In [17]:
```
!pip install folium
import pandas as pd
pd.set_option('display.max_columns',None)
pd.set_option('display.max_rows', None)
import numpy as np
import folium
import json
import requests
import os
from geopy.geocoders import Nominatim
import requests
from pandas.io.json import json_normalize
import matplotlib.cm as cm
import matplotlib.colors as colors
!pip install bs4
from bs4 import BeautifulSoup
from sklearn.cluster import KMeans
print("Libraries imported")
```

```
Requirement already satisfied: folium in c:\users\c3273214\appdata\local\c
ontinuum\anaconda3\lib\site-packages (0.10.0)
Requirement already satisfied: requests in c:\users\c3273214\appdata\local
\continuum\anaconda3\lib\site-packages (from folium) (2.21.0)
Requirement already satisfied: branca>=0.3.0 in c:\users\c3273214\appdata\
local\continuum\anaconda3\lib\site-packages (from folium) (0.3.1)
Requirement already satisfied: jinja2>=2.9 in c:\users\c3273214\appdata\lo
cal\continuum\anaconda3\lib\site-packages (from folium) (2.10)
Requirement already satisfied: numpy in c:\users\c3273214\appdata\local\co
ntinuum\anaconda3\lib\site-packages (from folium) (1.16.2)
Requirement already satisfied: idna<2.9,>=2.5 in c:\users\c3273214\appdata
\local\continuum\anaconda3\lib\site-packages (from requests->folium) (2.8)
Requirement already satisfied: urllib3<1.25,>=1.21.1 in c:\users\c3273214\
appdata\local\continuum\anaconda3\lib\site-packages (from requests->folium
) (1.24.1)
Requirement already satisfied: chardet<3.1.0,>=3.0.2 in c:\users\c3273214\
appdata\local\continuum\anaconda3\lib\site-packages (from requests->folium
) (3.0.4)
Requirement already satisfied: certifi>=2017.4.17 in c:\users\c3273214\app
data\local\continuum\anaconda3\lib\site-packages (from requests->folium) (
2019.3.9)
Requirement already satisfied: six in c:\users\c3273214\appdata\local\cont
inuum\anaconda3\lib\site-packages (from branca>=0.3.0->folium) (1.12.0)
Requirement already satisfied: MarkupSafe>=0.23 in c:\users\c3273214\appda
ta\local\continuum\anaconda3\lib\site-packages (from jinja2>=2.9->folium)
(1.1.1)
Requirement already satisfied: bs4 in c:\users\c3273214\appdata\local\cont
inuum\anaconda3\lib\site-packages (0.0.1)
Requirement already satisfied: beautifulsoup4 in c:\users\c3273214\appdata
\local\continuum\anaconda3\lib\site-packages (from bs4) (4.7.1)
Requirement already satisfied: soupsieve>=1.2 in c:\users\c3273214\appdata
\local\continuum\anaconda3\lib\site-packages (from beautifulsoup4->bs4) (1
.8)
Libraries imported
```

Creation of a Pandas dataframe for Dementia Burden by Australian States

In [18]:
```
df2 = {
    'States':['ACT','NSW','NT','QUEENSLAND','SOUTH AUSTRALIA','TASMANIA','
VICTORIA','WESTERN AUSTRALIA'],
    'Dem_Prev':[5932*100/447116, 149250*100/447116, 1764*100/447116, 84940
*100/447116, 37551*100/447116, 11270*100/447116, 114779*100/447116, 41630*
100/447116]}
ausdem_df2 = pd.DataFrame(df2, columns = ['States', 'Dem_Prev'])
print(ausdem_df2)
```

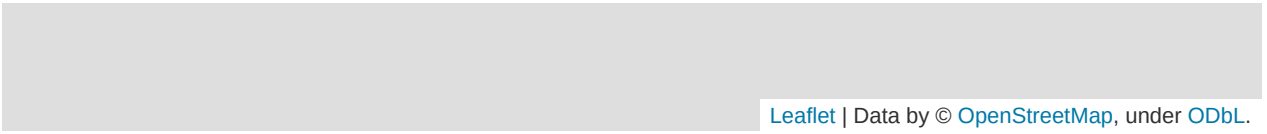
```
              States   Dem_Prev
0                ACT   1.326725
1                NSW  33.380599
2                 NT   0.394528
3         QUEENSLAND  18.997307
4    SOUTH AUSTRALIA   8.398492
5           TASMANIA   2.520599
6           VICTORIA  25.670967
7  WESTERN AUSTRALIA   9.310783
```

General view of the map of Australia

In [19]:
```
!pip install plotly
aus_map = folium.Map(location = [-25.734968, 134.489563], zoom_start = 4)
aus_map
```

```
Requirement already satisfied: plotly in c:\users\c3273214\appdata\local\c
ontinuum\anaconda3\lib\site-packages (4.1.0)
Requirement already satisfied: six in c:\users\c3273214\appdata\local\cont
inuum\anaconda3\lib\site-packages (from plotly) (1.12.0)
Requirement already satisfied: retrying>=1.3.3 in c:\users\c3273214\appdat
a\local\continuum\anaconda3\lib\site-packages (from plotly) (1.3.3)
```

Out[19]:



In [20]:
```
import plotly.graph_objects as go
```

Pandas Dataframe for the geographical coordinates of Australian states plus dementia frequency plus general hospital data

In [21]:
```
import folium
import pandas as pd
import json
aus_data = pd.DataFrame({
    'lat':[-35.473469, -31.840233, -19.491411, -20.917574, -30.000233, -41
.640079, -37.020100, -25.760321],
    'lon':[149.012375, 145.612793, 132.550964, 142.702789, 136.209152, 146
.315918, 144.964600, 122.805176],
    'name':['ACT', 'NSW', 'NT', 'QUEENSLAND', 'SOUTH AUSTRALIA', 'TASMANIA
', 'VICTORIA', 'WESTERN AUSTRALIA'],
    'value':[5932, 149250, 1764, 84940, 37551, 11270, 114779, 41630]
})
aus_data
```

Out[21]:

	lat	lon	name	value
0	-35.473469	149.012375	ACT	5932
1	-31.840233	145.612793	NSW	149250
2	-19.491411	132.550964	NT	1764
3	-20.917574	142.702789	QUEENSLAND	84940
4	-30.000233	136.209152	SOUTH AUSTRALIA	37551
5	-41.640079	146.315918	TASMANIA	11270
6	-37.020100	144.964600	VICTORIA	114779
7	-25.760321	122.805176	WESTERN AUSTRALIA	41630

In [22]:
```
hosp_data = pd.DataFrame({
    'name':['ACT', 'NSW', 'NT', 'QUEENSLAND', 'SOUTH AUSTRALIA', 'TASMANIA
', 'VICTORIA', 'WESTERN AUSTRALIA'],
    'pubhosp':[3, 214, 5, 119, 75, 22, 148, 87],
    'pubpsych':[0, 8, 0, 4, 2, 1, 3, 4],
    'privhosp':[0, 205, 0, 109, 56, 0, 169, 62],
    'total':[3, 427, 5, 232, 133, 23, 320, 153]
})
hosp_data
```

Out[22]:

	name	pubhosp	pubpsych	privhosp	total

0	ACT	3	0	0	3
1	NSW	214	8	205	427
2	NT	5	0	0	5
3	QUEENSLAND	119	4	109	232
4	SOUTH AUSTRALIA	75	2	56	133
5	TASMANIA	22	1	0	23
6	VICTORIA	148	3	169	320
7	WESTERN AUSTRALIA	87	4	62	153

Merging dataframes of dementia burden, geographical coordinates and hospital data

In [23]:
```
aus_gp = pd.merge(ausdem_df2, aus_data, how = 'left', left_on = 'States',
right_on = 'name')
aus_gp.drop(labels = 'name', axis = 1, inplace = True)
aus_gp
```

Out[23]:

	States	Dem_Prev	lat	lon	value
0	ACT	1.326725	-35.473469	149.012375	5932
1	NSW	33.380599	-31.840233	145.612793	149250
2	NT	0.394528	-19.491411	132.550964	1764
3	QUEENSLAND	18.997307	-20.917574	142.702789	84940
4	SOUTH AUSTRALIA	8.398492	-30.000233	136.209152	37551
5	TASMANIA	2.520599	-41.640079	146.315918	11270
6	VICTORIA	25.670967	-37.020100	144.964600	114779
7	WESTERN AUSTRALIA	9.310783	-25.760321	122.805176	41630

In [24]:
```
aus_gp2 = pd.merge(aus_gp, hosp_data, how = 'left', left_on = 'States', ri
ght_on = 'name', validate = "1:1")
aus_gp2.drop(labels = 'name', axis = 1, inplace = True)
aus_gp2
```

Out[24]:

	States	Dem_Prev	lat	lon	value	pubhosp	pubpsych	privhosp	total
0	ACT	1.326725	-35.473469	149.012375	5932	3	0	0	3
1	NSW	33.380599	-31.840233	145.612793	149250	214	8	205	427
2	NT	0.394528	-19.491411	132.550964	1764	5	0	0	5
3	QUEENSLAND	18.997307	-20.917574	142.702789	84940	119	4	109	232
4	SOUTH AUSTRALIA	8.398492	-30.000233	136.209152	37551	75	2	56	133

5	TASMANIA	2.520599	-41.640079	146.315918	11270	22	1	0	23
6	VICTORIA	25.670967	-37.020100	144.964600	114779	148	3	169	320
7	WESTERN AUSTRALIA	9.310783	-25.760321	122.805176	41630	87	4	62	153

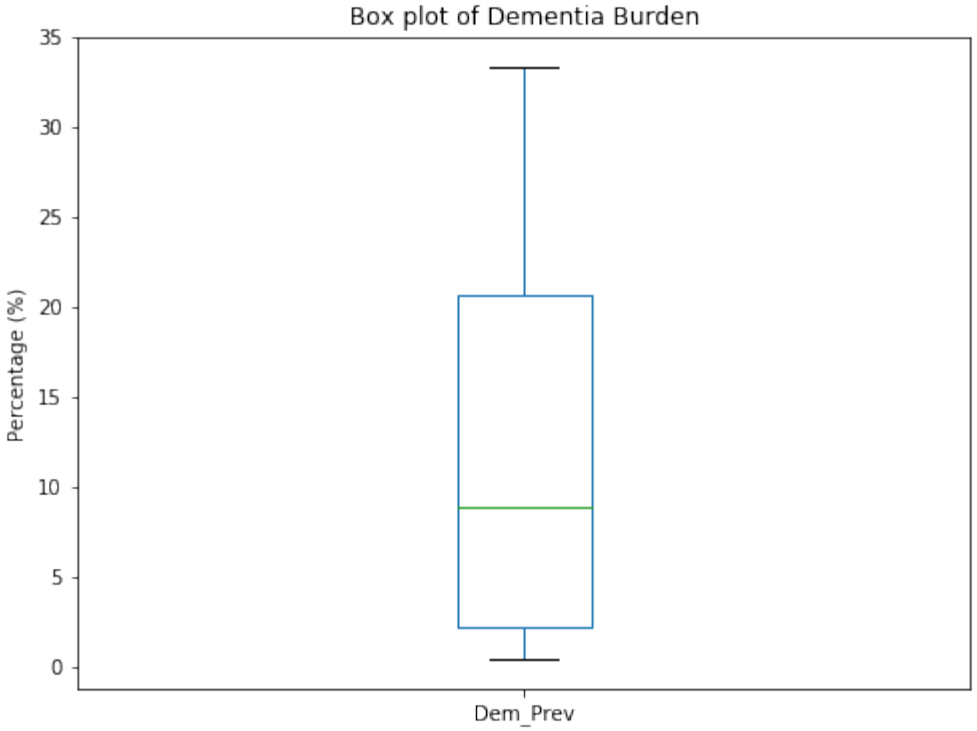
In [25]:
```
# Pre-processing

aus_gp2.dtypes
```

Out[25]:
```
States       object
Dem_Prev    float64
lat         float64
lon         float64
value         int64
pubhosp       int64
pubpsych      int64
privhosp      int64
total         int64
dtype: object
```

Descriptive data and visualizations

In [200]:
```
aus_gp2['Dem_Prev'].plot(kind = 'box', figsize = (8, 6))
plt.title('Box plot of Dementia Burden')
plt.ylabel('Percentage (%)')
```

Out[200]:
```
Text(0, 0.5, 'Percentage (%)')
```



In [26]:
```
# Descriptive data
```

```
aus_gp2.describe()
```

Out[26]:

