Paper 1692-2014

You Can Have It All: Building Cumulative Datasets

Myra A. Oltsik
Acorda Therapeutics, Inc., Ardsley, NY

ABSTRACT

We receive daily files from a vendor which are updated every day so that we have the most current information. The problem with this kind of file is that valuable pieces of information can change over time, so that historical information is lost. Whether it's an address, school, supplier, status of an order, length of time to deliver something, etc., this type of file can only be used to analyze what's happening now. But what if you wanted to capture information that changed? To avoid losing these changes, you have to build a cumulative dataset. I'll show you how to build it.

INTRODUCTION

Acorda Therapeutics, Inc. is a small but growing Biopharmaceutical company developing therapies to restore neurological function and improve the lives of people with spinal cord injury, multiple sclerosis and other disorders of the central nervous system. I work in the Sales Operations Department in the Commercial Division. My colleagues and I are responsible for all types of reports for senior management.

I was asked to do an ad hoc report about showing a change over time in one area, but couldn't do it with the daily file. I decided to go back to the beginning of our product's launch in March 2010 and put all the daily records into one cumulative file. With 3 years from launch, such a file would be cumbersome, and contain many duplicate records with only the timestamp differing. I needed to come up with a way to delete records where *most* of the variables were duplicated.

STEP 1: GATHER ALL FILE NAMES INTO A DATASET

By the time I started on this application, we had received over 550 daily files. The first task was to get a list of all these data files with their dates and times. The files had a standard naming convention — DRUG_NAME_yyyymmddhhmmss.txt — helping my programming to access them. Using the SAS®[1] X command, the DOS *dir* command was issued for all folders (one for each year) where the files resided and put into text files, as in the following example.

```
x 'dir "s:\XXX Daily Files\2010\*.txt" > "k:\files2010.txt"';
```

The directory text files, like files2010.txt, were put in a file name statement and then read into a dataset. The dataset would have each file name, date and time as variables.

[1] 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.

```
data macparm1;
   infile mydata lrecl=1000 truncover;
   input
      @ 40 file   $40.
      @ 29 fbytes comma10.
   ;
   if substr(file,1,12) = "DRUG_NAME";
   year   = substr(file,14,4);
   month  = substr(file,18,2);
   day    = substr(file,20,2);
   hour   = substr(file,22,2);
   minute = substr(file,24,2);
   second = substr(file,26,2);
   fdate  = mdy(month,day,year);
   ftime  = hms(hour,minute,second);
   format
      fbytes comma12.
   ;
   drop
      month -- second
   ;
run;
```

The path name was later added to the file name. Using a dictionary table, I created a macro variable with the number of observations equal to the number of files to be accumulated. Next, I created macro variables with the file name, date, and time. Finally, I created a macro to read in each individual file into a dataset, *daily_file*, and then compare the records. Each dataset would be sorted by the ID for the record, then the file date and time.

STEP 2: COMPARE TODAY'S RECORD WITH YESTERDAY'S

Out of over 30 variables for each daily record, the Timestamp variable was the only one guaranteed to change daily. All other variables had the potential to change. The idea was to compare all other variables with their values from the previous day's file for each ID.

I couldn't use PROC SQL with SELECT DISTINCT since the timestamp changed every day. Nor could I use PROC SORT with the NODUPE option. The process would need a data step solution.

The first *daily_file* was read in and copied to a file for accumulating the data. From the second iteration on, the cumulative file, *daily_file_cum*, was set with the newest *daily_file*, by the ID, date and time.

A lag variable was created and retained for each of the variables to be compared, and arrays were set up so the comparison could be done in a do loop. The number of items in the arrays equals the number of variables being compared. That value is established at the beginning of the program in the macro variable *&nlag*. For each first ID, the lag variable takes the value of the present variable. For each of the additional records for that ID, the lag value is compared to

the present value. If they are identical, a count variable is iterated. Then, the lag takes the present value. If the count is equal to the number of variables being compared, it is deemed a duplicate of the previous record and is not needed. The record is deleted. The essential part of the code is:

```
if first.INF_ID then do i = 1 to &nlag.;
   lag{i} = inf{i};
end;
else do i = 1 to &nlag.;
   if inf{i} = lag{i} then COUNT = COUNT + 1;
   lag{i} = inf{i};
end;
if COUNT = &nlag. then delete;
```

(The complete program can be found in Appendix A.)

