In [1]:
```
                              # Name : PICHAKALA JNANA ARUN KUMAR
                              #Reg-No : 21BCE9146

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

In [2]:
```
sns.get_dataset_names()
```

Out[2]:
```
['anagrams',
 'anscombe',
 'attention',
 'brain_networks',
 'car_crashes',
 'diamonds',
 'dots',
 'dowjones',
 'exercise',
 'flights',
 'fmri',
 'geyser',
 'glue',
 'healthexp',
 'iris',
 'mpg',
 'penguins',
 'planets',
 'seaice',
 'taxis',
 'tips',
 'titanic']
```

In [3]:
```
df = sns.load_dataset("car_crashes")
```

In [4]:
```
df
```

Out[4]:

	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
5	13.6	5.032	3.808	10.744	12.920	835.50	139.91	CO
6	10.8	4.968	3.888	9.396	8.856	1068.73	167.02	CT
7	16.2	6.156	4.860	14.094	16.038	1137.87	151.48	DE
8	5.9	2.006	1.593	5.900	5.900	1273.89	136.05	DC
9	17.9	3.759	5.191	16.468	16.826	1160.13	144.18	FL
10	15.6	2.964	3.900	14.820	14.508	913.15	142.80	GA
11	17.5	9.450	7.175	14.350	15.225	861.18	120.92	HI
12	15.3	5.508	4.437	13.005	14.994	641.96	82.75	ID
13	12.8	4.608	4.352	12.032	12.288	803.11	139.15	IL
14	14.5	3.625	4.205	13.775	13.775	710.46	108.92	IN
15	15.7	2.669	3.925	15.229	13.659	649.06	114.47	IA
16	17.8	4.806	4.272	13.706	15.130	780.45	133.80	KS
17	21.4	4.066	4.922	16.692	16.264	872.51	137.13	KY
18	20.5	7.175	6.765	14.965	20.090	1281.55	194.78	LA
19	15.1	5.738	4.530	13.137	12.684	661.88	96.57	ME
20	12.5	4.250	4.000	8.875	12.375	1048.78	192.70	MD
21	8.2	1.886	2.870	7.134	6.560	1011.14	135.63	MA
22	14.1	3.384	3.948	13.395	10.857	1110.61	152.26	MI
23	9.6	2.208	2.784	8.448	8.448	777.18	133.35	MN
24	17.6	2.640	5.456	1.760	17.600	896.07	155.77	MS
25	16.1	6.923	5.474	14.812	13.524	790.32	144.45	MO
26	21.4	8.346	9.416	17.976	18.190	816.21	85.15	MT
27	14.9	1.937	5.215	13.857	13.410	732.28	114.82	NE
28	14.7	5.439	4.704	13.965	14.553	1029.87	138.71	NV
29	11.6	4.060	3.480	10.092	9.628	746.54	120.21	NH
30	11.2	1.792	3.136	9.632	8.736	1301.52	159.85	NJ
31	18.4	3.496	4.968	12.328	18.032	869.85	120.75	NM
32	12.3	3.936	3.567	10.824	9.840	1234.31	150.01	NY
33	16.8	6.552	5.208	15.792	13.608	708.24	127.82	NC
34	23.9	5.497	10.038	23.661	20.554	688.75	109.72	ND

	total	speeding	alcohol	not_distracted	no_previous	ins_premium	ins_losses	abbrev
35	14.1	3.948	4.794	13.959	11.562	697.73	133.52	OH
36	19.9	6.368	5.771	18.308	18.706	881.51	178.86	OK
37	12.8	4.224	3.328	8.576	11.520	804.71	104.61	OR
38	18.2	9.100	5.642	17.472	16.016	905.99	153.86	PA
39	11.1	3.774	4.218	10.212	8.769	1148.99	148.58	RI
40	23.9	9.082	9.799	22.944	19.359	858.97	116.29	SC
41	19.4	6.014	6.402	19.012	16.684	669.31	96.87	SD
42	19.5	4.095	5.655	15.990	15.795	767.91	155.57	TN
43	19.4	7.760	7.372	17.654	16.878	1004.75	156.83	TX
44	11.3	4.859	1.808	9.944	10.848	809.38	109.48	UT
45	13.6	4.080	4.080	13.056	12.920	716.20	109.61	VT
46	12.7	2.413	3.429	11.049	11.176	768.95	153.72	VA
47	10.6	4.452	3.498	8.692	9.116	890.03	111.62	WA
48	23.8	8.092	6.664	23.086	20.706	992.61	152.56	WV
49	13.8	4.968	4.554	5.382	11.592	670.31	106.62	WI
50	17.4	7.308	5.568	14.094	15.660	791.14	122.04	WY

In [5]:
```
df.info
```

Out[5]:
```
<bound method DataFrame.info of     total  speeding  alcohol  not_distrac
ted  no_previous  ins_premium  \
0    18.8    7.332    5.640      18.048     15.040      784.55
1    18.1    7.421    4.525      16.290     17.014     1053.48
2    18.6    6.510    5.208      15.624     17.856      899.47
3    22.4    4.032    5.824      21.056     21.280      827.34
4    12.0    4.200    3.360      10.920     10.680      878.41
5    13.6    5.032    3.808      10.744     12.920      835.50
6    10.8    4.968    3.888       9.396      8.856     1068.73
7    16.2    6.156    4.860      14.094     16.038     1137.87
8     5.9    2.006    1.593       5.900      5.900     1273.89
9    17.9    3.759    5.191      16.468     16.826     1160.13
10   15.6    2.964    3.900      14.820     14.508      913.15
11   17.5    9.450    7.175      14.350     15.225      861.18
12   15.3    5.508    4.437      13.005     14.994      641.96
13   12.8    4.608    4.352      12.032     12.288      803.11
14   14.5    3.625    4.205      13.775     13.775      710.46
15   15.7    2.669    3.925      15.229     13.659      649.06
16   17.8    4.806    4.272      13.706     15.130      780.45
17   21.4    4.066    4.922      16.692     16.264      872.51
18   20.5    7.175    6.765      14.965     20.090     1281.55
19   15.1    5.738    4.530      13.137     12.684      661.88
20   12.5    4.250    4.000       8.875     12.375     1048.78
21    8.2    1.886    2.870       7.134      6.560     1011.14
22   14.1    3.384    3.948      13.395     10.857     1110.61
23    9.6    2.208    2.784       8.448      8.448      777.18
24   17.6    2.640    5.456       1.760     17.600      896.07
25   16.1    6.923    5.474      14.812     13.524      790.32
26   21.4    8.346    9.416      17.976     18.190      816.21
27   14.9    1.937    5.215      13.857     13.410      732.28
28   14.7    5.439    4.704      13.965     14.553     1029.87
29   11.6    4.060    3.480      10.092      9.628      746.54
30   11.2    1.792    3.136       9.632      8.736     1301.52
31   18.4    3.496    4.968      12.328     18.032      869.85
32   12.3    3.936    3.567      10.824      9.840     1234.31
33   16.8    6.552    5.208      15.792     13.608      708.24
34   23.9    5.497   10.038      23.661     20.554      688.75
35   14.1    3.948    4.794      13.959     11.562      697.73
36   19.9    6.368    5.771      18.308     18.706      881.51
37   12.8    4.224    3.328       8.576     11.520      804.71
38   18.2    9.100    5.642      17.472     16.016      905.99
39   11.1    3.774    4.218      10.212      8.769     1148.99
40   23.9    9.082    9.799      22.944     19.359      858.97
41   19.4    6.014    6.402      19.012     16.684      669.31
42   19.5    4.095    5.655      15.990     15.795      767.91
43   19.4    7.760    7.372      17.654     16.878     1004.75
44   11.3    4.859    1.808       9.944     10.848      809.38
45   13.6    4.080    4.080      13.056     12.920      716.20
46   12.7    2.413    3.429      11.049     11.176      768.95
47   10.6    4.452    3.498       8.692      9.116      890.03
48   23.8    8.092    6.664      23.086     20.706      992.61
49   13.8    4.968    4.554       5.382     11.592      670.31
50   17.4    7.308    5.568      14.094     15.660      791.14

    ins_losses abbrev
0       145.08     AL
1       133.93     AK
```

2	110.35	AZ
3	142.39	AR
4	165.63	CA
5	139.91	CO
6	167.02	CT
7	151.48	DE
8	136.05	DC
9	144.18	FL
10	142.80	GA
11	120.92	HI
12	82.75	ID
13	139.15	IL
14	108.92	IN
15	114.47	IA
16	133.80	KS
17	137.13	KY
18	194.78	LA
19	96.57	ME
20	192.70	MD
21	135.63	MA
22	152.26	MI
23	133.35	MN
24	155.77	MS
25	144.45	MO
26	85.15	MT
27	114.82	NE
28	138.71	NV
29	120.21	NH
30	159.85	NJ
31	120.75	NM
32	150.01	NY
33	127.82	NC
34	109.72	ND
35	133.52	OH
36	178.86	OK
37	104.61	OR
38	153.86	PA
39	148.58	RI
40	116.29	SC
41	96.87	SD
42	155.57	TN
43	156.83	TX
44	109.48	UT
45	109.61	VT
46	153.72	VA
47	111.62	WA
48	152.56	WV
49	106.62	WI
50	122.04	WY >

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

Out[6]:

	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

In [8]:
```
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 [9]:
```
df.head() #retrives first five records
```

Out[9]:

	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

In [10]:
```
df.tail()  #retrives last five records
```

Out[10]:

	total	speeding	alcohol	not_distracted	no_previous	ins_premium	ins_losses	abbrev
46	12.7	2.413	3.429	11.049	11.176	768.95	153.72	VA
47	10.6	4.452	3.498	8.692	9.116	890.03	111.62	WA
48	23.8	8.092	6.664	23.086	20.706	992.61	152.56	WV
49	13.8	4.968	4.554	5.382	11.592	670.31	106.62	WI
50	17.4	7.308	5.568	14.094	15.660	791.14	122.04	WY

In [11]:
```
#if you want to retrive first three records in dataset we can give as df.he
```

In [12]:
```
df.head(3)
```

Out[12]:

	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

In [13]:
```
df.isnull().sum()
```

Out[13]:
```
total             0
speeding          0
alcohol           0
not_distracted    0
no_previous       0
ins_premium       0
ins_losses        0
abbrev            0
dtype: int64
```

In [14]:
```
df.isnull().any()
```

Out[14]:
```
total             False
speeding          False
alcohol           False
not_distracted    False
no_previous       False
ins_premium       False
ins_losses        False
abbrev            False
dtype: bool
```

In [15]:
```
#let us the find the correlation
```

In [16]:
```
cor = df.corr()
```

In [17]:
```
cor
```

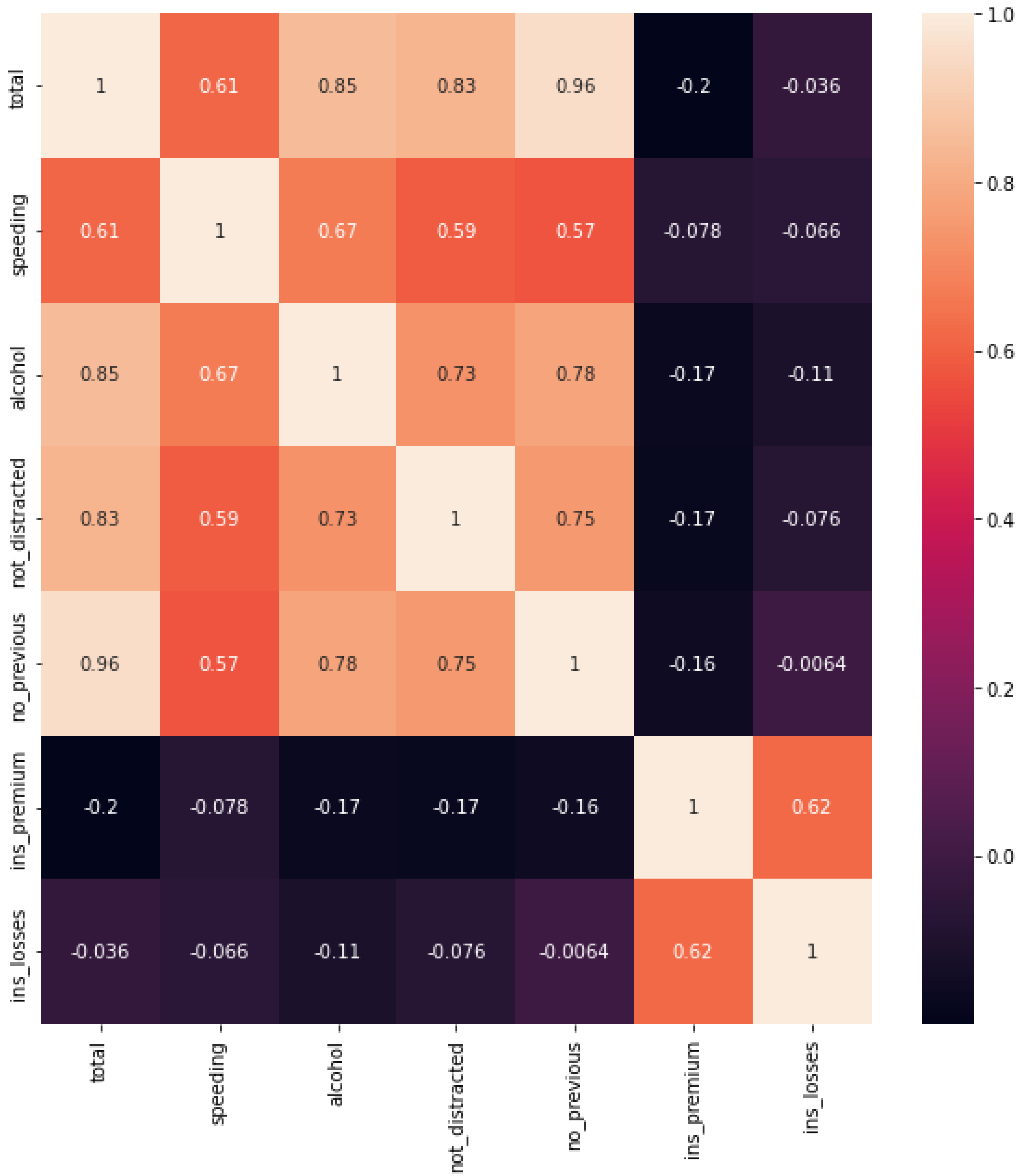
Out[17]:

	total	speeding	alcohol	not_distracted	no_previous	ins_premium	ins_losses
total	1.000000	0.611548	0.852613	0.827560	0.956179	-0.199702	-0.036011
speeding	0.611548	1.000000	0.669719	0.588010	0.571976	-0.077675	-0.065928
alcohol	0.852613	0.669719	1.000000	0.732816	0.783520	-0.170612	-0.112547
not_distracted	0.827560	0.588010	0.732816	1.000000	0.747307	-0.174856	-0.075970
no_previous	0.956179	0.571976	0.783520	0.747307	1.000000	-0.156895	-0.006359
ins_premium	-0.199702	-0.077675	-0.170612	-0.174856	-0.156895	1.000000	0.623116
ins_losses	-0.036011	-0.065928	-0.112547	-0.075970	-0.006359	0.623116	1.000000

#let us draw the correlation 2d matrix

In [24]:
```
plt.figure(figsize=(10,10))
sns.heatmap(cor,annot=True)
```

Out[24]: <AxesSubplot:>



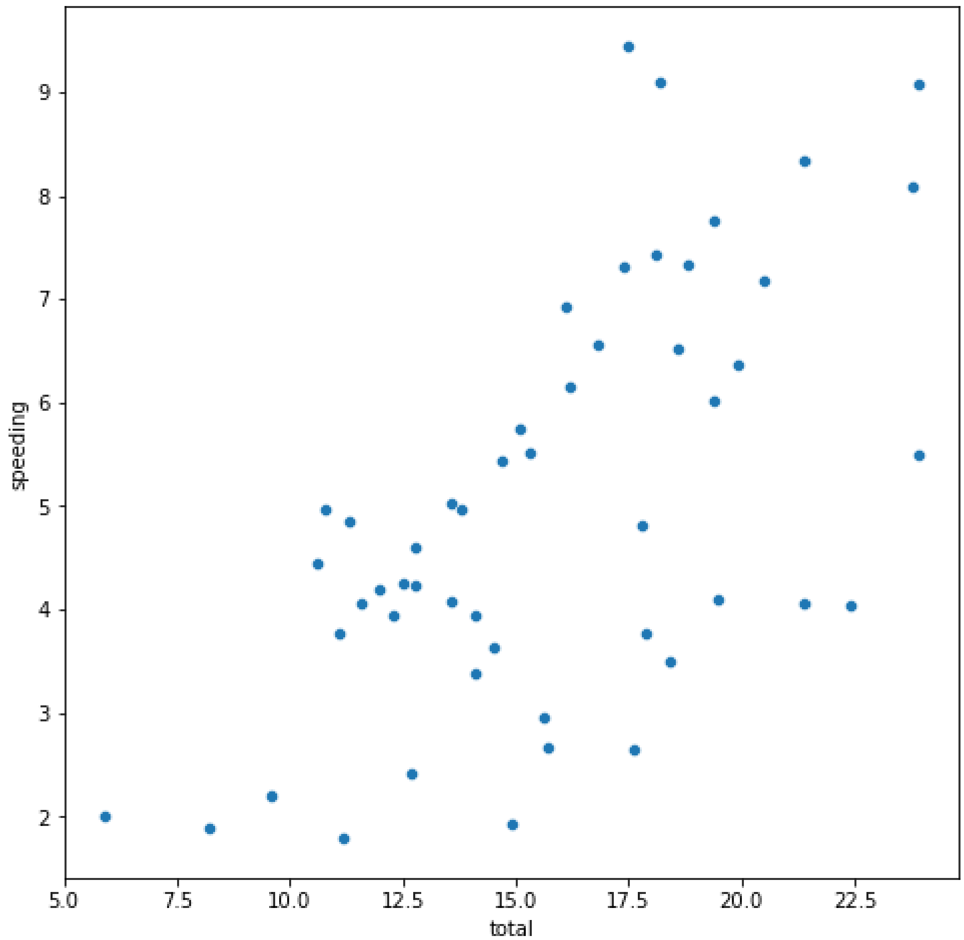
Inference from the above graph : from the above graph some are highly correlated (value >0.5) and some are less correlated (less than <0.5) ex : here both the features total and speeding are highly correlated because the value is greater than 0.61 which is greater than 0.5. if we take the features total and ins_losses they are negativley correlated or we can say they are less correlated because the value is -0.036 which is less than 0.5

scatter plot

plt.figure(figsize=(8,8)) sns.scatterplot(x="total",y="speeding",data=df)

In [47]:
```
plt.figure(figsize=(8,8))
sns.scatterplot(x="total",y="speeding",data=df)
```

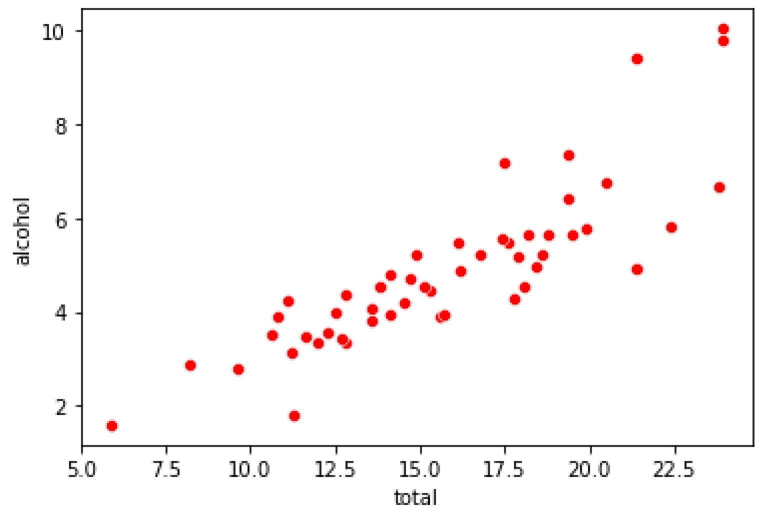
Out[47]: <AxesSubplot:xlabel='total', ylabel='speeding'>



Inference from the above graph : from the above graph we can we can say that the total number of drivers in fatal collisions is linearly propotional percentage of drivers involved in fatal collisons who were speeding

In [33]:
```
sns.scatterplot(x="total",y="alcohol",data=df,color="r")
```

Out[33]: <AxesSubplot:xlabel='total', ylabel='alcohol'>

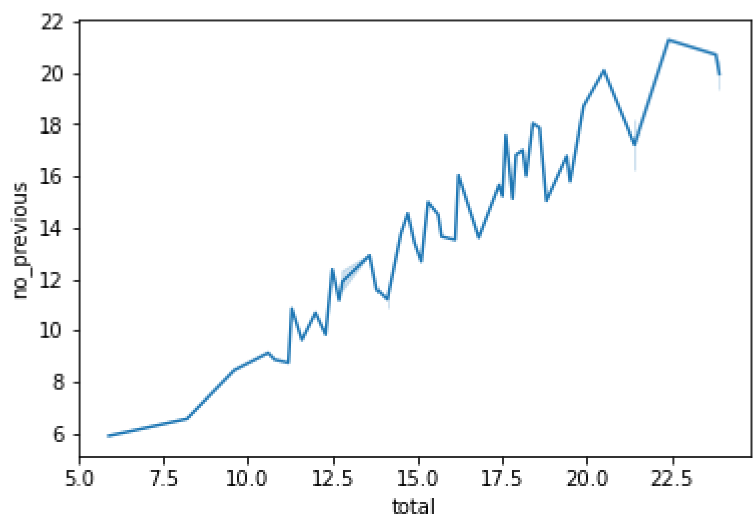


Inference from the above graph : From the above graph we can say that the total number of drivers involved in fatal collisions is linearly propotional percentage drivers involved in fatal collisions who were distracted

sns.lineplot(x="total",y="no_previous",data=df)

In [53]:
```
sns.lineplot(x="total",y="no_previous",data=df)
#LINEPLOT
```

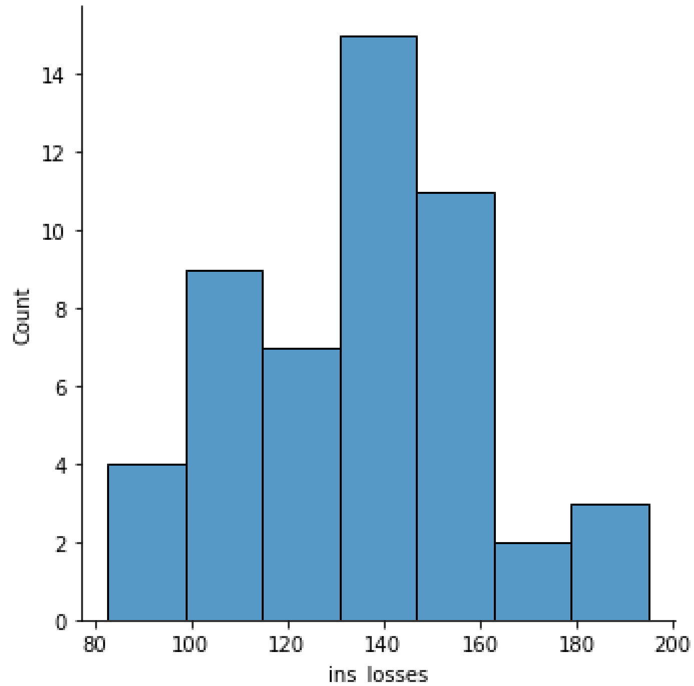
Out[53]: <AxesSubplot:xlabel='total', ylabel='no_previous'>



Inferecne from the above graph : from the above graph we can say that the total number of drivers involved in fatal collisons is linerly propotional to percentage of drivers involved in fatal collisons who do not have previous

In [50]:
```
sns.displot(df["ins_losses"])
```

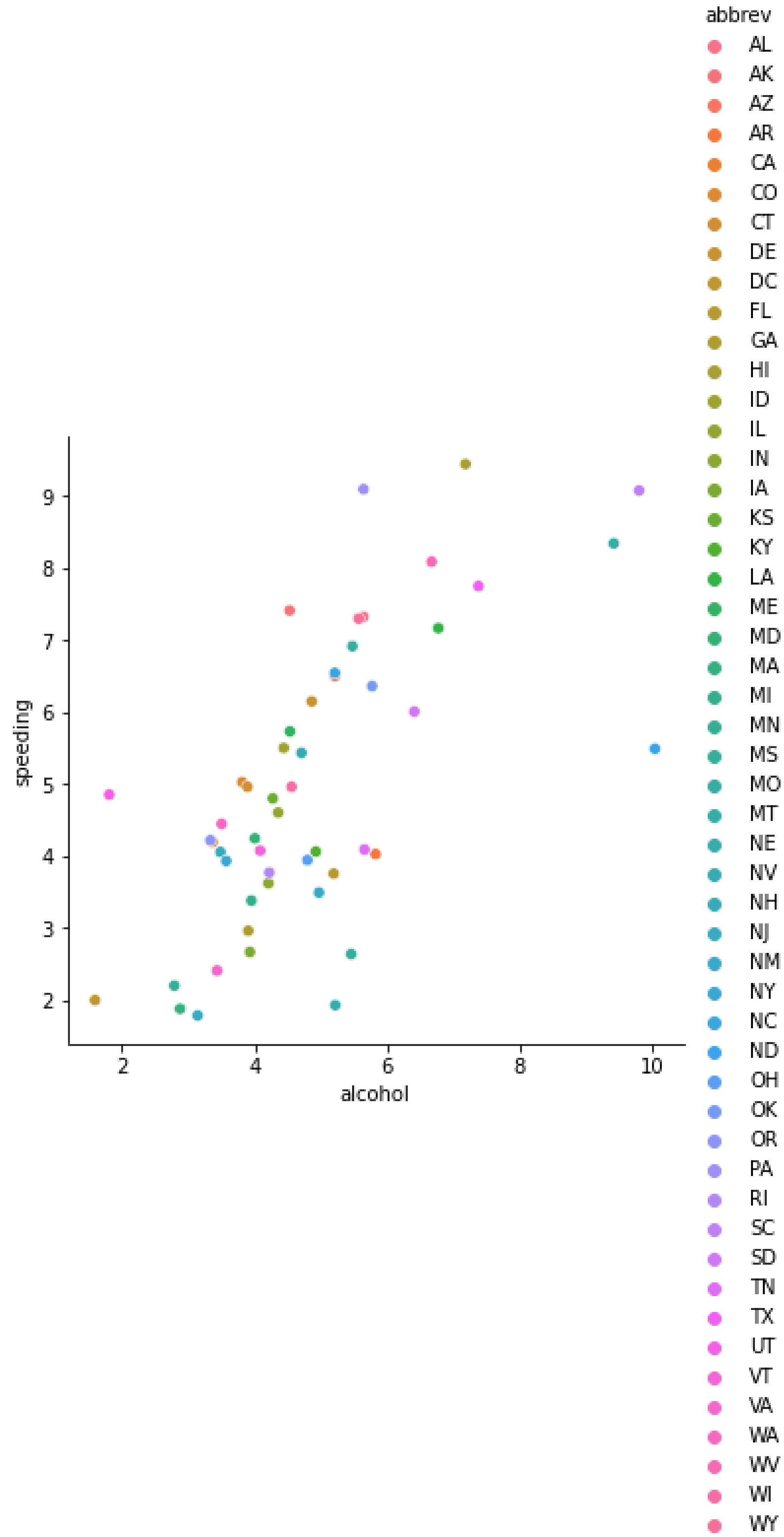
Out[50]: <seaborn.axisgrid.FacetGrid at 0x12211edf0>



Inference from the above graph : from the above graph we can say that ins_losses mostly lies b/w 100 and 160 and highest at 140

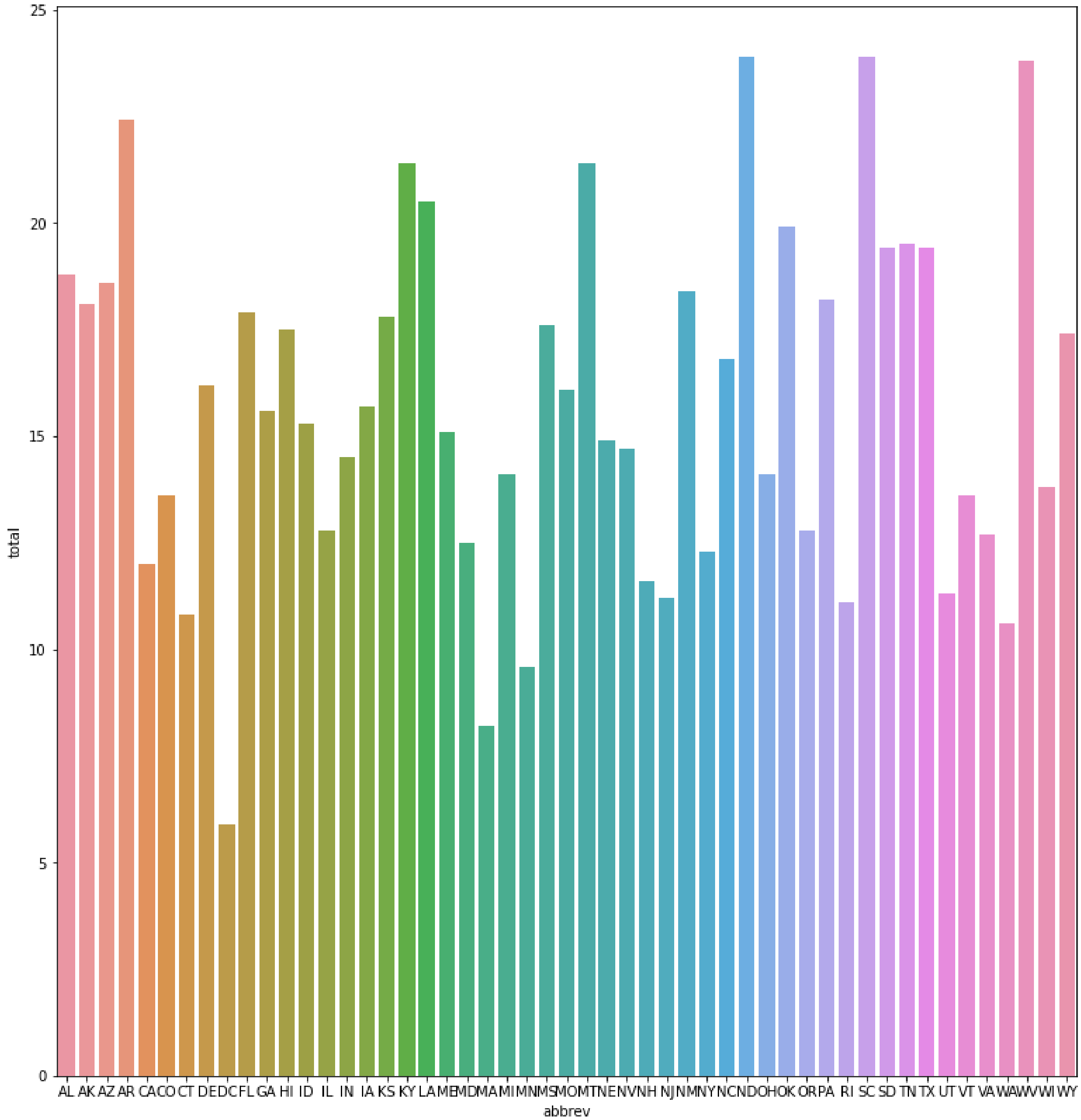
In [54]:
```
#RELPLOT :
sns.relplot(x="alcohol",y="speeding",data=df,hue="abbrev")
```

Out[54]: <seaborn.axisgrid.FacetGrid at 0x1232b58b0>



Inference from the above graph : from the above graph we can say that when alocohol consumption is increasing speeding also increases

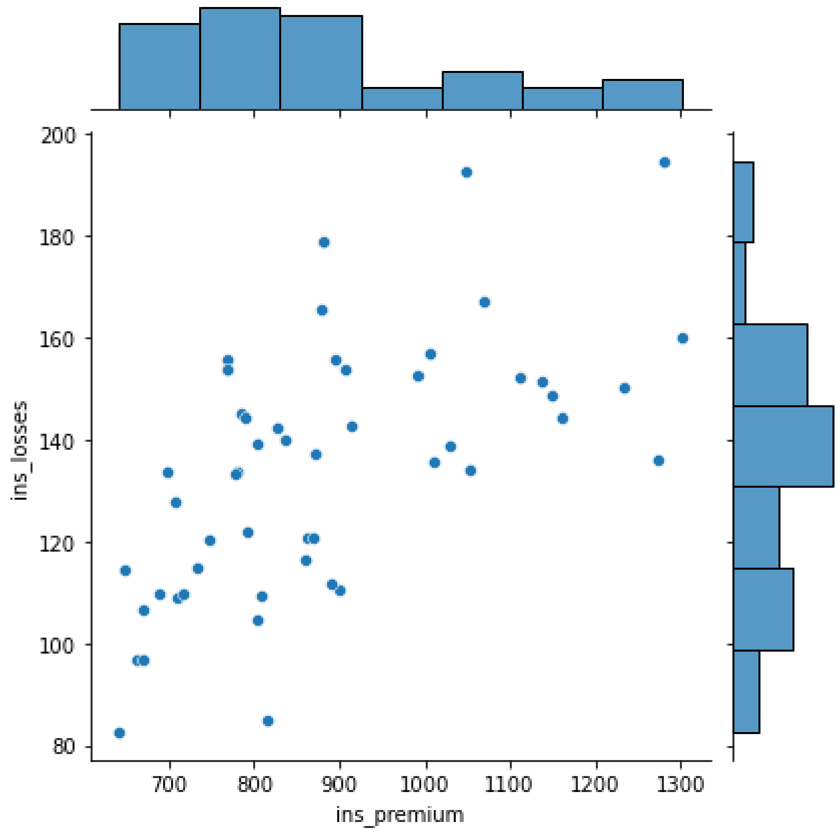
In [56]:
```
#BARPLOT :
plt.figure(figsize=(13,14))
sns.barplot(x="abbrev",y="total",data=df)
plt.show()
```



Inference from the above graph : among all state ND has total no.of highest collisions

In [59]:
```
#JOINTPLOT :
sns.jointplot(x="ins_premium",y="ins_losses",data=df)
```

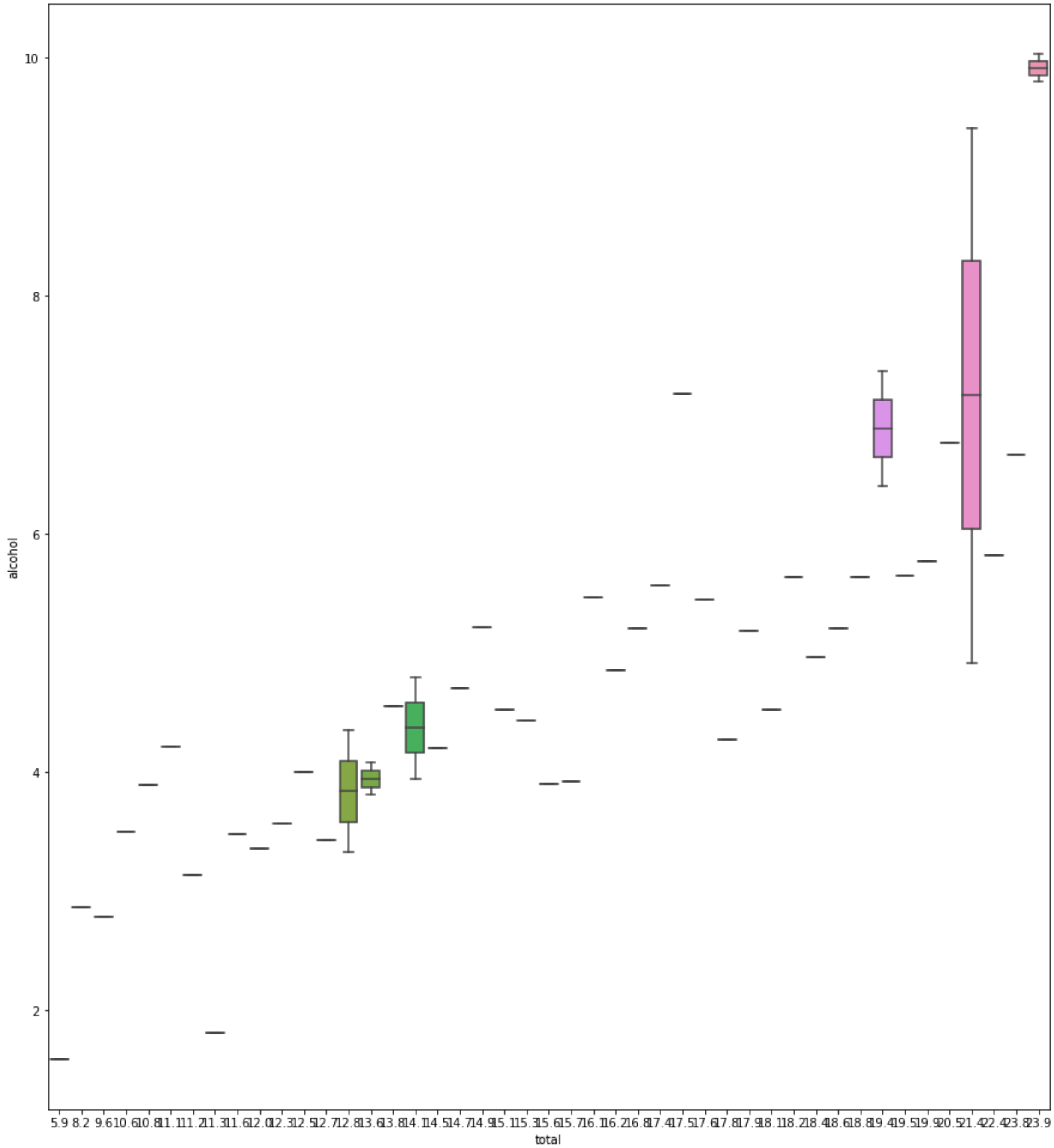
Out[59]: <seaborn.axisgrid.JointGrid at 0x123d41520>



Inference from the above graph : ins_premium and ins_looses are directly propotinsl

In [62]:
```
#BOXPLOT :
plt.figure(figsize=(15,17))
sns.boxplot(x=df["total"],y=df["alcohol"],data=df)
```

Out[62]: <AxesSubplot:xlabel='total', ylabel='alcohol'>



Inferecnce from the above graph : from the above graph we can say that there are no outliers

In [ ]: