Paper 4081-2020

Using Base SAS® Code to Dynamically Load Data to the LASR Analytic Server

Andrew Gannon, The Financial Risk Group

ABSTRACT

This paper serves as an introduction to loading data into the LASR Analytic Server for use in Visual Analytics (VA) using SAS code. Many organizations have robust solutions built in SAS that create reports. Some of these reports are in output files such as XLSX or PDF, others are in the form of OLAP Cubes. As Visual Analytics is swiftly moving into many organizations as the premier reporting tool, the need to automatically and dynamically load data into the platform becomes more crucial. Currently many users utilize the Data Preparation tab inside of Visual Analytics to load their data in. The purpose of this paper is to show how to use base SAS code to dynamically load data into the LASR Analytic Server for use in the Visual Analytics environment.

README

This paper will use several server specific assumptions. It is important to note that these can be different depending on the SAS installations. Material in this paper should work for most installations and is a good starting point. If you have any issues running this code, then you should reach out to your SAS Administrator for guidance in ensuring that the connections are correct. Metadata library, folder, and repository information used in this paper may need adjustments depending on your SAS installation. For the example data used in this paper, we will be referencing the CLASS dataset that can be found in the SASHELP library.

INTRODUCTION

SAS Visual Analytics (VA) is a powerful reporting tool for data presentation. Whether you are building intricate visual presentations or just want a simple cross-tabulation, it has the capabilities to be the single hub for all reporting. Part of VA is dedicated to data preparation and exploration with both the Data Preparation and Data Exploration tabs. Many individuals and organizations have legacy reporting processes built into their SAS solutions. Rather than continue executing those solutions and then manually loading their data into VA via the point-and-click interface, users can add in custom SAS code to automatically load their data into the LASR Server. From here, the data can then be used in in VA.

LOADING DATA TO LASR WITH SAS CODE

PROCESS FLOW

There are several steps involved in loading data directly to the LASR server. Not only do we need to copy the data to the LASR server, but we must also remove legacy data and mange metadata related to the tables. The order of the steps is important.

CONNECTING TO LASR

The first step is to connect to the LASR Server. This can be done via the SASIOLA LIBNAME Engine. This engine allows SAS to transmit data between the current SAS session and the specified LASR Server. Below is the general example of the connection:

```
LIBNAME LASRTEMP SASIOLA
   tag = vapublic
   port = 10031
   host = "url.com"
   signer = "http://url.com:7980/SASLASRAuthorization";
```

This creates a library named *lasrtemp* that is connected to the LASR Server for the specified URL (URL for the web application for VA). There are several other options available for the libname statement that can be found in the SASIOLA documentation.

DELETE UNDERLYING TABLES AND METADATA

Prior to loading a table into LASR, ensure that there is not currently a table with the same name. To do this, we can apply a simple check for the table name, and – if it already exists – we can delete it. This must be done because SAS will not allow the overwrite of an existing table on LASR. To drop the physical table, we simply execute the following code:

```
%if %sysfunc(exist(lasrtemp.class)) %then %do;
   proc sql noprint;
      drop table lasrtemp.class;
   quit;
%end;
```

To doubly ensure that we have removed all related information from the existing table, it is also good to delete any underlying table metadata. This can be done through executing the following code:

```
%if %sysfunc(metadata_pathobj(,/Shared Data/SAS Visual Analytics/Public/LASR/class,table,,)) %then %do;
   proc metalib;
      omr (library = "/Shared Data/SAS Visual Analytics/Public/Visual Analytics Public LASR"
           repname = "Foundation"
          );
      select = ("class");
      update_rule = (delete);
   run;
%end;
```

```
NOTE: A total of 1 tables were analyzed for library "Visual Analytics Public LASR".
NOTE: Metadata for 0 tables was updated.
NOTE: Metadata for 1 tables was deleted.
NOTE: Metadata for 0 tables was added.
NOTE: Metadata for 0 tables matched the data sources.
NOTE: 0 tables listed in the SELECT or EXCLUDE statement were not found in either the metadata or the data source.
NOTE: 0 other tables were not processed due to error or UPDATE_RULE.
```

Figure 1 - Metadata Deletion Output

This will ensure that both the metadata table objects and the physical tables have been completely removed prior to attempting to load data with the same name and attributes. Note that the *metalib* procedure will only delete the table metadata if the underlying physical table has already been deleted. Order is important here, you must first drop the table, then delete the metadata.

CREATE TABLES AND METADATA IN LASR

Now that we are connected to the LASR server and we have ensured that underlying physical and metadata tables have been removed, the table can be loaded. This is the simplest part. Because the SASIOLA Engine works like most other SAS Libname engines, we can simply assign the dataset through the data step like in the example below:

```
data lasrtemp.class;
   set sashelp.class;
run;

NOTE: There were 19 observations read from the data set SASHELP.CLASS.
NOTE: The data set LASRTEMP.CLASS has 19 observations and 5 variables.
```

Figure 2 - Load to LASR Output

We can see from the log that the SASHELP.CLASS dataset was copied to the LASRTEMP.CLASS dataset, which is now stored on the LASR Server.

Finally, we need to register that table in metadata. Similar to deleting underlying metadata, we will again use the *metalib* procedure to create new metadata. Below is the code for creating the metadata for the table:

```
%if %sysfunc(exist(lasrtemp.class)) %then %do;
   proc metalib;
      omr = ( library = "/Shared Data/SAS Visual Analytics/Public/Visual Analytics Public LASR",
              repname = "Foundation"
            );
      select ("class")
   run;
%end;
NOTE: A total of 1 tables were analyzed for library "Visual Analytics Public LASR".
NOTE: Metadata for 0 tables was updated.
NOTE: Metadata for 1 tables was added.
NOTE: Metadata for 0 tables matched the data sources.
NOTE: 0 tables listed in the SELECT or EXCLUDE statement were not found in either the metadata or the data source.
NOTE: 0 other tables were not processed due to error or UPDATE_RULE.
```

Figure 3 - Metadata Creation Output

The table is now loaded to the LASR Server and has underlying metadata. The table can now be viewed inside of VA.

Note on Memory

It's important to note, at this point, that the SAS LASR environment is *"a secure, multi-user environment for concurrent access to data that is loaded into memory."* The key to that statement being the *"into memory"* part. As you load more data into LASR, you will need to account for the memory usage on the given server.

VIEW ON VISUAL ANALYTICS PLATFORM

The tables are now available for use in VA. Navigate to the Report Designer page on VA. Click on the Data tab and select Add Data Source. A pop-up will appear with a list of available data sets for use. The data loaded previously, *class*, is now available in the list to be used.

Below is the *Class* dataset, in a height and weight scatter plot, in VA. The CLASS data can be found in the list of available data in the pop-up of the data tab.

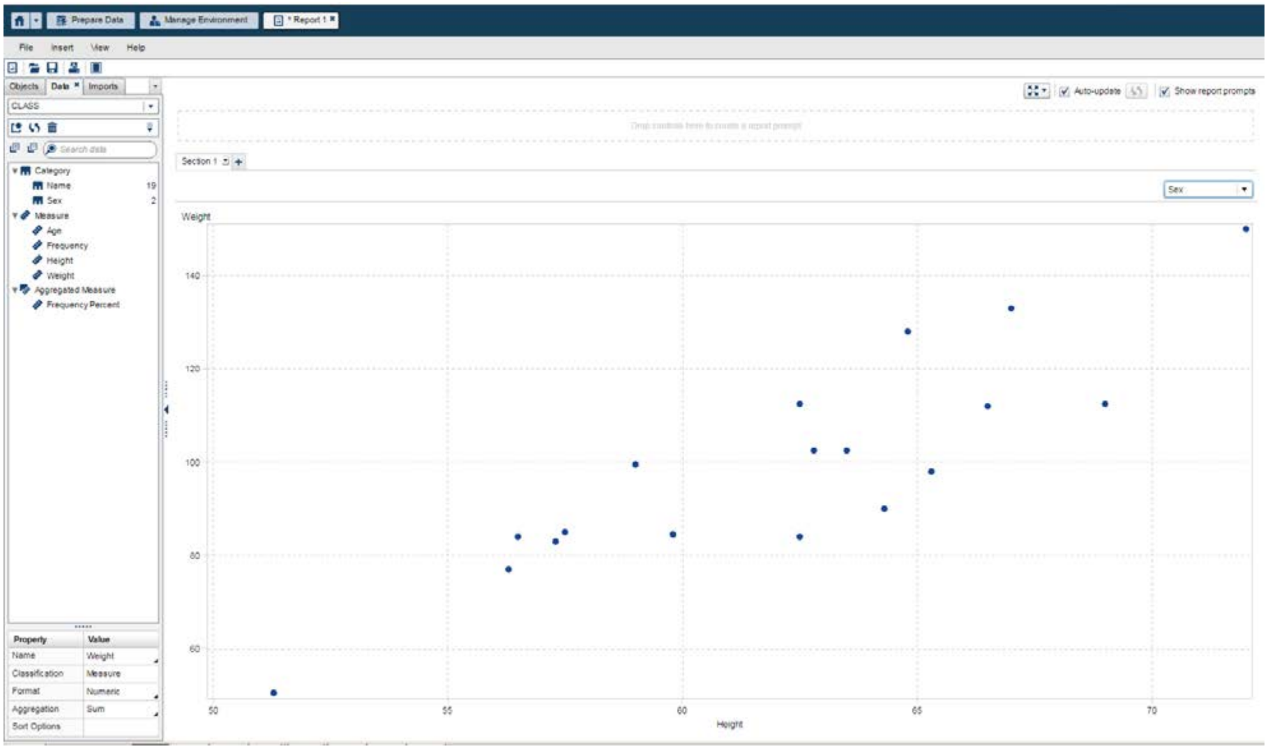

Figure 3 - CLASS Data in VA

CONCLUSION

The goal of this paper was to show how data can easily be loaded into the LASR server for use in SAS Visual Analytics with SAS code. With the ability to dynamically apply the code to already existing processes, as well as new processes, users can avoid having to manually load data into VA. When these processes are well defined and have been thoroughly tested, this avoidance on manual loads can save time, eliminate manual human error, and make the lives of business users simpler. The appendix of this paper contains macro versions of the code seen in this paper. These macros can be easily adjusted to fit the requirements of different solutions and systems.

REFERENCES

SAS Institute Inc. 2015. SAS® LASR™ Analytic Server 2.7: Reference Guide. Cary, NC: SAS Institute Inc.

SAS Institute Inc. 2016. SAS® 9.4 Language Interfaces to Metadata, Third Edition. Cary, NC: SAS Institute Inc.

CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the author at:

Andrew Gannon
The Financial Risk Group, Inc.
+1 (919) 439-3819
Andrew.Gannon@frgrisk.com
www.frgrisk.com

APPENDIX

Below is a macro that will load a given table to the LASR server:

```
/* ------
 * Macro Inputs:
 * IN_TABLE - table, with libname reference, to be copied.
 * OUT_TABLE - name of the table to be created on LASR.
 * ------ */
%macro load_to_lasr(in_table=sashelp.class, out_table=class);

   /* Connect to LASR Analytic Server */
   libname lasrtemp sasiola
      tag=vapublic
      port=10031
      host="url.com"
      signer="http://url.com:7980/SASLASRAuthorization";

   /* Drop Physical Table - if exists */
   %if %sysfunc(exist(lasrtemp.&out_table)) %then %do;
      proc sql noprint;
         drop table lasrtemp.&out_table;
      quit;
   %end;

   /* Drop Table Metadata - if exists */
   %if %sysfunc(metadata_pathobj(,/Shared Data/SAS Visual Analytics/Public/LASR/&out_table,,))
   %then %do;
      proc metalib;
         omr (library="/Shared Data/SAS Visual Analytics/Public/Visual Analytics Public LASR"
              repname="Foundation"
             );
         select = ("&out_table");
         update_rule = (delete);
      run;
   %end;

   /* Load Table to LASR Analytic Server */
   data lasrtemp.&out_table;
      set &in_table;
   run;

   /* Create Metadata for Table - if exists */
   %if %sysfunc(exist(lasrtemp.&out_table)) %then %do;
      proc metalib;
         omr (library="/Shared Data/SAS Visual Analytics/Public/Visual Analytics Public LASR"
              repname="Foundation"
             );
         select ("&out_table");
      run;
   %end;
   %else %do;
      %put ERROR: Physical Table &out_table Does Not Exist - No Metadata Created.;
   %end;

%mend load_to_lasr;
```