SAS® Cheat Sheet

SAS Language

**ATTRIB** var, <LENGTH=var-length> <LABEL=var-label> <FORMAT=var-format> <INFORMAT=var-informat>:
Associates a format, informat, label, and/or length with one or more variables.

**DATA** <dsset, <(dsset-options)>>:
Begins a DATA step and provides names for any output SAS data sets. See Data Set Options for options that are available in the DATA statement.

**DO** index-var=start_value TO end_value <BY step>;
**DO WHILE** (expression);
**DO UNTIL** (expression);

**FILE** filename <options>;
Specifies the current output file for PUT statements. Options include:
- **MOD** output is appended to an existing file.
- **OLD** output overwrites an existing file.

**IF** expression **THEN** statement; .... **ELSE** statement;
SAS evaluates the expression in an IF statement to produce a result that is either non-zero, zero, or missing. If result >0 then TRUE, else FALSE.

**INFILE** filename <options>;
Specifies an external file to read with an INPUT statement. Options include:
- **DELMITERS=delimiters**
  Specifies a delimiter for list input.
- **LENGTH=variable**
  Names a variable that SAS sets to the length of the current input line.

**INPUT** var<= $< startcol <endcol>() <dec> <@ | @@>;
**INPUT** <pointer-control> variable informat. <@ | @@>;
**INPUT** <pointer-control> variable <$< startcol <endcol>() <dec> <@ | @@>;

**MERGE** ds1 <(doptions)><...<dsn>(doptions)><END=var>;
Joins observations from two or more SAS data sets into single observations.

**OUTPUT** <data-set-name(s)>;
**PUT** var<= $< startcol <endcol>() <dec> <@ | @@>;
**PUT** <pointer-control> <text>[variable format.]<@ | @@>;

**RETAIN** variable, <initial-value>;
Causes a variable to retain its value from one iteration of the data step to the next.

**SET** <data-set(s)> <(data-set-options)>;
**POINT=varname** <NOBS=varname> <END=varname>;
Reads observations from one or more data sets.

**SUM** variable+expression
adds the result of an expression to an accumulator var.

**TITLE** <=text>

**WHERE** expression;
Selects observations from SAS data sets that meet a particular condition that is true.

SAS Data Set Options

**DROP=variable(s)** Excludes variables from processing.

**FIRSTOBS=n** Specifies the first observation to process

**KEEP=variable(s)** Selects variables for processing.

**LABEL=’label’** Specifies a label for a SAS data set

**OBS=n** Specifies the first n observations to process

**POINT=variable** Direct observation number variable

**RENAMe=(oldname1=newname1 <...oldnamen=newnamen>**
Changes the name of a variable.

**WHERE=expression, <logical-operator-expression>**;
Selects observations from a SAS data set that meet certain conditions before SAS brings them into the DATA or PROC step for processing.

SAS Functions

**BYTE(n)** Returns one character in the ASCII or EBCDIC collating sequence where n is an integer representing a specific ASCII or EBCDIC character

**COMPB(u)l(source)** Removes multiple blanks from a character string

**COMPRESS(source<characters-to-remove>)** Removes specific characters from a character string

**DATE()** Returns the current date as a SAS date value

**DATEPART(datetime)** Extracts the date from a SAS date value

**DATEVALUE()** Returns the current date and time of day

**DAY(date)** Returns the day of the month from a SAS date value

**HMS(hour,minute,second)** Returns a SAS time value from hour, minute, and second

**INDEX(source,exception)** Searches the source for the first occurrence of a character string specified by the exception

**LEFT(argument)** Left-aligns a SAS character string

**LENGTH(argument)** Returns the length of an argument

**LOWCASE(argument)** Converts all letters in an argument to lowercase

**MAX(argument,argument, ...)** Returns the largest value of the numeric arguments

**MDY(month,day,year)** Returns a SAS date value from month, day, and year

**MIN(argument,argument, ...)** Returns the smallest value of the numeric arguments

**MISSING(argument)** Indicates whether the argument contains a missing value

**MOD(argument1, argument2)** Returns the remainder

**MONTH(date)** Returns the month from a SAS date value

**RANK(x)** Returns the position of a character x in the ASCII or EBCDIC collating sequence

**REPEAT(‘character-expression’;**
Repeats a character expression n+1 times.

**RIGHT(argument)** Right-aligns a character expression

**ROUND(argument,round-off-unit)** Rounds to the nearest round-off unit

**SCAN(argument,n<,delimiters>**
Returns a given word from an ASCII or EBCDIC collating sequence where n is an integer representing a specific ASCII or EBCDIC character

**SEARCH(source,exception)** Searches the source for the first occurrence of a character string specified by the exception

**SHIFT(‘source’;**
Shifts a character argument one position to the left or right

**SPLIT(source,delimiter)** Splits a character string into substrings

**SUBSTR(source,position<n+1>)** Extracts a substring from an argument

**TIME()** Returns the current time of day

**TIMEPART(datetime)** Extracts a time value from a SAS date value

**TODAY()** Returns the current date as a SAS date value

**TRANSFORM(source, to,from)** Replaces specific characters in a character expression

**TRIM(argument)** Removes leading and trailing blanks

**UPCASE()** Converts all letters in an argument to uppercase

**WEEKDAY(date)** Returns the day of the week from a date value

**YEAR(date)** Returns the year from a SAS date value

SAS Formats

**w.d** standard numeric

**COMMAw.d** writes numeric values with commas and decimal points

**Zw.d** prints leading zeros

**$w** writes standard character data

**$CHARW** writes standard character data (including leading blanks)

**$VARYINGw** writes character data of varying length

SAS Informs

**w.d** reads standard numeric data

**DATETIME** reads date values (ddmmmyy)

**$w** reads standard character data

**$VARYINGw** reads character data of varying length

Compliments of:

David Franklin
New Hampshire, USA
Tel/Fax +1(603) 262-9160 Cell +1(603) 275-6809
Email 100316.3461@compuserve.com
http://ourworld.compuserve.com/homepages/dfranklinuk

Release 1.2b ©MMVII

Email 100316.3451@compuserve.com
http://ourworld.compuserve.com/homepages/dfranklinuk

Release 1.2b ©MMVII
SAS® Cheat Sheet

SAS Procedures
PROC COMPARE <BASE=dset> <COMPARE=dset>; 
 BY variable(s); 
 ID variable(s); 
 VAR variable(s);
PROC DATASETS <LIBRARY=libref> <MEMTYPE=(m-list)> 
 <DETAILS>NODETAILS <KILL>; 
 <NOLIST>; 
 APPEND BASE=dset <DATA=dset> <FORCE>; 
 CHANGE old-name= new-name= <MEMTYPE=(m-list)>; 
 CONTENTS <DATA=libref.member> <DIRECTORY>; 
 <MEMTYPE=(m-list)> <NODS>; 
 <VARNUM> <NOPRINT> <OUT=dset>; 
 COPY OUT=libref <IN=libref> <MEMTYPE=(m-list)> 
 <MOVE>; 
 EXCLUDE member-list <MEMTYPE=mtype>; 
 SELECT member-list <MEMTYPE=mtype>; 
 DELETE member-list <MEMTYPE=mtype>; 
 MODIFY member-name <LABEL='data-set-label'> > 
 <SORTEDBY='sort-information'>>; 
 FORMAT variable= name.; 
 INDEX CREATE variable </<UNIQUE> <NOMISS>>; 
 INDEX CREATE index=variable-list </<UNIQUE> 
 <NOMISS>>; 
 INDEX DELETE index-list; 
 LABEL variable= label-text; 
 RENAME variable= new-variable; ;
QUIT;

where
m-list one or more of the member types that 
 processing should be restricted to. 
 member-list list of members in the directory to 
 process. 
 mtype restricts processing to one member type. 
PROC EXPORT DATA=libref.dset 
 OUTFILE='filename' <REPLACE>; 
PROC IMPORT DATAFILE='filename' 
 OUT=libref.dset <REPLACE>; 
 The following filetypes are the most commonly used and 
 supported within filename by SAS: 
 filename.XLS (Microsoft Excel) 
 filename.TXT (tab delimited) 
 filename.CSV (comma separated value) 
PROC FORMAT <CNTLIN=dset> 
 <CNTLOUT=dset> 
 <LIBRARY=libref> <CATALOG>; 
 INVALUE <$>name <value-range-set(s)>; 
 PICTURE name=..value-range-set <picture-option(s)>; 
 VALUE <$>name <value-range-set>; 
 where
picture-options The following options are useful: 
 ROUND NOEDIT 
 PREFIX= FILL= 
PROC FREQ <DATA=dset> 
 <ORDER=DATA|EXTERNAL|FREQ|INTERNAL>; 
 BY= DESCENDING variable; 
 TABLES requests <tables-options>; 
 where
requests one or more variable names joined by 
 asterisks that specify the form of the 
 generated tables, e.g. A*B 
 tables-options Can be one or more of the following: 
 LIST MISSING 
 NOPRINT OUT=SAS-data-set 
 OUTPT= SPARSE 
PROC MEANS <DATA=dset> <DESCENDING> 
 <MISSING> <NOPRINT> <NWAY> 
 <ORDER=DATA|EXTERNAL|FREQ|INTERNAL>; 
 VAR variable(s); 
 class variable(s); 
 OUTPUT <OUT=sas-data-set> <OUTSTATISTIC>; 
 where
statistic-list Can be one or more of the following: 
 N NMISS MIN MAX RANGE 
 MEDIAN SUM MEAN VAR STD 
 Q1 Q3 T 
 out-statistic Specifies the statistics in the output and 
 also names the variable(s) that contain the 
 results. 
PROC REPORT <DATA=dset> <HEADLINE> <HEADSKIP> 
 <NOWINDOWS> <SPACING=number> 
 <DEFINE report-item, <report-item>(); 
 ('header', '<', '<', header, '> report-item(s)); 
 DEFINE report-item / <usage> <defines-options>; 
 COMPUTE <BEFORE> <AFTER report-item; 
 LINE <item item-format | 'text' | pointer-control>; 
 ENDCOMP; 
 BREAK BEFORE <AFTER break-variable <b-option(s)>; 
 QUIT;
where
report-item name or alias (established in the 
 COLUMN statement) of the data set or 
 computed variable, or statistic to define 
 usage Either ACROSS, ANALYSIS, 
 COMPUTED, DISPLAY, GROUP, 
 ORDER define-options The following options are available: 
 FORMAT=format ORDER= 
 SPACING= WIDTH= 
 DESCENDING FLOW 
 NOPRINT CENTER 
 LEFT RIGHT 
 COLOR='column-header' 
 b-options These include: 
 SKIP PAGE 
PROC SORT <DATA=dset> <OUT=dset> 
 <NO DUPKEY> <NODUPS>; 
 BY= DESCENDING variable-list; 
PROC TRANSPOSE <DATA=dset> <OUT=dset>; 
 BY= DESCENDING variable-list; 
 ID variable; 
 VAR variable; … variable; 

Macro Language
%DO macro-var=start_value %TO end_value %BY step; 
 Executes a section of a macro repetitively based on the 
 value of an index variable 
%DO %WHILE (expression); 
 Executes a section of a macro repetitively while a condition 
 is true 
%DO %UNTIL (expression); 
 Executes a section of a macro repetitively until a condition 
 is true 
%GLOBAL macro-variable(s); 
 Creates macro variables that are available during the 
 execution of an entire SAS session 
%IF expression %THEN action; <%ELSE action;> 
 Conditionally process a portion of a macro 
%LENGTH (character string | text expression) 
 Returns the length of a string 
%LET macro-variable =value>; 
 Creates a macro variable and assigns it a value 
%MACRO m-name (<pp><..<kp=v>)<kp1=value<..<kpn=v)>; 
 Begins a macro definition 
%MEND <macro-name>; 
 Ends a macro definition 
%SCAN (argument, <delimiters>)* 
 Search for a word that is specified by its position in a string 
%SUBSTR (argument, position, <length>) 
 Produce a substring of a character string 
%UPCASE (character string | text expression) 
 Convert values to uppercase 

Macro Quoting
%QUOTE | %NRQUOTE and %BQUOTE | %NRBQUOTE 
 Mask special characters and mnemonic operators in a 
 resolved value at macro execution 
%STR | %NRSTR 
 Mask special characters and mnemonic operators in 
 constant text at macro compilation 
%SUPERQ 
 Masks special characters/ mnemonics operators on macro 
 execution but prevents further resolution of the value. 

Compliments of: 
David Franklin 
New Hampshire, USA 
Tel/Fax +1(603) 262-9160 Cell +1(603) 275-6809 
Email 100316.3451@compuserve.com 
Google “Franklin SAS” 
http://ourworld.compuserve.com/homepages/franklininux 
Release 1.2b 6MMVII