Distributed COVID-19 Knowledge Graph Practical Assignment

Tobias Käfer*

14th May 2021

Scenario

Imagine you are running a courier service from a town in the Regierungsbezirke/NUTS-2 regions around the city of Karlsruhe: the regions of Karlsruhe and Darmstadt, or the very western part of Lower Franconia and the south-eastern part of Rhineland-Palatinate, see Figure 1.

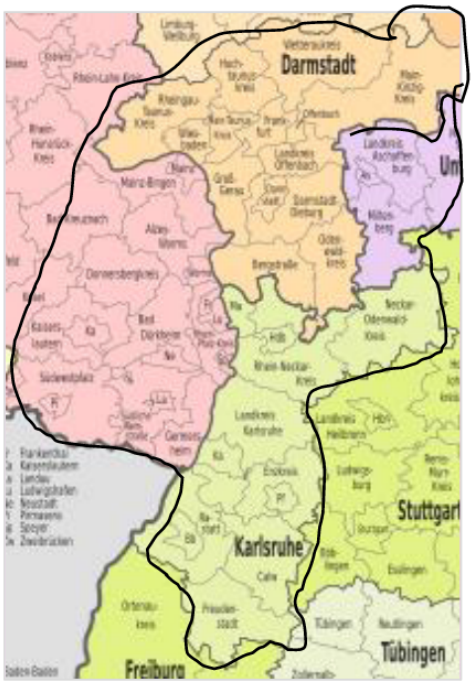

Figure 1: Karlsruhe and its surroundings. Based on https://de.wikipedia.org/wiki/Landkreis#/media/Datei:Germany,_administrative_divisions_(+districts)_-_de_-_colored.svg.

For your route planning, you need to know which lockdown rules are currently valid in the regions of your customers, which we also assume to be in those regions. As the lockdown rules

*Karlsruhe Institute of Technology (KIT). http://www.aifb.kit.edu/web/Tobias_K%C3%A4fer

have state-scope (Länder, NUTS-1), but the infection figures are published per region (Kreise, NUTS-3) we are facing an integration problem.

To assist the companies, we collaboratively build a data set in three phases:

1 Create and Publish Data

- Pick a town/municipality in those regions, e. g. whose name starts with the first letter of your first name.
- Invent a fictitious courier company, e. g. "ACME Inc.", which shall be located in this town
- Write a Turtle file with 15-25 RDF triples describing the company:
  - Some interesting properties to describe the company: name, post code, services offered, customers. The description of customers should have at least a post code.
  - Read the news about the NUTS-3 region (Kreis) of the company and add a n-ary relation with COVID-19 infections (daily total), timestamp, and postcode
  - Be creative and make up properties to describe the resources!
  - Per URI you mint, add at least one triple that uses `rdf:type` or its short form `a` as predicate, as it is good practice to say of what type a thing is.
- Create a SoLiD Pod[1] for the company
- Update the SoLiD Pod by copying and adapting triples from the Turtle file
- Hand in the URI of your fictitious company.

2 Query Data and Lift Data to RDF

- Lift the post-code-to-NUTS-mapping[2] to RDF, e. g. using tarql[3]. Make the URIs for the NUTS codes start with `http://data.europa.eu/nuts/code/`.
- Add the lifted data to your SoLiD Pod.
- Add to the data another n-ary relation with the current threshold for curfew in your state and the NUTS-1 code of your state
- Find out the WikiData URI for the NUTS region of your company and double-check that there is population data for the NUTS region of your company in Wikidata.
- Write a SPARQL query that finds out the infection numbers per 100 000 inhabitants in the regions of your company's customers. Hints: You may want to...
  - Have multiple `FROM` clauses for:
    - The customer data from your SoLiD Pod

[1] Find some providers at `https://solidproject.org/users/get-a-pod`, or use your student's webspace
[2] `https://ec.europa.eu/eurostat/web/nuts/correspondence-tables/postcodes-and-nuts`
[3] `https://tarql.github.io/`

    - The NUTS-postcode mapping from your SoLiD Pod (may also be included in the previous RDF graph)
    - The URI of the region in Wikidata
  - Use `BIND` to calculate the division and assign the result to a variable.
- Upload the SPARQL query to your Pod and hand in the URI of the query.

3 Integrate Data and Follow Links

- The supervisors provide you with URIs of companies in other students' data
- Add `foaf:based_near` links to your data that connect the your company to companies in adjacent NUTS-3 regions, thus building a list of local companies.
- Add `rdfs:subPropertyOf` links to your data that map the properties you defined in Phase 1 and 2 to:
  - `rdfs:label` for the name
  - `vcard:postal-code` for the post code
  - `swt:cvdinfections` for the infection number
  - `swt:cvdcurfewthreshold` for the threshold
- Add `rdfs:subClassOf` links to your data that map the classes you defined in Phase 1 and 2 to:
  - `foaf:Organization` for the class of all companies
- Again, hand in the URI of your company.

Putting it all together

The supervisors can then:

- Seeded with the URIs of the companies from Phase 1, explore the graph by following `foaf:based_near` links
- Integrate the data using RDFS entailment
- Build a SPARQL query that provides for each customer of a given company the current COVID-19 situation by:
  - calculating the infection number per 100 000 inhabitants for each NUTS-3 region
  - checking for each region (NUTS-3) whether the infection number per 100 000 inhabitants is above the state's (NUTS-1) threshold[4]

[4] Hereby, use the mappings between NUTS levels as provided at `https://data.europa.eu/euodp/repository/ec/estat/nuts/nuts.rdf`.

- Alternatively, build a SPARQL query that outputs the result per NUTS-3 region in order to display them on a map[5]

These steps can be built on top of Linked Data-Fu[6]. Hereby, the calculations and `BIND` have to be turned into triples that make use of the CWM built-in functions vocabulary[7].

Acknowledgements

Matthias Farnbauer-Schmidt provided helpful input on this exercise.

[5] Hereby, use the SPARQL XML result format, which can be transformed using an XSLT transformation to the KML format. KML can be imported to many mapping applications including Google Earth, which allows for nice display of the results.

[6] `http://linked-data-fu.github.io/`

[7] `https://www.w3.org/2000/10/swap/doc/CwmBuiltins`

Useful Tools

There are online services and tools to install on your local machine, which may be handy to check your solutions.

Online Services

- You can check whether your RDF is valid at `http://rdfshape.herokuapp.com/dataConversions`
- You can find out what commonly used prefixes to build CURIES typically refer to at `http://prefix.cc/`

Command-line Tools rapper and roqet

Using the `rapper` command, you can dereference a URI (or read from disk) and parse the received RDF:

```
rapper -i turtle <URI>
```

Using the `roqet` command, you can execute a SPARQL query from a given URI (or from disk), which dereferences the URIs given in `FROM` clauses of SPARQL queries:

```
roqet <URI>
```

The command-line tools `rapper` and `roqet` are available for different operating systems:

Installation on Linux

You can install the tools on Debian-based Linux distributions (such as Ubuntu) using:

```
sudo apt-get install rasqal-utils raptor2-utils
```

Installation on Windows

On Windows, the easiest is probably you have access to a Linux server via `ssh`. There are more (not-so-easy) options:

- You install Ubuntu for Windows[8] and follow the instructions for Linux (untested).
- You install the Cygwin subsystem and add the mentioned tools:
  1. Follow the instructions at `https://www.cygwin.com/install.html`
  2. Add the third-party repository `https://sourceware.org/cygwinports/`
  3. Install the packages `rasqal` and `raptor2`

Installation on MacOS

Using HomeBrew[9], install the packages `rasqal` and `raptor` (untested).

[8] `https://ubuntu.com/tutorials/ubuntu-on-windows`
[9] `https://brew.sh/`