Project Foundations for Data Science: FoodHub Data Analysis

Marks: 60 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 report.

Let us start by importing the required libraries

```
pip install numpy pandas matplotlib seaborn
```

```
Requirement already satisfied: numpy in /usr/local/lib/python3.11/dist-packages (2.0.2)
Requirement already satisfied: pandas in /usr/local/lib/python3.11/dist-packages (2.2.2)
Requirement already satisfied: matplotlib in /usr/local/lib/python3.11/dist-packages (3.10.0)
Requirement already satisfied: seaborn in /usr/local/lib/python3.11/dist-packages (0.13.2)
Requirement already satisfied: python-dateutil>=2.8.2 in /usr/local/lib/python3.11/dist-packages (from pandas) (2.9.0.post0)
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.11/dist-packages (from pandas) (2025.2)
Requirement already satisfied: tzdata>=2022.7 in /usr/local/lib/python3.11/dist-packages (from pandas) (2025.2)
Requirement already satisfied: contourpy>=1.0.1 in /usr/local/lib/python3.11/dist-packages (from matplotlib) (1.3.2)
Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.11/dist-packages (from matplotlib) (0.12.1)
Requirement already satisfied: fonttools>=4.22.0 in /usr/local/lib/python3.11/dist-packages (from matplotlib) (4.59.0)
Requirement already satisfied: kiwisolver>=1.3.1 in /usr/local/lib/python3.11/dist-packages (from matplotlib) (1.4.8)
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.11/dist-packages (from matplotlib) (25.0)
Requirement already satisfied: pillow>=8 in /usr/local/lib/python3.11/dist-packages (from matplotlib) (11.3.0)
Requirement already satisfied: pyparsing>=2.3.1 in /usr/local/lib/python3.11/dist-packages (from matplotlib) (3.2.3)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.11/dist-packages (from python-dateutil>=2.8.2->pandas) (1.17.0)
```

```python
# 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
print("Libraries imported successfully!")
```

```
Libraries imported successfully!
```

```python
from google.colab import files

uploaded = files.upload()
```

Choose Files foodhub_order.csv

- **foodhub_order.csv**(text/csv) - 123933 bytes, last modified: 8/2/2025 - 100% done

```
Saving foodhub_order.csv to foodhub_order.csv
```

Understanding the structure of the data

```python
# Read the data
df = pd.read_csv('foodhub_order.csv')
# Returns the first 5 rows
df.head()
```

	order_id	customer_id	restaurant_name	cuisine_type	cost_of_the_order	day_of_the_week	rating	food_preparation_time	delivery
0	1477147	337525	Hangawi	Korean	30.75	Weekend	Not given	25	
1	1477685	358141	Blue Ribbon Sushi Izakaya	Japanese	12.08	Weekend	Not given	25	
2	1477070	66393	Cafe Habana	Mexican	12.23	Weekday	5	23	
3	1477334	106968	Blue Ribbon Fried Chicken	American	29.20	Weekend	3	25	
4	1478249	76942	Dirty Bird to Go	American	11.59	Weekday	4	25	

Next steps: Generate code with df View recommended plots New interactive sheet

Question 1: How many rows and columns are present in the data? [0.5 mark]

```python
# Check the shape of the dataset
rows, columns = df.shape
print(f"Number of rows: {rows}")
print(f"Number of columns: {columns}") # Fill in the blank
```

```
Number of rows: 1898
Number of columns: 9
```

Question 2: What are the datatypes of the different columns in the dataset? [0.5 mark]

```python
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. [1 Mark]

```
# Checking for missing values in the data
print(df.isnull().sum())

# Fill missing values in object-type columns with their mode
for col in df.select_dtypes(include='object').columns:
    df[col] = df[col].fillna(df[col].mode()[0])

# Verify no missing values remain
print(df.isnull().sum())
```

```
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
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? [2 marks]

```
# Check statistical summary for the dataset (all numerical columns)
print(df.describe())

# Specifically get min, mean, and max of food preparation time
min_time = df['food_preparation_time'].min()
mean_time = df['food_preparation_time'].mean()
max_time = df['food_preparation_time'].max()

print(f"Minimum food preparation time: {min_time} minutes")
print(f"Average food preparation time: {mean_time:.2f} minutes")
print(f"Maximum food preparation time: {max_time} minutes")
```

```
           order_id    customer_id  cost_of_the_order  food_preparation_time  \
count  1.898000e+03    1898.000000        1898.000000            1898.000000
mean   1.477496e+06  171168.478398          16.498851              27.371970
std    5.480497e+02  113698.139743           7.483812               4.632481
min    1.476547e+06    1311.000000           4.470000              20.000000
25%    1.477021e+06   77787.750000          12.080000              23.000000
50%    1.477496e+06  128600.000000          14.140000              27.000000
75%    1.477970e+06  270525.000000          22.297500              31.000000
max    1.478444e+06  405334.000000          35.410000              35.000000

       delivery_time
count    1898.000000
mean       24.161749
std         4.972637
min        15.000000
25%        20.000000
50%        25.000000
75%        28.000000
max        33.000000
Minimum food preparation time: 20 minutes
Average food preparation time: 27.37 minutes
Maximum food preparation time: 35 minutes
```

Question 5: How many orders are not rated? [1 mark]

```
print(df.columns)
```

```
Index(['order_id', 'customer_id', 'restaurant_name', 'cuisine_type',
       'cost_of_the_order', 'day_of_the_week', 'rating',
       'food_preparation_time', 'delivery_time'],
      dtype='object')
```

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

rating	count
Not given	736
5	588
4	386
3	188

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.) [9 marks]

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 the 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
```

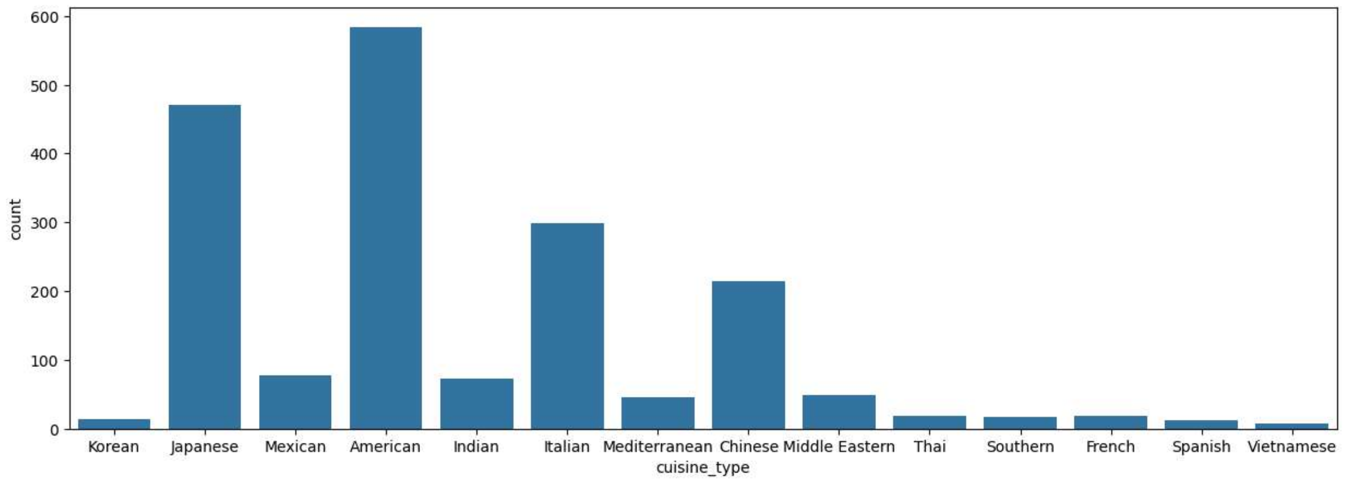
Cuisine type

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

```
14
```

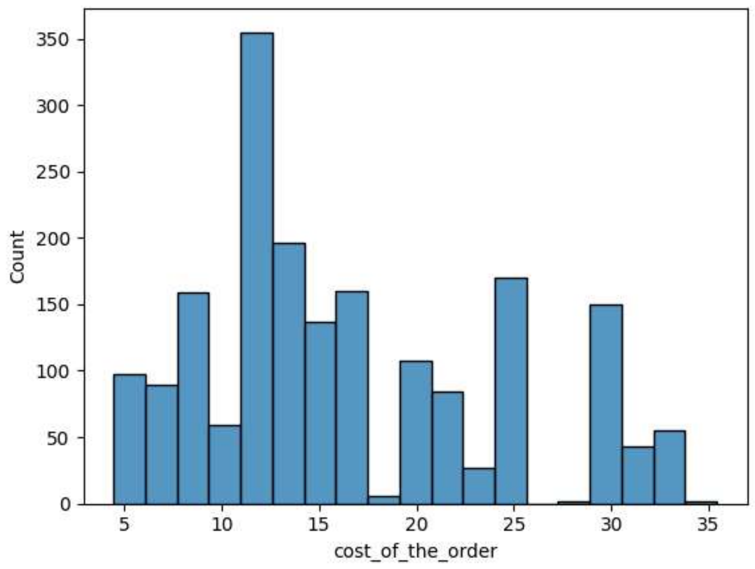
```
plt.figure(figsize = (15,5))
sns.countplot(data = df, x = 'cuisine_type') # Create a countplot for cuisine type
```

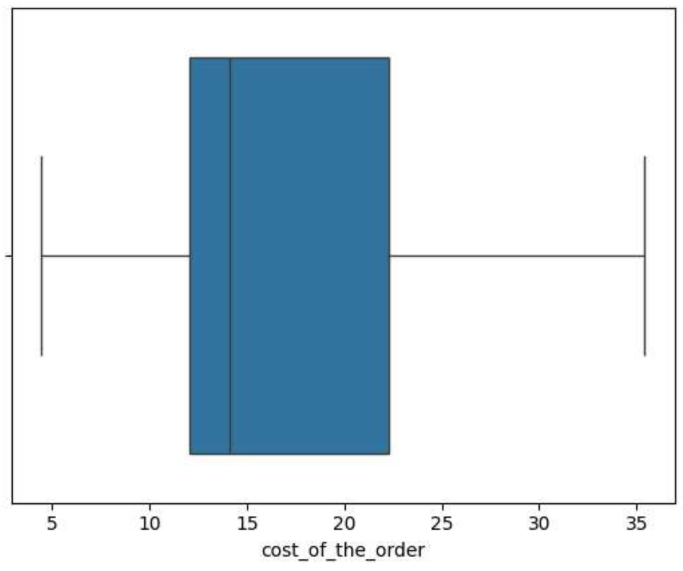
```
<Axes: xlabel='cuisine_type', ylabel='count'>
```



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()
```





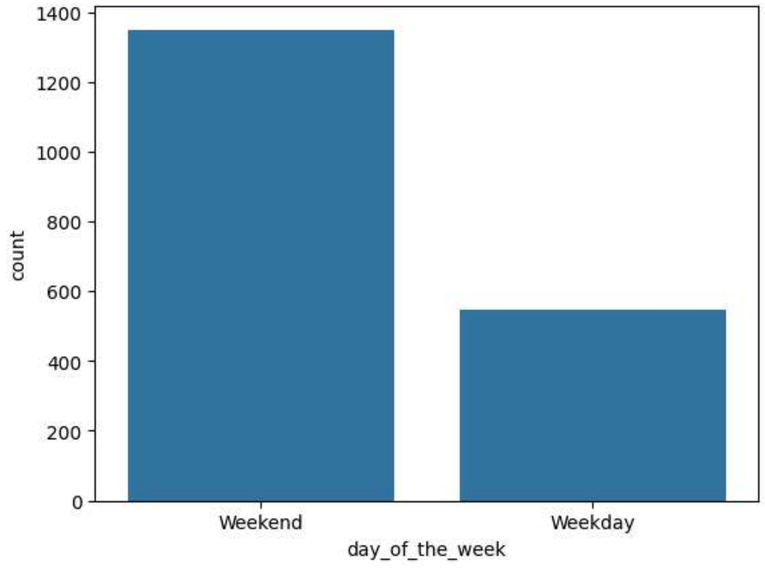
Day of the week

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

```
2
```

```
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'>
```



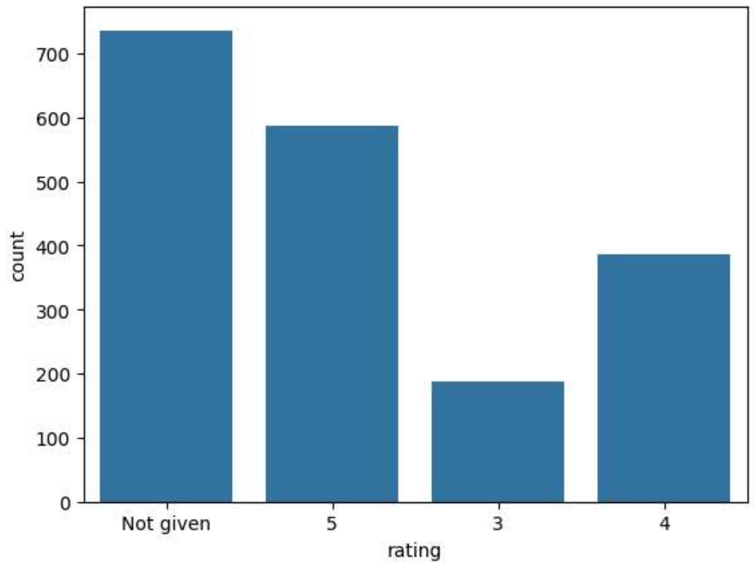
Rating

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

```
4
```

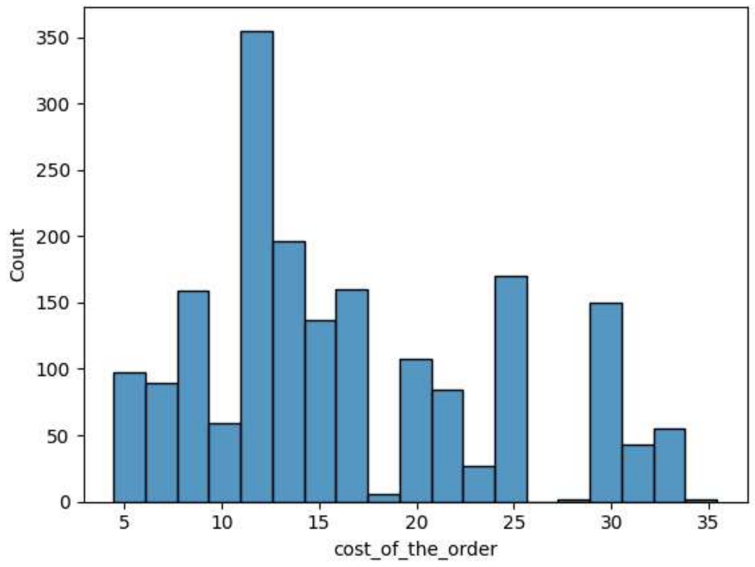
```
sns.countplot(data = df, x = 'rating') # Complete the code to plot bar graph for 'rating' column
```

