Visualization

In this lesson, we will discuss common plots to visualize data using Matplotlib and Seaborn. Seaborn works on top of Matplotlib and you will need to import both packages in most of the cases.

Reference:

- Seaborn Tutorial
- Matplotlib Tutorial

First, let's import the necessary packages in this notebook.

In [1]:

```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import numpy as np
```

In this notebook, we will still work with HDB resale price dataset to illustrate some visualization we can use. So let's import the dataset.

In [2]:

```
file_url = 'https://www.dropbox.com/s/jz8ck0obu9u1rng/resale-flat-prices-based-on-registration-date-from-jan-2017-onwards.csv?raw=1'
df = pd.read_csv(file_url)
df
```

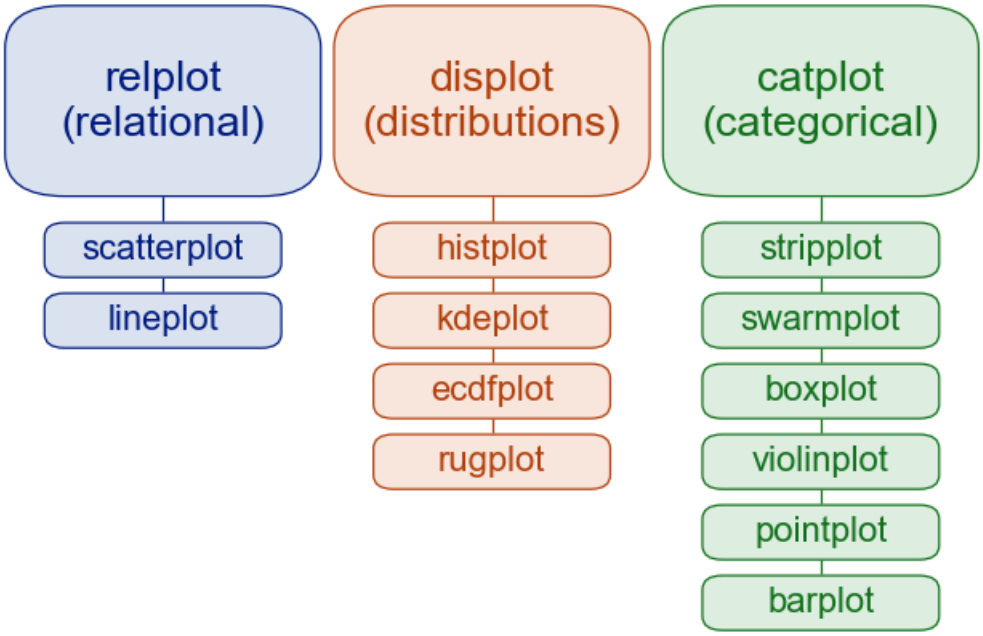
Out[2]:

	month	town	flat_type	block	street_name	storey_range	floor_area_sqm	flat_model	lease_commence_date	remaining_le
0	2017-01	ANG MO KIO	2 ROOM	406	ANG MO KIO AVE 10	10 TO 12	44.0	Improved	1979	61 year mo
1	2017-01	ANG MO KIO	3 ROOM	108	ANG MO KIO AVE 4	01 TO 03	67.0	New Generation	1978	60 year mo
2	2017-01	ANG MO KIO	3 ROOM	602	ANG MO KIO AVE 5	01 TO 03	67.0	New Generation	1980	62 year mo
3	2017-01	ANG MO KIO	3 ROOM	465	ANG MO KIO AVE 10	04 TO 06	68.0	New Generation	1980	62 year m
4	2017-01	ANG MO KIO	3 ROOM	601	ANG MO KIO AVE 5	01 TO 03	67.0	New Generation	1980	62 year mo
...	...	...	...	...	...	...	...	...	...	
95853	2021-04	YISHUN	EXECUTIVE	326	YISHUN RING RD	10 TO 12	146.0	Maisonette	1988	66 year mo
95854	2021-04	YISHUN	EXECUTIVE	360	YISHUN RING RD	04 TO 06	146.0	Maisonette	1988	66 year mo
95855	2021-04	YISHUN	EXECUTIVE	326	YISHUN RING RD	10 TO 12	146.0	Maisonette	1988	66 year mo
95856	2021-04	YISHUN	EXECUTIVE	355	YISHUN RING RD	10 TO 12	146.0	Maisonette	1988	66 year mo
95857	2021-04	YISHUN	EXECUTIVE	277	YISHUN ST 22	04 TO 06	146.0	Maisonette	1985	63 year mo

95858 rows × 11 columns

Categories of Plots

There are different categories of plot in Seaborn packages as shown in Seaborn documentation.



