Project 2 Data Cleaning

In this project we will clean up the car crash report and see how it looks.

Reading data from csv file and importing some of the import statement.

In [66]:

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

In [67]:

```
dataset = pd.read_csv("Traffic_Crashes_-_Crashes.csv")
```

In [68]:

```
dataset.head()
```

Out[68]:

	CRASH_RECORD_ID	CRASH_DATE	POSTED_SPEED_LIMIT	TRAFFIC_CO
0	4fd0a3e0897b3335b94cd8d5b2d2b350eb691add56c62d...	7/10/19 17:56	35	
1	009e9e67203442370272e1a13d6ee51a4155dac65e583d...	6/30/17 16:00	35	STOI
2	ee9283eff3a55ac50ee58f3d9528ce1d689b1c4180b4c4...	7/10/20 10:25	30	
3	f8960f698e870ebdc60b521b2a141a5395556bc3704191...	7/11/20 1:00	30	
4	8eaa2678d1a127804ee9b8c35ddf7d63d913c14eda61d6...	7/8/20 14:00	20	

5 rows × 27 columns

1. Removing extra attribute from the dataset.

In this first task we are removing extra attributes called crash record because it was not that impotant compared to othe data because it was just giving us a crash ID or report ID.

In [69]:

```
dataset.pop('CRASH_RECORD_ID')
dataset.head()
```

Out[69]:

	CRASH_DATE	POSTED_SPEED_LIMIT	TRAFFIC_CONTROL_DEVICE	DEVICE_CONDITION	WEATHER_COND
0	7/10/19 17:56	35	NO CONTROLS	NO CONTROLS	

	CRASH_DATE	POSTED_SPEED_LIMIT	TRAFFIC_CONTROL_DEVICE	DEVICE_CONDITION	WEATHER_COND
1	6/30/17 16:00	35	STOP SIGN/FLASHER	FUNCTIONING PROPERLY	
2	7/10/20 10:25	30	TRAFFIC SIGNAL	FUNCTIONING PROPERLY	
3	7/11/20 1:00	30	NO CONTROLS	NO CONTROLS	
4	7/8/20 14:00	20	NO CONTROLS	NO CONTROLS	

5 rows × 26 columns

2. Splitting attribute from the dataset to many new attributes.

While splitting new attributes from aattributes, I decided to split crash date first into time and date and deleting crash date from the dataset.

In [70]:

```
dataset[['DATE','TIME']] = dataset['CRASH_DATE'].str.split(' ', expand=True)
dataset.pop('CRASH_DATE')
dataset.head()
```

Out[70]:

	POSTED_SPEED_LIMIT	TRAFFIC_CONTROL_DEVICE	DEVICE_CONDITION	WEATHER_CONDITION	LIGHTI
0	35	NO CONTROLS	NO CONTROLS	CLEAR	
1	35	STOP SIGN/FLASHER	FUNCTIONING PROPERLY	CLEAR	
2	30	TRAFFIC SIGNAL	FUNCTIONING PROPERLY	CLEAR	
3	30	NO CONTROLS	NO CONTROLS	CLEAR	
4	20	NO CONTROLS	NO CONTROLS	CLEAR	

5 rows × 27 columns

Then I decided to split date first into month, day of month and year and dropped date and month from the new splitted dataset because month already existed in the dataset.

In [71]:

```
dataset[['DROP','CRASH_DAY_OF_MONTH','CRASH_YEAR']] = dataset['DATE'].str.split('/', ex
dataset.pop('DATE')
dataset.pop('DROP')
dataset.head()
```

Out[71]:

	POSTED_SPEED_LIMIT	TRAFFIC_CONTROL_DEVICE	DEVICE_CONDITION	WEATHER_CONDITION	LIGHTI
0	35	NO CONTROLS	NO CONTROLS	CLEAR	
1	35	STOP SIGN/FLASHER	FUNCTIONING PROPERLY	CLEAR	
2	30	TRAFFIC SIGNAL	FUNCTIONING PROPERLY	CLEAR	
3	30	NO CONTROLS	NO CONTROLS	CLEAR	
4	20	NO CONTROLS	NO CONTROLS	CLEAR	

5 rows × 28 columns

Then I decided to split time into hour and crash minute and dropped hour and time from the new splitted dataset because hour already existed in the dataset.

In [72]:

```
dataset[['HOUR','CRASH_MINUTE']] = dataset['TIME'].str.split(':', expand=True)
dataset.pop('HOUR')
dataset.pop('TIME')
dataset.head()
```

Out[72]:

	POSTED_SPEED_LIMIT	TRAFFIC_CONTROL_DEVICE	DEVICE_CONDITION	WEATHER_CONDITION	LIGHTI
0	35	NO CONTROLS	NO CONTROLS	CLEAR	
1	35	STOP SIGN/FLASHER	FUNCTIONING PROPERLY	CLEAR	
2	30	TRAFFIC SIGNAL	FUNCTIONING PROPERLY	CLEAR	
3	30	NO CONTROLS	NO CONTROLS	CLEAR	
4	20	NO CONTROLS	NO CONTROLS	CLEAR	

5 rows × 28 columns

To make sure all the dataset are stored correctly and still their so I printed dataframe.

