Paper 1734-2018

Using Maps with the JSON LIBNAME Engine in SAS®

Andrew Gannon, The Financial Risk Group, Cary NC

ABSTRACT

This paper serves as an introduction to reading JSON data via the JSON LIBNAME engine in SAS®. The engine includes an automap that dynamically reads the data into a generic data set. This paper goes into detail about specifying and creating your own map—effectively how to build custom data sets from the JSON data. It covers the input options that can be stated in the JSON map, as well as the subsequent effects those changes produce on the generated SAS data set.

README

For many years XML was a leading markup language, but now JavaScript Object Notation (JSON) is taking its place. The LIBNAME Engine has a built-in function for JSON and an automap feature to dynamically build a map based on the JSON data that is read in – first included in SAS 9.4 – M4 release. The engine also includes an option to include a user defined map. This map is a separate file that the engine uses to build the data structure and then read in JSON. The map can contain several sections, each of which build a separate dataset that will be stored in the library that is created with the LIBNAME statement. Without a specified map, the system will auto-define a map which can result in the creation of several different datasets – built from the hierarchy of the data (this is rarely recommended).

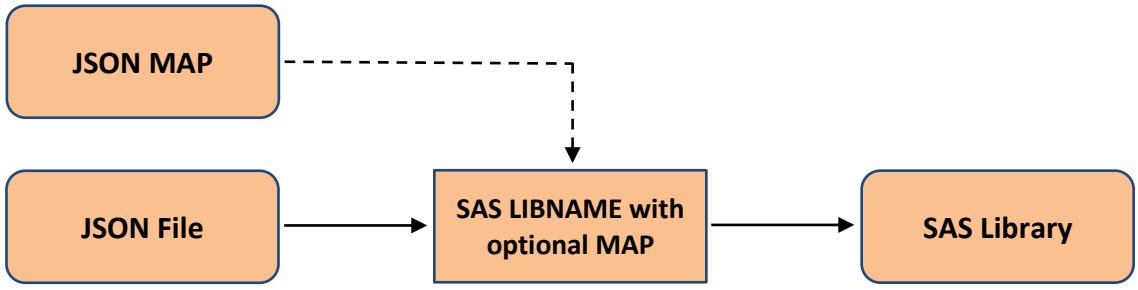

Figure 1. JSON LIBNAME Engine Flow

JSON FILE

The JSON file can follow all the JSON conventions and syntax specified in the convention. Both object and array statements in the JSON are acceptable for use with the JSON LIBNAME engine. Character values must be enclosed in double quotes – single will not work. There are also some important notes about how the LIBNAME engine handles missing values from the JSON file. For character variables, as stated in the JSON syntax rules, the values must be enclosed in quotes. If the character value is missing, empty quotes will work to represent the empty string. Numeric values are quite different though as they do not use quotes for their values. For an empty or missing numeric value the value **must** be set to *null* (without quotes around it). If the value of the numeric is missing or set to a period (as a missing SAS numeric might be) then the LIBNAME step in the SAS code will error.

JSON MAPS

REQUIRED MAP ATTRIBUTES

The JSON map that is used to create the resulting dataset(s) is very important for good data management. Without this map, the engine will auto-create one and use it – which is not necessarily a negative aspect, but the automap will read **all** the data in and create datasets for multiple hierarchies that exist in the data. The JSON map is available for efficiency, preventing one from reading in insignificant

data or having to read in all data and then create custom datasets from it. The map allows all of this to be completed inside of the LIBNAME statement.

The required attributes needed for the map are: DATASETS, DSNAME, TABLEPATH, and VARIABLES. These specify the various datasets and their names, as well as their location and variable attributes within the JSON file. The DATASETS attribute is a JSON array containing all the datasets that are to be created. Within each DATASET variable are the remaining attributes. DSNAME will create the name of the SAS dataset and the TABLEPATH will tell the engine where to find the data within the JSON file (lowest level of hierarchy for that data).

The VARIABLES statement is also an array and contains all the variables and their attributes that are to be included in the parent dataset. Inside the VARIABLES array, three pieces are required: NAME, TYPE, and PATH. NAME is the variable name that will appear on the SAS Dataset, TYPE is what the variable is (either numeric, character, or ordinal), and PATH is the location of this variable within the JSON file. The PATH and TABLEPATH values **are** case sensitive *(/home/test* is not the same as */home/Test*).

There are several other attributes that can be used that are optional that will be covered later in this paper. For now, let's look at a sketch of how the LIBNAME reads the map and subsequent JSON file to create a new SAS dataset. The JSON map tells the engine how to construct the dataset named 'Qtr1_Proft' and that it will have two variables. One variable is denoted as character and the other as numeric. This is a simple and minimum requirement map that can thusly be used to read in a file that has the same structure and variables. If you wanted the same LIBNAME statement to create a second dataset, all that is necessary is to add another object to the 'DATASETS' array after the first. If this is done correctly, then the library that is produced would have the Qtr1_Profit dataset as well as the other newly created dataset contained within it.

