In [1]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
%matplotlib inline
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
```

In [2]:
```
df=pd.read_csv('HousePrices.csv')
```

In [3]:
```
df.head(10)
```

Out[3]:

	Id	MSSubClass	MSZoning	LotFrontage	LotArea	Street	Alley	LotShape	LandContour	Utilities	...	PoolArea	PoolQC	Fence	MiscFeatur
0	1	60	RL	65.0	8450	Pave	NaN	Reg	Lvl	AllPub	...	0	NaN	NaN	Nal
1	2	20	RL	80.0	9600	Pave	NaN	Reg	Lvl	AllPub	...	0	NaN	NaN	Nal
2	3	60	RL	68.0	11250	Pave	NaN	IR1	Lvl	AllPub	...	0	NaN	NaN	Nal
3	4	70	RL	60.0	9550	Pave	NaN	IR1	Lvl	AllPub	...	0	NaN	NaN	Nal
4	5	60	RL	84.0	14260	Pave	NaN	IR1	Lvl	AllPub	...	0	NaN	NaN	Nal
5	6	50	RL	85.0	14115	Pave	NaN	IR1	Lvl	AllPub	...	0	NaN	MnPrv	She
6	7	20	RL	75.0	10084	Pave	NaN	Reg	Lvl	AllPub	...	0	NaN	NaN	Nal
7	8	60	RL	NaN	10382	Pave	NaN	IR1	Lvl	AllPub	...	0	NaN	NaN	She
8	9	50	RM	51.0	6120	Pave	NaN	Reg	Lvl	AllPub	...	0	NaN	NaN	Nal
9	10	190	RL	50.0	7420	Pave	NaN	Reg	Lvl	AllPub	...	0	NaN	NaN	Nal

10 rows × 81 columns

In [4]:
```
df.tail()
```

Out[4]:

	Id	MSSubClass	MSZoning	LotFrontage	LotArea	Street	Alley	LotShape	LandContour	Utilities	...	PoolArea	PoolQC	Fence	Miscl
1455	1456	60	RL	62.0	7917	Pave	NaN	Reg	Lvl	AllPub	...	0	NaN	NaN	
1456	1457	20	RL	85.0	13175	Pave	NaN	Reg	Lvl	AllPub	...	0	NaN	MnPrv	
1457	1458	70	RL	66.0	9042	Pave	NaN	Reg	Lvl	AllPub	...	0	NaN	GdPrv	
1458	1459	20	RL	68.0	9717	Pave	NaN	Reg	Lvl	AllPub	...	0	NaN	NaN	
1459	1460	20	RL	75.0	9937	Pave	NaN	Reg	Lvl	AllPub	...	0	NaN	NaN	

5 rows × 81 columns

In [5]:
```
df.shape
```

Out[5]:
```
(1460, 81)
```

In [6]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1460 entries, 0 to 1459
Data columns (total 81 columns):
 #   Column         Non-Null Count  Dtype  
---  ------         --------------  -----  
 0   Id             1460 non-null   int64  
 1   MSSubClass     1460 non-null   int64  
 2   MSZoning       1460 non-null   object 
 3   LotFrontage    1201 non-null   float64
 4   LotArea        1460 non-null   int64  
 5   Street         1460 non-null   object 
 6   Alley          91 non-null     object 
 7   LotShape       1460 non-null   object 
 8   LandContour    1460 non-null   object 
 9   Utilities      1460 non-null   object 
 10  LotConfig      1460 non-null   object 
 11  LandSlope      1460 non-null   object 
 12  Neighborhood   1460 non-null   object 
 13  Condition1     1460 non-null   object 
 14  Condition2     1460 non-null   object 
```

```
 15  BldgType       1460 non-null   object
 16  HouseStyle     1460 non-null   object
 17  OverallQual    1460 non-null   int64
 18  OverallCond    1460 non-null   int64
 19  YearBuilt      1460 non-null   int64
 20  YearRemodAdd   1460 non-null   int64
 21  RoofStyle      1460 non-null   object
 22  RoofMatl       1460 non-null   object
 23  Exterior1st    1460 non-null   object
 24  Exterior2nd    1460 non-null   object
 25  MasVnrType     1452 non-null   object
 26  MasVnrArea     1452 non-null   float64
 27  ExterQual      1460 non-null   object
 28  ExterCond      1460 non-null   object
 29  Foundation     1460 non-null   object
 30  BsmtQual       1423 non-null   object
 31  BsmtCond       1423 non-null   object
 32  BsmtExposure   1422 non-null   object
 33  BsmtFinType1   1423 non-null   object
 34  BsmtFinSF1     1460 non-null   int64
 35  BsmtFinType2   1422 non-null   object
 36  BsmtFinSF2     1460 non-null   int64
 37  BsmtUnfSF      1460 non-null   int64
 38  TotalBsmtSF    1460 non-null   int64
 39  Heating        1460 non-null   object
 40  HeatingQC      1460 non-null   object
 41  CentralAir     1460 non-null   object
 42  Electrical     1459 non-null   object
 43  1stFlrSF       1460 non-null   int64
 44  2ndFlrSF       1460 non-null   int64
 45  LowQualFinSF   1460 non-null   int64
 46  GrLivArea      1460 non-null   int64
 47  BsmtFullBath   1460 non-null   int64
 48  BsmtHalfBath   1460 non-null   int64
 49  FullBath       1460 non-null   int64
 50  HalfBath       1460 non-null   int64
 51  BedroomAbvGr   1460 non-null   int64
 52  KitchenAbvGr   1460 non-null   int64
 53  KitchenQual    1460 non-null   object
 54  TotRmsAbvGrd   1460 non-null   int64
 55  Functional     1460 non-null   object
 56  Fireplaces     1460 non-null   int64
 57  FireplaceQu    770 non-null    object
 58  GarageType     1379 non-null   object
 59  GarageYrBlt    1379 non-null   float64
 60  GarageFinish   1379 non-null   object
 61  GarageCars     1460 non-null   int64
 62  GarageArea     1460 non-null   int64
 63  GarageQual     1379 non-null   object
 64  GarageCond     1379 non-null   object
 65  PavedDrive     1460 non-null   object
 66  WoodDeckSF     1460 non-null   int64
 67  OpenPorchSF    1460 non-null   int64
 68  EnclosedPorch  1460 non-null   int64
 69  3SsnPorch      1460 non-null   int64
 70  ScreenPorch    1460 non-null   int64
 71  PoolArea       1460 non-null   int64
 72  PoolQC         7 non-null      object
 73  Fence          281 non-null    object
 74  MiscFeature    54 non-null     object
 75  MiscVal        1460 non-null   int64
 76  MoSold         1460 non-null   int64
 77  YrSold         1460 non-null   int64
 78  SaleType       1460 non-null   object
 79  SaleCondition  1460 non-null   object
 80  SalePrice      1460 non-null   int64
dtypes: float64(3), int64(35), object(43)
memory usage: 924.0+ KB
```

In [7]:

```
df.describe()
```

Out[7]:

	Id	MSSubClass	LotFrontage	LotArea	OverallQual	OverallCond	YearBuilt	YearRemodAdd	MasVnrArea	BsmtFinSF
count	1460.000000	1460.000000	1201.000000	1460.000000	1460.000000	1460.000000	1460.000000	1460.000000	1452.000000	1460.000000
mean	730.500000	56.897260	70.049958	10516.828082	6.099315	5.575342	1971.267808	1984.865753	103.685262	443.639726
std	421.610009	42.300571	24.284752	9981.264932	1.382997	1.112799	30.202904	20.645407	181.066207	456.098091
min	1.000000	20.000000	21.000000	1300.000000	1.000000	1.000000	1872.000000	1950.000000	0.000000	0.000000
25%	365.750000	20.000000	59.000000	7553.500000	5.000000	5.000000	1954.000000	1967.000000	0.000000	0.000000
50%	730.500000	50.000000	69.000000	9478.500000	6.000000	5.000000	1973.000000	1994.000000	0.000000	383.500000
75%	1095.250000	70.000000	80.000000	11601.500000	7.000000	6.000000	2000.000000	2004.000000	166.000000	712.250000
max	1460.000000	190.000000	313.000000	215245.000000	10.000000	9.000000	2010.000000	2010.000000	1600.000000	5644.000000

8 rows × 38 columns

In [8]:
```
df.dtypes
```

Out[8]:
```
Id                int64
MSSubClass        int64
MSZoning         object
LotFrontage     float64
LotArea           int64
                 ...
MoSold            int64
YrSold            int64
SaleType         object
SaleCondition    object
SalePrice         int64
Length: 81, dtype: object
```

In [9]:
```
df.columns
```

Out[9]:
```
Index(['Id', 'MSSubClass', 'MSZoning', 'LotFrontage', 'LotArea', 'Street',
       'Alley', 'LotShape', 'LandContour', 'Utilities', 'LotConfig',
       'LandSlope', 'Neighborhood', 'Condition1', 'Condition2', 'BldgType',
       'HouseStyle', 'OverallQual', 'OverallCond', 'YearBuilt', 'YearRemodAdd',
       'RoofStyle', 'RoofMatl', 'Exterior1st', 'Exterior2nd', 'MasVnrType',
       'MasVnrArea', 'ExterQual', 'ExterCond', 'Foundation', 'BsmtQual',
       'BsmtCond', 'BsmtExposure', 'BsmtFinType1', 'BsmtFinSF1',
       'BsmtFinType2', 'BsmtFinSF2', 'BsmtUnfSF', 'TotalBsmtSF', 'Heating',
       'HeatingQC', 'CentralAir', 'Electrical', '1stFlrSF', '2ndFlrSF',
       'LowQualFinSF', 'GrLivArea', 'BsmtFullBath', 'BsmtHalfBath', 'FullBath',
       'HalfBath', 'BedroomAbvGr', 'KitchenAbvGr', 'KitchenQual',
       'TotRmsAbvGrd', 'Functional', 'Fireplaces', 'FireplaceQu', 'GarageType',
       'GarageYrBlt', 'GarageFinish', 'GarageCars', 'GarageArea', 'GarageQual',
       'GarageCond', 'PavedDrive', 'WoodDeckSF', 'OpenPorchSF',
       'EnclosedPorch', '3SsnPorch', 'ScreenPorch', 'PoolArea', 'PoolQC',
       'Fence', 'MiscFeature', 'MiscVal', 'MoSold', 'YrSold', 'SaleType',
       'SaleCondition', 'SalePrice'],
      dtype='object')
```

In [10]:
```
df.sample(5)
```

Out[10]:

	Id	MSSubClass	MSZoning	LotFrontage	LotArea	Street	Alley	LotShape	LandContour	Utilities	...	PoolArea	PoolQC	Fence	Miscl
1083	1084	20	RL	80.0	8800	Pave	NaN	Reg	Lvl	AllPub	...	0	NaN	MnPrv	
1056	1057	120	RL	43.0	7052	Pave	NaN	IR1	Lvl	AllPub	...	0	NaN	NaN	
923	924	120	RL	50.0	8012	Pave	NaN	Reg	Lvl	AllPub	...	0	NaN	NaN	
132	133	20	RL	75.0	7388	Pave	NaN	Reg	Lvl	AllPub	...	0	NaN	NaN	
714	715	60	RL	NaN	13517	Pave	NaN	IR1	Lvl	AllPub	...	0	NaN	NaN	

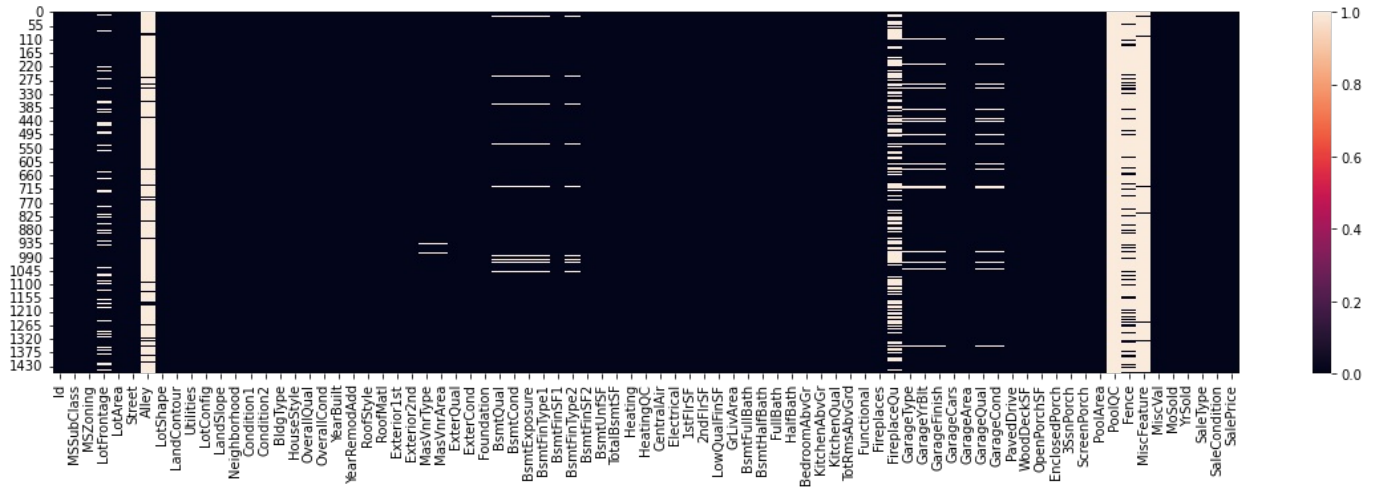
5 rows × 81 columns

In [11]:
```
df.isnull().sum()
```

Out[11]:
```
Id                0
MSSubClass        0
MSZoning          0
LotFrontage     259
LotArea           0
                 ...
MoSold            0
YrSold            0
SaleType          0
SaleCondition     0
SalePrice         0
Length: 81, dtype: int64
```

