LAB Assignment

This is a hands-on lab on knowledge graphs. We will use the Virtuoso triplestore for working with a knowledge graph. To prepare for the session, you are recommended to get familiar with the RDF Schema vocabulary , SPARQL specification and Virtuoso batch load procedure. In the sequel, we provide the environment setup and then exercises to be solved. Each group must upload the solution of the exercises to the Learn-SQL platform (look for the corresponding assignment event). One group member must submit the solutions (check below for a precise enumeration of what you need to submit) and list all group members in such document. Check the assignment deadline and be sure to meet it. It is a strict deadline!

Setup Instructions

For section A

For section A, you need to load the DBpedia ontology TBOX to Virtuoso on the Virtual Machine provided. Follow these steps:

- Go to: `cd /usr/local/virtuoso-opensource/share/virtuoso/vad`
- Download the ontology:
  `sudo wget http://downloads.dbpedia.org/2014/dbpedia_2014.owl.bz2`
- Unzip the downloaded file: `sudo bzip2 -d dbpedia_2014.owl.bz2`
- Create the *.graph* file necessary for the batch load to Virtuoso:
  `sudo touch dbpedia_2014.owl.graph`
- Add the line `http://localhost:8890/dbpedia` to `dbpedia_2014.owl.graph`.
  (you can use `sudo vi dbpedia_2014.owl.graph` to edit the file)

Similarly, we need to load some ABOX instances. Note that as this involves a large file, therefore some commands might take several minutes to execute. Run the following:

Terminal

- `sudo wget http://downloads.dbpedia.org/2014/en/instance_types_en.nt.bz2`
- `sudo bzip2 -d instance_types_en.nt.bz2`
- `sudo touch instance_types_en.nt.graph`
- add the line `http://localhost:8890/dbpedia` to `instance_types_en.nt.graph`
  (you can use `sudo vi instance_types_en.nt.graph` to edit the file)
- Finally, run the server: `sudo bash ~/SDM-Software/Virtuoso/start.sh`

Browser

- Open `http://localhost:8890` in the browser. Login using the credentials *'dba' 'dba'*
- Go to the tab *Database/Interactive SQL* and run:

```
ld_dir ('/usr/local/virtuoso-opensource/share/virtuoso/vad', '*.owl',
'http://localhost:8890/dbpedia') ; rdf_loader_run();
ld_dir ('/usr/local/virtuoso-opensource/share/virtuoso/vad', '*.nt',
'http://localhost:8890/dbpedia') ; rdf_loader_run();
```

- In the same tab, grant update permissions using: `grant SPARQL_UPDATE to "SPARQL"`
- Finally, go to the tab *Linked Data/Graphs*, check if there is a graph with the name `http://localhost:8890/dbpedia`. If yes, go to the tab *Linked Data/SPARQL* and put `http://localhost:8890/dbpedia` as the graph name

  Try:

  ```
  SELECT COUNT (*) WHERE { ?a ?b ?c . }
  ```

- The Virtuoso SPARQL endpoint can also be accessed directly at `http://localhost:8890/sparql`

What to deliver?

Upload ONE document with your solutions. Name the file as '`[Group]-[MemberSurname]+.pdf`'. The file must contain the three sections that you will find below and, within each section, insert your solution. SPARQL statements must compile and be fully correct (both syntactically and semantically). If you make any assumption not explicit in the statement, add a note before your SPARQL statement.

A Exploring Dbpedia

Dpedia

DBpedia[1] is a crowd-sourced community effort to extract structured content from the information created in various Wikimedia projects. This structured information resembles an open knowledge graph which is available for everyone on the Web. A knowledge graph is a special kind of database which stores knowledge in a machine-readable form and provides a means for information to be collected, organised, shared, searched and utilised. Google uses a similar approach to create those knowledge cards during search.

[1] https://wiki.dbpedia.org

Tasks

The most difficult part of working with external knowledge graphs is to understand what kind of information they contain. Unlike any other data repository, in a knowledge graph we must start exploring the TBOX to know what is in there and what can be interesting for us. In this first exercise you will be trained on how to explore information of a third party dataset. We propose a set of queries to explore DBpedia (in this case, for someone interested in actors):

1. Find the class representing an Actor in the dataset (using filters).
2. Find the super class for the class Actor.
3. Find all the actors in the dataset.
4. Get different classes that are defined as range of the properties that have the class Actor defined as their domain.
5. Find the super property of the goldenRaspberryAward property.
6. Return all the properties that have the class Actor as either their range or domain.
7. Return all persons that are not actors.

B Analytical queries on top of QBAirbase

European Air Quality Database

The European Air Quality Database[2] *contains data from different countries, integrated into a knowledge graph and thus available as Linked Data (LD). The LD structure of the European Air Quality Database is known as QBOAirbase*[3] *and contains the QB4OLAP vocabulary. QB4OLAP is an extension of QB vocabulary*[4] *to perform Business Intelligence over LD. It enables the user to represent OLAP cubes in RDF and to implement OLAP operators (such as Roll-up, Slice, and Dice) as SPARQL queries directly on the achieved RDF representation. QB4OLAP adds classes and properties to QB that gives datasets the capability of representing dimension levels, level members, rollup relations between levels and level members, and associating aggregate functions to measures.*

QBOAirbase is available via a SPARQL Endpoint: `http://lod.cs.aau.dk:8891/sparql`.

A visual representation of QBOAirbase's cube structure is given in Figure 1:

Finally a conceptual schema of the QBOAirbase dataset is shown in Figure 2.

[2] `https://www.eea.europa.eu/data-and-maps`

[3] `http://qweb.cs.aau.dk/qboairbase/`

[4] `https://www.w3.org/TR/vocab-data-cube/`

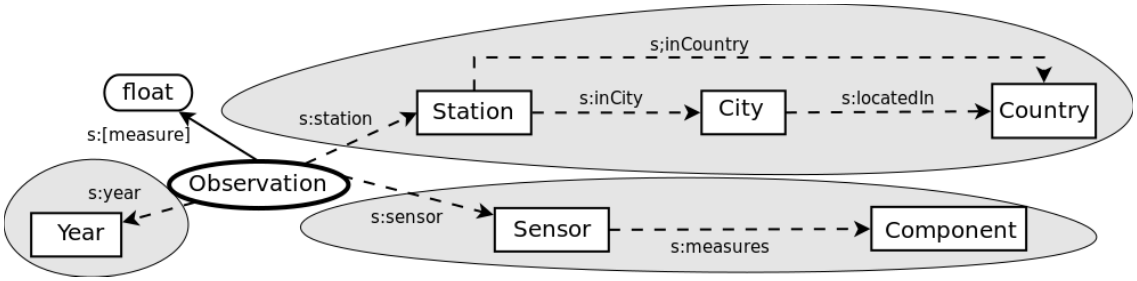

Figure 1: QBOAirbase cube structure

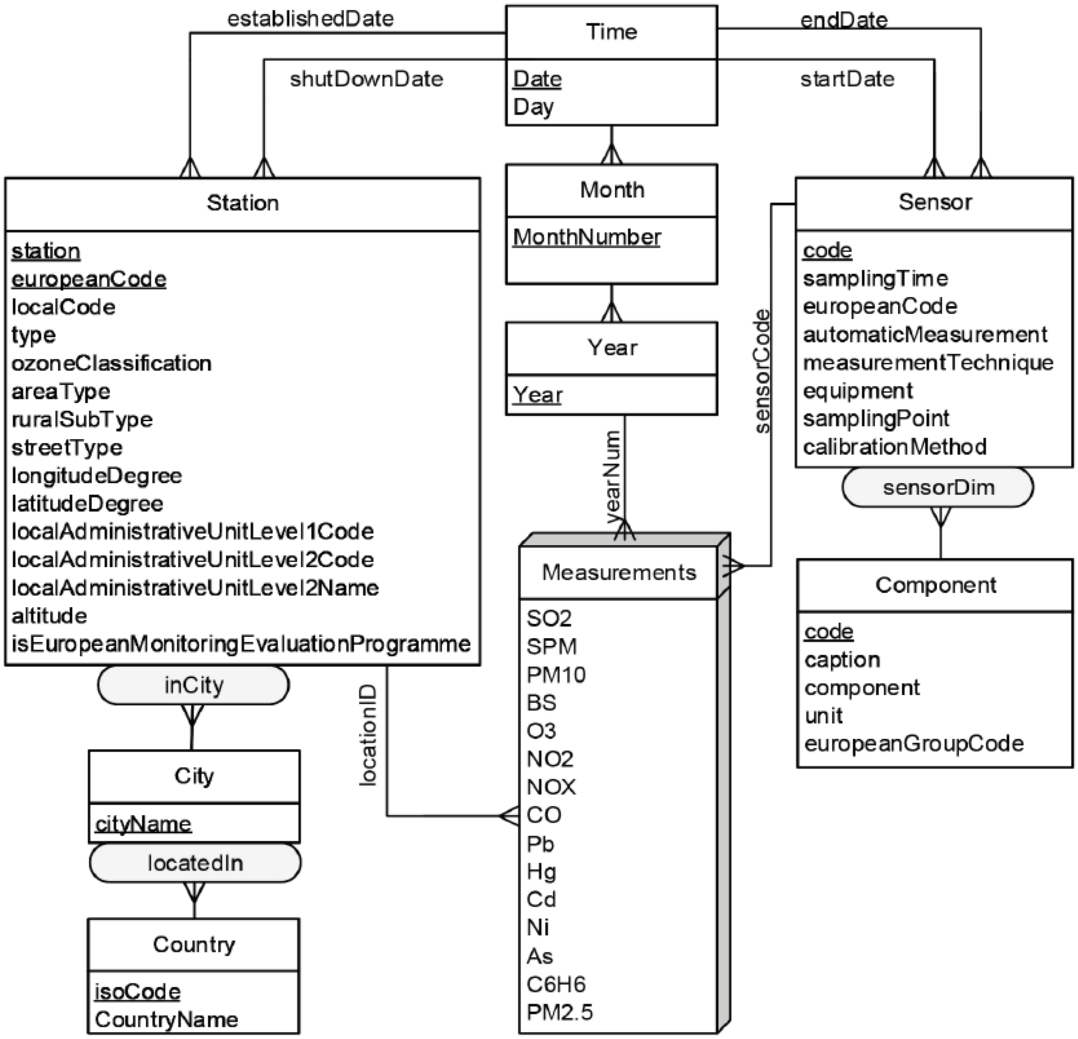

Figure 2: QBOAirbase conceptual schema

Tasks

In this exercise we will query an external integrated dataset of our interest. First, you will have to write some SPARQL queries about specific data aspects of the dataset. Then, once you are familiar with it, we will ask you to explore the data on your own and propose some additional queries.
Using the SPARQL endpoing provided in `http://lod.cs.aau.dk:8891/sparql`, write the SPARQL queries for the following:

1. List the country, station type, latitude, and longitude details of each station.
   Note: Limit the query to 25 results, and extract only the string values of the required object and not the whole IRIs.

2. List the 10 highest averages of C6H6 emission and the country and the year on which they were recorded.
   Note: A sensor has a property (defined through the prefix: `<http://qweb.cs.aau.dk/airbase/property/>`) `stastisticShortName`, and it can be Mean, Max, etc.

3. For each city and property type, give the yearly average emission for NO2, SO2, PB, and PM10.

4. Define 3 additional SPARQL queries (and their corresponding interpretation) that you think could be interesting for the domain of analyzing air quality/pollution.

C Ontology creation

In order to have a good grasp of the semantic web and the linked data initiative we need to know what ontologies are, how they are used, and how they are created. Note that in the world of knowledge graphs the main point is to reuse existing ontologies. That is, in order to facilitate data crossing and integration, it is worth to link our data with already existing data. In this assignment we will practice how to create your own ontology (i.e., a knowledge graph with a well-defined TBOX and ABOX).

C.1 TBOX definition

Define a TBOX for the research publication domain. Try to define it as complete as possible. The concepts to be used include, but are not limited to: *Author, Reviewer, Paper, Short Paper, Demo Paper, Survey Paper, Full Paper, Conference, Database Conference, Journal, Open Access Journal, Review, etc* along with their properties.

Note: To create the TBOX, you can use SPARQL and Virtuoso. Alternatively, you can make use of Protege[5], or VocBench[6], or any other graphical tool. If you are interested in it, you can use OWL instead of RDFS for this exercise.

Tasks:

1. Depending on how you created the TBOX, you need to provide either the SPARQL queries you used for creating the TBOX (in case you used SPARQL), or in case you used another tool, the methodology/method you used and the output generated in a graphical form (the lecturer should not install any additional tool to validate this part).
2. Provide a visual representation of the TBOX.

C.2 ABOX definition

In this section we want you to reuse the data you have about the research article domain (from the Assignment of Lab 1) and convert it into an ABOX, that is, define it as an RDF graph.

Note: There are many ways to convert CSV, or relational data, or JSON into RDF. You can use the Jena API[7] and do it programatically or use some existing software, see Any23[8], Open Refine[9] with its RDF extension[10], or any other available tool.

Tasks

1. Explain the method used to define the ABOX.

C.3 Linking ABOX to TBOX

In this section we want to link the ABOX with the TBOX defined. That is, use SPARQL to create new triples that will link the TBOX elements with ABOX elements.

Note: See the `CONSTRUCT` statement in SPARQL.

Tasks

1. Provide the SPARQL queries required to create the link between the ABOX and TBOX.

[5] `https://protege.stanford.edu`

[6] `http://vocbench.uniroma2.it/downloads`

[7] `http://jena.apache.org/index.html`

[8] `https://any23.apache.org`

[9] `http://openrefine.org/`

[10] `https://github.com/stkenny/grefine-rdf-extension/wiki`

2. Provide a summary table with simple statistics about the RDF graph obtained, e.g., the number of classes, the number of properties, the number of instances, etc.

C.4 Queries on top of the Ontology

In this section we want to pose a set of queries on top of our created ontology. We are specifically interested on the benefits of having an explicit TBOX defined. Thus, in this section we want to define queries that exploit the TBOX and queries that ignore the existence of the TBOX.

Tasks

Write two versions for each of the following queries (one exploiting the TBOX, and another assuming the TBOX does not exist). Please explicitly state any assumptions you make.

1. Find all the Authors.
2. Find all the properties whose domain is Author.
3. Find all the properties whose domain is either Conference or Journal.
4. Find all the things that Authors have created (either Reviews or Papers).