SESUG Paper 222-2019

Which report is appropriate? Let SAS figure it out!

Smith, Kelly D., Central Piedmont Community College

ABSTRACT

Central Piedmont Community College (CPCC) uses a series of progression and completion metrics to track multiple cohorts of new part-time and new full-time students through their first three years at the institution. Progression and completion metrics from matching cohorts in sequential years are also compared to determine if interventions are having a positive impact on student success.

As cohorts progress through the three year cycle, data from new metrics becomes available and a fuller picture of the cohort emerges. The frequency and complexity of the reporting process encouraged us to search for ways to minimize errors that can occur through the updating of numerous macrovariables and to find a better method to determine which report was appropriate for each cohort at specific points in time.

To minimize data entry errors, a series of macro variables have been defined using macro functions so that only an initial macro variable is updated by the user. To maximize efficiency and ease of use, macro functions and conditional processing within a macro identify which metrics and related output are appropriate based on a comparison of the current date to the starting point as defined by the user in the initial macro variable. The discussion includes macro and system functions such as %SYSFUNC, %EVAL, and %SUBSTR, in addition to the conditional processing functions %IF, %THEN DO, %ELSE %IF, and %END.

INTRODUCTION

In an ongoing effort to improve curriculum students' outcomes, Central Piedmont has been examining the academic progression of new students across their first three years at the college. Each fall, a cohort of students new to CPCC is established and demographic information is used to sort the students into twenty-one population groups. The academic progression and three year outcomes of each population group is tracked with fifteen metrics. The data obtained has provided insight not only into the student experience at CPCC but has also provided a lens to examine student equity at the college.

The first cohorts were established with historical student data which meant three full years of progression and outcome data were immediately available. However, three years of progression and outcome metrics are not available for cohorts established after 2016. A need for additional student information became apparent after the first few report cycles. As a consequence, the SAS code used to extract, sort, and analyze student data has transformed to accommodate ongoing project changes. Three major steps have been taken to increase the code's flexibility: (1) the number of macrovariables that require updating have been reduced through the use of building block macro variables and macro functions; (2) the original code has been split into four report macros based on metric timing; and (3) the code determines which metrics can be reported out based on an comparison of the cohort's start term and the current date.

MACRO VARIABLES & MACRO FUNCTIONS

Many of the codes used at CPCC contain specific term names throughout a program, making it difficult to update the code manually. Macro variables facilitate code updates by substituting specific text in selected spots throughout the code. Some macro variables are automatic, but macro variables can also be created with %LET or the CALL SYMPUT procedures. The majority of code written at CPCC starts with a series of macro variables. Macro functions can be used to manipulate and create text strings; macro functions can thereby work on macro variables, which consist of text strings (Carpenter, 2000; Stroupe, 2003).

The code used to track student progression and outcome pulls data from up to nine academic terms over three academic years. Each academic term identifier consists of the four digit year plus two digits that identify the time of year for the term (01=Spring, 02=Summer, 03=Fall). For the fall 2015 cohort, data is pulled from four calendar years (2015-2018). The list of nine original macro variables reads as follows:

```
%LET yt01=201503;     /* Year One */
%LET yt02=201601;
%LET yt03=201602;
%LET yt04=201603;     /* Year Two */
%LET yt05=201701;
%LET yt06=201702;
%LET yt07=201703;     /* Year Three */
%LET yt08=201801;
%LET yt09=201802;
```

Since manually updating numerous macro variables increases the chance of mistakes, macro functions are one way to simplify the updating process. Updating the code is also simplified by using building block macro variables to construct other macro variables. If additional text needs to be combined with text generated by a macro variable, the end point of the macro variable name should be identified with a delimiter such as a period or a space. Table 1 demonstrates how a number of macro variables can be built from one line of code, %LET yr01 = 2015. Updating one macro variable (yr01) automatically updates the macro variables yt01 – yt09.

Macro Code (%LET =)	**Resolved Macro**
yr02 = %EVAL(&yr01+1);	yr02 = 2016
yr03 = %EVAL(&yr01+2);	yr03 = 2017
yr04 = %EVAL(&yr01+3);	yr04 = 2018
yt01 = &yr01.03;	yt01 = 201503
yt01b = Fall &yr01	yt01b = Fall 2015
ft01= &yr01.CU3;	ft01 = 2015CU3
fy02 = &yr01&yr02;	fy02 = 20152016
yt07 = &yr03.03;	yt07 = 201703
yt08 = &yr04.01;	yt08 = 201801
yt09 = &yr04.02;	yt09 = 201802

Table 1. Using Macro Variables and Macro Functions to Build New Macro Variables

Using building block macro variables and macro functions not only simplifies the process for updating code but also improves quality assurance by limiting the number of manual changes made to the code while updating.

REPORT MACROS

The thirteen metrics tracked in the project are presented in Table 2. Three of the metrics (starred) check for a specific number of completed credits within a predetermined time frame. Although a large percentage of CPCC students attend school on a part-time basis, early project reports did not differentiate between part- and full- time students. All students were evaluated based on full-time credit goals.

Time Frame	Metric Name	Metric Definition
First Term	Zero Credits	Withdrew and/or failed all courses in first term
	Successful Completion 66% Credit Hours	Completed at least 66% of credit hours with A, B, C, or P
	12 or More Credits Earned *	Completed at least 12 credit hours with A, B, C, D, or P
First Year	Return First Spring	Present as FTE eligible student in first Spring term
	Successful Completion College English	Attempted and Completed college level English with A, B, or C
	Successful Completion College Math	Attempted and Completed college level Math with A, B, or C
	24 or More Credits Earned *	Completed at least 24 credit hours with A, B, C, D, or P
Second Year	Return Second Fall	Present as FTE eligible student in second Fall term
	48 or More Credits Earned *	Completed at least 48 credit hours with A, B, C, D, or P
Third Year	Three Year Outcome	
	Credential and Transfer	Credential and Transfer to 4 year institution within 3 years
	Credential, no Transfer	Credential within 3 years, but no Transfer to 4 year institution
	Transfer and no Credential	No Credential, but Transfer to 4 year institution within 3 years
	Enrolled Year 3	Enrolled in Fall, Spring, or Summer Term of Year 3
	Credentialed, Transferred or Enrolled	Sum of four previous categories
	Outcome Unknown	Not credentialed, transferred, or enrolled in third year

Note. Credential = Associate Degree, Certificate, or Diploma

Table 2. Thirteen Metrics used to Track Student Progression and Outcome

As the project continued to evolve, a request was made to differentiate between part- and full-time students. In order to accurately and efficiently create reports for both part- and full-time students while accounting for their respective credit goals, the code has been split into a series of four report macros based on the metrics' appropriate time frame (first term, first year, second year, third year). Each macro has the same five parameters (Table 3) for consistency. With the use of parameters, the same code can be used for each type of student. Each macro executes twice, once for part-time and once for full-time cohort students.

Parameter	Definition	Part-Time	Full-Time
ftpt00	Student status code	PT	FT
ftpt01	Student status name	Part-Time	Full-Time
fttm	First term credit goal	6	12
ftyr	First year credit goal	12	24
ftyr2	Second year credit goal	24	48

Table 3. Values of Report Macro Parameters

Splitting the code into a series of four parameter-based macros increased the flexibility and usability of the code. The combination of report macros with the use of macro functions and building block macro variables made the code more user-friendly and resulted in more actionable information in the reports.

ANALYZING TERMS AND DATES

The initial code tracked student progression and outcome across three full academic years; the requirement for three full years of data restricted reporting to older cohorts. Recently, data for all fall cohorts was requested, even if only a small part of the data was available. Originally, cohort data was updated at the end of the academic year. Currently, cohort data is updated at the end of each fall term and at the end of each summer term to provide more timely reports. The additional data extraction at the end of the fall term permitted the second year report macro to be split into two parts: fall to fall retention is reported at the end of the second fall term while second year credits are reported at the end of the second academic year.

In order to determine which reports are available for each cohort, the start term for the cohort is compared to the date of code execution. The automatic macro variable, &SYSDATE9, is used to establish the current date (DDMMMYYYY). The year portion of the current date is extracted with the %SUBSTR macro function. The analysis is performed within a macro through the use of macro functions for conditional processing through a logic tree (Figure 1):

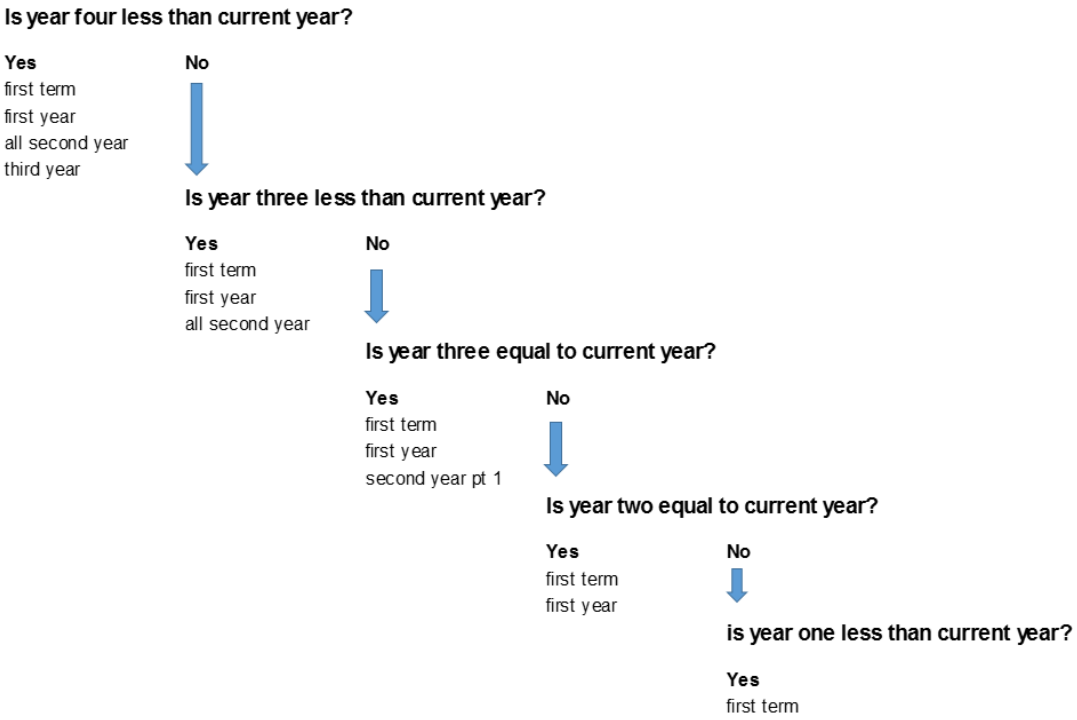

Figure 1. Logic Tree for Selecting Cohort Reports

Initially, the term analysis macro was situated at the bottom of the code and included macro execution calls for the appropriate report macros. A portion of the code follows:

```
%LET today=&SYSDATE9;
%LET now=%SUBSTR(&today,6,4);

%MACRO TermAnalysis;

%IF &yr04 LT &now %THEN %DO;
   %FirstTerm(PT,Part-Time,6,12,24)
   %FirstTerm(FT,Full-Time,12,24,48)
   %FirstYear(PT,Part-Time,6,12,24)
   %FirstYear(FT,Full-Time,12,24,48)
   %SecondYear2A(PT,Part-Time,6,12,24) /* F2F Retention */
   %SecondYear2A(FT,Full-Time,12,24,48)
   %SecondYear2B(PT,Part-Time,6,12,24) /* 2 Year Credits */
   %SecondYear2B(FT,Full-Time,12,24,48)
   %ThirdYear(PT,Part-Time,6,12,24)
   %ThirdYear(FT,Full-Time,12,24,48)
%END;
   %ELSE %IF &yr03 EQ &now %THEN %DO;
         %FirstTerm(PT,Part-Time,6,12,24)
         %FirstTerm(FT,Full-Time,12,24,48)
         %FirstYear(PT,Part-Time,6,12,24)
         %FirstYear(FT,Full-Time,12,24,48)
         %SecondYear2A(PT,Part-Time,6,12,24) /* F2F Retention */
         %SecondYear2A(FT,Full-Time,12,24,48)
         %END;
```

Since the term analysis macro was located at the bottom of the code, all of the code had to be processed, even if the only report to be generated was the first term report. To improve code efficiency, the code has been split into six distinct and separate sections of code: cohort establishment, term analysis, first term report, first year report, second year report, and third year report. The second year report code can generate one (2A) or both (2A, 2B) second year reports based on the logic tree (Figure 1). The term analysis code uses the macro function %INCLUDE to pull in the appropriate report codes based on the logic tree; %INCLUDE is directed to the location of a report code through the use of a FILENAME statement. The entire current term analysis code follows:

```
FILENAME RPT "C:\Users\SKD4306E\Desktop\2019 Conference\SASMacro";

%MACRO ANALYSIS;

%IF &yr04 LT &now %THEN %DO;
   %INCLUDE RPT(First_Term_Report);
   %INCLUDE RPT(First_Year_Report);
   %INCLUDE RPT(Second_Year_Report);
   %INCLUDE RPT(Third_Year_Report);
%END;
   %ELSE %IF &yr03 LE &now %THEN %DO;
         %INCLUDE RPT(First_Term_Report);
         %INCLUDE RPT(First_Year_Report);
         %INCLUDE RPT(Second_Year_Report);
         %END;
   %ELSE %IF &yr02 EQ &now %THEN %DO;
         %INCLUDE RPT(First_Term_Report);
         %INCLUDE RPT(First_Year_Report);
         %END;
   %ELSE %IF &yr01 LT &now %THEN %DO;
         %INCLUDE RPT(First_Term_Report);
         %END;

%MEND;
```

Using the updated term analysis code has improved the efficiency of the reporting process; however, the process still requires the regeneration of previous reports. Further code modifications are underway to eliminate the redundancy and to append new data to the previous reports.

CONCLUSION

CPCC first started tracking new fall cohorts in late 2017. The initial approach to generating cohort reports required three years of progression and outcome metrics, limiting the actionable nature of the information produced. Over the past two years, the code has evolved as the project has evolved. The use of macro variables, macro functions, and macro programs has improved the flexibility, usability, and efficiency of the code.

REFERENCES

Stroupe, J. (2003). Nine steps to get started using SAS® Macros. *Proceedings of the Pharmaceutical Software Users Group 2003*. Available at https://support.sas.com/resources/papers/proceedings/proceedings/sugi28/056-28.pdf

Carpenter, A. L. (2000). Using macro functions. *Proceedings of the SAS Users Group International 2000*. Available at https://support.sas.com/resources/papers/proceedings/proceedings/sugi25/25/aa/25p004.pdf

CONTACT INFORMATION

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

Kelly D. Smith, Ed.D.
Research Analyst, Sr.
Central Piedmont Community College
704-330-2022 x3810
kelly.smith@cpcc.edu