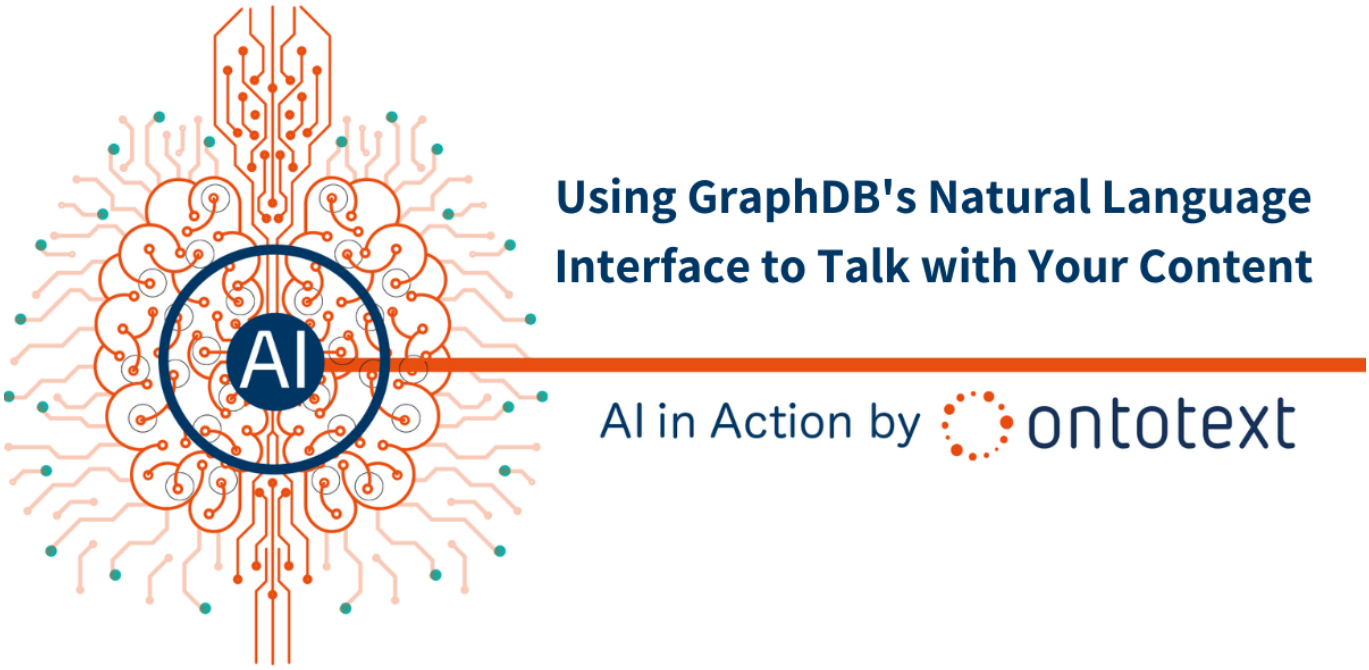


This is part of Ontotext's AI-in-Action initiative aimed at enabling data scientists and engineers to benefit from the AI capabilities of our products.

Ontotext is a knowledge graph company. We use ontologies to model all our enterprise data. The Ontotext Knowledge Graph (OTKG) is an internal project that models all of our important information using semantic web standards.

In this blog post, we will use a subset of the knowledge graph that models all of our:

- website content
- research projects
- reference taxonomies to model industries, applications and product capabilities

Ontotext Knowledge Graph

The OTKG's use of the schema.org schema, which was created by a consortium of major search engines to describe a broad variety of published information and activities. This gives

Copyright: Ontotext New York | Sofia | Basel
Reach out to us: info@ontotext.com

the OTKG greater interoperability with a range of existing data sources and new ones as they come up. As a side effect, all of Ontotext's website uses SEO optimized metadata, which reduces the ambiguity over the published data. The following shows a sample of the data in the OTKG:

```
@prefix s:   <http://schema.org/> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#>

<https://kg.ontotext.com/resource/researchProject/kconnect>
  a s:Organization, s:ResearchProject, s:Thing;
  s:alternateName "KConnect";
  s:description """The main objective of the project is to create a
medical
    text Data-Value Chain with a critical mass of participating
companies using
    cutting-edge commercial cloud-based services for multilingual
Semantic
    Annotation, Semantic Search and Machine Translation for Electronic
Health
    Records and medical publications.""";
  s:duration "P30M"^^xsd:duration;
  s:endDate "2017-07-31"^^xsd:date;
  s:funding <https://kg.ontotext.com/resource/grant/h2020>;
  s:logo
<https://github.com/VladimirAlexiev/onto-fp/raw/master/images/KConnect
.png>;
  s:mainEntityOfPage <http://www.kconnect.eu/>,
                     <https://cordis.europa.eu/project/id/644753>,
<https://www.ontotext.com/knowledgehub/current/kconnect/>;
  s:name """Khresmoi Multilingual Medical Text Analysis, Search and
Machine
        Translation Connected in a Thriving Data-Value Chain""";
  s:sameAs <http://www.wikidata.org/entity/Q104438297>;
  s:startDate "2015-02-01"^^xsd:date .
```

Copyright: Ontotext New York | Sofia | Basel
Reach out to us: info@ontotext.com

Figure 1: Research project part of Ontotext Knowledge Graph.

Use case

Once you have all this data in a knowledge graph, how do you get it out in a form that can contribute to your goals? One option is SPARQL, the W3C standard query language for semantic web data that is fully supported by Ontotext GraphDB. Instead of learning the syntax of a query language, though, many people would rather just ask natural language questions about the content of a knowledge graph.

For something like the OTKG, this could include requests such as:

- "Which GraphDB features were released when?"
- "Give me a summary of what this new feature does and how it works."
- Who has Ontotext worked with in Japan?"
- "Can knowledge graphs help a digital twin system?"
- "Has Ontotext worked on any oncology projects?"

Ontotext's Talk to Your Graph feature is a chat interface driven by a large language model (LLM) that lets you use natural language to ask questions about data in your knowledge graphs. In this blog post, we're going to see how to use a GraphDB connector for ChatGPT and the Weaviate vector database to set up the Talk to Your Graph feature so that we can get answers from the OTKG by asking natural language questions.

The final workflow we will implement is as follows:

1. Define a ChatGPT Connector to translate the graph model into a vector database
2. Index all OTKG data into the vector space compatible with OpenAI's ChatGPT
3. Ask a question through the Talk To Your Graph interface
4. Generate a prompt to OpenAI's API
5. Register the vector database as a context provider
6. Process the LLM query
7. Retrieve contextual information from OTKG
8. Return an answer to the answer

Copyright: Ontotext New York | Sofia | Basel
Reach out to us: info@ontotext.com

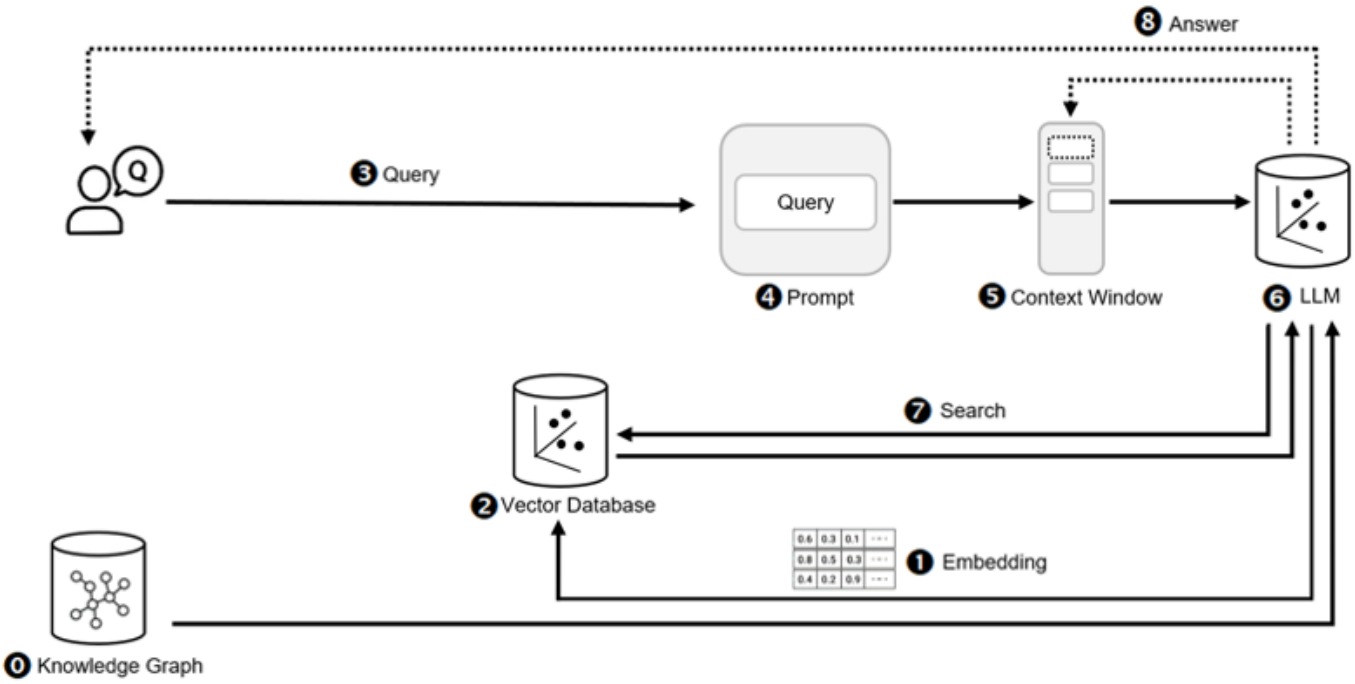


Figure 2: Retrieval Augmented Generation (RAG) with GraphDB and a vector database

The main benefits of this architectural approach are that first the vector database is fully in sync with the graph database. After each GraphDB update, all changes will be incrementally synchronized in a transactionally safe way into the vector database. Second, the LLM context window will not be limited by the size of the prompt, since the LLM will pull the data from the vector database.

Setting up

To set up all components you need Git and Docker installed locally on the machine and a valid OpenAI API key to connect to the LLM. We start by downloading our docker-compose yaml. This contains all you need to start talking to your graph.

Generate an authentication token for the Weaviate database using the online JSON Web Tokens tool https://jwt.io/ and paste this into the payload field on the right:

{

Copyright: Ontotext New York | Sofia | Basel
Reach out to us: info@ontotext.com

```
 "sub": "1234567890",
 "name": "Test",
 "iat": 1694775299
}
```

Provide the OpenAI API key and set the Weaviate bearing token by editing lines 33, 38 and 49 of the docker-composer.yml file:

```
30    environment:
31      OPENAI_API_KEY: 'sk-Q*********vr'
32      DATASTORE: 'weaviate'
33      WEAVIATE_URL: 'http://weaviate:8080'
34      WEAVIATE_CLASS: 'STARWARS'
35      CHUNK_SIZE: '400'
36      BEARER_TOKEN: 'ey*********v0'
…
42    environment:
43      GDB_JAVA_OPTS: >-
44        -Xmx2g -Xms2g
45        -Dhealth.max.query.time.seconds=60
46        -Dgraphdb.append.request.id.headers=true
47        -Dgraphdb.gpt.token=sk-Q*********vr
```

Run all services by starting:

```
docker-compose up
```

To load the data:

1. Open the GraphDB workbench at http://localhost:7200/.
2. Using the navigation bar on the left, click on Setup.
3. Under Setup, locate the Repositories button.
4. Open the repository creation page using "Create New Repository".
5. In the configuration form, set the repository name to "otkg".
6. Create a repository with the default settings.
7. Activate the repository from the dropdown on the top right.

Copyright: Ontotext New York | Sofia | Basel
Reach out to us: info@ontotext.com

8. Download the export of the OTKG's data (otkg.ttl) and note where it is stored.
9. Go to the Import menu in the navigation bar on the left.
10. Click on "Upload RDF files" and upload the otkg.ttl file that you downloaded above.
11. The data will now be listed on the Import screen. Click on the red "Import" button to the right of the otkg.ttl filename.
12. Keep all the default Import settings settings and click the Import Settings dialog boxes Import button.

To create a connector:

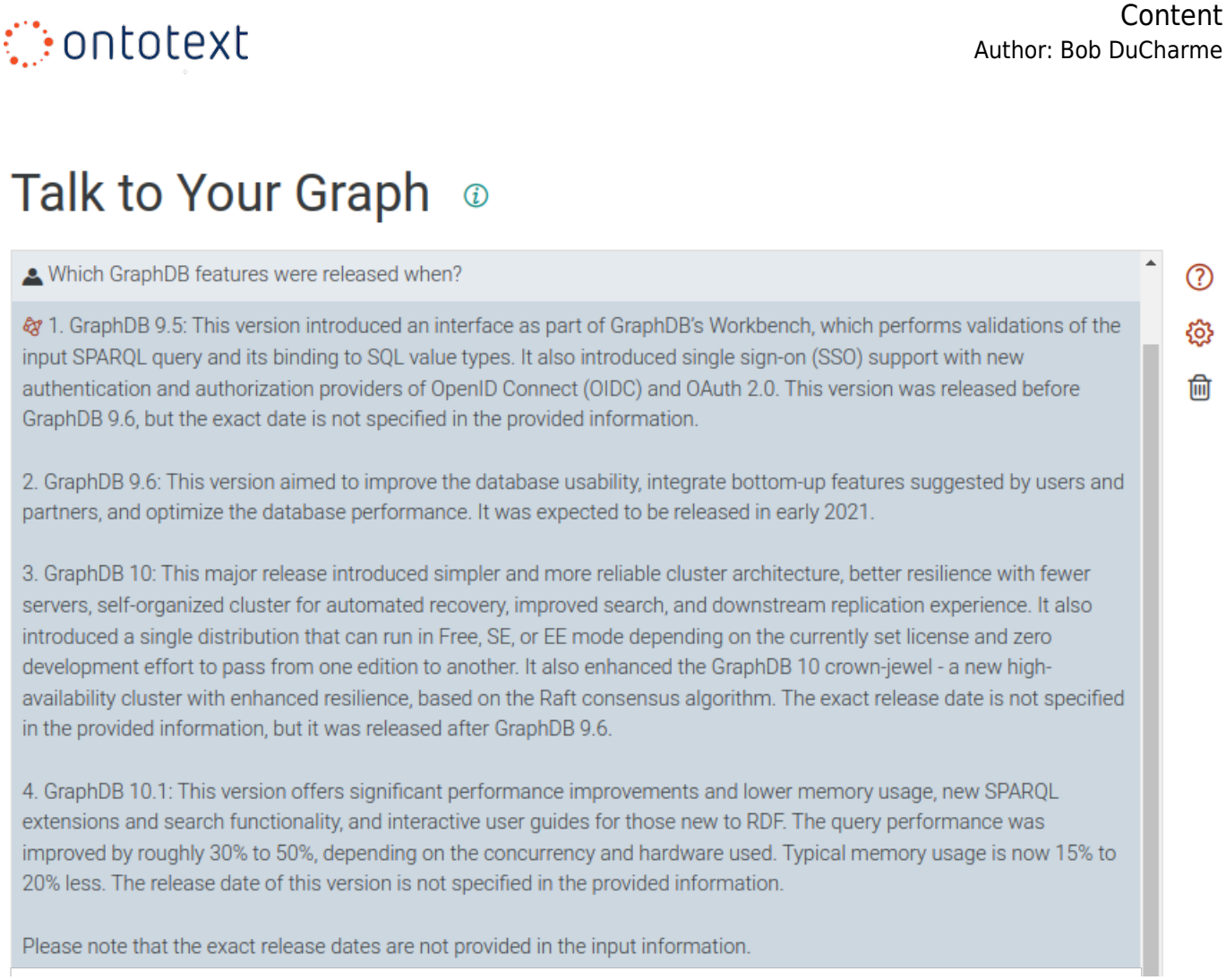
1. Download the file create_connector.rq. This contains the configuration for the weaviate connector.
2. Open the downloaded create_connector.rq file and replace the bearer token with the token you added to the docker-composer.yaml file earlier.
3. Paste the contents of this file's update request into the GraphDB Workbench SPARQL editor and run the request. This could take a minute or two; a progress bar will show how far along it is. Eventually, you should see the message "Created connector creativeWorks".

You can also open your Docker dashboard and check that the retrieval plugin logged that some entities are being persisted into Weaviate. If all goes well, you can go to the Talk To Your Graph page on the workbench and start asking questions.

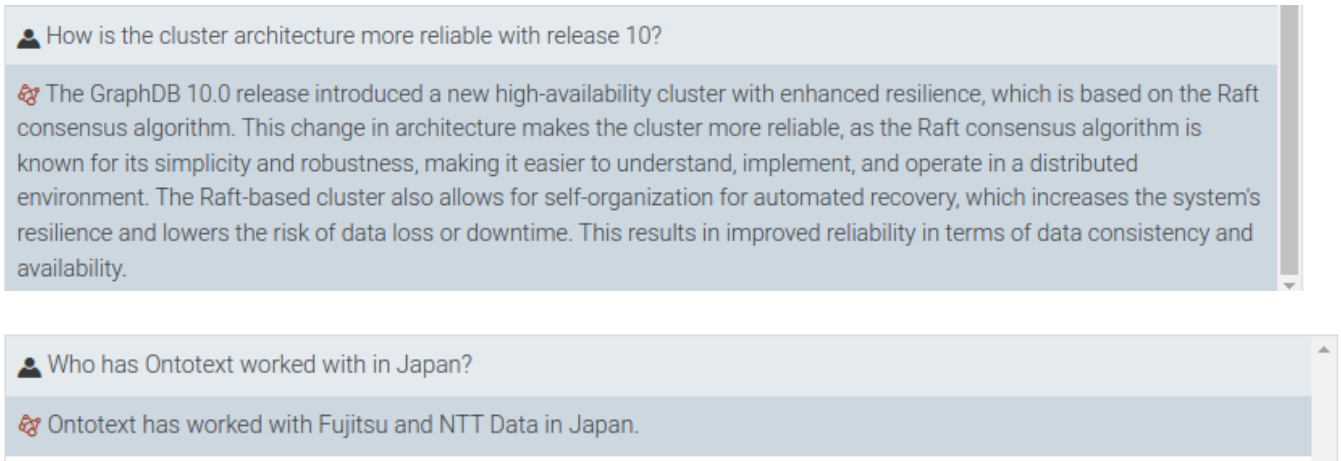
Asking the knowledge graph natural language questions

Ontotext's recent Talk To Your Graph feature, which you will find under the Lab choice of GraphDB's main menu, is a chatbot that lets you ask natural language questions about data in your knowledge graphs. The following shows some requests and responses that were made possible by the configuration described above:

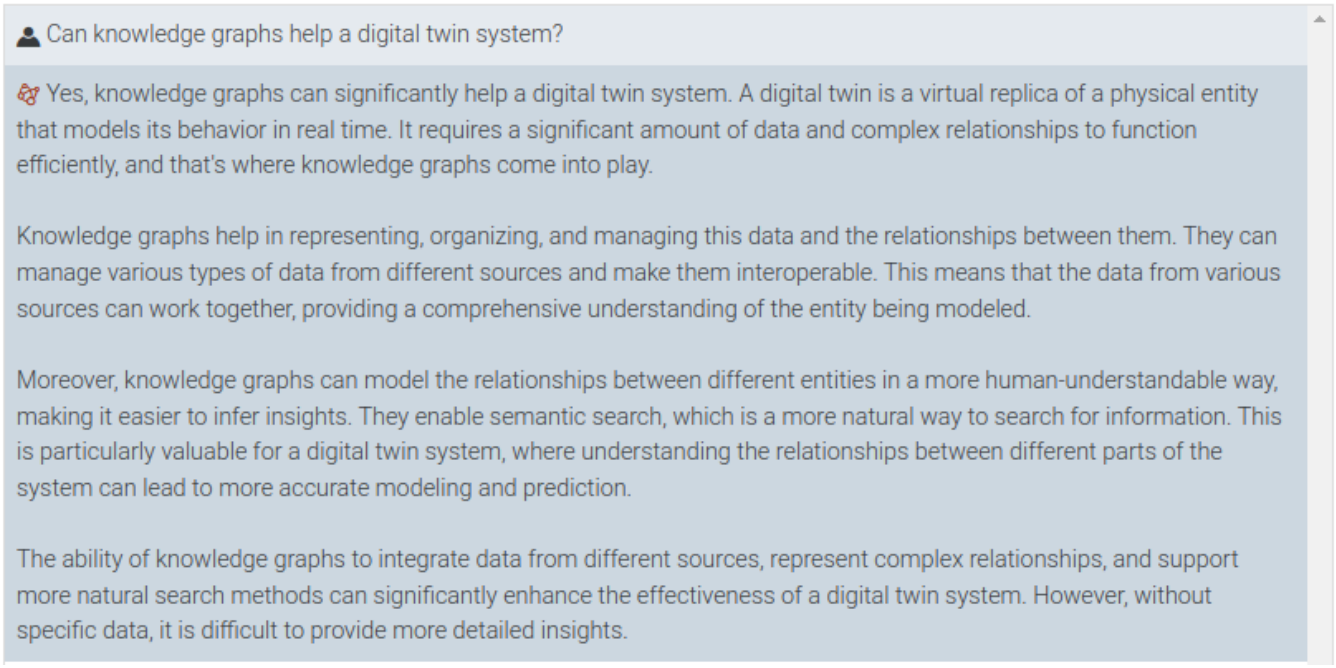
Copyright: Ontotext New York | Sofia | Basel
Reach out to us: info@ontotext.com

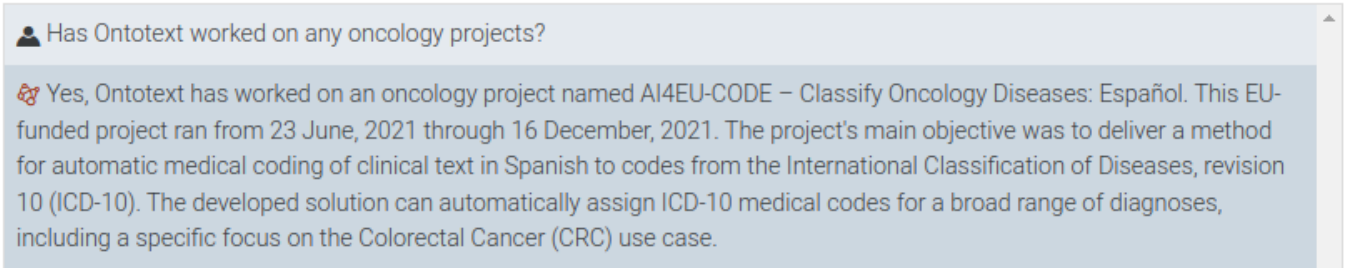


A follow-up to point 3 above:



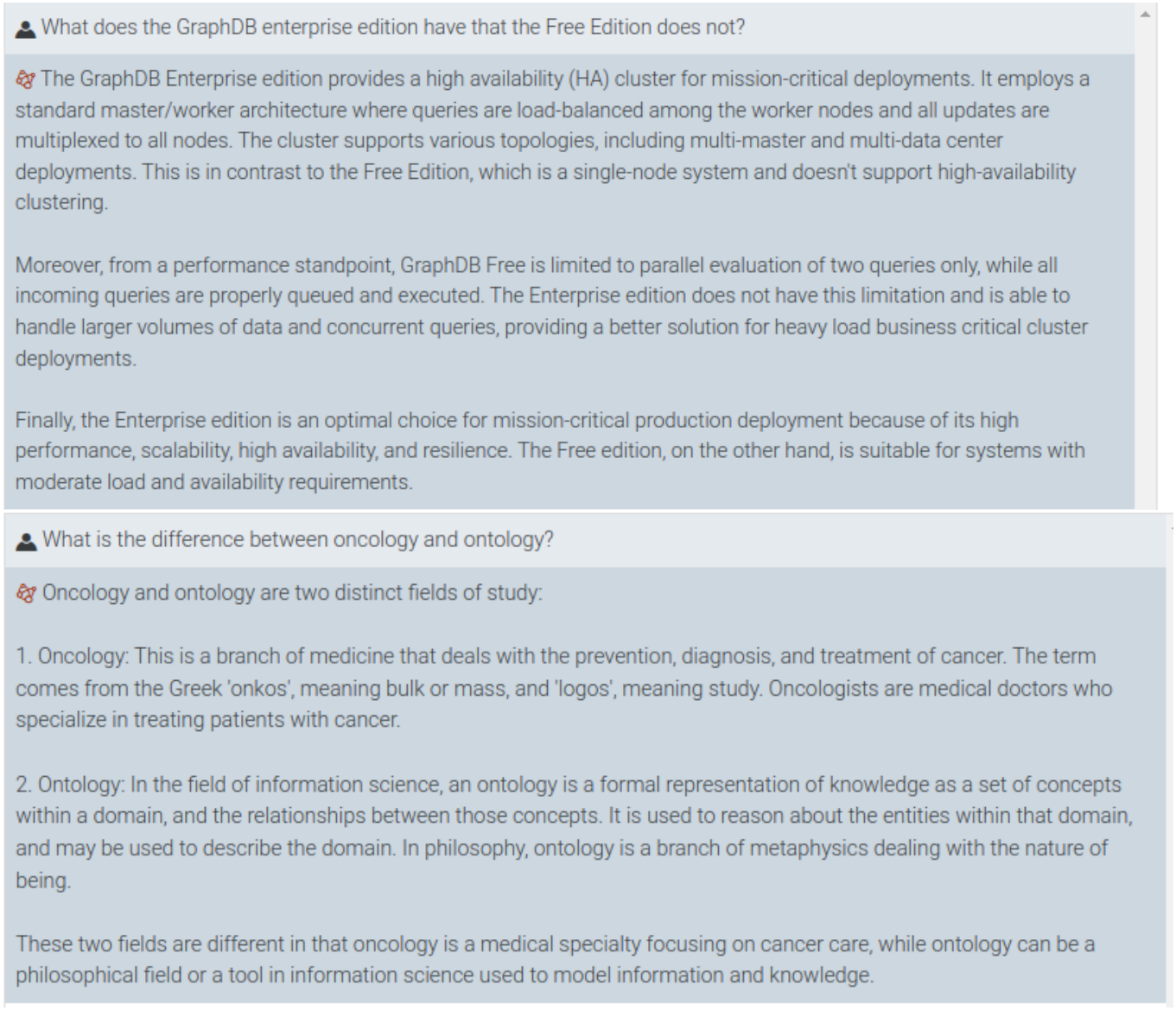
Copyright: Ontotext New York | Sofia | Basel
Reach out to us: info@ontotext.com





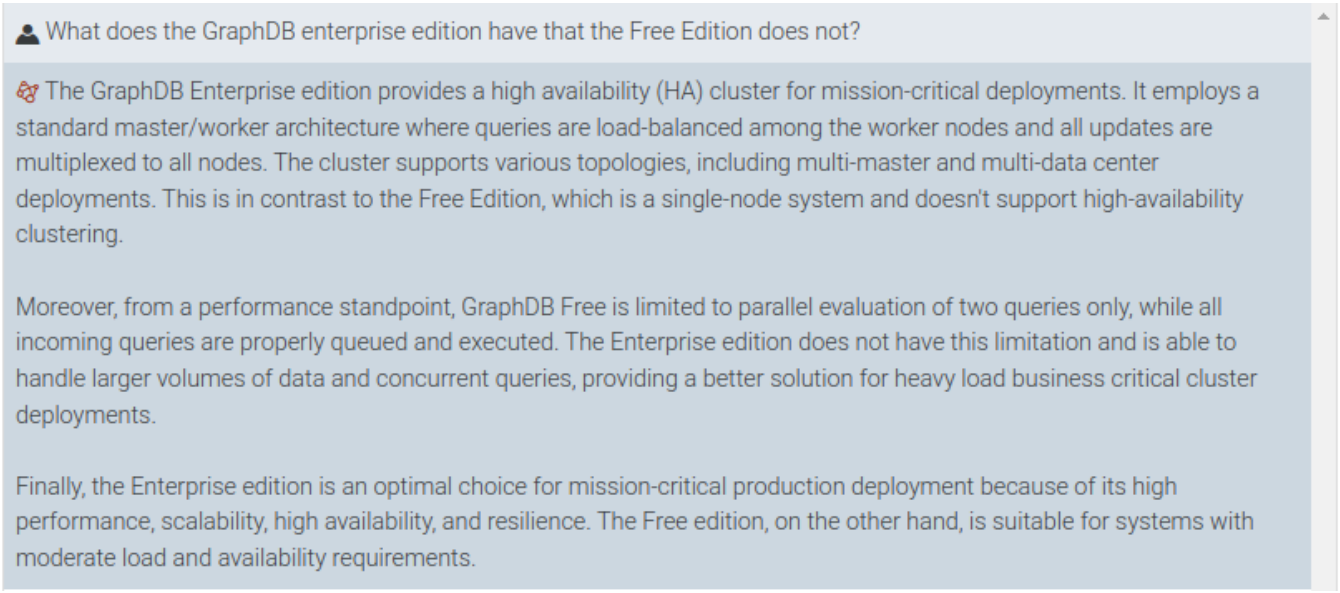
Note that these queries are not just querying the local graph that we set up above; the LLM is quite large and includes extensive additional helpful information. For example:

Copyright: Ontotext New York | Sofia | Basel
Reach out to us: info@ontotext.com



Let's look at one more request to the OTKG knowledge graph and the response:

Copyright: Ontotext New York | Sofia | Basel
Reach out to us: info@ontotext.com



Go and download the latest version of GraphDB, complete with Talk to Your Graph.

Give It a Try

Copyright: Ontotext New York | Sofia | Basel
Reach out to us: info@ontotext.com