# INTRODUCTION TO BIG DATA PROJECT ASSIGNMENT REPORT

Rwagapfizi Igor, 27329

Group E, INSY 8413

# Table of Contents

# Introduction

This assignment project analyzes the Uber Fares Dataset to uncover insights into fare patterns, ride durations, and operational metrics. The goal is to develop an interactive Power BI dashboard and present findings through a structured analytical report.

**Objectives**:

- Perform Exploratory Data Analysis (EDA) to understand dataset structure and quality.
- Conduct feature engineering to extract meaningful insights.
- Build an interactive Power BI dashboard with key visualizations.
- Generate a comprehensive report summarizing findings and recommendations.

# Assignment activity

## 1. Data Understanding and Preparation

The dataset is downloaded and cleaned (removing duplicate and missing values), then to be exported for Power BI

Step 1 & 2: Download Dataset and load in Python

```python
import pandas as pd

# Load the dataset
df = pd.read_csv('uber.csv')
```

Step 3: Initial Data Exploration

```python
# View the first few rows
print(df.head())

# Dataset shape
print("Dataset Shape: ", df.shape)

# Data types and non-null counts
print(df.info())

# Summary statistics for numerical columns
print("Dataset Description: \n", df.describe())
```

Step 5: Clean the data

```python
# Drop rows with any missing data in critical columns
df = df.dropna(subset=[
    'fare_amount',
    'pickup_datetime',
    'pickup_longitude', 'pickup_latitude',
    'dropoff_longitude', 'dropoff_latitude',
    'passenger_count'
])

# Remove invalid fare amounts (0 or negative)
df = df[df['fare_amount'] > 0]

# Remove invalid passenger counts (0 or unreasonable values like > 6)
df = df[(df['passenger_count'] >= 1) & (df['passenger_count'] <= 6)]

# Remove coordinates equal to 0 (invalid locations)
df = df[(df['pickup_longitude'] != 0) & (df['pickup_latitude'] != 0)]
df = df[(df['dropoff_longitude'] != 0) & (df['dropoff_latitude'] != 0)]

# Convert pickup_datetime to datetime format
df['pickup_datetime'] = pd.to_datetime(df['pickup_datetime'], errors='coerce')

# Drop rows with invalid datetime conversions (if any)
df = df.dropna(subset=['pickup_datetime'])

# Drop duplicates
df = df.drop_duplicates()

# Reset index after cleaning
df = df.reset_index(drop=True)
```

## 2. Exploratory Data Analysis (EDA)

We now generate descriptive statistics like median, mean, mode, standard deviation, quartiles, data ranges and outliers.

Step 1: Descriptive Statistics

```python
# Basic stats
print(df[['fare_amount', 'passenger_count']].describe())

# Median & Mode
print("Median Fare:", df['fare_amount'].median())
print("Mode Fare:", df['fare_amount'].mode()[0])
```

Generate + Code + Markdown

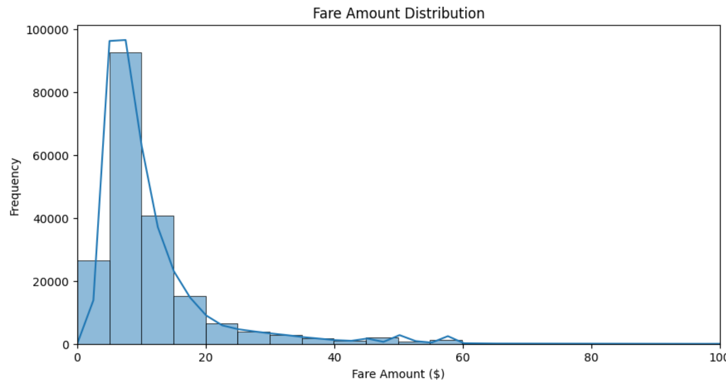
Step 2: Quartiles, Range & Outliers

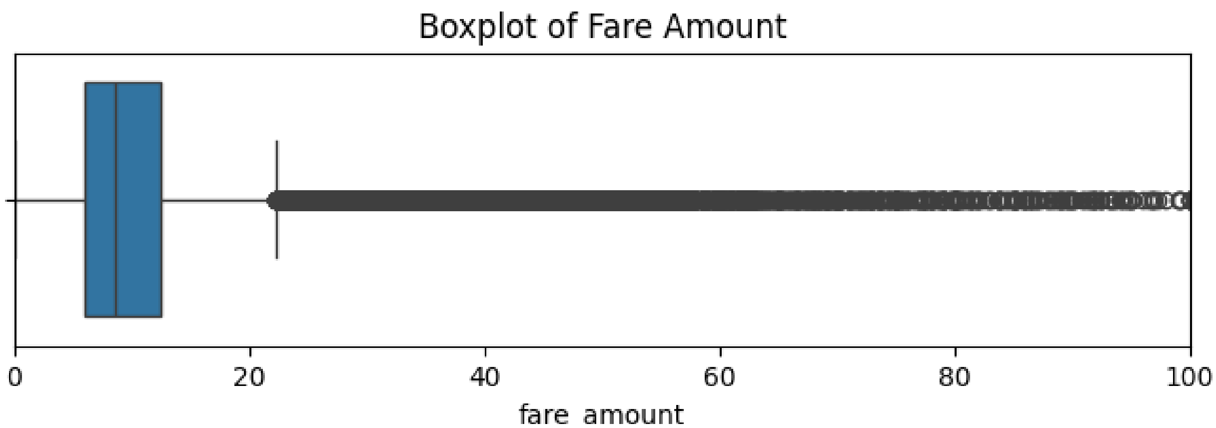
```python
Q1 = df['fare_amount'].quantile(0.25)
Q3 = df['fare_amount'].quantile(0.75)
IQR = Q3 - Q1

lower_bound = Q1 - 1.5 * IQR
upper_bound = Q3 + 1.5 * IQR

# Count outliers
outliers = df[(df['fare_amount'] < lower_bound) | (df['fare_amount'] > upper_bound)]
print("Number of outliers in fare_amount:", outliers.shape[0])
```

And also, using **matplotlib** and **seaborn**, visualizations of fare distribution patterns are created:

Step 3: Visualizing Fare Distribution

```
import matplotlib.pyplot as plt
import seaborn as sns

# Histogram of fare amount
plt.figure(figsize=(10, 5))
sns.histplot(df['fare_amount'], bins=100, kde=True)
plt.title("Fare Amount Distribution")
plt.xlabel("Fare Amount ($)")
plt.ylabel("Frequency")
plt.xlim(0, 100)  # Zoom in for better view
plt.show()

# Boxplot for outlier detection
plt.figure(figsize=(8, 2))
sns.boxplot(x=df['fare_amount'])
plt.title("Boxplot of Fare Amount")
plt.xlim(0, 100)  # Limit for visibility
plt.show()
```
✓ 5.1s

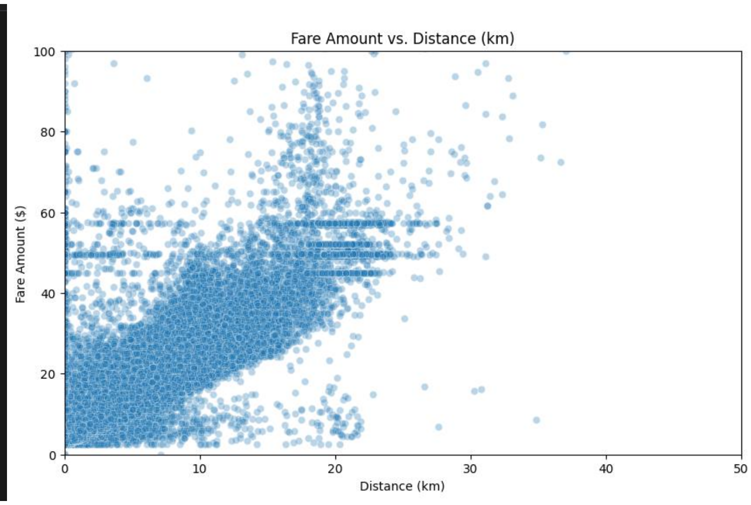



Relations between various key variables are compared and analysed:

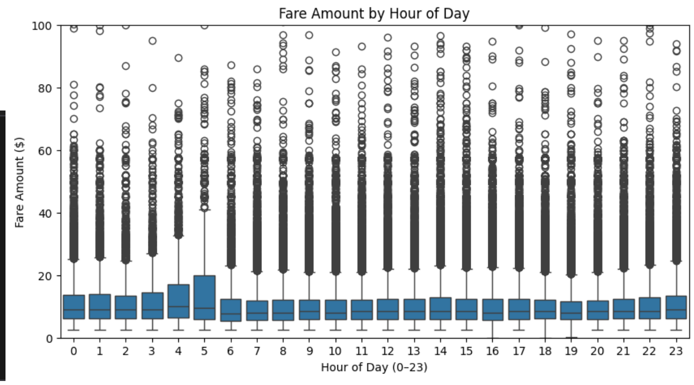
a) Fare amount vs. distance traveled

```
from math import radians, sin, cos, sqrt, atan2

# Haversine function
def haversine_distance(lat1, lon1, lat2, lon2):
    R = 6371  # Earth radius in kilometers

    phi1 = radians(lat1)
    phi2 = radians(lat2)
    delta_phi = radians(lat2 - lat1)
    delta_lambda = radians(lon2 - lon1)

    a = sin(delta_phi/2)**2 + cos(phi1)*cos(phi2)*sin(delta_lambda/2)**2
    c = 2 * atan2(sqrt(a), sqrt(1 - a))

    return R * c

# Apply to dataset
df['distance_km'] = df.apply(lambda row: haversine_distance(
    row['pickup_latitude'], row['pickup_longitude'],
    row['dropoff_latitude'], row['dropoff_longitude']
), axis=1)

# Scatter plot
plt.figure(figsize=(10, 6))
sns.scatterplot(data=df, x='distance_km', y='fare_amount', alpha=0.3)
plt.title("Fare Amount vs. Distance (km)")
plt.xlabel("Distance (km)")
plt.ylabel("Fare Amount ($)")
plt.xlim(0, 50)
plt.ylim(0, 100)
plt.show()
```
✓ 3.9s



b) Fare amount vs. time of day

```python
df['hour'] = df['pickup_datetime'].dt.hour

plt.figure(figsize=(10, 5))
sns.boxplot(x='hour', y='fare_amount', data=df)
plt.title("Fare Amount by Hour of Day")
plt.xlabel("Hour of Day (0-23)")
plt.ylabel("Fare Amount ($)")
plt.ylim(0, 100)
plt.show()
```
✓ 0.7s




## 3. Feature Engineering

We create new analytical features such as hour, day, month extracted from timestamps, day of week categorization and peak/off-peak time indicators. After that, we export a now cleaned and enhanced dataset to be used for Power BI

Step 1: Extract Date and Time features

```python
# Ensure datetime is in correct format
df['pickup_datetime'] = pd.to_datetime(df['pickup_datetime'])

# Extract time-based features
df['pickup_date'] = df['pickup_datetime'].dt.date
df['year'] = df['pickup_datetime'].dt.year
df['month'] = df['pickup_datetime'].dt.month
df['day'] = df['pickup_datetime'].dt.day
df['hour'] = df['pickup_datetime'].dt.hour
df['weekday'] = df['pickup_datetime'].dt.dayofweek  # 0 = Monday, 6 = Sunday
df['day_name'] = df['pickup_datetime'].dt.day_name()  # e.g., "Monday"
```
✓ 0.2s

Step 2: Peak vs Off-Peak Indicator

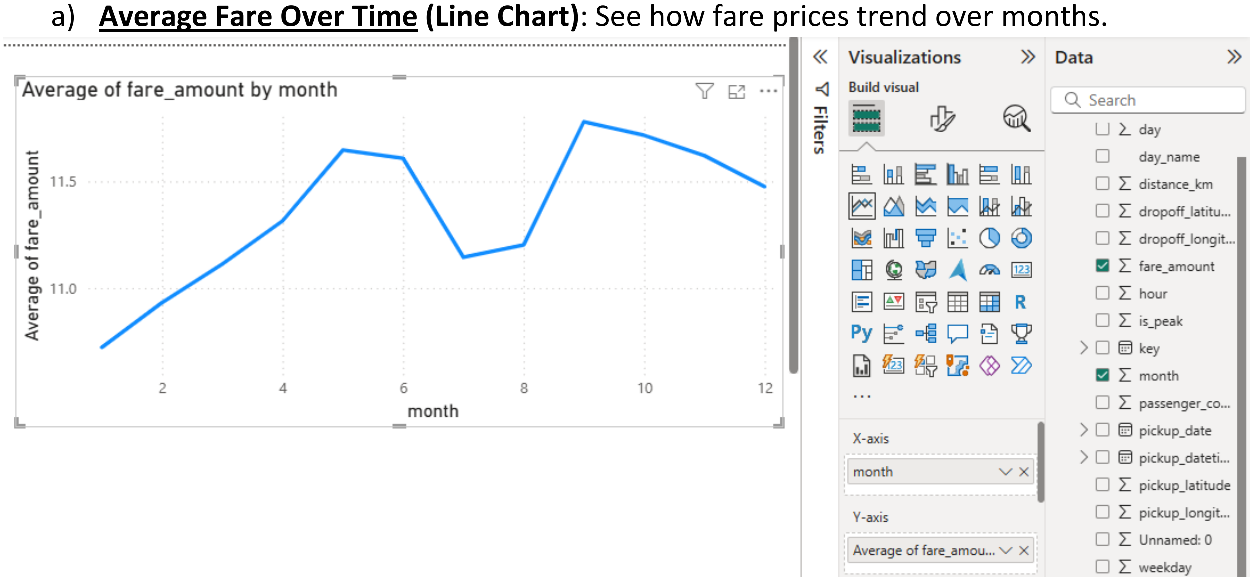
```python
def is_peak_hour(hour):
    return 1 if (7 <= hour <= 9) or (16 <= hour <= 19) else 0

df['is_peak'] = df['hour'].apply(is_peak_hour)
```
✓ 0.1s

```python
df.to_csv('uber_enhanced.csv', index=False)
print("✅ Enhanced dataset saved as 'uber_enhanced.csv'")
```

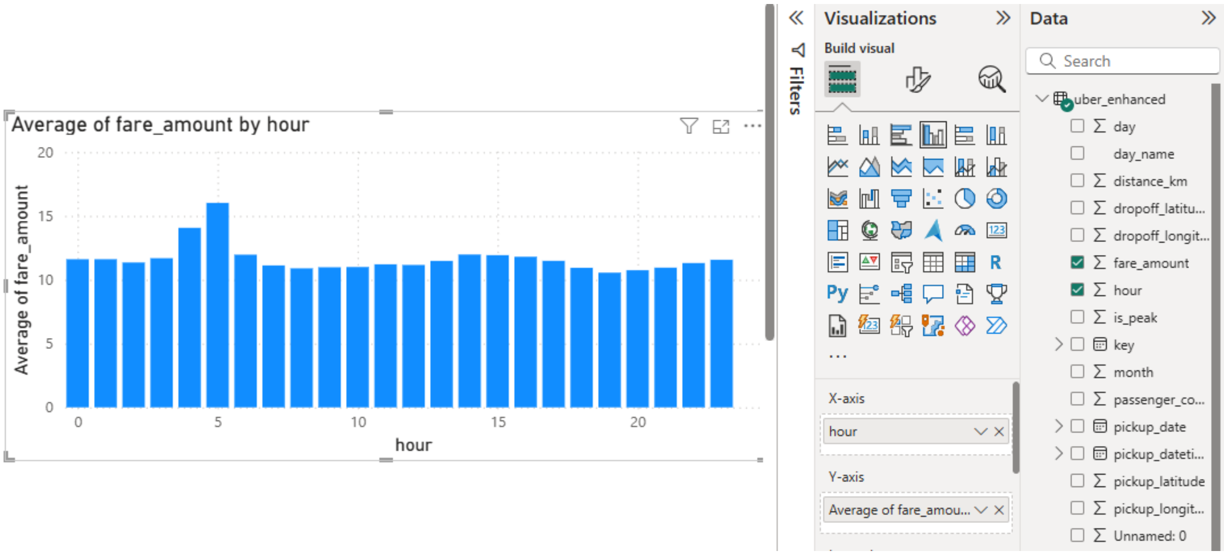
## 4. Data Analysis in Power BI

By using Power BI and importing data from the now enhanced dataset, we can now get visualization of needed data, for example:

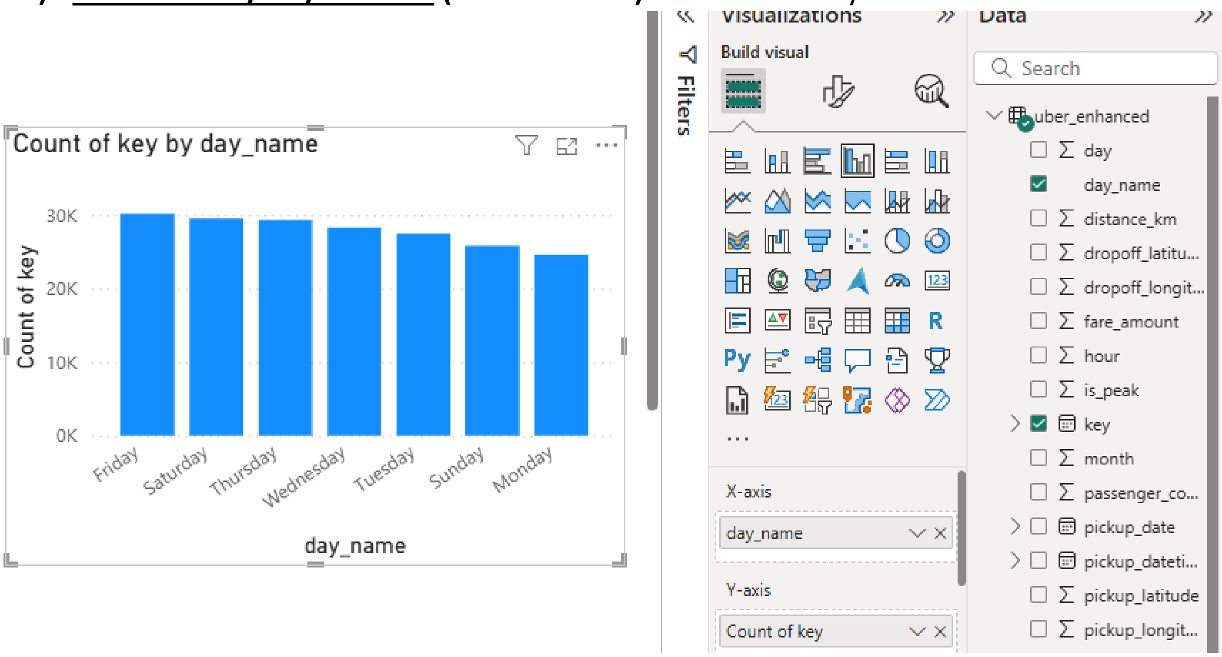
a) **<u>Average Fare Over Time</u> (Line Chart)**: See how fare prices trend over months.



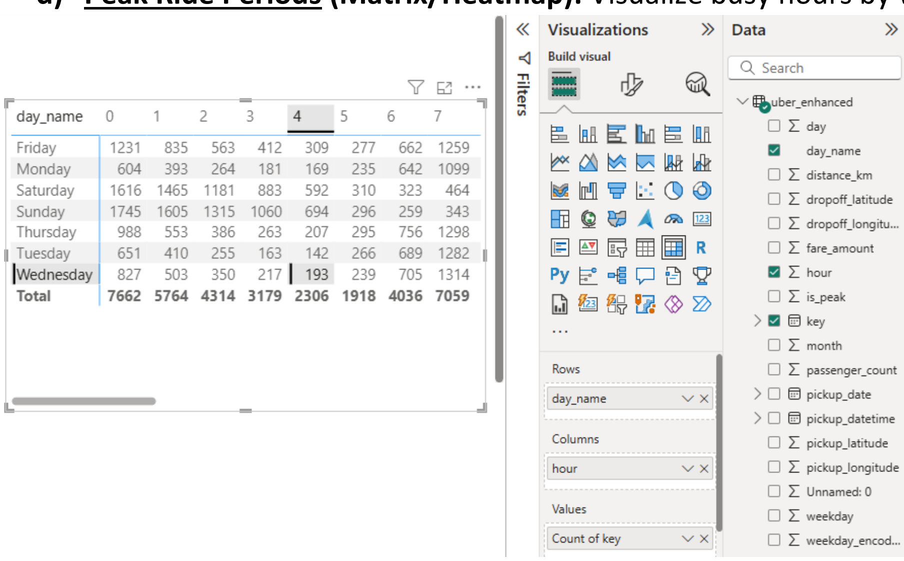
**b) <u>Average Fare by Hour of Day</u> (Bar Chart):** Compare average fare at each hour of the day.




**c) <u>Ride Count by Day of Week</u> (Column Chart):** See busiest days

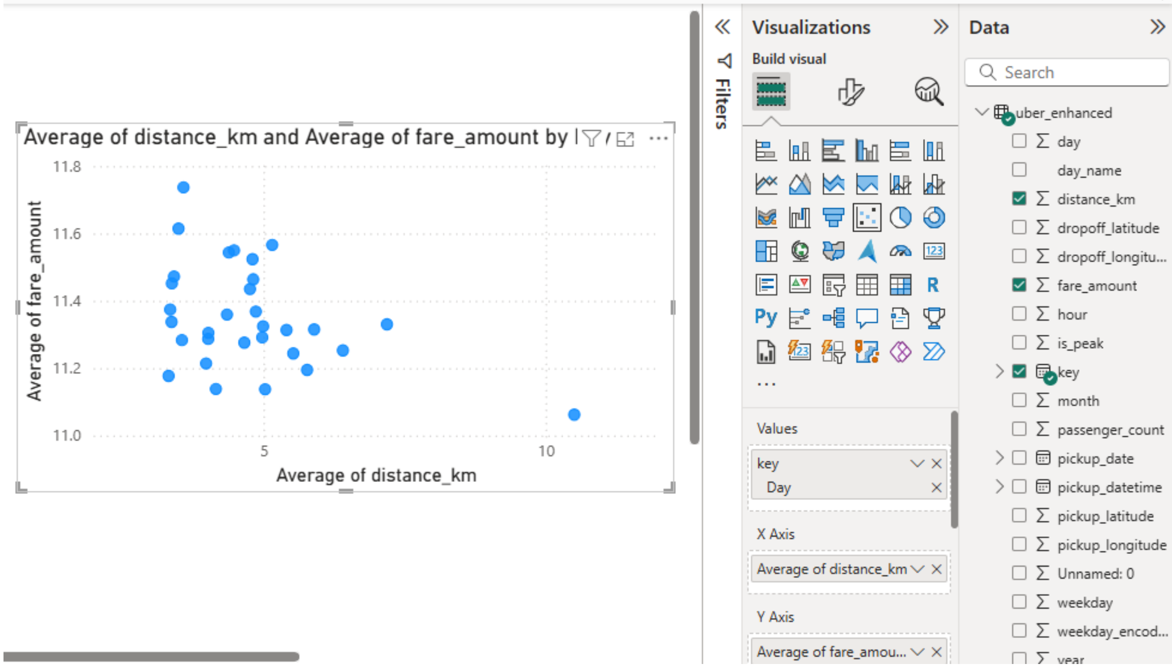



**d) <u>Peak Ride Periods</u> (Matrix/Heatmap):** Visualize busy hours by weekday



| day_name | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
|---|---|---|---|---|---|---|---|---|
| Friday | 1231 | 835 | 563 | 412 | 309 | 277 | 662 | 1259 |
| Monday | 604 | 393 | 264 | 181 | 169 | 235 | 642 | 1099 |
| Saturday | 1616 | 1465 | 1181 | 883 | 592 | 310 | 323 | 464 |
| Sunday | 1745 | 1605 | 1315 | 1060 | 694 | 296 | 259 | 343 |
| Thursday | 988 | 553 | 386 | 263 | 207 | 295 | 756 | 1298 |
| Tuesday | 651 | 410 | 255 | 163 | 142 | 266 | 689 | 1282 |
| Wednesday | 827 | 503 | 350 | 217 | 193 | 239 | 705 | 1314 |
| **Total** | **7662** | **5764** | **4314** | **3179** | **2306** | **1918** | **4036** | **7059** |

**e) <u>Fare vs Distance</u> (Scatterplot):** See correlation between distance and fare.




## 5. Dashboard Creation in Power BI

Here we're creating a dashboard page with various data visualization, such as:

a) Distribution of Fare Amounts (Histogram of Fare amounts)

b) Ride Distance (Average distance travelled by time of the day)

c) Fare Distribution (Average fare amount by month)

d) Temporal Patterns (Count of rides per months)

e) Map of Rides (Visual of rides' locations)

f) Cards used to show average Fare amount, total rides and total distance

g) Date Slicer, used to filter data shown by date