climate normals of Austin,TX from 1981-2010

and weather data of Austin, TX from 2011

In [1]:
```
import pandas as pd
```

In [2]:
```
ls
```

```
'sky_conditionFlag', 'visibilityFlag', 'wx_and_obs
NOAA_QCLCD_2011_hourly_13904.txt
Untitled.ipynb
Untitled1.ipynb
Untitled2.ipynb
Untitled3.ipynb
Untitled4.ipynb
Untitled5.ipynb
Untitled6.ipynb
austin_TX_weather_and_climate_analysis.ipynb
austin_airport_departure_data_2015_july.csv
auto-mpg.csv
clean_stock_data.csv
column_labels
life_expectancy_at_birth.csv
messy_stock_data.tsv
percent-bachelors-degrees-women-usa.csv
tips.csv
titanic.csv
weather_data_austin_2010.csv
world_ind_pop_data.csv
world_population.csv
```

In [3]:
```
df = pd.read_csv('NOAA_QCLCD_2011_hourly_13904.txt')
```

In [4]:
```
df.head(9)
```

Out[4]:

	13904	20110101	0053	12	OVC045		10.00	.1	.2	.3	...	.18	.19	29.95	.20	AA	.21	.22	.23	29.9
0	13904	20110101	153	12	OVC049		10.00				...			30.01		AA				30.0
1	13904	20110101	253	12	OVC060		10.00				...	030		30.01		AA				30.0
2	13904	20110101	353	12	OVC065		10.00				...			30.03		AA				30.0
3	13904	20110101	453	12	BKN070		10.00				...			30.04		AA				30.0
4	13904	20110101	553	12	BKN065		10.00				...	015		30.06		AA				30.0
5	13904	20110101	653	12	BKN065		10.00				...			30.10		AA				30.1
6	13904	20110101	753	12	SCT060		10.00				...			30.12		AA				30.1
7	13904	20110101	853	12	FEW060		10.00				...	034		30.16		AA				30.1
8	13904	20110101	953	12	FEW060		10.00				...			30.19		AA				30.1

9 rows × 44 columns

In [5]:
```
df = pd.read_csv('NOAA_QCLCD_2011_hourly_13904.txt',header=None)
```

In [6]:
```
df.head(9)
```

Out[6]:

	0	1	2	3	4	5	6	7	8	9	...	34	35	36	37	38	39	40	41	42	43
0	13904	20110101	53	12	OVC045		10.00				...			29.95		AA				29.95	
1	13904	20110101	153	12	OVC049		10.00				...			30.01		AA				30.02	
2	13904	20110101	253	12	OVC060		10.00				...	030		30.01		AA				30.02	
3	13904	20110101	353	12	OVC065		10.00				...			30.03		AA				30.04	
4	13904	20110101	453	12	BKN070		10.00				...			30.04		AA				30.04	
5	13904	20110101	553	12	BKN065		10.00				...	015		30.06		AA				30.06	
6	13904	20110101	653	12	BKN065		10.00				...			30.10		AA				30.10	
7	13904	20110101	753	12	SCT060		10.00				...			30.12		AA				30.12	
8	13904	20110101	853	12	FEW060		10.00				...	034		30.16		AA				30.16	

9 rows × 44 columns

In [7]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10337 entries, 0 to 10336
Data columns (total 44 columns):
0     10337 non-null int64
1     10337 non-null int64
2     10337 non-null int64
3     10337 non-null int64
4     10337 non-null object
5     10337 non-null object
6     10325 non-null object
7     10337 non-null object
8     10337 non-null object
9     10337 non-null object
10    10337 non-null object
11    10337 non-null object
12    10337 non-null object
13    10337 non-null object
14    10337 non-null object
15    10337 non-null object
16    10337 non-null object
17    10337 non-null object
18    10337 non-null object
19    10337 non-null object
20    10337 non-null object
21    10337 non-null object
22    10337 non-null object
23    10337 non-null object
24    10337 non-null object
25    10337 non-null object
26    10337 non-null object
27    10337 non-null object
28    10337 non-null object
29    10337 non-null object
30    10337 non-null object
31    10337 non-null object
32    10337 non-null object
33    10337 non-null object
34    10337 non-null object
35    10337 non-null object
36    10337 non-null object
37    10337 non-null object
38    10337 non-null object
39    10337 non-null object
40    10337 non-null object
41    10337 non-null object
42    10337 non-null object
43    10337 non-null object
dtypes: int64(4), object(40)
memory usage: 3.5+ MB
```

In [8]:
```
column_labels = 'Wban,date,Time,StationType,sky_condition,sky_conditionFlag,visibi
lity,visibilityFlag,wx_and_obst_to_vision,wx_and_obst_to_visionFlag,dry_bulb_faren
,dry_bulb_farenFlag,dry_bulb_cel,dry_bulb_celFlag,wet_bulb_faren,wet_bulb_farenFla
g,wet_bulb_cel,wet_bulb_celFlag,dew_point_faren,dew_point_farenFlag,dew_point_cel,
dew_point_celFlag,relative_humidity,relative_humidityFlag,wind_speed,wind_speedFla
g,wind_direction,wind_directionFlag,value_for_wind_character,value_for_wind_charac
terFlag,station_pressure,station_pressureFlag,pressure_tendency,pressure_tendencyF
lag,presschange,presschangeFlag,sea_level_pressure,sea_level_pressureFlag,record_t
ype,hourly_precip,hourly_precipFlag,altimeter,altimeterFlag,junk'
```

In [9]:
```
type(column_labels)
```

Out[9]:
```
str
```

In [10]:
```
column_label_list = column_labels.split(',')
```

In [11]:
```
column_label_list
```

Out[11]:
```
['Wban',
 'date',
 'Time',
 'StationType',
 'sky_condition',
 'sky_conditionFlag',
 'visibility',
 'visibilityFlag',
 'wx_and_obst_to_vision',
 'wx_and_obst_to_visionFlag',
 'dry_bulb_faren',
 'dry_bulb_farenFlag',
 'dry_bulb_cel',
 'dry_bulb_celFlag',
 'wet_bulb_faren',
 'wet_bulb_farenFlag',
 'wet_bulb_cel',
 'wet_bulb_celFlag',
 'dew_point_faren',
 'dew_point_farenFlag',
 'dew_point_cel',
 'dew_point_celFlag',
 'relative_humidity',
 'relative_humidityFlag',
 'wind_speed',
 'wind_speedFlag',
 'wind_direction',
 'wind_directionFlag',
 'value_for_wind_character',
 'value_for_wind_characterFlag',
 'station_pressure',
 'station_pressureFlag',
 'pressure_tendency',
 'pressure_tendencyFlag',
 'presschange',
 'presschangeFlag',
 'sea_level_pressure',
 'sea_level_pressureFlag',
 'record_type',
 'hourly_precip',
 'hourly_precipFlag',
 'altimeter',
 'altimeterFlag',
 'junk']
```

In [12]:
```
df.columns = column_label_list
```

In [13]:
```
df.head(9)
```

Out[13]:

	Wban	date	Time	StationType	sky_condition	sky_conditionFlag	visibility	visibilityFlag	wx_a
0	13904	20110101	53	12	OVC045		10.00		
1	13904	20110101	153	12	OVC049		10.00		
2	13904	20110101	253	12	OVC060		10.00		
3	13904	20110101	353	12	OVC065		10.00		
4	13904	20110101	453	12	BKN070		10.00		
5	13904	20110101	553	12	BKN065		10.00		
6	13904	20110101	653	12	BKN065		10.00		
7	13904	20110101	753	12	SCT060		10.00		
8	13904	20110101	853	12	FEW060		10.00		

9 rows × 44 columns

In [14]:
```
list_to_drop = ['sky_conditionFlag', 'visibilityFlag', 'wx_and_obst_to_vision', 'w
x_and_obst_to_visionFlag', 'dry_bulb_farenFlag', 'dry_bulb_celFlag', 'wet_bulb_far
enFlag', 'wet_bulb_celFlag', 'dew_point_farenFlag', 'dew_point_celFlag', 'relative
_humidityFlag', 'wind_speedFlag', 'wind_directionFlag', 'value_for_wind_character'
, 'value_for_wind_characterFlag', 'station_pressureFlag', 'pressure_tendencyFlag',
'pressure_tendency', 'presschange', 'presschangeFlag', 'sea_level_pressureFlag', '
hourly_precip', 'hourly_precipFlag', 'altimeter', 'record_type', 'altimeterFlag',
'junk']
```

In [15]:
```
list_to_drop
```

Out[15]:
```
['sky_conditionFlag',
 'visibilityFlag',
 'wx_and_obst_to_vision',
 'wx_and_obst_to_visionFlag',
 'dry_bulb_farenFlag',
 'dry_bulb_celFlag',
 'wet_bulb_farenFlag',
 'wet_bulb_celFlag',
 'dew_point_farenFlag',
 'dew_point_celFlag',
 'relative_humidityFlag',
 'wind_speedFlag',
 'wind_directionFlag',
 'value_for_wind_character',
 'value_for_wind_characterFlag',
 'station_pressureFlag',
 'pressure_tendencyFlag',
 'pressure_tendency',
 'presschange',
 'presschangeFlag',
 'sea_level_pressureFlag',
 'hourly_precip',
 'hourly_precipFlag',
 'altimeter',
 'record_type',
 'altimeterFlag',
 'junk']
```

In [16]:
```
df_dropped = df.drop(list_to_drop,axis='columns')
```

In [17]:
```
df_dropped.head(9)
```

Out[17]:

	Wban	date	Time	StationType	sky_condition	visibility	dry_bulb_faren	dry_bulb_cel	wet_bu
0	13904	20110101	53	12	OVC045	10.00	51	10.6	38
1	13904	20110101	153	12	OVC049	10.00	51	10.6	37
2	13904	20110101	253	12	OVC060	10.00	51	10.6	37
3	13904	20110101	353	12	OVC065	10.00	50	10.0	38
4	13904	20110101	453	12	BKN070	10.00	50	10.0	37
5	13904	20110101	553	12	BKN065	10.00	49	9.4	37
6	13904	20110101	653	12	BKN065	10.00	48	8.9	37
7	13904	20110101	753	12	SCT060	10.00	48	8.9	37
8	13904	20110101	853	12	FEW060	10.00	51	10.6	38

In [18]:
```
df_dropped.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10337 entries, 0 to 10336
Data columns (total 17 columns):
Wban                  10337 non-null int64
date                  10337 non-null int64
Time                  10337 non-null int64
StationType           10337 non-null int64
sky_condition         10337 non-null object
visibility            10325 non-null object
dry_bulb_faren        10337 non-null object
dry_bulb_cel          10337 non-null object
wet_bulb_faren        10337 non-null object
wet_bulb_cel          10337 non-null object
dew_point_faren       10337 non-null object
dew_point_cel         10337 non-null object
relative_humidity     10337 non-null object
wind_speed            10337 non-null object
wind_direction        10337 non-null object
station_pressure      10337 non-null object
sea_level_pressure    10337 non-null object
dtypes: int64(4), object(13)
memory usage: 1.3+ MB
```

In [19]:
```
# Convert the date column to string: df_dropped['date']
df_dropped['date'] = df_dropped['date'].astype(str)
```

In [20]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10337 entries, 0 to 10336
Data columns (total 44 columns):
Wban                            10337 non-null int64
date                            10337 non-null int64
Time                            10337 non-null int64
StationType                     10337 non-null int64
sky_condition                   10337 non-null object
sky_conditionFlag               10337 non-null object
visibility                      10325 non-null object
visibilityFlag                  10337 non-null object
wx_and_obst_to_vision           10337 non-null object
wx_and_obst_to_visionFlag       10337 non-null object
dry_bulb_faren                  10337 non-null object
dry_bulb_farenFlag              10337 non-null object
dry_bulb_cel                    10337 non-null object
dry_bulb_celFlag                10337 non-null object
wet_bulb_faren                  10337 non-null object
wet_bulb_farenFlag              10337 non-null object
wet_bulb_cel                    10337 non-null object
wet_bulb_celFlag                10337 non-null object
dew_point_faren                 10337 non-null object
dew_point_farenFlag             10337 non-null object
dew_point_cel                   10337 non-null object
dew_point_celFlag               10337 non-null object
relative_humidity               10337 non-null object
relative_humidityFlag           10337 non-null object
wind_speed                      10337 non-null object
wind_speedFlag                  10337 non-null object
wind_direction                  10337 non-null object
wind_directionFlag              10337 non-null object
value_for_wind_character        10337 non-null object
value_for_wind_characterFlag    10337 non-null object
station_pressure                10337 non-null object
station_pressureFlag            10337 non-null object
pressure_tendency               10337 non-null object
pressure_tendencyFlag           10337 non-null object
presschange                     10337 non-null object
presschangeFlag                 10337 non-null object
sea_level_pressure              10337 non-null object
sea_level_pressureFlag          10337 non-null object
record_type                     10337 non-null object
hourly_precip                   10337 non-null object
hourly_precipFlag               10337 non-null object
altimeter                       10337 non-null object
altimeterFlag                   10337 non-null object
junk                            10337 non-null object
dtypes: int64(4), object(40)
memory usage: 3.5+ MB
```

In [21]:
```
df_dropped.head(9)
```

Out[21]:

	Wban	date	Time	StationType	sky_condition	visibility	dry_bulb_faren	dry_bulb_cel	wet_bu
0	13904	20110101	53	12	OVC045	10.00	51	10.6	38
1	13904	20110101	153	12	OVC049	10.00	51	10.6	37
2	13904	20110101	253	12	OVC060	10.00	51	10.6	37
3	13904	20110101	353	12	OVC065	10.00	50	10.0	38
4	13904	20110101	453	12	BKN070	10.00	50	10.0	37
5	13904	20110101	553	12	BKN065	10.00	49	9.4	37
6	13904	20110101	653	12	BKN065	10.00	48	8.9	37
7	13904	20110101	753	12	SCT060	10.00	48	8.9	37
8	13904	20110101	853	12	FEW060	10.00	51	10.6	38

In [22]:
```
# Padding leading zeros to the Time column: df_dropped['Time']
df_dropped['Time'] = df_dropped['Time'].apply(lambda x:'{:0>4}'.format(x))
```

In [23]:
```
df_dropped.head(9)
```

Out[23]:

	Wban	date	Time	StationType	sky_condition	visibility	dry_bulb_faren	dry_bulb_cel	wet_bu
0	13904	20110101	0053	12	OVC045	10.00	51	10.6	38
1	13904	20110101	0153	12	OVC049	10.00	51	10.6	37
2	13904	20110101	0253	12	OVC060	10.00	51	10.6	37
3	13904	20110101	0353	12	OVC065	10.00	50	10.0	38
4	13904	20110101	0453	12	BKN070	10.00	50	10.0	37
5	13904	20110101	0553	12	BKN065	10.00	49	9.4	37
6	13904	20110101	0653	12	BKN065	10.00	48	8.9	37
7	13904	20110101	0753	12	SCT060	10.00	48	8.9	37
8	13904	20110101	0853	12	FEW060	10.00	51	10.6	38

In [24]:
```
# the new date and Time columns: date_string
date_string = df_dropped['date']+df_dropped['Time']
```

In [25]:
```
# the date_string Series to datetime: date_times
date_times = pd.to_datetime(date_string, format='%Y%m%d%H%M')
```

In [26]:
```
# the index to be the new date_times container: df_clean
df_clean = df_dropped.set_index(date_times)
```

In [27]:
```
df_clean.head(9)
```

Out[27]:

	Wban	date	Time	StationType	sky_condition	visibility	dry_bulb_faren	dry_bulb_ce
2011-01-01 00:53:00	13904	20110101	0053	12	OVC045	10.00	51	10.6
2011-01-01 01:53:00	13904	20110101	0153	12	OVC049	10.00	51	10.6
2011-01-01 02:53:00	13904	20110101	0253	12	OVC060	10.00	51	10.6
2011-01-01 03:53:00	13904	20110101	0353	12	OVC065	10.00	50	10.0
2011-01-01 04:53:00	13904	20110101	0453	12	BKN070	10.00	50	10.0
2011-01-01 05:53:00	13904	20110101	0553	12	BKN065	10.00	49	9.4
2011-01-01 06:53:00	13904	20110101	0653	12	BKN065	10.00	48	8.9
2011-01-01 07:53:00	13904	20110101	0753	12	SCT060	10.00	48	8.9
2011-01-01 08:53:00	13904	20110101	0853	12	FEW060	10.00	51	10.6

In [28]:
```
# dry_bulb_faren temperature between 8 AM and 9 AM on June 20, 2011
df_clean.loc['2011-6-20 08:00:00':'2011-6-20 09:00:00', 'dry_bulb_faren']
```

Out[28]:
```
2011-06-20 08:27:00     M
2011-06-20 08:28:00     M
2011-06-20 08:29:00     M
2011-06-20 08:30:00     M
2011-06-20 08:31:00     M
2011-06-20 08:32:00     M
2011-06-20 08:33:00     M
2011-06-20 08:34:00     M
2011-06-20 08:35:00     M
2011-06-20 08:53:00    83
Name: dry_bulb_faren, dtype: object
```

In [29]:
```
# converting dry_bulb_faren column to numeric values: df_clean['dry_bulb_faren']
df_clean.dry_bulb_faren = pd.to_numeric(df_clean.dry_bulb_faren,errors='coerce')
```

In [30]:
```
df_clean.loc['2011-6-20 08:00:00':'2011-6-20 09:00:00', 'dry_bulb_faren']
```

Out[30]:
```
2011-06-20 08:27:00     NaN
2011-06-20 08:28:00     NaN
2011-06-20 08:29:00     NaN
2011-06-20 08:30:00     NaN
2011-06-20 08:31:00     NaN
2011-06-20 08:32:00     NaN
2011-06-20 08:33:00     NaN
2011-06-20 08:34:00     NaN
2011-06-20 08:35:00     NaN
2011-06-20 08:53:00    83.0
Name: dry_bulb_faren, dtype: float64
```

In [31]:
```
# Converting the wind_speed and dew_point_faren columns to numeric values
df.wind_speed = pd.to_numeric(df_clean.wind_speed,errors='coerce')
df.dew_point_faren = pd.to_numeric(df_clean.dew_point_faren,errors='coerce')
```

In [32]:
```
'''Now I have the data read and cleaned, you can begin with statistical Explotary
Data Analysis.
First, i will analyze the 2011 Austin weather data.
'''
```

Out[32]:
```
'Now I have the data read and cleaned, you can begin with statistical Explotary
Data Analysis. \nFirst, i will analyze the 2011 Austin weather data.\n'
```

In [33]:
```
#median of the dry_bulb_faren column
df_clean.dry_bulb_faren.median()
```

Out[33]:
```
72.0
```

In [34]:
```
#median of the dry_bulb_faren column for the time range '2011-Apr':'2011-Jun'
df_clean.loc['2011-Apr':'2011-Jun','dry_bulb_faren'].median()
```

Out[34]:
```
78.0
```

In [35]:
```
#median of the dry_bulb_faren column for the month of January
df_clean.loc['2011-Jan','dry_bulb_faren'].median()
```

Out[35]:
```
48.0
```

In [36]:
```
#Downsampling df_clean by day and aggregate by mean: daily_mean_2011
daily_mean_2011 = df_clean.resample('D').mean()
```

In [37]:
```
daily_mean_2011.head(9)
```

Out[37]:

	Wban	StationType	dry_bulb_faren
2011-01-01	13904	12	50.166667
2011-01-02	13904	12	39.416667
2011-01-03	13904	12	46.846154
2011-01-04	13904	12	53.367347
2011-01-05	13904	12	57.965517
2011-01-06	13904	12	46.958333
2011-01-07	13904	12	51.916667
2011-01-08	13904	12	51.814815
2011-01-09	13904	12	43.613636

In [38]:
```
df_clean.head(9)
```

Out[38]:

	Wban	date	Time	StationType	sky_condition	visibility	dry_bulb_faren	dry_bulb_ce
2011-01-01 00:53:00	13904	20110101	0053	12	OVC045	10.00	51.0	10.6
2011-01-01 01:53:00	13904	20110101	0153	12	OVC049	10.00	51.0	10.6
2011-01-01 02:53:00	13904	20110101	0253	12	OVC060	10.00	51.0	10.6
2011-01-01 03:53:00	13904	20110101	0353	12	OVC065	10.00	50.0	10.0
2011-01-01 04:53:00	13904	20110101	0453	12	BKN070	10.00	50.0	10.0
2011-01-01 05:53:00	13904	20110101	0553	12	BKN065	10.00	49.0	9.4
2011-01-01 06:53:00	13904	20110101	0653	12	BKN065	10.00	48.0	8.9
2011-01-01 07:53:00	13904	20110101	0753	12	SCT060	10.00	48.0	8.9
2011-01-01 08:53:00	13904	20110101	0853	12	FEW060	10.00	51.0	10.6

In [39]:
```
daily_temp_2011 = daily_mean_2011.dry_bulb_faren.values
```

In [40]:
```
daily_temp_2011
```

```
Out[40]: array([50.16666667, 39.41666667, 46.84615385, 53.36734694, 57.96551724,
                46.95833333, 51.91666667, 51.81481481, 43.61363636, 38.27777778,
                34.74074074, 34.04166667, 35.875     , 43.29032258, 46.90625   ,
                49.39473684, 51.79310345, 52.97619048, 50.60714286, 47.44117647,
                35.25      , 43.        , 44.1       , 47.625     , 47.83333333,
                42.375     , 49.64      , 53.66666667, 60.65517241, 67.25806452,
                62.1875    , 38.40625   , 22.125     , 24.03571429, 26.15384615,
                42.33333333, 53.64      , 44.8       , 46.28      , 34.08571429,
                29.08333333, 35.08333333, 41.29166667, 52.25      , 53.89473684,
                65.52631579, 66.24242424, 68.5625    , 68.6969697 , 65.25714286,
                69.82758621, 68.81481481, 57.23076923, 66.        , 70.09677419,
                54.75      , 62.82857143, 72.23333333, 63.2       , 53.375     ,
                56.04166667, 57.67857143, 65.19354839, 55.96      , 50.16666667,
                58.03125   , 71.90625   , 58.92307692, 53.125     , 56.91666667,
                64.5862069 , 67.8125    , 61.8125    , 58.        , 66.64864865,
                70.79411765, 73.82758621, 72.84615385, 70.89655172, 71.4516129 ,
                71.625     , 70.63636364, 69.64      , 73.03333333, 76.34482759,
                62.69230769, 57.42857143, 61.83783784, 55.96153846, 58.45833333,
                72.12903226, 73.29032258, 74.23529412, 67.47222222, 57.625     ,
                66.84615385, 71.        , 79.22222222, 76.61290323, 76.85714286,
                70.23333333, 66.8       , 68.2       , 75.125     , 70.92      ,
                61.625     , 66.96      , 79.57142857, 80.19354839, 80.        ,
                78.51515152, 79.72413793, 79.64285714, 78.5       , 79.64516129,
                79.        , 78.32      , 65.04166667, 70.88      , 79.80645161,
                73.67857143, 52.21621622, 58.07142857, 62.83333333, 68.08333333,
                72.51851852, 77.2962963 , 81.76923077, 82.19230769, 80.3       ,
                78.6969697 , 70.48648649, 68.90909091, 68.58333333, 66.91666667,
                72.5       , 72.41666667, 75.51515152, 77.63157895, 80.45454545,
                77.37142857, 81.76470588, 82.18518519, 83.84615385, 86.44      ,
                79.33333333, 80.        , 83.62962963, 82.375     , 83.77419355,
                83.63636364, 80.2972973 , 81.70833333, 83.58333333, 80.3       ,
                82.13333333, 85.58333333, 83.75      , 83.68      , 82.53571429,
                82.65517241, 83.22580645, 82.45714286, 84.92857143, 85.31034483,
                86.1       , 86.17857143, 88.46153846, 88.06666667, 87.46153846,
                85.5862069 , 87.07407407, 75.32432432, 84.73076923, 83.5       ,
                83.51515152, 85.37931034, 86.20588235, 84.9375    , 84.08      ,
                84.70833333, 83.41176471, 84.88461538, 86.76923077, 85.33333333,
                84.79166667, 85.75      , 84.66666667, 86.79166667, 85.39285714,
                84.12121212, 84.375     , 85.12903226, 86.07692308, 86.79166667,
                88.41666667, 86.20833333, 85.66666667, 87.16666667, 84.2       ,
                84.13333333, 84.35714286, 85.5       , 85.65517241, 85.43333333,
                88.29166667, 88.875     , 88.45833333, 86.53125   , 86.32142857,
                85.66666667, 84.25925926, 89.125     , 91.29166667, 90.08333333,
                89.79166667, 89.54166667, 87.10714286, 87.5862069 , 88.72      ,
                86.53125   , 87.10714286, 87.80645161, 89.32142857, 85.90322581,
                88.23076923, 89.91666667, 88.44      , 86.19230769, 88.29166667,
                91.19230769, 89.5       , 86.88461538, 88.91666667, 89.08333333,
                89.04166667, 82.75      , 85.41666667, 91.83333333, 93.375     ,
                93.70833333, 88.20833333, 86.79166667, 85.25      , 88.58333333,
                87.54166667, 89.08333333, 79.46153846, 71.25      , 72.875     ,
                75.70833333, 76.        , 76.91666667, 81.875     , 84.54166667,
                87.8       , 85.625     , 82.45833333, 78.8125    , 76.46428571,
                77.21052632, 78.27586207, 75.91666667, 79.88      , 80.08333333,
                76.08333333, 77.41666667, 86.95833333, 83.70833333, 83.8125    ,
                81.33333333, 80.6       , 78.08333333, 70.32      , 68.375     ,
                70.08333333, 68.70833333, 74.        , 77.25925926, 80.90322581,
                77.375     , 65.71111111, 68.66666667, 72.125     , 74.5       ,
                72.625     , 67.45833333, 69.58333333, 68.53571429, 70.86666667,
                65.33333333, 59.91666667, 58.79166667, 68.33333333, 72.25      ,
                72.41666667, 72.08333333, 70.05128205, 70.58064516, 63.81081081,
                54.28      , 51.625     , 55.68      , 61.33333333, 64.58333333,
                68.21212121, 50.625     , 46.08333333, 59.84      , 74.67857143,
                74.8       , 67.78125   , 55.83333333, 51.66666667, 50.45833333,
                66.29166667, 71.66666667, 75.        , 69.06818182, 64.8       ,
                52.        , 57.42307692, 73.08571429, 78.16129032, 75.02631579,
```

In [41]:
```
ls
```

```
'sky_conditionFlag', 'visibilityFlag', 'wx_and_obs
NOAA_QCLCD_2011_hourly_13904.txt
Untitled.ipynb
Untitled1.ipynb
Untitled2.ipynb
Untitled3.ipynb
Untitled4.ipynb
Untitled5.ipynb
Untitled6.ipynb
austin_TX_weather_and_climate_analysis.ipynb
austin_airport_departure_data_2015_july.csv
auto-mpg.csv
clean_stock_data.csv
column_labels
life_expectancy_at_birth.csv
messy_stock_data.tsv
percent-bachelors-degrees-women-usa.csv
tips.csv
titanic.csv
weather_data_austin_2010.csv
world_ind_pop_data.csv
world_population.csv
```

In [42]:
```
df_climate_2010 = pd.read_csv('weather_data_austin_2010.csv')
```

In [43]:
```
df_climate_2010.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 8759 entries, 0 to 8758
Data columns (total 4 columns):
Temperature    8759 non-null float64
DewPoint       8759 non-null float64
Pressure       8759 non-null float64
Date           8759 non-null object
dtypes: float64(3), object(1)
memory usage: 273.8+ KB
```

In [44]:
```
df_climate_2010.head(9)
```

Out[44]:

	Temperature	DewPoint	Pressure	Date
0	46.2	37.5	1.0	20100101 00:00
1	44.6	37.1	1.0	20100101 01:00
2	44.1	36.9	1.0	20100101 02:00
3	43.8	36.9	1.0	20100101 03:00
4	43.5	36.8	1.0	20100101 04:00
5	43.0	36.5	1.0	20100101 05:00
6	43.1	36.3	1.0	20100101 06:00
7	42.3	35.9	1.0	20100101 07:00
8	42.5	36.2	1.0	20100101 08:00

In [45]:
```
df_climate_2010.Date = pd.to_datetime(df_climate_2010.Date)
```

In [46]:
```
df_climate_2010.head(9)
```

Out[46]:

	Temperature	DewPoint	Pressure	Date
0	46.2	37.5	1.0	2010-01-01 00:00:00
1	44.6	37.1	1.0	2010-01-01 01:00:00
2	44.1	36.9	1.0	2010-01-01 02:00:00
3	43.8	36.9	1.0	2010-01-01 03:00:00
4	43.5	36.8	1.0	2010-01-01 04:00:00
5	43.0	36.5	1.0	2010-01-01 05:00:00
6	43.1	36.3	1.0	2010-01-01 06:00:00
7	42.3	35.9	1.0	2010-01-01 07:00:00
8	42.5	36.2	1.0	2010-01-01 08:00:00

In [47]:
```
df_climate_2010.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 8759 entries, 0 to 8758
Data columns (total 4 columns):
Temperature    8759 non-null float64
DewPoint       8759 non-null float64
Pressure       8759 non-null float64
Date           8759 non-null datetime64[ns]
dtypes: datetime64[ns](1), float64(3)
memory usage: 273.8 KB
```

In [48]:
```
df_climate_2010 = df_climate_2010.set_index('Date')
```

In [49]:
```
df_climate_2010.info()
```

```
<class 'pandas.core.frame.DataFrame'>
DatetimeIndex: 8759 entries, 2010-01-01 00:00:00 to 2010-12-31 23:00:00
Data columns (total 3 columns):
Temperature    8759 non-null float64
DewPoint       8759 non-null float64
Pressure       8759 non-null float64
dtypes: float64(3)
memory usage: 273.7 KB
```

In [50]:
```
df_climate_2010.head(9)
```

Out[50]:

	Temperature	DewPoint	Pressure
Date			
2010-01-01 00:00:00	46.2	37.5	1.0
2010-01-01 01:00:00	44.6	37.1	1.0
2010-01-01 02:00:00	44.1	36.9	1.0
2010-01-01 03:00:00	43.8	36.9	1.0
2010-01-01 04:00:00	43.5	36.8	1.0
2010-01-01 05:00:00	43.0	36.5	1.0
2010-01-01 06:00:00	43.1	36.3	1.0
2010-01-01 07:00:00	42.3	35.9	1.0
2010-01-01 08:00:00	42.5	36.2	1.0

In [51]:
```
#down sampling by day and aggregate by mean
daily_climate_2010 = df_climate_2010.resample('D').mean()
```

In [52]:
```
daily_climate_2010.head(9)
```

Out[52]:

	Temperature	DewPoint	Pressure
Date			
2010-01-01	49.337500	37.716667	1.0
2010-01-02	49.795833	38.370833	1.0
2010-01-03	49.900000	38.279167	1.0
2010-01-04	49.729167	38.008333	1.0
2010-01-05	49.841667	38.087500	1.0
2010-01-06	49.679167	37.787500	1.0
2010-01-07	49.491667	37.487500	1.0
2010-01-08	49.366667	37.408333	1.0
2010-01-09	49.354167	37.583333	1.0

In [53]:
```
daily_temp_2010 = daily_climate_2010.reset_index()['Temperature']
```

In [54]:
```
daily_temp_2010
```

Out[54]:
```
0      49.337500
1      49.795833
2      49.900000
3      49.729167
4      49.841667
5      49.679167
6      49.491667
7      49.366667
8      49.354167
9      49.354167
10     49.245833
11     49.279167
12     49.275000
13     49.158333
14     49.154167
15     49.245833
16     49.200000
17     49.370833
18     49.645833
19     49.745833
20     50.166667
21     50.608333
22     50.820833
23     50.850000
24     50.970833
25     51.116667
26     51.145833
27     51.220833
28     51.233333
29     51.133333
         ...
335    54.858333
336    54.545833
337    54.195833
338    53.837500
339    53.587500
340    53.283333
341    53.037500
342    52.770833
343    52.408333
344    52.083333
345    51.954167
346    51.920833
347    51.775000
348    51.370833
349    50.883333
350    50.437500
351    49.891667
352    49.479167
353    49.433333
354    49.358333
355    49.216667
356    49.179167
357    49.300000
358    49.312500
359    49.216667
360    49.204167
361    48.979167
362    48.804167
363    49.008333
364    49.195833
Name: Temperature, Length: 365, dtype: float64
```

In [55]:
```
daily_temp_2010
```

Out[55]:
```
0      49.337500
1      49.795833
2      49.900000
3      49.729167
4      49.841667
5      49.679167
6      49.491667
7      49.366667
8      49.354167
9      49.354167
10     49.245833
11     49.279167
12     49.275000
13     49.158333
14     49.154167
15     49.245833
16     49.200000
17     49.370833
18     49.645833
19     49.745833
20     50.166667
21     50.608333
22     50.820833
23     50.850000
24     50.970833
25     51.116667
26     51.145833
27     51.220833
28     51.233333
29     51.133333
         ...
335    54.858333
336    54.545833
337    54.195833
338    53.837500
339    53.587500
340    53.283333
341    53.037500
342    52.770833
343    52.408333
344    52.083333
345    51.954167
346    51.920833
347    51.775000
348    51.370833
349    50.883333
350    50.437500
351    49.891667
352    49.479167
353    49.433333
354    49.358333
355    49.216667
356    49.179167
357    49.300000
358    49.312500
359    49.216667
360    49.204167
361    48.979167
362    48.804167
363    49.008333
364    49.195833
Name: Temperature, Length: 365, dtype: float64
```

In [56]:
```
#the difference between the two arrays and print the mean difference
difference = daily_temp_2011 - daily_temp_2010
```

In [57]:
```
difference
```

Out[57]:
```
0       0.829167
1     -10.379167
2      -3.053846
3       3.638180
4       8.123851
5      -2.720833
6       2.425000
7       2.448148
8      -5.740530
9     -11.076389
10    -14.505093
11    -15.237500
12    -13.400000
13     -5.868011
14     -2.247917
15      0.148904
16      2.593103
17      3.605357
18      0.961310
19     -2.304657
20    -14.916667
21     -7.608333
22     -6.720833
23     -3.225000
24     -3.137500
25     -8.741667
26     -1.505833
27      2.445833
28      9.421839
29     16.124731
         ...
335     4.416176
336     9.886599
337    -3.732870
338   -11.743750
339   -17.846759
340   -18.366667
341   -14.037500
342    -4.508929
343    -2.808333
344    -3.125000
345     2.453241
346    11.812500
347    19.030556
348     6.843452
349     0.009524
350     1.687500
351     5.322619
352    15.145833
353     3.325287
354    -2.512179
355     1.400980
356    -5.756090
357    -6.041935
358    -5.077206
359    -2.023118
360    -4.370833
361    -3.229167
362     1.515833
363     3.533333
364     5.262500
Name: Temperature, Length: 365, dtype: float64
```

In [58]:
```
#sunny or cloudy ? On average, how much hotter is it when the sun is shining?
#filtering out sunny or overcast days then compute the difference of the mean dail
y maximum temperature
```

In [59]:
```
sunny = df_clean.loc[df_clean.sky_condition=='CLR']
```

In [60]:
```
overcast = df_clean.loc[df_clean.sky_condition=='OVC']
```

In [61]:
```
sunny.head(9)
```

Out[61]:

	Wban	date	Time	StationType	sky_condition	visibility	dry_bulb_faren	dry_bulb_ce
2011-01-01 13:53:00	13904	20110101	1353	12	CLR	10.00	59.0	15.0
2011-01-01 14:53:00	13904	20110101	1453	12	CLR	10.00	59.0	15.0
2011-01-01 15:53:00	13904	20110101	1553	12	CLR	10.00	57.0	13.9
2011-01-01 16:53:00	13904	20110101	1653	12	CLR	10.00	55.0	12.8
2011-01-01 17:53:00	13904	20110101	1753	12	CLR	10.00	50.0	10.0
2011-01-01 18:53:00	13904	20110101	1853	12	CLR	10.00	47.0	8.3
2011-01-01 19:53:00	13904	20110101	1953	12	CLR	10.00	45.0	7.2
2011-01-01 20:53:00	13904	20110101	2053	12	CLR	10.00	41.0	5.0
2011-01-01 21:53:00	13904	20110101	2153	12	CLR	10.00	40.0	4.4

In [62]:
```
overcast.head(9)
```

Out[62]:

	Wban	date	Time	StationType	sky_condition	visibility	dry_bulb_faren	dry_bulb_cel	wet_bulb_fare

In [63]:
```
#here in overcast has not data :) because there is no OVC in sky_condition
#so need to filter concisely
```

In [64]:
```
df_clean.head(9)
```

Out[64]:

	Wban	date	Time	StationType	sky_condition	visibility	dry_bulb_faren	dry_bulb_ce
2011-01-01 00:53:00	13904	20110101	0053	12	OVC045	10.00	51.0	10.6
2011-01-01 01:53:00	13904	20110101	0153	12	OVC049	10.00	51.0	10.6
2011-01-01 02:53:00	13904	20110101	0253	12	OVC060	10.00	51.0	10.6
2011-01-01 03:53:00	13904	20110101	0353	12	OVC065	10.00	50.0	10.0
2011-01-01 04:53:00	13904	20110101	0453	12	BKN070	10.00	50.0	10.0
2011-01-01 05:53:00	13904	20110101	0553	12	BKN065	10.00	49.0	9.4
2011-01-01 06:53:00	13904	20110101	0653	12	BKN065	10.00	48.0	8.9
2011-01-01 07:53:00	13904	20110101	0753	12	SCT060	10.00	48.0	8.9
2011-01-01 08:53:00	13904	20110101	0853	12	FEW060	10.00	51.0	10.6

In [65]:
```
overcast = df_clean.loc[df_clean.sky_condition.str.contains('OVC')]
```

In [66]:
```
overcast.head(9)
```

Out[66]:

	Wban	date	Time	StationType	sky_condition	visibility	dry_bulb_faren	dry_bulb_ce
2011-01-01 00:53:00	13904	20110101	0053	12	OVC045	10.00	51.0	10.6
2011-01-01 01:53:00	13904	20110101	0153	12	OVC049	10.00	51.0	10.6
2011-01-01 02:53:00	13904	20110101	0253	12	OVC060	10.00	51.0	10.6
2011-01-01 03:53:00	13904	20110101	0353	12	OVC065	10.00	50.0	10.0
2011-01-03 07:53:00	13904	20110103	0753	12	OVC055	10.00	37.0	2.8
2011-01-03 08:53:00	13904	20110103	0853	12	OVC055	10.00	41.0	5.0
2011-01-03 09:53:00	13904	20110103	0953	12	OVC055	10.00	44.0	6.7
2011-01-03 10:53:00	13904	20110103	1053	12	OVC055	10.00	49.0	9.4
2011-01-03 11:53:00	13904	20110103	1153	12	OVC055	10.00	53.0	11.7

In [67]:
```
sunny_daily_max = sunny.resample('D').max()
```

In [68]:
```
overcast_daily_max = overcast.resample('D').max()
```

In [69]:
```
sunny_daily_max.mean()
```

Out[69]:
```
Wban              13904.000000
StationType          12.000000
dry_bulb_faren       75.560714
dtype: float64
```

In [70]:
```
overcast_daily_max.mean()
```

Out[70]:
```
Wban              13904.00000
StationType          12.00000
dry_bulb_faren       69.05641
dtype: float64
```

In [71]:
```
'''Weekly average temperature and visibility
Is there a correlation between temperature and visibility? Let's find out.
let's plot the weekly average temperature and visibility as subplots.
'''
```

Out[71]:
```
"Weekly average temperature and visibility\nIs there a correlation between tempe
rature and visibility? Let's find out.\nlet's plot the weekly average temperatur
e and visibility as subplots. \n"
```

In [72]:
```
weekly_mean = df_clean[['visibility','dry_bulb_faren']].resample('W').mean()
```

In [73]:
```
weekly_mean.head(9)
```

Out[73]:

	dry_bulb_faren
2011-01-02	44.791667
2011-01-09	50.246637
2011-01-16	41.103774
2011-01-23	47.194313
2011-01-30	53.486188
2011-02-06	38.235294
2011-02-13	40.082873
2011-02-20	65.100840
2011-02-27	64.976636

In [74]:
```
## ????????? visibility column has disappeared ??? let see what the problem could be??
```

In [75]:
```
df_clean.info()
```

```
<class 'pandas.core.frame.DataFrame'>
DatetimeIndex: 10337 entries, 2011-01-01 00:53:00 to 2011-12-31 23:53:00
Data columns (total 17 columns):
Wban                  10337 non-null int64
date                  10337 non-null object
Time                  10337 non-null object
StationType           10337 non-null int64
sky_condition         10337 non-null object
visibility            10325 non-null object
dry_bulb_faren        10326 non-null float64
dry_bulb_cel          10337 non-null object
wet_bulb_faren        10337 non-null object
wet_bulb_cel          10337 non-null object
dew_point_faren       10337 non-null object
dew_point_cel         10337 non-null object
relative_humidity     10337 non-null object
wind_speed            10337 non-null object
wind_direction        10337 non-null object
station_pressure      10337 non-null object
sea_level_pressure    10337 non-null object
dtypes: float64(1), int64(2), object(14)
memory usage: 1.7+ MB
```

In [76]:
```
df_clean.visibility.unique()
```

Out[76]:
```
array(['10.00', ' 9.00', ' 5.00', ' 4.00', ' 2.50', ' 1.50', ' 0.75',
       ' 3.00', ' 2.00', ' 8.00', ' 6.00', ' 7.00', ' 0.50', ' 1.25',
       ' 1.00', ' 1.75', ' 0.25', nan, ' 0.12', ' 0.00', 'M'],
      dtype=object)
```

In [77]:
```
# visibility is object, we need to convert it into float64,
#int doesn't work for the  values as seen above
```

In [78]:
```
df_clean.visibility = pd.to_numeric(df_clean.visibility,errors='coerce')
```

In [79]:
```
df_clean.info()
```

```
<class 'pandas.core.frame.DataFrame'>
DatetimeIndex: 10337 entries, 2011-01-01 00:53:00 to 2011-12-31 23:53:00
Data columns (total 17 columns):
Wban                  10337 non-null int64
date                  10337 non-null object
Time                  10337 non-null object
StationType           10337 non-null int64
sky_condition         10337 non-null object
visibility            10324 non-null float64
dry_bulb_faren        10326 non-null float64
dry_bulb_cel          10337 non-null object
wet_bulb_faren        10337 non-null object
wet_bulb_cel          10337 non-null object
dew_point_faren       10337 non-null object
dew_point_cel         10337 non-null object
relative_humidity     10337 non-null object
wind_speed            10337 non-null object
wind_direction        10337 non-null object
station_pressure      10337 non-null object
sea_level_pressure    10337 non-null object
dtypes: float64(2), int64(2), object(13)
memory usage: 1.7+ MB
```

In [80]:
```
#perfect!!! now our visibility columns is float64 let us try again
```

In [81]:
```
weekly_mean = df_clean[['visibility','dry_bulb_faren']].resample('W').mean()
```

In [82]:
```
weekly_mean.head(9)
```

Out[82]:

	visibility	dry_bulb_faren
2011-01-02	10.000000	44.791667
2011-01-09	8.275785	50.246637
2011-01-16	6.451651	41.103774
2011-01-23	8.370853	47.194313
2011-01-30	9.966851	53.486188
2011-02-06	9.242647	38.235294
2011-02-13	9.281768	40.082873
2011-02-20	8.711134	65.100840
2011-02-27	8.154206	64.976636

In [83]:
```
#let's see if there is a correlation between visibility and dry_bulb_faren
```

In [84]:
```
weekly_mean.corr()
```

Out[84]:

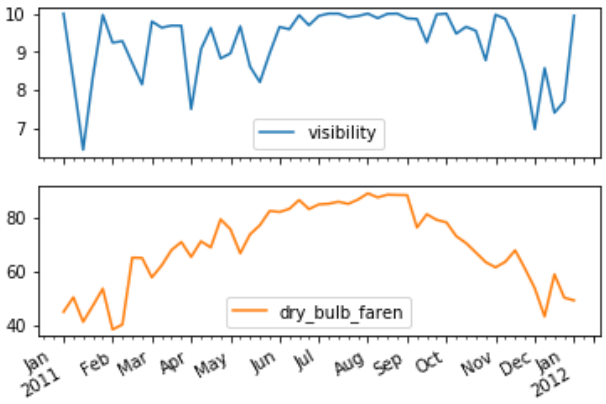
	visibility	dry_bulb_faren
visibility	1.000000	0.490328
dry_bulb_faren	0.490328	1.000000

In [85]:
```
#let's plot
```

In [86]:
```
import matplotlib.pyplot as plt
```

In [87]:
```
weekly_mean.plot(subplots=True)
```

Out[87]:
```
array([<matplotlib.axes._subplots.AxesSubplot object at 0x1182a8198>,
       <matplotlib.axes._subplots.AxesSubplot object at 0x11858f3c8>],
      dtype=object)
```



In [88]:
```
'''above I have analyzed the sky_condition column to explore
the difference in temperature on sunny days compared to overcast days
Now, the task is to resample sky_condition to hourly that I can extract the number
of sunny hours in a day and the number of total hours. then I can divide
the number of sunny hours by the number of total hours to generate boxplot
'''
```

Out[88]:
```
'above I have analyzed the sky_condition column to explore \nthe difference in t
emperature on sunny days compared to overcast days\nNow, the task is to resample
sky_condition to hourly that I can extract the number\nof sunny hours in a day a
nd the number of total hours. then I can divide\nthe number of sunny hours by th
e number of total hours to generate boxplot\n'
```

In [89]:
```
#creating a boolean series for sunny days
sunny_series = df_clean.sky_condition=='CLR'
```

In [90]:
```
sunny_series
```

Out[90]:
```
2011-01-01 00:53:00    False
2011-01-01 01:53:00    False
2011-01-01 02:53:00    False
2011-01-01 03:53:00    False
2011-01-01 04:53:00    False
2011-01-01 05:53:00    False
2011-01-01 06:53:00    False
2011-01-01 07:53:00    False
2011-01-01 08:53:00    False
2011-01-01 09:53:00    False
2011-01-01 10:53:00    False
2011-01-01 11:53:00    False
2011-01-01 12:53:00    False
2011-01-01 13:53:00     True
2011-01-01 14:53:00     True
2011-01-01 15:53:00     True
2011-01-01 16:53:00     True
2011-01-01 17:53:00     True
2011-01-01 18:53:00     True
2011-01-01 19:53:00     True
2011-01-01 20:53:00     True
2011-01-01 21:53:00     True
2011-01-01 22:53:00     True
2011-01-01 23:53:00     True
2011-01-02 00:53:00     True
2011-01-02 01:53:00     True
2011-01-02 02:53:00     True
2011-01-02 03:53:00     True
2011-01-02 04:53:00     True
2011-01-02 05:53:00     True
                       ...
2011-12-30 18:53:00     True
2011-12-30 19:53:00     True
2011-12-30 20:53:00     True
2011-12-30 21:53:00     True
2011-12-30 22:53:00     True
2011-12-30 23:53:00     True
2011-12-31 00:53:00     True
2011-12-31 01:53:00     True
2011-12-31 02:53:00     True
2011-12-31 03:53:00     True
2011-12-31 04:53:00     True
2011-12-31 05:53:00     True
2011-12-31 06:53:00    False
2011-12-31 07:53:00    False
2011-12-31 08:53:00    False
2011-12-31 09:53:00    False
2011-12-31 10:53:00    False
2011-12-31 11:53:00    False
2011-12-31 12:53:00    False
2011-12-31 13:53:00    False
2011-12-31 14:53:00    False
2011-12-31 15:53:00    False
2011-12-31 16:53:00    False
2011-12-31 17:53:00    False
2011-12-31 18:53:00     True
2011-12-31 19:53:00     True
2011-12-31 20:53:00     True
2011-12-31 21:53:00     True
2011-12-31 22:53:00     True
2011-12-31 23:53:00     True
Name: sky_condition, Length: 10337, dtype: bool
```

In [91]:
```
#we know that True=1 and False=0 so if we sum by daily we can get the total
#sunny hours per day
```

In [92]:
```
sunny_hours_series = sunny_series.resample('D').sum()
```

In [93]:
```
sunny_hours_series.min()
```

Out[93]:
```
0.0
```

In [94]:
```
#18 hours as min is very suspicious, even in the longest day in equavator countrie
s
#there cannot be 18 hours of min day time :)
#let us see the max
sunny_hours_series.max()
```

Out[94]:
```
24.0
```

In [95]:
```
#there is no way 56 hours day time in 24 hours :D for sure something is wrong
```

In [96]:
```
#let us investigate
sunny_hours_series
```

```
Out[96]: 2011-01-01    11.0
         2011-01-02     7.0
         2011-01-03     3.0
         2011-01-04     0.0
         2011-01-05     1.0
         2011-01-06     6.0
         2011-01-07     4.0
         2011-01-08     6.0
         2011-01-09     0.0
         2011-01-10     0.0
         2011-01-11     3.0
         2011-01-12     0.0
         2011-01-13     0.0
         2011-01-14     0.0
         2011-01-15     0.0
         2011-01-16     0.0
         2011-01-17     2.0
         2011-01-18     7.0
         2011-01-19     0.0
         2011-01-20     6.0
         2011-01-21    10.0
         2011-01-22    21.0
         2011-01-23     6.0
         2011-01-24     1.0
         2011-01-25     8.0
         2011-01-26     8.0
         2011-01-27     3.0
         2011-01-28     1.0
         2011-01-29     0.0
         2011-01-30     3.0
                        ...
         2011-12-02     0.0
         2011-12-03     0.0
         2011-12-04     0.0
         2011-12-05     0.0
         2011-12-06     5.0
         2011-12-07    10.0
         2011-12-08     1.0
         2011-12-09     0.0
         2011-12-10     0.0
         2011-12-11     0.0
         2011-12-12     0.0
         2011-12-13     0.0
         2011-12-14     0.0
         2011-12-15     0.0
         2011-12-16     0.0
         2011-12-17     0.0
         2011-12-18     0.0
         2011-12-19     0.0
         2011-12-20     0.0
         2011-12-21     0.0
         2011-12-22    12.0
         2011-12-23     2.0
         2011-12-24     0.0
         2011-12-25     0.0
         2011-12-26     8.0
         2011-12-27    24.0
         2011-12-28     7.0
         2011-12-29    14.0
         2011-12-30    18.0
         2011-12-31    12.0
         Freq: D, Name: sky_condition, Length: 365, dtype: float64
```

In [97]:
```
# I have seen the total hours is 56 on 2011-12-15 so let's investigate
df_clean2 = df_clean[['sky_condition']]
```

In [98]:
```
df_clean2.sky_condition.unique()
```

Out[98]:
```
array(['OVC045', 'OVC049', 'OVC060', ..., 'FEW019 BKN032 BKN095',
       'FEW019 SCT032 BKN100', 'SCT021 BKN028'], dtype=object)
```

In [99]:
```
#as far as our sky_condition is all clear :D
# we can either say all year austin TX weather is sunny day time (which is unreali
stic)
# or we can not conclude anything for sunny/cloudy based on existing data
#we need some another data.
```

In [100]:
```
'''heat or humidity'''
'''Dew point is a measure of relative humidity based on pressure and temperature.
A dew point above 65 is considered uncomfortable while a temperature above 90 is a
lso considered uncomfortable.'''
```

Out[100]:
```
'Dew point is a measure of relative humidity based on pressure and temperature.\
nA dew point above 65 is considered uncomfortable while a temperature above 90 i
s also considered uncomfortable.'
```

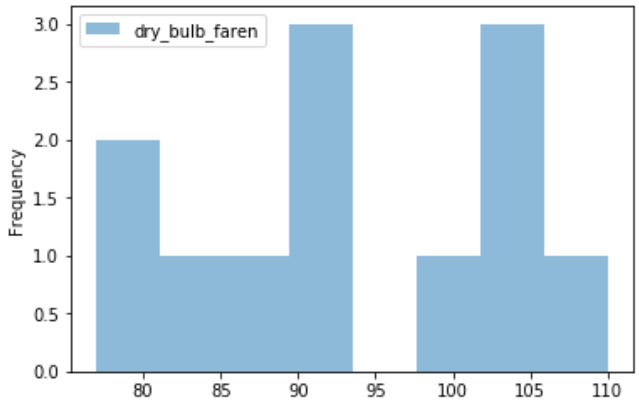
In [101]:
```
'''let's explore the maximum temperature and dew point of each month.'''
```

Out[101]:
```
"let's explore the maximum temperature and dew point of each month."
```

In [102]:
```
monthly_max = df_clean[['dew_point_faren','dry_bulb_faren']].resample('M').max()
```

In [103]:
```
monthly_max.plot(kind='hist',bins=8,alpha=0.5)
```

Out[103]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x116c55518>
```



In [104]:
```
#again had only one data probably the reason is data type of the caolumn let s see
```

In [105]:
```
df_clean.info()
```

```
<class 'pandas.core.frame.DataFrame'>
DatetimeIndex: 10337 entries, 2011-01-01 00:53:00 to 2011-12-31 23:53:00
Data columns (total 17 columns):
Wban                  10337 non-null int64
date                  10337 non-null object
Time                  10337 non-null object
StationType           10337 non-null int64
sky_condition         10337 non-null object
visibility            10324 non-null float64
dry_bulb_faren        10326 non-null float64
dry_bulb_cel          10337 non-null object
wet_bulb_faren        10337 non-null object
wet_bulb_cel          10337 non-null object
dew_point_faren       10337 non-null object
dew_point_cel         10337 non-null object
relative_humidity     10337 non-null object
wind_speed            10337 non-null object
wind_direction        10337 non-null object
station_pressure      10337 non-null object
sea_level_pressure    10337 non-null object
dtypes: float64(2), int64(2), object(13)
memory usage: 1.7+ MB
```

In [106]:
```
df_clean.dew_point_faren = pd.to_numeric(df_clean.dew_point_faren)
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
pandas/_libs/src/inference.pyx in pandas._libs.lib.maybe_convert_numeric()

ValueError: Unable to parse string "M"

During handling of the above exception, another exception occurred:

ValueError                                Traceback (most recent call last)
<ipython-input-106-c7ac6143fd62> in <module>
----> 1 df_clean.dew_point_faren = pd.to_numeric(df_clean.dew_point_faren)

~/anaconda3/envs/tfdeeplearning/lib/python3.6/site-packages/pandas/core/tools/nu
meric.py in to_numeric(arg, errors, downcast)
    131             coerce_numeric = False if errors in ('ignore', 'raise') else
True
    132             values = lib.maybe_convert_numeric(values, set(),
--> 133                                                coerce_numeric=coerce_num
eric)
    134
    135     except Exception:

pandas/_libs/src/inference.pyx in pandas._libs.lib.maybe_convert_numeric()

ValueError: Unable to parse string "M" at position 3552
```

In [107]:
```
#just forgot to errors='coerse' so that anything besides number can be NaN
df_clean.dew_point_faren = pd.to_numeric(df_clean.dew_point_faren,errors='coerse')
```

In [108]:
```
df_clean.info()
```

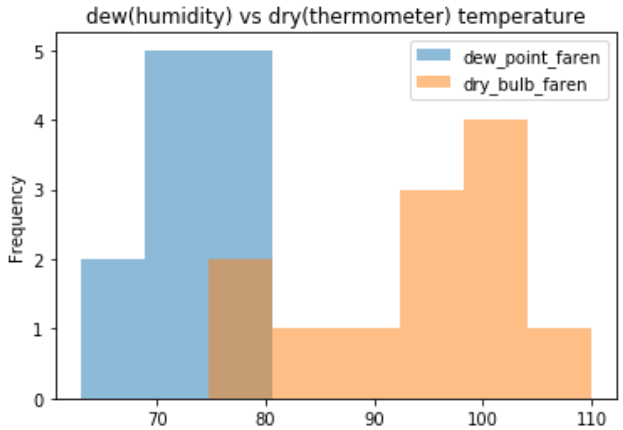
```
<class 'pandas.core.frame.DataFrame'>
DatetimeIndex: 10337 entries, 2011-01-01 00:53:00 to 2011-12-31 23:53:00
Data columns (total 17 columns):
Wban                  10337 non-null int64
date                  10337 non-null object
Time                  10337 non-null object
StationType           10337 non-null int64
sky_condition         10337 non-null object
visibility            10324 non-null float64
dry_bulb_faren        10326 non-null float64
dry_bulb_cel          10337 non-null object
wet_bulb_faren        10337 non-null object
wet_bulb_cel          10337 non-null object
dew_point_faren       10323 non-null float64
dew_point_cel         10337 non-null object
relative_humidity     10337 non-null object
wind_speed            10337 non-null object
wind_direction        10337 non-null object
station_pressure      10337 non-null object
sea_level_pressure    10337 non-null object
dtypes: float64(3), int64(2), object(12)
memory usage: 1.7+ MB
```

In [109]:
```
monthly_max = df_clean[['dew_point_faren','dry_bulb_faren']].resample('M').max()
```

In [110]:
```
'''The dry-bulb temperature (DBT) is the temperature of air
measured by a thermometer freely exposed to the air,
but shielded from radiation and moisture. DBT is the temperature
that is usually thought of as air temperature, and it is the true thermodynamic te
mperature.'''

'''The dew point is the temperature at which air is saturated with water vapor,
which is the gaseous state of water. When air has reached the dew-point temperatur
e
at a particular pressure, the water vapor in the air is in equilibrium with liquid
water,
meaning water vapor is condensing at the same rate at which liquid water is evapor
ating.'''
monthly_max.plot(title='dew(humidity) vs dry(thermometer) temperature',kind='hist'
,bins=8,alpha=0.5)
```

Out[110]: <matplotlib.axes._subplots.AxesSubplot at 0x117036a90>



In [111]:
```
monthly_max.plot(kind='hist',bins=8,alpha=0.5,subplots=True)
```

Out[111]:
```
array([<matplotlib.axes._subplots.AxesSubplot object at 0x118a37668>,
       <matplotlib.axes._subplots.AxesSubplot object at 0x118b03048>],
      dtype=object)
```