Retain/Sum, PROC SORT, and First./Last.

Shannon Pileggi

STAT 330

OUTLINE

RETAIN statement

- SAS DATA steps execute *line by line* and *observation by observation*.
- In doing so, SAS makes use of the Program Data Vector (PDV).
- The PDV erases all entries each time it cycles through the observations.
- `RETAIN` allows you to keep the value of a variable in the PDV.
- This allows you to carry values forward to a new observation.

Examples of `RETAIN`

- `RETAIN month1 - month5;`
  retains the values of 5 variables (`month1` through `month5`), all initial values set to missing
- `RETAIN month1 - month5 (10 20 30 40 50);`
  retains the values of 5 variables (`month1` through `month5`), initial values set as 10, 20, 30, 40, and 50 respectively
- `RETAIN month1 - month5 1 year 0 a b c "XYZ";`
  retains the values of nine variables and sets their initial values
  - initial values of `month1` through `month5` are set to 1
  - initial value of `year` is set to 0
  - the initial values of `a`, `b`, and `c` are set to the character value '`XYZ`'.

The data

SAS Code
```
DATA kids;
INPUT famid name $ birth age wt sex $ ;
DATALINES;
1 beth 1  9  75  f
. bob  2  6  45  m
. barb 3  3  20  f
2 andy 1  8  80  m
. al   2  6  50  m
. ann  3  2  25  f
3 pete 1  6  55  m
. pam  2  4  37  f
. phil 3  2  33  m
;
RUN;
```
SAS Code

Example 1 - Fix Family ID

SAS Code
```
DATA kids2 ;

   SET kids ;

   IF famid NE . THEN newid = famid ;

   RETAIN newid ;

   famid = newid ;

   DROP newid ;

RUN ;
```
SAS Code

SUM statement

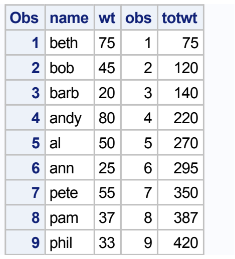
- A SUM statement looks like

  $\boxed{\textit{variable} + \textit{expression};}$

  no equal sign needed
- used to *cumulatively* add the value of an expression to a variable
- SUM is a special case of RETAIN
  - value of *expression* is added to the *variable*
  - *variable* value is retained for the next iteration of the PDV

Example 2 - Cumulative sums

SAS Code
```
DATA kids3 ;
   SET kids ;

   obs + 1 ;

   totwt + wt ;

RUN ;
```
SAS Code



Obs	name	wt	obs	totwt
1	beth	75	1	75
2	bob	45	2	120
3	barb	20	3	140
4	andy	80	4	220
5	al	50	5	270
6	ann	25	6	295
7	pete	55	7	350
8	pam	37	8	387
9	phil	33	9	420

Example 2 - Cumulative sums

SAS Code
```
DATA kids3 ;
   SET kids ;

   obs + 1 ;

   totwt + wt ;

RUN ;
```
SAS Code

What were the initial values of obs and `totwt`?

1. .
2. 0
3. 1, `wt`
4. “ ”

Example 3 - Equivalent sum statements

SAS Code
```
DATA kids4;
   SET kids;
   totwt + wt;
RUN;
```
SAS Code

- `totwt` implicitly initialized to zero
- use SUM statement *without* variable assignment (no equal sign)

SAS Code
```
DATA kids5;
   SET kids;
   RETAIN totwt 0;
   totwt = totwt + wt;
RUN;
```
SAS Code

- explicitly initialize `totwt` to zero
- use SUM statement *with* variable assignment (equal sign)

Discussion

SAS Code
```
DATA kids3 ;
   SET kids ;

   obs + 1 ;

   totwt + wt ;



RUN ;
```
SAS Code

On your own: How would I modify this code to create a *running average* of weights?

Additional `retain` notes

- the `RETAIN` statement executes once only when the program compiles
- so the placement of `RETAIN` in your data step doesn't matter (the following two sets of statements execute equivalently)

SAS Code
```
RETAIN totwt 0 ;
totwt = totwt + wt ;
```
SAS Code

SAS Code
```
totwt = totwt + wt ;
RETAIN totwt 0 ;
```
SAS Code

- **`RETAIN` only works with <u>new</u> variables**
- you <u>cannot</u> use `RETAIN` with existing variables

Retain/Sum

PROC SORT

First./Last.

PROC SORT syntax

SAS Code
```
PROC SORT DATA = originaldata OUT = newdata ;
    BY var1 var2 var3 ;
RUN ;
```
SAS Code

- the `OUT =` option is not required
  - with: *newdata* is created (copies *originaldata*) and is sorted
  - without: *originaldata* is sorted
- BY statement specifies one or more variables to sort by (variables can be either character or numeric)
  - default sorting order is ascending
  - to reverse, use DESCENDING option *before* the variable name

PROC SORT example

SAS Code
```
PROC SORT DATA = kids2 OUT = sortedkids ;
   BY DESCENDING famid sex ;
RUN ;
```
SAS Code

sortedkids
```
Obs famid kidname birth age wt sex
 1    3    pam      2     4  37  f
 2    3    pete     1     6  55  m
 3    3    phil     3     2  33  m
 4    2    ann      3     2  25  f
 5    2    andy     1     8  80  m
 6    2    al       2     6  50  m
 7    1    beth     1     9  75  f
 8    1    barb     3     3  20  f
 9    1    bob      2     6  45  m
```
sortedkids

Discussion

output
```
Obs famid kidname birth age  wt sex
1     1    barb     3     3  20  f
2     2    ann      3     2  25  f
3     3    pam      2     4  37  f
4     1    beth     1     9  75  f
5     3    phil     3     2  33  m
6     1    bob      2     6  45  m
7     2    al       2     6  50  m
8     2    andy     1     8  80  m
9     3    pete     1     6  55  m
```
output

Which BY statement was used in this PROC SORT?

1. `BY obs sex;`
2. `BY birth sex;`
3. `BY sex birth;`
4. `BY DESCENDING birth sex;`
5. `BY sex DESCENDING birth;`

Retain/Sum

PROC SORT

First./Last.

First./Last. overview

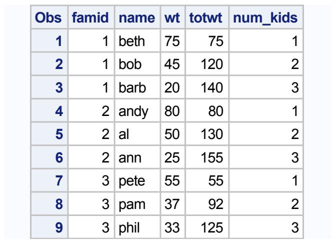
- Recall the automatic variables `_N_` and `_ERROR_`
- Two other automatic variables are FIRST.*varname* and LAST.*varname*
  - FIRST.*varname* is an indicator variable (0 or 1) that has a value of 1 when SAS processes the **first occurrence** of a new value for the variable *varname*
  - LAST.*varname* is an indicator variable (0 or 1) that has a value of 1 when SAS processes the **last occurrence** of a particular value for the variable *varname*
- To access these automatic variables,
  - use PROC SORT to sort your data BY *varname*
  - in your DATA step, use
    1. SET *sortedata* ;
    2. BY *varname* ;

Example 4 - create totals by family

SAS Code

```
PROC SORT DATA = kids2 ;
   BY famid ;
RUN ;

DATA kids6 ;
   SET kids2 ;
   BY famid ;

   IF FIRST.famid THEN DO ;
      totwt = 0 ;
      num_kids = 0 ;
   END ;

   totwt + wt ;
   num_kids + 1 ;

RUN ;
```

SAS Code



Obs	famid	name	wt	totwt	num_kids
1	1	beth	75	75	1
2	1	bob	45	120	2
3	1	barb	20	140	3
4	2	andy	80	80	1
5	2	al	50	130	2
6	2	ann	25	155	3
7	3	pete	55	55	1
8	3	pam	37	92	2
9	3	phil	33	125	3

Example 4 - create totals by family

SAS Code

```
DATA kids6 ;
   SET kids2 ;
   BY famid ;

   IF FIRST.famid THEN DO ;
      totwt = 0 ;
      num_kids = 0 ;


   END ;

   totwt + wt ;
   num_kids + 1 ;



RUN ;
```

SAS Code

On your own:

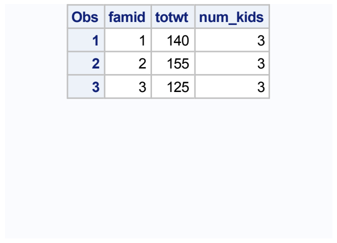
1. How could we modify this code to count the number of female and male children per family?
2. How can we view the values of the variable FIRST.famid?

Example 5 - save family level information

SAS Code

```
PROC SORT DATA = kids2 ;
   BY famid ;
RUN ;

DATA kids7 ;
   SET kids2 ;
   BY famid ;
   IF FIRST.famid THEN DO ;
      totwt = 0 ;
      num_kids = 0 ;
   END ;
   totwt + wt ;
   num_kids + 1 ;
   IF LAST.famid THEN OUTPUT;
   KEEP famid totwt num_kids;
RUN;
```

SAS Code



Obs	famid	totwt	num_kids
1	1	140	3
2	2	155	3
3	3	125	3