**Uber Data Analysis Project Documentation**

The goal of this project is to use SQL and Python to analyse and visualize Uber trip data. In order to obtain deeper insights into the data, the analysis involves data cleansing, descriptive statistics, time series analysis, geospatial analysis, trip duration analysis, revenue analysis, customer segmentation, and further studies. The tools used to make the visualizations are matplotlib, pandas, and seaborn.

**Step 1: Data Import**

We start by importing the CSV data into a SQL database and creating a table named `uber\_data`. Below is the table schema:

**Step 2: Descriptive Statistics**

We generate summary statistics to understand the central tendency, dispersion, and distribution of numerical columns.

SQL Query:

SELECT

AVG(passenger\_count) AS avg\_passenger\_count,

AVG(trip\_distance) AS avg\_trip\_distance,

AVG(fare\_amount) AS avg\_fare\_amount,

AVG(tip\_amount) AS avg\_tip\_amount,

AVG(total\_amount) AS avg\_total\_amount,

STDDEV(passenger\_count) AS stddev\_passenger\_count,

STDDEV(trip\_distance) AS stddev\_trip\_distance,

STDDEV(fare\_amount) AS stddev\_fare\_amount,

STDDEV(tip\_amount) AS stddev\_tip\_amount,

STDDEV(total\_amount) AS stddev\_total\_amount,

MIN(passenger\_count) AS min\_passenger\_count,

MAX(passenger\_count) AS max\_passenger\_count,

MIN(trip\_distance) AS min\_trip\_distance,

MAX(trip\_distance) AS max\_trip\_distance,

MIN(fare\_amount) AS min\_fare\_amount,

MAX(fare\_amount) AS max\_fare\_amount,

MIN(tip\_amount) AS min\_tip\_amount,

MAX(tip\_amount) AS max\_tip\_amount,

MIN(total\_amount) AS min\_total\_amount,

MAX(total\_amount) AS max\_total\_amount

FROM uber\_data;

**Explanation:** This query calculates the average, standard deviation, minimum, and maximum values for key columns like `passenger\_count`, `trip\_distance`, `fare\_amount`, `tip\_amount`, and `total\_amount`.

**Step 3: Time Series Analysis**

We analyze trends in pickups and dropoffs over time.

Number of Pickups per Day

SQL Query:

SELECT DATE(tpep\_pickup\_datetime) AS pickup\_date, COUNT(\*) AS pickup\_count

FROM uber\_data

GROUP BY pickup\_date

ORDER BY pickup\_date;

**Explanation:** This query groups the data by the pickup date and counts the number of pickups for each date.

Python Code:

import pandas as pd

import seaborn as sns

import matplotlib.pyplot as plt

**Load the data**

file\_path = '/mnt/data/uber\_data.csv'

data = pd.read\_csv(file\_path)

**Convert to datetime**

data['tpep\_pickup\_datetime'] = pd.to\_datetime(data['tpep\_pickup\_datetime'])

data['tpep\_dropoff\_datetime'] = pd.to\_datetime(data['tpep\_dropoff\_datetime'])

**Plot the number of pickups per day**

pickup\_counts\_per\_day = data['tpep\_pickup\_datetime'].dt.date.value\_counts().sort\_index()

