

XRPT: macros %ID and %SHOW for writing Exception Reports: perform range and logic checks on a data set; write file of exceptions to edit and use for updates

Ronald Fehd, Centers for Disease Control and Prevention, Atlanta GA

ABSTRACT

Data review can be viewed as a two step process:

1. Compare: review differences between two data sets.
2. Exception Report: review data consistency with range and logic checks.

A data set may be updated with a file of the form:

- * if ID = 1 then do;
- * <variable name> = <value>;
- * end;

This paper reviews common problems in writing exception reports for range and logic checking; the product is a file which contains update commands in an easily editable form. Summary information is written at the end of the update file.

INTRODUCTION

Data review is a necessary step in data management. Its purpose is to provide accurate and consistent data for data analysis. While a comparison report (see Fehd 1997, COMPARWS) may identify individual characters or numbers to be updated, the second half of data review consists of identifying values that are out of the range of expected values for a particular variable.

SAS® software provides procedures which may be used to identify outliers. Proc FREQ may be used for character variables; proc UNIVARIATE may be used for numerics.

This paper reports on a demonstration program and template which fulfills these criteria:

1. Concise exceptions report, per observation, easy to read and edit.
2. Exception report is written to file for later editing and subsequent use as an %included update file.
3. Summary report provided for data processing review.

The primary macros are:

- %ID: write ID line only once,
write message to reviewer
- %SHOW: write variable name and value,
using either variable name or array reference

Other macros are provided which facilitate quick, clear and easy writing of code to perform range checks of either

numeric or character data; data may be checked as individual variables or arrays.

Exception reports are unique to a data set and its variables. The test data set provided contains identifiers, individual variables and arrays in both character and numeric variables. The specifications may be tested by assigning either character or numeric variables to macro variables used as program parameters. A macro variable TESTING is provided in order to be able to test any new routines.

Specifications of correction file to be used for updates

In order for a text file to be usable as an %included update file, it must be written in the following format:

- | | |
|-----------------------|----------------|
| * if ID = 1 then do; | *1 ID; |
| * <exception note>; | *2 message; |
| * variable = <value>; | *3 variable; |
| * end; | *4 ID closure; |

1. The ID line opens an if-do statement for an observation with exceptions. This line is to be printed only once. This example shows a single numeric ID. If the ID is character then the ID value must be enclosed in quotes. Similarly, if other secondary identifiers are character, they must likewise be quoted. The first specification is:

- 1.1. *print ID line once per observation.*
- 1.2. *quote ID value(s), according to their type.*

2. The exception message indicates why the variable is being reviewed. The second specification is:

2. *print the exception message as a comment statement.*

3. This program was written to eliminate the errors associated with retyping variable names and values. As with identifiers, variables listed must be quoted, or not, according to their type. The variable assignment specification is:

3. *quote exception values, according to their type.*

4. Every open control statement must have closure. The last specification is:

4. *provide end statement.*

These specifications are implemented in a report writer in step 4.

This XRPTDEMO program contains five sections. The program may be run with the data provided or test data may be inserted into another data set in order to test other macros that call ID and SHOW.

1. Program parameters

TESTING: This macro variable is provided for three purposes: 1. When testing the report is written to file PRINT for quick programmer review. 2. The data set is subset by reducing the number of observations. 3. Invalid values may be inserted into the data by the TESTDATA macro.

ID1, ID2, ID3: Ids are the identifiers or keys of the data set. An exception report is unique to a data set. To add a third ID it is only necessary to enable the macro variable, one line in the ID macro and one line in the summary listings.

DATA_SET: The name of the data set is provided as a macro variable to be used in title and set statements; this ensures that the data evaluated is the correct version.

TITLEn is the number of the last title statement used. It is used to correctly number title lines in the summary.

PRNTFILE is the name of the output file. This file is expected to be edited, have corrections enabled, then used as an %include file for an update. See usage below.

2. Provide variable type to print methods

Macros ID and SHOW are described in object-oriented programming conventions as methods. For these methods to be able to quote character variables the variable type is supplied by creating macro variables from a proc CONTENTS of the data set.

3. Macro definitions

Q returns a single quote for character variables. It is used by both ID and SHOW.

MAKARRAY is used to declare an array of character or numeric variables. It provides the array type to SHOW by declaring a macro variable named with the array name.

ID prints two lines: 1. If the identifier line has not been printed, i.e., if this is the first exception line for an observation, then the line is printed. 2. The exception message is always printed.

SHOW prints a variable and its value, appropriately quoted. The parameter VAR may be either a variable name or an array reference. Where the variable type has been supplied by proc CONTENTS in step 2, the type of an array is supplied by the MAKARRAY macro in step 4.3.

Macros ID and SHOW are used as a pair in this sequence:

```
if <condition> then do;
  %ID(exception message);
  %SHOW(<variable-1> or <array reference>);
  %SHOW(<variable-2>);
end;
```

When a condition refers to two or more variables each variable is to be shown.

The CHK* macros:

To further reduce the number of lines of code in step 4.4, the programmer-written section of the report writer, three examples using ID and SHOW are provided. These examples illustrate the use of different conditions.

CHKARRAY illustrates reviewing an array.

CHKONE is used for individual character variables with a defined set of values.

CHKHIGH may be used with either character or numeric variables to check if the value exceeds a high point. High points for numeric variables may be discovered by using proc UNIVARIATE. Proc FREQ can be used to identify cutoffs for character variables.

4. Report Writer:

4.1. Setup: The data set is saved for use by the summary listings in step 5. The retained variables are used in the brief summary in step 4.5.

4.2. %TESTDATA: changing values to invalid: Test data has been provided with this demonstration program. To facilitate testing this program as it is written and debugged for other data sets the TESTDATA macro: 1. writes output to file PRINT; 2. reduces the number of observations; 3. provides opportunity to insert dummy values.

4.3. array declarations: If CHKARRAY is to be used arrays must be declared with MAKARRAY.

4.4. list of exceptions: The simple part begins here: for each variable it is only necessary to provide the checking macro, variable- or array-name and check-values. Each of the checking macros generates a minimum of 16 statements. Compare the savings in time spent writing and reviewing one page of code with 50 lines for 50 variables or $50 * 16 = 800$ lines: approximately 16 pages of code.

4.5. ID and data step closure, brief summary: This report provides a brief summary at the end of the print file: the number of observations, variables and cells in the data set, the number of identifiers of the exception data, the number of observations of the exception data set, and percentages of each. This data is intended as feedback to the data collection design process. The percentage of cells reported is the exception report rate.

Tutorials

Example of detail with summary data (from test data).
Exception report rate is at lower right: Pcnt : Cells

```
* if NID1 = 3 then do;
*   XRPT MSG;
*   D3      = '.' ;
*   end;*XrptInId =1 ;
***smry : Ids      Vars      Cells;
*** Base:  4       8       32;
*** Diff:  3       4;
*** Pcnt:  75%     12.5%;
```

5. Summary listings

When the report is written to a print file, the following data sets may be enabled to be printed: Exception-Identifiers, and Exception-Variables.

A list of the exception data set, by variable name, is provided so that errors within a variable may be reviewed.

Usage:

In a production environment, an exception report program is used to write and print both a correction file and a paper report. The paper report is reviewed by colleagues responsible for data collection. The marked up copy is returned and the desired corrections are enabled in the file by removing the asterisks in front of the appropriate statements. These updates are applied to the data and another data set created. A comparison report (Fehd 1997, COMPARWS) is used to verify that all indicated corrections were made.

Program XRPT

filename XRPT<filespec>;

note: writes report to file, print output for review.

Process: review report, note corrections to be made.

Process: edit corrections file, enable corrections.

Program UPDATE4: apply corrections

DATA <UPDATE4>;

set <DATA3>;

%include XRPT;

Program COMPAR34: confirm corrections

%COMPARWS(LIBRARY.<DATA3>

```
/* *****
* PROGRAM: XRPTDEMO: Exception Report demo, with macros ID and SHOW
* macro:  Q      : returns single quote for character vars
*          MAKARRAY: declare array, save type of array in mac-var
*          ID      : prints ID line once, exception message
*          SHOW    : prints variable or array-reference
*          CHKARRAY: check each value in array
*          CHKONE  : check one variable is in range
*          CHKHIGH : check one variable is less than high
* DESCRIPTION: data _NULL_ report writer reads data set
*              writes correction file with brief summary
*              writes summary listings of exceptions
* PROCESS: 0. make test data from cards
*          1. set program parameters
*          2. provide variable type, in (N,C), to print methods
*          3. macro definitions
*          4. data step: report writer
*          4.1. initialize vars
*          4.2. if testing, change values to invalid
* *****
```

,LIBRARY.<UPDATE4>

,IDNMBR);

note: print output for review and confirm update.

CONCLUSION

Many different procedures may be used to identify data that is not in the expected set of values for a particular variable. This procedure aids the data review process by writing exceptions to a file where corrections can be made by removing asterisks. This eliminates retyping of variables with their values and the associated edit and correct cycle.

REFERENCES

Fehd, Ronald (1997), "COMPARWS" *Proceedings of the Fifth Annual Conference of the SouthEast SAS Users Group*.

SAS is a registered trademark of SAS Institute, Inc. In the USA and other countries. ® indicates USA registration.

Author: Ronald Fehd e-mail: RJF2@cdc.gov
Centers for Disease Control MS-G25
4770 Buford Hwy NE
Atlanta GA 30341-3724 voice: 770/488-4316
SAS-L archives: send e-mail
to: SAScontrib@SASserv.uga.edu
for %COMPARWS subject: cntb0034: download
for XRPT subject: cntb0039: download

ACKNOWLEDGMENTS

These routines were developed over a period of five years while I crunched the numbers of sample and questionnaire survey data collected by the Model Performance Evaluation Program (MPEP) of the Division of Laboratory Systems, Public Health Practice Program Office of the Centers for Disease Control, and Prevention, Atlanta, Georgia. John Hancock, chief, Information Services Activity, encouraged me to write up these routines. I wish to thank Sharon Blumer, David Cross, Thomas Hearn, and William Schalla of the MPEP group for their data collection.

```
* *****
* ***programmer-written section ***** begin *
* 4.3. array declaration(s), if any
* 4.4. list of exceptions
* ***programmer-written section ***** end *
* 4.5 data step closure, brief summary
* 5. summary listing(s)
* NOTES: ID and SHOW are methods for printing variables
*        variable type is made available to these methods
*        thru mac-vars created from proc CONTENTS data
*        or mac-vars created by MAKARRAY
* KEYWORDS: "data _NULL_ report writer" "exception report"
*           "object methods"
* Author: Ronald Fehd, B.S. C.Sc                      e-mail: RJF2@cdc.gov
* Centers for Disease Control MS-G25
* 4770 Buford Hwy NE
* Atlanta GA 30341-3724                      voice: 770/488-4316
* *****
/*%0. TEST DATA    disable after first run *****
data XRPTDEMO; infile cards;
```

