

# Christine McClure

## Projet 2: Reorganizing Galvin Library's E-resources

Summary: The [databases page](#) of the Galvin Library website lists databases by name and subject. There is also another list of resources by [format](#), but this page is difficult to find and rarely used by students. I want to combine this information as part of the library website redesign in Drupal to make these items more easily found.

Github repository:

<https://github.com/christinemcclure/com541-proj2>

### Data Review and Cleanup

I started with looking at the existing data structure for the e\_resources table, which exists in a MySQL database, and copying it to [a file in my Github repository](#). I also did the same for the [resource formats](#) currently listed in the database. For the subjects, I had previously collected a list of course prefixes used and their corresponding subjects for another part of the website redesign, so included that as [another file](#).

I imported the resources table into a Microsoft Excel to review the data. There was a lot of cleanup to be done, mostly removing HTML code that had been embedded into the descriptions. The table also included two separate description fields: one limited-text field to use for an initial short description, and another memo field to use for longer descriptions. I kept only the long description because text can be truncated programmatically.

### JSON Formatting

After deciding which fields to keep from the existing table, I needed to get more than 250 records into JSON format. From a previous project, I found someone who had written a script to [export JSON records](#) from a Google spreadsheet. The process became a series of formatting records in Excel, exporting the [Excel file to a CSV file](#), importing that into Google Spreadsheets, then exporting as JSON. This was a lot of steps, but there was a lot of data clean up that I did by using Excel formulas. I could have translated those formulas into ones usable by a Google spreadsheet, but I'm used to using Excel so I stayed with that.

My database file was exported as a single JSON object, so I split that into [individual files](#). I then determined what types of file summaries would be helpful. I decided to create the following indexes:

[activeDatabases.json](#): An array of all active database ids.

[inactiveDatabases.json](#): An array of all inactive database ids.

[activeDatabaseTitles.json](#): An array of JSON objects for resources that are currently active and their titles.

[activeDatabasesBySubject.json](#): An array of objects that includes the subject key mapped to an array of corresponding database IDs.

[activeDatabasesByFormat.json](#): An array of objects that includes the format key mapped to an array of corresponding database IDs.

### **Data Considerations**

Transferring the data back and forth caused problems. First with the database ID field: Excel will allow you to create zero-filled number fields, but Google spreadsheets will not. I decided to not worry about this because it is a sample set of data, but it is something to be aware of if there is a large data translation included in the project.

There was also a problem with the JSON converter script putting all of the fieldname into lowercase, which I didn't want: camelCase is much easier to read when conjoined words are used as fieldnames and it is already a convention in JavaScript. I tried to modify the script to leave the fieldnames as they were, but couldn't get it working and so just did a search and replace once the files were created. The script also converted binary values into uppercase TRUE and FALSE values, which also wasn't necessary. For a larger project, it would be beneficial to write your own script to convert your own data.