PharmaSUG 2012 - Paper CC19

# DON'T HAVE A "WEEK" REPORT

Barbara Ross, DataX Ltd, Las Vegas, Nevada

## ABSTRACT

Often we need to generate reports on an hourly, daily, and weekly basis. Hourly and daily are pretty straightforward, but when it comes to week, there are many ways to define it, code for it, and display it. This paper attempts to cover a lot the various methods of classifying weeks within your longitudinal data. It will show how to use the SAS WEEK function and formats to enumerate your weeks, the INTNX function to apply date range labels and specify non-traditional weeks, and other helpful tips such as filling in sparse summary tables and placing date ranges in your title. The topics covered are appropriate for novice SAS users.

## INTRODUCTION

There are two mains ways to classify your data into weeks. You can number your weeks (e.g. 1, 2, 3…) or apply date range labels (e.g. 1/1 - 1/7 ). To enumerate your weeks, SAS supplies easy to use WEEK functions and formats. These methods are great if you just want to select a subset of your longitudinal data or if you need not display the date ranges. If you need to include the date ranges, then the INTNX function is best. The INTNX function can also be used to group non-traditional weeks such as weeks not starting on Sunday or 2 week cycles.

## ENUMERATING WEEKS

### WEEK FUNCTION

The SAS supplied WEEK function reads a SAS date value and returns a week-number value specifying the number of that week with the year. It also allows you to specify a descriptor that can change the start day of the week and how the beginning and end of the year are counted.

```
Syntax:   WEEK(<date>,<'descriptor'>);
Example:  WEEK('16dec2011'd,'u');
```

For descriptors U and W, a week value of zero represents days before the first week of the year. Once the first Sunday or Monday (respectively) of the year is encountered then the week will be 1. Descriptor V has no week 0 and days before the first Monday of the year are counted as the last week of the previous year.

Table 1 shows the differences among descriptors:

| Descriptor | Range | Week Starts On: | Ex: Sun, 1 Jan 2006 | Ex: Mon, 2 Jan 2006 |
|---|---|---|---|---|
| U (Default) | 0-53 | Sunday | 1 | 1 |
| W | 0-53 | Monday | 0 | 1 |
| V | 1-53 | Monday | 52 | 1 |

**Table 1. WEEK Function: Descriptor Settings**

### WEEK FORMATS

The WEEK formats write a week-number value specifying the number of the week with the year for a SAS date value. The WEEKU*w*. format uses the same algorithm as the 'U' descriptor for the WEEK function for calculating week. Respectively the WEEKW*w*. and WEEKV*w*. use the 'W' and 'V' algorithms. The default width for these formats is 11-200 which outputs the year, week, and day of that week. If you use the WEEKU*w*. format, day of the week will start with Sunday equaling 1. If you use the WEEKW*w*. or WEEKV*w.* formats then Monday will equal 1. Also note that the minimum width allowed is 3, so you will have to edit the result to get rid of the "W" that is placed before the week number.

Table 2 gives examples:

| SAS Variable Format | Statement: | Result: | Format: |
|---|---|---|---|
| WEEKU. | PUT('16DEC2011'd,weeku.); | 2011-W50-06 | yyyy-Www-dd |
| WEEKW. | PUT('16DEC2011'd,weekw.); | 2011-W50-05 | yyyy-Www-dd |
| WEEKU3. | PUT('16DEC2011'd,weeku3.); | W50 | Www |
| WEEKU5. | PUT('16DEC2011'd,weeku5.); | 11W50 | yyWww |

**Table 2. WEEK Format Example Output**

## CUSTOM WEEK FORMAT

If you're feeling ambitious you can make your own format to classify weeks. This can be done by using the PICTURE statement in the FORMAT procedure.

Let's say you want the week to read "Week 50, Thursday" for example…

```
PROC FORMAT;
   PICTURE cweek low-high='Week %U,%A      ' (DATATYPE=date);
   RUN;
DATA Last3weeks; SET global_all;
   WeekLabel = PUT(trans_date,cweek.);
   RUN;
```

Output:

VIEWTABLE: Work.Last3weeks

| | TRANS_DATE | WeekLabel |
|---|---|---|
| 1 | 11/27/11 | Week 48,Sunday |
| 2 | 11/28/11 | Week 48,Monday |
| 3 | 11/29/11 | Week 48,Tuesday |
| 4 | 11/30/11 | Week 48,Wednesday |
| 5 | 12/01/11 | Week 48,Thursday |
| 6 | 12/02/11 | Week 48,Friday |
| 7 | 12/03/11 | Week 48,Saturday |
| 8 | 12/04/11 | Week 49,Sunday |
| 9 | 12/05/11 | Week 49,Monday |
| 10 | 12/06/11 | Week 49,Tuesday |
| 11 | 12/07/11 | Week 49,Wednesday |
| 12 | 12/08/11 | Week 49,Thursday |
| 13 | 12/09/11 | Week 49,Friday |
| 14 | 12/10/11 | Week 49,Saturday |
| 15 | 12/11/11 | Week 50,Sunday |

