Paper 251

Submitting a Batch SAS® Job within the Display Manager Mode under UNIX®

Jingren Shi, Syntel Inc., Troy, Michigan
Shiling Zhang, Wayne State University, Detroit, Michigan

Abstract

Under the UNIX® operation system, there are two major approaches to submit a SAS® job. One is the batch mode; the other is the interactive SAS Display Manager Mode. The SAS Display Manager has a built-in editor that provides a rich set of editing commands and DM commands. A programmer can easily edit his programs, check log files, and view listing files under the SAS Display Manager mode. However, the batch mode allows a programmer to submit several SAS jobs at the same time without tying up one other. We present an approach that allows a programmer to submit a batch SAS job within the Display Manager Mode. Thereafter a programmer can fully take advantages of both modes. Unlike previous papers [4][5], our approach uses the information stored in the DICTIONARY.EXTFILES. Submitting a batch SAS job is similar to submitting a foreground SAS job and one function key or icon setup is used under our approach.

Introduction

SAS Display manager is a windowing facility that manages a SAS session. It has a built-in editor that provides a rich set of editing commands and DM commands. Under this mode, a programmer can easily access both SAS files and external files. For example, a programmer can edit his programs, check log files and listing files, and view members in a SAS data library or a catalog.

The most common reason to submit a SAS job in the batch mode is to allow a programmer to do something else interactively without waiting for the job to complete. The other great feature of the batch mode is that it allows a programmer to submit several SAS jobs simultaneously without tying up each other.

If we can define a method that allows a programmer to submit a batch SAS job within the Display Manager Mode, he will be able to exploit both advantages of the Display Manager Mode and the batch mode. The answer is positive. The SAS 6.07 release provides several system tables. One of them is called the DICTIONARY.EXTFILES. It lists external files that are currently associated with a SAS session.

[1]The following program uses the DESCRIBE TABLE statement in SQL to show you how DICTIONARY.EXTFILES is defined:

```
proc sql;
   describe table DICTIONARY.EXTFILES;
quit;
```

The DESCRIBE TABLE results are written to the SAS log, as shown below. The columns listed in this definition give information about all the external files referenced by your current SAS session.

```
(
FILEREF char(8) label='Fileref',
XPATH char(80) label='Path Name',
XENGINE char(8) label='Engine Name'
);
```

One may use the column names listed in the table definition in SELECT, WHERE, or other clauses when one wants to retrieve information from DICTIONARY.EXTFILES.

Information about a file name and its path is updated dynamically in DICTIONARY.EXTFILES table when SAVE, SAVE AS, and OPEN commands are issued.

Getting a File Name from DICTIONARY.EXTFILES

Under the UNIX platform, the syntax for a batch SAS job within Display Manager Mode is,

```
1)    call system("sas -sysin &_execpgm &");
2)     x "sas -sysin &_execpgm &";
3)     rc=system("sas -sysin &_execpgm &");
4)     %sysexec  sas -sysin &_execpgm &;
```

Obviously, in order to carry out any one of the above commands the macro variable '&_execpgm' should contain a full path and a file name of source codes in program window before it is submitted as a batch job. One can capture this information from the DICTIONARY.EXTFILES just after one issues a SAVE or a SAVE AS command.

The following SQL codes will catch this information dynamically and assign it to the macro variable _execpgm.

```
%macro dobatch;
%local _execpgm;
 /* clear pgm window */
*dm 'clear pgm';

/* getting file name information */
proc sql noprint;
  reset inobs=1;
   select xpath into: _execpgm
     from dictionary.extfiles
;
quit;

/* executing a batch job */
data _null_;
    call system ("sas -sysin &_execpgm &");
run;

%put &_execpgm is submitted as a batch job.;

%mend;
```

Macro Program Initialization

There are several ways to initialize the 'dobatch' macro. The simplest way is to add this macro into the default autoexec.sas so that it is available for every SAS session. The 'dobacth' macro can also be stored in an AUTOCALL library along with other macros. Any user can add the 'dobatch' macro into his own autoexec.sas and launch his SAS session with the option

```
–autoexec /myhome/mysasinit/autoexec.sas
```

An autoexec.sas file may look like,

```
***** example of autoexec.sas;
***** other SAS statements;

/* setting up a function key */
/* %keydef f12 "submit '%dobatch'"; */

%macro dobatch;
%local _execpgm;
 /* clear pgm window */
*dm 'clear pgm';

/* getting file name information */
proc sql noprint;
  reset inobs=1;
   select xpath into: _execpgm
     from dictionary.extfiles
;
quit;

/* executing a batch job */
data _null_;
   call system ("sas -sysin &_execpgm &");
run;

%put &_execpgm is submitted as a batch job.;

%mend;

/* setting up a function key */
%keydef f12 submit '"%dobatch"';
```

Function key or icon settings

Only one function key needs defined in our approach. There are several ways to set up a function key. One is to add (%keydef f12 “submit ‘%dobatch’”;) statement before the 'dobatch' macro in autoexec.sas. Note that putting (%keydef f12 “submit ‘%dobatch’”;) statement before the 'dobatch' macro is to avoid resolving ‘%dobatch’. However, SAS will issue a warning message "WARNING: Apparent invocation of macro DOBATCH not resolved".

If one does not like this message, he can add (%keydef f12 submit ‘"%dobatch"';) statement anywhere in autoexec.sas. The function key f12 will be assigned as "submit '%dobatch' ". Note there are triple single quotes around %dobatch. In this way the function will set up automatically in every SAS session.

The other is manually setup. A user may assign (submit ‘%dobatch’) to any function he likes. One could set up an icon in a similar way.

Conclusion Remarks

The approach is very simple in terms of function key settings and generic because it uses information stored in the DICTIONARY.EXTFILES. Submitting a batch SAS job is similar to submitting a foreground SAS job. The limitation of this approach is that one has to save a program in the program window first so that the 'dobatch' macro can capture the necessary information for a batch job. On the other hand, if there are a lot of users, one may modify the macro 'dobacth' so that it can have a function to collect some metrics information, such as user name, system time usage, redirection log files, etc.

References

[1] SAS® Institute Inc., SAS® Technical Report P-222: Changes and Enhancements To Base SAS® Software, Release 6.07, Cary, NC: SAS® Institute Inc., 1991

[2] SAS® Institute Inc., SAS® Macro Language: Reference, Version 6, Second Edition, Cary, NC: SAS® Institute Inc., 1997

[3] SAS® Institute Inc., SAS® Guide to Macro Processing, First Edition, Cary, NC: SAS® Institute Inc., 1990

[4] Blodgett, John. (1997), "Combining Display Manager and Batch Mode Under UNIX," Proceedings of the Twenty-Second Annual SAS® Users Group International Conference, 22, 64-47.

[5] Young, Jim. (1998), "User-Friendly Toolbox for Batch Processing within a UNIX® Interactive Display Manager Session," Proceedings of the Twenty-Third Annual SAS® Users Group International Conference, 23, 134-137.

SAS is a registered trademark or trademark of SAS Institute Inc. in the USA and other countries. ® indicates USA registration.

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

Authors:
Jingren Shi
Syntel Inc.
jshi2@ford.com

Shiling Zhang
The Department of Mathematics
Wayne State University
shiling@math.wayne.edu