Semantic Web
Lab Assignment 6 (Two Week Lab)

1. Create a simple federated system using Jena

   You are given two years of conference attendance information that you need to federate using Jena. Your program will find people at both events and federate them.

   The files you are provided are described in the table.

Filename	Notes
iswc-2018-complete.rdf	2018 conference attendance
iswc-2019-complete.ttl	2019 conference attendance
conference-ontology.owl	The Conference Ontology – a useful reference

   For both files, the original data has been modified so that each person includes the year ("person2018" or "person2019") in its namespace.

   a. Investigate the data and ontology information so that you are comfortable with the constructs.

      In the worksheet (see part 2), answer the following questions:

      1. What is the meaning of the FOAF property "mbox_sha1sum"?
      2. What conflicts or inconsistencies occur when you define entities to be identical with "owl:sameAs"?
      3. What are the issues with reasoning across federated datasets?

   b. Load the two data files into a Fuseki store using **two different in-memory datasets**, "FedSet1" and "FedSet2". To do so, start a Fuseki server and browse to "localhost:3030". Create the datasets using the "Manage Datasets → add new dataset". Use the "upload file" in each dataset to load the above conference files:
      i. 2018 into FedSet1
      ii. 2019 into FedSet2

      Select the file and pick "upload all" or "upload now" before changing to the next dataset. If you need to restart the Fuseki server, the datasets will be recreated, but you must load the files again.

HINT: Use the Fuseki SPARQL web page to work out your basic SPARQL query syntax and establish a known set of results for the following steps. Access the datasets using SERVICE statements for each set (it doesn't matter which dataset you use for the SPARQL endpoint if you use SERVICE statements for all of them).

c. Create a Jena program to use the Fuseki SPARQL endpoints (see https://jena.apache.org/documentation/query/index.html for more information—particularly the API, Fuseki, and Federated queries). Create a SPARQL query to find and print each URI for people that have matching FOAF "mbox_sha1sum" values and have created papers at both events—a person's URIs may be different between datasets, so show both. The "mbox" info and papers don't need to be shown at this time. During the query, sort the data and only show unique results.

NOTE: This process assumes that persons with unique "mbox_sha1sum" values are different people even if they have the same person URI. It also assumes that persons with the same "mbox_sha1sum" value are the same person even if they have different person URIs.

Print each result, then the number of results.

After all is listed, print "End of First Listing".

NOTE: If you use Filter in your SPARQL, not all SPARQL implementations use the same syntax for establishing equivalence.

d. Assert Equivalences: Create a new in-memory dataset called "Same". For each matching URI pair found in step c, assert triples into the "Same" dataset using "owl:sameAs" as the property between the pair. For each assertion, add reification data stating that you are the creator of the triple—use Dublin core properties for the "creator".

e. Manually implement "sameAs" capability as follows (this is as if someone is asking you about the papers published by a specific individual across the years).

> Write a simple 'find' query that returns only the **subject URI** of each of the "owl:sameAs" triples (created in step d)—this is your list of individuals.

Loop through the list of individuals and issue a second SPARQL query to find and print the person's literal name and all papers created by that person. This query must include the **subject URI** from the above query as a constant. This query

should include a pattern that finds and uses the "owl:sameAs" **object URI** as a variable (i.e., the equivalent URI at the other end of the "owl:sameAs" triples) and returns the papers for both person URIs (HINT: UNION). You should be **querying all 3 stores** within a single query with the SERVICE keyword and may need to do so in multiple sections of the query. During the query, sort the data and only show unique results. Count each result set and sum them over the loop.

Print each result, then the sum of the result sets.

After all are listed print "End of Second Listing"

f. Create a new query to get all of the results at once—copy the second SPARQL query from the loop in step e and modify. Instead of looping through the individuals as you did in step e, use a variable for both the **subject URI** and the **object URI** for the "owl:sameAs" triples you asserted in step d (i.e., change the **subject URI** constant to a variable). This should return all results in a single query. As before, print each person's name and all papers submitted by that person as you did in step e. During the query, sort the data and only show unique lines.

   Print each result, then the number of results. It should match step e's count.

   After all are listed print "End of Third Listing".

g. Output the in-memory model in the N3 format to filename Lab6_<YourID>.n3 in the default execution directory (i.e. ./) where <YourID> should be replaced with your Net ID.

h. Screen capture all your outputs that print to the screen in a Microsoft Word document. On the first page use the title "Laboratory 6" and place your name below it. Name the file Lab6_<YourID>.docx (or .docx). It must be readable in Word 2007 (Word 97 is also acceptable).

   i. Please understand that fraudulent screen grabs will result in a score of zero for the lab. It is much better to not have any.

2. Fill out and submit the Reasoning Worksheet. Submit as a .doc or .pdf file.

Please compress your project as well as the Reasoning worksheet into one zip file, and submit the zip file on eLearning.

Grading:

25% - Proper program documentation.

60% - Exercise 1 – Correct people merged in N3 file. Correct creator and paper names printed. No lines duplicated in output (no creator + paper occurring more than once). Reification data in N3 file. Screen captures of all screen outputs in .docx file (note that fraudulent screen captures earn the entire lab a 0 grade).

15% - Exercise 2 – Worksheet in allowable format. Correct answers.