Problem Definition

Our company's expansion and diversification plans include venturing into the aviation industry to own and operate airplanes for commercial and private enterprises. A key preliminary step for this consideration is risk assesment for different aircrafts to advise which aircrafts pose the lowest risk for the intended business endeavor. This projects seeks to assess risk potential from analysis of aviation accident data from 1962 to 2023.

The primary objective of this exercise is to identify the lowest-risk aircraft for our company to purchase and operate. The following are some of the key considerations we expect to make:

- Historical accident trends by aircraft type
- Severity and frequency of accidents
- Factors contributing to the accidents e.g. weather, pilot error or mechanical failures
- Any correlations between operational risk and aircraft characteristics

Data Preprocessing

This section prepare the provided aviation data for analysis. We intend to do the following:

- Dataset Overview - Load and understand the data
- Handling Missing Values using derived domain knowledge and imputation
- Data Cleaning e.g. standardizing categorical values, deriving useful date data, removing duplicates etc

Dataset Overview

It is imperative for us to understand the aviation dataset first i.e.:

- The data structure e.g. available columns, data types and presence of missing values
- Identify useful columns to focus on

Data Understanding will prescribe subsequent steps. Simple cleaning procedures e.g. capitalization will be done on the spot (immediately a need is observed) to ensure they are not forgotten. Complex cleaning procedures will be done in the **Data Cleaning** subsection

Initializing Relevant Libraries

First, we initialize common libraries we project to utilize in this exercise

```
#Import libraries i'm likely to use upfront
import pandas as pd #To create and manipulate pandas dataframes
import seaborn as sns #To Facilitate visualizations
import matplotlib.pyplot as plt #To facilitate visualizations
import numpy as np #To facilitate mathematical calculations
from sklearn.preprocessing import LabelEncoder #Use this to encode categorical data
%matplotlib inline
```

Loading the dataset

Then we load the provided dataset

```
#Load the aviation data
#Understand the data and get a snapshot of the data
aviation_df = pd.read_csv('Aviation_Data.csv')
aviation_df.info()
```

```
<ipython-input-99-2bbd9ca9aaa1>:3: DtypeWarning: Columns (6,7,28) have mixed types. Spec
  aviation_df = pd.read_csv('Aviation_Data.csv')
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 90348 entries, 0 to 90347
Data columns (total 31 columns):
 #   Column                  Non-Null Count  Dtype
---  ------                  --------------  -----
 0   Event.Id                88889 non-null  object
 1   Investigation.Type      90348 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            50132 non-null  object
 9   Airport.Name            52704 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     87507 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             81793 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           82505 non-null  object
 30  Publication.Date        73659 non-null  object
dtypes: float64(5), object(26)
memory usage: 21.4+ MB
```

Data Understanding

Sampling a few columns to assess the various categorical data present will help us define how to clean the data, e.g.:

- Investigation.Type
- Injury.Severity
- Aircraft.damage
- Aircraft.Category
- Make
- Model
- Amateur.Built
- Engine.Type

[ ] ↳ *41 cells hidden*

Summary Of the Dataset Overview

The dataset contains 31 columns and 90348 rows (including the header columns). There are several missing values in different columns. The only column without any missing data is the 'Investigation.Type' column, and this can form a good place to start with the data cleaning exercise.

Further perusal of the data in Microsoft Excel gave some preliminary insights that can advise the data cleaning exercise:

- Where 'Event.Id' is blank, all the other columns are also blank. These could be deleted from the onset
- 'Event.Date' uses a YYYY-MM-DD format whereas 'Publication.Date' uses a DD-MM-YYYY format. It would be better to standardize the date formats
- 'Investigation.Type' seems to have 2 relevant values, i.e. "Accident" and "Incident". The rest of the values seem to be dates, and where it is a date, the rest of the columns are empty. Such

rows can also be deleted from the onset.

- In the column 'Injury.Severity', there are too many categories since the number of fatalities is appended beside the label 'Fatal'. This is repetitive since there is another independent column 'Total.Fatal.Injuries' that details the number of fatalities. It may be better to just define the category 'fatal' for this column.
- In the column 'Make', capitalization differences have been noted e.g. 'CESSNA' vs 'Cessna'. This could make python consider these as two different makers. This needs to be standardized/corrected.

Data Cleaning

As Identified during data perusal, we need to clean the data as recommended during the data understanding phase

Remove Rows with Empty Event.ID Values

```
#Remove rows where Event.Id is blank since all other columns are also blank when that is the
aviation_df.dropna(subset = ['Event.Id'], inplace = True)
```

```
#Get a new snapshot view of the data after removing empty Event.Id rows.
aviation_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 88889 entries, 0 to 90347
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            50132 non-null  object
 9   Airport.Name            52704 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     87507 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             81793 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           82505 non-null  object
 30  Publication.Date        73659 non-null  object
dtypes: float64(5), object(26)
memory usage: 21.7+ MB
```

```
aviation_df['Investigation.Type'].value_counts(normalize = True) * 100
```

Investigation.Type	proportion
Accident	95.641755
Incident	4.358245

dtype: float64

After removing rows where 'Event.Id' was empty, the resultant data now only has 2 categories in the 'Investigation.Type' column i.e. 'Accident' and 'Incident'

Remove Rows with Empty Make Values

```
#Remove rows where Make is blank since such data is not useful to this exercise
aviation_df.dropna(subset = ['Make'], inplace = True)
```

```
#Get a new snapshot view of the data after removing empty 'Make' columns
aviation_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 88826 entries, 0 to 90347
Data columns (total 31 columns):
 #   Column                  Non-Null Count  Dtype
---  ------                  --------------  -----
 0   Event.Id                88826 non-null  object
 1   Investigation.Type      88826 non-null  object
 2   Accident.Number         88826 non-null  object
 3   Event.Date              88826 non-null  object
 4   Location                88774 non-null  object
```

```
 5   Country                 88601 non-null  object
 6   Latitude                34360 non-null  object
 7   Longitude               34351 non-null  object
 8   Airport.Code            50118 non-null  object
 9   Airport.Name            52689 non-null  object
 10  Injury.Severity         87843 non-null  object
 11  Aircraft.damage         85650 non-null  object
 12  Aircraft.Category       32275 non-null  object
 13  Registration.Number     87483 non-null  object
 14  Make                    88826 non-null  object
 15  Model                   88777 non-null  object
 16  Amateur.Built           88726 non-null  object
 17  Number.of.Engines       82791 non-null  float64
 18  Engine.Type             81781 non-null  object
 19  FAR.Description         31972 non-null  object
 20  Schedule                12543 non-null  object
 21  Purpose.of.flight       82676 non-null  object
 22  Air.carrier             16624 non-null  object
 23  Total.Fatal.Injuries    77432 non-null  float64
 24  Total.Serious.Injuries  76326 non-null  float64
 25  Total.Minor.Injuries    76904 non-null  float64
 26  Total.Uninjured         82925 non-null  float64
 27  Weather.Condition       84372 non-null  object
 28  Broad.phase.of.flight   61713 non-null  object
 29  Report.Status           82474 non-null  object
 30  Publication.Date        73599 non-null  object
dtypes: float64(5), object(26)
memory usage: 21.7+ MB
```

Clean Injury.Severity Categories

```
#Clean the 'Injury.Severity' column by removing the number of fatalies from the string
#This will make the data cleaner i.e. reduce the number of severity categories
#The code searches for any entries where the substring 'Fatal(' appears and replaces the whc
#This way, 2 seperate categories e.g. Fatal(20) and Fatal(104) will all be made into the sam
aviation_df.loc[aviation_df['Injury.Severity'].str.contains("Fatal\\(", na=False), 'Injury.S


#Confirm that the cleaning exercise has yielded clean data (removed number of fatalities frc
aviation_df['Injury.Severity'].value_counts(normalize = True) * 100
```

