# Intro to Data Science_ Handling Tabular Data

March 25, 2020

[ ]:
```
# This Python 3 environment comes with many helpful analytics libraries␣
 ↪installed
# It is defined by the kaggle/python docker image: https://github.com/kaggle/
 ↪docker-python
# For example, here's several helpful packages to load in

import numpy as np # linear algebra
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)

# Input data files are available in the "../input/" directory.
# For example, running this (by clicking run or pressing Shift+Enter) will list␣
 ↪all files under the input directory

import os
for dirname, _, filenames in os.walk('/kaggle/input'):
    for filename in filenames:
        print(os.path.join(dirname, filename))

# Any results you write to the current directory are saved as output.
```

This is the post about **Introduction to Data Scinece**, I am going to write about* handling tabular/dataframe* data in python3.

### 0.0.1 Importing Data

[ ]:
```
import pandas as pd
df = pd.read_csv('/kaggle/input/california-housing-prices/housing.csv')
display(df.head())
```

### 0.0.2 Indexing and Selection

selecting a column

[ ]:
```
#select individual columns
pop = df['population']
display(pop)

#select multiple columns
```

```
cols_to_select = ['population','total_rooms', 'total_bedrooms']
cols = df[cols_to_select]
display(cols)
```

change columns name

[ ]:
```
#inplace doesn't copy a new dataframe
df.rename(columns = {'population':'pop'}, inplace=True)
display(df.head(2))
```

### 0.1 ## Filtering

Filtering is the process of extracting a subset of your data based on some condition or constraint. These conditions can be on the values that the data items take. We filter data when we wish to look at a smaller part of the whole data. For instance, we may want:

- the data in a particular period of the year
- the data of the highest selling items
- the data for a specific group of items
- to remove extra or useless data

we know that there is a columns **ocean proximity**, let's find how many **unique** value it has.

[ ]:
```
unique = df['ocean_proximity'].unique()
print(unique)
```

Now we want to find the data those those have **ISLAND** in them.

[ ]:
```
island_cond = df['ocean_proximity'] == 'ISLAND'
island_data = df[island_cond]

print('shape of original data :', df.shape)
print('shape of filtered data : ', island_data.shape)

#have only 5 columns those ocean_proximity is ISLAND
display(island_data.head(10))
```

Let's get more filtered data

- population less than 10000
- population greather than 15000 and households less than 5000
- house near ocean or also near bay

[ ]:
```
print('shape of original data :', df.shape)

#population less than 10k
pop_cond = df['pop'] < 10000
pop_data = df[pop_cond]
print(pop_data.shape)
```

```
#population greather than 15000 and households less than 5000
pop_ho = (df['pop'] > 15000) & (df['households'] < 5000)
pod_ho_data = df[pop_ho]
print(pod_ho_data.shape)

#house near ocean or also near bay
cond = ['NEAR OCEAN', 'NEAR BAY']
near_ocean_bay = df['ocean_proximity'].isin(cond)
nead_ocean_bay_df = df[near_ocean_bay]
print(nead_ocean_bay_df.shape)
```