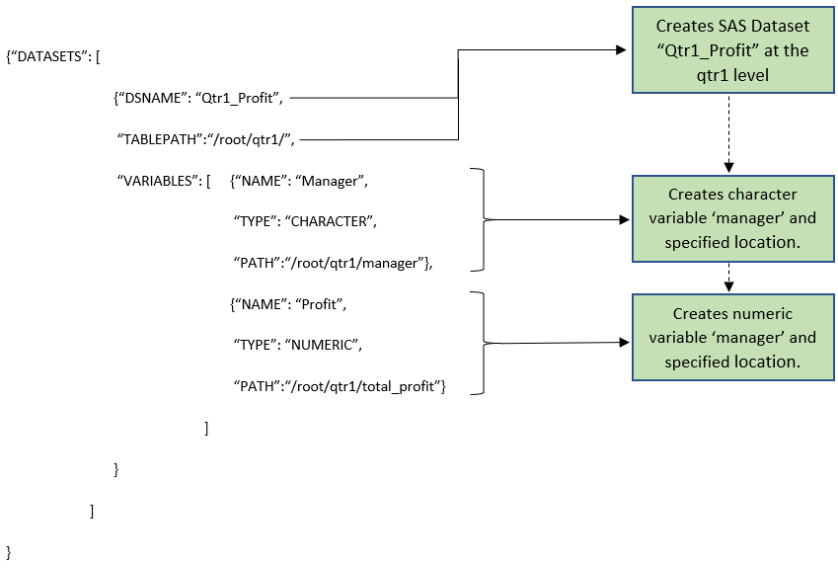

Figure 2. JSON Map Diagram

OPTIONAL MAP ATTRIBUTES

For character variables, it is usually a good practice to include a **LENGTH** attribute in the variable section. The length statement is simply an option that tells the dataset to use that length. It will initially default the length to the longest character string that is read in for that variable.

Map Attributes	**SAS Equivalent**
"INFORMAT": ["BEST",12,1]	BEST12.1
"FORMAT": ["COMMA",10,2]	COMMA10.2

Figure 3. Map Formatting Options

JSON Maps can also handle SAS formats and informats with the **FORMAT** and **INFORMAT** attributes. These attributes are specified within a variable array. Both attributes have the same structure, inside square brackets put the informat or format name, followed by the width, and then decimal specification. The example here shows what would be included in the variable section. Notice that you separate the pieces with commas and that

character variables must be in quotes. The first number after the format is the total length and the second is the number of decimals. You can use both a format and informat with the same variable.

The SAS dataset(s) that are created also have the option to have labels added upon being read in with the **LABEL** option. This is as simple as it sounds, inside the variable attributes include a label option with the value being the label that is desired.

The final option, and one of the most important, is the **OPTION** attribute. There are only two options currently, RETAIN and NORETAIN. This simply tells SAS to keep the variable name from one observation to the next if they are the same. If this is not specified, there we be nothing retained. For example, In the previous example with the manager and profit variables, if a manager had three items for profit, only the first line for the manager would contain a value. The two datasets provided are the difference between having the "OPTION":["RETAIN"] or ["NORETAIN"] included in the variable section of the JSON file. Note that variables designated as ordinal will always be retained. This is a very important attribute since typically all the data should be included for each record.

	Manager	Profit
1	JSmith	12.3
2		34.7
3		31.8
4	KJohnson	24.4
5		40.1

	Manager	Profit
1	JSmith	12.3
2	JSmith	34.7
3	JSmith	31.8
4	KJohnson	24.4
5	KJohnson	40.1

Figure 4. Retain vs. NoRetain Option

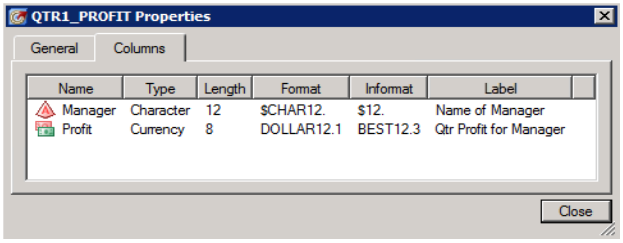

Figure 5. Dataset Metadata View

These options can fulfill all the attributes associated with a SAS dataset. The below figure shows that the optional attributes of the JSON map can specify any SAS format, informat, and label to a dataset during the LIBNAME invocation – giving the user the ability to manage data quality and data transfer all in one step.

JSON LIBNAME ENGINE

The JSON LIBNAME engine is what is used to take the JSON file, along with the JSON map, to create the desired dataset(s). It is very simple to use as it is only a special case of the LIBNAME statement.

```
LIBNAME JSONLIB JSON MAP=JSONMAP ACCESS=READONLY;
```

Start with the LIBNAME portion, followed by the name of the fileref that holds the JSON file. Next we specify JSON to tell the LIBNAME statement what engine to use. The MAP= option points to the map (located at fileref JSONMAP). Finally, the optional attribute ACCESS= is set to read only so that the resulting library can be used but not modified (this is personal preference).

There is an AUTOMAP= option that will create its own mappings based on the JSON file. As mentioned earlier, this paper focuses on maps that are user created. The AUTOMAP= can be quite helpful though, if it is set to CREATE, it will write out the map that is automatically created to the specified fileref. This allows you to see how SAS would build the map, then make changes as necessary (start with the auto generated map – use it as a template). In the below example, SAS creates a map based on the JSON file, then writes that map to the fileref *jsonmap,* while at the same time, creating the SAS datasets in the library *jsonlib.*

```
LIBNAME JSONLIB JSON MAP=JSONMAP AUTOMAP=CREATE ACCESS=READONLY;
```

Note that using this method will overwrite anything that was currently written to the fileref associated with the MAP= portion of the LIBNAME statement.

MORE INFORMATION

Included at the end of this paper are several appendices. They include a sample JSON input, JSON map, and SAS code to read in the input and map. It also includes the outcomes of running the code. For more information regarding the JSON LIBNAME engine, SAS has excellent documentation on the web (see references).

REFERENCES

SAS. “The JSON LIBNAME Engine: Real-World Applications” 2017. By M. Drutar http://support.sas.com/resources/papers/proceedings17/SAS0380-2017.pdf

JSON.ORG. “Introducing JSON” 2017. www.json.org

SAS. “LIBNAME Statement, JSON Engine” 2017. http://go.documentation.sas.com/?cdcId=vdmmlcdc&cdcVersion=8.1&docsetId=lestmtsref&docsetTarget=n1jfdetszx99ban1rl4zll6tej7j.htm&locale=en

CONTACT INFORMATION

If you have any comments or concerns, please feel free to reach out at any time. Please contact at:

Andrew Gannon

The Financial Risk Group, Inc.

+1 (919) 439-3819

Andrew.Gannon@frgrisk.com

www.frgrisk.com

APPENDX A --- JSON MAP

This is the map (json_map.map) for the test code provided in Appendix B.

```
{"DATASETS": [
             {"DSNAME": "all",
              "TABLEPATH": "/root/info/data",
              "VARIABLES": [
                           {"NAME": "division",
                            "TYPE": "CHARACTER",
                            "PATH": "/root/division"},
                           {"NAME": "manager",
                            "TYPE": "CHARACTER",
                            "LABEL": "Manager on Duty",
                            "OPTIONS": ["RETAIN"],
                            "LENGTH": 16,
                            "PATH": "/root/info/name"},
                           {"NAME": "month",
                            "TYPE": "CHARACTER",
                            "PATH": "/root/info/data/month"},
                           {"NAME": "profit",
                            "TYPE": "numeric",
                            "FORMAT": ["DOLLAR",12,1],
                            "PATH": "/root/info/data/profit"}
                           ]
             }
             ]
      }
```

APPENDX B --- SAS CODE

This is the SAS code that runs the JSON LIBNAME engine with the map specified in Appendix A.

```
/* This first part saves JSON to a macro variable named _data */
%let _data= [{"division": "sales",
             "info" : [
               { "name" : "JSmith" , "qtr" : 1, "data" : [
                  { "month" : "jan", "profit" : 23.6 },
                  { "month" : "feb", "profit" : 14.7 },
                  { "month" : "mar", "profit" : 21.2 }
                                                        ]
               },
               { "name" : "MAnderson",  "qtr" : 1, "data" : [
                  { "month" : "jan", "profit" : null },
                  { "month" : "feb", "profit" : 18.6 },
                  { "month" : "mar", "profit" : 19.2 }
                                                        ]
               }
                     ]
            },
            { "division": "service",
              "info" : [
                 { "name" : "THarris" , "qtr" : 1, "data" : [
                    { "month" : "jan",   "profit" : 7.3 },
                    { "month" : "feb", "profit" : 9.5 },
                    { "month" : "mar",   "profit" : 9.4 }
                                                        ]
                 }
                       ]

            }
           ];
/* Create Temporary file where macro variable _data can be written to */
/* Create fileref to the location of the map (from Appendix A)        */
filename jsontemp temp lrecl=32767;
filename jsonmap '/home/json/json_map.map';
/* Macro variable _url_return contains JSON output from URL */
/* Write macro variable to the temp file                     */
data _null_;
   length json $32767;
   file jsontemp;
   json = resolve('&_data');
   put json;
run;
/* Read in temp file with LIBNAME JSON engine */
/* Library JSONTEMP now available             */
libname jsontemp json map=jsonmap access=readonly;
```

	division	manager	month	profit
1	sales	JSmith	jan	$23.6
2		JSmith	feb	$14.7
3		JSmith	mar	$21.2
4		MAnderson	jan	.
5		MAnderson	feb	$18.6
6		MAnderson	mar	$19.2
7	service	THarris	jan	$7.3
8		THarris	feb	$9.5
9		THarris	mar	$9.4

Figure 6. Resulting Dataset