Macro List

(shown in alphabetical order)

Shortcut to utility macros

List of Clinical macros

```
Jul 30  2007 allocr.sas
Jul 30  2007 allocw.sas
May  8 10:47 dosemerge.sas
May  8 10:41 locf.sas
Mar 26  2008 nodata.sas
May 19 15:01 npctpvals.sas
Aug 19 08:51 npcttab.sas
May 26 17:36 popfmt.sas
May 19 14:54 rgpp.sas
May 12 18:10 rgpp_old.sas
May 19 14:56 unicat2word.sas
Sep 17 12:07 unicatrep.sas
Aug 28 22:50 unimap.sas
May 26 16:46 unipvals.sas
Aug 28 22:33 unistatlabel.sas
Sep 23 17:29 unistats.sas
```

Clinical macro purposes

```
Index of members in this directory with standard headers
========================================================
(this list was generated by the crindex script)

allocr.sas          - Spectre (Clinical) example macro to allocate data libraries and
                      formats in read mode.

                      Usage: %allocr


allocw.sas          - Spectre (Clinical) example macro to allocate data libraries and
                      formats in write mode.

                      Usage: %allocw


dosemerge.sas       - Clinical reporting macro to merge dose in with date

                      Usage:


locf.sas            - Clinical reporting macro to perform "Last Observation Carried
                      Forward" processing.

                      Usage:


nodata.sas          - To produce a "No Data" report
```

```
                         Usage: %if not %nobs(dset) %then %do;
                         %nodata
                         %goto skip;
                         %end;

npctpvals.sas          - Clinical reporting macro that calculates p-values for the
                         %npcttab macro.

                         Usage: %npctpvals(dsin=data1,byvars=byvar1 byvar2,trtvar=trtgrp,
                         respvar=resp,countvar=count,pvalstr=TRT9999)


npcttab.sas            - Clinical reporting macro to produce tables showing "n", the
                         percentage and optionally the number of events.

                         Usage: See tutorial with demonstrations on the Spectre web site


popfmt.sas             - Clinical reporting macro to create a treatment format that is the
                         same as an existing format but with the (N=xxx) at the end.

                         Usage: %popfmt(stat.acct(where=(xxx=1 and &_pop_.cd=1)),trtgroup)


rgpp.sas               - Create html graphical patient profiles

                         Usage: %rgpp

unicat2word.sas        - Clinical reporting macro to produce a Word-style cell table
                         from the dataset output from the %unistats macro of treatment-
                         transposed categories counts and statistics.

                         Usage: %unicat2word(dsin=_unitran,dest=print,dlim=';')


unicatrep.sas          - Clinical reporting macro to produce a report from the dataset
                         output from the %unistats macro of treatment-transposed
                         categories counts and statistics.

                         Usage: %unicatrep(dsin=_unitran)


unimap.sas             - Function-style clinical reporting macro to map proc univariate
                         labels to the actual stats keyword names.

                         Usage: %let stats=%unimap(&labels);

unipvals.sas           - Clinical reporting macro to calculate statistics values and
                         p-values for the %unistats macro.

                         Usage:
%unipvals(dsin=means,dsout=out,trtvar=tmt,respvar=val,type=N)


unistatlabel.sas       - To replace _statlabel values in the %unistats output dataset

                         Usage: N/A

unistats.sas           - Clinical reporting macro to calculate proc univariate
                         statistics and category counts with percentages with optional
```

```
                     statistics added and by default to print a report.

                     Usage: %unistats(dsin=means,dsout=out,dspout=pout,trtvar=tmt,
                     varlist=val,stats=n mean min max std,statvarlist=val);
```

List of System macros

```
May  8 12:01 allocr.sas
May  8 12:02 allocw.sas
Oct 12  2009 autoexec.sas
May  8 12:26 closerep.sas
May  8 12:12 crprotds.sas
May  8 12:14 crtitlesds.sas
May  8 12:15 ctitlepgmrk.sas
May  8 12:16 jobinfo.sas
May  8 12:21 layout2lsps.sas
May  8 12:25 openrep.sas
May  8 12:30 pagexofy.sas
May  8 12:36 proginfo.sas
May  8 12:42 protinfo.sas
May  8 12:43 titlegen.sas
May  8 12:46 titles.sas
May  8 12:50 xytitles.sas
```

System macro purposes

```
Index of members in this directory with standard headers
========================================================
(this list was generated by the crindex script)

allocr.sas          - Spectre (Clinical) example macro to allocate data libraries and
                      formats in read mode.

                      Usage: %allocr


allocw.sas          - Spectre (Clinical) example macro to allocate data libraries and
                      formats in write mode.

                      Usage: %allocw


closerep.sas        - Spectre (Clinical) macro to close the temporary file created
                      by the %openrep macro for redirected sas output and copy to a
                      final output file with page number labels added.

                      Usage: Should be used with the %titles and %openrep macros as
below.

                      %allocr
                      %titles
                      %openrep
                      <reporting code>
                      %closerep


crprotds.sas        - Spectre (Clinical) macro to create a protocol dataset from a
                      protocol details flat file.
```

```
                     Usage: %crprotds(flatfile,der.study)


crtitlesds.sas     - Spectre (Clinical) macro to create a titles dataset from a
                     titles flat file.

                     Usage: %crtitlesds(flatfile,der.titles)


ctitlepgmrk.sas    - Spectre (Clinical) macro to create a centered top title with a
                     right-most "FF"x page mark.

                     Usage: %ctitlepgmrk("centred title")


jobinfo.sas        - Spectre (Clinical) macro to store important job information in
                     global macro variables.

                     Usage: %jobinfo


layout2lsps.sas    - Spectre (Clinical) macro to calculate sas linesize and pagesize
                     values based on paper type, margins and layout.

                     Usage:
%layout2lsps(lmargin=1.0,rmargin=0.75,tmargin=1.0,bmargin=1.0,
                     paper=A4,layout=L10);

openrep.sas        - Spectre (Clinical) macro to redirect print output to a
                     temporary file.

                     Usage: Should be used with the %titles and %closerep macros as
below.

                     %allocr
                     %titles
                     %openrep
                     <reporting code>
                     %closerep


pagexofy.sas       - Spectre (Clinical) macro to add "Page x of Y" labels where
                     the 'FF'x character is found and to make other special
                     character substitutions.

                     Usage: %pagexofy(myfile.lst)
                     %pagexofy(myfile.lst,style="Page x of Y")
                     %pagexofy(myfile.lst,style="Seite x von Y")
                     %pagexofy(myfile.lst,style="(PAGE X OF Y)")
                     %pagexofy(myfile.lst,style="SEITE x")
                     %pagexofy(myfile.lst,style="[SEITE x]"

proginfo.sas       - Spectre (Clinical) macro to store important program information
                     in global macro variables.

                     Usage: %proginfo


protinfo.sas       - Spectre (Clinical) macro to store important protocol information
                     in global macro variables.
                     Usage: %protinfo


titlegen.sas       - Spectre (Clinical) macro to generate titles and footnotes from a
                     dataset of the style created by the %crtitlesds macro.

                     Usage: %titlegen(dsname)


titles.sas         - Spectre (Clinical) macro to create the titles and footnotes for
                     a standard report.

                     Usage: Should be used with the %openrep and %closerep macros as
below.

                     %allocr
                     %titles
                     %openrep
                     <reporting code>
                     %closerep


xytitles.sas       - Spectre (Clinical) macro to finish creating the header lines
                     for the imaginary XenuYama pharmaceutical company style.

                     Usage: Must be called from within the %titles macro and must not
be
                     used standalone.
```

