Aviation Risk and Investing Analysis

Business Understanding

This project analyzes aviation accidents from National Transportation Safety Board from 1962 to 2023 for private and commercial flights by location, airplane model and accident severity. These accidents range in severity from fatal to uninjured passengers, the goal is to assess risk by type, injury/fatality rate, make and flight type to provide recommendations for the business on the aircraft with the lowest risk and safest investment.

Data Understanding

The National Transportation Safety Board report is the most comprehensive dataset on aviation accidents with 88,889 instances recored from 1962 to 2023, ranging from domestic/internal flights, commercial vs private, location, weather conditions and injury statistics (number of fatal, serious, minor and uninjured passangers) for each incident are provided.

In [6]:
```
import pandas as pd
import matplotlib.pyplot as plt
```

In [7]:
```
aviation_data  = pd.read_csv('data/AviationData.csv',encoding='latin-1',low_
state_codes = pd.read_csv('data/USState_Codes.csv')
```

Aviation Data

The aviation_data dataset contains 88,889 recorded aviation accidents from 1962 to 2023, ranging from uninjured incidents to fatal accidents. Airplane model, event date, country, location, flight reason, and weather conditions are also present in the data set and will help assess level of risk for each category to help ascertain which venture is the safest.

In [8]:
```
aviation_data.head()
```

Out[8]:

	Event.Id	Investigation.Type	Accident.Number	Event.Date	Loca
0	20001218X45444	Accident	SEA87LA080	10/24/48	MC CREE
1	20001218X45447	Accident	LAX94LA336	7/19/62	BRIDGEP
2	20061025X01555	Accident	NYC07LA005	8/30/74	Saltvill
3	20001218X45448	Accident	LAX96LA321	6/19/77	EUREK
4	20041105X01764	Accident	CHI79FA064	8/2/79	Cantor

5 rows × 31 columns

In [9]:
```
aviation_data.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 [10]:
```
aviation_data.describe()
```

Out[10]:

	Number.of.Engines	Total.Fatal.Injuries	Total.Serious.Injuries	Total.Min
count	82805.000000	77488.000000	76379.000000	769
mean	1.146585	0.647855	0.279881	
std	0.446510	5.485960	1.544084	
min	0.000000	0.000000	0.000000	
25%	1.000000	0.000000	0.000000	
50%	1.000000	0.000000	0.000000	
75%	1.000000	0.000000	0.000000	
max	8.000000	349.000000	161.000000	3

In [11]:

```
state_codes.head()
```

Out[11]:

	US_State	Abbreviation
0	Alabama	AL
1	Alaska	AK
2	Arizona	AZ
3	Arkansas	AR
4	California	CA

In [12]:

```
state_codes.describe()
```

Out[12]:

	US_State	Abbreviation
count	62	62
unique	62	62
top	Kansas	WI
freq	1	1

In [13]:

```
aviation_data['Investigation.Type'].value_counts()
# Accident: 85015
# Incident: 3874
```

Out[13]:

```
Accident    85015
Incident     3874
Name: Investigation.Type, dtype: int64
```

In [14]:

```
aviation_data['Make'].value_counts()
```

Out[14]:
```
Cessna            22227
Piper             12029
CESSNA             4922
Beech              4330
PIPER              2841
                  ...
Kenny Deward          1
MILLS MICHAEL         1
Marvin T Eiland       1
Lowther               1
Apex Aircraft         1
Name: Make, Length: 8237, dtype: int64
```

In [15]:
```
aviation_data['Broad.phase.of.flight'].value_counts()
```

Out[15]:
```
Landing        15428
Takeoff        12493
Cruise         10269
Maneuvering     8144
Approach        6546
Climb           2034
Taxi            1958
Descent         1887
Go-around       1353
Standing         945
Unknown          548
Other            119
Name: Broad.phase.of.flight, dtype: int64
```

Data Preperation and Merging

Cleaning and creating a uniform format for each dataset, datetime fields like Event Date are standarized to YYYY-MM-DD and creating a new column called State in the Avaiation data set. This will be the main key that joins the Aviation_data set to the state_codes data set, this will help assess state location in the US for each incident occuring domestically.

82,248 of recorded incidents occured in the United States compared to 6,641 that occured outside the Unites States. Since %7.47 is not located in the U.S and unable to join state data to garner a location, we're going to primarily focus our obervations domestically.

In [59]:
```
# Look at the count of incidents by country and understand out focus area ba
incident_number_by_country = aviation_data.groupby('Country')['Event.Id'].co
incident_number_by_country.sort_values(by='Incidents_Per_Country', ascending
```

Out[59]:

	Country	Incidents_Per_Country
207	United States	82248
29	Brazil	374
35	Canada	359
127	Mexico	358
206	United Kingdom	344
...	...	...
172	Seychelles	1
173	Sierra Leone	1
40	Chad	1
186	St Lucia	1
109	Liberia	1

219 rows × 2 columns

In [16]:

```
#Cleaning the date format of Event.Date to YYYY-MM-DD
aviation_data['Event.Date'] = pd.to_datetime(aviation_data['Event.Date'])
aviation_data['Event.Date'] = aviation_data['Event.Date'].dt.strftime('%Y-%m
```

In [17]:

