Homework 1 Preparing Time Series Data

By Anthony Ayala, Section C

Info about assignment:

- This data set contains the number of daily births in Quebec, Canada from January 1, 1977 to Dec 31, 1990. Use the techniques that we have learned in class to aggregate these data, create an indexed time series, and create a time plot of the data.
- In particular, you should:

- Choose a time index and aggregate the data. You may choose the time index (monthly, yearly, etc) and the aggregation method (total, mean, etc).

- Be sure to state the time index and aggregation method chosen.
- Note that you will need to convert the date to a date class variable and extract the month and year in order to aggregate.
- Create a time plot of the aggregated data.
- Save your Jupyter Notebook as a pdf file for submission
- Please name your file "lastname_firstname_hw1.pdf"

In [24]:
```
conda install pandoc
conda install nbconvert[webpdf]
```

```
  Cell In[24], line 1
    conda install pandoc
          ^
SyntaxError: invalid syntax
```

In [25]:
```
import pandoc
import nbconvert[webpdf]
```

```
  Cell In[25], line 2
    import nbconvert[webpdf]
                    ^
SyntaxError: invalid syntax
```

In [35]:
```
!pip install pandoc
```

```
Requirement already satisfied: pandoc in /usr/local/lib/python3.10/dist-packages (2.3)
Requirement already satisfied: plumbum in /usr/local/lib/python3.10/dist-packages (from pandoc) (1.8.2)
Requirement already satisfied: pip in /usr/local/lib/python3.10/dist-packages (from pandoc) (23.1)
```

In [37]:
```
import pandoc
```

In [1]:
```
# Import the libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import statsmodels.api as sm
```

Load Daily Births Data into a data frame

In [2]:
```
# Import / load the Data Set
daily_births = pd.read_csv('/content/daily_birth.csv')
```

In [3]:
```
daily_births.head(5) # Preview the data
```

Out[3]:

	Date	Num_Births
0	1977-01-01	208
1	1977-01-02	241
2	1977-01-03	274
3	1977-01-04	256
4	1977-01-05	294

In [4]:
```
daily_births.info() # Check the data types and see if we have any nulls
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5113 entries, 0 to 5112
Data columns (total 2 columns):
 #   Column      Non-Null Count  Dtype 
---  ------      --------------  ----- 
 0   Date        5113 non-null   object
 1   Num_Births  5113 non-null   int64 
dtypes: int64(1), object(1)
memory usage: 80.0+ KB
```

In [12]:
```
daily_births.isna().count() # To confirm, we don't have any nulls
```

Out[12]:
```
Date          5113
Num_Births    5113
dtype: int64
```

In [13]:
```
daily_births.describe() # We can see that the average amount of births is 255.0, we have a standard deviation of 41.85 births.
```

Out[13]:

	Num_Births
count	5113.000000
mean	250.8820259
std	41.8595570
min	136.000000
25%	220.000000
50%	256.000000
75%	282.000000
max	366.000000

Let's handle the Date properly by converting it to a datetime format using pandas to_datetime function.

- This step is crucial for Time Series Analysis and Accumulating the Data

In [7]:
```
# Let's convert the date column to a date class variable
daily_births['Date'] = pd.to_datetime(daily_births['Date'], format = "%Y-%m-%d")
```

In [10]:
```
daily_births.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5113 entries, 0 to 5112
Data columns (total 2 columns):
 #   Column      Non-Null Count  Dtype         
---  ------      --------------  -----         
 0   Date        5113 non-null   datetime64[ns]
 1   Num_Births  5113 non-null   int64         
dtypes: datetime64[ns](1), int64(1)
memory usage: 80.0 KB
```

Let's just take a peak at the data

In [28]:
```
daily_births['Date'].dt.year
```

Out[28]:
```
0       1977
1       1977
2       1977
3       1977
4       1977
       ...  
5108    1990
5109    1990
5110    1990
5111    1990
5112    1990
Name: Date, Length: 5113, dtype: int64
```

In [32]:
```
# Let's get a count of the yearly data
daily_births['Date'].dt.year.value_counts()
```

Out[32]:
```
1980    366
1984    366
1988    366
1979    365
1978    365
1981    365
1982    365
1983    365
1985    365
1986    365
1987    365
1989    365
1990    365
Name: Date, dtype: int64
```

In [29]:
```
# We can add a year column to the data set
daily_births['Year'] = daily_births['Date'].dt.year
daily_births.head()
```

	Date	Num_Births	Year
0	1977-01-01	208	1977
1	1977-01-02	241	1977
2	1977-01-03	274	1977
3	1977-01-04	256	1977
4	1977-01-05	294	1977

In [31]:
```
daily_births['Date'].dt.month
```

Out[31]:
```
0       1
1       1
2       1
3       1
4       1
       ..
5108    12
5109    12
5110    12
5111    12
5112    12
Name: Date, Length: 5113, dtype: int64
```

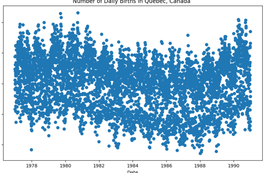
In [33]:
```
# Let's get a count of the yearly data
daily_births['Date'].dt.month.value_counts()
```

Out[33]:
```
1     434
3     434
5     434
7     434
8     434
10    434
12    434
4     420
6     420
9     420
11    420
2     395
Name: Date, dtype: int64
```

In [37]:
```
# We can add a year column to the data set
daily_births['Month'] = daily_births['Date'].dt.month
daily_births.head()
```

	Date	Num_Births	Year	Month
0	1977-01-01	208	1977	1
1	1977-01-02	241	1977	1
2	1977-01-03	274	1977	1
3	1977-01-04	256	1977	1
4	1977-01-05	294	1977	1

In [38]:
```
# Plot the (conventional) data
plt.figure(figsize=(15, 6))
plt.scatter(daily_births['Date'], daily_births['Num_Births'])
plt.title("Number of Daily Births in Quebec, Canada ")
plt.xlabel("Date")
plt.ylabel("Number of Births")
plt.show()
```



The first time index of my choice is **Monthly** and the aggregation method of choice will be the **Total (Sum)**

In [39]:
```
# Accumulate the data to a monthly level and will group by year and month
db_months = daily_births.groupby(['Year', 'Month'])['Num_Births'].sum().reset_index()
db_months
```

Out[39]:

	Year	Month	Num_Births
0	1977	1	8000
1	1977	2	7446
2	1977	3	8662
3	1977	4	8477
4	1977	5	8683
...	...	...	...
163	1990	8	8531
164	1990	9	8480
165	1990	10	8716
166	1990	11	7455
167	1990	12	7673

168 rows × 3 columns

In [39]:
```
# Create a monthly index for the series
db_months['Date'] = pd.to_datetime(db_months['Year'].astype(str) + '-' + db_months['Month'].astype(str), format="%Y-%m")
db_ts = pd.Series(db_months['Num_Births'].values, index=db_months['Date'])
db_ts.index.freq = 'MS'
```

Plot the Monthly Series

In [40]:
```
#Plot the monthly series
plt.figure(figsize=(15, 6))
db_ts.plot()
plt.title("Number of Daily Births in Months (Time Series)")
plt.xlabel("Date")
plt.ylabel("Number of Births")
plt.show()
```

The second time index of my choice is **Yearly** and the aggregation method of choice will be the **Mean (Average)**

In [57]:
```
# Accumulate the data to a monthly level and will group by year and month
db_yr = daily_births.groupby(['Year', 'Month'])['Num_Births'].mean().reset_index()
db_yr
```

Out[57]:

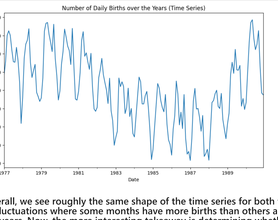
	Year	Month	Num_Births
0	1977	1	258.064516
1	1977	2	265.928571
2	1977	3	280.064516
3	1977	4	282.566667
4	1977	5	280.096774
...	...	...	...
163	1990	8	275.193548
164	1990	9	282.666667
165	1990	10	281.096774
166	1990	11	248.500000
167	1990	12	247.516129

168 rows × 3 columns

In [59]:
```
# Create a yearly index for the series
db_yr['Date'] = pd.to_datetime(db_yr['Year'].astype(str) + '-' + db_yr['Month'].astype(str), format="%Y-%m")
db_ts_2 = pd.Series(db_yr['Num_Births'].values, index=db_yr['Date'])
db_ts_2.index.freq = 'MS'
```

Plot the Yearly Time Series

In [52]:
```
#Plot the yearly series
plt.figure(figsize=(15, 6))
db_ts_2.plot()
plt.title("Number of Daily Births over the Years (Time Series)")
plt.xlabel("Date")
plt.ylabel("Number of Births")
plt.show()
```



Overall, we see roughly the same shape of the time series for both time indexes and the different aggregation methods. There is a lot of fluctuations where some months have more births than others and some months have less births, the same narrative can be said for years. Now, the more interesting takeaway is determining whether what we are seeing in our time series plots having a *cyclical* or *trend* component. This would require further analysis and research to find possible explanations for determing the component, but what is certain is that in Quebec, Canada in this specific time period was facing less births over time due to possible reasons like low fertitility rates, politics, teenage pregenancy, aging population and etc.