**SAS 3660-2016**

# Automating Load for Multiple Tables in SAS® Visual Analytics

Swetha Vuppalanchi, SAS Institute Inc

## ABSTRACT

The SAS® Visual Analytics autoload feature loads any files placed in the autoload location into the public SAS® LASR™ Analytic Server. But what if we want to autoload the tables into a private SAS LASR Analytic Server and we want to load database tables into SAS Visual Analytics and we also want to reload them every day so that the tables reflect any changes in the database? One option is to create a query in SAS® Visual Data Builder and schedule the table for load into the SAS LASR Analytic Server, but in this case, a query has to be created for each table that has to be loaded. It would be rather efficient if all tables could be loaded simultaneously; that would make the job a lot easier. This paper provides a custom solution for autoloading the tables in one step, so that tables in SAS Visual Analytics don't have to be reloaded manually or multiple queries don't have to be rerun whenever there is a change in the source data. This solution automatically unloads and reloads all the tables listed in a text file every day so that the data available in SAS Visual Analytics is always up-to-date. This paper focuses on autoloading the database tables into the private SAS LASR Analytic server in SAS Visual Analytics hosted in a UNIX environment and uses the SASIOLA engine to perform the load. The process involves four steps: initial load, unload, reload, and schedule unload and reload. This paper provides the scripts associated for each step. The initial load needs to be performed only once. Unload and reload have to be performed periodically to refresh the data source, so these steps have to be scheduled. Once the tables are loaded, a description is also added to the tables detailing the name of the table, time of the load, and user ID performing the load, which is similar to the description added when loading the tables manually into the SAS LASR Analytic Server using SAS Visual Analytics Administrator.

## INTRODUCTION

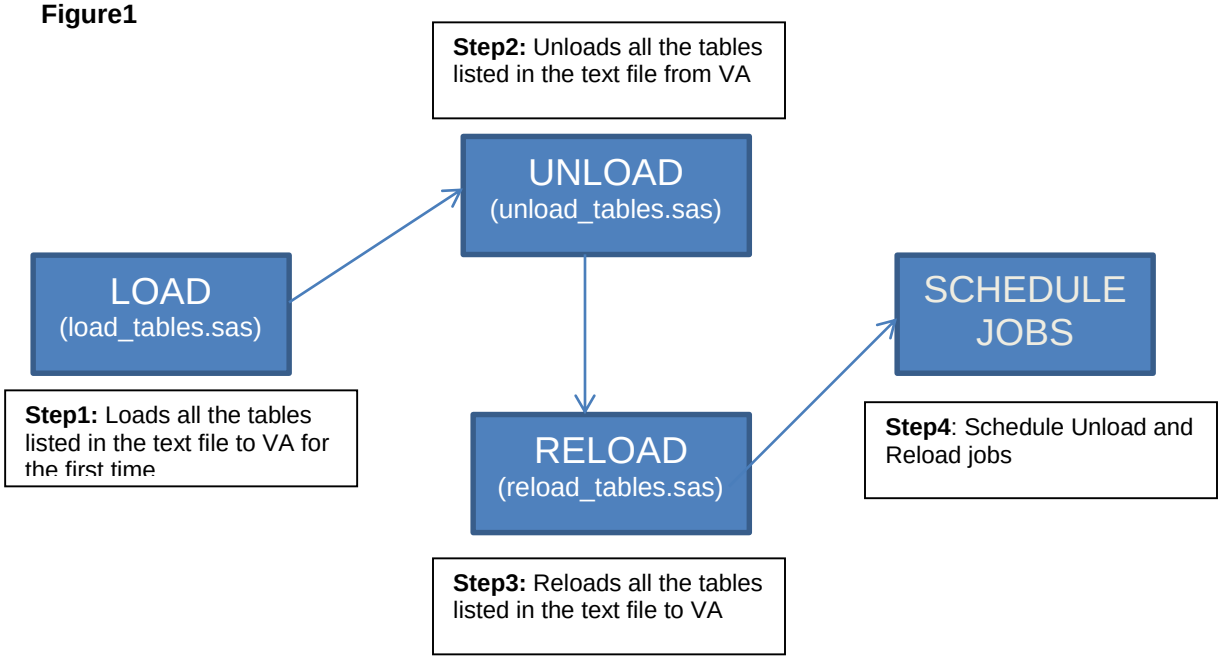
This paper focuses on getting data from database tables to the VA environment in a fast and efficient manner with little to no manual intervention once the process is set up. To get started with the automation process, certain pre-requisites have to be met.

- SAS/ACCESS to ODBC has to be licensed and system DSN for the database should be defined
- An ODBC library has to be defined in SAS metadata pointing to the appropriate DSN
- The database tables that you need to access in VA, have to be registered in the SAS metadata
- VA PRIVATE LASR has to be started
- List all the tables to be loaded into VA in a text file and place it on the server
- The ID performing the load should have read privileges on the SAS metadata tables that will be loaded into the VA environment

## PROCESS

This autoload mechanism involves four basic steps illustrated by Figure 1. SAS scripts have to be created for the steps 1 to 3 (load, unload and reload) process and the associated scripts for these 3 steps are provided in the Appendix section. Step 4 is documented in the Scheduling section. It should be noted that the highlighted sections of the code in the Appendix section should be changed according to the environment. The script for load should to be run once whereas unload and reload have to performed periodically to refresh the data source, so have to be scheduled. This would automate the process of reloading the database tables into VA as per the schedule set by the user.

**Figure1**

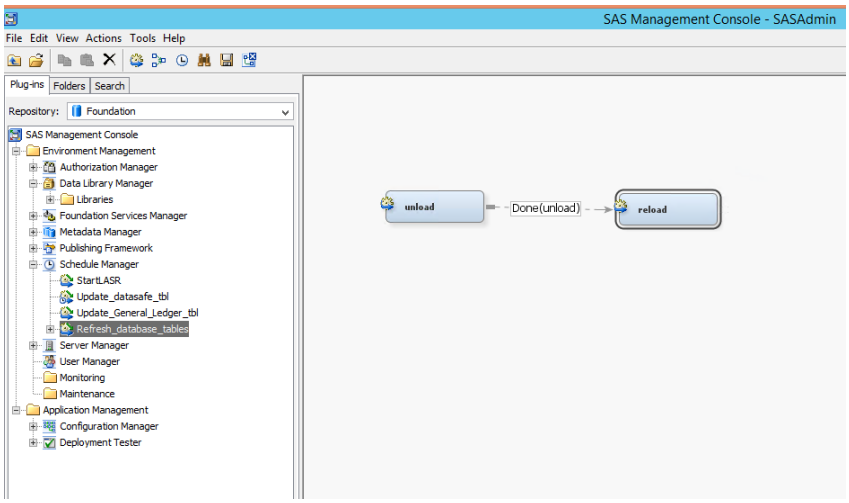



## SCHEDULING

The SAS scripts unload and reload have to be scheduled as they need to run periodically to refresh the data sources in VA. The scheduling of these scripts can be done using SAS Management Console using the following steps.

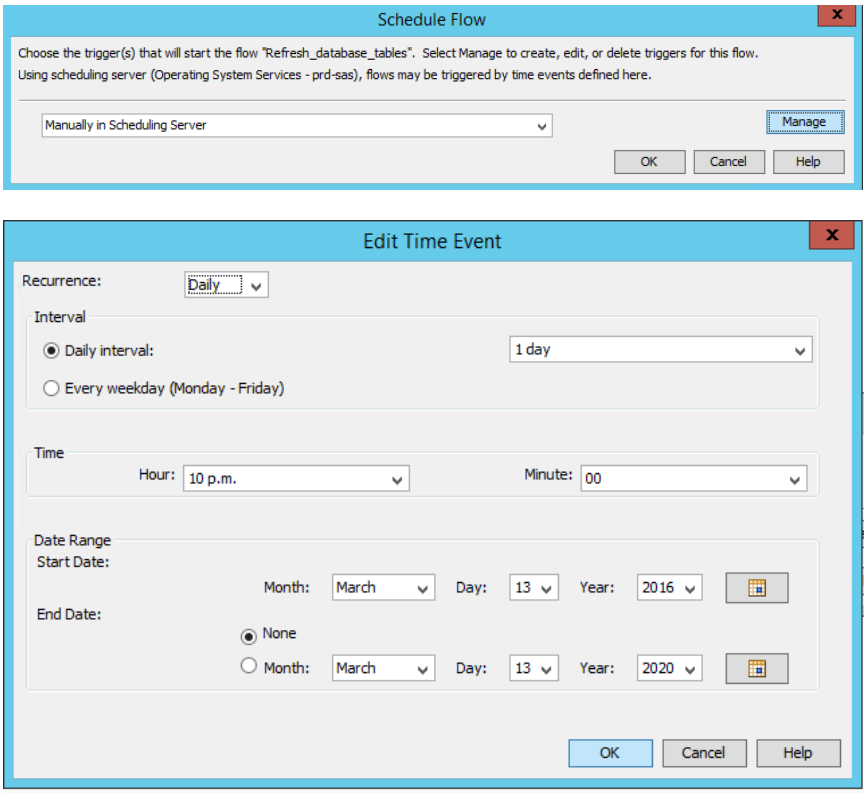
1) Deploy the scripts unload_tables.sas and reload_tables.sas as BASE SAS jobs
2) Create a new flow called 'refresh_database_tables' in Schedule Manager and add the two deployed jobs to the flow
3) Set up dependencies between the scripts so that unload_tables.sas runs first and reload_tables.sas runs only when unload process completes successfully

