```
import pandas as pd
import numpy as np
```

Dataset Selection:

```
df = pd.read_csv("/content/accidents_2017.csv.zip")
```

```
 # This is formatted as code
```

Exploring the Dataset:

The head() method returns a specified number of rows, string from the top. The head() method returns the first 5 rows if a number is not specified.

```
df.head(5)
```

	Id	District Name	Neighborhood Name	Street	Weekday	Month	Day	Hour	Part of the day	Mild injuries	Serious injuries	Victims	Vehicles involved	Longitud
0	2017S008429	Unknown	Unknown	Número 27 ...	Friday	October	13	8	Morning	2	0	2	2	2.12562
1	2017S007316	Unknown	Unknown	Número 3 Zona Franca / Número 50 Zona Franca ...	Friday	September	1	13	Morning	2	0	2	2	2.12045
2	2017S010210	Unknown	Unknown	Litoral (Besòs) ...	Friday	December	8	21	Afternoon	5	0	5	2	2.16735

The tail() method returns a specified number of last rows. The tail() method returns the last 5 rows if a number is not specified.

```
df.tail(5)
```

	Id	District Name	Neighborhood Name	Street	Weekday	Month	Day	Hour	Part of the day	Mild injuries	Serious injuries	Victims	Vehicle involve
10334	2017S003667	Sant Andreu	el Bon Pastor	Litoral (Llobregat) ...	Tuesday	April	25	8	Morning	1	0	1	
10335	2017S001896	Sant Andreu	el Bon Pastor	PL MONTERREY ...	Wednesday	March	8	12	Morning	1	0	1	
10336	2017S010718	Sant Andreu	el Bon Pastor	Litoral (Llobregat) ...	Thursday	December	28	8	Morning	1	0	1	
10337	2017S006145	Sant Andreu	el Bon Pastor	Litoral (Besòs) ...	Friday	July	14	14	Afternoon	1	0	1	
10338	2017S000178	Sant Andreu	el Bon Pastor	CIUTAT D'ASUNCIÓN ...	Sunday	January	8	20	Afternoon	0	0	0	

The sample() method returns a list with a specified number of randomly selected items from a sequence.

```
#Retrieving sample rows from a data frame.
df.sample(3)
```

	Id	District Name	Neighborhood Name	Street	Weekday	Month	Day	Hour	Part of the day	Mild injuries	Serious injuries	Victims	Vehicles involved	Longitu
9983	2017S008417	Sarrià-Sant Gervasi	les Tres Torres	Vergós ...	Thursday	October	12	16	Afternoon	1	0	1	2	2.1275
5202	2017S008467	Sant Martí	el Parc i la Llacuna del Poblenou	Llull / Joan d'Àustria ...	Saturday	October	14	18	Afternoon	2	0	2	2	2.1913

The info() method prints information about the DataFrame. The information contains the number of columns, column labels, column data types, memory usage, range index, and the number of cells in each column (non-null values).

```
#Retrieving information about the data frame
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10339 entries, 0 to 10338
Data columns (total 15 columns):
 #   Column             Non-Null Count  Dtype  
---  ------             --------------  -----  
 0   Id                 10339 non-null  object 
 1   District Name      10339 non-null  object 
 2   Neighborhood Name  10339 non-null  object 
 3   Street             10339 non-null  object 
 4   Weekday            10339 non-null  object 
 5   Month              10339 non-null  object 
 6   Day                10339 non-null  int64  
 7   Hour               10339 non-null  int64  
 8   Part of the day    10339 non-null  object 
 9   Mild injuries      10339 non-null  int64  
 10  Serious injuries   10339 non-null  int64  
 11  Victims            10339 non-null  int64  
 12  Vehicles involved  10339 non-null  int64  
 13  Longitude          10339 non-null  float64
 14  Latitude           10339 non-null  float64
dtypes: float64(2), int64(6), object(7)
memory usage: 1.2+ MB
```

the shape() method is used to fetch the dimensions of Pandas and NumPy type objects in python. Every value represented by the tuple corresponds to the actual dimension in terms of array or row/columns.

```
#Display the number of rows and columns.
df.shape
```

```
(10339, 15)
```

columns is an attribute that provides access to the column labels of a data frame. It returns an Index object representing the names of the columns in the DataFrame.

```
#Display columns name and data
df.columns
```

```
Index(['Id', 'District Name', 'Neighborhood Name', 'Street', 'Weekday',
       'Month', 'Day', 'Hour', 'Part of the day', 'Mild injuries',
       'Serious injuries', 'Victims', 'Vehicles involved', 'Longitude',
       'Latitude'],
      dtype='object')
```

This will print the starting 3 values of Id column

```
df['Id'].head(3)
```

```
0    2017S008429
1    2017S007316
2    2017S010210
Name: Id, dtype: object
```

The describe() method is used for calculating some statistical data like percentile, mean and std of the numerical values of the Series or DataFrame. It analyzes both numeric and object series and also the DataFrame column sets of mixed data types.

```
# Display summary statistics
df.describe(include='all')
```

	Id	District Name	Neighborhood Name	Street	Weekday	Month	Day	Hour	Part of the day	Mild injuries	Seriou injurie
count	10339	10339	10339	10339	10339	10339	10339.000000	10339.000000	10339	10339.000000	10339.00000
unique	10335	11	74	4253	7	12	NaN	NaN	3	NaN	Na
top	2017S008856	Eixample	la Dreta de l'Eixample	Corts Catalanes ...	Friday	November	NaN	NaN	Afternoon	NaN	Na
freq	2	3029	1167	219	1761	991	NaN	NaN	5082	NaN	Na
mean	NaN	NaN	NaN	NaN	NaN	NaN	15.775994	13.811394	NaN	1.154174	0.02331
std	NaN	NaN	NaN	NaN	NaN	NaN	8.763455	5.316490	NaN	0.742294	0.16380
min	NaN	NaN	NaN	NaN	NaN	NaN	1.000000	0.000000	NaN	0.000000	0.00000
25%	NaN	NaN	NaN	NaN	NaN	NaN	8.000000	10.000000	NaN	1.000000	0.00000
50%	NaN	NaN	NaN	NaN	NaN	NaN	16.000000	14.000000	NaN	1.000000	0.00000
75%	NaN	NaN	NaN	NaN	NaN	NaN	23.000000	18.000000	NaN	1.000000	0.00000
max	NaN	NaN	NaN	NaN	NaN	NaN	31.000000	23.000000	NaN	10.000000	4.00000

This Method is for Retrieving a Range of Rows

```
# for display 2nd to 6th rows
df[2:7]
```

	Id	District Name	Neighborhood Name	Street	Weekday	Month	Day	Hour	Part of the day	Mild injuries	Serious injuries	Victims	Vehicles involved	Longit
2	2017S010210	Unknown	Unknown	Litoral (Besòs) ...	Friday	December	8	21	Afternoon	5	0	5	2	2.167
3	2017S006364	Unknown	Unknown	Número 3 Zona Franca ...	Friday	July	21	2	Night	1	0	1	2	2.124
4	2017S004615	Sant Martí	el Camp de l'Arpa del Clot	Las Navas de Tolosa ...	Thursday	May	25	14	Afternoon	1	0	1	3	2.185
				Indústria										

```
# for display starting to 10th
df[:11]
```

	Id	District Name	Neighborhood Name	Street	Weekday	Month	Day	Hour	Part of the day	Mild injuries	Serious injuries	Victims	Vehicles involved	Longi
0	2017S008429	Unknown	Unknown	Número 27 ...	Friday	October	13	8	Morning	2	0	2	2	2.12
1	2017S007316	Unknown	Unknown	Número 3 Zona Franca / Número 50 Zona Franca ...	Friday	September	1	13	Morning	2	0	2	2	2.12
2	2017S010210	Unknown	Unknown	Litoral (Besòs) ...	Friday	December	8	21	Afternoon	5	0	5	2	2.16
3	2017S006364	Unknown	Unknown	Número 3 Zona Franca ...	Friday	July	21	2	Night	1	0	1	2	2.12
4	2017S004615	Sant Martí	el Camp de l'Arpa del Clot	Las Navas de Tolosa ...	Thursday	May	25	14	Afternoon	1	0	1	3	2.18
5	2017S007775	Sant Martí	el Camp de l'Arpa del Clot	Indústria / Trinxant ...	Wednesday	September	20	12	Morning	1	0	1	2	2.18
6	2017S004484	Sant Martí	el Camp de l'Arpa del Clot	Trinxant / Indústria ...	Saturday	May	20	21	Afternoon	1	0	1	2	2.18

```
# for display last two rows
df[-2:]
```

	Id	District Name	Neighborhood Name	Street	Weekday	Month	Day	Hour	Part of the day	Mild injuries	Serious injuries	Victims	Vehicles involved	Lo
10337	2017S006145	Sant Andreu	el Bon Pastor	Litoral (Besòs) ...	Friday	July	14	14	Afternoon	1	0	1	2	2
10338	2017S000178	Sant Andreu	el Bon Pastor	CIUTAT D'ASUNCIÓN ...	Sunday	January	8	20	Afternoon	0	0	0	1	2

This creates a copy or duplicate dataframe

```
# create new df_col dataframe from df.copy() method.
df_new = df.copy()
```

This renames a column

```
# rename columns name
df_new.rename(columns={"Neighborhood Name": "Area"}, inplace=True)
df_new.head(3)
```

	Id	District Name	Area	Street	Weekday	Month	Day	Hour	Part of the day	Mild injuries	Serious injuries	Victims	Vehicles involved	Longitude	La
0	2017S008429	Unknown	Unknown	Número 27 ...	Friday	October	13	8	Morning	2	0	2	2	2.125624	41
				Número 3 Zona Franca /											

This creates a new column while copying same data

```
# Add a People_Involved column whose value will be same as Victims
df_new['People_Involved'] = df_new['Victims']
df_new.head(3)
```

	Id	District Name	Area	Street	Weekday	Month	Day	Hour	Part of the day	Mild injuries	Serious injuries	Victims	Vehicles involved	Longitude	La
0	2017S008429	Unknown	Unknown	Número 27 ...	Friday	October	13	8	Morning	2	0	2	2	2.125624	41
1	2017S007316	Unknown	Unknown	Número 3 Zona Franca / Número 50 Zona Franca ...	Friday	September	1	13	Morning	2	0	2	2	2.120452	41
2	2017S010210	Unknown	Unknown	Litoral (Besòs) ...	Friday	December	8	21	Afternoon	5	0	5	2	2.167356	41

This deletes columns

```
# Drop unwanted columns
df_new.drop(['People_Involved'], axis=1, inplace=True)
df_new.head(3)
```

	Id	District Name	Area	Street	Weekday	Month	Day	Hour	Part of the day	Mild injuries	Serious injuries	Victims	Vehicles involved	Longitude	La
0	2017S008429	Unknown	Unknown	Número 27 ...	Friday	October	13	8	Morning	2	0	2	2	2.125624	41
				Número 3 Zona Franca /											

This shows duplicate values and deletes them

```
# Display duplicated entries
df_new.duplicated().sum()
# dropping ALL duplicate values
df_new.drop_duplicates(keep = 'first', inplace = True)
```

This fills empty or null values with "unknown"

```
df['Neighborhood Name'].fillna('Unknown', inplace=True)
df_new.head(3)
```

	Id	District Name	Area	Street	Weekday	Month	Day	Hour	Part of the day	Mild injuries	Serious injuries	Victims	Vehicles involved	Longitude	La
0	2017S008429	Unknown	Unknown	Número 27 ...	Friday	October	13	8	Morning	2	0	2	2	2.125624	41
				Número 3 Zona Franca /											

Finding Mode

```
#Finding Mode Of Month Column
df_new['Month'].mode()
```

```
0    November
Name: Month, dtype: object
```

Finding Mean

```
#Finding Mean In Mild Injuries Column
df_new['Mild injuries'].mean()
```

```
1.1542331881954524
```

Finding Median

```
#Finding Median In Mild Injuries Column
df_new['Mild injuries'].median()
```

```
1.0
```

Checking For Null Or Missing Values

```
#Checking Null Values
df_new.isnull().sum()
```

```
Id                   0
District Name        0
Area                 0
Street               0
Weekday              0
Month                0
Day                  0
Hour                 0
Part of the day      0
Mild injuries        0
Serious injuries     0
Victims              0
Vehicles involved    0
Longitude            0
Latitude             0
dtype: int64
```

Imputing forward fill or backfill by ffill and bfill. In ffill missing value impute from the value of the above row and for bfill it's taken from the below rows value.

```
df_new['Part of the day'].fillna(method='ffill', inplace=True)
```

Number of unique values in the category column

```
# for display how many unique values are there in Part of the day column
df_new['Part of the day'].nunique()
```

```
3
```

```
#Shows all unique values
# for display uniqe values of Part of the day column
df_new['Part of the day'].unique()
```

```
array(['Morning', 'Afternoon', 'Night'], dtype=object)
```

```
#Counts of unique values
df['Weekday'].value_counts()
```

```
Weekday
Friday       1761
Tuesday      1691
Thursday     1677
Wednesday    1650
Monday       1510
Saturday     1155
Sunday        895
Name: count, dtype: int64
```

```
# Calculate percentage of each category
df['Weekday'].value_counts(normalize=True)
```

```
Weekday
Friday       0.170326
Tuesday      0.163555
Thursday     0.162201
Wednesday    0.159590
Monday       0.146049
Saturday     0.111713
Sunday       0.086565
Name: proportion, dtype: float64
```

Sorting Values

```
# Sort Values by Hour
df.sort_values(by=['Hour']).head(3)
```

	Id	District Name	Neighborhood Name	Street	Weekday	Month	Day	Hour	Part of the day	Mild injuries	Serious injuries	Victims	Vehicles involved	Longitud
2619	2017S001382	Sant Martí	Provençals del Poblenou	SELVA DE MAR / Perú ...	Sunday	February	19	0	Night	1	0	1	2	2.20289
3988	2017S006134	Ciutat Vella	el Barri Gòtic	Colom / Antonio López ...	Friday	July	14	0	Night	2	0	2	2	2.18197
1716	2017S009425	Eixample	la Dreta de	Corts Catalanes	Tuesday	November	14	0	Night	2	0	2	2	2.16827

```
# Sort Values Victims with descending order
df.sort_values(by=['Victims'], ascending=False).head(3)
```

	Id	District Name	Neighborhood Name	Street	Weekday	Month	Day	Hour	Part of the day	Mild injuries	Serious injuries	Victims	Vehicles involved	Lo
8345	2017S009365	Sants-Montjuïc	el Poble-sec	Litoral (Llobregat) ...	Saturday	November	11	21	Afternoon	10	0	10	3	
10301	2017S008068	Sant Andreu	Sant Andreu	Torras i Bages ...	Saturday	September	30	13	Morning	10	0	10	2	
681	2017S005291	Sants-Montjuïc	la Marina del Prat Vermell	Litoral (Besòs) ...	Friday	June	16	20	Afternoon	9	0	9	5	

Conditional queries on Data If we want to apply a single condition then first we will give one condition then we pass on the data frame. For example, if we want to display all rows where Month is December then we use this:

```
# filtering - Only show December Accidents
condition = df['Month'] == 'December'
df[condition].head(5)
```

	Id	District Name	Neighborhood Name	Street	Weekday	Month	Day	Hour	Part of the day	Mild injuries	Serious injuries	Victims	Vehicles involved	Lo
2	2017S010210	Unknown	Unknown	Litoral (Besòs) ...	Friday	December	8	21	Afternoon	5	0	5	2	
7	2017S010680	Sant Martí	el Camp de l'Arpa del Clot	Indústria ...	Tuesday	December	26	20	Afternoon	2	0	2	1	
10	2017S010348	Sant Martí	el Camp de l'Arpa del Clot	Indústria ...	Thursday	December	14	20	Afternoon	1	0	1	2	
18	2017S010102	Sant Martí	el Camp de l'Arpa del Clot	Rosselló / Independència ...	Monday	December	4	0	Night	2	0	2	2	
66	2017S010414	Sant Martí	el Besòs i el Maresme	Eduard Maristany ...	Saturday	December	16	18	Afternoon	1	0	1	2	

We can apply multiple conditional queries like before. For example, if we want to display accidents in Sant MartÃ in December

```
# first create 2 condition
condition1 = df['District Name'] == 'Sant MartÃ'
condition2 = df['Month'] == 'December'


# we passing condition on our dataframe
df[condition1 & condition2].head(4)
```

Id	District Name	Neighborhood Name	Street	Weekday	Month	Day	Hour	Part of the	Mild injuries	Serious injuries	Victims	Vehicles involved	Longitude	Latitude

In Pandas group by function is more popular in data analysis parts. It allows to split and group data, apply a function, and combine the results. Grouping by one column: For example, if we want to find maximum values of District Name and Part of the day by number of Victims then we can use this:

```
df[['District Name', 'Part of the day']].groupby(df['Victims']).max()
```

Victims	District Name	Part of the day
0	Unknown	Night
1	Unknown	Night
2	Unknown	Night
3	Sarrià-Sant Gervasi	Night
4	Sarrià-Sant Gervasi	Night
5	Unknown	Night
6	Sarrià-Sant Gervasi	Night
7	Sants-Montjuïc	Night
8	Eixample	Morning
9	Sants-Montjuïc	Afternoon
10	Sants-Montjuïc	Morning

Creating List and Series

```
# importing module
from pandas import *

# reading CSV file
data = read_csv("/content/accidents_2017.csv.zip")

# converting column data to list
Id = data['Id'].tolist()
District = data['District Name'].tolist()
Neighborhood = data['Neighborhood Name'].tolist()
Street = data['Street'].tolist()
Weekday = data['Weekday'].tolist()
Month = data['Month'].tolist()
Day = data['Day'].tolist()
Hour = data['Hour'].tolist()
Time = data['Part of the day'].tolist()
Mild_injuries = data['Mild injuries'].tolist()
Serious_injuries = data['Serious injuries'].tolist()
Victims = data['Victims'].tolist()
Vehicles_involved = data['Vehicles involved'].tolist()
Longitude = data['Longitude'].tolist()
Latitude = data['Latitude'].tolist()

# printing list data
print('Id:', Id)
print('District:', District)
print('Neighborhood:', Neighborhood)
print('Street:', Street)
print('Weekday:', Weekday)
print('Month:', Month)
print('Day:', Day)
print('Hour:', Hour)
print('Time:', Time)
print('Mild injuries:', Mild_injuries)
print('Serious injuries:', Serious_injuries)
print('Victims:', Victims)
print('Vehicles involved:', Vehicles_involved)
print('Longitude:', Longitude)
print('Latitude:', Latitude)
```

```
Id: ['2017S008429    ', '2017S007316    ', '2017S010210    ', '2017S006364    ', '2017S004615    ', '2017S007775    ', '2017S004484    '
District: ['Unknown', 'Unknown', 'Unknown', 'Unknown', 'Sant Martí', 'Sant Martí', 'Sant Martí', 'Sant Martí', 'Sant Martí', 'Sant Martí
Neighborhood: ['Unknown', 'Unknown', 'Unknown', 'Unknown', "el Camp de l'Arpa del Clot", "el Camp de l'Arpa del Clot", "el Camp de l'Arp
Street: ['Número 27                                         ', 'Número 3 Zona Franca / Número 50 Zona Franca      ', 'Litoral (Besòs)
Weekday: ['Friday', 'Friday', 'Friday', 'Friday', 'Thursday', 'Wednesday', 'Saturday', 'Tuesday', 'Monday', 'Wednesday', 'Thursday', 'We
Month: ['October', 'September', 'December', 'July', 'May', 'September', 'May', 'December', 'June', 'May', 'December', 'January', 'June',
Day: [13, 1, 8, 21, 25, 20, 20, 26, 12, 3, 14, 11, 30, 4, 30, 17, 25, 9, 4, 17, 14, 20, 30, 20, 7, 22, 8, 7, 17, 14, 31, 15, 2, 11, 31,
Hour: [8, 13, 21, 2, 14, 12, 21, 20, 15, 20, 20, 7, 12, 16, 19, 14, 14, 22, 0, 16, 14, 20, 20, 19, 11, 20, 12, 6, 13, 16, 21, 18, 9, 15,
Time: ['Morning', 'Morning', 'Afternoon', 'Night', 'Afternoon', 'Morning', 'Afternoon', 'Afternoon', 'Afternoon', 'Afternoon', 'Afternoo
Mild injuries: [2, 2, 5, 1, 1, 1, 1, 2, 1, 1, 1, 1, 1, 0, 1, 1, 4, 1, 2, 1, 1, 0, 1, 1, 1, 1, 1, 2, 0, 1, 0, 1, 1, 1, 2, 1, 1, 1, 1, 1,
Serious injuries: [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0,
Victims: [2, 2, 5, 1, 1, 1, 1, 2, 1, 1, 1, 1, 1, 1, 1, 1, 4, 1, 2, 1, 1, 1, 1, 1, 1, 1, 1, 2, 0, 1, 1, 1, 1, 1, 2, 1, 1, 1, 1, 1, 1, 1,
Vehicles involved: [2, 2, 2, 2, 3, 2, 2, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 1, 2, 2, 2, 2, 2, 2, 1, 2, 2, 4, 2, 1, 1, 2, 2, 2, 3, 1, 1, 1, 2,
Longitude: [2.12562442, 2.12045245, 2.1673561, 2.12452894, 2.185272, 2.183245, 2.183245, 2.183561, 2.184059, 2.181225, 2.18229, 2.180453
Latitude: [41.34004482, 41.33942606, 41.3608855, 41.33766786, 41.416365, 41.416336, 41.416336, 41.416372, 41.416763, 41.413958, 41.41577
```

```
#creating series from lists
series1 = pd.Series(Id)
series2 = pd.Series(District)
series3 = pd.Series(Neighborhood)
series4 = pd.Series(Street)
series5 = pd.Series(Weekday)
series6 = pd.Series(Month)
series7 = pd.Series(Day)
series8 = pd.Series(Hour)
series9 = pd.Series(Time)
series10 = pd.Series(Mild_injuries)
series11 = pd.Series(Serious_injuries)
series12 = pd.Series(Victims)
series13 = pd.Series(Vehicles_involved)
series14 = pd.Series(Longitude)
series15 = pd.Series(Latitude)

# printing series data
print('Id:', series1)
print('District:', series2)
print('Neighborhood:', series3)
print('Street:', series4)
print('Weekday:', series5)
print('Month:', series6)
print('Day:', series7)
print('Hour:', series8)
print('Time:', series9)
print('Mild injuries:', series10)
print('Serious injuries:', series11)
print('Victims:', series12)
print('Vehicles involved:', series13)
print('Longitude:', series14)
print('Latitude:', series15)
```

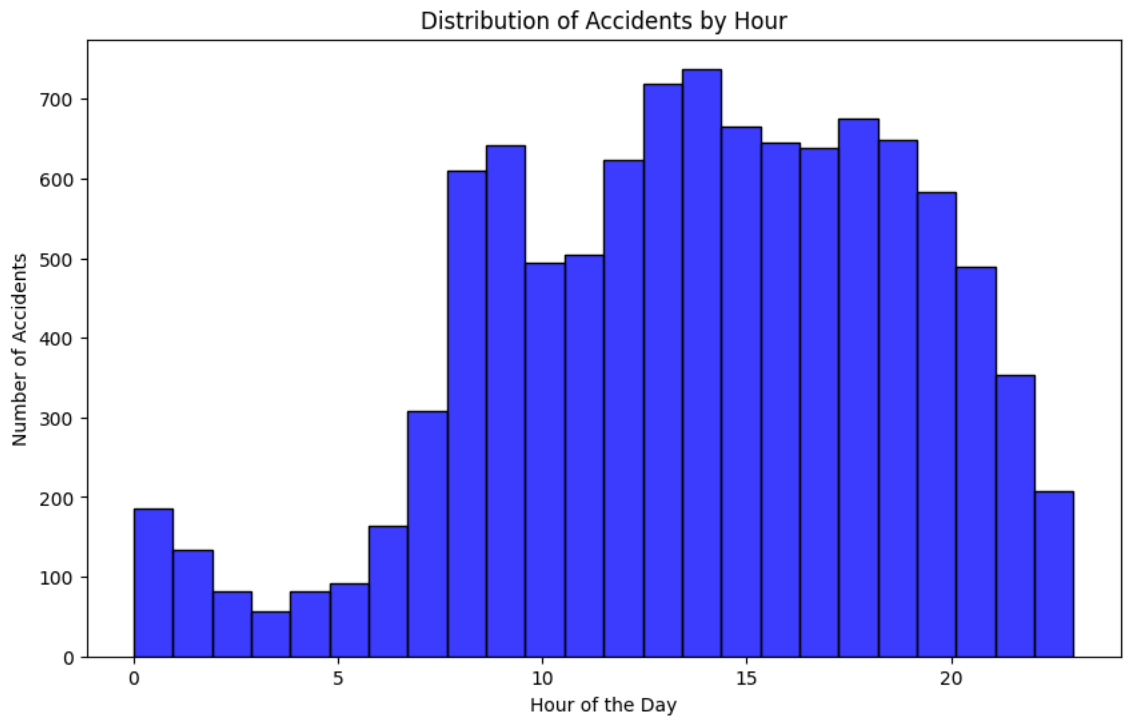
```
10337    1
10338    0
Length: 10339, dtype: int64
Vehicles involved: 0        2
1        2
2        2
3        2
4        3
        ..
10334    3
10335    2
10336    2
10337    2
10338    1
Length: 10339, dtype: int64
Longitude: 0        2.125624
1        2.120452
2        2.167356
3        2.124529
4        2.185272
           ...
10334    2.201800
10335    2.206013
10336    2.205607
10337    2.205118
10338    2.200956
Length: 10339, dtype: float64
Latitude: 0        41.340045
1        41.339426
2        41.360886
3        41.337668
4        41.416365
           ...
10334    41.392004
10335    41.443445
10336    41.443894
10337    41.444824
10338    41.437125
Length: 10339, dtype: float64
```

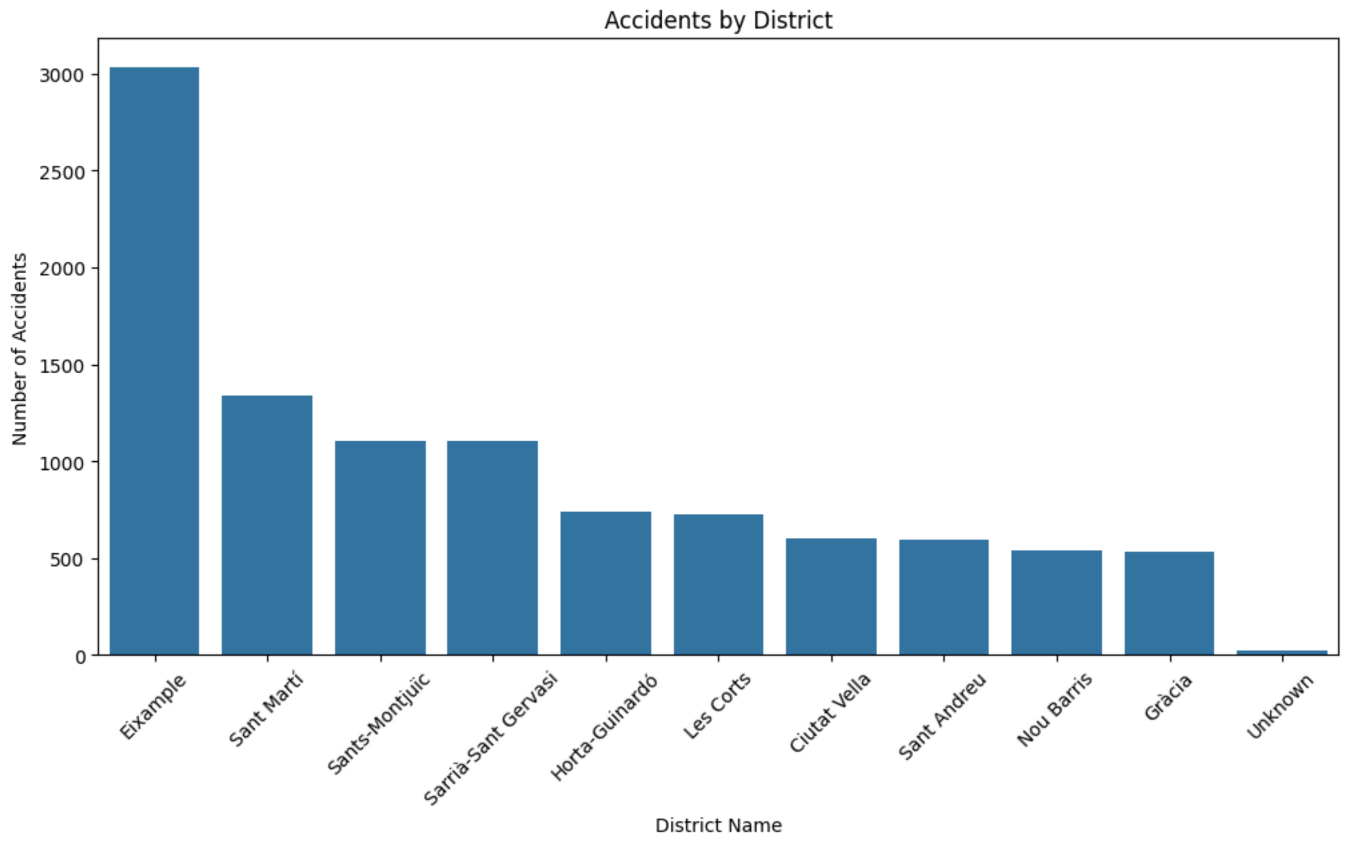
```
# display tenth value in the series
print(series1[9])
```

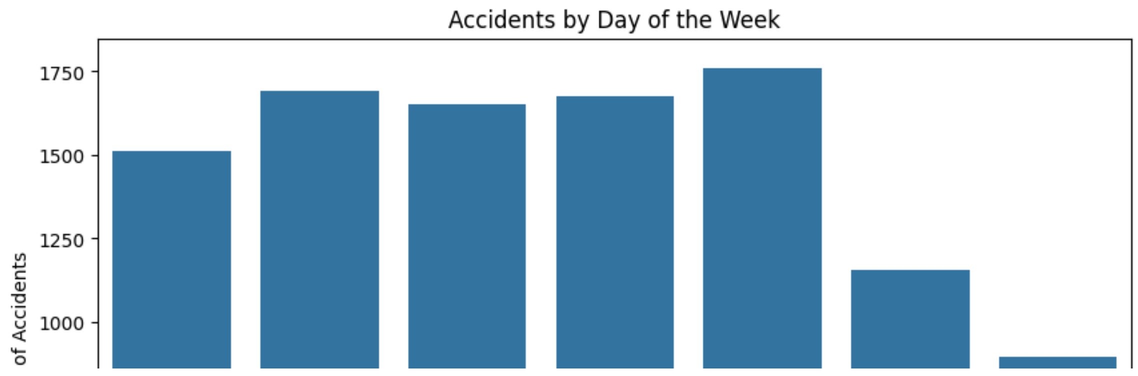
```
2017S003932
```

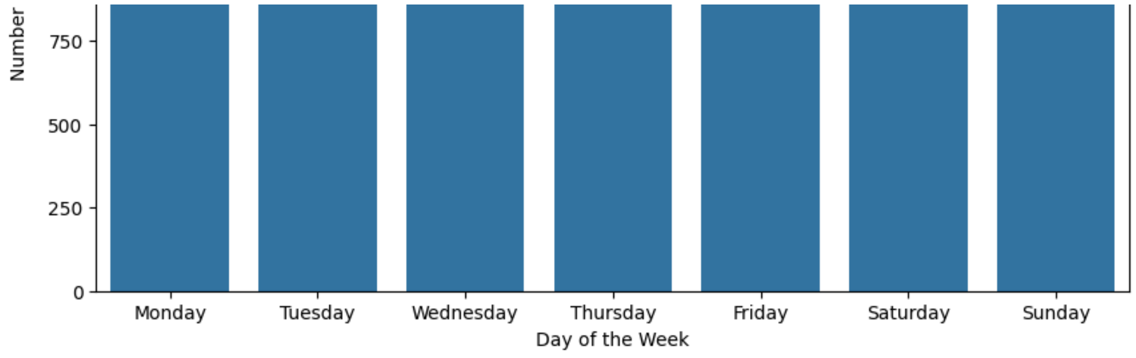
```
#Data Visualisation:

import matplotlib.pyplot as plt
import seaborn as sns

# Distribution of accidents by hour
plt.figure(figsize=(10, 6))
sns.histplot(df['Hour'], bins=24, kde=False, color='blue')
plt.title('Distribution of Accidents by Hour')
plt.xlabel('Hour of the Day')
plt.ylabel('Number of Accidents')
plt.show()

# Accidents by district
plt.figure(figsize=(12, 6))
sns.countplot(data=df, x='District Name', order=df['District Name'].value_counts().index)
plt.title('Accidents by District')
plt.xlabel('District Name')
plt.ylabel('Number of Accidents')
plt.xticks(rotation=45)
plt.show()

# Accidents by day of the week
plt.figure(figsize=(10, 6))
sns.countplot(data=df, x='Weekday', order=['Monday', 'Tuesday', 'Wednesday', 'Thursday', 'Friday', 'Saturday', 'Sunday'])
plt.title('Accidents by Day of the Week')
plt.xlabel('Day of the Week')
plt.ylabel('Number of Accidents')
plt.show()

# Correlation matrix - Excluding non-numeric columns
plt.figure(figsize=(12, 8))
# Select only numeric columns for correlation calculation
numeric_df = df.select_dtypes(include=['number'])
correlation_matrix = numeric_df.corr()
```

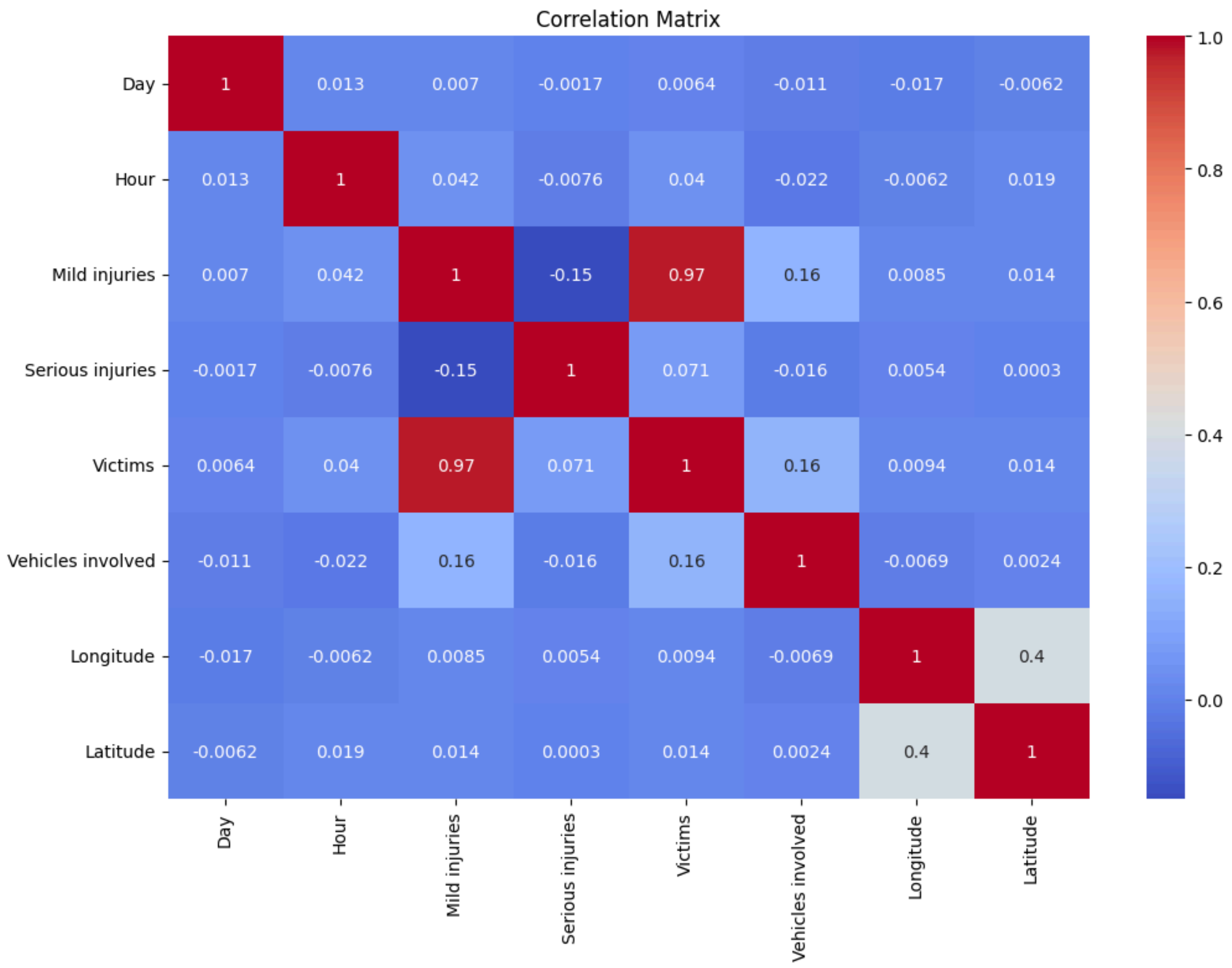
```
    sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm')
    plt.title('Correlation Matrix')
    plt.show()
```


Distribution of Accidents by Hour
Number of Accidents
0
100
200
300
400
500
600
700
0
5
10
15
20
Hour of the Day


Accidents by District
Number of Accidents
0
500
1000
1500
2000
2500
3000
Eixample
Sant Martí
Sants-Montjuïc
Sarrià-Sant Gervasi
Horta-Guinardó
Les Corts
Ciutat Vella
Sant Andreu
Nou Barris
Gràcia
Unknown
District Name


Accidents by Day of the Week
of Accidents
1000
1250
1500
1750





	Day	Hour	Mild injuries	Serious injuries	Victims	Vehicles involved	Longitude	Latitude
Day	1	0.013	0.007	-0.0017	0.0064	-0.011	-0.017	-0.0062
Hour	0.013	1	0.042	-0.0076	0.04	-0.022	-0.0062	0.019
Mild injuries	0.007	0.042	1	-0.15	0.97	0.16	0.0085	0.014
Serious injuries	-0.0017	-0.0076	-0.15	1	0.071	-0.016	0.0054	0.0003
Victims	0.0064	0.04	0.97	0.071	1	0.16	0.0094	0.014
Vehicles involved	-0.011	-0.022	0.16	-0.016	0.16	1	-0.0069	0.0024
Longitude	-0.017	-0.0062	0.0085	0.0054	0.0094	-0.0069	1	0.4
Latitude	-0.0062	0.019	0.014	0.0003	0.014	0.0024	0.4	1

Preprocessing the dataset and handling the missing values:

```
missing_values = df.isnull().sum()
missing_values
```

```
Id                   0
District Name        0
Neighborhood Name    0
Street               0
Weekday              0
Month                0
Day                  0
Hour                 0
Part of the day      0
Mild injuries        0
Serious injuries     0
Victims              0
Vehicles involved    0
Longitude            0
Latitude             0
dtype: int64
```

Selecting Input features and Label

```
input_features = ['Hour', 'Day', 'Month', 'District Name', 'Vehicles involved']
```

```
target_variable = 'Victims'
```

Documentation

Input Features:

Hour: Time of the day when the accident occurred.

Day: Day of the month. Month: Month of the year.

District Name: District where the accident occurred.

Vehicles involved: Number of vehicles involved in the accident.

Target Variable:

Victims: Number of victims in each accident.

```
# Determine if the problem is classification or regression
target_variable = 'Victims'

if df[target_variable].dtype in ['int64', 'float64']:
    print(f"The problem is a regression task, as the target variable '{target_variable}' is continuous.")
else:
    unique_values = df[target_variable].nunique()
    if unique_values == 2:
        print(f"The problem is a binary classification task, as the target variable '{target_variable}' has two distinct values.")
    else:
        print(f"The problem is a multiclass classification task, as the target variable '{target_variable}' has {unique_values} distinct val

# Multi-label or Multi-output determination
# Since we have identified it as a regression problem with a single target variable, it is neither multi-label nor multi-output.
print(f"The problem is not multi-label or multi-output, as we are predicting a single target variable '{target_variable}'.")
```

```
The problem is a regression task, as the target variable 'Victims' is continuous.
The problem is not multi-label or multi-output, as we are predicting a single target variable 'Victims'.
```

Determining whether the problem is a classification or regression task

Classification: The target variable is categorical, meaning the output falls into distinct classes or categories.

Regression: The target variable is continuous, meaning the output is a real number. In our dataset, the target variable is Victims, which represents the number of victims in each accident.