Paper 290-2011

Traffic-Lighting Your Reports the Easy Way with PROC REPORT and ODS

Andrew H. Karp
Sierra Information Services
Sonoma, California USA

Abstract

Effective combination of core tools in BASE SAS Software enables you to create effective and compelling reports of data that "traffic-light," or conditionally assign appearance attributes, to specific elements of the output as a function of rules that are either "outside the data set" (e.g., an industry standard or corporate policy/designation) or "inside the data set" (e.g., those data values that are above the average (mean) of all the values in a column are to be displayed in one color, and those below the average are to be displayed in a different color).

Combining appropriate elements of the SAS Output Delivery System, PROC REPORT and, at times, the SAS Format Facility and SAS Macro Language makes enables SAS users to easily generate reports and applications that reduce the number of steps and amount of time needed to create reports implementing both "internal" and "external" traffic-lighting "rules." This paper is designed to explain how these powerful BASE SAS Software elements can be combined together to quickly and accurately create the report you want.

This presentation will be helpful for SAS users who want to learn how to implement "traffic-lighting" to their reports and programs, and who already have a basic understanding of SAS data step programming and the SAS Macro Language. The materials assume you are already familiar with core Output Delivery System concepts such as "destination," "style," and "template." Experience with PROC REPORT is helpful but not required.

Initial Considerations

Perhaps one of the most common "how hard would it be" questions posed to SAS programmers and data analysts is to modify the appearance of your SAS Procedure-generated output based on the some "rule." For example, you may be asked to assign colors to certain ranges of values in the report, or call attention to some rows in the report based on some type of "decision rule." In some cases the "decision rule" may be "external" to the data, such as an "industry standard" or "policy" that "says" some data values are "good" and others "bad." In some of the examples I use in this paper, the "rules" are "acceptable" product rejection and acceptance rates at a hypothetical manufacturing company.

In other situations we are asked to change the appearance of our output based on "rules" that are, for lack of a better term, "in the data" themselves. We may have a reporting application that runs every time the underlying data store changes, which means that values, such as a mean or quantile statistic, change as the data we are reporting/analyzing changes. In this situation the value that "triggers" a change in the report appearance changes each time we run it, and our goal should then be to implement a solution which avoids manual intervention in the process. Instead, we will want to combine several different SAS tools in to an one automated process that does all the work for us.

Unfortunately, it's been my experience that many SAS users do not fully appreciate the wide range of effective and simple to implement tools available to implement efficient "traffic-lighting" processes. Instead, they are still spending excessive amounts of time manually cutting and pasting SAS Procedure output from the Output Window to some other file format and then manually, row by row, and cell by cell, doing the "traffic-lighting" by hand. Many people, it seems, waste valuable time manually "adjusting" the appearance of SAS-generated output during regular and ad hoc reporting projects because they aren't yet aware of ways to combine BASE SAS tools in to flexible and re-useable applications that require no, or at most, very little "maintenance" as the underlying data store from which the report is generated changes.

That's why I wrote this paper. In my experience, by combining the power of PROC REPORT with the SAS Output Delivery processes, and by harnessing the flexibility of the SAS Format Facility, you can easily generate reports that "tell the story" without excessive manual intervention in their generation.

Core Concepts

Mastery of three core "tools" in the SAS System is necessary to start taking advantage of traffic-lighting: 1) the SAS Output Delivery System; 2) PROC REPORT; and, 3) the SAS Format Facility. Adding just a little SAS Macro Language ability makes it even easier to add the traffic-lighting elements you need to your report. Once you understand how these "tools" work, and how they work together, you are well on your way to becoming a "traffic-lighting master" and you can then apply some of the more advanced example I present towards the end of this paper.

Output Delivery System

The SAS Output Delivery System, commonly referred to as "ODS," became a production element of BASE SAS with the initial release of SAS Version 8. It is an "embedded" part of SAS that controls how Procedure-generated output is "delivered" to one or more "destinations," of which the default is the "Listing Destination," or the Output Window for those of you using SAS in a non-batch environment. Since ODS is part of the BASE SAS "install," it is already available to you; there's no need to purchase additional "modules" or "solutions products" to apply its power and flexibility to your projects.

Addition of ODS to BASE SAS has completely revolutionized the way SAS users work with their output because it gives us complete control over what is being generated by our Procedure "steps," where that generated output is "delivered," and what its appearance. Much has been written about ODS and presented at SGF and other events for SAS users. In addition, the BASE SAS Community element of the SAS Customer Support website, support.sas.com, contains a wide range of information about current and anticipated future enhancements to ODS in future SAS Software releases.

In a nutshell, the ODS "receives" one or more Data Components generated by a SAS Procedure Step. (The number of generated Data Components depends on the PROC you are using, and the defaults, options and statements you supply to that Step. PROC REPORT, which is featured in this paper, only generates one data component, which simplifies our discussion of the interrelationship between it and the ODS, fortunately.) ODS then binds a Table Template and a Style Template to the Data Component, forming what is called the Output Object or Output Table. The Table Template structures the Data Components contents in to rows and columns, while the Style Template contains the "rules" governing the output appearance, such as fonts, colors, lines, shading, etc. In SAS 9.2 Software there are 53 "internal" ODS Style Templates, and PROC TEMPLATE can be used to create both customized Style and Table Templates. If a Style Template is not specified in and ODS Statement defining the Destination, the Default Style Template is used when creating the Output Object.

