

Paper 166-31

Identifying an Event within a Sequence of Events

Marge Scerbo, CHPDM/UMBC

Abstract

There are analyses that require the identification of an event within a series of events. Where does one event fall in the sequence? This poster will step through a process that was used in the study of teen pregnancies. A study was undertaken to trend teen pregnancies over a period of several years. Additionally, the analysis determined if the use of contraceptives occurred before, after, or around the delivery. The poster will include DATA step code including RETAIN, FIRST.var and LAST.var.

Introduction

The project described in this paper studied teenage girls to create percentages of those who had babies and those who had taken oral contraceptives. In addition, calculations included whether those teenage mothers had taken contraceptives before delivery, after delivery, or both. The goal was to show a trend towards decreasing teenage pregnancy and possible causes. To accomplish this goal, SAS was used to identify a series of populations from different datasets:

- Female teens from the eligibility files
- Teens who had delivered babies from the inpatient files
- Teens who had oral contraceptive prescriptions from the pharmacy files

Once each of these subsets was identified, one record per teen was stored and studied. This is the core of the paper and the poster.

The task at hand may seem easy at first glance, but as always, the data lends complexity to the project. As with any other study, it is imperative to understand the layout and quirks of the datasets to be used.

Manipulation of the data in SAS datasets is often needed before further analysis can take place. BASE SAS includes code that, if used properly, can make this process fairly easy. SAS DATA step code is large and varied, and so it is easy to miss some of the very useful statements, options, and temporary variables that are available. This project with its inherent complicated datasets was simplified by knowing how and when to use the correct SAS code.

Data Definitions

For each year that is to be analyzed, there are three separate SAS datasets that are used. These three datasets are Eligibility, Inpatient, and Pharmacy. Each of these files is quite large and analysis should be approached with efficiency in mind. Note the sizes of the three 2004 files:

File	Observations	Variables	Variables Required for Project
Eligibility	758,190	160	idcode, gender, age
Inpatient	39,923	131	idcode, primary_dx, enddate
Pharmacy	2,512,982	62	idcode, therapclass, rxdate

The code used includes WHERE, IF, and KEEP statements or options to limit the size of the datasets to be studied. Efficiency is important with the sizes of the files.

Let's take a minute to further discuss each of these input datasets:

- The eligibility files contain one record per recipient for the designated year.
 - Each recipient is identified by an id code (*idcode*) that is used to match across datasets.
 - Demographic data is attached to each enrollee.
 - With only one record per enrollee making selections from this file is fairly straightforward.
- The inpatient files contain claims for each inpatient stay.
 - There are multiple diagnoses (up to 12) for each claim. The project team elected to query only the primary diagnosis (*primary_dx*) to search for the desired codes.
- The pharmacy files are the largest files and contain one record per prescription. Therefore, in a year's time, one recipient can conceivably have up to 12 records for a single prescription drug.

- Each drug is defined by a National Drug Code, and each drug code is assigned an industry-standard therapeutic class (*therapclass*).
- For the purpose of this study, only one therapeutic class, 681200, will be selected.

Output Requested

The project request is short and appears easy to complete:

'Of the teenage (12-19) girls enrolled for each year, how many had babies? How many teenage girls took oral contraceptives? How many of those that had babies took contraceptives before and/or after giving birth?'

While this request appears quite straightforward, it does require in-depth knowledge of the datasets, how they are arranged, and how to use SAS code properly.

Process Flow

1. The first step in the process is to establish a population. This entails identifying all female enrollees between the ages of 12 and 19 from the eligibility file.
 - For efficiency, the SAS code stores only those fields needed in processing.
 - In SAS versions 8 and 9, IF and WHERE statements require similar computer resources, but using the WHERE statement allows use of the BETWEEN – AND operators.
2. In the next step, PROC FORMAT is used to create a user-defined format for the various diagnosis codes that identify a delivery.
3. The following step uses one DATA step to accomplish multiple tasks:
 - Using the recipient id code, the 'population' list containing teenage girls is merged with the inpatient file. This inpatient data is subset to include only those claims that contain one of the diagnoses codes identifying deliveries. This is accomplished using a WHERE statement with a PUT function and the user defined format.
 - Only recipients who occur in both files (teens with babies) are kept.
 - A flag is set to identify this group (*baby = 1*).
 - The date of service (*enddate*) is renamed as the date of birth (*babydob*) so in later steps it is clear that this date signifies the baby's date of birth.
4. Identifying teenagers from the population who have received prescriptions for oral contraceptives is accomplished in the next portion of the code.
 - The population file and the pharmacy file are merged by recipient id code.
 - Only those claims that have the correct therapeutic class are saved.
 - Only those recipients who occur in both files are stored.
 - Only the fields needed are used.
 - NOTE – it is possible that there are multiple claim lines for one person!
5. Pharmacy claims are built into one record per person.
 - The drugs dataset created in the last step is read.
 - A BY statement with key variables *idcode* and *rxdate* is included in the DATA step.
 - If the observation is the only claim for this id code, both the first date of prescription (*firstrx*) and last date of prescription (*lastrx*) are set to equal to the *rxdate*, and this updated observation is output.
 - If this record is the first record and not the last record, the first date of prescription (*firstrx*) is equal to the *rxdate*.
 - This date must be RETAINED or it will be overwritten when the next claim line is read.
 - This record is not output as it is not complete. The last prescription date has not been created.
 - When the last record for that id code is encountered, the last date of prescription (*lastrx*) is set equal to the *rxdate*, and the completed record is output.
6. The next step merges the final three datasets by id code resulting in one record per person.
 - A new variable called *teen* is set to 1. This allows a numeric variable for calculations in the next procedure.
 - This step will also determine how a person should be categorized:

Category	Baby	Drug	Baby DOB	First Rx	Last Rx
No Baby, No Contraceptives	0	0	-	-	-
No Baby, Contraceptives	0	1	-	-	-

Baby, No Contraceptives	1	0	Birth Date	-	-
Baby, Contraceptives Before & After	1	1	Birth Date	Before DOB	After DOB
Baby, Contraceptives Before	1	1	Birth Date	Before DOB	Before DOB
Baby, Contraceptives After	1	1	Birth Date	After DOB	After DOB

7. The final two steps prepare the results for easier interpretation.
- PROC MEANS is used to count the number of recipients in each category and create a summary dataset (*result04*). The top level of this output (`_TYPE_ = 0`) is equal to the total count of recipients and will be used as the denominator in the process.
 - A DATA step reads this output dataset and calculates percentages of each category of the entire population.

Data Transformation

The paper specifically mentions the use of the DATA step code RETAIN, FIRST.var and LAST.var. This section discusses the whys and how's of their use.

First, in order to use FIRST.var and/or LAST.var, the SET or MERGE statement must be followed by a BY statement. To use a BY statement, the dataset(s) must be sorted by the key variables identified. The BY statement will automatically create the SAS variables FIRST. and LAST. for each BY variable. If the first occurrence of the BY variable occurs, `FIRST.byvariable` is set to 1. So this code:

```
data drugsall04 (drop = rxdate);
  set drugs04;
  by idcode rxdate;
```

Causes these actions in the processing:

idcode	rxdate	first.idcode	last.idcode
00001	1/24/2004	1	1
00004	5/8/2004	1	0
00004	10/19/2004	0	1
00100	3/3/2004	1	0
00100	5/16/2004	0	0
00100	7/22/2004	0	0
00100	11/26/2004	0	1

The next section of code:

```
retain firstrx;
if first.idcode and last.idcode then do;
  firstrx = rxdate;
  lastrx = rxdate;
  output;
end;
else if first.idcode then firstrx = rxdate;
else if last.idcode then do;
  lastrx = rxdate;
  output;
end;
run;
```

Will cause the following dataset modifications:

idcode	rxdate	first.idcode	last.idcode	firstrx	lastrx	Output?
00001	1/24/2004	1	1	1/24/2004	1/24/2004	yes
00004	5/8/2004	1	0	5/8/2004	.	no

00004	10/19/2004	0	1	5/8/2004	10/19/2004	yes
00100	3/3/2004	1	0	3/3/2004	.	no
00100	5/16/2004	0	0	3/3/2004	.	no
00100	7/22/2004	0	0	3/3/2004	.	no
00100	11/26/2004	0	1	3/3/2004	11/26/2004	yes

Why is a RETAIN statement used? When SAS reads a new observation, all values from the previous record are overwritten, whether or not the new value is missing. So, in order to carry the first prescription date (*firstrx*) forward until the last observation for the recipient is encountered and output, RETAIN is used to save that date value.