**Display 1. Custom Week Formats**

The %U and %A are directives that display information about the date value. The %U tells it to output the week number using the U algorithm. The %A outputs the full weekday name. The DATATYPE= argument allows you to use the function for a date value. Also please note the spaces after %A in this example. The picture format will have the width that is used in its definition- without the spaces following %A, the format would cut the weekday label off after the first two letters.

Another useful custom format could include the week information and the date (such as: Week 50, 12/16/11):

```
PICTURE dweek low-high='Week %U, %m/%d/%y' (DATATYPE=date);
```

## SUBSET BY NUMBER OF WEEKS

The SAS WEEK function and formats are particularly useful for quickly grabbing a subset of data for your report. They keep you from having to subtract X amount of days or looking up the date of Sunday 4 weeks ago.

The following code creates a subset of my last 3 weeks of data:

```
DATA last3weeks; SET global_all;
   WHERE week(trans_date) >= week(today())-2;
   RUN;
```

## DATE RANGES FOR WEEKS

The week number is great for simply separating weeks, but if you want to define these weeks out in a report- you're probably going to need some date ranges. For this, the INTNX function is invaluable.

### INTNX FUNCTION

The INTNX function increments a date value by a given time interval and returns a date value. Useful for determining the start and end points of a period of time.

Syntax: `INTNX(interval, start-from, increment,<alignment>);`

### DATE RANGE LABEL

To create a date range week label, you will need to 1) Find the two INTNX values for the beginning and end of the week 2) Apply a format to those values since they are returned in a Julian format and 3) Concatenate the two values to make a range. If the dataset includes a partial last week (week to-date) , then you can use an IF statement to apply a correct label.

The complete code is as follows:

```
DATA last3weeks; SET last3weeks;
    start = intnx('week',trans_date,0,'beginning');
    end = intnx('week',trans_date,0,'end');
    DateRange = catx(' - ', PUT(start,mmddyy8.),put(end,mmddyy8.));
    IF week(trans_date) = week(today())
    THEN DateRange = catx(' - ',PUT(start,mmddyy8.),put(today(),mmddyy8.));
    RUN;
```

Output:

VIEWTABLE: Work.Last3weeks

| | TRANS_DATE | DateRange | WeekLabel |
|---|---|---|---|
| 1 | 11/27/11 | 11/27/11 - 12/03/11 | Week 48,Sunday |
| 2 | 11/28/11 | 11/27/11 - 12/03/11 | Week 48,Monday |
| 3 | 11/29/11 | 11/27/11 - 12/03/11 | Week 48,Tuesday |
| 4 | 11/30/11 | 11/27/11 - 12/03/11 | Week 48,Wednesda |
| 5 | 12/01/11 | 11/27/11 - 12/03/11 | Week 48,Thursday |
| 6 | 12/02/11 | 11/27/11 - 12/03/11 | Week 48,Friday |
| 7 | 12/03/11 | 11/27/11 - 12/03/11 | Week 48,Saturday |
| 8 | 12/04/11 | 12/04/11 - 12/10/11 | Week 49,Sunday |
| 9 | 12/05/11 | 12/04/11 - 12/10/11 | Week 49,Monday |
| 10 | 12/06/11 | 12/04/11 - 12/10/11 | Week 49,Tuesday |
| 11 | 12/07/11 | 12/04/11 - 12/10/11 | Week 49,Wednesda |
| 12 | 12/08/11 | 12/04/11 - 12/10/11 | Week 49,Thursday |
| 13 | 12/09/11 | 12/04/11 - 12/10/11 | Week 49,Friday |
| 14 | 12/10/11 | 12/04/11 - 12/10/11 | Week 49,Saturday |
| 15 | 12/11/11 | 12/11/11 - 12/17/11 | Week 50,Sunday |
| 16 | 12/12/11 | 12/11/11 - 12/17/11 | Week 50,Monday |

**Display 2. Applying Date Range Labels to Dates**

### ADDING WEEK LABEL TO TITLES

Most likely you want to add the date range to the title of the report or tables. This can be done using macros and a null DATA Step.

The following example creates a label for last week:

```
DATA _null_;
    start=PUT(intnx('week',today()-7,0,'beginning'),mmddyy8.);
    end=PUT(intnx('week',today()-7,0,'end'),mmddyy8.);
    range=catx(' - ',start,end);
    CALL SYMPUT ("last_week",range);
        RUN;
TITLE "Transaction Counts for &last_week";
```

If you are unsure of the date range for your report then you can use the following code which selects the first and last dates of the dataset:

```
PROC SORT DATA=last3weeks; BY trans_date; RUN;
DATA _null_;
    SET last3weeks;
    IF _n_=1 THEN CALL SYMPUT ('start',PUT(trans_date,mmddyy8.));
    CALL SYMPUT ('end',PUT(trans_date,mmddyy8.));
    range=catx(' - ',"&start","&end");
    CALL SYMPUT ("daterange",range);
    RUN;
TITLE "Transaction Counts for &daterange";
```

## NON-TRADITIONAL WEEKS

The quickest way to group weeks with a start date other than Sunday is by using Shift Operators with the INTNX function. If you recall the first parameter specified in the INTNX function is the interval. Multipliers and shift indexes can be used with the basic interval names to construct more complex intervals. When using multipliers and shift indexes your interval name will have the format: NAME*m.s*

NAME is the basic interval time. In our case, it would be WEEK

*m* is an optional multiplier which multiplies the interval by the specified value. For example, the interval WEEK2 consists of two-week periods.

*S* is an optional starting point for the interval. By default, the WEEK interval starts on Sunday with Sunday=1. So if you want a week to start on Wednesday (value =4) then the interval would be WEEK.4 .

Both the multipliers and shift index are optional and default to 1. So the intervals, WEEK,WEEK1.1,WEEK.1 are all equivalent to the basic week interval. Different intervals are shifted by different subperiods. For our purposes, week is shifted by days. Also the shift index cannot be greater than the number of subperiods in the whole interval (so no greater than 7).

For example, let's take our last3weeks dataset but define the weeks to start on Thursdays:

```
DATA last3weeks; SET last3weeks;
    start = intnx('week.5',trans_date,0,'beginning');
    end = intnx('week.5',trans_date,0,'end');
    ThuWeeks = catx(' - ', PUT(start2,mmddyy8.),put(end2,mmddyy8.));
    RUN;
```

Output:

VIEWTABLE: Work.Last3weeks

| | TRANS_DATE | ThuWeeks | WeekLabel | DateRange |
|---|---|---|---|---|
| 1 | 11/27/11 | 11/24/11 - 11/30/11 | Week 48,Sunday | 11/27/11 - 12/03/11 |
| 2 | 11/28/11 | 11/24/11 - 11/30/11 | Week 48,Monday | 11/27/11 - 12/03/11 |
| 3 | 11/29/11 | 11/24/11 - 11/30/11 | Week 48,Tuesday | 11/27/11 - 12/03/11 |
| 4 | 11/30/11 | 11/24/11 - 11/30/11 | Week 48,Wednesday | 11/27/11 - 12/03/11 |
| 5 | 12/01/11 | 12/01/11 - 12/07/11 | Week 48,Thursday | 11/27/11 - 12/03/11 |
| 6 | 12/02/11 | 12/01/11 - 12/07/11 | Week 48,Friday | 11/27/11 - 12/03/11 |
| 7 | 12/03/11 | 12/01/11 - 12/07/11 | Week 48,Saturday | 11/27/11 - 12/03/11 |
| 8 | 12/04/11 | 12/01/11 - 12/07/11 | Week 49,Sunday | 12/04/11 - 12/10/11 |
| 9 | 12/05/11 | 12/01/11 - 12/07/11 | Week 49,Monday | 12/04/11 - 12/10/11 |
| 10 | 12/06/11 | 12/01/11 - 12/07/11 | Week 49,Tuesday | 12/04/11 - 12/10/11 |
| 11 | 12/07/11 | 12/01/11 - 12/07/11 | Week 49,Wednesday | 12/04/11 - 12/10/11 |
| 12 | 12/08/11 | 12/08/11 - 12/14/11 | Week 49,Thursday | 12/04/11 - 12/10/11 |
| 13 | 12/09/11 | 12/08/11 - 12/14/11 | Week 49,Friday | 12/04/11 - 12/10/11 |
| 14 | 12/10/11 | 12/08/11 - 12/14/11 | Week 49,Saturday | 12/04/11 - 12/10/11 |
| 15 | 12/11/11 | 12/08/11 - 12/14/11 | Week 50,Sunday | 12/11/11 - 12/17/11 |

**Display 3. Applying Date Range Labels for Non-Traditional Weeks**

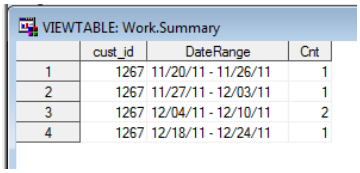
## FILLING IN WEEKS WITH SPARSE DATA

When reporting a week-by-week summary, you may find that you have no data for an entire week or more. To add in those weeks, you can make a "skeleton" table with to merge with your results. Then the missing weeks will be displayed and have zero or null values associated with them.

For example, I want to make a report for the number of transactions over the last 5 weeks for customer 1267:

```
DATA last5weeks; SET global_all;
    WHERE week(trans_date) >= week(today())-4;
    start = intnx('week',trans_date,0,'beginning');
    end = intnx('week',trans_date,0,'end');
    DateRange = catx(' - ', PUT(start,mmddyy8.),put(end,mmddyy8.));
    RUN;
PROC SQL;
    CREATE TABLE summary AS
    SELECT cust_id, daterange, count(*) as Cnt
    FROM last5weeks WHERE cust_id=1267
     GROUP BY cust_id, daterange ORDER BY cust_id, daterange;
    QUIT;
```

Output:


VIEWTABLE: Work.Summary

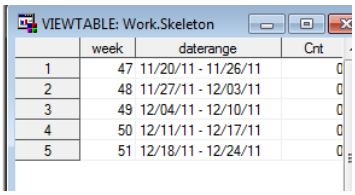
| | cust_id | DateRange | Cnt |
|---|---|---|---|
| 1 | 1267 | 11/20/11 - 11/26/11 | 1 |
| 2 | 1267 | 11/27/11 - 12/03/11 | 1 |
| 3 | 1267 | 12/04/11 - 12/10/11 | 2 |
| 4 | 1267 | 12/18/11 - 12/24/11 | 1 |

**Display 4. Five Week Summary Table with No Values for One Week**

As you can see there are only 4 weeks listed. One of the weeks had no observations and therefore does not show up. To solve this problem, I can create a skeleton table with all 5 weeks:

```
DATA skeleton;
start = INTNX('week',today(),0,'beginning'); /*returns a julian date value*/
week = week(today())-4;
DO i=(start-28) TO start BY 7; /*There are 28 days in 4 weeks, increment by 7 so there is one
value for each week*/
    DateRange = catx(' - ',PUT(i,mmddyy8.),PUT(i+6,mmddyy8.));
    Calls=0;
    Errors=0;
    OUTPUT;
    week+1;
END;
DROP start i;
RUN;
```

Output:


VIEWTABLE: Work.Skeleton

| | week | daterange | Cnt |
|---|---|---|---|
| 1 | 47 | 11/20/11 - 11/26/11 | 0 |
| 2 | 48 | 11/27/11 - 12/03/11 | 0 |
| 3 | 49 | 12/04/11 - 12/10/11 | 0 |
| 4 | 50 | 12/11/11 - 12/17/11 | 0 |
| 5 | 51 | 12/18/11 - 12/24/11 | 0 |

**Display 5. Five Week Summary Skeleton Table with Zero Placeholders**

Afterwards merge the skeleton dataset with the summary dataset. Be sure to place skeleton first in the merge statement so that data in the summary dataset will overwrite the skeleton null values. The resulting dataset has information for all weeks:

```
DATA summary2; MERGE skeleton summary;
   BY daterange;
```

```
RUN;
```

VIEWTABLE: Work.Summary2

| | week | daterange | Cnt |
|---|---|---|---|
| 1 | 47 | 11/20/11 - 11/26/11 | 1 |
| 2 | 48 | 11/27/11 - 12/03/11 | 1 |
| 3 | 49 | 12/04/11 - 12/10/11 | 2 |
| 4 | 50 | 12/11/11 - 12/17/11 | 0 |
| 5 | 51 | 12/18/11 - 12/24/11 | 1 |

**Display 6. Five Week Summary Table Merged with Skeleton Table to Add Missing Week**

## CONCLUSION

Armored with the right tools, defining weeks (even weeks that start on Thursday) becomes very quick and easy to do in SAS.

## REFERENCES

SAS V9.3 Help Documentation. Available online http://support.sas.com/software/93/index.html

SAS Online Doc V8, Chapter 3 Date Intervals, Formats, and Functions. Accessed online via http://www.okstate.edu/sas/v8/saspdf/ets/chap3.pdf

## CONTACT INFORMATION

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

Name: Barbara Ross
Enterprise: DataX Ltd
Address: 325 Warm Springs Rd Suite 200
City, State, ZIP: Las Vegas, NV 89119
Work Phone: 702-407-0707 ext: 8609
Email: bmharlan@gmail.com
Website: bharlan.weebly.com