

COMP 474/6741

Intelligent Systems

ROBOPROF: AN INTELLIGENT AGENT

Team Name: AZ_G_03

Team Members:

1. Sharanyu Pillai, ID: 40227794, Specialization: Python, SPARQL, Apache Fuseki, Rasa, rasa actions, REST API
2. Sushant Sharma, ID: 40227986, Specialization: Python, SPARQL, Apache Fuseki, Rasa, rasa actions, REST API

Project Repository: https://github.com/Sharanyu/RoboProf-An-Intelligent-Agent-Group-AZ_G_03-COMP_6741_Winter_2024

"We certify that this submission is the original work of members of the group and meets the Faculty's Expectations of Originality."

Signatures:

1. Sharanyu Pillai , ID: 40227794, Date: 15/04/2024

2. Sushant Sharma , ID: 40227986, Date: 15/04/2024

Contents

Introduction	3
Section 1: Vocabulary Design	4
1.1 Classes and Properties Overview	4
1.2 Reused Vocabularies	4
1.3 Justification for Vocabulary Choices	5
Section 2: Knowledge Base Construction.....	6
2.1 Detailed Dataset Overview.....	6
2.2 Processes and Development Tools.....	7
2.3 Execution Guidelines	8
Section 3: Graph Queries- Translation Overview	11
Section 4: SPARQL Endpoint Setup	16
4.1 Selection of Triplestore	16
4.2 Setup Process	16
4.3 Data loading	18
PART#2.....	21
Section 5: Report Updates	21
5.1 RDF Schema Structural Advancements	21
5.2 Automating KB update in script.....	21
5.3 Vocabulary File and Ontology Precision	21
Section 6: Knowledge Base Population	22
Section 7: Chatbot Design.....	25
7.1 Integrating New Intents and Conversation Flows	25
7.2 Connecting to Fuseki Server for Data Retrieval.....	27
7.3 Managing Unrecognized Words	28
7.4 ChatBot configuration (config.yml)	28
7.5 Strategy for Error Handling	29
7.6 Generating User-Friendly Responses	29
Section 8: Chatbot interaction: Prompts and responses.....	30

Introduction

In the ever-evolving landscape of educational technology, the quest for innovative solutions to enhance learning experiences remains paramount. The Roboprof project, undertaken by our Team AZ_G_03, embodies this pursuit by leveraging the powerful synergies of Knowledge Graphs and Natural Language Processing (NLP) to transform the accessibility and interaction with educational content. This initiative aims to construct an intelligent agent capable of answering intricate university course- and student-related queries, thereby facilitating a more engaged and informed academic community at Concordia University.

Roboprof is designed to model the educational domain in an end-to-end way with a bespoke ontology and by populating a knowledge graph with all details around courses, lectures, and students. The mainstay lies in the capability to semantically link various pieces of academic data, thereby enabling combined query capabilities and insights heretofore allowed only clumsily. We began with the development of a comprehensive vocabulary, crafted to encapsulate the nuances of educational materials, and student records. This vocabulary, coupled with an extensive dataset comprising course descriptions, lecture materials, and student records, laid the groundwork for constructing the knowledge graph. Subsequent phases involved the integration of this dataset into the graph, facilitated by custom Python scripts and the strategic use of RDF technologies.

The ultimate manifestation of our efforts is a triplestore powered by Apache Jena Fuseki, serving as the repository for our RDF data, and a SPARQL endpoint enabling web-based queries. This setup not only exemplifies the project's technical achievements but also signifies a step towards democratizing access to educational information. This report covers the Roboprof project's journey from conceptualization to implementation. It details the challenges encountered, the solutions devised, and the knowledge gained, offering a comprehensive overview of our endeavor to enhance the educational landscape through the power of semantic technologies and intelligent systems.

Section 1: Vocabulary Design

The Roboprof project's ontology, delineated within the 'vocabulary.ttl' file, serves as the structural backbone of our knowledge graph. It was meticulously crafted to model the educational domain, particularly focusing on courses, lectures, topics, and student information at Concordia University. This section explores the various elements of the ontology, including the reuse of established vocabularies and the introduction of project-specific extensions.

1.1 Classes and Properties Overview

1. **Lecture:** Central to the educational experience, lectures are modeled as a foundational class (roboprof:Lecture). This class captures the essence of course lectures, which are further detailed through properties like lectureNumber, lectureName, content, enabling precise identification and description of each lecture.
2. **Lecture Content Subclasses:** To encapsulate the diverse materials associated with lectures, properties such as roboprof:Slides, roboprof:Worksheets, and roboprof:Readings were defined under the broader roboprof:LectureContent class. This hierarchy allows for a nuanced representation of lecture resources, catering to the varied learning modalities present in academia.
3. **Lab:** To capture the Lab events of a course, Lab class is created which has lab title, lab number, lab contents. Lab is then connected to the course using containsLab property.
4. **Topics:** The roboprof:Topics class signifies the educational themes covered in lectures or courses. Each topic is uniquely identified (topicName) and linked to external knowledge bases (externalLink), such as DBpedia or Wikidata, enriching the graph with external semantic relationships. The provenance property records the origin of the topic's identification, ensuring traceability and context.
5. **Student Information:** Reflecting on the student-centric nature of educational data, the roboprof:Student class extends the FOAF vocabulary's Person class. It encompasses properties for student IDs (idNumber), completed courses (completedCourses), and grades (grade), facilitating a comprehensive model of student academic records and competencies.

1.2 Reused Vocabularies

The design leverages well-established vocabularies to ensure compatibility and interoperability:

1. **FOAF, RDFS and RDF:** Utilized for basic class and property definitions, enabling the integration of our schema within the broader semantic web. FOAF was used for Student class.

2. **Schema.org (schema:), purl.org (DBCORE and DCTERMS) and DBpedia Ontology (dbpedia_o:)**: These vocabularies are instrumental in defining and linking courses to the university, enhancing the ontology's expressiveness with widely recognized educational and organizational terms.
3. **Custom Extensions:** Custom extensions like “**roboprof**” were developed to fill gaps not covered by existing vocabularies, specifically tailored to the project's needs. Detailed modeling of lecture materials and student information showcases the project's focus on educational content and student achievement metrics. The introduction of properties like lectures and courses directly ties courses to their constituent lectures and the university offering them, reflecting the hierarchical nature of academic course structures.
4. **Tailored Vocabulary Extensions:** To address the unique requirements of the educational domain that are not covered by the used vocabularies, the schema introduces a series of custom classes and properties under the **roboprof** namespace. This includes:
 - a. **Domain-Specific Classes:** Such as Lecture, Lab, Topic, Student, and CourseAttempt, which are significant for a detailed representation of the Concordia ecosystem encompassing lectures, coursework, and student achievements.
 - b. **Educational Properties:** Including Lecture Content (Slides, Readings, Worksheets). containsLecture, containsLab, numberOfCredits, among others, enabling a comprehensive depiction of course materials, lecture content, and academic credits.

1.3 Justification for Vocabulary Choices

The decision to extend certain vocabularies and introduce specific classes and properties was driven by the project's aim to create a highly detailed and navigable educational knowledge graph. We also consulted our professor and the teaching assistant during the development of the vocabulary.

Custom extensions were necessary to accurately represent the complex relationships within academic data, such as the linkage between students, their courses, and associated lecture materials. Meanwhile, the reuse of established vocabularies ensures that our ontology remains grounded in standard semantic web practices, facilitating interoperability and data integration.

This carefully crafted ontology serves as the foundation for querying the knowledge graph, enabling insightful analyses, and supporting the project's overarching goal of enhancing the educational experience.

Section 2: Knowledge Base Construction

2.1 Detailed Dataset Overview

The Roboprof knowledge graph is rooted in a comprehensive dataset curated to reflect the diverse academic landscape of Concordia University. This dataset is pivotal for structuring the knowledge graph with genuine and detailed course information, encompassing:

1. **Open Data on Courses:** Utilizing the open data catalog from Concordia University (<https://opendata.concordia.ca/datasets/>), the project imports a wealth of course information, including course IDs, subjects, catalog numbers, titles, credit units, and components. This dataset is essential for detailing the academic offerings within the knowledge graph.

The files we Roboprof uses :

- [CU_SR_OPEN_DATA_CATALOG.csv](#)
- [CU_SR_OPEN_DATA_CATALOG_DESC.csv](#)

We cleaned and merged these two datasets to obtain a Course_info dataframe using pandas for feeding the graph with course and its descriptions details. data_loader.py handles this transformation.

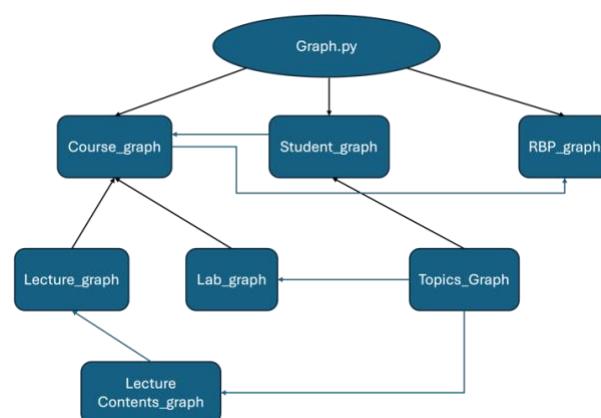
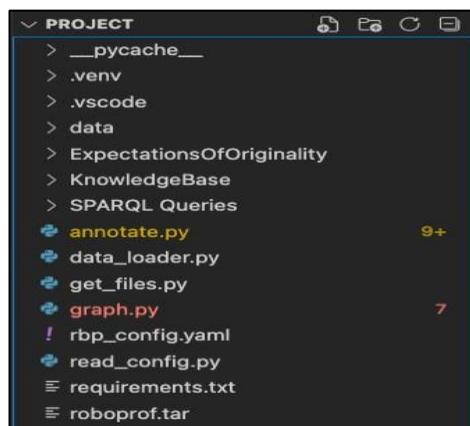
2. **Lecture Materials:** A systematically organized collection of PDF files, categorized by course and type of material (e.g., slides, worksheets, labs, assignments), represents the content delivered across courses. These materials serve as the foundation for linking educational content to relevant courses and topics within the graph. It was manually created in local to simulate a file server environment. These files are created with local (file://) prefix so that it can be accessed.
3. **Spotlight Annotations:** The lecture materials content is extracted and parsed using Apache Tika, cleaned, filtered using **spacy** and fed into the dbpedia spotlight open API <https://api.dbpedia-spotlight.org/en/annotate>. The URIs are saved into a csv file called topic_info.csv. This file will help us in populating the topics information for each course and lecture and add competencies to the student instances.
4. **Student Records:** This dataset includes sample student information capturing their name, student_id, Email, Subject, Code, Semester, Grade. It outlines the courses each student has completed along with their respective grades, aiding in modeling student competencies based on the covered topics. Most of the Student graph was modelled using FOAF.

5. **rbp_config.yaml**: A configuration file which stores the folder paths and hardcoded lists and details for easier debugging and single point of modification.

2.2 Processes and Development Tools

Initial Data Preparation

- **Merging Course Data:** The **data_loader.py** script plays a crucial role in combining course information from Concordia's open data catalog with detailed descriptions, enriching the dataset with both structural and descriptive aspects of each course.
- **Resource Listing and Categorization:** Employing **get_files.py**, this script navigates the structured directory in the local system of course materials to identify and categorize PDF files by their relevance to courses, crucial for organizing materials for subsequent annotation and integration.
- **Content Extraction and Semantic Annotation:** The **annotate.py** script utilizes Apache Tika for text extraction from PDF files and Spacy to filter the named entities from the extracted text and DBpedia Spotlight for semantic annotation. This process is integral for identifying the topics covered in each material, facilitating the dynamic linkage of educational content to the broader knowledge graph. The **annotate.py** utilizes Open API of DBPEDIA spotlight service to fetch the dbpedia resources. This script outputs **topic_info.csv** which can be found in **data/spotlight_data/** folder. This script needs to be executed only once per implementation.
- **Knowledge Base Creation:** The **graph.py** is the main script which utilizes the above created data and methods to build the knowledge base. We generate 5 graphs to generate the KB.

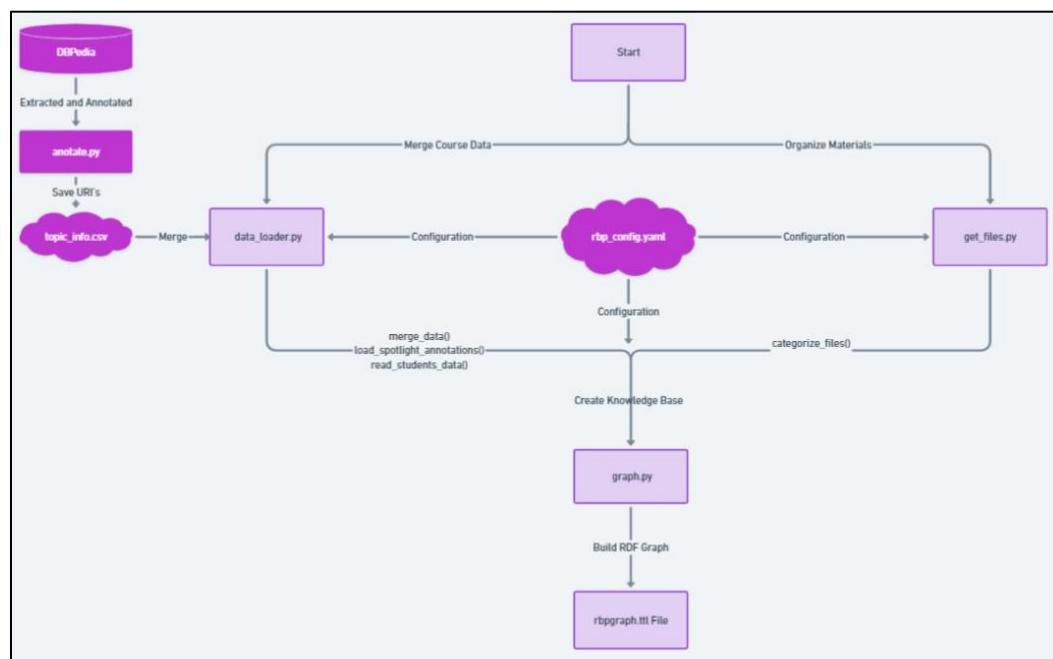


1. Course graph – graph to store the course details.
2. Lecture graph – graph to store the lecture details of a course.
3. Lecture_content graph – graph to store the lecture contents like readings, slides, worksheets of a lecture.
4. Lab graph – graph to store the lab details of a course.
5. Topic graph – graph to store the topics details from each lecture content fetching data from dbpedia.
6. Student graph – graph to store the student grade sheets.

7. RBP_graph – main graph where other graphs get merged and as a result the main rbp_graph.ttl file is generated.

“rbpgraph.ttl” Generation

The peak of the knowledge base construction process is the generation of the rbpgraph.ttl file. This Turtle (TTL) file encapsulates the Roboprof knowledge graph, representing a comprehensive RDF model of the academic offerings at Concordia University, annotated lecture materials, and student academic records. The rbpgraph.ttl file is the artifact of integrating structured and unstructured data sources through meticulous RDF modeling, spotlight annotation, and custom Python scripting.

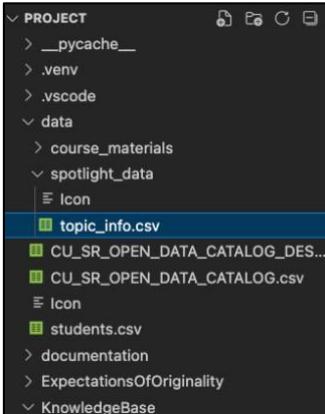


2.3 Execution Guidelines

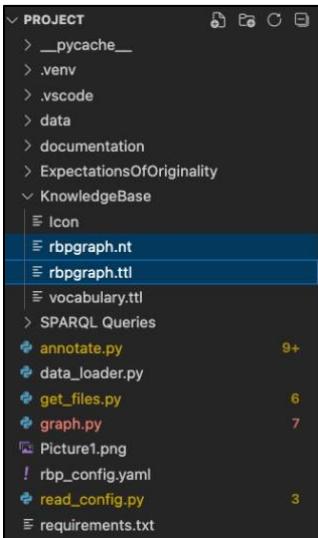
To populate the knowledge base, execute the Python scripts sequentially, reflecting their dependencies:

General Instructions:

1. Execute **annotate.py**. (This is a one-time execution course, execute this course when you have new lecture contents in the data folder.) In the terminal, go inside the project folder and type “python3 annotate.py” and press enter. This file will generate the topics from Dbpedia(**topic_info.csv**) inside the **data/spotlight_data/** folder.



2. Execute graph.py – This file generates the rbpgraph.ttl and rbpgraph.nt inside the KnowledgeBase folder. In the terminal, in the same project directory, type “python3 graph.py” and press enter. The rbpgraph files will be generated inside KnowledgeBase folder like below.



Note: If the files are already existing in the project, you can delete them and execute it again or execute it as it is as the files will be overwritten.

To Run the application in your Local PC:

- a) Extract the zipfile and open the project folder in IDE.
- b) Open the project in a terminal or IDE.
- c) Now run “python3 annotate.py” - This step is also optional as we have a topic_info.csv in the data folder already.
- d) Next run “python3 graph.py”.

```
root@814bd03962f5:/app# ls
Dockerfile      'SPARQL Queries'   data      graph.py      requirements.txt
ExpectationsOfOriginality __pycache___.py    data_loader.py  rbp_config.yaml
KnowledgeBase   annotate.py       get_files.py  read_config.py
root@814bd03962f5:/app# python3 annotate.py
root@814bd03962f5:/app# python3 graph.py
/usr/local/lib/python3.12/site-packages/rdflib/plugins/serializers/nt.py:40: UserWarning: NTSerializer always uses UTF-8
encoding. Given encoding was: None
  warnings.warn(
root@814bd03962f5:/app#
```

- e) Navigate to the KnowledgeBase folder to check if the rbpgraph files are generated.

```

root@814bd03962f5:/app# ls -lrt
total 68
drwx----- 3 root root 4096 Mar 22 01:24 'SPARQL Queries'
drwxr-xr-x 2 root root 4096 Mar 22 13:03 ExpectationsOfOriginality
-rw-r--r-- 1 root root 231 Mar 22 13:15 read_config.py
-rw----- 1 root root 2028 Mar 22 13:30 get_files.py
drwxr-xr-x 1 root root 4096 Mar 22 13:35 data
-rw----- 1 root root 2996 Mar 22 13:36 annotate.py
-rw-r--r-- 1 root root 1379 Mar 22 13:37 data_loader.py
drwxr-xr-x 2 root root 4096 Mar 22 13:38 __pycache__
drwxr-xr-x 1 root root 4096 Mar 22 13:40 KnowledgeBase
-rw-r--r-- 1 root root 818 Mar 22 13:47 requirements.txt
-rw-r--r-- 1 root root 417 Mar 22 14:31 rbp_config.yaml
-rw----- 1 root root 8700 Mar 22 14:32 graph.py
-rw-r--r-- 1 root root 118 Mar 22 15:00 Dockerfile
root@814bd03962f5:/app# cd KnowledgeBase/
root@814bd03962f5:/app/KnowledgeBase# ls -lrt
total 14068
-rw-r--r-- 1 root root 4145 Mar 22 11:38 vocabulary.ttl
-rw----- 1 root root 0 Mar 22 13:03 'Icon'$'\r'
-rw-r--r-- 1 root root 4808620 Mar 22 18:49 rbpgraph.ttl
-rw-r--r-- 1 root root 9584729 Mar 22 18:49 rbpgraph.nt
root@814bd03962f5:/app/KnowledgeBase#

```

f) Upload the rbpgraph.ttl to the fuseki server and name the dataset as "roboprof". Make sure the fuseki server is running for the next processes.

Next, to interact with the chatbot.

g) Open a new terminal in the IDE and run "rasa train" to create a model file. There should already be a model file existing in the models folder, just in case.

h) In the same terminal in the IDE, run "rasa shell" to start the rasa shell. meanwhile create another terminal parallelly.

i) Once the rasa shell command execution is complete, type "rasa run actions" in the newly created terminal.

j) Once the rasa run actions command is executed and the rasa actions server starts, Switch back to the previous terminal, you should now see a yellow arrow prompt icon to start typing.

k) Start with "Hi bot, How are you?" to check if bot is working.

l) Now you can start typing the project related prompts. Example an easy one. "List all the courses provided."

If you face any issues with the file paths, you can update the rbp_config.yaml based on your system.

Section 3: Graph Queries- Translation Overview

The translation process for each query involves identifying the key entities and relationships implied by the question and then crafting a SPARQL query that leverages the RDF schema's structure to retrieve the desired information. This involves selecting appropriate classes and properties, applying filters and joins as necessary, and sometimes aggregating or transforming data to match the query's specific output requirements.

Query 1: List all courses offered by [university]

To list all courses offered by a specified university, the query targets the **schema:Course** class and filters instances where the **schema:provider** matches the specified university's URI. This direct mapping from the question to SPARQL leverages RDF's triple pattern to intuitively associate courses with their offering institution.

Output: Returns a list of courses, including their URIs and titles, offered by the university.

course	courseTitle
1 <http://ROBOPROF.org/data#CIV212>	CIVI 212: Civil Engineering Drawing and Introduction to Design
2 <http://ROBOPROF.org/data#MSCA689>	MSCA 689: SEMINAR IN OPERATIONS MANA II
3 <http://ROBOPROF.org/data#MUS3491>	MUSI 491: Special Project in Music
4 <http://ROBOPROF.org/data#REL4140>	RELI 410: Honours Thesis
5 <http://ROBOPROF.org/data#PSYC803>	PSYC 803: SPECIAL TOPIC SEMINAR
6 <http://ROBOPROF.org/data#ENGL6089>	ENGL 608: Studies in Early English Literature and Medieval Literature
7 <http://ROBOPROF.org/data#BC1499>	LBC1 499: ADV TOPICS LBC1
8 <http://ROBOPROF.org/data#APL4691>	APLI 4691: THESIS
9 <http://ROBOPROF.org/data#FLU2230>	FLUZ 230: Introduction aux cultures de la francophonie
10 <http://ROBOPROF.org/data#DFT408>	DFTT 408: Supervised Internship I
11 <http://ROBOPROF.org/data#COMP248>	COMP 248: Object-Oriented Programming I
12 <http://ROBOPROF.org/data#ARTH471>	ARTH 471: SPEC. TOPICS IN ART HISTORY
13 <http://ROBOPROF.org/data#ECON340>	ECON 540: Market Design
14 <http://ROBOPROF.org/data#IND820>	INDI 820: DOCTORAL LEVEL STUDIES
15 <http://ROBOPROF.org/data#AH342>	AHSC 442: COMMUNITY DEVELOPMENT AND LEISURE
16 <http://ROBOPROF.org/data#MATH498>	MATH498: ADVANCED TOPICS IN MATH
17 <http://ROBOPROF.org/data#PSYC243>	PSYC 243: Learning Disabilities
18 <http://ROBOPROF.org/data#CHME6001>	CHME 6001: Project in Chemical and Materials Engineering
19 <http://ROBOPROF.org/data#CERA230>	CERA 230: Ceramics I
20 <http://ROBOPROF.org/data#THE603>	THEO 603: Method in Theology
21 <http://ROBOPROF.org/data#REL1225>	RELI 225: Introduction to Hinduism
22 <http://ROBOPROF.org/data#EDUC381>	EDUC 381: Inclusive Practices in Early Childhood Settings

Query 2: In which courses is [topic] discussed?

The query identifies courses containing lectures or materials discussing a specified topic by querying against the roboprof:Topics class. It utilizes a property path to traverse from topics to courses, capturing the relationship between the educational content and its container course.

Output: Lists courses, by URI and name, that cover the given topic.

course	courseTitle
1 <http://ROBOPROF.org/data#COMP6741>	INTELLIGENT SYSTEMS
2 <http://ROBOPROF.org/data#SOEN6431>	SOFT. COMP.& MAINTENANCE

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

This query extracts topics covered in a particular lecture of a course by specifying the course and lecture number. It involves joining information from lectures to their encompassing courses, filtering by the provided lecture number, and then linking to the covered topics.

Output: Outputs topics covered in the specified lecture, enhancing understanding of lecture content.

Table Response 45 results in 0.025 seconds		Simple view	Ellipsis	Filter query results	Page size: 50	?
topic		topicLabel				
1 <http://ROBOPROF.org/data#Discrete_Fourier_transform>		dft				
2 <http://ROBOPROF.org/data#Unit_vector>		normalized				
3 <http://ROBOPROF.org/data#Ontology>		ontology				
4 <http://ROBOPROF.org/data#Ontology>		ontological				
5 <http://ROBOPROF.org/data#Ontology>		ontologies				
6 <http://ROBOPROF.org/data#Definiteness_of_a_matrix>		matrix				
7 <http://ROBOPROF.org/data#Israel_Defense_Forces>		idf				
8 <http://ROBOPROF.org/data#I_p_space>		l2 norm				
9 <http://ROBOPROF.org/data#Length>		lengths				
10 <http://ROBOPROF.org/data#Mary_Shelley>		mary				

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

To find courses within a specific subject offered by a university, the query combines filters for the university provider and subject category. This approach relies on the semantic relationships defined in the RDF schema to associate courses with subjects and offering institutions.

Output: Generates a list of courses within the specified subject, including course details.

Table Response 114 results in 0.055 seconds		Simple view	Ellipsis	Filter query results	Page size: 50	?
course	courseDetails					
1 <http://ROBOPROF.org/data#COMP248>	COMP 248: Object-Oriented Programming I					
2 <http://ROBOPROF.org/data#COMP474>	COMP 474: Intelligent Systems					
3 <http://ROBOPROF.org/data#COMP665>	COMP 665: ALGORITHM DESIGN TECHNIQUES					
4 <http://ROBOPROF.org/data#COMP6591>	COMP 6591: INTRO/KNOWLEDGE-BASE SYSTEMS					
5 <http://ROBOPROF.org/data#COMP778>	COMP 7781: ADV IMAGE PROC					
6 <http://ROBOPROF.org/data#COMP6661>	COMP 6661: COMBINATORIAL ALGORITHMS					
7 <http://ROBOPROF.org/data#COMP6971>	COMP 6971: Project and Report					
8 <http://ROBOPROF.org/data#COMP5531>	COMP 5531: FILES AND DATABASES					
9 <http://ROBOPROF.org/data#COMP6781>	COMP 6781: COMP 6781 Statistical Natural Language Processing					
10 <http://ROBOPROF.org/data#COMP346>	COMP 346: Operating Systems					
11 <http://ROBOPROF.org/data#COMP5201>	COMP 5201: COMP. ORGANIZ N+ASSEMBL.LANG.					
12 <http://ROBOPROF.org/data#COMP352>	COMP 352: Data Structures and Algorithms					
13 <http://ROBOPROF.org/data#COMP6311>	COMP 6311: ANIMATION FOR COMPUTER GAMES					
14 <http://ROBOPROF.org/data#COMP208>	COMP 208: Computer Science Industrial Experience Reflective Learning II					
15 <http://ROBOPROF.org/data#COMP479>	COMP 479: Information Retrieval and Web Search					
16 <http://ROBOPROF.org/data#COMP438>	COMP 438: GEOMETRIC MODELING AND PROCESSING					
17 <http://ROBOPROF.org/data#COMP7241>	COMP 7241: PARALLEL ALGORITHMS+ARCHIT E					
18 <http://ROBOPROF.org/data#COMP7941>	COMP 7941: MASTER S RESEARCH AND THESIS					
19 <http://ROBOPROF.org/data#COMP6821>	COMP 6821: Bioinformatics Databases and Systems					
20 <http://ROBOPROF.org/data#COMP6231>	COMP 6231: DISTRIBUTED SYSTEM DESIGN					
21 <http://ROBOPROF.org/data#COMP5361>	COMP 5361: DISCRETE STR+FORMAL LANG.					
22 <http://ROBOPROF.org/data#COMP6711>	COMP 6711: COMPUTATIONAL GEOMETRY					
23 <http://ROBOPROF.org/data#COMP367>	COMP 367: Techniques in Symbolic Computation					

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

Focusing on recommended materials for a topic within a course, this query navigates from the course to specific lecture materials tagged with the topic of interest. It demonstrates the graph's ability to link detailed educational resources to broader course structures and topics.

Output: Displays recommended materials for the topic in the course, aiding in study preparation.

Table Response 15 results in 0.058 seconds		Simple view	Ellipsis	Filter query results	Page size: 50	?
course	topicLabel	source				
1 <http://ROBOPROF.org/data#COMP6741>	wikidata	<file:///Users/sharanyu/Desktop/CleanIS/data/course_materials/COMP6741/Labs/Lab05/COMP-474-2234-UU_Lab%2520Session_05_Moodle.pdf>				
2 <http://ROBOPROF.org/data#COMP6741>	wikidata	<file:///Users/sharanyu/Desktop/CleanIS/data/course_materials/COMP6741/Lectures/Lecture03/worksheets/worksheet03.pdf>				
3 <http://ROBOPROF.org/data#COMP6741>	wikidata	<file:///Users/sharanyu/Desktop/CleanIS/data/course_materials/COMP6741/Labs/Lab02/COMP-474-2234-UU_Lab%2520Session_02_Moodle.pdf>				
4 <http://ROBOPROF.org/data#COMP6741>	wikidata	<file:///Users/sharanyu/Desktop/CleanIS/data/course_materials/COMP6741/Labs/Lab04/COMP-474-2234-UU_Lab%2520Session_04_Moodle.pdf>				
5 <http://ROBOPROF.org/data#COMP6741>	wikidata	<file:///Users/sharanyu/Desktop/CleanIS/data/course_materials/COMP6741/Lectures/Lecture01/worksheets/worksheet01.pdf>				
6 <http://ROBOPROF.org/data#COMP6741>	wikidata	<file:///Users/sharanyu/Desktop/CleanIS/data/course_materials/COMP6741/Labs/Lab03/COMP-474-2234-UU_Lab%2520Session_03_Moodle.pdf>				
7 <http://ROBOPROF.org/data#COMP6741>	wikidata	<file:///Users/sharanyu/Desktop/CleanIS/data/course_materials/COMP6741/Lectures/Lecture02/slides/KnowledgeBase02.pdf>				
8 <http://ROBOPROF.org/data#COMP6741>	linked open data	<file:///Users/sharanyu/Desktop/CleanIS/data/course_materials/COMP6741/Lectures/Lecture03/slides/VocabulariesAndOntologies03.pdf>				

Query 6: How many credits is [course] [number] worth?

The query for retrieving a course's credit value straightforwardly selects the credits property associated with the specified course identifier. It exemplifies how direct property queries can yield specific pieces of academic information.

Output: Outputs the credit value of the course, critical for students managing their academic load.

Table			Response	1 result in 0.017 seconds	Simple view	Ellipse	Filter query results	Page size: 50	?
courseCode	subject	credits							
1 6431	SOEN	*4*^<http://www.w3.org/2001/XMLSchema#integer>							

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

We focussed on identifying any external links or resources that are associated with the course identified by a specific course number. These resources are linked as supplementary materials within the course metadata.

Output: This query would output a list of URLs for additional resources or web pages recommended for the specified course.

Table			Response	8 results in 0.033 seconds	Simple view	Ellipse	Filter query results	Page size: 50	?
additionalResources									
1 https://concordiauniversity.on.worldcat.org/oclc/954339855									
2 https://concordiauniversity.on.worldcat.org/oclc/801812987									
3 https://concordiauniversity.on.worldcat.org/oclc/314121652									
4 https://www.w3.org/People/Ivan/CorePresentations/RDFTutorial/									
5 https://www-nlp.stanford.edu/R-book/									
6 https://www.w3.org/TR/rdf11-primer/									
7 https://concordiauniversity.on.worldcat.org/oclc/960211579									
8 https://concordiauniversity.on.worldcat.org/oclc/897466408									

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

Detailing available content for a specific lecture involves querying for lecture-associated materials by filtering on both the course and the lecture number. This showcases the schema's capacity to organize and retrieve granular educational content.

Output: Lists detailed lecture content, enabling students to access specific lecture materials.

Table			Response	2 results in 0.029 seconds	Simple view	Ellipse	Filter query results	Page size: 50	?
ContentsAvailable									
1 file:///Users/sharanyu/Desktop/CleanIS/data/course_materials/COMP6741/Lectures/Lecture02/slides/KnowledgeBase02.pdf									
2 file:///Users/sharanyu/Desktop/CleanIS/data/course_materials/COMP6741/Lectures/Lecture02/worksheets/worksheet02.pdf									

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

To find recommended reading materials for a given topic within a specific course, the query first identifies the course by its identifier and then filters lecture materials to those tagged with the specified topic. This approach leverages the structured relationship between courses, lectures, and their content, narrowing down to readings that are explicitly linked to the topic of interest.

Output: Lists URIs or titles of reading materials recommended for the topic within the specified course, directly supporting targeted academic study.

Simple view Ellipse Filter query results Page size: 50		
course	topicLabel	source
1 <http://ROBOPROF.org/data#SOEN6431>	ieee software	<file:///Users/sharanyu/Desktop/CleanSJ/data/course_materials/SOEN6431/lectures/Lecture03/readings/TechnicalDebt03.pdf>
Showing 1 to 1 of 1 entries		

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

This query determines the competencies, represented as topics, that a student gains upon successfully completing a specified course. It necessitates tracing from the course to the topics covered within its lectures, assuming successful course completion implies competency in these topics.

Output: Provides a list of topic names, indicating the competencies students are expected to gain, reinforcing the course's educational objectives.

Simple view Ellipse Filter query results Page size: 50		
course	topic	topicLabel
1 <http://ROBOPROF.org/data#SOEN6431>	<http://ROBOPROF.org/data#Len_Bass>	len bass
2 <http://ROBOPROF.org/data#SOEN6431>	<http://ROBOPROF.org/data#Association_for_Computing_Machinery>	acm
3 <http://ROBOPROF.org/data#SOEN6431>	<http://ROBOPROF.org/data#Canada>	canada
4 <http://ROBOPROF.org/data#SOEN6431>	<http://ROBOPROF.org/data#Addison-Wesley>	addison-wesley professional
5 <http://ROBOPROF.org/data#SOEN6431>	<http://ROBOPROF.org/data#Vancouver>	vancouver
6 <http://ROBOPROF.org/data#SOEN6431>	<http://ROBOPROF.org/data#International_Standard_Book_Number>	isbn
7 <http://ROBOPROF.org/data#SOEN6431>	<http://ROBOPROF.org/data#Soen>	soen
8 <http://ROBOPROF.org/data#SOEN6431>	<http://ROBOPROF.org/data#IEEE_Software>	ieee software
9 <http://ROBOPROF.org/data#SOEN6431>	<http://ROBOPROF.org/data#Jim_Highsmith>	jim highsmith
10 <http://ROBOPROF.org/data#SOEN6431>	<http://ROBOPROF.org/data#Sonar>	sonar
11 <http://ROBOPROF.org/data#SOEN6431>	<http://ROBOPROF.org/data#Carnegie_Mellon_University>	cmu
12 <http://ROBOPROF.org/data#SOEN6431>	<http://ROBOPROF.org/data#Carnegie_Mellon_University>	carnegie mellon university

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

To retrieve the grades a student achieved in a particular course, the query matches the student to their course enrollments and grades, focusing on the specified course. This query exemplifies how student academic records are queried within the knowledge graph.

Output: Outputs the grade(s) the specified student achieved in the course, providing insight into academic performance.

Simple view Ellipse Filter query results Page size: 50			
firstName	lastName	sem	grade
2 Sharanyu	Pillai	Winter 2022	F
1 Sharanyu	Pillai	Fall 2023	A

Query 12: Which [students] have completed [course] [number]?

Identifying students who have completed a specific course involves filtering for enrollments where the course matches the query and the grade indicates successful completion.

Output: Lists names of students who have successfully completed the specified course, highlighting academic achievements.

student	firstName	lastName	sem
4 <http://ROBOPROF.org/data#Joshua_Lonaker_40227988>	Joshua	Lonaker	Winter 2022
6 <http://ROBOPROF.org/data#Maa'iz_al-Dia_40227996>	Maa'iz	al-Dia	Summer 2023
3 <http://ROBOPROF.org/data#Maja_Nicholson_40227997>	Maja	Nicholson	Winter 2023
2 <http://ROBOPROF.org/data#Natasha_Yarusso_40227990>	Natasha	Yarusso	Winter 2023
5 <http://ROBOPROF.org/data#Preston_Suarez_40227994>	Preston	Suarez	Fall 2023
1 <http://ROBOPROF.org/data#Rochelle_Johnson_40227992>	Rochelle	Johnson	Fall 2023
7 <http://ROBOPROF.org/data#Sushant_Sharma_40227986>	Sushant	Sharma	Winter 2023

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

Generating a student's transcript involves querying for all course enrollments associated with the student, including course identifiers and achieved grades. This comprehensive query illustrates the knowledge graph's ability to aggregate a student's academic history.

Output: Produces a detailed transcript for the specified student, including a list of all courses taken along with the grades received for each course.

firstName	lastname	courseTitle	Course	sem	grade
4 Sushant	Sharma	SOFT. COMP& MAINTENANCE	SOEN6431	Winter 2023	B+
3 Sushant	Sharma	INTELLIGENT SYSTEMS	COMP6741	Winter 2023	C
1 Sushant	Sharma	SOFT. COMP& MAINTENANCE	SOEN6431	Fall 2023	F
2 Sushant	Sharma	INTELLIGENT SYSTEMS	COMP6741	Winter 2022	F

Section 4: SPARQL Endpoint Setup

The deployment of a triplestore and the configuration of a SPARQL endpoint are critical components in making the Roboprof knowledge graph accessible and query-able. This section outlines the technologies chosen for these purposes, the setup process, challenges encountered, and solutions implemented to ensure efficient data storage and retrieval.

4.1 Selection of Triplestore

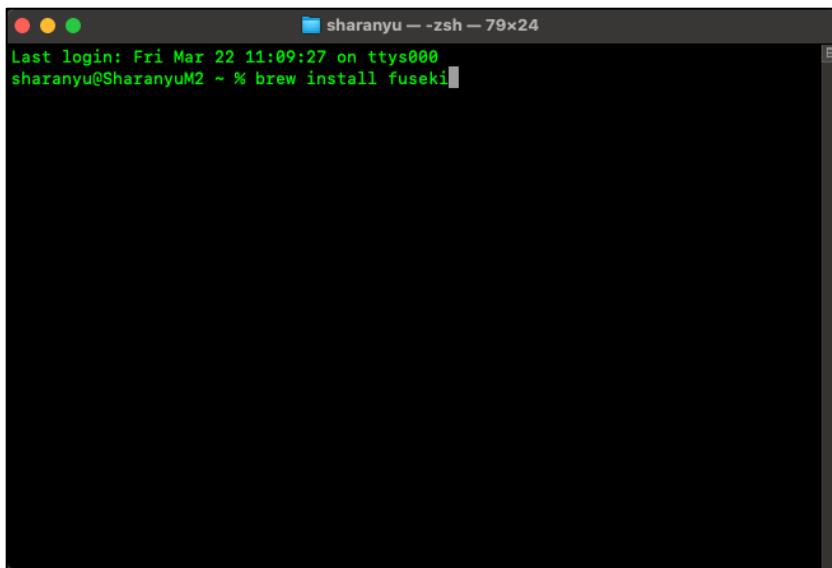
Apache Jena Fuseki: For the Roboprof project, we selected Apache Jena Fuseki as our triplestore. Fuseki is a widely recognized open-source platform that provides robust support for RDF data storage and SPARQL querying capabilities. This choice was driven by Fuseki's performance, scalability, and ease of integration with our existing data pipeline.

4.2 Setup Process

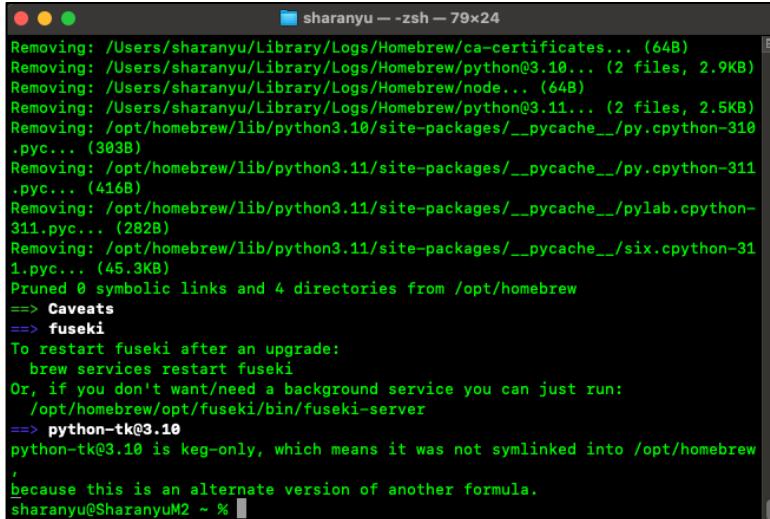
Installation and Configuration:

Installation is straight forward in MacOS.

Open the terminal and execute “brew install fuseki”.

A screenshot of a terminal window titled "sharanyu -- zsh -- 79x24". The window shows the command "brew install fuseki" being typed at the prompt. The background of the terminal is black, and the text is white.

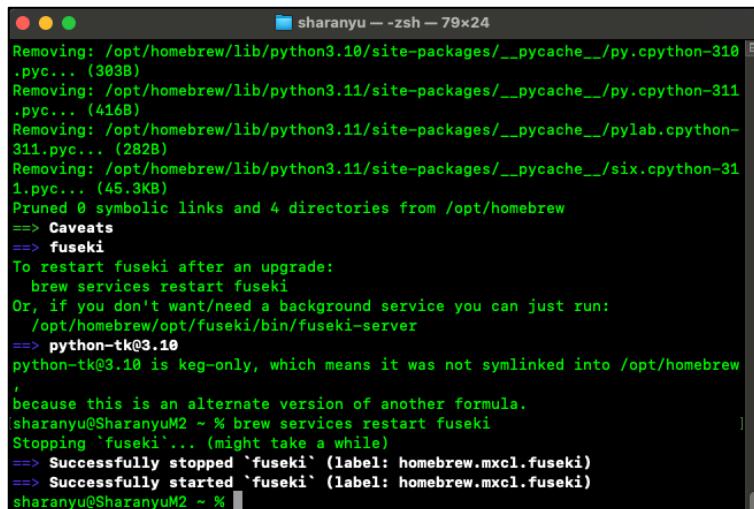
Once the terminal returns



```
sharanyu -- zsh - 79x24
Removing: /Users/sharanyu/Library/Logs/Homebrew/ca-certificates... (64B)
Removing: /Users/sharanyu/Library/Logs/Homebrew/python@3.10... (2 files, 2.9KB)
Removing: /Users/sharanyu/Library/Logs/Homebrew/node... (64B)
Removing: /Users/sharanyu/Library/Logs/Homebrew/python@3.11... (2 files, 2.5KB)
Removing: /opt/homebrew/lib/python3.10/site-packages/__pycache__/py.cpython-310
.pyc... (308B)
Removing: /opt/homebrew/lib/python3.11/site-packages/__pycache__/py.cpython-311
.pyc... (416B)
Removing: /opt/homebrew/lib/python3.11/site-packages/__pycache__/pylab.cpython-
311.pyc... (282B)
Removing: /opt/homebrew/lib/python3.11/site-packages/__pycache__/six.cpython-31
1.pyc... (45.3KB)
Pruned 0 symbolic links and 4 directories from /opt/homebrew
==> Caveats
==> fuseki
To restart fuseki after an upgrade:
  brew services restart fuseki
Or, if you don't want/need a background service you can just run:
  /opt/homebrew/opt/fuseki/bin/fuseki-server
==> python-tk@3.10
python-tk@3.10 is keg-only, which means it was not symlinked into /opt/homebrew
'
because this is an alternate version of another formula.
sharanyu@SharanyuM2 ~ %
```

Now type “brew services restart fuseki” and press enter.

The terminal will now return.



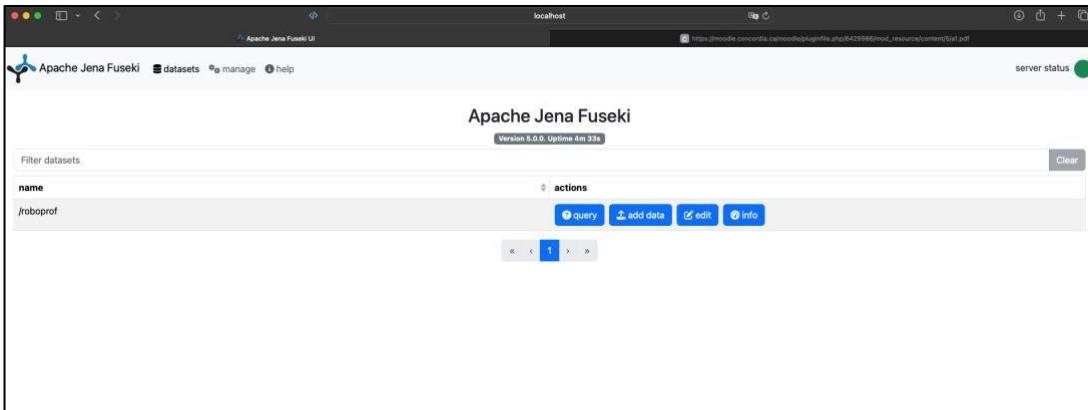
```
sharanyu -- zsh - 79x24
Removing: /opt/homebrew/lib/python3.10/site-packages/__pycache__/py.cpython-310
.pyc... (308B)
Removing: /opt/homebrew/lib/python3.11/site-packages/__pycache__/py.cpython-311
.pyc... (416B)
Removing: /opt/homebrew/lib/python3.11/site-packages/__pycache__/pylab.cpython-
311.pyc... (282B)
Removing: /opt/homebrew/lib/python3.11/site-packages/__pycache__/six.cpython-31
1.pyc... (45.3KB)
Pruned 0 symbolic links and 4 directories from /opt/homebrew
==> Caveats
==> fuseki
To restart fuseki after an upgrade:
  brew services restart fuseki
Or, if you don't want/need a background service you can just run:
  /opt/homebrew/opt/fuseki/bin/fuseki-server
==> python-tk@3.10
python-tk@3.10 is keg-only, which means it was not symlinked into /opt/homebrew
'
because this is an alternate version of another formula.
sharanyu@SharanyuM2 ~ % brew services restart fuseki
Stopping 'fuseki'... (might take a while)
==> Successfully stopped 'fuseki' (label: homebrew.mxcl.fuseki)
==> Successfully started 'fuseki' (label: homebrew.mxcl.fuseki)
sharanyu@SharanyuM2 ~ %
```

Apache Fuseki is now installed.

Difficulties faced: We tried to install Apache fuseki using conventional methods and also docker. This method seems to be more straightforward and time saving.

Starting with Apache fuseki:

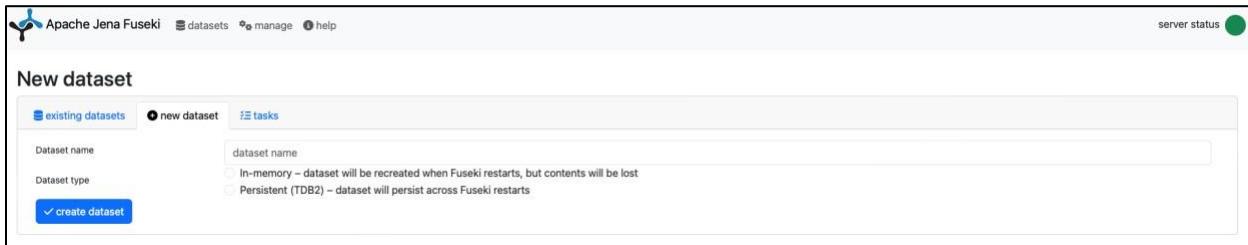
By Default, Apache fuseki starts in localhost:3030



In the right top corner, you can see the server status. Green – Server is up and running.

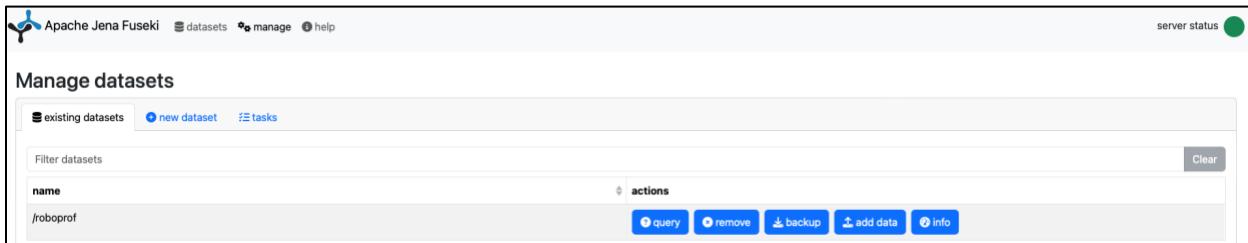
4.3 Data loading

To load data select manage in the left top corner of the homepage > select new dataset.

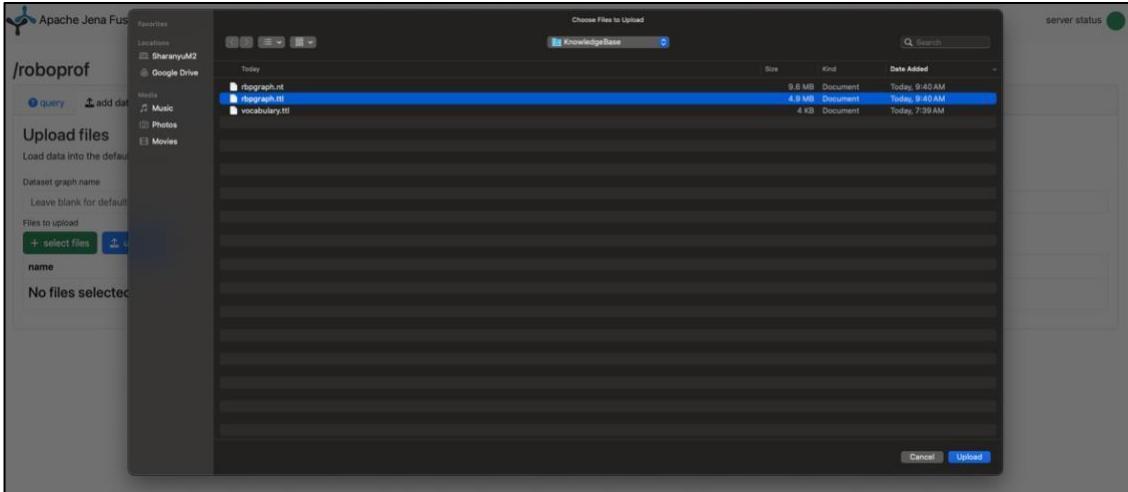


Here you can type the dataset you wish to set and choose the option in-memory as it was suitable for our development work.

Next select “add data”



Choose the turtle file (our knowledge base) and select upload.



If there are no errors in uploading the triples, then the status will be 100% and in green color like below:

name	size	speed	status	actions
rbgraph.ttl	4.63mb	2.63mb/s	<div style="width: 100%;">100.00</div>	upload now remove

Next click on query in the top left corner beside add data.

This is going to be our playground to test the data.

```

PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX schema: <http://schema.org/>
PREFIX rbl: <http://ROBOPROF.org/class#>
PREFIX rb_data: <http://ROBOPROF.org/data#>
PREFIX dc: <http://purl.org/dc/elements/1.1/>
PREFIX dcore: <http://purl.org/dc/terms/>

SELECT DISTINCT ?lecture ?slides
WHERE {
  ?course a schema:Course .
  ?course rbc:coreIsSubject ?subject .
  ?course rbc:containsLecture ?lecture .
  ?lecture rbslides ?slides .
  ?lecture rbc:coversTopic ?topic .
  ?topic rdfs:label ?topicLabel .
  FILTER(REGEX(?topicLabel, "version", "i"))
  FILTER(?subject = "SOEN")
}
    
```

lecture	slides
<http://ROBOPROF.org/data#MiningSoftwareRepositories>	<file:///Users/sharanyu/My%20Drive/Edu/Concordia/CU%20MACS/IS/Project/data/course_materials/SOEN6431/slides/MiningSoftwareRepositories04.pdf>
<http://ROBOPROF.org/data#SoftwareRepositories>	<file:///Users/sharanyu/My%20Drive/Edu/Concordia/CU%20MACS/IS/Project/data/course_materials/SOEN6431/slides/SoftwareRepositories05.pdf>
<http://ROBOPROF.org/data#ContinuousIntegration>	<file:///Users/sharanyu/My%20Drive/Edu/Concordia/CU%20MACS/IS/Project/data/course_materials/SOEN6431/slides/ContinuousIntegration05.pdf>

Now you can execute your SPARQL queries to test the Knowledge base and view the results below.
We can also download the data in different formats.

PART#2

Section 5: Report Updates

As we consolidate and revise our report from Part #1, we have undertaken a serious review process to address previous inaccuracies and implement significant updates according to the suggestions provided by the T.A. The following are the key changes and enhancements we have applied:

5.1 RDF Schema Structural Advancements

In alignment with Alejandra's feedback from our first project demonstration, we have elevated the graph.py script's structural design, transitioning lecture materials from direct properties to subclasses under the LectureContent class. We have added Lab sections and Lab content in this deliverable. This shift allows a more nuanced representation of these materials, enriching them with metadata and facilitating more complex query functionalities. Our RDF triples now reflect this enriched hierarchy, aligning with semantic web design principles and improving the depth of our knowledge graph. Also Updating of existing SPARQL queries was required due new schema structural advancements. Folder structure is also modified to align the course lecture and its contents, labs, and its contents.

5.2 Automating KB update in script

We have introduced a function called update_data in chatBot.py which will be executed after the graph.py has been executed and the KB graph has been created. This method will start and update the fuseki server with the latest knowledge base graph and the timestamp.

5.3 Vocabulary File and Ontology Precision

Addressing the critical feedback, we have consistently implemented the roboprof prefix across our knowledge base in the vocabulary.ttl file. This standardization avoids the direct use of entire URLs where a prefix usage is more appropriate, ensuring that our knowledge graph is well-structured and adheres to semantic web best practices. This careful differentiation augments the clarity and expressiveness of our ontology.

These enhancements, diligently applied to our scripts and ontology, reflect our dedication to evolving our project's infrastructure. Our revised report now encapsulates these updates, presenting a unified and comprehensive view of our project's progress.

Section 6: Knowledge Base Population

Our process for populating the knowledge base with topic triples has been significantly improved in the updated **annotate.py** script, following the instructions laid out in the project's second phase. This script is central to creating triples that represent the rich content of our courses, drawing from different documents like slides, lecture notes, and lab materials. Here's a detailed walk-through of the steps we implemented:

Data we used to create triples

data			
course_materials	Today at 1:19 PM	--	Folder
COMP6741	Mar 22, 2024 at 9:04 AM	--	Folder
assignments	Mar 22, 2024 at 9:04 AM	--	Folder
course_outline	Mar 22, 2024 at 9:04 AM	--	Folder
slides	Mar 22, 2024 at 9:04 AM	--	Folder
SOEN6431	Mar 22, 2024 at 9:04 AM	--	Folder
assignments	Mar 22, 2024 at 9:04 AM	--	Folder
course_outline	Mar 22, 2024 at 9:04 AM	--	Folder
readings	Mar 22, 2024 at 9:04 AM	--	Folder
slides	Mar 22, 2024 at 9:04 AM	--	Folder

data			
course_materials	Today at 1:19 PM	--	Folder
COMP6741	Mar 22, 2024 at 9:04 AM	--	Folder
assignments	Mar 22, 2024 at 9:04 AM	--	Folder
course_outline	Mar 22, 2024 at 9:04 AM	--	Folder
slides	Mar 22, 2024 at 9:04 AM	--	Folder
BaseQueries_and_...penDataset04.pdf	Mar 4, 2024 at 6:43 PM	24.2 MB	PDF Document
Introduction_to_L...nt_Systems01.pdf	Mar 4, 2024 at 6:42 PM	19.3 MB	PDF Document
KnowledgeBase02.pdf	Mar 4, 2024 at 6:43 PM	26 MB	PDF Document
MachineLearning06.pdf	Mar 4, 2024 at 6:44 PM	8.5 MB	PDF Document
RecommenderSystems05.pdf	Mar 4, 2024 at 6:44 PM	2.7 MB	PDF Document
VocabulariesAndOntologies03.pdf	Mar 4, 2024 at 6:43 PM	18.8 MB	PDF Document

Pre-processing with Tika

In alignment with the updated project requirements, we transitioned from using PDFMiner to the Apache Tika library for document processing. Previously, we relied on PDFMiner to convert various document formats into plain text, primarily focusing on PDF files. However, to enhance our document handling capabilities and follow the instructions provided in the project description, we switched to using Apache Tika. We utilized the Apache Tika library to convert various document formats, predominantly PDFs, into plain text. Tika excels in extracting text content from documents, which is crucial for the subsequent natural language processing steps. Our script invokes Tika's parser to process files found in the specified course material directories, transforming them into a format suited for entity detection and linking.

```
def extract_text(file_path):
    parsed = parser.from_file(file_path)["content"].splitlines()
    text = "\n".join(s.strip() for s in parsed if len(s.strip()) > 2)
    return text
```

Filtering with spaCy

To refine the entity linking results, we employed spaCy, a powerful natural language processing library. Our script leverages spaCy to process the text and extract named entities, which we then use to filter the Spotlight results. Only entities recognized by both Spotlight and spaCy as named entities are retained, ensuring the relevance and accuracy of our links.

```
def filter_entities_with_spacy(text, spotlight_entities):
    doc = nlp(text)
    named_entities = set(ent.text.lower() for ent in doc.ents)

    filtered_entities = {
        name: uri for name, uri in spotlight_entities.items() if name in named_entities
    }

    return filtered_entities
```

Entity Linking with DBpedia Spotlight

For entity linking, our choice was the DBpedia Spotlight service, which allows us to identify key entities within the text and link them to their corresponding resources in the DBpedia knowledge base. Our script sends the pre-processed text to the Spotlight API and receives annotated data in XML format, which is then parsed into a dictionary for further processing. This step is critical in connecting the textual content with the Linked Open Data cloud, making the content semantically correct and interconnected.

```
def get_spotlight_annotated_file_as_dictionary(file_content):
    headers = {"Accept": "text/xml"}
    data = {"confidence": "0.5", "support": "0", "text": file_content}
    try:
        response = requests.post(
            "https://api.dbpedia-spotlight.org/en/annotate", headers=headers, data=data
        )

        response.raise_for_status()
        annotated_dict = xmltodict.parse(response.content)
        return annotated_dict
    except requests.exceptions.HTTPError as http_err:
        print(f"HTTP error occurred: {http_err}")
    except requests.exceptions.RequestException as err:
        print(f"Request exception: {err}")
    except Exception as e:
        print(f"An error occurred: {e}")

    return None
```

Storing the annotations

Saved the data into a csv file called topic_info.csv under the data folder.

```
for file, text in dictionary.items():
    annotated_data = get_spotlight_annotated_file_as_dictionary(text)
    spotlight_entities = {}
    for annotation in (
        annotated_data.get("Annotation", {}).get("Resources", {}).get("Resource", {})):
        Name = annotation["@surfaceForm"].lower()
        uri = annotation["@URI"]
        spotlight_entities[Name] = uri
    filtered_entities = filter_entities_with_spacy(text, spotlight_entities)
    spotlight_dictionary[file] = filtered_entities

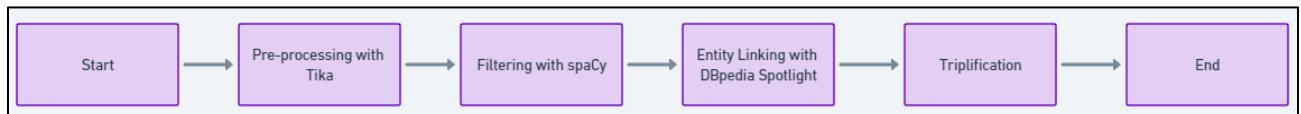
placeholder_df = defaultdict(list)
for k, v in spotlight_dictionary.items():
    k = quote(k)
    placeholder_df["lecture_content"] += [k] * len(v)
    placeholder_df["topic_name"] += list(v.keys())
    placeholder_df["topic_URI"] += list(v.values())

spotlight_df = pd.DataFrame(placeholder_df)

spotlight_df.to_csv(spotlight_path + "/topic_info.csv", index=False)
```

Triplification

With named entities appropriately filtered, our script proceeds to triplify the information. It creates RDF triples that encode the named entity information, linking them to the corresponding URLs of course materials and topics, thereby enriching our knowledge base. This step is carried out using a data structure that maps each document to its identified entities, transforming these associations into a series of RDF triples that are then appended to our graph .



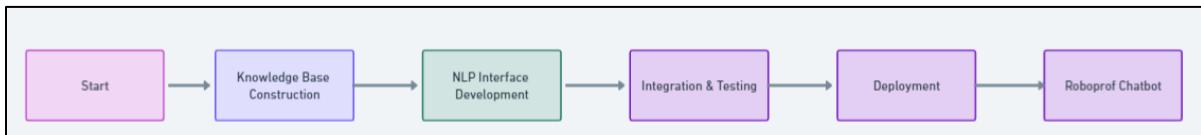
To provide an insight into the scale of the triplification process, we can look into the following statistics of the generated triples:

Sno.	Query	Statistic
1	No. of Triples	65645
2	No. of distinct topics	248
3	No. of topics in a course	101(SOEN6431) 159(COMP6741)

The annotate.py script is integral to this process, providing a robust and efficient mechanism to populate our knowledge base with detailed, connected triples that significantly enhance the capabilities of our roboprof chatbot.

Section 7: Chatbot Design

In this phase of the project, we have developed a natural language interface for our Roboprof chatbot using the Rasa framework, designed to facilitate user interaction with the educational content stored in our knowledge base. This chatbot is engineered to not only continue responding to the diverse queries we set up in Part #1 but also to handle new, specific questions about course details and topic coverage.



7.1 Integrating New Intents and Conversation Flows

We carefully defined new intents for each type of question our chatbot might encounter, such as inquiries about what a specific course entails or which topics are covered in particular course events. Alongside these intents, we crafted stories that guide the flow of conversations. This helps ensure that our chatbot can manage a series of user interactions smoothly, providing accurate and relevant information as needed.

Intents example:

```
intents:
- greet
- goodbye
- affirm
- deny
- mood_great
- mood_unhappy
- bot_challenge
- about_course
- topics_in_course_event
- course_event_coverage_topic
- list_cu_course
- list_cu_course_subject_wise
- course_credits
- additional_resources
- reading_materials_for_topic
- obtaining_topics_after_passing_course
- student_course_performance
- course_performance
- students_transcript
- course_lecture_contents
- course_content_for_topics
```

```
- intent: course_event_coverage_topic
examples: |
- Which course events cover [topic](topic)?
- Can you show which course events discuss [robotics](topic)?
- I need to find courses that include [machine learning](topic) in their events.
- Do any course events focus on [computer vision](topic)?
- Which sessions go into [natural language processing](topic)?
- Are there any seminars on [quantum computing](topic)?
- Find courses that have events discussing [blockchain technology](topic).
- Show me where [cybersecurity](topic) is covered in course events.
- I'm looking for course events that deal with [big data](topic).
- Which course events cover [__oov__](topic)?
- Can you show which course events discuss [robotics](topic)?
```

Stories example:

```
- story: topics obtained after passing the course
  steps:
    - intent: greet
    - action: utter_greet
    - intent: obtaining_topics_after_passing_course
    - action: action_topics_obtained_after_course

- story: student performance in a course
  steps:
    - intent: greet
    - action: utter_greet
    - intent: student_course_performance
    - action: action_student_course_performance

- story: students performance in a course
  steps:
    - intent: greet
    - action: utter_greet
    - intent: course_performance
    - action: action_course_performance
```

Actions:

```
actions:
  - action_course_info
  - action_topics_in_course_event
  - action_course_event_coverage_topic
  - action_list_cu_course
  - action_list_cu_course_from_subject
  - action_course_credits
  - action_additional_resources
  - action_list_reading_materials
  - action_topics_obtained_after_course
  - action_student_course_performance
  - action_course_performance
  - action_students_transcript
  - action_course_lecture_contents
  - action_get_course_content_for_topics
```

Entities:

```
entities:
  - course
  - topic
  - student
  - firstname
  - lastname
  - lecture_number
  - subject
  - event
  - event_type
```

Slots:

```
slots:
  course:
    type: any
    mappings:
      - type: from_entity
        entity: course
  firstname:
    type: text
    mappings:
      - type: from_entity
        entity: firstname
  lastname:
    type: text
    mappings:
      - type: from_entity
        entity: lastname
  student:
    type: any
    mappings:
      - type: from_entity
        entity: student
  subject:
    type: any
    mappings:
      - type: from_entity
        entity: subject
  topic:
    type: text
    mappings:
      - type: from_entity
        entity: topic
  lecture_number:
    type: any
    mappings:
      - type: from_entity
        entity: lecture_number
  event_type:
    type: categorical
    values:
      - lab
      - tutorial
      - lecture
    mappings:
      - type: from_entity
        entity: event_type
```

7.2 Connecting to Fuseki Server for Data Retrieval

To fetch the required data from our knowledge base, we set up actions within Rasa that construct and execute SPARQL queries directed at our Fuseki server. This configuration allows the chatbot to dynamically pull data based on the user's current needs, as dictated by the parsed intents and recognized entities from the user input.

```
FUSEKI_BASE_URL = "http://localhost:3030/roboprof/query"
RB_DATA = Namespace("http://ROBOPROF.org/data#")

def create_spqrql_query(query):
    try:
        print("im here make query")
        rows = requests.get(FUSEKI_BASE_URL, params={"query": query})
        print("make query success ?")
        rows.raise_for_status()
        print("before returning rows")
        return rows
    except requests.exceptions.RequestException as e:
        print(e)
        return None

def get_binding_data(query_result):
    print("results are here")

    results = query_result.get("results", None)
    print(results)
    if results is None:
        return None

    bindings = results.get("bindings", None)
    if bindings is None or not bindings:
        return None

    return bindings
```

7.3 Managing Unrecognized Words

Given the variability of user language, handling words or phrases that the chatbot doesn't recognize is crucial. We also integrated the OOV_token: `__oov__` in our configuration, which stands for "Out of Vocabulary." This special token is used to represent all unrecognized words during training and runtime. The OOV_token is particularly helpful as it allows the model to gracefully handle new or unexpected words that it has not been trained to understand. Instead of failing or misinterpreting these words, the model categorizes them as `__oov__`, which can then be processed in a way that either prompts the user for clarification or ignores the unknown terms without disrupting the flow of conversation.

7.4 ChatBot configuration (config.yml)

In our project, we've significantly improved our chatbot's ability to understand the wide range of ways users express themselves by incorporating a detailed Natural Language Understanding (NLU) setup within the Rasa platform. We use essential tools such as the `WhitespaceTokenizer`, `RegexFeaturizer`, and both word-level and character-level `CountVectorsFeaturizers` to analyze text thoroughly. The `DIETClassifier`, which we've trained for 200 cycles, excels in identifying and sorting user intentions and named entities, with help from the `EntitySynonymMapper` to standardize different terms to their recognized forms. Our dialogue management uses advanced strategies like the "TEDPolicy" and `UnexpectTEDIntentPolicy` to skillfully manage intricate conversations and unexpected questions, ensuring that the chatbot consistently provides relevant responses and smoothly continues interactions. This approach doesn't just reduce confusion; it greatly improves the user experience by effectively handling the variety of ways users communicate.

```
# Configuration for Rasa NLU.
# https://rasa.com/docs/rasa/nlu/components/
language: en

pipeline:
  # No configuration for the NLU pipeline was provided. The following default pipeline was used to train your model.
  # If you'd like to customize it, uncomment and adjust the pipeline.
  # See https://rasa.com/docs/rasa/tuning-your-model for more information.
  - name: WhitespaceTokenizer
  - name: RegexFeaturizer
  - name: LexicalSyntacticFeaturizer
  - name: CountVectorsFeaturizer
  - name: CountVectorsFeaturizer # word-level and character-level (char_wb) features
    analyzer: char_wb
    min_gram: 1
    max_gram: 4
  - name: DIETClassifier
    epochs: 200
    constrain_similarities: true
  - name: EntitySynonymMapper
  - name: ResponseSelector
    epochs: 100
    constrain_similarities: true
  - name: FallbackClassifier
    threshold: 0.3
    ambiguity_threshold: 0.1

  # Configuration for Rasa Core.
  # https://rasa.com/docs/rasa/core/policies/
policies:
  # No configuration for policies was provided. The following default policies were used to train your model.
  # If you'd like to customize them, uncomment and adjust the policies.
  # See https://rasa.com/docs/rasa/policies for more information.
  - name: MemoizationPolicy
  - name: RulePolicy
  - name: UnexpectTEDIntentPolicy
    max_history: 10
    epochs: 100
  - name: TEDPolicy
    max_history: 5
    epochs: 100
    constrain_similarities: true
assistant_id: 20240411-175339-lenient-lion
```

7.5 Strategy for Error Handling

We also prepared strategies to handle situations where the chatbot might not find an answer within the knowledge base. Our approach involves giving users constructive feedback or alternative questions they might ask. This helps maintain a helpful and engaging interaction, guiding users towards useful information even when their initial query can't be directly answered.

```
course = tracker.slots["course"]
if course is None or not course:
    dispatcher.utter_message(text=f"Sorry, I'm not sure I understand")
    return []

res = fus.get_course_description(course)
if res is None or res[0] is None or res[1] is None:
    dispatcher.utter_message(
        text=f"Sorry, I can't seem to find a description for {course}")
)
return []
```

7.6 Generating User-Friendly Responses

To translate the SPARQL query results back to the user, we use **standard templates** to structure the response, we get to keep the output neat and presentable to the user.

```
message = f"Student: {fname} {lname} performance in {course}:\n"
for records in instances:
    fullName, sem, grade = records
    if grade == "F":
        message += f"--> {fullName} ----> {sem} : {grade}( to be repeated)\n"
    else:
        message += f"--> {fullName} ----> {sem} : {grade}\n"

dispatcher.utter_message(text=message)
return []
```

Example:

```
Your input -> How did Edgar Sanchez perform in SOEN6431?
Sorry, I can't seem to find any records of the student Edgar Sanchez for SOEN6431!
Your input -> What was the performance of Sushant Sharma in COMP6741?
Student: Sushant Sharma performance in COMP6741:
--> Sushant Sharma ----> Winter 2022 : F( to be repeated)
--> Sushant Sharma ----> Winter 2023 : C
```

This setup allows us to generate responses that are not just accurate but are also phrased in a way that is easy for users to understand, making the interaction feel more natural and engaging.

Section 8: Chatbot interaction: Prompts and responses

1. Get course info.

Your input -> What is the course COMP6741 about ?

```
Entities recognized are: [{"entity': 'course', 'start': 19, 'end': 27, 'confidence_entity': 0.9997749924659729, 'value': 'COMP6741', 'extractor': 'DIETClassifier', 'processors': ['EntitySynonymMapper']}]
Intent recognized as : about_course
-----
QUERY GENERATED:
PREFIX schema1: <http://schema.org/>PREFIX rb: <http://ROBOPROF.org/class#>PREFIX rb_data: <http://ROBOPROF.org/data#>PREFIX dbpedia_o: <http://dbpedia.org/ontology/>
PREFIX dbcore: <http://purl.org/dc/elements/1.1/>PREFIX dbpedia_p: <http://dbpedia.org/property/>PREFIX schema_t: <http://schema.org/Thing>PREFIX dcterms: <http://purl.org/dc/terms/>PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>PREFIX foaf: <http://xmlns.com/foaf/0.1/>PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
SELECT ?title ?description WHERE { <http://ROBOPROF.org/data#COMP6741> a schema1:Course; dbcore:title ?title; schema_t:description ?description. }

-----
RAW RESULT:
{'bindings': [{}{'title': {'type': 'literal', 'value': 'INTELLIGENT SYSTEMS'}, 'description': {'type': 'literal', 'value': 'Description:\nKnowledge representation and reasoning. Uncertainty and conflict resolution. Design of intelligent systems. Grammar-based, rule-based, and blackboard architectures. A project is required.\nComponents:\nLecture; Laboratory two hours per week\nNotes:\nStudents who have completed COMP 474 may not take this course for credit.'}}]}
```

Here's what I found about COMP6741 - INTELLIGENT SYSTEMS:
Description:
Knowledge representation and reasoning. Uncertainty and conflict resolution. Design of intelligent systems. Grammar-based, rule-based, and blackboard architectures. A project is required.
Component(s):
Lecture; Laboratory two hours per week
Notes:
Students who have completed COMP 474 may not take this course for credit.

2. Listing topics found in Lecture/Lab # in course.

Your input -> Which topics are covered in lecture 05 of COMP6741 ?

```
Entities recognized are: [{"entity': 'event_type', 'start': 28, 'end': 35, 'confidence_entity': 0.8606035113334656, 'value': 'lecture', 'extractor': 'DIETClassifier', 'processors': ['EntitySynonymMapper']}, {"entity': 'lecture_number', 'start': 36, 'end': 38, 'confidence_entity': 0.99958735704422, 'value': '05', 'extractor': 'DIETClassifier', 'processors': ['EntitySynonymMapper']}, {"entity': 'course', 'start': 42, 'end': 50, 'confidence_entity': 0.9998948574066162, 'value': 'COMP6741', 'extractor': 'DIETClassifier', 'processors': ['EntitySynonymMapper']}]
Intent recognized as : topics_in_course_event
-----
QUERY GENERATED:
PREFIX schema1: <http://schema.org/>PREFIX rb: <http://ROBOPROF.org/class#>PREFIX rb_data: <http://ROBOPROF.org/data#>PREFIX dbpedia_o: <http://dbpedia.org/ontology/>
PREFIX dbcore: <http://purl.org/dc/elements/1.1/>PREFIX dbpedia_p: <http://dbpedia.org/property/>PREFIX schema_t: <http://schema.org/Thing>PREFIX dcterms: <http://purl.org/dc/terms/>PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>PREFIX foaf: <http://xmlns.com/foaf/0.1/>PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
SELECT ?lecture ?topicLabel ?source
WHERE {
    <http://ROBOPROF.org/data#COMP6741> a schema1:Course .
    ?course rb:containslecture ?lecture .
    ?lecture dbpedia_p:number "05" .
    ?lecture rb:coversTopic ?topic .
    ?topic rdfs:label ?topicLabel .
    ?topic dcterms:source ?source .
}
```

There are 52 topics covered in Lecture 05 of course COMP6741:
-> Lecture 05 - knn can be found in file:///Users/sharanyu/Desktop/CleanIS/data/course_materials/COMP6741/Lectures/Lecture05/worksheets/worksheet05.pdf
-> Lecture 05 - netflix can be found in file:///Users/sharanyu/Desktop/CleanIS/data/course_materials/COMP6741/Lectures/Lecture05/slides/RecommenderSystems05.pdf
f
-> Lecture 05 - netflix can be found in file:///Users/sharanyu/Desktop/CleanIS/data/course_materials/COMP6741/Lectures/Lecture06/slides/MachineLearning06.pdf
-> Lecture 05 - cambridge university press can be found in file:///Users/sharanyu/Desktop/CleanIS/data/course_materials/COMP6741/Lectures/Lecture05/slides/RecommenderS
ystems05.pdf
-> Lecture 05 - recommender systems can be found in file:///Users/sharanyu/Desktop/CleanIS/data/course_materials/COMP6741/Lectures/Lecture05/slides/RecommenderSystems0
5.pdf
f
-> Lecture 05 - recommender systems can be found in file:///Users/sharanyu/Desktop/CleanIS/data/course_materials/COMP6741/Lectures/Lecture04/worksheets/worksheet04.pdf
-> Lecture 05 - recommender systems can be found in file:///Users/sharanyu/Desktop/CleanIS/data/course_materials/COMP6741/Lectures/Lecture01/slides/Introduction_to_In
telligent_Systems01.pdf
-> Lecture 05 - cosine can be found in file:///Users/sharanyu/Desktop/CleanIS/data/course_materials/COMP6741/Lectures/Lecture05/slides/RecommenderSystems05.pdf
-> Lecture 05 - mary can be found in file:///Users/sharanyu/Desktop/CleanIS/data/course_materials/COMP6741/Lectures/Lecture05/slides/RecommenderSystems05.pdf
-> Lecture 05 - ap can be found in file:///Users/sharanyu/Desktop/CleanIS/data/course_materials/COMP6741/Lectures/Lecture05/slides/RecommenderSystems05.pdf
-> Lecture 05 - john berryman can be found in file:///Users/sharanyu/Desktop/CleanIS/data/course_materials/COMP6741/Lectures/Lecture05/slides/RecommenderSystems05.pdf
-> Lecture 05 - doug turnbull can be found in file:///Users/sharanyu/Desktop/CleanIS/data/course_materials/COMP6741/Lectures/Lecture05/slides/RecommenderSystems05.pdf
.pdf
-> Lecture 05 - prabhakar raghavan can be found in file:///Users/sharanyu/Desktop/CleanIS/data/course_materials/COMP6741/Lectures/Lecture05/slides/RecommenderSystems05
-> Lecture 05 - amazon can be found in file:///Users/sharanyu/Desktop/CleanIS/data/course_materials/COMP6741/Lectures/Lecture05/slides/RecommenderSystems05.pdf
-> Lecture 05 - anthony julius can be found in file:///Users/sharanyu/Desktop/CleanIS/data/course_materials/COMP6741/Lectures/Lecture05/slides/RecommenderSystems05.pdf
-> Lecture 05 - metadata can be found in file:///Users/sharanyu/Desktop/CleanIS/data/course_materials/COMP6741/Lectures/Lecture05/slides/RecommenderSystems05.pdf
-> Lecture 05 - 5.15 can be found in file:///Users/sharanyu/Desktop/CleanIS/data/course_materials/COMP6741/Lectures/Lecture05/slides/RecommenderSystems05.pdf
-> Lecture 05 - hassan can be found in file:///Users/sharanyu/Desktop/CleanIS/data/course_materials/SOEN6431/Lectures/Lecture05/slides/SoftwareRepositories05.pdf
-> Lecture 05 - microsoft visual sourcesafe can be found in file:///Users/sharanyu/Desktop/CleanIS/data/course_materials/SOEN6431/Lectures/Lecture05/slides/SoftwareRe
positories05.pdf
-> Lecture 05 - cvs can be found in file:///Users/sharanyu/Desktop/CleanIS/data/course_materials/SOEN6431/Lectures/Lecture05/slides/SoftwareRepositories05.pdf
-> Lecture 05 - cvs can be found in file:///Users/sharanyu/Desktop/CleanIS/data/course_materials/SOEN6431/Lectures/Lecture04/slides/MiningSoftwareRepositories04.pdf
5.pdf
-> Lecture 05 - cruisecontrol can be found in file:///Users/sharanyu/Desktop/CleanIS/data/course_materials/SOEN6431/Lectures/Lecture05/readings/ContiniousIntegration0
5.pdf
-> Lecture 05 - maven can be found in file:///Users/sharanyu/Desktop/CleanIS/data/course_materials/SOEN6431/Lectures/Lecture05/readings/ContiniousIntegration05.pdf
-> Lecture 05 - sonar can be found in file:///Users/sharanyu/Desktop/CleanIS/data/course_materials/SOEN6431/Lectures/Lecture06/slides/Source_code_analysis06.pdf

3. Labs/Lectures where the topic is covered.

Your input => Which course events cover artificial intelligence ?

Entities recognized are: [{entity': 'topic', 'start': 26, 'end': 49, 'confidence_entity': 0.9998462200164795, 'value': 'artificial intelligence', 'extractor': 'DIETClassifier', 'processors': ['EntitySynonymMapper']}]
Intent recognized as : course_event_coverage_topic

```

PREFIX schema1: <http://schema.org/>PREFIX rb: <http://ROBOPROF.org/class#>PREFIX rb_data: <http://ROBOPROF.org/data#>PREFIX dbpedia_o: <http://dbpedia.org/ontology/>
PREFIX dbcore: <http://purl.org/dc/elements/1.1/>PREFIX dbpedia_p: <http://dbpedia.org/property/>PREFIX schema_t: <http://schema.org/Thing>PREFIX dcterms: <http://purl.org/dc/terms/>PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>PREFIX foaf: <http://xmlns.com/foaf/0.1/>PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
SELECT ?course ?event ?eventType (COUNT(?topic) AS ?topicCount)
WHERE {
?course a schema1:Course .
{
?course rb:containsLab ?event .
BIND("Lab" AS ?eventType)
}
UNION
{
?course rb:containsLecture ?event .
BIND("Lecture" AS ?eventType)
}
?event rb:coversTopic ?topic .
?topic rdfs:label ?topicLabel .
?topic dcterms:source ?source .
FILTER regex(?topicLabel, "artificial intelligence", "i")
}
GROUP BY ?course ?event ?eventType ?topicLabel
ORDER BY DESC(?topicCount)

```

```

RAW RESULT:
{'bindings': [{'course': {'type': 'uri', 'value': 'http://ROBOPROF.org/data#COMP6741'}, 'event': {'type': 'uri', 'value': 'http://ROBOPROF.org/data#COMP6741_Lab01'}, 'eventType': {'type': 'literal', 'value': 'Lab'}, 'topicCount': {'type': 'literal', 'datatype': 'http://www.w3.org/2001/XMLSchema#integer', 'value': '7'}, {'course': {'type': 'uri', 'value': 'http://ROBOPROF.org/data#COMP6741'}, 'event': {'type': 'uri', 'value': 'http://ROBOPROF.org/data#COMP6741_Lab02'}, 'eventType': {'type': 'literal', 'value': 'Lab'}, 'topicCount': {'type': 'literal', 'datatype': 'http://www.w3.org/2001/XMLSchema#integer', 'value': '7'}, {'course': {'type': 'uri', 'value': 'http://ROBOPROF.org/data#COMP6741'}, 'event': {'type': 'uri', 'value': 'http://ROBOPROF.org/data#COMP6741_Lab03'}, 'eventType': {'type': 'literal', 'value': 'Lab'}, 'topicCount': {'type': 'literal', 'datatype': 'http://www.w3.org/2001/XMLSchema#integer', 'value': '7'}, {'course': {'type': 'uri', 'value': 'http://ROBOPROF.org/data#COMP6741'}, 'event': {'type': 'uri', 'value': 'http://ROBOPROF.org/data#COMP6741_Lab04'}, 'eventType': {'type': 'literal', 'value': 'Lab'}, 'topicCount': {'type': 'literal', 'datatype': 'http://www.w3.org/2001/XMLSchema#integer', 'value': '7'}, {'course': {'type': 'uri', 'value': 'http://ROBOPROF.org/data#COMP6741'}, 'event': {'type': 'uri', 'value': 'http://ROBOPROF.org/data#COMP6741_Lab05'}, 'eventType': {'type': 'literal', 'value': 'Lab'}, 'topicCount': {'type': 'literal', 'datatype': 'http://www.w3.org/2001/XMLSchema#integer', 'value': '7'}, {'course': {'type': 'uri', 'value': 'http://ROBOPROF.org/data#COMP6741_Lecture01'}, 'event': {'type': 'uri', 'value': 'http://ROBOPROF.org/data#COMP6741_Lecture02'}, 'eventType': {'type': 'literal', 'value': 'Lecture'}, 'topicCount': {'type': 'literal', 'datatype': 'http://www.w3.org/2001/XMLSchema#integer', 'value': '7'}, {'course': {'type': 'uri', 'value': 'http://ROBOPROF.org/data#COMP6741_Lecture03'}, 'event': {'type': 'uri', 'value': 'http://ROBOPROF.org/data#COMP6741_Lecture04'}, 'eventType': {'type': 'literal', 'value': 'Lecture'}, 'topicCount': {'type': 'literal', 'datatype': 'http://www.w3.org/2001/XMLSchema#integer', 'value': '7'}, {'course': {'type': 'uri', 'value': 'http://ROBOPROF.org/data#COMP6741_Lecture05'}, 'event': {'type': 'uri', 'value': 'http://ROBOPROF.org/data#COMP6741_Lecture06'}, 'eventType': {'type': 'literal', 'value': 'Lecture'}, 'topicCount': {'type': 'literal', 'datatype': 'http://www.w3.org/2001/XMLSchema#integer', 'value': '7'}}]}

```

Your input -> Which course events cover artificial intelligence ?
The Topic -> artificial intelligence are found in:
-> COMP6741's Lab instance ----> COMP6741_Lab01 and found <artificial intelligence> 7 times in course
-> COMP6741's Lab instance ----> COMP6741_Lab02 and found <artificial intelligence> 7 times in course
-> COMP6741's Lab instance ----> COMP6741_Lab03 and found <artificial intelligence> 7 times in course
-> COMP6741's Lab instance ----> COMP6741_Lab04 and found <artificial intelligence> 7 times in course
-> COMP6741's Lab instance ----> COMP6741_Lab05 and found <artificial intelligence> 7 times in course
-> COMP6741's Lecture instance ----> COMP6741_Lecture01 and found <artificial intelligence> 7 times in course
-> COMP6741's Lecture instance ----> COMP6741_Lecture02 and found <artificial intelligence> 7 times in course
-> COMP6741's Lecture instance ----> COMP6741_Lecture03 and found <artificial intelligence> 7 times in course
-> COMP6741's Lecture instance ----> COMP6741_Lecture06 and found <artificial intelligence> 7 times in course
Your input -> []

4. List all courses offered by [university]

Your input -> give me a list of Courses provided

Entities recognized are: []
Intent recognized as : list_cu_course

```

QUERY GENERATED:
PREFIX schema: <http://schema.org/>PREFIX rb: <http://ROBOPROF.org/class#>PREFIX rb_data: <http://ROBOPROF.org/data#>PREFIX dbpedia_o: <http://dbpedia.org/ontology/>
PREFIX dboare: <http://purl.org/dc/elements/1.1/>PREFIX dbpedia_p: <http://dbpedia.org/property/>PREFIX schema_t: <http://schema.org/Thing>PREFIX dcterms: <http://purl.org/dc/terms/>PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>PREFIX foaf: <http://xmlns.com/foaf/0.1/>PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
SELECT ?course { (CONCAT(?subject, " ", ?courseCode, " ", ?title) AS ?courseTitle) WHERE {
?university a dbpedia_o:University ;
rb:courses ?course .
?course a schema:Course ;
dboare:subject ?subject ;
dbpedia_p:number ?courseCode ;
dbcore:title ?title .
FILTER (?university = rb_data:Concordia_University)
} LIMIT 200

```

RAW RESULT:

```

{'bindings': [{'course': {'type': 'uri', 'value': 'http://ROBOPROF.org/data#AHSC411'}, 'courseTitle': {'type': 'literal', 'value': 'AHSC 411: Social change and analysis Methods'}, {'course': {'type': 'uri', 'value': 'http://ROBOPROF.org/data#RELI806'}, 'courseTitle': {'type': 'literal', 'value': 'RELI 806: TOPICS IN JUDAIC STUDIES'}}, {"course": {"type": "uri", "value": "http://ROBOPROF.org/data#FBR3341"}, "courseTitle": {"type": "literal", "value": "FBR3 341: Intermediate Fibre Structures"}}, {"course": {"type": "uri", "value": "http://ROBOPROF.org/data#GEOG498"}, "courseTitle": {"type": "literal", "value": "GEOG 498: Advanced Topics in Geography"}}, {"course": {"type": "uri", "value": "http://ROBOPROF.org/data#GPLL263"}, "courseTitle": {"type": "literal", "value": "GPLL 263: Peer Review Roundtable"}}, {"course": {"type": "uri", "value": "http://ROBOPROF.org/data#COMS895B"}, "courseTitle": {"type": "literal", "value": "COMS 895B: DOCTORAL THESIS RESEARCH"}}, {"course": {"type": "uri", "value": "http://ROBOPROF.org/data#FMST886"}, "courseTitle": {"type": "literal", "value": "FMST 886: SYNTHESIS EXAM"}}, {"course": {"type": "uri", "value": "http://ROBOPROF.org/data#GPDIS15"}, "courseTitle": {"type": "literal", "value": "GPDIS 15: Data Science & Communication with R"}}, {"course": {"type": "uri", "value": "http://ROBOPROF.org/data#POLL487Z"}, "courseTitle": {"type": "literal", "value": "POLL 487Z: Advanced Topic in Comparative Politics : Intergovernmental Relations in Selected Federal States"}}, {"course": {"type": "uri", "value": "http://ROBOPROF.org/data#LEIS398"}, "courseTitle": {"type": "literal", "value": "LEIS 398: SELECT TOP - LEISURE STUDIES"}}, {"course": {"type": "uri", "value": "http://ROBOPROF.org/data#ECON422"}, "courseTitle": {"type": "literal", "value": "ECON 422: Econometrics II"}}, {"course": {"type": "uri", "value": "http://ROBOPROF.org/data#HSC470"}, "courseTitle": {"type": "literal", "value": "HSC 470: BASIC HUMAN RELATIONS LABORATORY"}}, {"course": {"type": "uri", "value": "http://ROBOPROF.org/data#SPAN683"}, "courseTitle": {"type": "literal", "value": "SPAN 683: RESEARCH PAPER II"}}, {"course": {"type": "uri", "value": "http://ROBOPROF.org/data#COMS545"}, "courseTitle": {"type": "literal", "value": "COMS 545: TELEVISION STUDIES"}}, {"course": {"type": "uri", "value": "http://ROBOPROF.org/data#MECH6161"}, "courseTitle": {"type": "literal", "value": "MECH 6161: GAS TURBINE DESIGN"}}, {"course": {"type": "uri", "value": "http://ROBOPROF.org/data#HIST438"}, "courseTitle": {"type": "literal", "value": "HIST 438: ADVANCED STUDY-BRITISH HIST."}}, {"course": {"type": "uri", "value": "http://ROBOPROF.org/data#COMS545"}, "courseTitle": {"type": "literal", "value": "COMS 545: TELEVISION STUDIES"}}, {"course": {"type": "uri", "value": "http://ROBOPROF.org/data#TPER209"}, "courseTitle": {"type": "literal", "value": "TPER 209: Acting I"}}, {"course": {"type": "uri", "value": "http://ROBOPROF.org/data#ENGL387"}, "courseTitle": {"type": "literal", "value": "ENGL 387: South Asian Literature"}}, {"course": {"type": "uri", "value": "http://ROBOPROF.org/data#RELI619"}, "courseTitle": {"type": "literal", "value": "RELI 619: RDGS IN WORLD RELIGIONS"}}, {"course": {"type": "uri", "value": "http://ROBOPROF.org/data#ACCO425"}, "courseTitle": {"type": "literal", "value": "ACCO 425: Accounting for Managers"}]}

```

```
Your input -> give me a list of Courses provided
Total (200) courses provided in Concordia University:
Showing first 200 due to space constraints
-> http://ROBOPROF.org/data#AHSC411 ----> AHSC 411: Social change and analysis Methods
-> http://ROBOPROF.org/data#RELI806 ----> RELI 806: TOPICS IN JUDAIC STUDIES
-> http://ROBOPROF.org/data#FBR5341 ----> FBR5 341: Intermediate Fibre Structures
-> http://ROBOPROF.org/data#GEOG498 ----> GEOG 498: Advanced Topics in Geography
-> http://ROBOPROF.org/data#GPLL263 ----> GPLL 263: Peer Review Roundtable
-> http://ROBOPROF.org/data#COMS895B ----> COMS 895B: DOCTORAL THESIS RESEARCH
-> http://ROBOPROF.org/data#FMST886 ----> FMST 886: SYNTHESIS EXAM
-> http://ROBOPROF.org/data#GPDI515 ----> GPDI 515: Data Science & Communication with R
-> http://ROBOPROF.org/data#RELI378 ----> RELI 378: Death and Dying
-> http://ROBOPROF.org/data#POLI487Z ----> POLI 487Z: Advanced Topic in Comparative Politics : Intergovernmental Relations in Selected Federal States
-> http://ROBOPROF.org/data#LEIS398 ----> LEIS 398: SELECT TOP - LEISURE STUDIES
-> http://ROBOPROF.org/data#ECON422 ----> ECON 422: Econometrics II
-> http://ROBOPROF.org/data#AHSC470 ----> AHSC 470: BASIC HUMAN RELATIONS LABORATORY
-> http://ROBOPROF.org/data#SPAN683 ----> SPAN 683: RESEARCH PAPER II
-> http://ROBOPROF.org/data#HIST438 ----> HIST 438: ADVANCED STUDY-BRITISH HIST.
```

5. In which courses is [topic] discussed?

Your input -> - Show course talking about data.

```
Entities recognized are: [{'entity': 'topic', 'start': 28, 'end': 32, 'confidence_entity': 0.451277494430542, 'value': 'data', 'extractor': 'DIETClassifier', 'processors': ['EntitySynonymMapper']}]
Intent recognized as : course_content_for_topics
```

QUERY GENERATED:

```

PREFIX schema1: <http://schema.org/>
PREFIX rb: <http://ROBOPROF.org/class#>
PREFIX rb_data: <http://ROBOPROF.org/data#>
PREFIX dbpedia_o: <http://dbpedia.org/ontology/>
PREFIX dbcore: <http://purl.org/dc/elements/1.1/>
PREFIX dbpedia_p: <http://dbpedia.org/property/>
PREFIX schema_t: <http://schema.org/Thing>
PREFIX dcterms: <http://purl.org/dc/terms/>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>

SELECT DISTINCT ?course ?topicLabel ?source
WHERE {
  ?course a schema1:Course .
  {
    { ?course rb:containsLab ?event . BIND("Lab" AS ?eventType) }
    UNION
    { ?course rb:containsLecture ?event . BIND("Lecture" AS ?eventType) }
  }
  ?event rb:coversTopic ?topic .
  ?topic rdfs:label ?topicLabel .
  ?topic dcterms:source ?source .
  FILTER regex(?topicLabel, "data", "i")
  FILTER (?course = <http://ROBOPROF.org/data#COMP6741>)
}

```

RAW RESULT:

Your input => - Show course talking about data.

```
I found 15 Contents found for data in course- COMP6741:  
--> COMP6741 --> wikidata ----- file:///Users/sharanyu/Desktop/CleanIS/data/course_materials/COMP6741/Labs/Lab05/COMP-474-2234-UU_Lab%2520Session_05_Moodle.pdf  
--> COMP6741 --> wikidata ----- file:///Users/sharanyu/Desktop/CleanIS/data/course_materials/COMP6741/Lectures/Lecture03/worksheets/worksheet03.pdf  
--> COMP6741 --> wikidata ----- file:///Users/sharanyu/Desktop/CleanIS/data/course_materials/COMP6741/Labs/Lab02/COMP-474-2234-UU_Lab%2520Session_02_Moodle.pdf  
--> COMP6741 --> wikidata ----- file:///Users/sharanyu/Desktop/CleanIS/data/course_materials/COMP6741/Labs/Lab04/COMP-474-2234-UU_Lab%2520Session_04_Moodle.pdf  
--> COMP6741 --> wikidata ----- file:///Users/sharanyu/Desktop/CleanIS/data/course_materials/COMP6741/Lectures/Lecture01/worksheets/worksheet01.pdf  
--> COMP6741 --> wikidata ----- file:///Users/sharanyu/Desktop/CleanIS/data/course_materials/COMP6741/Labs/Lab03/COMP-474-2234-UU_Lab%2520Session_03_Moodle.pdf  
--> COMP6741 --> linked open data ----- file:///Users/sharanyu/Desktop/CleanIS/data/course_materials/COMP6741/Lectures/Lecture02/slides/KnowledgeBase02.pdf  
--> COMP6741 --> linked open data ----- file:///Users/sharanyu/Desktop/CleanIS/data/course_materials/COMP6741/Lectures/Lecture03/slides/VocabulariesAndOntologies03.pdf  
df  
--> COMP6741 --> linked open data ----- file:///Users/sharanyu/Desktop/CleanIS/data/course_materials/COMP6741/Lectures/Lecture04/slides/BaseQueries_and_LinkedOpenDataset04.pdf  
--> COMP6741 --> linked open data ----- file:///Users/sharanyu/Desktop/CleanIS/data/course_materials/COMP6741/Labs/Lab05/COMP-474-2234-UU_Lab%2520Session_05_Moodle.pdf  
--> COMP6741 --> linked data ----- file:///Users/sharanyu/Desktop/CleanIS/data/course_materials/COMP6741/Lectures/Lecture03/slides/VocabulariesAndOntologies03.pdf  
--> COMP6741 --> linked data ----- file:///Users/sharanyu/Desktop/CleanIS/data/course_materials/COMP6741/Lectures/Lecture04/slides/BaseQueries_and_LinkedOpenDataset04.pdf  
--> COMP6741 --> linked data ----- file:///Users/sharanyu/Desktop/CleanIS/data/course_materials/COMP6741/Labs/Lab05/COMP-474-2234-UU_Lab%2520Session_05_Moodle.pdf  
--> COMP6741 --> dataframes ----- file:///Users/sharanyu/Desktop/CleanIS/data/course_materials/COMP6741/Labs/Lab05/COMP-474-2234-UU_Lab%2520Session_05_Moodle.pdf  
--> COMP6741 --> metadata ----- file:///Users/sharanyu/Desktop/CleanIS/data/course_materials/COMP6741/Lectures/Lecture05/slides/RecommenderSystems05.pdf  
Your input -> □
```

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

Your input -> List all topics from Lecture 01 in SOEN6431.

```
Entities recognized as: {'entity': 'event_type', 'start': 23, 'end': 30, 'confidence_entity': 0.99972421169281, 'value': 'lecture', 'extractor': 'DIEClassifier', 'processors': ['EntitySynonymMapper']}, {'entity': 'lecture_number', 'start': 31, 'end': 33, 'confidence_entity': 0.9996438026428223, 'value': '01', 'extractor': 'DIEClassifier'}, {'entity': 'course', 'start': 37, 'end': 45, 'confidence_entity': 0.9999198913574219, 'value': 'SOEN6431', 'extractor': 'DIEClassifier', 'processors': ['EntitySynonymMapper']}}

Intent recognized as: topics_in_course_event
```

```
QUERY GENERATED:

PREFIX schema1: <http://schema.org/>
PREFIX rb: <http://ROBOPROF.org/class#>
PREFIX rb_data: <http://ROBOPROF.org/data#>
PREFIX dbpedia_o: <http://dbpedia.org/ontology/>
PREFIX dbocore: <http://purl.org/dc/elements/1.1/>
PREFIX dbpedia_p: <http://dbpedia.org/property/>
PREFIX schema_t: <http://schema.org/Thing>
PREFIX dcterms: <http://purl.org/dc/terms/>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>

SELECT ?lecture ?topicLabel ?source
WHERE {
  <http://ROBOPROF.org/data#SOEN6431> a schema1:Course .
  ?course rb:containsLecture ?lecture .
  ?lecture rb:containsTopic ?topic .
  ?topic rdfs:label ?topicLabel .
  ?topic dcterms:source ?source .
}
}
```

```
Your input -> - List all topics from Lecture 01 in SOEN6431.
There are 151 topics covered in Lecture 01 of course SOEN6431:
-> Lecture 01 - pirzada can be found in file:///Users/sharanyu/Desktop/CleanIS/data/course_materials/SOEN6431/Lectures/Lecture01/slides/IntroSoftwareEvolution01.pdf
-> Lecture 01 - acm can be found in file:///Users/sharanyu/Desktop/CleanIS/data/course_materials/SOEN6431/Lectures/Lecture03/readings/TechnicalDebt03.pdf
-> Lecture 01 - acm can be found in file:///Users/sharanyu/Desktop/CleanIS/data/course_materials/SOEN6431/Lectures/Lecture01/slides/IntroSoftwareEvolution01.pdf
-> Lecture 01 - spiral model can be found in file:///Users/sharanyu/Desktop/CleanIS/data/course_materials/SOEN6431/Lectures/Lecture01/slides/IntroSoftwareEvolution01.pdf
-> Lecture 01 - soon can be found in file:///Users/sharanyu/Desktop/CleanIS/data/course_materials/SOEN6431/Lectures/Lecture06/slides/Source_code_analysis06.pdf
-> Lecture 01 - soon can be found in file:///Users/sharanyu/Desktop/CleanIS/data/course_materials/SOEN6431/Lectures/Lecture06/readings/Impact_Analysis_and_RegressionTesting06.pdf
-> Lecture 01 - soon can be found in file:///Users/sharanyu/Desktop/CleanIS/data/course_materials/SOEN6431/Lectures/Lecture03/slides/TechnicalDebt03.pdf
-> Lecture 01 - soon can be found in file:///Users/sharanyu/Desktop/CleanIS/data/course_materials/SOEN6431/Lectures/Lecture05/slides/SoftwareRepositories05.pdf
-> Lecture 01 - soon can be found in file:///Users/sharanyu/Desktop/CleanIS/data/course_materials/SOEN6431/Lectures/Lecture07/slides/Clones07.pdf
-> Lecture 01 - soon can be found in file:///Users/sharanyu/Desktop/CleanIS/data/course_materials/SOEN6431/Lectures/Lecture01/slides/IntroSoftwareEvolution01.pdf
-> Lecture 01 - uml can be found in file:///Users/sharanyu/Desktop/CleanIS/data/course_materials/SOEN6431/Lectures/Lecture02/slides/SoftwareTraceability02.pdf
-> Lecture 01 - uml can be found in file:///Users/sharanyu/Desktop/CleanIS/data/course_materials/SOEN6431/Lectures/Lecture01/slides/IntroSoftwareEvolution01.pdf
-> Lecture 01 - software evolution can be found in file:///Users/sharanyu/Desktop/CleanIS/data/course_materials/SOEN6431/Lectures/Lecture01/slides/IntroSoftwareEvolution01.pdf
```

7. List all [courses] offered by [university] within the [subject] (e.g., "COMP", "SOEN").

Your input -> list all courses for [COMP]

Entities recognized are: [{'entity': 'subject', 'start': 22, 'end': 26, 'confidence_entity': 0.9993937015533447, 'value': 'COMP', 'extractor': 'DIETClassifier', 'processors': ['EntitySynonymMapper']}]
Intent recognized as : list_cu_course_subject_wise

```
QUERY GENERATED:  
  
PREFIX schema1: <http://schema.org/>PREFIX rb: <http://ROBOPROF.org/class#>PREFIX rb_data: <http://ROBOPROF.org/data#>PREFIX dbpedia_o: <http://dbpedia.org/ontology/>  
PREFIX dbcore: <http://purl.org/dc/elements/1.1/>PREFIX dbpedia_p: <http://dbpedia.org/resource/>PREFIX schema_t: <http://schema.org/Thing>PREFIX dcterms: <http://purl.org/dc/terms/>PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>PREFIX foaf: <http://xmlns.com/foaf/0.1/>PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>  
SELECT ?course (CONCAT(?subject, " ", ?courseCode, " ", ?title) AS ?courseDetails) WHERE {  
    ?university a dbpedia_o:University ;  
        rb:courses ?course .  
    ?course a schema1:Course ;  
        dbcore:subject ?subject ;  
        dbpedia_p:isNumber ?courseCode ;  
        dbcore:title ?title .  
    FILTER (?subject = "COMP")  
    FILTER (?university = rb_data:Concordia_University)  
}
```

RAW RESULT:

```
{'bindings': [{'course': {'type': 'uri', 'value': 'http://ROBOPROF.org/data#COMP5541'}, 'courseDetails': {'type': 'literal', 'value': 'COMP 5541: TOOLS+TECHNIQUES / SW ENGR.'}, {'course': {'type': 'uri', 'value': 'http://ROBOPROF.org/data#COMP367'}, 'courseDetails': {'type': 'literal', 'value': 'COMP 367: Techniques in Symbolic Computation'}}, {'course': {'type': 'uri', 'value': 'http://ROBOPROF.org/data#COMP426'}, 'courseDetails': {'type': 'literal', 'value': 'COMP 426: Multicore Programming'}}, {'course': {'type': 'uri', 'value': 'http://ROBOPROF.org/data#COMP6721'}, 'courseDetails': {'type': 'literal', 'value': 'COMP 6721: APPLIED ARTIFICIAL INTELLIGENCE'}}, {'course': {'type': 'uri', 'value': 'http://ROBOPROF.org/data#COMP6331'}, 'courseDetails': {'type': 'literal', 'value': 'COMP 6331: ADVANCED GAME DEVELOPMENT'}}, {'course': {'type': 'uri', 'value': 'http://ROBOPROF.org/data#COMP433'}, 'courseDetails': {'type': 'literal', 'value': 'COMP 433: INTRODUCTION TO DEEP LEARNING'}}, {'course': {'type': 'uri', 'value': 'http://ROBOPROF.org/data#COMP233'}, 'courseDetails': {'type': 'literal', 'value': 'COMP 233: Probability and Statistics for Computer Science'}}, {'course': {'type': 'uri', 'value': 'http://ROBOPROF.org/data#COMP6971'}, 'courseDetails': {'type': 'literal', 'value': 'COMP 6971: Project and Report'}}, {'course': {'type': 'uri', 'value': 'http://ROBOPROF.org/data#COMP6281'}, 'courseDetails': {'type': 'literal', 'value': 'COMP 6281: PARALLEL PROGRAMMING'}}, {'course': {'type': 'uri', 'value': 'http://ROBOPROF.org/data#COMP691'}, 'courseDetails': {'type': 'literal', 'value': 'COMP 691: TOPICS/COMPUTER SCIENCE'}}, {'course': {'type': 'uri', 'value': 'http://ROBOPROF.org/data#COMP6751'}, 'courseDetails': {'type': 'literal', 'value': 'COMP 6751: NATURAL LANGUAGE ANALYSIS'}}, {'course': {'type': 'uri', 'value': 'http://ROBOPROF.org/data#COMP5511'}, 'courseDetails': {'type': 'literal', 'value': 'COMP 5511: PRINCIPLES/DATA STRUCTURES'}}, {'course': {'type': 'uri', 'value': 'http://ROBOPROF.org/data#COMP445'}, 'courseDetails': {'type': 'literal', 'value': 'COMP 445: Data Communication and Computer Networks'}}, {'course': {'type': 'uri', 'value': 'http://ROBOPROF.org/data#COMP345'}, 'courseDetails': {'type': 'literal', 'value': 'COMP 345: ADVANCED PROGRAM DESIGN, C++'}}, {'course': {'type': 'uri', 'value': 'http://ROBOPROF.org/data#COMP790'}, 'courseDetails': {'type': 'literal', 'value': 'COMP 790: TOPICS/COMPUTER SCIENCE II'}}]
```

Your input -> **list all courses for COMP**

(114) courses provided in Concordia University for subject COMP:

- > http://ROBOPROF.org/data#COMP5541 -----> COMP 5541: TOOLS+TECHNIQUES / SW ENGR.
- > http://ROBOPROF.org/data#COMP367 -----> COMP 367: Techniques in Symbolic Computation
- > http://ROBOPROF.org/data#COMP426 -----> COMP 426: Multicore Programming
- > http://ROBOPROF.org/data#COMP6721 -----> COMP 6721: APPLIED ARTIFICIAL INTELLIGENCE
- > http://ROBOPROF.org/data#COMP6331 -----> COMP 6331: ADVANCED GAME DEVELOPMENT
- > http://ROBOPROF.org/data#COMP433 -----> COMP 433: INTRODUCTION TO DEEP LEARNING
- > http://ROBOPROF.org/data#COMP233 -----> COMP 233: Probability and Statistics for Computer Science
- > http://ROBOPROF.org/data#COMP6971 -----> COMP 6971: Project and Report
- > http://ROBOPROF.org/data#COMP6281 -----> COMP 6281: PARALLEL PROGRAMMING
- > http://ROBOPROF.org/data#COMP691 -----> COMP 691: TOPICS/COMPUTER SCIENCE
- > http://ROBOPROF.org/data#COMP6751 -----> COMP 6751: NATURAL LANGUAGE ANALYSIS
- > http://ROBOPROF.org/data#COMP5511 -----> COMP 5511: PRINCIPLES/DATA STRUCTURES
- > http://ROBOPROF.org/data#COMP791 -----> COMP 791: TOPICS IN COMP. SCIENCE II
- > http://ROBOPROF.org/data#COMP445 -----> COMP 445: Data Communication and Computer Networks
- > http://ROBOPROF.org/data#COMP345 -----> COMP 345: ADVANCED PROGRAM DESIGN, C++
- > http://ROBOPROF.org/data#COMP790 -----> COMP 790: TOPICS/COMPUTER SCIENCE II

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

Your input -> **Provide the reading materials for lecture 04 of SOEN6431.**

Entities recognized are: [{"entity': 'lecture_number', 'start': 42, 'end': 44, 'confidence_entity': 0.9995330572128296, 'value': '04', 'extractor': 'DIETClassifier', 'processors': ['EntitySynonymMapper']}, {"entity': 'course', 'start': 48, 'end': 56, 'confidence_entity': 0.9999314546585083, 'value': 'SOEN6431', 'extractor': 'DIETClassifier', 'processors': ['EntitySynonymMapper']}]
Intent recognized as : course_lecture_contents

QUERY GENERATED:

```
PREFIX schema: <http://schema.org/>PREFIX rb: <http://ROBOPROF.org/class#>PREFIX rb_data: <http://ROBOPROF.org/data#>PREFIX dbcore: <http://purl.org/dc/terms/>PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>PREFIX foaf: <http://xmlns.com/foaf/0.1/>PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
SELECT ?ContentsAvailable
WHERE {
  BIND(rb_data:SOEN6431_Lecture04_LectureContent AS ?lectureContent)
  {
    ?lectureContent rdf:type rb:LectureContent .
    ?lectureContent rb:slides ?contents .
    BIND(?contents AS ?ContentsAvailable)
  }
  UNION
  {
    ?lectureContent rdf:type rb:LectureContent .
    ?lectureContent rb:readings ?contents .
    BIND(?contents AS ?ContentsAvailable)
  }
  UNION
  {
    ?lectureContent rdf:type rb:LectureContent .
    ?lectureContent rb:worksheets ?contents .
    BIND(?contents AS ?ContentsAvailable)
  }
}
```

RAW RESULT:

```
{'bindings': [{'ContentsAvailable': {'type': 'literal', 'value': 'file:///Users/sharanyu/Desktop/CleanIS/data/course_materials/SOEN6431/Lectures/Lecture04/slides/MiningSoftwareRepositories04.pdf'}}, {'ContentsAvailable': {'type': 'literal', 'value': 'file:///Users/sharanyu/Desktop/CleanIS/data/course_materials/SOEN6431/Lectures/Lecture04/readings/MiningSoftwareRepositories04.pdf'}}]}
```

Your input -> **Provide the reading materials for lecture 04 of SOEN6431.**
The Content found for SOEN6431 in Lecture04:
-> file:///Users/sharanyu/Desktop/CleanIS/data/course_materials/SOEN6431/Lectures/Lecture04/slides/MiningSoftwareRepositories04.pdf
-> file:///Users/sharanyu/Desktop/CleanIS/data/course_materials/SOEN6431/Lectures/Lecture04/readings/MiningSoftwareRepositories04.pdf

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

Your input -> **how much credits is ACC0425 worth?**

```
Entities recognized are: [{"entity": "course", "start": 20, "end": 27, "confidence_entity": 0.7969359755516052, "value": "ACCO425", "extractor": "DIETClassifier"}]
Intent recognized as : course_credits
```

QUERY GENERATED:

```
PREFIX schema1: <http://schema.org/>PREFIX rb: <http://ROBOPROF.org/class#>PREFIX rb_data: <http://ROBOPROF.org/data#>PREFIX dbpedia_o: <http://dbpedia.org/ontology/>
PREFIX dccore: <http://purl.org/dc/elements/1.1/>PREFIX dbpedia_p: <http://dbpedia.org/property/>PREFIX schema_t: <http://schema.org/Thing>PREFIX dcterms: <http://purl.org/dc/terms/>PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>PREFIX foaf: <http://xmlns.com/foaf/0.1/>PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
SELECT ?subject ?courseCode ?credits WHERE { <http://ROBOPROF.org/data#ACCO425> a schema1:Course; dccore:subject ?subject; dbpedia_p:number ?courseCode ; rb:credits ?credits .}
```

RAW RESULT:

```
{"bindings": [{"subject": {"type": "literal", "value": "ACCO"}, "courseCode": {"type": "literal", "value": "425"}, "credits": {"type": "literal", "datatype": "http://www.w3.org/2001/XMLSchema#integer", "value": "3"}}]}
```

```
Your input -> how much credits is ACCO425 worth?
Total credits for ACCO-425 is 3
Your input -> [ ]
```

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

```
Your input -> I'm looking for additional resources for COMP6741, can you help?
```

```
Entities recognized are: [{"entity": "course", "start": 41, "end": 49, "confidence_entity": 0.9999138116836548, "value": "COMP6741", "extractor": "DIETClassifier", "processors": ["EntitySynonymMapper"]}]
Intent recognized as : additional_resources
```

QUERY GENERATED:

```
PREFIX schema1: <http://schema.org/>PREFIX rb: <http://ROBOPROF.org/class#>PREFIX rb_data: <http://ROBOPROF.org/data#>PREFIX dbpedia_o: <http://dbpedia.org/ontology/>
PREFIX dccore: <http://purl.org/dc/elements/1.1/>PREFIX dbpedia_p: <http://dbpedia.org/property/>PREFIX schema_t: <http://schema.org/Thing>PREFIX dcterms: <http://purl.org/dc/terms/>PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>PREFIX foaf: <http://xmlns.com/foaf/0.1/>PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
SELECT DISTINCT ?additionalResources
WHERE {
    ?course a schema1:Course .
    ?course dccore:subject ?subject .
    ?course dbpedia_p:number ?number .
    ?course rdfs:seeAlso ?additionalResources .
    FILTER (?course = <http://ROBOPROF.org/data#COMP6741>)
}
```

RAW RESULT:

```
{"bindings": [{"additionalResources": {"type": "literal", "value": "https://concordiauniversity.on.worldcat.org/oclc/897466408"}, {"additionalResources": {"type": "literal", "value": "https://www-nlp.stanford.edu/IR-book/"}, {"additionalResources": {"type": "literal", "value": "https://concordiauniversity.on.worldcat.org/oclc/960211579"}}, {"additionalResources": {"type": "literal", "value": "https://www.w3.org/People/Ivan/CorePresentations/RDFTutorial/'}}, {"additionalResources": {"type": "literal", "value": "https://www.w3.org/TR/rdf11-primer/'}}, {"additionalResources": {"type": "literal", "value": "https://concordiauniversity.on.worldcat.org/oclc/954339855"}}, {"additionalResources": {"type": "literal", "value": "https://concordiauniversity.on.worldcat.org/oclc/801812987"}}, {"additionalResources": {"type": "literal", "value": "https://concordiauniversity.on.worldcat.org/oclc/314121652'}}]}
```

```
Your input -> I'm looking for additional resources for COMP6741, can you help?
```

```
The additional resources found for COMP6741 is 8:
-> https://concordiauniversity.on.worldcat.org/oclc/897466408
-> https://www-nlp.stanford.edu/IR-book/
-> https://concordiauniversity.on.worldcat.org/oclc/960211579
-> https://www.w3.org/People/Ivan/CorePresentations/RDFTutorial/
-> https://www.w3.org/TR/rdf11-primer/
-> https://concordiauniversity.on.worldcat.org/oclc/954339855
-> https://concordiauniversity.on.worldcat.org/oclc/801812987
-> https://concordiauniversity.on.worldcat.org/oclc/314121652
```

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

```
Your input -> Retrieve the course materials for lecture 02 in SOEN6431.
```

```
Entities recognized are: [{"entity": "lecture_number", "start": 42, "end": 44, "confidence_entity": 0.999593198299408, "value": "02", "extractor": "DIETClassifier", "processors": ["EntitySynonymMapper"]}, {"entity": "course", "start": 48, "end": 56, "confidence_entity": 0.999930739402771, "value": "SOEN6431", "extractor": "DIETClassifier", "processors": ["EntitySynonymMapper"]}]
Intent recognized as : course_lecture_contents
```

```

QUERY GENERATED:

PREFIX schema: <http://schema.org/>PREFIX rb: <http://ROBOPROF.org/class#>PREFIX rb_data: <http://ROBOPROF.org/data#>PREFIX dbpedia_o: <http://dbpedia.org/ontology/>
PREFIX dbcore: <http://purl.org/dc/elements/1.1/>PREFIX dbpedia_p: <http://dbpedia.org/property/>PREFIX schema_t: <http://schema.org/Thing>PREFIX dcterms: <http://purl.org/dc/terms/>PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>PREFIX foaf: <http://xmlns.com/foaf/0.1/>PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
SELECT ?ContentsAvailable
WHERE {
  BIND(rb_data:SOEN6431_Lecture02_LectureContent AS ?lectureContent)
  {
    ?lectureContent rdf:type rb:LectureContent .
    ?lectureContent rbs:slides ?contents .
    BIND(?contents AS ?ContentsAvailable)
  }
  UNION
  {
    ?lectureContent rdf:type rb:LectureContent .
    ?lectureContent rbi:readings ?contents .
    BIND(?contents AS ?ContentsAvailable)
  }
  UNION
  {
    ?lectureContent rdf:type rb:LectureContent .
    ?lectureContent rbi:worksheets ?contents .
    BIND(?contents AS ?ContentsAvailable)
  }
}

```

RAW RESULT:

```
{"bindings": [{"ContentsAvailable": {"type": "literal", "value": "file:///Users/sharanyu/Desktop/CleanIS/data/course_materials/SOEN6431/Lectures/Lecture02/slides/SoftwareTraceability02.pdf"}]}
```

Your input -> **Retrieve the course materials for lecture 02 in SOEN6431.**
The Content found for SOEN6431 in Lecture02:
--> file:///Users/sharanyu/Desktop/CleanIS/data/course_materials/SOEN6431/Lectures/Lecture02/slides/SoftwareTraceability02.pdf

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

Your input -> **Provide the reading study materials for wikidata available in COMP6741.**

```

Entities recognized are: [{"entity": "topic", "start": 40, "end": 48, "confidence_entity": 0.9873314499855042, "value": "wikidata", "extractor": "DIETClassifier"}, {"entity": "course", "start": 62, "end": 70, "confidence_entity": 0.9998723268508911, "value": "COMP6741", "extractor": "DIETClassifier", "processors": ["EntitySynonymMapper"]}]
Intent recognized as : course_content_for_topics

```

QUERY GENERATED:

```

PREFIX schema: <http://schema.org/>PREFIX rb: <http://ROBOPROF.org/class#>PREFIX rb_data: <http://ROBOPROF.org/data#>PREFIX dbpedia_o: <http://dbpedia.org/ontology/>
PREFIX dbcore: <http://purl.org/dc/elements/1.1/>PREFIX dbpedia_p: <http://dbpedia.org/property/>PREFIX schema_t: <http://schema.org/Thing>PREFIX dcterms: <http://purl.org/dc/terms/>PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>PREFIX foaf: <http://xmlns.com/foaf/0.1/>PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
SELECT DISTINCT ?course ?topicLabel ?source
WHERE {
  ?course a schema:Course .
  {
    ?course rb:containsLab ?event . BIND("Lab" AS ?eventType)
    UNION
    ?course rb:containsLecture ?event . BIND("Lecture" AS ?eventType)
  }
  ?event rb:coversTopic ?topic .
  ?topic rdfs:label ?topicLabel .
  ?topic dcterms:source ?source .
  FILTER regex(?topicLabel, "wikidata", "i")
  FILTER (?course = <http://ROBOPROF.org/data#COMP6741>)
}

```

RAW RESULT:

```
{"bindings": [{"course": {"type": "uri", "value": "http://ROBOPROF.org/data#COMP6741"}, "topicLabel": {"type": "literal", "value": "wikidata"}, "source": {"type": "uri", "value": "file:///Users/sharanyu/Desktop/CleanIS/data/course_materials/COMP6741/Labs/Lab05/COMP-474-2234-UU_Lab%2520Session_05_Moodle.pdf"}, {"course": {"type": "uri", "value": "http://ROBOPROF.org/data#COMP6741"}, "topicLabel": {"type": "literal", "value": "wikidata"}, "source": {"type": "uri", "value": "file:///Users/sharanyu/Desktop/CleanIS/data/course_materials/COMP6741/Lectures/Lecture03/worksheets/worksheet03.pdf"}, {"course": {"type": "uri", "value": "http://ROBOPROF.org/data#COMP6741"}, "topicLabel": {"type": "literal", "value": "wikidata"}, "source": {"type": "uri", "value": "file:///Users/sharanyu/Desktop/CleanIS/data/course_materials/COMP6741/Labs/Lab02/COMP-474-2234-UU_Lab%2520Session_02_Moodle.pdf"}, {"course": {"type": "uri", "value": "http://ROBOPROF.org/data#COMP6741"}, "topicLabel": {"type": "literal", "value": "wikidata"}, "source": {"type": "uri", "value": "file:///Users/sharanyu/Desktop/CleanIS/data/course_materials/COMP6741/Labs/Lab04/COMP-474-2234-UU_Lab%2520Session_04_Moodle.pdf"}, {"course": {"type": "uri", "value": "http://ROBOPROF.org/data#COMP6741"}, "topicLabel": {"type": "literal", "value": "wikidata"}, "source": {"type": "uri", "value": "file:///Users/sharanyu/Desktop/CleanIS/data/course_materials/COMP6741/Lectures/Lecture01/worksheets/worksheet01.pdf"}, {"course": {"type": "uri", "value": "http://ROBOPROF.org/data#COMP6741"}, "topicLabel": {"type": "literal", "value": "wikidata"}, "source": {"type": "uri", "value": "file:///Users/sharanyu/Desktop/CleanIS/data/course_materials/COMP6741/Labs/Lab03/COMP-474-2234-UU_Lab%2520Session_03_Moodle.pdf"}, {"course": {"type": "uri", "value": "http://ROBOPROF.org/data#COMP6741"}, "topicLabel": {"type": "literal", "value": "wikidata"}, "source": {"type": "uri", "value": "file:///Users/sharanyu/Desktop/CleanIS/data/course_materials/COMP6741/Lectures/Lecture02/slides/KnowledgeBase02.pdf"}}]}
```

Your input -> **Provide the reading study materials for wikidata available in COMP6741.**

```

I found 7 Contents found for wikidata in course- COMP6741:
--> COMP6741 -> wikidata ----- file:///Users/sharanyu/Desktop/CleanIS/data/course_materials/COMP6741/Labs/Lab05/COMP-474-2234-UU_Lab%2520Session_05_Moodle.pdf
--> COMP6741 -> wikidata ----- file:///Users/sharanyu/Desktop/CleanIS/data/course_materials/COMP6741/Lectures/Lecture03/worksheets/worksheet03.pdf
--> COMP6741 -> wikidata ----- file:///Users/sharanyu/Desktop/CleanIS/data/course_materials/COMP6741/Labs/Lab02/COMP-474-2234-UU_Lab%2520Session_02_Moodle.pdf
--> COMP6741 -> wikidata ----- file:///Users/sharanyu/Desktop/CleanIS/data/course_materials/COMP6741/Labs/Lab04/COMP-474-2234-UU_Lab%2520Session_04_Moodle.pdf
--> COMP6741 -> wikidata ----- file:///Users/sharanyu/Desktop/CleanIS/data/course_materials/COMP6741/Lectures/Lecture01/worksheets/worksheet01.pdf
--> COMP6741 -> wikidata ----- file:///Users/sharanyu/Desktop/CleanIS/data/course_materials/COMP6741/Labs/Lab03/COMP-474-2234-UU_Lab%2520Session_03_Moodle.pdf
--> COMP6741 -> wikidata ----- file:///Users/sharanyu/Desktop/CleanIS/data/course_materials/COMP6741/Lectures/Lecture02/slides/KnowledgeBase02.pdf

```

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

Your input -> What knowledge areas does COMP6741 cover?

```
Entities recognized are: [{"entity': 'course', 'start': 26, 'end': 34, 'confidence_entity': 0.9998099207878113, 'value': 'COMP6741', 'extractor': 'DIETClassifier', 'processors': ['EntitySynonymMapper']}]
Intent recognized as : obtaining_topics_after_passing_course
```

```
PREFIX schema: <http://schema.org/>
PREFIX rb: <http://ROBOPROF.org/class#>
PREFIX rb_data: <http://ROBOPROF.org/data#>
PREFIX dbpedia_o: <http://dbpedia.org/ontology/>
PREFIX dbcore: <http://purl.org/dc/elements/1.1/>
PREFIX dbpedia_p: <http://dbpedia.org/property/>
PREFIX schema_t: <http://schema.org/Thing>
PREFIX dcterms: <http://purl.org/dc/terms/>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>

SELECT DISTINCT ?course ?topic ?topicLabel
WHERE {
  ?course a schema:Course .
  {
    ?course rb:containsLab ?event .
    BIND("Lab" AS ?eventType)
  }
  UNION
  {
    ?course rb:containsLecture ?event .
    BIND("Lecture" AS ?eventType)
  }
  ?event rb:coversTopic ?topic .
  ?topic rdfs:label ?topicLabel .
  FILTER (?course = <http://ROBOPROF.org/data#COMP6741>)
}
```

RAW RESULT:

```
{'bindings': [{}{'course': {'type': 'uri', 'value': 'http://ROBOPROF.org/data#COMP6741'}, 'topic': {'type': 'uri', 'value': 'http://ROBOPROF.org/data#MovieLens'}, 'topicLabel': {'type': 'literal', 'value': 'movielens'}}, {}{'course': {'type': 'uri', 'value': 'http://ROBOPROF.org/data#COMP6741'}, 'topic': {'type': 'uri', 'value': 'http://ROBOPROF.org/data#This_Week_(American_TV_program)'}, 'topicLabel': {'type': 'literal', 'value': 'this week'}}, {}{'course': {'type': 'uri', 'value': 'http://ROBOPROF.org/data#SPARQL'}, 'topicLabel': {'type': 'literal', 'value': 'sparql'}}, {}{'course': {'type': 'uri', 'value': 'http://ROBOPROF.org/data#Knowledge_Graph'}, 'topicLabel': {'type': 'literal', 'value': 'knowledge graph'}}, {}{'course': {'type': 'uri', 'value': 'http://ROBOPROF.org/data#COMP6741'}, 'topicLabel': {'type': 'literal', 'value': 'Microsoft_Window'}}, {}{'course': {'type': 'uri', 'value': 'http://ROBOPROF.org/data#Windows'}, 'topicLabel': {'type': 'literal', 'value': 'windows'}}, {}{'course': {'type': 'uri', 'value': 'http://ROBOPROF.org/data#Hypertext_Transfer_Protocol'}, 'topicLabel': {'type': 'literal', 'value': 'http'}}, {}{'course': {'type': 'uri', 'value': 'http://ROBOPROF.org/data#RDF_query_language'}, 'topicLabel': {'type': 'literal', 'value': 'rdf query language'}}, {}{'course': {'type': 'uri', 'value': 'http://ROBOPROF.org/data#Linkedin'}, 'topicLabel': {'type': 'literal', 'value': 'linkedin'}}, {}{'course': {'type': 'uri', 'value': 'http://ROBOPROF.org/data#COMP6741'}, 'topicLabel': {'type': 'uri', 'value': 'http://ROBOPROF.org/data#COMP6741'}}, {}{'course': {'type': 'uri', 'value': 'http://ROBOPROF.org/data#Siri'}, 'topicLabel': {'type': 'literal', 'value': 'siri'}}, {}{'course': {'type': 'uri', 'value': 'http://ROBOPROF.org/data#Apple_Siri'}, 'topicLabel': {'type': 'literal', 'value': 'apple siri'}}, {}{'course': {'type': 'uri', 'value': 'http://ROBOPROF.org/data#Berlin'}, 'topicLabel': {'type': 'literal', 'value': 'berlin'}}, {}{'course': {'type': 'uri', 'value': 'http://ROBOPROF.org/data#Artificial_intelligence'}, 'topicLabel': {'type': 'literal', 'value': 'intelligent systems'}}, {}{'course': {'type': 'uri', 'value': 'http://ROBOPROF.org/data#COMP6741'}, 'topicLabel': {'type': 'literal', 'value': 'artificial intelligence'}}, {}{'course': {'type': 'uri', 'value': 'http://ROBOPROF.org/data#COMP6741'}, 'topicLabel': {'type': 'uri', 'value': 'http://ROBOPROF.org/data#COMP6741'}}, {}{'course': {'type': 'uri', 'value': 'http://ROBOPROF.org/data#Montreal'}, 'topicLabel': {'type': 'literal', 'value': 'montreal'}}, {}{'course': {'type': 'uri', 'value': 'http://ROBOPROF.org/data#Azimuth'}, 'topicLabel': {'type': 'literal', 'value': 'ta'}}, {}{'course': {'type': 'uri', 'value': 'http://ROBOPROF.org/data#COMP6741'}, 'topicLabel': {'type': 'uri', 'value': 'v'}}]
```

Your input -> What knowledge areas does COMP6741 cover?

The Topics obtained after completing the course COMP6741 are:

```
--> COMP6741 --> MovieLens : movielens
--> COMP6741 --> This_Week_(American_TV_program) : this week
--> COMP6741 --> SPARQL : sparql
--> COMP6741 --> Knowledge_Graph : knowledge graph
--> COMP6741 --> Microsoft_Windows : windows
--> COMP6741 --> Hypertext_Transfer_Protocol : http
--> COMP6741 --> RDF_query_language : rdf query language
--> COMP6741 --> LinkedIn : linkedin
--> COMP6741 --> Siri : siri
--> COMP6741 --> Siri : apple_siri
--> COMP6741 --> Berlin : berlin
--> COMP6741 --> Artificial_intelligence : intelligent systems
--> COMP6741 --> Artificial_intelligence : artificial intelligence
--> COMP6741 --> Artificial_intelligence : ai
--> COMP6741 --> Montreal : montreal
--> COMP6741 --> Azimuth : ta
--> COMP6741 --> Canada : canada
--> COMP6741 --> RDF_Schema : rdfs
--> COMP6741 --> RDF_Schema : rdf schema
--> COMP6741 --> Dewan_Bahasa_dan_Pustaka : dbp
--> COMP6741 --> Netflix : netflix
--> COMP6741 --> Wikidata : wikidata
```

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

Your input -> What was the performance of Sushant Sharma in COMP6741?

```
Entities recognized are: [{"entity': 'firstname', 'start': 28, 'end': 35, 'confidence_entity': 0.9998157620429993, 'value': 'Sushant', 'extractor': 'DIETClassifier', 'processors': ['EntitySynonymMapper']}, {"entity': 'lastname', 'start': 36, 'end': 42, 'confidence_entity': 0.9994391798973083, 'value': 'Sharma', 'extractor': 'DIETClassifier', 'processors': ['EntitySynonymMapper']}, {"entity': 'course', 'start': 46, 'end': 54, 'confidence_entity': 0.999895930290222, 'value': 'COMP6741', 'extractor': 'DIETClassifier', 'processors': ['EntitySynonymMapper']}]
Intent recognized as : student_course_performance
```

```

QUERY GENERATED:

PREFIX schema1: <http://schema.org/>PREFIX rb: <http://ROBOPROF.org/class#>PREFIX rb_data: <http://ROBOPROF.org/data#>PREFIX dbpedia_o: <http://dbpedia.org/ontology/>
PREFIX dbcore: <http://purl.org/dc/elements/1.1/>PREFIX dbpedia_p: <http://dbpedia.org/property/>PREFIX schema_t: <http://schema.org/Thing>PREFIX dcterms: <http://purl.org/dc/terms/>PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>PREFIX foaf: <http://xmlns.com/foaf/0.1/>PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
SELECT (CONCAT(?firstName, " ", ?lastName) AS ?fullName) ?sem ?grade
WHERE {
?student rdf:type foaf:Person ;
    foaf:givenName ?firstName ;
    foaf:familyName ?lastName .
?attempt rb:course ?course ;
    rb:Grade ?grade ;
    rb:Semester ?sem .
?student rb:courseAttempted ?attempt .
    FILTER (?firstName = "Sushant" && ?lastName = "Sharma" && ?course = <http://ROBOPROF.org/data#COMP6741>)
}

```

```

RAW RESULT:

{"bindings": [{"fullName": {"type": "literal", "value": "Sushant Sharma"}, "sem": {"type": "literal", "value": "Winter 2022"}, "grade": {"type": "literal", "value": "F"}}, {"fullName": {"type": "literal", "value": "Sushant Sharma"}, "sem": {"type": "literal", "value": "Winter 2023"}, "grade": {"type": "literal", "value": "C"}]}

```

```

Your input -> How did Edgar Sanchez perform in SOEN6431?
Sorry, I can't seem to find any records of the student Edgar Sanchez for SOEN6431!
Your input -> What was the performance of Sushant Sharma in COMP6741?
Student: Sushant Sharma performance in COMP6741:
-> Sushant Sharma ----> Winter 2022 : F( to be repeated)
-> Sushant Sharma ----> Winter 2023 : C

```

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

```
Your input -> Which students have completed COMP6741?
```

```

Entities recognized are: [{"entity": "course", "start": 30, "end": 38, "confidence_entity": 0.9997738003730774, "value": "COMP6741", "extractor": "DIETClassifier", "processors": ["EntitySynonymMapper"]}]
Intent recognized as : course_performance

```

```

QUERY GENERATED:

PREFIX schema1: <http://schema.org/>PREFIX rb: <http://ROBOPROF.org/class#>PREFIX rb_data: <http://ROBOPROF.org/data#>PREFIX dbpedia_o: <http://dbpedia.org/ontology/>
PREFIX dbcore: <http://purl.org/dc/elements/1.1/>PREFIX dbpedia_p: <http://dbpedia.org/property/>PREFIX schema_t: <http://schema.org/Thing>PREFIX dcterms: <http://purl.org/dc/terms/>PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>PREFIX foaf: <http://xmlns.com/foaf/0.1/>PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
SELECT DISTINCT ?student ?firstName ?lastName ?sem
WHERE {
?student rdf:type foaf:Person ;
    foaf:givenName ?firstName ;
    foaf:familyName ?lastName .
?attempt rb:course ?course ;
    rb:Grade ?grade ;
    rb:Semester ?sem .
?student rb:courseAttempted ?attempt .
    FILTER (
        ?course = <http://ROBOPROF.org/data#COMP6741> &&
        ?grade != "F"
    )
}

```

```

RAW RESULT:

{"bindings": [{"student": {"type": "uri", "value": "http://ROBOPROF.org/data#Brittany_Sath_40228002"}, "firstName": {"type": "literal", "value": "Brittany"}, "lastName": {"type": "literal", "value": "Sath"}, "sem": {"type": "literal", "value": "Winter 2024"}}, {"student": {"type": "uri", "value": "http://ROBOPROF.org/data#Maja_Nicholson_40227997"}, "firstName": {"type": "literal", "value": "Maja"}, "lastName": {"type": "literal", "value": "Nicholson"}, "sem": {"type": "literal", "value": "Fall 2023"}}, {"student": {"type": "uri", "value": "http://ROBOPROF.org/data#Joshua_Lonaker_40227988"}, "firstName": {"type": "literal", "value": "Joshua"}, "lastName": {"type": "literal", "value": "Lonaker"}, "sem": {"type": "literal", "value": "Fall 2023"}}, {"student": {"type": "uri", "value": "http://ROBOPROF.org/data#Sharanyu_Pillai_40227985"}, "firstName": {"type": "literal", "value": "Sharanyu"}, "lastName": {"type": "literal", "value": "Pillai"}, "sem": {"type": "literal", "value": "Fall 2023"}}, {"student": {"type": "uri", "value": "http://ROBOPROF.org/data#Edgar_Sanchez_40228000"}, "firstName": {"type": "literal", "value": "Edgar"}, "lastName": {"type": "literal", "value": "Sanchez"}, "sem": {"type": "literal", "value": "Fall 2023"}}, {"student": {"type": "uri", "value": "http://ROBOPROF.org/data#Sushant_Sharma_40227986"}, "firstName": {"type": "literal", "value": "Sushant"}, "lastName": {"type": "literal", "value": "Sharma"}, "sem": {"type": "literal", "value": "Winter 2023"}]}

```

```

Your input -> Which students have completed COMP6741?
Students who have completed the course COMP6741:
-> http://ROBOPROF.org/data#Brittany_Sath_40228002 ----> Brittany Sath - Winter 2024
-> http://ROBOPROF.org/data#Maja_Nicholson_40227997 ----> Maja Nicholson - Fall 2023
-> http://ROBOPROF.org/data#Joshua_Lonaker_40227988 ----> Joshua Lonaker - Fall 2023
-> http://ROBOPROF.org/data#Sharanyu_Pillai_40227985 ----> Sharanyu Pillai - Fall 2023
-> http://ROBOPROF.org/data#Edgar_Sanchez_40228000 ----> Edgar Sanchez - Fall 2023
-> http://ROBOPROF.org/data#Sushant_Sharma_40227986 ----> Sushant Sharma - Winter 2023

```

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

```
Your input -> I need a academic record for Sharanyu Pillai
```

```
Entities recognized are: [{entity': 'firstname', 'start': 29, 'end': 37, 'confidence_entity': 0.9997931122779846, 'value': 'Sharanyu', 'extractor': 'DIETClassifier', 'processors': ['EntitySynonymMapper']}, {'entity': 'lastname', 'start': 38, 'end': 44, 'confidence_entity': 0.9991644620895386, 'value': 'Pillai', 'extractor': 'DIETClassifier', 'processors': ['EntitySynonymMapper']}]
Intent recognized as : students_transcript
```

QUERY GENERATED:

```
PREFIX schema1: <http://schema.org/>PREFIX rb: <http://ROBOPROF.org/class#>PREFIX rb_data: <http://ROBOPROF.org/data#>PREFIX dbpedia_o: <http://dbpedia.org/ontology/>
PREFIX dbcore: <http://purl.org/dc/elements/1.1/>PREFIX dbpedia_p: <http://dbpedia.org/property/>PREFIX schema_t: <http://schema.org/Thing>PREFIX dcterms: <http://purl.org/dc/terms/>PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>PREFIX foaf: <http://xmlns.com/foaf/0.1/>PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
SELECT ?courseTitle (CONCAT(?subject, ?number) AS ?Course) ?sem ?grade
WHERE {
?student rdf:type foaf:Person ;
foaf:givenName ?firstName ;
foaf:familyName ?lastName .
?student rb:courseAttempted ?attempt .
?attempt rb:course ?course ;
rb:Grade ?grade ;
rb:Semester ?sem .
?course dbcore:title ?courseTitle ;
dbcore:subject ?subject ;
dbpedia_p:number ?number .
FILTER (?firstName = "Sharanyu" && ?lastName = "Pillai")
}
```

RAW RESULT:

```
{'bindings': [{}{'courseTitle': {'type': 'literal', 'value': 'INTELLIGENT SYSTEMS'}, 'Course': {'type': 'literal', 'value': 'COMP6741'}, 'sem': {'type': 'literal', 'value': 'Winter 2022'}, 'grade': {'type': 'literal', 'value': 'F'}}, {'courseTitle': {'type': 'literal', 'value': 'INTELLIGENT SYSTEMS'}, 'Course': {'type': 'literal', 'value': 'COMP6741'}, 'sem': {'type': 'literal', 'value': 'Fall 2023'}, 'grade': {'type': 'literal', 'value': 'A'}}]}]
```

```
Your input -> I need a academic record for Sharanyu Pillai
Here is the transcript for student : Sharanyu Pillai:
=> COMP6741 - INTELLIGENT SYSTEMS -> Winter 2022 - F
=> COMP6741 - INTELLIGENT SYSTEMS -> Fall 2023 - A
```