Paper 122-27

A New User's Journey in Using PROC IMPORT and ODS:

An Application in the Electricity Market

David Wang, Public Utilities Commission of Ohio, Columbus, OH
M. Azharul Islam, Ohio Department of Aging, Columbus, OH

ABSTRACT

The Division of Market Monitoring & Assessment of the Public Utilities Commission of Ohio is responsible for monitoring the Ohio electricity market during this development period of deregulation of generation services. This paper describes a process in which data are taken from a Microsoft Access database and how SAS-PC is used to generate the statistics for monitoring the electric market. Additionally, the problems encountered and the lessons learned during this process are highlighted.

It has been observed that utilizing PROC IMPORT and ODS gives the user the ability to interface with a host of PC database software and to produce customized results. When we combine these tools with the power of the macro language, we are able to generate a user-friendly program that allows the user to produce the desired output. The audience that the paper is intended for is the beginner user of PROC IMPORT, the SAS Macro language and Output Delivery System (ODS).

The user will need a license for the following products:
SAS/ACCESS® for PC formats;
SAS/STAT®;
BASE SAS® (ODS and SAS Macro language).

INTRODUCTION

Since the early 1900's, electricity has been a regulated commodity. The electric distribution utility (EDU) provided the customer a bundled product that includes generation, transmission, and distribution. As several components of the telephone and gas industries became unregulated, it was inevitable that the electricity market would be deregulated. In January 2001, Ohio began its development period in which the generation portion of the electricity business would be deregulated. In order to make sure that all of the companies had a fair opportunity to compete, there was a need to monitor the development of competition and to check for any abuse of market power in the marketplace.

Forms developed in an Excel format were developed and posted on the Public Utilities Commission's web site. The participants in the Ohio electric market were responsible for downloading the forms, filling in the data, and e-mailing us the completed forms. With the use of some Excel macros and Access macros, the data were stored in an Access database. Traditionally, we were using SAS-PC version 6.12 for analysis.

However, after attending SUGI 26 in Long Beach, we were enlightened with the new enhancements of SAS version 8.02. We were introduced with the existence of SAS/ACCESS for PC formats and ODS.

THE FOCUS OF OUR PROJECT

Our goal was to take the data from an Access database and to generate quarterly statistics sorted by EDU service area and by type of provider (CRES Providers – competitive retail electric service providers and Aggregators). We were faced with the decision either to:

- Generate reports within Access which is limited in scope of the type of statistics that it can produce,
- Export to Excel tables and to generate the statistics using spreadsheets or
- Write a SAS program.

Some of the challenges that we faced were the following:

- The number of companies varied from quarter to quarter;
- The data from each form were structured differently;
- The format of the data varied from company to company;
- We were not sure which statistics that we would include in the future;
- Each quarter, companies would e-mail data submissions that varied from zero to three months of data.

RESULTS

We decided that SAS would give us the most flexibility in dealing with a dynamic situation in which the status of each company was transient and the type of statistics needed was always changing. With the use of PROC IMPORT and a few SAS macros, we developed a process that addressed all of the problems listed above. Also, we were able to control the output that we wish to view and allow the user to choose which data sets they wish to use.

The sample syntax that we used to import the data was as follows:

```
PROC IMPORT TABLE = "1 – 1 General Info"
  OUT = mm11
  DBMS = Access
  REPLACE;
  DATABASE = "c:\mm data base\mm database
                q&qrt y&year..mdb";
RUN;
```

Explanation

OUT = {name of output file – our example stores data in a temporary SAS file}
TABLE = {the name of spreadsheet in your excel book}
DBMS = {type of application – in this example, we used Microsoft Access}
REPLACE; → allows SAS to overwrite in the file specified in the "OUT =" statement
DATABASE = {source of the data}

We used the following naming convention to categorize our database:
"MM DATA BASE QxYzzzz.MDB"
where x = quarter number (01,02,03,04) and zzzz = 4-digit year.

For example, the database that contained second quarter data for 2001 would be labeled as
MM DATA BASE Q2Y2001.MDB. This allowed us to use the SAS® Macro language to identify which database we wished to

use.

ODS OUTPUT

In the past, we use to submit our SAS programs to generate our statistics and then export our results in ASCII format. Then we would use either Freelance or Excel to generate some nice looking tables or graphs. With ODS, we are now able to generate tables and graphs that are ready to be presented without having to use other PC software.

One of our resources that we used was a book titled Output Delivery System - The Basics by Lauren E. Haworth. This was a tremendous help in getting us started. We also thumbed through the SUGI26 proceedings for papers that dealt with ODS. Finally, there were a number of resources listed at www.sas.com. We listed in the back of this paper the references that were most helpful in developing our style template. Generating a PDF file was pretty easy, but trying to get the final output was definitely more of a challenge. The sample syntax that we used were as follows:

```
:
:
PROC TEMPLATE;
     DEFINE STYLE STYLES.test3;
     PARENT=STYLES.PRINTER;

REPLACE FONTS /
     'TITLEFONT2'=("ARIAL",9pt,BOLD)
     'TITLEFONT'=("HELVETICA",9pt,BOLD)
     'STRONGFONT'=("HELVETICA",9pt,BOLD)
     'EMPHASISFONT'=("HELVETICA",9pt,BOLD)
     'FIXEDEMPHASISFONT'=("COURIER NEW,COURIER",6PT,ITALIC)
      'FIXEDSTRONGFONT'=("COURIER NEW,COURIER",6PT,BOLD)
      'BATCHFIXEDFONT'=("SAS MONOSPACE,COURIER
                         NEW,COURIER",6pt,BOLD)
     'FIXEDFONT'=("ARIAL,COURIER",8pt,BOLD)
     'HEADINGEMPHASISFONT'=("ARIAL,TIMES",10pt,BOLD ITALIC)
     'HEADINGFONT'=("ARIAL,HELVETICA,HELV",10pt,BOLD)
     'DOCFONT'=("ARIAL,HELVETICA,HELV",8pt);

REPLACE TABLE FROM OUTPUT /
     BACKGROUND=_UNDEF_
     RULES=NONE
     FRAME=VOID
     CELLPADDING=2PT
     CELLSPACING=1
     BORDERWIDTH=2
     JUST=LEFT;

REPLACE DATA FROM DATA /
     JUST=CENTER;
     REPLACE COLOR_LIST
      "Colors used in the default style" /
         'link'          = white
         'bgH'           = grayBB
         'fg'                    = black
         'bg'                    = white;
     END;
     :
     :
     ODS PDF FILE = 'e:\sugi27\sugi output.pdf';
     OPTIONS BYLINE;
     :
     :
     PROC PRINT;
     RUN;
     ODS PDF CLOSE;
     :
     PROC TEMPLATE;
      DELETE STYLES.test3;
     RUN;
```

Explanation

- DEFINE STYLE STYLES.TEST3 → creates a style called TEST3 in a store called STYLES (the default location).
- PARENT = STYLES.PRINTER → creates a default set of parameters that SAS will use to create the PDF file from a style called PRINTER;
- ODS PDF FILE= → {the name and location of the output PDF file};
- ODS PDF CLOSE; → closes the ODS statement;
- OPTIONS BYLINE; → prints the output generated from a BY statement continuously instead of printing each segment on separate pages (e.g., if you printed by EDU service area and there are eight service areas, the NOBYLINE option would print the output on eight pages while the BYLINE option would print it on 2 pages).
- DELETE STYLES.test3 → delete the newly created style from the default store.

Our basic structure for our SAS program was one main macro that included a number of nested macros & %EVAL statements, a %INCLUDE statement and a few %LET placed outside of the main macro. We used the %INCLUDE to hide some of the statements which was more for esthetic reasons. The series of %LET statements were placed at the top of the program to let the user decide which output to view and which data sets to work with. The sample syntax were as follows:

```
%LET year       = 2001;
%LET qrt        = 02;
%LET mm12       = YES;

%MACRO all;
:
DATABASE = "c:\mm data base\mm data base q&qrt y&year..mdb";
:
%INCLUDE "\\t2000\roaming users$\wang\\mm statements.sas";
:

%MACRO mm12;
:
%MEND mm12;
:
:
%IF %EVAL(&mm12=YES) %THEN %DO;
      %mm12
%END;
:
%MEND all;
%all;
```

LESSONS LEARNED

Being novices with the new version of SAS-PC, we encountered a number of problems when trying to use PROC IMPORT. The first problem was an easy one to fix: what product would we need to have to read an Access database? We called the SAS representative and she told us that we needed the SAS/ACCESS® software for PC Formats installed on your PC in order to use IMPORT procedure. The second problem was a bit more difficult: what syntax do we need to be able to properly read the Access database? We were able to find information about importing Access databases under the SAS® on-line Help Menu, searching for the word IMPORT and under SAS® Procedures Guide. There were also examples on the SAS web site, but we still found it somewhat difficult knowing what options to choose. Finally, the last problem was to point to the right file that resided on the server. At first, we used Windows Explorer to obtain the drive designation, but the SAS® software was unable to locate the file. When you point to a file on a server, you must designate the server and share information and then the subdirectory that is

designated by Windows2000. An example is listed in the SAS code above.

After we were able to read the file, we encountered a different set of problems. We received a number of errors reading in the data from our Microsoft Access database. What we learned from importing Access were the following:

- Remember to insert an "_" if your variable name contains any spaces. For example, POWER MARKETER would be read by SAS as POWER_MARKETER.
- The variable name is not case sensitive but the value of a character variable is case sensitive so SAS reads Puco differently than PUCO.
- Be careful how the data is formatted in Access. It affects how SAS reads in the data.

Finally, outputting to a PDF format is a relatively new feature of the ODS. Most of the documentation and training emphasizes the HTML format that is much more extensive in syntax and options. Finding the necessary documentation for creating customized PDF will be a bit more challenging. The difficulties that we encountered in working with PDF output were as follows:

- In version 8.1, we sometimes generated output with a green background while using our own customized style definition. It seemed to appear when we tried to add a REPLACE COLOR_LIST statement and change some of the colors in the Template. We tried to replicate it in version 8.2, but were unable to duplicate the results.
- In version 8.1, whenever we used a custom definition and used the PARENT= statement, we had to include all of the parameters defined in FONTS section that was used in the Parent or else SAS issued an error in the SAS Log and the default style definition was used. In version 8.2, it appears that we no longer have to explicitly define all of the parameters in the FONTS section. SAS only issues a warning and it uses your custom style definition. Apparently, if you do not define a parameter, it uses the value defined by the Parent.
- In versions 8.1 and 8.2, whenever we use our customized style definition, SAS issues the following warning: "Could not find parent template: DATA". It does not seem to affect the output.
- When we ran certain parts of our SAS job, a black page was generated on the second page in the PDF file. Our solution was to break the output into two separate ODS output statements.
- When your procedures are called within your ODS statements, make sure that you include a RUN statement after your last procedure or else SAS will not output your PDF file.

CONCLUSION

PROC IMPORT and ODS are valuable tools that provides the user not only the ability to interface with other PC database software, but to produce an output that is comparable to many PC graphic packages. When we coupled these tools with the power of the macro language, we were able to generate a user-friendly program that enabled the user to limit the output that his or her program produced. The SAS macros gave us the capability of easily adding modules to include new statistical analyses that the user might wish to include in the future.

REFERENCES

Gupta, Sunil K. (2001), "Using Styles and Templates to Customize SAS® ODS Output", SUGI 26 paper 1.

Haworth, Lauren E., *Output Delivery System – The Basics*, Cary, NC: SAS Institute, Inc.

Hull, Bob (2001), "Now There is an Easy Way to Get to Word, Just Use PROC TEMPLATE, PROC REPORT, and ODS RTF", SUGI 26 paper 163.

McNeill, Sandy (2001), "Changes and Enhancements for ODS by Example", SUGI 26 paper 2.

Olinger, Chris, "ODS for Dummies" (2000), SUGI 25 paper 64.

SAS Institute, Inc. (1999), *The Complete Guide to the SAS® Output Delivery System, Version 8,* Cary, NC: SAS Institute, Inc.

SAS Institute Technical Help.

www.sas.com/rnd/base/topics/templateFAQ/Template_procedure.html
www.sas.com/rnd/base/topics/odsprinter/faq.html
www.sas.com/rnd/base/topics/templateFAQ/Template_rtf.html

CONTACT INFORMATION

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

David Wang
Public Utilities Commission of Ohio
180 East Broad Street
Columbus, OH, 43215
Work Phone: (614) 466-9164
Fax: (614) 752-8353
Email: david.wang@puc.state.oh.us