Using DDI Extensions as an Intermediary for Data Storage and Data Display

*by Patricia Cruse[*], Marsha Fanshier[*], Fredric Gey[†] and Margaret Low[*1]*

The Counting California project (http://countingcalifornia.cdlib.org)[1] makes available statistical information about the State of California, using data from California State and various federal agencies. The initial project design called for a metadata-driven system, based on the Data Documentation Initiative (DDI) document type definition (DTD). The goal is to provide consistency in *data discovery* and *data display* to the end user, regardless of the structure of the underlying data.

Proposed extensions to the DDI developed by Wendy Thomas of the University of Minnesota (http://www.socsci.umn.edu/~wlt/ddi) allow for the definition and documentation of statistical summary data in the form of multi-dimensional tables. The provision for these matrix variables allow for the creation of metadata**,** which not only characterizes the fundamental content of the statistical data, but also allows for flexibility in the choice of data storage structures and information display of the stored data. This paper describes how the Counting California project team utilized the proposed DDI extensions to implement a data system designed to be flexible in implementation and consistent in presentation.

Output Tables

Multi-dimensional data tables produced by the Bureau of the Census present certain challenges. Specifically, the relationship between the variables and the tables must be maintained. The proposed DDI extensions address this problem with a method for **defining output tables**. This definition is in XML and is referred to, in this paper, as the varMtx (variable matrix). While we found the varMtx to be extremely useful in working with the multi-dimensional tables, it could also be applied to other types of data files. In addition, we found that making the table our primary output object addressed multiple project goals.

Tables and Project Goals

By selecting the pre-defined table as our initial dynamic data display, we were able to address the following project_goals:

- Maintain consistent granularity of titles for *data discovery*
- Maintain consistent data display
- Control user interaction with the server for server-side *data delivery*
- Assign subjects and keywords at a logical and useful level

In addition, the use of the varMtx to define dynamic output tables provided the ability to treat online data objects with different properties in a consistent fashion. We gained:

- A structured metadata record for searching
- Harmonized representations of data

- Powerful and flexible metadata available to the data delivery system

Granularity of Titles
A primary goal of Counting California is to provide searching of metadata for the purpose of data discovery. Variable-level searching of both standard rectangular files and multi-dimensional files presented a granularity problem.

A standard rectangular file might have one variable (column) for race with five values:

Ex. 1: One Variable
Race
White
African American
Native American
Asian
Hispanic

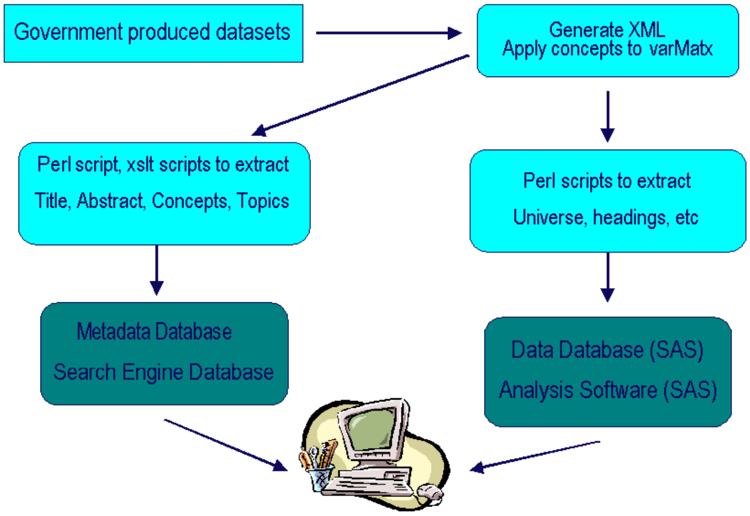
The multi-dimensional data file would have five columns:

Ex. 2: Five Columns				
Race- White	Race-African American	Race-Native American	Race- Asian	Race- Hispanic

A search of variables on 'race' would present one hit in example 1 and five hits in example 2**.** A cross-tab of Race by Sex would have ten columns for race and present ten hits. Utilizing the varMtx definition for both file types gave us the means to harmonize our definitions and present consistent discovery behavior to our users. By implementing discovery at the table level, a search on 'race' would yield both types of tables, but return one hit (***table title***) for each table. **Data discovery is done at the table level, not the variable level.**

XML and XML Tools
Since the metadata is in XML format, it provided an opportunity for the team to look at a variety of methods and tools for data discovery and display. The XML-tagged metadata was created in two different manners. For datasets with many tables (STF3, USA Counties), the XML code was generated using perl scripts and then manually edited using a commercial XML editor. For the smaller datasets, the XML code was manually created. The relationship between the tables and appropriate subjects and keywords was determined and then added into each of the varMtx XML tables. Once the XML-tagged metadata was generated for each study, perl scripts and XSL Transformations (XSLT) with style sheets were developed to transform the structure of the XML into the data needed for the data discovery, search and dynamic data display.

