Title: Exploratory Data Analysis on Housing Prices

Abstract

DESCRIPTIVE ABSTRACT: The dataset contains information from the Assessor's Office used in computing assessed values for individual residential properties sold in a US town, from 2006 to 2010. There are two datasets used, housing1.txt and housing2.txt, with additional information of the details of the variables in a data dictionaary. These files will be used for data ingestion, cleaning, wranglings, exploration, vizualization and report of detailed findings with summary and conclusion. Different data exploration is conducted to answer the focus question for the project. There were handful of findings and recommendations, with the challenges and limitations encountered reported

Background

The dataset contained in 'housing1 and housing2 are utilized in the computation and in assessing values to a residential building property for the sake of taxation. Property owners' obligations to pay property taxes may be impacted by these valuations.

Analysis Focus;

- The relationship between the year houses were built and the Sale price they well sold?
- Does building houses on larger Lot Area of land resulted into high selling prices?
- What is the preference of buyers in the amount ready to pay for a house, Ground Living Area or Lot Area?
- Does inflation has impact on the numbers of houses built across all the years?
- Does inflation has impact on the numbers of houses sold between year 2006 and year 2010 the years?
- The relationship between Sale Price and the Overall Quallity of each house?
- What is the correlation of the Sale Price with other variables using heatmap?

Data Ingestion

In [ ]:
```
# import necessary libraries files
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
import random
import seaborn as sns
```

Reading the files

There are two datasets files; housing1 and housing2. These datasets are in text format.

In [ ]:
```
# Reading the file in housing1 and saving it as 'housing1' and showing the class type the dataframe belongs to.
housing1 = pd.read_csv('Housing_1.txt', sep='\t')
print(type(housing1))
```

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

In [ ]:
```
# Reading the file in housing2 and saving it as 'housing2', showing the class type the dataframe belongs to.
housing2 = pd.read_csv('Housing_2.txt', sep='\t')
print(type(housing2))
```

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

Data Overview

Let run the first dataset, to study the characters, parameters, the shape, and the statistics of the dataset.

In [ ]:
```
# First Dataset "Housing_1"
housing1.head()
```

Out[ ]:

	Order	PID	MS SubClass	MS Zoning	Lot Frontage	Lot Area	Street	Alley	Lot Shape	Land Contour	...	Screen Porch	Pool Area	Pool QC	Fence	Misc Feature	Misc Val
0	1	526301100	20	RL	141.0	31770	Pave	NaN	IR1	Lvl	...	0	0	NaN	NaN	NaN	0
1	2	526350040	20	RH	80.0	11622	Pave	NaN	Reg	Lvl	...	120	0	NaN	MnPrv	NaN	0
2	3	526351010	20	RL	81.0	14267	Pave	NaN	IR1	Lvl	...	0	0	NaN	NaN	Gar2	12500
3	4	526353030	20	RL	93.0	11160	Pave	NaN	Reg	Lvl	...	0	0	NaN	NaN	NaN	0
4	5	527105010	60	RL	74.0	13830	Pave	NaN	IR1	Lvl	...	0	0	NaN	MnPrv	NaN	0

5 rows × 81 columns

In [ ]:
```
# Checking the index for the number of row.
housing1.index
```

Out[ ]:
```
RangeIndex(start=0, stop=2933, step=1)
```

In [ ]:
```
# Check the number rows and columns on the dataset.
housing1.shape
```

Out[ ]:
```
(2933, 81)
```

The number of rows in the dataset is 2,933 and the numbers of columns is 81.

Therefore, let set our maximum number of rows on display to 100. This will enable us see more into our data.

In [ ]:
```
# To have a display of about 100 in our query of data. we set our options as follows
pd.set_option('display.max_rows', 100)
```

In [ ]:
```
# Check the names of each columns in the dataset for proper identification.
housing1.columns
```

Out[ ]:
```
Index(['Order', 'PID', 'MS SubClass', 'MS Zoning', 'Lot Frontage', 'Lot Area',
       'Street', 'Alley', 'Lot Shape', 'Land Contour', 'Utilities',
       'Lot Config', 'Land Slope', 'Neighborhood', 'Condition 1',
       'Condition 2', 'Bldg Type', 'House Style', 'Overall Qual',
       'Overall Cond', 'Year Built', 'Year Remod/Add', 'Roof Style',
       'Roof Matl', 'Exterior 1st', 'Exterior 2nd', 'Mas Vnr Type',
       'Mas Vnr Area', 'Exter Qual', 'Exter Cond', 'Foundation', 'Bsmt Qual',
       'Bsmt Cond', 'Bsmt Exposure', 'BsmtFin Type 1', 'BsmtFin SF 1',
       'BsmtFin Type 2', 'BsmtFin SF 2', 'Bsmt Unf SF', 'Total Bsmt SF',
       'Heating', 'Heating QC', 'Central Air', 'Electrical', '1st Flr SF',
       '2nd Flr SF', 'Low Qual Fin SF', 'Gr Liv Area', 'Bsmt Full Bath',
       'Bsmt Half Bath', 'Full Bath', 'Half Bath', 'Bedroom AbvGr',
       'Kitchen AbvGr', 'Kitchen Qual', 'TotRms AbvGrd', 'Functional',
       'Fireplaces', 'Fireplace Qu', 'Garage Type', 'Garage Yr Blt',
       'Garage Finish', 'Garage Cars', 'Garage Area', 'Garage Qual',
       'Garage Cond', 'Paved Drive', 'Wood Deck SF', 'Open Porch SF',
       'Enclosed Porch', '3Ssn Porch', 'Screen Porch', 'Pool Area', 'Pool QC',
       'Fence', 'Misc Feature', 'Misc Val', 'Mo Sold', 'Yr Sold', 'Sale Type',
       'Sale Condition'],
      dtype='object')
```

In [ ]:
```
housing1.describe()
```

Out[ ]:

	Order	PID	MS SubClass	Lot Frontage	Lot Area	Overall Qual	Overall Cond	Year Built	Year Remod/Add	Mas V A
count	2933.000000	2.933000e+03	2933.000000	2443.000000	2933.000000	2933.000000	2933.000000	2933.000000	2933.000000	2910.000
mean	1464.151381	7.142811e+08	57.349131	69.237822	10147.789976	6.094102	5.562564	1971.344016	1984.241050	102.082
std	846.579702	1.887213e+08	42.632965	23.354888	7876.008408	1.410595	1.111114	30.232331	20.864849	179.172
min	1.000000	5.263011e+08	20.000000	21.000000	1300.000000	1.000000	1.000000	1872.000000	1950.000000	0.000
25%	731.000000	5.284770e+08	20.000000	58.000000	7441.000000	5.000000	5.000000	1954.000000	1965.000000	0.000
50%	1464.000000	5.354532e+08	50.000000	68.000000	9439.000000	6.000000	5.000000	1973.000000	1993.000000	0.000
75%	2197.000000	9.071811e+08	70.000000	80.000000	11553.000000	7.000000	6.000000	2001.000000	2004.000000	164.000
max	2930.000000	1.007100e+09	190.000000	313.000000	215245.000000	10.000000	9.000000	2010.000000	2010.000000	1600.000

8 rows × 38 columns

Next, Let run the second dataset. To study the characters, parameters, the shape, and the statistics of the dataset

In [ ]:
```
# Second Dataset "Housing_2" first five rows
housing2.head()
```

Out[ ]:

	Order	PID	MS SubClass	SalePrice
0	1	526301100	20	215000
1	2	526350040	20	105000
2	3	526351010	20	172000
3	4	526353030	20	244000
4	5	527105010	60	189900

In [ ]:
```
# Checking the index for the number of row.
```

```
housing2.index
```

Out[ ]:
```
RangeIndex(start=0, stop=2933, step=1)
```

In [ ]:
```
# Check the number rows and columns on the dataset.
housing2.shape
```

Out[ ]:
```
(2933, 4)
```

In [ ]:
```
# Checking the index of each columns for proper identification.
housing2.columns
```

Out[ ]:
```
Index(['Order', 'PID', 'MS SubClass', 'SalePrice'], dtype='object')
```

Data Cleaning

It is the process of removing corrupted, incorrect capturing, duplicate and incomplete data within a dataset. This could also happen at the point of merging dataset. (Tableau, 2023)

Housing_1 Dataset

Let determine the total numbers of null values present in the dataset.

In [ ]:
```
# Let determine the total numbers of null values in the dataset
housing1.isnull().sum().sum()
print(f'The total numbers of null values present in the dataset is =', housing1.isnull().sum().sum())
```

```
The total numbers of null values present in the dataset is = 15761
```

In [ ]:
```
# Finding the number of rows with null values
housing1_null = (housing1[housing1.isnull().any(axis=1)])
print('The numbers of rows with null values in the data set is,', len(housing1_null))
```

```
The numbers of rows with null values in the data set is, 2933
```

Note: It is all the rows that contain null value(s). Therefore, none will be removed.

Let check the Null values via columns

In [ ]:
```
# Now we consider dropping the columns with NULL values from the DataFrame
housing1_clean = housing1.dropna(axis=1)

# Checking the length of housing1_clean.
print(f'The numbers of clean columns with no Null values is, {len(housing1_clean.columns)}.')
```

```
The numbers of clean columns with no Null values is, 54.
```

In [ ]:
```
# Let check the numbers of columns with null values.
Columns_dropped = len(housing1.columns) - len(housing1_clean.columns)
print('The numbers of columns dropped is =', Columns_dropped)
print()

#Percentage of columns with Null values.
Columns_dropped_percentage = round((Columns_dropped/len(housing1.columns))*100, 2)
print(f'The percentage of columns dropped is', Columns_dropped_percentage,'%')
```

```
The numbers of columns dropped is = 27

The percentage of columns dropped is 33.33 %
```

In [ ]:
```
# Check if there is still any null values in our clean data 'hse.clean'
housing1_clean.isnull().sum()
```

Out[ ]:

Order	0
PID	0
MS SubClass	0
MS Zoning	0
Lot Area	0
Street	0
Lot Shape	0
Land Contour	0
Utilities	0
Lot Config	0
Land Slope	0
Neighborhood	0
Condition 1	0
Condition 2	0
Bldg Type	0
House Style	0
Overall Qual	0
Overall Cond	0
Year Built	0
Year Remod/Add	0
Roof Style	0
Roof Matl	0
Exterior 1st	0
Exterior 2nd	0
Exter Qual	0
Exter Cond	0
Foundation	0
Heating	0
Heating QC	0
Central Air	0
1st Flr SF	0
2nd Flr SF	0
Low Qual Fin SF	0
Gr Liv Area	0
Full Bath	0
Half Bath	0
Bedroom AbvGr	0
Kitchen AbvGr	0
Kitchen Qual	0
TotRms AbvGrd	0

```
Functional         0
Fireplaces         0
Paved Drive        0
Wood Deck SF       0
Open Porch SF      0
Enclosed Porch     0
3Ssn Porch         0
Screen Porch       0
Pool Area          0
Misc Val           0
Mo Sold            0
Yr Sold            0
Sale Type          0
Sale Condition     0
dtype: int64
```

In [ ]:
```
# Let see the cleaned "housing1" data set new dimensions
housing1_clean.shape
```

Out[ ]:
```
(2933, 54)
```

As calculated, The number of columns with NULL values that was dropped in 'housing1_clean' is 27, and the percentage of dropped columns is significant at, 33.3%. Hence, the full data of housing1 will be used.

Housing_2 Dataset

Let determine the total numbers of null values present in the dataset.

In [ ]:
```
housing2_clean = housing2.isnull().sum().sum()
print('The number of null values present in housing2 dataset is =', housing2_clean)
```

```
The number of null values present in housing2 dataset is = 0
```

In [ ]:
```
housing2.shape
```

Out[ ]:
```
(2933, 4)
```

No missing value in the 'housing2' dataset. Dataset "Housing2" will be merged with dataset "hosuing1".

Note: Merging dataset 'housing2' and 'housing1_clean' is easy because they both have the same numbers of rows. Duplicate will be check after dataset merging.

Preprocessing

Merging Dataset

The 'housing1' and 'housing2' will now merged into a single dataframe called 'hse'. This is also because both the housing1 and housing2 have the same number of rows. And they have the same PID numbers.

In [ ]:
```
# Merging Dataset
hse = housing2.merge(housing1, on=['Order', 'PID', 'MS SubClass'], how = 'inner')
# let see the first five rows
hse.head()
```

Out[ ]:

	Order	PID	MS SubClass	SalePrice	MS Zoning	Lot Frontage	Lot Area	Street	Alley	Lot Shape	...	Screen Porch	Pool Area	Pool QC	Fence	Misc Feature	Misc Val
0	1	526301100	20	215000	RL	141.0	31770	Pave	NaN	IR1	...	0	0	NaN	NaN	NaN	0
1	2	526350040	20	105000	RH	80.0	11622	Pave	NaN	Reg	...	120	0	NaN	MnPrv	NaN	0
2	3	526351010	20	172000	RL	81.0	14267	Pave	NaN	IR1	...	0	0	NaN	NaN	Gar2	12500
3	4	526353030	20	244000	RL	93.0	11160	Pave	NaN	Reg	...	0	0	NaN	NaN	NaN	0
4	5	527105010	60	189900	RL	74.0	13830	Pave	NaN	IR1	...	0	0	NaN	MnPrv	NaN	0

5 rows × 82 columns

Let see the data type and new shape of the newly merged dataset

In [ ]:
```
# Let check the newly merged dataset type;
print(type(hse))
```

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

In [ ]:
```
hse.shape
```

Out[ ]:
```
(2939, 82)
```

We now have 2,939 rows and 82 columns. The rows of our newly merged dataset has increased by 5 and the columns increased by 3.

