Multiple SET Statements in a DATA Step

Why and how to use them

Thomas Wollseifen

10 May 2016

inVentiv Health

Topics

1. SET statement very easy
2. How multiple SETs work or don‘t work
3. Excursion: Program data vector (PDV)
4. SQL vs. Multiple SETs
5. Do-loops or DoW-loops

© 2016 All Rights Reserved | CONFIDENTIAL

1) 

© 2016 All Rights Reserved | CONFIDENTIAL

DATA STEP – Basic scenarios

```
data C;
    set B;
    *Manipulate data here;
run;
```


© 2016 All Rights Reserved | CONFIDENTIAL

DATA STEP – Basic scenarios

```
data C;
    set B;
    *Manipulate data here;
run;
```

```
data C;
    set A  B;
    *Manipulate data here;
run;
```

A

Name	Age
Amy	32
Jake	28
Tom	55

B

Name	Age
Ben	36
Mary	35

C unsorted

Name	Age
Amy	32
Jake	28
Tom	55
Ben	36
Mary	35

© 2016 All Rights Reserved | CONFIDENTIAL

DATA STEP – Interleaving

```
data C;
    set  A  B;
    by  name;
    *Manipulate data here;
run;
```

C (sorted)

Name	Age
Amy	32
Ben	36
Jake	28
Mary	35
Tom	55

© 2016 All Rights Reserved | CONFIDENTIAL

2) How multiple SETs work or don‘t work

© 2016 All Rights Reserved | CONFIDENTIAL

Multiple SET statements – How they work (or don‘t)

```
data overwriteC;
    set A;
    set B;
run;
```

A

Name	Gender
Amy	F
Jake	M
Tom	M

B

Name	Age
Ben	36
Mary	35

© 2016 All Rights Reserved | CONFIDENTIAL

Multiple SET statements – How they work (or don‘t)

```
data overwriteC;
    set A;
    set B;
run;
```

A

Name	Gender
Amy	F
Jake	M
Tom	M

B

Name	Age
Ben	36
Mary	35

overwriteC

Name	Gender	Age
~~Amy~~ Ben	F	36
~~Jake~~ Mary	M	35

© 2016 All Rights Reserved | CONFIDENTIAL

Multiple SET statements – How they work (or don‘t)

```
data overwriteC;
    set A;
    set B;
run;
```

A

Name	Gender
Amy	F
Jake	M
Tom	M

B

Name	Age
Ben	36
Mary	35

overwriteC

Name	Gender	Age
~~Amy~~ Ben	F	36
~~Jake~~ Mary	M	35

1) Reads 1st obs from A and places it into the PDV

_N_	_Error_	Name	Gender
1	0	Amy	F

2) Reads 1st obs from B and places it into the PDV

SAS overwrites values of common variables from A with the new values from B

_N_	_Error_	Name	Gender	Age
1	0	Ben	F	36

© 2016 All Rights Reserved | CONFIDENTIAL

Excursion:
3) Program data vector (PDV)

© 2016 All Rights Reserved | CONFIDENTIAL

PDV – Program data vector

- “A logical area in memory where SAS builds a data set, one observation at a time.”

- **Automatic variables** created by the DATA step.
  - Retained from one iteration of the DATA step to the next
  - Not written to the output data set
  - Cannot be kept, dropped, or renamed
  - **_N_**: number of times the DATA step has iterated
  - **_ERROR_** (0= no error / 1= error)
  - **first.*BY-variable*** and **last.*BY-variable***
- **Temporary variables**
  - **IN**=*variable* (0 / 1)
  - **END**=*variable*: indicates the end of the input data reached.

© 2016 All Rights Reserved | CONFIDENTIAL

PDV – Program data vector

```
data a;
   put "after compile before execution: " _all_;
   do i=1 to 5;
      y=i*2;
      output;
      put _all_;
   end;
   put "at end of execution:" _all_;
run;
```

SAS LOG:

after compile before execution:
I= Y= _ERROR_=0 _N_=1
I=1 Y=2 _ERROR_=0 _N_=1
I=2 Y=4 _ERROR_=0 _N_=1
I=3 Y=6 _ERROR_=0 _N_=1
I=4 Y=8 _ERROR_=0 _N_=1
I=5 Y=10 _ERROR_=0 _N_=1
at end of execution:I=6 Y=10 _ERROR_=0 _N_=1
NOTE: The data set WORK.A has 5 observations and 2 variables.

© 2016 All Rights Reserved | CONFIDENTIAL

PDV – Program data vector

```
data a;
   set a(in=inoperator)  end=eof;
   by i;
   put _all_;
run;
```

```
INOPERATOR=1 EOF=0 I=1 Y=2 FIRST.I=1 LAST.I=1 _ERROR_=0 _N_=1
INOPERATOR=1 EOF=0 I=2 Y=4 FIRST.I=1 LAST.I=1 _ERROR_=0 _N_=2
INOPERATOR=1 EOF=0 I=3 Y=6 FIRST.I=1 LAST.I=1 _ERROR_=0 _N_=3
INOPERATOR=1 EOF=0 I=4 Y=8 FIRST.I=1 LAST.I=1 _ERROR_=0 _N_=4
INOPERATOR=1 EOF=1 I=5 Y=10 FIRST.I=1 LAST.I=1 _ERROR_=0 _N_=5

NOTE: There were 5 observations read from the data set WORK.A
```


© 2016 All Rights Reserved | CONFIDENTIAL

4a) Summaries with SQL vs. Multiple SETs

© 2016 All Rights Reserved | CONFIDENTIAL

Example: Calculate summary statistics with SQL

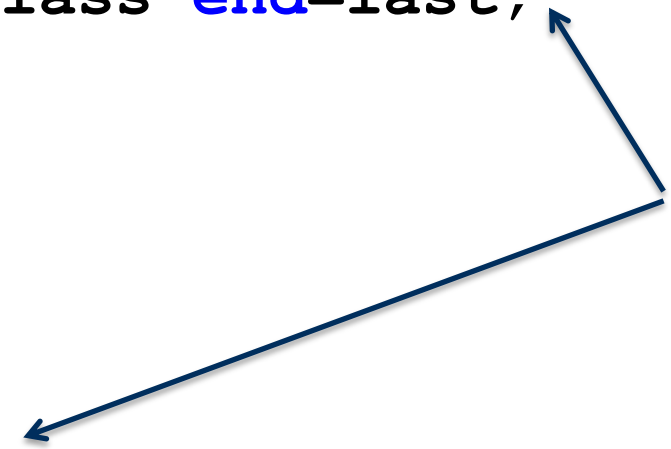
```
proc sql;
   create table b as select
         *
         ,sum(age) as sum_age
         ,count(*) as n_age
         ,mean(age) as m_age
   from sashelp.class;
quit;
```

	Name	Sex	Age	Height	Weight	sum_age	n_age	m_age
1	Alfred	M	14	69	112.5	253	19	13.315789...
2	Alice	F	13	56.5	84	253	19	13.315789...
3	Barbara	F	13	65.3	98	253	19	13.315789...
4	Carol	F	14	62.8	102.5	253	19	13.315789...
5	Henry	M	14	63.5	102.5	253	19	13.315789...
6	James	M	12	57.3	83	253	19	13.315789...
7	Jane	F	12	59.8	84.5	253	19	13.315789...
8	Janet	F	15	62.5	112.5	253	19	13.315789...
9	Jeffrey	M	13	62.5	84	253	19	13.315789...
10	John	M	12	59	99.5	253	19	13.315789...
11	Joyce	F	11	51.3	50.5	253	19	13.315789...
12	Judy	F	14	64.3	90	253	19	13.315789...
13	Louise	F	12	56.3	77	253	19	13.315789...
14	Mary	F	15	66.5	112	253	19	13.315789...
15	Philip	M	16	72	150	253	19	13.315789...
16	Robert	M	12	64.8	128	253	19	13.315789...
17	Ronald	M	15	67	133	253	19	13.315789...
18	Thomas	M	11	57.5	85	253	19	13.315789...
19	William	M	15	66.5	112	253	19	13.315789...

© 2016 All Rights Reserved | CONFIDENTIAL

Summary statistics with multiple SETs

```
data b;
   if _n_=1 then
  do until (last);
        set sashelp.class end=last;
        n+1;
        sum+age;
  end;
   m_age=sum/n;
   set sashelp.class;
run;
```



© 2016 All Rights Reserved | CONFIDENTIAL

Summary statistics with multiple SETs

```
data a;
  do _n_ = 1 by 1 until (last.sex);
    set class;
    by sex ;
    total = sum(total, age) ;
  end ;
   mean = total / _n_ ;
  do until (last.sex) ;
    set class;
    by sex;
    output;
  end;
run;
```

grouped summary

	NAME	SEX	AGE	HEIGHT	WEIGHT	TOTAL	MEAN
1	Alice	F	13	56.5	84	119	13.222222222
2	Barbara	F	13	65.3	98	119	13.222222222
3	Carol	F	14	62.8	102.5	119	13.222222222
4	Jane	F	12	59.8	84.5	119	13.222222222
5	Janet	F	15	62.5	112.5	119	13.222222222
6	Joyce	F	11	51.3	50.5	119	13.222222222
7	Judy	F	14	64.3	90	119	13.222222222
8	Louise	F	12	56.3	77	119	13.222222222
9	Mary	F	15	66.5	112	119	13.222222222
10	Alfred	M	14	69	112.5	134	13.4
11	Henry	M	14	63.5	102.5	134	13.4
12	James	M	12	57.3	83	134	13.4
13	Jeffrey	M	13	62.5	84	134	13.4
14	John	M	12	59	99.5	134	13.4
15	Philip	M	16	72	150	134	13.4
16	Robert	M	12	64.8	128	134	13.4
17	Ronald	M	15	67	133	134	13.4
18	Thomas	M	11	57.5	85	134	13.4
19	William	M	15	66.5	112	134	13.4

© 2016 All Rights Reserved | CONFIDENTIAL

Multiple Sets in do-loops / „Change from Baseline“

```
data vs;
  do until(last.subjidn);
     set vs;
     by subjidn paramcd;
     if ablfn=1  then base = aval;
     else chg = aval - base;
     output;
  end;
run;
```

	SUBJIDN	PARAMCD	VISITNUM	ABLFN	AVAL	BASE	CHG
1	1	Systolic Bloo…	0.0000		105.000		
2	1	Systolic Bloo…	1.1000		100.000		
3	1	Systolic Bloo…	1.1000	Y	100.000	100	
4	1	Systolic Bloo…	1.2800		105.000	100	5
5	1	Systolic Bloo…	1.5600		102.000	100	2
6	1	Systolic Bloo…	3.5600		95.000	100	-5
7	1	Systolic Bloo…	9900.0000		100.000	100	0
8	7	Systolic Bloo…	0.0000		120.000		
9	7	Systolic Bloo…	1.1000		115.000		
10	7	Systolic Bloo…	1.1000	Y	115.000	115	
11	7	Systolic Bloo…	1.2800		120.000	115	5
12	7	Systolic Bloo…	4.1400		135.000	115	20
13	7	Systolic Bloo…	6.1400		125.000	115	10
14	7	Systolic Bloo…	9900.0000		115.000	115	0

© 2016 All Rights Reserved | CONFIDENTIAL

4b)
Many-to-many merge
SQL
or
multiple SET?

© 2016 All Rights Reserved | CONFIDENTIAL

Many-to-many merge

Adverse Events

	SUBJIDN	AELLT	ASTDT	AEENDT
1	7	Stomachache	01NOV2013	01NOV2013
2	7	Headache	27OCT2013	27OCT2013
3	7	Common cold	05DEC2013	11DEC2013
4	7	Sore throat	29NOV2013	11DEC2013
5	7	Backache	17JAN2014	19JAN2014
6	7	Stomachache	19JAN2014	19JAN2014
7	7	Backache	31JAN2014	31JAN2014
8	7	Vomiting	10JUN2014	10JUN2014
9	7	Clavicle frac...	13JUL2014	25AUG2014
10	7	Fracture pain	13JUL2014	25AUG2014
11	7	Nausea	13JUL2014	20JUL2014
12	10	Cystitis	07NOV2013	09NOV2013
13	10	Dental impla...	27JAN2014	27JAN2014
14	10	Cystitis	27AUG2014	30SEP2014

Exposure

	SUBJIDN	VISITNUM	EXDT	TRTAN
1	7	1.0000	05SEP2013	3
2	7	2.0000	03OCT2013	3
3	7	3.0000	02NOV2013	3
4	7	4.0000	29NOV2013	3
5	7	5.0000	27DEC2013	3
6	7	6.0000	24JAN2014	3
7	7	7.0000	21FEB2014	3
8	7	8.0000	21MAR2014	3
9	7	9.0000	18APR2014	3
10	10	1.0000	04NOV2013	1
11	10	2.0000	31DEC2013	1
12	10	3.0000	21FEB2014	1
13	10	4.0000	23APR2014	1
14	10	5.0000	17JUN2014	1
15	10	6.0000	11AUG2014	1

- SQL join ☺
- Merge ☠💣

```
NOTE: MERGE statement has more
than one data set with repeats of BY values
```


© 2016 All Rights Reserved | CONFIDENTIAL

inVentiv Health

Why many-to-many merge fails

The **merge** statement fails to produce the correct results due to two important SAS rules:

1. In a data step, SAS reads observations sequentially.

2. Once an observation is read into the PDV, it is never re-read.

© 2016 All Rights Reserved | CONFIDENTIAL

Task: Merge Adverse Events with Exposure where treatment started 30 days before AE

Adverse Events

	SUBJIDN	AELLT	ASTDT	AEENDT
1	7	Stomachache	01NOV2013	01NOV2013
2	7	Headache	27OCT2013	27OCT2013
3	7	Common cold	05DEC2013	11DEC2013
4	7	Sore throat	29NOV2013	11DEC2013
5	7	Backache	17JAN2014	19JAN2014
6	7	Stomachache	19JAN2014	19JAN2014
7	7	Backache	31JAN2014	31JAN2014
8	7	Vomiting	10JUN2014	10JUN2014
9	7	Clavicle frac...	13JUL2014	25AUG2014
10	7	Fracture pain	13JUL2014	25AUG2014
11	7	Nausea	13JUL2014	20JUL2014
12	10	Cystitis	07NOV2013	09NOV2013
13	10	Dental impla...	27JAN2014	27JAN2014
14	10	Cystitis	27AUG2014	30SEP2014

Exposure

	SUBJIDN	VISITNUM	EXDT	TRTAN
1	7	1.0000	05SEP2013	3
2	7	2.0000	03OCT2013	3
3	7	3.0000	02NOV2013	3
4	7	4.0000	29NOV2013	3
5	7	5.0000	27DEC2013	3
6	7	6.0000	24JAN2014	3
7	7	7.0000	21FEB2014	3
8	7	8.0000	21MAR2014	3
9	7	9.0000	18APR2014	3
10	10	1.0000	04NOV2013	1
11	10	2.0000	31DEC2013	1
12	10	3.0000	21FEB2014	1
13	10	4.0000	23APR2014	1
14	10	5.0000	17JUN2014	1
15	10	6.0000	11AUG2014	1

© 2016 All Rights Reserved | CONFIDENTIAL

Task: Merge Adverse Events with Exposure where treatment started 30 days before AE

```
proc sql;
   create table ae_ex as
         select l.*
                ,r.visitnum
                ,r.exdt
                ,r.trtan
   from ae as l
   join ex as r
   on l.subjidn=r.subjidn and 0<=astdt-exdt<=30
   order by subjidn, aellt, visitnum;
quit;
```

	SUBJIDN 1	AELLT 2	ASTDT	AEENDT	VISITNUM 3	EXDT	TRTAN
1	7	Backache	17JAN2014	19JAN2014	5.0000	27DEC2013	3
2	7	Backache	31JAN2014	31JAN2014	6.0000	24JAN2014	3
3	7	Common cold	05DEC2013	11DEC2013	4.0000	29NOV2013	3
4	7	Headache	27OCT2013	27OCT2013	2.0000	03OCT2013	3
5	7	Sore throat	29NOV2013	11DEC2013	3.0000	02NOV2013	3
6	7	Sore throat	29NOV2013	11DEC2013	4.0000	29NOV2013	3
7	7	Stomachache	01NOV2013	01NOV2013	2.0000	03OCT2013	3
8	7	Stomachache	19JAN2014	19JAN2014	5.0000	27DEC2013	3
9	10	Cystitis	07NOV2013	09NOV2013	1.0000	04NOV2013	1
10	10	Cystitis	27AUG2014	30SEP2014	6.0000	11AUG2014	1
11	10	Dental impla…	27JAN2014	27JAN2014	2.0000	31DEC2013	1

© 2016 All R

Task: Merge Adverse Events with Exposure where treatment started 30 days before AE

```
data ae_ex2(drop=idnum);
     set ae;
     do i=1 to num;
          set ex(rename=(subjidn=idnum)) nobs=num point=i;
          if subjidn=idnum and  0<=astdt-exdt<=30 then output;
     end;
run;
```

	SUBJIDN	AELLT	ASTDT	AEENDT	VISITNUM	EXDT	TRTAN
1	7	Stomachache	01NOV2013	01NOV2013	2.0000	03OCT2013	3
2	7	Headache	27OCT2013	27OCT2013	2.0000	03OCT2013	3
3	7	Common cold	05DEC2013	11DEC2013	4.0000	29NOV2013	3
4	7	Sore throat	29NOV2013	11DEC2013	3.0000	02NOV2013	3
5	7	Sore throat	29NOV2013	11DEC2013	4.0000	29NOV2013	3
6	7	Backache	17JAN2014	19JAN2014	5.0000	27DEC2013	3
7	7	Stomachache	19JAN2014	19JAN2014	5.0000	27DEC2013	3
8	7	Backache	31JAN2014	31JAN2014	6.0000	24JAN2014	3
9	10	Cystitis	07NOV2013	09NOV2013	1.0000	04NOV2013	1
10	10	Dental impla...	27JAN2014	27JAN2014	2.0000	31DEC2013	1
11	10	Cystitis	27AUG2014	30SEP2014	6.0000	11AUG2014	1

© 2016 All Rights Reserved | CONFIDENTIAL

5) Do-loops or DoW-loops?

© 2016 All Rights Reserved | CONFIDENTIAL

Basic approach: DoW-loops / multiple Sets

- Named after **Ian Whitlock** and **Don Henderson**.
- Uses DO loops to read data for tasks which require break-event processing, such as:
  - BY-group processing (FIRST. and LAST.)
  - checking for a specific value or a missing value

© 2016 All Rights Reserved | CONFIDENTIAL

Basic approach: DoW-loops / multiple Sets

```
DATA ...;
   <Stuff done before break-event>;
  DO<Index Specs> UNTIL(Break-Event);
      Set A;
      <Stuff done for each record>;
  END;
   <Stuff done after break-event... >;
RUN;
```

Separates before-, during, after- actions →intuitively logic

© 2016 All Rights Reserved | CONFIDENTIAL

inVentiv Health

Reference

- ***In Lockstep with the DoW-Loop***, Paul M. Dorfman, SESUG 2011 proceedings, http://analytics.ncsu.edu/sesug/2011/SS01.Dorfman.pdf

© 2016 All Rights Reserved | CONFIDENTIAL

Thank you very much for your attention!

Questions & Answers

thomas.wollseifen@inventivhealth.com

© 2016 All Rights Reserved | CONFIDENTIAL