Basic Concepts #4
Data Step #3: Reading a SAS Data Set and Functions and Call Routines

JC Wang

SET Statement

SET SAS-data-name<(data-set-options)>;

- Selected data-set-options:
  - KEEP=/DROP= to read only wanted variables
  - WHERE=(expression) to read only wanted observations
  - RENAME=(old-var-name=new-name ...), separating old-var-name=new-name's by spaces, to rename variables
  - FIRSTOBS=/OBS= options to limit number of observations to read

- further statements to manipulate data
SAS Functions

\[ \text{function-name}(\text{argument-1}, \ldots, \text{argument-n}) ; \]

where arguments can be

- **variables**, for example, \( \text{mean(a,b,c)}, \text{sum(OF x1-x3,OF score[*]},y,z) \), \( \text{median(OF _NUMERIC_)} \)
- **constants**, for example, \( \text{sum(32,20,51)} \)
- **expressions**, for example, \( \text{sum(\text{mean(OF quiz[*])}, \text{mean(OF midterm[*])}, final, 2)} \)

Return values of functions are used in assignments. Use \( \text{function-name() \rangle} \) even if it does not require an argument.

SAS Call Routines

\[ \text{CALL routine-name}(\text{argument-1}, \ldots, \text{argument-n}) ; \]

where arguments can be

- **variables**, for example, \( \text{CATS(res,a,b,c)}, \text{CATS(res,OF s1-s3,OF x[*]},y,z) \)
- **constants**, for example, \( \text{CATS(res," a","b","c ")} \)
- **expressions**, for example, \( \text{CATS(res,\text{SUBSTR(a,4,7)}, \text{COMPBL(b)})} \)
- **cannot be used in assignment statement, they are used to change values of variables**

Each function and or routine has its unique usage.
**INPUT Function**

\[ \text{INPUT}(\text{source}, \langle ? | ?? \rangle \text{informat.}) \]

- \textit{source} is a SAS expression including variable and constant that is evaluated to a character string
- \textit{informat.} gives the informat to read the source,
- function value type depends on informat type, numeric or character
- optional modifier ? or ?? suppress error message to be given in SAS log, value of the automatic variable \_ERROR\_ is set to 1 for ? and set to 0 for ??

**PUT Function**

\[ \text{PUT}(\text{source}, \text{format.}) \]

- \textit{source} is a SAS expression including variable and constant
- \textit{format.} depends on type (numeric or character) of the value of \textit{source}
- the function always returns a character value
A Note About Automatic Type Conversion

Automatic type conversion occurs when

- the *expression* gives value of opposite type to existing *variable* in an assignment statement
- the operand/operands is/are of opposite type that the operator requires
- the argument of a function or of a call routine is of opposite type

**Warning:**

1. Automatic type conversion can produce unexpected outcomes or error
2. WHERE statement does not perform automatic type conversion

Some Character Functions

- `SCAN(string , n<, delimiter(s)>)` scans *string* containing words separated by delimiters (default in ASCII environment are blank . < ( + & $ * ); ^ - / , % |) and returns *n*th word; scanning words from the right end of the string if *n* is negative; returning blank if *string* has less than |*n*| words; returned word is padded by blanks with total length of 200.

- `<variable=>SUBSTR(string, position<, length>)` gives length-*length* substring from the source *string* starting from its (*position*)th position; SAS returns the remainder after *position* if *length* is omitted, *length* is non-positive, or *length* is greater than that of remainder (error message given for latter two); the returned value is padded with trailing blanks to match the length of the source *string*. 
TRIM(string) copies string, removes trailing blanks and returns the trimmed string; returns a blank if string is blank (0 or more blanks).

COMPRESS(<source>, chars, modifiers) removes characters, specified in chars list (given as a string), from the source string; modifiers, listed in string, modifies the action that COMPRESS takes (eg. k, t, i); COMPRESS() gives zero-length character string.

CATX and CATS functions

IFC and CHOOSEC functions (IFN and CHOOSEEN for numeric version)

INPUTC and PUTC functions (INPUTN and PUTN for numeric version)

COALESCEC function (COALESCE for numeric function)

TRIM, LEFT, STRIP and COMPBL functions

LOWCASE, UPCASE, and PROPCASE functions

MISSING function

See textbook for other character functions
Some Date/Time Functions

- Constructor functions: examples are
  \( \text{MDY}(\text{mon}, \text{day}, \text{year}) \), \text{TODAY}(), \text{DATE}(), \text{TIME}(), \text{DATETIME}(). \text{See also INTNX function.}.

- Extractor functions: examples are \( \text{fun(date)} \) where \text{fun} is one of \text{YEAR, QTR, MONTH, DAY, WEEKDAY}.

- Elapsed time period functions: \text{INTCK, DATDIF, YRDIF}.

Some Mathematical Functions

- \text{INT, ROUND, CEIL, and FLOOR}.
- \text{MOD, ABS, EXP, LOG, LOG10, LOG2}.
Some Statistical Functions

- Descriptive statistical functions: MIN, MAX, MEAN, MEDIAN, STD, STDERR, USS, CSS, SKEWNESS, KURTOSIS, ORDINAL, SMALLEST, ... Note that many of these functions require at least one non-missing argument to return non-missing value.

- Functions related to statistical distributions: PDF, CDF (and PROBBETA, PROBBNML, etc.), QUANTILE (and BETAINV, CINV, FINV, etc.), RAND (and RANBIN, RANNOR, RANPOI, etc.)

- Use CALL STREAMINIT(seed) to set random seed for random number generator functions so that the result is reproducible

See example RandomData.sas

NMISS and CMISS Functions

- NMISS(argument<,...,argument-n>) counts number of missing values in the numeric argument(s).

- For version 9.2 or later,
  CMISS(argument<,...,argument-n>) counts number of missing values in the argument(s). Arguments could be of different types.