```
# Create a State column for in the aviation_data to join on state_codes
avaiation_US = aviation_data[aviation_data['Country']=='United States']
aviation_data['State'] = avaiation_US['Location'].str[-2:]
```

In [18]:

```
# Since 82,248 of the 88,889 records are located in the United States, we ar
# a key part of our analysis and recommendations with the next highest being

aviation_data[aviation_data['Country']=='United States']
```

Out[18]:

	Event.Id	Investigation.Type	Accident.Number	Event.Date	
0	20001218X45444	Accident	SEA87LA080	2048-10-24	
1	20001218X45447	Accident	LAX94LA336	2062-07-19	BRI
2	20061025X01555	Accident	NYC07LA005	1974-08-30	Sa
3	20001218X45448	Accident	LAX96LA321	1977-06-19	EU
4	20041105X01764	Accident	CHI79FA064	1979-08-02	C
...	...	...	...	...	
88884	2.02212E+13	Accident	ERA23LA093	2022-12-26	'
88885	2.02212E+13	Accident	ERA23LA095	2022-12-26	Har
88886	2.02212E+13	Accident	WPR23LA075	2022-12-26	F
88887	2.02212E+13	Accident	WPR23LA076	2022-12-26	M
88888	2.02212E+13	Accident	ERA23LA097	2022-12-29	A

82248 rows × 32 columns

In [19]:

```
# Make column names easier to use (caused error's when rerunning cells)
# aviation_data.columns = aviation_data.columns.str.lower().str.replace(' ',
# state_codes.columns = state_codes.columns.str.lower().str.replace(' ', '_'
print(aviation_data.columns)
```

```
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.Descriptio
n',
       'Schedule', 'Purpose.of.flight', 'Air.carrier', 'Total.Fatal.Injurie
s',
       'Total.Serious.Injuries', 'Total.Minor.Injuries', 'Total.Uninjured',
       'Weather.Condition', 'Broad.phase.of.flight', 'Report.Status',
       'Publication.Date', 'State'],
      dtype='object')
```

Drop columns in aviation_data that are mostly null or not appliable to the risk analysis

- Latitude 34382 non-null

- Longitude 34373 non-null
- Aircraft.Category 32287 non-null
- FAR.Description 32023 non-null
- 2Schedule 12582 non-null

In [20]:
```
null_columns = ['Latitude', 'Longitude','Aircraft.Category','FAR.Descriptior
aviation_data = aviation_data.drop(columns=null_columns)
```

The injury columns are the primary metrics of the analysis that will help assess risk. We'll need to handle update null values with median that will not sku the injury data

In [27]:
```
aviation_data['Total.Fatal.Injuries'].describe()
aviation_data['Total.Fatal.Injuries'].fillna(aviation_data['Total.Fatal.Inju
```

Observations on Total Fatal Injuries

There is a large outlier that sku the mean up, in most instances a fatality does not occur and the median will be used to fill null data for Total.Fatal.Injuries

In [28]:
```
aviation_data['Total.Serious.Injuries'].describe()
aviation_data['Total.Serious.Injuries'].fillna(aviation_data['Total.Serious.
```

Obervations on Total Serious Injuries

There is a large outlier that sku the mean up, in most instances a serious injuries does not occur and the median will be used to fill null data for Total.Serious.Injuries

In [29]:
```
aviation_data['Total.Minor.Injuries'].describe()
aviation_data['Total.Minor.Injuries'].fillna(aviation_data['Total.Minor.Inju
```

Observations on Total Minor Injuries

There is a large outlier that sku the mean up, in most instances a Minor injuries does not occur and the median will be used to fill null data for Total.Minor.Injuries

In [30]:
```
aviation_data['Total.Uninjured'].describe()
aviation_data['Total.Uninjured'].fillna(aviation_data['Total.Uninjured'].mec
```

Obervations on Total Uninjured

There is a large outlier that sku the mean up, in most instances a Uninjured does not occur and the median will be used to fill null data for Total.Uninjured. There is a large standard deviation, meaning there is more spread in the data. To remain consistent we're going to use the median

Merging Data

Merging avaiation_data against the state_codes to pull in state names for accidents that occured in the United States.

In [31]:
```
# aviation_data.set_index('State')
# state_codes.set_index('Abbreviation', inplace=True)
```

In [32]:
```
# Merging
aviation_accidents = pd.merge(aviation_data, state_codes, how='left', left_c
aviation_accidents.head()
```

Out[32]:

	Event.Id	Investigation.Type	Accident.Number	Event.Date	Loca
0	20001218X45444	Accident	SEA87LA080	2048-10-24	MC CREE
1	20001218X45447	Accident	LAX94LA336	2062-07-19	BRIDGEP
2	20061025X01555	Accident	NYC07LA005	1974-08-30	Saltvill
3	20001218X45448	Accident	LAX96LA321	1977-06-19	EUREK
4	20041105X01764	Accident	CHI79FA064	1979-08-02	Cantor

5 rows × 29 columns

Exploratory Data Analysis

With a merged dataset, we can now groupby Make, State, Purpose of Flight to assess the mean, median, standard deviation and totals injuries to calculate which aircraft is the best investment and lowest risk. The two metrics that we're going to assess risk by is Purpose.of.flight (commercial, private, skydiving ect) then once we determine which purpose is the lowest risk we're going to determine which model within the lowest risk purpose category is best. We will also be adding in the count to assure we're removing any data points that are statistically insignifiant, we're only considering instances where the count occured over 30 times to be statistically signifigant.

In [36]:
```
aviation_grouped_make_data = aviation_accidents.groupby(['Make'])[['Total.Fa
aviation_grouped_make_data['Count'] = aviation_accidents.groupby('Make').siz

# Group by 'Purpose of flight' with list of columns
```

```
aviation_grouped_purpose_data = aviation_accidents.groupby(['Purpose.of.flig
aviation_grouped_purpose_data['Count'] = aviation_accidents.groupby('Purpose

# Group by 'Country' and 'US_State' with list of columns
aviation_grouped_location_data = aviation_accidents.groupby(['Country', 'US_
aviation_grouped_location_data['Count'] = aviation_grouped_location_data.gro
```

In [37]:

```
#columns are not flat due to group by and aggregate functions
aviation_grouped_make_data.columns = ['_'.join(col).strip() if col[1] != ''
aviation_grouped_purpose_data.columns = ['_'.join(col).strip() if col[1] !=
aviation_grouped_location_data.columns = ['_'.join(col).strip() if col[1] !=
```

Data Filtering and Statistical Signifigance

To assure we're analyzing statistically significant data, each dataset that is grouped by make, purpose and location is filtered out if there are less than 30 occurences. "The related law of large numbers holds that the central limit theorem is valid as random samples become large enough, usually defined as an n ≥ 30. In research-related hypothesis testing, the term "statistically significant" is used to describe when an observed difference or association has met a certain threshold"

Article title: Significance, Errors, Power, and Sample Size: The Blocking and Tackling of Statistics URL: https://pubmed.ncbi.nlm.nih.gov/29346210/#:~:text=The%20related%20law%20of%
Website title : Anesthesia and analgesia Date accessed: December 15th, 2023

In [38]:

```
aviation_grouped_make_sorted = aviation_grouped_make_data.sort_values(by='To
aviation_grouped_make_filtered = aviation_grouped_make_sorted[aviation_group
# aviation_grouped_purpose_data.head
aviation_grouped_make_filtered
```

Out[38]:

	Make	Total.Fatal.Injuries_sum	Total.Fatal.Injuries_mean	Total.I
2929	GRUMMAN ACFT ENG COR-SCHWEIZER	1.0	0.017241	
6176	Raven	4.0	0.046512	
6717	STINSON	5.0	0.054945	
3176	Grumman-schweizer	7.0	0.057851	
1434	COSTRUZIONI AERONAUTICHE TECNA	2.0	0.064516	
...	...	...	...	
2722	Fokker	217.0	3.741379	
2194	Douglas	963.0	3.776471	
1066	Boeing	6532.0	4.097867	
95	AIRBUS	1212.0	4.828685	
361	Airbus Industrie	1024.0	7.211268	

154 rows × 22 columns

In [39]:

```
filtered_make = aviation_data[aviation_data['Make']=='GRUMMAN ACFT ENG COR-S
```

In [46]:

```
#add the count of each occurence to the grouped by Data set

aviation_grouped_purpose_sorted = aviation_grouped_purpose_data.sort_values(
aviation_grouped_purpose_sorted
aviation_grouped_purpose_filtered = aviation_grouped_purpose_sorted[aviatior
aviation_grouped_purpose_filtered = aviation_grouped_purpose_filtered.sort_v
aviation_grouped_purpose_filtered
```

Out[46]:

	Purpose.of.flight	Total.Fatal.Injuries_sum	Total.Fatal.Injuries_mean	Total
22	Public Aircraft - Local	13.0	0.175676	
13	Glider Tow	16.0	0.301887	
6	Banner Tow	19.0	0.188119	
23	Public Aircraft - State	23.0	0.359375	
5	Air Race/show	34.0	0.576271	
11	Firefighting	37.0	0.925000	
9	External Load	39.0	0.317073	
21	Public Aircraft - Federal	41.0	0.390476	
4	Air Race show	42.0	0.424242	
12	Flight Test	130.0	0.320988	
24	Skydiving	234.0	1.285714	
10	Ferry	386.0	0.475369	
20	Public Aircraft	406.0	0.563889	
2	Aerial Observation	414.0	0.521411	
15	Other Work Use	511.0	0.404272	
1	Aerial Application	549.0	0.116511	
8	Executive/corporate	598.0	1.081374	
19	Positioning	635.0	0.385784	
14	Instructional	1913.0	0.180455	
7	Business	2313.0	0.575660	
25	Unknown	9789.0	1.439136	
18	Personal	18762.0	0.379429	

22 rows × 22 columns

In [41]:

```
aviation_grouped_location_data_sorted = aviation_grouped_location_data.sort_
aviation_grouped_location_data_sorted
```

Out[41]:

	Country	US_State	Total.Fatal.Injuries_sum	Total.Fatal.Injuries_mean
36	United States	North Dakota	86.0	0.153298
1	United States	Alaska	1297.0	0.228667
18	United States	Kansas	278.0	0.251812
15	United States	Illinois	555.0	0.269417
25	United States	Minnesota	404.0	0.277473
31	United States	New Hampshire	107.0	0.290761
27	United States	Missouri	461.0	0.294569
26	United States	Mississippi	248.0	0.305043
21	United States	Maine	155.0	0.306931
29	United States	Nebraska	223.0	0.307586
37	United States	Ohio	561.0	0.307735
22	United States	Maryland	251.0	0.307975
52	United States	Washington	808.0	0.309223
3	United States	Arkansas	470.0	0.309414
23	United States	Massachusetts	301.0	0.310630
39	United States	Oregon	569.0	0.321106
14	United States	Idaho	468.0	0.325905
32	United States	New Jersey	386.0	0.329915
16	United States	Indiana	437.0	0.331061
55	United States	Wisconsin	517.0	0.331410
50	United States	Virgin Islands	2.0	0.333333

	Country	US_State	Total.Fatal.Injuries_sum	Total.Fatal.Injuries_mean
47	United States	Texas	1979.0	0.334686
9	United States	Florida	1957.0	0.335966
7	United States	Connecticut	172.0	0.342629
44	United States	South Carolina	335.0	0.343943
28	United States	Montana	363.0	0.345714
40	United States	Pacific ocean	5.0	0.357143
33	United States	New Mexico	485.0	0.357143
56	United States	Wyoming	265.0	0.358593
2	United States	Arizona	1018.0	0.359210
6	United States	Colorado	1003.0	0.367938
49	United States	Vermont	89.0	0.369295
45	United States	South Dakota	167.0	0.374439
8	United States	Delaware	43.0	0.377193
46	United States	Tennessee	426.0	0.384477
24	United States	Michigan	796.0	0.392118
48	United States	Utah	528.0	0.395210
38	United States	Oklahoma	494.0	0.398387
43	United States	Rhode Island	64.0	0.405063
0	United States	Alabama	475.0	0.411969
17	United States	Iowa	340.0	0.415140
30	United States	Nevada	514.0	0.415858

	Country	US_State	Total.Fatal.Injuries_sum	Total.Fatal.Injuries_mean
10	United States	Georgia	843.0	0.416708
35	United States	North Carolina	699.0	0.420831
42	United States	Puerto Rico	50.0	0.438596
41	United States	Pennsylvania	794.0	0.443575
5	United States	California	3949.0	0.445862
51	United States	Virginia	586.0	0.459608
20	United States	Louisiana	559.0	0.459704
19	United States	Kentucky	305.0	0.469231
12	United States	Gulf of mexico	21.0	0.477273
54	United States	West Virginia	190.0	0.482234
13	United States	Hawaii	302.0	0.605210
34	United States	New York	1386.0	0.723760
4	United States	Atlantic ocean	15.0	0.882353
53	United States	Washington_DC	85.0	2.023810
11	United States	Guam	233.0	29.125000

57 rows × 23 columns

Summary and Analysis

Based on our analysis looking by looking at the mean fatality rate by Purpose of Flight, Aircraft Make and US State location and filtering for datapoints with over 30 occurences for statistical signfigance.

I recommend the business invest in Aerial Applications with the lowest mean fatality rate of any flight type at %13.0870 with 4712 recorded instances and the 'GRUMMAN ACFT ENG COR-SCHWEIZER' make which has the lowest fatality rate of any make with a average fatality of %1.7241 when an accident does occur.

Location was also considered, recommending that the Virgin Islands, Atlantic Ocean, Gult of Mexico, Washington D.C. and Guam are excluded as areas to fly since all 5 have a fatiliaty mean over 1.00, meaning a death is likely to occur if an accident occurs in these 5 areas.

Recomendation and Reason

- Purpose of Flight: Aerial Applications
  - Reason: lowest average fatality rate of any flight type at %13.0870 with 4712 recorded instances
- Aircraft Make: GRUMMAN ACFT ENG COR-SCHWEIZER
  - Reason: lowest average fatality rate of any make with %1.7241 rate per accident. 1 fatality out of 58 aviation incidents.
- State/Territory Location (US Only): Any location excluding Virgin Islands, Atlantic Ocean, Gult of Mexico, Washington D.C. and Guam
  - Reason: The 5 locations listed above have a mean of over 1.00 meaning death is likely to occur in an aviation accident.

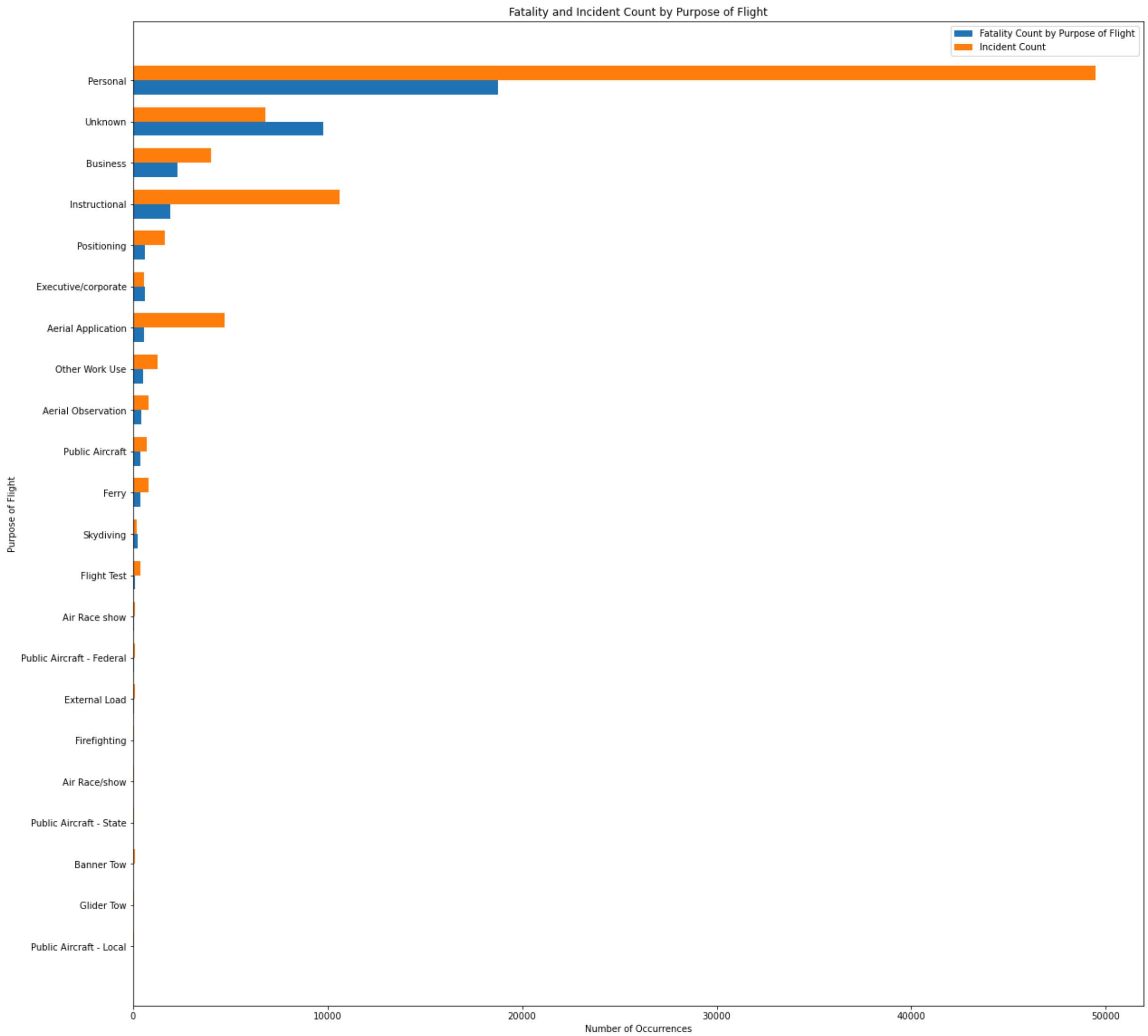
Analysis and Visuals

In [42]:
```
import matplotlib
import matplotlib.pyplot as plt
import numpy as np

%matplotlib inline
```

The bar chart below will show a visual comparison of the total accidents against total fatalities grouped by Purpose of Flight. This will indicate a high level which type of flight has the highest occurence off accidents and of those accidents, how often fatalities are involved. Personal flight are by far the most common reason for traveling and have the highest occurence of accidents and a relatively high number of fatalies relative to accidents. This bar chart confirms what the mean data verified is that Ariel Application is the safest investment relative to likelyhood a fatality will occur when an accident happens.

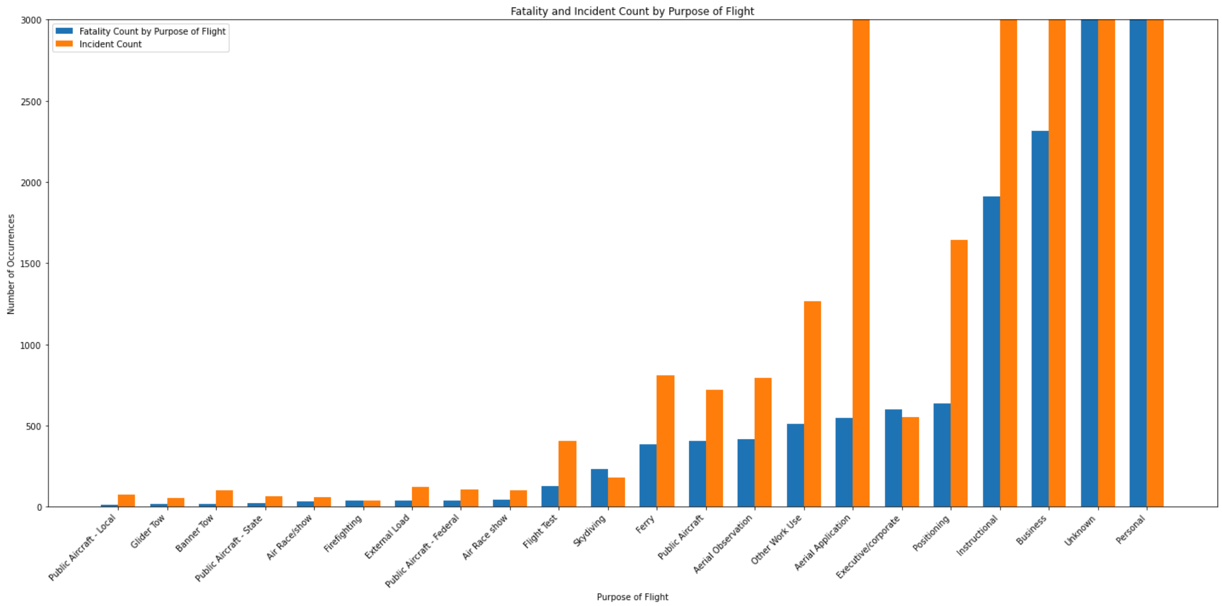
In [49]:
```
fig, ax = plt.subplots(figsize=(20, 20))
bar_height = 0.35

r1 = np.arange(len(aviation_grouped_purpose_filtered['Purpose.of.flight']))
r2 = [y + bar_height for y in r1]

plt.barh(r1, aviation_grouped_purpose_filtered['Total.Fatal.Injuries_sum'],
plt.barh(r2, aviation_grouped_purpose_filtered['Count'], height=bar_height,

plt.ylabel('Purpose of Flight')
plt.xlabel('Number of Occurrences')
```

```
plt.title('Fatality and Incident Count by Purpose of Flight')

plt.yticks([y + bar_height / 2 for y in range(len(aviation_grouped_purpose_f
plt.legend()
plt.show()
```

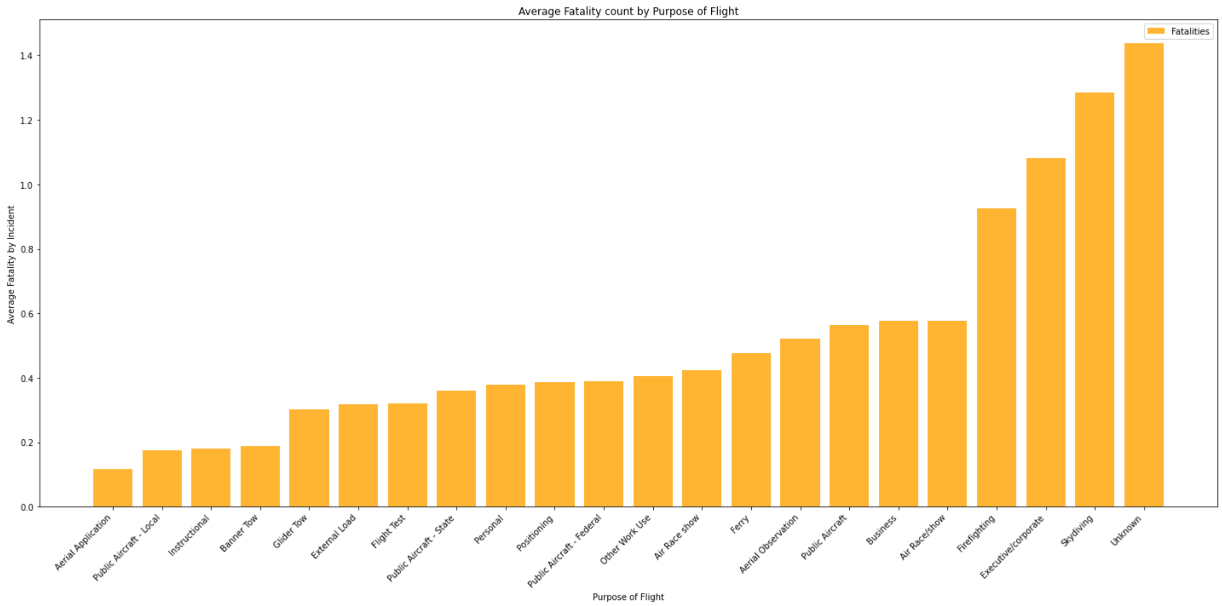


In [51]:
```
# horizontal graph that limits the Y axis to 3000 for readability and viewir

fig, ax = plt.subplots(figsize=(20, 10))
bar_width = 0.35

r1 = np.arange(len(aviation_grouped_purpose_filtered['Purpose.of.flight']))
r2 = [x + bar_width for x in r1]

plt.bar(r1, aviation_grouped_purpose_filtered['Total.Fatal.Injuries_sum'], w
plt.bar(r2, aviation_grouped_purpose_filtered['Count'], width=bar_width, lab

plt.xlabel('Purpose of Flight')
plt.ylabel('Number of Occurrences')
plt.title('Fatality and Incident Count by Purpose of Flight')

plt.ylim(0, 3000)
```

```
plt.xticks([x + bar_width / 2 for x in range(len(aviation_grouped_purpose_fi

plt.legend()
plt.tight_layout()
plt.show()
```

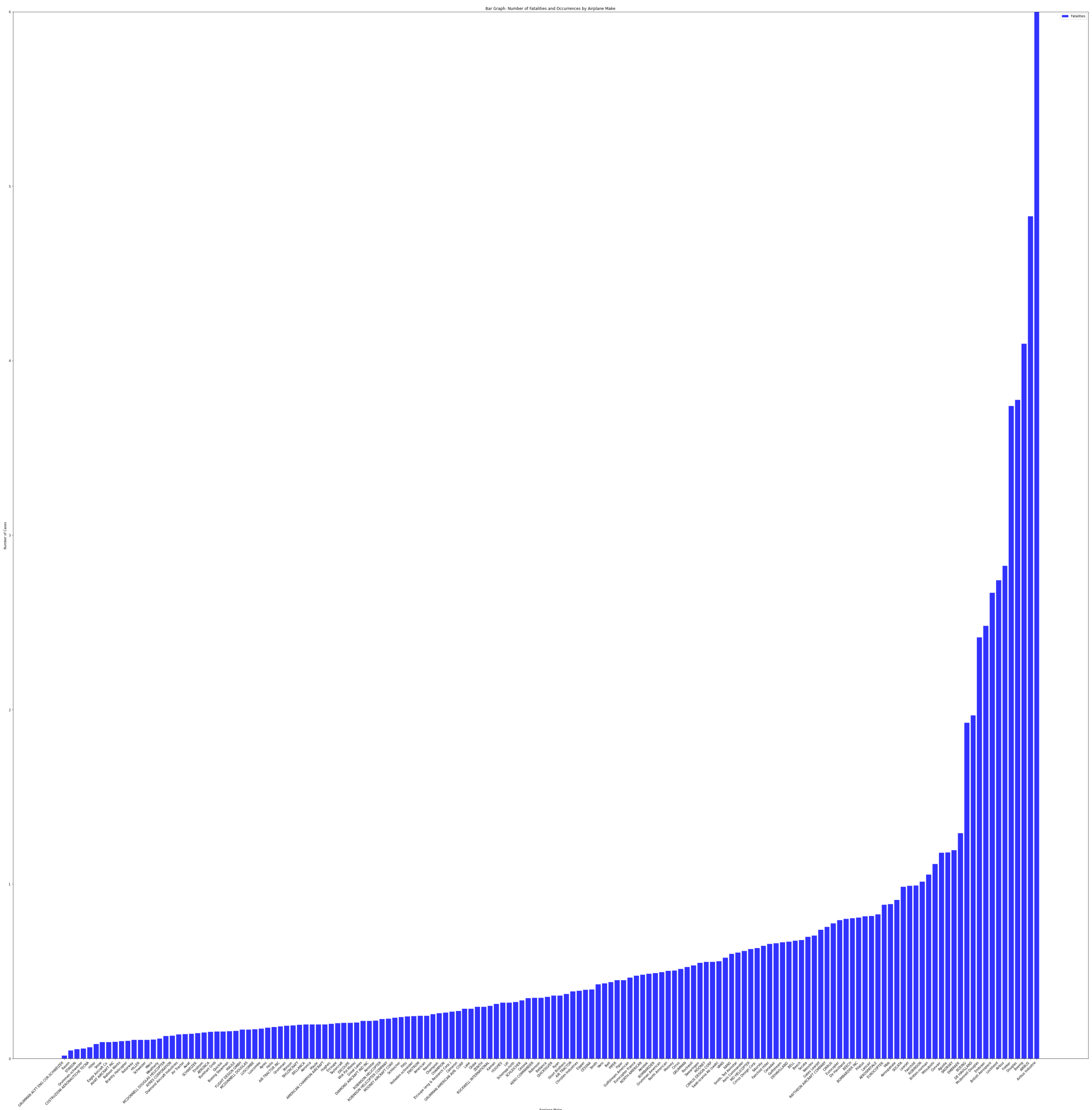


In [60]:
```
# Instead of looking at the comparison of number of fatalities to incidents,
plt.figure(figsize=(20, 10))

aviation_grouped_purpose_filtered = aviation_grouped_purpose_filtered.sort_v

plt.bar(aviation_grouped_purpose_filtered['Purpose.of.flight'], aviation_grc

plt.title('Average Fatality count by Purpose of Flight')
plt.xlabel('Purpose of Flight')
plt.ylabel('Average Fatality by Incident')
plt.xticks(rotation=45, ha='right')

plt.legend()
plt.tight_layout()
plt.show()
```

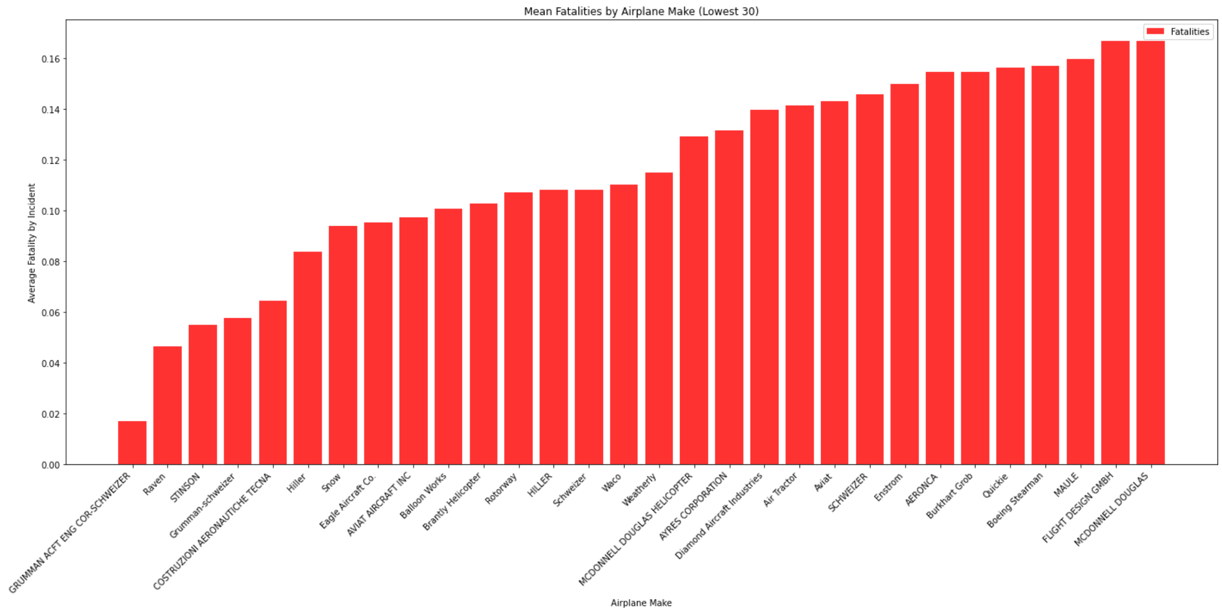


In [56]:

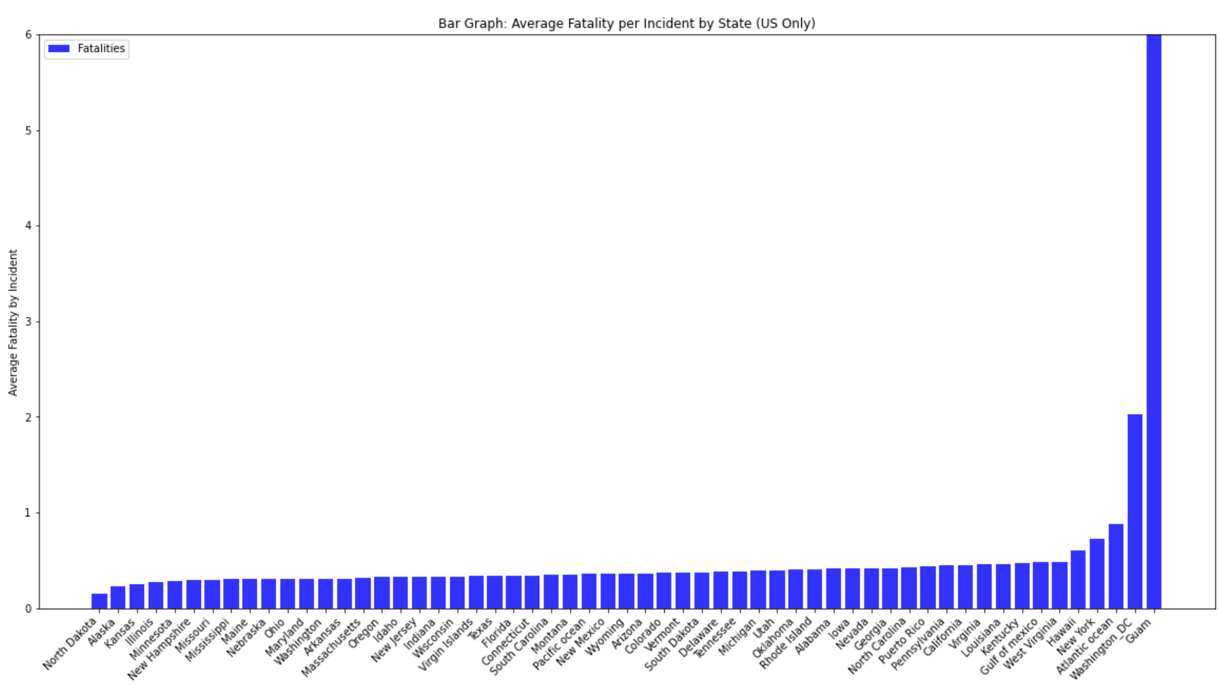
```
plt.figure(figsize=(60, 60))


plt.bar(aviation_grouped_make_filtered['Make'], aviation_grouped_make_filter
# plt.bar(aviation_grouped_make_filtered['Make'], aviation_grouped_make_filt

plt.title('Bar Graph: Number of Fatalities and Occurrences by Airplane Make'
plt.xlabel('Airplane Make')
plt.ylabel('Number of Cases')
plt.xticks(rotation=45, ha='right')

plt.ylim(0,6)

plt.legend()
plt.show()
# this is a pretty unreadable graph due to the scale on the x and y axis, we
```



In [63]:
```
# Filtered list to only show the top 30 safest aircrafts by Fatality Mean
aviation_grouped_make_filtered_30 = aviation_grouped_make_filtered.head(30)
# aviation_grouped_make_filtered

plt.figure(figsize=(20, 10))

plt.bar(aviation_grouped_make_filtered_30['Make'], aviation_grouped_make_fil

plt.title('Mean Fatalities by Airplane Make (Lowest 30)')
plt.xlabel('Airplane Make')
plt.ylabel('Average Fatality by Incident')
plt.xticks(rotation=45, ha='right')

plt.legend()
plt.tight_layout()
plt.show()
```



In [58]:
```
plt.figure(figsize=(20, 10))


plt.bar(aviation_grouped_location_data_sorted['US_State'], aviation_grouped_
# plt.bar(aviation_grouped_make_filtered['Make'], aviation_grouped_make_filt

plt.title('Bar Graph: Average Fatality per Incident by State (US Only)')
plt.xlabel('State/Location')
plt.ylabel('Average Fatality by Incident')
plt.xticks(rotation=45, ha='right')

plt.ylim(0,6)

plt.legend()
plt.show()
```



In [ ]: