Concordia University
Engineering and Computer Science

COMP 474/6741 Intelligent Systems (Winter 2024)

Project Assignment #1

Submitted by:
Manish Gautam(40191770)
Lakshya Kalia(40220721)

Index:

Group Details:

Group Name- AK_G_02

Member Details:

Member Name	Student ID	Member's Specialization	Github Username
Manish Gautam	40191770	Python Programming, SQL, Data Structures and Algorithms.	manish198
Lakshya Kalia	40220721	Hands-on experience with python coding and full-stack development.	lakshyakalia

Github Link:

https://github.com/lakshyakalia/comp6741_IS_Roboprof

Undertaking:

“We certify that this submission is the original work of members of the group and meets the Faculty’s Expectations of Originality”

Sign:	Lakshya Kalia	Sign:	Manish Gautam
Student ID:	40220721	Student ID:	40191770
Date:	22/03/2024	Date:	22/03/2024

Vocabulary:

The project's base vocabulary consists of various distinct classes and properties as described in the project assignment #1. Along with the predefined classes and properties, we had to create more custom classes and properties to extend the functionality of the Knowledge Base as well as simplify the queries.

The following classes were developed along with their properties-

1) Class University
    a) offersCourse - Used to link University with courses that it offers.
2) Class Course
    a) courseName - Contains the name of the course.
    b) courseSubject - Contains the alphabetical prefix in COMP6741.
    c) courseNumber - Contains the numerical suffix in COMP6741.
    d) courseCredit - Contains the weightage of the course.
    e) courseDescription - Contains description of the course.
    f) hasLecture - Contains the lecture object.
3) Class Lecture
    a) lectureNumber - Contains count for the Lecture.
    b) lectureName - Contains the name for the lecture.
    c) Link - Contains links to web pages with lecture information.
    d) hasTopicsCovered - Contains a specific topic object that is covered in a particular lecture.
    e) hasLectureContent - Contains objects for reading material, slides and labs.
4) Class Topic
    a) TopicName - Represents the name of the topic.
    b) TopicProvenance - Contains information about where a topic was covered in course.
    c) TopicLink - Links for containing topic links to DBpedia and WikiData
5) Class Student
    a) studentName - Contains name of the student.
    b) studentIDNumber - Contains the ID of the student.

c) studentEmail - Contains email of the student.
d) got_grade_A - Contains information if the student got grade A in a particular course.
e) got_grade_B - Contains information if the student got grade B in a particular course.
f) got_grade_C - Contains information if the student got grade C in a particular course.
g) got_grade_D - Contains information if the student got grade D in a particular course.
h) hasCompletedCourse - Contains the object for courses which the student has completed.
i) competencies - set of topics that the student has completed.

6) Class lectureContent - Parent class for Slides, Worksheets, otherMaterials, readingMaterials Class
7) Class Slides - Class for the lecture slides.
8) Class Worksheets - Class for the worksheets covered in class.
9) Class otherMaterials - Class for otherMaterials such as videos.
10) Class readingMaterials - Class for files containing reference for the content covered in the lecture.

Knowledge Base Construction:

Our dataset comprises of 2 parts-

1) Self-generated data: Data items consists of triples for the following-
   a) Topics:

```
ex:Deep_Learning a ex:Topics;
   rdfs:label "DL";
   rdfs:comment "Deep Learning Topic";
   ex:TopicName "Deep Learning";
   .
ex:Dynamic_Programming a ex:Topics;
```

```
    rdfs:label "DP";
    rdfs:comment "DP Topic";
    ex:TopicName "Dynamic Programming";
    .
```

b) Lectures:

```
ex:AI_Lecture_One a ex:Lecture;
    rdfs:label "AI lecture One";
    rdfs:comment "AI lecture One";
    ex:lectureName "AI Lecture One";
    ex:lectureNumber "1"^^xsd:integer;
    ex:hasTopicsCovered ex:Deep_Learning;
    ex:hasTopicsCovered ex:Dynamic_Programming;
        ex:hasLectureContent
aai_reading_materials:AAI_Reading_Material_01;
        ex:hasLectureContent aai_slides:01_intro_to_ai_and_history;
        ex:hasLectureContent
aai_labs:COMP6721_AI_Lab_1_Winter2024;
    .

ex:IS_Lecture_One a ex:Lecture;
    rdfs:label "IS lecture One";
    rdfs:comment "IS lecture One";
    ex:lectureName "IS Lecture One";
    ex:lectureNumber "1"^^xsd:integer;
    ex:hasTopicsCovered ex:Dynamic_Programming;
    ex:hasLectureContent isr_slides:IS_Slide_01;
    ex:hasLectureContent isr_labs:IS_Lab_01;
    ex:hasLectureContent isr_tutorials:IS_Worksheet_01;
    rdfs:seeAlso <http://concordia.ca/courses/comp6741/lecture_one>;
    .
```

c) Student:

```
ex:Manish_Gautam a ex:Student;
    ex:studentName "Manish Gautam";
    ex:hasCompletedCourse ex:INTELLIGENT_SYSTEMS;
    ex:hasCompletedCourse ex:APPLIED_ARTIFICIAL_INTELLIGENCE;
    ex:got_grade_A ex:INTELLIGENT_SYSTEMS;
    ex:got_grade_B ex:APPLIED_ARTIFICIAL_INTELLIGENCE;
    ex:competencies ex:Deep_Learning;
    ex:competencies ex:Dynamic_Programming;
    .
```

d) Lecture Content: Consists of slides, labs, tutorials and reading materials

```
isr_slides:IS_Slide_01 a ex:Slides;
    rdfs:label "IS_slide_01";
    rdfs:comment "Intelligent System Slide 01";
    .

isr_labs:IS_Lab_01 a ex:otherMaterials;
    rdfs:label "IS_Lab_01";
    rdfs:comment "Intelligent System Lab 01";
    .
```

2) Script-generated Data: Consists of data generated by using the python script on CU_SR_OPEN_DATA_CATALOG.csv to get a list of course data.

```
ex:APPLIED_ARTIFICIAL_INTELLIGENCE a ex:Course;
        ex:courseName "APPLIED ARTIFICIAL INTELLIGENCE";
        rdfs:label "AAI";
        rdfs:comment "Vocabulary for AAI course";
        ex:courseSubject "COMP";
        ex:courseNumber "6721"^^xsd:integer;
        ex:courseCredit "4.00"^^xsd:decimal;
        ex:hasLecture ex:AI_Lecture_One;
```

ex:courseDescription "LEC,Never Taken/Not Registered: COMP472";

.

ex:INTELLIGENT_SYSTEMS a ex:Course;

ex:courseName "INTELLIGENT SYSTEMS";

rdfs:label "IS";

rdfs:comment "Vocabulary for IS course";

ex:courseSubject "COMP";

ex:courseNumber "6741"^^xsd:integer;

ex:courseCredit "4.00"^^xsd:decimal;

ex:hasLecture ex:IS_Lecture_One;

ex:hasLecture ex:IS_Lecture_Two;

ex:courseDescription "LEC, Never Taken/Not Registered: COMP474";

a) Python code used for converting csv to triples: (triple_converter.py)

```
triple_string = ''

# Course Identifier
triple_string = triple_string + "ex:" + triple_identifier + " a ex:Course;\n"
# Course Name
triple_string = triple_string + '\tex:courseName "' + row[3] + '";\n'
# Course Label
triple_string = triple_string + '\trdfs:label "' + course_initial_join + '";\n'
# Course comment
triple_string = triple_string + '\trdfs:comment "Vocabulary for ' + course_initial_join + ' course";\n'
# Course subject
triple_string = triple_string + '\tex:courseSubject "' + row[1] + '";\n'
# Course Number
triple_string = triple_string + '\tex:courseNumber "' + row[2] + '"^^xsd:integer;\n'
# Course credit
triple_string = triple_string + '\tex:courseCredit "' + row[4] + '"^^xsd:integer;\n'
# Course Description/Pre-requisite
triple_string = triple_string + '\tex:courseDescription "' + row[5] + '";\n\t.'
```

To run: Go to csv_to_triple_converter folder and run triple_converter.py.

b) Python code for Knowledge Base Construction: (knowledge_base_construction.py)

```python
import rdflib

g = rdflib.Graph()

g.parse("vocabulary/schema.ttl", format="turtle")

for s,p,o in g:
    # Print the subject, predicate and the object
    print (s,p,o)
```

To run: Go to the knowledge_base_construction folder and run knowledge_base_construction.py.

Output:

```
http://dbpedia.org/resource/Concordia_University http://www.w3.org/1999/02/22-rdf-syntax-ns#type http://example.org/University
http://example.org/courseNumber http://www.w3.org/2000/01/rdf-schema#label courseNumber
http://example.org/got_grade_B http://www.w3.org/2000/01/rdf-schema#label B
http://example.org/got_grade_A http://www.w3.org/2000/01/rdf-schema#range http://www.w3.org/2000/01/rdf-schema#Course
http://example.org/got_grade_A http://www.w3.org/1999/02/22-rdf-syntax-ns#type http://www.w3.org/2000/01/rdf-schema#Property
http://example.org/lectureName http://www.w3.org/2000/01/rdf-schema#range: http://www.w3.org/2000/01/rdf-schema#Literal
http://example.org/TopicName http://www.w3.org/1999/02/22-rdf-syntax-ns#type http://www.w3.org/2000/01/rdf-schema#Property
http://example.org/TopicLink http://www.w3.org/2000/01/rdf-schema#label Link to DBpedia
http://example.org/studentEmail http://www.w3.org/2000/01/rdf-schema#domain http://example.org/Student
http://example.org/competencies http://www.w3.org/2000/01/rdf-schema#domain http://example.org/Student
http://example.org/hasCompletedCourse http://www.w3.org/2000/01/rdf-schema#comment Courses completed by student
http://example.org/Slides http://www.w3.org/2000/01/rdf-schema#label lectureSlides
http://example.org/courseDescription http://www.w3.org/2000/01/rdf-schema#comment description for a course
http://example.org/hasLectureContent http://www.w3.org/2000/01/rdf-schema#comment All the conent for a particular course
http://example.org/courseSubject http://www.w3.org/2000/01/rdf-schema#range http://www.w3.org/2000/01/rdf-schema#Literal
http://example.org/University http://www.w3.org/1999/02/22-rdf-syntax-ns#type http://www.w3.org/2002/07/owl#Class
http://example.org/hasLecture http://www.w3.org/2000/01/rdf-schema#range http://example.org/Lecture
http://example.org/courseOutline http://www.w3.org/2000/01/rdf-schema#label courseOutline
http://example.org/courseCredit http://www.w3.org/2000/01/rdf-schema#domain http://example.org/Course
http://example.org/TopicLink http://www.w3.org/2000/01/rdf-schema#domain http://example.org/Topics
```

Graph Queries

To get knowledge out of the knowledge graph, the graph was converted to a graph database. Graph Database stores every knowledge as triples. The triple consists of subject predicate and object. The database was queried using SPARQL queries to get information about the data from the database. Queries and their are listed below:

1. List all courses offered by [university].

Query:

```
SELECT DISTINCT ?courseName WHERE {
dbpedia:Concordia_University ex:offersCourse ?object.
?course ex:courseName ?courseName
}
```

Output:

Table | Response | 3955 results in 0.253 seconds | Simple view ☐ Ellipse ☑ | Filter query results | Page size: 50

	courseName
1	RESRCH PROPOSAL & QUAL EXAM
2	SUSTAINABLE RESOURCE MANAGEM
3	Special Topics in Computation Arts
4	SEL TOPS IN HIST OF GENDER
5	SEL TOPS IN CANADIAN HIST
6	TOPICS IN FILM DIRECTORS
7	Interdisciplinary Topics: Ethics in Research in the Creative Arts
8	SPECIAL TOPICS IN CERAMICS
9	Selected Topics in Applied Human Sciences
10	Methods in Film and Moving Image Studies I

2. In which courses is [topic] discussed?

Query:

```
SELECT DISTINCT ?courseName WHERE{
?courses ex:courseName ?courseName.
?courses ex:hasLecture ?lecture.
?lecture ex:hasTopicsCovered ex:Dynamic_Programming.
}
```

Output:

Table | Response | 2 results in 0.034 seconds | Simple view ☐ Ellipse ☑ | Filter query results | Page size: 50

	courseName
1	APPLIED ARTIFICIAL INTELLIGENCE
2	INTELLIGENT SYSTEMS

Showing 1 to 2 of 2 entries < 1 >

3. Which [topics] are covered in [course] during [lecture number]?

Query:

```
SELECT ?topic_name WHERE{
?course ex:courseName "APPLIED ARTIFICIAL INTELLIGENCE".
?course ex:hasLecture ?lecture.
?lecture ex:lectureNumber ?lectureNumber.
?lecture ex:hasTopicsCovered ?topics.
?topics ex:TopicName ?topic_name.
FILTER(?lectureNumber=1)
}
```

Output:

Table | Response | 2 results in 0.114 seconds | Simple view ☐ Ellipse ☑ | Filter query results | Page size: 50

	topic_name
1	Dynamic Programming
2	Deep Learning

Showing 1 to 2 of 2 entries < 1 >

4. List all [courses] offered by [university] within the [subject] (e.g., “COMP”, “SOEN”).

Query:

```
SELECT DISTINCT ?courseName WHERE {
dbpedia:Concordia_University ex:offersCourse ?object.
?course ex:courseSubject ?subject.
?course ex:courseName ?courseName
FILTER(?subject="COMP"^^xsd:string)
}
```

Output:

Table | Response | 109 results in 0.068 seconds | Simple view ☐ Ellipse ☑ | Filter query results | Page size: 50

	courseName
1	Digital Geometric Modelling
2	COMP.ANAL.NATURAL LANG.TEST
3	ADVANCED COMPUTER GRAPHICS
4	MASTER S RESEARCH AND THESIS
5	DOCTORAL RESEARCH AND THESIS
6	Pattern Recognition

5.What [materials] (slides, readings) are recommended for [topic] in [course] [number]?

Query:

```
SELECT DISTINCT ?content_label WHERE{
```

?course ex:courseSubject ?course_subject.

?course ex:courseNumber ?course_number.

?course ex:hasLecture ?lecture.

?lecture ex:hasTopicsCovered ?topic.

?topic ex:TopicName ?topic_name.

?lecture ex:hasLectureContent ?lecture_content.

?lecture_content rdfs:label ?content_label.

FILTER(?course_subject='COMP' && ?course_number=6741 && ?topic_name='Deep Learning')

}

Output:

Table | Response | 3 results in 0.108 seconds | Simple view ☐ Ellipse ☑ | Filter query results | Page size: 50

	content_label
1	IS_slide_02
2	IS_Lab_02
3	IS_Worksheet_02

Showing 1 to 3 of 3 entries < 1 >

6.How many credits is [course] [number] worth?

Query: SELECT ?credit WHERE{

?course ex:courseName ?name.

?course ex:courseCredit ?credit.

FILTER(?name="INTELLIGENT SYSTEMS")

}

Output

Table | Response | 1 result in 0.052 seconds | Simple view ☐ Ellipse ☑ | Filter query results | Page size: 50

	credit
1	"4.00"^^<http://www.w3.org/2001/XMLSchema#integer>

Showing 1 to 1 of 1 entries < 1 >

7. For [course] [number], what additional resources (links to web pages) are available?

Query: SELECT DISTINCT ?lecture_label ?webpages WHERE{

?course ex:courseSubject ?course_subject.

?course ex:courseNumber ?course_number.

?course ex:hasLecture ?lecture.

?lecture rdfs:label ?lecture_label.

?lecture rdfs:seeAlso ?webpages.

FILTER(?course_subject='COMP' && ?course_number=6741)

}

Table | Response | 2 results in 0.092 seconds | Simple view ☐ Ellipse ☑ | Filter query results | Page size: 50

	lecture_label	webpages
1	IS lecture One	<http://concordia.ca/courses/comp6741/lecture_one>
2	IS lecture Two	<http://concordia.ca/courses/comp6741/lecture_two>

Showing 1 to 2 of 2 entries < 1 >

8. Detail the content (slides, worksheets, readings) available for [lecture number] in [course] [number].

Query: SELECT DISTINCT ?content_label WHERE{

?course ex:courseSubject ?course_subject.

?course ex:courseNumber ?course_number.

?course ex:hasLecture ?lecture.

?lecture ex:hasTopicsCovered ?topic.

?lecture ex:hasLectureContent ?lecture_content.

?lecture_content rdfs:label ?content_label.

?lecture ex:lectureNumber ?lecture_number.

FILTER(?course_subject='COMP' && ?course_number=6741 && ?lecture_number=1)

}

Output:

Table | Response | 3 results in 0.066 seconds | Simple view ☐ Ellipse ☑ | Filter query results | Page size: 50

	content_label
1	IS_Lab_01
2	IS_Worksheet_01
3	IS_slide_01

Showing 1 to 3 of 3 entries < 1 >

9. What reading materials are recommended for studying [topic] in [course]?

Query: SELECT DISTINCT ?content_label WHERE{

?course ex:hasLecture ?lecture.

?lecture ex:hasTopicsCovered ?topic.

?topic ex:TopicName ?topic_name.

?lecture ex:hasLectureContent ?lecture_content.

?lecture_content a ex:readingMaterials.

?lecture_content rdfs:label ?content_label.

FILTER(?course=ex:APPLIED_ARTIFICIAL_INTELLIGENCE && ?topic_name='Deep Learning')

}

Output:

Table | Response | 1 result in 0.113 seconds | Simple view ☐ Ellipse ☑ | Filter query results | Page size: 50

	content_label
1	AAI_Reading_Material_01

Showing 1 to 1 of 1 entries < 1 >

10. What competencies [topics] does a student gain after completing [course] [number]?

Query:

```
SELECT DISTINCT ?topic_name WHERE{
?course ex:courseSubject ?course_subject;
ex:courseNumber ?course_number.
?student ex:hasCompletedCourse ?course.
?student ex:competencies ?competencies.
?competencies ex:TopicName ?topic_name.
FILTER(?course_subject="COMP" && ?course_number=6741)
}
```

Output:

Table | Response | 2 results in 0.08 seconds | Simple view ☐ Ellipse ☑ | Filter query results | Page size: 50

	topic_name
1	Dynamic Programming
2	Deep Learning

Showing 1 to 2 of 2 entries < 1 >

11. What grades did [student] achieve in [course] [number]?

Query:

```
SELECT DISTINCT ?name ?course_label ?grade_label  WHERE{
?student ex:studentName ?name.
```

```
?student ex:hasCompletedCourse ?course.
?course rdfs:label ?course_label.
?course ex:courseNumber ?course_number.
?student ?relation ?course.
?relation rdfs:label ?grade_label.
FILTER(?name="Manish Gautam" && ?relation!=ex:hasCompletedCourse
&& ?course_number=6721)
}
```

Table | Response | 1 result in 0.058 seconds | Simple view ☐ Ellipse ☑ | Filter query results | Page size: 50

	name	course_label	grade_label
1	Manish Gautam	AAI	B

Showing 1 to 1 of 1 entries

< 1 >

12. Which [students] have completed [course] [number]?

Query:

```
SELECT ?student_name WHERE{
?course ex:courseSubject ?course_subject;
ex:courseNumber ?course_number.
?student ex:hasCompletedCourse ?course.
?student ex:studentName ?student_name.
FILTER(?course_subject="COMP" && ?course_number=6741)
}
```

Output:

Table | Response | 3 results in 0.052 seconds | Simple view ☐ Ellipse ☑ | Filter query results | Page size: 50

	student_name
1	Aatish Neupane
2	Lakshya Kalia
3	Manish Gautam

Showing 1 to 3 of 3 entries

< 1 >

13. Print a transcript for a [student], listing all the course taken with their grades.

Query:

```
SELECT DISTINCT ?name ?course_label ?grade_label  WHERE{
?student ex:studentName ?name.
?student ex:hasCompletedCourse ?course.
?course rdfs:label ?course_label.
```

```
?student ?relation ?course.
?relation rdfs:label ?grade_label.
FILTER(?name="Manish Gautam" &&?relation!=ex:hasCompletedCourse)
}
```

Output:

19

Table | Response | 2 results in 0.05 seconds | Simple view ☐ Ellipse ☑ | Filter query results | Page size: 50

	name	course_label	grade_label
1	Manish Gautam	IS	A
2	Manish Gautam	AAI	B

Showing 1 to 2 of 2 entries

< 1 >

Triplestore and SPARQL Endpoint Setup

We have used Apache Jena TDB which is a high performance RDF database[1]. Apache Fuseki is a server which provides a web interface for the database and facilitates in creating a HTTP endpoint to the graph database using SPARQL query. Fuseki Server requires a JDK 11. After installing the JDK and then the Fuseki Server the server can be launched by running the script ./fuseki-server and the default port the server runs is the 3030.

After the server has been launched, we can set up the database by importing our knowledge base file within the database server. After importing the knowledge base, we queried the database using SPARQL endpoints in the query section of the server interface.

Since we have JDK version 8 in our system, we ran into a problem for some time and after that we installed the JDK11 and updated the path for JDK11 for JAVA_HOME. After that it started running smoothly.

References:

1. https://jena.apache.org/documentation/tdb/index.html