List of Utility macros

```
May  4 22:30 addautos.sas
May  4 22:35 adddecodevars.sas
May  4 22:36 after.sas
May  4 22:37 age.sas
May  4 22:39 agedec.sas
May  4 22:06 aligndp.sas
Apr 13 20:23 allfmtvals.sas
May  4 20:43 alluniq.sas
May  4 22:11 attrc.sas
May  4 22:08 attrn.sas
May  4 22:10 attrv.sas
May  4 19:41 bydrop.sas
May  4 18:09 bytitle.sas
May  4 19:49 capmac.sas
May  4 19:51 capvar.sas
May  4 19:53 casestrmac.sas
May  4 19:54 casestrvar.sas
May  4 19:37 char2num.sas
May 11 18:12 checkv6.sas
May  4 18:12 chkuniq.sas
May  4 19:55 chompw.sas
Jun 30 22:25 clashlibs.sas
Jun 30 17:05 clashvars.sas
May  4 19:35 clength.sas
May  4 19:56 commas.sas
May  8 13:20 complibs.sas
May  4 20:45 compress.sas
May  4 22:56 compvars.sas
May  4 20:46 crdte.sas
Sep 28  2008 datanulldemo.sas
May  4 19:32 delhex.sas
Apr  5 22:27 delifexist.sas
May  4 19:27 delzero.sas
May  4 22:59 dequote.sas
Jun 26 13:57 dir.sas
Jun 26 14:08 dirfpq.sas
Jul  2 10:59 dlm2sas.sas
Jun  9 17:41 doallitem.sas
May  4 22:52 dosfilesize.sas
May  4 19:24 dropvars.sas
May  4 18:13 dsall.sas
May  8 13:25 dsattrib.sas
May  4 20:47 dslabel.sas
May  4 18:15 dslist.sas
May  4 20:48 dtscale.sas
May  4 20:49 duplvars.sas
May  4 19:58 endwith.sas
Jul 25 20:59 env2ds.sas
Jul 25 21:00 env2dsw7.sas
May  4 19:59 eqsuff.sas
May  4 21:34 equals.sas
May  4 22:58 fixnames.sas
May  8 13:27 fixvars.sas
May  4 18:22 flatten.sas
May  4 18:26 fmtord.sas
Apr 13 20:25 fmtpath.sas
May  4 21:40 fmts2fda.sas
May 13 15:40 getfmts.sas
Sep 19 19:23 gettitles.sas
May  5 17:43 getvalue.sas
May  4 21:41 globexist.sas
May  4 22:40 globlist.sas
May  4 20:00 hasvars.sas
May  4 20:01 hasvarsc.sas
May  4 20:02 hasvarsn.sas
Jul  2 15:03 hexchars.sas
May  4 18:29 hexcnt.sas
Jul 26 18:24 killsas.sas
Jul 26 18:19 killsess.sas
May  4 20:03 lafootnote.sas
May  4 20:04 latitle.sas
May  4 19:47 lcralign.sas
May  4 21:42 left.sas
May  4 22:57 liblist.sas
May  4 19:14 ljustify.sas
May  4 18:35 lookahead.sas
May  4 20:05 lowcase.sas
May  4 19:13 lrafootnote.sas
May  4 19:10 lratitle.sas
Jun 26 13:56 ls.sas
Jun 26 14:11 lsfpq.sas
Sep  1 19:22 lstattrib.sas
May  4 22:22 ltgtm1.sas
May  1 13:06 match.sas
May  4 20:06 maxtitle.sas
Aug 12 19:02 md5sum.sas
May  4 18:39 misscnt.sas
May  4 22:55 missvars.sas
May  4 23:10 mkformat.sas
May  4 21:43 modte.sas
May  4 22:17 mtype.sas
May  4 22:43 mvarlist.sas
May  4 22:40 mvarvalues.sas
May  4 22:15 nlobs.sas
May  4 23:15 nobs.sas
May  4 21:46 nodup.sas
May  4 22:27 nodupkey.sas
May  4 20:07 noquotes.sas
May  4 22:00 now.sas
May  4 19:45 numchars.sas
May  4 20:08 nvars.sas
May  4 20:09 nvarsc.sas
May  4 20:10 nvarsn.sas
May  4 19:08 optlength.sas
May  4 22:20 partialdates.sas
Jun 12 11:11 prefix.sas
May  4 19:06 printall.sas
Feb  1  2011 prxnames.sas
May  4 21:47 putvars.sas
Sep 23 02:46 qcompress.sas
May  4 22:45 qdequote.sas
May  4 22:49 qdosfileinfo.sas
May  4 22:46 qgetenv.sas
Sep 23 02:47 qleft.sas
Sep 23 02:49 qreadpipe.sas
Sep 23 02:49 qtrim.sas
May  4 20:13 quotecnt.sas
May  4 22:16 quotelst.sas
May  4 20:14 quotescan.sas
May  4 19:04 rafootnote.sas
May  4 22:29 rannomac.sas
May  4 19:03 ratitle.sas
Aug 12 18:56 rcmd2ds.sas
Aug 12 18:57 rcmd2log.sas
Aug 12 18:52 rcmd2mvar.sas
May  4 21:55 readfile.sas
May  4 20:15 remove.sas
May  4 22:13 removew.sas
Feb 12  2011 rename8.sas
May  4 19:02 replhex.sas
Jun 12 19:22 rinclude.sas
May  4 18:58 round.sas
May  4 20:16 rxmatch.sas
Feb  2  2011 sas2xpt.sas
Jun 14 17:14 savopts.sas
Sep 19 19:25 scanfile.sas
Aug 19 00:25 scanlog.sas
May  4 18:44 showhex.sas
May  4 20:17 sortedby.sas
May  8 13:05 splitmac.sas
Aug 25 09:10 splitvar.sas
May  4 20:25 substrw.sas
Jun 12 11:10 suffix.sas
May  4 18:46 supasort.sas
May  4 22:06 sysfmtlist.sas
May  4 21:56 therest.sas
May  4 18:48 titlelen.sas
May  4 20:26 trim.sas
May  8 13:08 v_macros.sas
Sep  9 17:24 var2mvar.sas
May  4 20:27 varfmt.sas
May  4 21:57 varinfmt.sas
May  4 20:27 varlabel.sas
May  4 20:32 varlen.sas
May  4 20:31 varlist.sas
May  4 20:29 varlistc.sas
May  4 20:34 varlistn.sas
May  4 20:35 varnum.sas
May  4 20:36 vartype.sas
May  4 18:56 vaxis.sas
May  4 22:00 verify.sas
May  4 20:39 verifyb.sas
May  4 22:58 vwlist.sas
May  4 20:40 windex.sas
May  4 22:44 words.sas
Jun 26 14:22 xl2sas.sas
Sep 23 09:05 xlblocks.sas
Jun 26 14:21 xlsheets.sas
Feb  2  2011 xpt2sas.sas
May  4 22:23 yrcutoff.sas
May  4 18:54 zerogrid.sas
```

Utility macro purposes

```
Index of members in this directory with standard headers
========================================================
(this list was generated by the crindex script)

addautos.sas       - To concatenate a macro library onto the sasautos path

                     Usage: %addautos(mymacros)


adddecodevars.sas  - To add decode variables where a user format is specified

                     Usage: %adddecodevars(dsin=ds1,dsout=ds2)


after.sas          - Function-style macro to give you what comes directly after a
                     target string.

                     Usage: %let width=%after(&str,%str(width=),%str( w=));


age.sas            - In-datastep function-style macro to calculate the age of a person
                     on a date.

                     Usage: data test;
                     age=%age(dob,date);

agedec.sas         - In-datastep function-style macro to calculate the age of a person
                     on a date as a decimal age.

                     Usage: data test;
                     agedec=%agedec(dob,date);

aligndp.sas        - In-datastep macro to create a string from a numeric value with
                     decimal points aligned.

                     Usage: %aligndp(numvar,charvar,4);


allfmtvals.sas     - Create a dataset with every start value of a format in it

                     Usage: %allfmtvals(fmt=$country,var=country,dsout=temp1,length=2)
                     %allfmtvals(fmt=site,var=site,dsout=temp2)
```

alluniq.sas	- To create a dataset with all unique occurences of a variable throughout a library. Usage: %alluniq(in,subject,allsubj)

attrc.sas	- Function-style macro to return a character attribute of a dataset
	
	Usage: %let dslabel=%attrc(dsname,label);
	%let sortseq=%attrc(dsname,sortedby);

attrn.sas	- Function-style macro to return a numeric attribute of a dataset Usage: %let nobs=%attrn(dsname,nlobs);
attrv.sas	- Function-style macro to return a variable attribute Usage: %let vartype=%attrv(dsname,varname,vartype);
bydrop.sas	- To drop by-group residuals Usage: %bydrop(dsin,by1 by2)
bytitle.sas	- To drop the last title if it is a "by" title and write it to the global macro variable _bytitle_ instead. Usage: %bytitle
capmac.sas	- Function-style macro to capitalise the first letter of each word in a macro string. Usage: %let tidy=%capmac(%bquote(A, B AND C'S RESULTS));

capvar.sas	- In-datastep macro to tidy case of text in a variable
	
	Usage: data lparmcd;
	set lparmcd;
	%capvar(put(lparmcd,lparmcd.),newvar,
	ignore="SGOT" "SGPT" "PTT" "LDH" "GGT" "BUN");
	run;

casestrmac.sas	- Function-style macro to force mixed case forms of a string into the string itself for a macro expression. Usage: %let newtext=%casestrvar(&oldtext,Roland);

casestrvar.sas	- In-datastep macro to force mixed case forms of a string into the string itself.
	
	Usage: data test2;
	set test;
	%casestrvar(text,'Roland');
	run;

char2num.sas	- To "effectively" convert a list of character variables to numeric

```
                      Usage: %char2num(test,test2,char1 char2 char3 char4)


checkv6.sas         - Check a dataset for Version 6 compatibility

                      Usage: %checkv6(sasuser.myds);

chkuniq.sas         - To check for uniqueness in key variables.

                      Usage: %chkuniq(dsname)


chompw.sas          - Function-style macro to cut out a word from a macro string and
                      optionally cut out words before and/or after it.

                      Usage: %let str2=%chompw(&str1,&target,2,0,casesens=yes);


clashlibs.sas       - To identify where there is a clash of variable characteristics
for
                      the specified dataset(s) in the multiple assigned libraries and
to
                      output diagnostics. Case is important for variable names. To make
                      sure all variable names are created in upper case then use the
                      system option VALIDVARNAME=UPCASE before you create the datasets.

                      Usage: %clashlibs(myds)


clashvars.sas       - To identify where there is a clash of variable characteristics
for
                      datasets in a library and to output diagnostics.

                      Usage: %clashvars(mylib)


clength.sas         - To create a length statement to unify character lengths in a list
                      of data sets to the maximum variable length.

                      Usage: %clength(ds1 ds2 ds3);
                      data all;
                      &_clength_;
                      set ds1 ds2 ds3;
                      run;

commas.sas          - Function-style macro to separate the elements of a list with
                      commas.

                      Usage: order by %commas(&var1 &var2 &var3);


complibs.sas        - To "proc compare" identically-named datasets in two libraries

                      Usage: %complibs(base,comp)


compress.sas        - Function-style macro to compress a macro string

                      Usage: %let str2=%compress(&str,1234567890.);


compvars.sas        - To compare the differences in variables present in two datasets
```

```
                     and report the results to global macro variables.

                     Usage: %let ds1=dataset1;
                     %let ds2=dataset2;
                     %compvars(&ds1,&ds2)
                     options nosource;
                     %put NOTE: Variables found in &ds1 but not &ds2:;
                     %put &_left_;
                     %put NOTE: Variables found in &ds2 but not &ds1:;
                     %put _right_;
                     %put NOTE: Variables found in both &ds1 and &ds2:;
                     %put &_both_;
                     options source;

crdte.sas          - Function-style macro to return the creation datetime stamp of a
                     dataset.

                     Usage: %let crdte=%crdte(dsname);


datanulldemo.sas   - Clinical reporting sample code to do a stacked-column report
using

                     data _null_ that does not leave line gaps like proc report does.

                     Usage: Ordinary SAS code.


delhex.sas         - To delete occurrences of a specified hex character in a flat
file.


                     Usage: %delhex(infile,outfile,'FE'x)


delifexist.sas     - To delete a dataset if it exists

                     Usage: %delifexist(sasuser.myds)


delzero.sas        - To delete all datasets in a library with zero observations. This
                     macro was written for illustration purposes and is of limited
use.

                     Usage: %delzero(work)


dequote.sas        - Function-style macro to remove front and end matching quotes
                     from a macro string and return the result.

                     Usage: %let str=%dequote(%qreadpipe(echo '%username%'));


dir.sas            - Function-style macro to return a list of members of a directory
                     on a WINDOWS platform according to the file pattern you supply.
                     If you supply just the directory name then all members are
                     listed. This runs the MSDOS command in the form "dir /B mydir"

                     Usage: %let dirlist=%dir(C:\utilmacros);
                     %let dirlist=%dir(C:\utilmacros\*.sas);

dirfpq.sas         - Function-style macro to return a list of full-path quoted members
                     of a directory on a Windows platform according to the file
pattern

                     you supply.
```

```
                    Usage: %let dirlist=%dirfpq(C:\utilmacros);     %*- NO GOOD -;
                    %let dirlist=%dirfpq(C:\utilmacros\*);      %*- GOOD -;
                    %let dirlist=%dirfpq(C:\utilmacros\*.sas);  %*- GOOD -;

dlm2sas.sas       - To read in a delimited flat file and convert it to a sas dataset

                    Usage: %dlm2sas(C:\Mylib\myfile.csv,mydset)

doallitem.sas     - To execute code for each item in a space-delimited list

                    Usage: %doallitem(dsa dsb dsc,'proc sort data=&item;by
var;run;');


dosfilesize.sas   - Function-style macro to return a DOS file size

                    Usage: %let filesize=%dosfilesize(C:\spectre\unistats.html);

dropvars.sas      - To drop a list of unwanted variables in a list of datasets.

                    Usage: %dropvars(work._all,x1 x2)


dsall.sas         - To expand out the _all_ in a dataset list into all datasets in
the
                    library.

                    Usage: %dsall(sasuser.test work._all_);
                    %let dsall=&_dsall_;

dsattrib.sas      - To force a set of attributes, held in a template dataset,
                    on another dataset.

                    Usage: %dsattrib(template,inds,outds)


dslabel.sas       - Function-style macro to return a dataset label

                    Usage: %let dslabel=%dslabel(dsname);


dslist.sas        - To list all the datasets in a libref.

                    Usage: %dslist(work);
                    %let dslist=&_dslist_;

dtscale.sas       - To generate a date scale for sas/graph

                    Usage: %dtscale(&min,&max);


duplvars.sas      - Function-style macro to create a list of duplicate variables in a
                    second dataset so that they can be dropped before a merge.

                    Usage: data newds;
                    merge ds1 ds2(drop=%duplvars(ds1,ds2,&bylist));
                    by &bylist;
                    run;

endwith.sas       - Function-style macro to ensure any non-null value assigned to a
                    macro variable ends with the specified character.
```

```
                      Usage: filename outfile "%endwith(&outdir,/)output.txt";


env2ds.sas          - To write system and user environment variables to a dataset.

                      Usage: %env2ds;
                      %env2ds(OutputDatasetName);

env2dsw7.sas        - To write system and user environment variables to a dataset for
                      the Windows 7 operating system.

                      Usage: %env2dsw7;
                      %env2dsw7(OutputDatasetName);

eqsuff.sas          - Function-style macro to suffix a list of words (usually
variables)
                      with an equals sign.

                      Usage: put %eqsuff(&varlist);


equals.sas          - In-datastep function-style macro to compare two numeric values to
                      find if they are equal or very nearly equal.

                      Usage: if %equals(val1,7.3) then ...


fixnames.sas        - In-datastep macro to fix UTF-8 characters in a person's name by
                      converting the UTF-8 character pairs back to ascii.

                      Usage: data newpatinfo;
                      set patinfo;
                      %fixnames(invname)
                      run;

fixvars.sas         - To "fix" variables in a library so they are consistent

                      Usage: fixvars(mylib,w);


flatten.sas         - To "flatten" data so there is only one observation per "by group"

                      Usage: %flatten(dsin=test,bygroup=by1 by2,vars=str num)


fmtord.sas          - To create a numeric informat that maps a format label to its
                      order position.

                      Usage: %fmtord(agernge);


fmtpath.sas         - Function-style macro to get the full fmtsearch path

                      Usage: %let path=%fmtpath;


fmts2fda.sas        - To create sas code to generate formats as found in your data

                      Usage: %fmts2fda(mylib1 mylib2)
```