	Dem_Prev	lat	lon	value	pubhosp	pubpsych	privhosp	tc
count	8.000000	8.000000	8.000000	8.000000	8.000000	8.000000	8.000000	8.000(
mean	12.500000	-30.267928	140.021721	55889.500000	84.125000	2.750000	75.125000	162.000(
std	12.245438	7.823409	8.864748	54751.311765	74.600723	2.659216	79.549513	155.958
min	0.394528	-41.640079	122.805176	1764.000000	3.000000	0.000000	0.000000	3.000(
25%	2.222130	-35.860127	135.294605	9935.500000	17.750000	0.750000	0.000000	18.500(
50%	8.854637	-30.920233	143.833694	39590.500000	81.000000	2.500000	59.000000	143.000(
75%	20.665722	-24.549634	145.788574	92399.750000	126.250000	4.000000	124.000000	254.000(
max	33.380599	-19.491411	149.012375	149250.000000	214.000000	8.000000	205.000000	427.000(

In [27]:
```
aus_gp2.describe(include=['object'])
```

Out[27]:

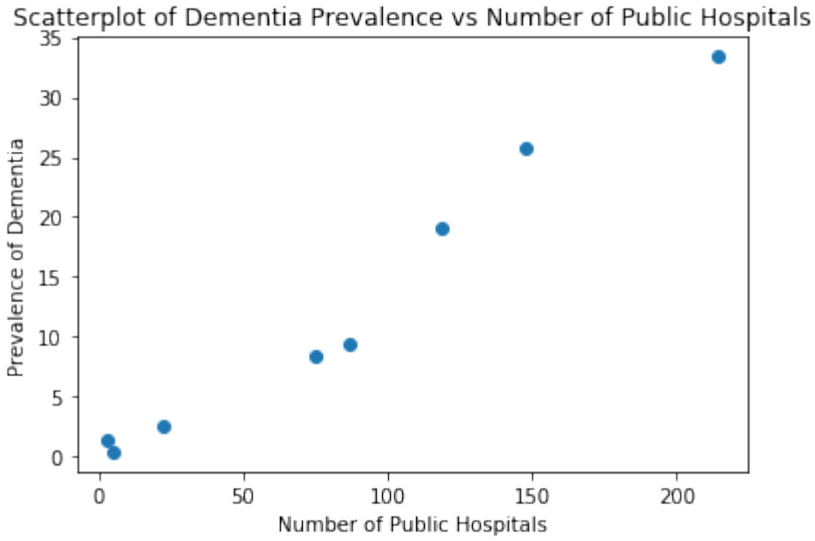
	States
count	8
unique	8
top	WESTERN AUSTRALIA
freq	1

In [28]:
```
import seaborn as sns
```

Data Visualization

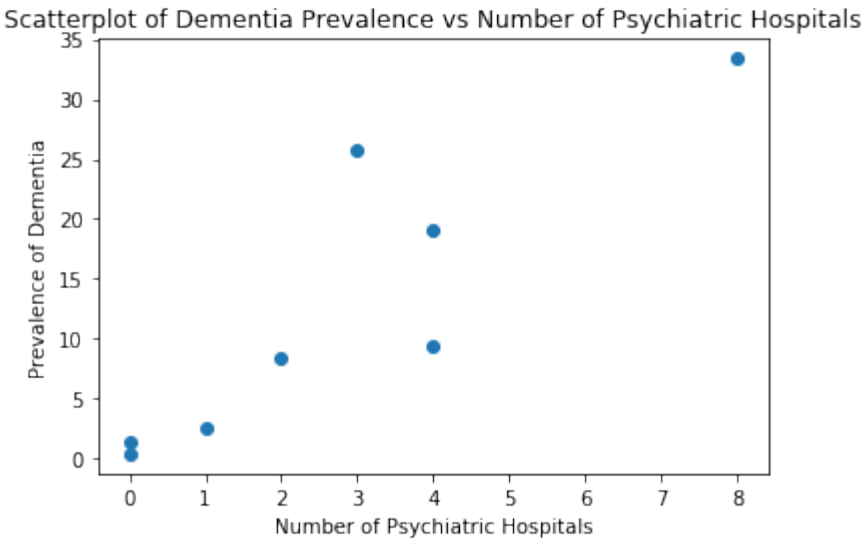
In [29]:
```
import matplotlib.pyplot as plt
y = aus_gp2["Dem_Prev"]
x = aus_gp2["pubhosp"]
plt.scatter(x, y)
plt.title("Scatterplot of Dementia Prevalence vs Number of Public Hospitals")
plt.xlabel("Number of Public Hospitals")
plt.ylabel("Prevalence of Dementia")
```

Out[29]:
```
Text(0, 0.5, 'Prevalence of Dementia')
```

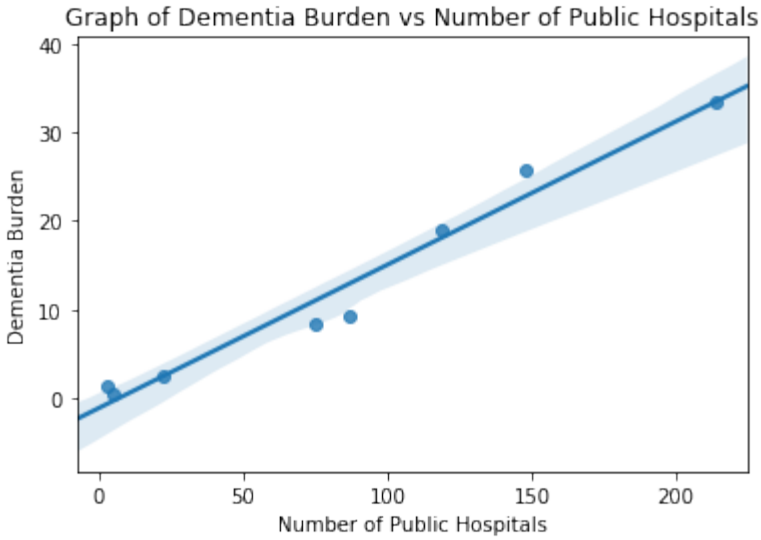


In [30]:
```
y = aus_gp2["Dem_Prev"]
x = aus_gp2["pubpsych"]
plt.scatter(x, y)
plt.title("Scatterplot of Dementia Prevalence vs Number of Psychiatric Hos
pitals")
plt.xlabel("Number of Psychiatric Hospitals")
plt.ylabel("Prevalence of Dementia")
```

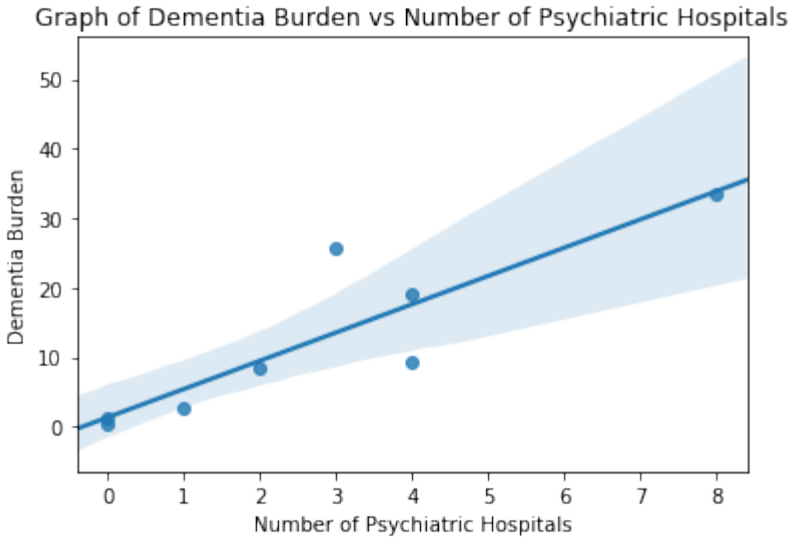
Out[30]:
```
Text(0, 0.5, 'Prevalence of Dementia')
```



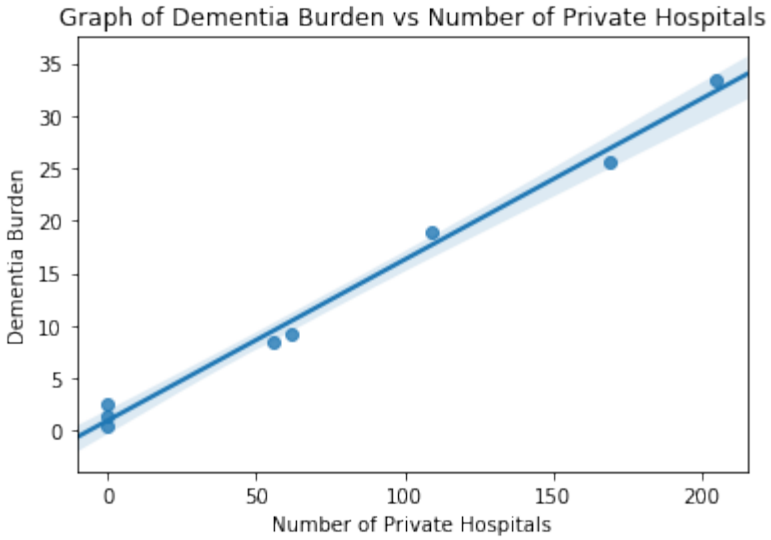
In [31]:
```
sns.regplot(x = "pubhosp", y = "Dem_Prev", data = aus_gp2)
plt.title("Graph of Dementia Burden vs Number of Public Hospitals")
plt.xlabel("Number of Public Hospitals")
plt.ylabel("Dementia Burden")
plt.show()
```



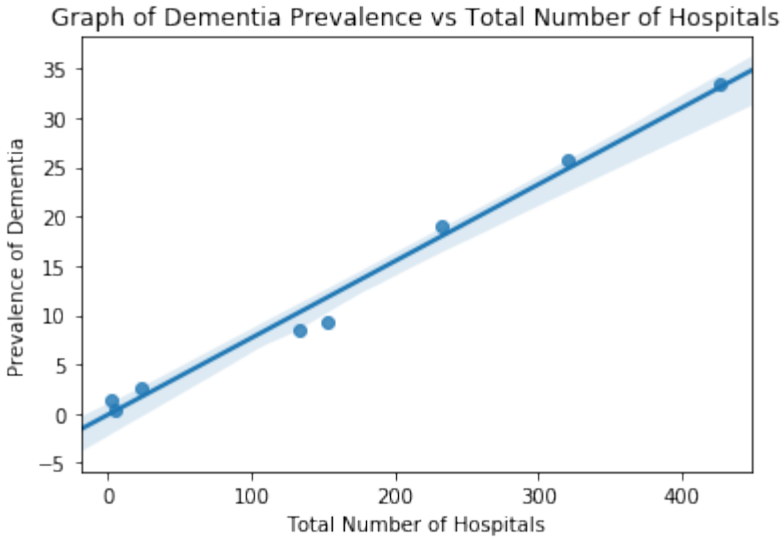
In [32]:
```
sns.regplot(x = "pubpsych", y = "Dem_Prev", data = aus_gp2)
plt.title("Graph of Dementia Burden vs Number of Psychiatric Hospitals")
plt.xlabel("Number of Psychiatric Hospitals")
plt.ylabel("Dementia Burden")
plt.show()
```



In [33]:
```
sns.regplot(x = "privhosp", y = "Dem_Prev", data = aus_gp2)
plt.title("Graph of Dementia Burden vs Number of Private Hospitals")
plt.xlabel("Number of Private Hospitals")
plt.ylabel("Dementia Burden")
plt.show()
```



In [34]:
```
sns.regplot(x = "total", y = "Dem_Prev", data = aus_gp2)
plt.title("Graph of Dementia Prevalence vs Total Number of Hospitals")
plt.xlabel("Total Number of Hospitals")
plt.ylabel("Prevalence of Dementia")
plt.show()
```



Correlation matrix including hospitals, dementia burden and geographical coordinates and, heatmap

In [179]:
```
aus_gp2[['Dem_Prev', 'pubhosp', 'privhosp', 'pubpsych', 'total', 'lat', 'lon']].corr()
```

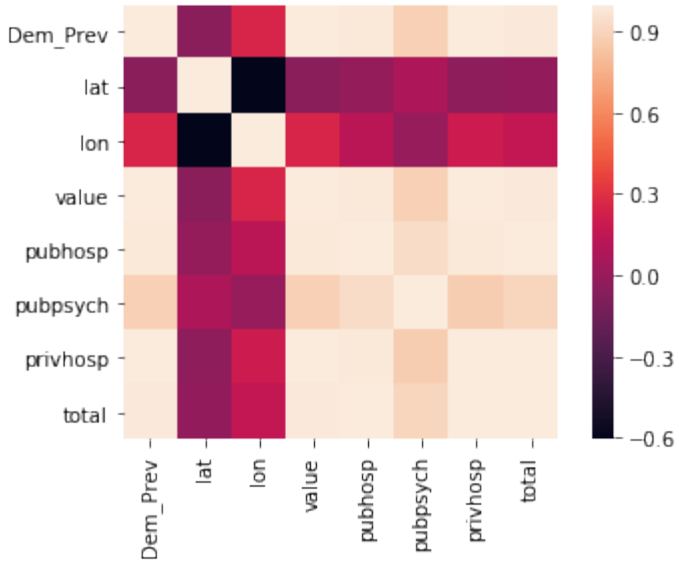
Out[179]:

	Dem_Prev	pubhosp	privhosp	pubpsych	total	lat	lon
Dem_Prev	1.000000	0.983992	0.995362	0.884466	0.993462	-0.052623	0.259231
pubhosp	0.983992	1.000000	0.984274	0.937057	0.996360	-0.005577	0.127752
privhosp	0.995362	0.984274	1.000000	0.871334	0.995738	-0.034383	0.201127
pubpsych	0.884466	0.937057	0.871334	1.000000	0.909718	0.080754	0.000511
total	0.993462	0.996360	0.995738	0.909718	1.000000	-0.018829	0.163705
lat	-0.052623	-0.005577	-0.034383	0.080754	-0.018829	1.000000	-0.606861

lon	0.259231	0.127752	0.201127	0.000511	0.163705	-0.606861	1.000000

In [180]:
```
cor = aus_gp2.corr()
sns.heatmap(cor, square = True)
```

Out[180]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x231b1980240>
```



Linear regression (simple and multiple linear regression) using Scikit-Learn and Scipy

In [40]:
```
import scipy
from scipy import stats
from scipy.stats import linregress
```

In [41]:
```
linregress(aus_gp2["pubhosp"], aus_gp2["Dem_Prev"])
```

Out[41]:
```
LinregressResult(slope=0.16151864982780498, intercept=-1.087756416764094,
rvalue=0.9839916069670939, pvalue=1.0133379289318787e-05, stderr=0.0119426
1524289986)
```

In [43]:
```
linregress(aus_gp2["pubpsych"], aus_gp2["Dem_Prev"])
```

Out[43]:
```
LinregressResult(slope=4.072881545394272, intercept=1.2995757501657526, rv
alue=0.8844657936818925, pvalue=0.0035290652990349813, stderr=0.8771917423
499831)
```

In [44]:
```
linregress(aus_gp2["privhosp"], aus_gp2["Dem_Prev"])
```

Out[44]:
```
LinregressResult(slope=0.15322089396826152, intercept=0.9892803406343535,
rvalue=0.9953623265663488, pvalue=2.485013124576096e-07, stderr=0.00604535
95219430905)
```

In [45]:
```
linregress(aus_gp2["total"], aus_gp2["Dem_Prev"])
```

Out[45]:
```
LinregressResult(slope=0.07800376053087549, intercept=-0.1366092060018289,
 rvalue=0.9934615614301954, pvalue=6.953925056560325e-07, stderr=0.0036595
678515175616)
```

In [46]:
```
from scipy.stats.stats import pearsonr
```

In [47]:
```
pearson_coef, p_value = stats.pearsonr(aus_gp2['pubhosp'], aus_gp2['Dem_Pr
ev'])
print("The Pearson correlation coefficient is", pearson_coef, "with a P-va
lue of P =", p_value)
```

```
The Pearson correlation coefficient is 0.9839916069670939 with a P-value o
f P = 1.0133379289318792e-05
```

In [48]:
```
from sklearn.linear_model import LinearRegression
lm = LinearRegression()
```

In [49]:
```
X = aus_gp2[['pubhosp']]
Y = aus_gp2[['Dem_Prev']]
lm.fit(X, Y)
```

Out[49]:
```
LinearRegression(copy_X=True, fit_intercept=True, n_jobs=None,
         normalize=False)
```

In [50]:
```
lm.coef_
```

Out[50]:
```
array([[0.16151865]])
```

In [51]:
```
X = aus_gp2[['pubhosp', 'lat', 'lon']]
Y = aus_gp2[['Dem_Prev']]
lm.fit(X, Y)
```

Out[51]:
```
LinearRegression(copy_X=True, fit_intercept=True, n_jobs=None,
         normalize=False)
```

In [52]:
```
lm.coef_
```

Out[52]:
```
array([[0.15799866, 0.08780633, 0.2352561 ]])
```

In [53]:
```
print("The R-square is:", lm.score(X, Y))
```

```
The R-square is: 0.9883353768586333
```

In [54]:
```
X = aus_gp2[['privhosp']]
Y = aus_gp2[['Dem_Prev']]
lm.fit(X, Y)
```

Out[54]:
```
LinearRegression(copy_X=True, fit_intercept=True, n_jobs=None,
         normalize=False)
```

In [55]:
```
lm.coef_
```

Out[55]:
```
array([[0.15322089]])
```

In [56]:
```
X = aus_gp2[['privhosp', 'lat', 'lon']]
```

```
Y = aus_gp2[['Dem_Prev']]
lm.fit(X, Y)
```

Out[56]:
```
LinearRegression(copy_X=True, fit_intercept=True, n_jobs=None,
         normalize=False)
```

In [57]:
```
lm.coef_
```

Out[57]:
```
array([[0.15089857, 0.04645849, 0.11062449]])
```

In [58]:
```
print("The R-square is:", lm.score(X, Y))
```

```
The R-square is: 0.9949279085600867
```

In [59]:
```
print("The R-square is:", lm.score(X, Y))
```

```
The R-square is: 0.9949279085600867
```

In [60]:
```
X = aus_gp2[['pubpsych']]
Y = aus_gp2[['Dem_Prev']]
lm.fit(X, Y)
```

Out[60]:
```
LinearRegression(copy_X=True, fit_intercept=True, n_jobs=None,
         normalize=False)
```

In [61]:
```
lm.coef_
```

Out[61]:
```
array([[4.07288155]])
```

In [62]:
```
X = aus_gp2[['pubpsych', 'lat', 'lon']]
Y = aus_gp2[['Dem_Prev']]
lm.fit(X, Y)
```

Out[62]:
```
LinearRegression(copy_X=True, fit_intercept=True, n_jobs=None,
         normalize=False)
```

In [63]:
```
print("The R-square is:", lm.score(X, Y))
```

```
The R-square is: 0.8509887036485619
```

In [64]:
```
lm.coef_
```

Out[64]:
```
array([[4.05256343, 0.08263982, 0.40172993]])
```

In [65]:
```
X = aus_gp2[['total']]
Y = aus_gp2[['Dem_Prev']]
lm.fit(X, Y)
```

Out[65]:
```
LinearRegression(copy_X=True, fit_intercept=True, n_jobs=None,
         normalize=False)
```

In [66]:
```
lm.coef_
```

Out[66]:
```
array([[0.07800376]])
```

In [67]:
```
X = aus_gp2[['total', 'lat', 'lon']]
```

```
Y = aus_gp2[['Dem_Prev']]
lm.fit(X, Y)
```

Out[67]:
```
LinearRegression(copy_X=True, fit_intercept=True, n_jobs=None,
         normalize=False)
```

In [68]:
```
lm.coef_
```

Out[68]:
```
array([[0.07645763, 0.065135  , 0.17277145]])
```

In [69]:
```
from sklearn.metrics import r2_score
```

In [70]:
```
X = aus_gp2[['total']]
Y = aus_gp2[['Dem_Prev']]
lm.fit(X, Y)
```

Out[70]:
```
LinearRegression(copy_X=True, fit_intercept=True, n_jobs=None,
         normalize=False)
```

In [71]:
```
print("The R-square is:", lm.score(X, Y))
```

```
The R-square is: 0.986965874039322
```

In [72]:
```
X = aus_gp2[['pubhosp']]
Y = aus_gp2[['Dem_Prev']]
lm.fit(X, Y)
```

Out[72]:
```
LinearRegression(copy_X=True, fit_intercept=True, n_jobs=None,
         normalize=False)
```

In [73]:
```
print("The R-square is:", lm.score(X, Y))
```

```
The R-square is: 0.9682394825816838
```

In [74]:
```
X = aus_gp2[['pubpsych']]
Y = aus_gp2[['Dem_Prev']]
lm.fit(X, Y)
```

Out[74]:
```
LinearRegression(copy_X=True, fit_intercept=True, n_jobs=None,
         normalize=False)
```

In [75]:
```
print("The R-square is:", lm.score(X, Y))
```

```
The R-square is: 0.7822797401933401
```

In [76]:
```
X = aus_gp2[['privhosp']]
Y = aus_gp2[['Dem_Prev']]
lm.fit(X, Y)
```

Out[76]:
```
LinearRegression(copy_X=True, fit_intercept=True, n_jobs=None,
         normalize=False)
```

In [77]:
```
print("The R-square is:", lm.score(X, Y))
```

```
The R-square is: 0.9907461611475749
```

In [189]:
```
X = aus_gp2[['lat']]
Y = aus_gp2[['Dem_Prev']]
lm.fit(X, Y)
lm.coef_
print("The R-square is>", lm.score(X, Y))
```

```
The R-square is> 0.002769198391169292
```

In [190]:
```
lm.coef_
```

Out[190]:
```
array([[-0.0823674]])
```

In [191]:
```
X = aus_gp2[['lon']]
Y = aus_gp2[['Dem_Prev']]
lm.fit(X, Y)
lm.coef_
print("The R-square is>", lm.score(X, Y))
```

```
The R-square is> 0.0672005628472816
```

In [192]:
```
lm.coef_
```

Out[192]:
```
array([[0.35809178]])
```

In [193]:
```
linregress(aus_gp2['lat'], aus_gp['Dem_Prev'])
```

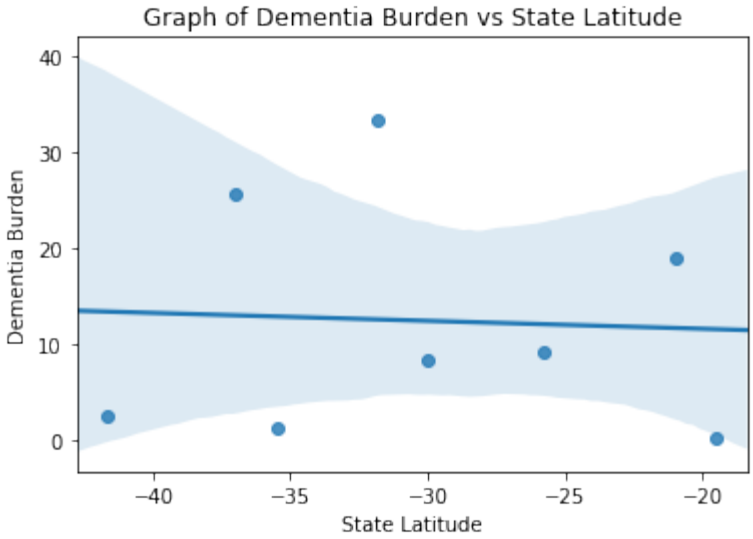
Out[193]:
```
LinregressResult(slope=-0.08236739564069347, intercept=10.006909640383675,
 rvalue=-0.05262317351860496, pvalue=0.9015135533349156, stderr=0.63811729
55272034)
```

In [194]:
```
linregress(aus_gp2['lon'], aus_gp['Dem_Prev'])
```

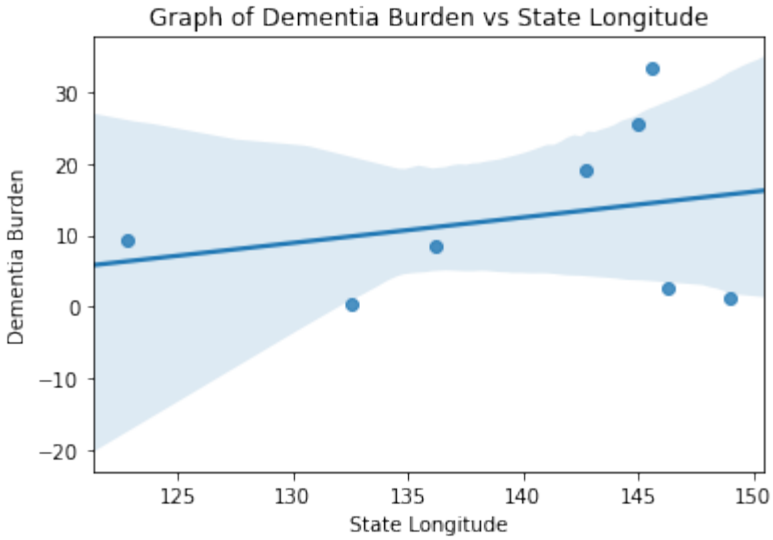
Out[194]:
```
LinregressResult(slope=0.35809178413369075, intercept=-37.6406278455984, r
value=0.2592307135493044, pvalue=0.5352789754040121, stderr=0.544661169340
0937)
```

In [231]:
```
# Regression plots for the association of geographical coordinates with de
mentia burden
```

In [195]:
```
sns.regplot(x = "lat", y = "Dem_Prev", data = aus_gp2)
plt.title("Graph of Dementia Burden vs State Latitude")
plt.xlabel("State Latitude")
plt.ylabel("Dementia Burden")
plt.show()
```

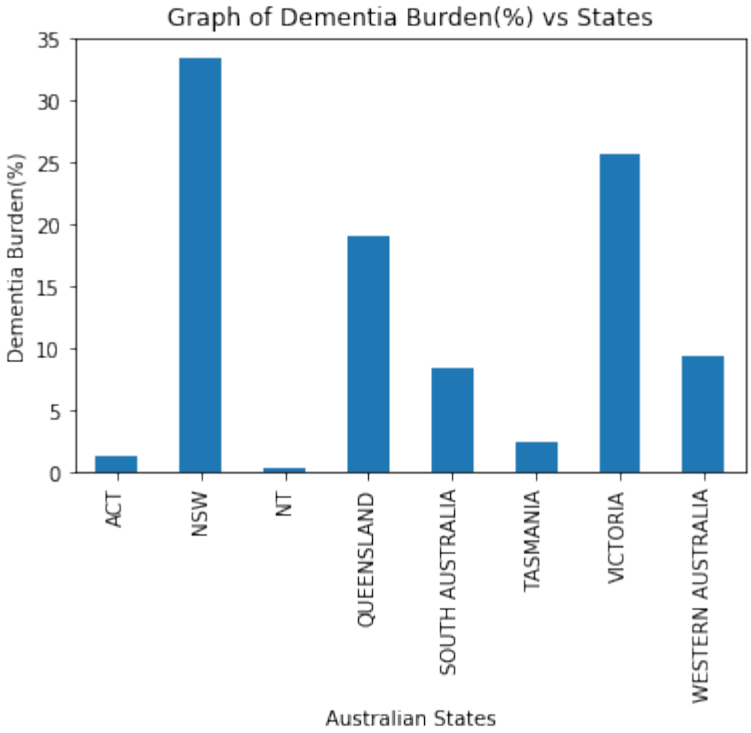


In [196]:
```
sns.regplot(x = "lon", y = "Dem_Prev", data = aus_gp2)
plt.title("Graph of Dementia Burden vs State Longitude")
plt.xlabel("State Longitude")
plt.ylabel("Dementia Burden")
plt.show()
```



In [232]:
```
# More visualizations: Bar graph

import matplotlib as mpl
aus_gp2.groupby('States').mean()['Dem_Prev'].plot(kind = 'bar')
plt.title("Graph of Dementia Burden(%) vs States")
plt.xlabel("Australian States")
plt.ylabel("Dementia Burden(%)")
```

Out[232]: Text(0, 0.5, 'Dementia Burden(%)')



Selected city geographical coordinates

In [233]:
```
# Creating a pandas dataframe from a csv file

city_data = pd.read_csv(r'C:\Users\c3273214\Documents\AusCities3.csv')
city_data.head()
```

Out[233]:

	City	Latitude	Longitude	State
0	Canberra	-35.282001	149.128998	ACT
1	Sunshine Coast	-26.650000	153.066666	QLD
2	Gold Coast	-28.016666	153.399994	QLD
3	Melbourne	-37.840935	144.946457	VIC
4	Adelaide	-34.921230	138.599503	SA

In [234]:
```
# Grouping cities data by State

city_datagp = city_data.groupby(["State", "Latitude", "Longitude"], as_ind
ex = False).agg(lambda x:",".join(x))
city_datagp
```

Out[234]:

	State	Latitude	Longitude	City
0	ACT	-35.343784	149.082977	Phillip
1	ACT	-35.282001	149.128998	Canberra
2	NSW	-36.080780	146.916473	Albury

3	NSW	-34.425072	150.893143	Wollongong
4	NSW	-33.917290	151.035889	Bankstown
5	NSW	-33.865143	151.209900	Sydney
6	NSW	-33.807690	150.987274	Westmead
7	NSW	-33.683212	151.224396	Terrey Hills
8	NSW	-33.425018	151.342224	Gosford
9	NSW	-33.283577	149.101273	Orange
10	NSW	-30.296276	153.114136	Coffs Harbour
11	NT	-12.462827	130.841782	Darwin
12	QLD	-28.016666	153.399994	Gold Coast
13	QLD	-27.529953	152.407181	Glenore Grove
14	QLD	-27.470125	153.021072	Brisbane
15	QLD	-26.650000	153.066666	Sunshine Coast
16	QLD	-23.843138	151.268356	Gladstone
17	QLD	-19.258965	146.816956	Townsville City
18	QLD	-16.925491	145.754120	Cairns City
19	SA	-37.824429	140.783783	Mount Gambier
20	SA	-34.921230	138.599503	Adelaide
21	SA	-34.906101	138.593903	North Adelaide
22	TAS	-41.429825	147.157135	Launceston
23	VIC	-37.840935	144.946457	Melbourne
24	VIC	-37.649967	144.880600	Ziyou Today
25	VIC	-36.757786	144.278702	Bendigo
26	VIC	-34.206841	142.136490	Mildura
27	WA	-31.953512	115.857048	Perth

Location of cities in Australia plus hospital data per state

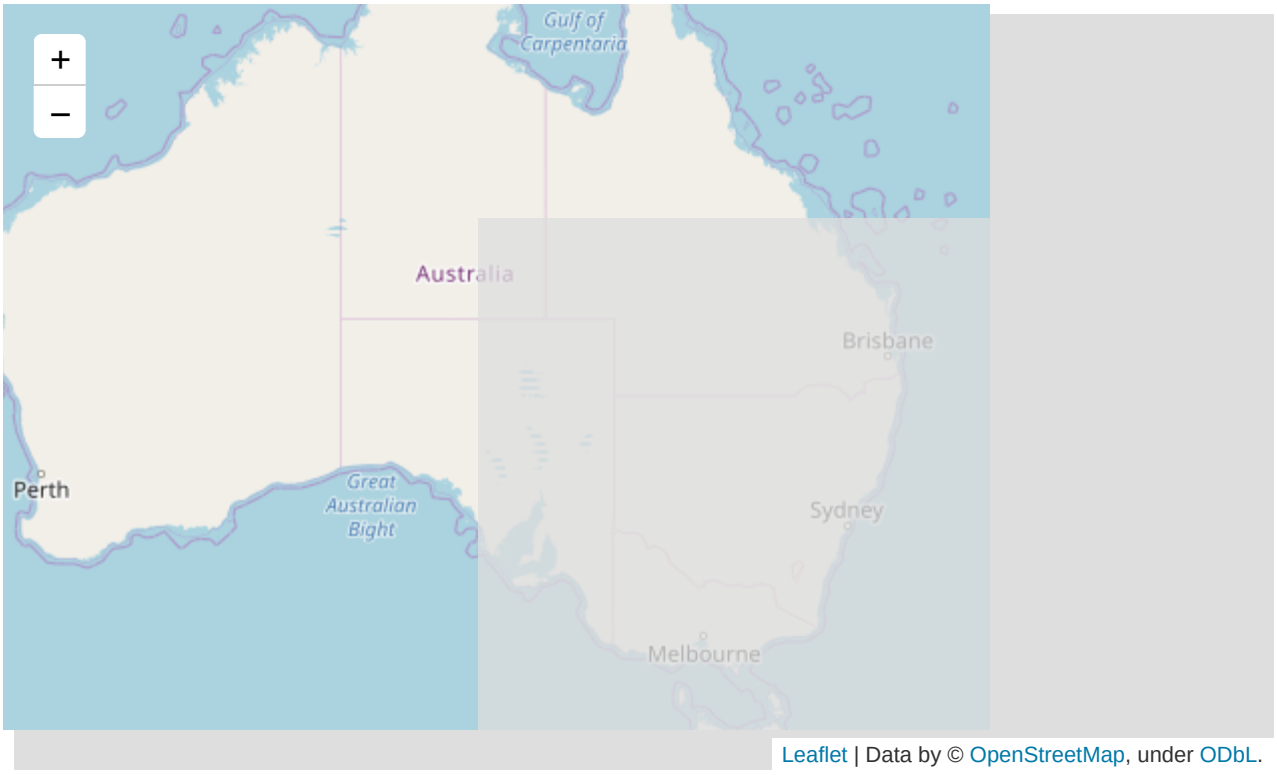
In [238]:
```
address = 'Australia'
geolocator = Nominatim(user_agent = 'my_application')
location = geolocator.geocode(address)
latitude = location.latitude
longitude = location.longitude
print("The geographical coordinates of Australia are {},{}.".format(latitude, longitude))
```

```
The geographical coordinates of Australia are -24.7761086,134.755.
```

In [239]:
```
!pip install plotly
aus_map2 = folium.Map(location = [latitude, longitude], zoom_start = 4)
for lat, lon, city, state in zip(city_data['Latitude'], city_data['Longitude'], city_data['City'], city_data['State']):
    label = '{}'.format(city, state)
    label = folium.Popup(label, parse_html = True)
    folium.CircleMarker(
       [lat, lon],
       radius = 5,
       color = 'blue',
       fill = True,
       fill_color = '#3186cc',
       fill_opacity = 0.7,
       parse_html = False).add_to(aus_map2)
aus_map2
```

```
Requirement already satisfied: plotly in c:\users\c3273214\appdata\local\continuum\anaconda3\lib\site-packages (4.1.0)
Requirement already satisfied: retrying>=1.3.3 in c:\users\c3273214\appdata\local\continuum\anaconda3\lib\site-packages (from plotly) (1.3.3)
Requirement already satisfied: six in c:\users\c3273214\appdata\local\continuum\anaconda3\lib\site-packages (from plotly) (1.12.0)
```

Out[239]:



A closer look at NSW

In [83]:
```
nsw_data = city_data[city_data['State'].str.contains('NSW')].reset_index(drop = True)
print("The shape of the dataframe:", nsw_data.shape)
nsw_data[0:10]
```

```
The shape of the dataframe: (9, 4)
```

Out[83]:

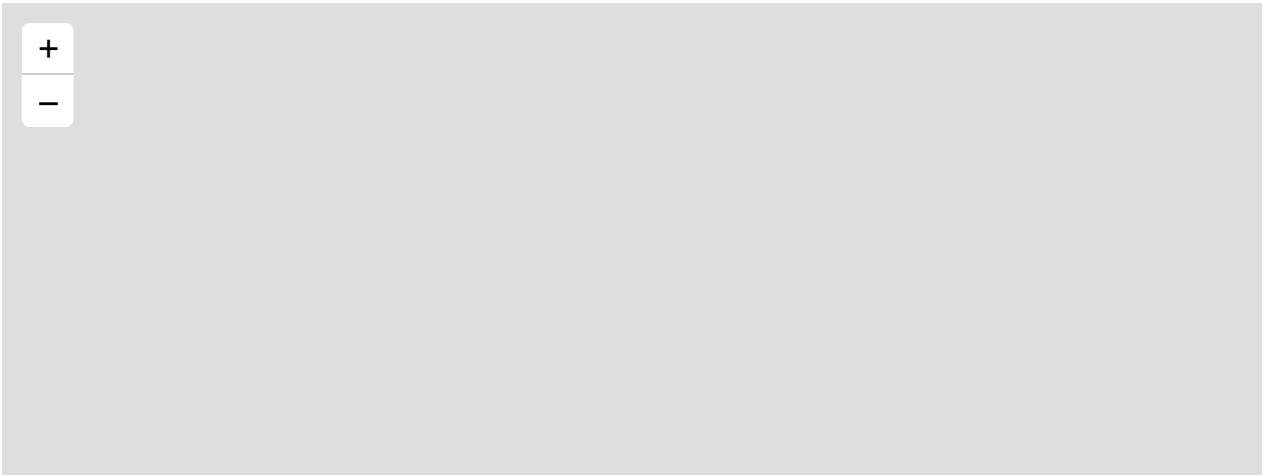
	City	Latitude	Longitude	State
0	Coffs Harbour	-30.296276	153.114136	NSW
1	Orange	-33.283577	149.101273	NSW
2	Albury	-36.080780	146.916473	NSW
3	Wollongong	-34.425072	150.893143	NSW
4	Terrey Hills	-33.683212	151.224396	NSW
5	Bankstown	-33.917290	151.035889	NSW
6	Westmead	-33.807690	150.987274	NSW
7	Gosford	-33.425018	151.342224	NSW
8	Sydney	-33.865143	151.209900	NSW

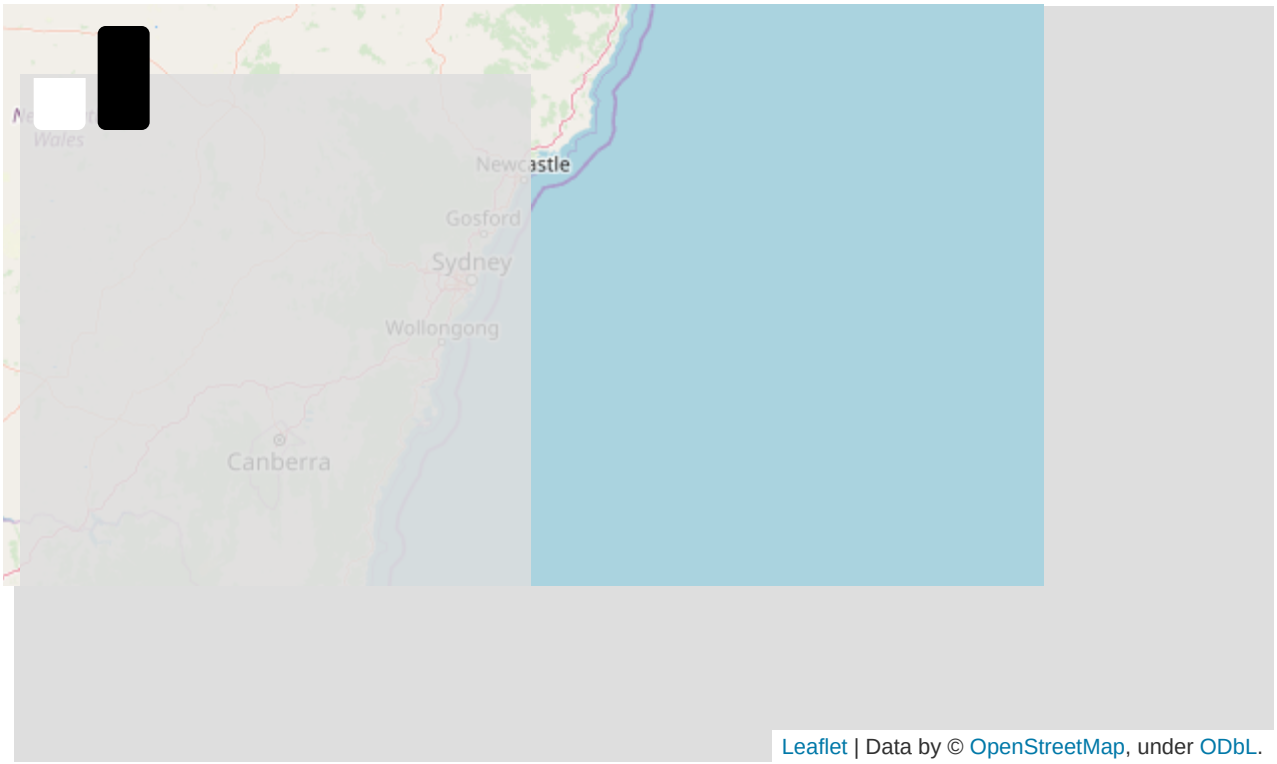
In [84]:
```
address = 'Sydney'
geolocator = Nominatim(user_agent = 'my_application')
location = geolocator.geocode(address)
latitude = location.latitude
longitude = location.longitude
print("The geographical coordinates of Sydney are {},{}.".format(latitude,
 longitude))
```

```
The geographical coordinates of Sydney are -33.8548157,151.2164539.
```

In [85]:
```
nsw_map = folium.Map(location = [latitude, longitude], zoom_start = 6)
for lat, lon, city, state in zip(nsw_data['Latitude'], nsw_data['Longitude
'], nsw_data['City'], nsw_data['State']):
    label = '{}'.format(city, state)
    label = folium.Popup(label, parse_html = True)
    folium.CircleMarker(
       [lat, lon],
       radius = 5,
       color = 'blue',
       fill = True,
       fill_color = '#3186cc',
       fill_opacity = 0.7,
       parse_html = False).add_to(nsw_map)
nsw_map
```

Out[85]:





In [86]:
```
CLIENT_ID = 'L351DHR2FSTSDETU0K5X3IZMC5XCTH5GJMOTYKAE0BEVPVO3'
CLIENT_SECRET = 'U0NV33KA4DBR4L5ZDYKM2XXXOO4RFH5DI1XNRFUGJDIF2MVR'
VERSION = '20180605'
print('Your Credentials:')
print('CLIENT_ID:' + CLIENT_ID)
print('CLIENT_SECRET:' + CLIENT_SECRET)
```

```
Your Credentials:
CLIENT_ID:L351DHR2FSTSDETU0K5X3IZMC5XCTH5GJMOTYKAE0BEVPVO3
CLIENT_SECRET:U0NV33KA4DBR4L5ZDYKM2XXXOO4RFH5DI1XNRFUGJDIF2MVR
```

In [87]:
```
nsw_data.loc[8, "City"]
```

Out[87]:
```
'Sydney'
```

In [88]:
```
city_latitude = nsw_data.loc[8, "Latitude"]
city_longitude = nsw_data.loc[8, "Longitude"]
city_name = nsw_data.loc[8, "City"]
print("The Latitude and Longitude of {} are {}, {}.".format(city_name, cit
y_latitude, city_longitude))
```

```
The Latitude and Longitude of Sydney are -33.865142999999996, 151.2099.
```

In [305]:
```
radius = 5000
limit = 100
query = "hospitals"
url = 'https://api.foursquare.com/v2/venues/explore?&client_id={}&client_s
ecret={}&ll={},{}&v={}&radius={}&limit={}&query={}'.format(CLIENT_ID, CLIE
NT_SECRET, city_latitude, city_longitude, VERSION, radius, limit, query)
```

In [306]:
```
results = requests.get(url).json()
```

In [91]:
```
def get_category_type(row):
    try:
        categories_list = row['categories']
    except:
        categories_list = row['venue.categories']
    if len(categories_list) == 0:
        return None
    else:
        return categories_list[0]['name']
```

In [92]:
```
venues = results['response']['groups'][0]['items']
nearby_venues = json_normalize(venues)
filtered_columns = ['venue.name', 'venue.categories', 'venue.location.lati
tude', 'venue.location.longitude']
nearby_venues = nearby_venues.loc[:, filtered_columns]
nearby_venues['venue.categories'] = nearby_venues.apply(get_category_type,
 axis = 1)
nearby_venues.columns = [col.split(".")[-1] for col in nearby_venues.colum
ns]
print("The size of the dataframe is:", nearby_venues.shape)
nearby_venues
```

```
The size of the dataframe is: (43, 4)
```

```
C:\Users\c3273214\AppData\Local\Continuum\anaconda3\lib\site-packages\pand
as\core\indexing.py:1494: FutureWarning:


Passing list-likes to .loc or [] with any missing label will raise
KeyError in the future, you can use .reindex() as an alternative.

See the documentation here:
https://pandas.pydata.org/pandas-docs/stable/indexing.html#deprecate-loc-r
eindex-listlike
```

Out[92]:

	name	categories	latitude	longitude
0	Sydney Hospital	Hospital	NaN	NaN
1	Sydney Hospital Hand Clinic	Hospital	NaN	NaN
2	Sydney Sexual Health Centre	Hospital	NaN	NaN
3	Sydney Eye Hospital	Hospital	NaN	NaN
4	Boneham Optometrist Eyecare Plus	Hospital	NaN	NaN
5	Genea	Hospital	NaN	NaN
6	East Sydney Private Hospital	Hospital	NaN	NaN
7	Sydney Cardiology	Hospital	NaN	NaN
8	St Vincent's Hospital	Hospital	NaN	NaN
9	St Luke's Hospital	Hospital	NaN	NaN
10	Sacred Heart Palliative Care	Hospital	NaN	NaN
11	O'Brien Centre, St Vincent's Hospital	Hospital	NaN	NaN

12	St Vincent's Private Hospital	Hospital	NaN	NaN
13	VeyeP	Hospital	NaN	NaN
14	Royal Prince Alfred Hospital (RPA)	Hospital	NaN	NaN
15	St Vincent's Clinic	Doctor's Office	NaN	NaN
16	Balmain Hospital	Hospital	NaN	NaN
17	Albion Street Centre	Hospital	NaN	NaN
18	Sydney Dental Hospital	Hospital	NaN	NaN
19	Mater Hospital	Hospital	NaN	NaN
20	Centenary Institute	Hospital	NaN	NaN
21	Wolper Jewish Hospital	Hospital	NaN	NaN
22	RPA Birth Centre	Hospital	NaN	NaN
23	Balanced Bods Health	Hospital	NaN	NaN
24	RPA Intensive Care Unit (ICU)	Hospital	NaN	NaN
25	RPA Women & Babies	Hospital	NaN	NaN
26	Gloucester House	Hospital	NaN	NaN
27	RPA QEII Building 10	Hospital	NaN	NaN
28	RPA Fracture Clinic	Hospital	NaN	NaN
29	RPA King George V Building	Hospital	NaN	NaN
30	The Chris O'Brien Lifehouse at RPA	Hospital	NaN	NaN
31	Mater Imaging	Hospital	NaN	NaN
32	RPA Radiation Oncology	Hospital	NaN	NaN
33	The Dietologist	Hospital	NaN	NaN
34	Alexandria Veterinary Hospital	Hospital	NaN	NaN
35	Rozelle Hospital	Hospital	NaN	NaN
36	RPA Emergency Department	Hospital	NaN	NaN
37	Alfred Medical Imaging	Hospital	NaN	NaN
38	Northside Clinic	Hospital	NaN	NaN
39	mosman private hospital	Hospital	NaN	NaN
40	Greenwich Hospital	Hospital	NaN	NaN
41	Paddington Cat Hospital	Veterinarian	NaN	NaN
42	Bondi Junction Veterinary Hospital	Veterinarian	NaN	NaN

A closer look at ACT and its hospitals

In [93]:
```
act_data = city_data[city_data['State'].str.contains('ACT')].reset_index(d
rop = True)
print("The shape of the dataframe:", nsw_data.shape)
```

```
act_data
```

```
The shape of the dataframe: (9, 4)
```

Out[93]:

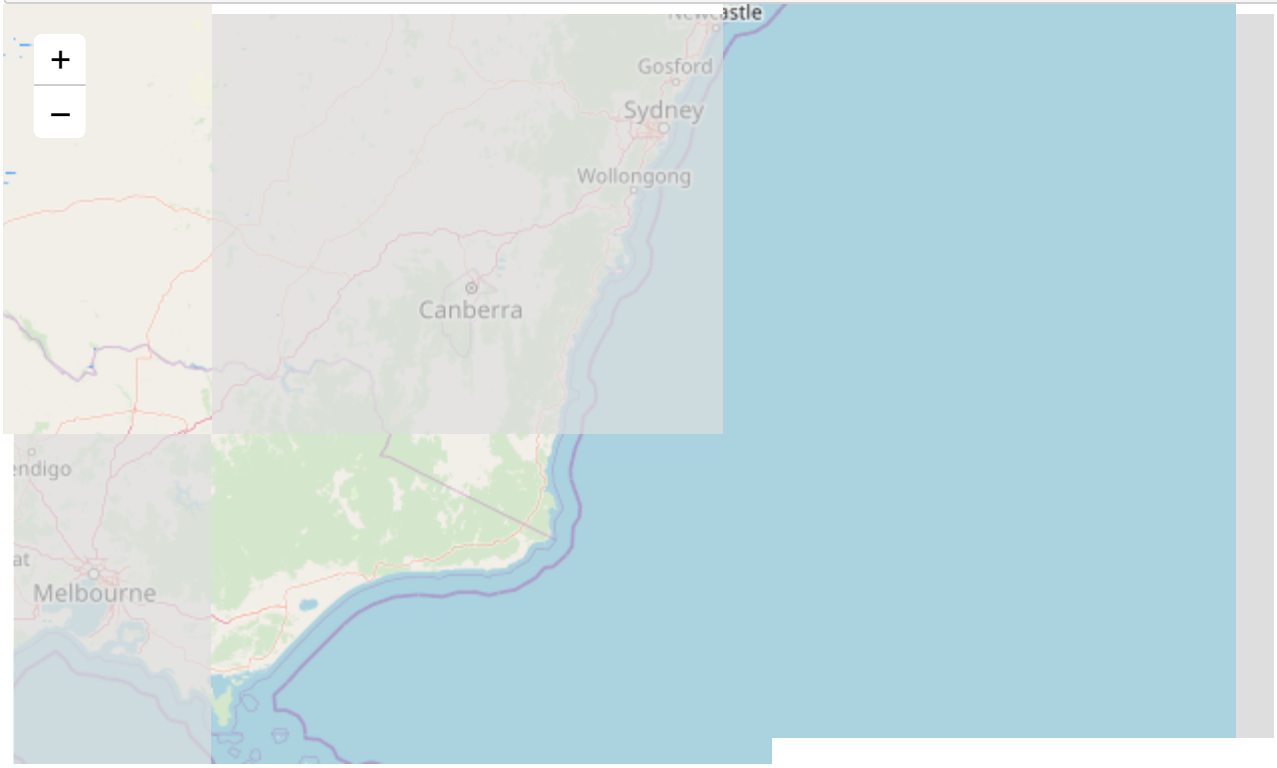
	City	Latitude	Longitude	State
0	Canberra	-35.282001	149.128998	ACT
1	Phillip	-35.343784	149.082977	ACT

In [94]:
```
address = 'Canberra'
geolocator = Nominatim(user_agent = 'my_application')
location = geolocator.geocode(address)
latitude = location.latitude
longitude = location.longitude
print("The geographical coordinates of Canberra are {},{}.".format(latitud
e, longitude))
```

```
The geographical coordinates of Canberra are -35.2975906,149.1012676.
```

In [95]:
```
act_map = folium.Map(location = [latitude, longitude], zoom_start = 6)
for lat, lon, city, state in zip(act_data['Latitude'], act_data['Longitude
'], act_data['City'], act_data['State']):
    label = '{}'.format(city, state)
    label = folium.Popup(label, parse_html = True)
    folium.CircleMarker(
       [lat, lon],
       radius = 5,
       color = 'blue',
       fill = True,
       fill_color = '#3186cc',
       fill_opacity = 0.7,
       parse_html = False).add_to(act_map)
act_map
```

Out[95]:



In [96]:
```
city_latitude = act_data.loc[0, "Latitude"]
city_longitude = act_data.loc[0, "Longitude"]
city_name = act_data.loc[0, "City"]
print("The Latitude and Longitude of {} are {}, {}.".format(city_name, cit
y_latitude, city_longitude))
```

```
The Latitude and Longitude of Canberra are -35.282001, 149.128998.
```

In [97]:
```
radius = 5000
limit = 100
query = "hospitals"
url2 = 'https://api.foursquare.com/v2/venues/explore?&client_id={}&client_
secret={}&ll={},{}&v={}&radius={}&limit={}&query={}'.format(CLIENT_ID, CLI
ENT_SECRET, city_latitude, city_longitude, VERSION, radius, limit, query)
url2
```

Out[97]:
```
'https://api.foursquare.com/v2/venues/explore?&client_id=L351DHR2FSTSDETU0
K5X3IZMC5XCTH5GJMOTYKAE0BEVPVO3&client_secret=U0NV33KA4DBR4L5ZDYKM2XXXOO4R
FH5DI1XNRFUGJDIF2MVR&ll=-35.282001,149.128998&v=20180605&radius=5000&limit
=100&query=hospitals'
```

In [307]:
```
results2 = requests.get(url2).json()
```

In [99]:
```
def get_category_type(row):
    try:
        categories_list2 = row['categories']
    except:
        categories_list2 = row['venue.categories']
    if len(categories_list2) == 0:
        return None
    else:
        return categories_list2[0]['name']
```

In [100]:
```
venues2 = results2['response']['groups'][0]['items']
nearby_venues2 = json_normalize(venues2)
filtered_columns = ['venue.name', 'venue.categories', 'venue.location.lati
tude', 'venue.location.longitude']
nearby_venues2 = nearby_venues2.loc[:, filtered_columns]
nearby_venues2['venue.categories'] = nearby_venues2.apply(get_category_typ
e, axis = 1)
nearby_venues2.columns = [col.split(".")[-1] for col in nearby_venues2.col
umns]
print("The size of the dataframe is:", nearby_venues2.shape)
nearby_venues2
```

```
The size of the dataframe is: (2, 4)
```

Out[100]:

	name	categories	latitude	longitude
0	Simpson Optometry	Hospital	NaN	NaN
1	Calvary Hospital	Hospital	NaN	NaN

In [241]:
```
# Queensland hospital data
```

In [101]:
```
qld_data = city_data[city_data['State'].str.contains('QLD')].reset_index(d
rop = True)
print("The shape of the dataframe:", qld_data.shape)
qld_data[0:10]
```

```
The shape of the dataframe: (7, 4)
```

Out[101]:

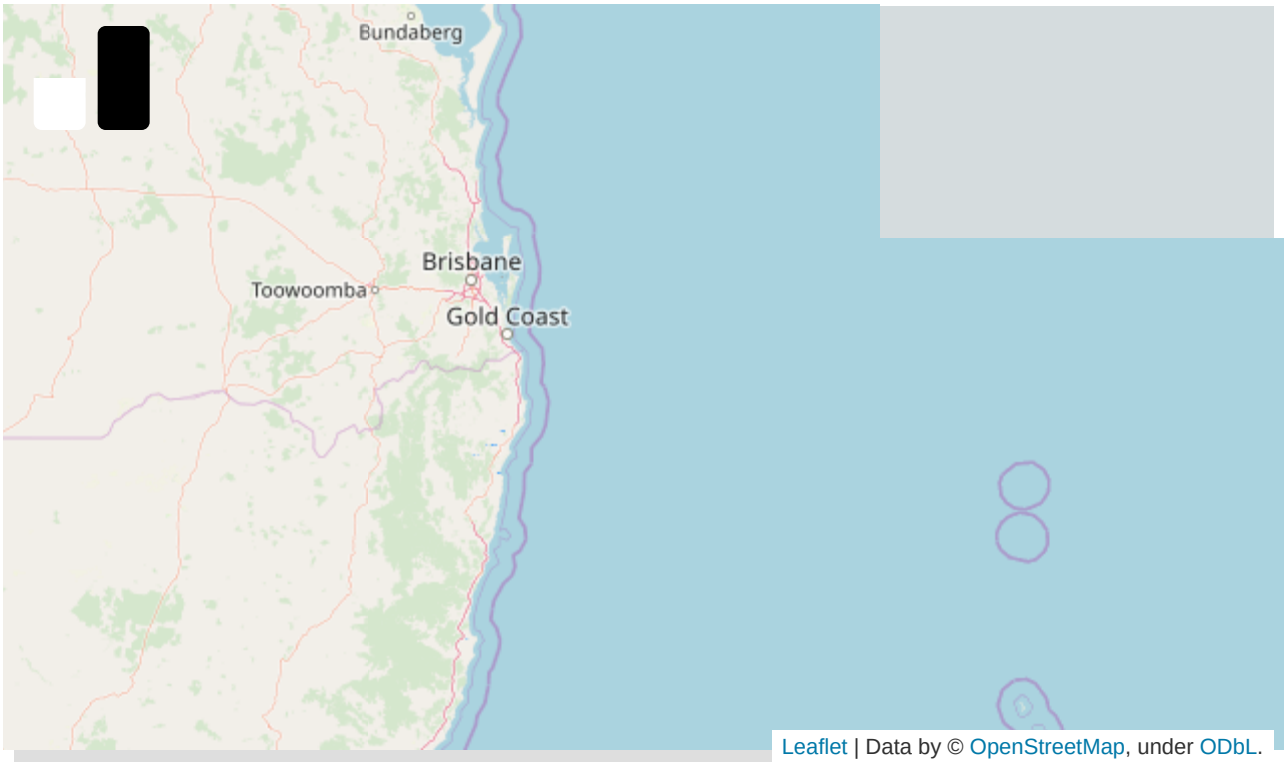
	City	Latitude	Longitude	State
0	Sunshine Coast	-26.650000	153.066666	QLD
1	Gold Coast	-28.016666	153.399994	QLD
2	Townsville City	-19.258965	146.816956	QLD
3	Cairns City	-16.925491	145.754120	QLD
4	Brisbane	-27.470125	153.021072	QLD
5	Gladstone	-23.843138	151.268356	QLD
6	Glenore Grove	-27.529953	152.407181	QLD

In [102]:
```
address = 'Brisbane'
geolocator = Nominatim(user_agent = 'my_application')
location = geolocator.geocode(address)
latitude = location.latitude
longitude = location.longitude
print("The geographical coordinates of Brisbane are {},{}.".format(latitud
e, longitude))
```

```
The geographical coordinates of Brisbane are -27.4689682,153.0234991.
```

In [103]:
```
nsw_map3 = folium.Map(location = [latitude, longitude], zoom_start = 6)
for lat, lon, city, state in zip(qld_data['Latitude'], qld_data['Longitude
'], qld_data['City'], qld_data['State']):
    label = '{}'.format(city, state)
    label = folium.Popup(label, parse_html = True)
    folium.CircleMarker(
       [lat, lon],
       radius = 5,
       color = 'blue',
       fill = True,
       fill_color = '#3186cc',
       fill_opacity = 0.7,
       parse_html = False).add_to(nsw_map3)
nsw_map3
```

Out[103]:



In [104]:
```
city_latitude = qld_data.loc[4, "Latitude"]
city_longitude = qld_data.loc[4, "Longitude"]
city_name = qld_data.loc[4, "City"]
print("The Latitude and Longitude of {} are {}, {}.".format(city_name, cit
y_latitude, city_longitude))
```

```
The Latitude and Longitude of Brisbane are -27.470125, 153.021072.
```

In [308]:
```
radius = 5000
limit = 100
query = "hospitals"
url3 = 'https://api.foursquare.com/v2/venues/explore?&client_id={}&client_
secret={}&ll={},{}&v={}&radius={}&limit={}&query={}'.format(CLIENT_ID, CLI
ENT_SECRET, city_latitude, city_longitude, VERSION, radius, limit, query)
```

In [309]:
```
results3 = requests.get(url3).json()
```

In [107]:
```
def get_category_type(row):
    try:
        categories_list3 = row['categories']
    except:
        categories_list3 = row['venue.categories']
    if len(categories_list3) == 0:
        return None
    else:
        return categories_list3[0]['name']
```

In [108]:
```
venues3 = results3['response']['groups'][0]['items']
nearby_venues3 = json_normalize(venues3)
filtered_columns = ['venue.name', 'venue.categories', 'venue.location.lati
tude', 'venue.location.longitude']
nearby_venues3 = nearby_venues3.loc[:, filtered_columns]
```

```
nearby_venues3['venue.categories'] = nearby_venues3.apply(get_category_type, axis = 1)
nearby_venues3.columns = [col.split(".")[-1] for col in nearby_venues3.columns]
print("The size of the dataframe is:", nearby_venues3.shape)
nearby_venues3
```

```
The size of the dataframe is: (29, 4)
```

Out[108]:

	name	categories	latitude	longitude
0	Brisbane Private Hospital	Hospital	NaN	NaN
1	Eyes on Edward	Hospital	NaN	NaN
2	St Andrew's Hospital	Hospital	NaN	NaN
3	Mater Adult Hospital	Hospital	NaN	NaN
4	St Vincent's Hospital	Hospital	NaN	NaN
5	Homlab Homeopathic Clinic	Hospital	NaN	NaN
6	Queensland Children's Hospital	Hospital	NaN	NaN
7	Lady Cilento Childrens Hospital (LCCH)	Hospital	NaN	NaN
8	Mater Medical Centre	Hospital	NaN	NaN
9	Mater Children's Private Hospital	Hospital	NaN	NaN
10	Salmon Building (ex Mater Children's Hospital)	Hospital	NaN	NaN
11	Mater Private Hospital	Hospital	NaN	NaN
12	Mater Hospital Specialist Clinics	Hospital	NaN	NaN
13	Mater Mothers' Hospital	Hospital	NaN	NaN
14	Mater Mothers Birthing Suites	Hospital	NaN	NaN
15	IMATIS PTY LTD	Hospital	NaN	NaN
16	The Wesley Hospital	Hospital	NaN	NaN
17	RBWH Block 7	Hospital	NaN	NaN
18	New Farm Clinic	Hospital	NaN	NaN
19	Royal Brisbane & Women's Hospital (RBWH)	Hospital	NaN	NaN
20	Hawken Eyes	Hospital	NaN	NaN
21	RBH Orthopaedic Clinic	Hospital	NaN	NaN
22	Joyce Tweddell Building	Hospital	NaN	NaN
23	RBWH - Oncology/Haematology	Hospital	NaN	NaN
24	Ned Hanlon Building	Hospital	NaN	NaN
25	RBWH- ICU	Hospital	NaN	NaN
26	Princess Alexandra Hospital	Hospital	NaN	NaN
27	QIMR Berghofer Central	Hospital	NaN	NaN

28	Ondol Oriental Medicine Clinic	Hospital	NaN	NaN

In [242]:
```
# South Australia Hospital Data

sa_data = city_data[city_data['State'].str.contains('SA')].reset_index(dro
p = True)
print("The shape of the dataframe:", sa_data.shape)
sa_data[0:10]
```

```
The shape of the dataframe: (3, 4)
```

Out[242]:

	City	Latitude	Longitude	State
0	Adelaide	-34.921230	138.599503	SA
1	North Adelaide	-34.906101	138.593903	SA
2	Mount Gambier	-37.824429	140.783783	SA

In [110]:
```
address = 'Adelaide'
geolocator = Nominatim(user_agent = 'my_application')
location = geolocator.geocode(address)
latitude = location.latitude
longitude = location.longitude
print("The geographical coordinates of Adelaide are {},{}.".format(latitud
e, longitude))
```

```
The geographical coordinates of Adelaide are -34.9281805,138.5999312.
```

In [111]:
```
city_latitude = sa_data.loc[0, "Latitude"]
city_longitude = sa_data.loc[0, "Longitude"]
city_name = sa_data.loc[0, "City"]
print("The Latitude and Longitude of {} are {}, {}.".format(city_name, cit
y_latitude, city_longitude))
```

```
The Latitude and Longitude of Adelaide are -34.92123, 138.599503.
```

In [112]:
```
radius = 5000
limit = 100
query = "hospitals"
url4 = 'https://api.foursquare.com/v2/venues/explore?&client_id={}&client_
secret={}&ll={},{}&v={}&radius={}&limit={}&query={}'.format(CLIENT_ID, CLI
ENT_SECRET, city_latitude, city_longitude, VERSION, radius, limit, query)
url4
```

Out[112]:
```
'https://api.foursquare.com/v2/venues/explore?&client_id=L351DHR2FSTSDETU0
K5X3IZMC5XCTH5GJMOTYKAE0BEVPVO3&client_secret=U0NV33KA4DBR4L5ZDYKM2XXXOO4R
FH5DI1XNRFUGJDIF2MVR&ll=-34.92123,138.599503&v=20180605&radius=5000&limit=
100&query=hospitals'
```

In [310]:
```
results4 = requests.get(url4).json()
```

In [114]:
```
def get_category_type(row):
    try:
        categories_list4 = row['categories']
```

```
    except:
        categories_list4 = row['venue.categories']
    if len(categories_list4) == 0:
        return None
    else:
        return categories_list4[0]['name']
```

In [115]:
```
venues4 = results4['response']['groups'][0]['items']
nearby_venues4 = json_normalize(venues4)
filtered_columns = ['venue.name', 'venue.categories', 'venue.location.lati
tude', 'venue.location.longitude']
nearby_venues4 = nearby_venues4.loc[:, filtered_columns]
nearby_venues4['venue.categories'] = nearby_venues4.apply(get_category_typ
e, axis = 1)
nearby_venues4.columns = [col.split(".")[-1] for col in nearby_venues4.col
umns]
print("The size of the dataframe is:", nearby_venues4.shape)
nearby_venues4
```

```
The size of the dataframe is: (23, 4)
```

Out[115]:

	name	categories	latitude	longitude
0	Shades Shop	Hospital	NaN	NaN
1	Opt Shop Optometry	Hospital	NaN	NaN
2	RAH Chest Clinic	Hospital	NaN	NaN
3	Adelaide Memorial Hospital	Hospital	NaN	NaN
4	CMAX	Hospital	NaN	NaN
5	Breastscreen SA	Hospital	NaN	NaN
6	New Royal Adelaide Hospital	Hospital	NaN	NaN
7	Womens and Childrens Hospital	Hospital	NaN	NaN
8	Calvary Wakefield Hospital	Hospital	NaN	NaN
9	Calvary North Adelaide Hospital	Hospital	NaN	NaN
10	Parkwynd Private Hospital	Hospital	NaN	NaN
11	Mary Potter Hospice	Hospital	NaN	NaN
12	Royal Adelaide Hospital	Hospital	NaN	NaN
13	St Andrew's Hospital	Hospital	NaN	NaN
14	The Adelaide Clinic	Hospital	NaN	NaN
15	Ashford Hospital	Hospital	NaN	NaN
16	SPORTSMED·SA Clinic and Hospital	Hospital	NaN	NaN
17	Calvary Rehabilitation Hospital	Hospital	NaN	NaN
18	Glenside Campus Hospital	Hospital	NaN	NaN
19	Helen Mayo House	Hospital	NaN	NaN

20	Glenside Health Services	Hospital	NaN	NaN
21	Burnside War Memorial Hospital	Hospital	NaN	NaN
22	Podiatry and Orthotics by Adelaide Foot and Ankle	Hospital	NaN	NaN

In [243]:
```
# Western Australia Hospital information

wa_data = city_data[city_data['State'].str.contains('WA')].reset_index(dro
p = True)
print("The shape of the dataframe:", wa_data.shape)
wa_data[0:10]
```

```
The shape of the dataframe: (1, 4)
```

Out[243]:

	City	Latitude	Longitude	State
0	Perth	-31.953512	115.857048	WA

In [117]:
```
address = 'Perth'
geolocator = Nominatim(user_agent = 'my_application')
location = geolocator.geocode(address)
latitude = location.latitude
longitude = location.longitude
print("The geographical coordinates of Perth are {},{}.".format(latitude,
longitude))
```

```
The geographical coordinates of Perth are -31.9527121,115.8604796.
```

In [118]:
```
city_latitude = wa_data.loc[0, "Latitude"]
city_longitude = wa_data.loc[0, "Longitude"]
city_name = wa_data.loc[0, "City"]
print("The Latitude and Longitude of {} are {}, {}.".format(city_name, cit
y_latitude, city_longitude))
```

```
The Latitude and Longitude of Perth are -31.953512, 115.85704799999999.
```

In [119]:
```
radius = 5000
limit = 100
query = "hospitals"
url5 = 'https://api.foursquare.com/v2/venues/explore?&client_id={}&client_
secret={}&ll={},{}&v={}&radius={}&limit={}&query={}'.format(CLIENT_ID, CLI
ENT_SECRET, city_latitude, city_longitude, VERSION, radius, limit, query)
url5
```

Out[119]:
```
'https://api.foursquare.com/v2/venues/explore?&client_id=L351DHR2FSTSDETU0
K5X3IZMC5XCTH5GJMOTYKAE0BEVPVO3&client_secret=U0NV33KA4DBR4L5ZDYKM2XXXOO4R
FH5DI1XNRFUGJDIF2MVR&ll=-31.953512,115.85704799999999&v=20180605&radius=50
00&limit=100&query=hospitals'
```

In [120]:
```
results5 = requests.get(url5).json()
```

In [121]:
```
def get_category_type(row):
    try:
        categories_list5 = row['categories']
```

```
    except:
        categories_list5 = row['venue.categories']
    if len(categories_list5) == 0:
        return None
    else:
        return categories_list5[0]['name']
```

In [122]:
```
venues5 = results5['response']['groups'][0]['items']
nearby_venues5 = json_normalize(venues5)
filtered_columns = ['venue.name', 'venue.categories', 'venue.location.lati
tude', 'venue.location.longitude']
nearby_venues5 = nearby_venues5.loc[:, filtered_columns]
nearby_venues5['venue.categories'] = nearby_venues5.apply(get_category_typ
e, axis = 1)
nearby_venues5.columns = [col.split(".")[-1] for col in nearby_venues5.col
umns]
print("The size of the dataframe is:", nearby_venues5.shape)
nearby_venues5
```

```
The size of the dataframe is: (19, 4)
```

Out[122]:

	name	categories	latitude	longitude
0	Royal Perth Hospital	Hospital	NaN	NaN
1	Royal Perth Hospital Radiology	Hospital	NaN	NaN
2	Mount Hospital	Hospital	NaN	NaN
3	Princess Margaret Hospital	Hospital	NaN	NaN
4	Hollywood Private Hospital	Hospital	NaN	NaN
5	St John of God Mt Lawley Hospital	Hospital	NaN	NaN
6	The Atomic Age Eyewear Company	Hospital	NaN	NaN
7	St John of God Hospital	Hospital	NaN	NaN
8	Genea Hollywood Fertility	Hospital	NaN	NaN
9	Subiaco Veterinary Hospital	Hospital	NaN	NaN
10	King Edward Memorial Hospital	Hospital	NaN	NaN
11	Australian Red Cross Blood Service	Medical Center	NaN	NaN
12	Sir Charles Gairdner Hospital	Hospital	NaN	NaN
13	Perth Children's Hospital	Hospital	NaN	NaN
14	SCGH Cancer centre	Hospital	NaN	NaN
15	Pathwest Pathology	Hospital	NaN	NaN
16	Queen Elizabeth II Medical Centre	Hospital	NaN	NaN
17	Rodin Clinic - Plastic Surgery Perth	Hospital	NaN	NaN
18	South Perth Hospital	Hospital	NaN	NaN

In [123]:
```
tas_data = city_data[city_data['State'].str.contains('TAS')].reset_index(d
```

```
rop = True)
print("The shape of the dataframe:", tas_data.shape)
tas_data[0:10]
```

```
The shape of the dataframe: (1, 4)
```

Out[123]:

	City	Latitude	Longitude	State
0	Launceston	-41.429825	147.157135	TAS

In [124]:
```
address = 'Launceston'
geolocator = Nominatim(user_agent = 'my_application')
location = geolocator.geocode(address)
latitude = location.latitude
longitude = location.longitude
print("The geographical coordinates of Launceston are {},{}.".format(latit
ude, longitude))
```

```
The geographical coordinates of Launceston are -41.4340813,147.1373496.
```

In [125]:
```
city_latitude = tas_data.loc[0, "Latitude"]
city_longitude = tas_data.loc[0, "Longitude"]
city_name = tas_data.loc[0, "City"]
print("The Latitude and Longitude of {} are {}, {}.".format(city_name, cit
y_latitude, city_longitude))
```

```
The Latitude and Longitude of Launceston are -41.429825, 147.157135.
```

In [126]:
```
radius = 5000
limit = 100
query = "hospitals"
url6 = 'https://api.foursquare.com/v2/venues/explore?&client_id={}&client_
secret={}&ll={},{}&v={}&radius={}&limit={}&query={}'.format(CLIENT_ID, CLI
ENT_SECRET, city_latitude, city_longitude, VERSION, radius, limit, query)
url6
```

Out[126]:
```
'https://api.foursquare.com/v2/venues/explore?&client_id=L351DHR2FSTSDETU0
K5X3IZMC5XCTH5GJMOTYKAE0BEVPVO3&client_secret=U0NV33KA4DBR4L5ZDYKM2XXXOO4R
FH5DI1XNRFUGJDIF2MVR&ll=-41.429825,147.157135&v=20180605&radius=5000&limit
=100&query=hospitals'
```

In [127]:
```
results6 = requests.get(url6).json()
```

In [128]:
```
def get_category_type(row):
    try:
        categories_list6 = row['categories']
    except:
        categories_list6 = row['venue.categories']
    if len(categories_list6) == 0:
        return None
    else:
        return categories_list6[0]['name']
```

In [129]:
```
venues6 = results6['response']['groups'][0]['items']
nearby_venues6 = json_normalize(venues6)
```

```
filtered_columns = ['venue.name', 'venue.categories', 'venue.location.lati
tude', 'venue.location.longitude']
nearby_venues6 = nearby_venues6.loc[:, filtered_columns]
nearby_venues6['venue.categories'] = nearby_venues6.apply(get_category_typ
e, axis = 1)
nearby_venues6.columns = [col.split(".")[-1] for col in nearby_venues6.col
umns]
print("The size of the dataframe is:", nearby_venues6.shape)
nearby_venues6
```

```
The size of the dataframe is: (4, 4)
```

Out[129]:

	name	categories	latitude	longitude
0	Calvary Hospital	Hospital	NaN	NaN
1	Tremaur Medical Centre	Hospital	NaN	NaN
2	St Vincents Hospital	Hospital	NaN	NaN
3	Launceston General Hospital	Hospital	NaN	NaN

In [244]:
```
# Victoria hospital data

vic_data = city_data[city_data['State'].str.contains('VIC')].reset_index(d
rop = True)
print("The shape of the dataframe:", vic_data.shape)
vic_data[0:10]
```

```
The shape of the dataframe: (4, 4)
```

Out[244]:

	City	Latitude	Longitude	State
0	Melbourne	-37.840935	144.946457	VIC
1	Mildura	-34.206841	142.136490	VIC
2	Ziyou Today	-37.649967	144.880600	VIC
3	Bendigo	-36.757786	144.278702	VIC

In [131]:
```
address = 'Melbourne'
geolocator = Nominatim(user_agent = 'my_application')
location = geolocator.geocode(address)
latitude = location.latitude
longitude = location.longitude
print("The geographical coordinates of Melbourne are {},{}.".format(latitu
de, longitude))
```

```
The geographical coordinates of Melbourne are -37.8142176,144.9631608.
```

In [132]:
```
city_latitude = vic_data.loc[0, "Latitude"]
city_longitude = vic_data.loc[0, "Longitude"]
city_name = vic_data.loc[0, "City"]
print("The Latitude and Longitude of {} are {}, {}.".format(city_name, cit
y_latitude, city_longitude))
```

```
The Latitude and Longitude of Melbourne are -37.840934999999995, 144.94645
```

7.

In [133]:
```
radius = 5000
limit = 100
query = "Hospital"
url7 = 'https://api.foursquare.com/v2/venues/explore?&client_id={}&client_secret={}&ll={},{}&v={}&radius={}&limit={}&query={}'.format(CLIENT_ID, CLIENT_SECRET, city_latitude, city_longitude, VERSION, radius, limit, query)
url7
```

Out[133]:
```
'https://api.foursquare.com/v2/venues/explore?&client_id=L351DHR2FSTSDETU0K5X3IZMC5XCTH5GJMOTYKAE0BEVPVO3&client_secret=U0NV33KA4DBR4L5ZDYKM2XXXOO4RFH5DI1XNRFUGJDIF2MVR&ll=-37.840934999999995,144.946457&v=20180605&radius=5000&limit=100&query=Hospital'
```

In [311]:
```
results7 = requests.get(url7).json()
```

In [135]:
```
def get_category_type(row):
    try:
        categories_list7 = row['categories']
    except:
        categories_list7 = row['venue.categories']
    if len(categories_list7) == 0:
        return None
    else:
        return categories_list7[0]['name']
```

In [136]:
```
venues7 = results7['response']['groups'][0]['items']
nearby_venues7 = json_normalize(venues7)
filtered_columns = ['venue.name', 'venue.categories', 'venue.location.latitude', 'venue.location.longitude']
nearby_venues7 = nearby_venues7.loc[:, filtered_columns]
nearby_venues7['venue.categories'] = nearby_venues7.apply(get_category_type, axis = 1)
nearby_venues7.columns = [col.split(".")[-1] for col in nearby_venues7.columns]
print("The size of the dataframe is:", nearby_venues7.shape)
nearby_venues7
```

```
The size of the dataframe is: (35, 4)
```

Out[136]:

	name	categories	latitude	longitude
0	Peter MacCallum Cancer Centre	Hospital	NaN	NaN
1	Alfred Hospital Outpatient Clinics	Hospital	NaN	NaN
2	The Alfred Hospital	Hospital	NaN	NaN
3	MyClinic Southbank	Hospital	NaN	NaN
4	Centre For Clinical Studies	Hospital	NaN	NaN
5	MyClinic St Kilda	Hospital	NaN	NaN
6	The Royal Melbourne Hospital	Hospital	NaN	NaN
7	MyClinic QV	Hospital	NaN	NaN

8	Epworth Richmond	Hospital	NaN	NaN
9	St. Vincents & Mercy Private	Hospital	NaN	NaN
10	The Royal Victorian Eye and Ear Hospital	Hospital	NaN	NaN
11	The Alfred Centre	Medical School	NaN	NaN
12	Frances Perry House	Hospital	NaN	NaN
13	St Vincent's Hospital	Hospital	NaN	NaN
14	MyClinic Bourke Street Mall	Hospital	NaN	NaN
15	Victoria Parade Surgery Centre	Hospital	NaN	NaN
16	Epworth Freemasons Hospital	Hospital	NaN	NaN
17	MyClinic South Yarra	Hospital	NaN	NaN
18	St Vincents - Healy Wing	Hospital	NaN	NaN
19	St Vincent's Daly Wing	Hospital	NaN	NaN
20	Lansdowne Eye Clinic	Hospital	NaN	NaN
21	Royal Women's Hospital	Hospital	NaN	NaN
22	Cliveden Hill Private Hospital	Hospital	NaN	NaN
23	Mercy / St Vincent's Private Hospital	Hospital	NaN	NaN
24	Peter McCallum Cancer Centre	Hospital	NaN	NaN
25	Womens Ultrasound Melbourne (WUME)	Hospital	NaN	NaN
26	Epworth Freemasons Hospital	Hospital	NaN	NaN
27	The Avenue Hospital	Hospital	NaN	NaN
28	The Royal Dental Hospital of Melbourne	Hospital	NaN	NaN
29	St Vincent's Private Radiology (CMMI)	Hospital	NaN	NaN
30	St Vincent's Hospital - Emergency Department	Hospital	NaN	NaN
31	Melbourne Private Hospital	Hospital	NaN	NaN
32	Melbourne Dental School	Hospital	NaN	NaN
33	370 St Kilda Rd	Office	NaN	NaN
34	Bupa Head Office	Office	NaN	NaN

In [245]:
```
# Northern Territory Hospital Data

darwin_data = city_data[city_data['State'].str.contains('NT')].reset_index
(drop = True)
print("The shape of the dataframe:", darwin_data.shape)
darwin_data[0:10]
```

```
The shape of the dataframe: (1, 4)
```

Out[245]:

	City	Latitude	Longitude	State

0	Darwin	-12.462827	130.841782	NT

In [138]:
```
address = 'Darwin'
geolocator = Nominatim(user_agent = 'my_application')
location = geolocator.geocode(address)
latitude = location.latitude
longitude = location.longitude
print("The geographical coordinates of Darwin are {},{}.".format(latitude,
 longitude))
```

```
The geographical coordinates of Darwin are -12.46044,130.8410469.
```

In [139]:
```
city_latitude = darwin_data.loc[0, "Latitude"]
city_longitude = darwin_data.loc[0, "Longitude"]
city_name = darwin_data.loc[0, "City"]
print("The Latitude and Longitude of {} are {}, {}.".format(city_name, cit
y_latitude, city_longitude))
```

```
The Latitude and Longitude of Darwin are -12.462827, 130.841782.
```

In [146]:
```
radius = 15000
limit = 100
query = "hospitals"
url8 = 'https://api.foursquare.com/v2/venues/explore?&client_id={}&client_
secret={}&ll={},{}&v={}&radius={}&limit={}&query={}'.format(CLIENT_ID, CLI
ENT_SECRET, city_latitude, city_longitude, VERSION, radius, limit, query)
url8
```

Out[146]:
```
'https://api.foursquare.com/v2/venues/explore?&client_id=L351DHR2FSTSDETU0
K5X3IZMC5XCTH5GJMOTYKAE0BEVPVO3&client_secret=U0NV33KA4DBR4L5ZDYKM2XXXOO4R
FH5DI1XNRFUGJDIF2MVR&ll=-12.462827,130.841782&v=20180605&radius=15000&limi
t=100&query=hospitals'
```

In [147]:
```
results8 = requests.get(url8).json()
```

In [148]:
```
def get_category_type(row):
    try:
        categories_list8 = row['categories']
    except:
        categories_list8 = row['venue.categories']
    if len(categories_list8) == 0:
        return None
    else:
        return categories_list8[0]['name']
```

In [149]:
```
venues8 = results8['response']['groups'][0]['items']
nearby_venues8 = json_normalize(venues8)
filtered_columns = ['venue.name', 'venue.categories', 'venue.location.lati
tude', 'venue.location.longitude']
nearby_venues8 = nearby_venues8.loc[:, filtered_columns]
nearby_venues8['venue.categories'] = nearby_venues8.apply(get_category_typ
e, axis = 1)
nearby_venues8.columns = [col.split(".")[-1] for col in nearby_venues8.col
umns]
print("The size of the dataframe is:", nearby_venues8.shape)
```

```
nearby_venues8
```

```
The size of the dataframe is: (4, 4)
```

Out[149]:

	name	categories	latitude	longitude
0	Integrated Health Solutions – Margaret Rolling...	Hospital	NaN	NaN
1	MB Naturopath	Hospital	NaN	NaN
2	Royal Darwin Hospital	Hospital	NaN	NaN
3	Darwin Private Hospital	Hospital	NaN	NaN

The use of DBSCAN for clustering cities

In [247]:
```
from sklearn.cluster import DBSCAN
from geopy.distance import great_circle
from sklearn import metrics
from sklearn.datasets.samples_generator import make_blobs
from sklearn.preprocessing import StandardScaler
coords = city_datagp.as_matrix(columns = ['Latitude', 'Longitude'])
```

```
C:\Users\c3273214\AppData\Local\Continuum\anaconda3\lib\site-packages\ipyk
ernel_launcher.py:6: FutureWarning:

Method .as_matrix will be removed in a future version. Use .values instead
.
```

In [248]:
```
kms_per_radian = 6371.0088
epsilon = 1.5/kms_per_radian
db = DBSCAN(eps = epsilon, min_samples = 1, algorithm = 'ball_tree', metri
c = 'haversine').fit(np.radians(coords))
cluster_labels = db.labels_
num_clusters = len(set(cluster_labels))
clusters = pd.Series([coords[cluster_labels == n] for n in range(num_clust
ers)])
print('Number of clusters: {}'.format(num_clusters))
```

```
Number of clusters: 28
```

Data for cities plus other urban areas

In [303]:
```
# Reading the csv file for the other cities and other urban areas

import pandas as pd
import io
file = open('C:/Users/c3273214/Documents/Auspop2.csv', encoding = 'latin-1
')
X = pd.read_csv(file)
X.head()
```

Out[303]:

	Rank	Area	Territory	June 2018[2]	2011_cens	Unnamed: 5	Unnamed: 6
0	1	Sydney	New South Wales	5,230,330	4,391,674	NaN	NaN
1	2	Melbourne	Victoria	4,936,349	3,999,982	NaN	NaN
2	3	Brisbane	Queensland	2,462,637	2,065,996	NaN	NaN
3	4	Perth	Western Australia	2,059,484	1,728,867	NaN	NaN
4	5	Adelaide	South Australia	1,345,777	1,262,940	NaN	NaN

In [304]:
```
# Grouped data for cities and other urban areas

Y = X.groupby(['Territory', 'Area'], as_index = False).agg(lambda x: ",".j
oin(x))
Y.head()
```

Out[304]:

	Territory	Area	2011_cens	June 2018[2]
0	Australian Capital Territory/New South Wales	Canberra_Queanbeyan	391,645	457,563
1	New South Wales	Armidale	22,464	24,504
2	New South Wales	Ballina	23,509	26,381
3	New South Wales	Batemans Bay	15,733	16,485
4	New South Wales	Bathurst	32,479	36,801

In [286]:
```
Y.describe().all
```

Out[286]:
```
<bound method DataFrame.all of                Territory      Area 2011_cens
 June 2018[2]
count                  96        96         96            96
unique                 11        96         96            95
top     New South Wales  Portland    14,043       26,381
freq                   33         1          1             2>
```

In [278]:
```
Y['June 2018[2]'].describe()
```

Out[278]:
```
count         96
unique        95
top       26,381
freq           2
Name: June 2018[2], dtype: object
```

In [295]:
```
Y.dtypes
```

Out[295]:
```
2011_cens       object
June 2018[2]    object
dtype: object
```

In [ ]:
```

```