Paper SAS1952-2015

# SAS® Visual Analytics Environment Stood Up? Check! Data Automatically Loaded and Refreshed? Not Quite

Jason Shoffner, SAS Institute Inc., Cary, NC

## ABSTRACT

Once you have a SAS® Visual Analytics environment up and running, the next important piece to the puzzle is to keep your users happy by keeping their data loaded and refreshed on a consistent basis. Loading data from the SAS Visual Analytics user interface is a great first start and great for ad hoc data exploring. But automating this data load so that users can focus on exploring the data and creating reports is where the power of SAS Visual Analytics comes into play. By using tried-and-true SAS® Data Integration Studio techniques (both out of the box and custom transforms), you can easily make this happen. Proven techniques such as "sweeping" from a source library and "stacking" similar HDFS tables into SAS® LASR™ Analytic Server for consumption by SAS Visual Analytics are presented using SAS Visual Analytics and SAS Data Integration Studio.

## INTRODUCTION

SAS® Visual Analytics has powerful integrated applications and processes to load data into SAS® LASR™ Analytic Server so that the data is consumable through reports and explorations. These include SAS® Visual Analytics Data Builder, the import data wizard used when selecting a data source from SAS Visual Analytics Designer or SAS Visual Analytics Explorer, and the autoloader. All of these are available out of the box with SAS Visual Analytics. However, in many cases they do not meet the needs for an enterprise-level SAS Visual Analytics environment in terms of flexibility and insulation between different groups of users. This paper introduces methods used to implement additional data-loading concepts to create a more robust batch environment for keeping the data loaded and refreshed within SAS Visual Analytics.

The sweeper concept enables you to load all (or a subset) of the data sets, tables or views from a source library to memory. After these objects have been loaded to memory, they are registered into metadata so that they can be consumed by reports and explorations in SAS Visual Analytics. The stacker, on the other hand, enables you to load a subset of similar source data sets, tables or views into a single table in memory. There are several transforms that make up these data-load techniques. These transforms can be used in tandem to gain all the functionality or independently for a more simplified approach.

**Figure 1** is a screenshot from the SAS® Data Integration Studio Transformations tab. It shows the 4 resulting transformations used to implement the expanded data-loading techniques.

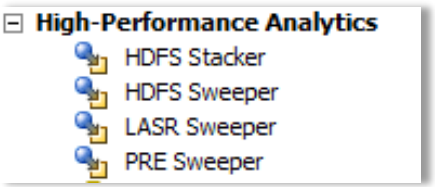


**Figure 1: Transformations Tab from SAS Data Integration Studio**

## REVIEW OF THE AUTOLOADER AND ITS COMPARISON TO THE SWEEPER TRANSFORMS

The autoloader introduced in SAS Visual Analytics 6.2 is powerful. Based on permission or role settings, it enables users to save files (CSV, XLS, XLSX, and SAS data sets) to a specific folder to load data into SAS LASR Analytic Server automatically in a background process. Depending on the location of these files, the autoloader will load, append, or remove the data to or from SAS LASR Analytic Server. A system administrator is required to configure this setup prior to use, and customization is limited. Using a SAS Data Integration Studio

transformation, the sweepers also allow automatic loading from source data locations, but they greatly extend the functionality to handle additional data types and independent use among client groups with no administrator intervention.

The autoloader and the sweeper concept have two main differences:

1. The HDFS Sweeper enables you to store larger data sets in HDFS. This is beneficial because the refresh window for the LASR table can be much shorter when loading from HDFS versus loading from source on large tables. The LASR Sweeper will automatically check to see whether a version of a source table is preloaded into HDFS and load from there when that is the case.
2. The sweeper concept enables you to pull from any source that can be referenced as a SAS library by using a LIBNAME statement. Some examples include Oracle, Microsoft SQL Server, SAS/SHARE®, and MySQL.

