Introduction to The Flatten Tool For OCDS

# Introduction

* Give an overview of the flatten tool and why or when to use it. Think about the challenge of working with JSON data for analysis.
* Also think about the need to convert between JSON and CSV/spreadsheet formats

Recently I was playing around with some open contracting data which was my first time working with data in json format. If you have ever had to work with open contracting data or generally any multi-deeply nested json file, you’ll relate to my trauma in trying to flatten out the file using pandas, json and json\_normalize in jupyter notebook. Even with that I still did not get my desired output, I had to further write a number of loops and functions. For a beginner to intermediate level data tinkerer like myself, this experience can get very frustrating.

Csv files have been the norm of most data projects I’ve worked on. On few occasions did I deal with spreadsheets which was pretty easy to convert to csv format on the click of a button or command in the terminal.

CSV formatted files are flat i.e. the data has rows and columns just like a table so it’s easier to sort, filter or group. However, data in JSON format is based on a key/value pair relationship. And it gets much complicated as the level of nesting goes deep. Think of it as a tree with many branches where each branch has multiple branches and these multiple branches also have branches which also have branches and so on. To be able to do some data wrangling we need to flatten out all of these branches into a straight long branch.

Don’t even think about it! Trying to totally flatten out the deep nests into a table format would require multiple lines of code I doubt you’re prepared to write when you can’t wait to get your hands dirty on some data.

As most of the data collected from or published to the web is in json format, it is inevitable you will come across a json file to work with in the course of your data journey. JSON files have their tremendous use cases but data analysis is definitely not one of them.

The Flatten tool saves you all the time and effort it takes in writing those multiple lines of code to convert json formatted data into csv’s and vice versa.

# Prerequisites

* What skill level should the reader/audience have
* What tools setup should they have

Before you start this tutorial, I assume that

- You know a thing or two about the command line/terminal and have probably written a few commands

- You have worked with open contracting data or any data in either one of the formats ie json or csv

- You have Jupyter Notebook installed

- You know some python and are familiar with using its libraries working with data

# Environment Setup/Downloading The Tool

* Think about the operating system (start with Ubuntu)

My environment set up was done in Ubuntu 18.04 hence the steps below might not apply in other versions or operating systems. The flatten tool runs on python 3.5 or later and a virtual environment. Virtual environments allow you to create and manage multiple projects with each relying on different versions of python to execute.

First, check your python version by running “python -V” in your terminal. If you’re not using Python 3, to create a virtual environment and install the flatten tool run the following:

sudo pip3 install virtualenv

cd ~

mkdir Virtualenvs

cd Virtualenvs

git clone <https://github.com/OpenDataServices/flatten-tool.git>

cd flatten-tool

virtualenv flatten\_env

source flatten\_env/bin/activate

pip install -r requirements\_dev.txt

If you are using Python 3, then you should already have the venv module from the standard library installed. Run the following:

git clone https://github.com/OpenDataServices/flatten-tool.git

cd flatten-tool

python3 -m venv .ve

source .ve/bin/activate

pip install -r requirements\_dev.txt

# Converting from OCDS JSON to OCDS CSV/Spreadsheet

* Use a sample file (not too big)
* Show examples of the different tool arguments
* Think about the problem that requires flattening a JSON file. For example, “I want a flattened file that just shows the award stage” or “I want a file that has all the stages”

For this tutorial, we’ll use a sample open contracting json file from <https://github.com/open-contracting/sample-data/blob/master/fictional-example/1.0/ocds-213czf-000-00001-04-award.json>

Our quest will be to flatten out this json format and identify:

- To whom the award was given?

- For which items (good/work/service)?

- And for how much ?

We will start our quest by using pandas & json library in jupyter notebook. After which we will explore the flatten tool. This approach will help us identify the limitations of some tools and why others are much efficient.

1. Using Pandas & JSON library

First import relevant libraries

\*insert JN cell\*

Read in our json file

\*insert JN cell\*

Print out the main keys of the json file to know which key has the data we need

\*insert JN cell\*

Flatten out ‘releases.” This gives us some data but looks like what we need is hidden in the awards column as a list of dictionary.

\*insert JN cell\*

Again, flatten out awards column to a new dataframe. Yet still, we are not able to find answers to our questions.

\*insert JN cell\*

Let’s see what awards/awards contains.

\*insert JN cell\*

Alas! There lies the answers we seek. But how do we flatten this into a dataframe?

\*insert JN cell\*

See how cumbersome this process is getting? For a simple json file, going through this process might not be a big deal. But a file as deeply nested as this, we’ll have to think twice.

2. Using Flatten

In your flatten tool environment, run: flatten-tool -h

\*insert terminal command\*

This prints out how to use the tool plus all the arguments it takes. In our case since we want to convert ocds json to ocds csv, let’s run see all the help we can get on flatten: flatten-tool flatten –help.

\*insert terminal command\*

From the output we are able to view all the arguments of flatten. Don;t worry if everything doesn’t make sense. As we determine what we need, we will understand which arguments can help us achieve our aim.

To convert our json file to csv, we run the command below:

flatten-tool flatten ocds-award.json -f=csv --root-id=ocid --main-sheet-name releases --root-list-path=releases -o=ocds-award.csv

Command Breakdown:

flatten: specifies that we want to flatten a file

ocds-award.json: name of the file to be flattened

-f=csv: format of the output file we want

--root-id=ocid: the root id of our file

--main-sheet-name releases: name of the key with the data we need

-o=ocds-award: name of output folder

In your flatten tool folder, you’ll find the folder “ocds-award” with the all the csv files.

We have multiple csv files because, the flatten tool creates a new csv file for every key that has a list value

\*insert image\*

We can then go ahead to merge the multiple files into a dataframe in Jupyter Notebook.

First, import the libraries we need

\*insert JN cell\*

Concatenate all the csv files into a dataframe

\*insert JN cell\*

List all the column heads to get a sense of what data we’re working with

\*insert JN cell\*

Now we have an ideas as to which columns contain the answers we seek.

To find details of supplier information ie to whom the award was given, we run the following code

\*insert JN cell\*

For items awarded;

\*insert JN cell\*

Lastly, the value of the award

\*insert JN cell\*

Easy peasy!

# Converting From OCDS CSV/Spreadsheet to OCDS JSON

* Convert the CSV/spreadsheet file you just generated back to the original JSON
* Show examples of the different tool arguments’

Now, let’s convert back our files from csv to json. Using the flatten tool, we can pass the argument “unflatten.” Run the code in your flatten environment on your terminal:

flatten-tool unflatten ocds-award-csv --root-id=ocid --input-format csv --output-name ocds\_award.json –root-list-path=releases

\*insert terminal command\*

By now, the arguments in the command should be familiar to you

Wait, don’t get too excited yet. You’ll realize that, this new json file does not contain the metadata that the original ocds json file includes. To rectify this we need to create a base json file that will be included in unflattening our csv files.

Found the Flatten tool cool? Read more here: <https://flatten-tool.readthedocs.io/en/latest/>

# Best Practices For Flatten Tool or Common Pitfalls

* Think about errors you faced in the process or using the tool
* Include the solution to them