**How to Ingest Data Into MongoDB with Node.js**

Now that we got data from MongoDB and loaded into Postgres with some transformation, let’s put the data back into MongoDB. This post is the continuation of the previous post about ingesting data from MongoDB. You should totally start from [**How to Ingest Data From MongoDB with Node.js**](https://www.mydatahack.com/how-to-ingest-data-from-mongodb-with-node-js/).

The aim is simple. We are going to get the data from the two tables, restaurants and grades (which were originally created [here](https://www.mydatahack.com/how-to-get-data-from-mongodb-with-python/) with Python or [here](https://www.mydatahack.com/how-to-ingest-data-from-mongodb-with-node-js/) with Node.js), reconstruct original JSON structures and load it back to MongoDB.

The transformation is a little bit more interesting because we have to wrangle normalised tables back into the original JSON format.

Let’s start with putting the JSON back together.

**joinRestaurants.js**

Creating the array with grades aggregated per unique id from the tabular table format is challenging. There may be a better way than using two for loops because for loops are notoriously slow. It works at least.

To rearrange the main table, I used the [node-json-transform](https://www.npmjs.com/package/node-json-transform) module. For any simple JSON wrangling, I always use it because it is simple to use. I don’t think the module does the transformation required for the grades table, though.

In the end, the two JSON objects are joined by the common key (\_id).

The function takes the parent and child (which means restaurants and grades) JSON object as argument parameters. We will export this module.

**mongoLoad.js**

This is a straight forward function that takes database name, collection name and JSON object as parameters. It truncate the collection first and load the data. The connection details are imported as in the [previous post](https://www.mydatahack.com/bulk-loading-postgres-with-node-js/).

**main.js**

Now everything comes together. In the main file, it queries two tables and pass the results to the join function. Once the data is joined, it gets passed to the data load function. The db connection details are imported as in the [previous post](https://www.mydatahack.com/bulk-loading-postgres-with-node-js/).