**RESULT SUMMARY**

**Goal - 1:** **Determine the most common vehicle speed present during fatal crashes**

spark.sql("SELECT INVAGE AS Speed, COUNT(\*) AS RecordCount FROM fatalities GROUP BY INVAGE ORDER BY RecordCount DESC LIMIT 1").show()

A close-up of a speed record

Description automatically generated

Visual Code:

import pandas as pd

import matplotlib.pyplot as plt

# Assuming 'result' is the output of your Spark SQL query

result = spark.sql("SELECT INVAGE AS Speed, COUNT(\*) AS RecordCount FROM fatalities GROUP BY INVAGE ORDER BY RecordCount DESC LIMIT 1").toPandas()

# Plotting the bar graph

plt.bar(result['Speed'], result['RecordCount'], color='blue')

plt.xlabel('Vehicle Speed')

plt.ylabel('Number of Crashes')

plt.title('Vehicle Speed Distribution in Crashes')

plt.show()

A graph of a vehicle speed distribution

Description automatically generated

This goal aims to analyze and identify the predominant vehicle speed associated with crashes. By examining crash data, the project will provide insights into the most common or prevalent speed at the time of accidents, offering valuable information for understanding patterns and potential contributing factors to improve road safety measures.

**Goal - 2:** **Highlight the district with the highest number of fatal crashes**

spark.sql("SELECT DISTRICT AS DISTRICT, COUNT(\*) AS Fatalities FROM fatalities GROUP BY DISTRICT ORDER BY Fatalities DESC LIMIT 1").show()

A close-up of a line

Description automatically generated

Visual Code:

import pandas as pd

import matplotlib.pyplot as plt

# Assuming 'result' is the output of your Spark SQL query

result = spark.sql("SELECT DISTRICT AS DISTRICT, COUNT(\*) AS Fatalities FROM fatalities GROUP BY DISTRICT ORDER BY Fatalities DESC LIMIT 1").toPandas()

# Plotting the bar graph

plt.bar(result['DISTRICT'], result['Fatalities'], color='red')

plt.xlabel('District')

plt.ylabel('Number of Fatalities')

plt.title('Fatalities by District')

plt.xticks(rotation=45) # Rotate x-axis labels for better readability

plt.tight\_layout() # Adjust layout to prevent clipping of labels

plt.show()

A red rectangular object with white text

Description automatically generated

This goal was implemented to determine which district experienced the highest number of fatal car crashes. By analyzing relevant data, the project aims to identify geographic patterns and areas with elevated risks. This information can be crucial for directing targeted interventions, implementing safety measures, and formulating policies to reduce the incidence of fatal car accidents in specific districts.

**Goal - 3: Identify the street with the highest number of fatal crashes**

spark.sql("SELECT STREET1 as Street\_Name, count(\*) AS Fatalities FROM fatalities GROUP BY Street\_Name ORDER BY Fatalities DESC LIMIT 10").show()

A white paper with black text

Description automatically generated

Visual Code:

import pandas as pd

import matplotlib.pyplot as plt

# Assuming 'result' is the output of your Spark SQL query

result = spark.sql("SELECT STREET1 as Street\_Name, count(\*) AS Fatalities FROM fatalities GROUP BY Street\_Name ORDER BY Fatalities DESC LIMIT 10").toPandas()

# Plotting the bar graph

plt.bar(result['Street\_Name'], result['Fatalities'], color='green')

plt.xlabel('Street Name')

plt.ylabel('Number of Fatalities')

plt.title('Fatalities by Street Name (Top 10)')

plt.xticks(rotation=45) # Rotate x-axis labels for better readability

plt.tight\_layout() # Adjust layout to prevent clipping of labels

plt.show()

**A graph of green bars

Description automatically generated**

This objective aims to identify the street that recorded the highest number of fatal crashes. By analyzing relevant data, the project will pinpoint specific locations with elevated risks, allowing for a focused approach in implementing safety measures, traffic management strategies, and targeted interventions to enhance road safety on the identified street.

**Goal - 4: Highlight the difference in fatal crash occurrences between day and night**

spark.sql("SELECT LIGHT as Light\_Conditions, count(\*) AS Fatalities FROM fatalities GROUP BY Light\_Conditions ORDER BY Fatalities DESC").show()

A screenshot of a computer

Description automatically generated

Visual Code:

import pandas as pd

import matplotlib.pyplot as plt

# Assuming 'result' is the output of your Spark SQL query

result = spark.sql("SELECT LIGHT as Light\_Conditions, count(\*) AS Fatalities FROM fatalities GROUP BY Light\_Conditions ORDER BY Fatalities DESC").toPandas()

# Plotting the bar graph

plt.bar(result['Light\_Conditions'], result['Fatalities'], color='purple')

plt.xlabel('Light Conditions')

plt.ylabel('Number of Fatalities')

plt.title('Fatalities by Light Conditions')

plt.xticks(rotation=45) # Rotate x-axis labels for better readability

plt.tight\_layout() # Adjust layout to prevent clipping of labels

plt.show()

A graph of purple bars

Description automatically generated with medium confidence

This goal involves analyzing data to quantify the number of fatal crashes that occurred during the day and night, as well as different artificial lighting conditions. By distinguishing between daytime and nighttime incidents, the project aims to provide insights into temporal patterns of fatal crashes. This information can be valuable for understanding potential factors contributing to accidents during specific periods, guiding efforts to improve safety measures, and enhancing overall road safety strategies.

**Goal - 5: Determine the impact of road condition on fatal crash occurrences**

spark.sql("SELECT RDSFCOND AS Road\_Conditions, count(\*) AS Fatalities FROM fatalities GROUP BY Road\_Conditions ORDER BY Fatalities DESC").show()

A white paper with black text

Description automatically generated

Visual Code:

import pandas as pd

import matplotlib.pyplot as plt

# Assuming 'result' is the output of your Spark SQL query

result = spark.sql("SELECT RDSFCOND AS Road\_Conditions, count(\*) AS Fatalities FROM fatalities GROUP BY Road\_Conditions ORDER BY Fatalities DESC").toPandas()

# Removing rows with NULL values

result = result.dropna()

# Plotting the bar chart

plt.figure(figsize=(10, 6))

plt.bar(result['Road\_Conditions'], result['Fatalities'], color='blue')

plt.xlabel('Road Conditions')

plt.ylabel('Number of Fatalities')

plt.title('Fatalities by Road Conditions')

plt.xticks(rotation=45) # Rotate x-axis labels for better readability

plt.tight\_layout() # Adjust layout to prevent clipping of labels

plt.show()

A graph with blue squares

Description automatically generated

This objective involves analyzing data to quantify the number of fatal crashes that occurred during wet road conditions compared to dry conditions. By distinguishing between these weather conditions, the project aims to provide insights into the impact of road surface conditions on fatal crashes. This information can be crucial for understanding the relationship between weather-related factors and road safety, guiding efforts to implement weather-specific safety measures and enhance overall road safety strategies.

**Goal - 6: Identify the year with the highest number of fatal car crashes**

spark.sql("SELECT YEAR as Year, count(\*) AS Fatalities FROM fatalities GROUP BY Year ORDER BY Year DESC LIMIT 1").show()

A close-up of a price tag

Description automatically generated

Visual Code:

import pandas as pd

import matplotlib.pyplot as plt

# Assuming 'result' is the output of your Spark SQL query

result = spark.sql("SELECT YEAR as Year, count(\*) AS Fatalities FROM fatalities GROUP BY Year ORDER BY Year DESC LIMIT 1").toPandas()

# Plotting the line plot

plt.figure(figsize=(10, 6))

plt.plot(result['Year'], result['Fatalities'], marker='o', color='green', linestyle='-')

plt.xlabel('Year')

plt.ylabel('Number of Fatalities')

plt.title('Fatalities by Year')

plt.grid(True)

plt.show()

A graph with green dots

Description automatically generated

This goal involves analyzing historical data to identify the year with the highest number of fatal car crashes. By examining trends over time, the project aims to pinpoint specific years that experienced elevated risks, providing valuable insights into potential contributing factors. This information can be essential for understanding temporal patterns in road safety and may help guide efforts to address factors associated with the identified year of peak fatal crashes.

**Goal - 7: Determine which traffic control had the most fatal collisions**

spark.sql("SELECT TRAFFCTL AS Traffic\_Control, count(\*) AS Fatalities FROM fatalities GROUP BY Traffic\_Control ORDER BY Fatalities DESC").show()

A white text with black text

Description automatically generated with medium confidence

Visual Code:

import pandas as pd

import matplotlib.pyplot as plt

# Assuming 'result' is the output of your Spark SQL query

result = spark.sql("SELECT TRAFFCTL AS Traffic\_Control, count(\*) AS Fatalities FROM fatalities GROUP BY TRAFFCTL ORDER BY Fatalities DESC").toPandas()

# Remove rows with None values

result = result.dropna()

# Plotting the bar chart

plt.figure(figsize=(10, 6))

plt.bar(result['Traffic\_Control'], result['Fatalities'], color='orange')

plt.xlabel('Traffic Control')

plt.ylabel('Number of Fatalities')

plt.title('Fatalities by Traffic Control')

plt.xticks(rotation=45) # Rotate x-axis labels for better readability

plt.tight\_layout() # Adjust layout to prevent clipping of labels

plt.show()

A graph of traffic signal

Description automatically generated with medium confidence

This objective involves analyzing data to identify the type of traffic control (e.g., stop signs, traffic signals, yield signs) associated with the highest number of fatal collisions. By examining patterns related to different traffic control measures, the project aims to provide insights into the effectiveness or vulnerabilities of specific control systems. This information can be valuable for enhancing traffic management strategies, improving signage, and implementing targeted interventions to reduce fatal collisions at intersections with the identified traffic control.

**Goal - 8: Provide insights into the prevalence of speeding in fatal crashes**

spark.sql("SELECT SPEEDING AS Speeding, count(\*) AS Fatalities FROM fatalities GROUP BY Speeding ORDER BY Fatalities DESC").show()

A close-up of a test

Description automatically generated

Visual Code:

import pandas as pd

import matplotlib.pyplot as plt

# Assuming 'result' is the output of your Spark SQL query

result = spark.sql("SELECT SPEEDING AS Speeding, count(\*) AS Fatalities FROM fatalities GROUP BY Speeding ORDER BY Fatalities DESC").toPandas()

# Plotting the donut chart

plt.figure(figsize=(8, 6))

# Colors for the pie chart

colors = ['lightblue', 'lightgreen']

# Plotting the pie chart

plt.pie(result['Fatalities'], labels=result['Speeding'], colors=colors, autopct='%1.1f%%', startangle=90, pctdistance=0.85)

# Draw a circle to create a donut chart

centre\_circle = plt.Circle((0,0),0.70,fc='white')

fig = plt.gcf()

fig.gca().add\_artist(centre\_circle)

# Equal aspect ratio ensures that pie is drawn as a circle

plt.axis('equal')

plt.title('Fatalities by Speeding')

plt.tight\_layout()

plt.show()

A blue circle with green and blue text

Description automatically generated

This goal involves analyzing data to quantify the number of fatal collisions that occurred when drivers were exceeding the speed limit. By examining the correlation between speed violations and fatal accidents, the project aims to provide insights into the impact of speeding on road safety. This information can be crucial for understanding the relationship between speed-related factors and fatal collisions, guiding efforts to enforce speed limits, implement targeted interventions, and enhance overall strategies for reducing speeding-related fatalities.

**Goal - 9: Highlight the significance of pedestrian safety in fatal crash incidents**

spark.sql("SELECT PEDACT AS Pedestrian\_Action\_Description, count(\*) AS Fatalities FROM fatalities WHERE PEDACT='Crossing with right of way' GROUP BY Pedestrian\_Action\_Description").show()

A close-up of a text

Description automatically generated

Visual code:

import pandas as pd

import seaborn as sns

import matplotlib.pyplot as plt

# Assuming 'result' is the output of your Spark SQL query

result = spark.sql("SELECT PEDACT AS Pedestrian\_Action\_Description, count(\*) AS Fatalities FROM fatalities WHERE PEDACT='Crossing with right of way' GROUP BY Pedestrian\_Action\_Description").toPandas()

# Plotting the horizontal bar plot with Seaborn

plt.figure(figsize=(10, 6))

sns.barplot(x='Fatalities', y='Pedestrian\_Action\_Description', data=result, palette='viridis')

plt.xlabel('Number of Fatalities')

plt.ylabel('Pedestrian Action Description')

plt.title('Fatalities by Pedestrian Action Description')

plt.tight\_layout()

plt.show() A graph with a number of collision crossing

Description automatically generated

This objective involves analyzing data to determine the count of fatal collisions that occurred while pedestrians were crossing the right of way. By examining incidents where pedestrians were following proper crossing procedures, the project aims to provide insights into pedestrian safety and potential risks associated with crossing designated areas. This information can be valuable for understanding the dynamics of pedestrian-involved fatal collisions, guiding efforts to enhance pedestrian safety measures, and informing public awareness campaigns to reduce accidents in crosswalks and designated pedestrian areas.

**Goal - 10: Identify the most common state of condition among drivers in fatal collisions**

spark.sql("SELECT ALCOHOL AS Alcohol, count(\*) AS Fatalities FROM fatalities WHERE ALCOHOL='Yes' GROUP BY Alcohol").show()

A black lines with words

Description automatically generated with medium confidence

Visual Code:

import pandas as pd

import matplotlib.pyplot as plt

# Assuming 'result' is the output of your Spark SQL query

result = spark.sql("SELECT ALCOHOL AS Alcohol, count(\*) AS Fatalities FROM fatalities WHERE ALCOHOL='Yes' GROUP BY Alcohol").toPandas()

# Plotting the line chart

plt.figure(figsize=(8, 6))

plt.plot(result['Alcohol'], result['Fatalities'], marker='o', color='blue', linestyle='-')

plt.xlabel('Alcohol Involved')

plt.ylabel('Number of Fatalities')

plt.title('Fatalities Involving Alcohol')

plt.grid(True)

plt.tight\_layout()

plt.show()   
A graph with a blue dot

Description automatically generated

This goal involves analyzing data to determine the drivers' state of condition associated with the most fatal collisions. By examining driver impairment, the project aims to provide insights into the primary contributing factors to fatal collisions. This information can be crucial for understanding the role of driver behavior and conditions in road safety, guiding efforts to address specific issues, and informing strategies for promoting safer driving practices to reduce the number of fatal collisions.