**Table 1** shows a full comparison between the two methods.

| | Autoloader in SAS Visual Analytics 7.1 | Sweeper |
|---|---|---|
| **File types that can be loaded** | • Delimited files (CSV)<br>• Spreadsheets (XLS, XLSX)<br>• SAS data sets | • Delimited files (CSV) –<br>• **PRE** Sweeper is required<br>• Spreadsheets (XLS, XLSX –<br>• **PRE** Sweeper is required<br>• SAS data sets<br>• SAS data sets on SAS/SHARE<br>• Oracle tables or views<br>• SQL Server tables or views<br>• Any LIBNAME-compatible source |
| **Scheduler** | • Cron<br>• Windows Scheduler | • Cron<br>• Windows Scheduler<br>• Process Manager Flow Manager (LSF) |
| **Can append to existing LASR tables** | Yes<br>• Must be placed in the Append folder | No<br>• See **HDFS Stacker** later in this paper for an append method. |
| **Can unload** | Yes<br>• Must be manually moved to the Unload folder | Yes<br>• Once a source is removed, it can automatically unload from LASR server (and HDFS) if that option is selected. |
| **Loads to and from HDFS** | No | Yes<br>• **HDFS Sweeper** will load to HDFS based on a size threshold.<br>• **LASR Sweeper** will load from HDFS when a corresponding SASHDAT file exists. |
| **Uses smart loading (only loads when data is changed)** | Yes<br>• Based on file attributes (modified date) | Yes<br>• Based on file attributes (modified date)<br>• When a table's modified data is not available (i.e. from RDBMS sources) it will load only when the record count changes (if that option is set) |
| **Setup required** | • Out of the box there is one autoload folder and one LASR server. To set up separate subfolders and LASR server, a system administrator will need to be heavily involved. | • Metadata-driven based on Source, HDFS, and LASR libraries.<br>• After library metadata is created, the transform will work with any Source, LASR, and HDFS library. |
| **Size limitations** | Yes (~4Gb)<br>• Based on file attributes (file size) | No |
| **Integrates with SAS Data Integration Studio** | No | Yes<br>• Custom transforms and SAS macros can be easily customized for your environment. |

**Table 1: Comparison between Autoloader and Sweeper**

## WHAT IS THIS SWEEPER PROCESS ANYWAY?

Now that you have seen a comparison between the autoloader and the sweeper, you probably want to see the sweeper in more detail. This paper assumes that you have a co-located HDFS environment in conjunction with a SAS LASR Analytic Server environment that is used by SAS Visual Analytics. If that is not the case, then the LASR Sweeper will still work without the use of the HDFS Sweeper. The two main sweeper transforms, LASR and HDFS, are very similar. (These transforms are explained in the following sections.) This consistency in design is intentional to promote ease of use and understanding between the two transformations. **Figure 2** represents how the sweepers work independently or in combination.

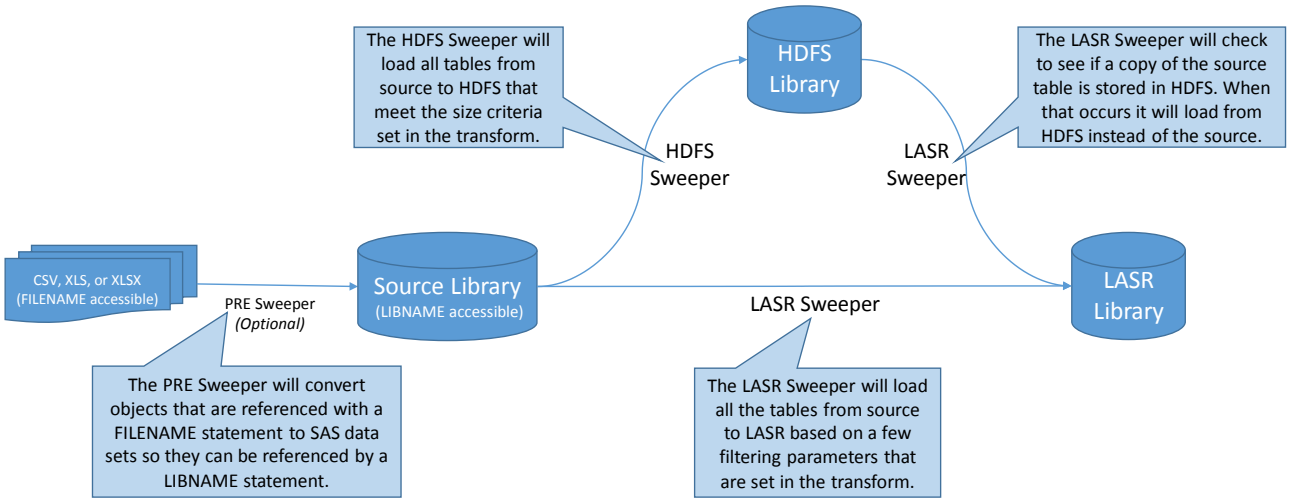


**Figure 2: Diagram of How the LASR Sweeper and the HDFS Sweeper Work Together**

### LASR SWEEPER

Much like the autoloader, the LASR Sweeper transform can load many data sets (tables or views) into SAS LASR Analytic Server automatically. The parameters to this transform give you complete control over the sweeper process. Furthermore, since this custom transform simply calls SAS macros, any experienced SAS developer can customized the transform further to meet your environment's requirements. Not all the default tabs available in the transform are used (i.e. Mappings, Table Options). The key tab that is used to set the parameters is the Options tab. These parameters are described in **Table 2**.

| Tab: Parameter Group | Parameter | Description |
|---|---|---|
| **Options: General Options** | Remove from LASR library | • No (default): All tables will remain in LASR library (and HDFS) even if the corresponding source is removed.<br>• Yes: Tables will be removed from LASR library (and HDFS) when the corresponding source is removed. |

| | | |
|---|---|---|
| | Reload even if record count doesn't change | Note: If PROC CONTENTS can return a "modified date" on the source data (such as Base SAS), then this option is ignored.<br>• No (default): The tables will be reloaded only when the number of observations is different in source and LASR tables.<br>• Yes: The tables will be reloaded even when the number of observations does not change. This is useful when existing records in source tables change. |
| | Extended Options | Extended options for variable names (i.e. spaces in names) can be enabled.<br>• No (default): Strick rules for variable names must be followed<br>• Yes: Variable names may have spaces and other characters in them. Much like Enterprise Guide allows by default. |
| | Data set filter (include) | This will filter the tables to be loaded. Use % as a wildcard or use a delimited list for multiple tables (separated by a blank, \|, :, ;). For example:<br>Contains "VA": %VA%<br>Starts with "VA": VA%<br>TableA and TableB: TableA\|TableB |
| | Data set filter (exclude) | This will filter the tables to be skipped during the load. The syntax is the same as the Data set filter (include) syntax. |
| | Restart LASR server | This option will restart the LASR server prior to loading data. Not commonly used. |
| **Options: Source** | Source library | Select the source library from the drop-down list. Further information can be found in the APPENDIX: Substructure for the transforms. |
| | Source metadata folder | The folder location for the metadata objects of the source tables to be registered. |
| | Remove labels | • No (default): Will keep the labels/descriptions for the variables in the tables.<br>• Yes: Will remove the labels/descriptions. SAS Visual Analytics will use the first 32 characters of a label for the variables in a LASR table. If no labels are available, it will use the variable name. The reason this option was included is because sometimes the first 32 characters of a variable are not unique in a data set, but variable names are required to be unique. |
| | (Optional) Source LIBNAME | If the %SET_LIBRARY macro does not work on the source libref, you can manually enter a LIBNAME statement here. Further information can be found in the APPENDIX: Substructure for the transforms. |
| **Options: LASR** | LASR library | Select the LASR library from the drop-down list. Further information can be found in the APPENDIX: Substructure for the transforms. |
| | LASR metadata folder | The folder location for the metadata objects of the LASR tables to be registered. |
| | Update metadata | • Yes (default): The metadata for the LASR tables will be refreshed each time it is loaded or reloaded.<br>• No: Metadata will not be updated after a reload. This is commonly used if the structure of the tables does not change, since it is more efficient. |
| | Metadata update override: | Note: This is an example of a customization that SAS IT put into place. Updates to LASR metadata can cause a performance hit to the overall environment. By default, the logic in the sweeper (with this customization) is that metadata doesn't update between 8am and 5pm (business hours).<br>• No (default): LASR metadata will not update during business hours.<br>• Yes: LASR metadata will update during business hours. |
| **Options: HDFS (optional)** | HDFS library | Select the HDFS library from the drop-down list. Further information can be found in the APPENDIX: Substructure for the transforms. |
| | HDFS metadata folder | The folder location for the metadata objects of the HDFS tables to be registered. |

**Table 2: LASR Sweeper Parameters**

## HDFS SWEEPER

For an environment that has a co-located HDFS environment, the HDFS Sweeper transform will be beneficial to put into the process flow just before the LASR Sweeper. This transform will load many data sets, tables or views into HDFS automatically. The important parameter to consider here is the size threshold to load into HDFS. This allows you to load all data sets greater than 10 GB (for example) into HDFS but skip the others. Then when the LASR Sweeper runs, it knows to look in the HDFS library before loading from source based on the HDFS parameters found in the LASR Sweeper. If a corresponding data set, by name, is in HDFS, it will lift that HDFS table (SASHDAT) to the LASR library instead of loading from source. The parameters in the HDFS Sweeper transform are described in **Table 3.** You will notice similarities to the LASR Sweeper.

| Tab: Parameter Group | Parameter | Description |
|---|---|---|
| **Options: General Options** | Minimum size to load to HDFS (in Gb) | Tables in the source library will be loaded to HDFS if their size is equal to or larger than the value set in this parameter. The size is calculated by multiplying the number of records by the record length (no compression is taken into account).<br><br>This is a drop-down list, but you can also type over the value to input anything you wish. Setting the parameter to 0 will load all source tables to HDFS. |
| | Remove from HDFS | • No (default): All tables will remain in HDFS even if the corresponding source is removed.<br><br>• Yes: Tables will be removed from HDFS when the corresponding source is removed. |
| | Reload even if record count doesn't change | Note: If PROC CONTENTS can return a "modified date" on the source data (that is, Base SAS), then this option is ignored.<br><br>• No (default): The tables will be reloaded only when the number of observations is different in source and HDFS tables.<br><br>• Yes: The tables will be reloaded even when the number of observations does not change. This is useful when existing records in source tables can be updated. |
| | Extended Options | Extended options for variable names (i.e. spaces in names) can be enabled.<br><br>• No (default): Strick rules for variable names must be followed<br><br>Yes: Variable names may have spaces and other characters in them. Much like Enterprise Guide allows by default. |
| | Data set filter (include) | This will filter the tables to be loaded. Use % as a wildcard or use a delimited list for multiple tables (separated by a blank, \|, :, ;). For example:<br><br>Contains "VA": %VA%<br><br>Starts with "VA": VA%<br><br>TableA and TableB: TableA\|TableB |
| | Data set filter (exclude) | This will filter the tables to be skipped during the load. The syntax is the same as the Data set filter (include) syntax. |
| **Options: Source** | Source library | Select the source library from the drop-down list. Further information can be found in the APPENDIX: Substructure for the transforms. |
| | Source metadata folder | The folder location for the metadata objects of the source tables to be registered. |
| | (Optional) Source LIBNAME | If the %SET_LIBRARY macro doesn't work on the source libref you can manually enter a LIBNAME statement here. Further information can be found in the APPENDIX: Substructure for the transforms. |
| **Options: HDFS** | HDFS library | Select the HDFS library from the drop-down list. Further information can be found in the APPENDIX: Substructure for the transforms. |
| | HDFS metadata folder | The folder location for the metadata objects of the HDFS tables to be registered. |
| | HDFS block size | Select the best block size for HDFS storage.<br><br>Automatic (default): This calculates a block size based on the table size and the number of data nodes on the grid. |

| | | |
|---|---|---|
| | HDFS copies | The number of extra copies of the file blocks to save for replication and node failover. Increasing this number will also increase the amount of HDFS space that is needed to store these tables. |

**Table 3: HDFS Sweeper Parameters**

### PRE SWEEPER

The sweeper data-load method is based on the ability to access and interrogate the data source via a SAS LIBNAME statement. If your data source is a file format, such as CSV, XLS and XLSX, that requires access via a SAS FILENAME statement, use the PRE Sweeper transformation. The PRE Sweeper was developed to convert these types of files to SAS data sets so that a LIBNAME statement can be used. This same methodology is used with the autoloader. For this transform to work correctly, you need to make sure the files are in a format that will work well as a SAS data set. For example, the first row needs to contain the column/variable names. These variable names must be valid SAS names. While the PRE Sweeper can handle many generic types of data imports, there might be nonstandard or complex imports that are currently beyond its scope. But by using the current PRE Sweeper macro as a guide, you can extend the capabilities to create more customizable import processes as needed. From there, the HDFS and LASR Sweepers will complete the data load. The parameters for the PRE Sweeper as shown in **Table 4**.

| Tab: Parameter Group | Parameter | Description |
|---|---|---|
| **Options: General Options** | Source folder | The location of the file(s) to import. |
| | Source encoding | The encoding of the input files. |
| | Target library | Select the target library from the drop-down list. This can be the same physical location as the source folder. Further information can be found in the APPENDIX: Substructure for the transforms. |

**Table 4: PRE Sweeper Parameters**

## HDFS STACKER

The sweeper technique is ideal when you have many different data sets you want to maintain independently. The tables need to stay separate. But what about when you have many data sets with the same structure that you want to eventually put into one LASR table? A great example for this scenario is analyzing daily web logs. After a day's file is complete, it never changes. Therefore, appending to what is already loaded is much more efficient. Since LASR tables can be dropped out of memory and lost forever (i.e. in the event of a LASR server reboot) a storage mechanism that is persistent is critical. The HDFS Stacker method does just that. Here are the basic steps:

1. If necessary, use the
2. PRE Sweeper or create a custom job to process each file. Store these files to a common location in HDFS (that is, in the same path) as a SASHDAT file. This can be done by using the HDFS Sweeper. Name the files with an informative naming convention to let you know what file is for what subset of data. In the weblog example, this would be something like weblog_YYYY_MM_DD, where YYYY, MM, and DD represent the year, month, and day of each file.
3. After you have all your data processed and stored in HDFS, point the HDFS Stacker to that HDFS location/path. It will then load each SASHDAT file to a single LASR table.

**Table 5** describes the parameters of the HDFS Stacker.

| Tab: Parameter Group | Parameter | Description |
|---|---|---|
| **Options: HDFS** | HDFS library | Select the HDFS library from the drop-down list. Further information can be found in the APPENDIX: Substructure for the transforms. |
| | Data set filter (include) | This will filter the tables to be loaded. Use % and ? as wildcard characters. For example: |

| | | |
|---|---|---|
| | | Contains "VA": %VA%<br>Starts with "VA": VA%<br>Patterned name like weblog_2015_04_28: weblog_????_??_?? |
| **Options: LASR** | LASR library | Select the LASR library from the drop-down list. Further information can be found in the APPENDIX: Substructure for the transforms. |
| | LASR metadata folder | The folder location for the metadata objects of the source tables to be registered. |
| | LASR table name | The name of the single table that all the HDFS tables will be stacked into. |

**Table 5: HDFS Stacker Parameters**

## CONCLUSION

SAS Visual Analytics has several powerful integrated applications and processes to load data into the SAS LASR Analytic Server. By using any of the PRE Sweeper, LASR Sweeper, HDFS Sweeper, or HDFS Stacker auto-loading techniques, you can extend that capability. These custom transforms can benefit your users by allowing them to focus on reporting and exploring the data instead of loading the data. Furthermore, the ability to fully automate the data loading and refresh processes allows your developers to focus on more complex challenges.

## APPENDIX

### APPENDIX: SUBSTRUCTURE FOR THE TRANSFORMS

The preceding transforms all require some underlying macros, tables, and jobs to work properly. All of these objects will be packaged together and available on the SAS Global Forum website for download. Here is a brief description of each object:

| Tab: Parameter Group | Description |
|---|---|
| **%SET_LIBRARY** | A macro that queries the metadata repository and generates LIBNAME statements from libraries that are registered in metadata. |
| **Artifacts Library** | A library that contains 1 table and 3 views. The table contains information about libraries that are registered in metadata. The views subset the table into source libraries, LASR libraries, and HDFS libraries. These views are used to populate the drop-down controls in the transforms. |
| **Update Artifacts Job** | A job that updates the Artifacts Library. This job can be scheduled to run at a certain time or can be manually kicked off anytime a library in metadata is changed, added, or removed. |

## CONTACT INFORMATION

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

Jason Shoffner
SAS Institute Inc.
919-531-2110
jason.shoffner@sas.com
www.sas.com