1.Import neccessary libaries nedded

In [ ]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

In [ ]:
```
print(sns.get_dataset_names())
```

```
['anagrams', 'anscombe', 'attention', 'brain_networks', 'car_crashes', 'diamonds',
'dots', 'dowjones', 'exercise', 'flights', 'fmri', 'geyser', 'glue', 'healthexp',
'iris', 'mpg', 'penguins', 'planets', 'seaice', 'taxis', 'tips', 'titanic']
```

2.Loading Dataset

In [ ]:
```
df=sns.load_dataset('car_crashes')
```

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

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 51 entries, 0 to 50
Data columns (total 8 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   total           51 non-null     float64
 1   speeding        51 non-null     float64
 2   alcohol         51 non-null     float64
 3   not_distracted  51 non-null     float64
 4   no_previous     51 non-null     float64
 5   ins_premium     51 non-null     float64
 6   ins_losses      51 non-null     float64
 7   abbrev          51 non-null     object
dtypes: float64(7), object(1)
memory usage: 3.3+ KB
```

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

Out[ ]:
```
(51, 8)
```

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

Out[ ]:

	total	speeding	alcohol	not_distracted	no_previous	ins_premium	ins_losses	abbrev
0	18.8	7.332	5.640	18.048	15.040	784.55	145.08	AL
1	18.1	7.421	4.525	16.290	17.014	1053.48	133.93	AK
2	18.6	6.510	5.208	15.624	17.856	899.47	110.35	AZ
3	22.4	4.032	5.824	21.056	21.280	827.34	142.39	AR
4	12.0	4.200	3.360	10.920	10.680	878.41	165.63	CA

plotting univariate distribution

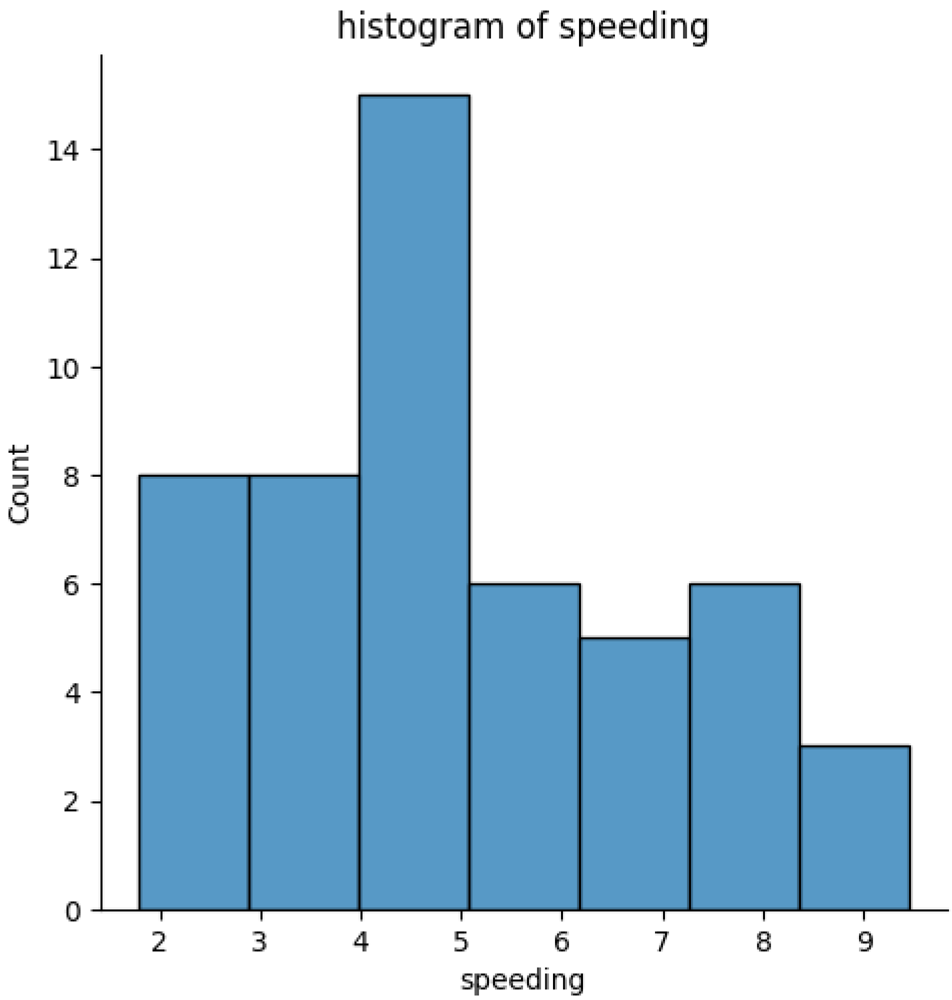
In [ ]:
```
df.describe()
```

Out[ ]:

	total	speeding	alcohol	not_distracted	no_previous	ins_premium	ins_losses
count	51.000000	51.000000	51.000000	51.000000	51.000000	51.000000	51.000000
mean	15.790196	4.998196	4.886784	13.573176	14.004882	886.957647	134.493137
std	4.122002	2.017747	1.729133	4.508977	3.764672	178.296285	24.835922
min	5.900000	1.792000	1.593000	1.760000	5.900000	641.960000	82.750000
25%	12.750000	3.766500	3.894000	10.478000	11.348000	768.430000	114.645000
50%	15.600000	4.608000	4.554000	13.857000	13.775000	858.970000	136.050000
75%	18.500000	6.439000	5.604000	16.140000	16.755000	1007.945000	151.870000
max	23.900000	9.450000	10.038000	23.661000	21.280000	1301.520000	194.780000

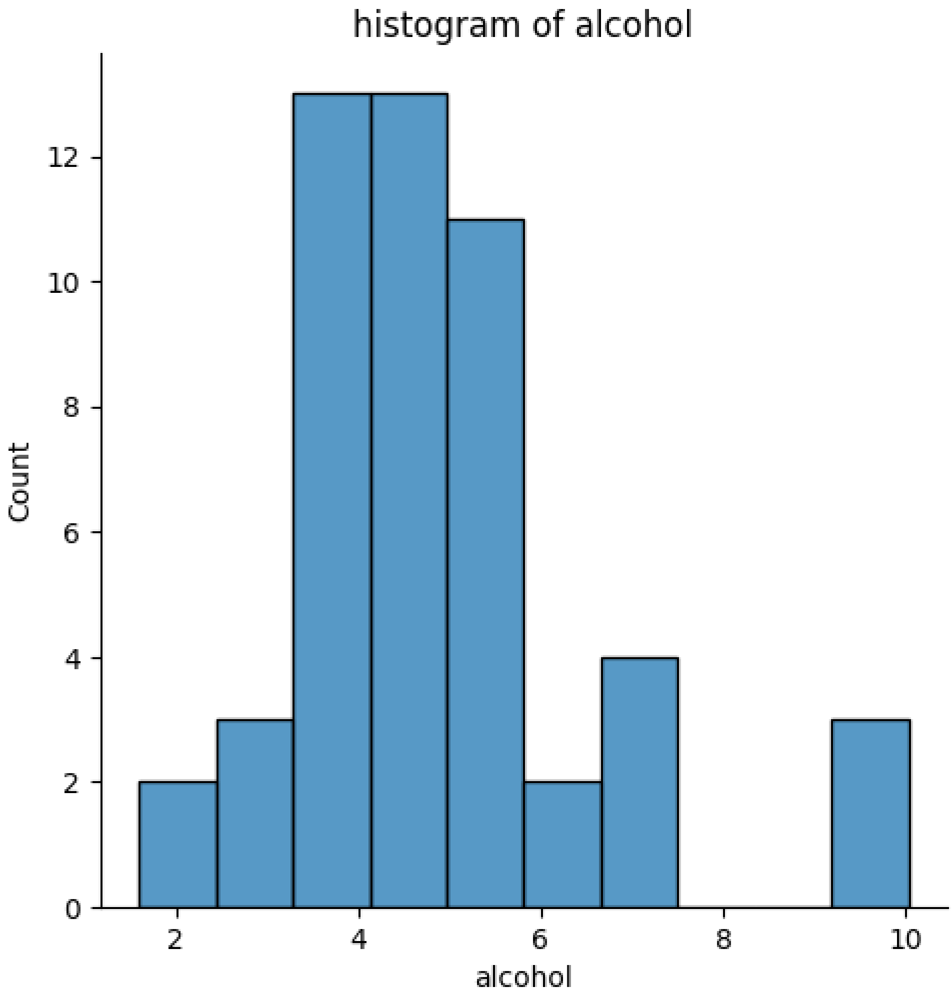
Inference: it shows how many accidents took per quantile and also shows standard deviation,mean,max

In [ ]:
```
#histogram
sns.displot(df['speeding'])
plt.title("histogram of speeding")
plt.show()
```



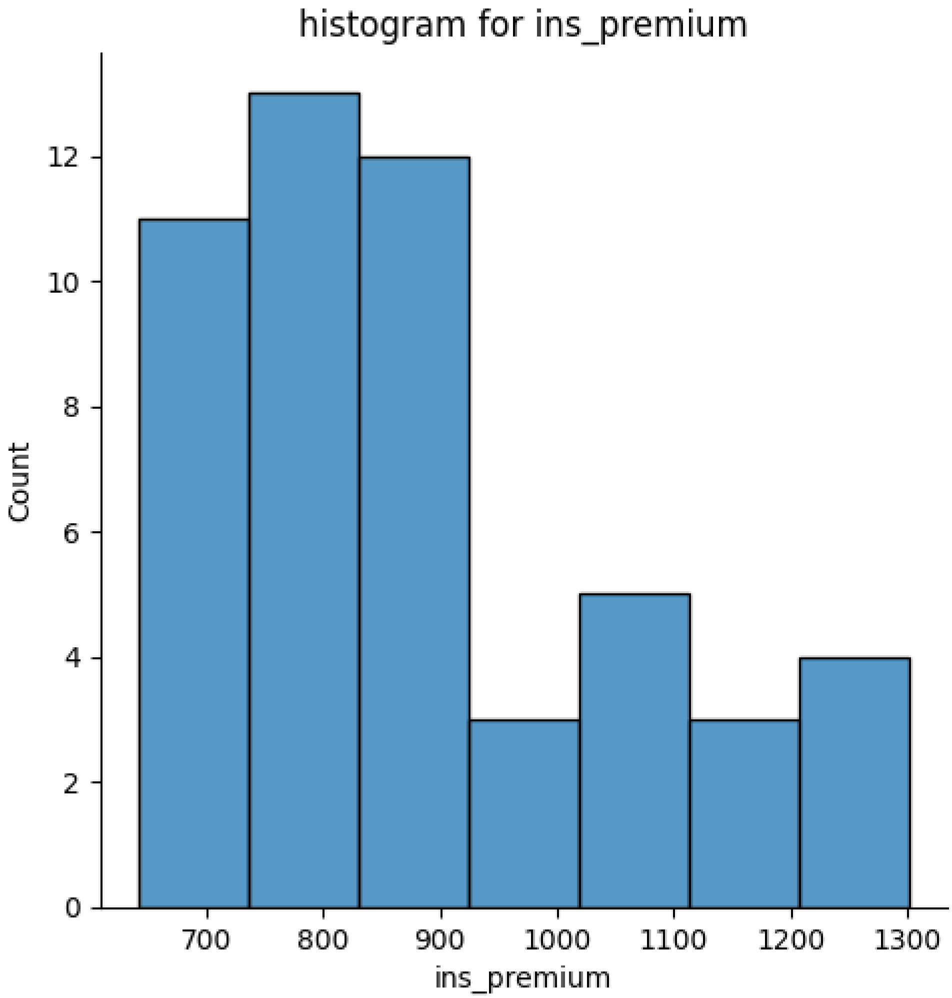
Inference: speeding ranges 4 to 5 has more count

In [ ]:
```
sns.displot(df['alcohol'])
plt.title("histogram of alcohol")
plt.show()
```



Inference:alcohol range from 4 to 5 hs highest count

In [ ]:
```
sns.displot(df['ins_premium'])
plt.title("histogram for ins_premium")
plt.show()
```

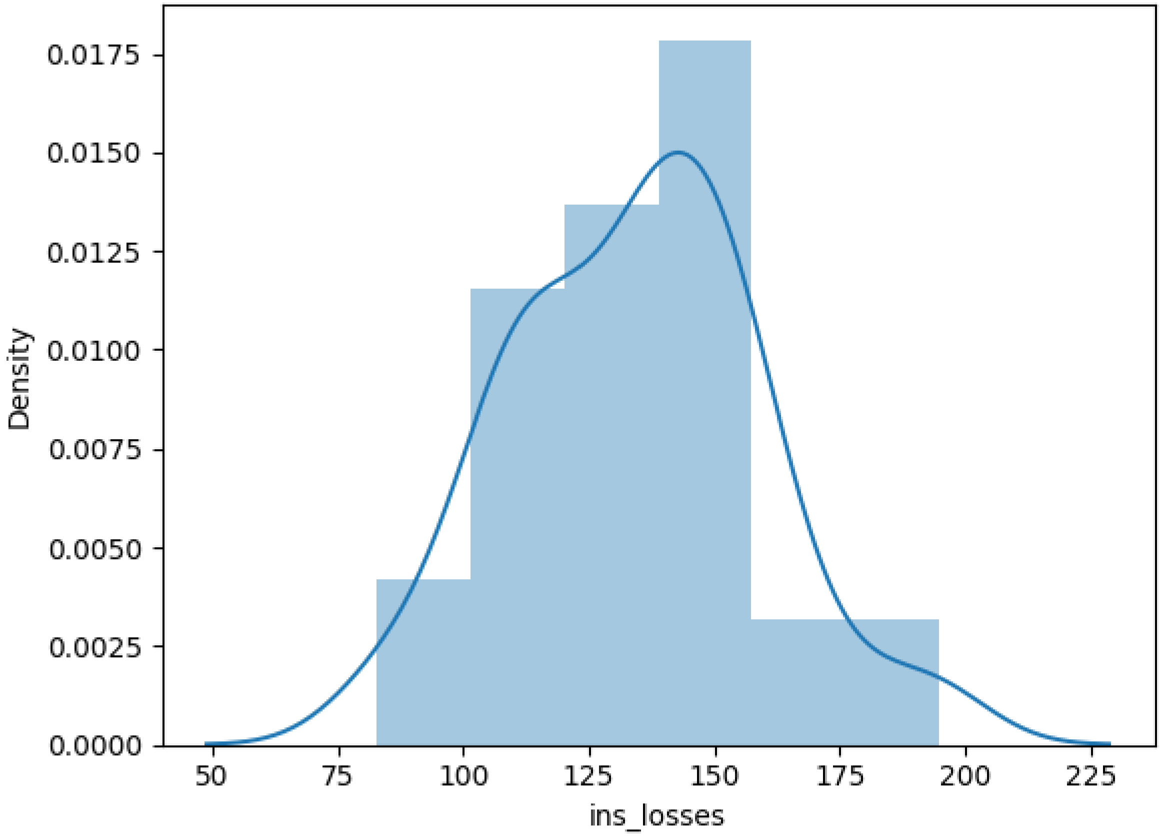


Inference:car ins_premium has highest count at 800

In [ ]:
```
sns.distplot(df["ins_losses"])
```

```
<ipython-input-5-46fdb0fb15ea>:1: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(df["ins_losses"])
```

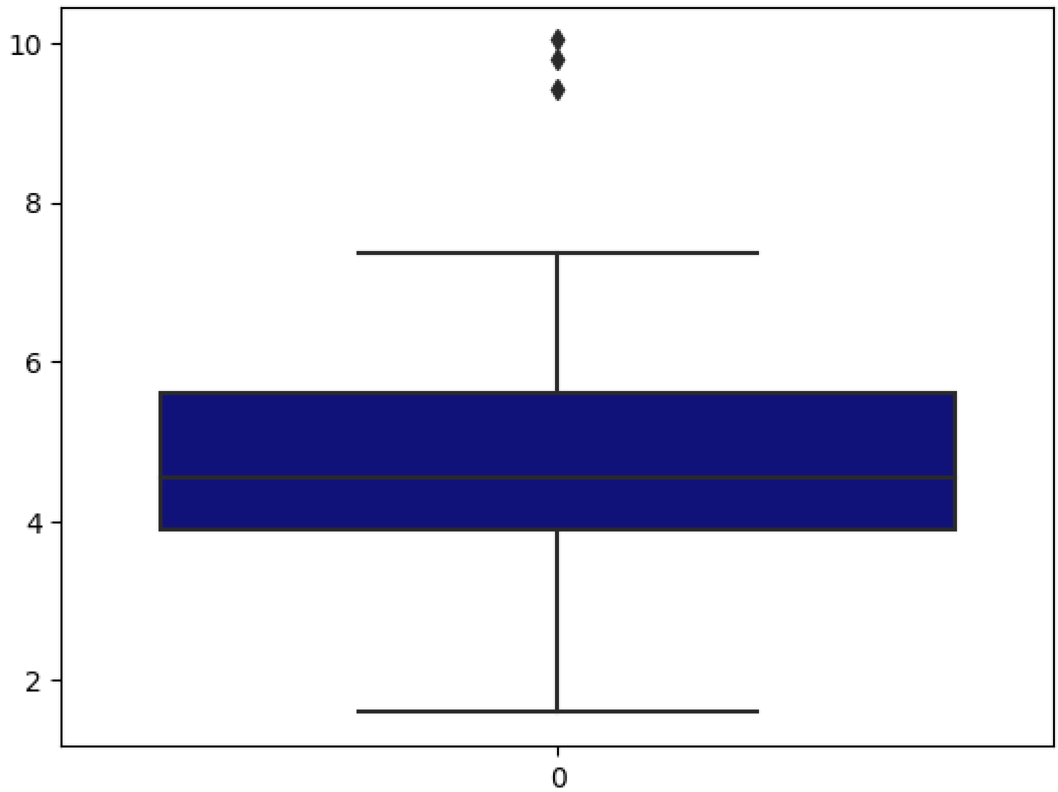
Out[ ]:
```
<Axes: xlabel='ins_losses', ylabel='Density'>
```



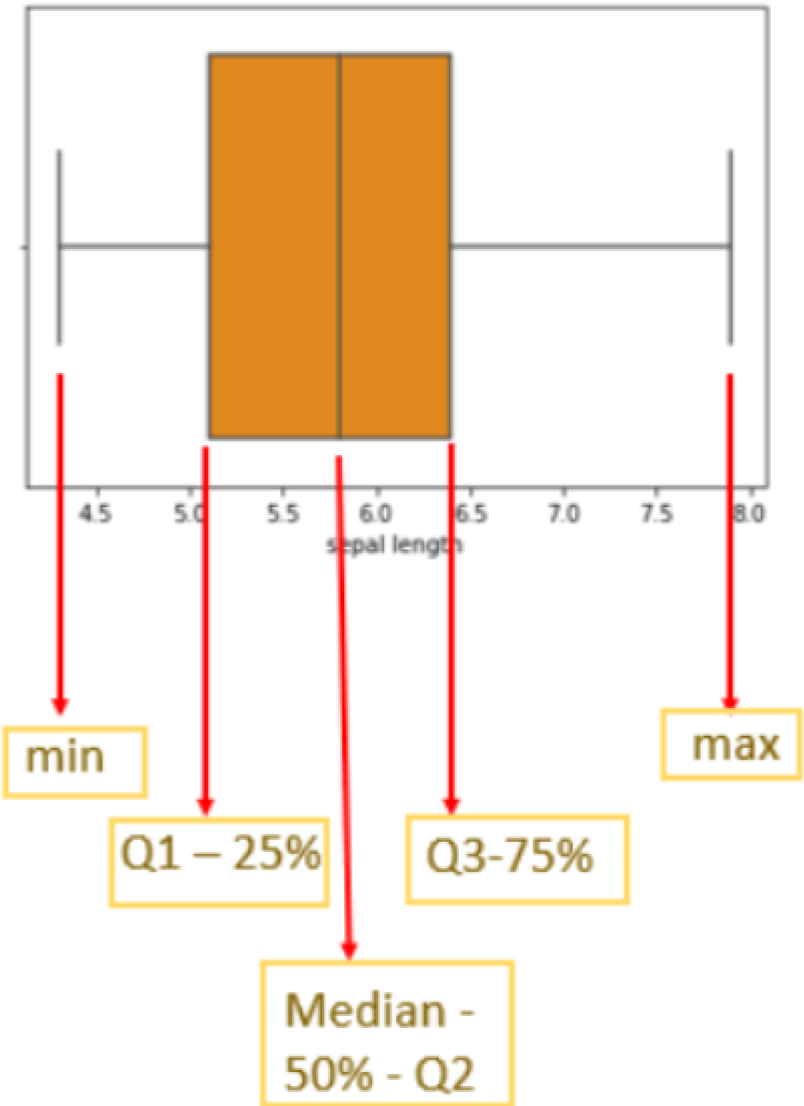
Inference: Losses incurred by insurance companies for collisions per insured driver occurs mostly at 150

In [ ]:
```
sns.boxplot(df['alcohol'],color='darkblue')
```

Out[ ]: <Axes: >



Inference: it shows the descriptive statistics in order to detect the outliers which is used further for data preprocessing .This represent the 5 point statistics in a graphical manner



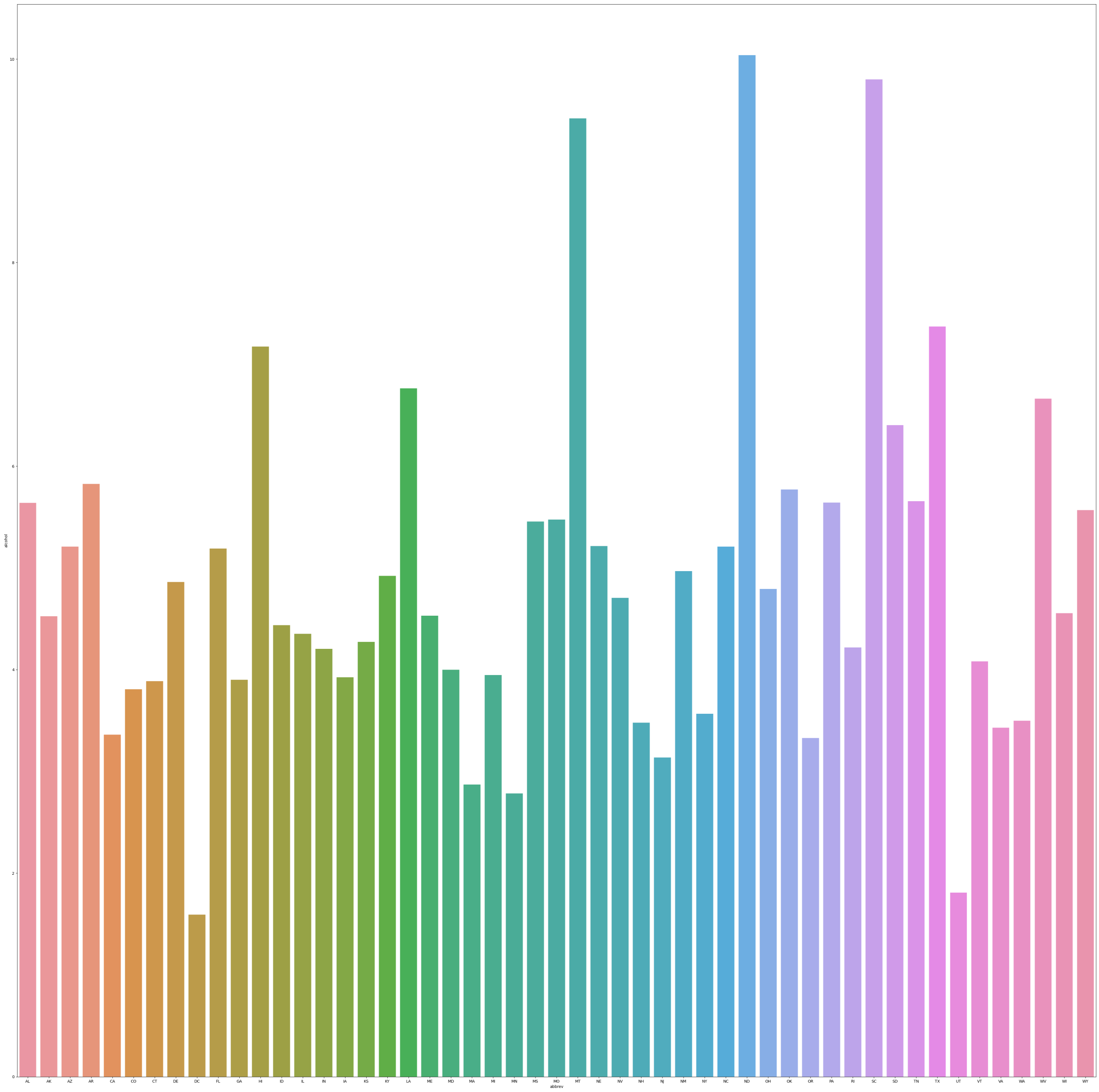

```
df['alcohol'].describe()
```

```
count     51.000000
mean       4.886784
std        1.729133
min        1.593000
25%        3.894000
50%        4.554000
75%        5.604000
max       10.038000
Name: alcohol, dtype: float64
```

Bivariate distribution

Bar Plot

In [ ]:
```
fig, ax1 = plt.subplots(figsize=(50, 50))
sns.barplot(x="abbrev",y="alcohol",data=df, ax=ax1)
plt.show()
```



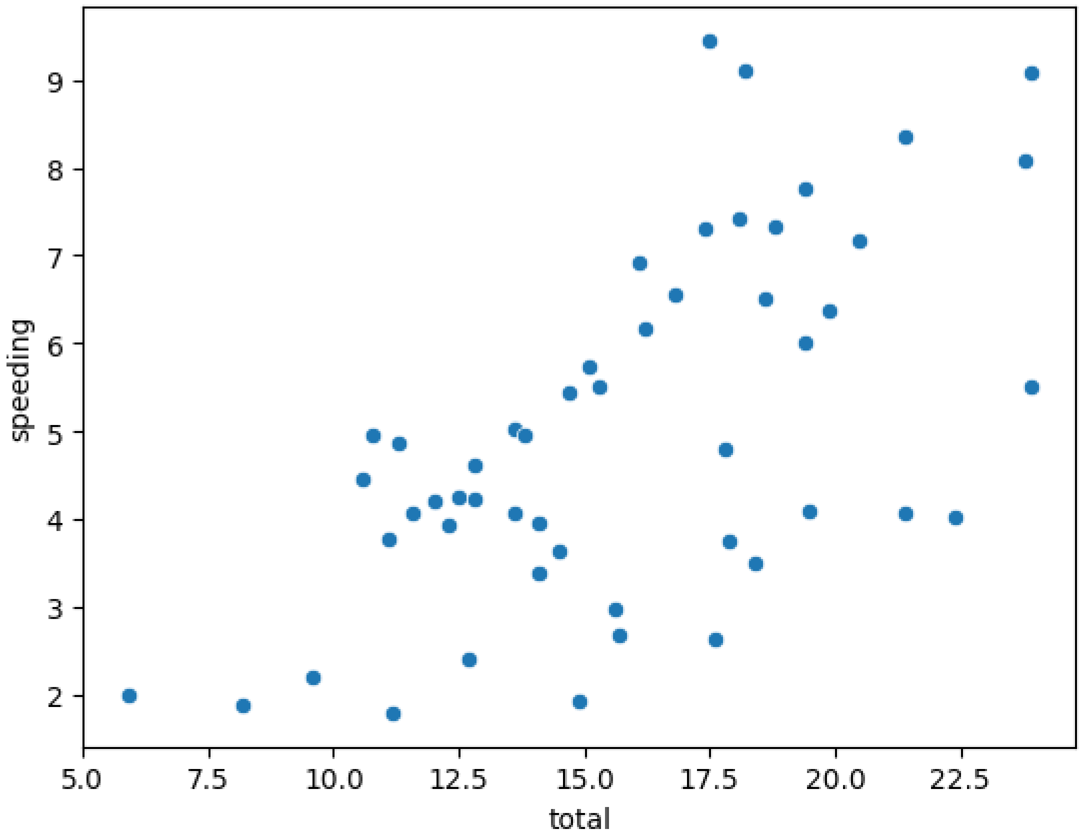
Inference:number of crashes that happened due to Alcohol intake is the highest in the country ND (North Dakota)

Scatter plot

numerical vs numerical

In [ ]:
```
sns.scatterplot(x="total",y="speeding",data=df)
```

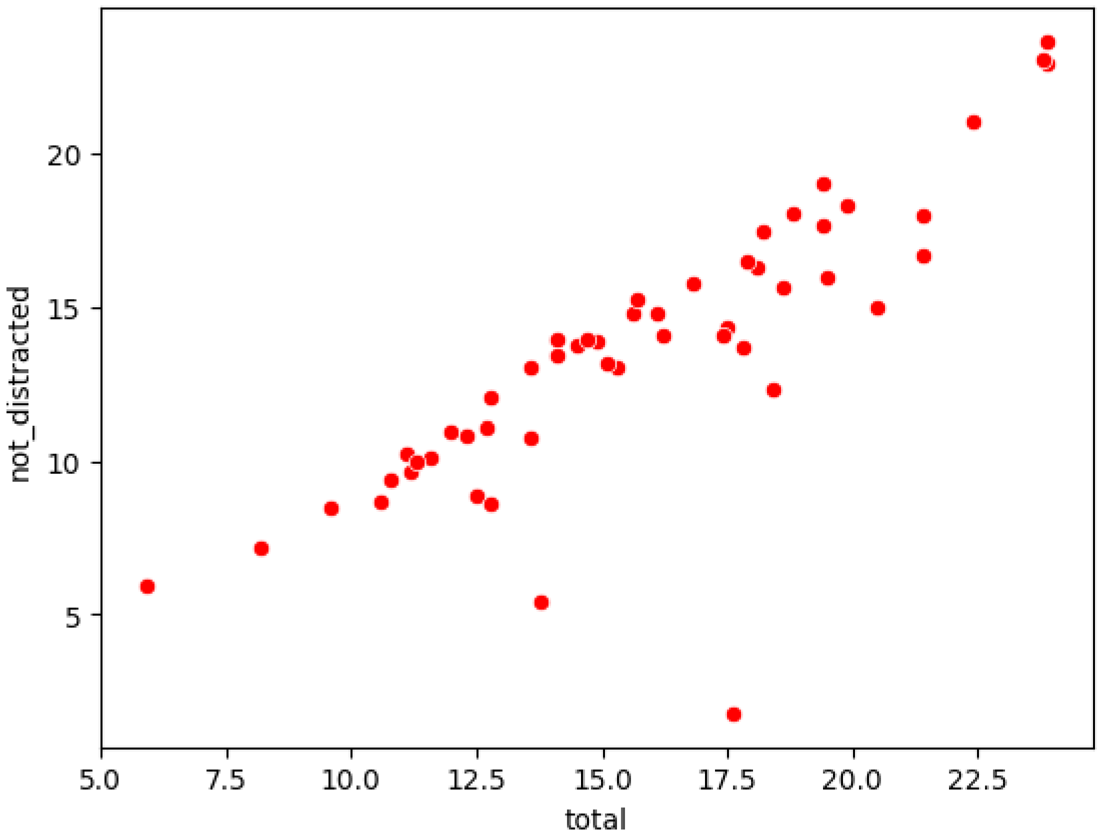
Out[ ]:
```
<Axes: xlabel='total', ylabel='speeding'>
```



Inference:from the plot we can say that as speeding increases total car_crashes accident increases and on an avergae accidents were ocuuring at positionn between 4 to 7

In [ ]:
```
sns.scatterplot(x="total",y="not_distracted",data=df,color="red")
```

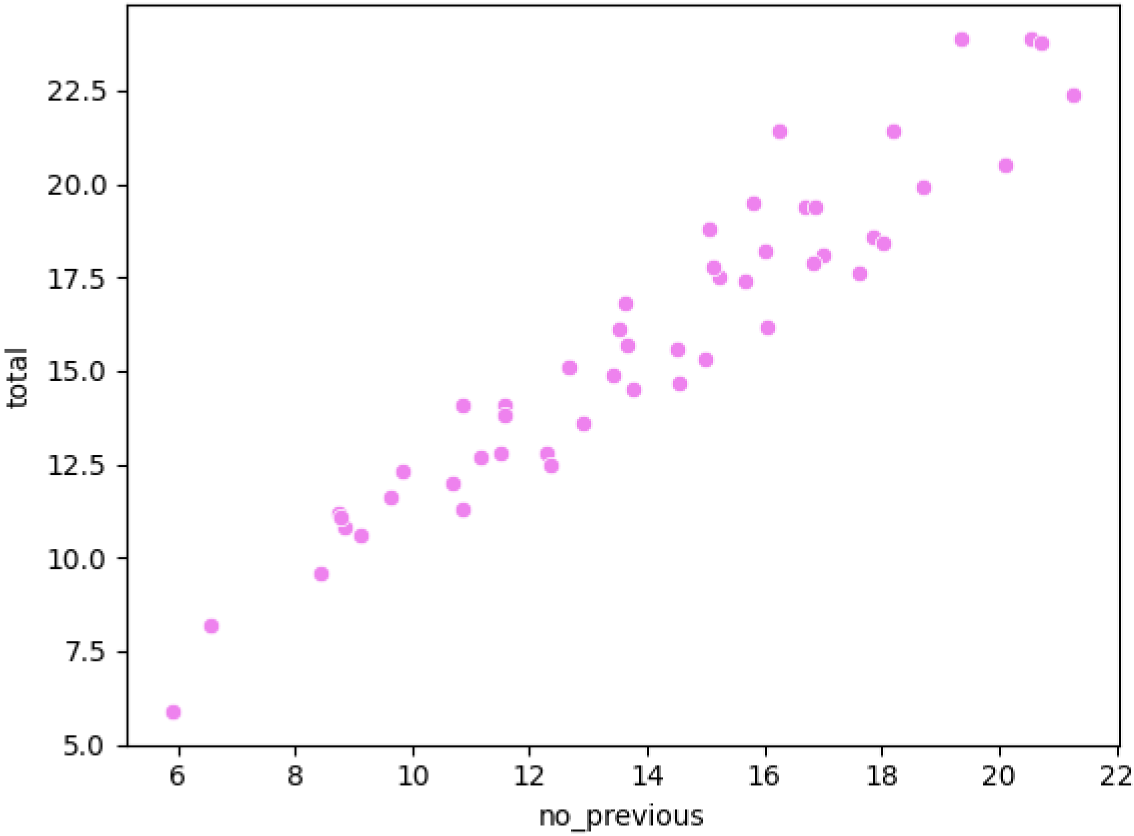
Out[ ]: `<Axes: xlabel='total', ylabel='not_distracted'>`



Inference: its shows a postive correlation whic represents the relation bw variables are propotionally incresing and aslo gives a best fit line

In [ ]:
```
sns.scatterplot(x="no_previous",y="total",data=df,color="violet")
```

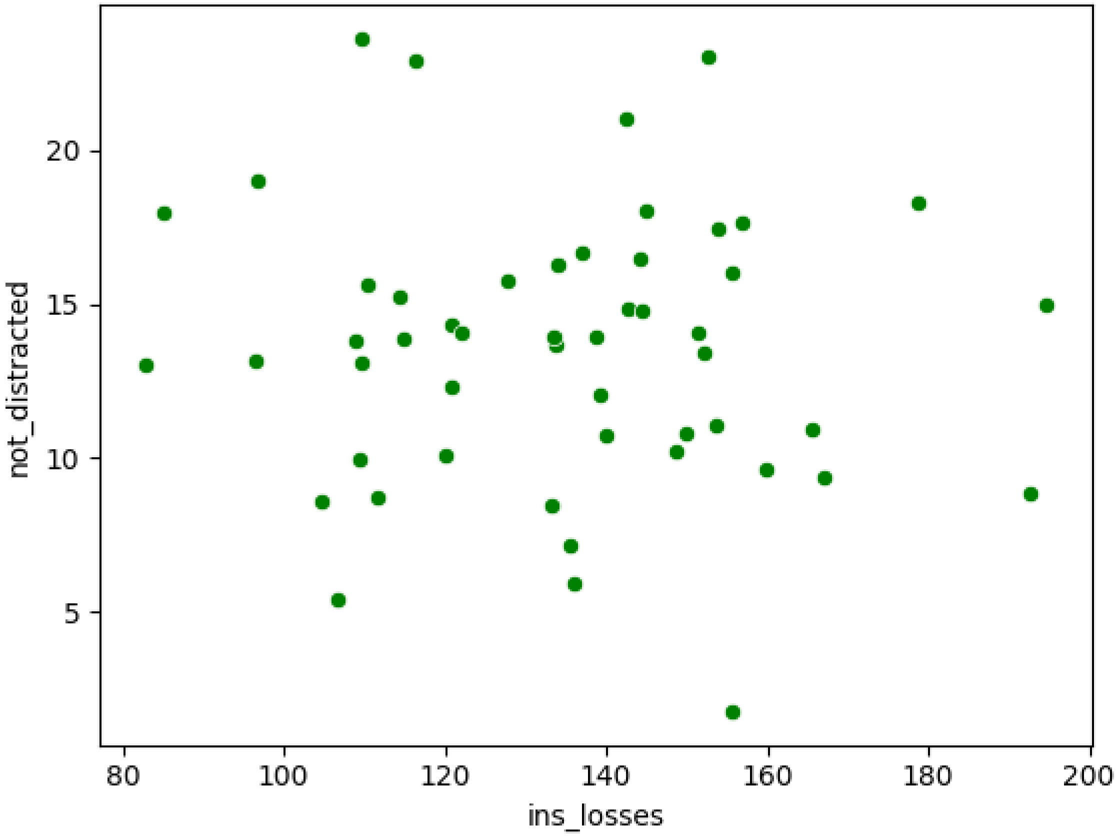
Out[ ]:
```
<Axes: xlabel='no_previous', ylabel='total'>
```



Inference: we can conclude that as total accidents and no_previous increases propotionally which shows positive relationship. it is possible to find a line of best fit

In [ ]:
```
sns.scatterplot(x="ins_losses",y="not_distracted",data=df,color="green")
```

Out[ ]:
```
<Axes: xlabel='ins_losses', ylabel='not_distracted'>
```

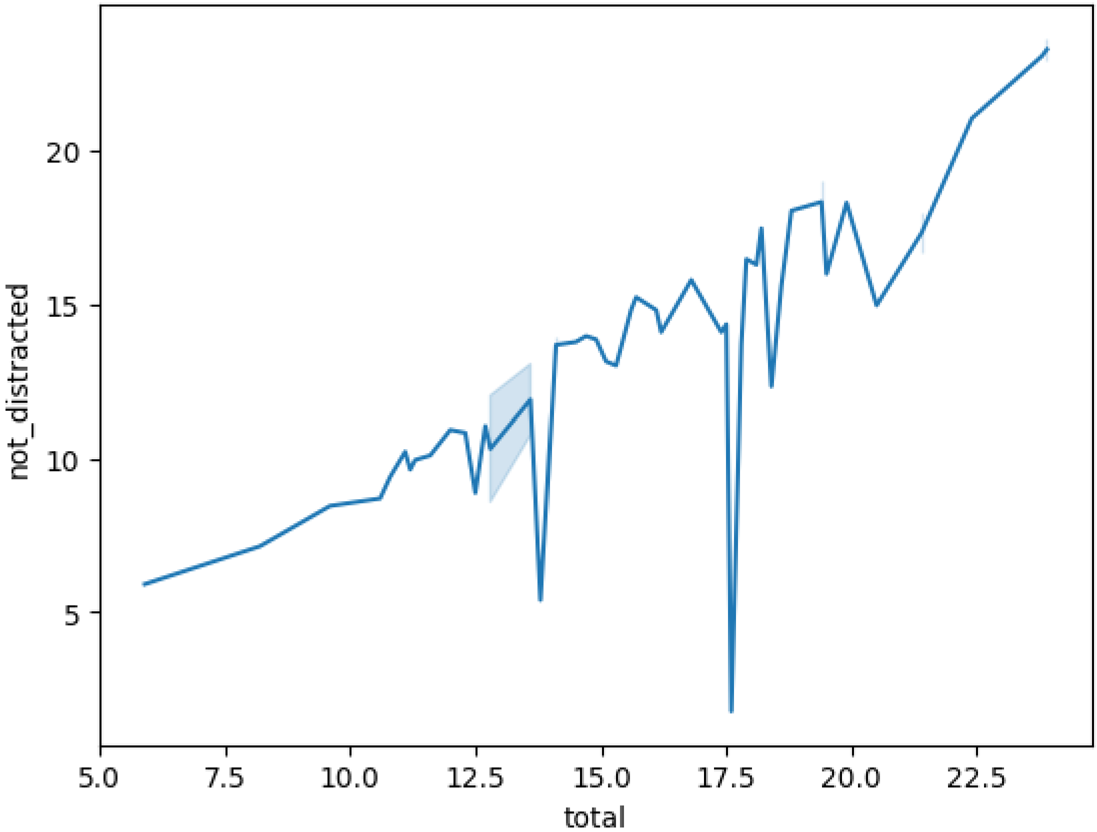


Inference: as ins_losses for company increases the not_distracted decreases which shows negative relationship.its a negative correlation

Line plot

In [ ]:
```
sns.lineplot(x="total",y="not_distracted",data=df)
```

Out[ ]:
```
<Axes: xlabel='total', ylabel='not_distracted'>
```

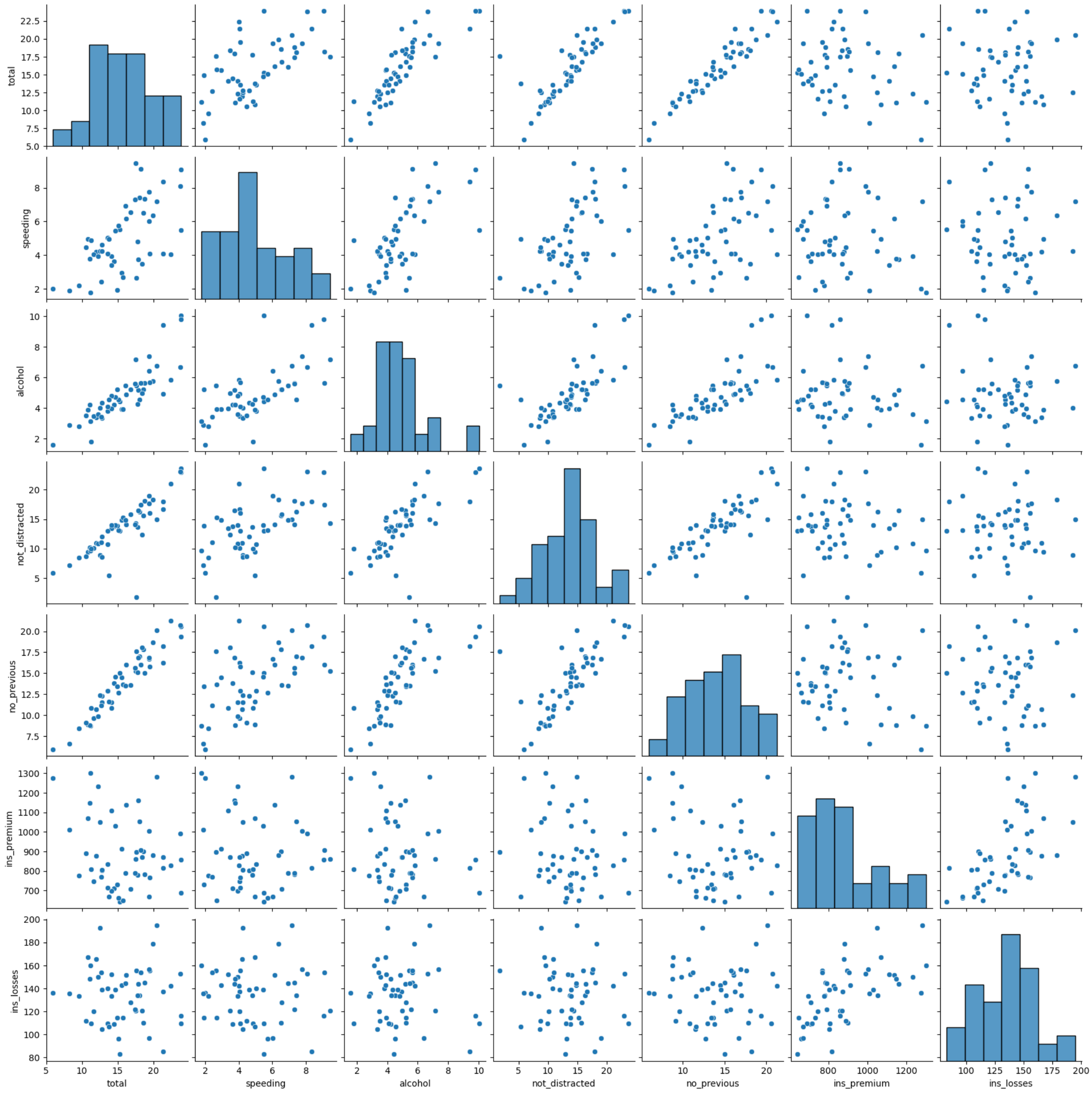


Inference: it can be inferenced that even though it has ups and downs, total number of crashes happened vs crashes happened even with drivers no distracted graph is increasing graph .

Pair Plot

In [ ]:
```
sns.pairplot(df)
```

Out[ ]:
```
<seaborn.axisgrid.PairGrid at 0x7e73f78a3fa0>
```



Inference: This Pair plot is used to understand the relations between two variables in whole datset,it is a matrix kind of plotting with x axis and y axis taking feature columns and also we can find the best set of features

1. The bars represent the distribution of the data for each variable
2. The dots representing the plotting of respective x axis and y axis :

In [ ]:

Finding correlation for all features

In [ ]:

```
p=df.corr()
p
```

```
<ipython-input-30-3a7aec4522bf>:1: FutureWarning: The default value of numeric_onl
y in DataFrame.corr is deprecated. In a future version, it will default to False.
Select only valid columns or specify the value of numeric_only to silence this war
ning.
  p=df.corr()
```

Out[ ]:

	total	speeding	alcohol	not_distracted	no_previous	ins_premium	ins_losse
total	1.000000	0.611548	0.852613	0.827560	0.956179	-0.199702	-0.03601
speeding	0.611548	1.000000	0.669719	0.588010	0.571976	-0.077675	-0.06592
alcohol	0.852613	0.669719	1.000000	0.732816	0.783520	-0.170612	-0.11254
not_distracted	0.827560	0.588010	0.732816	1.000000	0.747307	-0.174856	-0.07597
no_previous	0.956179	0.571976	0.783520	0.747307	1.000000	-0.156895	-0.00635
ins_premium	-0.199702	-0.077675	-0.170612	-0.174856	-0.156895	1.000000	0.62311
ins_losses	-0.036011	-0.065928	-0.112547	-0.075970	-0.006359	0.623116	1.00000

Inference: It gives data from the region of -1 to 1 where greater than 0 can be considered as positively correlated and less than 0 are considered as neagtively corelated.From above premium insurance and intial loses are independent variables so they were negatively correlated.Speeding and alcohol are high positively correlated and not_distracted attribute is positively correlated

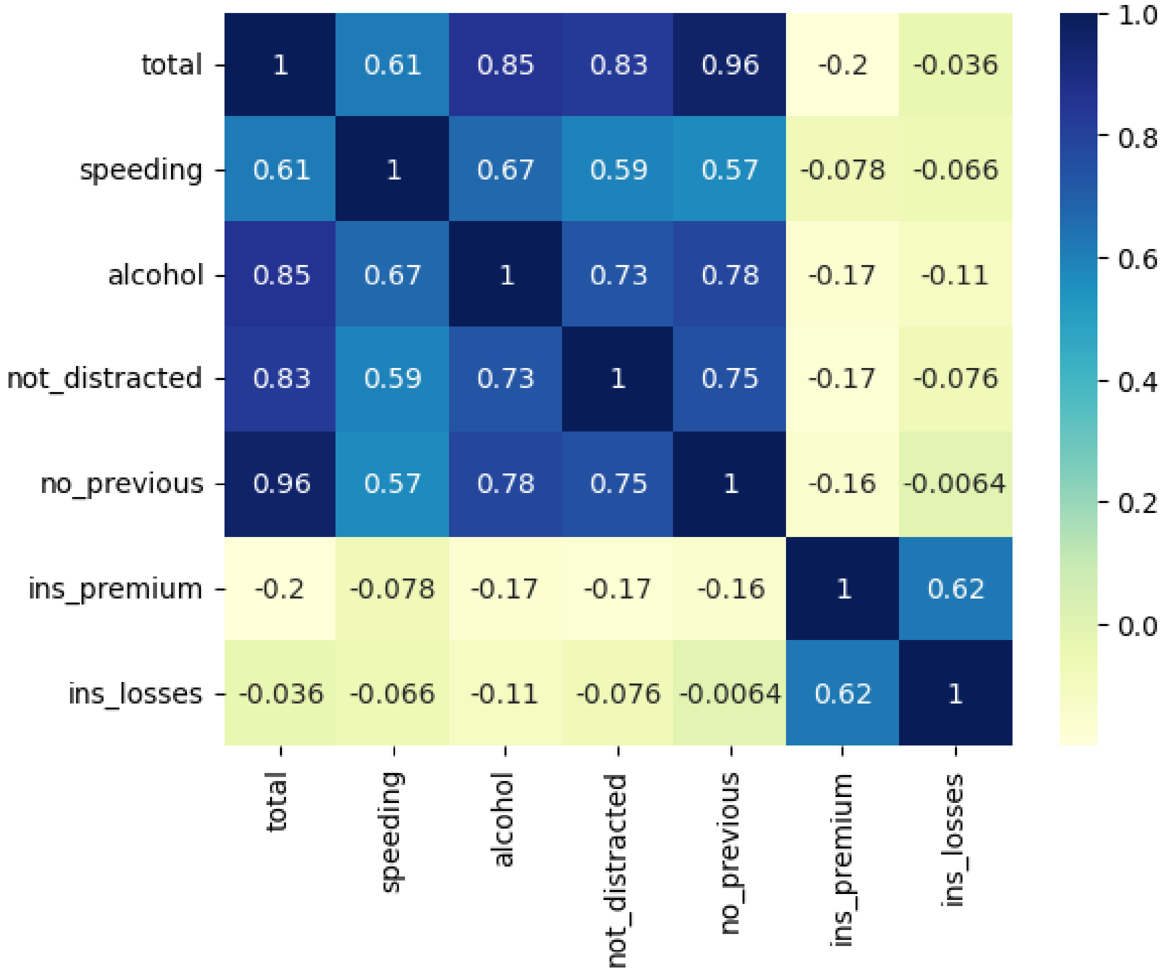
Heat Map

In [ ]:

```
sns.heatmap(p,annot=True,cmap="YlGnBu")
```

Out[ ]:

```
<Axes: >
```



Inference: In the correlation heatmap,we can identify strong positive or negative correlations between variables.This represent the strongest positive correlation is found between the features "total" and "no_previous".

The negative correlation is found between no_previous histrory of accidents and insurance loss of company.That is,no previous accident histroy does not affect the losses of the insurance companies.

We can get carcrashes more precisely like higher the speeding there is a chance of more likely to have accident.In this extreme values can be seen in dark blue and minimal values are seen in light green