Injury.Severity	proportion
Non-Fatal	76.650388
Fatal	20.273670
Incident	2.522683
Minor	0.248170
Serious	0.196942
Unavailable	0.108147

dtype: float64

Injury Severity has now been cleaned to 6 clear categories. The number of fatalities will be evaluated using Total.Fatal.Injuries data

Clean Make Data by Standardizing Capitalization

```
#Standardize the text capitalization to combine e.g. 'CESSNA' and 'Cessna'
#Use "title" capitalization format to rewrite the 'Make' column
aviation_df['Make'] = aviation_df['Make'].str.title()
aviation_df['Make'].value_counts(normalize = True) * 100 #Confirm if implementation has work
```

Make	proportion
Cessna	30.564249
Piper	16.740594
Beech	6.047779
Boeing	3.090311
Bell	3.064418
...	...
Cohen	0.001126
Kitchens	0.001126
Lutes	0.001126
Izatt	0.001126
Royse Ralph L	0.001126

7587 rows × 1 columns

dtype: float64

This has reduced the number of manufacturers from 8237 to 7587 i.e. we have cleaner "Make" data.

Clean Make Data by Aggregating Makers e.g. Boeing Vertol and Boeing-Brown are all just Boeing

Some of the major makers affected by this are Airbus, Boeing and Cessna

```
aviation_df['Make'] = aviation_df['Make'].apply(lambda x: 'Cessna' if 'Cessna' in x else x)
aviation_df['Make'] = aviation_df['Make'].apply(lambda x: 'Boeing' if 'Boeing' in x else x)
aviation_df['Make'] = aviation_df['Make'].apply(lambda x: 'Airbus' if 'Airbus' in x else x)
```

Clean Model Data by Standardizing Capitalization

```
#Standardize the text capitalization
#Use "title" capitalization format to rewrite the 'Model' column
aviation_df['Model'] = aviation_df['Model'].str.title()
aviation_df['Model'].value_counts(normalize = True) * 100 #Confirm if implementation has wor
```

Model	proportion
152	2.666231
172	1.977990
172N	1.311150
Pa-28-140	1.049821
150	0.933800
...	...
Challenger 2	0.001126
Bushby Mustang M-Ii	0.001126
Harvard Mk Ii	0.001126
Uc45J	0.001126
M-8 Eagle	0.001126

11640 rows × 1 columns

dtype: float64

Capitalization noise has been successfully removed

Clean Aircraft.Category by combining 'UNK' and 'Unknown'

'UNK' appears to mean 'Unknown' in this data set. We need to combine them where both appear

```
aviation_df['Aircraft.Category'] = aviation_df['Aircraft.Category'].replace('UNK', 'Unknown'
aviation_df['Aircraft.Category'].value_counts(normalize = True) * 100
```

Aircraft.Category	proportion
Airplane	85.539892
Helicopter	10.649109
Glider	1.573974
Balloon	0.715724
Gyrocraft	0.536019
Weight-Shift	0.498838
Powered Parachute	0.281952
Ultralight	0.092951
Unknown	0.049574
WSFT	0.027885
Powered-Lift	0.015492
Blimp	0.012393
Rocket	0.003098
ULTR	0.003098

dtype: float64

'UNK' and 'Unknown' successully merged

Clean Engine.Type by combining 'UNK' with 'Unknown'

'UNK' appears to mean 'Unknown' in this data set. We need to combine them where both appear

```
aviation_df['Engine.Type'] = aviation_df['Engine.Type'].replace('UNK', 'Unknown') #Replace '
aviation_df['Engine.Type'].value_counts(normalize = True) * 100
```

Engine.Type	proportion
Reciprocating	85.006297
Turbo Shaft	4.413005
Turbo Prop	4.146440
Turbo Fan	3.033712
Unknown	2.507917
Turbo Jet	0.859613
Geared Turbofan	0.014673
Electric	0.012228
LR	0.002446
NONE	0.002446
Hybrid Rocket	0.001223

dtype: float64

Clean Weather.Condition by Standardizing Capitalization

```
#Standardize the text capitalization
#Use "upper" capitalization format to rewrite the 'Model' column
aviation_df['Weather.Condition'] = aviation_df['Weather.Condition'].str.upper()
aviation_df['Weather.Condition'] = aviation_df['Weather.Condition'].replace('UNK', 'Unknown'
aviation_df['Weather.Condition'].value_counts(normalize = True) * 100 #Confirm if implementa
```

Weather.Condition	proportion
VMC	91.596738
IMC	7.079363
Unknown	1.323899

dtype: float64

Clean Schedule by Replacing UNK with Unknown

```
aviation_df['Schedule'] = aviation_df['Schedule'].replace('UNK', 'Unknown') #Replace 'UNK' e
aviation_df['Schedule'].value_counts(normalize = True) * 100 #Confirm if implementation has
```

Schedule	proportion
NSCH	35.581599
Unknown	32.679582
SCHD	31.738818

dtype: float64

Check for Duplicates

```
aviation_df.duplicated().sum()
```

```
0
```

There are no duplicated rowa in the dataset.

Convert Date Columns to standard datetime format

```
#Transform the date columns to standard datetime format
aviation_df['Event.Date'] = pd.to_datetime(aviation_df['Event.Date'], format = '%Y-%m-%d')
aviation_df['Publication.Date'] = pd.to_datetime(aviation_df['Publication.Date'], format = '

aviation_df['Event.Date'] = aviation_df['Event.Date'].dt.strftime('%d-%m-%Y')
aviation_df['Publication.Date'] = aviation_df['Publication.Date'].dt.strftime('%d-%m-%Y')

aviation_df.tail()
```

	Event.Id	Investigation.Type	Accident.Number	Event.Date	Location	Count
90343	20221227106491	Accident	ERA23LA093	26-12-2022	Annapolis, MD	Unit Stat
90344	20221227106494	Accident	ERA23LA095	26-12-2022	Hampton, NH	Unit Stat
90345	20221227106497	Accident	WPR23LA075	26-12-2022	Payson, AZ	Unit Stat
90346	20221227106498	Accident	WPR23LA076	26-12-2022	Morgan, UT	Unit Stat
90347	20221230106513	Accident	ERA23LA097	29-12-2022	Athens, GA	Unit Stat

5 rows × 31 columns

The dates are now written in a standard format

```
aviation_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 88826 entries, 0 to 90347
Data columns (total 31 columns):
 #   Column                  Non-Null Count  Dtype  
---  ------                  --------------  -----  
 0   Event.Id                88826 non-null  object 
 1   Investigation.Type      88826 non-null  object 
 2   Accident.Number         88826 non-null  object 
 3   Event.Date              88826 non-null  object 
 4   Location                88774 non-null  object 
 5   Country                 88601 non-null  object 
 6   Latitude                34360 non-null  object 
 7   Longitude               34351 non-null  object 
 8   Airport.Code            50118 non-null  object 
 9   Airport.Name            52689 non-null  object 
 10  Injury.Severity         87843 non-null  object 
 11  Aircraft.damage         85650 non-null  object 
 12  Aircraft.Category       32275 non-null  object 
 13  Registration.Number     87483 non-null  object 
 14  Make                    88826 non-null  object 
 15  Model                   88777 non-null  object 
 16  Amateur.Built           88726 non-null  object 
 17  Number.of.Engines       82791 non-null  float64
 18  Engine.Type             81781 non-null  object 
 19  FAR.Description         31972 non-null  object 
 20  Schedule                12543 non-null  object 
 21  Purpose.of.flight       82676 non-null  object 
 22  Air.carrier             16624 non-null  object 
 23  Total.Fatal.Injuries    77432 non-null  float64
 24  Total.Serious.Injuries  76326 non-null  float64
```

```
 25  Total.Minor.Injuries    76904 non-null  float64
 26  Total.Uninjured         82925 non-null  float64
 27  Weather.Condition       84372 non-null  object
 28  Broad.phase.of.flight   61713 non-null  object
 29  Report.Status           82474 non-null  object
 30  Publication.Date        73599 non-null  object
dtypes: float64(5), object(26)
memory usage: 21.7+ MB
```

