

Computing in Statistics

Lecture 22
Nov. 10, 2008

Kate Cowles
374 SH
kcowles@stat.uiowa.edu

3

– example: if an adverse event of type X occurs in one dataset, you expect an observation with the same ID number in another data set. In addition, the date of this observation must be after the adverse event and before the end of the study.

from *Cody's Data Cleaning Techniques Using SAS Software* by Ron Cody, SAS Institute, 1999.

Data checking and validation (“cleaning”)

- making sure that raw data were accurately entered into a computer-readable file
- checking that character variables contain only valid values
- checking that numeric values are within pre-determined ranges
- checking whether there are missing values for variables where complete data are necessary
- checking for and eliminating duplicate records
- checking for uniqueness of certain values, such as patient IDs
- checking for invalid date values
- checking that an ID number is present in each of several related files
- verifying that more complex multi-file rules have been followed

4

Example dataset: Patients.dat

```
001M11/11/1998 88140 80 10
002F11/13/1998 84120 78 X0
003X10/21/1998 68190100 31
004F01/01/1999101200120 5A
XX5M05/07/1998 68120 80 10
006 06/15/1999 72102 68 61
007M08/32/1998 88148102 0
    M11/11/1998 90190100 0
008F08/08/1998210 70
009M09/25/1999 86240180 41
010f10/19/1999 40120 10
011M13/13/1998 68300 20 41
012M10/12/98 60122 74 0
013208/23/1999 74108 64 1
014M02/02/1999 22130 90 1
002F11/13/1998 84120 78 X0
003M11/12/1999 58112 74 0
015F 82148 88 31
017F04/05/1999208 84 20
019M06/07/1999 58118 70 0
123M15/12/1999 60 10
321F 900400200 51
020F99/99/9999 10 20 8 0
022M10/10/1999 48114 82 21
023f12/31/1998 22 34 78 0
024F11/09/199876 120 80 10
025M01/01/1999 74102 68 51
027FN0TAVAIL NA 166106 70
028F03/28/1998 66150 90 30
029M05/15/1998 41
006F07/07/1999 82148 84 10
```

Files on course web page

- data file: `patients.dat`
- SAS program: `patients172.sas`

SAS Code to read in the data

```

-----*
| PROGRAM NAME: PATIENTS.SAS IN C:\CLEANING          |
| PURPOSE: TO CREATE A SAS DATA SET CALLED PATIENTS |
| DATE: MAY 29, 1998                                |
-----*

OPTIONS FORMCHAR = "|----+|----+|-\<>*" LINESIZE = 75  NODATE;

* LIBNAME CLEAN "C:\CLEANING";

*DATA CLEAN.PATIENTS;
DATA PATIENTS;
  *INFILE "C:\temp\patients.dat" PAD;
  INFILE "/group/ftp/pub/kcowles/datasets/patients.dat" PAD;
  INPUT @1  PATNO   $3.
        @4  GENDER  $1.
        @5  VISIT   MMDDYY10.
        @15 HR      3.
        @18 SBP     3.
        @21 DBP     3.
        @24 DX      $3.
        @27 AE      $1.;

  LABEL PATNO   = "Patient Number"
        GENDER  = "Gender"
        VISIT   = "Visit Date"
        HR      = "Heart Rate"
        SBP     = "Systolic Blood Pressure"
        DBP     = "Diastolic Blood Pressure"
        DX      = "Diagnosis Code"
        AE      = "Adverse Event?";

```

```

FORMAT VISIT MMDDYY10.;

RUN;

```

New aspects of this data step

- PAD option on `infile` statement
 - adds blanks to the end of short records to the default logical record length or a length specified by another `infile` option, `lrecl`
 - prevents skipping to the next record of data when a shorter line is encountered
- @ in `input` statement
 - tell SAS at which numeric column to begin reading each variable
 - needed when there are no **delimiters** between variable values in data file
- formats after each variable name
 - how many digits or characters in each value
 - identify character variables with \$
 - MMDDYY10. is built-in SAS format for reading dates

– must end with period

Proc contents: Getting SAS to describe the contents of a dataset

```

/*****
Extra code: getting a description of the dataset
*****/
PROC CONTENTS DATA = PATIENTS ;
RUN ;
    
```

The SAS System

The CONTENTS Procedure

```

Data Set Name: WORK.PATIENTS      Observations:      3
Member Type:   DATA              Variables:          8
Engine:        V8                 Indexes:           0
Created:       8:43 Friday, June 6, 2003  Observation Length: 41
Last Modified: 8:43 Friday, June 6, 2003  Deleted Observations: 0
Protection:                               Compressed:        N
Data Set Type:                               Sorted:            N
Label:
    
```

-----Engine/Host Dependent Information-----

```

Data Set Page Size:      8192
Number of Data Set Pages: 1
First Data Page:        1
Max Obs per Page:       203
Obs in First Data Page: 31
Number of Data Set Repairs: 0
File Name:               /usr/tmp/SAS_workEB5E00003805_
                        mouse/patients.sas7bdat
    
```

```

Release Created:      8.0202M0
Host Created:         HP-UX
Inode Number:         44658
Access Permission:    rw-----
Owner Name:           UNKNOWN
File Size (bytes):    16384
    
```

-----Alphabetic List of Variables and Attributes-----

#	Variable	Type	Len	Pos	Format	Label
8	AE	Char	1	39		Adverse Event?
6	DBP	Num	8	24		Diastolic Blood Pressure
7	DX	Char	3	36		Diagnosis Code
2	GENDER	Char	1	35		Gender
4	HR	Num	8	8		Heart Rate
1	PATNO	Char	3	32		Patient Number
5	SBP	Num	8	16		Systolic Blood Pressure
3	VISIT	Num	8	0	MMDDYY10.	Visit Date

Validity checks on character variables

Variable	Valid values
Gender	F, M
DX	numerals 1 through 999
AE	0,1

- Are there other (invalid) values for these variables in the dataset
- Are there missing values?
- Which observations in the dataset contain invalid or missing values?

Using proc freq to list all distinct values of a character variable that appear in the dataset

```

/*****
Program 1-2 Using PROC FREQ to list all the unique values for character
Variablesaa`M
*****/
PROC FREQ DATA=CLEAN.PATIENTS;
  TITLE "Frequency Counts for Selected Character Variables";
  TABLES GENDER DX AE / NOCUM NOPERCENT;
RUN;

```

Frequency Counts for Selected Character Variables

```

The FREQ Procedure

      Gender

GENDER   Frequency
-----
      2             1
      F            12
      M            14
      X             1
      f             2

Frequency Missing = 1

```

Diagnosis Code

```

DX      Frequency
-----
      1             7
      2             2
      3             3
      4             3
      5             3
      6             1
      7             2
      X             2

```

Frequency Missing = 8

Adverse Event?

```

AE      Frequency
-----
      0            19
      1            10
      A             1

```

Frequency Missing = 1

Using proc print to list invalid character values and identify the observations

- **where** statement in many procedures will exclude observations that don't meet a given logical condition
- simple logical conditions involve comparing the value in a variable to some specified value
- example: **where hr > 150**
- example: **where gender not in ('M' 'F' , ')**

```

/*****
Program 1-4 Using PROC PRINT to list invalid character values`M
*****/
PROC PRINT DATA=CLEAN.PATIENTS;
  TITLE "LISTING OF INVALID GENDER VALUES";
  WHERE GENDER NOT IN ('M' 'F' , ' ');
  ID PATNO;
  VAR GENDER;
RUN;

```

LISTING OF INVALID GENDER VALUES

```

PATNO   GENDER
-----
      003      X
      010      f
      013      2
      023      f

```

The Verify Function

- `VERIFY(character.variable, verify.string)`
- returns the first position in the `character.variable` that is not in the `verify.string`
- returns 0 if `character.variable` does not contain any invalid values

```

/*****
Program 1-5 Using PROC PRINT to list invalid character data for several
Variables
*****/
PROC PRINT DATA=CLEAN.PATIENTS;
  TITLE "LISTING OF INVALID CHARACTER VALUES";
  WHERE GENDER NOT IN ('M' 'F' ' ') OR
        VERIFY(DX, '0123456789') NE 0 OR
        AE NOT IN ('0' '1' ' ');
  ID PATNO;
  VAR GENDER DX AE;
RUN;

```

LISTING OF INVALID CHARACTER VALUES

PATNO	GENDER	DX	AE
002	F	X	0
003	X	3	1
004	F	5	A
010	f	1	0
013	2	1	
002	F	X	0
023	f		0

Formatting and documenting SAS code

- Make your SAS programs readable and understandable
 - to yourself (now and if you go back to the program 5 years from now!)
 - to your instructor or supervisor
 - to someone who takes over your job when you quit
- Use meaningful variable names
 - **gender** rather than **x1**
- Use white space
 - Skip lines between logical blocks of code
 - Indent for understandability

- Document code
 - Program headings
 - * program name and where it is on computer
 - * programmer's name
 - * date of last update
 - * purpose of program
 - * input files if any
 - * output files if any
 - Comments describing what blocks of code do (if it is not obvious)