PM table to json converter

SEED project

November 24, 2015

Contents

1 Matching and mapping SEED and PM

The following description of the connection between Portfolio Manager and SEED platform is summarized from [1]

For commercial building benchmarking, the city government need to acquire energy consumption data of buildings in the city. As a result, there are requirements to collect building energy consumption information for buildings greater than 5000 sq. ft. The city will create a list of buildings they need to collect energy information from. The owners of the buildings in the list will report the energy consumption data through the Portfolio Manager developed by U.S. Environmental Protection Agency (EPA)and share the report with the city. The city can government can download the data from the Portfolio Manager site into one big folder.

Figure 1 is the diagram of the SEED platform

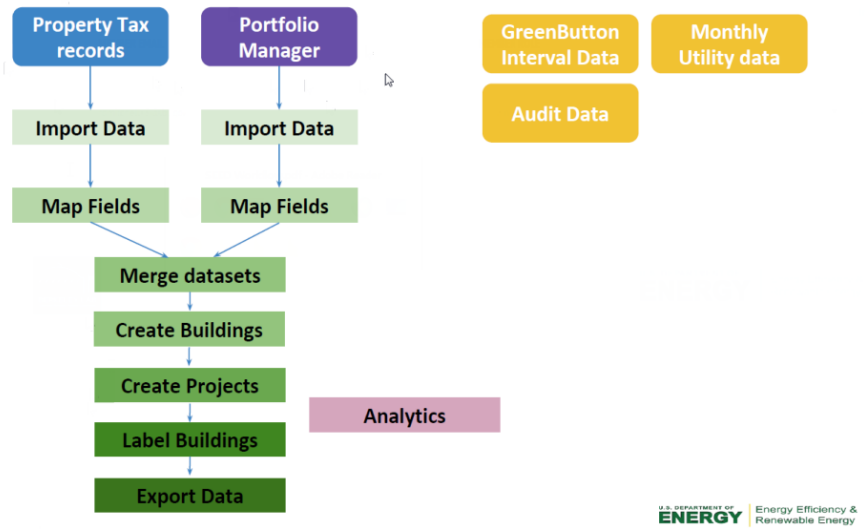

Figure 1: SEED flow diagram
[1]

In order to hold and process the large amount of data, a database is needed. DOE developed the SEED platform to assist cities to manage the building energy consumption data. The city government would import 1) the list of buildings into the SEED database, and 2) import the energy consumption data from the portfolio manager.

The two sets of data just imported would go through a process called **"Map Fields" (Mapping)** which standardized the fields of the two data sources. Then **"Matching"** (using fuzzy logic) is performed between the two so that the building is associated with its energy consumption records. Then the database is ready to carry out analysis.

In the "Mapping" process, the program will predict based on the original name of the excel sheet user uploaded (my understanding) and the user will confirm which field of the original uploaded data is associated to the standard database field.

Other sources of data started to be incorporated into the SEED platform: GreenButton Data (short interval energy consumption data) and Audit Data (Procedures For Commercial Building Energy Audits).

Building list: the fields in the building list include: UBI(?), GBA(Gross Building Area), BLDGS, Address, Owner, City, State, Zip, Property Type, AYB_YearBuilt. PM data has fields such as: Property ID, Property Name, YearEnding, PropertyFloorArea, Address 1, Address 2, City, ... Although the two might both have ID field, they might not be related. The suggested field for matching the building list with the PM data is with Address ("Address Line 1" in SEED). In general, there are four suggested fields that **improve the matching result: Tax Lot ID, PM Property ID ("Property ID" field in PM), Custom ID and Address Line 1**(Figure 2).

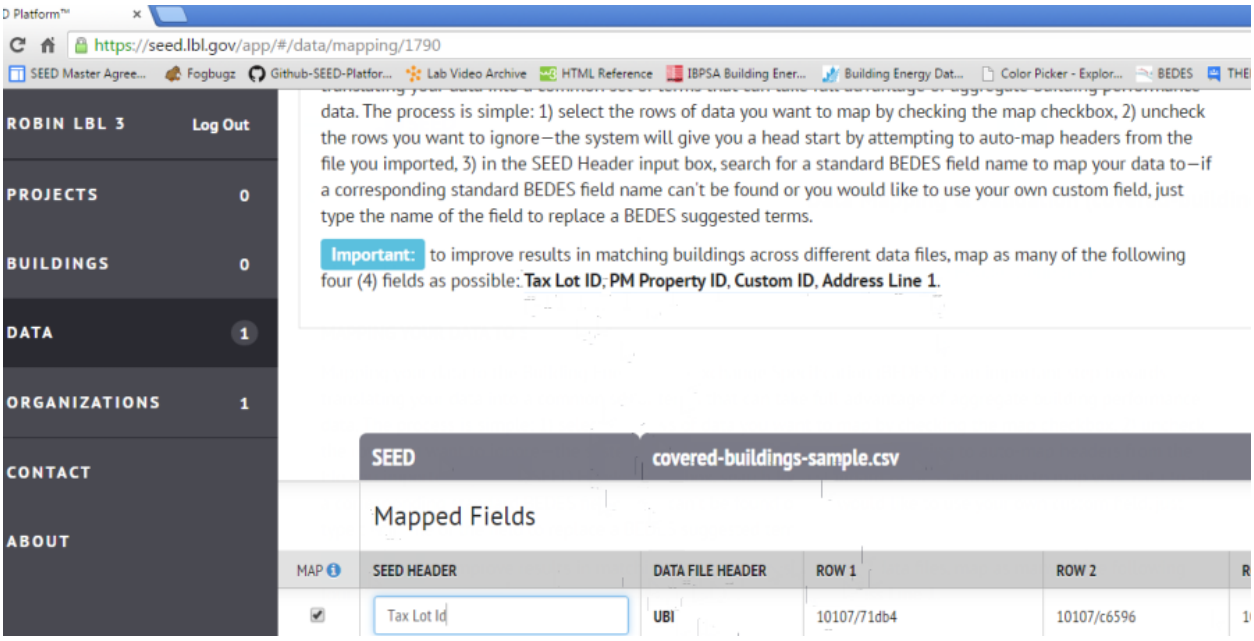

Figure 2: Matching requirement

[1]

2 Template Development

Based on the suggested field for improving data matching performance, the following template is created (Figure 3):

	A	B	C	D	E	F	G	H
1	Street Address	Portfolio Manager Meter ID	Meter Type	Start Date	End Date	Usage/Quantity	Usage Units	Cost ($)
2	5000 Forbes Ave	4674008	Natural Gas	11/1/14 0:00	12/1/14 0:00	100000	kBtu (thousand Btu)	9000
3								
4								
5								
6								

Figure 3: Excel template

3 Code

```
## ## ## ## ## ## ## ## ## ## ## ## ## ## ## ##
## converter.py
## ## ## ## ## ## ## ## ## ## ## ## ## ## ## ##
import os
import pandas as pd
import json

# replace the following directory name and filename
read_dir_name = os.getcwd() + "/PMfile/"
read_file_name = "template.xlsx"
read_sheet_name = 0
write_dir_name = os.getcwd() + "/Jsonfile/"
write_file_name = "test.txt"

def pm2json():
    meter_con_df = pd.read_excel(read_dir_name + read_file_name,
                                 sheetname=read_sheet_name, skiprows=0,
                                 header = 0)

    # Calculate time interval of days
    meter_con_df['interval'] = meter_con_df['End Date'] - meter_con_df['Start Date']
    meter_con_df['reading_kind'] = 'energy'

    # renaming columns of df
    name_lookup = {u'Start Date':u'start',
                   u'End Date':u'end',
                   u'Portfolio Manager Meter ID':u'meter_id',
                   u'Usage/Quantity':u'value',
                   u'Usage Units':'uom'}

    meter_con_df = meter_con_df.rename(columns=name_lookup)

    # the 'interval' output is in [ns], so use ns for all time object and
    # post process with postProcess.py
    meter_con_df.to_json(write_dir_name + write_file_name, 'records',
                         date_unit = 'ns')

## ## ## ## ## ## ## ## ## ## ## ## ## ## ## ##
## postProcess.py
## ## ## ## ## ## ## ## ## ## ## ## ## ## ## ##
import os
import converter as cv

dir_name = os.getcwd() + "/Jsonfile/"
in_file_name = cv.write_file_name
out_file_name = "post_test.txt"

# convert unit of time from 'ns' to 's'
def ns2s(string):
    return string.replace('000000000,', ',')

def postProcess():
    with open (dir_name + in_file_name, 'r+') as infile:
        unit_ns = infile.readline()
        unit_s = ns2s(unit_ns)
    with open (dir_name + out_file_name, 'w') as out:
        out.write(unit_s)

## ## ## ## ## ## ## ## ## ## ## ## ## ## ## ##
## mainroutine.py
## ## ## ## ## ## ## ## ## ## ## ## ## ## ## ##
import converter as cv
import postProcess as ps

cv.pm2json()
ps.postProcess()
```

4 Output

[{"Street Address":"5000 Forbes Ave","meter_id":4674008,"Meter Type":"Natural Gas","start":1414800000,
"end":1417392000,"value":100000,"uom":"kBtu (thousand Btu)","Cost ($)":9000,"interval":2592000,"reading_kind":"energy"}]

References

[1] Lawrence Berkeley National Laboratory. Seed 1.1 tutorial. web, November 2015. https://windows.lbl.gov/projects/SEED/IntroTutorial/SEED%201.2%20Overview.htm.