

Ad Hoc Report for Users

Michael A. Mace, SPS Software Services Inc., Canton, Ohio

ABSTRACT

The two programs or techniques described here will allow clients to produce their own customized (ad hoc) reports. This provides the client with independence and flexibility and frees the programmer from 'report boy' syndrome. One technique uses a series of %PUT and %INPUT statements to step the client through the process; the second simply uses a brief description to guide the client in entering an SQL statement.

INTRODUCTION

The techniques I present here resolve two reporting needs of clients. First, when a basic report simply needs to be produced based on new criteria, such as beginning and ending dates or a different location, code, et cetera. Second, when the client wants to explore the data or answer a simple question, like how many widgets are what color.

No matter how careful and far-sighted you are in developing an application in cooperation with a client to meet the reporting needs, invariably you will receive a phone call or e-mail asking for one more special report. Next month or next year you will be asked for yet another, and each time you have to refresh your memory regarding the purpose of the application and the variables used. If you have an exceptionally inquisitive, helpless, or demanding client, you will quickly become a 'report boy.' No longer! With the use of either or both of the techniques presented here, you can free yourself and your clients to do things you are both being paid to do.

I should also mention that when I work with my clients in creating an application, I constantly emphasize that they are the owners and users of the application and need to know the basics of what things are, how things work, and what can be reasonably expected. Just as they educate me about their needs, I try to educate them about the tools of the SAS® system. This mutual understanding helps not only in the initial development but also during any trouble-shooting or future modifications.

As you will see in this paper, I assume that the client knows the structure of the application and knows what he/she wants to get out of it, and will think about what is being requested and typed. Go ahead and laugh; I admit there are times I need the patience of Job and the tact of Solomon.

%PUT and %INPUT

The first technique steps the user through the criteria-gathering process. %PUT is used here to display the prompt; %INPUT is used to capture the response in a macro variable. Here is a short example:

```
%PUT Please enter the variable for which you want a
frequency table ;
%INPUT freqvar ;
PROC FREQ DATA= fileref.dsname ;
    TABLES &freqvar / options ;
RUN ;
```

When working with dates, the prompt is very specific:

```
%PUT Please enter the BEGINNING date as
DDMONYYYYY ;
%PUT i.e. 06APR2007 ;
%INPUT begndate ;
```

The response could be used thusly:

```
SET filerefef.dsn (WHERE=
                    ("&begndate"D<=datevar<="&enddate"D)
```

Remember that macro variable values are text. Therefore I request the date values in DATE9. format so that I do not need to make any data conversions. Also notice that I have requested a four-character year to make sure everyone remembers that the turn of the century is upon us.

One assumption here is that the client will enter exactly what is requested as requested; if the criteria are not entered correctly, an empty report will be produced. I do not do any 'error' checking when it comes to the client's inputs; that could be incorporated if deemed necessary.

You can also prompt for multiple values. There are several ways to do this and reasons why you will use one rather than another, in particular, how the response is going to be used. Here is an example within a macro.

```
%MACRO macronam ;
.
%PUT Please enter the first three characters of desired
homebase(s) ;
%PUT in single quotes, - separate multiple homebases
with commas ;
%PUT ex.: '041','042','037' ;
%INPUT hb ;
.
.
```

```

DATA dsname ;
.
.
  %IF %LENGTH(&hb) GT 0
  %THEN %DO ;
      IF SUBSTR(homebase,1,3) IN(&hb) ;
  %END ;
RUN ;
.
.
%MEND macronam ;
%macronam ;

```

Another accessory tool you need for a truly ad hoc report is the automatic macro variable, &SYSBUFFR, which accepts any response to a %INPUT prompt that is not matched with a specific variable.

```

%PUT Please enter whatever you want ;
%INPUT ;

```

The value of &SYSBUFFR would be whatever the user typed. This allows a great deal of flexibility, independence, and yes, uncertainty; however it is crucial for our purposes.

Please refer to *SAS Guide to Macro Processing, Version 6, Second Edition* for a thorough discussion about macros, macro variables, %PUT, %INPUT, and &SYSBUFFR.

Remember what I said earlier about the client knowing the application? Since the client is going to produce these ad hoc reports, it is absolutely necessary for them to know the variables. Therefore, be sure to provide a printout of PROC CONTENTS to each client and educate your clients using SAS/FSP products, FSEDIT or FSBROWSE, that by placing the cursor on a field and pressing F1, the name, type, format, and informat of the underlying variable will be displayed in the message line.

Finally, here is the complete ad hoc report program using %PUT, %INPUT, and &SYSBUFFR:

```

OPTIONS DATE NUMBER ;
LIBNAME realstat 'sccp.realstat.perm.sasdata' SERVER=
SHARE6P ;

FOOTNOTE "<ADHOCRE>" ;
TITLE1 "Company Name: REAL ESTATE" ;

%MACRO adhoc ;

CLEAR ;
%LET null = ;

%PUT Please enter which data set you want to report
against ;
%PUT AGREEMNT CONDEMN EXPENSE INSCERT
MUNIADRS PROPERTY ;
%PUT RAILROAD RENTAL SALE TAX TAXJCPL
TRANSROW TXCLLCTR VDRS ;
%INPUT dsn ;

```

```

%PUT %STR( ) ;
%PUT Please list ALL of the variables you want on the
report ;
%PUT in the desired order: ;
%PUT INCLUDING any variables by which you may want
to sort ;
%PUT THEN press <ENTER> ;
%INPUT ;
%LET var = &SYSBUFFR ;

```

```

%PUT %STR( ) ;
%PUT If you want to specify an additional subsetting
criterion, ;
%PUT please enter it as: variable comparison_operator
value ;
%PUT OR: FUNCTION(variable,parameters)
comparison_operator value ;
%INPUT ;
%LET criteria = &SYSBUFFR ;
TITLE4 "%UPCASE(&criteria)" ;

```

```

%MACRO criteria ;
%IF %LENGTH(&criteria) = 0
%THEN ;
%ELSE %STR(IF &criteria) ;
%MEND criteria ;
%PUT %STR( ) ;
%PUT Please enter the SORT variables in the desired
order: ;
%PUT NB: The sort variables MUST have been included
in the variable list;
%INPUT ;
%LET sortvar = &SYSBUFFR ;

```

```

%MACRO sortvar ;
%IF %LENGTH(&sortvar) = 0
%THEN ;
%ELSE %DO ;

```

```

PROC SORT DATA= &dsn
NODUPPLICATES ;
BY &sortvar ;
RUN ;

```

```

%END ;
%MEND sortvar ;
%PUT %STR( ) ;
%PUT Finally, if you want an additional TITLE, please
enter it: ;
%INPUT ;
%LET title3 = %UPCASE(&SYSBUFFR) ;
TITLE3 "&title3" ;

```

```

DATA dsn ;
SET realstat.&dsn (KEEP= &var ) ;

%criteria ;

RUN ;

```

```
%sortvar ;
```

```
PROC PRINT DATA= dsn NOOBS UNIFORM ;  
  VAR &var ;  
  BY &sortvar ;  
RUN ;
```

```
%MEND adhoc ;  
%adhoc ;
```

```
LIBNAME realstat CLEAR ;
```

PROC SQL

The second technique for producing ad hoc reports was literally thrown together in response to a client's comment that he knew how to write structured queries in other software languages.

%PUT is used to display all of the instructions at one time. %INPUT and &SYSBUFFR are used to accept the complete response. Here again, the client must be familiar with the variables, their names, types, et cetera, and must follow the instructions exactly or nothing will be produced.

```
LIBNAME rtk 'chem.gencortk.perm.sasdata' SERVER=  
share6p ;  
LIBNAME library 'chem.formats.perm.sasdata' SERVER=  
share6p ;
```

```
OPTIONS DATE NUMBER NOCAPS ;  
FOOTNOTE "<ADHOCSQL>" ;  
TITLE1 "Company Name: RIGHT-TO-KNOW" ;
```

```
CLEAR ;  
%PUT %STR( ) ;  
%PUT Enter your SQL statements below (RTK is the  
first-level dsn qualifier, i.e. fileref) ... ;  
%PUT %STR( ) ;  
%PUT SELECT variable, variable(s), ... <SPACE> ;  
%PUT FROM fileref.dataset alias, fileref.dataset alias ...  
<SPACE> ;  
%PUT WHERE condition(s) ... <SPACE> ;  
%PUT ORDER BY variable, variable(s), ... ;  
%PUT %STR( ) ;  
%PUT THEN, AND ONLY THEN, press <ENTER> ;  
%INPUT ;  
%LET sqlstmt = &SYSBUFFR ;
```

```
PROC SQL ;  
&sqlstmt ;  
;  
QUIT ;  
RUN ;
```

```
LIBNAME library CLEAR ;  
LIBNAME rtk CLEAR ;
```

CONCLUSION

One limitation to these techniques is that the output will be formatted by the procedure used, which could result in wrapping of output depending on how many variables are requested. Another limitation is that it is extremely unwieldy, though not impossible, to specify criteria for an array of variables, such as var1-var30. If the client really wants a report of these variables, there is a bit more programming that needs to be done beforehand.

I trust that these techniques will become valuable tools for your immediate and future use to enhance the applications you write to better serve your clients' reporting needs and free yourself from the 'report-boy' syndrome.

For more on user-customized report generation, please see my paper, "Using %WINDOW to Obtain User Criteria for Reports", also in these NESUG '97 proceedings.

SAS and SAS/FSP are registered trademarks or trademarks of SAS Institute Inc. in the USA and other countries. ® Indicates USA registration.

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

REFERENCES

SAS Institute Inc. (1990), *SAS Guide to Macro Processing, Version 6, Second Edition*, Cary, NC: SAS Institute Inc.

Michael A. Mace
1152 E 344 ST
Eastlake, OH 44095-2942
440-953-8924 (H)
330-796-3981 (W)
e-mail: michael.a.mace@worldnet.att.net