

- Modifying and Combining SAS Data Sets
- Merging with registry data

Åsa Klint

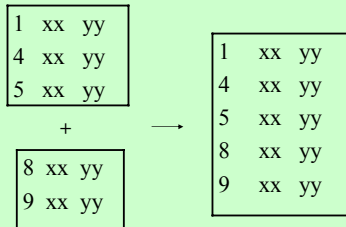
Modifying or Making a copy of a Data Set Using the SET Statement

```
data descr;
  set annat.descr_raw;
  if age<35 and sex=2;
  height_m=height/100;
  bmi=weight/(height_m**2);
  bmi=round(bmi,.1);
run;
proc print data=descr;
run;
```

Obs	patnr	age	sex	height	weight	bmi	smoke
1	2	30	2	160	55	21.5	0
2	6	34	2	192	90	24.4	1

Stacking two (or more) data sets

Data sets that have the same variables but contains different observations can be stacked on top of each other via the SET statement.



Stacking Data Sets Using the SET Statement

```
proc print data=descr1;
  var patnr centre age sex height weight;
run;
```

Obs	patnr	centre	age	sex	height	weight
1	11	KS	43	2	172	75
2	12	KS	30	2	160	55

```
proc print data=descr2;
  var patnr centre age sex height weight;
run;
```

Obs	patnr	centre	age	sex	height	weight
1	21	SOS	70	2	165	70
2	22	SOS	25	1	168	61
3	23	SOS	34	2	192	90

With two or more data sets in the Set statement SAS stacks the data sets on top of each other

```
data all;
  set descr1 descr2;
run;
proc print data=all;
  var patnr centre age sex height weight;
run;
```

Obs	patnr	centre	age	sex	height	weight
1	1	KS	43	2	172	75
2	2	KS	30	2	160	55
3	11	SOS	70	2	165	70
4	12	SOS	25	1	168	61
5	13	SOS	34	2	192	90

If only one of the data sets (here descr2) includes the variable smoke:

Obs	patnr	centre	age	height	weight	smoke
1	1	KS	43	172	75	.
2	2	KS	30	160	55	.
3	11	SOS	70	165	70	0
4	12	SOS	25	168	61	0
5	13	SOS	34	192	90	1

Merge statement

MERGE is a flexible statement that allows us to combine several related data sets into one.

Merge the data sets using a matching variable common to both data sets in order to avoid errors. This is done in a **BY** statement. The matching variable would typically be a unique id-number

1 Xx yy	+	1 zz	→	1 Xx yy zz
2 xx yy		2 zz		2 Xx yy zz
3 Xx yy		5 zz		3 Xx yy .
5 xx yy				5 Xx yy zz

Åsa Klint, november 2003

7

Note that all data sets must be sorted by the matching variable before merging.

Pay attention to the variable-names.

If the two data sets have variables with the same names (except the **BY** variable) then the variables from the second dataset will overwrite the variables from the first.

Check before merging and solve the problem through renaming of variables for example.

Warning:

using the **MERGE** statement without a matching variable (**BY** statement) can lead to errors. It is recommended that you always include a **BY** statement but beware that SAS accepts merging without it as well.

Åsa Klint, november 2003

8

Example of Combining Data Sets Using the MERGE Statement

DESCR						
Obs	patnr	age	sex	height	weight	smoke
1	1	43	2	172	75	0
2	2	30	2	180	55	0
3	3	27	1	180	94	1
4	4	70	2	165	70	0

BLOODPR			
Obs	patnr	week	sbp
1	1	1	143
2	1	2	140
3	2	1	120
4	2	2	125
5	3	1	135
6	3	2	131
7	4	1	151
8	4	2	146

Åsa Klint, november 2003

9

```
proc sort data=descr1;
  by patnr;
proc sort data=bloodpr;
  by patnr;
run;

data all;
  merge descr1 bloodpr;
  by patnr;
proc print data=all;
  var patnr sex height week sbp;
run;
```

Åsa Klint, november 2003

10

Data set ALL

Obs	patnr	sex	height	week	sbp
1	1	2	172	1	143
2	1	2	172	2	140
3	2	2	180	1	120
4	2	2	180	2	125
5	3	1	180	1	135
6	3	1	180	2	131
7	4	2	165	1	151
8	4	2	165	2	146

Åsa Klint, november 2003

11

Example: Something went wrong in Combining Data Sets using the MERGE Statement

```
data all;
  merge descr1 bloodpr;
proc print data=all;
  var patnr age sex height week sbp;
run;
```

Obs	patnr	age	sex	height	week	sbp
1	1	43	2	172	1	143
2	1	30	2	180	2	140
3	2	27	1	180	1	120
4	2	70	2	165	2	125
5	3	.	.	.	1	135
6	3	.	.	.	2	131
7	4	.	.	.	1	151
8	4	.	.	.	2	146

So don't forget the **BY** statement!

Åsa Klint, november 2003

12

Using SAS Data Set Options

The most frequently used data set options:

Options:	Tells SAS:
KEEP = <i>variable list</i>	variables to keep
DROP = <i>variable list</i>	variables to drop
RENAME = (<i>oldvar</i> = <i>newvar</i>)	rename variable
FIRSTOBS = <i>n</i>	start reading at obs <i>n</i>
OBS = <i>n</i>	stop reading at obs <i>n</i>
IN = <i>newvar-name</i>	temporary variable for tracking whether that data set contributed to the current observation.

Åsa Klint, november 2003

13

Using SAS Data Set Options

```
data in1;
  set annat.descr_raw (keep = patnr age sex smoke);

data in2;
  set annat.descr_raw (drop = height weight bmi);

data in3;
  set annat.descr_raw (rename = (height = heightcm));

proc print data=in3 (firstobs=2 obs=4);
run;
```

Åsa Klint, november 2003

14

Tracking observations using the IN= option (IN creates a new temporary variable)

```
data all;
  merge descr1 (in=indesc) bloodpr (in=inbp);
  by patnr;
```

The variables 'indesc' and 'inbp' exist only for the duration of current data step and are not added to the data set being created.

SAS gives IN=variables a value of 0 if that data set did not contribute to the current observation and a value of 1 if it did.

Åsa Klint, november 2003

15

Tracking and selecting observations with the IN=option

DESCR			BLOODPR		
Obs	patnr	age	Obs	patnr	sbp
1	12	43	1	12	150
2	13	30	2	13	128
3	14	27	3	14	122
			4	15	120

```
data combined;
  merge descr(in=indesc) bloodpr;
  by patnr;
  if indesc=1;
run;

/* Only patients that were in the data
set descr will be included in the
new combined data set*/
```

Åsa Klint, november 2003

16

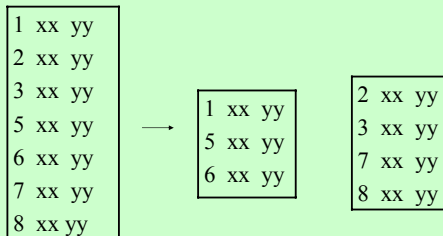
Tracking and selecting observations with the IN=option

Obs	patnr	age	sbp
1	12	43	150
2	13	30	128
3	14	27	122

Åsa Klint, november 2003

17

Creating several data sets from one using the OUTPUT statement



Åsa Klint, november 2003

18

Writing Multiple Data Sets Using the OUTPUT Statement

```

Data set RESULTS
-----
Obs   stud_nr   test   scores
1     100       1      56
2     100       2      40
3     100       3      89
4     101       1      60
5     101       2      55

data score1 score2 score3;
  set results;
  if test=1 then output score1;
  if test=2 then output score2;
  if test=3 then output score3;
run;

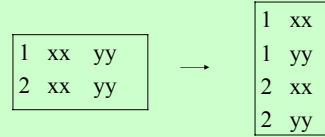
Data set SCORE1
-----
Obs   stud_nr   test   scores
1     100       1      56
2     101       1      60
  
```

Åsa Klint, november 2003

19

Making Several Observations from One Using the OUTPUT Statement

Usually SAS writes an observation to a data set at the end of the DATA step, but you can override this default using the OUTPUT statement. This statement gives you control over when an observation is written to a SAS data set.



Åsa Klint, november 2003

20

```

data generate;
  do x=1 to 6;
    y=x**2;
    output;
  end;
run;

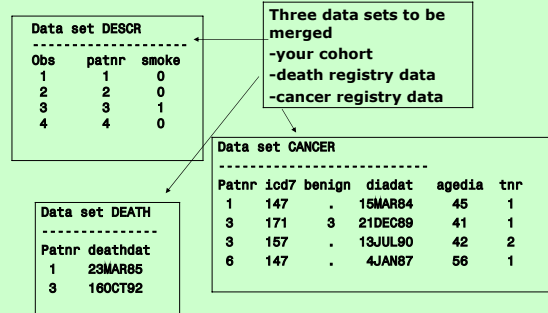
DO loop with 6 iterations.
Without the OUTPUT, SAS would
have written only 1 obs.
at the end of the data step.
  
```

Obs	x	y
1	1	1
2	2	4
3	3	9
4	4	16
5	5	25
6	6	36

Åsa Klint, november 2003

21

Example: Merging your data with registry data



Åsa Klint, november 2003

22

```

Proc sort data=descr;
  by patnr;
Proc sort data=cancer;
  by patnr;
Proc sort data=death;
  by patnr;
Run;

data analysis;
  merge descr(in=indescr) cancer(keep=patnr icd7 benign
  diadat) death;
  by patnr;
  if indescr=1 and benign ne 3;
run;
  
```

Make sure the data sets are sorted according to the matching variable

Keeping the relevant variables

Making sure we only keep individuals belonging to our cohort and excluding benign (tumours in this case)

Åsa Klint, november 2003

23

Data set Analysis

Patnr	smoke	icd7	benign	diadat	deathdat
1	0	147	.	15MAR84	23MAR85
2	0
3	1	157	.	13JUL90	16OCT92
4	0

Åsa Klint, november 2003

24

Using SAS Automatic Variables

FIRST.variable and LAST.variable: Available only in special circumstances i.e when you are using the BY statement in a data step.

Example: *FIRST.variable*

```
proc sort data=cancer;
  by patnr diadat;
```

```
/* Include only the first diagnosis of cancer*/
data firstcan;
  set cancer;
  by patnr diadat;
  if first.lopnr;
run;
```

Data set CANCER			
Patnr	icd7	benign	diadat
1	147	.	15MAR84
3	171	3	21DEC89
3	157	.	13JUL90

Åsa Klint, november 2003

25

Data set FIRSTCAN

Patnr	icd7	benign	diadat
1	147	.	15MAR84
3	171	3	21DEC89

Åsa Klint, november 2003

26

Flipping your data Using PROC TRANSPOSE

The TRANSPOSE procedure creates a restructured data set where observations have been turned into variables or variables into observations.

The output data set can be used in subsequent DATA or PROC steps for analysis, reporting, or further data manipulation.

X Y Z

1	A	ZZ
1	B	ZZ
2	A	ZZ
2	B	ZZ

→

X A B _NAME_

1	aa	bb	z
2	aa	bb	z

Åsa Klint, november 2003

27

```
proc transpose data = oldname out = newname;
  by variable-list;
  id variable;
  var variable-list;
run;
```

BY statement: for grouping variables you want to keep as variables

ID statement: names variable whose formatted values will become the new variable names

VAR statement: names the variables whose values you want to transpose

Åsa Klint, november 2003

28

Example: Using PROC TRANSPOSE

BLOODPR

Obs	patnr	week	sbp
1	1	1	143
2	1	2	140
3	2	1	120
4	2	2	125
5	3	1	135
6	3	2	131
7	4	1	151
8	4	2	148

?

BLOODPR

Obs	patnr	week	sbp
1	1	1	143
2	1	2	140
3	2	1	120
4	2	2	125
5	3	1	135
6	3	2	131
7	4	1	151
8	4	2	148

WISH

patnr	sbp_w1	sbp_w2
1	143	140
2	120	125
3	135	131
4	151	148

Åsa Klint, november 2003

29

Åsa Klint, november 2003

30

```
proc transpose data=bloodpr out=bp;
  by patnr;
  id week;
  var sbp;
run;
```

```
-----
Obs  patnr  _NAME_  _1  _2
  1    1    sbp    143 140
  2    2    sbp    120 125
  3    3    sbp    135 131
  4    4    sbp    151 146
```

Åsa Klint, november 2003

31

Thank you for attending this seminar!

Upcoming seminar:
Paul Dickman will give a SAS seminar at 1 pm,
tuesday Nov 18 in this room

Åsa Klint, november 2003

32