```
Project Submission
Please fill out:

Student name: VIVIAN NYAWIRA NJAGI
Student pace: Part time
Scheduled project review date: 9/9/2024
Instructor name: Williiam Okomba
```

AIRCRAFT RISK ASSESSMENT FOR COMMERCIAL AND PRIVATE OPERATIONS

Overview

This project aims to analyze the history of the Aviation data and decide which aircraft model has the lower risk for commercial and private enterprises.The analysis will help the company in determining which aircraft is safest to purchase and operate to minimize the potential risk of accidents.The audience of this analysis is the Head of the new aviation division who will help decide which aircraft to purchase,this will be possible after translating my findings and giving actionable insights.The Dataset 'AviationData.csv will help in analysing by identifying th e patterns,the trends and accidents rates based on the aircraft model,type of injuries, weather conditions and other factors that contribute to the risk of each aircraft.

Business Understanding

The company has decided to diversify its portifolio by purchasing commercial and private aircrafts but the problem is they have no clue on the risk associated with the aircraft model.They want to identify which aircraft model has the lowest risk and purchase that.

The Data

The data is from the National Transportation Safety Board that includes aviation accident data from 1962 to 2023 about civil aviation accidents and selected incidents in the United States and international waters.A preliminary report is available online within a few days of an accident. Factual information is added when available, and when the investigation is completed, the preliminary report is replaced with a final description of the accident and its probable cause.

Questions to consider

1. Which country had the highest number of accidents?
2. Which country had the highest cases of Fatal injuries?
3. Are there certain locations where accidents are most likely to occur?
4. In which weather condition were those flights conducted?

5. In which weather conditions did most fatal accidents occur?
6. Which aircraft make had the highest Injury severity?
7. Which make and Model has lowest risk of injuries?
8. At which broad phase of fight did accident occur the most?
9. Which aircraft are we going to purchase for both private and commercial purposes with lowest risk of accidents?
10. What is the trend of accidents over the years?

Data preparation and Cleaning

Objectives:

1. Loading files using python packages.
2. Inspecting the data and columns.
3. Handling missing and inaccurate data by identifying missing values and inaccurate values and fixing.
4. Ensure the desired observations are well organised.

1.0 Import python libraries

In [1]:

```
# numpy for numerical computations  and mathematical calculations on arrays
import numpy as np
# pandas for data manipulation and analysis and reading and writing csv fil
import pandas as pd
# seaborn and matplotlib for data visualization
import seaborn as sns
import matplotlib.pyplot as plt
%matplotlib inline
```

1.1 Loading the data

The dataset i will use contain information from 1962 and later about civil aviation accidents and selected incidents within the United States, its territories and possessions, and in international waters.

- Dataset link for download or accesss:[https://www.kaggle.com/datasets/khsamaha/aviation-accident-database-synopses (https://www.kaggle.com/datasets/khsamaha/aviation-accident-database-synopses)](https://www.kaggle.com/datasets/khsamaha/aviation-accident-database-synopses)

In [2]:

```
# Reading data from csv file and create data frame to be used
#index_col=0 will set the first column in the csv file as the index of the
#low_memory=False determine the best dtype for each column after reading en

aviationData= pd.read_csv('AviationData.csv',encoding='ISO-8859-1',low_memo
```

In [3]:
```
aviationData
```

Out[3]:

	Event.Id	Investigation.Type	Accident.Number	Event.Date	Location	Cou
0	20001218X45444	Accident	SEA87LA080	10/24/1948	MOOSE CREEK, ID	U S
1	20001218X45447	Accident	LAX94LA336	7/19/1962	BRIDGEPORT, CA	U S
2	20061025X01555	Accident	NYC07LA005	8/30/1974	Saltville, VA	U S
3	20001218X45448	Accident	LAX96LA321	6/19/1977	EUREKA, CA	U S
4	20041105X01764	Accident	CHI79FA064	8/2/1979	Canton, OH	U S
...	...	...	...	...	...	
88884	2.02E+13	Accident	ERA23LA093	12/26/2022	Annapolis, MD	U S
88885	2.02E+13	Accident	ERA23LA095	12/26/2022	Hampton, NH	U S
88886	2.02E+13	Accident	WPR23LA075	12/26/2022	Payson, AZ	U S
88887	2.02E+13	Accident	WPR23LA076	12/26/2022	Morgan, UT	U S
88888	2.02E+13	Accident	ERA23LA097	12/29/2022	Athens, GA	U S

88889 rows × 31 columns

1.2 previewing the dataset

In [4]:
```
# Preview the first 5 rows of the data
aviationData.head()
```

Out[4]:

	Event.Id	Investigation.Type	Accident.Number	Event.Date	Location	Country
0	20001218X45444	Accident	SEA87LA080	10/24/1948	MOOSE CREEK, ID	United States
1	20001218X45447	Accident	LAX94LA336	7/19/1962	BRIDGEPORT, CA	United States
2	20061025X01555	Accident	NYC07LA005	8/30/1974	Saltville, VA	United States
3	20001218X45448	Accident	LAX96LA321	6/19/1977	EUREKA, CA	United States
4	20041105X01764	Accident	CHI79FA064	8/2/1979	Canton, OH	United States

5 rows × 31 columns

In [5]:
```
# Preview the last 5 rows of the data
aviationData.tail()
```

Out[5]:

	Event.Id	Investigation.Type	Accident.Number	Event.Date	Location	Country	Latitu
88884	2.02E+13	Accident	ERA23LA093	12/26/2022	Annapolis, MD	United States	N
88885	2.02E+13	Accident	ERA23LA095	12/26/2022	Hampton, NH	United States	N
88886	2.02E+13	Accident	WPR23LA075	12/26/2022	Payson, AZ	United States	34152
88887	2.02E+13	Accident	WPR23LA076	12/26/2022	Morgan, UT	United States	N
88888	2.02E+13	Accident	ERA23LA097	12/29/2022	Athens, GA	United States	N

5 rows × 31 columns

1.3 Inspecting the data

In [6]:
```
# Accesing information about the dataset
aviationData.info(verbose=False)
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 88889 entries, 0 to 88888
Columns: 31 entries, Event.Id to Publication.Date
dtypes: float64(5), object(26)
memory usage: 21.0+ MB
```

- The above dataset is a dataframe and has 88889 entries and there are a number of columns with missing values.
- The data has float 64 and object data type.
- The memory usage is 23.5+ MB

In [7]:
```
# Checking the umber of rows and columns
aviationData.shape
```

Out[7]:
```
(88889, 31)
```

In [8]:
```
# Checking the columns
aviationData.columns
```

Out[8]:
```
Index(['Event.Id', 'Investigation.Type', 'Accident.Number', 'Event.Date',
       'Location', 'Country', 'Latitude', 'Longitude', 'Airport.Code',
       'Airport.Name', 'Injury.Severity', 'Aircraft.damage',
       'Aircraft.Category', 'Registration.Number', 'Make', 'Model',
       'Amateur.Built', 'Number.of.Engines', 'Engine.Type', 'FAR.Descript
ion',
       'Schedule', 'Purpose.of.flight', 'Air.carrier', 'Total.Fatal.Injur
ies',
       'Total.Serious.Injuries', 'Total.Minor.Injuries', 'Total.Uninjure
d',
       'Weather.Condition', 'Broad.phase.of.flight', 'Report.Status',
       'Publication.Date'],
      dtype='object')
```

In [9]:

```
# Calculating the summary statistics
aviationData.describe()
```

Out[9]:

	Number.of.Engines	Total.Fatal.Injuries	Total.Serious.Injuries	Total.Minor.Injuries	Total.
count	82805.000000	77488.000000	76379.000000	76956.000000	829
mean	1.146585	0.647855	0.279881	0.357061	
std	0.446510	5.485960	1.544084	2.235625	
min	0.000000	0.000000	0.000000	0.000000	
25%	1.000000	0.000000	0.000000	0.000000	
50%	1.000000	0.000000	0.000000	0.000000	
75%	1.000000	0.000000	0.000000	0.000000	
max	8.000000	349.000000	161.000000	380.000000	6

In [10]:

```
# Calculating the summary statistics of categorical columns
aviationData.describe(include='object')
```

Out[10]:

	Event.Id	Investigation.Type	Accident.Number	Event.Date	Location	Country
count	88889	88889	88889	88889	88837	88663
unique	84441	2	88863	14782	27758	219
top	2.02E+13	Accident	CEN23MA034	6/30/1984	ANCHORAGE, AK	United States
freq	3537	85015	2	25	434	82248

4 rows × 26 columns

In [11]:

```
# Checking for column data types
type(aviationData)
```

Out[11]: pandas.core.frame.DataFrame

1.4 Data Cleaning

This is the process of identifying,correcting or removing irrelevant,incomplete or inaccurate and duplicated data from a dataset.

In [12]:

```
# creating a new copy of a Dataframe

aviationData1 = aviationData.copy(deep=True)
```

1.4.1 Missing Values

There are various ways of handling missing values and this include Dropping an entire row or column if it has too many missing values or for columns with few missing value you impute with mean,mode,median or a placeholder.

In [13]:
```
#Detecting missing values
#checking total number of NaN values
aviationData1.isna().sum().sum()
```

Out[13]:
```
564742
```

In [14]:
```
#sorting the missing values in ascending order
aviationData1.isna().sum().sort_values(ascending= False)
```

Out[14]:
```
Schedule                  76307
Air.carrier               72241
FAR.Description           56866
Aircraft.Category         56602
Longitude                 54516
Latitude                  54507
Airport.Code              38640
Airport.Name              36099
Broad.phase.of.flight     27165
Publication.Date          13771
Total.Serious.Injuries    12510
Total.Minor.Injuries      11933
Total.Fatal.Injuries      11401
Engine.Type                7077
Report.Status              6381
Purpose.of.flight          6192
Number.of.Engines          6084
Total.Uninjured            5912
Weather.Condition          4492
Aircraft.damage            3194
Registration.Number        1317
Injury.Severity            1000
Country                     226
Amateur.Built               102
Model                        92
Make                         63
Location                     52
Event.Date                    0
Accident.Number               0
Investigation.Type            0
Event.Id                      0
dtype: int64
```

In [15]:
```
# Dropping irrelevant columns
#Explanation: We do not need the attribute in the analysis
aviationData1.drop(['Air.carrier','Aircraft.Category','Schedule','FAR.Descr
        'Publication.Date','Engine.Type','Report.Status','Number.of.Engine
```

In [16]:
```
# Checking the length of the columns
# the number of columns has dropped t0 18 after dropping irrelevant columns
len(aviationData1.columns)
```

Out[16]: 18

In [17]:
```
aviationData1.columns
```

Out[17]:
```
Index(['Event.Id', 'Investigation.Type', 'Accident.Number', 'Event.Date',
       'Location', 'Country', 'Injury.Severity', 'Aircraft.damage',
       'Registration.Number', 'Make', 'Model', 'Purpose.of.flight',
       'Total.Fatal.Injuries', 'Total.Serious.Injuries',
       'Total.Minor.Injuries', 'Total.Uninjured', 'Weather.Condition',
       'Broad.phase.of.flight'],
      dtype='object')
```

In [18]:
```
#selecting numeric columns to replace NaN with median,columns like Total.Se
#,Total.Fatal.Injuries and Total.Uninjured.
#we replace NaN with Median of the values
aviationData1.fillna(aviationData1.select_dtypes(include='number').median()
aviationData1.head(10)
```

Out[18]:

	Event.Id	Investigation.Type	Accident.Number	Event.Date	Location	Count
0	20001218X45444	Accident	SEA87LA080	10/24/1948	MOOSE CREEK, ID	Unite State
1	20001218X45447	Accident	LAX94LA336	7/19/1962	BRIDGEPORT, CA	Unite State
2	20061025X01555	Accident	NYC07LA005	8/30/1974	Saltville, VA	Unite State
3	20001218X45448	Accident	LAX96LA321	6/19/1977	EUREKA, CA	Unite State
4	20041105X01764	Accident	CHI79FA064	8/2/1979	Canton, OH	Unite State
5	20170710X52551	Accident	NYC79AA106	9/17/1979	BOSTON, MA	Unite State
6	20001218X45446	Accident	CHI81LA106	8/1/1981	COTTON, MN	Unite State
7	20020909X01562	Accident	SEA82DA022	1/1/1982	PULLMAN, WA	Unite State
8	20020909X01561	Accident	NYC82DA015	1/1/1982	EAST HANOVER, NJ	Unite State
9	20020909X01560	Accident	MIA82DA029	1/1/1982	JACKSONVILLE, FL	Unite State

In [19]:
```
# getting total number of NaN values in a Dataframe
aviationData1.isna().sum().sum()
```

Out[19]: 43793

In [20]:
```
# 1.4.2 Check for duplicates
aviationData1.duplicated().sum()
```

Out[20]: 0

The data has no duplicated rows

In [21]:
```
#sorting the missing values in ascending order
missing_values= aviationData1.isna().sum().sort_values(ascending= False)
# calculate percentage of the missing values
percentage_missV = (aviationData1.isna().sum() / len(aviationData1)).sort_v
# store in a dataframe
missing = pd.DataFrame({"Missing Values": missing_values, "Percentage(%)":
missing
```

Out[21]:

	Missing Values	Percentage(%)
Broad.phase.of.flight	27165	0.305606
Purpose.of.flight	6192	0.069660
Weather.Condition	4492	0.050535
Aircraft.damage	3194	0.035932
Registration.Number	1317	0.014816
Injury.Severity	1000	0.011250
Country	226	0.002542
Model	92	0.001035
Make	63	0.000709
Location	52	0.000585
Total.Fatal.Injuries	0	0.000000
Total.Serious.Injuries	0	0.000000
Total.Minor.Injuries	0	0.000000
Total.Uninjured	0	0.000000
Event.Date	0	0.000000
Accident.Number	0	0.000000
Investigation.Type	0	0.000000
Event.Id	0	0.000000

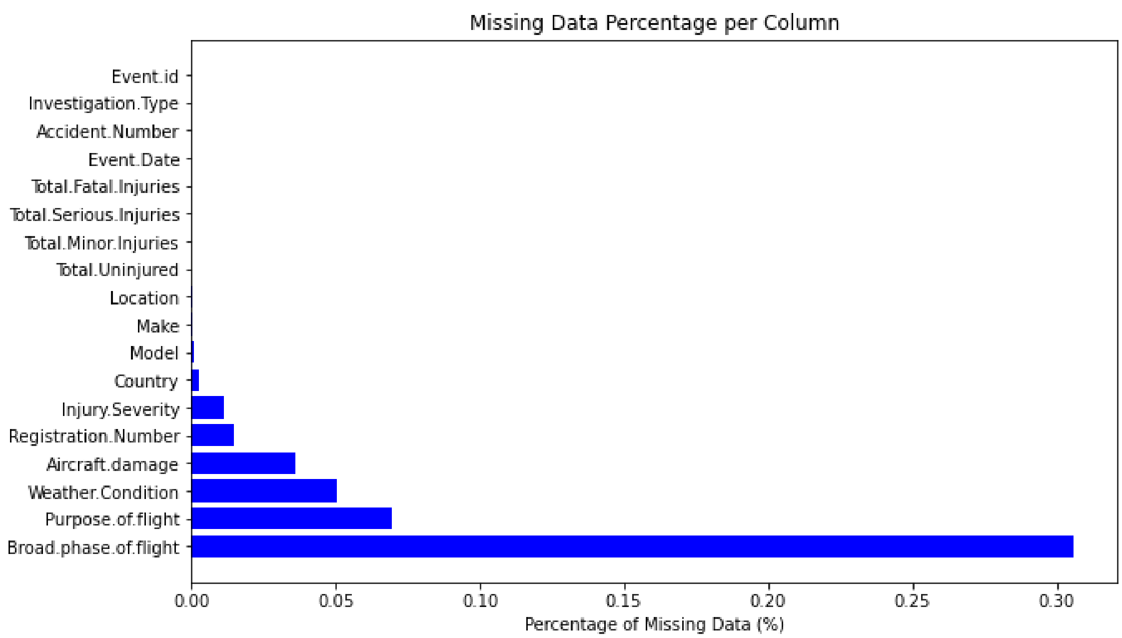
In [22]:

```
# Bar chart to show the Missing data in percentage for each column
columns = ['Broad.phase.of.flight','Purpose.of.flight', 'Weather.Condition'
'Model','Make', 'Location', 'Total.Uninjured', 'Total.Minor.Injuries', 'Tot
'Accident.Number', 'Investigation.Type','Event.id']

missing_values = [27165,6192, 4492, 3194, 1317, 1000, 226, 92, 63, 52, 0, 0
percentages = [0.305606,0.069660, 0.050535, 0.035932, 0.014816, 0.011250, 0
               0.000585, 0.000000, 0.000000, 0.000000, 0.000000, 0.000000,

# Create bar chart
fig,ax = plt.subplots(figsize=(10, 6))
#plot bargraph
ax.barh(columns, percentages, color='blue')
#labelling the axis
ax.set_xlabel('Percentage of Missing Data (%)')
#label the title
ax.set_title('Missing Data Percentage per Column')

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



In [23]:

