Data Science Assignment 1

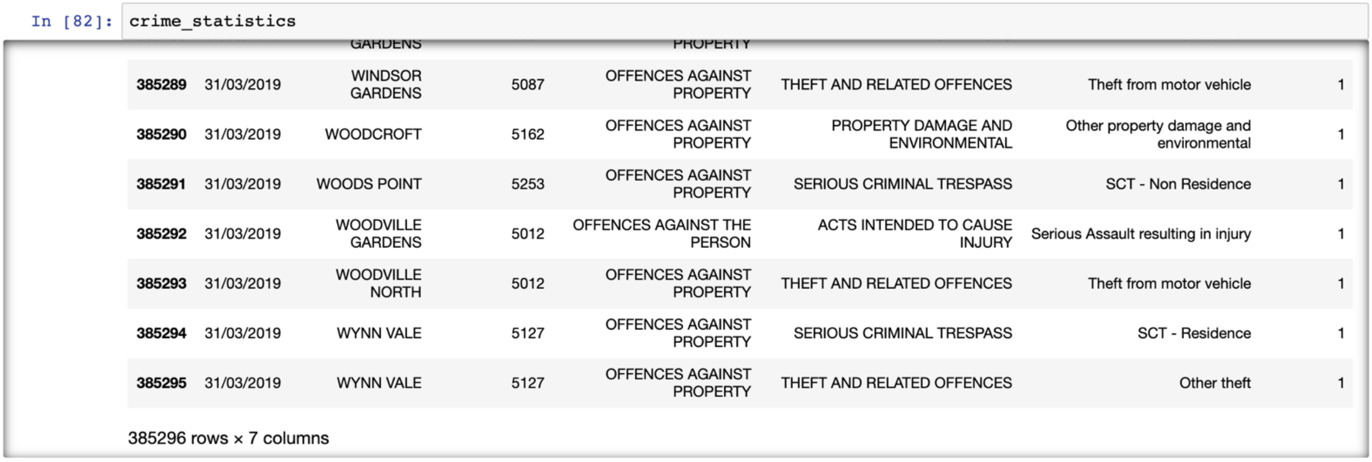
Name: Germaine Pok Yi Min   Student ID: 2979782

Task A: Data Exploration and Auditing

In this task, you are required to explore the dataset and do some data auditing on the crime statistics dataset. Have a look at the CSV file (Crime_Statistics_SA_2014_2019.csv) and then answer a series of questions about the data using Python.

A1. Dataset size

How many rows and columns exist in this dataset?

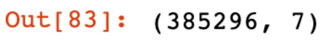


```
In [82]: crime_statistics
```

		GARDENS		PROPERTY			
385289	31/03/2019	WINDSOR GARDENS	5087	OFFENCES AGAINST PROPERTY	THEFT AND RELATED OFFENCES	Theft from motor vehicle	1
385290	31/03/2019	WOODCROFT	5162	OFFENCES AGAINST PROPERTY	PROPERTY DAMAGE AND ENVIRONMENTAL	Other property damage and environmental	1
385291	31/03/2019	WOODS POINT	5253	OFFENCES AGAINST PROPERTY	SERIOUS CRIMINAL TRESPASS	SCT - Non Residence	1
385292	31/03/2019	WOODVILLE GARDENS	5012	OFFENCES AGAINST THE PERSON	ACTS INTENDED TO CAUSE INJURY	Serious Assault resulting in injury	1
385293	31/03/2019	WOODVILLE NORTH	5012	OFFENCES AGAINST PROPERTY	THEFT AND RELATED OFFENCES	Theft from motor vehicle	1
385294	31/03/2019	WYNN VALE	5127	OFFENCES AGAINST PROPERTY	SERIOUS CRIMINAL TRESPASS	SCT - Residence	1
385295	31/03/2019	WYNN VALE	5127	OFFENCES AGAINST PROPERTY	THEFT AND RELATED OFFENCES	Other theft	1

385296 rows × 7 columns

crime_statistics.shape



```
Out[83]: (385296, 7)
```

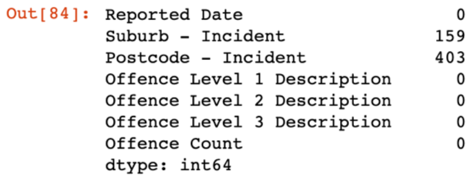
According to the code above, crime_statistics is the name of the data frame. By using crime_statistics.shape, we get the data for the rows and column of the data set.

There are 385296 rows and 7 columns in this data set

A2. Null values in the dataset

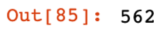
Are there any null values in this dataset?

crime_statistics.isnull().sum()



```
Out[84]: Reported Date                   0
         Suburb - Incident             159
         Postcode - Incident           403
         Offence Level 1 Description     0
         Offence Level 2 Description     0
         Offence Level 3 Description     0
         Offence Count                   0
         dtype: int64
```

crime_statistics.isnull().sum().sum()



```
Out[85]: 562
```

Through the .isnull() function, null values are detected. It is then summed up by the .sum() function twice to acquire the value. Without the .sum() function, a table of the dataset would be displayed instead, just displaying whether the value of the table is a null value or not. If it is not a null value, it will display 'False'.

There are 562 false values in this dataset

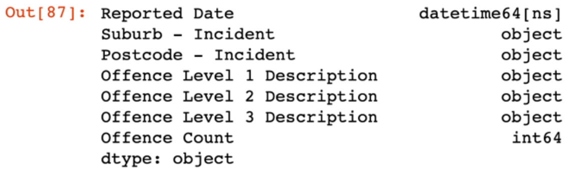
A3. Data Types

What are the min and max for column 'Reported Date '? Does this column have the correct data type? If no, convert it to an appropriate data type.

crime_statistics['Reported Date'] = pd.to_datetime(crime_statistics['Reported Date'])

The data type of 'Reported Date' is converted from object to datetime data type through the line of code above.

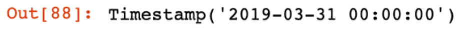
crime_statistics.dtypes



```
Out[87]: Reported Date                  datetime64[ns]
         Suburb - Incident                      object
         Postcode - Incident                    object
         Offence Level 1 Description            object
         Offence Level 2 Description            object
         Offence Level 3 Description            object
         Offence Count                           int64
         dtype: object
```

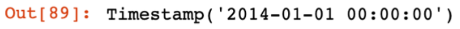
After the data is converted, the min and max is obtained using the .max() and .min() function.

crime_statistics["Reported Date"].max()



```
Out[88]: Timestamp('2019-03-31 00:00:00')
```

crime_statistics["Reported Date"].min()



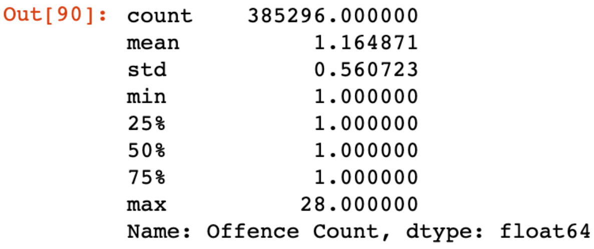
```
Out[89]: Timestamp('2014-01-01 00:00:00')
```

The earliest 'Reported Date' is 1st of January 2014 and the latest 'Reported Date is 31st March 2019.

A4. Descriptive statistics

Calculate the statistics for the "Offence Count" column (Find the count, mean, standard deviation, minimum and maximum).

crime_statistics['Offence Count'].describe()



```
Out[90]: count    385296.000000
         mean          1.164871
         std           0.560723
         min           1.000000
         25%           1.000000
         50%           1.000000
         75%           1.000000
         max          28.000000
         Name: Offence Count, dtype: float64
```

As shown by the data above which was obtained using the function .describe(), the statistics are :

Count : 385269 max : 28.0

Mean : 1.163871 min : 1.0

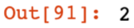
standard deviation : 0.560723

A5. Exploring Offence Level 1 Description

Now look at the Offence Level 1 Description column and answer the following questions

1. **How many unique values does "Offence Level 1 Description" column take?**

   crime_statistics['Offence Level 1 Description'].nunique()

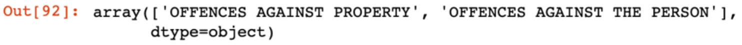
   ```
   Out[91]: 2
   ```

   The unique values are obtained using the .nunique() function.

   “Offence Level 1 Description” column take 2 unique values.

2. **Display the unique values of level 1 offences.**

   crime_statistics['Offence Level 1 Description'].unique()

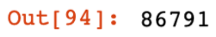
   

   ```
   Out[92]: array(['OFFENCES AGAINST PROPERTY', 'OFFENCES AGAINST THE PERSON'],
                 dtype=object)
   ```

   The unique values of level 1 offences are “OFFENCES AGAINST THE PROPERTY” and “OFFENCES AGAINST THE PERSON”. It is obtained using the .unique() function.

3. **How many records do contain "offences against the person"?**

   OATP = (crime_statistics['Offence Level 1 Description'].values == 'OFFENCES AGAINST THE PERSON').sum()



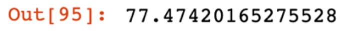
```
Out[94]: 86791
```

   OATP stands for Offence Against the Person. The code means that if the value within the ‘Offence Level 1 Description’ is “OFFENCES AGAINST THE PERSON”, sum the numbers of ‘True’ values.

   There are 86791 records that contains “OFFENCES AGAINST THE PERSON”.

4. **What percentage of the records are "offences against the property"?**

   *((crime_statistics['Offence Level 1 Description'].values == 'OFFENCES AGAINST PROPERTY').sum()/crime_statistics['Offence Level 1 Description'].count())*100*



```
Out[95]: 77.47420165275528
```

   The number of records that contains “OFFENCES AGAINST THE PROPERTY” is divided by total number of values in the “Offence Level 1 Description” and is then multiplied by 100 to obtain the percentage of records of offence against the property.

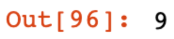
   The percentage of records that is offences against the property is 77.47%

A6. Exploring Offence Level 2 Description

Now look at the Offence Level 2 Description column and answer the following questions

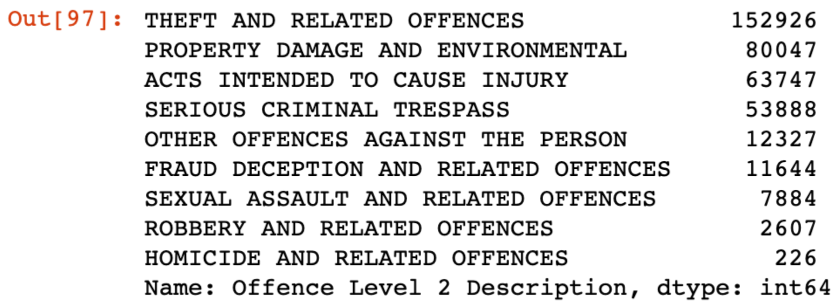
1. **How many unique values does "Offence Level 2 Description" column take? Display the unique values of level 2 offences together with their counts (i.e., how many times they have been repeated).**

   crime_statistics['Offence Level 2 Description'].nunique()



```
Out[96]: 9
```

   crime_statistics['Offence Level 2 Description'].value_counts()



```
Out[97]: THEFT AND RELATED OFFENCES                 152926
         PROPERTY DAMAGE AND ENVIRONMENTAL           80047
         ACTS INTENDED TO CAUSE INJURY               63747
         SERIOUS CRIMINAL TRESPASS                   53888
         OTHER OFFENCES AGAINST THE PERSON           12327
         FRAUD DECEPTION AND RELATED OFFENCES        11644
         SEXUAL ASSAULT AND RELATED OFFENCES          7884
         ROBBERY AND RELATED OFFENCES                 2607
         HOMICIDE AND RELATED OFFENCES                 226
         Name: Offence Level 2 Description, dtype: int64
```

“Offence Level 2 Description” column takes 9 unique values which is displayed by the .nunique() function.

The counts of the unique value is obtained by using the .value_counts() function.

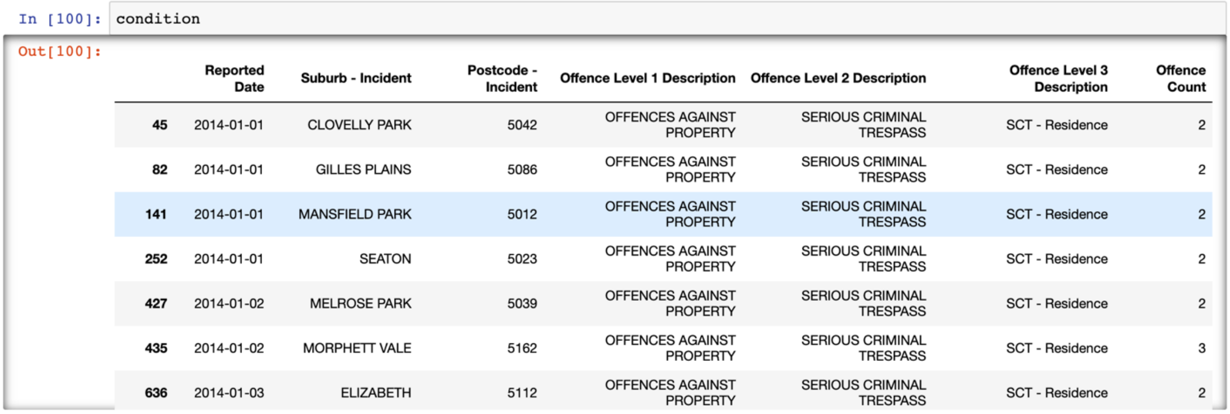
There are 152926 theft and related offences, 80047 property damage and environmental, 63747 acts intended to cause injury, 53888 serious criminal trespass, 12327 other offences against the person, 11644 fraud and deception and related offences, 7884 sexual assault and related offences, 2607 robbery and related offense and 266 homicide and related offences,

2. How many serious criminal trespasses have occurred with more than 1 offence count?

sct = crime_statistics[crime_statistics['Offence Level 2 Description'] == 'SERIOUS CRIMINA L TRESPASS']

sct stands for serious criminal trespass and it displays a dataset where the column “Offence Level 2 Descriptions” only has the value of ‘SERIOUS CRIMINAL TRESPASS’.

condition = sct[(sct['Offence Count'] > 1)]

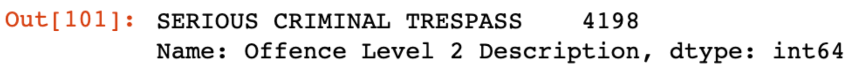


```
In [100]: condition
Out[100]:
```

	Reported Date	Suburb - Incident	Postcode - Incident	Offence Level 1 Description	Offence Level 2 Description	Offence Level 3 Description	Offence Count
45	2014-01-01	CLOVELLY PARK	5042	OFFENCES AGAINST PROPERTY	SERIOUS CRIMINAL TRESPASS	SCT - Residence	2
82	2014-01-01	GILLES PLAINS	5086	OFFENCES AGAINST PROPERTY	SERIOUS CRIMINAL TRESPASS	SCT - Residence	2
141	2014-01-01	MANSFIELD PARK	5012	OFFENCES AGAINST PROPERTY	SERIOUS CRIMINAL TRESPASS	SCT - Residence	2
252	2014-01-01	SEATON	5023	OFFENCES AGAINST PROPERTY	SERIOUS CRIMINAL TRESPASS	SCT - Residence	2
427	2014-01-02	MELROSE PARK	5039	OFFENCES AGAINST PROPERTY	SERIOUS CRIMINAL TRESPASS	SCT - Residence	2
435	2014-01-02	MORPHETT VALE	5162	OFFENCES AGAINST PROPERTY	SERIOUS CRIMINAL TRESPASS	SCT - Residence	3
636	2014-01-03	ELIZABETH	5112	OFFENCES AGAINST PROPERTY	SERIOUS CRIMINAL TRESPASS	SCT - Residence	2

A new dataset is created with the condition where the values in the ‘Offence Count’ column Is more than 1.

condition['Offence Level 2 Description'].value_counts()



```
Out[101]: SERIOUS CRIMINAL TRESPASS    4198
          Name: Offence Level 2 Description, dtype: int64
```

The number of occurrence of serious criminal trespass is obtained by using the function .value_counts()

There are 4198 occurrence of serious criminal trespass.

Task B: Investigating Offence Count in different suburbs and different years

In the task, you are required to visualise the relationship between the number of crimes in different suburbs and different years and exploring the relationship. Note: higher marks will be given to reports containing graphs with appropriately labelled axes, title and legend.

B1. Investigating the number of crimes per year

Find the number of crimes per year. Plot the graph and explain your understanding of the graph. Hint: you can extract 'year' from column "reported date" using method .dt and create a new column for the year in your dataframe as follows:

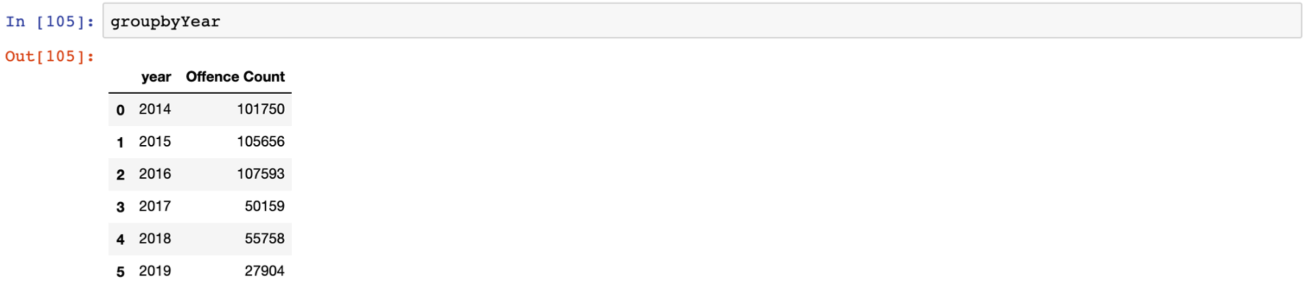
>>> your_dataframe['year']=your_dataframe['Reported Date'].dt.year

crime_statistics['year']=crime_statistics['Reported Date'].dt.year

A new column called year is added into the crime_statistics dataset.

groupbyYear = crime_statistics.groupby('year')['Offence Count'].sum()

groupbyYear = groupbyYear.reset_index()



```
In [105]: groupbyYear
Out[105]:
```

	year	Offence Count
0	2014	101750
1	2015	105656
2	2016	107593
3	2017	50159
4	2018	55758
5	2019	27904

A new data set containing the year and the number of offence count that occurred within the year is created through groupby and it is named groupbyYear. The dataset displays the number of offence counts that occurs every year.

groupbyYear.year = groupbyYear.year.astype(str)

The data type of year is converted into a string or not jupyter will include the year values as a value in the bar chart

Code for bar chart:

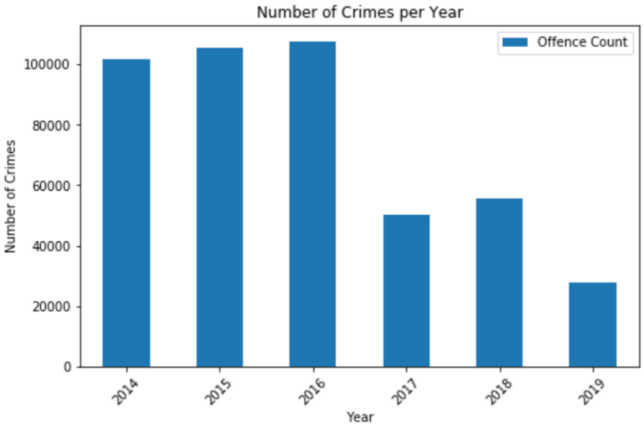
ax=groupbyYear.plot.bar(figsize=(8,5))

ax.set_xticklabels(groupbyYear['year'],rotation=45)

plt.xlabel('Year')

plt.ylabel('Number of Crimes')

plt.title('Number of Crimes per Year')



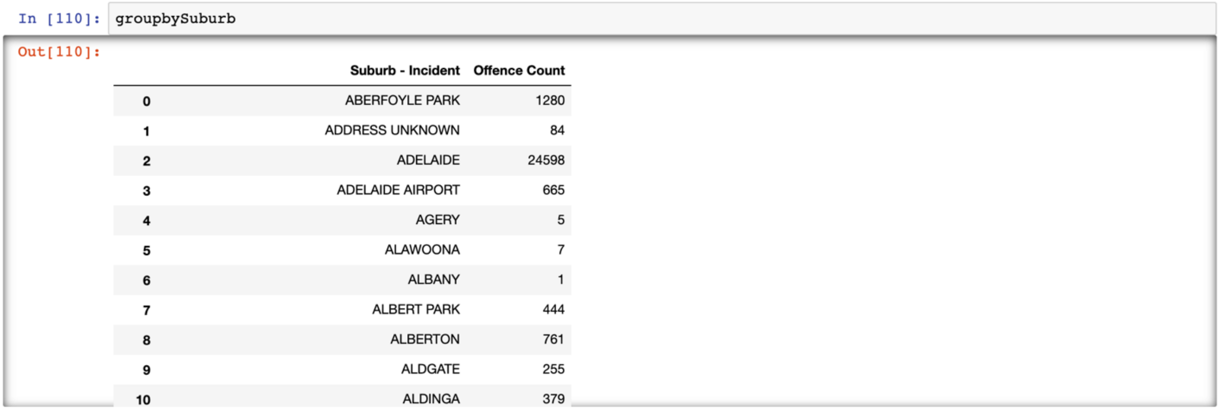
According to the bar chart, the most number of crimes was committed in 2016 while the least number of crimes was committed in 2019. Initially, from 2014 to 2016, the number of crimes increased significantly. However, there is a drastic drop of the number of crimes in 2017 compared to 2016. It increase significantly in 2018 and dropped in 2019, where crimes occurred the least.

B2. Investigating the total number of crimes in different suburbs

1. **Compute the total number of crimes in each suburb and plot a histogram of the total number of crimes in different suburbs**

groupbySuburb = crime_statistics.groupby('Suburb - Incident')['Offence Count'].sum()

groupbySuburb = groupbySuburb.reset_index()



```
In [110]: groupbySuburb
Out[110]:
```

	Suburb - Incident	Offence Count
0	ABERFOYLE PARK	1280
1	ADDRESS UNKNOWN	84
2	ADELAIDE	24598
3	ADELAIDE AIRPORT	665
4	AGERY	5
5	ALAWOONA	7
6	ALBANY	1
7	ALBERT PARK	444
8	ALBERTON	761
9	ALDGATE	255
10	ALDINGA	379

A new data set containing the suburb and the number of offence count that occurred in the su burb is created through groupby and it is named groupbySuburb. This data sets displays all values of offence counts that occurs in each suburb.

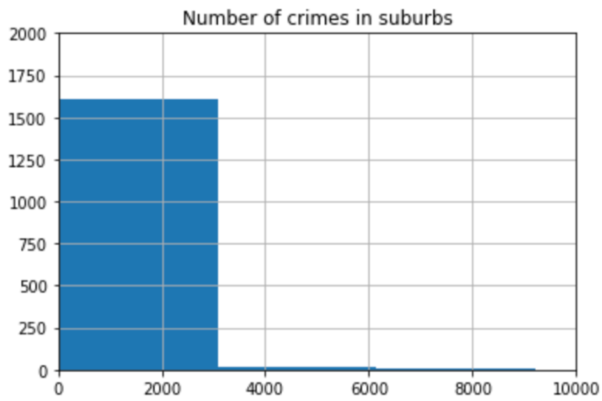
Code for historgram:

groupbySuburb['Offence Count'].hist(bins = 8)

plt.xlim(0,10000)   # setting limit on x-axis

plt.ylim(0,2000)   # setting limit on y-axis

plt.title('Number of crimes in suburbs')



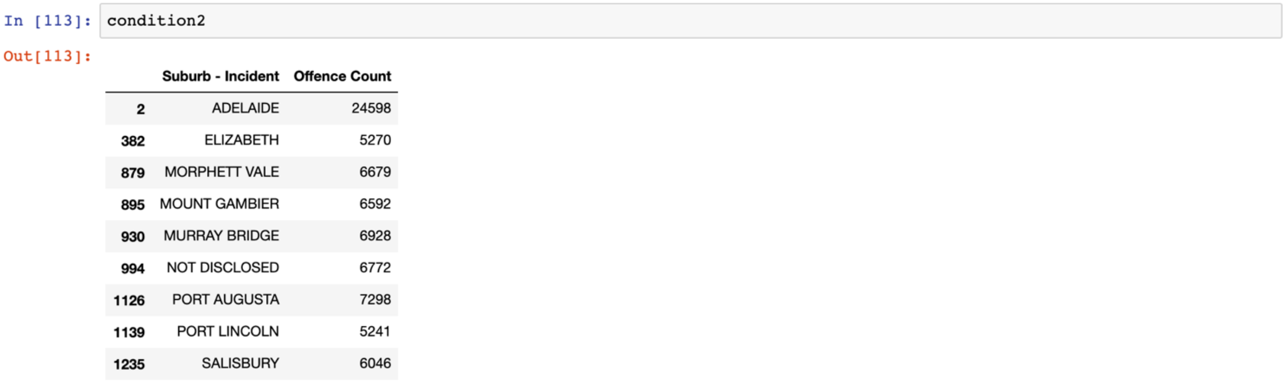
Histogram of total number of crimes in suburb

2. **Consider the shape of the histogram, what can you tell? Compare the mean and median values of the plotted histogram.**

   The histogram is skewed to the right therefore the mean is greater than the median

3. **In which suburbs the total number of crimes are greater than 5000? Plot the total number of crimes in the suburbs with the highest number of crimes (greater than 5000) using a bar chart.**

   condition2 = groupbySuburb[(groupbySuburb['Offence Count'] > 5000)]



```
In [113]: condition2
Out[113]:
```

	Suburb - Incident	Offence Count
2	ADELAIDE	24598
382	ELIZABETH	5270
879	MORPHETT VALE	6679
895	MOUNT GAMBIER	6592
930	MURRAY BRIDGE	6928
994	NOT DISCLOSED	6772
1126	PORT AUGUSTA	7298
1139	PORT LINCOLN	5241
1235	SALISBURY	6046

A new dataset is created from the dataset groupbySuburb with the condition where the values in the ‘Offence Count’ column is more than 5000. This data sets displays the values of offence counts that is more than 5000 that occurs in each suburb.

Code used to create bar chart:

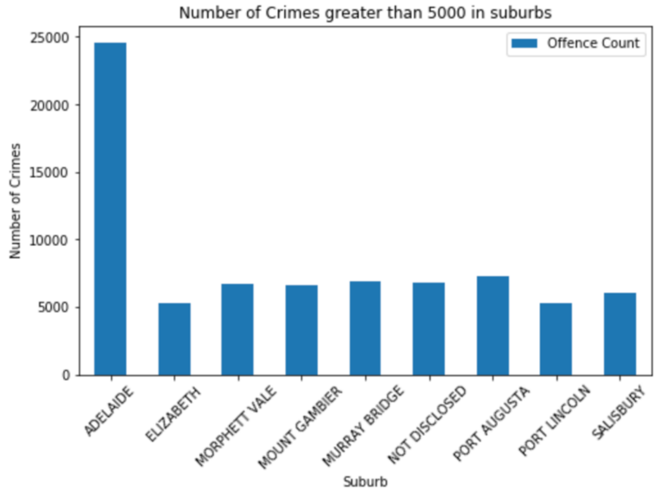
ax=condition2.plot.bar(figsize=(8,5))   #figsize sets size of plot

ax.set_xticklabels(condition2['Suburb - Incident'],rotation=45)

plt.xlabel('Suburb')   # setting a label for x axis

plt.ylabel('Number of Crimes')   # Setting a label for y axis

plt.title('Number of Crimes greater than 5000 in suburbs')   # Setting the title of chart

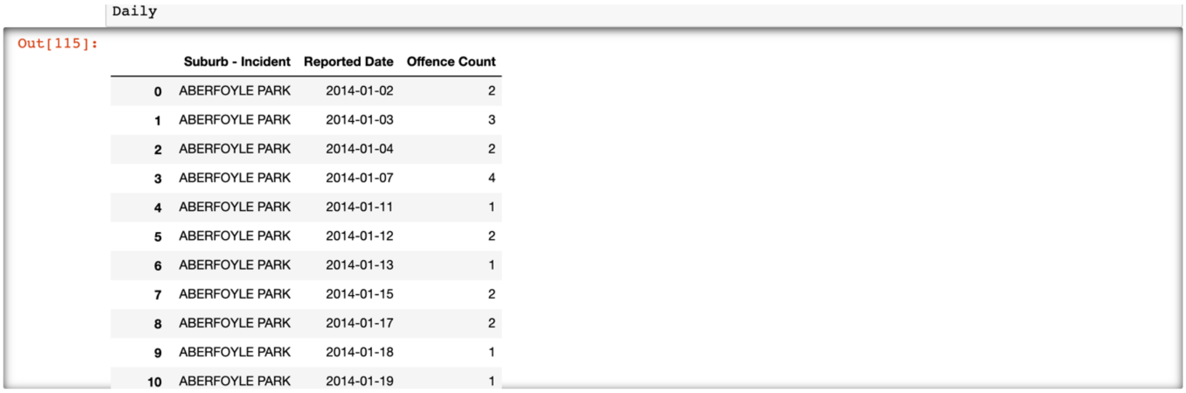


According to the bar chart, Adelaide is the suburb where the crimes are the greatest and Port Lincoln is the suburb where the crimes are the least.

B3. Daily number of crimes

1. **For each suburb, calculate the number of days that at least 15 crimes have occurred per day. (Note: your answer should contain all suburbs in the dataset together with a value showing the number of days that at least 15 crimes have happened)**

   Daily = crime_statistics.groupby(['Suburb - Incident', 'Reported Date'])['Offence Count'].sum()
   Daily = Daily.reset_index()

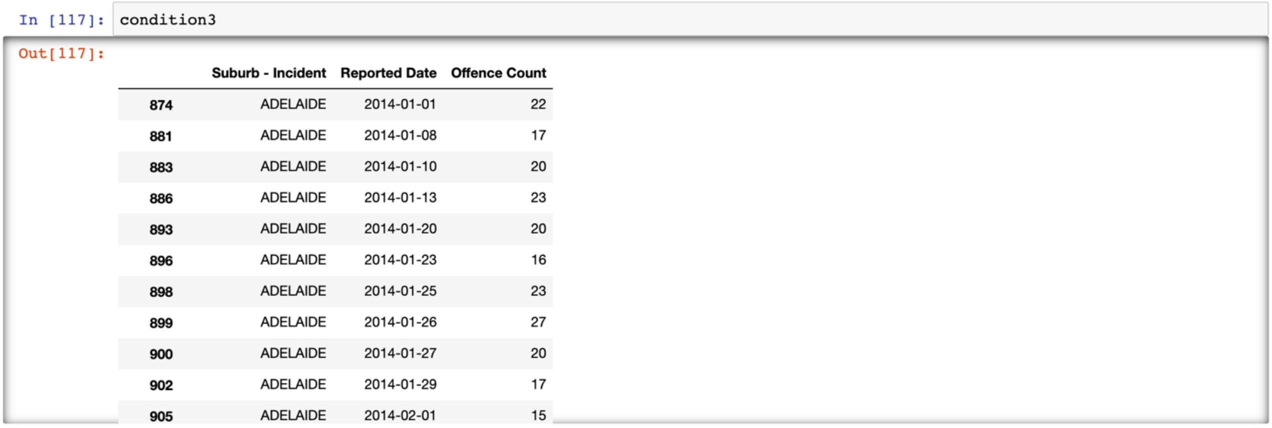


```
Daily
Out[115]:
```

	Suburb - Incident	Reported Date	Offence Count
0	ABERFOYLE PARK	2014-01-02	2
1	ABERFOYLE PARK	2014-01-03	3
2	ABERFOYLE PARK	2014-01-04	2
3	ABERFOYLE PARK	2014-01-07	4
4	ABERFOYLE PARK	2014-01-11	1
5	ABERFOYLE PARK	2014-01-12	2
6	ABERFOYLE PARK	2014-01-13	1
7	ABERFOYLE PARK	2014-01-15	2
8	ABERFOYLE PARK	2014-01-17	2
9	ABERFOYLE PARK	2014-01-18	1
10	ABERFOYLE PARK	2014-01-19	1

A new data set containing the suburb, Reported date and the number of offence count in the suburb is created through groupby and it is named Daily. This dataset displays the number of offence counts that happens daily in suburbs.

condition3 = Daily[Daily['Offence Count'] >= 15]



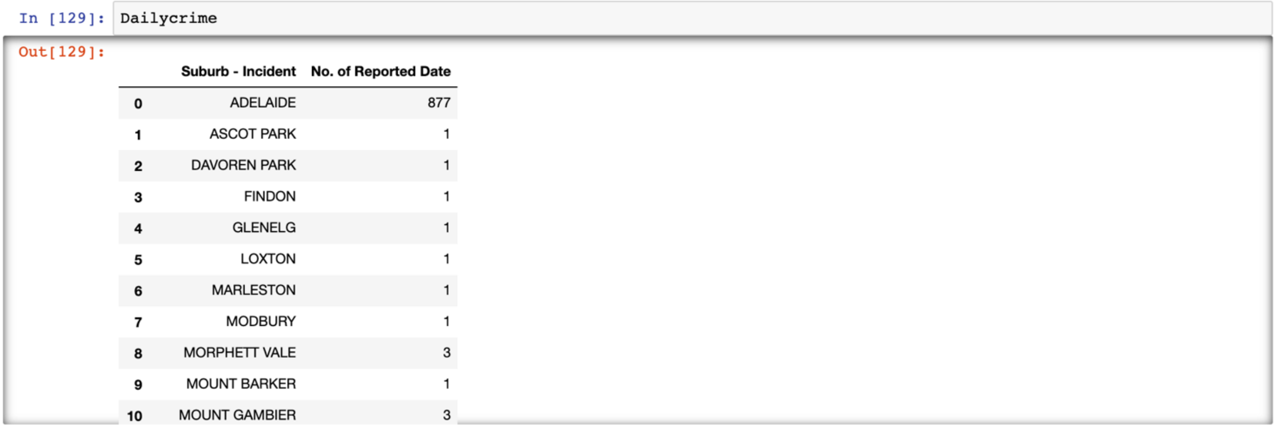
```
In [117]: condition3
Out[117]:
```

	Suburb - Incident	Reported Date	Offence Count
874	ADELAIDE	2014-01-01	22
881	ADELAIDE	2014-01-08	17
883	ADELAIDE	2014-01-10	20
886	ADELAIDE	2014-01-13	23
893	ADELAIDE	2014-01-20	20
896	ADELAIDE	2014-01-23	16
898	ADELAIDE	2014-01-25	23
899	ADELAIDE	2014-01-26	27
900	ADELAIDE	2014-01-27	20
902	ADELAIDE	2014-01-29	17
905	ADELAIDE	2014-02-01	15

A new dataset is created from the dataset Daily with the condition where the values in the ‘Off ence Count’ column is more than 15.

Dailycrime = condition3.groupby(['Suburb - Incident'])['Reported Date'].count()

Dailycrime = Dailycrime.reset_index()



```
In [129]: Dailycrime
Out[129]:
```

	Suburb - Incident	No. of Reported Date
0	ADELAIDE	877
1	ASCOT PARK	1
2	DAVOREN PARK	1
3	FINDON	1
4	GLENELG	1
5	LOXTON	1
6	MARLESTON	1
7	MODBURY	1
8	MORPHETT VALE	3
9	MOUNT BARKER	1
10	MOUNT GAMBIER	3

A new data set containing suburb and number of reported date is created through groupby and it is named Dailycrimes. This dataset displays the number of reported dates that has more than 15 offence counts in each suburbs.

2. **Now which suburbs do have at least one day where the daily number of crimes are more than 15. Plot the number of days that at least 15 crimes have occurred for the suburbs you found in this step (step 2) using a bar graph.**

Code for bar chart:

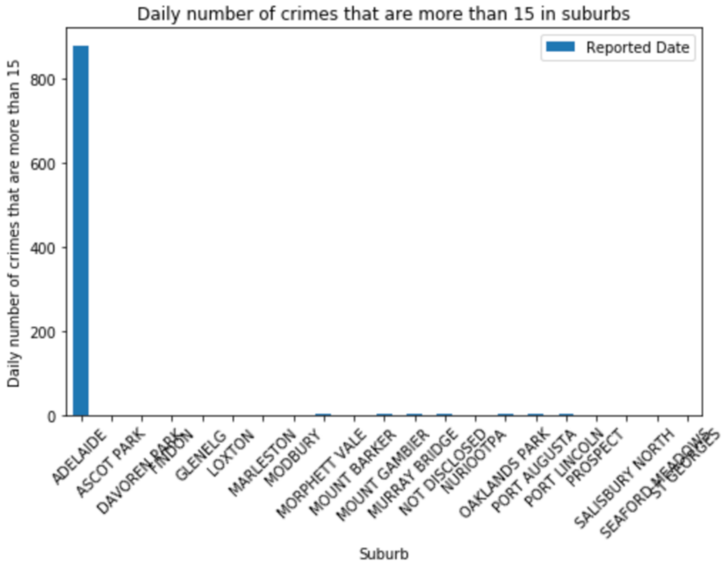
ax=Dailycrime.plot.bar(figsize=(8,5))# figsize sets size of plot

ax.set_xticklabels(Dailycrime['Suburb - Incident'],rotation=45)

plt.xlabel('Suburb')# setting a label for x axis

plt.ylabel('Daily number of crimes that are more than 15')# Setting a label for y axis

plt.title('Daily number of crimes that are more than 15 in suburbs')# Setting the title of chart



There are 21 suburbs that have at least one day where the daily number of crimes are more than 15. Adelaide has the most with 877 hence it making it hard to read the data in other suburbs.

3. Use an appropriate graph to visualize and detect outliers (extreme values) on the data from step 2 and remove them. Then, plot the data again using a bar graph.

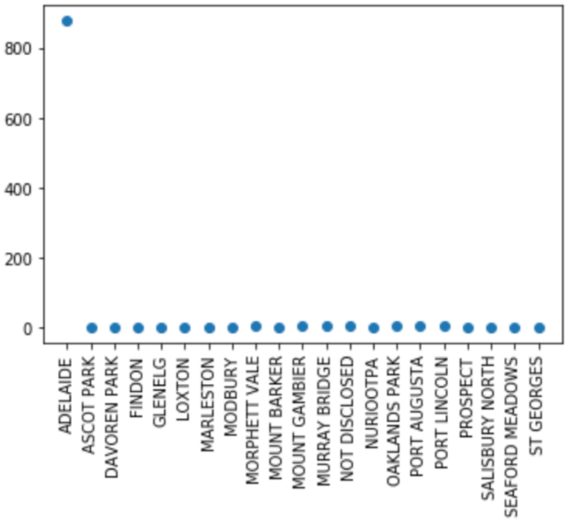
A scatter plot is used to visualize and detect outliers

Code for scatter plot:

```
plt.scatter(Dailycrime['Suburb - Incident'], Dailycrime['No. of Reported Date'])
```

```
plt.xticks(rotation=90)
```

```
plt.show()
```



Through the scatter plot, it is shown that Adelaide is the outlier because of the outlier, is hard to determine the values of the other suburbs. In order to remove the outlier, the IQR(Q3 – Q1) must be determined.

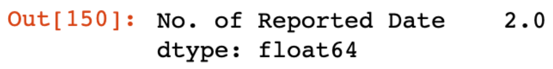
To determine IQR:

```
Q1 = Dailycrime.quantile(0.25)
```

```
Q3 = Dailycrime.quantile(0.75)
```

```
IQR = Q3 - Q1
```

```
IQR
```

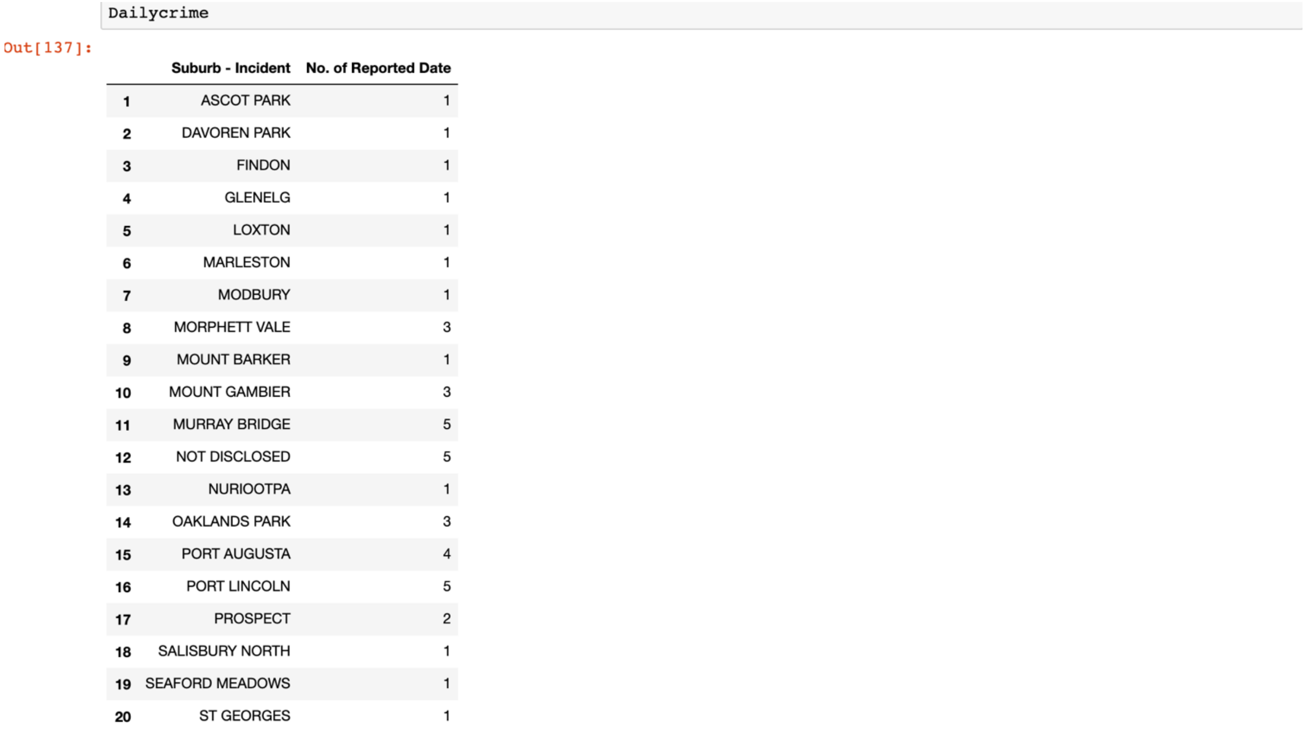


```
Out[150]: No. of Reported Date    2.0
          dtype: float64
```

The value if IQR = 2.0

```
Dailycrime = Dailycrime[~((Dailycrime < (Q1 - 1.5 * IQR)) |(Dailycrime > (Q3 + 1.5 * IQR))).any(axis=1)]
```

With the condition that if the value is not within the Reported Date, that value will be removed hence, the outlier is removed.



```
Dailycrime
```

Out[137]:

	Suburb - Incident	No. of Reported Date
1	ASCOT PARK	1
2	DAVOREN PARK	1
3	FINDON	1
4	GLENELG	1
5	LOXTON	1
6	MARLESTON	1
7	MODBURY	1
8	MORPHETT VALE	3
9	MOUNT BARKER	1
10	MOUNT GAMBIER	3
11	MURRAY BRIDGE	5
12	NOT DISCLOSED	5
13	NURIOOTPA	1
14	OAKLANDS PARK	3
15	PORT AUGUSTA	4
16	PORT LINCOLN	5
17	PROSPECT	2
18	SALISBURY NORTH	1
19	SEAFORD MEADOWS	1
20	ST GEORGES	1

As shown above, the outlier, Adelaide is removed.

A new bar chart is formed, showing clearer data.

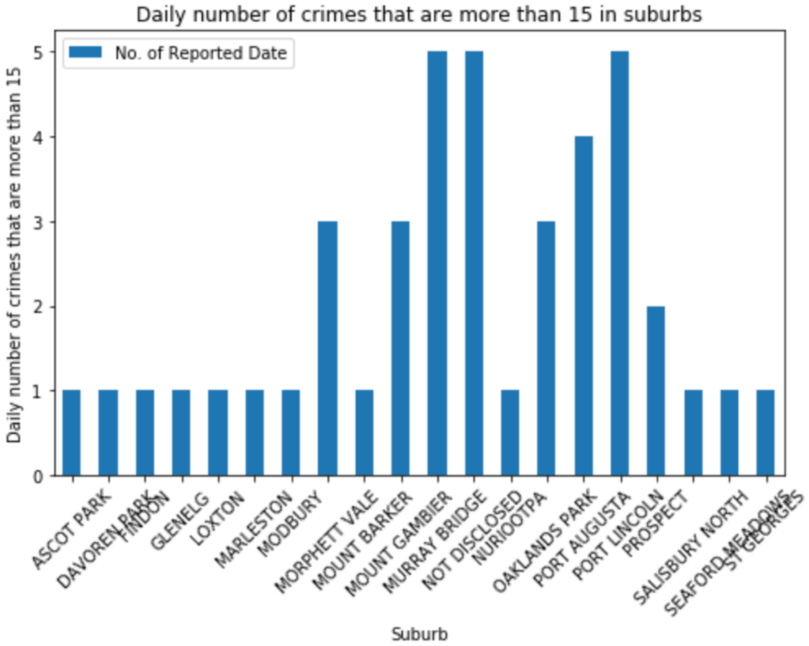
Bar chart code:

ax=Dailycrime.plot.bar(figsize=(8,5))# figsize sets size of plot

ax.set_xticklabels(Dailycrime['Suburb - Incident'],rotation=45plt.xlabel('Suburb')# setting a label for x axis

plt.ylabel('Daily number of crimes that are more than 15')# Setting a label for y axis

plt.title('Daily number of crimes that are more than 15 in suburbs')# Setting the title of chart



The data displayed is now much clearer with the removal of outliers. Excluding Adelaide, the most occurrence of offence count that occurred more than 15 times a day in 5 with 1 bring the least.

4. **Compare the bar graphs in step 2 and 3. Which bar graph is easier to interpret? Why?**

Bar graph in step 3 is easier to interpret because the outliers have been removed hence making it easier to read the rest of the other data