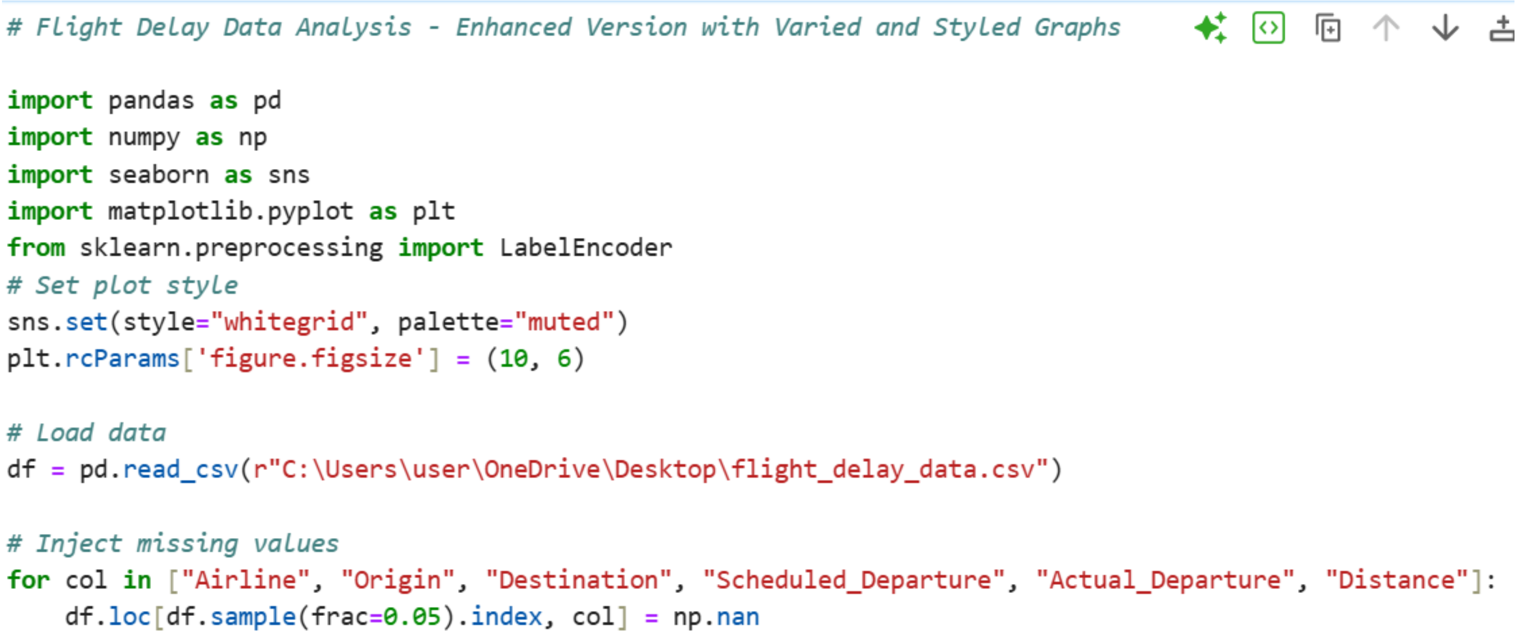
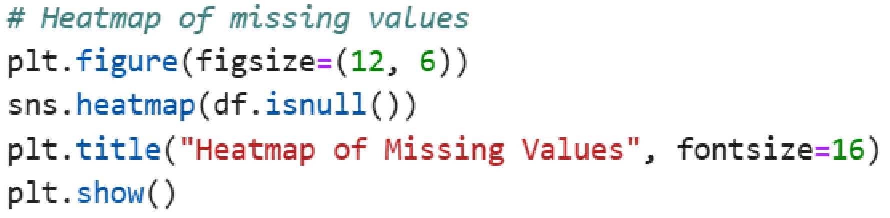
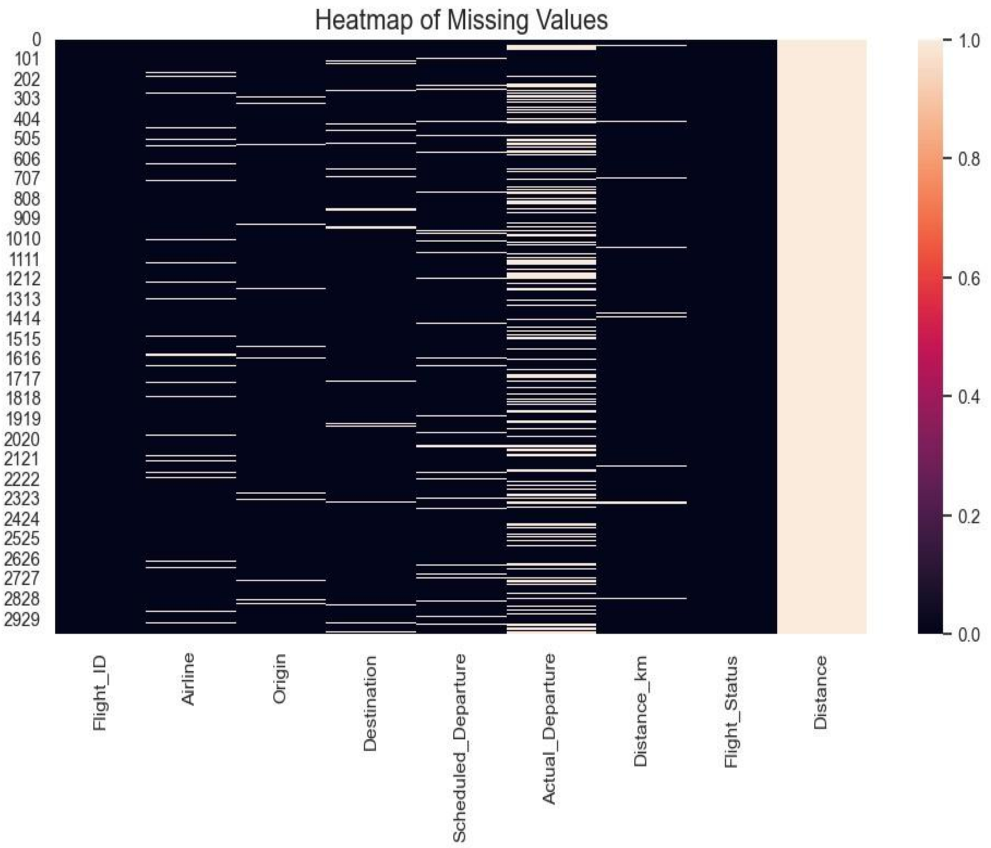
RESULTS

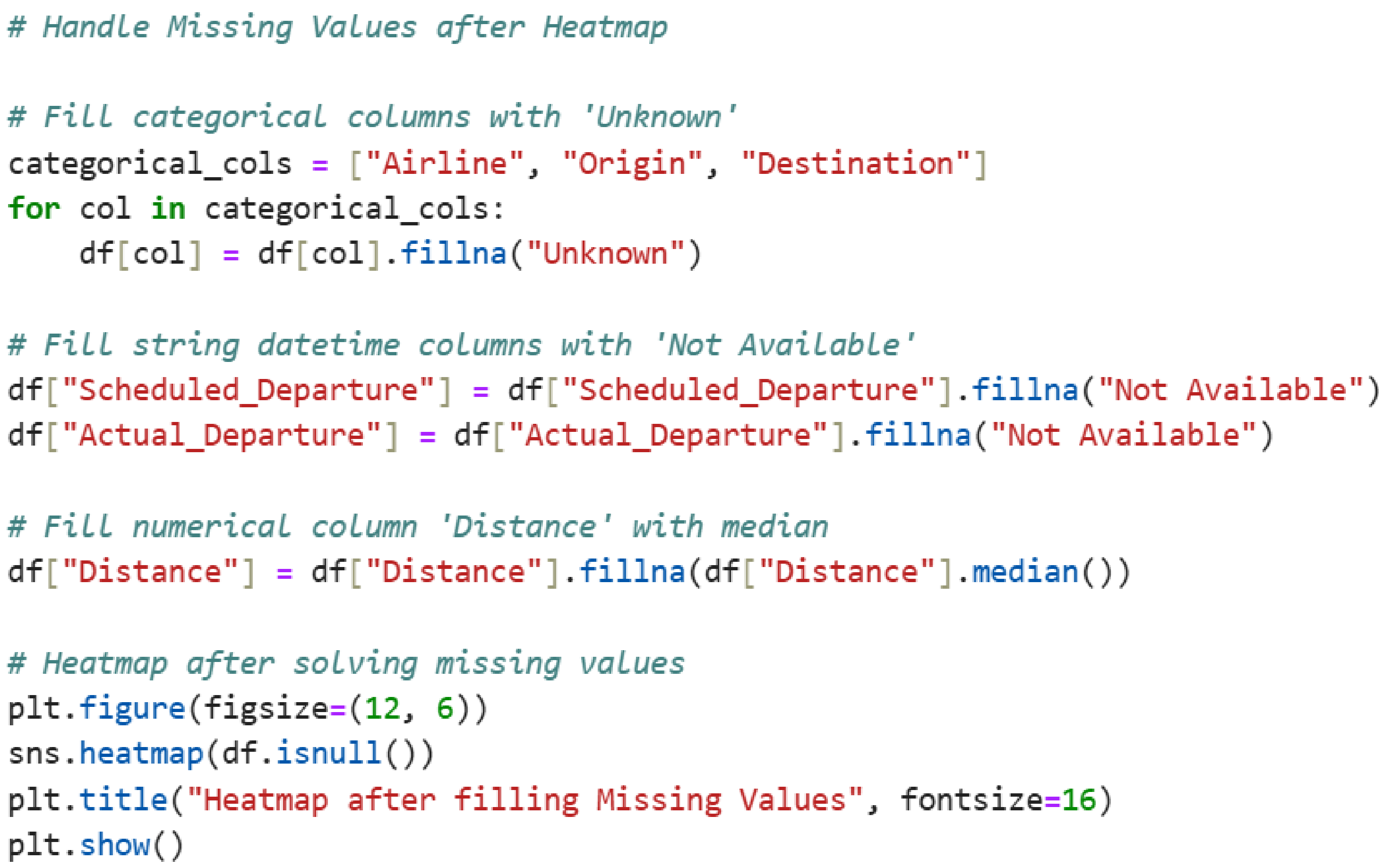


```
# Flight Delay Data Analysis - Enhanced Version with Varied and Styled Graphs

import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.preprocessing import LabelEncoder
# Set plot style
sns.set(style="whitegrid", palette="muted")
plt.rcParams['figure.figsize'] = (10, 6)

# Load data
df = pd.read_csv(r"C:\Users\user\OneDrive\Desktop\flight_delay_data.csv")

# Inject missing values
for col in ["Airline", "Origin", "Destination", "Scheduled_Departure", "Actual_Departure", "Distance"]:
    df.loc[df.sample(frac=0.05).index, col] = np.nan
```