```
# Broad phase of flight column
# Getting normalized value counts of Broad phase of flighr with NaN include
aviationData1['Broad.phase.of.flight'].value_counts(normalize=True,dropna=F
# want to replace the NaN values with the Unknown
aviationData1['Broad.phase.of.flight'].fillna('Unknown', inplace=True)
# Checking if the NaN has been replaced by unknown
aviationData1['Broad.phase.of.flight'].value_counts(normalize=True,dropna=F
```

Out[23]:

```
Unknown        0.311771
Landing        0.173565
Takeoff        0.140546
Cruise         0.115526
Maneuvering    0.091620
Approach       0.073642
Climb          0.022882
Taxi           0.022027
Descent        0.021229
Go-around      0.015221
Standing       0.010631
Other          0.001339
Name: Broad.phase.of.flight, dtype: float64
```

In [24]:
```
#Purpose of flight column:
# Getting normalized value counts of purpose of flight with NaN included
aviationData1['Purpose.of.flight'].value_counts(normalize=True,dropna=False
# Replace NaN with Unknown
aviationData1['Purpose.of.flight'].fillna('Unknown', inplace=True)
# Checking if the NaN has been replaced by unknown
aviationData1['Purpose.of.flight'].value_counts(normalize=True,dropna=False
```

Out[24]:
```
Personal                     0.556289
Unknown                      0.146182
Instructional                0.119261
Aerial Application           0.053010
Business                     0.045202
Positioning                  0.018517
Other Work Use               0.014220
Ferry                        0.009135
Aerial Observation           0.008932
Public Aircraft              0.008100
Executive/corporate          0.006221
Flight Test                  0.004556
Skydiving                    0.002047
External Load                0.001384
Public Aircraft - Federal    0.001181
Banner Tow                   0.001136
Air Race show                0.001114
Public Aircraft - Local      0.000832
Public Aircraft - State      0.000720
Air Race/show                0.000664
Glider Tow                   0.000596
Firefighting                 0.000450
Air Drop                     0.000124
ASHO                         0.000067
PUBS                         0.000045
PUBL                         0.000011
Name: Purpose.of.flight, dtype: float64
```

In [25]:
```
# Weather conditin column:
# VMC-Conditions suitable for visual flying.
# IMC-Conditions that require instrument flying.
# UNK- Weather condition unknown or unrecorded
aviationData1['Weather.Condition'].value_counts(normalize=True,dropna=False
# calculate the mode in weather.condition
most_frequent_conditions = aviationData1['Weather.Condition'].mode()
# Replace the NaN values with the most frequent condition
aviationData1['Weather.Condition'].fillna('VMC',inplace=True)
#changing Unk to uppercase
aviationData1['Weather.Condition'] = aviationData1['Weather.Condition'].str
aviationData1['Weather.Condition'].value_counts(normalize=True,dropna=False
```

Out[25]:
```
VMC    0.920193
IMC    0.067230
UNK    0.012577
Name: Weather.Condition, dtype: float64
```

In [26]:

```
# Injury severity column:
aviationData1['Injury.Severity'].value_counts().reset_index()
```

Out[26]:

	index	Injury.Severity
0	Non-Fatal	67357
1	Fatal(1)	6167
2	Fatal	5262
3	Fatal(2)	3711
4	Incident	2219
...	...	...
104	Fatal(270)	1
105	Fatal(144)	1
106	Fatal(206)	1
107	Fatal(141)	1
108	Fatal(121)	1

109 rows × 2 columns

In [27]:

```
# to remove brackets and numbers in Fatal
aviationData1['Injury.Severity']=aviationData1['Injury.Severity'].str.repla
```

In [28]:

```
# Injury severity column:
aviationData1['Injury.Severity'].value_counts().reset_index()
```

Out[28]:

	index	Injury.Severity
0	Non-Fatal	67357
1	Fatal	17826
2	Incident	2219
3	Minor	218
4	Serious	173
5	Unavailable	96

In [29]:
```
aviationData1.isna().sum()
```

Out[29]:
```
Event.Id                    0
Investigation.Type          0
Accident.Number             0
Event.Date                  0
Location                   52
Country                   226
Injury.Severity          1000
Aircraft.damage          3194
Registration.Number      1317
Make                       63
Model                      92
Purpose.of.flight           0
Total.Fatal.Injuries        0
Total.Serious.Injuries      0
Total.Minor.Injuries        0
Total.Uninjured             0
Weather.Condition           0
Broad.phase.of.flight       0
dtype: int64
```

In [30]:
```
aviationData1['Aircraft.damage'].value_counts().reset_index()
```

Out[30]:

	index	Aircraft.damage
0	Substantial	64148
1	Destroyed	18623
2	Minor	2805
3	Unknown	119

In [31]:
```
# Calculate the mode of Aircraft damage
aircraft_damage_mode= aviationData1['Aircraft.damage'].mode()[0]
```

In [32]:
```
# Replace the missing values in Aircraft damage with the mode
aviationData1['Aircraft.damage'].fillna(aircraft_damage_mode,inplace=True)
```

In [33]:
```
aviationData1.isna().sum().sort_values(ascending= False)
```

Out[33]:
```
Registration.Number       1317
Injury.Severity           1000
Country                    226
Model                       92
Make                        63
Location                    52
Aircraft.damage              0
Investigation.Type           0
Accident.Number              0
Event.Date                   0
Broad.phase.of.flight        0
Weather.Condition            0
Purpose.of.flight            0
Total.Fatal.Injuries         0
Total.Serious.Injuries       0
Total.Minor.Injuries         0
Total.Uninjured              0
Event.Id                     0
dtype: int64
```

In [34]:
```
# Dropping all the columns with missing values
aviationData1.dropna(inplace=True)
```

In [35]:
```
#to check no missing values
aviationData1.isna().sum()
```

Out[35]:
```
Event.Id                  0
Investigation.Type        0
Accident.Number           0
Event.Date                0
Location                  0
Country                   0
Injury.Severity           0
Aircraft.damage           0
Registration.Number       0
Make                      0
Model                     0
Purpose.of.flight         0
Total.Fatal.Injuries      0
Total.Serious.Injuries    0
Total.Minor.Injuries      0
Total.Uninjured           0
Weather.Condition         0
Broad.phase.of.flight     0
dtype: int64
```

In [36]:

```
aviationData1.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 86280 entries, 0 to 88888
Data columns (total 18 columns):
 #   Column                  Non-Null Count  Dtype  
---  ------                  --------------  -----  
 0   Event.Id                86280 non-null  object 
 1   Investigation.Type      86280 non-null  object 
 2   Accident.Number         86280 non-null  object 
 3   Event.Date              86280 non-null  object 
 4   Location                86280 non-null  object 
 5   Country                 86280 non-null  object 
 6   Injury.Severity         86280 non-null  object 
 7   Aircraft.damage         86280 non-null  object 
 8   Registration.Number     86280 non-null  object 
 9   Make                    86280 non-null  object 
 10  Model                   86280 non-null  object 
 11  Purpose.of.flight       86280 non-null  object 
 12  Total.Fatal.Injuries    86280 non-null  float64
 13  Total.Serious.Injuries  86280 non-null  float64
```

In [37]:

```
# To check remainning columns
aviationData1.columns
```

Out[37]:

```
Index(['Event.Id', 'Investigation.Type', 'Accident.Number', 'Event.Date',
       'Location', 'Country', 'Injury.Severity', 'Aircraft.damage',
       'Registration.Number', 'Make', 'Model', 'Purpose.of.flight',
       'Total.Fatal.Injuries', 'Total.Serious.Injuries',
       'Total.Minor.Injuries', 'Total.Uninjured', 'Weather.Condition',
       'Broad.phase.of.flight'],
      dtype='object')
```

1.5 Exporting the cleaned dataset

In [38]:

```
# Exporting of dataframe to csv file
aviationData1.to_csv('AviationData1.csv')
```

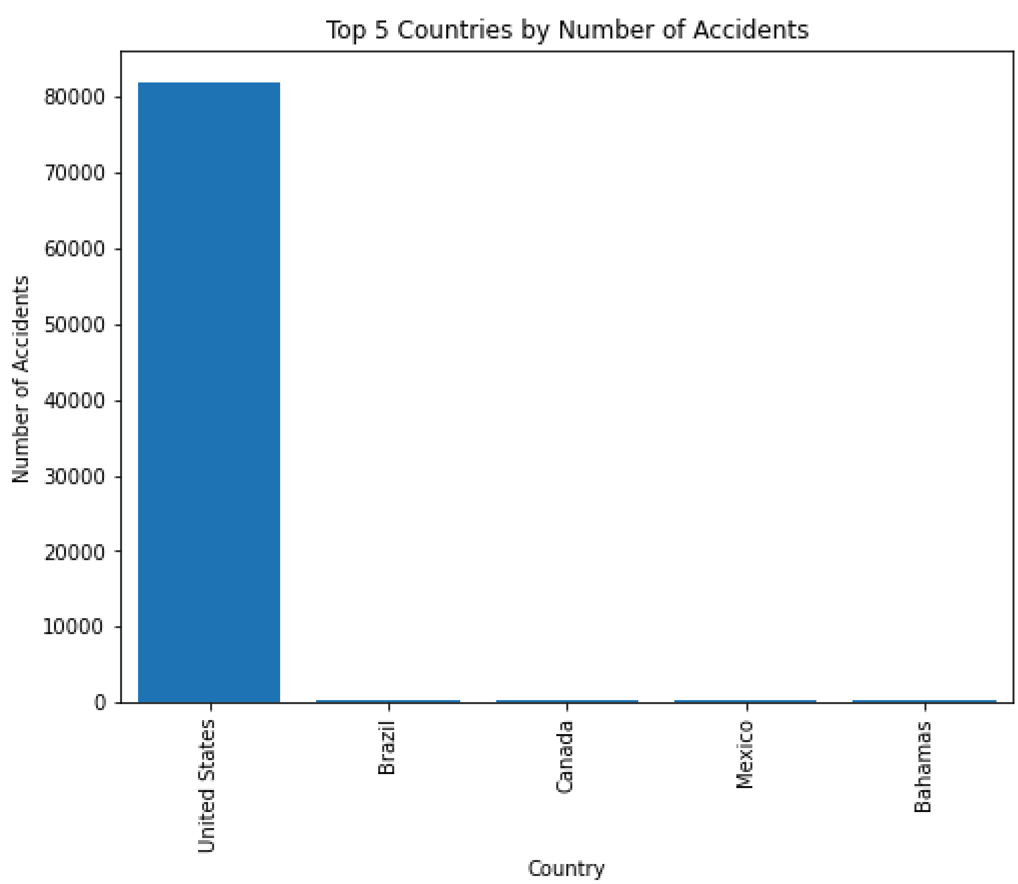
1.6 Answering questions

Which country had the highest number of accidents?

In [39]:
```
# group the accidents by country
accidents_by_country = aviationData1['Country'].value_counts().head()
accidents_by_country
```

Out[39]:
```
United States    82041
Brazil             314
Canada             279
Mexico             244
Bahamas            207
Name: Country, dtype: int64
```

In [40]:
```
# Plotting accident trends by country
fig,ax= plt.subplots(figsize=(8, 6))
# create a bar chart
ax.bar(accidents_by_country.index,accidents_by_country.values)
accidents_by_country.plot(kind='bar')
ax.set_title('Top 5 Countries by Number of Accidents')
ax.set_xlabel('Country')
ax.set_ylabel('Number of Accidents')
plt.show();
```



The above graph shows United states had the highest cases of accidents followed by Brazil and Canada

Which country had the highest cases of Fatal injuries?

In [41]:
```
#calculating fatal injuries by country
Fatal_injuries_by_country = aviationData1.groupby('Country')['Total.Fatal.I
# Sorting by the total fatal injuries in descending order
Fatal_injuries_by_country = Fatal_injuries_by_country.sort_values(by='Total
Fatal_injuries_by_country
```

Out[41]:

	Country	Total.Fatal.Injuries
181	United States	30152.0
20	Brazil	753.0
25	Canada	627.0
54	France	530.0
77	Indonesia	524.0
...	...	...
154	Somalia	0.0
58	Gambia	0.0
56	French Polynesia	0.0
8	Aruba	0.0
145	San Juan Islands	0.0

From the above,most Total Fatal injuries occured in United States

In [42]:
```
#countries with lowest accident rates
lowest_accident_rate = aviationData1['Country'].value_counts()
least_country=lowest_accident_rate[lowest_accident_rate ==1]
least_country.head()
```

Out[42]:
```
Unknown                1
Isle of Man            1
Antigua and Barbuda    1
Lebanon                1
BLOCK 651A             1
Name: Country, dtype: int64
```

In [43]:
```
#We Calculate the below:
#count of accidents
#Total fatalities
#Total serious injuries
#Total uninjured
#Group by Location to identify Location with highest number of accidents
Location_analysis =aviationData1.groupby(['Location']).agg({ 'Event.Id': 'c
'Total.Serious.Injuries': 'sum','Total.Minor.Injuries': 'sum','Total.Uninju
# Renaming columns
Location_analysis.columns = ['Location', 'Accident_Count', 'Total_Fatal_Inj
                              'Total_Serious_Injuries', 'Total_Minor_Ir

# Sort by accident count to find the safest private model
Location_analysis_sorted = Location_analysis.sort_values(by='Accident_Count
Location_analysis_sorted
```

Out[43]:

	Location	Accident_Count	Total_Fatal_Injuries	Total_Serious_Injuries	Total_Minor_Inju
11087	Irving, TX	1	0.0	0.0	
11086	Irvine, CA	1	0.0	0.0	
11085	Ironton, OH	1	0.0	2.0	
11084	Ironside, OR	1	0.0	1.0	
13127	La Ronge, SK,	1	0.0	0.0	

In which weather condition were those flights conducted?

In [44]:
```
# Checking the distribution of weather condition in the dataset
Weather_condition_aviation= aviationData1['Weather.Condition'].value_counts
Weather_condition_aviation
```

Out[44]:

	index	Weather.Condition
0	VMC	79574
1	IMC	5788
2	UNK	918

From the above,most flights occured under VMC(Visual Meteorological Condition)

In which weather conditions did most fatal accidents occur?

In [45]:

```
# Group by Weather conditions
#count of accidents
#Total fatalities
#Total serious injuries
#Total uninjured
# Group by weather conditions to access in which weather does most accident
Weather_analysis =aviationData1.groupby(['Weather.Condition']).agg({ 'Event
'Total.Serious.Injuries': 'sum','Total.Minor.Injuries': 'sum','Total.Uninju
# Sort by weather condition
Weather_analysis_sorted = Weather_analysis.sort_values(by='Weather.Conditio
Weather_analysis_sorted
```

Out[45]:

	Weather.Condition	Event.Id	Total.Fatal.Injuries	Total.Serious.Injuries	Total.Minor.Injuries
2	VMC	79574	30740.0	18080.0	23371.0
1	UNK	918	1427.0	244.0	261.0
0	IMC	5788	9130.0	1749.0	2193.0

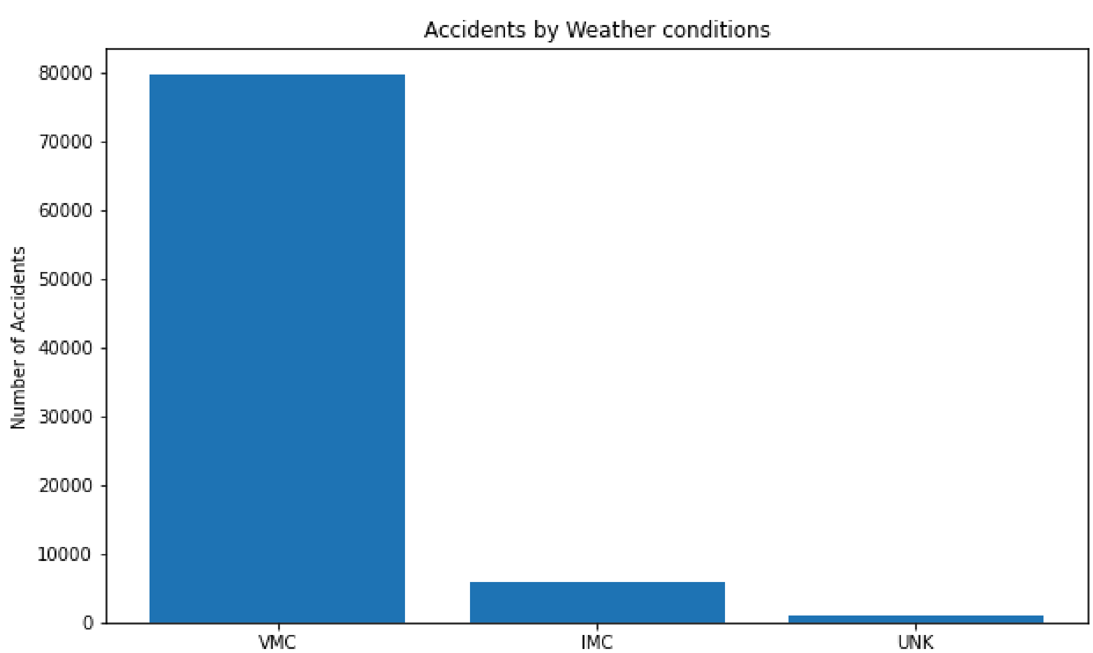
In [46]:

```
#Accident rates by weather conditions

accidents_by_weather= aviationData1['Weather.Condition'].value_counts().hea

# Plotting accident trends by weather conditions

fig,ax =plt.subplots(figsize=(10, 6))
ax.bar(accidents_by_weather.index,accidents_by_weather.values)
ax.set_title('Accidents by Weather conditions')
ax.set_xlabel('Weather Condition')
ax.set_ylabel('Number of Accidents')
plt.show()
```



From the above,most accidents occurred under VMC conditions

Which injuries occured most?

In [47]:

```
# calculating total sum of injury types
injury_types = {'Fatal Injuries': aviationData1['Total.Fatal.Injuries'].sur
'Minor Injuries': aviationData1['Total.Minor.Injuries'].sum(),'Uninjured':
injury_types
```

Out[47]:

```
{'Fatal Injuries': 41297.0,
 'Serious Injuries': 20073.0,
 'Minor Injuries': 25825.0,
 'Uninjured': 424584.0}
```

From the above,Fatal injuries had the most cases.

Which aircraft make had the highest Injury severity?

In [48]:

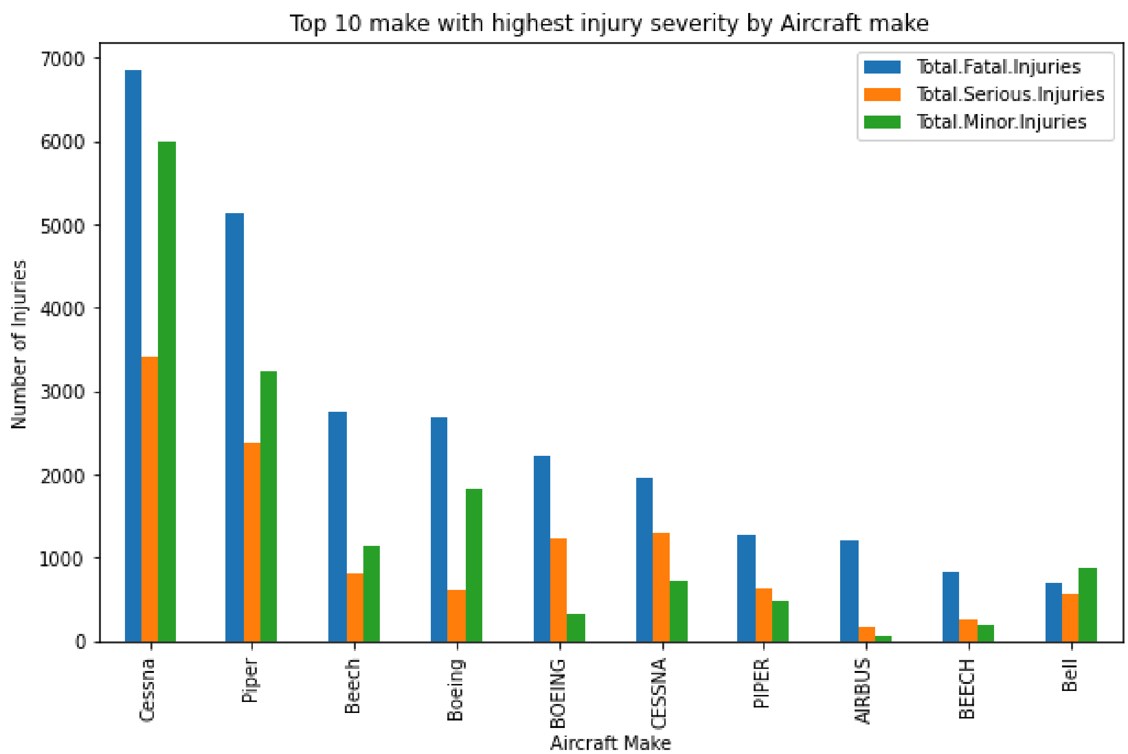
```
#Injury severity based on Aircraft make

injury_severity = aviationData1.groupby('Make')[['Total.Fatal.Injuries','To
                                                 'Total.Minor.Injuries']].su

#sort by Total Fatal injury
highest_injury_by_make= injury_severity.sort_values(by='Total.Fatal.Injurie

# Plotting accident trends by weather conditions

fig,ax = plt.subplots(figsize=(10, 6))
highest_injury_by_make.plot(kind='bar',ax=ax)
ax.set_title('Top 10 make with highest injury severity by Aircraft make')
ax.set_xlabel('Aircraft Make')
ax.set_ylabel('Number of Injuries')
plt.show();
```



In [49]:

```
#Grouping by make to sum the total Fatal injuries
Fatal_injuries_by_make = aviationData1.groupby(['Make'])['Total.Fatal.Injur
# Sorting by the total fatal injuries in ascending order to find the make w
Fatal_injuries_by_make_sorted = Fatal_injuries_by_make.sort_values(by='Tota
Fatal_injuries_by_make_sorted
```

Out[49]:

	Make	Total.Fatal.Injuries
4090	KITFOX	0.0
4891	Malone Henry O	0.0
4888	Malechek	0.0
4887	Mahre	0.0
4886	Mahoney	0.0
...	...	...
695	BOEING	2216.0
1056	Boeing	2695.0
926	Beech	2753.0
5758	Piper	5124.0
1553	Cessna	6847.0

8181 rows × 2 columns

Which aircraft make had the highest Injury severity?

In [50]:

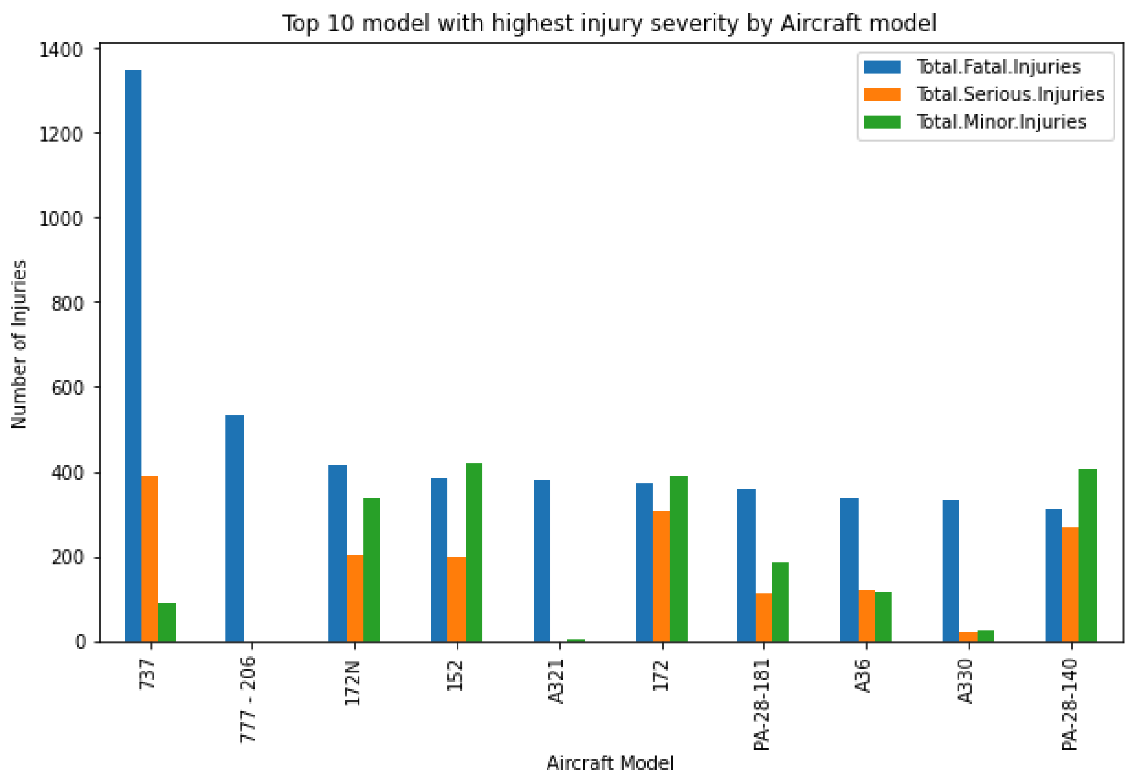
```
#Injury severity based on Aircraft make

injury_severity = aviationData1.groupby('Model')[['Total.Fatal.Injuries','T
                                                 'Total.Minor.Injuries']].su

#sort by Total Fatal injury
highest_injury_by_model= injury_severity.sort_values(by='Total.Fatal.Injuri

# Plotting accident trends by weather conditions

fig,ax= plt.subplots(figsize=(10, 6))
highest_injury_by_model.plot(kind='bar',ax=ax)
ax.set_title('Top 10 model with highest injury severity by Aircraft model')
ax.set_xlabel('Aircraft Model')
ax.set_ylabel('Number of Injuries')
plt.show();
```



From the above Cessna Make with Model 737 caused the highest injury severity.The above aircraft Models should be avoided as they potray Fatal and serious injuries.

Which Top 10 Aircraft models has the highest Accident rates?

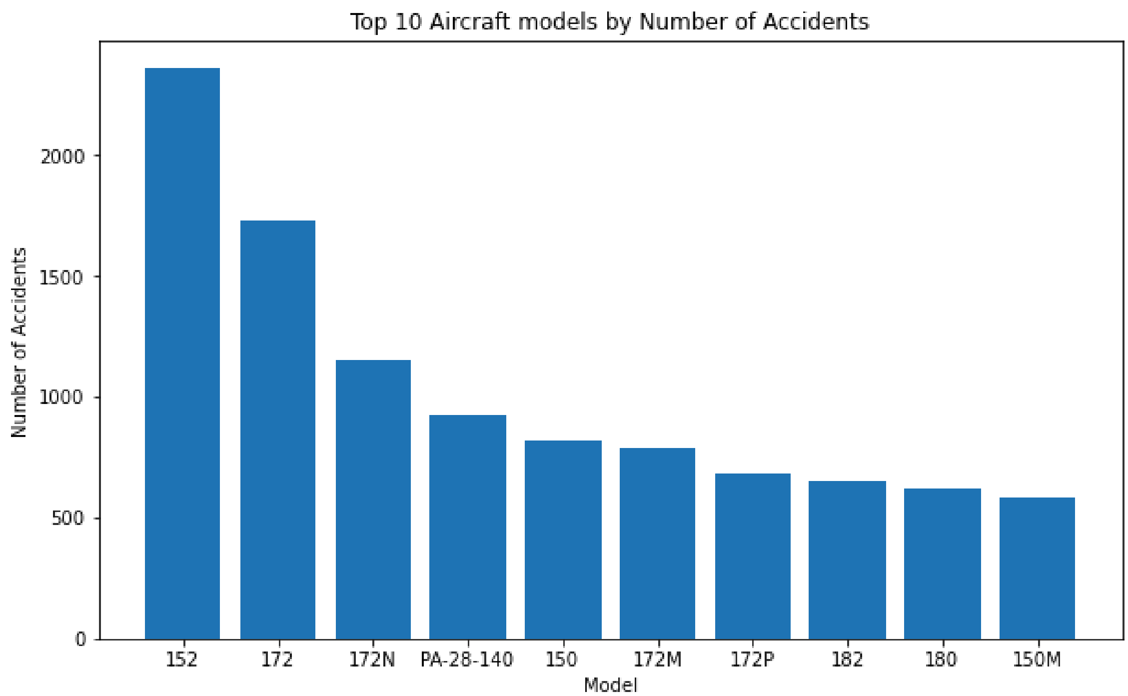
In [51]:

```
#Accident rates by aircraft model

accidents_by_model= aviationData1['Model'].value_counts().head(10)

# Plotting accident trends by model

fig,ax =plt.subplots(figsize=(10, 6))
ax.bar(accidents_by_model.index,accidents_by_model.values)
ax.set_title('Top 10 Aircraft models by Number of Accidents')
ax.set_xlabel('Model')
ax.set_ylabel('Number of Accidents')
plt.show()
```



When making a decision on the aircraft models to purchase,have a look out on the above models in the bar graph as they cause high number of accidents and avoid buying them.Check out for models with low risk of accidents.

Which Aircraft make has lowest risk of injuries?

In [52]:

```
#Grouping by make to sum the total serious injuries
serious_injuries_by_make = aviationData1.groupby(['Make'])['Total.Serious.I
# Sorting by the total serious injuries in descending order to find the mak
serious_injuries_by_make_sorted = serious_injuries_by_make.sort_values(by='
serious_injuries_by_make_sorted
```

Out[52]:

	Make	Total.Serious.Injuries
1553	Cessna	3418.0
5758	Piper	2381.0
1307	CESSNA	1292.0
695	BOEING	1221.0
926	Beech	822.0
...	...	...
3187	HALDERMAN, FLOYD G.	0.0
3183	HAGERTY	0.0
3181	HAEUSSLER RAY	0.0
3179	HACHEM ZACHERY S	0.0
8180	unknown	0.0

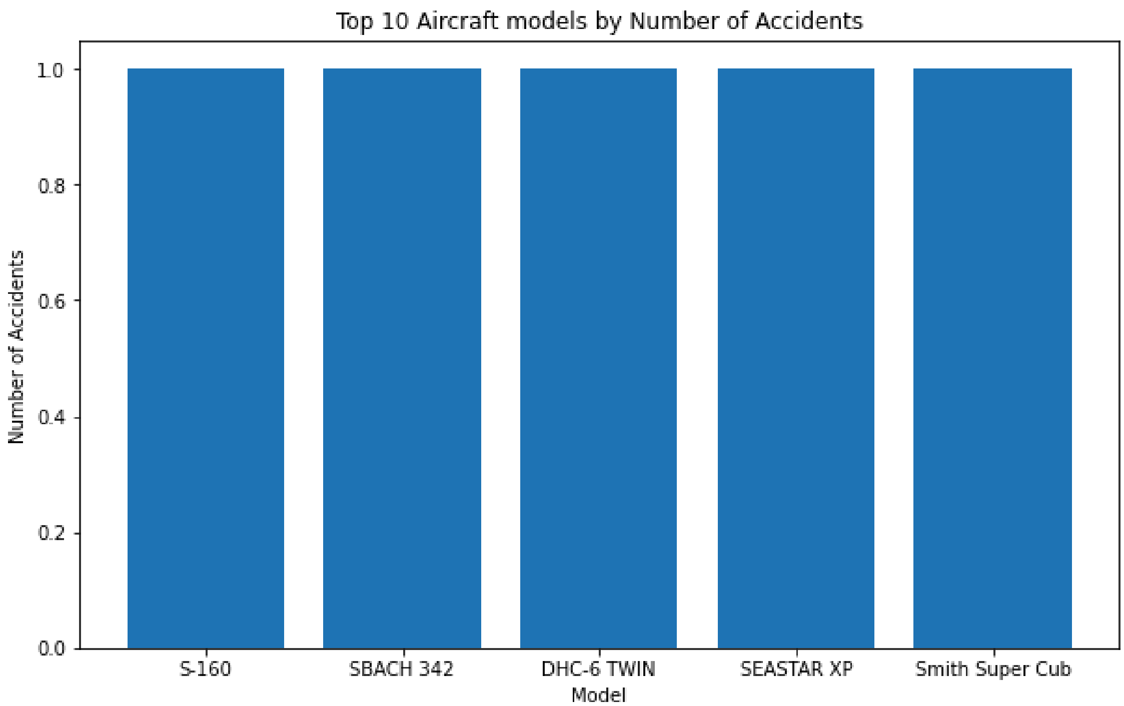
8181 rows × 2 columns

Cessna and Piper record the highest number of serious injuries,the ones below in the list are the ones with lowest risk

In [53]:
```
#Accident rates by aircraft model(Lowest risk)
#Checking the aircraft model with lowest risk of accident
accidents_by_model= aviationData1['Model'].value_counts().tail(5)

# Plotting accident trends by model

fig,ax = plt.subplots(figsize=(10, 6))
ax.bar(accidents_by_model.index,accidents_by_model.values)
ax.set_title('Top 10 Aircraft models by Number of Accidents')
ax.set_xlabel('Model')
ax.set_ylabel('Number of Accidents')
plt.show()
```



From the above,i can see that the aircraft with the lowest risk of accidents is Naval Aircraft Factory N3N which is used for Personal purposes and so it is for Private Use.

In [54]:
```
# Group by 'Make' and 'Model'
#calculate for each aircraft model:
#count of accidents
#Total fatalities
#Total serious injuries
#Total uninjured
aircraft_analysis = aviationData1.groupby(['Make', 'Model']).agg({ 'Event.I
'Total.Serious.Injuries': 'sum','Total.Minor.Injuries': 'sum','Total.Uninju
# Renaming columns
aircraft_analysis.columns = ['Make', 'Model', 'Accident.Count', 'Total.Fata
                              'Total.Serious.Injuries', 'Total.Minor.In


# Sort by accident count to find the most frequent aircraft accidents
aircraft_analysis_sorted = aircraft_analysis.sort_values(by='Accident.Count
aircraft_analysis_sorted
```

Out[54]:

	Make	Model	Accident.Count	Total.Fatal.Injuries	Total.Serious.Injuries	Total.
5480	Cessna	152	2161	342.0	166.0	
5502	Cessna	172	1239	202.0	188.0	
5545	Cessna	172N	988	351.0	145.0	
14661	Piper	PA-28-140	807	276.0	219.0	
5455	Cessna	150	712	75.0	102.0	
...	...	...	...	...	...	
8146	Eurocopter Deutschland	BK-117-B2	1	0.0	0.0	
8147	Eurocopter Deutschland	BK117	1	0.0	0.0	
8148	Eurocopter Deutschland	BK117C1	1	4.0	0.0	
8149	Eurocopter Deutschland	BO-105 CBS5	1	3.0	0.0	
19647	unknown	kit	1	0.0	0.0	

19648 rows × 7 columns

From the above it shows that the Aircraft with lower risk of accidents are the ones outside the top of the list with lower accidents and high number of unninjured.

Which private model has caused few accidents and low fatal rates?

In [55]:

```
#cheking private flights based on purpose of flight column
private_flights = aviationData1[aviationData1['Purpose.of.flight'].str.cont
#count of accidents
#Total fatalities
#Total serious injuries
#Total uninjured
#Group by make and model by private flights
private_analysis =private_flights.groupby(['Make', 'Model']).agg({ 'Event.I
'Total.Serious.Injuries': 'sum','Total.Minor.Injuries': 'sum','Total.Uninju
# Renaming columns
private_analysis.columns = ['Make', 'Model', 'Accident.Count', 'Total_Fatal
                                  'Total.Serious.Injuries', 'Total.Minor.Ir

# Sort by accident count to find the safest private model
private_analysis_sorted = private_analysis.sort_values(by='Accident.Count',
private_analysis_sorted
```

Out[55]:

	Make	Model	Accident.Count	Total_Fatal.Injuries	Total.Serious.Injuries
5358	GRAZHOPPER	TRIKE	1	1.0	1.0
5360	GREAT LAKES	2T-1A	1	0.0	0.0
5361	GREAT LAKES	2T-1A-1	1	0.0	2.0
5364	GREEN GARY E	THORP T-18	1	1.0	0.0
5365	GREEN L/GILNER D	RV-4	1	0.0	0.0
5366	GREENE R/GREENE S	RANS S 17	1	0.0	1.0
5367	GREG HOBBS	LIGHTNING	1	2.0	0.0
5368	GREG MILLER	GOT ROCKS	1	0.0	0.0
5369	GREGG ORIGER	PA-18 REPLICA	1	1.0	0.0
13506	unknown	kit	1	0.0	0.0

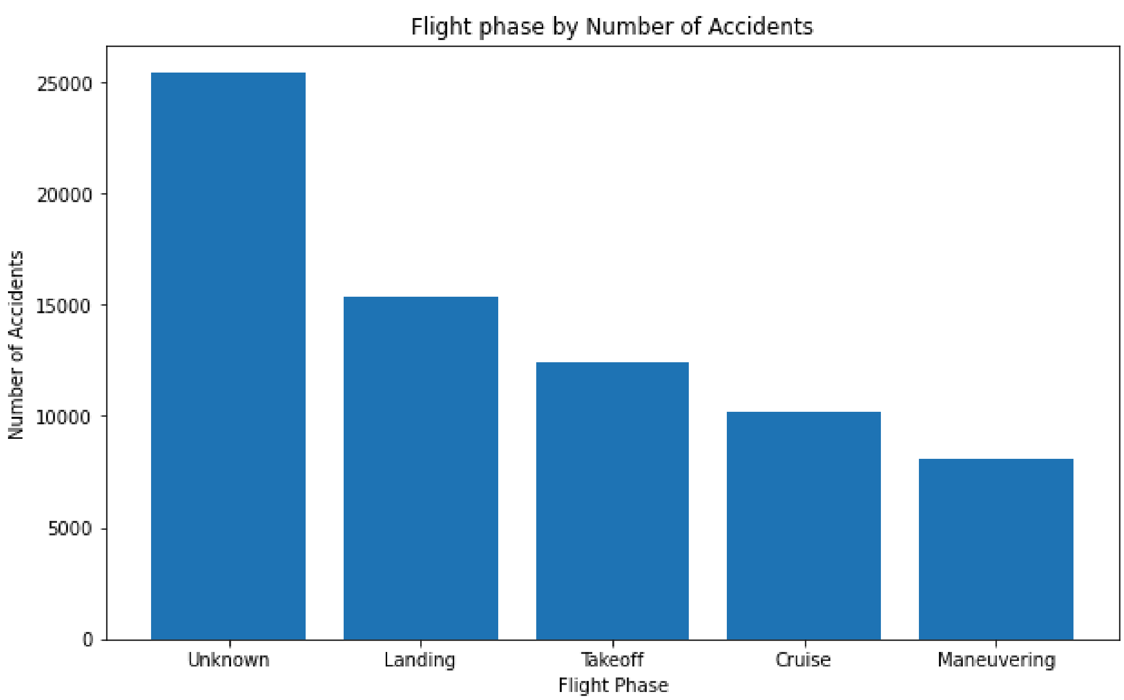
the private with low accident rates and low fatality and highest number of uninjured is the best for private use and the airplane model is vickers VC10 and it has caused no injuries and have 17 numbers of non-injured.

At which broad phase of fight did accident occur the most?

In [56]:
```
#checking the number of occurences in the phase of flight
phase_count= aviationData1['Broad.phase.of.flight'].value_counts().head()
phase_count
```

Out[56]:
```
Unknown        25396
Landing        15376
Takeoff        12425
Cruise         10212
Maneuvering     8114
Name: Broad.phase.of.flight, dtype: int64
```

In [57]:
```
# Plotting number of accident trends by phase of flight
fig,ax=plt.subplots(figsize=(10, 6))
ax.bar(phase_count.index,phase_count.values)
ax.set_title('Flight phase by Number of Accidents')
ax.set_xlabel('Flight Phase')
ax.set_ylabel('Number of Accidents')
plt.show()
```



From the above it shows most accidents happens when landing followed by Takeoff and cruise

Which aircraft are we going to purchase for both private and commercial purposes with lowest risk of accidents?

In [58]:
```
#checking the unique value in the purpose of flight
aviationData1['Purpose.of.flight'].unique()
```

Out[58]:
```
array(['Personal', 'Unknown', 'Business', 'Instructional', 'Ferry',
       'Executive/corporate', 'Aerial Observation', 'Aerial Application',
       'Public Aircraft', 'Skydiving', 'Other Work Use', 'Positioning',
       'Flight Test', 'Air Race/show', 'Air Drop',
       'Public Aircraft - Federal', 'Glider Tow',
       'Public Aircraft - Local', 'External Load',
       'Public Aircraft - State', 'Banner Tow', 'Firefighting',
       'Air Race show', 'PUBS', 'ASHO', 'PUBL'], dtype=object)
```

In [59]:
```
#categorize private and commercial aircrafts
private_flight_category = ['Executive/corporate','Personal','Business']
commercial_flight_category= ['Instructional','Public Aircraft','Positioning
                             'Air Race/show','Other Work Use']
#filter the dataset
privateFlights = aviationData1[aviationData1['Purpose.of.flight'].isin(priv
commercialFlights =aviationData1[aviationData1['Purpose.of.flight'].isin(co

#calculate frequency of the accident

# Group private and commercial models
private = privateFlights.groupby('Make').agg(accidents=('Event.Id','count')
commercial = commercialFlights.groupby('Make').agg(accidents=('Event.Id','c
#sort by accidents and total fatalities
private_aircraft_sorted = private.sort_values(by=['accidents','Total_Fatal_
commercial_aircraft_sorted = commercial.sort_values(by=['accidents','Total_
```

In [60]:
```
# Let's first identify the top 10 aircraft models for private and commercic
top_private_models = private_aircraft_sorted.head()
top_private_models
```

Out[60]:

	Make	accidents	Total_Fatal_injuries
1	1200	1	0.0
2	177MF LLC	1	0.0
3	1977 Colfer-chan	1	0.0
5	2003 Nash	1	0.0
6	2007 Savage Air LLC	1	0.0

What is the trend of accidents over the years?

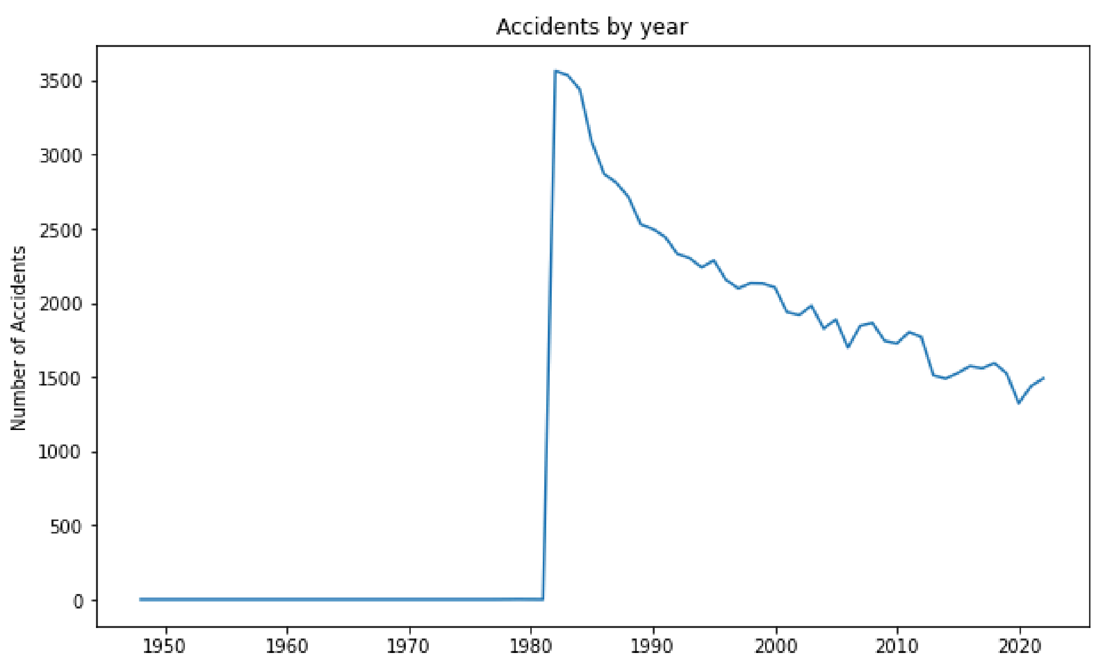
In [61]:
```
# Convrt date into year
aviationData1['Event.Date'] = pd.to_datetime(aviationData1['Event.Date'])
aviationData1['Year'] = aviationData1['Event.Date'].dt.year
# Display the first few rows
aviationData1[['Event.Date', 'Year']].head()
```

Out[61]:

	Event.Date	Year
0	1948-10-24	1948
1	1962-07-19	1962
2	1974-08-30	1974
3	1977-06-19	1977
4	1979-08-02	1979

In [62]:
```
# Group number of accidents by year and counts
accidents_by_Year = aviationData1.groupby('Year').agg(accidents=('Event.Id'
```

In [63]:
```
# Plotting number of accident trends by the Years
fig,ax =plt.subplots(figsize=(10,6))
ax.plot(accidents_by_Year['Year'],accidents_by_Year['accidents'])
ax.set_title('Accidents by year')
ax.set_xlabel('Year')
ax.set_ylabel('Number of Accidents')
plt.show()
```



The above line graph shows that from the year 1962 t0 1980 the numbae of accident has been constant(0)and from the year 1980 t0 2023 the number of accidents have been dropping as the year goes

Findings

1. private flights have slightly higher accident rates than the commercial flights but also have low injury severity rate.
2. Commercial flights experience few fatal injuries but depending on the model.
3. Based on the analysis,the aircraft model which posses low injury severity and accident rates are advised.
4. From my analysis,the weather condition did not affect much the performance of the aircraft .
5. Most accidents happened during the VMC conditions which was a conditional suitable for flying so the weather did not influence the performance of the aircraft.
6. Apart from the data that is unknown,most accidents were caused when the airplane model was Landing.
7. I can advise aircraft model like Naval Aircraft Factory N3N for private purposes as it has low injury severity and accidents
8. For commercial purpose,I can advise Public Aircraft vickers VC10 as it has low accident and fatality rate with a number of uninjured.

Metrics of success

My project would be successful if I would be able to identify:

1. Aircrafts models with high percentage of minor injuries or no injuries and low percentage of fatal injuries are preffered.
2. Aircraft models with lower accident rates are preferred and also identify aircrafts with high accident rates and serious fatal injuries so as to avoid them.
3. Aircraft models with low cost effectiveness but has low accidents and injury rates for investment.

Recommendations

1. Identify aircraft models with high risk of accident with fatal and serious injuries as they would posses a high risk and avoid them.
2. Prioritize aircraft model with low risk of Fatal and Injury rates.
3. Regular update risk assessment and re-evaluate aircraft models as new data is updated.