Importing seaborn

In [27]:
```
import seaborn as sns
import matplotlib.pyplot as plt
```

Load the dataset of car crashes

In [6]:
```
sns.get_dataset_names()
```

Out[6]:
```
['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 [7]:
```
df=sns.load_dataset('car_crashes')
df
```

Out[7]:

	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
35	14.1	3.948	4.794	13.959	11.562	697.73	133.52	OH

	total	speeding	alcohol	not_distracted	no_previous	ins_premium	ins_losses	abbrev
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 [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
```

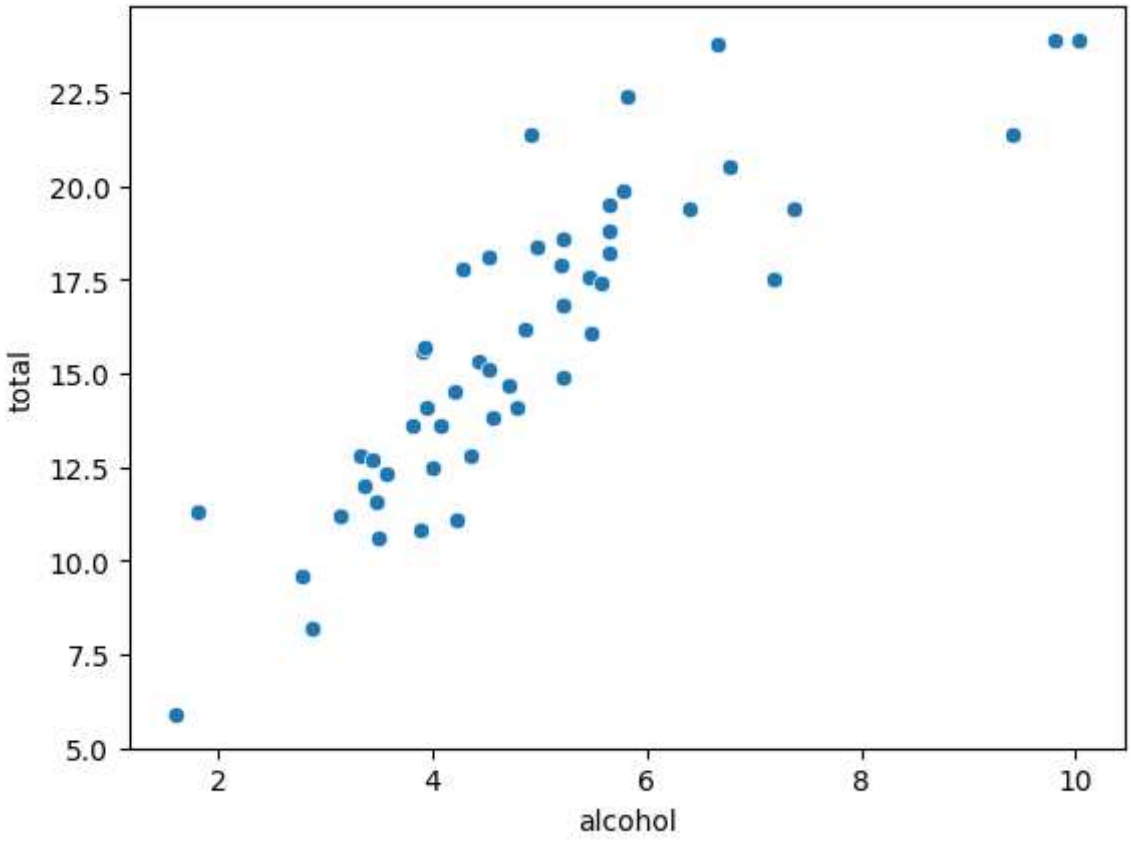
In [9]:
```
df.head()
```

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

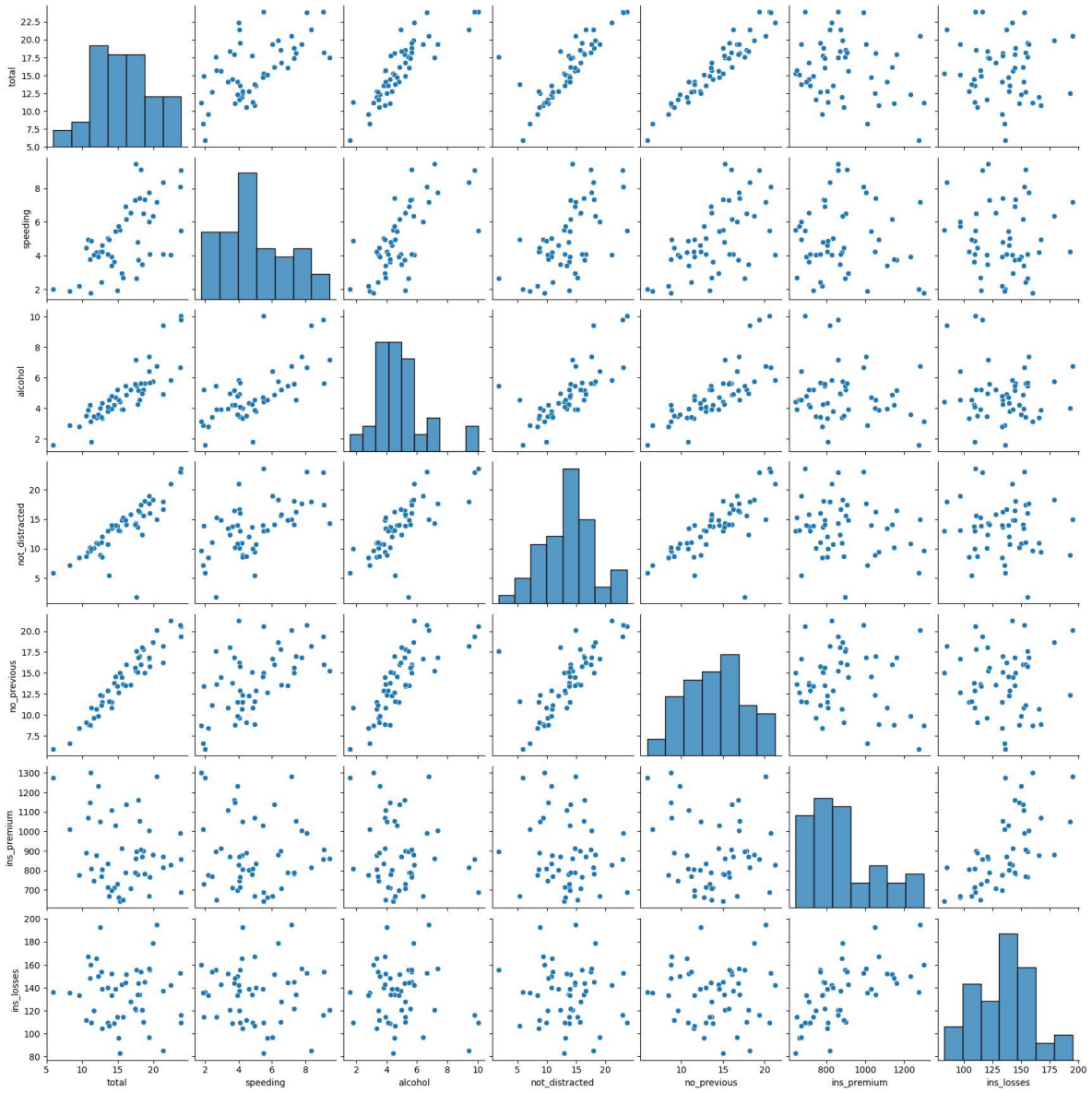
Out[10]:
```
<Axes: xlabel='alcohol', ylabel='total'>
```



Inference: From the plot we can say that as the alcohol related crashes increases then total number of car crashes also increases

In [14]:
```
sns.pairplot(df)
```

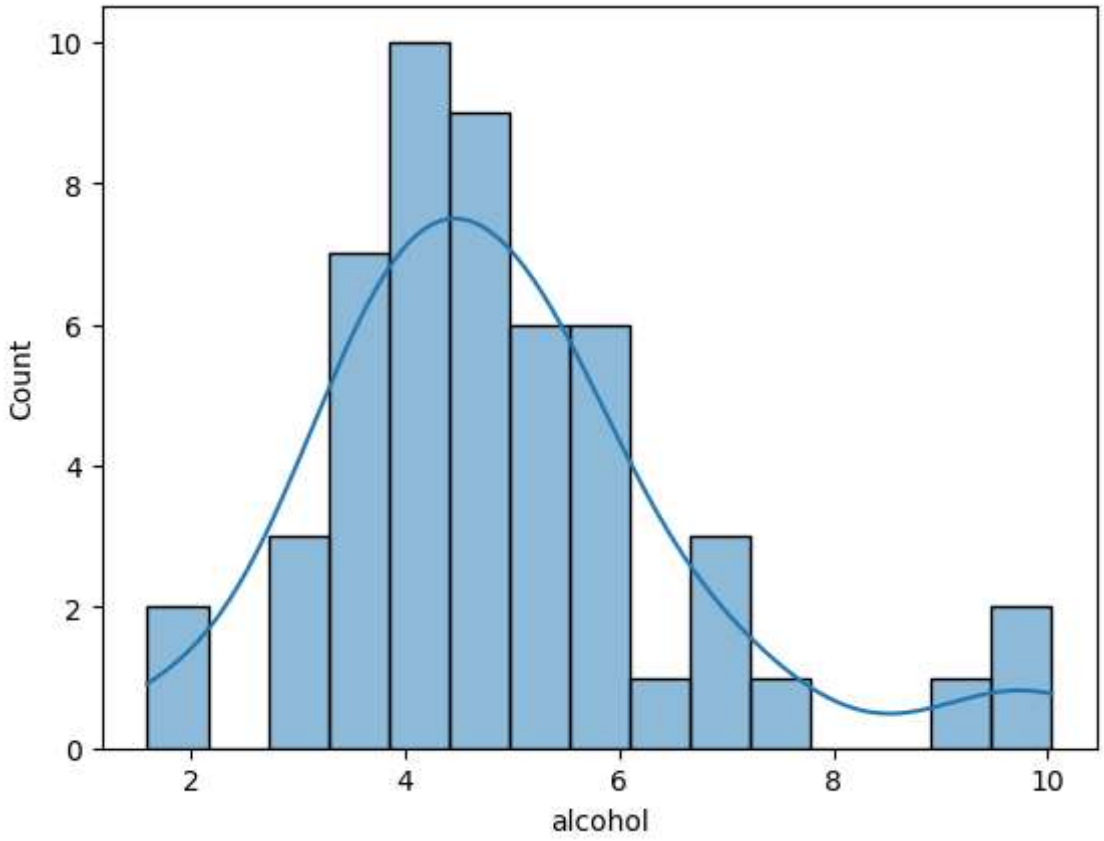
Out[14]: <seaborn.axisgrid.PairGrid at 0x7cf9e1a93e80>



Inference: To show the pairwise relationship in the given dataframe. It creates a matrix of scatterplots, where each variable in the dataset is plotted against every other variable.