Our file has ~75,000 records at this time. The cumulative file has nearly 1.5 million records. However, if I had not eliminated the [*nearly*] duplicate records, there could be millions more records without any additional information.

Step 3: Adding Daily Data to the Cumulative File

Once the cumulative file was built, I needed to write a similar program so that each subsequent days' files would be added and kept up to date. The heart of the program is the same: comparing the new file records with those already in the cumulative file.

In case there was a problem with a daily file, I wanted to keep several backup files to ensure no information was lost, and so the cumulative file would not have to be rebuilt. Using PROC DATASETS easily accomplished that:

```
proc datasets lib=d nolist;
   delete
      daily_cum_file_back4
   ;
   change
      daily_cum_file_back3 = daily_cum_file_back4
      daily_cum_file_back2 = daily_cum_file_back3
      daily_cum_file_back1 = daily_cum_file_back2
   ;
run; quit;
```

The file to which the new data is appended and tested is the one created the previous day:

```
proc sort data=ds.daily_cum_file
   out=ds.daily_cum_file_back1(compress=yes);
   by INF_ID INF_File_Date INF_File_Time;
run;
```

Next, *daily_file* is set with backup1 and then compared. The new cumulative file is saved to a permanent file. Appendix B has the code for this step.

CONCLUSION

With a few DOS commands and creating lag variables, one can produce a cumulative data file in which every record adds and saves crucial changes from daily files with only the most recent information.

Myra A Oltsik
Sales Operations Analyst
Acorda Therapeutics®
420 Saw Mill River Rd
Ardsley NY 10502
moltsik@acorda.com

APPENDIX A

```
/*****************************************************************************************/
/* PROGRAM:       File History Build                                                     */
/* AUTHOR:        Myra A Oltsik                                                          */
/* ORIGINAL DATE: 20121019                                                               */
/* PURPOSE:       Collects all the daily files from XXX, LAUNCH TO DATE, to create a     */
/*                history file with any changes made to a record.                        */
/* LAST CHANGE:   20121113 Add code so for exact duplicate records, only last is kept.   */
/*****************************************************************************************/

%let begtime = %sysfunc(datetime()) ;

%let drv   = \\sas\SO;
%let path  = &drv.\Warehouse;
%let mpath = &path.\Macros;
%let prgnm = File History Build;

/*****************************************************************************************/
/* NOTE THE NUMBER OF VARIABLES TO BE COMPARED FOR EACH RECORD.                          */
/*****************************************************************************************/
%let nlag  = 18;

libname fmts "&path.\Formats";
libname ds   "&path.\Datasets";

options
   compress=no
   errors=1
   fmtsearch=(genfmts)
   fullstimer
   linesize=139
   nocenter
   nomerror
   nomfile
   nomlogic
   nomprint
   notes
   noxsync
   noxwait
   obs=max
   pageno=1
   pagesize=40
   sasautos=("&mpath", sasautos)
   source
   ;
title;

data _null_;
   rundt = put(date(),yymmddn8.);
   runtm = put(input(compress(put(time(),time.),":"),6.),z6.);
   call symputx ('rundt',rundt);
   call symputx ('runtm',runtm);
```

```
run;

options
   noxwait;

/*************************************************************************************/
/* FIND AND READ IN THE XXX TEXT FILES LIKE 'DRUG_NAME_yyyymmddhhmmdd.txt' USEING DOS   */
/* COMMANDS.                                                                         */
/*************************************************************************************/
x 'dir "s:\XXX Daily Files\2010\*.txt" > "k:\files2010.txt"';
x 'dir "s:\XXX Daily Files\2011\*.txt" > "k:\files2011.txt"';
x 'dir "s:\XXX Daily Files\2012\*.txt" > "k:\files2012.txt"';
x 'dir "s:\XXX Daily Files\2013\*.txt" > "k:\files2013.txt"';

data _null_;
   rc = sleep(5);
run;

filename mydata
("k:\files2010.txt","k:\files2011.txt","k:\files2012.txt","k:\files2013.txt");

/*************************************************************************************/
/* FIND THE FILE DATE AND TIME IN THE FILE NAME AND CREATE VARIABLES FOR THEM.       */
/*************************************************************************************/
data macparm1;
   infile mydata lrecl=1000 truncover;
   input
      @ 40 file   $40.
      @ 29 fbytes comma10.
   ;
   if substr(file,1,12) = "DRUG_NAME";
   year   = substr(file,14,4);
   month  = substr(file,18,2);
   day    = substr(file,20,2);
   hour   = substr(file,22,2);
   minute = substr(file,24,2);
   second = substr(file,26,2);
   fdate  = mdy(month,day,year);
   ftime  = hms(hour,minute,second);
   format
      fbytes comma12.
   ;
   drop
      month -- second
   ;
run;

/*************************************************************************************/
/* ADD FULL FILE PATH TO THE FILE NAME VARIABLE.                                     */
/*************************************************************************************/
data macparm2;
   length
      fname $250.
```

```
   ;
   set macparm1;
   fname = "s:\XXX Daily Files\"||strip(year)||"\"||strip(file);
   drop
      file
      year
   ;
run;

/*****************************************************************************************/
/* FIND THE NUMBER OF OBSERVATIONS OF FILES THAT WILL BE READ IN.                        */
/*****************************************************************************************/
proc sql noprint;
   select
      nobs
   into
      :fnobs
   from
      sashelp.vtable
   where
      libname = "WORK" and
      memname = "MACPARM2"
   ;
quit;

%let fobs = %left(%strip(&fnobs));

%put &fobs;

/*****************************************************************************************/
/* CREATE MACRO VARIABLES FOR FILE NAME, DATE, AND TIME.                                 */
/*****************************************************************************************/
proc sql noprint;
   select
      fname,
      fdate,
      ftime
   into
      :fn1 - :fn&fobs,
      :fd1 - :fd&fobs,
      :ft1 - :ft&fobs
   from
      macparm2
   order by
      fname
   ;
quit;

%put &&fn&fobs &&fd&fobs &&ft&fobs;

/*****************************************************************************************/
/* THIS READ MACRO IS BASED ON THE RBD STANDARD LAYOUT.                                  */
/*****************************************************************************************/
```

```
%macro readfiles;
   proc datasets lib=work nolist;
      delete daily_file_cum;
   run; quit;
   %*do z = 1 %to 5;
   %do z = 1 %to &fnobs;
      %let fname  = &&fn&z;
      %let fdate  = &&fd&z;
      %let ftime  = &&ft&z;
      %*put &fdate &ftime;

      data daily_file;
         infile "&fname" delimiter='|' MISSOVER DSD lrecl=32767;
         informat
            INF_RecordTimestamp     $16.
            INF_ID                  $10.
            INF_EnrollmentDate      $16.
            INF_Gender              $1.
            INF_Status              $20.
            INF_StatusReasonCode    $50.
            ...
               <snip>
            ...
            INF_SpecialPg           $8.
            INF_SpecialPg_orderdate $10.   /* ADDED 02/06/2012 */
         ;
         format
            INF_RecordTimestamp     $16.
            INF_ID                  $10.
            INF_EnrollmentDate      $16.
            INF_Gender              $1.
            INF_Status              $20.
            INF_StatusReasonCode    $50.
            ...
               <snip>
            ...
            INF_SpecialPg           $8.
            INF_SpecialPg_orderdate $10.   /* ADDED 02/06/2012 */
         ;
         input
            INF_RecordTimestamp     $
            INF_ID                  $
            INF_EnrollmentDate      $
            INF_Gender              $
            INF_Status              $
            INF_StatusReasonCode    $
            ...
               <snip>
            ...
            INF_SpecialPg           $
            INF_SpecialPg_orderdate $  /* ADDED 02/06/2012 */
         ;
         format
```

```
            File_Date mmddyy10.
            File_Time time8.
         ;
         File_Date = &fdate.;
         File_Time = &ftime.;
      run;

      proc sort data=daily_file;
         by INF_ID File_Date File_Time;
      run;

      %if &z = 1 %then %do;
/*****************************************************************************/
/* FOR FIRST ITERATION CREATE THE CUMMULATIVE FILE.                          */
/*****************************************************************************/
         data daily_file_cum;
            set daily_file;
            by INF_ID File_Date File_Time;
         run;
      %end;
      %else %do;
/*****************************************************************************/
/* FOR THE REST OF THE FILES CONCATINATED THE NEXT FILE WITH THE CUMMULATIVE FILE.  */
/*****************************************************************************/
         data daily_file_cum;
            length
               REC 8.
            ;
            set
               daily_file_cum
               daily_file
            ;
            by INF_ID File_Date File_Time;
            REC = _N_;
/*****************************************************************************/
/* CREATE AND RETAIN A LAG VARIABLE FOR EACH FIELD/VARIABLE TO COMPARE. BECAUSE THE  */
/* RETAIN FIELDAUTOMATICALLY CREATES NUMERIC VARIABLES, THE CHARACTER VARIABLES MUST BE */
/* SPECIFICALLY STATED.                                                      */
/*****************************************************************************/
            length
               LAG_EnrollmentDate      $16.
               LAG_Gender              $1.
               LAG_Status              $20.
               LAG_StatusReasonCode    $50.
               ...
                  <snip>
               ...
               LAG_SpecialPg           $8.
               LAG_SpecialPg_orderdate $10.
            ;
            retain
               LAG_EnrollmentDate      $
               LAG_Gender              $
```

```
            LAG_Status              $
            LAG_StatusReasonCode    $
            ...
               <snip>
            ...
            LAG_SpecialPg           $
            LAG_SpecialPg_orderdate $
         ;
/*****************************************************************************************/
/* CREATE ARRAYS FOR THE FILE VARIABLES AND THE LAG VARIABLES.                           */
/*****************************************************************************************/
         array inf{&nlag.}
            INF_EnrollmentDate
            INF_Gender
            INF_Status
            INF_StatusReasonCode
            ...
               <snip>
            ...
            INF_SpecialPg
            INF_SpecialPg_orderdate
         ;
         array lag{&nlag.}
            LAG_EnrollmentDate
            LAG_Gender
            LAG_Status
            LAG_StatusReasonCode
            ...
               <snip>
            ...
            LAG_SpecialPg
            LAG_SpecialPg_orderdate
         ;
         COUNT = 0;
/*****************************************************************************************/
/* FOR EVERY ID, COMPARE ALL FIELDS FROM ONE RECORD WITH THE RECORD PRECEEDING IT. IF    */
/* ALL THE VARIABLES ARE EQUAL TO THOSE IN THE PREVIOUS RECORD, DELETE THE IT. THE       */
/* RECORD HAD NO CHANGE FROM ONE DAY TO THE NEXT SO IT IS NOT NEEDED.                    */
/*****************************************************************************************/
         if first.INF_ID then do i = 1 to &nlag.;
            lag{i} = inf{i};
         end;
         else do i = 1 to &nlag.;
            if inf{i} = lag{i} then COUNT = COUNT + 1;
            lag{i} = inf{i};
         end;
         if COUNT = &nlag. then delete;
         drop
            i
            LAG_:
         ;
      run;
   %end;
```

```
      proc datasets lib=work nolist;
         delete daily_file;
      run; quit;
   %end;
%mend;

options mprint;
%readfiles;

/*****************************************************************************************/
/* CREATE AND SAVE A PERMANENT DATASET TO THE LIBRARY.                                   */
/*****************************************************************************************/
proc sql;
   create table ds.daily_cum_file(compress=yes) as
   select
      File_Date as INF_File_Date,
      File_Time as INF_File_Time,
      INF_ID,
      INF_EnrollmentDate,
      INF_Gender,
      INF_Status,
      INF_StatusReasonCode,
      ...
         <snip>
      ...
      INF_SpecialPg,
      INF_SpecialPg_orderdate
   from
      daily_file_cum
   order by
      INF_ID,
      File_Date,
      File_Time,
   ;
quit;

%let tottime = %sysevalf ( %sysfunc(datetime()) - &begtime ) ;
%put Total Execution Time: %sysfunc(putn(&tottime,time9.)) ;
```

APPENDIX B

```
/*************************************************************************************/
/* PROGRAM:       File History Add                                                   */
/* AUTHOR:        Myra A Oltsik                                                      */
/* ORIGINAL DATE: 20130221                                                           */
/* PURPOSE:       Adds daily XXX Daily Files to cumulative history.                  */
/* LAST CHANGE:                                                                      */
/*************************************************************************************/

%let begtime = %sysfunc(datetime()) ;

/*************************************************************************************/
/* NOTE THE NUMBER OF VARIABLES TO BE COMPARED FOR EACH RECORD.                      */
/*************************************************************************************/
%let nlag    = 18;

proc datasets lib=sw nolist;
   delete
      daily_cum_file_back4
   ;
   change
      daily_cum_file_back3 = daily_cum_file_back4
      daily_cum_file_back2 = daily_cum_file_back3
      daily_cum_file_back1 = daily_cum_file_back2
   ;
run; quit;

options
   noxwait;

/*************************************************************************************/
/* FIND AND READ IN THE XXX TEXT FILE FROM THE PREVIOUS DAY LIKE                     */
/* 'DRUG_NAME_yyyymmddhhmmdd.txt' USEING DOS COMMANDS                                */
/*************************************************************************************/
x 'dir "s:\XXX_scrubbed\*.txt" > "K:\pat.txt"';

data _null_;
   rc = sleep(5);
run;

filename mydata "K:\pat.txt";

/*************************************************************************************/
/* FIND THE FILE DATE AND TIME IN THE FILE NAME AND CREATE VARIABLES FOR THEM.       */
/*************************************************************************************/
data macparm1;
   infile mydata lrecl=1000 truncover; *recfm=v;
   input
      @ 40 file   $40.
      @ 29 fbytes comma10.
   ;
   if substr(file,1,12) = "DRUG_NAME";
```

```
   year   = substr(file,14,4);
   month  = substr(file,18,2);
   day    = substr(file,20,2);
   hour   = substr(file,22,2);
   minute = substr(file,24,2);
   second = substr(file,26,2);
   fdate  = mdy(month,day,year);
   ftime  = hms(hour,minute,second);
   format
      fbytes comma12.
   ;
   drop
      year -- second
   ;
run;

/*****************************************************************************************/
/* ADD FULL FILE PATH TO THE FILE NAME VARIABLE.                                         */
/*****************************************************************************************/
data macparm2;
   length
      fname $250.
   ;
   set macparm1;
   fname = "s:\XXX_scrubbed\"||strip(file);
   drop
      file
   ;
run;

/*****************************************************************************************/
/* CREATE MACRO VARIABLES FOR FILE NAME, DATE, AND TIME.                                 */
/*****************************************************************************************/
proc sql noprint;
   select
      fname,
      fdate,
      ftime
   into
      :fn,
      :fd,
      :ft
   from
      macparm2
   order by
      fname
   ;
quit;

%put &fn &fd &ft;

/*****************************************************************************************/
/* THIS CODE IS BASED ON THE RBD STANDARD LAYOUT.                                        */
```

```
/*****************************************************************************/
```

```
data daily_file;
   infile "&fn" delimiter='|' MISSOVER DSD lrecl=32767;
   informat
      INF_RecordTimestamp     $16.
      INF_ID                  $10.
      INF_EnrollmentDate      $16.
      INF_Gender              $1.
      INF_Status              $20.
      INF_StatusReasonCode    $50.
      ...
         <snip>
      ...
      INF_SpecialPg           $8.
      INF_SpecialPg_orderdate $10.   /* ADDED 02/06/2012 */
   ;
   format
      INF_RecordTimestamp     $16.
      INF_ID                  $10.
      INF_EnrollmentDate      $16.
      INF_Gender              $1.
      INF_Status              $20.
      INF_StatusReasonCode    $50.
      ...
         <snip>
      ...
      INF_SpecialPg           $8.
      INF_SpecialPg_orderdate $10.   /* ADDED 02/06/2012 */
   ;
   input
      INF_RecordTimestamp     $
      INF_ID                  $
      INF_EnrollmentDate      $
      INF_Gender              $
      INF_Status              $
      INF_StatusReasonCode    $
      ...
         <snip>
      ...
      INF_SpecialPg           $
      INF_SpecialPg_orderdate $
   ;
   format
      File_Date mmddyy10.
      File_Time time8.
   ;
   File_Date = &fd.;
   File_Time = &ft.;
run;

proc sort data=daily_file;
   by INF_ID File_Date File_Time;
run;
```

```
/*****************************************************************************/
/* COPY THE PREVIOUS FILE TO IT'S NEXT GENERATION NAME.                      */
/*****************************************************************************/
proc sort data=ds.daily_cum_file out=ds.daily_cum_file_back1(compress=yes);
   by INF_ID INF_File_Date INF_File_Time;
run;

/*****************************************************************************/
/* CONCATINATE THE PREVIOUS FILE WITH THE NEW FILE.                          */
/*****************************************************************************/
data daily_file_cum;
   length
      REC 8.
   ;
   set
      ds.daily_cum_file_back1(rename=(
         INF_File_Date = File_Date
         INF_File_Time = File_Time
      ))
      daily_file
   ;
   by INF_ID File_Date File_Time;
   REC = _N_;
/*****************************************************************************/
/* CREATE AND RETAIN A LAG VARIABLE FOR EACH FIELD/VARIABLE TO COMPARE. BECAUSE THE     */
/* RETAIN FIELDAUTOMATICALLY CREATES NUMERIC VARIABLES, THE CHARACTER VARIABLES MUST BE */
/* SPECIFICALLY STATED.                                                      */
/*****************************************************************************/
   length
      LAG_EnrollmentDate        $16.
      LAG_Gender                $1.
      LAG_Status                $20.
      LAG_StatusReasonCode      $50.
      ...
         <snip>
      ...
      LAG_SpecialPg             $8.
      LAG_SpecialPg_orderdate   $10.
   ;
   retain
      LAG_EnrollmentDate
      LAG_Gender
      LAG_Status
      LAG_StatusReasonCode
      ...
         <snip>
      ...
      LAG_SpecialPg
      LAG_SpecialPg_orderdate
   ;
/*****************************************************************************/
/* CREATE ARRAYS FOR THE FILE VARIABLES AND THE LAG VARIABLES.               */
/*****************************************************************************/
```

```
   array inf{&nlag.}
      INF_EnrollmentDate
      INF_Gender
      INF_Status
      INF_StatusReasonCode
      INF_StatusChangeDateTime
      ...
         <snip>
      ...
      INF_SpecialPg
      INF_SpecialPg_orderdate
   ;
   array lag{&nlag.}
      LAG_EnrollmentDate
      LAG_Gender
      LAG_Status
      LAG_StatusReasonCode
      ...
         <snip>
      ...
      LAG_SpecialPg
      LAG_SpecialPg_orderdate
   ;
   COUNT = 0;
/*****************************************************************************************/
/* FOR EVERY ID, COMPARE ALL FIELDS FROM ONE RECORD WITH THE RECORD PRECEEDING IT. IF    */
/* ALL THE VARIABLES ARE EQUAL TO THOSE IN THE PREVIOUS RECORD, DELETE THE IT. THE       */
/* RECORD HAD NO CHANGE FROM ONE DAY TO THE NEXT SO IT IS NOT NEEDED.                    */
/*****************************************************************************************/
   if first.INF_ID then do i = 1 to &nlag.;
      lag{i} = inf{i};
   end;
   else do i = 1 to &nlag.;
      if inf{i} = lag{i} then COUNT = COUNT + 1;
      lag{i} = inf{i};
   end;
   if COUNT = &nlag. then delete;
   drop
      i
      LAG_:
   ;
run;

/*****************************************************************************************/
/* CREATE AND SAVE A PERMANENT DATASET TO THE LIBRARY.                                   */
/*****************************************************************************************/
proc sql;
   create table ds.daily_cum_file(compress=yes) as
   select
      File_Date as INF_File_Date,
      File_Time as INF_File_Time,
      INF_ID,
      INF_EnrollmentDate,
```

```
      INF_Gender,
      INF_Status,
      INF_StatusReasonCode,
      ...
         <snip>
      ...
      INF_SpecialPg,
      INF_SpecialPg_orderdate
   from
      daily_file_cum
   order by
      INF_ID,
      File_Date,
      File_Time
   ;
quit;

%let tottime = %sysevalf ( %sysfunc(datetime()) - &begtime ) ;
%put Total Execution Time: %sysfunc(putn(&tottime,time9.)) ;
```