In [1]:
```
#Import required packages
from pathlib import Path
import csv
import pandas as pd
import geopandas as gpd
import glob
#from shapely.geometry import Polygon, mapping
from shapely.geometry import Point
import numpy as np
#from shapely.geometry.polygon import Polygon
#pip install mlxtend
```

1. Crime data Clean

In [154]:
```
path = Path.cwd()
```

In [156]:
```
Crime = "Crime_Data.csv"
```

In [157]:
```
with open (path/Crime, newline = '') as file:
    Crime_df = pd.read_csv(file, skipinitialspace=True)

Crime_df
```

Out[157]:

	Offnc_ID	Offence_Date	Offence_hour	Offence_hour_grp	Rep_Mthd	Vict_Typ	Vict_Sex	Vict_Age	Dom_Flg	Alc_Flg	...	Place_Desc	Sub_Txt	WAPOL_Level_3
0	3087901	2/11/2012	22	21:00 - 23:59	In Person	NaN	NaN	Unknown	Y	N	...	House	EXMOUTH	21100
1	3159012	16/01/2013	17	15:00 - 17:59	Police In	NaN	NaN	Unknown	N	N	...	Street / Footpath	SOUTH HEDLAND	21100
2	3741089	30/09/2014	11	09:00 - 11:59	Phone	NaN	NaN	Unknown	N	N	...	Shop	WICKHAM	21200
3	3987783	3/06/2015	8	06:00 - 08:59	Police In	NaN	NaN	Unknown	N	N	...	Other Dwelling	CORAL BAY	21200
4	3987786	3/06/2015	8	06:00 - 08:59	Police In	NaN	NaN	Unknown	N	N	...	Other Dwelling	CORAL BAY	21200
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
145665	4759929	17/05/2017	10	09:00 - 11:59	In Person	Person	54.0	Male	N	N	...	House	SOUTH HEDLAND	12100
145666	2207309	10/01/2010	5	03:00 - 05:59	In Person	Person	37.0	Female	N	N	...	House	MILLARS WELL	12100
145667	2213839	18/01/2010	3	03:00 - 05:59	In Person	Business	NaN	Male	N	N	...	House	SOUTH HEDLAND	12100
145668	5003027	2/01/2018	0	00:00 - 02:59	In Person	Business	NaN	Unknown	N	N	...	Restaurant	SOUTH HEDLAND	12100
145669	2217482	21/01/2010	16	15:00 - 17:59	Phone	Business	NaN	Male	N	N	...	House	SOUTH HEDLAND	12500

145670 rows × 21 columns

In [ ]:

In [ ]:

Converting date to datetime, creating new column from date to Day_of_week.

*The raw data, day of week and financial year didn't match the crime recorded date, hence these two columns won't be used. Financial year will be created in powerBI

In [158]:
```
Crime_df['Offence_Date'] = pd.to_datetime(Crime_df['Offence_Date'], dayfirst = True)
```

In [159]:
```
Crime_df["Day_of_week"] = Crime_df.Offence_Date.dt.day_name()
```

In [160]:
```
Crime_df.columns
```

Out[160]:
```
Index(['Offnc_ID', 'Offence_Date', 'Offence_hour', 'Offence_hour_grp',
       'Rep_Mthd', 'Vict_Typ', 'Vict_Sex', 'Vict_Age', 'Dom_Flg', 'Alc_Flg',
       'Drug_Flg', 'Place_Desc', 'Sub_Txt', 'WAPOL_Level_3', 'WAPOL_Group_3',
       'WAPOL_Level_4', 'WAPOL_Group_4', 'Res_Desc', 'Offence_weekday',
       'Offence_year_cal', 'Offence_year_fin', 'Day_of_week'],
      dtype='object')
```

