TASK 5:- TRAFFIC ACCIDENT DATA ANALYSIS

The task here is to analyze traffic accident data to identify patterns related to road conditions, weather, and time of day. Visualize accident hotspots and contributing factors.

The dataset used here is Road Accidents Dataset from Kaggle.

https://www.kaggle.com/datasets/nezukokamaado/road-accident-casualties-dataset

IMPORTING THE LIBRARIES

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

IMPORTING THE DATASET

```
data = pd.read_csv('Accident Data.csv')
data
```

	Index	Accident_Severity	Accident Date	Latitude	Light_Conditions	Di
0	200701BS64157	Serious	05-06-2019	51.506187	Darkness - lights lit	Ken C
1	200701BS65737	Serious	02-07-2019	51.495029	Daylight	Ken C
2	200701BS66127	Serious	26-08-2019	51.517715	Darkness - lighting unknown	Ken C
3	200701BS66128	Serious	16-08-2019	51.495478	Daylight	Ken C
4	200701BS66837	Slight	03-09-2019	51.488576	Darkness - lights lit	Ken C
...	...	...	...	...	...	
660674	201091NM01760	Slight	18-02-2022	57.374005	Daylight	H
660675	201091NM01881	Slight	21-02-2022	57.232273	Darkness - no lighting	H
660676	201091NM01935	Slight	23-02-2022	57.585044	Daylight	H
660677	201091NM01964	Serious	23-02-2022	57.214898	Darkness - no lighting	H
660678	201091NM02142	Serious	28-02-2022	57.575210	Daylight	H

660679 rows × 14 columns

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 660679 entries, 0 to 660678
Data columns (total 14 columns):
 #   Column                   Non-Null Count   Dtype
---  ------                   --------------   -----
 0   Index                    660679 non-null  object
 1   Accident_Severity        660679 non-null  object
 2   Accident Date            660679 non-null  object
 3   Latitude                 660654 non-null  float64
 4   Light_Conditions         660679 non-null  object
 5   District Area            660679 non-null  object
 6   Longitude                660653 non-null  float64
 7   Number_of_Casualties     660679 non-null  int64
 8   Number_of_Vehicles       660679 non-null  int64
 9   Road_Surface_Conditions  659953 non-null  object
 10  Road_Type                656159 non-null  object
```

```
 11  Urban_or_Rural_Area      660664 non-null  object
 12  Weather_Conditions       646551 non-null  object
 13  Vehicle_Type             660679 non-null  object
dtypes: float64(2), int64(2), object(10)
memory usage: 70.6+ MB
```

```
# Handle missing values and clean the data
data.dropna(inplace=True)
```

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 642796 entries, 0 to 660678
Data columns (total 14 columns):
 #   Column                   Non-Null Count   Dtype
---  ------                   --------------   -----
 0   Index                    642796 non-null  object
 1   Accident_Severity        642796 non-null  object
 2   Accident Date            642796 non-null  object
 3   Latitude                 642796 non-null  float64
 4   Light_Conditions         642796 non-null  object
 5   District Area            642796 non-null  object
 6   Longitude                642796 non-null  float64
 7   Number_of_Casualties     642796 non-null  int64
 8   Number_of_Vehicles       642796 non-null  int64
 9   Road_Surface_Conditions  642796 non-null  object
 10  Road_Type                642796 non-null  object
 11  Urban_or_Rural_Area      642796 non-null  object
 12  Weather_Conditions       642796 non-null  object
 13  Vehicle_Type             642796 non-null  object
dtypes: float64(2), int64(2), object(10)
memory usage: 73.6+ MB
```

```
data.describe(include = 'all')
```

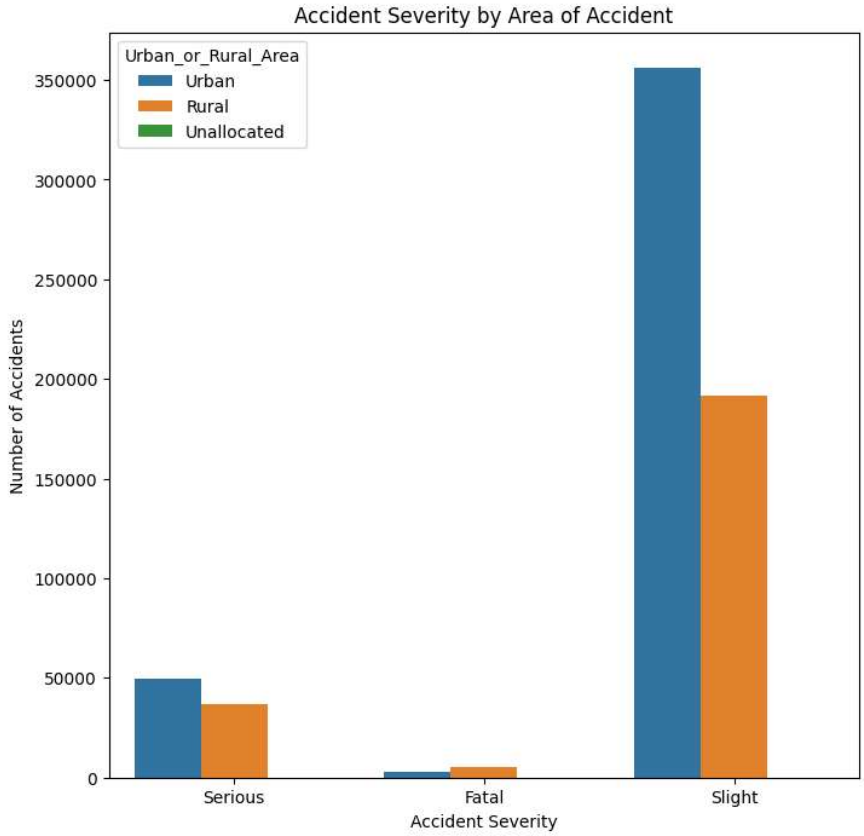
	Index	Accident_Severity	Accident Date	Latitude	Light_Conditions	Dist
count	642796	642796	642796	642796.000000	642796	64
unique	410527	3	1461	NaN	5	
top	2.01E+12	Slight	31-01-2019	NaN	Daylight	Birmin
freq	232088	547721	679	NaN	471923	1
mean	NaN	NaN	NaN	52.554473	NaN	
std	NaN	NaN	NaN	1.410298	NaN	
min	NaN	NaN	NaN	49.914430	NaN	
25%	NaN	NaN	NaN	51.490413	NaN	
50%	NaN	NaN	NaN	52.304737	NaN	
75%	NaN	NaN	NaN	53.455597	NaN	
max	NaN	NaN	NaN	60.757544	NaN	

```
data.head(10)
```

	Index	Accident_Severity	Accident Date	Latitude	Light_Conditions	Distric Are
0	200701BS64157	Serious	05-06-2019	51.506187	Darkness - lights lit	Kensingtc ar Chelse
1	200701BS65737	Serious	02-07-2019	51.495029	Daylight	Kensingtc ar Chelse
3	200701BS66128	Serious	16-08-2019	51.495478	Daylight	Kensingtc ar Chelse
5	200701BS67159	Serious	18-09-2019	51.497750	Daylight	Kensingtc ar Chelse
6	200701BS67207	Serious	05-09-2019	51.501405	Daylight	Kensingtc ar Chelse
7	200701BS67370	Fatal	03-10-2019	51.482260	Darkness - lights lit	Kensingtc ar Chelse
8	200701BS67515	Slight	31-10-2019	51.493319	Darkness - lights lit	Kensingtc ar Chelse
9	200701BS67543	Slight	18-10-2019	51.484539	Daylight	Kensingtc ar Chelse
10	200701BS67644	Serious	09-10-2019	51.491944	Darkness - lights lit	Kensingtc ar Chelse
11	200701BS67747	Fatal	02-11-2019	51.499127	Daylight	Kensingtc ar Chelse

ANALYSIS

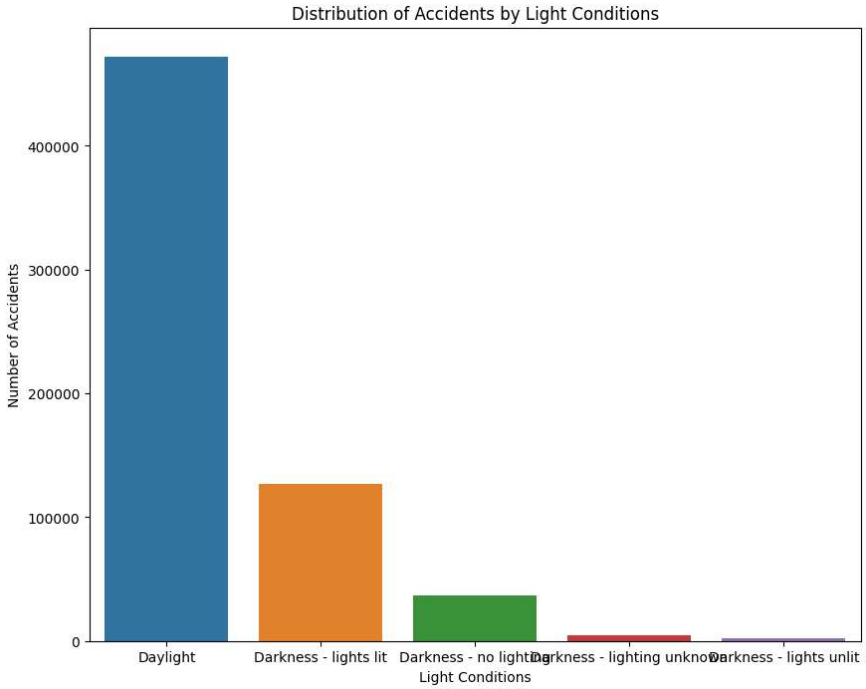
```
plt.figure(figsize=(8, 8))
sns.countplot(x='Accident_Severity', data=data, hue='Urban_or_Rural_Area')
plt.title('Accident Severity by Area of Accident')
plt.xlabel('Accident Severity')
plt.ylabel('Number of Accidents')
plt.show()
```



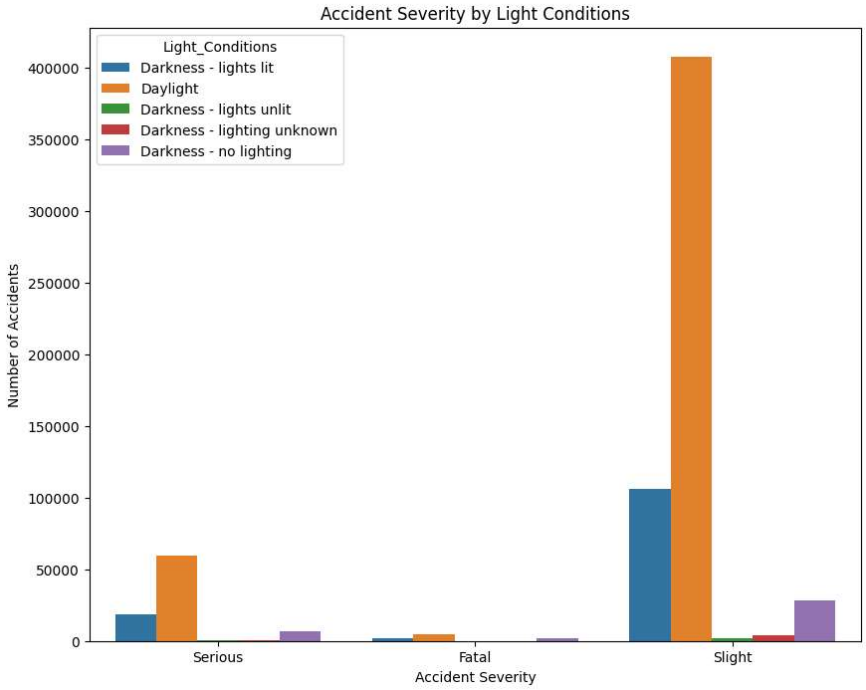

```
light_conditions_counts = data['Light_Conditions'].value_counts()
print(light_conditions_counts)

plt.figure(figsize=(10, 8))
sns.barplot(x=light_conditions_counts.index, y=light_conditions_counts.values)
plt.title('Distribution of Accidents by Light Conditions')
plt.xlabel('Light Conditions')
plt.ylabel('Number of Accidents')
plt.show()
```

```
Daylight                      471923
Darkness - lights lit         126767
Darkness - no lighting         36868
Darkness - lighting unknown     4766
Darkness - lights unlit         2472
Name: Light_Conditions, dtype: int64
```

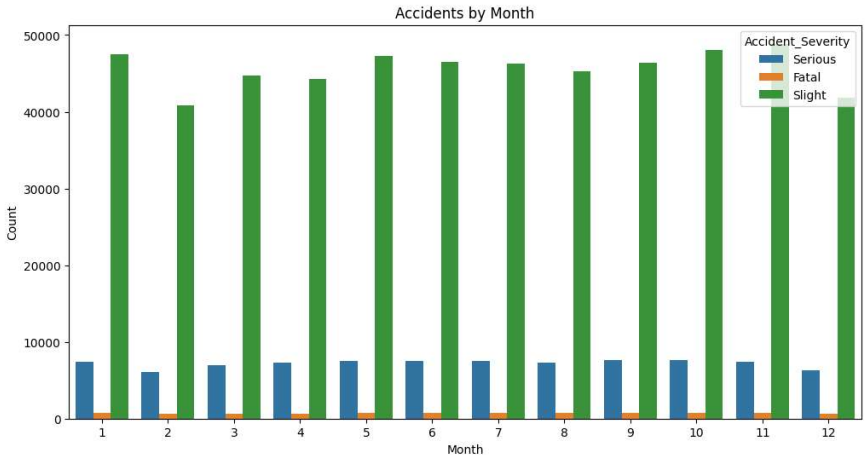



```
plt.figure(figsize=(10, 8))
sns.countplot(x='Accident_Severity', data=data, hue='Light_Conditions')
plt.title('Accident Severity by Light Conditions')
plt.xlabel('Accident Severity')
plt.ylabel('Number of Accidents')
plt.show()
```

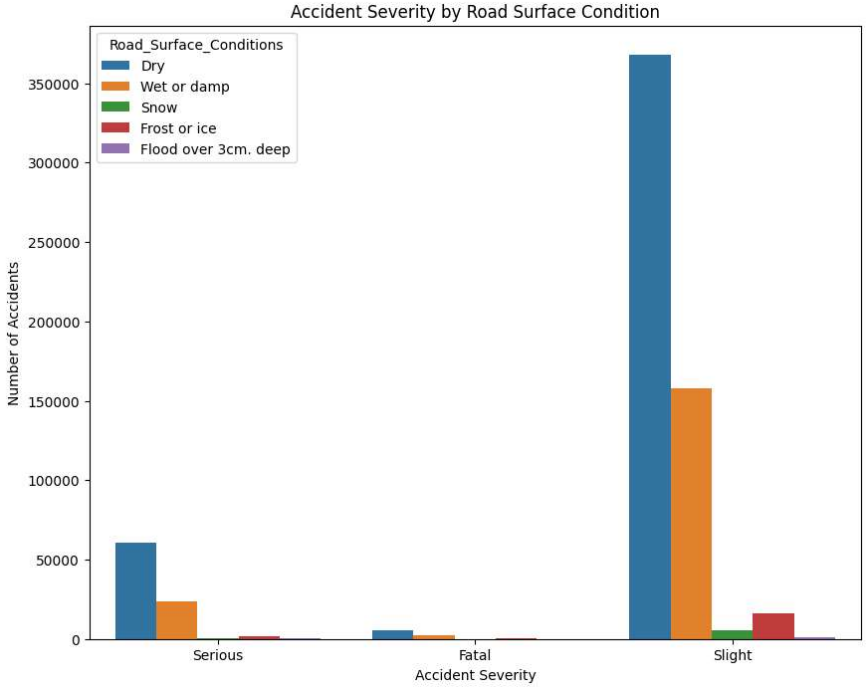



```
# Time analysis
data['Accident Date'] = pd.to_datetime(data['Accident Date'])
data['Month'] = data['Accident Date'].dt.month
plt.figure(figsize=(12, 6))
sns.countplot(x='Month', data=data, hue='Accident_Severity')
plt.title('Accidents by Month')
plt.xlabel('Month')
plt.ylabel('Count')
plt.show()
```

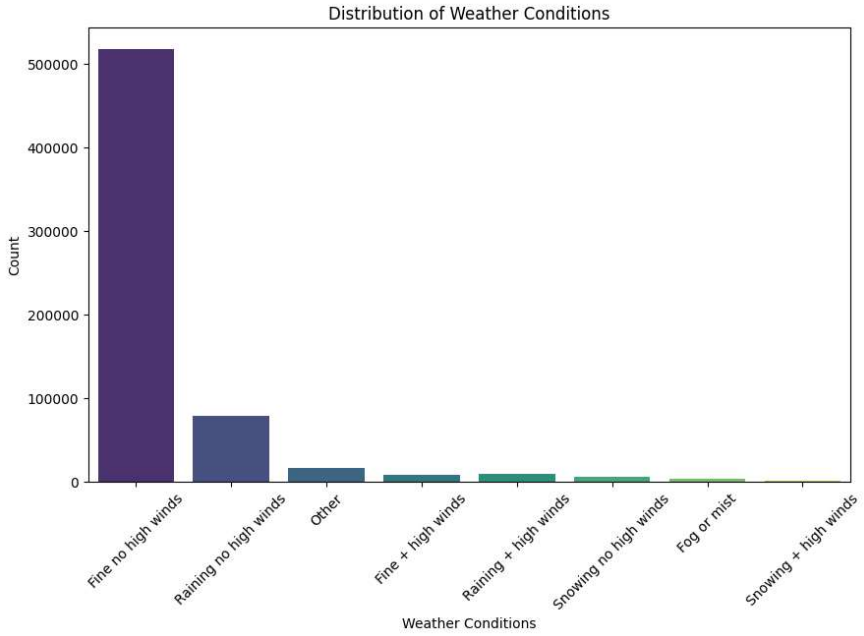
```
<ipython-input-59-9fd753c09edb>:2: UserWarning: Parsing dates in DD/MM/YYYY format wh
  data['Accident Date'] = pd.to_datetime(data['Accident Date'])
```



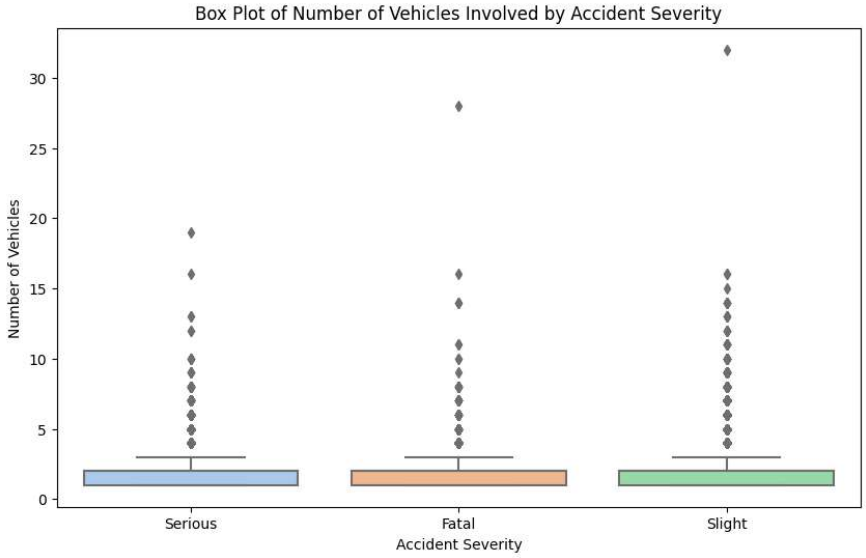
```
plt.figure(figsize=(10, 8))
sns.countplot(x='Accident_Severity', data=data, hue='Road_Surface_Conditions')
plt.title('Accident Severity by Road Surface Condition')
plt.xlabel('Accident Severity')
plt.ylabel('Number of Accidents')
plt.show()
```




```
plt.figure(figsize=(10, 6))
sns.countplot(x='Weather_Conditions', data=data, palette='viridis')
plt.title('Distribution of Weather Conditions')
plt.xlabel('Weather Conditions')
plt.ylabel('Count')
plt.xticks(rotation=45)
plt.show()
```




```
plt.figure(figsize=(10, 6))
sns.boxplot(x='Accident_Severity', y='Number_of_Vehicles', data=data, palette='pastel')
plt.title('Box Plot of Number of Vehicles Involved by Accident Severity')
plt.xlabel('Accident Severity')
plt.ylabel('Number of Vehicles')
plt.show()
```



Inference:-

The number of vehicles involved in accidents tends to increase with the severity of the accident. This is likely because more serious accidents often involve multiple vehicles colliding with each other, while slight accidents may only involve two vehicles.

The distribution of the number of vehicles involved in accidents is relatively symmetrical for all three severities. There are a few outliers in each of the severity categories. These outliers could be due to factors such as chain-reaction accidents or accidents involving large trucks or buses.

```
plt.figure(figsize=(10, 8))
columns = ['Latitude', 'Longitude', 'Number_of_Casualties', 'Number_of_Vehicles', 'Month']
correlation_matrix = data[columns].corr()
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm', linewidths=.5)
plt.title('Correlation Matrix')
plt.show()
```