Feature Engineering

```
# import libraries
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from scipy.stats import skew
from scipy.stats import chi2_contingency
from random import sample
import warnings
warnings.filterwarnings('ignore')

# Load dataset
fe_df = pd.read_csv('PEP1.csv', index_col=0)
fe_df.head()
```

```
    MSSubClass MSZoning  LotFrontage  LotArea Street Alley LotShape  \
Id
1           60       RL         65.0     8450   Pave   NaN      Reg
2           20       RL         80.0     9600   Pave   NaN      Reg
3           60       RL         68.0    11250   Pave   NaN      IR1
4           70       RL         60.0     9550   Pave   NaN      IR1
5           60       RL         84.0    14260   Pave   NaN      IR1

   LandContour Utilities LotConfig  ... PoolArea PoolQC Fence
MiscFeature  \
Id                                  ...

1          Lvl    AllPub    Inside  ...        0    NaN   NaN
NaN
2          Lvl    AllPub       FR2  ...        0    NaN   NaN
NaN
3          Lvl    AllPub    Inside  ...        0    NaN   NaN
NaN
4          Lvl    AllPub    Corner  ...        0    NaN   NaN
NaN
5          Lvl    AllPub       FR2  ...        0    NaN   NaN
NaN

   MiscVal MoSold  YrSold  SaleType  SaleCondition  SalePrice
Id
1        0      2    2008        WD         Normal     208500
2        0      5    2007        WD         Normal     181500
3        0      9    2008        WD         Normal     223500
4        0      2    2006        WD        Abnorml     140000
5        0     12    2008        WD         Normal     250000

[5 rows x 80 columns]
```

```
fe_df.columns
```

```
Index(['MSSubClass', 'MSZoning', 'LotFrontage', 'LotArea', 'Street',
'Alley',
       'LotShape', 'LandContour', 'Utilities', 'LotConfig',
'LandSlope',
       'Neighborhood', 'Condition1', 'Condition2', 'BldgType',
'HouseStyle',
       'OverallQual', 'OverallCond', 'YearBuilt', 'YearRemodAdd',
'RoofStyle',
       'RoofMatl', 'Exterior1st', 'Exterior2nd', 'MasVnrType',
'MasVnrArea',
       'ExterQual', 'ExterCond', 'Foundation', 'BsmtQual', 'BsmtCond',
       'BsmtExposure', 'BsmtFinType1', 'BsmtFinSF1', 'BsmtFinType2',
       'BsmtFinSF2', 'BsmtUnfSF', 'TotalBsmtSF', 'Heating',
'HeatingQC',
       'CentralAir', 'Electrical', '1stFlrSF', '2ndFlrSF',
'LowQualFinSF',
       'GrLivArea', 'BsmtFullBath', 'BsmtHalfBath', 'FullBath',
'HalfBath',
       'BedroomAbvGr', 'KitchebvGr', 'KitchenQual', 'TotRmsAbvGrd',
'Functiol',
       'Fireplaces', 'FireplaceQu', 'GarageType', 'GarageYrBlt',
       'GarageFinish', 'GarageCars', 'GarageArea', 'GarageQual',
'GarageCond',
       'PavedDrive', 'WoodDeckSF', 'OpenPorchSF', 'EnclosedPorch',
'3SsnPorch',
       'ScreenPorch', 'PoolArea', 'PoolQC', 'Fence', 'MiscFeature',
'MiscVal',
       'MoSold', 'YrSold', 'SaleType', 'SaleCondition', 'SalePrice'],
      dtype='object')
```

```
fe_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 1460 entries, 1 to 1460
Data columns (total 80 columns):
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   MSSubClass     1460 non-null   int64
 1   MSZoning       1460 non-null   object
 2   LotFrontage    1201 non-null   float64
 3   LotArea        1460 non-null   int64
 4   Street         1460 non-null   object
 5   Alley          91 non-null     object
 6   LotShape       1460 non-null   object
 7   LandContour    1460 non-null   object
 8   Utilities      1460 non-null   object
 9   LotConfig      1460 non-null   object
 10  LandSlope      1460 non-null   object
 11  Neighborhood   1460 non-null   object
```

```
12  Condition1     1460 non-null   object
13  Condition2     1460 non-null   object
14  BldgType       1460 non-null   object
15  HouseStyle     1460 non-null   object
16  OverallQual    1460 non-null   int64
17  OverallCond    1460 non-null   int64
18  YearBuilt      1460 non-null   int64
19  YearRemodAdd   1460 non-null   int64
20  RoofStyle      1460 non-null   object
21  RoofMatl       1460 non-null   object
22  Exterior1st    1460 non-null   object
23  Exterior2nd    1460 non-null   object
24  MasVnrType     1452 non-null   object
25  MasVnrArea     1452 non-null   float64
26  ExterQual      1460 non-null   object
27  ExterCond      1460 non-null   object
28  Foundation     1460 non-null   object
29  BsmtQual       1423 non-null   object
30  BsmtCond       1423 non-null   object
31  BsmtExposure   1422 non-null   object
32  BsmtFinType1   1423 non-null   object
33  BsmtFinSF1     1460 non-null   int64
34  BsmtFinType2   1422 non-null   object
35  BsmtFinSF2     1460 non-null   int64
36  BsmtUnfSF      1460 non-null   int64
37  TotalBsmtSF    1460 non-null   int64
38  Heating        1460 non-null   object
39  HeatingQC      1460 non-null   object
40  CentralAir     1460 non-null   object
41  Electrical     1459 non-null   object
42  1stFlrSF       1460 non-null   int64
43  2ndFlrSF       1460 non-null   int64
44  LowQualFinSF   1460 non-null   int64
45  GrLivArea      1460 non-null   int64
46  BsmtFullBath   1460 non-null   int64
47  BsmtHalfBath   1460 non-null   int64
48  FullBath       1460 non-null   int64
49  HalfBath       1460 non-null   int64
50  BedroomAbvGr   1460 non-null   int64
51  KitchebvGr     1460 non-null   int64
52  KitchenQual    1460 non-null   object
53  TotRmsAbvGrd   1460 non-null   int64
54  Functiol       1460 non-null   object
55  Fireplaces     1460 non-null   int64
56  FireplaceQu    770 non-null    object
57  GarageType     1379 non-null   object
58  GarageYrBlt    1379 non-null   float64
59  GarageFinish   1379 non-null   object
60  GarageCars     1460 non-null   int64
61  GarageArea     1460 non-null   int64
```

```
 62  GarageQual     1379 non-null   object
 63  GarageCond     1379 non-null   object
 64  PavedDrive     1460 non-null   object
 65  WoodDeckSF     1460 non-null   int64
 66  OpenPorchSF    1460 non-null   int64
 67  EnclosedPorch  1460 non-null   int64
 68  3SsnPorch      1460 non-null   int64
 69  ScreenPorch    1460 non-null   int64
 70  PoolArea       1460 non-null   int64
 71  PoolQC         7 non-null      object
 72  Fence          281 non-null    object
 73  MiscFeature    54 non-null     object
 74  MiscVal        1460 non-null   int64
 75  MoSold         1460 non-null   int64
 76  YrSold         1460 non-null   int64
 77  SaleType       1460 non-null   object
 78  SaleCondition  1460 non-null   object
 79  SalePrice      1460 non-null   int64
dtypes: float64(3), int64(34), object(43)
memory usage: 923.9+ KB
```

1.a. Identify the shape of the dataset

```
fe_df.shape
```

```
(1460, 80)
```

1.b. variables with null values

```
fe_df.isna().all().sum()
```

```
0
```

There is no column with all NaN values. Below are the columns with NaN values.

```
nan_counts = fe_df.isna().sum()
nan_counts[nan_counts > 0]
```

```
LotFrontage     259
Alley          1369
MasVnrType        8
MasVnrArea        8
BsmtQual         37
BsmtCond         37
BsmtExposure     38
BsmtFinType1     37
BsmtFinType2     38
Electrical        1
FireplaceQu     690
GarageType       81
GarageYrBlt      81
GarageFinish     81
GarageQual       81
```

```
GarageCond        81
PoolQC          1453
Fence           1179
MiscFeature     1406
dtype: int64
```

1.c Identify variables with unique values

```
for col in fe_df.columns:
    if (fe_df[col].nunique() == fe_df.shape[0]):
        print(col)
```

No column has all unique values.

2. Generate a separate dataset for numerical and categorical variables

```
numeric_cols = list(fe_df._get_numeric_data().columns)

categorical_cols = list(set(fe_df.columns) - set(numeric_cols))

fe_num_df = fe_df[numeric_cols]
fe_cat_df = fe_df[categorical_cols]

print(fe_num_df.shape)
print(fe_cat_df.shape)
```

```
(1460, 37)
(1460, 43)
```

3. EDA of numerical variables

a. Missing value treatment

```
nan_counts_num = fe_num_df.isna().sum()
nan_counts_num[nan_counts_num > 0]
```

```
LotFrontage    259
MasVnrArea       8
GarageYrBlt     81
dtype: int64
```

Above are the columns with missing value. We can replace it with median value.

```
fe_num_df['GarageYrBlt'].fillna(fe_num_df['GarageYrBlt'].median(),
inplace=True)
fe_num_df['MasVnrArea'].fillna(fe_num_df['MasVnrArea'].median(),
inplace=True)
fe_num_df['LotFrontage'].fillna(fe_num_df['LotFrontage'].median(),
inplace=True)

print(fe_num_df['GarageYrBlt'].isna().sum())
print(fe_num_df['MasVnrArea'].isna().sum())
print(fe_num_df['LotFrontage'].isna().sum())
```

```
0
0
0
```

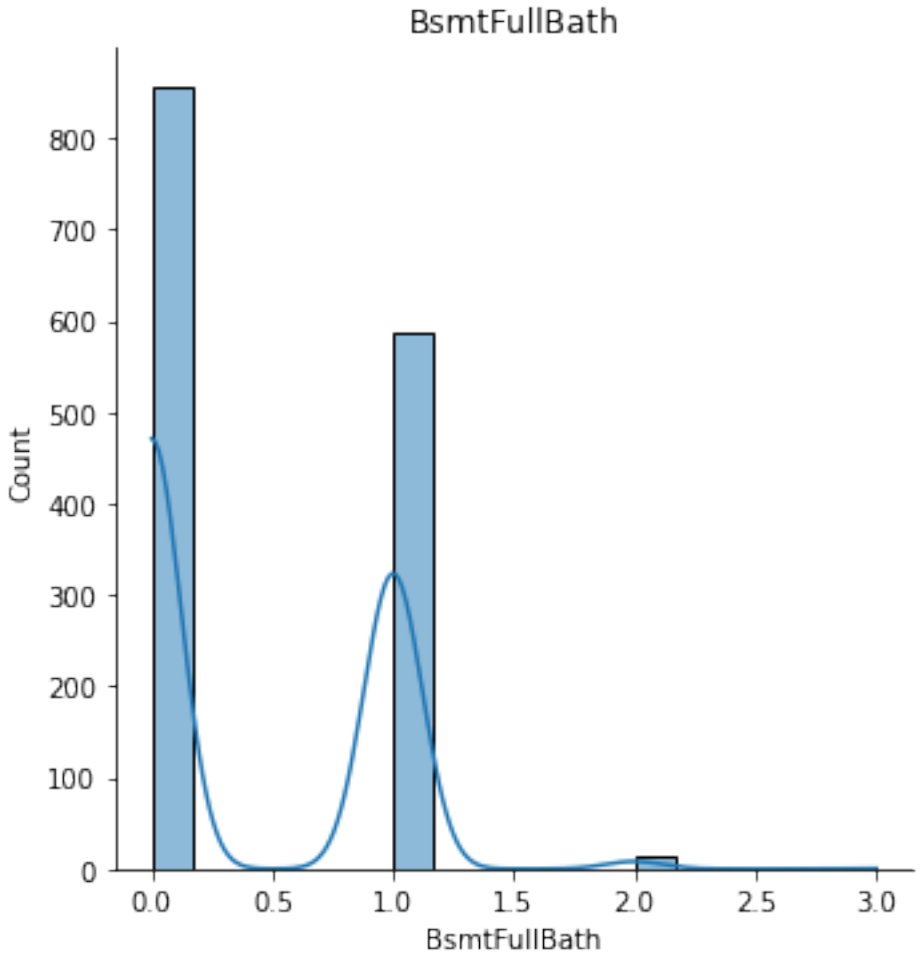
b. Identify the skewness and distribution

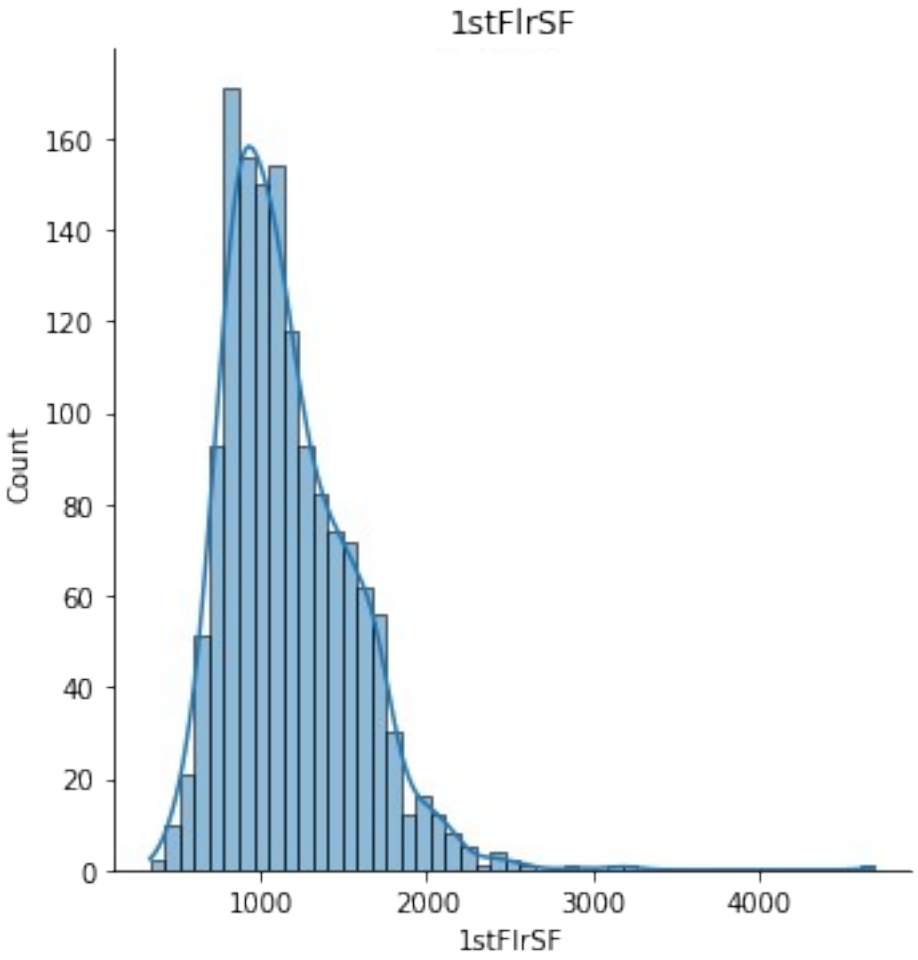
Lets pick random columns to get skewness and plot their distribution.

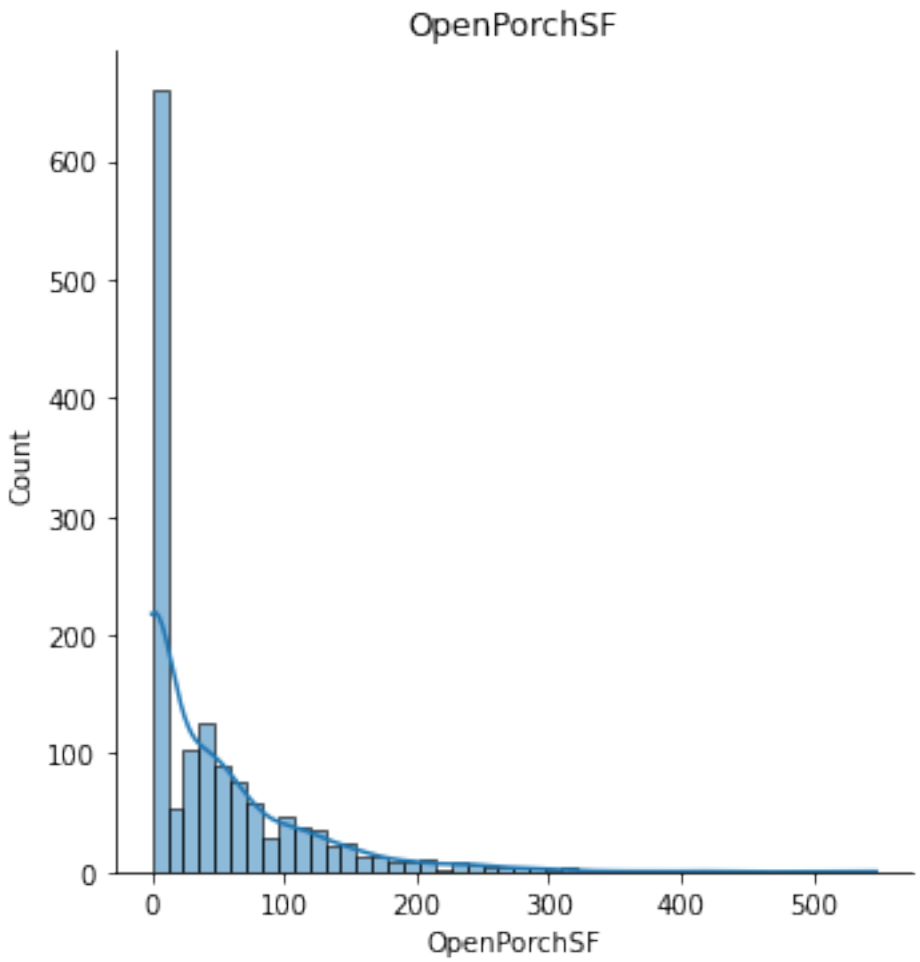
```
sample_numeric_cols = sample(numeric_cols, 4)
```

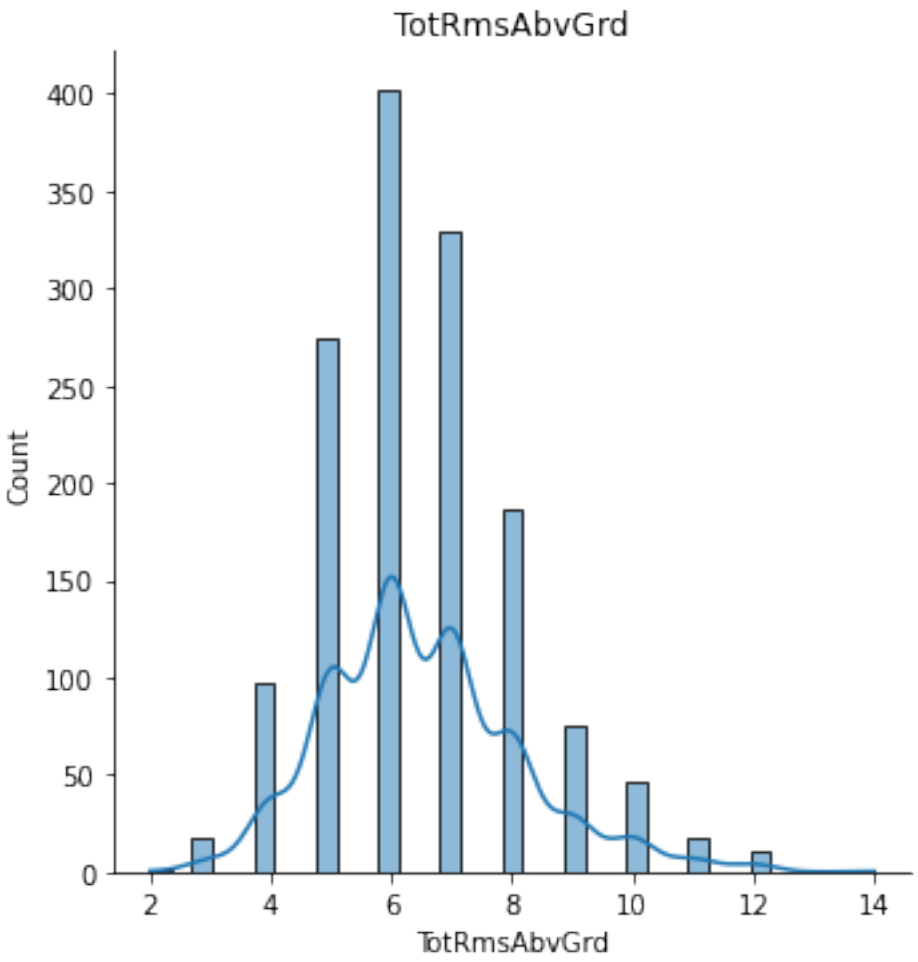
```
for col in sample_numeric_cols:
    col_val = fe_num_df[col]
    sns.displot(col_val, kde=True)
    plt.title(col)
    print('Skewness for column {}: {}'.format(col, skew(col_val)))
```

```
Skewness for column BsmtFullBath: 0.5954540376067279
Skewness for column 1stFlrSF: 1.3753417421837937
Skewness for column OpenPorchSF: 2.361911928568972
Skewness for column TotRmsAbvGrd: 0.6756457673102017
```



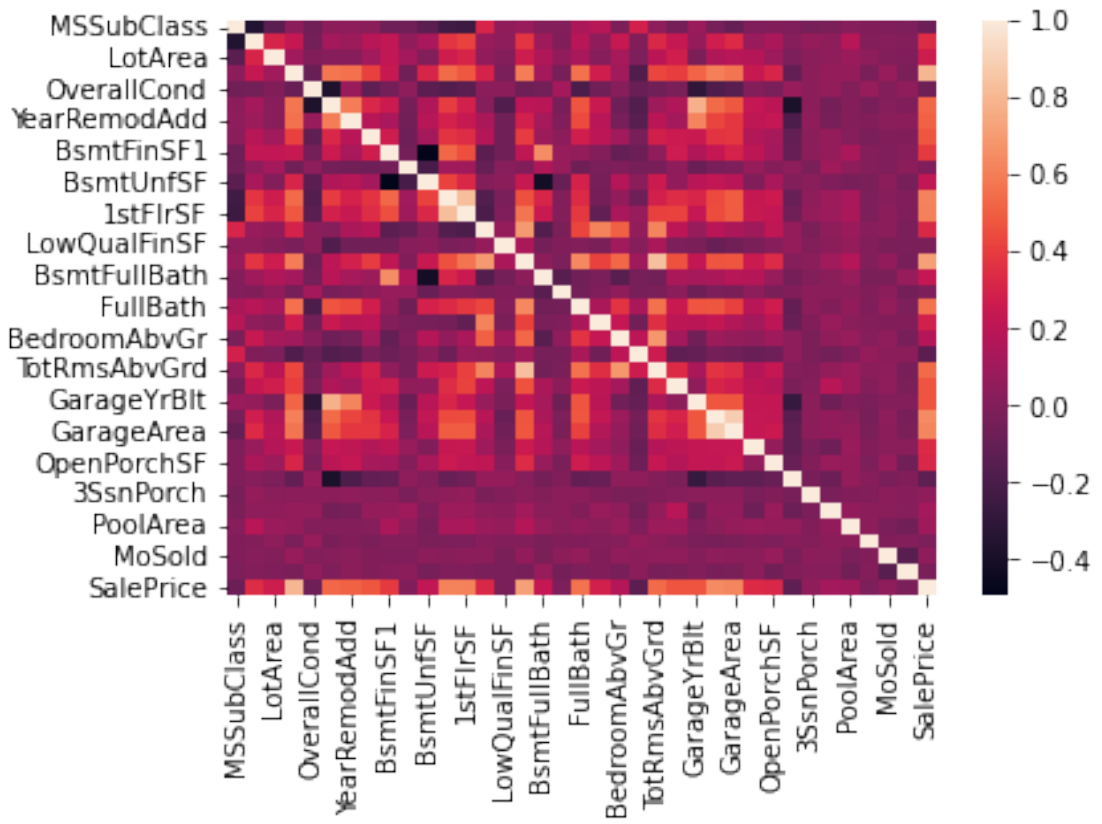

1stFlrSF
Count
160
140
120
100
80
60
40
20
0
1000
2000
3000
4000
1stFlrSF


OpenPorchSF
Count
600
500
400
300
200
100
0
0
100
200
300
400
500
OpenPorchSF



c. Identify significant variables using a correlation matrix

```
num_corr = fe_num_df.corr()

sns.heatmap(num_corr)
plt.show()
```



Lets find out highly correlated (>= 0.6) columns with the target column (Sales price).

```
corr_vals = num_corr['SalePrice']
corr_vals = corr_vals[(corr_vals > 0.6) | (corr_vals < -0.6)]
corr_vals
```

```
OverallQual    0.790982
TotalBsmtSF    0.613581
1stFlrSF       0.605852
GrLivArea      0.708624
GarageCars     0.640409
GarageArea     0.623431
SalePrice      1.000000
Name: SalePrice, dtype: float64
```

```
num_corr_cols = corr_vals.index[: -1].tolist()
len(num_corr_cols)
```

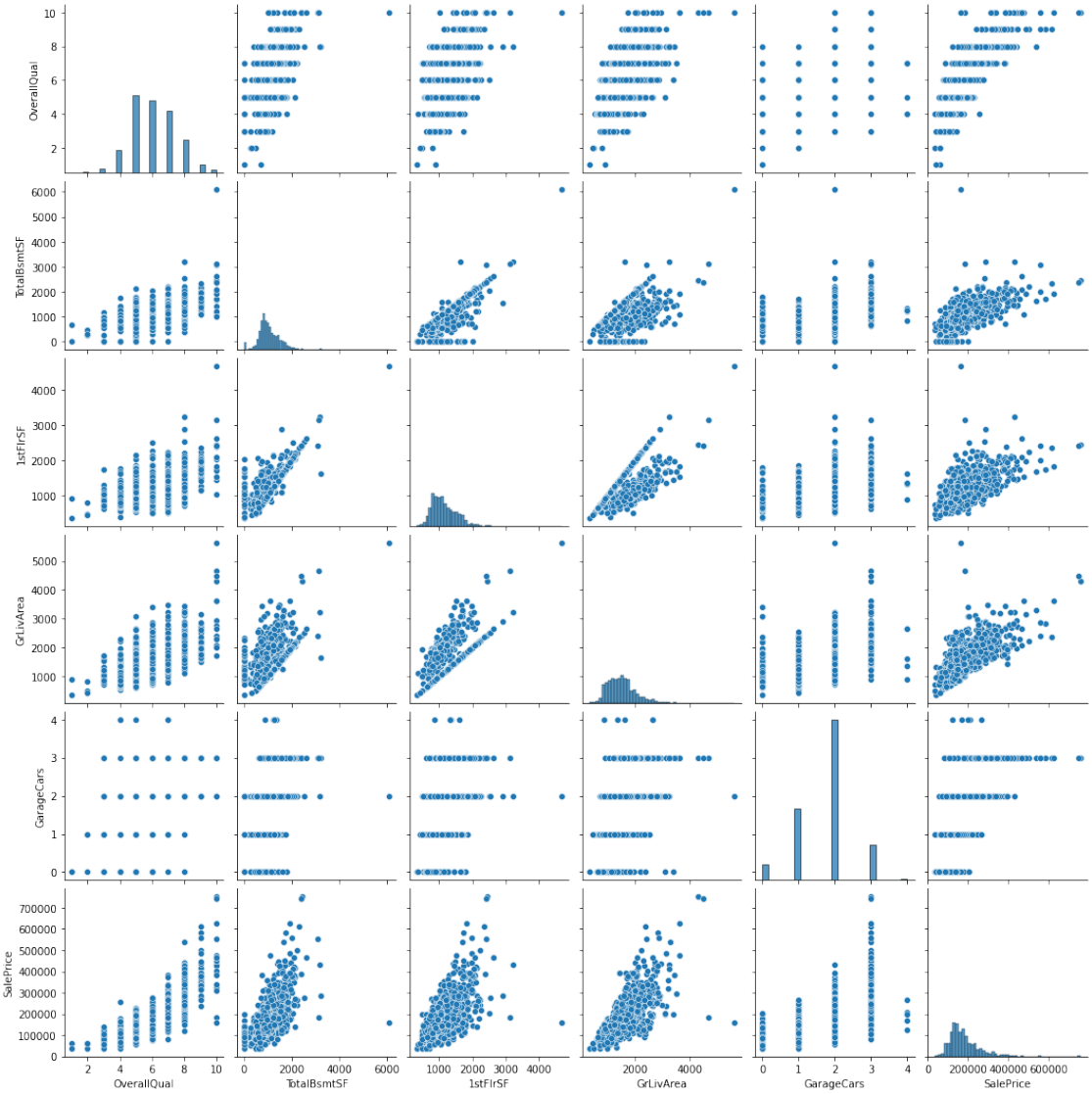
```
6
```

d. Pair plot for distribution and density

Lets go for a pairplot with 5 columns from the correlated columns and the SalePrice.

```
sample_num_df = fe_num_df[num_corr_cols[:-1] + ['SalePrice']]
sample_num_df.head()
```

```
    OverallQual  TotalBsmtSF  1stFlrSF  GrLivArea  GarageCars
SalePrice
Id

1              7          856       856       1710           2
208500
2              6         1262      1262       1262           2
181500
3              7          920       920       1786           2
223500
4              7          756       961       1717           3
140000
5              8         1145      1145       2198           3
250000

sns.pairplot(sample_num_df)
plt.show()
```



4. EDA of categorical variables

a. Missing value treatment

```
nan_counts_cat = fe_cat_df.isna().sum()
nan_counts_cat = nan_counts_cat[nan_counts_cat > 0]
nan_counts_cat
```

```
GarageCond        81
MasVnrType         8
GarageQual        81
MiscFeature     1406
Fence           1179
Alley           1369
BsmtQual          37
BsmtExposure      38
GarageFinish      81
FireplaceQu      690
BsmtCond          37
BsmtFinType2      38
Electrical         1
PoolQC          1453
BsmtFinType1      37
GarageType        81
dtype: int64
```

We can either delete the record or replace with mode value.

As there is a chance that we need to merge both datasets, its better not to delete any record.

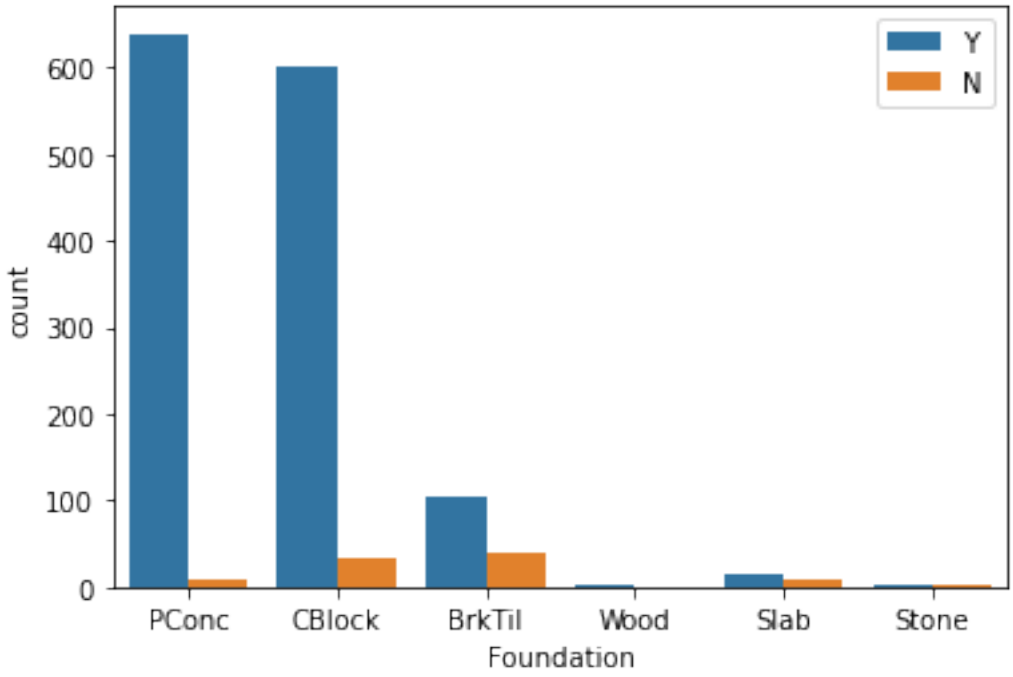
```
for col in nan_counts_cat.index:
    fe_cat_df[col] = fe_cat_df[col].fillna(fe_cat_df[col].mode()[0])

fe_cat_df.isna().sum().sum()
```

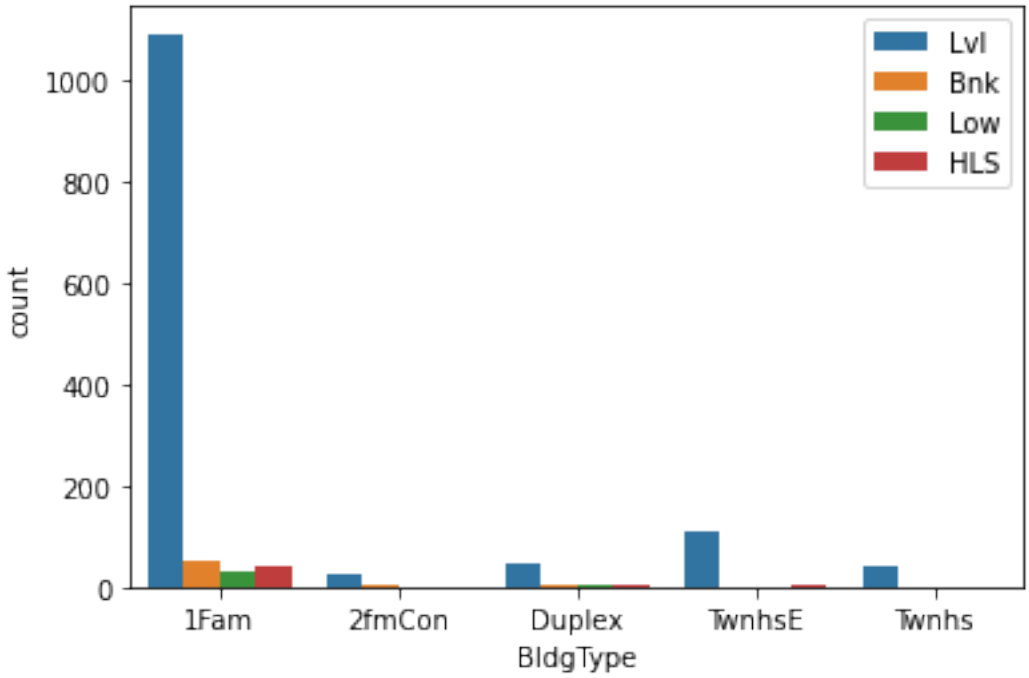
```
0
```

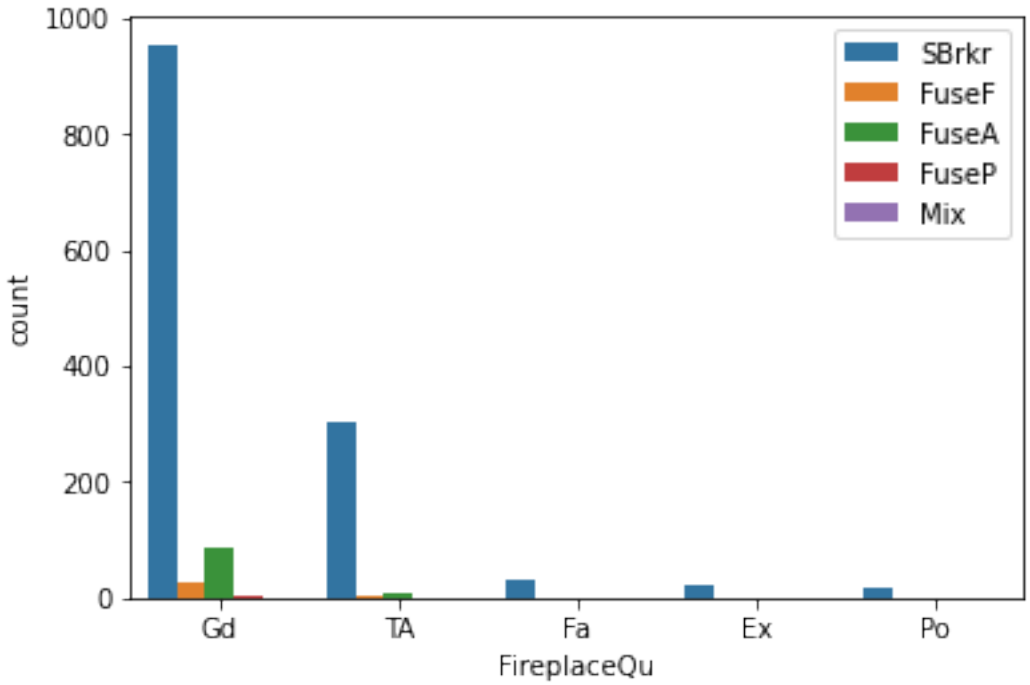
b. Count plot and box plot for bivariate analysis

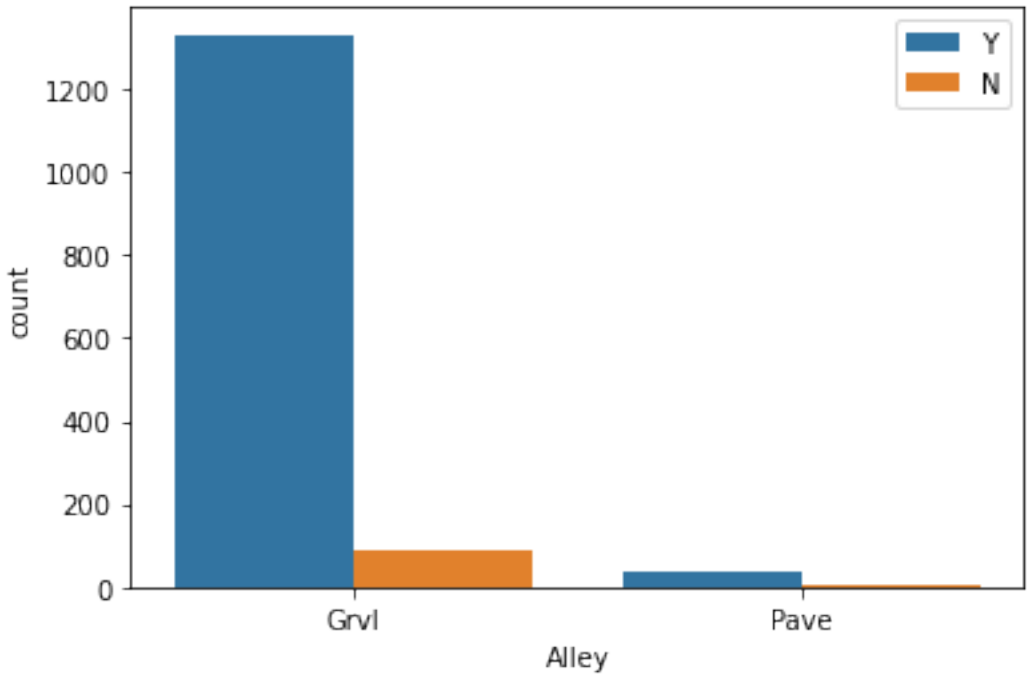
Lets take a sample of columns for EDA.

```
# Coun plot
for i in range(4):
    selected_cols = sample(categorical_cols, 2)
    sns.countplot(x=selected_cols[0], hue=selected_cols[1],
data=fe_cat_df[selected_cols])
    plt.legend(loc='upper right')
    plt.show()
```

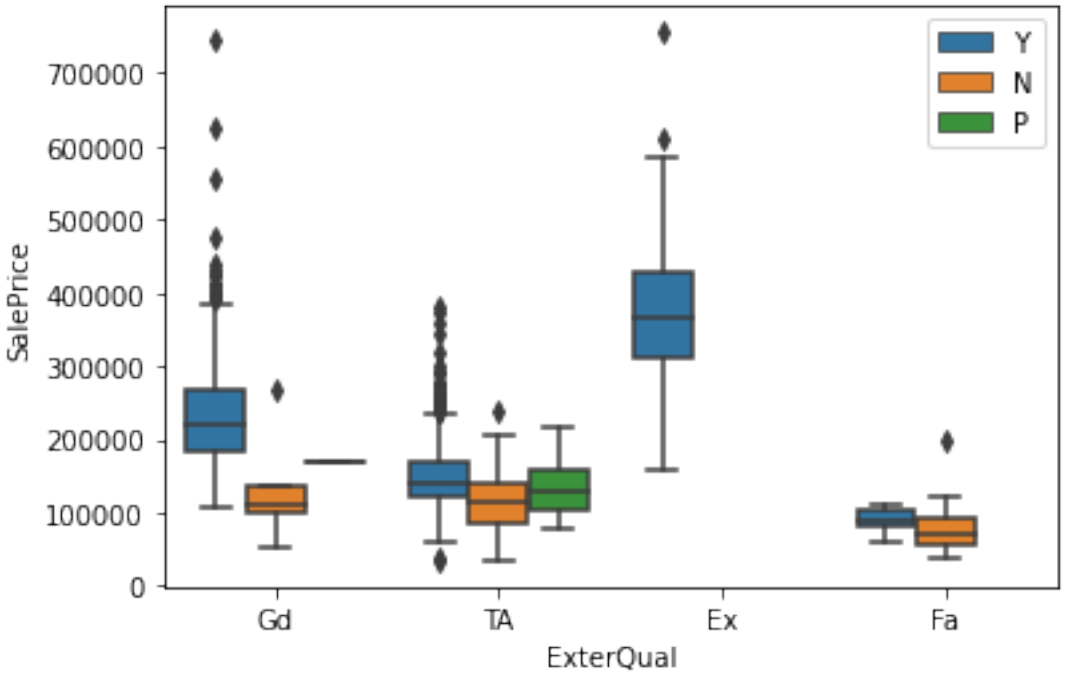

Y
N
600
500
400
300
200
100
0
count
PConc
CBlock
BrkTil
Wood
Slab
Stone
Foundation

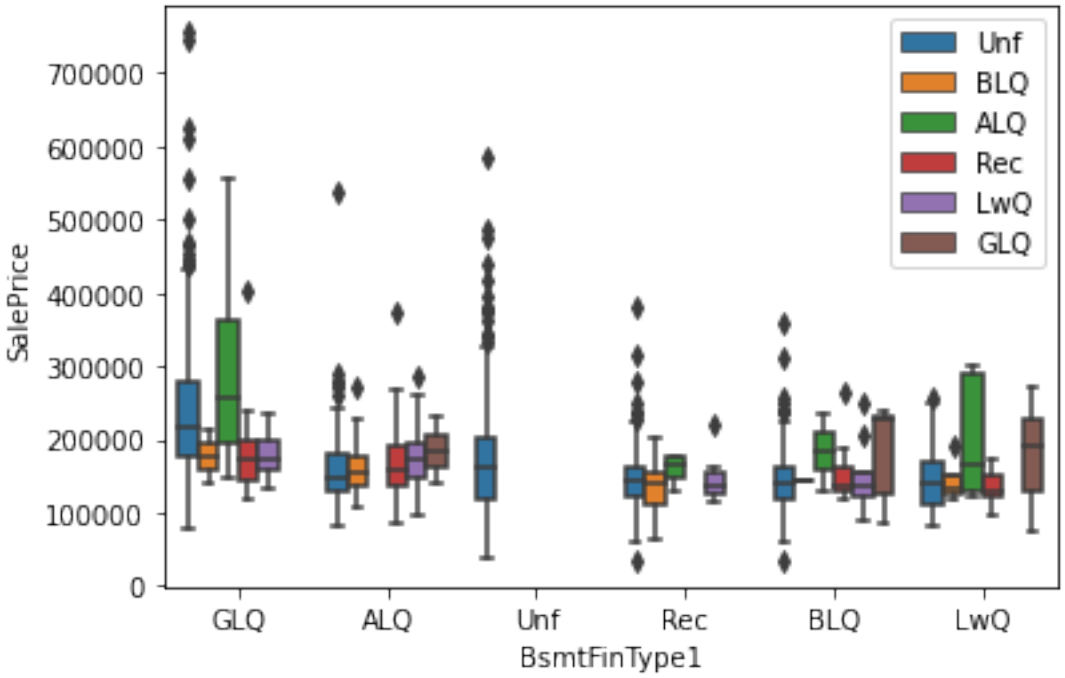

Lvl
Bnk
Low
HLS
1000
800
600
400
200
0
count
1Fam
2fmCon
Duplex
TwnhsE
Twnhs
BldgType

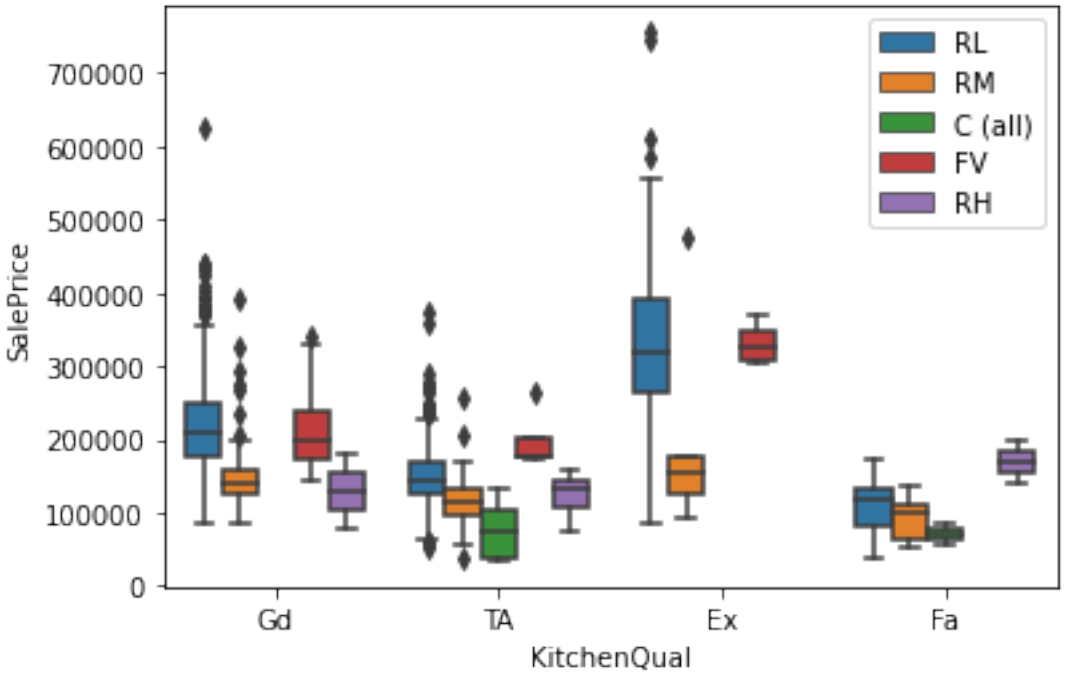


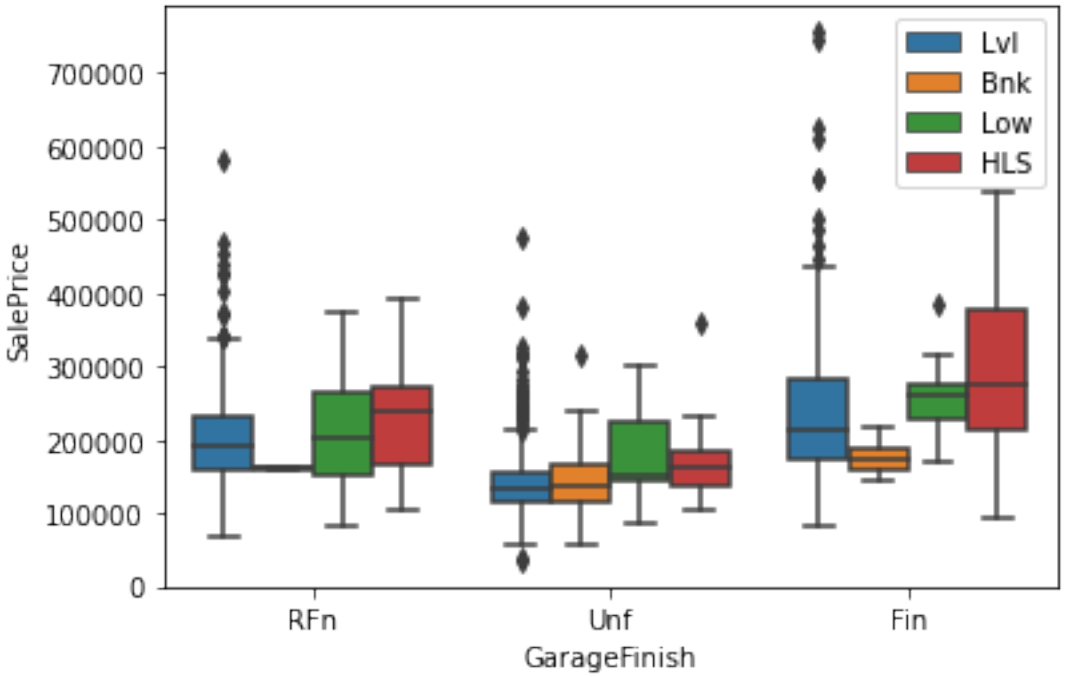



```
# Box plot
for i in range(4):
    selected_cols = sample(categorical_cols, 2)
    sns.boxplot(x =selected_cols[0], y ='SalePrice', data = fe_df, hue
=selected_cols[1])
    plt.legend(loc='upper right')
    plt.show()
```


Y
N
P
700000
600000
500000
400000
300000
200000
100000
0
SalePrice
Gd
TA
Ex
Fa
ExterQual


Unf
BLQ
ALQ
Rec
LwQ
GLQ
700000
600000
500000
400000
300000
200000
100000
0
SalePrice
GLQ
ALQ
Unf
Rec
BLQ
LwQ
BsmtFinType1





c. Identify significant variables using p-values and Chi-Square values

```
target = 'SalePrice'

cat_corr_cols = []

for feature in categorical_cols:
    fe_cat_df_cross_tab = pd.crosstab(index=fe_df[target],
columns=fe_df[feature])
```

```
    if (chi2_contingency(fe_cat_df_cross_tab)[1] < 0.05):
        cat_corr_cols.append(feature)

cat_corr_cols
```

```
['MasVnrType',
 'GarageQual',
 'LotConfig',
 'LotShape',
 'Heating',
 'KitchenQual',
 'BsmtQual',
 'BsmtExposure',
 'Neighborhood',
 'GarageFinish',
 'MSZoning',
 'FireplaceQu',
 'Street',
 'BsmtCond',
 'CentralAir',
 'ExterQual',
 'Foundation',
 'SaleType',
 'ExterCond',
 'SaleCondition']
```

5. Combine all the significant categorical and numerical variables

```
selected_cols = num_corr_cols + cat_corr_cols
len(selected_cols)
```

```
26
```

```
new_fe_df = fe_df[selected_cols]
```

```
num_nan_cols = ['GarageYrBlt', 'MasVnrArea', 'LotFrontage']
cat_nan_cols = list(nan_counts_cat.index)
```

```
nan_cols = new_fe_df.columns[new_fe_df.isna().sum() > 0]
```

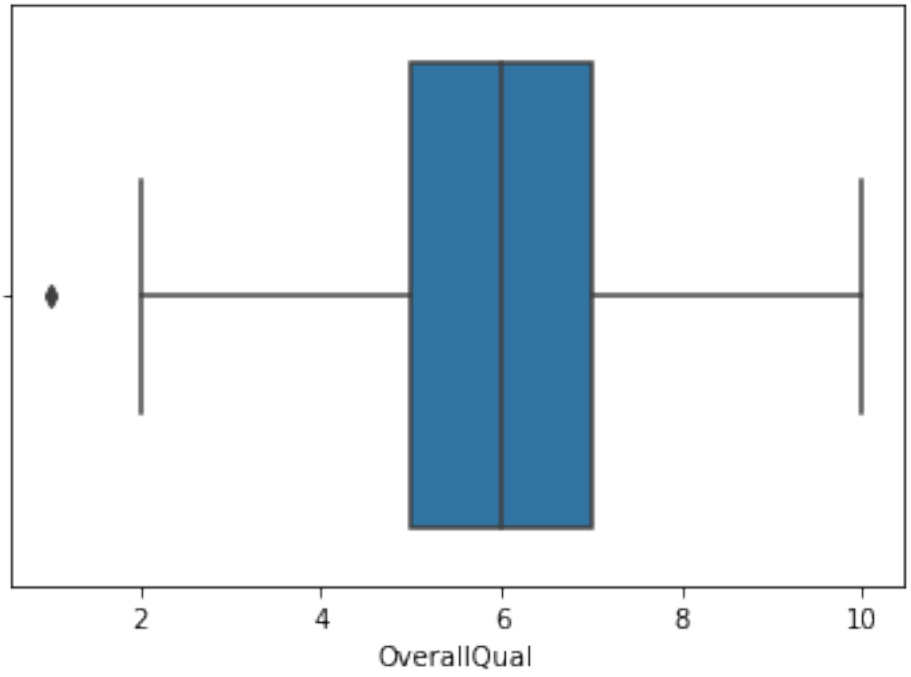
```
for col in nan_cols:
    if col in num_nan_cols:
        new_fe_df[col].fillna(new_fe_df[col].median(), inplace=True)
    elif col in cat_nan_cols:
        new_fe_df[col].fillna(new_fe_df[col].mode()[0], inplace=True)
```

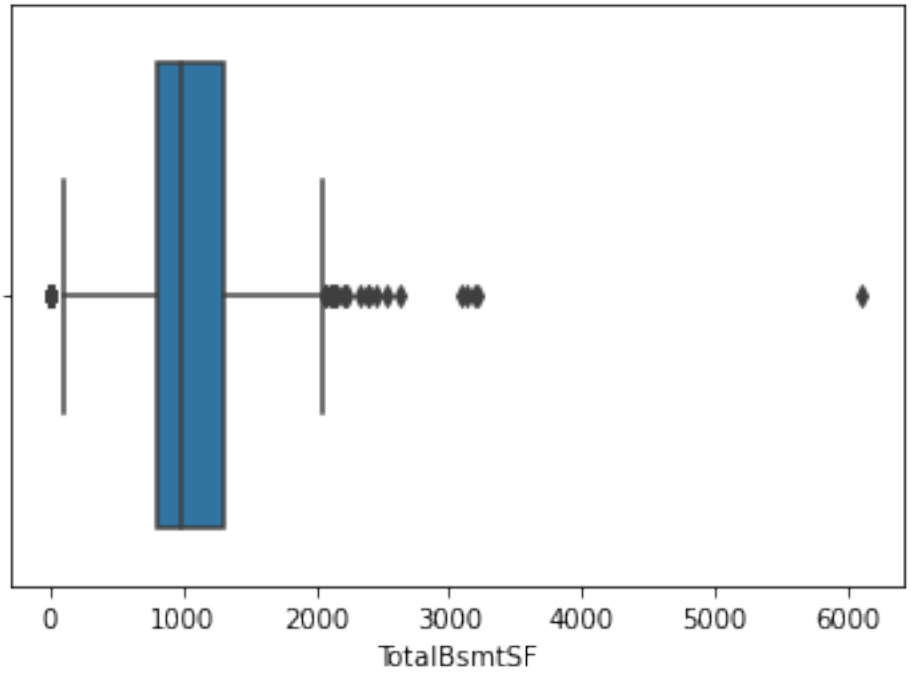
```
new_fe_df.isna().sum().sum()
```

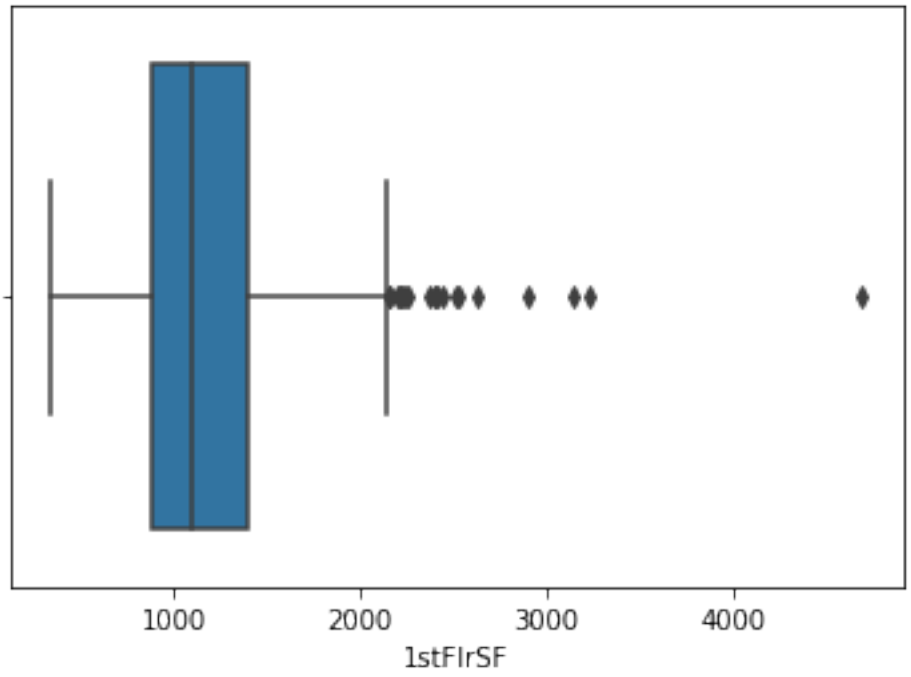
```
0
```

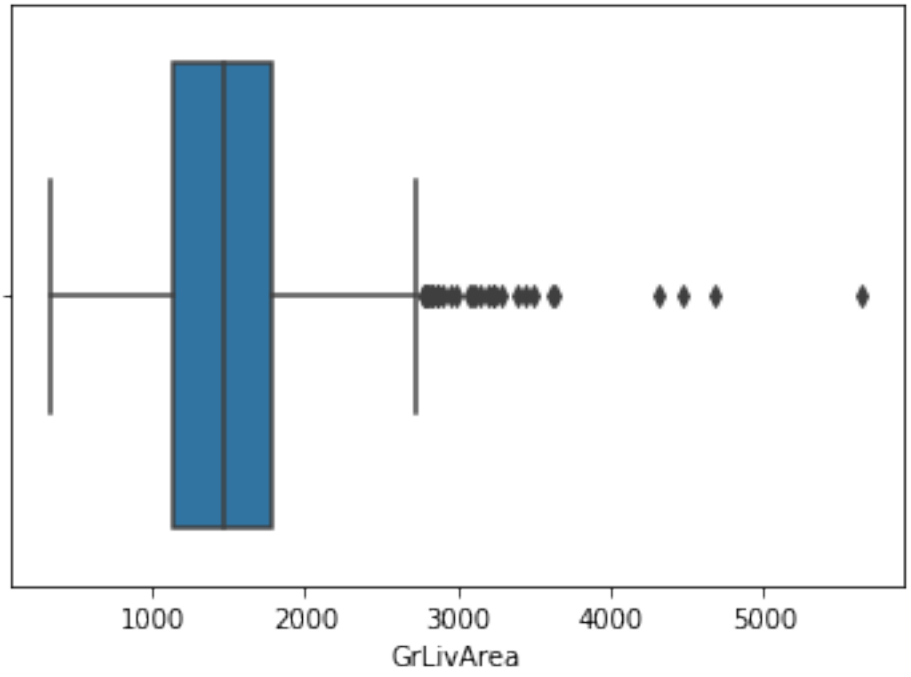
6. Plot box plot for the new dataset to find the variables with outliers

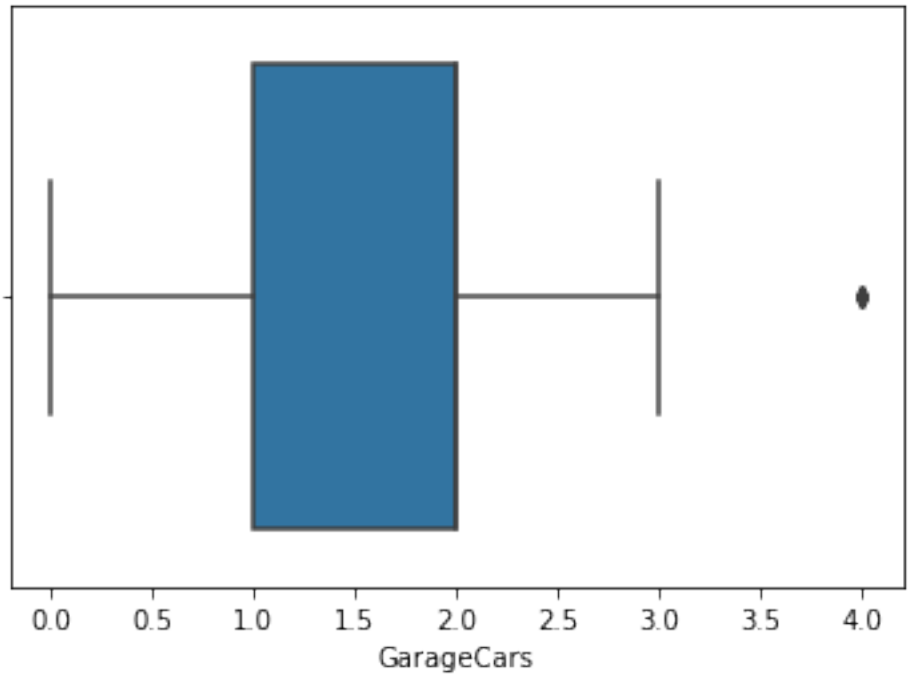
```
for col in num_corr_cols:
    sns.boxplot(new_fe_df[col])
    plt.show()
```

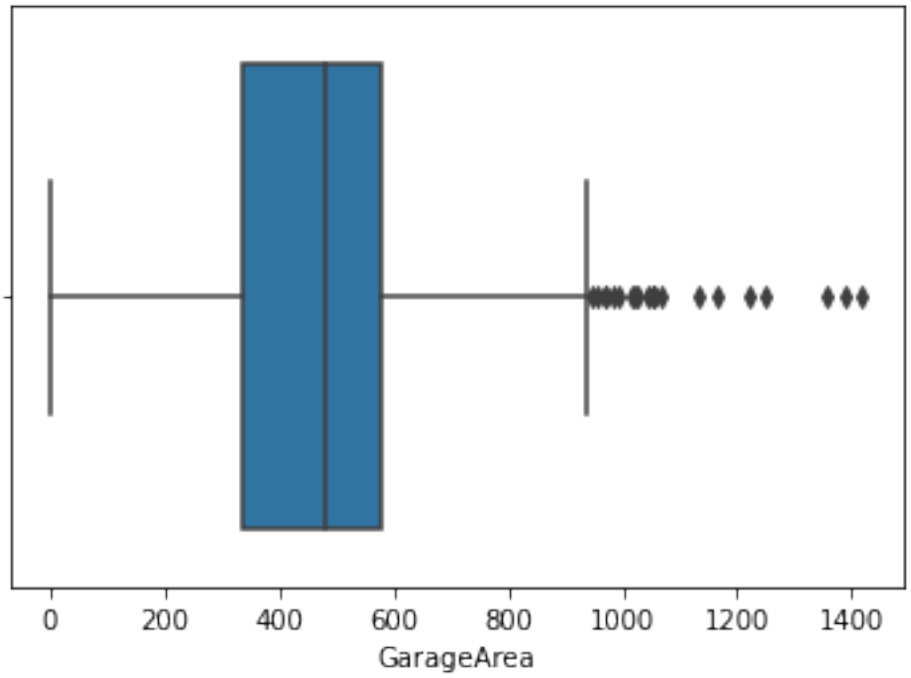





1000
2000
3000
4000
1stFlrSF


1000
2000
3000
4000
5000
GrLivArea

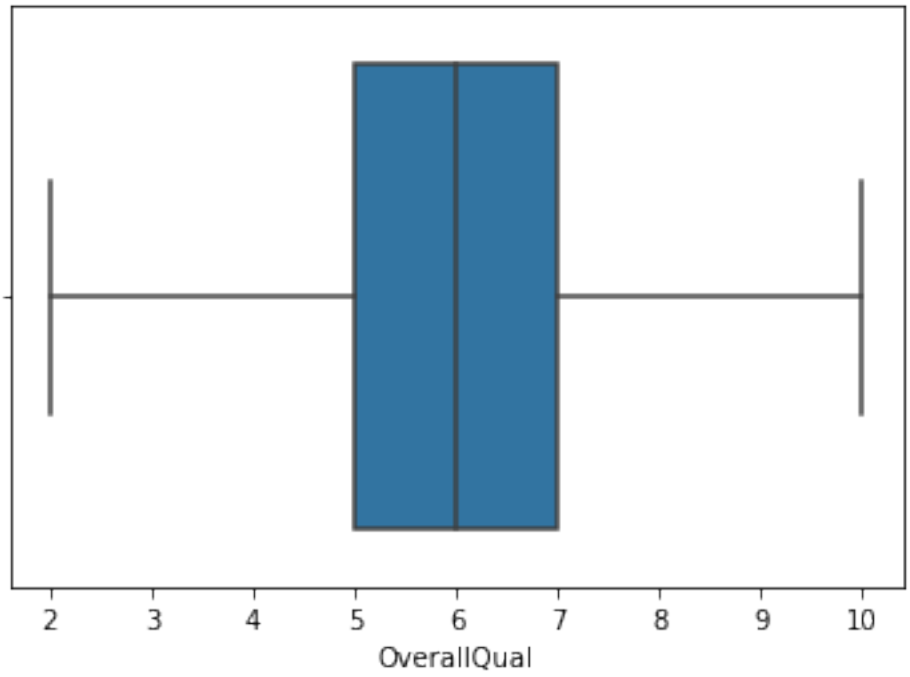


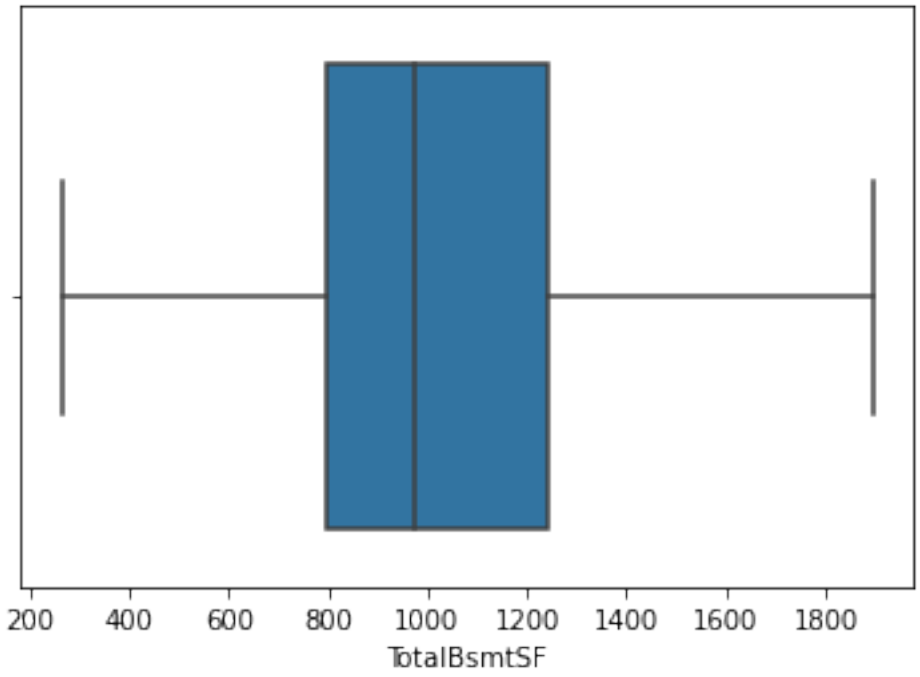


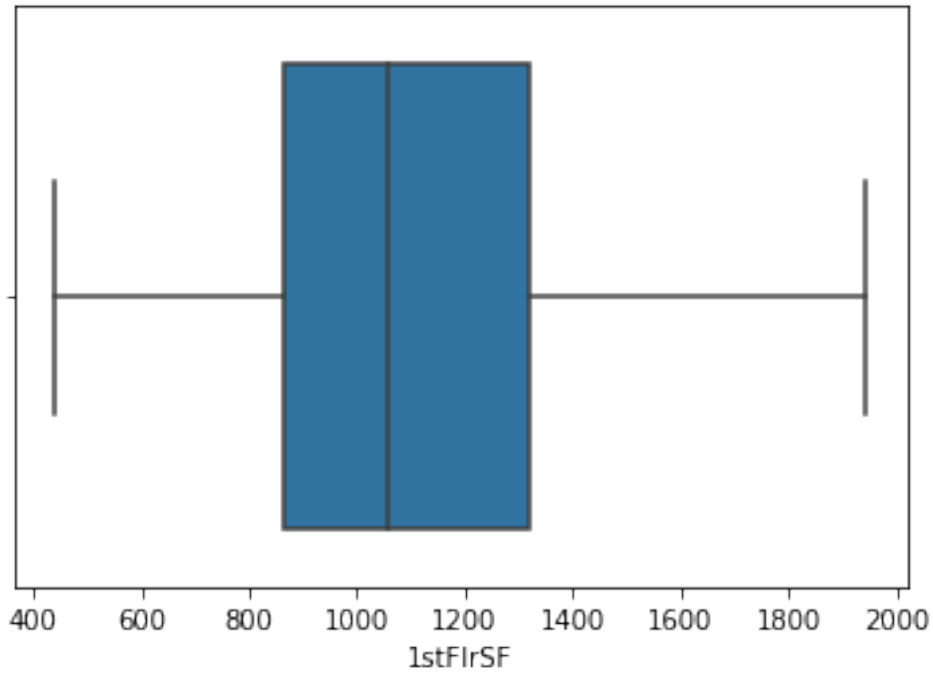

```
num_corr_cols
```

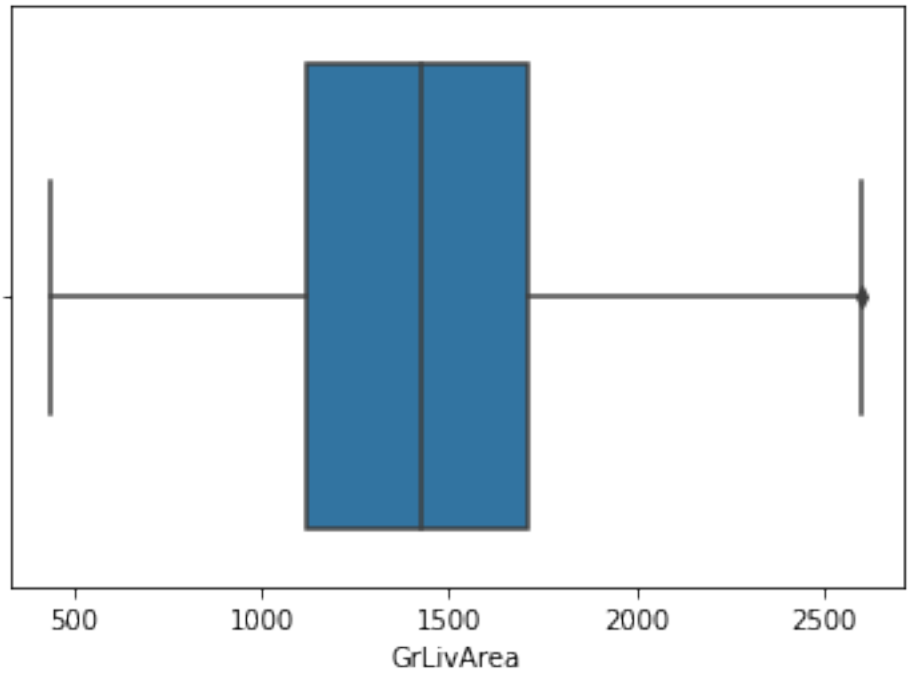
```
['OverallQual',
 'TotalBsmtSF',
 '1stFlrSF',
 'GrLivArea',
 'GarageCars',
 'GarageArea']
```

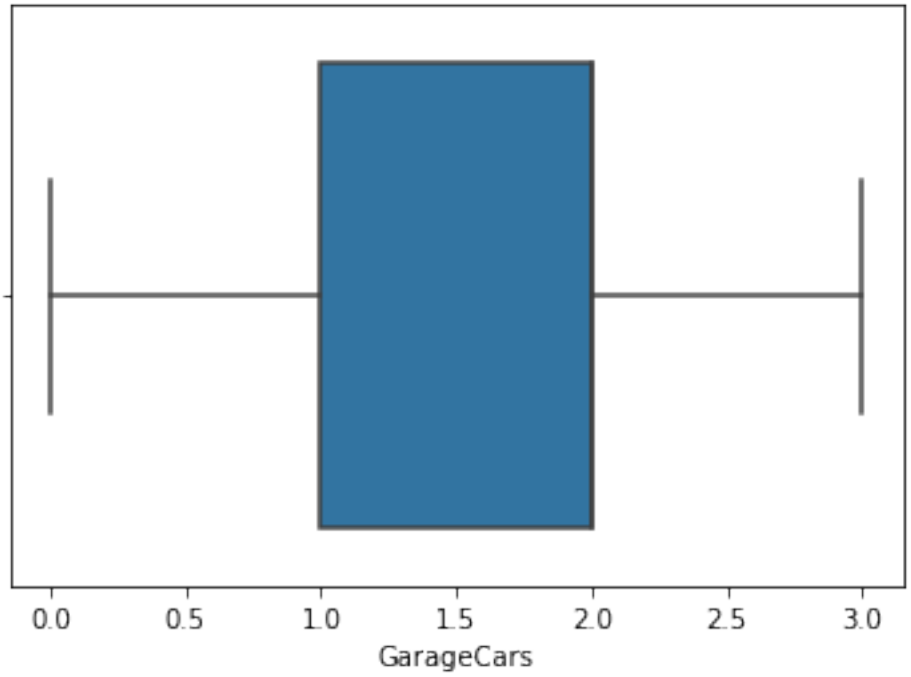
```
new_fe_df = new_fe_df[new_fe_df['OverallQual'] >= 2]
new_fe_df = new_fe_df[(new_fe_df['TotalBsmtSF'] >= 200) &
(new_fe_df['TotalBsmtSF'] <= 1900)]
new_fe_df = new_fe_df[new_fe_df['1stFlrSF'] <= 2000]
new_fe_df = new_fe_df[new_fe_df['GrLivArea'] <= 2600]
new_fe_df = new_fe_df[new_fe_df['GarageCars'] <= 3.0]
new_fe_df = new_fe_df[new_fe_df['GarageArea'] <= 900]

for col in num_corr_cols:
    sns.boxplot(new_fe_df[col])
    plt.show()
```

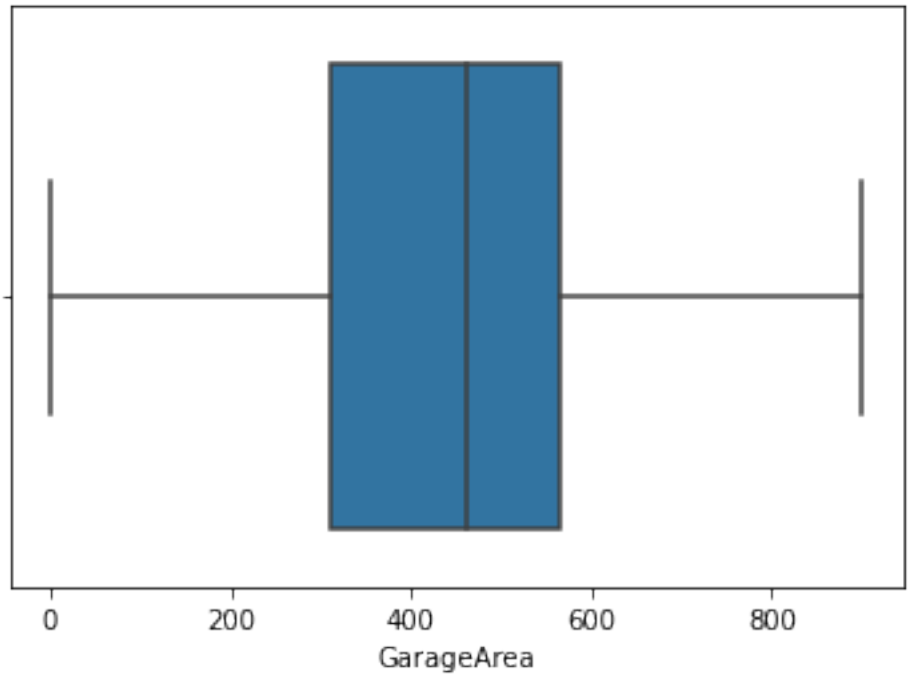



200
400
600
800
1000
1200
1400
1600
1800
TotalBsmtSF


400
600
800
1000
1200
1400
1600
1800
2000
1stFlrSF


500
1000
1500
2000
2500
GrLivArea


0.0
0.5
1.0
1.5
2.0
2.5
3.0
GarageCars



Now data is ready for modeling.