Requisite Packages - Below I bring in the packages I need

In [423]:
```
import pandas as pd # Installing Pandas
import requests # This is usefull with the API
import numpy as np # For performing numerical analysis
import matplotlib.pyplot as plt # Plotting

import seaborn as sns # Cool decals
```

In [2]:
```
url = "/Users/sample/Desktop/community-banking-structure-reference-data.csv" # We are assigning our CSV file data

# This information is only for community banks and we will need to merge information on its acquirers later on
```

Dataset contains information only on community banks. It does not provide an in depth examination of community banks hence, for these two reasons we will need to upload a complimentary file

In [3]:
```
FDIC = pd.read_csv(url) # Reading the file
```

In [4]:
```
FDIC
```

Out[4]:

	CERT	PROXIMATE_CERT	RPT_START_TYPE	RPT_START_YR	RPT_START_QTR	RPT_S
0	59140	NaN	1.0	2018.0	12.0	
1	59149	NaN	1.0	2018.0	12.0	
2	59108	NaN	1.0	2018.0	9.0	
3	59112	NaN	1.0	2018.0	6.0	
4	59114	NaN	1.0	2018.0	6.0	
5	59098	NaN	1.0	2018.0	3.0	
6	59099	NaN	1.0	2018.0	3.0	
7	59106	NaN	1.0	2018.0	3.0	
8	59101	NaN	1.0	2017.0	12.0	
9	59104	NaN	1.0	2017.0	9.0	

10	59105	NaN	1.0	2017.0	9.0
11	59093	NaN	1.0	2017.0	3.0
12	59094	NaN	1.0	2017.0	3.0
13	59086	NaN	1.0	2015.0	9.0
14	59074	NaN	1.0	2013.0	12.0
15	59070	NaN	1.0	2013.0	9.0
16	59026	NaN	1.0	2011.0	6.0
17	59060	NaN	1.0	2011.0	6.0
18	59028	NaN	1.0	2011.0	3.0
19	58952	NaN	1.0	2010.0	12.0
20	59015	NaN	1.0	2010.0	12.0
21	59052	NaN	1.0	2010.0	12.0
22	58961	NaN	1.0	2010.0	9.0
23	59004	NaN	1.0	2010.0	9.0
24	59017	NaN	1.0	2010.0	9.0
25	59049	NaN	1.0	2010.0	9.0
26	59051	NaN	1.0	2010.0	9.0
27	58991	NaN	1.0	2010.0	3.0
28	59010	NaN	1.0	2010.0	3.0
29	59030	NaN	1.0	2010.0	3.0
...	...	...	...	...	...
23170	22227	4977.0	NaN	NaN	NaN
23171	22303	9282.0	NaN	NaN	NaN
23172	22417	20450.0	NaN	NaN	NaN
23173	22566	12229.0	NaN	NaN	NaN

23174	22568	16877.0	NaN	NaN	NaN
23175	22744	0.0	NaN	NaN	NaN
23176	22791	2793.0	NaN	NaN	NaN
23177	22848	3432.0	NaN	NaN	NaN
23178	22861	23337.0	NaN	NaN	NaN
23179	23198	12229.0	NaN	NaN	NaN
23180	23335	25124.0	NaN	NaN	NaN
23181	23362	0.0	NaN	NaN	NaN
23182	23596	4979.0	NaN	NaN	NaN
23183	23656	24202.0	NaN	NaN	NaN
23184	24098	5250.0	NaN	NaN	NaN
23185	24758	0.0	NaN	NaN	NaN
23186	27961	29080.0	NaN	NaN	NaN
23187	29184	27920.0	NaN	NaN	NaN
23188	29898	29281.0	NaN	NaN	NaN

23189	29908	28736.0	NaN	NaN	NaN
23190	30227	28894.0	NaN	NaN	NaN
23191	30283	32068.0	NaN	NaN	NaN
23192	30304	17721.0	NaN	NaN	NaN
23193	30525	27878.0	NaN	NaN	NaN
23194	30798	32354.0	NaN	NaN	NaN
23195	30911	31244.0	NaN	NaN	NaN
23196	31038	28624.0	NaN	NaN	NaN
23197	31318	0.0	NaN	NaN	NaN
23198	31383	32208.0	NaN	NaN	NaN
23199	31847	31881.0	NaN	NaN	NaN

23200 rows × 18 columns

In [5]:
```
FDIC.head() # Gives us the first few entries of the data set
```

Out[5]:

	CERT	PROXIMATE_CERT	RPT_START_TYPE	RPT_START_YR	RPT_START_QTR	RPT_STAR
0	59140	NaN	1.0	2018.0	12.0	
1	59149	NaN	1.0	2018.0	12.0	
2	59108	NaN	1.0	2018.0	9.0	
3	59112	NaN	1.0	2018.0	6.0	
4	59114	NaN	1.0	2018.0	6.0	

In [6]:
```
FDIC.tail() # Gives us the last few entries of the data set
```

Out[6]:

	CERT	PROXIMATE_CERT	RPT_START_TYPE	RPT_START_YR	RPT_START_QTR	RPT_S
23195	30911	31244.0	NaN	NaN	NaN	
23196	31038	28624.0	NaN	NaN	NaN	
23197	31318	0.0	NaN	NaN	NaN	
23198	31383	32208.0	NaN	NaN	NaN	
23199	31847	31881.0	NaN	NaN	NaN	

23,200 - Total amount of banks that have been in existence in the U.S from 1984 to 2018

In [7]:
```
FDIC.shape #23,200 Rows - 18 Columns
```

Out[7]: (23200, 18)

PROXIMATE_CERT >>> MERG >>> The CERT of the acquiring institution.

Below we have filtered the column "PROXIMATE_CERT" by value. What this has done is filter the column that pulls information by amount of overall acqusitions in the banking industry. This information tells us that these certification numbers (unique FDIC certificate number for the insured institution) have acquired the largest amount of institutions since 1984.

In [8]:
```
FDIC.PROXIMATE_CERT.value_counts() # Counts the data that is most frequent and provides us with the amount - in this case it is the 'CERT' column
```

Out[8]:
```
0.0        607
12368.0    108
9846.0      87
27306.0     70
4979.0      65
3511.0      62
24344.0     55
849.0       55
3011.0      51
6548.0      49
3263.0      47
867.0       45
1020.0      45
9609.0      43
15802.0     42
4297.0      41
27476.0     41
11063.0     38
16835.0     38
993.0       36
33184.0     36
16571.0     36
5208.0      35
2558.0      34
14533.0     34
6560.0      34
873.0       33
3618.0      33
3566.0      33
11813.0     33
           ...
5181.0       1
35591.0      1
327.0        1
402.0        1
33889.0      1
13610.0      1
58467.0      1
```

```
33354.0      1
27920.0      1
26882.0      1
18613.0      1
32893.0      1
30255.0      1
16120.0      1
10007.0      1
16908.0      1
32890.0      1
33881.0      1
22560.0      1
15126.0      1
27623.0      1
26769.0      1
58744.0      1
3923.0       1
29227.0      1
15194.0      1
19899.0      1
32298.0      1
28517.0      1
30182.0      1
Name: PROXIMATE_CERT, Length: 6436, dtype: int64
```

The possible merger options according to the data set are:

“0” - (or missing) - no involvement in a merger

“1” - pooling-of-interests accounting method

“2” - acquisition (also called “purchase”) accounting method

“3” - acquisition (“purchase”) accounting method for an acquired failed institution

In [9]:
```
FDIC.MergMethDesc.value_counts() # Counts the data that is most freque
nt and provides us with the amount - in this case it is the 'CERT' col
umn
```

Out[9]:
```
1 - POOLING OF INTEREST ACCOUNTING METHOD    8704
2 - PURCHASE ACCOUNTING METHOD               5775
3 - ACQUIRED FAILED INSTITUTION              2712
0 - NO INVOLVMENT IN A MERGER                 606
Name: MergMethDesc, dtype: int64
```

These are the 8 banks with the largest amount of acqusitions since 1984 and the details of all of the acqusitions that they have made.

In [10]:
```
FDIC[FDIC.PROXIMATE_CERT == 12368] # Shows us the values that contain that specific banks 'CERT' number in column 'Proximate_CERT'
```

Out[10]:

	CERT	PROXIMATE_CERT	RPT_START_TYPE	RPT_START_YR	RPT_START_QTR	RPT_S
8309	57017	12368.0	NaN	NaN	NaN	
8491	35469	12368.0	NaN	NaN	NaN	
9040	26800	12368.0	NaN	NaN	NaN	
9457	35162	12368.0	NaN	NaN	NaN	
9485	4979	12368.0	NaN	NaN	NaN	
10401	20577	12368.0	NaN	NaN	NaN	
10478	23883	12368.0	NaN	NaN	NaN	
10483	24939	12368.0	NaN	NaN	NaN	
10608	33480	12368.0	NaN	NaN	NaN	
11134	21269	12368.0	NaN	NaN	NaN	
11153	26637	12368.0	NaN	NaN	NaN	

11257	8806	12368.0	NaN	NaN	NaN
11378	4937	12368.0	NaN	NaN	NaN
11406	13658	12368.0	NaN	NaN	NaN
11407	13659	12368.0	NaN	NaN	NaN
11616	45	12368.0	NaN	NaN	NaN
11620	1375	12368.0	NaN	NaN	NaN
11644	6094	12368.0	NaN	NaN	NaN
11711	27436	12368.0	NaN	NaN	NaN
11733	1290	12368.0	NaN	NaN	NaN
11760	10525	12368.0	NaN	NaN	NaN
11765	11251	12368.0	NaN	NaN	NaN
11783	16155	12368.0	NaN	NaN	NaN
11823	26214	12368.0	NaN	NaN	NaN
11854	33577	12368.0	NaN	NaN	NaN

11892	10071	12368.0	NaN	NaN	NaN
11949	27097	12368.0	NaN	NaN	NaN
11973	32706	12368.0	NaN	NaN	NaN
12080	25015	12368.0	NaN	NaN	NaN
12122	33027	12368.0	NaN	NaN	NaN
...	...	...	...	...	...
20316	55	12368.0	NaN	NaN	NaN
20355	9052	12368.0	NaN	NaN	NaN
20813	2772	12368.0	NaN	NaN	NaN
21304	16373	12368.0	NaN	NaN	NaN
21691	1734	12368.0	NaN	NaN	NaN
21933	19533	12368.0	NaN	NaN	NaN
21941	20751	12368.0	NaN	NaN	NaN

22003	854	12368.0	NaN	NaN	NaN
22017	2836	12368.0	NaN	NaN	NaN
22222	20322	12368.0	NaN	NaN	NaN
22285	75	12368.0	NaN	NaN	NaN
22293	1741	12368.0	NaN	NaN	NaN
22299	2826	12368.0	NaN	NaN	NaN
22300	2834	12368.0	NaN	NaN	NaN
22369	18686	12368.0	NaN	NaN	NaN
22396	21504	12368.0	NaN	NaN	NaN
22404	22603	12368.0	NaN	NaN	NaN
22450	66	12368.0	NaN	NaN	NaN
22581	71	12368.0	NaN	NaN	NaN

22582	78	12368.0	NaN	NaN	NaN
22590	2775	12368.0	NaN	NaN	NaN
22591	2833	12368.0	NaN	NaN	NaN
22602	5611	12368.0	NaN	NaN	NaN
22622	12069	12368.0	NaN	NaN	NaN
22644	17367	12368.0	NaN	NaN	NaN
22645	17397	12368.0	NaN	NaN	NaN
22661	19370	12368.0	NaN	NaN	NaN
22662	19455	12368.0	NaN	NaN	NaN
22720	2806	12368.0	NaN	NaN	NaN
22721	2816	12368.0	NaN	NaN	NaN

108 rows × 18 columns

In [ ]:
```
FDIC[FDIC.PROXIMATE_CERT == 9846] # Shows us the values that contain t
hat specific banks 'CERT' number in column 'Proximate_CERT'
```

In [ ]:
```
FDIC[FDIC.PROXIMATE_CERT == 27306] # Shows us the values that contain
that specific banks 'CERT' number in column 'Proximate_CERT'
```

In [ ]:
```
FDIC[FDIC.PROXIMATE_CERT == 4979] # Shows us the values that contain t
hat specific banks 'CERT' number in column 'Proximate_CERT'
```

In [ ]:
```
FDIC[FDIC.PROXIMATE_CERT == 3511] # Shows us the values that contain t
hat specific banks 'CERT' number in column 'Proximate_CERT'
```

In [ ]:
```
FDIC[FDIC.PROXIMATE_CERT == 24344] # Shows us the values that contain
that specific banks 'CERT' number in column 'Proximate_CERT'
```

In [ ]:
```
FDIC[FDIC.PROXIMATE_CERT == 849] # Shows us the values that contain th
at specific banks 'CERT' number in column 'Proximate_CERT'
```

In [ ]:
```
FDIC[FDIC.PROXIMATE_CERT == 3011] # Shows us the values that contain t
hat specific banks 'CERT' number in column 'Proximate_CERT'
```

These are some summary statistics about the data

In [11]:
```
FDIC.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 23200 entries, 0 to 23199
Data columns (total 18 columns):
CERT                23200 non-null int64
PROXIMATE_CERT      17797 non-null float64
RPT_START_TYPE      5360 non-null float64
RPT_START_YR        5360 non-null float64
RPT_START_QTR       5360 non-null float64
RPT_START_CALLYM    5360 non-null float64
RPT_STOP_TYPE       17840 non-null float64
RPT_STOP_YR         17840 non-null float64
RPT_STOP_QTR        17840 non-null float64
RPT_STOP_CALLYM     17840 non-null float64
ULTIMATE_CERT       17797 non-null float64
FAIL_YR             2819 non-null float64
FAIL_QTR            2819 non-null float64
FAIL_DATE           2819 non-null float64
MergMeth            17797 non-null float64
MergMethDesc        17797 non-null object
RUN_DATE            23200 non-null int64
Key                 23200 non-null int64
dtypes: float64(14), int64(3), object(1)
memory usage: 3.2+ MB
```

In [ ]:
```
FDIC[FDIC.PROXIMATE_CERT == 9846] # Shows us the values that contain t
hat specific banks 'CERT' number in column 'Proximate_CERT'
```

We now drop the columns that we will not be working with. There is no need to filter information by quarter since we have the exact date and year column with necessary information. Moreover, the run date will be the same for all data entries and does not affect our data exploration.

In [12]:
```
FDIC = FDIC.drop(['RPT_START_QTR', 'FAIL_QTR', 'RUN_DATE', 'RPT_START_
CALLYM', 'RPT_STOP_QTR', 'RPT_STOP_CALLYM'], axis=1)

# Drops the columns that we do not need
```

I have removed all of the "NaN" data inputs and have replaced them with "" or blanks. This is for viewing pleasure and easier to identify information.

In [13]:
```
FDIC = FDIC.replace(np.nan, '', regex=True)

# The replace function allows us to change one set of values to anothe
r. In this case we replace NaN's to blanks
```

Below I have filtered for the institutions that have not stopped reporting a Call Report or TFR. "RPT_STOP_YR" column is the last year for which there is financial statement data meaning that columns with a blank or "" entry in the column are still in existence.

FDIC STATUS = ACTIVE

There is a total of 5360 total institutions currrently active.

In [14]:
```
FDIC[FDIC.RPT_STOP_YR == ''] # Taking a look at all of the institution
s that stopped reporting
```

Out[14]:

	CERT	PROXIMATE_CERT	RPT_START_TYPE	RPT_START_YR	RPT_STOP_TYPE	RPT_ST
0	59140		1	2018		
1	59149		1	2018		
2	59108		1	2018		
3	59112		1	2018		
4	59114		1	2018		
5	59098		1	2018		
6	59099		1	2018		
7	59106		1	2018		
8	59101		1	2017		
9	59104		1	2017		
10	59105		1	2017		
11	59093		1	2017		
12	59094		1	2017		
13	59086		1	2015		
14	59074		1	2013		
15	59070		1	2013		
16	59026		1	2011		
17	59060		1	2011		

18	59028		1	2011	
19	58952		1	2010	
20	59015		1	2010	
21	59052		1	2010	
22	58961		1	2010	
23	59004		1	2010	
24	59017		1	2010	
25	59049		1	2010	
26	59051		1	2010	
27	58991		1	2010	
28	59010		1	2010	
29	59030		1	2010	
...	...	...	...	...	...
5330	25287		1	1984	
5331	25291		1	1984	
5332	25292		1	1984	
5333	25293		1	1984	
5334	25300		1	1984	
5335	25308		1	1984	
5336	25326		1	1984	
5337	25327		1	1984	
5338	30060		2	1984	
5339	32203		2	1984	
5340	32204		3	1984	
5341	32206		1	1984	
5342	32207		2	1984	
5343	32208		1	1984	
5344	32209		1	1984	
5345	32210		1	1984	
5346	32214		1	1984	
5347	32216		1	1984	
5348	32217		1	1984	

5349	32218	1	1984
5350	32219	1	1984
5351	32221	1	1984
5352	32222	1	1984
5353	32223	1	1984
5354	32224	1	1984
5355	32225	1	1984
5356	32226	1	1984
5357	32228	1	1984
5358	32229	1	1984
5359	32230	1	1984

5360 rows × 12 columns

Below is a list of all of the banking institutions that were reporting a Call Report or TFR prior to 1984 (they had blank input in the "RPT_START_YR") which means that they were already active in 1984.

The total number of active institutions at that point in time was 17,840.

In [15]:
```
FDIC[FDIC.RPT_START_YR == ''] # Taking a look at all of the institutio
ns that started reporting
```

Out[15]:

	CERT	PROXIMATE_CERT	RPT_START_TYPE	RPT_START_YR	RPT_STOP_TYPE	RPT_S
5360	266	17735			3	
5361	850	58255			3	
5362	1414	3832			3	
5363	1757	29700			3	
5364	2526	90213			3	

5365	2615	9712	3
5366	3145	22327	3
5367	4019	32629	3
5368	5190	15478	3
5369	7459	7551	2
5370	7609	713	3
5371	7774	7551	2
5372	7953	7875	3
5373	8097	17993	3
5374	8354	20130	3
5375	8472	1331	2
5376	8526	58228	3
5377	9042	14816	3
5378	9950	16958	3
5379	10179	1409	3

5380	10559	15684			3
5381	10621	28332			3
5382	11023	5496			3
5383	11286	34217			3
5384	11483	13868			3
5385	11886	3250			3
5386	12104	11329			3
5387	12515	11063			3
5388	13567	9463			3
5389	13585	58076			2
...	...	...	...	...	...
23170	22227	4977			3
23171	22303	9282			3
23172	22417	20450			3
23173	22566	12229			3

23174	22568	16877	1
23175	22744	0	1
23176	22791	2793	3
23177	22848	3432	1
23178	22861	23337	1
23179	23198	12229	3
23180	23335	25124	1
23181	23362	0	1
23182	23596	4979	3
23183	23656	24202	3
23184	24098	5250	1
23185	24758	0	4
23186	27961	29080	3
23187	29184	27920	5
23188	29898	29281	3

23189	29908	28736	3
23190	30227	28894	3
23191	30283	32068	5
23192	30304	17721	3
23193	30525	27878	3
23194	30798	32354	3
23195	30911	31244	3
23196	31038	28624	3
23197	31318	0	4
23198	31383	32208	5
23199	31847	31881	3

17840 rows × 12 columns

In [16]:
```
FDIC
```

Out[16]:

	CERT	PROXIMATE_CERT	RPT_START_TYPE	RPT_START_YR	RPT_STOP_TYPE	RPT_S
0	59140		1	2018		
1	59149		1	2018		

2	59108		1	2018	
3	59112		1	2018	
4	59114		1	2018	
5	59098		1	2018	
6	59099		1	2018	
7	59106		1	2018	
8	59101		1	2017	
9	59104		1	2017	
10	59105		1	2017	
11	59093		1	2017	
12	59094		1	2017	
13	59086		1	2015	
14	59074		1	2013	
15	59070		1	2013	
16	59026		1	2011	
17	59060		1	2011	
18	59028		1	2011	
19	58952		1	2010	
20	59015		1	2010	
21	59052		1	2010	
22	58961		1	2010	
23	59004		1	2010	
24	59017		1	2010	
25	59049		1	2010	
26	59051		1	2010	
27	58991		1	2010	
28	59010		1	2010	
29	59030		1	2010	
...	...	...	...	...	...
23170	22227	4977			3

23171	22303	9282	3
23172	22417	20450	3
23173	22566	12229	3
23174	22568	16877	1
23175	22744	0	1
23176	22791	2793	3
23177	22848	3432	1
23178	22861	23337	1
23179	23198	12229	3
23180	23335	25124	1
23181	23362	0	1
23182	23596	4979	3
23183	23656	24202	3
23184	24098	5250	1
23185	24758	0	4

23186	27961	29080	3
23187	29184	27920	5
23188	29898	29281	3
23189	29908	28736	3
23190	30227	28894	3
23191	30283	32068	5
23192	30304	17721	3
23193	30525	27878	3
23194	30798	32354	3
23195	30911	31244	3
23196	31038	28624	3
23197	31318	0	4
23198	31383	32208	5
23199	31847	31881	3

23200 rows × 12 columns

In [17]:
```
len(FDIC) # Length of our data set
```

Out[17]: 23200

Dramatic difference between 1984 and 2018.

More than (17840-5360) **12480** banks stopped reporting. We have the information to understand why they no longer exist.

The reason the institution stopped reporting. Possible values are:

“1” - failure,

“2” - merger without assistance, consolidation,

“3” - merger, acquisition (approximately 3.03% of this population received government assistance 180 days or less before the RPT_STOP_CALLYM),

“4” - self-liquidation, or

“5” - other. "

We will use this as a foundation to create a new number of charts to show the consolidation within the industry.

Now we will load the second data set that we will be working with. This data set is the universe of all banks that have been in existence with a greater detail of information which will include geographical location, certification numbers, financial metrics and key management decisions.

Here we have not only the information on the acquirers of community banks but also more detailed metrics on community banks themselves

In [18]:
```
universe = pd.read_csv("/Users/sample/Downloads/Universe of all Institutions - Universe of all Institutions.csv", low_memory=False) # Reading the file
```

In [19]:
```
universe
```

Out[19]:

	STNAME	CERT	ACTIVE	ADDRESS	ASSET	BKCLASS	CHANGEC1	CH
0	Colorado	91393	0	4100 East Mississippi Avenue	84,896	SM	223	
1	New Jersey	91363	0	800 Scudders Mill Road	9,520,237	NM	223	
2	Colorado	91327	0	6312 South Fiddlers Green Circle 270n	585,870	NM	223	
3	Georgia	91325	1	3290 Northside Parkway	234,794	N	510	
4	Colorado	91324	0	6312 South Fiddlers Green Circle	383,629	NM	223	
5	Colorado	91322	0	717 17th Street	1,640,488	NM	240	
6	Florida	91315	0	180 Royal Palm Way	4,046	N	223	
7	North Carolina	91309	0	301 North Elm Street, Suite 900	124,815	NM	223	
8	Arkansas	91280	1	200 North State Street	203,404	SM	412	
9	Colorado	91260	0	2271 North Frontage Road West	6,825	NM	223	
10	Florida	91247	0	777 South Flagler Drive, Suite 140-E	129,695	SM	223	
11	Colorado	91235	0	2725 Iris Avenue	4,434	NM	223	
12	Colorado	91212	0	580 Twenty-Four And One-Half Road	2,187	NM	223	
13	Colorado	91189	0	2950 Pearl Street	233,223	NM	211	
14	Colorado	91188	0	12131 East Iliff Avenue, Unit D	10,727	NM	223	
15	Colorado	91151	0	19590 East Main Street, Suite 101	61,935	SM	223	
				12790 West				

16	Colorado	91129	0	Alameda Parkway, Unit A	14,249	NM	223
17	Colorado	91123	0	601 Court Street	2,566	NM	223
18	Colorado	91098	0	825 Citadel Drive East	22,993	NM	223
19	Colorado	91079	0	1345 South Pueblo Boulevard	4,915	NM	223
20	Colorado	91075	0	2545 Youngfield Street	4,669	NM	223
21	Colorado	91048	0	119 First Street	55,734	NM	223
22	Colorado	91005	1	Peterson Air Force Base, 455 W. Paine St., Bld...	209,438	NM	520
23	Colorado	90958	0	15405 East Iliff Avenue	20,418	NM	223
24	Colorado	90885	0	7843 Wadsworth Boulevard	13,348	NM	223
25	Colorado	90713	0	405 Colorado Avenue	21,042	SM	223
26	Colorado	90666	0	1776 South Nevada Avenue	14,535	NM	223
27	Colorado	90637	0	117 West Elm	3,976	SM	221
28	Colorado	90634	0	3600 East Alameda Parkway, Suite 100	372,671	NM	223
29	Colorado	90582	0	501 Wilcox	145,037	SM	223
...	...	...	...	...	...	...	...
27579	Alabama	50	1	200 4th Avenue Sw	148,325	NM	810
27580	Alabama	49	1	146 West Front Street	62,181	NM	412
27581	Alabama	48	0	Main Street	7,741	NM	223
27582							

	Alabama	47	1	819 20th St	358,656	NM	660
27583	Alabama	46	1	900 Second Avenue, Sw	284,098	NM	660
27584	Alabama	45	0	510 East Laurel Street	173,273	NM	223
27585	Alabama	43	0	Depot Street	12,825	NM	223
27586	Alabama	42	0	U.S. Highway 231 North & 12th Avenue	32,225	NM	223
27587	Alabama	41	1	801 13th Street	184,522	NM	412
27588	Alabama	40	0	801 East Broad Street	43,800	NM	213
27589	Alabama	39	1	216 North Walnut Avenue	304,453	NM	412
27590	Alabama	38	0	923 Attalla Boulevard	63,927	NM	223
27591	Alabama	37	0	2608 Schuler Avenue	61,089	NM	223
27592	Alabama	36	0	17 North Court Square	56,420	NM	223
27593	Alabama	35	1	100 North Gay Street	818,917	SM	525
27594	New Hampshire	31	0	160 Main Street	204,423	NM	223
27595	Massachusetts	30	0	35 Church Street	23,379	NM	223
27596	Massachusetts	28	0	217 Essex Street	12,474	NM	223
27597	Massachusetts	27	0	1495 Hancock Street	414,795	NM	223
27598	Massachusetts	22	0	324 Main Street	159,727	NM	223
27599	Massachusetts	21	0	154 Main Street	153,274	SM	223
27600	Massachusetts	20	0	10 North Main Street	346,808	NM	223
27601	Massachusetts	18	0	1414 Massachusetts	1,435,310	NM	223

				Avenue			
27602	Massachusetts	15	0	30 Court Street	NaN	SM	223
27603	Massachusetts	14	1	1 Lincoln St. Fl 1	242,037,871	SM	0
27604	Maine	10	0	8 Washington Street	55,692	SM	223
27605	Maine	9	0	66 Main Street	539,169	SM	223
27606	Maine	8	0	One City Center	1,699,404	SM	223
27607	Connecticut	6	0	81 West Main Street	624,655	NM	223
27608	Connecticut	4	0	22 Main Street	48,570	NM	223

27609 rows × 35 columns

In [20]:
```
universe.head()
```

Out[20]:

	STNAME	CERT	ACTIVE	ADDRESS	ASSET	BKCLASS	CHANGEC1	CHANGEC2	CHAN
0	Colorado	91393	0	4100 East Mississippi Avenue	84,896	SM	223	0	
1	New Jersey	91363	0	800 Scudders Mill Road	9,520,237	NM	223	0	
2	Colorado	91327	0	6312 South Fiddlers Green Circle 270n	585,870	NM	223	0	
3	Georgia	91325	1	3290 Northside Parkway	234,794	N	510	0	
4	Colorado	91324	0	6312 South Fiddlers Green Circle	383,629	NM	223	0	

5 rows × 35 columns

In [21]:
```
len(universe)
```

Out[21]: 27609

We need to clean up the data set since some of the columns we will not be working with. Using the drop function we have eliminated a number of columns from our data set. universe = universe.drop([''], axis=1) "DOCKET", CHARTER CHRTAGNT CONSERVE CLCODE CMSA_NO CMSA DATEUPDT DENOVO EFFDATE FDICDBS FDICREGN FDICSUPV FED FED_RSSD FEDCHRTR FLDOFF IBA INSAGNT1 INSAGNT2 INSDATE INSBIF INSCOML INSDIF INSFDIC INSSAIF INSSAVE MSA_NO MSA NEWCERT OAKAR OTSDIST OTSREGNM PROCDATE QBPRCOML REGAGNT REPDTE RISDATE STCHRTR ROAQ ROEQ RUNDATE SASSER Law_Sasser_Flag STNUM TE01N528 TE02N528 TE03N528 TE04N528 TE05N528 TE06N528 TE07N528 TE08N528 TE09N528 TE10N528 TE01N529 TE02N529 TE03N529 TE04N529 TE05N529 TE06N529 ZIP SUPRV_FD OCCDIST UNINUM CFPBEFFDTE CFPBENDDTE CFPBFLAG REGAGENT2 OFFICES CERTCONS PARCERT CITYHCR FORM31 HCTMULT HCTMULT INSTAG NAMEHCR OFFOA RSSDHCR STALPHCR STMULT SUBCHAPS ROAPTXQ TRUST SPECGRPN TRACT CSA CSA_NO CSA_FLG CBSA CBSA_NO CBSA_METRO_NAME CBSA_METRO CBSA_METRO_FLG CBSA_MICRO_FLG CBSA_DIV CBSA_DIV_NO CBSA_DIV_FLG "CB"

Now we need to merge both data sets. Our "FDIC" data set contains the entities that we want to be working with. The "universe" data sets has a record of all banks, including subsidiaries. Moreover, the "universe" data set has the company name or official name of the bank for each 'CERT' number. Hence, we are merging on this column "CERT".

Knowing that the "universe" data set contains all institutions, including ones that are included within the "FDIC" data set, we would like to:

```
1. Match all unique FDIC certificate numbers in the "FDIC" data set with t
heir corresponding public names
2. Provide additional information/description for the institutions in the
"FDIC" data set. This information has been selected from the "universe" da
ta set (reason why we deleted the large # of rows in the cell above).
3. We will then clean up and group the data by changing the order of the c
olumns so it is aesthetically pleasing and easier to work with.
```

In [22]:
```
combo = pd.merge(FDIC, universe, how='outer', left_on='CERT', right_on
='CERT', indicator=True)

# We merge our two data sets - We use the 'CERT' column as the placeme
nt of the merge.
```

In [23]:
```
combo
```

Out[23]:

	CERT	PROXIMATE_CERT	RPT_START_TYPE	RPT_START_YR	RPT_STOP_TYPE	RPT_S
0	59140		1	2018		
1	59149		1	2018		
2	59108		1	2018		
3	59112		1	2018		
4	59114		1	2018		
5	59098		1	2018		
6	59099		1	2018		
7	59106		1	2018		
8	59101		1	2017		
9	59104		1	2017		
10	59105		1	2017		
11	59093		1	2017		
12	59094		1	2017		
13	59094	57903			3	
14	59086		1	2015		
15	59074		1	2013		
16	59070		1	2013		
17	59026		1	2011		
18	59026	4999			3	
19	59060		1	2011		
20	59060	57870			3	
21	59028		1	2011		
22	59028	0			4	
23	58952		1	2010		

24	59015		1	2010	
25	59015	59052			3
26	59052		1	2010	
27	58961		1	2010	
28	59004		1	2010	
29	59004	27599			3
...	...	...	...	...	...
31861	93	NaN	NaN	NaN	NaN
31862	90	NaN	NaN	NaN	NaN
31863	89	NaN	NaN	NaN	NaN
31864	88	NaN	NaN	NaN	NaN
31865	81	NaN	NaN	NaN	NaN
31866	77	NaN	NaN	NaN	NaN
31867	72	NaN	NaN	NaN	NaN
31868	70	NaN	NaN	NaN	NaN
31869	65	NaN	NaN	NaN	NaN
31870	61	NaN	NaN	NaN	NaN
31871	58	NaN	NaN	NaN	NaN
31872	54	NaN	NaN	NaN	NaN
31873	52	NaN	NaN	NaN	NaN

31874	51	NaN	NaN	NaN	NaN
31875	50	NaN	NaN	NaN	NaN
31876	49	NaN	NaN	NaN	NaN
31877	48	NaN	NaN	NaN	NaN
31878	47	NaN	NaN	NaN	NaN
31879	46	NaN	NaN	NaN	NaN
31880	42	NaN	NaN	NaN	NaN
31881	41	NaN	NaN	NaN	NaN
31882	40	NaN	NaN	NaN	NaN
31883	39	NaN	NaN	NaN	NaN
31884	35	NaN	NaN	NaN	NaN
31885	30	NaN	NaN	NaN	NaN
31886	15	NaN	NaN	NaN	NaN
31887	14	NaN	NaN	NaN	NaN
31888	10	NaN	NaN	NaN	NaN
31889	6	NaN	NaN	NaN	NaN
31890	4	NaN	NaN	NaN	NaN

31891 rows × 47 columns

In [24]:
```
combo.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 31891 entries, 0 to 31890
Data columns (total 47 columns):
CERT              31891 non-null int64
PROXIMATE_CERT    23200 non-null object
```

```
RPT_START_TYPE    23200 non-null object
RPT_START_YR      23200 non-null object
RPT_STOP_TYPE     23200 non-null object
RPT_STOP_YR       23200 non-null object
ULTIMATE_CERT     23200 non-null object
FAIL_YR           23200 non-null object
FAIL_DATE         23200 non-null object
MergMeth          23200 non-null object
MergMethDesc      23200 non-null object
Key               23200 non-null float64
STNAME            31659 non-null object
ACTIVE            31659 non-null float64
ADDRESS           31659 non-null object
ASSET             29750 non-null object
BKCLASS           31659 non-null object
CHANGEC1          31659 non-null float64
CHANGEC2          31659 non-null float64
CHANGEC3          31659 non-null float64
CHANGEC4          31659 non-null float64
CHANGEC5          31659 non-null float64
CITY              31659 non-null object
COUNTY            31659 non-null object
DEP               29750 non-null object
ENDEFYMD          31659 non-null object
EQ                5413 non-null object
ESTYMD            31659 non-null object
INACTIVE          31659 non-null float64
INSTCRCD          31659 non-null float64
NAME              31659 non-null object
ROA               5413 non-null float64
ROE               5413 non-null float64
STALP             31659 non-null object
STCNTY            31659 non-null float64
ULTCERT           31659 non-null float64
WEBADDR           5265 non-null object
DEPDOM            5422 non-null object
MUTUAL            5422 non-null float64
NAMEHCR           4213 non-null object
NETINC            5413 non-null object
NETINCQ           5413 non-null object
OFFDOM            5422 non-null float64
OFFFOR            5422 non-null float64
ROAPTX            5413 non-null float64
SPECGRP           5422 non-null float64
_merge            31891 non-null category
dtypes: category(1), float64(18), int64(1), object(27)
memory usage: 11.5+ MB
```

The merging of both data sets on the "CERT" column (the column that is referenced by both data sets) provides us with a total institution count of 31891.

In [25]:
```
len(combo)
```

Out[25]: 31891

Here we see there are 4282 unique institutions - meaning that this is the total amount of institutions that are on one data set but not the other and vice versa

In [26]:
```
31891-27609
```

Out[26]: 4282

Now that the information has been merged and we have a data that reflects a newly updated "FDIC" data set, we have to remove the other financial institutions that we are not interested in (the institutions that are on the "universe" data set but not on the "FDIC" data set.

This can be done a number of ways, but since I am familiar with both documents, I am aware that the "FDIC" data set has a unique column labeled "Key" which is the unique system generated table key - only applicable to the "FDIC" data set.

Lets make sure that the "Key" column has successfully transferred over when we did the merging.

In [27]:
```
list(combo.columns.values) # Allows us to see what columns we are work
ing with
```

Out[27]:
```
['CERT',
 'PROXIMATE_CERT',
 'RPT_START_TYPE',
 'RPT_START_YR',
 'RPT_STOP_TYPE',
 'RPT_STOP_YR',
 'ULTIMATE_CERT',
 'FAIL_YR',
 'FAIL_DATE',
 'MergMeth',
 'MergMethDesc',
 'Key',
 'STNAME',
 'ACTIVE',
 'ADDRESS',
 'ASSET',
 'BKCLASS',
 'CHANGEC1',
 'CHANGEC2',
 'CHANGEC3',
 'CHANGEC4',
 'CHANGEC5',
 'CITY',
 'COUNTY',
 'DEP',
 'ENDEFYMD',
 'EQ',
 'ESTYMD',
 'INACTIVE',
 'INSTCRCD',
 'NAME',
 'ROA',
 'ROE',
 'STALP',
 'STCNTY',
 'ULTCERT',
 'WEBADDR',
 'DEPDOM',
 'MUTUAL',
 'NAMEHCR',
 'NETINC',
 'NETINCQ',
 'OFFDOM',
 'OFFFOR',
 'ROAPTX',
 'SPECGRP',
 '_merge']
```

We see the 'Key' column labeled under our available columns.

Knowing this information, one of the ways to approach this is by understanding that 'NaN' values would populate in the 'Key' column for all institions that were not part of our "FDIC" data set. Since the "FDIC" data set is the one we want, we could drop all rows (institutions) that do not have a unique system generated key (which instead will have 'NaN' as a value) informaton in that given column.

In [28]:
```
chopped_combo = combo[pd.notnull(combo['Key'])] # What we did here is
erase all rows that do not have a particular value in a specific row -

# we now have a more in depth community bank data set (initial data se
t)
```

In [29]:
```
chopped_combo
```

Out[29]:

	CERT	PROXIMATE_CERT	RPT_START_TYPE	RPT_START_YR	RPT_STOP_TYPE	RPT_S
0	59140		1	2018		
1	59149		1	2018		
2	59108		1	2018		
3	59112		1	2018		
4	59114		1	2018		
5	59098		1	2018		
6	59099		1	2018		
7	59106		1	2018		
8	59101		1	2017		
9	59104		1	2017		
10	59105		1	2017		
11	59093		1	2017		
12	59094		1	2017		
13	59094	57903			3	
14	59086		1	2015		
15	59074		1	2013		

16	59070		1	2013	
17	59026		1	2011	
18	59026	4999			3
19	59060		1	2011	
20	59060	57870			3
21	59028		1	2011	
22	59028	0			4
23	58952		1	2010	
24	59015		1	2010	
25	59015	59052			3
26	59052		1	2010	
27	58961		1	2010	
28	59004		1	2010	
29	59004	27599			3
...	...	...	...	...	...
23170	22227	4977			3
23171	22303	9282			3
23172	22417	20450			3
23173	22566	12229			3
23174	22568	16877			1
23175	22744	0			1
23176	22791	2793			3
23177	22848	3432			1
23178	22861	23337			1
23179	23198	12229			3
23180	23335	25124			1
23181	23362	0			1

23182	23596	4979	3
23183	23656	24202	3
23184	24098	5250	1
23185	24758	0	4
23186	27961	29080	3
23187	29184	27920	5
23188	29898	29281	3
23189	29908	28736	3
23190	30227	28894	3
23191	30283	32068	5
23192	30304	17721	3
23193	30525	27878	3
23194	30798	32354	3
23195	30911	31244	3
23196	31038	28624	3
23197	31318	0	4
23198	31383	32208	5
23199	31847	31881	3

23200 rows × 47 columns

We see that the value of this new altered data set is now equal to 23,200 institutions which is equal to the amount of institutions in the "FDIC" data set.

In [30]:
```
len(chopped_combo)
```

Out[30]:
```
23200
```

In [31]:
```
chopped_combo.shape # We check how many rows and columns the document has
```

Out[31]:
```
(23200, 47)
```

I have run this data set as an excel file and verified the new properities of the new file that I am now working with. This new file labeled "chopped_combo" contains all of the existing and new columns for the initial row (institution) counts. I have saved the document as both a CSV file as well as an XLSX file to verify and keep for my records. This has allowed me to look at the data via a different format as well.

In [ ]:
```
chopped_combo.to_csv(r'/Users/sample/desktop/chopped_and_merged.csv')
# Save it as a CSV on the computer
```

In [ ]:
```
chopped_combo.to_csv(r'/Users/sample/desktop/chopped_and_merged.xlsx')
# Save it as a XLSX on the computer
```

Our third step of the merging and filtering process is grouping together columns from the same information cluster; this includes putting together columns that possess financial metrics, acqusition details and formation information.

We will first set the index to correspond to the official bank name so it is easily identifiable. We do so by setting the index value to the column "Name" that is now included in our data set.

In [32]:
```
chopped_combo.set_index('NAME',inplace=True) # We have changed the ind
ex of our document to the full name of the banks
```

In [33]:
```
chopped_combo.head(10)
```

Out[33]:

NAME	CERT	PROXIMATE_CERT	RPT_START_TYPE	RPT_START_YR	RPT_STOP_TYPI
Gulfside Bank	59140		1	2018	
Generations Commercial Bank	59149		1	2018	
Charles Schwab Trust Bank	59108		1	2018	
Studio Bank	59112		1	2018	
CommerceOne Bank	59114		1	2018	
Infinity Bank	59098		1	2018	
Endeavor Bank	59099		1	2018	
Beacon Community Bank	59106		1	2018	
Tennessee Bank & Trust	59101		1	2017	
The Bank of Austin	59104		1	2017	

10 rows × 46 columns

We will then merge by bank address. This will include columns in order from: "Address", "City", "County" and "STNAME".

This information provides us an overview of the official exact address of the institution that is on file along with the location of the headquarters and the number of total domestic, foreign and total offices.

We will then filter information by any financial metrics that are provdided. This includes columns such as "DEP, "EQ", "ROA", "ROE", "DEPDOM","NETIMC" and "ROAPTX".

Afterwards we will group all of the columns that focus on the merger information as well as the bank origination and closing dates (includes reasoning why they failed).

If you would like to open a checking or savings account, one of the last columns contains the web address for most institutions.

In [34]:
```
list(chopped_combo.columns.values)
```

Out[34]:
```
['CERT',
 'PROXIMATE_CERT',
 'RPT_START_TYPE',
 'RPT_START_YR',
 'RPT_STOP_TYPE',
 'RPT_STOP_YR',
 'ULTIMATE_CERT',
 'FAIL_YR',
 'FAIL_DATE',
 'MergMeth',
 'MergMethDesc',
 'Key',
 'STNAME',
 'ACTIVE',
 'ADDRESS',
 'ASSET',
 'BKCLASS',
 'CHANGEC1',
 'CHANGEC2',
 'CHANGEC3',
 'CHANGEC4',
 'CHANGEC5',
 'CITY',
 'COUNTY',
 'DEP',
 'ENDEFYMD',
 'EQ',
 'ESTYMD',
 'INACTIVE',
 'INSTCRCD',
 'ROA',
 'ROE',
 'STALP',
 'STCNTY',
 'ULTCERT',
 'WEBADDR',
 'DEPDOM',
 'MUTUAL',
 'NAMEHCR',
 'NETINC',
 'NETINCQ',
 'OFFDOM',
 'OFFFOR',
 'ROAPTX',
 'SPECGRP',
 '_merge']
```

In [ ]:
```
clean_combo = chopped_combo[['CERT','PROXIMATE_CERT','ULTIMATE_CERT','
ULTCERT','ADDRESS','CITY','COUNTY','STNAME','STALP','STCNTY','DEPDOM',
         'OFFDOM','OFFFOR','ASSET','DEP','EQ','ROA',
         'ROE','ROAPTX','NETINC','NETINCQ','INSTCRCD','MUTUAL','NAMEHC
R','SPECGRP','RPT_START_TYPE','RPT_START_YR',
         'RPT_START_CALLYM','RPT_STOP_TYPE','RPT_STOP_YR','RPT_STOP_QT
R','RPT_STOP_CALLYM','FAIL_YR','FAIL_DATE','ACTIVE',
         'INACTIVE','MergMeth','MergMethDesc','BKCLASS','CHANGEC1','CH
ANGEC2','CHANGEC3','CHANGEC4','CHANGEC5','ENDEFYMD',
         'ESTYMD','WEBADDR','_merge','Key',]]
```

In [36]:
```
clean_combo = chopped_combo[['CERT','PROXIMATE_CERT','ULTIMATE_CERT','
ULTCERT','ADDRESS','CITY','COUNTY','STNAME','STALP','STCNTY','DEPDOM',
         'OFFDOM','OFFFOR','ASSET','DEP','EQ','ROA',
         'ROE','ROAPTX','NETINC','NETINCQ','INSTCRCD','MUTUAL','NAMEHC
R','SPECGRP','RPT_START_TYPE','RPT_START_YR',
         'RPT_STOP_TYPE','RPT_STOP_YR','FAIL_YR','FAIL_DATE','ACTIVE',
         'INACTIVE','MergMeth','MergMethDesc','BKCLASS','CHANGEC1','CH
ANGEC2','CHANGEC3','CHANGEC4','CHANGEC5','ENDEFYMD',
         'ESTYMD','WEBADDR','_merge','Key',]]
```

In [37]:
```
clean_combo.head(10)
```

Out[37]:

NAME	CERT	PROXIMATE_CERT	ULTIMATE_CERT	ULTCERT	ADDRESS	CITY
Gulfside Bank	59140			59140.0	333 North Orange Avenue	Sarasota
Generations Commercial Bank	59149			59149.0	19 Cayuga Street	Seneca Falls
Charles Schwab Trust Bank	59108			59108.0	2360 Corporate Circle, Suite 400	Henderson
Studio Bank	59112			59112.0	124 12th Avenue South Suite 400	Nashville
CommerceOne Bank	59114			59114.0	2100 Southbridge Parkway, Suite 385	Birmingham
Infinity Bank	59098			59098.0	6 Hutton Centre Drive Suite 100	Santa Ana
Endeavor Bank	59099			59099.0	750 B Street Suite 3110	San Diego
Beacon Community Bank	59106			59106.0	578 East Bay Street, Suite D	Charleston
Tennessee Bank & Trust	59101			59101.0	4007 Hillsboro Pike	Nashville
The Bank of Austin	59104			59104.0	8611 N Mopac Expy Ste 101	Austin

10 rows × 46 columns

We now have an even more clean data set to work with. Lets find out some information on this data set.

In [38]:
```
clean_combo.shape
```

Out[38]:
```
(23200, 46)
```

In [39]:
```
clean_combo.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 23200 entries, Gulfside Bank to Commerce Federal Savings and
Loan Association
Data columns (total 46 columns):
CERT               23200 non-null int64
PROXIMATE_CERT     23200 non-null object
ULTIMATE_CERT      23200 non-null object
ULTCERT            22968 non-null float64
ADDRESS            22968 non-null object
CITY               22968 non-null object
COUNTY             22968 non-null object
STNAME             22968 non-null object
STALP              22968 non-null object
STCNTY             22968 non-null float64
DEPDOM             1260 non-null object
OFFDOM             1260 non-null float64
OFFFOR             1260 non-null float64
ASSET              22964 non-null object
DEP                22964 non-null object
EQ                 1260 non-null object
ROA                1260 non-null float64
ROE                1260 non-null float64
ROAPTX             1260 non-null float64
NETINC             1260 non-null object
NETINCQ            1260 non-null object
INSTCRCD           22968 non-null float64
MUTUAL             1260 non-null float64
NAMEHCR            879 non-null object
SPECGRP            1260 non-null float64
RPT_START_TYPE     23200 non-null object
RPT_START_YR       23200 non-null object
RPT_STOP_TYPE      23200 non-null object
RPT_STOP_YR        23200 non-null object
FAIL_YR            23200 non-null object
FAIL_DATE          23200 non-null object
ACTIVE             22968 non-null float64
INACTIVE           22968 non-null float64
MergMeth           23200 non-null object
MergMethDesc       23200 non-null object
BKCLASS            22968 non-null object
CHANGEC1           22968 non-null float64
CHANGEC2           22968 non-null float64
CHANGEC3           22968 non-null float64
```

```
CHANGEC4          22968 non-null float64
CHANGEC5          22968 non-null float64
ENDEFYMD          22968 non-null object
ESTYMD            22968 non-null object
WEBADDR           1224 non-null object
_merge            23200 non-null category
Key               23200 non-null float64
dtypes: category(1), float64(18), int64(1), object(26)
memory usage: 8.2+ MB
```

In [40]:
```
clean_combo.index
```

Out[40]:
```
Index([                                  'Gulfside Bank',
                            'Generations Commercial Bank',
                              'Charles Schwab Trust Bank',
                                           'Studio Bank',
                                     'CommerceOne Bank',
                                        'Infinity Bank',
                                        'Endeavor Bank',
                                   'Beacon Community Bank',
                                  'Tennessee Bank & Trust',
                                      'The Bank of Austin',
       ...
           'Kokomo Federal Savings and Loan Association',
       'Livingston Federal Savings and Loan Association',
                            'The Savings and Loan of Bangor',
                         'Home Savings and Loan Association',
                       'United Savings and Loan Association',
                     'Scandia Savings and Loan Association',
                            'Home Building and Loan Company',
                        'Galax Savings and Loan Association',
        'Pine Belt Federal Savings and Loan Association',
         'Commerce Federal Savings and Loan Association'],
      dtype='object', name='NAME', length=23200)
```

To work with the particular array that is of interest to us we will need to create a dictionary. The dictionary will allow us to graph the information that we would want. In this case it is for the 'RPT_START_YR' and 'RPT_STOP_YR' columns. The dictionary will count how many times a variable was listed in a specific column, in our case it is years, and provide us with total amounts. This will be given to us for both data arrays.

In [479]:
```
    reader = csv.DictReader(csvfile)
    pre_1984 = 0
    not_closed = 0
    closed_banks = {}
    open_banks = {}
    for row in reader:
        #for banks that have opened
        if row["RPT_START_YR"] == '':
            pre_1984 += 1
        elif row["RPT_START_YR"] in open_banks:
            open_banks[row['RPT_START_YR']] = open_banks[row['RPT_STAR
T_YR']] + 1
        else:
            open_banks[row['RPT_START_YR']] = 1

        #for banks that have closed
        if row["RPT_STOP_YR"] == '':
            not_closed += 1
        elif row["RPT_STOP_YR"] in closed_banks:
            closed_banks[row['RPT_STOP_YR']] = closed_banks[row['RPT_S
TOP_YR']] + 1
        else:
            closed_banks[row['RPT_STOP_YR']] = 1

    #output
    #output should read a long list the year along with the number of
banks closed or opened in the year
    print("Banks Pre 1984: ", pre_1984)
    print("Banks Still Open: ", not_closed)
    print("\nOpen Banks List:")
    for x, y in open_banks.items():
        print(x,"\t", y)
    print('\n')
    print("Closed Banks List:")
    for x, y in closed_banks.items():
        print(x,"\t", y)
```

```
Banks Pre 1984:  17840
Banks Still Open:  5360

Open Banks List:
2018.0   8
2017.0   5
2015.0   1
2013.0   2
2011.0   3
2010.0   11
2009.0   31
2008.0   100
```

```
2007.0   183
2006.0   196
2005.0   179
2004.0   129
2003.0   118
2002.0   95
2001.0   149
2000.0   230
1999.0   274
1998.0   225
1997.0   204
1996.0   163
1995.0   112
1994.0   69
1993.0   76
1986.0   450
1992.0   85
1991.0   123
1985.0   665
1990.0   199
1989.0   227
1988.0   320
1987.0   331
1984.0   397


Closed Banks List:
2018.0   272
2011.0   304
2017.0   248
2015.0   328
2016.0   269
2010.0   365
2013.0   273
2012.0   274
2014.0   303
2009.0   324
2008.0   329
2007.0   329
2006.0   349
2005.0   322
2004.0   334
2002.0   355
2003.0   291
2001.0   439
2000.0   548
1999.0   516
1998.0   684
1997.0   735
1996.0   680
```

```
1992.0   714
1995.0   745
1994.0   686
1993.0   708
1991.0   799
1989.0   991
1990.0   837
1988.0   1085
1987.0   882
1986.0   607
1985.0   533
1984.0   382
```

Topic Introduction and Overview

Legal Changes, Economies of Scale, and the Rise of Nationwide Banking

Historically, a series of federal laws limited the ability of banks to operate in more than one state. In addition most states were unit banking states, which means they had regulations prohibiting banks from having more than one branch. The U.S. system of many small, geographically limited banks was the result of political views that the power of banks should be limited by keeping them small and that the deposits banks received should be used only to fund loans in the local areas.

Many banks, economists and politicans argued that such a U.S. system was inefficient because it failed to take full advantage of economies of scale in banking while exposing banks to greater credit risk by concentrating their loans in one area.

Over time, restrictions on the size and geographic scope of banking were gradually removed.

What effect did that have on the banking industry? Lets take a look!

We will now begin to graph important conclusions that can be gathered from our new dataset. Collectively, the charts/graphs will showcase the consolidation within the banking industry. More specifically the charts will show us how many banks closed/opened during our time period, why they closed/opened, how relevant has been M&A in the closing/opening of banks and a synopsis of the major acqusition institutions will be provided.

We will use a number of different techniques and attributes to showcase the variety of approaches for working with and graphing data.

This will include creating sub-sets of data from our merged master data set. This will demonstrate the ability of creating unique dataframes directly from the search, filter and find capabilities that is provided to use via Python. Moreover, we will use our already existing data set to create other sets of graphs.

The graphs in this project will include line plots, bar graphs and pie charts; each with unique attributes.

We will first examine how many banks have closed and opened within the banking industry between 1984 and 2018. We will pinpoint the passing of the The Riegle-Neal Interstate Banking Act of 1994, which allowed financial institutions to expand to other states.

The two data arrays we will be working with is 'RPT_START_YR' and 'RPT_STOP_YR'.

We can graph this information using the dictionary we created in an earlier step but to show a step by step process of searching for needed information from our data set and then creating a new data array will showcase different functionalities. Moreover, this newly created data will be used again (concentrated) to show a more specific time frame.

These are the following steps

In [41]:
```
clean_combo.RPT_START_YR.value_counts()
```

Out[41]:
```
           17840
1985.0       665
1986.0       450
1984.0       397
1987.0       331
1988.0       320
1999.0       274
2000.0       230
1989.0       227
1998.0       225
1997.0       204
1990.0       199
2006.0       196
2007.0       183
2005.0       179
1996.0       163
2001.0       149
2004.0       129
1991.0       123
2003.0       118
1995.0       112
2008.0       100
2002.0        95
1992.0        85
1993.0        76
1994.0        69
2009.0        31
2010.0        11
2018.0         8
2017.0         5
2011.0         3
2013.0         2
2015.0         1
Name: RPT_START_YR, dtype: int64
```

We will not be able to graph many of the things that we would want to because our desired columns are not strings but rather a variety of different outputs.

We will have to convert these columns to strings.

In [42]:
```
clean_combo.dtypes # Tells us the type of the data in the columns "int
eger, object , float"
```

Out[42]:
```
CERT                int64
PROXIMATE_CERT     object
ULTIMATE_CERT      object
ULTCERT           float64
ADDRESS            object
CITY               object
COUNTY             object
STNAME             object
STALP              object
STCNTY            float64
DEPDOM             object
OFFDOM            float64
OFFFOR            float64
ASSET              object
DEP                object
EQ                 object
ROA               float64
ROE               float64
ROAPTX            float64
NETINC             object
NETINCQ            object
INSTCRCD          float64
MUTUAL            float64
NAMEHCR            object
SPECGRP           float64
RPT_START_TYPE     object
RPT_START_YR       object
RPT_STOP_TYPE      object
RPT_STOP_YR        object
FAIL_YR            object
FAIL_DATE          object
ACTIVE            float64
INACTIVE          float64
MergMeth           object
MergMethDesc       object
BKCLASS            object
CHANGEC1          float64
CHANGEC2          float64
CHANGEC3          float64
CHANGEC4          float64
CHANGEC5          float64
ENDEFYMD           object
ESTYMD             object
WEBADDR            object
_merge           category
Key               float64
dtype: object
```

In [43]:
```
clean_combo.convert_objects(convert_numeric=True) #converts the string to a numeric value - we can now use the entry
```

```
/Users/sample/anaconda3/lib/python3.7/site-packages/ipykernel_launcher.py:1: FutureWarning: convert_objects is deprecated.  To re-infer data dtypes for object columns, use DataFrame.infer_objects()
For all other conversions use the data-type specific converters pd.to_datetime, pd.to_timedelta and pd.to_numeric.
  """Entry point for launching an IPython kernel.
```

Out[43]:

NAME	CERT	PROXIMATE_CERT	ULTIMATE_CERT	ULTCERT	ADDRESS	
Gulfside Bank	59140	NaN	NaN	59140.0	333 North Orange Avenue	Sa
Generations Commercial Bank	59149	NaN	NaN	59149.0	19 Cayuga Street	Seneca
Charles Schwab Trust Bank	59108	NaN	NaN	59108.0	2360 Corporate Circle, Suite 400	Hend
Studio Bank	59112	NaN	NaN	59112.0	124 12th Avenue South Suite 400	Na
CommerceOne Bank	59114	NaN	NaN	59114.0	2100 Southbridge Parkway, Suite 385	Birmin
Infinity Bank	59098	NaN	NaN	59098.0	6 Hutton Centre Drive Suite 100	Sant
Endeavor Bank	59099	NaN	NaN	59099.0	750 B Street Suite 3110	San
Beacon Community Bank	59106	NaN	NaN	59106.0	578 East Bay Street, Suite D	Char
Tennessee Bank & Trust	59101	NaN	NaN	59101.0	4007 Hillsboro Pike	Na
The Bank of Austin	59104	NaN	NaN	59104.0	8611 N Mopac Expy Ste 101	A
					201 North	

Winter Park National Bank	59105	NaN	NaN	59105.0	New York Avenue, Suite 100	Winte
International Bank of Commerce	59093	NaN	NaN	59093.0	3817 N.W. Expressway	Oklahom
Blue Gate Bank	59094	NaN	NaN	57903.0	611 Anton Boulevard, Suite 1050	Costa
Blue Gate Bank	59094	57903.0	57903.0	57903.0	611 Anton Boulevard, Suite 1050	Costa
Primary Bank	59086	NaN	NaN	59086.0	207 Route 101	Be
Bank of Bird-in-Hand	59074	NaN	NaN	59074.0	309 North Ronks Road	Bird In
HarborOne Bank	59070	NaN	NaN	59070.0	68 Legion Parkway	Bro
Superior Bank, National Association	59026	NaN	NaN	4999.0	4350 West Cypress Street	T
Superior Bank, National Association	59026	4999.0	4999.0	4999.0	4350 West Cypress Street	T
AloStar Bank of Commerce	59060	NaN	NaN	4999.0	3595 Grandview Parkway, Suite 425	Birmin
AloStar Bank of Commerce	59060	57870.0	57870.0	4999.0	3595 Grandview Parkway, Suite 425	Birmin
NaN	59028	NaN	NaN	NaN	NaN	
NaN	59028	0.0	0.0	NaN	NaN	
Start Community Bank	58952	NaN	NaN	58952.0	299 Whalley Avenue	New H
Hillcrest Bank, National Association	59015	NaN	NaN	59052.0	11111 W 95th St	Overlanc
Hillcrest Bank, National Association	59015	59052.0	59052.0	59052.0	11111 W 95th St	Overlanc
					7800 East Orchard	Green

NBH Bank	59052	NaN	NaN	59052.0	Road Suite 200	\
Lakeside Bank	58961	NaN	NaN	58961.0	4735 Nelson Road	Lake C
Bay Bank, FSB	59004	NaN	NaN	27599.0	7151 Columbia Gateway Drive, Suite A	Col
Bay Bank, FSB	59004	27599.0	27599.0	27599.0	7151 Columbia Gateway Drive, Suite A	Col
...	...	...	...	...	...	
First Tennessee Bank N.A. Chattanooga	22227	4977.0	4977.0	4977.0	701 Market Street	Chatta
Merrillville Bank and Trust Company	22303	9282.0	16571.0	16571.0	7701 Broadway Avenue	Mer
The Women's Bank	22417	20450.0	9846.0	9846.0	1007 East Main Street	Rich
First Missouri Bank of West County	22566	12229.0	12229.0	12229.0	1399 Manchester Road	St. Louis C
Washington National Bank of Chicago	22568	16877.0	34967.0	34967.0	2539 North Kedzie Avenue, Room 5e	Ch
West Coast Bank	22744	0.0	0.0	22744.0	16311 Ventura Boulevard	Los An
SouthTrust Bank of Dale County	22791	2793.0	3511.0	3511.0	U.S. Highway 231 And County Road 59	Midlan
Garden Grove Community Bank	22848	3432.0	3511.0	3511.0	11050 Garden Grove Boulevard	Garden
State Bank of Mills	22861	23337.0	19184.0	19184.0	305 Wyoming	
First Missouri					1353	

Bank of Ellisville	23198	12229.0	12229.0	12229.0	Manchester Road	El
First National Bank	23335	25124.0	16835.0	16835.0	3610 College Avenue	S
Stewardship Bank of Oregon	23362	0.0	0.0	23362.0	1918 N.E. 181st Avenue	Po
Union Planters Bank of Nashville	23596	4979.0	12368.0	12368.0	3904 Hillsboro Road	Na
Flagship National Bank of Indian River County	23656	24202.0	867.0	867.0	2231 Indian River Boulevard	Vero I
First National Bank of Prior Lake	24098	5250.0	3511.0	3511.0	Brooksville Mall	Prio
Fidelity Union Trust Company of Florida, National Association	24758	0.0	0.0	24758.0	1515 North Federal Highway, Suite 315	Boca
American Fidelity Savings Association	27961	29080.0	19048.0	19048.0	522 South Broadway	
First Federal Savings and Loan Association of Grand Forks and Minot	29184	27920.0	6548.0	6548.0	201 S Fourth St	Grand
Heritage Savings and Loan Association	29898	29281.0	29281.0	29281.0	1900 E Allegheny Ave	Philad
Midland Federal Savings and Loan Association	29908	28736.0	6548.0	6548.0	444 17th St	D
Kokomo Federal Savings and Loan	30227	28894.0	28894.0	28894.0	325 North Main Street	Ko

Association						
Livingston Federal Savings and Loan Association	30283	32068.0	0.0	32686.0	201 Range Ave	Denham Sp
The Savings and Loan of Bangor	30304	17721.0	3510.0	3510.0	201 Main Street	B
Home Savings and Loan Association	30525	27878.0	6560.0	6560.0	201 South Main Street	North Balt
United Savings and Loan Association	30798	32354.0	9846.0	9846.0	259 North Main Street	Mour
Scandia Savings and Loan Association	30911	31244.0	6548.0	6548.0	518 East Locust St	Des M
Home Building and Loan Company	31038	28624.0	6653.0	6653.0	132 Main St	Newcomer
Galax Savings and Loan Association	31318	0.0	0.0	31318.0	118 N Main St	
Pine Belt Federal Savings and Loan Association	31383	32208.0	12368.0	12368.0	700 Hardy St	Hattie
Commerce Federal Savings and Loan Association	31847	31881.0	12368.0	12368.0	7005 Maynardville Highway	Kn

23200 rows × 46 columns

It looks like we have converted the data series to Floats. Now lets try graphing some things.

In [44]:
```
clean_combo.convert_objects(convert_numeric=True).dtypes
```

```
/Users/sample/anaconda3/lib/python3.7/site-packages/ipykernel_launch
er.py:1: FutureWarning: convert_objects is deprecated.  To re-infer
data dtypes for object columns, use DataFrame.infer_objects()
For all other conversions use the data-type specific converters pd.t
o_datetime, pd.to_timedelta and pd.to_numeric.
  """Entry point for launching an IPython kernel.
```

Out[44]:
```
CERT                int64
PROXIMATE_CERT    float64
ULTIMATE_CERT     float64
ULTCERT           float64
ADDRESS            object
CITY               object
COUNTY             object
STNAME             object
STALP              object
STCNTY            float64
DEPDOM            float64
OFFDOM            float64
OFFFOR            float64
ASSET             float64
DEP               float64
EQ                float64
ROA               float64
ROE               float64
ROAPTX            float64
NETINC            float64
NETINCQ           float64
INSTCRCD          float64
MUTUAL            float64
NAMEHCR            object
SPECGRP           float64
RPT_START_TYPE    float64
RPT_START_YR      float64
RPT_STOP_TYPE     float64
RPT_STOP_YR       float64
FAIL_YR           float64
FAIL_DATE         float64
ACTIVE            float64
INACTIVE          float64
MergMeth          float64
MergMethDesc       object
BKCLASS            object
CHANGEC1          float64
CHANGEC2          float64
CHANGEC3          float64
CHANGEC4          float64
CHANGEC5          float64
ENDEFYMD           object
ESTYMD             object
WEBADDR           float64
_merge           category
Key               float64
dtype: object
```

We will have to approach it differently since convert_objects is deprecated.

In [45]:
```
clean_combo.RPT_START_YR.unique() #we take a look at all of the entrie
s of the column 'RPT_START_YR'
```

Out[45]:
```
array([2018.0, 2017.0, '', 2015.0, 2013.0, 2011.0, 2010.0, 2009.0, 2
008.0,
       2007.0, 2006.0, 2005.0, 2004.0, 2003.0, 2002.0, 2001.0, 2000.
0,
       1999.0, 1998.0, 1997.0, 1996.0, 1995.0, 1994.0, 1993.0, 1986.
0,
       1992.0, 1991.0, 1985.0, 1990.0, 1989.0, 1988.0, 1987.0, 1984.
0],
      dtype=object)
```

We will use the pd.to_numeric function on 'RPT_START_YR' instead to convert the array into a float so we can work use it to graph.

In [46]:
```
clean_combo ['RPT_START_YR'] = pd.to_numeric(clean_combo['RPT_START_YR
'], errors='coerce') #converts the string to a numeric value - we can
now use the entry
```

In [47]:
```
clean_combo.RPT_START_YR.unique()
```

Out[47]:
```
array([2018., 2017.,   nan, 2015., 2013., 2011., 2010., 2009., 2008.
,
       2007., 2006., 2005., 2004., 2003., 2002., 2001., 2000., 1999.
,
       1998., 1997., 1996., 1995., 1994., 1993., 1986., 1992., 1991.
,
       1985., 1990., 1989., 1988., 1987., 1984.])
```

If problems arise because of the 'NaN' value, we can replace it with the below function:

clean_combo = clean_combo.replace(np.nan, 0, regex=True)

In [48]:
```
clean_combo.dtypes
```

Out[48]:
```
CERT                int64
PROXIMATE_CERT     object
ULTIMATE_CERT      object
ULTCERT           float64
ADDRESS            object
CITY               object
COUNTY             object
STNAME             object
STALP              object
STCNTY            float64
DEPDOM             object
OFFDOM            float64
OFFFOR            float64
ASSET              object
DEP                object
EQ                 object
ROA               float64
ROE               float64
ROAPTX            float64
NETINC             object
NETINCQ            object
INSTCRCD          float64
MUTUAL            float64
NAMEHCR            object
SPECGRP           float64
RPT_START_TYPE     object
RPT_START_YR      float64
RPT_STOP_TYPE      object
RPT_STOP_YR        object
FAIL_YR            object
FAIL_DATE          object
ACTIVE            float64
INACTIVE          float64
MergMeth           object
MergMethDesc       object
BKCLASS            object
CHANGEC1          float64
CHANGEC2          float64
CHANGEC3          float64
CHANGEC4          float64
CHANGEC5          float64
ENDEFYMD           object
ESTYMD             object
WEBADDR            object
_merge           category
Key               float64
dtype: object
```

In [49]:
```
clean_combo ['RPT_STOP_YR'] = pd.to_numeric(clean_combo['RPT_STOP_YR']
, errors='coerce')
```

It is easier for us to graph the number of banks that had started manually rather then creating new columns to reflect a new array containing the number of banks for this particular instance.

In [50]:
```
clean_combo.RPT_START_YR.value_counts() # counts the number of individ
ual values in the column 'RPT_START_YR'
```

Out[50]:
```
1985.0    665
1986.0    450
1984.0    397
1987.0    331
1988.0    320
1999.0    274
2000.0    230
1989.0    227
1998.0    225
1997.0    204
1990.0    199
2006.0    196
2007.0    183
2005.0    179
1996.0    163
2001.0    149
2004.0    129
1991.0    123
2003.0    118
1995.0    112
2008.0    100
2002.0     95
1992.0     85
1993.0     76
1994.0     69
2009.0     31
2010.0     11
2018.0      8
2017.0      5
2011.0      3
2013.0      2
2015.0      1
Name: RPT_START_YR, dtype: int64
```

In [51]:
```
clean_combo.RPT_STOP_YR.value_counts()
```

Out[51]:
```
1988.0    1085
1989.0     991
1987.0     882
1990.0     837
1991.0     799
1995.0     745
1997.0     735
1992.0     714
1993.0     708
1994.0     686
1998.0     684
1996.0     680
1986.0     607
2000.0     548
1985.0     533
1999.0     516
2001.0     439
1984.0     382
2010.0     365
2002.0     355
2006.0     349
2004.0     334
2007.0     329
2008.0     329
2015.0     328
2009.0     324
2005.0     322
2011.0     304
2014.0     303
2003.0     291
2012.0     274
2013.0     273
2018.0     272
2016.0     269
2017.0     248
Name: RPT_STOP_YR, dtype: int64
```

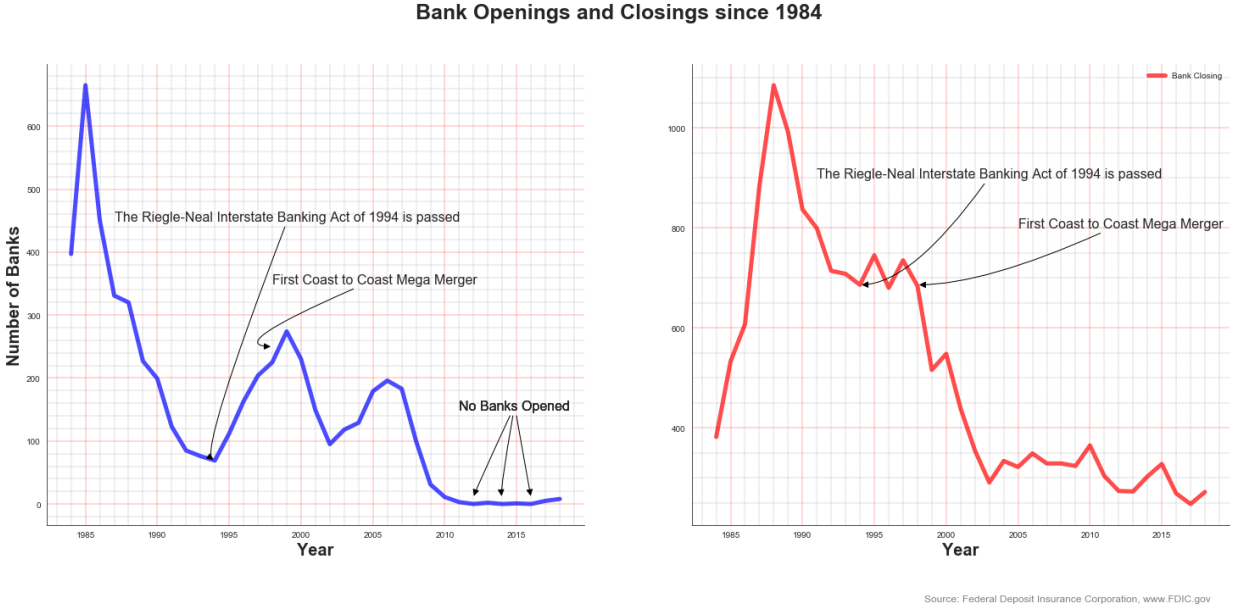
In [472]:
```
fig, ax = plt.subplots(nrows = 1, ncols = 2, sharex = True, figsize =
(25,10)) # Configure the size of our chart

year = [1984, 1985, 1986, 1987, 1988, 1989, 1990, 1991, 1992, 1993, 19
94, 1995, 1996, 1997, 1998, 1999, 2000, 2001, 2002, 2003, 2004, 2005,
2006,
       2007, 2008, 2009, 2010, 2011, 2012, 2013, 2014, 2015, 2016, 201
7, 2018]

Number_of_Banks_Opening = [397,665,450,331,320,227,199,123,85,76,69,11
```

```
2,163,204,225,274,230,149,95,118,129,179,196,183,100,31,11,3,0,2,0,1,0
,5,8]

Number_of_Banks_Closing = [382,533,607,882,1085,991,837,799,714,708,68
6,745,680,735,684,516,548,439,355,291,334,322,349,329,329,324,365,304,
274,273,303,328,269,248,272]

# We can make two graphs populate side by side

ax[0].plot(year, Number_of_Banks_Opening, color = "b", linewidth = 5,
         alpha = 0.7, label = "Bank Openings")
ax[1].plot(year, Number_of_Banks_Closing, color = "r", linewidth = 5,
         alpha = 0.7, label = "Bank Closing")

fig.suptitle("Bank Openings and Closings since 1984", fontsize = 25, f
ontweight = "bold") # Placed fontsize and boliding onto text

ax[0].set_ylabel("Number of Banks", fontweight = 'bold', size = 20)
ax[0].set_xlabel("Year", fontweight = 'bold', size = 20)
ax[1].set_xlabel("Year", fontweight = 'bold', size = 20)


for var in ax:

    var.spines["right"].set_visible(False)
    var.spines["top"].set_visible(False)

var.legend(frameon = False) # Placed a legend

# We will need to create an arrow to emphasis the passing of The Riegl
e-Neal Banking Act

my_message = "The Riegle-Neal Interstate Banking Act of 1994 is passed
"


ax[0].annotate(
    my_message,
    xy=(1994, 69), # This is where we point at...
    xycoords="data", # Not exactly sure about this
    xytext=(1987, 450), # This is about where the text is
    horizontalalignment="left", # How the text is alined
    arrowprops={
        "arrowstyle": "-|>", # This is stuff about the arrow
        "connectionstyle": "angle3,angleA=69,angleB=0",
        "color": "black"
    },
    fontsize=16,)

my_message = "The Riegle-Neal Interstate Banking Act of 1994 is passed
"
```

```
ax[1].annotate(
    my_message,
    xy=(1994, 686), # This is where we point at...
    xycoords="data", # Not exactly sure about this
    xytext=(1991, 900), # This is about where the text is
    horizontalalignment="left", # How the text is alined
    arrowprops={
        "arrowstyle": "-|>", # This is stuff about the arrow
        "connectionstyle": "angle3,angleA=55,angleB=0",
        "color": "black"
    },
    fontsize=16,)

##########################################################################
#####

my_message = "First Coast to Coast Mega Merger"

ax[0].annotate(
    my_message,
    xy=(1998, 250), # This is where we point at...
    xycoords="data", # Not exactly sure about this
    xytext=(1998, 350), # This is about where the text is
    horizontalalignment="left", # How the text is alined
    arrowprops={
        "arrowstyle": "-|>", # This is stuff about the arrow
        "connectionstyle": "angle3,angleA=23,angleB=0",
        "color": "black"
    },
    fontsize=16,)

my_message = "First Coast to Coast Mega Merger"

ax[1].annotate(
    my_message,
    xy=(1998, 686), # This is where we point at...
    xycoords="data", # Not exactly sure about this
    xytext=(2005, 800), # This is about where the text is
    horizontalalignment="left", # How the text is alined
    arrowprops={
        "arrowstyle": "-|>", # This is stuff about the arrow
        "connectionstyle": "angle3,angleA=25,angleB=0",
        "color": "black"
    },
    fontsize=16,)

##########################################################################
#####
```

```
my_message = "No Banks Opened"

ax[0].annotate(
    my_message,
    xy=(2014, 10), # This is where we point at...
    xycoords="data", # Not exactly sure about this
    xytext=(2011, 150), # This is about where the text is
    horizontalalignment="left", # How the text is alined
    arrowprops={
        "arrowstyle": "-|>", # This is stuff about the arrow
        "connectionstyle": "angle3,angleA=80,angleB=0",
        "color": "black"
    },
    fontsize=16,)

my_message = "No Banks Opened"

ax[0].annotate(
    my_message,
    xy=(2012, 10), # This is where we point at...
    xycoords="data", # Not exactly sure about this
    xytext=(2011, 150), # This is about where the text is
    horizontalalignment="left", # How the text is alined
    arrowprops={
        "arrowstyle": "-|>", # This is stuff about the arrow
        "connectionstyle": "angle3,angleA=65,angleB=0",
        "color": "black"
    },
    fontsize=16,)

my_message = "No Banks Opened"

ax[0].annotate(
    my_message,
    xy=(2016, 10), # This is where we point at...
    xycoords="data", # Not exactly sure about this
    xytext=(2011, 150), # This is about where the text is
    horizontalalignment="left", # How the text is alined
    arrowprops={
        "arrowstyle": "-|>", # This is stuff about the arrow
        "connectionstyle": "angle3,angleA=100,angleB=0",
        "color": "black"
    },
    fontsize=16,)

##########################################################################
#####

plt.figtext(.7, 0.00000001, 'Source: Federal Deposit Insurance Corpora
tion, www.FDIC.gov', color ='grey', size = 12) # Include citation
```

```
# We turn on the minor TICKS, which are required for the minor GRID
# We then Customize the major and minor grids
ax[0].minorticks_on()
ax[0].grid(which='major', linestyle='-', linewidth='0.3', color = 'red
')
ax[0].grid(which='minor', linestyle=':', linewidth='0.3', color='black
')

ax[1].minorticks_on()
ax[1].grid(which='major', linestyle='-', linewidth='0.3', color = 'red
')
ax[1].grid(which='minor', linestyle=':', linewidth='0.3', color='black
')

sns.set_style('white')


plt.show()
```



2012, 2014, 2016 - no banks opened.

1998 Merger of NationsBank, based in North Carolina, and Bank of America, based in California, produced the first bank with branches on both coasts.

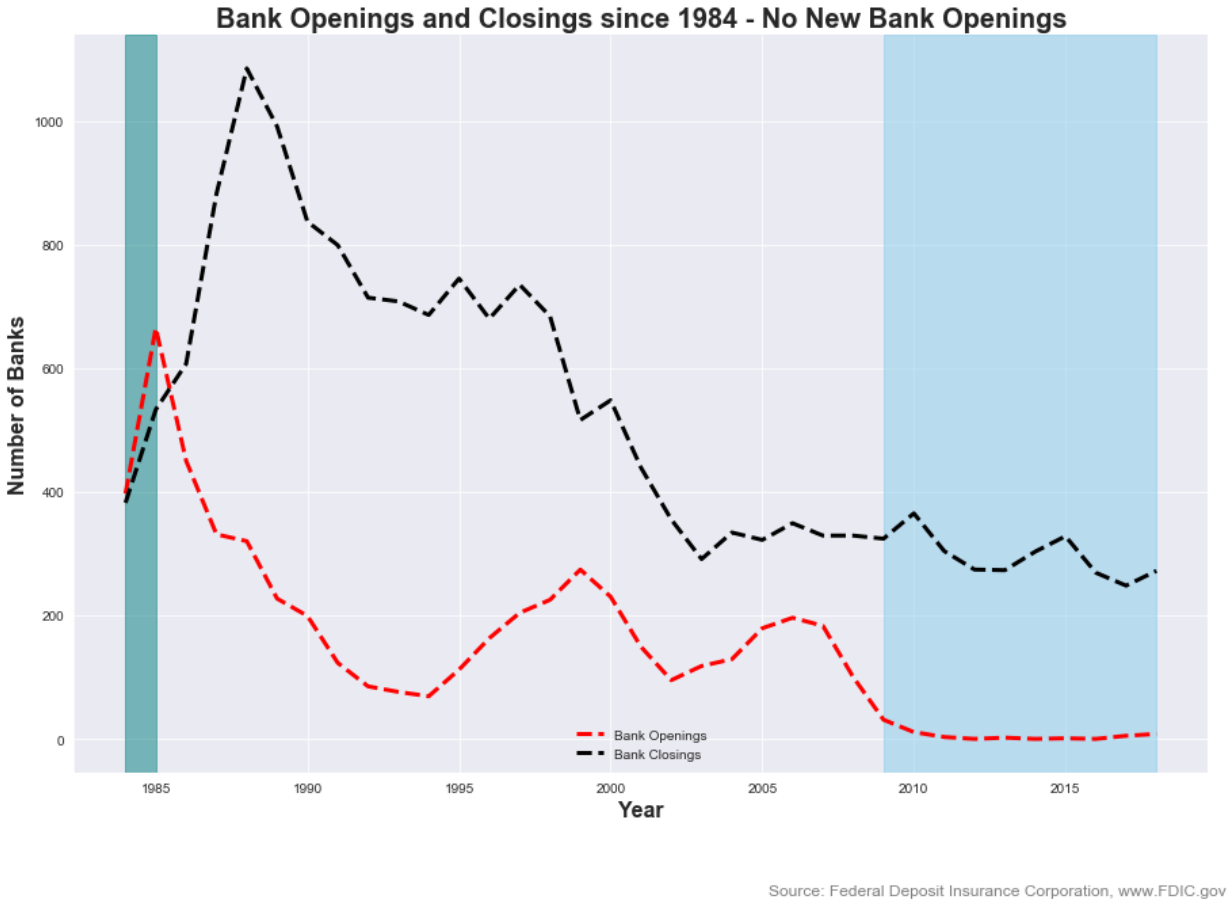
Interesting, these two charts show us that the trajectory has been mostly downward sloping for both the opening and closing of community banking institutions.

Consolidation and Economies of Scale

Rapid consolidation in the U.S. banking industry has resulted from regulatroy changes. Consolidation is what is expected in an industry with substantial economies of scale when firms are free to compete with each other, as has been the case since the removal of restrictions on banks branching and operating n more than one state.

Economies of scale allows firms to expand because they have lower average cost of producing goods or services. This lower cost allows the expanding firms to sell their goods or services at a lower price than smaller rivals, driving them out of business or forcing them to merge with other firms. Becuase large banks have lower costs than smaller banks, they can offer depositors higher interest rates, give borrowers lower interest rates and provide investment advice and other financial services at a lower price.

In [470]:
```
fig, ax = plt.subplots(figsize = (15,10))

year = [1984, 1985, 1986, 1987, 1988, 1989, 1990, 1991, 1992, 1993, 19
94, 1995, 1996, 1997, 1998, 1999, 2000, 2001, 2002, 2003, 2004, 2005,
2006,
       2007, 2008, 2009, 2010, 2011, 2012, 2013, 2014, 2015, 2016, 201
7, 2018]

Number_of_Banks_Opening = [397,665,450,331,320,227,199,123,85,76,69,11
2,163,204,225,274,230,149,95,118,129,179,196,183,100,31,11,3,0,2,0,1,0
,5,8]

Number_of_Banks_Closing = [382,533,607,882,1085,991,837,799,714,708,68
6,745,680,735,684,516,548,439,355,291,334,322,349,329,329,324,365,304,
274,273,303,328,269,248,272]

ax.plot(year, Number_of_Banks_Opening, color = "r", linewidth = 3,
        linestyle = '--', label = "Bank Openings")

ax.plot(year, Number_of_Banks_Closing, color = "k", linewidth = 3,
        linestyle = '--', label = "Bank Closings")

ax.legend(frameon = False)

ax.set_title("Bank Openings and Closings since 1984 - No New Bank Open
ings", fontsize = 20, fontweight = "bold")
ax.set_ylabel("Number of Banks", fontweight = 'bold', size = 16)
ax.set_xlabel("Year", fontweight = 'bold', size = 16)

ax.spines["right"].set_visible(False)
ax.spines["top"].set_visible(False)

plt.figtext(.6, 0.00000001, 'Source: Federal Deposit Insurance Corpora
tion, www.FDIC.gov', color ='grey', size = 12) # Include citation

plt.axvspan(2009, 2018, color = 'skyblue', alpha = 0.5)
sns.set_style('darkgrid')

plt.axvspan(1984, 1985, color = 'teal', alpha = 0.5)



plt.show()
```



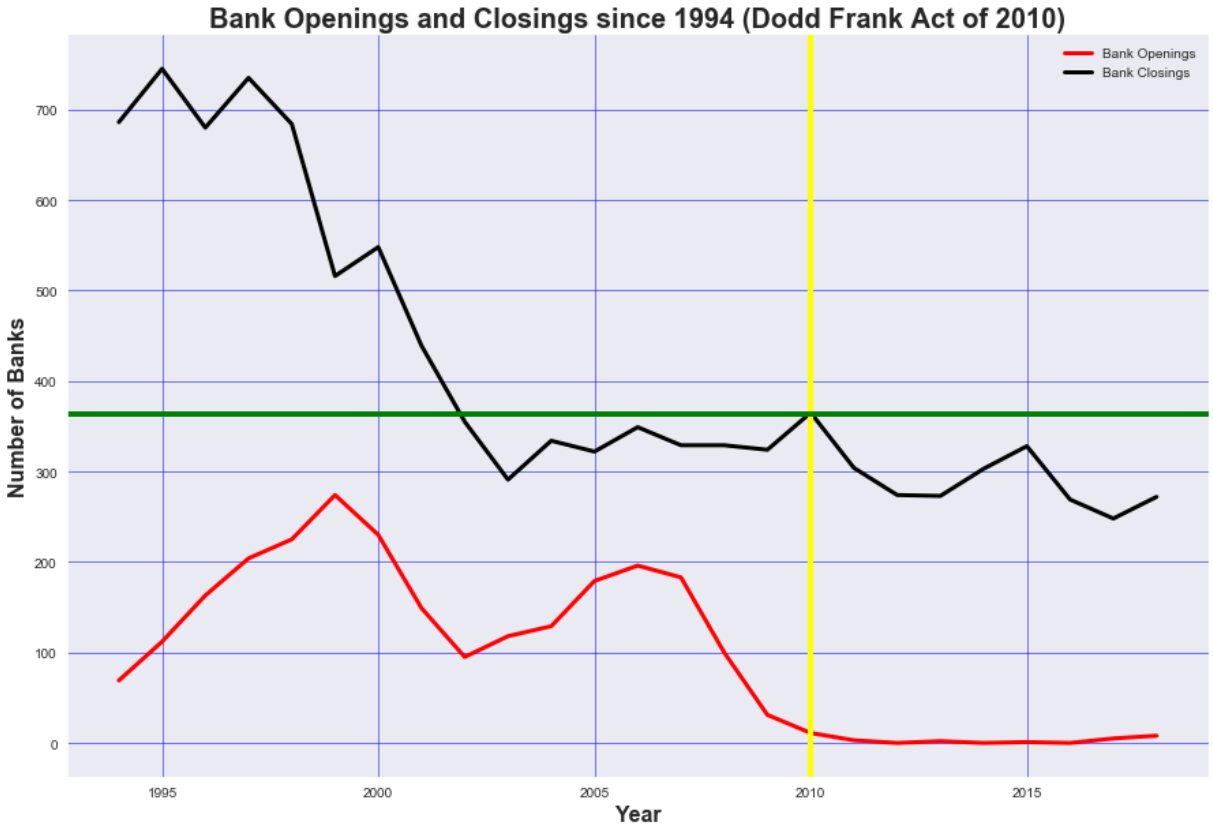
This a one to one comparison of the two arrays. It is easier for us to see the direct differences between bank openings and bank closing over the same time period.

Some interesting things that we can expand on is how the closing/opening of banks was both updward sloping up until 1985 when the rate of bank openings had begun a sudden and sharp decline. The rate of bank closings had continued to increase for a few years (so there were significantly less banks opening than bank closings). The bank closing rate had then also suddenly and sharply dropped although staying constant during the 1990s.

The rate of bank openings/closings has been steady and at 30 year lows (based on the highlighted area in blue).

As highlighted in blue, there are virtually no banks opening for the past 10 years yet they continue to close at a steady pace. Highlighted in teal showcases how for two short years (1984 and 1985) there were more bank openings than closings.

In [442]:
```
fig, ax = plt.subplots(figsize = (15,10))

year = [1994, 1995, 1996, 1997, 1998, 1999, 2000, 2001, 2002, 2003, 20
04, 2005, 2006,
       2007, 2008, 2009, 2010, 2011, 2012, 2013, 2014, 2015, 2016, 201
7, 2018]

Number_of_Banks_Opening = [69,112,163,204,225,274,230,149,95,118,129,1
79,196,183,100,31,11,3,0,2,0,1,0,5,8]

Number_of_Banks_Closing = [686,745,680,735,684,516,548,439,355,291,334
,322,349,329,329,324,365,304,274,273,303,328,269,248,272]

ax.plot(year, Number_of_Banks_Opening, color = "r", linewidth = 3,
        linestyle = '-', label = "Bank Openings")

ax.plot(year, Number_of_Banks_Closing, color = "k", linewidth = 3,
        linestyle = '-', label = "Bank Closings")

ax.legend(frameon = False)

ax.set_title("Bank Openings and Closings since 1994 (Dodd Frank Act of
2010)", fontsize = 20, fontweight = "bold")
ax.set_ylabel("Number of Banks", fontweight = 'bold', size = 16)
ax.set_xlabel("Year", fontweight = 'bold', size = 16)


ax.spines["right"].set_visible(False)
ax.spines["top"].set_visible(False)

plt.figtext(.6, 0.00000001, 'Source: Federal Deposit Insurance Corpora
tion, www.FDIC.gov', color ='grey', size = 12) # Include citation

plt.grid(linestyle='-', linewidth='0.5', color = 'blue')

plt.axvline(2010, color = 'yellow', linewidth = '4')
plt.axhline(363, color = 'green', linewidth = '4')

plt.show()
```



Source: Federal Deposit Insurance Corporation, www.FDIC.gov

Although, we used an arrow to highlight The Riegle-Neal Interstate Banking Act of 1994, we will now set a limit on the x-axis or the years that we are examining. We will set the date to show us the data only for years 1994 and after.

After the mid 1970s, most states eliminatd restrictions on branching within the state. In 1994, Congress passed the Riegle - Neale Interstate Banking and Branching Efficiency Act, which allowed for the phased removal of restrictions on interstate banking.

Since Congress passed the Dodd-Frank Act in 2010, all banks have faced greater regulation, in particular community banks. Additional regulations such as additional mandatory filings has raised costs as banks have to show detailed accounts of their activities, which are a particular burden on smaller banks because they require an extensive commitment of time from the bank's staff. In effect, these regulations are another large fixed cost that has to be spread over the smaller quantity of services a community bank provides.

This has the potential of furthern diminishing the opening rate of banks (although it is nearly non-existent in 2018 and accelerate the rate at which community banks are closing).

Since we know what how many banks have opened and closed, we now would like to see the leading reasons for why they had opened and closed.

As stated earlier, the choices for reasons why institutions started reporting are:

“1” - new institution,

“2” - institution established previously, but recently received insurance, or

“3” - other

The choices for reasons why institutions stopped reporting are:

“1” - failure,

“2” - merger without assistance, consolidation,

“3” - merger, acquisition

“4” - self-liquidation, or

“5” - other.

Using a bar graph we are able to show the main culprit for the closing of banks and the primary cause for the opening of banking institutions.

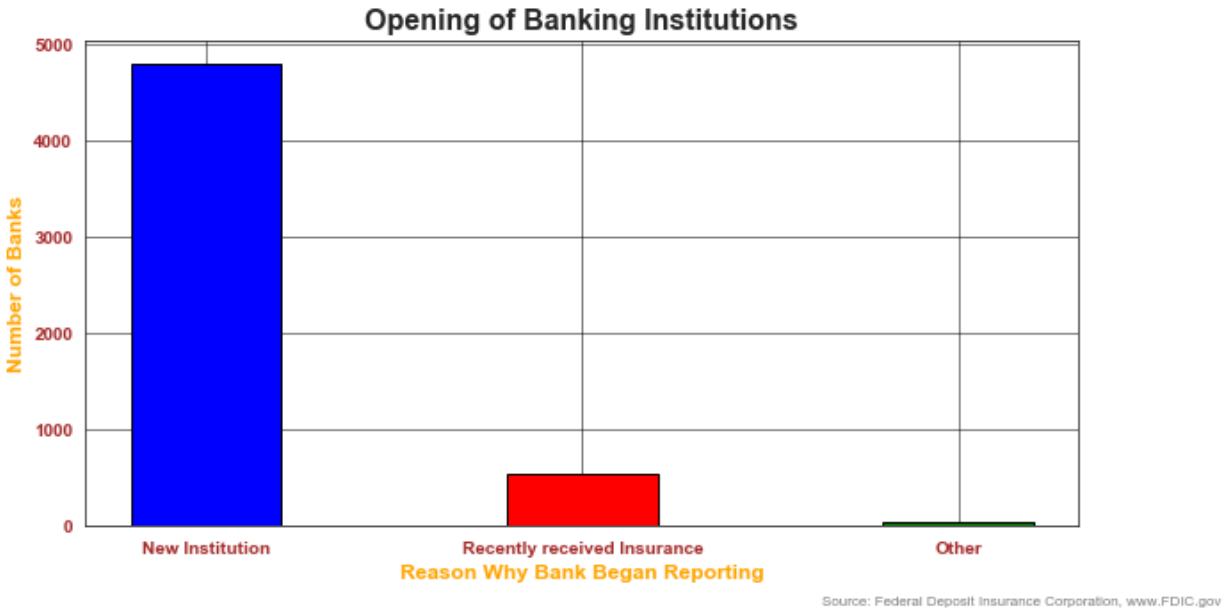
In [57]:
```
clean_combo ['RPT_START_TYPE'] = pd.to_numeric(clean_combo['RPT_START_
TYPE'], errors='coerce')
```

In [58]:
```
clean_combo ['RPT_STOP_TYPE'] = pd.to_numeric(clean_combo['RPT_STOP_TY
PE'], errors='coerce')
```

In [59]:
```
clean_combo.RPT_START_TYPE.value_counts()
```

Out[59]:
```
1.0    4799
2.0     534
3.0      27
Name: RPT_START_TYPE, dtype: int64
```

In [324]:
```
plt.figure(figsize=(10,5))

reason = [1,2,3] # x-coordinates of left sides of bars
amount = [4799,534,27] #this is the height of the bars

tick_label = ['New Institution', 'Recently received Insurance', 'Other
'] # this is what we are labeling the bars

plt.bar(reason, amount, tick_label = tick_label, # plotting the bar ch
art - includes color and width
       width = 0.4, color = ['blue', 'red', 'green'], edgecolor ='blac
k') # We have made the edges of the bars a different color

plt.xlabel('Reason Why Bank Began Reporting', size = 12, color = 'oran
ge', fontweight = "bold") #Changed the size, color and fontweight of t
he header
plt.ylabel('Number of Banks', size = 12, color = 'orange', fontweight
= "bold") #Changed the size, color and fontweight of the header
plt.title('Opening of Banking Institutions', fontweight = "bold", size
= 16)

plt.xticks(color = 'brown', fontweight = "bold") #Changing the header
of the axis
plt.yticks(color = 'brown', fontweight = "bold") #Changing the header
of the axis

plt.tick_params(axis = 'x', direction = 'out', length = 5, width = 3)
# this includes ticks in the x-axis

plt.figtext(.7, 0.00000001, 'Source: Federal Deposit Insurance Corpora
tion, www.FDIC.gov', color ='grey', size = 8) # Include citation

plt.grid(linestyle='-', linewidth='0.5', color = 'black')
sns.set_style('white')


plt.show()
```

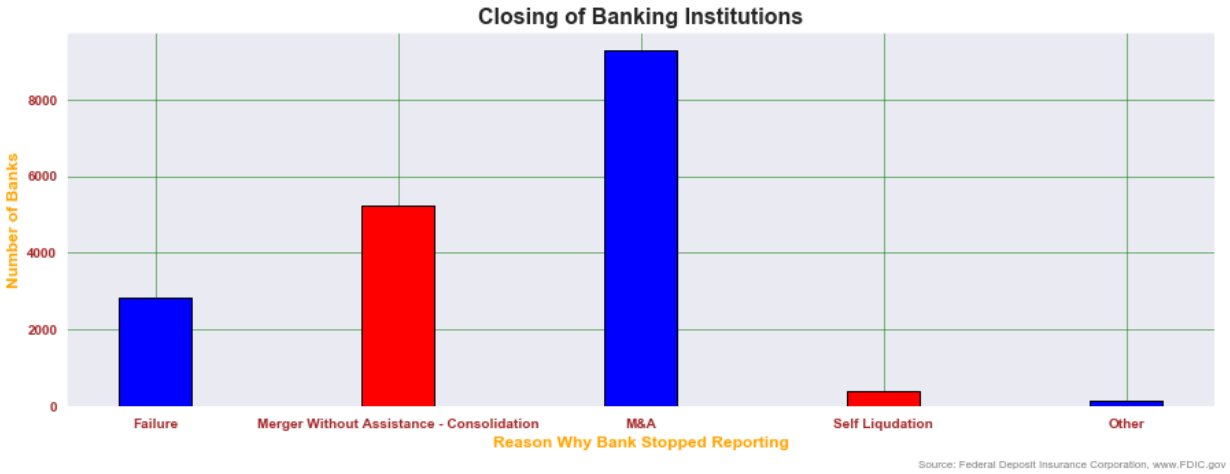


As we had initially presumed, the overwhelming majority of banking institutions began reporting primarily for the reason that they are 'new institutions' (4799). The other portion was formed prior but did not undertake FDIC insurance; preventing them from fully operating.

In [62]:
```
clean_combo.RPT_STOP_TYPE.value_counts()
```

Out[62]:
```
3.0    9290
2.0    5245
1.0    2816
4.0     368
5.0     121
Name: RPT_STOP_TYPE, dtype: int64
```

In [319]:
```
plt.figure(figsize=(15,5))

reason = [1,2,3,4,5] # x-coordinates of left sides of bars
amount = [2816,5245,9290,368,121] #this is the height of the bars

tick_label = ['Failure', 'Merger Without Assistance - Consolidation',
'M&A', 'Self Liqudation', 'Other'] # this is what we are labeling the
bars

plt.bar(reason, amount, tick_label = tick_label, # plotting the bar ch
art - includes color and width
       width = 0.3, color = ['blue', 'red'], edgecolor ='black') # We
have made the edges of the bars a different color

plt.xlabel('Reason Why Bank Stopped Reporting', size = 12, color = 'or
ange', fontweight = "bold" ) #Changed the size, color and fontweight o
f the header
plt.ylabel('Number of Banks', size = 12, color = 'orange', fontweight
= "bold") #Changed the size, color and fontweight of the header
plt.title('Closing of Banking Institutions',fontweight = "bold", size
= 16) #Changed the size and fontweight of the header

plt.xticks(color = 'brown', fontweight = "bold") #Changing the header
of the axis
plt.yticks(color = 'brown', fontweight = "bold") #Changing the header
of the axis

plt.tick_params(axis = 'x', color = 'red', direction = 'out', length =
5, width = 3) # this includes ticks in the x-axis

plt.figtext(.7, 0.00000001, 'Source: Federal Deposit Insurance Corpora
tion, www.FDIC.gov', color ='grey', size = 8) # Include citation

plt.grid(linestyle='-', linewidth='0.5', color = 'green')

sns.set_style('dark')



plt.show()
```



The closing of banking institutions was a result of a number of different scenarios. M&A is the leading reason for the closure of a bank - 9290 institutions stopped reporting for this reason. Consolidation - merger without assistance is an M&A transaction as well, although the process is somewhat different as the consolidation process commands a different ownership stake and operational outlook. The standard M&A transaction see's the acquired institutions join the portfolio of the acquirers bank holdings (acquirer can choose to leave the institution as is and operate under the parent companys umbrella or competely absorb the acquired institution - continue to operate independently although being owned by the parent corp). The consolidation process allows the acquired institution to to be 'equal' partners with the acquiring institution.

There is also a possibility that Institutions that underwent "Failure" - 2816 banks - were subject to a possible acqusition or merger.

Acqusition

In an acqusition, a new company does not emerge. Instead, the smaller company is often consumed and ceases to exist with its assets becoming part of the larger company. An acqusition takes place when one company takes over all of the operational management decisions of another company - takeover of one entity of another.

MERGER_

In a merger, one company takes over another, including all assets and liabilities. The company that takes over remains active, while the one that is acquired essentially ceases to exist. Two companies combine forces to create a new joint organization but operate under one of the merging companies.

Consolidation

In a consolidation, two or more companies merge to form one new, larger company. All of each company's assets and liabilities then become the property of the new company. Unlike a merger, the original entities cease to exist and there is now one combined organization - an entirely new company.

Now we will examine what type of mergers were most prevelant

1 - POOLING OF INTEREST ACCOUNTING METHOD 8704

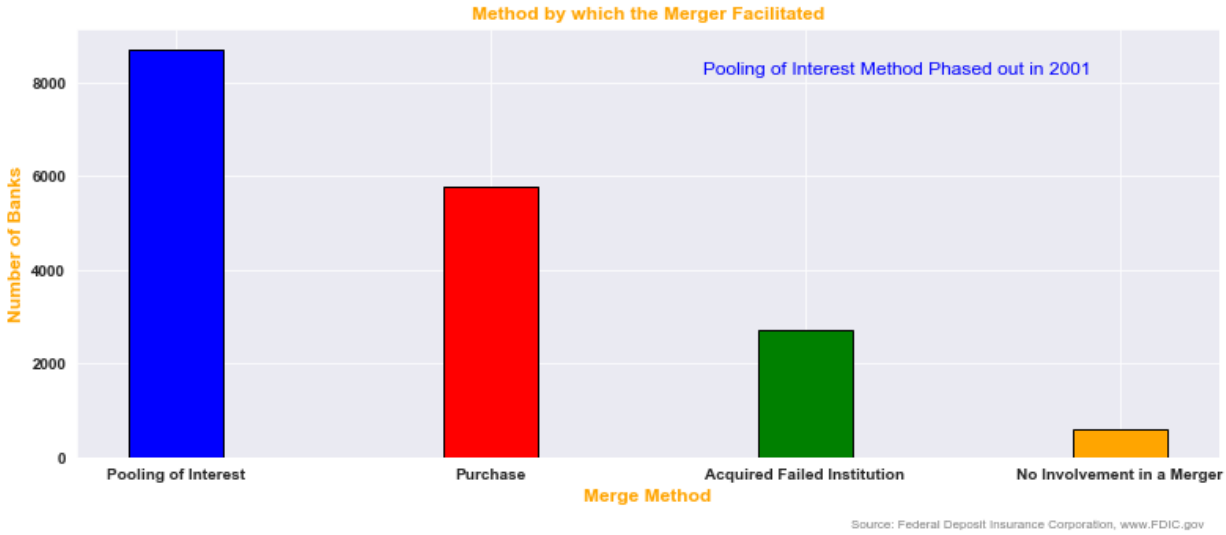
2 - PURCHASE ACCOUNTING METHOD 5775

3 - ACQUIRED FAILED INSTITUTION 2712

0 - NO INVOLVMENT IN A MERGER 606

In [64]:
```
clean_combo ['MergMethDesc'] = pd.to_numeric(clean_combo['MergMethDesc
'], errors='coerce')
```

In [464]:
```
plt.figure(figsize=(13,5))

reason = [1,2,3,4] # x-coordinates of left sides of bars
amount = [8704,5775,2712,606] #this is the height of the bars

tick_label = ['Pooling of Interest', 'Purchase', 'Acquired Failed Inst
itution', 'No Involvement in a Merger'] # this is what we are labeling
the bars

plt.bar(reason, amount, tick_label = tick_label, # plotting the bar ch
art - includes color and width
       width = 0.3, color = ['blue', 'red', 'green', 'orange' ], edgec
olor ='black') # We have made the edges of the bars a different color

plt.xlabel('Merge Method', size = 12, color = 'orange', fontweight = "
bold")
plt.ylabel('Number of Banks', size = 12, color = 'orange', fontweight
= "bold")
plt.title('Method by which the Merger Facilitated', size = 12, color =
'orange', fontweight = "bold")

plt.xticks( fontweight = "bold") #Changing the header of the axis - ch
anged from brown to just standard bold
plt.yticks( fontweight = "bold") #Changing the header of the axis - ch
anged from brown to just standard bold

plt.figtext(.55, .8, 'Pooling of Interest Method Phased out in 2001',
color = 'blue' , size = 12)

plt.figtext(.65, 0.00000001, 'Source: Federal Deposit Insurance Corpor
ation, www.FDIC.gov', color ='grey', size = 8) # Include citation

plt.show()
```


Method by which the Merger Facilitated
Pooling of Interest Method Phased out in 2001
Number of Banks
8000
6000
4000
2000
0
Pooling of Interest
Purchase
Acquired Failed Institution
No Involvement in a Merger
Merge Method
Source: Federal Deposit Insurance Corporation, www.FDIC.gov

This is a more in depth look at the 'M&A' and 'Merger Without Assistance - Consolidation' data. Here we explore the method by which the merger was facilitated. The two most frequent merge methods are 'Pooling of Interest' and 'Purchase', which are similar in concept but different in execution and financial guidance. Moreover, the accounting methods by which these two seperate transaction categories are underwritten and shown on the companys books vary.

Pooling of Interests Accounting Method

The pooling of interests method involves combining the balance sheets from the two firms into one. The assets and liabilities are recorded according to their respective account balances as recorded on the balance sheet. It is usually followed by a revaluation of the historical financial statements.This method was discontinued by the Financial Accounting Standards Board (FASB) in 2001, prior pooling of interests was the most preferred technique because it resulted in high earnings for the surviving firm.

Purchase (Acqusition) Accounting Method

Purchase acquisition is an accounting method used in mergers and acquisitions where there is no pooling of interests and the purchasing company treats the target firm as an investment. The assets of the target are valued and added to the balance sheet of the acquirer at fair value, increasing the acquirer's fair market value. Liabilities of the target are subtracted from the fair value of the assets. The amount paid by the acquirer over the net value of the target's assets and liabilities is considered goodwill, which is kept on the balance sheet and amortized yearly.

Pooling of Interests Accounting Method Vs. Purchase Method

- Pooling of interests is mainly applied when the process of combining businesses is in the nature of a merger. However, if the process is taking place in the form of a purchase, then the purchase price method is used.
- In pooling of interest, the balance sheet presents assets and liabilities at their book values. However, the purchase price approach records the fair market values of the assets and liabilities.
- Under pooling of interests, the assets and liabilities of the two businesses are combined. However, with the purchase price approach, only the assets and liabilities of the acquirer noted down.
- With pooling of interests, no changes are made to the identity of the acquired company’s reserves. However, under the latter technique, the identity of the acquired firm’s reserves will change, with the exception of the statutory reserves.

We know the types of different M&A possibilities and their frequencies, now we shall explore which institutions were responsible for the most M&A transactions in the industry

In [65]:
```
clean_combo.PROXIMATE_CERT.value_counts() # Values that repeat themsel
ves most frequently throughout the given requested column
```

Out[65]:
```
           5403
0.0         607
12368.0     108
9846.0       87
27306.0      70
4979.0       65
3511.0       62
24344.0      55
849.0        55
3011.0       51
6548.0       49
3263.0       47
867.0        45
1020.0       45
9609.0       43
15802.0      42
27476.0      41
4297.0       41
16835.0      38
11063.0      38
33184.0      36
993.0        36
16571.0      36
5208.0       35
6560.0       34
14533.0      34
2558.0       34
873.0        33
11813.0      33
3566.0       33
            ...
9064.0        1
10007.0       1
58236.0       1
9108.0        1
58181.0       1
58261.0       1
58281.0       1
9140.0        1
19408.0       1
```

```
58471.0       1
9176.0        1
9043.0        1
58173.0       1
25302.0       1
25340.0       1
58076.0       1
58081.0       1
8939.0        1
8942.0        1
25328.0       1
19006.0       1
58119.0       1
9013.0        1
8980.0        1
58136.0       1
8997.0        1
10013.0       1
4690.0        1
19393.0       1
23691.0       1
Name: PROXIMATE_CERT, Length: 6437, dtype: int64
```

In [66]:
```
top_acquirers = universe['CERT'].isin(['12368', '9846', '27306', '4979
', '3511', '24344', '849', '3011'])
```

In [67]:
```
universe[top_acquirers]
```

Out[67]:

	STNAME	CERT	ACTIVE	ADDRESS	ASSET	BKCLASS	CHANGEC1	CHANGEC
8898	Texas	27306	0	901 Main Street	62,921,254	N	223	
10739	Texas	24344	0	1000 Louisiana	7,565,976	N	223	
19757	Alabama	12368	1	1900 Fifth Avenue North	124,716,588	SM	660	
21323	North Carolina	9846	1	200 W 2nd St	219,071,000	NM	810	
24295	Tennessee	4979	0	6200 Poplar Avenue	34,722,095	N	223	
25374	South Dakota	3511	1	101 N. Phillips Avenue	1,689,351,000	N	0	
25702	Colorado	3011	0	1740 Broadway	14,100,333	N	223	
27076	Alabama	849	0	420 North 20th Street	59,720,179	SM	223	

8 rows × 35 columns

In [80]:
```
acquirers = universe[top_acquirers]
```

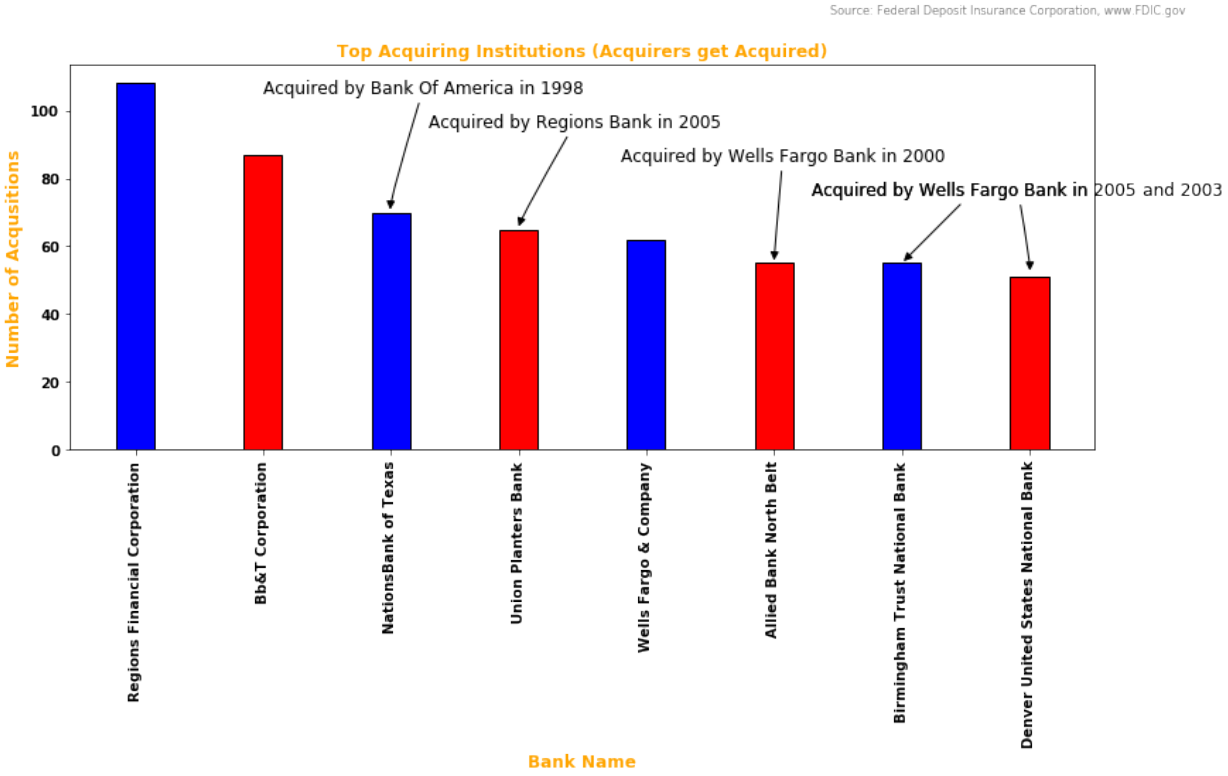
Using our data set we will examine not only who the top acquirers are, but we will research our data to see whether the acquirers have been acquired and by whom.

In [257]:
```
plt.figure(figsize=(13,5))

Bank_Name = [1,2,3,4,5,6,7,8] # x-coordinates of left sides of bars
Total_Acqusitions = [108,87,70,65,62,55,55,51] #this is the height of
the bars

tick_label = ['Regions Financial Corporation', 'Bb&T Corporation', 'Na
tionsBank of Texas',
              'Union Planters Bank', 'Wells Fargo & Company', 'Allied
Bank North Belt',
              'Birmingham Trust National Bank', 'Denver United States
National Bank'] # this is what we are labeling the bars
```

```
plt.bar(Bank_Name, Total_Acqusitions, tick_label = tick_label, # plott
ing the bar chart - includes color and width
       width = 0.3, color = ['blue', 'red',  ], edgecolor ='black') #
We have made the edges of the bars a different color

plt.xlabel('Bank Name', size = 12, color = 'orange', fontweight = "bol
d")
plt.ylabel('Number of Acqusitions', size = 12, color = 'orange', fontw
eight = "bold")
plt.title('Top Acquiring Institutions (Acquirers get Acquired)', size
= 12, color = 'orange', fontweight = "bold")

# Since our column names are long we are going to specify a rotation f
or the tick labels in degree
plt.xticks( rotation = 'vertical', fontweight = "bold") #Changing the
header of the axis - changed from brown to just standard bold
plt.yticks( fontweight = "bold") #Changing the header of the axis - ch
anged from brown to just standard bold

plt.figtext(.7, 0.98, 'Source: Federal Deposit Insurance Corporation,
www.FDIC.gov', color ='grey', size = 8) # Include citation

##########################################################################
##########

my_message = "Acquired by Bank Of America in 1998"

plt.annotate(
    my_message,
    xy=(3, 70), # This is where we point at...
    xycoords="data", # Not exactly sure about this
    xytext=(2, 105), # This is about where the text is
    horizontalalignment="left", # How the text is alined
    arrowprops={
        "arrowstyle": "-|>", # This is stuff about the arrow
        "connectionstyle": "angle3,angleA=73,angleB=0",
        "color": "black"
    },
    fontsize=12,)

##########################################################################
#########

my_message = "Acquired by Regions Bank in 2005"

plt.annotate(
    my_message,
    xy=(4, 65), # This is where we point at...
    xycoords="data", # Not exactly sure about this
    xytext=(3.3, 95), # This is about where the text is
```

```
    horizontalalignment="left", # How the text is alined
    arrowprops={
        "arrowstyle": "-|>", # This is stuff about the arrow
        "connectionstyle": "angle3,angleA=60,angleB=0",
        "color": "black"
    },
    fontsize=12,)

#########################################################################
#########

my_message = "Acquired by Wells Fargo Bank in 2000"

plt.annotate(
    my_message,
    xy=(6, 55), # This is where we point at...
    xycoords="data", # Not exactly sure about this
    xytext=(4.8, 85), # This is about where the text is
    horizontalalignment="left", # How the text is alined
    arrowprops={
        "arrowstyle": "-|>", # This is stuff about the arrow
        "connectionstyle": "angle3,angleA=85,angleB=0",
        "color": "black"
    },
    fontsize=12,)

#########################################################################
#########

my_message = "Acquired by Wells Fargo Bank in 2005"

plt.annotate(
    my_message,
    xy=(7, 55), # This is where we point at...
    xycoords="data", # Not exactly sure about this
    xytext=(6.3, 75), # This is about where the text is
    horizontalalignment="left", # How the text is alined
    arrowprops={
        "arrowstyle": "-|>", # This is stuff about the arrow
        "connectionstyle": "angle3,angleA=45,angleB=0",
        "color": "black"
    },
    fontsize=12,)

#########################################################################
#########
my_message = "Acquired by Wells Fargo Bank in          and 2003"

plt.annotate(
    my_message,
```

```
    xy=(8, 52), # This is where we point at...
    xycoords="data", # Not exactly sure about this
    xytext=(6.3, 75), # This is about where the text is
    horizontalalignment="left", # How the text is alined
    arrowprops={
        "arrowstyle": "-|>", # This is stuff about the arrow
        "connectionstyle": "angle3,angleA=100,angleB=0",
        "color": "black"
    },
    fontsize=12,)

plt.show()
```



Do our totals really matter?

If 5 out of the 8 top community bank acquirers were themselves acquired, then the actual acquiring rates are much higher for the largest institutions.

Now we will examine the asset holdings of these top acquirers among eachother

Big Banks Just Get Bigger and Bigger

In [103]:
```
acquirers = universe[top_acquirers].copy() # We need to copy the data set so we are able to work with it
```

In [105]:
```
acquirers['NAMEHCR']
```

Out[105]:
```
8898                                  NaN
10739                                 NaN
19757    Regions Financial Corporation
21323                 Bb&T Corporation
24295                                 NaN
25374           Wells Fargo & Company
25702                                 NaN
27076                                 NaN
Name: NAMEHCR, dtype: object
```

In [106]:
```
acquirers['NAMEHCR'] = ['NationsBank of Texas','Allied Bank North Belt','Regions Financial Corporation', 'Bb&T Corporation',
              'Union Planters Bank', 'Wells Fargo & Company',
              'Birmingham Trust National Bank', 'Denver United States National Bank']

# Allowing us to examine the dataset but it is a portion of a larger data set hence we need to make a copy of this data set so we can work with it
```

In [107]:
```
acquirers
```

Out[107]:

	STNAME	CERT	ACTIVE	ADDRESS	ASSET	BKCLASS	CHANGEC1	CHANGEC
8898	Texas	27306	0	901 Main Street	62,921,254	N	223	
10739	Texas	24344	0	1000 Louisiana	7,565,976	N	223	
19757	Alabama	12368	1	1900 Fifth Avenue North	124,716,588	SM	660	
21323	North Carolina	9846	1	200 W 2nd St	219,071,000	NM	810	
24295	Tennessee	4979	0	6200 Poplar Avenue	34,722,095	N	223	
25374	South Dakota	3511	1	101 N. Phillips Avenue	1,689,351,000	N	0	
25702	Colorado	3011	0	1740 Broadway	14,100,333	N	223	
27076	Alabama	849	0	420 North 20th Street	59,720,179	SM	223	

8 rows × 35 columns

In [102]:
```
acquirers['ASSET']
```

Out[102]:
```
8898    NaN
10739   NaN
19757   NaN
21323   NaN
24295   NaN
25374   NaN
25702   NaN
27076   NaN
Name: ASSET, dtype: float64
```

In [108]:
```
acquirers['NAMEHCR']
```

Out[108]:
```
8898                    NationsBank of Texas
10739                Allied Bank North Belt
19757          Regions Financial Corporation
21323                       Bb&T Corporation
24295                    Union Planters Bank
25374                  Wells Fargo & Company
25702          Birmingham Trust National Bank
27076    Denver United States National Bank
Name: NAMEHCR, dtype: object
```

In [109]:
```
acquirers.dtypes # We check the types since column 'ASSET' is not what we need it to be but we have changed it to an integer
```

Out[109]:
```
STNAME       object
CERT          int64
ACTIVE        int64
ADDRESS      object
ASSET        object
BKCLASS      object
CHANGEC1      int64
CHANGEC2      int64
CHANGEC3      int64
CHANGEC4      int64
CHANGEC5      int64
CITY         object
COUNTY       object
DEP          object
ENDEFYMD     object
EQ           object
ESTYMD       object
INACTIVE      int64
INSTCRCD      int64
NAME         object
ROA         float64
ROE         float64
STALP        object
STCNTY        int64
ULTCERT       int64
WEBADDR      object
DEPDOM       object
MUTUAL      float64
NAMEHCR      object
NETINC       object
NETINCQ      object
OFFDOM      float64
OFFFOR      float64
ROAPTX      float64
SPECGRP     float64
dtype: object
```

In [112]:
```
acquirers['ASSET'] = acquirers['ASSET'].str.replace(',','').astype(int
) # We need to replace the ',' in the 'ASSET' column with
```

In [113]:
```
acquirers
```

Out[113]:

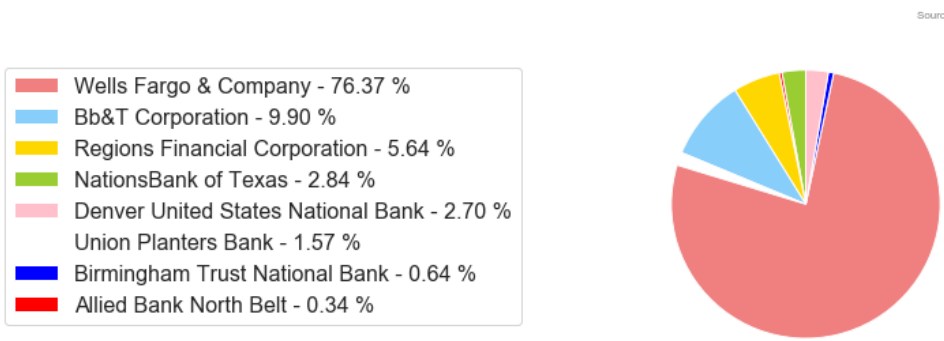
	STNAME	CERT	ACTIVE	ADDRESS	ASSET	BKCLASS	CHANGEC1	CHANGEC2
8898	Texas	27306	0	901 Main Street	62921254	N	223	0
10739	Texas	24344	0	1000 Louisiana	7565976	N	223	0
19757	Alabama	12368	1	1900 Fifth Avenue North	124716588	SM	660	0
21323	North Carolina	9846	1	200 W 2nd St	219071000	NM	810	0
24295	Tennessee	4979	0	6200 Poplar Avenue	34722095	N	223	0
25374	South Dakota	3511	1	101 N. Phillips Avenue	1689351000	N	0	0
25702	Colorado	3011	0	1740 Broadway	14100333	N	223	0
27076	Alabama	849	0	420 North 20th Street	59720179	SM	223	0

8 rows × 35 columns

In [114]:
```
acquirers.dtypes
```

Out[114]:
```
STNAME       object
CERT          int64
ACTIVE        int64
ADDRESS      object
ASSET         int64
BKCLASS      object
CHANGEC1      int64
CHANGEC2      int64
CHANGEC3      int64
CHANGEC4      int64
CHANGEC5      int64
CITY         object
COUNTY       object
DEP          object
ENDEFYMD     object
EQ           object
ESTYMD       object
INACTIVE      int64
INSTCRCD      int64
NAME         object
ROA         float64
ROE         float64
STALP        object
STCNTY        int64
ULTCERT       int64
WEBADDR      object
DEPDOM       object
MUTUAL      float64
NAMEHCR      object
NETINC       object
NETINCQ      object
OFFDOM      float64
OFFFOR      float64
ROAPTX      float64
SPECGRP     float64
dtype: object
```

In [531]:
```
fig, ax = plt.subplots()

# ax.pie( acquirers['ASSET'].values,
        # shadow=True, startangle=50, explode = (0.1, 0.1, 0, 0, 0.1,
0, 0, 0))

y = acquirers['ASSET'].values
x = acquirers['NAMEHCR'].values
plt.title

colors = ['yellowgreen','red','gold','lightskyblue','white','lightcora
l','blue','pink', 'darkgreen','yellow','grey','violet','magenta','cyan
']
percent = 100.*y/y.sum()

patches, texts = plt.pie(y, colors=colors, startangle=90, radius= 1)
labels = ['{0} - {1:1.2f} %'.format(i,j) for i,j in zip(x, percent)]

sort_legend = True
if sort_legend:
    patches, labels, dummy =  zip(*sorted(zip(patches, labels, y),
                                          key=lambda x: x[2],
                                          reverse=True))

plt.legend(patches, labels, loc='best', bbox_to_anchor=(-0.1, 1.),
           fontsize=16)

plt.figtext(.7, 0.98, 'Source: Federal Deposit Insurance Corporation,
www.FDIC.gov', color ='grey', size = 8) # Include citation



plt.axis('equal')
plt.show()
```



Now we will examine the deposit holdings and some other key financials of these top 3 acquirers who are still active

In [508]:
```
acquirers
```

Out[508]:

	STNAME	CERT	ACTIVE	ADDRESS	ASSET	BKCLASS	CHANGEC1	CHANGEC2
8898	Texas	27306	0	901 Main Street	62921254	N	223	0
10739	Texas	24344	0	1000 Louisiana	7565976	N	223	0
19757	Alabama	12368	1	1900 Fifth Avenue North	124716588	SM	660	0
21323	North Carolina	9846	1	200 W 2nd St	219071000	NM	810	0
24295	Tennessee	4979	0	6200 Poplar Avenue	34722095	N	223	0
25374	South Dakota	3511	1	101 N. Phillips Avenue	1689351000	N	0	0
25702	Colorado	3011	0	1740 Broadway	14100333	N	223	0
27076	Alabama	849	0	420 North 20th Street	59720179	SM	223	0

8 rows × 35 columns

In [517]:
```
active_top_acquirers = universe['CERT'].isin(['12368', '9846', '3511']
) # We only want to work with the institutions who are currently activ
e
```

In [518]:
```
universe[active_top_acquirers]
```

Out[518]:

	STNAME	CERT	ACTIVE	ADDRESS	ASSET	BKCLASS	CHANGEC1	CHANGEC2
19757	Alabama	12368	1	1900 Fifth Avenue North	124,716,588	SM	660	0
21323	North Carolina	9846	1	200 W 2nd St	219,071,000	NM	810	0
25374	South Dakota	3511	1	101 N. Phillips Avenue	1,689,351,000	N	0	0

3 rows × 35 columns

In [519]:
```
active_acquirers = universe[active_top_acquirers].copy()
```

In [521]:
```
active_acquirers = universe[active_top_acquirers].copy()
```

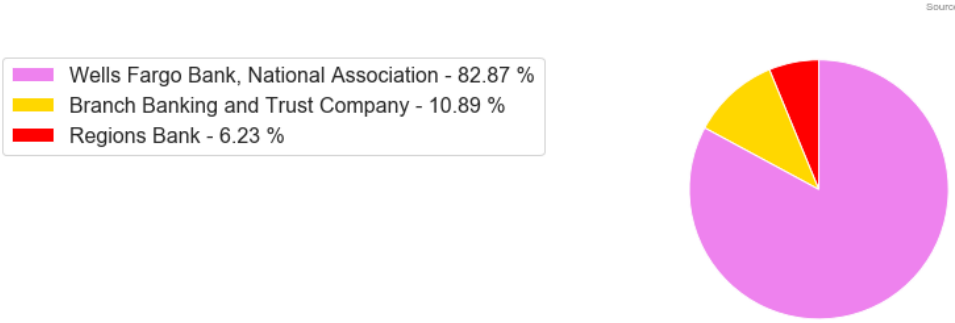
In [522]:
```
active_acquirers['DEPDOM']
```

Out[522]:
```
19757       96,461,044
21323      168,539,000
25374    1,282,404,000
Name: DEPDOM, dtype: object
```

In [523]:
```
active_acquirers['DEPDOM'] = active_acquirers['DEPDOM'].str.replace(',',
'','').astype(int) # We need to replace the ',' in the 'ASSET' column w
ith
```

These are the total deposits each firm holds among the top acquirers as of EOY 2018

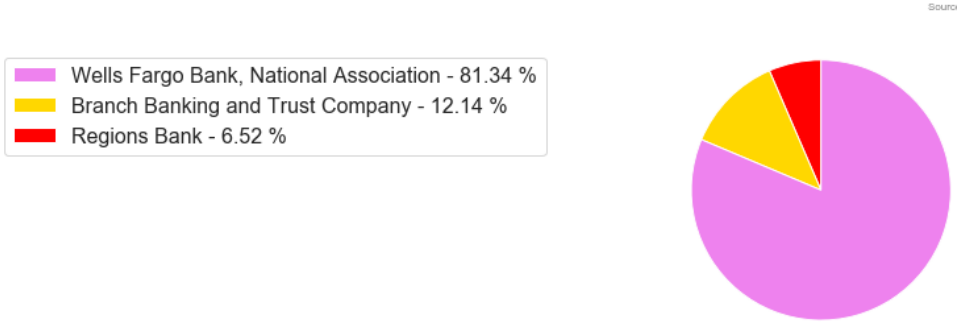
In [532]:
```
fig, ax = plt.subplots()

# ax.pie( acquirers['ASSET'].values,
        # shadow=True, startangle=50, explode = (0.1, 0.1, 0, 0, 0.1,
0, 0, 0))

y = active_acquirers['DEPDOM'].values
x = active_acquirers['NAME'].values
plt.title

colors = ['red','gold','violet']
percent = 100.*y/y.sum()

patches, texts = plt.pie(y, colors=colors, startangle=90, radius= 1)
labels = ['{0} - {1:1.2f} %'.format(i,j) for i,j in zip(x, percent)]

sort_legend = True
if sort_legend:
    patches, labels, dummy =  zip(*sorted(zip(patches, labels, y),
                                          key=lambda x: x[2],
                                          reverse=True))

plt.legend(patches, labels, loc='best', bbox_to_anchor=(-0.1, 1.),
           fontsize=16)
plt.figtext(.7, 0.98, 'Source: Federal Deposit Insurance Corporation,
www.FDIC.gov', color ='grey', size = 8) # Include citation



plt.axis('equal')
plt.show()
```



This is the percent of Net Income for each firm among the top acquirers as of EOY 2018

In [525]:
```
active_acquirers['NETINC'] = active_acquirers['NETINC'].str.replace(',
','').astype(int) # We need to replace the ',' in the 'ASSET' column w
ith
```

In [533]:
```
fig, ax = plt.subplots()

# ax.pie( acquirers['ASSET'].values,
        # shadow=True, startangle=50, explode = (0.1, 0.1, 0, 0, 0.1,
0, 0, 0))

y = active_acquirers['NETINC'].values
x = active_acquirers['NAME'].values
plt.title

colors = ['red','gold','violet']
percent = 100.*y/y.sum()

patches, texts = plt.pie(y, colors=colors, startangle=90, radius= 1)
labels = ['{0} - {1:1.2f} %'.format(i,j) for i,j in zip(x, percent)]

sort_legend = True
if sort_legend:
    patches, labels, dummy =  zip(*sorted(zip(patches, labels, y),
                                          key=lambda x: x[2],
                                          reverse=True))

plt.legend(patches, labels, loc='best', bbox_to_anchor=(-0.1, 1.),
           fontsize=16)

plt.figtext(.7, 0.98, 'Source: Federal Deposit Insurance Corporation,
www.FDIC.gov', color ='grey', size = 8) # Include citation



plt.axis('equal')
plt.show()
```



Diminishment in the number of Community Banks and why thats important

'Even though over the past 25 years there has been tremendous consolidation in the U.S. banking industry, 5,000 banks is still many more banks than in most other countries. So, it seems likely that further consolidation will take place, and the number of banks will continue to dwindle. The decline in the number of banks understates the degree of consolidation in the U.S. banking industry as the largest banks have a disproportionate share of all deposits, with the top 3 banks having more than one-third.'

'Larger banks may fail to provide the same services that community banks do. Many large banks have cut back on small business loans, preferring to concentrate on loans of at least 1 million, which cost the bank about the same to process as do smaller loans of 100,000 or less. From the peak before the financial crisis, lending to small businesses by the largest banks has declined by about 40%.'

'One of the reasons that large banks have reduced their lending to small businesses is that small business loans are more difficult to securitize than are mortage loans or balances on credit cards. Accordingly, some large banks have pushed their small business borrowers to rely on bank-issued credit cards, whcih carry higher interest rates, tharther than on traditional bank loans.'

Role of Community Banks

'Community banks attract small business borrowers by being more flexible in the standards they apply in granting loans. The standardized loan approval software used by most large banks ofte nrequires small businesses applying for loans to supply 10 years worth of financial data - an impossibility for a small startup. Although community banks may charge higher rates on business loans than do large banks, the interest rates are still lower than on credit cards or on most online loans. In addition, community banks are often more flexible in allowing a borrow to miss a loan payment or overdraw a credit line without severe penalties.'

'Community banks are highly capitalized, so they’re better prepared than their larger competitors for economic crises. And as local institutions, they reinvest in their communities and channel loans to their depositors’ neighborhoods—promoting localized growth that radiates out to the broader economy. Community banks have been instrumental in helping the nation recover from the financial crisis.'

'Currently, community banks offer clear advanages over larger banks for entrepreneurs seeking small business loans. But their small scale leaves them vulnerable in the future both to large banks developing lower - cost ways of offering business loans of $100,000 or less.'

--- Class Textbook, cited underneath

References

1.“Federal Deposit Insurance Corporation.” FDIC, www.fdic.gov/regulations/resources/cbi/data/public-ref-readme.html.

1. “Federal Deposit Insurance Corporation.” FDIC, research.fdic.gov/bankfind/detail.html?bank=3510&name=Bank%2Bof%2BAmerica%2C%2BNational%2BAssociation&searchName=BANK%2E
2. “A Computer Science Portal for Geeks.” GeeksforGeeks, www.geeksforgeeks.org/.
3. “Where Developers Learn, Share, & Build Careers.” Stack Overflow, stackoverflow.com/.
4. “Statistical Data Visualization¶.” Seaborn, seaborn.pydata.org/.
5. O'Brien, Anthony Patrick., and R. Glenn. Hubbard. Instructor's Resource Manual and Test Bank to Accompany Money, the Financial System, and the Economy, Second Edition, R. Glenn Hubbard. 3rd ed., Addison Wesley, 2016.

In [ ]: