In [213…

```
import  pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
%matplotlib inline
```

In [214…

```
#Read the csv
df = pd.read_csv("/Aviation_Data.csv", low_memory=False)
df
```

Out[214…

	Event.Id	Investigation.Type	Accident.Number	Event.Date	Location	Country	Latitude	Longitude	Airport
0	20001218X45444	Accident	SEA87LA080	1948-10-24	MOOSE CREEK, ID	United States	NaN	NaN	
1	20001218X45447	Accident	LAX94LA336	1962-07-19	BRIDGEPORT, CA	United States	NaN	NaN	
2	20061025X01555	Accident	NYC07LA005	1974-08-30	Saltville, VA	United States	36.922223	-81.878056	
3	20001218X45448	Accident	LAX96LA321	1977-06-19	EUREKA, CA	United States	NaN	NaN	
4	20041105X01764	Accident	CHI79FA064	1979-08-02	Canton, OH	United States	NaN	NaN	
...	...	...	...	...	...	...	...	...	
90343	20221227106491	Accident	ERA23LA093	2022-12-26	Annapolis, MD	United States	NaN	NaN	
90344	20221227106494	Accident	ERA23LA095	2022-12-26	Hampton, NH	United States	NaN	NaN	
90345	20221227106497	Accident	WPR23LA075	2022-12-26	Payson, AZ	United States	341525N	1112021W	
90346	20221227106498	Accident	WPR23LA076	2022-12-26	Morgan, UT	United States	NaN	NaN	
90347	20221230106513	Accident	ERA23LA097	2022-12-29	Athens, GA	United States	NaN	NaN	

90348 rows × 31 columns

Data exploration

In [215…

```
df.shape
```

Out[215…
```
(90348, 31)
```

In [216…
```
df.columns
```

Out[216…
```
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.Description',
       'Schedule', 'Purpose.of.flight', 'Air.carrier', 'Total.Fatal.Injuries',
       'Total.Serious.Injuries', 'Total.Minor.Injuries', 'Total.Uninjured',
       'Weather.Condition', 'Broad.phase.of.flight', 'Report.Status',
       'Publication.Date'],
      dtype='object')
```

In [217…
```
#First 5 rows
df.head(5)
```

Out[217…

	Event.Id	Investigation.Type	Accident.Number	Event.Date	Location	Country	Latitude	Longitude	Airport.Cod
0	20001218X45444	Accident	SEA87LA080	1948-10-24	MOOSE CREEK, ID	United States	NaN	NaN	Nal
1	20001218X45447	Accident	LAX94LA336	1962-07-19	BRIDGEPORT, CA	United States	NaN	NaN	Nal
2	20061025X01555	Accident	NYC07LA005	1974-08-30	Saltville, VA	United States	36.922223	-81.878056	Nal
3	20001218X45448	Accident	LAX96LA321	1977-06-19	EUREKA, CA	United States	NaN	NaN	Nal
4	20041105X01764	Accident	CHI79FA064	1979-08-02	Canton, OH	United States	NaN	NaN	Nal

5 rows × 31 columns

In [218…
```
#Type of objects & missing values visibility
df.info()
```

```
<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
```

Find missing values in a DataFrame

Data Cleaning & Missing Value Imputation

In [219…

```
# Your code here
# check missing values
missing = df.isnull().sum().sort_values(ascending=False)
print("Missing values:\n", missing)
```

```
Missing values:
 Schedule                  77766
Air.carrier               73700
FAR.Description           58325
Aircraft.Category         58061
Longitude                 55975
Latitude                  55966
Airport.Code              40216
Airport.Name              37644
Broad.phase.of.flight     28624
Publication.Date          16689
Total.Serious.Injuries    13969
Total.Minor.Injuries      13392
Total.Fatal.Injuries      12860
Engine.Type                8555
Report.Status              7843
Purpose.of.flight          7651
Number.of.Engines          7543
Total.Uninjured            7371
Weather.Condition          5951
Aircraft.damage            4653
Registration.Number        2841
Injury.Severity            2459
Country                    1685
Amateur.Built              1561
Model                      1551
Make                       1522
Location                   1511
Event.Date                 1459
Event.Id                   1459
Accident.Number            1459
Investigation.Type            0
dtype: int64
```

In [220…

```
# Drop irrelevant or consistently missing columns (example)
df.drop(columns=['investigation_type', 'publication_date'], inplace=True, errors='ignore')
```

To Standardize or Normalize all column names

In [221…

```
df.info()
```

```
<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
```

In [222…

```
#Clean Column Names
# Standardize column names
df.columns = df.columns.str.strip().str.lower().str.replace('.', '_').str.replace(' ', '_')
```

```
# Check again
print(df.columns.tolist())
```

```
['event_id', 'investigation_type', 'accident_number', 'event_date', 'location', 'country', 'latitude', 'longitude',
'airport_code', 'airport_name', 'injury_severity', 'aircraft_damage', 'aircraft_category', 'registration_number', 'ma
ke', 'model', 'amateur_built', 'number_of_engines', 'engine_type', 'far_description', 'schedule', 'purpose_of_fligh
t', 'air_carrier', 'total_fatal_injuries', 'total_serious_injuries', 'total_minor_injuries', 'total_uninjured', 'weat
her_condition', 'broad_phase_of_flight', 'report_status', 'publication_date']
```

In [223…

```
df['event_date']
```

Out[223…

	event_date
0	1948-10-24
1	1962-07-19
2	1974-08-30
3	1977-06-19
4	1979-08-02
...	...
90343	2022-12-26
90344	2022-12-26
90345	2022-12-26
90346	2022-12-26
90347	2022-12-29

90348 rows × 1 columns

dtype: object

In [224…

```
# Convert dates
df['event_date'] = pd.to_datetime(df['event_date'], errors='coerce')
df = df[df['event_date'].notnull()]  # remove rows with invalid dates
```

In [225…

```
df.head(3)
```

Out[225…

	event_id	investigation_type	accident_number	event_date	location	country	latitude	longitude	airport_code
0	20001218X45444	Accident	SEA87LA080	1948-10-24	MOOSE CREEK, ID	United States	NaN	NaN	NaN
1	20001218X45447	Accident	LAX94LA336	1962-07-19	BRIDGEPORT, CA	United States	NaN	NaN	NaN
2	20061025X01555	Accident	NYC07LA005	1974-08-30	Saltville, VA	United States	36.922223	-81.878056	NaN

3 rows × 31 columns

In [226…

```
df['aircraft_damage']
```

Out[226…

	aircraft_damage
0	Destroyed
1	Destroyed
2	Destroyed
3	Destroyed
4	Destroyed
...	...
90343	NaN
90344	NaN
90345	Substantial
90346	NaN
90347	NaN

88889 rows × 1 columns

dtype: object

In [227…

```
df['engine_type']
```

Out[227…

	engine_type
0	Reciprocating
1	Reciprocating
2	Reciprocating
3	Reciprocating
4	NaN
...	...
90343	NaN
90344	NaN
90345	NaN
90346	NaN
90347	NaN

88889 rows × 1 columns

dtype: object

In [228…

```
df['make']
```

Out[228…

	make
0	Stinson
1	Piper
2	Cessna
3	Rockwell
4	Cessna
...	...
90343	PIPER
90344	BELLANCA
90345	AMERICAN CHAMPION AIRCRAFT
90346	CESSNA
90347	PIPER

88889 rows × 1 columns

dtype: object

In [229…

```
df.loc[:, 'aircraft_damage'] = df['aircraft_damage'].fillna('Unknown')
df.loc[:, 'engine_type'] = df['engine_type'].fillna('Unknown')
df.loc[:, 'make'] = df['make'].fillna('Unknown')
```

Check the actual column names

In [230…

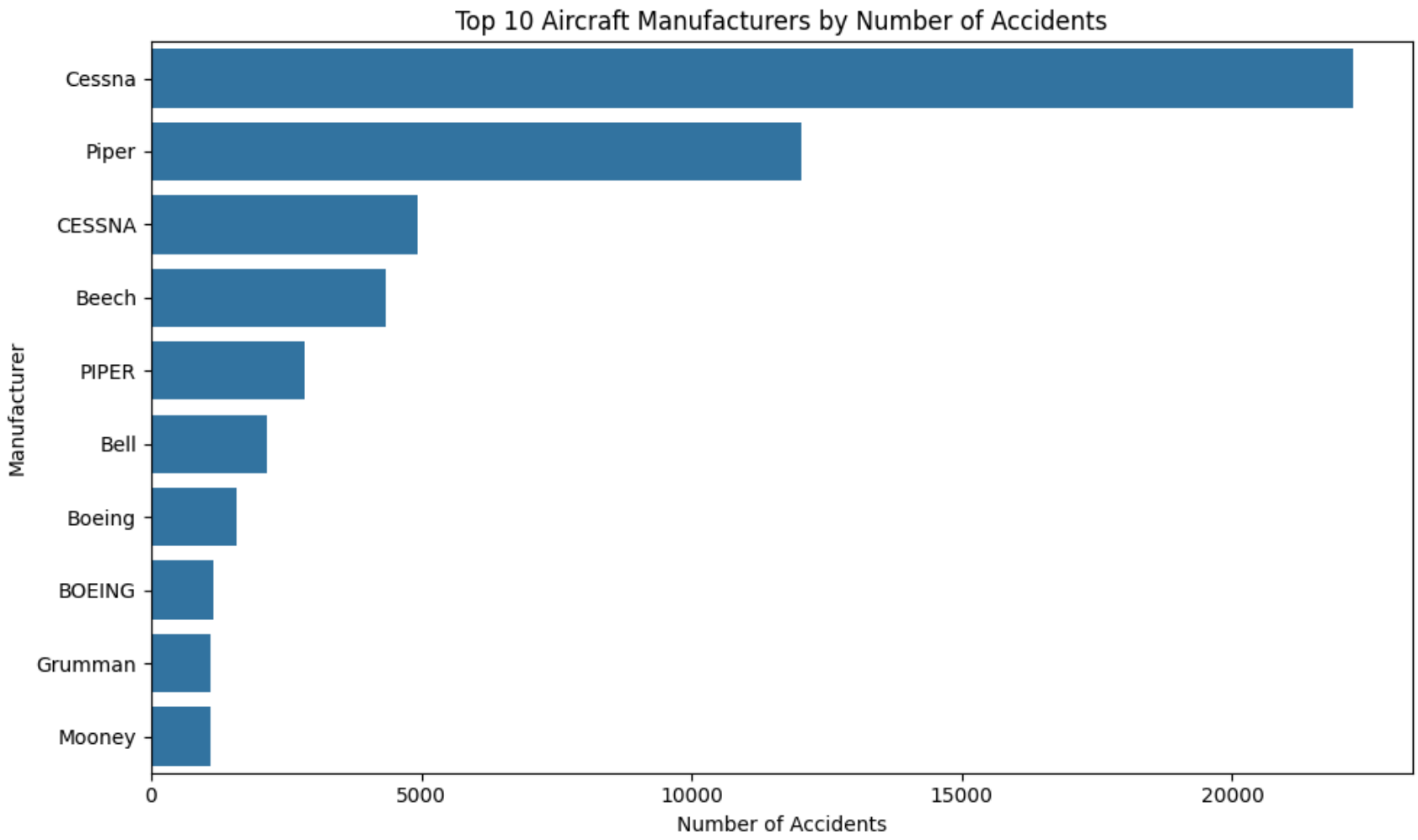
```
print(df.columns.tolist())
```

```
['event_id', 'investigation_type', 'accident_number', 'event_date', 'location', 'country', 'latitude', 'longitude',
'airport_code', 'airport_name', 'injury_severity', 'aircraft_damage', 'aircraft_category', 'registration_number', 'ma
ke', 'model', 'amateur_built', 'number_of_engines', 'engine_type', 'far_description', 'schedule', 'purpose_of_fligh
t', 'air_carrier', 'total_fatal_injuries', 'total_serious_injuries', 'total_minor_injuries', 'total_uninjured', 'weat
her_condition', 'broad_phase_of_flight', 'report_status', 'publication_date']
```

Risk Aggregation - Define risk based on fatalities, injuries, damage severity etc

In [231…

```
# Aircraft accident count by manufacturer
top_makes = df['make'].value_counts().head(10)

import matplotlib.pyplot as plt
import seaborn as sns

plt.figure(figsize=(10,6))
sns.barplot(x=top_makes.values, y=top_makes.index)
plt.title("Top 10 Aircraft Manufacturers by Number of Accidents")
plt.xlabel("Number of Accidents")
plt.ylabel("Manufacturer")
plt.tight_layout()
plt.show()
```



to check aircraft make, model, and their corresponding risk score and injury counts.

In [232…

```
# Filter valid dates and create a safe copy
df = df[df['event_date'].notnull()].copy()
```

In [233…

```
# Fill missing injury values
df.loc[:, 'total_fatal_injuries'] = df['total_fatal_injuries'].fillna(0)
df.loc[:, 'total_serious_injuries'] = df['total_serious_injuries'].fillna(0)
df.loc[:, 'total_minor_injuries'] = df['total_minor_injuries'].fillna(0)
```

```
# Calculate risk score
df.loc[:, 'risk_score'] = (
    df['total_fatal_injuries'] * 3 +
    df['total_serious_injuries'] * 2 +
    df['total_minor_injuries'] * 1
)

# Confirm changes by printing first few rows
print(df[['make', 'model', 'risk_score', 'total_fatal_injuries', 'total_serious_injuries', 'total_minor_injuries']].h
```

```
      make     model  risk_score  total_fatal_injuries  \
0   Stinson     108-3         6.0                   2.0
1     Piper  PA24-180        12.0                   4.0
2    Cessna      172M         9.0                   3.0
3  Rockwell       112         6.0                   2.0
4    Cessna       501         7.0                   1.0

   total_serious_injuries  total_minor_injuries
0                     0.0                   0.0
1                     0.0                   0.0
2                     0.0                   0.0
3                     0.0                   0.0
4                     2.0                   0.0
```

See Top Low-Risk Manufacturers (Summary Table)

In [234…

```
# Top low-risk Manufacures(Summary Table)
# Group by aircraft manufacturer
risk_summary = df.groupby('make').agg({
    'risk_score': 'mean',
    'event_id': 'count'
}).rename(columns={'event_id': 'total_incidents'}).sort_values(by='risk_score')

# Show top 10 manufacturers with lowest average risk
print(risk_summary.head(10))
```

```
                              risk_score  total_incidents
make
MID-SOUTH CUSTOM CRAFT INC           0.0                1
DIETRICH RYAN                        0.0                1
DIETERICH ROBERT A                   0.0                1
DIEHL WILLIAM A                      0.0                1
DIAMOND AIRCRAFT INDUSTRIES          0.0                1
MICHAEL ADAMCZYK                     0.0                1
MEYERS INDUSTRIES INC                0.0                3
MEYER ROBERT                         0.0                1
MEYER GEOFFREY A                     0.0                1
METCALFE ROBERT B                    0.0                1
```

In [235…

```python
print("SHAPE OF DF:", df.shape)
print(df[['make', 'risk_score']].dropna().head(10))
```

```
SHAPE OF DF: (88889, 32)
                make  risk_score
0            Stinson         6.0
1              Piper        12.0
2             Cessna         9.0
3           Rockwell         6.0
4             Cessna         7.0
5  Mcdonnell Douglas         1.0
6             Cessna        12.0
7             Cessna         0.0
8             Cessna         0.0
9     North American         3.0
```

In [236…

```python
import plotly.express as px

# Prepare summary DataFrame for scatter
risk_plot_df = risk_summary.reset_index()

fig = px.scatter(
    risk_plot_df,
    x='total_incidents',
    y='risk_score',
    hover_name='make',
    size='total_incidents',
    color='risk_score',
    color_continuous_scale='RdYlGn_r',
```

```
    title='Aircraft Manufacturers: Incidents vs. Risk Score'
)
fig.update_layout(xaxis_title='Total Incidents', yaxis_title='Average Risk Score')
fig.show()
```

Objective to find our the airline with less accident Aircfraft by model, category, flight

which aircraft types are involved in the most accidents

In [237…

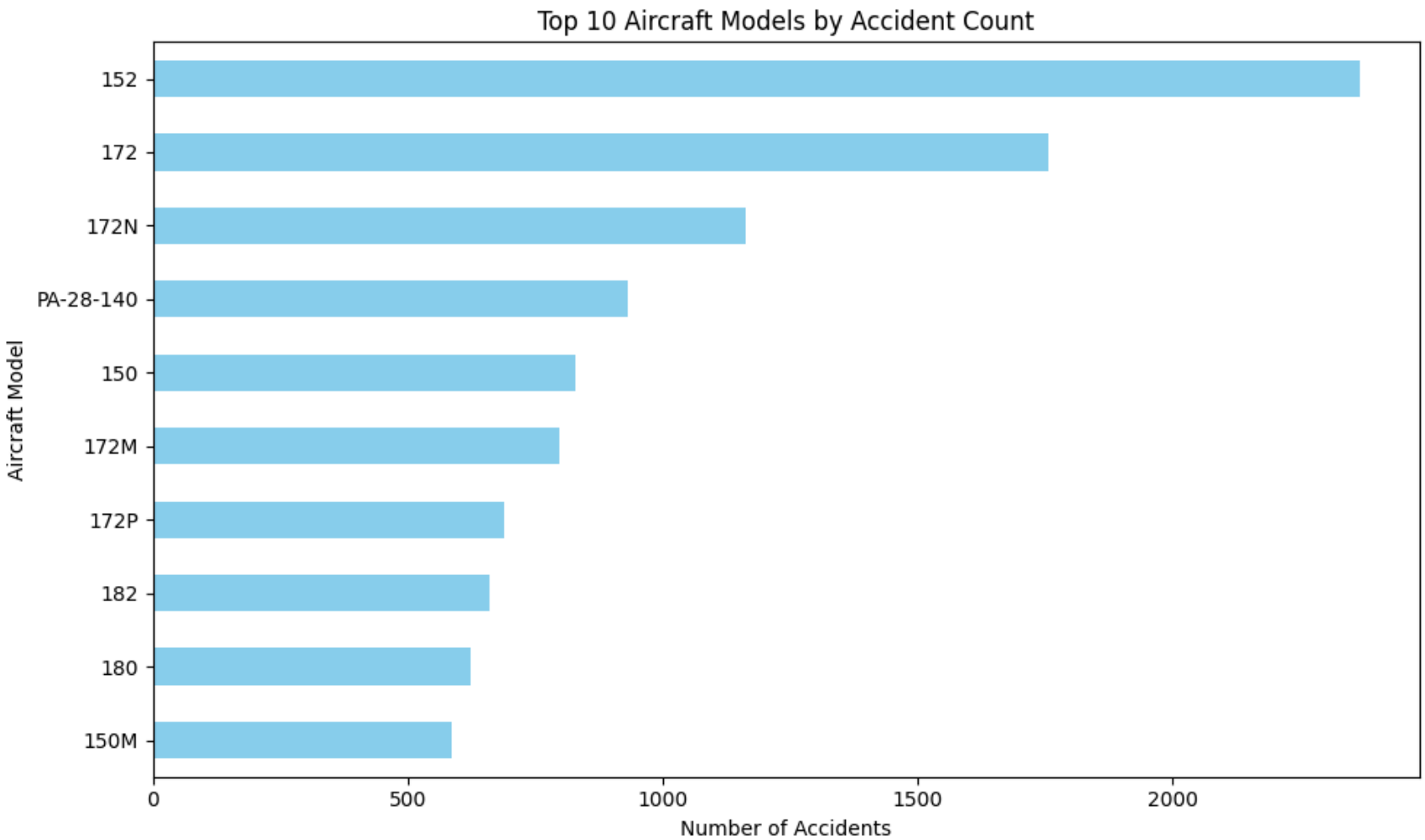
```
accidents_by_type = df['model'].value_counts()
print(accidents_by_type.head(10))
```

```
model
152          2367
172          1756
172N         1164
PA-28-140     932
150           829
172M          798
172P          689
182           659
180           622
150M          585
Name: count, dtype: int64
```

Visualization

In [238…

```
import matplotlib.pyplot as plt

top_models = accidents_by_type.head(10)

plt.figure(figsize=(10, 6))
top_models.plot(kind='barh', color='skyblue')
plt.xlabel('Number of Accidents')
plt.ylabel('Aircraft Model')
plt.title('Top 10 Aircraft Models by Accident Count')
plt.gca().invert_yaxis()
plt.tight_layout()
plt.show()
```



In [239…
```
print(df['investigation_type'].unique())
```

```
['Accident' 'Incident']
```

In [240…
```
print(df.columns.tolist())
```

```
['event_id', 'investigation_type', 'accident_number', 'event_date', 'location', 'country', 'latitude', 'longitude',
'airport_code', 'airport_name', 'injury_severity', 'aircraft_damage', 'aircraft_category', 'registration_number', 'ma
ke', 'model', 'amateur_built', 'number_of_engines', 'engine_type', 'far_description', 'schedule', 'purpose_of_fligh
t', 'air_carrier', 'total_fatal_injuries', 'total_serious_injuries', 'total_minor_injuries', 'total_uninjured', 'weat
her_condition', 'broad_phase_of_flight', 'report_status', 'publication_date', 'risk_score']
```

Filter only accidents (exclude incidents)

In [241…

```
accidents_df = df[df['investigation_type'] == 'Accident']  # ✅ correct
accidents_df
```

Out[241…

	event_id	investigation_type	accident_number	event_date	location	country	latitude	longitude	airport_
0	20001218X45444	Accident	SEA87LA080	1948-10-24	MOOSE CREEK, ID	United States	NaN	NaN	
1	20001218X45447	Accident	LAX94LA336	1962-07-19	BRIDGEPORT, CA	United States	NaN	NaN	
2	20061025X01555	Accident	NYC07LA005	1974-08-30	Saltville, VA	United States	36.922223	-81.878056	
3	20001218X45448	Accident	LAX96LA321	1977-06-19	EUREKA, CA	United States	NaN	NaN	
4	20041105X01764	Accident	CHI79FA064	1979-08-02	Canton, OH	United States	NaN	NaN	
...	...	...	...	...	...	...	...	...	
90343	20221227106491	Accident	ERA23LA093	2022-12-26	Annapolis, MD	United States	NaN	NaN	
90344	20221227106494	Accident	ERA23LA095	2022-12-26	Hampton, NH	United States	NaN	NaN	
90345	20221227106497	Accident	WPR23LA075	2022-12-26	Payson, AZ	United States	341525N	1112021W	
90346	20221227106498	Accident	WPR23LA076	2022-12-26	Morgan, UT	United States	NaN	NaN	
90347	20221230106513	Accident	ERA23LA097	2022-12-29	Athens, GA	United States	NaN	NaN	

85015 rows × 32 columns

In [242…

```
print(accidents_df['investigation_type'].value_counts())
```

```
investigation_type
Accident    85015
Name: count, dtype: int64
```

In [243…

```
# Step 5: Count number of accident records
num_accidents = accidents_df.shape[0]

print(f"Total number of accidents: {num_accidents}")
```

```
Total number of accidents: 85015
```

In [244…

```
# Correct column names
accidents_by_make_model = accidents_df.groupby(['make', 'model']).size().reset_index(name='accident_count')
accidents_by_make_model
```

Out[244…

	make	model	accident_count
0	107.5 Flying Corporation	One Design DR 107	1
1	1200	G103	1
2	177MF LLC	PITTS MODEL 12	1
3	1977 Colfer-chan	STEEN SKYBOLT	1
4	1st Ftr Gp	FOCKE-WULF 190	1
...	...	...	...
19353	de Havilland	DHC-3	1
19354	de Havilland	DHC-6-200	1
19355	de Havilland	DHC-8-202	1
19356	drone	Viper Pro	1
19357	unknown	kit	1

19358 rows × 3 columns

In [245…

```
# Sort by number of accidents in descending order
accidents_by_make_model_sorted = accidents_by_make_model.sort_values(by='accident_count', ascending=False)
accidents_by_make_model_sorted
```

Out[245…

	make	model	accident_count
5304	Cessna	152	2155
5326	Cessna	172	1250
5370	Cessna	172N	993
14368	Piper	PA-28-140	809
5279	Cessna	150	711
...	...	...	...
19340	Zivko Aeronautics	EDGE 540	1
5	2000 Mccoy	Genesis	1
0	107.5 Flying Corporation	One Design DR 107	1
19357	unknown	kit	1
14	A Pair Of Jacks	RV-6A	1

19358 rows × 3 columns

In [246…

```
# By manufacturer (Make)
df['make'].value_counts()
```

Out[246…

make	count
Cessna	22227
Piper	12029
CESSNA	4922
Beech	4330
PIPER	2841
...	...
SCOTT TERRY G	1
JAMES R DERNOVSEK	1
ORLICAN S R O	1
ROYSE RALPH L	1
RHINEHART	1

8237 rows × 1 columns

dtype: int64

In [247…

```
# Or by specific aircraft model
df['model'].value_counts()
```

Out[247…

model	count
152	2367
172	1756
172N	1164
PA-28-140	932
150	829
…	…
BHT-47-G3B1	1
TE-1	1
BL-7-GCBC	1
S2F-1	1
PTRAVLER	1

12318 rows × 1 columns

dtype: int64

In [248…

```
accidents_df = df[df['investigation_type'] == 'Accident']
accidents_df['make'].value_counts()
```

Out[248…

make	count
Cessna	21973
Piper	11885
CESSNA	4820
Beech	4170
PIPER	2799
...	...
Arthur Lee Beer	1
Boyd Young	1
Sandell	1
Fobes	1
Leonard Walters	1

8170 rows × 1 columns

dtype: int64

In [249…

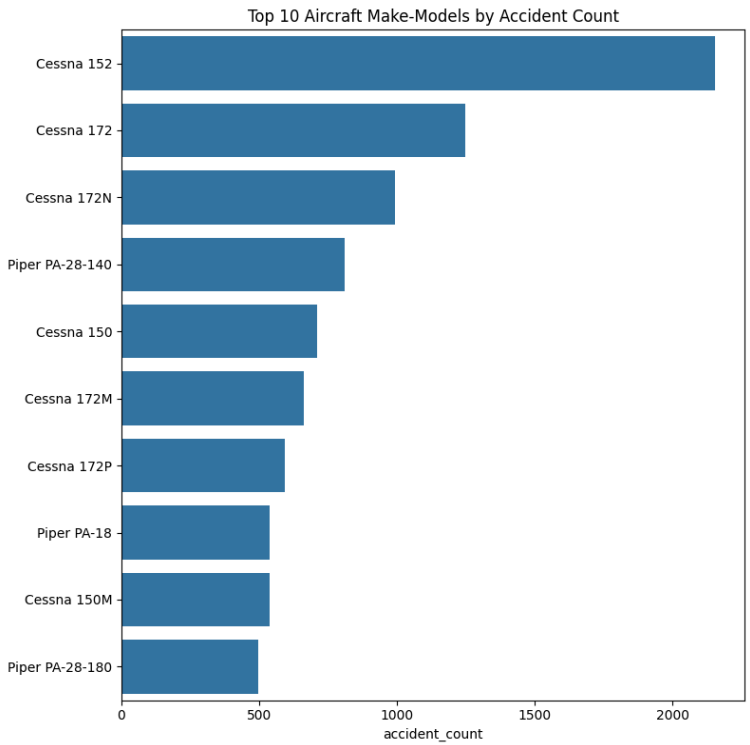
```
print(df.columns.tolist())
```

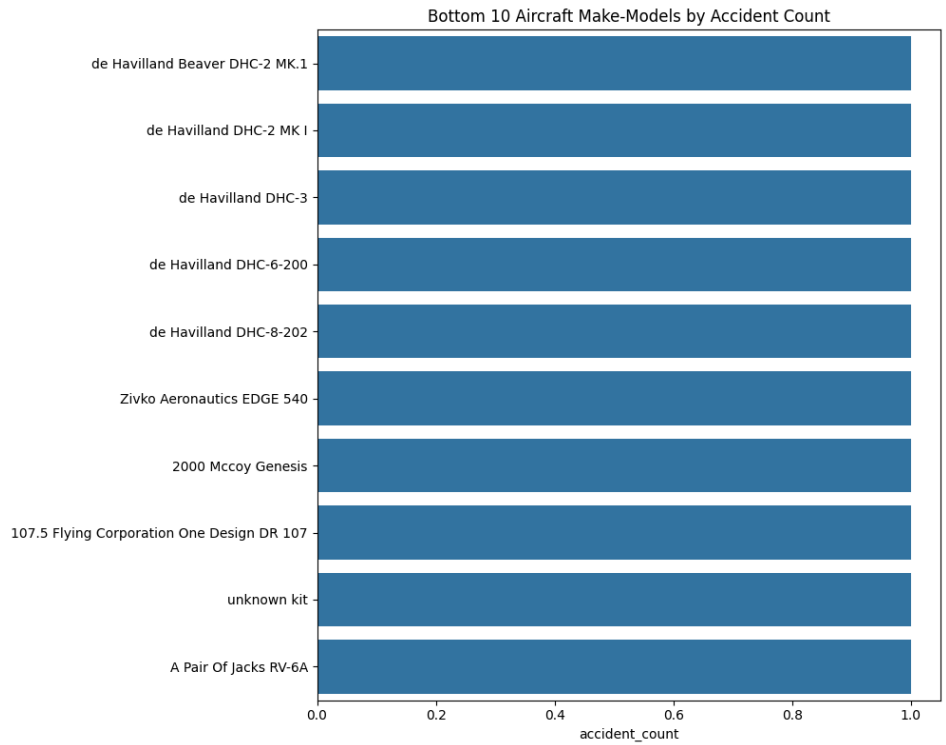
```
['event_id', 'investigation_type', 'accident_number', 'event_date', 'location', 'country', 'latitude', 'longitude',
'airport_code', 'airport_name', 'injury_severity', 'aircraft_damage', 'aircraft_category', 'registration_number', 'ma
ke', 'model', 'amateur_built', 'number_of_engines', 'engine_type', 'far_description', 'schedule', 'purpose_of_fligh
t', 'air_carrier', 'total_fatal_injuries', 'total_serious_injuries', 'total_minor_injuries', 'total_uninjured', 'weat
her_condition', 'broad_phase_of_flight', 'report_status', 'publication_date', 'risk_score']
```

In [250…

```
# Prepare data safely
top10 = accidents_by_make_model_sorted.head(10).copy()
bottom10 = accidents_by_make_model_sorted[accidents_by_make_model_sorted['accident_count'] > 0].tail(10).copy()

# Create aircraft label
```

```
top10['aircraft'] = top10['make'] + " " + top10['model']
bottom10['aircraft'] = bottom10['make'] + " " + bottom10['model']

# Set up subplots
fig, axes = plt.subplots(ncols=2, figsize=(18, 8))

# Top 10 plot
sns.barplot(data=top10, x='accident_count', y='aircraft', ax=axes[0])
axes[0].set_title("Top 10 Aircraft Make-Models by Accident Count")
axes[0].set_xlabel("accident_count")
axes[0].set_ylabel("")

# Bottom 10 plot
sns.barplot(data=bottom10, x='accident_count', y='aircraft', ax=axes[1])
axes[1].set_title("Bottom 10 Aircraft Make-Models by Accident Count")
axes[1].set_xlabel("accident_count")
axes[1].set_ylabel("")

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





In [251…

```
#common causes of accidents
print(df.columns.tolist())
```

```
['event_id', 'investigation_type', 'accident_number', 'event_date', 'location', 'country', 'latitude', 'longitude',
'airport_code', 'airport_name', 'injury_severity', 'aircraft_damage', 'aircraft_category', 'registration_number', 'ma
ke', 'model', 'amateur_built', 'number_of_engines', 'engine_type', 'far_description', 'schedule', 'purpose_of_fligh
t', 'air_carrier', 'total_fatal_injuries', 'total_serious_injuries', 'total_minor_injuries', 'total_uninjured', 'weat
her_condition', 'broad_phase_of_flight', 'report_status', 'publication_date', 'risk_score']
```

In [252…

```
# Check for common phases of flight where accidents occurred
top_causes = df['broad_phase_of_flight'].value_counts().head(10)

# Display the results
print("Top 10 Common Phases of Flight in Accidents:")
print(top_causes)
```

```
Top 10 Common Phases of Flight in Accidents:
broad_phase_of_flight
Landing        15428
Takeoff        12493
Cruise         10269
Maneuvering     8144
Approach        6546
Climb           2034
Taxi            1958
Descent         1887
Go-around       1353
Standing         945
Name: count, dtype: int64
```

Saving the data to a clean version

In [253…

```
df.to_csv("Refined_aviation_data.csv", index=False)
```