

Writing Macro Do Loops with Dates from Then to When

Ronald J. Fehd, Stakana Analytics, Atlanta, GA, USA

Abstract

Description : Dates are handled as numbers with formats in SAS[®] software. The SAS macro language is a text-handling language. Macro %do statements require integers for their start and stop values.

Purpose : This article examines the issues of converting dates into integers for use in macro %do loops. Two macros are provided: a template to modify for reports and a generic calling macro function which contains a macro %do loop which can either call another macro or return a token with elements in the associative array of dates. Example programs are provided which illustrate unit testing and calculations to produce reports for simple and complex date intervals.

Audience : macro programmers

Information : Programs in this paper are available on the web:
Fehd [11, sco.Macro-Loops-With-Dates]

Keywords : do and %do statements; interval incrementing (intnx): intervals and shift-index; month, putn, %yseval, %sysfunc, today, day-of-the-week (weekday), year

Quote : The White Rabbit put on his spectacles. 'Where shall I begin, please your Majesty?' he asked. '[Start] at the beginning,' the King said gravely, 'and go on till you come to the end: then stop.'
— Lewis Carroll, English author and recreational mathematician (1832–1898), Alice's Adventures in Wonderland.

Contents

Start: Introduction	2
Begin: Macros to Handle Dates	6
Macro Date-Report-Zero	7
Macro DateLoop	10
DateLoop Program Listing	11
DateLoop Testing: Simple and Complex Intervals	14
Parameter MacroName	16
End: Parameter MacroText	17
Stop: Summary	18
Bibliography	19

Introduction

Overview

This article examines the issues of obtaining integers from date literals for use in macro %do loops. Examine these examples.

- iterative do loop in data step
 - error in a macro %do loop
 - math or intnx
-

Data Loop

This data step shows an iterative do loop first with integers and then integers resolved from date literals, which are references to numbers.

```
1 DATA _Null_;
2 do date = 0 to 1;
3   put date= 8. date date.;
4   end;
5 put;
6 do date = '3Jan1960'd to '4Jan1960'd;
7   put date= 8. date date.;
8   end;
9 stop;
10 run;
```

```
date=0 01JAN60
date=1 02JAN60
```

```
date=2 03JAN60
date=3 04JAN60
```

Macro Error

This log shows that a macro %do loop cannot accept a reference but must have a value. This is an issue in all artificial languages and is addressed in Contributors [1, www-wiki.Call-by-Reference].

```
1 %DateLoop(start = '1Jan1960'd
2           ,stop = '7Jan1960'd)
ERROR: A character operand was found in the %EVAL function
       or %IF condition where a numeric operand is required.
       The condition was: &start
```

The problems addressed in this article are how to resolve references — date literals — so that a macro %do loop works like a data step loop.

Math or Intnx?

Can we calculate the beginning and ending dates of an interval, easily? Yes, if we remember the adjustments. Using the intnx function is easier.

```
%let today           = %sysvalf('01May2013'd);
%let day_of_week     = %sysfunc(weekday(&today));
%let this_week_begin = %eval(&today - &day_of_week + 1);
%let this_week_end   = %eval(&today - &day_of_week + 7);

this_week_begin: 19476           Sunday, April 28, 2013
today           : 19479           Wednesday, May 1, 2013
this_week_end   : 19482           Saturday, May 4, 2013
```

Information

Overview

Dates are numbers. The mathematical operations associated with dates are addition and subtraction. The macro language handles text. The problem is how to convert text references — a date literal which represents an integer — into an integer to be used as the argument to a macro %do loop.

Data Step

To obtain an integer use the `today` function or a date literal.

```
data _null_;
zero    = 0;
earlier = '21Dec2012'd;
today   = today();
put zero    = zero    weekdate29.;
put earlier= earlier  mmdyy10.  ;
put today  = today   date9.    ;
stop;
run;

zero    =    0 Friday, January 1, 1960
earlier=19348 12/21/2012
today   =19387 29JAN2013
```

Macros

In macros we use several different functions while working with dates.

assignment : use %sysfunc with either of `today` or `mdy` functions, or %sysevalf a date literal

```
%let today = %sysfunc(today());
%let today = %sysfunc(mdy(05,01,2013));
%let today = %sysevalf('01May2013'd);
```

intnx : get the surrounding dates of the interval by using the `intnx` function

```
*syntax: intnx(interval in (day, week, month, quarter, year)
, start-from: an integer of a date
, increment: an integer: negative zero or positive
, alignment in (begin, middle, end, same);
%let interval = week;
%let D_Begin  =%sysfunc(intnx(&interval,&today,0,begin));
%let D_End    =%sysfunc(intnx(&interval,&today,0,end ));
```

Note: in this example the third argument of `intnx` is zero which returns the begin and end of the current time interval. Later usage in calling programs has minus one to get the begin and end of the previous interval.

display : use the `putn` function with different formats to display the integer of a date in a macro variable

```
%put D_Begin: &D_Begin %sysfunc(putn(&D_Begin,mmdyy10. ));
%put today   : &today   %sysfunc(putn(&today   ,weekdate29.));
%put D_End   : &D_End   %sysfunc(putn(&D_End   ,date9.    ));
D_Begin: 19476                04/28/2013
today   : 19479   Wednesday, May 1, 2013
D_End   : 19482                04May2013
```

Fehd [5, sco.Macro-Vars-of-Date-and-Time] contains other functions and formats used to display dates in titles and footnotes.

Associative Array of Dates

Overview

An associative array is a natural language concept. Given, for instance, an abbreviation such as a two-letter U.S. state code, we associate other information with that code, such as the long name of the state, its capital, etc. This table shows a subset of U.S. federal information processing (fips) abbreviations to illustrate the concept.

fips	abbrev	state	capital	city
2	AK	Alaska	Juneau	Fairbanks
6	CA	California	Sacramento	San Francisco
13	GA	Georgia	Atlanta	Savannah

The fips code is an integer in the look-up table from which a program can get the two-letter state abbreviation and the long name.

```
do fips = 1 to 128;  
  Abbrev = fipstate(fips);  
  State = fipname1(fips);
```

Date Information

A SAS date is an index to the associative array of date information.

This table shows the functions and formats of the date look-up table.

name of	day	function	range	format	names
	month			downname	Sun–Sat
				monname	Jan–Dec
day of	week	weekday	1–7		
	month	day	1–31		
	year	juldate	001–366		
week	year	week	0–53		
month	year	month	1–12		
quarter	year	qtr	1–4		
year		year			

Continued on next page.

Info in Dates

This program shows how to extract each element from the associative array of dates.

```

1  /* name: info-in-dates.sas
2  description: provide associative array of values from dates
3  purpose    : show how to convert date integers to text
4              for use in any date-stamped object names
5  sas.help: About SAS Date, Time, and Datetime Values
6              Dictionary of Functions and CALL Routines
7              Working with Dates in the SAS System:
8              Comparing Durations and SAS Date Values
9              Understanding How SAS Handles Dates
10 reference:
11    http://www.sascommunity.org/wiki/Date_datetime_time_stamp
12 *****/
13 %let today = %sysfunc(today());
14 *let today = %sysfunc(mdy(04,1,2013));
15 *let today = %sysevalf('30Apr2013'd);
16
17 %let day   =%sysfunc(putn(&today,downname));
18 %let day3  =%sysfunc(putn(&today,downname3));
19 %let month=%sysfunc(putn(&today,monname));
20 %let mon   =%sysfunc(putn(&today,monname3));
21
22 %let day_month_N =%sysfunc(day(&today));
23 %let d   =%sysfunc(weekday(&today));
24 %let dd  =%sysfunc(putn(%sysfunc(day(&today)),z2));
25 %let ddd=%substr(%sysfunc(juldate7(&today)),5,3);
26
27 %let month_N =%sysfunc(month(&today));
28 %let mm      =%sysfunc(putn(%sysfunc(month(&today)),z2));
29 %let quarter =%sysfunc(qtr (&today));
30 %let year    =%sysfunc(year(&today));
31 %let yearddd =%sysfunc(juldate7(&today));
32 %let year_ddd=%sysfunc(year(&today))_%substr
33                (%sysfunc(juldate7(&today)),5,3);
34
35 %let week   =%sysfunc(week(&today,u));
36 %let week_v=%sysfunc(week(&today,v));
37 %let week_w=%sysfunc(week(&today,w));
38
39 %put today: &today %sysfunc(putn(&today,weekdate29.));
40 %put _global_;
```

The edited log shows the values of the macro variables of date information.

```
GLOBAL TODAY 19549
today: 19549    Wednesday, July 10, 2013

GLOBAL DAY Wednesday
GLOBAL DAY3 Wed

GLOBAL DAY_MONTH_N 10

GLOBAL D 4
GLOBAL DD 10
GLOBAL DDD 191

GLOBAL MON Jul
GLOBAL MONTH July
GLOBAL MONTH_N 7
GLOBAL MM 07

GLOBAL QUARTER 3

GLOBAL WEEK 27
GLOBAL WEEK_V 28
GLOBAL WEEK_W 27

GLOBAL YEAR 2013
GLOBAL YEAR_DDD 2013_191
GLOBAL YEARDDD 2013191
```

These macro variables are calculated in macro DateLoop when using the parameter MacroText.

Macros to Handle Dates

Overview

The following sections show macros Date-Report-Zero and DateLoop. The macro function DateLoop can call a reporting macro like Date-Report-Zero with the parameters Begin and End. DateLoop with the parameter Macro-Text can return a token with references to the elements of the associative array of dates. The programs provided here are basic unit tests and demonstrations. For other programs see Fehd [11, sco.Macro-Loops-With-Dates].

Header

Each program has a common header.

```
/* name: dateloop-demo.sas
description: testing
   purpose: template /*****/
options mprint;
%let today = %sysfunc(today());
```

Test Data

This program is used to generate test data.

```
1 /* name: test-data
2 description: provide data with dates
3 purpose : for use by other demo programs
4 usage:
5 %daterpt0(data = Library.TestData
6           ,var = date
7           ,...);
8 *****/
9 DATA Library.TestData;
10   attrib EntityId length = 4
11          Date length = 8 format = weekdate17.
12          Fact length = 8;
13 do Date = 0 to today();
14   EntityId = int(Date /17);
15   Fact = int(Date**3/19);
16   output;
17 end;
18 stop;
19 run;
```

Why Zero?

Dijkstra [2, utexas-trans.Why-Numbering-Starts-At-Zero] explains why zero is a good place to start: to avoid off-by-one errors.

Calendar

Use this calendar when comparing dates in logs.

2013						
day 1	day 2	3	4	5	6	day 7
Sun	Mon	Tues	Wed	Thurs	Fri	Sat
Mar 31	Apr 1	2	3	4	5	6
7	8	9	10	11	12	13
14	15	16	17	18	19	20
21	22	23	24	25	26	27
28	29	30	May 1			
SGF				2	3	4
5	6	7	8	9	10	11
12	13	14	15	16	17	18
19	20	21	22	23	24	25
26	27	28	29	30	31	

Macro Date-Report-Zero

Overview

This section presents a simple model — a template for a macro that can be customized — with formats and macro functions for assembling ODS filenames.

Model

A reporting macro needs the following parameters.

- data set name
- variable name
- low
- high

This example shows the data set option `where` with a between condition.

```
1 PROC Print data = sashelp.class
2     (where = ( 11 <= age <= 13) );
NOTE: There were 10 observations read from the
      data set SASHELP.CLASS.
      WHERE (age>=11 and age<=13);
4 PROC SQL; select *
5     from sashelp.class
6     where age between 11 and 13;
7     quit;
```

Continued on next page.

Program Date Report Zero

This simple macro shows the basics of a reporting macro subroutine.

- ODS
- data set name
- variable
- date: begin
- date: end

```
----- daterpt0.sas -----
1  /*   name: DateRpt0.sas
2  description: report with date-begin and -end
3  purpose: template
4  /*****/
5  %MACRO daterpt0
6      (data      = sashelp.class
7      ,var       = age
8      ,begin     = 11
9      ,end       = 13
10     ,testing   = 0
11     );
12  ods _all_ close;
13  ODS PDF
14     file = "demo-ods-%sysfunc(juldate7(&Begin)).pdf"
15     ;
16  PROC Print data = &Data.
17     (where = (&Begin <= &Var <= &End) );
18  title3 "data = &Data..&Var in range "
19     "%sysfunc(putn(&Begin,worddate18.)) "
20     "-- %sysfunc(putn(&End ,mmdyy10. ))"
21     ;
22  run;
23  ods _all_ close;
24  ods listing;
25  %mend daterpt0;
```

- Notes:
- the print procedure is used in this example; provide your own when modifying
 - to refresh date in title1 use `options dtreset;`
 - title and footnote statements may have multiple text strings as their arguments;
- ! → remember to end each string with a space
- dates may be displayed with various formats; see Fehd [5, sco.Macro-Vars-of-Date-and-Time] for other formats and functions; to justify text in title or footnote see Fehd [4, sco.Macro-TextLine]
 - unit tests of this macro are on the web: Fehd [11, sco.Macro-Loops-With-Dates] See program info-in-dates.sas and Fehd [5, sco.Macro-Vars-of-Date-and-Time] for functions and formats to use in placing date- and time-stamps in ODS filenames.

Continued on next page.

Date-Report-Zero Demo

This program can be used to test macro daterpt0 for any previous interval. The `intnx` increment value of minus one is used to provide dates of the previous interval.

```
_____ daterpt0-demo-prev-any.sas _____
8  *let today = %sysevalf('01May2013'd);
9
10 *choice: previous which?;
11 %let interval = year;
12 %let interval = quarter;
13 %let interval = month;
14 %let interval = week;
15
16 %let begin =%sysfunc(intnx(&interval.,&today,-1,begin));
17 %let end   =%sysfunc(intnx(&interval.,&today,-1,end  ));
18
19 %put today: &today %sysfunc(putn(&today,weekdate29.));
20 %put begin: &begin %sysfunc(putn(&begin,weekdate29.));
21 %put end   : &end   %sysfunc(putn(&end   ,weekdate29.));
22
23 %daterpt0(data = library.testdata
24           ,var  = date
25           ,begin = &begin
26           ,end   = &end
27           );
```

The log shows the beginning and ending dates of the previous week.

```
today: 19479 Wednesday, May 1, 2013
begin: 19469 Sunday, April 21, 2013
end : 19475 Saturday, April 27, 2013
```

Note: The above program shows the calculation of the begin and end dates before calling the macro in order to write the values and dates to the log.

In all later examples these calculation are in the macro call.

```
*report previous &interval by day;
%dateloop(start = %sysfunc(intnx(&interval.,&today,-1,begin))
           ,stop  = %sysfunc(intnx(&interval.,&today,-1,end  ))
           ,interval = day
           );
```

Reference or Value?

Contributors [1, www-wiki.Call-by-Reference] explains the difference between calling by reference and calling by value.

Macro DateLoop

Overview

Macro DateLoop is a function: its output is either a macro call, or tokens within a statement. Its purpose is to encapsulate a macro %do loop through a series of dates while generating calls to a macro reporting subroutine or resolving tokens within a statement.

It has parameters which support these design elements:

- %do loop
- `intnx` interval
- called macro, passing parameters to
- tokens: expanding references to elements of the associative array of dates
- testing

`%do loop` :

- start: begin
- stop: end

`intnx interval` : simple: year, quarter, month, week, day etc.;

complex:

- multiple: `intervalN`: week2 is 14 days
- shift-index: `interval.N`, the default is `week.1` which returns the week starting on Sunday; `interval=week.2` returns the week starting on Monday

`called macro` : and passing parameters to

- MacroName: the default is `put note`: for testing
- MacroParms: this parameter must be enclosed in the `%nrstr` function to hide the special characters equal sign and comma
- called macro — e.g.: Date-Report-Zero — must have parameters named Begin and End

`tokens` : MacroText contains one or more tokens containing references to elements of the associative array of dates

`testing` : this variable is provided for self-reporting, see: Fehd [12, nesug2007.cc12] and is related to:

- MacroName, whose default is `put note`:
- semicolon: if the MacroName is `put note`: then each macro call is a `%put` statement which must end with a semicolon

see also : Fehd [13, pnwsug2009.do-which] compares data and macro do iterative, until and while.

DateLoop Program Listing

```

                                dateloop.sas
1  /*      name: <UNC>\SAS-site\macros\dateloop.sas
2     author: Ronald J. Fehd  2013
3
4  Summary      : description   : encapsulates macro do loop of dates
5                  1. calls macro with parameters
6                      of begin and end
7                  2. returns tokens with references
8                      to associative array of dates
9     purpose    : facilitates calling of macros
10                 with date intervals
11
12 Contexts     : program group: list processing token generator
13                 program type: function
14                 SAS      type: macro function
15                 uses routines: macro named in parameter
16
17 Specifications: input   : required: Start, Stop, Interval
18                         either of: MacroName
19                         or MacroText
20                         optional: MacroParms for MacroName
21     NOTE: called macro must have parameters Begin and End
22     process: call MacroName with start and stop
23             or prepare macro vars for MacroText usage
24     output  : calls macro  &MacroName
25             or expansion of &MacroText
26
27 Parameters 1: MacroName = name of macro to call
28               ,MacroName = put :: default, for testing
29               ,MacroParms = additional parameters for called macro
30               ! -->      Constraint: must be enclosed in nrstr:
31               ,MacroParms = %nrstr(data=sashelp.class,var=Sex)
32     OR 2: ,MacroText = %nrstr(%nrstr(work.date&Year._&MM))
33               ,Semicolon = 0 :: no semicolon after macro call
34               ,Semicolon = 1 :: use when macroname is a statement
35                       note: reset when macroname=put
36               ,Testing   = 0 :: default, no extra messages in log
37               ,Testing   = 1 :: for testing, note: reset when
38                       options mprint source2;
39
40 usage: demo
41 %let today = %sysfunc(today());
42 %let interval = week;
43 %let Begin = %sysfunc(intnx(&interval,&today,-12,begin));
44 %let End = %sysfunc(intnx(&interval,&today,- 1,end ));
45 * testing;
46 %dateloop(start = &begin
47           ,stop = &end
48           ,interval = day
49           )
50 * add small data: concatenate data sets for report;
51 %put
52 %dateloop(start = &begin
53           ,stop = &end
54           ,interval = month
55           ,MacroText= %nrstr(work.Year_Month_&year._&mm._&month)
56           );

```

Continued on next page.

```

57 * subtract from big data: make subset for periodic snapshot(s)
58   of previous N intervals from transactions;
59 %let interval = month;
60 %let begin    = %sysfunc(intnx(&interval,&today,-1,begin));
61 %let end      = %sysfunc(intnx(&interval,&today,-1,end  ));
62 %dateloop(start    = &begin
63             ,stop    = &end
64             ,interval = week
65             ,MacroName = daterpt0
66             ,MacroParms = %nrstr(data=library.testdata,var=date)
67             )
68 see also:
69 sas.help: Incrementing Dates and Times
70           by Using Multipliers and by Shifting Intervals
71 sas.wiki: http://www.sascommunity.org/wiki/Macro\_Loops\_with\_Dates
72           http://www.sascommunity.org/wiki/Do\_which\_loop\_until\_or\_while
73 predecessors: http://www.sascommunity.org/wiki/Macro\_CallMacr
74               http://www.sascommunity.org/wiki/Macro\_CallText
75 *****/
76 %Macro dateloop
77     (start    =0 /* integer of date */
78     ,stop     =2 /* integer of date */
79     ,interval  =day /* simple in (week month quarter year)
80     /* intnx(interval: complex: week2==14 days week.2==Monday */
81     ,MacroName =put note:
82     ,MacroParms = /*%nrstr(data=sashelp.class,var=sex)*/
83     ,MacroText  =.
84     ,semicolon  =0
85     ,testing    =0
86     )/des = 'site: call macro w/dates or expand references'
87     /* ** store /* ** source /* */;
88 %local D_Begin D_End Format MacroCall
89     d dd ddd day day3  mm mon month
90     ccyy qtr year yy   week weekv weekw;
91 %let Format = weekdate29.;
92 %if "&MacroText" ne "." %then %let MacroName = ;
93 %if %scan(&MacroName,1) eq put %then %let Semicolon = 1;
94 %let Testing=%eval(    &Testing    or &Semicolon
95                    or %sysfunc(getoption(mprint )) eq MPRINT
96                    and %sysfunc(getoption(source2)) eq SOURCE2);
97 %put &SysMacroName start: %sysfunc(putn(&start,&format));
98 %put &SysMacroName stop: %sysfunc(putn(&stop ,&format));
99 %let D_Begin = %sysfunc(intnx(&Interval,&start,0,begin));
100

```

Continued on next page.

```

----- dateloop.sas -----
101 %do %until(&D_Begin gt &Stop);
102   %let D_End = %sysfunc(intnx(&Interval,&D_Begin,0,end));
103   %if &Testing %then %put
104     &SysMacroName: begin %sysfunc(putn(&D_Begin,&format));
105   %if "&MacroText" ne "." %then %do;
106     %let d = %sysfunc(weekday(&D_begin));
107     %let dd = %sysfunc(putn(%sysfunc(day(&D_begin)),z2));
108     %let ddd = %substr(%sysfunc(juldate7(&D_begin)),5,3);
109     %let day = %sysfunc(putn(&D_begin,downname));
110     %let day3 = %substr(&Day,1,3);
111     %let mm = %sysfunc(putn(%sysfunc(month(&D_begin)),z2));
112     %let mon = %sysfunc(putn(&D_begin,monname3));
113     %let month = %sysfunc(putn(&D_begin,monname));
114     %let qtr = %sysfunc(qtr (&D_begin));
115     %let ccyy = %sysfunc(year(&D_begin));
116     %let year = &ccyy;
117     %let yy = %substr(&ccyy,3,2);
118     %let week = %sysfunc(week(&D_begin,u));
119     %let weekv = %sysfunc(week(&D_begin,v));
120     %let weekw = %sysfunc(week(&D_begin,w));
121     %unquote(&MacroText)
122   %end;
123 %else %do;
124   %if &Testing %then %put
125     &SysMacroName: end %sysfunc(putn(&D_End ,&format));
126   %let MacroCall = &MacroName(;
127   %if %length(&MacroParms) %then
128     %let MacroCall = &MacroCall.%unquote(&MacroParms,);
129   %let MacroCall = &MacroCall.begin=&D_Begin,end=&D_End);
130   %put &SysMacroName calling &MacroCall;
131   %&MacroCall
132   %end;
133 %if &Semicolon %then %do;
134   ;
135 %end;
136 %let D_Begin = %eval(&D_End +1);
137 %end;
138 %mend dateloop;

```

- Notes:
- unit tests of this macro are in Fehd [11, sco.Macro-Loops-With-Dates]
 - parameters MacroName, MacroParms and Semicolon, and assemblage of MacroCall are from Fehd [7, sco.Macro-CallMacro]
 - parameter MacroText is from Fehd [8, sco.Macro-CallText]
 - macro variable Testing is explained in Fehd [12, nesug2007.cc12]
 - Fehd [13, pnwsug2009.do-which] compares data and macro do iterative, until and while
-

DateLoop Testing: Simple and Complex Intervals

Overview

The default value of the DateLoop parameter MacroName is `put note:.`. This feature facilitates testing because the macro writes two set of notes to the log using the format `weekdate29`.

1. start and stop
2. date-begin and -end inside the loop

start, stop : These notes are always written to the log; they resolve the dates passed to the macro.

```
DATELOOP start:      Sunday, April 21, 2013
DATELOOP stop:       Saturday, April 27, 2013
```

begin, end : These notes from within the loop are the values passed to the macro named in the parameter MacroName. Alternately, if the parameter MacroText is used then it is expanded from the `begin` date.

```
DATELOOP: begin      Sunday, March 31, 2013
DATELOOP: end        Saturday, April 6, 2013
```

Previous Month by Week

These logs show the use of the `week.shift-index` which can be used to change the date-start of each weekly report.

interval=week The log of this macro call shows the date-start is 2013-Apr-01. The default shift-index is `week.1`: reports begin on Sunday, the first day of the week. The date range for the first week report contains Monday, April 1, 2013.

```
13 %let today = %ysevalf('01May2013'd);
14 %let interval = month;
15 %let date_start = %sysfunc(intnx(&interval.,&today,-1,begin));
16 %let date_stop = %sysfunc(intnx(&interval.,&today,-1,end ));
17 *report first (short) week report includes first of month;
18 %dateloop(start = &date_start
19           ,stop = &date_stop
20           ,interval = week)
DATELOOP start:      Monday, April 1, 2013
DATELOOP stop:       Tuesday, April 30, 2013
DATELOOP: begin      Sunday, March 31, 2013
DATELOOP: end        Saturday, April 6, 2013
note: (begin=19448,end=19454)
...
DATELOOP: begin      Sunday, April 28, 2013
DATELOOP: end        Saturday, May 4, 2013
note: (begin=19476,end=19482)
```

interval=week.weekday Here the shift-index is soft-coded as the weekday of the first. The date range for the first week's report is always the 1st through the 7th.

```
29 * weekly reports begin on first: ThisDay in (1 8 15 22 29);
30 %dateloop(start = &date_start
31           ,stop = &date_stop
32           ,interval = week.%sysfunc(weekday(&date_start)) );
DATELOOP: begin      Monday, April 1, 2013
DATELOOP: end        Sunday, April 7, 2013
```

Continued on next page.

Testing Summary

This program can be used to test macro DateLoop for any previous interval, by any lesser interval. The `intnx` increment value of minus one is used to provide dates of the previous interval.

```
----- dateloop-demo-prev-any.sas -----
12 %let today = %sysfunc(today());
13 ** choices: previous which?, by?;
14 %let intervals = year-quarter;
15 *let intervals = year-month;
16 *let intervals = quarter-month;
17 *let intervals = quarter-week;
18 *let intervals = month-week;
19 *let intervals = month-day;
20 %let intervals = week-day;
21
22 %let int_main = %scan(&intervals,1,-);
23 %let int_by = %scan(&intervals,2,-);
24 ** report previous &int_main by &int_by;
25 %dateloop(start = %sysfunc(intnx(&int_main.,&today,-1,begin))
26           ,stop = %sysfunc(intnx(&int_main.,&today,-1,end ))
27           ,interval = &int_by
28           );
```

The log shows the date of the previous week.

```
DATELOOP start:          Sunday, June 30, 2013
DATELOOP stop:           Saturday, July 6, 2013
DATELOOP: begin          Sunday, June 30, 2013
DATELOOP: end             Sunday, June 30, 2013
note: (begin=19539,end=19539)
DATELOOP: begin          Monday, July 1, 2013
DATELOOP: end             Monday, July 1, 2013
note: (begin=19540,end=19540)
...
DATELOOP: begin          Saturday, July 6, 2013
DATELOOP: end             Saturday, July 6, 2013
note: (begin=19545,end=19545)
```

Parameter MacroName

Overview

Macro DateLoop has a parameter MacroName and optional parameter MacroParms which are used to call a reporting macro.

! → The reporting macro must have parameters `begin` and `end` which are passed in the macro call.

This feature is used to subtract a subset from a larger data set. In database terminology the report produced is called a periodic snapshot; see also Fehd [3, sco.DatabaseVocabulary].

Demo: Weekly Reports

Here is an example program.

```
_____ dateloop-demo-prev-month-by-week.sas _____
1 *weekly reports begin on first: ThisDay in (1 8 15 22 29);
2 %dateloop(start      = &begin
3     ,stop           = &end
4     ,interval       = week.%sysfunc(weekday(&begin))
5     ,MacroName      = daterpt0
6     ,MacroParms     = %nrstr(data=library.testdata,var=date)
7     ,testing        =1);
```

and the log:

```
DATELOOP start:      Monday, April 1, 2013
DATELOOP stop:       Tuesday, April 30, 2013
DATELOOP: begin      Monday, April 1, 2013
DATELOOP: end        Sunday, April 7, 2013
DATELOOP calling daterpt0(data=library.testdata
                          ,var=date,begin=19449,end=19455)
...
DATELOOP: begin      Monday, April 8, 2013
DATELOOP: end        Sunday, April 14, 2013
DATELOOP calling daterpt0(data=library.testdata
                          ,var=date,begin=19456,end=19462)
...
DATELOOP: begin      Monday, April 29, 2013
DATELOOP: end        Sunday, May 5, 2013
DATELOOP calling daterpt0(data=library.testdata
                          ,var=date,begin=19477,end=19483)
```

Parameter MacroText

Overview

Macro DateLoop is a function: it returns no statements, only tokens, which may be either macro calls, or text with references to elements of the associative array of dates, which are provided in the parameter MacroText.

This feature is used to add or concatenate a number of date-stamped data sets into a super-set.

Demo: Previous 7

This macro call returns a series of tokens containing a date-stamped data set name: libref.date-ccyy-mm.

This example shows how to check the resolution by using

```
MacroText=%nrstr(%put ...).
```

! → Remember to add semicolon=1 with this feature.

```
%dateloop(start = %sysfunc(intnx(&interval,&today,-7,sameday))
,stop = %sysfunc(intnx(&interval,&today,0,end ))
,interval = day
,MacroText= %nrstr(%put &Library..date&ccYY._&MM._&DD)
,semicolon=1
)
```

The log shows the set of tokens returned by DateLoop.

```
DATELOOP start:      Wednesday, July 3, 2013
DATELOOP stop:       Wednesday, July 10, 2013
work.date2013_07_03
work.date2013_07_04
...
work.date2013_07_09
work.date2013_07_10
```

Demo: Previous 12

This program shows a data step which reads the previous twelve months of date-stamped data set names.

```
%let Library = work;
DATA Previous_12;
do until(EndoFile);
  set %dateloop
    (start = %sysfunc(intnx(month,&today,-12,sameday))
,stop = %sysfunc(intnx(month,&today,-1 ,end ))
,interval = month
,MacroText = %nrstr(&Library..date&ccYY._&MM)
) end = EndoFile;
  output;
end;
stop;
run;
```

log

```
DATELOOP start:      Tuesday, July 10, 2012
DATELOOP stop:       Sunday, June 30, 2013
...
NOTE: There were 31 observations read from WORK.DATE2012_07.
NOTE: There were 31 observations read from WORK.DATE2012_08.
...
NOTE: There were 31 observations read from WORK.DATE2013_05.
NOTE: There were 31 observations read from WORK.DATE2013_06.
NOTE: WORK.PREVIOUS_12 has 366 observations and 5 variables.
```

Summary

Conclusion

Splitting a reporting task with dates into a reporting subroutine and a calling routine function which standardizes the macro %do loop and calculation of date intervals yields tools that are easy to test and reuse.

Further Reading

- Programs : and updates to this paper are in Fehd [11, sco.Macro-Loops-With-Dates]
- Inspiration : Rhodes [17, sugi31.015] provides a report of the previous week and all weeks preceding using formats.
- Database Vocabulary : Fehd [3, sco.DatabaseVocabulary].
- Dates and Times : Morgan [16, SAS-Press-2006.Morgan-Guide-Dates-Times]. Finley [15, sugi25.084] shows how to use year.N for fiscal year.
- Functions : nwkdom: N-week-day-of-month examples are in Schreier [18, sco.Generating-Holiday-Lists]. Fehd [7, sco.Macro-CallMacro] is the predecessor and model for the macro function DateLoop. Fehd [8, sco.Macro-CallText] is the predecessor and model for the macro function DateLoop parameter MacroText. Fehd [4, sco.Macro-TextLine] provides methods to justify text in titles and footnotes.
- List Processing : Fehd and Carpenter [14, sco.List-Processing-Basics] recommend separating the %do loop from the report.
Fehd [6, sco.SmryEachVar] is a list processing suite using parameterized %includes.
Fehd [10, sco.Routine-CxMacro] routine CxMacro is the predecessor of Macro Call-Macro.
- Macro %Do : These papers contain tips for modifying macro loops. Fehd [9, sco.Macro-Do-Loop] is a macro function that returns tokens in a statement; macro function DateLoop can call a macro function that does this. Fehd [13, pnwsug2009.do-which] compares data and macro do iterative, until and while. Woodruff and Dunn [19, sesug2010.ff01] compare do and %do loops.
- Testing : Fehd [12, nesug2007.cc12] explains the use of the macro variable Testing.
-

Bibliography

- [1] Wikipedia Contributors. Evaluation strategy. In *Wikipedia, The Free Encyclopedia*, 2013. URL http://en.wikipedia.org/w/index.php?title=Evaluation_strategy&oldid=540886400. call by reference or call by value; online; accessed 2013-Mar-01.
 - [2] Edsger W. Dijkstra. Why numbering should start at zero. In *Transcripts*. Univ/Texas, 1982. URL <http://www.cs.utexas.edu/users/EWD/transcriptions/EWD08xx/EWD831.html>. online; accessed 2013-Mar-01.
 - [3] Editor R.J. Fehd. Database vocabulary. In *sasCommunity.org*, 2007. URL http://www.sascommunity.org/wiki/Database_Vocabulary.
 - [4] Editor R.J. Fehd. Macro TextLine. In *sasCommunity.org*, 2008. URL http://www.sascommunity.org/wiki/Macro_TextLine. justify text within a title or footnote.
 - [5] Editor R.J. Fehd. Macro variables of date and time. In *sasCommunity.org*, 2008. URL http://www.sascommunity.org/wiki/Macro_Variables_of_Date_and_Time. formats and functions for date-stamping.
 - [6] Editor R.J. Fehd. SmryEachVar: A data-review suite for each variable in all data sets in a libref. In *sasCommunity.org*, 2008. URL http://www.sascommunity.org/wiki/SmryEachVar_A_Data_Review_Suite. list processing using parameterized includes.
 - [7] Editor R.J. Fehd. Macro Call-Macro. In *sasCommunity.org*, 2012. URL http://www.sascommunity.org/wiki/Macro_CallMacr. using SCL functions to read a data set and call macros.
 - [8] Editor R.J. Fehd. Macro Call-Text. In *sasCommunity.org*, 2012. URL http://www.sascommunity.org/wiki/Macro_CallText. using SCL functions to read a data set and return tokens within a statement.
 - [9] Editor R.J. Fehd. Macro do-loop. In *sasCommunity.org*, 2012. URL http://www.sascommunity.org/wiki/Macro_Do-Loop. macro function to return tokens inside a statement.
 - [10] Editor R.J. Fehd. Routine CxMacro. In *sasCommunity.org*, 2012. URL http://www.sascommunity.org/wiki/Routine_CxMacro. parameterized include for calling macros, preprocessor is SmryEachVar routine CxInclude.
 - [11] Editor R.J. Fehd. Macro loops with dates. In *sasCommunity.org*, 2013. URL http://www.sascommunity.org/wiki/Macro_Loops_with_Dates. example macros, programs and updates.
 - [12] Ronald J. Fehd. Writing testing-aware programs that self-report when testing options are true. In *NorthEast SAS Users Group Annual Conference Proceedings*, 2007. URL www.nesug.org/Proceedings/nesug07/cc/cc12.pdf. topics: functions: sysfunc, getoption; macro variable Testing.
 - [13] Ronald J. Fehd. Do which? loop, until or while? a review of data step and macro algorithms. In *Western Users of SAS Software Annual Conference Proceedings*, 2009. URL <http://www.lexjansen.com/pnwusug/2009/Fehd,%20Ron%20-%20Do%20Which%20Loop,%20Until%20or%20While.pdf>.
 - [14] Ronald J. Fehd and Art Carpenter. List processing basics: Creating and using lists of macro variables. In *sasCommunity.org*, 2009. URL http://www.sascommunity.org/wiki/List_Processing_Basics_Creating_and_Using_Lists_of_Macro_Variables. Hands On Workshop, 20 pp.; separate report macro from list processing macro; comparison of methods: making and iterating macro arrays, scanning macro variable, writing calls to macro variable, write to file then include, call execute and nrstr; 11 examples, bibliography.
 - [15] Wayne Finley. What fiscal year is this and when does it start and end? In *Proceedings of the 25th Annual SAS® Users Group International Conference*, 2000. URL <http://www2.sas.com/proceedings/sugi25/25/cc/25p084.pdf>. using interval=year.N for beginning of fiscal year.
 - [16] Derek Morgan. *The Essential Guide to SAS Dates and Times*. SAS Institute, 2006. ISBN 978-1-59047-884-4. URL <https://support.sas.com/pubscat/bookdetails.jsp?pc=59411>. 5 chapters: intro, displaying, converting, functions, macro variables, shifting and graphing; 172 pp., index.
 - [17] Dianne Louise Rhodes. Pretty dates all in a row. In *Proceedings of the 31st Annual SAS® Users Group International Conference*, 2006. URL <http://www2.sas.com/proceedings/sugi31/015-31.pdf>. topics: dates, functions: intck, intnx; intervals.
 - [18] Editor H. Schreier. Generating holiday lists. In *sasCommunity.org*, 2008. URL http://www.sascommunity.org/wiki/Generating_Holiday_Lists. illustrates use of functions: holiday, ifc, intnx, mdy, nwkdom, weekday.
 - [19] Sarah Woodruff and Toby Dunn. Take control: Understanding and controlling your do-loops. In *SouthEast SAS Users Group Annual Conference Proceedings*, 2010. URL <http://analytics.ncsu.edu/sesug/2010/FF01.Woodruff.pdf>. Foundations and Fundamentals, 21 pp.; comparison of data step and macro do loops.
-

Closure**Acknowledgements**

My thanks to SAS-L contributors Yevgeniy Bolotin and Howard Schreier for commentary and proofreading.

Contact Information:**Ronald J. Fehd**<mailto:Ron.Fehd.macro.maven@gmail.com>http://www.sascommunity.org/wiki/Ronald_J._Fehd**About the author:**

education:	B.S. Computer Science, U/Hawaii,	1986
	SAS User Group conference attendee since	1989
	SAS-L reader	since 1994
experience:	programmer: 25+ years	
	data manager using SAS:	17+ years
	statistical software help desk:	7+ years
	author: 30+ SUG papers	
	sasCommunity.org: 300+ pages	
SAS-L:	author: 6,000+ messages to SAS-L since	1997
	Most Valuable SAS-L contributor:	2001, 2003

Trademarks

SAS and all other SAS Institute Inc. product or service names are registered trademarks or trademarks of SAS Institute Inc. In the USA and other countries ® indicates USA registration.

Close Quote

A SAS date is an index to the associative array of date information.

— R. J. Fehd