Paper 55-2018

Bring the Vampire out of the Shadows: Understanding the RETAIN and COUNT functions in SAS®

Steve Black, Precision Medicine Group, Inc.

ABSTRACT

In SAS there are few things that can turn a good programmer into a pale, sleep deprived, shadow seeking individual like a RETAIN statement that is not working right. In this paper I hope to provide a deeper understanding on how to better count on and count with the RETAIN statement in SAS. I will illustrate a number of ways to count using the properties of the program data vector and provide some real life examples of how I have found using the RETAIN statement super helpful. In the end I hope to bring out of the shadows and into the light the RETAIN and count functions.

INTRODUCTION

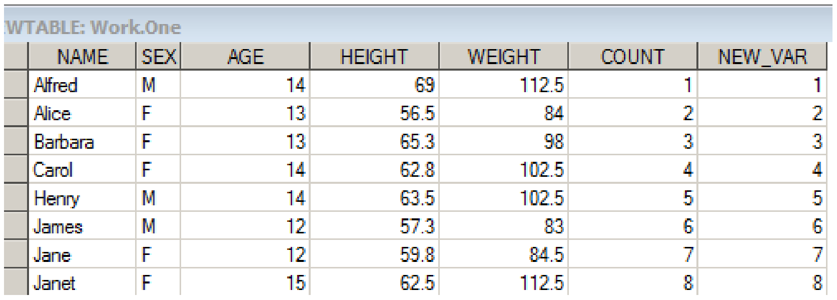
In my years of programming I've found that the retain statement is one those little bits of code that sometimes just drives me nuts, no matter how many different ways I tweak it and it still does not do what I want it to do. In many cases, I know what I want and I think I know how to do it but the data and the programming and my understanding of the logic just don't seem to match up. In this paper I hope to reduce some of this frustration by providing a basic illustration of how the retain statement works and a number of situations where I have found it to be amazingly helpful.

A SIMPLE WAY TO COUNT

Let's start off with a quick example of counting in SAS. In SAS there are usually two or more ways of doing things and counting is no different. Using the Program Data Vector (PDV) when reading in a dataset, SAS will has many internal counting variables than can be used when processing data. One of these hidden variables is the _n_. This variable incrementally counts the number of rows that are processed in the data step but can only be seen if referenced by another variable.

```
data one;
set sashelp.class;
*** option 1 ***;
new_var=_n_;
*** option 2 ***;
count+1;
run;
```

The second option uses the count function in SAS which again incrementally counts the number of rows being created in the program data vector as the new dataset is created. This function can be used to create any numerical value. In the example above I called it COUNT but you could call it anything you want. The number itself can also be anything you want too, if you use the logic COUNT+2, the counts would then begin with 2 and increase in by increments of 2 (2, 4, 6, 8...).


WTABLE: Work.One

	NAME	SEX	AGE	HEIGHT	WEIGHT	COUNT	NEW_VAR
	Alfred	M	14	69	112.5	1	1
	Alice	F	13	56.5	84	2	2
	Barbara	F	13	65.3	98	3	3
	Carol	F	14	62.8	102.5	4	4
	Henry	M	14	63.5	102.5	5	5
	James	M	12	57.3	83	6	6
	Jane	F	12	59.8	84.5	7	7
	Janet	F	15	62.5	112.5	8	8

Display 1. View of Data One

ADDING SOME COMPLEXITY USING THE BY STATEMENT

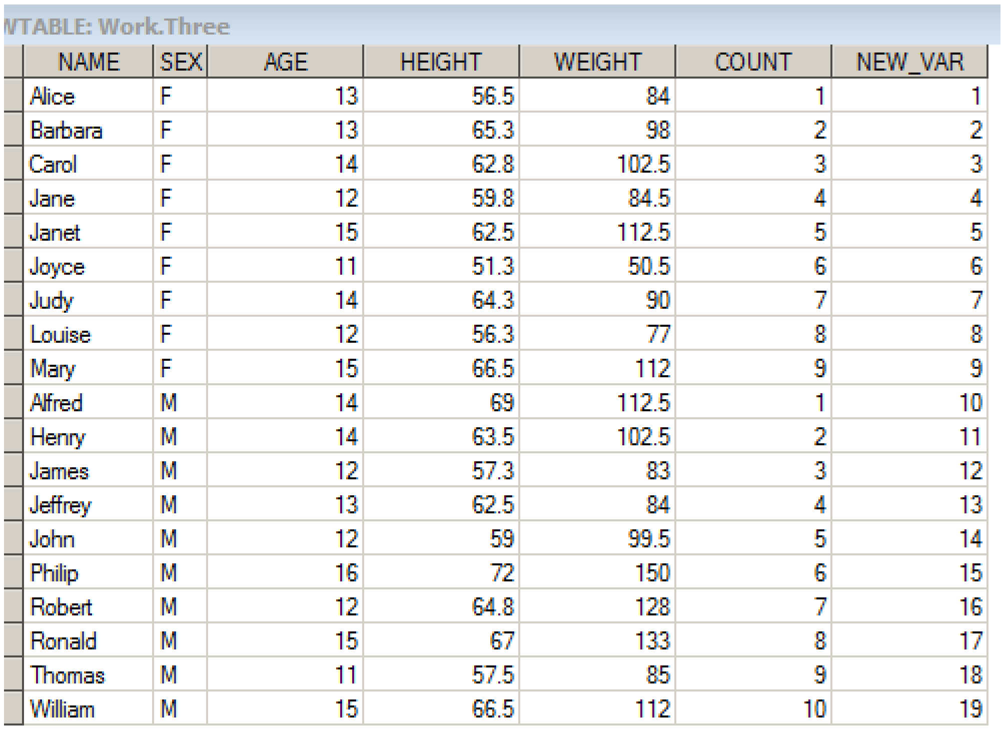
So now that have an idea on how SAS counts let's add some complexity by adding in a by variable and see how that changes or does not change things. First we sort the SASHELP.CLASS dataset by sex and then output it as a new dataset called TWO.

```
proc sort data=sashelp.class out=two;
by sex;
run;
```

Next using the new dataset created we create a new dataset called THREE and use the same BY statement to be able to use the sort more effectively and allow you to access the first. and last. functions. In the example below we want to restart the count when the sex changes. Using the FIRST.SEX code allows us to do this but we need to reset the value of count back to zero and then begin the count again.

```
data three; set two;
by sex;
*** sets the first new value to null ***;
if first.sex then count=.;
*** starts counting by 1 by sex ***;
count+1;
*** just for fun let's see what this guy still does ***;
new_var=_n_;
run;
```

Once run, we see that the COUNT variable restarts once the sex changes, and the NEW_VAR variable continues to count as it did previously.



VTABLE: Work.Three

NAME	SEX	AGE	HEIGHT	WEIGHT	COUNT	NEW_VAR
Alice	F	13	56.5	84	1	1
Barbara	F	13	65.3	98	2	2
Carol	F	14	62.8	102.5	3	3
Jane	F	12	59.8	84.5	4	4
Janet	F	15	62.5	112.5	5	5
Joyce	F	11	51.3	50.5	6	6
Judy	F	14	64.3	90	7	7
Louise	F	12	56.3	77	8	8
Mary	F	15	66.5	112	9	9
Alfred	M	14	69	112.5	1	10
Henry	M	14	63.5	102.5	2	11
James	M	12	57.3	83	3	12
Jeffrey	M	13	62.5	84	4	13
John	M	12	59	99.5	5	14
Philip	M	16	72	150	6	15
Robert	M	12	64.8	128	7	16
Ronald	M	15	67	133	8	17
Thomas	M	11	57.5	85	9	18
William	M	15	66.5	112	10	19

Display 2. View of Data Three

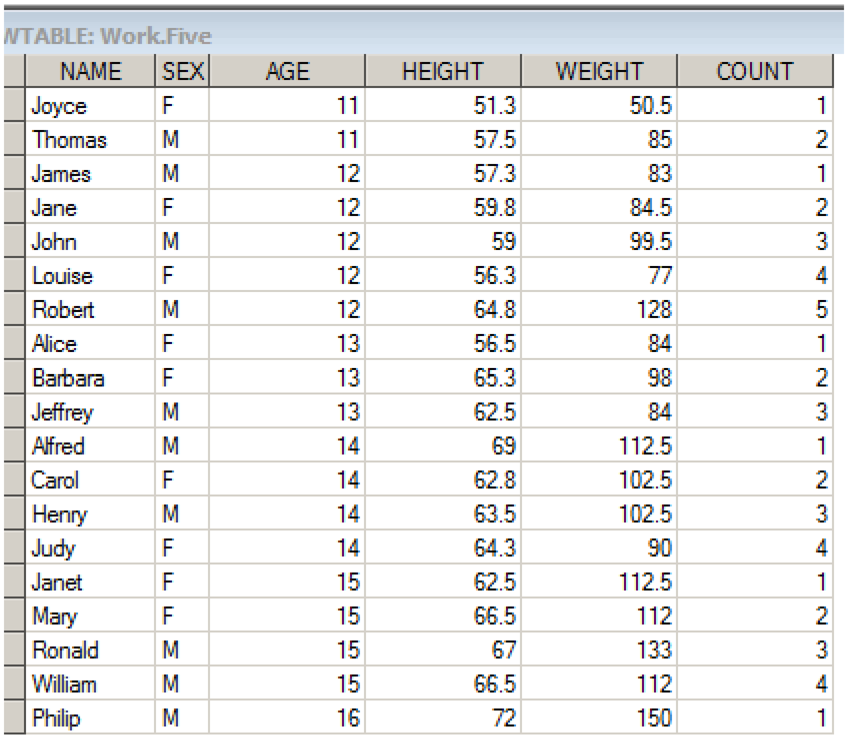
REPEATED WITH MULTIPLE BY GROUPS

To further Illustrate this we use the same SASHELP.CLASS data but this time we use the AGE variable which can be grouped into more than just two values.

```
proc sort data=sashelp.class out=four;
by age;
run;

data five; set four;
by age;
if first.age then count=.;
count+1;
run;
```

The resulting display illustrates that now per each of the different age groups in the dataset the value of COUNT will reset to 1.


WTABLE: Work.Five

NAME	SEX	AGE	HEIGHT	WEIGHT	COUNT
Joyce	F	11	51.3	50.5	1
Thomas	M	11	57.5	85	2
James	M	12	57.3	83	1
Jane	F	12	59.8	84.5	2
John	M	12	59	99.5	3
Louise	F	12	56.3	77	4
Robert	M	12	64.8	128	5
Alice	F	13	56.5	84	1
Barbara	F	13	65.3	98	2
Jeffrey	M	13	62.5	84	3
Alfred	M	14	69	112.5	1
Carol	F	14	62.8	102.5	2
Henry	M	14	63.5	102.5	3
Judy	F	14	64.3	90	4
Janet	F	15	62.5	112.5	1
Mary	F	15	66.5	112	2
Ronald	M	15	67	133	3
William	M	15	66.5	112	4
Philip	M	16	72	150	1

Display 3. View of Data Five

THE RETAIN STATEMENT

So now that we have demonstrated a simple way to count and how to count by a variable let's add some more complexity where we what to count the number of changes of age groups and not just the observations per age group. For this we will need to bring the retain statement with other pieces of the code that we have seen before.

The definition of the retain statement provided by SAS is: Causes a variable that is created by an INPUT or assignment statement to retain its value from one iteration of the DATA step to the next. With the retain statement there are a few options that can be used such as specifying an element list (var1 – var4) and or you can specify a default/initial value such as 0 or 1 or even a character value if set within quotes. Other options include using the _ALL_ option which retains all values from all variables that are defined earlier in the data step but not the values defined afterwards. Or if needed you can specify per type of variable such as _CHAR_ or _NUMERIC_ for character and numeric data only respectively. The retain statement can also be used to reorder the sequence of variables when bringing in a dataset to a specific order.

```
data _lab; retain subjid paramn param ady avisitn avisit;
set adam.adlb;
```

RETAINING THE COUNT

In the example below we use our sorted by age FOUR dataset, creating the SIX dataset. We retain the value of count until the value of age changes then we incrementally increase the count by 1 and hold that value again until the value of age changes.

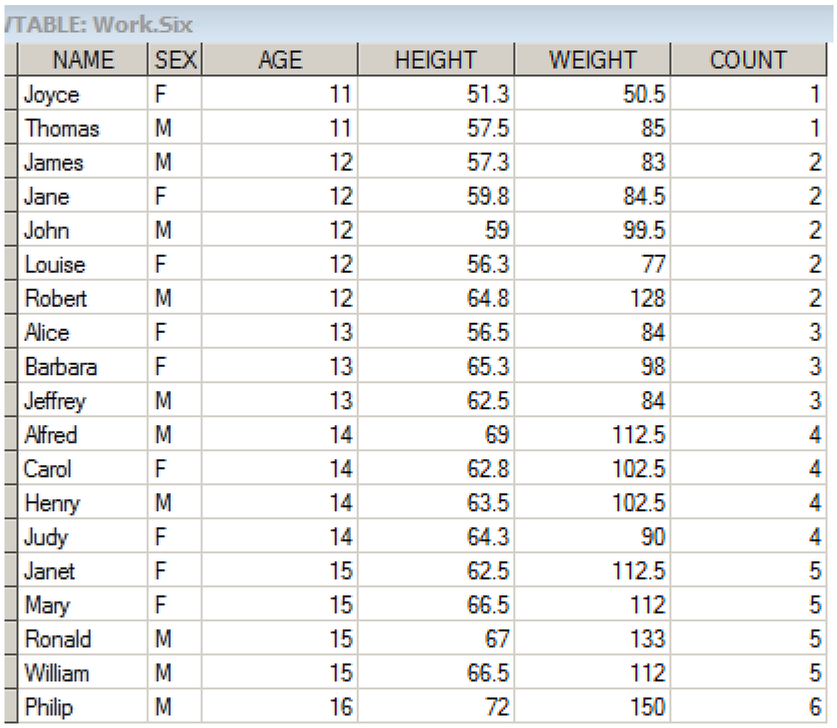
```
data six; set four;
by age;

*** holds the value of count and sets the default value ***;
retain count 0;

*** adds the value of count per each new value of age ***;
if first.age then count=count+1;

run;
```

In the resulting output, we see that now instead of a row by row count we created a count per our BY variable AGE. This is wonderful, and can be applied in so many different applications in the next section I'll provide a few example of how I have found this to be very helpful.


/TABLE: Work.Six

NAME	SEX	AGE	HEIGHT	WEIGHT	COUNT
Joyce	F	11	51.3	50.5	1
Thomas	M	11	57.5	85	1
James	M	12	57.3	83	2
Jane	F	12	59.8	84.5	2
John	M	12	59	99.5	2
Louise	F	12	56.3	77	2
Robert	M	12	64.8	128	2
Alice	F	13	56.5	84	3
Barbara	F	13	65.3	98	3
Jeffrey	M	13	62.5	84	3
Alfred	M	14	69	112.5	4
Carol	F	14	62.8	102.5	4
Henry	M	14	63.5	102.5	4
Judy	F	14	64.3	90	4
Janet	F	15	62.5	112.5	5
Mary	F	15	66.5	112	5
Ronald	M	15	67	133	5
William	M	15	66.5	112	5
Philip	M	16	72	150	6

Display 4. View of Data Six

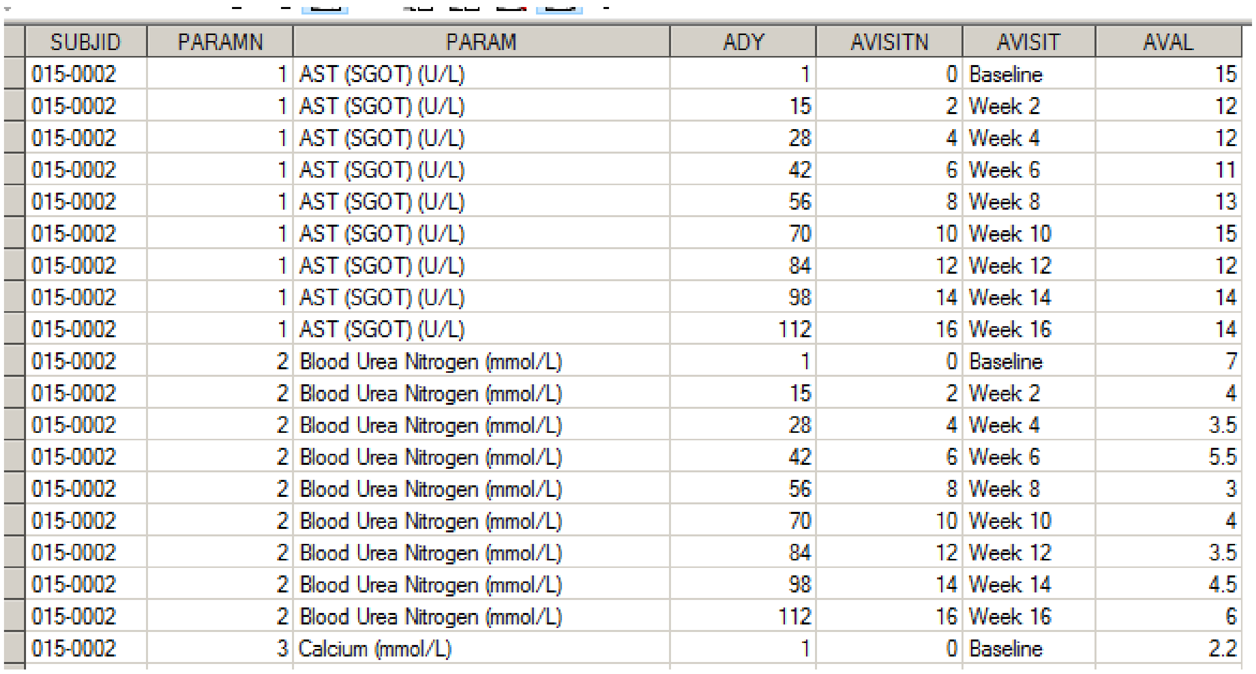
REAL WORLD SCENARIOS

I'll provide two examples where I have found the retain statement to be very helpful in providing a solution to a data problem.

A BASELINE EXAMPLE

In clinical reporting of outcomes it is very common to create a change from baseline value. The retain statement is perfect for this as we want to retain or hold the baseline value for all subsequent visits but change the value when either the parameter or subject changes.

In my sample data we have a few variables subject (subjid), numeric value of parameter (paramn), parameter (param), days of visit (ady), numeric value of visit (avisitn), analysis visit (avisit), and analysis value (aval).



SUBJID	PARAMN	PARAM	ADY	AVISITN	AVISIT	AVAL
015-0002	1	AST (SGOT) (U/L)	1	0	Baseline	15
015-0002	1	AST (SGOT) (U/L)	15	2	Week 2	12
015-0002	1	AST (SGOT) (U/L)	28	4	Week 4	12
015-0002	1	AST (SGOT) (U/L)	42	6	Week 6	11
015-0002	1	AST (SGOT) (U/L)	56	8	Week 8	13
015-0002	1	AST (SGOT) (U/L)	70	10	Week 10	15
015-0002	1	AST (SGOT) (U/L)	84	12	Week 12	12
015-0002	1	AST (SGOT) (U/L)	98	14	Week 14	14
015-0002	1	AST (SGOT) (U/L)	112	16	Week 16	14
015-0002	2	Blood Urea Nitrogen (mmol/L)	1	0	Baseline	7
015-0002	2	Blood Urea Nitrogen (mmol/L)	15	2	Week 2	4
015-0002	2	Blood Urea Nitrogen (mmol/L)	28	4	Week 4	3.5
015-0002	2	Blood Urea Nitrogen (mmol/L)	42	6	Week 6	5.5
015-0002	2	Blood Urea Nitrogen (mmol/L)	56	8	Week 8	3
015-0002	2	Blood Urea Nitrogen (mmol/L)	70	10	Week 10	4
015-0002	2	Blood Urea Nitrogen (mmol/L)	84	12	Week 12	3.5
015-0002	2	Blood Urea Nitrogen (mmol/L)	98	14	Week 14	4.5
015-0002	2	Blood Urea Nitrogen (mmol/L)	112	16	Week 16	6
015-0002	3	Calcium (mmol/L)	1	0	Baseline	2.2

Display 5. View of Example Lab Data

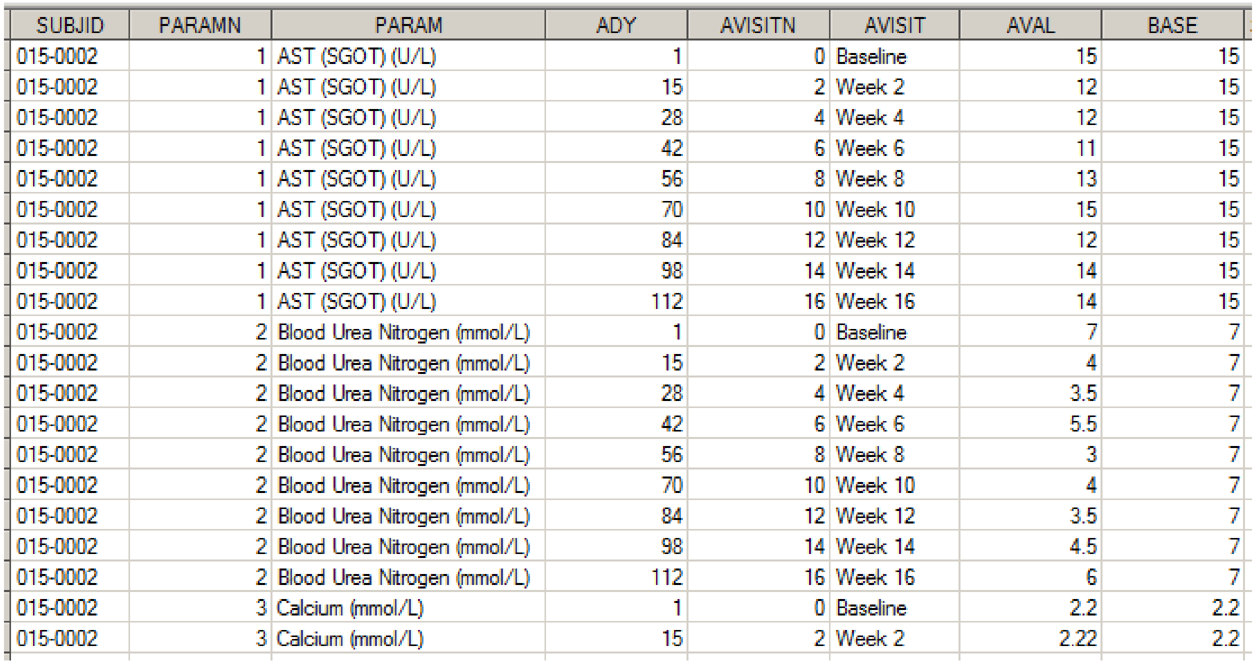
In our example above we are looking to create a baseline value using the value from the baseline visit. In our code below we bring the _LABS data by SUBJID, PARAMN, ADY and AVISITN. Then using the retain statement we hold the value of AVAL when AVIISTN=0 which is the baseline visit. However want to make sure that we reset the base value to null per each subject and parameter, this is done using the FIRST.PARAMN logic.

```
data _new;
set _labs;
by subjid paramn ady avisitn;

retain base ;
if first.paramn then base=.;
if avisitn=0 then base=aval;

run;
```

If there happen to be visits prior to the baseline value they would have a null value for the baseline value. If needed we could create a new sort variable to ensure these are placed after the baseline value prior to the datastep. So once output the variable BASE has been created which contains the retained value of AVAL.

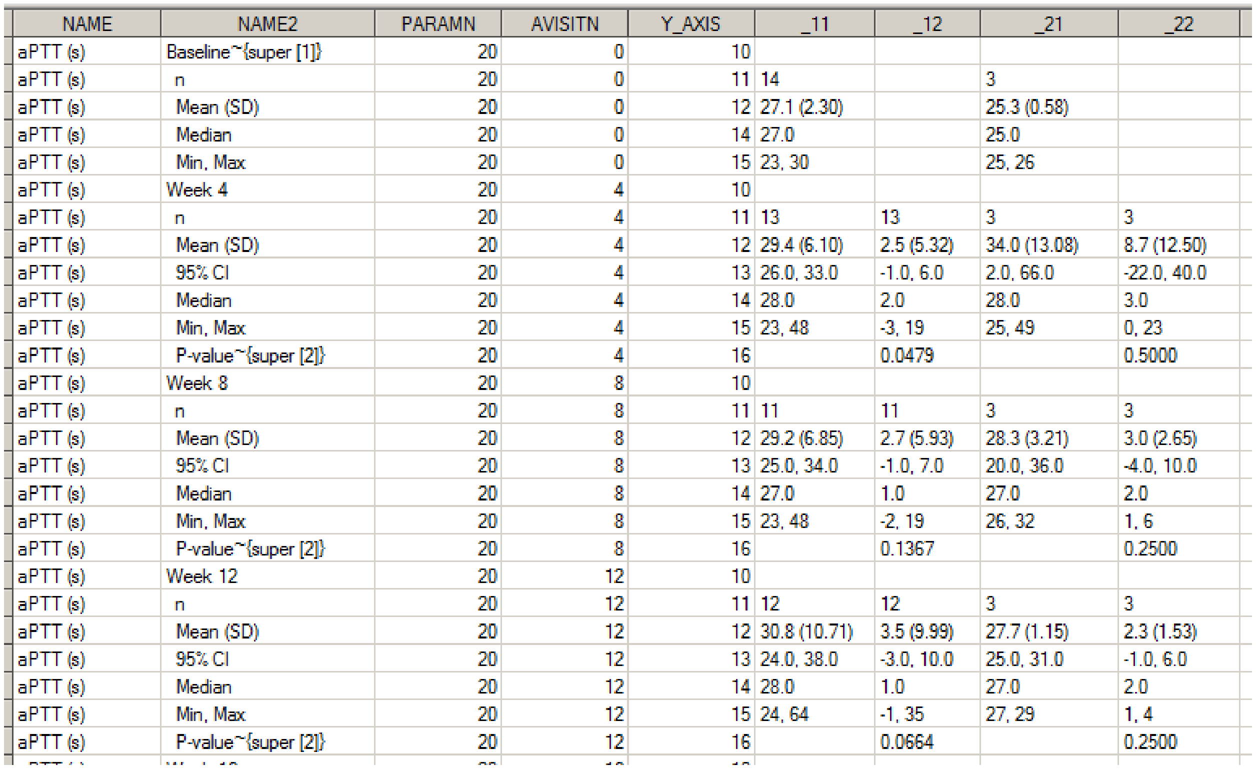


SUBJID	PARAMN	PARAM	ADY	AVISITN	AVISIT	AVAL	BASE
015-0002	1	AST (SGOT) (U/L)	1	0	Baseline	15	15
015-0002	1	AST (SGOT) (U/L)	15	2	Week 2	12	15
015-0002	1	AST (SGOT) (U/L)	28	4	Week 4	12	15
015-0002	1	AST (SGOT) (U/L)	42	6	Week 6	11	15
015-0002	1	AST (SGOT) (U/L)	56	8	Week 8	13	15
015-0002	1	AST (SGOT) (U/L)	70	10	Week 10	15	15
015-0002	1	AST (SGOT) (U/L)	84	12	Week 12	12	15
015-0002	1	AST (SGOT) (U/L)	98	14	Week 14	14	15
015-0002	1	AST (SGOT) (U/L)	112	16	Week 16	14	15
015-0002	2	Blood Urea Nitrogen (mmol/L)	1	0	Baseline	7	7
015-0002	2	Blood Urea Nitrogen (mmol/L)	15	2	Week 2	4	7
015-0002	2	Blood Urea Nitrogen (mmol/L)	28	4	Week 4	3.5	7
015-0002	2	Blood Urea Nitrogen (mmol/L)	42	6	Week 6	5.5	7
015-0002	2	Blood Urea Nitrogen (mmol/L)	56	8	Week 8	3	7
015-0002	2	Blood Urea Nitrogen (mmol/L)	70	10	Week 10	4	7
015-0002	2	Blood Urea Nitrogen (mmol/L)	84	12	Week 12	3.5	7
015-0002	2	Blood Urea Nitrogen (mmol/L)	98	14	Week 14	4.5	7
015-0002	2	Blood Urea Nitrogen (mmol/L)	112	16	Week 16	6	7
015-0002	3	Calcium (mmol/L)	1	0	Baseline	2.2	2.2
015-0002	3	Calcium (mmol/L)	15	2	Week 2	2.22	2.2

Display 6. View of Example New Lab Data

A PAGE BREAK EXAMPLE

Many times while creating a table I'll need the page breaks to be done by groups verse per individual rows per page. This can be handled very nicely using similar code as we have seen above. In the dataset below I have a parameter that has chucks of data that I do not want broken up between a page break I want all AVISITN groups to be on the same page.



NAME	NAME2	PARAMN	AVISITN	Y_AXIS	_11	_12	_21	_22
aPTT (s)	Baseline~{super [1]}	20	0	10				
aPTT (s)	n	20	0	11	14		3	
aPTT (s)	Mean (SD)	20	0	12	27.1 (2.30)		25.3 (0.58)	
aPTT (s)	Median	20	0	14	27.0		25.0	
aPTT (s)	Min, Max	20	0	15	23, 30		25, 26	
aPTT (s)	Week 4	20	4	10				
aPTT (s)	n	20	4	11	13	13	3	3
aPTT (s)	Mean (SD)	20	4	12	29.4 (6.10)	2.5 (5.32)	34.0 (13.08)	8.7 (12.50)
aPTT (s)	95% CI	20	4	13	26.0, 33.0	-1.0, 6.0	2.0, 66.0	-22.0, 40.0
aPTT (s)	Median	20	4	14	28.0	2.0	28.0	3.0
aPTT (s)	Min, Max	20	4	15	23, 48	-3, 19	25, 49	0, 23
aPTT (s)	P-value~{super [2]}	20	4	16		0.0479		0.5000
aPTT (s)	Week 8	20	8	10				
aPTT (s)	n	20	8	11	11	11	3	3
aPTT (s)	Mean (SD)	20	8	12	29.2 (6.85)	2.7 (5.93)	28.3 (3.21)	3.0 (2.65)
aPTT (s)	95% CI	20	8	13	25.0, 34.0	-1.0, 7.0	20.0, 36.0	-4.0, 10.0
aPTT (s)	Median	20	8	14	27.0	1.0	27.0	2.0
aPTT (s)	Min, Max	20	8	15	23, 48	-2, 19	26, 32	1, 6
aPTT (s)	P-value~{super [2]}	20	8	16		0.1367		0.2500
aPTT (s)	Week 12	20	12	10				
aPTT (s)	n	20	12	11	12	12	3	3
aPTT (s)	Mean (SD)	20	12	12	30.8 (10.71)	3.5 (9.99)	27.7 (1.15)	2.3 (1.53)
aPTT (s)	95% CI	20	12	13	24.0, 38.0	-3.0, 10.0	25.0, 31.0	-1.0, 6.0
aPTT (s)	Median	20	12	14	28.0	1.0	27.0	2.0
aPTT (s)	Min, Max	20	12	15	24, 64	-1, 35	27, 29	1, 4
aPTT (s)	P-value~{super [2]}	20	12	16		0.0664		0.2500

Display 7. Table Ready Dataset

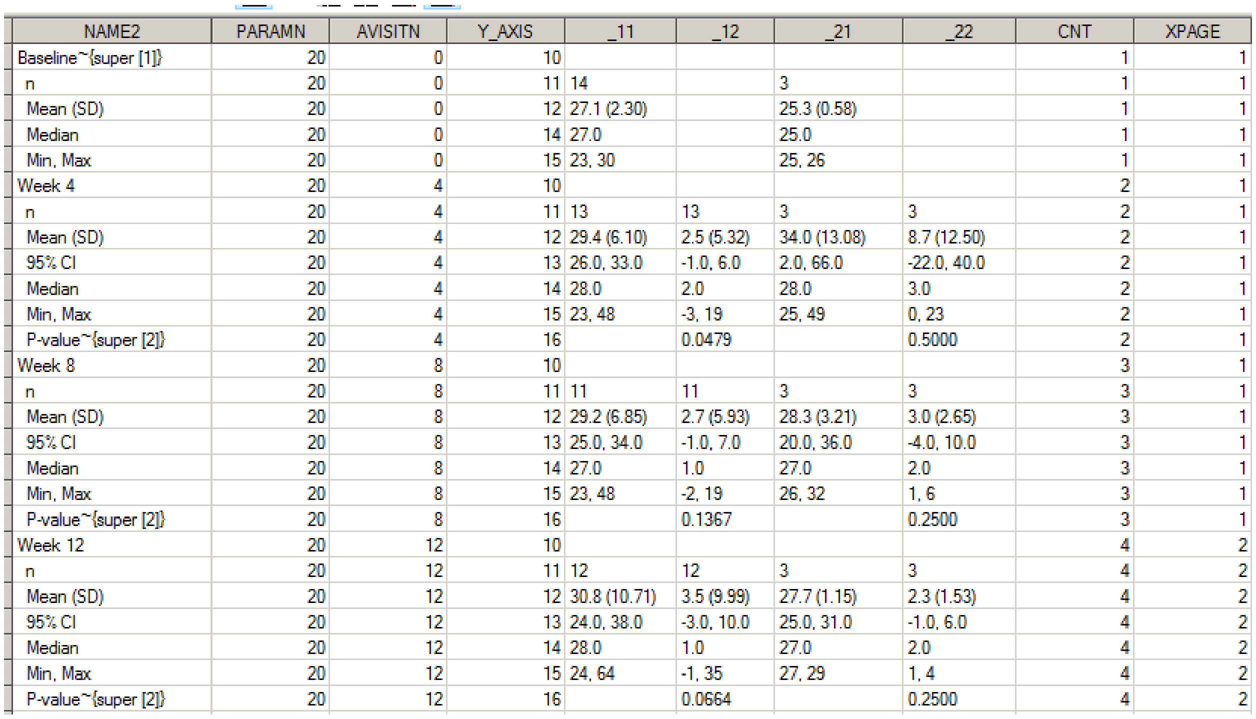
However some visits may or may not be present per parameter so I need to page breaks to be dynamic and not based off static code. I can accomplish this using this code:

```
data final;
set labs;
by paramn avisitn y_axis;

*** retain my counter var ***;
retain cnt 0;
*** create counter per each new avisitn ***;
if first.avisitn then cnt=cnt+1;
*** create new page counter per every 3 avisitns ***;
xpage=ceil(cnt/3);

run;
```

Similar to what was done in my previous examples I use a by statement to make sure my data is in the order I want and this allows me to use the FIRST.AVISITN code. I create a retain variable called CNT which changes value per each AVISITN. Then I create a third variable called XPAGE which changes value per every third count of CNT. I then use this XPAGE variable as my page break variable when creating my PROC REPORT.



NAME2	PARAMN	AVISITN	Y_AXIS	_11	_12	_21	_22	CNT	XPAGE
Baseline~{super [1]}	20	0	10					1	1
n	20	0	11	14		3		1	1
Mean (SD)	20	0	12	27.1 (2.30)		25.3 (0.58)		1	1
Median	20	0	14	27.0		25.0		1	1
Min, Max	20	0	15	23, 30		25, 26		1	1
Week 4	20	4	10					2	1
n	20	4	11	13	13	3	3	2	1
Mean (SD)	20	4	12	29.4 (6.10)	2.5 (5.32)	34.0 (13.08)	8.7 (12.50)	2	1
95% CI	20	4	13	26.0, 33.0	-1.0, 6.0	2.0, 66.0	-22.0, 40.0	2	1
Median	20	4	14	28.0	2.0	28.0	3.0	2	1
Min, Max	20	4	15	23, 48	-3, 19	25, 49	0, 23	2	1
P-value~{super [2]}	20	4	16		0.0479		0.5000	2	1
Week 8	20	8	10					3	1
n	20	8	11	11	11	3	3	3	1
Mean (SD)	20	8	12	29.2 (6.85)	2.7 (5.93)	28.3 (3.21)	3.0 (2.65)	3	1
95% CI	20	8	13	25.0, 34.0	-1.0, 7.0	20.0, 36.0	-4.0, 10.0	3	1
Median	20	8	14	27.0	1.0	27.0	2.0	3	1
Min, Max	20	8	15	23, 48	-2, 19	26, 32	1, 6	3	1
P-value~{super [2]}	20	8	16		0.1367		0.2500	3	1
Week 12	20	12	10					4	2
n	20	12	11	12	12	3	3	4	2
Mean (SD)	20	12	12	30.8 (10.71)	3.5 (9.99)	27.7 (1.15)	2.3 (1.53)	4	2
95% CI	20	12	13	24.0, 38.0	-3.0, 10.0	25.0, 31.0	-1.0, 6.0	4	2
Median	20	12	14	28.0	1.0	27.0	2.0	4	2
Min, Max	20	12	15	24, 64	-1, 35	27, 29	1, 4	4	2
P-value~{super [2]}	20	12	16		0.0664		0.2500	4	2

Display 8. Table Ready Dataset with Page Break Variable

SUMMARY

In this paper I have explained the fundamentals of the SAS Counting functions, and the RETAIN statement. I have provided several examples where I found this statement to be super useful and where it is easy to see these in action. My hope is that this will provide a solid background so that other may be able to utilize the retain statement in their SAS careers.

REFERENCES

Retain Statement Reference:
http://support.sas.com/documentation/cdl/en/lrdict/64316/HTML/default/viewer.htm#a000214163.htm

ACKNOWLEDGMENTS

Thanks goes to Albert Ortiz who always is able to solve my retain issues when my head gets too wrapped around a program to see it clearly.

CONTACT INFORMATION

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

Steve Black
Precision for Medicine Inc.
steve.black@precisionformedicine.com