Paper 196-2013

SAS® Essentials: Maximize the Efficiency of Your Most Basic Users

Julie Kezik MS, Yale University; Melissa Hill MPH, Yale University

ABSTRACT

If programming and research assistants were taught SAS essentials, job efficiency could be maximized with the ability to use SAS as a tool to do their own preparatory work for assigned tasks. This paper summarizes a supplemental training program which teaches basic SAS programming skills to enable support staff to be more independent.

INTRODUCTION

The purpose of this seminar is not only to teach how to do things, but also to provide a toolkit of essentials for new and infrequent users. Spending time with your employees while they explore SAS is the most economical way to teach basic users the skills they need to complete daily tasks. The ability to navigate various windows allows your novice user to be knowledgeable about their data. As new data sets are created they can be checked and errors detected within reasonable time frames.

"You can't manage knowledge — nobody can. What you can do is to manage the environment in which knowledge can be created, discovered, captured, shared, distilled, validated, transferred, adopted, adapted and applied." (Collison, 24)

Often a SAS programmer is offered the help of an assistant. These employees have been selected and hired for their attention to detail and patience with meticulous tasks, however they are not necessarily fluent in SAS. The typical solution is for the programmer to do excessive and simplistic programming to provide the output necessary for whatever task the assistant will be handling. This cycle creates extra work for the programmer and limits the independence and effectiveness. Taking the time to teach your employees the SAS Essentials will maximize job efficiency and enable support staff to be more self-sufficient.

Course Purpose and Outline (*A secondary introduction*)

The SAS slogan is 'the power to know.' When working in SAS it is important to remember that there are many 'right' ways to complete a task. SAS programming is a creative and iterative process designed to empower the user. One's personal style evolves over time. How to solve a data mystery or accomplish a data driven task is ultimately an independent decision made by the user.

I. *Navigation: Explanation of tabs which appear upon opening the product*
   a. *Editor*
   b. *Log*
   c. *Results*
   d. *Explorer*
   e. *View Table*

II. *Import Procedures: Bringing External data into a SAS dataset*
   a. *Import Wizard*
   b. *Alternatives in JMP®*

III. *Data Exploration: Viewing the dataset and investigating column names and labels*
   a. *Using the Explorer*
   b. *PROC CONTENTS*

IV. *Data Step: Written by the user to create and alter data sets*
   a. *Two types of datasets: temporary and permanent*
   b. *Storage of datasets*

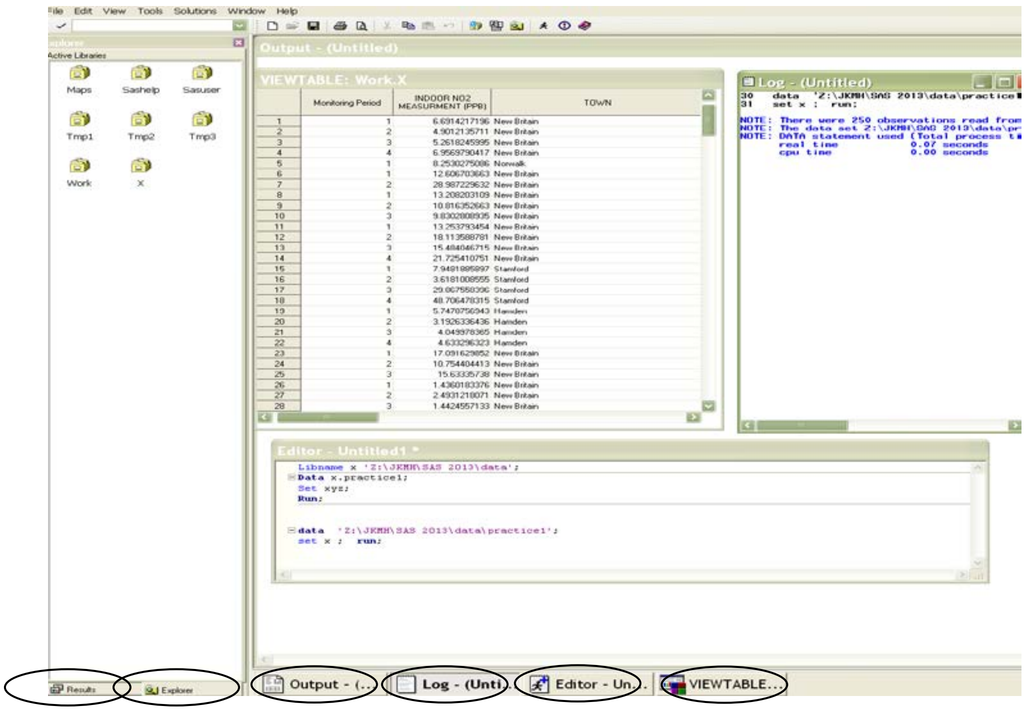
V. *Reading the log: Indicates if the program is running well*
   a. *Note*
   b. *Error*
   c. *Warning*

VI. *Procedures: Allow the user to manipulate and view their data in many ways*
   a. *PROC SORT*
   b. *PROC PRINT*
   c. *PROC FREQ*
   d. *PROC MEANS*

THE BASICS

Navigation:

Any programmer, novice or not, is ultimately more efficient when able to utilize the simple tools SAS provides upon opening the product. The editor window is where syntax is written. The log indicates how the program is running and if there are errors. Results and output tabs include any output requested. The explorer window contains organized folders of all the data. By double clicking a dataset from within the explorer tab, the viewtable tab appears and the user now can view the data table in the SAS window.



	Monitoring Period	INDOOR NO2 MEASURMENT (PPB)	TOWN
1	1	6.6914217196	New Britain
2	2	4.9012135711	New Britain
3	3	5.2618245995	New Britain
4	4	6.9569790417	New Britain
5	1	8.2530275086	Norwalk
6	1	12.606703663	New Britain
7	2	28.987229632	New Britain
8	1	13.208203109	New Britain
9	2	10.816352663	New Britain
10	3	9.8302808935	New Britain
11	1	13.253793454	New Britain
12	2	18.113588781	New Britain
13	3	15.484046715	New Britain
14	4	21.725410751	New Britain
15	1	7.9491885897	Stamford
16	2	3.6181008555	Stamford
17	3	29.067550396	Stamford
18	4	48.706478315	Stamford
19	1	5.7470756043	Hamden
20	2	3.1926336436	Hamden
21	3	4.049978365	Hamden
22	4	4.633296323	Hamden
23	1	17.091629852	New Britain
24	2	10.754404413	New Britain
25	3	15.63335738	New Britain
26	1	1.4360183376	New Britain
27	2	2.4931218071	New Britain
28	3	1.4424557133	New Britain

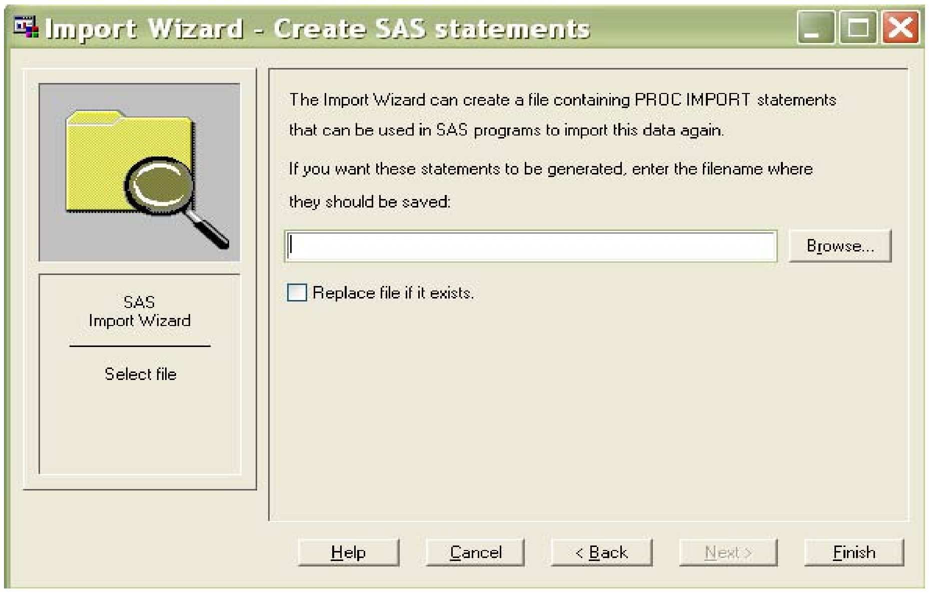
```
30     data   'Z:\JKMH\SAS 2013\data\practice
31     set x ;   run;

NOTE: There were 250 observations read from
NOTE: The data set Z:\JKMH\SAS 2013\data\pr
NOTE: DATA statement used (Total process t
      real time              0.07 seconds
      cpu time               0.00 seconds
```

```
Libname x 'Z:\JKMH\SAS 2013\data';
Data x.practice1;
Set xyz;
Run;


data  'Z:\JKMH\SAS 2013\data\practice1';
set x ;  run;
```

Display 1. Main Interface for SAS 9.2

Import Procedure:

Using the Import Wizard (located in the tool bar under File > Import Data) to import data files is an easy and efficient way of putting external data into a SAS dataset. Files can be imported from various formats such as Access, Excel and text files. The variables which are imported using the wizard are based on the records provided. The last screen shown by the import wizard indicates an option to save the import syntax, clicking the finish button will run the statements and create the file; alternatively, the PROC IMPORT can be saved as a new program or appended to an existing program for future use.



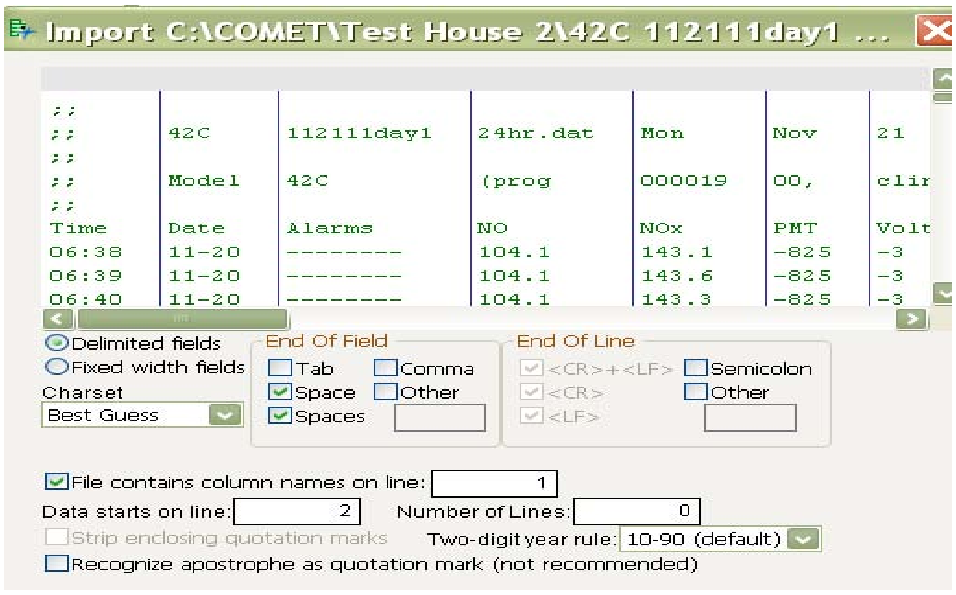
The Import Wizard can create a file containing PROC IMPORT statements that can be used in SAS programs to import this data again.

If you want these statements to be generated, enter the filename where they should be saved:

Replace file if it exists.

Display 2. Import Wizard

JMP also offers a helpful procedure when opening a data file. If the file has an awkward format, JMP offers an option of, “Data with preview”, providing the ability to manually move columns, as well as indicate delimiters and headings. The file is imported into a JMP dataset but can then be saved as a SAS dataset.



```
;;
;;      42C       112111day1   24hr.dat   Mon      Nov    21
;;
;;      Model     42C          (prog      000019   00,    clir
;;
Time    Date      Alarms       NO         NOx      PMT    Volt
06:38   11-20     ---------    104.1      143.1    -825   -3
06:39   11-20     ---------    104.1      143.6    -825   -3
06:40   11-20     ---------    104.1      143.3    -825   -3
```

[✓ Delimited fields] Fixed width fields
Charset: Best Guess

End Of Field: Tab, Comma, [✓ Space], Other, [✓ Spaces]

End Of Line: [✓ <CR>+<LF>], Semicolon, [✓ <CR>], Other, [✓ <LF>]

[✓ File contains column names on line:] 1

Data starts on line: 2 Number of Lines: 0

Strip enclosing quotation marks Two-digit year rule: 10-90 (default)

Recognize apostrophe as quotation mark (not recommended)

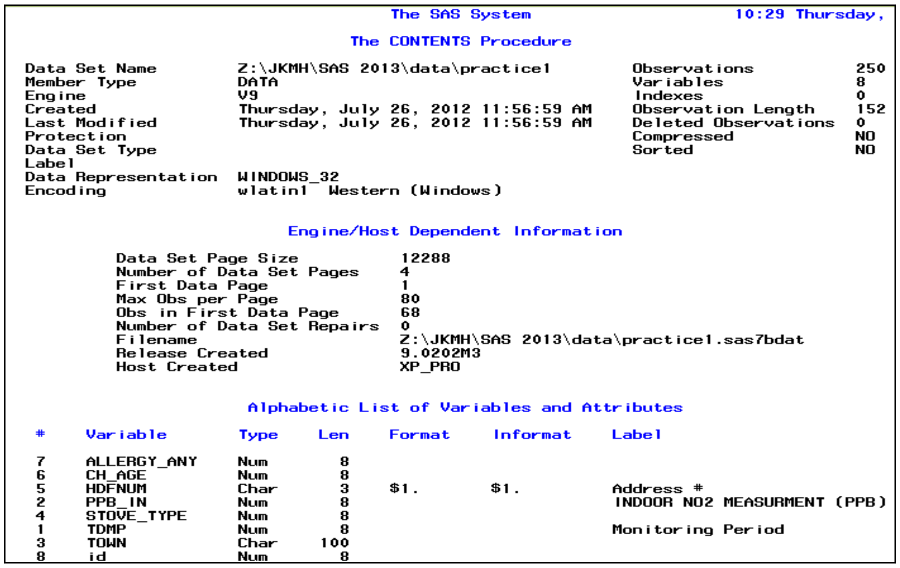
Display 3. JMP Import options

Exploring the Data:

One way to get acquainted with new data is to open the dataset through the explorer tab. Just looking at the dataset can be useful. Once a dataset is open, the view options on the toolbar become enabled and the user can investigate column names and column labels.

The best way to become familiar with a foreign dataset is to begin with a PROC CONTENTS (syntax written in the editor window), which gives a detailed description of what is in the SAS dataset. PROC CONTENTS automatically outputs a listing which includes the number of observations, and a list of variables specifying their names, formats, informats and labels.

```
proc contents data =  'Z:\JKMH\SAS 2013\data\practice1'; run;
```



```
                                   The SAS System                      10:29 Thursday,

                               The CONTENTS Procedure

Data Set Name         Z:\JKMH\SAS 2013\data\practice1          Observations          250
Member Type           DATA                                     Variables             8
Engine                V9                                       Indexes               0
Created               Thursday, July 26, 2012 11:56:59 AM      Observation Length    152
Last Modified         Thursday, July 26, 2012 11:56:59 AM      Deleted Observations  0
Protection                                                     Compressed            NO
Data Set Type                                                  Sorted                NO
Label
Data Representation   WINDOWS_32
Encoding              wlatin1  Western (Windows)

                        Engine/Host Dependent Information

            Data Set Page Size          12288
            Number of Data Set Pages    4
            First Data Page             1
            Max Obs per Page            80
            Obs in First Data Page      68
            Number of Data Set Repairs  0
            Filename                    Z:\JKMH\SAS 2013\data\practice1.sas7bdat
            Release Created             9.0202M3
            Host Created                XP_PRO

                     Alphabetic List of Variables and Attributes

 #    Variable        Type    Len    Format    Informat    Label

 7    ALLERGY_ANY     Num       8
 6    CH_AGE          Num       8
 5    HDFNUM          Char      3    $1.       $1.         Address #
 2    PPB_IN          Num       8                          INDOOR NO2 MEASURMENT (PPB)
 4    STOVE_TYPE      Num       8
 1    TDMP            Num       8                          Monitoring Period
 3    TOWN            Char    100
 8    id              Num       8
```

Output 1. Output from a Proc Contents Statement

The default for PROC CONTENTS is to list the variables alphabetically (see output 1). Including an order statement allows for the user to control the order in which variables are printed.

```
proc contents data =  'Z:\JKMH\SAS 2013\data\practice1' order =varnum ; run;
```

Order = varnum prints the variables in order of position and can be useful when sub-setting data.

Data Step:

Data steps are written by the programmer with the ultimate goal of creating a new SAS dataset. The following flow chart, obtained from the support.sas.com website, shows the steps the data step takes to accomplish its job.

Each individual data step is read by SAS before moving on to the next. Two types of datasets can be created, temporary and permanent. Temporary datasets are the default and are not saved when SAS is closed; in other words they are only saved as long as the session is open. For the duration of the session they reside in the work folder and can be called upon at anytime. Permanent datasets are written to an existing location.

In the DATA statement, the programmer names the dataset and tells SAS where to save the data. The programmer writes a full path in single quotes to permanently store the data or simply names the dataset without quotes to send the dataset to the work folder for temporary storage.

```
data  'Z:\JKMH\SAS 2013\data\practice1';
set x ;  run;
```

Another option is to create a libname statement before the data step. The user can then call on a path with a single letter or phrase for the duration of that SAS session.

```
Libname x 'Z:\JKMH\SAS 2013\data';
Data x.practice1;
Set xyz;
Run;
```

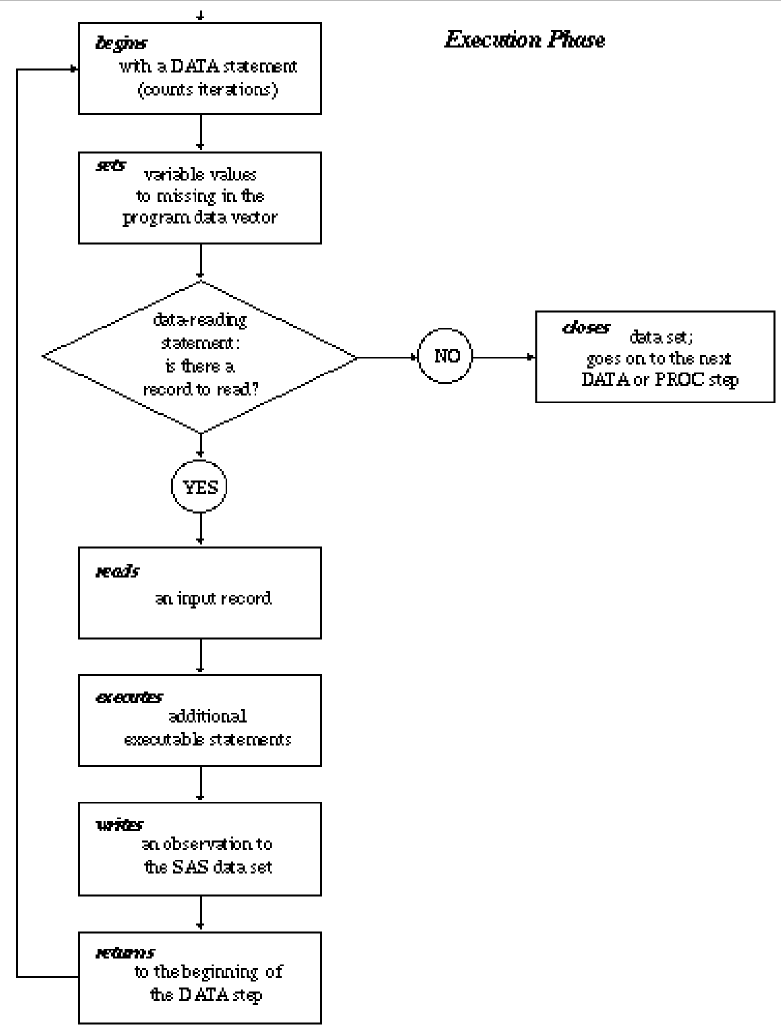

Execution Phase
begins with a DATA statement (counts iterations)
sets variable values to missing in the program data vector
data-reading statement: is there a record to read?
NO
closes data set; goes on to the next DATA or PROC step
YES
reads an input record
executes additional executable statements
writes an observation to the SAS data set
returns to the beginning of the DATA step

Figure 1. Flow Chart of Data Step

The SET or MERGE statement tells SAS where to look for the input data. If the dataset (s) called upon in the set statement contains readable data then the following requested steps are executed. There are many ways to manipulate data within the data step; common functions include sub-setting and creating new variables. Once additional statements are executed the new dataset is written.

A MERGE statement replaces the SET statement when combining observations from two SAS datasets into a single new dataset, using a single unique identifier.

```
data practice4;
merge practice1
practice3;
by id; run;
```

If datasets have a large number of variables, the double dash (--) is a common way to identify variables in order of position. The numerical order of variables can be obtained, as we have shown previously, by including an order=varnum statement in the PROC CONTENTS. This short-cut allows the programmer to skip the tedious task of typing all the variable names that are needed.

```
data x (keep= tdmp--
ch_age); set 'Z:\JKMH\SAS
2013\data\practice1' ;
run;
```

A series of data steps is often used to create datasets in succession until the goal dataset is completed. In order to execute syntax, it must end with the word "RUN." The *run* icon is located on the toolbar.

Reading the Log:

The log is documentation of everything done during a SAS session. Periodically checking the log for messages could save a great deal of time later. Just because a dataset was successfully created or output does not mean that it contains what was intended. Using the log is a great way to check and recheck that the task has been accomplished correctly.

The three most common types of messages include Note, Error and Warning. Line numbers are included in some messages to indicate the exact point in the program where an error occurred.

Note – appears in blue, it describes system processing and procedures run.

```
NOTE: There were 250 observations read from the data set WORK.X.
```

Error – appears in red, it identifies a problem that required SAS to stop running or create an invalid or empty dataset.

```
ERROR: File WORK.XYZ.DATA does not exist.
```

Warning – appears in green, this warns the user of potential hazards in the data. The warning statement should be read carefully, common warnings are due to variable lengths, incomplete datasets and the halt of data steps before completion.

```
WARNING: Multiple lengths were specified for the BY variable studnum by input data sets. This may
cause unexpected results

WARNING: Data set X.PRACTICE1 was not replaced because this step was stopped.
```

PROCEDURES

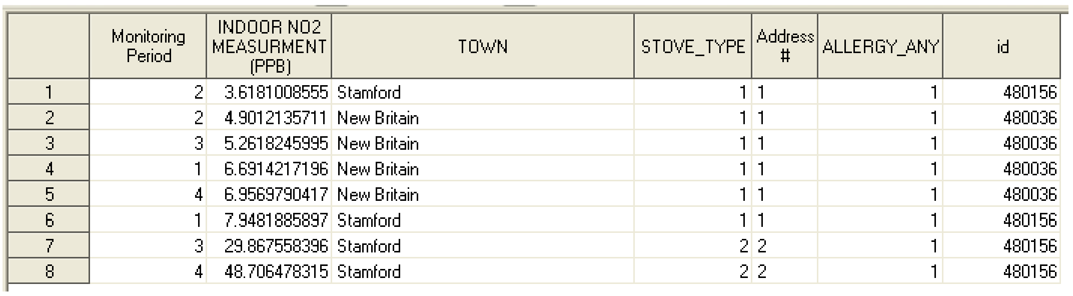
Once the dataset has been created and stored appropriately the options are limitless. Most tasks in SAS are completed using what is often referred to as a "PROC," an abbreviation for procedure. Each "PROC" allows the user to manipulate and/or view their data in a new way. SAS offers hundreds of different procedures and within each one, dozens of options can be employed, allowing the user ultimate flexibility in their final product. Four of the most common procedures are described below with some useful options.

PROC SORT

The SORT procedure orders a SAS data set by the value of the variable(s) that is (are) listed in the BY statement. The SORT procedure is a pre-requisite for invoking a BY statement in any future procedure used on that dataset. In other words, to use a BY statement in any PROC or DATA step the input dataset must already be sorted by the variables you list in the BY statement. Datasets can be sorted by multiple variables to further specify the order of observations. The SORT procedure orders the dataset in ascending order of each variable listed in the BY statement.

In such an example the dataset would be sorted so that all observations with the smallest value of *the first listed variable* would come first and then observations would be ordered by the value of *second listed variable*. When the BY statement includes a character value the observations are sorted in alphabetical order.

The database below is not sorted.

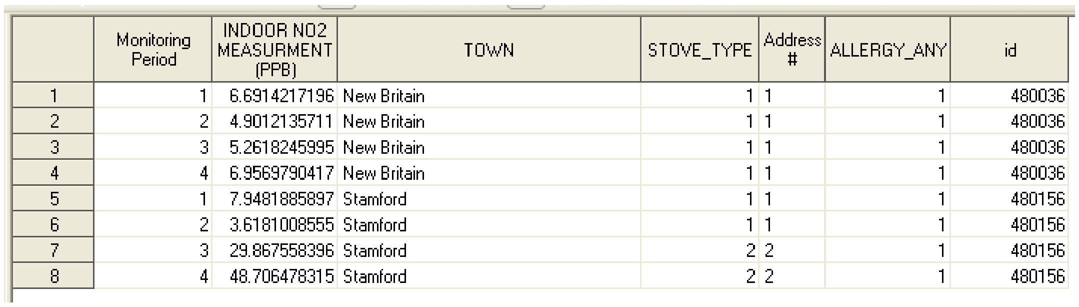


	Monitoring Period	INDOOR NO2 MEASURMENT (PPB)	TOWN	STOVE_TYPE	Address #	ALLERGY_ANY	id
1	2	3.6181008555	Stamford	1	1	1	480156
2	2	4.9012135711	New Britain	1	1	1	480036
3	3	5.2618245995	New Britain	1	1	1	480036
4	1	6.6914217196	New Britain	1	1	1	480036
5	4	6.9569790417	New Britain	1	1	1	480036
6	1	7.9481885897	Stamford	1	1	1	480156
7	3	29.867558396	Stamford	2	2	1	480156
8	4	48.706478315	Stamford	2	2	1	480156

Table 1. Unsorted Data

Using the following syntax, sorts the data by monitoring period within ID. The resulting dataset is also shown below.

```
PROC SORT DATA = X;
BY ID TDMP;
RUN;
```



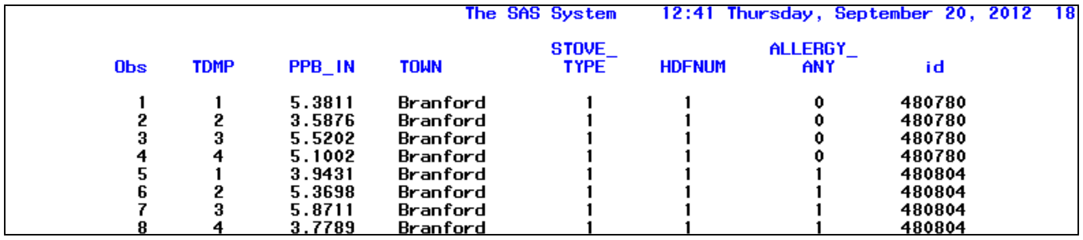
	Monitoring Period	INDOOR NO2 MEASURMENT (PPB)	TOWN	STOVE_TYPE	Address #	ALLERGY_ANY	id
1	1	6.6914217196	New Britain	1	1	1	480036
2	2	4.9012135711	New Britain	1	1	1	480036
3	3	5.2618245995	New Britain	1	1	1	480036
4	4	6.9569790417	New Britain	1	1	1	480036
5	1	7.9481885897	Stamford	1	1	1	480156
6	2	3.6181008555	Stamford	1	1	1	480156
7	3	29.867558396	Stamford	2	2	1	480156
8	4	48.706478315	Stamford	2	2	1	480156

Table 2. Sorted Data

PROC PRINT

The PRINT procedure 'prints' the contents of the specified dataset in the output window. From the output window, information can easily be copied and pasted into other software's such as Word or PowerPoint. The information can also be printed on paper directly from the output window. Basic syntax and standard output for the PRINT procedure is as follows:

```
PROC PRINT DATA = x;
RUN;
```



```
                                    The SAS System    12:41 Thursday, September 20, 2012  18

                                              STOVE_              ALLERGY_
    Obs    TDMP    PPB_IN    TOWN              TYPE     HDFNUM      ANY        id

      1      1     5.3811    Branford            1        1          0       480780
      2      2     3.5876    Branford            1        1          0       480780
      3      3     5.5202    Branford            1        1          0       480780
      4      4     5.1002    Branford            1        1          0       480780
      5      1     3.9431    Branford            1        1          1       480804
      6      2     5.3698    Branford            1        1          1       480804
      7      3     5.8711    Branford            1        1          1       480804
      8      4     3.7789    Branford            1        1          1       480804
```

Output 2. Proc Print

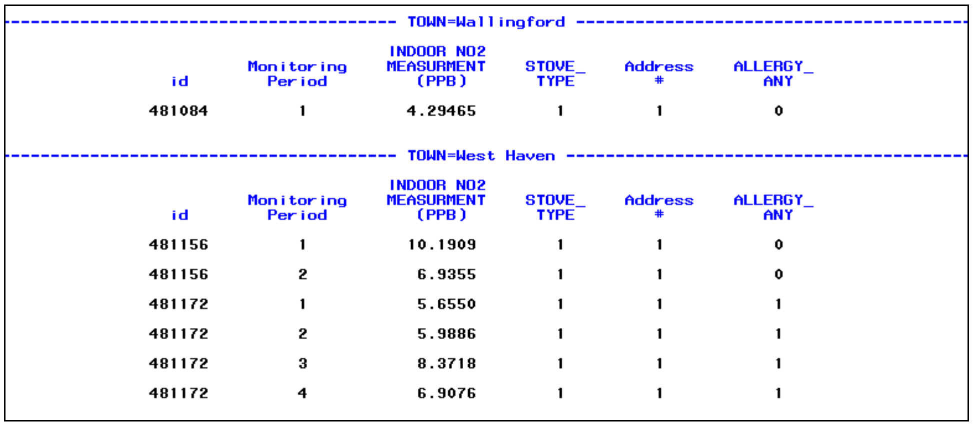
Options in the PROC PRINT statement allow the user to change the visual appearance of the output and additional statements within the procedure allow the user to modify which observations and variables are displayed in the output. Some of the most commonly used options and statements are listed in table 3.

Options in the PROC PRINT statement	
NOOBS	Removes the column of data providing the SAS assigned observation number
D	Double spaces the output
LABEL	Uses variable labels rather than variable names as column headings
Additional Statements in the PRINT procedure	
ID	specifies one or more variables to use instead of observation numbers to identify observations in the report
VAR	specifies a subset of variables to appear in the output
WHERE	subsets the input dataset and limits the output to those observations meeting the specified criteria
BY	produces a separate section of the report for each BY group
PAGEBY	starts a new page when the specified variable changes value or the variable in the BY statement changes value

Table 3. Proc Print Options

A sample of more complex syntax for the PRINT procedure is below along with a selection from the output. Note that the dataset was first sorted by *town* in order to employ the BY statement.

```
PROC SORT DATA = X;
BY TOWN;
RUN;

PROC PRINT DATA = X D LABEL;
ID ID;
BY TOWN;
RUN;
```



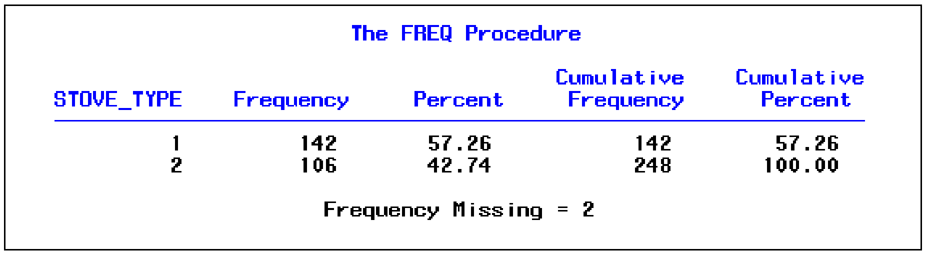
```
---------------------------------------- TOWN=Wallingford ----------------------------------------

                           INDOOR NO2
                Monitoring MEASURMENT   STOVE_   Address   ALLERGY_
       id         Period      (PPB)      TYPE       #         ANY

     481084         1        4.29465       1        1          0

---------------------------------------- TOWN=West Haven ----------------------------------------

                           INDOOR NO2
                Monitoring MEASURMENT   STOVE_   Address   ALLERGY_
       id         Period      (PPB)      TYPE       #         ANY

     481156         1        10.1909       1        1          0

     481156         2         6.9355       1        1          0

     481172         1         5.6550       1        1          1

     481172         2         5.9886       1        1          1

     481172         3         8.3718       1        1          1

     481172         4         6.9076       1        1          1
```

Output 3. Proc Print with D and LABEL options

PROC FREQ

The FREQ procedure produces one-way to *n*-way frequency tables and listings (printed in the output window) allowing the user to answer ever popular questions such as "How many?" or "What percentage?" Using any number of variables, which must be specified in the TABLES statement, the user can create multi-way crosstabs or cross-listings of variables. Basic syntax for the FREQ procedure is as follows:

```
PROC FREQ DATA = X;
TABLES STOVE_TYPE;
RUN;
```



```
                        The FREQ Procedure

                                              Cumulative    Cumulative
STOVE_TYPE    Frequency     Percent      Frequency       Percent
-----------------------------------------------------------------------
         1          142       57.26            142         57.26
         2          106       42.74            248        100.00

                      Frequency Missing = 2
```

Output 4. Proc Freq

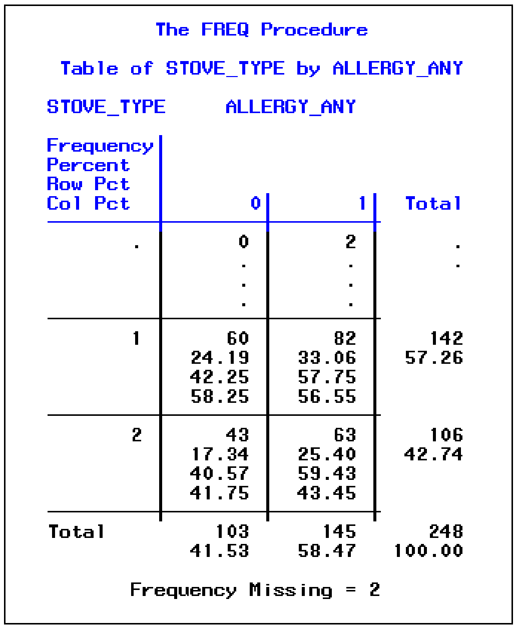
Various options in the TABLES statement are helpful in identifying data errors, patterns in coding/miscoding and solving all kinds of data mysteries. Adding additional statements, like those used in the PRINT procedure allow the user to subset or stratify the data displayed in the output window. A list of options specific to the TABLES statement is below:

Options in the PROC PRINT statement	
MISSPRINT	Displays missing values as part of the crosstab
LIST	Displays the output in list form, rather than as a crosstab
MISSING	Displays missing values as part of a list output

Table 4. Proc Freq options

Some examples of more intricate PROC FREQ syntax are as follows along with their respective outputs:

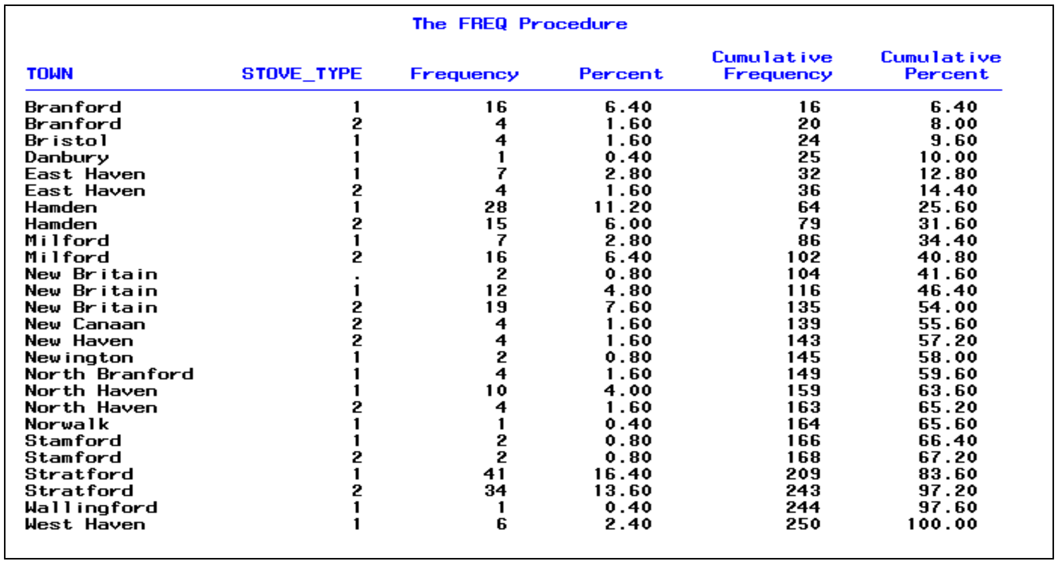
```
PROC FREQ DATA = X;
TABLES STOVE_TYPE*ALLERGY_ANY / MISSPRINT;
RUN;
```



```
          The FREQ Procedure

   Table of STOVE_TYPE by ALLERGY_ANY

STOVE_TYPE      ALLERGY_ANY

Frequency|
Percent  |
Row Pct  |
Col Pct  |       0|       1|  Total
---------+--------+--------+
       . |      0 |      2 |      .
         |      . |      . |      .
         |      . |      . |
         |      . |      . |
---------+--------+--------+
       1 |     60 |     82 |    142
         |  24.19 |  33.06 |  57.26
         |  42.25 |  57.75 |
         |  58.25 |  56.55 |
---------+--------+--------+
       2 |     43 |     63 |    106
         |  17.34 |  25.40 |  42.74
         |  40.57 |  59.43 |
         |  41.75 |  43.45 |
---------+--------+--------+
Total         103      145      248
            41.53    58.47   100.00

        Frequency Missing = 2
```

Output 5. Frequency table with missprint option

```
PROC FREQ DATA = X;
TABLES TOWN*STOVE_TYPE / LIST;
RUN;
```



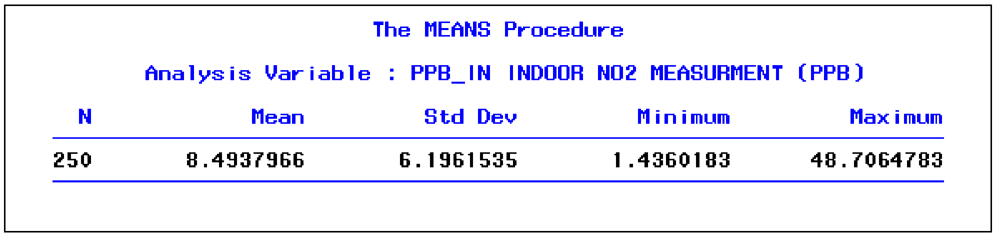
```
                                    The FREQ Procedure

                                                                  Cumulative    Cumulative
TOWN              STOVE_TYPE     Frequency       Percent         Frequency       Percent

Branford                   1            16          6.40                16          6.40
Branford                   2             4          1.60                20          8.00
Bristol                    1             4          1.60                24          9.60
Danbury                    1             1          0.40                25         10.00
East Haven                 1             7          2.80                32         12.80
East Haven                 2             4          1.60                36         14.40
Hamden                     1            28         11.20                64         25.60
Hamden                     2            15          6.00                79         31.60
Milford                    1             7          2.80                86         34.40
Milford                    2            16          6.40               102         40.80
New Britain                .             2          0.80               104         41.60
New Britain                1            12          4.80               116         46.40
New Britain                2            19          7.60               135         54.00
New Canaan                 2             4          1.60               139         55.60
New Haven                  2             4          1.60               143         57.20
Newington                  1             2          0.80               145         58.00
North Branford             1             4          1.60               149         59.60
North Haven                1            10          4.00               159         63.60
North Haven                2             4          1.60               163         65.20
Norwalk                    1             1          0.40               164         65.60
Stamford                   1             2          0.80               166         66.40
Stamford                   2             2          0.80               168         67.20
Stratford                  1            41         16.40               209         83.60
Stratford                  2            34         13.60               243         97.20
Wallingford                1             1          0.40               244         97.60
West Haven                 1             6          2.40               250        100.00
```

Output 6. Frequency table as a listing

PROC MEANS

The MEANS procedure provides data summarization tools to compute descriptive statistics for variables across all observations and within groups of observations. (ref: SAS Help and Documentation) This procedure is used to create descriptive statistics like mean, median, standard deviation, etc. Similar to the PRINT and FREQ procedures, additional statements such as BY and WHERE can be used to subset or stratify the output. Basic syntax for the MEANS procedure is provided below.

```
PROC MEANS DATA = X;
VAR PPB_IN;
RUN;
```

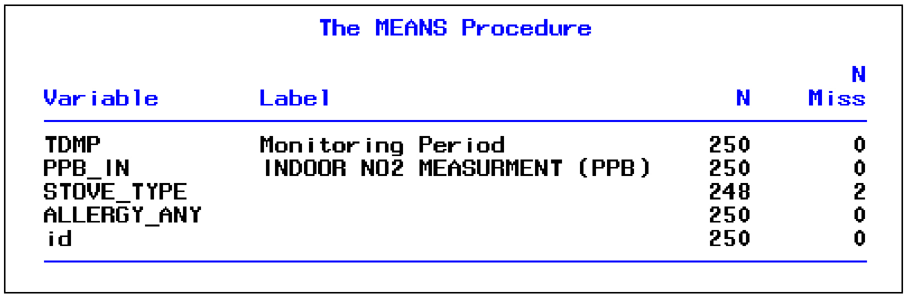


```
                        The MEANS Procedure

     Analysis Variable : PPB_IN INDOOR NO2 MEASURMENT (PPB)

  N           Mean         Std Dev         Minimum         Maximum
------------------------------------------------------------------
250      8.4937966       6.1961535       1.4360183      48.7064783
------------------------------------------------------------------
```

Output 7. Proc Means

Additionally, the MEANS procedure allows the user to print a list of all variables in a dataset which will provide the number of observations and missing observations for each variable. This is often a good place to start when familiarizing oneself with a new dataset. The syntax below produces such an output.

```
PROC MEANS DATA = X N NMISS;
RUN;
```



```
                          The MEANS Procedure

                                                                        N
Variable        Label                                   N           Miss
-------------------------------------------------------------------------
TDMP            Monitoring Period                     250              0
PPB_IN          INDOOR NO2 MEASURMENT (PPB)           250              0
STOVE_TYPE                                            248              2
ALLERGY_ANY                                           250              0
id                                                    250              0
-------------------------------------------------------------------------
```

Output 8. Proc means with options n and miss

CONCLUSION

The SAS Essentials training program is an easy course to implement for those who want their support staff to succeed and excel at interpreting, processing and summarizing data. Blending the idea of efficiency, empowerment and ultimately self-management is an opportunity for supervisors to fully utilize the time of all their staff and create a successful and productive team.

REFERENCES

1. "An Introduction to SAS Data Steps". Social Science Computing Cooperative. 26Jul2012. Available at http://www.ssc.wisc.edu/sscc/pubs/4-18.htm,

2. "Base SAS Glossary." Support.sas.com. 07Feb2013. Available at http://support.sas.com/documentation/cdl/en/mastergl/62860/HTML/default/viewer.htm#glossary.htm

3. Collison,Chris . 2005. *Learning to Fly - Practical Knowledge Management from Leading and Learning.* 24-25

4. "Overview of DATA Step Processing". Support.sas.com. 02Aug2012. Available at http://support.sas.com/documentation/cdl/en/lrcon/62955/HTML/default/viewer.htm#a000985872.htm

.

CONTACT INFORMATION

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

Julie Kezik
Yale University
1 Church Street, 6th Floor
New Haven, CT 06510
203-764-9375
203-764-9378
Julie.colburn@yale.edu