In [ ]:

Type *Markdown* and LaTeX: α^2

In [ ]:

In [12]:
```
# importing libraries
import pandas as pd # data science essentials
import matplotlib.pyplot as plt # data visualization
import seaborn as sns # enhanced data visualization
import statsmodels.formula.api as smf # regression modeling
from sklearn.model_selection import train_test_split # train/test split
import sklearn.linear_model # linear modeling in scikit-learn

# setting pandas print options
pd.set_option('display.max_rows', 500)
pd.set_option('display.max_columns', 500)
pd.set_option('display.width', 1000)


# specifying the path and file name
file = 'Ames Housing Dataset.xlsx'


# reading the file into Python
df = pd.read_excel(file)


# checking the file
df.head(n = 5)
```

Out[12]:

	Order	Lot_Area	Street	Lot_Config	Neighborhood	Overall_Qual	Overall_Cond	Mas_Vnr_Area	Total_Bsmt_SF	First_Flr_SF	Second_Flr_SF	Gr_Liv_Are
0	1	31770	Pave	Corner	NAmes	6	5	112.0	1080.0	1656	0	165
1	2	11622	Pave	Inside	NAmes	5	6	0.0	882.0	896	0	89
2	3	14267	Pave	Corner	NAmes	6	6	108.0	1329.0	1329	0	132
3	4	11160	Pave	Corner	NAmes	7	5	0.0	2110.0	2110	0	211
4	5	13830	Pave	Inside	Gilbert	5	5	0.0	928.0	928	701	162

In [18]:
```
df.isna().sum()
```

Out[18]:
```
Order             0
Lot_Area          0
Street            0
Lot_Config        0
Neighborhood      0
Overall_Qual      0
Overall_Cond      0
Mas_Vnr_Area     23
Total_Bsmt_SF     1
First_Flr_SF      0
Second_Flr_SF     0
Gr_Liv_Area       0
Full_Bath         0
Half_Bath         0
Kitchen_AbvGr     0
TotRms_AbvGr      0
Fireplaces        0
Garage_Cars       1
Garage_Area       1
Porch_Area        0
Pool_Area         0
Sale_Price        0
dtype: int64
```

