188.399 Introduction to Semantic Systems 2022W

Assignment 4 - Knowledge Graph Creation (17,5 pt)

Instructions

Deadline

Make sure to upload your results **by January 6th, 2023**!

What you should hand in

Please upload your work to TUWEL by January 6th, 2023. Please name all your files with the following format: A[x]_[family name]_[student number], where x represents the number of the assignment.

Your submission should be a zip file (**e.g., A4_Mustermann_1234.zip**) that contains:

- A short PDF report (details in Task 3)
- The updated film ontology in **.ttl** format
- The OntoRefine projects (in a compressed form) **and** the combined KG from CSV files (in **.ttl** format)
- The RML/YARRRML mappings (in a text file) **and** the combined KG from JSON files (in **.ttl** format)
- Your SPARQL queries (in a text file)
- The final knowledge graph in **.ttl** format

Questions

Please post general questions in the TUWEL discussion forum of the course. You can also discuss problems and issues you are facing there. We appreciate it if you help other students out with general problems or questions regarding the tools used. For obvious reasons, however, please do not post any solutions there.

You can also contact Filip Kovacevic and Laura Waltersdorfer directly (with specific questions) at:

- filip.kovacevic@tuwien.ac.at
- laura.waltersdorfer@tuwien.ac.at

Please use subject line ISS_2022_<your subject> to minimize the probability that your Email gets lost in our inbox.

188.399 Introduction to Semantic Systems 2022W

Assignment 4 - Knowledge Graph Creation (17,5 pt)

Introduction

In this assignment, you will exercise important skills and technologies needed for populating knowledge graphs. In particular, you will follow the following two tasks (each task is described in detail in the next section):

1. Creating a knowledge graph by (i) extending the Film ontology with necessary classes and properties, and (ii) populating an ontology with instances from several datasets.
2. Applying SPARQL queries on the resulting KG to further enrich it.

Tasks description

Task 1: Create a Knowledge Graph (12 points + 2,5 bonus points)

You will create a knowledge graph by populating an ontology with instances extracted from several datasets. To this end, you will need to use and extend the film ontology to include additional classes and properties as the following (*blue texts in Table 1 show the already existing classes and properties, while black texts are required extensions. Gray texts are optional*):

Table 1 Film Ontology classes and properties for KG population from CSV and JSON files

For CSV files	**For JSON files**
Film ● rdfs:label (String) ● releaseYear (integer) ● budget (float) ● duration (integer) ● id (String) ● isAdultFilm (boolean) ● homepage (url) ● hasIMDBResource (IMDBResource) ● description (String) ● keyword (String) ● tagline (String) ● revenue (integer) ● hasWikidataLink (url) - **OPTIONAL** IMDBResource ● id (String) ● url (String) ● vote_average (float) ● vote_count (integer)	Film ● originalTitle (String) ● hasFilmStudio (FilmStudio) ● hasGenre (Genre) ● hasDirector (Director) ● hasScriptWriter (ScriptWriter) ● hasActor (Actor) ● hasCast (Cast) ● hasSpokenLanguage (String) ● hasProductionCountry (String) Actor ● fullName (String) ● gender (String) Genre ● id (Integer) ● rdfs:label (String) FilmStudio ● id (Integer) ● rdfs:label (String) Cast ● hasCastActor (Actor) ● hasCastCharacter (String)

Assignment 4 - Knowledge Graph Creation (17,5 pt)

The datasets necessary for this task (three CSV files and two JSON files) are available online [1]. These datasets are linked through the consistent use of film IDs (i.e., "id") to allow for convenient RDF graph merging/integration.

For populating the data, you will need to use OpenRefine [2] for CSV files and RML [3] tools (e.g., caRML [4], carml-cli [5], RMLMapper [6] or Matey [7]) for JSON files.

Concretely:

- Please extend the film ontology to include the additional classes and properties (2 points).
- Please use OpenRefine to **define the RDFRefine mappings** and **generate the RDF graphs** from CSV datasets, containing instances based on classes and properties mentioned in Table 1 (4 points).
    - **Note**: You can have separate OpenRefine projects for each CSV file.
    - **Optionally**, you can execute entity reconciliation feature from OpenRefine to link all film instances in the CSV files to WikiData using "`hasWikidataLink`" relation (2,5 bonus points)
- Please **define the RML [2] mappings** and **generate the RDF graphs** from JSON datasets, containing instances based on classes and properties mentioned in Table 1 (6 points).
- Finally, briefly document your work in a short report as described on the first page of this document.

Task 2: Enrich your Knowledge Graph through SPARQL (3 points)

In this task you will enrich the knowledge graph resulting from Task 1 by applying SPARQL CONSTRUCT queries.

Concretely:

- Write at least 3 **SPARQL CONSTRUCT** queries to further enrich the integrated data created in Task 1 (combining RDF graphs from CSV and JSON files). Each query is worth 1 point (max 3 points).
    - You might use the Jena API to run a construct query on the knowledge graph and then add the result back into the KG, thus enriching it.
    - Alternatively, you can use GraphDB for storing your KG, executing the queries and using the query results to enrich the KG (e.g., you can replace CONSTRUCT keyword with INSERT[1]).
    - **Note**: we will need you to come up with your own original ideas for these SPARQL CONSTRUCT queries to enrich your RDF graphs. Obvious inference queries, such as adding types of instances based on `rdfs:subClassOf`, `rdfs:domain`, or `rdfs:range` are not allowed.
- Save the final, resulting knowledge graph as a .ttl file.

[1] http://vos.openlinksw.com/owiki/wiki/VOS/VirtTipsAndTricksGuideConstructINSERTStatement

Assignment 4 - Knowledge Graph Creation (17,5 pt)

Task 3: Documentation (2,5 points)

Briefly document your work in a short PDF report (max 10 pages) containing the following information:

- An overview and high-level description of the RML mappings, explaining what data is mapped to which parts of the ontology (the actual mappings need to be delivered in a text file);
- An example sub-graph from CSV files in TTL serialization containing (at least) an instance from each class that are connected to each other.
- Screenshots of RDFRefine mappings and high-level description of the mapping (the actual OpenRefine projects need to be delivered separately).
- An example sub-graph from JSON files in TTL serialization containing (at least) an instance from each class that are connected to each other.
- An overview and high level description of the SPARQL queries - i.e., what is the rationale behind these queries, how do they enrich the data?

Tools and resources:

[1] GDrive: http://bit.ly/3VIunLx
consist of five files: “1000_movies_metadata.json”, “1000_keywords.csv”, “1000_links.csv”, “1000_movies_metadata.csv”, and “1000_credits.json”.

[2] OpenRefine: https://openrefine.org/
RDFRefine Extension: https://github.com/stkenny/grefine-rdf-extension
An OpenRefine project with example RDF skeleton: https://bit.ly/iss_openrefine_example

[3] RML: http://semweb.mmlab.be/rml/spec.html;
RML Implementation Report: https://rml.io/implementation-report/

[4] caRML (RML engine): https://github.com/carml/carml
Example caRML implementation: https://github.com/fekaputra/carmlizer;
Example map: https://github.com/fekaputra/carmlizer/blob/master/rml/weather.rml.ttl

[5] carml-cli: https://github.com/netage/carml-cli

[6] RML Mapper: https://github.com/RMLio/rmlmapper-java

[7] YARRRML Matey: https://rml.io/yarrrml/matey/
YARRRML Specification: https://rml.io/yarrrml/spec/