In [1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

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

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

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

Out[3]:

	total	speeding	alcohol	not_distracted	no_previous	ins_premium	ins_losses	abb
0	18.8	7.332	5.640	18.048	15.040	784.55	145.08	
1	18.1	7.421	4.525	16.290	17.014	1053.48	133.93	
2	18.6	6.510	5.208	15.624	17.856	899.47	110.35	
3	22.4	4.032	5.824	21.056	21.280	827.34	142.39	
4	12.0	4.200	3.360	10.920	10.680	878.41	165.63	
5	13.6	5.032	3.808	10.744	12.920	835.50	139.91	
6	10.8	4.968	3.888	9.396	8.856	1068.73	167.02	
7	16.2	6.156	4.860	14.094	16.038	1137.87	151.48	
8	5.9	2.006	1.593	5.900	5.900	1273.89	136.05	
9	17.9	3.759	5.191	16.468	16.826	1160.13	144.18	
10	15.6	2.964	3.900	14.820	14.508	913.15	142.80	
11	17.5	9.450	7.175	14.350	15.225	861.18	120.92	
12	15.3	5.508	4.437	13.005	14.994	641.96	82.75	
13	12.8	4.608	4.352	12.032	12.288	803.11	139.15	
14	14.5	3.625	4.205	13.775	13.775	710.46	108.92	
15	15.7	2.669	3.925	15.229	13.659	649.06	114.47	
16	17.8	4.806	4.272	13.706	15.130	780.45	133.80	
17	21.4	4.066	4.922	16.692	16.264	872.51	137.13	
18	20.5	7.175	6.765	14.965	20.090	1281.55	194.78	
19	15.1	5.738	4.530	13.137	12.684	661.88	96.57	
20	12.5	4.250	4.000	8.875	12.375	1048.78	192.70	
21	8.2	1.886	2.870	7.134	6.560	1011.14	135.63	
22	14.1	3.384	3.948	13.395	10.857	1110.61	152.26	
23	9.6	2.208	2.784	8.448	8.448	777.18	133.35	
24	17.6	2.640	5.456	1.760	17.600	896.07	155.77	
25	16.1	6.923	5.474	14.812	13.524	790.32	144.45	
26	21.4	8.346	9.416	17.976	18.190	816.21	85.15	
27	14.9	1.937	5.215	13.857	13.410	732.28	114.82	
28	14.7	5.439	4.704	13.965	14.553	1029.87	138.71	
29	11.6	4.060	3.480	10.092	9.628	746.54	120.21	
30	11.2	1.792	3.136	9.632	8.736	1301.52	159.85	
31	18.4	3.496	4.968	12.328	18.032	869.85	120.75	
32	12.3	3.936	3.567	10.824	9.840	1234.31	150.01	

	total	speeding	alcohol	not_distracted	no_previous	ins_premium	ins_losses	abb
33	16.8	6.552	5.208	15.792	13.608	708.24	127.82	
34	23.9	5.497	10.038	23.661	20.554	688.75	109.72	
35	14.1	3.948	4.794	13.959	11.562	697.73	133.52	
36	19.9	6.368	5.771	18.308	18.706	881.51	178.86	
37	12.8	4.224	3.328	8.576	11.520	804.71	104.61	
38	18.2	9.100	5.642	17.472	16.016	905.99	153.86	
39	11.1	3.774	4.218	10.212	8.769	1148.99	148.58	
40	23.9	9.082	9.799	22.944	19.359	858.97	116.29	
41	19.4	6.014	6.402	19.012	16.684	669.31	96.87	
42	19.5	4.095	5.655	15.990	15.795	767.91	155.57	
43	19.4	7.760	7.372	17.654	16.878	1004.75	156.83	
44	11.3	4.859	1.808	9.944	10.848	809.38	109.48	
45	13.6	4.080	4.080	13.056	12.920	716.20	109.61	
46	12.7	2.413	3.429	11.049	11.176	768.95	153.72	
47	10.6	4.452	3.498	8.692	9.116	890.03	111.62	
48	23.8	8.092	6.664	23.086	20.706	992.61	152.56	
49	13.8	4.968	4.554	5.382	11.592	670.31	106.62	
50	17.4	7.308	5.568	14.094	15.660	791.14	122.04	

In [4]:
```
df.head(5)
```

Out[4]:

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

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

Out[5]:

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

Univariate: Histogram, Bar Chart, Pir chart, Box Plot, Count Plot,

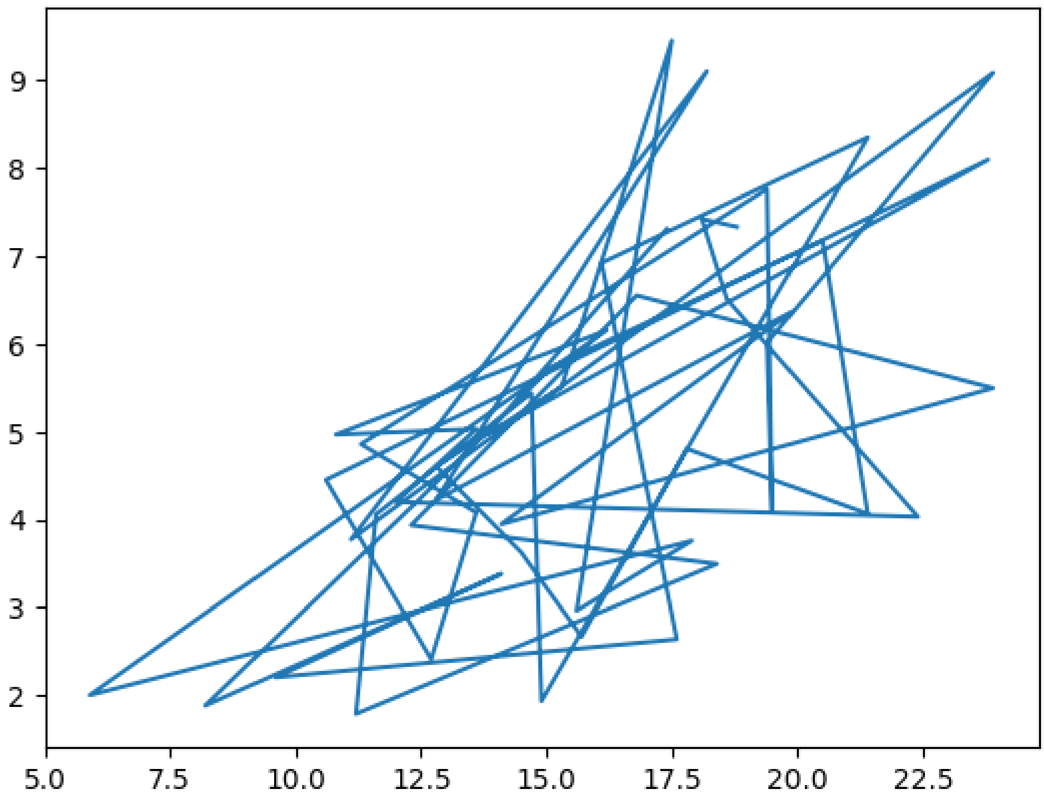
Bivariate: Scatter Plot, Line Chart, Anova, Chi Squared Error Test,

Multivariate: Heatmap, Clusturing Analysis, Principal Component Analysis,

In [6]:

```
plt.plot(df["total"],df["speeding"])
#Inference: This basic Graph plots the points using Total and Speeding Cases  Da
```

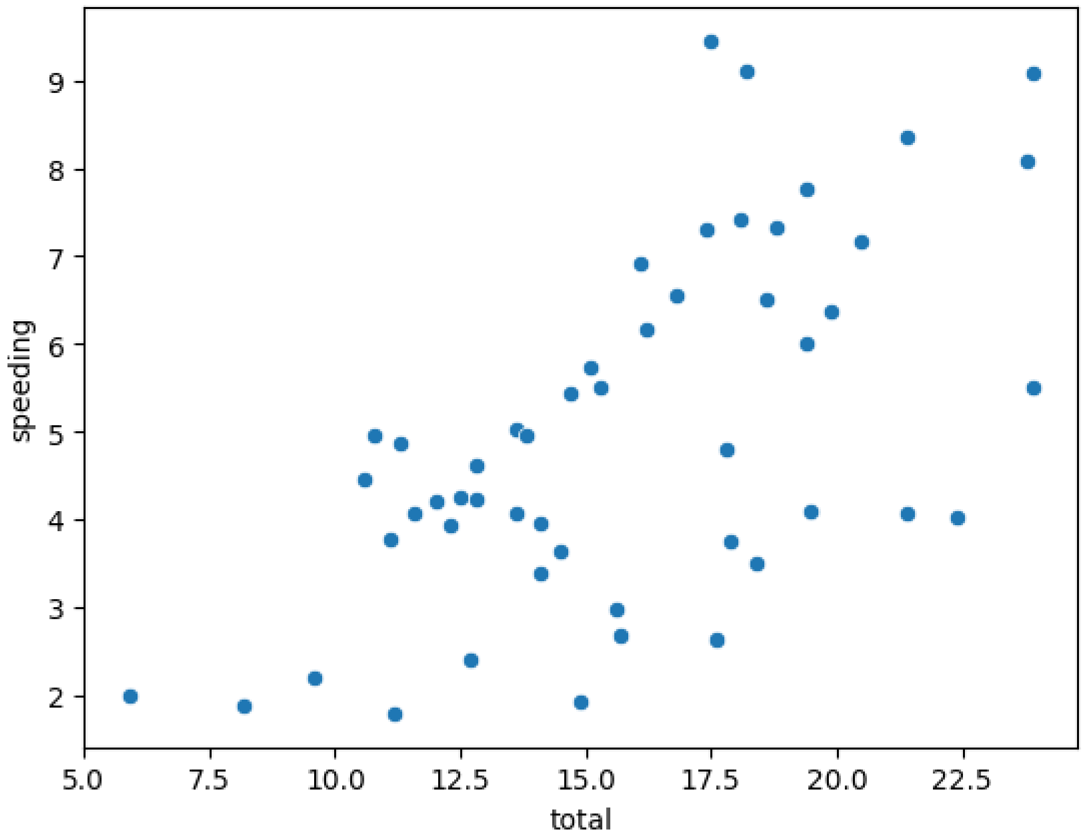
Out[6]: [<matplotlib.lines.Line2D at 0x1f0b4dff910>]



In [7]:

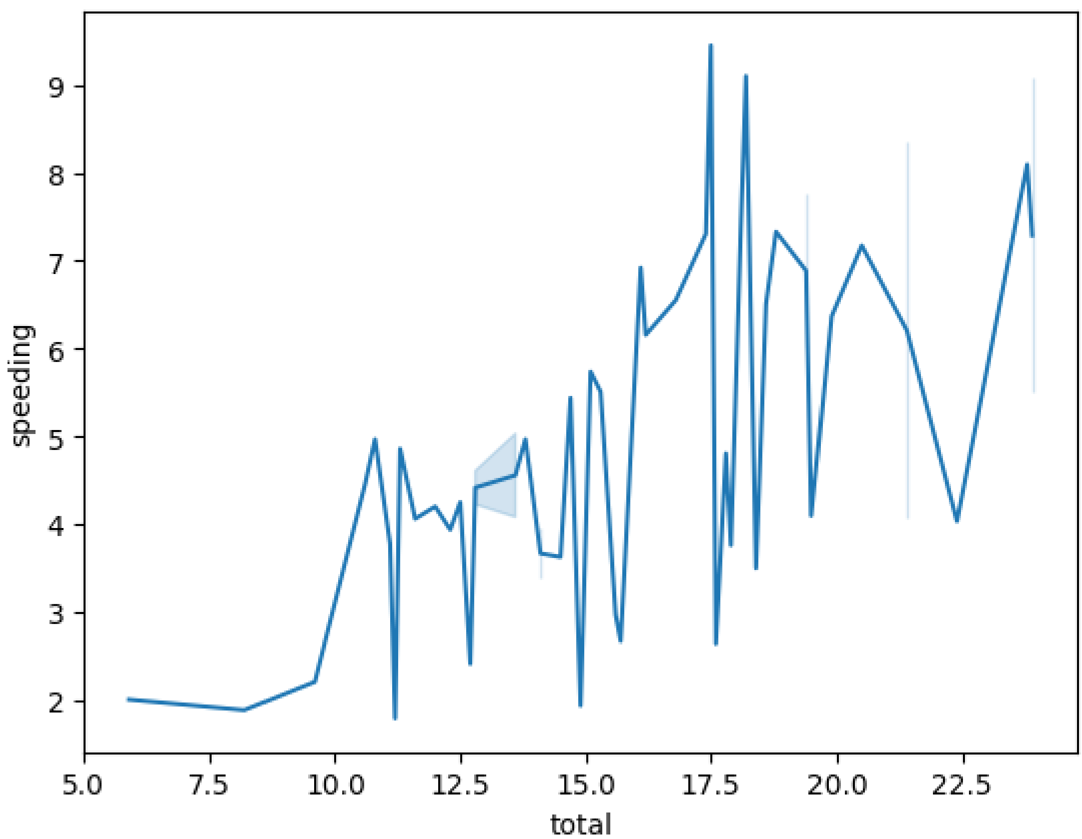
```
sns.scatterplot(x="total", y="speeding", data=df)
#Inference : The below plot shows that the number of speeding cases is directly
```

Out[7]: <Axes: xlabel='total', ylabel='speeding'>



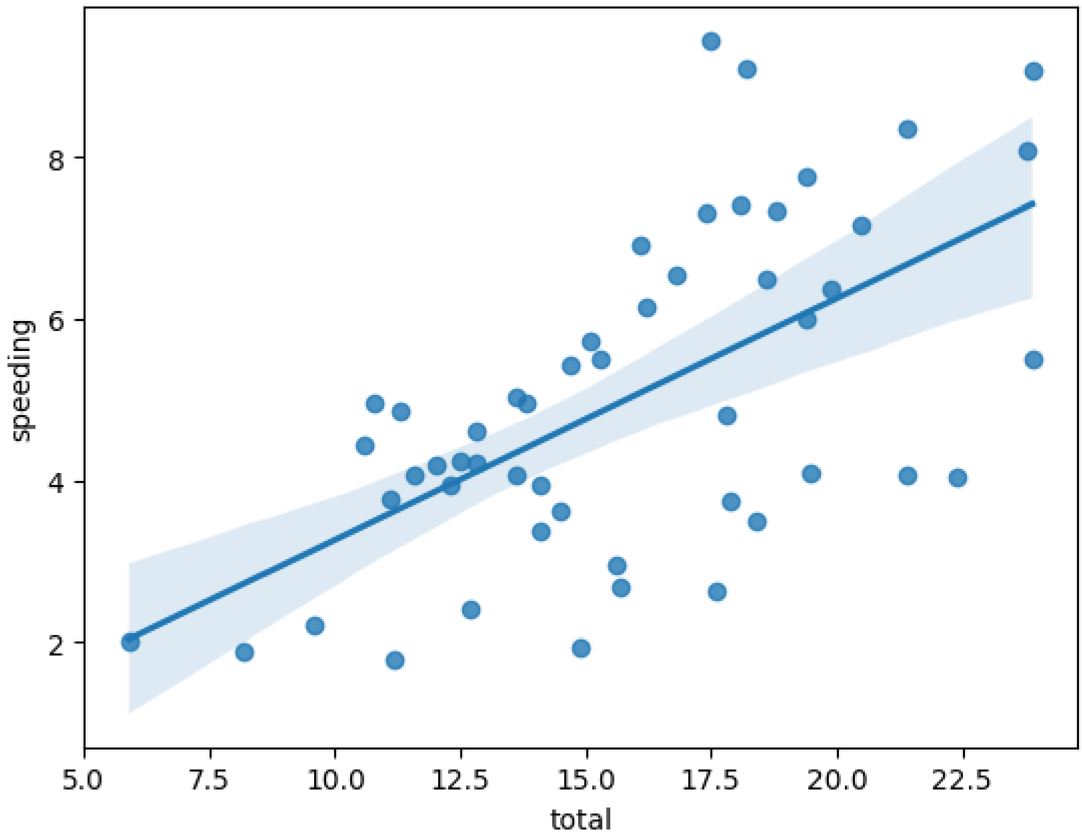
In [8]:
```
sns.lineplot(x="total", y="speeding", data=df)
#Inference: The below plot shows that the number of speeding cases is directly p
```

Out[8]: <Axes: xlabel='total', ylabel='speeding'>



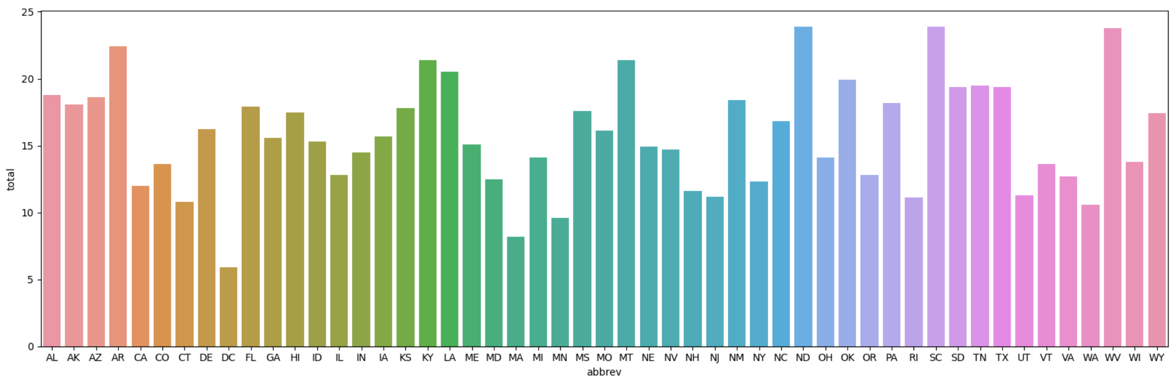
In [9]:
```
sns.regplot(x="total", y="speeding", data=df)
#Inference: The below plot shows that the number of speeding cases is directly p
```

Out[9]: <Axes: xlabel='total', ylabel='speeding'>



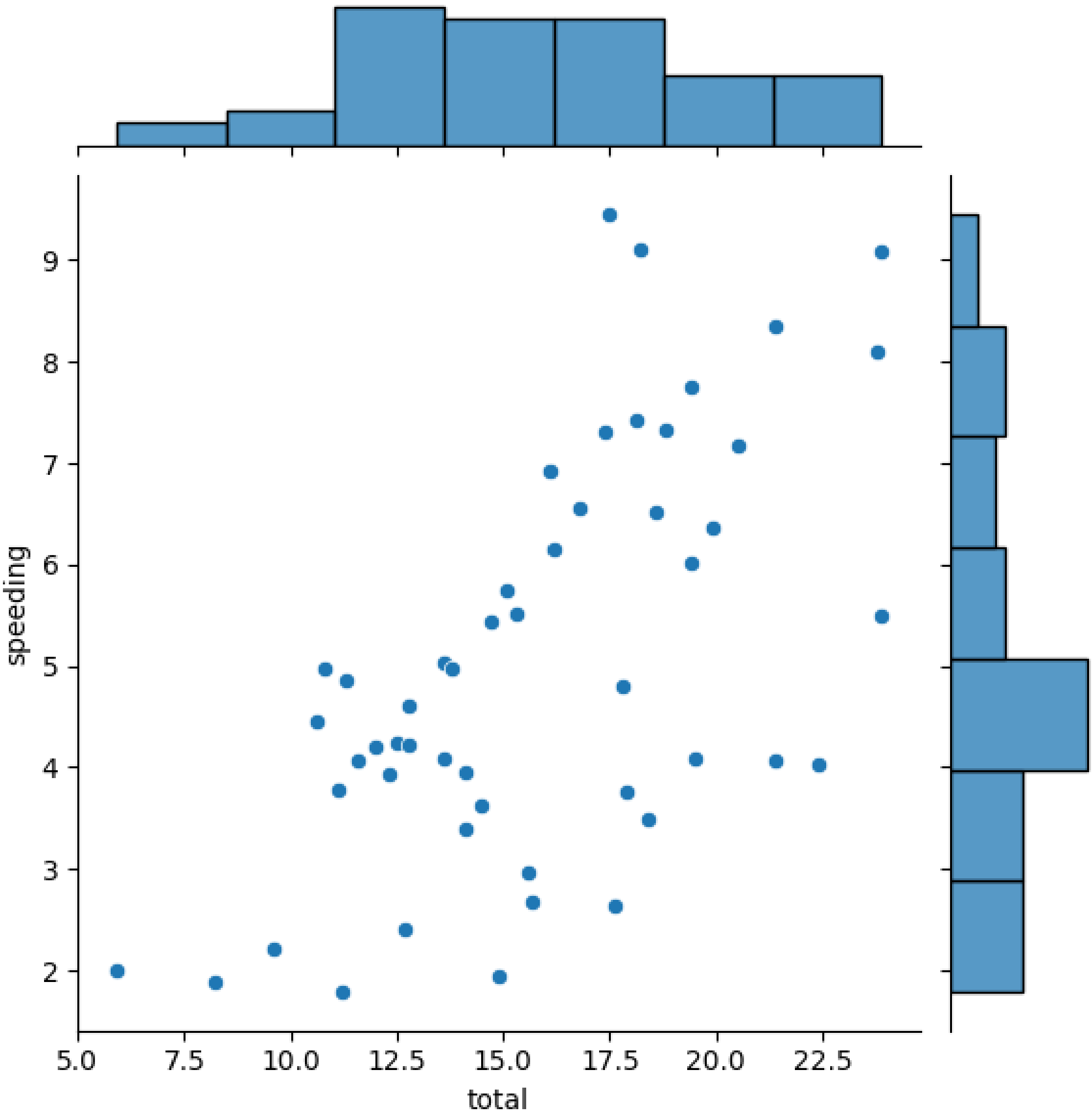
In [10]:
```
fig=plt.figure(figsize=(20,6))
sns.barplot(x="abbrev", y="total", data=df)
#Inference: This barplot plots the bar graph between abbrev and total cases whic
```

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



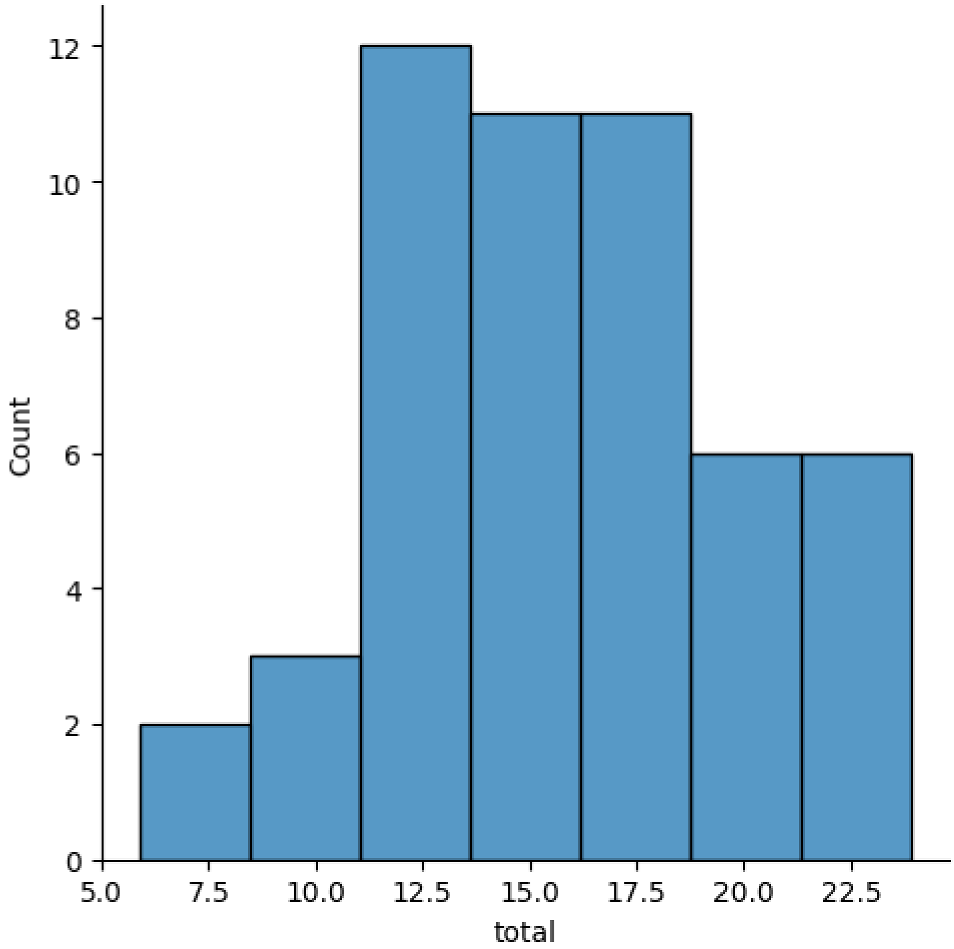
In [11]:
```
sns.jointplot(x="total", y="speeding", data=df)
#Inference : The above plot shows that the number of speeding cases is directly
```

Out[11]: <seaborn.axisgrid.JointGrid at 0x1f0b7316050>



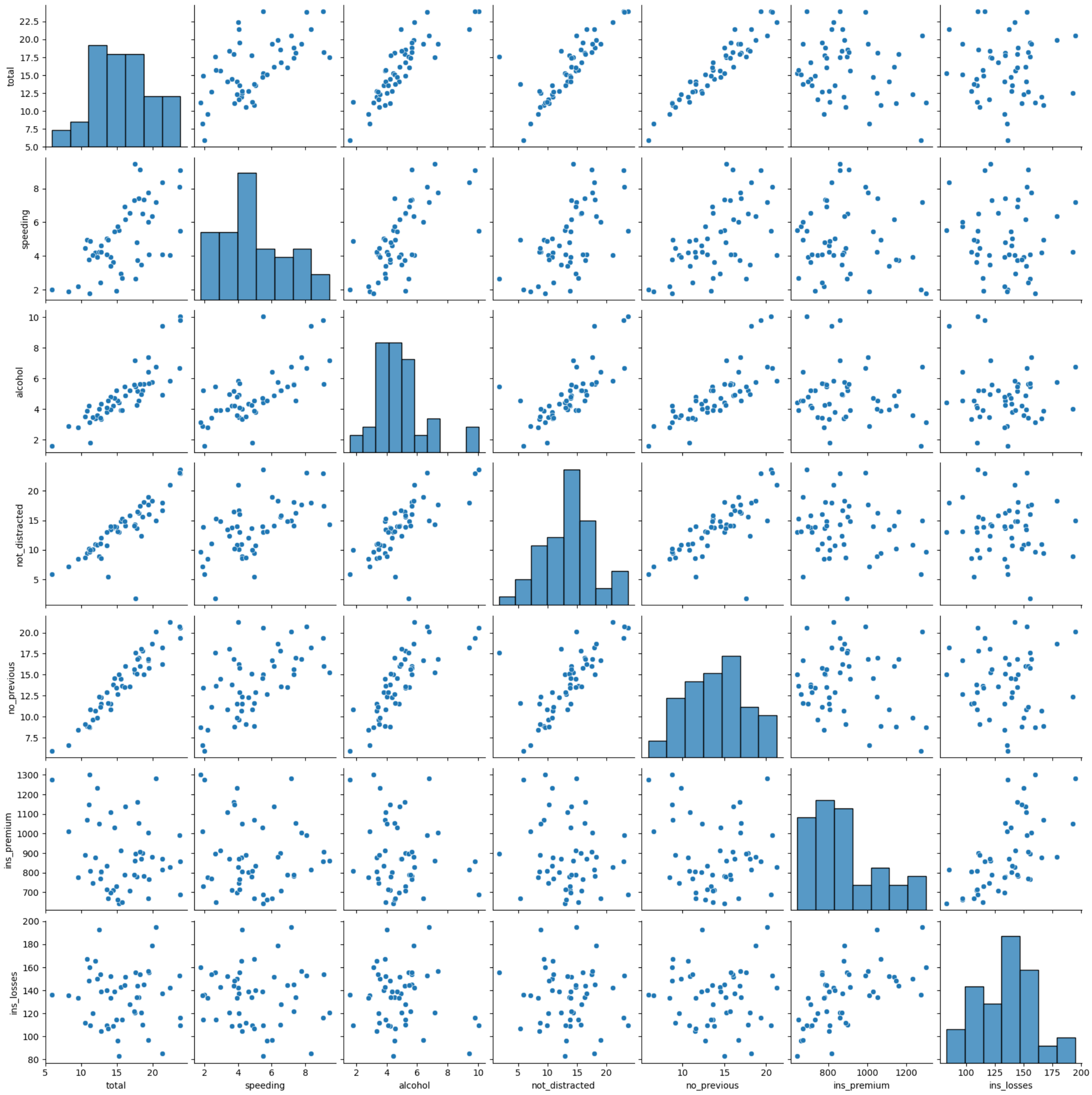
In [12]:
```
sns.displot(x="total", data=df)
#Inferecne: This Displot plots the histogram which is a univariate analysis of f
```

Out[12]: `<seaborn.axisgrid.FacetGrid at 0x1f0b76aaf50>`



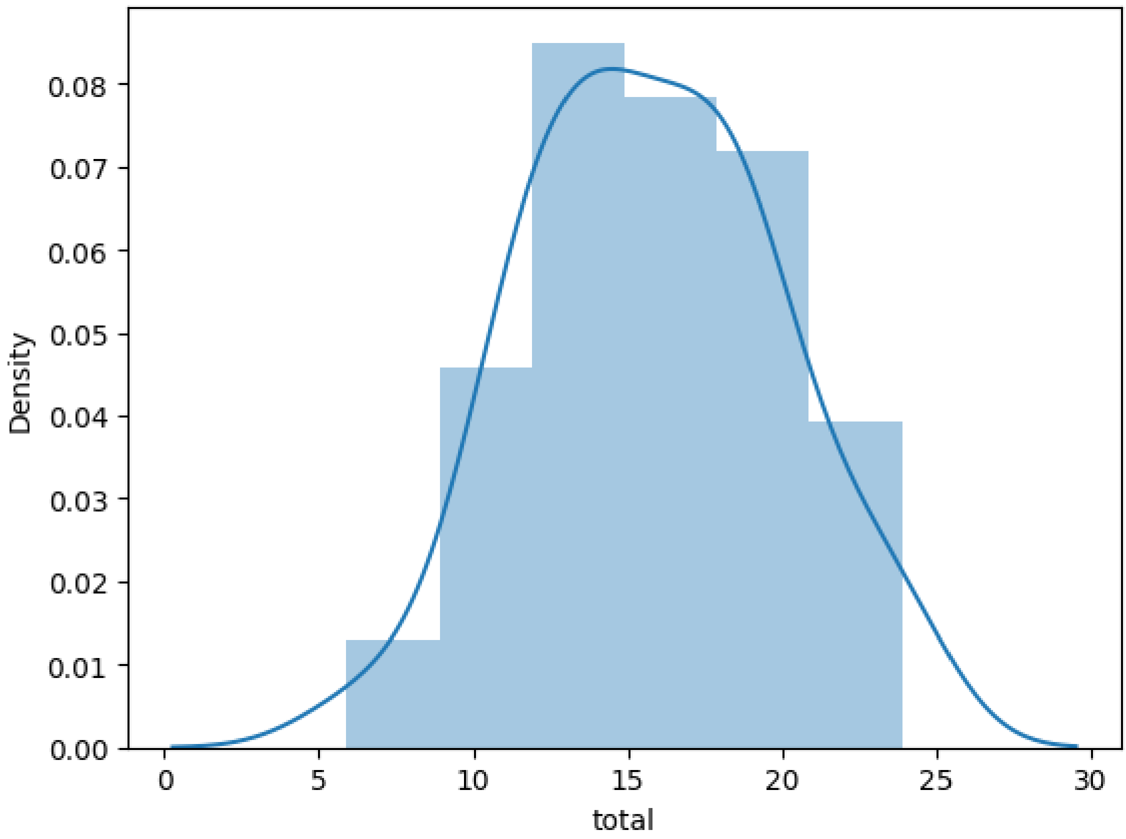
In [13]:
```
sns.pairplot(df)
#Inferecne: This pairplot Plots the Scatter plots between all the features od th
```

Out[13]: <seaborn.axisgrid.PairGrid at 0x1f0b7716f10>



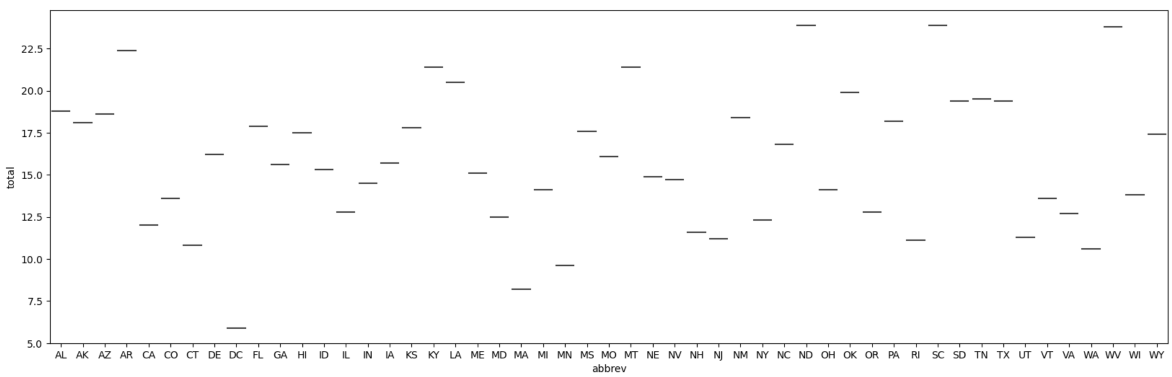
In [14]:
```
sns.distplot(df["total"])
#Inference : Dist plot plots the variation in data feature - total and plots a n
```

Out[14]: `<Axes: xlabel='total', ylabel='Density'>`



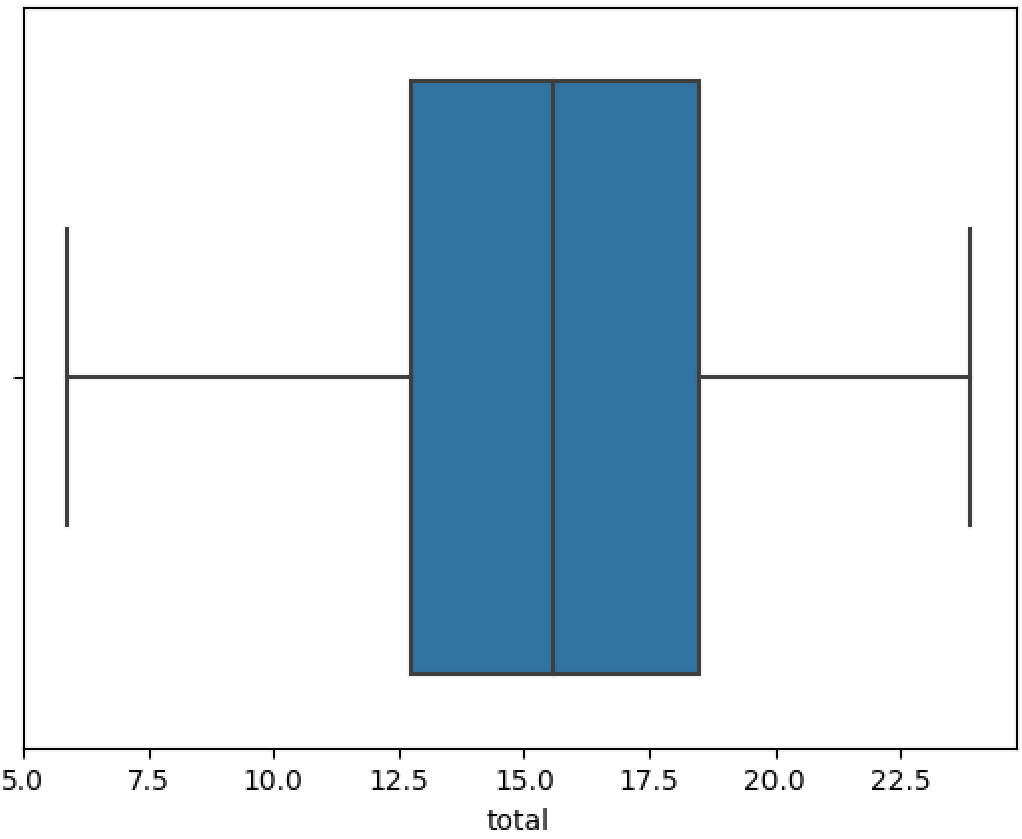
In [15]:
```
fig=plt.figure(figsize=(20,6))
sns.violinplot(x="abbrev", y="total", data=df)
#Inference: Violinplot shows the distribution of quantitative abbrev across seve
```

Out[15]: <Axes: xlabel='abbrev', ylabel='total'>



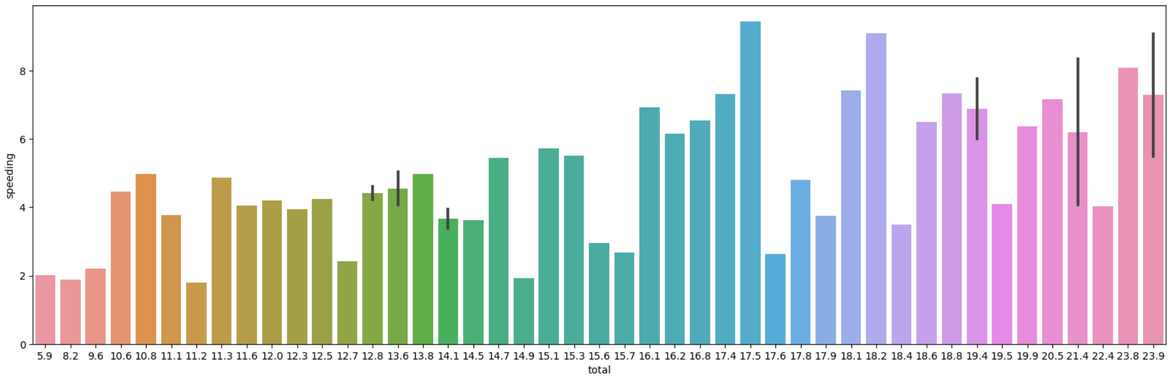
In [16]:
```
sns.boxplot(x="total", data=df)
#Inference: This Boxplot helps us to find if there are any outliers in the data
```

Out[16]: <Axes: xlabel='total'>



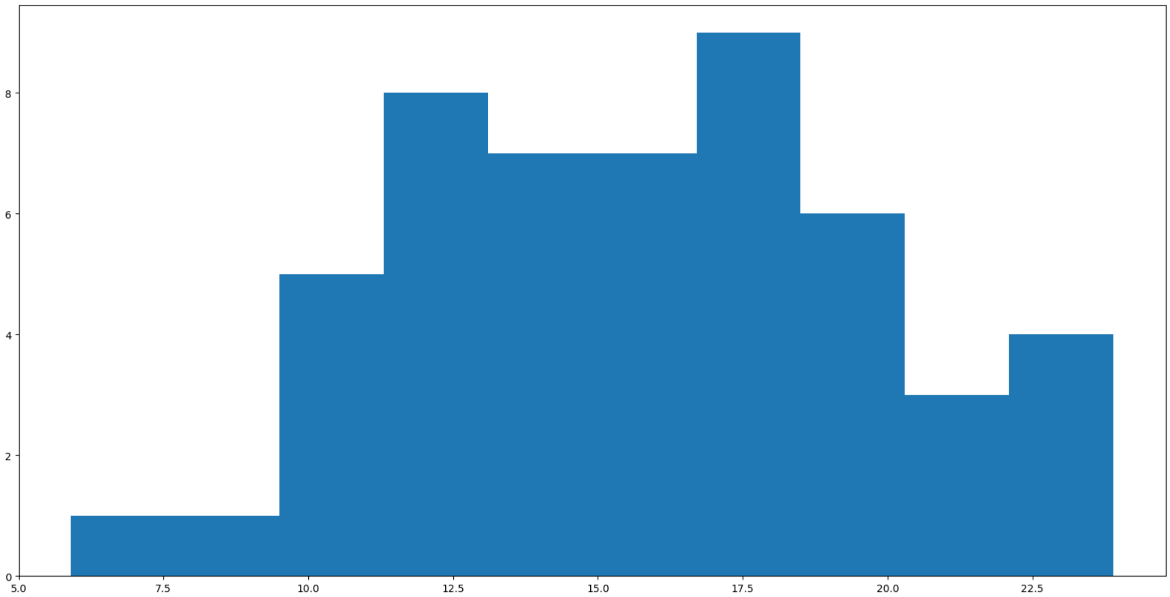
In [17]:
```
fig=plt.figure(figsize=(20,6))
sns.barplot(x=df.total, y=df.speeding)
#Inference: This Bar plots shows us the Which type of abbrev is having  speeding
```

Out[17]:
```
<Axes: xlabel='total', ylabel='speeding'>
```

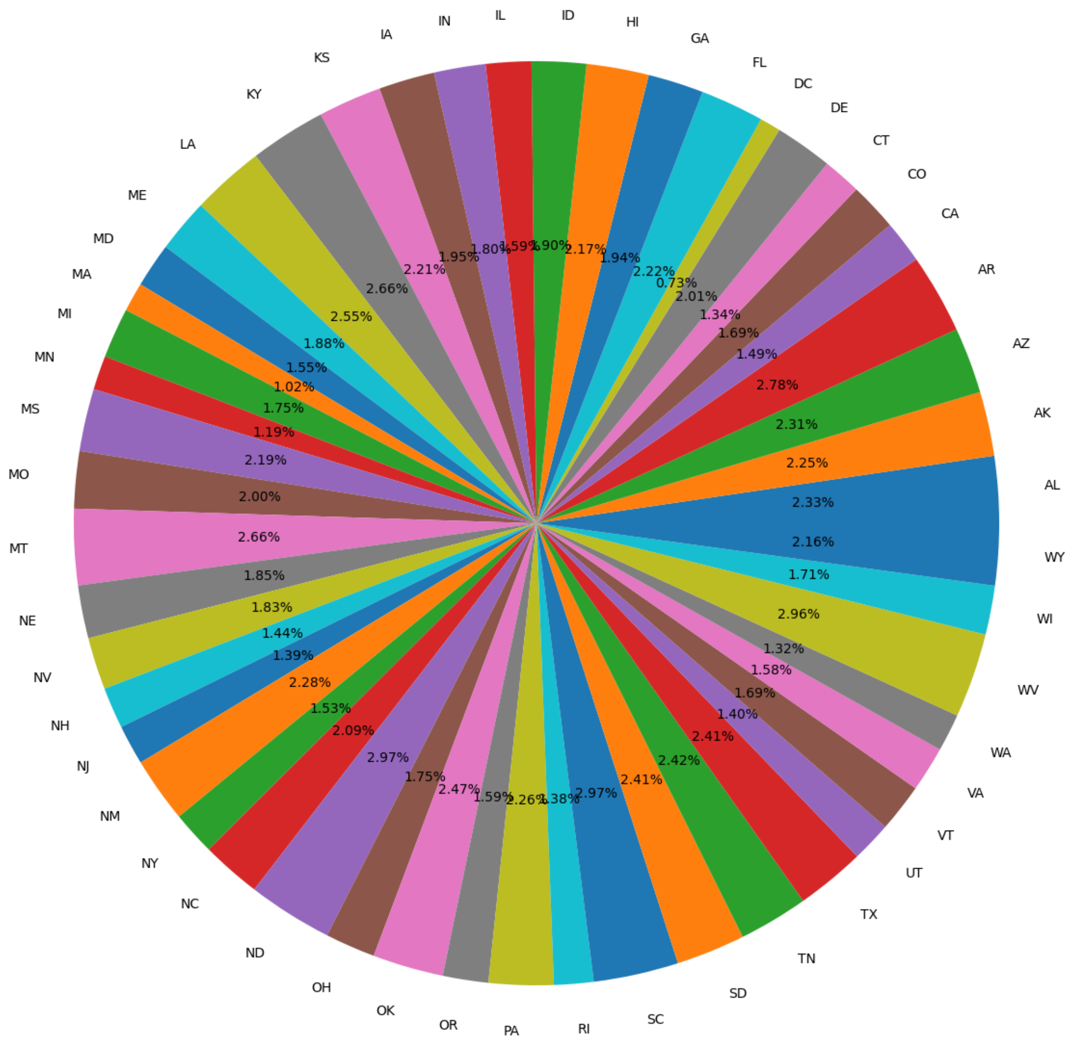


In [18]:
```
fig=plt.figure(figsize=(20,10))
plt.hist(df.total)
#Inference: It Groups the total_accidents into groups  and plots the density of
```

Out[18]:
```
(array([1., 1., 5., 8., 7., 7., 9., 6., 3., 4.]),
 array([ 5.9,  7.7,  9.5, 11.3, 13.1, 14.9, 16.7, 18.5, 20.3, 22.1, 23.9]),
 <BarContainer object of 10 artists>)
```



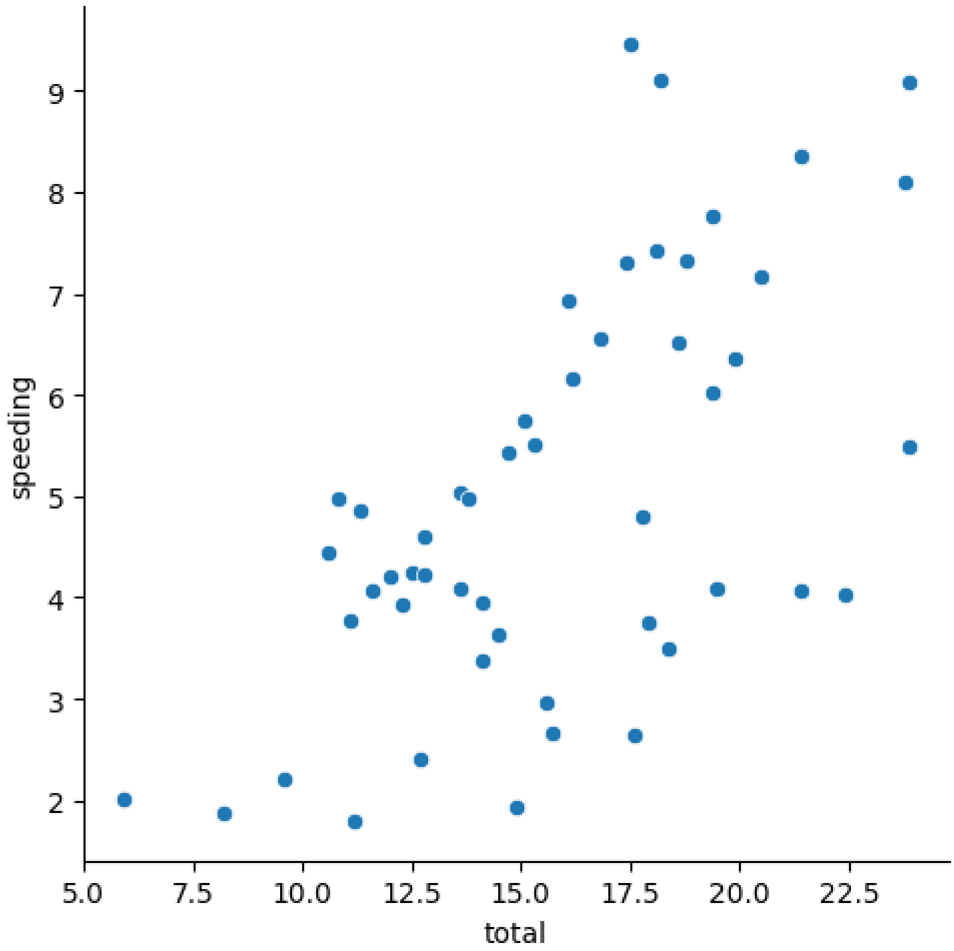
In [19]:
```
fig=plt.figure(figsize=(20,15))
axes1=fig.add_axes([0.1,0.1,0.8,0.8])
plt.pie(df.total,labels=df.abbrev,autopct="%0.2f%%")
plt.show()
#Inference: It Shows the data in the form of pie while showing how much part of
```



In [20]:
```python
fig=plt.figure(figsize=(30,6))
sns.relplot(x="total", y="speeding", data=df)
#Inference: The above plot shows the relation between total and speeding feature
```

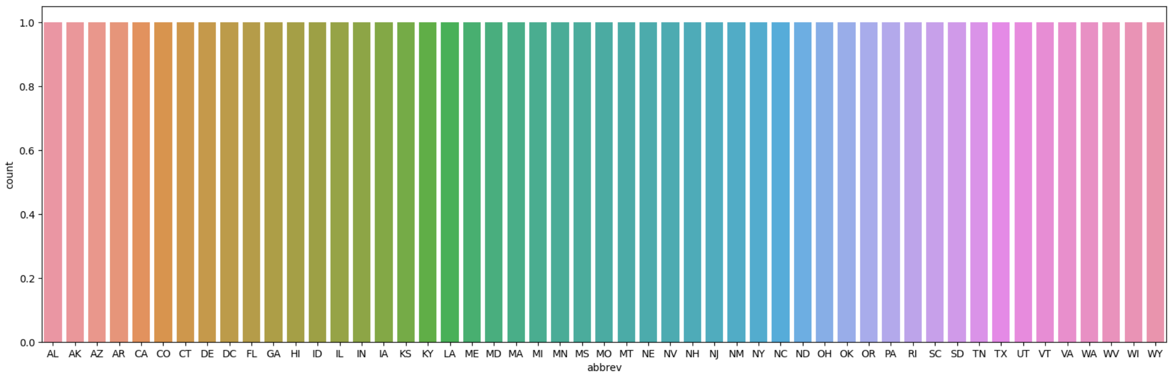
Out[20]:
```
<seaborn.axisgrid.FacetGrid at 0x1f0bdef6210>
```

```
<Figure size 3000x600 with 0 Axes>
```



In [21]:
```python
fig=plt.figure(figsize=(20,6))
sns.countplot(x="abbrev", data=df)
#Inference: This plots the count of each type of abbrev as here each type is max
```

Out[21]:
```
<Axes: xlabel='abbrev', ylabel='count'>
```



In [22]:
```python
corr=df.corr(numeric_only=True)
corr
```

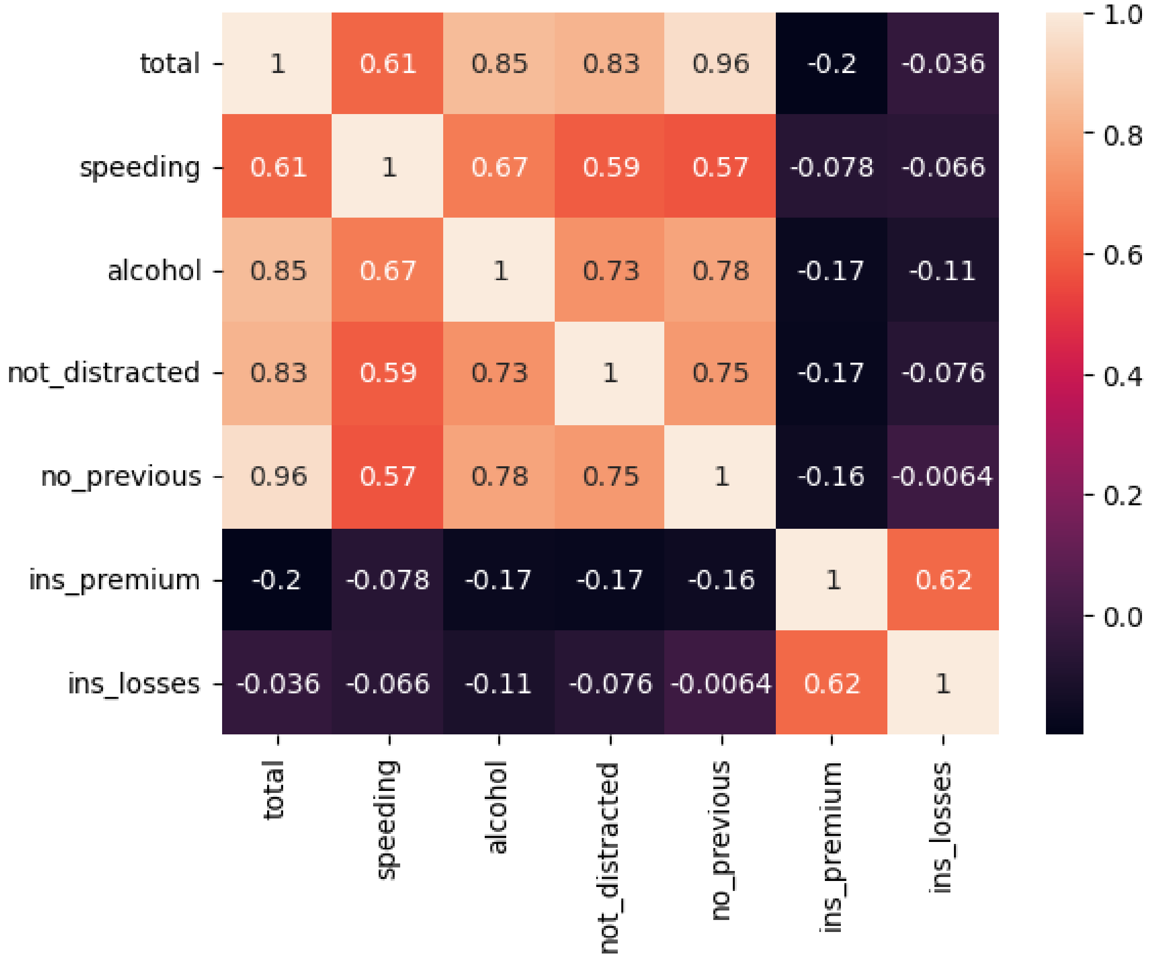
Out[22]:

	total	speeding	alcohol	not_distracted	no_previous	ins_premiur
total	1.000000	0.611548	0.852613	0.827560	0.956179	-0.19970
speeding	0.611548	1.000000	0.669719	0.588010	0.571976	-0.07767
alcohol	0.852613	0.669719	1.000000	0.732816	0.783520	-0.17061
not_distracted	0.827560	0.588010	0.732816	1.000000	0.747307	-0.17485
no_previous	0.956179	0.571976	0.783520	0.747307	1.000000	-0.15689
ins_premium	-0.199702	-0.077675	-0.170612	-0.174856	-0.156895	1.00000
ins_losses	-0.036011	-0.065928	-0.112547	-0.075970	-0.006359	0.62311

In [23]:

```
sns.heatmap(corr,annot=True)
#Inference: It shows us the  correlation between the features of the data using
```

Out[23]: <Axes: >



In [ ]: