sas innovate 2025

More Muggles, More Macros

Adding Advanced Data-Driven Wizardry to Your SAS® Programs

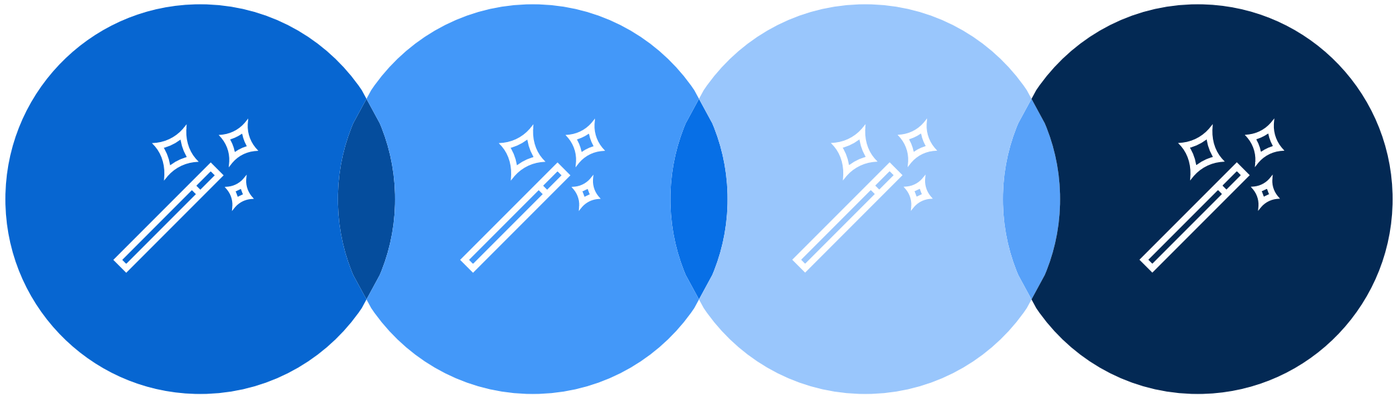
Josh Horstman, Nested Loop Consulting
Richann Watson, DataRich Consulting

Introduction

- Static “muggle” code is full of hardcodes and data dependencies
  - Not flexible: Breaks easily when unexpected inputs or conditions are present
  - Difficult to maintain: Modifications needed when data or environment changes
  - Difficult to reuse: Modifications needed to use for another project

- Macro Language “magic” can eliminate these problems!
  - Dynamic: Code automatically adapts to changing inputs and conditions
  - Data-Driven: Programming logic is controlled by the data and requires little maintenance
  - Reusable: Code can easily be used in a variety of situations with little to no modification

Overview



1 From Muggles to Macros
Recap of the First Spellbook

2 Control Tables Example
Spell #1

3 Call Execute Example
Spell #2

4 Resolve Example
Spell #3

From Muggle to Macros

Recap of the First Spellbook

sas innovate 2025

2024 Presentation

- “From Muggles to Macros: Transfiguring Your SAS Programs With Dynamic, Data-Driven Wizardry”

- Available as a SAS “Ask the Expert” Webinar
  - Watch on demand
  - Downloadable slides

https://www.sas.com/en_us/webinars/from-muggles-to-macros.html

Recap: Macro Processing Overview

- When a SAS program is submitted:
  - Word scanner parses statements into tokens.
  - Tokens are sent to compiler for syntax checking.
  - Execution occurs when step boundary is reached.
- If the word scanner detects macro triggers (% or &):
  - Macro elements routed to macro processor.
  - Macro variables resolved and macro statements executed.
  - Output from macro processor must be rescanned for additional macro language elements.

Recap: The Macro Variable List

- Macro Variable List – a series of macro variables, each storing one value
- Named with a common prefix and sequential suffix to enable processing in a loop
- Example: A macro variable list containing the unique values of the ORIGIN variable from the SASHELP.CARS dataset

```
%let origin1 = Asia;
%let origin2 = Europe;
%let origin3 = USA;
```

- But we want to create these dynamically, not by hard-coding!
- Must be created at execution time to have access to data values.

Recap: Using Macro Variable Lists

- Access individual list elements using macro variable reference:

&origin1 → Resolves to: **Asia**
&origin2 → Resolves to: **Europe**
&origin3 → Resolves to: **USA**

Cannot do &origin&i – macro processor interprets this as two separate macro variable references

- To access items from a macro variable list, use double ampersand:

```
%do i = 1 %to &numorigins;
  %put Item &i: &&origin&i;
%end;
```

Original: **&&origin&i**
1st pass: **&origin1** (&& resolves to &, origin is just text, &i resolves to 1)
2nd pass: **Asia** (resolved value of macro variable origin1)

Control Table Example

Incantation #1

Incantation #1: Reading from Excel with LIBNAME

- We can make programming logic more dynamic by isolating key inputs in an external file such as a Microsoft Excel spreadsheet.
- SAS provides several methods for interacting with XLSX files.

Statement	Description
LIBNAME	**Associates a library reference with a specific folder, file, or other data source**

```
LIBNAME libref <engine> "SAS-library";
```

libref: Shortcut name or nickname for a storage location

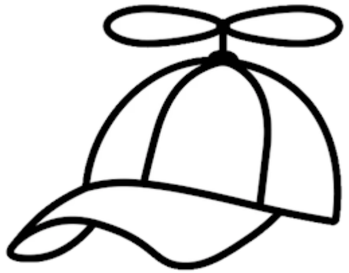
engine: Omit for SAS datasets, but use XLSX for .XLSX files

SAS-library: Physical location of folder or file
For an Excel file, include full path and filename.

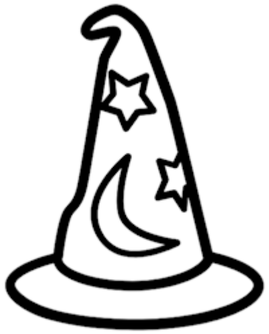
Example #1: Report Creation using Control Files

- Goal: Create a series of summary tables based on a common structure but having different titles, footnotes, and subsetting criteria.

- Muggle approach:
  - Write SAS code for one summary table.
  - Copy and change as necessary for each subsequent table.



- Macro Wizard approach:
  - Create a SAS macro to generate the summary table.
  - Manually call the macro with different parameter values for each table.



- Advanced Macro Wizard approach:
  - Place parameter values in a control file.
  - Use the LIBNAME XLSX engine to read the control file and dynamically generate the macro calls to create the tables.

