

## Reporting in the Age of Twitter: Concise Reports and Continuous Real-Time Monitoring

LeRoy Bessler PhD, Bessler Consulting and Research  
Fox Point, Milwaukee, Wisconsin, USA, Le\_Roy\_Bessler@wi.rr.com

### Abstract

Twitter is built on the concept of concise messages, and encourages, or at least facilitates, continuously keeping in touch.

This presentation will provide you two macros to create compact tabular reports. One of them “shows you the most with the least”, by implementing my long advocated design mantra of “Show Them What’s Important” and the concept of “Let Part Stand for the Whole”. The other provides a “twist”, literally and figuratively, on the traditional default wide display of columns, which can entail the annoying, inconvenient, anti-communicative requirement for sideways scrolling on a web page.

This presentation will also show you how to keep in touch with your data. You can build yourself a monitor to continuously check status of critical indicators and then send email alerts to people (including or only yourself if appropriate) with a need to know.

### Introduction

Back in 1987, when asked to replace a report to executive management, I developed the concept and macro shown below to “Show Them What’s Important” and “Let Part Stand for the Whole”.

Much more recently I created the TransposedPrint macro shown below as a personal convenience to be able to easily inspect the content of a data set.

About two years ago I developed a tool to monitor various measures of user activity on a SAS BI server, and last year I adapted it to specifically monitor CPU consumption and send email alerts to anyone whose SAS process might be in a loop. I don’t present that code here, but rather a simple, derivative toy program that provides the logical structure that you could adapt and use To Alert Them When They Need To Know about any particular measure(s) of interest that your application beneficiaries care about.

Except for the MonitorAndAlert, which produces only an email message (but could be adapted to send an attachment or to include a link, or at least identify the path, to a file of more information) examples of output are included.

## How To “Show Them What’s Important” and “Let Part Stand for the Whole”

Many years ago when I had to prepare a report for executive management, Kenneth J. Wesley counseled me with this: “If you want them to read it, it must fit on one page.”

Well, the fact of the matter is that usually a huge percent of what matters can be fit in a subsetting ranking report, with ranking from high to low if high is good, or vice-versa.

There are three ways to subset ranked data:

- The Top N;
- The Top P Percent (Show them as many observations from the top of the ranked list until you account for at least P Percent of the grand total of the measure of interest);
- Every observation where the measure of interest exceeds a threshold.

Ranking can be from high to low, or low to high, depending on whether large or small is good.

On the next three pages are examples of these three types of reports. A listing of the SubsettingRankingReport macro and the macro invocations used to produce the examples follow.

Note that the design of the reports and the construction of the macro is such that the reader not only gets the display of what’s important, but also the reader is told the grand total of the measure of interest, the grand total number of observations, and what part of these grand totals are accounted for by the subset that is presented. It is important to anticipate and answer questions as to how much has been left out.

For an adaptation of this concept to a horizontal bar chart, please see Reference 1. The horizontal bar chart design provided there, in effect, simply adds a visual comparator to the tabular listings below.

Top 10 Shoe Sales By City

This subset accounts for 50.6% of Total Shoe Sales

Grand Totals: City Count = 53, Shoe Sales = \$33,851,566

Rank	City	Shoe Sales
1	Vancouver	\$3,227,768
2	Tel Aviv	\$2,567,568
3	Kingston	\$2,235,204
4	Dubai	\$1,910,544
5	Chicago	\$1,565,585
6	New York	\$1,489,207
7	Al-Khobar	\$1,153,667
8	Minneapolis	\$1,099,937
9	Heidelberg	\$967,739
10	Lisbon	\$898,345
		=====
		\$17,115,564

Top 25 Shoe Sales By City

Selected to list the Top 80% of Total Shoe Sales  
(This subset actually accounts for 80.5%)

Grand Totals: City Count = 53, Shoe Sales = \$33,851,566

Rank	City	Shoe Sales
1	Vancouver	\$3,227,768
2	Tel Aviv	\$2,567,568
3	Kingston	\$2,235,204
4	Dubai	\$1,910,544
5	Chicago	\$1,565,585
6	New York	\$1,489,207
7	Al-Khobar	\$1,153,667
8	Minneapolis	\$1,099,937
9	Heidelberg	\$967,739
10	Lisbon	\$898,345
11	Manila	\$854,904
12	Caracas	\$789,323
13	Warsaw	\$786,714
14	London	\$762,009
15	Cairo	\$738,198
16	Los Angeles	\$737,312
17	Copenhagen	\$693,116
18	Jakarta	\$649,044
19	San Juan	\$639,155
20	Prague	\$636,721
21	Paris	\$621,877
22	Seattle	\$611,945
23	Moscow	\$560,976
24	Montreal	\$531,028
25	La Paz	\$530,506
		=====
		\$27,258,392

Ranked Shoe Sales not less than \$1,000,000 By City

This subset accounts for 45.0% of Total Shoe Sales

Grand Totals: City Count = 53, Shoe Sales = \$33,851,566

Rank	City	Shoe Sales
1	Vancouver	\$3,227,768
2	Tel Aviv	\$2,567,568
3	Kingston	\$2,235,204
4	Dubai	\$1,910,544
5	Chicago	\$1,565,585
6	New York	\$1,489,207
7	Al-Khobar	\$1,153,667
8	Minneapolis	\$1,099,937
		=====
		\$15,249,480

Here is the code used to produce the reports:

```
%MACRO SubsettedRankingReport
(DATA=,
CLASSVAR=, CVARLABL=,
RANKVAR=, RVARFMT=, RVARLABL=,
NN=, MINVALUE=, PERCENT=,
TITLESUFFIX=, DateAndPageNumber=NO);

PROC SORT DATA=&DATA(keep=&CLASSVAR &RANKVAR)
          OUT=RANKED;
BY DESCENDING &RANKVAR;
RUN;

PROC MEANS DATA=&DATA NOPRINT SUM N;
VAR &RANKVAR;
OUTPUT OUT=TOTSTATS SUM=TOTSUM N=TOTNN;
RUN;

DATA SUBSETTED;
RETAIN TOPSUM TOPPCT 0 TOTSUM;
SET RANKED END=LAST;
IF _N_ EQ 1 THEN DO;
  SET TOTSTATS;
  CALL SYMPUT('TOTSUM',trim(left(put(TOTSUM,&RVARFMT))));
  CALL SYMPUT('TOTNN',trim(left(TOTNN)));
END;
RANK = _N_;
%if %length(&MINVALUE) ne 0
%then %do;
IF &RANKVAR GE &MINVALUE
THEN DO;
  TOPSUM = TOPSUM + &RANKVAR;
END;
ELSE DO;
  TOPPCT = (TOPSUM / TOTSUM) * 100;
  CALL SYMPUT('TOPPCT',trim(left(put(TOPPCT,5.1))));
  CALL SYMPUT('TOPSUM',trim(left(put(TOPSUM,&RVARFMT))));
  CALL SYMPUT('TOPNN',trim(left(_N_ - 1)));
  STOP;
END;
%end;
%else
%if %length(&PERCENT) ne 0
%then %do;
TOPSUM = TOPSUM + &RANKVAR;
TOPPCT = (TOPSUM / TOTSUM) * 100;
IF TOPPCT GE &PERCENT
THEN DO;
  OUTPUT;
  CALL SYMPUT('TOPPCT',trim(left(put(TOPPCT,5.1))));
```

```

CALL SYMPUT('TOPSUM',trim(left(put(TOPSUM,&RVARFMT))));
CALL SYMPUT('TOPNN',trim(left(_N_)));
STOP;
END;
%end;
%else
%if %length(&NN) ne 0
%then %do;
IF _N_ LE &NN
THEN DO;
TOPSUM = TOPSUM + &RANKVAR;
END;
ELSE DO;
TOPPCT = (TOPSUM / TOTSUM) * 100;
CALL SYMPUT('TOPPCT',trim(left(put(TOPPCT,5.1))));
CALL SYMPUT('TOPSUM',trim(left(put(TOPSUM,&RVARFMT))));
CALL SYMPUT('TOPNN',trim(left(_N_ - 1)));
STOP;
END;
%end;
OUTPUT;
IF LAST;
TOPPCT = (TOPSUM / TOTSUM) * 100;
CALL SYMPUT('TOPPCT',trim(left(put(TOPPCT,5.1))));
CALL SYMPUT('TOPSUM',trim(left(put(TOPSUM,&RVARFMT))));
CALL SYMPUT('TOPNN',trim(left(_N_)));
RUN;

DATA _NULL_;
CALL SYMPUT('RANKLEN',trim(left(length(trim(left(&TOPNN))))));
%if %eval(&TOPNN < &TOTNN)
%then %do;
%if %length(&NN) ne 0
%then %do;
CALL SYMPUT('TITLEPREFIX',"Top &TOPNN &RVARLABL By &CVARLABL");
%end;
%else
%if %length(&MINVALUE) ne 0
%then %do;
CALL SYMPUT('TITLEPREFIX',
"Ranked &RVARLABL not less than "||
trim(left(put(&minvalue,&RVARFMT)))||
" By &CVARLABL");
%end;
%else
%if %length(&PERCENT) ne 0
%then %do;
CALL SYMPUT('TITLEPREFIX',"Top &TOPNN &RVARLABL By &CVARLABL");
%end;
%end;
%else %do;
CALL SYMPUT('TITLEPREFIX',"Ranked &RVARLABL By &CVARLABL");

```

```

%end;
RUN;

%if %upcase(&DateAndPageNumber) eq NO %then %do;
OPTIONS NODATE NONUMBER;
%end;
%if %length(&TITLESUFFIX) eq 0
%then %let TITLE1 = &TITLEPREFIX;
%else %let TITLE1 = &TITLEPREFIX &TITLESUFFIX;

TITLE1 "&TITLE1";
%let LastTitleNumber = 5;
%if %length(&PERCENT) ne 0
and
%eval(&TOPNN < &TOTNN)
%then %do;
TITLE3
"Selected to list the Top &PERCENT.% of Total &RVARLABL";
%if %sysevalf(&TOPPCT > &PERCENT)
%then %do;
TITLE4 "(This subset actually accounts for &TOPPCT.%)";
%let LastTitleNumber = %eval(&LastTitleNumber + 1);
%end;
%end;
%else
%if %eval(&TOPNN < &TOTNN)
%then %do;
TITLE3 "This subset accounts for &TOPPCT.% of Total &RVARLABL";
%end;
%if %eval(&TOPNN < &TOTNN)
%then %do;
TITLE&LastTitleNumber
"Grand Totals: &CVARLABL Count = &TOTNN, &RVARLABL = &TOTSUM";
%end;
/* to force line breaks in your labels, use _ */
PROC PRINT DATA=SUBSETTED NOOBS U LABEL SPLIT='_';
FORMAT RANK &RANKLEN.;
FORMAT &RANKVAR &RVARFMT;
LABEL RANK = 'Rank'
&CLASSVAR = "&CVARLABL"
&RANKVAR = "&RVARLABL";
VAR RANK &CLASSVAR &RANKVAR;
SUM &RANKVAR;
RUN;

%MEND SubsettedRankingReport;

proc summary nway data=sashelp.shoes;
class subsidiary;
var sales;
output out=work.ShoeSalesByCity(rename=(subsidiary=City)) sum=Sales;
run;

```



```
%SubsettedRankingReport (DATA=work.ShoeSalesByCity,  
    CLASSVAR=City,  
    CVARLABL=City,  
    RANKVAR=Sales,  
    RVARFMT=dollar11.,  
    RVARLABL=Shoe Sales,  
    NN=10);
```

```
%SubsettedRankingReport (DATA=work.ShoeSalesByCity,  
    CLASSVAR=City,  
    CVARLABL=City,  
    RANKVAR=Sales,  
    RVARFMT=dollar11.,  
    RVARLABL=Shoe Sales,  
    MINVALUE=1000000);
```

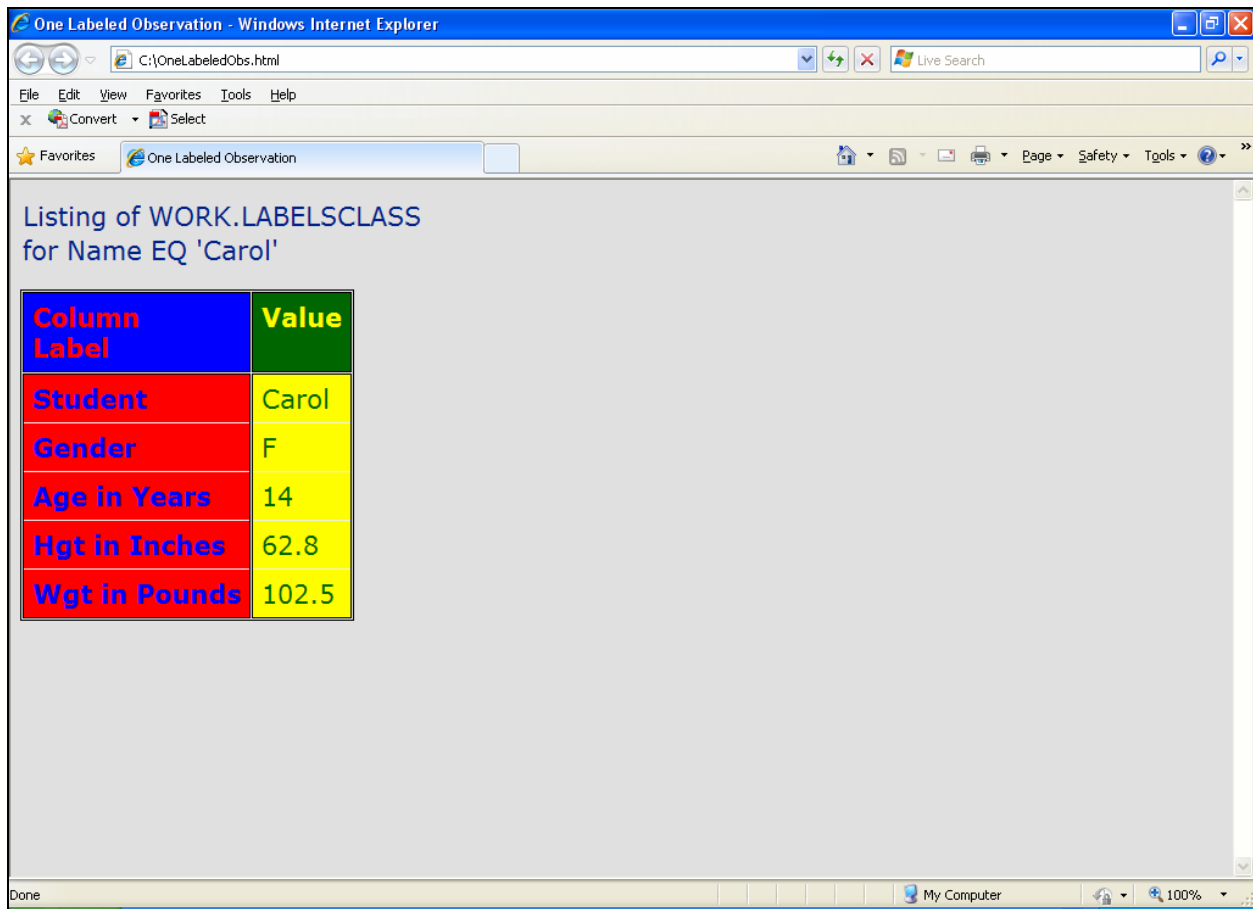
```
%SubsettedRankingReport (DATA=work.ShoeSalesByCity,  
    CLASSVAR=City,  
    CVARLABL=City,  
    RANKVAR=Sales,  
    RVARFMT=dollar11.,  
    RVARLABL=Shoe Sales,  
    PERCENT=80);
```

## **How To Create A Transposed Listing of Your Table Content**

Typical listings from PROC SQL, PROC PRINT, PROC REPORT or PROC TABULATE display data in a table with variable names, or labels, in the top row with values of each variable stacked in a column below its heading. If you have numerous variables, this standard convention can produce an excessively wide listing. Even for just a few observations, sideways scrolling is an on-line viewing nuisance, and the hard copy result of printing is unacceptable. It can be more usable for the on-line viewer or the hard-copy recipient to work with a transposed listing. On the next three pages are examples of transposed prints. A listing of the TransposedPrint macro and the macro invocations used to produce the examples follow.

## One Observation in HTML Format

(The data set, a copy of SASHELP.CLASS, has variable labels.)



The screenshot shows a web browser window titled "One Labeled Observation - Windows Internet Explorer". The address bar shows the file path "C:\OneLabeledObs.html". The browser's menu bar includes "File", "Edit", "View", "Favorites", "Tools", and "Help". The status bar at the bottom indicates "Done" and "My Computer" with a zoom level of "100%".

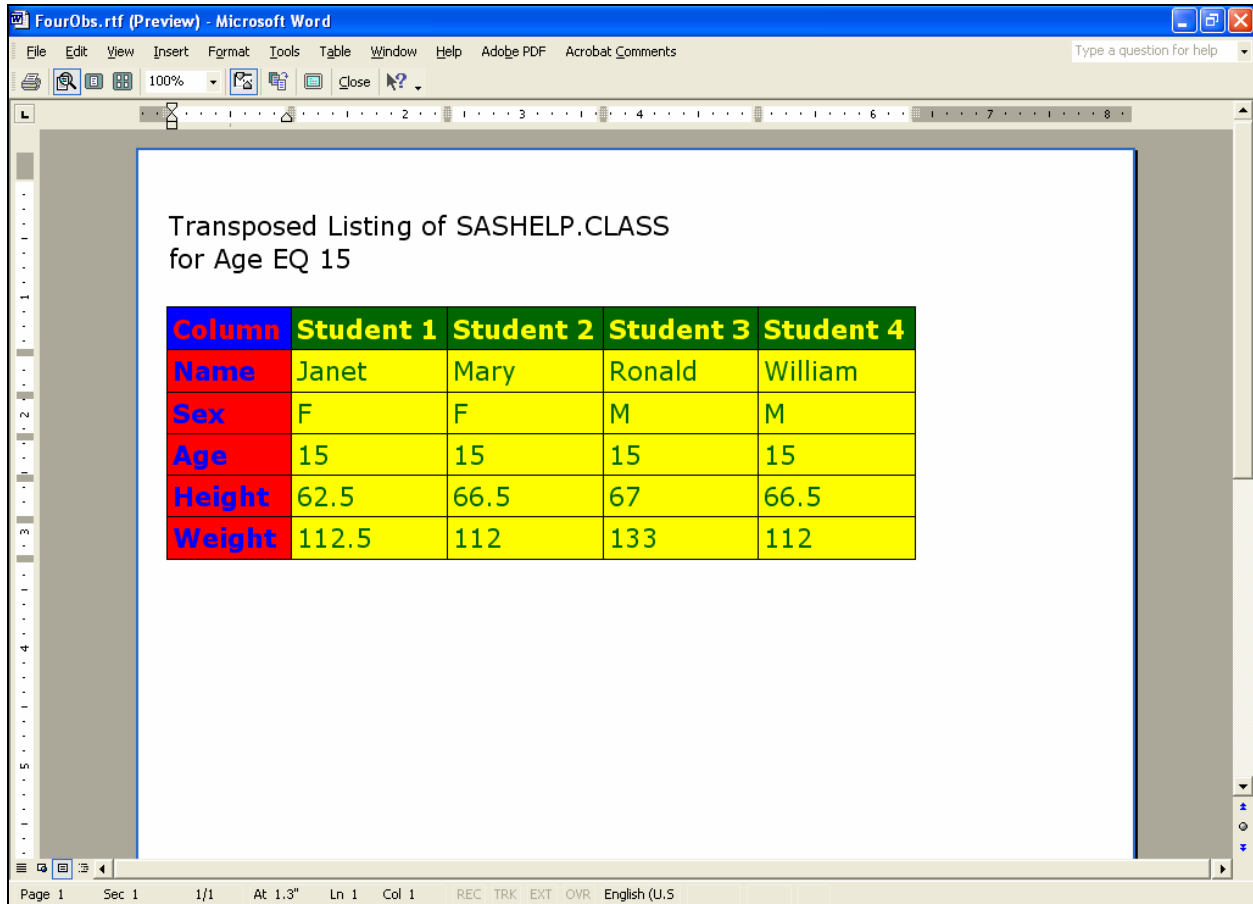
The main content area displays the following text:

Listing of WORK.LABELSCLASS  
for Name EQ 'Carol'

Column Label	Value
Student	Carol
Gender	F
Age in Years	14
Hgt in Inches	62.8
Wgt in Pounds	102.5

## Four Observations in RTF Format

(The data set has no variable labels. The header for the values in each row is customized with the word “Student”.)

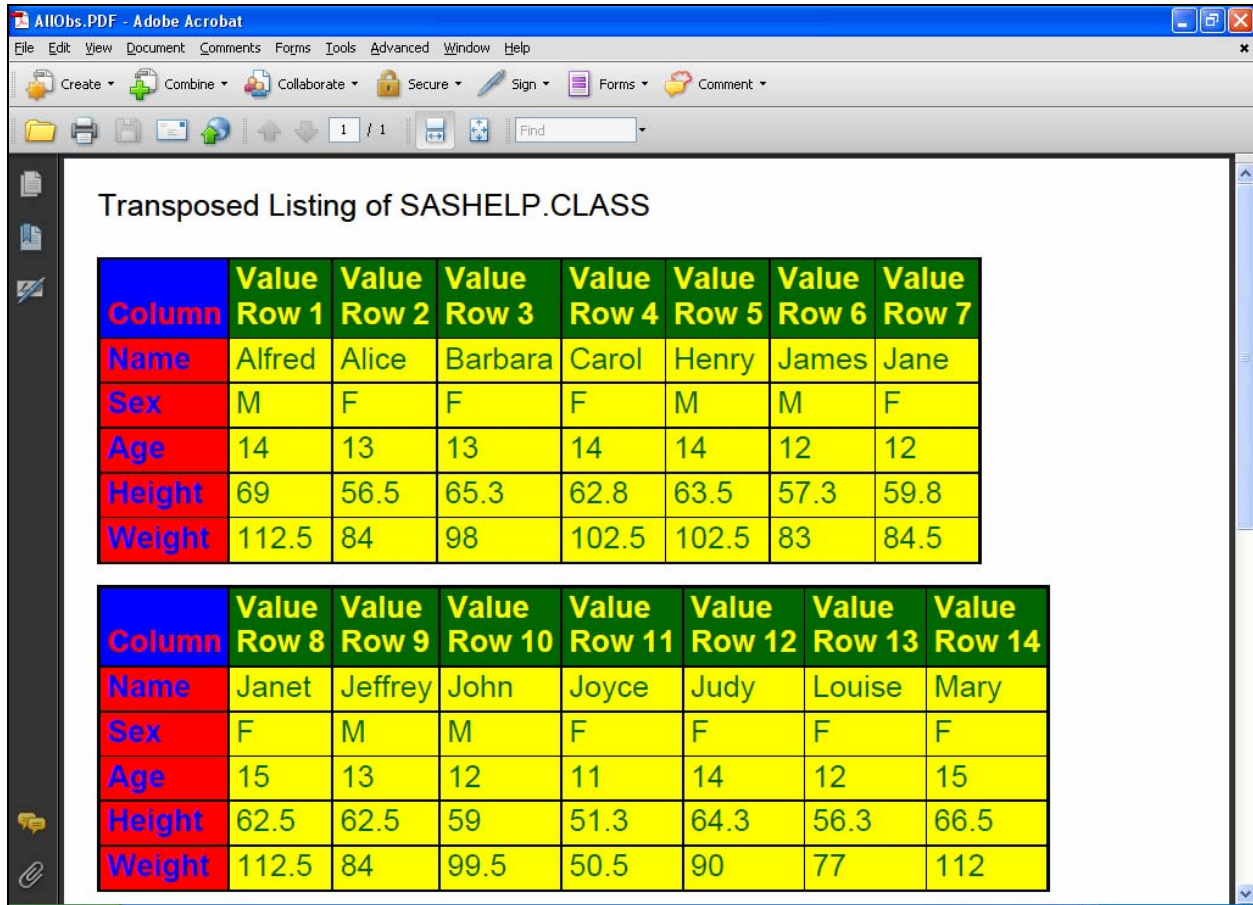


Transposed Listing of SASHELP.CLASS  
for Age EQ 15

Column	Student 1	Student 2	Student 3	Student 4
Name	Janet	Mary	Ronald	William
Sex	F	F	M	M
Age	15	15	15	15
Height	62.5	66.5	67	66.5
Weight	112.5	112	133	112

## Full Listing in PDF Format

(The bottom of the page was trimmed off for publication. Note the repeated column names when the listing wraps.)



Transposed Listing of SASHELP.CLASS

Column	Value Row 1	Value Row 2	Value Row 3	Value Row 4	Value Row 5	Value Row 6	Value Row 7
Name	Alfred	Alice	Barbara	Carol	Henry	James	Jane
Sex	M	F	F	F	M	M	F
Age	14	13	13	14	14	12	12
Height	69	56.5	65.3	62.8	63.5	57.3	59.8
Weight	112.5	84	98	102.5	102.5	83	84.5

Column	Value Row 8	Value Row 9	Value Row 10	Value Row 11	Value Row 12	Value Row 13	Value Row 14
Name	Janet	Jeffrey	John	Joyce	Judy	Louise	Mary
Sex	F	M	M	F	F	F	F
Age	15	13	12	11	14	12	15
Height	62.5	62.5	59	51.3	64.3	56.3	66.5
Weight	112.5	84	99.5	50.5	90	77	112

Here is the code used to produce the reports:

```
%MACRO TransposedPrint(
Data=
,Where=
,PreTitle=
,UseLabelsIfAvailable=YES
,ValueHeadingPrefix=Value_Row
,FontFace=Verdana
,FontSize=16pt
,VarNameHeaderTextColor=CXFF0000
,VarNameHeaderBackgroundColor=CX0000FF
,VarNameTextColor=CX0000FF
,VarNameBackgroundColor=CXFF0000
,ValueHeaderTextColor=CXFFFF00
,ValueHeaderBackgroundColor=CX006600
,ValueTextColor=CX006600
,ValueBackgroundColor=CXFFFF00
,UseCustomColors=YES);

/* Note: In this macro, the use of
   data set variable labels, Where=, and PreTitle= are optional. */

%LET FontFormatting =
%STR(FONT_FACE=&FontFace
FONT_SIZE=&FontSize);
%LET UseCustomColors =
%UPCASE(&UseCustomColors);
%LET Library =
%SUBSTR(&Data,1,%INDEX(&Data,.) - 1);
%LET DataSet =
%SUBSTR(&Data,%EVAL(%INDEX(&Data,.) + 1),
%EVAL(%LENGTH(&Data) - %INDEX(&Data,.) ));
%LET Library = %UPCASE(&Library);
%LET DataSet = %UPCASE(&DataSet);

PROC SQL NOPRINT;
SELECT COUNT(*)
INTO :N
FROM DICTIONARY.COLUMNS
WHERE LIBNAME EQ "&Library"
AND MEMNAME EQ "&DataSet";
SELECT NAME
INTO :COLUMN1 - :COLUMN%TRIM(%LEFT(&N))
FROM DICTIONARY.COLUMNS
WHERE LIBNAME EQ "&Library"
AND MEMNAME EQ "&DataSet";
QUIT;

%MACRO VarList;
%DO I = 1 %TO &N %BY 1;
```

```

&&COLUMN&I
%END;
%MEND VarList;

DATA WORK.ToTranspose;
SET &Library..&DataSet;
%IF %LENGTH(&Where) NE 0 %THEN %DO;
WHERE &Where;
%END;
RUN;

DATA _NULL_;
SET WORK.ToTranspose NOBS=HowMany;
CALL SYMPUT('ValueCount',HowMany);
STOP;
RUN;

PROC TRANSPOSE DATA=WORK.ToTranspose
OUT=WORK.ToPrint;
VAR %VarList;
RUN;

%MACRO Labels;
%DO I = 1 %TO &ValueCount %BY 1;
LABEL COL&I = "&ValueHeadingPrefix &I";
%END;
%MEND Labels;

%MACRO Cols;
%DO I = 1 %TO &ValueCount %BY 1;
COL&I
%END;
%MEND Cols;

%LET LabelVarFound = NO;
%IF %UPCASE(&UseLabelsIfAvailable) EQ YES
%THEN %DO;
PROC SQL NOPRINT;
SELECT NAME
INTO :LabelVarFound
FROM DICTIONARY.COLUMNS
WHERE LIBNAME EQ "WORK"
AND MEMNAME EQ "TOPRINT"
AND NAME EQ "_LABEL_";
QUIT;
%LET LabelVarFound =
%TRIM(%LEFT(&LabelVarFound));
%END;
OPTIONS NOCENTER LINESIZE=MAX PAGESIZE=MAX;
TITLE1 FONT="&FontFace" HEIGHT=&FontSize
%IF %LENGTH(&PreTitle) NE 0 %THEN %DO;
"&PreTitle"

```

```

JUSTIFY=LEFT
%END;
%IF &ValueCount GT 1 %THEN %DO;
"Transposed "
%END;
"Listing of &Library..&DataSet"
%IF %LENGTH(&Where) NE 0 %THEN %DO;
JUSTIFY=LEFT
"for &Where"
%END;
;
PROC PRINT DATA=WORK.ToPrint LABEL SPLIT='_';
%IF &LabelVarFound EQ _LABEL_ %THEN %DO;
ID _LABEL_ /
%END;
%ELSE %DO;
ID _NAME_ /
%END;
STYLE(HEADER) = [&FontFormatting
%IF &UseCustomColors EQ YES %THEN %DO;
BACKGROUND=&VarNameHeaderBackgroundColor
FOREGROUND=&VarNameHeaderTextColor
%END;
]
STYLE(DATA) = [&FontFormatting
%IF &UseCustomColors EQ YES %THEN %DO;
BACKGROUND=&VarNameBackgroundColor
FOREGROUND=&VarNameTextColor
%END;
];
%IF &LabelVarFound EQ _LABEL_ %THEN %DO;
LABEL _LABEL_ = 'Column_Label';
%END;
%ELSE %DO;
LABEL _NAME_ = 'Column';
%END;
VAR %Cols /
STYLE(HEADER) = [&FontFormatting
%IF &UseCustomColors EQ YES %THEN %DO;
BACKGROUND=&ValueHeaderBackgroundColor
FOREGROUND=&ValueHeaderTextColor
%END;
]
STYLE(DATA) = [&FontFormatting
%IF &UseCustomColors EQ YES %THEN %DO;
BACKGROUND=&ValueBackgroundColor
FOREGROUND=&ValueTextColor
%END;
];
%IF &ValueCount EQ 1 %THEN %DO;
LABEL COL1 = 'Value';
%END;

```



```

%ELSE %DO;
%Labels;
%END;
RUN;
%MEND TransposedPrint;

DATA work.LabelsCLASS;
SET sashelp.class;
LABEL Name='Student'
Sex='Gender'
Weight='Wgt in Pounds'
Height='Hgt in Inches'
Age='Age in Years';
RUN;

ODS LISTING CLOSE;
ODS NORESULTS; * Do not open output in SAS *;

GOPTIONS RESET=ALL; * Always do this. *;
ODS HTML PATH="C:\\" (URL=NONE)
BODY="OneLabeledObs.html"
(TITLE="One Labeled Observation");
%TransposedPrint(Data=work.LabelsCLASS
,Where=Name EQ 'Carol');
ODS HTML CLOSE;

GOPTIONS RESET=ALL; * Always do this. *;
OPTIONS NODATE NONUMBER;
ODS RTF FILE="C:\FourObs.rtf";
%TransposedPrint(Data=sashelp.class
,Where=Age EQ 15
,ValueHeadingPrefix=Student);
ODS RTF CLOSE;

GOPTIONS RESET=ALL; * Always do this. *;
ODS PDF NOTOC FILE="C:\AllObs.PDF";
%TransposedPrint(Data=sashelp.class
,FontFace=Helvetica);
ODS PDF CLOSE;

ODS LISTING;

```

## How To Alert Them When They Need To Know

More important than the trendy concept of “dashboards” is the act of notifying interested and/or responsible party immediately when there is a data situation of concern.

Since SAS can send email whenever you want it to, and since SAS can go to sleep when you don't want it to do anything, it is easy to create a system that does continuous (at intervals) monitoring and notifies people if a threshold has been reached. I first developed such an application to notify users of a shared SAS BI server whenever their SAS programs had consumed more than a defined amount of CPU time. (See Reference 2.)

The program below is a much simplified monitor-and-alert demonstration program.

For a real application, replace the TO and SUBJECT assignments in the FILENAME EMAIL statement, and insert your monitor logic at the point indicated. With SAS email you can send messages to a distribution list, use CC and BCC, attach files, imbed a URL in the message body, etc. For more about SAS email, see the SAS Companion Online Doc for your operating system and/or Reference X.

```
%macro
MonitorAndAlertDemo (WaitInSeconds=60,NumberOfChecksIfNotForever=) ;

FILENAME anyname EMAIL
TO = Le_Roy_Bessler@wi.rr.com           /* customize this */
SUBJECT = "Too Many Observations in Data Set"; /* customize this */

%let NumberOfChecks = 0;

%Check:

* START your monitor logic here *;

data _null_;
set sasuser.DataSetToChk nobs=Count;
call symput('ObsCount',trim(left(Count)));
stop;
run;

%if %eval(&ObsCount GT 1)
%then %do;
data _null_;
file anyname;
put "Number of Observations in Data Set is &ObsCount";
run;
%end;

* END your monitor logic here *;

%if %length(&NumberOfChecksIfNotForever) NE 0 %then %do;
  %let NumberOfChecks = %eval(&NumberOfChecks + 1);
  %if &NumberOfChecks EQ &NumberOfChecksIfNotForever
  %then %goto MacroExit;
%end;
```

```

%end;

data _null_;
WaitUntilNextCheck = sleep(&WaitInSeconds);
run;

%goto Check;

%MacroExit:

%mend MonitorAndAlertDemo;

options mprint;

* START of test that will generate NO alert *;
data sasuser.DataSetToChk;
x = 1;
output;
run;

%MonitorAndAlertDemo(WaitInSeconds=10,NumberOfChecksIfNotForever=1);
* END of test that will generate NO alert *;

* START of test that will generate AN alert *;
data sasuser.DataSetToChk;
x = 1;
output;
x = 1;
output;
run;

%MonitorAndAlertDemo(WaitInSeconds=10,NumberOfChecksIfNotForever=1);
* END of test that will generate AN alert *;

```

## Conclusion

I hope that you find some use for the tools provided above. They were fun and easy to develop. Their use offers a big potential return on a small development investment. I started my compact Ranked and Subsetted reporting long before the Age of Twitter.

## References

1. LeRoy Bessler, “Visual Display of Data in the Age of Twitter, Mobility, Web, Excel, and PowerPoint: Concise, Communication-Efficient, and Communication-Effective Graphs”, *Proceedings of the MidWest SAS Users Group Conference 2010*, MWSUG, Inc. (USA), 2010.
2. LeRoy Bessler, “More Ways to Use SAS to Manage, Monitor, and Control SAS or the SAS BI Server: Tools for the SAS User, Server Administrator, or Manager”, *Proceedings of SAS Global Forum 2010 Conference*, SAS Institute, Inc. (USA), 2010. Find this paper on the web at: <http://support.sas.com/resources/papers/proceedings10/279-2010.pdf>

## Author Information

Your questions, comments, suggestions, and examples of other concise tabular SAS reporting methods and tools are welcome.

LeRoy Bessler PhD

Bessler Consulting and Research, Fox Point, Milwaukee, Wisconsin, USA

[Le\\_Roy\\_Bessler@wi.rr.com](mailto:Le_Roy_Bessler@wi.rr.com)

A SAS user since 1978, Dr. LeRoy Bessler has shared his knowledge and experience with other users at conferences throughout the USA and in Montreal, London, Heidelberg, and Dublin. Though a SAS generalist with long experience in Base SAS, SAS macro language, and SAS tools for access to non-SAS data, his special interests include communication-effective visual communication and reporting, web information delivery, highly formatted Excel reporting, SAS/GRAPH®, ODS, creation of unique tools to support the SAS BI server and its users, and Software-Intelligent Application Development for Reliability, Reusability, Extendibility, and Maintainability. He is a regular contributor to *VIEWS News*, the web newsletter of the VIEWS International SAS Programmer Community.

SAS is a registered trademark or trademark of SAS Institute Inc. in the USA and other countries.

® indicates USA registration. Other product and brand names are trademarks or registered trademarks of their respective owners.