Paper 083-2009

Dynamically allocating exported datasets by the combination of pipes and ‘X’ statement
Xin Wei, Hoffmann-La Roche, Nutley, NJ

ABSTRACT
If you generate a thousand of tables in five minutes, you may also want to dump them into appropriate locations to show your clients that you are not only productive but also well-organized. In this paper we discuss the solution to dynamically export table into a destination whose name depends on data and you do not know in advance by the combination of system pipe option and ‘X’ statements in SAS data step.

INTRODUCTION
It is a frequent task for SAS programmers to write different result tables into the desired destinations in Windows system. Often times these destinations have to exist prior to the execution of exportation. It would be labor intensive and error-prone to manually create these subdirectories if many folders are needed or they have complex hierarchical structures. The problem is even compounded when the name of subdirectories to create for the exported table are data dependent. In the following examples, I will describe the solution to make the data exportation and creation of file folders a dynamic and coherent process by using pipe and ‘X’ statement.

CONCEPT
According to SAS support page, unnamed pipes “enable you to invoke a program outside of SAS and redirect the program’s input, output, and error messages to SAS. This capability enables you to capture data from a program external to SAS without creating an intermediate data file”. A ‘X’ statement runs an operating system command or a Windows application from within a SAS session.

The example I am using demonstrates writing study budget for different companies into separated subdirectories designated for a particular company. First of all, the names of existing subdirectories from a particular directory were written into a SAS dataset by system pipe command. Then company names from dataset to export are searched against the SAS dataset holding existing folder names. If the folder for the current company already exists, SAS goes ahead to export the study budget data into this folder. If not, SAS invokes MS-DOS command by ‘X’ statement to create a new folder with the company name and write the corresponding file in it. When the process is completed, all data are automatically saved in the desired locations.

PROCESS
STEP 1:
This step creates a test dataset that contains three columns: company name, study name and study budget. The goal is to output the studies and their budget into a company subdirectory.

```
data test;
  length company $20 study $20;  
  input company $ study $ budget;
  cards;
  ROCHE XXXinhibitor 10000
  ROCHE XXXantagonist 20000
  ROCHE XXXantibody 30000
  NOVARTIS XXXinhibitor 40000
  NOVARTIS XXXantagonist 50000
  NOVARTIS XXXantibody 60000
  PFIZER XXXinhibitor 70000
  PFIZER XXXantagonist 80000
  PFIZER XXXantibody 90000
; 
run;
```

STEP 2:
This step uses PROC SQL to parse all company names into a text string and then call a loop in SAS macro. Within each loop execution, a company name is parsed out of the company name list encoded by the macro variable &cmlst by %scan function. The study budget data will be exported for one company at a time. We will discuss what is going on within each loop in the next step.
**STEP 3:**

Step 3 covers the two main points of this report. Firstly a ‘pipe’ option is used in filename statement to read the folder name into variable ‘folder’ of SAS dataset ‘folder’. Once the desired subdirectory name/company names are compiled, it is compared to the company name that is parsed out of company name list from step 2. If the company folder is determined to exist, a flag variable (preset to 0 before the data step) is set to 1 by a call symput() function. Next the ‘X’ statement followed by a quoted MS-DOS command is conditionally executed to create the company subdirectory that does not exist. Here the ‘noxwait’ option is used to specify that the command processor automatically returns to the SAS session after the MS-DOS command is executed. Finally, a study budget table for a particular company is subset from master dataset through ‘where’ option and then exported to the company subdirectory that is guaranteed to be present at this moment.

```sas
proc sql noprint;
    select distinct company into: cmlst separated by ' ' from test;
quit;

%let i=1;
%do %while(%scan(&cmlst,&i,%str( )) ne);
    %let company=%scan(&cmlst,&i,%str( ));
    %let i=%eval(&i+1);
    XXXXXXXXXXX
%end;
```

**DISCUSSION**

You can actually control what information is read from system by using various pipe options. The followings are two examples of dataset that holds the name of subdirectories for three companies

**EXAMPLE 1:**

```sas
FileName MyDir Pipe 'dir "c:\test" /B';
data folder;
    infile MyDir lrecl=300 truncover;
    input @1 folder $100. @;
    format   folder $100.;
run;

%let flag=0;
data _null_;  
    set folder;
    if upcase(folder)=upcase("&company") then do;
        call symput('flag','1');
        stop;
    end;
run;
%if &flag=0 %then %do;
    options noxwait;
    data _null_; 
        x "mkdir c:\test\&company";
    run;
%end;
proc export data=test(where=(department="&company"))
    outfile="c:\test\&company\budget.xls" replace;
run;
```
By using /B option at the end of statement, you only read the name of current directories in column ‘folder’.

**EXAMPLE 2:**
```bash
FileName MyDir Pipe 'dir c:\test /AD /B /S';
```

By using /AD /B /S you read the full path of current directories into the SAS dataset ‘folder’.

In order to get the whole list of options and descriptions, run the following code in SAS session:

```bash
filename help pipe 'dir /?';
data null;
  infile help;
  input;
  list;
run;
```

Once that happens, you have the following help information display on log page:

Invoking x statement cause the MD-DOS windows being repeatedly turn on and off, which is slightly annoying. A better alternative in the framework of pipe can achieve the same without having DOS window pop up. Just replace the following code
```bash
data null;
ex "mkdir c:\test\&company";
run;
```
with the followings:
```bash
filename md pipe "mkdir c:\test\&company";
data null;
infile md;
run;
```

**CONCLUSION:**
This simple example gives you flavor about the utility of executing windows operations within a SAS session. The code can be easily tweaked to generate more complex Windows subdirectory hierarchy in a data dependent manner.
CONTACT INFORMATION

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

Name: Xin Wei
Enterprise: Hoffmann La Roche Ltd
Address: 340 kingsland street
City, State ZIP: Nutley, NJ, 09110
Work Phone: 973-235-2520
Fax: 973-235-2134
E-mail: xin.wei@roche.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.