ABSTRACT
A quick and easy way of generating entries for frequency table is presented. It involves the construction of indicator variables that correspond to those properties in the table in data steps and a call of PROC UNIVARIATE from SAS® on these indicator variables to generate a data set that contains all of the key quantities in the table. It then uses these key quantities in a data step to create all entries for the table. Finally the data set is presented in the way requested via PROC REPORT or DATA _NULL_.

INTRODUCTION
Statistical programmers are often asked to generate tables that contain many columns with different properties. For example, a table may have a column of the number of subjects with PQ value > Xo msec for each treatment at each time point, a column of the number of readings with PQ value > Xo msec over all available PQ readings for each treatment at each time point, and a percentage column. It may contain another three columns with similar structure for another measurement. And so on. One way statistical programmers generate this kind of table is to use PROC FREQ to generate counts needed for each column and then merge these columns together. The following code is often used to obtain the counts for columns of subjects of PQ value > Xo and events of PQ value > Xo over the total available PQ readings:

```
proc sort data=ecg(where=(pq>x0)) nodupkey out=ecg0;
by group timept subject;
run;

proc freq data=ecg0 noprint;
by group timept;
tables subject / out = out1 (keep = group timept count);
run;

proc freq data=ecg0 noprint;
by group timept;
tables subject / out = out2 (keep = group timept count);
where pq>x0;
run;

proc freq data=ecg0 noprint;
by group timept;
tables subject / out = out3 (keep = group timept count);
where pq ne .;
run;
```

A similar process is then used to generate data sets out4, out5, and out6 that contain counts needed for the next three columns. Finally a data step is used to merge data sets out1 through out6 together and to generate percentage columns and the x over y columns:

```
data prtdata;
merge out1 (rename=(...)) ... out6 (rename=(...));
by group timept;
length c4 c5 c7 c8 $8;
c4 = put(count2,2.)||'/'||put(count3,3.);
...
run;
```

Although a macro may be introduced to shorten the code, the process is still time consuming in terms of productivity. Another way of generating this kind of table is to use an idea from basic statistics of defining a number to represent an event that has occurred and then perform an operation on these defined numbers rather than on the events. The method I propose here is to first use a data step to define indicator variables that correspond to those properties in the table. Then use PROC UNIVARIATE on these indicator variables to generate a data set that contains all of the
key quantities in the table. And then use these key quantities in a data step to create all entries for the table. Finally the data set can be presented in the way requested via PROC REPORT or DATA _NULL_.

**HOW TO DO**

Let’s use the following table as an example. This table has a total of eight columns: treatment information in column 1, time point in column 2, number of subjects with PQ value > Xo msec for each treatment at each time point in column 3, number of readings with PQ value > Xo msec over all available PQ readings for each treatment at each time point for column 4, percentage of readings with PQ value > Xo msec among all available PQ readings for each treatment at each time point for column 5, number of subjects with QRS value > Yo msec for each treatment at each time point in column 6, number of readings with QRS value > Yo msec over all available QRS readings for each treatment at each time point for column 7, and percentage of readings with QRS value > Yo msec among all available QRS readings for each treatment at each time point for column 8. Treatment information in column 1 and time point in column 2 is usually from source data. The percentage in column 5 can be derived from the numbers in column 4 and column 8 can be obtained from column 7. So the key to this table is to generate numbers in columns 3, 4, 6, and 7.

```
Frequency of subjects and events with PQ and QRS abnormalities at pre and post dose

(1)       (2)        (3)   (4)     (5)      (6)   (7)     (8)
PQ Value > Xo msec    QRS Value > Yo msec
---------------------  -----------------------
Subj  Events  Percent  Subj  Events  Percent
Treatment  Timepoint   [a]   n/N    Events    [a]   n/N    Events
--------------------------------------------------------------------
Placebo    Pre-dose     xx  xx/xxx  (xx.x%)    xx  xx/xxx  (xx.x%)
Postdose   xx  xx/xxx  (xx.x%)    xx  xx/xxx  (xx.x%)
Trt1       Pre-dose     xx  xx/xxx  (xx.x%)    xx  xx/xxx  (xx.x%)
Postdose   xx  xx/xxx  (xx.x%)    xx  xx/xxx  (xx.x%)
...         ...
N = # of available readings for the period, n = # of abnormal readings
for the period. [a] Number of subjects with abnormal readings.
```

We first use a data step to define indicator variables. Let X1 be the indicator variable for the event that a subject has a reading with PQ value > Xo msec. That is, X1 is defined to be 1 the first time a subject has a value of PQ >Xo and for the rest of the records for the same subject X1 is defined to be missing. Let X2 be the indicator variable for the event that a reading of PQ is greater than Xo msec. That is, X2 is defined to be 1 whenever a record has a value of PQ >Xo msec and is missing for the other records. Let X3 be the indicator variable for the event of available PQ readings. That is, X3 has a value of 1 when a PQ reading is not missing and a missing value if a PQ reading is missing. Indicator variable Y1 for the event that a subject has a reading with QRS value > Yo msec, indicator variable Y2 for the event that a reading of QRS is greater than Yo msec, and indicator variable Y3 for the event of a non-missing QRS reading can be defined in a similar way.

The following code generates indicator variables X2, X3, Y2, and Y3:

```sas
data ecg;
set ecg;
if pq ne . then x3 = 1;
if qrs ne . then y3 = 1;
if pq > Xo then x2 = 1;
if qrs > Yo then y2 = 1;
keep group subject timept x: y:;
r;
```

The code for indicator variable X1:

```sas
proc sort data=ecg;
by group subject timept x2;
r;

data ecg;
set ecg;
```
by group subject timept X2;
if X2 and first.X2 then X1 = 1;
run;

Indicator variable Y1 can be obtained by replacing X2 with Y2 and X1 with Y1 in the code for X1:

    proc sort data=ecg;
    by group subject timept Y2;
    run;

data ecg;
set ecg;
by group subject timept Y2;
if Y2 and first.Y2 then Y1 = 1;
run;

We then use PROC UNIVARIATE on indicator variables X1, X2, X3, Y1, Y2, and Y3 to generate a data set that contains numbers for columns 3, 4, 6, 7:

    proc univariate data = ecg noprint;
    by group timept;
    var X1 X2 X3 Y1 Y2 Y3;
    output out = prtdata n = nX1 nX2 nX3 nY1 nY2 nY3;
    run;

Data set PRTDATA is manipulated further to create character variables for columns 4, 5, 7, and 8 before using PROC REPORT or DATA _NULL_ to finally generate the table:

    data prtdata;
    set prtdata;
    length c4 c5 c7 c8 $8;
    c4 = put(count2,2.)||’/’||put(count3,3.);
    ...
    run;

CONCLUSION
The use of indicator variables can simplify the task of generating a table. There is no need to obtain columns separately according to different column properties and then merge these columns together. No transposing is needed and no macros need to be called. The use of indicator variables also provides ease and flexibility when it comes to the time for changes since we can easily change the definition of indicator variables in data steps.
Although a frequency table is used as an example here, the idea can be easily extended to generate other kinds of tables. The definition of indicator variables could also be modified into the form seen most often by replacing missing values with 0 for the indicator variables and replacing n in PROC UNIVARIATE with sum.

ACKNOWLEDGMENTS
The author would like to thank Jim Edgington for reviewing the manuscript and providing valuable comments.

CONTACT INFORMATION
Your comments and questions are valued and encouraged. Contact the author at:

    Jinshi Zhou
    Quintiles, Inc.
    P.O. Box 9708
    Kansas City, MO 64134-0708
    Work Phone: (816) 767-6627
    Fax: (816) 767-7331
    Email: joe.zhou@quitiles.com

SAS and all other SAS Institute Inc. product or service names are registered trademarks or trademarks of SAS Institute Inc. in the USA and other countries. ® indicates USA registration.

Other brand and product names are trademarks of their respective companies.