Custom Errors, Warnings, and Notes in SAS®, Plus a Somewhat Practical Use of PROC EXPLODE

Aaron Brown, South Carolina Department of Education

ABSTRACT

This paper discusses how to put (via PUT statements) custom errors, warnings, and notes into the SAS® log. This paper also details a use of the EXPLODE procedure as a way to draw one's attention to important custom notices within the SAS log.

INTRODUCTION

The SAS® log is an incredibly useful tool for debugging your programs. This paper does not talk about how to read or use your SAS log; if you would like advice related to such, please read the paper noted in the Recommended Readings at the end of this paper.

Instead, this paper discusses how to write custom-made errors, warnings, and notes to the SAS log. There are numerous reasons one may want to make additional notes to the log, but two come readily to my mind.

1. If you are writing functions (via the FCMP procedure) or macros (via the macro language) that others may use, you might want to include error-handling in case a user gives invalid input. You could print an error, warning, or note according to the severity of the unexpected input.
2. You can put conditional logic (e.g., IF-ELSE statements) to check for abnormal data in your data sets. If you find an abnormality, you can print a message to the log to help alert you to the abnormality.

In the examples we will use in this paper, we will be checking for data abnormalities. Our example will be a small data set with one variable and four observations. We expect the variable to be a character string of length 4 that contains no numeric characters.

If the length is greater than 4, we will print an error.
If the length is less than 4, we will print a warning.
If the character string contains numeric characters (e.g. 0 or 1-9), we will print a note.

var
AAAA
AAAAA
AAA
4444

Table 1. Sample Data Set

THE BASICS: A PUT STATEMENT

Custom errors, warnings, and notes are generated via the PUT statement.   One of the uses of the PUT statement (but not the only one) is to enable one to write directly to the SAS log.   For example, a DATA step like the below

```
DATA _null_;
     PUT "Write this to the log.";
RUN;
```

yields the following message to the SAS log:

```
26          DATA _null_;
27               PUT "Write this to the log.";
28          RUN;
```

```
Write this to the log.
```
Output 2: Example of PUT Statement Output

While that utility itself can be very useful, we can expand upon it by making the message an error, warning, or note. This is done by appending ERROR:, WARNING:, or NOTE: to the beginning of the text one is printing the log. For environments such as SAS Enterprise Guide, that ensures that the text is noted as an error, warning, or note as well as that it shows up with the proper color formatting.

If we submit the following code

```
DATA _null_;
    SET mydata;
    IF LENGTH(var)>4 THEN PUT "ERROR: too long";
    ELSE IF LENGTH(var)<4 THEN PUT "WARNING: too short";
    ELSE IF ANYDIGIT(var) THEN PUT "NOTE: contains numeric";
RUN;
```

we receive the following notification in the SAS log:

```
ERROR: too long
WARNING: too short
NOTE: contains numeric
NOTE: There were 4 observations read from the data set WORK.MYDATA.
NOTE: DATA statement used (Total process time):
      real time            0.01 seconds
      cpu time             0.01 seconds
```

Output 2: Simple Example of Errors/Warnings/Notes

SLIGHTLY MORE ADVANCED

While the above is helpful to alert us to an issue, it does not greatly facilitate troubleshooting what unexpected value caused the error.   We can modify the PUT statements to include the value of the data anomaly by including the variable name.

Thus, if we submit the following code (note the addition of *var* to the end of each IF statement):

```
DATA _null_;
    SET mydata;
    IF LENGTH(var)>4 THEN PUT "ERROR: too long " var;
    ELSE IF LENGTH(var)<4 THEN PUT "WARNING: too short " var;
    ELSE IF ANYDIGIT(var) THEN PUT "NOTE: contains numeric " var;
RUN;
```

We will receive the following output, which identifies the anomalous data.

```
ERROR: too long AAAAA
WARNING: too short AAA
NOTE: contains numeric 4444
NOTE: There were 4 observations read from the data set WORK.MYDATA.
NOTE: DATA statement used (Total process time):
      real time           0.00 seconds
      cpu time            0.00 seconds
```

Output 3: Advanced Example of Errors/Warnings/Notes

THE EXPLODE PROCEDURE

For a history of the EXPLODE procedure, please read Chris Hemedinger's article in the Recommended Readings. This mostly-useless and perhaps currently unsupported procedure prints large letters to the LISTING output delivery system (ODS). While such can be amusing, it is usually not useful for manipulating data or debugging programs.

However, one somewhat-practical use is to print the EXPLODE output to a temporary text file, then PUT that text file's contents into the SAS log. This will print the large letters to the log, color-coded and flagged as an error, warning, or note if desired. Such could be helpful if we, out of laziness or undue confidence in our program, fail to diligently check the SAS log—it is harder to ignore huge letters (especially ones color-coded red) than it is to ignore a simple note.

Credit where credit is due: this idea is not mine, but was found in the comments on Chris Hemedinger's article. Again, see the Recommended Reading.

For example, you could have this in your SAS log if a critical data abnormality occurs:

Output 4: PROC EXPLODE Output in the SAS Log

As a less extreme example, you may wish to print a Note to the log to remind you to double-check something. Such could be especially useful in a program you run annually or quarterly.

For the full code to generate EXPLODE output to the log, see the Appendix. The code in the appendix prints it as an error, but you can have it as a warning, note, or plain text.

APPENDIX: FULL PROGRAM CODE

```
DATA mydata;
      infile cards;
      input var :$10.;
cards;
AAAA
AAAAA
AAA
4444
;
RUN;
PROC PRINT DATA=mydata noobs; RUN;

*an example of the PUT statement;
DATA _null_;
      PUT "Write this to the log.";
RUN;

*basic error/warning/note example;
DATA _null_;
      SET mydata;
      IF LENGTH(var)>4 THEN PUT "ERROR: too long";
      ELSE IF LENGTH(var)<4 THEN PUT "WARNING: too short";
      ELSE IF ANYDIGIT(var) THEN PUT "NOTE: contains numeric";
RUN;

*advanced error/warning/note example;
DATA _null_;
      SET mydata;
      IF LENGTH(var)>4 THEN PUT "ERROR: too long " var;
      ELSE IF LENGTH(var)<4 THEN PUT "WARNING: too short " var;
      ELSE IF ANYDIGIT(var) THEN PUT "NOTE: contains numeric " var;
RUN;

*PROC EXPLODE code;
FILENAME bigmsg TEMP;
OPTIONS ps=132 nonumber nodate;
ODS LISTING file=bigmsg;
FILENAME FT15F001 temp;
PROC EXPLODE;
parmcards;
1 A BIG ERROR
2 FOR A BIG MISTAKE
;
RUN;
ODS LISTING CLOSE;
DATA _null_;
 INFILE bigmsg;
 INPUT;
 PUT 'ERROR:' _infile_;
RUN;
```

4

ACKNOWLEDGMENTS

The author thanks the SESUG chair and staff for this conference and the opportunity to present.

RECOMMENDED READING

For advice on reading and using the SAS log, see Lora Delwiche and Susan Slaughter's *Errors, Warnings, and Notes (Oh My): A Practical Guide to Debugging SAS Programs*: https://support.sas.com/resources/papers/proceedings/proceedings/sugi28/059-28.pdf

For details on the history and usage of PROC EXPLODE, see Chris Hemedinger's *PROC EXPLODE: The SAS procedure that time forgot*: https://communities.sas.com/t5/SAS-Communities-Library/PROC-EXPLODE-The-SAS-procedure-that-time-forgot/ta-p/473102
One of Mr. Hemedinger's comments contains the code for posting PROC EXPLODE output to the SAS log.

CONTACT INFORMATION

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

Aaron R. Brown
South Carolina Department of Education
1429 Senate Street, Columbia, SC 29201
Work Phone: 803-734-8858
E-mail: ARBrown@ed.sc.gov

SAS and all other SAS Institute Inc. product or service names are registered trademarks or trademarks of SAS Institute Inc. in the USA and other countries. ® indicates USA registration.

Other brand and product names are trademarks of their respective companies.