Paper 3834-2019

Automate in a Dash with SAS®

Time-Saving Techniques for Building Quality Improvement Dashboards

Shavonne J. Standifer, Truman Medical Center

ABSTRACT

Building programs that leverage the analytic and reporting powers of SAS® reduce the time to solution for critical tasks. This paper discusses, through example, how Base SAS® tools, such as FILENAME, MACROS, and ODS combined with the "built in scheduler" housed in SAS® Enterprise Guide can be used to automate the process between raw data to dashboard view quickly and efficiently.

This paper is divided into two main parts. In the first part, we discuss how to use scripting language to bring data from a file location into the SAS® environment, how to build programs that clean and subset the data, and how to transform the process with a MACRO. In the conclusion, we will discuss how to use SAS procedures and the ODS to transform the resulting data into a quality improvement dashboard view that can be set to automatically run and send to team members at a scheduled time.

INTRODUCTION

Due to rapid emerging technologies the automation of report distribution is needed in today's growing data world. Healthcare business analysts are expected to provide insights on data results in real-time. This helps clinicians and administrators have access to the data results needed to identify areas for quality improvement. This paper demonstrates how SAS Enterprise Guide® and Microsoft® Windows Task Scheduler work together to automate report distribution by email. Examples in this paper were developed with SAS® Enterprise Guide® 4.3

There is a lot of really good published information around the SAS® procedures, statements and routines mentioned in this paper. This paper seeks to highlight some of the key ways to automate your data project. The reader is encouraged to dive deeper into the details and several resources are listed at the close of this paper to assist you on your continued learning journey.

ON YOUR MARK, GET SET

YOUR ENVIRONMENT

Before you begin to design the flow of your automation project, it's important that you understand the main details of your technical environment. Working with a SAS® Administrator is recommended if this is outside of the scope of your interaction with SAS® on a daily basis. The examples in this paper were created using the windows operating system.

Here are some key technical details to consider:

1. Ensure that you have local administrative privileges on the machine that will run the project.
2. Ensure that you are able to access and use Windows Task Scheduler.
3. Ensure that you are able to execute VBScript programs independently of SAS Enterprise Guide®.
4. Ensure that metadata credentials are cached in your connection profile.
5. Ensure that your project and VBScript file are saved locally.

While understanding your technical environment is key, the success of project automation in SAS Enterprise Guide® depends greatly on how efficiently your programs interact with environments outside of SAS®.

USING THE FILENAME STATEMENT

The FILENAME statement is an efficient way to bring your data into SAS®. It provides SAS® programs with a detailed reference towards the location of your source data. It allows your programs to be more flexible because once a FILENAME has been defined in a session, you can use the filename at any time to refer back to location of your data source. Including the FILENAME statement in your automated projects helps to save time because if the location of your source data changes, you only have to update your program in one place.

The FILENAME statement is also a great data management tool because it becomes quite flexible in processing data based on the options that you specify surrounding the statement.

USING A PIPE WITH MS-DOS COMMAND FEATURES

A pipe allows you to run a program outside of the SAS system and redirect the programs input, output, and error messages back into SAS®. It functions as a virtual file that holds information needed for a project to run successfully. The windows features below can communicate between the windows operating system directory and a SAS® session.

'dir' is a command that returns a list of files from a directory.

'/b' is a command parameter that *only* returns the filenames in a directory.

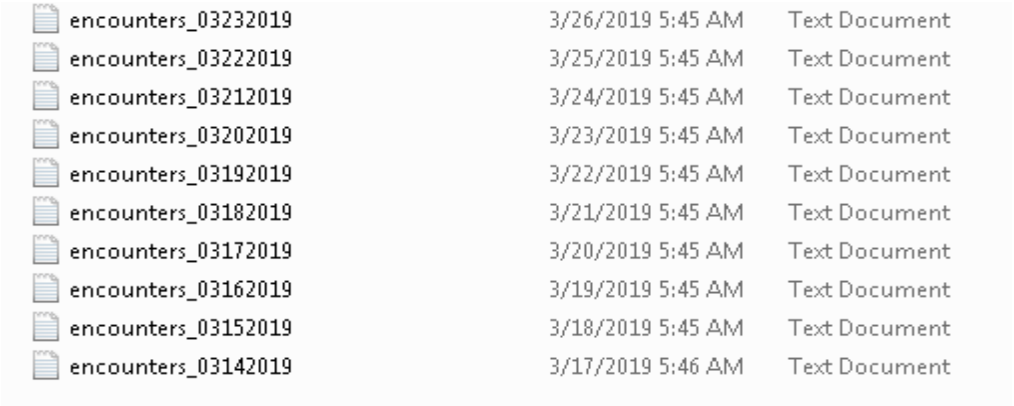
When these commands are used together, each file name in a directory can be brought into a SAS® dataset for further data cleansing and processing. These commands give you greater control over the way that your source data flows into your automated project.

BUSINESS EXAMPLE 1:

FIND THE MOST CURRENT FILE IN A DIRECTORY

This example includes a fictitious dataset that contains admission information at the encounter level. In this scenario the data is uploaded to a directory each day. The task here is to create a process that automatically identifies the name of the latest file uploaded into the directory and to use the latest file to perform daily processing.

Below shows a snapshot of the directory file structure:



We can combine the FILENAME statement with a PIPE, an asterisk wildcard (*) and windows command features to output the name of each file in the directory into a SAS® dataset.

Adding an asterisk wildcard into your directory path name allows you to read all .txt files that contain “encounters_” at the start of the filename into your project:

```
FILENAME dirname PIPE 'dir "\\encounters_*.txt" /b ';
```

When the code is executed a reference to your source data is created and you can include it in your project to perform further processing.

In the data step below, the file reference “dirname” is used to read in all of the names of each file in the directory.

```
DATA getfilenames;
  INFILE dirname lrecl=200 truncover;
  INPUT file_name $94.;
RUN;
```

The resulting dataset looks like this:

	file_name
1	encounters_03142019.txt
2	encounters_03152019.txt
3	encounters_03162019.txt
4	encounters_03172019.txt
5	encounters_03182019.txt
6	encounters_03192019.txt
7	encounters_03202019.txt
8	encounters_03212019.txt
9	encounters_03232019.txt
10	encounters_03242019.txt

Now that the file names are stored in a SAS dataset you are able to perform additional data sorting and cleaning. The following code uses a series of routine data techniques to identify the latest file in the directory.

```
DATA getfilenames2;
 SET Work.getfilenames;
 LENGTH name $100.;
 name=substr(file_name,1,11);
 date=substr(file_name,12,8);
 datevar=input(date,mmddyy10.);
 FORMAT datevar mmddyy10.;
 PROC SORT Data=Work.getfilenames2;
 BY datevar;
RUN;

DATA Work.curr_file;
 SET Work.getfilenames2;
 tday = '24MAR2019'D;
 FORMAT tday MMDDYY10.;
 IF datevar = tday THEN OUTPUT curr_file;
RUN;

DATA Work.curr_file2(KEEP=latest_file);
 SET curr_file;
  latest_file = file_name;
 CALL SYMPUT('latest_file',latest_file);
RUN;
```

Demonstrated in the code above, the CALL SYMPUT routine is used to store the name of the latest file in a macro variable for use with further processing later in the program. Below is what the resulting output looks like:

	latest_file
1	encounters_03242019.txt

Since CALL SYMPUT helped to store the name of the latest file uploaded into a macro variable, you can use the macro with an INFILE statement to only read in the specified file.

```
Data curr_munip;
 INFILE "\\&latest_file";
 INPUT
   enc                : ?? BEST8.
   pat_id             : ?? BEST8.
   admit_type         : $CHAR10.
   adate              : ?? MMDDYY10.
   ddate              : ?? MMDDYY10.
   hosp               : $CHAR4.;
RUN;
```

Storing the name of the latest file in a macro variable helps to make your code dynamic so that you can schedule your project code to run independently. Building flexible code is a powerful advantage when designing a data project for automation.

BUSINESS EXAMPLE 2:

READ IN ALL FILES FROM A DIRECTORY

This business scenario demonstrates how to bring all of the contents of a directory into your project based on file specifications that you choose. In this example, the FILENAME statement is included in the pathname of your directory.

```
DATA work.alldata;
 LENGTH getfilevar $ 256;
 INFILE "\\encounters_*.txt" FILENAME=getfilevar;

 getfilevar_final = getfilevar;

INPUT
   enc                : ?? BEST8.
   pat_id             : ?? BEST8.
   admit_type         : $CHAR10.
   adate              : ?? MMDDYY10.
   ddate              : ?? MMDDYY10.
   hosp               : $CHAR4.;

   IF pat_id ne . THEN OUTPUT;
RUN;
```

When the program is processed the resulting dataset includes all of the content within the files of your specified directory. Each observational row of encounter data will include a new variable that displays the actual file name of which the contents are sourced called “getfilevar_final.”

	admit_type	adate	ddate	hosp	enc	pat_id	getfilevar_final
1	Emergency	10/01/2018	10/01/2018	Loc1	74908	390	encounters_03232019.txt
2	Emergency	10/01/2018	10/01/2018	Loc1	74909	391	encounters_03232019.txt
3	Emergency	10/01/2018	10/01/2018	Loc1	74910	392	encounters_03232019.txt
4	Outpatient	10/01/2018	10/01/2018	Loc1	74911	393	encounters_03242019.txt

It’s important to note, the code provided in this example will read all of the contents of your directory into a SAS® dataset. This means that you will have several observations that will contain the header information from each file. It’s important to identify which observations from your input data will routinely produce invalid results so that it can be removed.

Cleaning the data at the import phase will save you time and ensure your data is both accurate and in the shape that you need to prepare for a dashboard view. In the example above we omitted observations where pat_id are empty as those were invalid. This example provides a quick way to read in all the data from your directory into your project so that you can perform additional processing.

The next section demonstrates how to transforming the data that you have brought into your project into a reporting view that you can set on auto pilot.

PART 2: GO

A huge part of quality improvement depends on how well you are able measure your data. Distributing hospital data in an efficient way ensures that hospital care teams have the tools they need to optimize health outcomes. Data visualization helps to ease gaps of understanding in data and leads to better healthcare planning and results.

Depending on the need, many times there is a need for routine information to be viewed daily. Below demonstrates a way to quickly create customized reports and distribute them by e-mail .

BUSINESS EXAMPLE 3:

AUTO DISTRIBUTE DAILY VOLUME BY E-MAIL

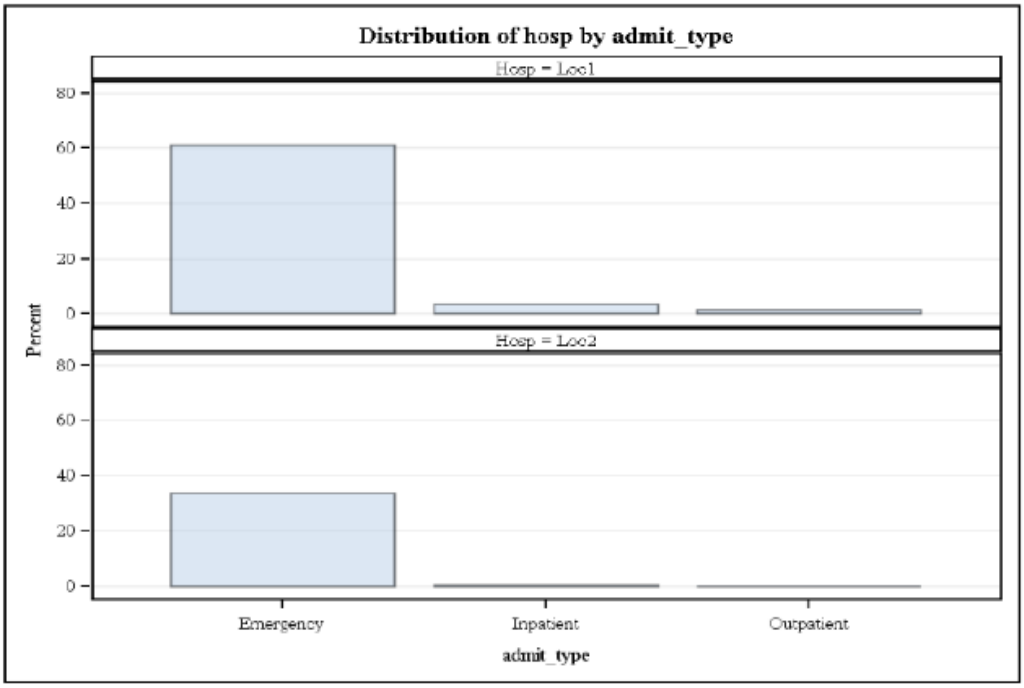
This example demonstrates a quick way to prepare a report that shows daily volume frequencies. The frequency procedure is useful in creating graphics in a simple view that presents the data point’s recipients need to see. You can combine a PLOT option with a TABLE statement to generate two different styles of graphs.

The code below demonstrates this approach:

```
ods graphics on;
ods pdf file="encounters.pdf";
proc freq data=work.alldata;
table hosp*admit_type / plots=freqplot(scale=percent);
run;
ods pdf close;
quit;
```

Once executed the following reports are the results:

Frequency Percent Row Pct Col Pct	Table of hosp by admit_type			
	admit_type			
hosp(Hosp)	Emergency	Inpatient	Outpatient	Total
Loc1	11473 61.30 92.73 64.60	661 3.53 5.34 94.29	239 1.28 1.93 94.09	12373 66.11
Loc2	6287 33.59 99.13 35.40	40 0.21 0.63 5.71	15 0.08 0.24 5.91	6342 33.89
Total	17760 94.90	701 3.75	254 1.36	18715 100.00



You can specify how the output of your graphs are handled via the Output Delivery System. The example above specified that the results are exported in pdf format.

Once your data visualization is output per your user needs, you can prepare to distribute the results throughout your organization via email.

REPORT DISTRIBUTION - E-MAILING FROM SAS®

Before you can send an e-mail from the data step, you must ensure that you have set up your communication configurations for your e-mail server. You can define your system options in your SAS®

configuration file or you can specify them in with your program. Saving your options in the configuration file saves time, however, if you can't access the file, you can directly specify in your program like below.

The following code shows an example of options that you can set.

```
EMAILSYS="SMTP"
EMAILID="sender information"
EMAILHOST="name of outgoing mail server"
EMAILPORT="port";
```

The program below demonstrates how to send the graphs created in business example 3 as a pdf attachment to a specifed recipiant.

```
FILENAME enctr EMAIL
SUBJECT = "Daily Encounters by Location"
Importance= "High"

To = "Shavonne.Standifer@tmcmed.org"
From = "Shavonne.Standifer@tmcmed.org"
attach = "encounters.pdf";


ods graphics on;
ods pdf file="encounters.pdf";
proc freq data=work.alldata;
table hosp*admit_type / plots=freqplot(scale=percent);
run;
ods pdf close;
quit;


data _null_;
file enctr;
put "Greetings,";
Put "This email shows daily encounter volumes.";
Put "Thanks,";
Put "Shavonne J. Standifer";
Run;
```

Utilizing this the e-mail feature in your automated projects can help you distribute information to others as well as yourself. It's a good practice to embed your reports with email communications that alert you if there is a fail or invalid data.

BUSINESS EXAMPLE 4:

AUTO DISTRIBUTE A REPORT THAT SHOWS MEASURE PERFORMANCE OVER TIME

We live in an open data world. In some cases, quality measure ratings concerning hospital quality are publicly reported so that patients and consumers can make informed decisions about care needs. Creating automated projects for care teams that monitor performance metrics help to identify areas of improvement quickly. Key Performance Indicator (KPI) charts are often used to help track quality improvement measures. The "traffic light" design helps reviewers to quickly identify the performance of a measure.

Once source data is brought into the SAS® environment, you can use the GKPI procedure to create a visualization of performance. The example below demonstrates how to design a project that automatically produces a KPI chart each time the source data is updated.

Below is a snapshot of the source data:

hospital	measure_name	q1	q2	q3	q4
Loc 1	Falls	3	2	1	0

You can use the DATA step to create a subset that filters based on location. Once filtered, you can use PROC SQL to store the results of each quarter into a macro for later use.

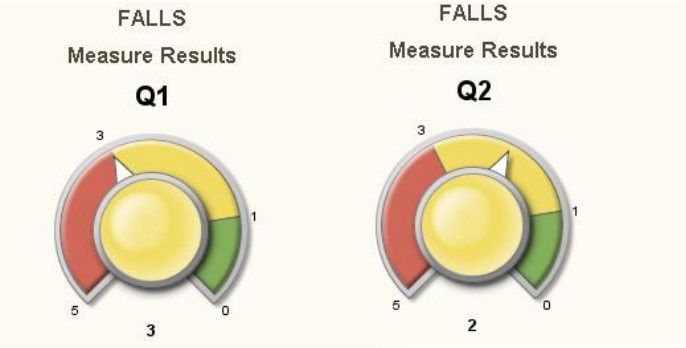
```
DATA loc1_falls;
 SET work.quality_measure_data;
 WHERE hospital = "Loc 1";
RUN;

PROC SQL noprint;
 SELECT q1 INTO:fallsQ1
 FROM work.loc1_falls;
quit;
PROC SQL noprint;
 SELECT q2 into:fallsQ2
 FROM work.loc1_falls;
QUIT;
```

Once the program executes, you can use the newly created macros with PROC GKPI to create a dial KPI chart. You can use the Output Delivery System to send the results to a pdf format. You can set options around the proceure to specify the look of your graph.

The code below demonstrates this approach:

```
 ods pdf file="fall_measures.pdf" startpage=no;
 ods graphics on;
 goptions reset=all rotate=landscape
 device=javaimg
 xpixels=240
 ypixels=200;
 title;
  Title1 'FALLS';
   Title2 'Measure Results';
PROC GKPI mode=raised;
 dial actual=&fallsq1 bounds=(5 3 1 0) / LABEL='Q1' lfont=(h=20 pt);
 dial actual=&fallsq2 bounds=(5 3 1 0) / LABEL='Q2' lfont=(h=20 pt);
RUN;
QUIT;
  ods pdf close;
   title;
RUN;
```

The resulting output is below:



The code in this example can be used to automatically generate e-mails to staff.

PROJECT AUTOMATION

SAS® Enterprise Guide is equipped with an intuitive interface that is designed to help you set up your automated project easily. Following the built in scheduling prompts will assist you in setting up your automation schedule. Once set up, a VBScript file that contains instructions for launching your project is automatically created. Many times the most challenging aspects to creating an automated project are adapting to the restraints of your technical environment. Here are some tips to consider when setting up your schedule.

- Be sure that your system options are set so that your project runs even if you are not logged in.
- Create an autoexec process flow and set your system options to automatically execute when opened.

CONCLUSION

This paper discussed useful techniques in data reporting automation that you can include in your reports and dashboards. The examples in this paper were created to help business analysts utilize the automation features of SAS® Enterprise Guide. The examples will give some ideas to use in your own custom reporting.

REFERENCES

Johnson, Christopher. 2015 “Integrating Microsoft® VBScript and SAS®”

Hemedinger, C. 2012. “Not Just for Scheduling: Doing More with SAS® Enterprise Guide Automation.” SAS Global Forum. SAS Institute Inc.

Schacherer, Chris. 2012. “The FILENAME Statement: Interacting with the world outside of SAS®”

Hunley, Chuck 2010. “SMTP E-Mail Access Method: Hints, Tips, and Tricks”

Tilanus, Erik W. 2008. “Sending E-mail from the DATA step”

ACKNOWLEDGMENTS

I’d like to thank Mellissa Peterson for her support and dedication to quality improvement outcomes.

I'd also like to thank Chris Hemedinger for his many works on SAS EG automation.

RECOMMENDED READING

- *Windows XP Under the Hood*
- *Carpenter's Guide to Innovative SAS® Techniques*

CONTACT INFORMATION

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

Shavonne J. Standifer
Truman Medical Center
Shavonne.Standifer@tmcmed.org