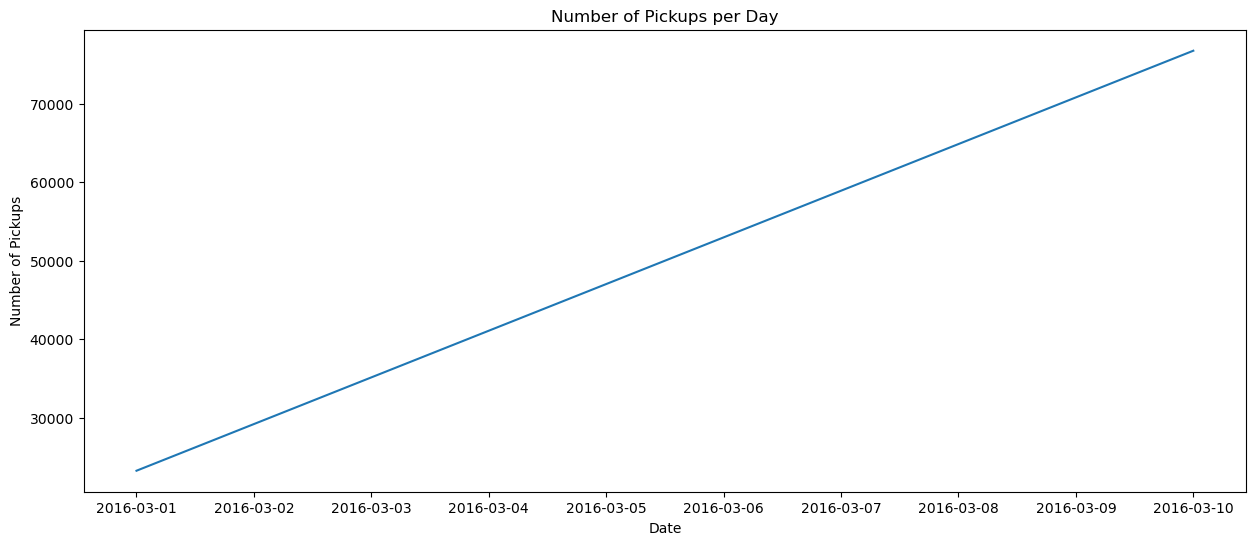
pickup\_counts\_per\_day.plot(kind='line', figsize=(15, 6), title='Number of Pickups per Day')

plt.xlabel('Date')

plt.ylabel('Number of Pickups')

plt.show()

**Visualization:**

****

**Explanation:** This code plots the number of pickups per day, providing a time series analysis of the pickups over the period covered by the data.

**Step 5: Trip Duration Analysis**

**We calculate and analyze trip durations.**

SQL Query:

SELECT

TIMESTAMPDIFF(MINUTE, tpep\_pickup\_datetime, tpep\_dropoff\_datetime) AS trip\_duration\_minutes

FROM uber\_data;

**Explanation**: This query calculates the duration of each trip in minutes.

Python Code:

data['trip\_duration'] = (data['tpep\_dropoff\_datetime'] - data['tpep\_pickup\_datetime']).dt.total\_seconds() / 60

**Plot the distribution of trip duration**

plt.figure(figsize=(12, 6))

sns.histplot(data['trip\_duration'], bins=100, kde=True)

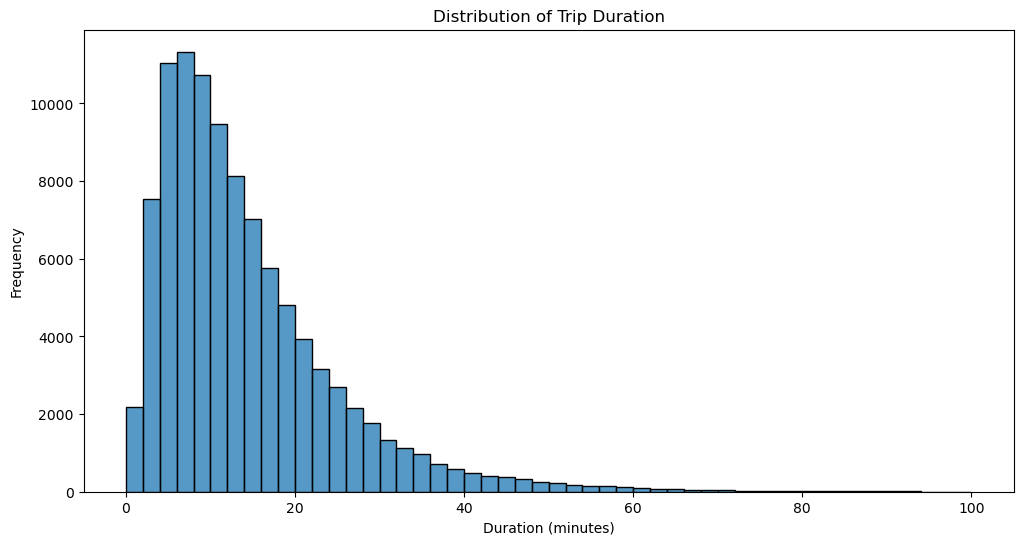
plt.title('Distribution of Trip Duration')

