Programming for Real Estate Analytics

Use Python to process and analyze real estate pricing and licensee datasets in the U.S.

Objective

The U.S. housing market has experienced numerous crisis big and small in the past decades while the Covid 19 pandemic has hit it dramatically heavy in 2020. By using the historical median home sales price data provided by Zillow, this project shows the change of home sale prices in all states and forecasts the housing prices in the coming years. It also demonstrates the markets with great potential by sorting the real estate licensee database.

Project Mission

- File of Median Home Sale Price
  - What is the percentage change in each county from 1996 to 2020?
  - Identifying the most expensive areas in the U.S
  - Forecast the increase/decrease rate of home sale price
- File of Licensee
  - The number of active brokerages and individual agents in a certain area
  - The business growth
  - Identify years and areas that brokerages experience significant business increase or decrease

Report Summary

- Housing price and percentage change in certain states from 2007 to 2020 is demonstrated by sorting and graph plotting functions. The functions show the dramatic price change clearly during the financial crisis, and Covid year.
- The brokerage business landscape has a strong correlation with the home selling prices. Specifically,

- In 2010, when the housing price went up, more salespersons were active on the market.
- In 2011, when the housing price went down, less people got their real estate licenses.
- In 2020, there are more transactions, more people selling houses, but overall, the price is slightly getting down.

Business Analysis

Conclusions based on the outputs above:

- High price leads to high commission, and high commission leads to high motivation. Therefore the number of real estate agents will increase greatly when the housing price goes up.
- There is a huge price gap between suburban housing market and the city's, so that even more people buying suburban, the median price and overall transaction deals are lower than cities.
- The price change in specifical area forecasts new opportunities on the market, in where developers could build new properties and brokerage could train more agents.

Data Sources

- **Zillow** Zillow's public housing database includes more than 110 million U.S. homes from sales to rentals.
- **ARELLO** The Association of Real Estate License Law Officials(ARELLO) supports regulatory agencies in the administration and enforcement of real estate license laws in their respective jurisdictions.

Team Members

JingJing Xu | Sara Basha | Maulik Pareliya | Bakary Kida

U.S. Median Housing Price Analysis

In [1]:
```
import pandas as pd
import numpy as np
import seaborn as sns
```

In [2]:
```
%matplotlib notebook
import matplotlib.pyplot as plt
```

In [5]:
```
##Showing the Total Active Salesperson for each Jurisdiction
## i and j are the location for the annotated text
i = 1.0
j = 2000
for i in range(len(activesalesDF['Jurisdiction'])):
    plt.annotate(activesalesDF['Active Salesperson'][i], (-0.1 + i, activesalesDF['Active Salesperson'][i] +
plt.legend(labels = ['Total Active Salesperson'])
```

Out[5]:
```
<matplotlib.legend.Legend at 0x7f9a65b50bb0>
```

In [6]:
```
## If we are interested in the Jurisdiction where the Active Salesperson is greater than 0
activesalesDF[(activesalesDF['Active Salesperson'] > 0)]
```

Out[6]:

	Jurisdiction	Active Salesperson	Inactive Salesperson	Active Broker	Inactive Broker	Active Associate Broker	Inactive Associate Broker	Active Managing Broker	Inactive Managing Broker	Active Firm	Inactive Firm	Total Active
0	ALABAMA	14460	6245.0	4015.0	NaN	3133.0	1562.0	NaN	NaN	3745.0	183.0	25353
1	ALASKA	1679	1.0	380.0	0.0	363.0	0.0	NaN	NaN	NaN	NaN	2422
3	ARKANSAS	7348	1984.0	3883.0	437.0	NaN	NaN	NaN	NaN	NaN	NaN	11231
4	CALIFORNIA	205937	86077.0	94630.0	11049.0	NaN	NaN	NaN	NaN	23425.0	709.0	323992
8	DISTRICT OF COLUMBIA	10509	25259.0	2679.0	3384.0	NaN	NaN	NaN	NaN	1310.0	2308.0	14498
10	GEORGIA	51766	12318.0	17711.0	2905.0	NaN	NaN	NaN	NaN	10938.0	456.0	80415
11	HAWAII	8782	3682.0	3468.0	582.0	NaN	NaN	NaN	NaN	1512.0	39.0	13762
12	IDAHO	8993	2425.0	2323.0	333.0	1197.0	NaN	1105.0	NaN	1190.0	NaN	14808
15	IOWA	6497	1154.0	2935.0	343.0	NaN	NaN	NaN	NaN	1098.0	NaN	10530
16	KANSAS	12781	710.0	3566.0	61.0	NaN	NaN	NaN	NaN	NaN	NaN	16347
19	MAINE	996	15.0	1491.0	181.0	2946.0	270.0	817.0	NaN	1033.0	NaN	7283
20	MARYLAND	38872	1806.0	4389.0	31.0	3092.0	79.0	NaN	NaN	NaN	NaN	46353
23	MISSISSIPPI	5254	1475.0	2314.0	385.0	970.0	129.0	NaN	NaN	1991.0	135.0	10529
26	NEBRASKA	5040	498.0	2052.0	129.0	NaN	NaN	NaN	NaN	NaN	NaN	7092
27	NEVADA	18240	2034.0	2986.0	758.0	3034.0	558.0	NaN	NaN	NaN	NaN	24260
28	NEW HAMPSHIRE	5964	1883.0	3676.0	441.0	NaN	NaN	NaN	NaN	1182.0	NaN	10822
32	NORTH DAKOTA	1730	217.0	357.0	32.0	294.0	10.0	NaN	NaN	384.0	0.0	2765
33	OHIO	34447	10598.0	1.0	2.0	868.0	416.0	3820.0	46.0	3473.0	4600.0	42609
34	OKLAHOMA	9189	2608.0	3943.0	628.0	NaN	NaN	NaN	NaN	2848.0	136.0	15980
38	SOUTH CAROLINA	29687	8116.0	NaN	1670.0	5166.0	NaN	7717.0	NaN	11397.0	NaN	53967
41	TEXAS	114456	30310.0	32737.0	1750.0	NaN	NaN	NaN	NaN	11268.0	NaN	158461

	Jurisdiction	Active Salesperson	Inactive Salesperson	Active Broker	Inactive Broker	Active Associate Broker	Inactive Associate Broker	Active Managing Broker	Inactive Managing Broker	Active Firm	Inactive Firm	Total Active
44	VIRGINIA	42321	8368.0	6037.0	390.0	5709.0	529.0	NaN	NaN	6047.0	NaN	60114
46	WEST VIRGINIA	3126	1553.0	813.0	96.0	295.0	55.0	NaN	NaN	NaN	NaN	4234
47	WYOMING	1294	571.0	604.0	98.0	627.0	201.0	NaN	NaN	571.0	NaN	3096

In [7]:
```
## Checking specific Jurisdiction row with all the columns the document have
user_state = input("Enter the Jurisdiction you would like to view: ").upper()
activesalesDF[(activesalesDF['Jurisdiction'] == user_state)]
```

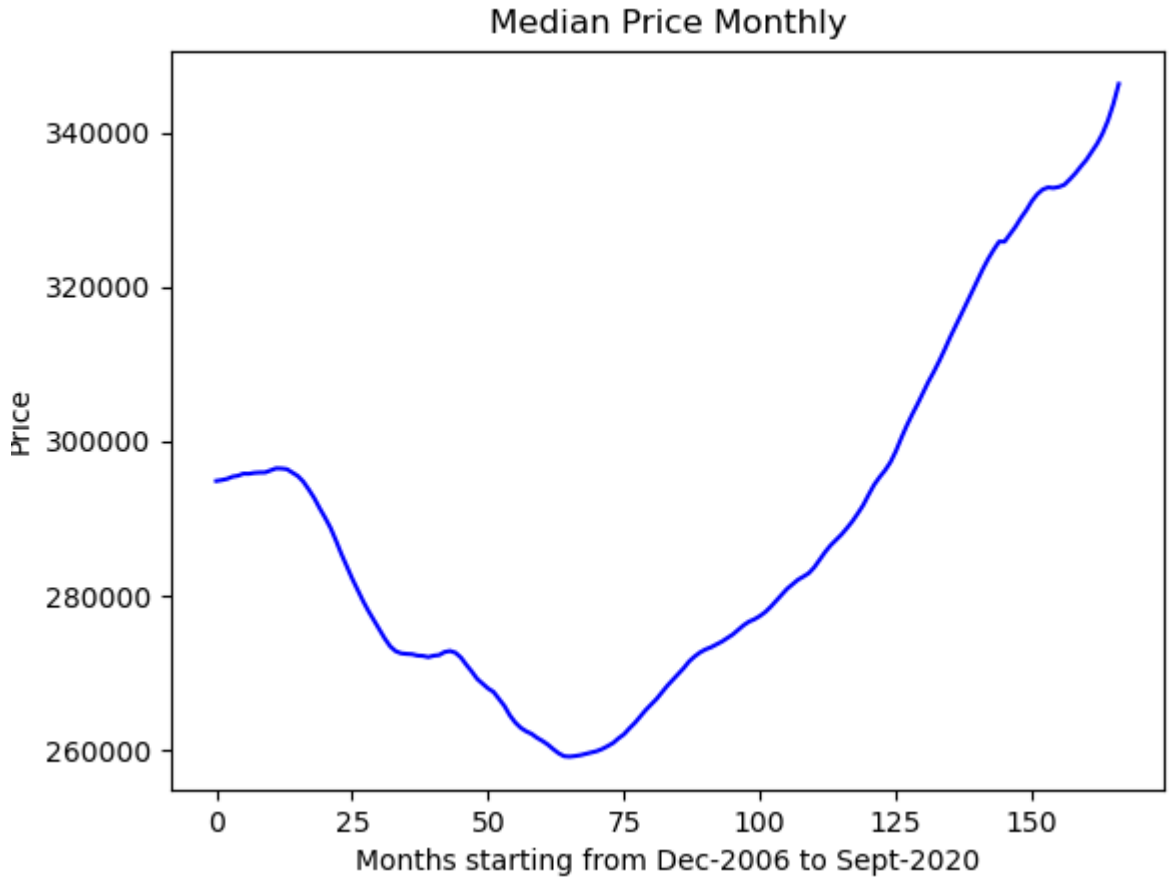
```
Enter the Jurisdiction you would like to view: california
```

Out[7]:

	Jurisdiction	Active Salesperson	Inactive Salesperson	Active Broker	Inactive Broker	Active Associate Broker	Inactive Associate Broker	Active Managing Broker	Inactive Managing Broker	Active Firm	Inactive Firm	Total Active
4	CALIFORNIA	205937	86077.0	94630.0	11049.0	NaN	NaN	NaN	NaN	23425.0	709.0	323992

In [8]:
```
state_zhvidf= pd.read_csv('State_zhvi_uc_sfrcondo_tier_0.33_0.67_sm_sa_mon.csv',index_col=['RegionName'])
myPlot = plt.figure().add_subplot(1,1,1)
m=['12/31/06','1/31/07','2/28/07','3/31/07','4/30/07','5/31/07','6/30/07','7/31/07','8/31/07','9/30/07','10/
myPlot.plot(range(len(m)),state_zhvidf.loc['New York',['12/31/06','1/31/07','2/28/07','3/31/07','4/30/07','5
myPlot.set_title("Median Price Monthly")
myPlot.set_ylabel("Price")
myPlot.set_xlabel("Months starting from Dec-2006 to Sept-2020")
```

```
<IPython.core.display.Javascript object>
```



Out[8]:
```
Text(0.5, 0, 'Months starting from Dec-2006 to Sept-2020')
```

In [10]:
```
## If you would like to analyze a specific region enter the region name
region_name=input("Enter the region name:").capitalize()
state_zhvidf.loc[region_name]
```

```
Enter the region name:alabama
```

Out[10]:
```
RegionID          4
SizeRank         22
RegionType     State
StateName         AL
1/31/96        86322
                ...
5/31/20       151635
6/30/20       152499
7/31/20       153389
8/31/20       154467
9/30/20       155782
Name: Alabama, Length: 301, dtype: object
```

In [11]:
```
#Unedited csv file
pricedf= pd.read_csv('usprice_cust.csv')
pricedf.head()
```

Out[11]:

	Median and Average Sales Prices of New Homes Sold in United States	Unnamed: 1	Unnamed: 2
0	NaN	NaN	NaN
1	Annual Data	NaN	NaN
2	NaN	NaN	NaN
3	Period	Median	Average
4	1963	18000	19300

In [12]:
```
## giving names to columns and removing the Nan values
pricedf= pd.read_csv('usprice_cust.csv',names=['Period','Median','Average'],header=None)
pricedfNew=pricedf.drop(pricedf.index[[0,1,2,3,4,-1,-2,-3]])
pricedfNew
```

Out[12]:

	Period	Median	Average
5	1963	18000	19300
6	1964	18900	20500
7	1965	20000	21500
8	1966	21400	23300
9	1967	22700	24600
10	1968	24700	26600
11	1969	25600	27900
12	1970	23400	26600
13	1971	25200	28300
14	1972	27600	30500
15	1973	32500	35500
16	1974	35900	38900
17	1975	39300	42600
18	1976	44200	48000
19	1977	48800	54200
20	1978	55700	62500
21	1979	62900	71800
22	1980	64600	76400
23	1981	68900	83000
24	1982	69300	83900
25	1983	75300	89800
26	1984	79900	97600
27	1985	84300	100800

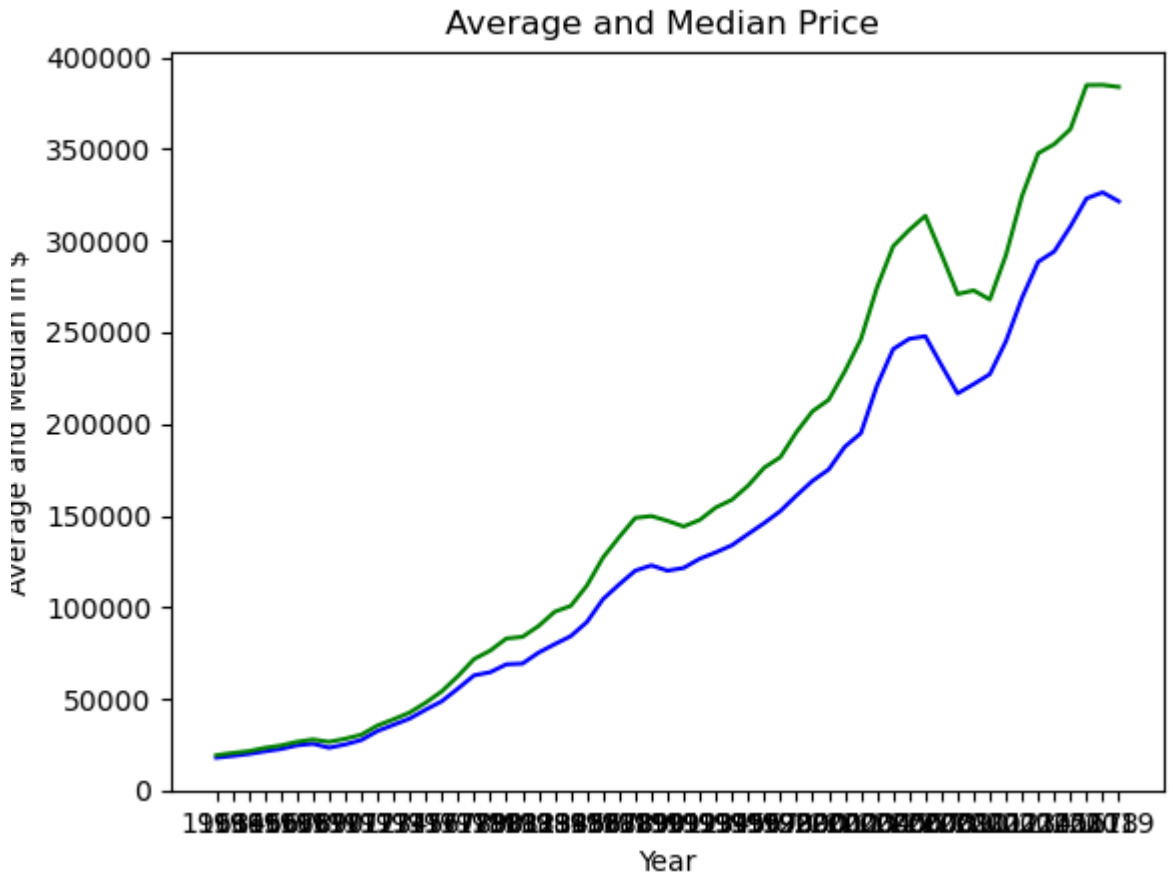
	Period	Median	Average
28	1986	92000	111900
29	1987	104500	127200
30	1988	112500	138300
31	1989	120000	148800
32	1990	122900	149800
33	1991	120000	147200
34	1992	121500	144100
35	1993	126500	147700
36	1994	130000	154500
37	1995	133900	158700
38	1996	140000	166400
39	1997	146000	176200
40	1998	152500	181900
41	1999	161000	195600
42	2000	169000	207000
43	2001	175200	213200
44	2002	187600	228700
45	2003	195000	246300
46	2004	221000	274500
47	2005	240900	297000
48	2006	246500	305900
49	2007	247900	313600
50	2008	232100	292600
51	2009	216700	270900
52	2010	221800	272900
53	2011	227200	267900
54	2012	245200	292200

	Period	Median	Average
55	2013	268900	324500
56	2014	288500	347700
57	2015	294200	352700
58	2016	307800	360900
59	2017	323100	384900
60	2018	326400	385000
61	2019	321500	383900

In [13]:
```
##Creating a line chart for the Median Price

Plot= plt.figure().add_subplot(1,1,1)
Plot.plot(pricedfNew['Period'],pricedfNew['Median'].astype(int),color='b')
Plot.set_title("Average and Median Price ")
Plot.set_ylabel("Average and Median in $")
Plot.set_xlabel("Year")
```

```
<IPython.core.display.Javascript object>
```



Out[13]: Text(0.5, 0, 'Year')

U.S. Licensee Analysis

In [1]:
```
import pandas as pd
import numpy as np
import seaborn as sns
```

In [2]:
```
df_list = []
excel_file_path = 'Licensee Counts Python.xlsx'
```

In [3]:
```
for year in range(2007,2021):
    temp = pd.read_excel(excel_file_path, str(year))
    temp.loc[:,'Year'] = int(year)
    df_list.append(temp)
```

In [4]:
```
df = pd.concat(df_list)
df['Jurisdiction'] = df.Jurisdiction.str.upper()

df = df.groupby(['Jurisdiction','Year']).agg({'Active Salesperson':'sum','Active Broker':'sum'}).reset_index
df.head()
states = df.Jurisdiction.unique().tolist()
```

In [8]:
```
df = df[['Jurisdiction', 'Year', 'Active Salesperson', 'Active Broker']]
```

In [9]:
```
df.loc[((df['Active Salesperson'] == ' ') | (df['Active Broker'] == ' ')),:]
```

72	COLORADO	2010		31826
73	COLORADO	2011		31826
186	ILLINOIS	2014		16773
268	MARYLAND	2007		
271	MARYLAND	2010		
272	MARYLAND	2011		
290	MASSACHUSETTS	2016		
291	MASSACHUSETTS	2017		
400	NEW MEXICO	2007		4612
402	NEW MEXICO	2009		4457
406	NEW MEXICO	2013		3662
478	OREGON	2013		14970
543	TENNESSEE	2011	27354	
615	WASHINGTON	2014	21752	

In [10]:
```
df = df.loc[~((df['Active Salesperson'] == ' ') | (df['Active Broker'] == ' ')),:]
```

In [11]:
```
for state in states:
    df.loc[df.Jurisdiction == str(state),'pc_Salesperson_YoY'] = df.loc[df.Jurisdiction == str(state),'Activ
    df.loc[df.Jurisdiction == str(state),'pc_Broker_YoY'] = df.loc[df.Jurisdiction == str(state),'Active Bro
```

In [12]:
```
df.fillna(value = 0,inplace = True)
df
```

Out[12]:

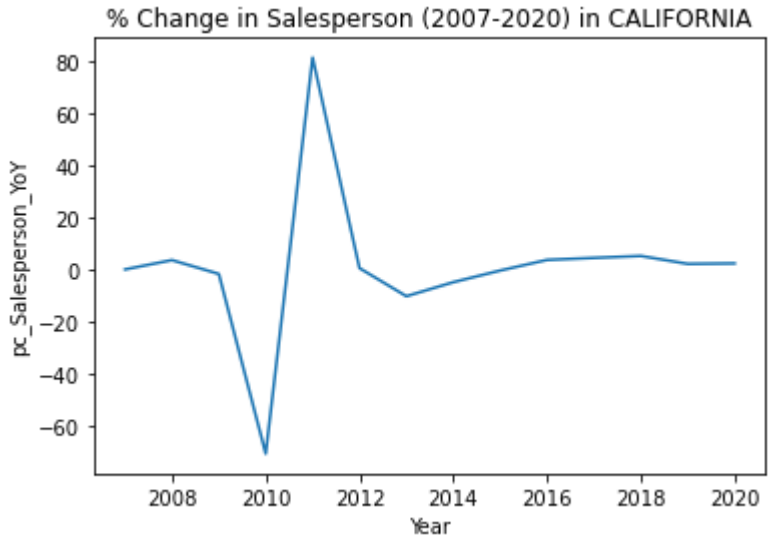
	Jurisdiction	Year	Active Salesperson	Active Broker	pc_Salesperson_YoY	pc_Broker_YoY
0	ALABAMA	2007	10330.0	5220.0	0.000000	0.000000
1	ALABAMA	2008	12957.0	7315.0	25.430784	40.134100
2	ALABAMA	2009	11467.0	6832.0	-11.499576	-6.602871
3	ALABAMA	2010	11734.0	4443.0	2.328421	-34.967799
4	ALABAMA	2011	9781.0	6742.0	-16.643941	51.744317
...	...	...	...	...	...	...
653	WYOMING	2016	1311.0	750.0	17.053571	7.913669
654	WYOMING	2017	1155.0	596.0	-11.899314	-20.533333
655	WYOMING	2018	1262.0	596.0	9.264069	0.000000
656	WYOMING	2019	1193.0	588.0	-5.467512	-1.342282
657	WYOMING	2020	1294.0	604.0	8.466052	2.721088

642 rows × 6 columns

In [13]:
```
user_state = input("Enter the Jurisdiction you would like to view: ").upper()
sns.lineplot(data=df.loc[df.Jurisdiction == user_state,:], x="Year", y="pc_Salesperson_YoY").set_title(f"% C
```

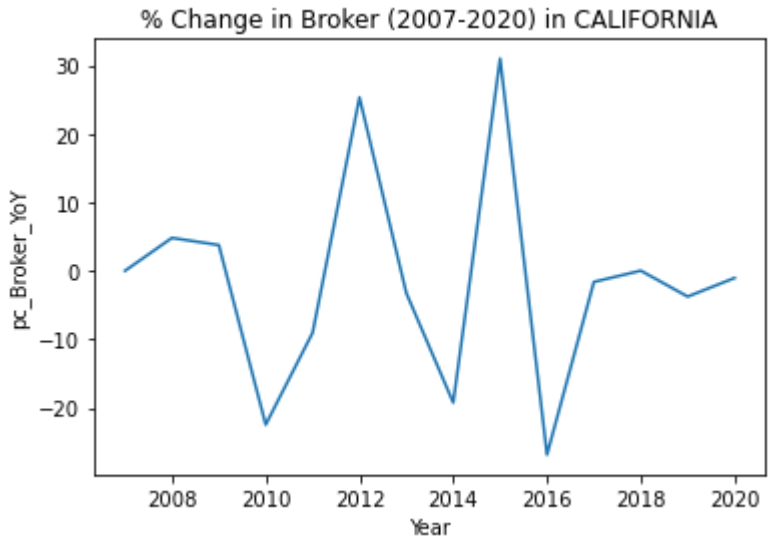
```
Enter the Jurisdiction you would like to view: california
```

Out[13]:
```
Text(0.5, 1.0, '% Change in Salesperson (2007-2020) in CALIFORNIA')
```



In [14]:
```
sns.lineplot(data=df.loc[df.Jurisdiction == user_state,:], x="Year", y="pc_Broker_YoY").set_title(f"% Change
```

Out[14]: Text(0.5, 1.0, '% Change in Broker (2007-2020) in CALIFORNIA')



In [12]:
```
df.to_csv("temp_df_states_per_change.csv",index = None)
```

Median Homes Sales Component

In [26]:
```
excel_file_path = 'Median Home Sale Price.xls'
df1 = pd.read_excel(excel_file_path)
df1.head()
```

Out[26]:

	Jurisdiction	2007-01-01 00:00:00	2007-02-28 00:00:00	2007-03-31 00:00:00	2007-04-30 00:00:00	2007-05-31 00:00:00	2007-06-30 00:00:00	2007-07-31 00:00:00	2007-08-31 00:00:00	2007-09-30 00:00:00	...	2019-12-31 00:00:00	2020-01-31 00:00:00	2020-00:00
0	Alabama	135012	135570	136121	136750	136989	137107	137079	137185	137264	...	147354	148117	148
1	Alaska	248720	249896	250758	251804	252725	253638	254392	254903	255547	...	298243	298195	296
2	Arizona	266963	266026	265318	264712	263735	262040	259995	258243	256060	...	266925	268908	270
3	Arkansas	123002	123190	123293	123425	123469	123545	123590	123789	123876	...	145843	146199	146
4	California	511756	509836	507675	505468	502119	498500	494091	489461	483965	...	554229	556903	560

5 rows × 166 columns

In [27]:
```
## Transpose df1

df1 = df1.T.reset_index()
```

In [28]:
```
## get the zeroth row to assign it as columns

df1.loc[0,:].values.tolist()
```

Out[28]:
```
['Jurisdiction',
 'Alabama',
 'Alaska',
 'Arizona',
 'Arkansas',
 'California',
 'Colorado',
 'Connecticut',
 'Delaware',
 'District of Columbia',
 'Florida',
 'Georgia',
 'Hawaii',
 'Idaho',
 'Illinois',
 'Indiana',
 'Iowa',
 'Kansas',
 'Kentucky',
 'Louisiana',
 'Maine',
 'Maryland',
 'Massachusetts',
 'Michigan',
 'Minnesota',
 'Mississippi',
 'Missouri',
 'Montana',
 'Nebraska',
 'Nevada',
 'New Hampshire',
 'New Jersey',
 'New Mexico',
 'New York',
 'North Carolina',
 'North Dakota',
 'Ohio',
 'Oklahoma',
```

```
 'Oregon',
 'Pennsylvania',
 'Rhode Island',
 'South Carolina',
 'South Dakota',
 'Tennessee',
 'Texas',
 'Utah',
 'Vermont',
 'Virginia',
 'Washington',
 'West Virginia',
 'Wisconsin',
 'Wyoming']
```

In [30]:

```
df1  = df1.loc[1:,:]
df1.Jurisdiction = pd.DatetimeIndex(df1['Jurisdiction']).year
df1 = df1.astype('int64')
df1 = df1.groupby('Jurisdiction').mean().reset_index() ## Average of all month in a given year
df1 = df1.astype('int64')
df_new = pd.DataFrame(columns = ['State','Year','Median_Sale_Price'])
df1.index = df1['Jurisdiction']
df1 = df1.iloc[:,1:]

for idx,row in df1.iterrows():
    for value, column in zip(row,df1.columns):
        df_new = df_new.append({'State':column,'Year':idx,'Median_Sale_Price':value},ignore_index=True)

df_new.sort_values(by = ['State'],inplace = True )
states = df_new.State.unique().tolist()
```

In [31]:

```
for state in states:
    df_new.loc[df_new.State == str(state),'pc_Sales_YoY'] = df_new.loc[df_new.State == str(state),'Median_Sa
```

In [32]:
```
df_new.fillna(value = 0,inplace= True)
df_new
```

Out[32]:

	State	Year	Median_Sale_Price	pc_Sales_YoY
0	Alabama	2007	136753	0.000000
459	Alabama	2016	126912	-7.196186
408	Alabama	2015	123679	-2.547434
561	Alabama	2018	137316	11.026124
357	Alabama	2014	121027	-11.862420
...	...	...	...	...
152	Wyoming	2009	201126	1.560835
101	Wyoming	2008	204967	1.909748
50	Wyoming	2007	201771	-1.559275
662	Wyoming	2019	248000	22.911618
713	Wyoming	2020	257636	3.885484

714 rows × 4 columns

In [33]:
```
user_state = input("Enter the Jurisdiction you would like to view: ").capitalize()
sns.lineplot(data=df_new.loc[df_new.State == user_state,:], x="Year", y="pc_Sales_YoY").set_title(f"% Change
```

```
Enter the Jurisdiction you would like to view: alabama
```

Out[33]:
```
Text(0.5, 1.0, '% Change in Median home sale price (2007-2020) in Alabama')
```