VIReC Corporate Data Warehouse Cyberseminar Series

Using Stata Tools to Access CDW

September 8, 2016

Elliott Lowy, PhD
VA Puget Sound Health Care System
University of Washington School of Public Health

Poll #1: I am interested in VA data primarily due to my role as _____.

- Research investigator
- Data manager
- Project coordinator
- Program specialist or analyst
- Other (specify)

Poll #2: What is your level of experience with CDW data?

- 1- Not worked with it at all
- 2
- 3
- 4
- 5- Very experienced with CDW data

Introduction

- This demonstration assumes that you are already familiar with Stata, SQL, and CDW.
- The commands are available at:
  - https://www.vapulse.net/groups/statabits
  - Stata command: ssc describe lowyseattle
- Download includes more (non-SQL related) commands, as well.
- The editor I'll use is not Stata's, but Stata's will work.
- Disclaimer: The commands were written to make me happy; your mileage may vary.

Overview of the demonstration

- The tasks I'll be reviewing, in order, are:
  - Setting the sql connection info (aka the sql path)
  - Using simple commands for simple tasks (eg, get, put)
  - Retrieving schema/table/column info
  - Running more complex sql scripts
- Docs are available via Stata's help command:
  - help lowy (table of contents)
  - help sql (index to most of the sql commands)
  - help: pair, ddt, codex (some sql-relevant commands)
- Please feel free to ask questions along the way.

sql path

- Sets and/or displays the connection to the remote database, including the default schema
- Is remembered going forward, including after relaunching Stata
- Other commands implicitly use this path.

sqli

- Directly executes sql code
  - Everything after the command itself is straight sql to execute:
  - sqli *any sql code you like*
- 'i' is for 'immediate' (as in Stata's *immediate* commands…)

sql put

- Puts current Stata data on the server
- Allows if/in & keep()
- Overwrites any existing table
- Creates any non-existent schema
- Overwrite & create are general features of all the commands

sql get

- Downloads a single table or part of a table
- where() & distinct options supply the relevant sql
- keep() specifies column names as a Stata varlist (ie, abbrevs & ranges)
- quick() is usually SQL top
  - But, for CDW fact tables, it uses data from the most recent, complete quarter.
- Reads metadata from the sql finish command

sql finish

- Saves everything from the dataset, except the data, to the remote server
  - ie: labels, formats, etc.

- If a 'finished' table is downloaded with sql get, it arrives with all labels, formats, etc.

sql move

- Gives a table a new name and/or schema
- Preserves any extra metadata

After any download

- ddt standardizes date and datetime fields in the dataset
  - date fields are converted/formatted as Stata dates
  - datetime fields are converted/formatted as Stata datetimes

- pair replaces a pair of fields (string & numeric) with a single, value-labeled, numeric field

sql tables

- A quick way to list tables in a schema or db.
- Table names are selected like a Stata *varlist*.

sql cols

- Lists columns in a table
- Column names are selected like a Stata *varlist*
- Produces two lists:
  - readable
  - copy-able
- Copy-able list can prefix each name with a table alias

sql clear

- A convenience for deleting a bunch of tables
- Eg, everything in an obsolete schema

sql dbdescription

- Produces a general catalog for a DB or schema:
  - For the DB: all schemas and tables
  - For each table: # of rows, size, mode date
  - For each column: name, type, size
- Schemas and tables can be expanded & collapsed for comparison.
- The description is saved in a local folder, for later review.
- Older descriptions are versioned.

sql do

- sql do executes sql script from a file, and the files and script include a bit of special formatting.
- The SQL can be included with, mixed in among, your stata code.
- The SQL code goes between these tags (each on their own line):

```
*---sql:
*---/sql
```

A necessary digression

- sql do executes a file saved on disk.
  - *Only the part between the sql tags.*
- It can execute the file that it is part of.
  - *The file must be saved to disk before execution.*
- Leaving the file name blank only works from an external editor.

SQL between the tags

- Mostly a 'common table expression' or CTE.
- It's a sequence of table definitions.
  - Each table definition starts with this special line:
    - ^t=*name*
  - Each table definition ends with a blank line.

End result of the sql

- sql do, with no options, creates a table (in the sql path) with the final table *name* in the script. (ie, from ^t=*name*)
- Options on the command line can specify:
  - different name
  - different schema/name
  - download to stata

Local data can be uploaded as a temporary table

- ^t=*name*
  ^*use*=*filepath* [if] [in] [, keep()]

- Syntax like a stata use command

sql do produces a comprehensive summary

- Which tables will be created/run
- How references are resolved
- Final destination
- SQL code

- There is a ‘preview only’ option.

^t: table references

Table references are resolved at execution, in one of three prioritized ways:

- A name can be supplied on the command line.
- Otherwise, it will refer to the current execution, if possible.
- Otherwise, it will refer to the current sql path.

Running subsets of code

- Execution can be restricted to any internally consistent subset of the tables.
  - only(*table list*)
  - not(*table list*)
  - *table list* is specified like a stata *varlist* (wildcards and ranges)
- The final table – among those executed – is created.

More subsets

- sql sections can be labeled
  - *---sql:*label*
  - *--/sql
- Only a single label is ever executed
  - ordinarily, the blank one
  - physically separated chunks of code can use the same label
- Labels and tables can be specified in combination
  - sql do, only(part3:)
  - sql do, only(t1-t4)
  - sql do, only(part3:t5-t8)

Other 'macros'

- ^g: (For 'general purpose')
- If no swap is specified, the macro resolves to itself:
  - ^g:labname -> labname
- where date>= '^g:begin' and date<'^g:fin'
  - swap(^g:begin(2015/1/1) ^g:fin(2016/1/1))

sql dbd, again

- sql do creates extra meta-data:
  - script execution time
  - the actual sql that created the table
  - the local file where the script originated
  - possibly a query description
- This meta-data is displayed by sql dbd
- A download to Stata, followed by sql put, will preserve the metadata

Code can be imported from external files

^t=*name*

^ssc=*filepath*

- The contents of *filepath* are effectively inserted into the current script
- The contents of *filepath* become the table definition for ^t=*name*

Additional Resources

VIReC CDW Cyberseminars

http://www.hsrd.research.va.gov/cyberseminars/catalog-archive-virec.cfm?#Archived

VIReC CDW Documentation

http://vaww.virec.research.va.gov/CDW/Documentation.htm

VIReC Researcher's Notebooks

http://vaww.virec.research.va.gov/Notebook/Overview.htm

- Issue #4: How do I access SQL database files in VHA's Corporate Data Warehouse using SPSS?
  - http://vaww.virec.research.va.gov/Notebook/RNB/RNB4-CDW-Access-Using-SPSS-CY15.pdf

Questions?

Elliot.Lowy@va.gov