Paper AD-172

A Methodology for Truly Dynamic Prompting in SAS® Stored Processes

Haikuo Bian, Regions Bank; Carlos Jimenez, Regions Bank;

David Maddox, Regions Bank

ABSTRACT

Dynamic prompts in SAS® stored processes may be developed by selecting the “dynamic list” option during prompt construction. The list is usually sourced from a SAS dataset that is pre-defined in the metadata. However, the process of refreshing the dataset is usually independent of the stored process and must be included somewhere in the application. Using SAS views as a source for dynamic prompts will insure that the list will be truly dynamic. This paper illustrates the process with a cascading prompt example.

BACKGROUND

SAS Views

SAS Views were first introduced in SAS Version 6 in the 1990s. A SAS View is compiled SAS code that when executed produces a virtual dataset. There are three types of SAS Views - Data step views, PROC SQL views and SAS/ACCESS views. However, since the type of SAS View does not affect the basic premise of this paper, they are not differentiated and will be referred to as ‘SAS Views’ in general. One of the many benefits of using SAS Views is that it is provides access to the most current data.

SAS Stored Process (STP)

According to the official definition, a stored process (STP) is a SAS program that is stored on a server and defined in metadata, and which can be executed as requested by client applications. To many STP users, it is just SAS code that can be invoked by many SAS internal applications (SAS Enterprise Guide®, BASE SAS etc.) and external (Java, .NET) applications if being deployed as a web service. It can be implemented both as a data source or data target. It can also act both as an API, or a functional processor.

A REAL LIFE SCENARIO

When defining dynamic prompts for an STP, the Data Source for the Prompt (DSP) has to be predefined in the metadata server. When a DSP needs to be updated, a separate data refresh process is usually needed. One option is to set up a scheduled job that runs routinely. However, it may be troublesome when the updating of the DSP is not based on a fixed schedule. Another option is to chain two STPs. The first one would update the DSP, and end users will likely need to be involved. As one of the Business Intelligence (BI) components, STP’s could be used by business users who are not necessarily savvy in either SAS programming or understanding the mechanisms of STP’s. Therefore, a good STP is friendly to end users through both providing introductive interactions (prompts) and eliminating manual housekeeping procedures as much as possible. How do we accomplish the task of updating a DSP without end users having to do it explicitly? SAS views can come to rescue. The following steps illustrate the whole process of building an STP with dynamic cascading prompts using data from the SASHELP tables.

Preparing the sample dataset /views.

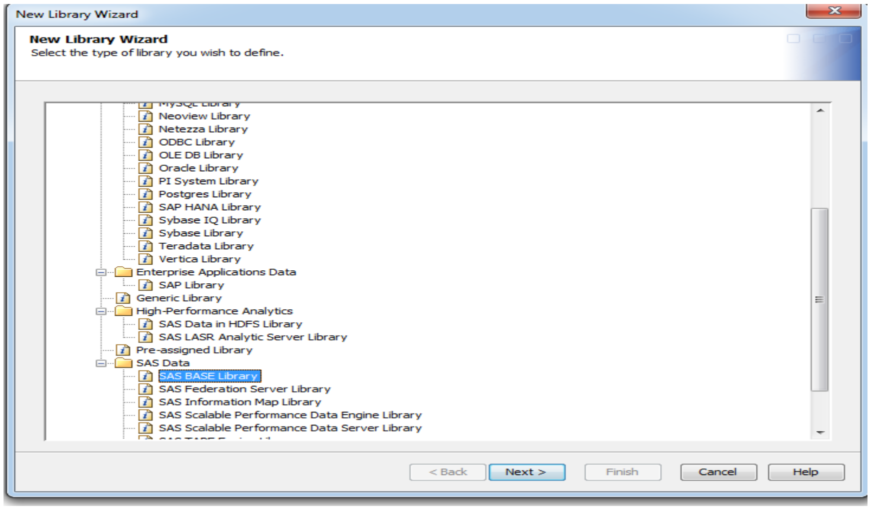
```
/*One will need a permanent library to host sample tables/views*/

LIBNAME SESUG '/frl/Users/SESUG/';

data SESUG.stp_1;

set sashelp.prdsal2;

/*The commented out portion can be switched back on to change the data source*/

/*if upcase(country) in ('U.S.A.' 'CANADA');  */

if upcase(country) in ('MEXICO' 'CANADA');  /*This can be also commented out to change
the data source*/

run;

 proc sql;

create VIEW SESUG.COUNTRY_VIEW as
```

```
select distinct COUNTRY from SESUG.stp_1;
quit;

proc sql;
create VIEW SESUG.STATE_VIEW as
select distinct COUNTRY, STATE from SESUG.stp_1;
quit;
```

Step 1. Define the metadata library

Define (or predefine) a SAS metadata library using any SAS applications that are capable of writing/modifying metadata, such as SAS Enterprise Guide, SAS Management Console (SMC), SAS Data Integration Studio or even BASE SAS. The following example is based on the SMC.

Open SMC >> Plug-ins >> Data Library Manager >> Libraries (Right click) >> New Libraries…

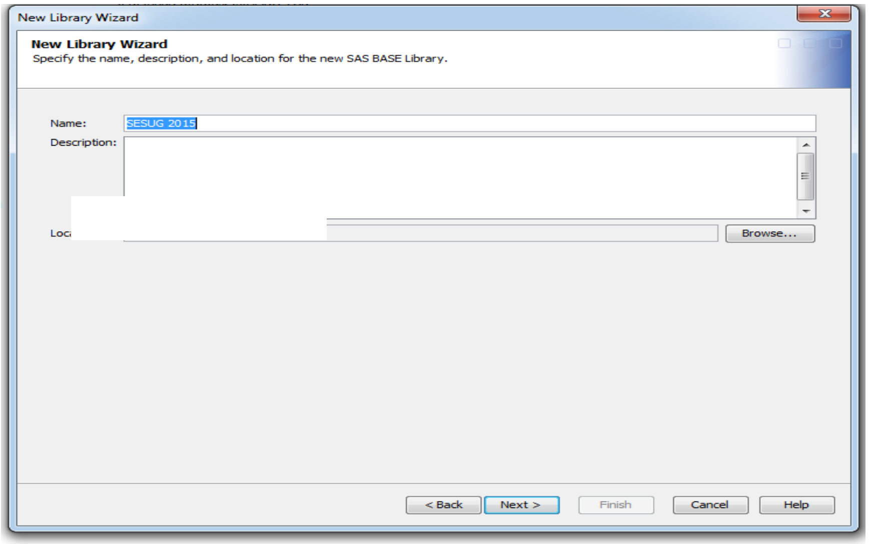


New Library Wizard

Select the type of library you wish to define.

Display 1. Define the metadata library-1

The example is for SAS BASE Library.



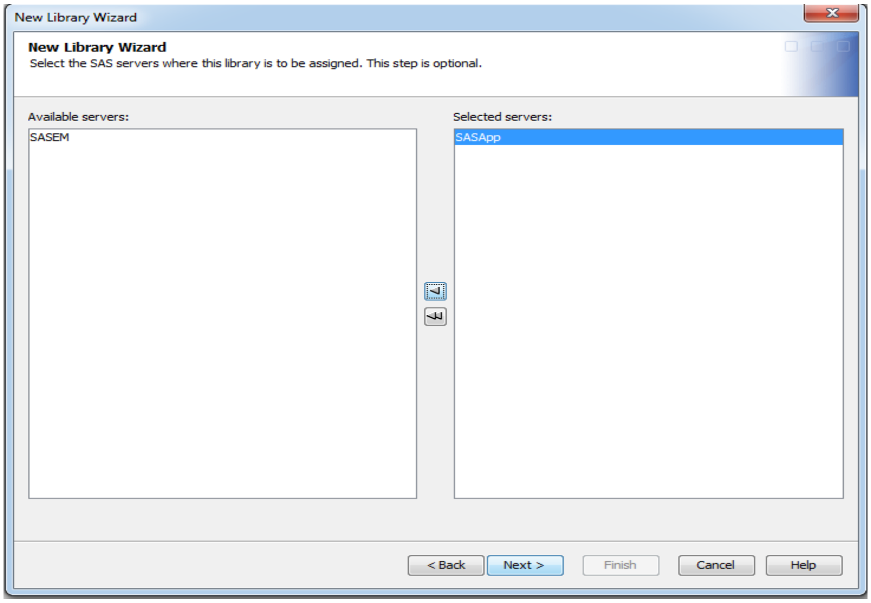
New Library Wizard

Specify the name, description, and location for the new SAS BASE Library.

Name: SESUG 2015

Display 2. Define the metadata library-2

This metadata library name ‘SESUG 2015’ is not a Libref, and it does not have to comply with SAS Libref naming restrictions.


New Library Wizard

New Library Wizard

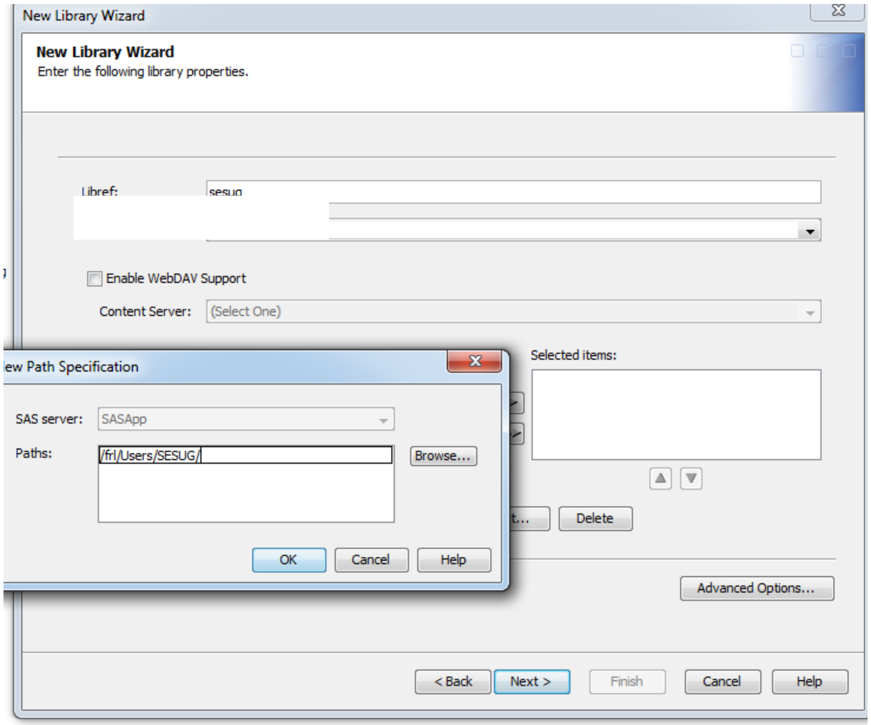
Select the SAS servers where this library is to be assigned. This step is optional.

Available servers: SASEM

Selected servers: SASApp

Display 3. Define the metadata library-3

This is to assign an application server for this library.


New Library Wizard

New Library Wizard

Enter the following library properties.

