

Efficient and Smart Ways to Manage Datasets for Clinical Data- Let SAS® Do the Dirty Laundry!

Gowri Madhavan, Cincinnati Children's Hospital and Medical Center, Cincinnati, OH

Alan Leach, Cincinnati Children's Hospital and Medical Center, Cincinnati, OH

ABSTRACT

The ASQ (ages and stages) is an important questionnaire that is delivered to parents to assess any developmental delay in children 9-24 months. If the child fails an ASQ, he/she is referred to developmental agency at the initial stages for treatment. A child can have several well child care visits, the child can either pass or fail an ASQ, a child can pass once and fail subsequently or vice versa. It is important to capture the number of tests administered and the most recent test results for failures. However, dealing with these test results on a weekly basis for reporting public health information can be daunting! A patient's clinical history can vary and we don't always know what datasets to expect. Fortunately SAS® provides us ways to dynamically intake and process this information with a minimum of effort to maintain. This paper discusses time saving techniques used to manage the intake and processing of numerous clinical data sources. Using macros, we will read in number of clinical datasets which can vary in number and name from week to week. Through the use of PROC TRANPOSE we will reshape the data for further processing. Using DICTIONARY.TABLES to capture information about our data, we will initialize MACRO variables dynamically and build ARRAYS and DO LOOPS to process each patient's ASQ test results history enabling us to categorize them per requirements. Then by summarizing these results and exporting into Excel spreadsheets, we can automatically email results to stakeholders.

INTRODUCTION

Early identification of developmental disorders is critical to the well-being of children and their families. Delayed or disordered development may also indicate an increased risk of behavior disorders or associated developmental disorders. These can be caused by specific medical conditions, and if detected early, they can be evaluated, diagnosed and treated. The ASQ (Ages and Stages Questionnaire) is a standardized developmental screening tool, developed for use in primary care, but is also used locally in childcare centers and home visiting programs. The ASQ addresses multiple domains of development, and it is recommended that the results are always taken into consideration along with clinical judgment about whether there is really a delay. It is completed by the parent, usually taking about 10 minutes for the parents to fill it out. It is based on parent report of the child's developmental milestones, not direct observation. The American Academy of Pediatrics recommends administering this screening test at the 9-, 18-, and 30- month visits. (Because the 30-month visit is not yet a part of the preventive care system and is often not reimbursable by third-party payers at this time, developmental screening can be performed at 24 months of age. The detection of developmental disorder is an integral component of well-child care. Kids with developmental delay are less likely to be ready for school. Children diagnosed with developmental disorders are identified as children with special health care needs, and typically, chronic-condition management is initiated. This could drive a range of treatment plans from medical treatment of the child to family planning for his or her parents. Kids with delay who get referred for services to early intervention programs have an improved developmental pathway than kids who do not.

The Division of General and Community Pediatrics at Cincinnati Children's Hospital offers comprehensive primary well-child care visits. We at the CCHMC affiliated primary care clinics, administer the age appropriate ASQ at every well child visit between 6 months to 30 months. The ASQ data is collected during the clinic visit via a paper-pencil administration of the survey (see example below). The survey is administered by the clinician and contains both observational and parent reported data. The questionnaire and scoring sheet results are recorded in EPIC (explained elsewhere in the data architecture section).

Double-Click example below:



ASQ 3 Ages & Stages Questionnaires®
15 months 0 days through 16 months 30 days
16 Month Questionnaire

Please provide the following information. Use black or blue ink only and print legibly when completing this form.

Date ASQ completed: 9/20/2008

Child's information

Child's first name: Annie Middle initial: M. Child's last name: Roberts
 Child's date of birth: 5/5/2007 if child was born 2 or more weeks prematurely, if of week premature: Male Female

Person filling out questionnaire

First name: Jennifer Middle initial: M. Last name: Roberts
 Street address: 33 Main Street Relationship to child: Parent Guardian Teacher Child care provider
 Grandparent or other relative Foster parent Other:
 City: Jonestown State/Province: IN ZIP/Postal code: 61924
 Country: USA Home telephone number: 219-888-0021 Other telephone number: 219-912-2100
 E-mail address: jennifer_roberts@email.com
 Names of people assisting in questionnaire completion:

Program Information

Child ID #: 36759111023412358 Age at administration in months and days: 16 months, 15 days
 Program ID #: 6222001439183664 if premature, adjusted age in months and days:
 Program name: Jonestown Child Care Center

P101160100 Ages & Stages Questionnaires Third Edition (ASQ-3™) Squires & Bricker © 2009 Paul H. Brookes Publishing Co. All rights reserved.

Figure 1. Sample ASQ Questionnaire

Description of the Data Architecture and Rationale of the measurement-

The repository for this ASQ data, which is an integral component of the well children visit encounter, is EPIC (Electronic Privacy Information Center). We at Cincinnati Children's use EPIC which is a fully integrated clinical and hospital information system for managing health records that encompasses not only clinical information, but also registration, patient scheduling and billing. EPIC Clarity is EPIC's Enterprise data warehouse build upon either Oracle or Microsoft SQL Server relational database. The Clarity feature is used to generate scheduled and automated weekly reports for a list of patients who are scheduled for well child visits at the primary care clinics. This report is the source for information on the patient's demographic information, well child visits, immunization, results of ASQ along with dates and age when administered.

The customer's (Division of General Pediatrics) request is to generate a monthly report that would facilitate them to track and report the number of successful intakes with developmental specialist for the children with primary care clinics who fail an ASQ between 9 and 24 months of age. Several criteria's were specified that include age (between 9 and 24 months), if ASQ was administered or not (yes/no), status of ASQ test (pass/fail), repeated measurements of ASQ test and results, capturing the most recent results for failures (a child can pass the test and subsequently fail or vice-versa), date of test and ages when administered.

The goal is to produce a monthly report that will display the number of children who fail an ASQ test with date and age in months and date administered repeated measurements wherever applicable for the child and demographic information. The statistic is expressed as the number of successful intakes with developmental specialists for children who fail an ASQ between 9 months and 24 months of age. The number is reported monthly, displayed in tabular format and plotted in a chart

OBTAINING THE MOST RECENT RAW DATA FILE

The raw data files that are received for this project vary in a couple major aspects making automated processing a challenge. However, we wanted to minimize manual intervention (e.g. code changes) as much as possible. The first

source of variance in the files was the name. The names of the files vary according the date that they are produced. For example, we receive excel files called 'ppc_*mmddyy*.xls' where *mmddyy* is the date that the file was produced. To keep from changing our code to accommodate new files we used a routine to determine what was the most recent file received based on this *_mmddyy* suffix.

First we used a SAS® supplied macro to get a listing of all the raw files in our directory:

```
%DIRLISTWIN ( R:\MWSUG , OUT=PPC_FILES ) ;
```

The %DIRLISTWIN is a powerful utility macro that provides information about the files residing in the specified WINDOWS® directory. In the above invocation we use just one optional parameter – OUT – to specify the name of a data set to house the file information. Our dataset WORK.PPC_FILES looks like this:

	path	filename	filepath	owner	date	time	size
1	R:\MWSUG\raw_files						
2	R:\MWSUG\raw_files	ppc_07232013.xls	R:\MWSUG\raw_files\ppc_07232013.xls	CHMCCORP\LEAFF4	19585	42780	1.4E6
3	R:\MWSUG\raw_files	ppc_07302013.xls	R:\MWSUG\raw_files\ppc_07302013.xls	CHMCCORP\LEAFF4	19585	39720	1.4E6
4	R:\MWSUG\raw_files	ppc_08062013.xls	R:\MWSUG\raw_files\ppc_08062013.xls	CHMCCORP\LEAFF4	19585	42780	1.4E6
5	R:\MWSUG\raw_files	ppc_08132013.xls	R:\MWSUG\raw_files\ppc_08132013.xls	CHMCCORP\LEAFF4	19585	32520	1.03E6

(NOTE: The %DIRLISTWIN macro makes a call to the operating system. So your SAS® session needs to be able to issue those operating system commands, e.g. X COMMAND. In some installations of SAS Enterprise Guide® this may be disabled by default. However, if your site allows it you can configure SAS Enterprise Guide® to allow system commands using the tip listed in the references.)

Once we have a dataset with the names of the raw files we can pick the raw file with most recent date suffix. A simple PROC SQL® routine is used to query the dataset and populate a SAS MACRO® variable with the name of the most recent file:

```
proc sql ;

    select trim(filename) ,
           input ( scan ( filename, 2, "_" ) , mmddyy8. )
           into :last_report, :last_report_date
           from PPC_FILES
           where filename like "%.xls" and
           input ( scan ( filename, 2, "_" ) , mmddyy8. )
           = ( select max ( input ( scan ( filename, 2, "_" ) , mmddyy10. ) )
               from PPC_FILES
               where filename like "%.xls"
             )
           ;

quit;
```

In the above code the nested functions “input (scan (filename, 2, "_") , mmddyy8.)” parse the report date from the name of the file and convert it to SAS date time. We use an inline query to select just the file with the latest date:

```
input ( scan ( filename, 2, "_" ) , mmddyy8. )
      = ( select max ( input ( scan ( filename, 2, "_" ) , mmddyy10. ) )
          from PPC_FILES
          where filename like "%.xls"
        )
```

The “into” statement sets the value of the macro variables &LAST_REPORT and the &LAST_REPORT_DATE to the value of the name of last raw file and the date of last raw file , respectively. Now that we have dynamically determined the newest file we can pass that to our libname statement.

```
libname ppc excel R:\MWSUG\&last_report";
```

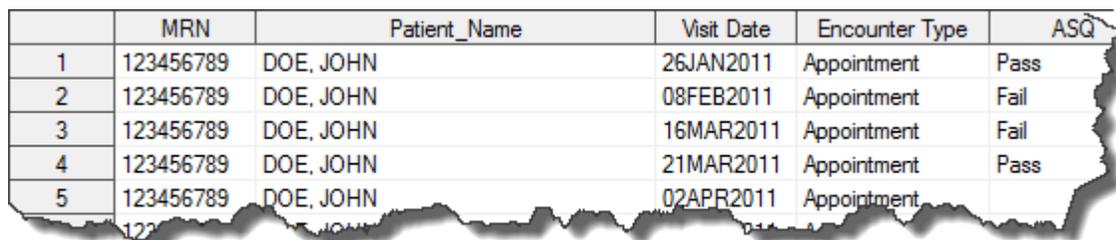
Using the Excel LIBNAME engine makes retrieval of the raw data syntactically very simple. In our raw spreadsheets there is tab named 'Primary Care Visits'. When we assign the Excel LIBNAME the data on this tab is available as a SAS® dataset. The dataset is referenced using a name literal:

```
data ppc;  
    set ppc.'Primary Care Visits$'n;  
    where visit_date < &LAST_REPORT_DATE ;  
run;
```

We have the WHERE clause just to limit visits occurring during the prior month. To do this we make use of date of last report determined previously.

PROCESSING THE RAW DATA

Now that we have retrieved the data from the correct raw file we need to analyze it by examining all ASQ scores for the current set of visits. In order to do that we need to deal with another source of variability in the file, namely, that the number of visits for each patient changes. So we need to be able to process different numbers of visits from file to file. The raw data contains records where each patient visit is a separate record. For example:



	MRN	Patient_Name	Visit Date	Encounter Type	ASQ
1	123456789	DOE, JOHN	26JAN2011	Appointment	Pass
2	123456789	DOE, JOHN	08FEB2011	Appointment	Fail
3	123456789	DOE, JOHN	16MAR2011	Appointment	Fail
4	123456789	DOE, JOHN	21MAR2011	Appointment	Pass
5	123456789	DOE, JOHN	02APR2011	Appointment	

Because we have requirements to examine the ASQ scores at each visit and the dates of the visits we transpose the data. After sorting by MRN and VISIT_DATE we executed the following

```
proc transpose data=ppc out=tran1 prefix=visit ;  
    by mrn;  
    var visit_date ;  
run;  
  
proc transpose data=ppc out=tran2 prefix=asq ;  
    by mrn;  
    var asq;  
run;
```

Once we have transposed the ASQ scores and the visits dates we merge them back together by MRN. Now we have a single record for each patient with their complete scoring history. The resulting dataset – TRANSPOSED - looks like this:

	MRN	visit1	visit2	visit3	visit4	visit5	asq1	asq2	asq3	asq4	asq5
1	11111111	05MAR2012	30JAN2013	.	.	.	Pass	Pass			
2	11111112	14NOV2012					Pass				
3	11111113	16NOV2011	07MAR2012	13JUN2012	04OCT2012	13FEB2013	Pass	Pass	Pass	Pass	Pass

Notice now that we have columns for each visit date and for the score at each visit. Also, each patient's history is in sequential and consecutive order. To examine these visits we found it helpful to use an ARRAY. However, to set up the array we needed to know how many *VISIT_n* and *ASQ_n* columns are in the newly transposed data. To determine this we used the `DICTIONARY.TABLES` available PROC SQL to look at all the *VISIT_n* columns in the above table and return the largest value.

```
proc sql noprint;
  select left ( put ( count ( * ), 3. ) ) into :MAX_VISITS
  from dictionary.columns
  where libname = "WORK" and MEMNAME = "TRANPOSED"
  and name like "visit%"
  ;
quit;
```

The above creates a SAS® MACRO variable - `&MAX_VISITS` – that has a value of the maximum number of visits and ASQ scores in the data. This can be used in a subsequent data step to create an ARRAY:

```
data final;

  set transposed ;

  array asq_dates { &MAX_VISITS } visit1 -- visit&MAX_VISITS ;
  array asq_scores { &MAX_VISITS } asq1 -- asq&MAX_VISITS ;

  found = 1 ;
  i = 1 ;
  number_of_tests = 0 ;

  do while ( found = 1 and i le &MAX_VISITS) ;

    if asq_scores [ i ] ne " " then number_of_tests + 1 ;
    else found = 0 ;

    i + 1 ;

  end;

  last_asq_date = asq_dates [ number_of_tests ] ;
  last_asq_results = asq_scores [ number_of_tests ] ;

  if number_of_tests > 1 then do;

    if asq_scores [ number_of_tests ] = 'Fail' and asq_scores [ number_of_tests - 1 ]
    = 'Pass' then pass_then_fail = 1 ;

  end;

  format last_asq_date mmddyy10.;
  drop i ;

run;
```

Using the array in the above data step we can conveniently group the visit dates and ASQ scores for processing. If your variables are sequentially numbered you can easily specify those as the elements of the array using the VAR1 – VARn syntax where n is the last variable. We know the last VISITn and ASQn variables by the using &MAX_VISITS macro variable we calculated above.

Once we have the arrays established we can examine each patient's visits and their scores. This allows us to process visits and look for patterns in the scores. To provide summaries to the clinicians need to be able to count the number of scores and identify the last score. To do this we use the DO WHILE loop. DO WHILE loops are evaluated at the start of the loop. Thus, before we enter the loop we initialize some variables that will be used to keep track of conditions as we process the visits:

```
found = 1 ;
i = 1 ;
number_of_tests = 0 ;
```

The FOUND variable will be set to 1 every time we find an ASQ score for a patient. The I variable will serve as an index to keep track of where we are in the ARRAY. As we walk through the ARRAY of visits and scores we will be able to reference an element of the ARRAY using this index. For example, asq_scores [i]. NUMBER_OF_TESTS will be a counter for the number of the ASQ scores for each person. We will walk through each patient's visits while we keep finding ASQ scores (found=1) and we haven't reached the maximum number of visits (i le &MAX_VISITS) using the following DO WHILE loop:

```
do while ( found = 1 and i le &MAX_VISITS ) ;
```

As we walk through each patient's visits we will count each time an ASQ score is found (if asq_scores [i] ne " " then number_of_tests + 1). If we don't find a score then know we have reached the last visit and we can tell the loop to stop (else found = 0).

Once we finish processing the loop we record the last visit date and ASQ score. We know the last index number for a person because we can have been keeping track of it in the loop. Thus, we use the simple assignment statement:

```
last_asq_date = asq_dates [ number_of_tests ] ;
last_asq_results = asq_scores [ number_of_tests ] ;
```

One final step is to set a flag if the last visit was a 'Fail' but it was preceded by a 'Pass'. Clinicians will want to follow-up on these patients. To do this we have the simple IF/THEN block.

```
if number_of_tests > 1 then do;
    if asq_scores [ number_of_tests ] = 'Fail' and asq_scores [ number_of_tests - 1 ]
    = 'Pass' then pass_then_fail = 1 ;
end;
```

DELIVERING THE RESULTS

Once we have processed the data we need to report deliver reports to the clinicians. We do this by using SAS® to create a Microsoft® Excel file. Using SAS® ODS and PROC REPORT along with the some additional macro variables we create a simple report that will summarize the visits for the clinicians and alert them to any patients that passed the ASQ test on their previous visit but failed on their most recent visit.

First we will create a two new SAS® MACRO variables to contain the report date in different formats.

```
%LET TITLE_DATE = %sysfunc ( putN ( &LAST_REPORT_DATE, worddate20. ) ) ;
%LET FILE_DATE = %sysfunc ( putN ( &LAST_REPORT_DATE, yymmddn8. ) ) ;
```

Next we will use PROC REPORT wrapped in SAS® ODS to generate an excel file. Note that we use the MSOFFICE2K_X tagset to generate the Microsoft® Excel as well as one the standard SAS® ODS styles – Festival. The code is as follows:

```
ods tagsets.msoffice2k_x style=styles.festival
file="R:\MWSUG\Reports\PPC_REPORT_&FILE_DATE..xls";

proc report data = PPC_REPORT nowindows;

    column pass_then_fail MRN last_asq_date last_asq_results ( "Visit Dates" visit1
-- Visit&MAX_VISITS ) ( "ASQ Scores" ASQ1--ASQ&MAX_VISITS );

    define pass_then_fail / noprint ;

    define MRN / display 'MRN' ;

    define last_asq_date / display 'Date of Last ASQ Score' ;

    define last_asq_results / display 'Last ASQ Score' ;

    compute MRN ;

        if _c1_ = 1 then do;

            call define('_C2_', 'style', 'style={background=RED}');

        end;

    endcomp ;

    title "ASQ Report for &TITLE_DATE";

run;

ods tagsets.msoffice2k close;
```

Because we need to alert the clinicians when a patient has failed their latest ASQ test, but passed the previous test we use the COMPUTE block to conditionally set the background color of the MRN based on the value of the PASS_THEN_FAIL column. As follows:

```
compute MRN ;

    if _c1_ = 1 then do;

        call define('_C2_', 'style', 'style={background=RED}');

    end;

endcomp ;
```

The resulting report looks like this:

	A	B	C	D	E	F	G	H	I	J	K	L	M
1	ASQ Report for August 13, 2013												
2													
3						Visit Dates					ASQ Scores		
4	MRN	Date of Last ASQ Score	Last ASQ Score	visit1	visit2	visit3	visit4	visit5	asq1	asq2	asq3	asq4	asq5
5	11111111	5/24/2012	Fail	24-May-12	Fail
6	11111112	3/11/2013	Fail	25-Apr-12	11-Mar-13	.	.	.	Pass	Fail	.	.	.

Now that we have used SAS® to create a nicely formatted report we can have SAS® deliver that report. SAS® allows you to email directly from the SAS® session and attach documents to the email. So again we have SAS do the work for us:

```
options EMAILSYS=SMTP EMAILHOST=mailx.cchmc.org;

filename OUTBOX EMAIL ;

data _null_;

  file outbox

  subject="ASQ Results for &TITLE_DATE"

  to = ( "Alan.Leach@cchmc.org"    "Gowri.Madhavan@cchmc.org"      )

  attach = "R:\MWSUG\Reports\PPC_REPORT_&FILE_DATE..xls"

  ;
  put "Please Review the Attached ASQ Test Results for &TITLE_DATE";

run;

filename outbox;
```

NOTE: The email settings at your location will differ from ours. You will need to check with your system administrators to determine what should be the proper values for EMAILSYS and EMAILHOST at your location.

CONCLUSION

The advantages of using SAS® for developing programs for the health sector and clinical data include speed, efficiency, minimizing error rates and help to establish repeated run of reports that can be automated. The reports can deliver consistent, trusted and verifiable clinical information. SAS® has the capability of accessing and integrating patient data from virtually any source in any format and then transform, cleanse and standardize the data so that it is ready for analysis. Furthermore, using some simple SAS® MACRO variables and some ARRAYS we were able to minimize the manual intervention needed to do the work. We have applied these SAS® techniques to other population based initiatives, such as asthma care coordination, health outcomes and patient safety. Replacing excel driven data manipulations with SAS® reports has enhanced and improved accuracy in the reports. It has also improved performance, measurement and tracking more effectively and efficiently.

REFERENCES

- Identifying infants and young children with developmental disorders in the medical home: an algorithm for developmental surveillance and screening. 2006. Pediatrics
- SAS Support Note: "Sample 24820: Creating a Directory Listing Using SAS for Windows". <http://support.sas.com/kb/24/820.html>

- Hemedinger, Chris. "Using the X and SYSTASK commands from SAS Enterprise Guide". <http://blogs.sas.com/content/sasdummys/2009/11/19/using-the-x-and-systask-commands-from-sas-enterprise-guide/>
- Choate, Paul A., Martell, Carol A, "De-Mystifying the SAS® LIBNAME Engine in Microsoft Excel: A Practical Guide" Paper 024-31, SUGI 31

ACKNOWLEDGMENTS

We thank Courtney Brown, MD, Instructor, Division of General and Community Pediatrics, CCHMC and Kevin Stanford, Biostatistician, Anderson Center, CCHMC for their content knowledge in this paper.

RECOMMENDED READING

- Identifying infants and young children with developmental disorders in the medical home: an algorithm for developmental surveillance and screening. 2006. Pediatrics
- <http://agesandstages.com/>

CONTACT INFORMATION

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

Name: Gowri Madhavan
Enterprise: Cincinnati Children's Hospital and
Medical Center
Address: 3333 Burnet Avenue, MLC 5040
City, State ZIP: Cincinnati, OH 45229
Work Phone: 513-803-1847
E-mail: gowri.madhavan@cchmc.org

Name: Alan Leach
Enterprise: Cincinnati Children's Hospital and
Medical Center
Address: 3333 Burnet Avenue, MLC 5040
City, State ZIP: Cincinnati, OH 45229
Work Phone: 513-470-8862
E-mail: alan.leach@cchmc.org

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.