San Francisco Building Permits

> 5 years and 200k building permits

Introduction

> A building permit is an official approval document issued by a government agency that allows you or your contractor to proceed with a construction for remodeling project on one's property. Each city or country has its own officerelated to buildings, that can do multiple functions like issuing permits, inspecting buildings to enforce safety measures, modifying rules to accommodate needs of the growing population and so on. For the city of San Francisco, permit issuing is taken care by San Francisco government.

This dataset related to building permits is downloaded from kaggle. There are some certain features of the dataset which are given below :-

> - The data getting updated by San Francisco Open Data portal every Saturday.
> - There are 43 columns with 1,98,900 rows wuth we can do different sort of things according one's own creativity.
> - The data's column described in codes which are explained in different excel file which to access.
> - Address is divided in different parts from block to street name suffix which make it easy to get information through it.
> - Supervisor District and permit are assigned in form of numerical data that makes for both us and python to go through easily then check by another excel sheet about designated value.
> - Dates for every stage is given such as filed, issued, completed and expiration date.

Downloading and Importing the Dataset

1. Downloading Dataset

There are saveral ways through which we can download the datasets from particular website such as by using **urlretrieve library** or by **opendatasets library** but as I have manually downloaded my dataset from link that is given below, I am going to directlyimport the dataset by using **pandas**.

link:- https://www.kaggle.com/datasets/aparnashastry/building-permit-applications-data (https://www.kaggle.com/datasets/aparnashastry/building-permit-applications-data)

In [1]:

```python
import os #to know the files that are downloaded
```

In [2]:

```python
os.listdir('building dataset')
```

Out[2]:

```
['Building_Permits.csv', 'DataDictionaryBuildingPermit.xlsx']
```

- DataDictionaryBuildingPermit.xlsx - The list of shortcodes for each column main dataset.
- Building_permits.csv - The full information about building permits according to their ids.

2. Importing Dataset

> Now, I will import csv file and xlsx file by **Pandas** for better tabular visualization then I take out columns from the pandas data frame then limit our field of analysis to certain range of data provided.

In [3]:

```python
import pandas as pd #to import dataset in tabular form and clean it
import numpy as np #to do the complex calculations
```

In [4]:

```python
building_raw_df = pd.read_csv('building dataset/Building_Permits.csv')
```

```
C:\Users\ABC\AppData\Local\Temp\ipykernel_3764\828121404.py:1: DtypeWarning:
Columns (22,32) have mixed types. Specify dtype option on import or set low_
memory=False.
  building_raw_df = pd.read_csv('building dataset/Building_Permits.csv')
```

In [5]:

```
building_raw_df
```

Out[5]:

	Permit Number	Permit Type	Permit Type Definition	Permit Creation Date	Block	Lot	Street Number	Street Number Suffix	Stree Name
0	201505065519	4	sign - erect	05/06/2015	0326	023	140	NaN	Ellis
1	201604195146	4	sign - erect	04/19/2016	0306	007	440	NaN	Gear
2	201605278609	3	additions alterations or repairs	05/27/2016	0595	203	1647	NaN	Pacifi
3	201611072166	8	otc alterations permit	11/07/2016	0156	011	1230	NaN	Pacifi
4	201611283529	6	demolitions	11/28/2016	0342	001	950	NaN	Marke
...	...	...	...	...	...	...	...	...	..
198895	M862628	8	otc alterations permit	12/05/2017	0113	017A	1228	NaN	Montgomer
198896	201712055595	8	otc alterations permit	12/05/2017	0271	014	580	NaN	Bush
198897	M863507	8	otc alterations permit	12/06/2017	4318	019	1568	NaN	Indiana
198898	M863747	8	otc alterations permit	12/06/2017	0298	029	795	NaN	Sutte
198899	M864287	8	otc alterations permit	12/07/2017	0160	006	838	NaN	Pacifi

