



Utilizing LLMs and ontologies to query educational knowledge graphs

Christina Antoniou
Aristotle University of Thessaloniki
Thessaloniki, Greece
antoniouc@csd.auth.gr

Nick Bassiliades
Aristotle University of Thessaloniki
Thessaloniki, Greece
nbassili@csd.auth.gr

Abstract

Knowledge Graphs (KGs) provide knowledge and data in a structured format with content from various fields. But the access to the knowledge graphs is done by experienced users, that is, users who know the syntax of the SPARQL language and the KG vocabulary (either in RDF Schema or in OWL) in order to be able to ask questions to exploit the knowledge graphs. However, this requires a lot of time and effort for most of the users, which makes KGs inaccessible to a large number of users. Large Language Models (LLMs) that have appeared recently can provide an alternative way to query knowledge graphs without the need to learn SPARQL and/or know the schema and vocabulary of them, eliminating the time and effort that ordinary users need to spend in order to use them. In this article, we present some experiments and their results illustrating how ChatGPT can help ordinary users to generate SPARQL queries, without knowing SPARQL, to effectively use knowledge graphs and exploit their wealth of data. We experimented with ChatGPT to explore whether it can generate SPARQL queries based on user's natural language input and a given vocabulary (ontology) about an educational knowledge graph. To this end we have devised a specific prompt template. Results indicate that LLMs can indeed help in this direction, given the fact that they are prompted properly, using good English language. We have also discussed some practical lessons learned through this experiment.

CCS Concepts

• **Software and its engineering**; • **Computing methodologies**
→ Artificial intelligence; Artificial intelligence; Natural language processing;

Keywords

ChatGPT, knowledge graphs, RDF, large language model use cases, AI application

ACM Reference Format:

Christina Antoniou and Nick Bassiliades. 2024. Utilizing LLMs and ontologies to query educational knowledge graphs. In *28th Pan-Hellenic Conference on Progress in Computing and Informatics (PCI 2024)*, December 13–15, 2024, Egaleo, Greece. ACM, New York, NY, USA, 9 pages. <https://doi.org/10.1145/3716554.3716598>



This work is licensed under a Creative Commons Attribution International 4.0 License.

PCI 2024, Egaleo, Greece

© 2024 Copyright held by the owner/author(s).

ACM ISBN 979-8-4007-1317-0/24/12

<https://doi.org/10.1145/3716554.3716598>

1 Introduction

Knowledge representation in modern systems often takes the form of RDF (Resource Description Framework) triples, which serve as the foundation of knowledge graphs (KGs) [8]. Prominent examples of knowledge graphs include DBpedia¹, Wikidata², and YAGO³, each facilitating tasks such as text summarization, opinion mining, classification, semantic search, and question answering [4]. Knowledge graphs generally fall into two main categories. The first category comprises datasets containing general knowledge from diverse domains, making them widely recognized. For instance, the encyclopaedic content of Wikipedia is represented in RDF format within DBpedia. The second category focuses on domain-specific knowledge, encompassing datasets in fields such as music (e.g., MusicBrainz⁴, Music Ontology [14]), biomedicine (e.g., SIDER⁵, Diseasesome [19], Drugbank⁶), and geography (LinkedGeoData⁷).

To query and utilize data on the Semantic Web, RDF serves as the standard data model, with SPARQL as the corresponding query language. A Knowledge Graph, adhering to the RDF standard, represents information as a collection of triples: <subject, predicate, object>. These triples form an RDF graph, where vertices correspond to subjects and objects, and edges represent predicates. The RDF graph structure is stored in triplestore repositories and accessed using SPARQL, a query language designed to handle the syntax and semantics of RDF-based data. While SPARQL provides powerful querying capabilities, its complexity and the intricate structure of RDF graphs can pose challenges for non-expert users [22].

An RDF graph's structure can be defined using an ontology, which provides a formal framework for describing relationships and categories of data [3]. Ontologies are typically expressed in RDF Schema (RDFS) or the Web Ontology Language (OWL). RDFS defines basic classes and properties that describe how resources are related, allowing for the creation of a hierarchical structure of concepts (e.g., "Person" is a subclass of "Mammal"). OWL extends this by offering more expressive capabilities, such as specifying complex relationships between classes, cardinality constraints, and logical inferences. Using RDFS or OWL, RDF graphs gain a structured vocabulary and rules, enabling more precise data organization and reasoning over the data.

As mentioned above, in the Semantic Web, only specialized users can effectively use SPARQL to query for data in knowledge

¹<https://www.dbpedia.org/>

²<https://www.wikidata.org/>

³<https://yago-knowledge.org/>

⁴<https://musicbrainz.org/>

⁵<https://bio.tools/sider>

⁶<https://go.drugbank.com/>

⁷<https://linkedgeo.org/>

graphs. Ordinary users, unfamiliar with SPARQL, are unable to perform such queries. Moreover, even if a user knows how to write SPARQL queries, they must also understand the ontology of the knowledge graph in order to use the correct predicates to navigate the graph effectively. One solution to this problem is the use of question answering systems, specifically Semantic Question Answering (SQA) systems. In SQA systems, users can ask questions in natural language and receive answers generated by querying a knowledge graph. These systems address two major challenges: the need to learn a query language like SPARQL and the requirement to know the specific vocabulary of the knowledge base [7].

Research on semantic question answering systems using linked data (or knowledge graphs) is very active and there is a wide variety of methodologies [2]. However, the use of natural language also has disadvantages such as ambiguity. For this reason, these systems have many different steps in order to understand the user's questions and answer the questions. To understand and translate the user's question into a SPARQL question, techniques from natural language processing are used in many systems. Also, the creation and development of such systems is time-consuming and costs a lot. Another solution for the above problem is Large Language Models (LLMs) [18], their most prominent representative being ChatGPT, published by OpenAI⁸. LLMs have a good performance in tasks related to natural language processing and especially question-answering [17]. Also, LLMs significantly improve the performance of artificial intelligence applications that use natural language processing such as chatbots, text generation, and language translation [9].

There is a complementary relationship between LLMs and knowledge graphs [12]. Natural language processing has improved human-machine communication. However, LLMs have some shortcomings such as hallucinations and lack of domain specific knowledge so that many real-world tasks do not perform as expected. These issues can be addressed by knowledge graphs [1], which bring reliable results to the question answering systems [13]. On the other hand, problems of knowledge graphs such as their construction and validation can be addressed by LLMs ([11], [21]). Therefore, there is an interdependent relationship [9].

In this paper we explore one such example interaction between Knowledge Graphs and Large Language Models, that of utilizing LLMs to generate SPARQL queries from natural language user prompts and a formal ontology (as input), so that answers to these questions can be found in KGs. To this end, we experimented with ChatGPT and an educational knowledge graph (EvdoGraph [5]), that contains information about Greek universities, departments, courses and books used in these courses. To this end, we have developed a prompt template that describes the task, including the ontology and the user's question in natural language. In the article, we present the results of these experiments illustrating how ChatGPT can help ordinary users to generate SPARQL queries to effectively use knowledge graphs and exploit their wealth of information. Results indicate that LLMs can indeed help in this direction, given the fact that they are prompted properly, using good English language. We also discuss some practical lessons learned through this experiment.

⁸<https://openai.com/index/chatgpt/>

The paper is structured as follows: In Section 2 related work is presented in a comprehensive manner. Section 3 describes in detail our methodology, experiments and results. Section 4 discusses the lessons we learned for prompting ChatGPT. Finally, Section 5 concludes the papers and discusses some of our future plans for this line of research.

2 Related Work

There are a few examples of works that discuss the use of LLMs for querying KGs using SPARQL. For example, reference [6] introduces a system that translates user questions into accurate federated SPARQL queries over bioinformatics KGs, leveraging LLMs to enhance accuracy and reduce hallucinations. The authors used a dataset of bioinformatics questions and corresponding SPARQL queries. They employed an LLM to translate natural language questions into SPARQL queries. The system demonstrated high accuracy in generating correct SPARQL queries, significantly reducing the error rate compared to traditional methods. The use of LLMs helped in understanding complex queries and generating precise SPARQL queries.

Furthermore, in [10] experiments demonstrating the advantages of blending KGs with LLMs, including the generation of SPARQL queries from natural language inputs, were presented. The experiments were conducted on a variety of KGs, including DBpedia and Wikidata. The LLM was used to generate SPARQL queries from natural language questions. The experiments showed that integrating LLMs with KGs improved the accuracy and relevance of the answers. The system was able to handle complex queries and provide detailed answers by generating accurate SPARQL queries.

Another work in [15] discusses a system that answers bibliographic natural language questions by leveraging LLMs in a few-shot manner to generate SPARQL queries. The authors used a dataset of bibliographic questions and corresponding SPARQL queries. They employed an LLM in a few-shot learning manner to generate SPARQL queries. The system achieved high precision and recall in generating SPARQL queries. The use of LLMs enabled the system to understand nuanced questions and generate accurate queries, improving the overall performance of the question-answering system.

The above publications are only some of such examples, that provide detailed insights into how LLMs can be used to query KGs effectively. The experiments conducted highlight the potential of LLMs in enhancing the accuracy and efficiency of querying knowledge graphs using SPARQL [20].

Our work paves in similar ways to the above, except that we focus on manual tweaking of prompts, instead of building systems or models that are trained by SPARQL-prompts datasets. Furthermore, our purpose is to find qualitative evidence about the ability of LLMs to generate sound and complete complex SPARQL queries, rather than to quantitatively evaluate a specific methodology. Finally, we summarize our findings in practical lessons learned so that other researchers or even casual users can take advantage of.

3 Description of the Experiment with ChatGPT

In this section, we describe in detail the experiment we have performed in order to evaluate the capabilities of ChatGPT in generating SPARQL queries and present some of the results. Experiments were conducted with ChatGPT-4. As inspiration, we used the questions that were designed in the context of the EvdoGraph⁹ GitHub project, which is a knowledge graph that contains information extracted from the EVDOXUS textbook management service for the Greek Universities. The Knowledge Graph of Evdoxus is based on an RDF Schema ontology that is shown in Figure 1. The main entities involved are Universities (Figure 2) that have Departments which have Courses of study, different for each academic year. Courses consist of several Modules and for each Module one or more Books are suggested by the Module instructors. More details about the ontology, the Knowledge Graph and the extraction process can be found in [5].

In the GitHub project there exist 18 competency questions both in natural language and their corresponding queries in the SPARQL language, that were manually handcrafted by the author of [5]. See for example the first question in Figure 3. In our experiment, in this paper, we have used the text of the questions in natural language as prompts (actually parts of prompts) for ChatGPT and we have asked ChatGPT to generate an appropriate SPARQL query to answer each question. The full prompt template we have developed can be seen in Figure 4. Notice that in the actual prompt we include inline the whole text of the RDFS ontology in Turtle (Figure 1) as well as the questions in natural text for each specific query (Table 1 shows some indicative questions). Of course, in several cases we needed to experiment a bit with paraphrasing the natural language question in order to generate a correct SPARQL query, but this is natural since the ontology that the knowledge graph is based on is not always clear to ChatGPT and the queries that should be asked are quite complex in several cases.

Then, after we get back the answer from ChatGPT, we copy the SPARQL query, and we paste and execute it at the SPARQL endpoint for the Evdoxus KG¹⁰. We also do the same for the original SPARQL query (found at the GitHub project) and, finally, we compare the results of the two queries. Figure 5 shows the comparison between the two queries for a sample question, namely the first question found in the GitHub project, which asks to return all modules that a certain book (with Evdoxus code 94700120, namely the book at reference [16]) is used at (actually suggested to be used) during the academic year 2022, along with the corresponding Department and University. As it can be easily seen, the number of answers returned by the two queries is exactly the same (83). However, the two queries slightly differ, because the humanly handcrafted one includes the Module / Department / University names, whereas the ChatGPT-generated one returns just the URIs of the corresponding entities. However, we consider that this is a successful answer since it is easy to tamper with the prompt a bit to get the exact answer. Actually, we have managed to get correct and complete SPARQL queries for all 18 competency questions of the Evdoxus KG project.

⁹<https://github.com/nbassili/EvdoGraph>

¹⁰<http://lod.csd.auth.gr:7200/sparql>; the "Evdoxus" repository should be selected for the pull-down menu at upper-right corner.

4 Discussion

In this section we discuss some lessons we have learnt from this experiment. First of all, the language used for prompting LLMs is quite sensitive sometimes, but some other times the LLM is completely insensitive, meaning that however the prompt changes, the answer is always the same. When the latter happens, the solution is to quit the session and start over again, because the context given to the LLM has already focused the LLM on a certain direction of thought that cannot be "forgotten". So, the first lesson is in order to have trusted findings about the ability of an LLM to generate a correct SPARQL query, the LLM should be prompted once with a cold start.

The second lesson is that some natural language expressions that are easily understood by humans could be easily misunderstood by the LLM. For example, negation is not always easy to comprehend by LLM. Notice, for example, query number 5 (Table 1), where sometimes the 3 bold lines inside the FILTER NOT EXISTS construct were returned (by ChatGPT) outside the negation, not comprehending that the negation includes all the modules of the department for the previous academic year and not just a module. So, the 3rd line should always be included inside the negation.

A third lesson is that it is not easy for ChatGPT to understand exactly all the intricacies of the ontological model just by having the RDF Schema code as input. For example, in query number 7 (Table 1), we needed to explain that in the Evdoxus KG, the modules each academic year are represented by a different Module instance, connected to the corresponding course of studies. In order to uniquely identify the module throughout the different academic years, either the module code or the module title should be used. In query 7, we have chosen to use the module title. However, if we do not put in the prompt the explanatory phrase "*uniquely identified by its title*", ChatGPT generates a SPARQL query without the bold lines which are necessary to ensure that the book is not used in the previous academic year in the same module and not in any module of the course. The same explanatory hint was needed in the prompt of query 13.

An initial unsuccessful attempt we tried did not include in the prompt the ontology inline, but through a URI reference. This resulted in the SPARQL query of Figure 6, where it is obvious that ChatGPT never consulted the ontology but tried to invent the names of the RDF properties from the prompt itself. However, the structure of the query, with 6 triple patterns, was correct! From this we have learnt that ChatGPT cannot read data from external sources, even from publicly available URLs.

5 Conclusions and Future Work

In the Semantic Web, only expert users can effectively access and query knowledge graphs, as they are familiar with both the SPARQL query language and the structure of the KG. Ordinary users, who lack this knowledge, are unable to ask questions and retrieve answers from KGs. Large language models can help address this issue by generating SPARQL queries on behalf of users. However, it is crucial to provide these models with the relevant ontology, as they do not have direct access or knowledge of the KG structure. In this paper we experimented with ChatGPT, and an educational

```

@prefix evdx: <https://w3id.org/evdokus#> .

evdx:AcademicEntity rdf:type rdfs:Class ;

    rdfs:subClassOf rdfs:Resource .

evdx:Book rdf:type rdfs:Class ;

    rdfs:subClassOf evdx:LearningEntity .

evdx:Course rdf:type rdfs:Class ;

    rdfs:subClassOf evdx:LearningEntity .

evdx:Department rdf:type rdfs:Class ;

    rdfs:subClassOf evdx:AcademicEntity .

evdx:LearningEntity rdf:type rdfs:Class ;

    rdfs:subClassOf rdfs:Resource .

evdx:Module rdf:type rdfs:Class ;

    rdfs:subClassOf evdx:LearningEntity .

evdx:University rdf:type rdfs:Class ;

    rdfs:subClassOf evdx:AcademicEntity .

evdx:hasBook rdf:type rdf:Property ;

    rdfs:domain evdx:Module ; rdfs:range evdx:Book .

evdx:hasCode rdf:type rdf:Property ;

    rdfs:domain evdx:LearningEntity ; rdfs:range xsd:string .

evdx:hasCourse rdf:type rdf:Property ;

    rdfs:domain evdx:Department ; rdfs:range evdx:Course .

evdx:hasDepartment rdf:type rdf:Property ;

```

Figure 1: The Evdokus ontology

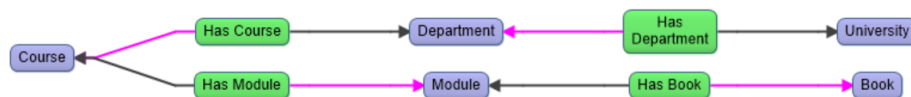


Figure 2: Relations among the main Evdokus ontology entities.

Table 1: Indicative prompts and ChatGPT-generated SPARQL queries.

Prompt No	Question (ChatGPT)	SPARQL (ChatGPT)
1	Return all modules for the courses of the year 2022 that the book with code 94700120 is used in, along with the Department and the University.	<pre> SELECT ?university ?department ?module WHERE { ?university evdx:hasDepartment ?department . ?department evdx:hasCourse ?course . ?course evdx:year "2022"^^xsd:integer ; evdx:hasModule ?module . ?module evdx:hasBook ?book . ?book evdx:hasCode "94700120" . } </pre>
5	Which Department uses, in any of its Modules, the book with code 102070469 in the course of the academic year 2022, while they did not use the book in any module of the same department in the previous academic year 2021. In the answer include also the University of the Department and the Modules.	<pre> SELECT ?university ?department ?module WHERE { ?university evdx:hasDepartment ?department . ?department evdx:hasCourse ?course . ?course evdx:hasModule ?module . ?module evdx:hasBook ?book . ?book evdx:hasCode "102070469" . ?course evdx:year 2022 . FILTER NOT EXISTS { ?department evdx:hasCourse ?coursePrev . ?coursePrev evdx:year 2021 . ?coursePrev evdx:hasModule ?modulePrev . ?modulePrev evdx:hasBook ?bookPrev . ?bookPrev evdx:hasCode "102070469" . } } </pre>
7	Which Module uses the book 102070469 in the course of the academic year 2022, while the same Module (uniquely identified by its title) did not use the book in the course of previous academic year 2021. In the answer include also the Department and the University of the Module.	<pre> SELECT ?universityName ?departmentName ?moduleTitle WHERE { ?university evdx:hasDepartment ?department . ?department evdx:hasCourse ?course2022 . ?course2022 evdx:hasModule ?module2022 . ?module2022 evdx:hasBook ?book . ?book evdx:hasCode "102070469" . ?course2022 evdx:year 2022 . ?module2022 evdx:title ?moduleTitle . ?university evdx:name ?universityName . ?department evdx:name ?departmentName . FILTER NOT EXISTS { ?department evdx:hasCourse ?course2021 . ?course2021 evdx:year 2021 ; evdx:hasModule ?module2021 . ?module2021 evdx:title ?moduleTitle ; evdx:hasBook ?book2021 . ?book2021 evdx:hasCode "102070469" . } } </pre>
8	Compare how many Modules, of how many Departments and how many Universities each of the two books with codes 94700120 and 102070469 are used in a course of the academic year 2022	<pre> SELECT ?bookCode (COUNT(DISTINCT ?module) AS ?moduleCount) (COUNT(DISTINCT ?department) AS ?departmentCount) (COUNT(DISTINCT ?university) AS ?universityCount) WHERE { ?course evdx:year 2022 ; evdx:hasModule ?module . ?module evdx:hasBook ?book . ?book evdx:hasCode ?bookCode . FILTER (?bookCode IN ("94700120", "102070469")) ?department evdx:hasCourse ?course . ?university evdx:hasDepartment ?department . } GROUP BY ?bookCode </pre>

9	Which modules use the book with code 102070469 in the academic year 2022 and the same modules do not use the book with code 94700120 in the same academic year. Include also the names of Universities and Departments.	<pre> SELECT ?universityName ?departmentName ?moduleName WHERE { ?university evdx:hasDepartment ?department . ?university evdx:name ?universityName . ?department evdx:name ?departmentName . ?department evdx:hasCourse ?course . ?course evdx:year "2022"^^xsd:integer . ?course evdx:hasModule ?module . ?module evdx:hasBook ?book102070469 . ?module evdx:title ?moduleName . ?book102070469 evdx:hasCode "102070469" . FILTER NOT EXISTS { ?module evdx:hasBook ?book94700120 . ?book94700120 evdx:hasCode "94700120" . } }</pre>
13	Which Departments, of which Universities and of which Modules and how many Modules, have started using books with code 102070469 and 94700120 of the academic year 2022. Notice that a module is uniquely identified by its title, but not from its URI, which is different for every module instance per year.	<pre> SELECT ?universityName ?departmentName ?moduleTitle (COUNT(DISTINCT ?module) AS ?numModules) WHERE { ?university evdx:hasDepartment ?department . ?department evdx:hasCourse ?course . ?course evdx:hasModule ?module . ?course evdx:year 2022 . ?module evdx:hasBook ?book1 . ?book1 evdx:hasCode "102070469" . ?module evdx:hasBook ?book2 . ?book2 evdx:hasCode "94700120" . ?university evdx:name ?universityName . ?department evdx:name ?departmentName . ?module evdx:title ?moduleTitle . }</pre>
16	Which Universities do not have any Department with a course in the academic year 2022 with any module?	<pre> GROUP BY ?universityName ?departmentName ?moduleTitle SELECT ?universityName WHERE { # Find all universities ?university rdf:type evdx:University ; evdx:name ?universityName . OPTIONAL { ?university evdx:hasDepartment ?department . ?department evdx:hasCourse ?course . ?course evdx:year 2022 . ?course evdx:hasModule ?module . } FILTER(!BOUND(?module)) }</pre>
18	Return a list of all books and their title of the year 2022 and the number of departments and number of modules that use the specified books. The order of ranking should be in descending order based on the count of the department and module.	<pre> SELECT ?book ?bookTitle (COUNT(DISTINCT ?department) AS ?departmentCount) (COUNT(DISTINCT ?module) AS ?moduleCount) WHERE { ?module evdx:hasBook ?book ; evdx:semester ?semester ; evdx:title ?moduleTitle . ?course evdx:hasModule ?module ; evdx:year "2022"^^xsd:integer . ?book evdx:title ?bookTitle . ?department evdx:hasCourse ?course . } GROUP BY ?book ?bookTitle ORDER BY DESC(?departmentCount) DESC(?moduleCount)</pre>

SPARQL queries

The following SPARQL queries implement several Competency Questions for the Evdoxus Ontology-Knowledge Graph. These competency questions have been derived from the various predicates of the [EvdoStats](#) project.

1. Return all modules that the book is used, along with the Department and the University

```
PREFIX evdx: <https://w3id.org/evdoxus#>
select (?un as ?University) (?dn as ?Department) (?mt as ?Module)
where {
  ?s a evdx:Book .
  VALUES ?code {"94700120"}
  ?s evdx:hasCode ?code .
  ?m a evdx:Module .
  ?m evdx:title ?mt .
  ?m evdx:hasBook ?s .
  ?c a evdx:Course .
  ?c evdx:year 2022 .
  ?c evdx:hasModule ?m .
  ?d a evdx:Department .
  ?d evdx:hasCourse ?c .
  ?d evdx:name ?dn .
  ?u a evdx:University .
  ?u evdx:hasDepartment ?d .
  ?u evdx:name ?un .
}
```

Inside VALUES multiple book IDs can be used (e.g. various editions of the same book).

Figure 3: Sample question from the Evdoxus GitHub project.

The following is an RDF Schema ontology that describes entities about Universities, Departments, Courses, Modules and Books.

<HERE GOES THE ONTOLOGY (Figure 1)>

Can you create a SPARQL query that answers the following question?

<HERE GOES THE PROMPT QUESTION IN NATURAL LANGUAGE (Table 1)>

In the SPARQL query, you must use the exact predicate/property names that you can find in the above ontology, but not all of them, just the ones mentioned in the statement above.

Figure 4: The prompt template.

knowledge graph and we have presented the results of these experiments illustrating how ChatGPT can help ordinary users to generate SPARQL queries to effectively query knowledge graphs. To this end, we have developed a prompt template that describes the task, including the ontology and the user’s question in natural language. Results indicate that LLMs can indeed help in this direction, given the fact that they are prompted properly, using good English language. We have also discussed some practical lessons learned through this experiment.

Some of the limitations of this study are that all the experimentation is done manually, on a single domain / KG and using a single LLM. Therefore, in the future, we would like to experiment with more LLMs and more domains / KGs, so that our methodology and results become more robust. Furthermore, we would like to build a system that automates this methodology and also generalizes it so that it can be used in several other domains. Finally, automating tests with several questions would help to assess the soundness and completeness of the methodology faster.

References

- [1] Agrawal, G., Kumarage, Th., Alghamdi, Z., Liu, H. 2024. Can Knowledge Graphs Reduce Hallucinations in LLMs? : A Survey. In Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers), pages 3947–3960, Mexico City, Mexico. Association for Computational Linguistics.
- [2] Antoniou, C., Bassiliades, N. 2022. A Survey on Semantic Question Answering Systems. The Knowledge Engineering Review, vol. 37, p. e2, <https://doi.org/10.1017/S0269888921000138>
- [3] Antoniou, G., Groth, P., Van Harmelen, F., & Hoekstra, R. 2012. A Semantic Web primer (3rd edition). The MIT Press.
- [4] Atzori, M., Gao, S., Mazzeo, G. M. & Zaniolo, C. 2016. Answering end-user questions, queries and searches on Wikipedia and its history. *IEEE Data Engineering* 39(3), 85–96.
- [5] Bassiliades, N. (2023). EvdoGraph: A Knowledge Graph for the EVDOXUS Textbook Management Service for Greek Universities. In Proc. of the 15th Inter. Joint Conf. on Knowledge Discovery, Knowledge Engineering and Knowledge Management - KEOD, SciTePress, pp 17–28.
- [6] Emonet, V., Bolleman, J., Duvaud, S., de Farias, T. M., & Sima, A. C. 2024. LLM-based SPARQL Query Generation from Natural Language over Federated Knowledge Graphs. <https://arxiv.org/abs/2410.06062>
- [7] Höffner, K., Walter, S., Marx, E., Usbeck, R., Lehmann, J. & Ngonga Ngomo, A. C. 2017. Survey on challenges of question answering in the semantic web. *Semantic Web* 8(6), 895–920.
- [8] Hogan, A., et al. 2021. Knowledge Graphs. *ACM Comput. Surv.*, 54(4).
- [9] Kau, A., He, X., Nambissan, A., Astudillo, A., Yin, H., & Aryani, A. (2024). Combining Knowledge Graphs and Large Language Models. *arXiv preprint*

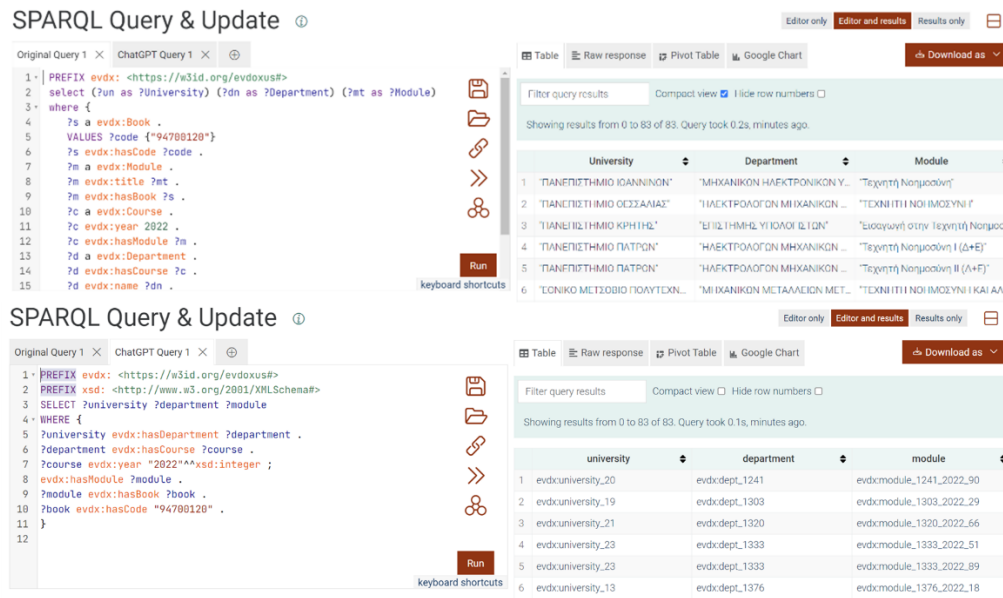


Figure 5: Comparison between original and ChatGPT SPARQL query for a sample question.

Prompt:

In the following URL <https://w3id.org/evdoxus> there is an RDF Schema ontology that describes entities about Universities, Departments, Courses, Modules and Books. Can you create a SPARQL query that answers the following question? Return all modules for the courses of the year 2022 that the book with code 94700120 is used in, along with the Department and the University. In the SPARQL query, you must use the exact predicate/property names that you can find in the above ontology, but not all of them, just the ones mentioned in the statement above.

Answer:

```

PREFIX evdoxus: <https://w3id.org/evdoxus#>

PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>

SELECT ?module ?department ?university

WHERE {

    ?course evdoxus:hasYear "2022" ;

    evdoxus:hasModule ?module .

```

Figure 6: Initial unsuccessful attempt.

arXiv:2407.06564.

- [10] Koohborfardhaghghi, S., De Geyter, G., Kaliner, E. 2024. Unlocking the Power of LLM-Based Question Answering Systems: Enhancing Reasoning, Insight, and Automation with Knowledge Graphs. In: Abraham, A., Bajaj, A., Hanne, T. (eds) Intelligent Systems Design and Applications. ISDA 2023. Lecture Notes in Networks and Systems, vol 1052. Springer, Cham. https://doi.org/10.1007/978-3-031-64776-5_16
- [11] Meyer, L. P., Stadler, C., Frey, J., Radtke, N., Junghanns, K., Meissner, R., ... & Martin, M. (2023, June). Llm-assisted knowledge graph engineering: Experiments

with chatgpt. In *Working conference on Artificial Intelligence Development for a Resilient and Sustainable Tomorrow* (pp. 103-115). Wiesbaden: Springer Fachmedien Wiesbaden.

- [12] Pan, S., Luo, L., Wang, Y., Chen, C., Wang, J., & Wu, X. (2024). Unifying large language models and knowledge graphs: A roadmap. *IEEE Transactions on Knowledge and Data Engineering*.
- [13] Pusch, L., & Conrad, T. O. (2024). Combining LLMs and Knowledge Graphs to Reduce Hallucinations in Question Answering. *arXiv preprint arXiv:2409.04181*.
- [14] Raimond, Y., Abdallah, S., Sandler, M., Giasson, F. 2007. The music ontology. In: *Proceedings of the International Conference on Music Information Retrieval*, pp.

- 417–422.
- [15] Taffa, T. A., Usbeck, R., Semantic Systems, Universität Hamburg, & Leuphana Universität Lüneburg. 2023. Leveraging LLMs in scholarly knowledge Graph question answering. In *Scholarly-QALD-23: Scholarly QALD Challenge at The 22nd International Semantic Web Conference (ISWC 2023)*, November 6 – 10, 2023, Athens, Greece.. <https://ceur-ws.org/Vol-3592/paper5.pdf>
 - [16] Vlahavas, I., Kefalas, P., Bassiliades, N., Kokkoras, F., & Sakellariou, I. (2020). *Artificial Intelligence*, 4th edition. Thessaloniki: University of Macedonia Press.
 - [17] Wang, C., Xu, Y., Peng, Z., Zhang, C., Chen, B., Wang, X., ... & An, B. (2023). keqing: knowledge-based question answering is a nature chain-of-thought mentor of LLM. *arXiv preprint arXiv:2401.00426*.
 - [18] Wang, Z., Chu, Z., Doan, T.V. *et al*. History, development, and principles of large language models: an introductory survey. *AI Ethics* (2024). <https://doi.org/10.1007/s43681-024-00583-7>
 - [19] Wysocki, K., & Ritter, L. (2011). Diseasome: an approach to understanding gene-disease interactions. *Annual review of nursing research*, 29, 55–72. <https://doi.org/10.1891/0739-6686.29.55>.
 - [20] Zhang, M., Yang, G., Liu, Y. *et al* .2024. Knowledge graph accuracy evaluation: an LLM-enhanced embedding approach. *Int J Data Sci Anal*. <https://doi.org/10.1007/s41060-024-00661-3>
 - [21] Zhu, Y., Wang, X., Chen, J. *et al*. LLMs for knowledge graph construction and reasoning: recent capabilities and future opportunities. *World Wide Web* 27, 58 (2024). <https://doi.org/10.1007/s11280-024-01297-w>
 - [22] Zou, L., Huang, R., Wang, H., Yu, J. X., He, W. & Zhao, D. 2014. Natural language question answering over RDF: a graph data driven approach. In *Proceedings of the 2014 ACM SIGMOD International Conference on Management of Data*, 313–324.