Akasa Air Data Engineering & Analytics Task Report

Ravi Ray - 2347142

Objective

The primary objective of this task is to clean, normalize, and analyze a sample dataset from the aviation industry, which contains flight details such as flight numbers, departure and arrival times, airlines, and delays. The goal is to derive actionable insights that can be used to improve operational efficiency and punctuality.

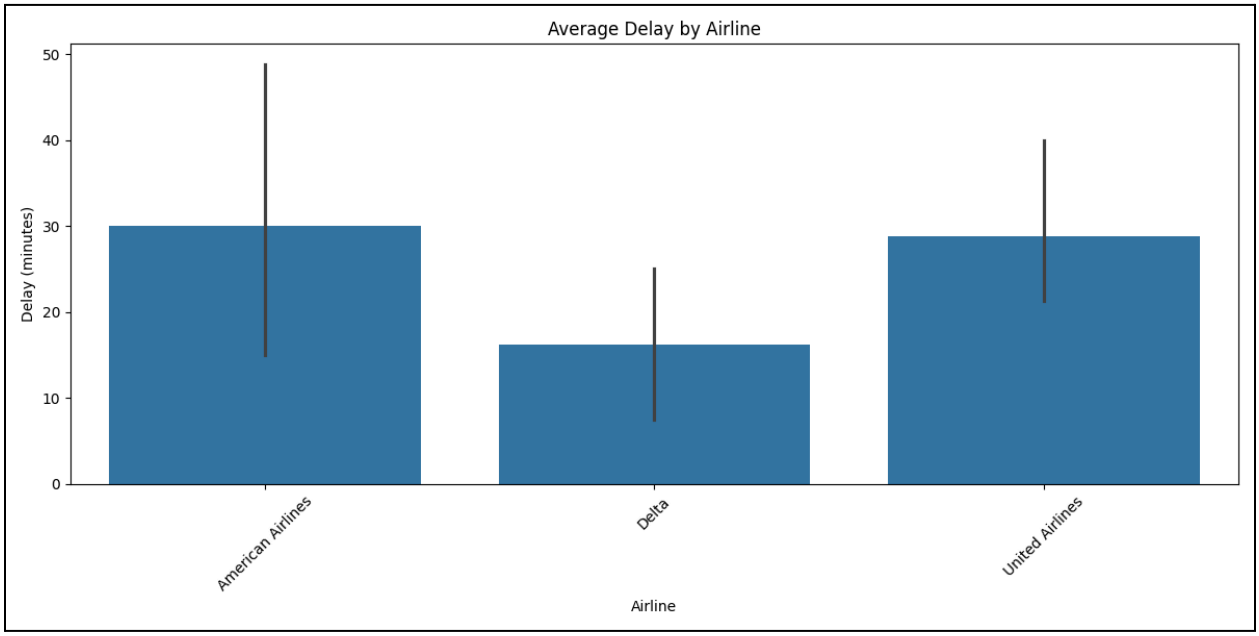
Summary of the Data:

1. **Flight Information:**
    - **Flight Numbers:** There are 12 flight entries, with **3 unique flight numbers**. The most frequent flight is **AA1234**, which appears **4 times**.
    - **Departure and Arrival Dates:** The data includes **4 unique departure dates** and **4 unique arrival dates**. The most frequent date is **09/01/2023**.
    - **Departure and Arrival Times:** There are **9 unique departure and arrival times**. The most common departure time is **08:30 AM**, while the most common arrival time is **10:45 AM**.
2. **Airline Information:**
    - There are **3 unique airlines** in the dataset:
        - **American Airlines** (most frequent, appears 4 times)
        - **United Airlines**
        - **Delta**
3. **Delay Information:**
    - The **DelayMinutes** column has data for 10 out of the 12 entries.
    - The average delay across all flights is **25 minutes**, with a **standard deviation of 16.67 minutes**.
    - The minimum delay is **5 minutes**, while the maximum delay is **60 minutes**.
    - The 50th percentile (median) delay is **22.5 minutes**, and the 75th percentile delay is **28.75 minutes**.

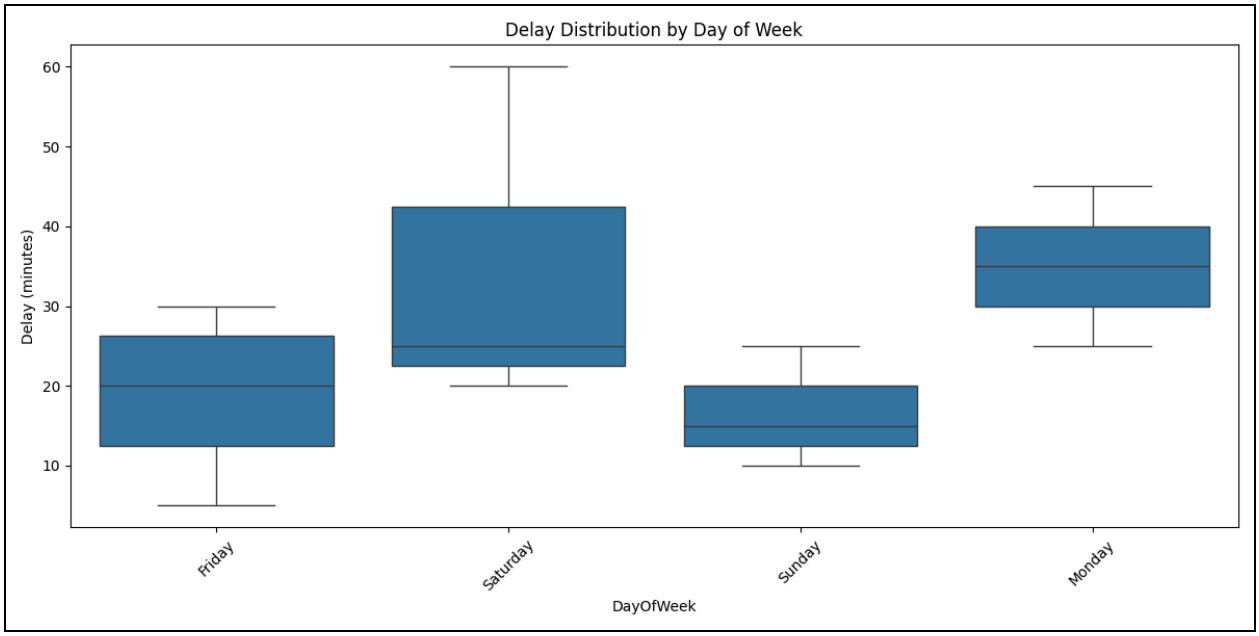
Key Insights Derived With Visualization

The analysis of the dataset reveals several significant insights:

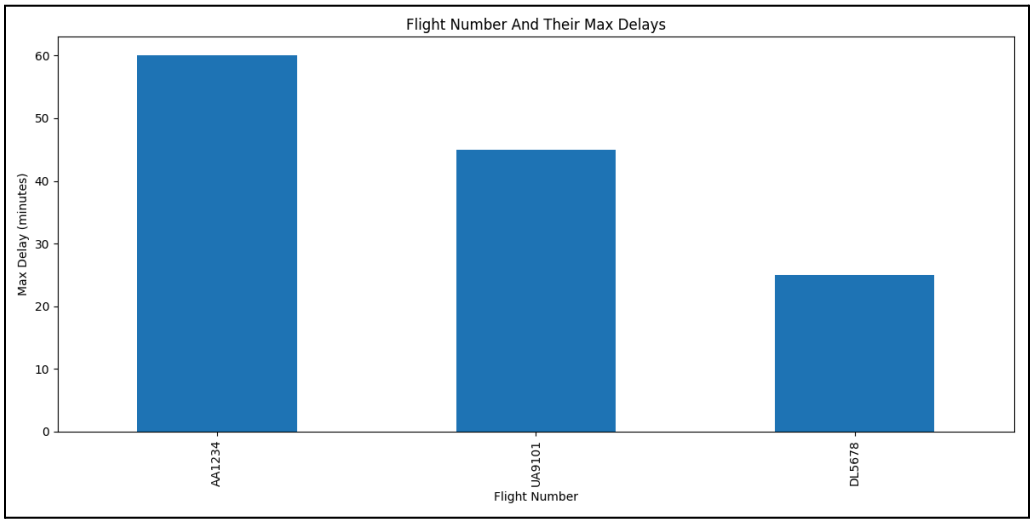
1. **Average Delay by Airline:**
    - **American Airlines** has the highest average delay (30 minutes), followed by **United Airlines** (28.75 minutes), and **Delta** (16.25 minutes)..



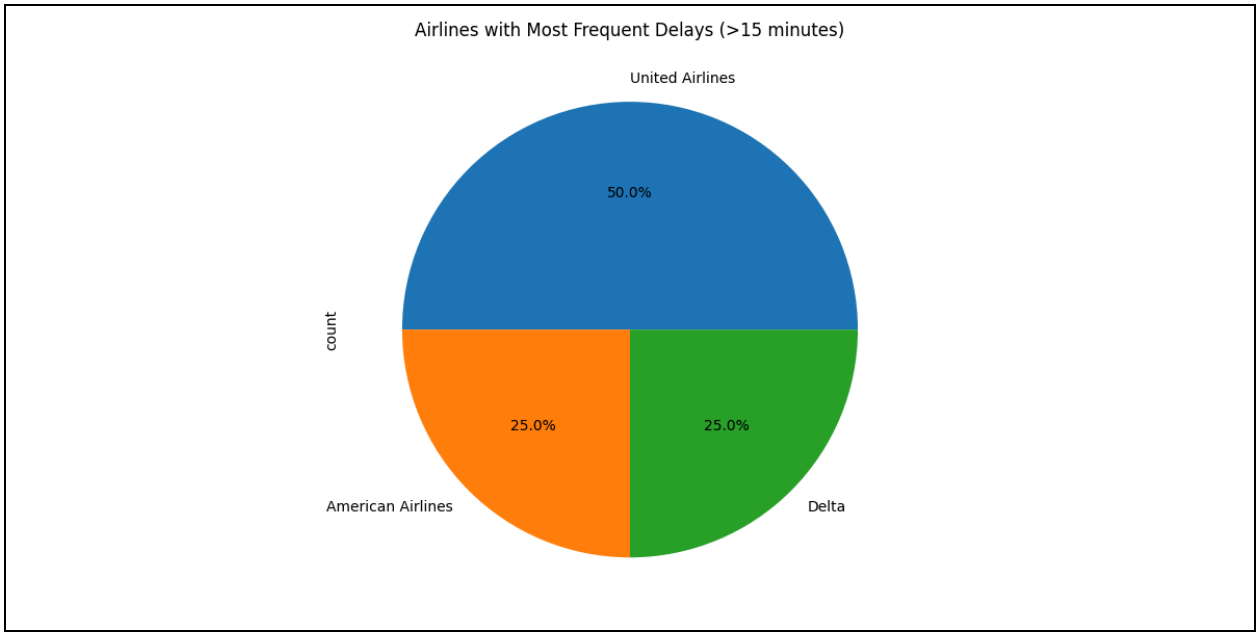
2. **Correlation Between Departure Time and Delay**:
    - The correlation coefficient between **departure time and delay** is **0.61**, indicating a moderate positive relationship. Delays tend to increase at certain times of the day.
3. **Average Delay by Day of the Week**:
    - **Monday** and **Saturday** have the highest average delays (35 minutes each), followed by **Friday** (18.75 minutes) and **Sunday** (16.67 minutes).



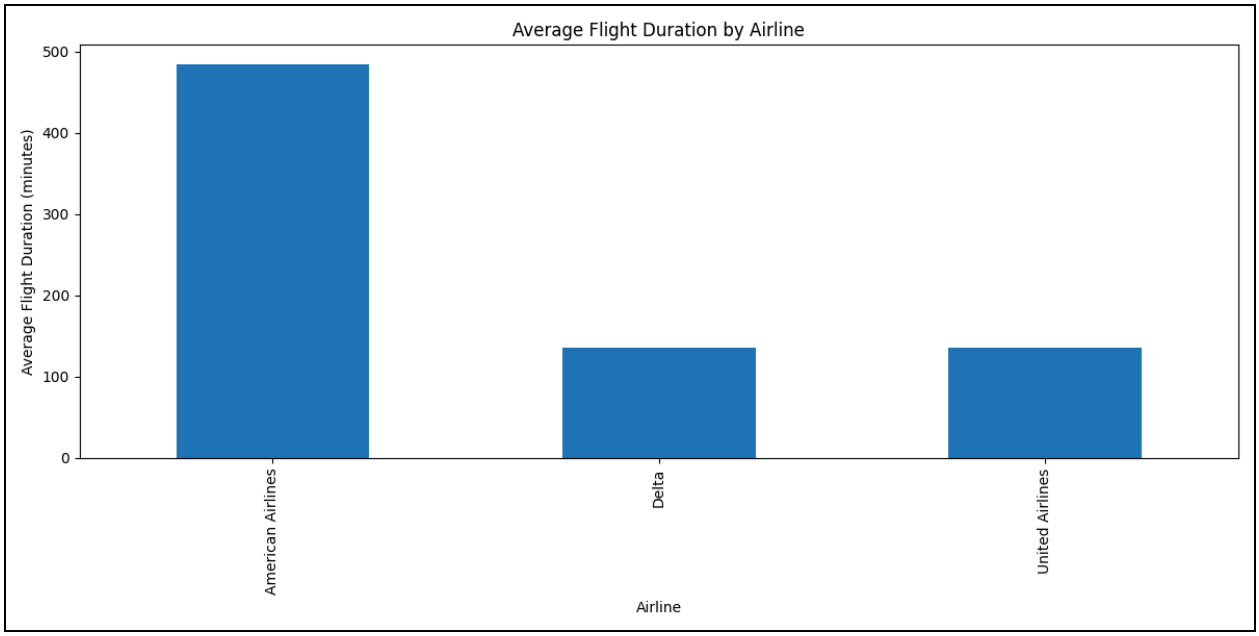
4. **Flights with the Longest Delays**:
    - The top five longest delays include **American Airlines Flight AA1234** (60 minutes) and **United Airlines Flight UA9101** (45 minutes).



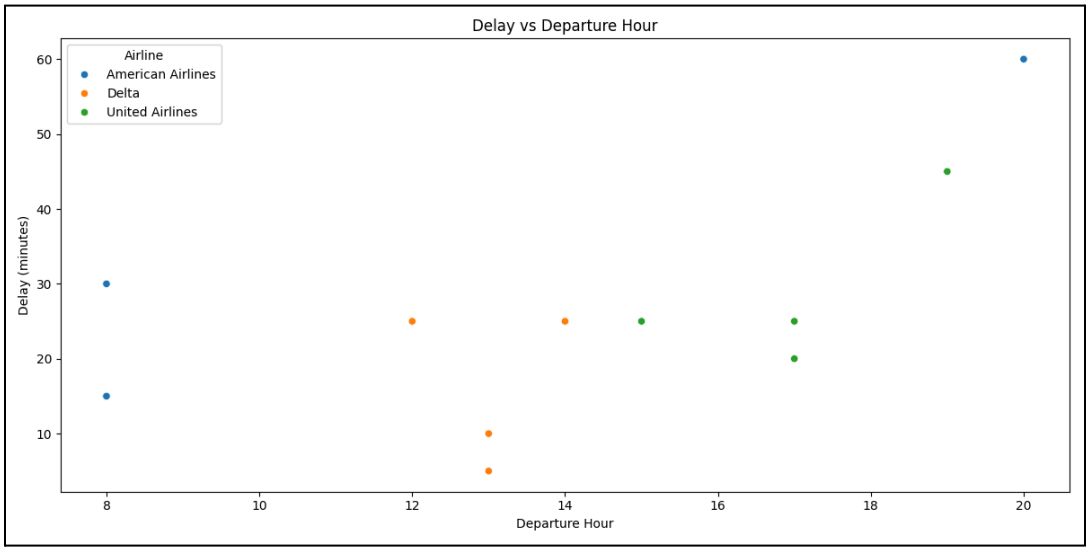
5. **Airlines with Most Frequent Delays(>15 minutes)**:
    - **United Airlines** had the most frequent delays with 4 occurrences, followed by **American Airlines** and **Delta** with 2 each.



6. **Average Flight Duration by Airline**:
    - **American Airlines** had the longest average flight duration (483.75 minutes), while **Delta** and **United Airlines** averaged around 136.25 minutes.



7. **Departure Time And Delays**:
    - It shows the relationship between **departure time and delays**, with different colors for each airline. This helps in visualizing the correlation between late departures and higher delays.



Recommendations

Based on the insights derived from the analysis, the following recommendations are made:

1. **Target Operational Improvements:**
    - Focus on reducing delays for **American Airlines** and **United Airlines** due to their higher average delays. Investigating specific reasons for frequent delays could offer solutions.
2. **Investigate Monday and Saturday Delays:**
    - Given the significant delays on **Monday** and **Saturday**, operations on these days should be reviewed to identify bottlenecks or capacity issues.
3. **Optimize Departure Times:**
    - Since there's a moderate correlation between departure time and delays, airlines may need to adjust schedules or introduce buffer times to mitigate delays during peak hours.
4. **Improve Monitoring for Top Delayed Flights:**
    - Particular attention should be given to flights like **AA1234** and **UA9101**, which have experienced the longest delays. Monitoring these flights more closely may help preemptively address delay causes.

Flight Delay Analysis Pipeline Documentation

Step 1: Install Required Libraries

Install the necessary libraries using the following command:

```
pip install pandas mysql-connector-python matplotlib seaborn
```

Step 2: Create MySQL Database

Ensure you have MySQL installed and running. Create a database named `akasa` using the following SQL command:

```
CREATE DATABASE akasa;
```

Step 3: Python Script Setup

Create a Python script and follow these steps.

Step 3.1: Import Libraries

```
import pandas as pd
import mysql.connector
from mysql.connector import Error
import matplotlib.pyplot as plt
import seaborn as sns
from datetime import timedelta
```

Step 3.2: Connect to MySQL Database

```
def create_connection():
    try:
```

```
        connection = mysql.connector.connect(
            host='host',
            database='database_name',
            user='username',
            password='password'
        )
        if connection.is_connected():
            return connection
    except Error as e:
        print(f"Error: {e}")
        return None
```

Step 3.3: Create Flights Table

```
def create_flights_table(connection):
    cursor = connection.cursor()
    cursor.execute('DROP TABLE IF EXISTS flights')
    cursor.execute('''
        CREATE TABLE flights (
            id INT AUTO_INCREMENT PRIMARY KEY,
            FlightNumber VARCHAR(10),
            DepartureDateTime DATETIME,
            ArrivalDateTime DATETIME,
            Airline VARCHAR(50),
            DelayMinutes INT,
            FlightDuration INT,
            DayOfWeek VARCHAR(10)
        )
    ''')
    connection.commit()
```

Step 4: Clean and Normalize Data

```
def clean_and_normalize_data(df):
    # Convert date and time columns
    df['DepartureDateTime'] = pd.to_datetime(df['DepartureDat
e'] + ' ' + df['DepartureTime'])
    df['ArrivalDateTime'] = pd.to_datetime(df['ArrivalDate']
+ ' ' + df['ArrivalTime'])

    # Remove duplicates
    df = df.drop_duplicates()

    # Handle missing values in DelayMinutes
    df['DelayMinutes'] = df['DelayMinutes'].fillna(df['DelayM
inutes'].mean())

    # Calculate flight duration
    df['FlightDuration'] = (df['ArrivalDateTime'] - df['Depar
tureDateTime']).dt.total_seconds() / 60

    # Correct inconsistent time entries
    mask = df['ArrivalDateTime'] <= df['DepartureDateTime']
    df.loc[mask, 'ArrivalDateTime'] += timedelta(days=1)
    df['FlightDuration'] = (df['ArrivalDateTime'] - df['Depar
tureDateTime']).dt.total_seconds() / 60

    # Add day of the week
    df['DayOfWeek'] = df['DepartureDateTime'].dt.day_name()

    return df[['FlightNumber', 'DepartureDateTime', 'ArrivalD
ateTime', 'Airline', 'DelayMinutes', 'FlightDuration', 'DayOf
Week']]
```

Step 5: Insert Data into MySQL

```
def insert_data(connection, df):
    cursor = connection.cursor()
    for _, row in df.iterrows():
        cursor.execute('''
            INSERT INTO flights (FlightNumber, DepartureDateT
ime, ArrivalDateTime, Airline, DelayMinutes, FlightDuration,
DayOfWeek)
            VALUES (%s, %s, %s, %s, %s, %s, %s)
        ''', tuple(row))
    connection.commit()
```

Step 6: Analyze Flight Delays

```
def analyze_delays(df):
    print("1. Average delay by airline:")
    airline_delays = df.groupby('Airline')['DelayMinutes'].me
an().sort_values(ascending=False)
    print(airline_delays)

    print("\n2. Correlation between departure time and dela
y:")
    df['DepartureHour'] = df['DepartureDateTime'].dt.hour
    correlation = df['DepartureHour'].corr(df['DelayMinute
s'])
    print(f"Correlation coefficient: {correlation:.2f}")

    print("\n3. Average delay by day of week:")
    day_delays = df.groupby('DayOfWeek')['DelayMinutes'].mean
().sort_values(ascending=False)
    print(day_delays)

    print("\n4. Flights with longest delays (top 5):")
    long_delays = df.nlargest(5, 'DelayMinutes')[['FlightNumb
```

```
er', 'Airline', 'DelayMinutes']]
    print(long_delays)

    print("\n5. Airlines with most frequent delays (>15 minut
es):")
    frequent_delays = df[df['DelayMinutes'] > 15]['Airline'].
value_counts()
    print(frequent_delays)

    print("\n6. Average flight duration by airline:")
    avg_duration = df.groupby('Airline')['FlightDuration'].me
an().sort_values(ascending=False)
    print(avg_duration)
```

Step 7: Visualize the Data

```
def visualize_delays(df):
    # Barplot: Average Delay by Airline
    plt.figure(figsize=(12, 6))
    sns.barplot(x='Airline', y='DelayMinutes', data=df)
    plt.title('Average Delay by Airline')
    plt.ylabel('Delay (minutes)')
    plt.xticks(rotation=45)
    plt.tight_layout()
    plt.savefig('average_delay_by_airline.png')
    plt.show()

    # Histogram: Delay Distribution
    plt.figure(figsize=(12, 6))
    sns.histplot(data=df, x='DelayMinutes', bins=20, kde=Tru
e)
    plt.title('Delay Distribution')
    plt.xlabel('Delay (minutes)')
    plt.ylabel('Frequency')
    plt.tight_layout()
```

```
    plt.savefig('delay_distribution.png')
    plt.show()

    # Boxplot: Delay Distribution by Day of Week
    plt.figure(figsize=(12, 6))
    sns.boxplot(x='DayOfWeek', y='DelayMinutes', data=df)
    plt.title('Delay Distribution by Day of Week')
    plt.ylabel('Delay (minutes)')
    plt.xticks(rotation=45)
    plt.tight_layout()
    plt.savefig('delay_by_day_of_week.png')
    plt.show()

    # Scatterplot: Delay vs Departure Hour
    plt.figure(figsize=(12, 6))
    sns.scatterplot(x='DepartureHour', y='DelayMinutes', data
=df, hue='Airline')
    plt.title('Delay vs Departure Hour')
    plt.xlabel('Departure Hour')
    plt.ylabel('Delay (minutes)')
    plt.tight_layout()
    plt.savefig('delay_vs_departure_hour.png')
    plt.show()
```

Step 8: Main Execution

```
def main():
    # Load CSV file
    df = pd.read_csv('aviation_data.csv')

    # Clean and normalize data
    cleaned_df = clean_and_normalize_data(df)

    # Export cleaned dataset as CSV
    cleaned_df.to_csv('cleaned_aviation_data.csv', index=Fals
```

```
e)
    print("Cleaned dataset exported as 'cleaned_aviation_dat
a.csv'")

    # Create MySQL connection
    connection = create_connection()
    if connection is None:
        return

    # Create flights table
    create_flights_table(connection)

    # Insert data into MySQL
    insert_data(connection, cleaned_df)

    # Perform data analysis
    analyze_delays(cleaned_df)

    # Create visualizations
    visualize_delays(cleaned_df)

    # Close MySQL connection
    connection.close()
    print("Data processing and analysis complete.")
```

Execution

This method converts your `.ipynb` file to a Python script and executes it.

1. Enter Your Database Credentials
2. **Install** `nbconvert` (if you haven't already):

   ```
   pip install nbconvert
   ```

3. **Convert and run the notebook**:

You can convert your `.ipynb` file to a Python script and execute it directly in the terminal:

```
jupyter nbconvert --to script your_notebook.ipynb
python your_notebook.py
```

Outputs:

- Cleaned dataset saved as `cleaned_aviation_data.csv`.
- Visualizations saved as PNG files (e.g., `average_delay_by_airline.png`).
- Analytical insights printed in the console.