Handling Missing Values

We will use different strategies to handle different categories of missing data

- Most of the longitude and latitude data is missing, and it does not seem to be relevant to this study. Additionally, we also have location data which is more readily available. Thus, we can drop the longitude and latitude columns
- FAR.Description data seems irrelelevant to the study
- For the few missing values of relevant categorical columns, we can fill missing values with "Unknown". We may be able to extract insight even with "Unknown" parameters if others related parameters are known e.g. you may have an unknown model but know the manufacturer. This remains relevant to our study.
- For the data on total number of injuries, it is best to assume that the data meant to be there is '0' e.g. if Total.Serious.Injuries is empty, we assume it was 0.

```
#These columns seem irrelevant at this time for the study. We will drop them for now
aviation_df.drop(['Longitude', 'Latitude','FAR.Description'], axis = 1, inplace = True)
```

```
#replace missing categorical data with "Unknown"
aviation_df['Location'] = aviation_df['Location'].fillna('Unknown')
aviation_df['Country'] = aviation_df['Country'].fillna('Unknown')
aviation_df['Injury.Severity'] = aviation_df['Injury.Severity'].fillna('Unknown')
aviation_df['Airport.Code'] = aviation_df['Airport.Code'].fillna('Unknown')
aviation_df['Airport.Name'] = aviation_df['Airport.Name'].fillna('Unknown')
aviation_df['Injury.Severity'] = aviation_df['Injury.Severity'].fillna('Unknown')
aviation_df['Aircraft.damage'] = aviation_df['Aircraft.damage'].fillna('Unknown')
aviation_df['Aircraft.Category'] = aviation_df['Aircraft.Category'].fillna('Unknown')
aviation_df['Registration.Number'] = aviation_df['Registration.Number'].fillna('Unknown')
aviation_df['Model'] = aviation_df['Model'].fillna('Unknown')
aviation_df['Amateur.Built'] = aviation_df['Amateur.Built'].fillna('Unknown')
aviation_df['Number.of.Engines'] = aviation_df['Number.of.Engines'].fillna('Unknown')
aviation_df['Air.carrier'] = aviation_df['Air.carrier'].fillna('Unknown')
aviation_df['Purpose.of.flight'] = aviation_df['Purpose.of.flight'].fillna('Unknown')
aviation_df['Schedule'] = aviation_df['Schedule'].fillna('Unknown')
aviation_df['Weather.Condition'] = aviation_df['Weather.Condition'].fillna('Unknown')
aviation_df['Broad.phase.of.flight'] = aviation_df['Broad.phase.of.flight'].fillna('Unknown'
aviation_df['Report.Status'] = aviation_df['Report.Status'].fillna('Unknown')
```

```
aviation_df['Engine.Type'] = aviation_df['Engine.Type'].fillna('Unknown')
#aviation_df[''] = aviation_df[''].fillna('Unknown')
aviation_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 88826 entries, 0 to 90347
Data columns (total 28 columns):
 #   Column                  Non-Null Count  Dtype  
---  ------                  --------------  -----  
 0   Event.Id                88826 non-null  object 
 1   Investigation.Type      88826 non-null  object 
 2   Accident.Number         88826 non-null  object 
 3   Event.Date              88826 non-null  object 
 4   Location                88826 non-null  object 
 5   Country                 88826 non-null  object 
 6   Airport.Code            88826 non-null  object 
 7   Airport.Name            88826 non-null  object 
 8   Injury.Severity         88826 non-null  object 
 9   Aircraft.damage         88826 non-null  object 
 10  Aircraft.Category       88826 non-null  object 
 11  Registration.Number     88826 non-null  object 
 12  Make                    88826 non-null  object 
 13  Model                   88826 non-null  object 
 14  Amateur.Built           88826 non-null  object 
 15  Number.of.Engines       88826 non-null  object 
 16  Engine.Type             88826 non-null  object 
 17  Schedule                88826 non-null  object 
 18  Purpose.of.flight       88826 non-null  object 
 19  Air.carrier             88826 non-null  object 
 20  Total.Fatal.Injuries    77432 non-null  float64
 21  Total.Serious.Injuries  76326 non-null  float64
 22  Total.Minor.Injuries    76904 non-null  float64
 23  Total.Uninjured         82925 non-null  float64
 24  Weather.Condition       88826 non-null  object 
 25  Broad.phase.of.flight   88826 non-null  object 
 26  Report.Status           88826 non-null  object 
 27  Publication.Date        73599 non-null  object 
dtypes: float64(4), object(24)
memory usage: 19.7+ MB
```

```
#Replace missing quantitative data with 0
aviation_df['Total.Fatal.Injuries'] = aviation_df['Total.Fatal.Injuries'].fillna(0)
aviation_df['Total.Serious.Injuries'] = aviation_df['Total.Serious.Injuries'].fillna(0)
aviation_df['Total.Minor.Injuries'] = aviation_df['Total.Minor.Injuries'].fillna(0)
aviation_df['Total.Uninjured'] = aviation_df['Total.Uninjured'].fillna(0)

aviation_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 88826 entries, 0 to 90347
Data columns (total 28 columns):
 #   Column                  Non-Null Count  Dtype  
---  ------                  --------------  -----  
 0   Event.Id                88826 non-null  object 
```

```
 1   Investigation.Type      88826 non-null  object
 2   Accident.Number         88826 non-null  object
 3   Event.Date              88826 non-null  object
 4   Location                88826 non-null  object
 5   Country                 88826 non-null  object
 6   Airport.Code            88826 non-null  object
 7   Airport.Name            88826 non-null  object
 8   Injury.Severity         88826 non-null  object
 9   Aircraft.damage         88826 non-null  object
 10  Aircraft.Category       88826 non-null  object
 11  Registration.Number     88826 non-null  object
 12  Make                    88826 non-null  object
 13  Model                   88826 non-null  object
 14  Amateur.Built           88826 non-null  object
 15  Number.of.Engines       88826 non-null  object
 16  Engine.Type             88826 non-null  object
 17  Schedule                88826 non-null  object
 18  Purpose.of.flight       88826 non-null  object
 19  Air.carrier             88826 non-null  object
 20  Total.Fatal.Injuries    88826 non-null  float64
 21  Total.Serious.Injuries  88826 non-null  float64
 22  Total.Minor.Injuries    88826 non-null  float64
 23  Total.Uninjured         88826 non-null  float64
 24  Weather.Condition       88826 non-null  object
 25  Broad.phase.of.flight   88826 non-null  object
 26  Report.Status           88826 non-null  object
 27  Publication.Date        73599 non-null  object
dtypes: float64(4), object(24)
memory usage: 19.7+ MB
```