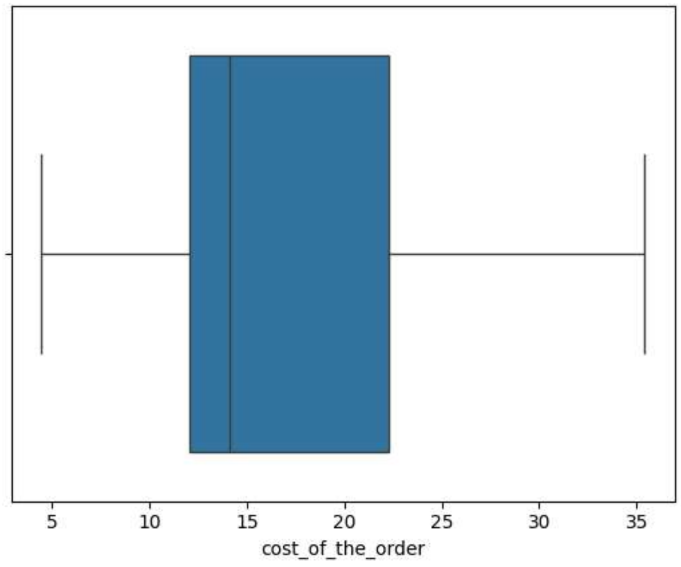
```
<Axes: xlabel='rating', ylabel='count'>
```



Food Preparation time

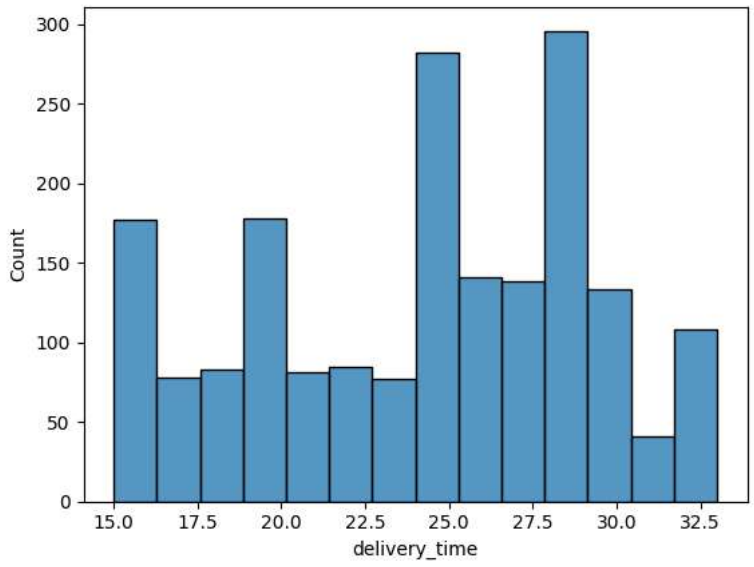
```
sns.histplot(data=df,x='cost_of_the_order') # Complete the code to plot the histogram for the cost of order
plt.show()
sns.boxplot(data=df,x='cost_of_the_order') # Complete the code to plot the boxplot for the cost of order
plt.show()
```

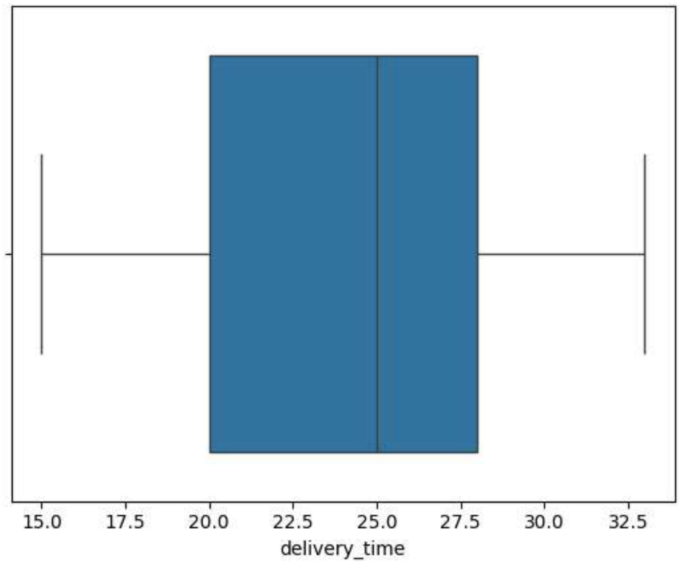




Delivery time

```
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? [1 mark]

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

restaurant_name	count
Shake Shack	219
The Meatball Shop	132
Blue Ribbon Sushi	119
Blue Ribbon Fried Chicken	96
Parm	68

dtype: int64

Question 8: Which is the most popular cuisine on weekends? [1 mark]

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

```
'American'
```

Question 9: What percentage of the orders cost more than 20 dollars? [2 marks]

```
# 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? [1 mark]

```
# 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. [1 mark]

```
# Get the counts of each customer_id
df['customer_id'].value_counts().head(3)  # Write the appropriate column name to get the 5 most frequent customers
```

customer_id	count
52832	13
47440	10
83287	9

dtype: int64

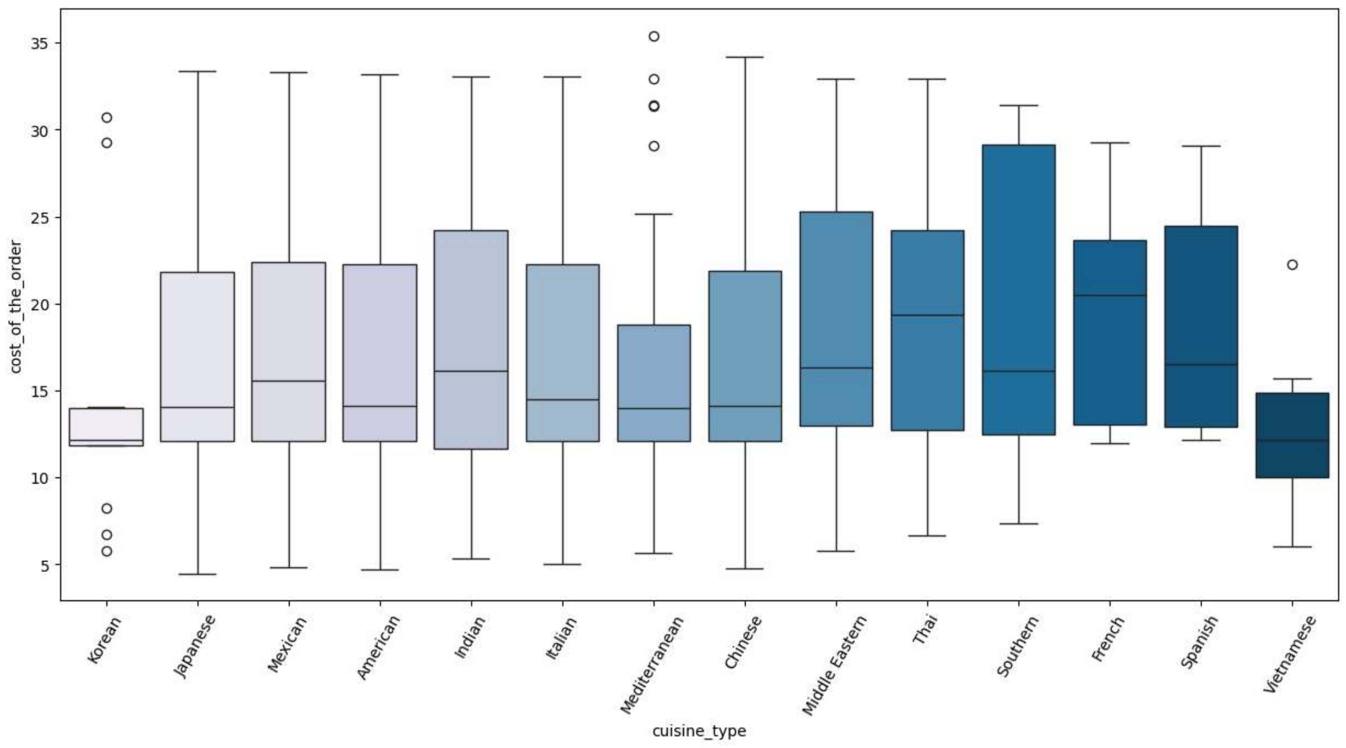
Multivariate Analysis

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) [10 marks]

Cuisine vs Cost of the order

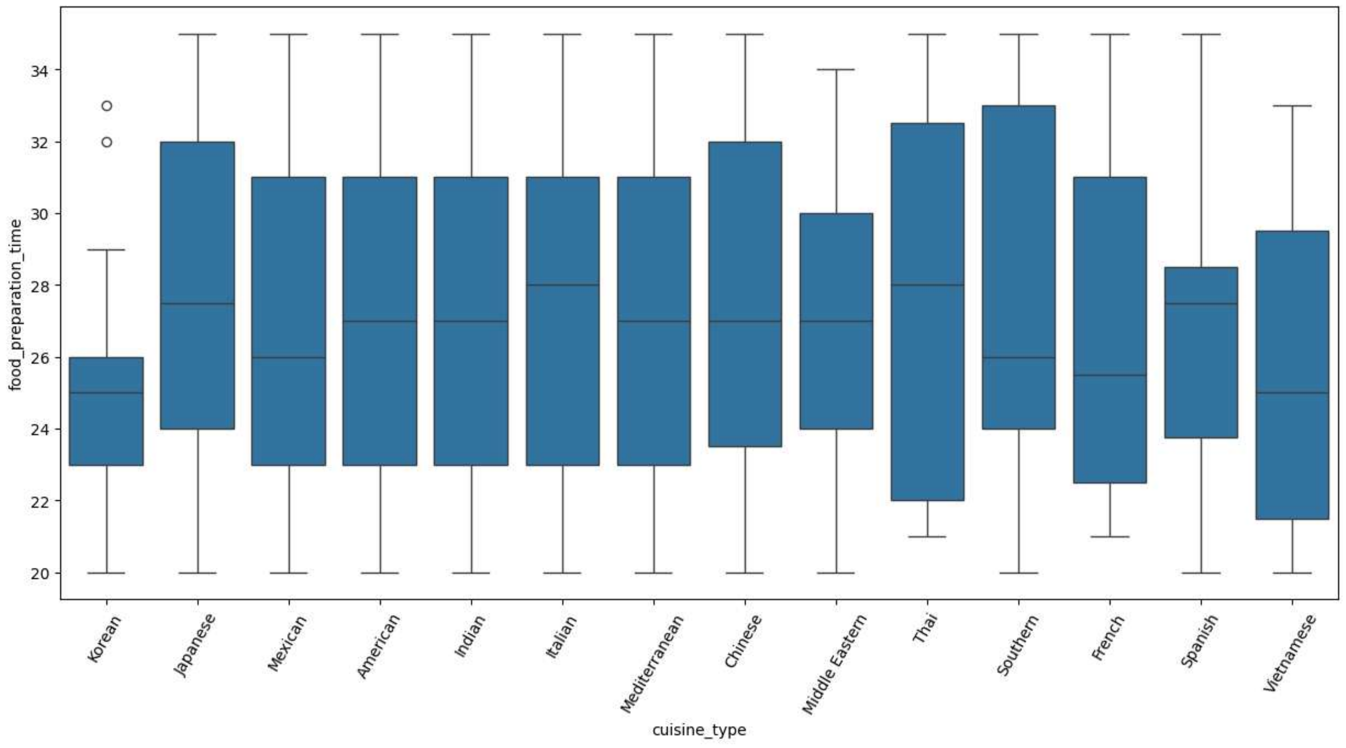
```
# Relationship between cost of the order and cuisine type
plt.figure(figsize=(15,7))
sns.boxplot(x = "cuisine_type", y = "cost_of_the_order", data = df, palette = 'PuBu')
plt.xticks(rotation = 60)
plt.show()
```

```
/tmp/ipython-input-2537546578.py: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 `le

  sns.boxplot(x = "cuisine_type", y = "cost_of_the_order", data = df, palette = 'PuBu')
```



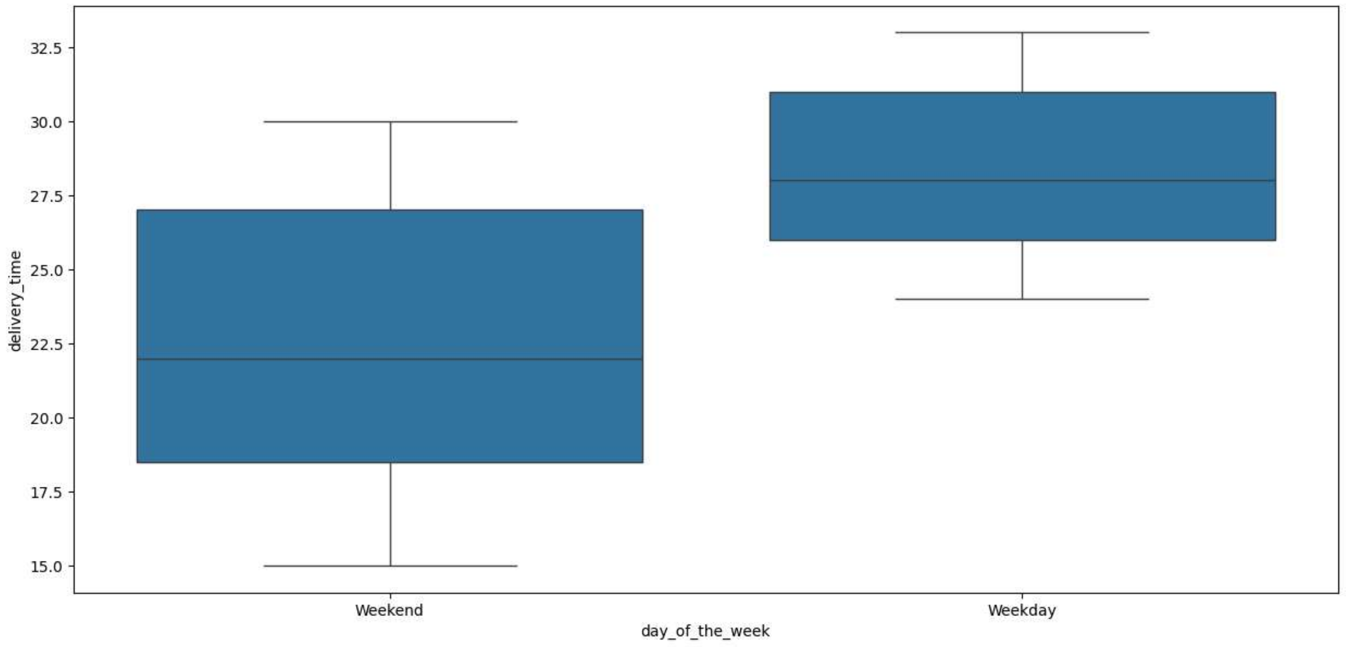
Cuisine vs Food Preparation time

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



Day of the Week vs Delivery time

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



