TASK- 5

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

In [1]:

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

In [11]:

```
df_USA=pd.read_csv('D:\\mlworld\\mlobs\\usa.csv')
df_USA.head()
```

Out[11]:

	ID	Source	Severity	Start_Time	End_Time	Start_Lat	Start_Lng	End_Lat	End_Lng	D
0	A-1	Source2	3	08-02-2016 05:46	08-02-2016 11:00	39.865147	-84.058723	NaN	NaN	
1	A-2	Source2	2	08-02-2016 06:07	08-02-2016 06:37	39.928059	-82.831184	NaN	NaN	
2	A-3	Source2	2	08-02-2016 06:49	08-02-2016 07:19	39.063148	-84.032608	NaN	NaN	
3	A-4	Source2	3	08-02-2016 07:23	08-02-2016 07:53	39.747753	-84.205582	NaN	NaN	
4	A-5	Source2	2	08-02-2016 07:39	08-02-2016 08:09	39.627781	-84.188354	NaN	NaN	

5 rows × 46 columns

In [8]:

```
df_USA.columns
```

Out[8]:

```
Index(['ID', 'Source', 'Severity', 'Start_Time', 'End_Time', 'Start_Lat',
       'Start_Lng', 'End_Lat', 'End_Lng', 'Distance(mi)', 'Description',
       'Street', 'City', 'County', 'State', 'Zipcode', 'Country', 'Timezon
e',
       'Airport_Code', 'Weather_Timestamp', 'Temperature(F)', 'Wind_Chill
(F)',
       'Humidity(%)', 'Pressure(in)', 'Visibility(mi)', 'Wind_Direction',
       'Wind_Speed(mph)', 'Precipitation(in)', 'Weather_Condition', 'Ameni
ty',
       'Bump', 'Crossing', 'Give_Way', 'Junction', 'No_Exit', 'Railway',
       'Roundabout', 'Station', 'Stop', 'Traffic_Calming', 'Traffic_Signa
l',
       'Turning_Loop', 'Sunrise_Sunset', 'Civil_Twilight', 'Nautical_Twili
ght',
       'Astronomical_Twilight'],
      dtype='object')
```

In [9]:

```
df_USA.dtypes.value_counts()
```

Out[9]:

```
object     20
bool       13
float64    12
int64       1
dtype: int64
```

In [12]:

```
df_USA.shape
```

Out[12]:

```
(1048575, 46)
```

In [13]:

```
df_USA.describe()
```

Out[13]:

	Severity	Start_Lat	Start_Lng	End_Lat	End_Lng	Distance(mi)	Temper
count	1.048575e+06	1.048575e+06	1.048575e+06	0.0	0.0	1.048575e+06	1.032(
mean	2.321943e+00	3.606598e+01	-9.463107e+01	NaN	NaN	1.748895e-01	6.483(
std	5.464841e-01	4.899255e+00	1.732592e+01	NaN	NaN	1.475837e+00	1.726(
min	1.000000e+00	2.455480e+01	-1.244974e+02	NaN	NaN	0.000000e+00	-7.780(
25%	2.000000e+00	3.293387e+01	-1.173193e+02	NaN	NaN	0.000000e+00	5.400(
50%	2.000000e+00	3.527291e+01	-8.793123e+01	NaN	NaN	0.000000e+00	6.700(
75%	3.000000e+00	4.011330e+01	-8.089787e+01	NaN	NaN	1.000000e-02	7.700(
max	4.000000e+00	4.899809e+01	-6.816079e+01	NaN	NaN	3.365700e+02	1.960(

In [14]:

```
df_USA.State.unique
```

Out[14]:

```
<bound method Series.unique of 0          OH
1          OH
2          OH
3          OH
4          OH
           ..
1048570    TX
1048571    TX
1048572    TX
1048573    TX
1048574    TX
Name: State, Length: 1048575, dtype: object>
```

In [15]:

```
df1=df_USA[df_USA['State']=='CA']
df1['IDD'] = df1['ID'].astype('str').str.extractall('(\d+)').unstack().fillna('').sum(ax
df1
```

```
C:\Users\91939\AppData\Local\Temp\ipykernel_13252\206313945.py:2: SettingW
ithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-doc
s/stable/user_guide/indexing.html#returning-a-view-versus-a-copy (https://
pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-
view-versus-a-copy)
  df1['IDD'] = df1['ID'].astype('str').str.extractall('(\d+)').unstack().f
illna('').sum(axis=1).astype(int)
```

Out[15]:

	ID	Source	Severity	Start_Time	End_Time	Start_Lat	Start_Lng	End_Lat
728	A-729	Source2	3	21-06-2016 10:34	21-06-2016 11:04	38.085300	-122.233017	NaN
729	A-730	Source2	3	21-06-2016 10:30	21-06-2016 11:16	37.631813	-122.084167	NaN
730	A-731	Source2	2	21-06-2016 10:49	21-06-2016 11:19	37.896564	-122.070717	NaN
731	A-732	Source2	3	21-06-2016 10:41	21-06-2016 11:11	37.334255	-122.032471	NaN
732	A-733	Source2	2	21-06-2016 10:16	21-06-2016 11:04	37.250729	-121.910713	NaN
...	...	...	...	...	...	...	...	...
1047395	A-1057166	Source2	3	06-05-2021 17:56	06-05-2021 18:41	33.909714	-117.281723	NaN
1047396	A-1057167	Source2	3	06-05-2021 18:47	06-05-2021 19:17	34.029724	-118.402946	NaN
1047397	A-1057168	Source2	2	06-05-2021 19:15	06-05-2021 19:44	34.075108	-118.231964	NaN
1047398	A-1057169	Source2	3	06-05-2021 19:40	06-05-2021 20:40	33.913177	-118.125137	NaN
1047399	A-1057170	Source2	3	06-05-2021 20:14	06-05-2021 21:28	34.026962	-118.250504	NaN

264077 rows × 47 columns

In [16]:

```
df1.duplicated().sum()
```

Out[16]:

```
0
```

In [17]:

```
d1f=df1.dropna(subset=['Precipitation(in)'])
```

In [18]:

```
df1=df1.dropna(subset=['Temperature(F)','Wind_Chill(F)','Humidity(%)','Pressure(in)','Vi
                       'Weather_Condition'])
```

In [19]:

```
df1.shape
```

Out[19]:

```
(92087, 47)
```

In [20]:

```
df1.isna().sum()/len(df1)*100
```

Out[20]:

```
ID                        0.000000
Source                    0.000000
Severity                  0.000000
Start_Time                0.000000
End_Time                  0.000000
Start_Lat                 0.000000
Start_Lng                 0.000000
End_Lat                 100.000000
End_Lng                 100.000000
Distance(mi)              0.000000
Description               0.001086
Street                    0.321435
City                      0.000000
County                    0.000000
State                     0.000000
Zipcode                   0.000000
Country                   0.000000
Timezone                  0.000000
Airport_Code              0.000000
Weather_Timestamp         0.000000
Temperature(F)            0.000000
Wind_Chill(F)             0.000000
Humidity(%)               0.000000
Pressure(in)              0.000000
Visibility(mi)            0.000000
Wind_Direction            0.000000
Wind_Speed(mph)           0.000000
Precipitation(in)        10.120864
Weather_Condition         0.000000
Amenity                   0.000000
Bump                      0.000000
Crossing                  0.000000
Give_Way                  0.000000
Junction                  0.000000
No_Exit                   0.000000
Railway                   0.000000
Roundabout                0.000000
Station                   0.000000
Stop                      0.000000
Traffic_Calming           0.000000
Traffic_Signal            0.000000
Turning_Loop              0.000000
Sunrise_Sunset            0.062984
Civil_Twilight            0.062984
Nautical_Twilight         0.062984
Astronomical_Twilight     0.062984
IDD                       0.000000
dtype: float64
```

In [21]:

```
df1=df1.dropna(subset=['City','Sunrise_Sunset',
       'Civil_Twilight', 'Nautical_Twilight', 'Astronomical_Twilight'])
df1.isna().sum()/len(df1)*100
```

Out[21]:

```
ID                        0.000000
Source                    0.000000
Severity                  0.000000
Start_Time                0.000000
End_Time                  0.000000
Start_Lat                 0.000000
Start_Lng                 0.000000
End_Lat                 100.000000
End_Lng                 100.000000
Distance(mi)              0.000000
Description               0.001087
Street                    0.321638
City                      0.000000
County                    0.000000
State                     0.000000
Zipcode                   0.000000
Country                   0.000000
Timezone                  0.000000
Airport_Code              0.000000
Weather_Timestamp         0.000000
Temperature(F)            0.000000
Wind_Chill(F)             0.000000
Humidity(%)               0.000000
Pressure(in)              0.000000
Visibility(mi)            0.000000
Wind_Direction            0.000000
Wind_Speed(mph)           0.000000
Precipitation(in)        10.127242
Weather_Condition         0.000000
Amenity                   0.000000
Bump                      0.000000
Crossing                  0.000000
Give_Way                  0.000000
Junction                  0.000000
No_Exit                   0.000000
Railway                   0.000000
Roundabout                0.000000
Station                   0.000000
Stop                      0.000000
Traffic_Calming           0.000000
Traffic_Signal            0.000000
Turning_Loop              0.000000
Sunrise_Sunset            0.000000
Civil_Twilight            0.000000
Nautical_Twilight         0.000000
Astronomical_Twilight     0.000000
IDD                       0.000000
dtype: float64
```

In [22]:

```
df1['Weather_Condition'].value_counts()
```

Out[22]:

```
Fair                            57558
Cloudy                          10793
Mostly Cloudy                    6496
Partly Cloudy                    4798
Haze                             2537
Fog                              2246
Light Rain                       2157
Clear                            1749
Fair / Windy                      826
Smoke                             698
Rain                              622
Overcast                          423
Heavy Rain                        240
Scattered Clouds                  131
Mostly Cloudy / Windy              80
Partly Cloudy / Windy              76
Light Rain / Windy                 66
Cloudy / Windy                     61
Rain / Windy                       57
Shallow Fog                        46
Light Snow                         43
Patches of Fog                     37
Light Drizzle                      36
Heavy Rain / Windy                 34
Mist                               34
Thunder in the Vicinity            25
Drizzle                            24
T-Storm                            21
Showers in the Vicinity            21
Snow                               16
Haze / Windy                       10
Thunder                             9
Blowing Dust / Windy                6
Fog / Windy                         6
Blowing Dust                        6
Smoke / Windy                       6
Light Rain with Thunder             5
Heavy Snow                          5
N/A Precipitation                   4
Snow / Windy                        3
Light Rain Shower                   3
Light Freezing Fog                  3
Widespread Dust / Windy             3
Widespread Dust                     2
Light Rain Showers                  2
Light Snow / Windy                  1
Light Freezing Rain                 1
Rain Showers                        1
Light Thunderstorms and Rain        1
Heavy T-Storm                       1
Name: Weather_Condition, dtype: int64
```

In [24]:

```
df_cat=df1.select_dtypes('object')
df_num=df1.select_dtypes(np.number)
df_cat=df_cat.drop('ID',axis=1)
```

In [25]:

```
df_cat=df1.select_dtypes('object')
col_name=[]
length=[]

for i in df_cat.columns:
    col_name.append(i)
    length.append(len(df_cat[i].unique()))
df_2=pd.DataFrame(zip(col_name,length),columns=['feature','count_of_unique_values'])
df_2
```

Out[25]:

	feature	count_of_unique_values
0	ID	92029
1	Source	2
2	Start_Time	82191
3	End_Time	76183
4	Description	82672
5	Street	9200
6	City	938
7	County	58
8	State	1
9	Zipcode	10733
10	Country	1
11	Timezone	2
12	Airport_Code	132
13	Weather_Timestamp	46093
14	Wind_Direction	23
15	Weather_Condition	50
16	Sunrise_Sunset	2
17	Civil_Twilight	2
18	Nautical_Twilight	2
19	Astronomical_Twilight	2

In [26]:

```
df1.drop(['Description','Zipcode','Weather_Timestamp'],axis=1,inplace=True)
del df1['Airport_Code']
df_num.columns
```

Out[26]:

```
Index(['Severity', 'Start_Lat', 'Start_Lng', 'End_Lat', 'End_Lng',
       'Distance(mi)', 'Temperature(F)', 'Wind_Chill(F)', 'Humidity(%)',
       'Pressure(in)', 'Visibility(mi)', 'Wind_Speed(mph)',
       'Precipitation(in)', 'IDD'],
      dtype='object')
```

In [27]:

```
len(df_num.columns)
```

Out[27]:

```
14
```

In [28]:

```
df_cat.columns
```

Out[28]:

```
Index(['ID', 'Source', 'Start_Time', 'End_Time', 'Description', 'Street',
       'City', 'County', 'State', 'Zipcode', 'Country', 'Timezone',
       'Airport_Code', 'Weather_Timestamp', 'Wind_Direction',
       'Weather_Condition', 'Sunrise_Sunset', 'Civil_Twilight',
       'Nautical_Twilight', 'Astronomical_Twilight'],
      dtype='object')
```

In [31]:

```
len(df_cat['City'].unique())
```

Out[31]:

```
938
```

In [32]:

```
df_num=df1.select_dtypes(np.number)
col_name=[]
length=[]

for i in df_num.columns:
    col_name.append(i)
    length.append(len(df_num[i].unique()))
df_2=pd.DataFrame(zip(col_name,length),columns=['feature','count_of_unique_values'])
df_2
```

Out[32]:

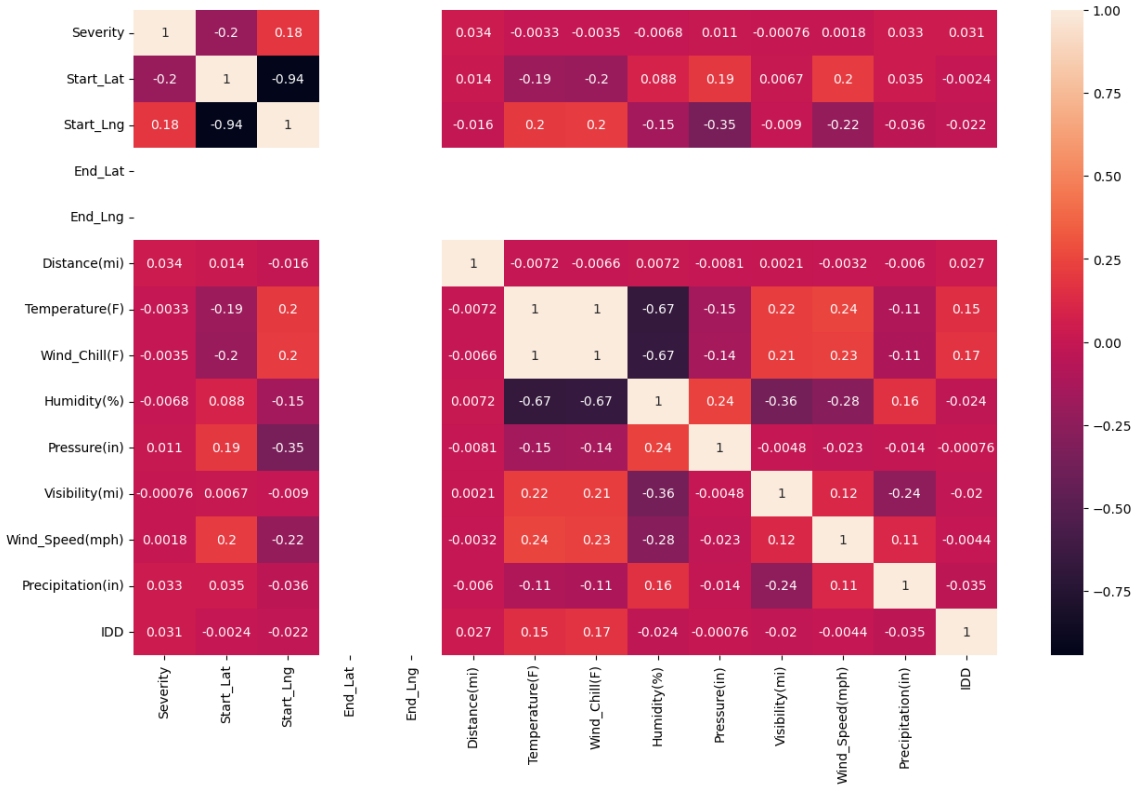
	feature	count_of_unique_values
0	Severity	4
1	Start_Lat	46496
2	Start_Lng	46057
3	End_Lat	1
4	End_Lng	1
5	Distance(mi)	533
6	Temperature(F)	163
7	Wind_Chill(F)	285
8	Humidity(%)	98
9	Pressure(in)	628
10	Visibility(mi)	37
11	Wind_Speed(mph)	66
12	Precipitation(in)	53
13	IDD	92029

In [33]:

```
plt.figure(figsize=(15 ,9))
sns.heatmap(df_num.corr() , annot=True)
```

Out[33]:

<AxesSubplot:>



In [34]:

```
cities = df1['City'].unique()
len(cities)
```

Out[34]:

938

In [35]:

```
accidents_by_cities = df1['City'].value_counts()
accidents_by_cities
```

Out[35]:

```
Los Angeles         8329
Sacramento          3160
San Jose            2560
San Diego           2168
Oakland             1574
                    ...
Potter Valley          1
Birds Landing          1
Trona                  1
Raymond                1
Fall River Mills       1
Name: City, Length: 938, dtype: int64
```

In [36]:

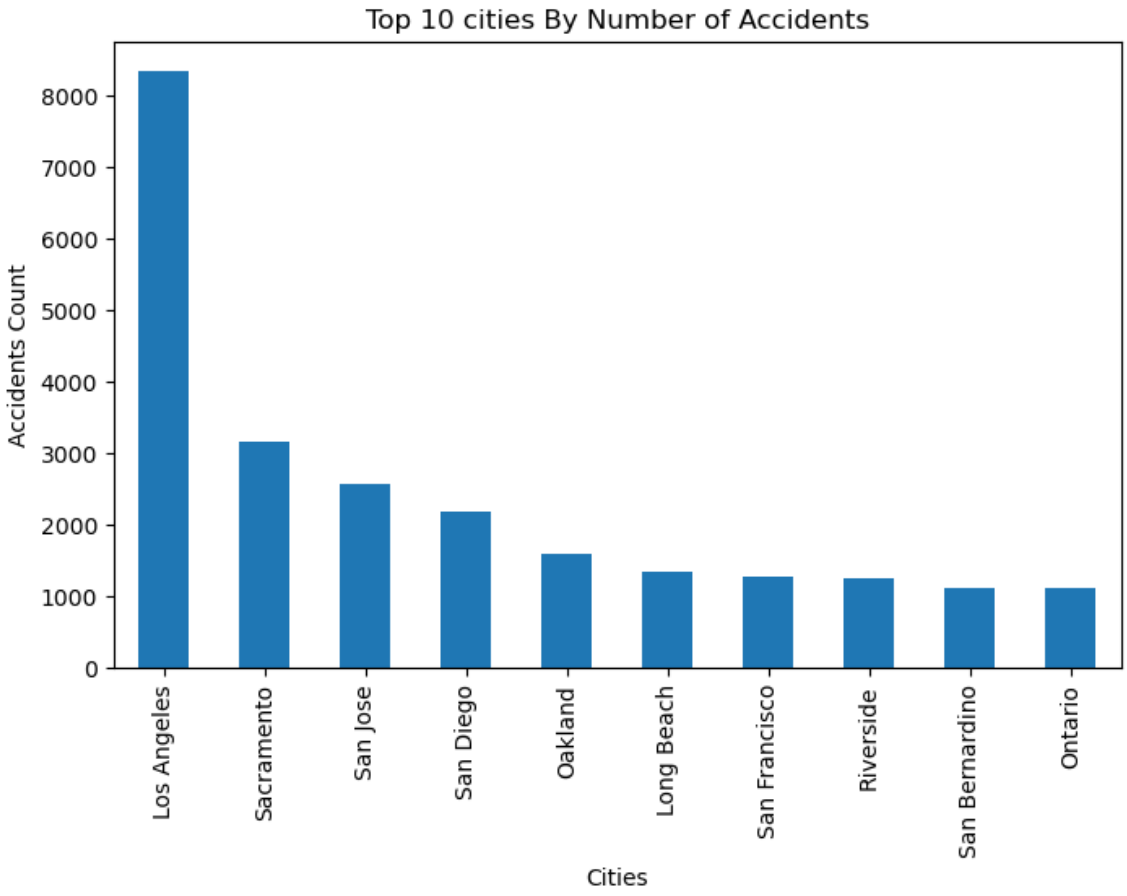
```
#top 10 cities by number of accident
accidents_by_cities[:10]
```

Out[36]:

```
Los Angeles       8329
Sacramento        3160
San Jose          2560
San Diego         2168
Oakland           1574
Long Beach        1332
San Francisco     1273
Riverside         1253
San Bernardino    1109
Ontario           1108
Name: City, dtype: int64
```

In [37]:

```
fig, ax = plt.subplots(figsize=(8,5))
accidents_by_cities[:10].plot(kind='bar')
ax.set(title = 'Top 10 cities By Number of Accidents',
       xlabel = 'Cities',
       ylabel = 'Accidents Count')
plt.show()
```



In [38]:

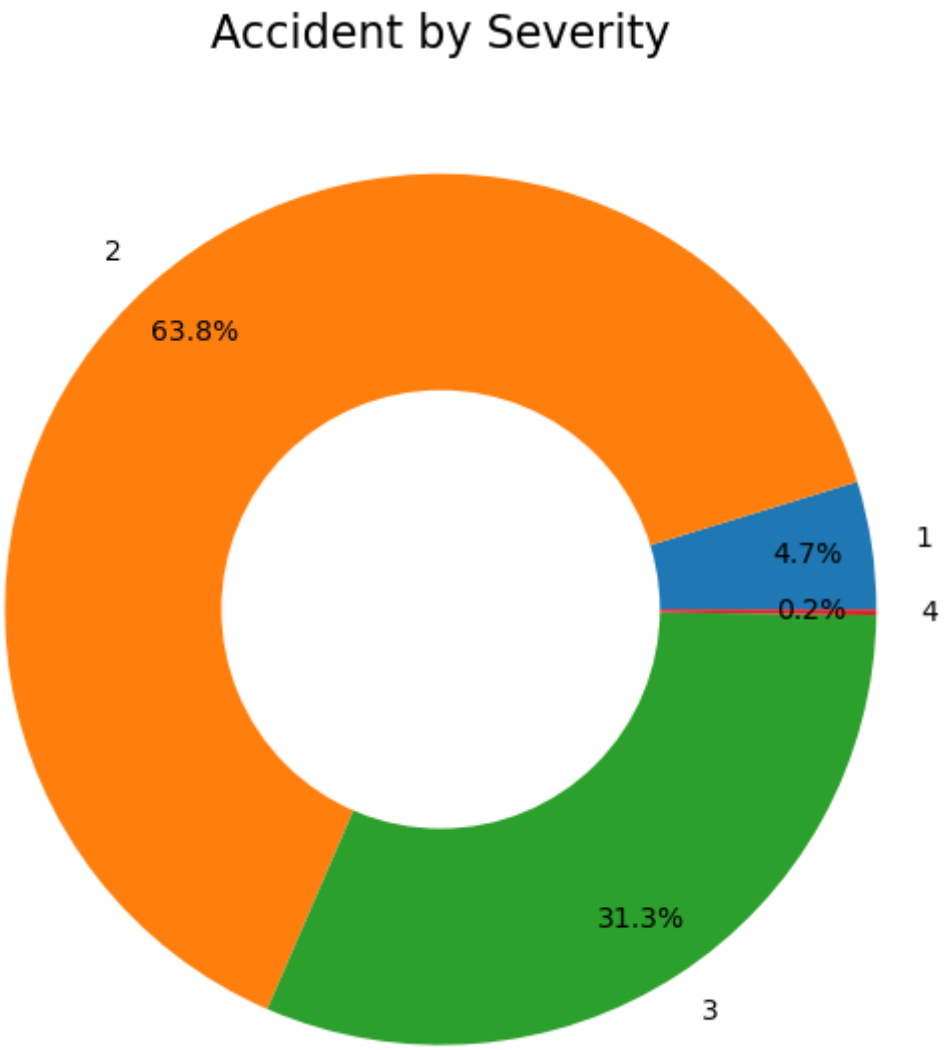
```
accidents_severity = df1.groupby('Severity').count()['ID']
accidents_severity
```

Out[38]:

```
Severity
1     4354
2    58678
3    28800
4      197
Name: ID, dtype: int64
```

In [39]:

```
fig, ax = plt.subplots(figsize=(8, 6), subplot_kw=dict(aspect="equal"))
label = [1,2,3,4]
plt.pie(accidents_severity, labels=label,
        autopct='%1.1f%%', pctdistance=0.85)
circle = plt.Circle( (0,0), 0.5, color='white')
p=plt.gcf()
p.gca().add_artist(circle)
ax.set_title("Accident by Severity",fontdict={'fontsize': 16})
plt.tight_layout()
plt.show()
```



In [41]:

```
df1['Start_Time'].dtypes
```

Out[41]:

```
dtype('O')
```

In [42]:

```
df1['End_Time'].dtypes
```

Out[42]:

```
dtype('O')
```

In [44]:

```
df1 = df1.astype({'Start_Time': 'datetime64[ns]', 'End_Time': 'datetime64[ns]'})
df1['Start_Time'].dtypes
```

Out[44]:

```
dtype('<M8[ns]')
```

In [52]:

```
df1['Start_Time']
```

Out[52]:

```
5041      2016-11-30 16:07:00
5063      2016-11-30 18:32:00
5073      2016-11-30 19:20:00
5075      2016-11-30 19:33:00
5080      2016-11-30 19:40:00
                  ...
1047395   2021-06-05 17:56:00
1047396   2021-06-05 18:47:00
1047397   2021-06-05 19:15:00
1047398   2021-06-05 19:40:00
1047399   2021-06-05 20:14:00
Name: Start_Time, Length: 92029, dtype: datetime64[ns]
```

In [54]:

```
df1['End_Time']
```

Out[54]:

```
5041      2016-11-30 17:22:00
5063      2016-11-30 19:17:00
5073      2016-11-30 20:05:00
5075      2016-11-30 20:18:00
5080      2016-11-30 20:25:00
                  ...
1047395   2021-06-05 18:41:00
1047396   2021-06-05 19:17:00
1047397   2021-06-05 19:44:00
1047398   2021-06-05 20:40:00
1047399   2021-06-05 21:28:00
Name: End_Time, Length: 92029, dtype: datetime64[ns]
```

In [55]:

```
df1['start_date'] = [d.date() for d in df1['Start_Time']]
df1['start_time'] = [d.time() for d in df1['Start_Time']]
```

In [56]:

```
df1['end_date'] = [d.date() for d in df1['End_Time']]
df1['end_time'] = [d.time() for d in df1['End_Time']]
df1['end_time']
```

Out[56]:

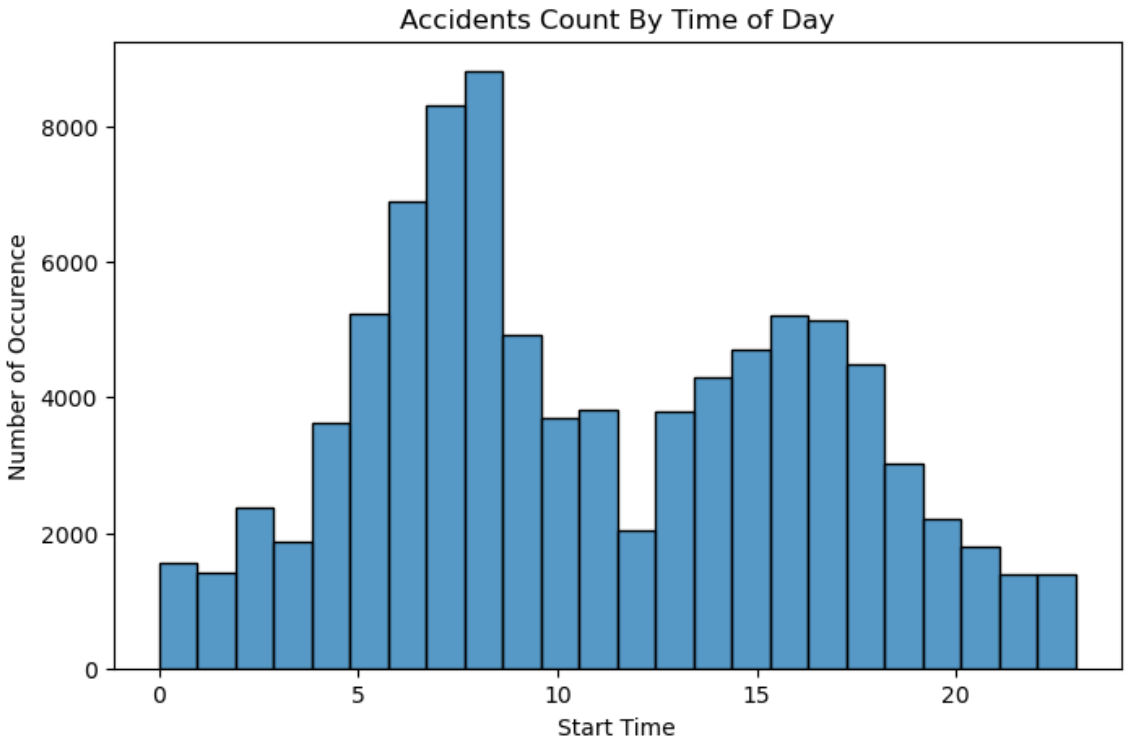
```
5041       17:22:00
5063       19:17:00
5073       20:05:00
5075       20:18:00
5080       20:25:00
             ...
1047395    18:41:00
1047396    19:17:00
1047397    19:44:00
1047398    20:40:00
1047399    21:28:00
Name: end_time, Length: 92029, dtype: object
```

In [57]:

```
fig, ax = plt.subplots(figsize=(8,5))
sns.histplot(df1['Start_Time'].dt.hour, bins = 24)

plt.xlabel("Start Time")
plt.ylabel("Number of Occurence")
plt.title('Accidents Count By Time of Day')

plt.show()
```

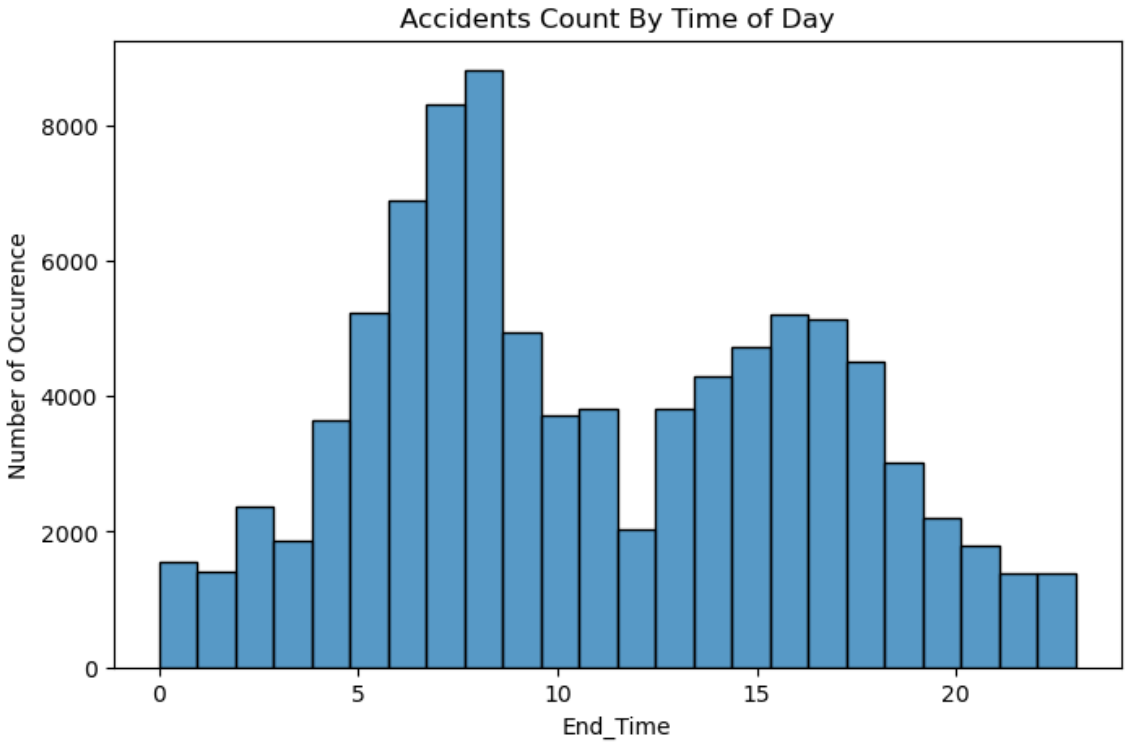


In [58]:

```
fig, ax = plt.subplots(figsize=(8,5))
sns.histplot(df1['Start_Time'].dt.hour, bins = 24)

plt.xlabel("End_Time")
plt.ylabel("Number of Occurence")
plt.title('Accidents Count By Time of Day')

plt.show()
```



In [67]:

```
del df1['Start_Time']
del df1['End_Time']
```

In [68]:

```
%matplotlib inline
import os
```

In [77]:

```
df1.groupby('Severity').count()['IDD']
```

Out[77]:

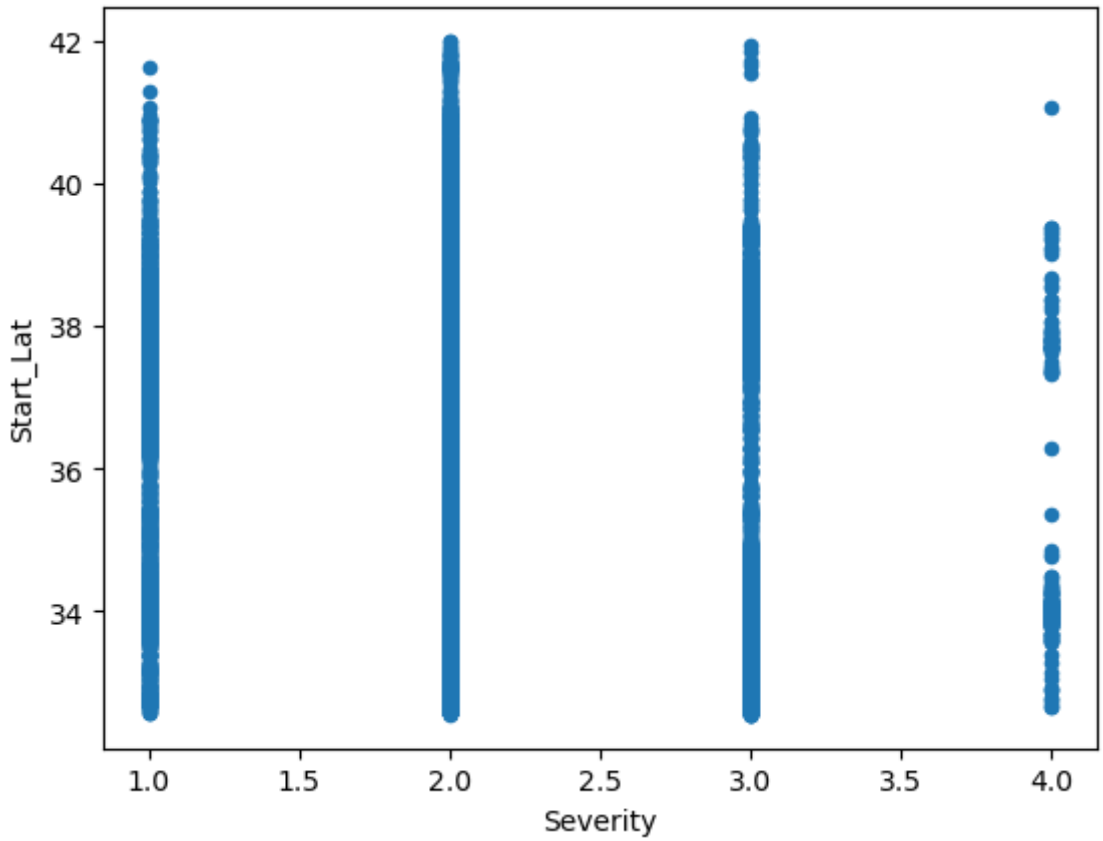
```
Severity
1     4354
2    58678
3    28800
4      197
Name: IDD, dtype: int64
```

In [78]:

```
df_num.plot(kind='scatter', y='Start_Lat', x='Severity')
```

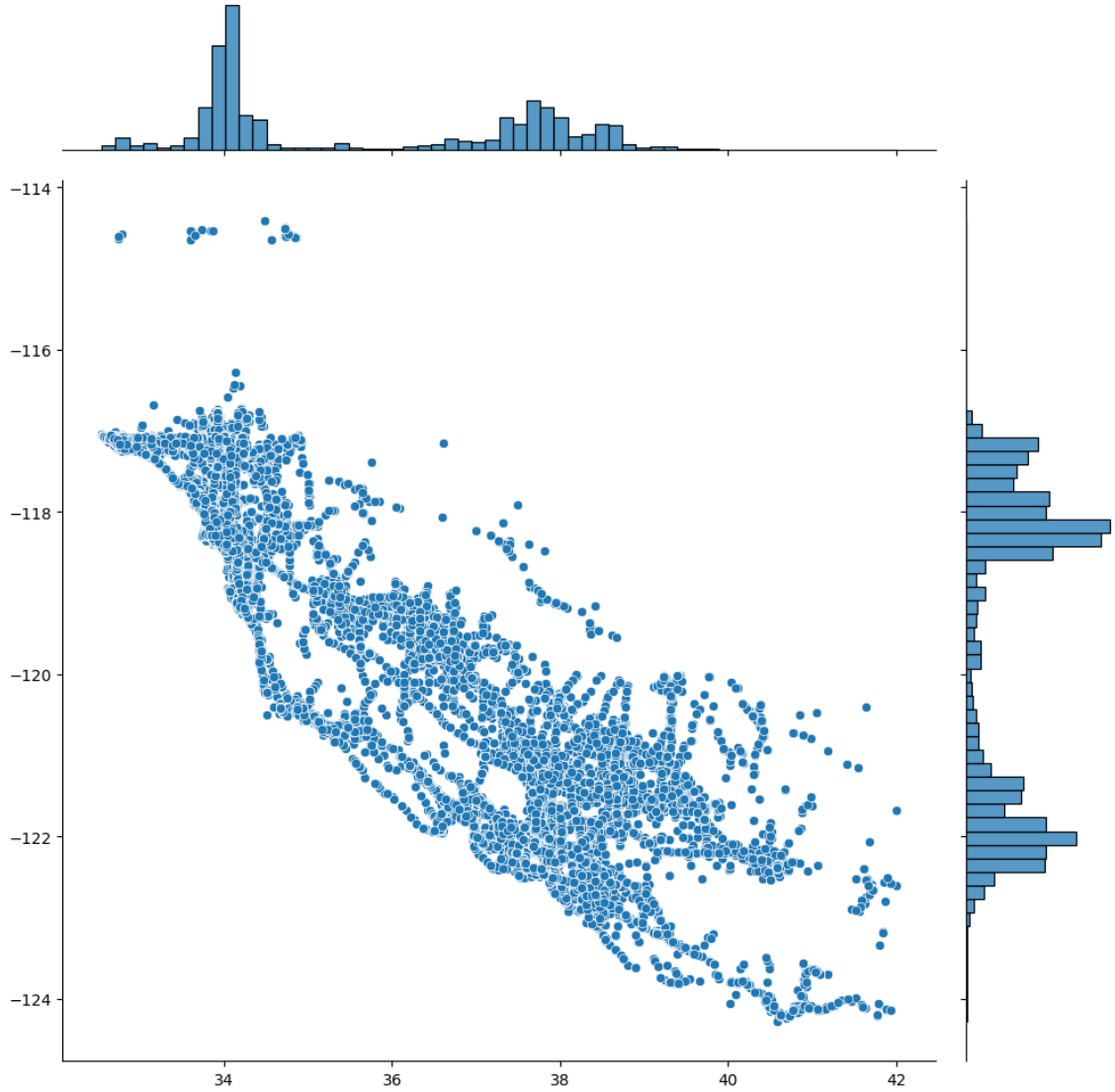
Out[78]:

```
<AxesSubplot:xlabel='Severity', ylabel='Start_Lat'>
```



In [79]:

```
sns.jointplot(x=df_num.Start_Lat.values , y=df_num.Start_Lng.values,height=10)
plt.ylabel('Start lattitude', fontsize=12)
plt.xlabel('Start lattitude', fontsize=12)
plt.show()
```



In [ ]:

```

```

In [ ]:

```

```