```
/*-------------------------------------------------------------------
  SAS Macro UNISTATS
  Version 2.2  December 2017

  UNISTATS makes PROC UNIVARIATE statistics more convenient by
  presenting one row for each analysis variable (like PROC MEANS).

  UNISTATS calculates for a given set of analysis variables every
  PROC UNIVARIATE statistic and outputs them in one row per analysis
  variable and by-group. An output data set is always created.
  By-group-processing and WHERE-clause is optionally supported.

  UNISTATS is quick and easy to use because of its naming conventions
  for analysis variables e.g. x1-x10, a--z, abc:, _numeric_

  Any percentile (quantile) e.g. 97.5th can be computed with every
  SAS percentile definition (PCTLDEF=1-5) and beside that according to
  the Microsoft Excel function QUANTIL(Matrix, Alpha).

  Originally developed in September and October 1997 by Ian Whitlock
  (whitloi1@westat.com) and Jozsef Vitrai (h12388vit@ella.hu) in SAS-L.
  Michael Friendly (friendly@hotspur.psych.yorku.ca) gave the idea to
  support special variable lists like x1-x10, a--z, abc:, _numeric_.
  With version 1.0 (1998) by-group-processing and WHERE-clause was
  supported and performance speed was improved by Heinrich Stuerzl.

  With version 2.0 (2009) user-defined percentile calculation according
  to SAS and Excel was added by Heinrich Stuerzl and Dr. Cornelius
  Gutenbrunner. Also available are the robust statistic keywords of
  PROC UNIVARIATE, label of the analysis variable in the output data
  set and the syntax help via %unistats(?).

  Version 2.1 (2010) fixes an error in the labels of analysis variables
  if they are not used in the original order. Heinrich Stuerzl.

  Version 2.2 (2017) by Heinrich Stuerzl:
  New input parameter DEBUG to detect undefined macro variables.
  Macro variables NVARS SQLEXITCODE SQLOBS SQLOOPS SQLRC SQLXOBS V1 are
  defined locally to prevent unexpected interactions with calling macros.

  ---------------------------------
  Licence:

  This program is published by Heinrich Stuerzl unter the Creative Commons licence
  Attribution-ShareAlike 3.0 Unported (CC BY-SA 3.0).
  This means that free and even commercial usage is permitted as long as you credit
  the author and the license in the following way:
  "Source Code by Heinrich Stuerzl, CC BY-SA 3.0 (https://creativecommons.org/licenses/by-sa/3.0/deed.en)"
  You are free:
  - to share = to copy, distribute and transmit the work
  - to remix = to adapt the work
  Under the following conditions:
  - attribution = You must attribute the work in the manner specified by the author or licensor
                  (but not in any way that suggests that they endorse you or your use of the work).
  - share alike = If you alter, transform, or build upon this work, you may distribute
                  the resulting work only under the same or similar license to this one.

  This program is distributed in the hope that it will be useful, but
  WITHOUT ANY WARRANTY; without even the implied warranty of
  MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

  ---------------------------------
  %unistats(?) shows a brief syntax help in the log

  DATA    (optional) Input data set, if omitted _LAST_ is used

  VARS    (optional) List of (numeric) analysis variables (max. 99999)
          Special lists possible like x1-x12, x--z, abc: _numeric_
          If omitted _numeric_ is used i.e. all numeric variables

  BY      (optional) BY-variable(s) for by-group-processing
          Output is sorted in ascending order of by-groups
          (input data set needs not to be sorted by them)

  WHERE   (optional) WHERE-clause for the selection of observations
          in the input data set. Only the subset which meets the
          where clause is used for the analysis

  STATS   (optional) List of statistics keywords of PROC UNIVARIATE
           or _ALL_
          Statistics are output in the given order
```

```
          Descriptive Statistic Keywords:
            CSS CV KURTOSIS MAX MEAN MIN MODE N NMISS NOBS RANGE
            SKEWNESS STD STDMEAN SUM SUMWGT USS VAR

          Quantile Statistic Keywords:
            P1 P5 P10 Q1 MEDIAN Q3 P90 P95 P99 QRANGE

          Robust Statistics Keywords:
            GINI MAD QN SN STD_GINI STD_MAD STD_QN STD_QRANGE STD_SN

          Hypothesis Testing Keywords:
            MSIGN NORMAL|NORMALTEST SIGNRANK PROBM PROBN PROBS PROBT T

          Default: STATS=N MEDIAN MEAN STD CV MIN MAX

  PCTLPTS (optional) List of any user-defined percentiles (quantiles)
          according to the same option of proc univariate
          e.g. PCTLPTS=2.5 97.5
          Note: Maximal two decimals are possible
          Default percentiles like P5 Q1 MEDIAN can also be computed
          by PCTLPTS=5 25 50
          Percentiles are output after other statistics

  PCTLDEF Method of computation percentiles according to
          the same option of proc univariate in SAS or to MS Excel.
          The used setting is output in the variable _PctlDef_
          Valid settings are:
          PCTLDEF=1
          PCTLDEF=2
          PCTLDEF=3
          PCTLDEF=4
          PCTLDEF=5 (Default)
          PCTLDEF=EXCEL
            According to the Excel function QUANTIL(Matrix, Alpha)
            (Results were compared with Microsoft Excel 2002 and 2003
             under Windows XP and did not differ)
            with following restrictions:
            1. Default percentiles (P1 P5 P10 Q1 MEDIAN Q3 P90 P95 P99
               QRANGE) are not allowed in STATS because they are not
               computed according to Excel.
               Use the according percentiles via PCTLPTS= instead.
               E.g. instead STATS=P5 Q1 MEDIAN use PCTLPTS=5 25 50
            2. Only one analysis variable allowed
               (use separate macro calls for each analysis variable
                and combine the output data sets with SET statement)
            3. No By-variable(s)
               (use separate macro calls for each by-group with
                an according Where-clause
                and combine the output data sets with SET statement)

  OUT     Output data set (Default: WORK.STAT1)
          Contains one observation per analysis variable and by-group
          and one variable for each statistic (named in the given
          notation in STATS) and beside the by-variable(s) used
          the following variables:

          VName     "Name"      $32:  Name of the analysis variable
          VLabel    "Label"    $256:  Label of the analysis variable
          _PctlDef_ "PctlDef"    $5:  Percentile definition used
          By-variable(s) used

          Output data set is sorted in ascending order by the
          by-variable(s) used

  PRINT   Printing option
          Y - Standard output with PROC PRINT and LABEL (Default)
          N - no printing

  DEBUG   Debug mode
          Debug=1: Debug mode checking for undefined macro variables.
                   Useful during the development und change phase for all branches of the program.
                   Creates the work data sets mVarsBegin and mVarsEnd, which are not deleted.
                   Note to the Log at the begin: '>>> Running in the Debug Mode <<<'
                   Note to the Log at the end:
                   'WARNING: >>> Undefined macro variable(s):' + list of undefined macro variables if there are some or
                   '>>> No undefined macro variables found!' otherwise.
          Debug=0: Normal mode (default).

Examples:

%unistats(?); * shows syntax help in the log;
```

```
%unistats (data=sashelp.class);
* computes N MEDIAN MEAN STD CV MIN MAX for all numeric variables;

%unistats (
  , data=sashelp.class
  , by=sex
  );
* by-group processing;

%unistats (
  , data=sashelp.class
  , vars=Age Weight
  , stats=n min p5 Q1 median mean Q3 p95 max
  );
* given statistics and variables;

%unistats (
  , data=sashelp.class
  , vars=Age Weight
  , stats=n min median mean max
  , pctlpts=1 2.5 5 95 97.5 99
  );
* user-defined percentiles (default SAS percentile definiton PCTLDEF=5);

%unistats (
  , data=sashelp.class
  , vars=Age Weight
  , by=sex
  , where=sex="F"
  , stats=n min median mean max
  , pctlpts=1 2.5 5 95 97.5 99
  );
* analysis for one by-group only (where clause);

%unistats (
  , data=sashelp.class
  , vars=Age
  , by=
  , where=sex="F"
  , stats=n min mean max
  , pctlpts=1 2.5 5 50 95 97.5 99
  , pctldef=Excel
  );
* Excel percentiles for one analysis variable and by-group;

Notes:
- Needs SAS Version 9.1 or higher
- The global macro variable UNISTATS_RC can have one of the following
  return codes:
    0  OK
    1  Incomplete or wrong macro parameter setting. No computing is done.
    2  Runtime error

- The plot options of PROC UNIVARIATE can not be used

Used sub macros:
none

-----------------------------------------------------------------------*/


%macro UNISTATS
  (help,
   data=&syslast,
   where=,
   by=,
   vars=_numeric_,
   out=work.stat1,
   print=Y,
   stats=N MEDIAN MEAN STD CV MIN MAX,
   pctlpts=,
   pctldef=5,
   debug=0
   )
   / des="Univariate statistics 2.2";

%local mcrVersion; %let mcrVersion=2.2;
%put ********** Start Macro &SYSMACRONAME Version &mcrVersion **********;

%global unistats_rc;
%let unistats_rc=0;
```

```
%local i j stat hvar stdstats nstats h1 h2 nnumvar npctlpts pctlpts2 opt1
       N NOBS NMISS SUM MEDIAN MEAN VAR STD CV MIN MAX Q1 Q3 P1 P5 P10 P90 P95 P99
       NORMAL NORMALTEST PROBN T PROBT MSIGN PROBM SIGNRANK PROBS
       STDMEAN USS CSS SKEWNESS KURTOSIS SUMWGT RANGE QRANGE MODE
       GINI MAD QN SN STD_GINI STD_MAD STD_QN STD_QRANGE STD_SN
       NVARS SQLEXITCODE SQLOBS SQLOOPS SQLRC SQLXOBS V1 ;

%if (&debug=1) %then %do;
  %put >>> Running in the Debug Mode <<<;
  * Create list of all macro variables to find undefined macro variables;
  proc sort data=sashelp.vmacro out=mVarsBegin;
    where scope ne 'AUTOMATIC';
    by scope name;
  run;
%end;

%let stdstats = N NOBS NMISS SUM MEDIAN MEAN VAR STD CV MIN MAX Q1 Q3 P1 P5 P10 P90 P95 P99 NORMAL
PROBN T PROBT MSIGN PROBM SIGNRANK PROBS STDMEAN USS CSS SKEWNESS KURTOSIS SUMWGT RANGE QRANGE MODE
GINI MAD QN SN STD_GINI STD_MAD STD_QN STD_QRANGE STD_SN ;


%if (&help=?) %then %do;
  %put Macro UNISTATS makes proc univariate statistics more convenient;
  %put presenting one row for each analysis variable and by-group;
  %put ;
  %put %nrstr(%unistats %();
  %put %nrstr(  , data=   input data set. Default: &syslast);
  %put ;
  %put %nrstr(  , vars=   analysis variable(s) e.g. x1 x3 x5 | x1-x10 | x: | _numeric_ (Default));
  %put ;
  %put %nrstr(  , by=     by-variable(s) (optional));
  %put ;
  %put %nrstr(  , where=  where condition for input data set (optional));
  %put ;
  %put %nrstr(  , stats=  any statistic keyword(s) of proc univariate);
  %put %nrstr(            Default: N MEDIAN MEAN STD CV MIN MAX);
  %put %str(            &stdstats);
  %put ;
  %put %nrstr(  , pctlpts=user-defined percentile(s) (optional));
  %put ;
  %put %nrstr(  , pctldef=definition for computing percentiles e.g. 1-5 | Excel);
  %put %nrstr(            Default: 5);
  %put %nrstr(            Excel: according to MS Excel function QUANTIL(Matrix, Alpha) );
  %put ;
  %put %nrstr(  , out=    Output data set containing one observation for each analysis variable and by-group);
  %put %nrstr(            Default=work.stat1);
  %put ;
  %put %nrstr(  , print=  Printing Option. Default=Y);
  %put %nrstr(  %));
  %put ;
  %put Example: Calculating N MEDIAN MEAN STD CV MIN MAX and 2.5 97.5 percentile ;
  %put %nrstr(         for all numerical variables of sashelp.class data set);
  %put ------------------------------------------------------------------------;
  %put %nrstr(%unistats %();
  %put %nrstr(  , data=sashelp.class);
  %put %nrstr(  , vars=_numeric_);
  %put %nrstr(  , by=);
  %put %nrstr(  , where=);
  %put %nrstr(  , stats=N MEDIAN MEAN STD CV MIN MAX);
  %put %nrstr(  , pctlpts=2.5 97.5);
  %put %nrstr(  , pctldef=5);
  %put %nrstr(  , out=work.stat1);
  %put %nrstr(  , print=y);
  %put %nrstr(  %);;);

  %goto ende;
%end;

%let opt1=%sysfunc(getoption(notes)); * save current option setting;
option nonotes; * Set options during the macro run;

* Check input parameters;
%if "&stats"="" %then %let stats = N MEDIAN MEAN STD CV MIN MAX ;
%if "%upcase(&stats)"="_ALL_" %then %let stats = &stdstats;

%if "&data"="" %then %let data = &syslast;

%if "&vars"="" %then %let vars = _numeric_;

%if "&out"=""  %then %let out = work.stat1;

%let PCTLDEF=%upcase(&PCTLDEF);
```

```
%if not ( (&PCTLDEF=1) or (&PCTLDEF=2) or (&PCTLDEF=3) or (&PCTLDEF=4) or (&PCTLDEF=5) or (&PCTLDEF=EXCEL) ) %then %do;
  %put ERROR: Invalid definition for computing percentiles! Valid is: 1, 2, 3, 4, 5, Excel;
  %put ERROR: PCTLDEF=&PCTLDEF;
  %let unistats_rc=1;
  %goto ende;
%end;

data _null_;
  length _msg_ $ 100;
  if not (exist("&data")) then do;
    _msg_=sysmsg();
    put "ERROR: Input data set &data not found";
    put _msg_;
    call symput ('unistats_rc', "1");
  end;
run;
%if (&unistats_rc=1) %then %goto ende;

* Check existence and type of the variables (1=numeric) ;
data _null_;
  set &data (obs=1 keep= &vars);
run;
%if (&SYSERR ne 0) %then %do;
  %put ERROR: Not all specified analysis variables exist in data set &data !;
  %put ERROR: Specified analysis variable(s)=&vars;
  %let unistats_rc=1;
  %goto ende;
%end;

data _null_;
  set &data (obs=1 keep= &by);
run;
%if (&SYSERR ne 0) %then %do;
  %put ERROR: Not all specified By-variables exist in data set &data !;
  %put ERROR: Specified By-variable(s)=&by;
  %let unistats_rc=1;
  %goto ende;
%end;

data _null_;
  set &data (obs=1);
  array _xx_ &vars;
  do i=1 to dim(_xx_);
    _xx_[i]=_xx_[i]*1;
  end;
  if _error_ then do;
    call symput ('unistats_rc', "1");
  end;
  else call symput('nnumvar', left(put(dim(_xx_),3.))); * Number of numeric variables;
run;
%if (&SYSERR ne 0) or (&unistats_rc=1) %then %do;
  %put ERROR: Not all specified analysis variables are numeric!;
  %put ERROR: Specified analysis variable(s)=&vars;
  %let unistats_rc=1;
  %goto ende;
%end;

%if (&pctlpts ne ) %then %do;
  %if (%sysfunc(compress("&PctlPts",".0123456789")) ne ) %then %do;
    %put ERROR: Invalid percentiles specified!;
    %put ERROR: PctlPts=&PctlPts;
    %let unistats_rc=1;
    %goto ende;
  %end;
%end;


* copy of the input data set applying where clause;
%if "&by"^="" %then %do;
  proc sort data=&data out=_unistat_0 (keep=&vars &by);
    where &where;
    by &by;
  run;
  %if (&SYSERR ne 0) %then %do;
    %put ERROR: Invalid WHERE condition!;
    %put ERROR: Specified WHERE condition=&where;
    %let unistats_rc=1;
    %goto ende;
  %end;
%end;
%else %do;
  data _unistat_0;
```

```
    set &data;
    where &where;
    keep &vars;
  run;
  %if (&SYSERR ne 0) %then %do;
    %put ERROR: Invalid WHERE condition!;
    %put ERROR: Specified WHERE condition=&where;
    %let unistats_rc=1;
    %goto ende;
  %end;
%end;


* Label definitions;
%let n       = N ;
%let nobs    = N total ;
%let nmiss   = N missing ;
%let sum     = Sum ;
%let median  = Median ;
%let mean    = Mean ;
%let var     = Variance ;
%let std     = Standard deviation ;
%let cv      = %nrstr(CV % ) ;
%let min     = Min ;
%let max     = Max ;
%let q1      = Lower quartile ;
%let q3      = Upper quartile ;
%let p1      = %nrstr(Percentile 1% ) ;
%let p5      = %nrstr(Percentile 5% ) ;
%let p10     = %nrstr(Percentile 10% ) ;
%let p90     = %nrstr(Percentile 90% ) ;
%let p95     = %nrstr(Percentile 95% ) ;
%let p99     = %nrstr(Percentile 99% ) ;
%let normal  = Normality ;
%let normaltest= Normality ; * new in 2.0;
%let probn   = Prob Norm ;
%let t       = T ;
%let probt   = Prob >|T| ;
%let msign   = M(Sign) ;
%let probm   = Prob >=|M| ;
%let signrank= Sgn Rank ;
%let probs   = Prob >=|S| ;
%let stdmean = STDMEAN ;
%let uss     = uncorr. SSQ ;
%let css     = corr. SSQ ;
%let skewness= Skewness ;
%let kurtosis= Kurtosis ;
%let sumwgt  = Sum of weights ;
%let range   = Range ;
%let qrange  = Q3-Q1 ;
%let mode    = Mode ;
%let GINI      = GINI ; * new in 2.0;
%let MAD       = MAD ; * new in 2.0;
%let QN        = Qn ; * new in 2.0;
%let SN        = Sn ; * new in 2.0;
%let STD_GINI  = Std GINI ; * new in 2.0;
%let STD_MAD   = Std MAD ; * new in 2.0;
%let STD_QN    = Std Qn ; * new in 2.0;
%let STD_QRANGE= Std QRANGE ; * new in 2.0;
%let STD_SN    = Std Sn ; * new in 2.0;

* Parse variables list if it contains special lists (x1-x10, a--z, abc:, _numeric_);
%let by   = %upcase(&by);
%if %index(%quote(&vars),-) > 0 or %index(%quote(&vars),:) > 0 or "%upcase(&vars)"="_NUMERIC_" %then %do;

  data _null_;
    set _unistat_0 (obs=1);
    array _xx_ &vars;
    call symput('nnumvar', left(put(dim(_xx_),3.))); * Number of numeric variables;
  run;

  * write up to 992 variable names into one character variable _VL# (993*(32+1) > 32767 characters);
  data _null_;
    length _vname_ $32;
    set _unistat_0 (obs=1);
    array _xx_ &vars;
    array _vl [%sysfunc(ceil((&nnumvar-0.5)/992))] $32767.;
    do i=1 to &nnumvar;
      call vname(_xx_[i],_vname_);
      _vl[ceil((i-0.5)/992)]=trim(_vl[ceil((i-0.5)/992)])||' '||trim(_vname_);
    end;
    do i=1 to dim(_vl);
```

```
      hilf="V"||left(put(i, 2.)); * for makro variable V1, V2, ...;
      call symput (hilf, compbl(_vl[i]));
    end;
  run;

  %let vars=;
  %do i=1 %to %sysfunc(ceil((&nnumvar-0.5)/992)); * add all makro variables;
    %let vars=&vars &&V&i;
  %end;
%end;

* Create sequence of the variables according to their definiton in VARS (for correct labels) (Version 2.1);
data _unistat_0;
  retain &vars;
  set _unistat_0;
run;


* Special case PCTLDEF=EXCEL: Plausibilty check:
  Only one analysis variable
  No By-Variable(s)
  No standard percentiles (P1 P5 P10 Q1 MEDIAN Q3 P90 P95 P99 QRANGE) because they can not be calculated according to Microsoft Exc
;
%if (&PCTLDEF=EXCEL) %then %do;
  %if (%qscan(&vars,2) ne ) %then %do;
    %put ERROR: For PCTLDEF=EXCEL only one analysis variable is allowed!;
    %put ERROR: Specified analysis variable(s)=&vars;
    %put ERROR: Recall the macro for each analysis variable instead;
    %let unistats_rc=1;
    %goto ende;
  %end;
  %if (&by ne ) %then %do;
    %put ERROR: For PCTLDEF=EXCEL no By-processing is possible!;
    %put ERROR: Recall the macro for each By-group with an apropriate Where condition instead;
    %put ERROR: Specified By-variable(s)=&by;
    %let unistats_rc=1;
    %goto ende;
  %end;

  %let i = 1 ;
  %let stat = %qscan (&stats,&i) ;
  %do %while ( &stat ^= ) ;
    %let stat = %upcase(&stat);
    %if (&stat=P1) or (&stat=P5) or (&stat=P10) or (&stat=Q1) or (&stat=MEDIAN) or (&stat=Q3) or
        (&stat=P90) or (&stat=P95) or (&stat=P99) or (&stat=QRANGE) %then %do;
      %put ERROR: Invalid statistic for PCTLDEF=EXCEL: &stat;
      %let h1=Fehler;
    %end;
    %let i = %eval(&i + 1) ;
    %let stat = %qscan(&stats,&i) ;
  %end ;
  %if (&h1=Fehler) %then %do;
    %put ERROR: For PCTLDEF=EXCEL no standard percentile (P1 P5 P10 Q1 MEDIAN Q3 P90 P95 P99 QRANGE) is allowed,;
    %put ERROR: because they will not be calculated according to Microsoft Excel !;
    %put ERROR: Specify those percentiles with PCTLPTS instead e.g. instead of P10 Q1 MEDIAN use PCTLPTS=10 25 50;
    %let unistats_rc=1;
    %goto ende;
  %end;
%end;

* Show input parameter in the log;
  %put %nrstr(%unistats %();
  %put %str(  , data=&data);
  %put %str(  , vars=&vars);
  %put %str(  , by=&by);
  %put %str(  , where=&where);
  %put %str(  , stats=&stats);
  %put %str(  , pctlpts=&pctlpts);
  %put %str(  , pctldef=&pctldef);
  %put %str(  , out=&out);
  %put %str(  , print=&print);
  %put %nrstr(  %));


%if (&PCTLDEF=EXCEL) %then %do;
* Calculate percentiles according to MS Excel function QUANTIL(Matrix, Alpha)
  1. Recalculate the percentile points according to Pmod=100*(P*(n-1)/(n+1) + 1/(n+1))
     with P=given percentile, Pmod=modified percentile, n=number of data values
  2. Calculate the modified percentiles with PCTLDEF=4
  3. Rename and label the result variables according to the PCTLPTS definition
;
```

```
  * Save number of values in macro variable;
  proc sql noprint;
    select n(&vars) into :h1
      from _unistat_0;
  quit;
  %let h1=&h1; * reduce blancs;

  %* Recalculate the percentile points ;
  %let pctlpts2=;
  %let j=1 ;
  %let stat = %qscan ( &pctlpts , &j, " " ) ;
  %do %while ( &stat ^= ) ;
    %let h2=%sysevalf(100*(&stat/100*(&h1-1)/(&h1+1) + 1/(&h1+1)));
    %let pctlpts2=&pctlpts2 &h2;
    %let j = %eval ( &j + 1 ) ;
    %let stat = %qscan ( &pctlpts , &j, " " ) ;
  %end ;
  %let npctlpts = %eval ( &j - 1 ) ;
/*  %put pctlpts =#&pctlpts#;*/
/*  %put pctlpts2=#&pctlpts2#;*/

  proc univariate data=_unistat_0  noprint PCTLDEF=4;
    var &vars ;
    output out=_unistat_2 pctlpts=&pctlpts2 pctlpre=PEXC_;
  run;

  * Save list of generated variable names in macro variable;
  proc contents data=_unistat_2 noprint out=_unistat_3;
  run;

  proc sql noprint;
    select name into :pctlpts2 separated by ' '
      from _unistat_3
      where substr(name,1,5)="PEXC_"
      order by VARNUM
    ;
  quit;

  * generierte Variablennamen umbenennen und labeln entsprechend der gewünschten PCTLPTS Definition;
  data _unistat_2;
    set _unistat_2;

    %do i=1 %to &npctlpts;
      %let h1=%scan(&pctlpts2, &i," ");
      %let h2=%sysfunc(translate(%scan(&pctlpts, &i," "),"_","."));
      rename &h1=P_&h2;
      label  &h1="Percentile %scan(&pctlpts, &i," ")%";
    %end;
  run;

%end; %* Calculate percentiles according to MS Excel;


  proc univariate data=_unistat_0 normal noprint
  %if NOT (&PCTLDEF=EXCEL) %then %do;
    PCTLDEF=&PCTLDEF
  %end;
    ;
   var &vars ;
   output out = _unistat_1

    %let i = 1 ; %* Statistics;
    %let stat = %qscan ( &stats , &i ) ;
    %do %while ( &stat ^= ) ;

       &stat=

       %let i = %sysfunc ( putn ( &i , z2. ) ) ;
       %let j = 1 ;  %* Variables;
       %let hvar = %qscan ( &vars , &j ) ;
       %do %while ( &hvar ^= ) ;
           %let j = %sysfunc ( putn ( &j , z5. ) ) ;
           v&j._s&i.
           %let j = %eval ( &j + 1 ) ;
           %let hvar = %qscan ( &vars , &j ) ;
       %end ;
       %let nvars = %eval ( &j - 1 ) ;
       %let i = %eval ( &i + 1 ) ;
       %let stat = %qscan ( &stats , &i ) ;
    %end ;
    %let nstats = %eval ( &i - 1 ) ;
```

```
    %if (&pctlpts ne ) and not (&PCTLDEF=EXCEL) %then %do; %* Define percentile points;

      pctlpts=&pctlpts

      %* Perzentilpunkte zählen;
      %let i = 1 ;
      %let stat = %qscan ( &pctlpts , &i, " " ) ;
      %do %while ( &stat ^= ) ;
         %let i = %eval ( &i + 1 ) ;
         %let stat = %qscan ( &pctlpts , &i, " " ) ;
      %end ;
      %let npctlpts = %eval ( &i - 1 ) ;

      %* Define percentile prefix for each analysis variable;
      pctlpre=
      %let i = 1 ;
      %let hvar = %qscan ( &vars , &i ) ;
      %do %while ( &hvar ^= ) ;
         %let i = %sysfunc ( putn ( &i , z5. ) ) ;
         v&i._P
         %let i = %eval ( &i + 1 ) ;
         %let hvar = %qscan ( &vars , &i ) ;
      %end ;

    %end;
    %else %let npctlpts=0;

        ;
   by &by;
run ;
%if &syserr=1 %then %do;
  %let unistats_rc=2;
  %goto ende;
%end;

* Save labels of all analysis variables in a list for the output;
proc contents data=_unistat_0 (keep=&vars) noprint out=_unistat_3;
run;
proc sql noprint;
  select quote(trim(label)) into :pctlpts2 separated by '@'
    from _unistat_3
    order by VARNUM
  ;
quit;
/*%put pctlpts2=#&pctlpts2#;*/

data &out;
  length VName $32 VLabel $256 _PctlDef_ $5;
  set _unistat_1;
  keep &by vname vlabel _PctlDef_ &stats
    %if (&pctlpts ne ) and not (&PCTLDEF=EXCEL) %then  %do;
      P_:
    %end;
  ;

  _PctlDef_="&pctldef";

  %do i=1 %to &nvars;
    vname="%scan(&vars, &i)";
    vlabel=%scan(&pctlpts2, &i,'@');
    %do j=1 %to &nstats;
      %let stat=%scan(&stats, &j);
      &stat=v%sysfunc(putn(&i,z5.))_s%sysfunc(putn(&j,z2.));
    %end;
    %do j=1 %to &npctlpts;
      %let stat=%sysfunc(translate(%scan(&pctlpts, &j," "),"_","."));
      P_&stat=v%sysfunc(putn(&i,z5.))_P&stat;
    %end;
    output;
  %end;

  label vname=Name vlabel=Label _PctlDef_="PctlDef"
    %do i = 1 %to &nstats ;
      %let stat = %upcase ( %qscan ( &stats , &i ) ) ;
      %if %index ( &stdstats , &stat ) %then  &stat = &&&stat ;
    %end ;
    %do i=1 %to &npctlpts;
      %let stat=%sysfunc(translate(%scan(&pctlpts, &i," "),"_","."));
      P_&stat="Percentile %scan(&pctlpts, &i," ")%"  %* variables Label definieren;
    %end;
  ;
run;
```

```
%if (&PCTLDEF=EXCEL) %then %do; * Merge the separately calculated percentiles;
  data &out;
    set &out;
    set _unistat_2;
  run;
%end;

* delete temporary data sets;
proc datasets lib = work nolist ;
  delete _unistat_: ;
quit;

%if "%upcase(&print)" = "Y" %then %do;
  proc print data=&out label uniform noobs;
    by &by;
  run ;
%end;

%ende:
options &opt1; * Reset options;

%if (&debug=1) %then %do;
  * Create list of all macro variables to find undefined macro variables;
  proc sort data=sashelp.vmacro out=mVarsEnd;
    where scope ne 'AUTOMATIC';
    by scope name;
  run;

  * Compare both lists of all macro variables to find undefined macro variables;
  data _null_;
    length n 8. VarList $2000. ;
    retain n 0  VarList;
    merge mVarsEnd (in=_end keep=scope name value) mVarsBegin (in=_begin keep=scope name) end=eof ;
    by scope name;
    if _end and not _begin then do;
      if n=0 then put 'WARNING: >>> Undefined macro variable(s):';
      put scope= @30 name @60 value;
      n = n + 1;
      VarList = catx(' ', VarList, name);
    end;
    if eof then do;
      if n=0 then put '>>> No undefined macro variables found!';
      else put VarList;
    end;
  run;
%end;

/*%put UNISTATS_RC=&UNISTATS_RC;*/
%put ********** End Macro &SYSMACRONAME Version &mcrVersion **********;
%mend unistats ;
```