In [73]:
```
print(pd.DataFrame(dataset))
```

```
        POSTED_SPEED_LIMIT TRAFFIC_CONTROL_DEVICE      DEVICE_CONDITION  \
0                       35            NO CONTROLS           NO CONTROLS
1                       35      STOP SIGN/FLASHER  FUNCTIONING PROPERLY
2                       30         TRAFFIC SIGNAL  FUNCTIONING PROPERLY
3                       30            NO CONTROLS           NO CONTROLS
4                       20            NO CONTROLS           NO CONTROLS
...                    ...                    ...                   ...
481618                  30            NO CONTROLS           NO CONTROLS
481619                  25            NO CONTROLS               UNKNOWN
481620                  30            NO CONTROLS           NO CONTROLS
481621                  30         TRAFFIC SIGNAL  FUNCTIONING PROPERLY
481622                  30            NO CONTROLS           NO CONTROLS

       WEATHER_CONDITION      LIGHTING_CONDITION      FIRST_CRASH_TYPE  \
0                  CLEAR                DAYLIGHT               TURNING
1                  CLEAR                DAYLIGHT               TURNING
2                  CLEAR                DAYLIGHT              REAR END
3                  CLEAR                DARKNESS  PARKED MOTOR VEHICLE
4                  CLEAR                DAYLIGHT  PARKED MOTOR VEHICLE
...                  ...                     ...                   ...
481618           UNKNOWN                 UNKNOWN  PARKED MOTOR VEHICLE
481619              SNOW                DARKNESS  PARKED MOTOR VEHICLE
481620             CLEAR  DARKNESS, LIGHTED ROAD  PARKED MOTOR VEHICLE
481621             CLEAR  DARKNESS, LIGHTED ROAD               TURNING
481622             CLEAR  DARKNESS, LIGHTED ROAD              REAR END

                        TRAFFICWAY_TYPE ROADWAY_SURFACE_COND ROAD_DEFECT  \
0                               ONE-WAY                  DRY  NO DEFECTS
1                           NOT DIVIDED                  DRY  NO DEFECTS
2                              FOUR WAY                  DRY  NO DEFECTS
3       DIVIDED - W/MEDIAN (NOT RAISED)                  DRY  NO DEFECTS
4                              DRIVEWAY                  DRY  NO DEFECTS
...                                 ...                  ...         ...
481618                      NOT DIVIDED              UNKNOWN     UNKNOWN
481619                          ONE-WAY        SNOW OR SLUSH  NO DEFECTS
481620       DIVIDED - W/MEDIAN BARRIER                  WET  NO DEFECTS
481621                         FOUR WAY                  DRY  NO DEFECTS
481622                      NOT DIVIDED                  DRY  NO DEFECTS

                              CRASH_TYPE  ... INJURIES_FATAL  \
0                 NO INJURY / DRIVE AWAY  ...            0.0
1       INJURY AND / OR TOW DUE TO CRASH  ...            0.0
2                 NO INJURY / DRIVE AWAY  ...            0.0
3                 NO INJURY / DRIVE AWAY  ...            0.0
4                 NO INJURY / DRIVE AWAY  ...            0.0
...                                  ...  ...            ...
481618            NO INJURY / DRIVE AWAY  ...            0.0
481619            NO INJURY / DRIVE AWAY  ...            0.0
481620            NO INJURY / DRIVE AWAY  ...            0.0
481621            NO INJURY / DRIVE AWAY  ...            0.0
481622            NO INJURY / DRIVE AWAY  ...            0.0

       INJURIES_INCAPACITATING INJURIES_NON_INCAPACITATING  \
0                          0.0                         0.0
1                          0.0                         0.0
2                          0.0                         0.0
3                          0.0                         0.0
4                          0.0                         0.0
```

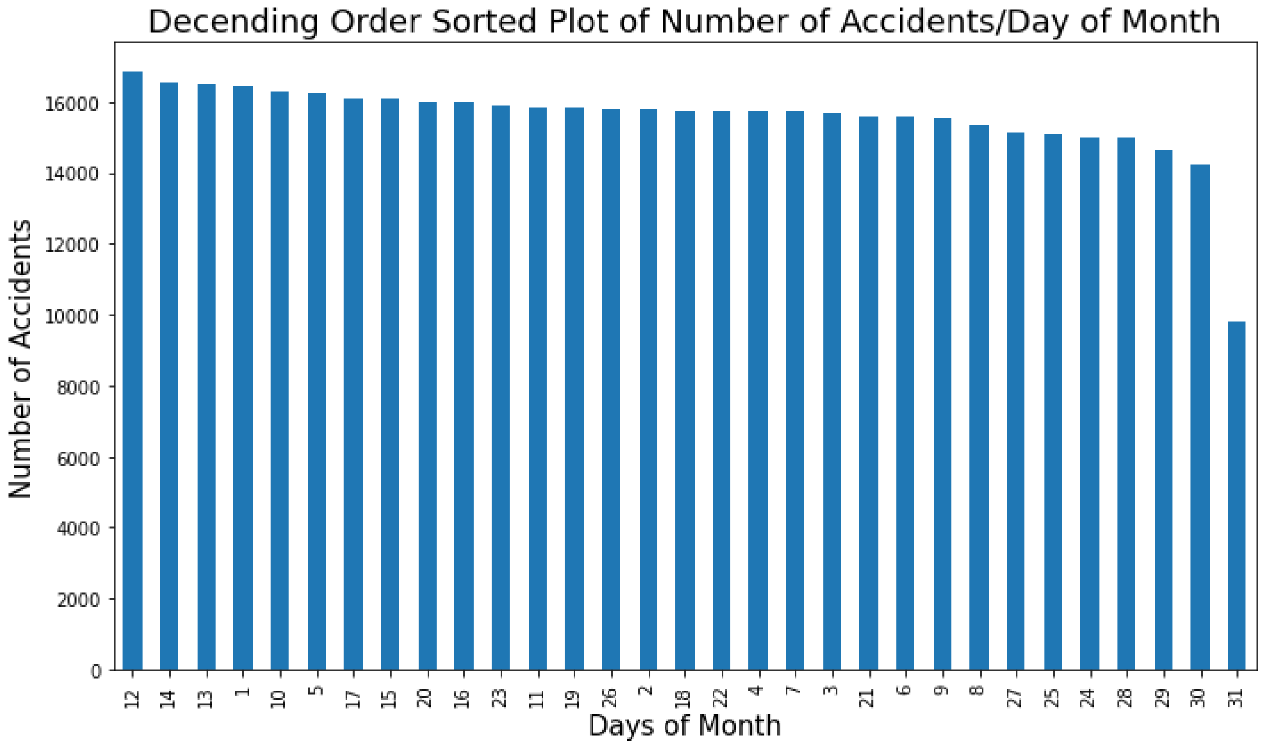
```
...                        ...                         ...
481618                     0.0                         0.0
481619                     0.0                         0.0
481620                     0.0                         0.0
481621                     0.0                         0.0
481622                     0.0                         0.0

       INJURIES_REPORTED_NOT_EVIDENT CRASH_HOUR CRASH_DAY_OF_WEEK  \
0                                0.0         17                 4
1                                0.0         16                 6
2                                0.0         10                 6
3                                0.0          1                 7
4                                0.0         14                 4
...                              ...        ...               ...
481618                           0.0          9                 2
481619                           0.0         21                 3
481620                           0.0         20                 4
481621                           0.0         17                 4
481622                           0.0         17                 4

        CRASH_MONTH  CRASH_DAY_OF_MONTH  CRASH_YEAR  CRASH_MINUTE
0                 7                  10          19            56
1                 6                  30          17            00
2                 7                  10          20            25
3                 7                  11          20            00
4                 7                   8          20            00
...             ...                 ...         ...           ...
481618            1                  18          21            00
481619            1                  19          21            23
481620            1                  20          21            20
481621            1                  20          21            00
481622            1                  20          21            50

[481623 rows x 28 columns]
```

3. some insights about the crashes and date/time.

In this section we are looking for which date most of the accident takes place and which day is it of the week that has most accidents. By below data we can see that on the 12th of the month and becoming specific about the day of week friday has the most number of accidents in the list.

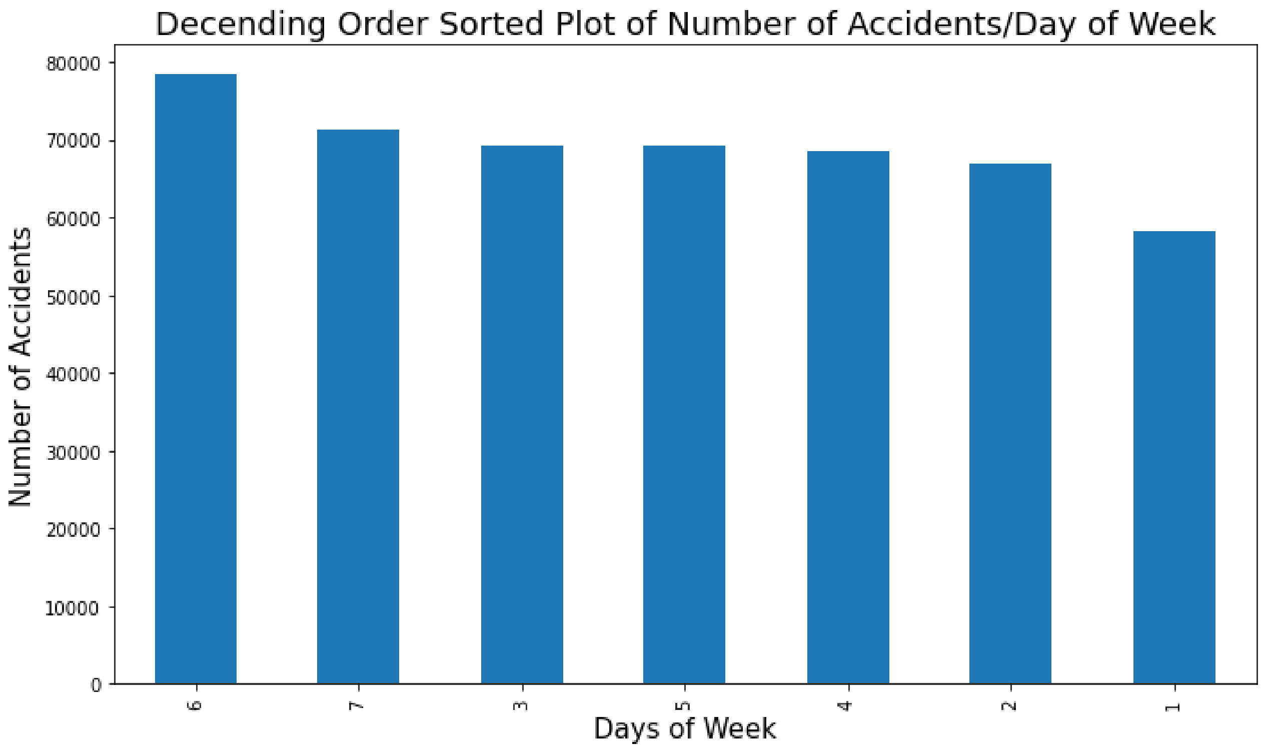
In [74]:

```
dataset1 = dataset
dataset1.sort_values("CRASH_DAY_OF_MONTH")
plt.figure(figsize=(10,6))
graph = dataset1['CRASH_DAY_OF_MONTH'].value_counts().plot.bar()
plt.xlabel("Days of Month", size=15)
plt.ylabel("Number of Accidents", size=15)
plt.title("Decending Order Sorted Plot of Number of Accidents/Day of Month", size=18)
plt.tight_layout()
```



In [75]:

```
dataset1.sort_values("CRASH_DAY_OF_WEEK")
plt.figure(figsize=(10,6))
graph = dataset1['CRASH_DAY_OF_WEEK'].value_counts().plot.bar()
plt.xlabel("Days of Week", size=15)
plt.ylabel("Number of Accidents", size=15)
plt.title("Decending Order Sorted Plot of Number of Accidents/Day of Week", size=18)
plt.tight_layout()
```



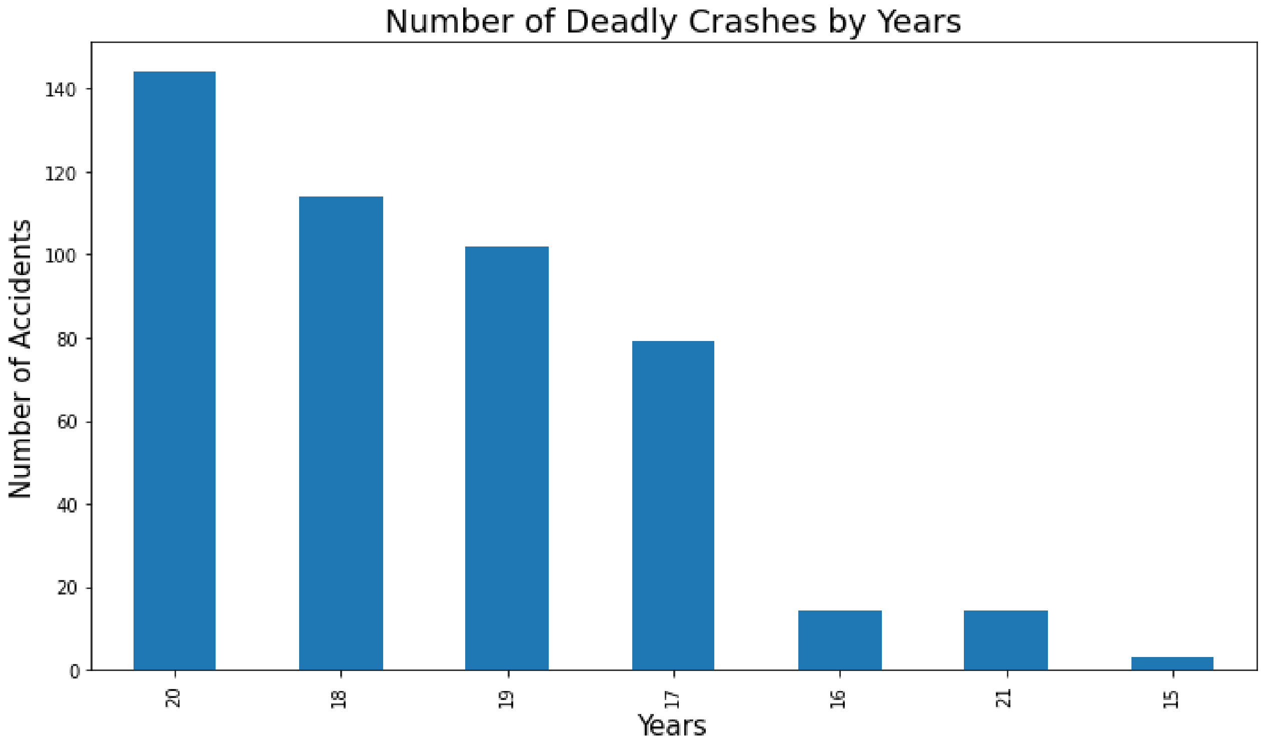
4. Number of deadly crashes in recent years.

In this we are trying to look at the deadly crashes that took place in recent years and by the data we can say that deadly crashes has increased by around 50% from year 2015 to 2020 and thats a significant amount of numbers increased in the crash.

In [76]:

```
dataset2 = dataset
dataset2 = dataset2.replace('NaN', np.nan)
dataset2 = dataset2.fillna(0)
print(dataset2['INJURIES_FATAL'].value_counts())
dataset2 = dataset2.sort_values("INJURIES_FATAL")
dataset3 = dataset2.iloc[481153: , :]
print(dataset3['INJURIES_FATAL'].value_counts())
print(dataset3['CRASH_YEAR'].value_counts())
plt.figure(figsize=(10,6))
graph = dataset3['CRASH_YEAR'].value_counts().plot.bar()
plt.xlabel("Years", size=15)
plt.ylabel("Number of Accidents", size=15)
plt.title("Number of Deadly Crashes by Years", size=18)
plt.tight_layout()
```

```
0.0    481153
1.0       437
2.0        27
3.0         5
4.0         1
Name: INJURIES_FATAL, dtype: int64
1.0    437
2.0     27
3.0      5
4.0      1
Name: INJURIES_FATAL, dtype: int64
20    144
18    114
19    102
17     79
16     14
21     14
15      3
Name: CRASH_YEAR, dtype: int64
```



5. Investigate number and type of injuries based on the speed limit

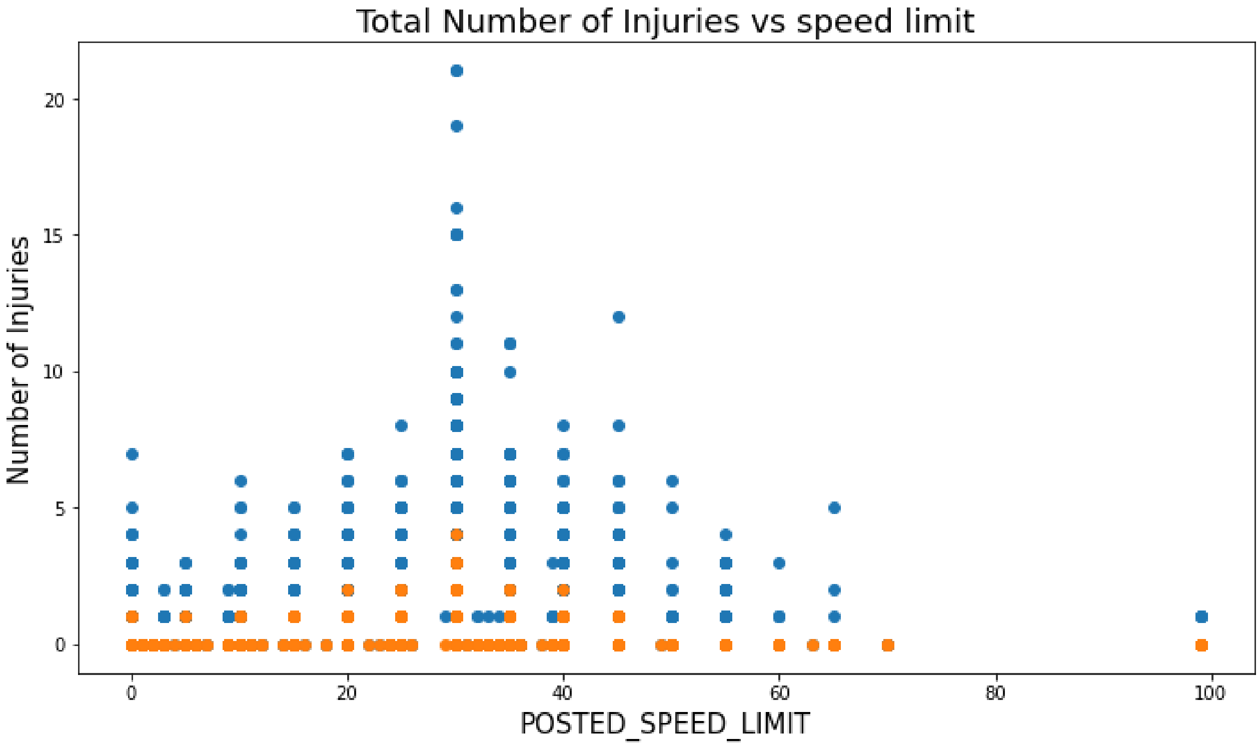
In this section we figured out number of injuries and different type of injuries took place in the comparison of the speed limit. Mainly we figured out that most of the injuries took place at speed limit of 30 and we can see that by data and also from value counts and it shows that. In first graph we are showing fatal injuries in comparison of speed limit. In, second graph we see injuries incapability and non incapability compared to speed limit.

In [80]:

```
dataset4 = dataset
dataset4 = dataset4.replace('NaN', np.nan)
dataset4 = dataset4.fillna(0)
print(dataset4['POSTED_SPEED_LIMIT'].value_counts())
plt.figure(figsize=(10,6))
plt.scatter(dataset4['POSTED_SPEED_LIMIT'], dataset4['INJURIES_TOTAL'])
plt.scatter(dataset4['POSTED_SPEED_LIMIT'], dataset4['INJURIES_FATAL'])
plt.xlabel("POSTED_SPEED_LIMIT", size=15)
plt.ylabel("Number of Injuries", size=15)
plt.title("Total Number of Injuries vs speed limit", size=18)
plt.tight_layout()
```

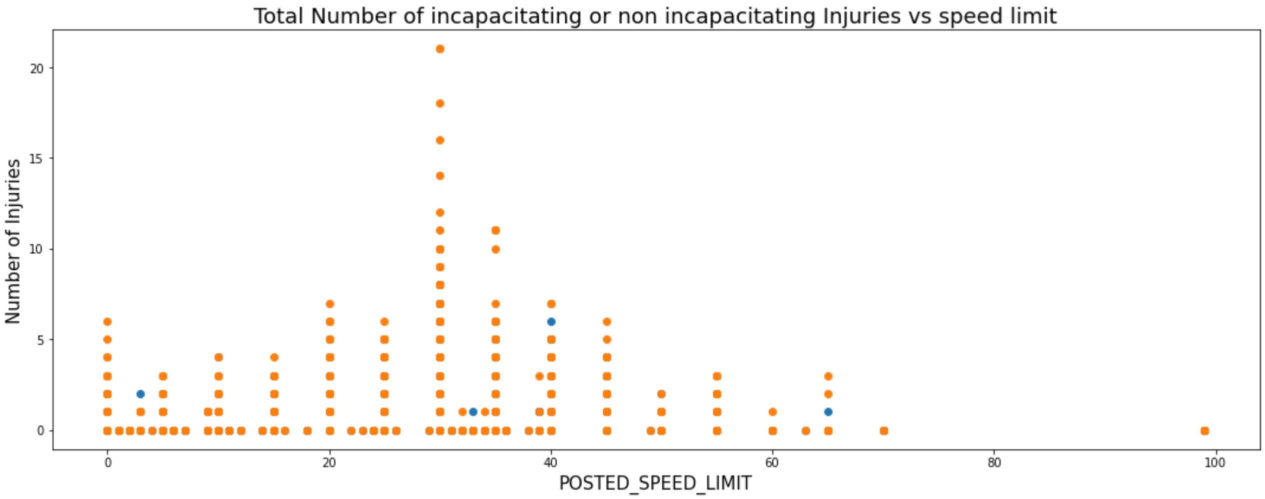
```
30    354381
35     33243
25     29334
20     18892
15     16820
10     10162
0       6766
40      4396
5       3694
45      2872
55       451
```

```
3       116
50      103
9        91
99       66
39       53
1        35
60       27
2        19
24       16
32       14
65       12
34       10
33       10
6         7
11        5
36        5
70        3
31        2
18        2
14        2
26        2
7         2
12        2
16        1
49        1
63        1
38        1
23        1
22        1
4         1
29        1
Name: POSTED_SPEED_LIMIT, dtype: int64
```



In [81]:

```
plt.figure(figsize=(15,6))
plt.scatter(dataset4['POSTED_SPEED_LIMIT'], dataset4['INJURIES_INCAPACITATING'])
plt.scatter(dataset4['POSTED_SPEED_LIMIT'], dataset4['INJURIES_NON_INCAPACITATING'])
plt.xlabel("POSTED_SPEED_LIMIT", size=15)
plt.ylabel("Number of Injuries", size=15)
plt.title("Total Number of incapacitating or non incapacitating Injuries vs speed limit
plt.tight_layout()
```

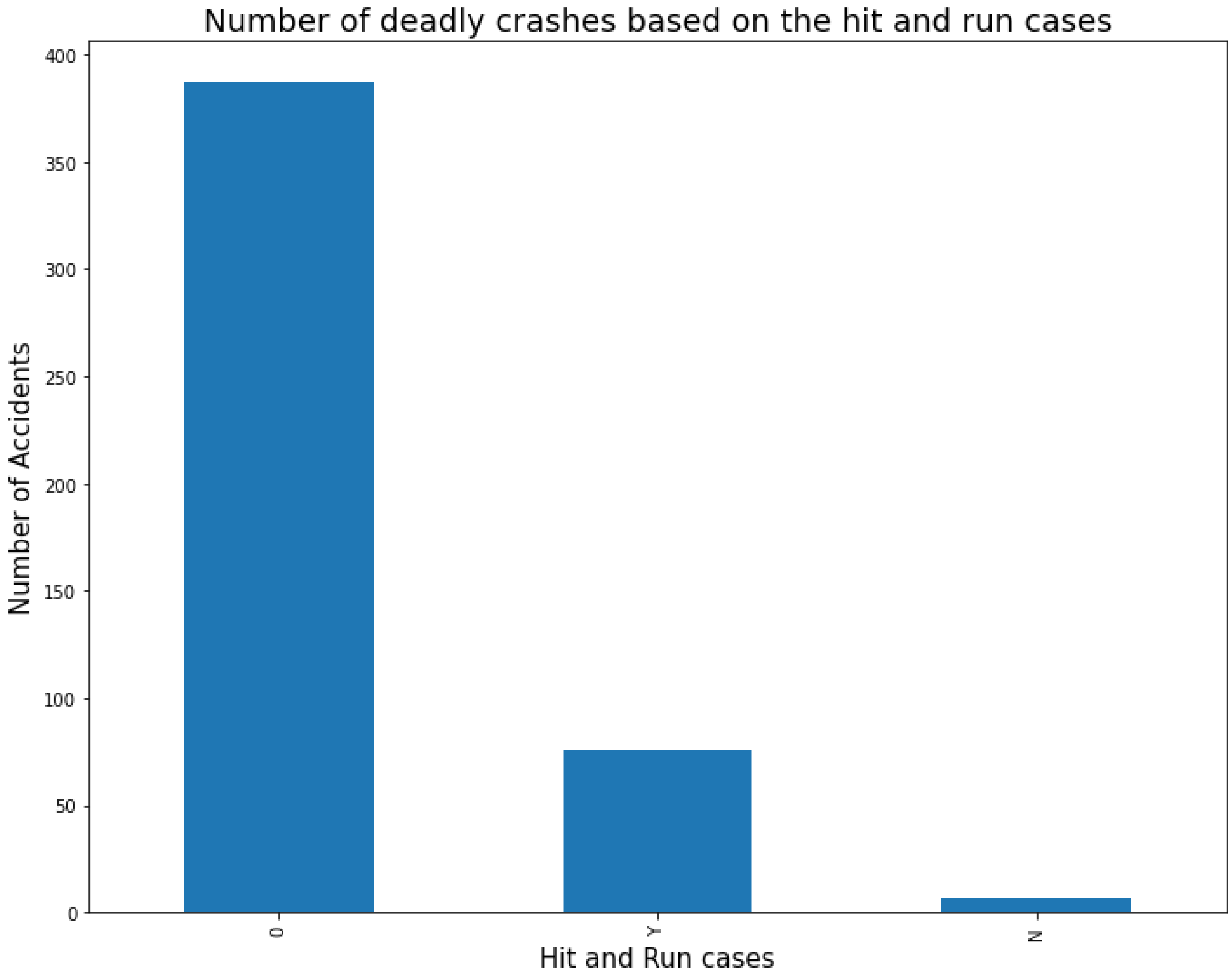


6. Is there a relationship between hit and run crashes and number of fatal injuries?

This is the data below showing how many hit and run cases have fatal injuries and the number is quiet high because it's around 100.

In [97]:

```
dataset6 = dataset
dataset6 = dataset6.replace('NaN', np.nan)
dataset6 = dataset6.fillna(0)
plt.figure(figsize=(10,8))
graph = dataset6.loc[dataset6['INJURIES_FATAL'] > 0, 'HIT_AND_RUN_I'].value_counts().pl
plt.xlabel("Hit and Run cases", size=15)
plt.ylabel("Number of Accidents", size=15)
plt.title("Number of deadly crashes based on the hit and run cases", size=18)
plt.tight_layout()
```

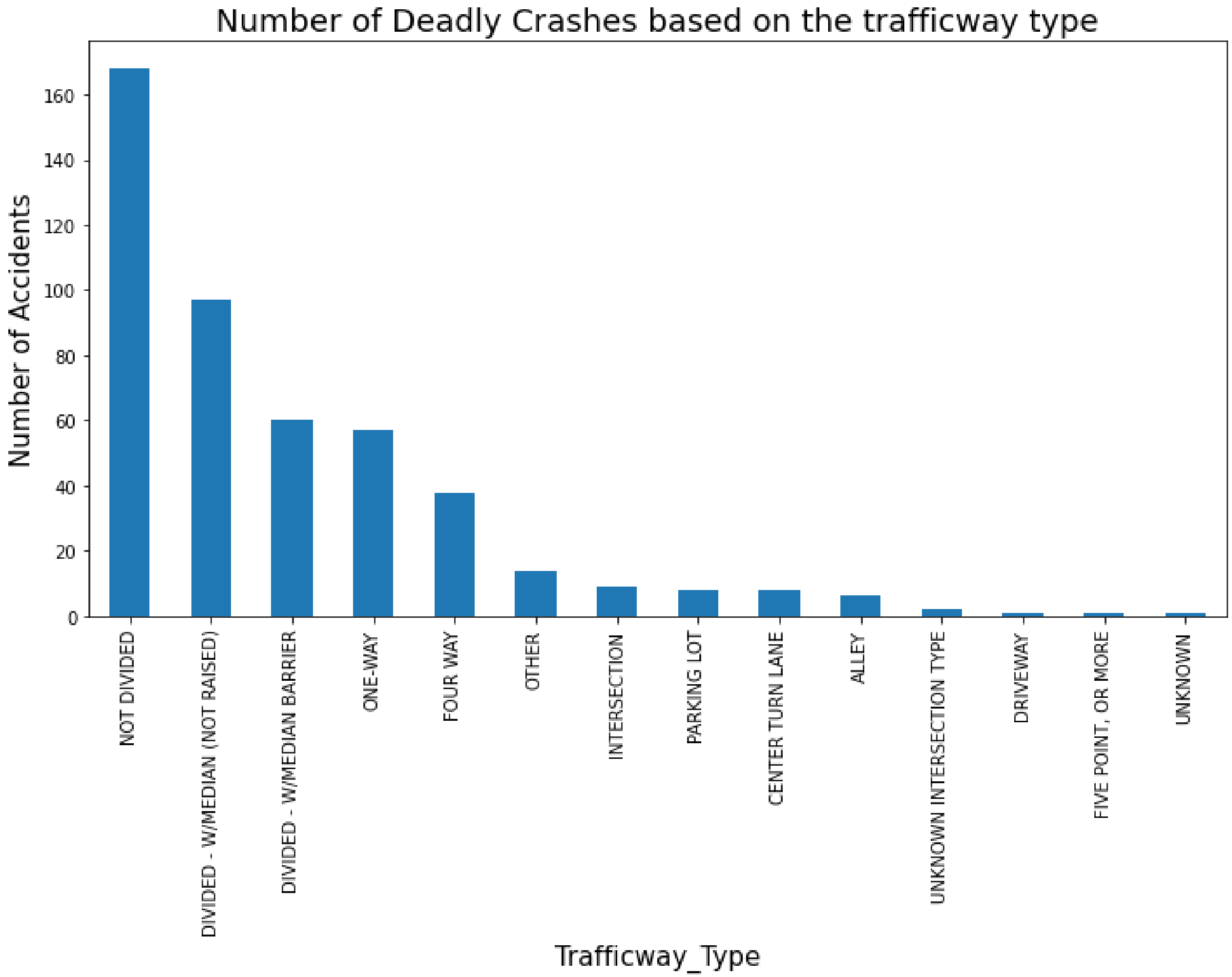


7. Do intersection-related crashes result in more fatal injuries?

The most fatal injuries is caused by the not divided traffic type but their is also fatal injuries in intersection but it's very low and it's less then 20.

In [84]:

```
dataset5 = dataset
dataset5['TRAFFICWAY_TYPE'] = dataset5['TRAFFICWAY_TYPE'].replace(['T-INTERSECTION','Y-
dataset5 = dataset5.replace('NaN', np.nan)
dataset5 = dataset5.fillna(0)
plt.figure(figsize=(10,8))
graph=dataset5.loc[dataset5['INJURIES_FATAL'] > 0, 'TRAFFICWAY_TYPE'].value_counts().pl
plt.xlabel("Trafficway_Type", size=15)
plt.ylabel("Number of Accidents", size=15)
plt.title("Number of Deadly Crashes based on the trafficway type", size=18)
plt.tight_layout()
```

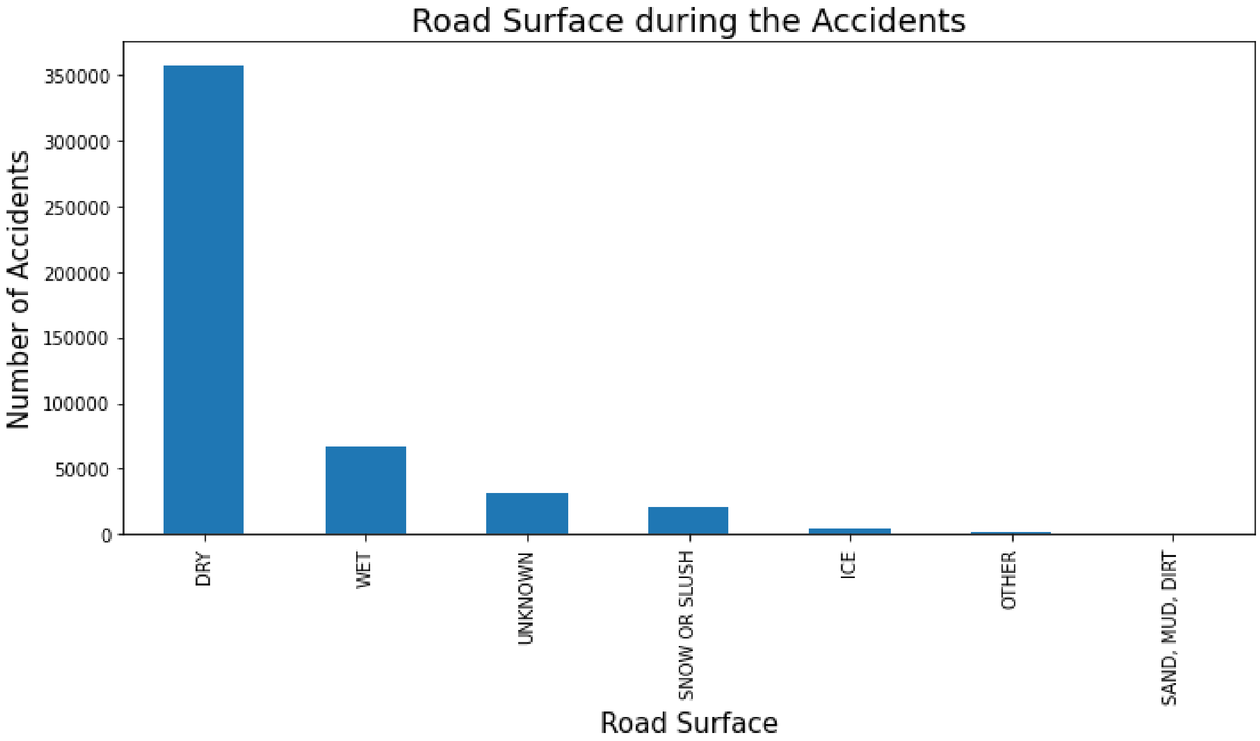


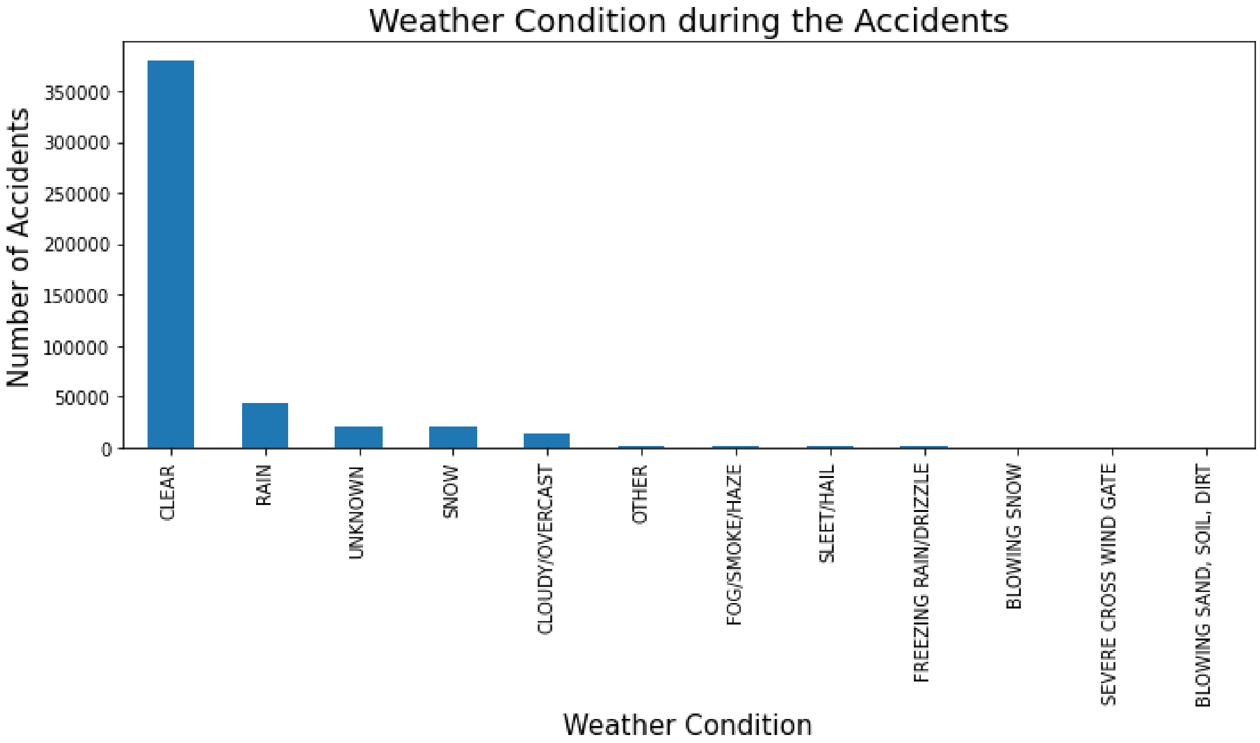
8. Come up with at least two more interesting insights and visualize them.

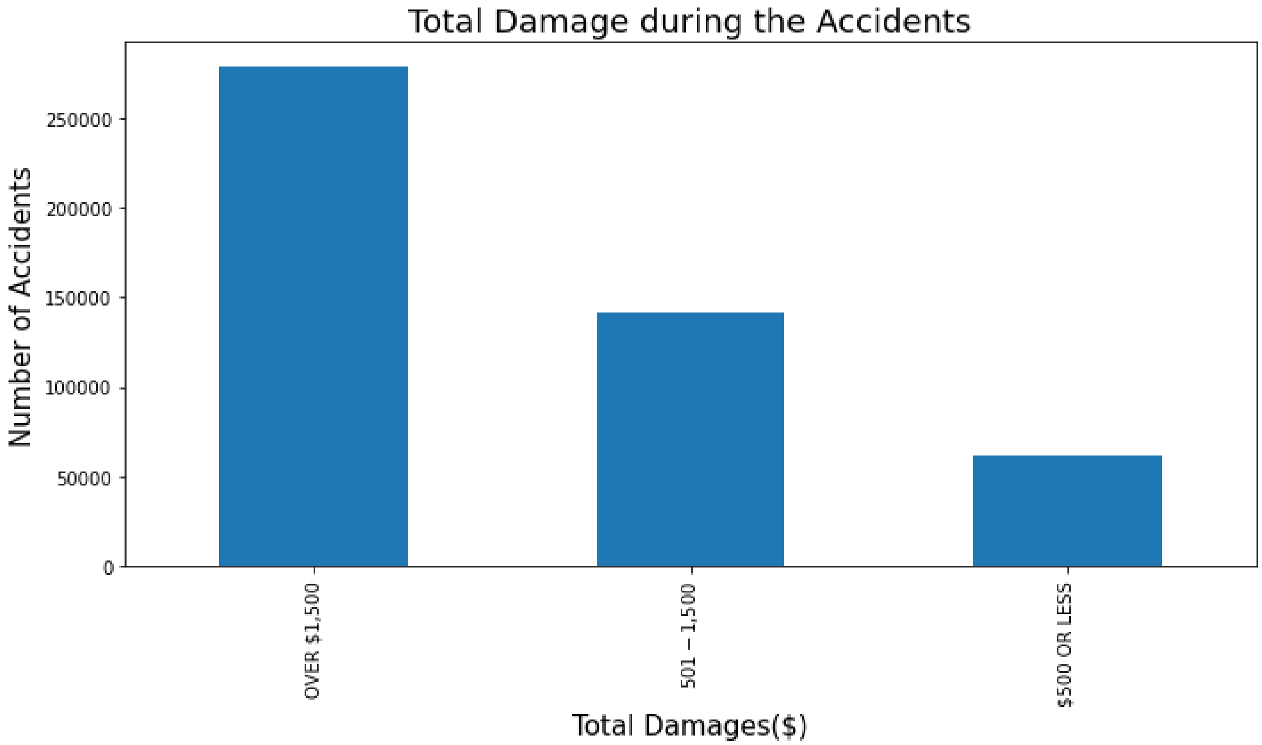
For the two insights facts, I plotted weather condition, road surfaces and total damages took place graph below to see some data.

In [98]:

```
dataset8 = dataset
dataset8.sort_values("ROADWAY_SURFACE_COND")
plt.figure(figsize=(10,6))
graph = dataset8['ROADWAY_SURFACE_COND'].value_counts().plot.bar()
plt.xlabel("Road Surface", size=15)
plt.ylabel("Number of Accidents", size=15)
plt.title("Road Surface during the Accidents", size=18)
plt.tight_layout()
dataset8.sort_values("WEATHER_CONDITION")
plt.figure(figsize=(10,6))
graph = dataset8['WEATHER_CONDITION'].value_counts().plot.bar()
plt.xlabel("Weather Condition", size=15)
plt.ylabel("Number of Accidents", size=15)
plt.title("Weather Condition during the Accidents", size=18)
plt.tight_layout()
dataset8.sort_values("DAMAGE")
plt.figure(figsize=(10,6))
graph = dataset8['DAMAGE'].value_counts().plot.bar()
plt.xlabel("Total Damages($)", size=15)
plt.ylabel("Number of Accidents", size=15)
plt.title("Total Damage during the Accidents", size=18)
plt.tight_layout()
```






Total Damage during the Accidents
Number of Accidents
250000
200000
150000
100000
50000
0
OVER $1,500
501 – 1,500
$500 OR LESS
Total Damages($)

In [ ]: