# Using ARQ to investigate RDF data cube

*mja@statgroup.dk*

*2016-01-16*

## Contents



## SPARQL scripts for the demographics cube (DC-DEMO-sample.ttl)

The examples below uses `arq` from Apache Jena (http://jena.apache.org). To install arq - download and unpack the latest version of apache-jena from (http://jena.apache.org/download/index.cgi). Then you need some way of invoking `arq`; I use a not-so-clever-approach: `cd ~/bin; ln -s /opt/apache-jena-2.13.0/bin/arq`.

Given a SPARQL query and RDF data, `arq` returns the result of the query. So this is the command line way of making a SPARQL query.

The use of arq is described in many places, see for example (http://www.learningsparql.com/).

All `arq` commands below are to be run in the directory with the sample files, which is `inst/extdata/CUBE-standards-rdf` directory or `extdata/CUBE-standards-rdf` depending on the whether the development version or the installed version of the package is used.

The cd below in each code block is included because I could not find a quick way to get the code chunk executed in that directory. knitr is flexible enough to do it, I have not yet found the right way to do it. So, ignore the repeated cd ..

For making the SPARQL queries I used a simple trick - copy the turtle definition from the cube.ttl, and do a replace regexp in emacs using pattern `+\([^:]+\):\([^ ]+\) .*$` and replacement `\1:\2 ?\2 ;`.

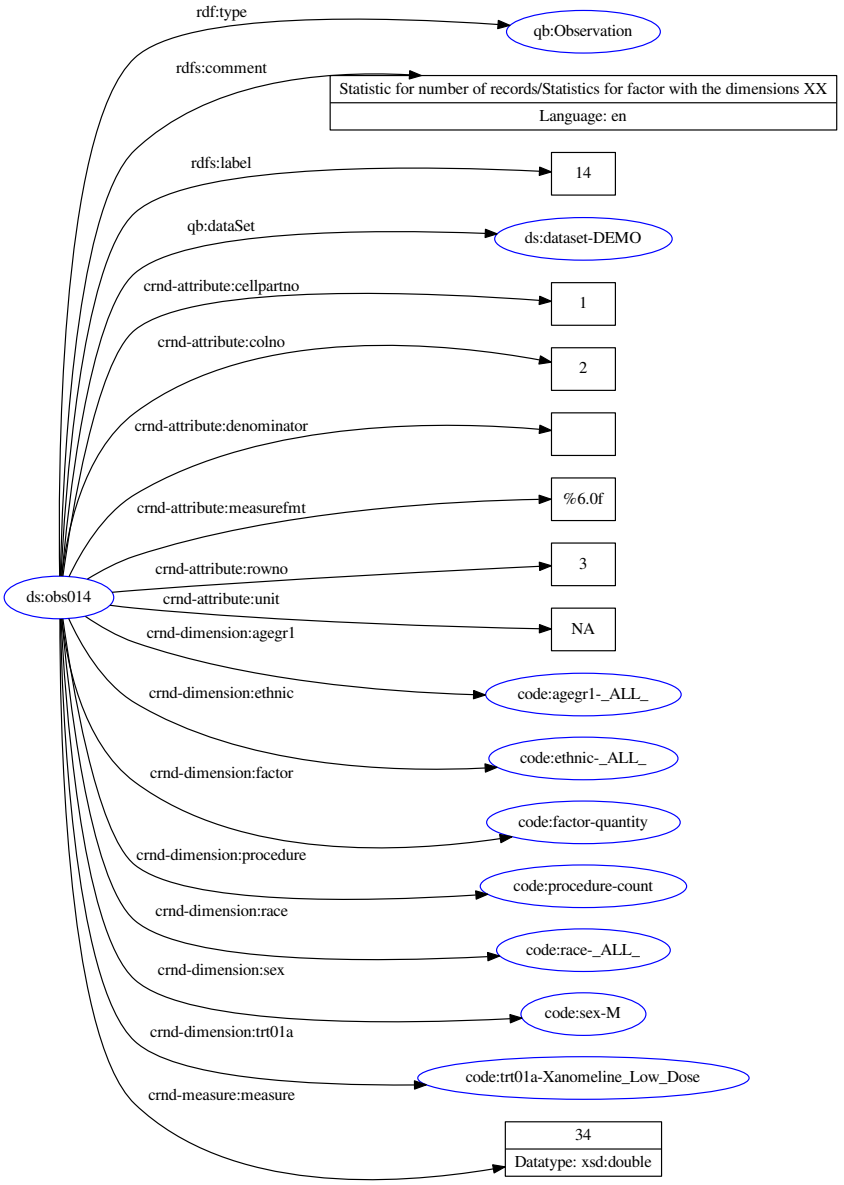
### Get all member of qb:ComponentProperty

```
cd ../extdata/sample-rdf
arq --results ttl --data ../../../../rrdfqb/inst/extdata/cube-vocabulary-rdf/cube.ttl --query qb-constru
rapper -i turtle -o dot fordot.ttl > fordot.dot
dot -x -Tpng -ograph.png fordot.dot
rm -f fordot.dot
```

```
## rapper: Parsing URI file:///home/ma/projects/R-projects/rrdfqbcrnd0/rrdfqbcrndex/inst/extdata/sample-
## rapper: Serializing with serializer dot
## rapper: Parsing returned 29 triples
```

ToDo(MJA): location for cube.ttl should be generated by the program - not using a directory reference

```
knitr::include_graphics("../extdata/sample-rdf/graph.png")
```




Model:
(Unknown)

Namespaces:
crnd-measure: http://www.example.org/dc/measure#
dccs: http://www.example.org/dc/demo/dccs/
code: http://www.example.org/dc/code/
cts: http://rdf.cdisc.org/ct/schema#
pav: http://purl.org/pav
owl: http://www.w3.org/2002/07/owl#
xsd: http://www.w3.org/2001/XMLSchema#
skos: http://www.w3.org/2004/02/skos/core#
rdfs: http://www.w3.org/2000/01/rdf-schema#
crnd-attribute: http://www.example.org/dc/attribute#
ds: http://www.example.org/dc/demo/ds/
qb: http://purl.org/linked-data/cube#
mms: http://rdf.cdisc.org/mms#
crnd-dimension: http://www.example.org/dc/dimension#
dct: http://purl.org/dc/terms/
rdf: http://www.w3.org/1999/02/22-rdf-syntax-ns#
rrdfqbcrnd0: http://www.example.org/rrdfqbcrnd0/
dcat: http://www.w3.org/ns/dcat#
prov: http://www.w3.org/ns/prov#

The file is needed for rendering, so no clean up.

```
cd ../extdata/sample-rdf
rm -f graph.png
```

## How to run this .Rmd file

.. add text ..