Paper 413-2011

Using SAS® to Automatically Back Up Files across Multiple Directories

Laura Lawrence, Quality Data Services, Inc., King of Prussia, PA, USA

ABSTRACT

Often times we need to reference earlier work done on a project. Therefore it is necessary to save many different versions of certain files as we work. For example, you might need to compare a report you did 2 months ago to a more recent report or you need to compare an older SAS data set to a newer one. Therefore you must have all these older versions saved in a location that is easy to identify. This paper will take you through a macro useful in backing up various files of any type. The purpose being to create a new folder, named as the last modified date of the files to be copied, and copy over all files of the user-specified directory. This macro is set up so it can be run over multiple directories with ease. It works on the assumption that you want the entire directory backed up. This would come in handy when the programmer is responsible for saving all versions of their code and other files, such as the SAS data sets.

INTRODUCTION

When working on a project many iterations of code, data, and output are generated. It is usually necessary to save all these files at numerous times during the study in case reference to an earlier version is needed. If the project is quite large it can be cumbersome to create the dated folders and then copy and paste files. Sometimes such a task can slip our minds and then previous versions are lost as we might overwrite them.

This paper demonstrates a way to create the necessary folders named with the last modified date of the files and to copy over all those files into the appropriate directory. This macro will back up any external file type; it doesn't have to be just SAS files. The foundation of this program is that it relies on completing tasks by submitting commands in the operating environment rather than directly inside SAS.

CREATE FOLDER

First step is to create a folder. In this example I wanted to create a folder labeled with the last date modified for the files to be copied over (ignored other subdirectories). This macro checks if a folder exists already with the name of the last modified date and if so it creates another folder with that same date but adds a (2) at the end, i.e. if "20100729" already exists then create "20100729 (2)". If you are like me and forgot to back up until a few days later then setting this macro up so it takes the last modified date will be more meaningful to you when you reference it later.

```
%let fref=%unquote(%str(%')dir /Q %str(%")&path.\*.* %str(%")%str(%'));
```

The macro variable path must contain a user specified path. This is the parameter used in the macro called `dircopy`, referenced in Section "EXTEND TO MULTIPLE DIRECTORIES".

```
filename allfil pipe &fref;
data test;
  length v1 v2 v3 v4  owner filnam $100;
  infile allfil truncover end=eof;
  input v1 v2 v3 v4 owner filnam;
  if index(v1,'/') and v4^='<DIR>';  /*** v4 indicates if it is a subdirectory ***/
  MYDATE = trim(left(scan(v1,3,'/')))||
     trim(left(scan(v1,1,'/')))||trim(left(scan(v1,2,'/')));
  keep mydate;
run;

proc sql noprint;
       select max(mydate) into: moddate from test;
```

The purpose so far in the code was to get a date into the macro variable `moddate` for use when we create the folder. The variable `MYDATE` was created to get the date into a particular format needed. If this particular format is not preferable, you will need to replace it with your own preferred format. You cannot use the date as it comes out of this file since it contains symbols in it and folder names cannot contain such things.

We use that date to check if the folder has already been created, if so we add (2) at the end, otherwise we create as new.

```
options noxwait;
%local rc fileref ;
%let rc = %sysfunc(filename(fileref,%qcmpres(&path\&moddate))) ;
%if %sysfunc(fexist(&fileref))  %then
      %sysexec md "%qcmpres(&path\&moddate(2))" ;
%else %do ;
      %sysexec md "%qcmpres(&path\&moddate)" ;
%end ;
%let rc=%sysfunc(filename(fileref)) ;
```

To create our folder we used the `md` command with the `%sysexec` function. The command `md` stands for "make directory" and the macro `%sysexec` means you are submitting operating environment commands and then returning to your SAS session. Now we have a new folder created inside our specified directory. Next step is to populate it with all the files from the specified directory.

COPY FILES

We must get a data set that contains all the file names, extensions included, of the current directory being backed up. From that data set we will create a macro to cycle through each name and copy the files over individually.

```
%let indir=&path;
%let nfiles=0;
filename indir "&indir";

data _filenames1(where=(index(filename,'.')));
   format filename $45. ;
   did=dopen('indir');
   do i=1 to dnum(did);
       filename=dread(did,i);
              output;
   end;
   rc=dclose(did);
   keep filename ;
run;

data filenames;
       set _filenames1;
       ord=_n_;
run;

proc sql noprint;
       select max(ord) into: maxord
       from filenames;
quit;
```

We did a couple of things here. First we set up the `indir` directory which is set to your current specified path. Then we created a data set called `filenames`, using this directory as the input. We also created a variable called `ord` which is the observation number of each record in the data set. Lastly we took the highest observation number and set it into a macro variable called `maxord`. The macro variable `maxord` is needed because when we do the copying it will tell SAS when to stop looking for file names in the data set.

Next order of business is to set up a macro which will copy the files over into the new folder we created.

```
%macro copyfiles(num);
       proc sql noprint;
              select filename into: name
              from filenames
              where ord=&num;
       quit;

   options noxwait;
   %local rc fileref ;
```

```
   %let rc = %sysfunc(filename(fileref,%qcmpres(&path\&moddate(2)))) ;
   %if %sysfunc(fexist(&fileref))  %then %do;
    %sysexec copy "%qcmpres(&path\&name)"
            "%qcmpres(&path\%qcmpres(&moddate(2))\&name)"  ;
      %end;
   %else %do ;
   %let rc = %sysfunc(filename(fileref,%qcmpres(&path\&moddate))) ;
    %sysexec copy "%qcmpres(&path\&name)"
            "%qcmpres(&path\%qcmpres(&moddate)\&name)"  ;
   %end ;
   %let rc=%sysfunc(filename(fileref)) ;
%mend;
```

Again we used the `%sysexec` function to submit operating environment commands but this time we used the `copy` command. It is best to use this method as opposed to `file`/`infile` statements under `data _null_` because the `copy` command will not change the date of the file; the file will be kept completely intact without any changes. Whereas using a `data _null_` step in conjunction with `file`/`infile` statements means you are actually writing new files so the date will match the date you run this program.

We have set up some conditional statements for copying the files. SAS is checking first to see if there exists a folder with (2) as part of the name. If that exists, we want to copy the files there; otherwise we copy it to the folder that has only the date as the file name.

```
%macro gocopy;
       %do i=1 %to &maxord;
             %copyfiles(&i);
       %end;
%mend;
%gocopy;
```

The macro `gocopy` cycles through the file names by starting with the first observation (`i=1`) and ending on the last observation (`maxord`). All files will be accounted for this way.

To copy the files we used the data set called filenames to get the appropriate name of the file to be used in the macro variable name. We used `%sysexec` with the `copy` function to tell SAS the name of the file to be copied and where to copy it to. The additional macro `gocopy` will run through all the names within the data set filenames.

EXTEND TO MULTIPLE DIRECTORIES

All the code previously presented can be extended to multiple directories by placing it inside the `dircopy` macro whose only parameter is the user specified path. So we can automate this to back up across multiple directories.

```
%macro dircopy(path);

…
…

%mend;
%dircopy(&code);  /*** SAS CODE directory ***/
%dircopy(&list);  /*** OUTPUT directory ***/
%dircopy(&data);  /*** DATA directory ***/
```

CONCLUSION

I demonstrated a macro that can efficiently and quickly back up files across multiple directories using operating environment commands. This macro can be easily modified to use a different naming structure for the folders or only to copy certain file extensions if need be.

REFERENCES

SAS Institute Inc. (1999), *SAS Macro Language: Reference, Version 8,* Cary, NC: SAS Institute Inc.

ACKNOWLEDGMENTS

I would like to acknowledge the contribution of Xiaobo Du of Quality Data Services, Inc., who completed quality control checks on the `%dircopy` macro.

CONTACT INFORMATION

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

Name: Laura Lawrence
Enterprise: Quality Data Services, Inc.
Address: 2500 Renaissance Blvd
City, State ZIP: King of Prussia, PA 19406
Work Phone: (610) 354-0404x118
Fax: (610) 354-0402
E-mail: llawrence@qdservices.com
Web: www.qdservices.com