Libref: sesug

☐ Enable WebDAV Support

Content Server: (Select One)

Selected items:

ew Path Specification

SAS server: SASApp

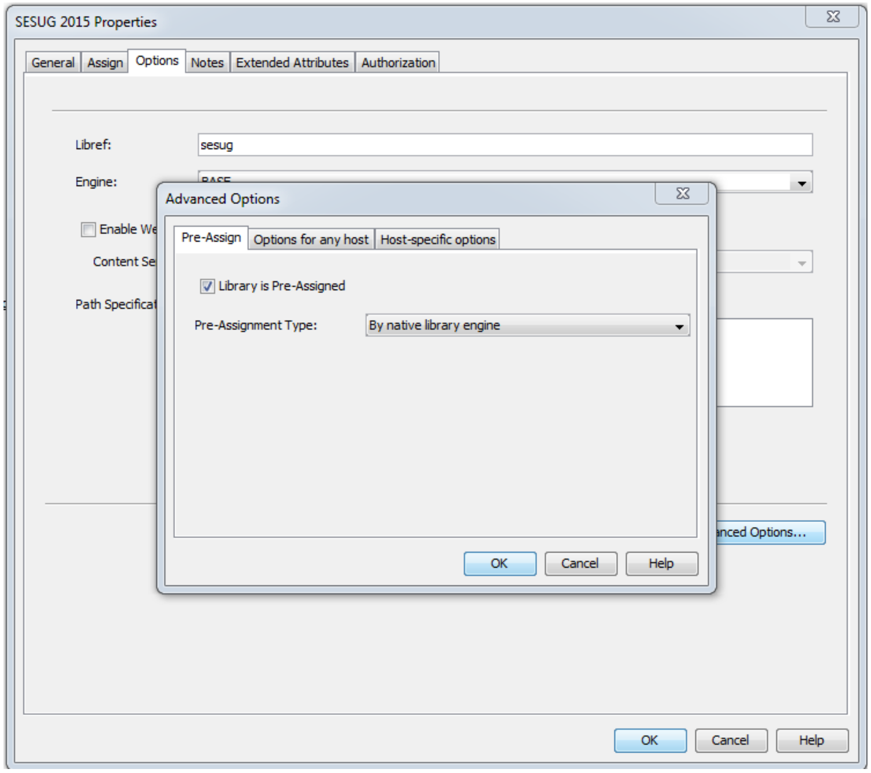
Paths: /frl/Users/SESUG/

Display 4. Define the metadata library-4

Input ‘sesug’ as the SAS Libref. This is the libname as used in Base SAS, and it needs to follow SAS naming conventions. The length must eight (8) characters or less. Select New… and input the physical address of the library. Click ‘OK’, ‘Next’, ‘Finish’

Step 2. Pre-Assign the metadata library

Right Click the library name ‘SESUG 2015’ and choose Properties >> Options >> Advanced Options >> Check ‘Library is Pre-Assigned’.



Libref: sesug

Engine: BASE

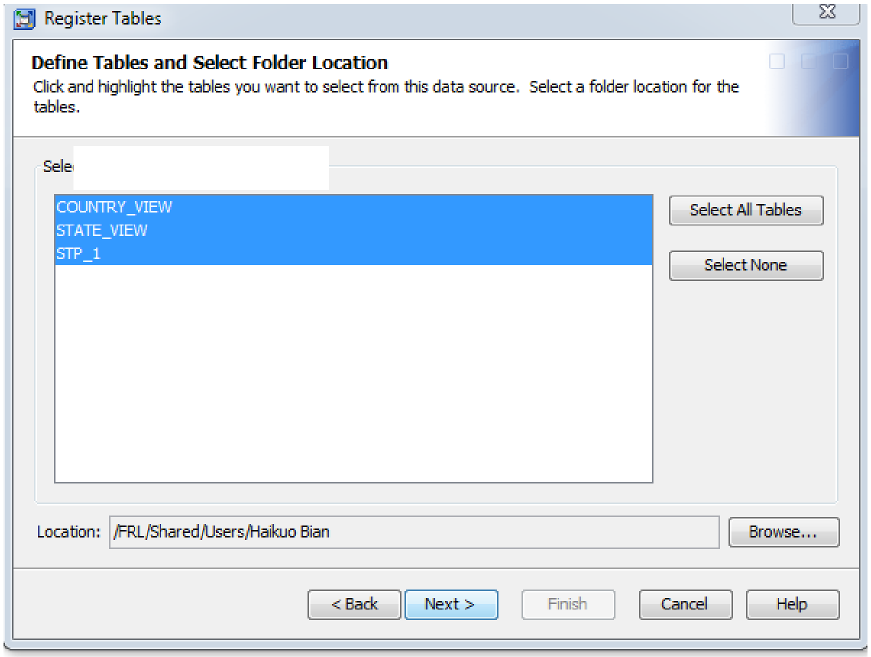
☑ Library is Pre-Assigned

Pre-Assignment Type: By native library engine

Display 5. Pre-Assign the metadata library

Step 3. Registering Tables/Views

Right Click the library name 'SESUG 2015' and choose Register Tables… >> (check all of the information) Next



Define Tables and Select Folder Location

Click and highlight the tables you want to select from this data source. Select a folder location for the tables.

