Paper BB12-2015

WOW! YOU DID THAT WITH SAS® STORED PROCESSES?

Dave Mitchell, Solution Design Team, Littleton, Colorado

ABSTRACT

In this paper you will be introduced to advance SAS® 9.4 functions developed with SAS® Enterprise Guide 6.1 and surfaced thru HTML pages.

The application design is based on SAS® 9.4 client server architecture featuring a discussion of integration with metadata objects in the SAS® Management Console.

The audience will also learn practical techniques used to create stored processes that will cover:

- Direct Call to Stored Processes
- Macro Code
- Stored Process Linking
- Prompt Dependencies
- Hidden Prompts
- Prompt Groupings
- Dynamic Prompting
- Formats and Outputs – email and pdf

This application is a set of linked stored processes that allow tracking of internal customer contracts. The application integrates several SAS® tables driven by stored processes to:

Maintaining a customer database
Create projects from contracts
Create work orders
Allow posting of time spent for consultants
Create output for a multitude of reports, emails and PDF files.
Special tables for maintaining unique keys
Tables to maintain fairly static data
History Files for all project work completed
Special Tables for Stored Processes that maintain Tables and Reports

INTRODUCTION

This paper is presented as a case study highlighting a business problem and how it was addressed using SAS® Stored Processes. The customer was dealing with a situation that involved maintaining multiple spread sheets, extra time, mistakes, and dirty data. The process was further complicated with maintaining data quality when transferring to the special forms to fill out for invoicing and time sheets for tracking projects. This was taking a lot of time between the resources and accounting. The customer wanted to streamline this process as well as track projects and resources over time, so they could get a better handle statistically on how the company was doing with each project. So we designed a customer tracking system and decided to write it with Stored Processes. It was designed to be used with any SAS® bundle that has Integrated Technologies and utilizes SAS® Management Console and SAS® Access for ODBC.

As stated in the abstract above, we have used a combination of SAS® techniques to maximize the functionality of our application. Unfortunately, it would take much more than 20 minutes to cover all of them. This paper is going to explain how our application was developed, touching on, but not detailing behind the scenes development. Note: all of the development SAS® code was built using SAS® Enterprise Guide 7.1.

TABLES FLOWCHART

This is a flowchart of the tables we developed for the application.
ACCOUNT_DIM – has customer information. Key accountid.

PROJECTS – has project information
links to account_dim by accountid
must enter workorder #(3 characters max), purchase order(optional), accountexec(optional), Projectmanager(opt), requested start date and end date(req), Type(t&m,Fixed), Expenses(opt)

WORKORDERS-Has customer work order information
Linked to Projects table by accountid and workorder(dependant prompt)
Customer work order created.
Prompt for consultant from consultant table,
Load rate to pay consultant on this project

POSTING – consultant enters information for work being done by day. Can have multiple postings per day
Linked to consultant(workorder table)
Linked 2nd by accountid(dependant on consultant)
Linked 3nd by Workorder(dependant on consultant and accountid)
Enter day of work, hours of work, and activity(can add or link to activity table).

These are independent tables for specific needs

ACCOUNTEXECS
ACTIVITIES
CONSULTANTS
PROJMGRS

These are for Main Tracking list
APPLICATIONS
REPORTS

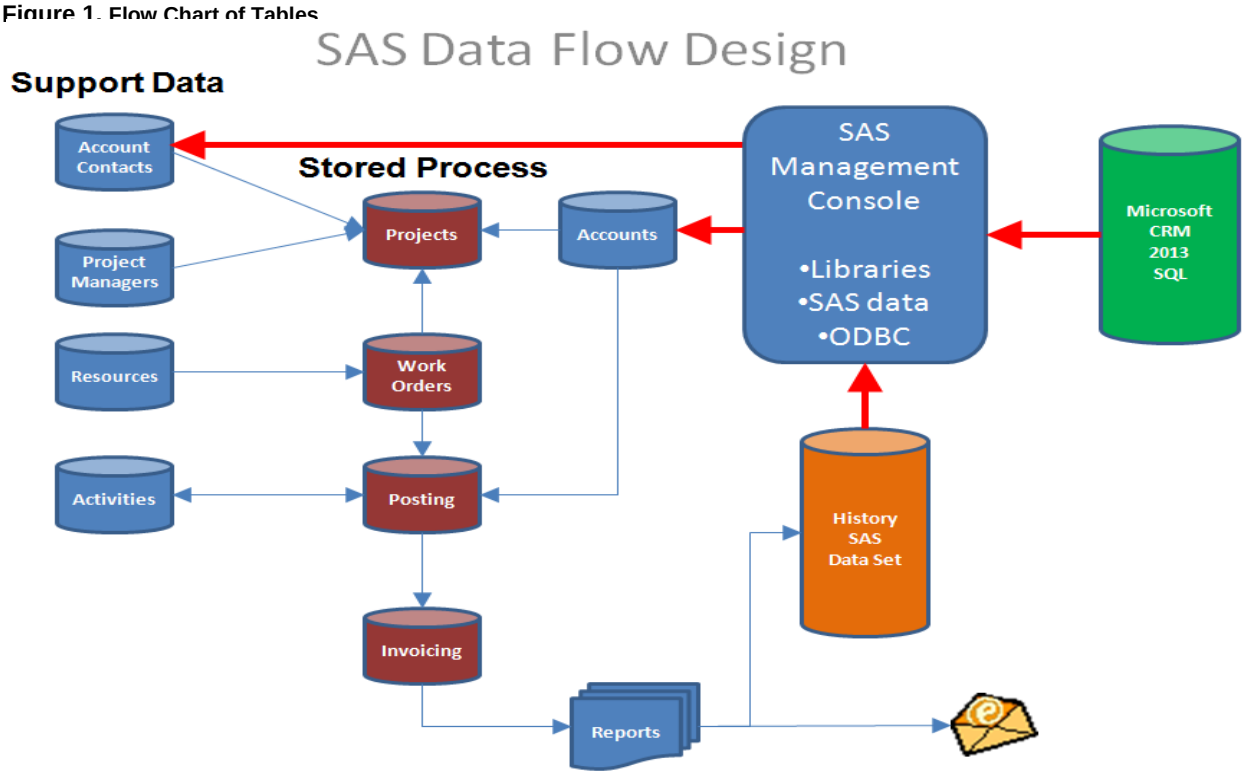
This table links info from above two tables directly to Stored Process name
URL

These are for keys created automatically
LAST_ACCOUNTID
LAST_AE_KEY
LAST_CONSULT_KEY
LAST_PM_KEY
LAST_WO

This one is combination of project/.workorder for better identifying in reports
PROJ_WO

These are for History for forecasting and statistics
POSTING_HISTORY
PROJECTS_HISTORY
WORKORDERS_HISTORY

Figure 1. Flow Chart of Tables



DIRECT CALL TO STORED PROCESSES

GUEST ACCOUNT

In order to be able to give the fastest access to this application we wanted to bypass the SAS® login screen. There are several ways to accomplish this. We chose the one that uses the web guest account, webanon@saspw as our method. This will use <my-domain>\sassrv as its external id to authenticate with Windows.

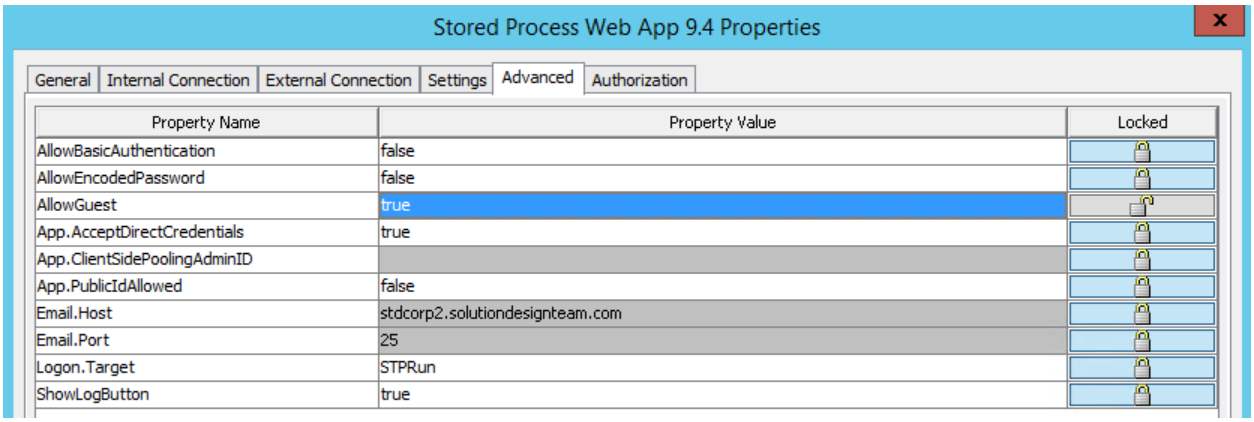
Procedure to bypass the SAS® login screen for 9.4

1. Invoke Management Console
2. Under Plug-ins>Configuration Manager>SAS® Application Infrastructure>Store Process Web App 9.4
3. Properties
4. Advanced
5. Change AllowGuest from false to True
6. OK

The SAS® [Config-Lev1] WebAppServer SASServer_1 service must be restarted(this can take some time)

In the case of this paper, the MainTracking Stored Process can now be invoke on the web directly with this url

http://<yourSAS®server>/SAS®StoredProcess/guest?_action=form,properties,execute,nobanner,newwindow&_program=%2FApplications%2FSolutionDesignTeam%2FTracking%2FStoredProcesses%2FMainTracking



Display 1. Stored Process Web App 9.4 Properties settings

USING PROMPTS

Our entire application is dependent on stored process prompts. To complete the application, we used prompts in several forms.

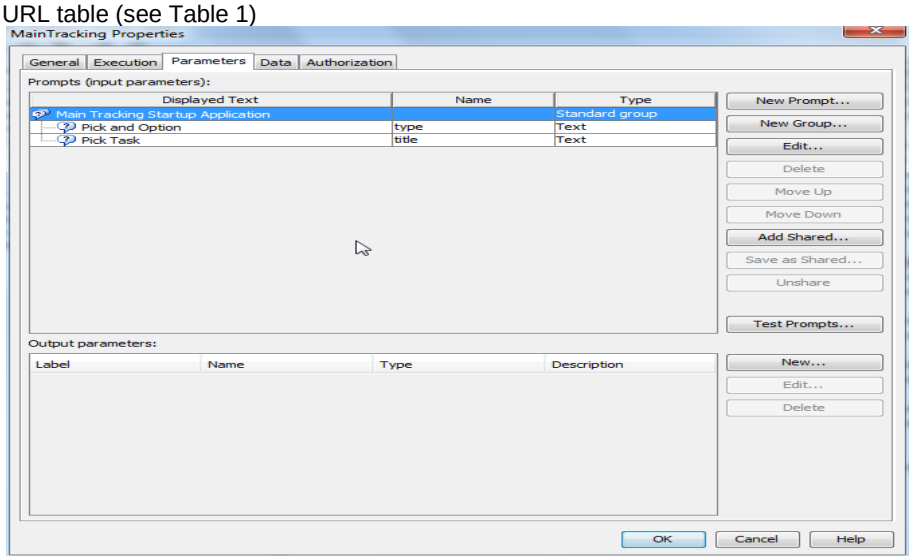
- Simple Prompts
- Dynamic Prompting
- Prompt Dependencies
- Hidden Prompts
- Prompt Groupings

```
data _null_; set tracking.url(where=(TYPE="&TYPE" and TITLE="&TITLE"));
    call symput('storedprocess',COMPRESS(storedprocess));
run;
```

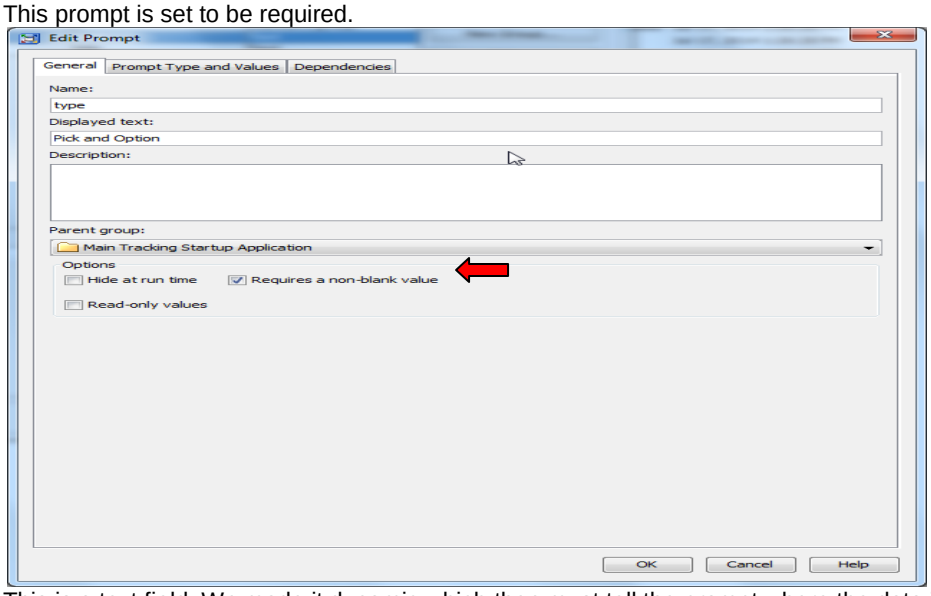
DYNAMIC PROMPTING

Dynamic prompting is a very powerful prompt that links to a column in a table. The table can be SAS® or any other database that has been registered in Management Console. In the MainTracking Stored Process that we are using as an example there are two prompts, option and type.

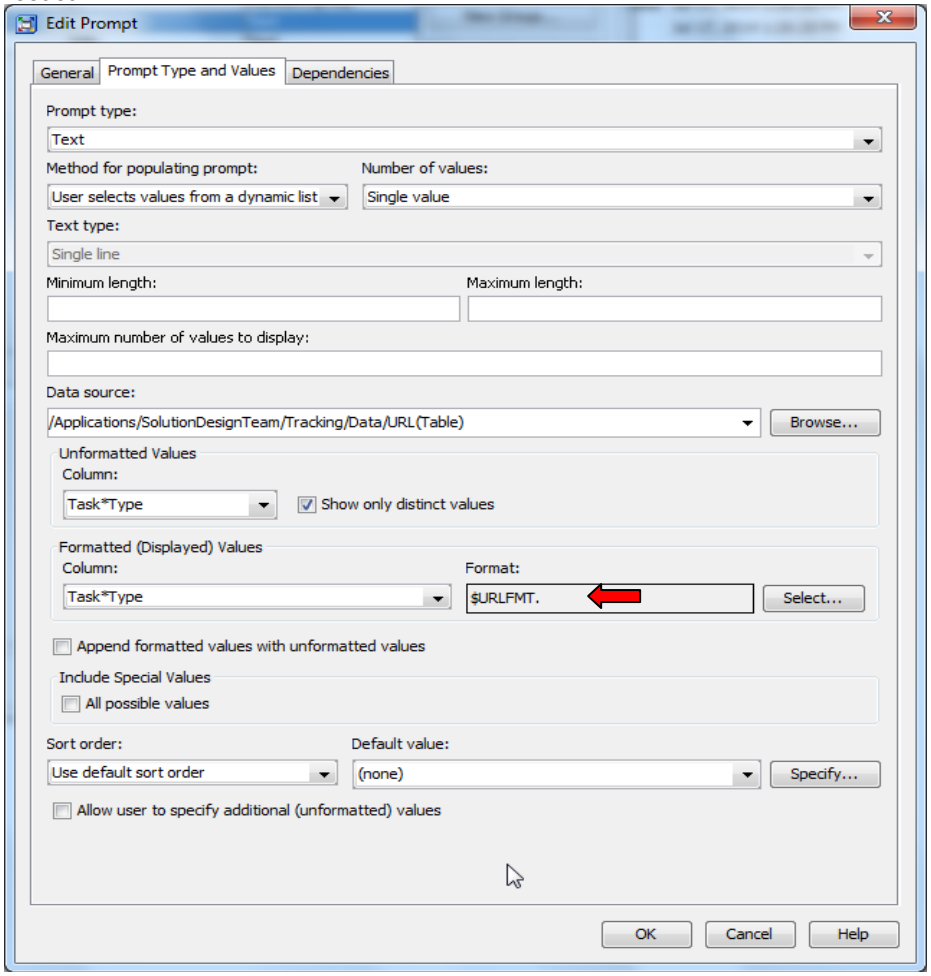
NOTE: You can created and modify prompts two ways, Enterprise Guide and Management Console. THEY DO NOT HAVE THE EXACT SAME LOOK AND FEEL. The first prompt we will discuss is dynamic. The data for this is in the URL table (see Table 1)



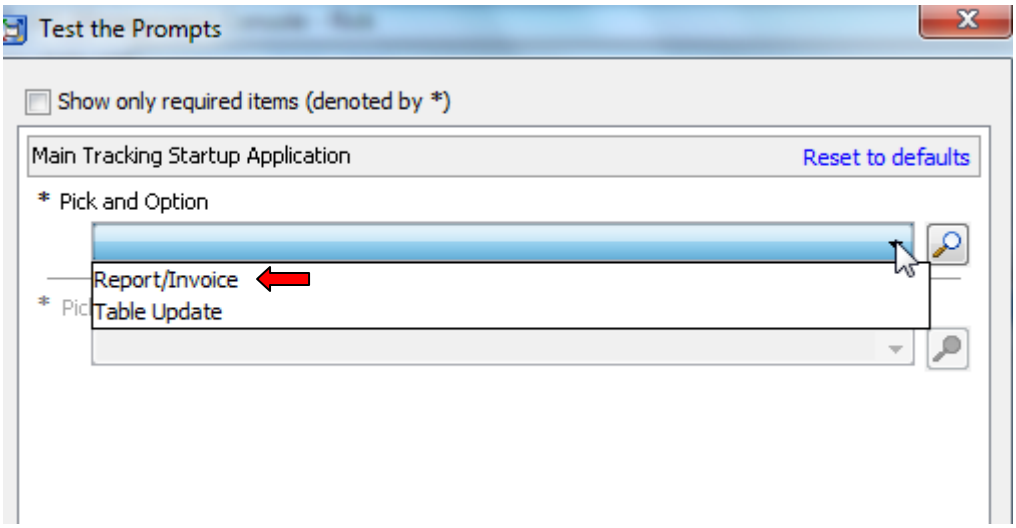
This prompt is set to be required.



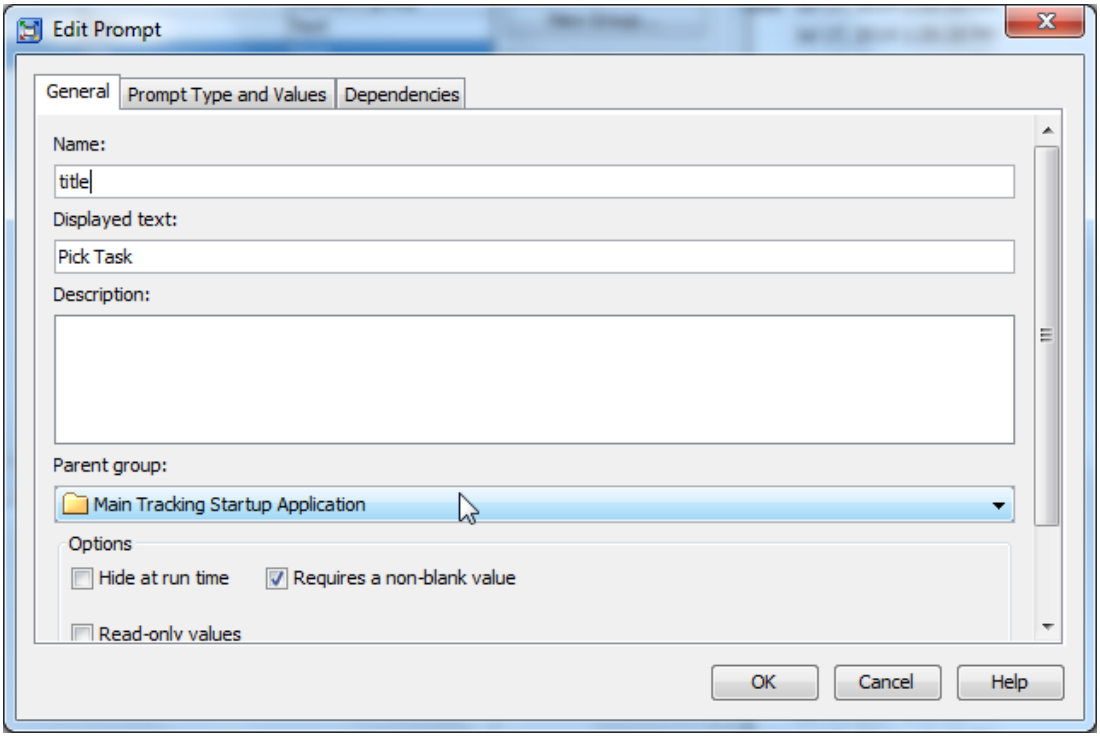
This is a text field. We made it dynamic which then must tell the prompt where the data is located, and the column needed.



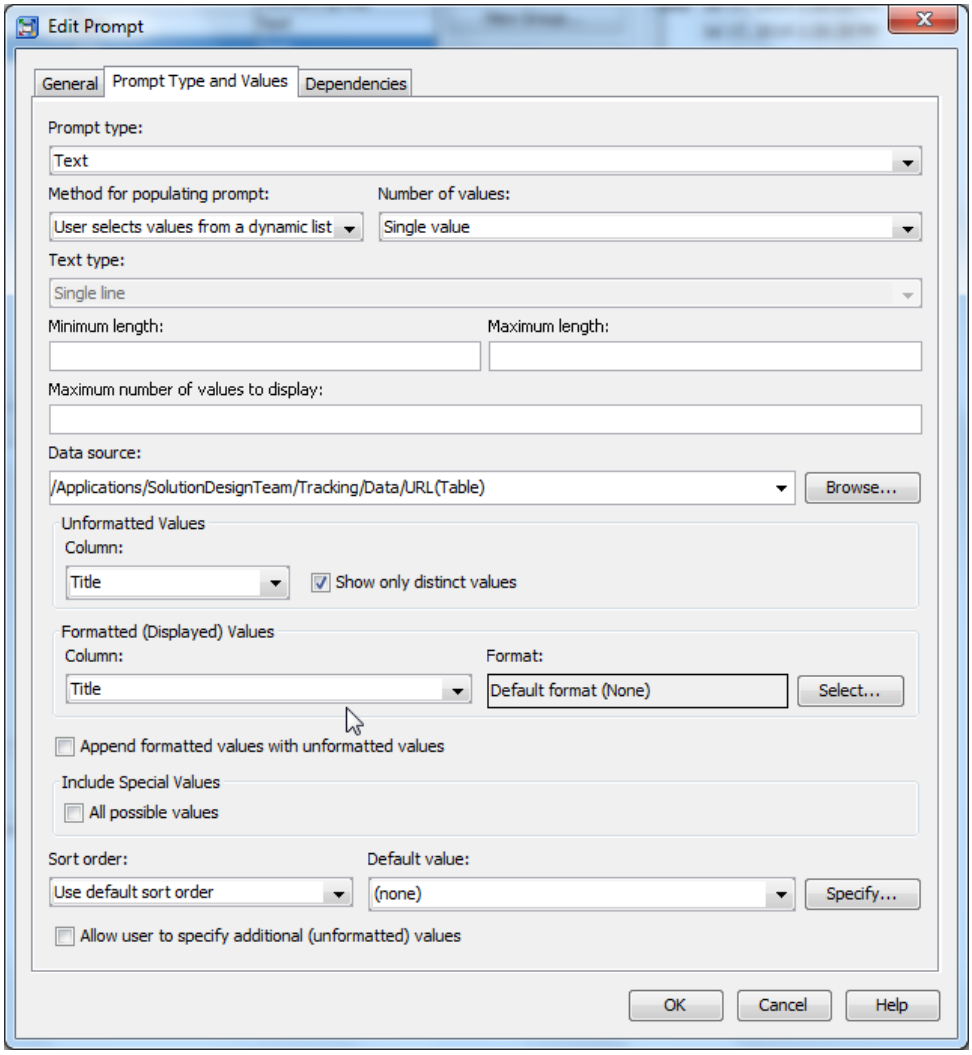
As can be seen above, the column label shows up. We also put a format we had created so the two rows would show better definition.
R will show Report/Invoice from the format
T will show Table Update from the format

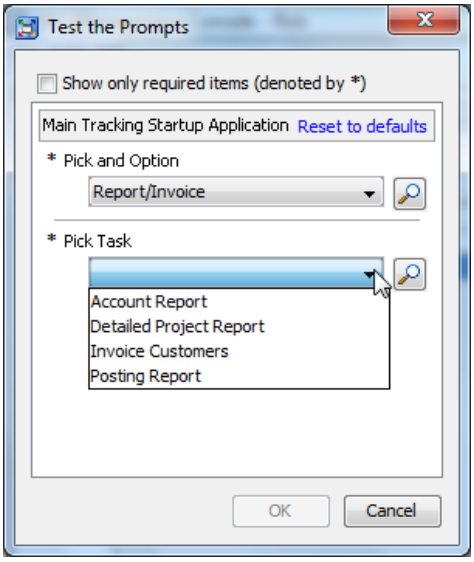


In this case, we are choosing Report/Invoice
Now we will discuss the second prompt title.



We have also made this prompt dynamic using the same table but with the title column. This title column was created to be more descirptive than the stored process name. as can be seen, this does not have a format.


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
Text type:
Single line
Minimum length:
Maximum length:
Maximum number of values to display:
Data source:
/Applications/SolutionDesignTeam/Tracking/Data/URL(Table)
Browse...
Unformatted Values
Column:
Title
Show only distinct values
Formatted (Displayed) Values
Column:
Format:
Title
Default format (None)
Select...
Append formatted values with unformatted values
Include Special Values
All possible values
Sort order:
Use default sort order
Default value:
(none)
Specify...
Allow user to specify additional (unformatted) values
OK
Cancel
Help

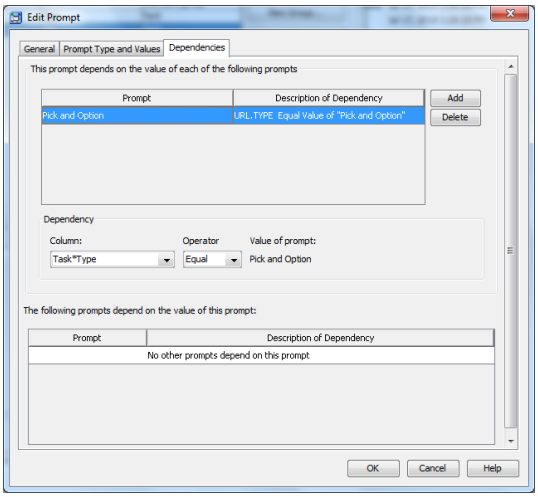

Test the Prompts
Show only required items (denoted by *)
Main Tracking Startup Application
Reset to defaults
* Pick and Option
Report/Invoice
* Pick Task
Account Report
Detailed Project Report
Invoice Customers
Posting Report
OK
Cancel

PROMPT DEPENDENCIES

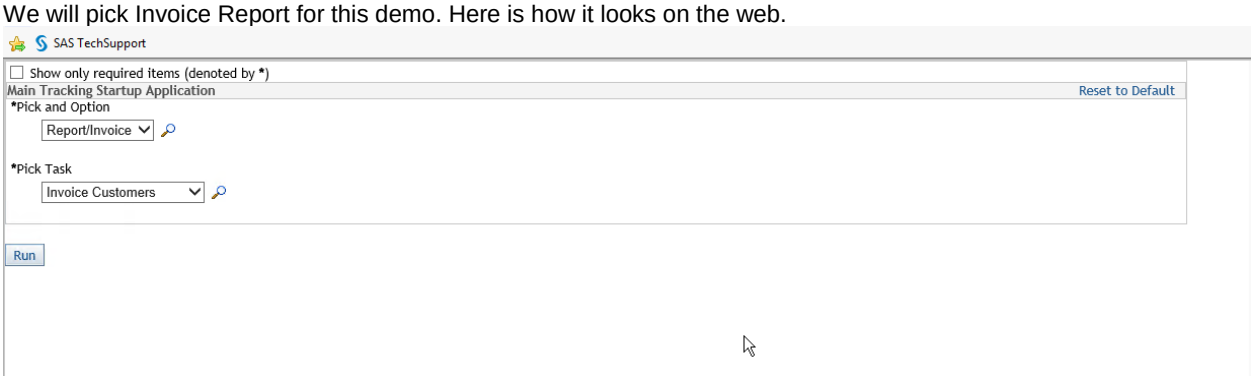
Before we continue, there is one more issue to address. Since each stored process is tied to a record and the the url table, we want to make sure that happens. To do this, we will make the title prompt dependent on the type prompt.

NOTE: Dependancies are only available if dynamic prompts are used.

We have made the title prompt dependant on the type prompt. We also made it an exact match using equals.



We will pick Invoice Report for this demo. Here is how it looks on the web.



We we click on Run, the following macro will execute.

```
%macro maintracking;
  data _null_; set tracking.url(where=(TYPE="&TYPE" and TITLE="&TITLE"));
     call symput('storedprocess',COMPRESS(storedprocess));
run;
  data _null_;
   file _webout;
   put '<script type="text/javascript">';
  put
%unquote(%str(%'self.location="http://sdtdev2.solutiondesignteam.com/SAS®StoredProcess/guest?_action=form,properties,execute,nobanner,ne
wwindow
%nrstr(&_program)=%2FApplications%2FSolutionDesignTeam%2FTracking%2FStoredProcesses%2F&storedprocess"%'));
                    put '</script>';
                    run;
%mend;

%maintracking;
```

The first part of the macro is DATA _NULL_ code to get the final value storedprocess from the url table. We then compress it so it will be clean(no hidden characters) and store it as another macro.

```
data _null_; set tracking.url(where=(TYPE="&TYPE" and TITLE="&TITLE"));
     call symput('storedprocess',COMPRESS(storedprocess));
run;
```

LINKING BETWEEN STORED PROCESSES

Some Stored Processes we used need to directly link to others. Prompts and macrocode are used. In this case, we have created a stored process as a startup to all other stored processes. This allows the application to be dynamic. How did we do this? We have a special table called url. Table 1

This allows us to add any report or table, give it a title to show the user, and the stored process that will be linked. This also make the program simple.

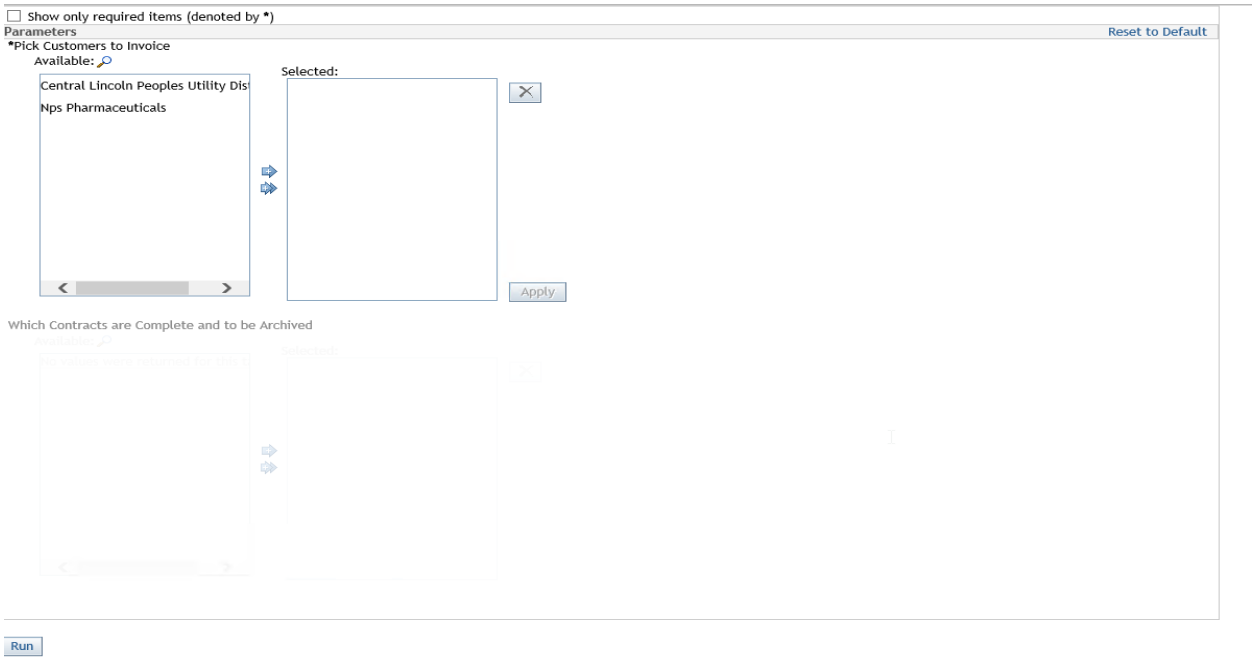
```
%macro maintracking;
  data _null_; set tracking.url(where=(TYPE="&TYPE" and TITLE="&TITLE"));
     call symput('storedprocess',COMPRESS(storedprocess));
run;
  data _null_;
   file _webout;
   put '<script type="text/javascript">';
  put
%unquote(%str(%'self.location="http://sdtdev2.solutiondesignteam.com/SAS®StoredProcess/guest?_action=form,properties,execute,nobanner,ne
wwindow
%nrstr(&_program)=%2FApplications%2FSolutionDesignTeam%2FTracking%2FStoredProcesses%2F&storedprocess"%'));
                    put '</script>';
                    run;
%mend;

%maintracking;
```

We used the data _null_ code with a file _webout statement. _webout is the way to push web output to the browser.

```
  data _null_;
   file _webout;
   put '<script type="text/javascript">';
   put
%unquote(%str(%'self.location="http://sdtdev2.solutiondesignteam.com/SAS®StoredProcess/guest?_action=form,properties,execute,nobanner,ne
wwindow
%nrstr(&_program)=%2FApplications%2FSolutionDesignTeam%2FTracking%2FStoredProcesses%2F&storedprocess"%'));
  put '</script>';
 run;
```

This small amount of code allows the user to pick any report or table option from the prompt list, and will be directed to the stored process associated with that stored process.

The link will go directly to the stored process from Type-R Title-Invoice Report and call InvoiceReport Stored Process.



TYPE	TITLE	STOREDPROCESS
R	Account Report	AccountReport
R	Invoice Custome...	InvoiceReport
R	Posting Report	PostingReport
R	Detailed Project...	DetailProjectReport
T	1-Company/Acco...	Accounts
T	2-Projects	Projects
T	3-Work Orders	WorkOrders
T	4-Project Manag...	ProjMgrs
T	5-Account Execu...	AEs
T	6-Consultants	Consultants
T	7-Activities	Activities
T	8-Postings	Posting

Table 1 URL Table:

TYPE	TITLE	STOREDPROCESS
R	Account Report	AccountReport
R	Invoice Custome...	InvoiceReport
R	Posting Report	PostingReport
R	Detailed Project...	DetailProjectReport
T	1-Company/Acco...	Accounts
T	2-Projects	Projects
T	3-Work Orders	WorkOrders
T	4-Project Manag...	ProjMgrs
T	5-Account Execu...	AEs
T	6-Consultants	Consultants
T	7-Activities	Activities
T	8-Postings	Posting

Table 1. URL table to call Stored Processes

HIDDEN PROMPTS

Since the customer continually adds contacts to their database, for historical purposes, all data is kept. However, if every customer that has been contracted was shown, the list would be cumbersome. By using a prompt that is hidden with a single value of Y for Active, and then making the customer field dependent on that field, that allows only those customer that have current contracts to show. When a customer is new/modified, it will only show those customers that are active. This is also true when a project is created. Only Active Customers will show. A stored process was created to modify the active status to N when the project for that customer was completed.

PROMPT GROUPINGS

Some of the needed stored processes have been built to complete different tasks. An example would be:
Adding or modifying a tables rows
Activate/De-Activate a customer
Create a customer report
Each of these has independent prompting.
Adding will have prompts for all the columns needed in the table
Activate will have list of Customers and their status
Report might allow a report of individual Customers or the entire table.
This is accomplished with
Standard Groups Prompts
Selection Groups Prompts
Selection-Dependent Group Prompts
All other type prompts

OTHER PROMPT USES IN THIS APPLICATION

Creating Key Number

All the tables have a key. In four of the main tables those keys are created by the application to assure they are unique and sequenced. Macros are used for this.

Where clause

Using multiple selection prompts creates array name macro values. These macro variables can be extrapolated to create where clauses for table access.

Titles

Creating Data _NULL_ statements with macros can create Titles to help in reporting needs.

Emailing

As the tracking system is used for consultant time posting, management needs to know when hours are approaching the maximum allowed for the project. Using macro code, when the hours get to a certain point it triggers an email to persons that need to be aware of those hours and make decisions accordingly.

Using ODS to create output and pdf files

Many reports are created with this application. In most cases since this application is used via the web, HTML reports are created with a variety of style sheets for different backgounds, etc.

The application also creates timesheets and Invoices. These need verified and then emailed to people outside our domain. One Stored Process generates the reports and then stores via United Naming Convention mapping PDF files.

CONCLUSION

Although we can only show a small part of the Tracking Application, it is hoped that we were able to show some of the incredible functionality that can be used with a Stored Process. SAS® Stored Processes provide a way to implement client/server distributed applications in SAS®. With Enterprise Guide it is relatively straightforward to convert existing code to a Stored Process, which should pay immediate benefits for programs that are run repeatedly. Of course there is a significant initial investment required to set up the metadata correctly and to configure the Stored Process server, but once this has been done, and it only needs to happen once and end users can, given the proper authorization, create applications which are reliable, useful, and truly powerful.
Recommended Reading

- The 50 Keys to Learning SAS® Stored Processes: Must have guide for SAS® Developers
- Solution Design Team Training –We can work directly with your team using your data

CONTACT INFORMATION

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

Name:	Dave Mitchell
Enterprise:	Solution Design Team
Address:	6001 W Alder Ave.
City, State ZIP:	Littleton, CO 80128
Work Phone:	303.979.4122
Fax:	303.973.9690
E-mail:	trojan@solutiondesignteam.com
Web:	http://www.solutiondesignteam.com/