Spring 2012 OASUS Questions and Answers

The following answers are provided to the benefit of the OASUS Users Group and are not meant to replace SAS Technical Support. Also, an Enterprise Guide project is provided as a companion to this document. The project is available on the OASUS web site (www.oasus.ca) under the Fall 2012 meeting since this is when these answers have been presented formally.

Question 1 : With PROC SORT, you can sort by descending/ascending values of a variable. Is there something similar with PROC SQL?

The short answer is yes, thanks to the ORDER BY clause of the SELECT statement. By default, the order is ascending unless specified otherwise. This order can also be specific for each variable that are listed in the ORDER BY clause.

```
/* Sorting with different order specifications in PROC SQL*/
title "Sort with PROC SQL";

proc sql number ;
      select * from sashelp.class order by name;
      select * from sashelp.class order by name desc;
      select * from sashelp.class order by sex desc,name ;
      select * from sashelp.class order by sex asc,name desc;
quit;

/* Equivalent in PROC SORT */
title "Sort with PROC SORT";

proc sort data = sashelp.class out = sort_out1;
      by name;
run;

proc sort data = sashelp.class out = sort_out2;
      by descending name;
run;

proc sort data = sashelp.class out = sort_out3;
      by  descending sex   name;
run;

proc sort data = sashelp.class out = sort_out4;
      by  sex  descending name;
run;
```

The capabilities of the sort routine in PROC SQL are very similar to PROC SORT but not identical. Certain features are more trivial in PROC SORT but accessible in PROC SQL if you know the SQL language reasonably well (a good example of that is the NODUPKEY facility).

```
/* Sort and eliminate records with duplicate keys */

/* Using PROC SORT (easy!) */
proc sort data = sashelp.class out = sort_out nodupkey;
     by age sex;
run;

/* Using PROC SQL (not so trivial) */
proc sql;
     create table sql_out(drop = count)
          as select *,monotonic() as count from sashelp.class
               group by age,sex
               having min(count) = count
               order by age,sex;
quit;
```

Question 2 : Can you explain the usefulness of the ERROABEND option?

The ERRORABEND option allows you to end the SAS process when an error is encountered. This is mostly useful in production systems where you don't want to continue processing should any error occurred in a SAS job. If not specified, SAS will either attempt to "recover" the error and continue processing data. If the error is not so severe, the rest of the job may actually run to completion. However, if the SYNTAXCHECK option is turned on, the rest of the job will run instead in syntax check mode!

Here is an example that uses the ERRORABEND option.

```
/* Setting the ERRORABEND option will force SAS */
/* to stop all processing when an error is encountered*/
options errorabend ;
/* DATA step with a syntax error */
data _null_;

     a==2;

run;
/* Valid DATA step. Will not run if the ERRORABEND option is
turned on */
data _null_;

     putlog 'Hello World';

run;
```

Question 3 : When will SAS EG support MS 2007 Excel (XSLX)? How can I import Excel 2007 spreadsheets with more than 255 columns?

In EG 5.1, the Excel XLSX format is fully supported via the EG import/export facility which uses native Windows API calls. In EG 4.3, the “out-of-the-box” wizard cannot export (but can import!) XLSX spreadsheets with more than 255 columns. If you attempt to do so, the wizard will fail. As an alternative, you can download a custom task from SAS support called “Export to Microsoft Excel 2007/2010” (http://support.sas.com/kb/41/132.html).

If you want to read and/or write XLSX formatted workbook files through code, you have to use the SAS/ACCESS Interface to PC Files. This interface provides LIBNAME engines for various PC Files formats as well as IMPORT and EXPORT procedures. In SAS 9.2, only the first 255 columns of an XLSX workbook file can be imported and you have to use the EXCEL DBMS designation in order to do so.

```
PROC IMPORT OUT= WORK.test2
            DATAFILE= "F:\Spring 2012\large.xlsx"
            DBMS=EXCEL REPLACE;
     RANGE="Sheet1$";
RUN;
```

Starting with SAS 9.3M1, you can read/or write XLSX workbook files beyond 255 columns by using the XLSX DBMS designation (under SAS 9.3, the EXCEL designation yields the same behavior in SAS 9.2). Furthermore, SAS will read up to the limits that the XLSX format support (16,384 columns and 1,048,576 rows)

The following invocation of the IMPORT procedure will read up to 16,384 columns.

```
PROC IMPORT OUT= WORK.test2
            DATAFILE= "F:\Spring 2012\large.xlsx"
            DBMS=XLSX REPLACE;
     RANGE="Sheet1$";
RUN;
```

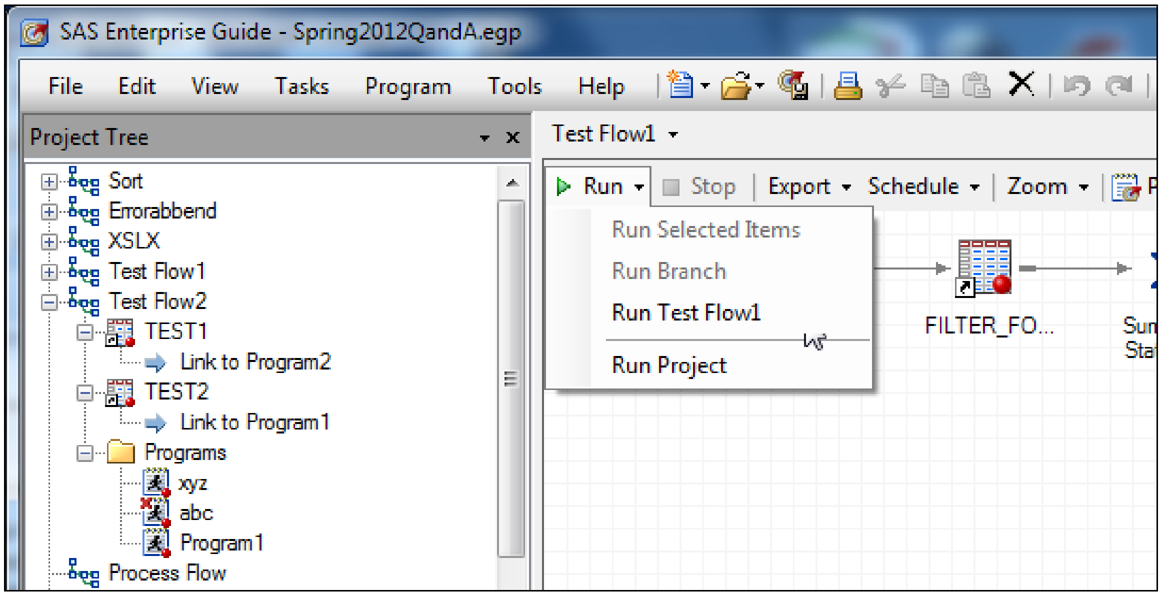
If you are not running 9.3M1 or better, you have to use workarounds. Before you start an import, you can save the Excel file as a csv file and then import it into SAS. For exporting, there are at least two publically available macros that can help you deal with datasets with more than 255 columns:

1. *Split255*: This macro is available from the SAS support website at http://support.sas.com/kb/36/904.html. It divides up your SAS dataset into different sheets consisting of 255 characters. If the number of columns in the SAS dataset is not a multiple of 255, this macro puts the remainder of the columns over and above the multiple of 255 into a separate worksheet.

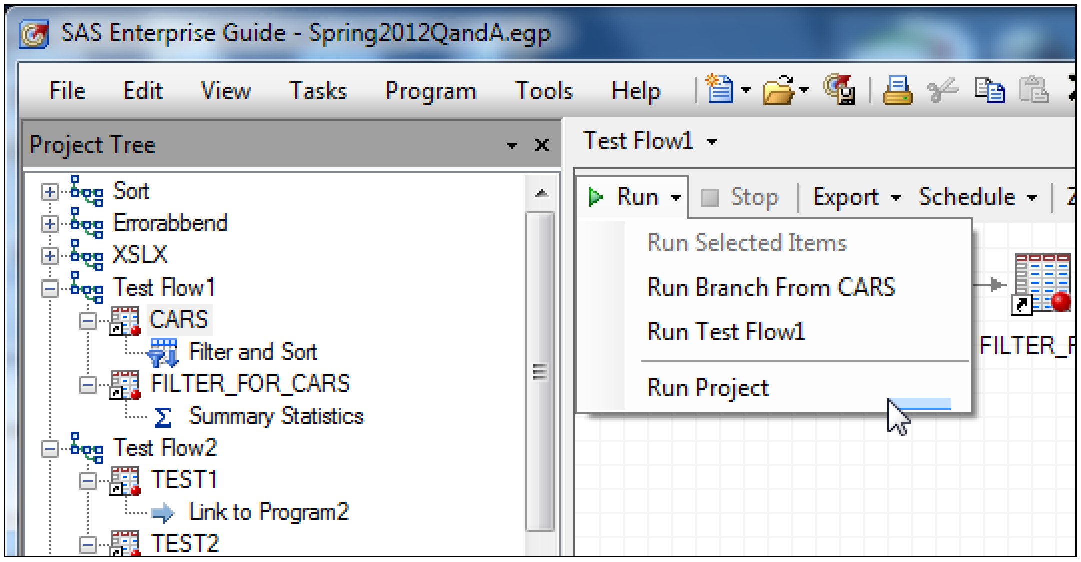
2. *ExcelLoad*: This macro was found on the SAS communities website at https://communities.sas.com/docs/DOC-1260. It is an alternative to the Split255 macro and copies the whole dataset to a single sheet.

Question 4 : In EG, when a project has multiple process flows, how do you run a subset?

It is fairly easy to run one process flow; you just need to specify that you want to run the entire flow when it is opened.



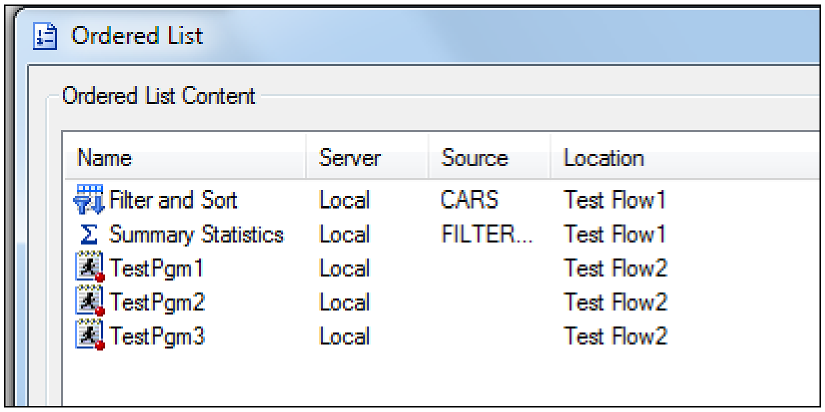
You can also run the entire EG project which run all the process flows one after the other in the order that they are specified in the project tree.



Although there is no direct way of running a subset of the project flows, three approaches are possible to achieve a similar outcome.

1. Ordered Lists

   You can create a list of tasks and programs in the current project that can be run in the order that you choose. You can select objects from more than one process flow. You have to be very careful when building an ordered lists since these are “tasks oriented”. You have to make sure that you include all the tasks from the desired process flows and this in the proper order.



Ordered List

Ordered List Content

Name	Server	Source	Location
Filter and Sort	Local	CARS	Test Flow1
Summary Statistics	Local	FILTER...	Test Flow1
TestPgm1	Local		Test Flow2
TestPgm2	Local		Test Flow2
TestPgm3	Local		Test Flow2

2. VB scripts

   You can run any EG project in “batch mode” by using the Visual Basic Script language. SAS Enterprise Guide can automatically generate a basic automation script to run the entire project or a single process flow and integrate with the Windows scheduler. You can adapt a generated script to run multiple process flows not just one.

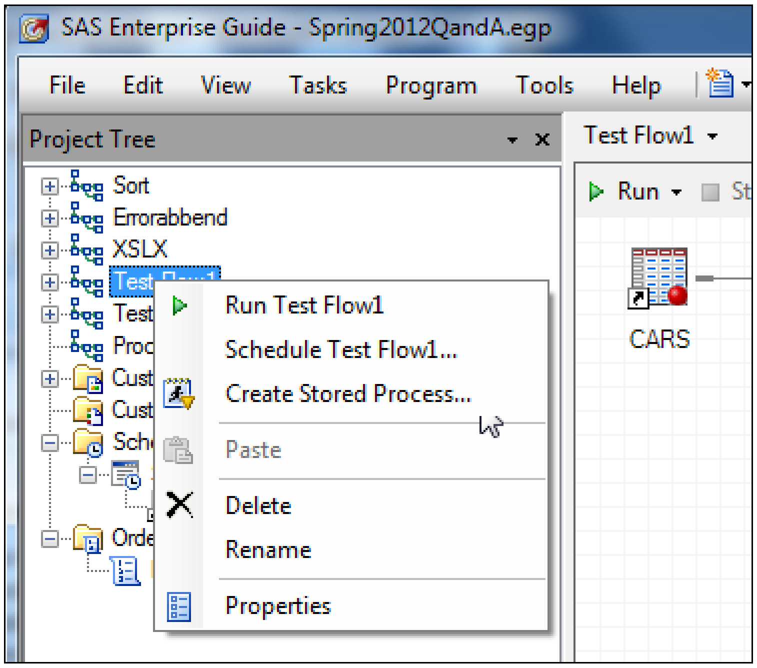
```
………
Dim prjName      ' As String
Dim prjObject    ' As SASEGuide.Project
Dim containerName      ' As String
Dim containerObject    ' As SASEGuide.Container
Dim containerColl      ' As SASEGuide.ContainerCollection

prjName = "E:\Spring 2012\Spring2012QandA.egp" ' Project Name
containerName = "Test Flow1" ' Container Name

Set app = CreateObject("SASEGObjectModel.Application.4.3")
If Checkerror("CreateObject") = True Then
Exit Sub
End If
………
```

3. Stored Processes

A stored process is a SAS program that is stored on a server and can be executed as required by requesting applications. EG allows you to very easily turn an entire process into a stored process. So, once all the process flows of interest have been turned into a stored process, you can simply create an ordered list or even include them into a new process flow! Running them as a group becomes very easy. Note that you need access to a SAS platform to use and create stored processes.



References

SAS Global Forum 2012, Paper 298-2012, Not Just for Scheduling: Doing More with SAS® Enterprise Guide® Automation, Chris Hemedinger, SAS Institute Inc, Cary, NC.