AVIATION ACCIDENTS DATA CLEANSING & ANALYSIS

Introduction

This project aims at cleaning and analyzing data for aviation accidents that occurred in the United States, all its territories as well as international waters. The analysis will derive insights that will help one of my clients to make a decision on the type of aircrafts to purchase.

Objectives

- Cleaning the data
- Analysing the data and getting insights
- Determining which aircrafts has low risk
- Presenting the findings and actionable insights to the client

Data content & Source

The aviation accidents data contains information about aircraft accidents in USA, its territories and international waters. It is contained in 2 CSV files downloaded from Kaggle.com (URL: https://www.kaggle.com/datasets/khsamaha/aviation-accident-database-synopses?select=AviationData.csv)

Data Analysis Methods

The data is analyzed using python libraries including:

- Pandas: Data manipulation and analysis
- Matplotlib: Data visualization
- Seaborn: Statistical plotting

In [1]:
```
#Importing the necessary python libraries
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

DATA CLEANSING

In [2]:
```
#Importing and displaying the first data set
df1=pd.read_csv("AviationData.csv",encoding='cp1252',low_memory=False)
df1.head(2)
```

Out[2]:

	Event.Id	Investigation.Type	Accident.Number	Event.Date	Location	Country	Latitude	L
0	20001218X45444	Accident	SEA87LA080	1948-10-24	MOOSE CREEK, ID	United States	NaN	

	Event.Id	Investigation.Type	Accident.Number	Event.Date	Location	Country	Latitude	L
1	20001218X45447	Accident	LAX94LA336	1962-07-19	BRIDGEPORT, CA	United States	NaN	

2 rows × 31 columns

In [3]:
```
#Importing and displaying the second data set
df2=pd.read_csv("USState_Codes.csv",encoding='cp1252',low_memory=False)
df2.head(2)
```

Out[3]:

	US_State	Abbreviation
0	Alabama	AL
1	Alaska	AK

In [4]:
```
#Checking the dimensions of the first data sets
df1.shape
```

Out[4]: (88889, 31)

In [5]:
```
#Checking the dimensions of the second data sets
df2.shape
```

Out[5]: (62, 2)

In [6]:
```
#Getting a summary of the first data set
df1.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 88889 entries, 0 to 88888
Data columns (total 31 columns):
 #   Column                  Non-Null Count  Dtype
---  ------                  --------------  -----
 0   Event.Id                88889 non-null  object
 1   Investigation.Type      88889 non-null  object
 2   Accident.Number         88889 non-null  object
 3   Event.Date              88889 non-null  object
 4   Location                88837 non-null  object
 5   Country                 88663 non-null  object
 6   Latitude                34382 non-null  object
 7   Longitude               34373 non-null  object
 8   Airport.Code            50249 non-null  object
 9   Airport.Name            52790 non-null  object
 10  Injury.Severity         87889 non-null  object
 11  Aircraft.damage         85695 non-null  object
 12  Aircraft.Category       32287 non-null  object
 13  Registration.Number     87572 non-null  object
 14  Make                    88826 non-null  object
 15  Model                   88797 non-null  object
 16  Amateur.Built           88787 non-null  object
 17  Number.of.Engines       82805 non-null  float64
 18  Engine.Type             81812 non-null  object
 19  FAR.Description         32023 non-null  object
 20  Schedule                12582 non-null  object
 21  Purpose.of.flight       82697 non-null  object
 22  Air.carrier             16648 non-null  object
```

```
 23  Total.Fatal.Injuries    77488 non-null  float64
 24  Total.Serious.Injuries  76379 non-null  float64
 25  Total.Minor.Injuries    76956 non-null  float64
 26  Total.Uninjured         82977 non-null  float64
 27  Weather.Condition       84397 non-null  object
 28  Broad.phase.of.flight   61724 non-null  object
 29  Report.Status           82508 non-null  object
 30  Publication.Date        75118 non-null  object
dtypes: float64(5), object(26)
memory usage: 21.0+ MB
```

In [7]:
```
#Getting a summary of the second data set
df2.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 62 entries, 0 to 61
Data columns (total 2 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   US_State      62 non-null     object
 1   Abbreviation  62 non-null     object
dtypes: object(2)
memory usage: 1.1+ KB
```

In [8]:
```
# getting the percentage of null values in each column (first data set)
percent_missing = df1.isnull().sum() * 100 / len(df1)
percent_missing
```

Out[8]:
```
Event.Id                  0.000000
Investigation.Type        0.000000
Accident.Number           0.000000
Event.Date                0.000000
Location                  0.058500
Country                   0.254250
Latitude                 61.320298
Longitude                61.330423
Airport.Code             43.469946
Airport.Name             40.611324
Injury.Severity           1.124999
Aircraft.damage           3.593246
Aircraft.Category        63.677170
Registration.Number       1.481623
Make                      0.070875
Model                     0.103500
Amateur.Built             0.114750
Number.of.Engines         6.844491
Engine.Type               7.961615
FAR.Description          63.974170
Schedule                 85.845268
Purpose.of.flight         6.965991
Air.carrier              81.271023
Total.Fatal.Injuries     12.826109
Total.Serious.Injuries   14.073732
Total.Minor.Injuries     13.424608
Total.Uninjured           6.650992
Weather.Condition         5.053494
Broad.phase.of.flight    30.560587
Report.Status             7.178616
Publication.Date         15.492356
dtype: float64
```

In [9]:
```
# getting the percentage of null values in each column (second data set)
percent_missing = df2.isnull().sum() * 100 / len(df2)
percent_missing
```

Out[9]:
```
US_State        0.0
Abbreviation    0.0
dtype: float64
```

In [10]:
```
#dropping columns with more than 30% of missing data
threshold = 0.3
min_count = int((1 - threshold) * len(df1))
df1 = df1.dropna(axis=1, thresh=min_count)
```

In [11]:
```
# getting the percentage of null values in the new data set
percent_missing = df1.isnull().sum() * 100 / len(df1)
percent_missing
```

Out[11]:
```
Event.Id                   0.000000
Investigation.Type         0.000000
Accident.Number            0.000000
Event.Date                 0.000000
Location                   0.058500
Country                    0.254250
Injury.Severity            1.124999
Aircraft.damage            3.593246
Registration.Number        1.481623
Make                       0.070875
Model                      0.103500
Amateur.Built              0.114750
Number.of.Engines          6.844491
Engine.Type                7.961615
Purpose.of.flight          6.965991
Total.Fatal.Injuries      12.826109
Total.Serious.Injuries    14.073732
Total.Minor.Injuries      13.424608
Total.Uninjured            6.650992
Weather.Condition          5.053494
Report.Status              7.178616
Publication.Date          15.492356
dtype: float64
```

In [12]:
```
#checking the dimension of the new data frame 1
df1.shape
```

Out[12]:
```
(88889, 22)
```

In [13]:
```
#check for duplicate
df1_duplicates=df1.duplicated()
df1_duplicates
```

Out[13]:
```
0        False
1        False
2        False
3        False
4        False
         ...
88884    False
88885    False
88886    False
88887    False
88888    False
Length: 88889, dtype: bool
```

In [14]:
```
#Replace full stop with hivens on column names in df1
df1.columns = df1.columns.str.replace('.', '_')
```

In [15]:
```
#displaying columns
print(list(df1.columns))
```

