Paper S126-2012

A SAS® Solution to Create a Weekly Format

Susan Bakken, Aimia, Plymouth, MN

ABSTRACT

As programmers, we are frequently asked to report by periods that do not necessarily correspond to weeks on the calendar – for example, a client may want to see sales by week since the day a product was released. This paper presents simple, easy-to-use, self-contained SAS code that utilizes date functions, macro variables, a DO/UNTIL loop, and the FORMAT procedure to create formatted values for any weekly periods needed, based on dates specified by the user. It also presents a second example of code that can be used with different start dates within the same data set.

INTRODUCTION

No matter the industry, SAS programmers generally do a lot of programming with dates. SAS has an extremely robust set of date formats and functions that can be used in many cases, but at times we need to be a bit clever to set up groupings that don't fit a standard week (or other period of time). The two SAS programs presented in this paper can be used to bucket your dates as needed, in order to quantify into groupings based on a specific starting date or dates.

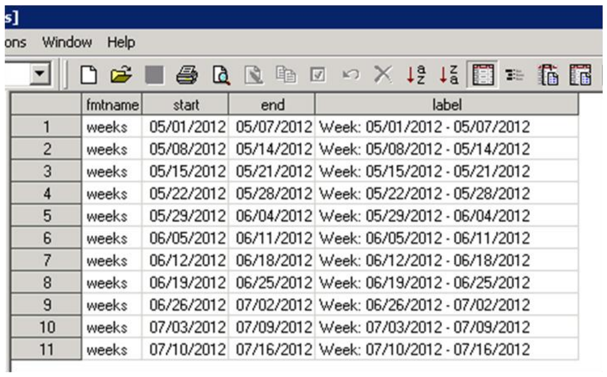
CODE FOR A SINGLE STARTING POINT

This code can be used if there is a single starting point for all observations in your data set. For example, the number of visits to a website after it went live. The only requirement to run the code is to let SAS know the start and end dates of the period of interest by populating them into macro variables:

```
%LET start='01May2012'd;
%LET end = date();
```

Now that the start and end dates of the period of interest have been set, we need to create a data set that will be used by PROC FORMAT. The following DATA step figures out the start and end dates of each 7-day period between the start and end dates set above. Note that the variables FMTNAME, START, END, and LABEL will all be used by PROC FORMAT to set the parameters of your format.

```
DATA SET_WEEKS;
 RETAIN fmtname 'weeks';
 start=&start;
 end=start+6;
 label="Week: "||PUT(start,mmddyy10.)||" - "||PUT(end,mmddyy10.);
 OUTPUT;
 DO UNTIL (end > &end);
  start=start+7;
  end=end+7;
  label="Week: "||PUT(start,mmddyy10.)||" - "||PUT(end,mmddyy10.);
  OUTPUT;
 END;
 KEEP fmtname start end label;
 FORMAT start end mmddyy10.;
RUN;
```



	fmtname	start	end	label
1	weeks	05/01/2012	05/07/2012	Week: 05/01/2012 - 05/07/2012
2	weeks	05/08/2012	05/14/2012	Week: 05/08/2012 - 05/14/2012
3	weeks	05/15/2012	05/21/2012	Week: 05/15/2012 - 05/21/2012
4	weeks	05/22/2012	05/28/2012	Week: 05/22/2012 - 05/28/2012
5	weeks	05/29/2012	06/04/2012	Week: 05/29/2012 - 06/04/2012
6	weeks	06/05/2012	06/11/2012	Week: 06/05/2012 - 06/11/2012
7	weeks	06/12/2012	06/18/2012	Week: 06/12/2012 - 06/18/2012
8	weeks	06/19/2012	06/25/2012	Week: 06/19/2012 - 06/25/2012
9	weeks	06/26/2012	07/02/2012	Week: 06/26/2012 - 07/02/2012
10	weeks	07/03/2012	07/09/2012	Week: 07/03/2012 - 07/09/2012
11	weeks	07/10/2012	07/16/2012	Week: 07/10/2012 - 07/16/2012

Figure 1.
Screenshot of data set set_weeks

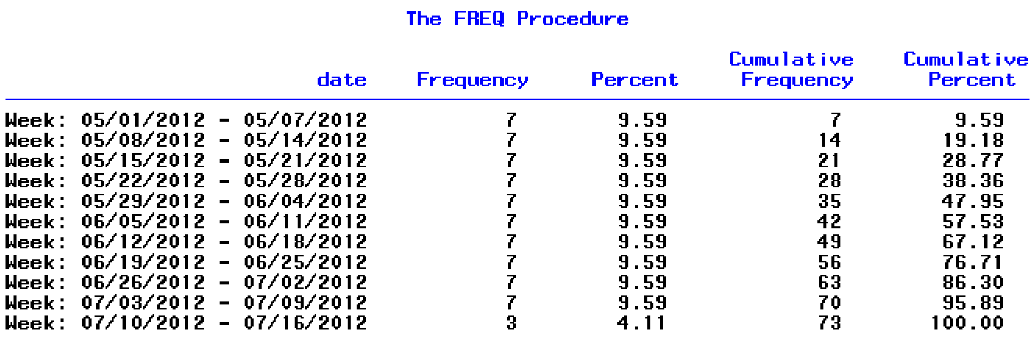
We can now use PROC FORMAT to create our date format:

```
PROC FORMAT CNTLIN=SET_WEEKS;
RUN;
```

We now have a format that we can use to group our dates by the periods we need. The following code shows an example of how the format can be used:

```
DATA test;
 DO date=&start TO &end;
  OUTPUT;
 END;
 KEEP date;
RUN;

PROC FREQ data=test;
 TABLES date;
 FORMAT date weeks.;
RUN;
```



```
                              The FREQ Procedure

                                                              Cumulative    Cumulative
                         date    Frequency     Percent     Frequency      Percent
----------------------------------------------------------------------------------
Week: 05/01/2012 - 05/07/2012           7        9.59             7         9.59
Week: 05/08/2012 - 05/14/2012           7        9.59            14        19.18
Week: 05/15/2012 - 05/21/2012           7        9.59            21        28.77
Week: 05/22/2012 - 05/28/2012           7        9.59            28        38.36
Week: 05/29/2012 - 06/04/2012           7        9.59            35        47.95
Week: 06/05/2012 - 06/11/2012           7        9.59            42        57.53
Week: 06/12/2012 - 06/18/2012           7        9.59            49        67.12
Week: 06/19/2012 - 06/25/2012           7        9.59            56        76.71
Week: 06/26/2012 - 07/02/2012           7        9.59            63        86.30
Week: 07/03/2012 - 07/09/2012           7        9.59            70        95.89
Week: 07/10/2012 - 07/16/2012           3        4.11            73       100.00
```

Figure 2.
Screenshot of output

CODE FOR MULTIPLE STARTING POINTS

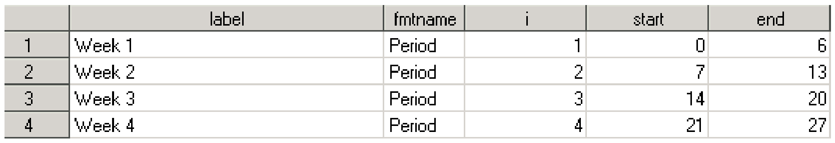
What if we have different starting dates for different key variables? In the following code, we want to identify sales by week once after a member enrolls in a loyalty program. Since each member could have a different enrollment date, we need to be able to calculate the sales by week for each individual based on when he/she enrolled.

The first step is to identify the period of interest, give it a name, and indicate how many periods we want to track:

```
%LET DaysInPeriod = 7;
%LET PeriodName = Week;
%LET TotalPeriods = 26;
```

The next step now uses those inputs to create a data set that will be used by PROC FORMAT. This time, instead of using a date range for the format parameters, we will be setting a range for each period based on the number of days from the starting period:

```
DATA DriveFormat;
 LENGTH label $50;
 RETAIN fmtname 'Period';
 i = 1;
 start = 0;
 end = &DaysInPeriod - 1;
 label = compbl("&PeriodName "||put(i,best.));
 OUTPUT;
 DO i = 2 to &TotalPeriods;
  start = end + 1;
  end = start + &DaysInPeriod - 1;
 label = compbl("&PeriodName "||put(i,best.));
 OUTPUT;
 END;
RUN;
```

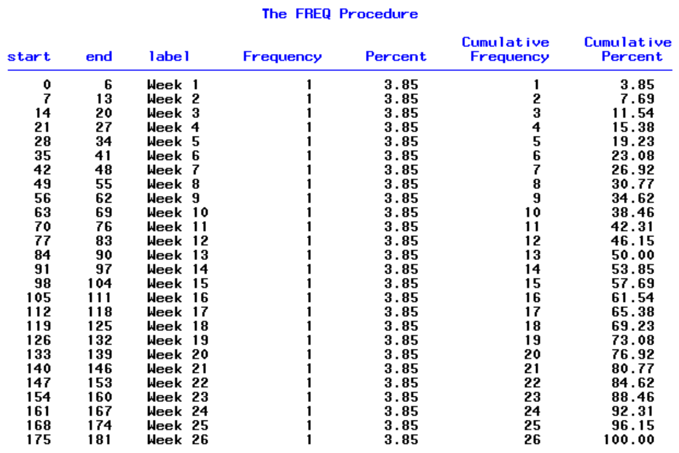


	label	fmtname	i	start	end
1	Week 1	Period	1	0	6
2	Week 2	Period	2	7	13
3	Week 3	Period	3	14	20
4	Week 4	Period	4	21	27

Figure 4.
Screenshot of data set DriveFormat

We can use the FREQUENCY procedure to make sure the code above set the periods of interest correctly:

```
PROC FREQ DATA=DriveFormat;
 TABLES start * end * label / LIST MISSING;
RUN;
```



```
                               The FREQ Procedure

                                                           Cumulative    Cumulative
start     end    label        Frequency     Percent     Frequency       Percent
--------------------------------------------------------------------------------
    0       6    Week 1               1        3.85              1          3.85
    7      13    Week 2               1        3.85              2          7.69
   14      20    Week 3               1        3.85              3         11.54
   21      27    Week 4               1        3.85              4         15.38
   28      34    Week 5               1        3.85              5         19.23
   35      41    Week 6               1        3.85              6         23.08
   42      48    Week 7               1        3.85              7         26.92
   49      55    Week 8               1        3.85              8         30.77
   56      62    Week 9               1        3.85              9         34.62
   63      69    Week 10              1        3.85             10         38.46
   70      76    Week 11              1        3.85             11         42.31
   77      83    Week 12              1        3.85             12         46.15
   84      90    Week 13              1        3.85             13         50.00
   91      97    Week 14              1        3.85             14         53.85
   98     104    Week 15              1        3.85             15         57.69
  105     111    Week 16              1        3.85             16         61.54
  112     118    Week 17              1        3.85             17         65.38
  119     125    Week 18              1        3.85             18         69.23
  126     132    Week 19              1        3.85             19         73.08
  133     139    Week 20              1        3.85             20         76.92
  140     146    Week 21              1        3.85             21         80.77
  147     153    Week 22              1        3.85             22         84.62
  154     160    Week 23              1        3.85             23         88.46
  161     167    Week 24              1        3.85             24         92.31
  168     174    Week 25              1        3.85             25         96.15
  175     181    Week 26              1        3.85             26        100.00
```

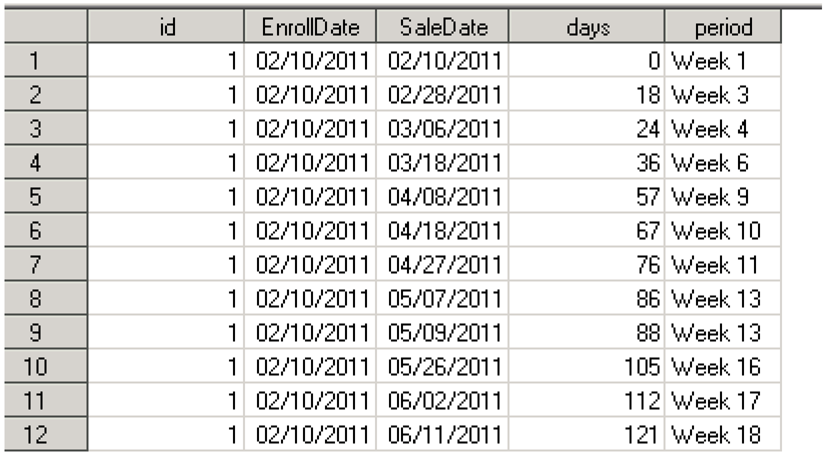
Figure 5.
Screenshot of output

We can now use PROC FORMAT to create our date format:

```
PROC FORMAT CNTLIN=DriveFormat;
RUN;
```

Here is an example of using the format. The data set SampleData contains 3 fields: ID, Enrollment Date, and Sales Date. The first thing we need to do is to calculate the number of days between each member's enrollment date and each sale. We will delete any records for sales that took place prior to enrollment, as we are only interested in behavior post-enrolment

```
DATA CalculateDays;
 SET SampleData;
  days = SaleDate - EnrolLDate;
  if days < 0 then delete;
  if days > (&DaysInPeriod * &TotalPeriods) - 1 then delete;
 period = PUT(days, period.);
RUN;
```

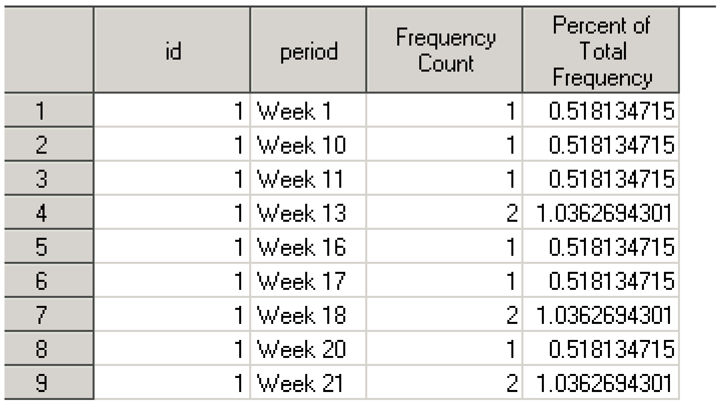


	id	EnrollDate	SaleDate	days	period
1	1	02/10/2011	02/10/2011	0	Week 1
2	1	02/10/2011	02/28/2011	18	Week 3
3	1	02/10/2011	03/06/2011	24	Week 4
4	1	02/10/2011	03/18/2011	36	Week 6
5	1	02/10/2011	04/08/2011	57	Week 9
6	1	02/10/2011	04/18/2011	67	Week 10
7	1	02/10/2011	04/27/2011	76	Week 11
8	1	02/10/2011	05/07/2011	86	Week 13
9	1	02/10/2011	05/09/2011	88	Week 13
10	1	02/10/2011	05/26/2011	105	Week 16
11	1	02/10/2011	06/02/2011	112	Week 17
12	1	02/10/2011	06/11/2011	121	Week 18

Figure 6.
Screenshot of data set CalculateDays

Now we use PROC FREQ to get the number of sales for each member in each period:

```
PROC FREQ DATA=CalculateDays noprint;
 TABLES id * period / LIST MISSING OUT=freqs;
RUN;
```

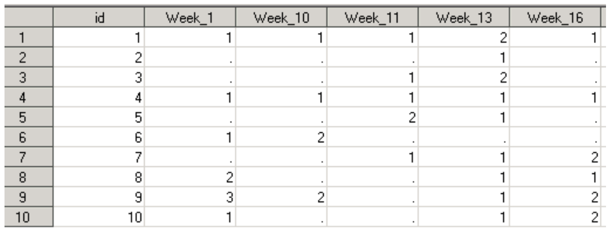


	id	period	Frequency Count	Percent of Total Frequency
1	1	Week 1	1	0.518134715
2	1	Week 10	1	0.518134715
3	1	Week 11	1	0.518134715
4	1	Week 13	2	1.0362694301
5	1	Week 16	1	0.518134715
6	1	Week 17	1	0.518134715
7	1	Week 18	2	1.0362694301
8	1	Week 20	1	0.518134715
9	1	Week 21	2	1.0362694301

Figure 7.
Screenshot of data set FREQS

In this example, we want each weekly period available as a variable, so we will use the TRANSPOSE procedure to create our weekly variables.

```
PROC TRANSPOSE DATA=freqs OUT=Transposed(DROP=_name_ _label_);
 BY id;
 VAR count;
 ID period;
RUN;
```

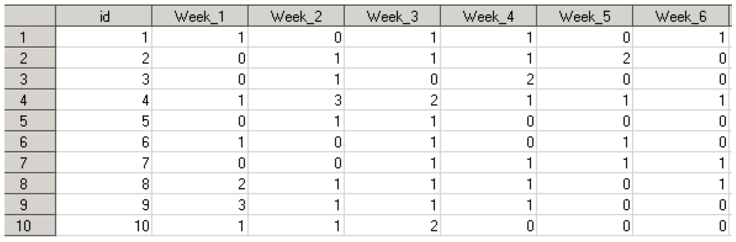


	id	Week_1	Week_10	Week_11	Week_13	Week_16
1	1	1	1	1	2	1
2	2	.	.	.	1	.
3	3	.	.	1	2	.
4	4	1	1	1	1	1
5	5	.	.	2	1	.
6	6	1	2	.	.	.
7	7	.	.	1	1	2
8	8	2	.	.	1	1
9	9	3	2	.	1	2
10	10	1	.	.	1	2

Figure 8.
Screenshot of data set Transposed

As shown in the example above, the weekly periods do not necessarily get created in the order we might want to have them stored in our data set. The following section of code shows how to re-order your variables and to set any missing weekly variables to zero:

```
DATA Ordered;
 LENGTH id &PeriodName._1 - &PeriodName._&TotalPeriods 8;
 SET transposed;
 ARRAY xvar{*} &PeriodName._1 - &PeriodName._&TotalPeriods;
 DO i = 1 TO dim(xvar);
  IF xvar{i} = . THEN xvar{i} = 0;
 END;
 DROP i;
RUN;
```

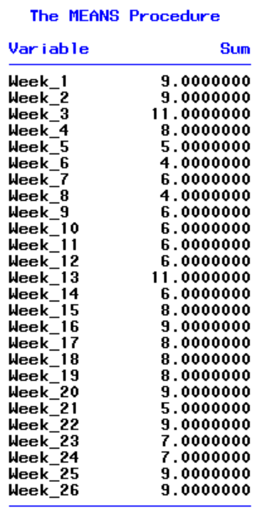


	id	Week_1	Week_2	Week_3	Week_4	Week_5	Week_6
1	1	1	0	1	1	0	1
2	2	0	1	1	1	2	0
3	3	0	1	0	2	0	0
4	4	1	3	2	1	1	1
5	5	0	1	1	0	0	0
6	6	1	0	1	0	1	0
7	7	0	0	1	1	1	1
8	8	2	1	1	1	0	1
9	9	3	1	1	1	0	0
10	10	1	1	2	0	0	0

Figure 9.
Screenshot of data set Ordered

Now we can do anything we want to with our new variables:

```
PROC MEANS DATA=ordered SUM;
 VAR &PeriodName._1 - &PeriodName._&TotalPeriods;
RUN;
```



```
The MEANS Procedure

Variable              Sum
Week_1          9.0000000
Week_2          9.0000000
Week_3         11.0000000
Week_4          8.0000000
Week_5          5.0000000
Week_6          4.0000000
Week_7          6.0000000
Week_8          4.0000000
Week_9          6.0000000
Week_10         6.0000000
Week_11         6.0000000
Week_12         6.0000000
Week_13        11.0000000
Week_14         6.0000000
Week_15         8.0000000
Week_16         9.0000000
Week_17         8.0000000
Week_18         8.0000000
Week_19         8.0000000
Week_20         9.0000000
Week_21         5.0000000
Week_22         9.0000000
Week_23         7.0000000
Week_24         7.0000000
Week_25         9.0000000
Week_26         9.0000000
```

Figure 10.
Screenshot of output

CAVEATS

The first code will group dates into formatted values that are each 7 days long, with the exception of the last formatted value. If the time period specified is not evenly divisible by 7, the final formatted value will contain all fewer than 7 days. This is important to remember to either clarify results or adjust time periods as necessary.

When using the second bit of code, be aware of any members in your dataset who may have a "master date" that is not far enough in the past to allow for each period's activity. For example, if you are looking for 8 weeks of sales post-enrollment, and one of your members enrolled 4 weeks ago, that person will show no sales for weeks 5-8, when in actuality weeks 4-8 have simply not happened yet for that individual. You may wish to account for this when setting up your dataset or add logic to this code to remove everyone that does not yet have the full period available.

CONCLUSION

In conclusion, some basic SAS statements, functions, formats, and procedures can allow you to bucket information into the date groupings you need. These pieces of code can be easily modified to adjust to many different unique situations.

REFERENCE – FIRST CODE IN ITS ENTIRITY FOR EASY COPY/PASTE

Here is the entire first program for anyone who wants do copy, paste, and use!

```
*********************************************************************************;
** start date will be used as day 1 of week 1 - all weeks will be 7-day weeks   **;
** that start on the same day of the week as this date                          **;
*********************************************************************************;
%LET start='01May2012'd;
%LET end = date();

*********************************************************************************;
** create a dataset that will drive proc format - note, this will stop once it  **;
** reaches a date greater than the end date specified                           **;
*********************************************************************************;
DATA SET_WEEKS;
 RETAIN fmtname 'weeks';
 start=&start;
 end=start+6;
 label="Week: "||PUT(start,mmddyy10.)||" - "||PUT(end,mmddyy10.);
 OUTPUT;
 DO UNTIL (end > &end);
  start=start+7;
  end=end+7;
  label="Week: "||PUT(start,mmddyy10.)||" - "||PUT(end,mmddyy10.);
  OUTPUT;
 END;
 KEEP fmtname start end label;
 FORMAT start end mmddyy10.;
RUN;

*********************************************************************************;
** use proc format to create the format to the work folder - the format's name  **;
** is weeks.                                                                   **;
*********************************************************************************;
PROC FORMAT CNTLIN=SET_WEEKS;
RUN;

*********************************************************************************;
** this section just a sample showing how the format can now be used           **;
*********************************************************************************;
DATA test;
 DO date=&start TO &end;
  OUTPUT;
 END;
 KEEP date;
RUN;

PROC FREQ data=test;
 TABLES date;
 FORMAT date weeks.;
RUN;

*********************************************************************************;
** end of program                                                              **;
*********************************************************************************;
```

REFERENCE – SECOND CODE IN ITS ENTIRITY FOR EASY COPY/PASTE

Here is the entire first program for anyone who wants do copy, paste, modify, and use!

```
*********************************************************************************;
** set macro variables                                                         **;
*********************************************************************************;
   %LET DaysInPeriod = 7;
   %LET PeriodName = Week;
   %LET TotalPeriods = 26;

*********************************************************************************;
** this step creates the dataset that will drive PROC format                   **;
*********************************************************************************;
DATA DriveFormat;
 LENGTH label $50;
 RETAIN fmtname 'Period';
 i = 1;
 start = 0;
 end = &DaysInPeriod - 1;
 label = compbl("&PeriodName "||put(i,best.));
 OUTPUT;
 DO i = 2 to &TotalPeriods;
  start = end + 1;
  end = start + &DaysInPeriod - 1;
 label = compbl("&PeriodName "||put(i,best.));
 OUTPUT;
 END;
RUN;

*********************************************************************************;
** frequency to check our periods and make sure they calculated properly       **;
*********************************************************************************;
PROC FREQ DATA=DriveFormat;
 TABLES start * end * label / LIST MISSING;
RUN;

*********************************************************************************;
** create the format                                                           **;
*********************************************************************************;
PROC FORMAT CNTLIN=DriveFormat;
RUN;

*********************************************************************************;
** the dataset SAMPLEDATA contains 3 fields - ID, enrollment date, sale date    **;
** - calculate the days between enrollment date and each sale date, and        **;
**   remove any sales that took place before enrollment                        **;
*********************************************************************************;
DATA CalculateDays;
 SET SampleData;
  days = SaleDate - EnrolLDate;
  if days < 0 then delete;
  if days > (&DaysInPeriod * &TotalPeriods) - 1 then delete;
 period = PUT(days, period.);
RUN;

*********************************************************************************;
** count the number of sales for each member, grouping by our formatted value   **;
*********************************************************************************;
PROC FREQ DATA=CalculateDays noprint;
 TABLES id * period / LIST MISSING OUT=freqs;
RUN;

*********************************************************************************;
** now we can use PROC transpose to have a separate variable for each period    **;
*********************************************************************************;
PROC TRANSPOSE DATA=freqs OUT=Transposed(DROP=_name_ _label_);
 BY id;
 VAR count;
 ID period;
RUN;

*********************************************************************************;
** this step shows how to use our macro variables to order the new variables    **;
*********************************************************************************;
DATA Ordered;
 LENGTH id &PeriodName._1 - &PeriodName._&TotalPeriods 8;
 SET transposed;
 ARRAY xvar{*} &PeriodName._1 - &PeriodName._&TotalPeriods;
 DO i = 1 TO dim(xvar);
  IF xvar{i} = . THEN xvar{i} = 0;
 END;
 DROP i;
RUN;
```

```
*********************************************************************************;
** now we can use all of our weekly variables however we want                 **;
*********************************************************************************;
PROC MEANS DATA=ordered SUM;
 VAR &PeriodName._1 - &PeriodName._&TotalPeriods;
RUN;

********************************************************************************;
** end of program                                                             **;
********************************************************************************;
```

CONTACT INFORMATION

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

Name: Susan Bakken
Enterprise: Aimia
Address: 1405 Xenium Lane
City, State ZIP: Plymouth, MN 55441
Work Phone: 763.445.3619
E-mail: susan.bakken@aimia.com