emmaredfoot / **RAVEN_APS**

Branch: master ▾ | RAVEN_APS / **eia_datetime_general.py** | Find file | Copy path

emmaredfoot Python document which organizes eia data 9b17cab a minute ago

1 contributor

56 lines (52 sloc) | 2.98 KB

```python
#Code to break the EIA APS demand data into seperate seasons
import csv
import os
#Create a csv file to write the data
def writeFile(outfile, monthB, monthE, dayB, dayE):
    #Open the Arizona Public Service Demand data downloaded from EIA
    dataset = csv.reader(open('RAVEN/APS_data_1.csv', newline=''), delimiter=',')
    with open(outfile, 'w') as subfile:
        # Rewrite the first column in the eia data so that it gives the year, the month and day, and the times
        subfile.writelines('Time,Demand'+os.linesep)
        for row in dataset:
            day = row[0][8:10]
            month = row[0][5:7]
            hour = row[0][-5:-3]
            demand =row[1]
            #First check if the months correspond to those in the given season
            if monthB <= int(month)<= monthE:
                #If the month is the last month, make sure that it is only those days on or before the beginning
                if int(month) == monthE:
                    if int(day) <= dayE:
                        subfile.writelines('{0},{1}'.format(row[0][-5:-3], row[1]+os.linesep))
                elif monthB < int(month) < monthE:
                    subfile.writelines('{0},{1}'.format(row[0][-5:-3], row[1]+os.linesep))
                #If the month is the first month, only write those days on or after the final day
                elif int(day) >= dayB:
                    subfile.writelines('{0},{1}'.format(row[0][-5:-3], row[1]+os.linesep))

def writeWinter(outfile, monthB, monthE, dayB, dayE):
    #Open the Arizona Public Service Demand data downloaded from EIA
    dataset = csv.reader(open('RAVEN/APS_data_1.csv', newline=''), delimiter=',')
    with open(outfile, 'w') as subfile:
        # Rewrite the first column in the eia data so that it gives the year, the month and day, and the times
        subfile.writelines('Time,Demand'+os.linesep)
        for row in dataset:
            day = row[0][8:10]
            month = row[0][5:7]
            hour = row[0][-5:-3]
            demand =row[1]
            #First check if the months correspond to those in the given season
            if monthB <= int(month)+12 <= monthE+12:
                #If the month is the last month, make sure that it is only those days on or before the beginning
                if int(month) == monthE:
                    if int(day) <= dayE:
                        subfile.writelines('{0},{1}'.format(row[0][-5:-3], row[1]+os.linesep))
                elif monthB < int(month)+12 < monthE+12:
                    subfile.writelines('{0},{1}'.format(row[0][-5:-3], row[1]+os.linesep))
                #If the month is the first month, only write those days on or after the final day
                elif int(day) >= dayB:
                    subfile.writelines('{0},{1}'.format(row[0][-5:-3], row[1]+os.linesep))


Spring = writeFile('RAVEN/SpringEIA.csv', 3, 6, 20, 20)
Summer = writeFile('RAVEN/SummerEIA.csv', 6, 9, 22, 21)
Fall =  writeFile('RAVEN/FallEIA.csv', 9, 12, 22, 21)
Winter = writeWinter('RAVEN/WinterEIA.csv', 12, 3, 21, 19)
```