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	cost_of_the_order
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

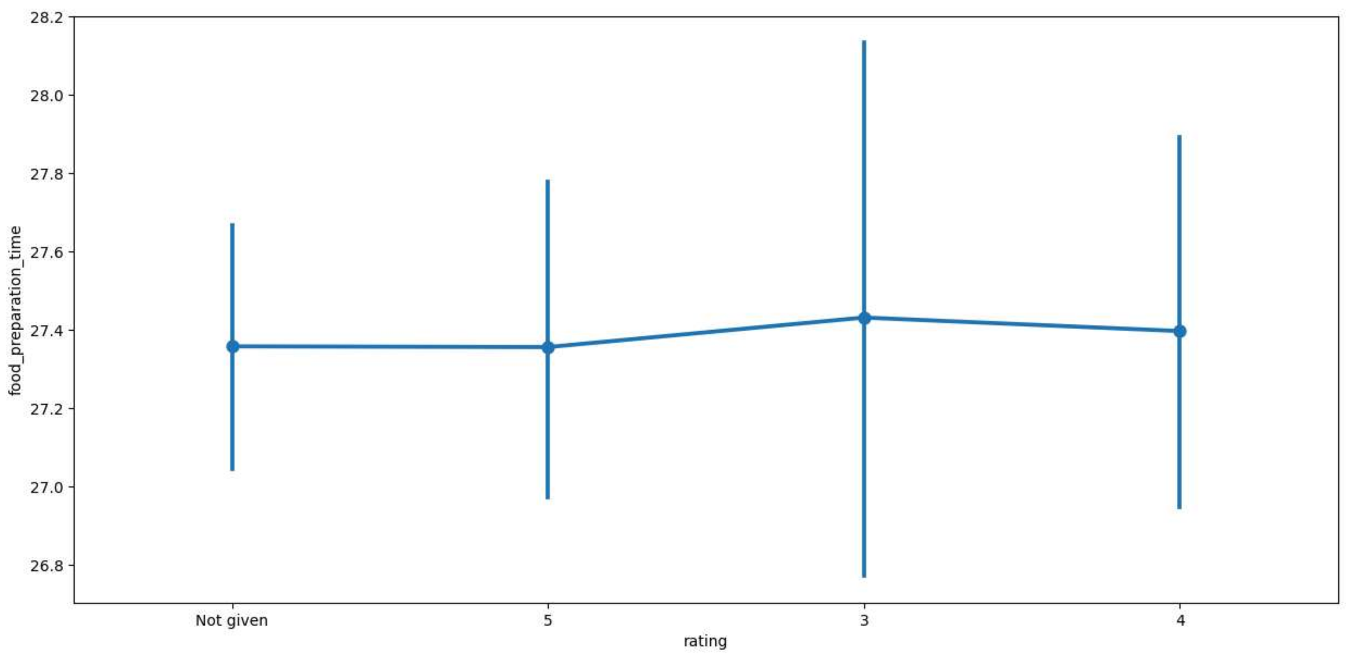
dtype: float64

Rating vs Delivery time

[ ] ↳ 1 cell hidden

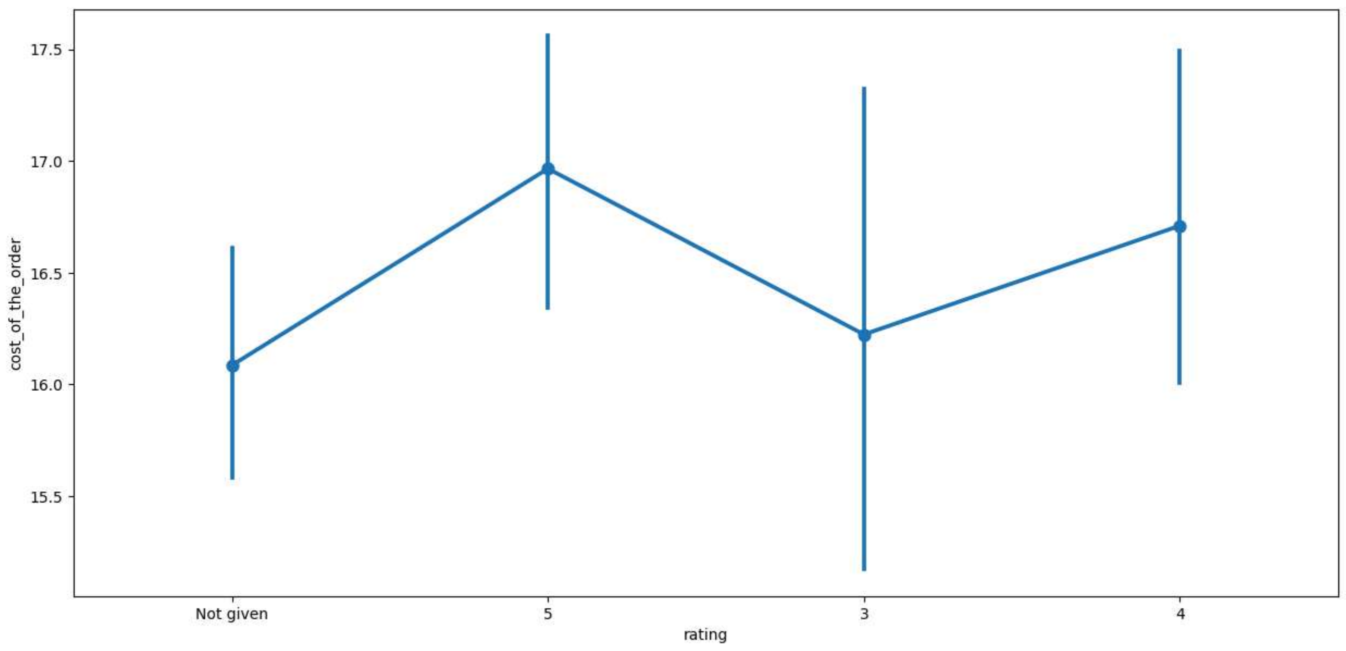
Rating vs Food preparation time

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



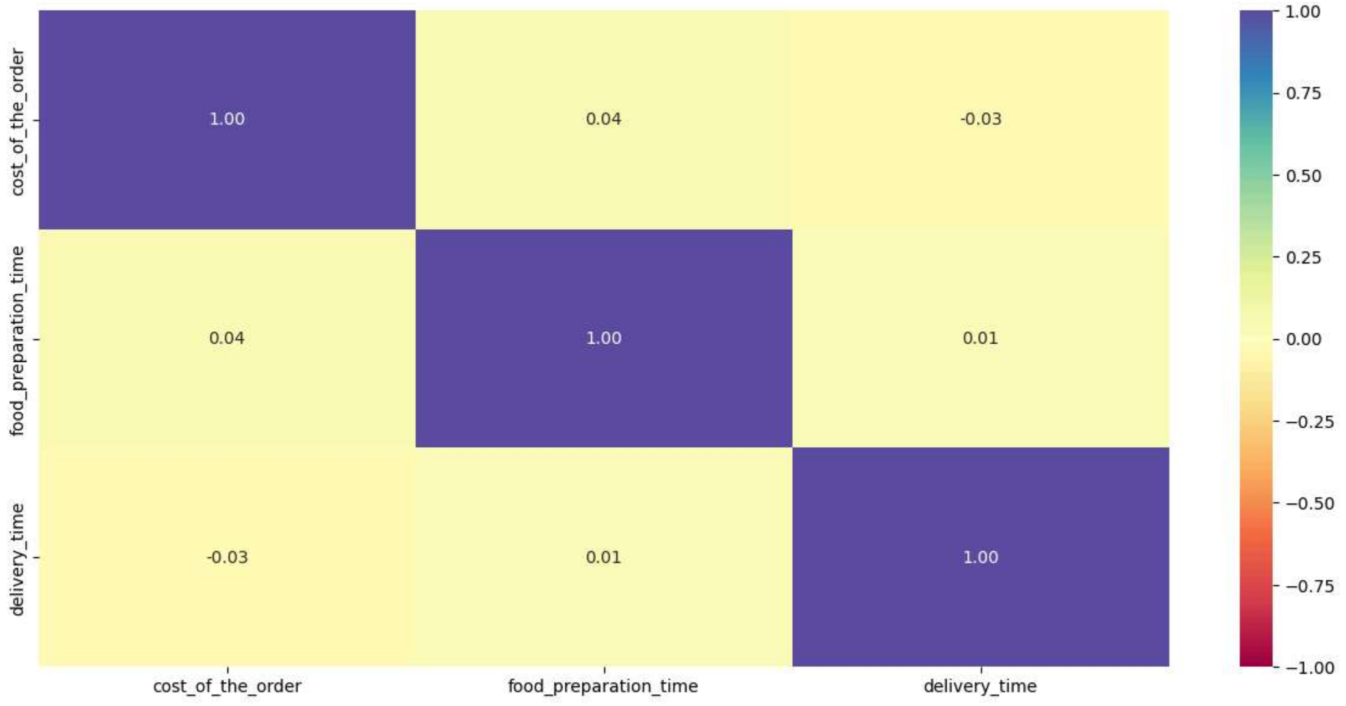
Rating vs Cost of the order

```
# Relationship between rating and cost of the order
plt.figure(figsize=(15, 7))
sns.pointplot(x='rating', y='cost_of_the_order', data=df)   # Complete the code to visualize the relationship between rating and cost of
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. [3 marks]

```
# 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()
```

	restaurant_name	rating
0	Shake Shack	133
1	The Meatball Shop	84
2	Blue Ribbon Sushi	73
3	Blue Ribbon Fried Chicken	64
4	RedFarm Broadway	41

Next steps: Generate code with df_rating_count View recommended plots New interactive sheet

```
# 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

# 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
```

	restaurant_name	rating
0	The Meatball Shop	4.511905
1	Blue Ribbon Fried Chicken	4.328125
2	Shake Shack	4.278195
3	Blue Ribbon Sushi	4.219178

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. [3 marks]

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

	order_id	customer_id	restaurant_name	cuisine_type	cost_of_the_order	day_of_the_week	rating	food_preparation_time	delivery
0	1477147	337525	Hangawi	Korean	30.75	Weekend	Not given	25	
1	1477685	358141	Blue Ribbon Sushi Izakaya	Japanese	12.08	Weekend	Not given	25	
2	1477070	66393	Cafe Habana	Mexican	12.23	Weekday	5	23	
3	1477334	106968	Blue Ribbon Fried Chicken	American	29.20	Weekend	3	25	
4	1478249	76942	Dirty Bird to Go	American	11.59	Weekday	4	25	

Next steps: Generate code with df View recommended plots New interactive sheet

```
# 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 6166.3 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.) [2 marks]

```
# 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 referen
```

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? [2 marks]

```
# 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
```