Importing Libraries

In [1]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings("ignore")
```

Reading the DataSet

In [2]:
```
df=pd.read_csv("C:/Users/maazs/Marriage and divorce in Iran.csv")
```

I have picked this data from Kaggle to analyse the marriages and divorces in Iran from year 1358 to 1399.

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

Out[3]:

	Year	marriage(country)	divorce(country)	marriage(city)	divorce(city)	marriage(village)	divorce(
0	1358	302667	21170	173963	15445	128704	
1	1359	337119	23987	196029	19293	141090	
2	1360	294499	24423	183546	20449	110953	
3	1361	353944	31221	215077	25773	138867	
4	1362	410799	35867	259433	30808	151366	

Exploring the DataSet

In [4]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 42 entries, 0 to 41
Data columns (total 7 columns):
 #   Column             Non-Null Count  Dtype 
---  ------             --------------  ----- 
 0   Year               42 non-null     object
 1   marriage(country)  42 non-null     int64 
 2   divorce(country)   42 non-null     int64 
 3   marriage(city)     42 non-null     object
 4   divorce(city)      42 non-null     object
 5   marriage(village)  42 non-null     object
 6   divorce(village)   42 non-null     object
dtypes: int64(2), object(5)
memory usage: 2.4+ KB
```

We dont find a null value in this data set YET.

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

Out[5]:
```
(42, 7)
```

From above we can notice we have 42 rows and 7 columns in this dataset.

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

Out[6]:

	marriage(country)	divorce(country)
count	42.000000	42.000000
mean	570815.952381	80200.785714
std	181609.291571	56408.667813
min	294499.000000	21170.000000
25%	429055.500000	34848.000000
50%	541869.500000	52420.500000
75%	699875.000000	134336.750000
max	891627.000000	183193.000000

In [7]:
```
df
```

Out[7]:

	Year	marriage(country)	divorce(country)	marriage(city)	divorce(city)	marriage(village)	divo
0	1358	302667	21170	173963	15445	128704	
1	1359	337119	23987	196029	19293	141090	
2	1360	294499	24423	183546	20449	110953	
3	1361	353944	31221	215077	25773	138867	
4	1362	410799	35867	259433	30808	151366	
5	1363	384876	35178	247253	29047	137623	
6	1364	408282	38983	263883	31869	144399	
7	1365	340342	35211	225011	29379	115331	
8	1366	346652	33433	225566	27588	121086	
9	1367	361945	33114	239095	26893	122850	
10	1368	458708	33943	295982	27626	162726	
11	1369	454963	37827	309438	30656	145525	
12	1370	448851	39336	311020	33210	137831	
13	1371	422457	33983	299725	28289	122732	
14	1372	463487	29312	332629	25469	130858	
15	1373	453671	32706	329687	28385	123984	
16	1374	462855	34738	340807	30277	122048	
17	1375	479263	37817	357138	32697	122125	
18	1376	511401	41816	387276	36459	124125	
19	1377	531490	42391	412565	37626	118925	
20	1378	611073	51044	476284	45274	134789	
21	1379	646498	53797	499143	47936	147355	
22	1380	640710	60559	495629	54603	145081	
23	1381	650960	67256	513772	61074	137188	
24	1382	681034	72359	522160	64213	158874	
25	1383	602347	63125	443466	54563	158881	
26	1384	787818	84241	558582	70023	229236	
27	1385	778023	94040	556658	78801	221365	
28	1386	841107	99852	...	...	...	
29	1387	881592	110510	...	...	...	
30	1388	890208	125747	...	...	...	
31	1389	891627	137200	...	...	...	
32	1390	874792	142841	...	...	...	
33	1391	829968	150324	678870	136913	151098	

	Year	marriage(country)	divorce(country)	marriage(city)	divorce(city)	marriage(village)	divo
34	1392	774513	155369	664133	144083	110380	
35	1393	724324	163569	607389	149644	116935	
36	1394	685352	163765	562671	146714	122681	
37	1395	704716	181049	616042	168774	88674	
38	1396(2)	608867	179709	542917	168359	65950	
39	1397(2)	550565	175614	486661	164124	63904	
40	1398	533174	176814	470687	164657	62487	
41	1399	556731	183193	490438	170688	66293	

After checking the complete dataset it's evident the reason .describe() function did not calculate the last four columns is because row 28 to row 32 contained "..." which is not a numeric value and technically we can consider it as a null value.

Feature Engineering

After analysing the first few rows it looks apparent that the city contributes more numbers to marriage than the village. This also means more marriage results into more divorces so what we can do is split the available numbers of marriage(country) and divorce(country) in the ratio of 70:30 and fill the marriage(city) and divorce(city) columns with 70% of marriage(country) and divorce(country) and fill the marriage(village) and divorce(village) columns with 30% of marriage(country) and divorce(country).

FOR ROW NO. 28

In [8]:
```
a=(841107*70)/100
b=(841107*30)/100
c=(99852*70)/100
d=(99852*30)/100
print("Marriages for city is ",a)
print("Marriages for village is ",b)
print("Divorces for city is ",c)
print("Divorces for village is ",d)
```

```
Marriages for city is  588774.9
Marriages for village is  252332.1
Divorces for city is  69896.4
Divorces for village is  29955.6
```

FOR ROW NO. 29

In [9]:
```
a=(881592*70)/100
b=(881592*30)/100
c=(110510*70)/100
d=(110510*30)/100
print("Marriages for city is ",a)
print("Marriages for village is ",b)
print("Divorces for city is ",c)
print("Divorces for village is ",d)
```

```
Marriages for city is  617114.4
Marriages for village is  264477.6
Divorces for city is  77357.0
Divorces for village is  33153.0
```

FOR ROW NO. 30

In [10]:
```
a=(890208*70)/100
b=(890208*30)/100
c=(125747*70)/100
d=(125747*30)/100
print("Marriages for city is ",a)
print("Marriages for village is ",b)
print("Divorces for city is ",c)
print("Divorces for village is ",d)
```

```
Marriages for city is  623145.6
Marriages for village is  267062.4
Divorces for city is  88022.9
Divorces for village is  37724.1
```

FOR ROW NO. 31

In [11]:
```
a=(891627*70)/100
b=(891627*30)/100
c=(137200*70)/100
d=(137200*30)/100
print("Marriages for city is ",a)
print("Marriages for village is ",b)
print("Divorces for city is ",c)
print("Divorces for village is ",d)
```

```
Marriages for city is  624138.9
Marriages for village is  267488.1
Divorces for city is  96040.0
Divorces for village is  41160.0
```

FOR ROW NO. 32

In [12]:
```
a=(874792*70)/100
b=(874792*30)/100
c=(142841*70)/100
d=(142841*30)/100
print("Marriages for city is ",a)
print("Marriages for village is ",b)
print("Divorces for city is ",c)
print("Divorces for village is ",d)
```

```
Marriages for city is  612354.4
Marriages for village is  262437.6
Divorces for city is  99988.7
Divorces for village is  42852.3
```

Now I will take the mean of marriages in city,marriages in village,divorces in city and divorces in village of these 5 rows we have calculated above and fill up their respective columns with the mean value.

In [13]:
```
a=(588774.9+ 617114.4+623145.6+ 624138.9+612354.4)/5
print("Mean marriages for city is ",a)
```

```
Mean marriages for city is  613105.6399999999
```

In [14]:
```
a=(252332.1+262437.6+267488.1+267062.4+ 264477.6)/5
print("Mean marriages for village is ",a)
```

```
Mean marriages for village is  262759.55999999994
```

In [15]:
```
a=(99988.7+ 96040.0+88022.9+77357.0+ 69896.4)/5
print("Mean divorces for city is ",a)
```

```
Mean divorces for city is  86261.0
```

In [16]:
```
a=(42852.3+ 41160.0+37724.1+ 33153.0+29955.6)/5
print("Mean divorces for village is ",a)
```

```
Mean divorces for village is  36969.0
```

Now lets fill up the missing columns.

In [17]:
```
df['marriage(city)'].replace(['…'],['613106'],inplace=True)
```

In [18]:
```
df['divorce(city)'].replace(['…'],['86261'],inplace=True)
```

In [19]:
```
df['marriage(village)'].replace(['…'],['262760'],inplace=True)
```

In [20]:
```
df['divorce(village)'].replace(['…'],['36969'],inplace=True)
```

In [21]:
```
df
```

Out[21]:

	Year	marriage(country)	divorce(country)	marriage(city)	divorce(city)	marriage(village)	d
0	1358	302667	21170	173963	15445	128704	
1	1359	337119	23987	196029	19293	141090	
2	1360	294499	24423	183546	20449	110953	
3	1361	353944	31221	215077	25773	138867	
4	1362	410799	35867	259433	30808	151366	
5	1363	384876	35178	247253	29047	137623	
6	1364	408282	38983	263883	31869	144399	
7	1365	340342	35211	225011	29379	115331	
8	1366	346652	33433	225566	27588	121086	
9	1367	361945	33114	239095	26893	122850	
10	1368	458708	33943	295982	27626	162726	
11	1369	454963	37827	309438	30656	145525	
12	1370	448851	39336	311020	33210	137831	
13	1371	422457	33983	299725	28289	122732	
14	1372	463487	29312	332629	25469	130858	
15	1373	453671	32706	329687	28385	123984	
16	1374	462855	34738	340807	30277	122048	
17	1375	479263	37817	357138	32697	122125	
18	1376	511401	41816	387276	36459	124125	
19	1377	531490	42391	412565	37626	118925	
20	1378	611073	51044	476284	45274	134789	
21	1379	646498	53797	499143	47936	147355	
22	1380	640710	60559	495629	54603	145081	
23	1381	650960	67256	513772	61074	137188	
24	1382	681034	72359	522160	64213	158874	
25	1383	602347	63125	443466	54563	158881	
26	1384	787818	84241	558582	70023	229236	
27	1385	778023	94040	556658	78801	221365	
28	1386	841107	99852	613106	86261	262760	
29	1387	881592	110510	613106	86261	262760	
30	1388	890208	125747	613106	86261	262760	
31	1389	891627	137200	613106	86261	262760	
32	1390	874792	142841	613106	86261	262760	
33	1391	829968	150324	678870	136913	151098	

	Year	marriage(country)	divorce(country)	marriage(city)	divorce(city)	marriage(village)	c
34	1392	774513	155369	664133	144083	110380	
35	1393	724324	163569	607389	149644	116935	
36	1394	685352	163765	562671	146714	122681	
37	1395	704716	181049	616042	168774	88674	
38	1396(2)	608867	179709	542917	168359	65950	
39	1397(2)	550565	175614	486661	164124	63904	
40	1398	533174	176814	470687	164657	62487	
41	1399	556731	183193	490438	170688	66293	

In [22]:
```
df['Year'].replace(['1396(2)'],['1396'],inplace=True)
df['Year'].replace(['1397(2)'],['1397'],inplace=True)
```

In [23]:
```
df
```

Out[23]:

	Year	marriage(country)	divorce(country)	marriage(city)	divorce(city)	marriage(village)	div
0	1358	302667	21170	173963	15445	128704	
1	1359	337119	23987	196029	19293	141090	
2	1360	294499	24423	183546	20449	110953	
3	1361	353944	31221	215077	25773	138867	
4	1362	410799	35867	259433	30808	151366	
5	1363	384876	35178	247253	29047	137623	
6	1364	408282	38983	263883	31869	144399	
7	1365	340342	35211	225011	29379	115331	
8	1366	346652	33433	225566	27588	121086	
9	1367	361945	33114	239095	26893	122850	
10	1368	458708	33943	295982	27626	162726	
11	1369	454963	37827	309438	30656	145525	
12	1370	448851	39336	311020	33210	137831	
13	1371	422457	33983	299725	28289	122732	
14	1372	463487	29312	332629	25469	130858	
15	1373	453671	32706	329687	28385	123984	
16	1374	462855	34738	340807	30277	122048	
17	1375	479263	37817	357138	32697	122125	
18	1376	511401	41816	387276	36459	124125	
19	1377	531490	42391	412565	37626	118925	
20	1378	611073	51044	476284	45274	134789	
21	1379	646498	53797	499143	47936	147355	
22	1380	640710	60559	495629	54603	145081	
23	1381	650960	67256	513772	61074	137188	
24	1382	681034	72359	522160	64213	158874	
25	1383	602347	63125	443466	54563	158881	
26	1384	787818	84241	558582	70023	229236	
27	1385	778023	94040	556658	78801	221365	
28	1386	841107	99852	613106	86261	262760	
29	1387	881592	110510	613106	86261	262760	
30	1388	890208	125747	613106	86261	262760	
31	1389	891627	137200	613106	86261	262760	
32	1390	874792	142841	613106	86261	262760	
33	1391	829968	150324	678870	136913	151098	

	Year	marriage(country)	divorce(country)	marriage(city)	divorce(city)	marriage(village)	div
34	1392	774513	155369	664133	144083	110380	
35	1393	724324	163569	607389	149644	116935	
36	1394	685352	163765	562671	146714	122681	
37	1395	704716	181049	616042	168774	88674	
38	1396	608867	179709	542917	168359	65950	
39	1397	550565	175614	486661	164124	63904	
40	1398	533174	176814	470687	164657	62487	
41	1399	556731	183193	490438	170688	66293	

We have succesfully dealt with the missing values.

In [24]:
```
a=df['marriage(city)'].mean()
b=df['marriage(city)'].median()
print("Mean of marriages in city is ",a)
print("Median of marriages in city is ",b)
```

```
Mean of marriages in city is  4.141980857837703e+249
Median of marriages in city is  457076.5
```

From above calculations we observe mean is less than median so it's -ve skewed.

In [25]:
```
a=df['marriage(village)'].mean()
b=df['marriage(village)'].median()
print("Mean of marriages in village is ",a)
print("Median of marriages in village is ",b)
```

```
Mean of marriages in village is  3.064384311669308e+244
Median of marriages in village is  135988.5
```

From above calculations we observe mean is more than median so it's +ve skewed.

In [26]:
```
a=df['divorce(city)'].mean()
b=df['divorce(city)'].median()
print("Mean of divorces in city is ",a)
print("Median of divorces in city is ",b)
```

```
Mean of divorces in city is  3.677426888582125e+216
Median of divorces in city is  46605.0
```

From above calculations we observe mean is less than median so it's -ve skewed.

In [27]:
```
a=df['divorce(village)'].mean()
b=df['divorce(village)'].median()
print("Mean of divorces in city is ",a)
print("Median of divorces in city is ",b)
```

```
Mean of divorces in city is  1.3632070094632018e+182
Median of divorces in city is  6269.0
```

From above calculations we observe mean is more than median so it's +ve skewed.

Visualisation

Lets cover UNIVARIATE ANALYSIS first.

In [28]:
```
df=pd.read_csv('C:/Users/maazs/Marriage and divorce in Iran neww.csv')
```

In [29]:
```
df
```

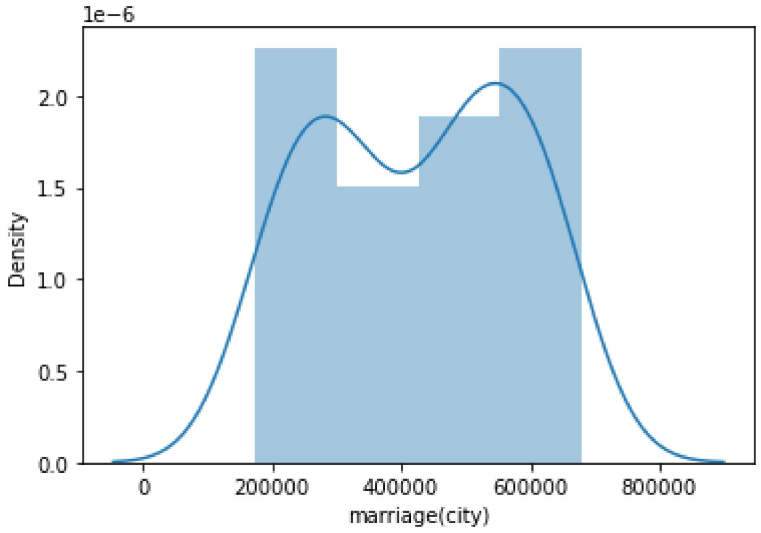
Out[29]:

	Year	marriage(country)	divorce(country)	marriage(city)	divorce(city)	marriage(village)	div
0	1358	302667	21170	173963	15445	128704	
1	1359	337119	23987	196029	19293	141090	
2	1360	294499	24423	183546	20449	110953	
3	1361	353944	31221	215077	25773	138867	
4	1362	410799	35867	259433	30808	151366	
5	1363	384876	35178	247253	29047	137623	
6	1364	408282	38983	263883	31869	144399	
7	1365	340342	35211	225011	29379	115331	
8	1366	346652	33433	225566	27588	121086	
9	1367	361945	33114	239095	26893	122850	
10	1368	458708	33943	295982	27626	162726	
11	1369	454963	37827	309438	30656	145525	
12	1370	448851	39336	311020	33210	137831	
13	1371	422457	33983	299725	28289	122732	
14	1372	463487	29312	332629	25469	130858	
15	1373	453671	32706	329687	28385	123984	
16	1374	462855	34738	340807	30277	122048	
17	1375	479263	37817	357138	32697	122125	
18	1376	511401	41816	387276	36459	124125	
19	1377	531490	42391	412565	37626	118925	
20	1378	611073	51044	476284	45274	134789	
21	1379	646498	53797	499143	47936	147355	
22	1380	640710	60559	495629	54603	145081	
23	1381	650960	67256	513772	61074	137188	
24	1382	681034	72359	522160	64213	158874	
25	1383	602347	63125	443466	54563	158881	
26	1384	787818	84241	558582	70023	229236	
27	1385	778023	94040	556658	78801	221365	
28	1386	841107	99852	613106	86261	262760	
29	1387	881592	110510	613106	86261	262760	
30	1388	890208	125747	613106	86261	262760	
31	1389	891627	137200	613106	86261	262760	
32	1390	874792	142841	613106	86261	262760	
33	1391	829968	150324	678870	136913	151098	

	Year	marriage(country)	divorce(country)	marriage(city)	divorce(city)	marriage(village)	div
34	1392	774513	155369	664133	144083	110380	
35	1393	724324	163569	607389	149644	116935	
36	1394	685352	163765	562671	146714	122681	
37	1395	704716	181049	616042	168774	88674	
38	1396	608867	179709	542917	168359	65950	
39	1397	550565	175614	486661	164124	63904	
40	1398	533174	176814	470687	164657	62487	
41	1399	556731	183193	490438	170688	66293	

DISTPLOTS

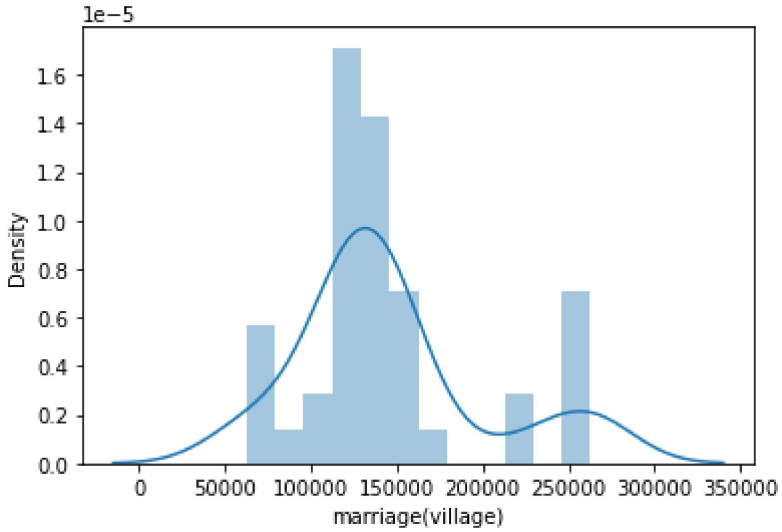
In [30]:
```
sns.distplot(df['marriage(city)'])
plt.show()
```



-In above distplot we can conclude that the range of marriages in city is between around 1,50,000 to 7,00,000.

-Probable density shoots upto 2,70,000 and then dips a little to 4,10,000 then rises again upto 56,000 then goes downhill.

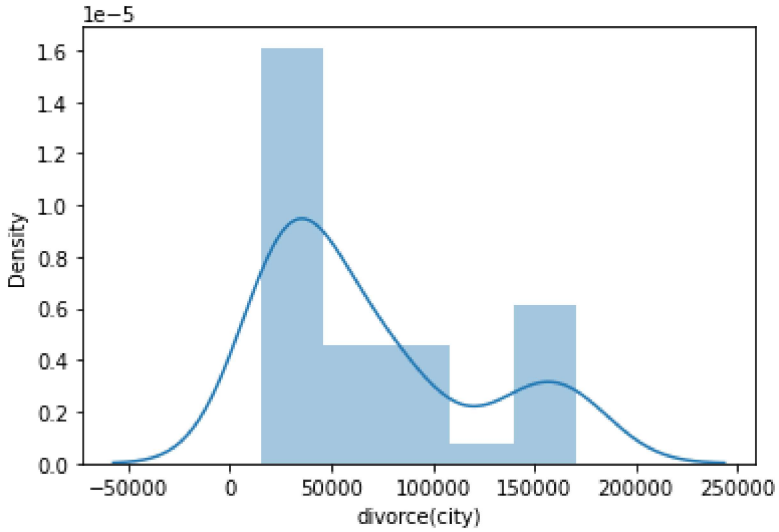
In [31]:
```
sns.distplot(df['marriage(village)'])
plt.show()
```



-In villages marriage ranges from 70,000 to 2,70,000 which is significantly alot less compared to the city.

-Here probability desnity rises from 70,000 upto 1,35,000 and then goes down.

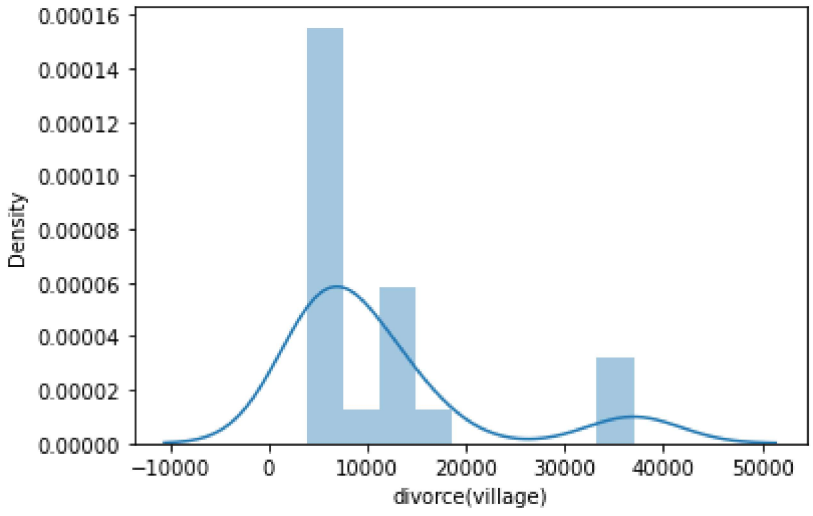
In [32]:
```
sns.distplot(df['divorce(city)'])
plt.show()
```



-Divorce rates ranges from 10,000 to 1,75,000.

-Divorce rates between 10,000 to 40,000 seems to have the most count.

In [33]:
```
sns.distplot(df['divorce(village)'])
plt.show()
```



-Divorce rates in villages range from 4,000 to 38,000.

-4,000 to 8,000 have the highest count.

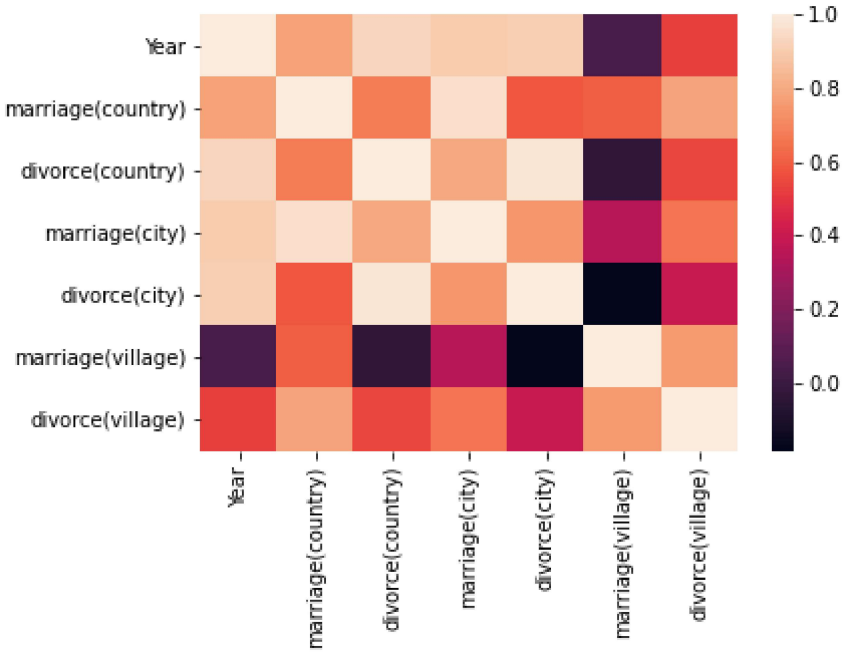
BIVARIATE ANALYSIS.

In [34]:
```
df.corr()
```

Out[34]:

	Year	marriage(country)	divorce(country)	marriage(city)	divorce(city)	marria
Year	1.000000	0.781261	0.926984	0.900624	0.910614	
marriage(country)	0.781261	1.000000	0.673444	0.959257	0.583722	
divorce(country)	0.926984	0.673444	1.000000	0.798112	0.981391	
marriage(city)	0.900624	0.959257	0.798112	1.000000	0.748836	
divorce(city)	0.910614	0.583722	0.981391	0.748836	1.000000	
marriage(village)	0.040549	0.602053	-0.032324	0.354376	-0.183671	
divorce(village)	0.530359	0.782970	0.545513	0.654027	0.402746	

In [35]:
```
corr=df.corr()
sns.heatmap(corr)
plt.show()
```



-Year seems to have a +ve correlation with marriage(country),marriage(city),divorce(country),divorce(city) and no correlation with marriage(village) and moderate +ve correlation with divorce(village).

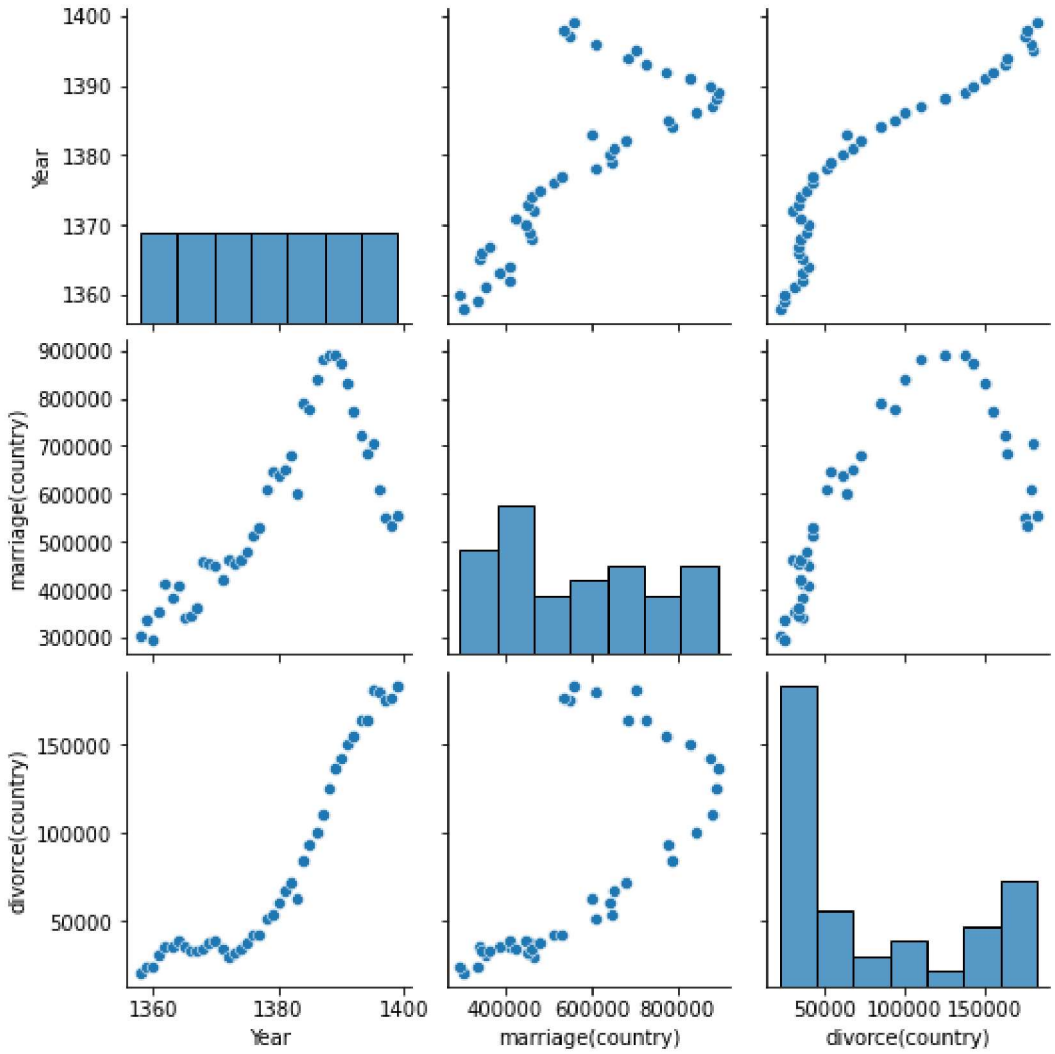
-Marrige(country) have a strong +ve correlation with marriage(city) and marriage(village).

-Divorce(country) has a strong +ve correlation with divorce(city) and marriage(city) and a weak -ve correlation with marriage(village).

-Divorce(city) has a weak -ve correlation with marriage(village).

-Marriage(village) has a strong +ve correlation with divorce(village).

In [36]:
```
sns.pairplot(df[['Year',"marriage(country)","divorce(country)"]])
plt.show()
```



-Marriages in country peaks around 1385 to 1392.

-Divorces in the country peak at the later stages as we head towards 1400.

-What's interesting is after around 1387 we saw a decline in marriage yet divorce rate kept going up.

-Around 1383 we saw a slight dip in divorces but then a couple years later somewhere around 1385 we saw a slight and sudden dip in marriages too.

Overall Conclusion

We noticed the trend of marriages was directly proportional to the divorces. Only in the end we noticed the trend transitioned into an inversely proportional.