Once the Output Object is created, ODS "delivers" it to all "Destinations" that are open at that point in your program. By default, only the LISTING Destination is open when you start your SAS session. Other destinations can be opened (and closed) as necessary during your program, and multiple destinations can be open simultaneously so that the same Output Object can be "delivered" to multiple output file formats at the same time. For example, it is possible to run one PROC REPORT Step and have its output "delivered" to your Output Window (LISTING Destination), a Portable Document File (PDF Destination) and a Rich Text Format (RTF Destination) file in one, rather than three, "steps."

PROC REPORT

This BASE SAS Procedure was added to SAS in Version 6, and, in my opinion, remains underutilized and under-appreciated by the SAS user community. It combines the data summarization and analysis tools of PROCs MEANS and SUMMARY with the reporting features of PROC TABULATE, an "older" PROC that is still part of BASE SAS Software. PROC REPORT also provides a much richer array of data display capabilities than PROC PRINT, and eliminates the need for "old school" DATA _NULL_ programming for customized report applications. Mastering PROC REPORT's impressive array of capabilities makes it very easy for SAS users to generate the reports they need with a minimum of SAS coding and in fewer "steps" than you would need without it.

While there is some overlap in its capabilities with other BASE SAS reporting and analysis tools, only PROC REPORT can "accept" SAS Programming Language statements in its syntax. It is therefore the only BASE SAS procedure that crosses traditional "boundary" between Data and Procedure Steps, as COMPUTE "block" syntax allows you to include what is often thought of as "data step code" within the PROC REPORT

"task." This procedure therefore combines a powerful suite of SAS data summarization, analysis and display tools WITH a unique array of data computation capabilities that are not "available" in other BASE SAS Module Procedures. It is therefore uniquely suited for applications and reports where "traffic-lighting" it to be employed.

SAS Format Facility

SAS Formats control the display, or "external representation," of the values of variables in a SAS data set. Formats are independent of SAS data sets, and are stored in "Format Catalogs" in a SAS Library. There are over 1,000 "internal" or "SAS-supplied" formats that are part of your standard SAS install. You can create customized formats using PROC FORMAT, in the BASE SAS Module. Mastery of SAS Format Facility capabilities enables SAS users to reduce tedious and error prone tasks such as "Data Step Recodes" or manually replacing "codes" in a data set with text strings explaining what the code's value represents. In the context of "traffic lighting," a SAS Value Format is an easy way to supply the traffic-lighting "rules" governing what colors will be assigned to display the values of variables in output generated by PROC REPORT and delivered via ODS to, for example, a PDF or HTML file.

SAS Macro Language

A core element of BASE SAS Software, the SAS Macro Language enables users to create re-usable, flexible applications using the Macro Language. For our purposes, the two core concepts we need to know about the Macro Language are: 1) Values of "Macro Variables" are stored in a special SAS file called the Macro Symbol Table; and 2) SAS "resolves" references to Macro Symbol Table variables you write in Data and Procedure Steps before carrying out the commands in the step. So, for the purposes of becoming expert traffic-lighters, all we need to remember about the SAS Macro Language and Macro Symbol Table variables is "text substitution." We will supply values to Macro Symbol Table variables at one stage of our project, and then refer to them in subsequent stages, thus eliminating unnecessary programming (or "re-programming") and/or manual intervention in a complex report generation process. There's a lot more that the Macro Language offers SAS users, but this is enough for us to start using it in our traffic-lighting projects.

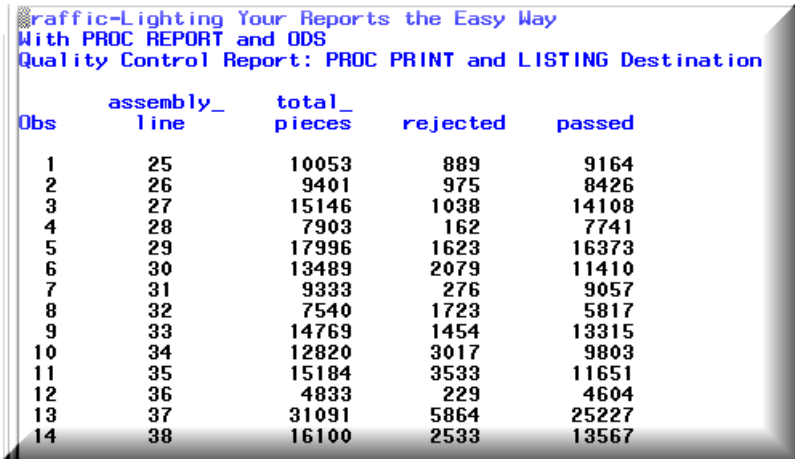
Example 1: PROC PRINT and POLDO (Plain Old Listing Destination Output)

To get started, let's look at the PROC PRINT task shown below in Figure 1A.

```
17 options nodate nonumber nocenter;
18 proc print data=traffic.quality;
19 title "Traffic-Lighting Your Reports the Easy Way";
20 title2 "With PROC REPORT and ODS";
21 title3 'Quality Control Report: PROC PRINT and LISTING Destination';
22 run;
```

Figure 1A: PROC PRINT and POLDO

This task is very straightforward: after using some SAS System OPTIONS to control output display in the Output Window, a PROC PRINT task is executed to display the data portion of a data set containing quality control data from a company that manufactures digital video disks (DVDs). Each row in the data set is displayed by PROC PRINT (or, more formally, "delivered to the Listing Destination") in the Output Window, as shown in Figure 1B below.



Traffic-Lighting Your Reports the Easy Way
With PROC REPORT and ODS
Quality Control Report: PROC PRINT and LISTING Destination

Obs	assembly_ line	total_ pieces	rejected	passed
1	25	10053	889	9164
2	26	9401	975	8426
3	27	15146	1038	14108
4	28	7903	162	7741
5	29	17996	1623	16373
6	30	13489	2079	11410
7	31	9333	276	9057
8	32	7540	1723	5817
9	33	14769	1454	13315
10	34	12820	3017	9803
11	35	15184	3533	11651
12	36	4833	229	4604
13	37	31091	5864	25227
14	38	16100	2533	13567

Figure 1B: PROC PRINT and POLDO

This example shows several inherent limitations of both PROC PRINT and the default Listing Destination. Management is probably going to want two things in this report that PROC PRINT cannot do for us: 1) compute the percentage of pieces that either passed or failed inspection; and, 2) show column totals for total pieces produced, total rejected and total passed at the bottom of the report, as well the overall acceptance and rejection rates across all assembly lines in the factory. While the PROC PRINT's SUM Statement could give us the column statements, this PROC simply cannot divide the values in one column by the values of another column in an existing data, and display the resulting values. That limitation brings us to Example 2: PROC REPORT and the COMPUTE "Block."

Example 2: PROC REPORT and the COMPUTE "Block"

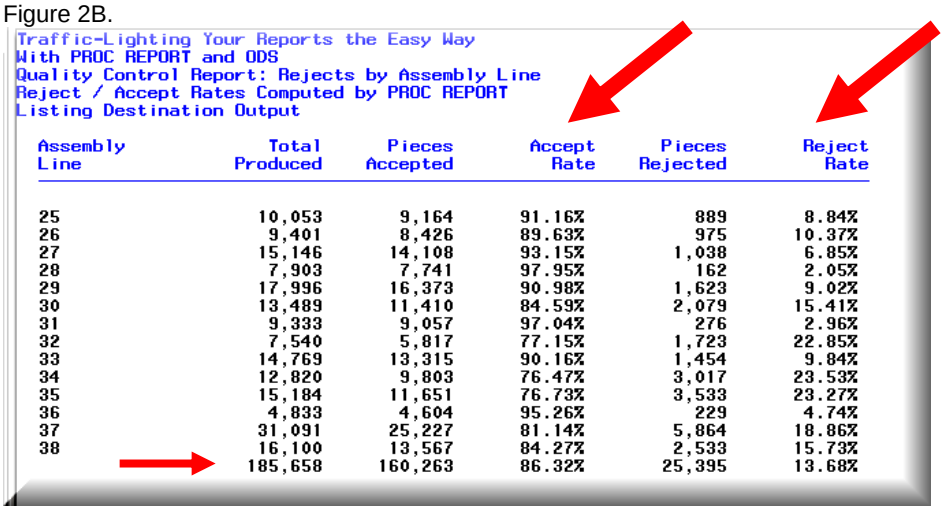
Here is our first example of why PROC REPORT offers an excellent reporting tool to users of BASE SAS Software. In just a few lines of code, a nice report (at this point in our discussion, subject to the severe limitations of the Listing Destination) is created using PROC REPORT syntax.

```
25 proc report data=traffic.quality nowindows headline headskip split='*';
26   column assembly_line total_pieces passed passed_pct rejected rejected_pct;
27   define assembly_line / group 'Assembly*Line' width=12;
28   define total_pieces / analysis 'Total*Produced' width = 12 format=comma10.;
29   define passed / analysis 'Pieces*Accepted' format=comma10.;
30   define passed_pct/computed 'Accept*Rate'format=percent10.2 width=11;
31   define rejected/analysis 'Pieces*Rejected' format=comma10.;
32   define rejected_pct/computed 'Reject*Rate' format=percent10.2 width=11;
33   compute passed_pct;
34      passed_pct = passed.sum / total_pieces.sum;
35         endcomp;
36   compute rejected_pct;
37      rejected_pct = rejected.sum / total_pieces.sum;
38         endcomp;
39 rbreak after/summarize skip; * < generate column-totals;
40 title3 'Quality Control Report: Rejects by Assembly Line';
41 title4 'Reject / Accept Rates Computed by PROC REPORT';
42 title5 'Listing Destination Output';
43 run;
```

Figure 2A: PROC REPORT and the COMPUTE "Block"

The statements in lines 33 to 38 in Figure 2A show examples of how COMPUTE "blocks" are used to assign values to new columns in the report. (Note, however, that these columns only appear in the output generated by the PROC REPORT task, and are *not added* to the existing data set referenced in the DATA= option on line 25.) Also, the RBREAK AFTER Statement on Line 39 instructs PROC REPORT to compute and display the column sums of the all variables in the task that have been defined as either ANALYSIS or

COMPUTED. This one line of code generates the "summary" line at the bottom of the output displayed in Figure 2B.



Traffic-Lighting Your Reports the Easy Way
With PROC REPORT and ODS
Quality Control Report: Rejects by Assembly Line
Reject / Accept Rates Computed by PROC REPORT
Listing Destination Output

Assembly Line	Total Produced	Pieces Accepted	Accept Rate	Pieces Rejected	Reject Rate
25	10,053	9,164	91.16%	889	8.84%
26	9,401	8,426	89.63%	975	10.37%
27	15,146	14,108	93.15%	1,038	6.85%
28	7,903	7,741	97.95%	162	2.05%
29	17,996	16,373	90.98%	1,623	9.02%
30	13,489	11,410	84.59%	2,079	15.41%
31	9,333	9,057	97.04%	276	2.96%
32	7,540	5,817	77.15%	1,723	22.85%
33	14,769	13,315	90.16%	1,454	9.84%
34	12,820	9,803	76.47%	3,017	23.53%
35	15,184	11,651	76.73%	3,533	23.27%
36	4,833	4,604	95.26%	229	4.74%
37	31,091	25,227	81.14%	5,864	18.86%
38	16,100	13,567	84.27%	2,533	15.73%
	185,658	160,263	86.32%	25,395	13.68%

Figure 2B: PROC REPORT and the COMPUTE Block

I've added some red arrows to the screen capture in Figure 2B to help identify what PROC REPORT has been able to do for us so far. Notice that we now have columns in our output with the acceptance and rejection rates computed and displayed for us by the COMPUTE Blocks in Figure 2A. At the bottom of the report, we have the "summary line" showing the total items produced, total accepted and total rejected combined across all rows in the data set. And, we have the overall rejection and overall acceptance rates computed for us too. Of all the reporting PROCs in BASE SAS Software, only the REPORT procedure can perform all of this work for us in one compact step. And, once this code is written, it can be re-used anytime the underlying data set changes. No revisions are necessary in the code if, for example, the number of assembly lines is changed or if the number of items that produced, accepted or rejected per assembly line changes.

PROC REPORT has several other tools, not discussed in this paper, which could "pretty up" the report even further. For an in-depth treatment of PROC REPORT's many capabilities, please see *Carpenter's Complete Guide to the Report Procedure* (SAS Press, 2007), or use the Online Proceedings search facility at sasglobalforum.org

But, we're just scratching the surface when it comes to PROC REPORT and traffic-lighting. Right now we still have POLDO: that "Plain Old Listing Destination Output" that is not very exciting to look at and hard to share with others. And, we still have not changed the appearance of the output based on the values displayed in the report. Before you even think about cutting and pasting our POLDO in to, say, an Excel worksheet and doing the traffic lighting manually before saving your work in Excel as a PDF, let's move on to Example 3: The ODS PDF Destination.

Example 3: The ODS PDF Destination

This example introduces another core tool in the traffic-lighting project: delivery of SAS Procedure-generated output to a "destination" other than the Output Window, or Listing Destination. The code shown in Figure 3A is the same code used to generate the output in Example 2, but with an Output Delivery System "wrapper" around it. I've put quite a few comments in this code (the text colored green) to help newer users of SAS programming syntax what various statements are doing for us. First, the ODS LISTING CLOSE statement on Line 45 instructs ODS to "close" the default Listing Destination. That means no Procedure-generated output will be "delivered" to that "destination" for the duration of my SAS session or, as the case in Example 3, I "re-open" that Destination using the ODS LISTING statement on line 70, at the end of the Figure 3A.

```
45 ods listing close; * << do not send output to LISTING destination;
46 * next line opens PDF destination, eliminates default TOC, assigns STYLE template;
47 ods pdf file = 'C:\traffic_lighting\report1.pdf' style=sasdocprinter notoc;
48 proc report data=traffic.quality nowindows headline headskip split='*';
49   column assembly_line total_pieces passed passed_pct rejected rejected_pct;
50   define assembly_line / group 'Assembly*Line' width=12;
51   define total_pieces / analysis 'Total*Produced' width = 12 format=comma10.;
52   define passed / analysis 'Pieces*Accepted' format=comma10.;
53   define passed_pct/computed 'Accept*Rate'format=percent10.2 width=11;
54   define rejected/analysis 'Pieces*Rejected' format=comma10.;
55   define rejected_pct/computed 'Reject*Rate' format=percent10.2 width=11;
56   compute passed_pct;
57      passed_pct = passed.sum / total_pieces.sum;
58         endcomp;
59   compute rejected_pct;
60      rejected_pct = rejected.sum / total_pieces.sum;
61         endcomp;
62 rbreak after/summarize skip; * < generate column-totals;
63 title3 'Quality Control Report: Rejects by Assembly Line';
64 title4 'Reject / Accept Rates Computed by PROC REPORT';
65 title5 'PDF Destination Output with SASDOCPRINTER Style Template';
66 run;
67 * close PDF destination;
68 ods pdf close;
69 * reopen LISTING destination;
70 ods listing;
```

Figure 3B: The ODS PDF Destination

Let's now take a look at Line 47 in Figure 3B above. This is where I "open the PDF Destination" and give ODS commands on where to save the PDF file it is about to create and which of the 53 SAS-supplied Style Templates it should use to control the appearance of the PROC REPORT-generated output. In this example I specified the SASDOCPRINTER Style, which is available in both SAS 8 and SAS 9. In my opinion, it provides better-looking results than the DEFAULT Style Template when "crisp" black-and-white output is desired. (In SAS 9.1, also check out the JOURNAL Style Template and in 9.2 check out the JOURNAL2 style template as potential 'competitors' to SASDOCPRINTER when generating high-quality black-and-white output. In these examples SASDOCPRINTER was used because it is available to ALL users using Version 8.0 or above.)

Figure 3B (below and to the right) shows the same output generated in Example 2, but now it has been "styled" with the SASDOCPRINTER template and "delivered" to a PDF file. This version of the report is both easy to look at and easy to share with others in the organization.

There are two more things to mention before moving on to the next example. First, you'll notice that the last (summary) line of the report is in italics. That is a function of the specified style template. Second, the software "drivers" that convert SAS output to PDF files is containing within the Output Delivery System, so there's no need to purchase additional software to convert SAS generated output to Portable Document Files.

Traffic-Lighting Your Reports the Easy Way
With PROC REPORT and ODS
Quality Control Report: Rejects by Assembly Line
Reject / Accept Rates Computed by PROC REPORT
PDF Destination Output with SASDOCPRINTER Style Template

Assembly Line	Total Produced	Pieces Accepted	Accept Rate	Pieces Rejected	Reject Rate
25	10,053	9,164	91.16%	889	8.84%
26	9,401	8,426	89.63%	975	10.37%
27	15,146	14,108	93.15%	1,038	6.85%
28	7,903	7,741	97.95%	162	2.05%
29	17,996	16,373	90.98%	1,623	9.02%
30	13,489	11,410	84.59%	2,079	15.41%
31	9,333	9,057	97.04%	276	2.96%
32	7,540	5,817	77.15%	1,723	22.85%
33	14,769	13,315	90.16%	1,454	9.84%
34	12,820	9,803	76.47%	3,017	23.53%
35	15,184	11,651	76.73%	3,533	23.27%
36	4,833	4,604	95.26%	229	4.74%
37	31,091	25,227	81.14%	5,864	18.86%
38	16,100	13,567	84.27%	2,533	15.73%
	185,658	*160,263*	*86.32%*	*25,395*	*13.68%*

Figure 3B: The ODS PDF Destination

Example 4: Assigning Colors to Data Values in the Report

Now that we've seen how easy it is to "deliver" our PROC REPORT-generated analysis to a nice-looking PDF file using ODS, the next step is to enhance the report's appearance by assigning colors to the data values in the acceptance and rejection rate values. Some SAS users labor under the misunderstanding that the "only way" to accomplish this sort of requirement is to transfer (or "export") the PROC REPORT output to a file format that can be opened in a spreadsheet application and then manually assign colors to the data in the cells of that file.

That—misguided, in my opinion--approach offers many drawbacks: 1) it is manually intensive and tedious; 2) these tasks must be carried out each time the report is generated; 3) manual errors in the assignment of appearance elements to the cell are very possible, especially with a long and complex report.

Instead, we can combine the SAS tools we've seen so far with two more capabilities that, taken together, offer a convenient, flexible and easily maintained "traffic-lighting" application. In this example, we will look at how a user-created SAS Value Format and an ODS Style Statement are used to implement a "hands-free" approach to traffic lighting.

Figure 4A shows a PROC FORMAT task that assigns colors to ranges of acceptance and rejection rates for out DVD quality control data in two Value Formats: one "the rules" for acceptance rates and the other with "the rules" for rejection rates. Notice that all I did here was assign a color such as "red" or "blue" to the value ranges in the ACCCEPTF and REJECTF formats. These are examples of three of over 400 Predefined Color Values in the SAS System, and which will be described in further detail later in this paper.

Executing the PROC FORMAT task in Figure 4A creates these formats and stores them in a SAS Formats Catalog. They are now available to use in ANY subsequent SAS task during the current SAS session.

```
142 * create formats for use in traffic-lighting output;
143 proc format;
144    value rejectf
145      low -< .10 = 'green'
146      .10 -< .20 = 'blue'
147      .20 - high = 'red';
148    value acceptf
149    .90 - high = 'green'
150    .80 -< .90 = 'blue'
151    low -< .80 = 'red';
152       run;
```

Figure 4A: Value Formats with Traffic-Lighting "Rules"

We can now implement these "rules" in our PROC REPORT task by adding an ODS Style Statement in the Define Statements for the acceptance and rejection rate columns. (To simplify the example, I only used the one for rejection rates. Figure 4B shows how the DEFINE Statement for the rejected_pct column has been augmented with the proper ODS Style Statement syntax to implement the desired traffic-lighting in the report. This statement over-rides the default display commands in the SASDOCPRINTER Style template for that column. The FOREGROUND = ACCEPTF. instructs ODS to apply the "coloring rules" in that Value Format to the values displayed in that column and FONT_WEIGHT=BOLD instructs ODS to display the values in bold face type.

As it turns out, that's all we need to do to implement our traffic-lighting plan for this report. We can now run the PROC FORMAT and PROC REPORT steps any time we want to generate the report. If the data store from which the report is generated changes, we don't have to do anything to modify the report. As the acceptance and rejection rates change for each assembly line, the combination of PROC REPORT, PROC FORMAT and ODS capabilities dynamically assign the "right" color to the data in the cells of the report. But, if the "rules change" for assigning colors to the rejection or acceptance rates, all we have to do is make the appropriate, but minor, changes to the PROC FORMAT step and re-run it.

```
154 * assign colors to the reject rates;
155 ods pdf file = 'c:\traffic_lighting\Traffic2.pdf' style=sasdocprinter notoc;
156 ods listing close;
157 proc report data=traffic.quality nowindows headline headskip split='*';
158   column assembly_line total_pieces passed passed_pct rejected rejected_pct;
159   define assembly_line / group 'Assembly*Line' width=12;
160   define total_pieces / analysis 'Total*Produced' format=comma10. width = 11;
161   define passed / analysis 'Pieces*Accepted' format=comma10.;
162   define passed_pct/computed 'Accept*Rate'format=percent10.2 width=11;
163   define rejected/analysis 'Pieces*Rejected' format=comma10.;
164 define rejected_pct/computed 'Reject*Rate' format=percent10.2 width=11
165 /* ODS STYLE statement assigns colors to values in this COLUMN */
166           style(column) = [font_weight=bold foreground=rejectf.];
167 compute passed_pct;
168     passed_pct = passed.sum / total_pieces.sum;
169       endcomp;
170   compute rejected_pct;
171     rejected_pct = rejected.sum / total_pieces.sum;
172           endcomp;
173 rbreak after/summarize skip;
174 title 'Traffic-Lighting Your Reports the Easy Way with ODS and PROC REPORT';
175 title2 'Quality Control Report: Rejects by Assembly Line';
176 title3 'Assigning Colors to the Reject Rates';
177 run;
178 ods pdf close;
179 ods listing;
```

Figure 4B: PROC REPORT with ODS STYLE Statement

In my opinion, Example 4 shows how many SAS users can easily avoid unnecessary, tedious, time-consuming and error-prone manual "traffic lighting" exercises and instead let SAS do all the work for you with little or no manual intervention in the report generation process. The results are shown in Figure 4C.

Traffic-Lighting Your Reports the Easy Way with ODS and PROC REPORT
Quality Control Report: Rejects by Assembly Line
Assigning Colors to the Reject Rates

Assembly Line	Total Produced	Pieces Accepted	Accept Rate	Pieces Rejected	Reject Rate
25	10,053	9,164	91.16%	889	8.84%
26	9,401	8,426	89.63%	975	10.37%
27	15,146	14,108	93.15%	1,038	6.85%
28	7,903	7,741	97.95%	162	2.05%
29	17,996	16,373	90.98%	1,623	9.02%
30	13,489	11,410	84.59%	2,079	15.41%
31	9,333	9,057	97.04%	276	2.96%
32	7,540	5,817	77.15%	1,723	22.85%
33	14,769	13,315	90.16%	1,454	9.84%
34	12,820	9,803	76.47%	3,017	23.53%
35	15,184	11,651	76.73%	3,533	23.27%
36	4,833	4,604	95.26%	229	4.74%
37	31,091	25,227	81.14%	5,864	18.86%
38	16,100	13,567	84.27%	2,533	15.73%
	185,658	*160,263*	*86.32%*	*25,395*	*13.68%*

Figure 4C: PROC REPORT with ODS STYLE Statement

Example 5: Assigning Appearance Attributes to an Entire Row

This example discusses how to "trigger" desired appearance attributes for an entire row of your report based on the value in one cell in that row. In this example we will also take our first look at several other very

useful SAS features we can use to implement more complex traffic-lighting projects. Again working with the DVD quality control data, we will look at the use of: 1) PROC REPORT's CALL DEFINE Statement in a COMPUTE "block" to conditionally implement ODS Style Statement commands, 2) additional ODS Style Statement commands; 3) Use of the "%LET" command to store variable values in the Macro Symbol Table (MST) and, 4) specification of an RGB (red/green/blue) or hexadecimal color value.

Before delving in to the details of Figure 5A, let's first review the objectivea of this task. What we want is a report where the appearance of an entire row (results from one assembly line at the DVD factory) is triggered by proportion of disks that passed inspection. If the acceptance rate was 90% or higher then the row is have a background color of green and the text in that row to have a white color. Rows where the acceptance rate is greater than or equal to 80%, but less than 90%, are to appear with a "yellow-ish" background and the text in red.

```
* assign colors to the rows based on acceptance rates;
* assign macro variable constants;
%let color1 = cx336633; * green;
%let color2 = cxffff66; * yellow-ish;

ods listing close;
ods pdf file = 'c:\traffic_lighting\Traffic3.pdf' style=sasdocprinter notoc;
proc report data=traffic.quality nowindows headline headskip split='*';
  column assembly_line total_pieces passed passed_pct rejected rejected_pct;
  define assembly_line / group 'Assembly*Line' width=12;
  define total_pieces / analysis 'Total*Produced' format=comma10. width = 11;
  define passed / analysis 'Pieces*Accepted' format=comma10.;
  define passed_pct/computed 'Accepted*Rate'format=percent10.2 width=11;
  define rejected/analysis 'Pieces*Rejected' format=comma10.;
  define rejected_pct/computed 'Rejected*Rate' format=percent10.2 width=11;
compute passed_pct;
   passed_pct = passed.sum / total_pieces.sum;
   if .9 <= passed_pct <= 1 then do;
     /* use RGB color values stored as macro variable constants */;
   call define(_row_,"style","style=[foreground=white font_weight=bold
                                     background=&color1]");
      end;
else if .8 <= passed_pct < .9 then do;
  call define(_row_,"style","style=[foreground=red font_weight=bold
                                    background=&color2]");
  end;
```

Figure 5A: Assigning Appearance Attributes to an Entire Row

Figure 5A shows the combination of SAS Macro Language commands, ODS Statements and PROC REPORT Syntax necessary to generate the desired output. First, a pair of %LET statements are issued to store the precise RGB color values we want for the rows in our report as values of variables in the MST (Macro Symbol Table). (Please see subsequent examples in this paper for more information about RGB color values and their implementation in the SAS System.) While this is not strictly required, it provides a good introductory example, for our purposes, of supplying values to variables in the MST and then using them in subsequent steps in our project.

The "action" occurs in lines 222 to 232 on the screen capture comprising Figure 5A. This COMPUTE "block" first calculates the proportion of disks passing inspection (i.e., the acceptance rate). Then, that computed value is tested and if it is between .9 and 1.0 (90% to 100%), a CALL DEFINE command is executed to assign the desired appearance attributes for that row in the report. Notice that the background color (see line 227 of the screen capture) is assigned by referring to the stored value of the variable COLOR1 in the MST. And, if the acceptance rate is greater than or equal to .8, but less than .9, a second CALL DEFINE command is executed to implement the desired appearance attributes for those rows. Any row where the acceptance rate is less than .8 is "styled" using the default

Traffic-Lighting Your Reports the Easy Way with ODS and PROC REPORT
Quality Control Report: Rejects by Assembly Line
Assigning Background Colors to the Reject Rates

Assembly Line	Total Produced	Pieces Accepted	Accepted Rate	Pieces Rejected	Rejected Rate
25	10,053	9,164	91.16%	889	8.84%
26	9,401	8,426	89.63%	975	10.37%
27	15,146	14,108	93.15%	1,038	6.85%
28	7,903	7,741	97.95%	162	2.05%
29	17,996	16,373	90.98%	1,623	9.02%
30	13,489	11,410	84.59%	2,079	15.41%
31	9,333	9,057	97.04%	276	2.96%
32	7,540	5,817	77.15%	1,723	22.85%
33	14,769	13,315	90.16%	1,454	9.84%
34	12,820	9,803	76.47%	3,017	23.53%
35	15,184	11,651	76.73%	3,533	23.27%
36	4,833	4,604	95.26%	229	4.74%
37	31,091	25,227	81.14%	5,864	18.86%
38	16,100	13,567	84.27%	2,533	15.73%

Figure 5B: Assigning Appearance Attributes to an Entire Row

appearance attributes in the SASDOCPRINTER Style Template. The results are displayed in Figure 5B.

Example 6: Traffic-Lighting with Graphics Images

This example shows how to conditionally assign the appearance of graphics images in your report as a potential alternative to using some of the coloring approaches demonstrated above. Using a graphics image, such as an arrow, to "point" to rows in your report is, in my opinion, a very effective method by which to draw the reader's attention to "important" parts of the output. This example combines use of the previously-shown %LET Statement to assign a value to an MST variable and CALL DEFINE Statement, as well as the PREIMAGE ODS Style Statement. We will also see a few additional enhancements to the PROC REPORT DEFINE statement.

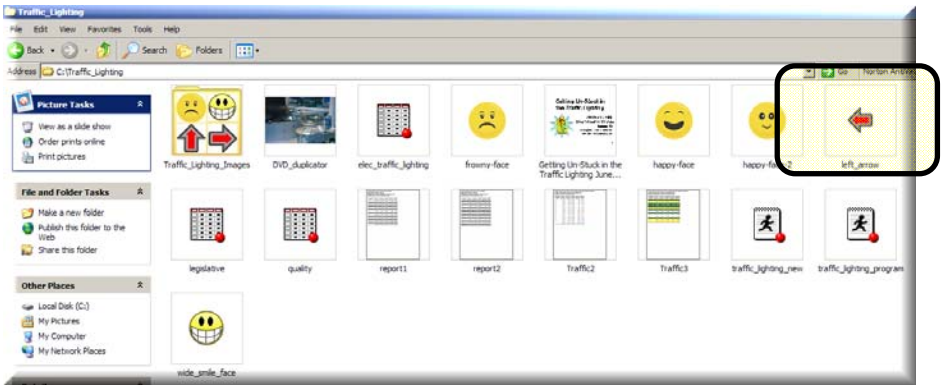


Figure 6A: Traffic-Lighting with Graphics Images

The goal of this reporting effort is to insert a left-pointing red arrow in the rightmost column of the report to draw the reader's attention to assembly lines with a rejection rate of 10% or more. We can easily accomplish this goal by, as with the previous examples, combining several BASE SAS tools that, once completed, give us a "hands-off," reusable reporting application.

Figure 6A shows a capture of a Windows Explorer screen showing the contents of a folder on my computer's hard drive with some graphics images I like to use for my reports. Notice in the upper right hand corner of Figure 6A is a graphics image with the left-pointing arrow I want to introduce to my report. What we need to do is instruct PROC REPORT to "grab" and "insert" that arrow in to my report for those assembly lines (i.e., rows in my report, where the rejection rate is 10% of higher.

The code needed to generate this report is shown in Figure 6B. First, a %LET Statement is used to store a MST variable value with the "path" to the folder storing my graphics images. Again, this is not required, but can save you lots of time, especially if you are writing a complex reporting application with many "calls" to that folder. Next, we have a PROC REPORT step that includes a COMPUTED column called "Arrow." This will be the right-most in the report and will conditionally display the red arrow image stored in the specified folder on my hard drive.

```
242  %let graphics_file = C:\traffic_lighting;
243  ods listing close;
244  ods pdf file = 'c:\traffic_lighting\Traffic4.pdf' style=sasdocprinter notoc;
245  proc report data=traffic.quality nowindows headline headskip split='*';
246    column assembly_line total_pieces passed
247          passed_pct rejected rejected_pct arrow;
248    define assembly_line / group 'Assembly*Line' width=12;
249    define total_pieces / analysis 'Total*Produced' format=comma10. width = 11;
250    define passed / analysis 'Pieces*Accepted' format=comma10.;
251    define passed_pct/computed 'Accepted*Rate'format=percent10.2 width=11;
252    define rejected/analysis 'Pieces*Rejected' format=comma10.;
253    define rejected_pct/computed 'Rejected*Rate' format=percent10.2 width=11;
254    define arrow/computed '  ' center; * create a 'blank' column-header;
255       compute passed_pct;
256          passed_pct = passed.sum / total_pieces.sum; endcomp;
257       compute rejected_pct;
258          rejected_pct = rejected.sum / total_pieces.sum; endcomp;
259  compute arrow/char;
260     if rejected_pct => .10 then do;
261     call define(_col_,"style",'style=[preimage =
262      "&graphics_file\left_arrow.jpg" vjust=center just=center]');
263        end; endcomp;
264  title3 'Display Left Arrow when Reject Rate is 10% or Higher';
265  run;
266  ods pdf close;
267  ods listing;
```

Figure 6B: Traffic-Lighting with Graphics Images

The COMPUTE "block" contains the instructions PROC REPORT will follow to conditionally insert the red arrow image for those rows in the report with a computed rejection rate of 10% or higher. Note that the /CHAR option is used to assign the computed "Arrow" column as character, rather than the default numeric. If that option is not specified PROC REPORT will assume numeric values will be assigned in the column, and the default SAS missing value designator (the period, or "dot") will be displayed in the that column, in addition to the specified graphics. Let me offer you one other caution when implementing this type of traffic-lighting project: you must first re-size the graphics image to fit your report, SAS does not "right-size" it for you. You may therefore need to use a graphics software product such as Adobe Photoshop to first reduce the size of the image you want to use before you can introduce it in to your SAS-generated output.

The results are shown in Figure 6C. Assembly lines with a rejection rates of 10% or more are now "flagged" with a red arrow pointing towards it. As with the other examples shown so far in this paper, it is very easy to re-run this report any time new data become available and have SAS "do the work for you" to determine which rows now "deserve the arrow" and which do not.

Adding a simple graphics element such as the red arrow make it very easy to call the reader's attention to certain rows of the report based on a "decision rule" that you specify.

Traffic-Lighting Your Reports the Easy Way with ODS and PROC REPORT
Quality Control Report: Rejects by Assembly Line
Display Left Arrow When Reject Rate is 10% or Higher

Assembly Line	Total Produced	Pieces Accepted	Accepted Rate	Pieces Rejected	Rejected Rate	
25	10,053	9,164	91.16%	889	8.84%	
26	9,401	8,426	89.63%	975	10.37%	←
27	15,146	14,108	93.15%	1,038	6.85%	
28	7,903	7,741	97.95%	162	2.05%	
29	17,996	16,373	90.98%	1,623	9.02%	
30	13,489	11,410	84.59%	2,079	15.41%	←
31	9,333	9,057	97.04%	276	2.96%	
32	7,540	5,817	77.15%	1,723	22.85%	←
33	14,769	13,315	90.16%	1,454	9.84%	
34	12,820	9,803	76.47%	3,017	23.53%	←
35	15,184	11,651	76.73%	3,533	23.27%	←
36	4,833	4,604	95.26%	229	4.74%	
37	31,091	25,227	81.14%	5,864	18.86%	←
38	16,100	13,567	84.27%	2,533	15.73%	←

Figure 6C: Traffic-Lighting with Graphics Images

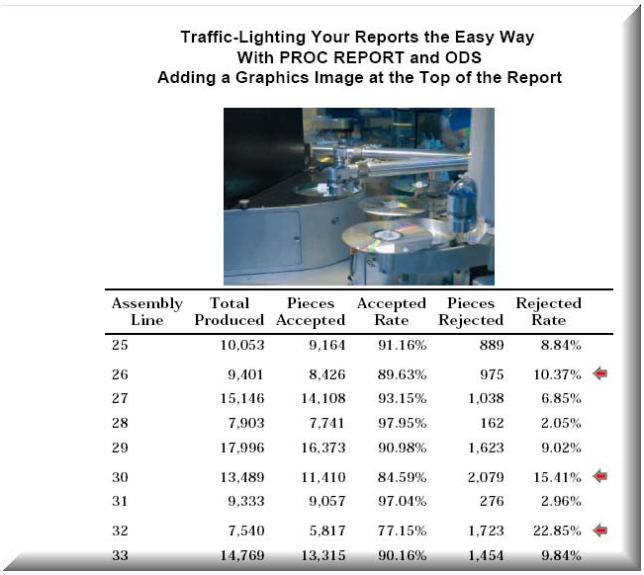
Example 7: Adding a Graphics Image at the Top of the Report

This is not truly a "traffic-lighting" issue, but a question that arises frequently when discussing ODS and SAS Software reporting capabilities. How can I put my company logo, government agency seal, photograph of a product, or some other image at the top of my PROC REPORT-generated output that I am going to "deliver" as a PDF or HTML file to my clients? Placing the PREIMAGE ODS Style Statement in the PROC REPORT Statement easily allows you to place a graphics image at the top of your report.

```
7  ods pdf file = 'c:\traffic_lighting\traffic7.pdf' style=sasdocprinter notoc;
8  proc report data=traffic.quality nowindows headline headskip split='*'
9      style(report)=[preimage="&graphics_file\dvd_duplicator.jpg" vjust=center];
0    column assembly_line total_pieces passed
```

Figure 7A: Adding a Graphics Image at the "Top" of the Report

Figure 7A shows how I added a JPG image of a DVD duplicating machine at the top of the output generated and discussed in Example 6. The VJUST=CENTER option does just that: It centers the image in the middle of the report. The results are shown in Figure 7B. That's all you need to do to add an additional aspect of visual appeal and/or customization to your report. The results are shown in Figure 7B below.



Assembly Line	Total Produced	Pieces Accepted	Accepted Rate	Pieces Rejected	Rejected Rate	
25	10,053	9,164	91.16%	889	8.84%	
26	9,401	8,426	89.63%	975	10.37%	◄
27	15,146	14,108	93.15%	1,038	6.85%	
28	7,903	7,741	97.95%	162	2.05%	
29	17,996	16,373	90.98%	1,623	9.02%	
30	13,489	11,410	84.59%	2,079	15.41%	◄
31	9,333	9,057	97.04%	276	2.96%	
32	7,540	5,817	77.15%	1,723	22.85%	◄
33	14,769	13,315	90.16%	1,454	9.84%	

Figure 7B: Adding a Graphics Image at the "Top" of the Report

Conclusions

Combining PROC REPORT, ODS, PROC FORMAT and other BASE SAS tools enables you to create effective reports that include "traffic-lighting" elements that draw the reader's attention to specific parts of the output. Mastery of the tools and techniques suggested in this paper will enable you to generate the reports you need, quickly and easily, that change as the underlying data stores change. All of the approaches suggested above can be implemented with BASE SAS Software in Version 8.2 and above.

Author contact
Andrew H. Karp
Sierra Information Services
19229 Sonoma Highway PMB 264
Sonoma, California 94115 USA
707 996 7380
Andrew@sierrainformation.com
www.SierraInformation.com