Group 5: Technical report on Assignment 1

Arthur Dorzee, Christoph Bensch, Elena Heinze, Oliver Bensch, Tjerk Kuil

February 17, 2020

This document reports on the process of creating a knowledge graph when provided with two data files. To this end, it will first consider the conversion of the two data files of different types to RDF. After that, to combine the information retrieved from the different files, the graphs created by the Converted RDF Files will be linked. Finally, to test the working of the conversion and linking, several SPARQL queries will be executed on the graph.

1 Conversion from the datasets to RDF using RML

The conversion of the data to RDF format was performed using a RML mapper. This section will go further into the choices of the ontologies that were used. The RML mapping file we use was added as "mapping_adapted.ttl". It contains the mapping for the CSV file as well as for the XML.

The first section of the mapping_adapted.ttl file considers the mapping of data in the CSV file. In order not to have too many different dependencies and prefixes, we have relied on the ontology of DBpedia, as it is the largest and best maintained. DBpedia provides a lot of complex prefixes like "dbo:iso31661Code" for the numeric postcode. Unfortunately we were not able to find all prefixes. For "postal code regex" we therefore used dqm from purl.org and classified it as "dqm:regex". Additionally we used the previously delivered prefixes from geonames for "continent" and "country".

In the second part of the mapping_adapted.ttl file the mapping for the data in the XML file takes place. For the Country and year variable, we could use the previously used dbo ontology. Only for the GDP we included the property of DBpedia and classified it as "dbp:gdp".

In the end, the number of RDF Triples generated by applying the mapping file was: 82668 (the total for both the csv and the XML file).

2 Linking the RDF graphs

Linking the RDF triples created in the previous section was done using LIMES. For this process an adaptation of the XML configuration file provided by the

teaching staff was used. The main points of interest are the settings for the metric that we used. The link to the list of metrics available on the website of LIMES gave a 404 page not found error, so we were limited in our choice between levenshtein,cosine and trigrams. Trigrams returned a lot more connections to be reviewed, but we found that most were incorrect. The levenshtein metric returned fewer triples to be reviewed but with more correctly identified triples. For the settings, the accepted threshold was 0,6 and the review threshold is 0,5. During the experiments, it was found that for an review threshold of 0,6, the 2 review connections were correct but for 0,5 it started to return mostly wrong connections in the review section. The cosine function did slightly better than levenshtein: it returned more triples to be reviewed but the amount was manageable and most of them were correct. So the eventual choice for metric was cosine with an acceptance threshold of 0,85 and review threshold 0,7. With these settings, eventually we managed to link countries cross data set, with a total of number of linked triples across the dataset of: 10503, meaning we created a triple for the gdp value in a year between 1960 and 2017 for 181 countries.

3 Querying the linked RDF data

3.1 Query 1

List all countries with population less than 50,000 and order them from the smallest to the largest in terms of landmass area (square kilometres)

3.1.1 Explanation

This query creates instances of "geonames.org/Country" and checks if they have a population¡=50000. Afterwards it is specified with "order by asc(?area)" that these records are sorted in ascending order by the size of the country. Finally the output is limited to 10 with "LIMIT 10".

3.2 Query 2

List the countries with the top 10 highest GDP values in 2017.

3.2.1 Explanation

This query creates instances of "GdpEntry". Additionally, dbo:Year "2017" specifies that only entries from the year 2017 are displayed. Here the entries are sorted with "ORDER BY DESC(xsd:float(?gdp))" in descending order of GDP. The cast to float is necessary for the GDP to be recognized as a number for sorting. Again, the output is limited to 10.

We noticed that results like World / high income / OECD members are shown here. Therefore we have added a modified version, which only shows countries, as an additional query 7.

3.3 Query 3

List the countries with the top 10 highest increases in GDP between 1960 and 2017.

3.3.1 Explanation

Similar to Query 2, we create instances of GdpEntry. Only here we create 2 instances. One for the year 1960 and one for the year 2017 and check with the filter (?country1 = ?country2) that this is the same country. Then we subtract the gdp of 1960 from that of 2017 and save this value in ?gdp-new. Finally we sort the output by the value ?gdp-new and limit it to 10.

As in query 2, results world / high income etc. are also displayed here. A solution that only shows the countries can be found in the additional query 8.

3.4 Query 4

For each continent, count the number of countries in that continent that are in the top 20 for highest increases in GDP between 1960 and 2017.

3.4.1 Explanation

This query checks the GDP growth from 1960-2017 as described in query 2 and then checks whether it is a country, as described in query 8. This query is enclosed by a query that displays all country - gn:continent - continent tuples and checks if they are in the inner output filter(?country3 = ?countryUrl). Then these tuples are counted and sorted in descending order and displayed as continent.

3.5 Query 5

Construct the triples representing the GDP per capita for each country in 2017:

3.5.1 Explanation

This query collects all GdpEntrys from the year 2017 in the inner loop and searches for the appropriate gn:Country data set in the outer loop. The GDP per Capita is then calculated with the bind and stored in ?gdppc. The CONSTRUCT at the beginning defines this new tuple with the prefix grossDomesticProductPerCapita.

3.6 Query 6

Directly insert the triples representing the GDP per capita for each country in 2017, into your triplestore on GraphDB in the graph with namespace: “http://kg-course/query”

3.6.1 Explanation

This query is structured like query 5, except that at the beginning with INSERT the instruction is given to save the output in the new graph kg-course/query.

3.7 Bonus Queries

3.8 Query 7 - Question 2

limited to real countries and not including world, EU and such

3.8.1 Explanation

We solved this by enclosing query 2 with a new query that lists all countries and checks with sameAs if the GdpEntry also has a country entry.

3.9 Query 8 - Question 3

Is limited to real countries and not including world, EU and such

3.9.1 Explanation

As in query 7, we check here whether a gn:Country entry also exists for the inner GdpEntry.

4 Conclusion

Finding working ontologies and setting up the database, mapping and linking systems turned out to be a substantial amount of work. But the database is working, commonly used ontologies and the answers to the queries are feasible responses to what was asked, so the project can be considered successful.

5 Appendix

5.1 Query 1

```
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX dbo: <http://dbpedia.org/ontology/>
PREFIX dbop: <http://dbpedia.org/ontology/PopulatedPlace/>

select ?instance ?label ?population ?area
where {
    ?instance a <http://geonames.org/Country> .
    ?instance dbo:population ?population .
    ?instance dbop:areaTotal ?area
    FILTER (?population <= 50000)
```

```
    OPTIONAL { ?instance rdfs:label ?label . }             # Display the label if one
    OPTIONAL { ?instance dbo:population ?population . }    # Display the population
    OPTIONAL { ?instance dbop:areaTotal ?area . }    # Display the population
}
order by asc(?area) LIMIT 10
```

5.2 Query 2

```
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX dbo: <http://dbpedia.org/ontology/>
PREFIX dbp: <http://dbpedia.org/property/>
PREFIX wb: <http://worldbank.org/>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>

select ?instance ?gdp ?year ?label ?countryname
where {
    ?instance a wb:GdpEntry .
    ?instance rdfs:label ?label .
    ?instance dbo:Year "2017" .
    ?instance dbp:gdp ?gdp .
    ?instance dbo:Country ?countryname .

}ORDER BY DESC(xsd:float(?gdp)) LIMIT 10
```

5.3 Query 3

```
    PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX dbo: <http://dbpedia.org/ontology/>
PREFIX dbp: <http://dbpedia.org/property/>
PREFIX wb: <http://worldbank.org/>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX aGeo: <http://example.org/geo#>

select ?country1 ?country2 ?gdp1 ?year2 ?gdp2 ?gdp_new
where {
    ?instance a wb:GdpEntry .
    ?instance dbo:Year "1960".
    ?instance rdfs:label ?country1 .
    ?instance dbp:gdp ?gdp1 .

    ?instance2 a wb:GdpEntry .
    ?instance2 dbo:Year ?year2 .
    ?instance2 rdfs:label ?country2 .
    ?instance2 dbp:gdp ?gdp2 .
```

```
    Filter(?country1 = ?country2)
    Filter(?year2 = "2017")

    bind(xsd:float(?gdp2) - xsd:float(?gdp1) as ?gdp_new)
}order by desc(?gdp_new) LIMIT 10
```

5.4 Query 4

```
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX dbo: <http://dbpedia.org/ontology/>
PREFIX dbp: <http://dbpedia.org/property/>
PREFIX wb: <http://worldbank.org/>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX aGeo: <http://example.org/geo#>
PREFIX gn: <http://geonames.org/>
PREFIX owl: <http://www.w3.org/2002/07/owl#>

select ?continent (COUNT(*) AS ?count)
where {
    ?country3 gn:continent ?continent.
    {
        select ?countryUrl ?country ?gdp1960 ?gdp2017 ?gdp_difference
        where {
            ?countryUrl a gn:Country .
            {
                select ?instance ?gdp1960 ?gdp2017 ?country ?instance2  ?cn2 ?year2 ?gdp
                where {
                    ?instance a wb:GdpEntry .
                    ?instance dbo:Year "1960".
                    ?instance rdfs:label ?country .
                    ?instance dbp:gdp ?gdp1960 .

                    ?instance2 a wb:GdpEntry .
                    ?instance2 dbo:Year ?year2 .
                    ?instance2 rdfs:label ?cn2 .
                    ?instance2 dbp:gdp ?gdp2017 .
                    Filter(?country = ?cn2)
                    Filter(?year2 = "2017")

                    bind(xsd:float(?gdp2017) - xsd:float(?gdp1960) as ?gdp_difference)
                }
            }
            ?countryUrl owl:sameAs ?instance .

        }order by desc(?gdp_difference) LIMIT 20
```

```
    }
    Filter(?country3 = ?countryUrl)

}Group by(?continent) ORDER BY DESC(COUNT(*))
```

5.5 Query 5

```
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX dbo: <http://dbpedia.org/ontology/>
PREFIX dbp: <http://dbpedia.org/property/>
PREFIX wb: <http://worldbank.org/>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX aGeo: <http://example.org/geo#>
PREFIX gn: <http://geonames.org/>
PREFIX owl: <http://www.w3.org/2002/07/owl#>

CONSTRUCT {
    ?countryUrl dbo:grossDomesticProductPerCapita ?gdppc .
}
where {
    ?countryUrl a gn:Country .
    ?countryUrl dbo:population ?population .
    {
        select ?instance ?gdp ?country
        where {
            ?instance a wb:GdpEntry .
            ?instance dbo:Year "2017".
            ?instance rdfs:label ?country .
            ?instance dbp:gdp ?gdp .
        }
    }
?countryUrl owl:sameAs ?instance .

    bind(xsd:float(?gdp) / ?population as ?gdppc)
}
```

5.6 Query 6

```
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX dbo: <http://dbpedia.org/ontology/>
PREFIX dbp: <http://dbpedia.org/property/>
PREFIX wb: <http://worldbank.org/>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX aGeo: <http://example.org/geo#>
```

```
PREFIX gn: <http://geonames.org/>
PREFIX owl: <http://www.w3.org/2002/07/owl#>

INSERT {
 GRAPH <http://kg-course/query> {
  ?countryUrl dbo:grossDomesticProductPerCapita ?gdppc .
 }
}
where {
    ?countryUrl a gn:Country .
    ?countryUrl dbo:population ?population .
    {
        select ?instance ?gdp ?country
        where {
            ?instance a wb:GdpEntry .
            ?instance dbo:Year "2017".
            ?instance rdfs:label ?country .
            ?instance dbp:gdp ?gdp .
        }
    }
?countryUrl owl:sameAs ?instance .

    bind(xsd:float(?gdp) / ?population as ?gdppc)
}
```

5.7 Query 7

Like question 2 but just countries, no world, EU and the like.

```
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX dbo: <http://dbpedia.org/ontology/>
PREFIX dbp: <http://dbpedia.org/property/>
PREFIX wb: <http://worldbank.org/>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX gn: <http://geonames.org/>
PREFIX owl: <http://www.w3.org/2002/07/owl#>

select ?countryURL ?gdp ?country
where {
    ?countryURL a gn:Country .
    {
        select ?instance ?gdp ?year ?country
        where {
            ?instance a wb:GdpEntry .
```

```
            ?instance rdfs:label ?country .
            ?instance dbo:Year "2017" .
            ?instance dbp:gdp ?gdp .
        }
    }
    ?countryURL owl:sameAs ?instance .
}ORDER BY DESC(xsd:float(?gdp)) LIMIT 10
```

5.8 Query 8

Like question 3 but just for countries.

```
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX dbo: <http://dbpedia.org/ontology/>
PREFIX dbp: <http://dbpedia.org/property/>
PREFIX wb: <http://worldbank.org/>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX aGeo: <http://example.org/geo#>
PREFIX gn: <http://geonames.org/>
PREFIX owl: <http://www.w3.org/2002/07/owl#>

select ?countryUrl ?country ?gdp1960 ?gdp2017 ?gdp_difference
where {
    ?countryUrl a gn:Country .
    {
        select ?instance ?gdp1960 ?gdp2017 ?country ?instance2  ?cn2 ?year2 ?gdp_differe
        where {
            ?instance a wb:GdpEntry .
            ?instance dbo:Year "1960".
            ?instance rdfs:label ?country .
            ?instance dbp:gdp ?gdp1960 .

            ?instance2 a wb:GdpEntry .
            ?instance2 dbo:Year ?year2 .
            ?instance2 rdfs:label ?cn2 .
            ?instance2 dbp:gdp ?gdp2017 .
            Filter(?country = ?cn2)
     Filter(?year2 = "2017")

            bind(xsd:float(?gdp2017) - xsd:float(?gdp1960) as ?gdp_difference)
        }
    }
?countryUrl owl:sameAs ?instance .
```

```
}order by desc(?gdp_difference) LIMIT 10
```