Coursework **Fariz Darari, Werner Nutt**

Coursework C3: Data Quality Analysis over DBpedia using SPARQL

Intention

The idea of this coursework is to:

- Introduce you to problem of data quality in general.
- Make you aware of some data quality issues over DBpedia.
- Let you use SPARQL to perform data quality analysis over DBpedia.

Background Information on Data Quality

Data quality is often defined as "fitness for use" of data. In particular, in some often-cited papers Wang and Strong [4] defined data quality as data that is fit for use by data consumers, while Olson [3] defined that data has quality if it satisfies the requirements of its intended use.

There are several data quality aspects [2], which include:

- Representational consistency: Data are always presented in the same format and are compatible with the previous data.
- Timeliness: The age of the data is appropriate for the task at hand.
- Completeness: Data are of sufficient depth, breadth and scope for the task at hand.

On the Semantic Web, anyone can say anything about any topic [1], therefore data quality in the Semantic Web can largely vary. Oftentimes, data quality is different among data sources. A data source can be complete for "All actors in Tarantino movies", while another data source is not.

Goal

DBpedia, as a Semantic Web data source, has several data quality issues. In the tasks below, you are asked to analyze some of these issues and take them account in formulating SPARQL queries.

Aspect 1: Representational Inconsistencies over DBpedia

Over DBpedia, there are various ways to represent that a resource is an actor, two of which are by using the class `u:Actor`[1] and the class `yago:Actor109765278`.[2] Moreover, if a resource has a property with the domain of actor, the resource is therefore inferred to be of type actor. Similarly, if a resource is the value of a property with the range of actor, the resource is inferred to be of type actor, too.[3] As a consequence, to query about actors over DBpedia, one has to consider all these cases.

Task 1: Using SPARQL queries, count how many resources

- have the type `u:Actor` but not `yago:Actor109765278`,
- have the type `yago:Actor109765278`, but not `u:Actor`,
- have both types `yago:Actor109765278`, but not `u:Actor`.

Task 2: Formulate the query

"Italian actors born between 1920 to 1970",

in such a way that you have to handle the representational inconsistency of DBpedia. Explain which inconsistencies are handled in which way by your query.
Task 3: Generalize the query to a pattern of the form

"Actors of nationality X born between Y and Z".

Test your pattern and assess the quality of the answers.

Instructions: Consider in your query different ways of DBpedia to represent that a resource is an Italian, a resource is an actor[4] and a resource was born between 1920 to 1970. Please, provide as well an explanation how your query can handle these representational inconsistencies.

Aspect 2: Out-of-Date Data over DBpedia

Besides the standard English DBpedia[5], there is another version of DBpedia, called the Live DBpedia[6]. The motivation behind the development of Live DBpedia is to have a continuous synchronization between DBpedia and Wikipedia. On the other hand, the standard DBpedia is updated approximately twice a year.[7] As a consequence, data over Live DBpedia is more timely than that over DBpedia.

SPARQL has the `SERVICE` extension[8] that allows a query author to direct a portion of a query to a particular SPARQL endpoint. Results are returned to the federated query

[1] http://umbel.org/umbel/rc/Actor

[2] http://dbpedia.org/class/yago/Actor109765278

[3] Recall the RDFS inference.

[4] There can be other ways to represent a resource is an actor than using the classes `u:Actor` and `yago:Actor109765278`.

[5] http://dbpedia.org/

[6] http://live.dbpedia.org/

[7] http://wiki.dbpedia.org/faq

[8] http://www.w3.org/TR/sparql11-federated-query/

processor and are combined with results from the rest of the query. As an example, to execute remotely a query of Tarantino movies over DBpedia, you can access the SPARQL query page[9] and execute the following query:

```
PREFIX dbpedia-owl: <http://dbpedia.org/ontology/>
PREFIX dbpedia: <http://dbpedia.org/resource/>

SELECT *
WHERE {
  SERVICE <http://dbpedia.org/sparql> {
    ?m dbpedia-owl:director dbpedia:Quentin_Tarantino .
  }
}
```

Task 4: Formulate a query that will return the countries where the presidents in DBpedia and Live DBpedia differ.

Task 5: Formulate a query that will return the ratio of out-of-date presidents (if the presidents of countries in DBpedia are not equal to corresponding values in Live DBpedia) to up-to-date presidents (if the presidents of countries in DBpedia are equal to corresponding values in Live DBpedia).[10]

Hint: As a reference on how DBpedia and Live DBpedia represent a presidency, you might have a look at the resource of President of Indonesia over DBpedia[11] and Live DBpedia.[12]

Aspect 3: Data Completeness

There are many versions of DBpedia in different languages: English[13], German[14] and Italian.[15] Each version might have different degrees of completeness.

Task 6: Analyze the completeness of these different versions of DBpedia for data about Quentin Tarantino. Compare in the three versions of DBpedia using SPARQL queries with the `SERVICE` operator:

- The number of movies directed by Tarantino.
- The number of actors in the movie Reservoir Dogs.
- The number of triples about Quentin Tarantino, that is, the triples where Tarantino appearing in the subject or object position.

[9] http://sparql.org/query.html

[10] For all the tasks here, we assume that different URIs represent different people.

[11] http://dbpedia.org/page/President_of_Indonesia

[12] http://live.dbpedia.org/page/President_of_Indonesia

[13] http://dbpedia.org/

[14] http://de.dbpedia.org/

[15] http://it.dbpedia.org/

Deliverables

Submit your solutions to `fariz.darari@stud-inf.unibz.it` by Friday, 9 January. Your file must have the name “yourname-CW3.txt”.

References

[1] D. Allemang and J. A. Hendler. *Semantic Web for the Working Ontologist - Effective Modeling in RDFS and OWL, Second Edition*. Morgan Kaufmann, 2011.

[2] C. Batini and M. Scannapieco. *Data Quality: Concepts, Methodologies and Techniques*. Data-Centric Systems and Applications. Springer, 2006.

[3] J. E. Olson. *Data Quality: The Accuracy Dimension*. Morgan Kaufmann, 2003.

[4] R. Y. Wang and D. M. Strong. Beyond Accuracy: What Data Quality Means to Data Consumers. *J. of Management Information Systems*, 12(4):5–33, 1996.