plt.xlabel('Duration (minutes)')

plt.ylabel('Frequency')

plt.show()

**Visualization:**



**Explanation:** This code calculates trip durations and plots their distribution, showing how long trips typically take.

**Step 6: Revenue Analysis**

We analyze fare amounts and other cost components.

**Total Revenue**

SQL Query:

SELECT SUM(total\_amount) AS total\_revenue FROM uber\_data;

**Explanation:** This query calculates the total revenue generated from all trips.

**Average Revenue per Trip**

SQL Query:

SELECT AVG(total\_amount) AS avg\_revenue\_per\_trip FROM uber\_data;

**Explanation:** This query calculates the average revenue per trip.

**Distribution of Fare Amounts**

SQL Query:

SELECT

fare\_amount,

COUNT(\*) AS count

FROM uber\_data

GROUP BY fare\_amount

ORDER BY fare\_amount;

**Explanation:** This query groups the data by fare amount and counts the number of occurrences of each fare amount.

Python Code:

**Plot the distribution of fare amounts**

plt.figure(figsize=(12, 6))

sns.histplot(data['fare\_amount'], bins=100, kde=True)

plt.title('Distribution of Fare Amounts')

plt.xlabel('Fare Amount ($)')

plt.ylabel('Frequency')

plt.show()

**Calculate total revenue**

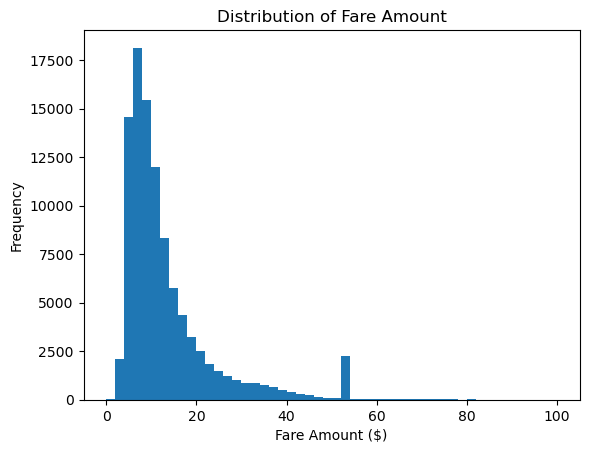
total\_revenue = data['total\_amount'].sum()

print(f'Total Revenue: ${total\_revenue:.2f}')

**Average revenue per trip**

average\_revenue\_per\_trip = data['total\_amount'].mean()

print(f'Average Revenue per Trip: ${average\_revenue\_per\_trip:.2f}')

**Visualization:** 

**Explanation:** This code plots the distribution of fare amounts, showing how much passengers typically pay for trips.

**Step 7: Customer Segmentation**

We segment customers based on the number of rides taken.

SQL Query:

SELECT passenger\_count, COUNT(\*) AS ride\_count

FROM uber\_data

GROUP BY passenger\_count

ORDER BY passenger\_count;

**Explanation:** This query groups the data by passenger count and counts the number of rides for each passenger count.

Python Code:

**Calculate the number of rides per passenger count**

rides\_per\_passenger = data['passenger\_count'].value\_counts()

**Plot the number of rides per passenger count**

plt.figure(figsize=(12, 6))

sns.barplot(x=rides\_per\_passenger.index, y=rides\_per\_passenger.values)

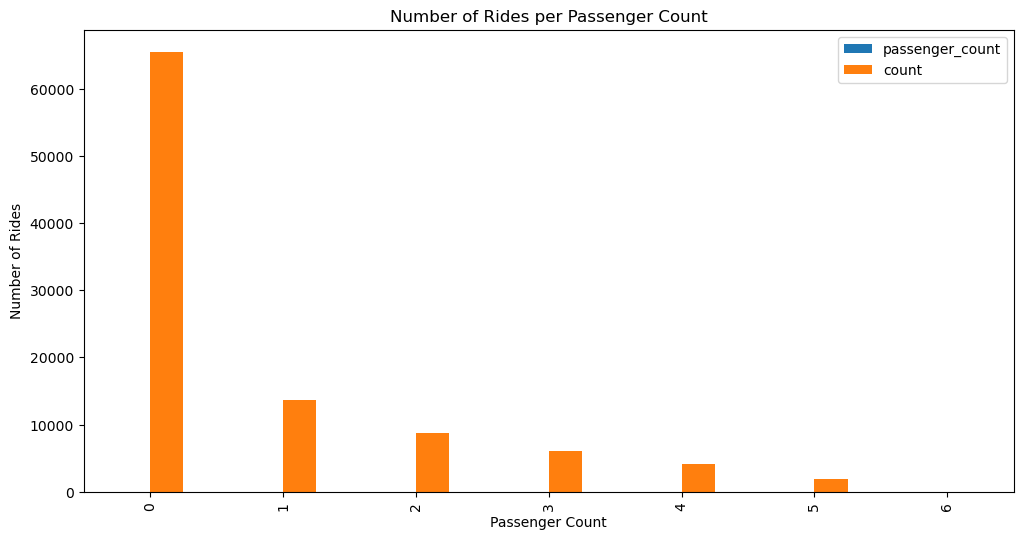
plt.title('Number of Rides per Passenger Count')

plt.xlabel('Passenger Count')

plt.ylabel('Number of Rides')

plt.show()

**Visualization:**



**Explanation:** This code plots the number of rides for each passenger count, showing how many rides typically have how many passengers.

**Step 8: Additional Analysis**

We explore more insights from the data with the following analyses:

**Tip Amount Analysis**

Python Code:

**Plot the distribution of tip amounts**

plt.figure(figsize=(12, 6))

sns.histplot(data['tip\_amount'], bins=100, kde=True)

plt.title('Distribution of Tip Amounts')

plt.xlabel('Tip Amount ($)')

plt.ylabel('Frequency')

plt.show()

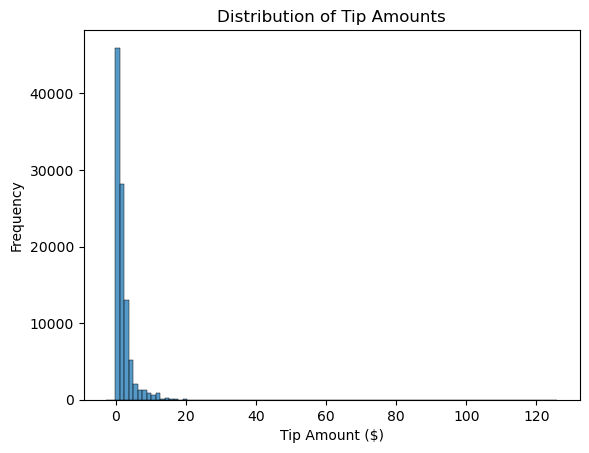
**Calculate the average tip amount**

average\_tip\_amount = data['tip\_amount'].mean()

print(f'Average Tip Amount: ${average\_tip\_amount:.2f}')

SQL Query:

SELECT AVG(tip\_amount) AS avg\_tip\_amount FROM uber\_data;

**Visualization: **

**Explanation:** This analysis examines tipping behavior by plotting the distribution of tip amounts and calculating the average tip.

**Hourly Revenue Analysis**

Python Code:

Extract the hour of pickup

data['pickup\_hour'] = data['tpep\_pickup\_datetime'].dt.hour

Calculate average total amount by hour

hourly\_revenue = data.groupby('pickup\_hour')['total\_amount'].mean().reset\_index()

Plot hourly revenue

plt.figure(figsize=(12, 6))

sns.barplot(x='pickup\_hour', y='total\_amount', data=hourly\_revenue)

plt.title('Average Revenue by Hour of Day')

plt.xlabel('Hour of Day')

plt.ylabel('Average Total Amount ($)')

plt.show()

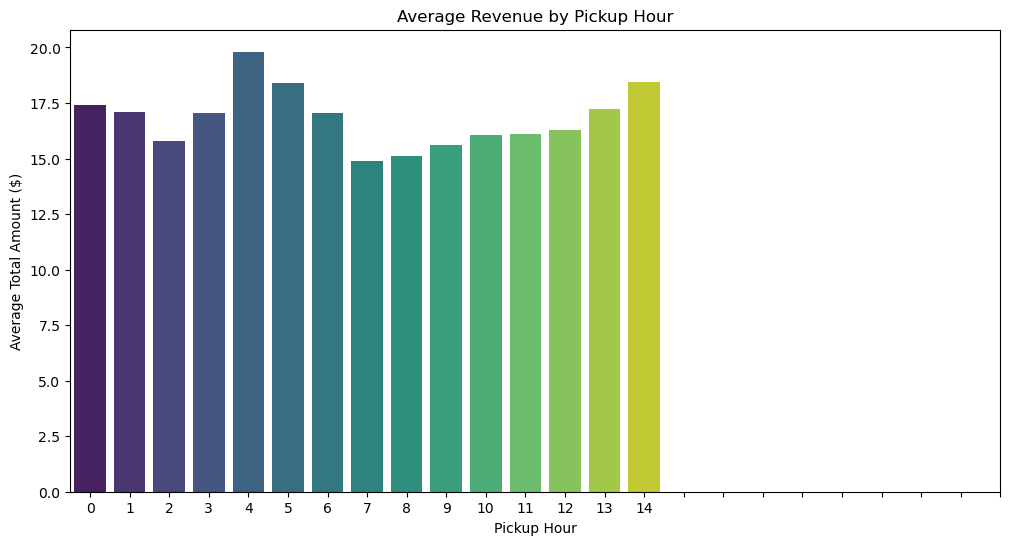
SQL Query:

SELECT HOUR(tpep\_pickup\_datetime) AS pickup\_hour, AVG(total\_amount) AS avg\_total\_amount

FROM uber\_data

GROUP BY pickup\_hour;

**Visualization:**



Explanation: This analysis examines revenue patterns by hour, identifying peak revenue times.

**Trip Distance vs. Trip Duration**

Python Code:

Scatter plot of trip distance vs. trip duration

plt.figure(figsize=(12, 6))

sns.scatterplot(x='trip\_distance', y='trip\_duration', data=data, alpha=0.3)

plt.title('Trip Distance vs. Trip Duration')

plt.xlabel('Trip Distance (miles)')

plt.ylabel('Trip Duration (minutes)')

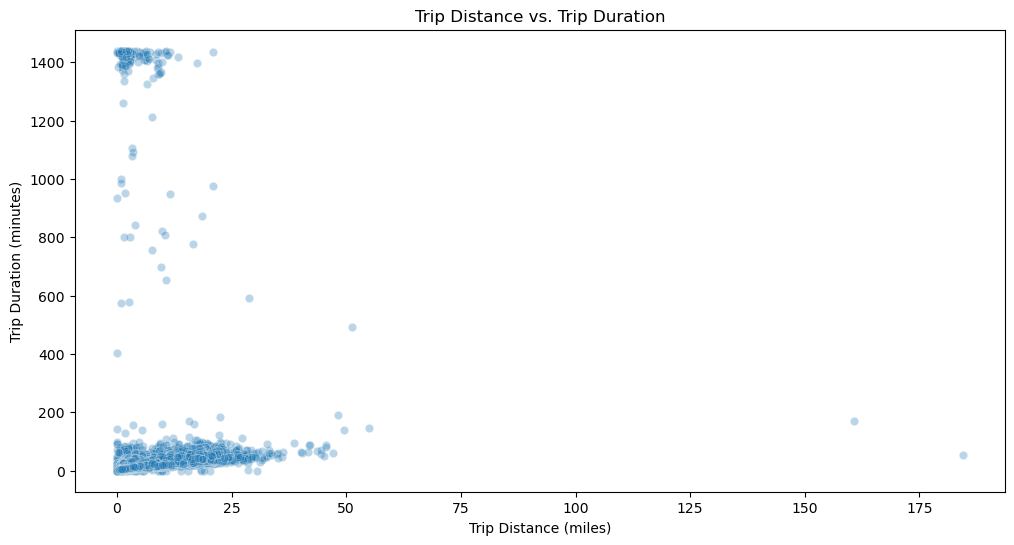
plt.show()

SQL Query:

SELECT trip\_distance, TIMESTAMPDIFF(MINUTE, tpep\_pickup\_datetime, tpep\_dropoff\_datetime) AS trip\_duration\_minutes

FROM uber\_data;

**Visualization:**



Explanation: This analysis explores the relationship between trip distance and duration using a scatter plot.

**Trips by Day of the Week**

Python Code:

Extract the day of the week

data['day\_of\_week'] = data['tpep\_pickup\_datetime'].dt.dayofweek

Calculate the number of trips per day of the week

trips\_by\_day\_of\_week = data['day\_of\_week'].value\_counts().sort\_index()

Plot the number of trips per day of the week

plt.figure(figsize=(12, 6))

sns.barplot(x=trips\_by\_day\_of\_week.index, y=trips\_by\_day\_of\_week.values)

plt.title('Number of Trips by Day of the Week')

plt.xlabel('Day of the Week')

plt.ylabel('Number of Trips')

plt.xticks(ticks=range(7), labels=['Monday', 'Tuesday', 'Wednesday', 'Thursday', 'Friday', 'Saturday', 'Sunday'])

plt.show()

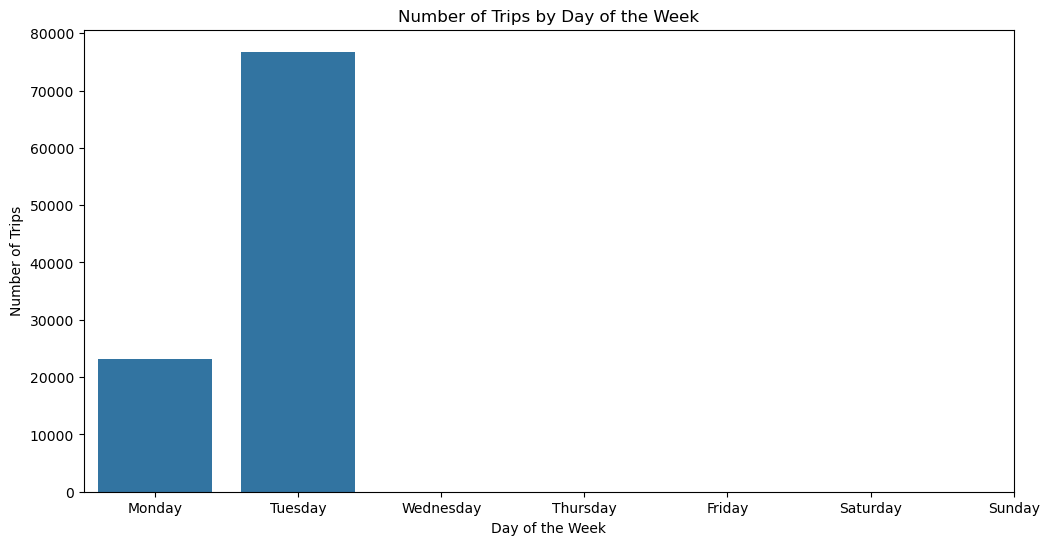
SQL Query:

SELECT DAYOFWEEK(tpep\_pickup\_datetime) AS day\_of\_week, COUNT(\*) AS trip\_count

FROM uber\_data

GROUP BY day\_of\_week;

Visualization:



Explanation: This analysis examines trip patterns by day of the week, identifying the busiest days.

**Trip Speed Analysis**

Python Code:

python

Calculate trip speed (miles per hour)

data['trip\_duration\_hours'] = data['trip\_duration'] / 60

data['trip\_speed'] = data['trip\_distance'] / data['trip\_duration\_hours']

Plot the distribution of trip speeds

plt.figure(figsize=(12, 6))

sns.histplot(data['trip\_speed'].dropna(), bins=100, kde=True)

plt.title('Distribution of Trip Speeds')

plt.xlabel('Speed (miles per hour)')

plt.ylabel('Frequency')

plt.show()

Calculate the average trip speed

average\_trip\_speed = data['trip\_speed'].mean()

print(f'Average Trip Speed: {average\_trip\_speed:.2f} mph')

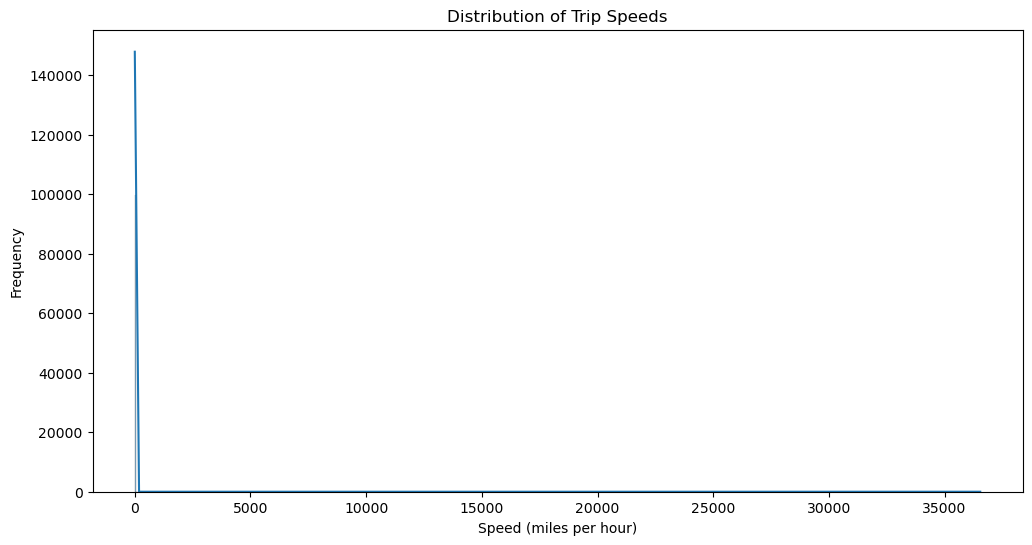
SQL Query:

SELECT

trip\_distance / (TIMESTAMPDIFF(MINUTE, tpep\_pickup\_datetime, tpep\_dropoff\_datetime) / 60.0) AS trip\_speed

FROM uber\_data;

Visualization:



**Explanation:** This analysis calculates and visualizes trip speeds, providing insights into the average speed of trips.

**Conclusion**

This thorough examination of the Uber trip data offers insightful information about a number of trip-related topics, such as passenger behavior, revenue trends, trip lengths, and the geographic distribution of pickups and drops. A complete examination of the data is ensured by the combination of Python visuals and SQL queries.   
  
  
  
The goal of this project is to use SQL and Python to analyze and visualize Uber trip data. In order to obtain deeper insights into the data, the analysis involves data cleansing, descriptive statistics, time series analysis, geospatial analysis, trip duration analysis, revenue analysis, customer segmentation, and further studies. The tools used to make the visualizations are matplotlib, pandas, and seaborn.