Task 5

Analyze traffic accident data to identify patterns related to road conditions, weather, and time of day. Visualize accident hotspots and contributing factors.

Loading Libraries and Data

In [56]:
```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
import warnings
warnings.filterwarnings('ignore')
```

In [57]:
```
df=pd.read_csv("RTA Dataset.csv")
df.head()
```

Out[57]:

	Time	Day_of_week	Age_band_of_driver	Sex_of_driver	Educational_level	Vehicle_driver_r
0	17:02:00	Monday	18-30	Male	Above high school	En
1	17:02:00	Monday	31-50	Male	Junior high school	En
2	17:02:00	Monday	18-30	Male	Junior high school	En
3	1:06:00	Sunday	18-30	Male	Junior high school	En
4	1:06:00	Sunday	18-30	Male	Junior high school	En

5 rows × 32 columns

In [58]:
```
df.shape
```

Out[58]:
```
(12316, 32)
```

In [59]:
```
df.describe()
```

Out[59]:

	Number_of_vehicles_involved	Number_of_casualties
count	12316.000000	12316.000000
mean	2.040679	1.548149
std	0.688790	1.007179
min	1.000000	1.000000
25%	2.000000	1.000000
50%	2.000000	1.000000
75%	2.000000	2.000000
max	7.000000	8.000000

In [60]:
```
df.describe(include="all")
```

Out[60]:

	Time	Day_of_week	Age_band_of_driver	Sex_of_driver	Educational_level	Vehicle_dr
count	12316	12316	12316	12316	11575	
unique	1074	7	5	3	7	
top	15:30:00	Friday	18-30	Male	Junior high school	
freq	120	2041	4271	11437	7619	
mean	NaN	NaN	NaN	NaN	NaN	
std	NaN	NaN	NaN	NaN	NaN	
min	NaN	NaN	NaN	NaN	NaN	
25%	NaN	NaN	NaN	NaN	NaN	
50%	NaN	NaN	NaN	NaN	NaN	
75%	NaN	NaN	NaN	NaN	NaN	
max	NaN	NaN	NaN	NaN	NaN	

11 rows × 32 columns

In [61]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 12316 entries, 0 to 12315
Data columns (total 32 columns):
 #   Column                       Non-Null Count  Dtype 
---  ------                       --------------  ----- 
 0   Time                         12316 non-null  object
 1   Day_of_week                  12316 non-null  object
 2   Age_band_of_driver           12316 non-null  object
 3   Sex_of_driver                12316 non-null  object
 4   Educational_level            11575 non-null  object
 5   Vehicle_driver_relation      11737 non-null  object
 6   Driving_experience           11487 non-null  object
 7   Type_of_vehicle              11366 non-null  object
 8   Owner_of_vehicle             11834 non-null  object
 9   Service_year_of_vehicle      8388 non-null   object
 10  Defect_of_vehicle            7889 non-null   object
 11  Area_accident_occured        12077 non-null  object
 12  Lanes_or_Medians             11931 non-null  object
 13  Road_allignment              12174 non-null  object
 14  Types_of_Junction            11429 non-null  object
 15  Road_surface_type            12144 non-null  object
 16  Road_surface_conditions      12316 non-null  object
 17  Light_conditions             12316 non-null  object
 18  Weather_conditions           12316 non-null  object
 19  Type_of_collision            12161 non-null  object
 20  Number_of_vehicles_involved  12316 non-null  int64 
 21  Number_of_casualties         12316 non-null  int64 
 22  Vehicle_movement             12008 non-null  object
 23  Casualty_class               12316 non-null  object
 24  Sex_of_casualty              12316 non-null  object
 25  Age_band_of_casualty         12316 non-null  object
 26  Casualty_severity            12316 non-null  object
 27  Work_of_casuality            9118 non-null   object
 28  Fitness_of_casuality         9681 non-null   object
 29  Pedestrian_movement          12316 non-null  object
 30  Cause_of_accident            12316 non-null  object
 31  Accident_severity            12316 non-null  object
dtypes: int64(2), object(30)
memory usage: 3.0+ MB
```

Exploratory Data Analysis

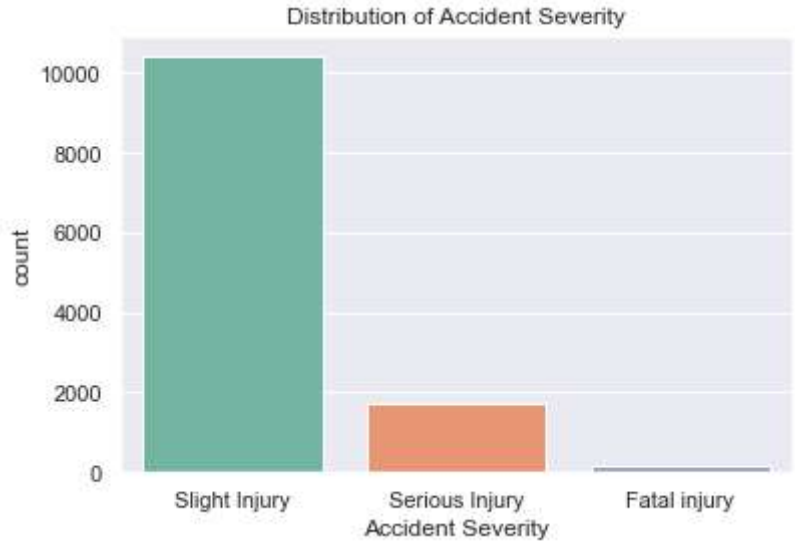
In [62]:
```
df.duplicated().sum()
```

Out[62]:
```
0
```

In [63]:
```
df['Accident_severity'].value_counts()
```

Out[63]:
```
Slight Injury     10415
Serious Injury     1743
Fatal injury        158
Name: Accident_severity, dtype: int64
```

In [64]:
```
import seaborn as sns
import matplotlib.pyplot as plt
sns.countplot(x=df['Accident_severity'], palette='Set2')
plt.title('Distribution of Accident Severity')
plt.xlabel('Accident Severity')
plt.show()
```



Handling missing values

In [65]:
```
df.isna().sum()
```

Out[65]:
```
Time                           0
Day_of_week                    0
Age_band_of_driver             0
Sex_of_driver                  0
Educational_level            741
Vehicle_driver_relation      579
Driving_experience           829
Type_of_vehicle              950
Owner_of_vehicle             482
Service_year_of_vehicle     3928
Defect_of_vehicle           4427
Area_accident_occured        239
Lanes_or_Medians             385
Road_allignment              142
Types_of_Junction            887
Road_surface_type            172
Road_surface_conditions        0
Light_conditions               0
Weather_conditions             0
Type_of_collision            155
Number_of_vehicles_involved    0
Number_of_casualties           0
Vehicle_movement             308
Casualty_class                 0
Sex_of_casualty                0
Age_band_of_casualty           0
Casualty_severity              0
Work_of_casuality           3198
Fitness_of_casuality        2635
Pedestrian_movement            0
Cause_of_accident              0
Accident_severity              0
dtype: int64
```

In [66]:
```
df.drop(['Service_year_of_vehicle','Defect_of_vehicle','Work_of_casuality', 'F
        axis = 1, inplace = True)
df.head()
```

Out[66]:

	Day_of_week	Age_band_of_driver	Sex_of_driver	Educational_level	Vehicle_driver_relation	D
0	Monday	18-30	Male	Above high school	Employee	
1	Monday	31-50	Male	Junior high school	Employee	
2	Monday	18-30	Male	Junior high school	Employee	
3	Sunday	18-30	Male	Junior high school	Employee	
4	Sunday	18-30	Male	Junior high school	Employee	

5 rows × 27 columns

In [67]:
```
categorical=[i for i in df.columns if df[i].dtype=='O']
print('The categorical variables are',categorical)
```

```
The categorical variables are ['Day_of_week', 'Age_band_of_driver', 'Sex_of_
driver', 'Educational_level', 'Vehicle_driver_relation', 'Driving_experienc
e', 'Type_of_vehicle', 'Owner_of_vehicle', 'Area_accident_occured', 'Lanes_o
r_Medians', 'Road_allignment', 'Types_of_Junction', 'Road_surface_type', 'Ro
ad_surface_conditions', 'Light_conditions', 'Weather_conditions', 'Type_of_c
ollision', 'Vehicle_movement', 'Casualty_class', 'Sex_of_casualty', 'Age_ban
d_of_casualty', 'Casualty_severity', 'Pedestrian_movement', 'Cause_of_accide
nt', 'Accident_severity']
```

In [68]:
```
for i in categorical:
    df[i].fillna(df[i].mode()[0],inplace=True)
```

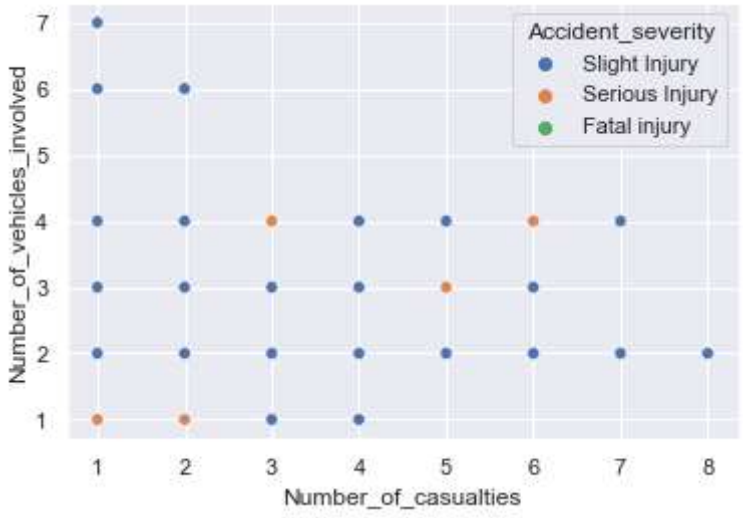
In [69]:
```
df.isna().sum()
```

Out[69]:
```
Day_of_week                  0
Age_band_of_driver           0
Sex_of_driver                0
Educational_level            0
Vehicle_driver_relation      0
Driving_experience           0
Type_of_vehicle              0
Owner_of_vehicle             0
Area_accident_occured        0
Lanes_or_Medians             0
Road_allignment              0
Types_of_Junction            0
Road_surface_type            0
Road_surface_conditions      0
Light_conditions             0
Weather_conditions           0
Type_of_collision            0
Number_of_vehicles_involved  0
Number_of_casualties         0
Vehicle_movement             0
Casualty_class               0
Sex_of_casualty              0
Age_band_of_casualty         0
Casualty_severity            0
Pedestrian_movement          0
Cause_of_accident            0
Accident_severity            0
dtype: int64
```

Data Visualization

In [70]:
```
sns.scatterplot(x=df['Number_of_casualties'], y=df['Number_of_vehicles_involve
```

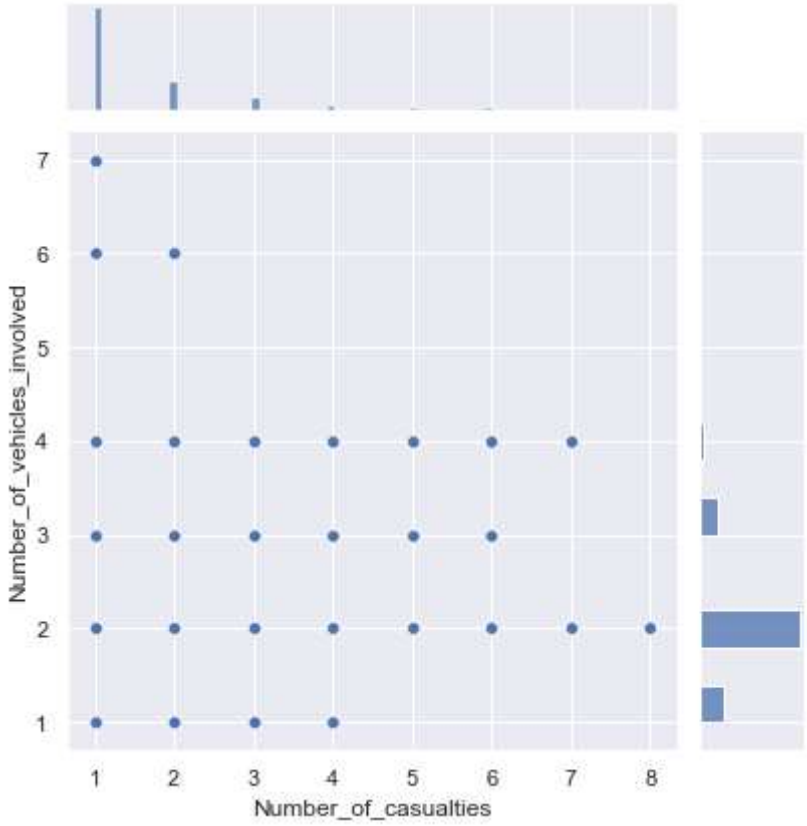
Out[70]:
```
<AxesSubplot:xlabel='Number_of_casualties', ylabel='Number_of_vehicles_invol
ved'>
```



There is no visible correlation between Number_of_casualties and Number_of_vehicles_involved columns

In [71]:
```
sns.jointplot(x='Number_of_casualties',y='Number_of_vehicles_involved',data=d
```

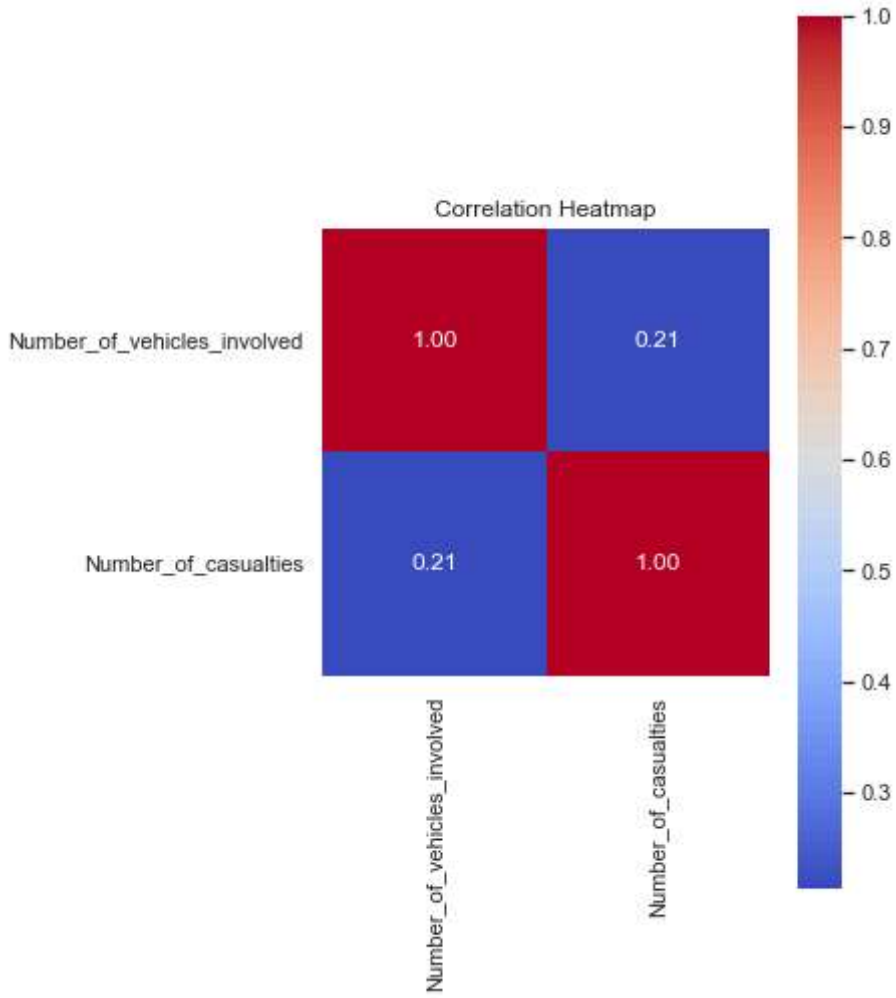
Out[71]:
```
<seaborn.axisgrid.JointGrid at 0x234715d8910>
```



In [72]:
```
print(df.info())
numeric_df = df.select_dtypes(include=['number'])
correlation_matrix = numeric_df.corr()
print(correlation_matrix)
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 12316 entries, 0 to 12315
Data columns (total 27 columns):
 #   Column                       Non-Null Count  Dtype 
---  ------                       --------------  ----- 
 0   Day_of_week                  12316 non-null  object
 1   Age_band_of_driver           12316 non-null  object
 2   Sex_of_driver                12316 non-null  object
 3   Educational_level            12316 non-null  object
 4   Vehicle_driver_relation      12316 non-null  object
 5   Driving_experience           12316 non-null  object
 6   Type_of_vehicle              12316 non-null  object
 7   Owner_of_vehicle             12316 non-null  object
 8   Area_accident_occured        12316 non-null  object
 9   Lanes_or_Medians             12316 non-null  object
 10  Road_allignment              12316 non-null  object
 11  Types_of_Junction            12316 non-null  object
 12  Road_surface_type            12316 non-null  object
 13  Road_surface_conditions      12316 non-null  object
 14  Light_conditions             12316 non-null  object
 15  Weather_conditions           12316 non-null  object
 16  Type_of_collision            12316 non-null  object
 17  Number_of_vehicles_involved  12316 non-null  int64 
 18  Number_of_casualties         12316 non-null  int64 
 19  Vehicle_movement             12316 non-null  object
 20  Casualty_class               12316 non-null  object
 21  Sex_of_casualty              12316 non-null  object
 22  Age_band_of_casualty         12316 non-null  object
 23  Casualty_severity            12316 non-null  object
 24  Pedestrian_movement          12316 non-null  object
 25  Cause_of_accident            12316 non-null  object
 26  Accident_severity            12316 non-null  object
dtypes: int64(2), object(25)
memory usage: 2.5+ MB
None
                             Number_of_vehicles_involved  Number_of_casualti
es
Number_of_vehicles_involved                     1.000000              0.2134
27
Number_of_casualties                            0.213427              1.0000
00
```

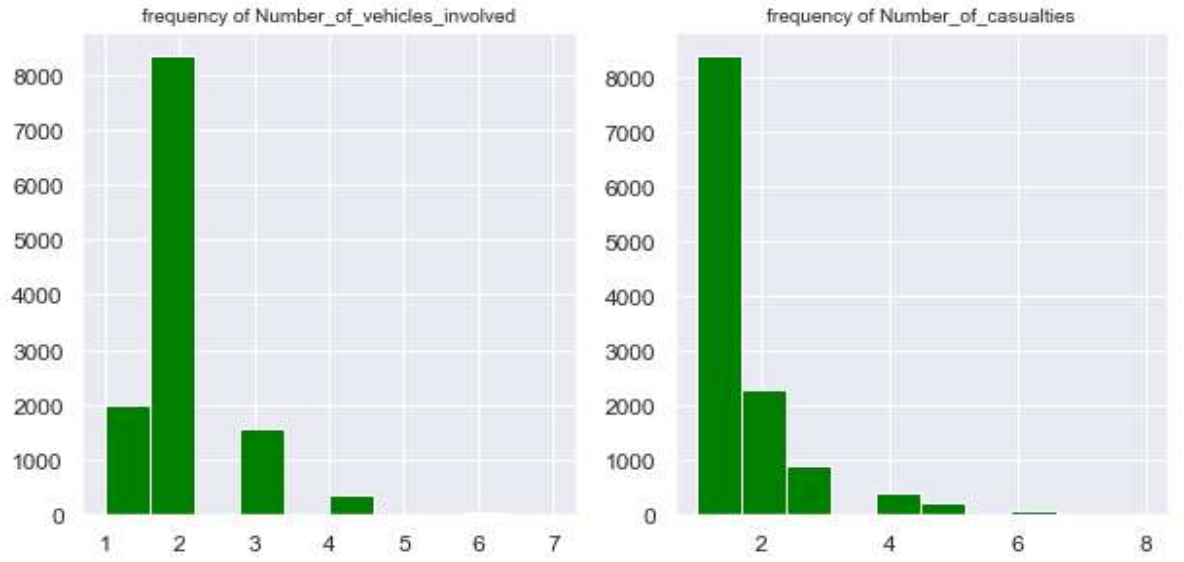
In [73]:
```
correlation_matrix = df.select_dtypes(include=['number']).corr()
plt.figure(figsize=(5, 8))
sns.heatmap(correlation_matrix, annot=True, fmt=".2f", cmap='coolwarm', square
plt.title('Correlation Heatmap')
plt.show()
```



In [74]:
```
numerical=[i for i in df.columns if df[i].dtype!='O']
print('The numerica variables are',numerical)
```

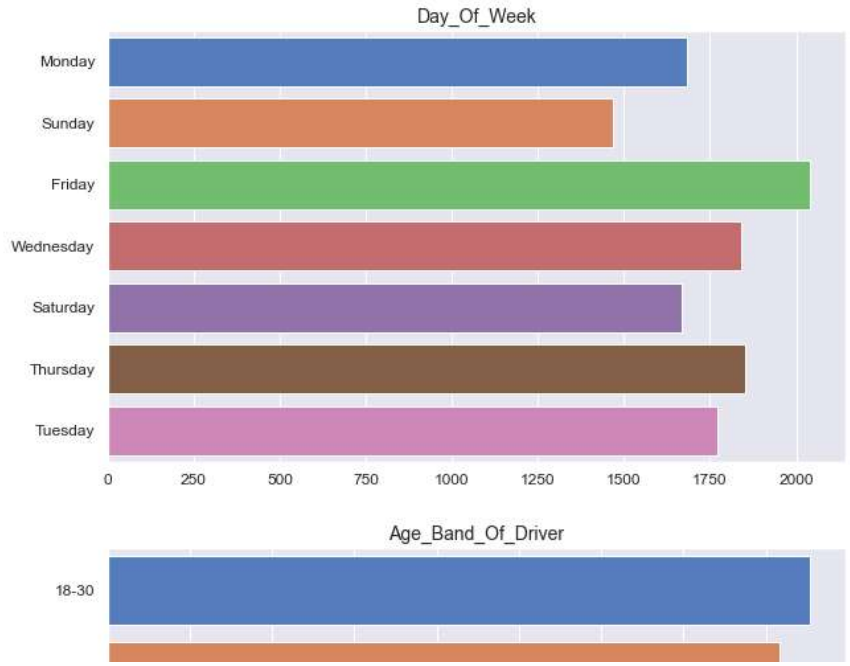
```
The numerica variables are ['Number_of_vehicles_involved', 'Number_of_casual
ties']
```

In [75]:
```
#distribution for numerical columns
plt.figure(figsize=(10,10))
plotnumber = 1
for i in numerical:
    if plotnumber <= df.shape[1]:
        ax1 = plt.subplot(2,2,plotnumber)
        plt.hist(df[i],color='green')
        plt.xticks(fontsize=12)
        plt.yticks(fontsize=12)
        plt.title('frequency of '+i, fontsize=10)
    plotnumber +=1
```



Most accidents are occured when 2 vehicles are involved and 1 casuality is happend mostly in the accidents.

In [76]:
```
#count plot for categorical values
plt.figure(figsize=(10,200))
plotnumber = 1
for col in categorical:
    if plotnumber <= df.shape[1] and col!='Pedestrian_movement':
        ax1 = plt.subplot(28,1,plotnumber)
        sns.countplot(data=df, y=col, palette='muted')
        plt.xticks(fontsize=12)
        plt.yticks(fontsize=12)
        plt.title(col.title(), fontsize=14)
        plt.xlabel('')
        plt.ylabel('')
    plotnumber +=1
```



Handling Categorical Values

In [77]:
```
df.dtypes
```

Out[77]:
```
Day_of_week                   object
Age_band_of_driver            object
Sex_of_driver                 object
Educational_level             object
Vehicle_driver_relation       object
Driving_experience            object
Type_of_vehicle               object
Owner_of_vehicle              object
Area_accident_occured         object
Lanes_or_Medians              object
Road_allignment               object
Types_of_Junction             object
Road_surface_type             object
Road_surface_conditions       object
Light_conditions              object
Weather_conditions            object
Type_of_collision             object
Number_of_vehicles_involved    int64
Number_of_casualties           int64
Vehicle_movement              object
Casualty_class                object
Sex_of_casualty               object
Age_band_of_casualty          object
Casualty_severity             object
Pedestrian_movement           object
Cause_of_accident             object
Accident_severity             object
dtype: object
```

Since there are so many categorical values, we need to use feature selection We need to perform label encoding before applying chi square analysis

In [78]:
```
from sklearn.preprocessing import LabelEncoder
le=LabelEncoder()
df1=pd.DataFrame()
#adding all the categorical columns except the output to new data frame
for i in categorical:
    if i!= 'Accident_severity':
        df1[i]=le.fit_transform(df[i])
```

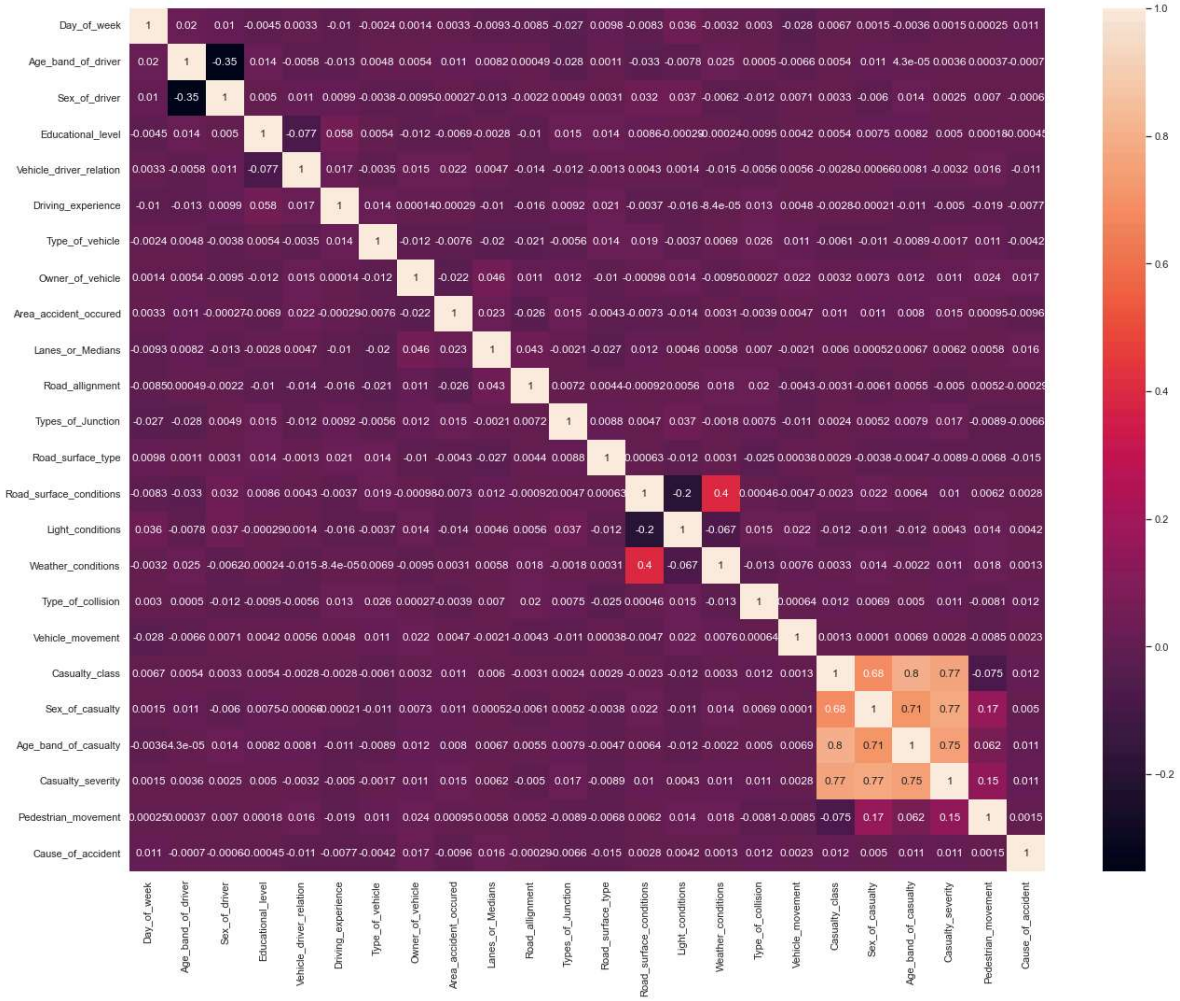
In [79]:
```
#confirming the data type
df1.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 12316 entries, 0 to 12315
Data columns (total 24 columns):
 #   Column                    Non-Null Count  Dtype
---  ------                    --------------  -----
 0   Day_of_week               12316 non-null  int32
 1   Age_band_of_driver        12316 non-null  int32
 2   Sex_of_driver             12316 non-null  int32
 3   Educational_level         12316 non-null  int32
 4   Vehicle_driver_relation   12316 non-null  int32
 5   Driving_experience        12316 non-null  int32
 6   Type_of_vehicle           12316 non-null  int32
 7   Owner_of_vehicle          12316 non-null  int32
 8   Area_accident_occured     12316 non-null  int32
 9   Lanes_or_Medians          12316 non-null  int32
 10  Road_allignment           12316 non-null  int32
 11  Types_of_Junction         12316 non-null  int32
 12  Road_surface_type         12316 non-null  int32
 13  Road_surface_conditions   12316 non-null  int32
 14  Light_conditions          12316 non-null  int32
 15  Weather_conditions        12316 non-null  int32
 16  Type_of_collision         12316 non-null  int32
 17  Vehicle_movement          12316 non-null  int32
 18  Casualty_class            12316 non-null  int32
 19  Sex_of_casualty           12316 non-null  int32
 20  Age_band_of_casualty      12316 non-null  int32
 21  Casualty_severity         12316 non-null  int32
 22  Pedestrian_movement       12316 non-null  int32
 23  Cause_of_accident         12316 non-null  int32
dtypes: int32(24)
memory usage: 1.1 MB
```

Correlation

In [80]:
```
plt.figure(figsize=(22,17))
sns.set(font_scale=1)
sns.heatmap(df1.corr(), annot=True)
```

Out[80]: <AxesSubplot:>



In [81]:
```
df1.head()
```

Out[81]:

	Day_of_week	Age_band_of_driver	Sex_of_driver	Educational_level	Vehicle_driver_relation	D
0	1	0	1	0	0	
1	1	1	1	4	0	
2	1	0	1	4	0	
3	3	0	1	4	0	
4	3	0	1	4	0	

5 rows × 24 columns

In [82]:
```
from sklearn.feature_selection import chi2
f_p_values=chi2(df1,df['Accident_severity'])
```

In [83]:
```
#f_p_values will return Fscore and pvalues
f_p_values
```

Out[83]:
```
(array([ 0.15822071,  8.91539214,  0.1431894 ,  0.17458477,  5.34534549,
         4.49967858,  1.07767124,  1.10426215,  3.61654037,  3.28161464,
         0.1319306 ,  3.08648691,  6.99480557,  0.61510308, 16.08282359,
         1.14934538, 10.09632283,  2.20071197,  3.2168602 ,  0.12594479,
        13.77841337,  0.20273788,  0.39747982,  3.19366551]),
 array([9.23937958e-01, 1.15890328e-02, 9.30908116e-01, 9.16409114e-01,
        6.90673790e-02, 1.05416165e-01, 5.83427189e-01, 5.75721597e-01,
        1.63937473e-01, 1.93823502e-01, 9.36163348e-01, 2.13686893e-01,
        3.02759144e-02, 7.35244973e-01, 3.21854237e-04, 5.62889079e-01,
        6.42112839e-03, 3.32752607e-01, 2.00201664e-01, 9.38969394e-01,
        1.01872169e-03, 9.03599597e-01, 8.19763078e-01, 2.02536988e-01]))
```

In [84]:
```
#for better understanding and ease of access adding them to a new dataframe
f_p_values1=pd.DataFrame({'features':df1.columns, 'Fscore': f_p_values[0], 'Pv
f_p_values1
```

Out[84]:

	features	Fscore	Pvalues
0	Day_of_week	0.158221	0.923938
1	Age_band_of_driver	8.915392	0.011589
2	Sex_of_driver	0.143189	0.930908
3	Educational_level	0.174585	0.916409
4	Vehicle_driver_relation	5.345345	0.069067
5	Driving_experience	4.499679	0.105416
6	Type_of_vehicle	1.077671	0.583427
7	Owner_of_vehicle	1.104262	0.575722
8	Area_accident_occured	3.616540	0.163937
9	Lanes_or_Medians	3.281615	0.193824
10	Road_allignment	0.131931	0.936163
11	Types_of_Junction	3.086487	0.213687
12	Road_surface_type	6.994806	0.030276
13	Road_surface_conditions	0.615103	0.735245
14	Light_conditions	16.082824	0.000322
15	Weather_conditions	1.149345	0.562889
16	Type_of_collision	10.096323	0.006421
17	Vehicle_movement	2.200712	0.332753
18	Casualty_class	3.216860	0.200202
19	Sex_of_casualty	0.125945	0.938969
20	Age_band_of_casualty	13.778413	0.001019
21	Casualty_severity	0.202738	0.903600
22	Pedestrian_movement	0.397480	0.819763
23	Cause_of_accident	3.193666	0.202537

In [85]:
```
#since we want lower Pvalues we are sorting the features
f_p_values1.sort_values(by='Pvalues',ascending=True)
```

Out[85]:

	features	Fscore	Pvalues
14	Light_conditions	16.082824	0.000322
20	Age_band_of_casualty	13.778413	0.001019
16	Type_of_collision	10.096323	0.006421
1	Age_band_of_driver	8.915392	0.011589
12	Road_surface_type	6.994806	0.030276
4	Vehicle_driver_relation	5.345345	0.069067
5	Driving_experience	4.499679	0.105416
8	Area_accident_occured	3.616540	0.163937
9	Lanes_or_Medians	3.281615	0.193824
18	Casualty_class	3.216860	0.200202
23	Cause_of_accident	3.193666	0.202537
11	Types_of_Junction	3.086487	0.213687
17	Vehicle_movement	2.200712	0.332753
15	Weather_conditions	1.149345	0.562889
7	Owner_of_vehicle	1.104262	0.575722
6	Type_of_vehicle	1.077671	0.583427
13	Road_surface_conditions	0.615103	0.735245
22	Pedestrian_movement	0.397480	0.819763
21	Casualty_severity	0.202738	0.903600
3	Educational_level	0.174585	0.916409
0	Day_of_week	0.158221	0.923938
2	Sex_of_driver	0.143189	0.930908
10	Road_allignment	0.131931	0.936163
19	Sex_of_casualty	0.125945	0.938969

we need higher Fscore and lower the Pvalues, so by evaluating, we can remove Owner_of_vehicle, Type_of_vehicle, Road_surface_conditions, Pedestrian_movement,Casualty_severity,Educational_level,Day_of_week,Sex_of_driver,Road_a Sex_of_casualty

In [86]:
```
#after evaluating we are removing lesser important columns and storing to a ne
df2=df.drop(['Owner_of_vehicle', 'Type_of_vehicle', 'Road_surface_conditions',
        'Casualty_severity','Educational_level','Day_of_week','Sex_of_driver',
        'Sex_of_casualty'],axis=1)
df2.head()
```

Out[86]:

	Age_band_of_driver	Vehicle_driver_relation	Driving_experience	Area_accident_occured	Lanes
0	18-30	Employee	1-2yr	Residential areas	Tw wi
1	31-50	Employee	Above 10yr	Office areas	Undi
2	18-30	Employee	1-2yr	Recreational areas	
3	18-30	Employee	5-10yr	Office areas	
4	18-30	Employee	2-5yr	Industrial areas	

In [87]:
```
df2.shape
```

Out[87]:
```
(12316, 17)
```

In [88]:
```
df2.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 12316 entries, 0 to 12315
Data columns (total 17 columns):
 #   Column                       Non-Null Count  Dtype 
---  ------                       --------------  ----- 
 0   Age_band_of_driver           12316 non-null  object
 1   Vehicle_driver_relation      12316 non-null  object
 2   Driving_experience           12316 non-null  object
 3   Area_accident_occured        12316 non-null  object
 4   Lanes_or_Medians             12316 non-null  object
 5   Types_of_Junction            12316 non-null  object
 6   Road_surface_type            12316 non-null  object
 7   Light_conditions             12316 non-null  object
 8   Weather_conditions           12316 non-null  object
 9   Type_of_collision            12316 non-null  object
 10  Number_of_vehicles_involved  12316 non-null  int64 
 11  Number_of_casualties         12316 non-null  int64 
 12  Vehicle_movement             12316 non-null  object
 13  Casualty_class               12316 non-null  object
 14  Age_band_of_casualty         12316 non-null  object
 15  Cause_of_accident            12316 non-null  object
 16  Accident_severity            12316 non-null  object
dtypes: int64(2), object(15)
memory usage: 1.6+ MB
```

In [89]:
```
#to check distinct values in each categorical columns we are storing them to c
categorical_new=[i for i in df2.columns if df2[i].dtype=='O']
print(categorical_new)
```

```
['Age_band_of_driver', 'Vehicle_driver_relation', 'Driving_experience', 'Are
a_accident_occured', 'Lanes_or_Medians', 'Types_of_Junction', 'Road_surface_
type', 'Light_conditions', 'Weather_conditions', 'Type_of_collision', 'Vehic
le_movement', 'Casualty_class', 'Age_band_of_casualty', 'Cause_of_accident',
'Accident_severity']
```

In [90]:
```
for i in categorical_new:
    print(df2[i].value_counts())
```

```
18-30       4271
31-50       4087
Over 51     1585
Unknown     1548
Under 18     825
Name: Age_band_of_driver, dtype: int64
Employee    10206
Owner        1973
Other         123
Unknown        14
Name: Vehicle_driver_relation, dtype: int64
5-10yr        4192
2-5yr         2613
Above 10yr    2262
1-2yr         1756
Below 1yr     1342
No Licence     118
unknown         33
Name: Driving_experience, dtype: int64
Other                                  4058
```

In [91]:
```
dummy=pd.get_dummies(df2[['Age_band_of_driver', 'Vehicle_driver_relation', 'Dr
                          'Area_accident_occured', 'Lanes_or_Medians', 'Types_
                          'Light_conditions', 'Weather_conditions', 'Type_of_c
                          'Casualty_class', 'Age_band_of_casualty', 'Cause_of_
dummy.head()
```

Out[91]:

	Age_band_of_driver_31-50	Age_band_of_driver_Over 51	Age_band_of_driver_Under 18	Age_band_of_d
0	0	0	0	
1	1	0	0	
2	0	0	0	
3	0	0	0	
4	0	0	0	

5 rows × 102 columns

In [92]:
```
df3=pd.concat([df2,dummy],axis=1)
df3.head()
```

Out[92]:

	Age_band_of_driver	Vehicle_driver_relation	Driving_experience	Area_accident_occured	Lanes
0	18-30	Employee	1-2yr	Residential areas	Tw wi
1	31-50	Employee	Above 10yr	Office areas	Undi
2	18-30	Employee	1-2yr	Recreational areas	
3	18-30	Employee	5-10yr	Office areas	
4	18-30	Employee	2-5yr	Industrial areas	

5 rows × 119 columns

In [93]:
```
#dropping dummied columns
df3.drop(['Age_band_of_driver', 'Vehicle_driver_relation', 'Driving_experience
          'Types_of_Junction', 'Road_surface_type', 'Light_conditions', 'Weath
          'Vehicle_movement','Casualty_class', 'Age_band_of_casualty', 'Cause_
df3.head()
```

Out[93]:

	Number_of_vehicles_involved	Number_of_casualties	Accident_severity	Age_band_of_driver_31 5
0	2	2	Slight Injury	
1	2	2	Slight Injury	
2	2	2	Serious Injury	
3	2	2	Slight Injury	
4	2	2	Slight Injury	

5 rows × 105 columns

Seperating Independent and Dependent

In [94]:
```
x=df3.drop(['Accident_severity'],axis=1)
x.shape
```

Out[94]: (12316, 104)

In [95]:
```
x.head()
```

Out[95]:

	Number_of_vehicles_involved	Number_of_casualties	Age_band_of_driver_31-50	Age_band_of_dr
0	2	2	0	
1	2	2	1	
2	2	2	0	
3	2	2	0	
4	2	2	0	

5 rows × 104 columns

In [96]:
```
y=df3.iloc[:,2]
y.head()
```

Out[96]:
```
0     Slight Injury
1     Slight Injury
2    Serious Injury
3     Slight Injury
4     Slight Injury
Name: Accident_severity, dtype: object
```

In [97]:
```
y.value_counts()
```

Out[97]:
```
Slight Injury     10415
Serious Injury     1743
Fatal injury        158
Name: Accident_severity, dtype: int64
```

Splitting the data

KNN Model Creation

Prediction

In [98]:
```
from sklearn.model_selection import train_test_split

x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.2, rando
from sklearn.neighbors import KNeighborsClassifier

model_KNN = KNeighborsClassifier(n_neighbors=5)
model_KNN.fit(x_train, y_train)
```

Out[98]:
```
KNeighborsClassifier()
```

In [99]:
```
y_pred=model_KNN.predict(x_test)
```

In [100]:
```
y_pred
```

Out[100]:
```
array(['Slight Injury', 'Slight Injury', 'Slight Injury', ...,
       'Slight Injury', 'Slight Injury', 'Slight Injury'], dtype=object)
```

Checking Accuracy, Classification Report, Confusion Matrix

In [101]:
```
from sklearn.metrics import confusion_matrix,classification_report,accuracy_sc
```

Classification Report

In [102]:
```
report_KNN=classification_report(y_test,y_pred)
print(report_KNN)
```

```
                precision    recall  f1-score   support

  Fatal injury       0.00      0.00      0.00        37
Serious Injury       0.23      0.04      0.07       363
 Slight Injury       0.84      0.98      0.90      2064

      accuracy                           0.82      2464
     macro avg       0.36      0.34      0.32      2464
  weighted avg       0.74      0.82      0.77      2464
```

Accuracy Score

In [103]:
```
accuracy_KNN=accuracy_score(y_test,y_pred)
print(accuracy_KNN)
```

```
0.8238636363636364
```

Confusion Matrix

In [104]:
```
matrix_KNN=confusion_matrix(y_test,y_pred)
print(matrix_KNN,'\n')
print(ConfusionMatrixDisplay.from_predictions(y_test,y_pred))
```

```
[[   0    2   35]
 [   2   15  346]
 [   0   49 2015]]

<sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay object at 0x0
00002346F4E5A00>
```