EDA Capstone Project 2

In [62]:

```
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
import seaborn as sns
%matplotlib inline
pd.pandas.set_option('display.max_columns',None)
```

In [63]:

```
pwd
```

Out[63]:

```
'C:\\Users\\Admin\\Desktop\\SPRINGBOARDFILES\\Unit 11\\Unit 11.5'
```

Data Collection

Load the data from CSV File

In [64]:

```
df=pd.read_csv(r'C:\Users\Admin\Desktop\SPRINGBOARDFILES\Unit 11\Unit 11.5\house-prices-a
dvanced-regression-techniques\Wrangled.csv')
```

In [65]:

```
df.head()
```

Out[65]:

	Unnamed: 0	Id	MSSubClass	MSZoning	LotFrontage	LotArea	Street	LotShape	LandContour	Utilities	LotConfig	LandSl
0	0	1	60	RL	65.0	8450	Pave	Reg	Lvl	AllPub	Inside	
1	1	2	20	RL	80.0	9600	Pave	Reg	Lvl	AllPub	FR2	
2	2	3	60	RL	68.0	11250	Pave	IR1	Lvl	AllPub	Inside	
3	3	4	70	RL	60.0	9550	Pave	IR1	Lvl	AllPub	Corner	
4	4	5	60	RL	84.0	14260	Pave	IR1	Lvl	AllPub	FR2	

In [66]:

```
df.shape
```

Out[66]:

```
(1460, 77)
```

In [67]:

```
df.columns
```

Out[67]:

```
Index(['Unnamed: 0', 'Id', 'MSSubClass', 'MSZoning', 'LotFrontage', 'LotArea',
       'Street', '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', 'KitchenAbvGr', 'KitchenQual',
       'TotRmsAbvGrd', 'Functional', 'Fireplaces', 'FireplaceQu', 'GarageType',
       'GarageFinish', 'GarageCars', 'GarageArea', 'GarageQual', 'GarageCond',
       'PavedDrive', 'WoodDeckSF', 'OpenPorchSF', 'EnclosedPorch', '3SsnPorch',
       'ScreenPorch', 'PoolArea', 'MiscVal', 'MoSold', 'YrSold', 'SaleType',
       'SaleCondition', 'SalePrice'],
      dtype='object')
```

Calculating the percentage of null values in each feture

Numerical Variables

In [68]:

```
numeric_features=df.select_dtypes(include=[np.number])

numeric_features.columns
```

Out[68]:

```
Index(['Unnamed: 0', 'Id', 'MSSubClass', 'LotFrontage', 'LotArea',
       'OverallQual', 'OverallCond', 'YearBuilt', 'YearRemodAdd', 'MasVnrArea',
       'BsmtFinSF1', 'BsmtFinSF2', 'BsmtUnfSF', 'TotalBsmtSF', '1stFlrSF',
       '2ndFlrSF', 'LowQualFinSF', 'GrLivArea', 'BsmtFullBath', 'BsmtHalfBath',
       'FullBath', 'HalfBath', 'BedroomAbvGr', 'KitchenAbvGr', 'TotRmsAbvGrd',
       'Fireplaces', 'GarageCars', 'GarageArea', 'WoodDeckSF', 'OpenPorchSF',
       'EnclosedPorch', '3SsnPorch', 'ScreenPorch', 'PoolArea', 'MiscVal',
       'MoSold', 'YrSold', 'SalePrice'],
      dtype='object')
```

Discrete numeric variables

In [69]:

```
## Numerical variables are usually of 2 type
## 1. Continous variable and Discrete Variables

discrete_feature=[feature for feature in numeric_features if len(df[feature].unique())<2
5 and feature not in year_feature+['Id']]
print("Discrete Variables Count: {}".format(len(discrete_feature)))
```

```
Discrete Variables Count: 17
```

In [70]:

```
discrete_feature
```

Out[70]:

```
['MSSubClass',
 'OverallQual',
 'OverallCond',
 'LowQualFinSF',
 'BsmtFullBath',
 'BsmtHalfBath',
 'FullBath',
 'HalfBath',
 'BedroomAbvGr',
 'KitchenAbvGr',
 'TotRmsAbvGrd',
 'Fireplaces',
 'GarageCars',
 '3SsnPorch',
```

```
 'PoolArea',
 'MiscVal',
 'MoSold']
```

In [71]:

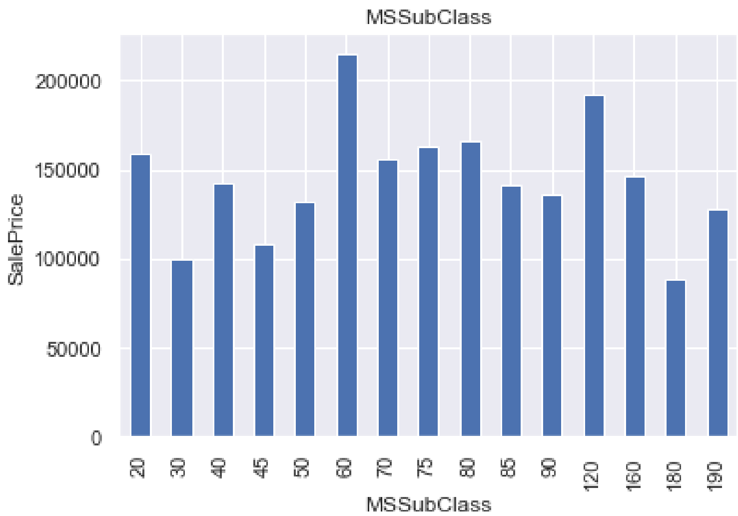
```
df[discrete_feature].head()
```

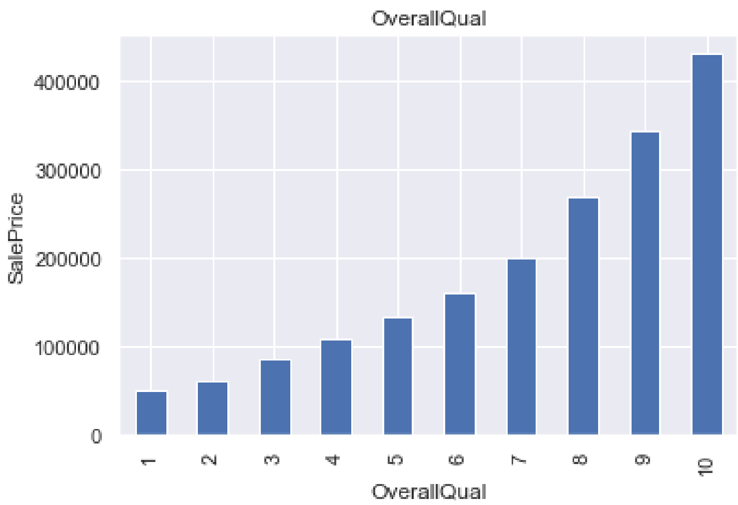
Out[71]:

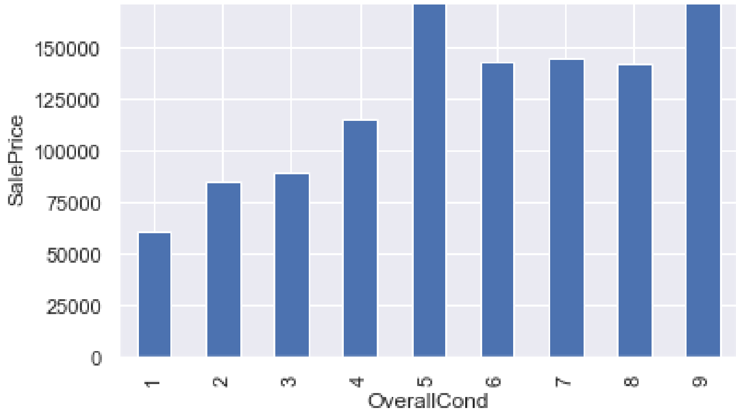
	MSSubClass	OverallQual	OverallCond	LowQualFinSF	BsmtFullBath	BsmtHalfBath	FullBath	HalfBath	BedroomAbvGr	Ki
0	60	7	5	0	1	0	2	1	3	
1	20	6	8	0	0	1	2	0	3	
2	60	7	5	0	1	0	2	1	3	
3	70	7	5	0	1	0	1	0	3	
4	60	8	5	0	1	0	2	1	4	

In [72]:

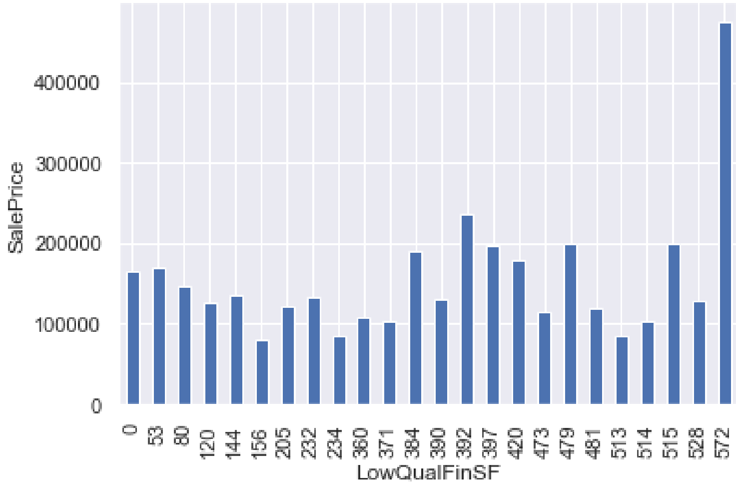
```
## Lets Find the realtionship between them and Sale PRice

for feature in discrete_feature:
    data=df.copy()
    data.groupby(feature)['SalePrice'].median().plot.bar()
    plt.xlabel(feature)
    plt.ylabel('SalePrice')
    plt.title(feature)
    plt.show()
```



