**REST API Data Ingestion with Node.js**

The classic REST API data ingestion pattern is (1) to make an API call to the endpoint, (2) get the data, (3) transform it to a structured table and (4) load it to a database. Let’s have a go at it with Node.js. We are using [JSONPlaceholder](https://jsonplaceholder.typicode.com/) which offers a few different REST API endpoints for testing or experimenting. Let’s ingest the photos dataset into Postgres database.

To make REST API calls to http endpoints, we can use the [https](https://nodejs.org/api/https.html) module, which comes with Node.js installation and you do not need to install it.

We are using [node-postgres](https://node-postgres.com/features/connecting) for connecting to Postgres, [pg-copy-streams](https://github.com/brianc/node-pg-copy-streams) for bulk loading data, [json2csv](https://www.npmjs.com/package/json2csv) for converting JSON to CSV.

Apart from making GET requests with https, all the data ingestion techniques in this blog have been covered in the previous posts. Check them out for further details.

* [**Bulk Loading Postgres with Node.js**](https://www.mydatahack.com/bulk-loading-postgres-with-node-js/)
* [**Converting JSON to CSV and Loading it to Postgres with Node.js**](https://www.mydatahack.com/converting-json-to-csv-and-loading-it-to-postgres-with-node-js/)

OK, here comes the code.

The GET request is an asynchronous function. During the function execution, we keep appending the incoming data to the buffer variable. Once the ‘end’ event is emitted, we convert the JSON data into CSV and load it to Postgres sequentially. This execution pattern works well.