Housing Market

Introduction:

This time we will create our own dataset with fictional numbers to describe a house market. As we are going to create random data don't try to reason of the numbers.

Step 1. Import the necessary libraries

In [1]:

```
import pandas as pd
import numpy as np
```

Step 2. Create 3 differents Series, each of length 100, as follows:

1. The first a random number from 1 to 4
2. The second a random number from 1 to 3
3. The third a random number from 10,000 to 30,000

In [2]:

```
s1 = pd.Series(np.random.randint(1, high=5, size=100, dtype='l'))
s2 = pd.Series(np.random.randint(1, high=4, size=100, dtype='l'))
s3 = pd.Series(np.random.randint(10000, high=30001, size=100, dtype='l'))

print(s1, s2, s3)
```

```
0     1
1     3
2     3
3     4
4     2
     ..
95    2
96    3
97    4
98    4
99    3
Length: 100, dtype: int32 0     1
1     3
2     1
3     2
4     3
     ..
95    1
96    2
97    1
98    3
99    3
Length: 100, dtype: int32 0     23690
1     28508
2     22220
3     29866
4     27563
      ...
95    25900
96    22739
97    12154
98    19921
99    10133
Length: 100, dtype: int32
```

Step 3. Let's create a DataFrame by joinning the Series by column

In [3]:

```
housemkt = pd.concat([s1, s2, s3], axis=1)
housemkt.head()
```

Out[3]:

	0	1	2
0	1	1	23690
1	3	3	28508
2	3	1	22220
3	4	2	29866
4	2	3	27563

Step 4. Change the name of the columns to bedrs, bathrs, price_sqr_meter

In [4]:

```
housemkt.rename(columns = {0: 'bedrs', 1: 'bathrs', 2: 'price_sqr_meter'}, inplace=True)
housemkt.head()
```

Out[4]:

	bedrs	bathrs	price_sqr_meter
0	1	1	23690
1	3	3	28508
2	3	1	22220
3	4	2	29866
4	2	3	27563

Step 5. Create a one column DataFrame with the values of the 3 Series and assign it to 'bigcolumn'

In [5]:

```
# join concat the values
bigcolumn = pd.concat([s1, s2, s3], axis=0)

# it is still a Series, so we need to transform it to a DataFrame
bigcolumn = bigcolumn.to_frame()
print(type(bigcolumn))

bigcolumn
```

```
<class 'pandas.core.frame.DataFrame'>
```

Out[5]:

	0
0	1
1	3
2	3
3	4
4	2
...	...
95	25900
96	22739
97	12154
98	19921
99	10133

300 rows × 1 columns

Step 6. Oops, it seems it is going only until index 99. Is it true?

In [6]:

```
bigcolumn.shape
```

Out[6]:

```
(300, 1)
```

Step 7. Reindex the DataFrame so it goes from 0 to 299

In [7]:

```
bigcolumn.reset_index(drop=True, inplace=True)
bigcolumn
```

Out[7]:

	0
0	1
1	3
2	3
3	4
4	2
...	...
295	25900
296	22739
297	12154
298	19921
299	10133

300 rows × 1 columns