# Lab 3 Data Basics

Installations

- Install Python 3
- Install Jupyter Notebooks
- Install modules you need, such as pandas and numpy

In [6]:
```
# import modules you need
import pandas as pd
import numpy as np
import os
```

# Part A: Python Review

In [1]:
```
# Create a string called string1 that contains your first and last name
string1 = 'JN'

# Print the string
print(string1)
```

```
JN
```

In [8]:
```
# Use list comprehension to divide each number in "mylist" by 2
mylist = [8,1,2,4,1,29]
answer = [x/2 for x in mylist]
print(answer)
```

```
[4.0, 0.5, 1.0, 2.0, 0.5, 14.5]
```

In [9]:
```
# Create a dictionary to map names of the days of the week (e.g. Sunday, Monday, Tuesd
# to numbers, where Sunday = 0, Monday = 1, Tuesday = 2, etc.
d = {'Sunday': 0, 'Monday': 1, 'Tuesday': 2, 'Wednesday': 3, 'Thursday': 4, 'Friday':
# Print the dictionary
for key,value in d.items():
    print(key, ':', value)
```

```
Sunday : 0
Monday : 1
Tuesday : 2
Wednesday : 3
Thursday : 4
Friday : 5
Saturday : 6
```

In [10]:
```
# Create a lambda function to multiply a number by 100. Name this function "mult_by_10
mult_by_100 = lambda x: x*100
```

In [11]:
```
# Use your lambda function, mult_by_100, to multiply the number 50 by 100
mult_by_100(50)
```

Out[11]:
```
5000
```

# Part B: Introduction Data Frames

## Read in weather data

In [12]:
```
# read in the weather data from csv (WeatherData.csv)
f = pd.read_csv('WeatherData.csv')
```

In [13]:
```
# print the first few rows of the weather data
f.head()
```

Out[13]:

| | STATION | NAME | DATE | AWND | AWND_ATTRIBUTES | PRCP | PRCP_ATTRIBUTES | SNOW |
|---|---|---|---|---|---|---|---|---|
| **0** | USW00013743 | WASHINGTON REAGAN NATIONAL AIRPORT, VA US | 2017-01-01 | 4.25 | ,,W | 0.00 | ,,W,2400 | 0.( |
| **1** | USW00013743 | WASHINGTON REAGAN NATIONAL AIRPORT, VA US | 2017-01-02 | 7.83 | ,,W | 0.35 | ,,W,2400 | 0.( |
| **2** | USW00013743 | WASHINGTON REAGAN NATIONAL AIRPORT, VA US | 2017-01-03 | 5.82 | ,,W | 0.86 | ,,W,2400 | 0.( |
| **3** | USW00013743 | WASHINGTON REAGAN NATIONAL AIRPORT, VA US | 2017-01-04 | 9.84 | ,,W | 0.00 | T,,W,2400 | 0.( |
| **4** | USW00013743 | WASHINGTON REAGAN NATIONAL AIRPORT, VA US | 2017-01-05 | 6.49 | ,,W | 0.00 | T,,W,2400 | 0.( |

5 rows × 41 columns

In [14]:
```
# print column and row dimensions for the weather data
f.shape
```

Out[14]:
```
(90, 41)
```

In [15]:
```
# practice slicing weather data
# create a new data frame that has only the date (DATE) and the amount of precipitatic
# make sure you save this slice to a new object (e.g. ndf)
ndf = f[['DATE', 'PRCP']]
ndf
```

Out[15]:

| | DATE | PRCP |
|---|---|---|
| **0** | 2017-01-01 | 0.00 |
| **1** | 2017-01-02 | 0.35 |
| **2** | 2017-01-03 | 0.86 |
| **3** | 2017-01-04 | 0.00 |
| **4** | 2017-01-05 | 0.00 |
| **...** | ... | ... |
| **85** | 2017-03-27 | 0.01 |
| **86** | 2017-03-28 | 0.02 |
| **87** | 2017-03-29 | 0.00 |
| **88** | 2017-03-30 | 0.00 |
| **89** | 2017-03-31 | 0.96 |

90 rows × 2 columns

In [16]:
```
# export the slice of your weather data frame with only date and precipitation (from t
ndf.to_csv('WeatherCols.csv')
```

# Read in trip data

In [17]:
```
# read in trip data (TripData.csv)
t = pd.read_csv('TripData.csv')
```

In [18]:
```
# print the first few rows of the trip data
t.head()
```

Out[18]:

| | Duration | Start date | End date | Start station number | Start station | End station number | End station | Bike number | Member Type |
|---|---|---|---|---|---|---|---|---|---|
| **0** | 1048876 | 3/31/2017 23:59 | 4/1/2017 0:17 | 31213 | 17th & K St NW | 31606 | Potomac & Pennsylvania Ave SE | W20784 | Registered |
| **1** | 223449 | 3/31/2017 23:59 | 4/1/2017 0:03 | 31104 | Adams Mill & Columbia Rd NW | 31103 | 16th & Harvard St NW | W20825 | Registered |
| **2** | 423494 | 3/31/2017 23:58 | 4/1/2017 0:05 | 31627 | M St & Delaware Ave NE | 31614 | 11th & H St NE | W20773 | Registered |
| **3** | 687015 | 3/31/2017 23:57 | 4/1/2017 0:08 | 31404 | 9th & Upshur St NW | 31281 | 8th & O St NW | W01307 | Registered |
| **4** | 257919 | 3/31/2017 23:57 | 4/1/2017 0:02 | 31602 | Park Rd & Holmead Pl NW | 31400 | Georgia & New Hampshire Ave NW | W21760 | Registered |

In [19]:
```
# print the dimensions of the trip data
t.shape
```

Out[19]:
```
(646508, 9)
```

In [20]:
```
# practice slicing the trip data
# create a new data frame with only the start date (Start date) and the duration (Dura
sl = t[['Start date', 'Duration']]
sl
```

Out[20]:

| | Start date | Duration |
|---|---|---|
| **0** | 3/31/2017 23:59 | 1048876 |
| **1** | 3/31/2017 23:59 | 223449 |
| **2** | 3/31/2017 23:58 | 423494 |
| **3** | 3/31/2017 23:57 | 687015 |
| **4** | 3/31/2017 23:57 | 257919 |
| **...** | ... | ... |
| **646503** | 1/1/2017 0:07 | 1356956 |
| **646504** | 1/1/2017 0:07 | 1327901 |
| **646505** | 1/1/2017 0:07 | 1636768 |
| **646506** | 1/1/2017 0:06 | 1676854 |
| **646507** | 1/1/2017 0:00 | 221834 |

646508 rows × 2 columns

In [21]:
```
# write the new trip data frame with only the start date and duration to a csv file cc
sl.to_csv('TripsCol.csv')
```

## Create day of the week data frame

In [22]:
```
# you can create a new data frame using dictionaries and lists
# create a dictionary with two keys "date" and "dayofweek"
# the date value should be a list of 7 dates from 2017-01-01 to 2017-01-07"
# the dayofweek value should a list of 7 days of the week starting with Sunday in orde
# the dictionary should be of the format:
# mydict = {'colname1':['list','of','values','in','order'],'colname2':['list','of','va
mydict = {'date':['2017-01-01', '2017-01-02', '2017-01-03', '2017-01-04', '2017-01-05'
```

In [23]:
```
# turn this dictionary into a data frame
datedict = pd.DataFrame(mydict)
```

In [24]:
```
# print the first few rows of the data frame
datedict.head()
```

Out[24]:

| | date | dayofweek |
|---|---|---|
| **0** | 2017-01-01 | Sunday |
| **1** | 2017-01-02 | Monday |
| **2** | 2017-01-03 | Tuesday |
| **3** | 2017-01-04 | Wednesday |
| **4** | 2017-01-05 | Thursday |

In [25]:
```
# print the dimensions of the data frame
datedict.shape
```

Out[25]:
```
(7, 2)
```

In [26]:
```
# write this data frame to csv file
datedict.to_csv('DateDictionary.csv')
```

# Part C: Summarizing Data

## Summarize weather data (from Part B)

In [27]:
```
# get the frequency distribution for the name column (NAME) using the value_counts met
w = pd.read_csv('WeatherData.csv', encoding = 'unicode_escape')
w["NAME"].value_counts()
```

Out[27]:
```
WASHINGTON REAGAN NATIONAL AIRPORT, VA US    90
Name: NAME, dtype: int64
```

In [28]:
```
# get descriptive statistics for all numeric columns in the weather data set
w.describe()
```

Out[28]:

| | AWND | PRCP | SNOW | SNWD | TAVG | TMAX | TMIN | WDF2 | |
|---|---|---|---|---|---|---|---|---|---|
| **count** | 90.000000 | 90.000000 | 90.000000 | 90.000000 | 90.000000 | 90.000000 | 90.000000 | 90.000000 | 90.( |
| **mean** | 10.165111 | 0.073556 | 0.037778 | 0.055556 | 45.122222 | 54.111111 | 37.122222 | 235.777778 | 238.( |
| **std** | 4.222339 | 0.190046 | 0.185168 | 0.274828 | 10.125979 | 12.386514 | 9.478335 | 98.630827 | 97.3 |
| **min** | 3.580000 | 0.000000 | 0.000000 | 0.000000 | 20.000000 | 25.000000 | 15.000000 | 10.000000 | 10.( |
| **25%** | 6.765000 | 0.000000 | 0.000000 | 0.000000 | 39.000000 | 46.000000 | 30.000000 | 190.000000 | 190.( |
| **50%** | 9.510000 | 0.000000 | 0.000000 | 0.000000 | 44.500000 | 53.500000 | 36.500000 | 270.000000 | 270.( |
| **75%** | 12.472500 | 0.020000 | 0.000000 | 0.000000 | 51.750000 | 61.750000 | 45.000000 | 320.000000 | 320.( |
| **max** | 21.920000 | 0.960000 | 1.100000 | 2.000000 | 65.000000 | 80.000000 | 57.000000 | 360.000000 | 350.( |

8 rows × 21 columns

In [29]:
```
# get the sum of "total precipitation" (PRCP) using the sum function
w["PRCP"].sum()
```

Out[29]:
```
6.62
```

# Summarize trip/rideshare data (from Part B)

In [30]:
```
# get frequency distribution for "Start station" using the value_counts method
tr = pd.read_csv('TripData.csv', encoding = 'unicode_escape')
```

In [31]:
```
# get descriptive statistics for all numeric columns in the trip/rideshare data set
tr.describe()
```

Out[31]:

| | Duration | Start station number | End station number |
|---|---|---|---|
| **count** | 6.465080e+05 | 646508.000000 | 646508.000000 |
| **mean** | 1.010939e+06 | 31317.412205 | 31320.159981 |
| **std** | 2.087019e+06 | 218.034575 | 216.394488 |
| **min** | 6.000400e+04 | 31000.000000 | 31000.000000 |
| **25%** | 3.732280e+05 | 31202.000000 | 31207.000000 |
| **50%** | 6.263670e+05 | 31251.000000 | 31250.000000 |
| **75%** | 1.064398e+06 | 31505.000000 | 31505.000000 |
| **max** | 8.606654e+07 | 32223.000000 | 32223.000000 |

In [32]:
```
# get the mode of "End station" using the mode method
tr['End station'].mode()
```

Out[32]:
```
0    Columbus Circle / Union Station
Name: End station, dtype: object
```

# Part D: Visualizing Data

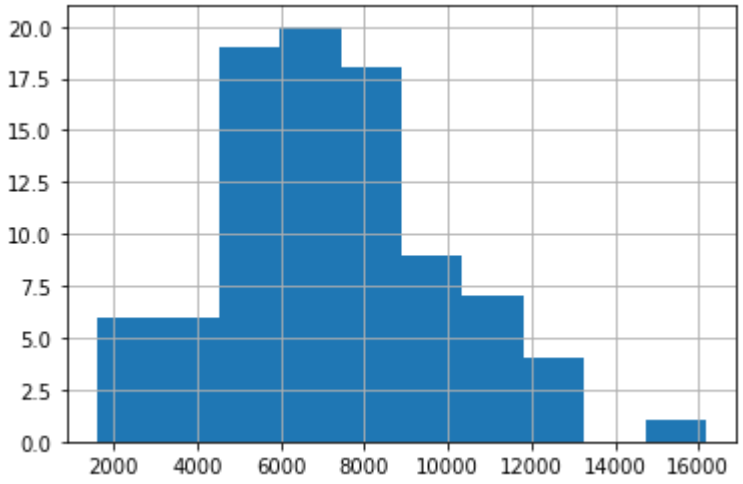
In [33]:
```
# import module for plotting & use jupyter magic to display pictures in cells
import matplotlib.pyplot as plt
%matplotlib inline
```

In [34]:
```
# read in the combined weather and trip data (WeatherTrips.csv)
wt = pd.read_csv('WeatherTrips.csv', encoding = 'unicode_escape')
```

## Create histograms for each numeric variable. Each histogram must have axis labels AND a title

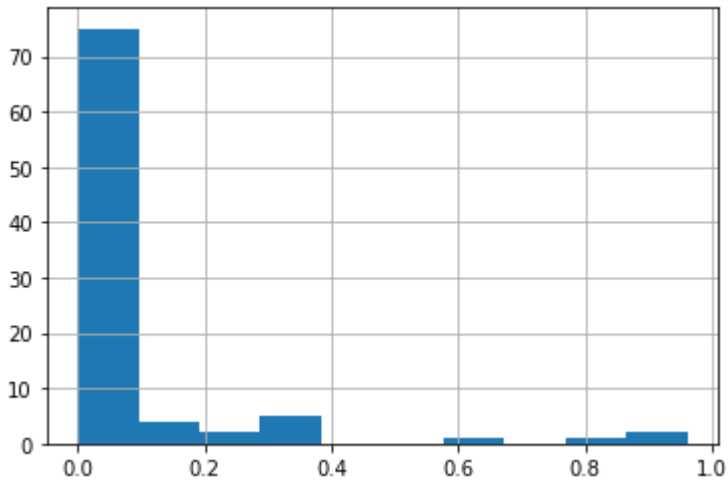
In [35]:
```
# histogram of number of trips (NumTrips)
wt["NumTrips"].hist()
```

Out[35]:
```
<AxesSubplot:>
```



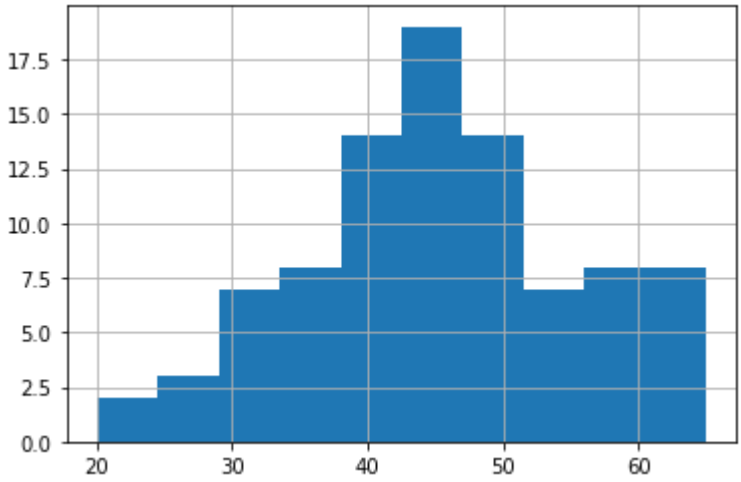

In [36]:
```
# histogram of precipitation (PRCP)
wt["PRCP"].hist()
```

Out[36]:
```
<AxesSubplot:>
```



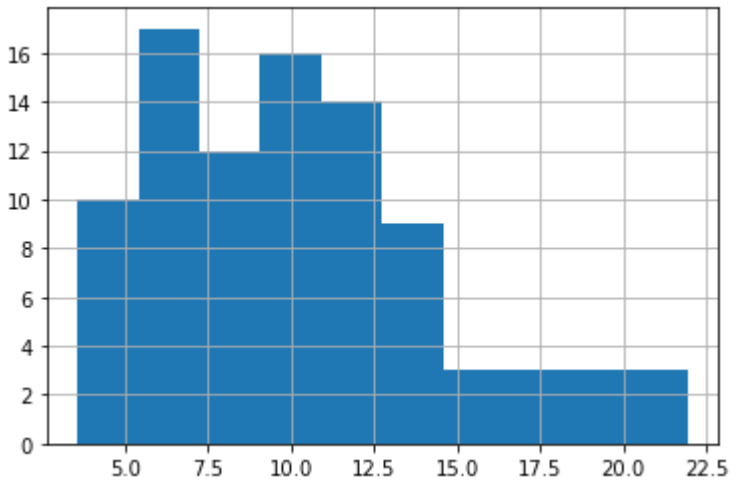

In [37]:
```
# histogram of average temperature (TAVG)
wt["TAVG"].hist()
```

Out[37]:
```
<AxesSubplot:>
```



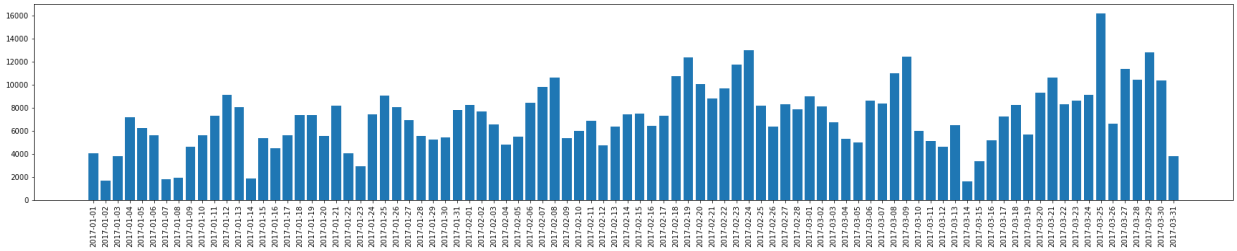

In [38]:
```
# histogram of wind speed (AWND)
wt["AWND"].hist()
```

Out[38]:
```
<AxesSubplot:>
```

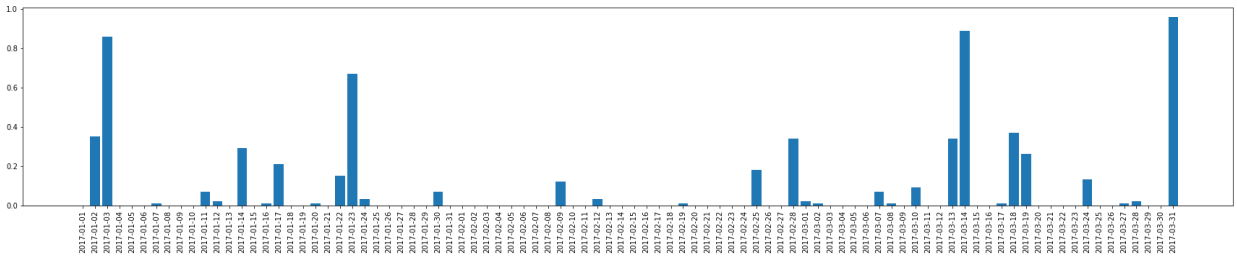



# Create barcharts that display each numeric variable over time

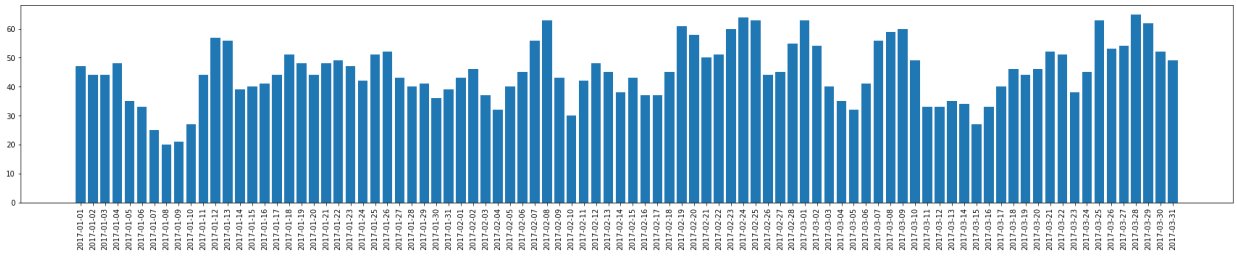
In [50]:
```
# Do NOT DELETE -- these options will make your chart easier to read!
plt.figure(figsize=(30,5))
plt.xticks(rotation=90)

# barchart of date (DATE) vs number of trips (NumTrips)
plt.bar(wt["DATE"], wt["NumTrips"])
plt.show()
```



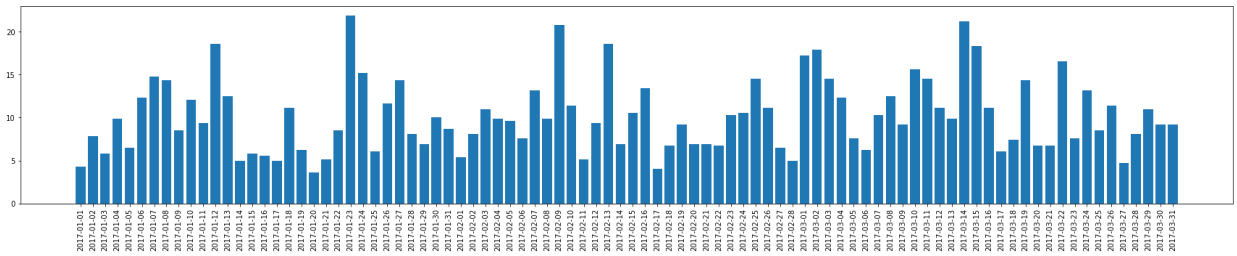
In [49]:
```
# Do NOT DELETE -- these options will make your chart easier to read!
plt.figure(figsize=(30,5))
plt.xticks(rotation=90)

# barchart of date (DATE) vs precipitation (PRCP)
plt.bar(wt["DATE"],wt["PRCP"])
plt.show()
```



In [51]:
```
# Do NOT DELETE -- these options will make your chart easier to read!
plt.figure(figsize=(30,5))
plt.xticks(rotation=90)

# barchart of date (DATE) vs average temparature (TAVG)
plt.bar(wt["DATE"],wt["TAVG"])
plt.show()
```



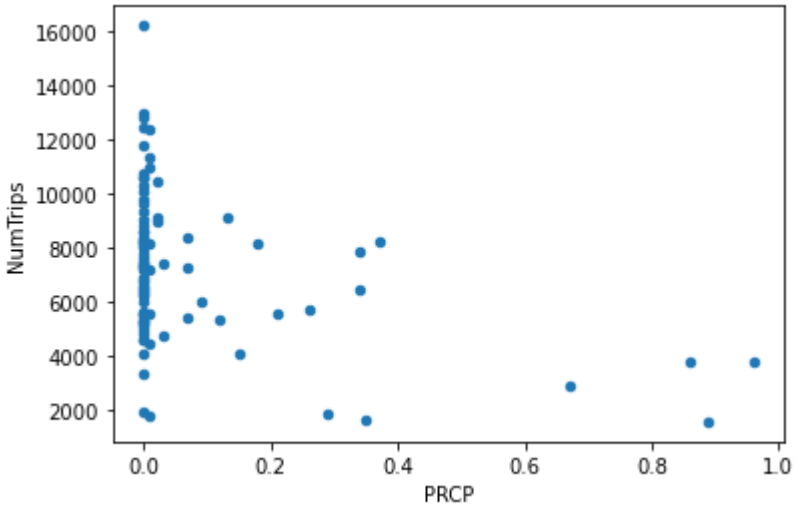
In [52]:
```
# Do NOT DELETE -- these options will make your chart easier to read!
plt.figure(figsize=(30,5))
plt.xticks(rotation=90)

# barchart of date (DATE) vs wind speed (AWND)
plt.bar(wt["DATE"],wt["AWND"])
plt.show()
```



# Create scatterplots of each numeric variable with number of trips

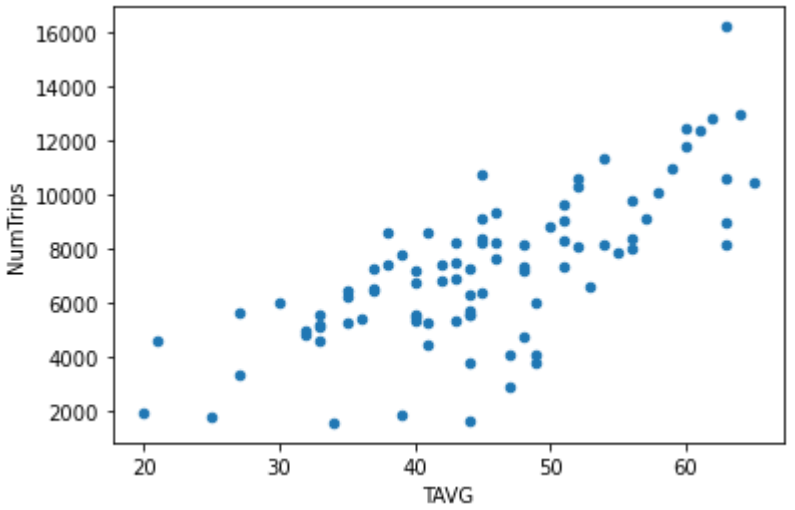
In [43]:
```
# scatterplot of precipitation (PRCP) vs number of trips (NumTrips)
wt.plot.scatter(x = 'PRCP', y = 'NumTrips')
```

Out[43]:
```
<AxesSubplot:xlabel='PRCP', ylabel='NumTrips'>
```



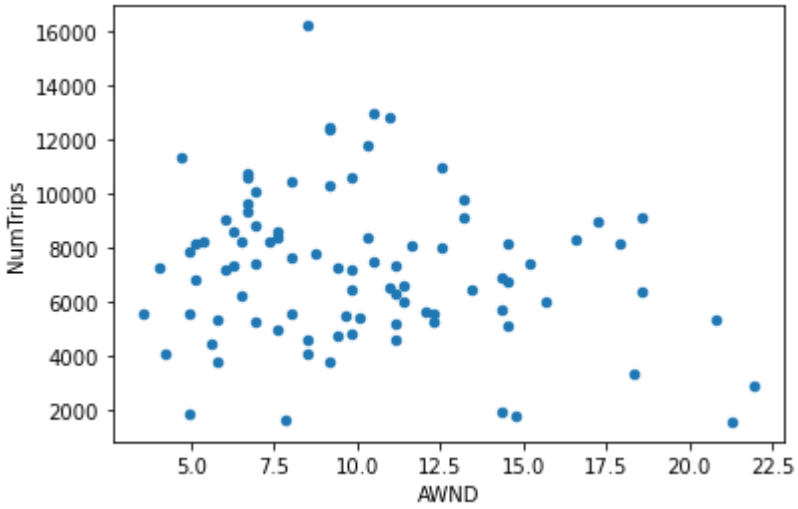

In [44]:
```
# scatterplot of average temperature (TAVG) vs number of trips (NumTrips)
wt.plot.scatter(x = 'TAVG', y = 'NumTrips')
```

Out[44]:
```
<AxesSubplot:xlabel='TAVG', ylabel='NumTrips'>
```




In [45]:
```
# scatterplot of average wind speed (AWND) vs number of trips (NumTrips)
wt.plot.scatter(x = 'AWND', y = 'NumTrips')
```

Out[45]:
```
<AxesSubplot:xlabel='AWND', ylabel='NumTrips'>
```


16000
14000
12000
10000
8000
6000
4000
2000
NumTrips
5.0
7.5
10.0
12.5
15.0
17.5
20.0
22.5
AWND