getfmts.sas	- To get details of a list of user formats defined in a dataset Usage: %getfmts(dsin=fmtlist,fmtvar=format,dsout=allfmts);
gettitles.sas	- To read the title lines of an LST file and write them to a global macro variable _titles_ . Usage: %gettitles(C:\temp\myfile.lst)
getvalue.sas	- Function-style macro to return a variable's value Usage: %let value=%getvalue(dsname,varname,1);
globexist.sas	- Function-style macro to return true if all the global macro variables listed exist. Usage: %if %globexist(globvar) %then %do ....
globlist.sas	- Function-style macro to return a list of current global macro variable names. Usage: %let glist=%globlist;
hasvars.sas	- Function-style macro to return true if a dataset has all the variables defined to a list. Usage: %if not %hasvars(dsname,aa bb cc) %then %do ....
hasvarsc.sas	- Function-style to return true if a dataset has all the character variables defined to a list. Usage: %if not %hasvarsc(dsname,aa bb cc) %then %do ....
hasvarsn.sas	- Function-style macro to return true if a dataset has all the numeric variables defined to a list. Usage: %if not %hasvarsn(dsname,aa bb cc) %then %do ....
hexchars.sas	- To show up ascii non-printables characters in a flat file by displaying their ascii codes as hexadecimal numbers in "< >" symbols. Usage: %hexchars(infile.ext) %hexchars(infile.ext,"outfile.ext") %hexchars(infile.ext,outfile.ext) %hexchars("infile.ext") %hexchars("infile.ext",print) %hexchars("infile.ext","log")
hexcnt.sas	- To count the strange hex character in character variables Usage: %hexcnt(dsname,droplist,globcnt=_hexcnt_,globvars=_hexvars_);

killsas.sas	- To kill any user's SAS session except the one running this macro Usage: %killsas
killsess.sas	- To kill a Windows SAS session Usage: %killsess %killsess(2)
lafootnote.sas	- To create a left-aligned footnote Usage: %lafootnote(2," second footnote indented two spaces")
latitle.sas	- To create a left-aligned title Usage: %latitle(2," second title indented two spaces")
lcralign.sas	- Write to a macro variable with the supplied text left, center and right-aligned. Usage: %let macvar=; %lcralign(macvar,50,"left bit","center bit","right bit") %put macvar=*&macvar*;
left.sas	- Function-style macro to left-align the contents of a macro variable. Usage: %let macvar=%left(&macvar);

liblist.sas	- To list all the libraries.
	
	Usage: %liblist;
	%let liblist=&_liblist_;

ljustify.sas	- To left-justify all character fields in a dataset Usage: %ljustify(dset)
lookahead.sas	- To do the opposite of lag and allow the user to look ahead at the variables in the following observations in the same by group. Usage:
lowcase.sas	- Function-style macro to return a lower-case version of a macro variable's contents. Usage: %let lcase=%lowcase(&string);
lrafootnote.sas output	- To left and right-align a two part footnote for a pure text Usage: %lrafootnote(5,"Left aligned","Right-aligned")

```
lratitle.sas        - To left and right-align a two part title for a pure text output

                      Usage: %lratitle(5,"Left aligned","Right-aligned")


ls.sas              - Function-style macro to return a list of members of a directory
                      on a Unix platform according to the file pattern you supply.
                      If you supply just the directory name then all members are
                      listed. This runs the Unix command in the form "ls -1 mydir" .

                      Usage: %let dirlist=%ls(/usr/utilmacros);
                      %let dirlist=%ls(/usr/utilmacros/*.sas);

lsfpq.sas           - Function-style macro to return a list of full-path quoted members
                      of a directory on a Unix platform according to the file pattern
                      you supply.

                      Usage: %let dirlist=%lsfpq(/usr/mylib);     %*- NO GOOD -;
                      %let dirlist=%lsfpq(/usr/mylib/*);      %*- GOOD -;
                      %let dirlist=%lsfpq(/usr/mylib/*.sas);  %*- GOOD -;

lstattrib.sas       - Lists the variable attributes of the specified dataset in the
                      form of a LENGTH statement and ATTRIB statement that can be used
                      in sas code.

                      Usage: %lstattrib(sasuser.demog)

ltgtm1.sas          - In-datastep macro to turn a text numeric value into a numeric
                      value and handle "<" and ">" signs preceding and adjust the value
                      according to a rule (method 1).

                      Usage: %ltgtm1(textvar,numvar);


match.sas           - Function-style macro to return elements of a list that match
those
                      in a reference list.

                      Usage: %let match=%match(aa bb,aa cc);


maxtitle.sas        - To find the highest number title and footnote and output to
global
                      macro variables.

                      Usage: %maxtitles


md5sum.sas          - To write the md5 checksum to the log for a two-level dataset
                      stored on Unix or Linux.

                      Usage: %md5sum(outads.basco)


misscnt.sas         - To create a list of variables and their missing value count

                      Usage: %misscnt(dsname,droplist,globvar=_miss_);

missvars.sas        - To create a list of all-missing variables

                      Usage: %missvars(dsname);
                      run;
```

```
                    data dsname;
                    set dsname(drop=&_miss_);
                    run;

mkformat.sas      - To create a format out of a "coded" and "decoded" variable in a
                    specified dataset.

                    Usage: %mkformat(dsname(where=
(x>1)),varcd,vardcd,fmtname,fmtcat);


modte.sas         - Function-style macro to return the last modification datetime
                    stamp of a dataset.

                    Usage: %let modte=%modte(dsname);


mtype.sas         - Function-style macro to return the member type of a dataset
                    (i.e. whether DATA or VIEW).

                    Usage: %let mtype=%mtype(dsname);


mvarlist.sas      - Function-style macro to return a list of macro variable names
                    satisfying the supplied scope.

                    Usage: %macro dummy(a=123,b=345,c=);
                    %let setparmlist=%mvarlist(dummy,s);
                    %mend dummy;

mvarvalues.sas    - Lists and resolves macro variables one per line for a supplied
                    macro variable list.

                    Usage: %mvarvalues(&mvarlist,*);
                    %mvarvalues('%mvarlist(dummy9,a)',**);

nlobs.sas         - Function-style macro to return the number of logical observations
                    (i.e. not marked for deletion) in a dataset or view. This will
                    either be a positive integer or forced to zero.

                    Usage: %put >>>>>> %nlobs(sashelp.class) >>>>;
                    %put >>>>>> %nlobs(sashelp.class(where=(sex="M"))) >>>>;
                    %put >>>>>> %nlobs(sashelp.vtable) >>>>;

nobs.sas          - Function-style macro to return the number of observations in a
                    dataset or view. This will either be a positive integer or forced
                    to zero.

                    Usage: %put >>>>>> %nobs(sashelp.class) >>>>;
                    %put >>>>>> %nobs(sashelp.class(where=(sex="M"))) >>>>;
                    %put >>>>>> %nobs(sashelp.vtable) >>>>;

nodup.sas         - Function-style macro to drop duplicates in a space-delimited list

                    Usage: %let str=%nodup(aaa bbb aaa);


nodupkey.sas      - To sort "nodupkey" but list observations being dropped so that
                    they can be investigated and accounted for.

                    Usage: %nodupkey(ds,var1 var2 var3)
```

noquotes.sas - Function-style macro to remove all quoted strings from a macro expression.

```
Usage: %let noquotes=%noquotes(&str);
```

now.sas - Function-style macro to return the current timestamp

```
Usage: %put Stage1: %now;
```

numchars.sas - To create a list of character variables that contain numeric-like text.

```
Usage: %numchars(dsname,globvar=_numchars_);
%put ######## &_numchars_;
```

nvars.sas - Function-style macro to return the number of variables in a dataset.

```
Usage: %let nvars=%nvars(dsname);
```

nvarsc.sas - Function-style macro to return the number of character variables in a dataset.

```
Usage: %let nvarsc=%nvarsc(dsname);
```

nvarsn.sas - Function-style macro to return the number of numeric variables in a dataset.

```
Usage: %let nvarsn=%nvarsn(dsname);
```

optlength.sas - To create a length statement for character variables that take up less length than that allotted to the variable.

```
Usage: optlength(dset)
data dset;
&_optlength_;
set dset;
run;
```

partialdates.sas - In-datastep macro to impute partial dates to a high or low value

```
Usage: data test;
datestr="--feb08";
%partialdates(datetext=datestr,datevar=date,pattern="ddmmmyy",
lohi=high);
format date date9.;
put date= datestr=;
run;
29FEB2008
```

prefix.sas - Function-style macro to return a list with a prefix added.

```
Usage: %let preflist=%prefix(C:\mylib\,fname1 "fname 2" fname3);
```

printall.sas - To print every observation in a library where a variable

```
satisfies
                     a specified condition.

                     Usage: %printall(work,%str(num>1))


prxnames.sas       - Function-style macro to convert a space-delimited list of sas
                     names (variable or dataset names) to a Pearl Regular Expression
                     for use in the prxmatch() function that takes into account the
                     ending colon notation.

                     Usage: %let dslist=var1 var2 vx:;
                     ....where prxmatch(%prxnames(&dslist),memname);

putvars.sas        - To list variables in a dataset suffixed with an equals sign
                     suitable for a "put" statement written to the log.

                     Usage: put %putvars(ds);


qcompress.sas      - Function-style macro to compress a macro variable string and
                     return the result MACRO QUOTED.

                     Usage: %let tidy=%qcompress(&string);


qdequote.sas       - Function-style macro to remove front and end matching quotes
                     from a macro string and return the result MACRO QUOTED.

                     Usage: %let str=%qdequote(%qreadpipe(echo '%username%'));
                     CLASS %unquote(%qdequote('&trtvar')) ;

qdosfileinfo.sas   - Function-style macro to return information about a DOS file
                     and return the result MACRO QUOTED.

                     Usage: %let filesize=%qdosfileinfo(C:\spectre\unistats.html,z);

qgetenv.sas        - Function-style macro to get the contents of a system or user
                     environment variable and return the result MACRO QUOTED.

                     Usage: %let newvar=%qgetenv(uservar);


qleft.sas          - Function-style macro to left-align the contents of a macro
                     variable and return the result MACRO QUOTED.

                     Usage: %let macvar=%qleft(&macvar);


qreadpipe.sas      - Function-style macro to read the output of a system command and
                     return the result trimmed and MACRO QUOTED.

                     Usage: %let mvar=%qreadpipe(echo $USER);


qtrim.sas          - Function-style macro to trim the contents of a macro variable and
                     return the result MACRO QUOTED.

                     Usage: %let macvar=%qtrim(&macvar);


quotecnt.sas       - Function-style macro to count quoted strings in a macro
expression
```

```
                    Usage: %let count=%quotecnt(&str);


quotelst.sas      - Function-style macro to quote the elements of a list

                    Usage: %if %index(%quotelst(varnames),"varname") %then...


quotescan.sas     - Function-style macro to scan for a quoted string in a macro
                    expression.

                    Usage: %let scan=%quotescan(&str,2);


rafootnote.sas    - To right-align a footnote for a pure text output

                    Usage: %rafootnote(5,"This footnote 5 will be right-aligned")


rannomac.sas      - Compile Roland's annotate macros

                    Usage: filename webout "C:\spectre\";

                    goptions reset=all xpixels=1000 ypixels=6000 hpos=50 vpos=300
                    dev=gif gsfmode=replace transparency border
                    ftext='Arial' htext=1 cell ctext=CX483D8C; * DarkSlateBlue ;

                    ods listing close;
                    ods html path=webout body="annotest.html";

                    %rannomac

                    data test;
                    %dclannovars
                    %rarrow(y=298,x1=20,x2=48)
                    %text(y=298,x=19,position='<',text="right-aligned text")
                    %rarrow(y=297,x1=20,x2=48,fillcolor='green',
                    linecolor='black',fillpattern='mempty')
                    %text(y=297,x=19,position='<',text="next line of text")
                    %text(y=296,x=19,position='<',
                    text="This has a hotspot but misaligned on the left",
                    color='maroon',
                    html="alt='This hotspot is misaligned on the left of the text'")
                    %box(y=295,x=25)
                    %box(y=295,x=35)
                    %box(y=295,x=45,html="alt='Third Box Hotspot'")
                    %text(y=295,x=19,position='<',
                    text="The third box ONLY should have a hotspot")
                    %bigbox(x1=20,y1=294.5,x2=50,y2=298.5,linecolor="brown")
                    run;

                    *- Set description to a space to stop whole output area -;
                    *- from having a hotspot and give the gif the same name -;
                    *- as the html body file. -;
                    proc ganno annotate=test description=" " name="annotest";
                    run;

                    *- If you rerun this code then you need to delete the -;
                    *- "annotest" grseg member in work.gseg so it can be  -;
                    *- reused as a name in the "proc ganno" step.  -;
                    proc greplay igout=gseg nofs;
                    delete annotest;
                    run;
                    quit;

                    ods html close;
                    ods listing;


ratitle.sas       - To right-align a title for a pure text output

                    Usage: %ratitle(5,"This title 5 will be right-aligned")


rcmd2ds.sas       - To run a system command on the remote host and write the output
to
                    the dataset RWORK._rcmd.

                    Usage: %rcmd2ds(ls /root/usr/mylib)


rcmd2log.sas      - To run a system command on the remote host and write the output
to
                    the log.

                    Usage: %rcmd2log(ps -fu userid); *- see details of a user-id -;
                    %rcmd2log(ps -fp 12345);  *- see details of a process-id -;

rcmd2mvar.sas     - To run a system command on the remote host and write the output
to
                    a macro variable on the local host.

                    Usage: %rcmd2mvar(ps -fu userid,mymvar); *- see details of a
user-id -;
                    %rcmd2mvar(ps -fp 12345,mymvar); *- see details of a process-id
-;

readfile.sas      - Function-style macro to read in a flat file and assign the
                    contents to a macro variable.

                    Usage: %let mvar=%readfile(filename);


remove.sas        - Function-style macro to remove all occurrences of the target
                    string(s) from another string.

                    Usage: %let string2=%remove(&string1,XXX,yyy,YYY);


removew.sas       - Function-style macro to remove all occurrences of the target
                    word(s) from a source list of words.

                    Usage: %let colors2=%remove(&rainbow,green yellow);


rename8.sas       - Function-style macro to return a variable rename list for
variable
                    names longer than 8 characters to shorten them to 8 characters.

                    Usage: data myds2;
                    set myds;
                    rename %rename8(myds);
                    run;

replhex.sas       - To replace occurrences of a specified hex character in a flat
file
```

```
                          with another specified character.

                          Usage: %replhex(infile,outfile,'FE'x,' ')


rinclude.sas            - To submit local sas code members in the remote session

                          Usage: %rinclude(mylib(mymacro1.sas) "C:\mylib\mymacro2.sas"
                          %dirfpq(C:\macros\*.sas);


round.sas               - To round all the numeric variables in a list of datasets.

                          Usage: %round(work._all_)


rxmatch.sas             - Function-style macro to return those space-delimited elements of
a
                          list that match a specified rxparse pattern.

                          Usage: %let match=%rxmatch(apopa pop aapop popaa,pop $s);
                          %put &match;
                          pop aapop

sas2xpt.sas             - Create multiple transport files from sas datasets

                          Usage: %sas2xpt((INDSLIB),%nrstr("V:\SAS\Two Parts\X&Y\temp\"));

savopts.sas             - Function-style macro to return a list of active sas options so
                          that these options can be restored at a later point.

                          Usage: %let savopts=%savopts(missing mprint);
                          option &savopts;

scanfile.sas            - Counts the number of lines of text in a file that contain the
                          string or the regular expression you specify within the line
limit
                          you choose and optionally writes the line or blocks of lines to
                          the log.

                          Usage: %scanfile(C:\temp\myfile.lst,Treated,3,casesens=no)

                          *-- Complex example of scanning all the sas programs   --;
                          *-- in a library and printing the "proc format" steps. --;
                          %doallitem(%qreadpipe(dir /B C:\Mylib\*.sas),
                          '%scanfile(C:\Mylib\&item,proc format,
                          untilstr=run,notstr=cntlin,casesens=no)');

scanlog.sas             - To scan sas log file(s) or the log window for important messages
                          optionally using a "rules" file.

                          Usage: %scanlog("full-file-path-name")
                          %scanlog("full-file-path-name-1" "full-path-name-2")
                          %scanlog(fileref)
                          %scanlog(fileref(a.log) fileref(b.log))
                          %scanlog(fileref "full-path-name")
                          %scanlog(%lsfpq(/usr/mypath/*.log))
                          %scanlog(%dirfpq(C:\temp\*.log))
                          %scanlog(fileref(a.log) "full-path-name" %dirfpq(C:\temp\*.log))
                          %scanlog;          *- this is for interactive sas sessions -;
                          %scanlog(,log);    *- this is for interactive sas sessions -;
                          %scanlog(fileref,"output-file")
                          %scanlog(rulesfile=C:\temp\myrules.txt)
```

```
                    %scanlog(rulesfile="C:\temp\myrules.txt")
                    %scanlog(rulesfile="C:\temp\myrules.txt",prx=yes)
                    or in command line box for interactive sessions (note syntax):
                    gsubmit '%scanlog;'

showhex.sas       - To create a new dataset where hex characters in character
                    variables are highlighted.

                    Usage: %showhex(test1,test2,cvar1 cvar2 cvar3)


sortedby.sas      - Function-style macro to return the variables a dataset is sorted
                    by, or null if not sorted.

                    Usage: %let sortedby=%sortedby(dsname);


splitmac.sas      - Function-style macro to insert split characters in a macro string

                    Usage: %let str=The quick brown fox jumped over the lazy dog;
                    %let splitstr=%splitmac(&str,10);

splitvar.sas      - In-datastep macro to insert split characters in a string variable

                    Usage: data aaa;
                    set aaa;
                    %splitvar(oldvar,newvar,10,split=/,hindent=0);
                    run;

substrw.sas       - Function-style macro to substring words assigned to a macro
                    variable.

                    Usage: %let whatsleft=%substrw(&mvar,4);
                    %let twothree=%substrw(&str,2,2);

suffix.sas        - Function-style macro to return a list with a suffix added.

                    Usage: %let sufflist=%suffix(.sas,fname1 "fname 2" fname3);


supasort.sas      - To sort a list of datasets by variables if they exist in the
                    datasets.

                    Usage: %supasort(work._all_,date time)


sysfmtlist.sas    - In-datastep macro to list all the system formats

                    Usage: if format in (" " %sysfmtlist) then _fmt="SYS";
                    else _fmt="USR";

therest.sas       - Function-style macro to give you everything following any found
                    target string character.

                    Usage: %let rest=%therest(&str,\/);


titlelen.sas      - To create a copy of sashelp.vtitle but with the length added.

                    Usage: %titlelen
```

trim.sas	- Function-style macro to trim the contents of a macro variable Usage: %let macvar=%trim(&macvar);
v_macros.sas	- To compile the validation macros %mmm, %fmm, %dmm and set up global macro variables "mut", "rut", "exp" and "act". Usage: %v_macros %let mut=removew; %let rut=req001 req002; %let days=mon tue wed thu fri sat; %let act=%&mut(&days,tue fri); %let exp=mon wed thu sat; %mmm
var2mvar.sas	- To write data in a variable to a global macro variable Usage: %var2mvar(sashelp.class(where=(name=:"A")),name); %put **&_mvar_**; **Alfred Alice**
varfmt.sas	- Function-style macro to return a variable format Usage: %let varfmt=%varfmt(dsname,varname);
varinfmt.sas	- Function-style macro to return a variable informat Usage: %let varinfmt=%varinfmt(dsname,varname);
varlabel.sas	- Function-style macro to return a variable label Usage: %let varlabel=%varlabel(dsname,varname);
varlen.sas	- Function-style macro to return a variable length Usage: %let varlen=%varlen(dsname,varname);
varlist.sas	- Function-style macro to return a list of variables in a dataset Usage: %let varlist=%varlist(dsname);
varlistc.sas	- Function-style macro to return a list of character variables in a dataset. Usage: %let varlistc=%varlistc(dsname);
varlistn.sas	- Function-style macro to return a list of numeric variables in a dataset. Usage: %let varlistn=%varlistn(dsname);

```
varnum.sas          - Function-style macro to return the variable position in a dataset
                      or 0 if not in dataset.

                      Usage: %let varnum=%varnum(dsname,varname);


vartype.sas         - Function-style macro to return a variable type as either C or N

                      Usage: %let vartype=%vartype(dsname,varname);


vaxis.sas           - To generate the values to construct a vaxis scale

                      Usage: %vaxis(&min,&max,spare=1)


verify.sas          - Function-style macro to return the position of the first
character

                      in a string that does not match any character in a reference
                      string.

                      Usage: %let pos=%verify(&text,%str( )); %*- first non-blank
character -;


verifyb.sas         - Function-style macro to return the position of the first
character

                      in a string that does not match any character in a reference
                      string BUT STARTING FROM THE BACK.

                      Usage: %let pos=%verifyb(&text,%str( )); %*- last non-blank
character -;


vwlist.sas          - To list all the views in a libref.

                      Usage: %vwlist(work);
                      %let vwlist=&_vwlist_;

windex.sas          - Function-style macro to return the word count position in a
string


                      Usage: %let windex=%windex(string,target);


words.sas           - Function-style macro to return the number of words in a string

                      Usage: %let words=%words(string);

xl2sas.sas          - Read an Excel spreadsheet into a sas dataset using DDE

                      Usage: %xl2sas(xlfile=C:\myfiles\My Spread
Sheet.xls,sheetname=Sheet1,
                      dsout=sasuser.myspread,compress=no,vpref=_col,vlen=50,
                      startrow=5,startcol=1,endrow=95,endcol=10)

xlblocks.sas        - Read an Excel spreadsheet sheet containing blocks of
                      information using DDE with each block output as a numbered
                      dataset.

                      Usage: %xlblocks(xlfile=C:\myfiles\My Sheet.xls,sheetname=Sheet
One,
                      dspref=sasuser.myspread,compress=no,vpref=col,vlen=40);

xlsheets.sas      - Get a list of sheet names (topics) from an Excel spreadsheet
                      using DDE and write them to a global macro variable.

                      Usage: %xlsheets(C:\Mydata\Spread Sheet Name.xls);

xpt2sas.sas       - Convert all the .xpt files in a folder to sas datasets

                      Usage: %xpt2sas(%nrstr("V:\SAS\Two Parts\X&Y\"),
                      %nrstr("V:\SAS\Two Parts\X&Y\temp\"));

yrcutoff.sas      - To set the year cutoff option to a number of years previous to
the

                      current year. 90 is the default which is suitable for clinical
                      reporting.

                      Usage: %yrcutoff


zerogrid.sas      - To create a "grid" of combined values with a variable set to zero
                      for all combinations of values.

                      Usage: %zerogrid(dsout=grid,var1=subject,ds1=demog,var2=tmtarm,
                      ds2=demog,zerovar=count,sortby=tmtarm subject)
                      %zerogrid(zerovar=str,zero="  0 (  0.0)",var1=trtrand ddose,
                      ds1=period1,var2=day,ds2=period1)
```

Use the "**Back**" button of your browser to return to the previous page