XML Tools for Data Discovery and Display

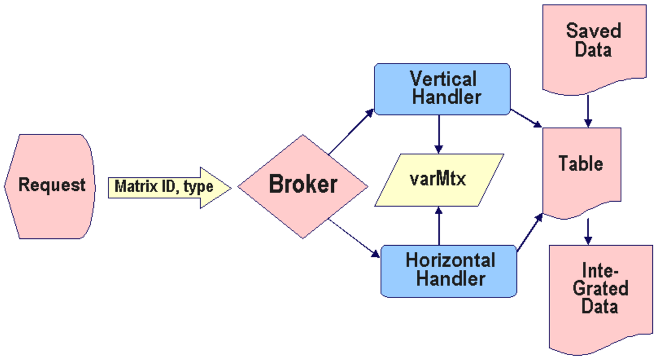


The metadata for a selected table, contained in a single, structured record in the metadata database, can be fed into multiple software packages, including an RDBMS database, text search engine and analysis software. Only the analysis software cares about the type and structure of the statistical data; all others rely only on the metadata.

Harmonized Representations of Data

While the varMtx provides a way of defining the multi-dimensional tables, it also can be used to define tables derived from standard rectangular files. One property of the multi-dimensional tables is that the aggregated data comes in short/fat files. Traditional rectangular files tend to be long and skinny.

Detailed**,** in the next pag**e,** is how we used the varMtx as a *middle-tier* to work with both types of data.



A data request is routed to either the vertical or the horizontal handler, depending on the type of data. Each type requires the same parameters. Minimal parameters are necessary since the analysis software runs its own lookup on table elements. Outputs created include a table and cached data. Further operations, such as graphing, subsetting and exporting, utilize the cached data for input. Results from both horizontal and vertical data are discussed below:

Horizontal *(Short/Fat)* Data Example

Areaname	H0460001	H0460002	H0460003	H0460004	H0460005	H0460006	H0460007	H0460008	H0460009	H0460010	H0460011	H0460012	H0460013	H0460014
Alameda	7413	9971	39964	76247	42186	17473	3454	567	956	6405	11121	5378	1831	378
Alpine	25	18	59	51	5	0	23	0	2	0	0	0	0	1

The data for the table ‘Hispanic Origin~~s~~ by Gross Rent’ comes from the 1990 STF3A and is pre-summarized for display. To reproduce this table, the system needs to maintain the pre-defined relationships between variables as well as complex header information.

	Not of Hispanic orgin							Hispanic orgin						
	With cash rent							With cash rent						
areaname	less than $200	$200 to 299	$300 to $400	$500 to $749	$750 to $999	$1,000 or more	no cash rent	less than $200	$200 to $299	$300 to $499	$500 to $749	$750 to $999	$1,000 or more	No cash rent
Alameda County	7,413	9,971	39,964	76,247	42,186	17,473	3,454	567	956	6,405	11,121	5,378	1,831	378
Alpine County	25	18	59	51	5	0	23	0	2	0	0	0	0	1

Hispanic origin by gross rent Specified renter-occupied housing units

This table has two dimensions (Hispanic Origin and Gross Rent). The varMtx specifies, in compact format, the dimensions of the table, labels as well as cell quantity and location information. *The full record for this example, is available at* http://countingcalifornia.cdlib.org/MoreInfo/ddi_ext_app.txt.

```
<varMtx ID=”H046" name=”H046" files=”stfh050 stfh060 stfh040 stfh155">
<labl level=”matrix” source=”producer”>HISPANIC ORIGIN BY GROSS RENT</labl>
```

```
<dmnsQnty>2</dmnsQnty>
<cellQnty>14</cellQnty>

<mtxdmns ID=”H046_mtx_1" sdatrefs=”ABS”>
<coord>1</coord>
<labl level=”mtxdmns” source=”producer”>HISPANIC ORIGIN</labl>
<cohQnty>2</cohQnty>

<mtxdmns ID=”H046_mtx_2">
<coord>2</coord>
<labl level=”mtxdmns” source=”producer”>GROSS RENT</labl>
<cohQnty>7</cohQnty>
```

The variable knows its location within the table. In this example, it would be the first cell in both dimensions (coordinate1=1, coordinate2=1).

```
<var name=’H0460001' varMtx=’H046' files=’’ >
  <coordVal coord=’1'>1</coordVal>
  <coordVal coord=’2'>1</coordVal>
  <labl>Less than $200</labl>
</var>
```

Properties

- Data is pre-summarized
- Data is layed out in horizontal, linear order

Functional Requirements

- Select Variables for Table
- Roll-up Headers
  - □ Cohorts repeat consistently (e.g., variable values)
  - □ Sub-Headers are attached to specific cohorts

Data Statements

The SQL command for extracting this data would be:

```
select H040001..H0460014
from STF3DATA
;
```

Counting California uses the SAS procedure **Proc Report** to roll-up headers.

Vertical *(Long/Skinny)* Data Example

name	native	sex	year	age
Alameda	208	MALE	1970	0- 4
Alameda	214	MALE	1970	5-9
Alameda	208	MALE	1970	10-14
Alameda	251	MALE	1970	15-19

Alameda	372	MALE	1970	20-24
Alameda	209	MALE	1970	25-29
Alameda	171	MALE	1970	30-34
Alameda	155	MALE	1970	35-39
Alameda	129	MALE	1970	40-44
Alameda	114	MALE	1970	45-49
Alameda	86	MALE	1970	50-54
Alameda	82	MALE	1970	55-59
Alameda	80	MALE	1970	60-64
Alameda	61	MALE	1970	65-69
Alameda	33	MALE	1970	70-74
Alameda	19	MALE	1970	75-79
Alameda	24	MALE	1970	80-84
Alameda	31	MALE	1970	85 and up

Vertical data appears to be more complex given that values must be applied to give meaning to the variables as well as the tabulations performed. It is actually a simpler process, since analysis software is designed to accommodate this format.

	Native American															
	Male															
	Ages															
	0-4	5-9	10-14	15-19	20-24	25-29	30-34	35-39	40-44	45-49	50-54	55-59	60-64	65-69	70-74	80 & up
Alameda	208	214	208	251	372	209	171	155	129	114	86	82	80	61	33	24

The varMtx for this type of data _uses SQL commands to subset the data. In the example below the universe is *Native American Males*. Limiting to *Native American* is a matter of variable selection since that summarization has already been completed. The data must be subset to limit to *males*. That is accomplished with an *SQL WHERE* statement. The parameters for the WHERE statement are coded into the **derivation** element.

```
<mtxdmns>
<coord>2</coord>
<labl level=”mtxdmns”>Sex</labl>
<cohQnty>1</cohQnty>


  <qstn var=”sex”>
  </qstn>
  <derivation>
  <drvdesc>Males only</drvdesc>
  <drvcmd> (sex=’MALE’)</drvcmd>
  </derivation>
```

Properties

- Data may or may not be pre-summarized
- Data is not cross-tabulated

Functional Requirements

- Select Variables for Table
- Subset Data
- Run Tabulations/Cross-Tabulations
- Apply Values to Variables
- Roll-up Headers

Data Statements

The SQL command for extracting this data would be:

```
select name, native, sex, year, age
from RACE1970
where sex=’Male’
;
```

Counting California uses the SAS procedure **Proc Tabulate** to accomplish functional requirements 3-5, listed above.

Generating Tables without Modifying Data or Programs

The new U.S. Census Public Law data has four tables that contain more than 70 cells each. An example of a label for one cell is:

> White; Black or African American; American Indian and American Native; Native Hawaiian and Other Pacific Islander; Some other race.

The display of one of these tables is 71 columns wide, an unfortunate size for printing.

In order to present a more useful and readable display to the end-user, a new representation of the data was created, selecting fewer data elements. The example below, RACE [14], shows an additional table that was created by modifying the metadata, but **without modifying either the data or the program**. This method provided the opportunity to display multiple tables, utilizing the basic 71 variables, via the metadata-driven program.

Race [14]Total population													
		Population of one race											
Areaname	Total	Total	White alone	Black or African American alone	American Indian and Alaska Native alone	Asian alone	Native Hawaiian and Other Pacific Islander alone	Some other race alone	Population of two races	Population of three races	Population of four races	Population of five races	Population of six races
Alameda county	1,443,741	1,362,517	704,334	215,598	9,146	295,218	9,142	129,079	74,604	5,938	557	123	2
Alpine county	1,208	1,147	890	7	228	4		17	58	3	0	0	0

Conclusion

The original DDI DTD, as developed by an international working group, was envisioned as an archival documentation standard for statistical microdata datasets. The idea was to encompass the bibliographic description level as well as details of file structure and layout. The DDI extensions developed at the University of Minnesota extend the variable definition section to allow the definition of the logical structure of multi-dimensional summary data often encountered in Census files. Our contribution has been to recognize that these extensions and the DDI in general can also be utilized as an active agent in controlling both data storage layout for statistical summary data and in providing flexible data display of multi-dimensional tables. By separating the logical framework of the data from the layouts, it is possible to interface to multiple storage formats and to prepare a metatable display capability without having to hard-code display structure or order into the report preparation code.

Other suggested and possible uses of the output table definition include:

- A means of communication between remote/disparate systems, i.e., web services
- A way of transporting data between analysis packages
- A dynamic method of creating time-series data, e.g., include variable/file definitions

Appendix

Please see http://countingcalifornia.cdlib.org/MoreInfo/ddi_ext_app.txt for varMtx examples.* Paper presented at the IASSIST/IFDO conference 2001 in Amsterdam.

Footnotes:

* California Digital Library, University of California.

[t] University of California, Berkeley

[1] The Counting California project is funded by the University of California's California Digital Library and the state-based Library of California. Additional funding for continuing development comes through a federal grant from the Library Services and Technology Act (LSTA), administered by the California State Library.