If the RETAIN statement is omitted, the final dataset will have missing values for the *firstrx* variable:

idcode	rxdate	first.idcode	last.idcode	firstrx	lastrx	Output?
00001	1/24/2004	1	1	1/24/2004	1/24/2004	yes
00004	5/8/2004	1	0	5/8/2004	.	no
00004	10/19/2004	0	1	.	10/19/2004	yes
00100	3/3/2004	1	0	3/3/2004	.	no
00100	5/16/2004	0	0	.	.	no
00100	7/22/2004	0	0	.	.	no
00100	11/26/2004	0	1	.	11/26/2004	yes

Results

This process will create a dataset for each year studied. These datasets will contain counts and percentages, all extremely useful in analysis of health care data. Here is an example of the results for one year:

Final	Count	Percentage
Baby, Contraceptives After	993	1.2%
Baby, Contraceptives Before	45	0.0005%
Baby, Contraceptives Before & After	27	0.0003%
Baby, No Contraceptives	2,106	2.7%
No Baby, Contraceptives	7,031	9.0%
No Baby, No Contraceptives	67,468	86.8%

This information can then be compared across years to assist in policy decisions.

Conclusion

SAS provides many tools to create datasets that meet the needs of a project. Learn to use the DATA step and its statements and special temporary variables. Test new code on subsets of the data rather than on large datasets. Ask questions of other programmers. Access SAS online help and check the proceedings of SAS User conferences.

Contact Information

For more information contact:

Marge Scerbo
 CHPDM/UMBC, 1000 Hilltop Circle
 Sondheim Building, Room 309, Baltimore, MD 21250
 Email: scerbo@chpdm.umbc.edu

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.

SAS Code

```

libname elig "/data/mmis2/fiscal04/elig";

*create population of teenage girls;
data teens04;
    set elig.elig04 (keep = idcode age gender);
    where age between 12 and 19 and and gender = 'F';
run;

*format defining delivery diagnoses;
proc format;
    value $matdiag '650'-'66999','V30'-'V3999', 'V27'-'V2799' = 'ok';
run;

libname claims "/data/fiscal04/claims";

*select teenage girls who had babies;
data babies04;
    merge teens04 (in = in1)
          claims.inpt04 (in = in2 where = (put(primary_dx,$matdiag.) = 'ok'));
    by idcode;
    if in1 and in2;

    baby = 1;
    babydob = enddate;
    format babydob mmdyy10.;
    keep idcode babydob baby;
run;

*select teenagers who have had oral contraceptive prescriptions;
data drugs04;
    merge teens04 (in = in1)
          claims.pharm04 (in = in2
                          keep = idcode therapclass rxdate
                          where = (therapclass = '681200'));
    by idcode;
    if in1 and in2;

    drug = 1;
    keep idcode rxdate drug;
run;

data drugsall04 (drop = rxdate);
    set drugs04;
    by idcode rxdate;
    retain firstrx;
    format firstrx lastrx mmdyy10.;

    if first.idcode and last.idcode then do;
        firstrx = rxdate;
        lastrx = rxdate;
        output;
    end;

```

```

        else if first.idcode then firstrx = rxdate;
        else if last.idcode then do;
            lastrx = rxdate;
            output;
        end;
run;

*merge three analytic files, all one record per person;
data allteens04;
    merge babies04
           drugsall04
           teens04;
    by idcode;

    teen = 1;

    length final $36;
    if baby = . and drug = . then final = 'No Baby, No Contraceptives';
    else if baby = . and drug = 1 then final = 'No Baby, Contraceptives';
    else if baby = 1 and drug = . then final = 'Baby, No Contraceptives';
    else if baby = 1 and drug = 1 then do;
        if firstrx lt babydob and lastrx gt babydob then
            final = 'Baby, Contraceptives Before & After';
        else if firstrx gt babydob and lastrx gt babydob then
            final = 'Baby, Contraceptives After';
        else if firstrx lt babydob and lastrx lt babydob then
            final = 'Baby, Contraceptives Before';
    end;
run;

*summarize the results;
proc means data = allteens04 noprint;
    var teen;
    class final;
    output out = result04 n = count;
run;

*format the results for final report.
data result04 (keep = final count totalcount percent);
    set result04;

    retain totalcount;
    if _type_ = 0 then totalcount = count;
    else do;
        percent = count/totalcount;
        output;
    end;
run;

```