Paper 115-28

# Date Parameters for Interval Reporting

Nina L. Werner, Dean Health Plan, Inc., Madison, WI

## ABSTRACT

Once you realize how easy it is to let SAS® get your interval dates for you, you will never be entering date parameters again.

The intended audience is Base SAS® users of all experience levels on any platform.

## INTRODUCTION

To simplify the running of repeating reports which are scheduled at regular intervals, we can use SAS® functions to create macro variables to represent the date range for the current execution of our report. This can prevent accidental introduction of errors and make us more productive, since this code is reusable. In a batch production environment, where SAS® jobs can be scheduled to run on a regular basis without human intervention, this avoids the necessity of changing parameters each time we run. These techniques will be applicable even if we need to submit from our Editor Window, so we will not have to revise date-specific code every time.

## Problem: Run Quarterly Report

Four times a year, a report must be generated to compare current year and quarter year-to-date with the previous year. We do not want to make changes to the program or have to enter or hard-code date parameters for each run.

Constraints:
1. Data collection requires at least thirty days after the end of the quarter to assure that all data has been entered.
2. This report needs to run on a quiet night, not in contention with weekly or monthly jobs.

It will be put on the production schedule to run automatically every three months, on the second Thursday night of the month after the month after the quarter ends (or about six weeks after each quarter.) In 2002, it ran on May 9, August 8, November 14 and February 13, 2003 for 2002 - Q1, Q2, Q3, Q4.

How can we get the start and end dates for the report automatically in our program?

```
DATA  _NULL_ ;
   CALL SYMPUT('qtr_end', "'"|| TRIM(LEFT(PUT(
               INTNX('MONTH',DATE(),-2,'END'),
                   DATE9.))) ||"'d") ;
RUN ;
%PUT End of Quarter= &qtr_end ;
```

### CALL SYMPUT

Use CALL SYMPUT in a DATA _NULL_ step to move a value into a new macro variable. The syntax is:

**CALL SYMPUT('varname', character string value).**

Here, for example, we create a macro variable called 'qtr_end' which will contain the last day of the quarter of interest in SAS® DATE9 format. We can retrieve the value later by using the name with an ampersand: &qtr_end.

The first parameter (*varname*) in the CALL SYMPUT is 'qtr_end'. The single quotes are required. The second parameter (*character string value*) after the comma contains several functions, including the concatenation symbol **||** which is used to surround the actual DATE9 string with single quotes and the letter **d** following, to create a SAS® date constant. The TRIM and LEFT functions will left justify the string and remove following blanks. Note where the corresponding close parentheses appear.

Also embedded in the character string value are the INTNX and PUT functions.

### INTNX

The INTNX date function will move from a given date to the desired date, forward or backward, based on four parameters. The result of using this function is a SAS® date (numeric) value. The syntax is:

**INTNX('interval', from date, number of intervals, 'alignment').**

The first parameter is an interval key word enclosed in single quotes. This can be YEAR, QTR, MONTH, DAY, several others. The second parameter is a SAS® date, in this case the current (run) date obtained from the system date function, DATE(). The older version of this SAS® date function, TODAY(), still works also. The third parameter is a number, how many intervals we wish to move. Here, we are looking back two months from the run date, so the value we need is -2. The last parameter is an alignment key word enclosed in single quotes which can be BEGINNING [or BEG or B], MIDDLE [or MID or M], END [or E]. This parameter is optional and defaults to B (first day of the period).

### PUT

The PUT function moves a numeric value, today's date, into a character string based on the format provided. The syntax is:

**PUT(numeric value, format.).**

The numeric value being passed here is the output of INTNX, the date of the last day of the month two months ago. The format is DATE9.

## FINAL PROGRAM CODE

To compare this year and quarter year-to-date with last year, we require start and end dates for each year. The four macro variables &start and &end for this year and &start0 and &end0 for last year are created in the program segment below. Later in the program, they are used for data selection. Notice that the end date is selected here with the INTNX parameters 'QTR', -1 instead of 'MONTH', -2. They are equivalent in this case. The prior year value is 'QTR', -5. The start dates are the first day of this year 'YEAR', 0,'BEG' and last year 'YEAR', -1,'BEG'. The last parameter is optional here because 'BEG' is the default value.

```
DATA  _NULL_ ;
  CALL SYMPUT('start', "'"|| TRIM(LEFT(PUT(
              INTNX('YEAR',DATE(),0,'BEG'),
                        DATE9.))) ||"'d") ;
  CALL SYMPUT('end', "'"|| TRIM(LEFT(PUT(
              INTNX('QTR',DATE(),-1,'END'),
                         DATE9.))) ||"'d") ;
  CALL SYMPUT('start0', "'"|| TRIM(LEFT(PUT(
             INTNX('YEAR',DATE(),-1,'BEG'),
                         DATE9.))) ||"'d") ;
  CALL SYMPUT('end0', "'"|| TRIM(LEFT(PUT(
              INTNX('QTR',DATE(),-5,'END'),
                          DATE9.))) ||"'d") ;
RUN ;
%PUT This year  &start - &end ;
%PUT Last year &start0 - &end0 ;
* ... ;
PROC  SQL ;
  SELECT ...
    FROM ...
   WHERE (filedate BETWEEN &start
                       AND &end
      OR  filedate BETWEEN &start0
                       AND &end0) ;
QUIT ;
```

## ALTERNATE CODE

For data retrieval from databases other than SAS®, the date formats may be MMDDYY10 or DATETIME instead of DATE9. Then, the single quotes and the letter **d** will not be necessary.

```
DATA  _NULL_ ;
   CALL SYMPUT('qtr_end', TRIM(LEFT(PUT(
               INTNX('MONTH',DATE(),-2,'END'),
                                MMDDYY10.))) ) ;
RUN ;
%PUT End of Quarter= &qtr_end ;
```

## CONCLUSION

Use SAS® functions to create macro variables to represent the date range for the current execution of repeating reports. Avoid the necessity of changing parameters each time.

## REFERENCES

SAS® Online Doc – Version 8, 1999, SAS Institute Inc., Cary, NC, USA..

## ACKNOWLEDGMENTS

Thanks to LeRoy Bessler, Bessler Consulting & Research, Milwaukee, WI, for encouraging me to give another presentation.

## CONTACT INFORMATION

Your comments and questions are valued and encouraged.
Contact the author at:

**Nina L. Werner**, BA, MBA, MHP
Dean Health Plan, Inc.
1277 Deming Way
Madison, WI 53717
Work Phone: (608) 827-4224
Fax: (608) 836-6335
Email: **Nina.Werner@Deancare.com**
Web: **www.Deancare.com**

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.

Key words: **CALL SYMPUT**, **DATE()**, **INTNX**, **PUT**, **macro variable, TODAY()**.