Final Project Submission

Please fill out:

- Student name: PRISCILLAH MURUGA GIRIAMA
- Student pace: PART TIME
- Scheduled project review date/time:
- Instructor name: FIDELIS WANALWENGE
- Blog post URL:

Aviation Safety Analysis:Importing relevant libraries To begin our analysis of aircraft risk factors: Importing relevant Python libraries is the first step.

In [1]:
```
# Your code here - remember to use markdown cells for comments as well!
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
```

In [2]:
```
#Loading the data
df = pd.read_csv('data/Aviation_Data.csv',low_memory=False)
```

In [3]:
```
#Exploring how the data appears
df.head()
```

Out[3]:

	Event.Id	Investigation.Type	Accident.Number	Event.Date	Location	Cou
0	20001218X45444	Accident	SEA87LA080	1948-10-24	MOOSE CREEK, ID	Ui S
1	20001218X45447	Accident	LAX94LA336	1962-07-19	BRIDGEPORT, CA	Ui S
2	20061025X01555	Accident	NYC07LA005	1974-08-30	Saltville, VA	Ui S
3	20001218X45448	Accident	LAX96LA321	1977-06-19	EUREKA, CA	Ui S
4	20041105X01764	Accident	CHI79FA064	1979-08-02	Canton, OH	Ui S

5 rows × 31 columns

In [4]:
```
# Calculate the count of missing values per column,
# sort columns in descending order of missingness,
# and display the top 20 columns with most missing data
df.isnull().sum().sort_values(ascending=False).head(20)
```

```
Out[4]: 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
        dtype: int64
```

```
In [5]: #To check all the columns in the dataset
        print(list(df.columns))
```

```
['Event.Id', 'Investigation.Type', 'Accident.Number', 'Event.Date', 'Location',
'Country', 'Latitude', 'Longitude', 'Airport.Code', 'Airport.Name', 'Injury.Sever
ity', 'Aircraft.damage', 'Aircraft.Category', 'Registration.Number', 'Make', 'Mod
el', 'Amateur.Built', 'Number.of.Engines', 'Engine.Type', 'FAR.Description', 'Sch
edule', 'Purpose.of.flight', 'Air.carrier', 'Total.Fatal.Injuries', 'Total.Seriou
s.Injuries', 'Total.Minor.Injuries', 'Total.Uninjured', 'Weather.Condition', 'Bro
ad.phase.of.flight', 'Report.Status', 'Publication.Date']
```

My reasoning was such that:Risk analysis using the dataset to decide what poses more of a risk to the business and what poses less of a risk to the business.That leads me to my next step which is selecting relevant columns form the dataset. Relevant columns include:

1. Make- Identifying the aircraft manufacturer allows comparion across other different brands
2. Model- Each model has different safety profiles
3. Injury.Severity- Direct indicator of accident seriousness
4. Aircraft.damage- Proxies cost and risk
5. Total.Fatal.Injuries- Quantifies the most severe outcome of cases in an incident
6. Total.Serious.Injuries- Adds granularity on injury impact
7. Total.Minor.Injuries- Completes the final image of the injury profile
8. Weather.Condition- External factors often correlates with accidents
9. Location- Helps identify regional or geographic risk trends
10. Number.of.Engines- Aircraft characteristic design
11. Engine.Type- Type of engine directly influences maintenance and reliability frequency
12. Broad.phase.of.flight-At what stage is an accident likely to happen
13. Report Status- Nature of the accident in general
14. Purpose.of.flight- To assess whether they should take the private or public airline approach

15. Event.Date- Assess when most accidents happen
16. Country- Are some countries prone to more accidents and why
17. Airport.Name- The maintenance of airports could be a direct factor to the accidents

In [6]:
```
#List of relevant columns to keep
relevant_columns=[  'Location', 'Event.Date','Injury.Severity', 'Aircraft.damage
df_clean= df[relevant_columns]
df_clean=df_clean.dropna()
```

In [7]:
```
#To confirm my code worked
df_clean.head()
```

Out[7]:

	Location	Event.Date	Injury.Severity	Aircraft.damage	Airport.Name	Country
7	PULLMAN, WA	1982-01-01	Non-Fatal	Substantial	BLACKBURN AG STRIP	United States
8	EAST HANOVER, NJ	1982-01-01	Non-Fatal	Substantial	HANOVER	United States
9	JACKSONVILLE, FL	1982-01-01	Non-Fatal	Substantial	JACKSONVILLE INTL	United States
11	TUSKEGEE, AL	1982-01-01	Non-Fatal	Substantial	TUSKEGEE	United States
13	HEARNE, TX	1982-01-02	Fatal(1)	Destroyed	HEARNE MUNICIPAL	United States

In [8]:
```
#Print the columns I found relevant
print(list(df_clean.columns))
```

```
['Location', 'Event.Date', 'Injury.Severity', 'Aircraft.damage', 'Airport.Name',
'Country', 'Make', 'Model', 'Number.of.Engines', 'Engine.Type', 'Total.Fatal.Inju
ries', 'Total.Serious.Injuries', 'Total.Minor.Injuries', 'Total.Uninjured', 'Weat
her.Condition', 'Broad.phase.of.flight', 'Report.Status', 'Purpose.of.flight']
```

In [9]:
```
#Checking the datatypes in each relevant column
df_clean.dtypes
```

Out[9]:
```
Location                    object
Event.Date                  object
Injury.Severity             object
Aircraft.damage             object
Airport.Name                object
Country                     object
Make                        object
Model                       object
Number.of.Engines          float64
Engine.Type                 object
Total.Fatal.Injuries       float64
Total.Serious.Injuries     float64
Total.Minor.Injuries       float64
Total.Uninjured            float64
Weather.Condition           object
Broad.phase.of.flight       object
Report.Status               object
Purpose.of.flight           object
dtype: object
```

In [10]:
```
#Converting the floats to object for uniformity in the data
# List of float64 columns to convert
float_columns = ['Number.of.Engines', 'Total.Fatal.Injuries', 'Total.Serious.Inj
                 'Total.Minor.Injuries', 'Total.Uninjured']

# Convert them to object (string) type
df_clean[float_columns] = df_clean[float_columns].astype(str)

# Verify the changes
df_clean.dtypes
```

Out[10]:
```
Location                  object
Event.Date                object
Injury.Severity           object
Aircraft.damage           object
Airport.Name              object
Country                   object
Make                      object
Model                     object
Number.of.Engines         object
Engine.Type               object
Total.Fatal.Injuries      object
Total.Serious.Injuries    object
Total.Minor.Injuries      object
Total.Uninjured           object
Weather.Condition         object
Broad.phase.of.flight     object
Report.Status             object
Purpose.of.flight         object
dtype: object
```

In [11]:
```
# Define the columns to check for missing values
columns_to_check = [ 'Location',  'Injury.Severity', 'Aircraft.damage',  'Make',

# Calculate missing values and sort in descending order
missing_df = df[columns_to_check].isnull().sum().sort_values(ascending=False)

# Filter to only show columns with missing values and calculate percentages
missing_df = missing_df[missing_df > 0]
missing_percentage = (missing_df / len(df)) * 100
```

```
# Create a summary DataFrame
missing_summary = pd.DataFrame({
    'Missing Count': missing_df,
    'Missing %': missing_percentage.round(2)
})

# results
missing_summary
```

Out[11]:

	Missing Count	Missing %
Broad.phase.of.flight	28624	31.68
Total.Serious.Injuries	13969	15.46
Total.Minor.Injuries	13392	14.82
Total.Fatal.Injuries	12860	14.23
Engine.Type	8555	9.47
Number.of.Engines	7543	8.35
Total.Uninjured	7371	8.16
Weather.Condition	5951	6.59
Aircraft.damage	4653	5.15
Injury.Severity	2459	2.72
Model	1551	1.72
Make	1522	1.68
Location	1511	1.67

In [12]:

```
#Filling the missing values in the columns
categorical_cols=['Location', 'Injury.Severity', 'Aircraft.damage',  'Make', 'Mo
df_clean[categorical_cols]=df_clean[categorical_cols].fillna('Unknown')
injury_cols=['Total.Fatal.Injuries', 'Total.Serious.Injuries',
                 'Total.Minor.Injuries', 'Total.Uninjured']
df_clean[injury_cols]=df_clean[injury_cols].fillna(0)
df_clean['Number.of.Engines']=df_clean['Number.of.Engines'].fillna(0)
df_clean['Airport.Name']=df_clean['Airport.Name'].fillna('Unknown')
df_clean['Event.Date']=df_clean['Event.Date'].fillna('Unknown')
df_clean['Location']=df_clean['Location'].fillna('Unknown')
df_clean['Country']=df_clean['Country'].fillna('Unknown')
```

In [13]:

```
#To confirm there are no missing values
df_clean.isnull().sum()
```

Out[13]:
```
Location                  0
Event.Date                0
Injury.Severity           0
Aircraft.damage           0
Airport.Name              0
Country                   0
Make                      0
Model                     0
Number.of.Engines         0
Engine.Type               0
Total.Fatal.Injuries      0
Total.Serious.Injuries    0
Total.Minor.Injuries      0
Total.Uninjured           0
Weather.Condition         0
Broad.phase.of.flight     0
Report.Status             0
Purpose.of.flight         0
dtype: int64
```

Moving on with a dataset with no missing values!!Hooray!!

DATA VISUALISATION BEGINS My approach to whether this is a safe investment to the business was to assess risks and approach it from the angle of low risks

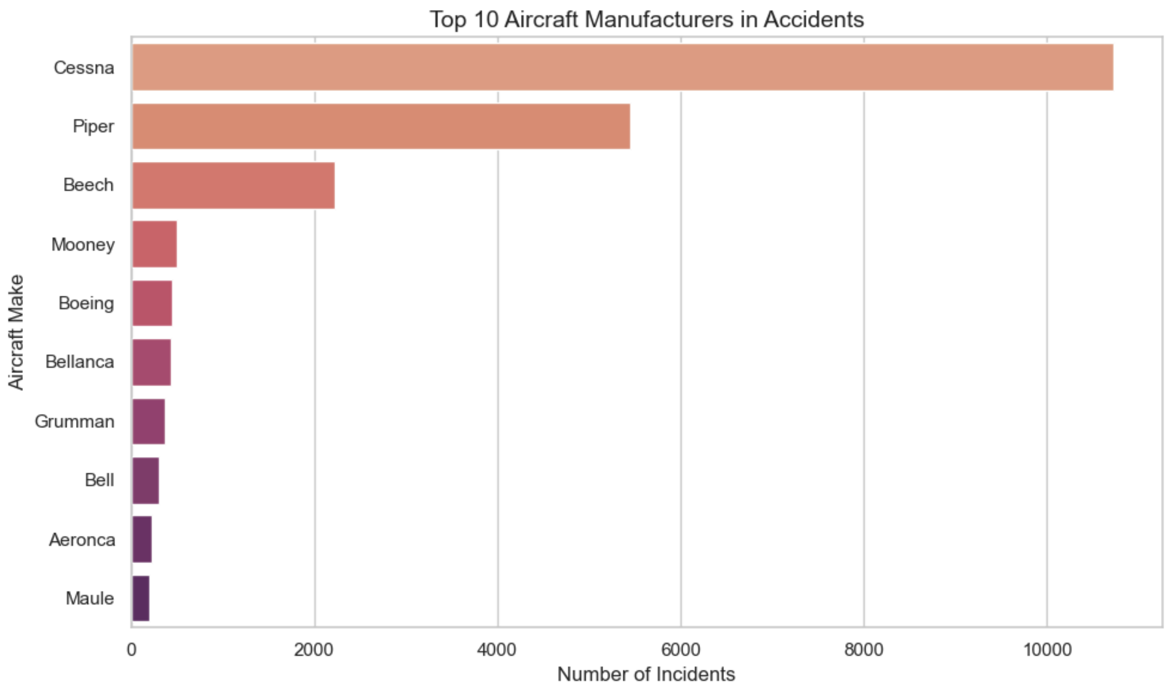
1. Comparison between the Airplane Make and Number of Incidents in a barplot
2. Comparison between the engine type versus phase of flight
3. Number of accidents according to the phase of flight in respect to the location
4. Aircraft Make versus the weather to investigate the frequency of accidents

In [14]:
```
import matplotlib.pyplot as plt
import seaborn as sns
import pandas as pd
# Set style and figure size
sns.set(style='whitegrid')
plt.figure(figsize=(10, 6))

# Get top 10 aircraft manufacturers
top_makes = df_clean['Make'].value_counts().nlargest(10)

# Create bar plot (use 'hue' to avoid FutureWarning)
sns.barplot(
    x=top_makes.values,
    y=top_makes.index,
    palette='flare',  # Corrected palette name (options: 'viridis', 'rocket', 'm
    hue=top_makes.index,  # Assign 'hue' to avoid warning
    legend=False  # Hide legend since it's redundant
)

# Add titles and labels
plt.title('Top 10 Aircraft Manufacturers in Accidents', fontsize=14)
plt.xlabel('Number of Incidents', fontsize=12)
plt.ylabel('Aircraft Make', fontsize=12)
plt.tight_layout()
plt.show()
```



1. Ranking by Incidents:
   - Cessna leads the list with the highest number of incidents (likely close to 10,000 based on the axis).
   - Piper follows as the second most accident-prone manufacturer, with significantly fewer incidents than Cessna but still notably high.
   - The remaining manufacturers (Beech,Mooney,Boeing, etc.) show progressively fewer incidents, with Maule at the bottom of the top 10.
2. Possible Explanations:
   - Higher Usage: Cessna and Piper are widely used in training and private flights, leading to more opportunities for accidents.
   - Pilot Experience: General aviation may involve less experienced pilots compared to commercial aviation.
   - Data Scope: The dataset might exclude non-fatal incidents for larger manufacturers (e.g., Boeing), skewing the representation. The plot highlights Cessna and Piper as the most accident-prone manufacturers, likely due to their prevalence in general aviation.

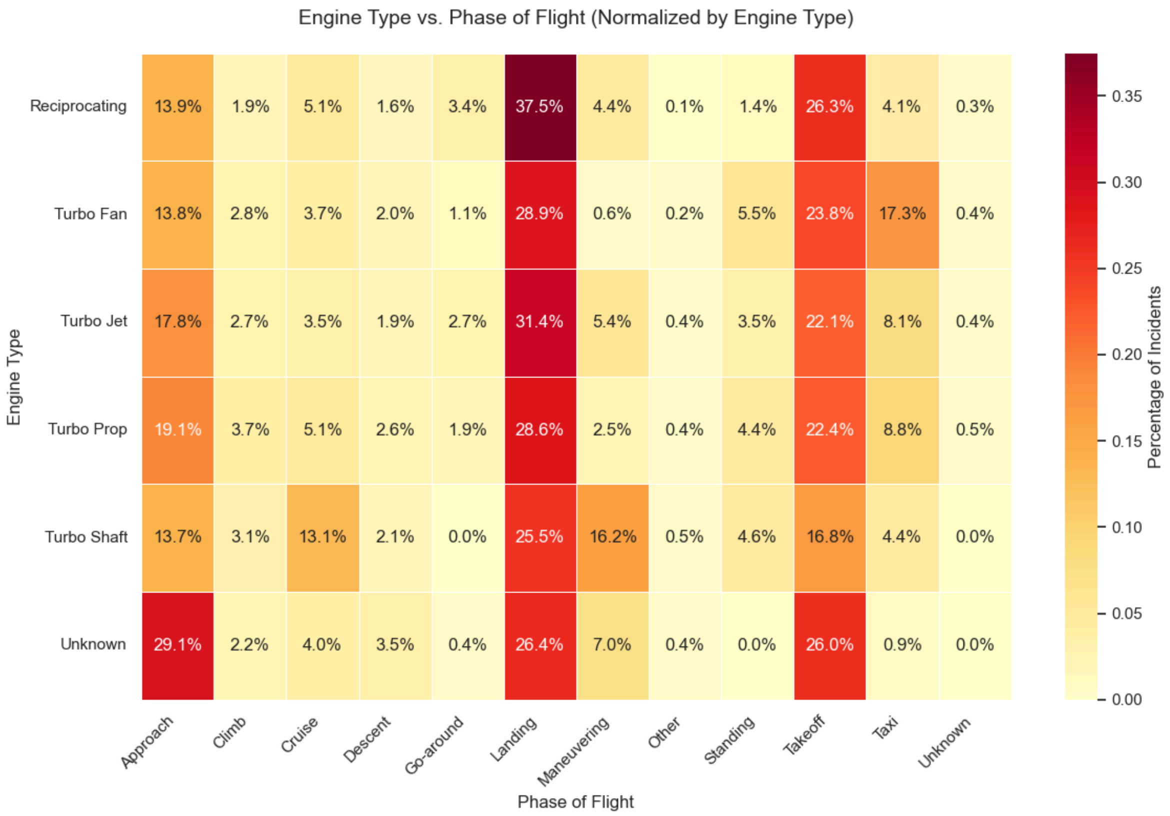
In [15]:
```
import matplotlib.pyplot as plt
import seaborn as sns
import pandas as pd
# Set style and figure size
sns.set(style="whitegrid")
plt.figure(figsize=(12, 8))

# Create cross-tabulation
engine_phase = pd.crosstab(
    df_clean['Engine.Type'],
    df_clean['Broad.phase.of.flight'],
    normalize='index'  # Show percentages by engine type (remove for raw counts)
)

# Plot heatmap
```

```python
sns.heatmap(
    engine_phase,
    annot=True,
    fmt=".1%",  # Format as percentage (use "d" for raw counts)
    cmap="YlOrRd",
    linewidths=0.5,
    cbar_kws={'label': 'Percentage of Incidents'}
)

# Customize labels
plt.title("Engine Type vs. Phase of Flight (Normalized by Engine Type)", pad=20,
plt.xlabel("Phase of Flight", fontsize=12)
plt.ylabel("Engine Type", fontsize=12)
plt.xticks(rotation=45, ha='right')
plt.yticks(rotation=0)
plt.tight_layout()
plt.show()
```



Overview

The heat map (or table) displays the normalized distribution of incidents across different phases of flight, categorized by engine type. Each row represents an engine type, and each column represents a flight phase (e.g., Approach, Climb, Cruise, etc.). The percentages indicate the proportion of incidents occurring in each phase for a given engine type. Most Critical Phases of Flight

- Landing (Highest Risk):
  - Reciprocating (37.5%), Turbo Jet (31.4%), Turbo Prop (28.6%), and Turbo Fan (28.9%) all show a high concentration of incidents during landing.
  - Turbo Shaft (25.5%) has slightly fewer landing incidents, likely because helicopters (which often use turbo-shaft engines) have different operational risks.

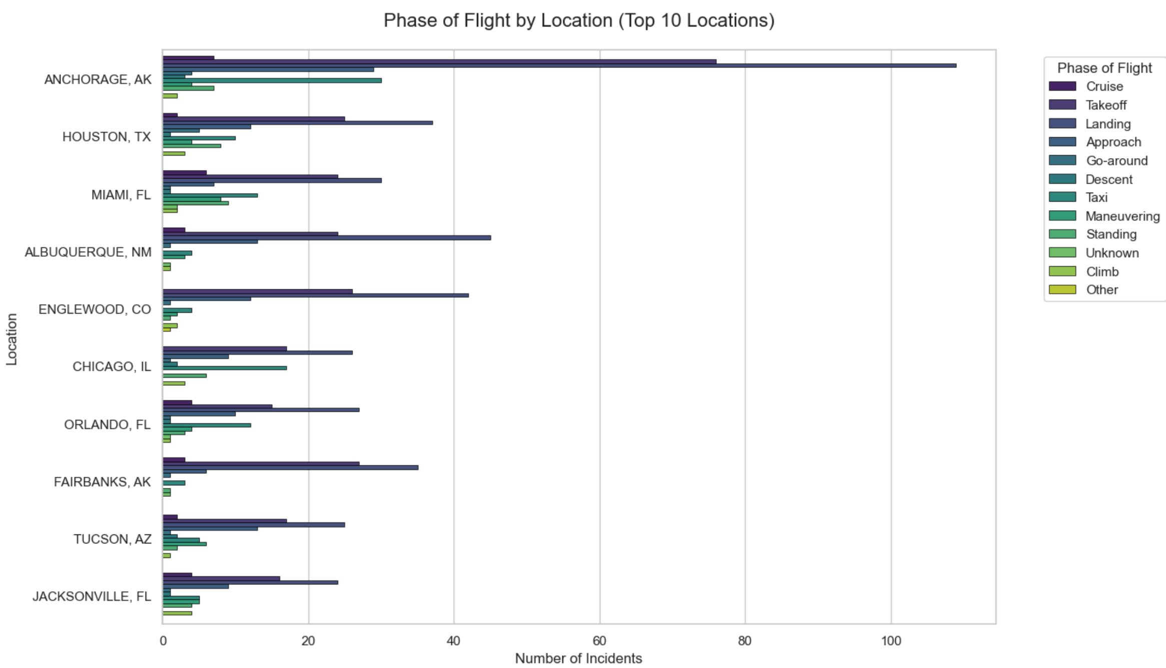
- Taxi (Second Highest Risk):
  - Reciprocating (26.3%), Turbo Fan (23.8%), Turbo Jet (22.1%), and Turbo Prop (22.4%) show significant incidents during taxiing.
  - Turbo Shaft (16.8%) is again lower, possibly due to helicopters spending less time taxiing. Turbo Prop (19.1%) and Turbo Jet (17.8%) have notably higher approach-phase incidents compared to others.

In [16]:
```
import matplotlib.pyplot as plt
import seaborn as sns
import pandas as pd
# Set style
sns.set(style="whitegrid")
plt.figure(figsize=(14, 8))

# Filter for most common locations (top 10)
top_locations = df_clean['Location'].value_counts().nlargest(10).index
df_filtered = df_clean[df_clean['Location'].isin(top_locations)]

# Create count plot
sns.countplot(
    data=df_filtered,
    y='Location',
    hue='Broad.phase.of.flight',
    palette='viridis',
    order=top_locations,  # Sort by most common locations
    edgecolor='black',
    linewidth=0.5
)

# Customize plot
plt.title('Phase of Flight by Location (Top 10 Locations)', fontsize=16, pad=20)
plt.xlabel('Number of Incidents', fontsize=12)
plt.ylabel('Location', fontsize=12)
plt.legend(title='Phase of Flight', bbox_to_anchor=(1.05, 1), loc='upper left')
plt.tight_layout()
plt.show()
```



Overview
The plot displays the number of aviation incidents (y-axis) occurring in the top 10 U.S. locations (x-axis), likely segmented by phase of flight (e.g., takeoff, landing, cruise). The data appears to be sourced from incident reports (e.g., FAA/NTSB).

Key Observations

- Locations:
  - Anchorage, AK leads with the highest number of incidents (bar extends furthest to the right, likely near 100).
  - Other high-risk locations include *Houston, TX, Miami, FL, and Albuquerque, NM.
  - Smaller bars for Englewood, CO, Chicago, IL, etc., suggest fewer incidents.
- Phase of Flight:
  - The plot implies (but does not explicitly show) that incidents are categorized by flight phase (e.g., takeoff, landing).
  - Hypothesis: Anchorage's extreme weather (e.g., icy runways, low visibility) may contribute to higher incidents during landing/takeoff. -Further Insght -In as much as some factors are in play to cause the accidents.Staff training is also a factor that shouldnt be ignored,i.e the training of pilots should be looked into and a dive into their backgrounds when employed should provide more insight on their skillset.

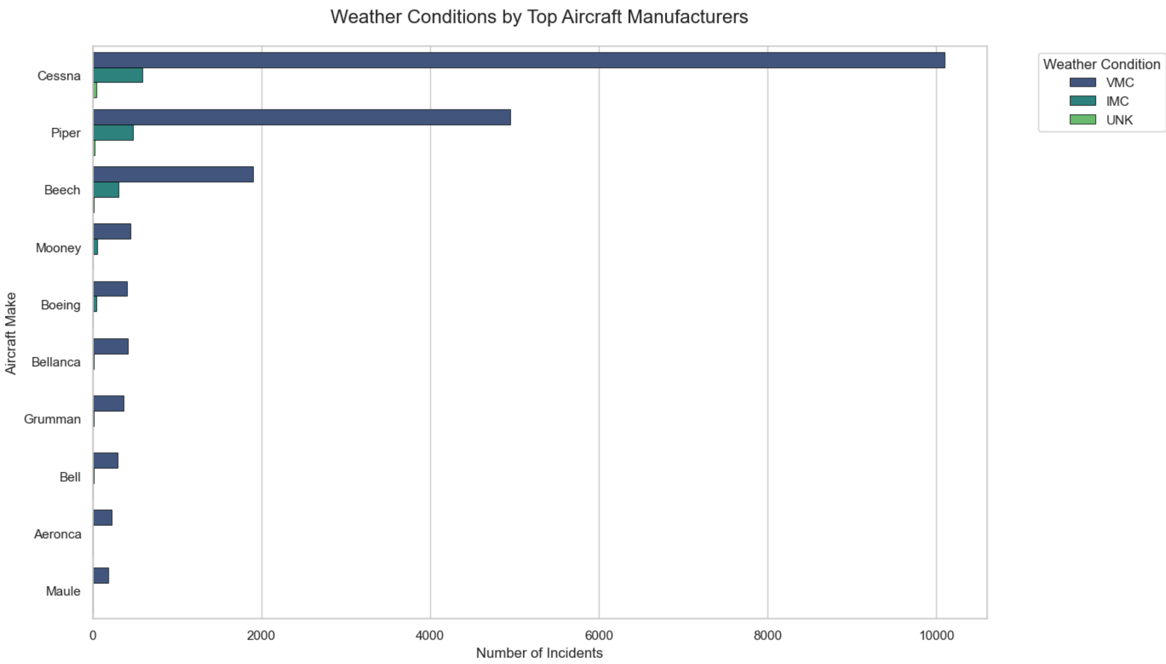
In [17]:
```
import matplotlib.pyplot as plt
import seaborn as sns
import pandas as pd
# Set style
sns.set(style="whitegrid")
plt.figure(figsize=(14, 8))

# Filter top 10 aircraft makes
top_makes = df_clean['Make'].value_counts().nlargest(10).index
df_filtered = df_clean[df_clean['Make'].isin(top_makes)]

# Create a count plot (bar plot)
sns.countplot(
    data=df_filtered,
    y='Make',
    hue='Weather.Condition',
    palette='viridis',  # Color scheme (try 'mako', 'rocket', or 'flare')
    order=top_makes,  # Sort by top makes
    edgecolor='black',
    linewidth=0.5
)

# Customize the plot
plt.title('Weather Conditions by Top Aircraft Manufacturers', fontsize=16, pad=2
plt.xlabel('Number of Incidents', fontsize=12)
plt.ylabel('Aircraft Make', fontsize=12)
plt.legend(title='Weather Condition', bbox_to_anchor=(1.05, 1), loc='upper left'
```

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



Key Observations

Manufacturer Incident Trends

- Cessna dominates with the highest incidents (~10,000), consistent with its prevalence in general aviation.
- Piper and Beech follow, reflecting their widespread use in small aircraft operations.
- Boeing (commercial jets) has fewer incidents, possibly due to stricter safety protocols or lower fleet numbers in the dataset.
  Weather Correlation
- The title suggests weather is a factor, but not a direct factor to the crashes.
- Hypothesis: Smaller aircraft (Cessna/Piper) are more vulnerable to weather-related incidents due to lighter frames and fewer advanced navigation systems. Potential Insights
- General Aviation Risks: High incidents for Cessna/Piper imply weather awareness is critical for small aircraft pilots.
- Commercial Aviation: Boeing's lower incidents may reflect better weather mitigation tech (e.g., de-icing systems).
- Outliers: Mooney and Maule (fewer incidents) might operate in less hazardous climates or have robust designs.