

ID statement in Proc Transpose: why, how and when we have to use it.

Alexander M. Feigin (Synthes USA, West Chester, PA feiginy@aol.com)

Proc Transpose is an intensively used tool in Base SAS. It has a simple syntax and is very effective in terms of exchanging the significance of row and column identifiers. This paper discusses two issues: 1) the inappropriate usage of Proc Transpose without the ID statement and 2) creating complex ID variables in order to transpose across more than one variable simultaneously.

Though it is clearly formulated in all manuals that in order to transpose across the variable, this variable should be used as an ID one, it is widely spread mistake to rely upon SAS in choosing a certain variable as ID one without even bothering to write the ID statement. The reason for this mistake is an ability of SAS to perform transposing without ID statement. In some cases the absence of ID statement does not harm the performance of Proc Transpose. In other cases it does.

For example, below we create the data set ONE which contains information on pulse, sex and treatment group (trt) for every single subject (pid).

```
data one ;
  input pid trt sex pulse;
  cards;
1 1 1 60
2 1 2 62
3 1 1 65
4 1 1 65
5 1 2 76
6 2 1 59
7 2 2 73
8 2 1 73
9 2 2 79
10 3 1 69
11 3 2 71
12 3 2 71
13 3 1 63
14 4 2 72
15 4 2 72
16 4 1 80
17 4 1 77
18 4 2 77
;
run;
```

Using Proc Means we obtain a data set STAT with descriptive statistics for variable PULSE by SEX and treatment group (TRT).

```
proc means data=one nway;
  class sex trt;
  var pulse;
  output out=stat n=n mean=mean std=std min=min max=max;
run;
proc print;run;
```

OBS	SEX	TRT	_TYPE_	_FREQ_	N	MEAN	STD	MIN	MAX
1	1	1	3	3	3	63.3333	2.88675	60	65
2	1	2	3	2	2	66.0000	9.89949	59	73
3	1	3	3	2	2	66.0000	4.24264	63	69
4	1	4	3	2	2	78.5000	2.12132	77	80
5	2	1	3	2	2	69.0000	9.89949	62	76
6	2	2	3	2	2	76.0000	4.24264	73	79
7	2	3	3	2	2	71.0000	0.00000	71	71
8	2	4	3	3	3	73.6667	2.88675	72	77

Let say we need to transpose the results by SEX and across TRT. Whether we will do it using Proc Transpose without any ID statement or with TRT as ID variable

```
proc transpose data=stat out=tstat prefix=treat;
  var n mean std min max;
  by sex;
  id trt;
run;

proc print;run;
```

we will have the same result:

OBS	SEX	_NAME_	TREAT1	TREAT2	TREAT3	TREAT4
1	1	N	3.0000	2.0000	2.0000	2.0000
2	1	MEAN	63.3333	66.0000	66.0000	78.5000
3	1	STD	2.8868	9.8995	4.2426	2.1213
4	1	MIN	60.0000	59.0000	63.0000	77.0000
5	1	MAX	65.0000	73.0000	69.0000	80.0000
6	2	N	2.0000	2.0000	2.0000	3.0000
7	2	MEAN	69.0000	76.0000	71.0000	73.6667
8	2	STD	9.8995	4.2426	0.0000	2.8868
9	2	MIN	62.0000	73.0000	71.0000	72.0000
10	2	MAX	76.0000	79.0000	71.0000	77.0000

So, SAS automatically chooses TRT as ID variable just because there is no other choice. Does it mean that ID statement in this case is excessive? Actually, it does not. Imagine

that the original data are slightly different, in particular, all patients in treatment group 3 have sex=1.

```
data two ;
  input pid trt sex pulse;
  trtsex=compress(trt)||compress(sex);
  cards;
1 1 1 60
2 1 2 62
3 1 1 65
4 1 1 65
5 1 2 76
6 2 1 59
7 2 2 73
8 2 1 73
9 2 2 79
10 3 1 69
11 3 1 71
12 3 1 71
13 3 1 63
14 4 2 72
15 4 2 72
16 4 1 80
17 4 1 77
18 4 2 77
;
run;

proc means data=two nway;
  class sex trt;
  var pulse;
  output out=stat2 n=n mean=mean std =std min=min max=max;
run;
proc print;run;
```

OBS	SEX	TRT	_TYPE_	_FREQ_	N	MEAN	STD	MIN	MAX
1	1	1	3	3	3	63.3333	2.88675	60	65
2	1	2	3	2	2	66.0000	9.89949	59	73
3	1	3	3	4	4	68.5000	3.78594	63	71
4	1	4	3	2	2	78.5000	2.12132	77	80
5	2	1	3	2	2	69.0000	9.89949	62	76
6	2	2	3	2	2	76.0000	4.24264	73	79
7	2	4	3	3	3	73.6667	2.88675	72	77

As expected we do not have a row for sex=1 and trt=3 in the data set STAT2.

After transposing data set STAT2 using Proc Transpose with TRT as ID variable

```
proc transpose data=stat out=tstat2 prefix=treat;
```

```

var n mean std min max;
by sex;
id trt;
run;

```

```
proc print;run;
```

we obtain also reasonable result: the statistics for trt=3 and sex=1 are missing:

OBS	SEX	_NAME_	TREAT1	TREAT2	TREAT3	TREAT4
1	1	N	3.0000	2.0000	4.0000	2.0000
2	1	MEAN	63.3333	66.0000	68.5000	78.5000
3	1	STD	2.8868	9.8995	3.7859	2.1213
4	1	MIN	60.0000	59.0000	63.0000	77.0000
5	1	MAX	65.0000	73.0000	71.0000	80.0000
6	2	N	2.0000	2.0000	.	3.0000
7	2	MEAN	69.0000	76.0000	.	73.6667
8	2	STD	9.8995	4.2426	.	2.8868
9	2	MIN	62.0000	73.0000	.	72.0000
10	2	MAX	76.0000	79.0000	.	77.0000

However, if we would not specify TRT as an ID variable for transposing STAT2

```

proc transpose data=stat out=tstat2 prefix=treat;
var n mean std min max;
by sex;
run;

```

```
proc print;run;
```

the result is following:

OBS	SEX	_NAME_	TREAT1	TREAT2	TREAT3	TREAT4
1	1	N	3.0000	2.0000	4.0000	2.0000
2	1	MEAN	63.3333	66.0000	68.5000	78.5000
3	1	STD	2.8868	9.8995	3.7859	2.1213
4	1	MIN	60.0000	59.0000	63.0000	77.0000
5	1	MAX	65.0000	73.0000	71.0000	80.0000
6	2	N	2.0000	2.0000	3.0000	.
7	2	MEAN	69.0000	76.0000	73.6667	.
8	2	STD	9.8995	4.2426	2.8868	.
9	2	MIN	62.0000	73.0000	72.0000	.
10	2	MAX	76.0000	79.0000	77.0000	.

Instead of statistics in column TREAT3 to be missing, there are missing values in column TREAT4.

Generally speaking Proc Transpose gives us a wrong result whenever we skip the ID statement having “shortage” of values of across variable for one or more value of the BY groups compare to total number of values of across variable. This is the case for the data set STAT2 – we have totally four values of across variable TRT : 1,2,3, and 4, but the BY group with sex=2 has only three values:1,2, and 4.

2. Transposing across more than one variable.

Let's say we need to transpose the data set STAT (see above) across both SEX and TRT. The common approach to this task is following:

```
proc transpose data=stat out=tstat1 prefix=treat1;
  where sex=1;
  var n mean std min max;
  id trt;
run;
```

```
proc print;run;
OBS _NAME_ TREAT11 TREAT12 TREAT13 TREAT14

1 N 3.0000 2.0000 2.0000 2.0000
2 MEAN 63.3333 66.0000 66.0000 78.5000
3 STD 2.8868 9.8995 4.2426 2.1213
4 MIN 60.0000 59.0000 63.0000 77.0000
5 MAX 65.0000 73.0000 69.0000 80.0000
```

```
proc transpose data=stat out=tstat2 prefix=treat2;
  where sex=2;
  var n mean std min max;
  id trt;
run;
```

```
proc print;run;
OBS _NAME_ TREAT21 TREAT22 TREAT23 TREAT24

1 N 2.0000 2.0000 2 3.0000
2 MEAN 69.0000 76.0000 71 73.6667
3 STD 9.8995 4.2426 0 2.8868
4 MIN 62.0000 73.0000 71 72.0000
5 MAX 76.0000 79.0000 71 77.0000
```

```
data final;
  merge tstat1 tstat2;
run;
proc print;run;
```

```
OBS _NAME_ TREAT11 TREAT12 TREAT13 TREAT14 TREAT21 TREAT22 TREAT23 TREAT24

1 N 3.0000 2.0000 2.0000 2.0000 2.0000 2.0000 2 3.0000
2 MEAN 63.3333 66.0000 66.0000 78.5000 69.0000 76.0000 71 73.6667
3 STD 2.8868 9.8995 4.2426 2.1213 9.8995 4.2426 0 2.8868
4 MIN 60.0000 59.0000 63.0000 77.0000 62.0000 73.0000 71 72.0000
5 MAX 65.0000 73.0000 69.0000 80.0000 76.0000 79.0000 71 77.0000
```

Imagine we have more values for variable sex 😊! The code will be longer or we will have to write a macro. However there is a shortcut to reduce the number of code lines to minimum, no matter how many across variables we have and how many values each of them has. We just create an ID variable SEXTRT by concatenating SEX and TRT.

```

data stat;
  set stat;
  sextrt=compress(sex)||compress(trt);
run;

proc transpose data=stat prefix=SEXTRT out=tran;
  var n mean std min max;
  id sextrt;
run;

```

OBS	_NAME_	SEXTRT11	SEXTRT12	SEXTRT13	SEXTRT14	SEXTRT21	SEXTRT22	SEXTRT23	SEXTRT24
1	N	3.0000	2.0000	2.0000	2.0000	2.0000	2.0000	2	3.0000
2	MEAN	63.3333	66.0000	66.0000	78.5000	69.0000	76.0000	71	73.6667
3	STD	2.8868	9.8995	4.2426	2.1213	9.8995	4.2426	0	2.8868
4	MIN	60.0000	59.0000	63.0000	77.0000	62.0000	73.0000	71	72.0000
5	MAX	65.0000	73.0000	69.0000	80.0000	76.0000	79.0000	71	77.0000