In [12]:
```
plt.figure(figsize=(20, 5))
sns.heatmap(df.isnull())
```

Out[12]:
```
<AxesSubplot:>
```



In [13]:
```
df.describe(include='object')
```

Out[13]:

	MSZoning	Street	Alley	LotShape	LandContour	Utilities	LotConfig	LandSlope	Neighborhood	Condition1	...	GarageType	GarageFi
count	1460	1460	91	1460	1460	1460	1460	1460	1460	1460	...	1379	1
unique	5	2	2	4	4	2	5	3	25	9	...	6	
top	RL	Pave	Grvl	Reg	Lvl	AllPub	Inside	Gtl	NAmes	Norm	...	Attchd	
freq	1151	1454	50	925	1311	1459	1052	1382	225	1260	...	870	

4 rows × 43 columns

In [14]:
```
nullcols = df.isnull().sum()
nullcols = nullcols[nullcols>0]
nullcols
```

Out[14]:
```
LotFrontage      259
Alley           1369
MasVnrType         8
MasVnrArea         8
BsmtQual          37
BsmtCond          37
BsmtExposure      38
BsmtFinType1      37
BsmtFinType2      38
Electrical         1
FireplaceQu      690
GarageType        81
GarageYrBlt       81
GarageFinish      81
GarageQual        81
GarageCond        81
PoolQC          1453
Fence           1179
MiscFeature     1406
dtype: int64
```

In [15]:
```
df=df.drop(['PoolArea','LotShape','Utilities','LotConfig','LandContour','Fence','FireplaceQu','YearRemodAdd', 'Ex
```

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

Out[16]:

	Id	MSSubClass	MSZoning	LotFrontage	LotArea	LandSlope	Neighborhood	Condition1	BldgType	HouseStyle	...	OpenPorchSF	Enclosed
0	1	60	RL	65.0	8450	Gtl	CollgCr	Norm	1Fam	2Story	...	61	
1	2	20	RL	80.0	9600	Gtl	Veenker	Feedr	1Fam	1Story	...	0	
2	3	60	RL	68.0	11250	Gtl	CollgCr	Norm	1Fam	2Story	...	42	
3	4	70	RL	60.0	9550	Gtl	Crawfor	Norm	1Fam	2Story	...	35	
4	5	60	RL	84.0	14260	Gtl	NoRidge	Norm	1Fam	2Story	...	84	

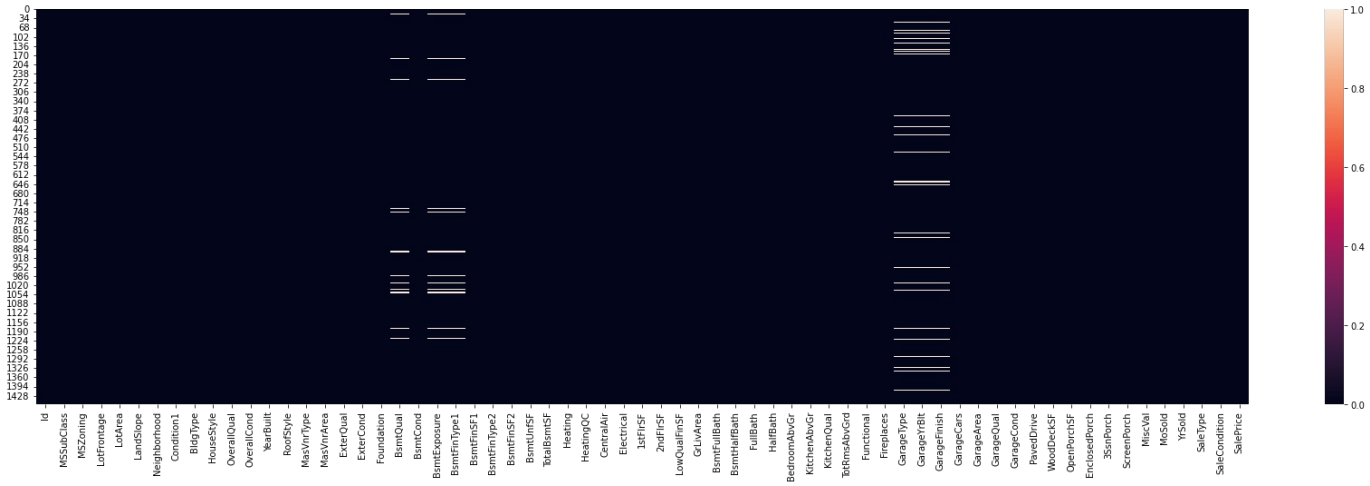
5 rows × 65 columns

In [17]:
```
df['LotFrontage'] = df['LotFrontage'].fillna(df['LotFrontage'].mean())
df['BsmtCond'] = df['BsmtCond'].fillna(df['BsmtCond'].mode()[0])
df['BsmtFinType2'] = df['BsmtFinType2'].fillna(df['BsmtFinType2'].mode()[0])
df['GarageQual'] = df['GarageQual'].fillna(df['GarageQual'].mode()[0])
df['GarageCond'] = df['GarageCond'].fillna(df['GarageCond'].mode()[0])
df['MasVnrArea'] = df['MasVnrArea'].fillna(0)
df['MasVnrType'] = df['MasVnrType'].fillna(df['MasVnrType'].mode()[0])
df['Electrical'] = df['Electrical'].fillna(df['Electrical'].mode()[0])
```

DATA VISUALIZATION

In [18]:
```
plt.figure(figsize=(30, 8))
sns.heatmap(df.isnull())
```

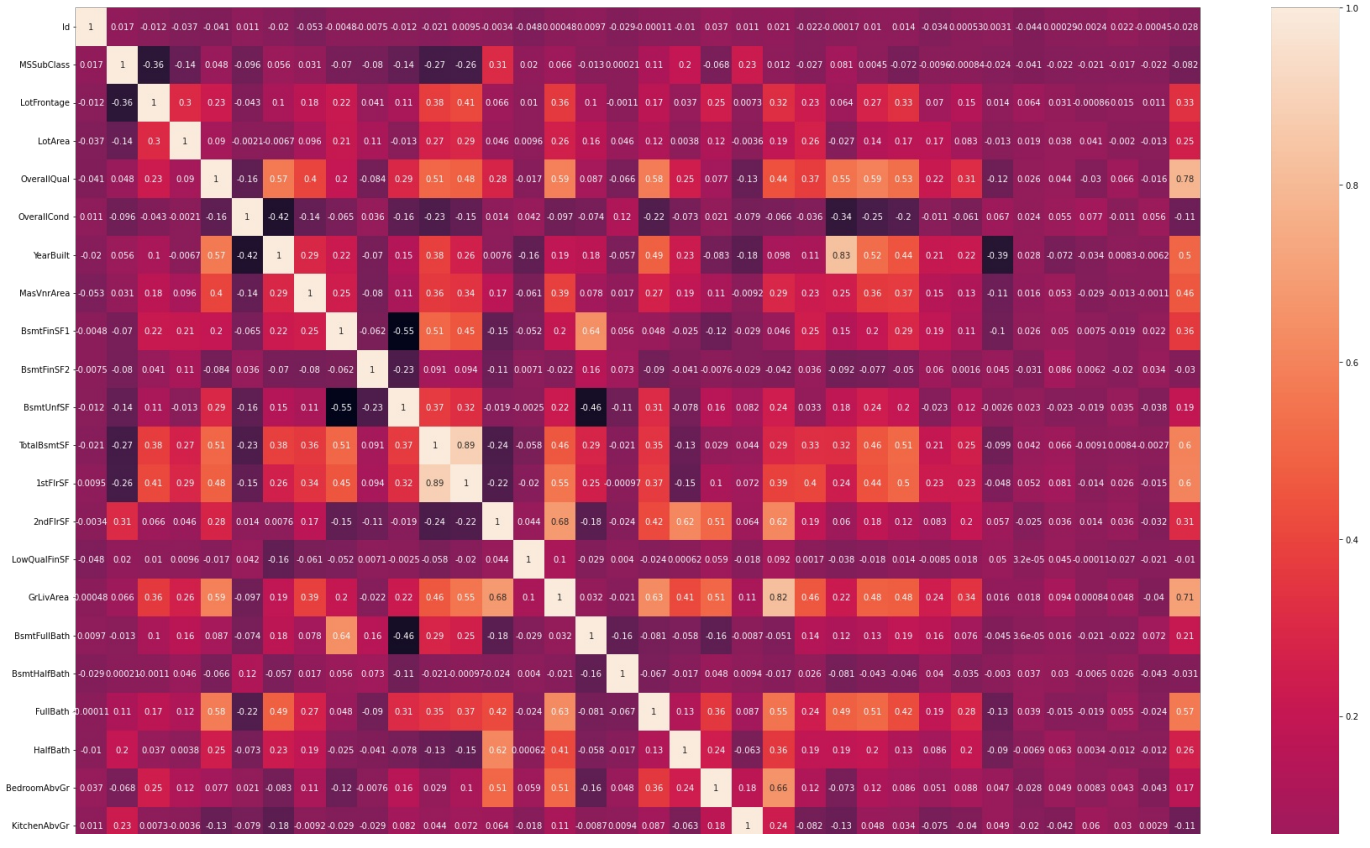
Out[18]:
```
<AxesSubplot:>
```

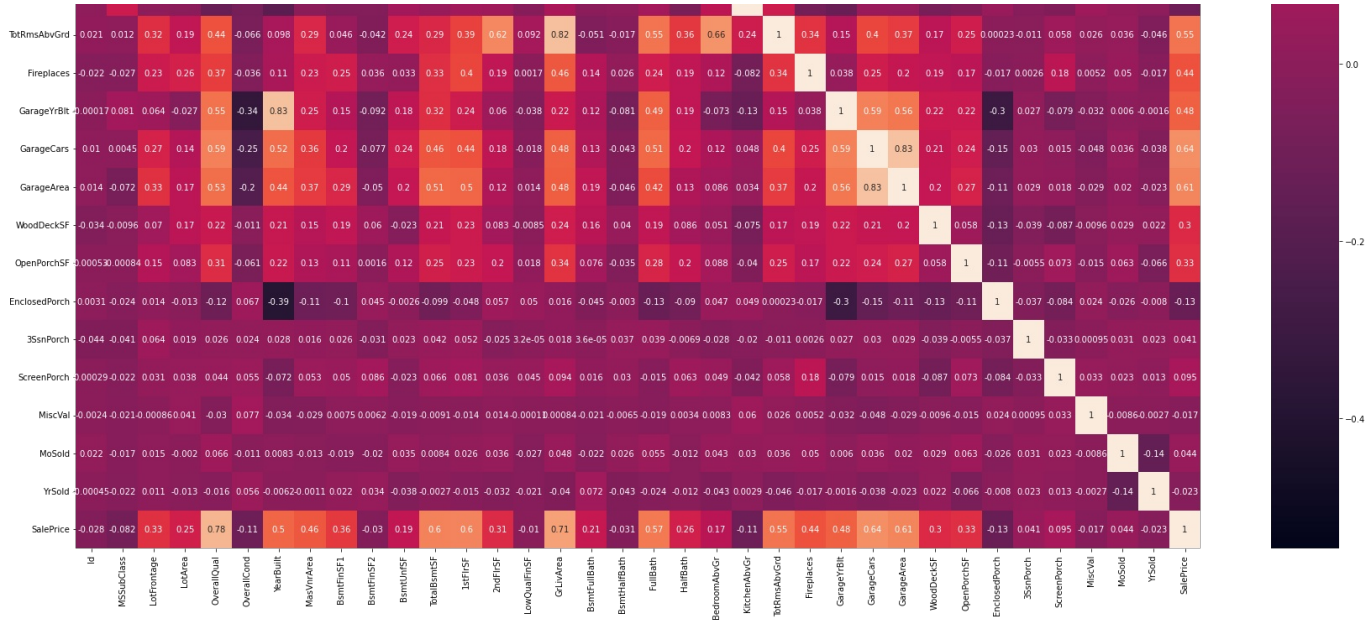


In [19]:
```
df = df.dropna()
```

In [20]:
```
plt.figure(figsize=(30, 30))
sns.heatmap(df.corr(), annot=True)
```

Out[20]:
```
<AxesSubplot:>
```



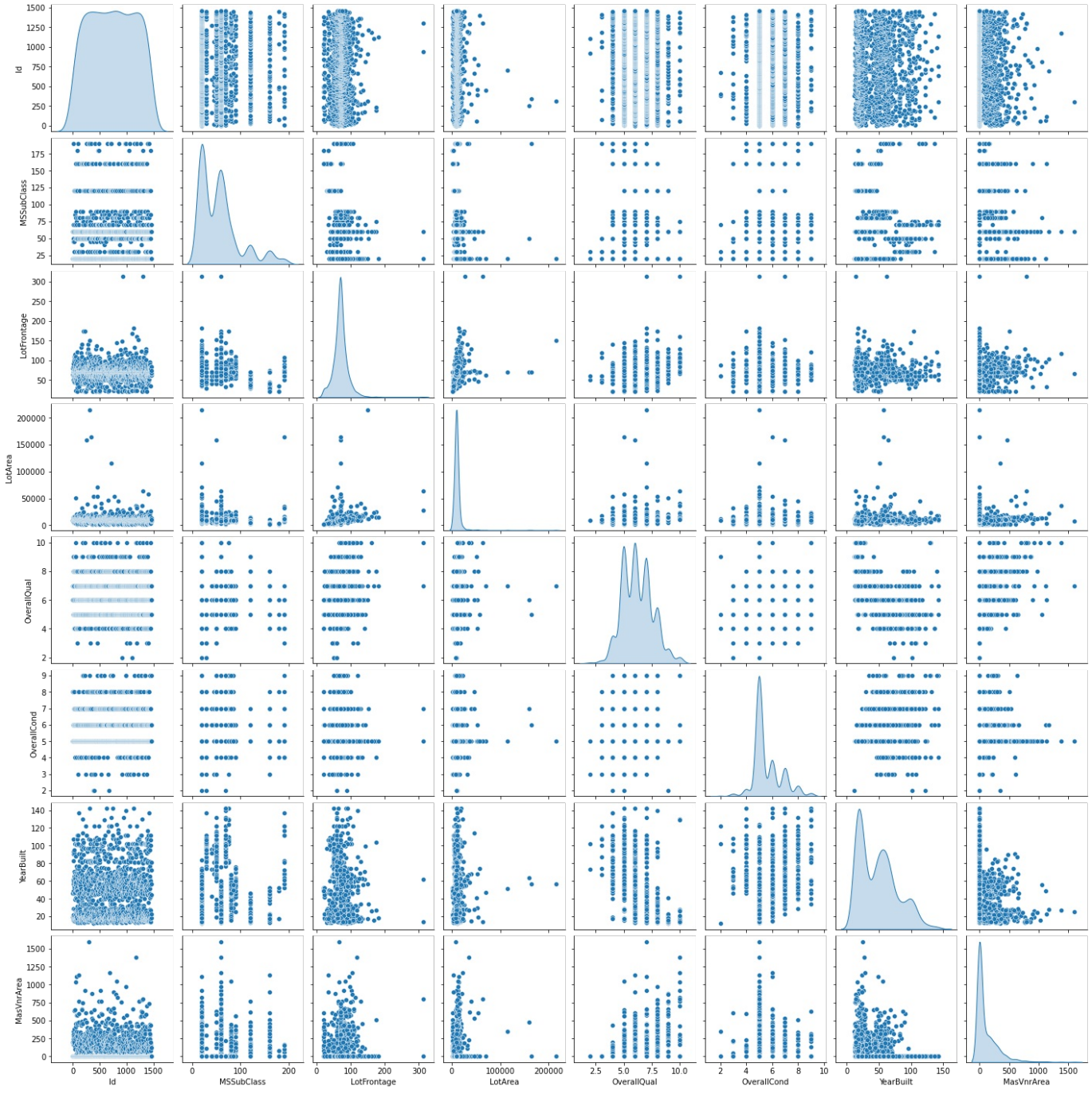


In [37]:

```
df_attr = df.iloc[:, 0:8]
sns.pairplot(df_attr, diag_kind='kde')
```

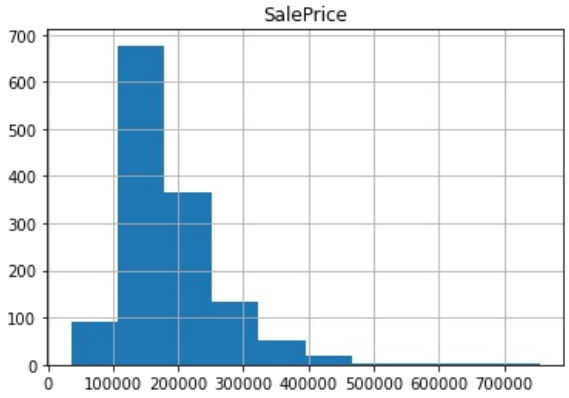
Out[37]:

```
<seaborn.axisgrid.PairGrid at 0x162ec74df70>
```



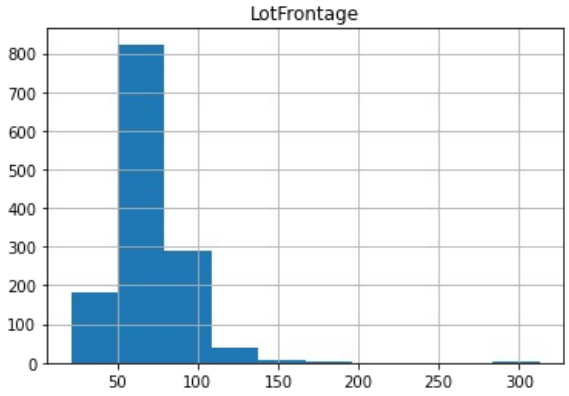
In [22]:
```
df.hist(['SalePrice'])
```

Out[22]:
```
array([[<AxesSubplot:title={'center':'SalePrice'}>]], dtype=object)
```



In [23]:
```
df.hist('LotFrontage')
```

Out[23]:
```
array([[<AxesSubplot:title={'center':'LotFrontage'}>]], dtype=object)
```



In [24]:
```
df.head()
```

Out[24]:

	Id	MSSubClass	MSZoning	LotFrontage	LotArea	LandSlope	Neighborhood	Condition1	BldgType	HouseStyle	...	OpenPorchSF	Enclosed
0	1	60	RL	65.0	8450	Gtl	CollgCr	Norm	1Fam	2Story	...	61	
1	2	20	RL	80.0	9600	Gtl	Veenker	Feedr	1Fam	1Story	...	0	
2	3	60	RL	68.0	11250	Gtl	CollgCr	Norm	1Fam	2Story	...	42	
3	4	70	RL	60.0	9550	Gtl	Crawfor	Norm	1Fam	2Story	...	35	
4	5	60	RL	84.0	14260	Gtl	NoRidge	Norm	1Fam	2Story	...	84	

5 rows × 65 columns

In [25]:
```
df['YearBuilt'] = df['YearBuilt'].apply(lambda x: 2022-x)
df['YrSold'] = df['YrSold'].apply(lambda x: 2022-x)
```

In [26]:
```
df.head()
```

Out[26]:

	Id	MSSubClass	MSZoning	LotFrontage	LotArea	LandSlope	Neighborhood	Condition1	BldgType	HouseStyle	...	OpenPorchSF	Enclosed
0	1	60	RL	65.0	8450	Gtl	CollgCr	Norm	1Fam	2Story	...	61	
1	2	20	RL	80.0	9600	Gtl	Veenker	Feedr	1Fam	1Story	...	0	
2	3	60	RL	68.0	11250	Gtl	CollgCr	Norm	1Fam	2Story	...	42	
3	4	70	RL	60.0	9550	Gtl	Crawfor	Norm	1Fam	2Story	...	35	
4	5	60	RL	84.0	14260	Gtl	NoRidge	Norm	1Fam	2Story	...	84	

5 rows × 65 columns

In [27]:
```
cat_columns = []
for col in df.columns:
    if df[col].dtype == 'object':
        cat_columns.append(col)
```

In [28]:
```
df = pd.get_dummies(df, columns = cat_columns, drop_first=True)
```

In [29]:
```
display(df.head())
```

	Id	MSSubClass	LotFrontage	LotArea	OverallQual	OverallCond	YearBuilt	MasVnrArea	BsmtFinSF1	BsmtFinSF2	...	SaleType_ConLI	Sal
0	1	60	65.0	8450	7	5	19	196.0	706	0	...	0	
1	2	20	80.0	9600	6	8	46	0.0	978	0	...	0	
2	3	60	68.0	11250	7	5	21	162.0	486	0	...	0	
3	4	70	60.0	9550	7	5	107	0.0	216	0	...	0	
4	5	60	84.0	14260	8	5	22	350.0	655	0	...	0	

5 rows × 173 columns

DATA SPLIT

In [30]:
```
X = df.drop('SalePrice', axis =1)
y = df['SalePrice']
```

In [31]:
```
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.20, random_state=42)
```

In [32]:
```
model = LinearRegression()
model.fit(X_train, y_train)
```

Out[32]:
```
LinearRegression()
```

In [33]:
```
print(model.score(X_test, y_test))
```

```
0.8562710955897329
```

In [34]:
```
print(model.score(X_train, y_train))
```

```
0.8849322936304943
```

FIT MODEL

In [35]:
```
from sklearn.preprocessing import PolynomialFeatures
from sklearn import linear_model

poly = PolynomialFeatures(degree=1, interaction_only=True)
X_train2 = poly.fit_transform(X_train)
X_test2 = poly.fit_transform(X_test)

poly_clf = linear_model.LinearRegression()

poly_clf.fit(X_train2, y_train)

y_pred = poly_clf.predict(X_test2)

print(poly_clf.score(X_train2, y_train))
```

```
0.8849322936304943
```

In [36]:
```
print(poly_clf.score(X_test2, y_test))
```

```
0.8562710955897681
```