In [1]:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import folium
import json
import plotly.express as px

from matplotlib import style
from folium import plugins
from plotly.subplots import make_subplots

#from mlxtend.evaluate import bias_variance_decomp

from warnings import filterwarnings
filterwarnings('ignore')

%matplotlib inline
```

In [2]:

```
# Choropleth Function

#Zipcode choropleth maps with average values per a zipcode (King County)
def map_choropleth_zip(df, column, title, column_name):
    fig=px.choropleth_mapbox(data_frame=df, locations='zipcode', geojson=KC_zip_json, color=column,
                         mapbox_style='open-street-map', zoom=8.5, height=900, featureidkey='properties.ZCTA5CE10',
                        center={'lat': 47.403768, 'lon': -122.005863}, opacity=0.4,
                        color_continuous_scale=px.colors.sequential.YlOrRd,
                        title=title,
                        template = "plotly_dark",
                        labels={
                            column: column_name})
    fig.update_layout(
    font_family="Arial",
    font_size=16,
    font_color="white",
    title_font_family="Arial",
    title_font_color="white",
    title_font_size=20)

    fig.update_layout(
    title={
        'y':0.98,
        'x':0.5,
        'xanchor': 'center',
        'yanchor': 'top',
    })

    fig.show()
    return None

```

In [3]:

```
# Importing raw data
df=pd.read_csv('data/kc_house_data.csv')
df_test=pd.read_csv('data/df_test.csv', dtype={'recent_renovation_new_str': str})
df_zipcode_viz=pd.read_csv('data/df_zipcode_vs.csv')
KC_zip_json=json.load(open('data/wa_washington_zip_codes_geo.min.json', 'r'))
```

**Displaying DataFrames**

In [4]:
```
1 df_test.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5779 entries, 0 to 5778
Data columns (total 13 columns):
 #   Column                     Non-Null Count  Dtype
---  ------                     --------------  -----
 0   Unnamed: 0                 5779 non-null   int64
 1   sqft_living_st             5779 non-null   float64
 2   distance_st                5779 non-null   float64
 3   bathrooms_st               5779 non-null   float64
 4   grade_st                   5779 non-null   float64
 5   log_price                  5779 non-null   float64
 6   recent_renovation_new      5779 non-null   float64
 7   sqft_living                5779 non-null   int64
 8   distance                   5779 non-null   float64
 9   grade                      5779 non-null   int64
 10  bathrooms                  5779 non-null   float64
 11  price                      5779 non-null   float64
 12  recent_renovation_new_str  5779 non-null   object
dtypes: float64(9), int64(3), object(1)
memory usage: 587.1+ KB
```

In [5]:
```
1 df_zipcode_viz
```

Out[5]:

| | Unnamed: 0 | zipcode | price | bedrooms | bathrooms | sqft_living | sqft_lot | floors | waterfront | view | condition | grade | yr_built |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **0** | 0 | 98001 | 2.813837e+05 | 3.402235 | 2.012570 | 1908.256983 | 14966.393855 | 1.432961 | 0.000000 | 0.094972 | 3.329609 | 7.296089 | 1981.047486 |
| **1** | 1 | 98002 | 2.340839e+05 | 3.314721 | 1.837563 | 1627.416244 | 7509.248731 | 1.332487 | 0.000000 | 0.010152 | 3.751269 | 6.695431 | 1967.695431 |
| **2** | 2 | 98003 | 2.942254e+05 | 3.358696 | 2.053442 | 1931.438406 | 10625.137681 | 1.309783 | 0.000000 | 0.217391 | 3.373188 | 7.550725 | 1976.873188 |
| **3** | 3 | 98004 | 1.355200e+06 | 3.853968 | 2.530159 | 2910.730159 | 13084.374603 | 1.431746 | 0.003175 | 0.307937 | 3.495238 | 8.688889 | 1971.542857 |
| **4** | 4 | 98005 | 8.102897e+05 | 3.851190 | 2.424107 | 2656.803571 | 19928.785714 | 1.279762 | 0.000000 | 0.095238 | 3.696429 | 8.488095 | 1969.744048 |
| **...** | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... |
| **65** | 65 | 98177 | 6.770352e+05 | 3.397638 | 2.099409 | 2325.511811 | 11921.086614 | 1.277559 | 0.003937 | 0.814961 | 3.496063 | 7.976378 | 1960.822835 |
| **66** | 66 | 98178 | 3.106226e+05 | 3.306202 | 1.738372 | 1736.744186 | 8308.457364 | 1.186047 | 0.034884 | 0.542636 | 3.325581 | 6.829457 | 1955.325581 |
| **67** | 67 | 98188 | 2.893271e+05 | 3.437037 | 1.870370 | 1806.125926 | 10136.644444 | 1.225926 | 0.000000 | 0.148148 | 3.333333 | 7.044444 | 1965.681481 |
| **68** | 68 | 98198 | 3.029441e+05 | 3.178182 | 1.792727 | 1749.530909 | 10553.425455 | 1.225455 | 0.032727 | 0.603636 | 3.450909 | 7.109091 | 1966.865455 |
| **69** | 69 | 98199 | 7.919480e+05 | 3.208861 | 2.166930 | 2162.246835 | 5439.721519 | 1.466772 | 0.003165 | 0.553797 | 3.503165 | 8.009494 | 1956.569620 |

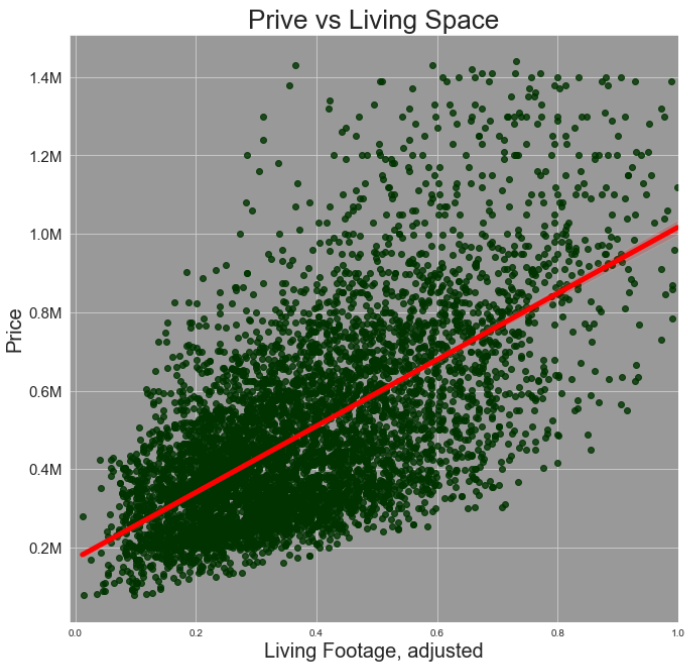
70 rows × 15 columns

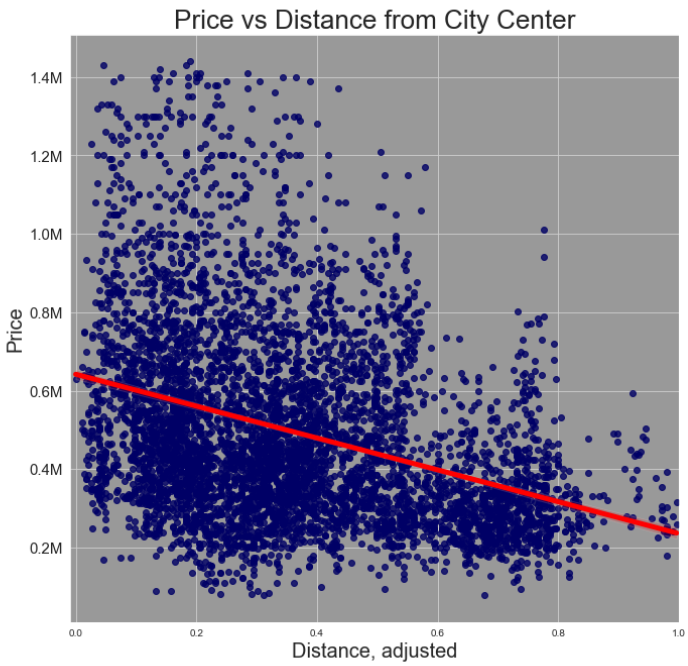
## Visualization

In [6]:

```
# Regplots for all four variables

sns.set_style("darkgrid", {"axes.facecolor": ".6"})
fig, axes = plt.subplots(figsize=(20,20), ncols=2, nrows=2)

g1=sns.regplot(data=df_test, x="sqft_living_st", y="price", color="#003300", fit_reg=True,
               ax=axes[0,0], line_kws={"color": "red", "lw":5});
g2=sns.regplot(data=df_test, x="distance_st", y="price", color="#000066",
               ax=axes[0,1], line_kws={"color": "red", "lw":5});
g3=sns.regplot(data=df_test, x="bathrooms_st", y="price", color="#009900",
               ax=axes[1,0], line_kws={"color": "red", "lw":5});
g4=sns.regplot(data=df_test, x="grade_st", y="price", color="#0000ff",
               ax=axes[1,1], line_kws={"color": "red", "lw":5});

axes[0,0].set_title("Prive vs Living Space", fontsize=26);
axes[0,0].set_ylabel('Price', fontsize=20)
axes[0,0].set_xlabel('Living Footage, adjusted', fontsize=20)
axes[0,0].set_xlim(-0.01, 1.0)
ylabels = ['{:,.1f}'.format(x) + 'M' for x in g1.get_yticks()/1000000]
axes[0,0].set_yticklabels(ylabels, size=15)
axes[0,0].grid(color='lightgrey')

axes[0,1].set_title("Price vs Distance from City Center", fontsize=26);
axes[0,1].set_ylabel('Price', fontsize=20)
axes[0,1].set_xlabel('Distance, adjusted', fontsize=20)
axes[0,1].set_xlim(-0.01, 1.0)
ylabels = ['{:,.1f}'.format(x) + 'M' for x in g2.get_yticks()/1000000]
axes[0,1].set_yticklabels(ylabels, size=15)
axes[0,1].grid(color='lightgrey')

axes[1,0].set_title("Price vs Number of Bathrooms", fontsize=26);
axes[1,0].set_ylabel('Price', fontsize=20)
axes[1,0].set_xlabel('Number of Bathrooms, adjusted', fontsize=20)
axes[1,0].set_xlim(-0.01, 1.0)
ylabels = ['{:,.1f}'.format(x) + 'M' for x in g3.get_yticks()/1000000]
axes[1,0].set_yticklabels(ylabels, size=15)
axes[1,0].grid(color='lightgrey')

axes[1,1].set_title("Price vs Building Grade", fontsize=26);
axes[1,1].set_ylabel('Price', fontsize=20)
axes[1,1].set_xlabel('Building Grade, adjusted', fontsize=20)
axes[1,1].set_xlim(-0.01, 1.0)
ylabels = ['{:,.1f}'.format(x) + 'M' for x in g4.get_yticks()/1000000]
axes[1,1].set_yticklabels(ylabels, size=15)
axes[1,1].grid(color='lightgrey')


plt.suptitle("Regression plots of Price vs Four Independent Variables", size=30, c="Blue")
plt.tight_layout(pad=3)

```

# Regression plots of Price vs Four Independent Variables

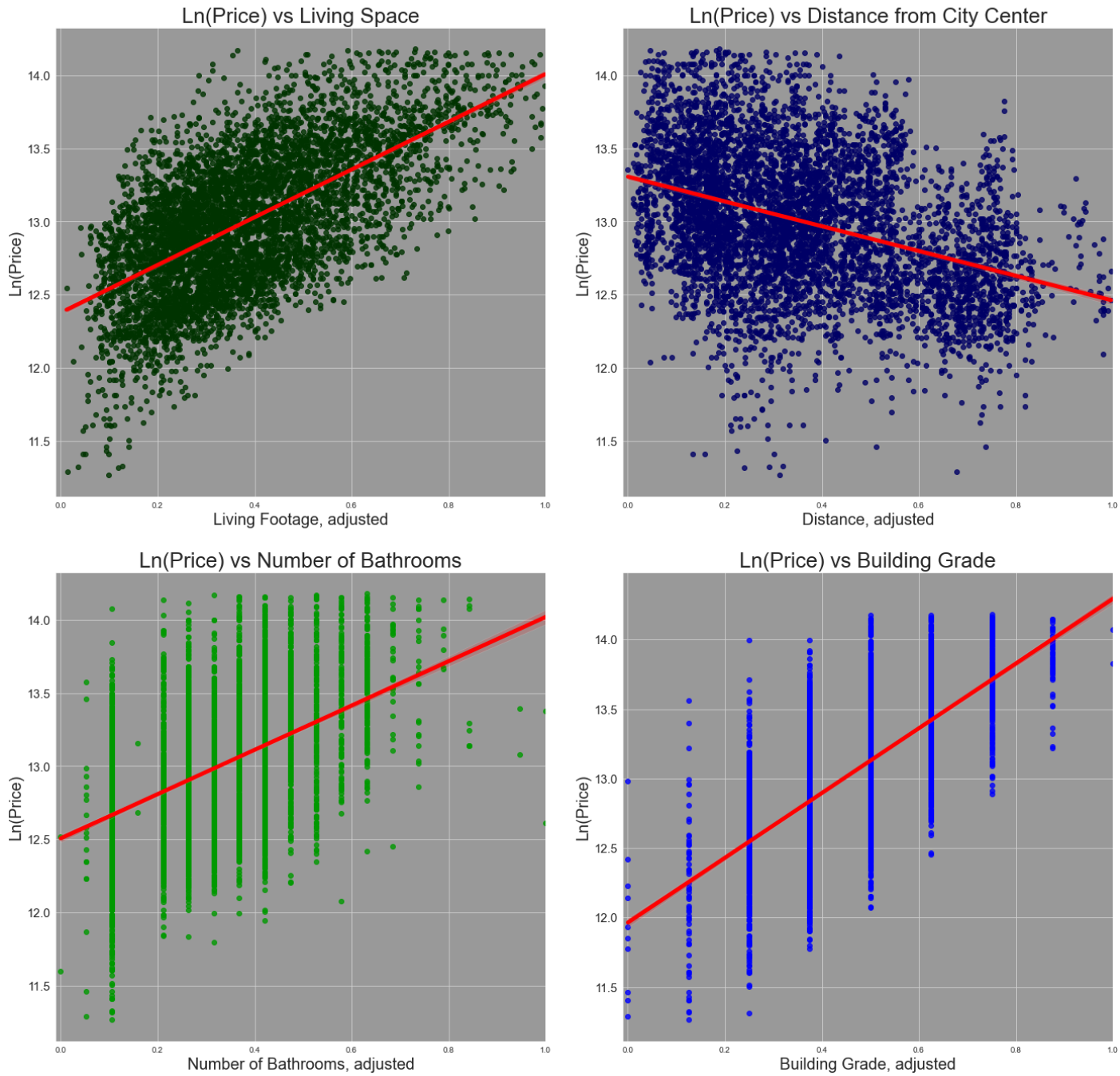




In [7]:

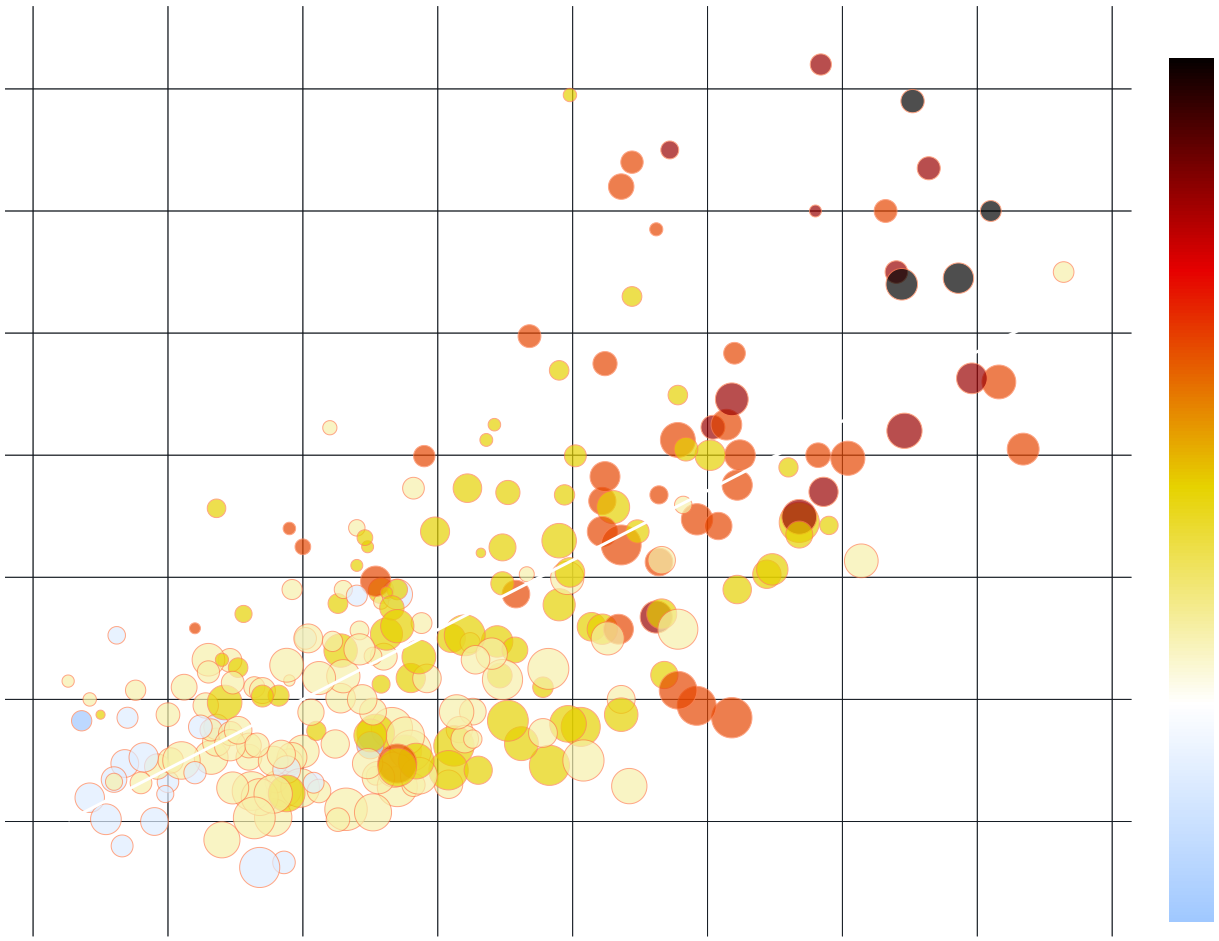
```
# Regplots for all four variables

sns.set_style("darkgrid", {"axes.facecolor": ".6"})
fig, axes = plt.subplots(figsize=(20,20), ncols=2, nrows=2)

g1=sns.regplot(data=df_test, x="sqft_living_st", y="log_price", color="#003300", fit_reg=True,
               ax=axes[0,0], line_kws={"color": "red", "lw":5});
g2=sns.regplot(data=df_test, x="distance_st", y="log_price", color="#000066",
               ax=axes[0,1], line_kws={"color": "red", "lw":5});
g3=sns.regplot(data=df_test, x="bathrooms_st", y="log_price", color="#009900",
               ax=axes[1,0], line_kws={"color": "red", "lw":5});
g4=sns.regplot(data=df_test, x="grade_st", y="log_price", color="#0000ff",
               ax=axes[1,1], line_kws={"color": "red", "lw":5});

axes[0,0].set_title("Ln(Price) vs Living Space", fontsize=26);
axes[0,0].set_ylabel('Ln(Price)', fontsize=20)
axes[0,0].set_xlabel('Living Footage, adjusted', fontsize=20)
axes[0,0].set_xlim(-0.01, 1.0)
ylabels = ['{:,.1f}'.format(x) for x in g1.get_yticks()]
axes[0,0].set_yticklabels(ylabels, size=15)
axes[0,0].grid(color='lightgrey')

axes[0,1].set_title("Ln(Price) vs Distance from City Center", fontsize=26);
axes[0,1].set_ylabel('Ln(Price)', fontsize=20)
axes[0,1].set_xlabel('Distance, adjusted', fontsize=20)
axes[0,1].set_xlim(-0.01, 1.0)
ylabels = ['{:,.1f}'.format(x) for x in g2.get_yticks()]
axes[0,1].set_yticklabels(ylabels, size=15)
axes[0,1].grid(color='lightgrey')

axes[1,0].set_title("Ln(Price) vs Number of Bathrooms", fontsize=26);
axes[1,0].set_ylabel('Ln(Price)', fontsize=20)
axes[1,0].set_xlabel('Number of Bathrooms, adjusted', fontsize=20)
axes[1,0].set_xlim(-0.01, 1.0)
ylabels = ['{:,.1f}'.format(x) for x in g3.get_yticks()]
axes[1,0].set_yticklabels(ylabels, size=15)
axes[1,0].grid(color='lightgrey')

axes[1,1].set_title("Ln(Price) vs Building Grade", fontsize=26);
axes[1,1].set_ylabel('Ln(Price)', fontsize=20)
axes[1,1].set_xlabel('Building Grade, adjusted', fontsize=20)
axes[1,1].set_xlim(-0.01, 1.0)
ylabels = ['{:,.1f}'.format(x) for x in g4.get_yticks()]
axes[1,1].set_yticklabels(ylabels, size=15)
axes[1,1].grid(color='lightgrey')

plt.suptitle("Regression plots of Ln(Price) vs Four Independent Variables", size=30, c="Blue")
plt.tight_layout(pad=3)
```




In [8]:
```
1 df_test_sample=df_test.sample(n=250, random_state=123)
```

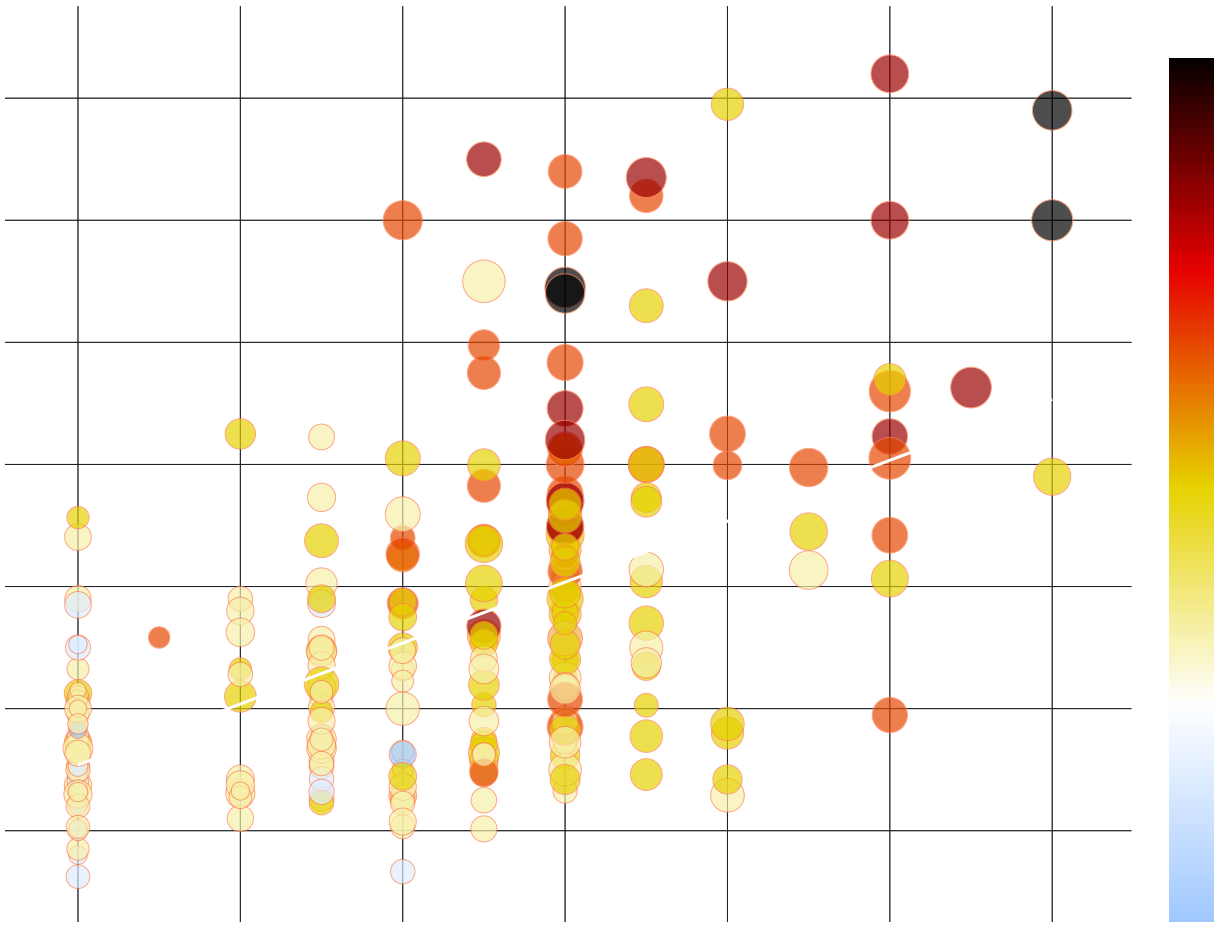
In [9]:

```
fig = px.scatter(df_test_sample, x='sqft_living', y='price',  trendline='ols', trendline_color_override='white',
                 color='grade', size='distance',width=1000, height=800, size_max=20,
                 color_continuous_scale=px.colors.sequential.Blackbody_r,
                 labels={
                     "price": "Price",
                     "sqft_living": "Living Space (sq ft)",
                     "grade": "Building Grade"
                  },
                title="<b>Correlation: Property Price vs Living Space Footage</b><br><i>Sized by Distance</i>",  template

fig.update_traces(marker=dict(
            line=dict(
                color='coral',
                width=0.5
            )))
fig.update_layout(
    font_family="Arial",
    font_size=18,
    font_color="white",
    title_font_family="Arial",
    title_font_color="white",

)
fig.update_layout(
    title={
        'y':0.95,
        'x':0.5,
        'xanchor': 'center',
        'yanchor': 'top',
})


fig.show()
```



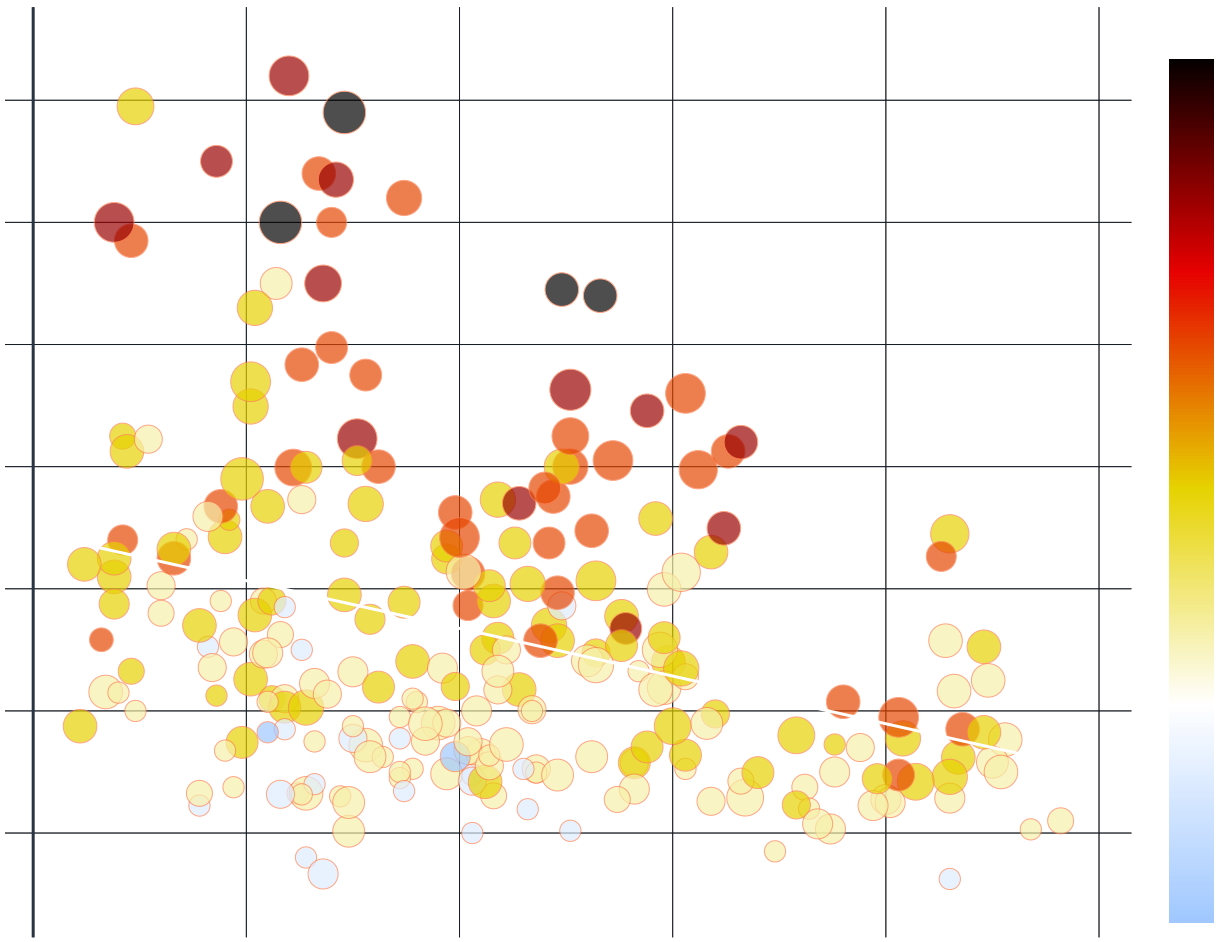
In [10]:

```
fig = px.scatter(df_test_sample, x='bathrooms', y='price', color_continuous_scale=px.colors.sequential.Blackbody_r,
                 size='sqft_living',size_max=20,
                 trendline='ols', trendline_color_override='white', color='grade',
                 width=1000, height=800, labels={
                     "price": "Price",
                     "bathrooms": "Number of bathrooms",
                     "grade": "Building Grade"
                 },
                title="<b>Correlation: Property Price vs Number of Bathrooms</b><br><i>Sized by Living Space</i>",
                 template = "plotly_dark")


fig.update_traces(marker=dict(
            line=dict(
                color='coral',
                width=0.5
            )))
fig.update_layout(
    font_family="Arial",
    font_size=18,
    font_color="white",
    title_font_family="Arial",
    title_font_color="white",

)

fig.update_layout(
    title={
        'y':0.95,
        'x':0.5,
        'xanchor': 'center',
        'yanchor': 'top',
})


fig.show()
```



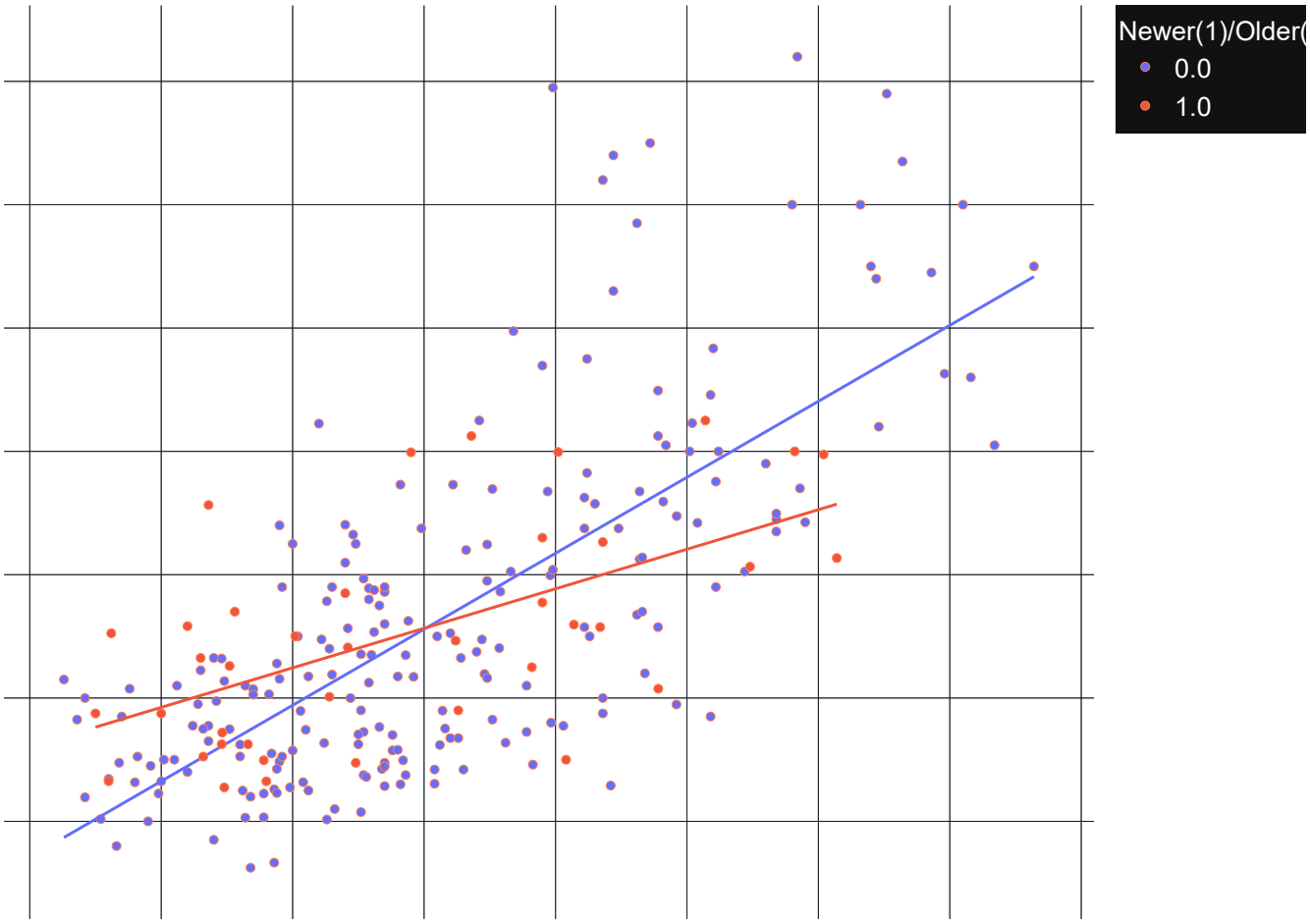
In [11]:

```
fig = px.scatter(df_test_sample, x='distance', y='price',  trendline='ols', trendline_color_override='white',
                 color='grade', size='bathrooms',width=1000, height=800, size_max=20,
                 color_continuous_scale=px.colors.sequential.Blackbody_r,
                 labels={
                     "price": "Price",
                     "distance": "Distance",
                     "grade": "Building Grade"
                  },
                title="<b>Correlation: Property Price vs Distance from the City Center</b><br><i>Sized by Number of Bathro

fig.update_traces(marker=dict(
            line=dict(
                color='coral',
                width=0.5
            )))
fig.update_layout(
    font_family="Arial",
    font_size=18,
    font_color="white",
    title_font_family="Arial",
    title_font_color="white",

)
fig.update_layout(
    title={
        'y':0.95,
        'x':0.5,
        'xanchor': 'center',
        'yanchor': 'top',
})


fig.show()
```



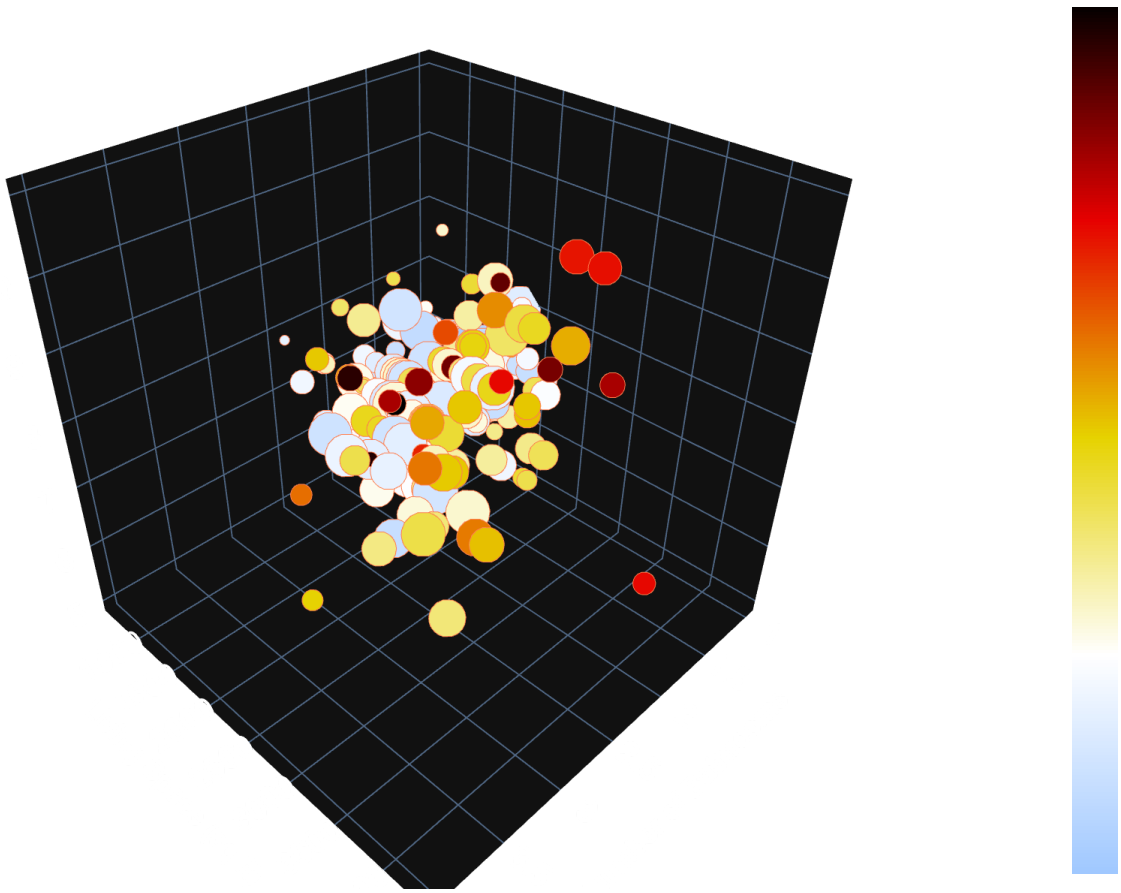
In [12]:

```
fig = px.scatter(df_test_sample, x='sqft_living', y='price',  trendline='ols',
                 color='recent_renovation_new_str', width=1000, height=800, size_max=20,
                 labels={
                     "price": "Price",
                     "sqft_living": "Living Space (sq ft)",
                     "recent_renovation_new_str": "Newer(1)/Older(0)"
                  },
                title="Correlation: Property Price vs Living Space Footage of Newer vs Older Properties",
                 template = "plotly_dark")

fig.update_traces(marker=dict(
            line=dict(
                color='coral',
                width=0.5
            )))
fig.update_layout(
    font_family="Arial",
    font_size=18,
    font_color="white",
    title_font_family="Arial",
    title_font_color="white",

)
fig.update_layout(
    title={
        'y':0.95,
        'x':0.5,
        'xanchor': 'center',
        'yanchor': 'top',
})


fig.show()
```



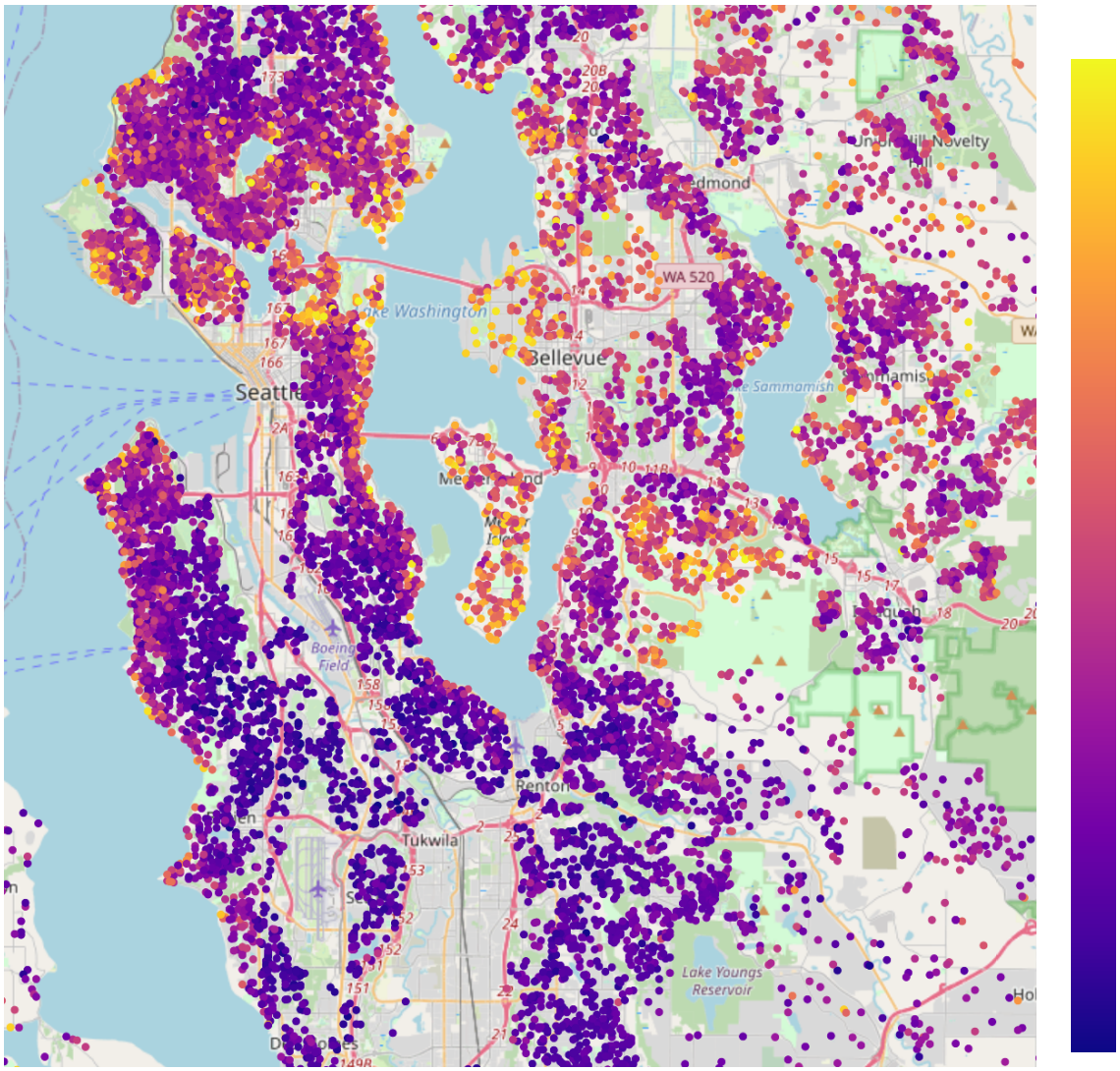
In [13]:

```
fig = px.scatter_3d(df_test_sample, x='bathrooms', z='grade', y='sqft_living',
                color='price', size='distance', size_max=50, opacity=1, width=1000, height=800,
                     color_continuous_scale=px.colors.sequential.Blackbody_r,
                     labels={
                       "bathrooms": "Number of Bathrooms",
                       "sqft_living": "Living Space (sq ft)",
                       "grade": "Grade",
                         "price": "Price"
                    },
                   title="3D plot: Living Space Footage, Number of Bathrooms and Grade of Sold Properties",
                      template = "plotly_dark")

fig.update_traces(marker=dict(
             line=dict(
                 color='coral',
                 width=0.5
             )))
fig.update_layout(
    font_family="Arial",
    font_size=16,
    font_color="white",
    title_font_family="Arial",
    title_font_color="white"
)
fig.update_layout(
    title={
        'y':0.9,
        'x':0.5,
        'xanchor': 'center',
        'yanchor': 'top',
})


fig.show()
```



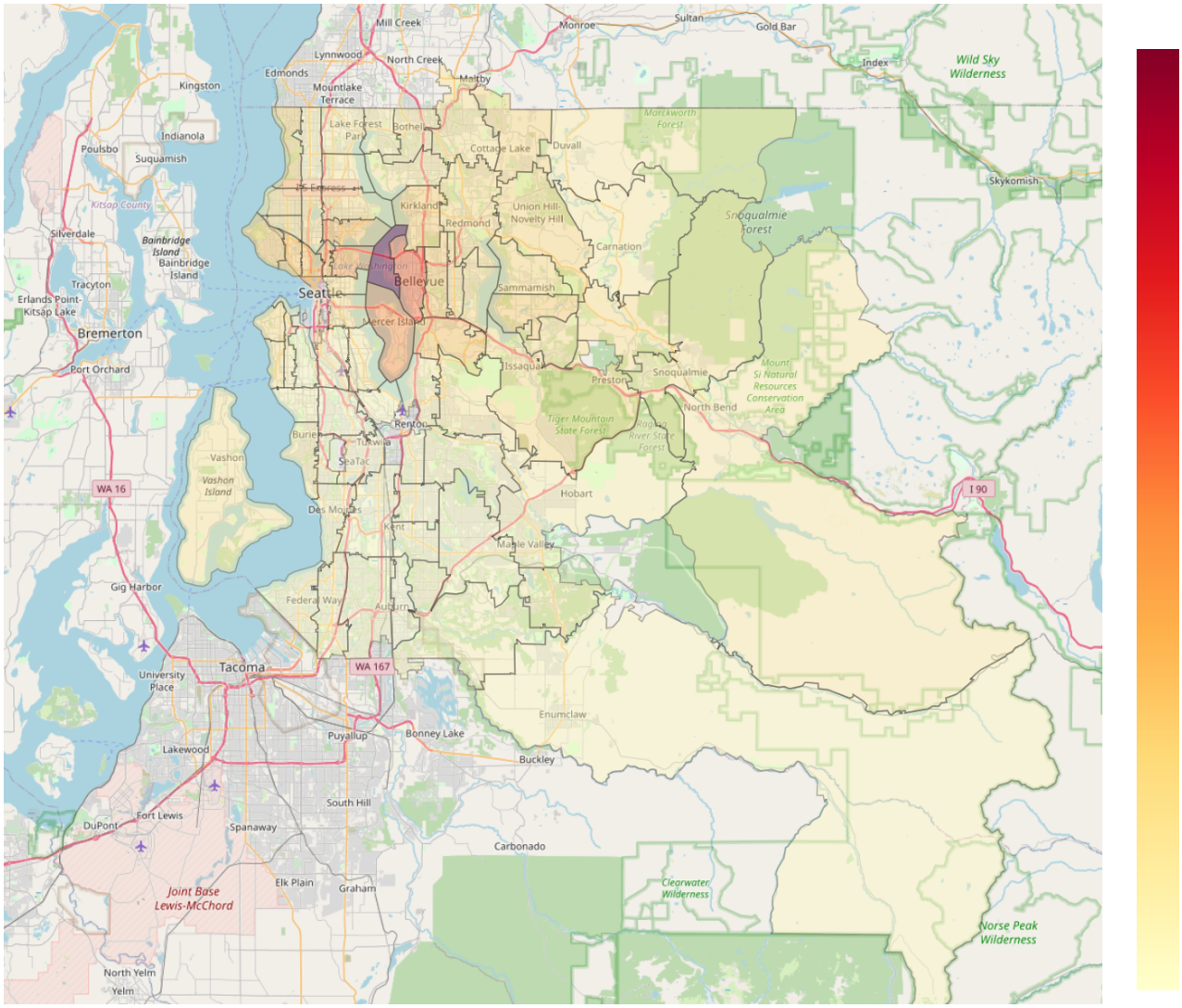
In [14]:

```
df_cut_off=df.copy()
df_cut_off=df_cut_off[(df_cut_off['price']<1500000)]

fig = px.scatter_mapbox(df_cut_off, lat="lat", lon="long", color="price",
                  color_continuous_scale=px.colors.sequential.Plasma, zoom=10,
                        mapbox_style='open-street-map', width=900, height=900,
                        title="Properties Sold in King County in 2014-2015",
                   template = "plotly_dark")

fig.update_layout(
    font_family="Arial",
    font_size=20,
    font_color="white",
    title_font_family="Arial",
    title_font_color="white"
)
fig.update_layout(
    title={
        'y':0.98,
        'x':0.5,
        'xanchor': 'center',
        'yanchor': 'top',
})


fig.show()
```



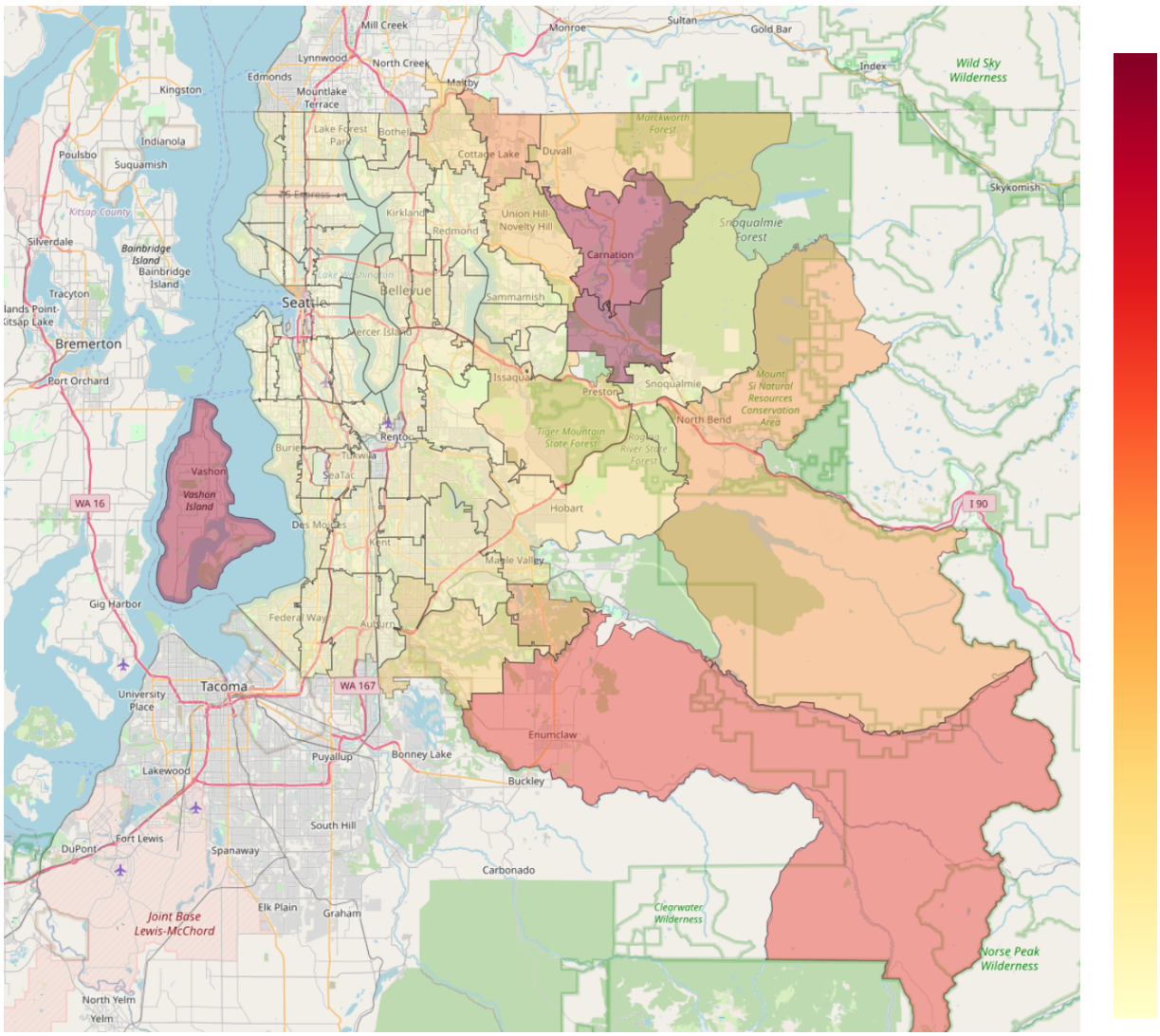
In [15]:

```
map_choropleth_zip(df_zipcode_viz, 'price', "Average Prices of Sold Properties per Zipcode (King County, 2014-2015)",
                   "Price")
```



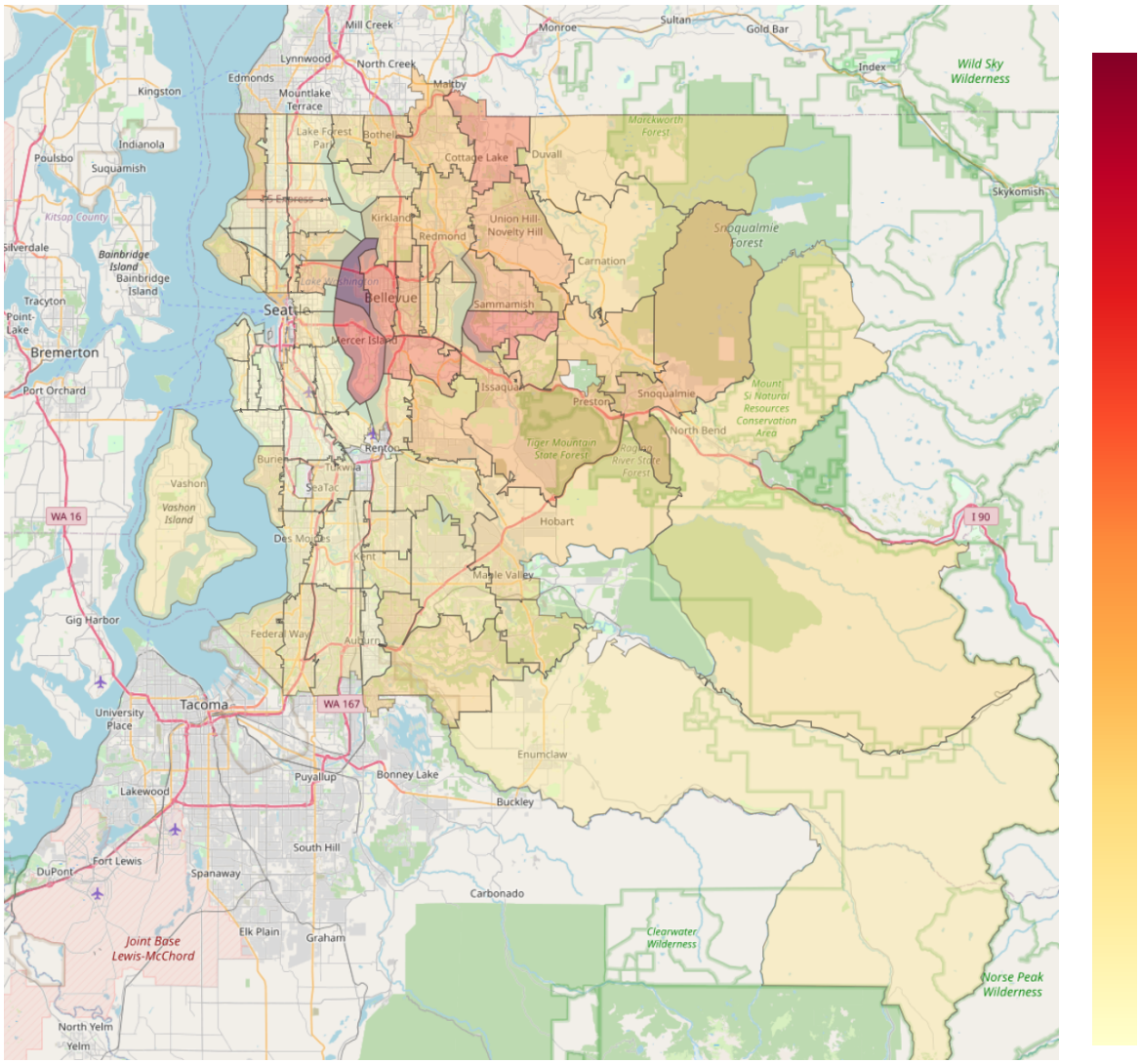
In [16]:

```
map_choropleth_zip(df_zipcode_viz, 'sqft_lot', "Average Lot Size of Sold Properties per Zipcode (King County, 2014-2015)",
                   "Lot size (sq ft)")
```



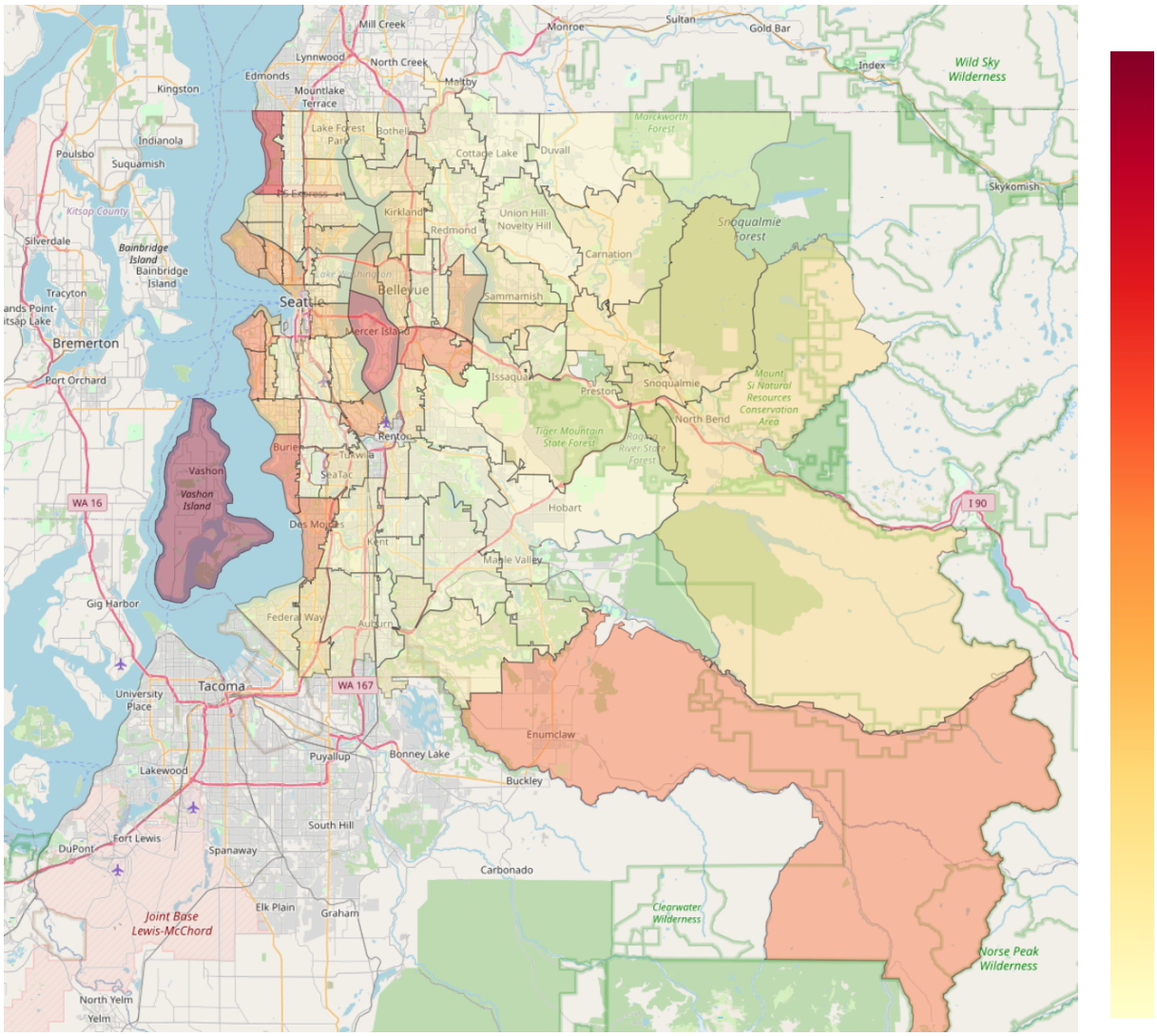
In [17]:

```
map_choropleth_zip(df_zipcode_viz, 'sqft_living', "Average Living Space of Sold Properties per Zipcode (King County, 2014
                   "Living Space (sq ft) ")
```



In [18]:

```
map_choropleth_zip(df_zipcode_viz, 'view', "Average View Category of Sold Properties per Zipcode (King County, 2014-2015)
                   "View Category")
```



In [19]:

```
map_choropleth_zip(df_zipcode_viz, 'yr_built', "Average Year Built of Sold Properties per Zipcode (King County, 2014-2015
                   "Year Built")
```