In [16]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2930 entries, 0 to 2929
Data columns (total 22 columns):
 #   Column         Non-Null Count  Dtype  
---  ------         --------------  -----  
 0   Order          2930 non-null   int64  
 1   Lot_Area       2930 non-null   int64  
 2   Street         2930 non-null   object 
 3   Lot_Config     2930 non-null   object 
 4   Neighborhood   2930 non-null   object 
 5   Overall_Qual   2930 non-null   int64  
 6   Overall_Cond   2930 non-null   int64  
 7   Mas_Vnr_Area   2907 non-null   float64
 8   Total_Bsmt_SF  2929 non-null   float64
 9   First_Flr_SF   2930 non-null   int64  
 10  Second_Flr_SF  2930 non-null   int64  
 11  Gr_Liv_Area    2930 non-null   int64  
 12  Full_Bath      2930 non-null   int64  
 13  Half_Bath      2930 non-null   int64  
 14  Kitchen_AbvGr  2930 non-null   int64  
 15  TotRms_AbvGr   2930 non-null   int64  
 16  Fireplaces     2930 non-null   int64  
 17  Garage_Cars    2929 non-null   float64
 18  Garage_Area    2929 non-null   float64
 19  Porch_Area     2930 non-null   int64  
 20  Pool_Area      2930 non-null   int64  
 21  Sale_Price     2930 non-null   int64  
dtypes: float64(4), int64(15), object(3)
memory usage: 503.7+ KB
```

In [13]:
```
df.describe()
```

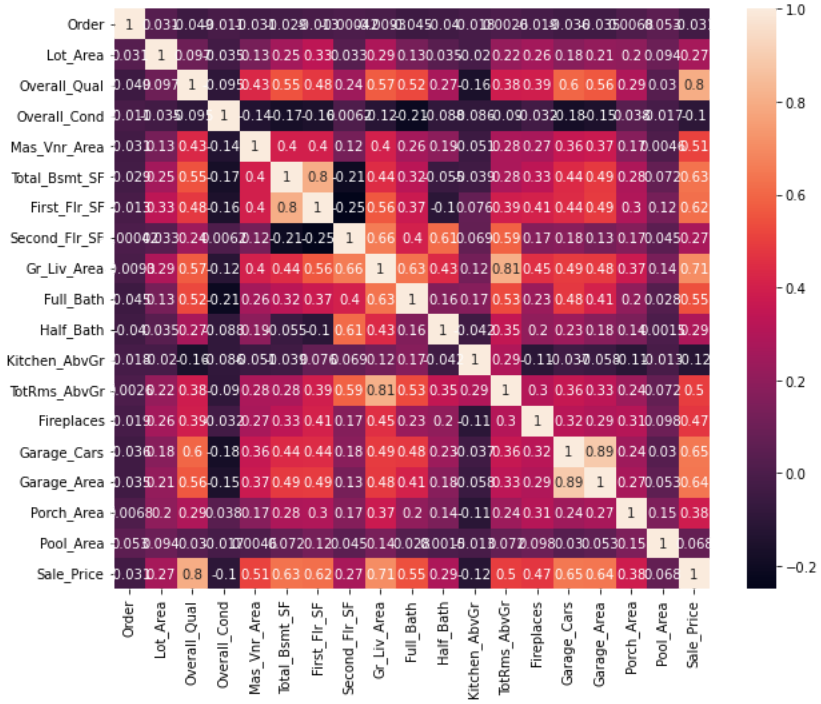
Out[13]:

	Order	Lot_Area	Overall_Qual	Overall_Cond	Mas_Vnr_Area	Total_Bsmt_SF	First_Flr_SF	Second_Flr_SF	Gr_Liv_Area	Full_Bath	H
count	2930.00000	2930.000000	2930.000000	2930.000000	2907.000000	2929.000000	2930.000000	2930.000000	2930.000000	2930.000000	293
mean	1465.50000	10147.921843	6.094881	5.563140	101.896801	1051.614544	1159.557679	335.455973	1499.690444	1.566553	
std	845.96247	7880.017759	1.411026	1.111537	179.112611	440.615067	391.890885	428.395715	505.508887	0.552941	
min	1.00000	1300.000000	1.000000	1.000000	0.000000	0.000000	334.000000	0.000000	334.000000	0.000000	
25%	733.25000	7440.250000	5.000000	5.000000	0.000000	793.000000	876.250000	0.000000	1126.000000	1.000000	
50%	1465.50000	9436.500000	6.000000	5.000000	0.000000	990.000000	1084.000000	0.000000	1442.000000	2.000000	
75%	2197.75000	11555.250000	7.000000	6.000000	164.000000	1302.000000	1384.000000	703.750000	1742.750000	2.000000	
max	2930.00000	215245.000000	10.000000	9.000000	1600.000000	6110.000000	5095.000000	2065.000000	5642.000000	4.000000	

In [19]:
```
df.columns
```

Out[19]:
```
Index(['Order', 'Lot_Area', 'Street', 'Lot_Config', 'Neighborhood', 'Overall_Qual', 'Overall_Cond', 'Mas_Vnr_Area', 'Total_B
smt_SF', 'First_Flr_SF', 'Second_Flr_SF', 'Gr_Liv_Area', 'Full_Bath', 'Half_Bath', 'Kitchen_AbvGr', 'TotRms_AbvGr', 'Firepla
ces', 'Garage_Cars', 'Garage_Area', 'Porch_Area', 'Pool_Area', 'Sale_Price'], dtype='object')
```

In [17]:
```
#plot heatmap
plt.figure(figsize=(10,8))
sns.heatmap(df.corr(),annot=True)
plt.show()
```



In [ ]:

Step 1: Hypothesize on Features to Engineer

Write a 2-3 sentence hypothesis on the effect you believe each new feature will have on the response variables (*Sale_Price* AND *log_Sale_Price*) in a markdown cell.

New Feature 1

Neighbourhood cannot be handled as it is, because it is an object. However, there are so many unique values that it is not a good idea to do one hot encodign. Therefore, here we create a new feature, the average house price for each Neighbourhood. As location is a major determinant of house prices, it will be correlated with the response variable.

In [ ]:

In [ ]:

Step 2: Code the New Features

Use the code cell below to develop the features you have hypothesized.

In [45]:
```
#New Feature 1
```

In [40]:
```
mapping = df.groupby('Neighborhood')['Sale_Price'].mean().to_dict()
mapping
```

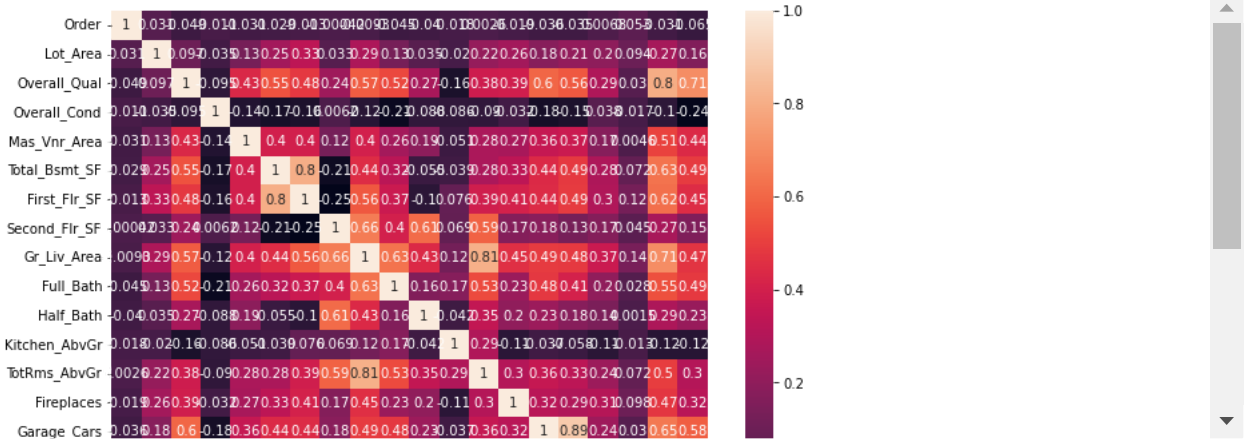
Out[40]:
```
{'Blmngtn': 196661.67857142858,
 'Blueste': 143590.0,
 'BrDale': 105608.33333333333,
 'BrkSide': 124756.25,
 'ClearCr': 208662.0909090909,
 'CollgCr': 201803.43445692884,
 'Crawfor': 207550.83495145632,
 'Edwards': 130843.38144329897,
 'Gilbert': 190646.57575757575,
 'Greens': 193531.25,
 'GrnHill': 280000.0,
 'IDOTRR': 103752.90322580645,
 'Landmrk': 137000.0,
 'MeadowV': 95756.48648648648,
 'Mitchel': 162226.63157894736,
 'NAmes': 145097.34988713317,
 'NPkVill': 140710.86956521738,
 'NWAmes': 188406.90839694656,
 'NoRidge': 330319.1267605634,
 'NridgHt': 322018.265060241,
 'OldTown': 123991.89121338913,
 'SWISU': 135071.9375,
 'Sawyer': 136751.1523178808,
 'SawyerW': 184070.184,
 'Somerst': 229707.32417582418,
 'StoneBr': 324229.1960784314,
 'Timber': 246599.54166666666,
 'Veenker': 248314.58333333334}
```

In [42]:
```
df['Neighborhood_New'] = df['Neighborhood'].map(mapping)
```

Step 3: Check the Results

Develop a correlation matrix or a heatmap to show the linear relationships between your five new features and the response variables (*Sale_Price* AND *log_Sale_Price*).

In [48]:
```
#plot heatmap
plt.figure(figsize=(10,8))
sns.heatmap(df.corr(),annot=True)
plt.show()
```



In [ ]:
```
```

Step 4: Explain the Results

Did your engineered features have the effect that was expected? Explain in 1-2 sentences for each engineered feature.

Neighborhood_New and Sale Price

As expected , these two new features are correlated(r=0.76).When actually building machine learning models, you need to worry about over fitting, but that is out of scope, so this is fine here.

In [ ]: