My grades for **Assignment 1**

Q1

8.5 / 10

Submit here the pdf printout of your entire notebook, including the outputs of your code.

CSCI 4146/6409 - Process of Data Science

Assignment 1

Winter 2024

Due on: 26 January 2024, 23:59 ADT

In [1]:
```
import numpy as np
import pandas as pd
import seaborn as sns
!! pip install nbconvert
pd.set_option("display.max_rows",None)
pd.set_option("display.max_columns",None)
pd.set_option("display.float_format",'{:.2f}'.format)
```

Q1. [0.3] Business Understanding

1. Formulate a business problem that can be solved using the dataset.

   Real Estate Pricing Tool for Sellers:

   The problem statement is to create a tool that will help real estate brokers and homeowners appropriately price houses for sale by taking into account the attributes and current market conditions. Setting competitive prices to draw in buyers without undervaluing the property would be made easier with the help of this tool.

   External Data Requirements:

   To improve accuracy, more data may be required. Examples of this type of data include recent sales of houses nearby that are comparable, current market trends, and economic indicators.

2. For the business problem, propose 3 data science solutions and assess their feasibility (required data, required business capacity). Select the final solution and explain your decision.

   ### *Advanced Predictive Pricing Model:*

   This model forecasts the selling price of real estate properties by combining

recent sales data with comprehensive property information. The model can give sellers and real estate agents a data-driven estimated price for a property that takes into account the current status of the market as well as the property's distinctive features.

Data Requirements: Sales data and comprehensive property details.

Business Capacity: Access to real estate market data and a data science team with machine learning expertise are required.

Feasibility: High, given access to pertinent information and experience.

Comparative Market Analysis (CMA) Tool:

Focusing on market comparison to establish competitive pricing, this tool assesses a property's value by comparing it to similar recently sold properties. It is necessary to comprehend the position of a property in the present market.

Data Requirements: Neighborhood information, real estate listings, and recent comparable sales.

Business Capacity: Software development experience and knowledge of real estate market analysis.

Feasibility: High, standard practice in real estate but requires up-to-date market data.

Interactive Pricing Recommendation System:

With this system, sellers can input specific features of their property and obtain pricing recommendations, providing a dynamic and interactive approach to real estate pricing. The system makes sure that pricing recommendations are in line with the state of the market by keeping up with the most recent developments in the industry.

Data Requirements: Market trends, customer reviews, and dynamic property features.

Business Capacity: proficiency in machine learning, ongoing data upgrades, and user interface design.

Feasibility: Moderate, reliant on interactive technologies and an ongoing system for updating data.

3. For the final solution, identify the prediction subject, its domain concepts and sub-concepts (if there are any). Draw a hierarchical graph of the concepts. (You may use draw.io (https://app.diagrams.net/) to create daigrams)

Advanced Predictive Pricing Model **: This model is chosen because it directly**

supports the objective of accurate pricing, relies on machine learning techniques, and makes use of extensive data for accuracy.

1) Prediction Subject: Property Price

This is the central subject of the model, influenced by various property characteristics, location features, and economic indicators.

2) Domain Concepts:

Property Characteristics:

Encompasses features intrinsic to the property itself.

Location Features:

Concerned with the property's location and its environmental and societal context.

Economic Indicators:

Relates to broader economic factors that influence property value.

3) Sub-Domain Concepts:

Property Characteristics:

Size and Structure (e.g., LotArea, TotalBsmtSF) Age and Condition (e.g., YearBuilt, OverallCond) Style and Quality (e.g., HouseStyle)

Location Features:

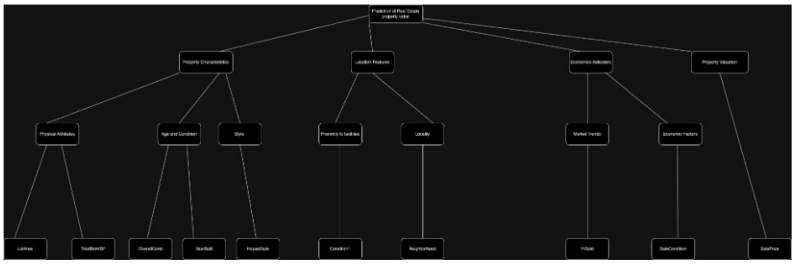
Neighborhood (e.g., Neighborhood) Proximity to Facilities (e.g., Condition1)

Economic Indicators:

Market Trends (e.g., YrSold) Broader Economic Factors (e.g., SaleCondition)

Hierarchical Diagram:

https://drive.google.com/file/d/13daKZNaDcYNp6uqCEShmLAcByCQsql72/view?usp=sharing (https://drive.google.com/file/d/13daKZNaDcYNp6uqCEShmLAcByCQsql72/view?usp=sharing)



4. For the selected domain concepts , design descriptive features that best describe a concept using data from the dataset. Information about data fields is available here (https://www.kaggle.com/competitions/house-prices-advanced-regression-techniques/data?select=data_description.txt). Summarize the resulting **Analytics base table (ABT)** in a table with the following columns:

   i. Feature Name

   ii. Domain Concept

   iii. Feature Description

   iv. Feature Type

   v. Data Type

In [2]:
```
FeatureName = [
    'LotArea', 'TotalBsmtSF', 'YearBuilt', 'OverallCond',
    'HouseStyle', 'Neighborhood', 'Condition1', 'YrSold', 'SaleCond
]


DomainConcept = [
    'Property Characteristics', 'Property Characteristics', 'Proper
    'Property Characteristics', 'Property Characteristics', 'Locati
    'Location Features', 'Economic Indicators', 'Economic Indicator
]


FeatureDescription = [
    'Lot size in square feet', 'Total square feet of basement area'
    'Overall condition rating', 'Style of dwelling', 'Physical loca
    'Proximity to main road or railroad', 'Year Sold', 'Condition o
]


FeatureType = [
    'Size and Structure', 'Size and Structure', 'Age and Condition'
    'Age and Condition', 'Style and Quality', 'Neighborhood',
    'Proximity to Facilities', 'Market Trends', 'Broader Economic F
]


DataType = [
    'Numerical', 'Numerical', 'Numerical',
    'Categorical', 'Categorical', 'Categorical',
    'Categorical', 'Numerical', 'Categorical'
]
```

In [3]:
```
#Analysis Base Table (ABT)
ABT=pd.DataFrame({"FeatureName":FeatureName,"DomainConcept":DomainC
```

In [4]:
```
ABT
```

Out[4]:

	FeatureName	DomainConcept	FeatureDescription	FeatureType	DataType
0	LotArea	Property Characteristics	Lot size in square feet	Size and Structure	Numerical
1	TotalBsmtSF	Property Characteristics	Total square feet of basement area	Size and Structure	Numerical
2	YearBuilt	Property Characteristics	Original construction date	Age and Condition	Numerical
3	OverallCond	Property Characteristics	Overall condition rating	Age and Condition	Categorical
4	HouseStyle	Property Characteristics	Style of dwelling	Style and Quality	Categorical

5	Neighborhood	Location Features	Physical locations within Ames city limits	Neighborhood	Categorical
6	Condition1	Location Features	Proximity to main road or railroad	Proximity to Facilities	Categorical
7	YrSold	Economic Indicators	Year Sold	Market Trends	Numerical
8	SaleCondition	Economic Indicators	Condition of sale	Broader Economic Factors	Categorical

Q2. [0.3] Data Exploration

1. Build the data quality report of the resulting ABT.

In [5]:
```
#Reading the dataset
sales_data=pd.read_csv("house-prices-advanced-regression-techniques
sales_data_ABT=sales_data.loc[: ,FeatureName]
```

In [6]:
```
#Overview of the Dataframe
sales_data_ABT.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1460 entries, 0 to 1459
Data columns (total 9 columns):
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   LotArea        1460 non-null   int64
 1   TotalBsmtSF    1460 non-null   int64
 2   YearBuilt      1460 non-null   int64
 3   OverallCond    1460 non-null   int64
 4   HouseStyle     1460 non-null   object
 5   Neighborhood   1460 non-null   object
 6   Condition1     1460 non-null   object
 7   YrSold         1460 non-null   int64
 8   SaleCondition  1460 non-null   object
dtypes: int64(5), object(4)
memory usage: 102.8+ KB
```

In [7]:
```
#Convert to appropriate data types
sales_data_ABT.convert_dtypes()
```

	LotArea	TotalBsmtSF	YearBuilt	OverallCond	HouseStyle	Neighborhood	Condition1
0	8450	856	2003	5	2Story	CollgCr	Norm
1	9600	1262	1976	8	1Story	Veenker	Feedr
2	11250	920	2001	5	2Story	CollgCr	Norm
3	9550	756	1915	5	2Story	Crawfor	Norm
4	14260	1145	2000	5	2Story	NoRidge	Norm
5	14115	796	1993	5	1.5Fin	Mitchel	Norm
6	10084	1686	2004	5	1Story	Somerst	Norm
7	10382	1107	1973	6	2Story	NWAmes	PosN
8	6120	952	1931	5	1.5Fin	OldTown	Artery
9	7420	991	1939	6	1.5Unf	BrkSide	Artery
10	11200	1040	1965	5	1Story	Sawyer	Norm
11	11924	1175	2005	5	2Story	NridgHt	Norm

In [8]:
```
#Display first row of the dataframe
sales_data_ABT.loc[0]
```

Out[8]:
```
LotArea             8450
TotalBsmtSF          856
YearBuilt           2003
OverallCond            5
HouseStyle        2Story
Neighborhood     CollgCr
Condition1          Norm
YrSold              2008
SaleCondition     Normal
Name: 0, dtype: object
```

In [9]:
```
sales_data_ABT
```

Out[9]:

	LotArea	TotalBsmtSF	YearBuilt	OverallCond	HouseStyle	Neighborhood	Condition1
0	8450	856	2003	5	2Story	CollgCr	Norm
1	9600	1262	1976	8	1Story	Veenker	Feedr
2	11250	920	2001	5	2Story	CollgCr	Norm
3	9550	756	1915	5	2Story	Crawfor	Norm
4	14260	1145	2000	5	2Story	NoRidge	Norm
5	14115	796	1993	5	1.5Fin	Mitchel	Norm
6	10084	1686	2004	5	1Story	Somerst	Norm
7	10382	1107	1973	6	2Story	NWAmes	PosN
8	6120	952	1931	5	1.5Fin	OldTown	Artery
9	7420	991	1939	6	1.5Unf	BrkSide	Artery
10	11200	1040	1965	5	1Story	Sawyer	Norm

Continous feature Report

1. Min
2. 1st quartile
3. Mean
4. 2nd quartile - Median
5. 3rd quartile
6. Max
7. Standard deviation
8. Total num of instances
9. % missing values
10. Cardinality - num of distinct values for a feature

In [10]:
```
#Generating a report for continuous features in the DataFrame.
#It calculates statistics and information about continuous features

contin_feat_names = sales_data_ABT[sales_data_ABT.select_dtypes("nu
contin_feat_names
category_feat_names=sales_data_ABT[sales_data_ABT.select_dtypes(exc
category_feat_names

def continuous(df):
    cardinality = df.nunique()
    missing = df.isna().sum()/len(sales_data_ABT)*100
    missing_df=pd.DataFrame()
    missing_df['missing']=missing
    cardinality_df=pd.DataFrame()
    cardinality_df['cardinality']=cardinality
    continuous = df.describe(include=['number'])
    continuous = continuous.append(missing_df.T)
    continuous = continuous.append(cardinality_df.T)
    return continuous
continuous_feat_report=continuous(contin_feat_names)
continuous_feat_report
```

```
/var/folders/st/v8wvfcqn1h1g73vw8kxsbprh0000gn/T/ipykernel_22610/3
73572730.py:17: FutureWarning: The frame.append method is deprecat
ed and will be removed from pandas in a future version. Use panda
s.concat instead.
  continuous = continuous.append(missing_df.T)
/var/folders/st/v8wvfcqn1h1g73vw8kxsbprh0000gn/T/ipykernel_22610/3
73572730.py:18: FutureWarning: The frame.append method is deprecat
ed and will be removed from pandas in a future version. Use panda
s.concat instead.
  continuous = continuous.append(cardinality_df.T)
```

Out[10]:

	LotArea	TotalBsmtSF	YearBuilt	OverallCond	YrSold
count	1460.00	1460.00	1460.00	1460.00	1460.00
mean	10516.83	1057.43	1971.27	5.58	2007.82
std	9981.26	438.71	30.20	1.11	1.33
min	1300.00	0.00	1872.00	1.00	2006.00
25%	7553.50	795.75	1954.00	5.00	2007.00
50%	9478.50	991.50	1973.00	5.00	2008.00
75%	11601.50	1298.25	2000.00	6.00	2009.00

max	215245.00	6110.00	2010.00	9.00	2010.00
missing	0.00	0.00	0.00	0.00	0.00
cardinality	1073.00	721.00	112.00	9.00	5.00

Categorical feature Report

1. Mode - the most frequent value
2. 2nd mode - the second most frequent value
3. Frequency of mode
4. Proportion of mode in the dataset
5. Frequency of 2nd mode
6. Proportion of 2nd mode in the dataset
7. % missing values
8. Cardinality

In [11]:
```
#Generating a report for categorical features in the DataFrame.
#It calculates statistics and information about the categorical fea

mode1=-1
mode2=-1
mode1_freq=-1
mode2_freq=-1
def mode_calc(df):
    left=df.value_counts().items()
    x=0
    for i,j in left:
        if x==0:
            mode1=i
            mode1_freq=j

        elif x==1:
            mode2=i
            mode2_freq=j
            return mode1,mode2,mode1_freq,mode2_freq
        x+=1

m1={"mode":[]}
m2={"mode2":[]}
m1_freq={"mode_freq":[]}
m2_freq={"mode2_freq":[]}
```

```
m2_freq={"mode2_freq":[]}

for i in range(len(sales_data_ABT.select_dtypes(exclude="number").c
    mode1,mode2,mode1_freq,mode2_freq=mode_calc(sales_data_ABT[sale
    if len(m1["mode"])==0:
        m1["mode"]=[mode1]
    elif len(m1["mode"])>0:
        m1["mode"].append(mode1)

    if len(m2["mode2"])==0:
        m2["mode2"]=[mode2]
    elif len(m2["mode2"])>0:
        m2["mode2"].append(mode2)
```



```
    if len(m1_freq["mode_freq"])==0:
        m1_freq["mode_freq"]=[mode1_freq]
    elif len(m1_freq["mode_freq"])>0:
        m1_freq["mode_freq"].append(mode1_freq)

    if len(m2_freq["mode2_freq"])==0:
        m2_freq["mode2_freq"]=[mode2_freq]
    elif len(m2_freq["mode2_freq"])>0:
        m2_freq["mode2_freq"].append(mode2_freq)


m1=pd.DataFrame(m1)
```

```
m2=pd.DataFrame(m2)
m1_freq=pd.DataFrame(m1_freq)
m2_freq=pd.DataFrame(m2_freq)

m1_prop=m1_freq/len(sales_data_ABT)*100
m2_prop=m2_freq/len(sales_data_ABT)*100
missing_val=pd.DataFrame(category_feat_names.isna().sum()/len(sales
cardinality=pd.DataFrame(category_feat_names.nunique())
categorical_feat_report=pd.concat([m1,m2,m1_freq,m1_prop,m2_freq,m2
categorical_feat_report.set_axis(['Mode','2nd mode','Frequency of m
print(sales_data_ABT.select_dtypes(exclude="number").columns.to_lis
categorical_feat_report
categorical_feat_report["Index"]=sales_data_ABT.select_dtypes(exclu
categorical_feat_report.set_index("Index",inplace=True)
categorical_feat_report=pd.concat([categorical_feat_report,missing_
categorical_feat_report.set_axis(['Mode','2nd mode','Frequency of m
categorical_feat_report.T
```

```
['HouseStyle', 'Neighborhood', 'Condition1', 'SaleCondition']
```

Out[11]:

	HouseStyle	Neighborhood	Condition1	SaleCondition
Mode	7200.00	0.00	2006.00	5.00
2nd mode	9600.00	864.00	2005.00	6.00
Frequency of mode	25.00	37.00	67.00	821.00
Proportion of mode	1.71	2.53	4.59	56.23
Proportion of 2nd mode	24.00	35.00	64.00	252.00
Frequency of 2nd mode	1.64	2.40	4.38	17.26
% missing values	0.00	0.00	0.00	0.00
Cardinality	8.00	25.00	9.00	6.00

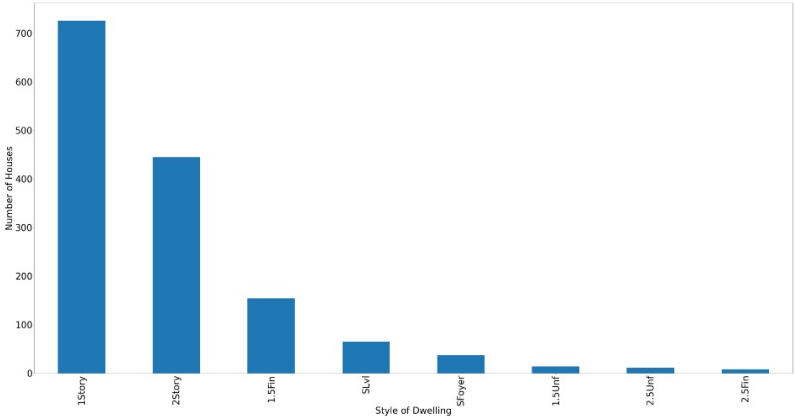
Feature Visualization

Categorical

In [12]:
```
#Plot bar graph for HouseStyle vs Number of houses
```

```
from matplotlib import pyplot as plt
plt.rcParams["figure.figsize"]=[60,30]
plt.rcParams["font.size"]=40
plt.xlabel("Style of Dwelling")
plt.ylabel("Number of Houses")
category_feat_names["HouseStyle"].value_counts().plot.bar()
```

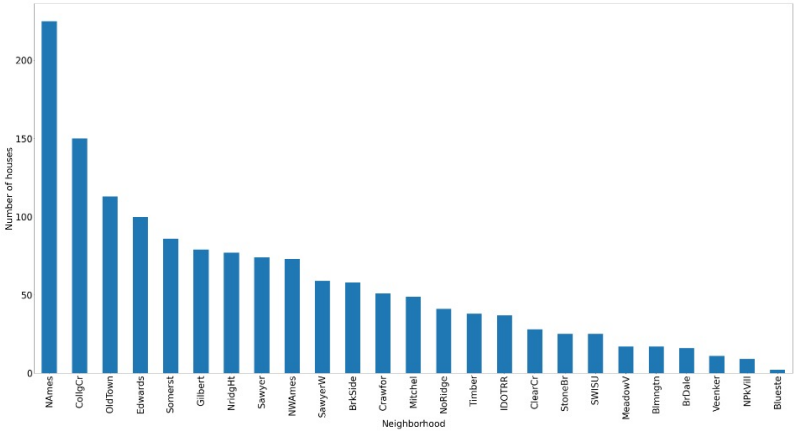
Out[12]: <AxesSubplot:xlabel='Style of Dwelling', ylabel='Number of Houses'
>



The bar plot displays the frequency of various house styles within a dataset. Every bar symbolizes a distinct housing type, and the number of houses in that style is indicated by the height of the bar. The most typical type of house is one story, followed by two stories. The "2.5 finished" level houses are the least prevalent of all the styles, with the others being far less common.

In [13]:
```
#Plot bar graph for Neighborhood vs Number of houses
plt.rcParams["figure.figsize"]=[60,30]
plt.rcParams["font.size"]=40
plt.xlabel("Neighborhood")
plt.ylabel("Number of houses")
category_feat_names["Neighborhood"].value_counts().plot.bar()
```

Out[13]: <AxesSubplot:xlabel='Neighborhood', ylabel='Number of houses'>

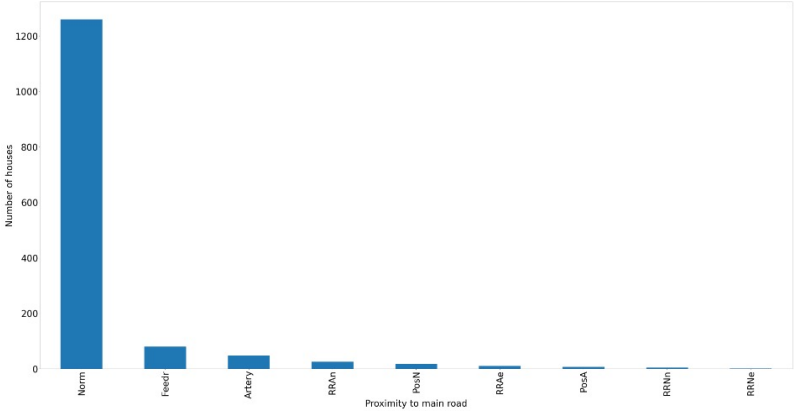


The distribution of houses throughout different neighborhoods is shown by the bar plot. With more than 200 homes, NAmes is the most populous neighborhood, followed by CollgCr and OldTown. There is a significant difference in the distribution of houses throughout each neighborhood as we move to the right of the plot, with neighborhoods such as Blueste having the fewest number of houses.

In [14]:
```
#Plot bar graph for Condition1 vs Number of houses
plt.rcParams["figure.figsize"]=[60,30]
plt.rcParams["font.size"]=40
plt.xlabel("Proximity to main road")
```

```
plt.ylabel("Number of houses")
category_feat_names["Condition1"].value_counts().plot.bar()
```

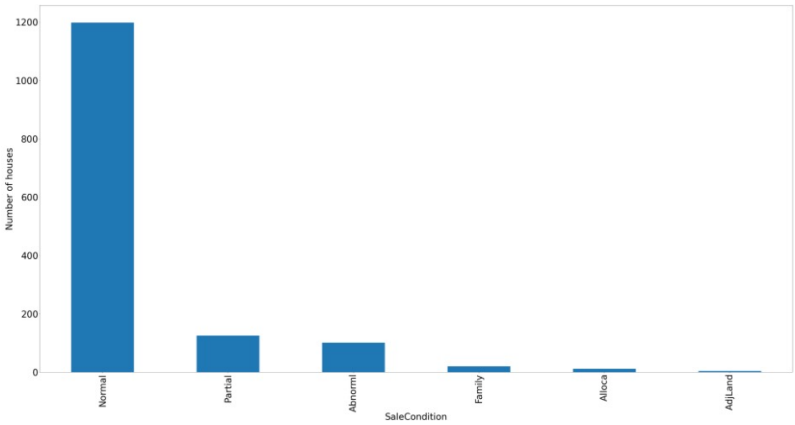
Out[14]: <AxesSubplot:xlabel='Proximity to main road', ylabel='Number of houses'>



The bar plot shows the number of houses categorized by their proximity to main roads or railroads. It shows that the majority of homes are classified as being in a typical vicinity to major roadways, with very few being close to railroads or major thoroughfares. Houses close to railroads, feeders, or arteries have substantially lower counts in these categories, indicating that they are not as frequent. Overall, the data indicates a distinct preference for homes situated in close proximity to standard roads.

In [15]:
```
#Plot bar graph for SaleCondition vs Number of houses
plt.rcParams["figure.figsize"]=[60,30]
plt.rcParams["font.size"]=40
plt.xlabel('SaleCondition')
plt.ylabel("Number of houses")
category_feat_names["SaleCondition"].value_counts().plot.bar()
```

Out[15]: <AxesSubplot:xlabel='SaleCondition', ylabel='Number of houses'>



The bar plot displays the frequency of different sale conditions for houses.The bar plot indicates that 'Normal' sales of houses accounted for the vast majority of sales. 'Partial' sales are less common and may indicate new homes or warranties. 'Normal' circumstances dominate the market, with 'Abnorml', 'Family', 'Alloca', and 'AdjLand' conditions accounting for only a minor portion of sales.

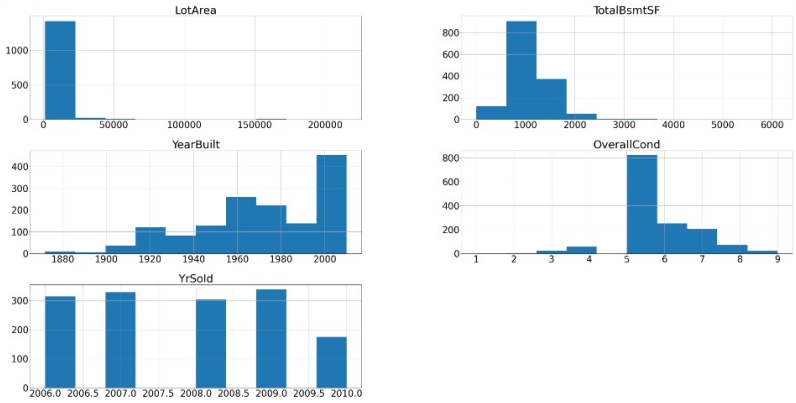
Histogram

In [16]:
```
contin_feat_names
```

Out[16]:

	LotArea	TotalBsmtSF	YearBuilt	OverallCond	YrSold
0	8450	856	2003	5	2008
1	9600	1262	1976	8	2007
2	11250	920	2001	5	2008
3	9550	756	1915	5	2006
4	14260	1145	2000	5	2008
5	14115	796	1993	5	2009
6	10084	1686	2004	5	2007
7	10382	1107	1973	6	2009
8	6120	952	1931	5	2008
9	7420	991	1939	6	2008
10	11200	1040	1965	5	2008

In [17]:
```
#Plot Histogram for continuous features
contin_feat_names.hist();
```



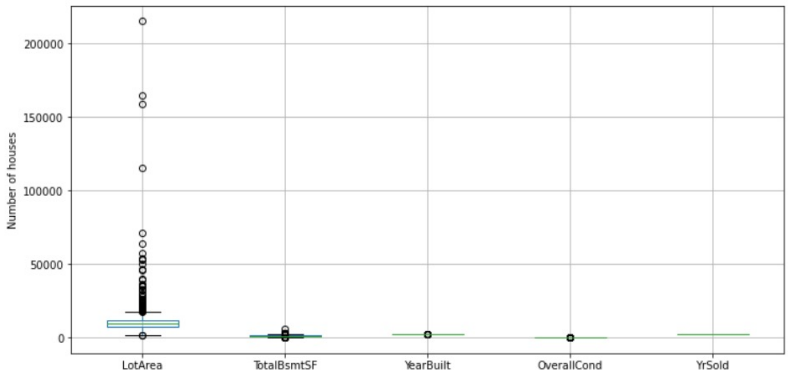
These histograms give us an idea of how different characteristics of houses are distributed:

1. LotArea: As lot size grows, there is a noticeable decline in the majority of lot areas. Very few homes are situated on very large lots.
2. TotalBsmtSF: For the majority of homes, the basement space is also on the smaller side. Though not many, there are some with bigger basements.
3. Year Built: Although homes were constructed in a range of years, there appear to have been two distinct times during which a disproportionate number of homes were constructed. The 1920s and the late 1990s and early 2000s are the starting points of the two periods.
4. OverallCond: A rating of five or lower indicates that most homes are in average condition. At the extremes, there aren't many homes.
5. YearSold: There has been a discernible decline in home sales in the past year, but overall, sales appear to be rather stable.

Boxplot [Continuous]

In [18]:
```
#Box plot for Continuous features
plt.rcParams["figure.figsize"] = [50, 20]
plt.figure(figsize=(12,6))
plt.rcParams["font.size"]=10
plt.ylabel('Number of houses')
contin_feat_names.boxplot()
```

Out[18]: <AxesSubplot:ylabel='Number of houses'>



1. Lot Area: This section of the plot displays the sizes of the yards associated with each property. There are a lot of outliers in the wide range of values displayed by this variable. The majority of the lot areas appear to be on the smaller side, with a tiny number of lots being noticeably larger, as indicated by the median's low comparison to the highest value.
2. TotalBsmtSF: This indicates how big a house's basement is. Compared to Lot Area, the boxplot for Total Basement Square Footage is substantially tighter, indicating less variability. A few of the upper end outliers indicate that some homes have remarkably huge basements. There is a bias towards smaller basement sizes as the median is nearer the first quartile than the third.
3. Year Built: The houses were built about this time. There are no outliers and very little variability in the plot. This could imply that the houses were constructed in a constrained amount of time.
4. OverallCond: This provides information on the general state of the houses. Plot has no outliers and is extremely tight. Given that the median is centered, the

Plot has no outliers and is extremely tight. Given that the median is centered, the data appears to be symmetrically distributed around a central value. This would suggest that the condition ratings of most houses are comparable.

5. YrSold: The lack of variability or outliers may indicate that the sales took place over a limited amount of time.

2. Identify data quality issues and build the data quality plan.

Missing Values

1. There are no missing values across all columns in the dataset.
2. The below code output shows there are no null values

In [19]:
```
#1460 non null values
sales_data_ABT.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1460 entries, 0 to 1459
Data columns (total 9 columns):
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   LotArea        1460 non-null   int64
 1   TotalBsmtSF    1460 non-null   int64
 2   YearBuilt      1460 non-null   int64
 3   OverallCond    1460 non-null   int64
 4   HouseStyle     1460 non-null   object
 5   Neighborhood   1460 non-null   object
 6   Condition1     1460 non-null   object
 7   YrSold         1460 non-null   int64
 8   SaleCondition  1460 non-null   object
dtypes: int64(5), object(4)
memory usage: 102.8+ KB
```

Irregular Cardinality

For Categorical Features:

1. HouseStyle has 8 unique values.
2. Neighborhood has 25 unique values.
3. Condition1 has 9 unique values.
4. SaleCondition has 6 unique values.

- As the number of unique values are less for each feature, the cardinality seems regular.

For Continuous Features:

1. LotArea has 1073 unique values.
2. TotalBsmtSF has 721 unique values.
3. YearBuilt has 112 unique values.
4. OverallCond has 9 unique values. This is a rating scale from 1 to 9. The number of unique values is equal to the scale range, which is normal.
5. YrSold has has 5 unique values. This indicates that the dataset covers sales from 5 different years, which is expected and not irregular.

In [20]:
```
#Cardinality
sales_data_ABT.nunique()
```

Out[20]:
```
LotArea          1073
TotalBsmtSF       721
YearBuilt         112
OverallCond         9
HouseStyle          8
Neighborhood       25
Condition1          9
YrSold              5
SaleCondition       6
dtype: int64
```

Outliers

1. LotArea: Outliers range from a minimum of 1,300 to a maximum of 215,245.

2. TotalBsmtSF: Outliers range from a minimum of 0 to a maximum of 6,110.
3. YearBuilt: There is an outlier value in the year 1880.
4. OverallCond: Outliers range from a minimum of 3 to a maximum of 9.
5. YrSold column does not have outliers based on the IQR method.

Data Quality Plan

Handling Missing Values:

Since there are no missing values in the dataset, no action is required in this area.

Addressing Irregular Cardinality:

The cardinalities of the categorical variables (HouseStyle, Neighborhood, Condition1, SaleCondition) are within expected ranges.

Removing Outliers

Remove the data points that are significantly different from others (outliers) in specific columns of a dataset to enhance its accuracy and reliability

Q3. [0.2] Data Preparation

1. Preprocess your data according to the data quality plan.

In [21]:
```
#Calculate the Inter Quartile Range
Q1 = sales_data_ABT["LotArea"].quantile(0.25)
Q3 = sales_data_ABT["LotArea"].quantile(0.75)

IQR = Q3-Q1
print(f"IQR = {Q3} - {Q1} = {IQR}")
```

```
IQR = 11601.5 - 7553.5 = 4048.0
```

In [22]:
```
#Calculate number of outliers
outliers_df = sales_data_ABT[(sales_data_ABT["LotArea"] < (Q1 - 1.5
print("Num of outliers: ", len(outliers_df))
```

```
Num of outliers:  69
```

In [23]:
```
sales_data_ABT
```

125	6780	520	1935	8	1.5Fin	IDOTRR	Norm
126	4928	1078	1976	5	1Story	NPkVill	Norm
127	4388	672	1930	7	1.5Unf	OldTown	Feedr
128	7590	660	1966	5	2Story	NAmes	PosN
129	8973	1008	1958	7	1Story	NAmes	Norm
130	14200	924	1966	6	2Story	NAmes	Norm
131	12224	992	2000	5	2Story	Gilbert	Norm
132	7388	1063	1959	6	1Story	NAmes	Norm
133	6853	1267	2001	5	1Story	Timber	Norm
134	10335	1461	1968	6	1Story	Sawyer	Norm
135	10400	1304	1970	6	1Story	NWAmes	Norm
136	10355	1214	1967	5	1Story	NAmes	Norm

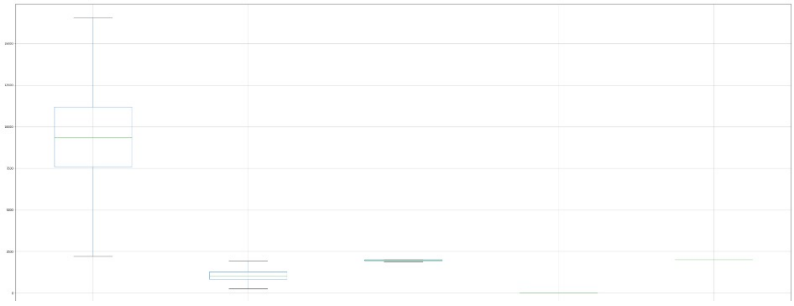
In [24]:
```
import pandas as pd


def remove_outliers_iteratively(df, column):
    outlier_indices = []
    print(column)
    while True:
        Q1 = df[column].quantile(0.25)
        Q3 = df[column].quantile(0.75)
        IQR = Q3 - Q1
        current_outliers = df[(df[column] < (Q1 - 1.5 * IQR)) | (df

        # If no more outliers, break the loop
        if len(current_outliers) == 0:
            break

        # Append current outliers to the list
        outlier_indices.extend(current_outliers)

        # Drop current outliers
        df = df.drop(current_outliers)

    # Return dataframe without outliers and the indices of all outl
    return df, outlier_indices

for j in range(4):
    for i in contin_feat_names.columns:
        sales_data_ABT, removed_outliers_indices = remove_outliers_
```

```
LotArea
TotalBsmtSF
YearBuilt
OverallCond
YrSold
LotArea
TotalBsmtSF
YearBuilt
OverallCond
```

```
OverallCond
YrSold
LotArea
TotalBsmtSF
YearBuilt
OverallCond
YrSold
LotArea
TotalBsmtSF
YearBuilt
OverallCond
YrSold
```


In [25]:
```
sales_data_ABT.boxplot()
```

Out[25]:
```
<AxesSubplot:>
```



After removing the outliers, the above boxplot represents with it.

Q4. [0.2] Data Insights

1. Build the correlation heatmap of the features in the ABT. Derive insights from it and relate it to the business problem being addressed.

   i. What are the descriptive features that highly correlate with the target feature? Propose hypotheses to explain the correlation.

   ii. What are the domain concepts that highly correlate with each other? Propose hypotheses explaining the correlation.

   iii. Are there any features that are useless for a predictive model?

In [26]:
```
new_data=sales_data_ABT.copy()
```

In [27]:
```
new_data["SalePrice"]=sales_data["SalePrice"]
```

In [28]:
```
new_data.to_csv("present_sales_data.csv")
```

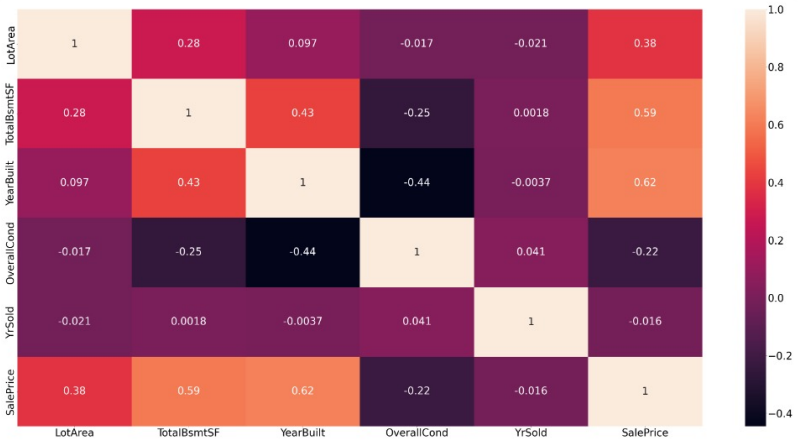

In [29]:
```
#Calculate the correlation matrix
correlation_matrix = new_data.corr()
correlation_matrix
```

Out[29]:

	LotArea	TotalBsmtSF	YearBuilt	OverallCond	YrSold	SalePrice
LotArea	1.00	0.28	0.10	-0.02	-0.02	0.38
TotalBsmtSF	0.28	1.00	0.43	-0.25	0.00	0.59
YearBuilt	0.10	0.43	1.00	-0.44	-0.00	0.62
OverallCond	-0.02	-0.25	-0.44	1.00	0.04	-0.22
YrSold	-0.02	0.00	-0.00	0.04	1.00	-0.02

SalePrice	0.38	0.59	0.62	-0.22	-0.02	1.00

In [85]:
```
# plotting the heatmap
hm = sns.heatmap(data=correlation_matrix,
                 annot=True)
# displaying the plotted heatmap
plt.show()
```



i. What are the descriptive features that highly correlate with the target feature? Propose hypotheses to explain the correlation.

Descriptive Features Highly Correlating with the Target Feature (Sale Price)

LotArea (0.38):

According to this positive correlation, larger lots typically result in higher sale prices. This could be because larger homes have more room for future additions, gardens, and privacy.

TotalBsmtSF (0.59):

A high positive association suggests that larger basements increase the value of a property, because they provide more room for storage.

YearBuilt (0.62):

Because of their contemporary style, upgraded facilities, and lesser expected maintenance, newer homes tend to sell for more money.

ii. What are the domain concepts that highly correlate with each other? Propose hypotheses explaining the correlation.

Domain Concepts Highly Correlating with Each Other

The association between TotalBsmtSF and YearBuilt (0.43) points to a possible trend in building practices throughout time, with newer homes likely having larger basements, maybe reflecting contemporary demands for more living space below ground. The negative association between YearBuilt and OverallCond (-0.44) suggests that older homes may be in worse overall condition than newer ones, which is to be expected given their age and wear over time.

iii. Are there any features that are useless for a predictive model?

Features Useless for a Predictive Model

YrSold (-0.016 with Sale Price):

There is no discernible correlation between the year of sale and the sale price, indicating that the year of sale may not be a reliable indicator on its own.

OverallCond (-0.22 with Sale Price):

A house's overall condition has a reasonably modest negative correlation (but not zero) with Sale Price. This may be due to the fact that purchasers prioritize other characteristics over condition, such as size and age, or it may be the case that "OverallCond" is a subjective metric with broad interpretation ranges.

Business Problem Insights

In the context of real estate sales, the insights from this heatmap can inform several business strategies, such as:

Prioritizing Features in Listings:*

In marketing materials, it could be more effective to highlight the year of construction and the square footage.

Making Investment Decisions:

Newer structures with larger basements may be preferred by investors searching for properties with potential for appreciation in value.

Renovation Focus:

By aligning with aspects that favorably influence sale values, renovations that increase basement space or improve general condition may provide a higher return on investment for older homes.

In [ ]:

S&D a-Adequate 3

Func b-Adequate 3.5

2

Improve Clarity: Use mark-down appropriately and con-sistently.

http://localhost:8888/notebooks/Downloads/Gajjala_Jahnavi_Kasibhatla_Venkata_Sreeniv

Page 26 of 26

Q2

0 / 0

Submit here your Python notebook (ipynb) as an attachment

Gajjala-Jahnavi-Kasibhatla-Venkata Sreenivas Prasad-Assignment-1.ipynb