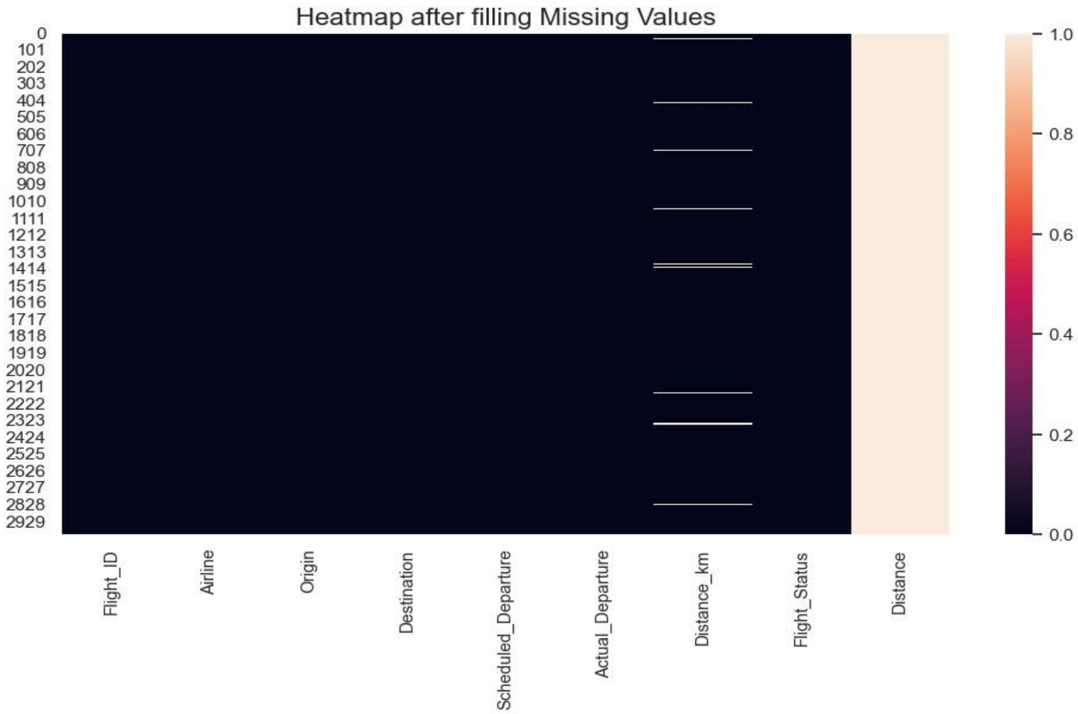


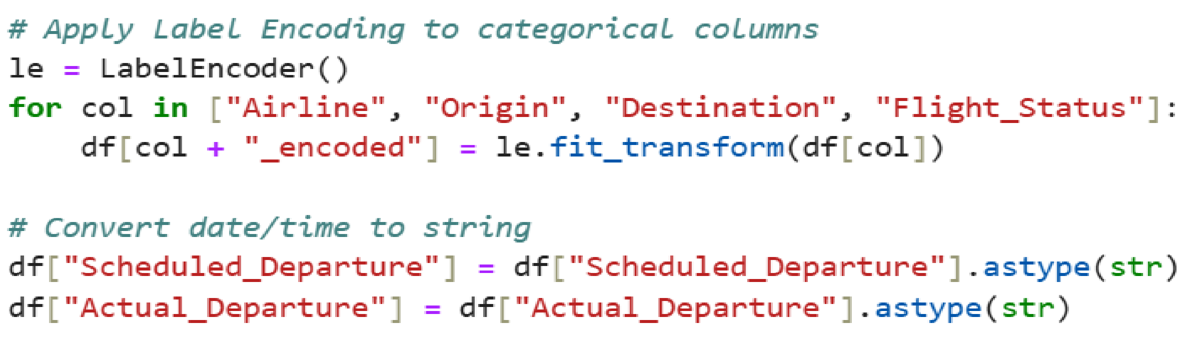
```
# Heatmap of missing values
plt.figure(figsize=(12, 6))
sns.heatmap(df.isnull())
plt.title("Heatmap of Missing Values", fontsize=16)
plt.show()
```





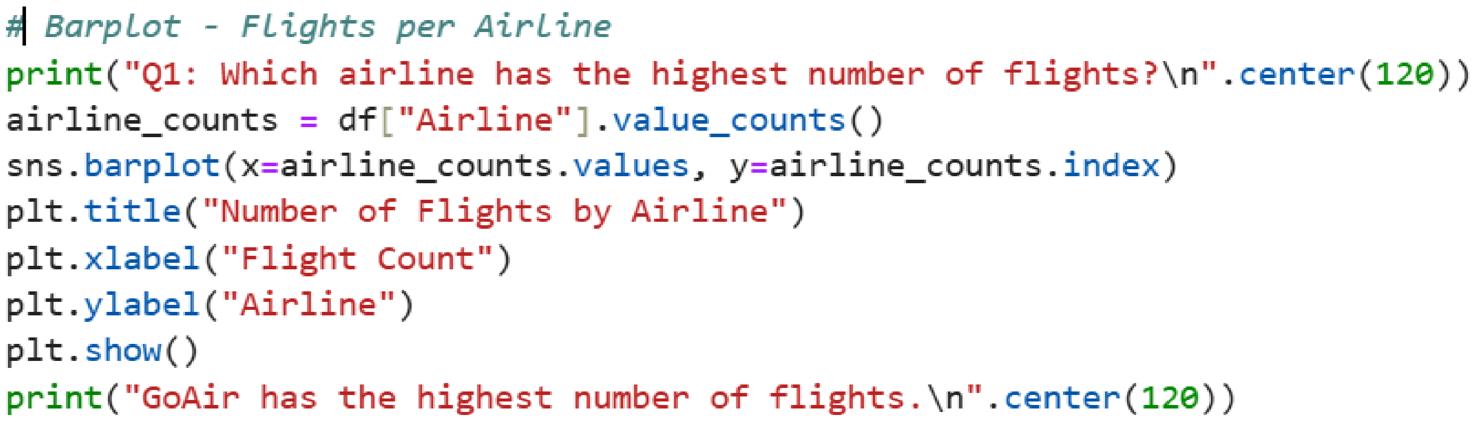
```python
# Handle Missing Values after Heatmap

# Fill categorical columns with 'Unknown'
categorical_cols = ["Airline", "Origin", "Destination"]
for col in categorical_cols:
    df[col] = df[col].fillna("Unknown")

# Fill string datetime columns with 'Not Available'
df["Scheduled_Departure"] = df["Scheduled_Departure"].fillna("Not Available")
df["Actual_Departure"] = df["Actual_Departure"].fillna("Not Available")

# Fill numerical column 'Distance' with median
df["Distance"] = df["Distance"].fillna(df["Distance"].median())

# Heatmap after solving missing values
plt.figure(figsize=(12, 6))
sns.heatmap(df.isnull())
plt.title("Heatmap after filling Missing Values", fontsize=16)
plt.show()
```



