df['cost_of_the_order']>20] ## Write the
appropriate column name to get the orders having cost above $20

# Calculate the number of total orders where the cost is above 20
dollars
print('The number of total orders that cost above 20 dollars is:',
df_greater_than_20.shape[0])

# Calculate percentage of such orders in the dataset
percentage = (df_greater_than_20.shape[0] / df.shape[0]) * 100

print("Percentage of orders above 20 dollars:", round(percentage, 2),
'%')
```

```
The number of total orders that cost above 20 dollars is: 555
Percentage of orders above 20 dollars: 29.24 %
```

Question 10: What is the mean order delivery time?

```
# Get the mean delivery time
mean_del_time = df['delivery_time'].mean()  ## Write the appropriate
function to obtain the mean delivery time

print('The mean delivery time for this dataset is',
round(mean_del_time, 2), 'minutes')
```

```
The mean delivery time for this dataset is 24.16 minutes
```

Question 11: The company has decided to give 20% discount vouchers to the top 3 most frequent customers. Find the IDs of these customers and the number of orders they placed

```
# Get the counts of  each customer_id
df['customer_id'].value_counts().head(5)  ## Write the appropriate
column name to get the top 5 cmost frequent customers
```

```
customer_id
52832     13
47440     10
83287      9
250494     8
259341     7
Name: count, dtype: int64
```

Multivariate Analysis

Question 12: Perform a multivariate analysis to explore relationships between the important variables in the dataset. (It is a good idea to explore relations between numerical variables as well as relations between numerical and categorical variables)

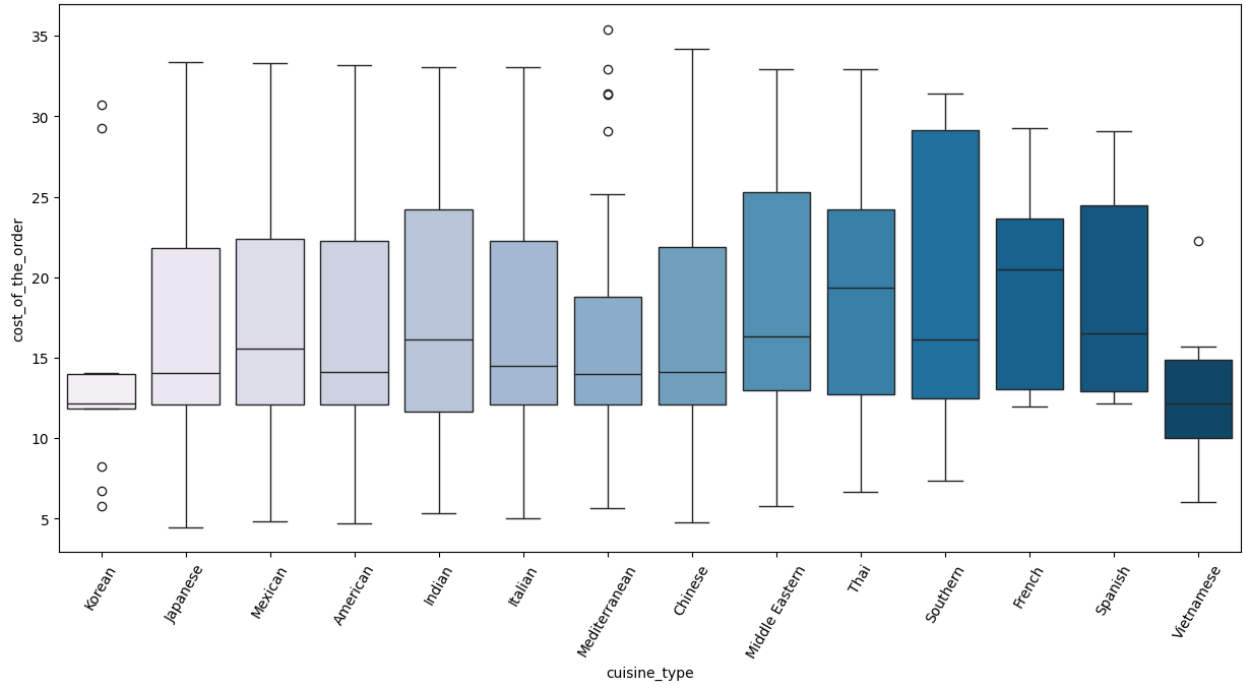
Cuisine vs Cost of the order

```
# Relationship between cost of the order and cuisine type
plt.figure(figsize=(15,7))
sns.boxplot(x = "cuisine_type", y = "cost_of_the_order", data = df,
palette = 'PuBu')
plt.xticks(rotation = 60)
plt.show()
```

```
<ipython-input-42-d4845c8bfb45>:3: 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 = "cuisine_type", y = "cost_of_the_order", data = df,
palette = 'PuBu')
```



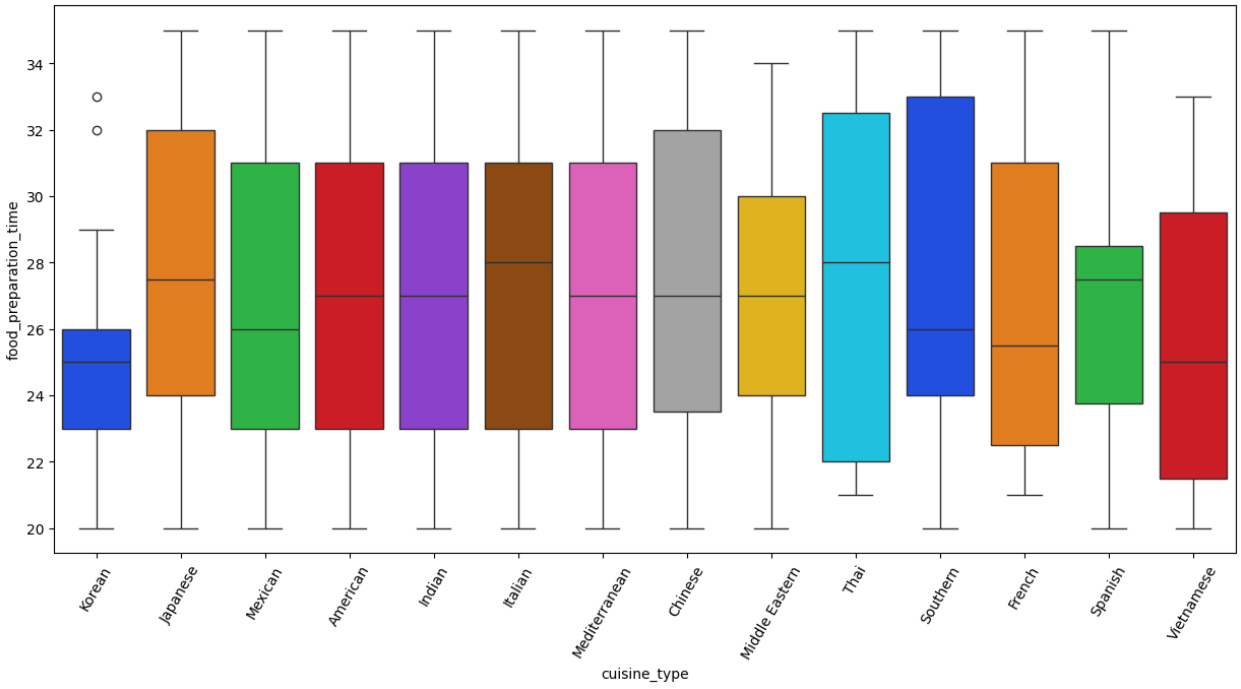
Cuisine vs Food Preparation time

```
# Relationship between food preparation time and cuisine type
plt.figure(figsize=(15,7))
sns.boxplot(data=df, x='cuisine_type', y='food_preparation_time',
palette='bright')  ## Complete the code to visualize the relationship
between food preparation time and cuisine type using boxplot
plt.xticks(rotation = 60)
plt.show()
```

```
<ipython-input-44-feb017ed52f5>:3: 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(data=df, x='cuisine_type', y='food_preparation_time',
palette='bright')  ## Complete the code to visualize the relationship
between food preparation time and cuisine type using boxplot
```



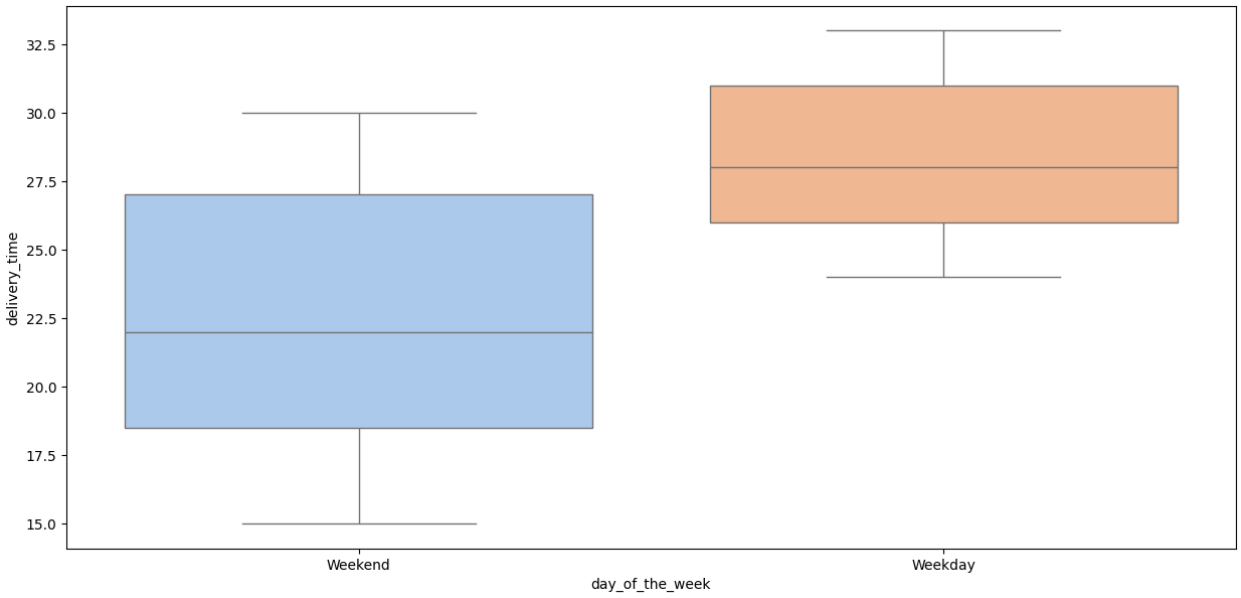
Day of the Week vs Delivery time

```
# Relationship between day of the week and delivery time
plt.figure(figsize=(15,7))
sns.boxplot(data=df, x='day_of_the_week', y='delivery_time',
palette='pastel')  ## Complete the code to visualize the relationship
between day of the week and delivery time using boxplot
plt.show()
```

```
<ipython-input-46-031b6f2d0250>:3: 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(data=df, x='day_of_the_week', y='delivery_time', palette='pastel')  ## Complete the code to visualize the relationship between day of the week and delivery time using boxplot
```



Run the below code and write your observations on the revenue generated by the restaurants

```
df.groupby(['restaurant_name'])
['cost_of_the_order'].sum().sort_values(ascending = False).head(14)
```

```
restaurant_name
Shake Shack                      3579.53
The Meatball Shop                2145.21
Blue Ribbon Sushi                1903.95
Blue Ribbon Fried Chicken        1662.29
Parm                             1112.76
RedFarm Broadway                  965.13
RedFarm Hudson                    921.21
TAO                               834.50
Han Dynasty                       755.29
Blue Ribbon Sushi Bar & Grill     666.62
Rubirosa                          660.45
Sushi of Gari 46                  640.87
Nobu Next Door                    623.67
```

```
Five Guys Burgers and Fries       506.47
Name: cost_of_the_order, dtype: float64
```

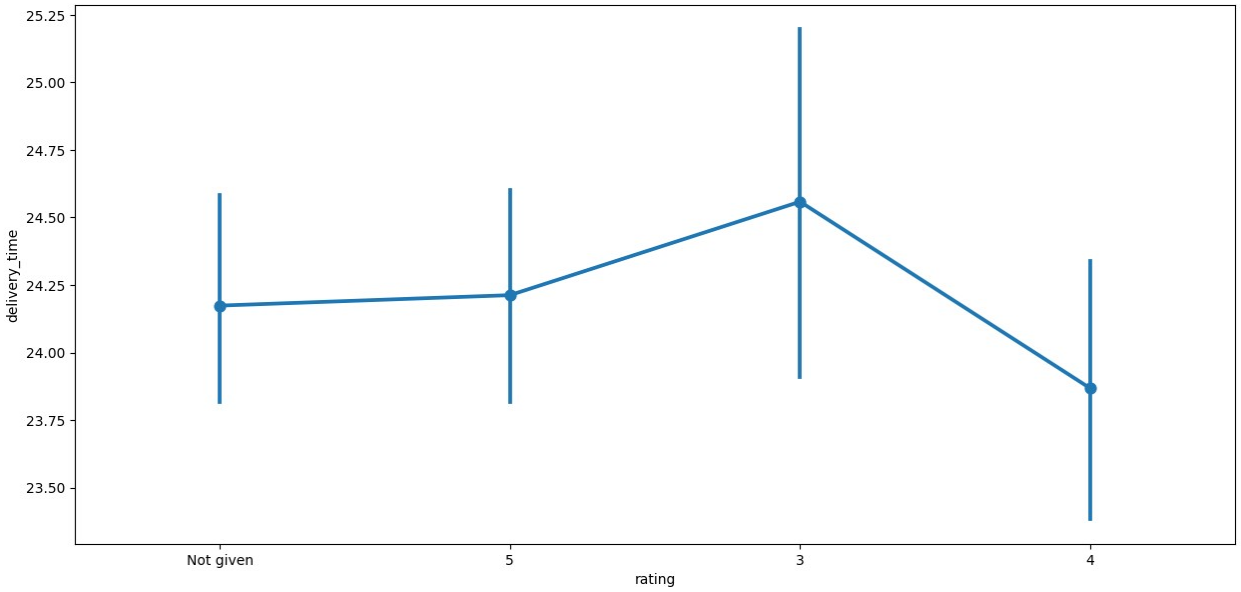
```
df['restaurant_name'].value_counts().head(14)
```

```
restaurant_name
Shake Shack                      219
The Meatball Shop                132
Blue Ribbon Sushi                119
Blue Ribbon Fried Chicken         96
Parm                              68
RedFarm Broadway                  59
RedFarm Hudson                    55
TAO                               49
Han Dynasty                       46
Blue Ribbon Sushi Bar & Grill     44
Nobu Next Door                    42
Rubirosa                          37
Sushi of Gari 46                  37
Momoya                            30
Name: count, dtype: int64
```

Based on the information given above, it is evident that shack shack is highest revenue generator restaurant with $3579.53 revenue by completing 219 orders so far. The meatball shop with revenue of $2145.21 is placed second highest in this list. The top 3 restaurants generated significant values with the minimal drop. However, Blue ribbon fried chicken ($1662.29)had a substantial downfall of revenue compared to Blue ribbon shushi($1903.95). The top 5 restaurants performed well among total 178 restaurants in the list. Red farm broadway and red farm hudson generated approximately similar revenues with minimal differences. These two restaurants are on the verge of getting on the top and these two can improve by implementing better services. Five guys burgers and fries is the lowest revenue generator in the list with only $507.46 which is around 6 times less than the highest revenue. By taking consideration of the fact that american cuisine is most ordered by the customers, Five guys burgers and fries couldn't reach even in the mid-level. Considering red farm broadway and red farm hudson, location is the major aspect in the difference between the revenues of these two restaurants.

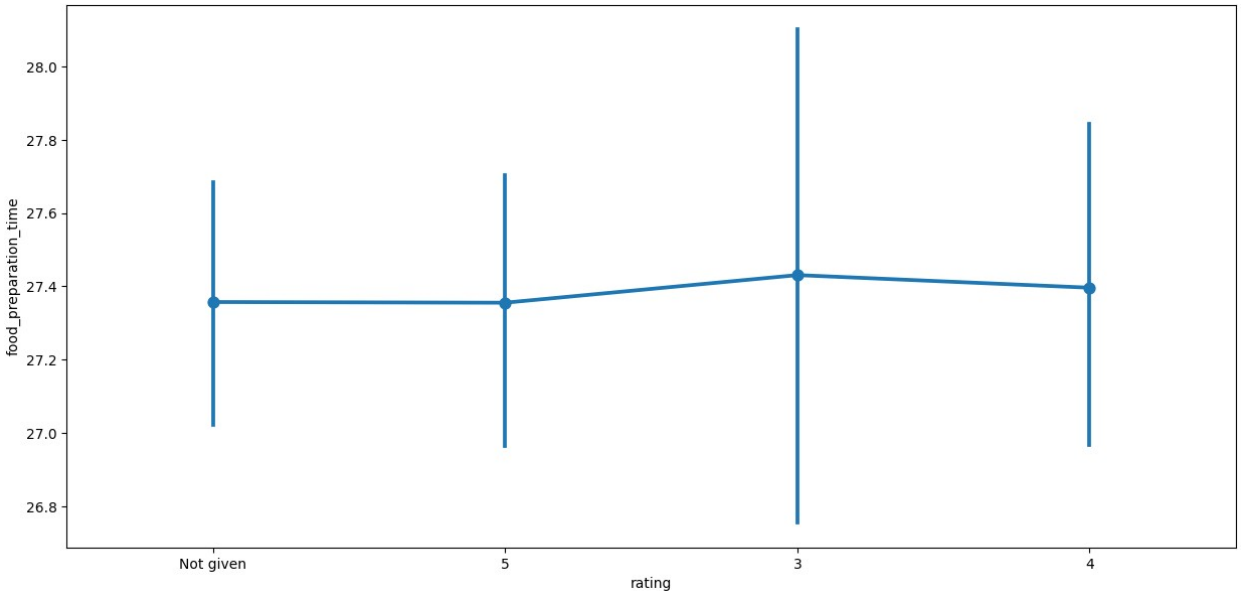
Rating vs Delivery time

```
# Relationship between rating and delivery time
plt.figure(figsize=(15, 7))
sns.pointplot(x = 'rating', y = 'delivery_time', data = df)
plt.show()
```



Rating vs Food preparation time

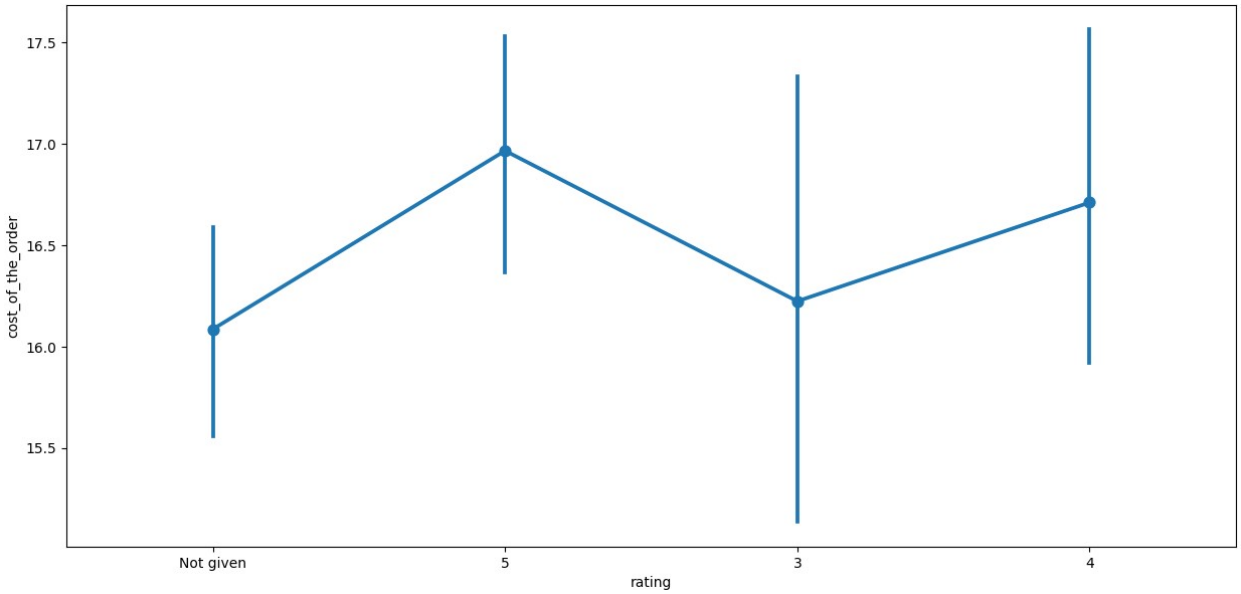
```
# Relationship between rating and food preparation time
plt.figure(figsize=(15, 7))
sns.pointplot(data=df, x='rating', y='food_preparation_time')  ## Complete the code to visualize the relationship between rating and food preparation time using pointplot
plt.show()
```



Rating vs Cost of the order

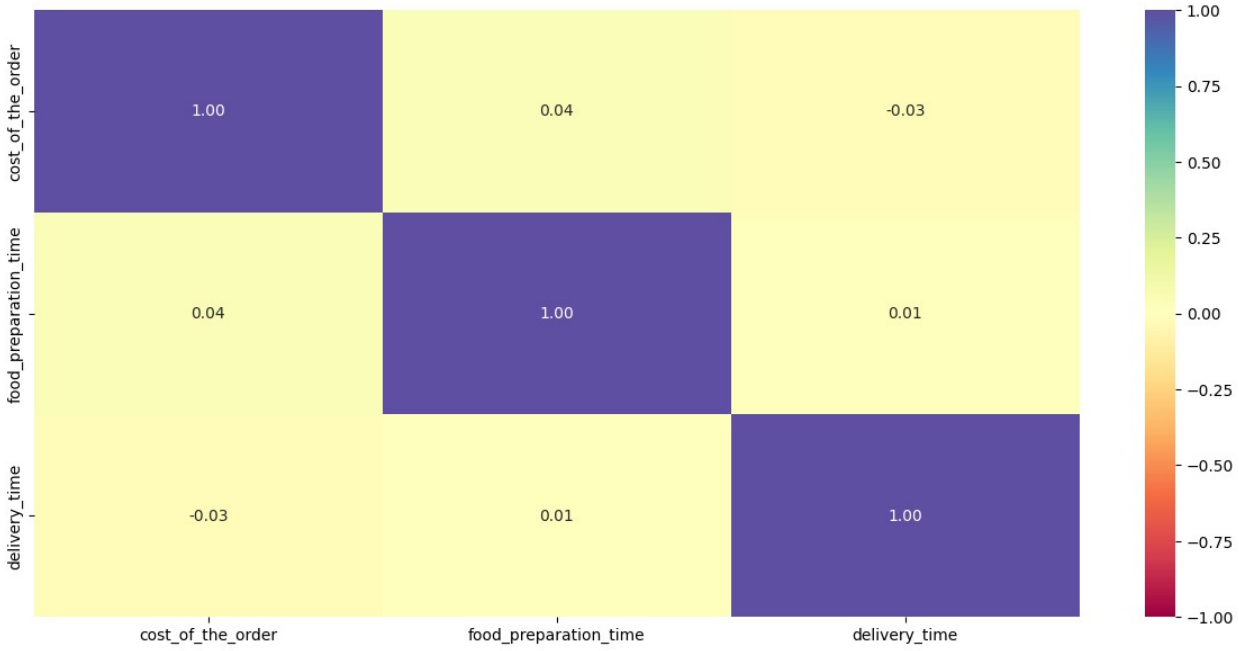
```
# Relationship between rating and cost of the order
plt.figure(figsize=(15, 7))
```

```
sns.pointplot(data=df, x='rating', y='cost_of_the_order')   ##
Complete the code to visualize the relationship between rating and
cost of the order using pointplot
plt.show()
```



Correlation among variables

```
# Plot the heatmap
col_list = ['cost_of_the_order', 'food_preparation_time',
'delivery_time']
plt.figure(figsize=(15, 7))
sns.heatmap(df[col_list].corr(), annot=True, vmin=-1, vmax=1,
fmt=".2f", cmap="Spectral")
plt.show()
```



Question 13: The company wants to provide a promotional offer in the advertisement of the restaurants. The condition to get the offer is that the restaurants must have a rating count of more than 50 and the average rating should be greater than 4. Find the restaurants fulfilling the criteria to get the promotional offer

```
# Filter the rated restaurants
df_rated = df[df['rating'] != 'Not given'].copy()

# Convert rating column from object to integer
df_rated['rating'] = df_rated['rating'].astype('int')

# Create a dataframe that contains the restaurant names with their
rating counts
df_rating_count = df_rated.groupby(['restaurant_name'])
['rating'].count().sort_values(ascending = False).reset_index()
df_rating_count.head()
```


```
{"summary":"{\n  \"name\": \"df_rating_count\",\n  \"rows\": 156,\n
\"fields\": [\n    {\n      \"column\": \"restaurant_name\",\n
\"properties\": {\n        \"dtype\": \"string\",\n
\"num_unique_values\": 156,\n        \"samples\": [\n
\"Benihana\",\n          \"Dickson's Farmstand Meats\",\n
\"Le Grainne Cafe\"\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"rating\",\n      \"properties\": {\n        \"dtype\": \"number\",\n
\"std\": 15,\n        \"min\": 1,\n        \"max\": 133,\n
\"num_unique_values\": 29,\n        \"samples\": [\n          2,\n
13,\n          19\n        ],\n        \"semantic_type\": \"\",\n
```

```
\"description\": \"\"\n      }\n    }\n  ]\
n}","type":"dataframe","variable_name":"df_rating_count"}

# Get the restaurant names that have rating count more than 50
rest_names = df_rating_count[df_rating_count['rating']>50]
['restaurant_name'] ## Complete the code to get the restaurant names
having rating count more than 50

# Filter to get the data of restaurants that have rating count more
than 50
df_mean_4 =
df_rated[df_rated['restaurant_name'].isin(rest_names)].copy()

# Group the restaurant names with their ratings and find the mean
rating of each restaurant
df_mean_4.groupby(['restaurant_name'])
['rating'].mean().sort_values(ascending =
False).reset_index().dropna() ## Complete the code to find the mean
rating

{"summary":"{\n  \"name\": \"df_mean_4\",\n  \"rows\": 4,\n
\"fields\": [\n    {\n      \"column\": \"restaurant_name\",\n
\"properties\": {\n        \"dtype\": \"string\",\n
\"num_unique_values\": 4,\n        \"samples\": [\n          \"Blue
Ribbon Fried Chicken\",\n          \"Blue Ribbon Sushi\",\n
\"The Meatball Shop\"\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"rating\",\n      \"properties\": {\n        \"dtype\": \"number\",\n
\"std\": 0.1264678402938812,\n        \"min\": 4.219178082191781,\n
\"max\": 4.511904761904762,\n        \"num_unique_values\": 4,\n
\"samples\": [\n          4.328125,\n          4.219178082191781,\n
4.511904761904762\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    }\n  ]\n}","type":"dataframe"}
```

Following above data, we can clearly observe that only 4 restaurants are eligible for the promotional offer. The eligible restaurants are

- The meatball shop
- Blue ribbon fried chicken
- shack shack
- blue ribbon shushi

Question 14: The company charges the restaurant 25% on the orders having cost greater than 20 dollars and 15% on the orders having cost greater than 5 dollars. Find the net revenue generated by the company across all orders

```
#function to determine the revenue
def compute_rev(x):
```

```
    if x > 20:
        return x*0.25
    elif x > 5:
        return x*0.15
    else:
        return x*0
```

```
df['Revenue'] = df['order_id'].apply(compute_rev) ## Write the
apprpriate column name to compute the revenue
df.head()
```


```
{"summary":"{\n  \"name\": \"df\",\n  \"rows\": 1898,\n  \"fields\":
[\n    {\n      \"column\": \"order_id\",\n      \"properties\": {\n
\"dtype\": \"number\",\n        \"std\": 548,\n        \"min\":
1476547,\n        \"max\": 1478444,\n        \"num_unique_values\":
1898,\n        \"samples\": [\n          1477722,\n          1478319,\
n          1477650\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"customer_id\",\n      \"properties\": {\n        \"dtype\":
\"number\",\n        \"std\": 113698,\n        \"min\": 1311,\n
\"max\": 405334,\n        \"num_unique_values\": 1200,\n
\"samples\": [\n          351329,\n          49987,\n          345899\
n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"restaurant_name\",\n      \"properties\": {\n        \"dtype\":
\"category\",\n        \"num_unique_values\": 178,\n
\"samples\": [\n          \"Tortaria\",\n          \"Osteria
Morini\",\n          \"Philippe Chow\"\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"cuisine_type\",\n
\"properties\": {\n        \"dtype\": \"category\",\n
\"num_unique_values\": 14,\n        \"samples\": [\n
\"Thai\",\n          \"French\",\n          \"Korean\"\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"cost_of_the_order\",\n
\"properties\": {\n        \"dtype\": \"number\",\n        \"std\":
7.48381211004957,\n        \"min\": 4.47,\n        \"max\": 35.41,\n
\"num_unique_values\": 312,\n        \"samples\": [\n          21.29,\
n          7.18,\n          13.34\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"day_of_the_week\",\n
\"properties\": {\n        \"dtype\": \"category\",\n
\"num_unique_values\": 2,\n        \"samples\": [\n
\"Weekday\",\n          \"Weekend\"\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"rating\",\n      \"properties\":
{\n        \"dtype\": \"category\",\n        \"num_unique_values\":
4,\n        \"samples\": [\n          \"5\",\n          \"4\"\n
],\n        \"semantic_type\": \"\",\n        \"description\": \"\"\n
}\n    },\n    {\n      \"column\": \"food_preparation_time\",\n
```

```
\"properties\": {\n        \"dtype\": \"number\",\n        \"std\":
4,\n        \"min\": 20,\n        \"max\": 35,\n
\"num_unique_values\": 16,\n        \"samples\": [\n          25,\n
23\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"delivery_time\",\n      \"properties\": {\n        \"dtype\":
\"number\",\n        \"std\": 4,\n        \"min\": 15,\n
\"max\": 33,\n        \"num_unique_values\": 19,\n        \"samples\":
[\n          20,\n          21\n        ],\n        \"semantic_type\":
\"\",\n        \"description\": \"\"\n      }\n    },\n    {\n
\"column\": \"Revenue\",\n      \"properties\": {\n        \"dtype\":
\"number\",\n        \"std\": 137.01243100536536,\n        \"min\":
369136.75,\n        \"max\": 369611.0,\n        \"num_unique_values\":
1898,\n        \"samples\": [\n          369430.5,\n
369579.75\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    }\n  ]\
n}","type":"dataframe","variable_name":"df"}

# get the total revenue and print it
total_rev = df['Revenue'].sum() ## Write the appropriate function to
get the total revenue
print('The net revenue is around', round(total_rev, 2), 'dollars')
```

```
The net revenue is around 701071614.75 dollars
```

Question 15: The company wants to analyze the total time required to deliver the food. What percentage of orders take more than 60 minutes to get delivered from the time the order is placed? (The food has to be prepared and then delivered)

```
# Calculate total delivery time and add a new column to the dataframe
df to store the total delivery time
df['total_time'] = df['food_preparation_time'] + df['delivery_time']


## Write the code below to find the percentage of orders that have
more than 60 minutes of total delivery time (see Question 9 for
reference)
df_greater_than_60 = df[df['total_time']>60]
percentage = (df_greater_than_60.shape[0] / df.shape[0]) * 100
print("Percentage of orders that take more than 60 minutes to get
delivered:", round(percentage, 2), '%')
```

```
Percentage of orders that take more than 60 minutes to get delivered:
10.54 %
```

Question 16: The company wants to analyze the delivery time of the orders on weekdays and weekends. How does the mean delivery time vary during weekdays and weekends?

```
# Get the mean delivery time on weekdays and print it
print('The mean delivery time on weekdays is around',
      round(df[df['day_of_the_week'] == 'Weekday']
['delivery_time'].mean()),
     'minutes')


## Write the code below to get the mean delivery time on weekends and
print it
# Get the mean delivery time on weekdays and print it
print('The mean delivery time on weekdays is around',
      round(df[df['day_of_the_week'] == 'Weekday']
['delivery_time'].mean()),
     'minutes')


## Write the code below to get the mean delivery time on weekends and
print it
mean_delivery_time_weekends = df[df['day_of_the_week'] == 'Weekends']
['delivery_time'].mean()

if pd.isna(mean_delivery_time_weekends):
    print('The mean delivery time on weekends cannot be calculated due
to missing data.')
else:
    print('The mean delivery time on weekends is around',
round(mean_delivery_time_weekends), 'minutes')



The mean delivery time on weekdays is around 28 minutes
The mean delivery time on weekdays is around 28 minutes
The mean delivery time on weekends cannot be calculated due to missing
data.
```

Question 17: What are your conclusions from the analysis? What recommendations would you like to share to help improve the business? (You can use cuisine type and feedback ratings to drive your business recommendations)

objective:

- The food aggregator company has stored the data of the different orders made by the registered customers in their online portal. They want to analyze the data to get a fair idea about the demand of different restaurants which will help them in

enhancing their customer experience. Suppose you are a Data Scientist at Foodhub and the Data Science team has shared some of the key questions that need to be answered. Perform the data analysis to find answers to these questions that will help the company to improve the business.

conclusions:

There are total 1898 rows and 9 columns in the data. there are no missing values in the data. The mean and median values of the food preparation time lies close which suggests that there is negligible skew. After reviewing the ' column, it is observable that customers have not given 2 or 1 rating to any restaurant. Maximum customers(736) haven't given any ratings to any order, followed by 5 rating given by 588 most satisfied customers. Out of total 14 cuisines, American cuisine leads food category and Japanese being the second most preferred cuisine. Italian ordered approximately 3 times less than American and Spanish and Vietnamese is lowest choice of cuisine in New York. The median cost of the order is around 14$ with min and max are 5$ and 35$ respectively. People ordered mostly on weekends as compared to weekdays with 6 times more orders placed. The median delivery time is 25 minutes while mean time is 24.16 minutes and first Quartile and third quartile of orders had delivery time of 20 minutes and 27.5 minutes respectively. customer id 52832 ordered the most. shack shack is the highest revenue generator followed by the meatball shop. By looking at the Heatmap, it is noticeable that the cost of order and food preparation time are highly correlated. The net revenue generated by the company is 701071614.75$.

New Section

Recommendations:

Company should encourage lowest revenue generating restaurants by giving them promotional offers of some percentage in return of every good performance so that they can improve their food quality and provide better services to the customers. as it will create biasness to only specific restaurants in the market and it will give chance to lowest rated restaurants to improve.

Company should focus on minimizing the delivery time by displaying the most nearby restaurants, it will definitely improve customer satisfaction,

Company should encourage customers by giving special discounts on feedback so that more and more customers give rating on the orders. It will provide more clarification if customer satisfied or not and we can have more insights on the services offered by the company.

Company should focus on promoting other cuisine like spanish and korean which are least ordered. Company can engage more and more people from cross cultural background living in the new york city and advertise by providing special deals on every order of different types of cuisines that were least ordered by people. It will generate more revenue from restaurants with all type of cuisines.