Fabian Depry, IMS
NAACCR 2018
Pittsburgh, PA

NAACCR XML & SAS®

Table of Contents

SAS Refresher

- Statistical Analysis System.
- Developed at North Carolina State University in 1966.
- First release in 1972.
- Owned and maintained by SAS Institute Inc. since 1976.
- License-based, closed ecosystem.
- Using a SAS program to process NAACCR XML requires changes to the DATA step (PROC step is not involved).

Understanding the Challenges

- Two common use cases for using SAS:
  - Read data and process a small number of variables.
  - Read data, apply recodes and write out the data.
- Those two use cases might require different solutions.

- NAACCR XML introduces two different types of issues:
  - Data model incompatibilities
  - Processing speed

Understanding the SAS Data Model

- SAS requires the data to be organized into data sets.
- A data set is a table of variables and observations.
- There is no relationship possible between two data sets.

VARIABLES

OBSERVATIONS

Registry	Patient	Site
0000000000	00000001	C509
0000000000	00000002	C619
0000000000	00000002	C447

Data Model: SAS vs NAACCR Fixed Columns

- SAS data sets map to NAACCR Fixed Columns Data Files:

OBSERVATIONS

VARIABLES

Registry	Patient	Site
0000000000	00000001	C509
0000000000	00000002	C619
0000000000	00000002	C447

LINES

START/END

Registry	Patient ID	Site
0000000000	00000001	C509
0000000000	00000002	C619
0000000000	00000002	C447

Data Model: SAS vs NAACCR XML

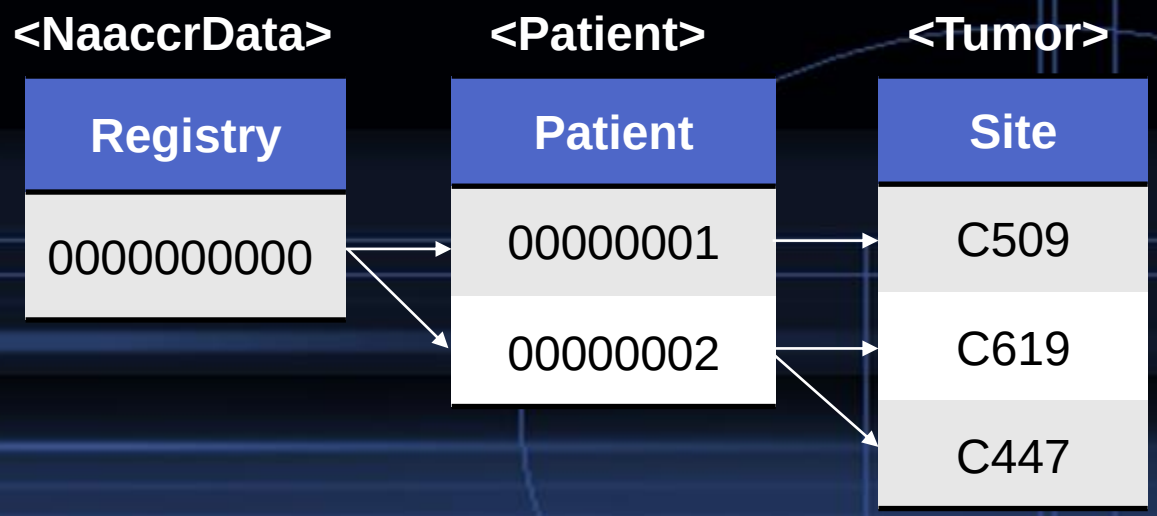
- SAS data sets map to ?

```
<NaaccrData>
  <Item naaccrId="registryId">0000000000</Item>
  <Patient>
    <Item naaccrId="patientIdNumber">00000001</Item>
    <Tumor>
      <Item naaccrId="primarySite">C509</Item>
    </Tumor>
  </Patient>
  ...
</NaaccrData>
```

VARIABLES

OBSERVATIONS

Registry	Patient	Site
0000000000	00000001	C509
0000000000	00000002	C619
0000000000	00000002	C447



Reading XML using the XML Mapper

- Description file that tells SAS how to map an XML tag to a data set, observation or variable.
- Mapping supports a single level of data.
  - The three levels need to be MERGED to be used in SAS.
- Every item defined in the mapping is read as a variable.
  - Can result in slow processing for many variables.
  - Defining fewer variables in the mapping will speed things up.
  - The NAACCR XML Utility tool can create the mappings on the fly based on the user selecting some variables.

XML Mapper Definition File

```
<!-- ############################################################ -->
<!-- SAS XML Libname Engine Map -->
<!-- Generated by NAACCR XML Java library v4.9-SNAPSHOT -->
<!-- ############################################################ -->
<SXLEMAP description="NAACCR XML v180 mapping" name="naaccr_xml_map_180" version="2.1">

    <!-- ############################################################ -->
    <TABLE description="Patients data set" name="patients">
        <TABLE-PATH syntax="XPath">/NaaccrData/Patient</TABLE-PATH>

        <COLUMN class="ORDINAL" name="PatientKey" retain="YES">
            <INCREMENT-PATH beginend="BEGIN" syntax="XPath">/NaaccrData/Patient</INCREMENT-PATH>
            <TYPE>numeric</TYPE>
            <DATATYPE>integer</DATATYPE>
        </COLUMN>

        <COLUMN name="patientIdNumber">
            <PATH syntax="XPath">/NaaccrData/Patient/Item[@naaccrId="patientIdNumber"]</PATH>
            <DESCRIPTION>Patient ID Number [Item #20]</DESCRIPTION>
            <TYPE>character</TYPE>
            <DATATYPE>string</DATATYPE>
            <LENGTH>8</LENGTH>
        </COLUMN>
```

Using the XML Mapper in SAS

```
1  filename testdata 'synthetic-data_naaccr-180-incidence_10-tumors.xml';
2
3  filename xmldef 'naaccr-xml-sas-def-180-incidence.map';
4  libname testdata XMLV2 xmlmap=xmldef access=READONLY;
5
6  data naaccrdata;
7    set testdata.naaccrdata;
8
9  data patients;
10   set testdata.patients;
11
12 data tumors;
13   set testdata.tumors;
14
15 data alldata;
16   merge naaccrdata patients;
17     by naaccrdatakey;
18
19 data alldata;
20   merge alldata tumors;
21     by patientkey;
22
23 proc freq;
24     tables primarySite;
25 run;
```

1

2

XML Mapper Performance

	5 variables		50 variables		All incidence variables	
	N18 Flat	**N18 XML**	**N18 Flat**	**N18 XML**	**N18 Flat**	**N18 XML**
1K Tumors	< 1 sec	< 1 sec	< 1 sec	< 1 sec	< 1 sec	5 sec
10K Tumors	< 1 sec	6 sec	< 1 sec	8 sec	< 1 sec	55 sec
100K Tumors	2 sec	1 min	2 sec	2 min	2 sec	9 min
1M Tumors	17 sec	11 min	17 sec	12 min	21 sec	1 hour
10M Tumors	3 m	1 h	3 min	2 hours	4 min	15 hours

Fun facts:

- NAACCR XML Utility Tool took 30 minutes to convert the 10M Tumors Flat file into XML.
- Resulting file was 1.7GB compressed and had 1,078,034,056 lines!
- Reading that file “patient by patient” took 15 minutes but “line by line” took 3 minutes

Writing XML using a NAACCR XML Tagset

- Set of instructions that tell SAS how to reformat a data set.
- Event-driven, very basic syntax.
  - Event-driven means powerful but inefficient.
  - Basic means simple but limited.
- Instructions can be packaged into a template file.
- Current template requires the data set to contain all variables, including special level keys.
  - Goes hand-in-hand with the XML Mapper.
- Current template doesn’t support user-defined dictionaries.

XML Tagset Template File

```
11      define event doc;
12      start:
13          put "<?xml version=""1.0""?>" nl nl;
14          put "<NaaccrData baseDictionaryUri=""http://naaccr.org/naaccrxml/naaccr-dictionary-";
15          put $options['naaccr_version'];
16          put ".xml"" recordType=""";
17          put $options['record_type'];
18          put """ specificationVersion=""1.3"" xmlns=""http://naaccr.org/naaccrxml"">" nl;
19          eval $data_written 0;
20          eval $pat_written 0;
21          eval $in_data 0;
22          eval $in_pat 0;
23          eval $in_tum 0;
24          eval $cur_pat 0;
25          set $prev_pat '0';
26          ndent;
27          break;
28      finish:
29          trigger endPatient /if $in_tum = 1;
30          xdent;
31          put "</NaaccrData>";
32          break;
33      end;
```

Using the XML Tagset in SAS

```
1  filename inxml 'synthetic-data_naaccr-180-incidence_10-tumors.xml';
2  filename outxml 'synthetic-data_naaccr-180-incidence_10-tumors_copy.xml';
3
4  filename xmldef 'naaccr-xml-sas-def-180-incidence.map';
5  libname inxml XMLV2 xmlmap=xmldef access=READONLY;
6  data naaccrdata;
7    set inxml.naaccrdata;
8  data patients;
9    set inxml.patients;
10 data tumors;
11   set inxml.tumors;
12 data patients;
13   merge naaccrdata patients;
14     by naaccrdatakey;
15 data xmldata;
16   merge patients tumors;
17     by patientkey;
18
19 ods listing close;
20 ods path(prepend) work.templat(update);
21
22 %include "naaccr-xml-sas-tags.tpl";
23
24 ods markup tagset=naaccrxml file=outxml options(naaccr_version='180' record_type='I');
25 proc print data=xmldata;
26 run;
27 ods markup close;
```

1

2

3

XML Tagset Performance

	N18 Incidence Flat	N18 Incidence XML
1K Tumors	< 1 sec	35 sec
10K Tumors	< 1 sec	6 min
100K Tumors	7 sec	1 hour
1M Tumors	1 min	10 hours
10M Tumors	12 min	days???

Other solutions: Calling Java from SAS

- Limited support for calling Java from a SAS program was added in version 9.2.
- Requires a Java Archive (JAR) library.
- Calls to the library can be intimidating for non Java users but they can be simplified by using SAS macros.
- Those macros can be distributed with the JAR library for the benefit of the entire NAACCR community.

Calling Java from SAS

```
/** After calling this macro, the data will be available in a "alldata" dataset */
%include "read_naaccr_xml.sas";
%readNaaccrXml(
  libpath="naaccr-xml-4.10-SNAPSHOT-sas.jar",
  srcfile="synthetic-data_naaccr-180-incidence_100-tumors.xml.gz",
  naaccrversion="180",
  recordtype="I"
);

proc freq;
    tables primarySite;
run;

/** Prior to calling this macro, the data needs to be available in a "alldata" dataset */
%include "write_naaccr_xml.sas";
%writeNaaccrXml(
  libpath="naaccr-xml-4.10-SNAPSHOT-sas.jar",
  targetfile=synthetic-data_naaccr-180-incidence_100-tumors-copy.xml.gz",
  naaccrversion="180",
  recordtype="I"
);
```

1

2

Calling Java from SAS Performance

	Read data			Read & Write data		
	N18 Flat	N18 XML (Mapper)	N18 XML (Java)	N18 Flat	N18 XML (Mapper + Tagset)	N18 XML (Java)
1K Tumors	< 1 sec	5 sec	9 sec	< 1 sec	35 sec	10 sec
10K Tumors	< 1 sec	55 sec	55 sec	< 1 sec	6 min	55 sec
100K Tumors	2 sec	9 min	3 min	7 sec	1 hour	3 min
1M Tumors	21 sec	1 hour	5 min	1 min	10 hours	8 min
10M Tumors	4 min	15 hours	18 min	12 min	days???	1 hour

Other solutions: Using an External Tool

- NAACCR provides a standalone tool to transform an XML file into a Fixed-column file.
  - Good alternative until NAACCR Fixed-column format is retired (scheduled for 2020).
- After that, other tools might be created to flatten XML into a non-NAACCR SAS-friendly format.
- Non-SAS solutions should also be considered.
  - Data can be written to a database and SQL used for its analysis.
  - Other tools like the SEER Data Viewer can be used for recoding.

Conclusions

- Native XML support in SAS is limited.
- XML Mapper and Tagset work but are slow.
- Other solutions exist but might not be as convenient.
- Hopefully more solutions will be developed in the future.
  - The community needs to get involved.
  - https://www.naaccr.org/forums/topic/using-sas-with-naaccr-xml/
- In the meantime, the NAACCR XML Utility Tool can be used to convert XML into fixed-column.

Thank you!!!

Special thanks:

- Jennifer Stevens (IMS)
- Rick Firth (IMS)
- Linda Coyle (IMS)
- Isaac Hands (KCR)
- Angela Eckstrand (Alberta Health Services)
- NAACCR XML forum participants

All SAS programs and (small) data files available online:

- https://github.com/imsweb/naaccr-xml/wiki/