In [ ]:
```
# Checking for total missing values in the newly merged dataset.
hse.isnull().sum().sum()
print("The total missing values in the newly merged dataset is =", hse.isnull().sum().sum())
```

```
The total missing values in the newly merged dataset is = 15785
```

The numbers of null values has increased by 24 after merged. This is for our information at this point.

Additional Data Cleaning

Now we consider dropping the columns with NULL values in the newly merged DataFrame

In [ ]:
```
# dropping columns
hse_clean = hse.dropna(axis=1)

# Checking the length of hse_clean.
print('The numbers of clean columns with no null values is,', len(hse_clean.columns))

# Let check the numbers of columns with null values.
Columns_dropped = len(hse.columns) - len(hse_clean.columns)
print('Numbers of dropped columns is', Columns_dropped)

#Percentage of columns with Null values.
Columns_dropped_percentage = round((Columns_dropped/len(hse.columns))*100, 2)
print('The percentage of dropped columns is', Columns_dropped_percentage,'%')
```

```
The numbers of clean columns with no null values is, 55
Numbers of dropped columns is 27
The percentage of dropped columns is 32.93 %
```

Again, this percentage of null values in the newly merged dataset 'hse' is high. Hence, should not be dropped. However, relatively to the data function or vizualization, this would be considered.

Duplicate Check

Let check for duplicate in our dataset.

In [ ]:
```
hse.duplicated().sum()
print(f'There are {hse.duplicated().sum()} duplicates in our newlt created dataset')
```

```
There are 9 duplicates in our newlt created dataset
```

Let check the duplicated rows that we have and drop them.

In [ ]:
```
# checking the duplicated rows
hse[hse.duplicated()]
```

Out[ ]:

	Order	PID	MS SubClass	SalePrice	MS Zoning	Lot Frontage	Lot Area	Street	Alley	Lot Shape	...	Screen Porch	Pool Area	Pool QC	Fence	Misc Feature	Mis Va
146	146	535175070	20	167500	RL	73.0	9300	Pave	NaN	Reg	...	143	0	NaN	NaN	NaN	
147	146	535175070	20	167500	RL	73.0	9300	Pave	NaN	Reg	...	143	0	NaN	NaN	NaN	
148	146	535175070	20	167500	RL	73.0	9300	Pave	NaN	Reg	...	143	0	NaN	NaN	NaN	
150	147	535175180	20	108538	RL	87.0	10725	Pave	NaN	IR1	...	0	0	NaN	NaN	NaN	
151	147	535175180	20	108538	RL	87.0	10725	Pave	NaN	IR1	...	0	0	NaN	NaN	NaN	
152	147	535175180	20	108538	RL	87.0	10725	Pave	NaN	IR1	...	0	0	NaN	NaN	NaN	
154	148	535179020	20	159500	RL	80.0	10032	Pave	NaN	Reg	...	160	0	NaN	GdWo	NaN	
155	148	535179020	20	159500	RL	80.0	10032	Pave	NaN	Reg	...	160	0	NaN	GdWo	NaN	
156	148	535179020	20	159500	RL	80.0	10032	Pave	NaN	Reg	...	160	0	NaN	GdWo	NaN	

9 rows × 82 columns

In [ ]:
```
# Let drop the duplicated rows to avoid repetition and arriving at wrong values.
hse = hse.drop_duplicates()
```

In [ ]:
```
hse.shape
```

Out[ ]:
```
(2930, 82)
```

Let see the complete list of the columns, their counts, their names and their data type.

In [ ]:
```
# Let see the info of the newly merged dataset
hse.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 2930 entries, 0 to 2938
Data columns (total 82 columns):
 #   Column           Non-Null Count  Dtype
---  ------           --------------  -----
 0   Order            2930 non-null   int64
 1   PID              2930 non-null   int64
 2   MS SubClass      2930 non-null   int64
 3   SalePrice        2930 non-null   int64
 4   MS Zoning        2930 non-null   object
 5   Lot Frontage     2440 non-null   float64
 6   Lot Area         2930 non-null   int64
 7   Street           2930 non-null   object
 8   Alley            198 non-null    object
 9   Lot Shape        2930 non-null   object
 10  Land Contour     2930 non-null   object
 11  Utilities        2930 non-null   object
 12  Lot Config       2930 non-null   object
 13  Land Slope       2930 non-null   object
 14  Neighborhood     2930 non-null   object
 15  Condition 1      2930 non-null   object
 16  Condition 2      2930 non-null   object
 17  Bldg Type        2930 non-null   object
 18  House Style      2930 non-null   object
 19  Overall Qual     2930 non-null   int64
 20  Overall Cond     2930 non-null   int64
 21  Year Built       2930 non-null   int64
 22  Year Remod/Add   2930 non-null   int64
 23  Roof Style       2930 non-null   object
 24  Roof Matl        2930 non-null   object
 25  Exterior 1st     2930 non-null   object
 26  Exterior 2nd     2930 non-null   object
 27  Mas Vnr Type     1155 non-null   object
 28  Mas Vnr Area     2907 non-null   float64
 29  Exter Qual       2930 non-null   object
 30  Exter Cond       2930 non-null   object
 31  Foundation       2930 non-null   object
 32  Bsmt Qual        2850 non-null   object
 33  Bsmt Cond        2850 non-null   object
 34  Bsmt Exposure    2847 non-null   object
 35  BsmtFin Type 1   2850 non-null   object
```

```
36  BsmtFin SF 1     2929 non-null   float64
37  BsmtFin Type 2   2849 non-null   object
38  BsmtFin SF 2     2929 non-null   float64
39  Bsmt Unf SF      2929 non-null   float64
40  Total Bsmt SF    2929 non-null   float64
41  Heating          2930 non-null   object
42  Heating QC       2930 non-null   object
43  Central Air      2930 non-null   object
44  Electrical       2929 non-null   object
45  1st Flr SF       2930 non-null   int64
46  2nd Flr SF       2930 non-null   int64
47  Low Qual Fin SF  2930 non-null   int64
48  Gr Liv Area      2930 non-null   int64
49  Bsmt Full Bath   2928 non-null   float64
50  Bsmt Half Bath   2928 non-null   float64
51  Full Bath        2930 non-null   int64
52  Half Bath        2930 non-null   int64
53  Bedroom AbvGr    2930 non-null   int64
54  Kitchen AbvGr    2930 non-null   int64
55  Kitchen Qual     2930 non-null   object
56  TotRms AbvGrd    2930 non-null   int64
57  Functional       2930 non-null   object
58  Fireplaces       2930 non-null   int64
59  Fireplace Qu     1508 non-null   object
60  Garage Type      2773 non-null   object
61  Garage Yr Blt    2771 non-null   float64
62  Garage Finish    2771 non-null   object
63  Garage Cars      2929 non-null   float64
64  Garage Area      2929 non-null   float64
65  Garage Qual      2771 non-null   object
66  Garage Cond      2771 non-null   object
67  Paved Drive      2930 non-null   object
68  Wood Deck SF     2930 non-null   int64
69  Open Porch SF    2930 non-null   int64
70  Enclosed Porch   2930 non-null   int64
71  3Ssn Porch       2930 non-null   int64
72  Screen Porch     2930 non-null   int64
73  Pool Area        2930 non-null   int64
74  Pool QC          13 non-null     object
75  Fence            572 non-null    object
76  Misc Feature     106 non-null    object
```

```
 77  Misc Val         2930 non-null   int64
 78  Mo Sold          2930 non-null   int64
 79  Yr Sold          2930 non-null   int64
 80  Sale Type        2930 non-null   object
 81  Sale Condition   2930 non-null   object
dtypes: float64(11), int64(28), object(43)
memory usage: 1.9+ MB
```

In [ ]:
```
#Let look at the statisics of the newly merged data
hse.describe()
```

Out[ ]:

	Order	PID	MS SubClass	SalePrice	Lot Frontage	Lot Area	Overall Qual	Overall Cond	Year Built	Remod/
count	2930.00000	2.930000e+03	2930.000000	2930.000000	2440.000000	2930.000000	2930.000000	2930.000000	2930.000000	2930.00
mean	1465.50000	7.144645e+08	57.387372	180796.060068	69.224590	10147.921843	6.094881	5.563140	1971.356314	1984.26
std	845.96247	1.887308e+08	42.638025	79886.692357	23.365335	7880.017759	1.411026	1.111537	30.245361	20.86
min	1.00000	5.263011e+08	20.000000	12789.000000	21.000000	1300.000000	1.000000	1.000000	1872.000000	1950.00
25%	733.25000	5.284770e+08	20.000000	129500.000000	58.000000	7440.250000	5.000000	5.000000	1954.000000	1965.00
50%	1465.50000	5.354536e+08	50.000000	160000.000000	68.000000	9436.500000	6.000000	5.000000	1973.000000	1993.00
75%	2197.75000	9.071811e+08	70.000000	213500.000000	80.000000	11555.250000	7.000000	6.000000	2001.000000	2004.00
max	2930.00000	1.007100e+09	190.000000	755000.000000	313.000000	215245.000000	10.000000	9.000000	2010.000000	2010.00

8 rows × 39 columns

Let use transpose function on the dataset to review the complete statistics on all dataset columns.

In [ ]:
```
hse.describe().transpose()
```

Out[ ]:

	count	mean	std	min	25%	50%	75%	max
Order	2930.0	1.465500e+03	8.459625e+02	1.0	7.332500e+02	1465.5	2.197750e+03	2.930000e+03
PID	2930.0	7.144645e+08	1.887308e+08	526301100.0	5.284770e+08	535453620.0	9.071811e+08	1.007100e+09
MS SubClass	2930.0	5.738737e+01	4.263802e+01	20.0	2.000000e+01	50.0	7.000000e+01	1.900000e+02
SalePrice	2930.0	1.807961e+05	7.988669e+04	12789.0	1.295000e+05	160000.0	2.135000e+05	7.550000e+05
Lot Frontage	2440.0	6.922459e+01	2.336533e+01	21.0	5.800000e+01	68.0	8.000000e+01	3.130000e+02
Lot Area	2930.0	1.014792e+04	7.880018e+03	1300.0	7.440250e+03	9436.5	1.155525e+04	2.152450e+05
Overall Qual	2930.0	6.094881e+00	1.411026e+00	1.0	5.000000e+00	6.0	7.000000e+00	1.000000e+01
Overall Cond	2930.0	5.563140e+00	1.111537e+00	1.0	5.000000e+00	5.0	6.000000e+00	9.000000e+00
Year Built	2930.0	1.971356e+03	3.024536e+01	1872.0	1.954000e+03	1973.0	2.001000e+03	2.010000e+03
Year Remod/Add	2930.0	1.984267e+03	2.086029e+01	1950.0	1.965000e+03	1993.0	2.004000e+03	2.010000e+03
Mas Vnr Area	2907.0	1.018968e+02	1.791126e+02	0.0	0.000000e+00	0.0	1.640000e+02	1.600000e+03
BsmtFin SF 1	2929.0	4.426296e+02	4.555908e+02	0.0	0.000000e+00	370.0	7.340000e+02	5.644000e+03
BsmtFin SF 2	2929.0	4.972243e+01	1.691685e+02	0.0	0.000000e+00	0.0	0.000000e+00	1.526000e+03
Bsmt Unf SF	2929.0	5.592625e+02	4.394942e+02	0.0	2.190000e+02	466.0	8.020000e+02	2.336000e+03
Total Bsmt SF	2929.0	1.051615e+03	4.406151e+02	0.0	7.930000e+02	990.0	1.302000e+03	6.110000e+03
1st Flr SF	2930.0	1.159558e+03	3.918909e+02	334.0	8.762500e+02	1084.0	1.384000e+03	5.095000e+03
2nd Flr SF	2930.0	3.354560e+02	4.283957e+02	0.0	0.000000e+00	0.0	7.037500e+02	2.065000e+03
Low Qual Fin SF	2930.0	4.676792e+00	4.631051e+01	0.0	0.000000e+00	0.0	0.000000e+00	1.064000e+03
Gr Liv Area	2930.0	1.499690e+03	5.055089e+02	334.0	1.126000e+03	1442.0	1.742750e+03	5.642000e+03
Bsmt Full Bath	2928.0	4.313525e-01	5.248202e-01	0.0	0.000000e+00	0.0	1.000000e+00	3.000000e+00
Bsmt Half Bath	2928.0	6.113388e-02	2.452536e-01	0.0	0.000000e+00	0.0	0.000000e+00	2.000000e+00
Full Bath	2930.0	1.566553e+00	5.529406e-01	0.0	1.000000e+00	2.0	2.000000e+00	4.000000e+00

	count	mean	std	min	25%	50%	75%	max
Half Bath	2930.0	3.795222e-01	5.026293e-01	0.0	0.000000e+00	0.0	1.000000e+00	2.000000e+00
Bedroom AbvGr	2930.0	2.854266e+00	8.277311e-01	0.0	2.000000e+00	3.0	3.000000e+00	8.000000e+00
Kitchen AbvGr	2930.0	1.044369e+00	2.140762e-01	0.0	1.000000e+00	1.0	1.000000e+00	3.000000e+00
TotRms AbvGrd	2930.0	6.443003e+00	1.572964e+00	2.0	5.000000e+00	6.0	7.000000e+00	1.500000e+01
Fireplaces	2930.0	5.993174e-01	6.479209e-01	0.0	0.000000e+00	1.0	1.000000e+00	4.000000e+00
Garage Yr Blt	2771.0	1.978132e+03	2.552841e+01	1895.0	1.960000e+03	1979.0	2.002000e+03	2.207000e+03
Garage Cars	2929.0	1.766815e+00	7.605664e-01	0.0	1.000000e+00	2.0	2.000000e+00	5.000000e+00
Garage Area	2929.0	4.728197e+02	2.150465e+02	0.0	3.200000e+02	480.0	5.760000e+02	1.488000e+03
Wood Deck SF	2930.0	9.375188e+01	1.263616e+02	0.0	0.000000e+00	0.0	1.680000e+02	1.424000e+03
Open Porch SF	2930.0	4.753345e+01	6.748340e+01	0.0	0.000000e+00	27.0	7.000000e+01	7.420000e+02
Enclosed Porch	2930.0	2.301160e+01	6.413906e+01	0.0	0.000000e+00	0.0	0.000000e+00	1.012000e+03
3Ssn Porch	2930.0	2.592491e+00	2.514133e+01	0.0	0.000000e+00	0.0	0.000000e+00	5.080000e+02
Screen Porch	2930.0	1.600205e+01	5.608737e+01	0.0	0.000000e+00	0.0	0.000000e+00	5.760000e+02
Pool Area	2930.0	2.243345e+00	3.559718e+01	0.0	0.000000e+00	0.0	0.000000e+00	8.000000e+02
Misc Val	2930.0	5.063515e+01	5.663443e+02	0.0	0.000000e+00	0.0	0.000000e+00	1.700000e+04
Mo Sold	2930.0	6.271331e+00	3.572365e+00	1.0	4.000000e+00	6.0	8.000000e+00	1.200000e+02
Yr Sold	2930.0	2.007790e+03	1.316613e+00	2006.0	2.007000e+03	2008.0	2.009000e+03	2.010000e+03

Full representation of each column title and the type of data contained

- Order_x = Observation number
- PID = Parcel identification number
- MS SubClass_x = Identifies the type of dwelling involved in the sale

- SalePrice = Sale price $$
- Order_y = Observation number
- MS SubClass_y = Identifies the type of dwelling involved in the sale
- MS Zoning = Identifies the general zoning classification of the sale
- Lot Frontage = Linear feet of street connected to property

- Lot Area = Lot size in square feet
- Street = Type of road access to property
- Alley = Type of Alley access to property
- Lot Shape = General shape of property
- Land Contour = Flatness of the property
- Utilities = Type of utilities available
- Lot Config = Lot configuration
- Land Slope = Slope of property

- Neighborhood = Physical locations within Ames city limits (map available)
- Condition 1 = Proximity to various conditions
- Condition 2 = Proximity to various conditions (if more than one is present)
- Bldg Type = Type of dwelling
- House Style = Style of dwelling
- Overall Qual = Rates the overall material and finish of the house
- Overall Cond = Rates the overall condition of the house
- Year Built = Original construction date

- Year Remod/Add = Remodel date (same as construction date if no remodeling or additions)
- Roof Style = Type of roof
- Roof Matl = Roof material
- Exterior 1st = Exterior covering on house
- Exterior 2nd = Exterior covering on house (if more than one material)
- Mas Vnr Type = Masonry veneer type
- Mas Vnr Area = Masonry veneer area in square feet

- Exter Qual = Evaluates the quality of the material on the exterior
- Exter Cond = Evaluates the present condition of the material on the exterior
- Foundation = Type of foundation
- Bsmt Qual = Evaluates the height of the basement
- Bsmt Cond = Evaluates the general condition of the basement
- Bsmt Exposure = Refers to walkout or garden level walls
- BsmtFin Type 1 = Rating of basement finished area
- BsmtFin SF 1 = Type 1 finished square feet

- BsmtFin Type 2 = Rating of basement finished area (if multiple types)
- BsmtFin SF 2 = Type 2 finished square feet
- Bsmt Unf SF = Unfinished square feet of basement area
- Total Bsmt SF = Total square feet of basement area
- Heating = Type of heating
- Heating QC = Heating quality and condition
- Central Air = Central air conditioning
- Electrical = Electrical system

- 1st Flr SF = First Floor square feet
- 2nd Flr SF = Second floor square feet
- Low Qual Fin SF = Low quality finished square feet (all floors)
- Gr Liv Area = Above grade (ground) living area square feet
- Bsmt Full Bath = Basement full bathrooms
- Bsmt Half Bath = Basement half bathrooms
- Full Bath = Full bathrooms above grade
- Half Bath = Half baths above grade

- Bedroom AbvGr = Bedrooms above grade (does NOT include basement bedrooms)
- Kitchen AbvGr = Kitchens above grade
- Kitchen Qual = Kitchen quality
- TotRms AbvGrd = Total rooms above grade (does not include bathrooms)
- Functional = Home functionality (Assume typical unless deductions are warranted)

- Fireplaces = Number of fireplaces
- Fireplace Qu = Fireplace quality

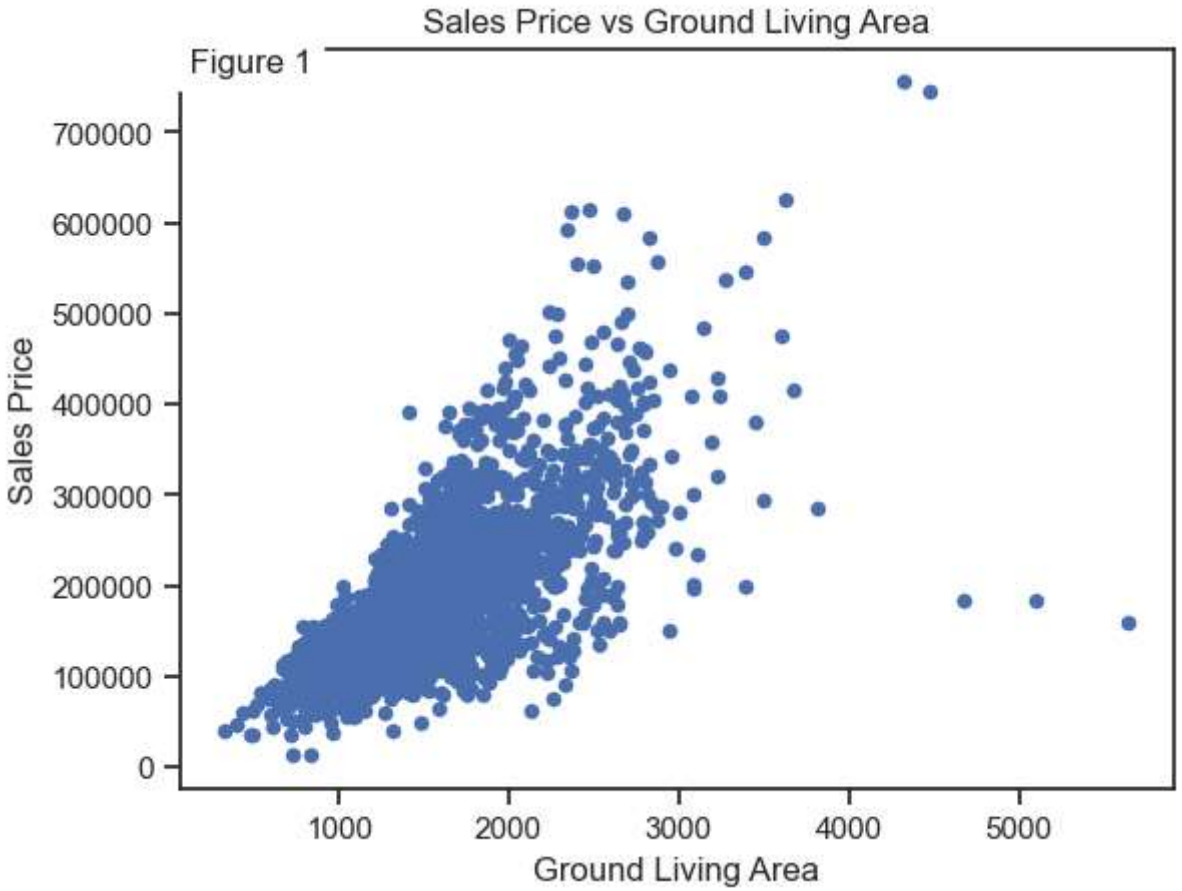
- Garage Type = Garage location
- Garage Yr Blt = Year garage was built
- Garage Finish = Interior finish of the garage
- Garage Cars = Size of garage in car capacity
- Garage Area = Size of garage in square feet
- Garage Qual = Garage quality
- Garage Cond = Garage condition
- Paved Drive = Paved driveway

- Wood Deck SF = Wood deck area in square feet
- Open Porch SF = Open porch area in square feet
- Enclosed Porch = Enclosed porch area in square feet
- 3Ssn Porch = Three season porch area in square feet
- Screen Porch = Screen porch area in square feet
- Pool Area = Pool area in square feet
- Pool QC = Pool quality
- Fence = Fence quality

- Misc Feature = Miscellaneous feature not covered in other categories
- Misc Val = $Value of miscellaneous feature
- Mo Sold = Month Sold (MM)
- Yr Sold = Year Sold (YYYY)
- Sale Type = Type of sale
- Sale Condition = Condition of sale

As recommended, the outliers in the Ground Living Area above 4000 are to be dropped. However, let see the graphical plot representation before dropping them.

In [ ]:
```
# Scatter plot
hse.plot.scatter(y='SalePrice', x='Gr Liv Area', title='Sales Price vs Ground Living Area')

# Name the axes
plt.xlabel('Ground Living Area')
plt.ylabel('Sales Price')

# insert figure number
plt.text(plt.xlim()[0], plt.ylim()[1], ' Figure 1', ha='left', va='top', fontsize=12, bbox=dict(facecolor='white', alpha=1.0))

# Show the plot
plt.show()
```



Now, Removing houses with more that 4000 feet from the column 'Gr Liv Area'

In [ ]:
```
# Let drop the Ground Living Area with more than 4000
hse_a = hse['Gr Liv Area'] > 4000
hse = hse.drop(hse[hse_a].index)

# Let check the dimensions of the dataset at this point
print('The dimension of the new dataset is =',hse.shape)
```

```
The dimension of the new dataset is = (2925, 82)
```

Note: The five (5) rows with Ground Living Area feet above 4000 has been removed as suggested.

Let have the understanding of the proportion of the Null values in the combined dataset.

In [ ]:
```
#Total null values in the dataset
null_count_total = hse.isnull().sum().sum()
print('The total number of null values in the dataset =', null_count_total)
```

```
The total number of null values in the dataset = 15732
```

In [ ]:
```
# Total count of the values in the whole dataset
total_count = hse.count(0, 1).sum()
print('The total count of the values in the whole dataset =', total_count)
```

```
The total count of the values in the whole dataset = 113393
```

In [ ]:
```
Grand_null_percent = round(null_count_total/total_count * 100, 1)
Grand_null_percent
print(f'The percentage of null values is', Grand_null_percent,'%')
```

```
The percentage of null values is 13.9 %
```

The percentage of null values is 13.9%. This is more than 10%. And it could be substantial for some decision making. However, let count the number of Null values per column to understand the mark-up of total null values.

In [ ]:
```
# Null values count per column
null_count = hse.isnull().sum()
null_count
```

Out[ ]:

Order	0
PID	0
MS SubClass	0
SalePrice	0
MS Zoning	0
Lot Frontage	490
Lot Area	0
Street	0
Alley	2727
Lot Shape	0
Land Contour	0
Utilities	0
Lot Config	0
Land Slope	0
Neighborhood	0
Condition 1	0
Condition 2	0
Bldg Type	0
House Style	0
Overall Qual	0
Overall Cond	0
Year Built	0
Year Remod/Add	0
Roof Style	0
Roof Matl	0
Exterior 1st	0
Exterior 2nd	0
Mas Vnr Type	1774
Mas Vnr Area	23
Exter Qual	0
Exter Cond	0
Foundation	0
Bsmt Qual	80
Bsmt Cond	80
Bsmt Exposure	83
BsmtFin Type 1	80
BsmtFin SF 1	1
BsmtFin Type 2	81
BsmtFin SF 2	1
Bsmt Unf SF	1

```
Total Bsmt SF         1
Heating               0
Heating QC            0
Central Air           0
Electrical            1
1st Flr SF            0
2nd Flr SF            0
Low Qual Fin SF       0
Gr Liv Area           0
Bsmt Full Bath        2
Bsmt Half Bath        2
Full Bath             0
Half Bath             0
Bedroom AbvGr         0
Kitchen AbvGr         0
Kitchen Qual          0
TotRms AbvGrd         0
Functional            0
Fireplaces            0
Fireplace Qu       1422
Garage Type         157
Garage Yr Blt       159
Garage Finish       159
Garage Cars           1
Garage Area           1
Garage Qual         159
Garage Cond         159
Paved Drive           0
Wood Deck SF          0
Open Porch SF         0
Enclosed Porch        0
3Ssn Porch            0
Screen Porch          0
Pool Area             0
Pool QC            2914
Fence              2354
Misc Feature       2820
Misc Val              0
Mo Sold               0
Yr Sold               0
Sale Type             0
```

```
Sale Condition        0
dtype: int64
```

Let also determine the percentage of null values per column

In [ ]:
```
# Percentage of null values per columns
column_total = len(hse)
null_percent = round(null_count/column_total * 100, 1)

# Individual column null values percentage
missing_hse = pd.DataFrame({'Column': null_percent.index, '% of null': null_percent.values})
missing_hse
```

Out[ ]:

	Column	% of null
0	Order	0.0
1	PID	0.0
2	MS SubClass	0.0
3	SalePrice	0.0
4	MS Zoning	0.0
5	Lot Frontage	16.8
6	Lot Area	0.0
7	Street	0.0
8	Alley	93.2
9	Lot Shape	0.0
10	Land Contour	0.0
11	Utilities	0.0
12	Lot Config	0.0
13	Land Slope	0.0
14	Neighborhood	0.0
15	Condition 1	0.0
16	Condition 2	0.0
17	Bldg Type	0.0
18	House Style	0.0
19	Overall Qual	0.0
20	Overall Cond	0.0
21	Year Built	0.0

	Column	% of null
22	Year Remod/Add	0.0
23	Roof Style	0.0
24	Roof Matl	0.0
25	Exterior 1st	0.0
26	Exterior 2nd	0.0
27	Mas Vnr Type	60.6
28	Mas Vnr Area	0.8
29	Exter Qual	0.0
30	Exter Cond	0.0
31	Foundation	0.0
32	Bsmt Qual	2.7
33	Bsmt Cond	2.7
34	Bsmt Exposure	2.8
35	BsmtFin Type 1	2.7
36	BsmtFin SF 1	0.0
37	BsmtFin Type 2	2.8
38	BsmtFin SF 2	0.0
39	Bsmt Unf SF	0.0
40	Total Bsmt SF	0.0
41	Heating	0.0
42	Heating QC	0.0
43	Central Air	0.0

	Column	% of null
44	Electrical	0.0
45	1st Flr SF	0.0
46	2nd Flr SF	0.0
47	Low Qual Fin SF	0.0
48	Gr Liv Area	0.0
49	Bsmt Full Bath	0.1
50	Bsmt Half Bath	0.1
51	Full Bath	0.0
52	Half Bath	0.0
53	Bedroom AbvGr	0.0
54	Kitchen AbvGr	0.0
55	Kitchen Qual	0.0
56	TotRms AbvGrd	0.0
57	Functional	0.0
58	Fireplaces	0.0
59	Fireplace Qu	48.6
60	Garage Type	5.4
61	Garage Yr Blt	5.4
62	Garage Finish	5.4
63	Garage Cars	0.0
64	Garage Area	0.0
65	Garage Qual	5.4

	Column	% of null
66	Garage Cond	5.4
67	Paved Drive	0.0
68	Wood Deck SF	0.0
69	Open Porch SF	0.0
70	Enclosed Porch	0.0
71	3Ssn Porch	0.0
72	Screen Porch	0.0
73	Pool Area	0.0
74	Pool QC	99.6
75	Fence	80.5
76	Misc Feature	96.4
77	Misc Val	0.0
78	Mo Sold	0.0
79	Yr Sold	0.0
80	Sale Type	0.0
81	Sale Condition	0.0

The 'Pool QC', 'Fence', 'Misc feature', FirePlace Quality, 'Mas Vnr Type', 'Lot Frontage' and 'Alley' have high percentage of Null values.

Let drop the columns with Null values percentage about 15%.

In [ ]:
```
columns2dropp = ['Mas Vnr Type', 'Alley', 'Pool QC', 'Fireplace Qu', 'Fence', 'Lot Frontage', 'Misc Feature']
hse = hse.drop(columns=columns2dropp)
```

In [ ]:
```
hse.shape
print('The updated dimension of the dataset is ', hse.shape)
```

```
The updated dimension of the dataset is  (2925, 75)
```

Let confirm the removal of the columns with null values percentage above 15%

In [ ]:
```
# Null values count per column
null_count = hse.isnull().sum()
null_count

# Percentage of null values per columns
column_total = len(hse)
null_percent = round(null_count/column_total * 100, 1)

# Individual column null values percentage
missing_hse = pd.DataFrame({'Column': null_percent.index, '% of null': null_percent.values})
missing_hse
```

Out[ ]:

	Column	% of null
0	Order	0.0
1	PID	0.0
2	MS SubClass	0.0
3	SalePrice	0.0
4	MS Zoning	0.0
5	Lot Area	0.0
6	Street	0.0
7	Lot Shape	0.0
8	Land Contour	0.0
9	Utilities	0.0
10	Lot Config	0.0
11	Land Slope	0.0
12	Neighborhood	0.0
13	Condition 1	0.0
14	Condition 2	0.0
15	Bldg Type	0.0
16	House Style	0.0
17	Overall Qual	0.0
18	Overall Cond	0.0
19	Year Built	0.0
20	Year Remod/Add	0.0
21	Roof Style	0.0

	Column	% of null
22	Roof Matl	0.0
23	Exterior 1st	0.0
24	Exterior 2nd	0.0
25	Mas Vnr Area	0.8
26	Exter Qual	0.0
27	Exter Cond	0.0
28	Foundation	0.0
29	Bsmt Qual	2.7
30	Bsmt Cond	2.7
31	Bsmt Exposure	2.8
32	BsmtFin Type 1	2.7
33	BsmtFin SF 1	0.0
34	BsmtFin Type 2	2.8
35	BsmtFin SF 2	0.0
36	Bsmt Unf SF	0.0
37	Total Bsmt SF	0.0
38	Heating	0.0
39	Heating QC	0.0
40	Central Air	0.0
41	Electrical	0.0
42	1st Flr SF	0.0
43	2nd Flr SF	0.0

	Column	% of null
44	Low Qual Fin SF	0.0
45	Gr Liv Area	0.0
46	Bsmt Full Bath	0.1
47	Bsmt Half Bath	0.1
48	Full Bath	0.0
49	Half Bath	0.0
50	Bedroom AbvGr	0.0
51	Kitchen AbvGr	0.0
52	Kitchen Qual	0.0
53	TotRms AbvGrd	0.0
54	Functional	0.0
55	Fireplaces	0.0
56	Garage Type	5.4
57	Garage Yr Blt	5.4
58	Garage Finish	5.4
59	Garage Cars	0.0
60	Garage Area	0.0
61	Garage Qual	5.4
62	Garage Cond	5.4
63	Paved Drive	0.0
64	Wood Deck SF	0.0
65	Open Porch SF	0.0

	Column	% of null
66	Enclosed Porch	0.0
67	3Ssn Porch	0.0
68	Screen Porch	0.0
69	Pool Area	0.0
70	Misc Val	0.0
71	Mo Sold	0.0
72	Yr Sold	0.0
73	Sale Type	0.0
74	Sale Condition	0.0

Let reconfirm the overall percentage of null values.

In [ ]:
```
#Total null values in the dataset
null_count_total = hse.isnull().sum().sum()
print('The total number of counted null values in our latest dataframe is', null_count_total)
#print()
# Total count of the values in the whole dataset
total_count = hse.count(0, 1).sum()
print('The number of values in our new dataframe is', total_count)
#print()
Grand_null_percent = round(null_count_total/total_count * 100, 1)
print(f"The percentage of null values in our latest dataframe is {Grand_null_percent}%")
```

```
The total number of counted null values in our latest dataframe is 1231
The number of values in our new dataframe is 110958
The percentage of null values in our latest dataframe is 1.1%
```

1.1% null values percentage is very negliable and therefore can not affect our report.

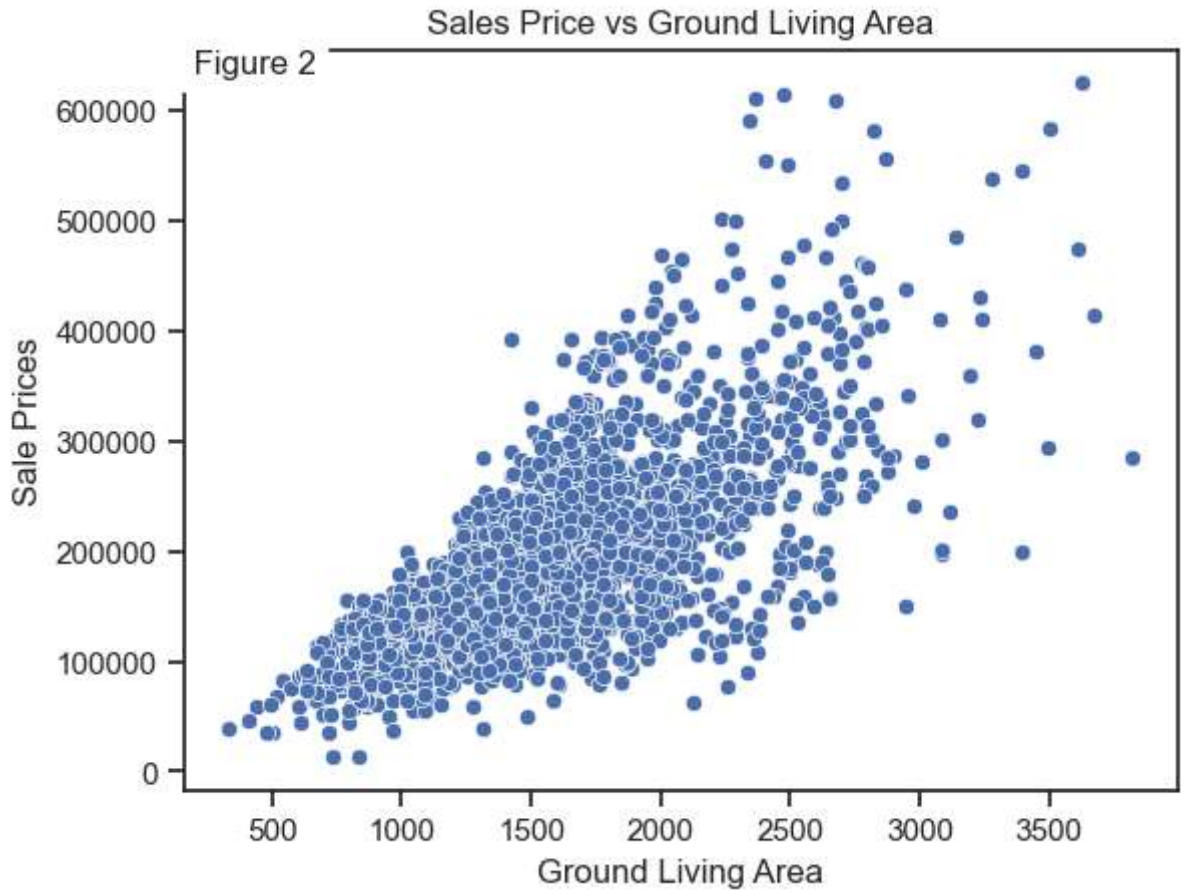
Exploratory Data Analysis

Focus:

What is the preference of buyers in the amount ready to pay for a house, Ground Living Area or Lot Area?

Having removed the Ground Living Area above 4000 feet ealier. Let plot the Sale Prices relationship aganist the Ground Living Area. To understand the correlation of the Ground Living Area with the price paid for the houses.

In [ ]:
```
# Scatter plot with seaborn
sns.scatterplot(x='Gr Liv Area', y='SalePrice', data=hse)

# Name axes and title
plt.xlabel('Ground Living Area')
plt.ylabel('Sale Prices')
plt.title('Sales Price vs Ground Living Area')

# insert figure number
plt.text(plt.xlim()[0], plt.ylim()[1], ' Figure 2', ha='left', va='top', fontsize=12, bbox=dict(facecolor='white', alpha=1.0))


plt.show()
```



Graphically, we deduced that Ground Living Area has a good positive correlation with Sale Price. We would now calculate the correlation in percentage.

In [ ]:
```
# Calculating correlation between Sale Price and Ground Living Area
Sale_GLA_Corr = round(hse_clean['SalePrice'].corr(hse_clean['Gr Liv Area'])*100, 1)
print(f"The correlation between 'Sale Price' and 'Ground Living Area' is {Sale_GLA_Corr}%")
```

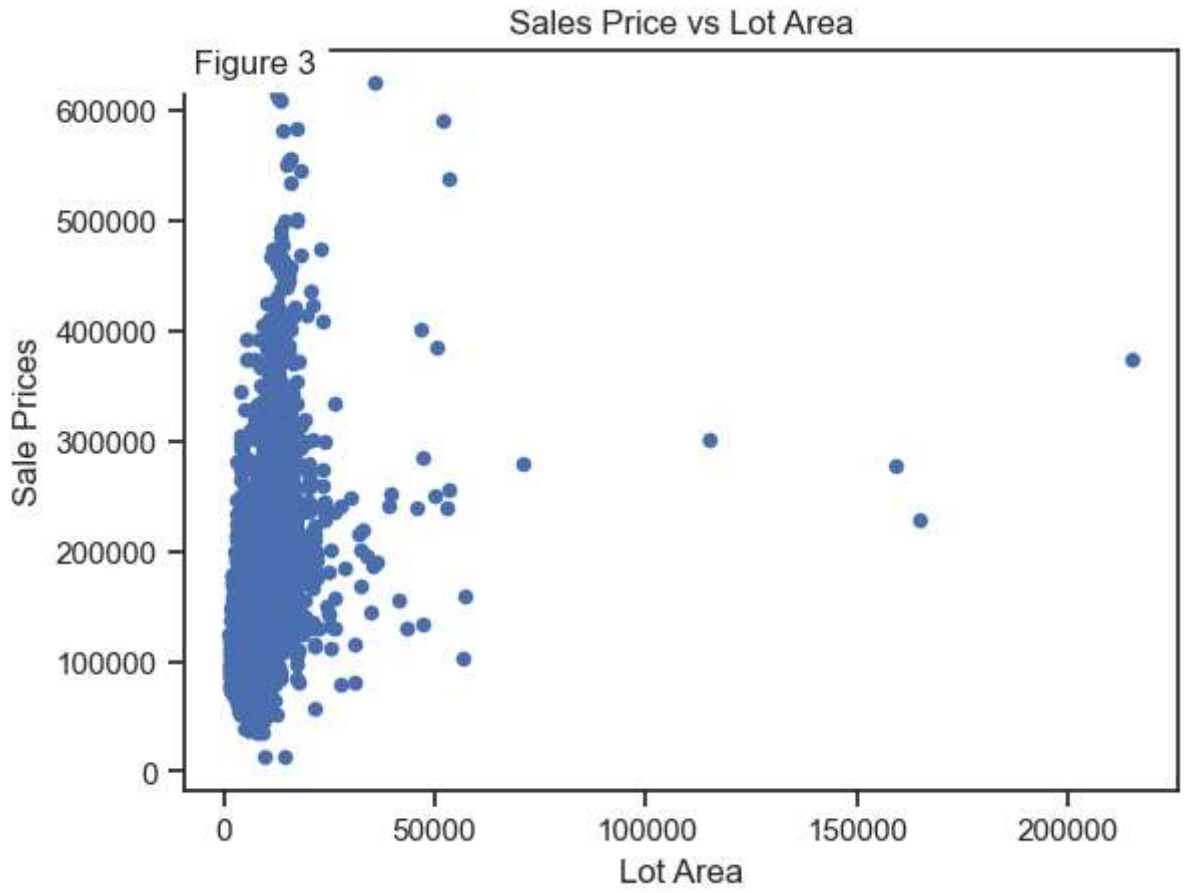
```
The correlation between 'Sale Price' and 'Ground Living Area' is 70.7%
```

There is 70.7% positive correlation between sales price and Ground Living Area. Therefore, as the size of the Ground Living Area increases, the sales price tends to increase as well.

Next

Larger Lot Area is often excepted to sell for high prices, and vice versa. Let explore the graphically representation of the relationship between Sale Prices and Lot Area.

In [ ]:
```
# Using scatter plot
hse.plot.scatter(y='SalePrice', x='Lot Area')

# Labelling
plt.xlabel('Lot Area')
plt.ylabel('Sale Prices')
plt.title('Sales Price vs Lot Area')

# name the figure
plt.text(plt.xlim()[0], plt.ylim()[1], ' Figure 3', ha='left', va='top', fontsize=12, bbox=dict(facecolor='white', alpha=1.0))

plt.show()
```

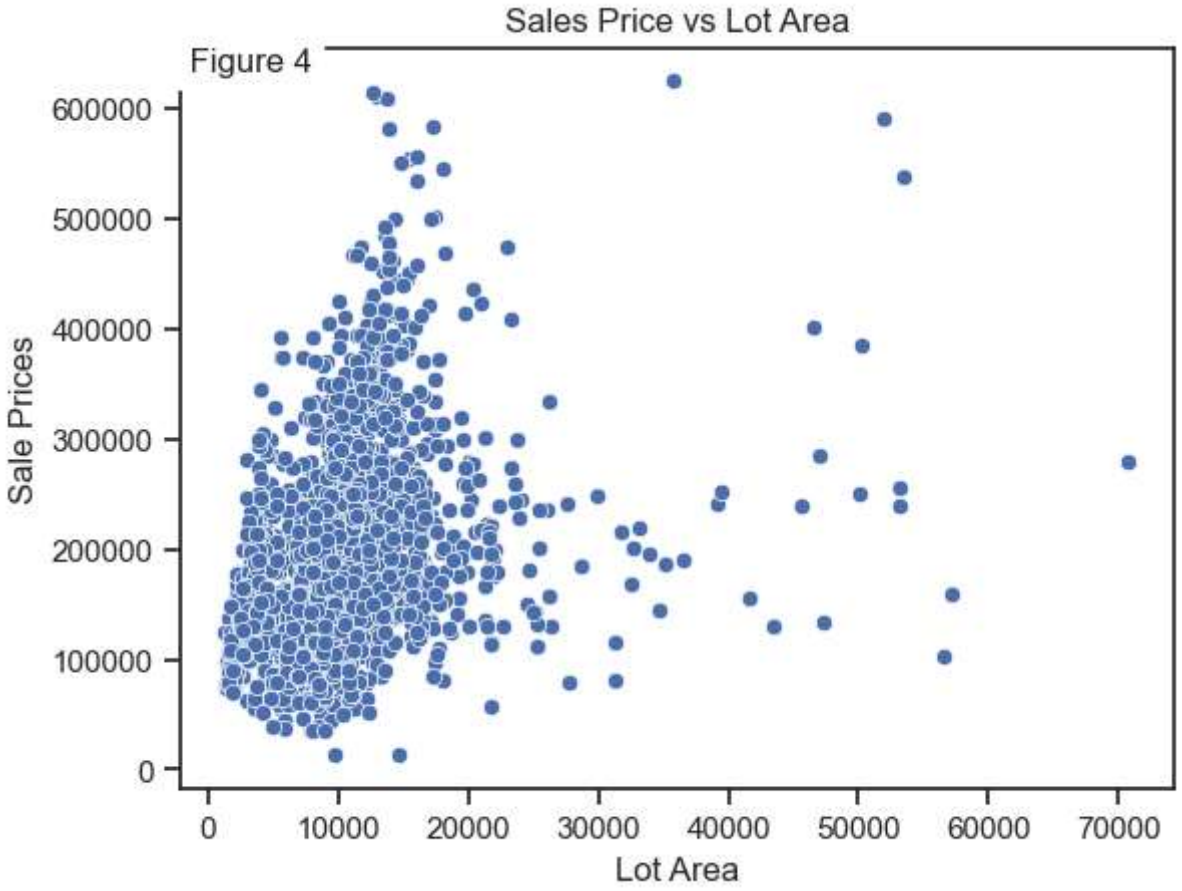


There are 4 outliers among these variables from the plot representation. The variables above 75,000 feet Lot Area will now be removed for closer look into the correlation between the Sale Price and Lot Area in seaborn package.

In [ ]:
```
# Removing outliers
hse_b = hse['Lot Area'] > 75000
hse = hse.drop(hse[hse_b].index)
```

In [ ]:
```
# Scatter plot with seaborn
sns.scatterplot(x='Lot Area', y='SalePrice', data=hse)

# labelling
plt.xlabel('Lot Area')
```

```
plt.ylabel('Sale Prices')
plt.title('Sales Price vs Lot Area')

# name the figure
plt.text(plt.xlim()[0], plt.ylim()[1], ' Figure 4', ha='left', va='top', fontsize=12, bbox=dict(facecolor='white', alpha=1.0))

plt.show()
```



Above is a well spread representation after removing the outliers. Let calculate the correlation co-efficient.

In [ ]:
```
Corr_LotArea = round(hse_clean['SalePrice'].corr(hse_clean['Lot Area'])*100, 1)
print(f"The correlation co-efficient between Sale Price and Lot Area is {Corr_LotArea}%.")
```

```
The correlation co-efficient between Sale Price and Lot Area is 26.6%.
```

Key Findings and Interpretation

From calculation above, the correlation between the Sale prices and Lot Area is a weak positive correlation of 26.6%. This observation buttressed the graphical plot representation above. There is weak positive correlation between the two variables. The Sale prices will not likely increase as a result of increase in Lot Area. Buyers might be considering other features such as the location of the houses, proximity to major road etc. Although, Tzu-Chin Lin and Alan W. Evans,(2000, pp.387) observed that, prices increases with increase in Lot Area, but our statistic shows otherwise. The prices buyers are ready to pay is not totally relative to the Lot Area.

Therefore, it can be deduced that, the sales price is not dependent on the Lot Area.

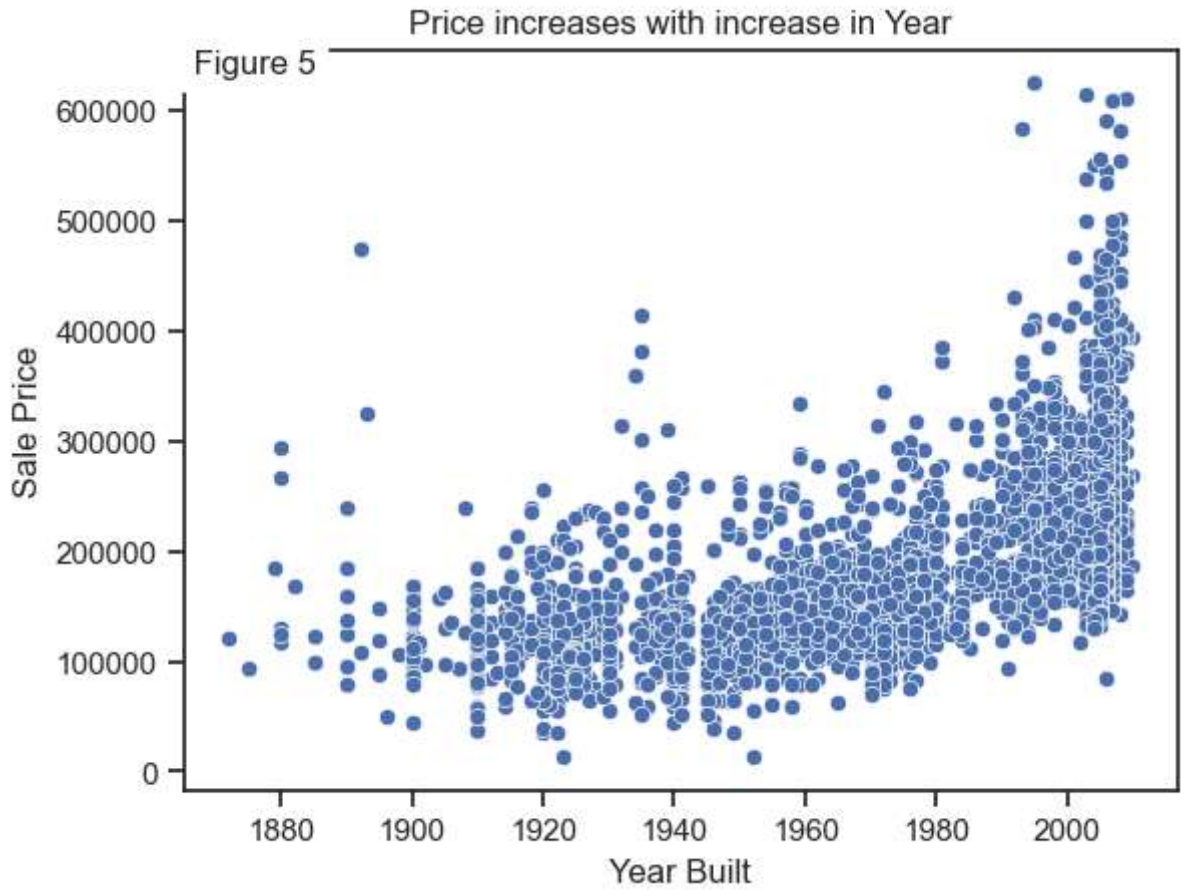
From the Sale Price correlation of 71.7% with the Ground Living Area and 26.6% correlation with Lot Area, it could be concluded that the buyers interest in the price to pay for a house is in the feet of the Ground Living Area and not the Lot Area.

Focus:

- What is the relationship between the year houses were built and the Sale price they well sold?
- Does inflation has impact on the numbers of houses built across all the years?

Next, we check the Sale Price relationship with the year house was build using scatter plot representation.

In [ ]:
```
# Scatter plot with seaborn
sns.scatterplot(x='Year Built', y='SalePrice', data=hse)

# Labelling
plt.xlabel('Year Built')
plt.ylabel('Sale Price')
plt.title('Price increases with increase in Year')

# name the figure
plt.text(plt.xlim()[0], plt.ylim()[1], ' Figure 5', ha='left', va='top', fontsize=12, bbox=dict(facecolor='white', alpha=1.0))

plt.show()
```



In [ ]:
```
Sales_YearBuilt_Corr = round(hse_clean['Year Built'].corr(hse_clean['SalePrice'])*100, 1)
print(f"The 'Sale Price' correlation with 'Year Built' is {Sales_YearBuilt_Corr}%.")
```

```
The 'Sale Price' correlation with 'Year Built' is 55.9%.
```

Key Findings and Interpretation

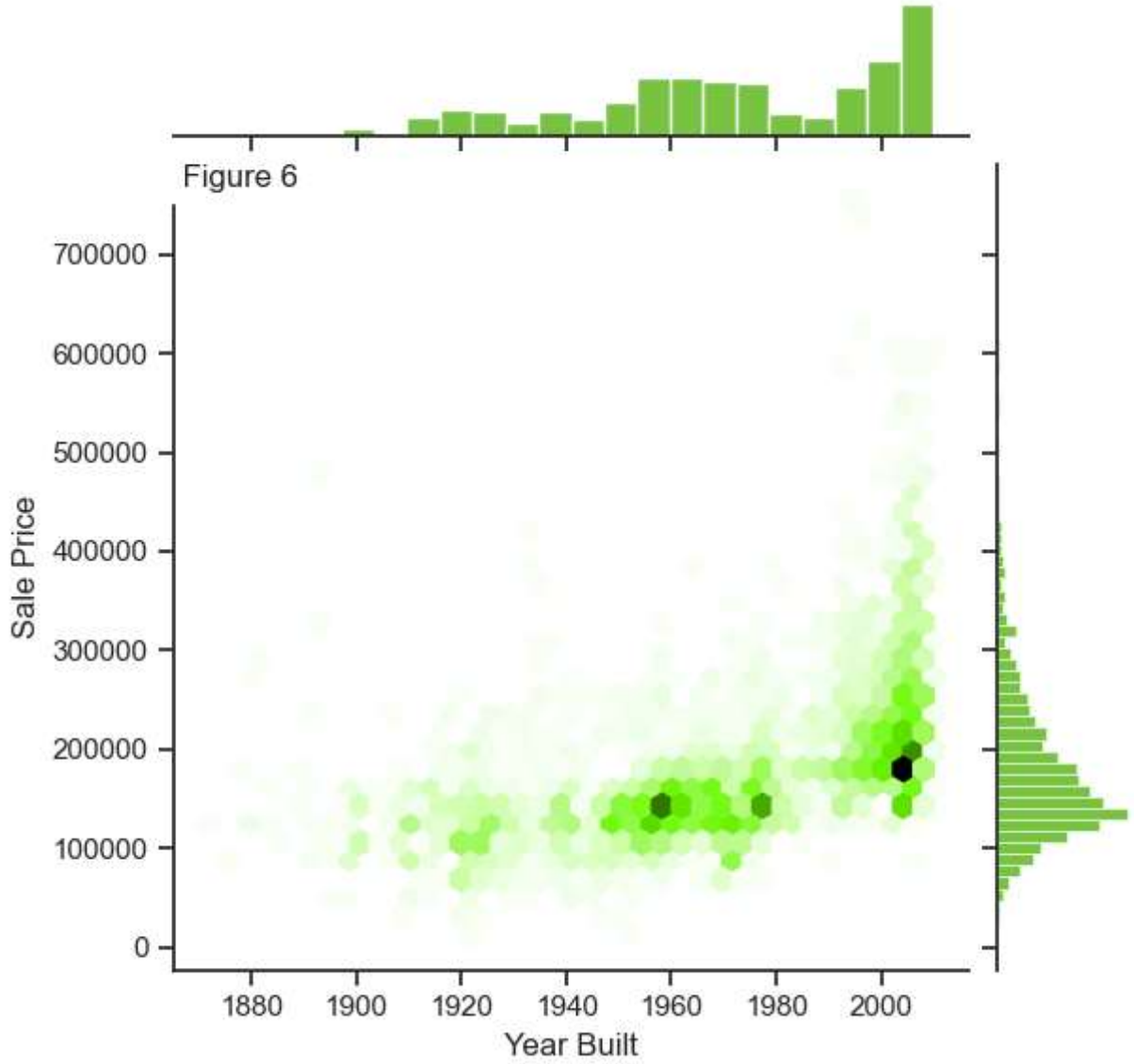
We can conclude that the Sale Price increase with increase in the year houses were built.

Sale Price has a positive correlation of 55.9% with year the house was built. The house buyers will pay for a house based on how modern the structure is, and the quality of the facilities in it. Also, it could be said that the impact of inflation over the years on the cost of productions

(building) is responsible for the increase in Sale Price and hence the buyers payments. The impact of inflation and other governmental monetary policy on the economic led to increase in cost of production and hence the Sale Price also. The average inflation rate of USA between 1960 to 2022 is 3.8% according to WorldData info. (2022).

Let use jointplot to see the correlation of both variables and individual performance histogram representation in a jointplot of seaborn package.

In [ ]:
```
#
sns.set_theme(style="ticks")
sns.jointplot(x=hse_clean['Year Built'], y=hse_clean['SalePrice'], kind="hex", color="#4CB301")

# Labelling
plt.xlabel('Year Built')
plt.ylabel('Sale Price')

# name the figure
plt.text(plt.xlim()[0], plt.ylim()[1], ' Figure 6', ha='left', va='top', fontsize=12, bbox=dict(facecolor='white', alpha=1.0))


plt.show()
```



Key Findings and Interpretation

The numbers of houses built generally has increase across the year since 1880. However, It is also observed that, some of the great recession also affected housing Sale Prices. It is noticable from the histogram on the top side, that there were some fall in numbers in about two different periods of the numbers of houses built, in 1930s and 1980s. Further details of the periods of recession in USA is available on Wikipedia (Wikipedia, 2023).

Also, the Sale Price histogram at the right sides of the representation shows that Sale Price has increased over the years.

However, the graphical representation did not clearly review the impact of year December 2007 to June 2009 great global economic recession on the house built, which particularly emanated from USA. Therefore, let look into the Sale Price and number of built houses experience during this period.(Investopedia, 2023).

We shall consider period year 2006 to year 2010, a five year (5) period. But first, let plot the line graph of the relationship between Sale Price and Year built for the whole dataset. In doing this, we shall see the average yearly price.

In [ ]:
```
# Calculate the averageyearly price.
hse_yearly_sales = round(hse.groupby('Year Built')['SalePrice'].mean().reset_index(), 2)
hse_yearly_sales
```

Out[ ]:

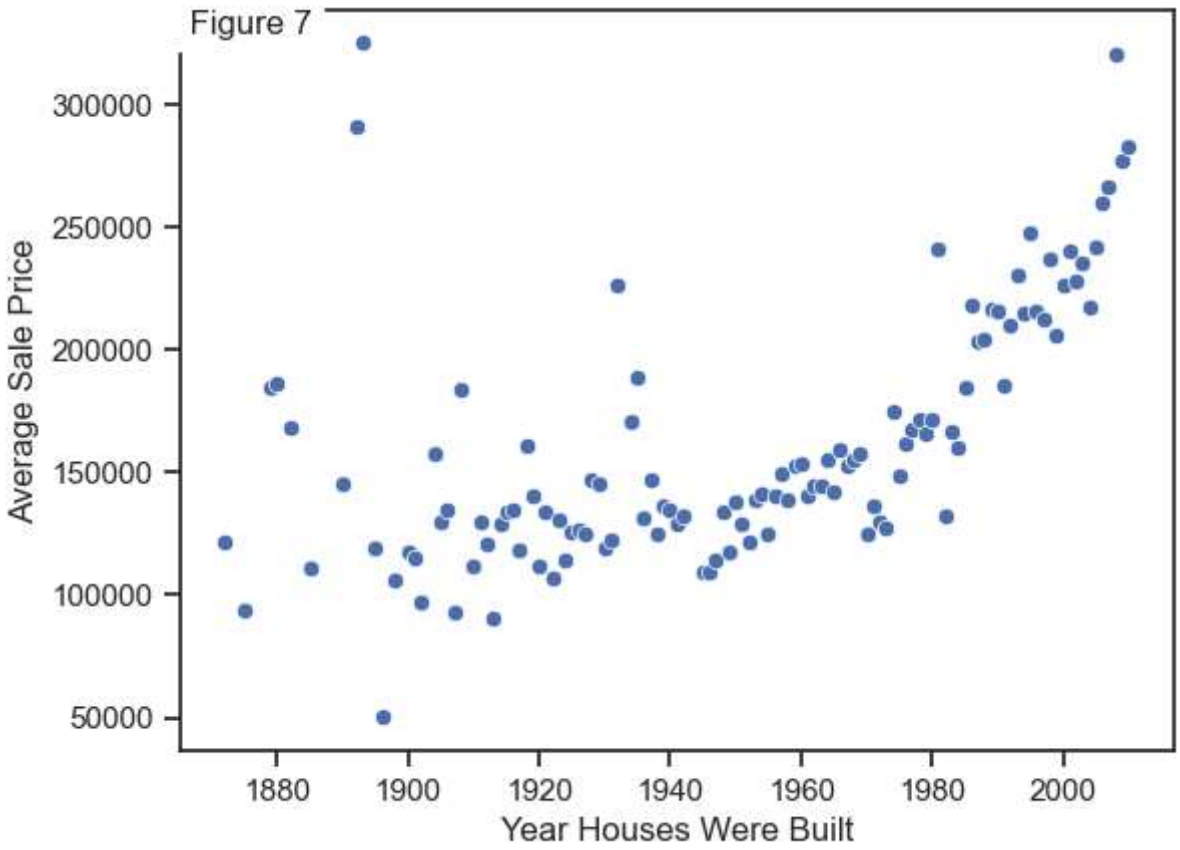
	Year Built	SalePrice
0	1872	122000.00
1	1875	94000.00
2	1879	185000.00
3	1880	186695.80
4	1882	168000.00
...	...	...
113	2006	259623.85
114	2007	266676.56
115	2008	320358.74
116	2009	277097.44
117	2010	283116.00

118 rows × 2 columns

Let see the scatter plot of the average sale prices of each year before studing the line graph.

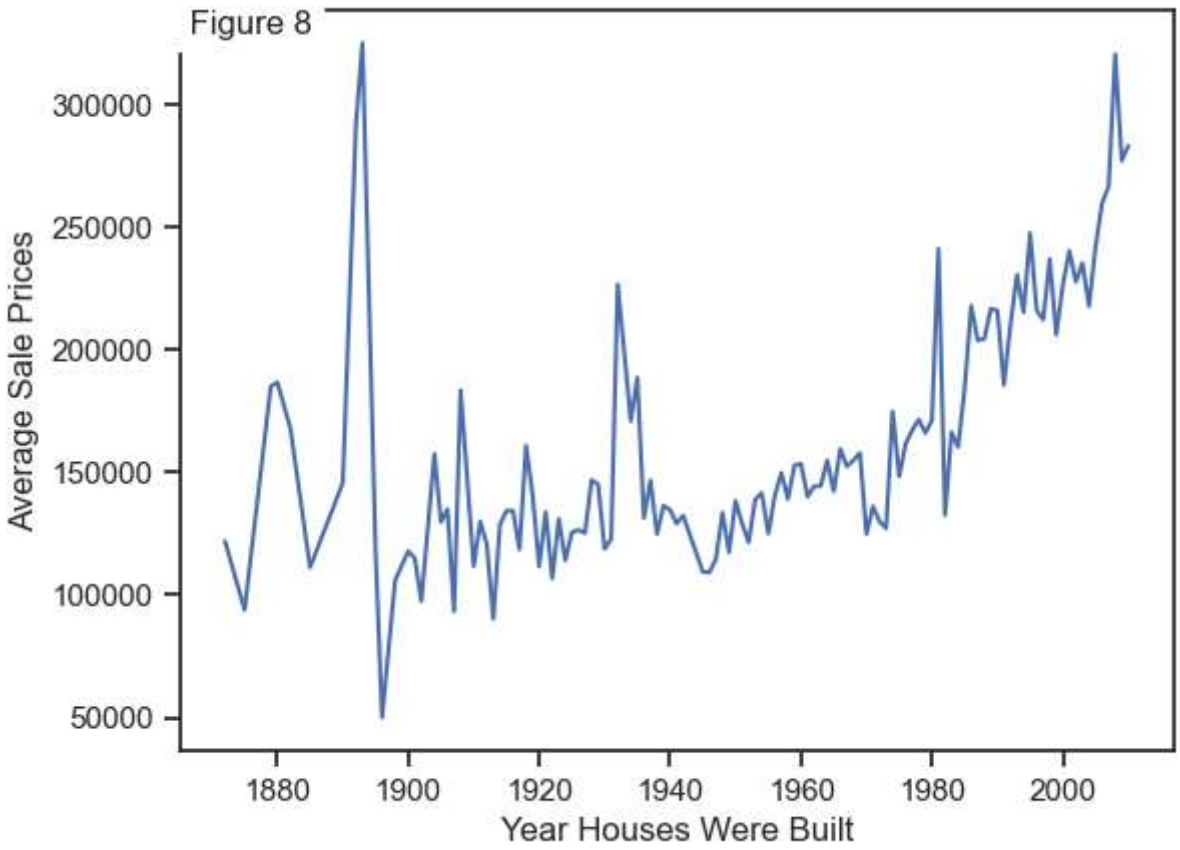
In [ ]:

```
# Seaborn plot
sns.scatterplot(x='Year Built', y='SalePrice', data=hse_yearly_sales)

# Labelling
plt.xlabel('Year Houses Were Built')
plt.ylabel('Average Sale Price')

# name the figure
plt.text(plt.xlim()[0], plt.ylim()[1], ' Figure 7', ha='left', va='top', fontsize=12, bbox=dict(facecolor='white', alpha=1.0))

plt.show()
```



The line graph of the Average Sale Price against the built year.

In [ ]:
```
# Plot the line graph of the average yearly price.
plt.plot(hse_yearly_sales['Year Built'], (hse_yearly_sales['SalePrice']))

# Labelling
plt.xlabel('Year Houses Were Built')
plt.ylabel('Average Sale Prices')
# name the figure
plt.text(plt.xlim()[0], plt.ylim()[1], ' Figure 8', ha='left', va='top', fontsize=12, bbox=dict(facecolor='white', alpha=1.0))

plt.show()
```



Key Findings and Interpretation

According to Figure 5, there are few number of houses built between year 1880 to year 1900. However, there is high average sale price in the figure above for this period. This is because, the lower the numbers of houses built, the higher the possible average sale price achieved.

The constant incessant recession between year 1880 to year 1890 (Wikipedia) was reponsible for the low numbers of houses that could be avoided or built in those year. Overall, average sale price has gradually and significantly increase over the years except for impact of reception in year 1930 and 1980. (Wikipedia).

It can be concluded that, during inflation, the numbers of houses built were reduced.

Univariate

Univariate is the simplest form of data analysis. it requires just one variable and does not deal with causes or relationship. Its purpose is to describe; summarise and find pattern with data.

Focus:

- Does inflation has impact on the numbers of houses sold between year 2006 and year 2010 the years?

Again, following our data dictionary, we will vizualize the impact of inflation on the numbers of houses sold between 2006 and year 2010 (the period of the last great recession between Dec 2007 to June 2009 ((Wikipedia) using piechart plot. This inflation particularly started in USA. A good one for our study).

When shall consider period year 2006 to year 2010, a five year (5) period. Describe, summarise and express the patterns.

Let count the numbers of houses sold in each year.

In [ ]:
```
# Selecting the rows for the period 2006 to 2010.
hse06_10 = hse['Yr Sold'].value_counts().reset_index()
hse06_10
```

Out[ ]:

	Yr Sold	count
0	2007	688
1	2009	647
2	2006	625
3	2008	620
4	2010	341

Using the pie-chart (percentage) to represent our findings on the numbers of houses sold during year 2006 and 2010.

In [ ]:
```
# Selecting the rows for the period 2005 to 2010.
hse06_10 = hse['Yr Sold'].value_counts().reset_index()
hse06_10_sorted = hse06_10.sort_values(by='Yr Sold')

# Pie chart
plt.pie(hse06_10_sorted['count'], labels=hse06_10_sorted['Yr Sold'], autopct='%1.1f%%', startangle=80)

# Labelling
plt.title('% of Number of Houses Sold between 2006 - 2010')
plt.xlabel('Figure 9')

# plot
plt.show()
```

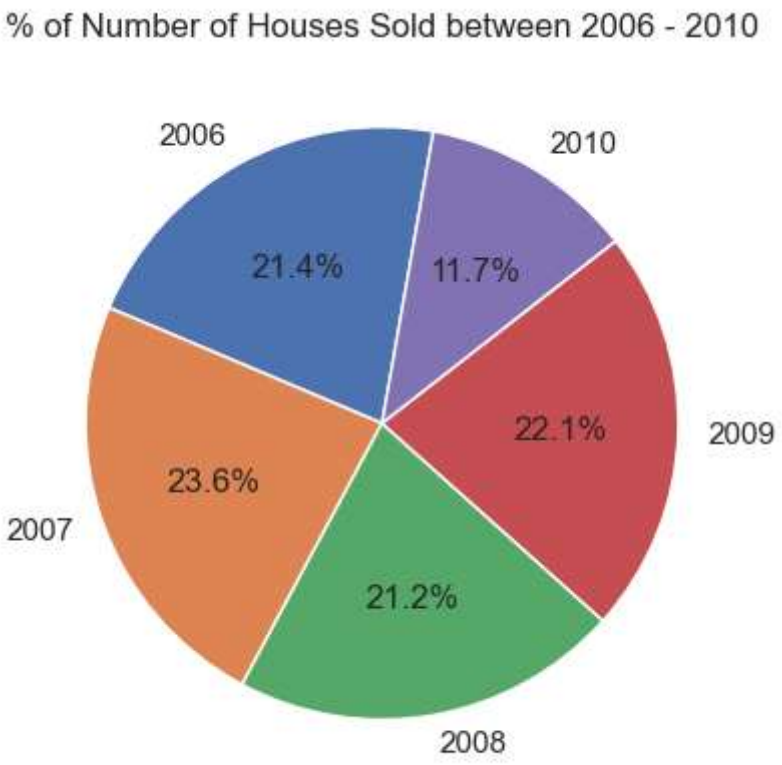


Figure 9

Key Findings and Interpretation

The impact of the global great recession on the number of houses sold in USA between year 2006 to 2010 became visible in year 2010. The fall started in year 2008 with 2.4% decrease from year 2007 23.6%. However, an immediate slight increase of 0.9% was realised in 2009. But amazingly, the impact of the inflation was well visible in year 2010 with a whooping 10.4% fall in the percentage of houses sold between this period.

It can be concluded that inflation has a significant impacted on sales during the period. However, it was heavy in year 2010.

Analysis Focus

- Which house type was highly sold between year 2006 and year 2010?

In [ ]:
```
# copting a dataset
hse_house_style = hse[['PID', 'SalePrice', 'House Style']].copy()
hse_house_style.head()
```

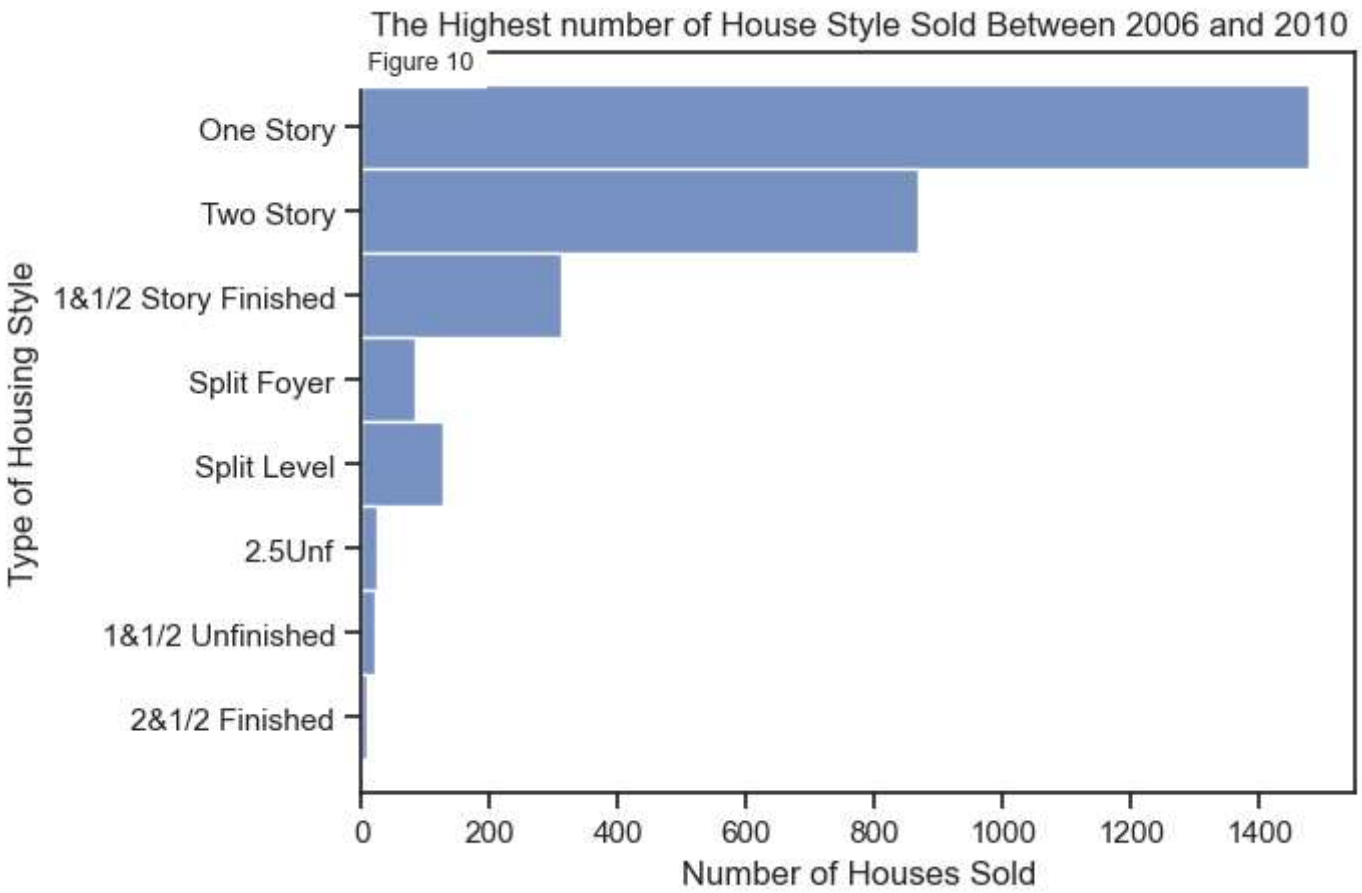
Out[ ]:

	PID	SalePrice	House Style
0	526301100	215000	1Story
1	526350040	105000	1Story
2	526351010	172000	1Story
3	526353030	244000	1Story
4	527105010	189900	2Story

There is 8 categories of houses types, let see which of the house types was highly sold between the year 2006 and year 2010. We will first replace the name of each house type in the House Style column for better representation in the histogram.

In [ ]:
```
# Replacing the names of the variable in  House Style column
hse_house_style['House Style'] = hse_house_style['House Style'].replace(
    {'1Story':'One Story', '2Story':'Two Story', '1.5Fin':'1&1/2 Story Finished',
    'SLvl':'Split Level', 'SFoyer':'Split Foyer', '2.5Fin':'2&1/2 Story Unfinished',
```

```
    '1.5Unf':'1&1/2 Unfinished', '2.5Fin':'2&1/2 Finished'})

# plotting
sns.histplot(data=hse_house_style, y='House Style', bins=len(hse_house_style['House Style'].unique()))

# labelling
plt.xlabel('Number of Houses Sold')
plt.ylabel('Type of Housing Style')
plt.title('The Highest number of House Style Sold Between 2006 and 2010')

# naming the figure
plt.text(plt.xlim()[0], plt.ylim()[1], ' Figure 10', ha='left', va='top', fontsize=9, bbox=dict(facecolor='white', alpha=1.0))

#plot
plt.show()

hse_house_style = hse_house_style['House Style'].value_counts().reset_index()
hse_house_style
```


The Highest number of House Style Sold Between 2006 and 2010
Figure 10
One Story
Two Story
1&1/2 Story Finished
Split Foyer
Split Level
2.5Unf
1&1/2 Unfinished
2&1/2 Finished
Type of Housing Style
0
200
400
600
800
1000
1200
1400
Number of Houses Sold

Out[ ]:

	House Style	count
0	One Story	1478
1	Two Story	869
2	1&1/2 Story Finished	312
3	Split Level	128
4	Split Foyer	83
5	2.5Unf	24
6	1&1/2 Unfinished	19
7	2&1/2 Finished	8

Key Findings and Interpretation

The 'One-Story building' house type is the most sold house type between the year 2006 and year 2010 with 1,478 houses sold. The list sold house style is the two and half story _2nd levl finshed houses. There will require a further investigation.

Bivariate

Analysis Focus;

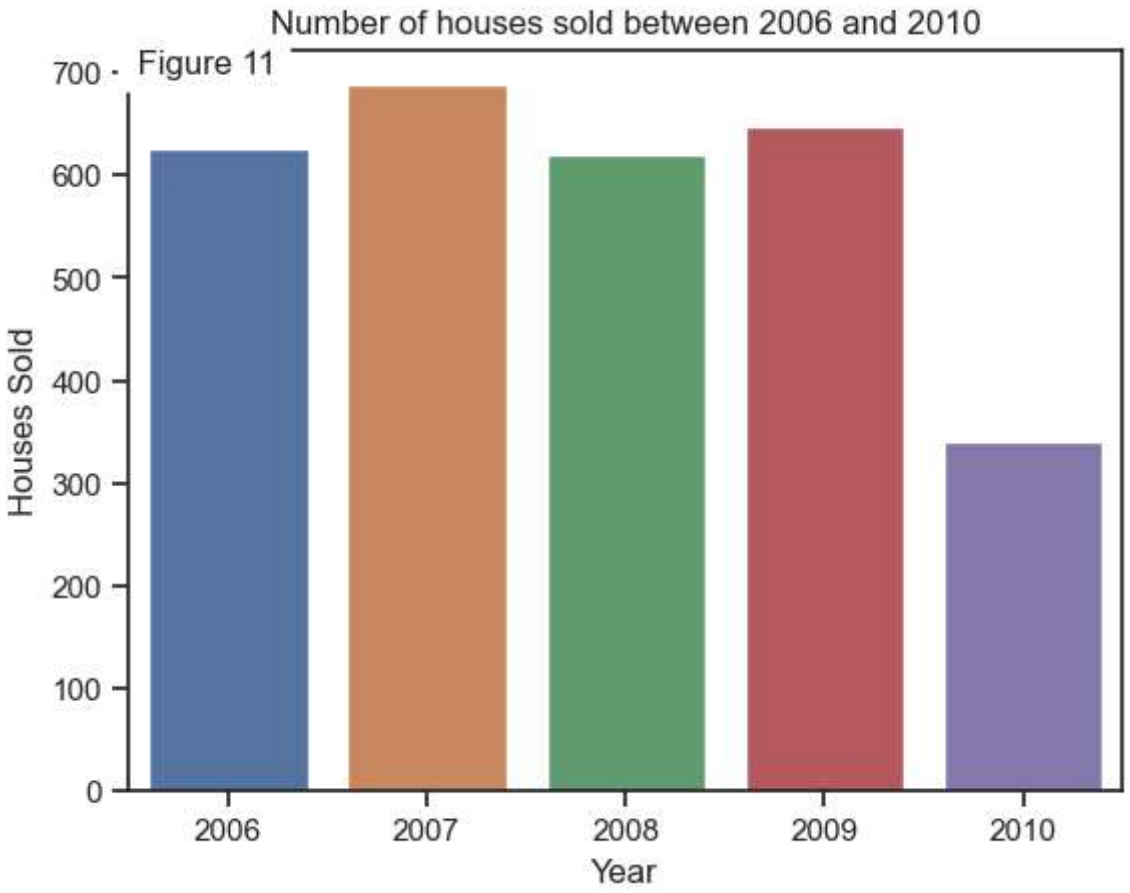
- Using bivariate, let examinse the effect inflation has impact on the numbers of houses sold between year 2006 and year 2010 the years?

Bivariate is a statistic approach that analyse how two different variable related. The aim is to determine if the relationship between the two variable is strong, and in which direction they are linked together. (QuestionPro, 2023)

Let examine the relationship between the numbers of houses sold between year 2006 and year 2010 in relation to the impact of inflation

In [ ]:
```
# plot
sns.barplot(x='Yr Sold', y='count', data=hse06_10, estimator='mean', errorbar=None)
```

```
plt.xlabel('Year')
plt.ylabel('Houses Sold')
plt.title('Number of houses sold between 2006 and 2010')

plt.text(plt.xlim()[0], plt.ylim()[1], ' Figure 11', ha='left', va='top', fontsize=12, bbox=dict(facecolor='white', alpha=1.0)

plt.show()
```



Key Findings and Interpretation

The numbers of houses sold was on an increasing trend before inflation hit in the last month of year 2007. The immediate impact in 2008 was a slow down which was going to regain strength in year 2009 but was evident in year 2010 with a heavy drop in the numbers of houses people could avoid to purchase.

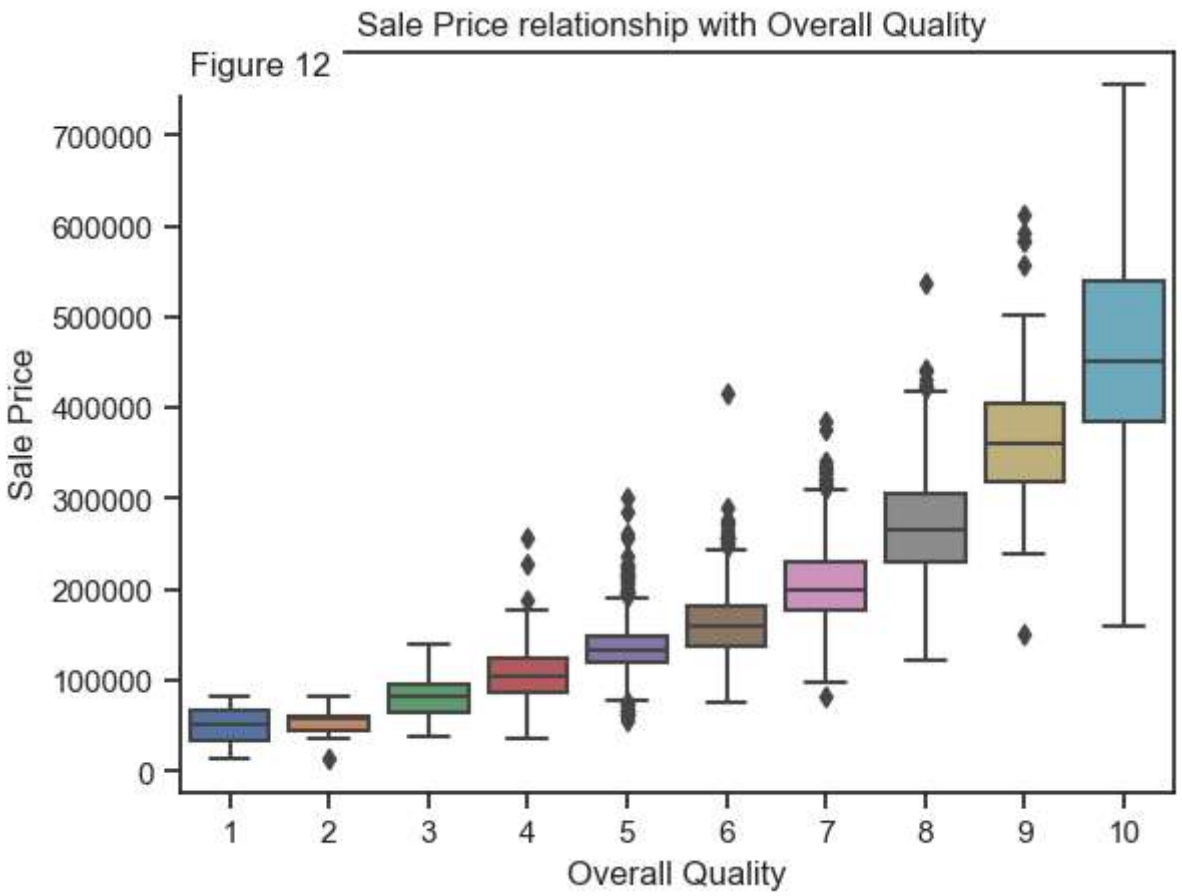
There is impact of imflation on numbers of houses sold.

Focus:

- The relationship between Sale Price and the Overall Quallity of each house?

Next, let examine the relationship of Sales Price with level of the overall quality of the materials and the finishing of the houses. We will use boxplot for this examination.

In [ ]:
```
# plot
sns.boxplot(x='Overall Qual', y='SalePrice', data=hse_clean)

# labelling
plt.xlabel('Overall Quality')
plt.ylabel('Sale Price')
plt.title('Sale Price relationship with Overall Quality ')

# name the figure
plt.text(plt.xlim()[0], plt.ylim()[1], ' Figure 12', ha='left', va='top', fontsize=12, bbox=dict(facecolor='white', alpha=1.0)

plt.show()
```



Key Findings and Interpretation

Figure 12, reviews that the Sale Price of each house increases based on the level of the overall quality state of the house. The average price of the overall quality level of 10 is about $500,000. While the overall quality level below 5 rating is only about$ 100,000.

It can be concluded that people are ready to pay for houses with higher quality if made available.

Focus: What is the correlation of the Sale Price with other variables using heatmap?

Let use heatmap plot to obtain (study) and plot the correlation co-efficient of the following variables 'SalePrice', 'Gr Liv Area', 'Lot Area', 'Overall Qual', 'Overall Cond', 'Year Built', 'Year Remod/Add' against Sale Price.

In [ ]:
```
key_features = hse_clean[['SalePrice', 'Gr Liv Area', 'Lot Area', 'Overall Qual', 'Overall Cond', 'Year Built', 'Year Remod/Ad
key_features
```

Out[ ]:

	SalePrice	Gr Liv Area	Lot Area	Overall Qual	Overall Cond	Year Built	Year Remod/Add
SalePrice	1.000000	0.706812	0.266377	0.799261	-0.100918	0.558581	0.533204
Gr Liv Area	0.706812	1.000000	0.285517	0.570854	-0.115104	0.241950	0.317143
Lot Area	0.266377	0.285517	1.000000	0.097156	-0.034720	0.023269	0.021690
Overall Qual	0.799261	0.570854	0.097156	1.000000	-0.093879	0.597163	0.569974
Overall Cond	-0.100918	-0.115104	-0.034720	-0.093879	1.000000	-0.367922	0.049408
Year Built	0.558581	0.241950	0.023269	0.597163	-0.367922	1.000000	0.612065
Year Remod/Add	0.533204	0.317143	0.021690	0.569974	0.049408	0.612065	1.000000

In [ ]:
```
price_correlation = key_features.loc['SalePrice']
print()
print(f'The individual column correlation with Sale Price is {price_correlation}')
```

```
The individual column correlation with Sale Price is SalePrice         1.000000
Gr Liv Area       0.706812
Lot Area          0.266377
Overall Qual      0.799261
Overall Cond     -0.100918
Year Built        0.558581
Year Remod/Add    0.533204
Name: SalePrice, dtype: float64
```

In [ ]:
```
plt.figure(figsize=(10, 8))

sns.heatmap(key_features, annot=True, cmap='coolwarm', linewidths=1, square=False,
mask=key_features.isnull(), xticklabels=False, yticklabels=True)
```

```
# labelling
plt.title('Sale Price Correlation Relation with Key Features')
plt.xlabel('Figure 13') # use to name the figure

plt.show()
```

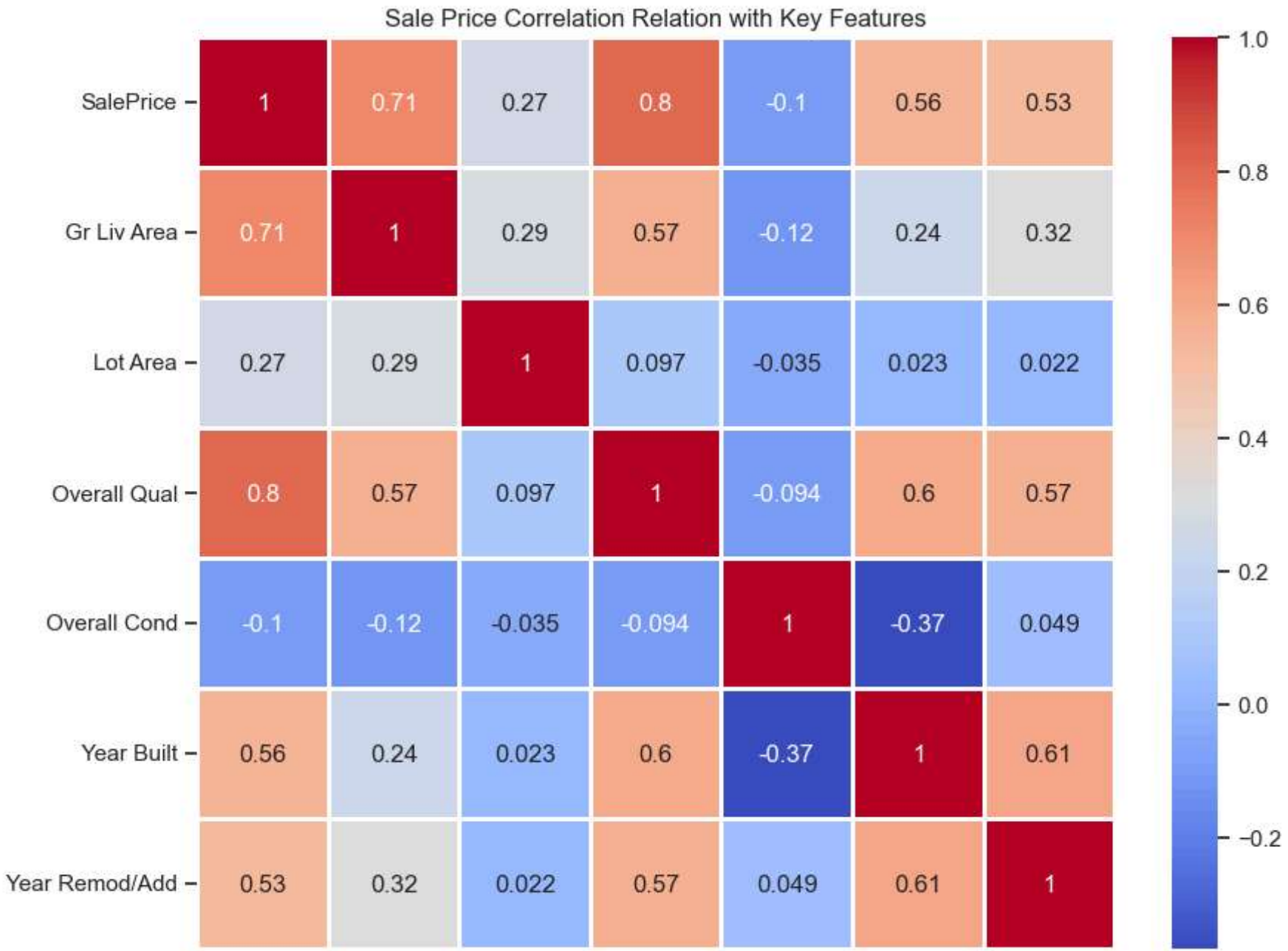

Sale Price Correlation Relation with Key Features
SalePrice
Gr Liv Area
Lot Area
Overall Qual
Overall Cond
Year Built
Year Remod/Add
1
0.71
0.27
0.8
-0.1
0.56
0.53
0.71
1
0.29
0.57
-0.12
0.24
0.32
0.27
0.29
1
0.097
-0.035
0.023
0.022
0.8
0.57
0.097
1
-0.094
0.6
0.57
-0.1
-0.12
-0.035
-0.094
1
-0.37
0.049
0.56
0.24
0.023
0.6
-0.37
1
0.61
0.53
0.32
0.022
0.57
0.049
0.61
1
1.0
0.8
0.6
0.4
0.2
0.0
−0.2

Figure 13

The Kernel crashed while executing code in the the current cell or a previous cell. Please review the code in the cell(s) to id entify a possible cause of the failure. Click here for more info. View Ju pyter log for further details.

Key Findings and Interpretation

From the heatmap, there are variables which are strongly positively correlated with the Sale Price and some which are weakly positively correlated with Sale Price. However, there are some that are negatively correlated with Sale Price.

The Ground Living Area and the Overall Quality are strongly positively correlated with Sale price, which mean the Sale Price would increase with the increase in the Ground Living Area and the quality of the material used in the buidling, e.g. the quality of the roof materials, roof style, the land slope, the quality of the exterior materials among others. However, the Sales price weakly response to the increase in Lot Area as they are weakly correlated. The Lot land size would not succeed in putting pressure on the price for a house. Buyers preferences are not majorly towards the Lot Area.

The Overall Quality, Year Built and Year Remoded are positively correlated with the Sale Price while the Qverall Conditions of the houses is weakly and negatively correlated with Sale Price. That is, the Sale Price increases in line with the Overall Quality, the year the houses were built and or year Remoded. However, the Overall Conditions is well unlikely. This could mean that the buyers have other usage preference, for example, if they have the mindset of buying to rebuild or restructure the house to their taste or purpose for purchasing.

Conclusion
Overall, It could be said that the buyers of the houses considered different criterias in their selection of houses to buy and the amount they are ready to pay a house. For example, buyers are concerns more about the groung living Area that the lot Area. Also, the overall quality of a house is more important over overall conditions, e.g. proxomity, access to road etc.

Limitation and Recommendation.

There are 27 columns with null values in the dataset, 32.9% of the whole columns. The heavy present of null values in columns like, Alley, FirePlace Quality, Masonry veneer type, Pool QC, fence and iscellaneous feature not covered in other categories. The impacts that these have on the overall conditions cannot be full assessed or investigated. On the other hand, replacing any of this variable null value with a position either by 'mean' or 'median' could largely misrepresent the population. Hence, this was not considered.

I recommend that the reasons for high null values is particular columns are investigated, to understand if this is due to data collection reason or other issues. And additional data can be collection to fill the missing gap for further analysis.

References

Tzu-Chin Lin, Alan W. Evans, 2000, Land Economics, The Relationship between the Price of Land and Size of Plot When Plots Are Small. Vol. 76, No. 3. , pp. 386-394. [Viewed 30th November 2023] Available at: https://www.jstor.org/stable/3147036

WorldData.Info, 2023. online. [Viewed 01 November 2023] Available at: https://www.worlddata.info/america/usa/inflation-rates.php#:~:text=During%20the%20observation%20period%20from,year%20inflation%20rate%20was%203.2%25.

Investopedia, 2023. 2008 Recession: What It Was and What Caused It. Online. [Viewed 01 November 2023] Available at: https://www.investopedia.com/terms/g/great-recession.asp

Wikipedia, 2023, List of recessions in USA. [Viewed 01 November 2023] Available at: https://en.wikipedia.org/wiki/List_of_recessions_in_the_United_States

StatisticHowTo, 2023. Univariate Analysis: Definition, Examples. Online. [Viewed 02 December 2023] Available at: https://www.statisticshowto.com/univariate/

QuestionPro, 2023. Bivariate Analysis: What is it, Types + Examples. Online. [Viewed 02 December 2023] Available at: https://www.questionpro.com/blog/bivariate-analysis/#:~:text=Bivariate%20analysis%20is%20a%20statistical,which%20direction%20that%20link%20is

Tableau, 2023. Guide To Data Cleaning: Definition, Benefits, Components, And How To Clean Your Data. Online. [Viewed 03 December 2023] Available at: https://www.tableau.com/learn/articles/what-is-data-cleaning#:~:text=Data%20cleaning%20is%20the%20process,to%20be%20duplicated%20or%20mislabeled.