```
import pandas as pd
import numpy as np
```

```
df=pd.read_csv('/content/amazon_delivery.csv')
```

```
df.head()
```

	Order_ID	Agent_Age	Agent_Rating	Store_Latitude	Store_Longitude	Drop_Latitude	Drop_Longitude	Order_Date	Order_Time
0	ialx566343618	37	4.9	22.745049	75.892471	22.765049	75.912471	2022-03-19	11:30:00
1	akqg208421122	34	4.5	12.913041	77.683237	13.043041	77.813237	2022-03-25	19:45:00
2	njpu434582536	23	4.4	12.914264	77.678400	12.924264	77.688400	2022-03-19	08:30:00
3	rjto796129700	38	4.7	11.003669	76.976494	11.053669	77.026494	2022-04-05	18:00:00
4	zguw716275638	32	4.6	12.972793	80.249982	13.012793	80.289982	2022-03-26	13:30:00

Next steps: Generate code with df | New interactive sheet

```
df["Traffic"].unique()
```

```
array([1, 2, 3, 4, 5, 0])
```

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 38401 entries, 0 to 38400
Data columns (total 16 columns):
 #   Column           Non-Null Count  Dtype  
---  ------           --------------  -----  
 0   Order_ID         38401 non-null  object 
 1   Agent_Age        38401 non-null  int64  
 2   Agent_Rating     38351 non-null  float64
 3   Store_Latitude   38401 non-null  float64
 4   Store_Longitude  38400 non-null  float64
 5   Drop_Latitude    38400 non-null  float64
 6   Drop_Longitude   38400 non-null  float64
 7   Order_Date       38400 non-null  object 
 8   Order_Time       38400 non-null  object 
 9   Pickup_Time      38400 non-null  object 
 10  Weather          38319 non-null  object 
 11  Traffic          38400 non-null  object 
 12  Vehicle          38400 non-null  object 
 13  Area             38400 non-null  object 
 14  Delivery_Time    38400 non-null  float64
 15  Category         38400 non-null  object 
dtypes: float64(6), int64(1), object(9)
memory usage: 4.7+ MB
```

```
df.isnull().sum()
```

	0
Order_ID	0
Agent_Age	0
Agent_Rating	50
Store_Latitude	0
Store_Longitude	1
Drop_Latitude	1
Drop_Longitude	1
Order_Date	1
Order_Time	1
Pickup_Time	1
Weather	82
Traffic	1
Vehicle	1
Area	1
Delivery_Time	1
Category	1

dtype: int64

```
df["Agent_Rating"] = df["Agent_Rating"].ffill()
df["Weather"] = df["Weather"].ffill()
```

```
df.isnull().sum().sum()
```

```
np.int64(11)
```

```
df = df.fillna(df.mean)
```

```
df.describe()
```

	Agent_Age	Agent_Rating	Store_Latitude
count	38401.000000	38401.000000	38401.000000
mean	29.578553	4.632918	17.211763
std	5.822843	0.336169	7.779033
min	15.000000	1.000000	-30.902872
25%	25.000000	4.500000	12.933298
50%	30.000000	4.700000	18.554382
75%	35.000000	4.900000	22.732225
max	50.000000	6.000000	30.914057

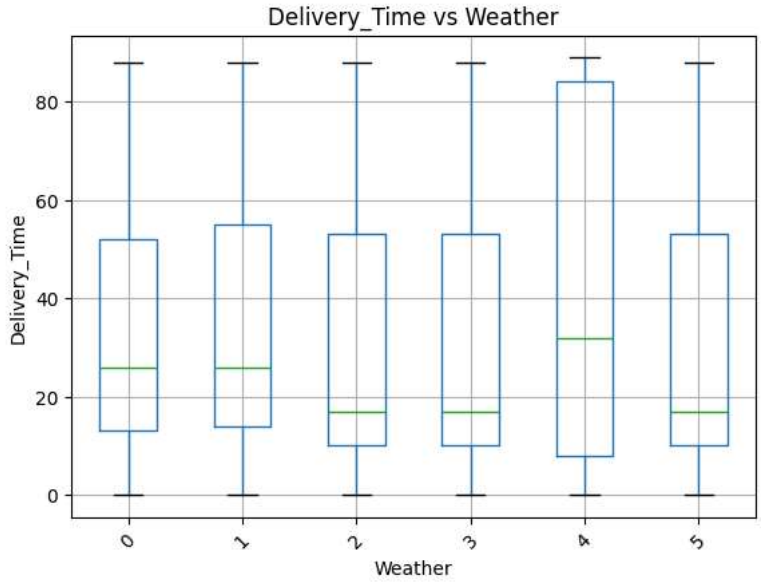
```
# Haversine formula
import math

def haversine(lat1, lon1, lat2, lon2):
    R = 6371.0  # Earth radius in km
    lat1, lon1, lat2, lon2 = map(math.radians, [lat1, lon1, lat2, lon2])
    dlat = lat2 - lat1
    dlon = lon2 - lon1
    a = math.sin(dlat/2)**2 + math.cos(lat1)*math.cos(lat2)*math.sin(dlon/2)**2
    c = 2 * math.atan2(math.sqrt(a), math.sqrt(1-a))
    return R * c

# Add distance column
# Drop rows with non-numeric values in latitude or longitude columns
df_cleaned = df.dropna(subset=["Store_Latitude", "Store_Longitude", "Drop_Latitude", "Drop_Longitude"])
```

```
df["Known_Distance_km"] = df_cleaned.apply(
    lambda row: haversine(row["Store_Latitude"], row["Store_Longitude"],
                          row["Drop_Latitude"], row["Drop_Longitude"]), axis=1
)

# See first few rows
print(df.head())

# Optionally, save back to CSV
df.to_csv("amazon_delivery_with_distance.csv", index=False)
```

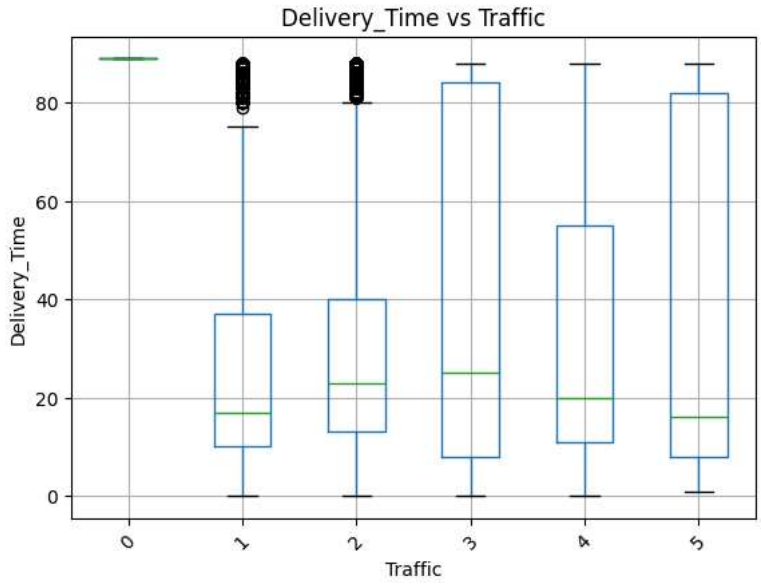
```
        Order_ID  Agent_Age  Agent_Rating  Store_Latitude  Store_Longitude  \
0  ialx566343618         37           4.9       22.745049              176
1  akqg208421122         34           4.5       12.913041              275
2  njpu434582536         23           4.4       12.914264              274
3  rjto796129700         38           4.7       11.003669              234
4  zguw716275638         32           4.6       12.972793              335

   Drop_Latitude  Drop_Longitude Order_Date  Order_Time  Pickup_Time  Weather  \
0           2688            1522 2022-03-19       11.50    11.750000        4
1            821            2827 2022-03-25       19.75    19.833333        3
2            637            2711 2022-03-19        8.50     8.750000        2
3            284            2327 2022-04-05       18.00    18.166667        4
4            752            3439 2022-03-26       13.50    13.750000        0

   Traffic  Vehicle  Area  Delivery_Time  Category  Known_Distance_km
0        1        2     4              7         3        9091.353322
1        2        3     1             20         5        9614.565371
2        3        2     4             10        15       11338.227111
3        4        2     1              2         4       10597.151727
4        1        3     1             16        16       13164.608662
```

```
import matplotlib.pyplot as plt

# Define the categorical columns to compare with Delivery_Time
categorical_cols = ["Weather", "Traffic", "Vehicle", "Area", "Category"]
numeric_col = "Known_Distance_km"
target = "Delivery_Time"

# Plot categorical variables vs Delivery_Time (boxplots)
for col in categorical_cols:
    plt.figure(figsize=(8, 5))
    df.boxplot(column=target, by=col)
    plt.title(f"{target} vs {col}")
    plt.suptitle("")  # Remove default title
    plt.ylabel(target)
    plt.xticks(rotation=45)
    plt.show()

# Plot Known_Distance_km vs Delivery_Time (scatter)
plt.figure(figsize=(8, 5))
plt.scatter(df[numeric_col], df[target], alpha=0.6)
plt.title(f"{target} vs {numeric_col}")
plt.xlabel(numeric_col)
plt.ylabel(target)
plt.show()
```

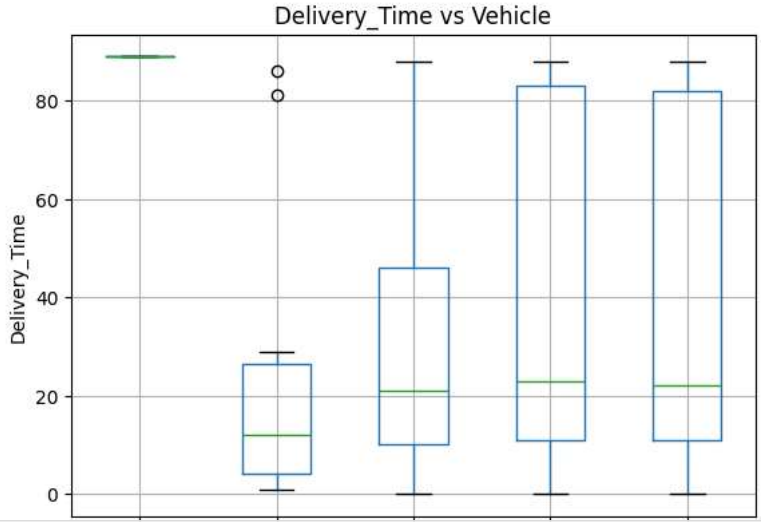
```
<Figure size 800x500 with 0 Axes>
```




```
<Figure size 800x500 with 0 Axes>
```




```
<Figure size 800x500 with 0 Axes>
```




```
# Calculate correlation coefficient between Known_Distance_km and Delivery_Time
correlation = df["Known_Distance_km"].corr(df["Delivery_Time"])
correlation
correlation = df["Known_Distance_km"].corr(df["Delivery_Time"])
```