Importing Libraries

In [38]:
```
import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns

# to suppress warnings
from warnings import filterwarnings
filterwarnings('ignore')

# Display all columns
pd.options.display.max_columns = None
```

File Loading

In [39]:
```
df = pd.read_csv('Uk Accident Project/UK_Accident.csv')
df.head(5)
```

Out[39]:

	Unnamed: 0	Accident_Index	Location_Easting_OSGR	Location_Northing_OSGR	Longitu
0	0	200501BS00001	525680	178240	
1	1	200501BS00002	524170	181650	
2	2	200501BS00003	524520	182240	
3	3	200501BS00004	526900	177530	
4	4	200501BS00005	528060	179040	

Summary of Dataset

- The Data is about UK Road Accident from 2005 - 2014.
- There are a total of 33 columns, some of the key columns are:

  • Accident_Index - PRIMARY KEY

  • Locations - Longitude & Longitude

- Accident_Severity - Seriousness of the Crash
- Number_of_Vehicles - How many Vechiles involvement in the incident.
- LSOA_of_Accident_Location - Exact location of the incident.

In [40]:
```
df.shape
```

Out[40]: (1504150, 33)

In [41]:
```
df.describe()
```

Out[41]:

	Unnamed: 0	Location_Easting_OSGR	Location_Northing_OSGR	Longitude	Latitude
count	1504150	1504049	1504150	1504049	150415(
mean	253043	439621	300138	-1	5:
std	148916	95116	161022	1	2
min	0	64950	0	-8	(
25%	125345	375060	178260	-2	51
50%	250691	439960	268800	-1	52
75%	376037	523060	398150	-0	5:
max	570010	655370	1208800	2	61

In [42]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1504150 entries, 0 to 1504149
Data columns (total 33 columns):
 #   Column                                       Non-Null Count    Dtype  
---  ------                                       --------------    -----  
 0   Unnamed: 0                                   1504150 non-null  int64  
 1   Accident_Index                               1504150 non-null  object 
 2   Location_Easting_OSGR                        1504049 non-null  float64
 3   Location_Northing_OSGR                       1504150 non-null  float64
 4   Longitude                                    1504049 non-null  float64
 5   Latitude                                     1504150 non-null  float64
 6   Police_Force                                 1504150 non-null  int64  
 7   Accident_Severity                            1504150 non-null  int64  
 8   Number_of_Vehicles                           1504150 non-null  int64  
 9   Number_of_Casualties                         1504150 non-null  int64  
 10  Date                                         1504150 non-null  object 
 11  Day_of_Week                                  1504150 non-null  int64  
 12  Time                                         1504033 non-null  object 
 13  Local_Authority_(District)                   1504150 non-null  int64  
 14  Local_Authority_(Highway)                    1504150 non-null  object 
 15  1st_Road_Class                               1504150 non-null  int64  
 16  1st_Road_Number                              1504150 non-null  int64  
 17  Road_Type                                    1504150 non-null  object 
 18  Speed_limit                                  1504150 non-null  int64  
 19  Junction_Control                             901315 non-null   object 
 20  2nd_Road_Class                               1504150 non-null  int64  
 21  2nd_Road_Number                              1504150 non-null  int64  
 22  Pedestrian_Crossing-Human_Control            1504133 non-null  object 
 23  Pedestrian_Crossing-Physical_Facilities      1504116 non-null  object 
 24  Light_Conditions                             1504150 non-null  object 
 25  Weather_Conditions                           1504150 non-null  object 
 26  Road_Surface_Conditions                      1504150 non-null  object 
 27  Special_Conditions_at_Site                   36582 non-null    object 
 28  Carriageway_Hazards                          27250 non-null    object 
 29  Urban_or_Rural_Area                          1504150 non-null  int64  
 30  Did_Police_Officer_Attend_Scene_of_Accident  1504150 non-null  object 
 31  LSOA_of_Accident_Location                    1395912 non-null  object 
 32  Year                                         1504150 non-null  int64  
dtypes: float64(4), int64(14), object(15)
memory usage: 378.7+ MB
```

Fixing Wrong Data Type

In [43]:
```
# Time is in object, changing it to datetime
df['Time'] = pd.to_datetime(df['Time']).dt.time
```

Duplicates

In [44]:
```
df.duplicated().sum()
```

Out[44]: 0

Fixing Null Value

In [45]:
```
df.isna().sum() # This will give no of Null values present in each columns
```

Out[45]:
```
Unnamed: 0                                      0
Accident_Index                                  0
Location_Easting_OSGR                         101
Location_Northing_OSGR                          0
Longitude                                     101
Latitude                                        0
Police_Force                                    0
Accident_Severity                               0
Number_of_Vehicles                              0
Number_of_Casualties                            0
Date                                            0
Day_of_Week                                     0
Time                                          117
Local_Authority_(District)                      0
Local_Authority_(Highway)                       0
1st_Road_Class                                  0
1st_Road_Number                                 0
Road_Type                                       0
Speed_limit                                     0
Junction_Control                           602835
2nd_Road_Class                                  0
2nd_Road_Number                                 0
Pedestrian_Crossing-Human_Control              17
Pedestrian_Crossing-Physical_Facilities        34
Light_Conditions                                0
Weather_Conditions                              0
Road_Surface_Conditions                         0
Special_Conditions_at_Site                1467568
Carriageway_Hazards                       1476900
Urban_or_Rural_Area                             0
Did_Police_Officer_Attend_Scene_of_Accident     0
LSOA_of_Accident_Location                  108238
Year                                            0
dtype: int64
```

So there are some Null values present in the Dataset. In general there are 3 common ways to fix it:

- Filling the Missing Values – Imputation
- Deleting the row/column with missing data
- Filling with a Regression Model

In [46]:
```
# To fix the null values of "Carriageway_Hazards" , "Special_Conditions_at_Site"
# I'll go with deleting the columns because more than 90% of the data is null.

df.drop(['Special_Conditions_at_Site','Carriageway_Hazards','Junction_Control'],
```

Using Imputation method to fix null values of "Pedestrian_Crossing-

Human_Control" , "Pedestrian_Crossing-Physical_Facilities" , "Time" , "Longitude" , "Location_Easting_OSGR"

In [47]:
```
# "Location_Easting_OSGR"  PART - 1

# Group by 'Longitude' and find the most common non-null value for each group
most_common_per_longitude = df[df['LSOA_of_Accident_Location'].notnull()]\
    .groupby('Longitude')['LSOA_of_Accident_Location'].agg(lambda x: x.value_cou

# Create a dictionary to map 'Longitude' to the most common 'LSOA_of_Accident_Lo
common_mapping = most_common_per_longitude.to_dict()

# Replace null values in 'LSOA_of_Accident_Location' based on 'Longitude'
df['LSOA_of_Accident_Location'] = df.apply(
    lambda row: common_mapping.get(row['Longitude'], row['LSOA_of_Accident_Locat
)
```

In [48]:
```
# "Location_Easting_OSGR"  PART - 2

# Group by 'Longitude' and find the most common non-null value for each group
most_common_per_Latitudes = df[df['LSOA_of_Accident_Location'].notnull()]\
    .groupby('Latitude')['LSOA_of_Accident_Location'].agg(lambda x: x.value_coun

# Create a dictionary to map 'Longitude' to the most common 'LSOA_of_Accident_Lo
common_mappings = most_common_per_Latitudes.to_dict()

# Replace null values in 'LSOA_of_Accident_Location' based on 'Longitude'
df['LSOA_of_Accident_Location'] = df.apply(
    lambda row: common_mappings.get(row['Latitude'], row['LSOA_of_Accident_Locat
)
```

In [49]:
```
df.dropna(subset=['LSOA_of_Accident_Location'], inplace=True)
```

- The Logic behind the above code was to Save as much data as possible from "Location_Easting_OSGR" because it is one of the important columns for analysis.
- What exactly I did is by replacing them with the most common non-null value for each 'Longitude and did same with Latitude.

This code should replace the null values with the most common 'LSOA_of_Accident_Location' for each 'Longitude' & 'Latitude'.

In [ ]:
```

```

In [50]:
```
#'Pedestrian_Crossing-Physical_Facilities'

df['Pedestrian_Crossing-Physical_Facilities'].value_counts()
```

Out[50]:
```
Pedestrian_Crossing-Physical_Facilities
No physical crossing within 50 meters          1173066
Pedestrian phase at traffic signal junction      93857
non-junction pedestrian crossing                 73274
Zebra crossing                                   39060
Central refuge                                   26619
Footbridge or subway                              4170
Name: count, dtype: int64
```

In [51]:
```
df['Pedestrian_Crossing-Physical_Facilities'].fillna('No physical crossing withi
```

In [ ]:

In [52]:
```
#'Pedestrian_Crossing-Human_Control'

df['Pedestrian_Crossing-Human_Control'].value_counts()
```

Out[52]:
```
Pedestrian_Crossing-Human_Control
None within 50 metres                 1403255
Control by other authorised person       3612
Control by school crossing patrol        3194
Name: count, dtype: int64
```

In [53]:
```
df['Pedestrian_Crossing-Human_Control'].fillna('None within 50 metres ', inplace
```

In [ ]:

In [54]:
```
#Time

df['Time'].value_counts()
```

Out[54]:
```
Time
17:00:00    13259
17:30:00    12776
16:00:00    12026
15:30:00    12025
18:00:00    11994
            ...
04:16:00       47
04:01:00       47
04:34:00       46
04:41:00       45
04:46:00       42
Name: count, Length: 1439, dtype: int64
```

In [55]:
```
df['Time'].fillna('17:00', inplace = True)
```

In [ ]:

In [56]:
```
df.isna().sum()
```

Out[56]:
```
Unnamed: 0                                      0
Accident_Index                                  0
Location_Easting_OSGR                           0
Location_Northing_OSGR                          0
Longitude                                       0
Latitude                                        0
Police_Force                                    0
Accident_Severity                               0
Number_of_Vehicles                              0
Number_of_Casualties                            0
Date                                            0
Day_of_Week                                     0
Time                                            0
Local_Authority_(District)                      0
Local_Authority_(Highway)                       0
1st_Road_Class                                  0
1st_Road_Number                                 0
Road_Type                                       0
Speed_limit                                     0
2nd_Road_Class                                  0
2nd_Road_Number                                 0
Pedestrian_Crossing-Human_Control               0
Pedestrian_Crossing-Physical_Facilities         0
Light_Conditions                                0
Weather_Conditions                              0
Road_Surface_Conditions                         0
Urban_or_Rural_Area                             0
Did_Police_Officer_Attend_Scene_of_Accident     0
LSOA_of_Accident_Location                       0
Year                                            0
dtype: int64
```

All the Nulls are fixed.

In [ ]:

Outlier

I'm not removing the Outliers from the Data because this dataset is related to road accidents, so every incident is important to help control the crashes and will give a brief analysis of the locations.

In [ ]:

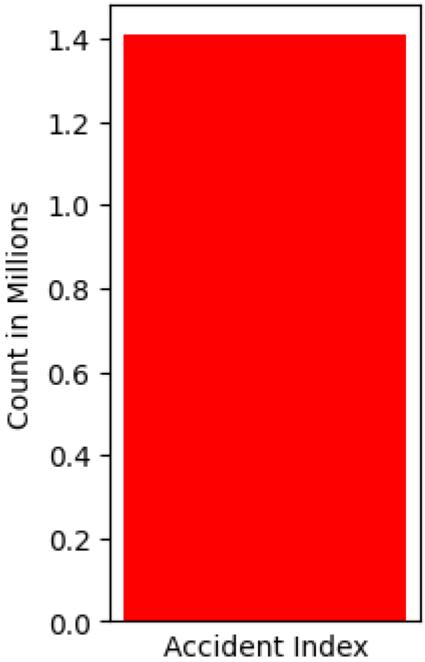
Irrelevant Columns

In [57]:
```
df.drop(['2nd_Road_Number','Unnamed: 0','1st_Road_Class','1st_Road_Number','2nd_
```

In [ ]:

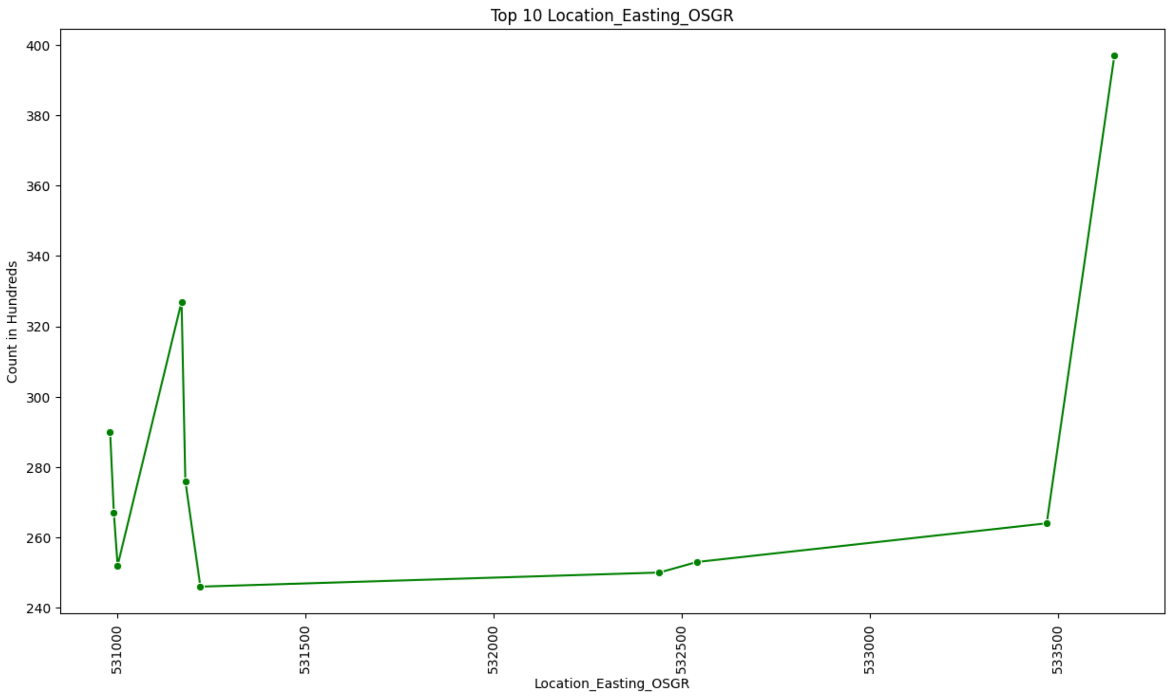
Univarate Analysis

In [58]:
```
count_index = df['Accident_Index'].count()
height = count_index / 1000000
plt.figure(figsize=(2, 4))  # Adjust the figure size
plt.bar(x='Count',height=height, width=0.01, color='red')
plt.xticks([])
plt.xlabel('Accident Index')
plt.ylabel('Count in Millions')
plt.show()
print("Total Accident",count_index)
```

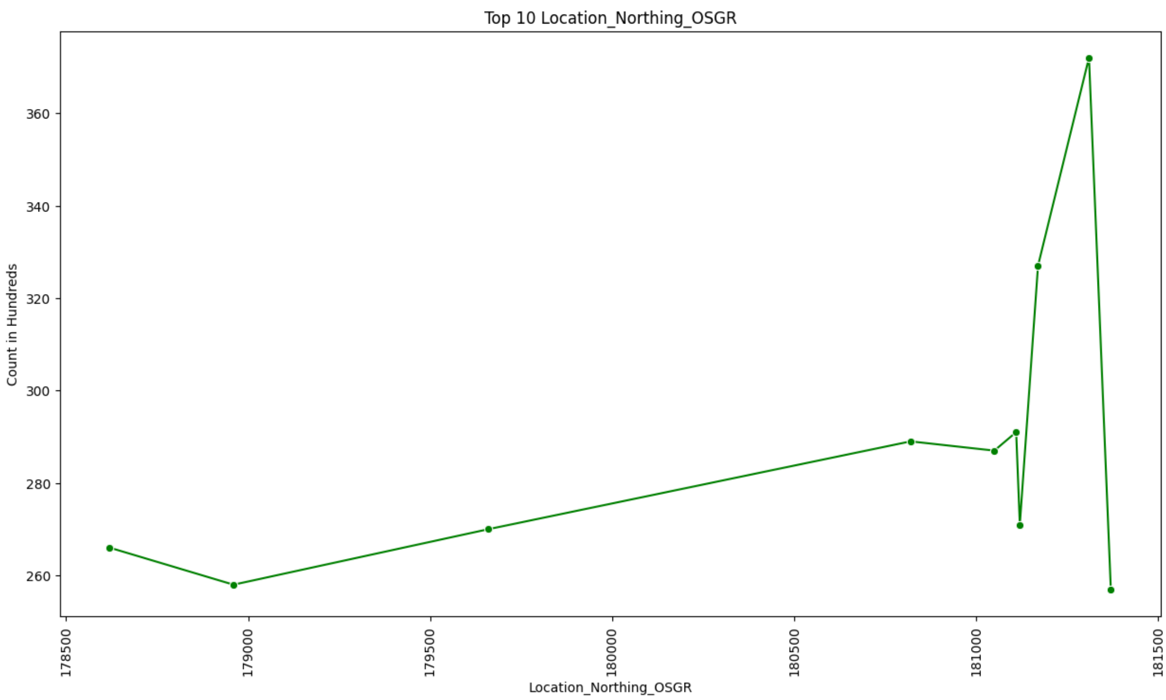



```
Total Accident 1410077
```

In [59]:
```
top10_easting = df.Location_Easting_OSGR.value_counts(ascending=False).head(10)

sns.lineplot(data=top10_easting, marker='o', color='green')

plt.title('Top 10 Location_Easting_OSGR')
plt.ylabel('Count in Hundreds')
plt.xticks(rotation=90)

plt.show()
```



In [60]:
```
top10_Northing = df.Location_Northing_OSGR.value_counts(ascending=False).head(10

sns.lineplot(data=top10_Northing, marker='o', color='green')

plt.title('Top 10 Location_Northing_OSGR')
plt.ylabel('Count in Hundreds')
plt.xticks(rotation=90)

plt.show()
```

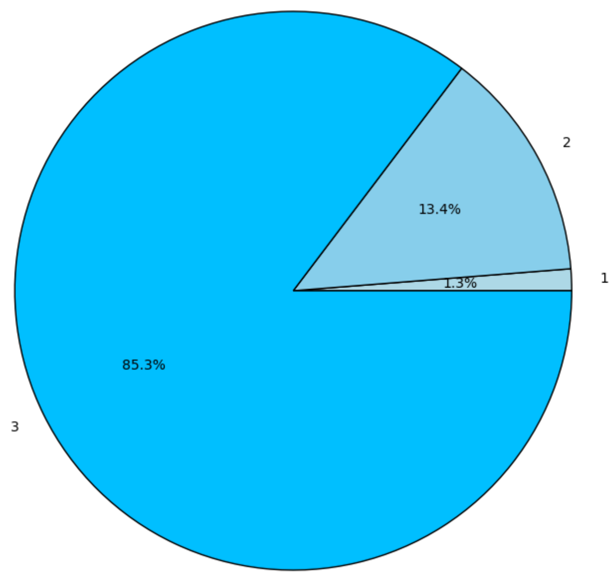


- Location_Easting_OSGR - Easting coordinates indicate their horizontal position in the east-west direction.
  - 533500 is the highest point where accidents happen.

- Location_Northing_OSGR - Northing coordinates indicate their vertical position in the north-south direction.
    - 181000 to 181500 are red zone.

Distrubition Of Severity

In [61]:
```
sev = df.groupby('Accident_Severity')['Accident_Index'].count()

colors = ['lightblue', 'skyblue', 'deepskyblue']
wedgeprops = {'edgecolor': 'black'}

plt.pie(sev.values,labels=sev.index,autopct='%1.1f%%',colors=colors, wedgeprops=

plt.axis('equal')

plt.show()
```

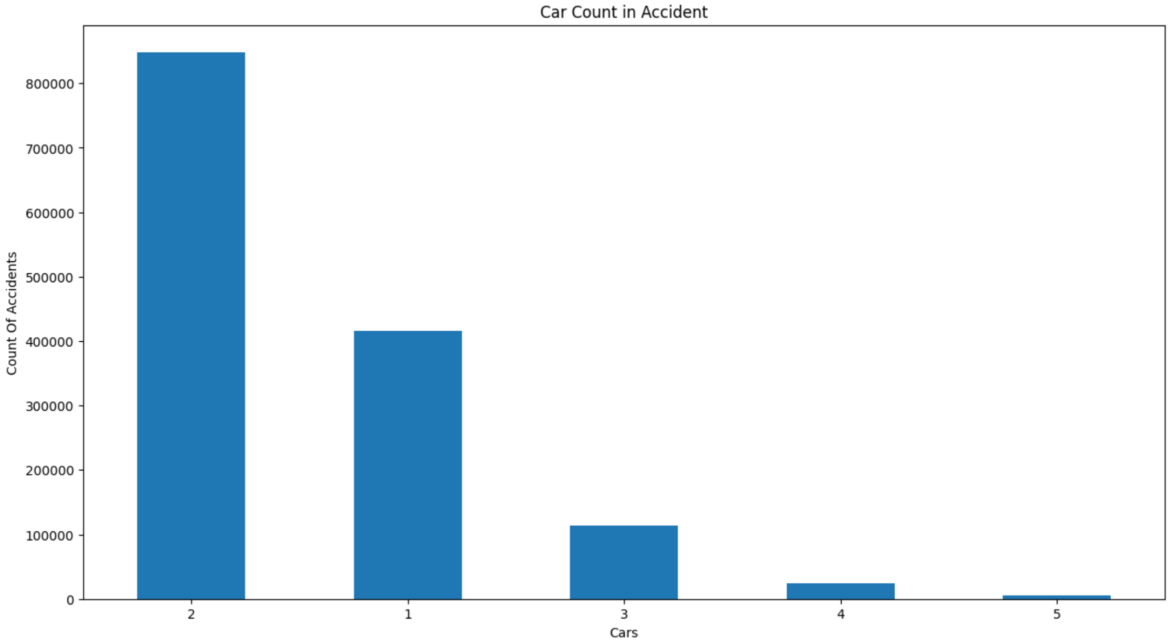


3 - Fatal Casualties || 2 - Serious Casualties || 1 - Minor Casualties

The majority of accidents occurring are of a more severe nature.

Vehicle Count in Accident

In [62]:
```
n_cars = df.groupby('Number_of_Vehicles')['Accident_Index'].count().sort_values(

n_cars.plot.bar(xlabel = 'Cars',ylabel = 'Count Of Accidents')

plt.title('Car Count in Accident')

plt.xticks(rotation=0)

plt.show()
```

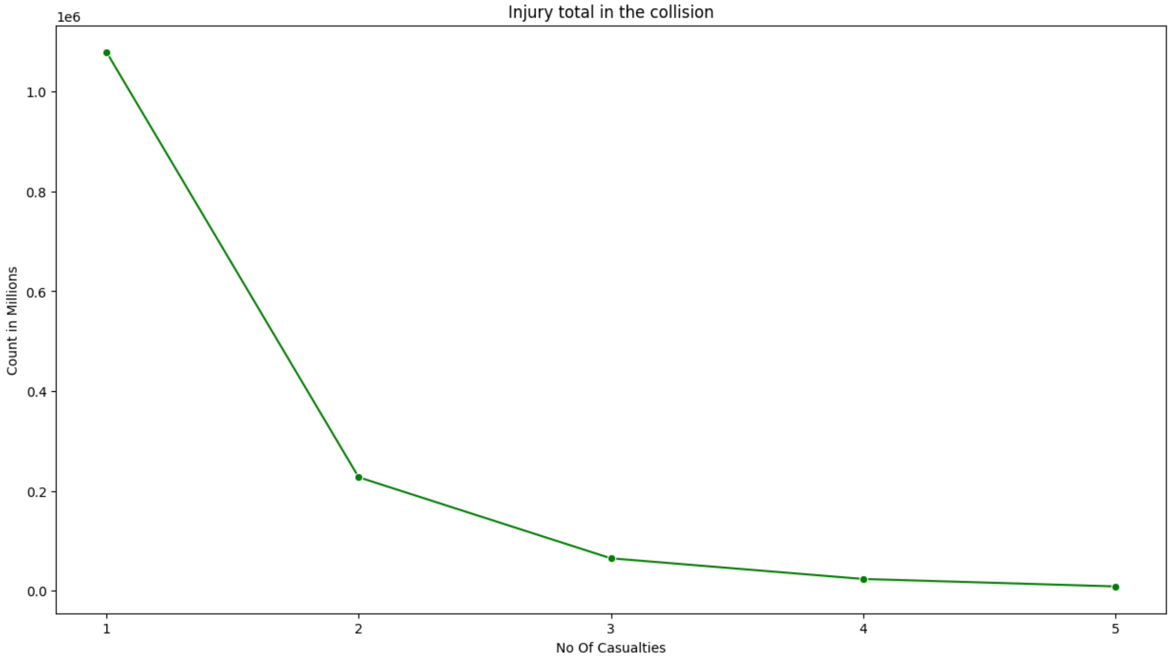


More frequent interaction of two vehicles.

Total injuries in the accident

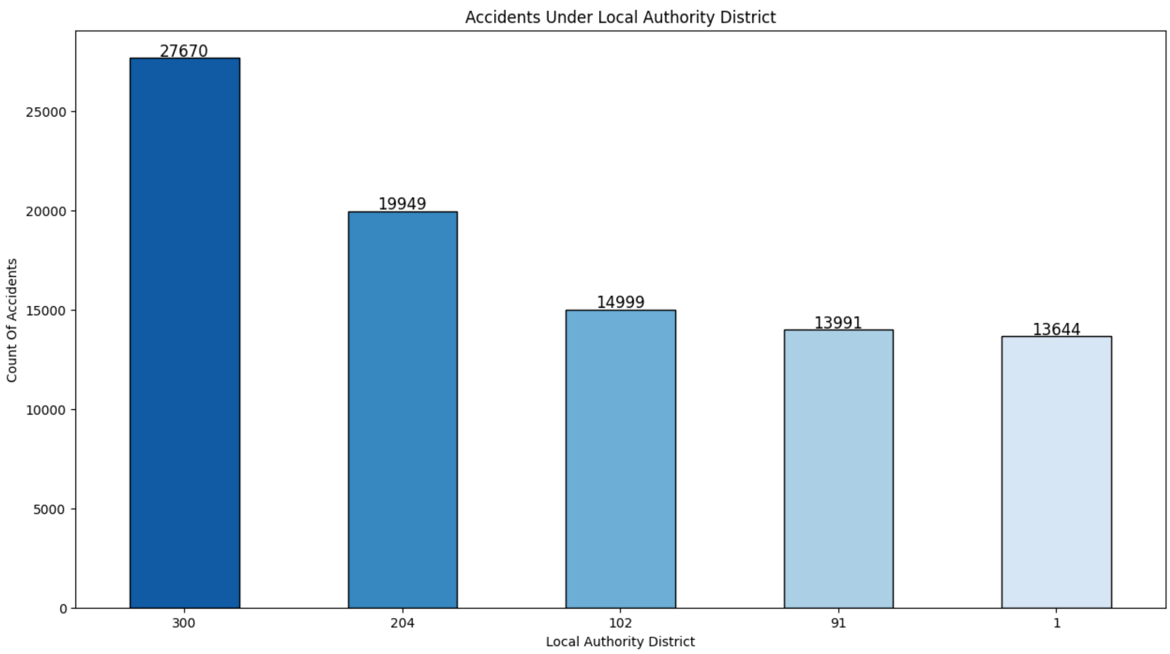
In [209…

```
n_casu = df.groupby('Number_of_Casualties')['Accident_Index'].count().sort_value

sns.lineplot(data = n_casu,marker ='o',color='green')

plt.xlabel('No Of Casualties')

plt.ylabel('Count in Millions')

plt.title('Injury total in the collision')

# Set the x-axis ticks to integers (whole numbers)
plt.xticks(list(map(int, n_casu.index)))

plt.show()
```

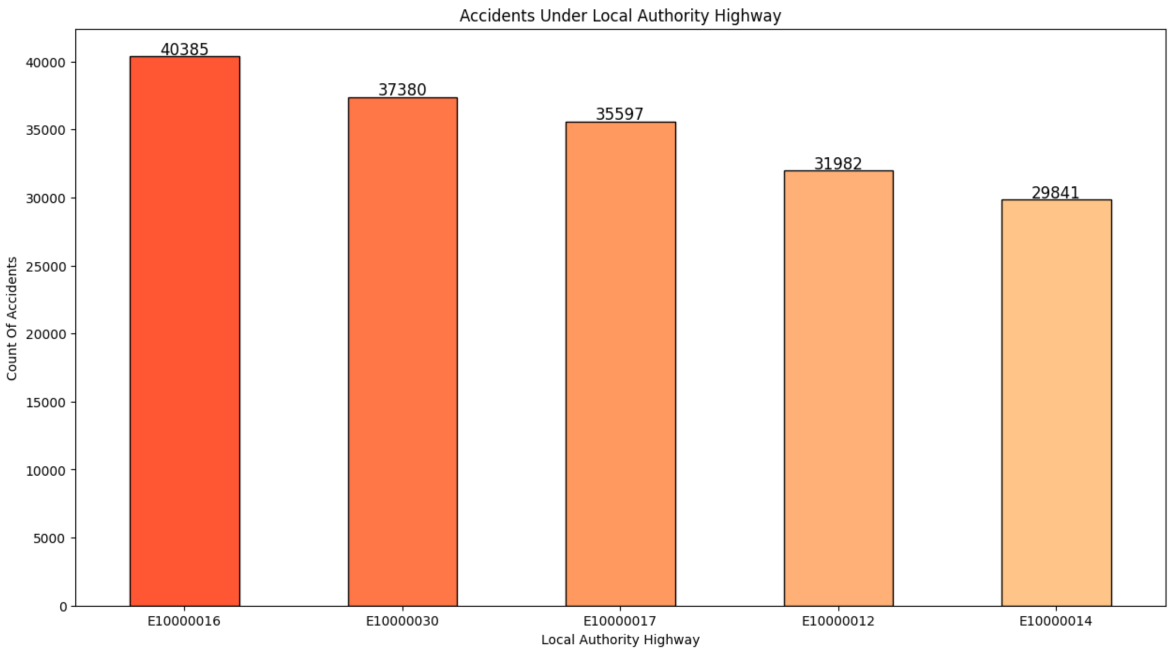


Occurrences of single-person involvement are higher.

In [64]:
```
district = df.groupby('Local_Authority_(District)')['Accident_Index'].count().sc

custom_palette = sns.color_palette("Blues_r", n_colors=len(district))

ax = district.plot(kind='bar', color=custom_palette, edgecolor='black')

plt.xlabel('Local Authority District')
plt.ylabel('Count Of Accidents')
plt.title('Accidents Under Local Authority District')

for p in ax.patches:
    ax.annotate(f'{p.get_height()}', (p.get_x() + p.get_width() / 2., p.get_heig

plt.xticks(rotation=0)
plt.show()
```



In [65]:
```
highway = df.groupby('Local_Authority_(Highway)')['Accident_Index'].count().sort

custom_palette = ["#FF5733", "#FF7746", "#FF9960", "#FFB077", "#FFC588"]

ay = highway.plot(kind='bar', color=custom_palette, edgecolor='black')

plt.xlabel('Local Authority Highway')
plt.ylabel('Count Of Accidents')
plt.title('Accidents Under Local Authority Highway')

for p in ay.patches:
    ay.annotate(f'{p.get_height()}', (p.get_x() + p.get_width() / 2., p.get_heig

plt.xticks(rotation=0)
plt.show()
```



- Local_Authority_(District) - Local_Authority_(District) is the district's governing authority.

  • 300 & 204 are the highest.

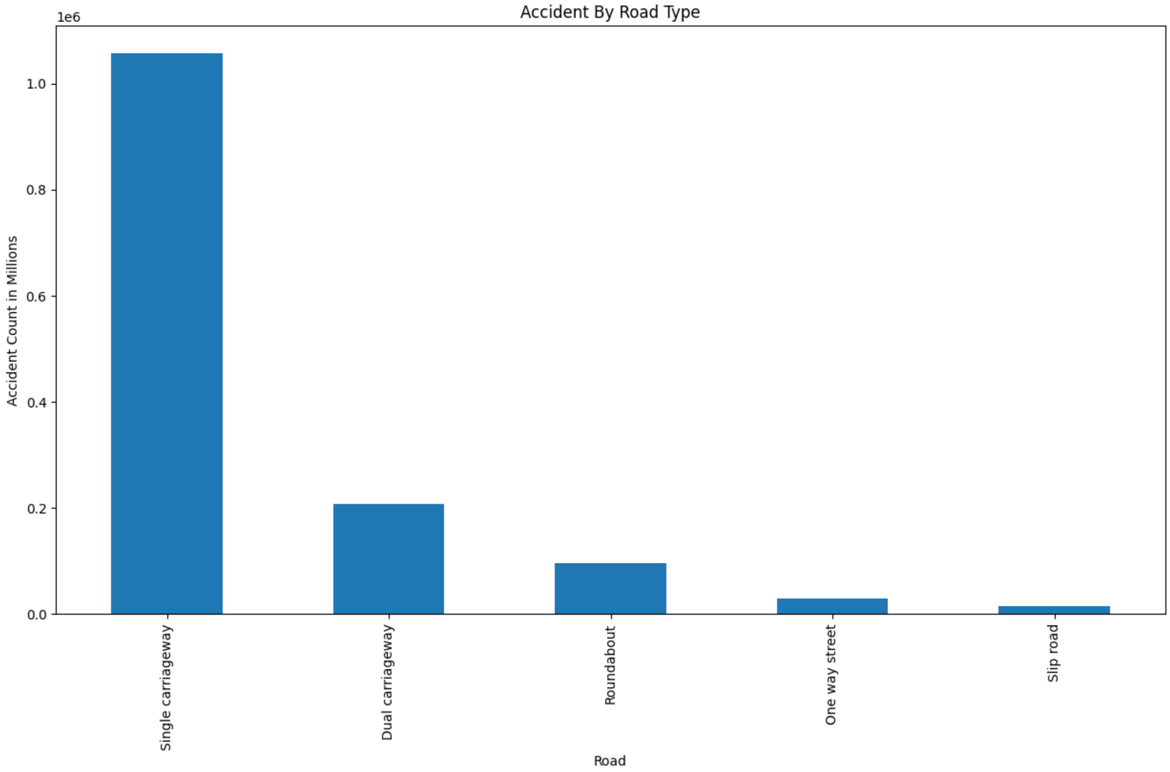
- Local_Authority_(Highway) - Local_Authority_(Highway) manages highways.

  • E1000016 & E1000030 have higher numbers.

Road Type

In [66]:
```
Road = df.groupby('Road_Type')['Accident_Index'].count().sort_values(ascending =
```

In [67]:
```
Road.plot.bar(xlabel = 'Road',ylabel = 'Accident Count in Millions')

plt.title('Accident By Road Type')

plt.show()
```



Single carriageway have larger numbers.

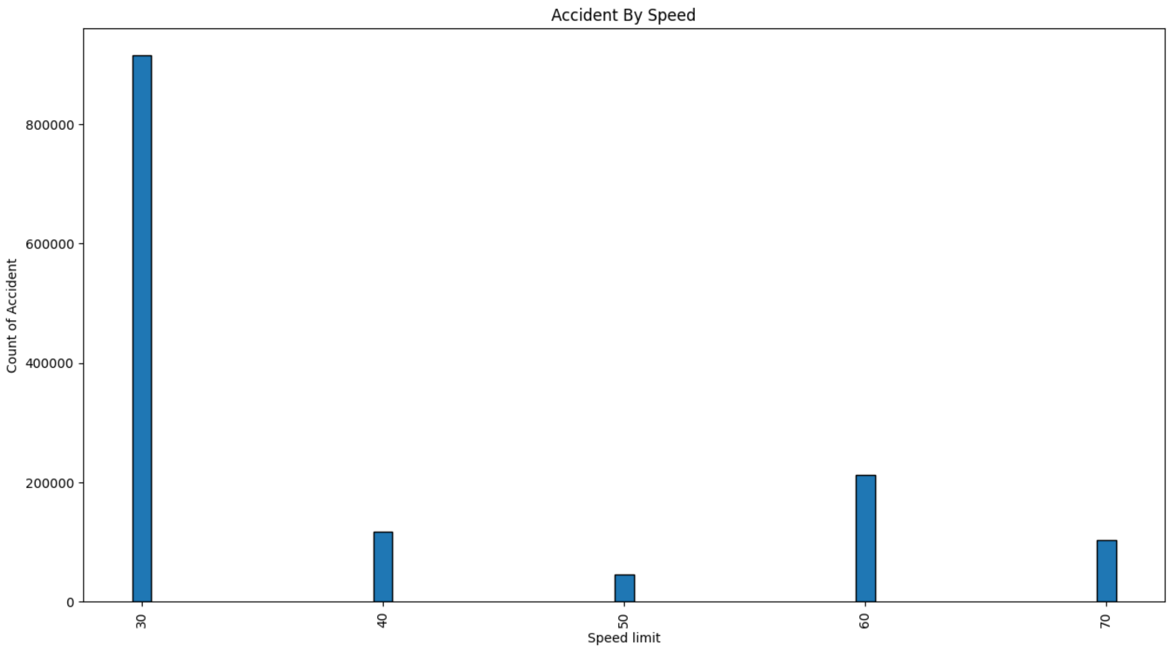
Speed Limit

In [68]:

```
spd = df.groupby('Speed_limit')['Accident_Index'].count().sort_values(ascending
```

In [114…

```
plt.bar(spd.index, spd, edgecolor='black')
plt.xlabel('Speed limit')
plt.ylabel('Count of Accident')
plt.title('Accident By Speed')
plt.xticks(rotation=90)
plt.xticks([int(x) for x in spd.index], rotation=90)

plt.show()
```

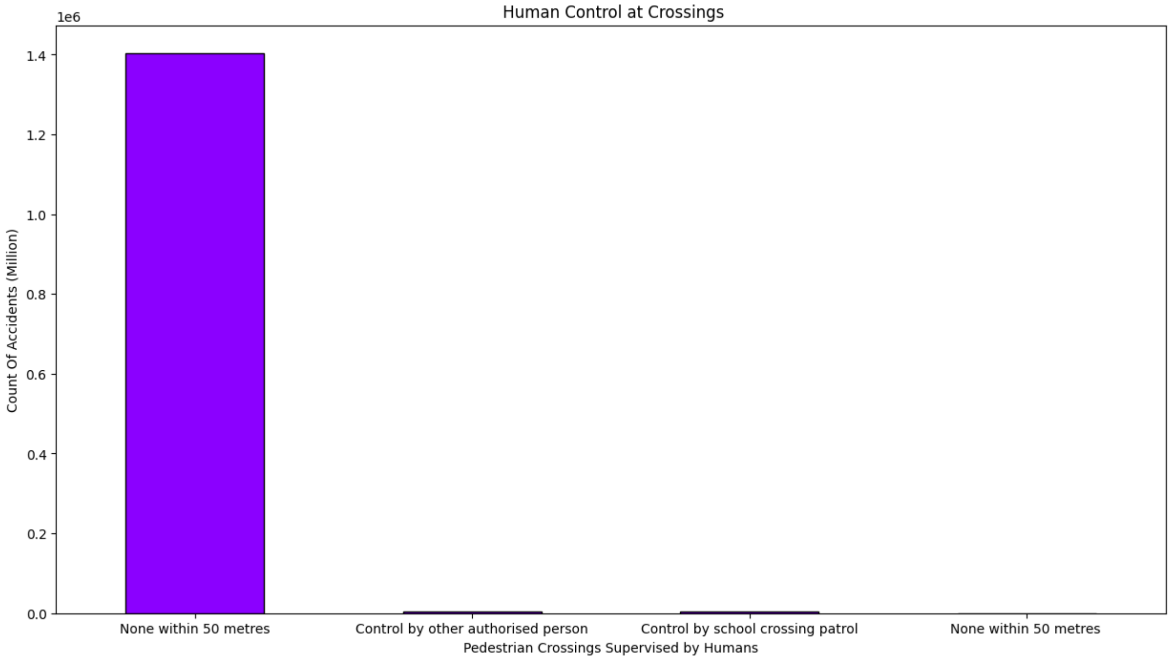


30 & 60 are where most of the accidents happen.

Pedestrian Crossing

In [71]:
```
pchc = df.groupby('Pedestrian_Crossing-Human_Control')['Accident_Index'].count()
```

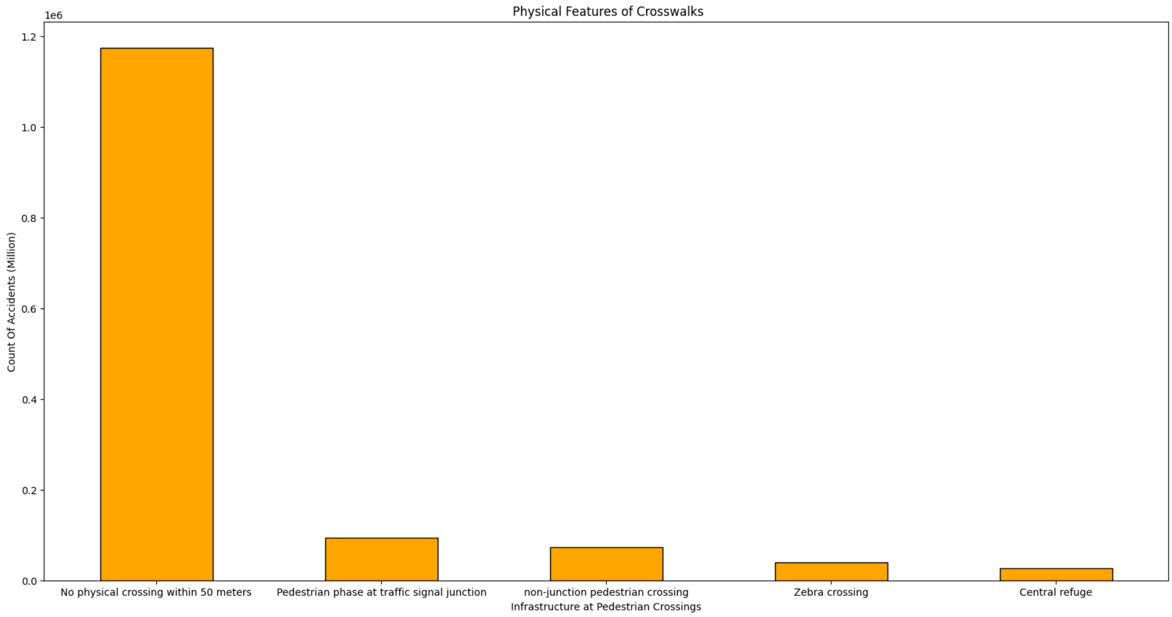
In [75]:
```
pchc = df.groupby('Pedestrian_Crossing-Human_Control')['Accident_Index'].count()

custom_palette = ["#8B00FF", "#8B00FF", "#8B00FF", "#8B00FF", "#8B00FF"]

az = pchc.plot(kind='bar', color=custom_palette, edgecolor='black')

plt.xlabel('Pedestrian Crossings Supervised by Humans')
plt.ylabel('Count Of Accidents (Million)')
plt.title(' Human Control at Crossings')

for p in az.patches:
    ay.annotate(f'{p.get_height()}', (p.get_x() + p.get_width() / 2., p.get_heig

plt.xticks(rotation=0)
plt.show()
```



None within 50 meters is the highest in number.

In [77]:
```
pcpc = df.groupby('Pedestrian_Crossing-Physical_Facilities')['Accident_Index'].c
```

In [85]:
```
pcpc = df.groupby('Pedestrian_Crossing-Physical_Facilities')['Accident_Index'].c

plt.figure(figsize=(20,10))

custom_palette = ["#FFA500", "#FFA500", "#FFA500", "#FFA500", "#FFA500"]

aa = pcpc.plot(kind='bar', color=custom_palette, edgecolor='black')

plt.xlabel('Infrastructure at Pedestrian Crossings')
plt.ylabel('Count Of Accidents (Million)')
plt.title(' Physical Features of Crosswalks')

for p in aa.patches:
    ay.annotate(f'{p.get_height()}', (p.get_x() + p.get_width() / 2., p.get_heig

plt.xticks(rotation=0)
plt.show()
```

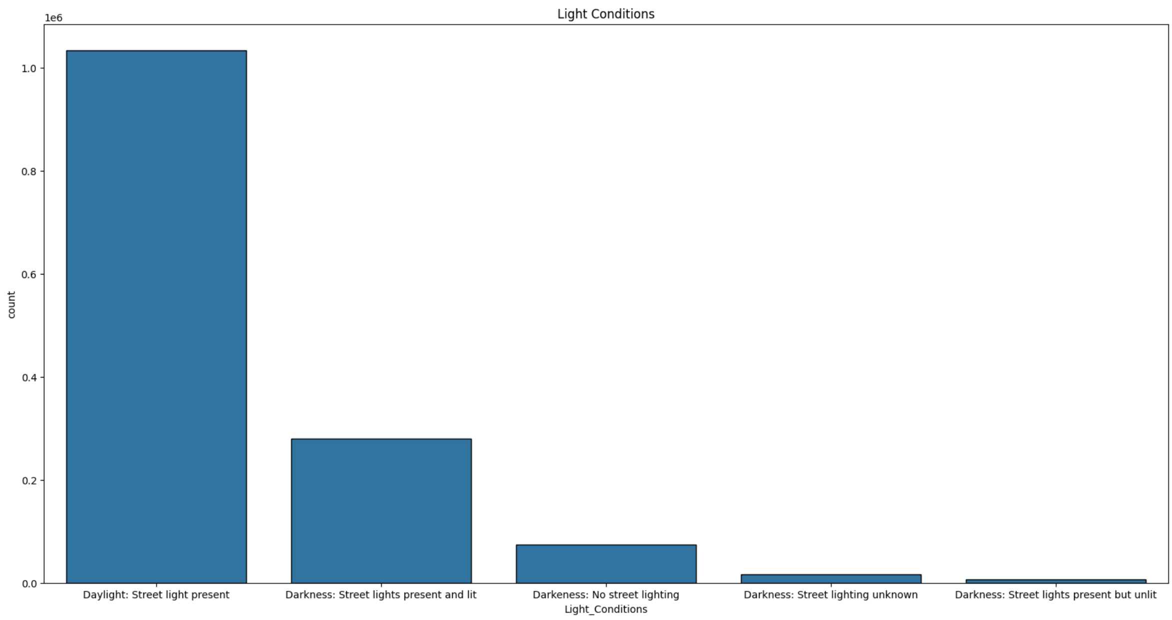


No Physical infrastructure within 50 meters is higher.

Accident By Light Conditions

In [95]:
```
lc = df.Light_Conditions.value_counts(ascending = False)
```

In [93]:
```
plt.figure(figsize=(20,10))
sns.barplot(data = lc, edgecolor = 'black')
plt.title('Light Conditions')
plt.show()
```

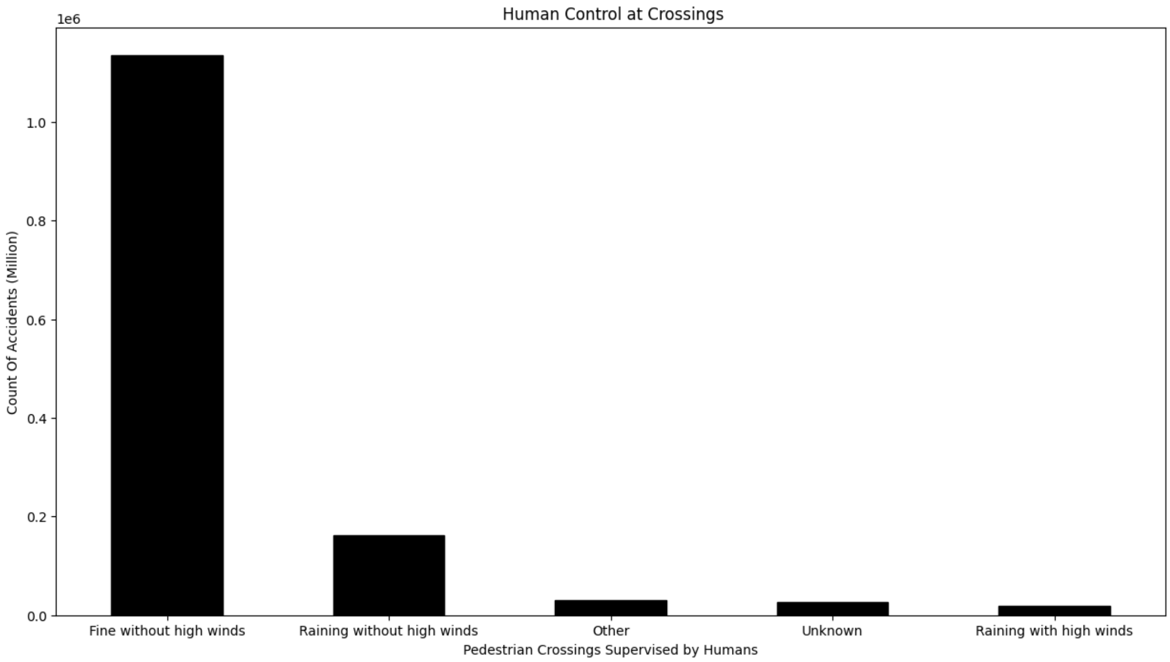


Daylight: Street light present has highest no.

Accident By Weather Conditions

In [100…

```
wc = df.groupby('Weather_Conditions')['Accident_Index'].count().sort_values(asce

custom_palette = ["black"]

ab = wc.plot(kind='bar', color=custom_palette, edgecolor='black')

plt.xlabel('Pedestrian Crossings Supervised by Humans')
plt.ylabel('Count Of Accidents (Million)')
plt.title(' Human Control at Crossings')

for p in ab.patches:
    ay.annotate(f'{p.get_height()}', (p.get_x() + p.get_width() / 2., p.get_heig

plt.xticks(rotation=0)
plt.show()
```



Fine without high winds has highest no.

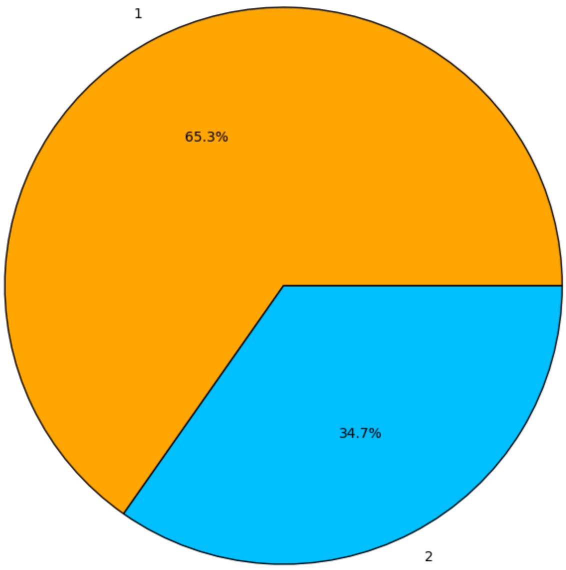
Urban Or Rural Accident Percentage

In [105…

```
ur = df.Urban_or_Rural_Area.value_counts(ascending = False).head(2)
```

In [108…

```
colors = ["orange", 'deepskyblue']
wedgeprops = {'edgecolor': 'black'}

plt.pie(ur.values,labels=ur.index,autopct='%1.1f%%',colors=colors, wedgeprops=we

plt.axis('equal')

plt.show()
```



1 = Urban || 2 = Rural

Accidents in Urban area is higher.

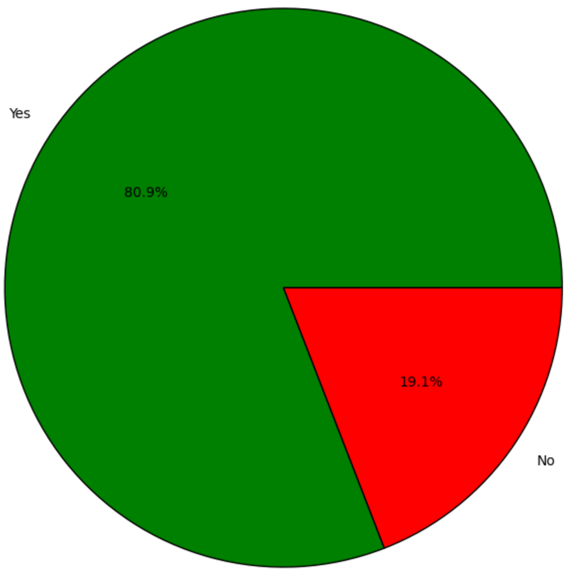
Police Attend Accident

In [112…

```
pa = df.Did_Police_Officer_Attend_Scene_of_Accident.value_counts(ascending = Fal
```

In [131…

```
colors = ["Green", 'Red']
wedgeprops = {'edgecolor': 'black'}

plt.pie(pa.values,labels=pa.index,autopct='%1.1f%%',colors=colors, wedgeprops=we

plt.axis('equal')

plt.show()
```



Attendance of police on the accident spot is 80%.

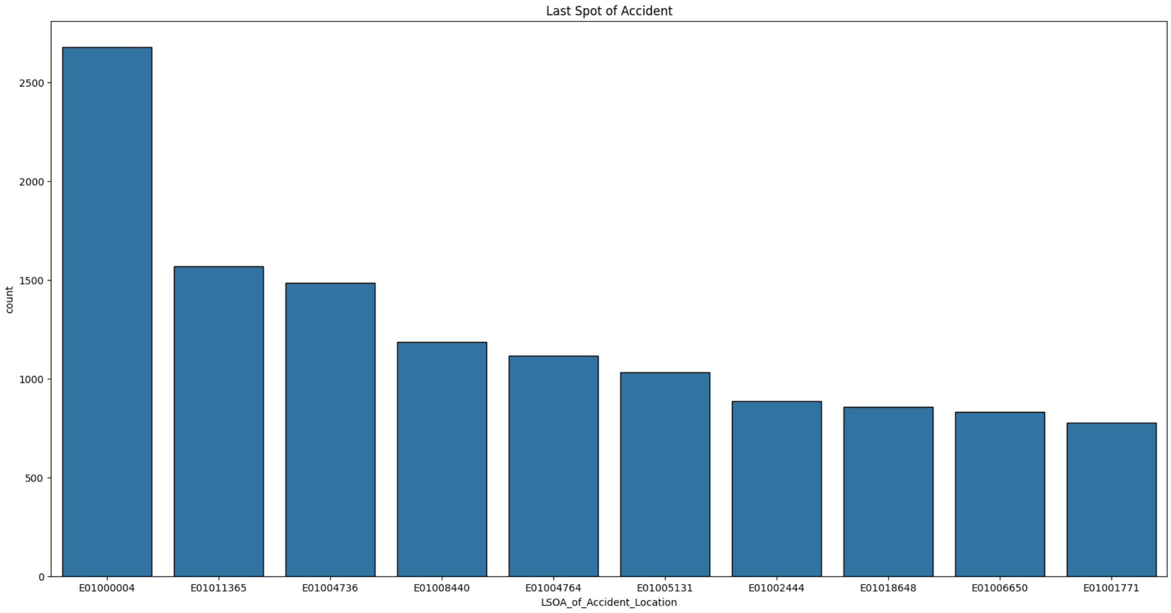
Last Spot of Accident

In [143...

```
lsoa = df.LSOA_of_Accident_Location.value_counts(ascending = False).head(10)
```

In [156...

```
plt.figure(figsize=(20,10))
sns.barplot(data = lsoa, edgecolor = 'black')
plt.title('Last Spot of Accident')
plt.show()
```



E01000004 & E01011365 are highest LSOA.

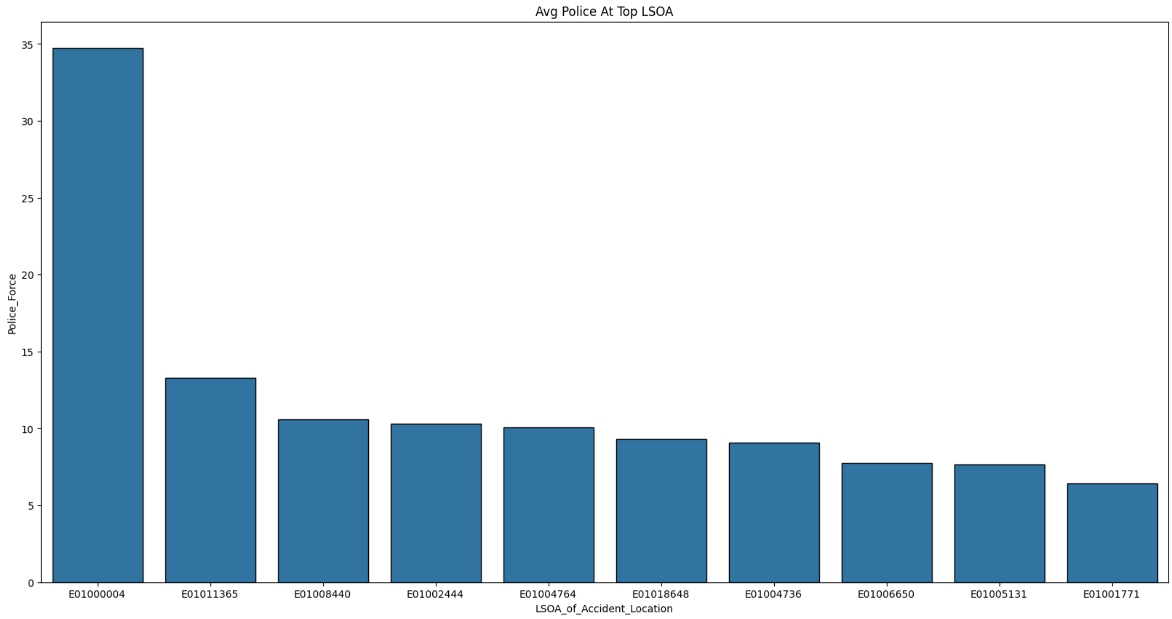
Average Police at Top 10 LSOA

In [154...

```
top_lsoa = df['LSOA_of_Accident_Location'].value_counts().head(10).index

filtered_df = df[df['LSOA_of_Accident_Location'].isin(top_lsoa)]

mean_police_force = filtered_df.groupby('LSOA_of_Accident_Location')['Police_For

top_10_mean_police_force = mean_police_force.sort_values(ascending=False).head(1
```

In [162...

```
plt.figure(figsize=(20,10))
sns.barplot(data = top_10_mean_police_force, edgecolor = 'black')
plt.title('Avg Police At Top LSOA')
plt.show()
```



Average Police E01000004 & E01011365 is 35 & 13

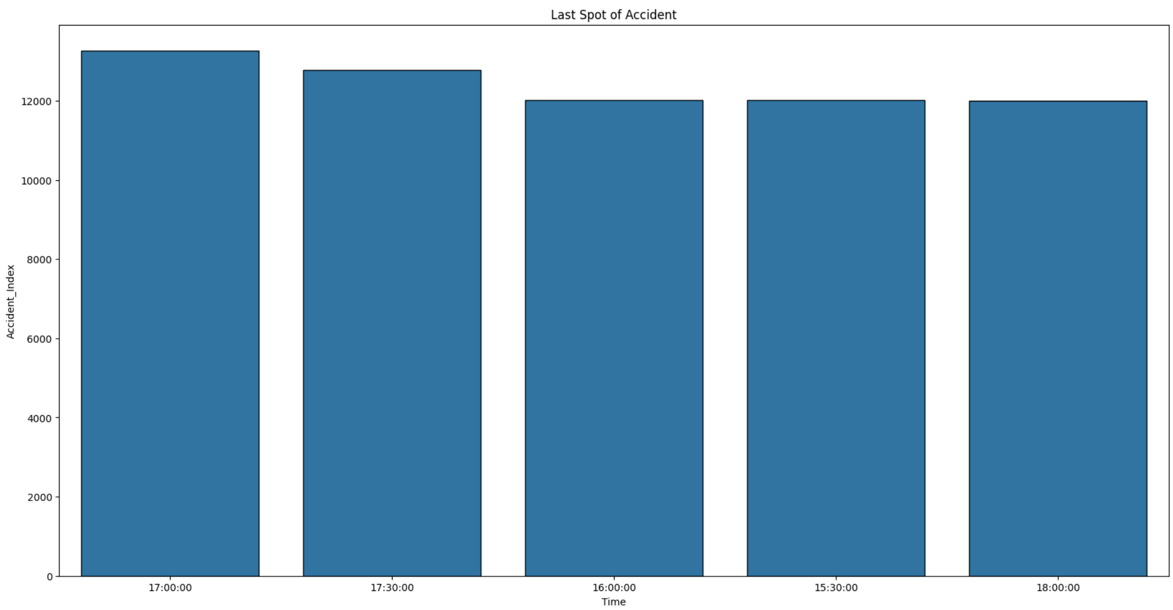
Accident By Time

In [217…

```
ti = df.groupby('Time')['Accident_Index'].count().sort_values(ascending = False)
```

In [218…

```
plt.figure(figsize=(20,10))
sns.barplot(data = ti, edgecolor = 'black')
plt.title('Last Spot of Accident')
plt.show()
```



Mostly in evening time accident happens.

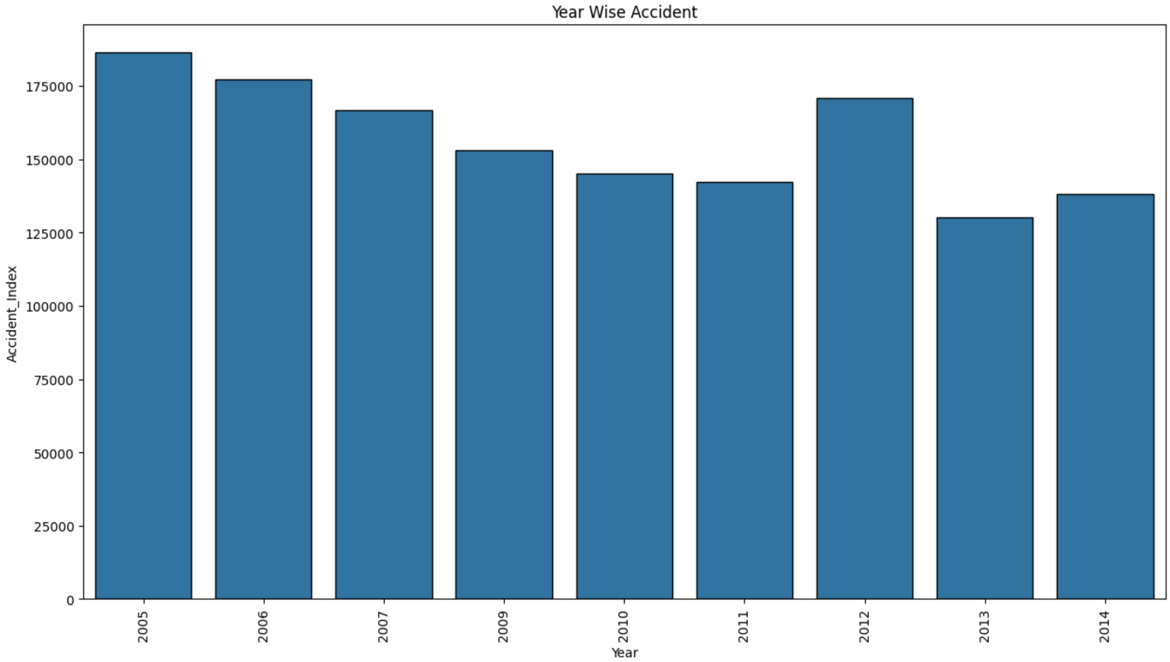
Accident By Year

In [198…

```
yr = df.groupby('Year')['Accident_Index'].count().sort_values(ascending = False)
```

In [199…

```
sns.barplot(data= yr, edgecolor='black')
plt.title('Year Wise Accident')
plt.xticks(rotation=90)
plt.show()
```



Year 2005,2006 & 2012 has maximum Accidents.

SUMMARY

- The Data was all about road accidents in the UK from various locations. The Data record was of 9 years from 2005 to 2014.

The target was to address from which locations the maximum accident occurs. The locations include Easting OSGR, Northing OSGR, and LSOA_of_Accident_Location.

- Total Accident was 1410077.
- For Location_Easting_OSGR 533500 is the highest point where accidents happen.
- 181000 to 181500 are red zone in Location_Northing_OSGR.
- Most of people are in fatal severity.
- The collation of two vehicles is higher in number.
- It found that occurrences of single-person involvement are higher in accidents.
- 300 & 204 are the highest point of accident in Local Authority District.
- E1000016 & E1000030 have higher numbers Local Authority Highway.
- Accident at Single carriageway have larger numbers.

- 30 & 60 are Speed where most of the accidents happen.
- Attendance of police on the accident spot is 80%.
- Accident on wet road is higher.
- Most accidents happen when there where no Crossing infrastructure within 50 meters.
- Most accidents happen when there where no Physical infrastructure within 50 meters.
- Daylight: Street light present has highest number.
- Fine without high winds has highest number.
- Accidents in Urban area is higher.
- Average Police E01000004 & E01011365 is 35 & 13
- E01000004 & E01011365 are highest LSOA.
- Mostly in evening time accident happens.
- Year 2005,2006 & 2012 has maximum Accidents.

The way to reduce accidents:

During the analysis, it clearly shows most of the accidents happened on Sinngle roadway, and no crossing infrastructure either Physical or Human there within 50 meters. Also, in comparison to other LSOAs where there is a high accident zone, there are fewer Police officers are there. So we have to increase Human supervisors and physical infrastructure, and also deploy more police at the Top 10 LSOA.