Swapping Age and Gender columns

In [161]:
```
Crime_df.columns = ['Offnc_ID', 'Offence_Date', 'Offence_hour', 'Offence_hour_grp',
       'Rep_Mthd', 'Vict_Typ',   'Vict_Age', 'Vict_Sex', 'Dom_Flg', 'Alc_Flg',
       'Drug_Flg', 'Place_Desc', 'Sub_Txt', 'WAPOL_Level_3', 'WAPOL_Group_3',
       'WAPOL_Level_4', 'WAPOL_Group_4', 'Res_Desc', 'Offence_weekday',
       'Offence_year_cal', 'Offence_year_fin', 'Day_of_week']
#Crime_df=Crime_df.reindex(columns=columns_titles)
```

In [ ]:

Finding all the columns without Nan Values

In [162]:
```python
Crime_df[Crime_df.columns[~Crime_df.isnull().any()]].columns
```

Out[162]:
```
Index(['Offnc_ID', 'Offence_Date', 'Offence_hour', 'Offence_hour_grp',
       'Rep_Mthd', 'Vict_Sex', 'Dom_Flg', 'Alc_Flg', 'Sub_Txt',
       'WAPOL_Level_3', 'WAPOL_Group_3', 'WAPOL_Level_4', 'WAPOL_Group_4',
       'Day_of_week'],
      dtype='object')
```

Finding all the columns with Nan Values

In [163]:
```python
#all the columns with Nan Values
Crime_df[Crime_df.columns[Crime_df.isnull().any()]].columns
```

Out[163]:
```
Index(['Vict_Typ', 'Vict_Age', 'Drug_Flg', 'Place_Desc', 'Res_Desc',
       'Offence_weekday', 'Offence_year_cal', 'Offence_year_fin'],
      dtype='object')
```

Making sure all the crime Offnc ID is unique

In [164]:
```python
Crime_df.Offnc_ID.is_unique
```

Out[164]:
```
True
```

Checking unique values for hour_grp

In [165]:
```python
Crime_df.Offence_hour_grp.unique()
```

Out[165]:
```
array(['21:00 - 23:59', '15:00 - 17:59', '09:00 - 11:59', '06:00 - 08:59',
       '00:00 - 02:59', '12:00 - 14:59', '18:00 - 20:59', '03:00 - 05:59'],
      dtype=object)
```

Updating letter cases for suburb columns

In [166]:
```python
Crime_df.Sub_Txt = Crime_df.Sub_Txt.str.title()
```

filling all the null values with N for Drug_flg cloumns, to keep it consistent with other flag columns

In [167]:
```python
sum(Crime_df.Drug_Flg.isnull())
```

Out[167]:
```
34904
```

In [168]:
```python
Crime_df["Drug_Flg"].fillna("U", inplace = True)
```

Finally strip all the white spaces

In [183]:
```python
Crime_df['Place_Desc']=Crime_df['Place_Desc'].str.strip()
```

In [184]:
```python
Crime_df.to_csv("Crime_df.csv",index=False,header=True,sep=',')
```

In [173]:
```python
Crime_df
```

Out[173]:

	Offnc_ID	Offence_Date	Offence_hour	Offence_hour_grp	Rep_Mthd	Vict_Typ	Vict_Age	Vict_Sex	Dom_Flg	Alc_Flg	...	Sub_Txt	WAPOL_Level_3	WAPOL_Grou
0	3087901	2012-11-02	22	21:00 - 23:59	In Person	NaN	NaN	Unknown	Y	N	...	Exmouth	21100	Drug Offer
1	3159012	2013-01-16	17	15:00 - 17:59	Police In	NaN	NaN	Unknown	N	N	...	South Hedland	21100	Drug Offer
2	3741089	2014-09-30	11	09:00 - 11:59	Phone	NaN	NaN	Unknown	N	N	...	Wickham	21200	Receiving Possessic Stolen Prop
3	3987783	2015-06-03	8	06:00 - 08:59	Police In	NaN	NaN	Unknown	N	N	...	Coral Bay	21200	Receiving Possessic Stolen Prop
4	3987786	2015-06-03	8	06:00 - 08:59	Police In	NaN	NaN	Unknown	N	N	...	Coral Bay	21200	Receiving Possessic Stolen Prop
...	...	...	...	...	...	...	...	...	...	...	...	...	...	
145665	4759929	2017-05-17	10	09:00 - 11:59	In Person	Person	54.0	Male	N	N	...	South Hedland	12100	Burg
145666	2207309	2010-01-10	5	03:00 - 05:59	In Person	Person	37.0	Female	N	N	...	Millars Well	12100	Burg
145667	2213839	2010-01-18	3	03:00 - 05:59	In Person	Business	NaN	Male	N	N	...	South Hedland	12100	Burg
145668	5003027	2018-01-02	0	00:00 - 02:59	In Person	Business	NaN	Unknown	N	N	...	South Hedland	12100	Burg
145669	2217482	2010-01-21	16	15:00 - 17:59	Phone	Business	NaN	Male	N	N	...	South Hedland	12500	Property Dam

145670 rows × 22 columns

In [ ]:

2. Cleaning suburb column naming convension, reading in suburb population data, and downloading suburb JSON files from

https://data.gov.au/dataset/ds-dga-6a0ec945-c880-4882-8a81-4dbcb85e74e5/details?q=wa%20suburb%20boundary (https://data.gov.au/dataset/ds-dga-6a0ec945-c880-4882-8a81-4dbcb85e74e5/details?q=wa%20suburb%20boundary)

In [266]:
```
unique_sub = Crime_df.Sub_Txt.unique()
```

All the unique suburb names in raw crime data

In [267]:
```
unique_sub
```

Out[267]:
```
array(['Exmouth', 'South Hedland', 'Wickham', 'Coral Bay', 'Karratha',
       'Roebourne', 'Marble Bar', 'Millars Well', 'Port Hedland',
       'Newman', 'Pegs Creek', 'Nickol', 'Onslow', 'Tom Price',
       'Barrow Island', 'Fortescue', 'Talandji', 'Bulgarra', 'Whim Creek',
       'Baynton', 'Wedgefield', 'Karijini', 'Strelley', 'Mount Sheila',
       'Dampier', 'Pannawonica', 'Paraburdoo', 'Learmonth', 'Stove Hill',
       'Telfer', 'Boodarie', 'Millstream', 'Nullagine', 'Point Samson',
       'Pippingarra', 'Capricorn', 'Cape Lambert', 'Cossack',
       'Mulga Downs', 'Cooya Pooya', 'Karratha Industrial Estate',
       'Pardoo', 'Maitland', 'Mardie', 'Gap Ridge', 'Indee', 'Jigalong',
       'Antonymyre', 'Cane', 'Mundabullangana', 'Hamersley Range',
       'Peedamulla', 'Juna Downs', 'Burrup', 'Nanutarra', 'De Grey',
       'Balla Balla', 'Wittenoom', 'Cleaverville', 'Rocklea', 'Redbank',
       'Dampier Archipelago', 'Innawanga', 'Mount Anketell',
       'North West Cape', 'Finucane', 'Sherlock', 'Exmouth Gulf',
       'Gnoorea', 'Yannarie', 'Mulataga', 'Ningaloo',
       'Cape Range National Park'], dtype=object)
```

In [ ]:

Read in population file tabs

In [173]:
```
District_pop = "District_Pop.csv"
```

In [174]:
```
with open (path/District_pop, newline = '') as file:
    District_df = pd.read_csv(file, skipinitialspace=True)

District_df
```

Out[174]:

	Suburb	Station	District	2015	2016	2017	2018	2019	2020	2021	2022
0	MOUNT SHEILA	TOM PRICE	PILBARA	1031.0	1083.0	1060	1011	955	900	836	835
1	MULATAGA	KARRATHA	PILBARA	1.0	0.0	0	0	0	0	0	0
2	MULGA DOWNS	TOM PRICE	PILBARA	331.0	189.0	69	58	47	37	25	24
3	MUNDABULLANGANA	SOUTH HEDLAND	PILBARA	17.0	13.0	15	13	12	12	14	13
4	NANUTARRA	ONSLOW	PILBARA	64.0	76.0	181	146	110	75	38	38
...	...	...	...	...	...	...	...	...	...	...	...
70	GNOOREA	DAMPIER	PILBARA	10.0	8.0	11	10	9	8	8	8
71	HAMERSLEY RANGE	PANNAWONICA	PILBARA	0.0	0.0	993	802	605	410	208	208
72	INDEE	SOUTH HEDLAND	PILBARA	47.0	37.0	16	14	13	13	15	15
73	INNAWANGA	PARABURDOO	PILBARA	21.0	26.0	30	24	18	12	6	6
74	STOVE HILL	KARRATHA	PILBARA	51.0	42.0	8	8	7	8	6	7

75 rows × 11 columns

In [175]:
```
District_df.Suburb = District_df.Suburb.str.title()
```

In [176]:
```
District_df.Station = District_df.Station.str.title()
```

In [177]:
```
District_df
```

Out[177]:

	Suburb	Station	District	2015	2016	2017	2018	2019	2020	2021	2022
0	Mount Sheila	Tom Price	PILBARA	1031.0	1083.0	1060	1011	955	900	836	835
1	Mulataga	Karratha	PILBARA	1.0	0.0	0	0	0	0	0	0
2	Mulga Downs	Tom Price	PILBARA	331.0	189.0	69	58	47	37	25	24
3	Mundabullangana	South Hedland	PILBARA	17.0	13.0	15	13	12	12	14	13
4	Nanutarra	Onslow	PILBARA	64.0	76.0	181	146	110	75	38	38
...	...	...	...	...	...	...	...	...	...	...	...
70	Gnoorea	Dampier	PILBARA	10.0	8.0	11	10	9	8	8	8
71	Hamersley Range	Pannawonica	PILBARA	0.0	0.0	993	802	605	410	208	208
72	Indee	South Hedland	PILBARA	47.0	37.0	16	14	13	13	15	15
73	Innawanga	Paraburdoo	PILBARA	21.0	26.0	30	24	18	12	6	6
74	Stove Hill	Karratha	PILBARA	51.0	42.0	8	8	7	8	6	7

75 rows × 11 columns

In [292]:
```
Station_Pop_df = District_df.iloc[:,[1,3,4,5,6,7,8,9,10]]
```

Caculating each police station coverage zone population based on suburb populations for each year

In [271]:
```
Station_Pop_df_1 = Station_Pop_df.groupby('Station').sum()
Station_Pop_df_1
```

Out[271]:

Station	2015	2016	2017	2018	2019	2020	2021	2022
Dampier	4223.0	4161.0	3758	3355	2900	2492	2072	2111
Exmouth	2974.0	3000.0	3096	3177	3267	3399	3460	3559
Jigalong	0.0	0.0	362	359	350	340	326	321
Karratha	17073.0	16363.0	16508	16837	17317	17761	17892	18229
Marble Bar	1649.0	1553.0	1960	1903	1842	1779	1691	1664
Newman	8434.0	7799.0	7145	7067	7018	6988	6935	6921
Nullagine	1606.0	1751.0	1305	1297	1288	1276	1245	1226
Onslow	3582.0	4154.0	1655	1506	1346	1188	1020	1019
Pannawonica	766.0	764.0	2684	2310	1922	1538	1134	1133
Paraburdoo	1801.0	1723.0	1783	1737	1680	1627	1562	1561
Port Hedland	4459.0	4392.0	4314	4277	4258	4268	4307	4315
Roebourne	3777.0	3729.0	3830	3726	3623	3577	3558	3539
South Hedland	10806.0	10564.0	10631	10903	11314	11851	12341	12659
Tom Price	4649.0	4398.0	4297	4222	4125	4056	3966	4016

In [272]:
```
Station_Pop_df_1 = Station_Pop_df_1.reset_index()
```

Creating a simple Station -> Suburb dimension table

In [249]:
```
sub_sta = District_df.iloc[:,[0,1]]
sub_sta
```

Out[249]:

	Suburb	Station
0	Mount Sheila	Tom Price
1	Mulataga	Karratha
2	Mulga Downs	Tom Price
3	Mundabullangana	South Hedland
4	Nanutarra	Onslow
...	...	...
70	Gnoorea	Dampier
71	Hamersley Range	Pannawonica
72	Indee	South Hedland
73	Innawanga	Paraburdoo
74	Stove Hill	Karratha

75 rows × 2 columns

In [250]:
```
sub_sta.to_csv("sub_sta.csv",index=False,header=True,sep=',')
```

In [248]:
```
Station_Pop_df_1.to_csv("Station_Pop_df.csv",index=False,header=True,sep=',')
```

Unique suburb names from district population table

In [180]:
```
#Unique suburb names from District_df
unique_sub_2 = District_df.Suburb.unique()
```

Combine all the unique suburb names, from Crime data tab, and district population tab, in order to prepare the shape file for the best coverage in Power BI visualisation.

In [183]:
```
all_suburbs = np.unique(np.concatenate((unique_sub,unique_sub_2),0))
```

In [275]:
```
all_suburbs
```

Out[275]:
```
array(['Angelo River', 'Antonymyre', 'Balla Balla', 'Barrow Island',
       'Baynton', 'Boodarie', 'Bulgarra', 'Burrup', 'Cane',
       'Cape Lambert', 'Cape Range National Park', 'Capricorn',
       'Chichester', 'Cleaverville', 'Cooya Pooya', 'Coral Bay',
       'Cossack', 'Dampier', 'Dampier Archipelago', 'De Grey', 'Exmouth',
       'Exmouth Gulf', 'Finucane', 'Fortescue', 'Gap Ridge', 'Gnoorea',
       'Hamersley Range', 'Indee', 'Innawanga', 'Jigalong', 'Juna Downs',
       'Karijini', 'Karratha', 'Karratha Industrial Estate', 'Learmonth',
       'Maitland', 'Marble Bar', 'Mardie', 'Millars Well', 'Millstream',
       'Mount Anketell', 'Mount Sheila', 'Mulataga', 'Mulga Downs',
       'Mundabullangana', 'Nanutarra', 'Newman', 'Nickol', 'Ningaloo',
       'North West Cape', 'Nullagine', 'Onslow', 'Pannawonica',
       'Paraburdoo', 'Pardoo', 'Peedamulla', 'Pegs Creek', 'Pippingarra',
       'Point Samson', 'Port Hedland', 'Redbank', 'Rocklea', 'Roebourne',
       'Sherlock', 'South Hedland', 'Stove Hill', 'Strelley', 'Talandji',
       'Telfer', 'Tom Price', 'Wallareenya', 'Wedgefield', 'Whim Creek',
       'Wickham', 'Wittenoom', 'Yannarie'], dtype=object)
```

Read JSON file with GeoPanda Package

In [190]:
```
suburbs_json = gpd.read_file('https://data.gov.au/geoserver/wa-suburb-locality-boundaries-psma-administrative-boundaries/wfs?request=GetFe
```

In [193]:
```
suburbs_json.wa_local_2 = suburbs_json.wa_local_2.str.title()
```

Only keep the relavant Pilbara region area with all the above prepared suburb name list in shapefile, by masking out all the other non relavant suburbs

In [194]:
```
suburb_mask = suburbs_json.wa_local_2.isin(all_suburbs)
District_df_2 = suburbs_json [suburb_mask]
```

In [ ]:

A quick cross reference: find out all the suburbs in the district population table, but not in crime data table. Meaning these suburbs has no crimes over the year, or simply not recorded. Populations for these suburbs are extremely low.

In [228]:
```
different_values = [element for element in unique_sub_2 if element not in unique_sub]
```

In [229]:
```
different_values
```

Out[229]:
```
['Chichester', 'Wallareenya', 'Angelo River']
```

In [284]:
```
District_df.loc[District_df['Suburb'].isin(['Chichester', 'Wallareenya', 'Angelo River']),:]
```

Out[284]:

	Suburb	Station	District	2015	2016	2017	2018	2019	2020	2021	2022
11	Chichester	Roebourne	PILBARA	38.0	22.0	129	109	89	67	46	46
17	Wallareenya	South Hedland	PILBARA	0.0	0.0	0	0	0	0	0	0
45	Angelo River	Newman	PILBARA	9.0	8.0	5	5	5	5	5	5

In [ ]:

A quick cross reference, find out all the suburbs in the crime data table, but not in district population table. Meaning these suburbs has no population info, but has crime cases recorded over the year in this suburb. Only 1 crime case found below in Cape Lambert

In [230]:
```
different_values_1 = [element for element in unique_sub if element not in unique_sub_2]
```

In [231]:
```
different_values_1
```

Out[231]:
```
['Cape Lambert']
```

In [277]:
```
Crime_df[Crime_df["Sub_Txt"] == 'Cape Lambert']
```

Out[277]:

	Offnc_ID	Offence_Date	Offence_hour	Offence_hour_grp	Rep_Mthd	Vict_Typ	Vict_Age	Vict_Sex	Dom_Flg	Alc_Flg	...	Sub_Txt	WAPOL_Level_3	WAPOL_Group_3
718	5153809	2018-05-11	13	12:00 - 14:59	Phone	Business	NaN	Female	N	N	...	Cape Lambert	74100	Trespass

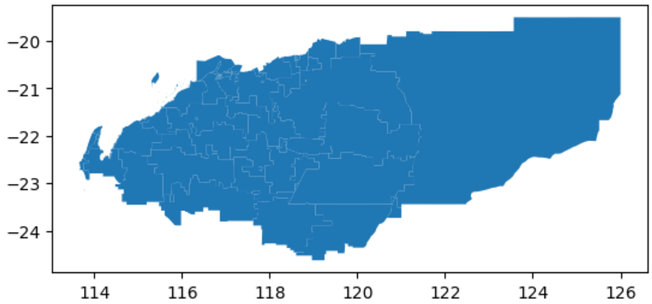
1 rows × 22 columns

In [ ]:

Write final Geojson to file, convert into TopoJson format for Power BI.

In [200]:
```
District_df_2.to_file('District_df.geojson')
```

In [112]:
```
District_df.to_csv("District_df.csv",index=False,header=True,sep=',')
```

In [286]:
```
District_df_2.plot()
```
Out[286]:
```
<Axes: >
```


3. Association Rule Mining

In [ ]:
```
#pip install mlxtend
```
In [10]:
```
from mlxtend.frequent_patterns import apriori, association_rules
```
In [3]:
```
A_rules = "Association_Rules.csv"
```
In [64]:
```
with open (path/A_rules, newline = '') as file:
    Arules_df = pd.read_csv(file, skipinitialspace=True)
Arules_df
```
Out[64]:

	Offnc_ID	Offence_Date	Place_Desc	Sub_Txt	WAPOL_Group_3
0	94438477	25/09/2022	Street / Footpath	BULGARRA	Sexual Offences
1	85045679	1/07/2022	Office	SOUTH HEDLAND	Property Damage
2	90285477	23/08/2022	Airport	NEWMAN	Stealing of Motor Vehicle
3	109175881	31/01/2023	Mining Site	PIPPINGARRA	Stealing of Motor Vehicle
4	109174077	31/01/2023	Mining Site	PIPPINGARRA	Stealing of Motor Vehicle
...	...	...	...	...	...
7568	118653077	25/04/2023	Commercial Workshop	NEWMAN	Stealing
7569	119658118	31/05/2023	Shop	NEWMAN	Stealing
7570	114169877	22/03/2023	Shop	KARRATHA	Stealing
7571	119633567	17/05/2023	Hotel / Tavern	NICKOL	Stealing
7572	114698277	24/02/2023	House	PEGS CREEK	Sexual Offences

7573 rows × 5 columns

Selecting suburbs, then group by date, aggregate all unique crime types into list as itemset

In [65]:
```
Arules = Arules_df.loc[Arules_df['Sub_Txt'] == 'SOUTH HEDLAND']
```
In [144]:
```
final_Arules = Arules.groupby('Offence_Date',as_index=False).agg(Crime_list = ('WAPOL_Group_3', lambda x : ','.join(x.unique())))
```
In [145]:
```
final_Arules
```
Out[145]:

	Offence_Date	Crime_list
0	1/01/2023	Assault,Threatening Behaviour,Property Damage,...
1	1/02/2023	Assault,Sexual Offences,Property Damage
2	1/03/2023	Assault,Stealing
3	1/04/2023	Assault,Property Damage
4	1/05/2023	Burglary,Assault,Property Damage,Stealing
...	...	...
331	9/08/2022	Property Damage,Threatening Behaviour,Stealing...
332	9/09/2022	Stealing,Sexual Offences,Burglary,Property Dam...
333	9/10/2022	Stealing,Threatening Behaviour,Assault,Propert...
334	9/11/2022	Assault,Stealing of Motor Vehicle,Property Dam...
335	9/12/2022	Burglary,Assault,Stealing

336 rows × 2 columns

In [68]:
```
Itemsets = list(final_Arules["Crime_list"].apply(lambda x: x.split(",")))
```
In [69]:
```
Itemsets[:5]
```
Out[69]:
```
[['Assault', 'Threatening Behaviour', 'Property Damage', 'Robbery'],
 ['Assault', 'Sexual Offences', 'Property Damage'],
 ['Assault', 'Stealing'],
 ['Assault', 'Property Damage'],
 ['Burglary', 'Assault', 'Property Damage', 'Stealing']]
```

In [70]:
```
from mlxtend.preprocessing import TransactionEncoder
```

Using TransactionEncoder, convert the list to a One-Hot Encoded Boolean list.

In [71]:
```
a = TransactionEncoder()
a_data = a.fit(Itemsets).transform(Itemsets)
df = pd.DataFrame(a_data,columns=a.columns_)
df
```

Out[71]:

	Arson	Assault	Burglary	Deprivation of Liberty	Property Damage	Robbery	Sexual Offences	Stealing	Stealing of Motor Vehicle	Threatening Behaviour
0	False	True	False	False	True	True	False	False	False	True
1	False	True	False	False	True	False	True	False	False	False
2	False	True	False	False	False	False	False	True	False	False
3	False	True	False	False	True	False	False	False	False	False
4	False	True	True	False	True	False	False	True	False	False
...	...	...	...	...	...	...	...	...	...	...
331	False	True	False	False	True	False	False	False	True	True
332	False	True	True	False	True	False	True	True	False	False
333	False	True	False	False	True	False	False	True	False	True
334	False	True	True	False	True	False	False	True	True	False
335	False	True	True	False	False	False	False	True	False	False

336 rows × 10 columns

Create the Apriori Model, min_support set as 30%

In [72]:
```
df_1 = apriori(df, min_support = 0.3, use_colnames = True, verbose = 1)
df_1
```

```
Processing 4 combinations | Sampling itemset size 43
```

Out[72]:

	support	itemsets
0	0.898810	(Assault)
1	0.491071	(Burglary)
2	0.607143	(Property Damage)
3	0.654762	(Stealing)
4	0.380952	(Threatening Behaviour)
5	0.449405	(Burglary, Assault)
6	0.556548	(Property Damage, Assault)
7	0.583333	(Stealing, Assault)
8	0.351190	(Assault, Threatening Behaviour)
9	0.330357	(Property Damage, Burglary)
10	0.324405	(Stealing, Burglary)
11	0.404762	(Stealing, Property Damage)
12	0.306548	(Property Damage, Burglary, Assault)
13	0.369048	(Stealing, Property Damage, Assault)

Set 60% as minimum confidence value. When antecedents happend, the likelihood of consequents also happens is 60% or more.

In [123]:
```
df_ar = association_rules(df_1, metric = "confidence", min_threshold = 0.6)
df_ar
```

Out[123]:

	antecedents	consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction	zhangs_metric
0	(Burglary)	(Assault)	0.491071	0.898810	0.449405	0.915152	1.018182	0.008025	1.192602	0.035088
1	(Property Damage)	(Assault)	0.607143	0.898810	0.556548	0.916667	1.019868	0.010842	1.214286	0.049587
2	(Assault)	(Property Damage)	0.898810	0.607143	0.556548	0.619205	1.019868	0.010842	1.031677	0.192513
3	(Stealing)	(Assault)	0.654762	0.898810	0.583333	0.890909	0.991210	-0.005173	0.927579	-0.025043
4	(Assault)	(Stealing)	0.898810	0.654762	0.583333	0.649007	0.991210	-0.005173	0.983603	-0.080574
5	(Threatening Behaviour)	(Assault)	0.380952	0.898810	0.351190	0.921875	1.025662	0.008787	1.295238	0.040417
6	(Burglary)	(Property Damage)	0.491071	0.607143	0.330357	0.672727	1.108021	0.032207	1.200397	0.191560
7	(Burglary)	(Stealing)	0.491071	0.654762	0.324405	0.660606	1.008926	0.002870	1.017219	0.017383
8	(Stealing)	(Property Damage)	0.654762	0.607143	0.404762	0.618182	1.018182	0.007228	1.028912	0.051724
9	(Property Damage)	(Stealing)	0.607143	0.654762	0.404762	0.666667	1.018182	0.007228	1.035714	0.045455
10	(Property Damage, Burglary)	(Assault)	0.330357	0.898810	0.306548	0.927928	1.032397	0.009619	1.404018	0.046861
11	(Burglary, Assault)	(Property Damage)	0.449405	0.607143	0.306548	0.682119	1.123490	0.033695	1.235863	0.199633
12	(Burglary)	(Property Damage, Assault)	0.491071	0.556548	0.306548	0.624242	1.121633	0.033243	1.180156	0.213081
13	(Stealing, Property Damage)	(Assault)	0.404762	0.898810	0.369048	0.911765	1.014414	0.005244	1.146825	0.023871
14	(Stealing, Assault)	(Property Damage)	0.583333	0.607143	0.369048	0.632653	1.042017	0.014881	1.069444	0.096774
15	(Property Damage, Assault)	(Stealing)	0.556548	0.654762	0.369048	0.663102	1.012737	0.004641	1.024754	0.028361
16	(Property Damage)	(Stealing, Assault)	0.607143	0.583333	0.369048	0.607843	1.042017	0.014881	1.062500	0.102639

In [ ]:

Below is just to mine the data from different angle, where take the crime location into considerations

In [185]:
```
Arules_df['Place_Desc'] = Arules_df['Place_Desc'].str.strip()
```

In [196]:
```
Arules_df_2 = Arules_df[['Place_Desc', 'WAPOL_Group_3']].agg(','.join, axis=1)
```

In [203]:
```
Arules_df_2 =  Arules_df_2.str.title()
```

In [ ]:

In [205]:
```
Itemsets_2 = list(Arules_df_2.apply(lambda x: x.split(",")))
```

In [ ]:

In [214]:
```
b = TransactionEncoder()
b_data = b.fit(Itemsets_2).transform(Itemsets_2)
df_2 = pd.DataFrame(b_data,columns=b.columns_)
```

In [213]:
```
df_3 = apriori(df_2, min_support = 0.05, use_colnames = True, verbose = 1)
```

```
Processing 15 combinations | Sampling itemset size 3
```

In [211]:
```
df_ar_3 = association_rules(df_3, metric = "confidence", min_threshold = 0.4)
df_ar_3
```

Out[211]:

	antecedents	consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction	zhangs_metric
0	(Assault)	(House)	0.361548	0.571108	0.250759	0.693572	1.214432	0.044277	1.399650	0.276560
1	(House)	(Assault)	0.571108	0.361548	0.250759	0.439075	1.214432	0.044277	1.138214	0.411689
2	(Burglary)	(House)	0.129539	0.571108	0.077512	0.598369	1.047734	0.003531	1.067876	0.052339
3	(Property Damage)	(House)	0.168361	0.571108	0.088076	0.523137	0.916004	-0.008076	0.899404	-0.099312

In [ ]:

In [ ]: