Create a Report Showing Monthly, Quarter-To-Date and Year-To-Date Information Without Changing Date Parameters Monthly

Rodger Zhang, TD Canada Trust, Toronto, Ontario, Canada

ABSTRACT

Tracking sales force’s performance is a common task in an organization. It’s easy to create a report showing monthly data only. However, you may also be asked to create a report to track the monthly, quarter-to-date (QTD) and year-to-date (YTD) performance. As different organizations’ fiscal year vary, how would you do this without changing the date parameters every month when you run the report? SAS® offers many ways to accomplish this task. This paper will focus on how to automatically define the 12 months and their corresponding 4 quarters. SAS functions, the Call Symput statement, DO Loop and a few tricky programming tips and techniques are illustrated. The code demonstrated in this paper can be used as the template to define any fiscal year’s 12 months and corresponding 4 quarters. Base SAS is used and the examples shown are sample data only. The paper is for beginner or intermediate SAS programmers.

INTRODUCTION

Assume that a financial institute starts its fiscal year in Nov. It has been decided that in the new fiscal year, a report will be created to track the sales progress of each branch on a monthly basis. You are given the monthly plan numbers for each branch and are asked to create a report to track the sales performance of the reporting month, the current quarter and year-to-date summary on the fifth business day of the next month after the reporting month ends.

The following data are simulated data for a financial institute. They include the monthly plan numbers of 12 months pre-set for each branch. Each record has 13 variables containing branch_no, plan1stm, plan2ndm, … plan12thm as follows (sample):

```
1111 10 20 30 20 10 3 5 8 90 10 30
2222 100 20 300 20 40 10 3 5 80 90 10 300
```

Starting from the second month (which is Dec in this case), you are going to extract the actual sales from the system and create a report showing the plan numbers and actual sales of the reporting month, current quarter and year-to-date total from starting month of the fiscal year (in your situation here, you run the report on Aug 5th for the month of July). The report will look like:

```
<table>
<thead>
<tr>
<th>Branch</th>
<th>Reporting Month</th>
<th>Plan Month</th>
<th>Actual Month</th>
<th>Plan QTD</th>
<th>Actual QTD</th>
<th>Plan YTD</th>
<th>Actual YTD</th>
</tr>
</thead>
<tbody>
<tr>
<td>1111</td>
<td>200607</td>
<td>8</td>
<td>8</td>
<td>16</td>
<td>17</td>
<td>146</td>
<td>144</td>
</tr>
<tr>
<td>2222</td>
<td>200607</td>
<td>80</td>
<td>75</td>
<td>88</td>
<td>83</td>
<td>578</td>
<td>584</td>
</tr>
</tbody>
</table>
```

Since the organization starts the fiscal year in Nov, when you run the report in Aug, the reporting month will be Jul (which is the month before you run the report), and the quarter-to-date (QTD) will be the total of May, Jun and Jul (3rd Quarter) while the year-to-date (YTD) will always be the total of all the months as of the reporting month starting from Nov. There are 12 months and 4 quarters in a year. Once the first month of the fiscal year is known, then you know how to define the 12 months and the corresponding quarters. As mentioned above, the reporting month will always be the month before you run the report. QTD will be the reporting month or the summary of reporting month and the previous month or previous two months depending on if the reporting month is the beginning, middle or end of the quarter and the YTD the summary of all the previous months before the month you run the report.

DEFINING MACRO VARIABLES

As different organizations may start their fiscal year differently, the first step you need to do is to create one global macro variable specifying the start date your organization starts the fiscal year:

```
%let fiscalyear='01nov2005'd;
```

Then you use the SAS date interval function INTNX to define the 12 months and their corresponding quarters:

```
DATA _null_;
The following points need to be noticed from the code above:

1. A SAS function performs a computation or system manipulation on arguments and returns a value. Most functions use arguments supplied by the user, but a few obtain their arguments from the operating environment such as TODAY() etc. TODAY() produces the current date in the form of a SAS date value, which is the number of days since January 1, 1960. It is identical to the DATE function. MONTH function returns a numeric value ranging from 1 to 12 from a SAS date value while YEAR function returns the year from a SAS date value. The YEAR function produces a four-digit numeric value that represents the year.

2. Manipulation and calculation of SAS date value: since SAS date values are numeric values, they can be used in calculations. See examples below in INTNX function section.

3. Call SYMPUT: The SYMPUT routine allows you to create macro variables or assign a new value to an existing one. More than 20 macro variables have been created here.

4. INTNX function: Syntax as follows:

   \[
   \text{INTNX(interval, from, n <, alignment >)}
   \]

   INTNX advances a date, time, or datetime value by a given interval, and returns a date, time, or datetime value, for example, if you submit the following code in Aug 2006: \text{intnx('month', today(),-1)} will return a SAS date which is one month before the running date 07/01/2006; \text{intnx('month', today(),0)} will return 08/01/2006 and \text{intnx('month', today(),+1)} 09/01/2006. A SAS date can be put into calculations, so once you have the return value, you can use SAS date function e.g YEAR, MONTH, etc.
to obtain the value you need. For example, `year(intnx('month',today(),-2))*100+month(intnx('month', today(),-2);` will return “200606”.

5. **NULL** Statement: Your purpose is to create macro variables so that you can reference them later in your code, so you just want to execute a DATA step but do not want to create a SAS data set.

Once you define 12 months and their corresponding 4 quarters, you can go ahead to generate your monthly report with monthly, QTD and YTD summaries.

You should know the plan numbers for the coming year before you start the reporting. However, the actual sales number won’t be available until the reporting month is over. Therefore two different ways will be provided to get the monthly, QTD and YTD data.

### PLAN QTD AND YTD MANIPULATION

For plans, you read the raw data first:

```sas
data planraw;
  input branch plan&mth1 plan&mth2 plan&mth3 plan&mth4 plan&mth5 plan&mth6 plan&mth7 plan&mth8 plan&mth9 plan&mth10 plan&mth11 plan&mth12;
datalines;
1111 10 20 30 20 40 10 3 5 8 90 10 30
2222 100 20 300 20 40 10 3 5 80 90 10 300
; run;
```

Then you manipulate the data set to get the data for reporting month, QTD and YTD summaries depending on the month to be reported on.

```sas
data plan;
  set planraw;
  plan=plan&rptm;
  if &curmonth eq &sttmonth then do;
    plan&rptmpr=0;
    plan&rptmpr2=0;
  end;
  else if &curmonth eq &sttmonthaf then do;
    plan&rptmpr=plan&dummy1;
    plan&rptmpr2=0;
  end;
  else do;
    plan&rptmpr=plan&dummy1;
    plan&rptmpr2=plan&dummy2;
  end;
  if index("&qtrstart","&curmonth")>0 then do;
    planQTD=plan;
  end;
  else if index("&qtrmid","&curmonth")>0 then do;
    planQTD=plan+plan&rptmpr;
  end;
  else if index("&qtrend","&curmonth")>0 then do;
    planQTD=plan+plan&rptmpr+plan&rptmpr2;
  end;
  planYTD=sum(OF plan&mth1 -- plan&rptm);
  keep branch plan planQTD planYTD;
run;
```

In the code above, you specified that QTD number is the number of the reporting month or the number of the reporting month plus its previous months. You use the INDEX function to check the value of reporting month against the macro
variables you created for quarter start months, quarter mid months or quarter end months. Macro variable &qtrstart represents beginning months of each quarter. It includes 4 months’ value and leading zero is included. For example, the first quarter of a normal calendar year is month 01, 02 and 03 and the second quarter is 04, 05, 06 and so on. So the beginning months of each quarter would be 01-04-07-10, middle months represented by macro variable &qtrmid would be 02-05-08-11 and ending months represented by macro variable &qtrend would be 03-06-09-12. Keep in mind that when you run the first quarter’s report, QTD and YTD numbers will be exactly the same. That’s why you set the value of the previous month and the month before the previous month to 0 when you run the first report and set the value of the month before the previous month to 0 when you run the second report. As for YTD summary, it’s always the total of all the months as of reporting month, which is accomplished by using the SUM function.

VERTICAL DATA QTD AND YTD

It’s a different story for actual sales number. They will only be available when the reporting month is over. Each month (from the second month), you will need to extract the sales data first:

```plaintext
data actualraw&rptm;
   input branch account actual;
datalines;
1111 5234098 1
1111 7320455 2
1111 8421547 5
2222 5225784 10
2222 5324710 65
;
run;
```

Since the data you extract is at an account level, you need to roll them up to a branch level. PROC SQL is used here:

```plaintext
proc sql;
create table actualsum&rptm as
select branch,
   sum(actual) as actual,
   sum(actual)  as actual&rptm
from actualraw&rptm
group by branch;
```

As this is the monthly data, going forward, you need to summarize monthly data to get the QTD and YTD information. What you need to do is to create a permanent dummy data set using the month before the first reporting month as its name suffix. This means YTD list as of reporting month will be the YTD list as of previous month plus the current or reporting month. In the dummy data set, all the variables value will be set to 0 except branch that has to be any one of the real branches. That’s why you use one of the branches from the plan data set:

```plaintext
libname ytd 'c:\sales report\';
data ytd.ytdlist&base;
   set plan(keep=branch) end=last;
   if last;
      actual=0;
      actual&mth1=0;  actual&mth2=0 ;
      /* ...(more code for rest of &mth)... */
      actual&mth11=0 ;actual&mth12=0 ;
      
   run;
```

```plaintext
data ytd.ytdlist&rptm;
   set ytd.ytdlist&rptmpr actualsum&rptm;
   if actual&mth1 =. then actual&mth1 = 0;
   /*...(more code for rest of &mth)...*/
   if actual&mth12 =. then actual&mth12 = 0;
run;
```

The YTD list as of reporting month includes 14 variables containing branch_no, actual1stm, actual2ndm, ... actual12thm and actual that equals to that of reporting month. The following code will create the YTD data:
PROC SQL;
create table sum as
select branch,
sum(actual) as actualYTD,
sum(actual&mth1 ) as actual&mth1,
/*...more code for rest of &mth sum...*/
sum(actual&mth12) as actual&mth12
from ytd.ytdlist&rptm
group by branch;

Now you should have the same data layout as that of the plan and you can use the same logic to get the monthly and QTD data as follows:

data actual;
set sum;
actual=actual&rptm;
/*(to create variable actualQTD here, using the exactly the same logic as for that of the plan, but change the "plan" to "actual", refer to code above)*/
....*/
Run;

COMBINE PLAN AND ACTUAL DATA

Finally you can combine plan and actual data to get the final result:

data final;
merge plan actual;
by branch;
reporting_period=&rptm;
keep reporting_period branch
actual plan actualQTD
planQTD actualYTD planYTD;
run;

CONCLUSION

Monthly data are easy to report, but reporting QTD and YTD information automatically may be a bit frustrating as different organizations start their fiscal year differently. By using SAS functions, CALL Symput routines, and plus some programming techniques, you may generate the report showing monthly, QTD and YTD information at ease once you know when the first month of the fiscal year starts. You don’t need to change any date parameters monthly when you run the monthly report after the end of the reporting month. Many date value calculations are involved in the codes and the codes can be used as the template to define any fiscal year’s 12 months and corresponding 4 quarters.

ACKNOWLEDGMENTS

The author would like to thank Business & Information Senior Manager Mr. Joseph Virey for his support. Special thanks go to Louie Luo and Isabel Huang for their excellent suggestion and help in proofreading this paper.

REFERENCES

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.

Other brands and product names are trademarks of their respective companies.

CONTACT INFORMATION

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

Rodger Zhang
TD Canada Trust
Toronto, Ontario
416 307 6056
email: rodger.zhang@td.com