Road Accidents

Road safety continues to be a major developmental issue, a public health concern and a leading cause of death and injury across the world. At least one out of 10 people killed on roads across the world is from India, according to the World Health Organization. The cost of road accidents is borne not only by the victims and their family, but by the economy as a whole in terms of untimely deaths, injuries, disabilities and loss of potential income. It is indeed a matter of great concern that despite the continuing efforts of the Government in this regard and our commitments for halving fatalities we have not been able to register significant progress on this front. During the year 2021, a total number of 4,12,432 road accidents have been reported in the country, claiming 1,53,972 lives and causing injuries to 3,84,448 persons. Unfortunately, the worst affected age group in Road accidents is 18-45 years, which accounts for about 67 percent of total accidental deaths.

In [62]:
```
import pandas as pd
import numpy as np
import seaborn as sns #its a visualiztion tools (library)
import matplotlib.pyplot as plt #its a library
import os
%matplotlib inline
```

In [2]:
```
#here we are checking the current working directory
os.getcwd()
```

Out[2]:
```
'C:\\Users\\satee'
```

In [3]:
```
#now we are changing the directory and want to read and see the csv file
os.chdir('C:\Desktop\Data Analyst Project\Road Accident')
```

In [4]:
```
#current working directory
os.getcwd()
```

Out[4]:
```
'C:\\Desktop\\Data Analyst Project\\Road Accident'
```

import data

In [5]:
```
#reading the file from directry
df=pd.read_csv(r"C:\Desktop\Data Analyst Project\Road Accident\Road Accident Data.csv")
```

we have successfully imported the dataset and now we can view by using descriibe method

```
In [6]: df.describe
```

```
Out[6]: <bound method NDFrame.describe of        Accident_Index  Date          Month  Year Accident Date Day_of_Week  \
0       200901BS70001     1    January 2021  2021     01-Jan-21    Thursday
1       200901BS70002     5    January 2021  2021     05-Jan-21      Monday
2       200901BS70003     4    January 2021  2021     04-Jan-21      Sunday
3       200901BS70004     5    January 2021  2021     05-Jan-21      Monday
4       200901BS70005     6    January 2021  2021     06-Jan-21     Tuesday
...               ...   ...           ...   ...           ...         ...
307790  201091NM01760    18   February 2022  2022     18-Feb-22    Thursday
307791  201091NM01881    21   February 2022  2022     21-Feb-22      Sunday
307792  201091NM01935    23   February 2022  2022     23-Feb-22     Tuesday
307793  201091NM01964    23   February 2022  2022     23-Feb-22     Tuesday
307794  201091NM02142    28   February 2022  2022     28-Feb-22      Sunday

                Junction_Control                      Junction_Detail  \
0           Give way or uncontrolled            T or staggered junction
1           Give way or uncontrolled                         Crossroads
2           Give way or uncontrolled            T or staggered junction
3                Auto traffic signal            T or staggered junction
4                Auto traffic signal                         Crossroads
...                              ...                                ...
307790  Data missing or out of range  Not at junction or within 20 metres
307791  Data missing or out of range  Not at junction or within 20 metres
307792      Give way or uncontrolled            T or staggered junction
307793      Give way or uncontrolled            T or staggered junction
307794      Give way or uncontrolled            T or staggered junction

       Accident_Severity   Latitude  ... Casualities Number_of_Vehicles  \
0                Serious  51.512273  ...           1                  2
1                Serious  51.514399  ...          11                  2
2                 Slight  51.486668  ...           1                  2
3                Serious  51.507804  ...           1                  2
4                Serious  51.482076  ...           1                  2
...                  ...        ...  ...         ...                ...
307790            Slight  57.374005  ...           2                  1
307791            Slight  57.232273  ...           1                  1
307792            Slight  57.585044  ...           1                  3
307793           Serious  57.214898  ...           1                  2
307794           Serious  57.575210  ...           1                  1

               Police_Force  Road_Surface_Conditions           Road_Type  \
0       Metropolitan Police                      Dry     One way street
1       Metropolitan Police               Wet or damp  Single carriageway
2       Metropolitan Police                      Dry  Single carriageway
3       Metropolitan Police             Frost or ice  Single carriageway
4       Metropolitan Police                      Dry  Single carriageway
...                     ...                      ...                 ...
307790             Northern                      Dry  Single carriageway
307791             Northern             Frost or ice  Single carriageway
307792             Northern             Frost or ice  Single carriageway
307793             Northern              Wet or damp  Single carriageway
307794             Northern              Wet or damp    Dual carriageway

        Speed_limit   Time   Area     Weather_Conditions  \
0                30  15:11  Urban     Fine no high winds
1                30  10:59  Urban     Fine no high winds
2                30  14:19  Urban     Fine no high winds
3                30  08:10  Urban                  Other
4                30  17:25  Urban     Fine no high winds
...             ...    ...    ...                    ...
307790           60  07:00  Rural     Fine no high winds
307791           60  03:00  Rural     Fine no high winds
307792           30  09:38  Rural     Fine no high winds
307793           60  18:25  Rural     Fine no high winds
307794           60  15:45  Rural  Snowing no high winds

                 Vehicle_Type
0                         Car
1       Taxi/Private hire car
2       Taxi/Private hire car
3       Motorcycle over 500cc
4                         Car
...                       ...
307790                    Car
307791                    Car
307792                    Car
307793  Motorcycle over 500cc
307794                    Car

[307795 rows x 24 columns]>
```

To view all the columns data type, and those null values use info method

In [7]:
```
df.info() # to view all the columns data types and check if there null values
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 307795 entries, 0 to 307794
Data columns (total 24 columns):
 #   Column                      Non-Null Count   Dtype  
---  ------                      --------------   -----  
 0   Accident_Index              307795 non-null  object 
 1   Date                        307795 non-null  int64  
 2   Month                       307795 non-null  object 
 3   Year                        307795 non-null  int64  
 4   Accident Date               307795 non-null  object 
 5   Day_of_Week                 307795 non-null  object 
 6   Junction_Control            307795 non-null  object 
 7   Junction_Detail             307795 non-null  object 
 8   Accident_Severity           307795 non-null  object 
 9   Latitude                    307795 non-null  float64
 10  Light_Conditions            307795 non-null  object 
 11  Local_Authority_(District)  307795 non-null  object 
 12  Carriageway_Hazards         307792 non-null  object 
 13  Longitude                   307795 non-null  float64
 14  Casualities                 307795 non-null  int64  
 15  Number_of_Vehicles          307795 non-null  int64  
 16  Police_Force                307795 non-null  object 
 17  Road_Surface_Conditions     307482 non-null  object 
 18  Road_Type                   306439 non-null  object 
 19  Speed_limit                 307795 non-null  int64  
 20  Time                        307778 non-null  object 
 21  Area                        307795 non-null  object 
 22  Weather_Conditions          301916 non-null  object 
 23  Vehicle_Type                307795 non-null  object 
dtypes: float64(2), int64(5), object(17)
memory usage: 56.4+ MB
```

now to see the total number of rows and columns use shape method

In [8]:
```
df.shape #it show in the rows and columns format
```

Out[8]:
```
(307795, 24)
```

```
here we can see the total rows and columns
```

In [9]:
```
#to see all the columns of present data frame
df.columns
```

Out[9]:
```
Index(['Accident_Index', 'Date', 'Month', 'Year', 'Accident Date',
       'Day_of_Week', 'Junction_Control', 'Junction_Detail',
       'Accident_Severity', 'Latitude', 'Light_Conditions',
       'Local_Authority_(District)', 'Carriageway_Hazards', 'Longitude',
       'Casualities', 'Number_of_Vehicles', 'Police_Force',
       'Road_Surface_Conditions', 'Road_Type', 'Speed_limit', 'Time', 'Area',
       'Weather_Conditions', 'Vehicle_Type'],
      dtype='object')
```

Data Clean-up (Missing value Treatment)

Drop all the null values from the all columns

In [10]:
```
#count the number of missing value or null values
df.isnull().sum()
```

Out[10]:
```
Accident_Index                 0
Date                           0
Month                          0
Year                           0
Accident Date                  0
Day_of_Week                    0
Junction_Control               0
Junction_Detail                0
Accident_Severity              0
Latitude                       0
Light_Conditions               0
Local_Authority_(District)     0
Carriageway_Hazards            3
Longitude                      0
Casualities                    0
Number_of_Vehicles             0
Police_Force                   0
Road_Surface_Conditions      313
Road_Type                   1356
Speed_limit                    0
Time                          17
Area                           0
Weather_Conditions          5879
Vehicle_Type                   0
dtype: int64
```

```
as we can see there is null values in few columns ,here we used sum method to sum of the null values
```

Now drop All the null values from dataset to make free from unusual data to database

In [11]:
```
#remove all the null values by using dropna mothod
df.dropna(inplace=True)
```

```
#here we have successfully removed all the null values from the dataset
and the inpalce=True means it permanently removes the null values from columns
```

In [12]:
```
#here recheck the null values
df.isnull().sum()
```

Out[12]:
```
Accident_Index                0
Date                          0
Month                         0
Year                          0
Accident Date                 0
Day_of_Week                   0
Junction_Control              0
Junction_Detail               0
Accident_Severity             0
Latitude                      0
Light_Conditions              0
Local_Authority_(District)    0
Carriageway_Hazards           0
Longitude                     0
Casualities                   0
Number_of_Vehicles            0
Police_Force                  0
Road_Surface_Conditions       0
Road_Type                     0
Speed_limit                   0
Time                          0
Area                          0
Weather_Conditions            0
Vehicle_Type                  0
dtype: int64
```

now we can see that no null values are ,so now we can move move forward for the next process in data analysis

In [13]:
```
#now recheck data,is it removed or not
df.shape
```

Out[13]:
```
(300492, 24)
```

now we can see that the number of rows had decreased

show table data

```
In [15]: df.head(5)
```

Out[15]:

	Accident_Index	Date	Month	Year	Accident Date	Day_of_Week	Junction_Control	Junction_Detail	Accident_Severity	Latitude	...	Casualities	Number_of_V
0	200901BS70001	1	January 2021	2021	01-Jan-21	Thursday	Give way or uncontrolled	T or staggered junction	Serious	51.512273	...	1	
1	200901BS70002	5	January 2021	2021	05-Jan-21	Monday	Give way or uncontrolled	Crossroads	Serious	51.514399	...	11	
2	200901BS70003	4	January 2021	2021	04-Jan-21	Sunday	Give way or uncontrolled	T or staggered junction	Slight	51.486668	...	1	
3	200901BS70004	5	January 2021	2021	05-Jan-21	Monday	Auto traffic signal	T or staggered junction	Serious	51.507804	...	1	
4	200901BS70005	6	January 2021	2021	06-Jan-21	Tuesday	Auto traffic signal	Crossroads	Serious	51.482076	...	1	

5 rows × 24 columns

Data clean up correcting the data type

check all the variables that need to be change

```
In [16]: df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 300492 entries, 0 to 307794
Data columns (total 24 columns):
 #   Column                      Non-Null Count   Dtype  
---  ------                      --------------   -----  
 0   Accident_Index              300492 non-null  object 
 1   Date                        300492 non-null  int64  
 2   Month                       300492 non-null  object 
 3   Year                        300492 non-null  int64  
 4   Accident Date               300492 non-null  object 
 5   Day_of_Week                 300492 non-null  object 
 6   Junction_Control            300492 non-null  object 
 7   Junction_Detail             300492 non-null  object 
 8   Accident_Severity           300492 non-null  object 
 9   Latitude                    300492 non-null  float64
 10  Light_Conditions            300492 non-null  object 
 11  Local_Authority_(District)  300492 non-null  object 
 12  Carriageway_Hazards         300492 non-null  object 
 13  Longitude                   300492 non-null  float64
 14  Casualities                 300492 non-null  int64  
 15  Number_of_Vehicles          300492 non-null  int64  
 16  Police_Force                300492 non-null  object 
 17  Road_Surface_Conditions     300492 non-null  object 
 18  Road_Type                   300492 non-null  object 
 19  Speed_limit                 300492 non-null  int64  
 20  Time                        300492 non-null  object 
 21  Area                        300492 non-null  object 
 22  Weather_Conditions          300492 non-null  object 
 23  Vehicle_Type                300492 non-null  object 
dtypes: float64(2), int64(5), object(17)
memory usage: 57.3+ MB
```

```
    From this information, we can see that , Accident_Index,Month,Time ,Area  are in numeric type but still some of the col
    umns need to be change are Date , Year,casualities,Number Of Vehicles ,Speed limit
```

```
In [17]: #here we are changing Date into integer type
         df['Date']=df['Date'].astype('int')
```

```
In [18]: # here we are changing Year into integer type
         df['Year']=df['Year'].astype('int')
```

```
In [19]: #here we are changing Casualities into type integer
         df['Casualities']=df['Casualities'].astype('int')
```

```
In [20]: # here we are changing Number_of_vehicles  float type into integer type
         df['Number_of_Vehicles'] =df['Number_of_Vehicles'].astype('int')
```

```
In [21]: # here we are changing Speed limit float type into integer type
         df['Speed_limit']=df['Speed_limit'].astype('int')
```

Here we successfully changed date,Year ,Casuaities,,Number_Of_Vehicles, and Speed linmit (float type) in to integer data type

In [22]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 300492 entries, 0 to 307794
Data columns (total 24 columns):
 #   Column                   Non-Null Count   Dtype  
---  ------                   --------------   -----  
 0   Accident_Index           300492 non-null  object 
 1   Date                     300492 non-null  int32  
 2   Month                    300492 non-null  object 
 3   Year                     300492 non-null  int32  
 4   Accident Date            300492 non-null  object 
 5   Day_of_Week              300492 non-null  object 
 6   Junction_Control         300492 non-null  object 
 7   Junction_Detail          300492 non-null  object 
 8   Accident_Severity        300492 non-null  object 
 9   Latitude                 300492 non-null  float64
 10  Light_Conditions         300492 non-null  object 
 11  Local_Authority_(District)  300492 non-null  object 
 12  Carriageway_Hazards      300492 non-null  object 
 13  Longitude                300492 non-null  float64
 14  Casualities              300492 non-null  int32  
 15  Number_of_Vehicles       300492 non-null  int32  
 16  Police_Force             300492 non-null  object 
 17  Road_Surface_Conditions  300492 non-null  object 
 18  Road_Type                300492 non-null  object 
 19  Speed_limit              300492 non-null  int32  
 20  Time                     300492 non-null  object 
 21  Area                     300492 non-null  object 
 22  Weather_Conditions       300492 non-null  object 
 23  Vehicle_Type             300492 non-null  object 
dtypes: float64(2), int32(5), object(17)
memory usage: 51.6+ MB
```

now we can see , we have converted identifies columns into integer datatype

In [23]:
```
df.head()
```

Out[23]:

	Accident_Index	Date	Month	Year	Accident Date	Day_of_Week	Junction_Control	Junction_Detail	Accident_Severity	Latitude	...	Casualities	Number_of_V
0	200901BS70001	1	January 2021	2021	01-Jan-21	Thursday	Give way or uncontrolled	T or staggered junction	Serious	51.512273	...	1	
1	200901BS70002	5	January 2021	2021	05-Jan-21	Monday	Give way or uncontrolled	Crossroads	Serious	51.514399	...	11	
2	200901BS70003	4	January 2021	2021	04-Jan-21	Sunday	Give way or uncontrolled	T or staggered junction	Slight	51.486668	...	1	
3	200901BS70004	5	January 2021	2021	05-Jan-21	Monday	Auto traffic signal	T or staggered junction	Serious	51.507804	...	1	
4	200901BS70005	6	January 2021	2021	06-Jan-21	Tuesday	Auto traffic signal	Crossroads	Serious	51.482076	...	1	

5 rows × 24 columns

In [24]:
```
df['Area'].value_counts()
```

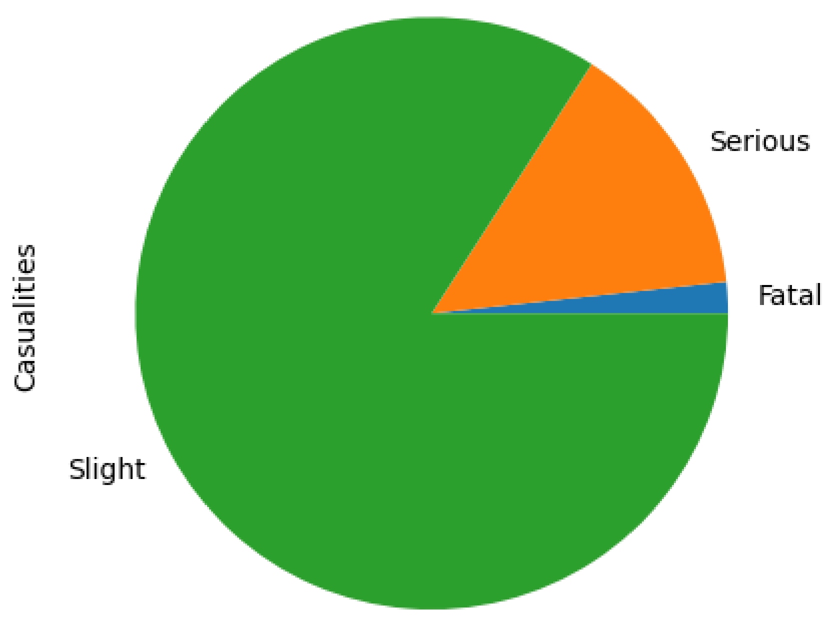
Out[24]:
```
Urban    193340
Rural    107152
Name: Area, dtype: int64
```

```
here we can see the number of accident cases in urban area is more than as compared to Rural area
```

Perform Basic EDA

Whats are the sum of casualities in term of accident severity effect

```
In [11]: df.groupby(['Accident_Severity'])['Casualities'].sum().plot(kind='pie')
         plt.show()
```



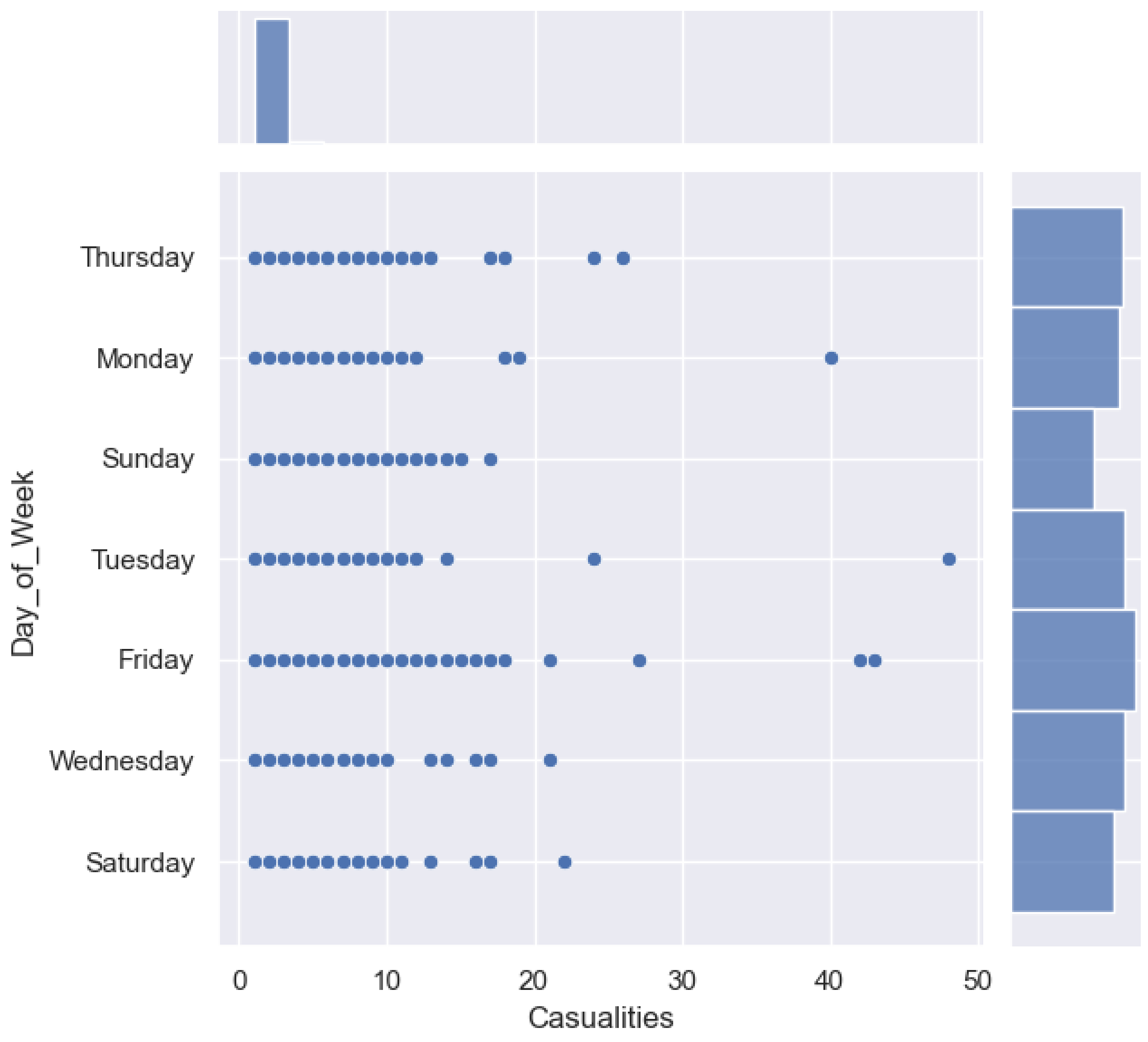
```
here we can see the number of slight cases are 351235,serious cases are 59298 and fatal cases are 7135 so now we can say most
of the cases are slight and the total of fatal and serious cases very less as comapred to slight
```

show the trend week days with the casualities how does it changes and affect.

```
In [12]: df.groupby(['Day_of_Week'])['Casualities'].sum()
```

```
Out[12]: Day_of_Week
         Friday       68269
         Monday       58443
         Saturday     59061
         Sunday       48839
         Thursday     60510
         Tuesday      61245
         Wednesday    61301
         Name: Casualities, dtype: int64
```

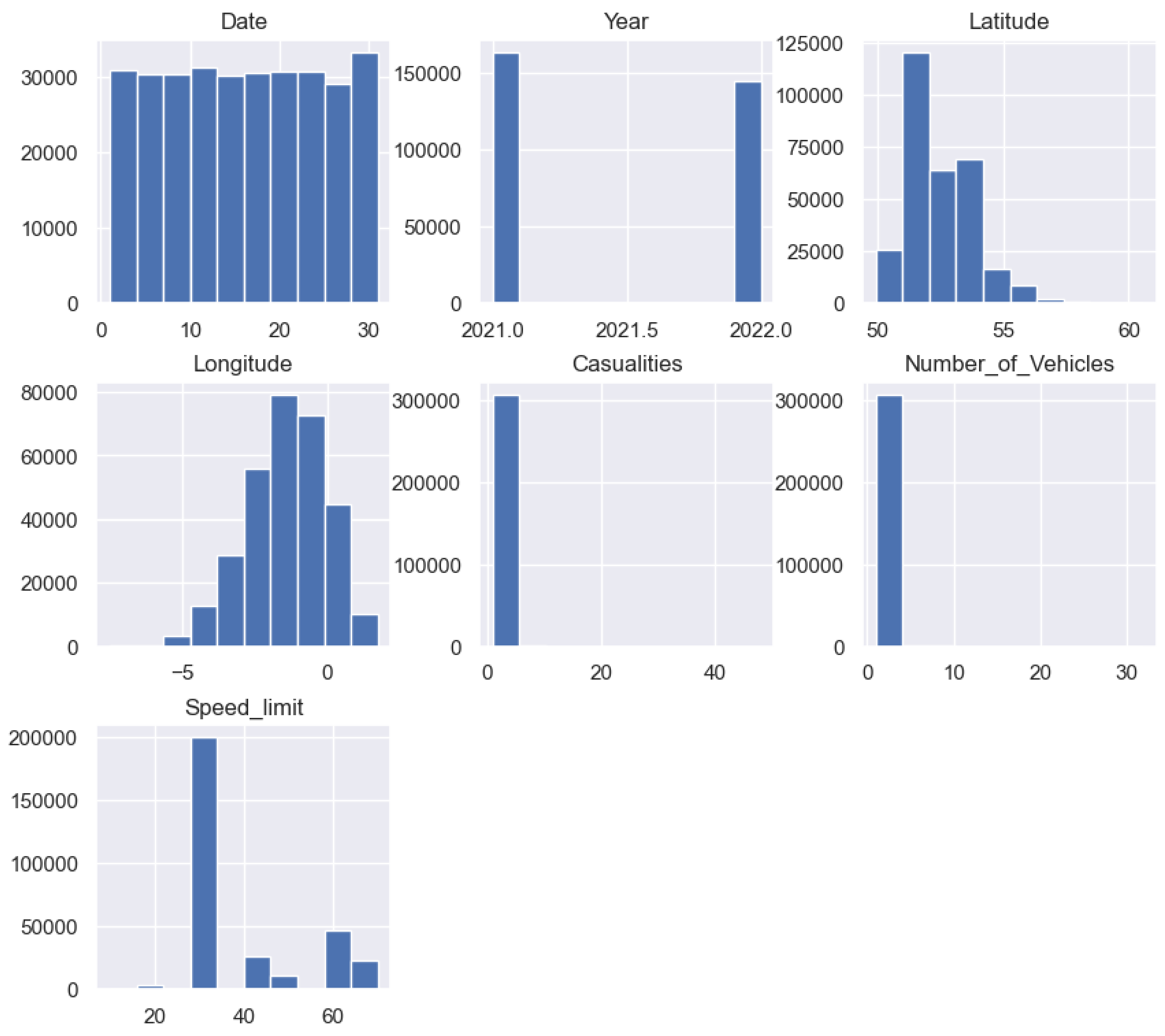
```
In [23]: sns.jointplot(y='Day_of_Week',x='Casualities',data=df)
         plt.show()
```



here we can see the most number of cases are at friday and the least on os sunday it might be because ,most of the person prefer to live at home

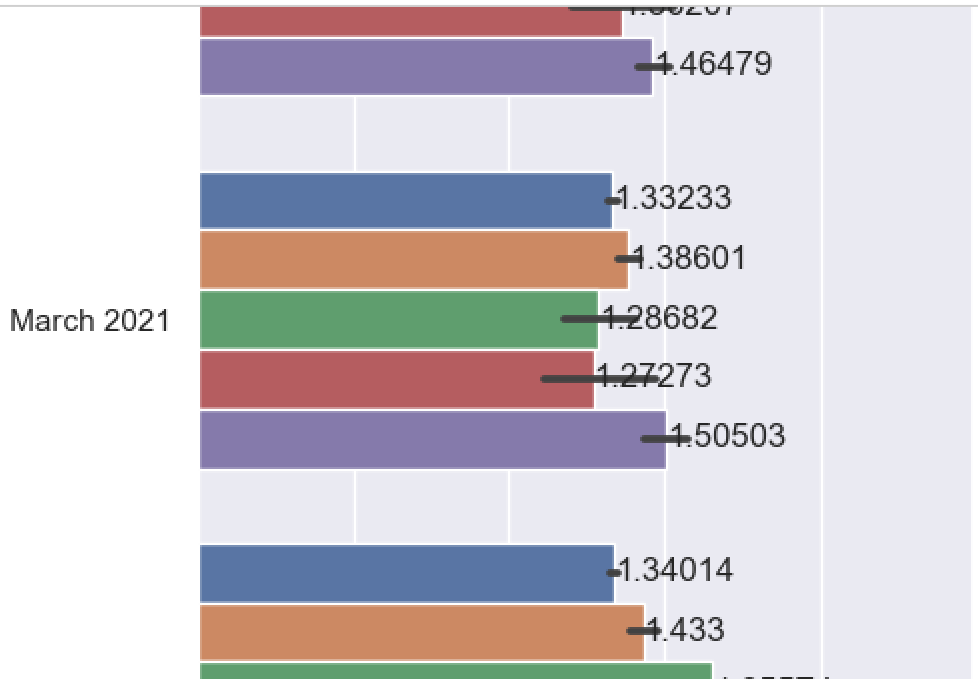
make histogram to show the data for a particular interval of time like date, year,longitude,etc

In [22]:
```
df.hist()
sns.set(rc={'figure.figsize':(10,9)})
plt.show()
```



Now Lets see how the casualities with Light Condition and Months.

In [18]:
```
ax=sns.barplot(data=df,y='Month',x='Casualities',hue='Light_Conditions')
sns.set(rc={'figure.figsize':(5,68)})
for bars in ax.containers:
    ax.bar_label(bars)
    plt.xticks(rotation=80)
    plt.show()
```

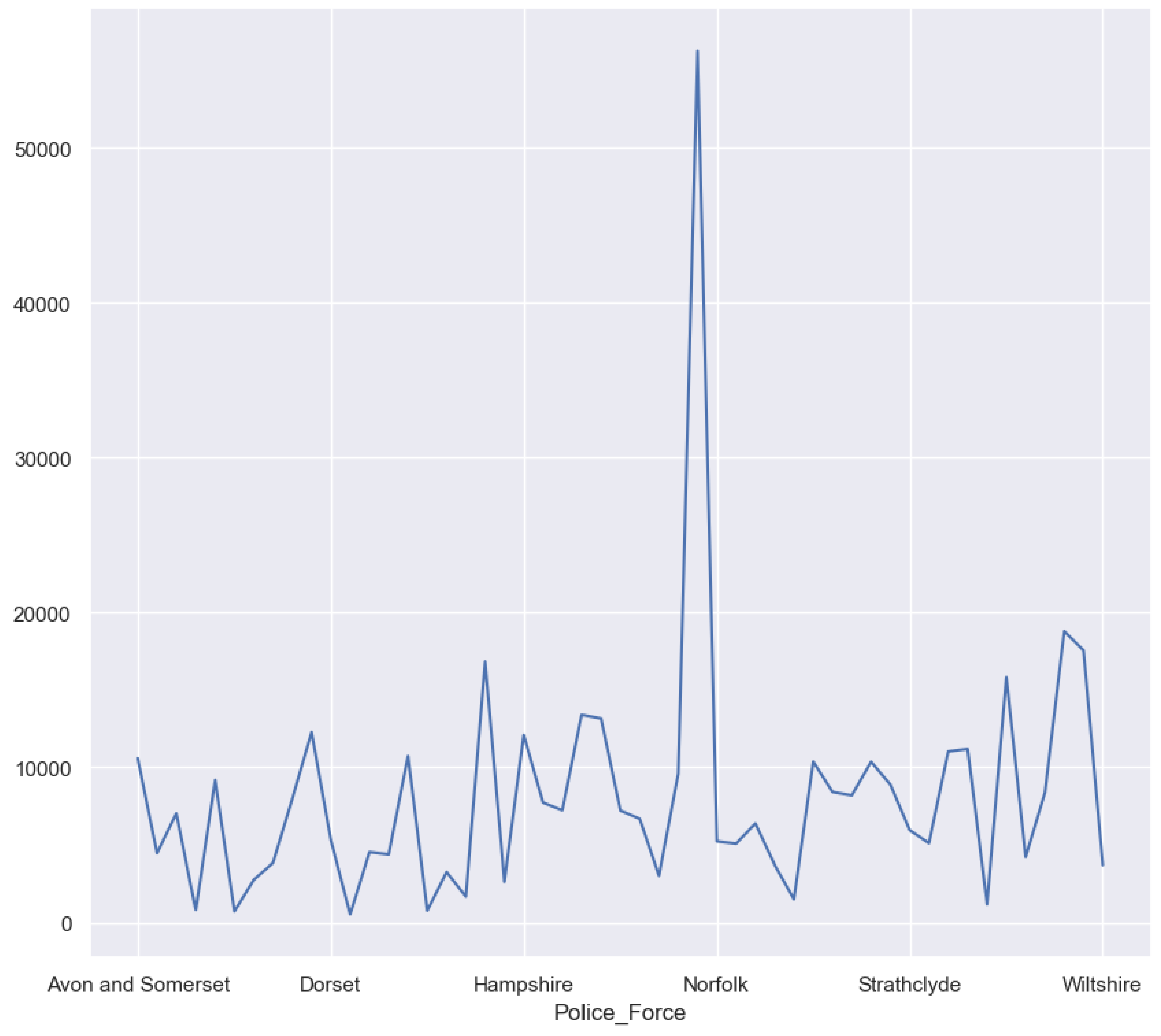


The Number of casualities with police force

In [24]:
```
df.groupby(['Police_Force'])['Casualities'].sum()
```

Out[24]:
```
Police_Force
Avon and Somerset        10585
Bedfordshire              4457
Cambridgeshire            7032
Central                    815
Cheshire                  9184
City of London             723
Cleveland                 2754
Cumbria                   3848
Derbyshire                7976
Devon and Cornwall       12277
Dorset                    5302
Dumfries and Galloway      533
Durham                    4529
Dyfed-Powys               4382
Essex                    10738
Fife                       765
Gloucestershire           3259
Grampian                  1673
Greater Manchester       16837
Gwent                     2632
Hampshire                12099
Hertfordshire             7725
Humberside                7229
Kent                     13395
Lancashire               13157
Leicestershire            7209
Lincolnshire              6678
Lothian and Borders       3009
Merseyside                9606
Metropolitan Police      56235
Norfolk                   5226
North Wales               5078
North Yorkshire           6376
Northamptonshire          3709
Northern                  1500
Northumbria              10374
Nottinghamshire           8411
South Wales               8193
South Yorkshire          10362
Staffordshire             8889
Strathclyde               5949
Suffolk                   5106
Surrey                   11029
Sussex                   11186
Tayside                   1172
Thames Valley            15831
Warwickshire              4209
West Mercia               8369
West Midlands            18800
West Yorkshire           17558
Wiltshire                 3698
Name: Casualities, dtype: int64
```

```
In [25]: df.groupby(['Police_Force'])['Casualities'].sum().plot(kind='line')
         sns.set(rc={'figure.figsize':(8,4)})
         plt.show()
```



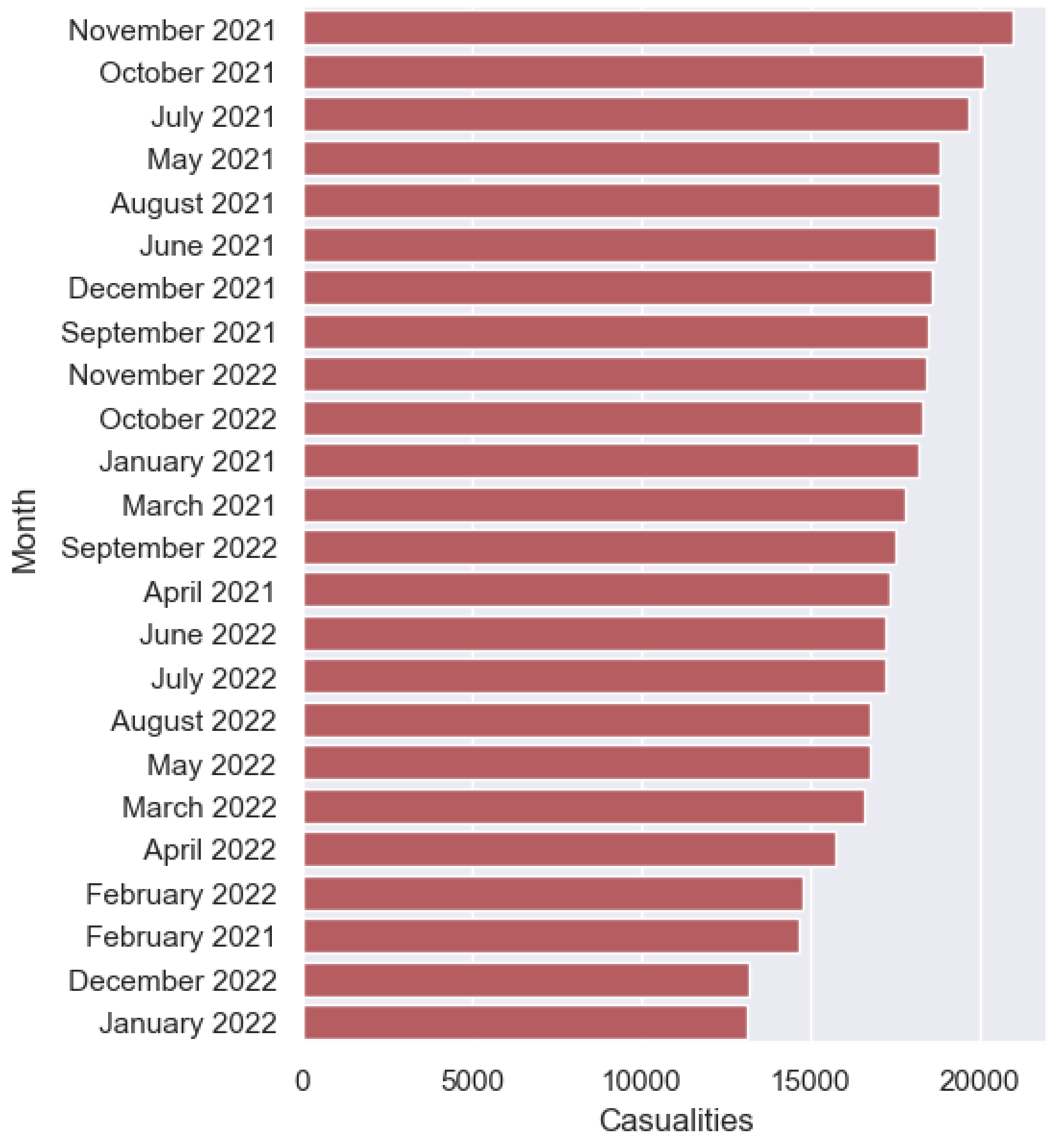
here we used line plot to display ,and we can see that the number ofcasualities on Metropolitan Police have highest around 57 thousand

Months vs Total number of casualities

```
In [29]: a=df.groupby('Month')['Casualities'].sum()
         a
```

```
Out[29]: Month
          April 2021        17328
          April 2022        15766
          August 2021       18787
          August 2022       16786
          December 2021     18573
          December 2022     13184
          February 2021     14636
          February 2022     14802
          January 2021      18160
          January 2022      13157
          July 2021         19657
          July 2022         17194
          June 2021         18714
          June 2022         17217
          March 2021        17809
          March 2022        16573
          May 2021          18836
          May 2022          16767
          November 2021     20965
          November 2022     18432
          October 2021      20105
          October 2022      18281
          September 2021    18446
          September 2022    17493
         Name: Casualities, dtype: int64
```

In [30]:
```
am=df.groupby(['Month'],as_index=False)['Casualities'].sum().sort_values(by='Casualities',ascending=False)
sns.barplot(data=am,y='Month',x='Casualities',color='r')
sns.set(rc={'figure.figsize':(5,7)})
plt.show()
```



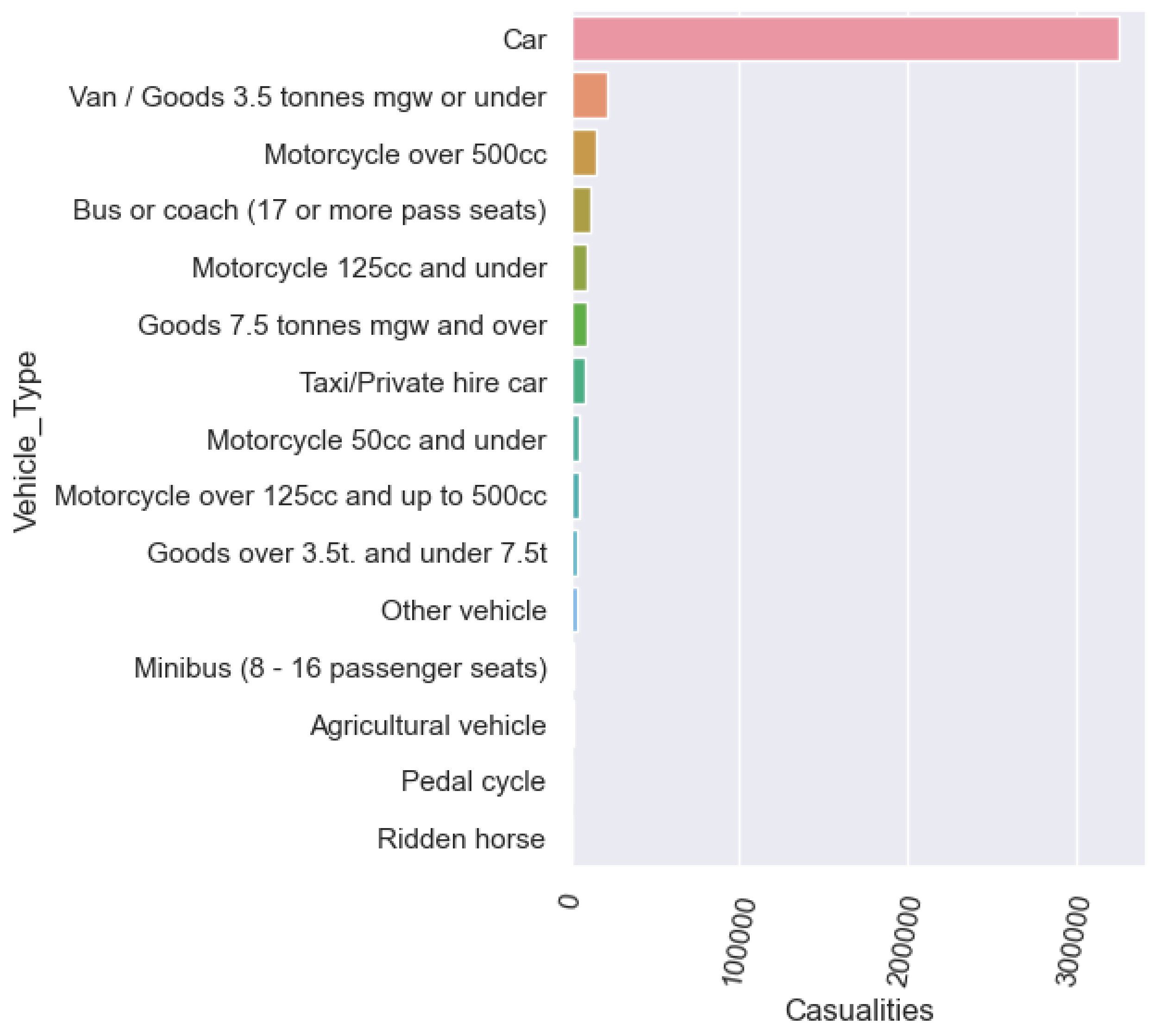
here we can see that monthly casualities in range 12 thousands to 23 thousands and its decreasing with respect to time as per given data

transport namethat mostly get affected

In [31]:
```
ax=df.groupby('Vehicle_Type')['Casualities'].sum()
ax
```

Out[31]:
```
Vehicle_Type
Agricultural vehicle                      1032
Bus or coach (17 or more pass seats)     11702
Car                                     325744
Goods 7.5 tonnes mgw and over             8766
Goods over 3.5t. and under 7.5t           3403
Minibus (8 - 16 passenger seats)          1088
Motorcycle 125cc and under                9108
Motorcycle 50cc and under                 4943
Motorcycle over 125cc and up to 500cc     4466
Motorcycle over 500cc                    15137
Other vehicle                             3328
Pedal cycle                                 92
Ridden horse                                 3
Taxi/Private hire car                     7563
Van / Goods 3.5 tonnes mgw or under      21293
Name: Casualities, dtype: int64
```

In [33]:
```
ax=df.groupby(['Vehicle_Type'],as_index=False)['Casualities'].sum().sort_values(by='Casualities',ascending=False)
sns.barplot(data=ax, y='Vehicle_Type', x='Casualities')
sns.set(rc={'figure.figsize':(4,6)})
plt.xticks(rotation=80)
plt.show()
```

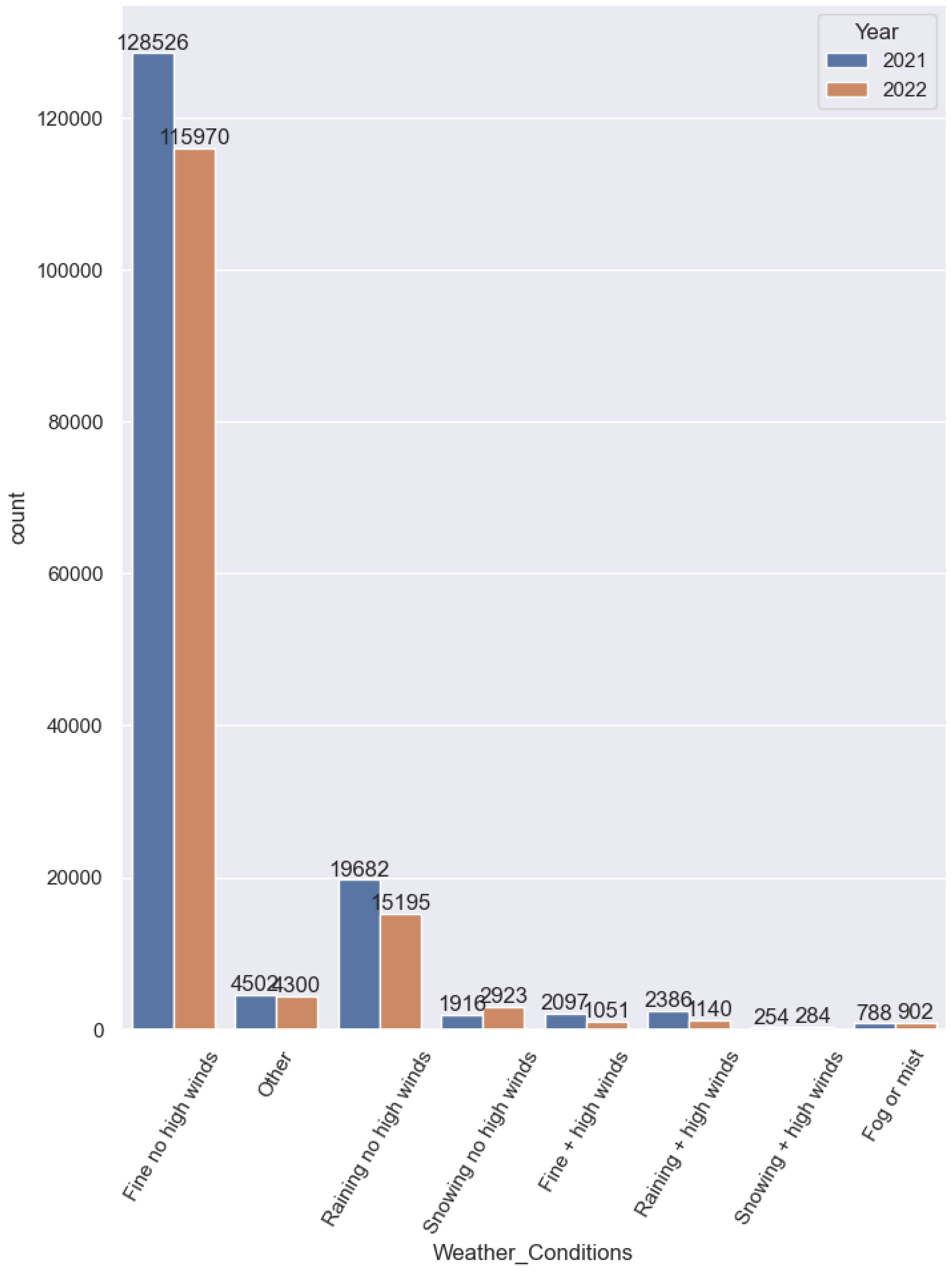


Here we can see that most number of accident happpen with cars its 35 thousands least number of cases 3 with horses and all other are varing from 3 to 15000

What will be the casualities in 2021 and 2022 and how does it changes with weather display?

plot a bar to display casualities Yearly cases vs weather condition year wise

In [38]:
```
cx=sns.countplot(data=df,x='Weather_Conditions',hue='Year')
sns.set(rc={'figure.figsize':(8,10)})
for bars in cx.containers:
    cx.bar_label(bars)
    plt.xticks(rotation=60)
```



here we can see fine no high winds have hoghest number of cases and after this raining and hgh winds respectively

Make a pairplot with the columns of accident severity,speed limit,casualities ,number of ans show how does they change with other columns

In [ ]:
```
sns.pairplot(df,vars=['Accident_Severity','Speed_limit','Casualities','Number_of_Vehicles'])
sns.set(rc={'figure.figsize':(33,60)})
plt.xticks(rotation=80, textsize=7)
plt.show()
```

what is the total number of casualities as per the given area

In [ ]:
```
a=df.groupby(['Area'])['Casualities'].sum()
a
```

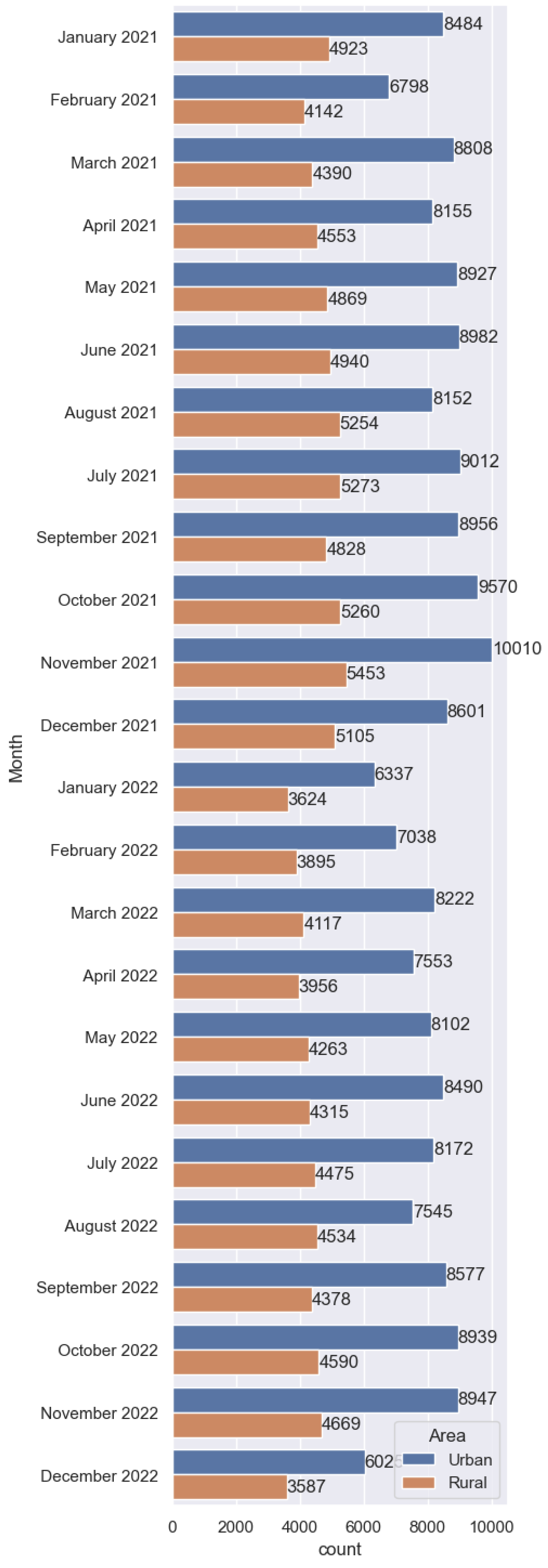
here we can see the casualities percentages on urban areas are more than that of rural area

```
In [ ]: df.columns #to view all the columns use df.columns
```

Name the month have highest and lowest number of cases on urban and rural areas ?

Month vs Area Bar Chart to display casualities of urban and rural area monthwise

In [42]:
```python
ax=sns.countplot(data=df,y='Month',hue='Area')
sns.set(rc={'figure.figsize':(4,28)})
for bars in ax.containers:
    ax.bar_label(bars, label=24)
    plt.xticks()
```



here we can see that the number of rural cases are lesser as compared to urban area , and in urban area highest cases on novwmbwe 2021 and lowest on december 2020 while on other side in rural areas the highest number of cases are on novembwe 2021 and least on December 2022

On which day most number of cases occurs and least as well ?

Display the sum of Casualities in week days

In [46]:
```
df.groupby(['Day_of_Week'])['Casualities'].sum()
```

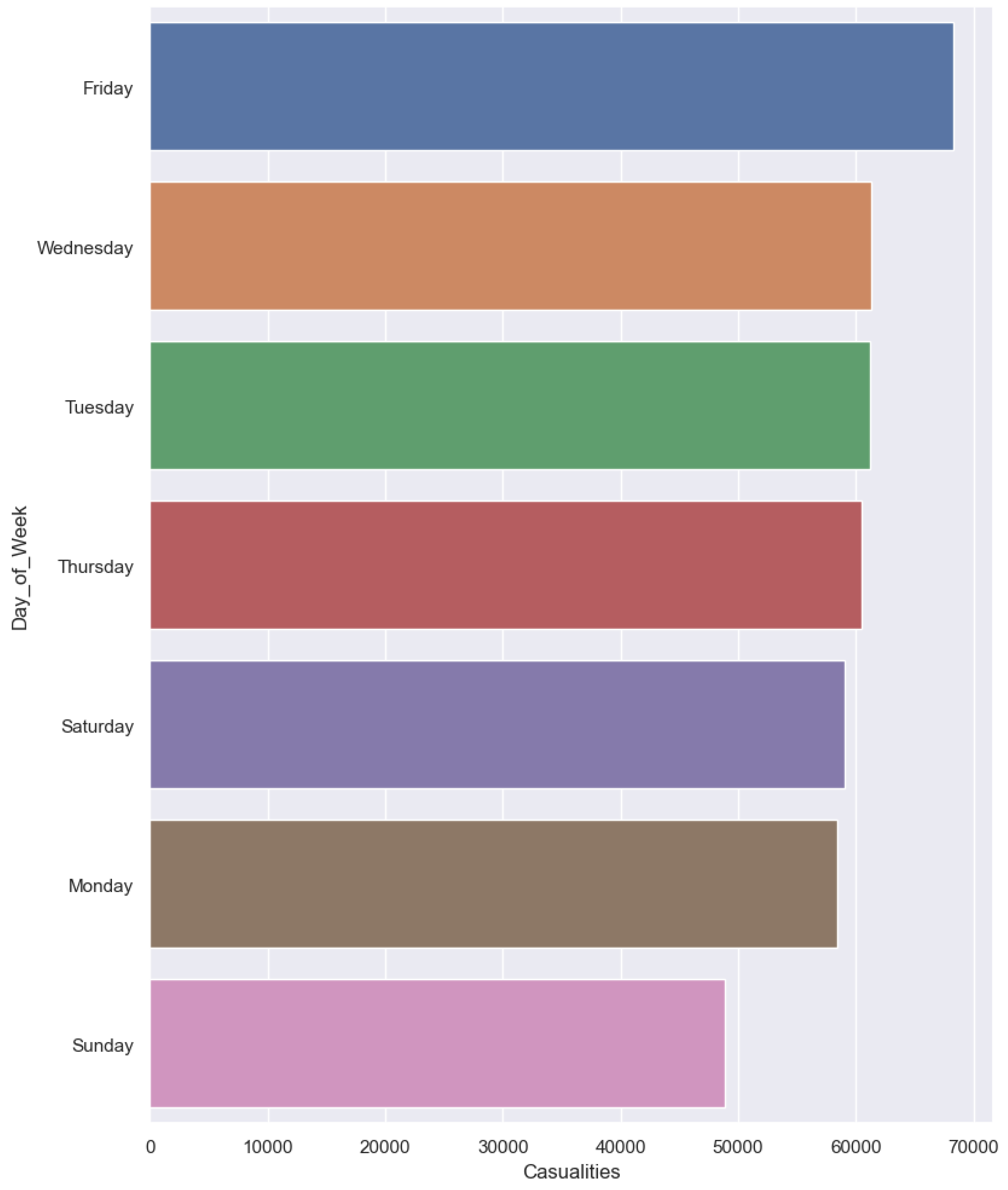
Out[46]:
```
Day_of_Week
Friday       68269
Monday       58443
Saturday     59061
Sunday       48839
Thursday     60510
Tuesday      61245
Wednesday    61301
Name: Casualities, dtype: int64
```

here we can see the highet number of cases onn friday and wednesday respectively

Make a bar plot to dispaly the total casualities on week days

In [49]:
```
 bx=df.groupby(['Day_of_Week'],as_index=False)['Casualities'].sum().sort_values(by='Casualities',ascending=False)
sns.barplot(data=bx,y='Day_of_Week',x='Casualities')

sns.set(rc={'figure.figsize':(9,12)})
plt.show()
```



Do casualities changes with Road type ,if it does then explain it?

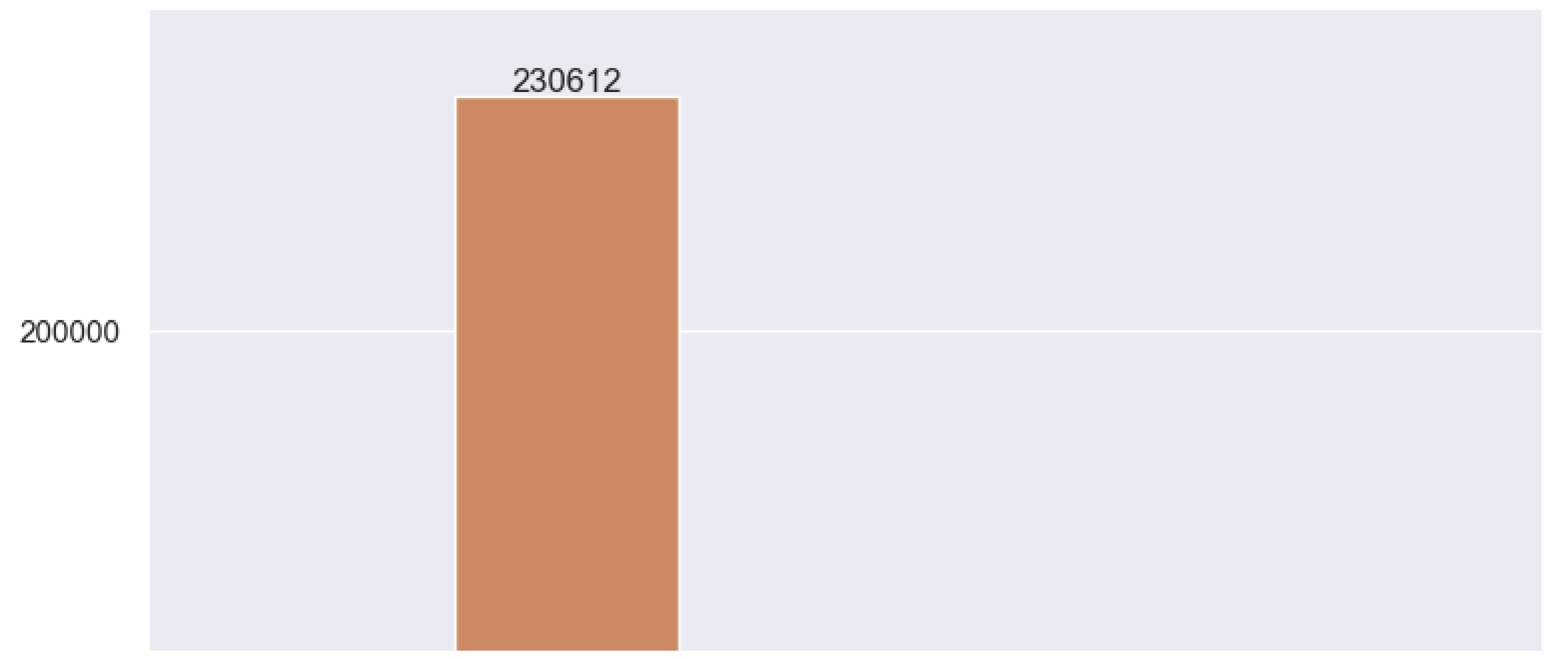
In [50]:
```
cv=df.groupby(['Road_Type'])['Casualities'].count()
cv
```

Out[50]:
```
Road_Type
Dual carriageway       45467
One way street          6197
Roundabout             20929
Single carriageway    230612
Slip road               3234
Name: Casualities, dtype: int64
```

here we can see the most number casualities on single carriageway 2.26lakh and dual carriageway ,Roundabout are around 45 thousand and 20 thousand respectively

make chart a display the casualities on basis of road type ,how does the rate increases and decreases

In [51]:
```
ax=sns.countplot(data=df,x='Road_Type')
sns.set(rc={'figure.figsize':(9,4)})
for bars in ax.containers:
    ax.bar_label(bars)
```

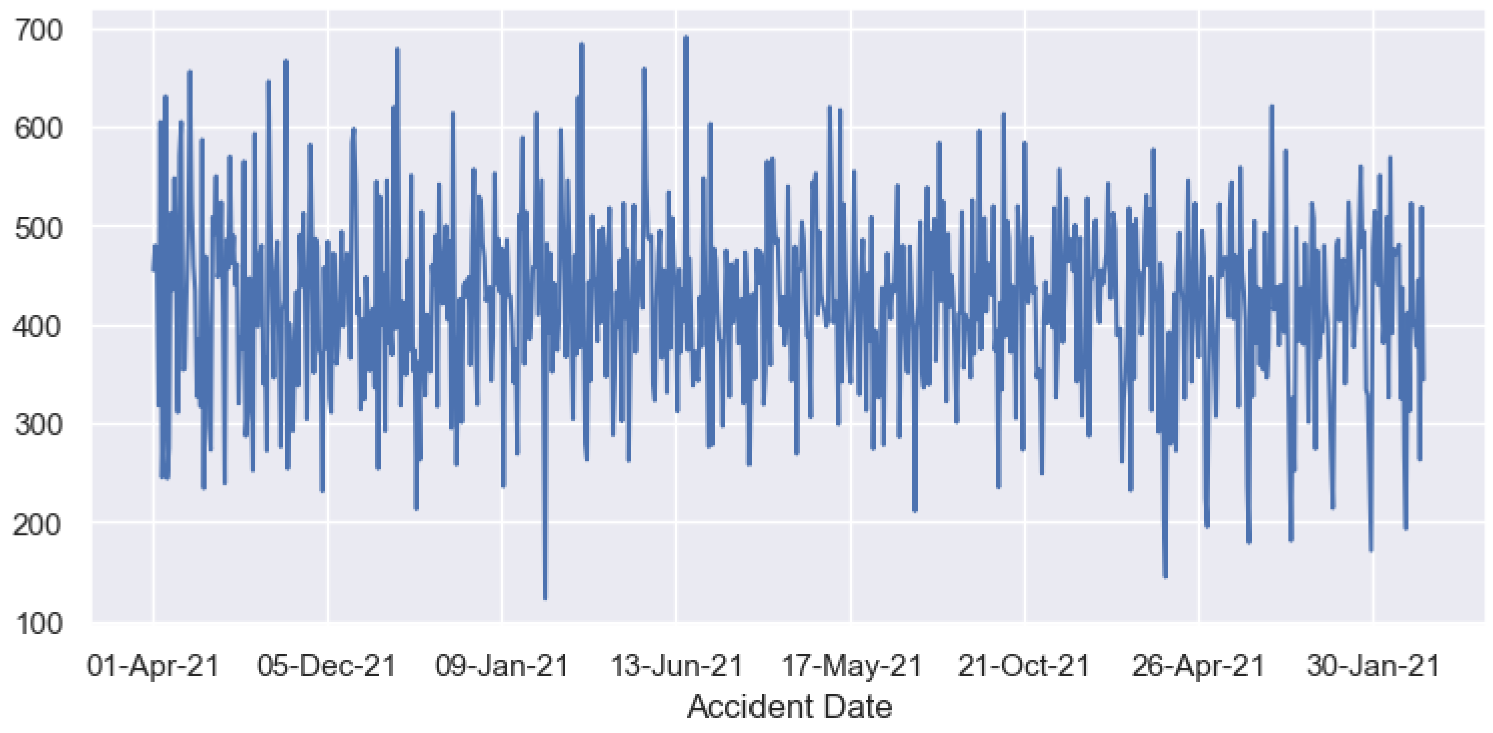


```
one way road and slip road are 6 thousand and 3.1 thousand respectivery which is very much lesser as comapred to single
carriageway
```

do the casualities changes with ,how much and how do ?

make a line plot to display the casualities with the accident date

In [52]:
```
df.groupby(['Accident Date'])['Casualities'].count().plot(kind='line')
sns.set(rc={'figure.figsize':(8,11)})
```



In [53]:
```
df.columns
```

Out[53]:
```
Index(['Accident_Index', 'Date', 'Month', 'Year', 'Accident Date',
       'Day_of_Week', 'Junction_Control', 'Junction_Detail',
       'Accident_Severity', 'Latitude', 'Light_Conditions',
       'Local_Authority_(District)', 'Carriageway_Hazards', 'Longitude',
       'Casualities', 'Number_of_Vehicles', 'Police_Force',
       'Road_Surface_Conditions', 'Road_Type', 'Speed_limit', 'Time', 'Area',
       'Weather_Conditions', 'Vehicle_Type'],
      dtype='object')
```

In [54]:
```python
ab=df.groupby(['Accident Date'])['Casualities'].sum()
ab
```

Out[54]:
```
Accident Date
01-Apr-21    595
01-Apr-22    670
01-Aug-21    712
01-Aug-22    441
01-Dec-21    799
            ...
31-Mar-22    497
31-May-21    631
31-May-22    356
31-Oct-21    731
31-Oct-22    509
Name: Casualities, Length: 730, dtype: int64
```

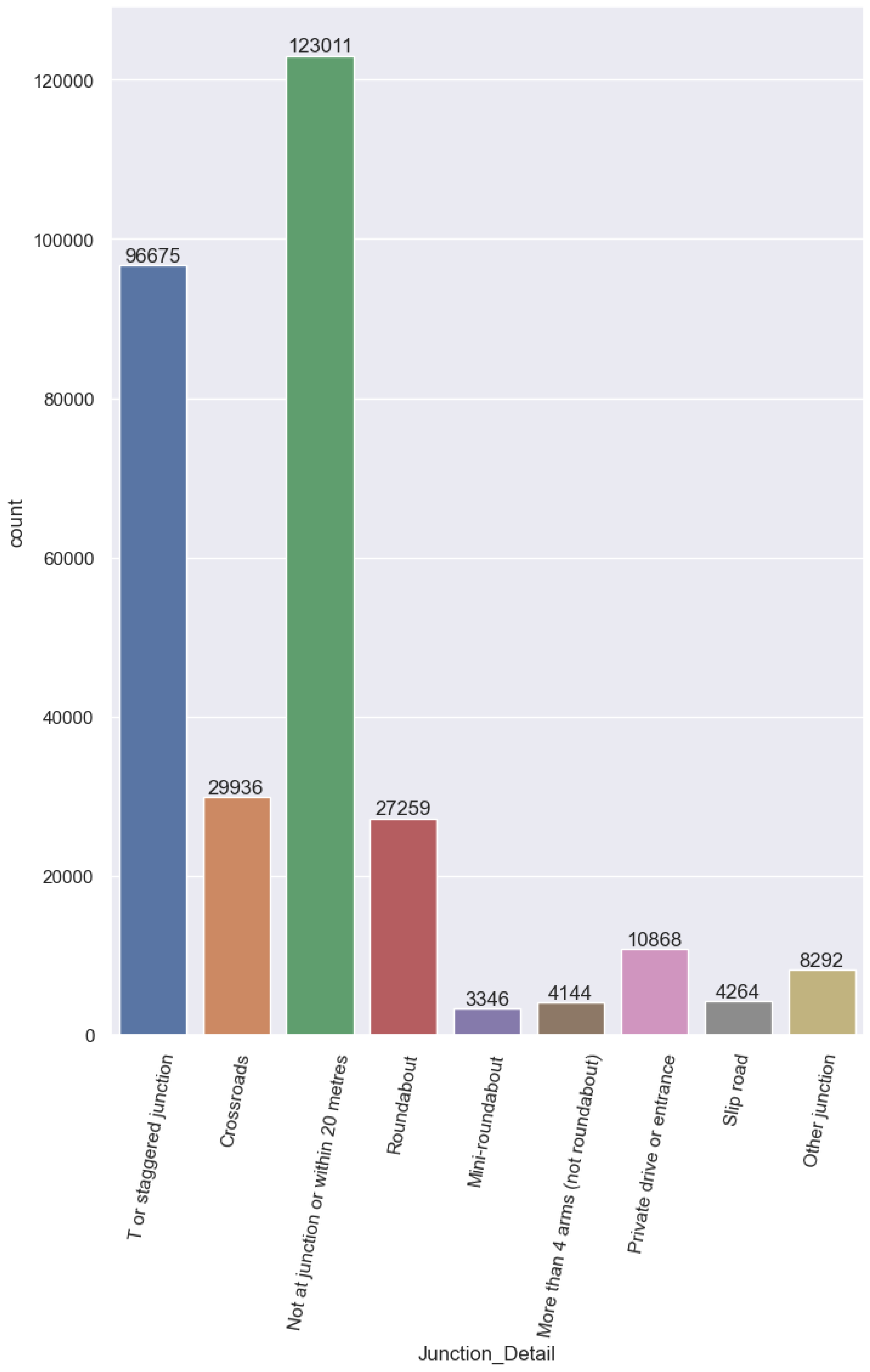
how many cases are on different junctions?

In [55]:
```python
b=df.groupby(['Junction_Detail'])['Casualities'].count()
b
```

Out[55]:
```
Junction_Detail
Crossroads                              29936
Mini-roundabout                          3346
More than 4 arms (not roundabout)        4144
Not at junction or within 20 metres    123011
Other junction                           8292
Private drive or entrance               10868
Roundabout                              27259
Slip road                                4264
T or staggered junction                 96675
Name: Casualities, dtype: int64
```

```
here we can see that highest number of cases are within the range of 20 meter
```

In [56]:
```
ax=sns.countplot(data=df, x='Junction_Detail')
sns.set(rc={'figure.figsize':(8,6)})
plt.xticks(rotation=80)
for bars in ax.containers:
    ax.bar_label(bars)
```



In [57]:
```
a=df.groupby(['Time'])['Casualities'].sum()
a
```

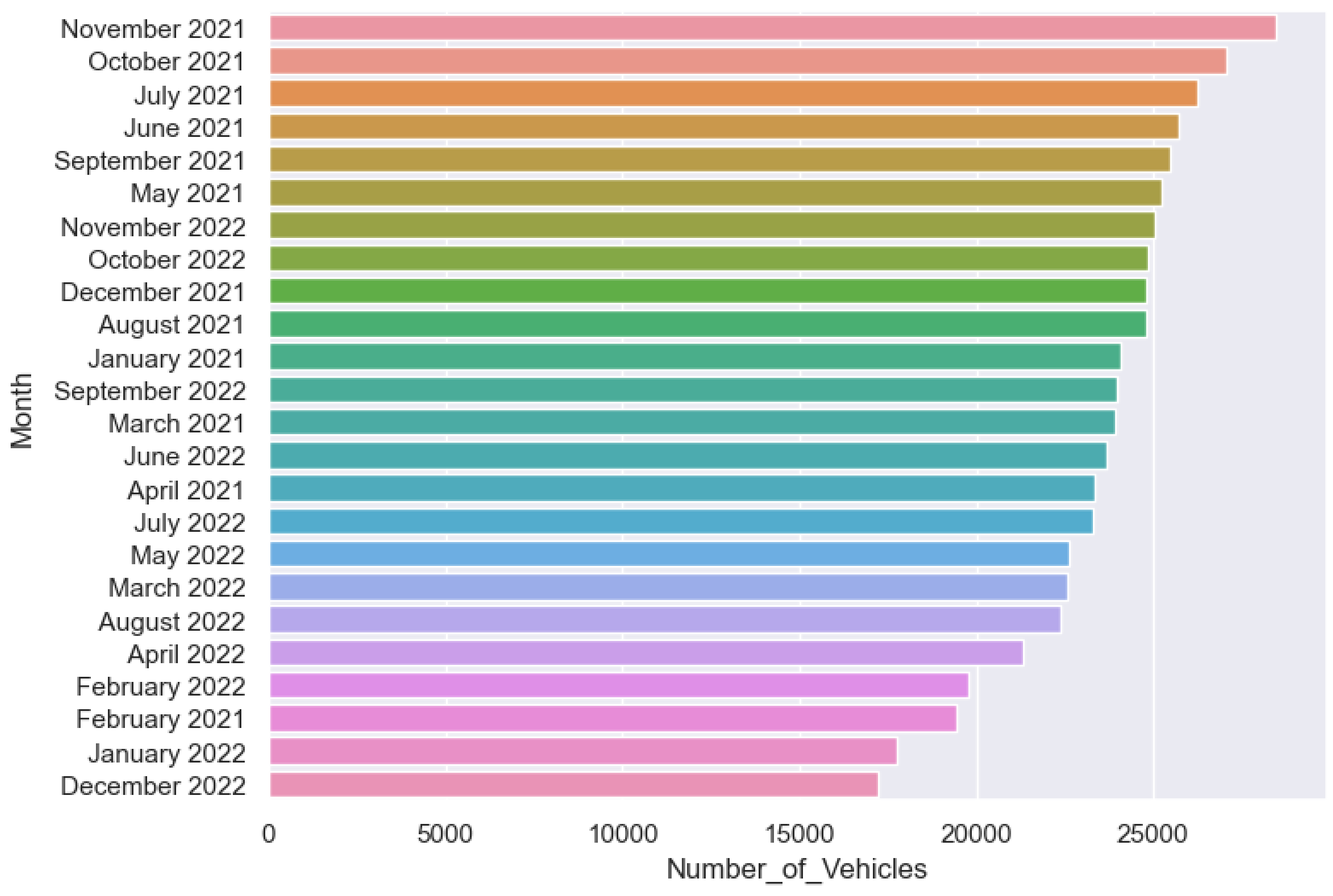
Out[57]:
```
Time
00:01    607
00:02     94
00:03     63
00:04     76
00:05    336
        ...
23:55    295
23:56     62
23:57     56
23:58     74
23:59    100
Name: Casualities, Length: 1439, dtype: int64
```

#whats the total number vehicals accidents occurs monthwise ?

In [58]:
```
a=df.groupby(['Month'])['Number_of_Vehicles'].sum()
a
```

Out[58]:
```
Month
 April 2021        23352
 April 2022        21338
 August 2021       24771
 August 2022       22404
 December 2021     24776
 December 2022     17243
 February 2021     19424
 February 2022     19774
 January 2021      24081
 January 2022      17753
 July 2021         26233
 July 2022         23307
 June 2021         25692
 June 2022         23665
 March 2021        23943
 March 2022        22565
 May 2021          25241
 May 2022          22613
 November 2021     28473
 November 2022     25009
 October 2021      27051
 October 2022      24856
 September 2021    25462
 September 2022    23969
Name: Number_of_Vehicles, dtype: int64
```

In [59]:
```
ax=df.groupby(['Month'],as_index=False)['Number_of_Vehicles'].sum().sort_values(by='Number_of_Vehicles',ascending=False)
sns.barplot(data=ax,y='Month',x='Number_of_Vehicles')
sns.set(rc={'figure.figsize':(5,8)})
```



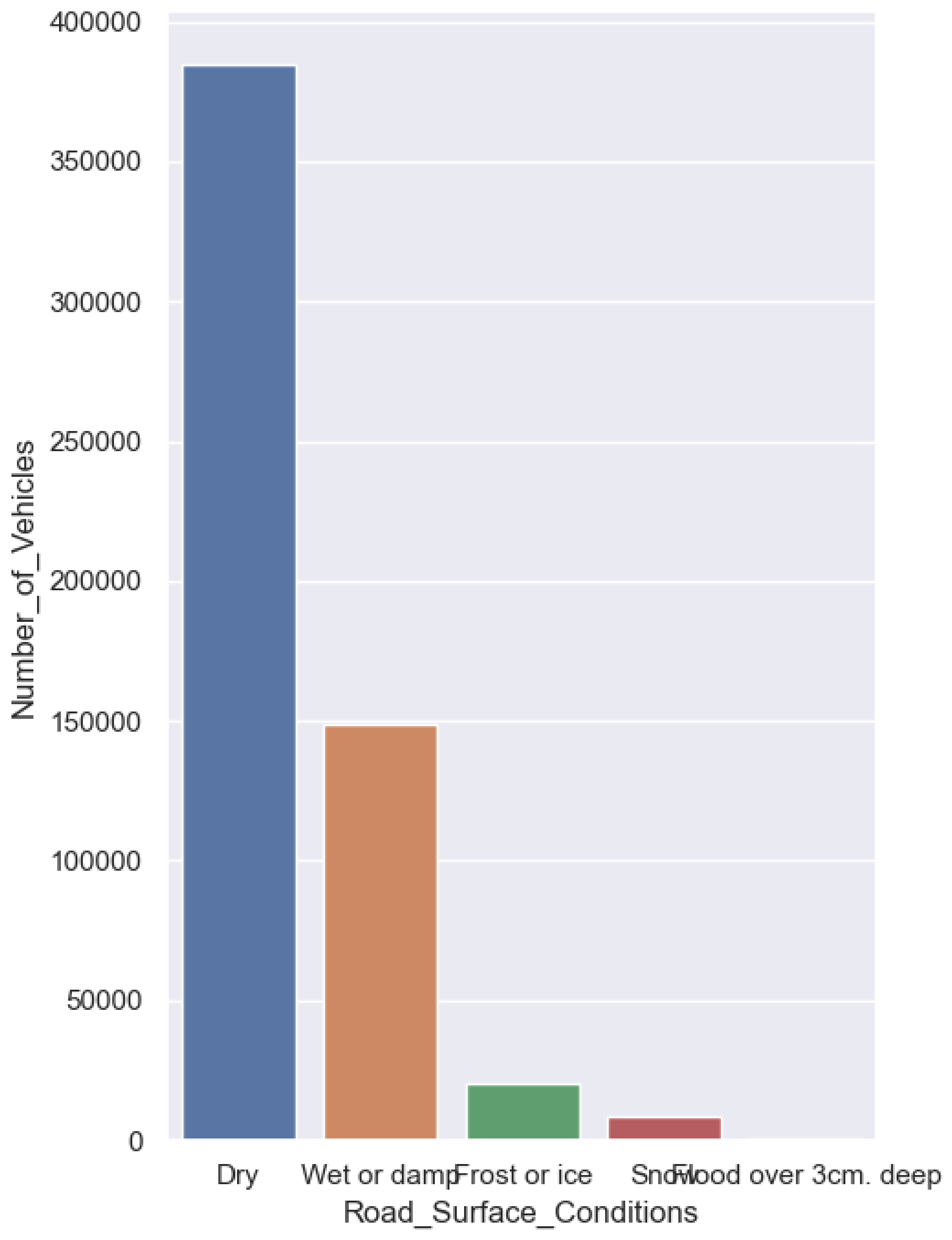
we can esee here highest number of vehicles accident on november and least accident on december 2022

what is the number of casualities with different road conditions?

In [60]:
```
df.groupby(['Road_Surface_Conditions'])['Number_of_Vehicles'].sum()
```

Out[60]:
```
Road_Surface_Conditions
Dry                      384507
Flood over 3cm. deep        607
Frost or ice              19982
Snow                       8337
Wet or damp              149027
Name: Number_of_Vehicles, dtype: int64
```

```
In [61]: zx=df.groupby(['Road_Surface_Conditions'],as_index=False)['Number_of_Vehicles'].sum().sort_values(by='Number_of_Vehicles',ascend
         sns.barplot(data=zx,x='Road_Surface_Conditions',y='Number_of_Vehicles')
         sns.set(rc={'figure.figsize':(6,5)})
```



here we can see that `most number of casualities are on dry road and after this on wet or damp ,froast or ice snow respectively