In [17]:
```
sns.histplot(df["alcohol"], bins=15, kde=True)
```

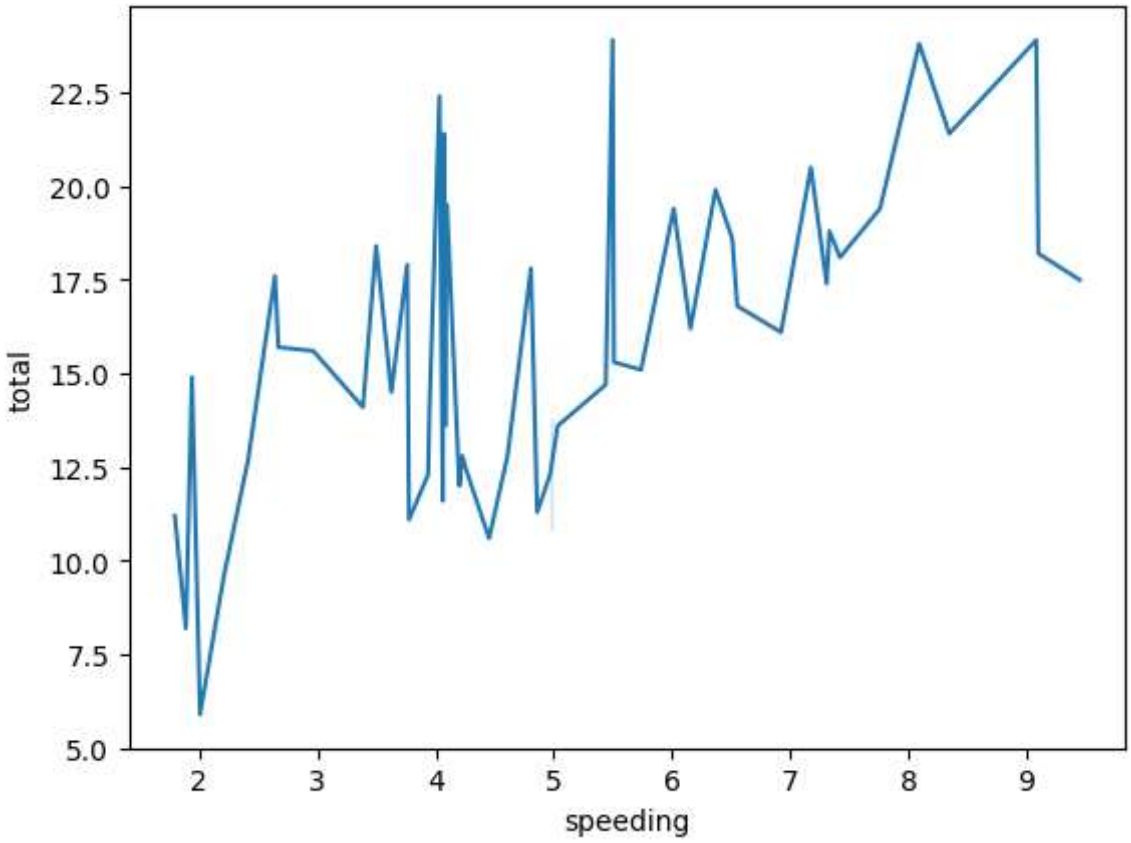
Out[17]:
```
<Axes: xlabel='alcohol', ylabel='Count'>
```



Inference: It shows that most regions have a relatively low number of alcohol-related crashes, with a peak around 5-10 crashes.

In [19]:
```
sns.lineplot(x="speeding", y="total", data=df)
```

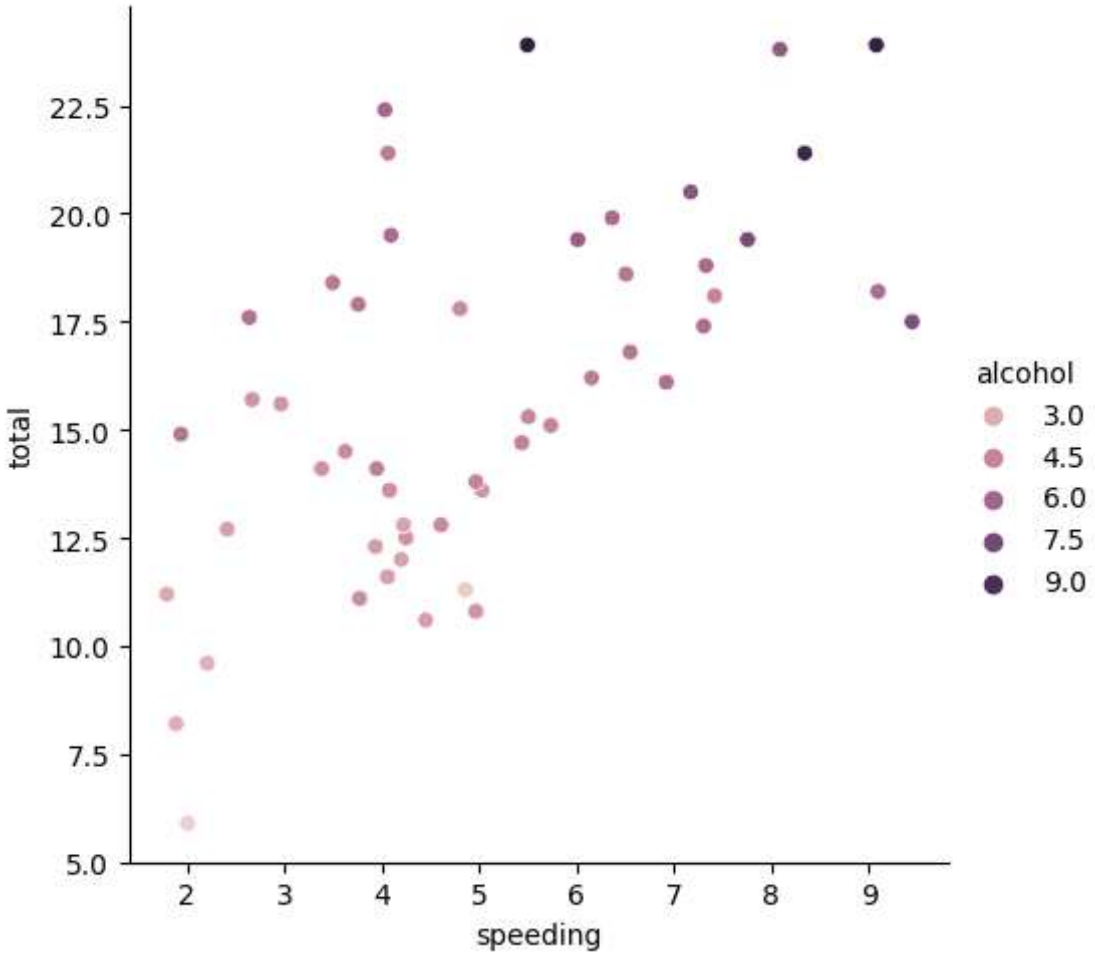
Out[19]:
```
<Axes: xlabel='speeding', ylabel='total'>
```



Inference: As the number speeding based car crashes increases, then the total crashes has also increased

In [21]:
```
sns.relplot(x="speeding", y="total", hue="alcohol", data=df)
```

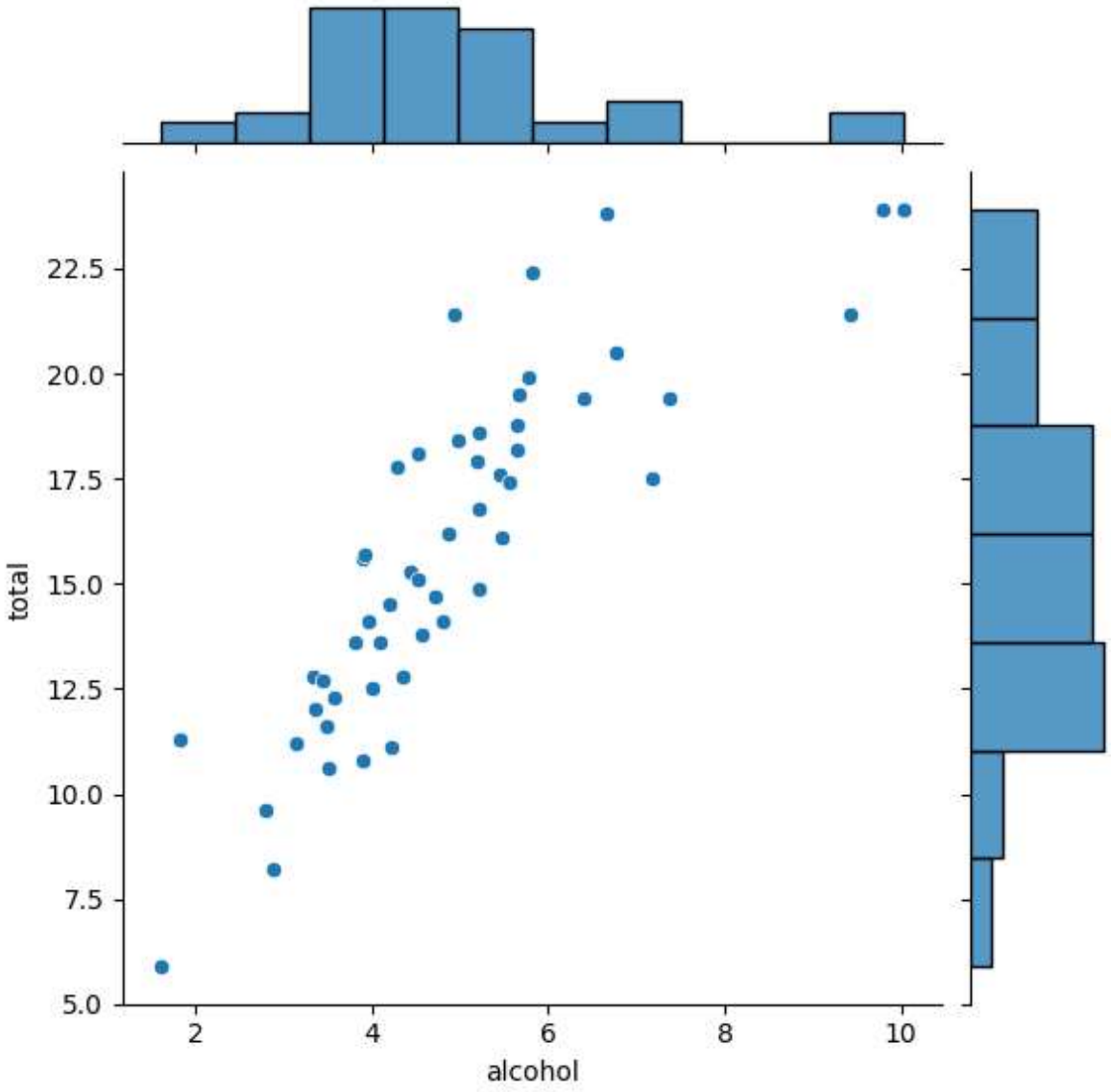
Out[21]:
```
<seaborn.axisgrid.FacetGrid at 0x7cf9e1a91210>
```



Inference: The relplot is used to visualize the relationship between speeding-related crashes (x-axis) and total crashes (y-axis) while differentiating by alcohol-related crashes (color). This allows you to explore how alcohol-related crashes impact the speeding and total crash relationship.

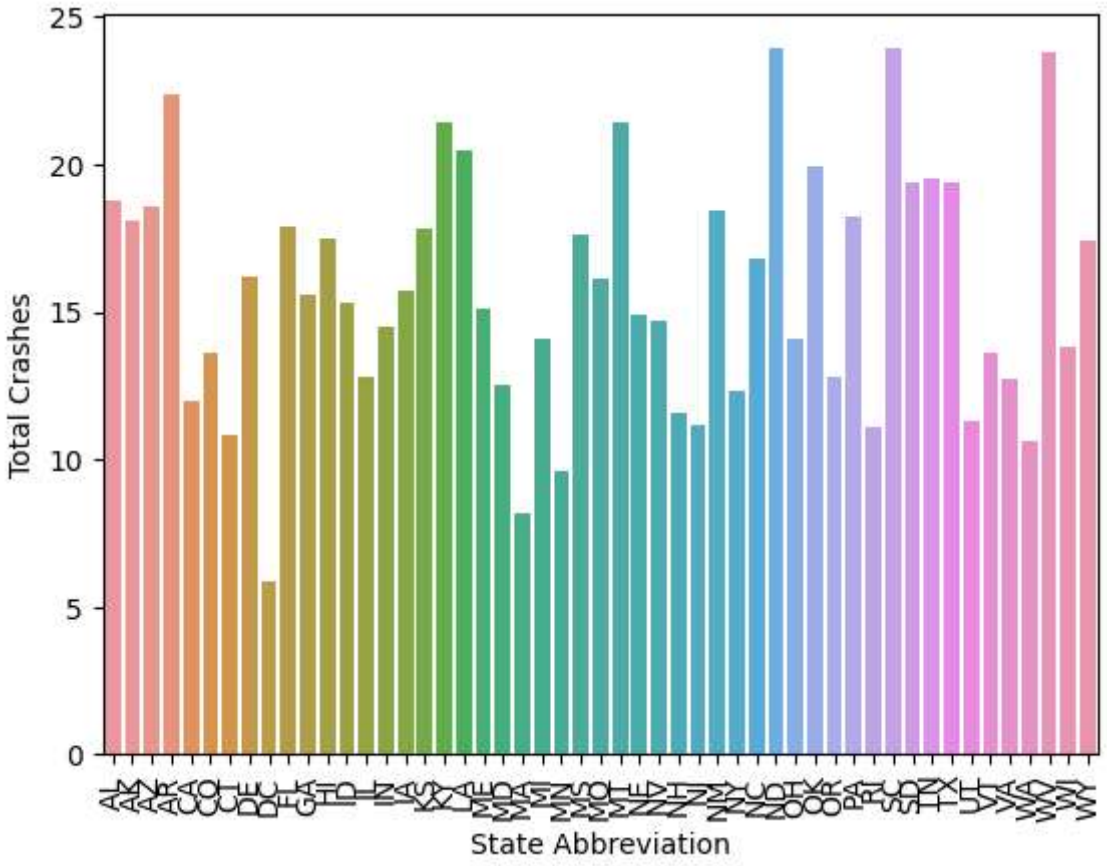
In [23]:
```
sns.jointplot(x="alcohol", y="total", data=df, kind="scatter")
```

Out[23]:
```
<seaborn.axisgrid.JointGrid at 0x7cf9d8603fd0>
```



Inference: The jointplot provides a scatter plot of alcohol-related crashes versus total crashes along with histograms for each variable. This helps in visualizing the bivariate distribution and marginal distributions of the two variables.

In [34]:
```
sns.barplot(x="abbrev", y="total", data=df,width=0.8)
plt.xlabel("State Abbreviation")
plt.ylabel("Total Crashes")
plt.xticks(rotation=90)
plt.show()
```

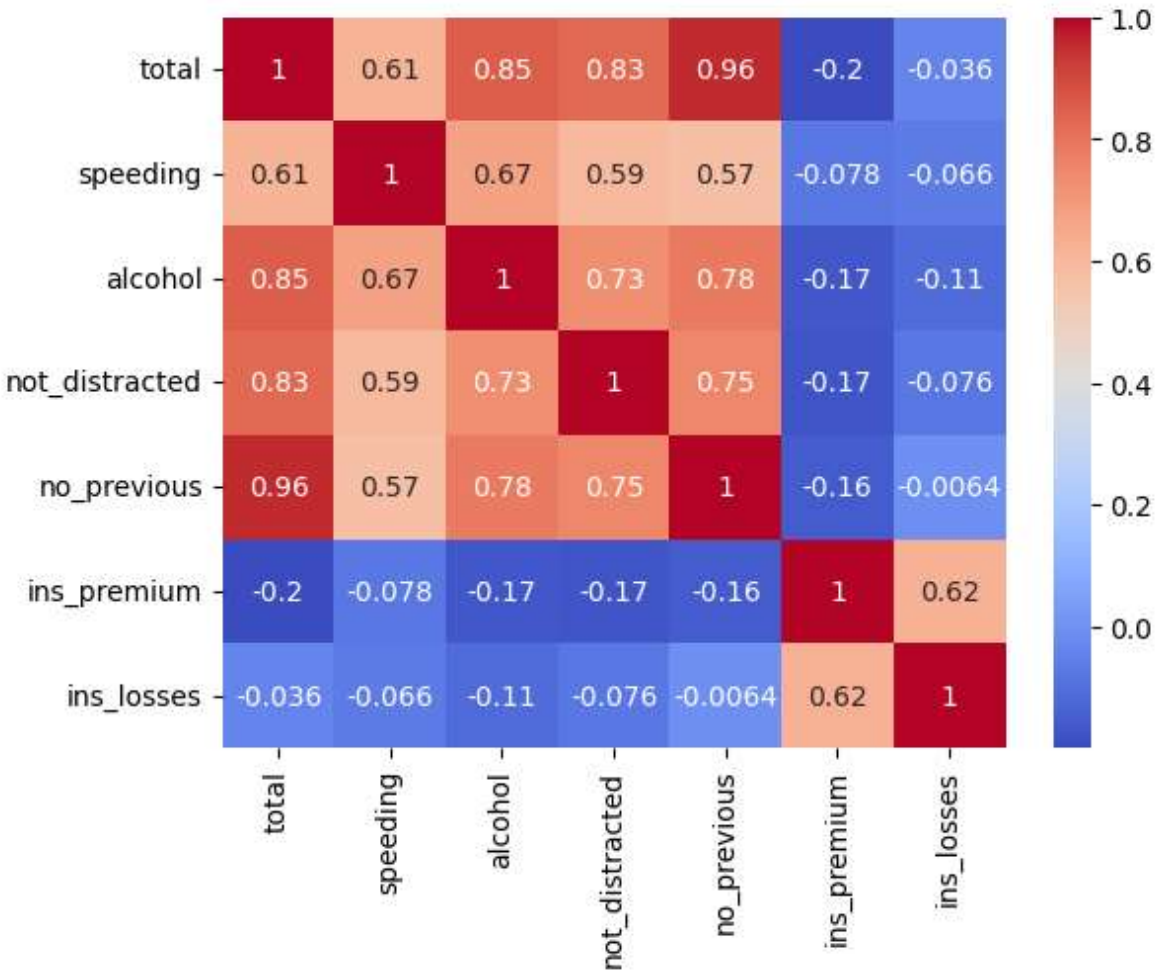


Inference: Shows the total crashes in different states using their abbreviations on the x-axis. You can use this to compare crash counts across states.

In [36]:
```
corr_matrix = df.corr()
sns.heatmap(corr_matrix, annot=True, cmap="coolwarm")
```

```
<ipython-input-36-01a5e2923d95>:1: FutureWarning: The default value of numer
ic_only in DataFrame.corr is deprecated. In a future version, it will defaul
t to False. Select only valid columns or specify the value of numeric_only t
o silence this warning.
  corr_matrix = df.corr()
```

Out[36]: <Axes: >



Inference:"total" and "alcohol" have a relatively strong positive correlation.