Chapter 2 - Ex1: Housing prices

Cho dữ liệu housing-prices-dataset/train.csv

Yêu cầu: Thực hiện các công việc sau

1. Xác định các thuộc tính
2. Phân tích đơn biến
   2.1 Để dự đoán giá nhà, giả sử cần các thông tin sau: 'LotArea', 'YearBuilt', '1stFlrSF', '2ndFlrSF', 'FullBath', 'BedroomAbvGr', 'TotRmsAbvGrd' => phân tích các biến này
3. Phân tích hai biến
4. Xử lý dữ liệu thiếu
5. Phát hiện và xử lý ngoại lệ

- Note: Đây là một project với rất nhiều thuộc tính. Chúng ta chỉ thực hành trên một số thuộc tính để biết cách làm. Khi triển khai một project thực tế cần phải lựa chọn các thuộc tính phù hợp và thực hiện các công việc trên tất cả các thuộc tính được lựa chọn.

In [1]:
```
# link: https://www.kaggle.com/alphaepsilon/housing-prices-dataset
```

In [2]:
```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
import scipy
```

In [3]:
```
df = pd.read_csv("housing-prices-dataset/train.csv")
df.shape
```

Out[3]:
```
(1460, 81)
```

In [4]:
```
# df.info()
```

In [5]:
```
# df.isnull().sum() # dùng để đếm các giá trị null từ các cột
```

1. Xác định các thuộc tính

1. Input: <> SalePrice
2. Output: SalePrice
3. Type of variable:
   3.1 Predictor: khác SalePrice
   3.2 Target: SalePrice
4. Data Type:
   4.1 Charactor/String
   4.2 Numeric

5. Variable Category:

In [6]:
```
numbers = [f for f in df.columns if df.dtypes[f] != 'object'] # Quantitative
```

In [7]:
```
list_nums = ', '.join(numbers)
list_nums
```

Out[7]:
```
'Id, MSSubClass, LotFrontage, LotArea, OverallQual, OverallCond, YearBuilt, Yea
rRemodAdd, MasVnrArea, BsmtFinSF1, BsmtFinSF2, BsmtUnfSF, TotalBsmtSF, 1stFlrS
F, 2ndFlrSF, LowQualFinSF, GrLivArea, BsmtFullBath, BsmtHalfBath, FullBath, Hal
fBath, BedroomAbvGr, KitchenAbvGr, TotRmsAbvGrd, Fireplaces, GarageYrBlt, Garag
eCars, GarageArea, WoodDeckSF, OpenPorchSF, EnclosedPorch, 3SsnPorch, ScreenPor
ch, PoolArea, MiscVal, MoSold, YrSold, SalePrice'
```

In [8]:
```
objects = [f for f in df.columns if df.dtypes[f] == 'object'] # Qualitative
```

In [9]:
```
list_obj = ', '.join(objects)
list_obj
```

Out[9]:
```
'MSZoning, Street, Alley, LotShape, LandContour, Utilities, LotConfig, LandSlop
e, Neighborhood, Condition1, Condition2, BldgType, HouseStyle, RoofStyle, RoofM
atl, Exterior1st, Exterior2nd, MasVnrType, ExterQual, ExterCond, Foundation, Bs
mtQual, BsmtCond, BsmtExposure, BsmtFinType1, BsmtFinType2, Heating, HeatingQC,
CentralAir, Electrical, KitchenQual, Functional, FireplaceQu, GarageType, Garag
eFinish, GarageQual, GarageCond, PavedDrive, PoolQC, Fence, MiscFeature, SaleTy
pe, SaleCondition'
```

5. Variable
   5.1 Categorical:
   5.2 Continuous

In [10]:
```
# Categorical:
i = 1
for obj in objects:
    print(i, "/", obj, "\t",len(df[obj].unique()),":", df[obj].unique())
    i = i+1
```

```
1 / MSZoning     5 : ['RL' 'RM' 'C (all)' 'FV' 'RH']
2 / Street       2 : ['Pave' 'Grvl']
3 / Alley        3 : [nan 'Grvl' 'Pave']
4 / LotShape     4 : ['Reg' 'IR1' 'IR2' 'IR3']
5 / LandContour          4 : ['Lvl' 'Bnk' 'Low' 'HLS']
6 / Utilities    2 : ['AllPub' 'NoSeWa']
7 / LotConfig    5 : ['Inside' 'FR2' 'Corner' 'CulDSac' 'FR3']
8 / LandSlope    3 : ['Gtl' 'Mod' 'Sev']
9 / Neighborhood         25 : ['CollgCr' 'Veenker' 'Crawfor' 'NoRidge' 'Mitche
l' 'Somerst' 'NWAmes'
 'OldTown' 'BrkSide' 'Sawyer' 'NridgHt' 'NAmes' 'SawyerW' 'IDOTRR'
 'MeadowV' 'Edwards' 'Timber' 'Gilbert' 'StoneBr' 'ClearCr' 'NPkVill'
 'Blmngtn' 'BrDale' 'SWISU' 'Blueste']
10 / Condition1          9 : ['Norm' 'Feedr' 'PosN' 'Artery' 'RRAe' 'RRNn' 'RRA
n' 'PosA' 'RRNe']
11 / Condition2          8 : ['Norm' 'Artery' 'RRNn' 'Feedr' 'PosN' 'PosA' 'RRA
n' 'RRAe']
12 / BldgType    5 : ['1Fam' '2fmCon' 'Duplex' 'TwnhsE' 'Twnhs']
13 / HouseStyle          8 : ['2Story' '1Story' '1.5Fin' '1.5Unf' 'SFoyer' 'SLv
l' '2.5Unf' '2.5Fin']
14 / RoofStyle   6 : ['Gable' 'Hip' 'Gambrel' 'Mansard' 'Flat' 'Shed']
15 / RoofMatl    8 : ['CompShg' 'WdShngl' 'Metal' 'WdShake' 'Membran' 'Tar&Grv'
'Roll'
 'ClyTile']
16 / Exterior1st         15 : ['VinylSd' 'MetalSd' 'Wd Sdng' 'HdBoard' 'BrkFac
e' 'WdShing' 'CemntBd'
 'Plywood' 'AsbShng' 'Stucco' 'BrkComm' 'AsphShn' 'Stone' 'ImStucc'
 'CBlock']
17 / Exterior2nd         16 : ['VinylSd' 'MetalSd' 'Wd Shng' 'HdBoard' 'Plywoo
d' 'Wd Sdng' 'CmentBd'
 'BrkFace' 'Stucco' 'AsbShng' 'Brk Cmn' 'ImStucc' 'AsphShn' 'Stone'
 'Other' 'CBlock']
18 / MasVnrType          5 : ['BrkFace' 'None' 'Stone' 'BrkCmn' nan]
19 / ExterQual   4 : ['Gd' 'TA' 'Ex' 'Fa']
20 / ExterCond   5 : ['TA' 'Gd' 'Fa' 'Po' 'Ex']
21 / Foundation          6 : ['PConc' 'CBlock' 'BrkTil' 'Wood' 'Slab' 'Stone']
22 / BsmtQual    5 : ['Gd' 'TA' 'Ex' nan 'Fa']
23 / BsmtCond    5 : ['TA' 'Gd' nan 'Fa' 'Po']
24 / BsmtExposure        5 : ['No' 'Gd' 'Mn' 'Av' nan]
25 / BsmtFinType1        7 : ['GLQ' 'ALQ' 'Unf' 'Rec' 'BLQ' nan 'LwQ']
26 / BsmtFinType2        7 : ['Unf' 'BLQ' nan 'ALQ' 'Rec' 'LwQ' 'GLQ']
27 / Heating     6 : ['GasA' 'GasW' 'Grav' 'Wall' 'OthW' 'Floor']
28 / HeatingQC   5 : ['Ex' 'Gd' 'TA' 'Fa' 'Po']
29 / CentralAir          2 : ['Y' 'N']
30 / Electrical          6 : ['SBrkr' 'FuseF' 'FuseA' 'FuseP' 'Mix' nan]
31 / KitchenQual         4 : ['Gd' 'TA' 'Ex' 'Fa']
32 / Functional          7 : ['Typ' 'Min1' 'Maj1' 'Min2' 'Mod' 'Maj2' 'Sev']
33 / FireplaceQu         6 : [nan 'TA' 'Gd' 'Fa' 'Ex' 'Po']
34 / GarageType          7 : ['Attchd' 'Detchd' 'BuiltIn' 'CarPort' nan 'Basmen
t' '2Types']
35 / GarageFinish        4 : ['RFn' 'Unf' 'Fin' nan]
```

```
36 / GarageQual          6 : ['TA' 'Fa' 'Gd' nan 'Ex' 'Po']
37 / GarageCond          6 : ['TA' 'Fa' nan 'Gd' 'Po' 'Ex']
38 / PavedDrive          3 : ['Y' 'N' 'P']
39 / PoolQC      4 : [nan 'Ex' 'Fa' 'Gd']
40 / Fence       5 : [nan 'MnPrv' 'GdWo' 'GdPrv' 'MnWw']
41 / MiscFeature         5 : [nan 'Shed' 'Gar2' 'Othr' 'TenC']
42 / SaleType    9 : ['WD' 'New' 'COD' 'ConLD' 'ConLI' 'CWD' 'ConLw' 'Con' 'Oth']
43 / SaleCondition       6 : ['Normal' 'Abnorml' 'Partial' 'AdjLand' 'Alloca' 'Family']
```

In [11]:
```
# Categorical & Continueous
i = 1
for obj in numbers:
    print(i, "/", obj, "\t",len(df[obj].unique()),
          ":", df[obj].unique() if len(df[obj].unique())<150 else '')
    i = i+1
```

```
1 / Id   1460 :
2 / MSSubClass   15 : [ 60  20  70  50 190  45  90 120  30  85  80 160  75 180
40]
3 / LotFrontage          111 : [ 65.  80.  68.  60.  84.  85.  75.  nan  51.  5
0.  70.  91.  72.  66.
 101.  57.  44. 110.  98.  47. 108. 112.  74. 115.  61.  48.  33.  52.
 100.  24.  89.  63.  76.  81.  95.  69.  21.  32.  78. 121. 122.  40.
 105.  73.  77.  64.  94.  34.  90.  55.  88.  82.  71. 120. 107.  92.
 134.  62.  86. 141.  97.  54.  41.  79. 174.  99.  67.  83.  43. 103.
  93.  30. 129. 140.  35.  37. 118.  87. 116. 150. 111.  49.  96.  59.
  36.  56. 102.  58.  38. 109. 130.  53. 137.  45. 106. 104.  42.  39.
 144. 114. 128. 149. 313. 168. 182. 138. 160. 152. 124. 153.  46.]
4 / LotArea      1073 :
5 / OverallQual          10 : [ 7  6  8  5  9  4 10  3  1  2]
6 / OverallCond          9 : [5 8 6 7 4 2 3 9 1]
7 / YearBuilt    112 : [2003 1976 2001 1915 2000 1993 2004 1973 1931 1939 1965
2005 1962 2006
 1960 1929 1970 1967 1958 1930 2002 1968 2007 1951 1957 1927 1920 1966
 1959 1994 1954 1953 1955 1983 1975 1997 1934 1963 1981 1964 1999 1972
 1921 1945 1982 1998 1956 1948 1910 1995 1991 2009 1950 1961 1977 1985
 1979 1885 1919 1990 1969 1935 1988 1971 1952 1936 1923 1924 1984 1926
 1940 1941 1987 1986 2008 1908 1892 1916 1932 1918 1912 1947 1925 1900
 1980 1989 1992 1949 1880 1928 1978 1922 1996 2010 1946 1913 1937 1942
 1938 1974 1893 1914 1906 1890 1898 1904 1882 1875 1911 1917 1872 1905]
8 / YearRemodAdd         61 : [2003 1976 2002 1970 2000 1995 2005 1973 1950 196
5 2006 1962 2007 1960
 2001 1967 2004 2008 1997 1959 1990 1955 1983 1980 1966 1963 1987 1964
 1972 1996 1998 1989 1953 1956 1968 1981 1992 2009 1982 1961 1993 1999
 1985 1979 1977 1969 1958 1991 1971 1952 1975 2010 1984 1986 1994 1988
 1954 1957 1951 1978 1974]
9 / MasVnrArea   328 :
10 / BsmtFinSF1          637 :
11 / BsmtFinSF2          144 : [   0   32  668  486   93  491  506  712  362
41  169  869  150  670
   28 1080  181  768  215  374  208  441  184  279  306  180  580  690
  692  228  125 1063  620  175  820 1474  264  479  147  232  380  544
  294  258  121  391  531  344  539  713  210  311 1120  165  532   96
  495  174 1127  139  202  645  123  551  219  606  612  480  182  132
  336  468  287   35  499  723  119   40  117  239   80  472   64 1057
  127  630  128  377  764  345 1085  435  823  500  290  324  634  411
  841 1061  466  396  354  149  193  273  465  400  682  557  230  106
  791  240  547  469  177  108  600  492  211  168 1031  438  375  144
   81  906  608  276  661   68  173  972  105  420  546  334  352  872
  110  627  163 1029]
12 / BsmtUnfSF   780 :
13 / TotalBsmtSF         721 :
14 / 1stFlrSF    753 :
15 / 2ndFlrSF    417 :
16 / LowQualFinSF        24 : [  0 360 513 234 528 572 144 392 371 390 420 473
156 515  80  53 232 481
```

```
 120 514 397 479 205 384]
17 / GrLivArea   861 :
18 / BsmtFullBath        4 : [1 0 2 3]
19 / BsmtHalfBath        3 : [0 1 2]
20 / FullBath    4 : [2 1 3 0]
21 / HalfBath    3 : [1 0 2]
22 / BedroomAbvGr        8 : [3 4 1 2 0 5 6 8]
23 / KitchenAbvGr        4 : [1 2 3 0]
24 / TotRmsAbvGrd        12 : [ 8  6  7  9  5 11  4 10 12  3  2 14]
25 / Fireplaces          4 : [0 1 2 3]
26 / GarageYrBlt         98 : [2003. 1976. 2001. 1998. 2000. 1993. 2004. 1973.
1931. 1939. 1965. 2005.
 1962. 2006. 1960. 1991. 1970. 1967. 1958. 1930. 2002. 1968. 2007. 2008.
 1957. 1920. 1966. 1959. 1995. 1954. 1953.   nan 1983. 1977. 1997. 1985.
 1963. 1981. 1964. 1999. 1935. 1990. 1945. 1987. 1989. 1915. 1956. 1948.
 1974. 2009. 1950. 1961. 1921. 1900. 1979. 1951. 1969. 1936. 1975. 1971.
 1923. 1984. 1926. 1955. 1986. 1988. 1916. 1932. 1972. 1918. 1980. 1924.
 1996. 1940. 1949. 1994. 1910. 1978. 1982. 1992. 1925. 1941. 2010. 1927.
 1947. 1937. 1942. 1938. 1952. 1928. 1922. 1934. 1906. 1914. 1946. 1908.
 1929. 1933.]
27 / GarageCars          5 : [2 3 1 0 4]
28 / GarageArea          441 :
29 / WoodDeckSF          274 :
30 / OpenPorchSF         202 :
31 / EnclosedPorch       120 : [  0 272 228 205 176  87 172 102  37 144  64 114
202 128 156  44  77 192
 140 180 183  39 184  40 552  30 126  96  60 150 120 112 252  52 224 234
 244 268 137  24 108 294 177 218 242  91 160 130 169 105  34 248 236  32
  80 115 291 116 158 210  36 200  84 148 136 240  54 100 189 293 164 216
 239  67  90  56 129  98 143  70 386 154 185 134 196 264 275 230 254  68
 194 318  48  94 138 226 174  19 170 220 214 280 190 330 208 145 259  81
  42 123 162 286 168  20 301 198 221 212  50  99]
32 / 3SsnPorch   20 : [  0 320 407 130 180 168 140 508 238 245 196 144 182 162
23 216  96 153
 290 304]
33 / ScreenPorch         76 : [  0 176 198 291 252  99 184 168 130 142 192 410
224 266 170 154 153 144
 128 259 160 271 234 374 185 182  90 396 140 276 180 161 145 200 122  95
 120  60 126 189 260 147 385 287 156 100 216 210 197 204 225 152 175 312
 222 265 322 190 233  63  53 143 273 288 263  80 163 116 480 178 440 155
 220 119 165  40]
34 / PoolArea    8 : [  0 512 648 576 555 480 519 738]
35 / MiscVal     21 : [    0   700   350   500   400   480   450 15500  1200
800  2000   600
  3500  1300    54   620   560  1400  8300  1150  2500]
36 / MoSold      12 : [ 2  5  9 12 10  8 11  4  1  7  3  6]
37 / YrSold      5 : [2008 2007 2006 2009 2010]
38 / SalePrice   663 :
```

In [12]:
```
# Quan sát 2 kết quả trên để kết luận
```

2. Phân tích đơn biến

In [13]:
```
features = ['LotArea', 'YearBuilt', '1stFlrSF', '2ndFlrSF',
            'FullBath', 'BedroomAbvGr', 'TotRmsAbvGrd']
```

In [14]:
```
i = 1
for obj in features:
    print(i, "/", obj, "\t",len(df[obj].unique()),
          ":", df[obj].unique() if len(df[obj].unique())<150 else '')
    i = i+1
```

```
1 / LotArea      1073 :
2 / YearBuilt    112 : [2003 1976 2001 1915 2000 1993 2004 1973 1931 1939 1965
2005 1962 2006
 1960 1929 1970 1967 1958 1930 2002 1968 2007 1951 1957 1927 1920 1966
 1959 1994 1954 1953 1955 1983 1975 1997 1934 1963 1981 1964 1999 1972
 1921 1945 1982 1998 1956 1948 1910 1995 1991 2009 1950 1961 1977 1985
 1979 1885 1919 1990 1969 1935 1988 1971 1952 1936 1923 1924 1984 1926
 1940 1941 1987 1986 2008 1908 1892 1916 1932 1918 1912 1947 1925 1900
 1980 1989 1992 1949 1880 1928 1978 1922 1996 2010 1946 1913 1937 1942
 1938 1974 1893 1914 1906 1890 1898 1904 1882 1875 1911 1917 1872 1905]
3 / 1stFlrSF     753 :
4 / 2ndFlrSF     417 :
5 / FullBath     4 : [2 1 3 0]
6 / BedroomAbvGr         8 : [3 4 1 2 0 5 6 8]
7 / TotRmsAbvGrd         12 : [ 8  6  7  9  5 11  4 10 12  3  2 14]
```

Continuous variable

- LotArea
- 1stFlrSF
- 2ndFlrSF

In [15]:
```
# LotArea
# Central Tendency
df['LotArea'].describe(include='all')
```

Out[15]:
```
count      1460.000000
mean      10516.828082
std        9981.264932
min        1300.000000
25%        7553.500000
50%        9478.500000
75%       11601.500000
max      215245.000000
Name: LotArea, dtype: float64
```

In [16]:
```
LotArea_median = df.LotArea.median()
LotArea_median
```

Out[16]:
```
9478.5
```

In [17]:
```
LotArea_mode = df.LotArea.mode()
LotArea_mode
```

Out[17]:
```
0    7200
dtype: int64
```

In [18]:
```
# Measure of Dispersion
LotArea_range = df.LotArea.ptp()
LotArea_range
```

```
c:\program files\python36\lib\site-packages\ipykernel_launcher.py:2: FutureWarn
ing: Method .ptp is deprecated and will be removed in a future version. Use num
py.ptp instead.
```

Out[18]:
```
213945
```

In [19]:
```
Q1 = np.percentile(df.LotArea, 25)
Q1
```

Out[19]:
```
7553.5
```

In [20]:
```
Q3 = np.percentile(df.LotArea, 75)
Q3
```

Out[20]:
```
11601.5
```

In [21]:
```
LotArea_iqr = scipy.stats.iqr(df.LotArea)
LotArea_iqr
```

Out[21]:
```
4048.0
```

In [22]:
```
LotArea_var = df.LotArea.var()
LotArea_var
```

Out[22]:
```
99625649.65034176
```

In [23]:
```
LotArea_std = df.LotArea.std()
LotArea_std
```

Out[23]:
```
9981.26493237915
```

In [24]:
```
LotArea_skew = df.LotArea.skew()
LotArea_skew
```

Out[24]:
```
12.207687851233496
```

In [25]:
```
# LotArea_skew > 0 => phân phối lệch phải
scipy.stats.skew(df.LotArea)
```

Out[25]:
```
12.195142125084478
```

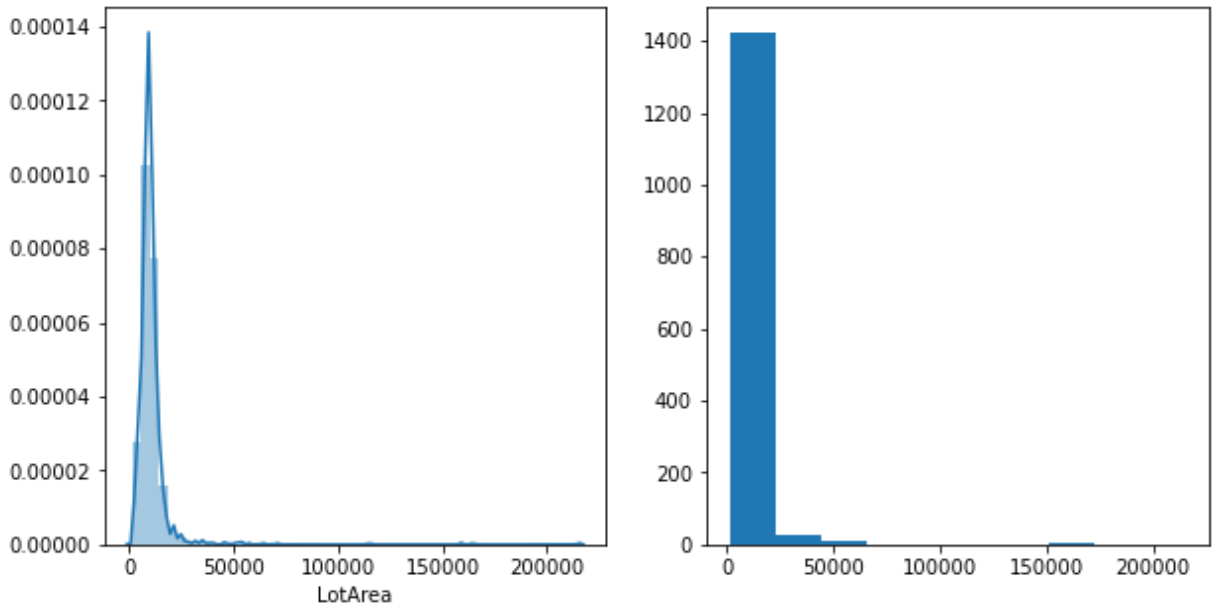
In [26]:
```
LotArea_kur = df.LotArea.kurtosis()
LotArea_kur
```

Out[26]:
```
203.24327101886033
```

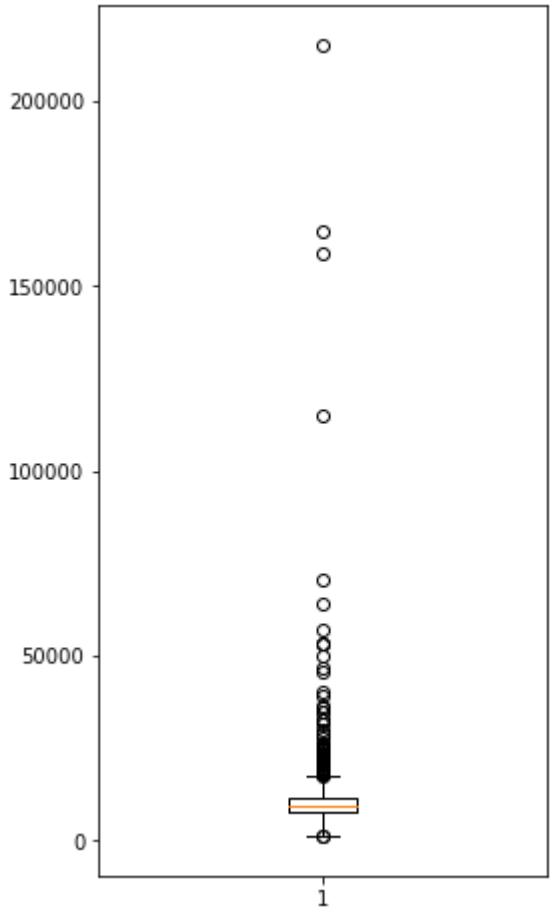
In [27]:
```
# LotArea_kur > 0: phân phối nhọn hơn phân phối chuẩn
scipy.stats.kurtosis(df.LotArea)
```

Out[27]:
```
202.5437927513529
```

In [28]:
```
# Visualization
# Histogram
plt.figure(figsize=(10, 5))
plt.subplot(1, 2, 1)
sns.distplot(df.LotArea)
plt.subplot(1, 2, 2)
plt.hist(df.LotArea)
plt.show()
```



In [29]:
```
# Boxplot
plt.figure(figsize=(4,8))
plt.boxplot(df.LotArea)
plt.show()
```



In [30]:
```
# Number of upper outliers
n_O_upper = df[df.LotArea > (Q3 + 1.5*LotArea_iqr)].shape[0]
n_O_upper
```

Out[30]: 67

In [31]:
```
# Number of lower outliers
n_O_lower = df[df.LotArea < (Q1 - 1.5*LotArea_iqr)].shape[0]
n_O_lower
```

Out[31]: 2

In [32]:
```
# Percentage of ouliers
outliers_per = (n_O_lower + n_O_upper)/df.shape[0]
outliers_per
```

Out[32]: 0.04726027397260274

In [33]:
```
# xem xét loại bỏ outliers ???
```

Note: Hai thuộc tính còn lại ((1stFlrSF, 2ndFlrSF)) làm tương tự như trên

Categorical Variables

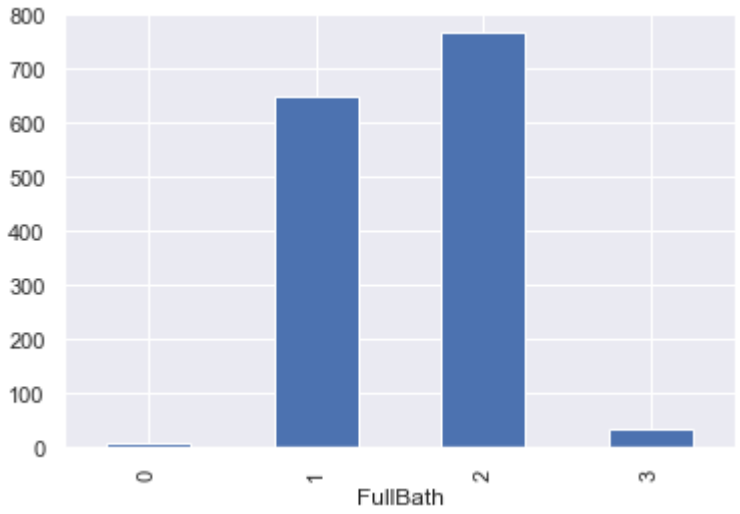
- FullBath
- BedroomAbvGr
- TotRmsAbvGrd

In [34]:
```
# FullBath
FullBath_count = (df.groupby("FullBath").count())["Id"]
FullBath_count
```

Out[34]:
```
FullBath
0      9
1    650
2    768
3     33
Name: Id, dtype: int64
```

In [35]:
```
sns.set()
FullBath_count.plot.bar()
```

Out[35]:
```
<AxesSubplot:xlabel='FullBath'>
```



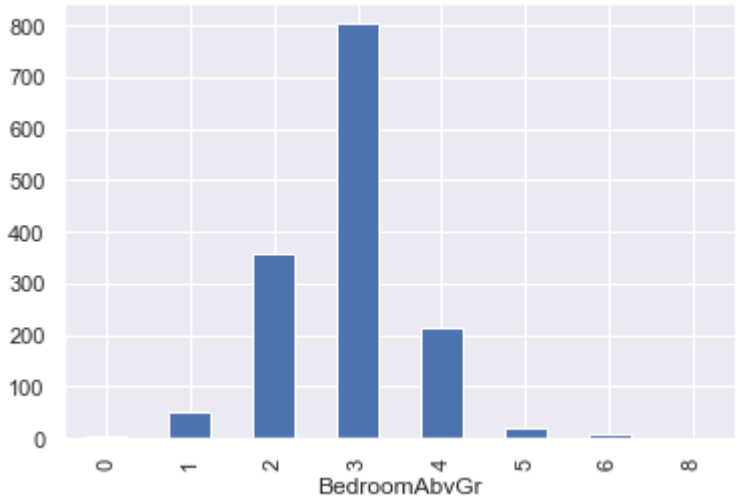
- Note: Hai thuộc tính còn lại ((BedroomAbvGr, TotRmsAbvGrd)) làm tương tự như trên

In [36]:
```
# BedroomAbvGr
BedroomAbvGr_count = (df.groupby("BedroomAbvGr").count())["Id"]
BedroomAbvGr_count
```

Out[36]:
```
BedroomAbvGr
0      6
1     50
2    358
3    804
4    213
5     21
6      7
8      1
Name: Id, dtype: int64
```

In [37]:
```
sns.set()
BedroomAbvGr_count.plot.bar()
```

Out[37]:
```
<AxesSubplot:xlabel='BedroomAbvGr'>
```

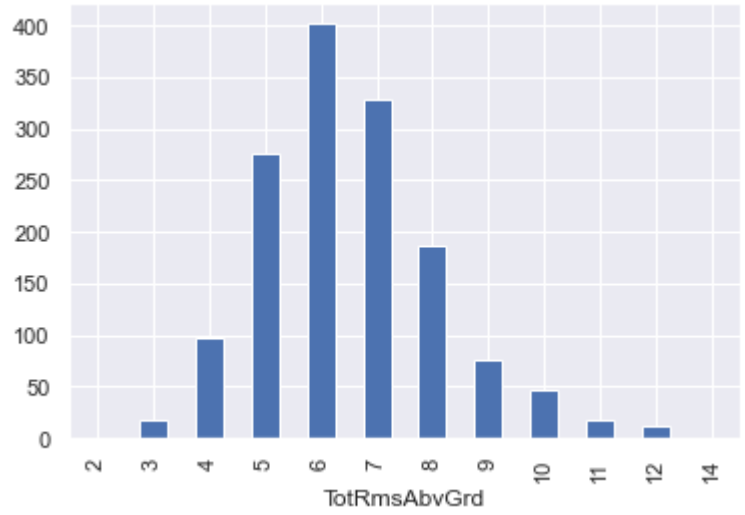


In [38]:
```
# TotRmsAbvGrd
TotRmsAbvGrd_count = (df.groupby("TotRmsAbvGrd").count())["Id"]
TotRmsAbvGrd_count
```

Out[38]:
```
TotRmsAbvGrd
2       1
3      17
4      97
5     275
6     402
7     329
8     187
9      75
10     47
11     18
12     11
14      1
Name: Id, dtype: int64
```

In [39]:
```
sns.set()
TotRmsAbvGrd_count.plot.bar()
```

Out[39]:
```
<AxesSubplot:xlabel='TotRmsAbvGrd'>
```

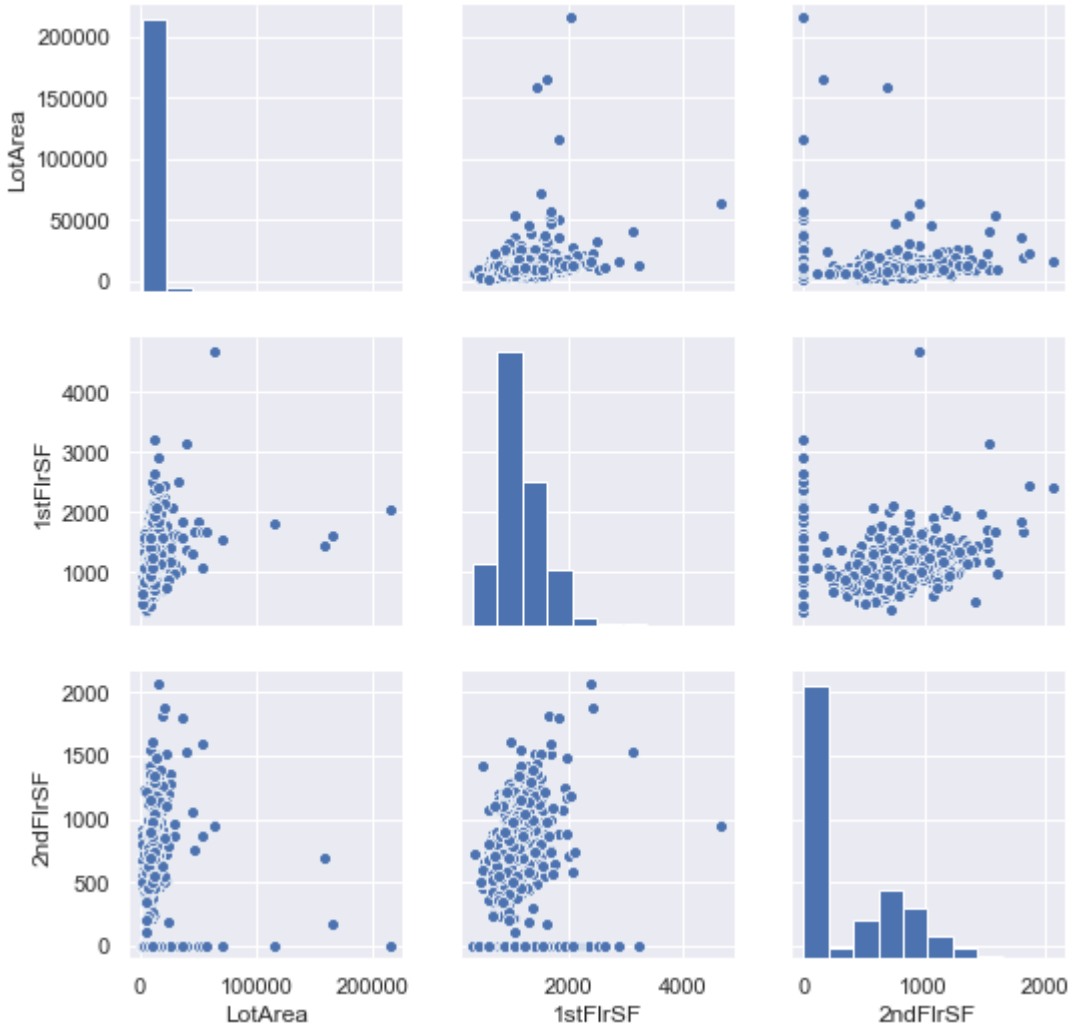


3. Phân tích hai biến

- Continuous & Continous
- Categorical & Categorical
- Categorical & Continuous

In [40]:
```
# Continuous & Continous
sns.pairplot(df[['LotArea', '1stFlrSF', '2ndFlrSF']])
```

Out[40]:
```
<seaborn.axisgrid.PairGrid at 0x29697b18f28>
```



In [41]:
```
# Không quan hệ tuyến tính
df[['LotArea', '1stFlrSF', '2ndFlrSF']].corr()
```

Out[41]:

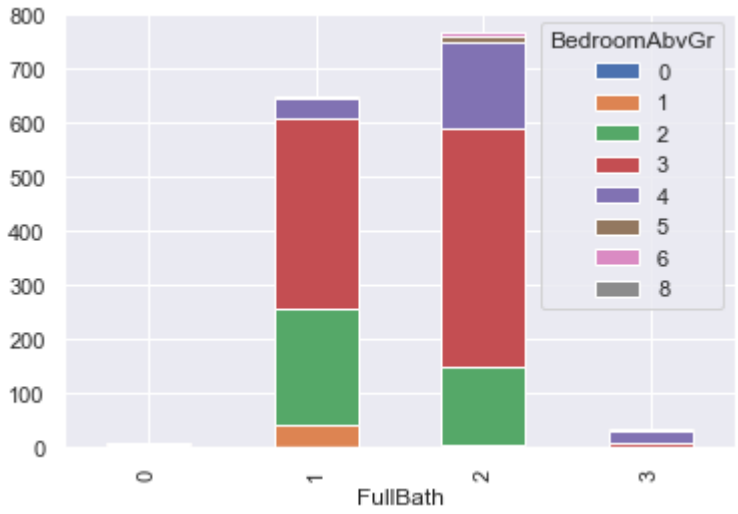
	LotArea	1stFlrSF	2ndFlrSF
LotArea	1.000000	0.299475	0.050986
1stFlrSF	0.299475	1.000000	-0.202646
2ndFlrSF	0.050986	-0.202646	1.000000

In [42]:
```
# Categorical & Categorical
# 'FullBath', 'BedroomAbvGr', 'TotRmsAbvGrd'
# Contingency table:  Ho: 'FullBath' and 'BedroomAbvGr' independent
table_FB = pd.crosstab(df['FullBath'], df['BedroomAbvGr'])
table_FB
```

Out[42]:

BedroomAbvGr	0	1	2	3	4	5	6	8
FullBath								
0	5	2	2	0	0	0	0	0
1	1	42	213	352	37	5	0	0
2	0	6	143	443	156	12	7	1
3	0	0	0	9	20	4	0	0

In [43]:
```
table_FB.plot(kind='bar', stacked=True)
plt.show()
```



In [44]:
```
# chi-squared test with similar proportions
from scipy.stats import chi2_contingency
from scipy.stats import chi2
```

In [45]:
```
# Chi-Square Test
stat, p, dof, expected = chi2_contingency(table_FB)
print('dof=%d' % dof)
print('p=', p)
```

```
dof=21
p= 9.022959522651409e-177
```

In [46]:
```
# interpret test-statistic
prob = 0.95
critical = chi2.ppf(prob, dof)
print('probability=%.3f, critical=%.3f, stat=%.3f' % (prob, critical, stat))
```

```
probability=0.950, critical=32.671, stat=898.930
```

In [47]:
```
# interpret p-value
alpha = 1.0 - prob
print('significance=%.3f, p=%.3f' % (alpha, p))
if p <= alpha:
    print('Dependent (reject H0)')
else:
    print('Independent (fail to reject H0)')
```

```
significance=0.050, p=0.000
Dependent (reject H0)
```

Note: Các cặp còn lại làm tương tự như trên

In [48]:
```
# Categorical & Continuous
# 'FullBath' có bị ảnh hưởng bởi 'LotArea'? => ANOVA ONEWAY
table_FB = pd.crosstab(df['FullBath'], df['BedroomAbvGr'])
table_FB
```

Out[48]:

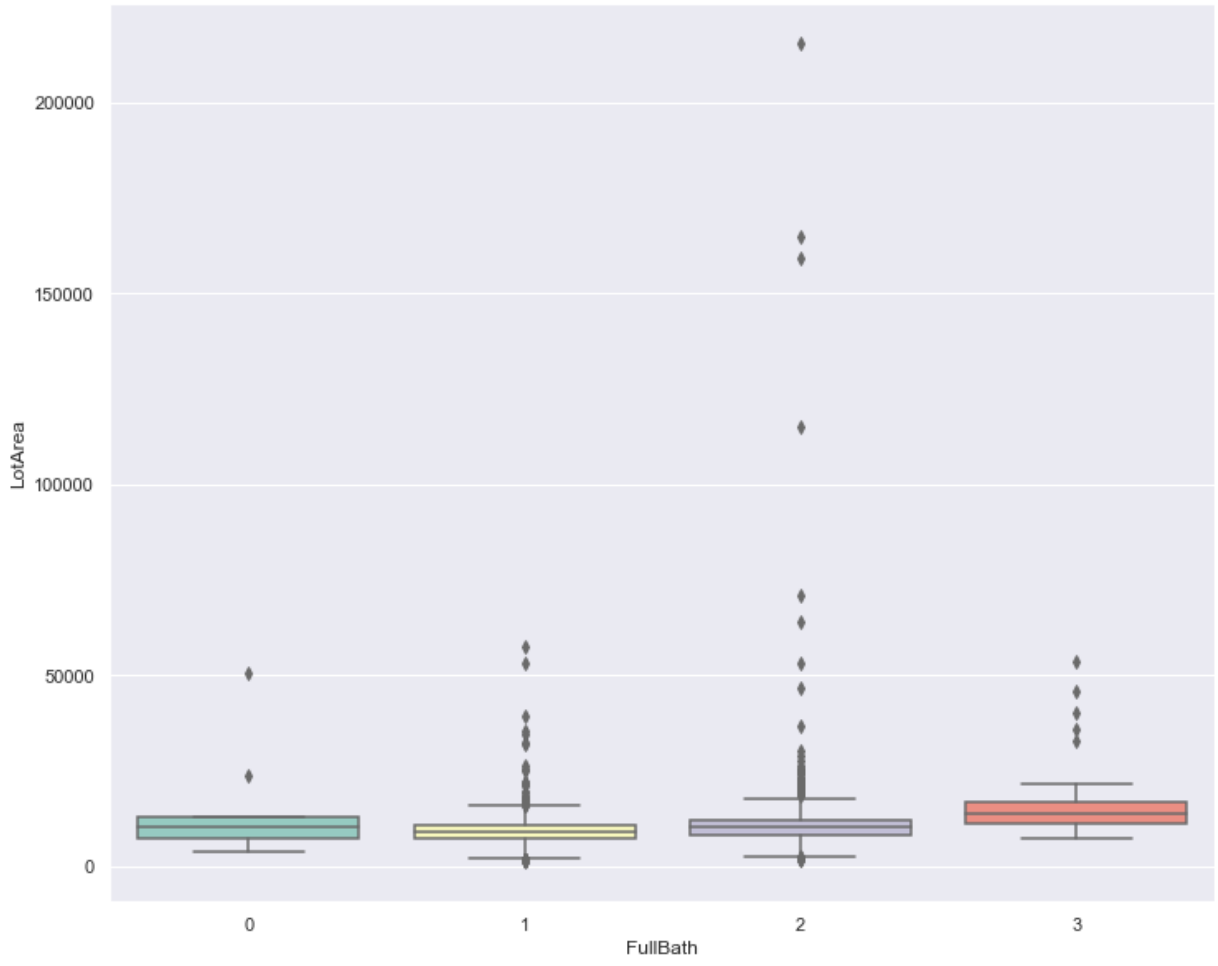
BedroomAbvGr	0	1	2	3	4	5	6	8
FullBath								
0	5	2	2	0	0	0	0	0
1	1	42	213	352	37	5	0	0
2	0	6	143	443	156	12	7	1
3	0	0	0	9	20	4	0	0

In [49]:
```
df_sub = df[['FullBath','LotArea']]
df_sub.head()
```

Out[49]:

	FullBath	LotArea
0	2	8450
1	2	9600
2	2	11250
3	1	9550
4	2	14260

In [50]:
```
import matplotlib.pyplot as plt
plt.figure(figsize=(12,10))
sns.boxplot(x="FullBath", y="LotArea", data=df_sub, palette="Set3")
plt.show()
```



In [51]:
```
import statsmodels.api as sm
from statsmodels.formula.api import ols
```

In [52]:
```
model = ols('LotArea ~ C(FullBath)', data=df_sub).fit()
anova_table = sm.stats.anova_lm(model, typ=2)
anova_table
```

Out[52]:

	sum_sq	df	F	PR(>F)
C(FullBath)	3.332090e+09	3.0	11.386809	2.207543e-07
Residual	1.420217e+11	1456.0	NaN	NaN

- Giải thích: P-value thu được từ phân tích ANOVA cho LotArea và FullBath phối hợp có ý nghĩa thống kê (P <0.05).
- Kết luận: LotArea ảnh hưởng đáng kể đến FullBath

In [53]:
```
from statsmodels.stats.multicomp import pairwise_tukeyhsd
# perform multiple pairwise comparison (Tukey HSD)
m_comp = pairwise_tukeyhsd(endog=df_sub['LotArea'],
                           groups=df_sub['FullBath'],
                           alpha=0.05)
print(m_comp)
```

```
     Multiple Comparison of Means - Tukey HSD, FWER=0.05
=============================================================
group1 group2  meandiff  p-adj     lower      upper    reject
-------------------------------------------------------------
     0      1 -5695.6583 0.3146 -14221.4451  2830.1286  False
     0      2 -3587.0109 0.6772  -12103.848  4929.8263  False
     0      3  2477.0404    0.9  -7075.4392   12029.52  False
     1      2  2108.6474  0.001    754.7975  3462.4974   True
     1      3  8172.6987  0.001   3639.9003 12705.4971   True
     2      3  6064.0513 0.0032   1548.1091 10579.9935   True
-------------------------------------------------------------
```

- Các kết quả trên từ Tukey HSD cho thấy 0-1, 0-2, 0-3: chấp nhận Ho, các so sánh cặp khác về số phòng bác bỏ Ho và chỉ ra sự khác biệt đáng kể về mặt thống kê.

4. Xử lý dữ liệu thiếu

In [54]:
```
# Kiểm tra dữ liệu thiếu
df_now = df[['LotArea', 'YearBuilt', '1stFlrSF', '2ndFlrSF',
             'FullBath', 'BedroomAbvGr', 'TotRmsAbvGrd']]
df_now.isnull().sum()
```

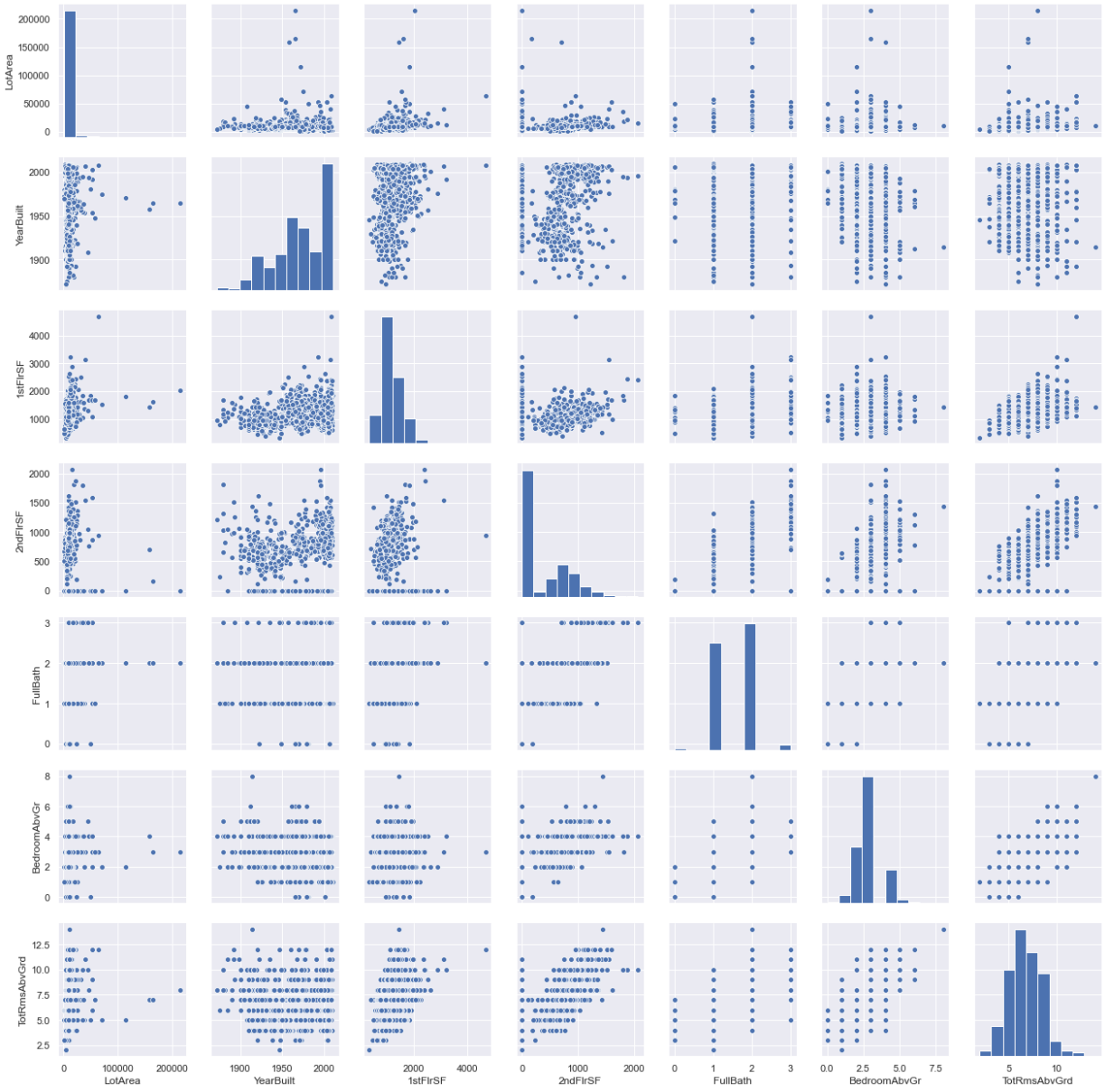
Out[54]:
```
LotArea         0
YearBuilt       0
1stFlrSF        0
2ndFlrSF        0
FullBath        0
BedroomAbvGr    0
TotRmsAbvGrd    0
dtype: int64
```

In [55]:
```
# Không có dữ liệu thiếu
```

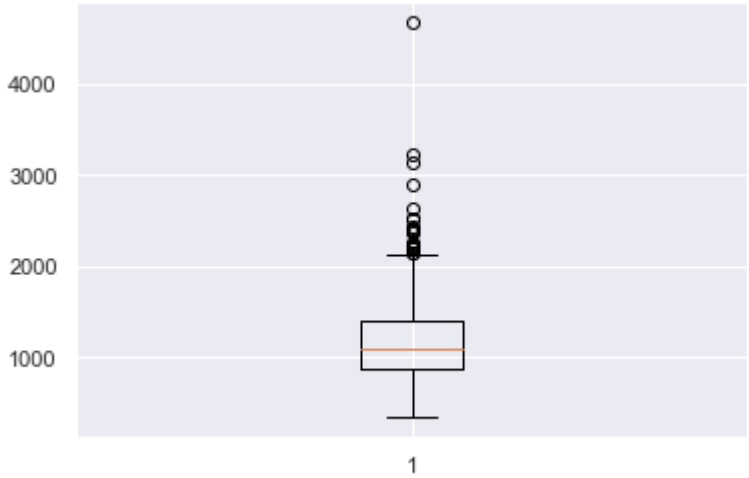
5. Phát hiện và xử lý ngoại lệ

In [56]:
```
sns.pairplot(df[features])
```

Out[56]: <seaborn.axisgrid.PairGrid at 0x2969803ba20>



In [57]:
```
plt.boxplot(df['1stFlrSF'])
plt.show()
```



In [58]:
```
Q11 = np.percentile(df['1stFlrSF'], 25)
Q11
```

Out[58]: 882.0

In [59]:
```
Q31 = np.percentile(df['1stFlrSF'], 75)
Q31
```

Out[59]: 1391.25

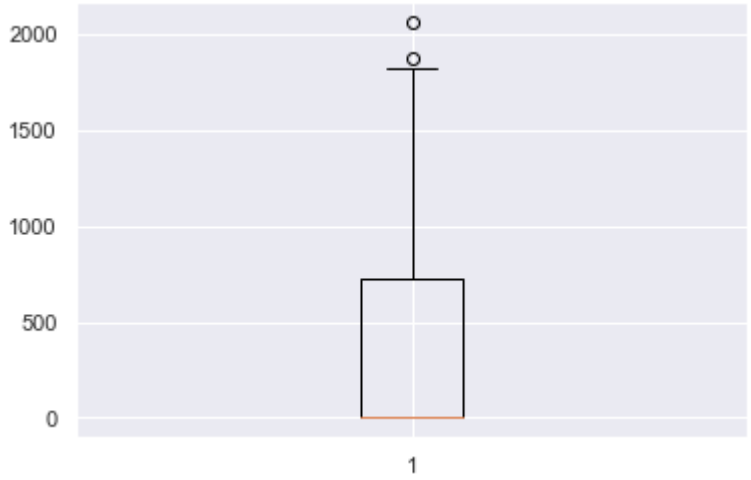
In [60]:
```
IQR1 = scipy.stats.iqr(df['1stFlrSF'])
IQR1
```

Out[60]: 509.25

In [61]:
```
# Number of upper outliers
n_O_upper1 = df[df['1stFlrSF'] > (Q31 + 1.5*IQR1)].shape[0]
n_O_upper1
```

Out[61]: 20

In [62]:
```
plt.boxplot(df['2ndFlrSF'])
plt.show()
```



In [63]:
```
Q12 = np.percentile(df['2ndFlrSF'], 25)
Q12
```

Out[63]: 0.0

In [64]:
```
Q32 = np.percentile(df['2ndFlrSF'], 75)
Q32
```

Out[64]: 728.0

In [65]:
```
IQR2 = scipy.stats.iqr(df['2ndFlrSF'])
IQR2
```

Out[65]: 728.0

In [66]:
```
# Number of upper outliers
n_O_upper2 = df[df['2ndFlrSF'] > (Q32 + 1.5*IQR2)].shape[0]
n_O_upper2
```

Out[66]: 2

In [67]:
```
# Percentage of ouliers
outliers_per = (n_O_lower + n_O_upper + n_O_upper1 + n_O_upper2 )/df.shape[0]
outliers_per
```

Out[67]: 0.06232876712328767

- Có thể drop doutliers của '2ndFlrSF', '1stFlrSF', 'LotArea': vì tổng số outliers là 6% dữ liệu
- Cũng có thể không cần drop thay vào đó khi áp dụng Machine Learning thì dùng thuật toán Decision Tree/ Random Forest (LDS6)

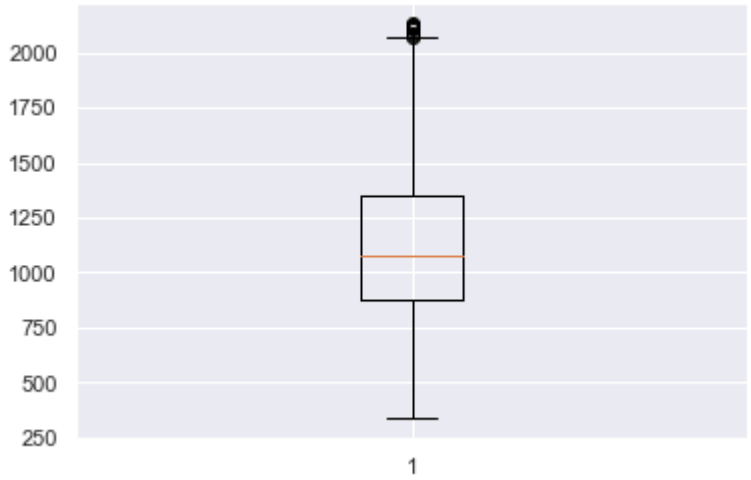
In [68]:
```
df_now = df_now[(df_now['2ndFlrSF'] <= (Q32 + 1.5*IQR2))]
```

In [69]:
```
df_now = df_now[(df_now['1stFlrSF'] <= (Q31 + 1.5*IQR1))]
```

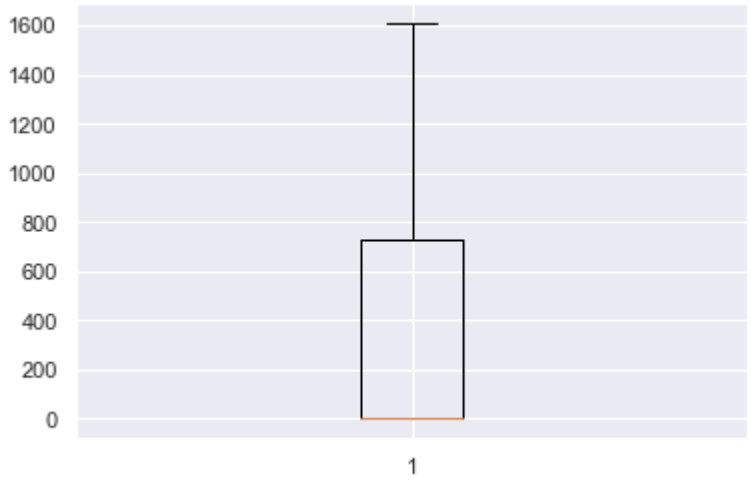
In [70]:
```
df_now = df_now[(df_now.LotArea >= (Q1 - 1.5*LotArea_iqr)) & (df_now.LotArea
<=(Q3 + 1.5*LotArea_iqr))]
```

In [71]:
```
plt.boxplot(df_now['LotArea'])
plt.show()
```

In [72]:
```
plt.boxplot(df_now['1stFlrSF'])
plt.show()
```



In [73]:
```
plt.boxplot(df_now['2ndFlrSF'])
plt.show()
```



In [74]:
```
# Xem xét thêm biên ngoài: major
# https://m.wikihow.com/Calculate-Outliers
# Công thức: O_u = Q3 + 3*IQR, O_l = Q1 - 3*IQR
# hoặc tính trung bình trước và sau khi loại bỏ outlier
```

In [75]:
```
df_now.LotArea.mean()
```

Out[75]: 9228.756168359942

In [76]:
```
df.LotArea.mean()
```

Out[76]: 10516.828082191782

In [77]:
```
# Nên loại bỏ ngoại lệ vì chênh lệch lớn
```

In [78]:
```
df_now['1stFlrSF'].mean()
```

Out[78]: 1132.8526850507983

In [79]:
```
df['1stFlrSF'].mean()
```

Out[79]:
```
1162.626712328767
```

In [80]:
```
# Không nhất thiết phải loại bỏ ngoại lệ vì chênh lệch nhỏ
```

In [81]:
```
df_now['2ndFlrSF'].mean()
```

Out[81]:
```
340.67851959361394
```

In [82]:
```
df['2ndFlrSF'].mean()
```

Out[82]:
```
346.99246575342465
```

In [83]:
```
# Không nhất thiết phải loại bỏ ngoại lệ vì chênh lệch nhỏ
```

In [84]:
```
# Vì 3 phân phối này đều không là phân phối chuẩn => không drop theo z-score
```

In [85]:
```
# Hoặc có thể xem xét chỉnh dữ liệu log(cột)
```