

bases\_sas\_tips\_may10.sas

```
/*  
Base SAS Tips and Tricks for the Victoria and Vancouver User Groups in May 2010
```

```
Rob Wilson  
*/
```

```
*=====;  
/*
```

```
Colouring your log:
```

```
*/
```

```
*=====;  
/*
```

```
Some sorting techniques
```

- A general rule-of-thumb for calculating sort space required is three times the space of the dataset.

- You can simply set Compress=yes on a dataset and then sort it. This can result in time savings.

- The Tagsort option is very good on large datasets where key is small. In one example I saved 49% of the elapsed time. When the sort key approaches the size of the observation, the effectiveness of this technique decreases and can eventually increase time taken.

- You should almost always use the Noequals option on PROC SORT. All that this option does is tells SAS not to worry about the order of observations within BY groups. EQUALS causes SAS to keep that order the same as the observations were in the input data. An extra overhead that is virtually never required.

```
*/
```

```
*=====;
```

```
/*
```

```
Finding secret SAS options
```

You can find out the settings of all documented and undocumented SAS options by using the following code: \*/

```
proc options internal;  
run;
```

```
*=====;  
/*
```

```
Mixed numeric informats
```

Unquoted numerics in the informat definition are treated as numbers. Quoted text is treated as character. This can be quite useful if reading data which has mixed values, which need to be interpreted in different ways.\*/

```
proc format ;  
  invalue mixed  
    'LOW' = -99  
    1-10 = 1
```

bases\_sas\_tips\_may10.sas

```
11-20 = 2  
'BIG' = 99  
other = 0 ;
```

run ;

```
data sample;  
  informat range mixed.;  
  input range;
```

DATALINES;

LOW

1

5

11

50

BIG

;

\*=====;

/\* Easter Calculation;

\* This works for all dates from 1583 (or first Gregorian date) through to 4099;  
\* translated to SAS from the Basic program developed by R W Mallen of the  
Astronomical Society of South Australia;  
\* and published at <http://www.assa.org.au/edm.html#Computer> ;  
Email: [chris@mollingtonconsultants.com](mailto:chris@mollingtonconsultants.com)  
\*/

%macro calc\_Easter(yy=);

%\*options nomprint nomlogic nosymbolgen;

%local firstDig Remain19 temp ta tb tc td te dd mm result;

%let firstDig = %sysfunc(int(&yy / 100)); /\* first two digits of year;

%let remain19 = %sysfunc(mod(&yy, 19)); /\* remainder of year;

%put FirstDig=&firstDig;

%put Remain19=&remain19;

/\* calculate PFM date;

%let temp = %eval(%sysfunc(int((&firstDig. - 15) / 2)) + 202 - 11 \*  
&remain19.);

%put TEMP=&temp;

%let dayarr=%str(21,24,25,27,28,29,30,31,32,34,35,38);

%let day2arr=%str(33,36,37,39,40);

%if %sysfunc(find(&dayarr.,&firstDig.)) %then %do;

%let temp = %eval(&temp. - 1); %put TEMP - 1=&temp;

%end;

%else %if %sysfunc(find(&day2arr.,&firstDig.)) %then %do;

%let temp = %eval(&temp. - 2); %put TEMP - 2=&temp;

%end;

%let temp = %sysfunc(mod(&temp., 30)); %put TEMP mod 30=&temp;

%let tA= %eval(&temp + 21); %put TA=&tA;

%if &temp. = 29 %then %do;

%let tA = %eval(&tA - 1); %put TA (temp=29)=&tA;

%end;

%if &temp. = 28 and &remain19. > 10 %then %do;

%let tA = %eval(&tA - 1); %put TA (temp=28 and remain19 > 10)=&tA;

%end;

/\* Find the next Sunday;

%let tB = %sysfunc(mod(%eval(&tA - 19),7)); %put TB=&tB;

bases\_sas\_tips\_may10.sas

```

%let tC = %sysfunc(mod(%eval(40 - &firstDig),4)); %put TC=&tC;
%if &tC = 3 %then %do;
    %let tC = %eval(&tC + 1); %put TC (TC=3)=&tC;
%end;
%if &tC > 1 %then %do;
    %let tC = %eval(&tC + 1); %put TC (TC > 1)=&tC;
%end;

%let temp = %sysfunc(mod(&yy.,100)); %put TEMP=&temp;
%let tD = %sysfunc(mod((&temp + %sysfunc(int(&temp. / 4))),7)); %put TD=&tD;

%let tE = %eval(%sysfunc(mod(%eval(20 - &tB -&tC - &tD),7)) + 1); %put
TE=&tE;
%let Dd = %eval(&tA + &tE); %let DD=&Dd;

%* Return the date;
%if &Dd > 31 %then %do;
    %let Dd = %eval(&Dd - 31); %put DD=&Dd;
    %let mM = 4; %put MM=&mM;
%end;
%else %do;
    %let mm = 3; %put MM=&mm;
%end;
%let result=&yy.-&mm.-&dd;
&result
%mend;

%let easter=%calc_easter(yy=2016);
%put EASTER=&easter.;

/*****
****/
/* This sample code is based upon an algorithm described on the U.S. Naval
Observatory */
/* website: http://aa.usno.navy.mil/faq/docs/easter.htm
*/
/*
*/
/* This sample provided 'as is', no warranties expressed or implied.
*/
/*****
****/

data a (keep=easter_date);
    format easter_date worddate.;
    do year=1980 to 2024;
        c=int(year/100);
        n=year-(19*int(year/19));
        k=int((c-17)/25);
        i=c-int(c/4)-int((c-k)/3)+(19*n)+15;
        i=i-(30*int(i/30));
        i=i-(int(i/28)*(1-int(i/28)*int(29/(i+1))*int((21-n)/11)));
        j=year+int(year/4)+i+2-c+int(c/4);
        j=j-(7*int(j/7));
        month=3+int((i-j+40)/44);
        day=i-j+28-(31*int(month/4));
        easter_date=mdy(month,day,year);
    output;
    end;

run;

```

bases\_sas\_tips\_may10.sas

```
proc print;  
run;
```

```
*=====;
```

```
/* Proc Calendar  
Create a calendar */
```

```
data year;  
input sta : date7. act $ 11-30 dur;  
datalines;  
01JAN10 Start 0  
31DEC10 Finish 0  
;
```

```
data holidays;  
input sta : date7. act $ 11-30 dur;  
datalines;  
01JAN10 New Year's 1  
11MAR10 Robs B-day 1  
25DEC10 Christmas Break 5  
;
```

```
options nodate pageno=1 linesize=132 pagesize=30;
```

```
proc calendar data=year holidata=holidays fill interval=workday;
```

```
start sta;  
dur dur;
```

```
holistart sta;  
holivar act;  
holidur dur;
```

```
title1 'Calendar of Holidays Only';
```

```
run;
```

```
/* Create a Calendar Using Proc Report (Higher Quality) vs Proc Calendar */
```

```
%macro calendar_report (month, year);
```

```
data grid;  
date1 = mdy (&month,1,&year);  
date2 = intnx ('month', date1, 1) - 1;  
do date = date1 to date2;  
wim = intck ('week', date1, date);  
dim = date-date1+1;  
dow = weekday (date);  
output;  
end;
```

```
run;
```

```
proc format;  
value dayname  
1 = 'Sunday'  
2 = 'Monday'  
3 = 'Tuesday'  
4 = 'Wednesday'  
5 = 'Thursday'  
6 = 'Friday'  
7 = 'Saturday'
```

```

;
run;
options missing=' ';
proc report nowindows data=grid
  style (column) = { just=center cellheight=50pt };
  column wim ("%sysfunc(mdy(&month,1,&year), monname) &year" dow, dim) ;
  define wim / group noprint ;
  define dow / across ' ' format=dayname. order=internal;
  define dim / ' ';
run;
options missing='.';
%mend;

ods html file='c:\temp\calendar.html';
  %calendar_report (1,2010);

data caldat;
  date=mdy(5,04,2010);
  subject='Victoria';
  visit='VUG';
  duration=1;
  output;

run;

proc calendar data=caldat formchar(12)=' ';
  start date;
  dur duration;
run;
ods html close;

*=====;
/*
Sample 25074: Listing all files that are located in a specific directory
-----*/

%macro drive(dir,ext);

  %let filrf=mydir;

  /* Assigns the fileref of mydir to the directory and opens the directory */
  %let rc=%sysfunc(filename(filrf,&dir));
  %let did=%sysfunc(dopen(&filrf));

  /* Returns the number of members in the directory */
  %let memcnt=%sysfunc(dnum(&did));

```

bases\_sas\_tips\_may10.sas

```
/* Loops through entire directory */
%do i = 1 %to &memcnt;

  /* Returns the extension from each file */

  %let name=%qscan(%qsysfunc(dread(&did,&i)),-1,.);

  /* Checks to see if file contains an extension */
  %if %qupcase(%qsysfunc(dread(&did,&i))) ne %qupcase(&ext) %then %do;

    /* Checks to see if the extension matches the parameter value */
    /* If condition is true prints the full name to the log */
    %if (%superq(ext) ne and %qupcase(&name) = %qupcase(&ext)) or
        (%superq(ext) = and %superq(name) ne) %then
      %put %qsysfunc(dread(&did,&i));
    %end;
  %end;

  /* Closes the directory */
  %let rc=%sysfunc(dclose(&did));

%mend drive;

/* First parameter is the directory of where your files are stored. */
/* Second parameter is the extension you are looking for. */
/* Leave 2nd paramater blank if you want a list of all the files. */
%drive("C:\",xls);

*=====;
/* Perl Regular Expressions Tips */
*Basic Example;
data _null_;
  pos=prxmatch('/world/','Hello world!');
```

bases\_sas\_tips\_may10.sas

```
put pos=;

txt=prxchange('s/world/planet/',-1, 'Hello world!');

put txt=;
run;

*Output:
*      pos=7;
*      txt=Hello planet;

*Data Validation;

data phone_numbers;
  length first last phone $ 16;
  input first last phone & $16.;
datalines;
Thomas Archer (919)319-1677
Lucy Barr 800-899-2164
Tom Joad (508) 852-2146
Laurie Gil (252)152-7583
;

data invalid;
  set phone_numbers;
  where not prxmatch("/\([2-9]\d\d\) ?" || "[2-9]\d\d-\d\d\d\d/",phone);
run;
proc sql; /* Same as prior data step */
  create table invalid2 as
  select * from phone_numbers
  where not prxmatch("/\([2-9]\d\d\) ?" || "[2-9]\d\d-\d\d\d\d/",phone);
quit;

*Output:
*Obs first last phone
*1 Lucy Barr 800-899-2164
*2 Laurie Gil (252)152-7583

*Search and Replace #1;
data _null_;
  input;
  _infile_ = prxchange('s/ </&lt;/', -1, _infile_);
  put _infile_;
datalines;
x + y < 15
x < 10 < y
y < 11
;

/*
*Output:
*x + y &lt; 15;
*x &lt; 10 &lt; y;
*y &lt; 11;
*/

*Search and Replace #2;
data reversed_names;
  input name & $32.;
datalines;
Jones, Fred
Kavich, Kate
```

Turley, Ron  
 Dulix, Yolanda  
 ;

```
data names;
  set reversed_names;
  name = prxchange('s/(\w+), (\w+)/$2 $1/', -1, name);
run;
proc sql; /* Same as prior data step */
  create table names2 as
  select prxchange('s/(\w+), (\w+)/$2 $1/', -1, name)
  as name
  from reversed_names;
quit;
```

```
*Output;;
*Obs name;
*1 Fred Jones;
*2 Kate Kavich;
*3 Ron Turley;
*4 Yolanda Dulix;
```

```
*Search and Extract;
data _null_;
  length first last phone $ 16;
  retain re;
  if _N_ = 1 then
    do;
      re = prxparse("/\(([2-9]\d\d)\) ?" || "[2-9]\d\d-\d\d\d\d/");
    end;

  input first last phone & $16.;

  if prxmatch(re, phone) then
    do;
      area_code = prxposn(re, 1, phone);
      if area_code ^in ("828" "336" "704" "910" "919" "252")
        then putlog "NOTE: Not in NC: "
          first last phone;
    end;
  datalines;
Thomas Archer (919)319-1677
Lucy Barr (800)899-2164
Tom Joad (508) 852-2146
Laurie Gil (252)352-7583
;
```

```
*Output;;
*NOTE: Not in NC, Lucy Barr (800)899-2164;
*NOTE: Not in NC, Tom Joad (508) 852-2146;
```

```
*=====;
/* Determine what States are adjacent to each other;
```

```
Solution by
Mike Zdeb
U@Albany School of Public Health
1 University Drive
Rensselaer, NY 12144-3456
(P)518-402-6479 */
```

```
*** merge a SAS/GRAPH map data set with itself;
Page 8
```

```

bases_sas_tips_may10.sas
*** find common vertices (lat/long locations with same values);

proc sql;
create table borders as
select distinct d0.state as state0,
               d1.state as state1
from maps.states d0, maps.states d1
where (d0.x eq d1.x and d0.y eq d1.y);
quit;

*** extra data step to eliminate state bordering itself;
*** also adds state names to the data set;

data borders;
set borders;
by state0;
if state0 eq state1 then state1 = .;
if not (first.state0 and last.state0) and state1 eq . then delete;
sname0 = fipname(state0);
sname1 = fipname(state1);
run;

proc print data=borders;
var state1 sname1;
by state0 sname0;
run;

*** rearrange data to create a matrix of bordering states;

proc transpose data=borders out=new (drop=_) prefix=state;
by state0;
run;

proc print data=new;
run;

*** alternative --- eliminates states bordering itself, but also
*** eliminates states that border no other states (ALASKA, HAWAII, PUERTO RICO);
*** merge a SAS/GRAPH map data set with itself;
*** find common vertices (lat/long locations with same values);

proc sql;
create table borders as
select distinct d0.state as state0,
               d1.state as state1
from maps.states d0, maps.states d1
where (d0.x eq d1.x and d0.y eq d1.y) and state0 ne state1;
quit;

*=====;

/****
Example of Getting User Input Interactively
Author: Charles Patridge
Demonstration of Simple Use of using WINDOW statement to
solicit input from User and Process the input

Assumption is made you only have BASE SAS available

You should have your program window in MAXMIUM WINDOW size before
executing this.

```

bases\_sas\_tips\_may10.sas

This is a simple example of WINDOWS and has a couple of macros embedded to display some data values - however, this is not a realistic example as your data could easily fill up more than 1 screen to display such values -hence not very useable.

This was intended to show how to use windows to gather INPUT.

You should read up on the window statement to get more info about the parameters and how they would.

\*\*\*/  
/

/\*\*

Example of Getting User Input Interactively

\*\*\*/  
/

```
Data _null_;  
  window start  
    #5 @26 'welcome to the SAS system' color=red  
    #7 @26 'This Program is for Demonstration' color=black  
    #9 @26 'Press the Enter Key to Continue' color=blue  
  ;  
  display start;  
  stop;  
run;
```

/\*\* end of simple display \*\*\*/  
/

\*=====;

/\* Parsing/extracting multiple email addresses from a text field

Examples of the data have been entered include:

```
Name@MSN.COM Name@NC.RR.COM  
Name@NCOL.NET...Name@ncol.net  
name@msn.com, name@msn.com  
name@aol.com//name2022@nc.rr.com  
name@peoplepc.com/name.1@netzero.net  
name@BASF-CORP.COM or name20@msn.com  
name's@cs.com name-name.place@btitelecom.net
```

/\*\*

use the translate / tranwrđ functions to ensure the delimiting character is space, then use scan to rip it apart. Some clean up remains but this is enough to get you going.\*\*\*/

```
data addresses (drop = _: );  
  length email_addr $100;  
  _a = "Name@MSN.COM Name@NC.RR.COM Name@NCOL.NET...Name@ncol.net  
name@msn.com, name@msn.com name@aol.com//name2022@nc.rr.com  
name@peoplepc.com/name.1@netzero.net name@BASF-CORP.COM or name20@msn.com  
name's@cs.com name-name.place@btitelecom.net Name@NCOL.NET....Name@ncol.net  
Name@xoryorz.com";  
  
  _a = translate(_a, ' ', '/', ',');  
  
  _a = tranwrđ(_a, ' or ', ', '); /*** added by Howard Schreier ***/  
/
```

Notice the blanks surrounding the word "or"; important since "or" alone could occur within an address.\*\*\*/

bases\_sas\_tips\_may10.sas

```
*** _a = compbl(tranwrld(_a,'..',' ')); /**** replaced with Howard Schreier
suggestion ****/
```

```
_a = tranwrld(compbl(tranwrld(_a,'..',' ')),'.',' '); /**** added by Howard
Schreier ****/
/****
```

The multiple dots are tricky, since of course dots appear within addresses. I would wrap another TRANWRD invocation around Harry formula.\*\*\*\*/

```
put _a=;
do until(0);
  _i + 1;
  email_addr = scan(_a,_i,' ');
  if missing(email_addr) then leave;
  if index(email_addr,'@') then output;
end;
run;
```

```
proc print;
run;
```

```
*=====;
```

```
/* How to Find Each Wednesday (1st, 2nd, 3rd, 4th and 5th) of the Month ***/
```

```
data test;
  do i = 1 to 12;
    date = mdy( i, 01, 2010); /**** first day of of the month ****/
    firstwed=intnx('week.4',date,0); /**** maybe 1st wednesday of the Month ****/

    if firstwed lt date then firstwed = firstwed + 7; /**** make 1st wednesday of
the month just in case ****/
    secondwed = firstwed + 7 ; /**** 2nd wednesday of month ****/
    thirdwed = firstwed + 14; /**** 3rd wednesday of month ****/
    fourthwed = firstwed + 21; /**** 4th wednesday of month ****/
    fifthwed = firstwed + 28; /**** 5th wednesday of month ****/

    if fifthwed ge mdy(i+1,01,year(date)) then fifthwed = .; /**** make missing in
case there is no 5th wed ****/

    put (firstwed--fifthwed) (weekdate.)
       date weekdate.;

  end;
run;
```

```
/* How to find just the 3rd Wednesday of the Month */
```

```
data test (drop=i);
  do i = 1 to 12;
    date=mdy(i,01,2003);
    if weekday(date)=4 then thirdwed=intnx('week.4',date,2);
    else thirdwed=intnx('week.4',date,3);
    output;
  end;
  format date weekdate.
         thirdwed weekdate.;
run;
```

```
/* Use the date of the first of the following month and apply this:
lastfri=intnx('week.6',date-1,0); /**** last Friday of the Month ****/
```

bases\_sas\_tips\_may10.sas

```
data test;
  do i = 1 to 12;
    date = mdy( i, 01, 2002); /**** first day of following month ****/
    lastfri=intnx('week.6',date-1,0); /**** last Friday of the Month ****/
    put lastfri weekdate. date weekdate.;
  end;
run;
```

=====;

/\* How to determine the number of variables in a data set?

You can programmatically determine the number of variables in a data set by using the following code:\*/

```
data _null_;
  set sashelp.vtable (where=(libname='WORK' and memname='TEST'));
  call symput('nvar',nvar);
run;
%put &nvar;
```

\*----- OR -----;

```
proc sql noprint;
  select nvar
  into   :nvar2
  from   dictionary.tables
  where  libname='WORK'
         and
         memname='TEST';
```

```
quit;
%put &nvar2;
```

\*----- OR -----;

```
%let nvar3=%sysfunc(attrn(%sysfunc(open(work.test,i)),nvars));
%put &nvar3;
```

=====;

/\* How to determine the Character Set on a specific Platform. \*/

```
data _null_;
  file print;
  do a=1 to 255;
    b=byte(a);
    put @1 a 3. +1 '=' +1 b;
  end;
run;
```

=====;

/\* SAS provides a variety of State & ZIP functions that convert ZIP codes to state names and state postal codes.

The sample code below illustrates three State & ZIP functions. \*/

```
data a;
x=zipname('27511');
put x=;
y=zipname1('27511');
```

bases\_sas\_tips\_may10.sas

```
put y=;
z=zipstate('27511');
put z=; run;
```

```
/* Results of PUT statements written to the log.
x=NORTH CAROLINA
y=North Carolina
z=NC
```

The ZIPNAME, ZIPNAMEL, and ZIPSTATE functions take the same argument but return different values:

ZIPNAME returns the uppercase name of the state or U.S. territory that corresponds to its five-character ZIP code argument.

ZIPNAMEL returns the upper- and lowercase name of the state or U.S. territory that corresponds to its five-character ZIP code argument.

ZIPSTATE returns the uppercase two-character state postal code (or world-wide GSA geographic code for U.S. territories) that corresponds to its five-character ZIP code argument.

Other State and ZIP code functions provided by SAS include:

FIPNAME which converts FIPS codes to state names in uppercase

FIPNAMEL which converts FIPS codes to state names in uppercase and lowercase

FIPSTATE which converts FIPS codes to two-character postal codes

STFIPS which converts state postal codes to FIPS state codes

STNAME which converts state postal codes to state names (all uppercase)

STNAMEL which converts state postal codes to state names in uppercase and lowercase

ZIPFIPS which converts ZIP codes to FIPS state codes

```
*=====;
```

```
/* How to add sign (+/-) when outputting a number
```

```
How about a picture format. Something like: */
```

```
proc format;
  picture plusmns
    low-<0 = '0009.99' (prefix="-")
    0      = '0.00'   (noedit)
    0-high = '0009.99' (prefix="+")
;
run;
```

```
data _null_;
  do i = -5 to 5;
    put i=plusmns.;
  end;
run;
```

```
*=====;
```

```
*=====;
```

```
/*Program 1.6: Alternative program to capitalize the first letter of each
word in a string
```

```
***First and last name are two separate variables.*/
```

```
DATA PROPER;
  INFORMAT FIRST LAST $30.;
  INPUT FIRST LAST;
```

bases\_sas\_tips\_may10.sas

```
LENGTH NAME $ 60;
CALL CATX(' ', NAME, FIRST, LAST);
NAME = PROPCASE(NAME);
DATALINES;
ronald cODy
THomaS eDISON
albert einstein
;
PROC PRINT DATA=PROPER NOOBS;
TITLE "Listing of Data Set PROPER";
RUN;
```

\*=====;

```
/*Program 1.18: Reading dates in a mixture of formats */
***Primary function: INDEXC
***Other function: INPUT;
```

\*\*\*Note: version 9 has some enhanced date reading ability;

```
***Program to read mixed dates;
DATA MIXED_DATES;
INPUT @1 DUMMY $15.;
IF INDEXC(DUMMY,'/-:') NE 0 THEN DATE = INPUT(DUMMY,MMDDYY10.);
ELSE DATE = INPUT(DUMMY,DATE9.);
FORMAT DATE WORDDATE.;
DROP DUMMY;
DATALINES;
10/21/1946
06JUN2002
5-10-1950
7:9:57
;
PROC PRINT DATA=MIXED_DATES NOOBS;
TITLE " ";
VAR DATE;
RUN;
```

\*=====;

```
/* Dynamically reorder the variables in a SAS Data set to be in alphabetical order.
*/
```

```
/* Create sample data */
```

```
data a;
input x a j z$ q $ ;
datalines;
1 2 3 qwe asd
;
proc sql noprint;
select distinct name
into : varlist separated by ' '
from dictionary.columns
where libname='WORK' and memname='A';
quit;
```

```
/* write macro variable value to the log to see the results */
```

```
%put &varlist;
```

bases\_sas\_tips\_may10.sas

```
*=====;
/*Program 8.2: Using the MOD function to choose every nth observation
              from a SAS data set
***Primary function: MOD
***Other functions: INT, RANUNI;*/

***Create test input data set;
DATA BIG;
  DO SUBJ = 1 TO 20;
    X = INT(10*RANUNI(0)); /* Random integers from 0 to 9 */
    OUTPUT;
  END;
RUN;

%LET N = 4; /* Every 4th observation will be selected */

DATA EVERY_N;
  SET BIG;
  IF MOD(_N_ ,&N) = 1;
  /* Selects every nth observation, starting with the 1st */
RUN;

PROC PRINT DATA=EVERY_N NOOBS;
  TITLE "Listing of Data Set EVERY_N";
RUN;

/*Program 8.3: Using the MOD function as a toggle switch, alternating
              group assignments
***Primary function: MOD;
***In this program, we want to assign every other subject into
              group A or group B;*/

DATA SWITCH;
  DO SUBJ = 1 TO 10;
    IF MOD(SUBJ,2) EQ 1 THEN GROUP = 'A';
    ELSE GROUP = 'B';
    OUTPUT;
  END;
RUN;
PROC PRINT DATA=SWITCH NOOBS;
  TITLE "Listing of Data Set SWITCH";
RUN;

*=====;
/* the hat */
data hat;
  do x = -5 to 5 by .5;
    do y = -5 to 5 by .5;
      z = sin(sqrt(y*y + x*x));
      output;
    end;
  end;
proc g3d;
  plot y*x=z;
run;

*=====;
/* proc spell */

filename temp temp ;
```

bases\_sas\_tips\_may10.sas

```
data _null_ ;
  input word: $12. @@ ;
  list ;
  file temp ;
  put word ;
cards ;
let's see if sas spell procedure
can bee used to verify whether tha
seperate words in this, uhm, flie are, uhm,
valid against a standart internal dictionary
;
run;

option nodate nonumber nocenter ;
title ;

proc spell in = temp verify ;
run ;

proc spell in = temp suggest verify ;
run ;

*=====;
/* program vector reads ahead */

data test;
  input var1 var2 $;
  put var1 var2;
cards ;
1 A
2 B
3 C
4 D
5 E
;
run;

data test2;
  set test;

  if eof=0 then
  set test(
    firstobs = 2
    keep      = var2
    rename    = (var2=nextvar2)
  )
  end=eof;

  else nextvar2=' ';

run;

/* merging a dataset with itself */

data test3;
  merge test
  test(
    firstobs = 2
    keep      = var2
    rename    = (var2=nextvar2)
  );

run;
```