Questions & Answers OASUS Fall 2005 General Meeting

Question #1: Within Excel using SAS Addin for MS-Office, can I do ANOVA or other SAS tasks accessing the Master SAS data table.

Answer: YES is the correct answer. In fact, all the analytics are done by the SAS server accessing the Master SAS table. The Excel Window is only used to browse a SAS table and is limited to 65536 rows (limit of Excel). However, you can set a filter on a SAS table to create a subset and run an analysis against that subset. To set the active SAS table, use the Active Data item under the SAS menu.

Question #2: How to export dataset w SAS formats to Excel?

Answer:

When you export, many of the variable's attributes can be sent to excel. At times this is automatic. For example as you use the SAS Wizard, the labels and formats follow the data. But you need to look at the default wizard settings. At other times, as you proc export the options for this proc may not be the same as other approaches. With Proc export (see an example of one type of code in PowerPoint slides presented by Laki Kourakis) you may find yourself turning an option on (or off) by coding it.

Another approach is to use a data step and output the data using the Excel engine and transforming the data as required:

```
libname myxls "C:\temp\test1.xls";

data myxls.yves7;
  set sashelp.air;
  date2= put(date,MONYY.);
run;
```

Another way of exporting to Excel is by using ODS and HTML. With this approach, in order to assign an Excel format, you can use the proprietary CSS attribute mso-number-format. The SUGI paper #52-28 by Vincent DelGobbo ("A Beginner's Guide to incorporating SAS Output in Microsoft Office Applications") explains this technique (and others) quite well.

Question #3: How to export a graph after creating it with PROC GPLOT. I was able to export it in a JPEG format, but the resolution is very poor. Is there a way to improve the resolution? I would like to use the graph in a Word document.

Answer:

GPLOT is one of the procedures available in SAS/GRAPH. SAS/GRAPH is the data visualization and presentation (graphics) component of the SAS System. SAS/GRAPH allows the user to control almost any aspect of a graph that it generates. To improve the resolution, you can use the following SAS/GRAPH options (with the goptions statement):

```
GSFNAME=TEST    -> fileref that point to the jpeg file
GSFMODE=REPLACE  -> if file exists, replace
DEVICE=JPEG      -> you want to export to the jpeg format
XPIXELS=1500 XMAX=5 -> 1500 pixels by 5 inches = 300 dpi
YPIXELS=1500 YMAX=5 -> 1500 pixels by 5 inches = 300 dpi
```

Question #4: I've downloaded several, fairly complex SQL tables. Can I bring them into SAS in a way that recognizes the relationship between the tables?

Answer:

When downloading SQL tables into a regular a SAS session, you lose those relationships. However, products such as ETL Studio and Enterprise Guide will recognize some relationships automatically.

Question #5: What do you do when you're running out of memory in a data step or a proc (proc freq for example)?

Answer:

There are really two sets of solutions for this kind of problem: you increase the amount of memory available for SAS and/or you reduce the memory requirements for the task.

To maximise memory available to SAS on Windows platform (the principles however are applicable to other operating systems):

- Other software should be shut down and not left idling on the task bar. Any program that is open will be using memory, both real and virtual.
- On a Windows PC, SAS will try to use all the memory that is available. When there is not enough for a very demanding job, one can try increasing the virtual memory setting found in Control Panel --> System --> Advanced tab --> Performance Options. One needs administrator privileges to do this.
- Cleaning up the temp files area can take pressure off the hard drive when you want to increase swap space. Previously crashed SAS jobs can leave a lot of junk in temp.
- Make sure that the system option MEMSIZE is set to its default of 0 which means that there is no limit except the operating system limit.

To reduce the memory requirements for a given task, a good approach is to partioned the problem. Each smaller problem is then solved separately bringing down the memory requirements at any given time. A good example of this approach is the use of BY group processing with PROC FREQ. PROC FREQ stores all statistics in memory for each combination of the variables provided in the TABLES statement.

So,

```
      PROC FREQ DATA=SALES;
      TABLES STATE*SALESMAN*YEAR:

Will require much more memory (but less CPU ) than

      PROC FREQ DATA=SALES;
      TABLES SALESMAN*YEAR:
      BY STATE;
```

Question #6: How to view data using SAS formats in SAS IOM Provider using VB.Net and ADO.Net?

Answer:

Within VB.NET, you cannot use ADO.NET directly because it does not support yet SAS formatted data. Here's an excerpt from a SAS Tech Support document (you can find this document at http://support.sas.com/ctx/samples/index.jsp) explaining how to do it using ADODB:

It isn't possible to access formatted SAS data using the OleDbDataAdapter as described in the previous section. You must use ADODB in order to retrieve formatted SAS data. This is achieved by using the built-in COM Interop assembly for ADODB. This technique for accessing SAS data is demonstrated in the method GetFormattedSasData() in the file SampleUtils.cs.

The code to retrieve a workspace from the ObjectPool is the same as in the previous section. The differences here are in the way the data is retrieved using ADODB instead of using the OleDbDataAdapter.

First, the required ADODB objects are created. The adoConnection object is opened using the default connection string for the IOM Provider. The adoCommand object is then configured to use the new connection and the SQL command that was passed into GetFormattedSasData(). The custom Recordset property "SAS Formats" is set to "_ALL_" to enable formatting of all columns with their default SAS format. The adoRecordset is then opened using the adoCommand object.

In order to use this data for data binding in an ASP.NET application, the data must be in an ADO.NET DataSet. In order to move data from an ADODB.Recordset object into an ADO.NET DataSet, the OleDbDataAdapter provides another Fill method which will fill a DataSet with the data contained in a Recordset.

This technique is a bit slower than using ADO.NET party because the SAS server has to apply the formats to the data. Also, more speed and memory are consumed because two objects have to be created in memory to represent the same data.

For this reason, it is recommended that you avoid using ADODB unless SAS formats are necessary. If you are dealing with formatted SAS data that takes some time to retrieve, then you might also need to look into caching this data in order to enhance performance of your application. For an example of using the built in Cache object, see the method GetTables() in ViewSasData.aspx.cs.

```
//Get the formatted data using ADODB...
//  Then put the ADODB RecordSet into ADO.NET
ADODB.Connection adoConnection = new ADODB.ConnectionClass();
ADODB.Recordset adoRecordset = new ADODB.Recordset();
ADODB.Command adoCommand = new ADODB.CommandClass();

adoConnection.Open("provider=sas.iomprovider.1; SAS Workspace ID=" +
    sasWorkspace.UniqueIdentifier, "", "", 0);

adoCommand.ActiveConnection = adoConnection;
adoCommand.CommandText = selectCommandText;
adoCommand.Properties["SAS Formats"].Value = "_ALL_";
adoCommand.CommandType = ADODB.CommandTypeEnum.adCmdText;

adoRecordset.Open(adoCommand, System.Reflection.Missing.Value,
    ADODB.CursorTypeEnum.adOpenDynamic, ADODB.LockTypeEnum.adLockReadOnly, 1);

//Fill an ADO.NET DataSet using the ADODB Recordset
System.Data.OleDb.OleDbDataAdapter sasDataAdapter =
    new System.Data.OleDb.OleDbDataAdapter();

System.Data.DataSet sasData = new System.Data.DataSet();
sasDataAdapter.Fill(sasData, adoRecordset, "sasdata");
```