Encoding Categorical Data to Facilitate Correlation Calculations

To enable us calculate correlations between various categorical attributes, it becomes important to codify the categorical data. We would need to create a new dataframe for storing the encoded data

```
encoded_aviation_df = aviation_df.copy()
#encoded_aviation_df.drop(['Accident.Number', 'Publication.Date','Registration.Number','Ever
encoded_aviation_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 88826 entries, 0 to 90347
Data columns (total 28 columns):
 #   Column                  Non-Null Count  Dtype
---  ------                  --------------  -----
 0   Event.Id                88826 non-null  object
 1   Investigation.Type      88826 non-null  object
 2   Accident.Number         88826 non-null  object
 3   Event.Date              88826 non-null  object
 4   Location                88826 non-null  object
 5   Country                 88826 non-null  object
 6   Airport.Code            88826 non-null  object
 7   Airport.Name            88826 non-null  object
```

```
 8   Injury.Severity         88826 non-null  object
 9   Aircraft.damage         88826 non-null  object
 10  Aircraft.Category       88826 non-null  object
 11  Registration.Number     88826 non-null  object
 12  Make                    88826 non-null  object
 13  Model                   88826 non-null  object
 14  Amateur.Built           88826 non-null  object
 15  Number.of.Engines       88826 non-null  object
 16  Engine.Type             88826 non-null  object
 17  Schedule                88826 non-null  object
 18  Purpose.of.flight       88826 non-null  object
 19  Air.carrier             88826 non-null  object
 20  Total.Fatal.Injuries    88826 non-null  float64
 21  Total.Serious.Injuries  88826 non-null  float64
 22  Total.Minor.Injuries    88826 non-null  float64
 23  Total.Uninjured         88826 non-null  float64
 24  Weather.Condition       88826 non-null  object
 25  Broad.phase.of.flight   88826 non-null  object
 26  Report.Status           88826 non-null  object
 27  Publication.Date        73599 non-null  object
dtypes: float64(4), object(24)
memory usage: 19.7+ MB
```

```
encoded_aviation_df['Investigation.Type.encoded'] = pd.Categorical(encoded_aviation_df['Inve
encoded_aviation_df['Location.encoded'] = pd.Categorical(encoded_aviation_df['Location']).co
encoded_aviation_df['Country.encoded'] = pd.Categorical(encoded_aviation_df['Country']).code
encoded_aviation_df['Airport.Code.encoded'] = pd.Categorical(encoded_aviation_df['Airport.Co
encoded_aviation_df['Airport.Name.encoded'] = pd.Categorical(encoded_aviation_df['Airport.Na
encoded_aviation_df['Injury.Severity.encoded'] = pd.Categorical(encoded_aviation_df['Injury.
encoded_aviation_df['Aircraft.damage.encoded'] = pd.Categorical(encoded_aviation_df['Aircraf
encoded_aviation_df['Aircraft.Category.encoded'] = pd.Categorical(encoded_aviation_df['Aircr
encoded_aviation_df['Make.encoded'] = pd.Categorical(encoded_aviation_df['Make']).codes
encoded_aviation_df['Model.encoded'] = pd.Categorical(encoded_aviation_df['Model']).codes
encoded_aviation_df['Amateur.Built.encoded'] = pd.Categorical(encoded_aviation_df['Amateur.B
#encoded_aviation_df['Number.of.Engines.encoded'] = pd.Categorical(encoded_aviation_df['Numb
encoded_aviation_df['Engine.Type.encoded'] = pd.Categorical(encoded_aviation_df['Engine.Type
encoded_aviation_df['Schedule.encoded'] = pd.Categorical(encoded_aviation_df['Schedule']).co
encoded_aviation_df['Purpose.of.flight.encoded'] = pd.Categorical(encoded_aviation_df['Purpo
encoded_aviation_df['Weather.Condition.encoded'] = pd.Categorical(encoded_aviation_df['Weath
encoded_aviation_df['Broad.phase.of.flight.encoded'] = pd.Categorical(encoded_aviation_df['B
#encoded_aviation_df['.encoded'] = pd.Categorical(encoded_aviation_df['']).codes

encoded_aviation_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 88826 entries, 0 to 90347
Data columns (total 44 columns):
 #   Column                          Non-Null Count  Dtype
---  ------                          --------------  -----
 0   Event.Id                        88826 non-null  object
 1   Investigation.Type              88826 non-null  object
 2   Accident.Number                 88826 non-null  object
```

```
 3   Event.Date                     88826 non-null  object
 4   Location                       88826 non-null  object
 5   Country                        88826 non-null  object
 6   Airport.Code                   88826 non-null  object
 7   Airport.Name                   88826 non-null  object
 8   Injury.Severity                88826 non-null  object
 9   Aircraft.damage                88826 non-null  object
 10  Aircraft.Category              88826 non-null  object
 11  Registration.Number            88826 non-null  object
 12  Make                           88826 non-null  object
 13  Model                          88826 non-null  object
 14  Amateur.Built                  88826 non-null  object
 15  Number.of.Engines              88826 non-null  object
 16  Engine.Type                    88826 non-null  object
 17  Schedule                       88826 non-null  object
 18  Purpose.of.flight              88826 non-null  object
 19  Air.carrier                    88826 non-null  object
 20  Total.Fatal.Injuries           88826 non-null  float64
 21  Total.Serious.Injuries         88826 non-null  float64
 22  Total.Minor.Injuries           88826 non-null  float64
 23  Total.Uninjured                88826 non-null  float64
 24  Weather.Condition              88826 non-null  object
 25  Broad.phase.of.flight          88826 non-null  object
 26  Report.Status                  88826 non-null  object
 27  Publication.Date               73599 non-null  object
 28  Investigation.Type.encoded     88826 non-null  int8
 29  Location.encoded               88826 non-null  int16
 30  Country.encoded                88826 non-null  int16
 31  Airport.Code.encoded           88826 non-null  int16
 32  Airport.Name.encoded           88826 non-null  int16
 33  Injury.Severity.encoded        88826 non-null  int8
 34  Aircraft.damage.encoded        88826 non-null  int8
 35  Aircraft.Category.encoded      88826 non-null  int8
 36  Make.encoded                   88826 non-null  int16
 37  Model.encoded                  88826 non-null  int16
 38  Amateur.Built.encoded          88826 non-null  int8
 39  Engine.Type.encoded            88826 non-null  int8
 40  Schedule.encoded               88826 non-null  int8
 41  Purpose.of.flight.encoded      88826 non-null  int8
 42  Weather.Condition.encoded      88826 non-null  int8
 43  Broad.phase.of.flight.encoded  88826 non-null  int8
dtypes: float64(4), int16(6), int8(10), object(24)
memory usage: 21.5+ MB
```

```
#Output data to CSV for EDA with Tableau
encoded_aviation_df.to_csv('encoded_aviation_data.csv', index = False)
```

```
encoded_aviation_df['Total.Fatal.Injuries'] = encoded_aviation_df['Total.Fatal.Injuries'].as
encoded_aviation_df['Total.Serious.Injuries'] = encoded_aviation_df['Total.Serious.Injuries'
encoded_aviation_df['Total.Minor.Injuries'] = encoded_aviation_df['Total.Minor.Injuries'].as
encoded_aviation_df['Total.Uninjured'] = encoded_aviation_df['Total.Uninjured'].astype(int)
```

```
encoded_aviation_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 88826 entries, 0 to 90347
Data columns (total 44 columns):
 #   Column                           Non-Null Count  Dtype 
---  ------                           --------------  ----- 
 0   Event.Id                         88826 non-null  object
 1   Investigation.Type               88826 non-null  object
 2   Accident.Number                  88826 non-null  object
 3   Event.Date                       88826 non-null  object
 4   Location                         88826 non-null  object
 5   Country                          88826 non-null  object
 6   Airport.Code                     88826 non-null  object
 7   Airport.Name                     88826 non-null  object
 8   Injury.Severity                  88826 non-null  object
 9   Aircraft.damage                  88826 non-null  object
 10  Aircraft.Category                88826 non-null  object
 11  Registration.Number              88826 non-null  object
 12  Make                             88826 non-null  object
 13  Model                            88826 non-null  object
 14  Amateur.Built                    88826 non-null  object
 15  Number.of.Engines                88826 non-null  object
 16  Engine.Type                      88826 non-null  object
 17  Schedule                         88826 non-null  object
 18  Purpose.of.flight                88826 non-null  object
 19  Air.carrier                      88826 non-null  object
 20  Total.Fatal.Injuries             88826 non-null  int64 
 21  Total.Serious.Injuries           88826 non-null  int64 
 22  Total.Minor.Injuries             88826 non-null  int64 
 23  Total.Uninjured                  88826 non-null  int64 
 24  Weather.Condition                88826 non-null  object
 25  Broad.phase.of.flight            88826 non-null  object
 26  Report.Status                    88826 non-null  object
 27  Publication.Date                 73599 non-null  object
 28  Investigation.Type.encoded       88826 non-null  int8  
 29  Location.encoded                 88826 non-null  int16 
 30  Country.encoded                  88826 non-null  int16 
 31  Airport.Code.encoded             88826 non-null  int16 
 32  Airport.Name.encoded             88826 non-null  int16 
 33  Injury.Severity.encoded          88826 non-null  int8  
 34  Aircraft.damage.encoded          88826 non-null  int8  
 35  Aircraft.Category.encoded        88826 non-null  int8  
 36  Make.encoded                     88826 non-null  int16 
 37  Model.encoded                    88826 non-null  int16 
 38  Amateur.Built.encoded            88826 non-null  int8  
 39  Engine.Type.encoded              88826 non-null  int8  
 40  Schedule.encoded                 88826 non-null  int8  
 41  Purpose.of.flight.encoded        88826 non-null  int8  
 42  Weather.Condition.encoded        88826 non-null  int8  
 43  Broad.phase.of.flight.encoded    88826 non-null  int8  
dtypes: int16(6), int64(4), int8(10), object(24)
memory usage: 21.5+ MB
```

```
#Prepare a purely numerical datafram to facilitate correlation analysis
correlation_aviation_df = encoded_aviation_df.copy() #Make a copy of the dataset
correlation_aviation_df = correlation_aviation_df.iloc[:, 20:] #Drop the non-numerical data
correlation_aviation_df.drop(['Weather.Condition','Broad.phase.of.flight','Report.Status','F
correlation_aviation_df.info()
```

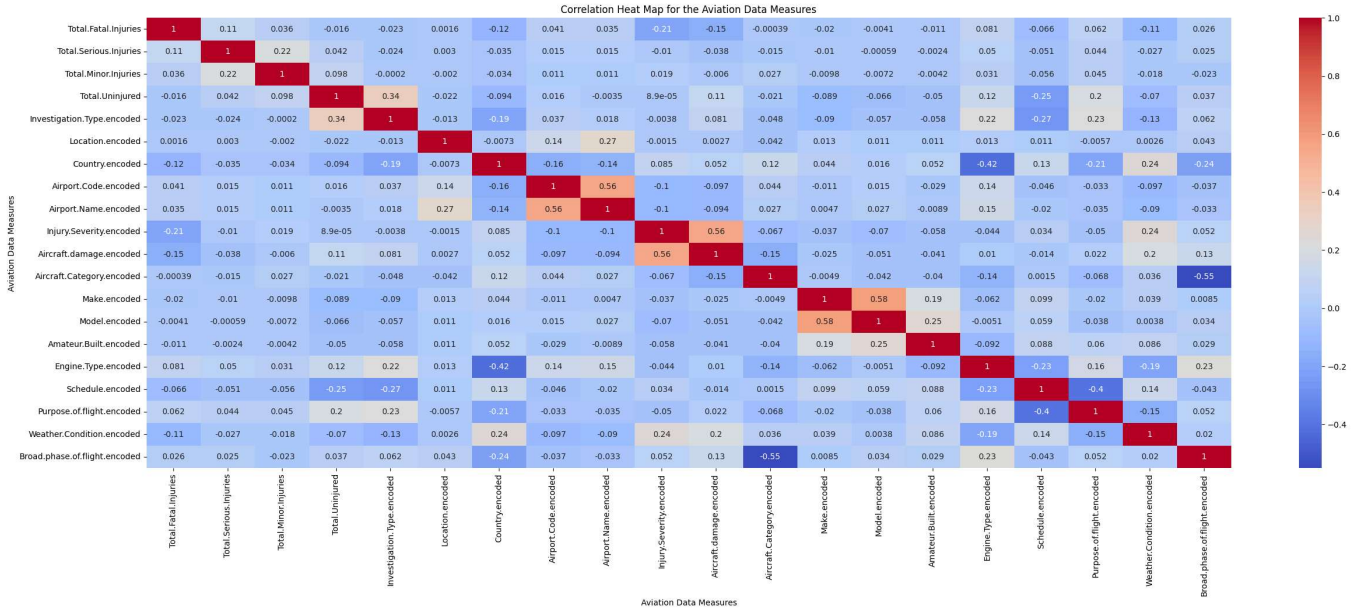
```
<class 'pandas.core.frame.DataFrame'>
Index: 88826 entries, 0 to 90347
Data columns (total 20 columns):
 #   Column                          Non-Null Count  Dtype
---  ------                          --------------  -----
 0   Total.Fatal.Injuries            88826 non-null  int64
 1   Total.Serious.Injuries          88826 non-null  int64
 2   Total.Minor.Injuries            88826 non-null  int64
 3   Total.Uninjured                 88826 non-null  int64
 4   Investigation.Type.encoded      88826 non-null  int8
 5   Location.encoded                88826 non-null  int16
 6   Country.encoded                 88826 non-null  int16
 7   Airport.Code.encoded            88826 non-null  int16
 8   Airport.Name.encoded            88826 non-null  int16
 9   Injury.Severity.encoded         88826 non-null  int8
 10  Aircraft.damage.encoded         88826 non-null  int8
 11  Aircraft.Category.encoded       88826 non-null  int8
 12  Make.encoded                    88826 non-null  int16
 13  Model.encoded                   88826 non-null  int16
 14  Amateur.Built.encoded           88826 non-null  int8
 15  Engine.Type.encoded             88826 non-null  int8
 16  Schedule.encoded                88826 non-null  int8
 17  Purpose.of.flight.encoded       88826 non-null  int8
 18  Weather.Condition.encoded       88826 non-null  int8
 19  Broad.phase.of.flight.encoded   88826 non-null  int8
dtypes: int16(6), int64(4), int8(10)
memory usage: 5.3 MB
```

Exploratory Data Analysis with Python & Tableau

We can check for correlation in the measures in our data:

```
#Use a heatmap to compare correlation between various measures (including the numerically er
plt.figure(figsize = (30,10))
sns.heatmap(correlation_aviation_df.corr(), annot = True, cmap = 'coolwarm')
plt.xlabel('Aviation Data Measures')
plt.ylabel("Aviation Data Measures")
plt.title("Correlation Heat Map for the Aviation Data Measures")
```