Example #1: Report Creation using Control Files

Desired Output

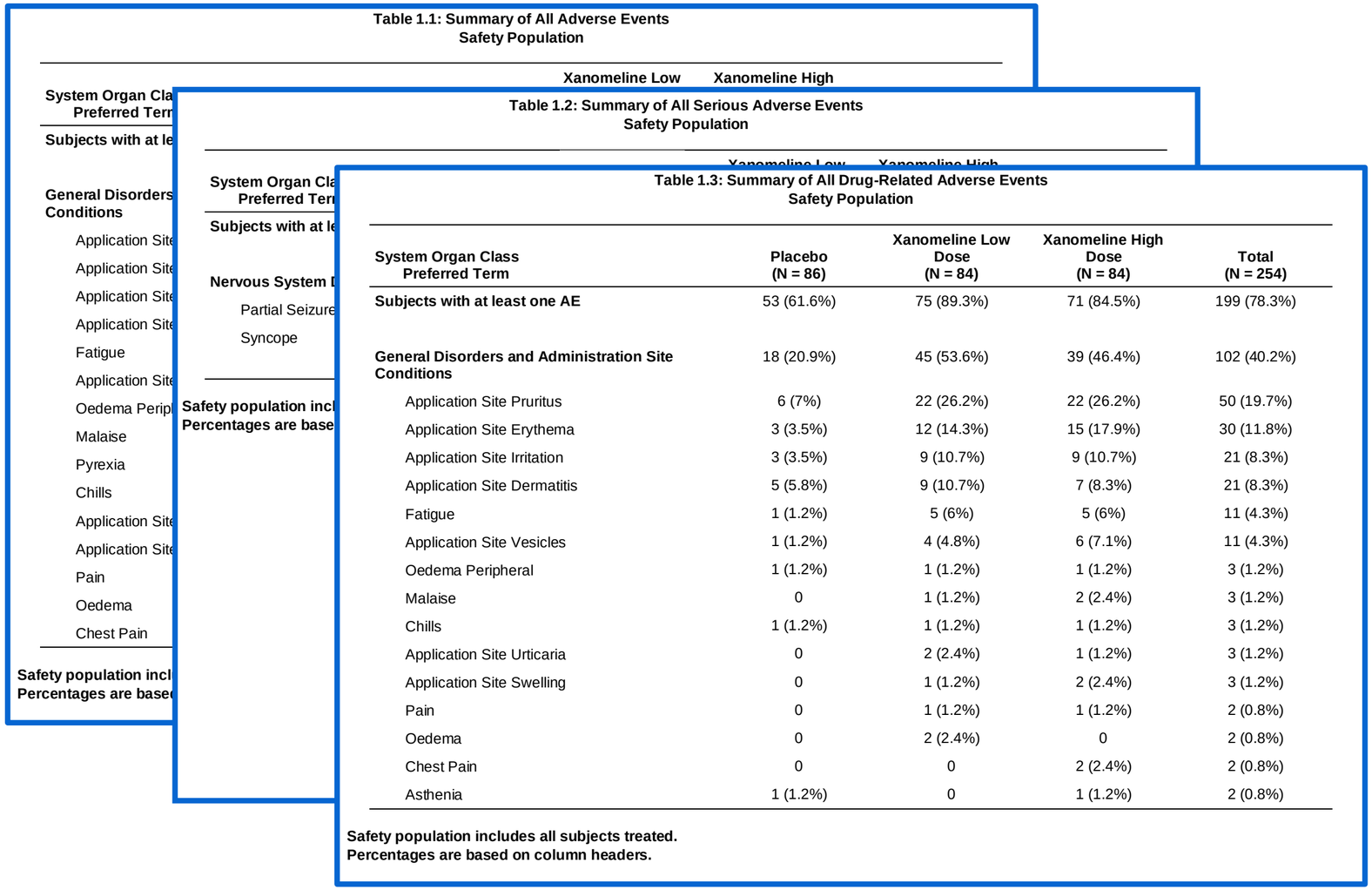


Table 1.1: Summary of All Adverse Events
Safety Population

System Organ Cla Preferred Terr	Xanomeline Low	Xanomeline High
Subjects with at le		
General Disorders Conditions		
Application Site		
Application Site		
Application Site		
Application Site		
Fatigue		
Application Site		
Oedema Perip		
Malaise		
Pyrexia		
Chills		
Application Site		
Application Site		
Pain		
Oedema		
Chest Pain		

Safety population incl
Percentages are base

Table 1.2: Summary of All Serious Adverse Events
Safety Population

System Organ Cla Preferred Terr	Xanomeline Low	Xanomeline High
Subjects with at le		
Nervous System D		
Partial Seizure		
Syncope		

Safety population incl
Percentages are base

Table 1.3: Summary of All Drug-Related Adverse Events
Safety Population

System Organ Class Preferred Term	Placebo (N = 86)	Xanomeline Low Dose (N = 84)	Xanomeline High Dose (N = 84)	Total (N = 254)
Subjects with at least one AE	53 (61.6%)	75 (89.3%)	71 (84.5%)	199 (78.3%)
General Disorders and Administration Site Conditions	18 (20.9%)	45 (53.6%)	39 (46.4%)	102 (40.2%)
Application Site Pruritus	6 (7%)	22 (26.2%)	22 (26.2%)	50 (19.7%)
Application Site Erythema	3 (3.5%)	12 (14.3%)	15 (17.9%)	30 (11.8%)
Application Site Irritation	3 (3.5%)	9 (10.7%)	9 (10.7%)	21 (8.3%)
Application Site Dermatitis	5 (5.8%)	9 (10.7%)	7 (8.3%)	21 (8.3%)
Fatigue	1 (1.2%)	5 (6%)	5 (6%)	11 (4.3%)
Application Site Vesicles	1 (1.2%)	4 (4.8%)	6 (7.1%)	11 (4.3%)
Oedema Peripheral	1 (1.2%)	1 (1.2%)	1 (1.2%)	3 (1.2%)
Malaise	0	1 (1.2%)	2 (2.4%)	3 (1.2%)
Chills	1 (1.2%)	1 (1.2%)	1 (1.2%)	3 (1.2%)
Application Site Urticaria	0	2 (2.4%)	1 (1.2%)	3 (1.2%)
Application Site Swelling	0	1 (1.2%)	2 (2.4%)	3 (1.2%)
Pain	0	1 (1.2%)	1 (1.2%)	2 (0.8%)
Oedema	0	2 (2.4%)	0	2 (0.8%)
Chest Pain	0	0	2 (2.4%)	2 (0.8%)
Asthenia	1 (1.2%)	0	1 (1.2%)	2 (0.8%)

Safety population includes all subjects treated.
Percentages are based on column headers.

Example #1: Report Creation using Control Files

Muggle Code

This code must be repeated for each summary table needed.

```
proc sql noprint;
  create table ae as select * from adae;
```

```
proc sql noprint;
  create table ae as select * from adae where aeser='Y';
```

```
proc sql noprint;
  create table ae as select * from adae where relgr1='RELATED';
  create table cnts_subj as
    select TRTAN, 1 as SCTORD, 'Subjects with at least one AE' as ROWLBL length = 200, count(distinct USUBJID) as NUMSUBJ
    from ae group by TRTAN order by TRTAN, NUMSUBJ desc;
  create table cnts_soc as
    select TRTAN, 1 as SCTORD, AEBODSYS, AEBODSYS as ROWLBL length = 200, count(distinct USUBJID) as NUMSUBJ
    from ae group by TRTAN, AEBODSYS order by TRTAN, NUMSUBJ desc;
  create table cnts_pt as
    select TRTAN, 1 as SCTORD, AEBODSYS, AEDECOD, AEDECOD as ROWLBL length = 200, count(distinct USUBJID) as NUMSUBJ
    from ae group by TRTAN, AEBODSYS, AEDECOD order by TRTAN, AEBODSYS, NUMSUBJ desc;
quit;
/* Additional code for processing and formatting report. */
ods rtf file = "t_aerel.rtf"; title1 "Table 1.3: Summary of Drug-Related Adverse Events"; title2 "Safety Population";
footnote1 j=l "Safety population includes all subjects treated."; footnote2 j=l "Percentages are based on column headers.";

proc report data = final split = "`";
  * PROC REPORT code ... ;
run;

ods rtf close;
```

Example #1: Report Creation using Control Files

Macro Wizard Code – Part 1 of 2

```
%macro t_ae(output_name = , output_number = , subset = , title1 = , title2 = , footnote1 = , footnote2 = );

  proc sql noprint;
   create table ae as
   select * from adae %if %bquote(&subset) ne %then where &subset; ;

   %aecnts(typ = subj, ord = 1, byvar = ,                          lbl = %str('Subjects with at least one AE'))
   %aecnts(typ = soc,  ord = 2, byvar = AEBODSYS,                  lbl = AEBODSYS)
   %aecnts(typ = pt,   ord = 2, byvar = %str(AEBODSYS, AEDECOD), lbl = AEDECOD)
  quit;

  /* Additional code for processing and format

  ods rtf file = "&output_name..rtf";
  title1 "Table &output_number: &title1";
  title2 "&title2";
  footnote1 j=l "&footnote1";
  footnote2 j=l "&footnote2";
  proc report data = final split = "`";
    * PROC REPORT code ... ;
  run;
  ods rtf close;
%mend t_ae;
```

A nested macro call is used to create the individual select statements.

Output name, number, titles, and footnotes are controlled by macro parameters.

Example #1: Report Creation using Control Files

Macro Wizard Code – Part 2 of 2

```
%t_ae(output_name   = t_ae,
      output_number = 1.1,
      subset        = ,
      title1        = Summary of All Adverse Events,
      title2        = Safety Population,
      footnote1     = Safety population includes all subjects treated.,
      footnote2     = Percentages are based on column headers.);

%t_ae(output_name   = t_aeser,
      output_number = 1.2,
      subset        = %str(AESER = 'Y'),
      title1        = Summary of All Serious Adverse Events,
      title2        = Safety Population,
      footnote1     = Safety population includes all subjects treated.,
      footnote2     = Percentages are based on column headers.);

%t_ae(output_name   = t_aerel,
      output_number = 1.3,
      subset        = %str(RELGR1 = 'RELATED'),
      title1        = Summary of All Drug-Related Adverse Events,
      title2        = Safety Population,
      footnote1     = Safety population includes all subjects treated.,
      footnote2     = Percentages are based on column headers.);
```

Example #1: Report Creation using Control Files

The Control File

One column per macro parameter

One row per macro call

Example_1_control_file.xlsx • Last Modified: Just now

Search

File Home Insert Page Layout Formulas Data Review View Automate Developer Help Acrobat

Comments Share

C3 aeser='Y'

	A	B	C	D	E	F	G	H	I	J
1	OUTPUT_NAME	OUTPUT_NUMBER	SUBSET	TITLE1	TITLE2	FOOTNOTE1	FOOTNOTE2			
2	t_ae	1.1		Summary of All Adverse Events	Safety Population	Safety population includes all subjects treated.	Percentages are based on column headers.			
3	t_aeser	1.2	aeser='Y'	Summary of Serious Adverse Events	Safety Population	Safety population includes all subjects treated.	Percentages are based on column headers.			
4	t_aerel	1.3	relgr1='RELATED'	Summary of Drug-Related Adverse Events	Safety Population	Safety population includes all subjects treated.	Percentages are based on column headers.			
5										

Tables

Ready Accessibility: Good to go 100%

Example #1: Report Creation using Control Files

Advanced Macro Wizard Code – Part 2A

```
libname control xlsx 'C:\Example_1_control_file.xlsx';


data _null_;
        set control.tables end=eof;
        call symputx(cats('OUTPUTNAME', _n_), output_name  );
        call symputx(cats('OUTPUTNUM' , _n_), output_number);
        call symputx(cats('SUBSET'    , _n_), subset       );
        call symputx(cats('TITLE1_'   , _n_), title1       );
        call symputx(cats('TITLE2_'   , _n_), title2       );
        call symputx(cats('FOOTNOTE1_', _n_), footnote1    );
        call symputx(cats('FOOTNOTE2_', _n_), footnote2    );
      if eof then call symputx('NUMOUTPUTS', _n_);
run;
```

LIBNAME statement with XLSX engine

_N_ is an automatic data step variable that provides the current record count.

Each parameter value is read from the control file and placed into a macro variable with a sequentially numbered suffix.

Create one additional macro variable to track the number of macro calls to generate.

Example #1: Report Creation using Control Files

Advanced Macro Wizard Code – Part 2B

```
%macro run_t_ae;

        %do i = 1 %to &NUMOUTPUTS;

                %t_ae(
                        output_name   = &&OUTPUTNAME&i,
                        output_number = &&OUTPUTNUM&i,
                        subset        = &&SUBSET&i,
                        title1        = &&TITLE1_&i,
                        title2        = &&TITLE2_&i,
                        footnote1     = &&FOOTNOTE1_&i,
                        footnote2     = &&FOOTNOTE2_&i,
                );

        %end;

%mend run_t_ae;

%run_t_ae;
```

Macro %DO loop executes once for each record read from the control file.

Macro variables containing values read from the control file are passed in as parameter values.

Example #1: Report Creation using Control Files

Output

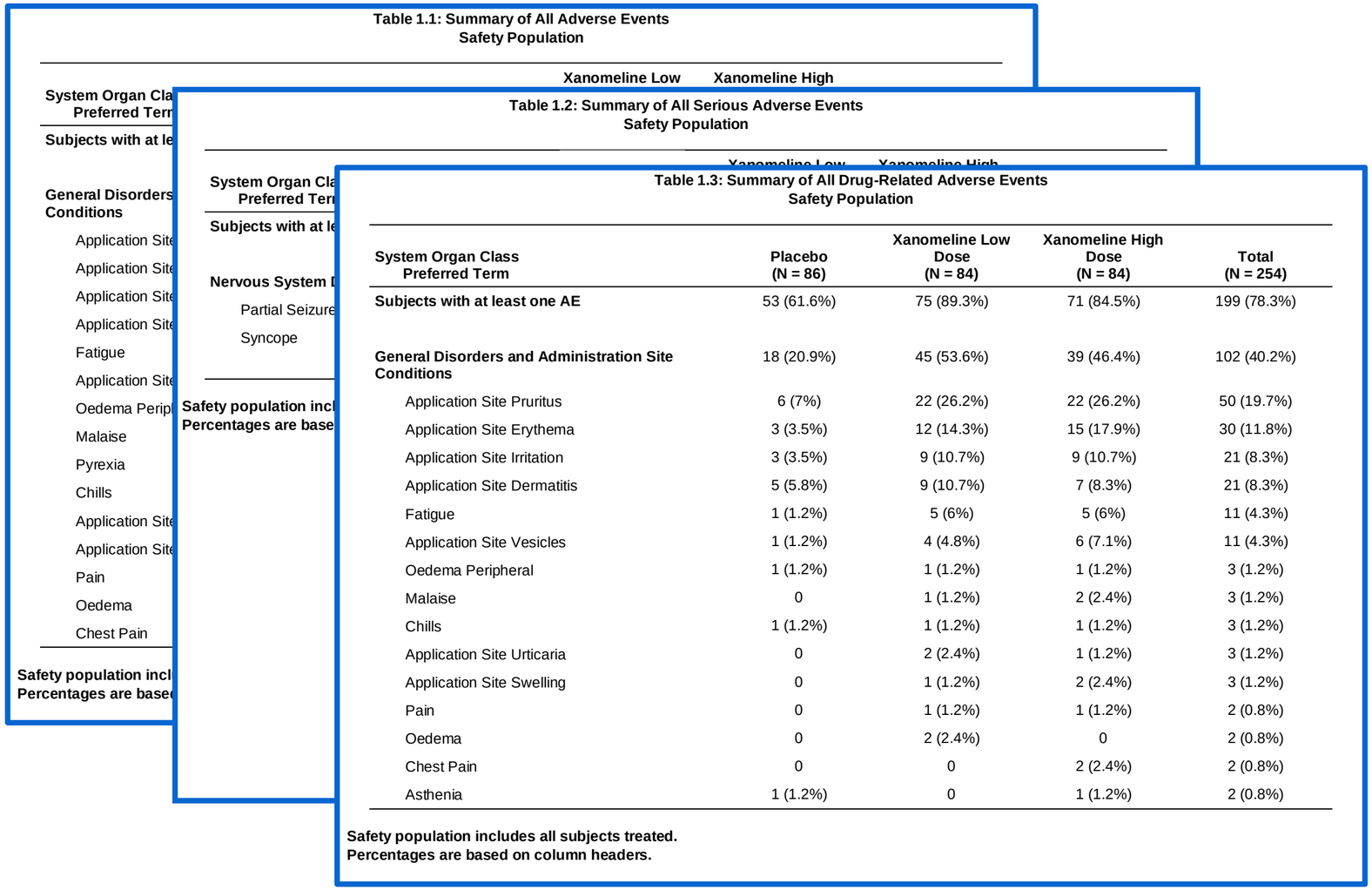


Table 1.1: Summary of All Adverse Events
Safety Population

System Organ Cla Preferred Ter	Xanomeline Low	Xanomeline High
Subjects with at le		
General Disorders Conditions		
Application Site		
Application Site		
Application Site		
Application Site		
Fatigue		
Application Site		
Oedema Perip		
Malaise		
Pyrexia		
Chills		
Application Site		
Application Site		
Pain		
Oedema		
Chest Pain		

Safety population incl
Percentages are base

Table 1.2: Summary of All Serious Adverse Events
Safety Population

System Organ Cla Preferred Ter	Xanomeline Low	Xanomeline High
Subjects with at le		
Nervous System D		
Partial Seizure		
Syncope		

Safety population incl
Percentages are base

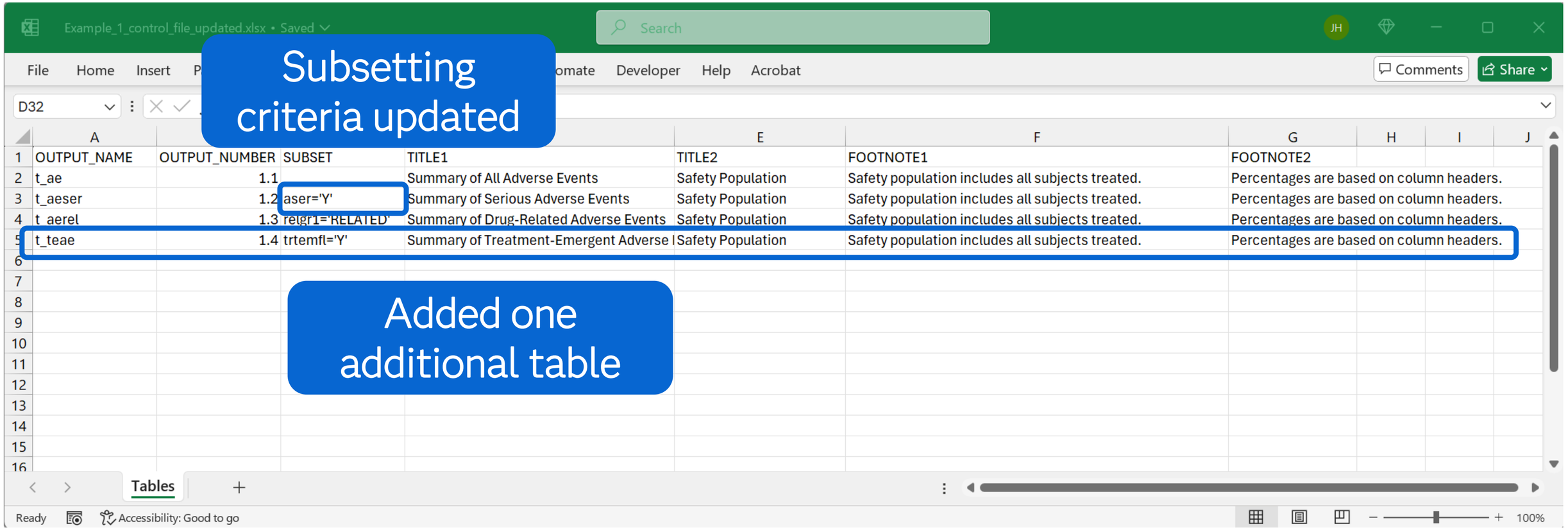
Table 1.3: Summary of All Drug-Related Adverse Events
Safety Population

System Organ Class Preferred Term	Placebo (N = 86)	Xanomeline Low Dose (N = 84)	Xanomeline High Dose (N = 84)	Total (N = 254)
Subjects with at least one AE	53 (61.6%)	75 (89.3%)	71 (84.5%)	199 (78.3%)
General Disorders and Administration Site Conditions	18 (20.9%)	45 (53.6%)	39 (46.4%)	102 (40.2%)
Application Site Pruritus	6 (7%)	22 (26.2%)	22 (26.2%)	50 (19.7%)
Application Site Erythema	3 (3.5%)	12 (14.3%)	15 (17.9%)	30 (11.8%)
Application Site Irritation	3 (3.5%)	9 (10.7%)	9 (10.7%)	21 (8.3%)
Application Site Dermatitis	5 (5.8%)	9 (10.7%)	7 (8.3%)	21 (8.3%)
Fatigue	1 (1.2%)	5 (6%)	5 (6%)	11 (4.3%)
Application Site Vesicles	1 (1.2%)	4 (4.8%)	6 (7.1%)	11 (4.3%)
Oedema Peripheral	1 (1.2%)	1 (1.2%)	1 (1.2%)	3 (1.2%)
Malaise	0	1 (1.2%)	2 (2.4%)	3 (1.2%)
Chills	1 (1.2%)	1 (1.2%)	1 (1.2%)	3 (1.2%)
Application Site Urticaria	0	2 (2.4%)	1 (1.2%)	3 (1.2%)
Application Site Swelling	0	1 (1.2%)	2 (2.4%)	3 (1.2%)
Pain	0	1 (1.2%)	1 (1.2%)	2 (0.8%)
Oedema	0	2 (2.4%)	0	2 (0.8%)
Chest Pain	0	0	2 (2.4%)	2 (0.8%)
Asthenia	1 (1.2%)	0	1 (1.2%)	2 (0.8%)

Safety population includes all subjects treated.
Percentages are based on column headers.

Example #1: Report Creation using Control Files

Updating the Control File



	A	B	C	D	E	F	G
1	OUTPUT_NAME	OUTPUT_NUMBER	SUBSET	TITLE1	TITLE2	FOOTNOTE1	FOOTNOTE2
2	t_ae	1.1		Summary of All Adverse Events	Safety Population	Safety population includes all subjects treated.	Percentages are based on column headers.
3	t_aeser	1.2	aser='Y'	Summary of Serious Adverse Events	Safety Population	Safety population includes all subjects treated.	Percentages are based on column headers.
4	t_aerel	1.3	relgr1='RELATED'	Summary of Drug-Related Adverse Events	Safety Population	Safety population includes all subjects treated.	Percentages are based on column headers.
5	t_teae	1.4	trtemfl='Y'	Summary of Treatment-Emergent Adverse I	Safety Population	Safety population includes all subjects treated.	Percentages are based on column headers.

Example #1: Report Creation using Control Files

Output

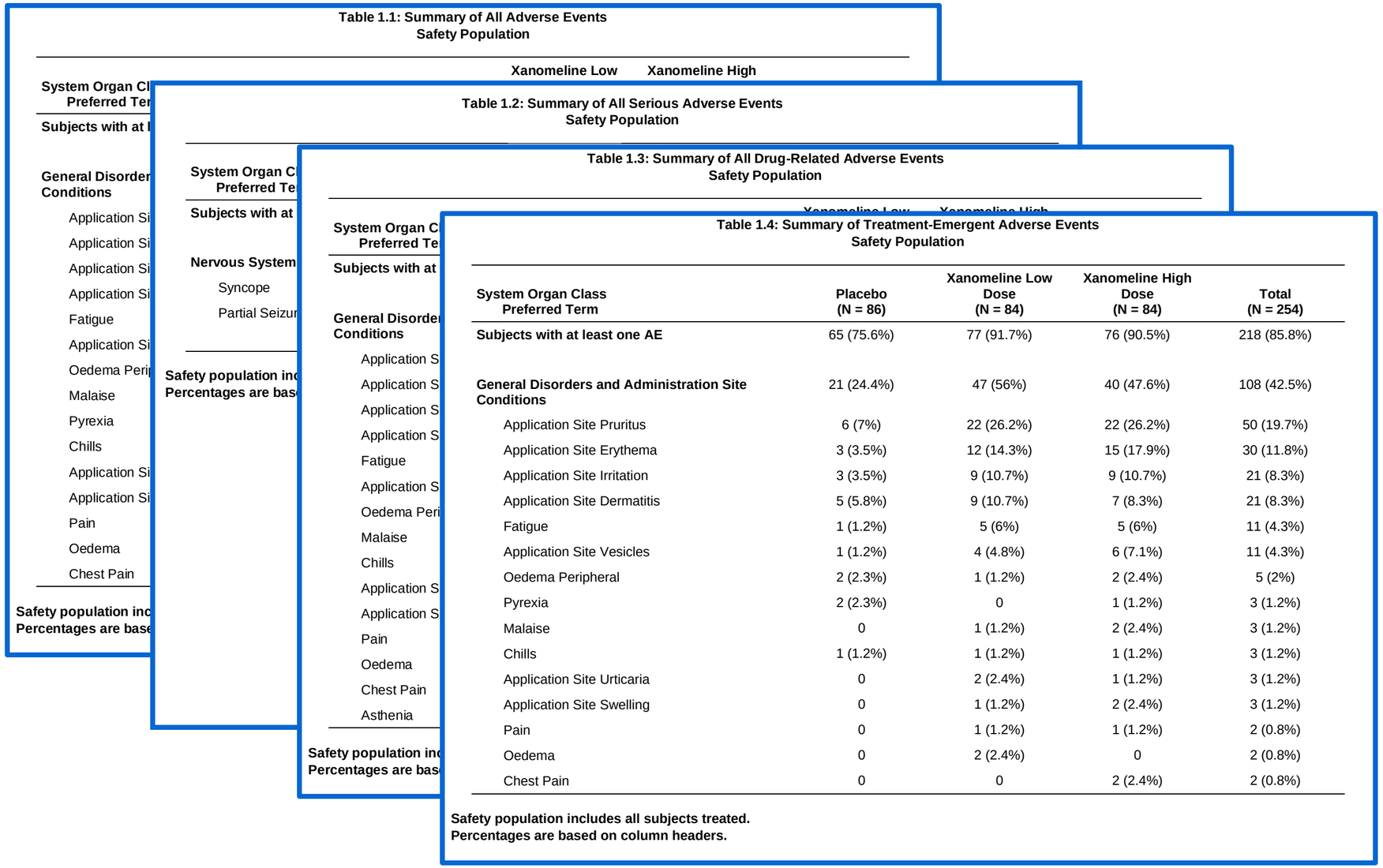


Table 1.1: Summary of All Adverse Events
Safety Population

System Organ Cl… Preferred Ter…	Xanomeline Low	Xanomeline High
Subjects with at l…		
General Disorder… Conditions		
Application Si…		
Application Si…		
Application Si…		
Application Si…		
Fatigue		
Application Si…		
Oedema Peri…		
Malaise		
Pyrexia		
Chills		
Application Si…		
Application Si…		
Pain		
Oedema		
Chest Pain		

Safety population inc…
Percentages are base…

Table 1.2: Summary of All Serious Adverse Events
Safety Population

System Organ C… Preferred Te…	
Subjects with at …	
Nervous System…	
Syncope	
Partial Seizu…	

Safety population inc…
Percentages are bas…

Table 1.3: Summary of All Drug-Related Adverse Events
Safety Population

System Organ C… Preferred Te…	
Subjects with at …	
General Disorder… Conditions	
Application S…	
Application S…	
Application S…	
Application S…	
Fatigue	
Application S…	
Oedema Peri…	
Malaise	
Chills	
Application S…	
Application S…	
Pain	
Oedema	
Chest Pain	
Asthenia	

Safety population inc…
Percentages are bas…

Table 1.4: Summary of Treatment-Emergent Adverse Events
Safety Population

System Organ Class Preferred Term	Placebo (N = 86)	Xanomeline Low Dose (N = 84)	Xanomeline High Dose (N = 84)	Total (N = 254)
Subjects with at least one AE	65 (75.6%)	77 (91.7%)	76 (90.5%)	218 (85.8%)
General Disorders and Administration Site Conditions	21 (24.4%)	47 (56%)	40 (47.6%)	108 (42.5%)
Application Site Pruritus	6 (7%)	22 (26.2%)	22 (26.2%)	50 (19.7%)
Application Site Erythema	3 (3.5%)	12 (14.3%)	15 (17.9%)	30 (11.8%)
Application Site Irritation	3 (3.5%)	9 (10.7%)	9 (10.7%)	21 (8.3%)
Application Site Dermatitis	5 (5.8%)	9 (10.7%)	7 (8.3%)	21 (8.3%)
Fatigue	1 (1.2%)	5 (6%)	5 (6%)	11 (4.3%)
Application Site Vesicles	1 (1.2%)	4 (4.8%)	6 (7.1%)	11 (4.3%)
Oedema Peripheral	2 (2.3%)	1 (1.2%)	2 (2.4%)	5 (2%)
Pyrexia	2 (2.3%)	0	1 (1.2%)	3 (1.2%)
Malaise	0	1 (1.2%)	2 (2.4%)	3 (1.2%)
Chills	1 (1.2%)	1 (1.2%)	1 (1.2%)	3 (1.2%)
Application Site Urticaria	0	2 (2.4%)	1 (1.2%)	3 (1.2%)
Application Site Swelling	0	1 (1.2%)	2 (2.4%)	3 (1.2%)
Pain	0	1 (1.2%)	1 (1.2%)	2 (0.8%)
Oedema	0	2 (2.4%)	0	2 (0.8%)
Chest Pain	0	0	2 (2.4%)	2 (0.8%)

Safety population includes all subjects treated.
Percentages are based on column headers.

Call Execute Example

Incantation #2

Incantation #2: Call Execute to Build Macro Calls

- SAS provides us with a variety of tools to help implement data-driven techniques

Subroutine	Description
CALL EXECUTE	**Resolves the argument and the resolved argument executes at the next step boundary**

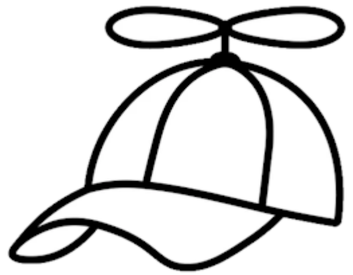
CALL EXECUTE(argument)

argument:

- character string enclosed in quotation marks
- variable in a data step
- character expression that resolves to a macro text expression or SAS statement

Example #2: Producing Lab Figures

- Goal: Generate graphs for each laboratory test for which at least 20 subjects have post-baseline records. Each graph should be stored in its own file and the Y axis range should be scaled based on the minimum and maximum data values for each laboratory test.

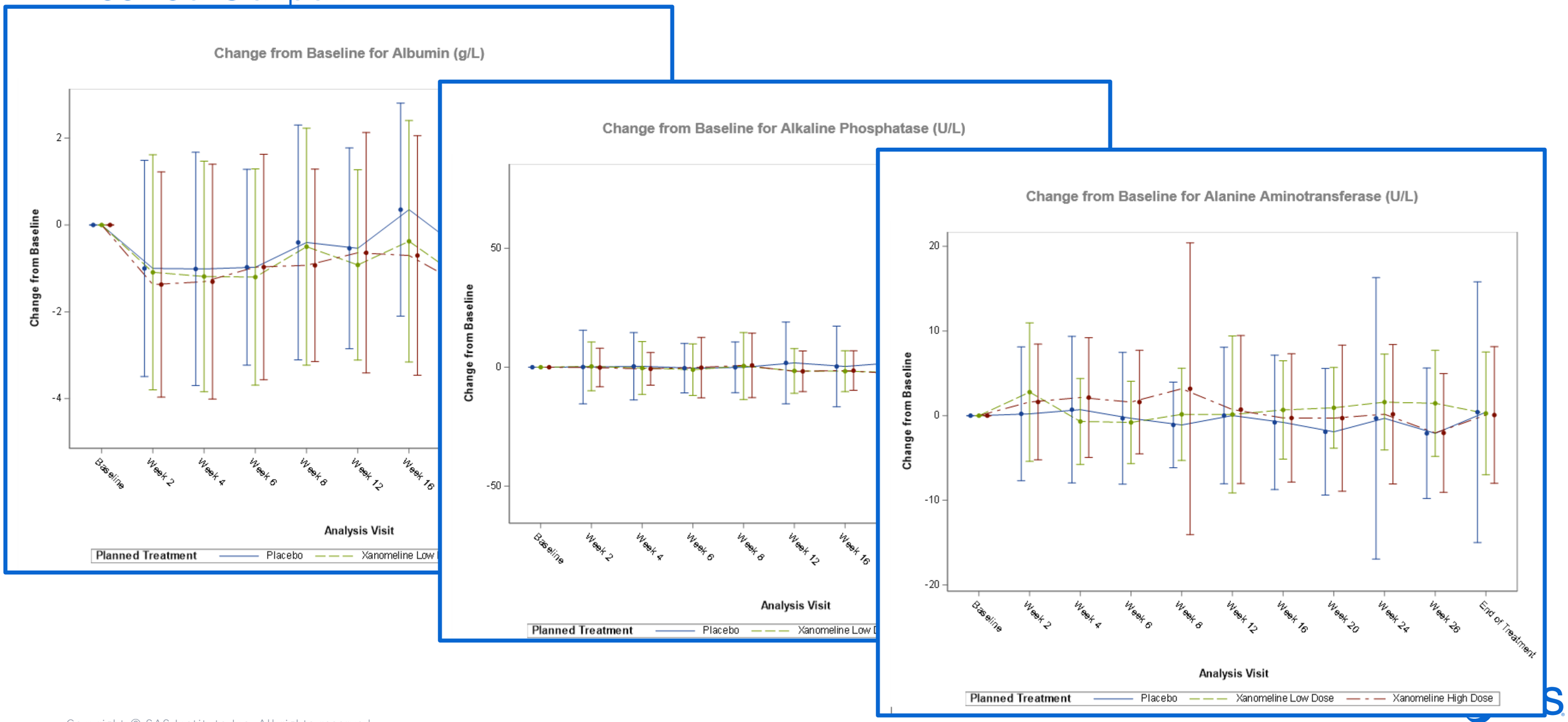


- Muggle approach:
  - Determine which laboratory tests have at least 20 subjects
  - Code a separate call to PROC SGPLOT for each unique laboratory test.

- Macro Wizard approach:
  - Use CALL EXECUTE to dynamically generate the calls to PROC SGPLOT.

Example #2: Producing Lab Figures

Desired Output



Example #2: Producing Lab Figures

Muggle Code

```
ods rtf file = "ALB.rtf" style = STYLES.DAISY nogtitle;

title "Change from Baseline for Albumin (g/L)";
proc
   wh
   xa

   ya
   se
   sc


run;
ods r
```

```
ods rtf file = "ALP.rtf" style = STYLES.DAISY nogtitle;

title "Change from Baseline for Alkaline Phosphatase (U/L)";
proc
   wh
   xa

   ya
   se
   sc


run;
ods r
```

```
ods rtf file = "ALT.rtf" style = STYLES.DAISY nogtitle;

title "Change from Baseline for Alanine Aminotransferase (U/L)";
proc sgplot data = lab_summ;
   where PARAMCD = "ALT";
   xaxis type = discrete label = 'Analysis Visit'
         valuesformat = $avis. valuesrotate = diagonal fitpolicy = rotatealways;
   yaxis type = linear label = 'Change from Baseline' min = -17 max = 21;
   series x = AVISITN y = mean / group = TRTP;
   scatter x = AVISITN y = mean / group = TRTP groupdisplay = cluster clusterwidth = .5
                                  markerattrs = (symbol = circlefilled)
                                  yerrorlower = lower yerrorupper = upper;

run;
ods rtf close;
```

This code must be repeated for each laboratory test that had at least 20 subjects

Example #2: Producing Lab Figures

Macro Wizard Code without CALL EXECUTE

```
%macro sgcrtfig(prmcd = , prm = , min = , max = );

   ods rtf file = "&prmcd..rtf" style = STYLES.DAISY nogtitle;

   title "Change from Baseline for &prm";
   proc sgplot data = lab_summ;
      where PARAMCD = "&prmcd";
      xaxis type = discrete label = 'Analysis Visit'
            valuesformat = $avis. valuesrotate = diagonal fitpolicy = rotatealways;
      yaxis type = linear label = 'Change from Baseline' min = &min max = &max;
      series x = AVISITN y = mean / group = TRTP;
      scatter x = AVISITN y = mean / group = TRTP groupdisplay = cluster clusterwidth = .5
                                     markerattrs = (symbol = circlefilled)
                                     yerrorlower = lower
                                     yerrorupper = upper;
   run;
   ods rtf close;
%mend sgcrtfig;
```

Assign macro parameters for portions that change based on the laboratory test

Example #2: Producing Lab Figures

Macro Wizard Code without CALL EXECUTE

```
%sgcrtfig(prmcd = ALB,     prm = Albumin (g/L),                    min = -5,  max = 3)
%sgcrtfig(prmcd = ALP,     prm = Alkaline Phosphatase (U/L),       min = -63, max = 83)
%sgcrtfig(prmcd = ALT,     prm = Alanine Aminotransferase (U/L),   min = -17, max = 21)
%sgcrtfig(prmcd = AST,     prm = Aspartate Aminotransferase (U/L), min = -20, max = 22)
%sgcrtfig(prmcd = BASO,    prm = Basophils (GI/L),                 min = -1,  max = 1)
%sgcrtfig(prmcd = BILI,    prm = Bilirubin (umol/L),               min = -11, max = 14)
%sgcrtfig(prmcd = BUN,     prm = Blood Urea Nitrogen (mmol/L),     min = -2,  max = 2)

…

%sgcrtfig(prmcd = SODIUM,  prm = Sodium (mmol/L),                  min = -4,  max = 6)
%sgcrtfig(prmcd = URATE,   prm = Urate (umol/L),                   min = -75, max = 50)
%sgcrtfig(prmcd = WBC,     prm = Leukocytes (GI/L),                min = -2,  max = 3)
```

Example #2: Producing Lab Figures

Macro Wizard Code with CALL EXECUTE

```
data _null_;
   set lab_summ;
   by PARAMCD AVISITN;
   retain axis_min axis_max num_placebo num_xlow num_xhigh vissubj numsubj;
   /** additional code to count the number of subjects per laboratory test **/
   if not missing(lower) then axis_min = min(axis_min, lower);
   if not missing(upper) then axis_max = max(axis_max, upper);
   if last.PARAMCD;
   numsubj = sum(of num_:);
   length maccall $200;
   if numsubj > 20;
   maccall = cats('%sgcrtfig(prmcd = ', PARAMCD, ', prm = ', %str(PARAM),
                  ', min = ', floor(axis_min), ', max = ', ceil(axis_max), ');');

   call execute(%nrstr(maccall));
run;
```

Build our macro call based on the data

Mask the macro call so it doesn't execute within the DATA step

Example #2: Producing Lab Figures

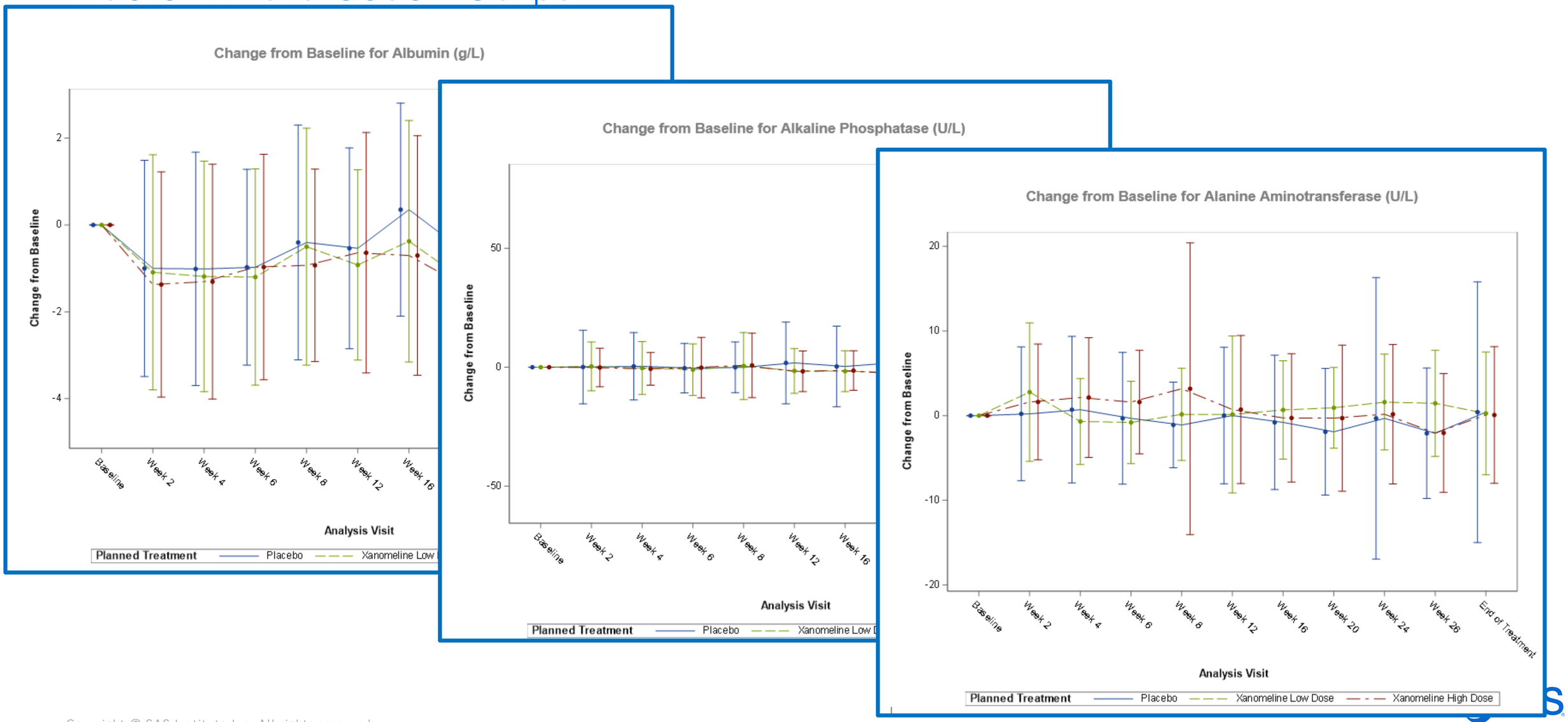
Macro Wizard Code with CALL EXECUTE

```
NOTE: CALL EXECUTE generated line.
1          + ods rtf file = "sg_ALB.rtf"     style = STYLES.DAISY nogtitle;
                     title "Change from Baseline for Albumin (g/L)";
NOTE: Writing RTF Body file: sg_ALB.rtf
2          +   proc sgplot data = lab_summ;  where PARAMCD = "ALB";
                    xaxis type = discrete label = 'Analysis Visit'
                    valuesformat = $avis. valuesrotate = diagonal fitpolicy = rotatealways;
                    yaxis type = linear label = 'Change from Baseline'
3          +        min = -5 max = 3;   series x = AVISITN y = mean / group = TRTP;
                    scatter x = AVISITN y = mean / group = TRTP
                    groupdisplay = cluster clusterwidth = .5  markerattrs = (symbol = circlefilled)
4          +        yerrorlower = lower                  yerrorupper = upper;    run;
ods rtf close;;
NOTE: PROCEDURE SGPLOT used (Total process time):
      real time           0.55 seconds
      cpu time            0.34 seconds

NOTE: There were 33 observations read from the data set WORK.LAB_SUMM.
      WHERE PARAMCD='ALB';
```

Example #2: Producing Lab Figures

Macro Wizard Code - Output



Resolve Example

Incantation #3

Incantation #3: Resolve Function Macro Function

- The DATA step has several functions for working with macros.

Function	Description
RESOLVE	**Resolves the value of the argument during DATA step execution**

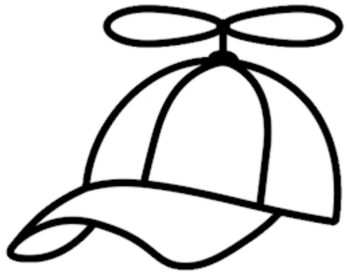
RESOLVE(argument)

argument: character constant, variable, or expression with a value that is a macro expression

Example #3: Denominators & Headers for Reports

- Goal: Generate reports of counts and percentages with percentages based on counts in the column headers

- Muggle approach:
  - Determine and merge in subject counts
  - Manually code the numbers into the column headers



- Macro Wizard approach:
  - Store the header counts in macro variables
  - Use the RESOLVE function to create the denominator in a DATA step

Example #3: Denominators & Headers for Reports

Desired Output

		Placebo (N=86)	Xanomeline Low Dose (N=84)	Xanomeline High Dose (N=84)
AGEGR1	65-80	42 (48.8%)	47 (56%)	55 (65.5%)
	<65	14 (16.3%)	8 (9.5%)	11 (13.1%)
	>80	30 (34.9%)	29 (34.5%)	18 (21.4%)
BMIBLGR1	25-<30	21 (24.4%)	27 (32.1%)	28 (33.3%)
	<25	59 (68.6%)	46 (54.8%)	44 (52.4%)
	>=30	6 (7%)	10 (11.9%)	12 (14.3%)
DURDSGR1	<12	5 (5.8%)	3 (3.6%)	4 (4.8%)
	>=12	81 (94.2%)	81 (96.4%)	80 (95.2%)

Example #3: Denominators & Headers for Reports

Muggle Code

```
proc freq data = adsl noprint;
   tables TRT01PN * TRT01P / out = trtcnt (drop = PERCENT);
run;

ods output crosstabfreqs = ctf (where = (not missing(TRT01PN)));
proc freq data = adsl;
   tables TRT01PN * (AGEGR1 BMIBLGR1 DURDSGR1);
run;

/** additional code to combine the two data sets and prep for report **/
```

TRT01PN	TRT01P	COUNT
1	Placebo	86
2	Xanomeline Low Dose	84
3	Xanomeline High Dose	84

Example #3: Denominators & Headers for Reports

Muggle Code

```
proc report data = tcnt_pct;
    columns SCTORD SCTLBL ROWORD ROWLBL TRT1 TRT2 TRT3;
    define SCTORD / order noprint;
    define SCTLBL / order ' ';
    define ROWORD / order noprint;
    define ROWLBL / order ' ';
    define TRT1   / 'Placebo (N=86)';
    define TRT2   / 'Xanomeline Low Dose (N=84)';
    define TRT3   / 'Xanomeline High Dose (N=84)';
run;
```

These labels and numbers must be manually entered

Repeat for multiple reports!

Repeat entire process when new reports are needed!

Example #3: Denominators & Headers for Reports

Macro Wizard Code with RESOLVE Function

```
proc sql noprint;
   select distinct TRT01PN, count(distinct USUBJID), TRT01P
          into :t1 -, :n1 -, :l1 -
   from adsl
   group by TRT01PN
   order by TRT01PN;
   %let numtrt = &sqlobs;
quit;

ods output crosstabfreqs = ctf (where = (not missing(TRT01PN)));
proc freq data = adsl;
   tables TRT01PN * (AGEGR1 BMIBLGR1 DURDSGR1);
run;
```

```
36          %put _user_;
GLOBAL L1 Placebo
GLOBAL L2 Xanomeline Low Dose
GLOBAL L3 Xanomeline High Dose
GLOBAL N1 86
GLOBAL N2 84
GLOBAL N3 84
GLOBAL NUMTRT 3
GLOBAL T1 1
GLOBAL T2 2
GLOBAL T3 3
```

Example #3: Denominators & Headers for Reports

Macro Wizard Code with RESOLVE Function

```
data cnt_pct (keep = TRT: SCT: ROW: FREQUENCY DENO PERCE
    length ROWSCT $8 ROWLBL $20;
    set ctf;
    /** additional code to create sort variables for repor
    DENO = input(resolve(cats('&n', TRT01PN)), 8.);
    PERCENT = round((FREQUENCY / DENO) * 100, .1);
    RESULT = catx(' ', FREQUENCY, cats('(', PERCENT, '%)'
run;
```

&n is masked at compile time

CATS function builds a macro variable

```
GLOBAL N1 86
GLOBAL N2 84
GLOBAL N3 84
```

ROWSCT	ROWLBL	TRT01PN	Frequency	SCTLBL	SCTORD	ROWORD	DENO	PERCENT	RESULT
	65-80	1	42	AGEGR1	1	1	86	48.8	42 (48.8%)
	65-80	2	47	AGEGR1	1	1	84	56	47 (56%)
	65-80	3	55	AGEGR1	1	1	84	65.5	55 (65.5%)
	25-<30	1	21	BMIBLGR1	2	1	86	24.4	21 (24.4%)
	25-<30	2	27	BMIBLGR1	2	1	84	32.1	27 (32.1%)
	25-<30	3	28	BMIBLGR1	2	1	84	33.3	28 (33.3%)
	<12	1	5	DURDSGR1	3	1	86	5.8	5 (5.8%)
	<12	2	3	DURDSGR1	3	1	84	3.6	3 (3.6%)
	<12	3	4	DURDSGR1	3	1	84	4.8	4 (4.8%)

Example #3: Denominators & Headers for Reports

Macro Wizard Code with RESOLVE Function

```
%macro crtrpt;
   proc report data = tcnt_pct;
      columns SCTORD SCTLBL ROWORD ROWLBL
              %do i = 1 %to &numtrt; TRT&i %end;
              ;
      define SCTORD / order noprint;
      define SCTLBL / order ' ';
      define ROWORD / order noprint;
      define ROWLBL / order ' ';
      %do j = 1 %to &numtrt;
         define TRT&j / "&&l&j (N=&&n&j)";
      %end;
   run;
%mend crtrpt;

%crtrpt
```

Create list of variables using NUMTRT macro variable

Build DEFINE statements using the macro variables

```
GLOBAL L1 Placebo
GLOBAL L2 Xanomeline Low Dose
GLOBAL L3 Xanomeline High Dose
GLOBAL N1 86
GLOBAL N2 84
GLOBAL N3 84
GLOBAL NUMTRT 3
```

Example #3: Denominators & Headers for Reports

Macro Wizard Code with RESOLVE Function - Output

		Placebo (N=86)	Xanomeline Low Dose (N=84)	Xanomeline High Dose (N=84)
AGEGR1	65-80	42 (48.8%)	47 (56%)	55 (65.5%)
	<65	14 (16.3%)	8 (9.5%)	11 (13.1%)
	>80	30 (34.9%)	29 (34.5%)	18 (21.4%)
BMIBLGR1	25-<30	21 (24.4%)	27 (32.1%)	28 (33.3%)
	<25	59 (68.6%)	46 (54.8%)	44 (52.4%)
	>=30	6 (7%)	10 (11.9%)	12 (14.3%)
DURDSGR1	<12	5 (5.8%)	3 (3.6%)	4 (4.8%)
	>=12	81 (94.2%)	81 (96.4%)	80 (95.2%)

Wrap Up

Conclusion

- The SAS Macro Language provides powerful data-driven magic!
- Cast these spells to build robust programs:
  - Include dynamic logic
  - Avoid hard-coding
  - Adapt to changes in data or computing environment
- Advantages:
  - Less likely to require change
  - Easier to maintain
  - Greater potential for reuse

Any Questions?

josh@nestedloopconsulting.com
richann.watson@datarichconsulting.com

sas innovate 2025

Recommended Resources

- Carpenter, Art. 2016. *Carpenter's Complete Guide to the SAS® Macro Language, Third Edition*. Cary, NC: SAS Institute Inc.
- SAS Institute Inc. 2016. *SAS® 9.4 Macro Language: Reference, Fifth Edition*. Cary, NC: SAS Institute Inc.