

INVALID: a Data Review Macro Using PROC FORMAT Option OTHER=INVALID to Identify and List Outliers

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

ABSTRACT

Data cleansing, or as it is more euphemistically known: data review, often occupies too much of a programmer's time and energy. With a properly written data dictionary, a data set will contain appropriate formats for each variable; one can then cut and paste the format definitions into proc FORMAT value statements and label all outliers with the value statement option: other="INVALID". This routine combines a FORMATS catalogue and CONTENTS data set, then uses that information to write data steps which select all outliers. Reports are written, by identifier, for print review and to file, for later use as includes for updating purposes. Expected audience is intermediate and advanced programmers and macro users.

INTRODUCTION

Data review consists of several steps:

1. Determining acceptable values
2. Finding unacceptable values
3. Comparing unacceptable values with data collection form
4. Deciding whether to change the data
5. Recording changes
6. Updating the data

This macro is a method for a data set. Its necessary parameters are a data set name, and a list of identifiers. It is assumed that variables have formats assigned to them, and that the formats have the option other="&INVALID.". Output from the macro includes both summary and detail reports. The primary detail report is written to a file. This file can be edited to record changes to the data and then used as a %include file to update the data.

This macro facilitates step 2: finding all unacceptable and invalid values. However these errant values must be identified in their respective proc FORMAT value statements with the option: other="&INVALID.".

Decisions as to whether to change data – step 4 – are eased by macro INVALID, with its summary reports. Frequencies of both variables and identifiers provide an overview of difficulties in the data set. In addition, a percentage-of-invalid report is attached to the detail, by identifier, report.

Editing the detail, by identifier, report, which is written to a file, addresses the difficulties inherent in step 5. where variable names may be spelled incorrectly, etc.

Finally, updating the data – step 6 – is accomplished easily, using the edited detail report file as a %include file.

DISCUSSION

There must be 50 ways to review variables. Both Fehd (1998) DemoXrpt and Cody(1999) demonstrate using the proc FORMAT value option other='INVALID'. Fehd ensures standardization of the label by using a global macro variable INVALID. Cody and Handsfield (1998) provide ways to mathematically find outliers of numerical variables. McQuown (2000) offers his experience in writing a data review of a large survey.

These authors provide ideas and tools to write customized data review. This macro automates half of that task: it does intra-variable checking, but not inter-variable checking: Doing logic checks between pairs of variables, is a task left for another paper.

The report writer section of INVALID follows that developed in Fehd (1998) DemoXrpt and Fehd (1998) COMPARWS.

The relation of data dictionary and the proc FORMAT program is discussed in Fehd (2001) Beginner's Tour.

How it works

The pseudo-code for macro %INVALID is:

1. For all variables of a data set
2. Choose all observations with invalid values
3. Print summary and detail reports

1. For all variables of a data set

Proc CONTENTS output data set provides variable names, formats and type. This information is also available from SASHELP.VCOLUMN and proc SQL: DICTIONARY.COLUMNS. Refer to lines 164:167.

2. Choose all observations with invalid values

Invalid values are defined in each value statement of a proc FORMAT program. Refer to the Test Data beginning at line 425. This routine depends on having a global macro variable defined as:

```
%LET INVALID = INVALID;
```

This is used to standardize all the value statement options:

```
other = '&INVALID.'
```

Macro INVALID identifies all format values with that label in the Control-Out output data set of proc FORMAT. This list of format names is used to create a format in the WORK library named \$INVALID. Lines 192:220.

The CONTENTS data set is used to write a series of macro calls to the data review macro: CHK4NVLD in lines 224:243. Before these calls are executed, the report data set INVALID structure is created at lines 249:262.

This CHK4NVLD macro implements a choose with the phrase:

```
put(Var,Format.) eq '&INVALID.'
```

at line 413. If any invalid observations are output to the DETAILS data set, then the DETAILS data set is appended to the report data set INVALID. See line 421:423.

3. Print summary and detail reports

Four reports are provided, with parameters to enable each:

Report	at lines:	parameter
1. Summary: Identifiers	340:344	SMRYIDS
2. Summary: Variables	346:350	SMRYVARS
3. Details: by Variable	352:362	SMRYNAME
4. Details: by Identifiers	292:337	DETAILS

Examples of the parameters and reports are given in the Test Data.

Assumptions

INVALID provides named parameters for the location of the data set and the format library. These are both assumed to be in
LIBNAME LIBRARY '<libref>';
The Test Data section – lines 450:467 – illustrates accessing a data set and formats in the WORK library.

Gotcha! Step 0: Checking assumptions

As I began testing on my production data I found that reports from INVALID did not match my custom-written exception reports. Imagine my surprise when I examined my data sets and found that variables had formats that were not in the data dictionary and there were formats present in the proc FORMAT program that were not attributed to any variable. In order to compare the data set formats with the format in the format library I have added a special report at lines 106:155. Refer to the Test Data section for an example.

Notes

The macro is written according to Fehd (2000) Writing for Reading SAS Style Sheet. Fehd (1997) %ARRAY contains the ARRAY macro. Fehd (2001) Beginner's Tour contains the NOBS macro.

Summary

The routine depends on the value of a global macro variable INVALID. This macro variable is appropriately set in autoexec.sas. See Fehd (2001) Macro Tour for further discussion. Two separate programs use this macro variable: one containing proc FORMAT value statements, and another which calls %INVALID.

CONCLUSION

A properly written data dictionary contains the other='INVALID' option on all format values. This facilitates data review.

Date review was and is time consuming. This routine provides a comprehensive report of invalid values in a data set. The summary reports provide a valuable overview of which variables and which identifiers have invalid values. If data cleansing is necessary, the detail report can be edited and %included to perform an update. This file then contains a record of the updates for later referral.

REFERENCES

Cody, Ron, *Cody's Data Cleaning Techniques Using SAS® Software*, Cary, NC: SAS Institute Inc., 1999. 226 pp.

Fehd, Ronald, *%ARRAY: construction and usage of arrays of macro variables*. Proceedings of the 22nd Annual SAS® Users Group International Conference, Cary, NC: SAS Institute Inc., 1997.
<http://www2.sas.com/proceedings/sugi22/CODERS/PAPER80.PDF>

Fehd, Ronald, *%COMPARWS: Compare with summary: a macro using proc COMPARE to write a file of differences to edit and use for updates*. Proceedings of the 23rd Annual SAS® Users Group International Conference, Cary, NC: SAS Institute Inc., 1998.
<http://www2.sas.com/proceedings/sugi23/Posters/p170.pdf>

Fehd, Ronald, *DEMOXRPT: macros for writing Exception Reports: perform range and logic checks on a data set; write file of exceptions to edit and use for updates*. Proceedings of the 23rd Annual SAS® Users Group International Conference, Cary, NC: SAS Institute Inc., 1998.
<http://www2.sas.com/proceedings/sugi23/Appdevel/p7.pdf>

Fehd, Ronald, *A Beginner's Tour of a Project using SAS® Macros Led by SAS-L's Macro Maven*, Proceedings of the 26th Annual SAS® Users Group International Conference, Cary, NC: SAS Institute Inc., 2001.
<http://www2.sas.com/proceedings/sugi26/p066-26.pdf>

Handsfield, James, *CHEKOUT: A SAS® Program to Screen for Outliers*. Proceedings of the 23rd Annual SAS® Users Group International Conference, Cary, NC: SAS Institute Inc., 1998.
<http://www2.sas.com/proceedings/sugi23/Posters/p197.pdf>

McQuown, Gary, *SAS® Macros Are the Cure for Quality Control Pains*. Proceedings of the 13th Annual Northeast SAS® Users Group Conference Cary, NC: SAS Institute Inc., 2000.

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: 770/488-8102**

ACKNOWLEDGMENTS

This macro is the result of a decade of data cleaning efforts. I'd like to thank my colleagues at CDC for all their dirty data. I couldn't have done it without you.

001	;/*****	2001May19	001
002	MACRO: INVALID	NOTE: uses macros ARRAY, NOBS	002
003		NOTE: uses global mac-var INVALID	003
004	method for data set + format library		004
005	where user-written formats have OTHER="&INVALID."		005
006			006
007	USAGE: 1.1 %INVALID();	/*list formats: &LIBRARY.._ALL_ and &FMTLIB;	007
008	1.2 %INVALID(DATA);	/*list formats: &LIBRARY..DATA and &FMTLIB;	008
009	2.1 %INVALID(DATA,IDO);	one ID	009
010	2.2 %INVALID(DATA,ID1 ID2);	two IDs	010
011	2.3 %INVALID(DATA,IDO,PRNTFILE=<fileref>);	write report to file	011
012	2.4 %INVALID(DATA,IDO,TESTING=1);	testing	012
013	2.5 TITLE1 'title for this project';		013
014	%INVALID(DATA,IDO,TITLEN=2);		014
015			015
016	DESCRIPTION: read	all formats in FMTLIB	016
017	choose	formats with OTHER="&INVALID."	017
018	make	format \$INVALID	018
019	review	all variables/columns in data set	019
020	choose	invalid values	020
021	write	file of invalid values	021
022		to be used later as an %INCLUDE file for updates	022
023	read	and print file of invalid values	023
024	print	summary report(s) of invalid values	024
025	search for <%**>		025
026	PROCESS: 1. if IDLIST blank,	print list of formats exit	026
027	2. else: make %ARRAY of ID(s)		027
028	3. make %ARRAY of ID(s)'s attributes		028
029	4. if not exist WORK.FORMATS.\$INVALID then create:		029
030	4.1. get all formats with Label=INVALID		030
031	4.2. if none w/Label=INVALID, print exit-msg exit		031
032	4.3. prepare proc FORMAT CNTLIN data set for value \$INVALID		032
033	4.4. make proc FORMAT value \$INVALID		033
034	5. read CONTENTS, write macro calls		034
035	6. prepare empty data set INVALID for proc APPEND		035
036	7. execute Check-for-Invalid: build data set INVALID		036
037	8. if INVALID empty, print exit-msg exit		037
038	9. make values used in summary		038
039	10. write corex to PRNTFILE		039
040	11. if wanted, print summary report(s)		040
041	12. if wanted read and print PRNTFILE		041
042	13. macro: Check-for-Invalid:		042
043	if obs w/invalid values, append to data set INVALID		043
044			044
045	KEYWORDS: %ARRAY %NOBS		045
046	"invalid values" "data review" "data checking"		046
047	SESUG 2001: invalid outlier FORMAT fmtlib other=		047
048			048
049	NOTE: expected number of ID vars is 2, see %LOCAL DIM_IDS, need as parm?		049
050	NOTE: this warning is acceptable		050
051	WARNING: Variable VALUENUM has different lengths on BASE and DATA files		051
052	(BASE 8 DATA 4).		052
053			053
054	Author: Ronald Fehd, B.S. C.Sc	e-mail: RJF2@cdc.gov	054
055	Centers for Disease Control MS-G25		055
056	4770 Buford Hwy NE		056
057	Atlanta GA 30341-3724	voice: 770/488-8102	057
058	RJF2 99Feb25 begun	NOTE: lines 059:080 change notes deleted	058
081	;/*****	*/	081
082	%MACRO INVALID(/*	*/	082
083	DATA	/*data set name	083

```

084 ,IDLIST =. /* list of primary-key(s) == by-vars */ 084
085 ,LIBRARY =LIBRARY/*libname of DATA */ 085
086 ,FMTLIB =LIBRARY/*libname of catalog FORMATS */ 086
087 ,PRNTFILE=PRINT /*output destination, default = PRINT ** 087
088 /*may be 'external-file' -- note quotes -- or fileref*/ 088
089 ,PRNTLIST=PRINT /*turn off detail listing w/PRNTLIST=NULL */ 089
090 ,DETAILS =1/*?print details? used when only summary wanted */ 090
091 ,SUMMARY =1/*?print any of SUMMARY report(s)? */ 091
092 ,SMRYIDS =1/*?print FREQ of IDS? */ 092
093 ,SMRYNAME=1/*?print detail of variables, by name? */ 093
094 ,SMRYVARS=1/*?print FREQ of variables with invalid? */ 094
095 ,TESTING =0/*?enable test msg and prints? */ 095
096 ,TITLE =2/*TITLE line # */ 096
097 )/*store des = 'method to list INVALID values in data set'/*.....*/ 097
098 ;%LOCAL LENLABEL; %LET LENLABEL=60;%*data INVALID & DETAILS; 098
099 %LOCAL DIM_IDS; %LET DIM_IDS = 2;%*how many IDs?; 099
100 %LOCAL VALIDVARNAME; %*reset at exit; 100
101 %LET VALIDVARNAME=%sysfunc(getoption(validvarname,keyword)); 101
102 OPTIONS ValidVarName = UPCASE;%*align CONTENTS & user-supplied IDLIST; 102
103 %IF &TESTING %THEN %DO; OPTIONS mprint; %END; 103
104 %ELSE %DO; OPTIONS nomprint; %END; 104
105 105
106 %**1. if IDLIST blank, print list of formats w/INVALID, exit; 106
107 %IF "&IDLIST." eq "." %THEN %DO;%*-----; 107
108 %IF "&DATA." eq "" %THEN %DO; %LET DATA = _ALL_; %END; 108
109 proc CONTENTS data = &LIBRARY..&DATA. 109
110 memtype = data 110
111 noprint 111
112 out = CONTENTS 112
113 (keep = Format 113
114 where = (Format not in (' ','$CHAR'))); 114
115 proc SORT data = CONTENTS 115
116 nodupkey; 116
117 by Format; 117
118 proc FORMAT library = &FMTLIB. 118
119 cntlout = FMTLIB 119
120 (keep = FmtName Label Type 120
121 where = (Label = "&INVALID.") 121
122 rename = (FmtName = Format ) ); 122
123 %IF &TESTING %THEN %DO;*PROC PRINT;%END; 123
124 proc SORT data = FMTLIB 124
125 nodupkey; 125
126 by Format; 126
127 127
128 data FMTLIB; 128
129 drop Label Type; 129
130 *length Format $ 9; 130
131 set FMTLIB; 131
132 if Type = 'C' then Format = '$' !! Format; 132
133 133
134 proc SORT data = FMTLIB; 134
135 by Format; 135
136 136
137 DATA FMTLIB; 137
138 do until(EndoFile); 138
139 merge CONTENTS(in = haveC) 139
140 FMTLIB (in = haveF) 140
141 end = EndoFile; 141
142 by Format; 142
143 Data = HaveC; 143
144 FmtLib = HaveF; 144

```

```

145 if      haveC and      haveF then Ok = 'OK';                                145
146 else if haveC and not haveF then Ok = '??';                                146
147 else                                     Ok = '..';                                147
148                                     output;                                end; 148
149                                     stop;                                stop; 149
150 proc PRINT      data = FMTLIB noobs;                                150
151      TITLE&TITLE. "INVALID: compare formats in DATA:"                                151
152      "&LIBRARY..&DATA. and FMTLIB:&FMTLIB.";                                152
153 proc DATASETS library = WORK nolist;                                153
154      delete      CONTENTS FMTLIB (memtype = data);                                quit; 154
155 %GOTO ENDOMACR;%*..... %IF IDLIST EQ "."; %END; 155
156 %*ELSE *DO: IDLIST not BLANK;                                156
157 %**2 make %ARRAY of ID(s);                                %local I;                                157
158 %DO I = 1 %TO &DIM_IDS;                                %local IDS&I.;                                %END; 158
159 %LET IDLIST = %upcase(&IDLIST.);                                %*proc CONTENTS.Name is upcase *;                                159
160 %ARRAY(IDS,&IDLIST);run;                                %*ARRAY mac-vars are local;                                160
161 %DO I = 1 %TO &DIM_IDS;                                %local TYPE&I. Q&I. LEN&I.;                                %END; 161
162                                     162
163 %**3 make %ARRAY of ID(s) attributes;                                163
164 proc CONTENTS data = &LIBRARY..&DATA.                                164
165      noprint                                165
166      out = CONTENTS                                166
167      (keep = Format Formatd Formatl Length Name Type VarNum);                                167
168                                     168
169 DATA      _NULL_;                                169
170 length Q $ 1;                                170
171 do until(EndoFile);                                171
172 set CONTENTS (where=(Name in (                                172
173 %DO I = 1 %TO &DIM_IDS.;                                "&IDS&I"                                173
174 %IF &I lt &DIM_IDS %THEN                                ,;                                %END; 174
175 ))) end = EndoFile;                                175
176 %* Type:: (1:numeric, 2:character) from proc CONTENTS.Type*;                                176
177 if Type = 1 then do; C_N = ' '; Q = ' ';                                end; 177
178 else do; C_N = '$'; Q = '"';                                end; 178
179 %DO I = 1 %TO &DIM_IDS.;                                179
180 if Name = "&IDS&I." then do; call symput("TYPE&I.",C_N);                                180
181 call symput( "Q&I.",Q );                                181
182 call symput( "LEN&I.",                                182
183 compress(put(Length,3.))); end; 183
184 %**DO I=1; %END; 184
185 %*..... do until(EndoFile)*; end; 185
186 stop; 186
187 %**4 does format $INVALID? exist?, because of previous use of macro?; 187
188 %*func catalog-exist(LibName.MemName.ObjName.ObjType);                                188
189 %IF not %sysfunc(cexist(WORK.FORMATS.INVALID.FORMATC)) %THEN %DO;%*---*; 189
190                                     190
191 %**4.1 get all formats which have Label=INVALID;                                191
192 proc FORMAT library =&FMTLIB                                192
193      cntlout = FORMAT_CNTL                                193
194      (keep = FmtName Label Type                                194
195      where = (Label = "&INVALID."));                                195
196      %*note: global &INVALID;                                196
197 %**4.2 if none w/Label=INVALID, exit;                                197
198 %local ANY_FRMT;                                %NOBS(ANY_FRMT);run;                                198
199 %IF not &ANY_FRMT %THEN %DO; %PUT no formats with label=<&INVALID.>; 199
200 %GOTO ENDOMACR;                                %END; 200
201                                     201
202 %**4.3 make proc FORMAT CntlIn data set for value $INVALID;                                202
203 DATA      FORMAT_CNTL      (drop=Type);                                203
204 retain Label '1'                                %*one==True;                                204
205      FmtName '$INVALID'                                205

```

```

206      HLO      ' ';                                206
207 do until(EndoFile);                                207
208   set FORMAT_CNTL (keep = FmtName Type              208
209                      rename = (FmtName = Start))    209
210                      end = EndoFile;                210
211   if Type = 'C' then Start = '$' !! Start;          211
212   output;                                           212
213   HLO = '0';                                       %*do until(EoF);end; 213
214   Label = '0';                                       %*Oh==Other;         214
215   Start = '**OTHER**';                               %*zero==False;      215
216   output;                                           stop; 216
217                                                    217
218 %**4.4 make FORMAT value $INVALID;                  218
219 proc FORMAT cntlin = FORMAT_CNTL                    219
220     library = WORK;%* . . . . . %IF not cexist; %END; 220
221                                                    221
222 filename TEMPTXT catalog 'sasuser.profile.sasinp.source'; 222
223                                                    223
224 %**5 read CONTENTS, write macro calls;              224
225 %local ANY_FRMT;                                     %LET ANY_FRMT=0;%*flag for exit; 225
226 %local CHARVLEN;                                     %LET CHARVLEN=0;%*max char var length; 226
227 data _NULL_;                                         227
228   file %IF &TESTING %THEN %DO; PRINT                %END; 228
229       %ELSE                                         %DO; TEMPTXT          %END; 229
230                                           %*file closure;; 230
231   retain CharVLen 1                                  231
232       Any_Frmt '0';                                  232
233   do until(EndoFile);                                233
234       set CONTENTS                                  234
235       (where=(input(put(Format,$INVALID.),1.)))      235
236       end = EndoFile;                                236
237   Any_Frmt = '1';                                    237
238   if Type = 2 then CharVLen = max(CharVLen,Length);  238
239   put 'CHK4NVLD(' '&DATA.' ',' Name ',' Type ',' Format ',' VarNum ')'; 239
240 %*..... do until(EndoFile); end; 240
241 call symput('ANY_FRMT', Any_Frmt );                 241
242 call symput('CHARVLEN',compress(put(CharVLen,3.))); stop; 242
243 %*note CHARVLEN used by CHK4NVLD; run; 243
244 %IF &TESTING %THEN %put ANY_FRMT<&ANY_FRMT.> CHARVLEN<&CHARVLEN.>; 244
245                                                    245
246 %IF not &ANY_FRMT %THEN %DO;%PUT data &DATA has no formats w/&INVALID.; 246
247                                     %GOTO ENDOMACR; %END; 247
248                                                    248
249 %**6 prepare empty data set for proc APPEND;         249
250 DATA INVALID;                                       250
251 length                                              251
252 %DO I = 1 %TO &DIM_IDS.; &&IDS&I &&TYPE&I &&LEN&I %END; 252
253   Var $ 32                                           253
254   Var_Info $ 60                                       254
255   Type $ 1 %*note: CONTENTS.Type is numeric;        255
256   ValueChr $ &CHARVLEN.                              256
257   ValueNum 8 %*WARNING different lengths on BASE and DATA; 257
258   VarNum 4                                           258
259   Label $ &LENLABEL                                  259
260   N 4 %*use: report counter;                          260
261   Format $ 8                                          261
262   ;                                                    stop; 262
263                                                    263
264 %**7;%include TEMPTXT %IF &TESTING %THEN %DO; /source2 %END; 264
265                                     %*include closure;; 265
266 filename TEMPTXT clear;                               266

```

```

267                                                                    267
268 %local DATANOBS; %NOBS(DATANOBS,data = &LIBRARY..&DATA.);          268
269 %local NVLDCELS; %NOBS(NVLDCELS,data = INVALID);run;              269
270                                                                    270
271 TITLE&TITLEN. "invalid values in &DATA. Obs:&DATANOBS.";          271
272                                                                    272
273 %**8;%IF not &NVLDCELS %THEN %DO; DATA _NULL_;                    273
274                                                                    file &PRNTFILE;          274
275                                                                    put '***no invalid values;'; 275
276                                                                    %GOTO ENDOMACR;              276
277 %**9 make values used in summary;                                    277
278 proc FREQ data = INVALID; %*do cross-tab of IDs;                  278
279     tables &IDS1                                                    279
280     %IF &DIM_IDS ge 2 %THEN                                          280
281     %DO I = 2 %TO &DIM_IDS; * &&IDS&I.                               281
282     / list noprint                                                  282
283     out = INVALID_IDS;                                              283
284     tables Var                                                       284
285     / list noprint                                                  285
286     out = INVALID_VARS;                                             286
287                                                                    287
288 %local NVLDIDS ; %NOBS(NVLDIDS ,data = INVALID_IDS );              288
289 %local NVLDVARS; %NOBS(NVLDVARS,data = INVALID_VARS);              289
290 %local DATAVARS; %NOBS(DATAVARS,data = CONTENTS);run;            290
291                                                                    291
292 %IF &DETAILS. %THEN %DO;%*-----; 292
293 %**10 write corex to PRNTFILE; options pageno=1;                  293
294 TITLE&TITLEN. "invalid values in &DATA. Obs:&DATANOBS.";          294
295                                                                    295
296 PROC SORT data = INVALID;                                           296
297     by &IDLIST.;                                                    297
298                                                                    298
299 DATA _NULL_;                                                       299
300 file &PRNTFILE;                                                     300
301 retain B -1                                                         301
302     Q ""; %*quote;                                                  302
303 do until(EndoFile);                                                 303
304     set INVALID end = EndoFile;                                     304
305     by &IDLIST;                                                      305
306     if first.&&IDS&DIM_IDS then do;                                  306
307         put "*/if &IDS1 = &Q1" &IDS1 +B "&Q1." 307
308         %IF &DIM_IDS. ge 2 %THEN                                     308
309         %DO I = 2 %TO &DIM_IDS.; " and &&IDS&I = &&Q&I." &&IDS&I +B "&&Q&I." 309
310                                                                    %END; 310
311         " then do;"; %*if first.*; end; 311
312                                                                    312
313     if Type eq '1' then put '* ' Var @13 '=' @15 ValueNum ';'; 313
314     if Type eq '2' then put '* ' Var @13 '=' @15 Q +B ValueChr +B Q ';'; 314
315                                                                    315
316     if last.&&IDS&DIM_IDS then put '*/end;'; 316
317 %*..... do until(EndoFile)*; end; 317
318 %**;%macro MAKEPCNT(VAR,NUM,DENOM);%*create % trim value to &D decimals; 318
319 %local D I TRIMLEN; %LET D = 3; %LET TRIMLEN = 0;                  319
320 %LET &VAR = %sysevalf(100 * &&NUM / &&DENOM);                      320
321 %LET I = %index(&&VAR,.);%*if dot in value trim number of decimals; 321
322 %IF &I %THEN %DO; %LET TRIMLEN = %eval(%length(&&VAR) - &I);      322
323     %IF &TRIMLEN gt &D %THEN %LET TRIMLEN = &D;                  323
324     %LET &VAR = %substr(&&VAR,1,&I + &TRIMLEN); %END; 324
325 %*PUT &VAR = &&VAR;%*.....MAKEPCNT*; %MEND; 325
326 %** print summary percentages at end of file;                      326
327 %local DATA CELLS;%LET DATA CELLS = %eval(&DATANOBS * &DATAVARS); 327

```

```

328 %local PCNTNOBS;%MAKEPCNT(PCNTNOBS,NVLDIDS ,DATANOBS); 328
329 %local PCNTVARS;%MAKEPCNT(PCNTVARS,NVLDVARS,DATAVARS); 329
330 %local PCNTCELS;%MAKEPCNT(PCNTCELS,NVLDCELS,DATACELS); 330
331 retain C1 12 C2 22 C3 32; 331
332 put "****smry : " @C1 "Ids" @C2 "Vars" @C3 "Cells;"; 332
333 put "**** Base:" @C1 "&DATANOBS" @C2"&DATAVARS" @C3 "&DATACELS;"; 333
334 put "****Invalid:" @C1 "&NVLDIDS" @C2"&NVLDVARS" @C3 "&NVLDCELS;"; 334
335 put "**** Pcnt:" @C1 "&PCNTNOBS%" @C2"&PCNTVARS%" @C3 "&PCNTCELS%"; 335
336 stop; 336
337 %*..... %IF &DETAILS *; %END; 337
338 %**11 print desired summary report(s); 338
339 %IF &SUMMARY. %THEN %DO;options pageno=1;%*-----; 339
340 %IF &SMRYIDS %THEN %DO; 340
341 proc PRINT data = INVALID_IDS; 341
342 sum Count; 342
343 TITLE%eval(&TITLEN.+1) "summary: IDs listed " 343
344 "invalid<&NVLDIDS.> / data<&DATANOBS.> = &PCNTNOBS.%"; %END; 344
345 345
346 %IF &SMRYVARS %THEN %DO; 346
347 proc PRINT data = INVALID_VARS; 347
348 sum Count; 348
349 TITLE%eval(&TITLEN.+1) "summary: Variables listed " 349
350 "invalid<&NVLDVARS.> / data<&DATAVARS.> = &PCNTVARS.%"; %END; 350
351 351
352 %IF &SMRYNAME %THEN %DO; 352
353 proc SORT data = INVALID; 353
354 by Var &IDLIST.; 354
355 proc PRINT data = INVALID label ;* (drop = Type VarNum Label); 355
356 var &IDLIST. Value;; 356
357 sum N; 357
358 by Var_Info; 358
359 id Var_Info; 359
360 label Var_Info = 'Var Format Label'; 360
361 TITLE%eval(&TITLEN.+1) "detail: by Var-Name " 361
362 "invalid<&NVLDCELS.> / data<&DATACELS.> = &PCNTCELS.%"; %END; 362
363 %**12;%IF "&PRNTLIST" eq "PRINT" 363
364 and "&PRNTFILE" ne "PRINT" %THEN %DO;%*-----; 364
365 data _NULL_; %*print detail list*; options pageno=1; 365
366 TITLE%eval(&TITLEN.+1); 366
367 file PRINT; 367
368 do until(EndoFile);%local LRECL;%LET LRECL = 72; 368
369 infile &PRNTFILE end = EndoFile pad lrecl = &LRECL.; 369
370 input @1 Line $char&LRECL.; 370
371 put @1 Line $char&LRECL.; %*do until(EndoFile)*; end; 371
372 stop; 372
373 %*..... %IF PRNTLIST eq PRINT & PRNTFILE ne PRINT*; %END; 373
374 %*..... %IF &SUMMARY *; %END; 374
375 %ENDOMACR: run;TITLE%eval(&TITLEN.);options &VALIDVARNAME.; 375
376 %IF not &TESTING %THEN %DO; 376
377 proc DATASETS library = WORK nolist; 377
378 delete CONTENTS DETAILS FORMAT_CNTL 378
379 INVALID INVALID_IDS INVALID_VARS 379
380 (memtype = data); quit; 380
381 %*..... %IF not &TESTING*; %END; 381
382 OPTIONS nomprint; run;%*.....INVALID*; %MEND; 382
383 %**13;%macro CHK4NVLD(DATA,VAR,TYPE,FORMAT,VARNUM);%* - - - - -; 383
384 %*note CHARVLEN created by calling macro INVALID; 384
385 %local HAVEDATA;%LET HAVEDATA = 0;%*flag for append; 385
386 386
387 data DETAILS(drop = HaveData 387
388 rename = (&VAR = 388

```

```

389          %IF "&TYPE" = "2" %THEN ValueChr; 389
390          %ELSE ValueNum; 390
391 length Var $ 32 391
392 Var_Info $ 60 392
393 Type $ 1 %*NOTE: CONTENTS.Type is numeric; 393
394 %IF "&TYPE" = "2" %THEN %DO; ValueNum 8 %END; 394
395 %ELSE %DO; ValueChr $ &CHARVLEN. %END; 395
396 VarNum 4 396
397 Label $ &LENLABEL 397
398 N 4 %*use: report counter; 398
399 Format $ 8; 399
400 retain Var "&VAR." 400
401 Var_Info "&VAR." 401
402 HaveData '0' 402
403 Type "&TYPE." 403
404 %IF "&TYPE"="2" %THEN %DO; ValueNum . %END; 404
405 %ELSE %DO; ValueChr '.' %END; 405
406 VarNum &VARNUM. 406
407 Label "." 407
408 N 1 408
409 Format "&FORMAT."; 409
410 do until(EndoFile); 410
411 set &LIBRARY..&DATA. 411
412 (keep = &IDLIST &VAR. 412
413 where = (put(&VAR.,&FORMAT..) eq "&INVALID.") 413
414 ) end = EndoFile; 414
415 if Label eq '.' then do; call label(&VAR.,Label); %*load only once; 415
416 Var_Info = "&VAR. &FORMAT. " !! Label; end; 416
417 HaveData='1'; %*HaveData updated when 1 obs meets where cond; 417
418 output; %*do until(EoF);end; 418
419 call symput('HAVEDATA',HaveData); stop; 419
420 run; 420
421 %IF &HAVEDATA %THEN %DO; 421
422 proc APPEND base = INVALID 422
423 data = DETAILS; %END; 423
424 run;%*.....; %MEND; 424
425 ;/*TEST DATA ***** to enable, end this line with slash (/) ** 425
426 *PROC CATALOG catalog=WORK.FORMATS kill;quit;%*zap previous; 426
427 %LET INVALID=INVALID;%*in either AUTOEXEC or LETFRMT; 427
428 proc FORMAT; 428
429 value one_3_ 1,2,3 = 'OK' other="&INVALID."; 429
430 value two_4_ 2,3 , 4 = 'OK' other="&INVALID."; 430
431 value $thre_5_ '3','4','5' = 'OK' other="&INVALID."; 431
432 value no_chk 1,2,3 , 4 , 5 = 'OK'; 432
433 value $notused '5' = 'OK' other="&INVALID."; 433
434 434
435 DATA TESTNVLD; 435
436 attrib IDChr length= $ 3 label = 'ID1Chr' 436
437 IDNmbr length= 4 label = 'ID1Num' 437
438 FormNmbr length= $ 2 label = 'ID2' 438
439 A length= 4 format=one_3_. label = 'A: number 1' 439
440 B length= 4 format=two_4_. label = 'B: number 2' 440
441 C length= $ 1 format=$thre_5_. label = 'C: char var' 441
442 I length= 4 format=no_chk. label = 'loop counter' 442
443 ; 443
444 retain FormNmbr '01'; 444
445 do I = 1 to 5; IdChr = put(I,z3.); IdNmbr = I; 445
446 A = I; B = sum(I,2); 446
447 C = put(I,1.); output; end; 447
448 stop; 448
449 %*Comparison report of formats in LIBRARY and FMTLIB; 449

```

```

450 %INVALID(,LIBRARY = WORK
451 ,FMTLIB = WORK);
452 %*EXPECTED REPORT:
453 INVALID: compare formats in DATA:WORK._ALL_ and FMTLIB:WORK
454 FORMAT DATA FMTLIB OK Comment
455 $NOTUSED 0 1 .. no problem
456 $THRE_5_ 1 1 OK
457 NO_CHK 1 0 ?? no <other=INVALID>, is problem?
458 ONE_3_ 1 1 OK
459 TWO_4_ 1 1 OK
460 NOTE: only formats with CHK=OK will be used by this routine;
461
462 %INVALID(TESTNVLD
463 ,IDLIST =IdChr FormNmbr
464 ,LIBRARY =WORK
465 ,FMTLIB =WORK
466 ,PRNTFILE="ZIRreport.SAS"
467 );
468 %*one detail line from the detail, by identifier, report:
469 %*EXPECTED OUTPUT:* if IDCHR = '00001' and FORMNMBR = '01' then do;
470 %* char ^ ^ quoted
471 %* summary written at end of detail report:
472 ***smry : Ids Vars Cell
473 *** Base: 5 7 35
474 *Invalid: 5 3 7
475 *** Pcnt: 100% 42.857% 20%
476 NOTE: re Cell Pcnt range: <1%==good, 1:2%==OK, >2%==trouble!;
477
478 %INVALID(TESTNVLD
479 ,IDLIST =IDNMBR FORMNMBR
480 ,LIBRARY =WORK
481 ,FMTLIB =WORK
482 ,PRNTFILE='C:\TEMP\ZINVALID.SAS'
483 );%*NOTE hard-coded output fileref;
484 %*EXPECTED OUTPUT:*if IDNMBR = 1 and FORMNMBR = '01' then do;
485 %* numeric ^ ^ no quotes;
486
487 %* Four Summary Reports;;
488
489 %* ,SMRYIDS =1 *?print FREQ of IDS? *
490
491 invalid values in TESTNVLD Obs:5
492 summary: IDs listed invalid<5> / data<5> = 100%
493
494 Obs IDNMBR FORMNMBR COUNT PERCENT
495
496 1 1 01 1 14.2857
497 2 2 01 1 14.2857
498 3 3 01 1 14.2857
499 4 4 01 2 28.5714
500 5 5 01 2 28.5714
501 =====
502 7;
503
504 %*,SMRYVARS=1 *?print FREQ of variables with invalid? *
505
506 invalid values in TESTNVLD Obs:5
507 summary: Variables listed invalid<3> / data<7> = 42.857%
508
509 Obs VAR COUNT PERCENT
510

```

```

511 1      A      2      28.5714      511
512 2      B      3      42.8571      512
513 3      C      2      28.5714      513
514      =====      514
515      7;      515
516      516
517 %* ,SMRYNAME=1 *?print detail of variables, by name? *      517
518 invalid values in TESTNVLD Obs:5      518
519 detail: by Var-Name invalid<7> / data<35> = 20%      519
520      520
521 Var Format Label      IDNMBR FORMNMBR VALUECHR VALUENUM N      521
522      522
523 A ONE_3_ A: number 1      4      01      .      4      1      523
524      5      01      .      5      1      524
525 -----      525
526 A ONE_3_ A: number 1      2;      526
527      527
528 run; /*****/ 528

```