We can use either scatterplot or lineplot if we want to see relationship between two or more data. On the other hand, we have a few options to see distribution of data. The common one would be a histogram. The last category is categorical plot. We can use box plot, for example, to see the statistics of different categories. We will illustrate this more in the following sections.

Histogram and Boxplot

One of the first thing we may want to do in understanding the data is to see its distribution and its descriptive statistics. To do this, we can use `histplot` to show the histogram of the data and `boxplot` to show the five-number summary of the data.

Let's see the resale price in the area around Tampines. First, let's check what are the town listed in this data set.

In [3]:

```
np.unique(df['town'])
```

Out[3]:

```
array(['ANG MO KIO', 'BEDOK', 'BISHAN', 'BUKIT BATOK', 'BUKIT MERAH',
       'BUKIT PANJANG', 'BUKIT TIMAH', 'CENTRAL AREA', 'CHOA CHU KANG',
       'CLEMENTI', 'GEYLANG', 'HOUGANG', 'JURONG EAST', 'JURONG WEST',
       'KALLANG/WHAMPOA', 'MARINE PARADE', 'PASIR RIS', 'PUNGGOL',
       'QUEENSTOWN', 'SEMBAWANG', 'SENGKANG', 'SERANGOON', 'TAMPINES',
       'TOA PAYOH', 'WOODLANDS', 'YISHUN'], dtype=object)
```

Now, let's get the data for resale in Tampines only.

In [4]:

```
df_tampines = df.loc[df['town'] == 'TAMPINES',:]
df_tampines
```

Out[4]:

	month	town	flat_type	block	street_name	storey_range	floor_area_sqm	flat_model	lease_commence_date	remaini
917	2017-01	TAMPINES	2 ROOM	299A	TAMPINES ST 22	01 TO 03	45.0	Model A	2012	94
918	2017-01	TAMPINES	3 ROOM	403	TAMPINES ST 41	01 TO 03	60.0	Improved	1985	67
919	2017-01	TAMPINES	3 ROOM	802	TAMPINES AVE 4	04 TO 06	68.0	New Generation	1984	66
920	2017-01	TAMPINES	3 ROOM	410	TAMPINES ST 41	01 TO 03	69.0	Improved	1985	67
921	2017-01	TAMPINES	3 ROOM	462	TAMPINES ST 44	07 TO 09	64.0	Simplified	1987	69

	month	town	flat_type	block	street_name	storey_range	floor_area_sqm	flat_model	lease_commence_date	remaini
95671	2021-04	TAMPINES	EXECUTIVE	495E	TAMPINES ST 43	04 TO 06	147.0	Apartment	1994	71
95672	2021-04	TAMPINES	EXECUTIVE	477	TAMPINES ST 43	04 TO 06	153.0	Apartment	1993	71
95673	2021-04	TAMPINES	EXECUTIVE	497J	TAMPINES ST 45	10 TO 12	139.0	Premium Apartment	1996	74
95674	2021-04	TAMPINES	EXECUTIVE	857	TAMPINES ST 83	01 TO 03	154.0	Maisonette	1988	
95675	2021-04	TAMPINES	MULTI-GENERATION	454	TAMPINES ST 42	01 TO 03	132.0	Multi Generation	1987	65

6392 rows × 11 columns

Now, we can plot its resale price distribution using `histplot`.

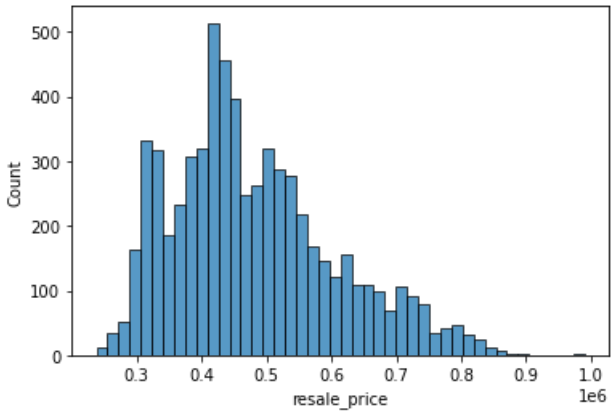
See documentation for histplot

In [5]:

```
sns.histplot(x='resale_price', data=df_tampines)
```

Out[5]:

```
<AxesSubplot:xlabel='resale_price', ylabel='Count'>
```



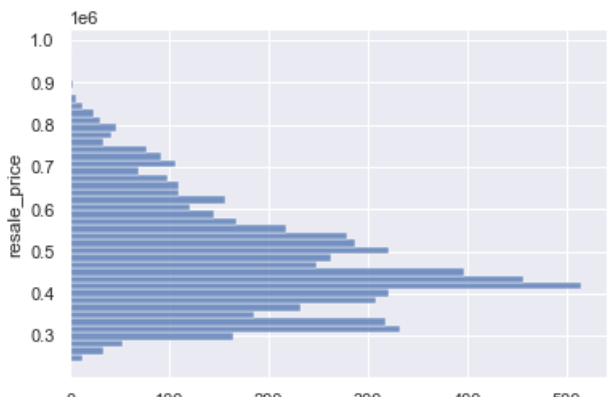
In the above plot, we use `df_tampines` as our data source and use `resale_price` column as our x-axis. We can change the plot if we want to show it vertically.

In [6]:

```
sns.set()
sns.histplot(y='resale_price', data=df_tampines)
```

Out[6]:

```
<AxesSubplot:xlabel='Count', ylabel='resale_price'>
```



Notice that the background changes. This is because we have called `sns.set()` which set Seaborn default setting instead of using Matplotlib's setting. For example, Matplotlib uses whitebackground and no grid. Seaborn by default displays some white grid on gray background.

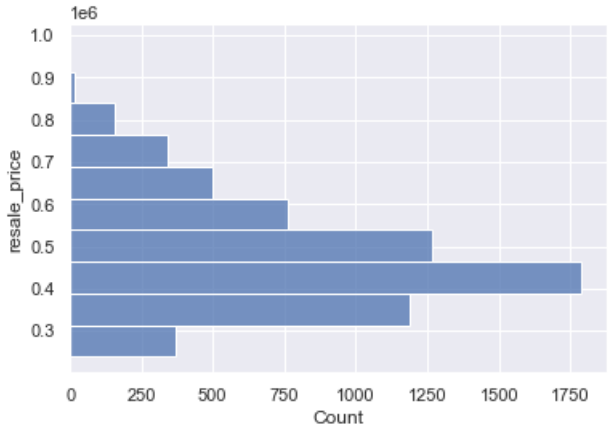
By default, the `bins` argument is `auto` and Seaborn will try to calculate how many bins should be used. But we can specify this manually.

In [7]:

```
sns.histplot(y='resale_price', data=df_tampines, bins=10)
```

Out[7]:

```
<AxesSubplot:xlabel='Count', ylabel='resale_price'>
```



We can see that majority of the sales of resale HDB in Tampines is priced at about \$400k to \$500k.

We can also use the `boxplot` to see some descriptive statistics of the data.

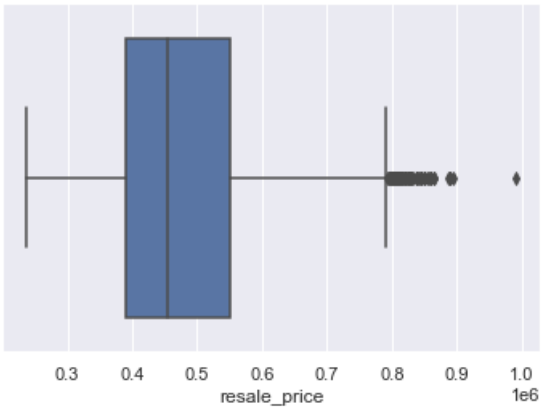
See documentation on boxplot

In [8]:

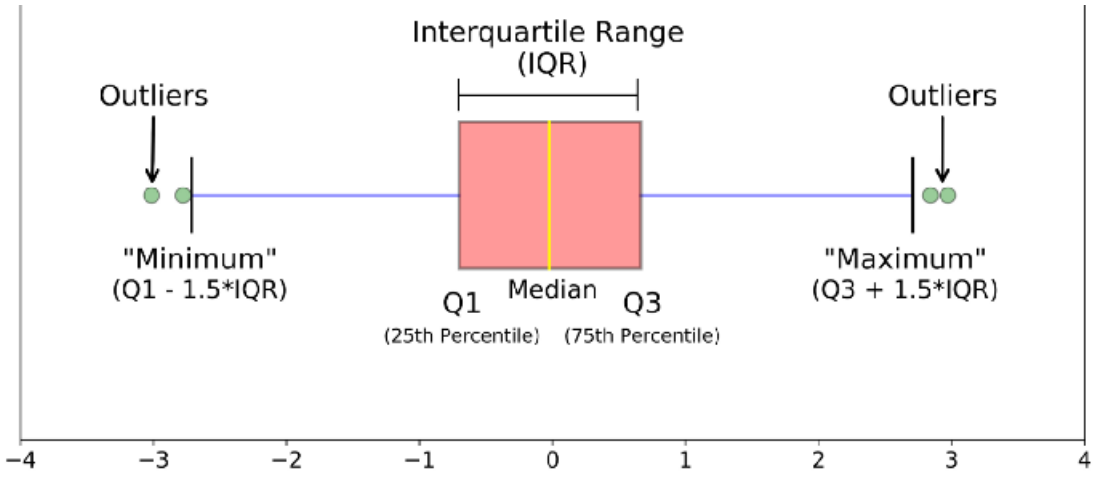
```
sns.boxplot(x='resale_price', data=df_tampines)
```

Out[8]:

```
<AxesSubplot:xlabel='resale_price'>
```



See Understanding Boxplot for more detail. But the figure in that website summarizes the different information given in a boxplot.



The box gives you the 25th percentile and the 75th percentile boundary. The line inside the box gives you the median of the data. As we can see the median is about \$400k to \$500k. The difference between the 75th percentile (Q3) and the 25th percentile (Q1) is called the *Interquartile Range* (IQR). This definition is needed to understand what defines **outliers**. The minimum and the maximum here is not the minimum and the maximum value in the data, but rather is capped at
$$min = Q1 - 1.5\times IQR$$$$max = Q3 + 1.5\times IQR$$

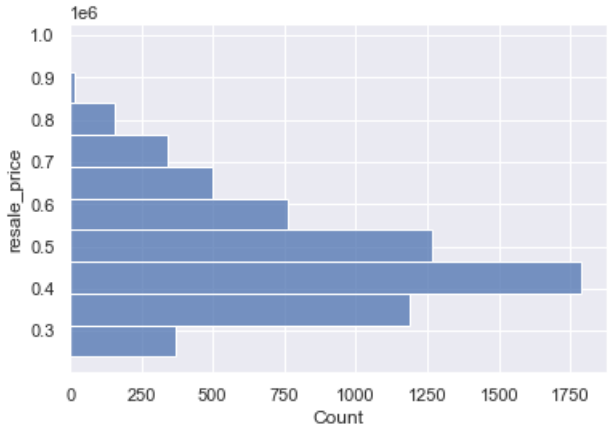
Anything below or above these "minimum" and "maximum" are considered an outlier in the box plot. In the figure above, for example, we have quite a number of outliers on the high end of the resale price.

Modifying Labels and Titles

Since Seaborn is built on top of Matplotlib, we can use some of Matplotlib functions to change the figure's labels and title. For example, we can change the histogram's plot x and y labels and its titles using `plt.xlabel()`, `plt.ylabel()`, and `plt.title`. You can access these Matplotlib's functions by first storing the output of your Seaborn plot.

In [31]:

```
myplot = sns.histplot(y='resale_price', data=df_tampines, bins=10)
```



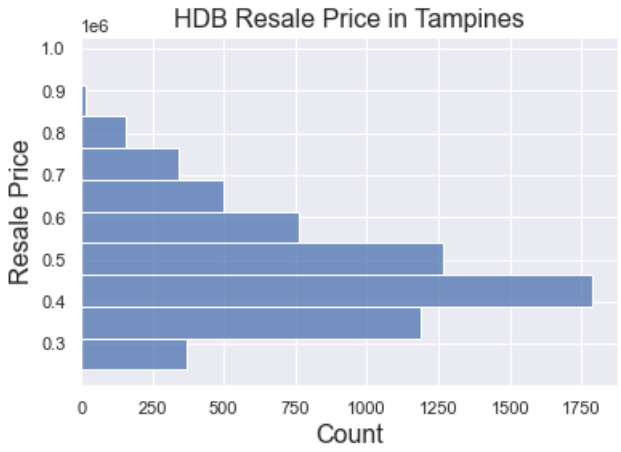
Once you obtain the handle, you can call Matplotlib's function by adding the word `set_` in front of it. For example, if the Matplotlib's function is `plt.xlabel()`, you call it as `myplot.set_xlabel()`. See the code below.

In [34]:

```
myplot = sns.histplot(y='resale_price', data=df_tampines, bins=10)
myplot.set_xlabel('Count', fontsize=16)
myplot.set_ylabel('Resale Price', fontsize=16)
myplot.set_title('HDB Resale Price in Tampines', fontsize=16)
```

Out[34]:

```
Text(0.5, 1.0, 'HDB Resale Price in Tampines')
```



Notice now that the plot has a title and both the x and y label has changed.

The above plot will be much easier if we plots in thousands of dollars. So let's create a new column of resale price in \$1000.

In [14]:

```
df_tampines['resale_price_1000'] = df_tampines['resale_price'].apply(lambda price: price/1000)
df_tampines['resale_price_1000'].describe()
```

```
<ipython-input-14-ca31af08fd34>:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  df_tampines['resale_price_1000'] = df_tampines['resale_price'].apply(lambda price: price/1000)
```

Out[14]:

```
count    6392.000000
mean      479.670371
std       125.569977
min       238.000000
25%       390.000000
50%       455.000000
75%       550.000000
max       990.000000
Name: resale_price_1000, dtype: float64
```

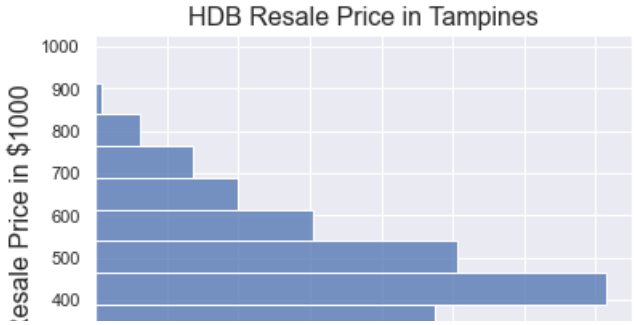
Now, let's plot it one more time.

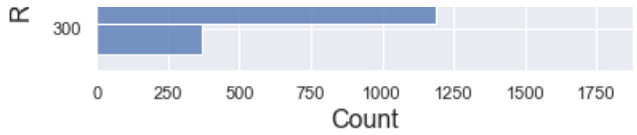
In [35]:

```
myplot = sns.histplot(y='resale_price_1000', data=df_tampines, bins=10)
myplot.set_xlabel('Count', fontsize=16)
myplot.set_ylabel('Resale Price in $1000', fontsize=16)
myplot.set_title('HDB Resale Price in Tampines', fontsize=16)
```

Out[35]:

```
Text(0.5, 1.0, 'HDB Resale Price in Tampines')
```





Using Hue

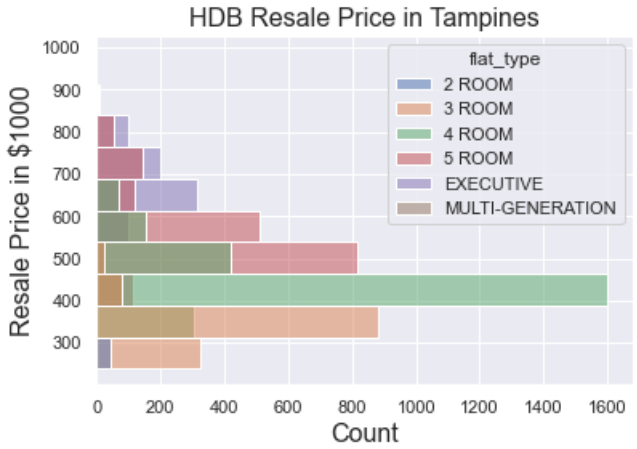
Seaborn make it easy to plot the same data and colour those data depending on another data. For example, we can see the distribution of the resale price according to the room number or the storey range. Seaborn has an argument called `hue` to specify which data column you want to use to colour this.

In [45]:

```
myplot = sns.histplot(y='resale_price_1000', hue='flat_type', data=df_tampines, bins=10)
myplot.set_xlabel('Count', fontsize=16)
myplot.set_ylabel('Resale Price in $1000', fontsize=16)
myplot.set_title('HDB Resale Price in Tampines', fontsize=16)
```

Out[45]:

```
Text(0.5, 1.0, 'HDB Resale Price in Tampines')
```



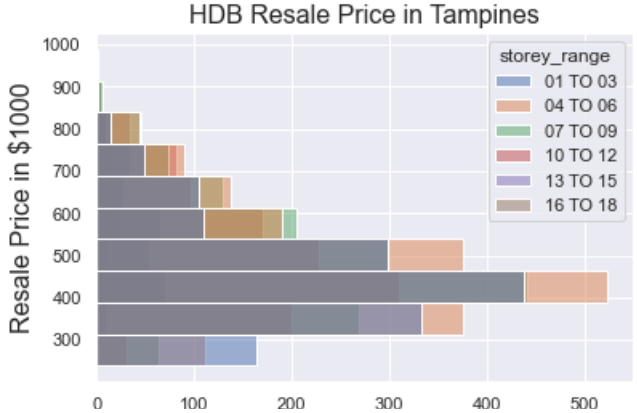
So we can see from the distribution that 4-room flats in Tampines contributes roughly the largest sales. We can also see that 4-room flat resale price is around the median of the all the resale flats in this area.

In [46]:

```
myplot = sns.histplot(y='resale_price_1000', hue='storey_range', data=df_tampines, bins=10)
myplot.set_xlabel('Count', fontsize=16)
myplot.set_ylabel('Resale Price in $1000', fontsize=16)
myplot.set_title('HDB Resale Price in Tampines', fontsize=16)
```

Out[46]:

```
Text(0.5, 1.0, 'HDB Resale Price in Tampines')
```



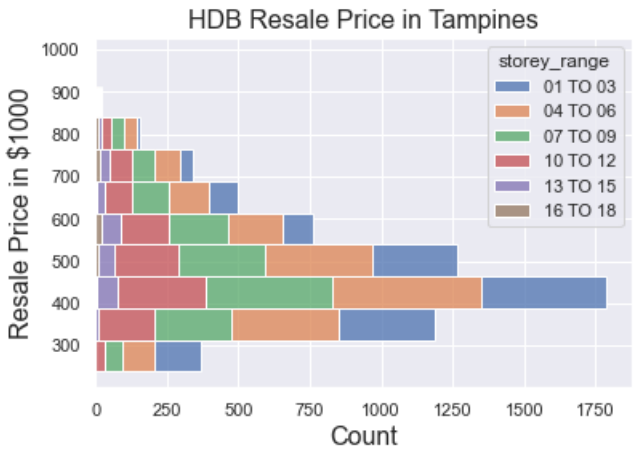
The above colouring is not so obvious because they are on top of one another, one way is to change the settings in such a way that it is stacked. We can do this by setting the `multiple` argument for the case when there are multiple data in the same area.

In [47]:

```
myplot = sns.histplot(y='resale_price_1000', hue='storey_range',
                      multiple='stack',
                      data=df_tampines, bins=10)
myplot.set_xlabel('Count', fontsize=16)
myplot.set_ylabel('Resale Price in $1000', fontsize=16)
myplot.set_title('HDB Resale Price in Tampines', fontsize=16)
```

Out[47]:

```
Text(0.5, 1.0, 'HDB Resale Price in Tampines')
```



Scatter Plot and Line Plot

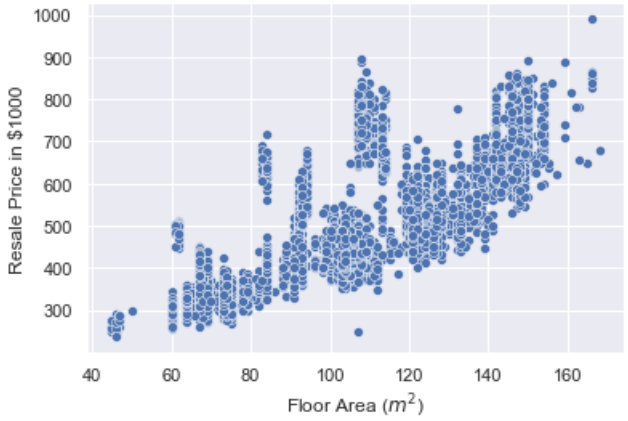
We use scatter plot and line plot to visualize relationship between two or more data. For example, we can plot the floor area and resale price to see if there is any relationship.

In [89]:

```
myplot = sns.scatterplot(x='floor_area_sqm', y='resale_price_1000', data=df_tampines)
myplot.set_xlabel('Floor Area ($m^2$)')
myplot.set_ylabel('Resale Price in $1000')
```

Out[89]:

```
Text(0, 0.5, 'Resale Price in $1000')
```



As we can see from the plot above, that the price tend to increase with the increase in floor area. You can again use the `hue`

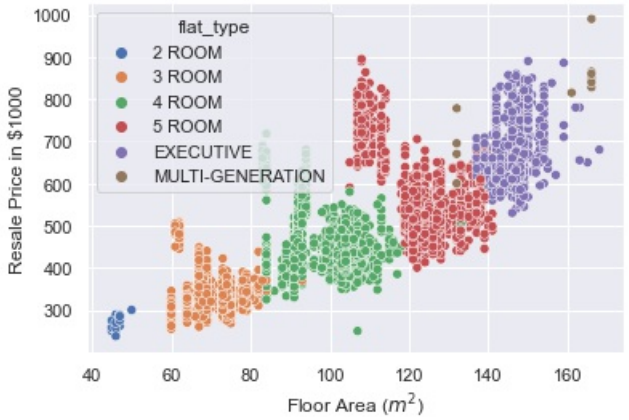
argument to see any category in the plot.

In [91]:

```
myplot = sns.scatterplot(x='floor_area_sqm', y='resale_price_1000',
                         hue='flat_type',
                         data=df_tampines)
myplot.set_xlabel('Floor Area ($m^2$)')
myplot.set_ylabel('Resale Price in $1000')
```

Out[91]:

```
Text(0, 0.5, 'Resale Price in $1000')
```



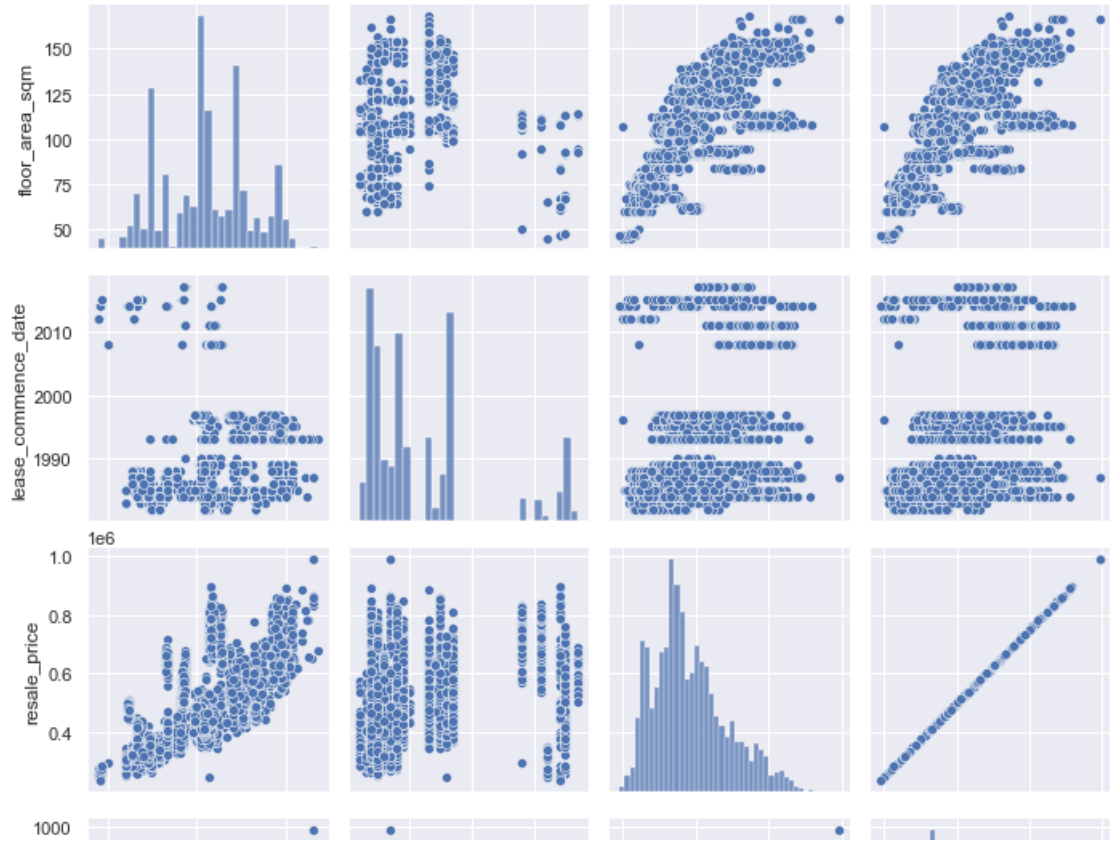
We can see that flat type in a way also has relationship with the floor area.

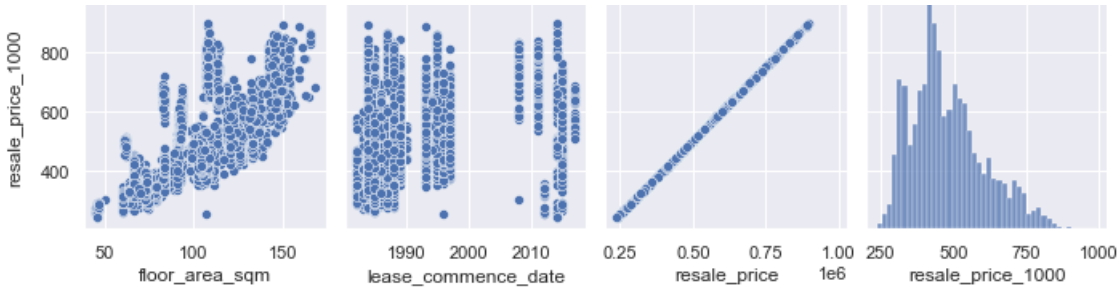
Pair Plot

One useful plot is called Pair Plot in Seaborn where it plots the relationship on multiple data columns.

In [92]:

```
myplot = sns.pairplot(data=df_tampines)
```





The above plots immediately plot different scatter plots and histogram in a matrix form. The diagonal of the plot shows the histogram of that column data. The rest of the cell shows you the scatter plot of two columns in the data frame. From these, we can quickly see the relationship between different columns in the data frame.