**Converting JSON to CSV and Loading it to Postgres with Node.js**

To convert JSON to CSV, I love using [json2csv](https://www.npmjs.com/package/json2csv). It really does all the hard work of working the JSON structure out and converting it to a flat file. For nested JSON elements, you can simply specify them by the dot notation (like transaction.itemId). When it contains an array element, it automatically expands it and creates a denormalised table for you. Well, let’s see how it works.

**Input Data**

Here is the input data, transaction.json. I think it is a good representation of standard JSON file you are likely to ingest. It has nested elements in Payment and an array element in Transaction. It follows schemaless patterns with missing elements when they do not exist in the dataset.

[cc lang="text" tab\_size="4" lines="-1"]

[/cc]

**Setup**

As usual, let’s create a project folder and install required modules.

[cc lang="bash" tab\_size="4" lines="-1" theme="mac-classic" line\_numbers="false"]

[/cc]

We also need to create a table that contains all the elements in the source file.

[cc lang="sql" tab\_size="4" lines="-1" theme="mac-classic"]

[/cc]

**Code**

For bulk loading data in to Postgres, see the post [here](https://www.mydatahack.com/bulk-loading-postgres-with-node-js/). The key to convert JSON to CSV is to define the fields and which element to unwind. As Transaction has an array, we will unwind with it. This will automatically create a denormalised table.

There are a few different ways to convert json to csv as in the [documentation](https://www.npmjs.com/package/json2csv). In this example, we are going to use readable stream and pipe the conversion function. This method is memory efficient especially when you have a large JSON file.

After json2csv emits the ‘end’ event, we will call the query execution function. This will ensure that the queries get executed after the csv file creation.

<pre class=" toolbar:2 lang:js decode:true" >

</pre>

Fun!