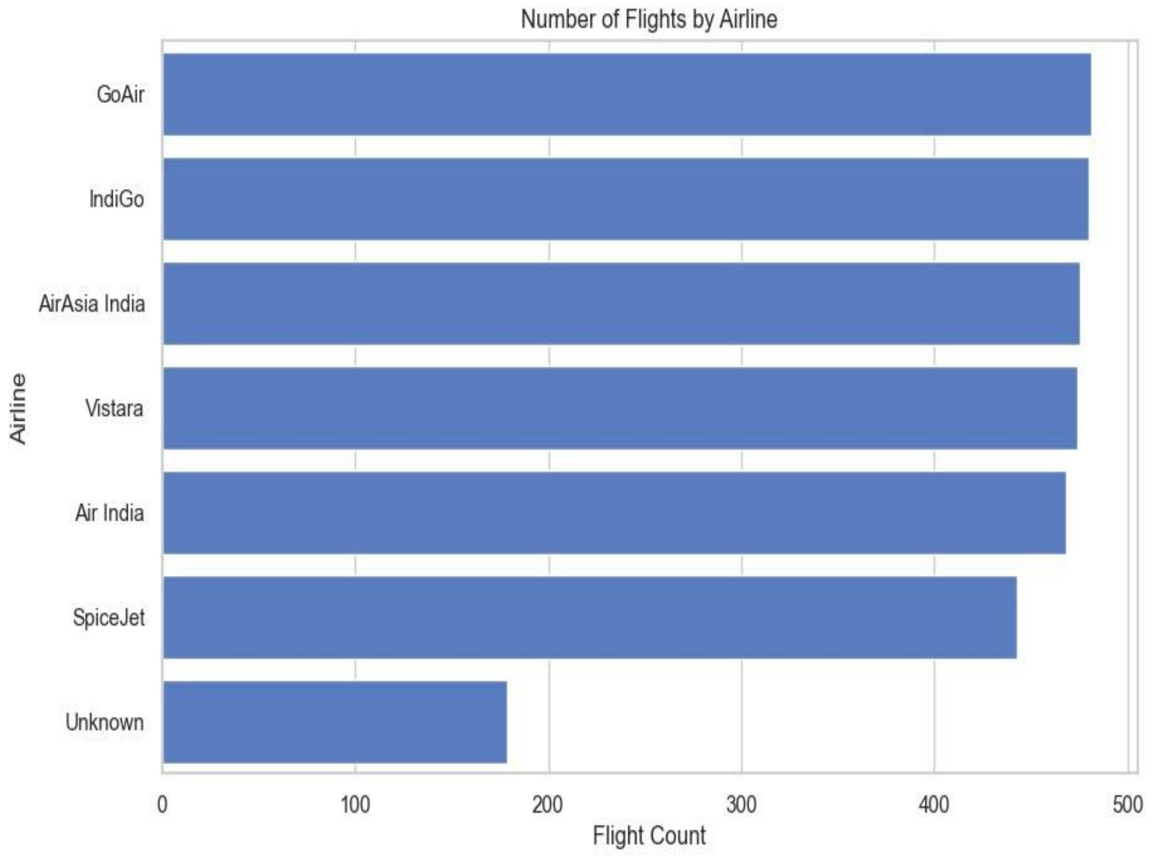


```python
# Apply Label Encoding to categorical columns
le = LabelEncoder()
for col in ["Airline", "Origin", "Destination", "Flight_Status"]:
    df[col + "_encoded"] = le.fit_transform(df[col])

# Convert date/time to string
df["Scheduled_Departure"] = df["Scheduled_Departure"].astype(str)
df["Actual_Departure"] = df["Actual_Departure"].astype(str)
```

Q1: Which airline has the highest number of flights?

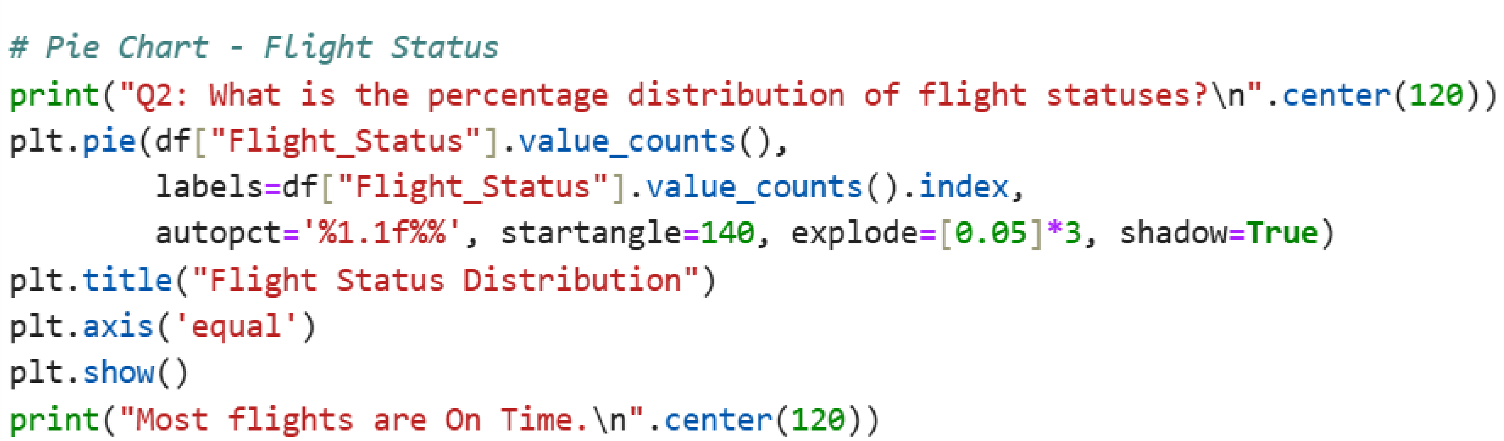


```
# Barplot - Flights per Airline
print("Q1: Which airline has the highest number of flights?\n".center(120))
airline_counts = df["Airline"].value_counts()
sns.barplot(x=airline_counts.values, y=airline_counts.index)
plt.title("Number of Flights by Airline")
plt.xlabel("Flight Count")
plt.ylabel("Airline")
plt.show()
print("GoAir has the highest number of flights.\n".center(120))
```

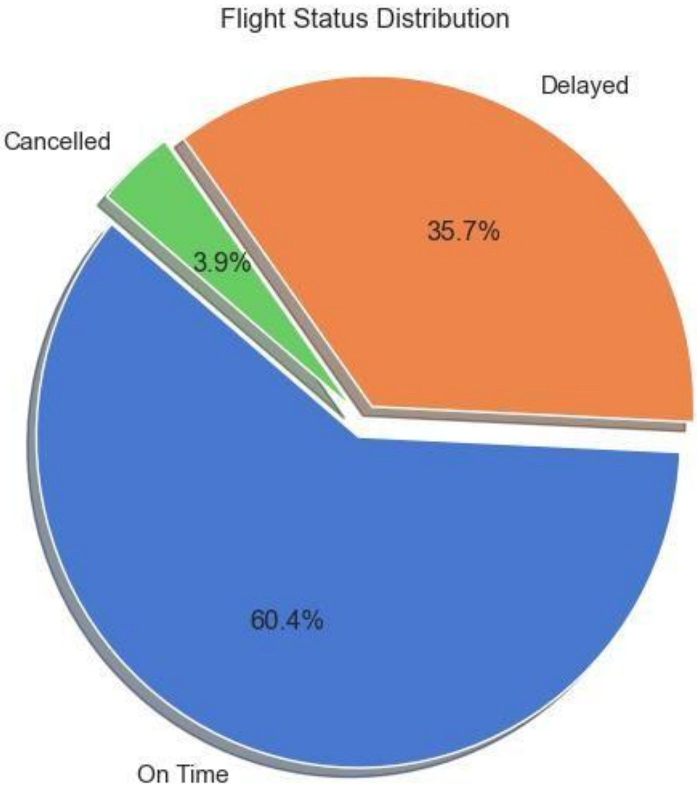


```
GoAir has the highest
  number of flights.
```

Q2: What is the percentage distribution of flight statuses?

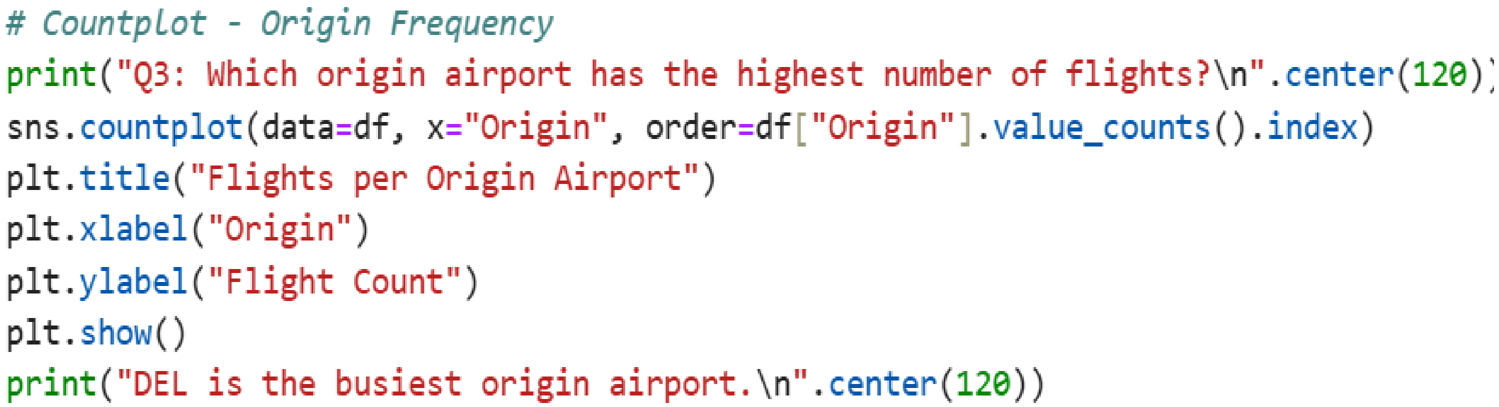


```
# Pie Chart - Flight Status
print("Q2: What is the percentage distribution of flight statuses?\n".center(120))
plt.pie(df["Flight_Status"].value_counts(),
        labels=df["Flight_Status"].value_counts().index,
        autopct='%1.1f%%', startangle=140, explode=[0.05]*3, shadow=True)
plt.title("Flight Status Distribution")
plt.axis('equal')
plt.show()
print("Most flights are On Time.\n".center(120))
```

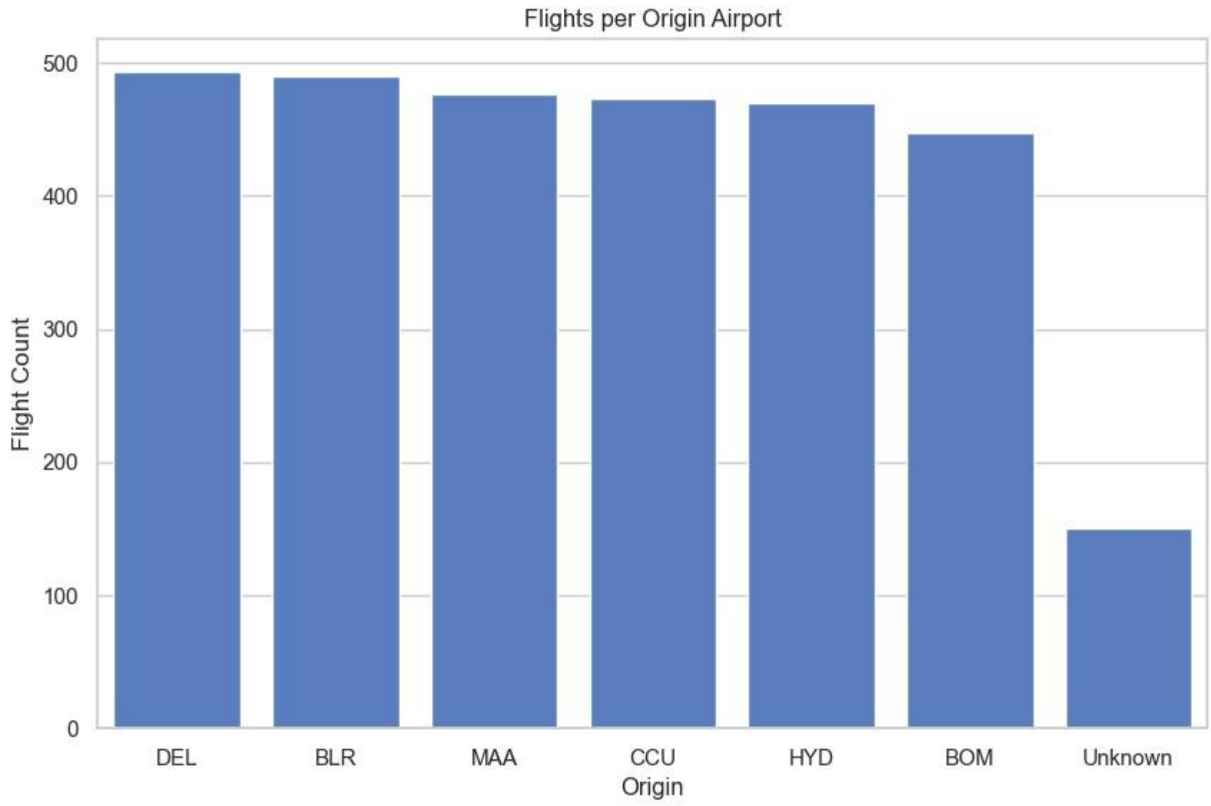


Most Flights are on Time

Q3: Which origin airport has the highest number of flights?

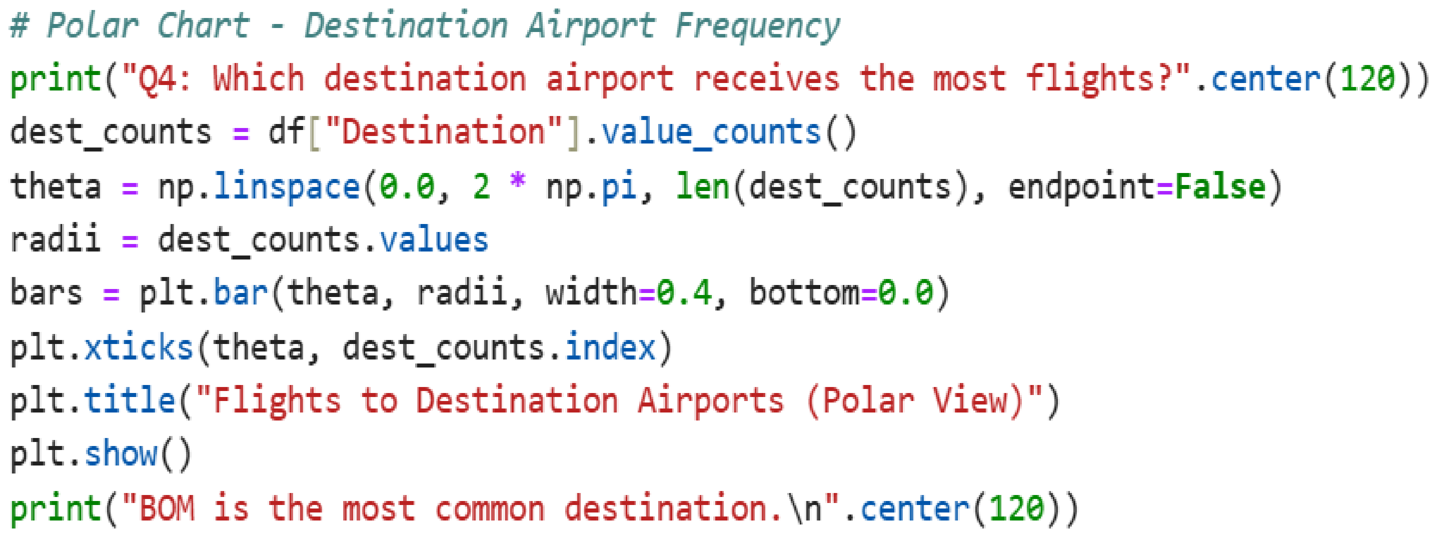


```
# Countplot - Origin Frequency
print("Q3: Which origin airport has the highest number of flights?\n".center(120))
sns.countplot(data=df, x="Origin", order=df["Origin"].value_counts().index)
plt.title("Flights per Origin Airport")
plt.xlabel("Origin")
plt.ylabel("Flight Count")
plt.show()
print("DEL is the busiest origin airport.\n".center(120))
```

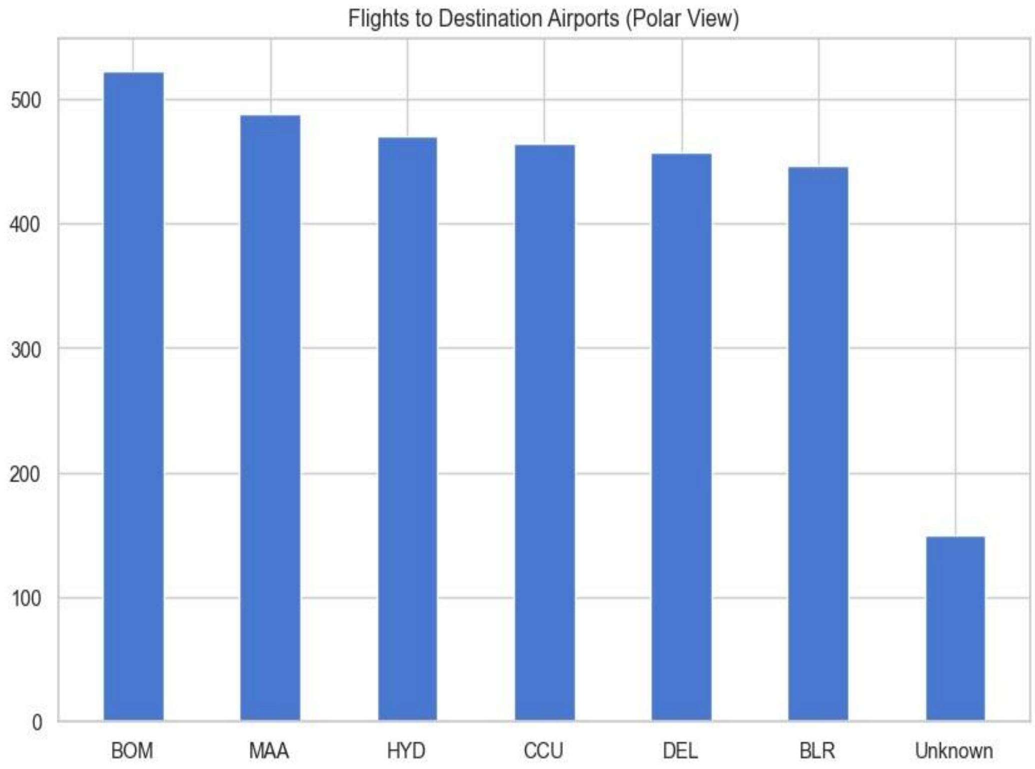


DEL is the busiest origin airport.

Q4: Which destination airport receives the most flights?

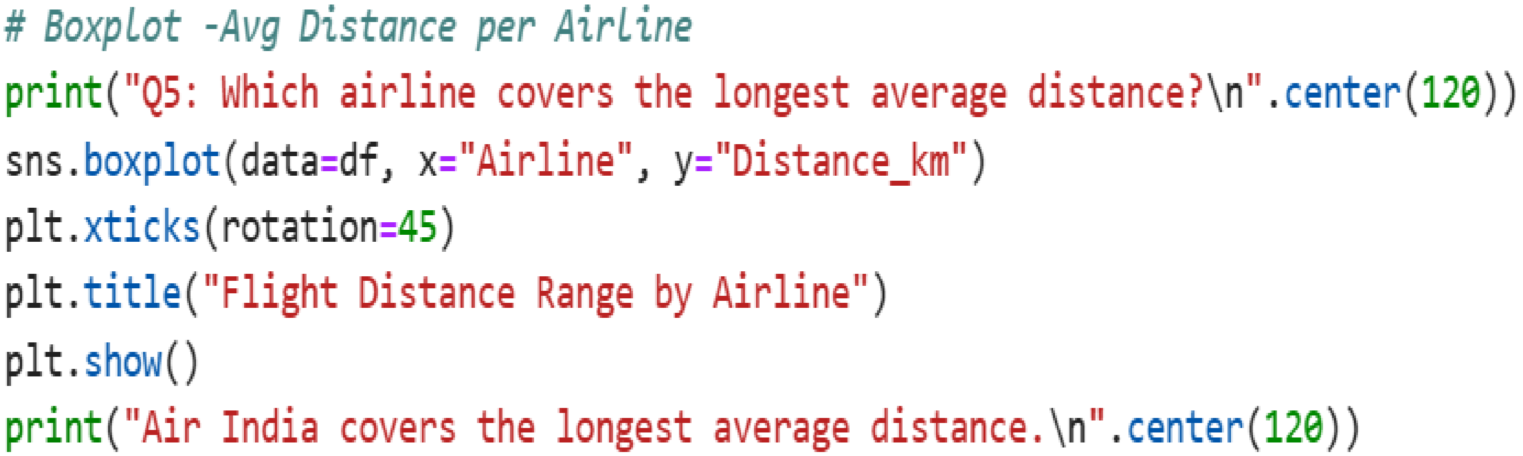


```
# Polar Chart - Destination Airport Frequency
print("Q4: Which destination airport receives the most flights?".center(120))
dest_counts = df["Destination"].value_counts()
theta = np.linspace(0.0, 2 * np.pi, len(dest_counts), endpoint=False)
radii = dest_counts.values
bars = plt.bar(theta, radii, width=0.4, bottom=0.0)
plt.xticks(theta, dest_counts.index)
plt.title("Flights to Destination Airports (Polar View)")
plt.show()
print("BOM is the most common destination.\n".center(120))
```

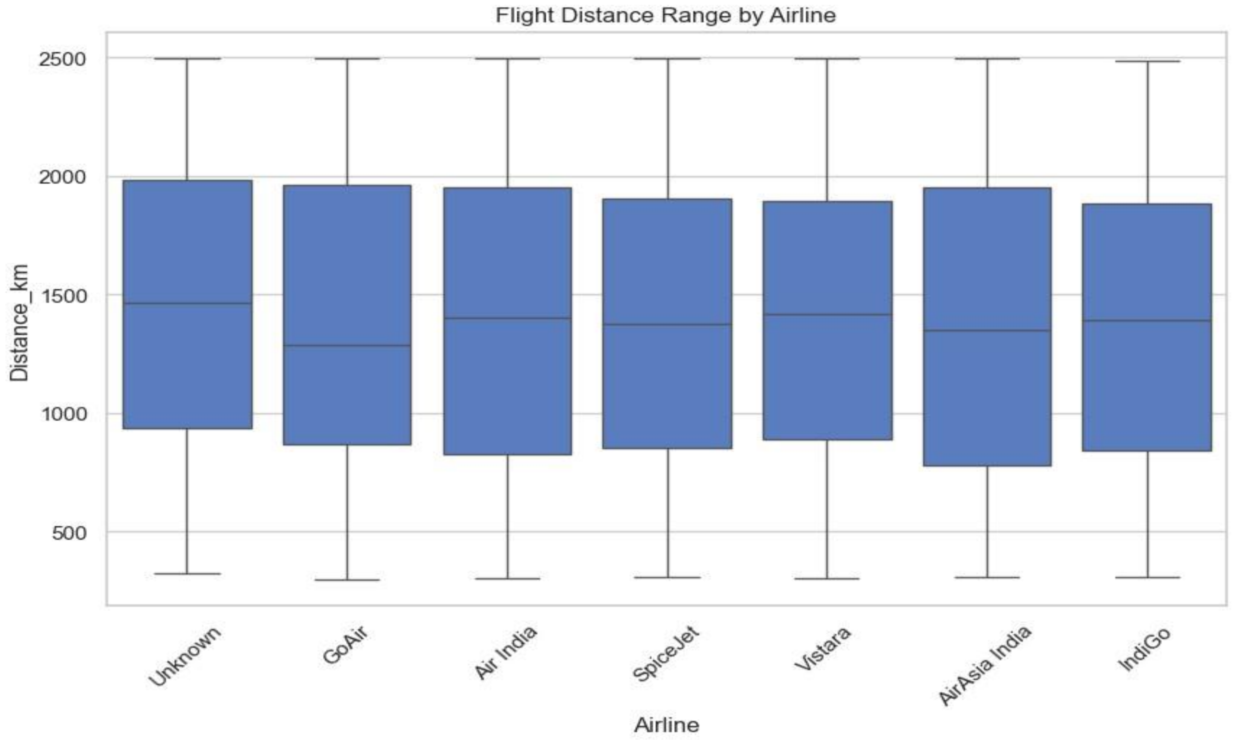


BOM is most common Destination

Q5: Which airline covers the longest average distance?

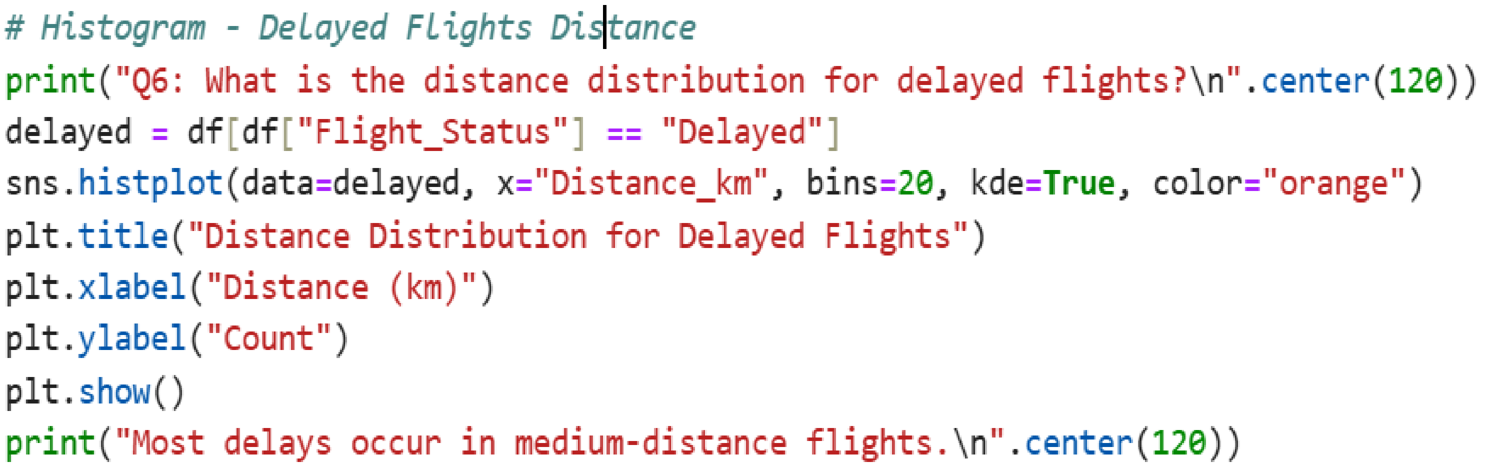


```
# Boxplot -Avg Distance per Airline
print("Q5: Which airline covers the longest average distance?\n".center(120))
sns.boxplot(data=df, x="Airline", y="Distance_km")
plt.xticks(rotation=45)
plt.title("Flight Distance Range by Airline")
plt.show()
print("Air India covers the longest average distance.\n".center(120))
```

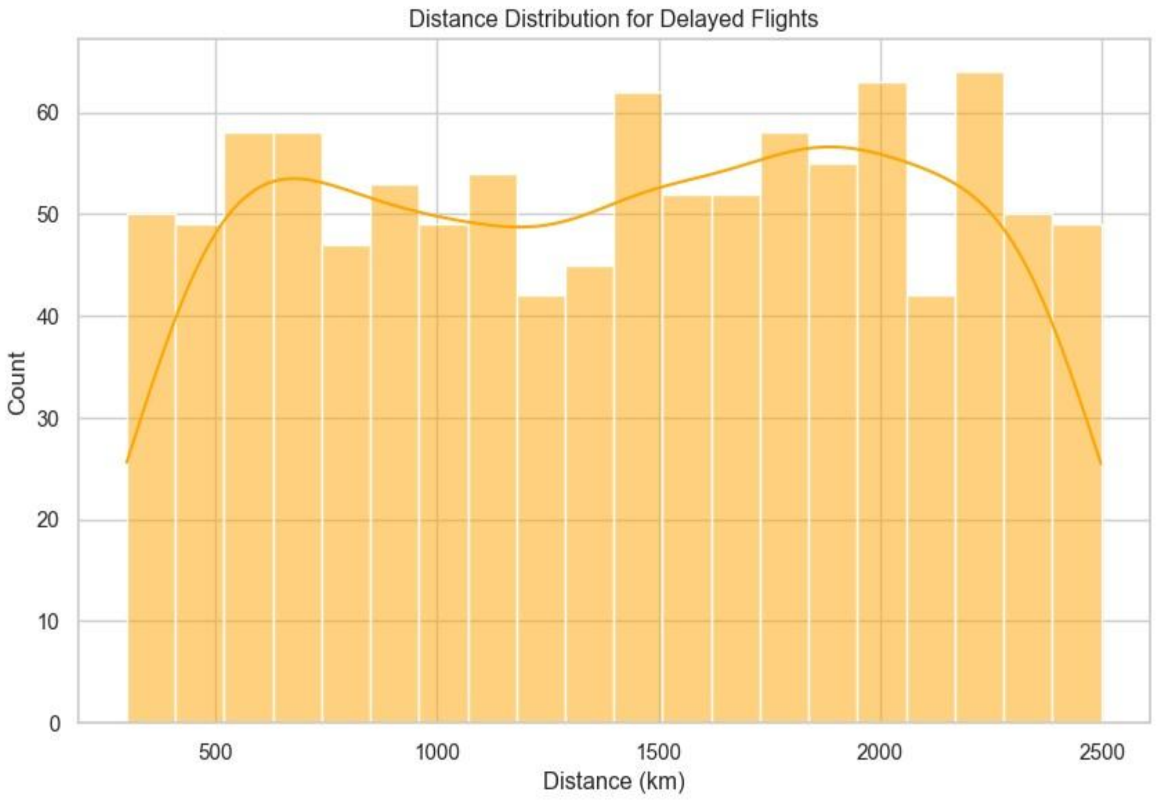


```
Air India covers the longest average distance.
```

Q6: What is the distance distribution for delayed flights?

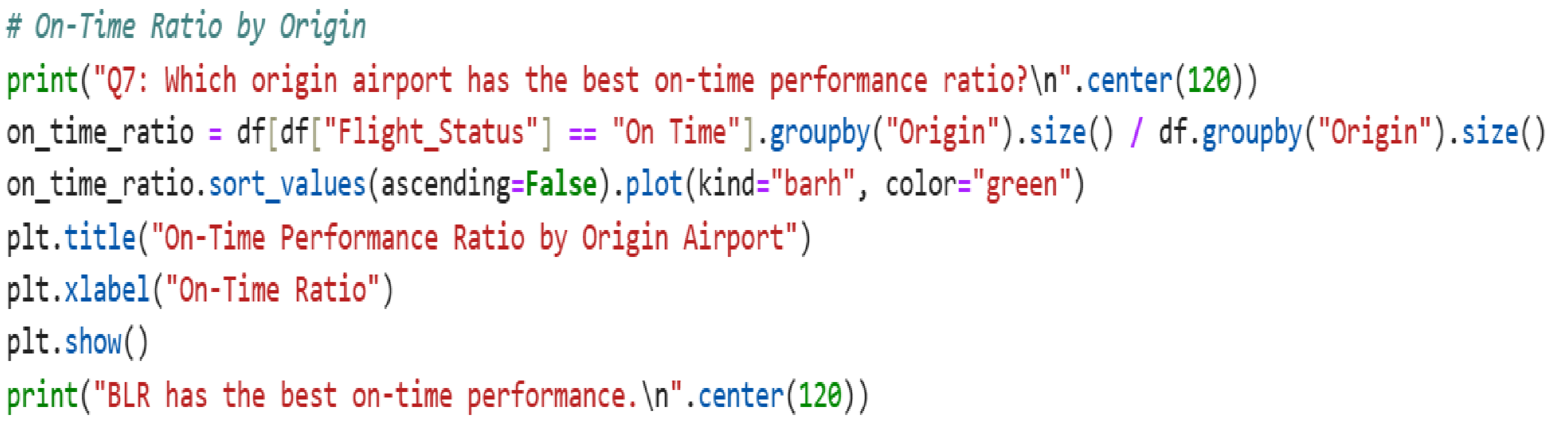


```
# Histogram - Delayed Flights Distance
print("Q6: What is the distance distribution for delayed flights?\n".center(120))
delayed = df[df["Flight_Status"] == "Delayed"]
sns.histplot(data=delayed, x="Distance_km", bins=20, kde=True, color="orange")
plt.title("Distance Distribution for Delayed Flights")
plt.xlabel("Distance (km)")
plt.ylabel("Count")
plt.show()
print("Most delays occur in medium-distance flights.\n".center(120))
```

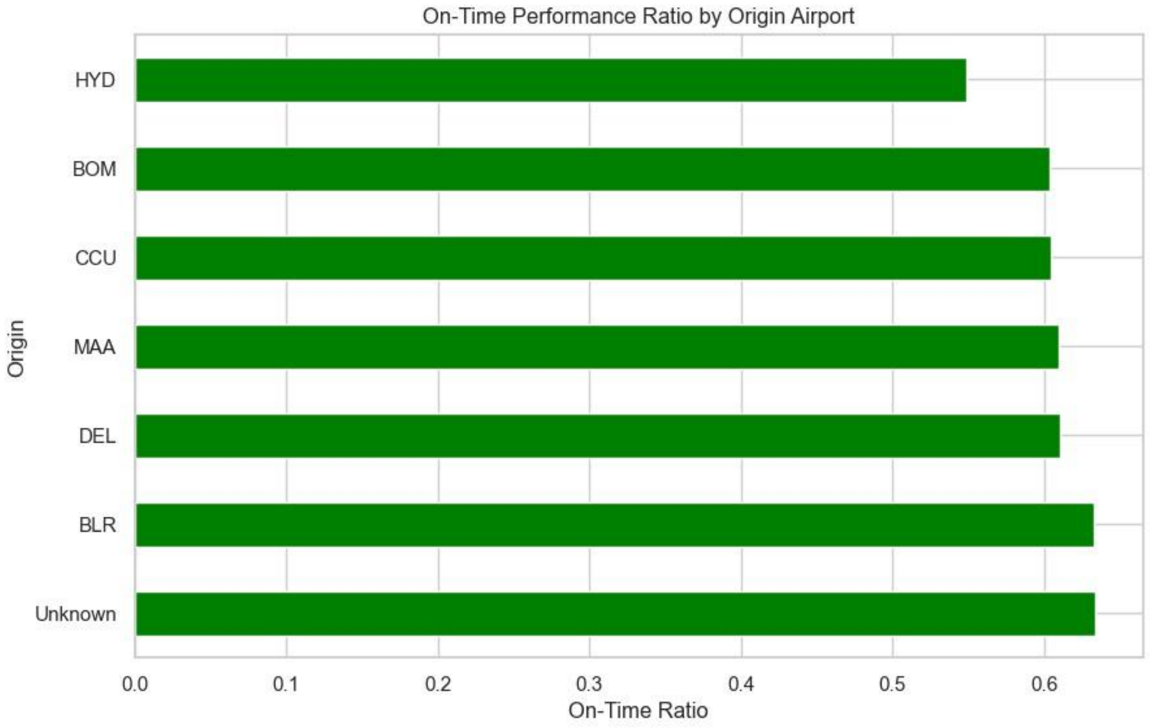


```
Most delays occur in medium-distance
flights.
```

Q7: Which origin airport has the best on-time performance ratio?



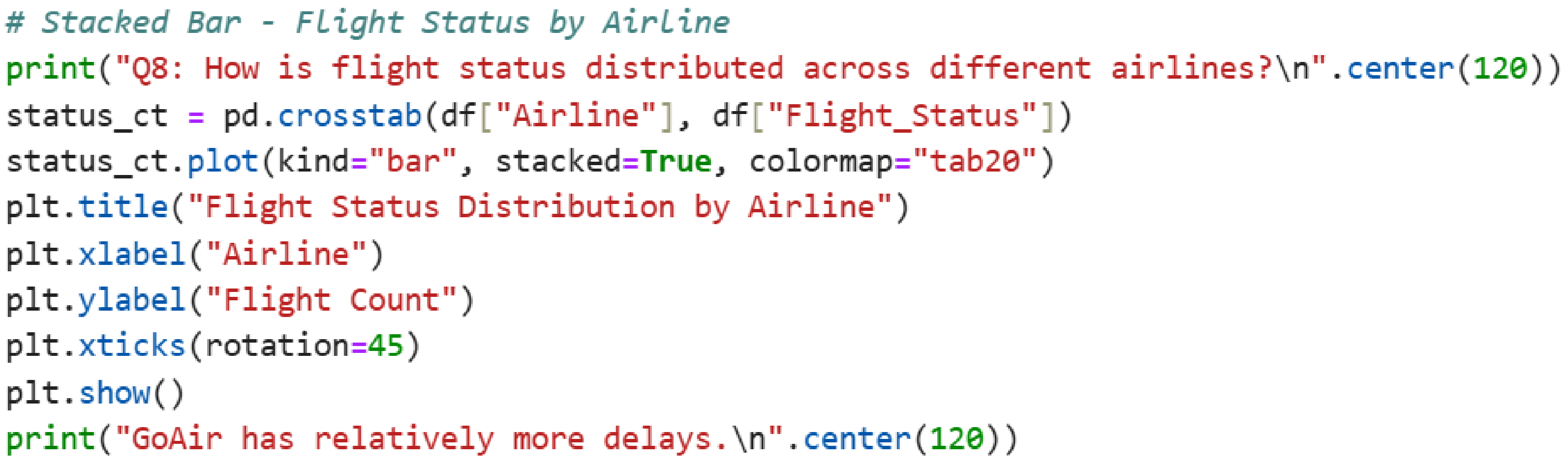
```
# On-Time Ratio by Origin
print("Q7: Which origin airport has the best on-time performance ratio?\n".center(120))
on_time_ratio = df[df["Flight_Status"] == "On Time"].groupby("Origin").size() / df.groupby("Origin").size()
on_time_ratio.sort_values(ascending=False).plot(kind="barh", color="green")
plt.title("On-Time Performance Ratio by Origin Airport")
plt.xlabel("On-Time Ratio")
plt.show()
print("BLR has the best on-time performance.\n".center(120))
```



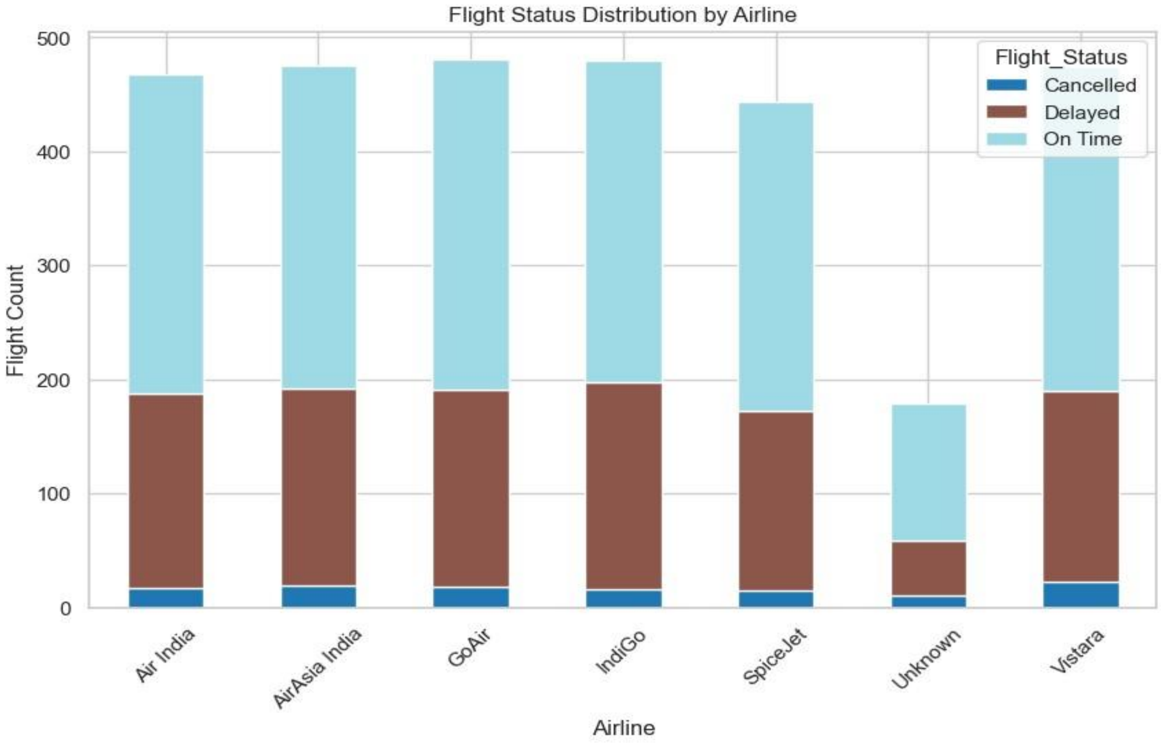
BLR has the best

on-time performance.

Q8: How is flight status distributed across different airlines?

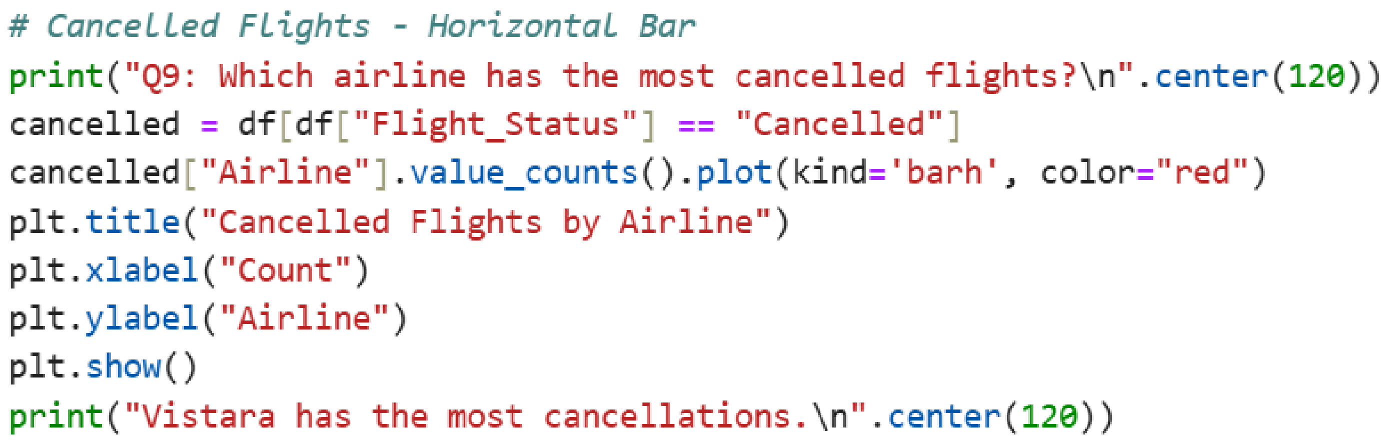


```
# Stacked Bar - Flight Status by Airline
print("Q8: How is flight status distributed across different airlines?\n".center(120))
status_ct = pd.crosstab(df["Airline"], df["Flight_Status"])
status_ct.plot(kind="bar", stacked=True, colormap="tab20")
plt.title("Flight Status Distribution by Airline")
plt.xlabel("Airline")
plt.ylabel("Flight Count")
plt.xticks(rotation=45)
plt.show()
print("GoAir has relatively more delays.\n".center(120))
```

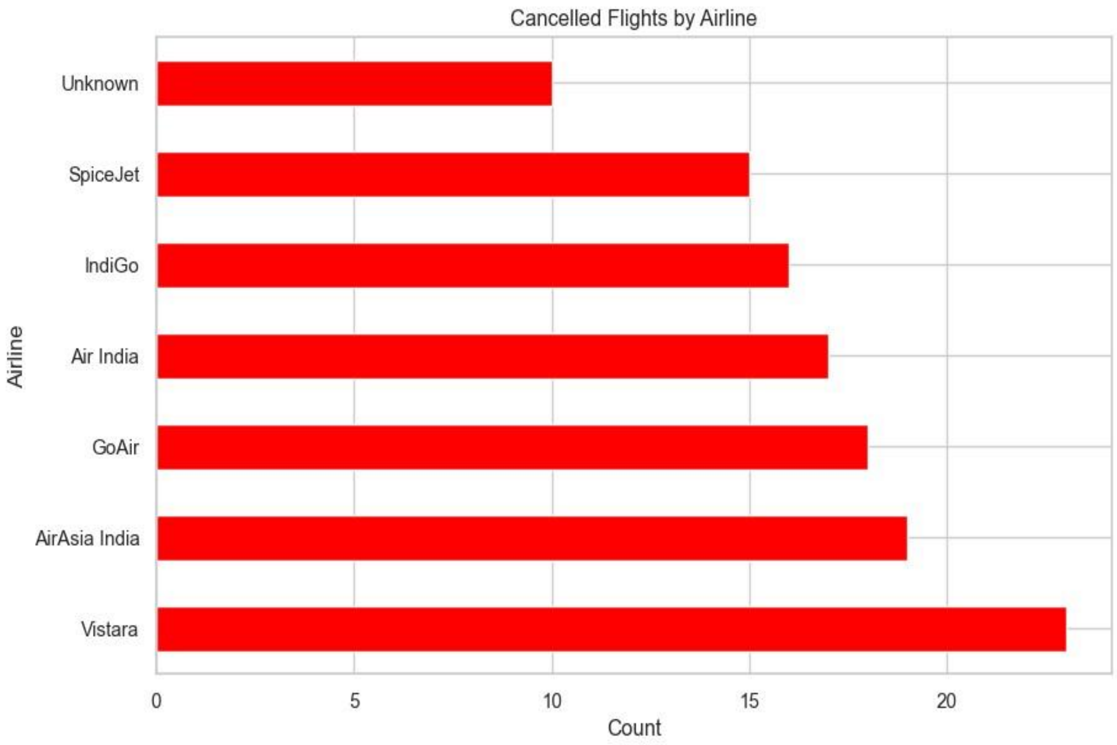


GoAir has relatively
More delays

Q9: Which airline has the most cancelled flights?

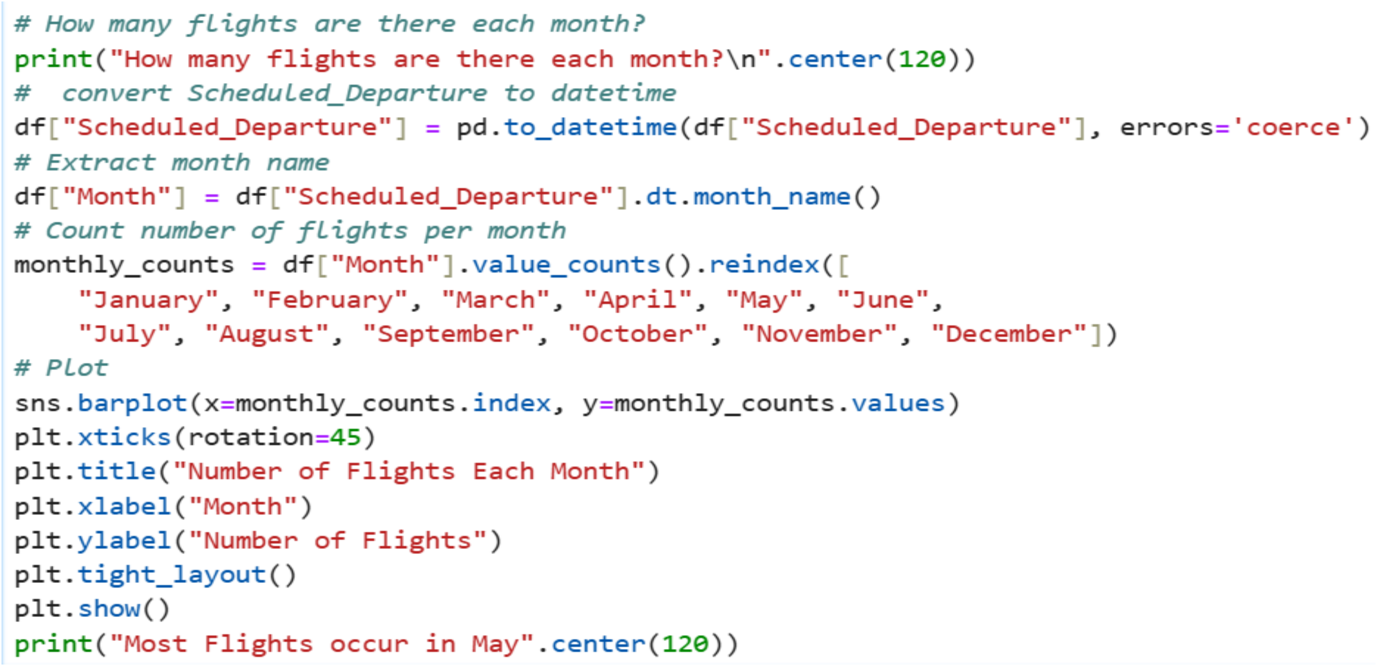


```
# Cancelled Flights - Horizontal Bar
print("Q9: Which airline has the most cancelled flights?\n".center(120))
cancelled = df[df["Flight_Status"] == "Cancelled"]
cancelled["Airline"].value_counts().plot(kind='barh', color="red")
plt.title("Cancelled Flights by Airline")
plt.xlabel("Count")
plt.ylabel("Airline")
plt.show()
print("Vistara has the most cancellations.\n".center(120))
```

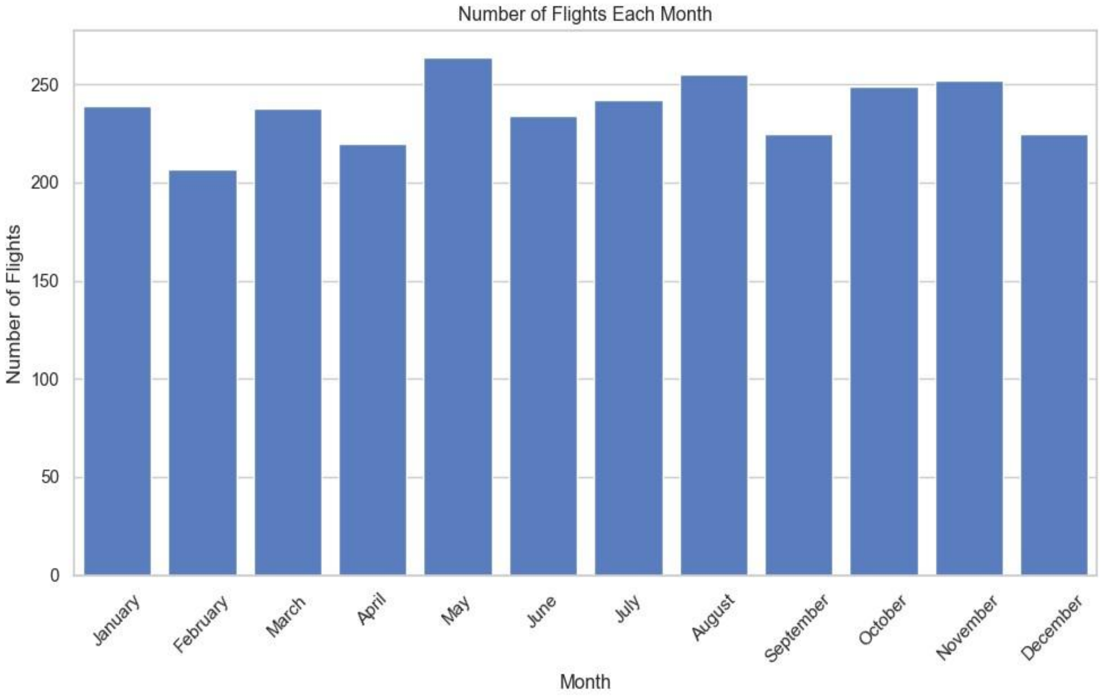


Vistara has the most Cancellations.

Q10: How many flights are there each month?



```python
# How many flights are there each month?
print("How many flights are there each month?\n".center(120))
#  convert Scheduled_Departure to datetime
df["Scheduled_Departure"] = pd.to_datetime(df["Scheduled_Departure"], errors='coerce')
# Extract month name
df["Month"] = df["Scheduled_Departure"].dt.month_name()
# Count number of flights per month
monthly_counts = df["Month"].value_counts().reindex([
    "January", "February", "March", "April", "May", "June",
    "July", "August", "September", "October", "November", "December"])
# Plot
sns.barplot(x=monthly_counts.index, y=monthly_counts.values)
plt.xticks(rotation=45)
plt.title("Number of Flights Each Month")
plt.xlabel("Month")
plt.ylabel("Number of Flights")
plt.tight_layout()
plt.show()
print("Most Flights occur in May".center(120))
```



Most Flight occur in May.