

A Macro to Account for Missing Data in a Table

Margie Bell, Stephen Black, and Jennifer Fulton
Applied Logic Associates, Inc.
Houston, TX

Abstract

The purpose of this macro is to provide supplementary information to a table, which can explain why the table does not contain all of the data that the user expects to observe. The macro takes a user-specified data set that contains the variable to be tabulated and combines it with another user-specified data set that contains a record for EVERY item, which could potentially have data in the table. It then creates a set of counts to be printed beneath the table, which account for discrepancies between the total counts in the table and the counts that would appear if there were no missing data issues. The categories of counts include "No Record in Table Data", "Missing Value in Table Data", "Number in Table", "True Total", and a listing identifying those individual items that have missing data related to the table. The user also has the option to print another column that would explain why certain data is missing in the table when the explanation does not fall into the "No Record" or "Missing Value" categories. The output created by this macro is to be supplementary to an existing table and is not intended to stand alone. This macro was written using the SAS® System for Windows version 6.12 on a Windows '95 platform, and uses Base SAS®. The user should have at least a moderate familiarity with SAS® macro language.

Introduction

Although we have attempted to write this macro in such a way as to make it usable and applicable for any industry, its creation was motivated by efforts to present a complete picture in tables presented to the FDA (Food and Drug Administration) containing clinical trials data. Most, if not all, companies in the pharmaceutical industry who have made presentations to FDA have had to explain why all of the patients known to be enrolled in a study are not represented in a particular table of patient data. There may be multiple acceptable reasons that some patients are excluded from a table. For example, the table may represent vital signs recorded at the follow-up visit of a study, and not all patients who participated in the study returned for this follow-up visit. Or a patient may show up for a particular visit, but his clinical measurement is below the level of detection for the tool being used, so he has other values for that visit but this measurement is missing. Such discrepancies are usually explained in detail in a clinical report that accompanies the tables created in SAS®, but a succinct summary of such issues located on the same page as the table can quickly and efficiently answer questions of the FDA reviewers and provide a comfort level that the presentation accounts for all of the data.

Since the accompanying clinical report will explain in depth the reasons that certain patients appear in some tables but not others, we have written our macro to simply categorize the table data as follows:

“No Record in Table Data” – This is a count of patients (or whatever unit is being tabulated) who do not have a record in the data set that creates this table.

“Missing Value in Table Data” – This is a count of patients (or whatever unit is being tabulated) who have a record in the data set that creates this table, but have a missing value for the particular variable being tabulated.

“*User-Specified Label*” (Optional) – This column allows the user to provide another reason that patients (or whatever unit is being tabulated) do not appear in the table besides having a missing record or a missing value (i.e. if the table intentionally includes only a subset of the total). For example, this column might include a count of all of the male patients if the table includes only female patients.

“Number in Table” – This count should match the count in the table. It is included so it can be easily compared to the other counts and the user can see easily that the values in the above categories should sum to the “True Total”.

“True Total” – This is the count of all patients (or whatever unit is being tabulated) who could have appeared in the table. The counts in all other categories should sum to this value.

“IDs With Missing Data” – This is a listing of the patient ids (or whatever unit is being tabulated) who are not represented in the table because of a missing record or a missing value in the table data.

Since these categories make little sense without an accompanying table, the output from our macro is written only to be supplementary to a table and is not intended to stand alone.

Example

Below is an example of a data set (VITALS.SD2) that might be seen in a typical clinical trial. Each patient is represented by a number (PATID), and the vital signs which include pulse (PULSE), temperature in °F (TEMP), systolic blood pressure (SYBP), and diastolic blood pressure (DIBP) have been measured at several different times within the study (VISIT). TREAT is the treatment that each patient received during the study and is used as the row variable in PROC TABULATE for our example.

PATID	TREAT	VISIT	PULSE	TEMP	SYBP	DIBP
0001	DRUG	DAY1	.	97.2	.	.
0001	DRUG	DAY2	88	96.9	130	80
0001	DRUG	DAY3	90	97.5	.	.
0002	CONTROL	DAY1	.	95.0	.	.
0002	CONTROL	DAY2	.	95.1	.	.
0002	CONTROL	DAY3	86	96.4	120	80
0004	DRUG	DAY1	72	98.0	160	90
0004	DRUG	DAY2	92	97.1	130	84
0004	DRUG	DAY3	90	97.2	.	.

The data set we will compare to is TRTMT.SD2, which contains one record for every patient in the study. It looks like:

PATID	TREAT
0001	DRUG
0002	CONTROL
0003	CONTROL
0004	DRUG
0005	DRUG

Perhaps we would like to make one table for each visit showing the summary statistics for pulse for each treatment. These tables might look like:

TABLE 1.1
DAY 1

	PULSE				
	N	MEAN	STD DEV	MINIMUM	MAXIMUM
TREAT	0				
CONTROL					
DRUG	1	72.00		72.00	72.00

TABLE 1.2
DAY 2

	PULSE				
	N	MEAN	STD DEV	MINIMUM	MAXIMUM
TREAT	0				
CONTROL					
DRUG	2	90.00	2.83	88.00	92.00

TABLE 1.3
DAY 3

	PULSE				
	N	MEAN	STD DEV	MINIMUM	MAXIMUM
TREAT	1	86.00		86.00	86.00
CONTROL					
DRUG	2	90.00	0.00	90.00	90.00

An FDA reviewer who knows that a total of 5 patients were enrolled in this study would like to know why the value of N is less than 5 in each case. We know from looking at the data sets that patients 0003 and 0005 appear to have never had their vital signs measured during days 1 to 3, while patients 0001 and 0002 came in at each appointed visit, but pulse was not recorded for them at all three visits. After the code that produces each table, the SAS programmer could include the following macro call to produce a succinct summary of the vital signs data beneath each table:

```
%DATAcnt1(_BYVAR= BY TREAT, _ID=PATID,
            INDSN1=VITALS, INDSNALL=TRTMT, VAR=PULSE);
```

Below is what the resulting output would look like:

Table 1.1
DAY 1

	PULSE				
	N	MEAN	STD DEV	MINIMUM	MAXIMUM
TREAT	0				
CONTROL					
DRUG	1	72.00		72.00	72.00

DATA ACCOUNTING

TREAT MISSING DATA	NO RECORD IN TABLE DATA	MISSING VALUE IN TABLE DATA	NUMBER IN TABLE	TRUE TOTAL	IDs WITH
CONTROL	1	1	0	2	0002, 0003
DRUG	1	1	1	3	0001, 0005

Table 1.2
DAY 2

	PULSE				
	N	MEAN	STD DEV	MINIMUM	MAXIMUM
TREAT	0				
CONTROL					
DRUG	2	90.00	2.83	88.00	92.00

DATA ACCOUNTING

TREAT MISSING DATA	NO RECORD IN TABLE DATA	MISSING VALUE IN TABLE DATA	NUMBER IN TABLE	TRUE TOTAL	IDs WITH
CONTROL	1	1	0	2	0002, 0003
DRUG	1	0	2	3	0005

Table 1.3
DAY 3

	PULSE				
	N	MEAN	STD DEV	MINIMUM	MAXIMUM
TREAT	1				
CONTROL		86.00		86.00	86.00
DRUG	2	90.00	0.00	90.00	90.00

DATA ACCOUNTING

TREAT MISSING DATA	NO RECORD IN TABLE DATA	MISSING VALUE IN TABLE DATA	NUMBER IN TABLE	TRUE TOTAL	IDs WITH
CONTROL	1	0	1	2	0003
DRUG	1	0	2	3	0005

In each of the above tables, all 5 patients enrolled in the study are accounted for, whether or not they had pulse data to tabulate.

Conclusion

The example we presented included simple data sets and simple accompanying tables in order to clearly illustrate the intended use of our macro. However, the macro may be very useful for more complex presentations of many different types of data. The primary benefit of using this macro is that it enables the presenter to tell the complete story on a single page. In industries different from the pharmaceutical industry, the tables produced to summarize pertinent data may not normally be accompanied by a detailed report, which is an even stronger reason to make sure that those tables provide the most complete picture of the data possible. We also sometimes use this macro as a validation tool to make sure that the program that creates the table properly represents all of the intended data, and that any data that is not in the table is excluded for a good reason. More information presented in a clean, succinct manner can only improve your presentations, be they to an FDA review board or any other audience you want to impress with a minimum number of questions.

Note: All data is fabricated and are for demonstration purposes only.

SAS is a registered trademark of SAS Institute Inc. in the USA and other countries.

® indicates USA registration

Please address correspondence to:

Applied Logic Associates
Stephen Black
Manager, Biostatistical Programming
5615 Kirby Dr., Suite 710
Houston, TX 77005-2495
E-MAIL: sblack@alogic.com

Program Code

```
/******  
* Client:      All Clients of Applied Logic Associates, Inc.  
* Protocol:    All Protocols  
* Program:     DATACNT1.SAS  
* Input Files: User-specified datasets with one containing the a SAMPLE  
*               of the variable to be counted and another containing  
*               ALL data for the variable to be tabulated (used to count  
*               missing values)  
* External Macros: None  
* Output Files: Tables to LST File of Calling Program  
* Tables:      None  
* Programmer:  SBLACK  
* Date Completed: 31JUL95  
* *****/
```

```

* Description:      This Program takes a user-specified dataset that contains
*                  a SAMPLE of the variable to be tabulated (such as patient
*                  blood pressure at visit 1 of a study) and combines it with
*                  another user-specified dataset that contains ALL of the
*                  data for the variable to be tabulated (such as blood
*                  pressure for all patients at all visits in the study). It
*                  then creates a list of the counts to be printed beneath the
*                  table including: Missing Data for whole Table,
*                  Missing Data for This Item (an individual segment of the
*                  table), Number in Table, TRUE TOTAL (the true count of the
*                  variable being tabulated), as well as a listing of all id's
*                  with missing data related to the table. It also includes an
*                  optional count of values within a user-specifiable range of
*                  values.
*                  NOTE: This output is to be supplementary to an existing
*                  table and is not intended to stand alone
*                  (Title= Data Accounting').

```

DESCRIPTION OF MACRO PARAMETERS SENT FROM YOUR PROGRAM:

```

* _BYVAR : By Variable(s) across which the id counts are calculated
*          (for example, if each row in your table of patient blood
*          pressure is a different study visit, then _BYVAR should be your
*          visit variable, so that the list of counts created by this macro
*          will be separated by visit as well)
*
* _ID : Variable containing the item being counted in the table
*       (for example, _ID = SUBNO for patient id's).
*
* INDSN1 : User-specified dataset containing SAMPLE POPULATION of the
*          variable to be tabulated. (DEFAULT VALUE: _LAST_)
*
* INDSNALL: User-specified dataset containing ALL data for the variable
*           being tabulated (including all values of the _BYVAR variable(s))
*
* _NUMMIS : Maximum number of ids with missing values to be printed out by
*           this macro. The ids are listed in lines of 10 per line.
*           This value must be greater than 0. (DEFAULT VALUE: 70)
*
* _VAR : Variable containing the characteristic of the id's that are
*        being counted (such as patient blood pressure).
*
* _XSUBLB : (OPTIONAL) User-specified label to appear in the Table above the
*           additional _XSUBVAR count information
*
* _XSUBVAR: (OPTIONAL) User-specified variable that may help explain the
*           differences between the counts in the Table Data and counts
*           presented by this macro (if data in Table Data was subset there
*           may be discrepancies)
*
* _XSUBVHI: (OPTIONAL) User-specified value of the _XSUBVAR variable which
*           will be used as the upper bound on the range of _XSUBVAR values
*           included in the additional count (count includes upper bound
*           value). This range should be exclusive of the table values
*           (an id cannot appear in both counts)
*
* _XSUBVLO: (OPTIONAL) User-specified value of the _XSUBVAR variable which
*           will be used as the lower bound on the range of _XSUBVAR values
*           included in the additional count (count includes lower bound
*           value). This range should be exclusive of the table values
*           (an id cannot appear in both counts)

```

RESTRICTIONS:

- 1) Length of _ID variable values cannot exceed 10 characters
- 2) values of Range for additional count must be exclusive of the range of values included in the table (an id cannot appear in both places)
- 3) Any dataset subsetting that is to be performed for a subset analysis must be executed prior to the calling of this macro
- 4) _NUMMIS value must be greater than zero.

```

*****
%MACRO DATACNT1(_BYVAR=,
               _ID=,
               INDSN1=_LAST_,
               INDSNALL=,
               _NUMMIS=70,
               _VAR=,
               _XSUBVAR=,
               _XSUBVLO=,
               _XSUBVHI=,
               _XSUBLB=);

```

```

/*-----*
* Create variable _BYVARKP to
* strip off the word BY for use
* in DROP/KEEP statements.
*-----*/
%LET _BYVARKP = %SUBSTR(&_BYVAR, 4);

```

```

/*-----*
* Read in dataset which contains all data on variable
* being tabulated, including all values of the _BYVAR variables*
*-----*/
DATA _ALL1 (KEEP = _PATNO &_BYVARKP _ALLCNT);
  LENGTH _PATNO $10;
  SET &INDSNALL ;

  _PATNO = TRIM(LEFT(%UPCASE(&_ID)));
  _ALLCNT = 1;

RUN;

PROC SORT DATA = _ALL1 ;
  &_BYVAR _PATNO ;

```

```

/*-----*
* Keep only one observation per id
*-----*/
DATA _ALL;
  SET _ALL1;
  &_BYVAR _PATNO;
  IF FIRST._PATNO;
RUN;

/*-----*
* Count total number of ids for each _BYVAR
*-----*/
PROC MEANS N DATA=_ALL NOPRINT;
  VAR _ALLCNT;
  &_BYVAR;
  OUTPUT OUT=_ALLCNT(DROP=_FREQ_ _TYPE_) N=_TOTCNT;
RUN;

/** Initialize Flag Variable **/
%LET _MISSSUB = 0;
RUN;

/*-----*
* Read in dataset which contains _VAR values
* Including values of the _BYVAR variables
*-----*/
DATA _MDAT (KEEP = _PATNO &_BYVARKP _MTYPE &_VAR);
  LENGTH _PATNO $10 _XLABEL $40;
  SET &INDSN1;

  _PATNO = TRIM(LEFT(%UPCASE(&_ID)));
  _XLABEL = '';

  IF UPCASE(TRIM(LEFT(&_VAR))) EQ '.' OR
     UPCASE(TRIM(LEFT(&_VAR))) EQ '' THEN _MTYPE = 2; /* Count as missing value **/
  ELSE IF UPCASE(TRIM(LEFT(&_VAR))) NE '.' AND
          UPCASE(TRIM(LEFT(&_VAR))) NE '' THEN _MTYPE = 3; /* Count as table data **/

  %IF %UPCASE(&_XSUBVLO) ^= AND %UPCASE(&_XSUBVHI) ^= %THEN %DO;
    /* Count as IN Range for Additional Count and OUTSIDE TABLE RANGE **/
    IF UPCASE("&_XSUBVLO") <= &_XSUBVAR <= UPCASE("&_XSUBVHI")
       THEN _MTYPE = 4;

    /* Count as MISSING _XSUBVAR values **/
    ELSE IF UPCASE(TRIM(LEFT(&_XSUBVAR))) EQ '.' OR
          UPCASE(TRIM(LEFT(&_XSUBVAR))) EQ '' THEN DO;
      _MTYPE = 5;

      /* Use variable label from dataset **/
      CALL LABEL(&_XSUBVAR, _XLABEL);
      CALL SYMPUT('EXMISLAB', TRIM(LEFT(_XLABEL)));
      CALL SYMPUT('_MISSSUB', '1');
    END;
  %END;

RUN;

PROC SORT DATA = _MDAT;
  &_BYVAR _PATNO;
RUN;

/*-----*
* Merge data with ALL ids with the dataset
* of values in order to perform the counts
*-----*/
DATA _ALLPATA _ERRDAT;
  MERGE _MDAT (IN=DATA) _ALL (IN=ALL);
  &_BYVAR _PATNO;

  _PATCNT = 1;
  IF ALL AND ^DATA THEN _MTYPE = 1; /* Count as missing id **/

  IF DATA AND ^ALL THEN OUTPUT _ERRDAT;
  ELSE OUTPUT _ALLPATA;
RUN;

PROC SORT DATA = _ALLPATA;
  &_BYVAR _PATNO;
RUN;

OPTIONS FORMDLIM='';

PROC PRINT DATA=_ERRDAT;
  TITLE1 "Ids with Table Data (in &INDSN1) and";
  TITLE2 "NO ALL Data (NOT IN &INDSNALL)";
  TITLE4 "This is a Problem and Should be Corrected";
RUN;

/*-----*
* Keep only one observation per id
*-----*/
DATA _ALLPATB;
  SET _ALLPATA;
  &_BYVAR _PATNO;

  IF FIRST._PATNO;
  TITLE;
RUN;

PROC SORT DATA = _ALLPATB;
  &_BYVAR _MTYPE;
RUN;

```

```

/*-----*
* Count the number of ids in each _MTYPE group
* and save the values for output in the table
*-----*/
PROC MEANS N DATA = _ALLPATB NOPRINT;
  &_BYVAR _MTYPE;
  VAR _PATCNT;
  OUTPUT OUT = _PATCNTA(DROP=_FREQ_ _TYPE_) N = _PATCNT;
RUN;

PROC TRANSPOSE DATA=_PATCNTA OUT=_PATCNTB PREFIX=_COUNT;
  VAR _PATCNT;
  &_BYVAR;
  ID _MTYPE;
RUN;

/*-----*
* Combine total id count with other counts
*-----*/
PROC SORT DATA=_ALLCNT;
  &_BYVAR;
RUN;

PROC SORT DATA=_PATCNTB;
  &_BYVAR;
RUN;

DATA _BOTHCNT;
  MERGE _ALLCNT _PATCNTB;
  &_BYVAR;
RUN;

%LET _DATFLAG = 0;

/*-----*
* Keep ids who have missing values in order to
* display the list of ids with missing data
*-----*/
DATA _ALLPATC (KEEP = _PATNO &_BYVARKP);
  SET _ALLPATB;
  WHERE _MTYPE = 1 OR _MTYPE = 2;
  IF _N_ > 0 THEN CALL SYMPUT('_DATFLAG', '1');
RUN;

/** If data exists for missing counts **/
%IF &_DATFLAG = 1 %THEN %DO;

  PROC SORT DATA=_ALLPATC;
    &_BYVAR _PATNO;
  RUN;

  /*-----*
  * Transpose id number data to display the list of
  * ids with missing data (if fewer than &_NUMMIS)
  *-----*/
  PROC TRANSPOSE DATA=_ALLPATC OUT=_PATNOS PREFIX=_PATN;
    VAR _PATNO;
    &_BYVAR;
  RUN;

  /*-----*
  * Merge id counts with the id numbers to
  * be displayed as missing
  *-----*/
  DATA _ALLDAT1;
    MERGE _BOTHCNT _PATNOS;
    &_BYVAR;
    _TVAR = 1;
    IF _COUNT1 = . THEN _COUNT1 = 0;
    IF _COUNT2 = . THEN _COUNT2 = 0;
    IF _COUNT3 = . THEN _COUNT3 = 0;
    IF _COUNT4 = . THEN _COUNT4 = 0;
    IF _COUNT5 = . THEN _COUNT5 = 0;
  RUN;

  %END;

/** If data Does NOT exist for missing counts **/
%ELSE %DO;

  DATA _ALLDAT1;
    SET _BOTHCNT;
    &_BYVAR;
    _TVAR = 1;
    IF _COUNT1 = . THEN _COUNT1 = 0;
    IF _COUNT2 = . THEN _COUNT2 = 0;
    IF _COUNT3 = . THEN _COUNT3 = 0;
    IF _COUNT4 = . THEN _COUNT4 = 0;
    IF _COUNT5 = . THEN _COUNT5 = 0;
  RUN;

  %END;

PROC SORT DATA=_ALLDAT1;
  &_BYVAR _TVAR;
RUN;

/*-----*
* Combine IDs of missing data into several large
* strings in order to be printed nicely in the table
*-----*/
DATA _ALLDAT2 (KEEP = &_BYVARKP _TOTCNT _COUNT1 - _COUNT5 _MISPATS);
  LENGTH _MISPATS $120;
  ARRAY _PATNS (&_NUMMIS) $10 _PATN1 - _PATN&_NUMMIS;
  ARRAY _COUNT (5) _COUNT1 - _COUNT5;
  SET _ALLDAT1;
  &_BYVAR _TVAR;

```

```

    /*** Initialize variables ****/
    IF FIRST._TVAR THEN DO;
        _TOTMIS = .;
    END;

    /*** Determine the total number of missing ids ***
    *** (_COUNT1 and _COUNT2 are for missing ids ****/
    _TOTMIS = _COUNT1 + _COUNT2;

    /*** If the number of missing ids is zero then ***
    *** do not print anything for missing IDs ****/
    IF _TOTMIS = 0 THEN DO;
        _MISPATS = '';
        OUTPUT;
    END;

    /*** If the number of missing ids is greater than *****
    *** &_NUMMIS, then print Too Many IDs to List Message ***/
    ELSE IF _TOTMIS > &_NUMMIS THEN DO;
        _MISPATS = "TOO MANY MISSING TO LIST id IDs";
        OUTPUT;
    END;

    /*** If the number of missing ids is >0 and <=&_NUMMIS then ***
    *** print out a list of IDs with missing values *****/
    ELSE DO;
        _TEMPMIS = _TOTMIS;
        _TEMPCNT = 0;

        /*** Separate missing ids into lines of 10 ids each ***/
        DO _I = 1 TO _TOTMIS BY 10;
            _MISPATS = '';
            IF _TEMPMIS <= 10 THEN _XX = _TEMPMIS;
            ELSE DO;
                _XX = 10;
                _TEMPMIS = _TEMPMIS - 10;
            END;

            DO _J = _TEMPCNT TO (_TEMPCNT+_XX-1);
                _TEMPCNT = _TEMPCNT + 1;
                IF _MISPATS = '' AND _TEMPCNT < _TOTMIS THEN
                    _MISPATS = TRIM(LEFT(_PATNS(_TEMPCNT)))||",";
                ELSE IF _MISPATS = '' THEN
                    _MISPATS = TRIM(LEFT(_PATNS(_TEMPCNT)));
                ELSE IF _TEMPCNT = _TOTMIS THEN
                    _MISPATS = TRIM(LEFT(_MISPATS))||" "||TRIM(LEFT(_PATNS(_TEMPCNT)));
                ELSE _MISPATS = TRIM(LEFT(_MISPATS))||" "||TRIM(LEFT(_PATNS(_TEMPCNT)))||",";
            END;
            OUTPUT;
        END;
    END;

    RUN;

    /*-----*
    * Print output data according to the existence of *
    * the additional range id count *
    *-----*/
    %IF %UPCASE(&_XSUBVLO)= AND %UPCASE(&_XSUBVHI)= %THEN %DO;

        /*** Extra Subset Value HAS NO Missing Values ***/
        %IF &_MISSSUB = 0 %THEN %DO;
            PROC SORT DATA=_ALLDAT2;
                &_BYVAR _COUNT1 _COUNT2 _COUNT3 _COUNT4 _TOTCNT;
            RUN;

            PROC PRINT DATA=_ALLDAT2 NOOBS LABEL;
                VAR _MISPATS;
                LABEL _COUNT1 = "NO RECORD IN/TABLE DATA"
                    _COUNT2 = "MISSING VALUE IN/TABLE DATA"
                    _COUNT3 = "NUMBER IN TABLE"
                    _COUNT4 = &_XSUBLB
                    _TOTCNT = "TRUE TOTAL"
                    _MISPATS = "IDS WITH MISSING DATA";
                &_BYVAR _COUNT1 _COUNT2 _COUNT3 _COUNT4 _TOTCNT;
                ID &_BYVARKP _COUNT1 _COUNT2 _COUNT3 _COUNT4 _TOTCNT;
                TITLE "DATA ACCOUNTING";
            RUN;
        %END;

        /*** Extra Subset Value HAS Missing Values ***/
        %ELSE %DO;
            PROC SORT DATA=_ALLDAT2;
                &_BYVAR _COUNT1 _COUNT2 _COUNT3 _COUNT4 _COUNT5 _TOTCNT;
            RUN;

            PROC PRINT DATA=_ALLDAT2 NOOBS LABEL;
                VAR _MISPATS;
                LABEL _COUNT1 = "NO RECORD IN/TABLE DATA"
                    _COUNT2 = "MISSING VALUE IN/TABLE DATA"
                    _COUNT3 = "NUMBER IN TABLE"
                    _COUNT4 = &_XSUBLB
                    _COUNT5 = "MISSING &_XMSLAB VALUES"
                    _TOTCNT = "TRUE TOTAL"
                    _MISPATS = "IDS WITH MISSING DATA";
                &_BYVAR _COUNT1 _COUNT2 _COUNT3 _COUNT4 _COUNT5 _TOTCNT;
                ID &_BYVARKP _COUNT1 _COUNT2 _COUNT3 _COUNT4 _COUNT5 _TOTCNT;
                TITLE "DATA ACCOUNTING";
            RUN;
        %END;
    %END;

```

```

%ELSE %DO;
  PROC SORT DATA=_ALLDAT2;
    &_BYVAR _COUNT1 _COUNT2 _COUNT3 _TOTCNT;
  RUN;

  PROC PRINT DATA=_ALLDAT2 NOOBS LABEL SPLIT = '/';
    VAR _MISPATS;
    LABEL _COUNT1 = "NO RECORD IN/TABLE DATA"
          _COUNT2 = "MISSING VALUE IN/TABLE DATA"
          _COUNT3 = "NUMBER IN TABLE"
          _TOTCNT = "TRUE TOTAL"
          _MISPATS = "IDS WITH MISSING DATA";
    &_BYVAR _COUNT1 _COUNT2 _COUNT3 _TOTCNT;
    ID &_BYVARPK _COUNT1 _COUNT2 _COUNT3 _TOTCNT;
    TITLE "DATA ACCOUNTING";
  RUN;
%END;

TITLE;

*-----*;
%MEND DATACNT1;

```