```
['Event_Id', 'Investigation_Type', 'Accident_Number', 'Event_Date', 'Location', 'Countr
y', 'Injury_Severity', 'Aircraft_damage', 'Registration_Number', 'Make', 'Model', 'Amate
ur_Built', 'Number_of_Engines', 'Engine_Type', 'Purpose_of_flight', 'Total_Fatal_Injurie
s', 'Total_Serious_Injuries', 'Total_Minor_Injuries', 'Total_Uninjured', 'Weather_Condit
ion', 'Report_Status', 'Publication_Date']
```

In [16]:
```
#dropping other unecessary columns
df1 = df1.drop(['Event_Id', 'Accident_Number','Registration_Number','Publication_Date']
```

In [17]:
```
#displaying remaining columns
print(list(df1.columns))
```

```
['Investigation_Type', 'Event_Date', 'Location', 'Country', 'Injury_Severity', 'Aircraft
_damage', 'Make', 'Model', 'Amateur_Built', 'Number_of_Engines', 'Engine_Type', 'Purpose
_of_flight', 'Total_Fatal_Injuries', 'Total_Serious_Injuries', 'Total_Minor_Injuries',
'Total_Uninjured', 'Weather_Condition', 'Report_Status']
```

In [18]:
```
#convert Event_Date column to datetime
df1['Event_Date'] = pd.to_datetime(df1['Event_Date'])
```

In [19]:
```
#dropping rows with missing entries
df1 = df1.dropna()
```

In [20]:
```
#confirming there are no rows with misisng numbers
percent_missing = df1.isnull().sum() * 100 / len(df1)
percent_missing
```

Out[20]:
```
Investigation_Type        0.0
Event_Date                0.0
Location                  0.0
Country                   0.0
Injury_Severity           0.0
Aircraft_damage           0.0
Make                      0.0
Model                     0.0
Amateur_Built             0.0
Number_of_Engines         0.0
Engine_Type               0.0
Purpose_of_flight         0.0
Total_Fatal_Injuries      0.0
Total_Serious_Injuries    0.0
Total_Minor_Injuries      0.0
Total_Uninjured           0.0
Weather_Condition         0.0
Report_Status             0.0
dtype: float64
```

In [21]:
```
#confirming number of remaining rows and columns
df1.shape
```

Out[21]:
```
(63085, 18)
```

In [22]:
```
#Removing white spaces in column 'Location'
df1['Location'] = df1['Location'].str.replace(' ', '')
```

In [23]:
```
#Coming up with a column 'location_code' from location
df1['Location_code']=df1['Location'].str.split(',').str[1].str.strip()
```

In [24]:
```
#displaying columns
print(list(df1.columns))
```

```
['Investigation_Type', 'Event_Date', 'Location', 'Country', 'Injury_Severity', 'Aircraft
_damage', 'Make', 'Model', 'Amateur_Built', 'Number_of_Engines', 'Engine_Type', 'Purpose
_of_flight', 'Total_Fatal_Injuries', 'Total_Serious_Injuries', 'Total_Minor_Injuries',
'Total_Uninjured', 'Weather_Condition', 'Report_Status', 'Location_code']
```

In [25]:
```
#displaying the first 2 rows
df1.head(1)
```

Out[25]:

	Investigation_Type	Event_Date	Location	Country	Injury_Severity	Aircraft_damage	Make	M
0	Accident	1948-10-24	MOOSECREEK,ID	United States	Fatal(2)	Destroyed	Stinson	

In [26]:
```
#checking what percentage of data is from United States
Countrycount = df1.groupby('Country').size()
proportion = (Countrycount / Countrycount.sum()) * 100
print(proportion)
```

```
Country
ATLANTIC OCEAN     0.085599
Afghanistan        0.001585
American Samoa     0.004755
Angola             0.003170
Antarctica         0.001585
                    ...
UN                 0.007926
United Kingdom     0.001585
United States     99.131331
Venezuela          0.003170
West Indies        0.006341
Length: 81, dtype: float64
```

In [27]:
```
#dropping other countries given that USA represent 99% of the data
df1 = df1[df1['Country'] == 'United States']
```

In [28]:
```
#checking what percentage of data is from United States
Countrycount = df1.groupby('Country').size()
proportion = (Countrycount / Countrycount.sum()) * 100
print(proportion)
```

```
Country
United States    100.0
dtype: float64
```

In [29]:
```
#changing the name of 'Abbreviation' column to Location_code
df2 = df2.rename(columns={'Abbreviation': 'Location_code'})
```

In [30]:
```
# Merge the 2 data sets using the 'Location_code' column
df = pd.merge(df1, df2, on='Location_code')
#displaying columns
print(list(df.columns))
```

```
['Investigation_Type', 'Event_Date', 'Location', 'Country', 'Injury_Severity', 'Aircraft
_damage', 'Make', 'Model', 'Amateur_Built', 'Number_of_Engines', 'Engine_Type', 'Purpose
_of_flight', 'Total_Fatal_Injuries', 'Total_Serious_Injuries', 'Total_Minor_Injuries',
'Total_Uninjured', 'Weather_Condition', 'Report_Status', 'Location_code', 'US_State']
```

In [31]:
```
# Displaying the columns and their data types
print(df.dtypes)
```

```
Investigation_Type               object
Event_Date                 datetime64[ns]
Location                         object
Country                          object
Injury_Severity                  object
Aircraft_damage                  object
Make                             object
Model                            object
Amateur_Built                    object
Number_of_Engines               float64
Engine_Type                      object
Purpose_of_flight                object
Total_Fatal_Injuries            float64
Total_Serious_Injuries          float64
Total_Minor_Injuries            float64
Total_Uninjured                 float64
Weather_Condition                object
Report_Status                    object
Location_code                    object
US_State                         object
dtype: object
```

In [32]:
```
#For data uniformity capitalize data under 'Make' and 'Model'
df['Make'] = df['Make'].str.upper()
df['Model'] = df['Model'].str.upper()
```

In [33]:
```
#create a column called year from 'Event_Date' column
df['Event_Date'] = pd.to_datetime(df['Event_Date'], format='%Y-%m-%d')
df['Year']=df['Event_Date'].dt.strftime('%Y')
```

In [34]:
```
# Specify the file path to eport df
file_path = 'C:/Users/jgatonye/Documents/DS_COURSE/PHASE1/PROJECT1/Project-Phase-1/df.c
# Export to CSV
df.to_csv(file_path, index=False)
```

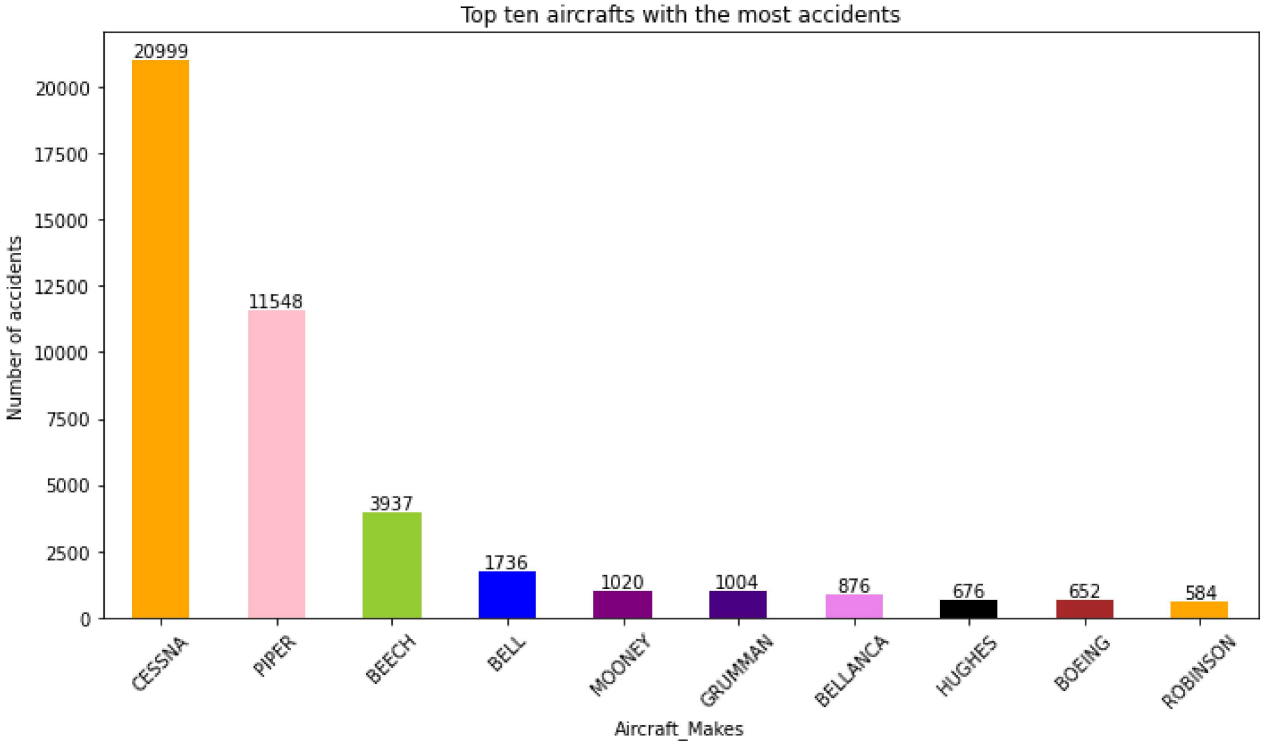
Data Analysis & Visualization

In [35]:
```
#Top 10 most used aircraft makes by percentage-We will assume that aircraft makes with
top_10_makes = df['Make'].value_counts(normalize=True).head(10) * 100
print(top_10_makes)
```

```
CESSNA      33.603239
PIPER       18.479461
BEECH        6.300107
BELL         2.778000
MOONEY       1.632235
GRUMMAN      1.606631
BELLANCA     1.401802
HUGHES       1.081756
BOEING       1.043350
ROBINSON     0.934535
Name: Make, dtype: float64
```

Bar chart showing top ten aircrafts with the most accidents

In [36]:
```
# getting a count of every aircraft make and sorting top 10
top_ten_aircrafts_makes_accidents = df['Make'].value_counts().head(10)
# getting colours
colors = ['orange','pink','yellow' 'green', 'blue', 'purple','indigo','violet','black',
# Plotting the a bar graph of to 10 aircrafts
plt.figure(figsize=(10, 6))
bar=top_ten_aircrafts_makes_accidents.plot(kind='bar', color=colors)
#capture number of accidents on top of the bar
for p in bar.patches:
            bar.annotate(f'{int(p.get_height())}', (p.get_x() + p.get_width() / 2., p.g
                        ha='center', va='center', fontsize=10, color='black', xytext=(0
                        textcoords='offset points')

plt.title('Top ten aircrafts with the most accidents')
plt.xlabel('Aircraft_Makes')
plt.ylabel('Number of accidents')
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```



Deductions from the above chart:

- *Top 10 flown aircrafts in the USA include Cessna,Piper,Beech,Bell,Mooney,Grumman,Bellanca,Hughes,Boeing,Robinson*
- *Top 3 aircrafts with the most accidents include Cessna,Piper,Beech*
- *Top 3 aircrafts with the least accidents include Robinson,Boeing,Hughes*

A pivot table showing total number of accidents for top 10 aircrafts since 2010

In [37]:
```
#Generating new data set showing the total number of accidents for the top 10 aircrafts
df['Year'] = df['Year'].astype(int)
df_3 = df[df['Year'] > 2009]
# List of selected variables to include in the pivot
selected_variables = ['CESSNA', 'BOEING','PIPER','BEECH','BELL','MOONEY','GRUMMAN','BEL
# Filtering the data set
filtered_df = df_3[df_3['Make'].isin(selected_variables)]
# Create a pivot table to count accidents of each aircraft per 'year'
pivot = pd.pivot_table(filtered_df, index='Year', columns='Make', aggfunc='size', fill_
pivot
```

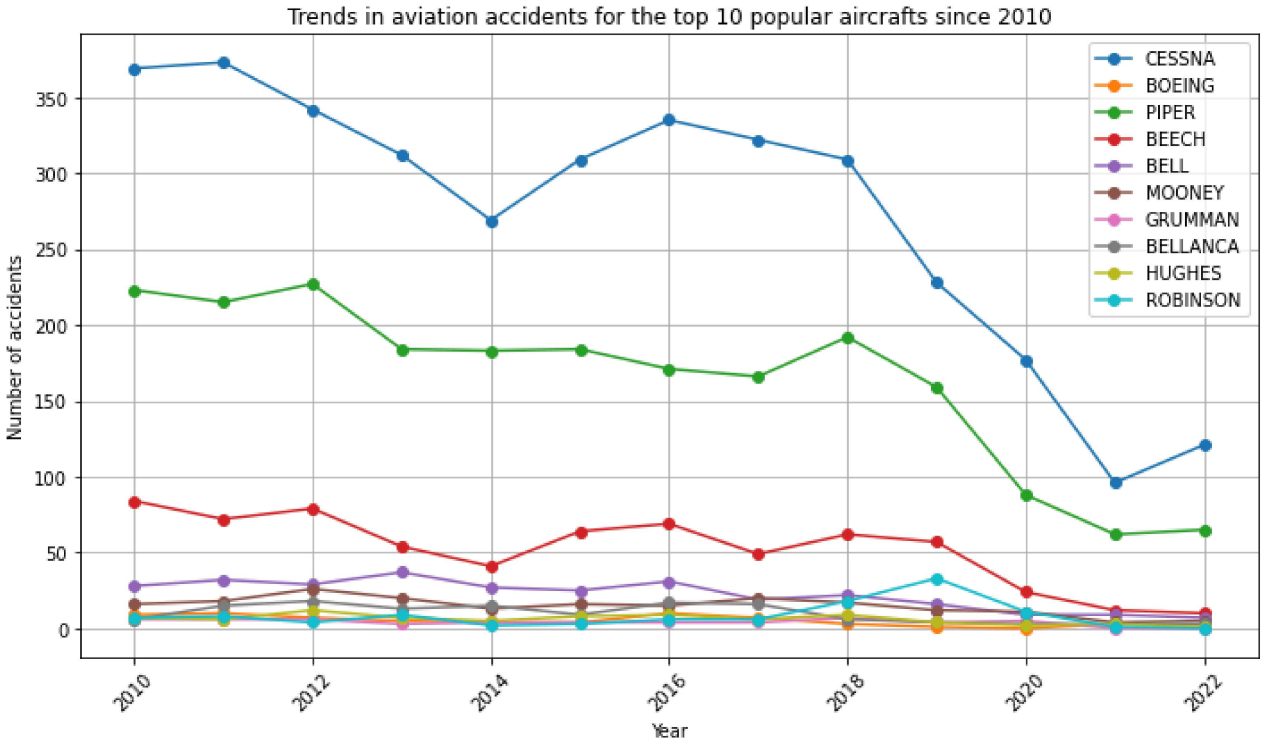
Out[37]:

Make	BEECH	BELL	BELLANCA	BOEING	CESSNA	GRUMMAN	HUGHES	MOONEY	PIPER	ROBINSON
Year										
2010	84	28	6	9	369	6	7	16	223	7
2011	72	32	15	10	373	6	6	18	215	8
2012	79	29	18	7	342	6	12	26	227	4
2013	54	37	13	5	312	3	7	20	184	9
2014	41	27	15	4	269	4	5	13	183	2
2015	64	25	9	4	309	4	8	16	184	3
2016	69	31	17	10	335	4	9	15	171	6
2017	49	19	16	7	322	4	6	20	166	6
2018	62	22	6	3	309	7	9	17	192	18
2019	57	16	4	1	228	4	4	12	159	33
2020	24	9	3	0	177	5	2	11	88	11
2021	12	9	3	3	96	0	3	4	62	1
2022	10	7	3	2	121	0	1	5	65	0

A line graph showing top ten aircrafts accidents trends since 2010

In [38]:
```
plt.figure(figsize=(10, 6))
plt.plot(pivot.index, pivot['CESSNA'], label='CESSNA', marker='o')
plt.plot(pivot.index, pivot['BOEING'], label='BOEING', marker='o')
plt.plot(pivot.index, pivot['PIPER'], label='PIPER', marker='o')
plt.plot(pivot.index, pivot['BEECH'], label='BEECH', marker='o')
plt.plot(pivot.index, pivot['BELL'], label='BELL', marker='o')
plt.plot(pivot.index, pivot['MOONEY'], label='MOONEY', marker='o')
plt.plot(pivot.index, pivot['GRUMMAN'], label='GRUMMAN', marker='o')
plt.plot(pivot.index, pivot['BELLANCA'], label='BELLANCA', marker='o')
plt.plot(pivot.index, pivot['HUGHES'], label='HUGHES', marker='o')
plt.plot(pivot.index, pivot['ROBINSON'], label='ROBINSON', marker='o')
```

```
# Adding labels and title
plt.xlabel('Year')
plt.ylabel('Number of accidents')
plt.title('Trends in aviation accidents for the top 10 popular aircrafts since 2010')
plt.legend()
plt.grid(True)
plt.xticks(rotation=45)  # Rotate year labels for better readability

# Show the plot
plt.tight_layout()
plt.show()
```



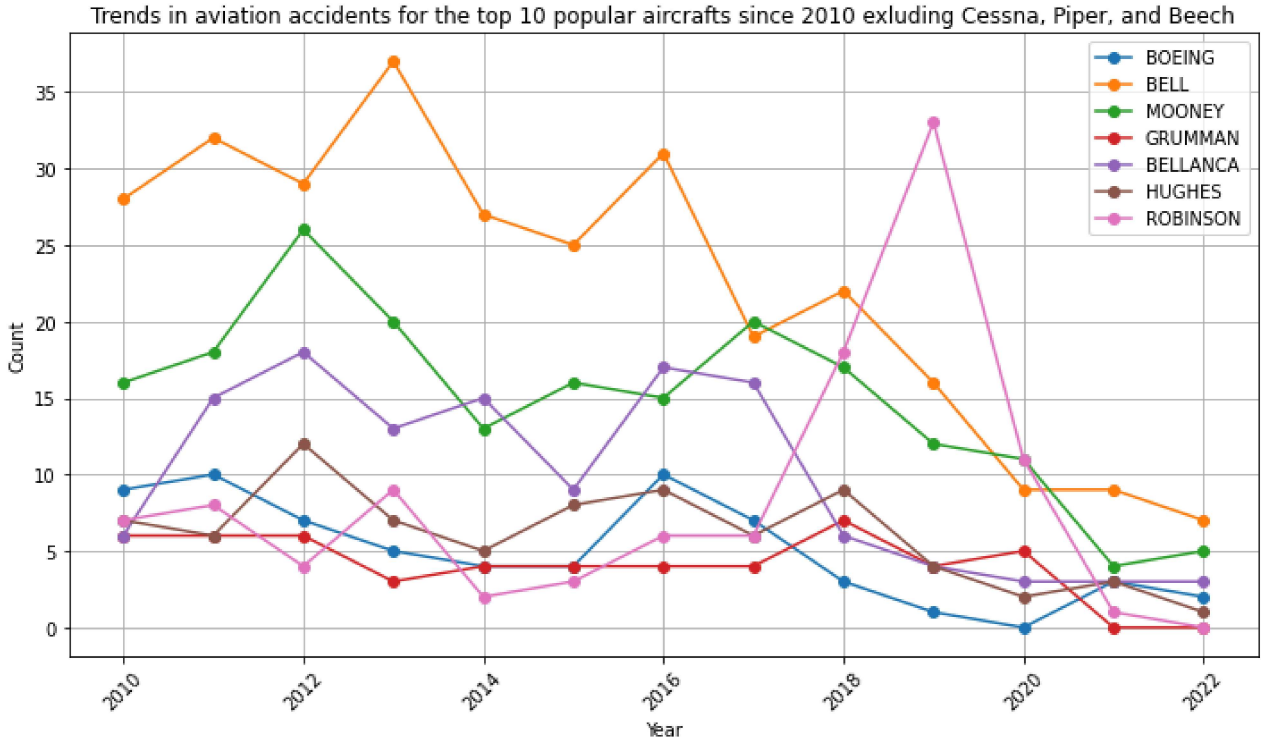
Deductions from the above chart:

- *Aviation accidents have been reducing over time probably due to improvement in technology, training and regulations*
- *Cessna, Mooney and Piper aircraft accidents were in a downward trend but went up in 2022 contrary to the downward trend sustained by the other aircrafts*

In [39]:
```
##**We exclude top 3 aircrafts with the most accidents from the data for the line graph
```

In [40]:
```
plt.figure(figsize=(10, 6))
plt.plot(pivot.index, pivot['BOEING'], label='BOEING', marker='o')
plt.plot(pivot.index, pivot['BELL'], label='BELL', marker='o')
plt.plot(pivot.index, pivot['MOONEY'], label='MOONEY', marker='o')
plt.plot(pivot.index, pivot['GRUMMAN'], label='GRUMMAN', marker='o')
plt.plot(pivot.index, pivot['BELLANCA'], label='BELLANCA', marker='o')
plt.plot(pivot.index, pivot['HUGHES'], label='HUGHES', marker='o')
plt.plot(pivot.index, pivot['ROBINSON'], label='ROBINSON', marker='o')
# Adding labels and title
plt.xlabel('Year')
plt.ylabel('Count')
plt.title('Trends in aviation accidents for the top 10 popular aircrafts since 2010 exl
```

```
plt.legend()
plt.grid(True)
plt.xticks(rotation=45)  # Rotate year labels for better readability

# Show the plot
plt.tight_layout()
plt.show()
```



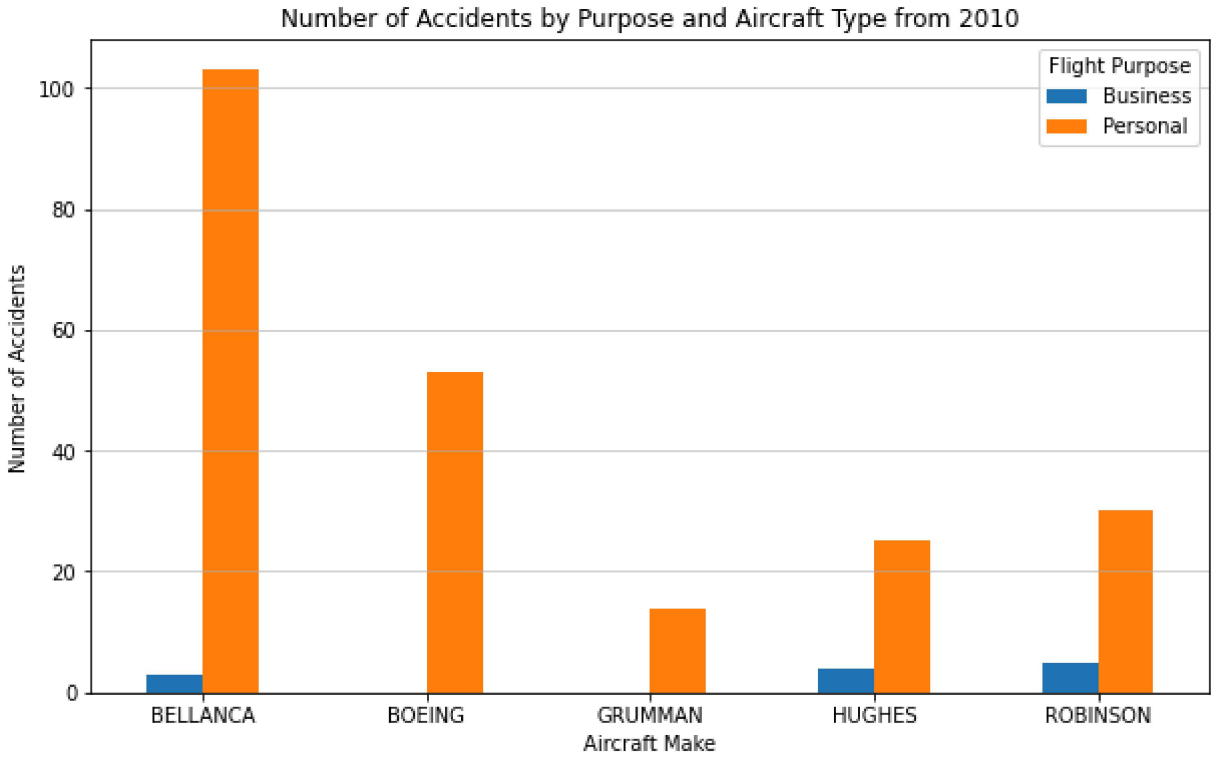
Deductions from the above chart:

- *Top 5 safest aircrafts with steadily decreasing number of accidents over the last 10 years include*
  1. *Robinson*
  2. *Grumman*
  3. *Hughes*
  4. *Boeing*
  5. *Bellanca*

A bar graph showing the number of accudents for the 5 safest aircrafts by purpose

In [41]:
```
# Generate data for the 5 aircrafts with the least accidents from 2010 henceforth for Bu
df4 = df[(df['Year'] >= 2010) & (df['Make'].isin(['ROBINSON', 'BOEING', 'HUGHES','BELLA

# Group the data by 'Make' and 'Purpose_of_flight' and then count the number of acciden
grouped_data = df4.groupby(['Make', 'Purpose_of_flight']).size().unstack()

# Plotting the bar chart
grouped_data.plot(kind='bar', figsize=(10, 6))

# Labels for the bar chart
plt.title('Number of Accidents by Purpose and Aircraft Type from 2010')
```

```
plt.xlabel('Aircraft Make')
plt.ylabel('Number of Accidents')
plt.xticks(rotation=0)
plt.legend(title="Flight Purpose")
plt.grid(axis='y', linestyle='-', alpha=0.7)
# Show the plot
plt.show()
```



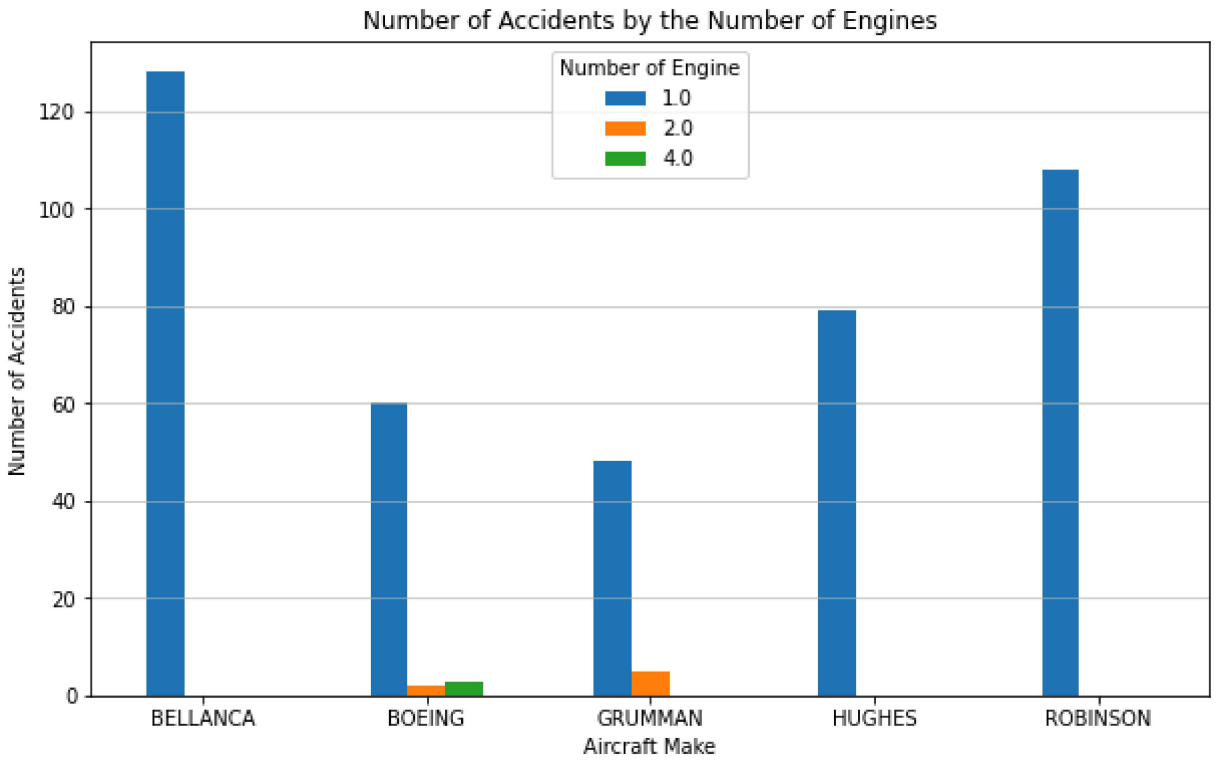
Deductions from the above chart:

1. *Boeing and Grumman Aircrafts are the safest for the business flights because they did not have any accidents from 2010*
2. Grumman are the safest for the personal flights-

In [42]:
```
##**A bar graph showing the number of accidents for the 5 safest aircrafts by the Numbe
```

In [43]:
```
# Generate data for the 5 aircrafts with the least accidents from 2010 henceforth for Bu
df4 = df[(df['Year'] >= 2010) & (df['Make'].isin(['ROBINSON', 'BOEING', 'HUGHES','BELLA

# Group the data by 'Make' and 'Purpose_of_flight' and then count the number of accident
grouped_data = df4.groupby(['Make', 'Number_of_Engines']).size().unstack()

# Plotting the bar chart
grouped_data.plot(kind='bar', figsize=(10, 6))

# Labels for the bar chart
plt.title('Number of Accidents by the Number of Engines')
plt.xlabel('Aircraft Make')
plt.ylabel('Number of Accidents')
plt.xticks(rotation=0)
plt.legend(title="Number of Engine")
plt.grid(axis='y', linestyle='-', alpha=0.7)
```

```
# Showing the chart
plt.show()
```



Deductions from the above chart:

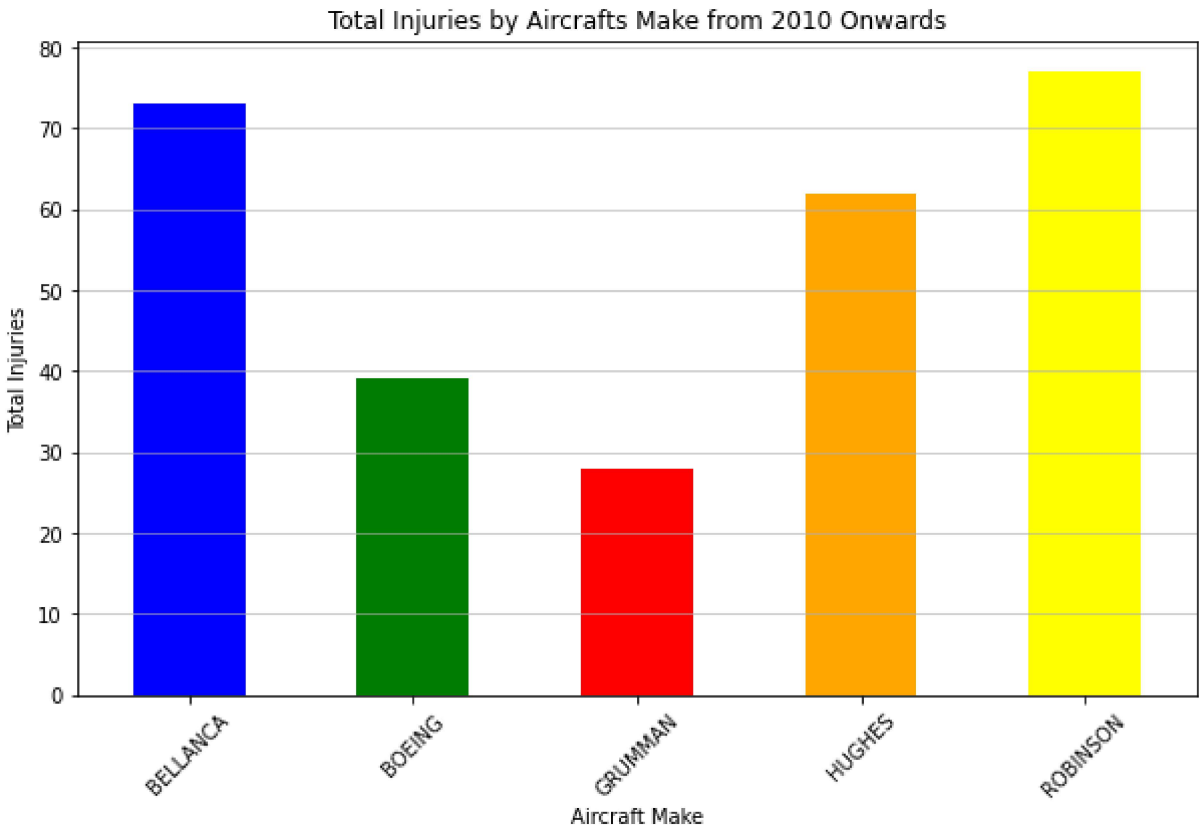
1. *The number of accidents reduces as the number of engines increases*

Plotting a graph that shows the Total Passangers' Injuries for the 5 safest aircrafts from 2010

In [44]:

```
# Include data from 2010
df = df[df['Year'] >= 2010]

# Computation of total injuries
df['Total_Injuries'] = df[['Total_Fatal_Injuries', 'Total_Serious_Injuries', 'Total_Min

# Group data by 'Make'
injuries_group = df.groupby('Make')['Total_Injuries'].sum()

# Including Least_Accidents_Aircrafts only
Least_Accidents_Aircrafts = ['ROBINSON', 'BOEING', 'HUGHES','BELLANCA','GRUMMAN']
injuries_group  = injuries_group [injuries_group .index.isin(Least_Accidents_Aircrafts)

# Plot the chart
plt.figure(figsize=(10, 6))
bars = injuries_group.plot(kind='bar', color=['blue', 'green', 'red','orange','yellow']
# Labels and title
plt.xlabel('Aircraft Make')
plt.ylabel('Total Injuries')
plt.title('Total Injuries by Aircrafts Make from 2010 Onwards')
plt.xticks(rotation=45)
plt.grid(axis='y', linestyle='-', alpha=0.7)
```

```
# Show the chart
plt.show()
```



Deductions from the above chart:

1. *Boeing and Grumman have the lowest number of injuries since 2010 meaning they are the top 2 safest aircrafts.*

CONCLUSION

1. Top 3 highest risk aircrafts with the most accidents include Cessna, Piper, Beech
2. Top 3 low risk aircrafts with the least accidents include Robinson, Boeing, Hughes
3. Aviation accidents have been reducing over time probably due to improvement in technology, training and regulations
4. Aircrafts with more than one engine are less likely to be involved in accidents
5. Boeing and Grumman Aircrafts are the safest for the business flights because they did not have any accidents from 2010
6. Grumman are the safest for the personal flights as they had the lowest number of accidents since 2010.
7. Boeing and Grumman have the lowest number of injuries since 2010 meaning they are the top 2 safest aircrafts.

RECOMMENDATION

From the analysis, it would be prudent to acquire the latest models of Boeing for business flights and Grumman for personal flights. Further, acquiring an aircraft with more than one engine significantly reduce the risk of accidents.