COUNTRY_VIEW
STATE_VIEW
STP_1

Location: /FRL/Shared/Users/Haikuo Bian

Display 6. Registering Tables/Views

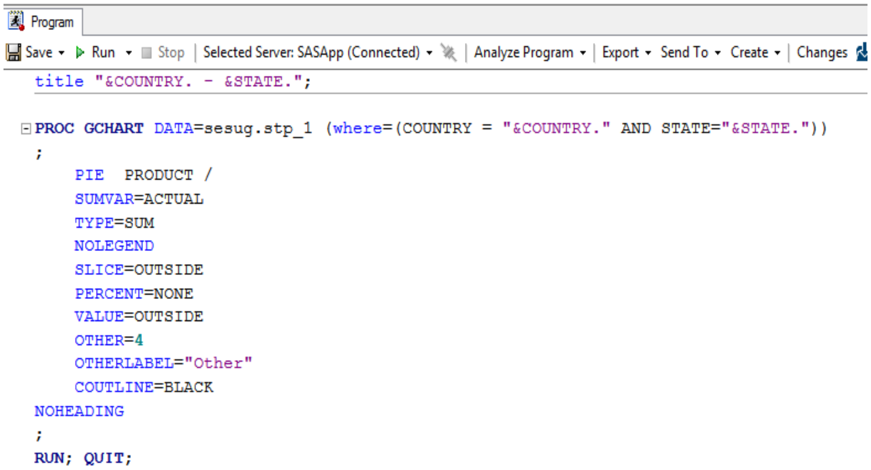
>> Select All Tables >> (using Browse… to select the folder location for metadata objects. This is the metadata folder, NOT the physical path) Next >> Finish

Step 4. Building an STP.

Since this paper does not include a STP tutorial, only a brief explanation is provided in general with the exception of dynamic prompting.

This is the SAS code for the STP, and the output is a simple pie chart. As you can see, two possible prompts are COUNTRY and STATE.

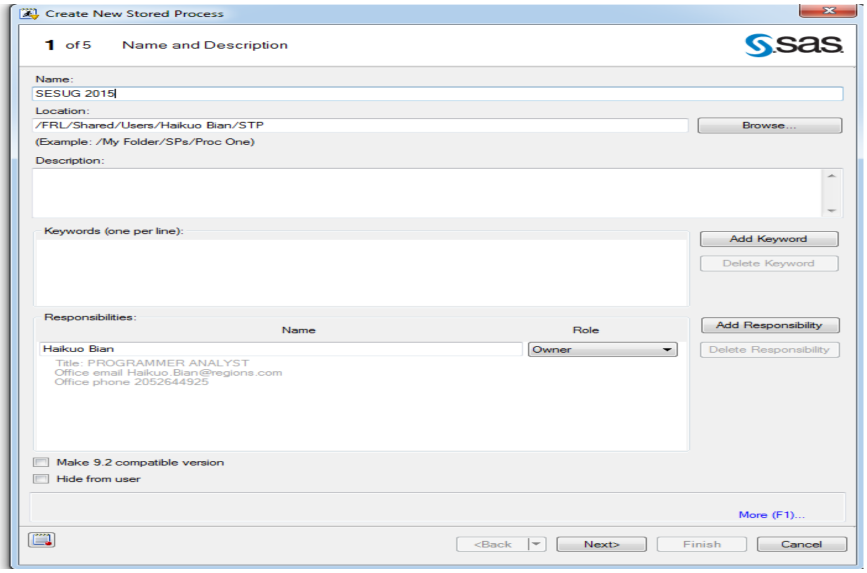
```
title "&COUNTRY. - &STATE.";

PROC GCHART DATA=SESUG.stp_1 (where=(COUNTRY = "&COUNTRY." AND STATE="&STATE."))

;

        PIE      PRODUCT /

        SUMVAR=ACTUAL

        TYPE=SUM

        NOLEGEND

        SLICE=OUTSIDE

        PERCENT=NONE

        VALUE=OUTSIDE

        OTHER=4

        OTHERLABEL="Other"

        COUTLINE=BLACK

NOHEADING

;

RUN; QUIT;
```



```
title "&COUNTRY. - &STATE.";

PROC GCHART DATA=sesug.stp_1 (where=(COUNTRY = "&COUNTRY." AND STATE="&STATE."))
;
    PIE  PRODUCT /
    SUMVAR=ACTUAL
    TYPE=SUM
    NOLEGEND
    SLICE=OUTSIDE
    PERCENT=NONE
    VALUE=OUTSIDE
    OTHER=4
    OTHERLABEL="Other"
    COUTLINE=BLACK
NOHEADING
;
RUN; QUIT;
```

Display 7. Building a STP-1

In your Program Editor, Create >> Stored Process



Name: SESUG 2015

Location: /FRL/Shared/Users/Haikuo Bian/STP

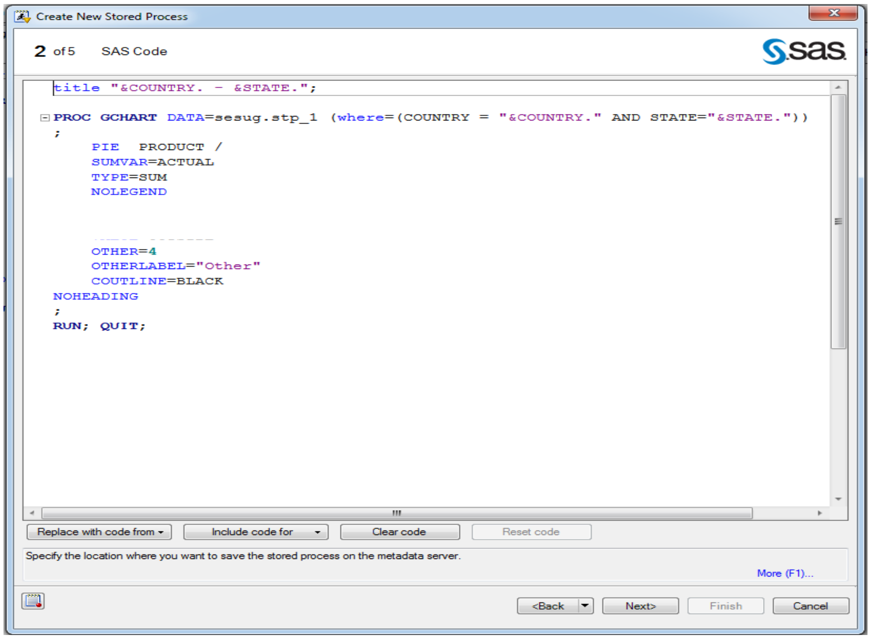
Name	Role
Haikuo Bian Title: PROGRAMMER ANALYST Office email Haikuo.Bian@regions.com Office phone 2052644925	Owner

☐ Make 9.2 compatible version

☐ Hide from user

Display 8. Building a STP-2

The name of the STP and where to store it, >> Next

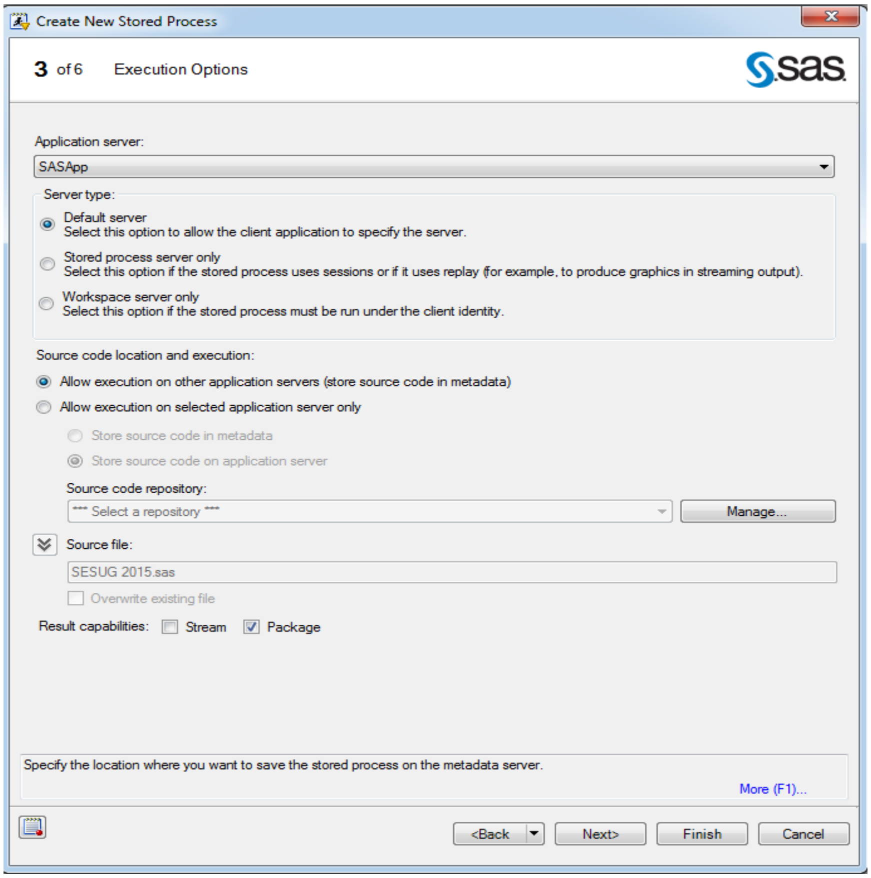


```
title "&COUNTRY. - &STATE.";

PROC GCHART DATA=sesug.stp_1 (where=(COUNTRY = "&COUNTRY." AND STATE="&STATE."))
;
     PIE  PRODUCT /
     SUMVAR=ACTUAL
     TYPE=SUM
     NOLEGEND


     OTHER=4
     OTHERLABEL="Other"
     COUTLINE=BLACK
NOHEADING
;
RUN; QUIT;
```

Specify the location where you want to save the stored process on the metadata server.

Display 9. Building a STP-3

Display 9 illustrates the SAS code portion of the STP, which can be edited or imported code here. >> Next



Create New Stored Process

3 of 6 Execution Options

Application server:

SASApp

Server type:

● Default server
Select this option to allow the client application to specify the server.

○ Stored process server only
Select this option if the stored process uses sessions or if it uses replay (for example, to produce graphics in streaming output).

○ Workspace server only
Select this option if the stored process must be run under the client identity.

Source code location and execution:

● Allow execution on other application servers (store source code in metadata)

○ Allow execution on selected application server only

○ Store source code in metadata

● Store source code on application server

Source code repository:

*** Select a repository *** Manage...

Source file:

SESUG 2015.sas

☐ Overwrite existing file

Result capabilities: ☐ Stream ☑ Package

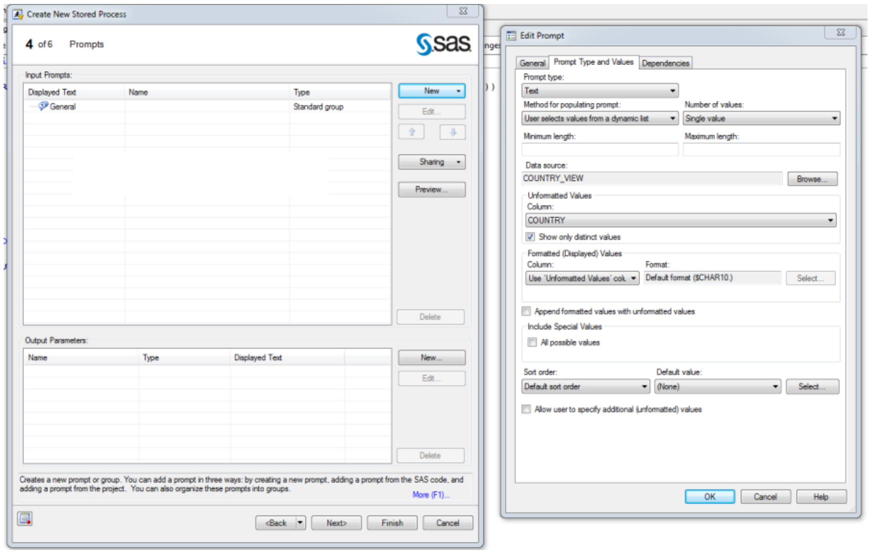
Specify the location where you want to save the stored process on the metadata server.

More (F1)...

Display 10. Building a STP-4

The definition of the STP execution options and where the source code is stored are illustrated in Display 10.

>> Next >> (to the Prompts window) New >> Prompts from SAS code >> COUNTRY >> Prompt Type and Values



Create New Stored Process

4 of 6 Prompts

Input Prompts:

Displayed Text	Name	Type
General		Standard group

Output Parameters:

Name	Type	Displayed Text

Creates a new prompt or group. You can add a prompt in three ways: by creating a new prompt, adding a prompt from the SAS code, and adding a prompt from the project. You can also organize these prompts into groups.

More (F1)...

Edit Prompt

General | Prompt Type and Values | Dependencies

Prompt type: Text

Method for populating prompt: User selects values from a dynamic list

Number of values: Single value

Minimum length:

Maximum length:

Data source: COUNTRY_VIEW

Unformatted Values

Column: COUNTRY

☑ Show only distinct values

Formatted (Displayed) Values

Column: Use 'Unformatted Values' col.

Format: Default format ($CHAR10.)

☐ Append formatted values with unformatted values

Include Special Values

☐ All possible values

Sort order: Default sort order

Default value: (None)

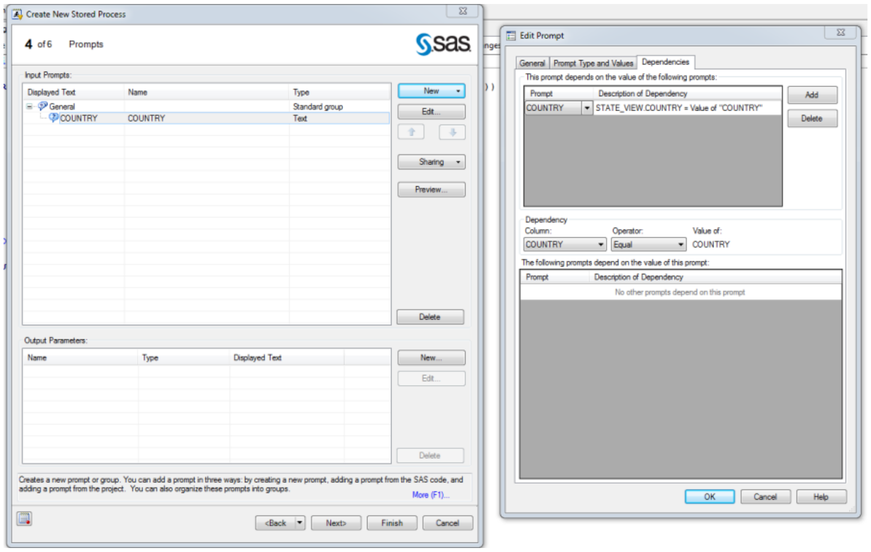
☐ Allow user to specify additional (unformatted) values

Display 11. Building a STP-5

>>Method for populating prompt >> User selects values from a dynamic list >> Browse… to choose the COUNTRY_VIEW from the metadata folder where this metadata object is stored. >> Column >> COUNTRY

The process is repeated to create a new prompt STATE. Prompt STATE is a cascade prompt, whose value will depend on the value of prompt COUNTRY.

New >> Prompts from SAS code >> COUNTRY >> Prompt Type and Values >>Method for populating prompt >> User selects values from a dynamic list >> Browse… to choose the STATE_VIEW from the metadata folder where this metadata object is stored. >> Column >> STATE >> Dependencies >> Add >> Prompt COUNTRY >> Dependency Column: COUNTRY >> Operator: Equal >>OK



Input Prompts:

Displayed Text	Name	Type
General		Standard group
COUNTRY	COUNTRY	Text

Output Parameters:

Name	Type	Displayed Text

Creates a new prompt or group. You can add a prompt in three ways: by creating a new prompt, adding a prompt from the SAS code, and adding a prompt from the project. You can also organize these prompts into groups.

This prompt depends on the value of the following prompts:

Prompt	Description of Dependency
COUNTRY	STATE_VIEW.COUNTRY = Value of "COUNTRY"

Dependency

Column: COUNTRY

Operator: Equal

Value of: COUNTRY

The following prompts depend on the value of this prompt:

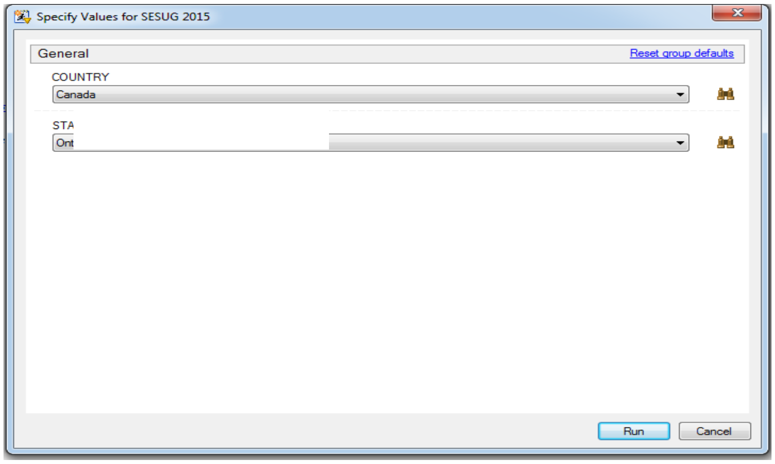
Prompt	Description of Dependency
No other prompts depend on this prompt	

Display 12. Building a STP-6

>> Next >> Next >> Finish

Step 5. Testing the STP.

Now when the STP ‘SESUG 2015’ is executed, the following window will pop up:



COUNTRY: Canada

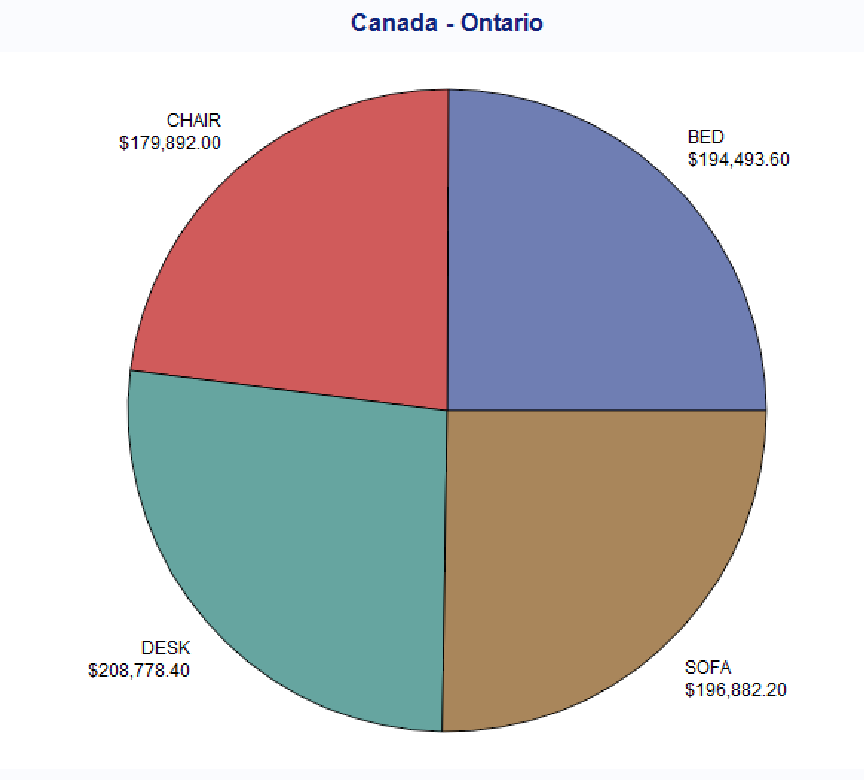
STA: Ont

Display 13. Testing STP-1

Under COUNTRY, only Canada and Mexico can be selected, which is the result of the following SAS code when we prepare the source table:

```
data SESUG.stp_1;

set sashelp.prdsal2;

/*The commented out portion can be switched back on to change the data source*/

/*if upcase(country) in ('U.S.A.' 'CANADA');  */

if up                    ('MEXICO' 'CANADA');  /*This can be also commented out to change
the data source*/

run;
```

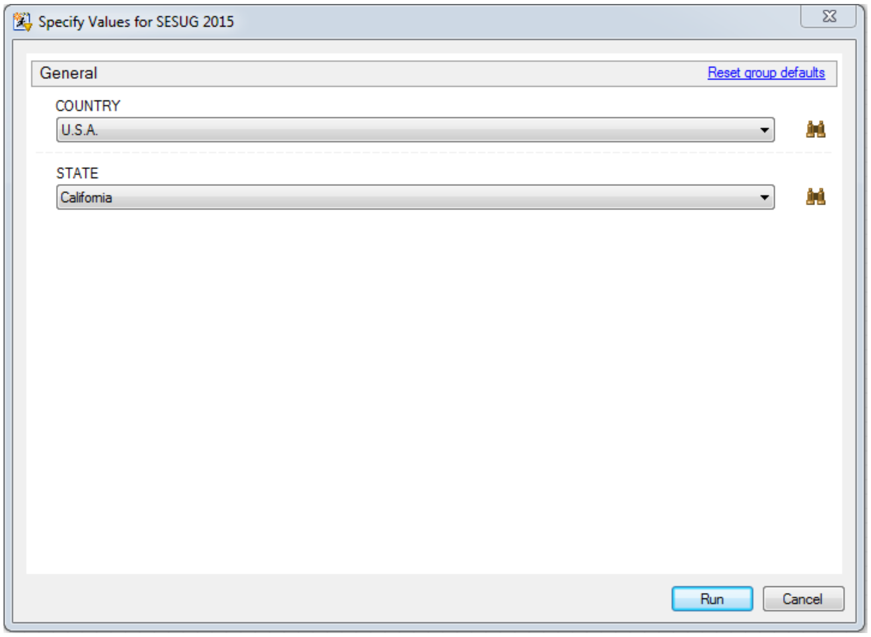
After COUNTRY is set to ‘Canada’, only provinces of ‘Canada’ can be select under STATE, select ‘Ontario’. The following is the final outcome for this particular selection:



Display 14. Testing STP-2

Now to test if the VIEW works, modify the SAS code to update the source data. In the example, all of the filters have been commented out.

```
data SESUG.stp_1;

set sashelp.prdsal2;

/*The                  tion can be switched back on to change the data source*/

/*if upcase(country) in ('U.S.A.' 'CANADA');  */

/*if upcase(country) in ('MEXICO' 'CANADA');*/  /*This can be also commented out to
change the data source*/

run;
```



Specify Values for SESUG 2015

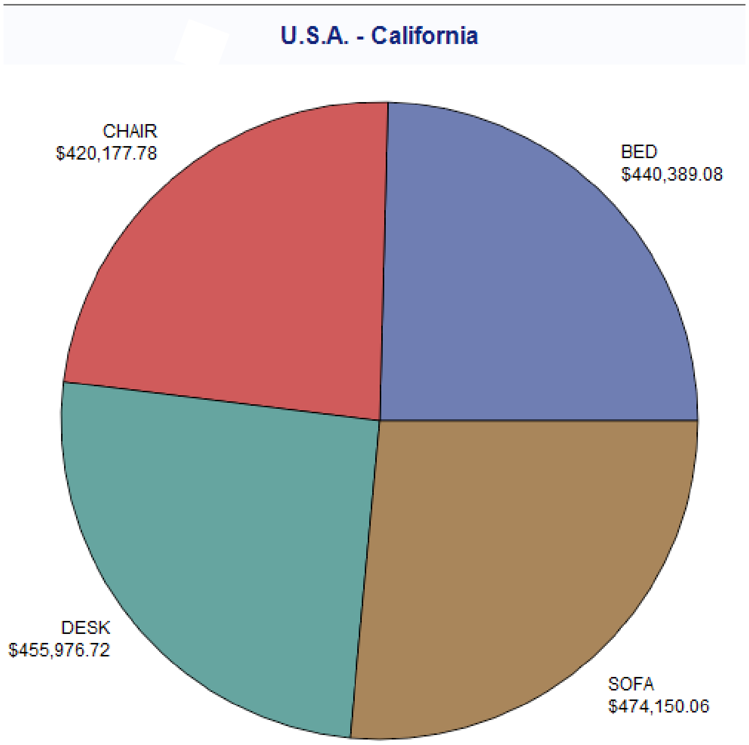
General

COUNTRY: U.S.A.

STATE: California

Display 15. Testing STP-3

Had we used tables instead of SAS Views as the DSP, we will still get the same prompts as in Display 13.We are able to see ‘U.S.A’ under COUNTRY, after that, many US states under STATE,



Display 16. Testing STP-4

The outcome is exactly what we are expecting: U.S.A – California.

DISCUSSION

There is something of a generation gap between users of SAS Views and users of STP’s in both literal and metaphoric terms. There were more than ten years between the introduction of these two tools, and the user populations are likely to have distinct features as well. SAS View users tend to be skilled old school programmers/analysts focusing on data management. They are more likely to come from the era where hardware resources (CPU/Storage) were scarce. STP developers are likely to be new generation programmers/analysts who have more resources at their disposal, and reporting is usually emphasized. Now by using these tools together, we know that the there is an advantage that can be realized.

Some items worth addressing of this application are presented in the form of Q & A:

Q1. Does the library for source data and SAS Views have to be pre-assigned?

A1. Yes. Prompts, in BASE SAS terms, are macro variables. Long before SAS code is compiled, the macro variables must have values which come from SAS Views. Those values in turn come from source data. Therefore it is not possible to wait for defining this particular library during the execution of STP. We are aware that doing so will result in an overhead for application server, but we also believe this can be one of the reasons that pre-assigned libraries exist along with non-pre-assigned libraries. One feature of pre-assigned library is that the data source do not have to be registered by metadata server in this context, as a result, it will facilitate easier maintenance.

Q2. Do the SAS Views have to be registered?

A2. Yes. Registering SAS Views will generate metadata objects upon which dynamic prompts are defined. The metadata libraries that hosts SAS Views need to be pre-assigned, and they can be different from the libraries that host source data in both physical location and metadata folder location.

Q3. Are there any other options besides SAS Views for non-programming oriented environment?

A3. Yes. If you have licensed SAS Information Map, same functionality can also be achieved without coding.

CONCLUSION

Using SAS views as the data source of dynamic prompts is capable of making the prompts truly ‘dynamic’.

REFERENCES

SAS® 9.4 Stored Processes Developer's Guide Third Edition, SAS Institute Inc., Cary, NC, USA

Robert Ray, Save Time Today Using SAS® Views
Proceedings of the Twenty-Seventh Annual SAS Users Group International Meeting, Seattle, WA

James C. Stokes, SAS® Data Views Simply Stated
Proceedings of the Twenty-Ninth Annual SAS Users Group International Meeting, Montreal, Canada

Joe Flynn, A SAS® Programmer's Guide to Stored Processes
Proceedings of the 2011 SAS Global Forum, Las Vegas, NV

CONTACT INFORMATION

Haikuo Bian
Risk Compliance Department
Regions Bank
1900 5th Avenue North, 9th floor Birmingham, AL 35203
Phone: 205-264-4925
haikuo.bian@regions.com

Garland D. (David) Maddox, Jr
Business Loan Center – Performance Analytics and Reporting
Regions Bank
250 Riverchase Parkway East, Birmingham, AL 35244
Phone: 205-560-6339
Garland.MaddoxJr@Regions.com

Carlos Jimenez
Risk Compliance Department
Regions Bank
1900 5th Avenue North, 9th floor Birmingham, AL 35203
Phone: 205-264-5416
Carlos.m.Jimenez@regions.com