

## Looking for the Missing(ness) Piece

Louise S. Hadden, Independent Consultant, Somerville, MA

### ABSTRACT

Reporting on missing and/or non-response data is of paramount importance when working with longitudinal surveillance, laboratory, and medical record data. Reshaping the data over time to produce such statistics is a tried-and-true technique, but for a quick initial look at data files for problem areas, there is an easier way. This quick tip will speed up your data cleaning reconnaissance and help you find your missing(ness) piece. Additional tips on making true missingness easy to identify are included in this paper.

### INTRODUCTION

There are myriad ways to determine if you have missing data (PROC FREQ, PROC MEANS, PROC SUMMARY, PROC UNIVARIATE, etc.). Most SAS® statistical procedures can report out on the number of missing values. Depending on procedure options in PROC FREQ, missing values can be included, or not, in counts of observations. Other statistical procedures simply drop records with missing values. Reporting on missing values can occur via “list” output, procedural output or ODS output objects. Most statistical procedures do not distinguish between distinct types of missing but PROC FREQ and reporting procedures do. This paper explores using PROC FREQ and PROC UNIVARIATE to report on missing values by variable in a data set. This presentation is suitable for all levels, industries, and job roles.

### PREPARING MISSING VALUES

It is common in survey output to assign codes such as 95 for other specify, 96 for other, 97 for refused, 98 for not answered, and 99 for not applicable. These values represent distinct types of missing values. Unfortunately, to statistical procedures, they are just numbers and are treated as such. SAS can assign up to 28 special missing value codes, ., .\_, and .A through .Z. These represent extremely small numbers that are greater than 0 – different extremely small numbers. SAS can and does distinguish between these special missing values in reporting procedures and PROC FREQ. It is recommended that analysts recode the 9x codes (and the like) to special missing values – for example, 95 (other) could be .O, and a valid skip (determined by looking at a survey instrument or data dictionary for skip patterns) could be coded .V. Knowing what is truly missing data is of the utmost importance. Note information on special missing value recodes in both variable and value labels. PROC FORMAT will report on the different missing values when used with reporting procedures and PROC FREQ. When the format below is applied to a variable, .O, .M, and .V are all reported out separately.

```
proc format;  
  value varx_f .O = '.O: Other Specify'  
             .M = '.M: Missing'  
             .V = '.V: Valid Skip'  
             other = 'Non-Missing';  
run;
```

### PROC FREQ, ODS OUTPUT OBJECTS, AND NLEVELS

Regardless of how your data was prepared with respect to missing values, SAS has a sadly underutilized variant of PROC FREQ that allows you to produce a missingness report with ease. This procedural option is NLEVELS. If your data set has special missing numeric values, these will be reported as “levels” of missing. Character variables have a single “level” of missing. The syntax for PROC FREQ NLEVELS is as follows:

```

Ods trace;

proc freq data=int.&infi. nlevels;
  ods output nlevels=nlevels0;
  tables _all_ / noprint;
run;
ods output close;

ods trace off;

proc print data=nlevels0 (obs=5) noobs;
title 'Test nlevels output';
run;

proc contents data=nlevels0 varnum;
run;

```

The only ODS output object that PROC FREQ NLEVELS produces is NLEVELS. The NLEVELS0 data set contains 5 variables: TABLEVAR (variable name), TABLEVARLABEL (variable label), NLEVELS (number of different values, including missing), NMISSLEVELS (number of different missing values), and NNONMISSLEVELS (number of different non-missing values). Using `_all_` in the table statement means that all variables in the data set are tabulated. Thus, you can generate a single line with missing value statistics for each variable in a single report.

PROC FREQ with the `_ALL_` tables option will report on variables in the order in which they entered the PDV, so if a different order is desired, prepare your data set for reporting by reordering your variables. Additionally, ensure that all variables are labelled prior to reporting.

The listing output from PROC FREQ NLEVELS is not ideal for reporting. We use a PROC REPORT step, using the ODS OUTPUT object generated by the procedure.

## PROC REPORT, ODS OUTPUT OBJECTS, AND NLEVELS

In preparation for reporting, we set the NLEVELS0 ODS output object, creating a temporary file for printing. We label the variables and create a non-printing variable that allows us to flag any variables without any non-missing value levels.

```

data nlevels;
  set nlevels0;
  label TableVar = "Variable Name"
        TableVarLabel = "Variable Description"
        NLevels = "# of Variable values"
        NMissLevels = "# of Missing Value Levels"
        NNonMissLevels = "# of Non- Missing Value Levels";
  shadeit=(nnonmisslevels=0);
run;

```

We use PROC REPORT and ODS RTF to set up our Missingness report, highlighting a high degree of missingness using the shadeit variable created above. Note that we could modify this using cardinality ratios created from the NLEVELS, NMISSLEVELS, and NNONMISSLEVELS variables to refine our reporting. SHADEIT is set as a non-printing variable in the DEFINE statement but must be in the columns statement for the conditional shading of the row to work. Note that this shading can also be applied to single cells using options in the COMPUTE statement.

```

ods rtf file="Missingingness.rtf" path=odsout style=styles.pearl;
title2 "Missingness Report for &infi - N = &nobs";
proc report nowd data=nlevels

```

```

style(report)=[cellpadding=3pt vjust=b]
style(header)=[just=center font_face="Helvetica" font_weight=bold
font_size=8pt]
style(lines)=[just=left font_face="Helvetica"] split='|';
columns TableVar TableVarLabel NLevels NmissLevels NNonMissLevels
shadeit;
define shadeit / display ' ' noprint;
define TableVar / style(COLUMN)={just=l font_face="Helvetica"
font_size=8pt cellwidth=295 }
style(HEADER)={just=l font_face="Helvetica" font_weight=bold
font_size=8pt };
define TableVarLabel / style(COLUMN)={just=l font_face="Helvetica"
font_size=8pt cellwidth=395 }
style(HEADER)={just=l font_face="Helvetica" font_weight=bold
font_size=8pt };
define Nlevels / style(COLUMN)={just=c font_face="Helvetica"
foreground=navy
font_size=8pt cellwidth=95 }
style(HEADER)={just=c font_face="Helvetica" font_weight=bold
font_size=8pt };
define NMissLevels / style(COLUMN)={just=c font_face="Helvetica"
foreground=navy
font_size=8pt cellwidth=95 }
style(HEADER)={just=c font_face="Helvetica" font_weight=bold
font_size=8pt };
define NNonMissLevels / style(COLUMN)={just=c font_face="Helvetica"
foreground=navy
font_size=8pt cellwidth=95 }
style(HEADER)={just=c font_face="Helvetica" font_weight=bold
font_size=8pt };
compute shadeit;
if (shadeit eq 1) then call
define(_row_, "STYLE", "STYLE=[BACKGROUND=PINK]");
endcomp;
run;

ods rtf close;

```

## REPORT ON VARIABLE LEVELS

We can produce a report from a data set with thousands of variables with a few lines of PROC FREQ and PROC REPORT code, instantly highlighting records for variables which may have a missingness problem. Using the SHADEIT variable to screen, we could produce a report with variables with only missing values to research.

| Variable Name       | Variable Description  | # of Variable values | # of Missing Value Levels | # of Non-Missing Value Levels |
|---------------------|---|----------------------|---------------------------|-------------------------------|
| IN_DATA_EXTRCT_DT   | Mo 4: Date of data extraction   | 1                    | 0                         | 1                             |
| INF_IDENTIFIER1     | Mo 4: Infant identifier-#1  | 1550                 | 0                         | 1550                          |
| IN_AMB_VISIT_DT     | Mo 4: Date of any ambulatory care visit, including antenatal care, ED, telemedicine.    | 1                    | 1                         | 0                             |
| IN_CHLOROQ_END_DATE | Mo 4: First administration of treatment - End Date: Chloroquine Phosphate (Chloroquine) | 1                    | 0                         | 1                             |

| Variable Name        | Variable Description  | # of Variable values | # of Missing Value Levels | # of Non-Missing Value Levels |
|----------------------|---|----------------------|---------------------------|-------------------------------|
| IN_CHLOROQ_STRT_DATE | Mo 4: First administration of treatment - Start Date: Chloroquine Phosphate (Chloroquine) | 1                    | 0                         | 1                             |

**Table 1. Missingness Report**

## BUT WAIT, THERE'S MORE!

The number of observations in a file is an important datum that can be used to calculate numerous useful statistics, for example, the cardinality ratio of each variable in a file. To obtain the N of a file, or a series of files, we use a macro to grab the NOBS. There are many ways to determine the number of observations. Since we analyze several different files, we use a macro driven solution to accommodate that need.

```
%macro filecheck(inlib=out,inmem=recover_surveillance_&delivdate.);

proc sql noprint;
  create table filelist0 as
  select libname, memname, nobs
  from dictionary.tables
  where libname = upcase("&inlib");
quit;

data filelist;
  set filelist0 (where=(memname = upcase("&inmem.")));
run;

%mend;

%filecheck(inlib=out,inmem=recover_surveillance_&delivdate.);

data _null_;
  set filelist;
  call symput('fileobs',trim(nobs));
run;
```

This macro is included in a larger program which calculates cardinality. This program has a few additions to the code snippet we saw above.

```
%include ".\INCLUDE_Nobs.sas";
run;

data nlevels out.nlevels_surveillance_&delivdate.;
  length cr_type $ 12;
  set nlevels0;
  label TableVar = "Variable Name"
  TableVarLabel = "Variable Description"
  NLevels = "# of Variable values"
  NMissLevels = "# of Missing Value Levels"
  NNonMissLevels = "# of Non- Missing Value Levels";
  shadeit=(nnonmisslevels=0);
  /* shades variables with NO non missing values */
  nobs=&fileobs;
  cardinality=nobs/nlevels;
  label nobs = "# of Observations"
```

```

cardinality = "Cardinality Ratio";
select;
    when(nlevels eq 1 ) cr_type = '.unique';
    when(nlevels gt 10) cr_type = 'many';
    otherwise cr_type = 'few';
end;
label cr_type='Cardinality Type';
run;

```

We then can produce a more robust missingness report.

```
ods rtf file="Missingness_Report_Surveillance_&delivdate..rtf" path=odsout
style=styles.pearl;
```

```

title2 "Missingness Report for Deliverable Surveillance data set for
&delivdate";
proc report nowd data=nlevels
    style(report)=[cellpadding=3pt vjust=b]
    style(header)=[just=center font_face="Helvetica" font_weight=bold
font_size=8pt]
    style(lines)=[just=left font_face="Helvetica"] split='|';
    columns TableVar TableVarLabel NLevels NmissLevels NNonMissLevels nob
cardinality shadeit;
    define shadeit / display ' ' noprint;
    define TableVar / style(COLUMN)={just=1 font_face="Helvetica"
font_size=8pt cellwidth=250 }
style(HEADER)={just=1 font_face="Helvetica"
font_weight=bold
font_size=8pt };
    define TableVarLabel / style(COLUMN)={just=1 font_face="Helvetica"
font_size=8pt cellwidth=375 }
style(HEADER)={just=1 font_face="Helvetica"
font_weight=bold
font_size=8pt };
    define Nlevels / style(COLUMN)={just=c font_face="Helvetica"
foreground=navy
font_size=8pt cellwidth=75 }
style(HEADER)={just=c font_face="Helvetica"
font_weight=bold
font_size=8pt };
    define NmissLevels / style(COLUMN)={just=c font_face="Helvetica"
foreground=navy
font_size=8pt cellwidth=75 }
style(HEADER)={just=c font_face="Helvetica"
font_weight=bold
font_size=8pt };
    define NNonMissLevels / style(COLUMN)={just=c font_face="Helvetica"
foreground=navy
font_size=8pt cellwidth=75 }
style(HEADER)={just=c font_face="Helvetica"
font_weight=bold
font_size=8pt };
    define Nobs / style(COLUMN)={just=c font_face="Helvetica" foreground=navy
font_size=8pt cellwidth=75 }
style(HEADER)={just=c font_face="Helvetica"
font_weight=bold
font_size=8pt };

```


```

define Cardinality / style(COLUMN)={just=c font_face="Helvetica"
foreground=navy
font_size=8pt cellwidth=110 }
style(HEADER)={just=c font_face="Helvetica"
font_weight=bold
font_size=8pt };
compute shadeit;
if (shadeit eq 1) then call
define(_row_, "STYLE", "STYLE=[BACKGROUND=PINK]");
endcomp;
run;

ods _all_ close;

```

What does the cardinality ratio give us? My company uses cardinality ratios to determine how to report on variables.



| Variable Name     | Variable Description  | # of Variable values | # of Missing Value Levels | # of Non-Missing Value Levels | # of Observations | Cardinality Ratio |
|-------------------|---|----------------------|---------------------------|-------------------------------|-------------------|-------------------|
| target_4_target_1 | Marshfield lab data: Lab-Swab 1: Target 4 Target: Name/type of target that the next 2 fields (Cq and call) are referencing.   | 3                    | 1                         | 2                             | 304838            | 101612.67         |
| target_4_cq_1     | Marshfield lab data: Lab-Swab 1: Target 4 Cq: 'CQ' value is the numeric data point that comes from the PCR test for this target. There is a cut-off that determines positivity vs negativity which is assay specific. | 1309                 | 1                         | 1308                          | 304838            | 232.87853         |
| target_4_call_1   | Marshfield lab data: Lab-Swab 1: Target 4 Call: The 'call' is the written interpretation of the CQ value.   | 3                    | 1                         | 2                             | 304838            | 101612.67         |
| target_5_target_1 | Marshfield lab data: Lab-Swab 1: Target 5 Target: Name/type of target that the next 2 fields (Cq and call) are referencing.   | 1                    | 1                         | 0                             | 304838            | 304838            |
| target_5_cq_1     | Marshfield lab data: Lab-Swab 1: Target 5 Cq: 'CQ' value is the numeric data point that comes from the PCR test for this target. There is a cut-off that determines positivity vs negativity which is assay specific. | 1                    | 1                         | 0                             | 304838            | 304838            |
| target_5_call_1   | Marshfield lab data: Lab-Swab 1: Target 5 Call: The 'call' is the written interpretation of the CQ value.   | 1                    | 1                         | 0                             | 304838            | 304838            |
| assay_1           | Marshfield lab data: Lab-Swab 1: This is the name of the assay that was being used at Marshfield when this test was conducted.  | 6                    | 1                         | 5                             | 304838            | 50806.333         |

### Frequency on CR\_TYPE

| Cardinality Type |           |         |                      |                    |
|------------------|-----------|---------|----------------------|--------------------|
| cr_type          | Frequency | Percent | Cumulative Frequency | Cumulative Percent |
| .unique          | 140       | 13.21   | 140                  | 13.21              |
| few              | 551       | 51.98   | 691                  | 65.19              |
| many             | 369       | 34.81   | 1060                 | 100.00             |

A cr\_type of .unique likely is identifiable data and won't be reported on. A cr\_type of few will be handled with a frequency table. A cr\_type of many will be handled with a summary / means / univariate.

## WHEN LESS IS MORE

Sometimes all that is needed is a simple missing / presence report, by site. Perhaps you have thousands of variables, with distinct value types. SAS, the missing function, proc univariate outtable, and some clever macro coding can produce exactly the report that is required. The code snippets below provide the answer with an informative table by site of missing and present variables, regardless of variable type.

```
*****;
*** create contents to drive processing ***;
*****;

PROC SQL NOPRINT;
CREATE TABLE CONTS_P AS
SELECT name format = $32. length = 32
, type format = $4. length = 4
, length format= 8. length = 8
, label format = $250. length = 250
, varnum format =8. length = 8
FROM DICTIONARY.COLUMNS
WHERE libname="INLIB" and memname=upcase("&infi.");
QUIT;

proc print data=conts_p (obs=5) noobs;
run;

proc sort data=conts_p;
    by type;
run;

*** get the maximum n of each type and create macro vars ***;

proc freq data=conts_p;
    tables type / noprint out=max_type (keep=type count);
run;

data _null_;
    set max_type;

    if type='char' then do;
        ccount=put(count,z4.);
        CALL SYMPUTX("cmax",ccount);
    end;

    if type='num' then do;
        ncount=put(count,z4.);
        CALL SYMPUTX("nmax",ncount);
    end;

run;

*** use &cmax for character and &nmax for numeric ***;

*****;
*** List variable names by type and number ***;
```

```

*****;

data nums;
  length name $ 32;
  set conts_p (where=(type='num'));
  retain nnum 0;
  by type;

  if first.type then nnum=0;
  nnum=nnum+1;

run;

data chars;
  length name $ 32;
  set conts_p (where=(type='char'));
  retain cnum 0;
  by type;

  if first.type then cnum=0;
  cnum=cnum+1;

run;

data dd.conts_p;
  length name $ 32;
  set chars nums;
  by type;
run;

. . .

proc import dbms=xlsx out = xwalk
  datafile = "&outfolder.\xwalkformat_p.xlsx" replace;
  getnames=YES;
run;

data namexwalk (keep=fmtname type start label fmtdesc);
  length start $ 8 name label $ 32;
  set xwalk (keep=name type cnum label varnum rename=(type=vtype
label=vlabel));
  fmtname = "namexwalk";
  type='c';
  fmtdesc = "Use to convert cnnn/nnnn to correct variable names";
  start = cats(vtype, 'var', put(cnum, z4.));
  label = name;
run;

proc format library=work cntlin=namexwalk fmtlib;
run;

data ordxwalk (keep=fmtname type start label fmtdesc);
  length start $ 8 ;
  set xwalk (keep=name type cnum label varnum rename=(type=vtype
label=vlabel));
  fmtname = "ordxwalk";
  type='c';

```



```

        fmtdesc = "Use to convert cnum/nnnn to obtain variable order";
        start = cats(vtype, 'var', put(cnum, z4.));
        label = varnum;
run;

proc format library=work cntlin=ordxwalk fmtlib;
run;

data labxwalk (keep=fmtname type start label fmtdesc);
    length start $ 8 vlabel label $ 250;
    set xwalk (keep=name type cnum label varnum rename=(type=vtype
label=vlabel));
    fmtname = "labxwalk";
    type='c';
    fmtdesc = "Use to convert cnum/nnnn to obtain variable label";
    start = cats(vtype, 'var', put(cnum, z4.));
    label = vlabel;
run;

proc format library=work cntlin=labxwalk fmtlib;
run;

*****;
*** create macro variables to list character and numeric variables      ***;
*****;

proc sql noprint;
    select name format = $32. length = 32 into :clist separated by ' '
    from conts_p
    where type='char';
quit;

proc sql noprint;
    select name format = $32. length = 32 into :nlist separated by ' '
    from conts_p
    where type='num';
quit;

*****;
*** use missing function to create numeric reps of char and num vars    ***;
*****;

data temp;
    set inlib.&infi;

    array clist (*) &clist;
    array cton (*) cvar0001 - cvar&cmax. ;

    array nlist (*) &nlist;
    array nton (*) nvar0001 - nvar&nmax. ;

    tempc=.;
    tempn=.;

    do i=1 to dim(clist);
        tempc=missing(clist(i));
        if tempc=0 then cton(i)=1;
    end;

```

```

        else if tempc=1 then cton(i)=.;
    end;

    do j=1 to dim(nlist);
        tempn=missing(nlist(j));
        if tempn=0 then nton(j)=1;
        else if tempn=1 then nton(j)=.;
    end;

    drop i j;
run;

proc freq data=temp;
    tables site;
run;

proc sort data=temp;
    by site study_id;
run;

*****;
*** macro to process by site using univariate outtable ***;
*****;

%macro outtable(site=1);

proc univariate data=temp (where=(site=&site))
    outtable=TempTable_Site&site (keep=_var_ _nobs_ _nmiss_
    rename=( _var_=varname _nobs_=n&site _nmiss_=nmiss&site)) noprint;
    var cvar: nvar: ;
run;

proc sort data=temptable_site&site;
    by varname;
run;

proc print data=TempTable_Site&site (obs=50) label noobs;
    var varname n&site nmiss&site;
    label varname='Variable';
title3 "Site &site";
run;

%mend;

%outtable(site=1);
%outtable(site=2);
%outtable(site=3);
%outtable(site=4);
%outtable(site=5);
%outtable(site=6);

*****;
*** combine site level runs together with two columns per site ***;
*****;

data TempTable_AllSites;
    length name $ 32 varlabel $ 250;

```

```

merge temptable_site: ;
by varname;

varname=trim(varname);
varnum=input(put(varname,$ordxwalk.),8.);
varlabel=put(varname,$labxwalk.);
name=put(varname,$namexwalk.);
format n1-n6 nmiss: comma7.0 name $32.;
/* relabel */

label name='Variable Name'
varlabel='Variable Description'
n1='# present'
n2='# present'
n3='# present'
n4='# present'
n5='# present'
n6='# present'
nmiss1='# missing'
nmiss2='# missing'
nmiss3='# missing'
nmiss4='# missing'
nmiss5='# missing'
nmiss6='# missing';

run;

proc sort data=temptable_allsites;
by varnum;
run;

proc print data=temptable_allsites (obs=25) noobs;
var varnum name varlabel n1 nmiss1 n2 nmiss2 n3 nmiss3 n4 nmiss4 n5
nmiss5 n6 nmiss6;
format varlabel $50.;
title3 "All sites";
run;

ods listing close;
title1;
title2;

ods escapechar='^';

*****;
*** Open the Excel Spreadsheet making sure to set options ***;
*** Column width to improve appearance ***;
*** Name the worksheet ***;
*** Sheet breaks set to none so tables follow each other ***;
*****;

ods excel file="&outfolder.\Person_Missing.xlsx" style=styles.excel
options(sheet_interval="none" embedded_titles="yes"
sheet_name="PersonMissingness");

```

```

proc report nowd data=temptable_allsites
  style(report)=[cellpadding=3pt vjust=b ]
  style(header)=[just=center font_face="Helvetica" font_size=8pt]
  style(lines)=[just=left font_face="Helvetica" font_size=8pt] split='|';
  columns ("^{style [just=l font_weight=bold font_size=8pt
background=ligr]Missingness by Site}" name varlabel)
    ("^{style [font_weight=bold font_size=8pt background=ligr]BSWH}"
n1 nmiss1)
    ("^{style [font_weight=bold font_size=8pt background=ligr]KPNW}"
n2 nmiss2)
    ("^{style [font_weight=bold font_size=8pt background=ligr]St.
Luke's}" n3 nmiss3)
    ("^{style [font_weight=bold font_size=8pt background=ligr]AZ
Tucson}" n4 nmiss4)
    ("^{style [font_weight=bold font_size=8pt
background=ligr]University of UT}" n5 nmiss5)
    ("^{style [font_weight=bold font_size=8pt background=ligr]Miami}"
n6 nmiss6)
  ;

define name / style(COLUMN)={just=l font_face="Helvetica"
font_size=8pt cellwidth=250 }
  style(HEADER)={just=l font_face="Helvetica"
font_size=8pt };
define varlabel / style(COLUMN)={just=l font_face="Helvetica"
font_size=8pt cellwidth=400 }
  style(HEADER)={just=l font_face="Helvetica"
font_size=8pt };
define n1 / style(COLUMN)={just=c font_face="Helvetica"
font_size=8pt cellwidth=70 }
  style(HEADER)={just=c font_face="Helvetica"
font_size=8pt };
define nmiss1 / style(COLUMN)={just=c font_face="Helvetica"
font_size=8pt cellwidth=70 }
  style(HEADER)={just=c font_face="Helvetica"
font_size=8pt };
define n2 / style(COLUMN)={just=c font_face="Helvetica"
font_size=8pt cellwidth=70 }
  style(HEADER)={just=c font_face="Helvetica"
font_size=8pt };
define nmiss2 / style(COLUMN)={just=c font_face="Helvetica"
font_size=8pt cellwidth=70 }
  style(HEADER)={just=c font_face="Helvetica"
font_size=8pt };
define n3 / style(COLUMN)={just=c font_face="Helvetica"
font_size=8pt cellwidth=70 }
  style(HEADER)={just=c font_face="Helvetica"
font_size=8pt };
define nmiss3 / style(COLUMN)={just=c font_face="Helvetica"
font_size=8pt cellwidth=70 }
  style(HEADER)={just=c font_face="Helvetica"
font_size=8pt };
define n4 / style(COLUMN)={just=c font_face="Helvetica"
font_size=8pt cellwidth=70 }
  style(HEADER)={just=c font_face="Helvetica"
font_size=8pt };
define nmiss4 / style(COLUMN)={just=c font_face="Helvetica"

```

```

font_size=8pt cellwidth=70 }
style(HEADER)={just=c font_face="Helvetica"
font_size=8pt };
define n5 / style(COLUMN)={just=c font_face="Helvetica"
font_size=8pt cellwidth=70 }
style(HEADER)={just=c font_face="Helvetica"
font_size=8pt };
define nmiss5 / style(COLUMN)={just=c font_face="Helvetica"
font_size=8pt cellwidth=70 }
style(HEADER)={just=c font_face="Helvetica"
font_size=8pt };
define n6 / style(COLUMN)={just=c font_face="Helvetica"
font_size=8pt cellwidth=70 }
style(HEADER)={just=c font_face="Helvetica"
font_size=8pt };
define nmiss6 / style(COLUMN)={just=c font_face="Helvetica"
font_size=8pt cellwidth=70 }
style(HEADER)={just=c font_face="Helvetica"
font_size=8pt };

run;

ods excel close;

ods listing;

```

| Missingness by Site |  | Site 1    |           | Site 2    |           | Site 3    |           | Site 4    |           | Site 5    |           | Site 6    |        |
|---------------------|--|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|--------|
| Missingness by Site | Variable Description   | # present | # missing | # present | # missing | # present | # missing | # present | # missing | # present | # missing | # present | # miss |
| study_id            | Admin: Study ID  | 609       | 0         | 711       | 0         | 907       | 0         | 254       | 0         | 970       | 0         | 890       | 0      |
| site                | Admin: Study site  | 609       | 0         | 711       | 0         | 907       | 0         | 254       | 0         | 970       | 0         | 890       | 0      |
| study               | Admin: HR study  | 609       | 0         | 711       | 0         | 907       | 0         | 254       | 0         | 970       | 0         | 890       | 0      |
| der_study_part      | [derived variable] Study participation status based on consent and S1 blood draw                       | 609       | 0         | 711       | 0         | 907       | 0         | 254       | 0         | 970       | 0         | 890       | 0      |
| enroll_status       | [derived variable] Study participation status based on consent and swab collection/active surveillance | 609       | 0         | 711       | 0         | 907       | 0         | 254       | 0         | 970       | 0         | 890       | 0      |
| surv_st_week        | [derived variable] MMWR year and week of the first surveillance record                                 | 520       | 89        | 658       | 53        | 872       | 35        | 249       | 5         | 898       | 72        | 832       | 56     |
| swab_st_week        | [derived variable] MMWR year and week of the first lab record  | 511       | 98        | 655       | 56        | 859       | 48        | 245       | 9         | 892       | 78        | 824       | 66     |
| der_surv_st_week    | [derived variable] First of either swab date or completed any survey in study after enrollment         | 521       | 88        | 658       | 53        | 872       | 35        | 249       | 5         | 901       | 69        | 864       | 26     |
| der_n_surv          | [derived variable] Number of MMWR weeks with surveillance in the data                                  | 609       | 0         | 711       | 0         | 907       | 0         | 254       | 0         | 970       | 0         | 890       | 0      |

## CONCLUSION

PROC FREQ with the NLEVELS option can provide an excellent broad stroke report on missingness in your data sets. PROC REPORT can generate a traffic-lighted report based on the number of total levels on each individual variable in the data set. Macro coding, the missing function, and proc univariate outtables can produce a simple but effective table of presence and absence for an unlimited number of variables without user intervention. We have found the Missing(ness) Piece!

## REFERENCES

- Boniface, Christopher J. and Wysocki, Janet L. April 2016. "You Can Bet On It, The Missing Rows are Preserved with PRELOADFMT and COMPLETETYPES". *Proceedings of the SAS Global 2016 Conference*, Las Vegas, NV: SAS Institute.
- Bost, Christopher. April 2011. "To FREQ, Perchance to MEANS". *Proceedings of the SAS Global 2011 Conference*, Las Vegas, NV: SAS Institute.
- Fehd, Ronald J. April 2013. "Data Review Information: N-Levels or Cardinality Ratio". *Proceedings of the SAS Global 2013 Conference*, San Francisco, CA: SAS Institute.

Fehd, Ronald J. September 2022. "Calculating Cardinality Ratio in Two Steps". *Proceedings of the WUSS 2022 Conference, San Francisco, CA*. <https://www.lexjansen.com/wuss/2022/WUSS-2022-Paper-69.pdf>

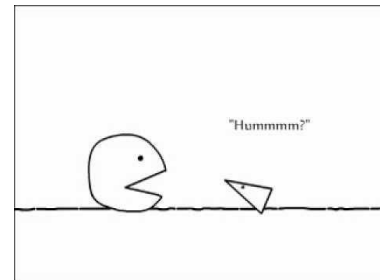
Jia, Justin and Lin, Amanda. April 2016. "Missing Values, They are NOT Nothing". *Proceedings of the SAS Global 2016 Conference, Las Vegas, NV*: SAS Institute.

Ramezani, Niloofar. April 2020. "Analyzing Non-normal Data: Application to Missing Data Problems". *Proceedings of the SAS Global 2020 Conference, Virtual*: SAS Institute.

Shan, Xia Ke and Bremser, Kurt. June 2020. "Five Simple Ways to Know If Variables in a Table Are All Missing". *Proceedings of the SAS Global 2020 Conference, Virtual*: SAS Institute.

Stutzman, Paul. June 2017. "Check Your Data: Tools for Automating Data Assessment". *Proceedings of the PharmaSUG 2017 Conference, Baltimore, MD*: PharmaSUG.

Zdeb, Mike. October 2016. "An Easy Route to a Missing Data Report with ODS+PROC FREQ+A Data Step". *Proceedings of the 2016 Southeast SAS Users Group Conference, North Bethesda, MD*: SESUG.



## CONTACT INFORMATION

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

Louise S. Hadden  
Independent Consultant  
[Saslouisehadden@gmail.com](mailto:Saslouisehadden@gmail.com)  
[girlwiththesastattoo@gmail.com](mailto:girlwiththesastattoo@gmail.com)

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