Paper BB-11

Creating Stored Processes with Dynamic, Cascading Prompts

Harry Droogendyk, Stratia Consulting Inc., Lynden, ON

ABSTRACT

SAS® Stored Processes are an important component of the SAS Enterprise Business Intelligence suite. Server-based SAS Stored Processes (STP) allow the encapsulation of common business rules to generate consistent results, the elusive "single version of the truth". STPs can be created a number ways, including from Enterprise Guide projects, and invoked from various interfaces including Excel, the STP web interface, Information Maps and the Portal. User specified parameters increase the flexibility and utility of this useful facility. In the latest release of SAS, STP parameters can be both dynamic and cascading, providing the ability to generate relational, data-driven prompts to customize STP results.

INTRODUCTION

SAS Stored Processes (STP) are strategic for a number of reasons. They are hosted on a central server, registered in the metadata, secure, encapsulate business rules to provide a consistent single version of the truth and are flexible in that they may be invoked in multiple ways through multiple clients.

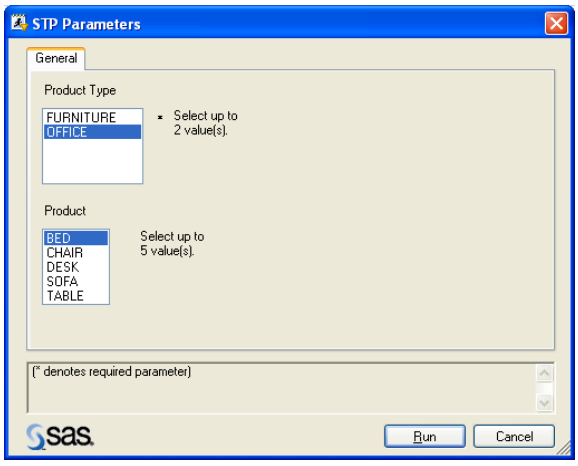
STP may be executed through the Add-in for Microsoft Office (AMO), Web Report Studio (WRS), Information Delivery Portal (IDP) , the STP web interface and through SAS Enterprise Guide (EG). Depending on the configuration and security needs, the STP can run on the server's Workspace or Stored Process Server.

SAS Stored Processes may be created using EG, Data Integration Studio (DIS) and SAS Management Console (SMC) via a simple .sas file. STPs have a number of output options and can receive user-specified input criteria to provide flexible, customized output. User specified parameters can include output destination choices, selection and grouping criteria and other options to regulate the results generated by the STP.

Current versions of SAS (9.2 and higher) are able to generate dynamic, cascading prompts for STPs, thus providing accurate, data-driven, maintenance free input parameters to drive customized results for STPs. This presentation will demonstrate how these dynamic, cascading input prompts are created and invoked.

INPUT PARAMETERS – CIRCA 9.1.3

Stored Processes often begin life as an EG process flow. EG provides a convenient, graphical method to create process flows and define input parameters to those flows. The STP wizard helpfully converts the EG prompts to STP input parameters and life is good.



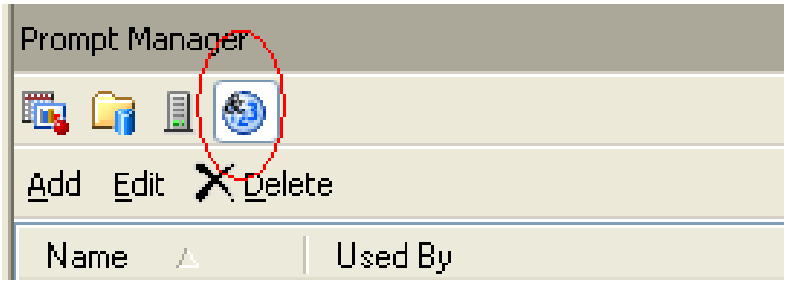
Unfortunately, in EG 4.1 projects and 9.1.3 STPs, while the values for the input parameter could be derived from a data source (e.g. SAS dataset, RDBMS table), once defined, those values were essentially "hard-coded" into the STP. And, the parameter values for one prompt could not be determined by the choice(s) made in a previous prompt. These limitations led to a maintenance problem, and just as bad, disconnected prompt choices resulted in the selections from one prompt being mutually exclusive of those made in another.

If you are familiar with the product hierarchy within the SASHELP.PRDSALE table, you will know that the following choices will generate no results, there are no BEDS in OFFICE furniture. Additionally, since the list of parameter values is built when the STP is defined, the addition of a new product line (e.g. kitchen appliances) would necessitate rebuilding the STP prompts. Unhappiness, but things begin to look up in 9.2.

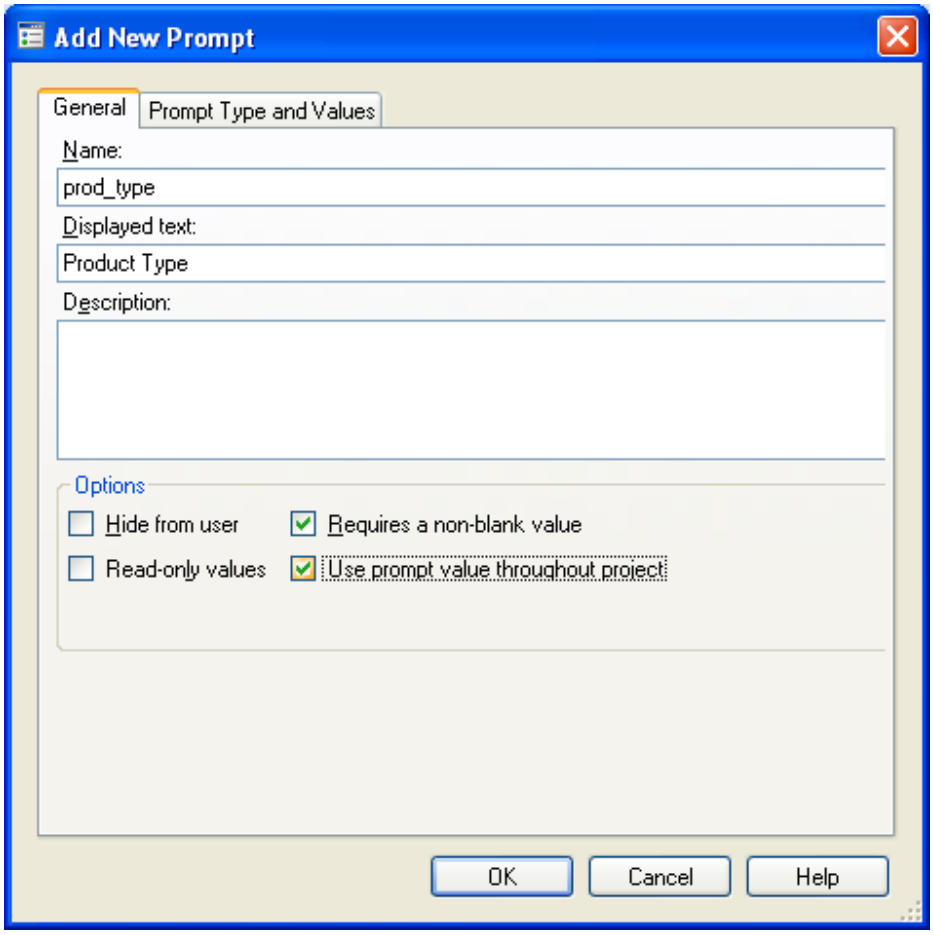
INPUT PARAMETERS – 9.2 AND LATER

In 9.2, the prompt facility was enhanced within EG and STP facility. The rest of this paper walks through portions of an EG project to demonstrate the creation of dynamic prompting, and the use of the EG STP wizard to generate dynamic, cascading prompts from the EG project prompts.

Click the Prompt Manager icon and click Add to create a new prompt:



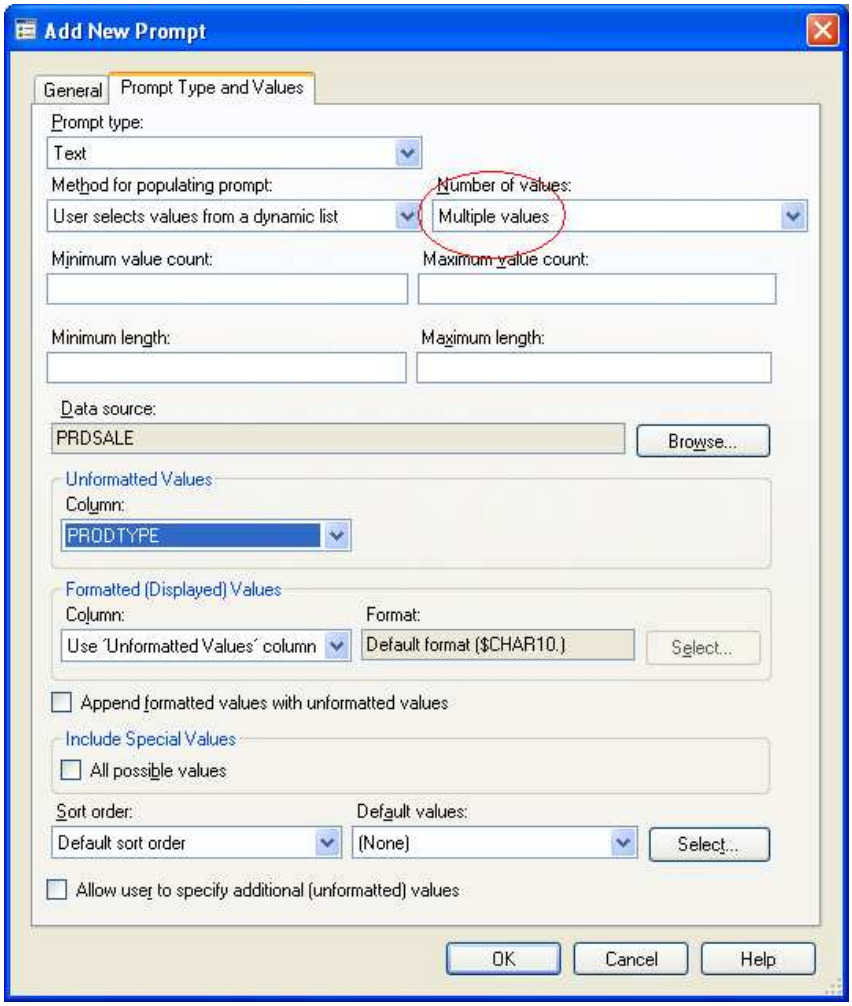
Name the prompt and provide descriptive text to be displayed with the prompt:



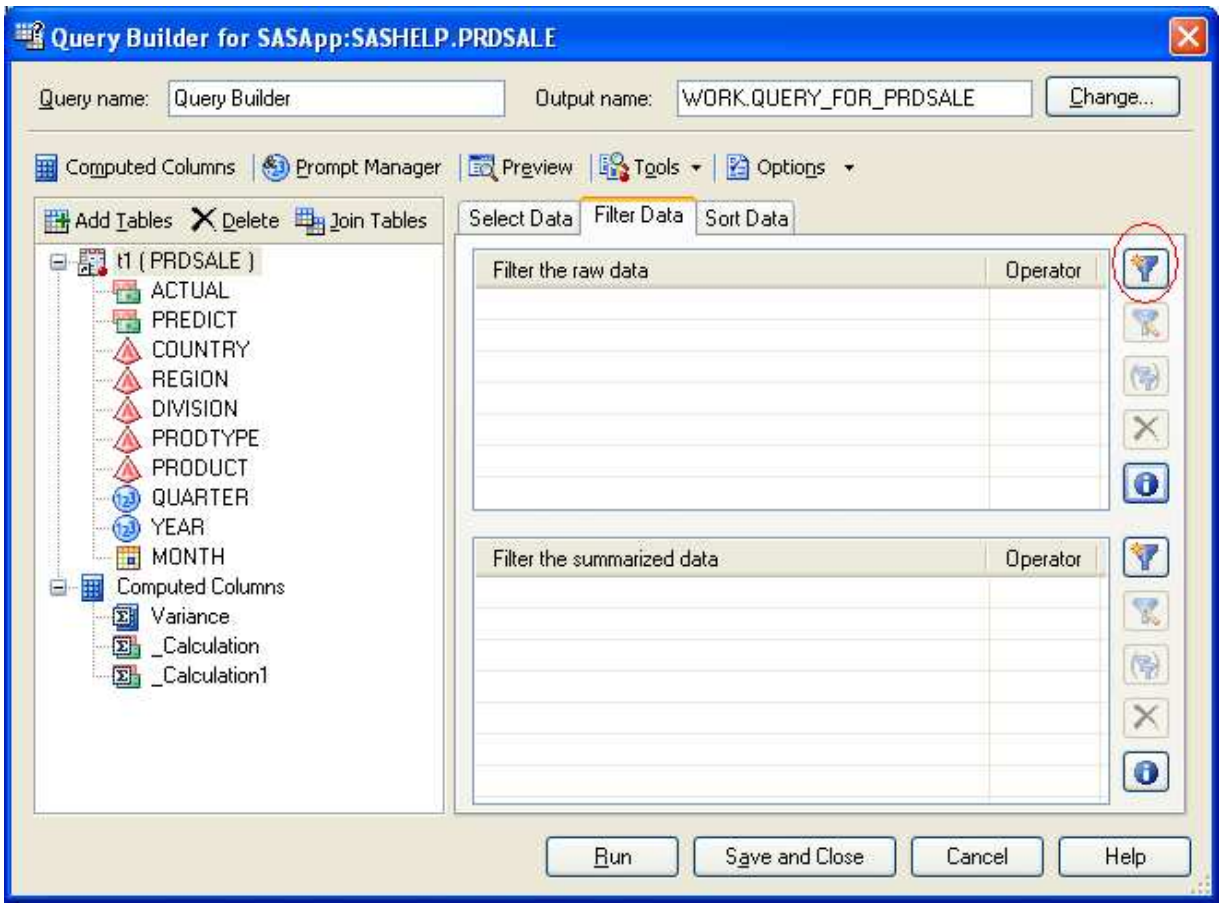
Click the Prompt Type and Values tab. Note the options available:

- Prompt type: **Text**, additional choices are numeric, date, color etc…
- Method for populating prompt **User selects values from a dynamic list**, static or user-entered
- Number of values **Multiple values**, could be single value, range of values
- Data source **Browse to data source location**, must be registered in the metadata
- Column **Column** from the data source that will source the parameter list

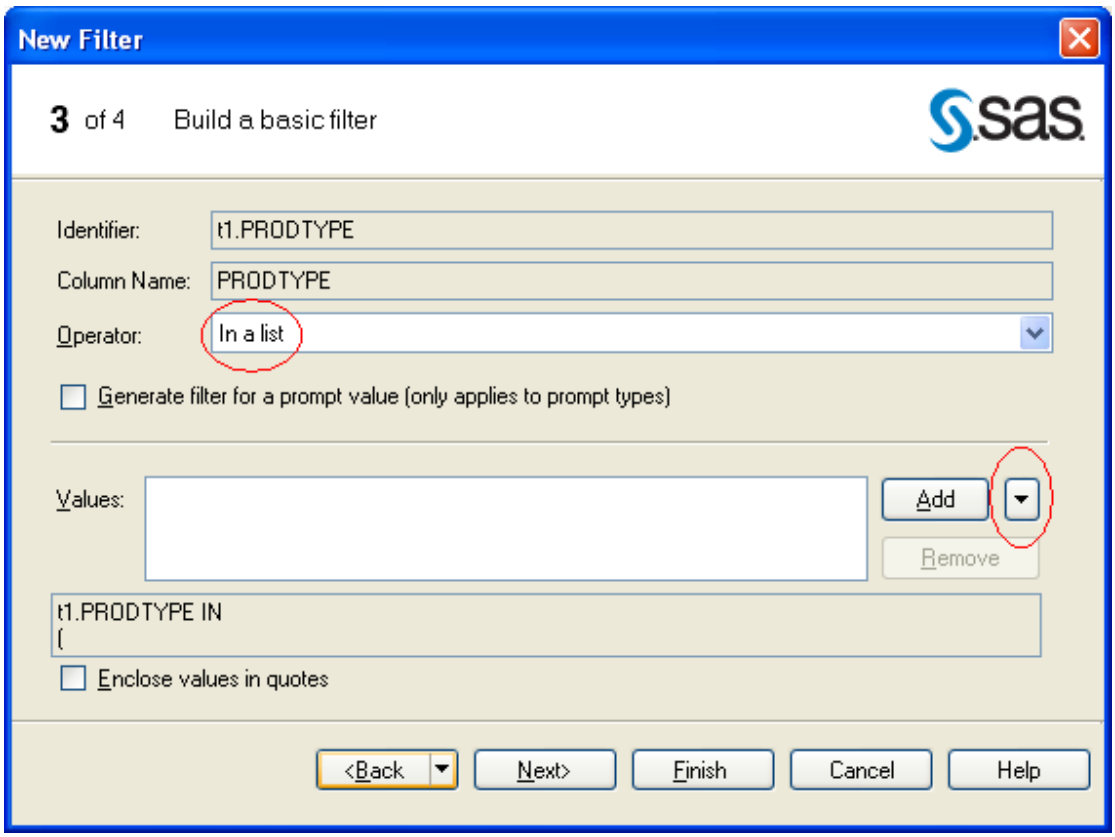
Other options may or may not be required depending on the data item selected or desired prompt usage.



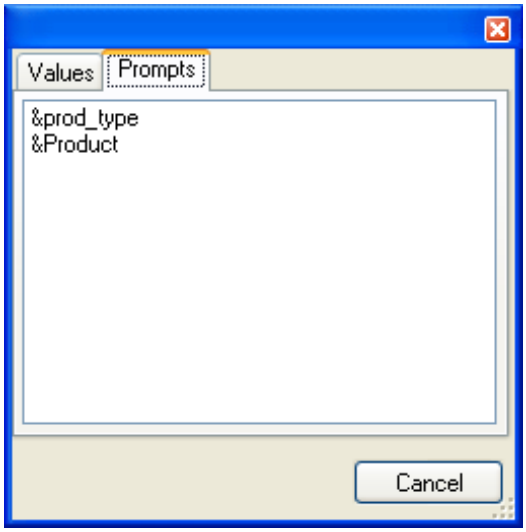
In similar fashion, create the prompt for the Product parameter. Once complete, customize the Query Builder task from the EG palette by selecting the Filter Data tab and adding a filter.



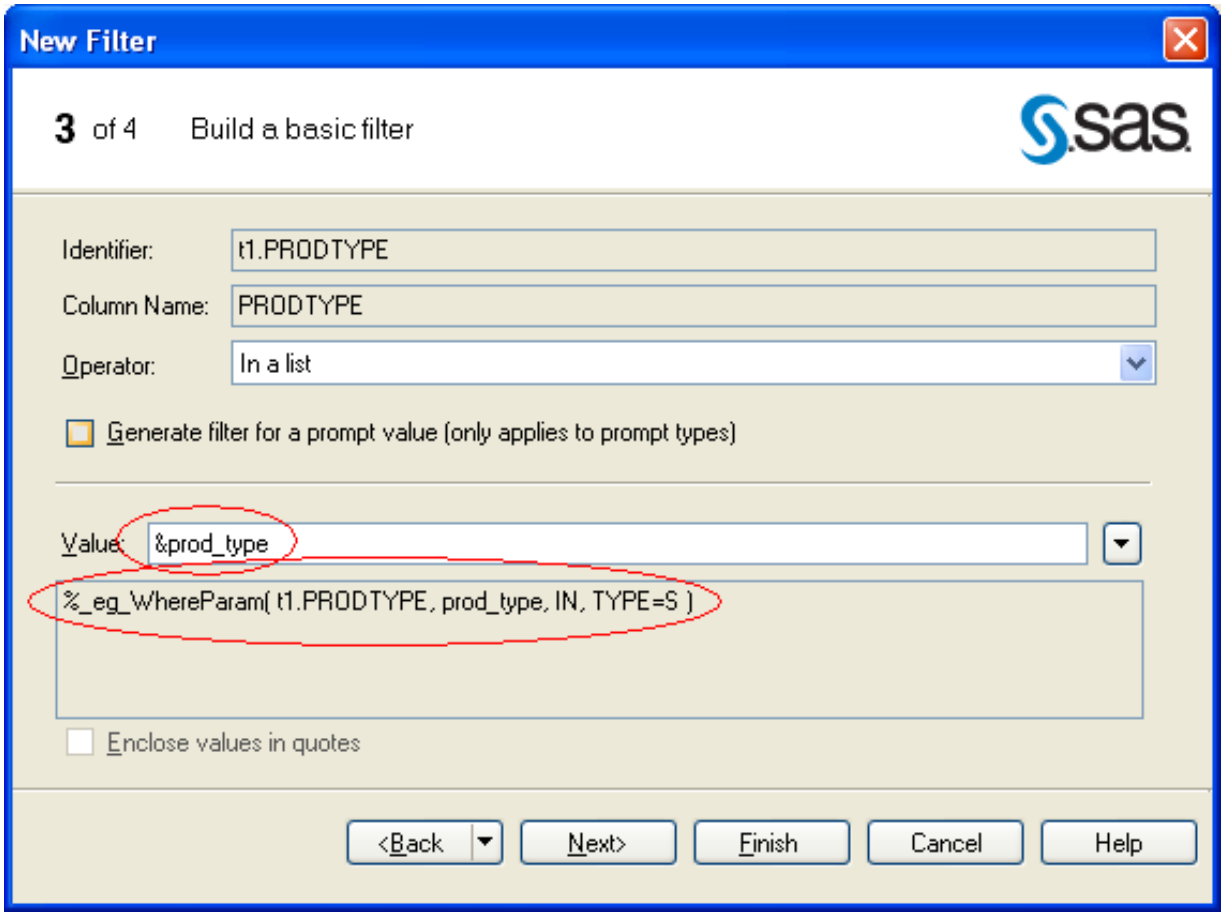
Click the drop-down arrow beside “Add” to bring up the Values / Prompts window:



Select the appropriate prompt :

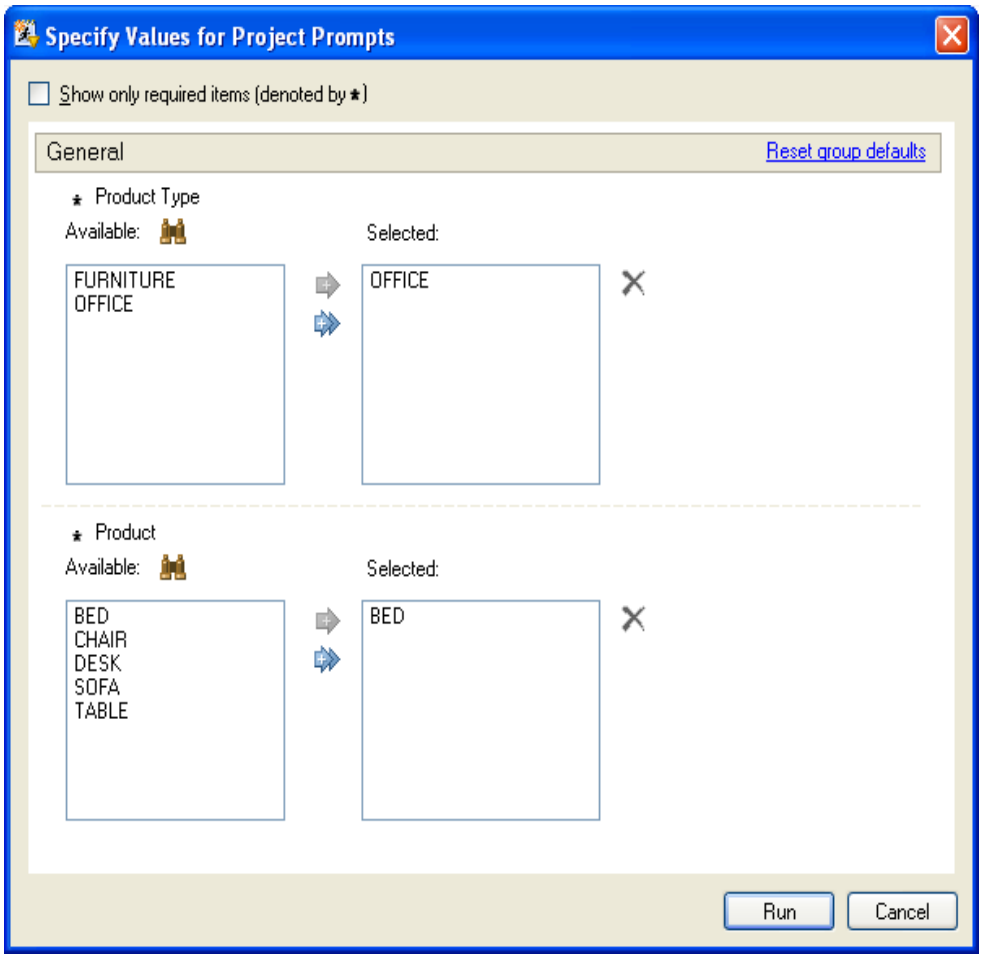


EG has done something special with the filter. Many basic filters will generate a simple macro variable substitution, but because we're accepting multiple values, EG interposes a macro invocation to deal with the *multiple* values coming from the prompt. Note the **%_eg_WhereParam** macro parms which specify the table field, the prompt name, the type of prompt and the data type of the prompt value.

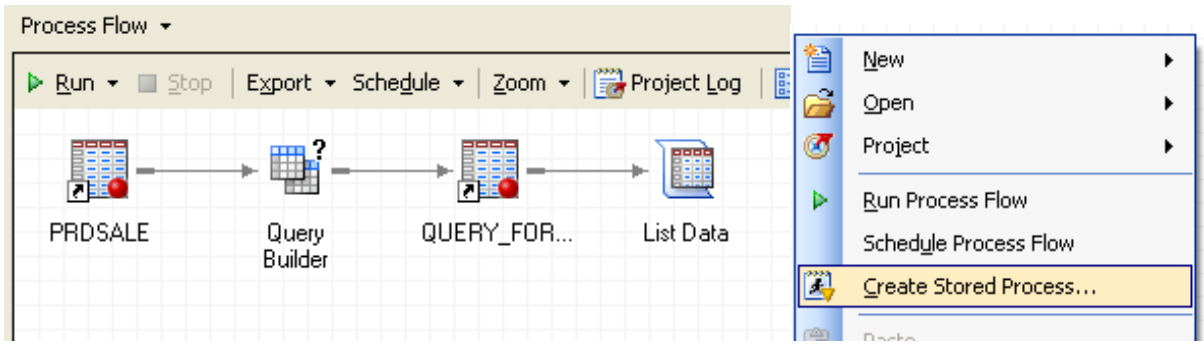


Define the other prompt to Query Builder and run the process.

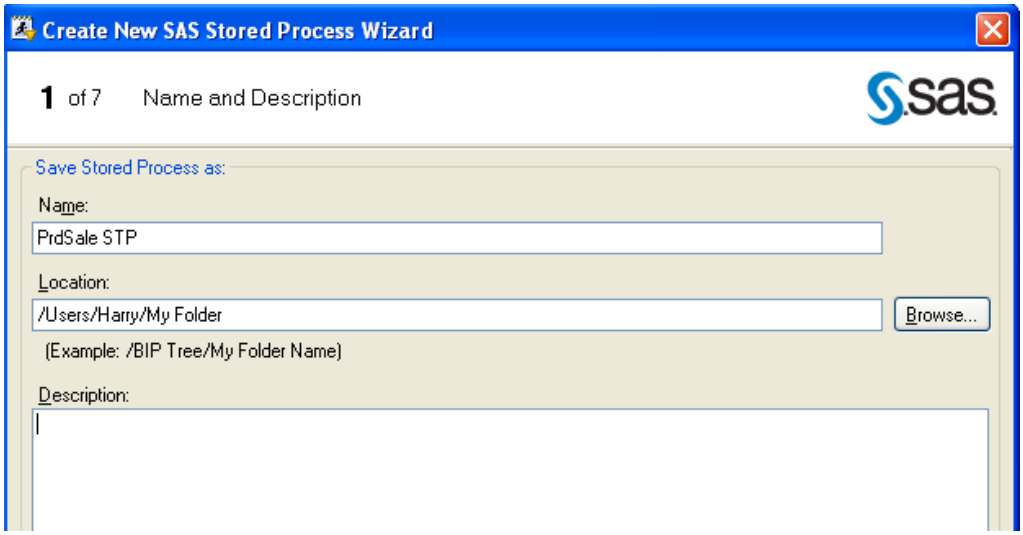
Unfortunately, while EG 4.2/4.3 prompt values are derived *dynamically* at run time, the second prompt is still not dependant on the value selected in the first. The Office / Bed combination will generate an empty report. EG 4.2 / 4.3 are not able to generate *cascading* prompts.



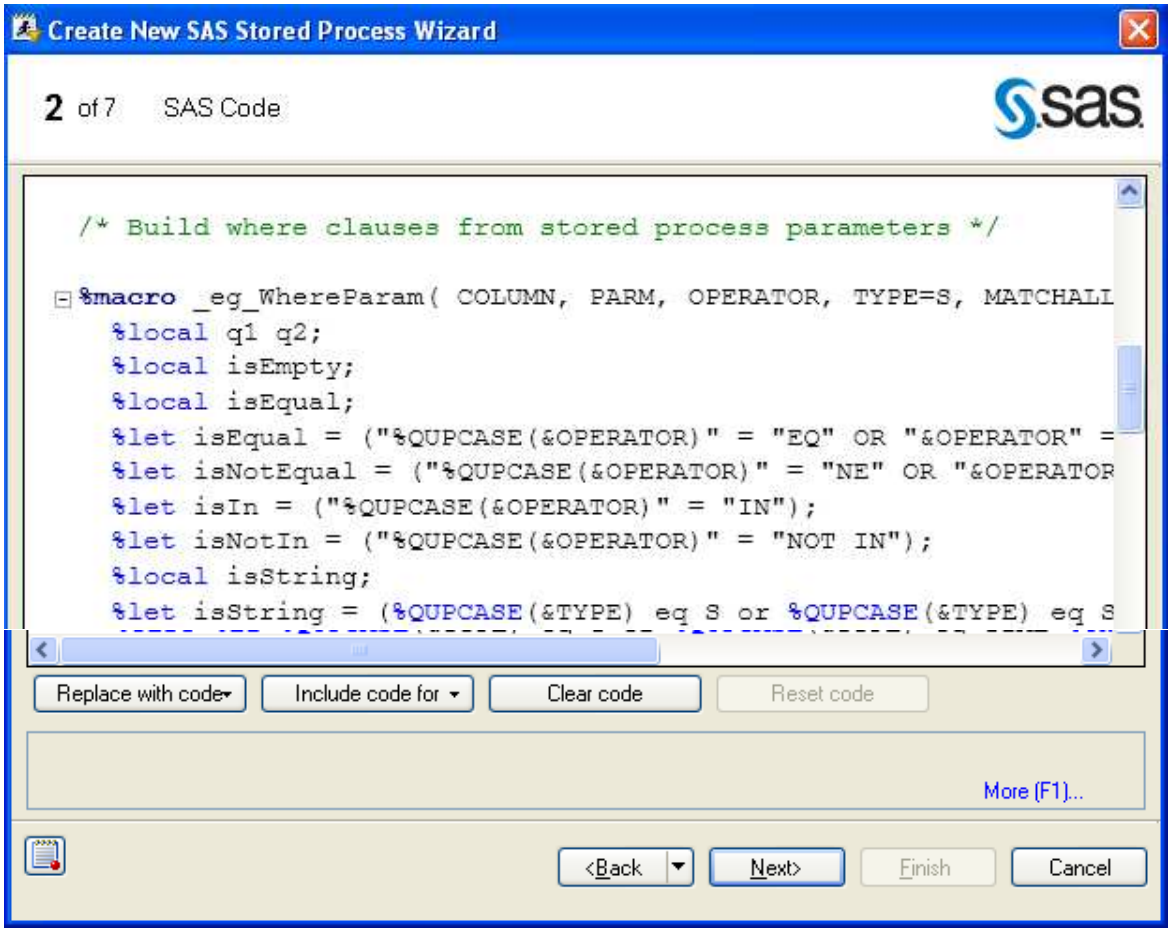
The answer is to create a Stored Process from the EG project using the STP Wizard:



Rather than walking through each step of the STP Wizard, the paper will focus on the highlights which illustrate the steps necessary to utilize dynamic, cascading prompts.



The second pane of the process allows the user to view the STP code. Scrolling through the code will display the **%_eg_WhereParam** macro definition. That's the piece of SAS-supplied code that makes sense of multiple selections, min and max range values, deals with SAS date prompts etc…

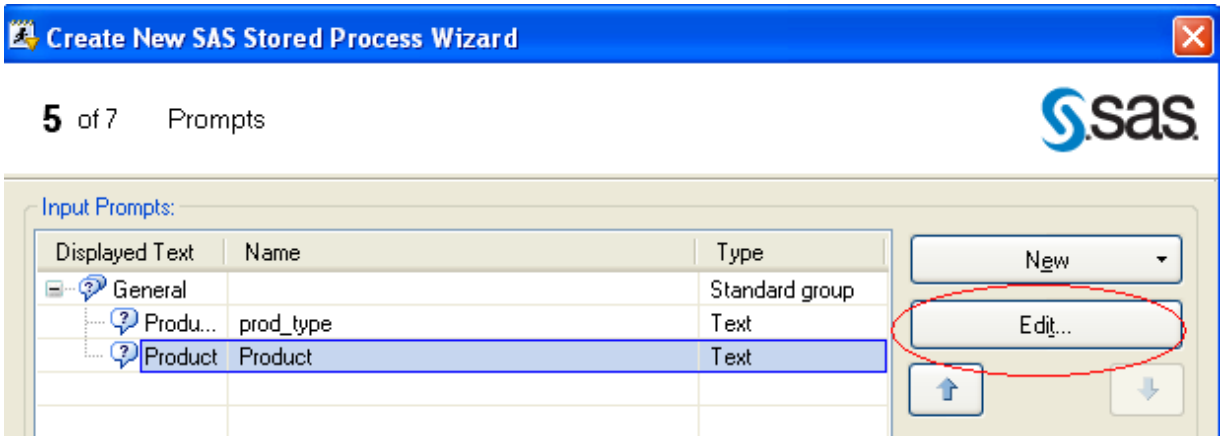


When date (/time) parameters are encountered, the values will be deal with using the familiar “&date_parm”d and “&datetime_parm”dt syntax. Ranges are converted to *column* between *&param*_min and *&param*_max. IN lists will ultimately be generated as *column* IN (*&param1, &param*2 … *&param*N).

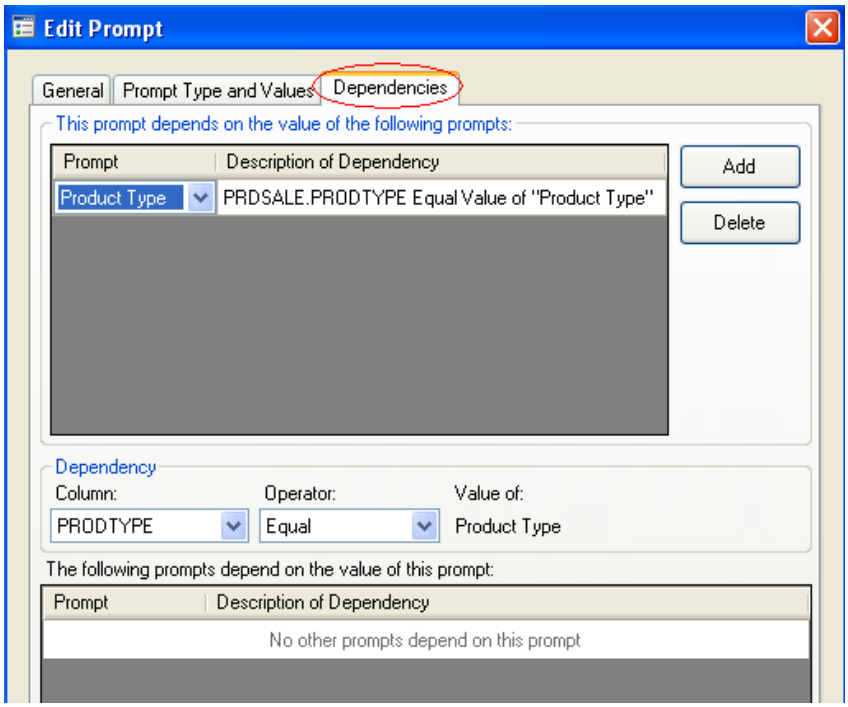
The IN list or multiple selection prompt requires a little more explanation. The number of macro variables and their values created by the parameter process depends on the number of selections made. In SAS/Intranet applications created in a bygone era, one had to check if the *&param*0 variable had been created and deal with single selection situations where it had not. Since *&param*_count is always available regardless of the number of selections made, even if none are selected, the *&param*0 macro variable can be ignored entirely.

Items Selected	Macro Variables Created	Comment
0	param_count	value of zero
1	param_count, param1	param_count = 1
> 1	param_count, param0, param1, param2 … param*N*	param_count = *N* param0 exists only when *N* > 1

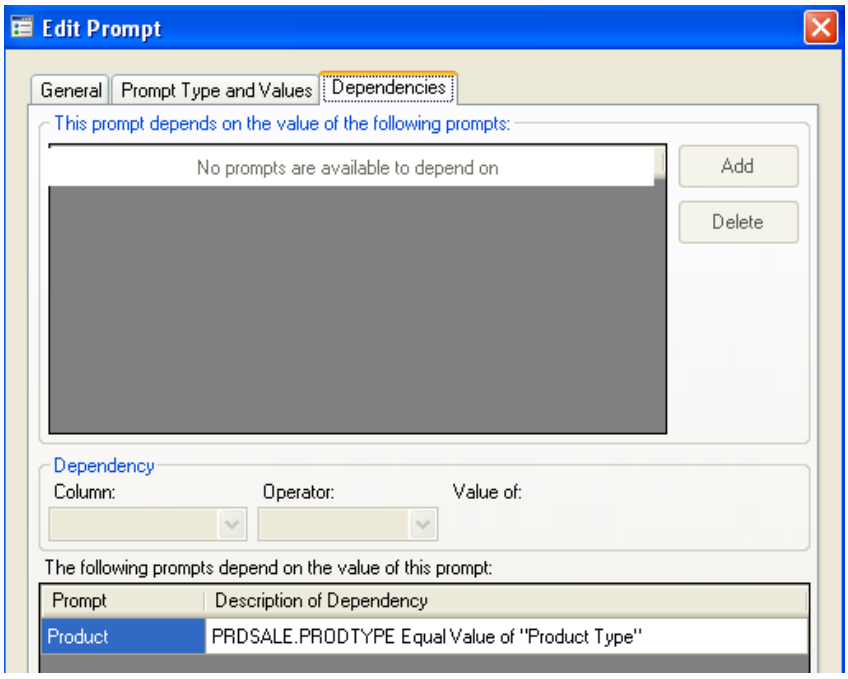
The next STP wizard panel that is of interest is where the STP Prompts are defined. The two EG prompts have been incorporated, but each prompt must be edited to create the *cascading* prompts functionality.



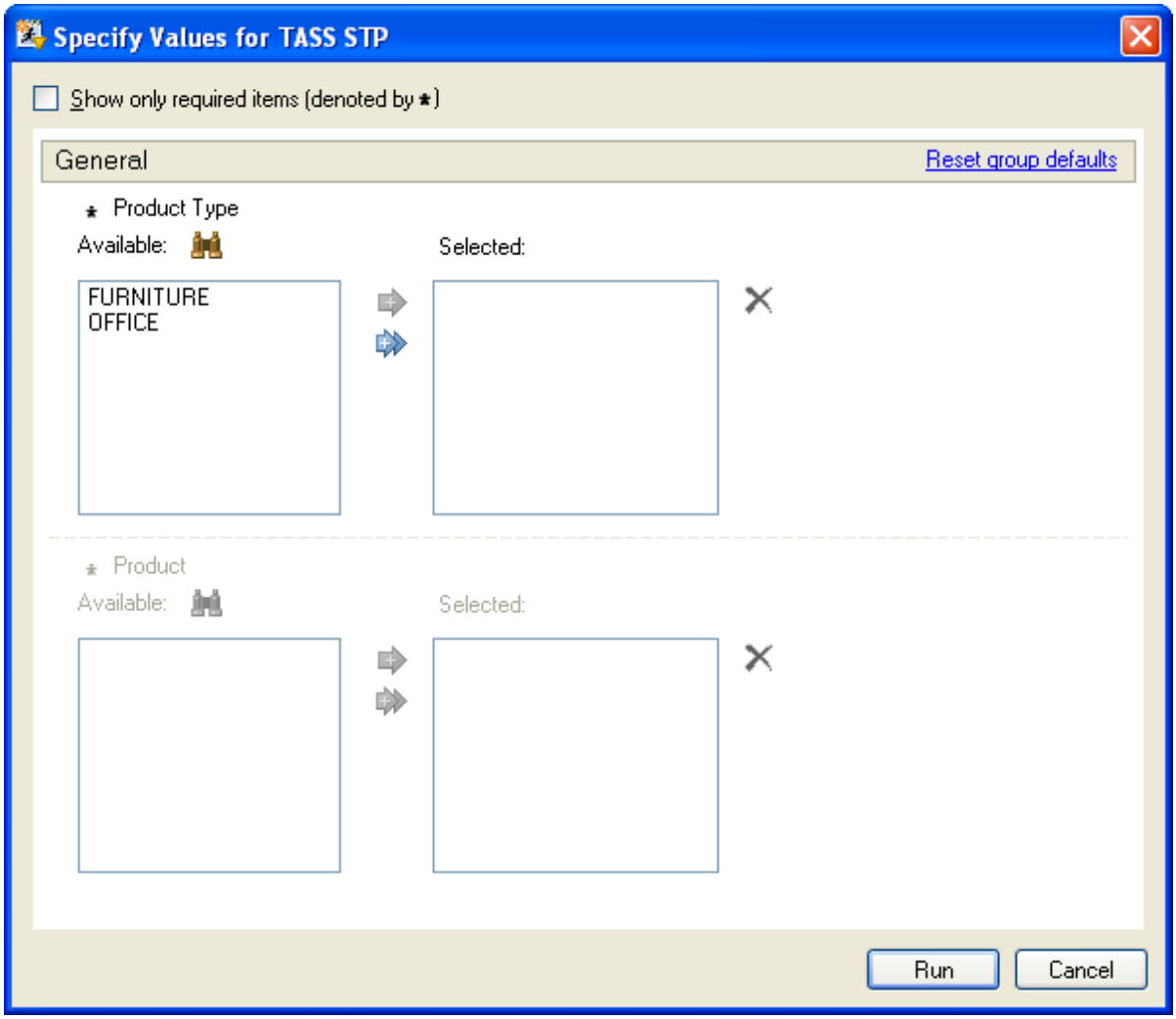
The edit prompt facility shows a third Dependencies tab. The Product prompt is defined as “dependent” on the value selected in the Product Type prompt. The bottom of the dialog shows that no other prompt is dependent on the value selected in the Product prompt, i.e. the Product prompt is the bottom of the hierarchy.



The Dependencies tab for the Product Type prompt shows no dependencies, but does show that the Product prompt is dependent on Product Type.



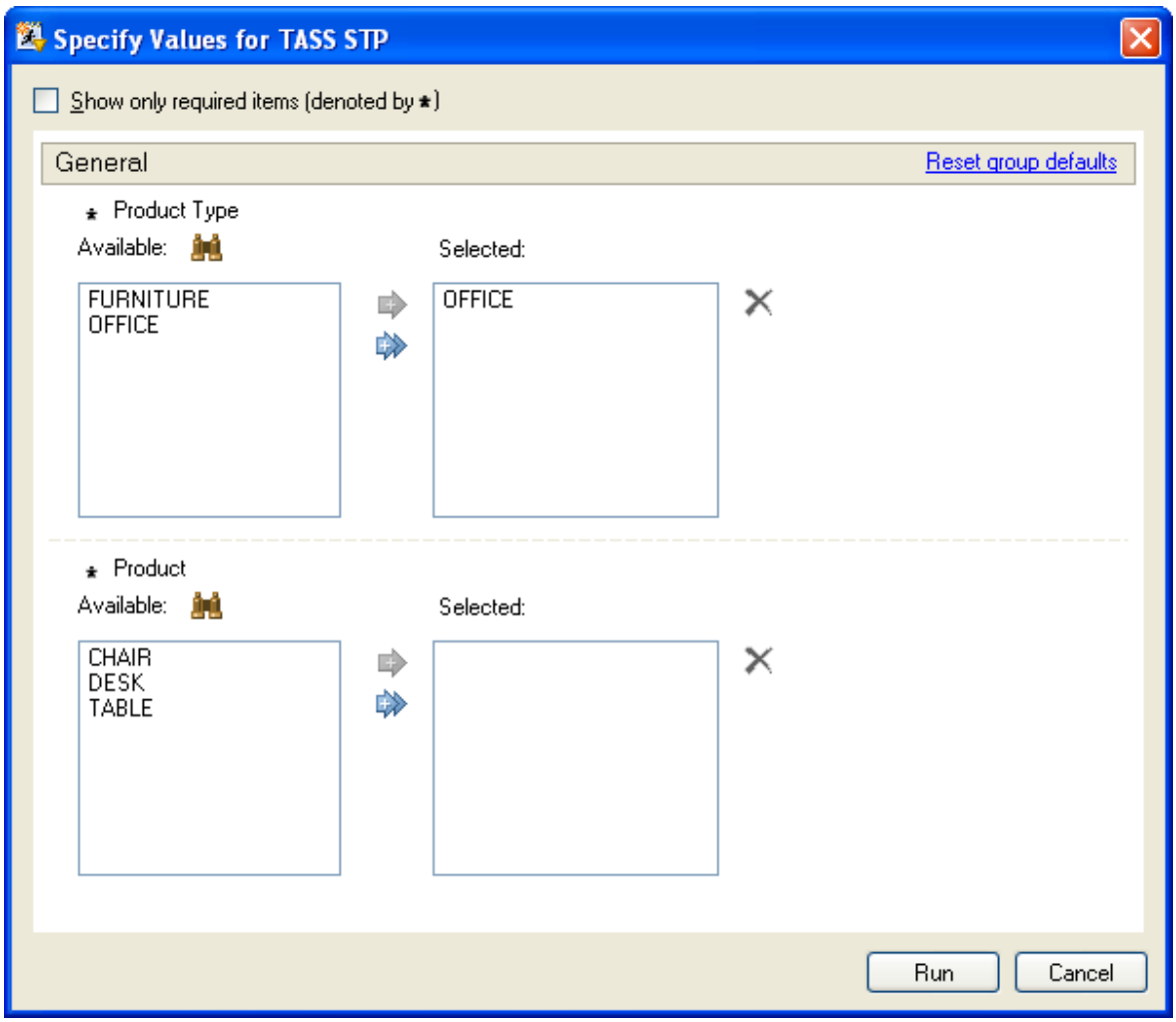
Executing the STP shows the dynamic and cascading behaviour. When the prompt facility is invoked, only the Product Type values are displayed and available for selection.



The Pooled Workspace Server is utilized to troll through the dataset specified in the prompt definition to return the unique values of the applicable column to populate each prompt.

Care must be taken to source dynamic prompts from tables that aren't too large or response time will negatively impacted. If necessary, create outrigger tables of the unique prompt column values to feed the dynamic prompts.

Only when Product Type has been selected are the Product values displayed for selection. The only Product type values displayed are those that pertain to the selected, cascaded OFFICE Product Type.

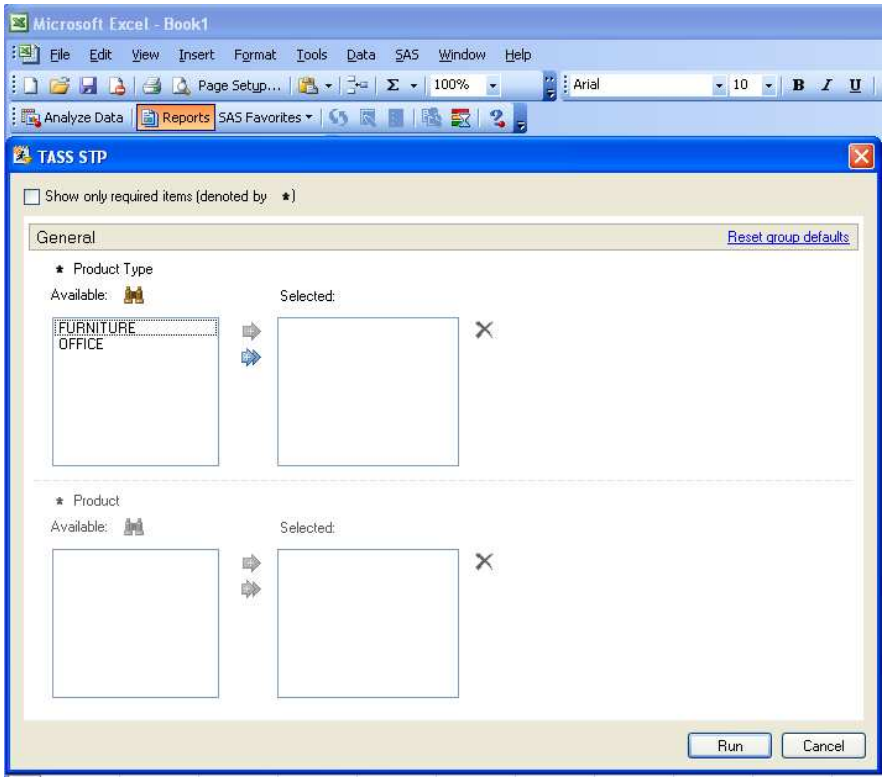


DESK was selected and the report executed, user specified results below:

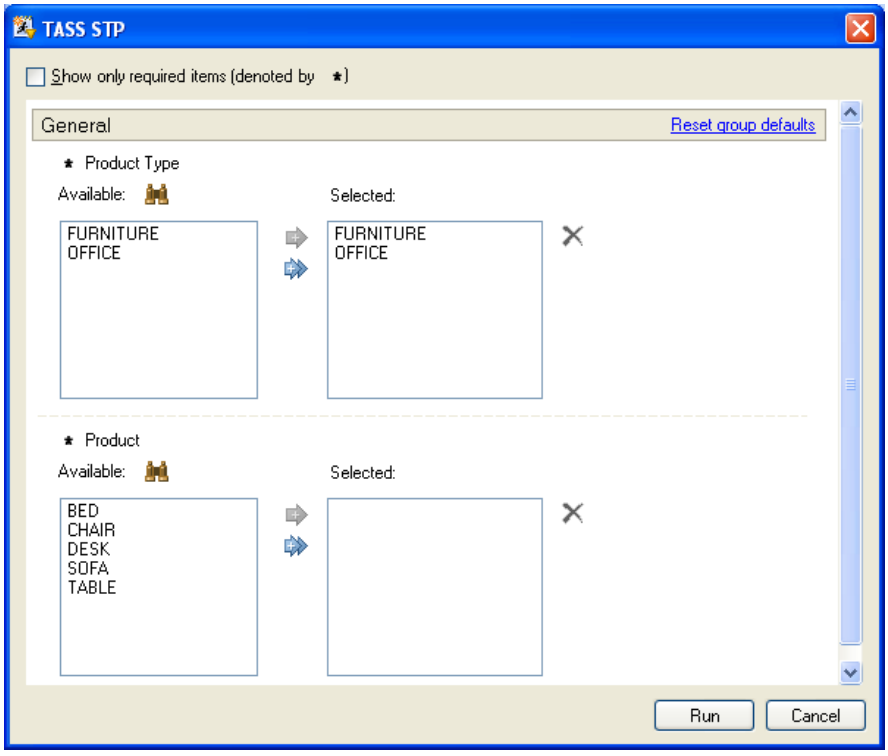
Report Listing

Row number	Year	Product type	Product	SUM_of_ACTUAL	SUM_of_PREDICT	SUM_of_Variance
1	1993	OFFICE	DESK	$76,167.00	$68,827.00	$7,340
2	1994	OFFICE	DESK	$73,065.00	$77,368.00	$-4,303

If the STP is invoked from Excel via the AMO, the prompt mechanism is exactly the same:



In this execution, because both values of Product Type have been selected, all the values of Product are available:

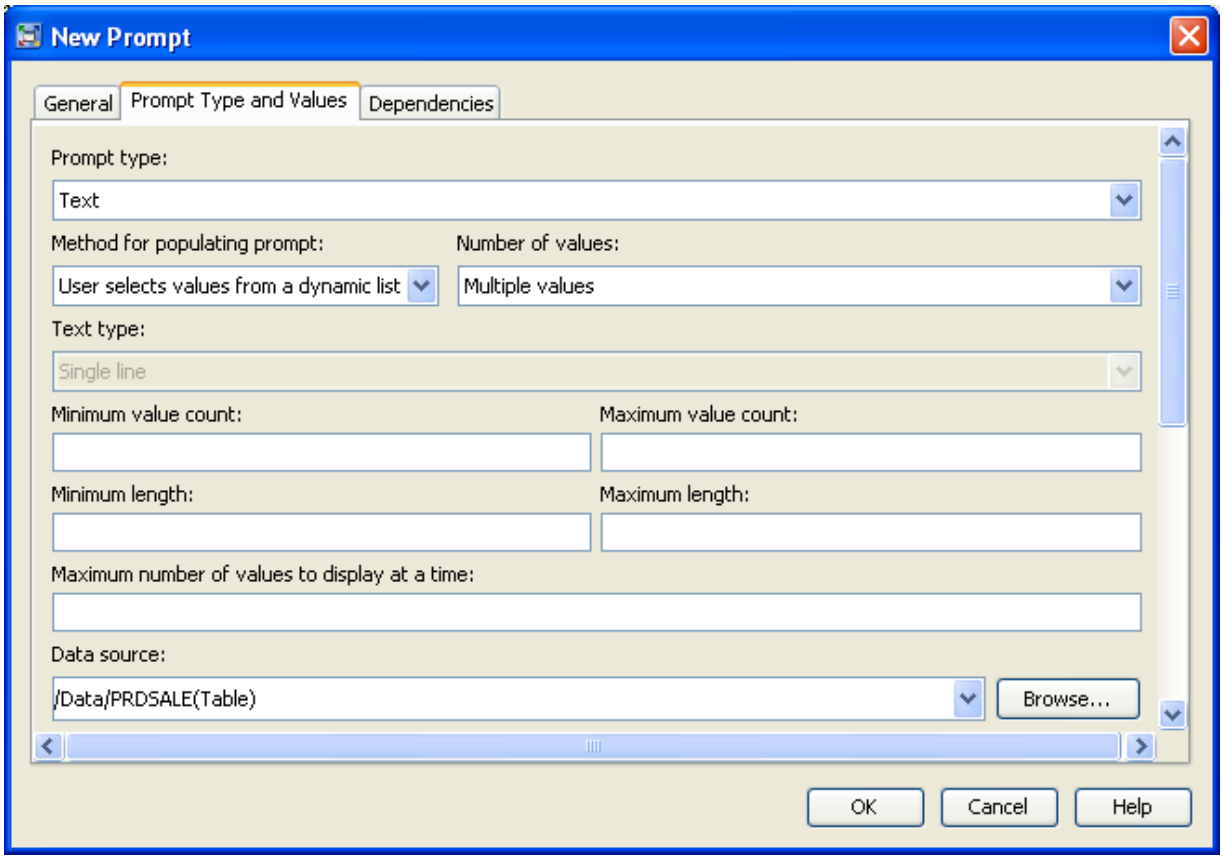


When the STP is invoked from Excel, the results can returned directly to Excel:

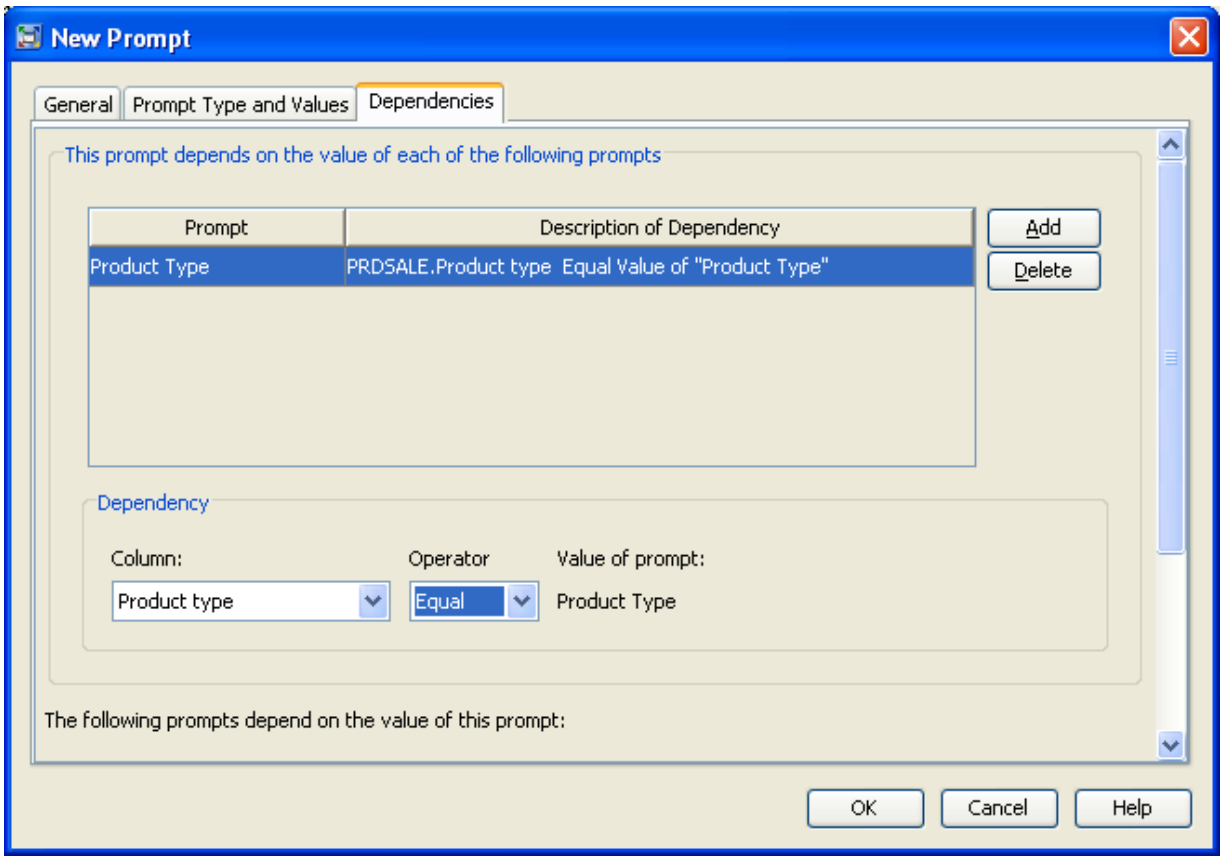
	A	B	C	D	E	F	G
1	Report Listing						
2							
3	Row number	Year	Product type	Product	SUM_of_ACTUAL	SUM_of_PREDICT	SUM_of_Variance
4	1	1993	FURNITURE	BED	$69,463.00	$70,923.00	($1,460)
5	2	1993	OFFICE	DESK	$76,167.00	$68,827.00	$7,340
6	3	1994	FURNITURE	BED	$72,574.00	$66,944.00	$5,630
7	4	1994	OFFICE	DESK	$73,065.00	$77,368.00	($4,303)

STPs with dynamic, cascading prompts can be defined in SMC as well from a .sas file. While the prompt dialogs are similar to those in EG, they do differ in some respects. It would be far better to have consistent behaviour between software within the SAS suite of products !!

In the SMC wizard, the Dependencies tab is only displayed when the 2nd prompt is defined.



The Dependencies tab, while providing the same functionality, highlights the differences between the EG and SMC interfaces:



SAS MACRO TO GENERATE NATIVE EXCEL FROM STP WEB INTERFACE

The STP web interface creates HTML output by default. To create Excel files from the STP web interface, include this macro definition and invocation in the STP .sas file ***before* the %stpbegin** line. When the STP process is invoked from the STP web interface, HTML will not be generated. Rather, the user will be prompted to save the Excel file.

```
%global _odsdest _odsoptions _odsstyle ;

%macro check_web;
      %if %upcase(%scan(%bquote(&_client),1)) = STOREDPROCESSSERVICE %then %do;
            data _null_;
              rc = stpsrv_header('Content-type','application/vnd.ms-excel');
              rc = stpsrv_header('Content-disposition',"attachment;
                        filename=detail_extract_&sysdate9._&systime..xls");
            run;

            %let _ODSDEST       = tagsets.excelXP;
            %let _ODSOPTIONS    = options (autofilter='all' frozen_headers='yes' );
            %let _ODSSTYLE      = seaside;
      %end;
%mend check_web;

%check_web

%stpbegin;
```

CONCLUSION

SAS Stored Processes are very useful to define a single version of the truth, ensuring embedded business rules are reported consistently through the enterprise. The increased functionality of SAS 9.2 have made it possible to define dynamic, cascading prompts in SAS Stored Processes. STPs created with this facility will enhance reporting and processing flexibility and increase user satisfaction as the customized results they need are delivered dynamically.

CONTACT INFORMATION

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

Harry Droogendyk
Stratia Consulting Inc.
www.stratia.ca