**Figure 2:**



4) Schedule the flow ‘refresh_database_tables’ and assign the time trigger

**Figure 3:**




## CONCLUSION

This paper provides an approach of creating custom code for reloading multiple tables into LASR at once and automating the process. This method is best suited for database tables as they have large number of tables and are refreshed on a daily basis so it makes sense to have the most updated tables in the VA environment. The method can also be used for SAS tables by just assigning BASE SAS library in the load & reload scripts instead of ODBC library. It is important to remember to update the text file containing the list of tables, if any new tables have to be added or any existing tables have to be retired from the load process. It should also be noted that, when bulk loading these tables, tables of same database type have to be loaded at once when using these scripts which means that all SAS tables can be loaded at once or all database tables can be loaded at once but they cannot be combined together in the text file to load together. Also it is recommended to load only the tables that are necessary into LASR as it would consume lot of memory when a large number of tables are loaded into LASR so caution needs to be taken to stay within the limits of the hardware requirements allocated for the server.

## REFERENCES


- Scheduling Jobs using schedule manager in SAS Management Console

  http://support.sas.com/documentation/cdl/en/scheduleug/68697/HTML/default/viewer.htm#p1msjx6z2e25s0n1vlhi9ves4b2s.htm

## RECOMMENDED READING

- Inbuilt Autoload for PUBLIC LASR

  https://support.selerity.com.au/entries/66036838-How-To-Loading-Data-in-SAS-Visual-Analytics-Autoloading-Tables

- New Implementations of Autoload for PRIVATE LASR

  http://support.sas.com/training/tutorial/gel/gelva03.html

- Creating a data query to schedule a load

  https://support.selerity.com.au/entries/65428153-How-To-Loading-Data-in-SAS-Visual-Analytics-Loading-a-table-from-a-Query

  https://support.selerity.com.au/entries/66559276-How-To-Loading-Data-in-SAS-Visual-Analytics-Scheduling-a-Query

## CONTACT INFORMATION

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

**Swetha Vuppalanchi**
*Sr.Systems Engineer*
SAS Institute Inc.
Cell:732-397-1022
swetha.vuppalanchi@sas.com
www.sas.com

## APPENDIX

```
/*************************************************************************

Name: load_tables.sas
Purpose: Loads all the tables that are listed in the tblist.txt into LASR

*************************************************************************/

/* Metadata connection parameters */

options metaserver="server.example.com"
metaport=8561
metaprotocol="bridge"
metauser="testuser"
metapass="xxxxxxxxxx";

/* Get current date & time to capture it in load information */

data _null_;
dtime=left(put(today(),weekdate30.)||','||put(time(),time10.));
call symputx('dtime',dtime);
run;

/* Get User ID of the user performing the load */

%let userid=%sysget(USER);

%let _ENCODING=UTF-8;
%macro disablelisting;
   /* Disable listing file output */
   filename nulpath dummy;
   proc printto print = nulpath;
   run;
%mend;
%disablelisting;

options VALIDVARNAME=ANY VALIDMEMNAME=EXTEND;
/* Status Checkpoint Macro */
%macro statuscheckpoint(maxokstatus=4, varstocheck=SYSERR SYSLIBRC SYSDBRC );

   %GLOBAL LASTSTEPRC;
   %LET pos=1;
   %let var=notset;
   %let var=%SCAN(&varstocheck.,&pos.);
   %DO %WHILE ("&VAR." ne "");
      /* Retrieve the next return code to check */
          %if (%symexist(&VAR.)) %then %do;
             %let val=&&&VAR..;
             %if  (("&VAL."  ne  "")  and  %eval(&VAL.  >  &maxokstatus.))  %then
%do;
                    %put FAIL = &VAR.=&VAL. / SYSCC=&SYSCC.;
           %let LASTSTEPRC=&VAL.;
                 %end;
          %end;
          %let pos = %eval(&pos.+1);
      %let var=%SCAN(&varstocheck.,&pos.);
```

```
   %END;
%mend;
/* Register Table Macro */
%macro  registertable(  REPOSITORY=Foundation,  REPOSID=,  LIBRARY=,  TABLE=,
FOLDER=, TABLEID=, PREFIX= );

/* Mask special characters */

   %let REPOSITORY=%superq(REPOSITORY);
   %let LIBRARY   =%superq(LIBRARY);
   %let FOLDER    =%superq(FOLDER);
   %let TABLE     =%superq(TABLE);

   %let REPOSARG=%str(REPNAME="&REPOSITORY.");
   %if ("&REPOSID." ne "") %THEN %LET REPOSARG=%str(REPID="&REPOSID.");

   %if ("&TABLEID." ne "") %THEN %LET SELECTOBJ=%str(&TABLEID.);
   %else                         %LET SELECTOBJ=&TABLE.;

   %if ("&FOLDER." ne "") %THEN
      %PUT INFO: Registering &FOLDER./&SELECTOBJ. to &LIBRARY. library.;
   %else
      %PUT INFO: Registering &SELECTOBJ. to &LIBRARY. library.;

   proc metalib;
      omr (
         library="&LIBRARY."
         %str(&REPOSARG.)
          );
      %if ("&TABLEID." eq "") %THEN %DO;
         %if ("&FOLDER." ne "") %THEN %DO;
            folder="&FOLDER.";
         %end;
      %end;
      %if ("&PREFIX." ne "") %THEN %DO;
         prefix="&PREFIX.";
      %end;
      select ("&SELECTOBJ.");
   run;
   quit;

%mend;
%statuscheckpoint;
/* Skip Next Step If We Have a Bad Status Code */
%macro loadintolasr(dsn);
   %GLOBAL LASTSTEPRC;
   %if %symexist(LASTSTEPRC) %then %do;
      %if %eval(&LASTSTEPRC. <= 4) %then %do;

         /* Define ODBC libname statement*/

 LIBNAME odbclib ODBC  DSN=sqldsn user=dbuser password='xxxxxxxx';

         /* SASIOLA load. */
         /* Assign the target library */
LIBNAME VALIBLA SASIOLA  TAG=HPS  PORT=10011 HOST="server.example.com"
SIGNER="http://server.example.com:7981/SASLASRAuthorization" ;
```

```
         data  VALIBLA.&dsn(  label=%unquote(%nrbquote('Loaded  by  load  script
on "&dtime" from "sakila.&dsn" by "&userid"'))) ;
            set odbclib.&dsn;
         run;

      %end;
   %end;
%mend;

/* Create the SAS dataset containing the list of all tables that have to be
loaded */

data dsnnames;
  infile "/home/testuser/vacode/tblist.txt";
  input tbname :$50.;
run;

/* Run the load macro for all the tables listed in the above dataset */

data _null_;
  set dsnnames;
  call execute('%loadintolasr('||tbname||')');
run;

%statuscheckpoint;
/* Skip Next Step If We Have a Bad Status Code */
%macro registerlasrmeta(dsn);
   %GLOBAL LASTSTEPRC;
   %if %symexist(LASTSTEPRC) %then %do;
      %if %eval(&LASTSTEPRC. <= 4) %then %do;

         /* Synchronize table registration */
         %registerTable(
              LIBRARY=%nrstr(/Products/SAS          Visual          Analytics
Administrator/Visual Analytics LASR)
            , REPOSITORY=%nrstr(Foundation)
            , TABLE=&dsn
            , FOLDER=%nrstr(/Shared Data/LASR Tables and Libraries)
            );

      %end;
   %end;
%mend;

/* Run the macro that regsiters the lazr load in metadata for all the tables
listed in the dataset created above */

data _null_;
  set dsnnames;
  call execute('%registerlasrmeta('||tbname||')');
run;
```

```
/*************************************************************************

Name: unload_tables.sas
Purpose: Unloads all the tables that are listed in the tblist.txt from LASR

*************************************************************************/

/* Metadata Connection Parameters */

options metaserver="server.example.com"
metaport=8561
metaprotocol="bridge"
metauser="testuser"
metapass="xxxxxxxx";

/* Get the list of all table names in a SAS dataset that have to be unloaded
*/

data dsnnames;
  infile "/home/testuser/vacode/tblist.txt";
  input tbname :$50.;
run;

%let _ENCODING=UTF-8;
%macro disablelisting;
   /* Disable listing file output */
   filename nulpath dummy;
   proc printto print = nulpath;
   run;
%mend;
%disablelisting;

options VALIDVARNAME=ANY VALIDMEMNAME=EXTEND;
/* Status Checkpoint Macro */
%macro statuscheckpoint(maxokstatus=4, varstocheck=SYSERR SYSLIBRC SYSDBRC );

   %GLOBAL LASTSTEPRC;
   %LET pos=1;
   %let var=notset;
   %let var=%SCAN(&varstocheck.,&pos.);
   %DO %WHILE ("&VAR." ne "");
      /* Retrieve the next return code to check */
          %if (%symexist(&VAR.)) %then %do;
             %let val=&&&VAR..;
             %if (("&VAL." ne "") and %eval(&VAL. > &maxokstatus.)) %then
%do;
                    %put FAIL = &VAR.=&VAL. / SYSCC=&SYSCC.;
           %let LASTSTEPRC=&VAL.;
                 %end;
          %end;
          %let pos = %eval(&pos.+1);
      %let var=%SCAN(&varstocheck.,&pos.);
   %END;
%mend;
%statuscheckpoint;
/* Skip Next Step If We Have a Bad Status Code */
```

```
/* Macro to unload tables from LASR */
%macro unloadfromlasr;
   %GLOBAL LASTSTEPRC;
   %if %symexist(LASTSTEPRC) %then %do;
      %if %eval(&LASTSTEPRC. <= 4) %then %do;

LIBNAME VALIBLA SASIOLA  TAG=HPS  PORT=10011 HOST="server.example.com"
SIGNER="http://server.example.com:7981/SASLASRAuthorization" ;

         /* Remove data from the server */
         %macro deletedsifexists(lib,name);
            %if %sysfunc(exist(&lib..&name.)) %then %do;
                  proc datasets library=&lib. nolist;
                  delete &name.;
            quit;
         %end;
         %mend deletedsifexists;

         /* Run the macro that deletes the data from VALIBLA library for all
the tables listed in the dataset that is created above */
         data _null_;
            set dsnnames;
            call execute('%deletedsifexists(VALIBLA,'||tbname||')');
         run;

      %end;
   %end;
%mend;
      %unloadfromlasr;
```

```
/*************************************************************************

Name: reload_tables.sas
Purpose: Reloads all the tables that are listed in the tblist.txt into LASR

*************************************************************************/

/* Metadata Connection Paramters */

options metaserver="server.example.com"
metaport=8561
metaprotocol="bridge"
metauser="testuser"
metapass="xxxxxxxxxx";

/* Get current date & time to capture it in load information */

data _null_;
dtime=left(put(today(),weekdate30.)||','||put(time(),time10.));
call symputx('dtime', dtime);
run;

/* Get the USER ID of the user performing the load */

%let userid=%sysget(USER);

%let _ENCODING=UTF-8;
%macro disablelisting;
   /* Disable listing file output */
   filename nulpath dummy;
   proc printto print = nulpath;
   run;
%mend;
%disablelisting;

options VALIDVARNAME=ANY VALIDMEMNAME=EXTEND;
/* Status Checkpoint Macro */
%macro statuscheckpoint(maxokstatus=4, varstocheck=SYSERR SYSLIBRC SYSDBRC );

   %GLOBAL LASTSTEPRC;
   %LET pos=1;
   %let var=notset;
   %let var=%SCAN(&varstocheck.,&pos.);
   %DO %WHILE ("&VAR." ne "");
      /* Retrieve the next return code to check */
          %if (%symexist(&VAR.)) %then %do;
             %let val=&&&VAR..;
             %if (("&VAL." ne "") and %eval(&VAL. > &maxokstatus.)) %then
%do;
                    %put FAIL = &VAR.=&VAL. / SYSCC=&SYSCC.;
           %let LASTSTEPRC=&VAL.;
                 %end;
          %end;
          %let pos = %eval(&pos.+1);
      %let var=%SCAN(&varstocheck.,&pos.);
   %END;
%mend;
```

```
/* Register Table Macro */
%macro registertable( REPOSITORY=Foundation, REPOSID=, LIBRARY=, TABLE=,
FOLDER=, TABLEID=, PREFIX= );

/* Mask special characters */

   %let REPOSITORY=%superq(REPOSITORY);
   %let LIBRARY   =%superq(LIBRARY);
   %let FOLDER    =%superq(FOLDER);
   %let TABLE     =%superq(TABLE);

   %let REPOSARG=%str(REPNAME="&REPOSITORY.");
   %if ("&REPOSID." ne "") %THEN %LET REPOSARG=%str(REPID="&REPOSID.");

   %if ("&TABLEID." ne "") %THEN %LET SELECTOBJ=%str(&TABLEID.);
   %else                        %LET SELECTOBJ=&TABLE.;

   %if ("&FOLDER." ne "") %THEN
      %PUT INFO: Registering &FOLDER./&SELECTOBJ. to &LIBRARY. library.;
   %else
      %PUT INFO: Registering &SELECTOBJ. to &LIBRARY. library.;

   proc metalib;
      omr (
         library="&LIBRARY."
         %str(&REPOSARG.)
          );
      %if ("&TABLEID." eq "") %THEN %DO;
         %if ("&FOLDER." ne "") %THEN %DO;
            folder="&FOLDER.";
         %end;
      %end;
      %if ("&PREFIX." ne "") %THEN %DO;
         prefix="&PREFIX.";
      %end;
      select ("&SELECTOBJ.");
   run;
   quit;

%mend;
%statuscheckpoint;

/* Skip Next Step If We Have a Bad Status Code */
%macro reloadintolasr(dsn);
   %GLOBAL LASTSTEPRC;
   %if %symexist(LASTSTEPRC) %then %do;
      %if %eval(&LASTSTEPRC. <= 4) %then %do;

         /* Load into server */

LIBNAME odbclib ODBC DSN=sqldsn user=dbuser password='xxxxxxxxxxxxx';

         /* SASIOLA load. */
         /* Assign the target library */
LIBNAME VALIBLA SASIOLA  TAG=HPS  PORT=10011 HOST="server.example.com"
SIGNER="http://server.example.com:7981/SASLASRAuthorization" ;
```

```
         data VALIBLA.&dsn( label=%unquote(%nrbquote('Loaded by reload script
on "&dtime" from "sakila.&dsn" by "&userid"')));
            set odbclib.&dsn;
         run;

      %end;
   %end;
%mend;

/* Get the list of all table names in a SAS dataset that have to be reloaded
*/
data dsnnames;
  infile "/home/testuser/vacode/tblist.txt";
  input tbname :$50.;
run;

/* Run the reload macro for all the tables listed in the dataset created
above */

data _null_;
  set dsnnames;
  call execute('%reloadintolasr('||tbname||')');
run;
%statuscheckpoint;
/* Skip Next Step If We Have a Bad Status Code */
%macro registerlasrmeta(dsn);
   %GLOBAL LASTSTEPRC;
   %if %symexist(LASTSTEPRC) %then %do;
      %if %eval(&LASTSTEPRC. <= 4) %then %do;

         /* Synchronize table registration */
         %registerTable(
              LIBRARY=%nrstr(/Products/SAS Visual Analytics
Administrator/Visual Analytics LASR)
            , REPOSITORY=%nrstr(Foundation)
            , TABLE=&dsn
            , FOLDER=%nrstr(/Shared Data/LASR Tables and Libraries)
            );

      %end;
   %end;
%mend;

/* Run the macro that registers the LASR load in metadata for all the tables
listed in the dataset created above */
data _null_;
  set dsnnames;
  call execute('%registerlasrmeta('||tbname||')');
run;
```