New Section

```
import pandas as pd
import numpy as np
```

```
data=pd.read_csv('/train.csv')
```

```
data.head()
```

	beds	baths	size	size_units	lot_size	lot_size_units	zip_code	price
0	3	2.5	2590.0	sqft	6000.00	sqft	98144	795000.0
1	4	2.0	2240.0	sqft	0.31	acre	98106	915000.0
2	4	3.0	2040.0	sqft	3783.00	sqft	98107	950000.0
3	4	3.0	3800.0	sqft	5175.00	sqft	98199	1950000.0
4	2	2.0	1042.0	sqft	NaN	NaN	98102	950000.0

Next steps: Generate code with data View recommended plots New interactive sheet

```
data.shape
```

```
(2016, 8)
```

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2016 entries, 0 to 2015
Data columns (total 8 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   beds            2016 non-null   int64
 1   baths           2016 non-null   float64
 2   size            2016 non-null   float64
 3   size_units      2016 non-null   object
 4   lot_size        1669 non-null   float64
 5   lot_size_units  1669 non-null   object
 6   zip_code        2016 non-null   int64
 7   price           2016 non-null   float64
dtypes: float64(4), int64(2), object(2)
memory usage: 126.1+ KB
```

```
for column in data.columns:
    print(data[column].value_counts())
    print("*"*20)
```

```
98136     60
98102     60
98121     59
98112     57
98178     44
98168     44
98146     41
98108     33
98177     27
98101     23
98104     14
98164      1
Name: count, dtype: int64
********************
price
750000.0     27
700000.0     25
850000.0     23
950000.0     20
900000.0     19
             ..
205000.0      1
3400000.0     1
1278500.0     1
6250000.0     1
659000.0      1
Name: count, Length: 767, dtype: int64
********************
```

```
data.isna().sum()
```

```
beds               0
baths              0
size               0
size_units         0
lot_size         347
lot_size_units   347
zip_code           0
price              0
dtype: int64
```

```
data.drop(columns=['lot_size','lot_size_units'],inplace=True)
```

```
data.describe()
```

	beds	baths	size	zip_code	price
count	2016.000000	2016.000000	2016.000000	2016.000000	2.016000e+03
mean	2.857639	2.159970	1735.740575	98123.638889	9.636252e+05
std	1.255092	1.002023	920.132591	22.650819	9.440954e+05
min	1.000000	0.500000	250.000000	98101.000000	1.590000e+05
25%	2.000000	1.500000	1068.750000	98108.000000	6.017500e+05
50%	3.000000	2.000000	1560.000000	98117.000000	8.000000e+05
75%	4.000000	2.500000	2222.500000	98126.000000	1.105250e+06
max	15.000000	9.000000	11010.000000	98199.000000	2.500000e+07

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2016 entries, 0 to 2015
Data columns (total 6 columns):
 #   Column      Non-Null Count  Dtype
---  ------      --------------  -----
 0   beds        2016 non-null   int64
 1   baths       2016 non-null   float64
 2   size        2016 non-null   float64
 3   size_units  2016 non-null   object
 4   zip_code    2016 non-null   int64
 5   price       2016 non-null   float64
dtypes: float64(3), int64(2), object(1)
memory usage: 94.6+ KB
```

```
data['beds'].value_counts()
```

```
beds
3     645
2     560
4     398
1     256
5     123
```

```
6       22
9        5
7        3
8        2
15       1
14       1
Name: count, dtype: int64
```

```
data.head()
```

	beds	baths	size	size_units	zip_code	price
0	3	2.5	2590.0	sqft	98144	795000.0
1	4	2.0	2240.0	sqft	98106	915000.0
2	4	3.0	2040.0	sqft	98107	950000.0
3	4	3.0	3800.0	sqft	98199	1950000.0
4	2	2.0	1042.0	sqft	98102	950000.0

Next steps: Generate code with data View recommended plots New interactive sheet

```
data['price_per_sqft'] = data['price'] * 100000 / data['size']
```

```
data['price_per_sqft']
```

```
0       3.069498e+07
1       4.084821e+07
2       4.656863e+07
3       5.131579e+07
4       9.117083e+07
            ...
2011    6.642336e+07
2012    6.186727e+07
2013    5.373832e+07
2014    7.421384e+07
2015    3.853801e+07
Name: price_per_sqft, Length: 2016, dtype: float64
```

```
data.describe()
```

	beds	baths	size	zip_code	price	price_per_sqft
count	2016.000000	2016.000000	2016.000000	2016.000000	2.016000e+03	2.016000e+03
mean	2.857639	2.159970	1735.740575	98123.638889	9.636252e+05	5.915851e+07
std	1.255092	1.002023	920.132591	22.650819	9.440954e+05	8.327952e+07
min	1.000000	0.500000	250.000000	98101.000000	1.590000e+05	6.796117e+06
25%	2.000000	1.500000	1068.750000	98108.000000	6.017500e+05	4.452221e+07
50%	3.000000	2.000000	1560.000000	98117.000000	8.000000e+05	5.529762e+07
75%	4.000000	2.500000	2222.500000	98126.000000	1.105250e+06	6.595389e+07
max	15.000000	9.000000	11010.000000	98199.000000	2.500000e+07	3.424658e+09

```
data.describe()
```

	beds	baths	size	zip_code	price	price_per_sqft
count	2016.000000	2016.000000	2016.000000	2016.000000	2.016000e+03	2.016000e+03
mean	2.857639	2.159970	1735.740575	98123.638889	9.636252e+05	5.915851e+07
std	1.255092	1.002023	920.132591	22.650819	9.440954e+05	8.327952e+07
min	1.000000	0.500000	250.000000	98101.000000	1.590000e+05	6.796117e+06
25%	2.000000	1.500000	1068.750000	98108.000000	6.017500e+05	4.452221e+07
50%	3.000000	2.000000	1560.000000	98117.000000	8.000000e+05	5.529762e+07
75%	4.000000	2.500000	2222.500000	98126.000000	1.105250e+06	6.595389e+07
max	15.000000	9.000000	11010.000000	98199.000000	2.500000e+07	3.424658e+09

```
data
```

	beds	baths	size	size_units	zip_code	price	price_per_sqft
0	3	2.5	2590.0	sqft	98144	795000.0	3.069498e+07
1	4	2.0	2240.0	sqft	98106	915000.0	4.084821e+07
2	4	3.0	2040.0	sqft	98107	950000.0	4.656863e+07
3	4	3.0	3800.0	sqft	98199	1950000.0	5.131579e+07
4	2	2.0	1042.0	sqft	98102	950000.0	9.117083e+07
...	...	...	...	...	...	...	...
2011	3	2.0	1370.0	sqft	98112	910000.0	6.642336e+07
2012	1	1.0	889.0	sqft	98121	550000.0	6.186727e+07
2013	4	2.0	2140.0	sqft	98199	1150000.0	5.373832e+07
2014	2	2.0	795.0	sqft	98103	590000.0	7.421384e+07
2015	3	2.0	1710.0	sqft	98133	659000.0	3.853801e+07

2016 rows × 7 columns

Next steps: Generate code with `data` View recommended plots New interactive sheet