198900 rows × 43 columns

```
In [6]:  building_raw_df.columns
```

```
Out[6]: Index(['Permit Number', 'Permit Type', 'Permit Type Definition',
               'Permit Creation Date', 'Block', 'Lot', 'Street Number',
               'Street Number Suffix', 'Street Name', 'Street Suffix', 'Unit',
               'Unit Suffix', 'Description', 'Current Status', 'Current Status Dat
        e',
               'Filed Date', 'Issued Date', 'Completed Date',
               'First Construction Document Date', 'Structural Notification',
               'Number of Existing Stories', 'Number of Proposed Stories',
               'Voluntary Soft-Story Retrofit', 'Fire Only Permit',
               'Permit Expiration Date', 'Estimated Cost', 'Revised Cost',
               'Existing Use', 'Existing Units', 'Proposed Use', 'Proposed Units',
               'Plansets', 'TIDF Compliance', 'Existing Construction Type',
               'Existing Construction Type Description', 'Proposed Construction Typ
        e',
               'Proposed Construction Type Description', 'Site Permit',
               'Supervisor District', 'Neighborhoods - Analysis Boundaries', 'Zipcod
        e',
               'Location', 'Record ID'],
              dtype='object')
```

> Now, let's get the overview of dataset that I have just imported and check the columns that I can work with efficiently. Firstly, load the other dataset that is giving information about the columns of main data.

```
In [7]:  data_dictionary_df = pd.read_excel('building dataset/DataDictionaryBuildi
```

In [8]:

```
data_dictionary_df
```

Out[8]:

	Column name	Description
Sl No		
1.0	Permit Number	Number assigned while filing
2.0	Permit Type	Type of the permit represented numerically.
3.0	Permit Type Definition	Description of the Permit type, for example\n ...
4.0	Permit Creation Date	Date on which permit created, later than \nor ...
5.0	Block	Related to address
6.0	Lot	Related to address
7.0	Street Number	Related to address
8.0	Street Number Suffix	Related to address
9.0	Street Name	Related to address
10.0	Street Name Suffix	Related to address
11.0	Unit	Unit of a building
12.0	Unit suffix	Suffix if any, for the unit
13.0	Description	Details about purpose of the permit.\n Example...
14.0	Current Status	Current status of the permit application.
15.0	Current Status Date	Date at which current status was entered
16.0	Filed Date	Filed date for the permit
17.0	Issued Date	Issued date for the permit
18.0	Completed Date	The date on which project was completed, \napp...
19.0	First Construction Document Date	Date on which construction was documented
20.0	Structural Notification	Notification to meet some legal need, given or...
21.0	Number of Existing Stories	Number of existing stories in the building. \n...
22.0	Number of Proposed Stories	Number of proposed stories for the constructio...
23.0	Voluntary Soft-Story \nRetrofit	Soft story to meet earth quake regulations
24.0	Fire Only Permit	Fire hazard prevention related permit
25.0	Permit Expiration Date	Expiration date related to issued permit.
26.0	Estimated Cost	Initial estimation of the cost of the project
27.0	Revised Cost	Revised estimation of the cost of the project
28.0	Existing Use	Existing use of the building
29.0	Existing Units	Existing number of units
30.0	Proposed Use	Proposed use of the building
31.0	Proposed Units	Proposed number of units
32.0	Plansets	Plan representation indicating the general des...

Sl No	Column name	Description
33.0	TIDF Compliance	TIDF compliant or not, this is a new legal req...
34.0	Existing Construction Type	Construction type, existing,as categories \nre...
35.0	Existing Construction Type Description	Description of the above, for example, \nwood ...
36.0	Proposed Construction Type	Construction type, proposed, as categories\n r...
37.0	Proposed Construction Type Description	Description of the above
38.0	Site Permit	Permit for site
39.0	Supervisor District	Supervisor District to which the building loca...
40.0	Neighborhoods - Analysis Boundaries	Neighborhood to which the building location be...
41.0	Zipcode	Zipcode of building address
42.0	Location	Location in latitude, longitude pair.
43.0	Record ID	Some ID, not useful for this

As I have imported the dataset and get the information about the columns that they provide. Now, I am going to select the columns which I am going to use in our further section.

Data Preparation and Cleaning

Now, firstly I am going to limit our view over some certain fields to get the best information out of this dataset.

In [9]:

```
selected_columns = ['Permit Type',
                     'Permit Type Definition',
                    'Permit Creation Date',
                    'Street Name',
                    'Street Suffix',
                    'Description',
                    'Filed Date',
                    'Issued Date',
                     'Permit Expiration Date',
                    'Estimated Cost',
                    'Revised Cost',
                    'Existing Construction Type',
                    'Existing Construction Type Description',
                    'Proposed Construction Type',
                    'Proposed Construction Type Description',
                    'Supervisor District']
```

In [10]:

```
len(selected_columns)
```

Out[10]:

```
16
```

I have selected **16 columns** to limits my area of analysis and get the useful information that I think I can extract from the given information.

I will now extract a copy of the data from these columns into a new data frame **building_df**. Hence, data can be modified further without having affect on original dataset.

In [11]:

```
building_df = building_raw_df[selected_columns].copy()
building_df
```

Out[11]:

	Permit Type	Permit Type Definition	Permit Creation Date	Street Name	Street Suffix	Description	Filed Date	
0	4	sign - erect	05/06/2015	Ellis	St	ground fl facade: to erect illuminated, electr...	05/06/2015	11/
1	4	sign - erect	04/19/2016	Geary	St	remove (e) awning and associated signs.	04/19/2016	08/
2	3	additions alterations or repairs	05/27/2016	Pacific	Av	installation of separating wall	05/27/2016	
3	8	otc alterations permit	11/07/2016	Pacific	Av	repair dryrot & stucco at front of bldg.	11/07/2016	07/
4	6	demolitions	11/28/2016	Market	St	demolish retail/office/commercial 3-story buil...	11/28/2016	12/
...	...	...	...	...	...	...	...	
198895	8	otc alterations permit	12/05/2017	Montgomery	St	street space	12/05/2017	12/
198896	8	otc alterations permit	12/05/2017	Bush	St	fire alarm upgrade ref 201704123852	12/05/2017	12/
198897	8	otc alterations permit	12/06/2017	Indiana	St	street space	12/06/2017	12/
198898	8	otc alterations permit	12/06/2017	Sutter	St	street space permit	12/06/2017	12/
198899	8	otc alterations permit	12/07/2017	Pacific	Av	street space permit	12/07/2017	12/

198900 rows × 16 columns

Now, as we have selected the the columns from which we are going to extract information. So, let's get started with the description.

In [12]:

```
building_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 198900 entries, 0 to 198899
Data columns (total 16 columns):
 #   Column                                  Non-Null Count   Dtype  
---  ------                                  --------------   -----  
 0   Permit Type                             198900 non-null  int64  
 1   Permit Type Definition                  198900 non-null  object 
 2   Permit Creation Date                    198900 non-null  object 
 3   Street Name                             198900 non-null  object 
 4   Street Suffix                           196132 non-null  object 
 5   Description                             198610 non-null  object 
 6   Filed Date                              198900 non-null  object 
 7   Issued Date                             183960 non-null  object 
 8   Permit Expiration Date                  147020 non-null  object 
 9   Estimated Cost                          160834 non-null  float64
 10  Revised Cost                            192834 non-null  float64
 11  Existing Construction Type              155534 non-null  float64
 12  Existing Construction Type Description  155534 non-null  object 
 13  Proposed Construction Type              155738 non-null  float64
 14  Proposed Construction Type Description  155738 non-null  object 
 15  Supervisor District                     197183 non-null  float64
dtypes: float64(5), int64(1), object(10)
memory usage: 16.7+ MB
```

Most of the columns have data type object, either because they contain values of differenttypes or empty values(Nan). It appears that every column contains some empty values since the Non-Null count for every column is lower than total number of rows(198,900).

But if we analyze the the columns there are some information that is provided in non-numerical way and hence, it is in object datatype. And rest of the columns that have numeric values are in float or integer data types. Hence, there is no need to manually change the non-numeric values to Nan because it is already done in data set and thats the reason inspite less non-null values than the total number of rows, these columns have float or int datatype.

In [13]:

```
missing_value_count = building_df.isnull().sum()
missing_value_count

total_cells = np.product(building_df.shape)
total_missing = missing_value_count.sum()

percent_missing = (total_missing/total_cells)*100
print(percent_missing)
```

```
9.074377828054299
```

Above, I have just checked the number of null values that are pesent in the data set and there are around 9%.

In [14]:

```
building_df.describe()
```

Out[14]:

	Permit Type	Estimated Cost	Revised Cost	Existing Construction Type	Proposed Construction Type	Supervisor District
count	198900.000000	1.608340e+05	1.928340e+05	155534.000000	155738.000000	197183.000000
mean	7.522323	1.689554e+05	1.328562e+05	4.072878	4.089529	5.538403
std	1.457451	3.630386e+06	3.584903e+06	1.585756	1.578766	2.887041
min	1.000000	1.000000e+00	0.000000e+00	1.000000	1.000000	1.000000
25%	8.000000	3.300000e+03	1.000000e+00	3.000000	3.000000	3.000000
50%	8.000000	1.100000e+04	7.000000e+03	5.000000	5.000000	6.000000
75%	8.000000	3.500000e+04	2.870750e+04	5.000000	5.000000	8.000000
max	8.000000	5.379586e+08	7.805000e+08	5.000000	5.000000	11.000000

There seems to be a problem with the **Estimated Cost** and **Revised Cost** , as the minimum estimated cost is 1.0 while, minimum revised cost is 0.0 and 25% rows have 1.0 revised cost. We will simple fix it by ignoring the rows that hold this value by using **.drop** function.

In [30]:

```
building_df.drop(building_df[building_df['Estimated Cost'] <10].index, in
building_df.drop(building_df[building_df['Revised Cost'] <10].index, inpl
```

As, we I have simply removed the values from the selected column that does not fit right in the dataset in accordance with others.

In [33]:

```
building_df['Street Suffix'].value_counts()
```

Out[33]:

```
St    95725
Av    32343
Wy     2731
Bl     2726
Dr     2724
Tr     1053
Ct      560
Pl      391
Rd      307
Ln      288
Hy      187
Pz      157
Pk       96
Cr       80
Al       58
Wk        4
Rw        3
So        2
No        2
Sw        1
Hl        1
Name: Street Suffix, dtype: int64
```

In the above line of code I just checked the adress of building related to their street to ensure that they are not mixed up. Now, we are ready to do our analysis as there no mistake in street suffix and other data.

Exploratory Analysis and Visualization

Before answering some questions lets extract some information further to understand the way information is provided.

Let's import the **matplotlib** and **seaborn** library.

In [36]:

```
import seaborn as sns
import matplotlib
import matplotlib.pyplot as plt

sns.set_style('darkgrid')
```

Permit Type and Permit Type Definition

Let's have a look over the permit types that are given and check which type permit is most taken by buildings.

In [35]:

```
building_df[['Permit Type','Permit Type Definition']]
```

Out[35]:

	Permit Type	Permit Type Definition
0	4	sign - erect
2	3	additions alterations or repairs
3	8	otc alterations permit
4	6	demolitions
5	8	otc alterations permit
...	...	...
198890	3	additions alterations or repairs
198892	8	otc alterations permit
198893	8	otc alterations permit
198894	8	otc alterations permit
198896	8	otc alterations permit

141407 rows × 2 columns

In [89]:

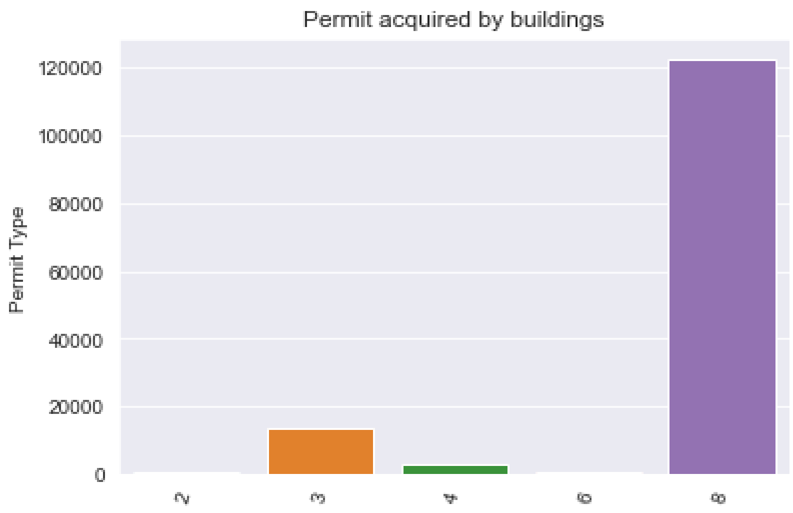
```
permit = building_df['Permit Type'].value_counts().head(5)
permit
```

Out[89]:

```
8    122395
3     13725
4      2871
2       888
6       599
Name: Permit Type, dtype: int64
```

In [90]:

```
plt.xticks(rotation =75)
plt.title('Permit acquired by buildings')
sns.barplot(x = permit.index, y = permit );
```



> As we can clearly almost all of the buildings acquire permit type **8** which is an **otc alteration permit** followed by permit type **3** which is an **addition alterations or repairs**.

(OTC) Permit - Over The Counter Permit

The Department of Building Inspection reviews every building permit application for life safety and building code compliance. Officials provide over-the-counter review for simple permit application. It works in 5 simple steps:-

- Review the list on website of gonvernment to see if project qualifies for OTC review.
- Check if any plan is needed.
- Go through required form for certain project.
- Follow instructions to fill out the forms for that certain project.
- Visit Permit Center.

Perdiction made on the basis of selected data

> I am going to use correlation function to know the relation between the columns of data set and try to figure out if I can make any perdiction.

In [46]:

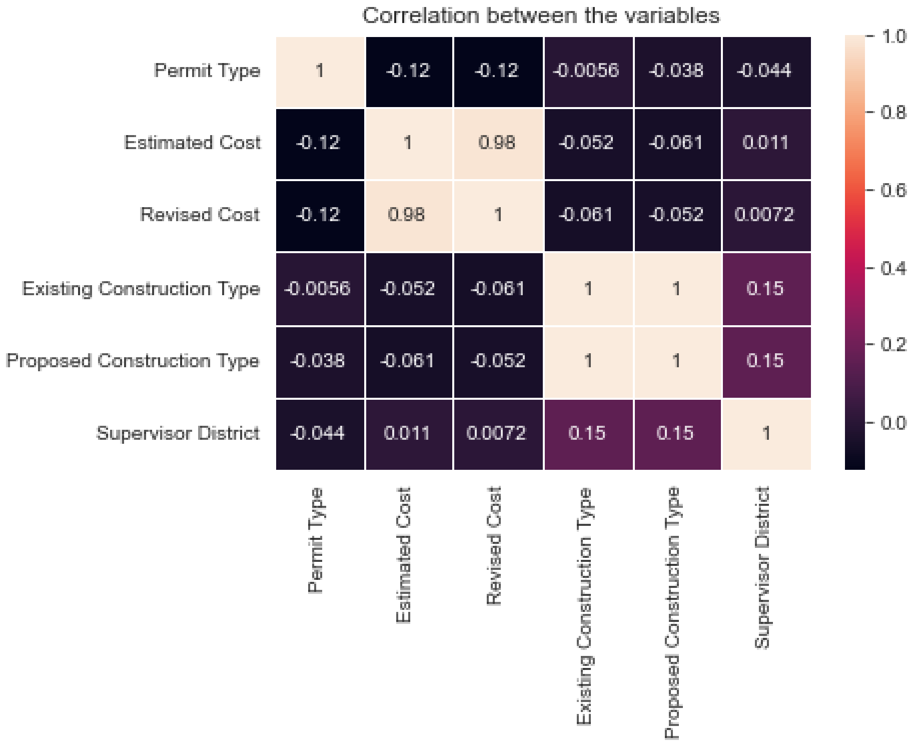
```python
building_df.corr()
```

Out[46]:

	Permit Type	Estimated Cost	Revised Cost	Existing Construction Type	Proposed Construction Type	Supervisor District
Permit Type	1.000000	-0.123945	-0.121594	-0.005608	-0.038195	-0.043730
Estimated Cost	-0.123945	1.000000	0.979829	-0.051697	-0.060934	0.010588
Revised Cost	-0.121594	0.979829	1.000000	-0.061429	-0.052310	0.007222
Existing Construction Type	-0.005608	-0.051697	-0.061429	1.000000	0.999033	0.152595
Proposed Construction Type	-0.038195	-0.060934	-0.052310	0.999033	1.000000	0.149904
Supervisor District	-0.043730	0.010588	0.007222	0.152595	0.149904	1.000000

In [48]:

```python
sns.heatmap(building_df.corr(), linewidths = 1, annot = True)
plt.title('Correlation between the variables');
```



From heatmap we can account that **revised cost** and **estimated cost** somehow depended on each other. Hence, with **increase** in **revised cost** would likely also **increase estimated cost** or vise-versa.

Interestingly, **Existing construction type** and **porposed construction type** entirely depend on each other.

File Issued and Expiration Date

Firstly, I will convert the the data type of dates which are object right now into datetime data type and extract information in form of graphs.

In [55]:

```
building_df['Issued Date'] = pd.to_datetime(building_df['Issued Date'])
building_df['Permit Expiration Date'] = pd.to_datetime(building_df['Permi
```

Now, i will introduce new column from these dates in form of month and weekdays. So, that I can get the time when most of the permits issues and expires.

In [57]:

```
# For permit issued date
building_df['Issued_month'] = pd.DatetimeIndex(building_df['Issued Date']
building_df['Issued_weekday'] = pd.DatetimeIndex(building_df['Issued Date
# For permit expirate date
building_df['Expiration_month'] = pd.DatetimeIndex(building_df['Permit Ex
building_df['Expiration_weekday'] = pd.DatetimeIndex(building_df['Permit 
```

In [58]:

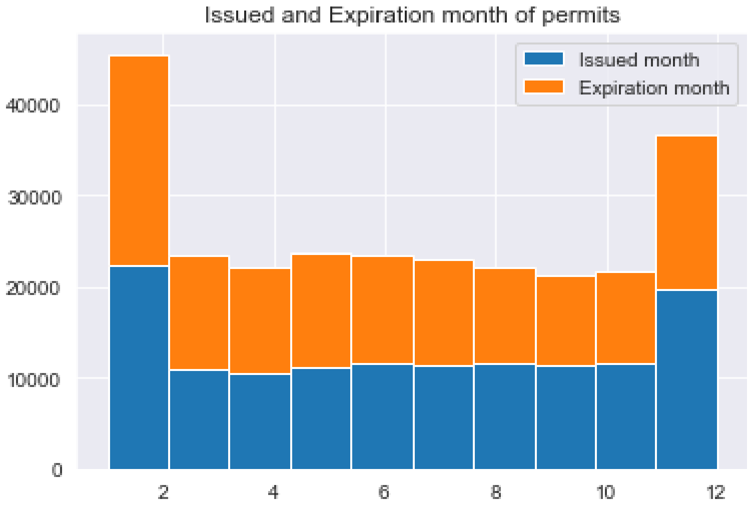
```
building_df.head(5)
```

Out[58]:

	Permit Type	Permit Type Definition	Permit Creation Date	Street Name	Street Suffix	Description	Filed Date	Issued Date	Pe Expir
0	4	sign - erect	05/06/2015	Ellis	St	ground fl facade: to erect illuminated, electr...	05/06/2015	2015-11-09	201
2	3	additions alterations or repairs	05/27/2016	Pacific	Av	installation of separating wall	05/27/2016	NaT	
3	8	otc alterations permit	11/07/2016	Pacific	Av	repair dryrot & stucco at front of bldg.	11/07/2016	2017-07-18	201
4	6	demolitions	11/28/2016	Market	St	demolish retail/office/commercial 3-story buil...	11/28/2016	2017-12-01	201
5	8	otc alterations permit	06/14/2017	Indiana	St	evac maps	06/14/2017	2017-07-06	201

In [65]:

```
plt.title('Issued and Expiration month of permits')
plt.hist([building_df['Issued_month'],building_df['Expiration_month']], s
plt.legend(['Issued month', 'Expiration month']);
```

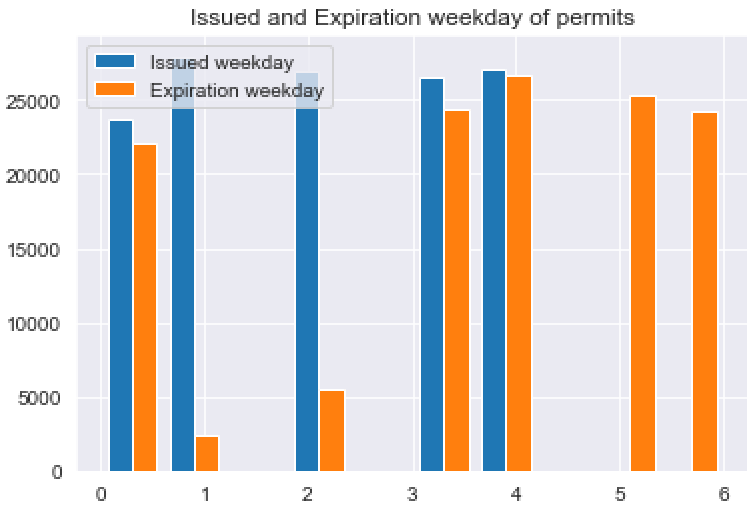


It is clear that most of the permits issues and expires on **first month** of the year then forthcoming with end of the year.

Now, i will take look over weekday data and then come for an conclusion.

In [66]:

```python
plt.title('Issued and Expiration weekday of permits')
plt.hist([building_df['Issued_weekday'], building_df['Expiration_weekday'
         stacked = False)
plt.legend(['Issued weekday', 'Expiration weekday']);
```



Most of the issued permits were on **Tuesday** and most of the expired permits were **Friday**.

From the analyzation it is clear that the best month for applying new permit is **April** since least number of permits issued and expired in this month, hence there will less crowd in the office at that time of period.

And in April month **Mondays** are the best because there are least permits that are issued on that day.

Note:- 5th and 6th day are not taken in account because Saturdays and Sundays are off for government offices.

Buildings Street Address

I will now take look over the addresses of the buildings in which street they reside.

In [75]:

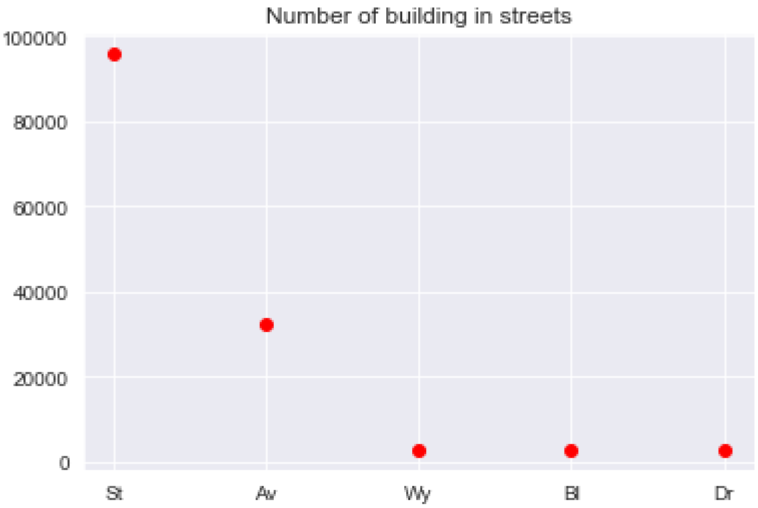
```
street = building_df['Street Suffix'].value_counts().head(5)
street
```

Out[75]:

```
St    95725
Av    32343
Wy     2731
Bl     2726
Dr     2724
Name: Street Suffix, dtype: int64
```

In [76]:

```
plt.plot(street.index, street, 'or' )
plt.title('Number of building in streets');
```



> There is too much difference in building counting from some certain streets. **St** have the most buildings that have acquired permits which shows that development of buildings are most in this part of the city.

Supervisor District

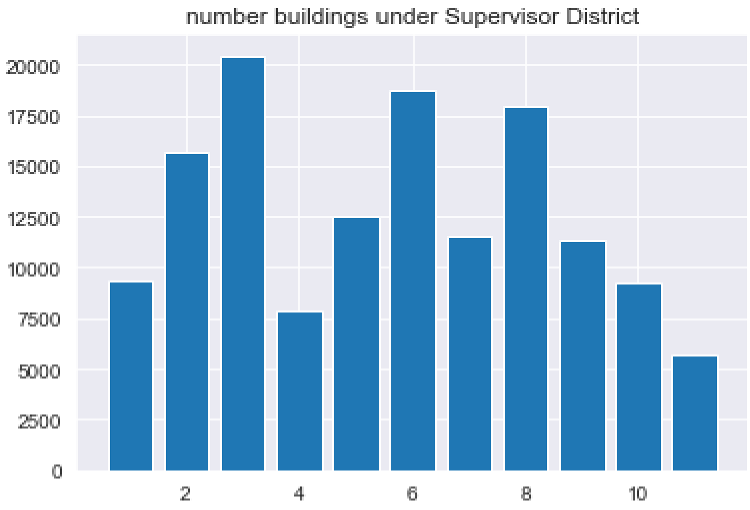
> From this we can get the information about the building's permit record that is there to every supervisor district and easily know the changes that were mostly taking under which district

In [82]:

```
district = building_df['Supervisor District'].value_counts()
```

In [88]:

```
plt.title('number buildings under Supervisor District')
plt.bar(district.index, district);
```



Hence, it is clear that **3.0** district supervisor have the most buildings that were going through changes.

Inferences and Conclusion

In the dataset the excess amount of data is provided with alot of null values and wrong values and still get valuable information after data cleaning and sorting the data such as:

- Maximum permit type that is acquired by buildings.
- Correlation between different variables to know if any factor is dependable and can be perdictable.
- Best time to reach office and chances to get file issued.
- Most development that take place in particular area.
- Number that come under each Supervisor district.

References and Future Work

Check out the following resources to know more about the dataset and tools used in this notebook:

> - For more information about permit issuing: https://www.thespruce.com/what-is-a-building-permit-1398344 (https://www.thespruce.com/what-is-a-building-permit-1398344)
> - Pandas user guide: https://pandas.pydata.org/docs/user_guide/index.html (https://pandas.pydata.org/docs/user_guide/index.html) (https://pandas.pydata.org/docs/user_guide/index.html (https://pandas.pydata.org/docs/user_guide/index.html))
> - Matplotlib user guide: https://matplotlib.org/3.3.1/users/index.html (https://matplotlib.org/3.3.1/users/index.html) (https://matplotlib.org/3.3.1/users/index.html (https://matplotlib.org/3.3.1/users/index.html))
> - Seaborn user guide & tutorial: https://seaborn.pydata.org/tutorial.html (https://seaborn.pydata.org/tutorial.html) (https://seaborn.pydata.org/tutorial.html (https://seaborn.pydata.org/tutorial.html))
> - Opendatasets Python library: https://github.com/JovianML/opendatasets (https://github.com/JovianML/opendatasets) (https://github.com/JovianML/opendatasets (https://github.com/JovianML/opendatasets)
> - Project uploaded link: https://github.com/Jappreet-Singh/My-projects (https://github.com/Jappreet-Singh/My-projects)

In [ ]: