

Standardizing Output: PROC TRANSPOSE to the Rescue

Judy Loren, Independent Consultant

Let's suppose you have to write some code that will produce a particular summary report on a regular basis, using different data each time. Let's further suppose that one of your class variables may not be fully represented, that is the data may be sparse. If you simply use PROC SUMMARY or PROC MEANS to produce the report, the combination of values that isn't represented in the data source just doesn't appear in the final report.

For example, if your class variables are SEX, AGE and MUSIC preference, and your report is run monthly, there might not be any Females in their 30's who like Classical music in the May data.

Sometimes the users are taking the reports and fitting them into a predefined format (let's say an Excel spreadsheet) and they would really like the rows to be exactly the same every month (in other words, all combinations always represented even if the data values are missing). What do you do then?

It might first occur to you to process the results of the PROC SUMMARY and put out your own blank rows. But this requires that you hard code the potential values, making maintenance down the road more difficult.

Here's another possible solution: Use the PROC TRANSPOSE procedure twice. Note that this only solves the problem when you can count on each possible value of each class variable occurring at least once in the dataset. It will produce rows for all combinations of represented values. In the above example, where you had no Females in their 30's who like Classical music, it would solve the problem as long as at least one observation had a value of F for sex, and at least one observation had a value of 30's for age and at least one observation had a value of Classical for music preference. With truly sparse data, this method will not work.

The purpose of PROC TRANSPOSE is to reshape your dataset, creating observations out

of variables and variables out of observations. The best way to understand this may be to look at an example.

Figure 1 shows the raw data, which may have come from a survey.

In this case we're going to assume that the variables SEX and AGE are fully represented, that is there is at least one observation for each possible combination of SEX and AGE. The variable we need to square up is MUSIC.

First we put the raw data through a PROC SUMMARY:

```
proc summary nway;
class sex age music;
var party dances;
output out=summed
mean(dances)=avgdances
sum(party)=atparty
n=atconf;
```

The first 10 obs of the results are printed in Figure 2. You'll notice that the values of MUSIC are different for females in their 20's and females in their 30's.

Next we use the TRANSPOSE procedure:

```
proc transpose data=summed out=trans;
id music;
by sex age;
```

The results are shown in Figure 3 (which you may note is produced with a PROC REPORT NAMED WRAP instead of a PROC PRINT because it was too wide for this format). PROC TRANSPOSE creates an observation for each numeric variable within the BY variables SEX and AGE. The numeric variables in the incoming dataset become separate observations, with the name of the variable stored in the variable _NAME_. By specifying ID MUSIC, we asked TRANSPOSE to create separate columns for each value of the MUSIC variable in the incoming dataset, and use the values to

name the variables in the new dataset. Note that it automatically makes the values into legal variables names by replacing the & with an underscore.

The values that were stored in the observation identified by sex="f" and age="2" and music="country" are now stored in 5 observations, in the column marked "country". The average number of dances danced by females in their 20's who prefer country music, 2.4, is still stored in the intersection of "avgdances" and "country", it's just that now "avgdances" is an observation instead of a variables and "country" is now a variable instead of an observation.

TRANSPPOSE creates variables for each value of MUSIC that it finds. The first value it finds is "country", followed by "jazz" and "r&r". This is all it finds for the class variables sex="f" and age="2". When it processes sex="f" and age="3" it encounters music="altern" for the first time. Therefore, all the obs (all the variables from the incoming dataset) with sex="f" and age="2" get missing values for the variable named "altern".

The final step is to convert the dataset back to the original format, with the addition of the missing value rows to represent potential values of sex, age and music that did not actually occur in the survey data. This is accomplished with a second PROC TRANSPPOSE.

```
proc transpose data=trans out=back
name=music;
where not (_NAME_ in
(" _TYPE_", "_FREQ_"));
by sex age;
```

The output is shown in Figure 4. Comparing this with the output of the PROC SUMMARY, Figure 2, will show you where the new rows are.

Figure 1. raw data

OBS	SEX	AGE	MUSIC	PARTY	DANCES
1	m	2	altern	1	8
2	m	5	r&r	0	0
3	f	2	country	0	0
4	f	3	r&r	0	0
5	m	4	jazz	1	0
6	m	2	class	0	0

Figure 2. results of summary

SEX	AGE	MUSIC	AVGDANCES	ATPARTY	ATCONF
f	2	country	2.4	1	7
f	2	jazz	3.0	1	5
f	2	r&r	27.5	2	2
f	3	altern	30.5	2	2
f	3	country	17.0	6	6
f	3	r&b	19.0	2	2
f	3	r&r	33.7	14	15
f	4	country	0.0	0	1
f	4	jazz	25.0	2	2
f	4	r&r	50.0	1	1

Figure 3. results of first transpose

```
SEX=f AGE=2 _NAME_=TYPE COUNTRY=7
JAZZ=7 R_R=7 ALTERN=. R_B=. CLASS=.
```

```
SEX=f AGE=2 _NAME_=FREQ COUNTRY=7
JAZZ=5 R_R=2 ALTERN=. R_B=. CLASS=.
```

```
SEX=f AGE=2 _NAME_=AVGDANCES COUNTRY=2.4
JAZZ=3 R_R=27.5 ALTERN=. R_B=. CLASS=.
```

```
SEX=f AGE=2 _NAME_=ATPARTY COUNTRY=1
JAZZ=1 R_R=2 ALTERN=. R_B=. CLASS=.
```

```
SEX=f AGE=2 _NAME_=ATCONF COUNTRY=7
JAZZ=5 R_R=2 ALTERN=. R_B=. CLASS=.
```

```
SEX=f AGE=3 _NAME_=TYPE COUNTRY=7
JAZZ=. R_R=7 ALTERN=7 R_B=7 CLASS=.
```

```
SEX=f AGE=3 _NAME_=FREQ COUNTRY=6
JAZZ=. R_R=15 ALTERN=2 R_B=2 CLASS=.
```

Figure 4. results of second transpose

SEX	AGE	MUSIC	ATCONF	ATPARTY	AVGDANCES
f	2	COUNTRY	7	1	2.4
f	2	JAZZ	5	1	3.0
f	2	R_R	2	2	27.5
f	2	ALTERN	.	.	.
f	2	R_B	.	.	.
f	2	CLASS	.	.	.
f	3	COUNTRY	6	6	17.0
f	3	JAZZ	.	.	.
f	3	R_R	15	14	33.7
f	3	ALTERN	.	2	30.5

Trademarks

SAS is registered trademark of SAS Institute Inc., Cary, NC.