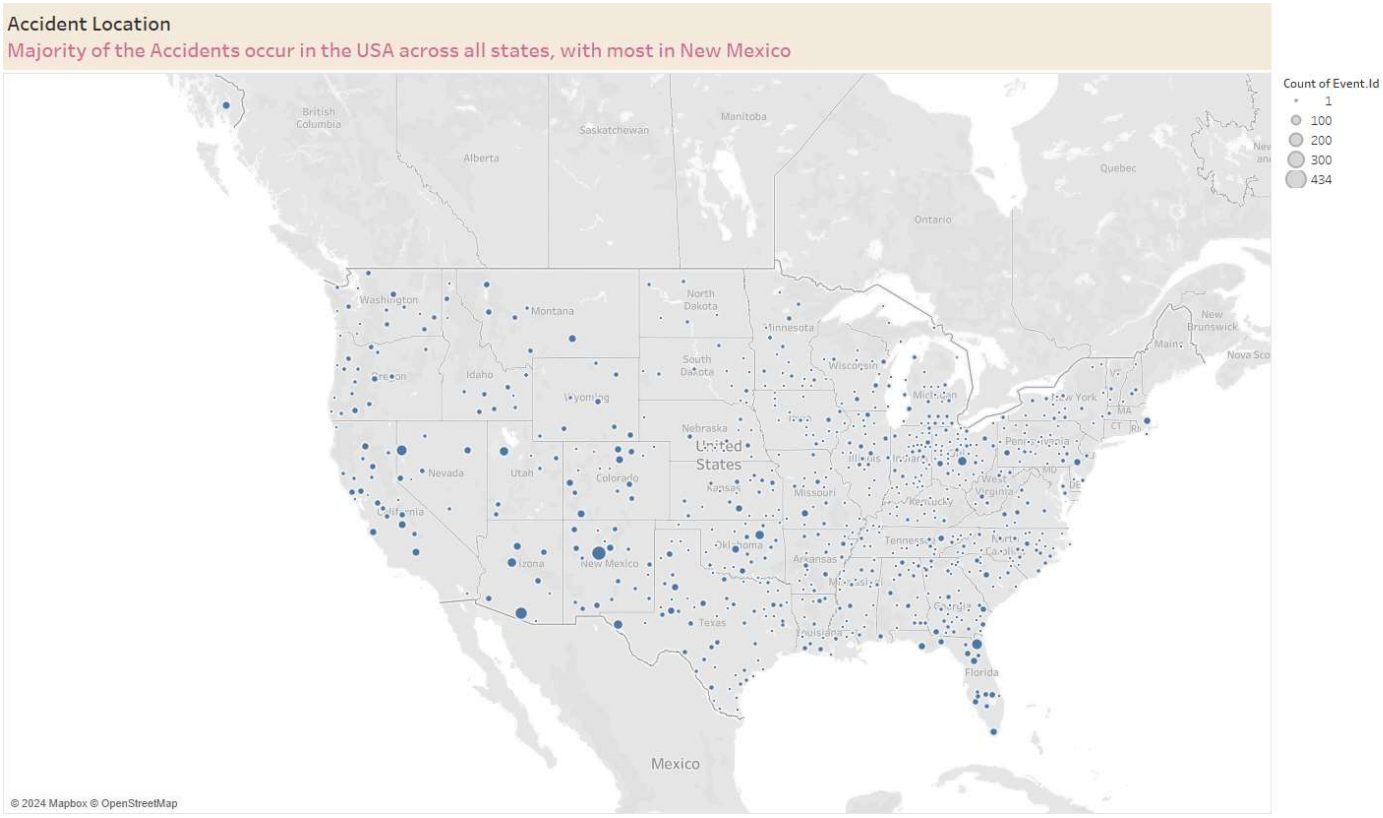
```
Text(0.5, 1.0, 'Correlation Heat Map for the Aviation Data Measures')
```



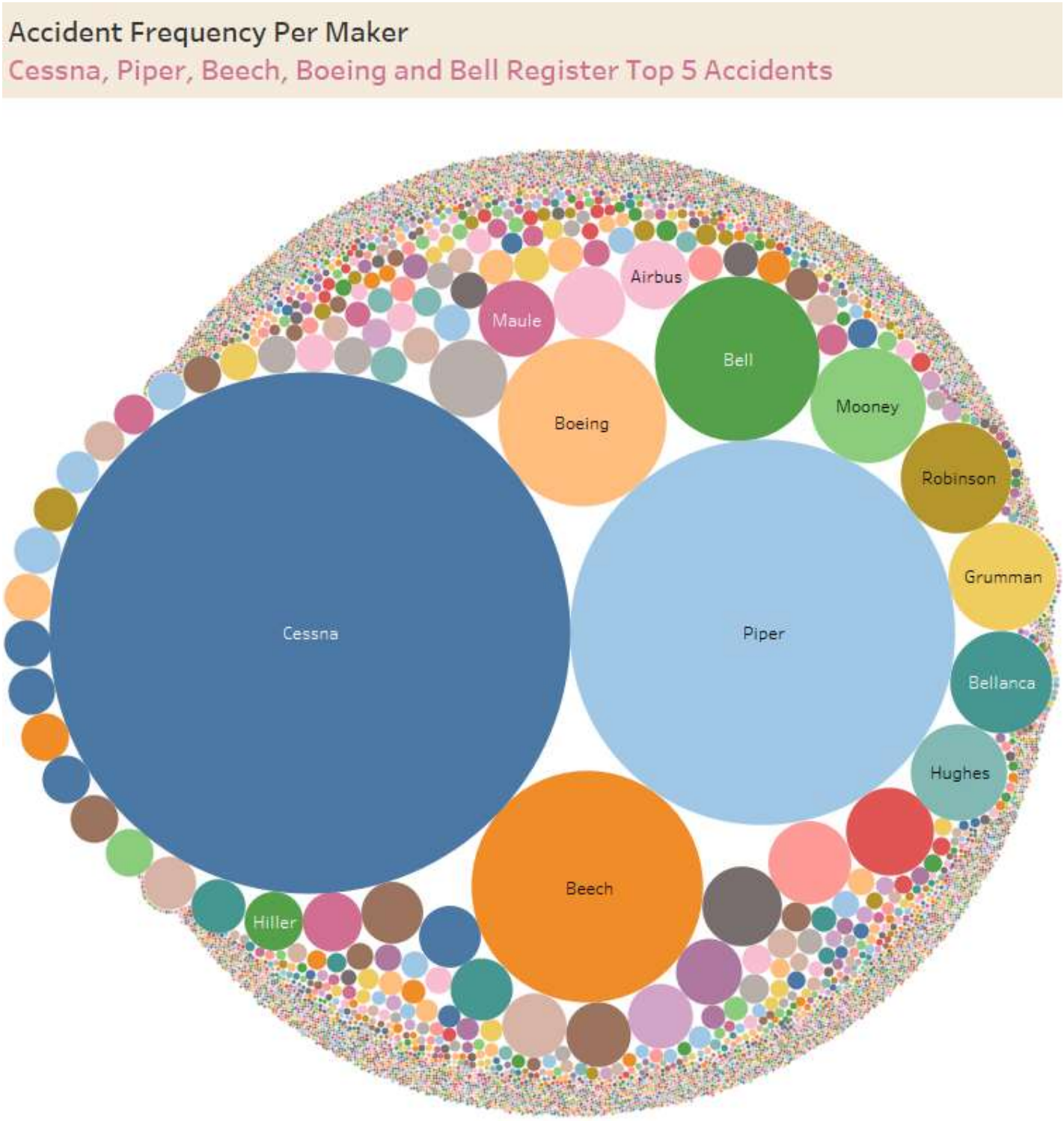
There is no immediate correlationary insight noted from correlation heat map. This could be due to the several cases of "Unknown" category. It is prudent to carry out further exploratory analysis using Tableau.

Tableau EDA

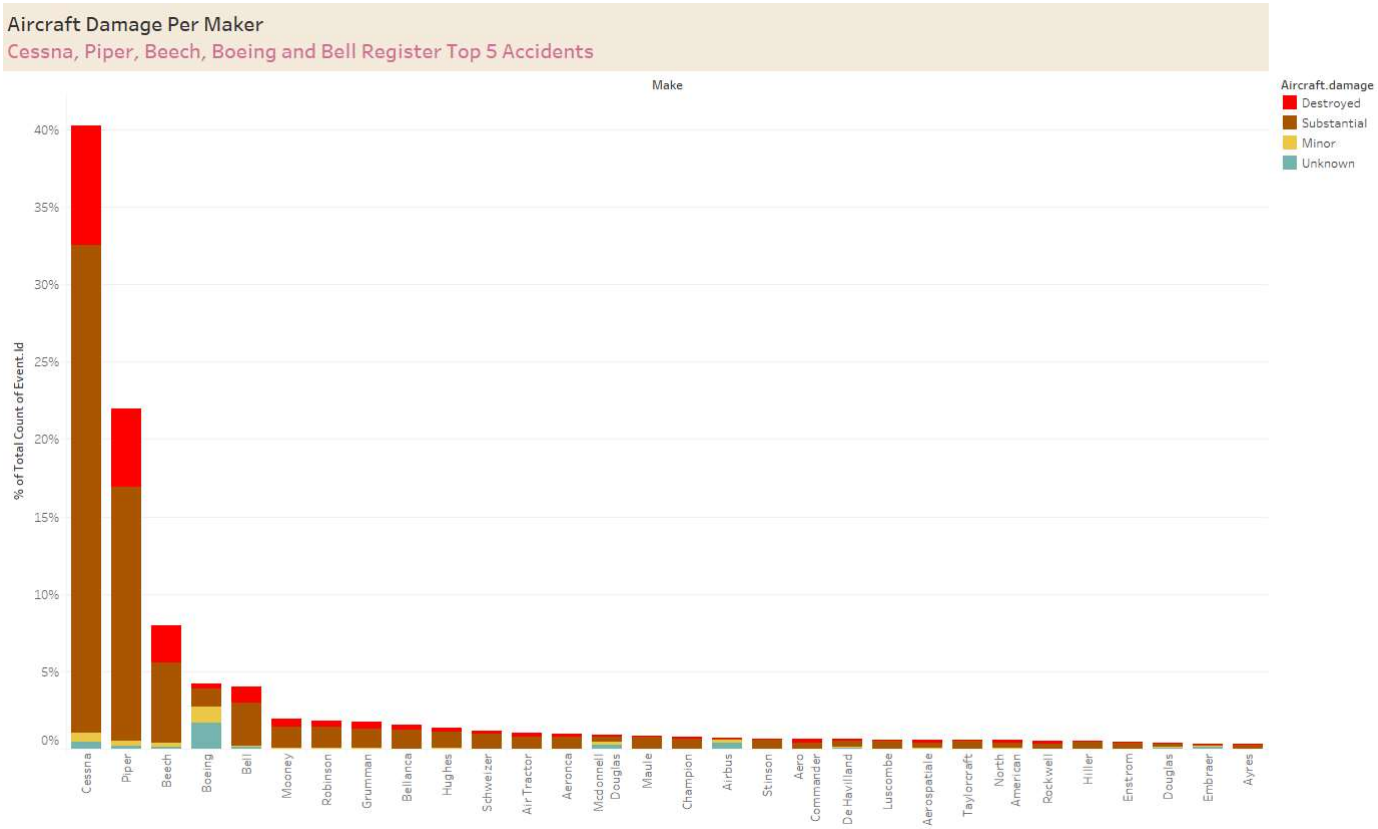
Most of the accidents occur in the USA, distributed across almost all the states.



Cessna, Piper, Beech, Boeing and Bell registered the most accidents.

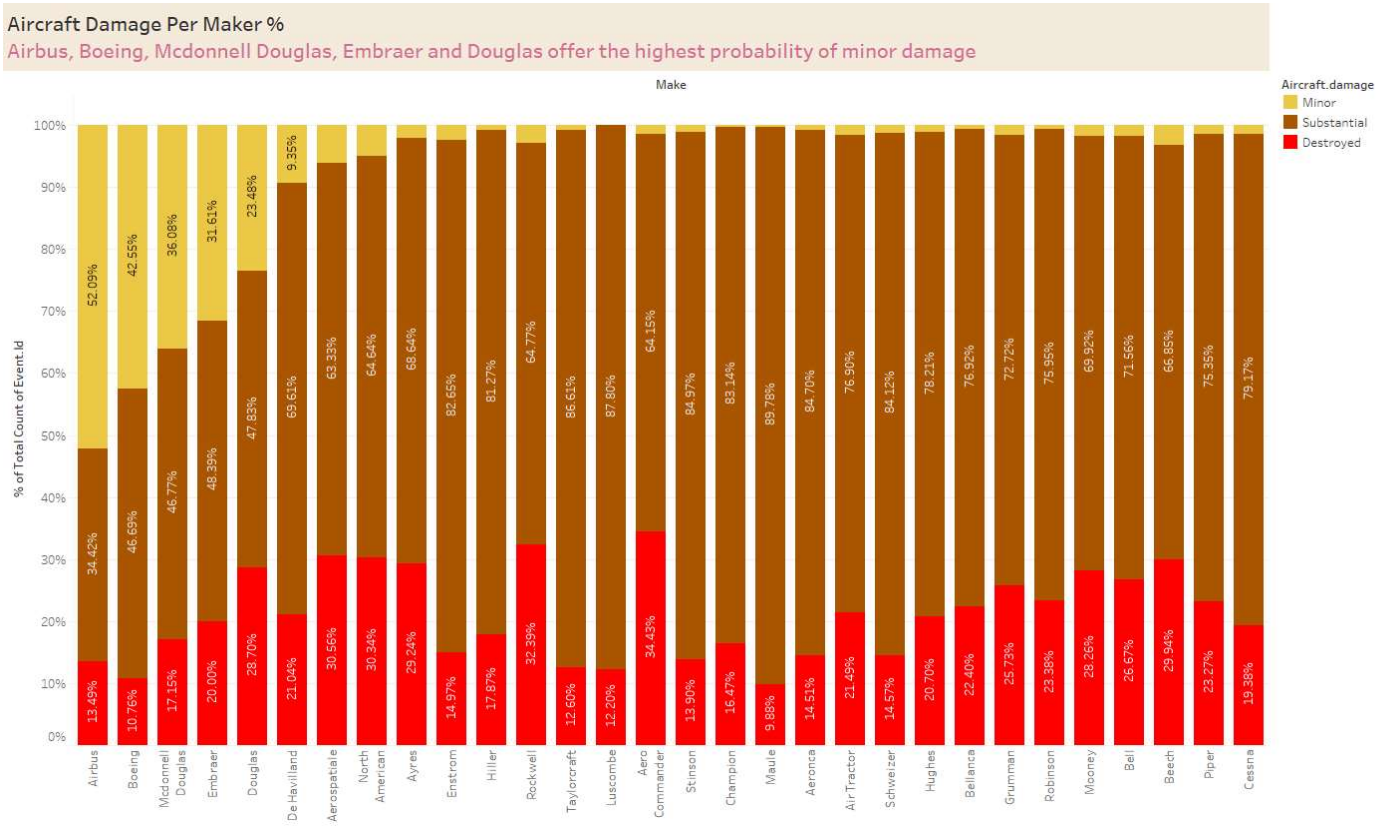


In the event of an accident, it is almost guaranteed that the aircraft damage will be substantial to totally damaged for most of the makers.

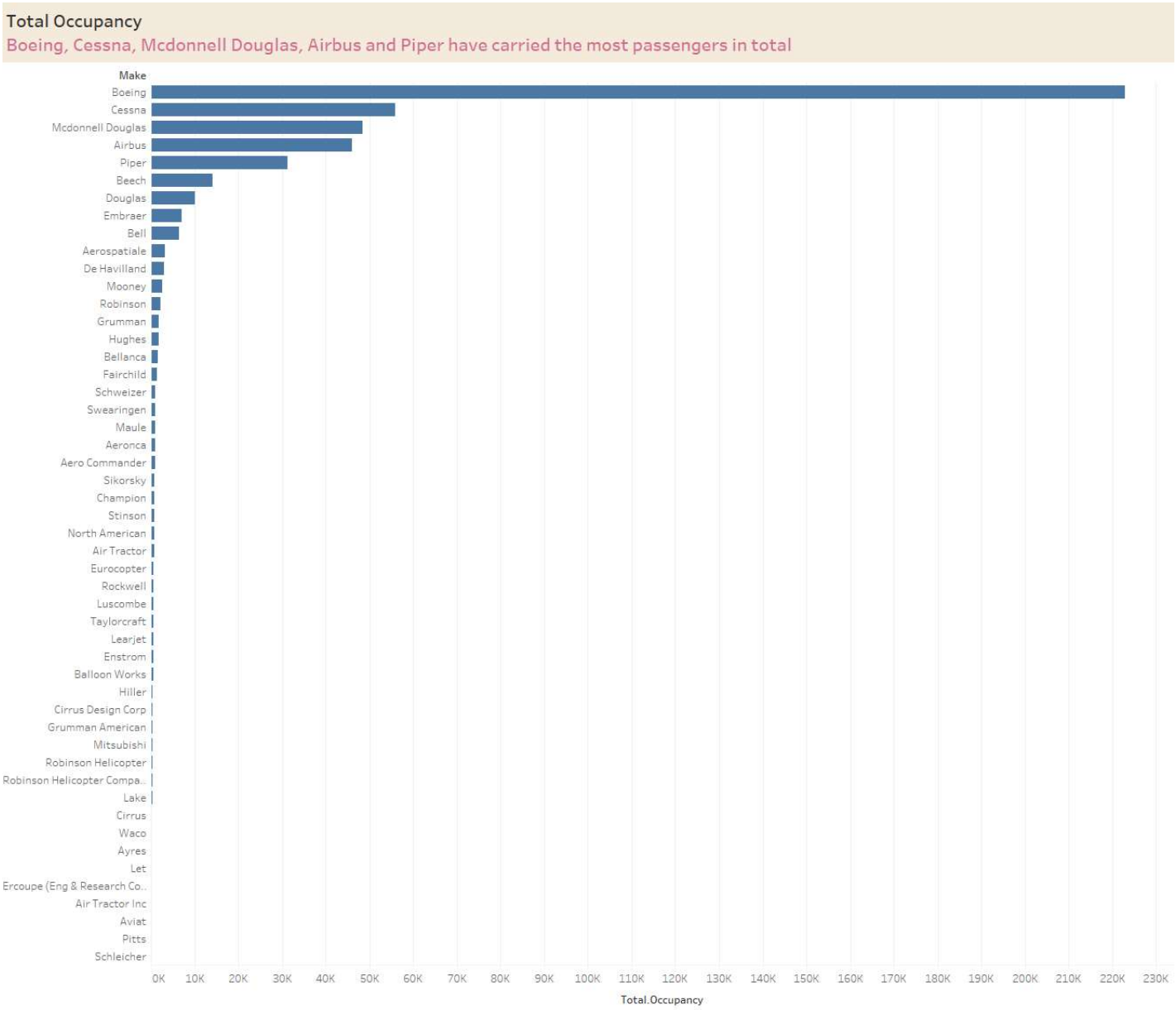


If you remove events where the aircraft damage is "Unknown", the following makers emerge as better candidates where there is a high likelihood of minor the accident only resulting in minor damage:

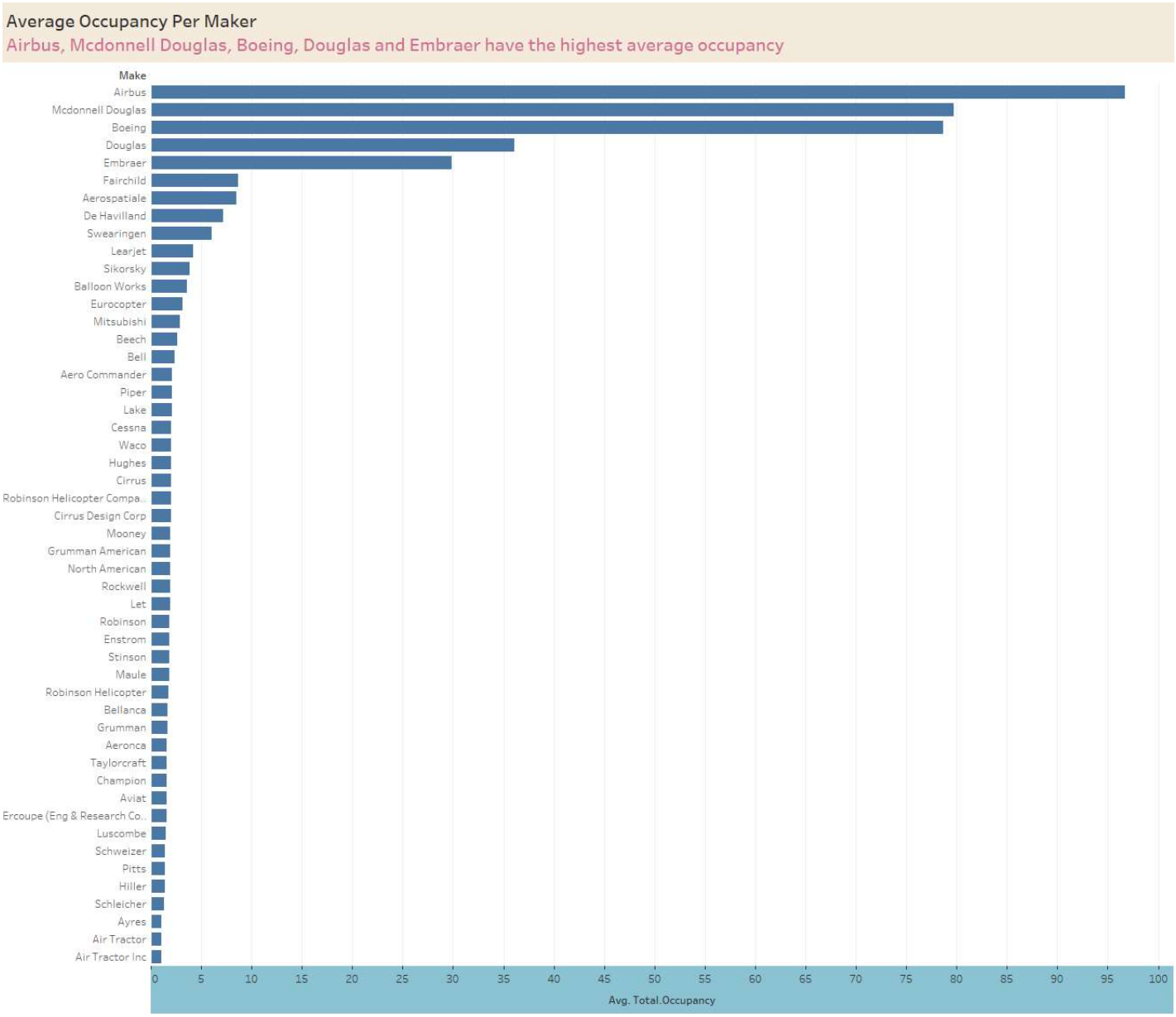
- Airbus
- Boeing
- Mcdonnell Douglas
- Embraer
- Douglas



To analyze fatalities in the event of an accident, it is important to understand the total number of passengers involved in this data. This can be done by summing up all the fatalities, serious injuries, minor injuries and uninjured passengers.



We go a step further to assess the average occupancy per flight from the data. This will tell us the relative sizes of the aircrafts



Airbus, Mcdonnel Douglas, Boeing, Douglas and Embraer are generally bigger planes carrying more people, hence their high number of average occupancy. Cessna has a very low average occupancy, i.e. they make small aircrafts carrying very few people (around 2).

It is important to consider the purpose of the flight from the accident data

Accident Frequency Based on Purpose of Flight

56% of accidents occur on personal flights