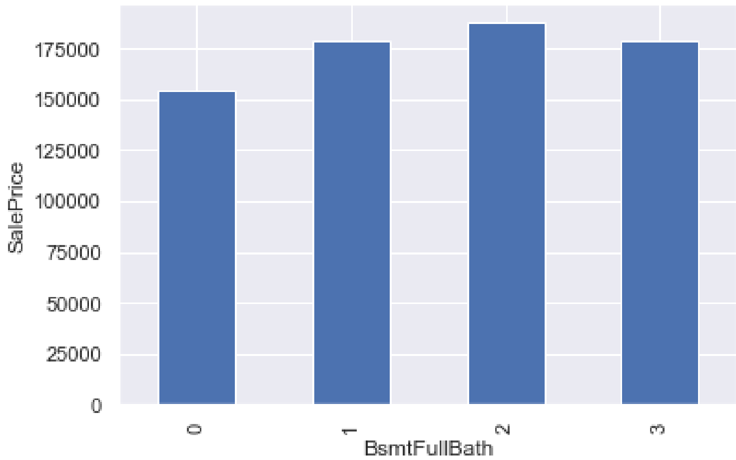




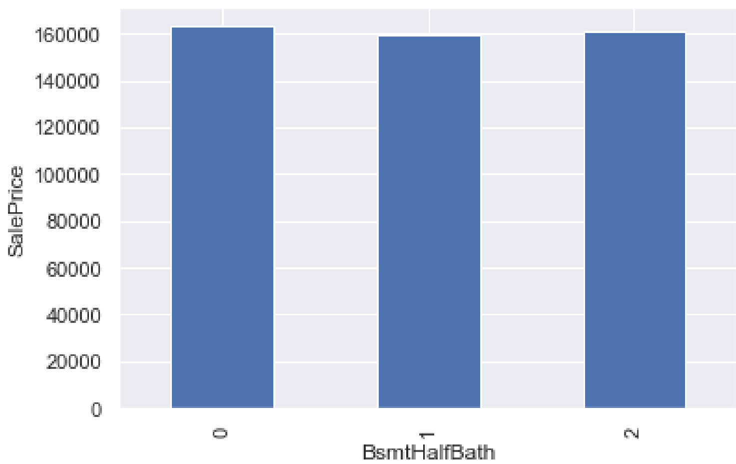
LowQualFinSF



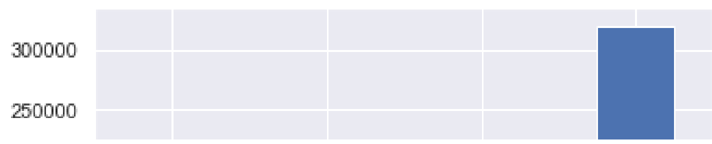
BsmtFullBath

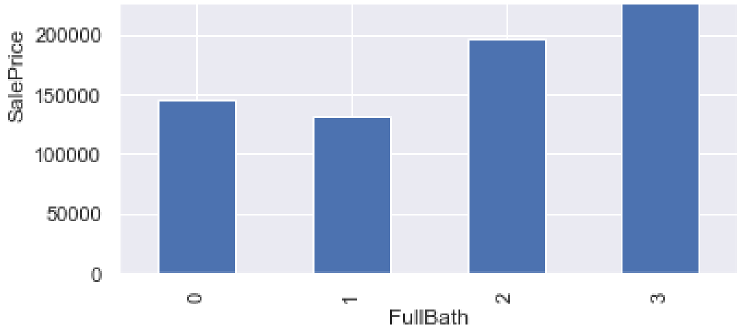


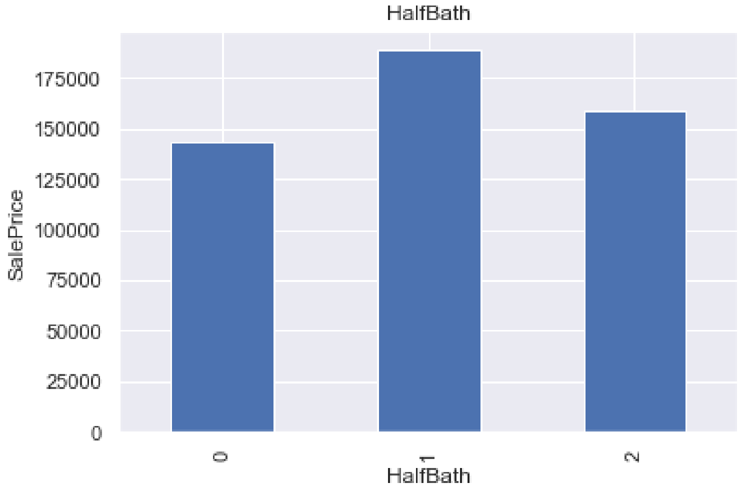
BsmtHalfBath

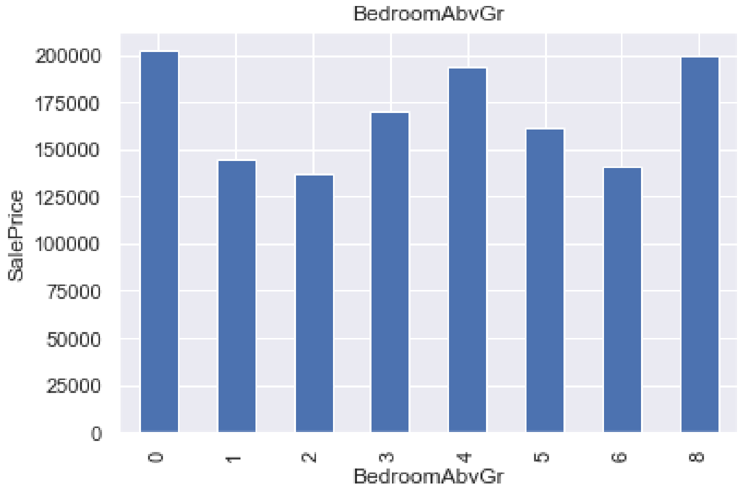


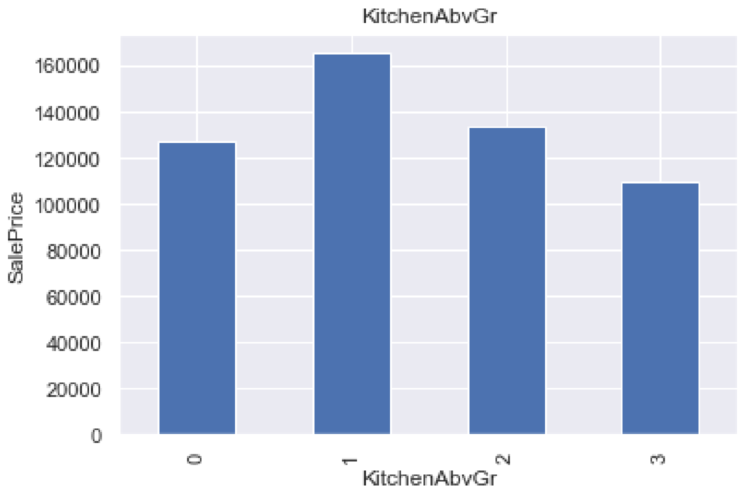
FullBath

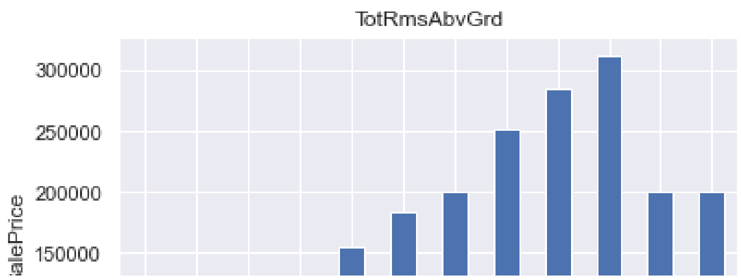


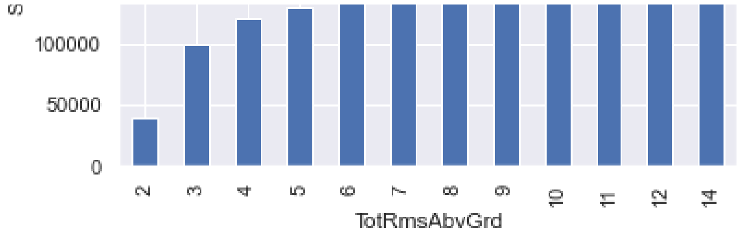

SalePrice
200000
150000
100000
50000
0
0
1
2
3
FullBath

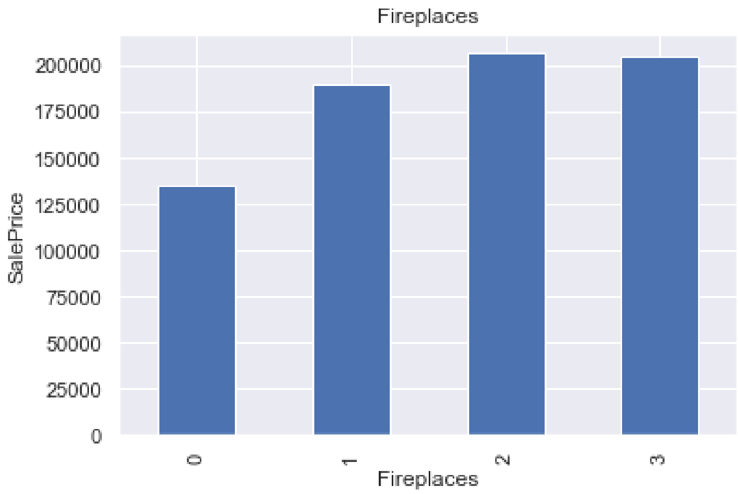

HalfBath
SalePrice
175000
150000
125000
100000
75000
50000
25000
0
0
1
2
HalfBath

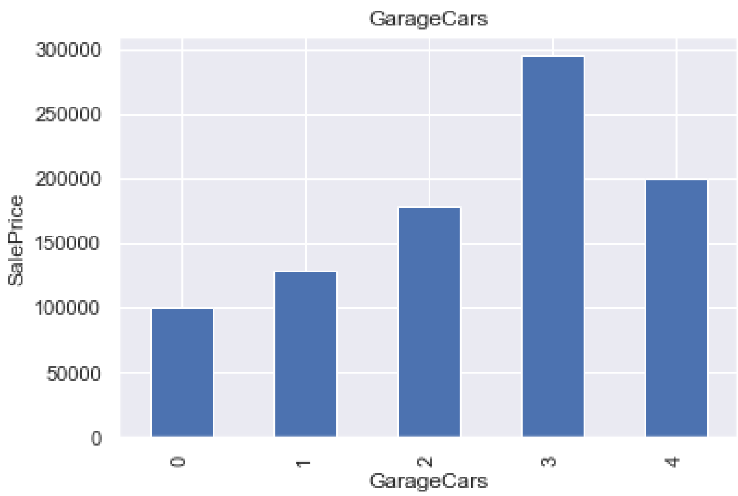

BedroomAbvGr
SalePrice
200000
175000
150000
125000
100000
75000
50000
25000
0
0
1
2
3
4
5
6
8
BedroomAbvGr

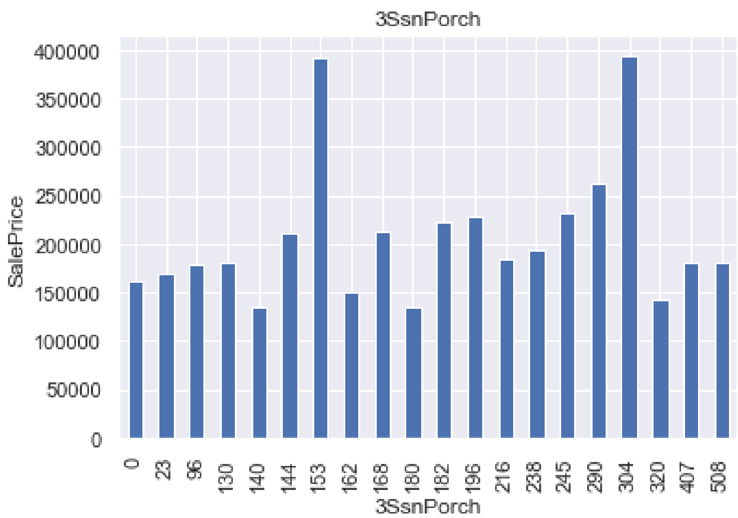

KitchenAbvGr
SalePrice
160000
140000
120000
100000
80000
60000
40000
20000
0
0
1
2
3
KitchenAbvGr

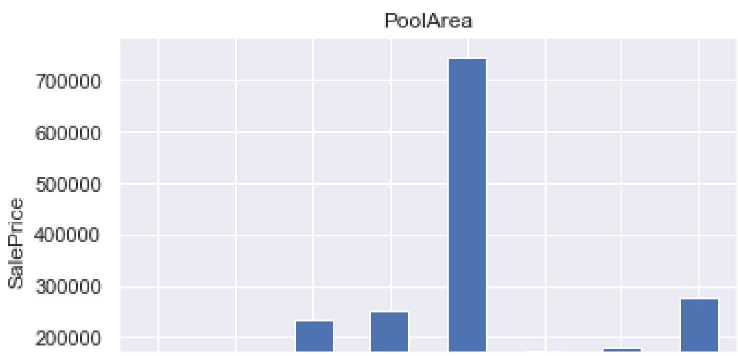

TotRmsAbvGrd
SalePrice
300000
250000
200000
150000

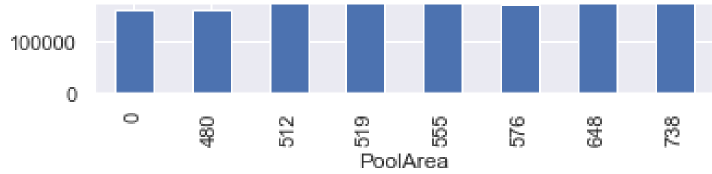

S
100000
50000
0
2
3
4
5
6
7
8
9
10
11
12
14
TotRmsAbvGrd

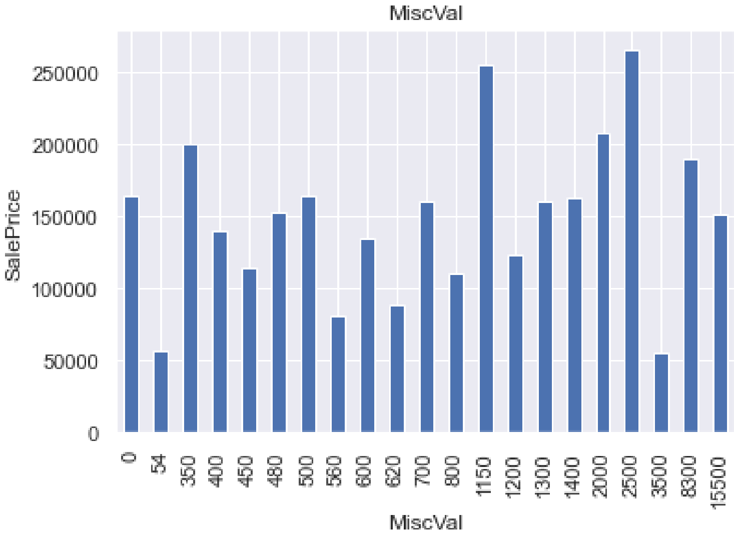

Fireplaces
SalePrice
200000
175000
150000
125000
100000
75000
50000
25000
0
0
1
2
3
Fireplaces

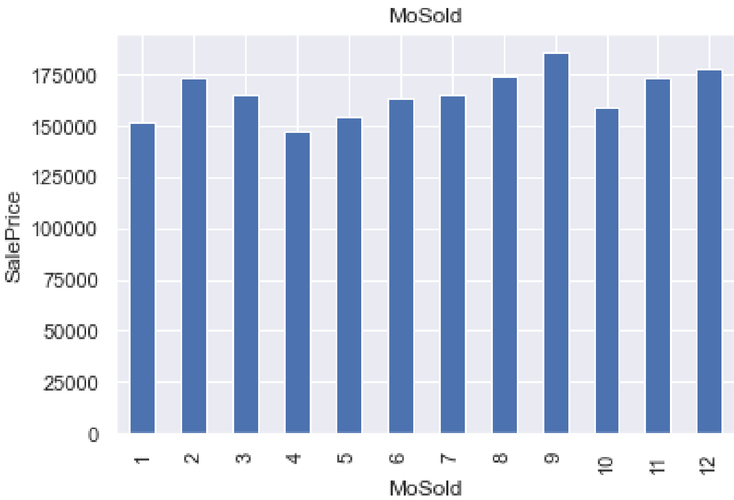

GarageCars
SalePrice
300000
250000
200000
150000
100000
50000
0
0
1
2
3
4
GarageCars


3SsnPorch
SalePrice
400000
350000
300000
250000
200000
150000
100000
50000
0
0
23
96
130
140
144
153
162
168
180
182
196
216
238
245
290
304
320
407
508
3SsnPorch


PoolArea
SalePrice
700000
600000
500000
400000
300000
200000







In [ ]:

In [ ]:

Correlation of all the numerical variables with respect to the target variable 'SaleProce'

In [73]:

```
correlation=numeric_features.corr()
print(correlation['SalePrice'].sort_values(ascending=False))
```

```
SalePrice        1.000000
OverallQual      0.790982
GrLivArea        0.708624
GarageCars       0.640409
GarageArea       0.623431
TotalBsmtSF      0.613581
1stFlrSF         0.605852
FullBath         0.560664
TotRmsAbvGrd     0.533723
YearBuilt        0.522897
YearRemodAdd     0.507101
MasVnrArea       0.475241
Fireplaces       0.466929
```

```
BsmtFinSF1       0.386420
LotFrontage      0.334901
WoodDeckSF       0.324413
2ndFlrSF         0.319334
OpenPorchSF      0.315856
HalfBath         0.284108
LotArea          0.263843
BsmtFullBath     0.227122
BsmtUnfSF        0.214479
BedroomAbvGr     0.168213
ScreenPorch      0.111447
PoolArea         0.092404
MoSold           0.046432
3SsnPorch        0.044584
BsmtFinSF2      -0.011378
BsmtHalfBath    -0.016844
MiscVal         -0.021190
Id              -0.021917
Unnamed: 0      -0.021917
LowQualFinSF    -0.025606
YrSold          -0.028923
OverallCond     -0.077856
MSSubClass      -0.084284
EnclosedPorch   -0.128578
KitchenAbvGr    -0.135907
Name: SalePrice, dtype: float64
```

In [74]:

```
year_feature = [feature for feature in numerical_features if 'Yr' in feature or 'Year' in feature]
```

Continuous numerical variables

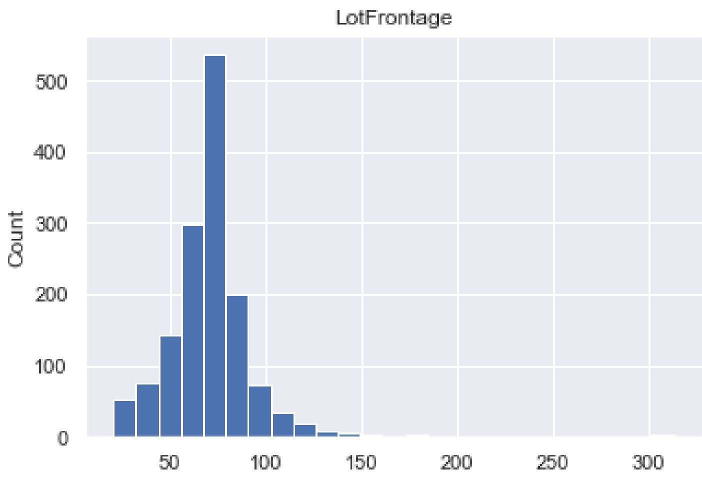
In [75]:

```
continuous_feature=[feature for feature in numeric_features if feature not in discrete_feature+year_feature+['Id','Unnamed: 0']]
print("Continuous feature Count {}".format(len(continuous_feature)))
```

```
Continuous feature Count 16
```

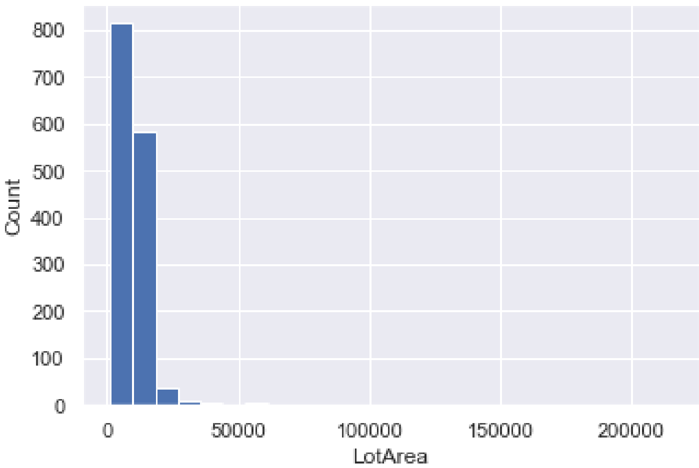
In [76]:

```
## Lets analyse the continuous values by creating histograms to understand the distribution

for feature in continuous_feature:
    data=df.copy()
    data[feature].hist(bins=25)
    plt.xlabel(feature)
    plt.ylabel("Count")
    plt.title(feature)
    plt.show()
```

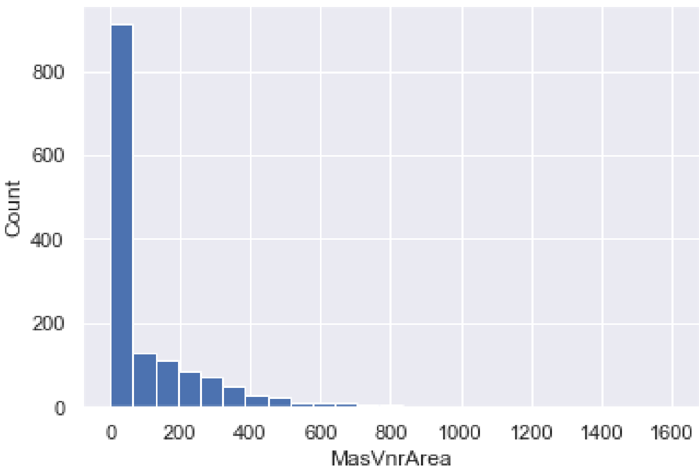


LotFrontage

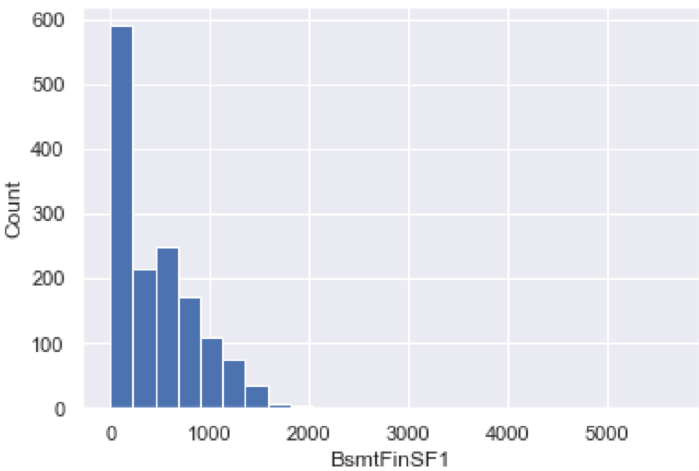
LotArea



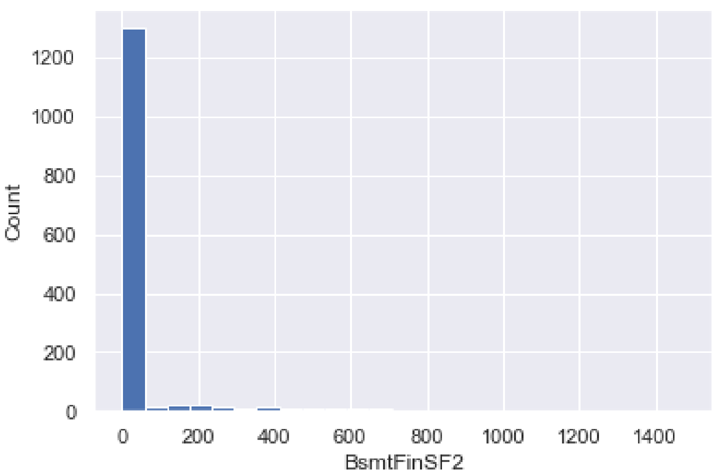
MasVnrArea

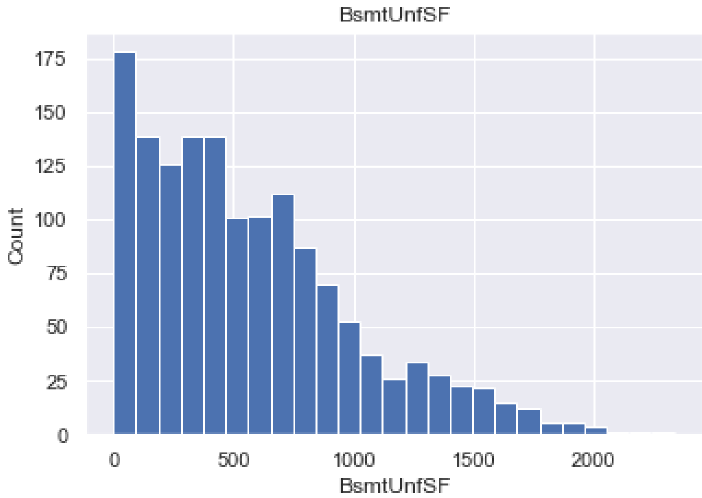


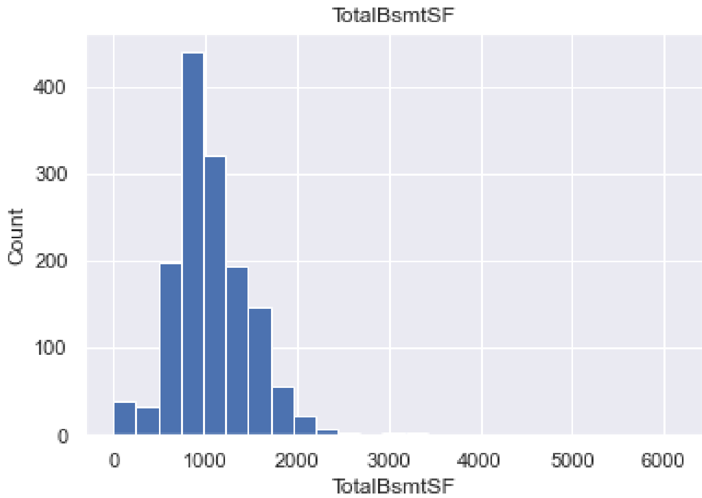
BsmtFinSF1

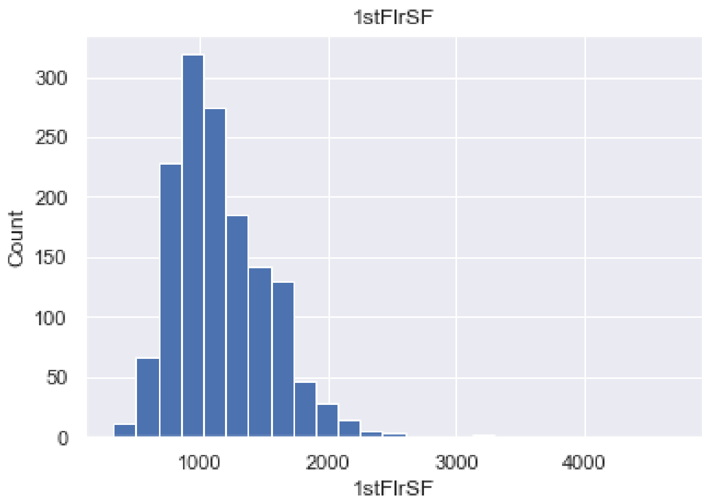


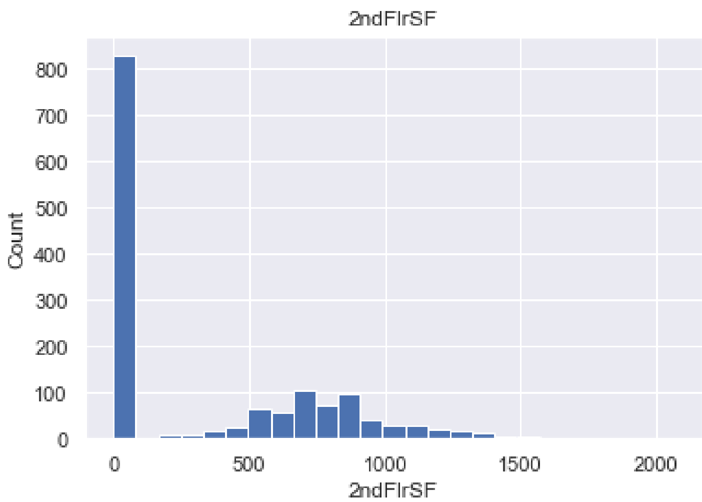
BsmtFinSF2



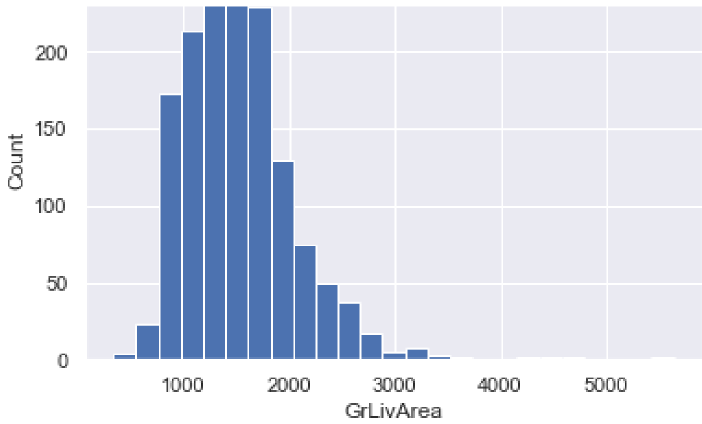

BsmtUnfSF
Count
BsmtUnfSF


TotalBsmtSF
Count
TotalBsmtSF

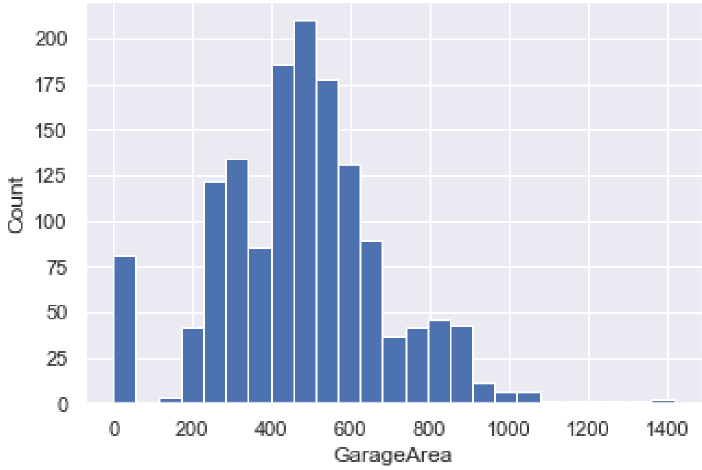

1stFlrSF
Count
1stFlrSF


2ndFlrSF
Count
2ndFlrSF

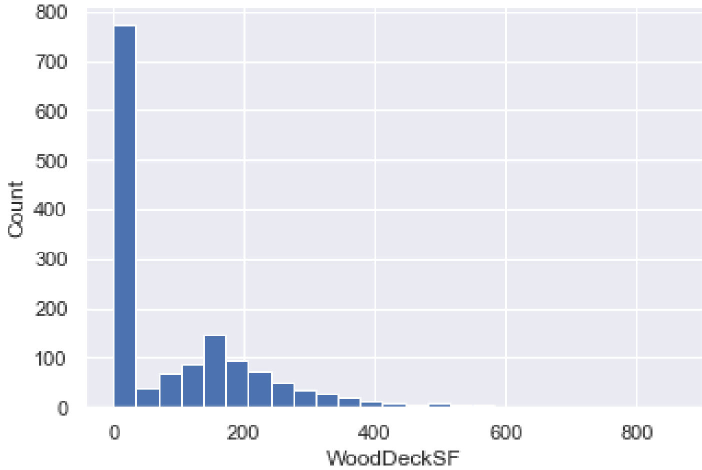

GrLivArea



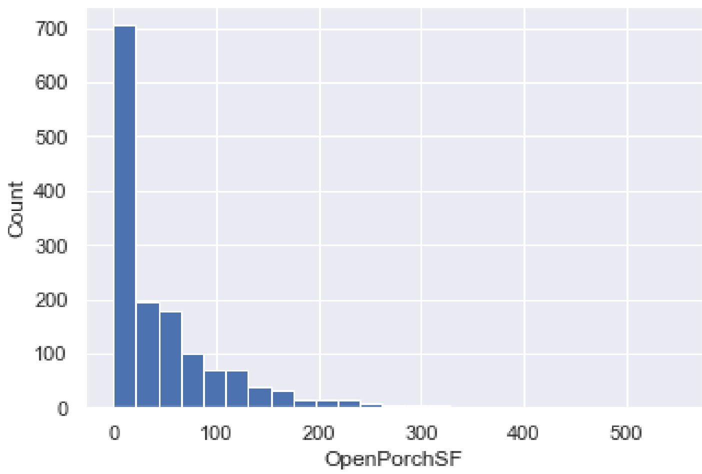
GarageArea



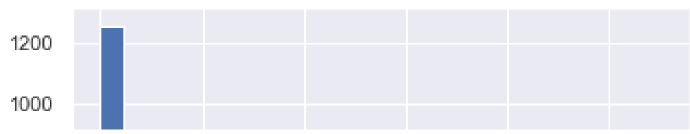
WoodDeckSF

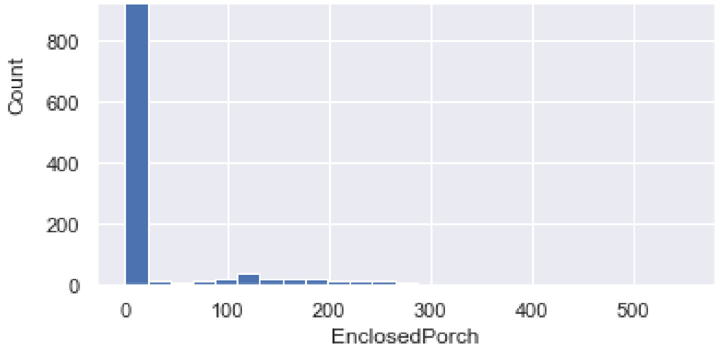


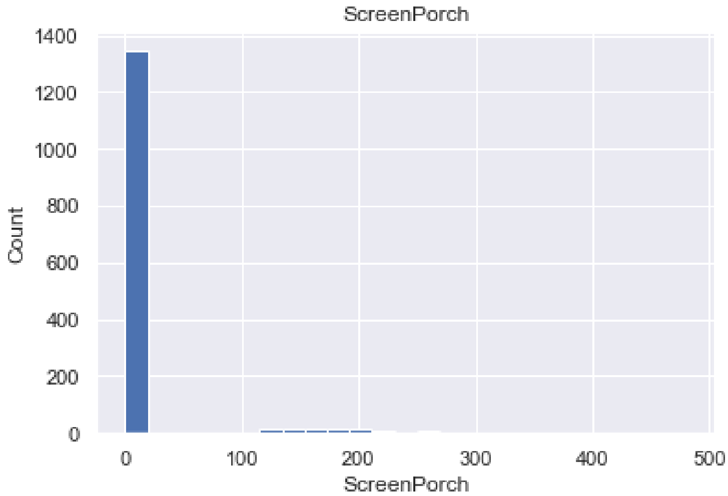
OpenPorchSF

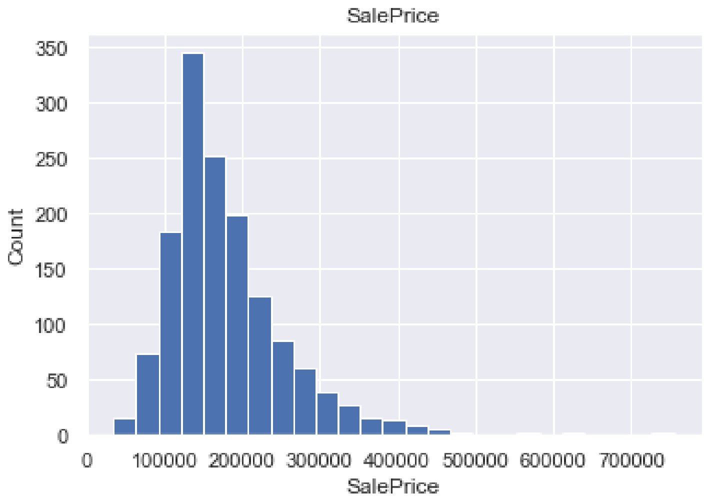


EnclosedPorch





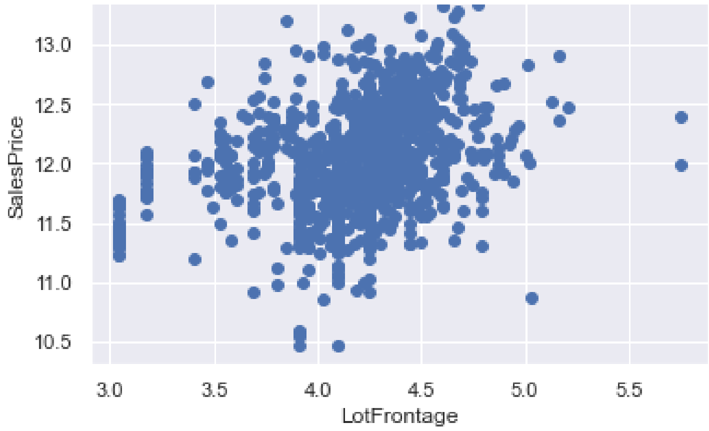




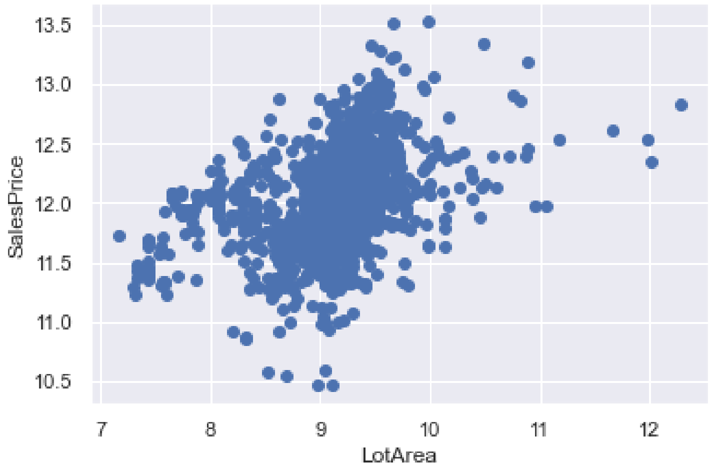
In [ ]:

In [77]:

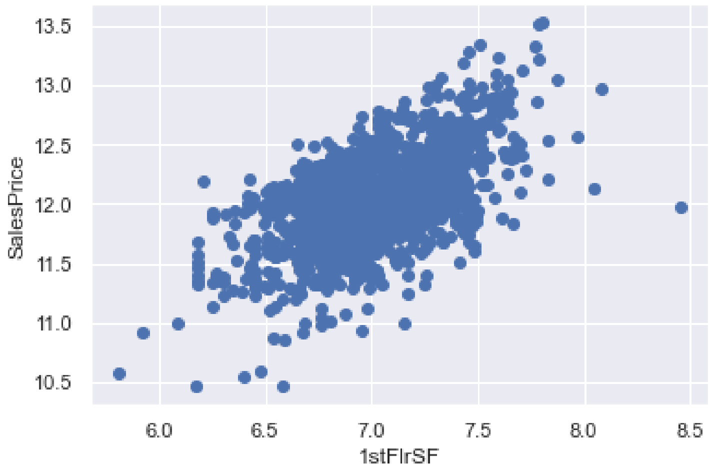
```
## We will be using logarithmic transformation


for feature in continuous_feature:
    data=df.copy()
    if 0 in data[feature].unique():
        pass
    else:
        data[feature]=np.log(data[feature])
        data['SalePrice']=np.log(data['SalePrice'])
        plt.scatter(data[feature],data['SalePrice'])
        plt.xlabel(feature)
        plt.ylabel('SalesPrice')
        plt.title(feature)
        plt.show()
```



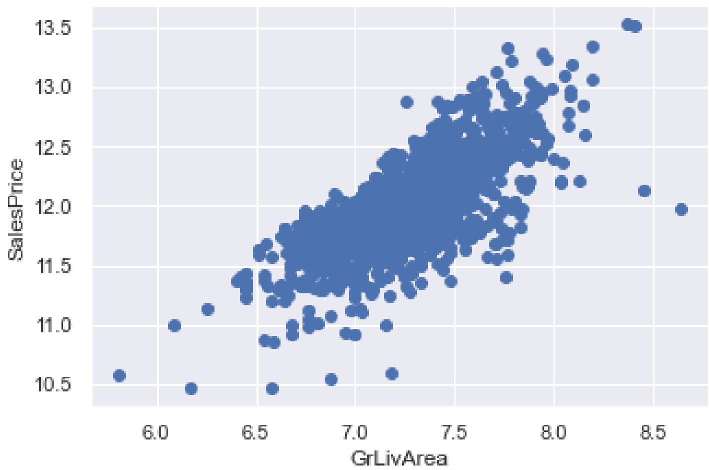
LotArea



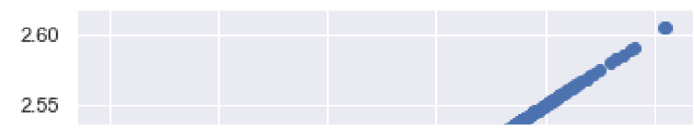
1stFlrSF

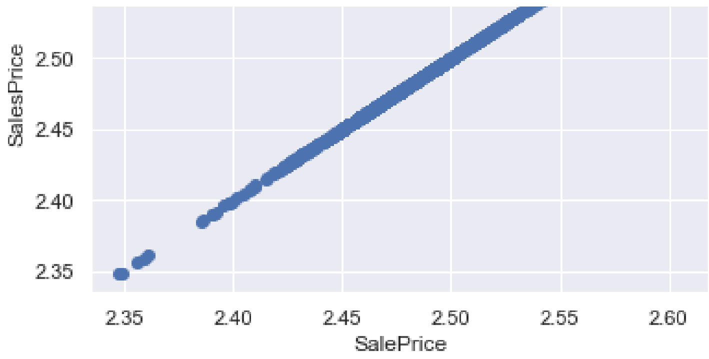


GrLivArea



SalePrice

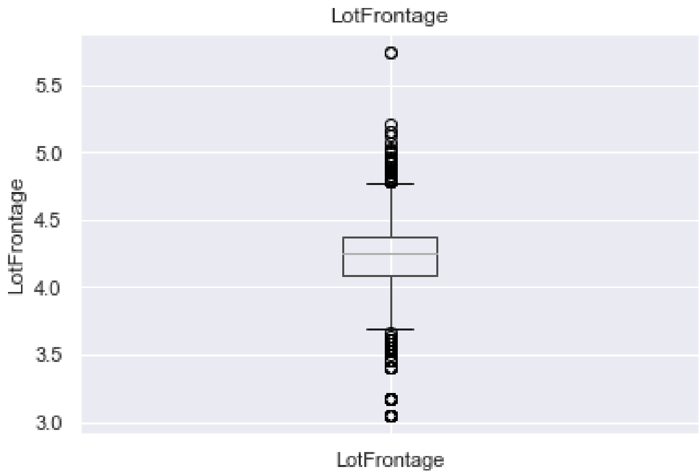


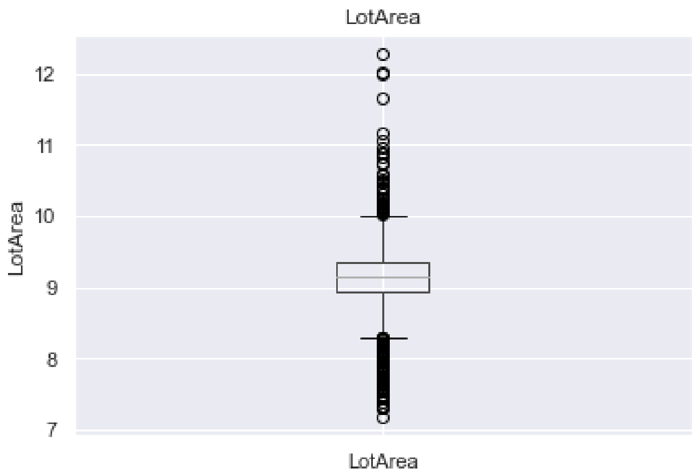


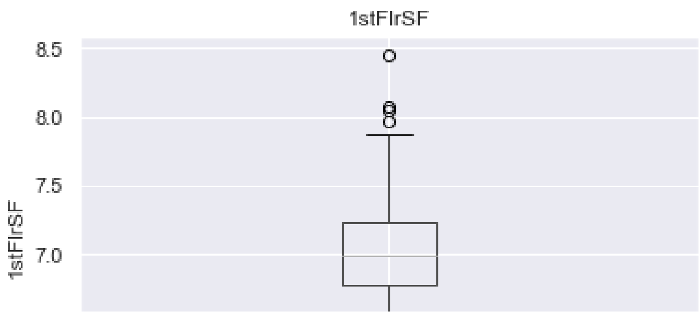
Outliers:

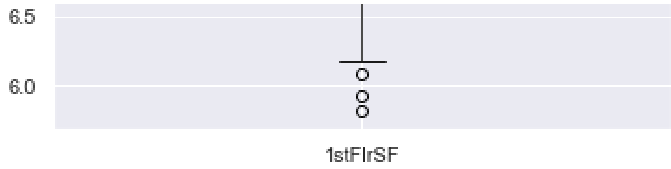
In [78]:

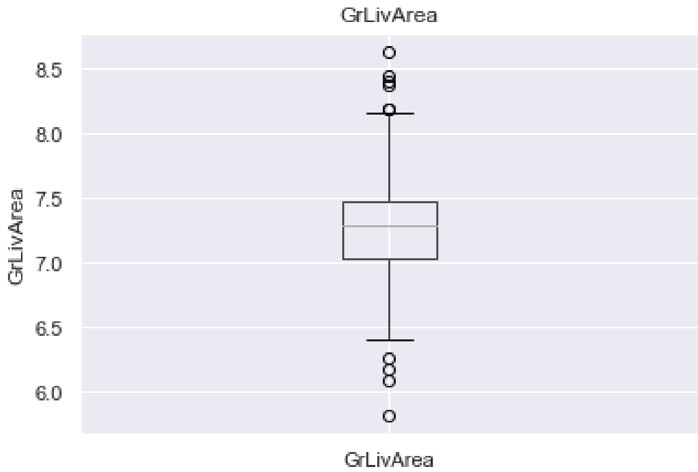
```
for feature in continuous_feature:
    data=df.copy()
    if 0 in data[feature].unique():
        pass
    else:
        data[feature]=np.log(data[feature])
        data.boxplot(column=feature)
        plt.ylabel(feature)
        plt.title(feature)
        plt.show()
```











Categorical Variables

In [79]:

```
categorical_features=[feature for feature in df.columns if data[feature].dtypes=='O']
categorical_features
```

Out[79]:

```
['MSZoning',
 'Street',
 'LotShape',
 'LandContour',
 'Utilities',
 'LotConfig',
 'LandSlope',
 'Neighborhood',
 'Condition1',
 'Condition2',
 'BldgType',
 'HouseStyle',
 'RoofStyle',
 'RoofMatl',
 'Exterior1st',
 'Exterior2nd',
 'MasVnrType',
 'ExterQual',
 'ExterCond',
 'Foundation',
 'BsmtQual',
 'BsmtCond',
 'BsmtExposure'
```

```
 'BsmtExposure',
 'BsmtFinType1',
 'BsmtFinType2',
 'Heating',
 'HeatingQC',
 'CentralAir',
 'Electrical',
 'KitchenQual',
 'Functional',
 'FireplaceQu',
 'GarageType',
 'GarageFinish',
 'GarageQual',
 'GarageCond',
 'PavedDrive',
 'SaleType',
 'SaleCondition']
```

In [80]:

```
df[categorical_features].head()
```

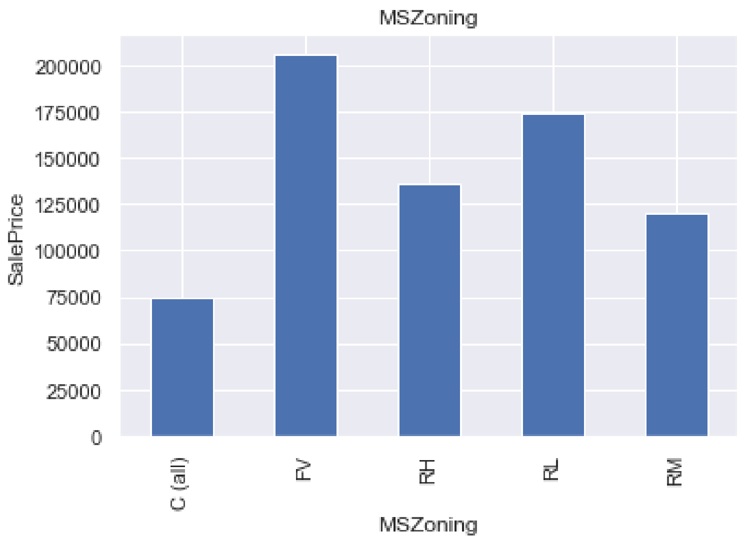
Out[80]:

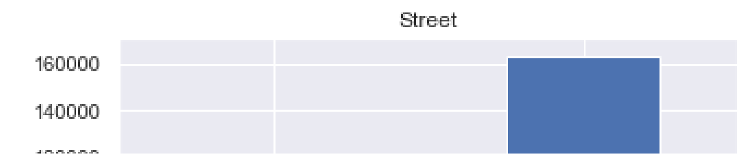
	MSZoning	Street	LotShape	LandContour	Utilities	LotConfig	LandSlope	Neighborhood	Condition1	Condition2	BldgTyp
0	RL	Pave	Reg	Lvl	AllPub	Inside	Gtl	CollgCr	Norm	Norm	1Fa
1	RL	Pave	Reg	Lvl	AllPub	FR2	Gtl	Veenker	Feedr	Norm	1Fa
2	RL	Pave	IR1	Lvl	AllPub	Inside	Gtl	CollgCr	Norm	Norm	1Fa
3	RL	Pave	IR1	Lvl	AllPub	Corner	Gtl	Crawfor	Norm	Norm	1Fa
4	RL	Pave	IR1	Lvl	AllPub	FR2	Gtl	NoRidge	Norm	Norm	1Fa

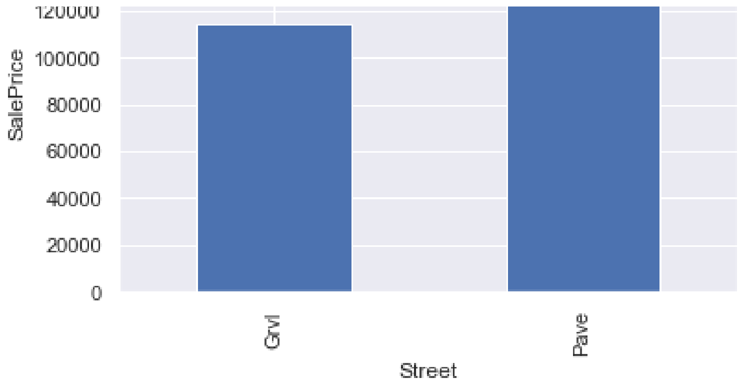
Find out the relationship between categorical variable and dependent feature SalesPrice

In [81]:

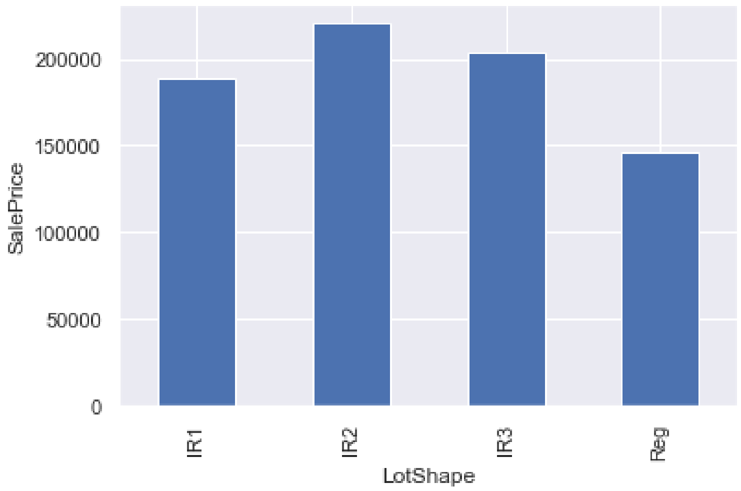
```
for feature in categorical_features:
    data=df.copy()
    data.groupby(feature)['SalePrice'].median().plot.bar()
    plt.xlabel(feature)
    plt.ylabel('SalePrice')
    plt.title(feature)
    plt.show()
```



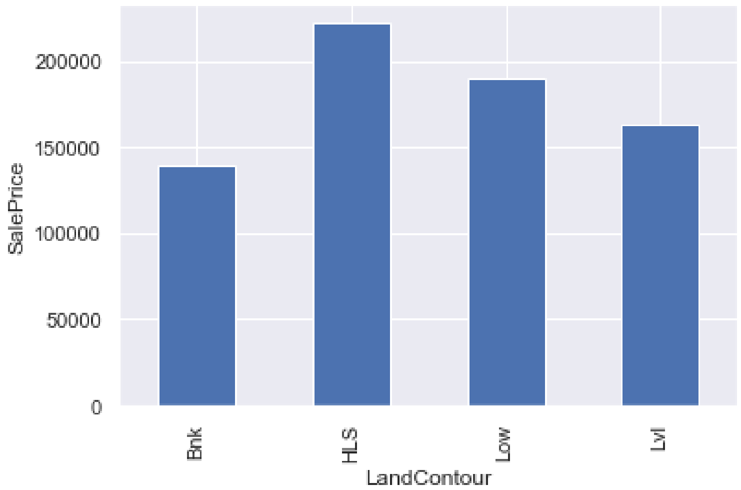




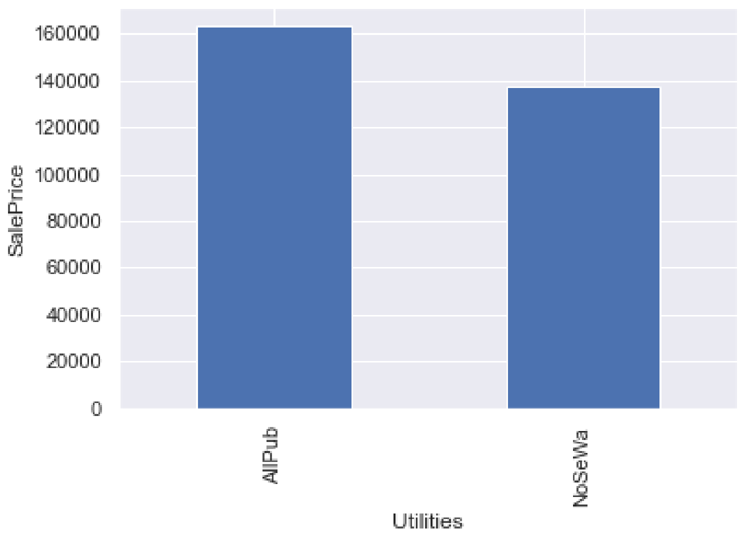
LotShape



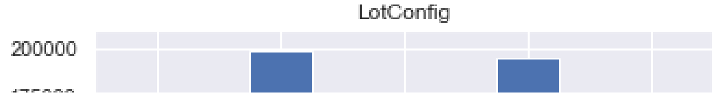
LandContour

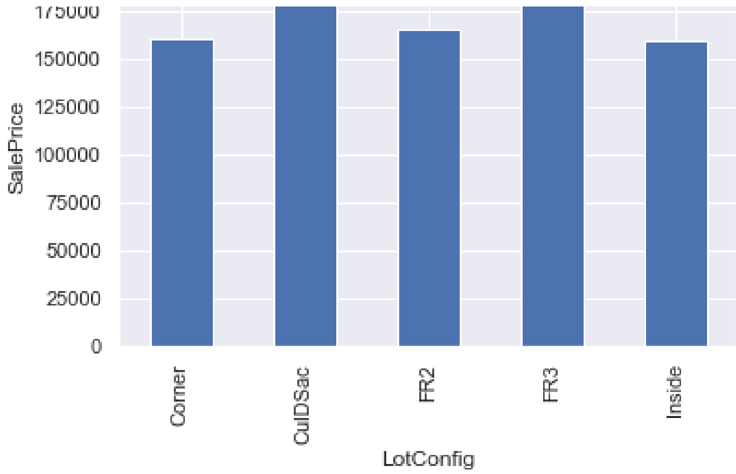


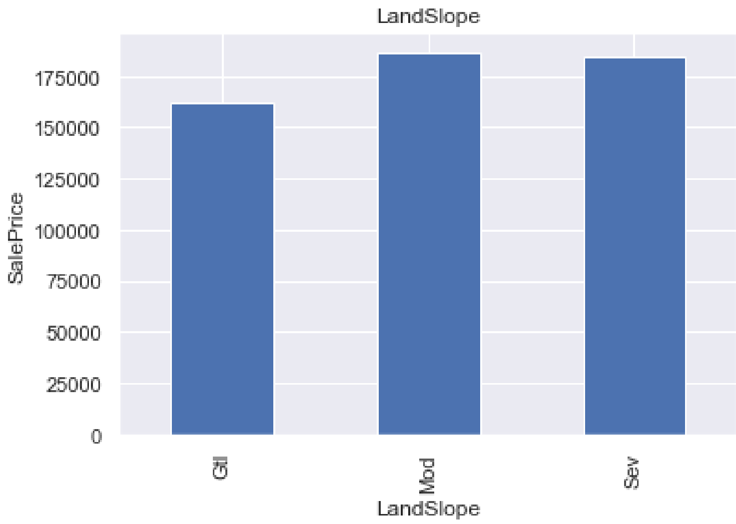
Utilities

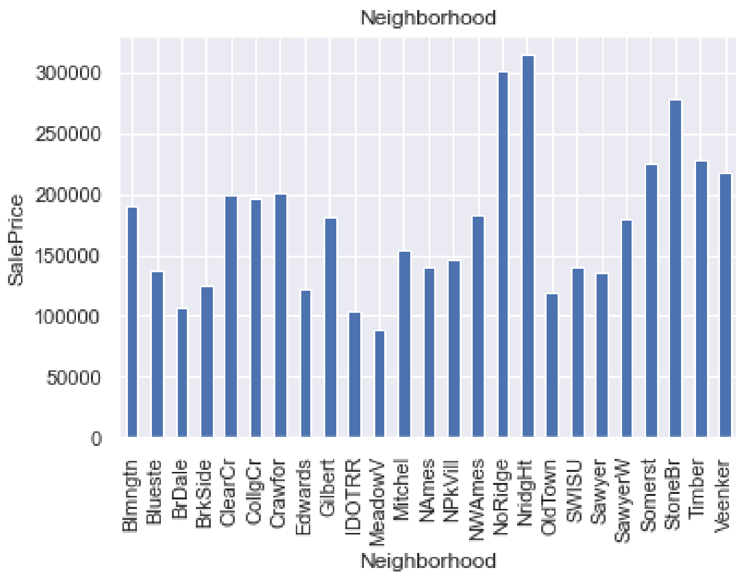


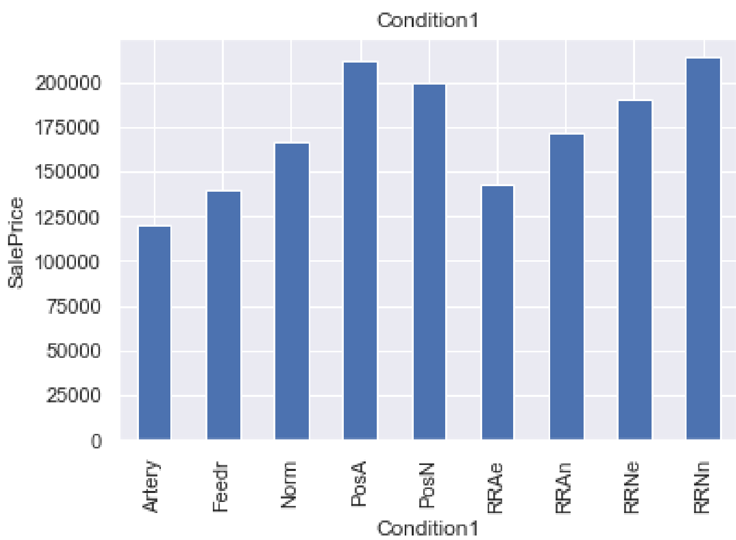
LotConfig

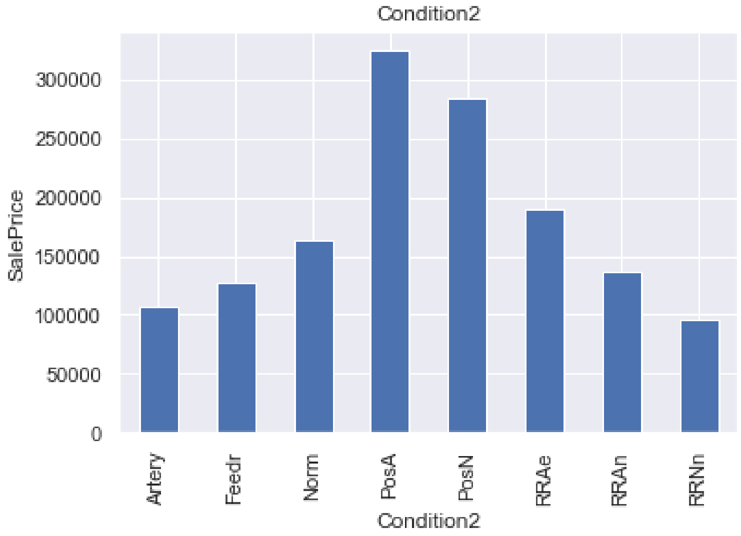


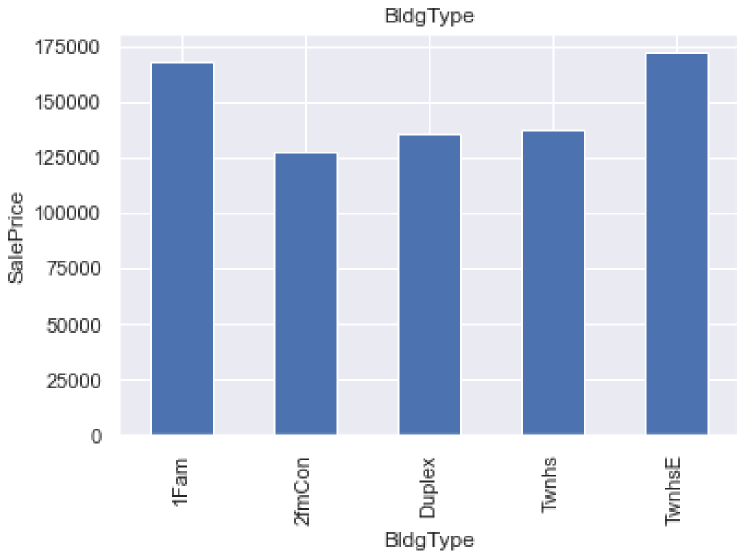

SalePrice
175000
150000
125000
100000
75000
50000
25000
0
Corner
CulDSac
FR2
FR3
Inside
LotConfig

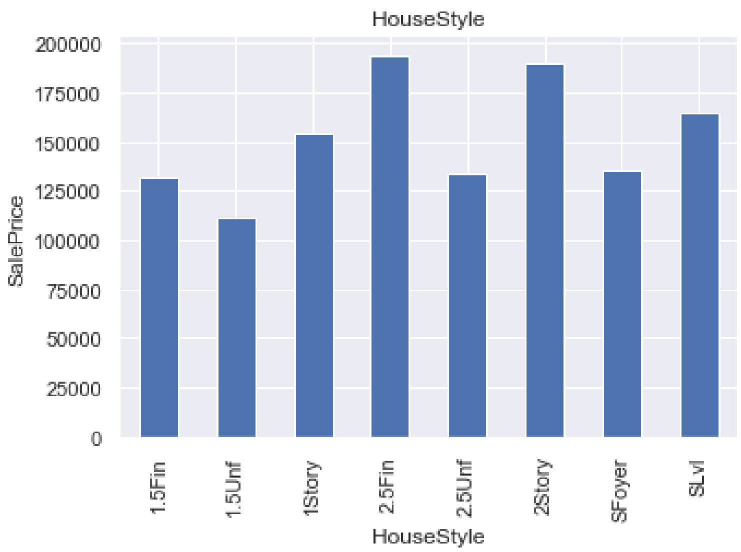

LandSlope
SalePrice
175000
150000
125000
100000
75000
50000
25000
0
Gtl
Mod
Sev
LandSlope

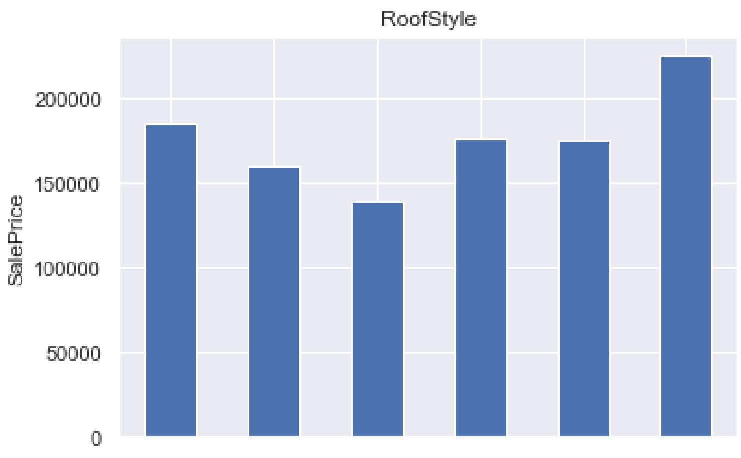

Neighborhood
SalePrice
300000
250000
200000
150000
100000
50000
0
Blmngtn
Blueste
BrDale
BrkSide
ClearCr
CollgCr
Crawfor
Edwards
Gilbert
IDOTRR
MeadowV
Mitchel
NAmes
NPkVill
NWAmes
NoRidge
NridgHt
OldTown
SWISU
Sawyer
SawyerW
Somerst
StoneBr
Timber
Veenker
Neighborhood


Condition1
SalePrice
200000
175000
150000
125000
100000
75000
50000
25000
0
Artery
Feedr
Norm
PosA
PosN
RRAe
RRAn
RRNe
RRNn
Condition1

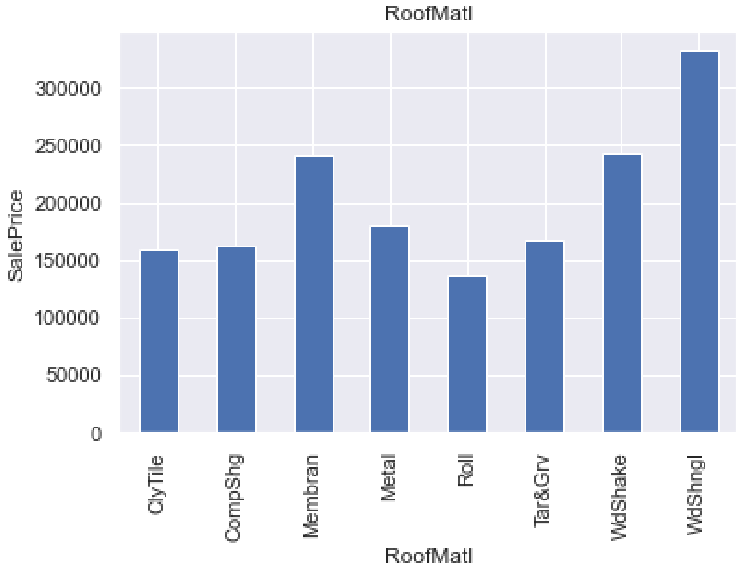

Condition2
SalePrice
300000
250000
200000
150000
100000
50000
0
Artery
Feedr
Norm
PosA
PosN
RRAe
RRAn
RRNn
Condition2

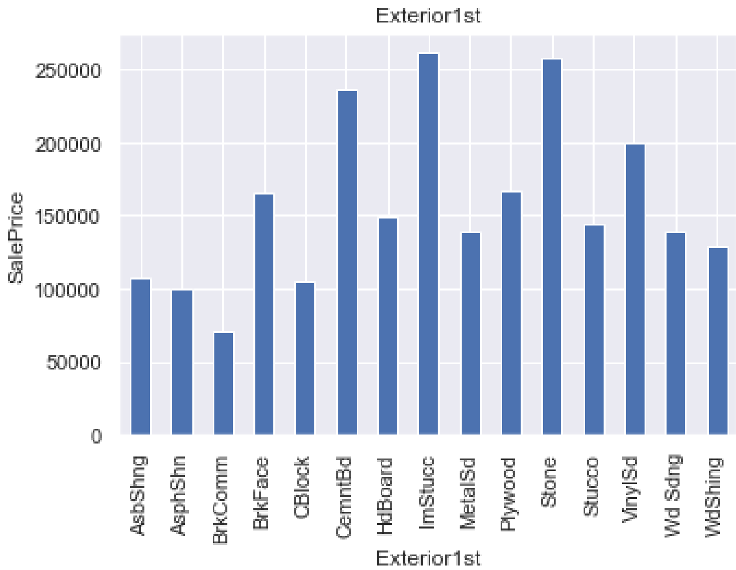

BldgType
SalePrice
175000
150000
125000
100000
75000
50000
25000
0
1Fam
2fmCon
Duplex
Twnhs
TwnhsE
BldgType

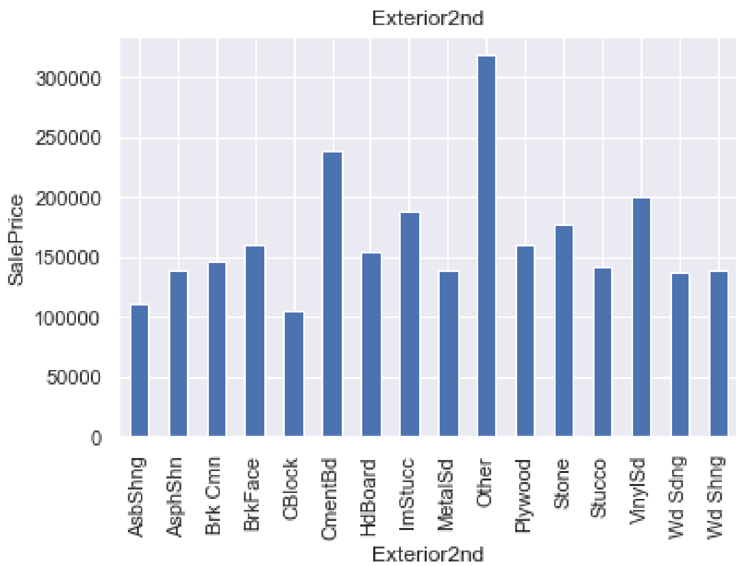

HouseStyle
SalePrice
200000
175000
150000
125000
100000
75000
50000
25000
0
1.5Fin
1.5Unf
1Story
2.5Fin
2.5Unf
2Story
SFoyer
SLvl
HouseStyle

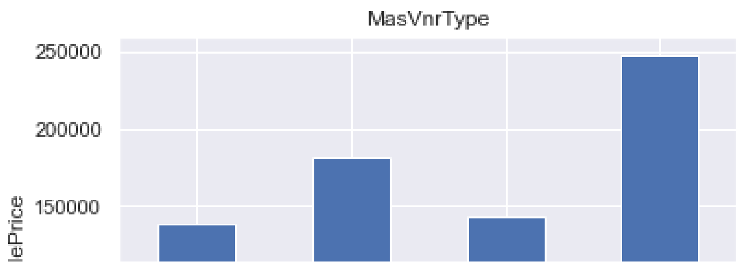

RoofStyle
SalePrice
200000
150000
100000
50000
0

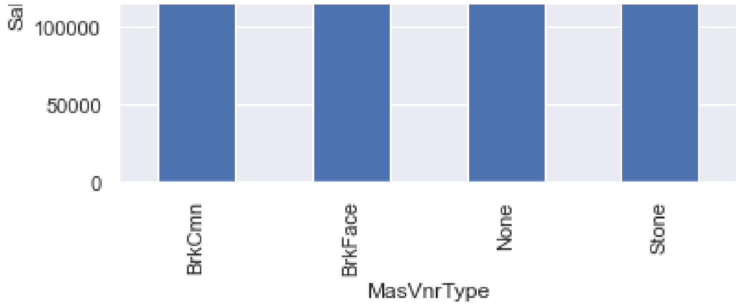

Flat
Gable
Gambrel
Hip
Mansard
Shed
RoofStyle

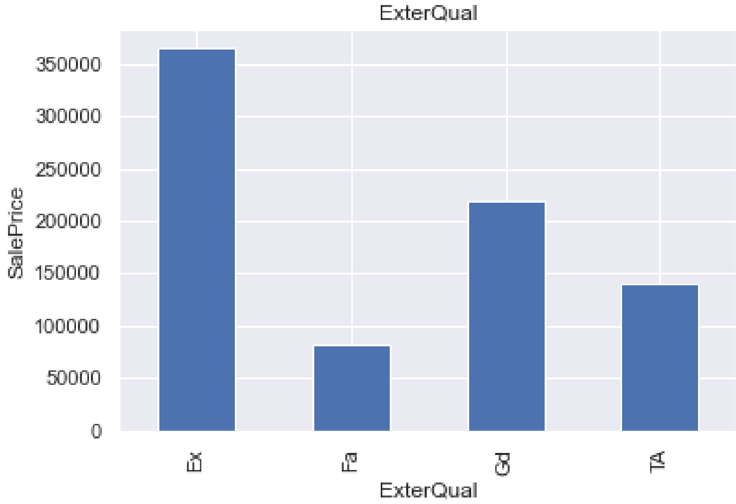

RoofMatl
SalePrice
0
50000
100000
150000
200000
250000
300000
ClyTile
CompShg
Membran
Metal
Roll
Tar&Grv
WdShake
WdShngl
RoofMatl

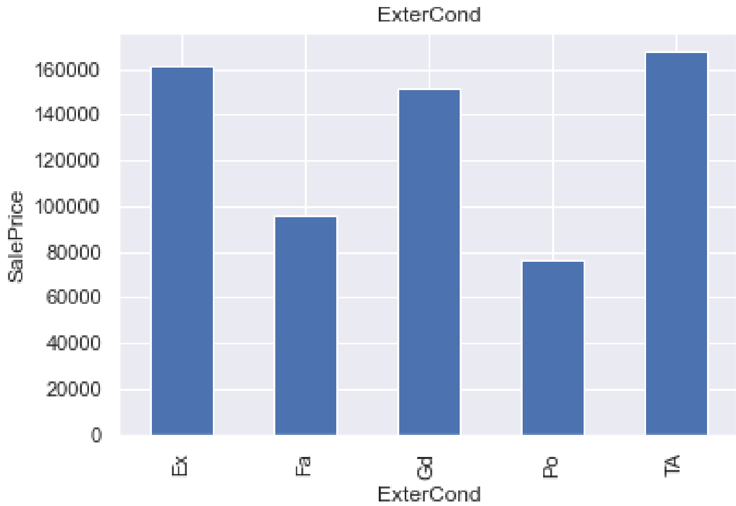

Exterior1st
SalePrice
0
50000
100000
150000
200000
250000
AsbShng
AsphShn
BrkComm
BrkFace
CBlock
CemntBd
HdBoard
ImStucc
MetalSd
Plywood
Stone
Stucco
VinylSd
Wd Sdng
WdShing
Exterior1st

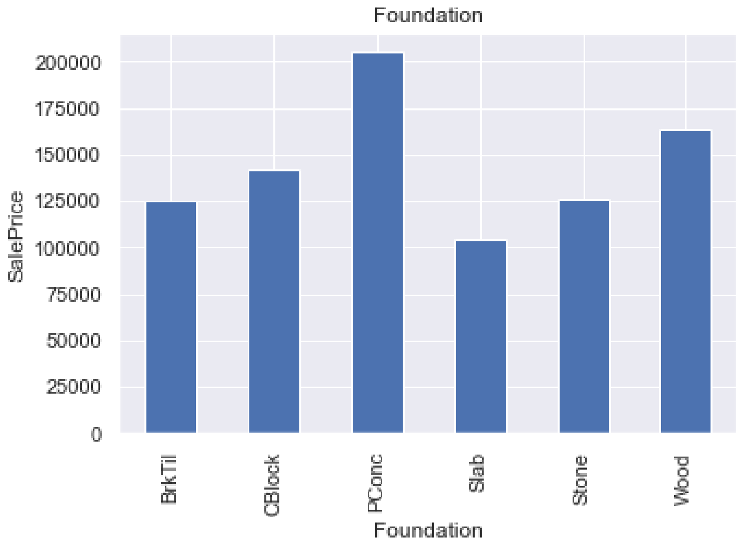

Exterior2nd
SalePrice
0
50000
100000
150000
200000
250000
300000
AsbShng
AsphShn
Brk Cmn
BrkFace
CBlock
CmentBd
HdBoard
ImStucc
MetalSd
Other
Plywood
Stone
Stucco
VinylSd
Wd Sdng
Wd Shng
Exterior2nd

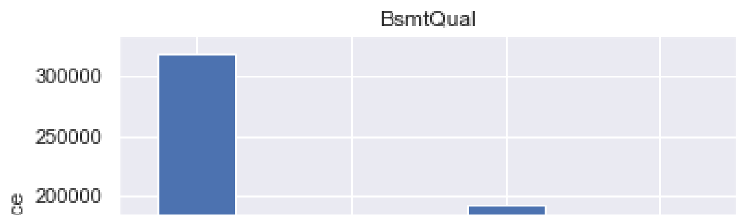

MasVnrType
150000
200000
250000

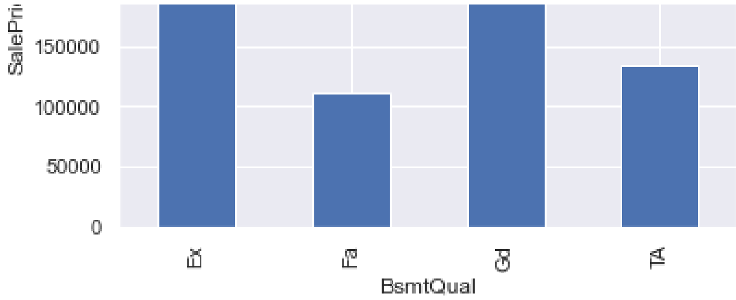

Sal
100000
50000
0
BrkCmn
BrkFace
None
Stone
MasVnrType


ExterQual
SalePrice
350000
300000
250000
200000
150000
100000
50000
0
Ex
Fa
Gd
TA
ExterQual

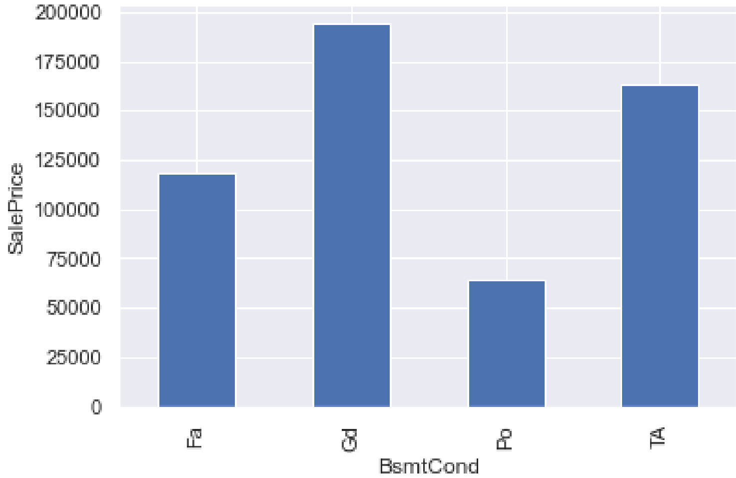

ExterCond
SalePrice
160000
140000
120000
100000
80000
60000
40000
20000
0
Ex
Fa
Gd
Po
TA
ExterCond


Foundation
SalePrice
200000
175000
150000
125000
100000
75000
50000
25000
0
BrkTil
CBlock
PConc
Slab
Stone
Wood
Foundation

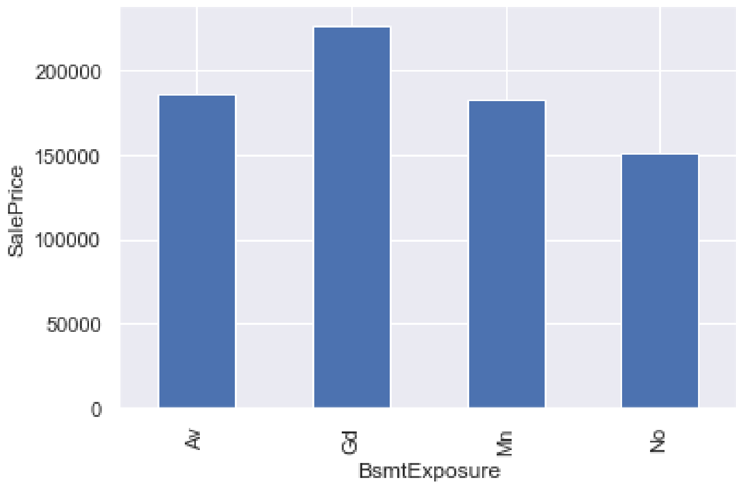

BsmtQual
300000
250000
200000
ce



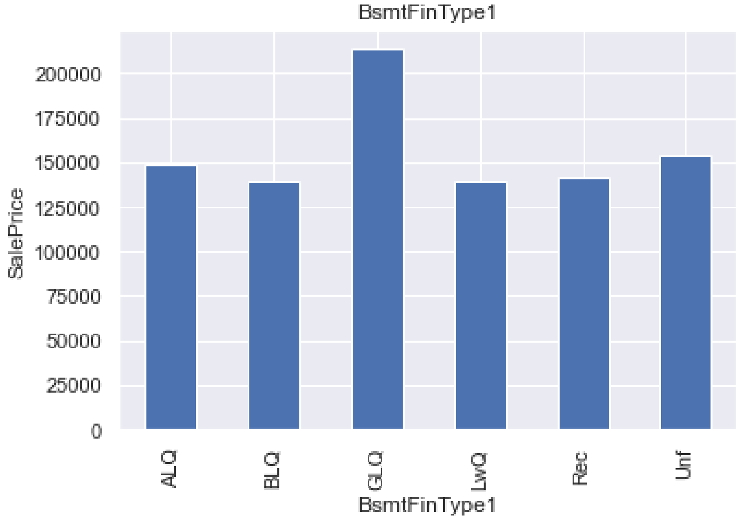
BsmtCond



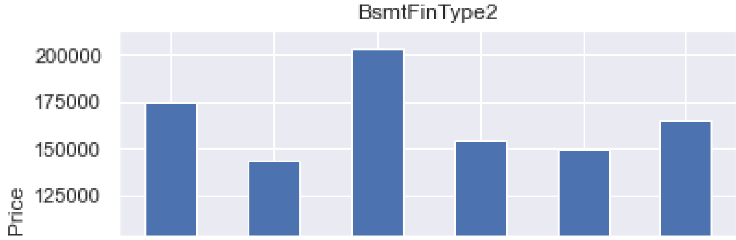
BsmtExposure

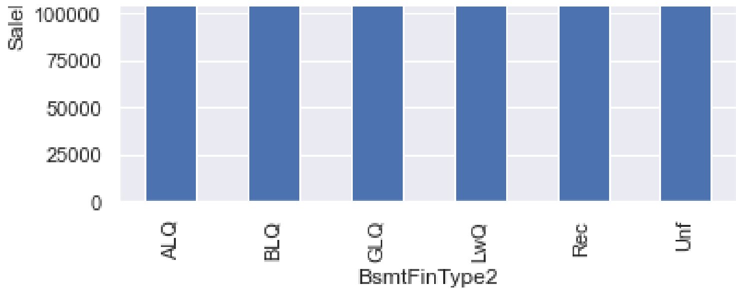


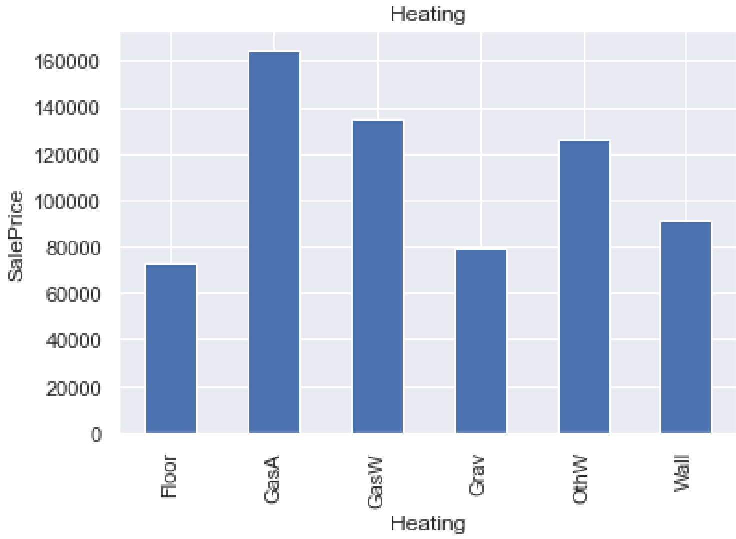
BsmtFinType1

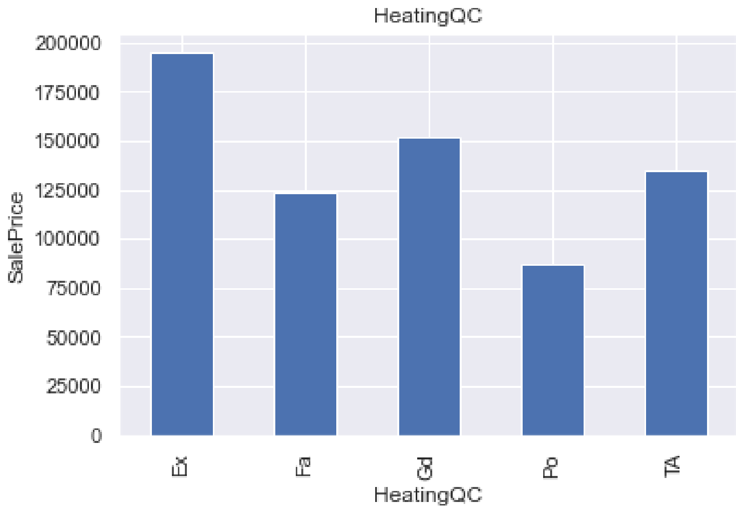


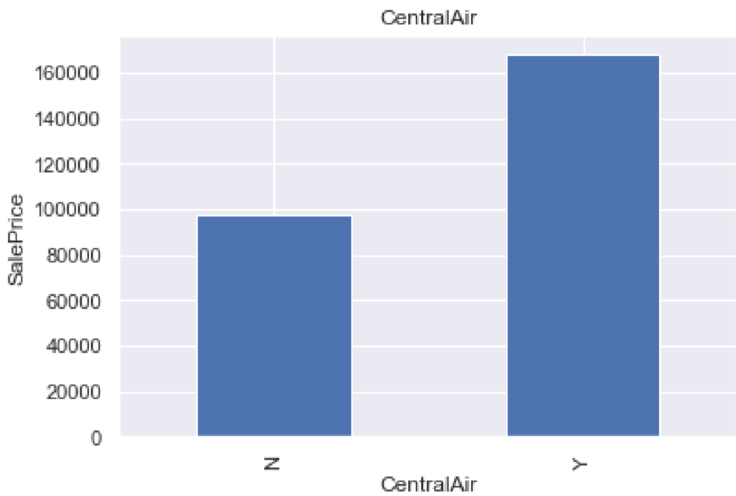
BsmtFinType2

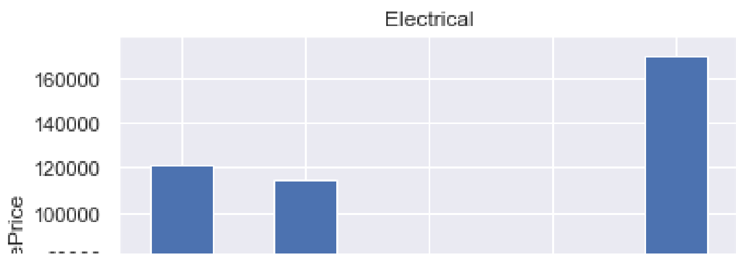


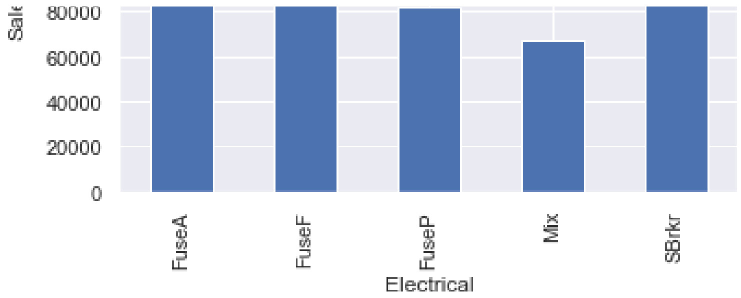

Salel
100000
75000
50000
25000
0
ALQ
BLQ
GLQ
LwQ
Rec
Unf
BsmtFinType2


Heating
SalePrice
160000
140000
120000
100000
80000
60000
40000
20000
0
Floor
GasA
GasW
Grav
OthW
Wall
Heating

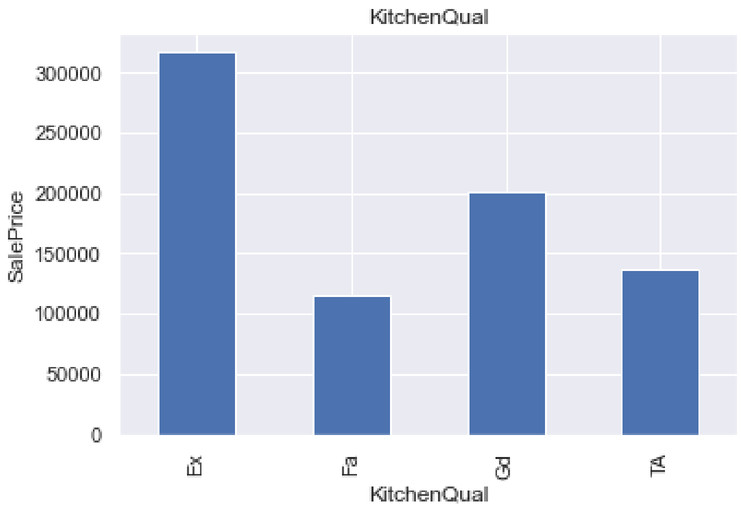

HeatingQC
SalePrice
200000
175000
150000
125000
100000
75000
50000
25000
0
Ex
Fa
Gd
Po
TA
HeatingQC


CentralAir
SalePrice
160000
140000
120000
100000
80000
60000
40000
20000
0
N
Y
CentralAir

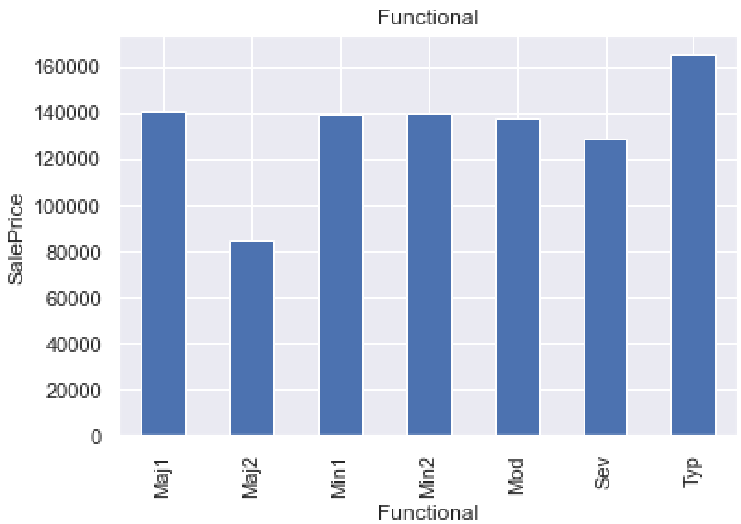

Electrical
160000
140000
120000
100000



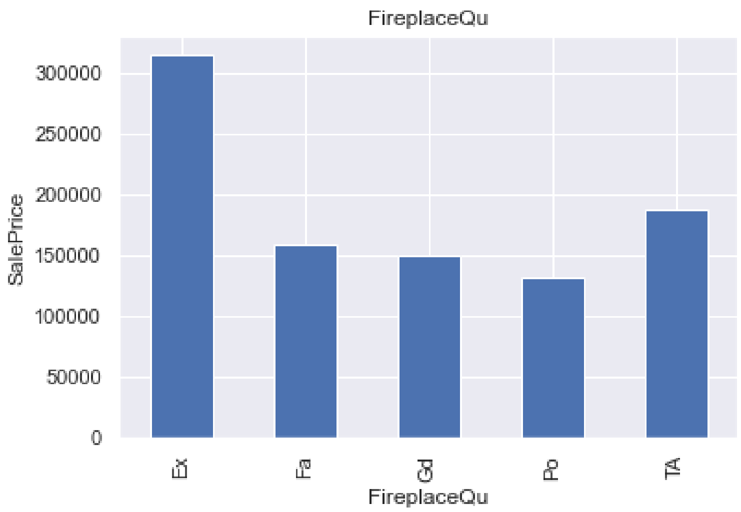
Sal
80000
60000
40000
20000
0
FuseA
FuseF
FuseP
Mix
SBrkr
Electrical



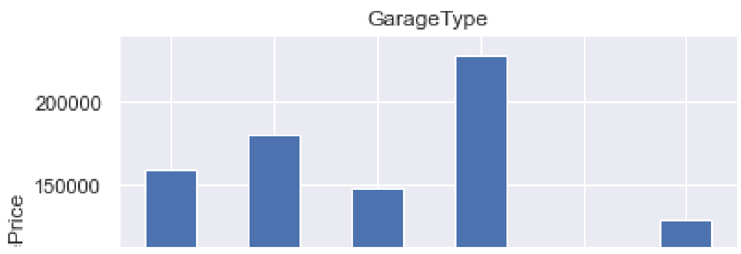
KitchenQual
SalePrice
300000
250000
200000
150000
100000
50000
0
Ex
Fa
Gd
TA
KitchenQual



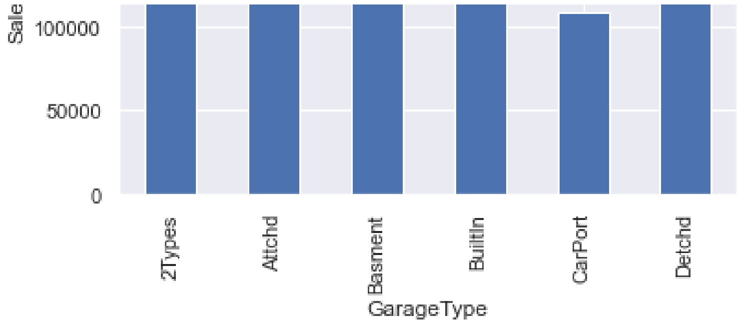
Functional
SalePrice
160000
140000
120000
100000
80000
60000
40000
20000
0
Maj1
Maj2
Min1
Min2
Mod
Sev
Typ
Functional

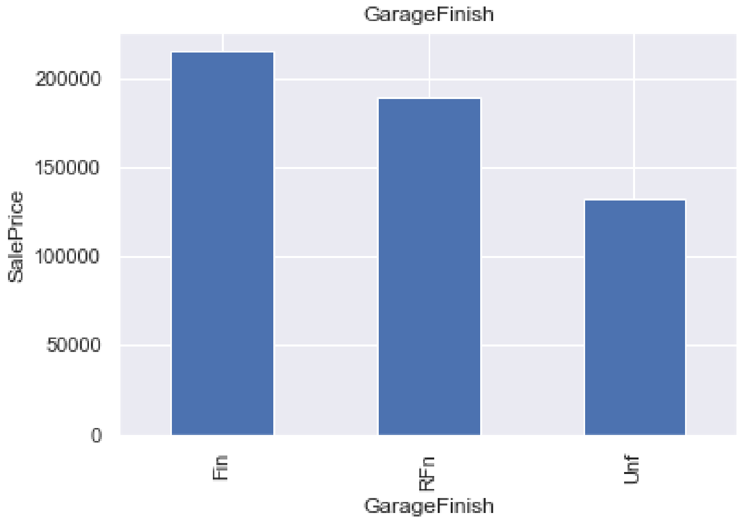


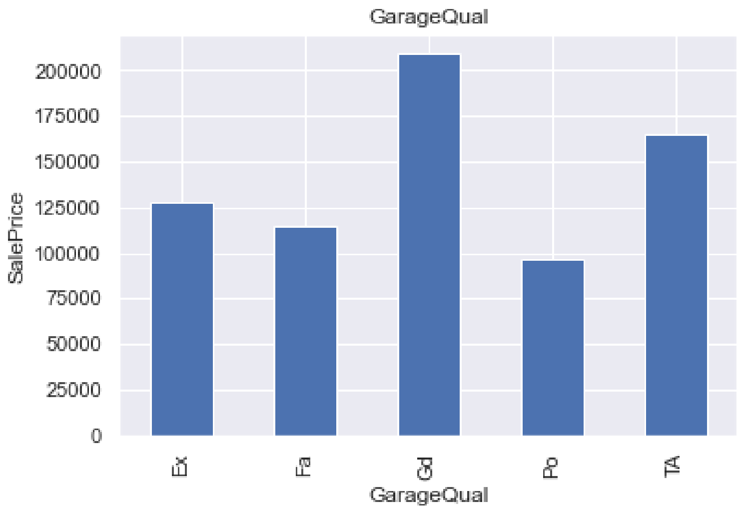
FireplaceQu
SalePrice
300000
250000
200000
150000
100000
50000
0
Ex
Fa
Gd
Po
TA
FireplaceQu

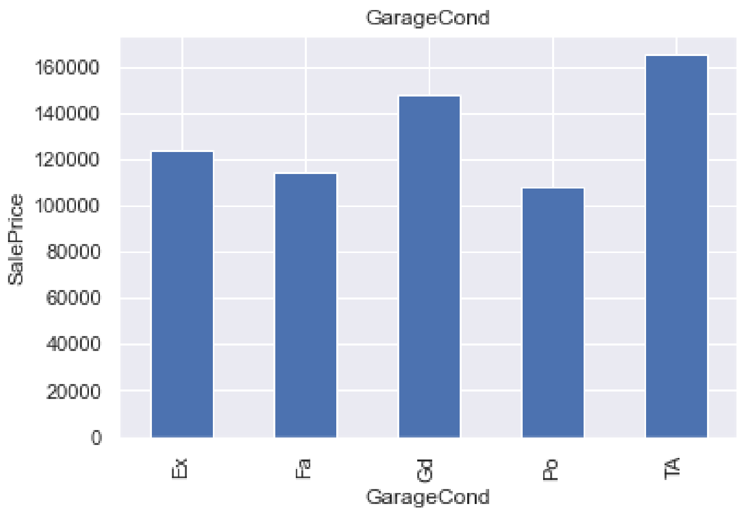


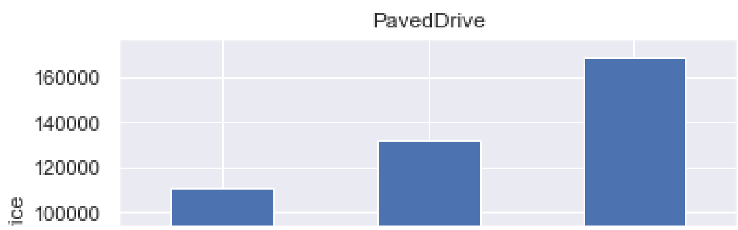
GarageType
200000
150000
Price

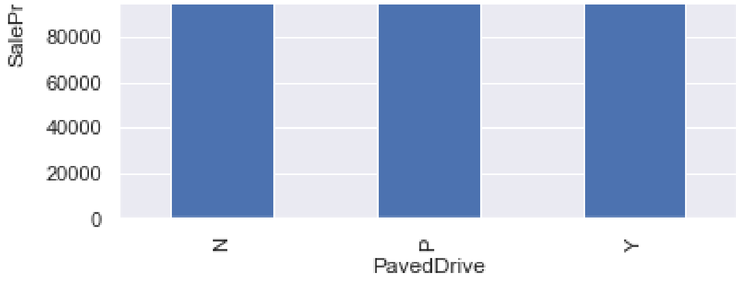

Sale
100000
50000
0
2Types
Attchd
Basment
BuiltIn
CarPort
Detchd
GarageType

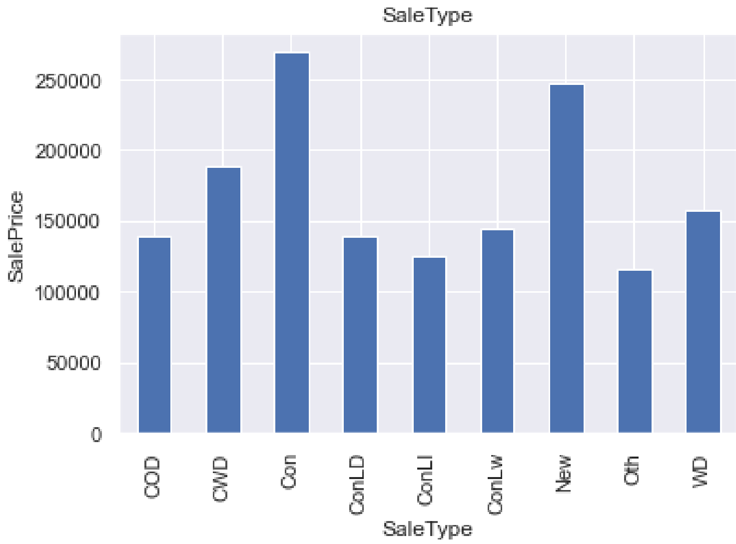

GarageFinish
SalePrice
200000
150000
100000
50000
0
Fin
RFn
Unf
GarageFinish

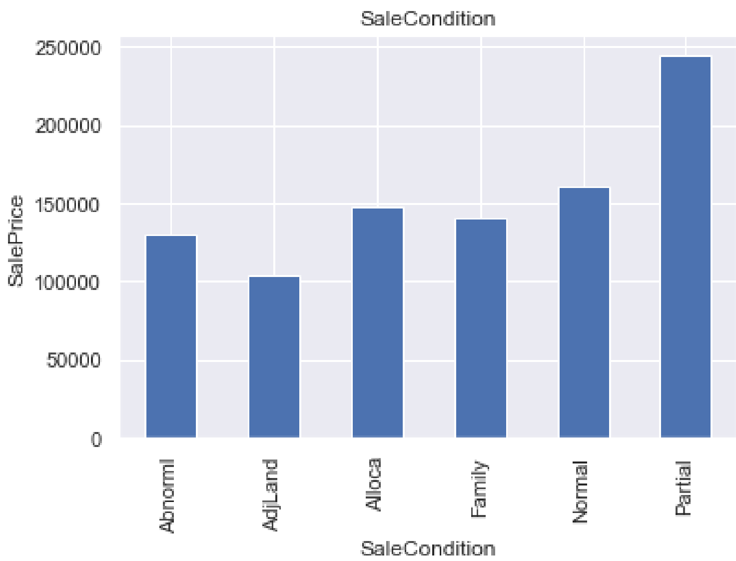

GarageQual
SalePrice
200000
175000
150000
125000
100000
75000
50000
25000
0
Ex
Fa
Gd
Po
TA
GarageQual


GarageCond
SalePrice
160000
140000
120000
100000
80000
60000
40000
20000
0
Ex
Fa
Gd
Po
TA
GarageCond


PavedDrive
ice
160000
140000
120000
100000







In [ ]:

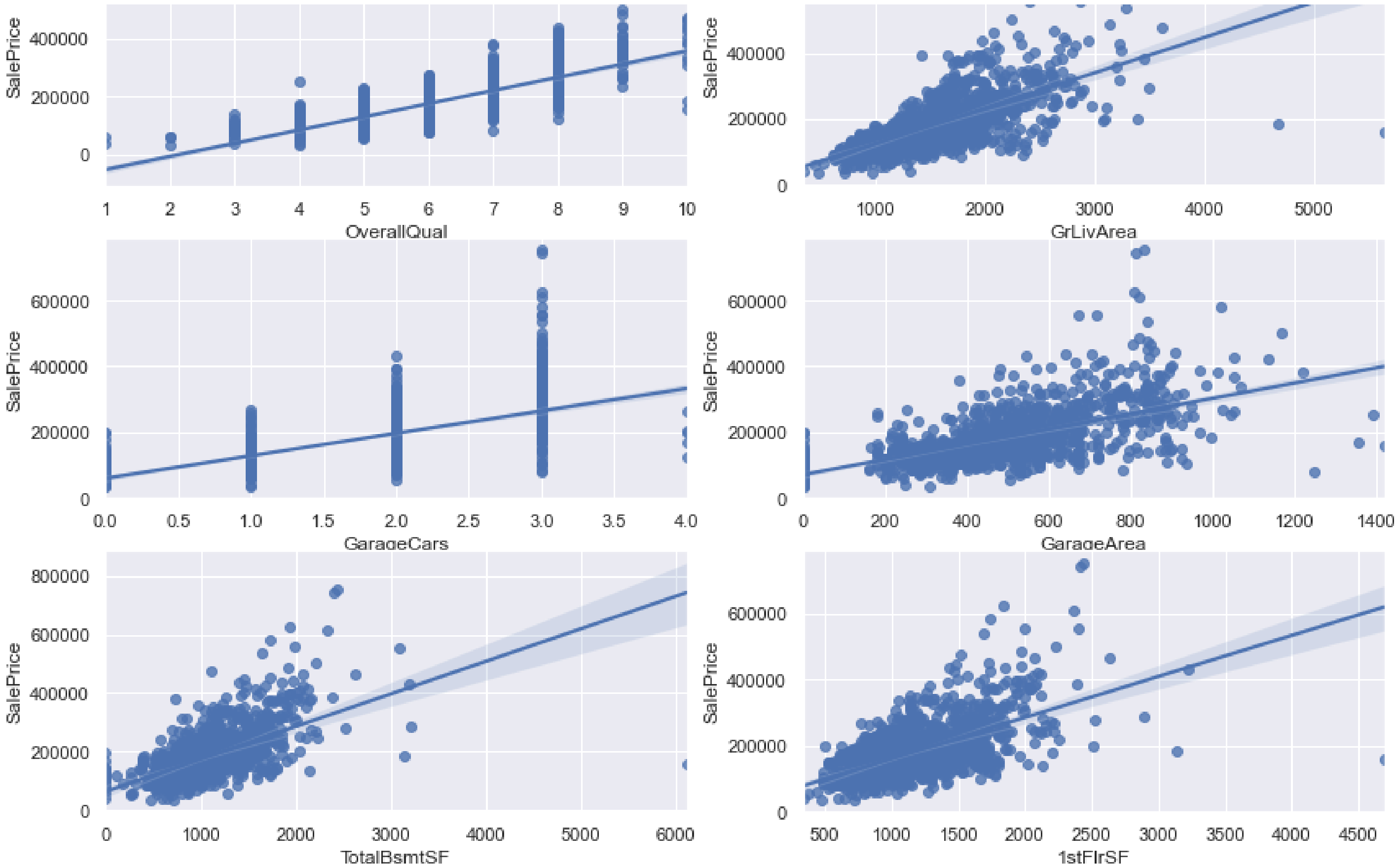
Scatter plot between variables which are most correlated with the target variable 'SalePrice'

In [82]:

```
fig,((ax1,ax2),(ax3,ax4),(ax5,ax6))=plt.subplots(nrows=3,ncols=2,figsize=(14,10))
sns.regplot(x='OverallQual',y='SalePrice', data=df, scatter=True,ax=ax1)
sns.regplot(x='GrLivArea',y='SalePrice', data=df, scatter=True,ax=ax2)
sns.regplot(x='GarageCars',y='SalePrice', data=df, scatter=True,ax=ax3)
sns.regplot(x='GarageArea',y='SalePrice', data=df, scatter=True,ax=ax4)
sns.regplot(x='TotalBsmtSF',y='SalePrice', data=df, scatter=True,ax=ax5)
sns.regplot(x='1stFlrSF',y='SalePrice', data=df, scatter=True,ax=ax6)
```

Out[82]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x23d449f5be0>
```



In [ ]:

```

```

In [ ]:

```

```

Temporal Variables(Eg: Datetime Variables)

In [83]:

```
# list of variables that contain year information
year_feature = [feature for feature in numerical_features if 'Yr' in feature or 'Year' i
n feature]

year_feature
```

Out[83]:

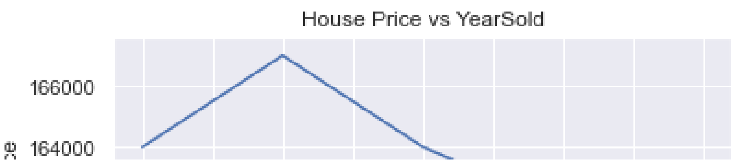
```
['YearBuilt', 'YearRemodAdd', 'YrSold']
```

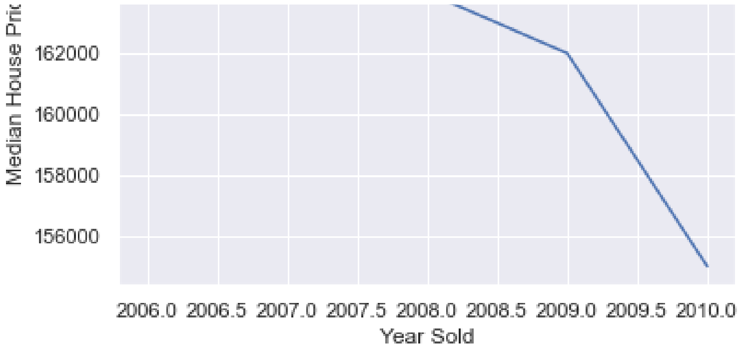
In [84]:

```
## We will check whether there is a relation between year the house is sold and the sales
price

df.groupby('YrSold')['SalePrice'].median().plot()
plt.xlabel('Year Sold')
plt.ylabel('Median House Price')
plt.title("House Price vs YearSold")
```

Out[84]:

```
Text(0.5, 1.0, 'House Price vs YearSold')
```





This shows the saleprice decreases with respet to time.

In [ ]:

```

```

In [ ]:

```

```

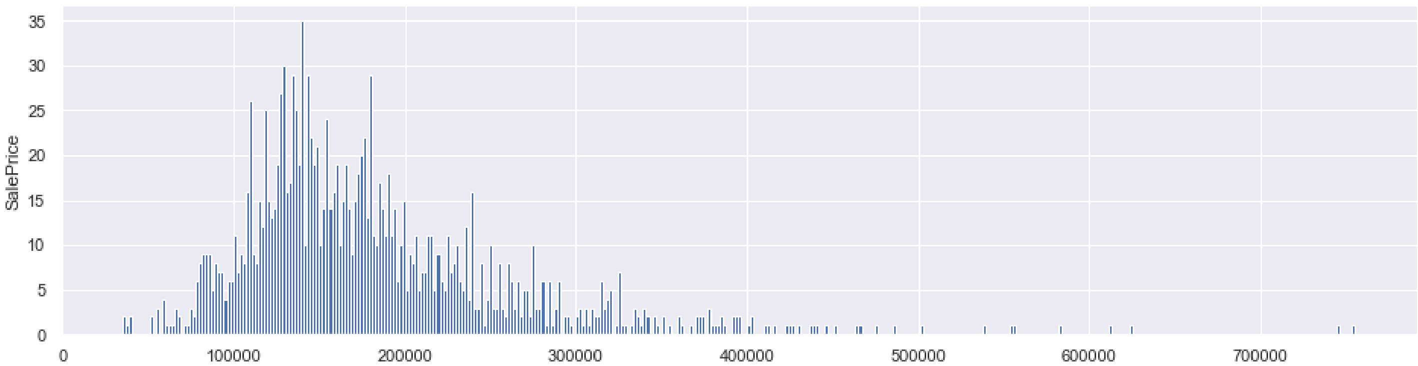
Plotting the 'SalePrice' in a histogram to check any outliers are present or not.

In [85]:

```
fig_size = plt.rcParams["figure.figsize"]
fig_size[0] =16.0
fig_size[1] = 4.0

x =df['SalePrice']
plt.hist(x,bins=400)
plt.ylabel('SalePrice')
```

Out[85]:

```
Text(0, 0.5, 'SalePrice')
```



Plotting the 'SalePrice' in a Boxplot to check any outliers are present or not

In [86]:

```
sns.boxplot(x=df['SalePrice'])
```

Out[86]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x23d431784f0>
```

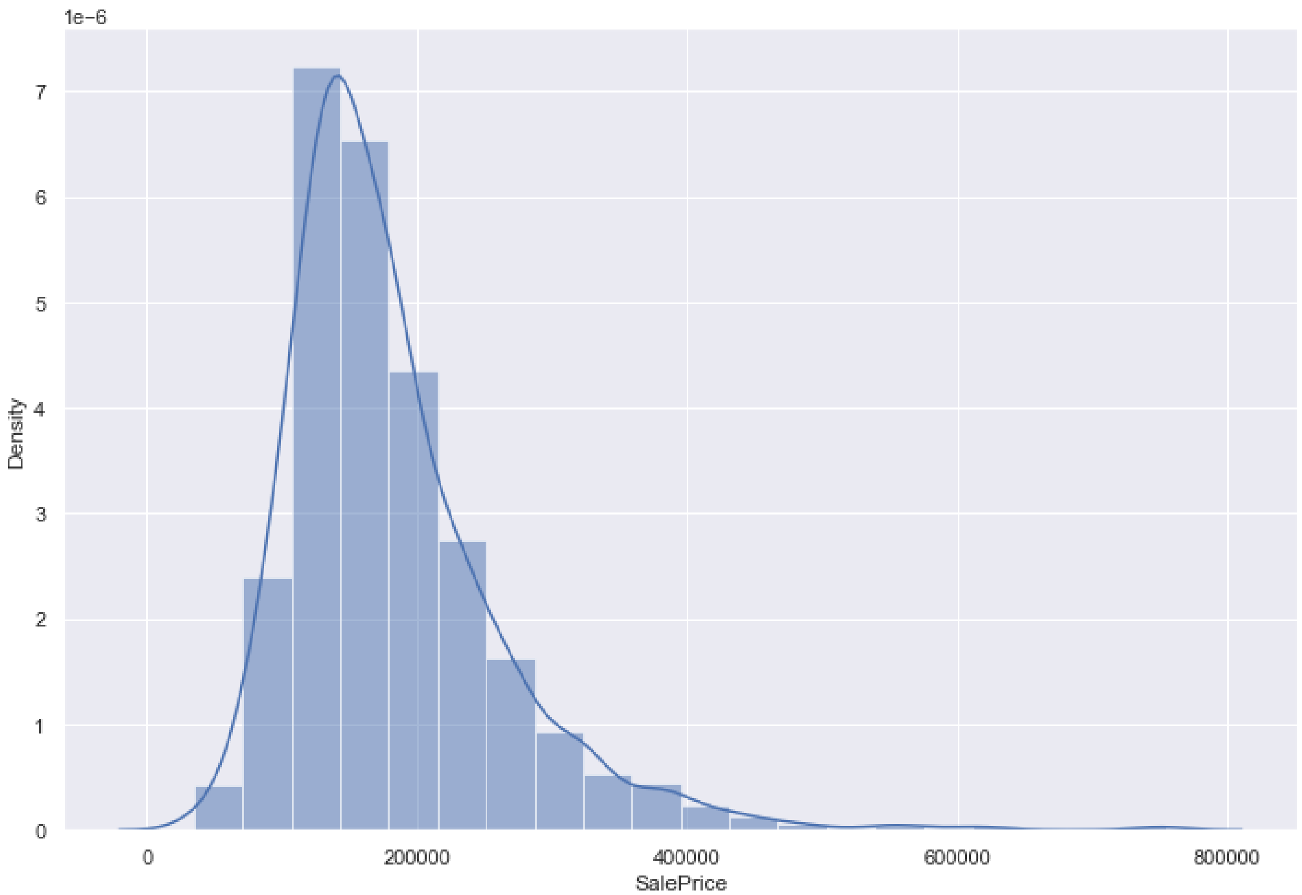
The above histogram and boxplot shows many outlies ar present.

Plotting histogram using seaborn

In [87]:

```
sns.set(rc={'figure.figsize':(12,8)})
sns.distplot(df['SalePrice'], color='b', bins=20, hist_kws={'alpha': 0.5});
```

```
C:\Users\Admin\anaconda3\lib\site-packages\seaborn\distributions.py:2551: FutureWarning: `distplot` is a deprecated function and will be removed in a future version. Please adapt your code to use either `displot` (a figure-level function with similar flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```



The above histogram shows the distribution is right skewed.

The box plot below will show the outliers in more clear.

In [88]:

```
print(df['SalePrice'].describe())
```

```
count      1460.000000
mean     180921.195890
std       79442.502883
min       34900.000000
25%      129975.000000
50%      163000.000000
75%      214000.000000
max      755000.000000
```

```
max       755000.000000
Name: SalePrice, dtype: float64
```

In [ ]: