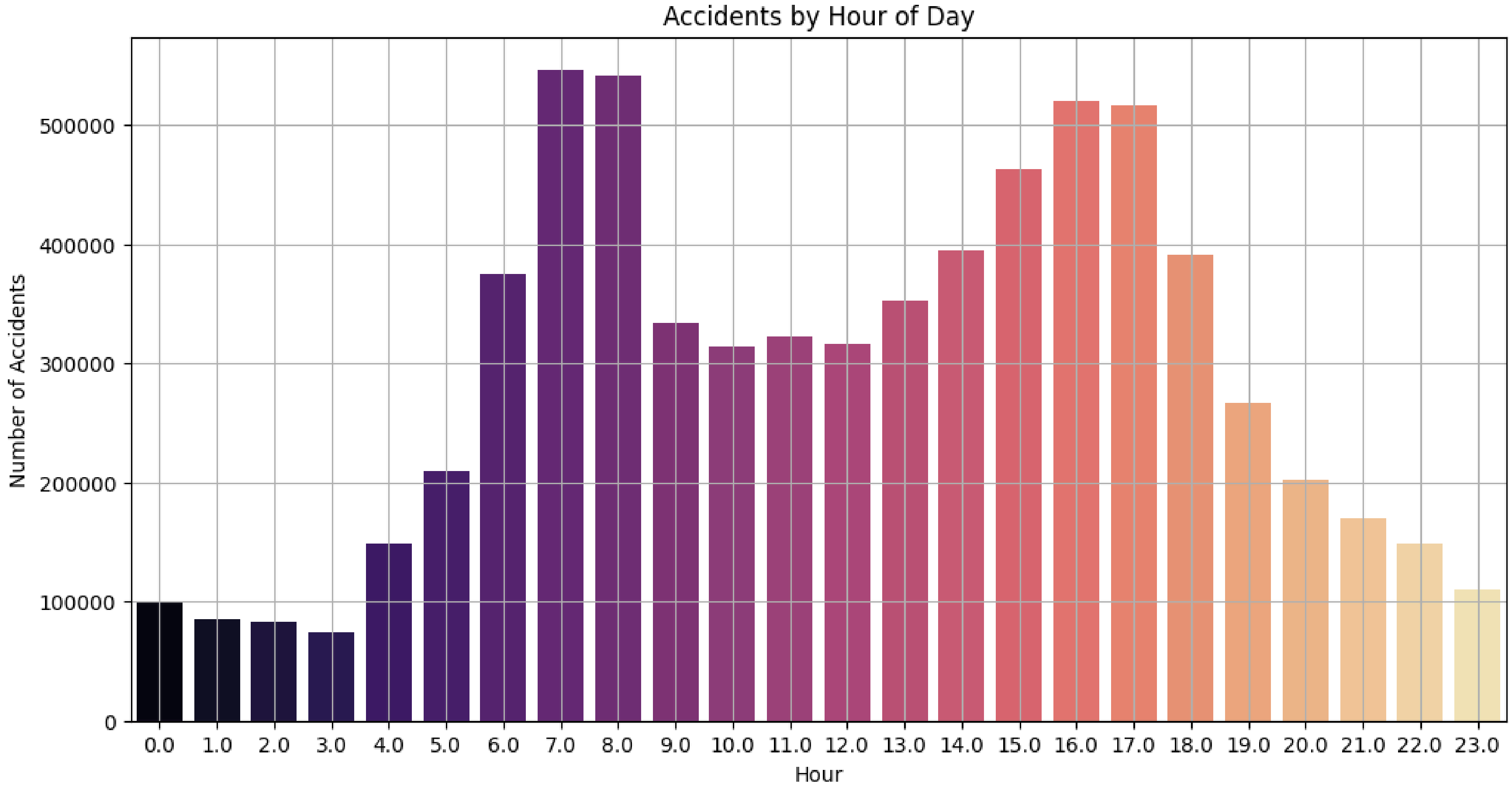
```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
import plotly.express as px
import folium
from folium.plugins import HeatMap
```

```
df1= pd.read_csv('/content/drive/MyDrive/US_Accidents_March23.csv', nrows=0).columns.tolist()
```

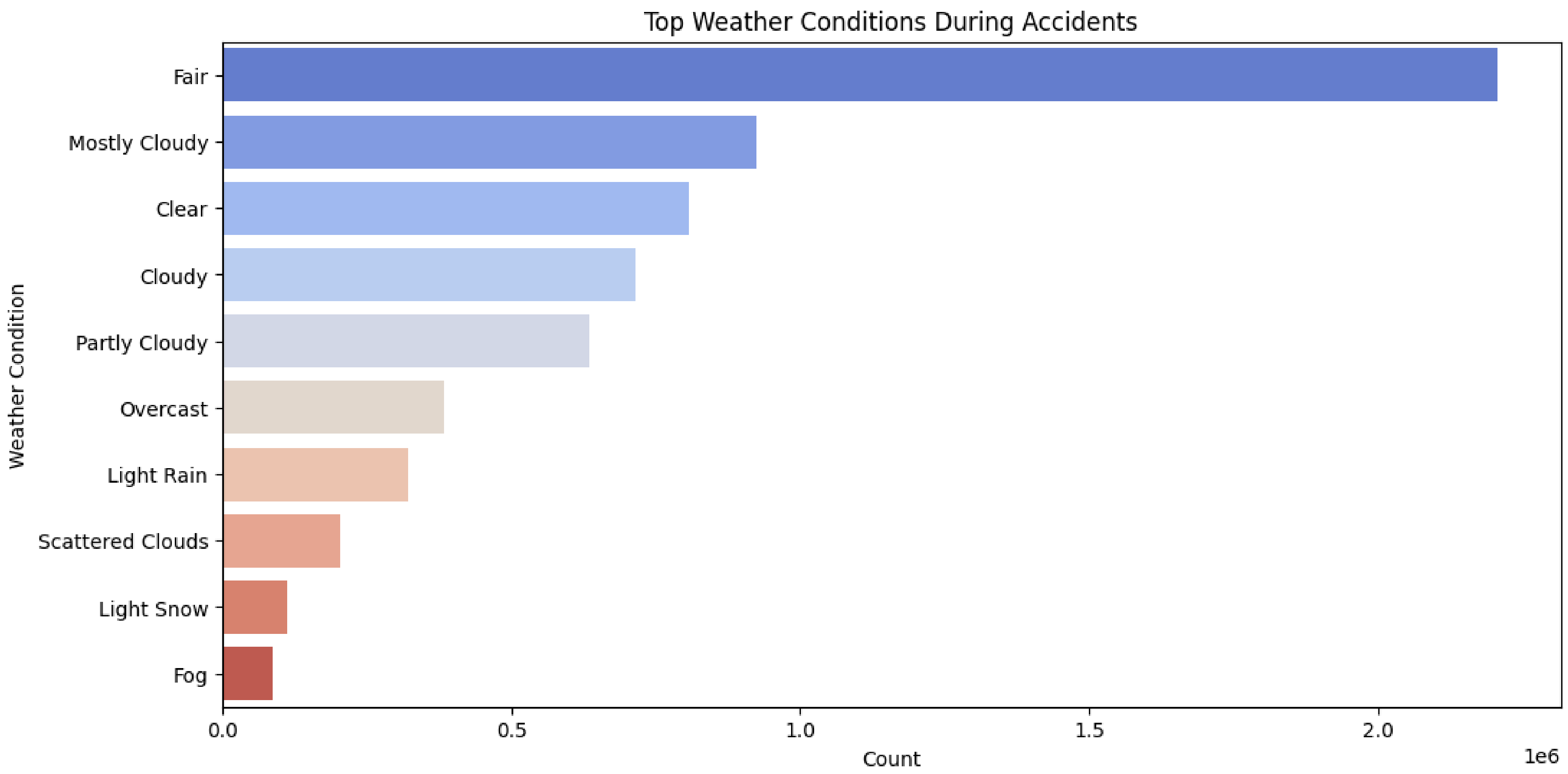
```
df1
```

```
['ID',
 'Source',
 'Severity',
 'Start_Time',
 'End_Time',
 'Start_Lat',
 'Start_Lng',
 'End_Lat',
 'End_Lng',
 'Distance(mi)',
 'Description',
 'Street',
 'City',
 'County',
 'State',
 'Zipcode',
 'Country',
 'Timezone',
 'Airport_Code',
 'Weather_Timestamp',
 'Temperature(F)',
 'Wind_Chill(F)',
 'Humidity(%)',
 'Pressure(in)',
 'Visibility(mi)',
 'Wind_Direction',
 'Wind_Speed(mph)',
 'Precipitation(in)',
 'Weather_Condition',
 'Amenity',
 'Bump',
 'Crossing',
 'Give_Way',
      'Junction',
      'No_Exit',
      'Railway',
      'Roundabout',
      'Station',
      'Stop',
      'Traffic_Calming',
      'Traffic_Signal',
      'Turning_Loop',
      'Sunrise_Sunset',
      'Civil_Twilight',
      'Nautical_Twilight',
      'Astronomical_Twilight']


# Load only necessary columns
cols = ['Severity', 'Start_Time', 'Start_Lat', 'Start_Lng',
'City', 'State', 'Weather_Condition', 'Temperature(F)', 'Humidity(%)',
'Visibility(mi)', 'Wind_Speed(mph)', 'Precipitation(in)',
'Amenity', 'Crossing', 'Junction', 'Traffic_Signal', 'Sunrise_Sunset']


df1['Start_Time'] = pd.to_datetime(df['Start_Time'], errors='coerce')
df['Hour'] = df['Start_Time'].dt.hour
df['Day'] = df['Start_Time'].dt.day_name()
df.dropna(subset=['Start_Lat', 'Start_Lng', 'Start_Time'], inplace=True)


plt.figure(figsize=(12,6))
sns.countplot(x='Hour', data=df1, palette='magma')
plt.title("Accidents by Hour of Day")
plt.xlabel("Hour")
plt.ylabel("Number of Accidents")
plt.grid(True)
plt.show()
```

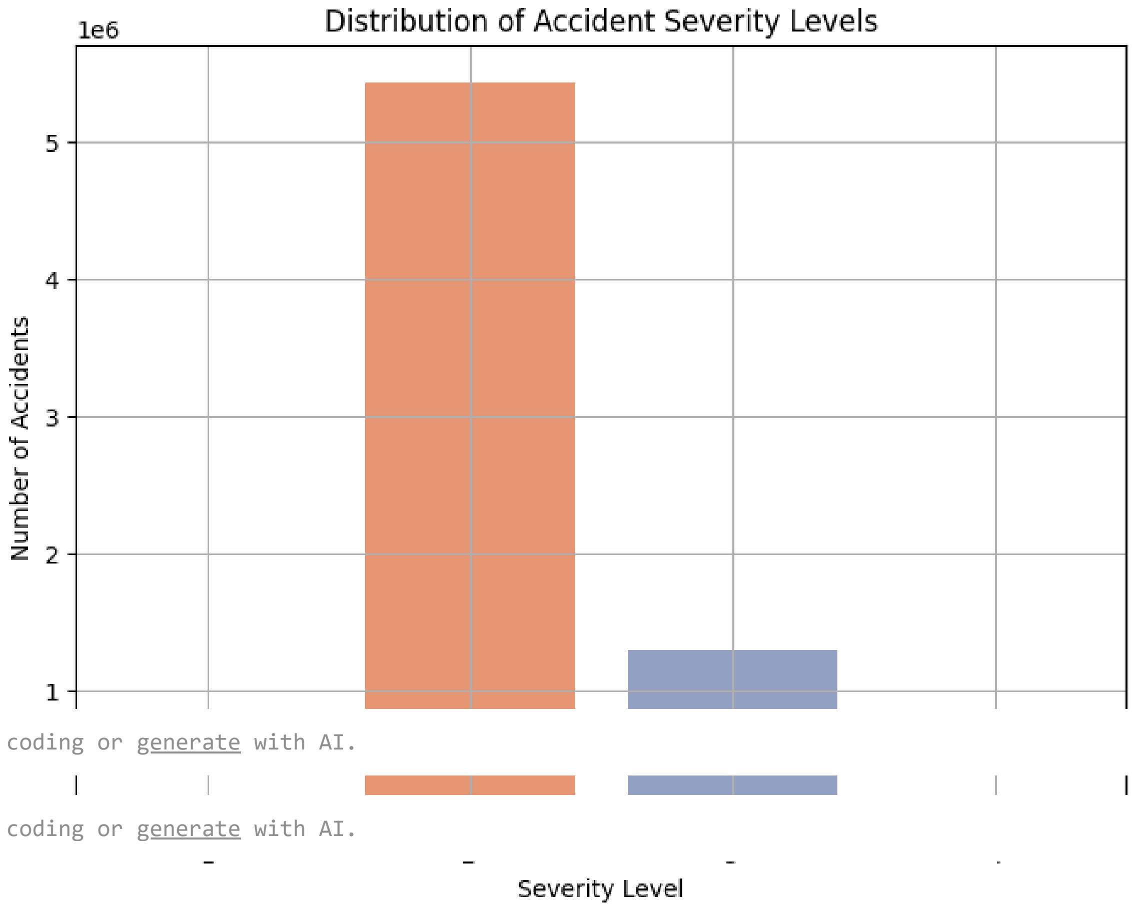
```
/tmp/ipython-input-6-3387798337.py:2: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `legend=False

  sns.countplot(x='Hour', data=df, palette='magma')
```



```
plt.figure(figsize=(12,6))
top_weather = df1['Weather_Condition'].value_counts().nlargest(10).index
sns.countplot(y='Weather_Condition', data=df1[df['Weather_Condition'].isin(top_weather)], order=top_weather, palette='coolwarm')
plt.title("Top Weather Conditions During Accidents")
plt.xlabel("Count")
plt.ylabel("Weather Condition")
plt.show()
```



```
plt.figure(figsize=(8,6))
sns.countplot(x='Severity', data=df1, palette='Set2')
plt.title("Distribution of Accident Severity Levels")
plt.xlabel("Severity Level")
plt.ylabel("Number of Accidents")
plt.grid(True)
plt.show()
```



Start coding or generate with AI.

Start coding or generate with AI.