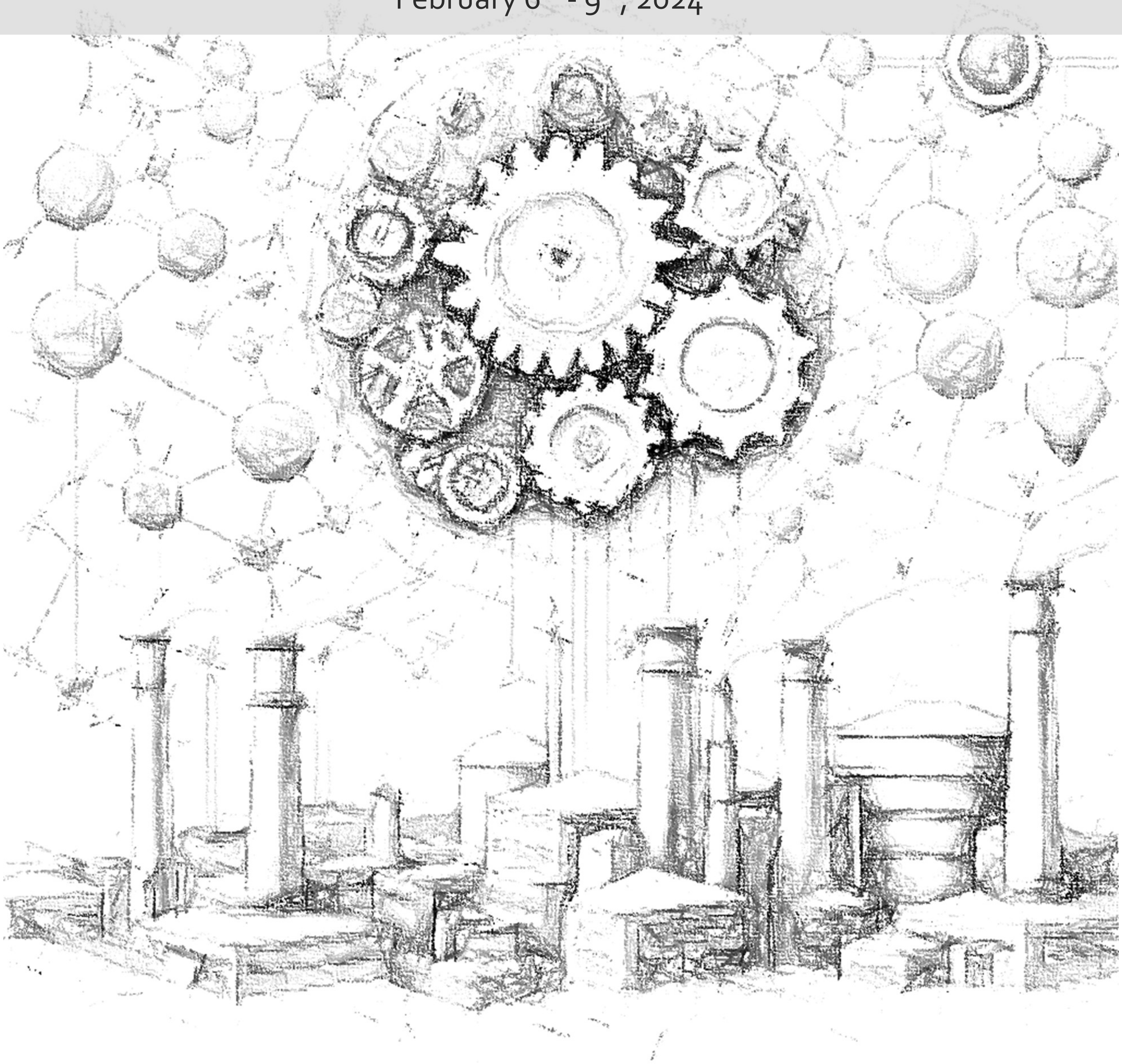
IOF DIGITAL THREAD TUTORIAL

2024 Open Industrial Digital Ecosystem Summit

in Tempe, Arizona

February 6th - 9th, 2024



Synopsis:

This tutorial will walk the participants through how to use IOF ontology constructs to capture information about digital artifacts and physical artifacts (information about physical product, process, and their relationships). Participants will create individuals and properties based on provided example data about requirement, design, plan, simulation/digital twin model, as-manufactured product, as-used product, as-maintained product, etc. Example reasoning and queries may also be exercised.

Dušan Šormaz and Saruda Seeharit
sormaz@ohio.edu or ss748014@ohio.edu

This page was intentionally left blank.

Table of Contents

This page was intentionally left blank.

Part 1: Using Protégé

1.1 *Introduction to Protégé*

Protege is a powerful open-source ontology editor and knowledge-based framework. In this tutorial, we will walk you through the process of creating and managing ontologies using Protege.

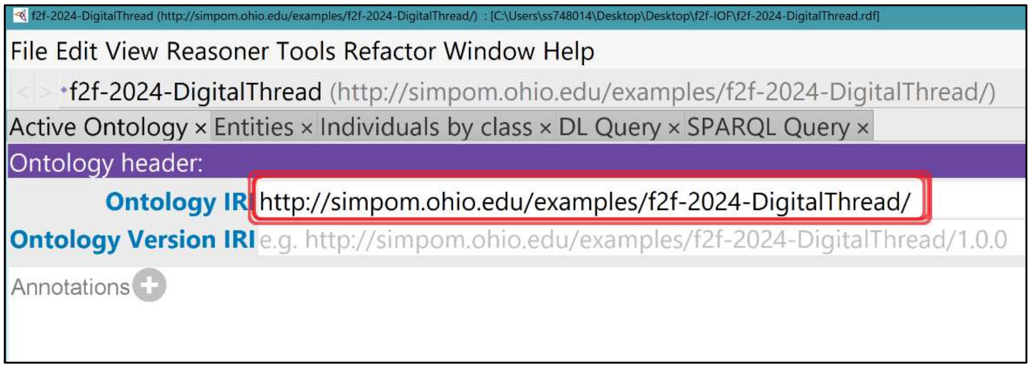
1.1.1 Download and Install Protégé

You can download it easily from their official website (https://protege.stanford.edu/download/). Please do ensure that your computer is installed with Java because Protégé works based on Java application.

1.1.2 Launch Protégé and Setup

1. After you have successfully installed Protégé on your device, launch the application.
2. The first step is to set up an IRI for your ontology. In the Ontology Information section, there is an **'Ontology IRI'** field where you can enter your own IRI. For example, set the Ontology IRI to: http://simpom.ohio.edu/examples/f2f-2024-DigitalThread/, as shown below.

FYI: Note: We will focus on basic tasks such as importing ontologies, creating classes and instances, and adding relationships for this ontology only. So, feel free to keep it as straightforward as possible.


f2f-2024-DigitalThread (http://simpom.ohio.edu/examples/f2f-2024-DigitalThread/) : [C:\Users\ss748014\Desktop\Desktop\f2f-IOF\f2f-2024-DigitalThread.rdf]

f2f-2024-DigitalThread (http://simpom.ohio.edu/examples/f2f-2024-DigitalThread/)

Ontology header:

Ontology IRI http://simpom.ohio.edu/examples/f2f-2024-DigitalThread/

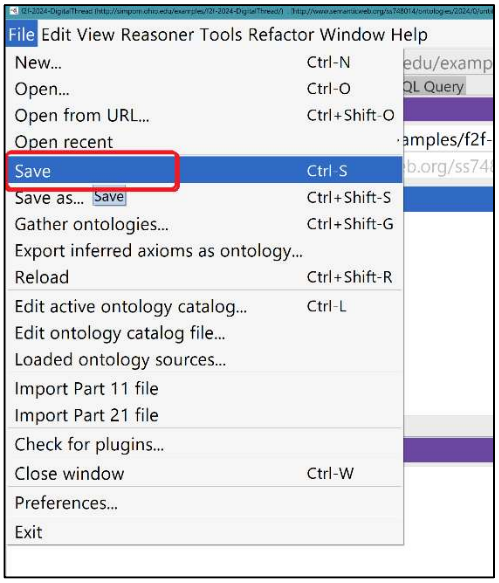
Ontology Version IRI e.g. http://simpom.ohio.edu/examples/f2f-2024-DigitalThread/1.0.0

File Edit View Reasoner Tools Refactor Window Help
Active Ontology × Entities × Individuals by class × DL Query × SPARQL Query ×
Annotations

Figure 1: Setting Up an IRI for Your Ontology

NOTE: IRI of an ontology in Protege is very important in ensuring uniqueness, accessibility, interoperability, documentation, and hence effective use of the ontology. Choose meaningful and well-structured IRIs for your ontology and its components that serve a wider goal of ontology engineering and the Semantic Web while working with Protege or any other development tool for ontologies.

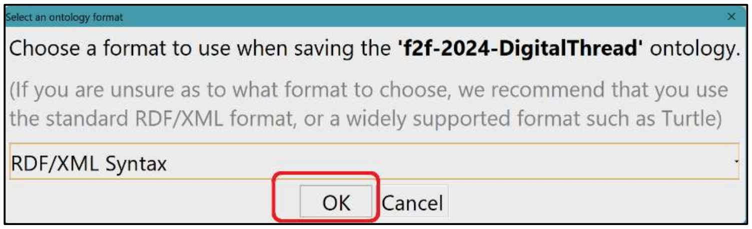
3. Once you are satisfied with the Ontology IRI name, select **'File'**, then choose **'Save'** (as shown **in** Figure 2: Saving Ontology File), or simply press Ctrl +S (or Cmd + S on Mac).


f2f-2024-DigitalThread (http://simpom.ohio.edu/examples/f2f-2024-DigitalThread/) : [http://www.semanticweb.org/ss748014/ontologies/2024/0/unti

File Edit View Reasoner Tools Refactor Window Help
New... Ctrl-N
Open... Ctrl-O
Open from URL... Ctrl+Shift-O
Open recent
Save Ctrl-S
Save as... Ctrl+Shift-S
Save
Gather ontologies... Ctrl+Shift-G
Export inferred axioms as ontology...
Reload Ctrl+Shift-R
Edit active ontology catalog... Ctrl-L
Edit ontology catalog file...
Loaded ontology sources...
Import Part 11 file
Import Part 21 file
Check for plugins...
Close window Ctrl-W
Preferences...
Exit
edu/examp
QL Query
amples/f2f-
b.org/ss748

Figure 2: Saving Ontology File

4. A dialog box will appear,
   - First, select the desired file format for saving your ontology.
     - For example, we have selected ***RDF/XML Syntax*** for this tutorial.
   - Then, click **'OK'** to proceed (as shown in Figure 3).



Choose a format to use when saving the **'f2f-2024-DigitalThread'** ontology.

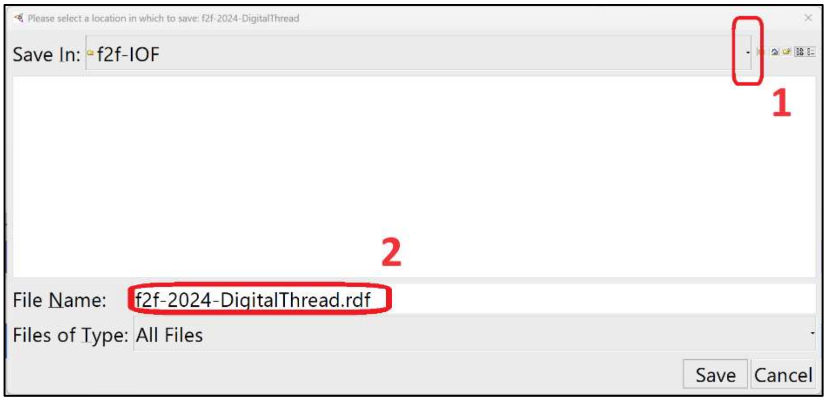
(If you are unsure as to what format to choose, we recommend that you use the standard RDF/XML format, or a widely supported format such as Turtle)

RDF/XML Syntax

Figure 3: Selecting an Ontology Format

5. Choose a location where you wish to save your file (Figure 4: '1')
6. Enter your prefer *File Name*, for example, **'f2f-2024-DigitalThread.rdf'** (Figure 4: '2'). This helps in remembering the name since it aligns with the Ontology IRI **'http://simpom.ohio.edu/examples/f2f-2024-DigitalThread/'**.

> **Tip:** Ensure you append **'*.rdf*'** at the end of your chosen file name. The *'File of Type'* field can be ignored.


Please select a location in which to save: f2f-2024-DigitalThread
1
2
Save
Cancel

Save In: f2f-IOF

File Name: f2f-2024-DigitalThread.rdf

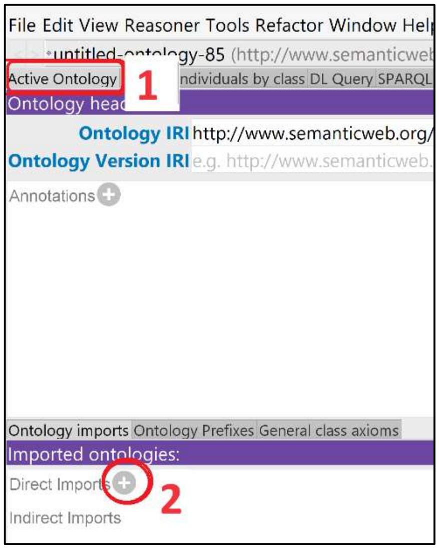
Files of Type: All Files

Figure 4: Choosing a Location to Save Your File.

1.2 *Import Ontology Files*

Importing existing ontologies into Protégé is straightforward! In this section, we will import **"Core.rdf"** (our IOF core ontology file), which will automatically include **"bfo-2020.owl"** (Basic Formal Ontology) along with **"AnnotationVocabulary".**

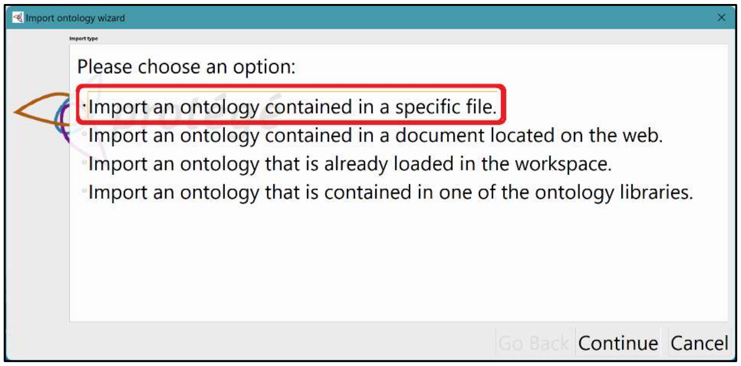
1. Ensure you are in the **"Active Ontology"** tab (Figure 5: Importing Ontology File marked as '1'). If not, simply click on the tab.
2. Click on the **plus button** ⊕ (located next to the phrase **"direct import"** and marked as '2' in Figure 5: Importing Ontology File). This action will launch the Import Ontology wizard as shown in Figure 6.


File Edit View Reasoner Tools Refactor Window Help
untitled-ontology-85 (http://www.semanticweb
Active Ontology
1
ndividuals by class DL Query SPARQL
Ontology head
Annotations
Ontology imports Ontology Prefixes General class axioms
Imported ontologies:
Direct Imports
2
Indirect Imports

Ontology IRI http://www.semanticweb.org/

Ontology Version IRI e.g. http://www.semanticweb.

Figure 5: Importing Ontology File.

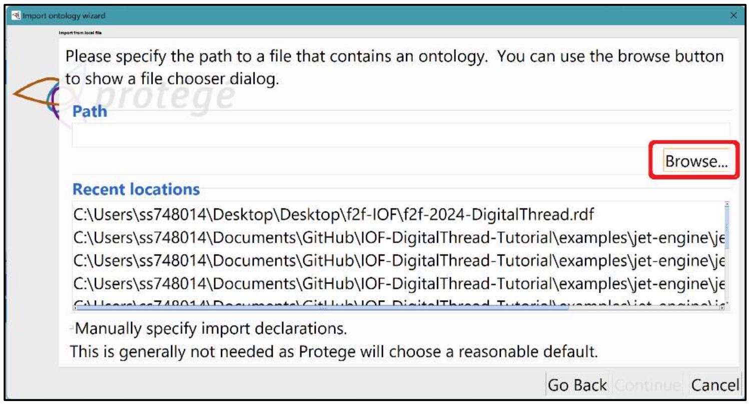


Please choose an option:

- Import an ontology contained in a specific file.
- Import an ontology contained in a document located on the web.
- Import an ontology that is already loaded in the workspace.
- Import an ontology that is contained in one of the ontology libraries.

Figure 6: Import Ontology Wizard.

3. In this tutorial, we will choose the first option: **"Import an ontology contained in a specific file"** (see Figure 6: Import Ontology Wizard.
4. Next, click **"Continue"** (located at the bottom right corner of the popup window), and then select **"Browse..."** as illustrated in Figure 7:



Please specify the path to a file that contains an ontology. You can use the browse button to show a file chooser dialog.

C:\Users\ss748014\Desktop\Desktop\f2f-IOF\f2f-2024-DigitalThread.rdf
C:\Users\ss748014\Documents\GitHub\IOF-DigitalThread-Tutorial\examples\jet-engine\je
C:\Users\ss748014\Documents\GitHub\IOF-DigitalThread-Tutorial\examples\jet-engine\je
C:\Users\ss748014\Documents\GitHub\IOF-DigitalThread-Tutorial\examples\jet-engine\je

Manually specify import declarations.
This is generally not needed as Protege will choose a reasonable default.

Figure 7: Browsing File to Import.

5. Locate the **"Core.rdf"** file you previously downloaded and extracted on your computer, which can be found in the Tutorial folder, and proceed to import it.
6. For instance, if you cloned the Digital Thread tutorial repository onto your local computer, the file path could resemble: GitHub**\IOF-DigitalThread-Tutorial\import\Core.rdf** (refer to Figure 8).
   a. If you downloaded it as a zip file into your download folder, it should be located at: Downloads\IOF-DigitalThread-Tutorial\IOF-DigitalThread-Tutorial\import\Core.rdf.
7. To import the file, you can double-click on its name. Alternatively, you can select the file and then choose "Open".

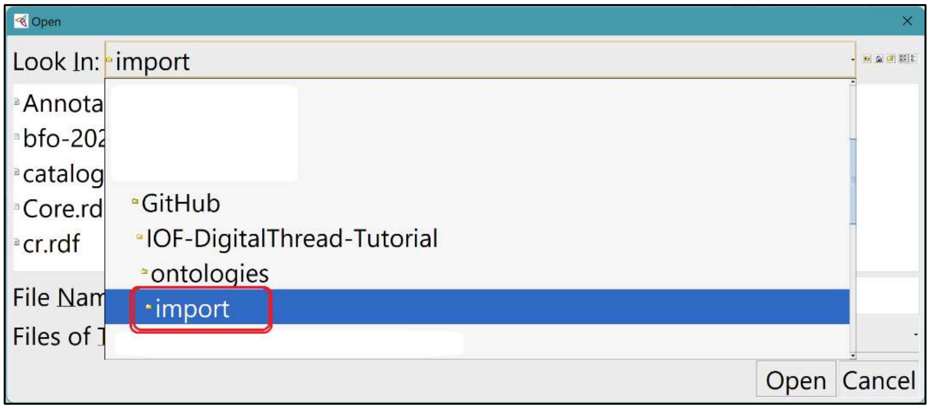
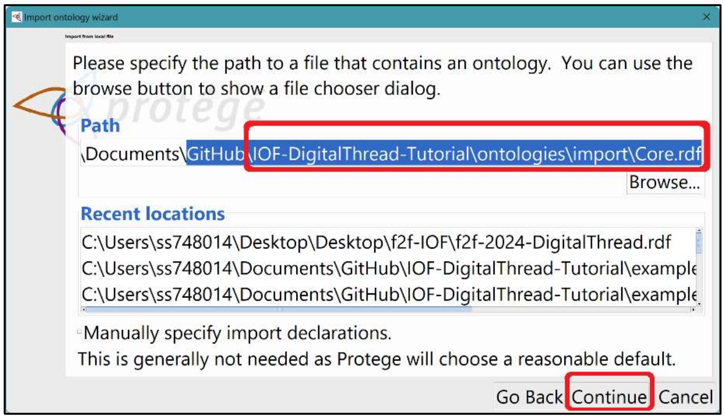


Figure 8: Locating Ontology File for Importing.

8. Verify that the part file is correct, then click "Continue" (as shown in Figure 9:).
9. Then click "Finish" (Figure 10:)!



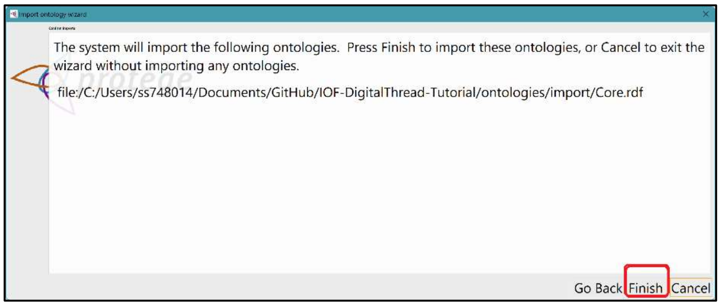
Please specify the path to a file that contains an ontology. You can use the browse button to show a file chooser dialog.

\Documents\GitHub\IOF-DigitalThread-Tutorial\ontologies\import\Core.rdf

C:\Users\ss748014\Desktop\Desktop\f2f-IOF\f2f-2024-DigitalThread.rdf
C:\Users\ss748014\Documents\GitHub\IOF-DigitalThread-Tutorial\example
C:\Users\ss748014\Documents\GitHub\IOF-DigitalThread-Tutorial\example

Manually specify import declarations.
This is generally not needed as Protege will choose a reasonable default.

Figure 9: Import Ontology Wizard (2).

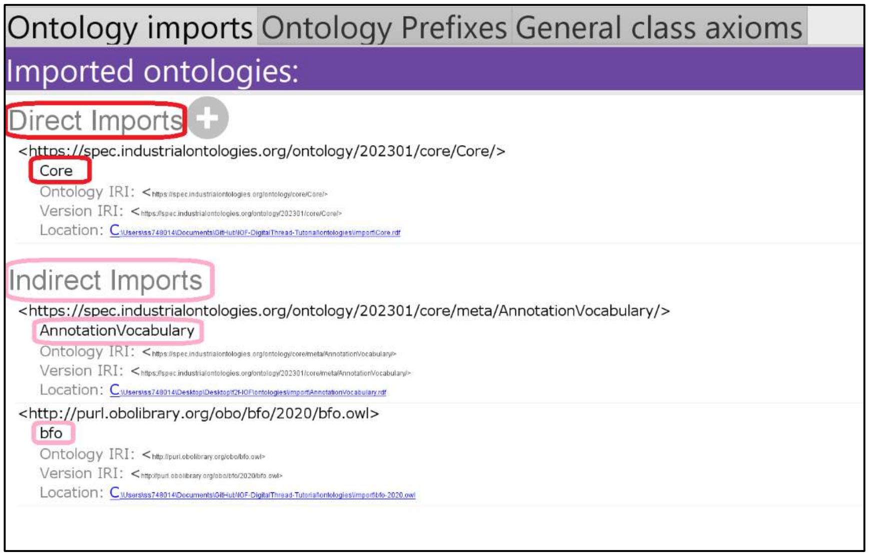


The system will import the following ontologies. Press Finish to import these ontologies, or Cancel to exit the wizard without importing any ontologies.

file:/C:/Users/ss748014/Documents/GitHub/IOF-DigitalThread-Tutorial/ontologies/import/Core.rdf

Figure 10: Import Ontology Wizard (3).

10. If the import process was successful, your Protégé application should now display (as shown in Figure 11):
    a. Direct Import:
        i. Core (https://spec.industrialontologies.org/ontology/202301/core/Core/)
    b. b. Indirect Imports:
        i. AnnotationVocabulary (https://spec.industrialontologies.org/ontology/202301/core/meta/AnnotationVocabulary/)
        ii. bfo (http://purl.obolibrary.org/obo/bfo/2020/bfo.owl) *{using a local file due to some issue}*



<https://spec.industrialontologies.org/ontology/202301/core/Core/>
Core
Ontology IRI: <https://spec.industrialontologies.org/ontology/core/Core/>
Version IRI: <https://spec.industrialontologies.org/ontology/202301/core/Core/>
Location: C:\Users\ss748014\Documents\GitHub\IOF-DigitalThread-Tutorial\ontologies\import\Core.rdf

<https://spec.industrialontologies.org/ontology/202301/core/meta/AnnotationVocabulary/>
AnnotationVocabulary
Ontology IRI: <https://spec.industrialontologies.org/ontology/core/meta/AnnotationVocabulary/>
Version IRI: <https://spec.industrialontologies.org/ontology/202301/core/meta/AnnotationVocabulary/>
Location: C:\Users\ss748014\Desktop\Desktop\f2f-IOF\ontologies\import\AnnotationVocabulary.rdf

<http://purl.obolibrary.org/obo/bfo/2020/bfo.owl>
bfo
Ontology IRI: <http://purl.obolibrary.org/obo/bfo.owl>
Version IRI: <http://purl.obolibrary.org/obo/bfo/2020/bfo.owl>
Location: C:\Users\ss748014\Documents\GitHub\IOF-DigitalThread-Tutorial\ontologies\import\bfo-2020.owl

Figure 11: Imported Ontology View (Direct and Indirect Imports).

1.3 *Create Classes*

To align with our case studies, we will **create two Classes: *Jet Engine* *and* *Compressor*.**

1. Go to the **"Entities"** tab (Figure 12 marked as '1'). Protégé will automatically take you to the **"Classes"** tab. If not, click on the Classes sub tab of the Entities.
2. Click on **owl:Thing** (Figure 12 marked as '2')
3. Click the **"View"** tab at the top (Figure 12 marked as '3')
4. Select the **"Expand all"** option (see Figure 12) to view all the classes imported into your ontology.

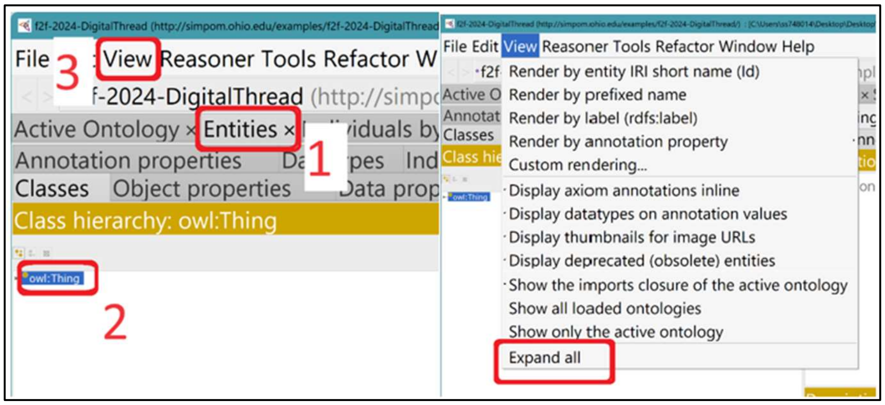
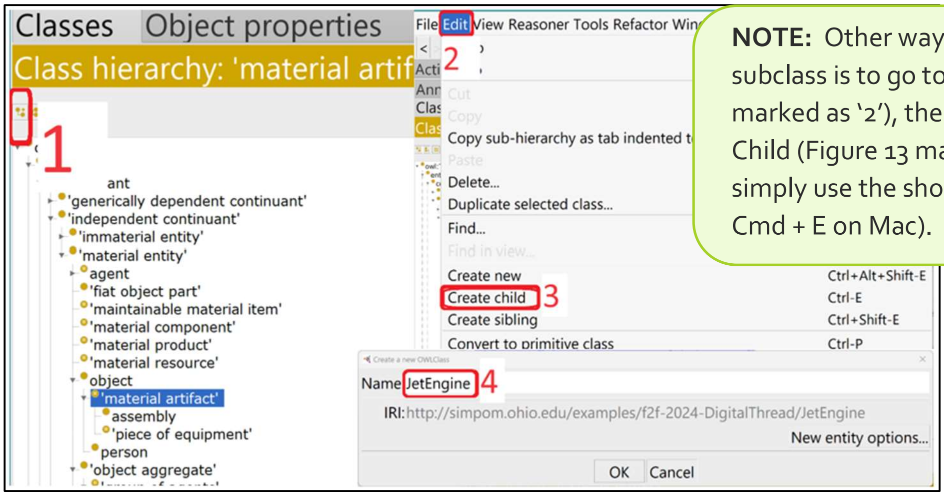

f2f-2024-DigitalThread (http://simpom.ohio.edu/examples/f2f-2024-DigitalThread
File 3 View Reasoner Tools Refactor W
f-2024-DigitalThread (http://simpo
Active Ontology × Entities × Individuals by
Annotation properties Datatypes Ind
Classes Object properties Data prop
1
Class hierarchy: owl:Thing
owl:Thing
2
File Edit View Reasoner Tools Refactor Window Help
Render by entity IRI short name (Id)
Render by prefixed name
Render by label (rdfs:label)
Render by annotation property
Custom rendering...
Display axiom annotations inline
Display datatypes on annotation values
Display thumbnails for image URLs
Display deprecated (obsolete) entities
Show the imports closure of the active ontology
Show all loaded ontologies
Show only the active ontology
Expand all

Figure 12: View All Imported Classes.

5. Locate the class named as **"material artifact"** by following this path: *owl:Thing* → *entity* → *continuant* → *independent continuant* → *material entity* → *object* → ***material artifact.***

> **TIP:** To **quickly activate the search function**, simply press Ctrl + F (or Cmd + F on Mac), then enter the keyword of the item you're searching for. All related entries will be displayed! Just double-click on the item you're interested in to locate it. It's that easy!

6. While class **"material artifact"** is selected, to **Add a subclass** choose , as shown in Figure 13 marked as '1'.
7. In the popup dialog box, type in the **Name** field: **JetEngine**.
8. Then click **OK.**
9. Repeat the same process to create a class named Compressor (Name: **Compressor**)
   a. You could also create sibling for the JetEngine class! (next to '1' in Figure 13 or Ctrl + Shift E)
10. Lastly, create a class named "Thrust" as a Child class of **quality.**



Name: JetEngine

IRI: http://simpom.ohio.edu/examples/f2f-2024-DigitalThread/JetEngine

> **NOTE:** Other way to add a subclass is to go to Edit (Figure 13 marked as '2'), then select Create Child (Figure 13 marked as '3'), or simply use the shortcut Ctrl + E (or Cmd + E on Mac).

Figure 13: Adding Class.

1.4 Create Individuals (instances)

1. Navigate to **"Individuals"** tab (Figure 14 marked as '1'), which is part of the "Entities" group tab.
2. Click "**Create individual"** (found under the purple **individual view**, as show in Figure 14 marked as '2').

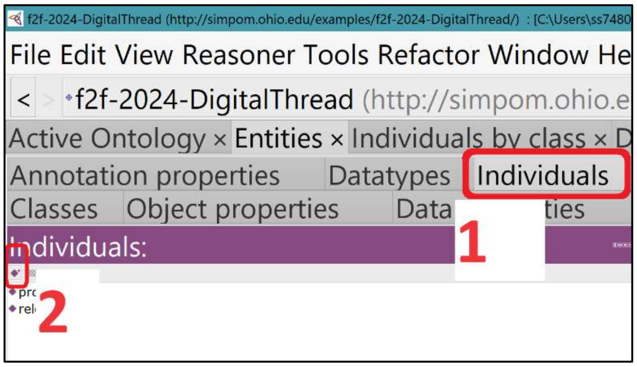
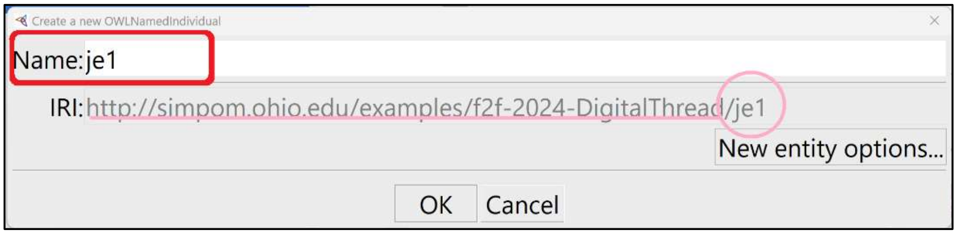

f2f-2024-DigitalThread (http://simpom.ohio.edu/examples/f2f-2024-DigitalThread/) : [C:\Users\ss7480
File Edit View Reasoner Tools Refactor Window He
< > f2f-2024-DigitalThread (http://simpom.ohio.e
Active Ontology × Entities × Individuals by class × D
Annotation properties Datatypes Individuals
Classes Object properties Data ties
Individuals:
1
pro
rel
2

Figure 14: Create an Individual (Instance).

3. A dialog box titled *"Create a new OWLNamedIndividual"* will appear.
4. In the **Name** field, type **je1** (as shown in Figure 15). *Note** je1 is short name for jet engine number 1.*

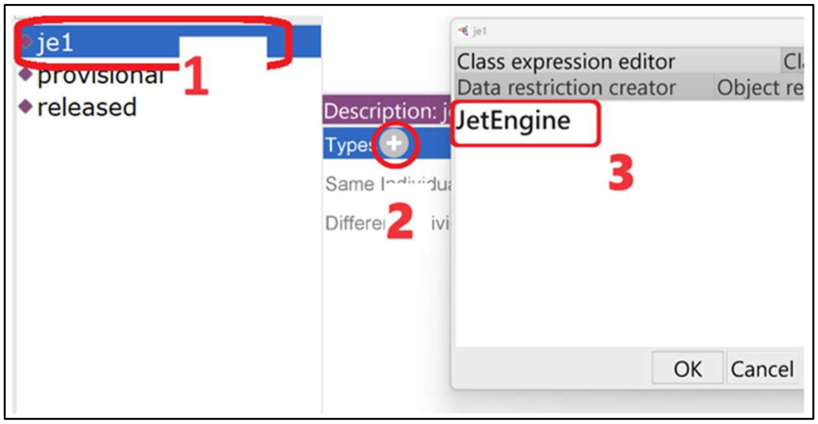


Name: je1

IRI: http://simpom.ohio.edu/examples/f2f-2024-DigitalThread/je1

Figure 15: Create a new OWLNamedIndividual.

5. *Ensure you select the* ***je1*** individual (Figure 16 '1'). Just click on it, and a blue tab will indicate your selection.
6. In the **Description Box**, look for the **plus sign** ⊕ **next to Type**, then click on it (Figure 16 '2'). A box will pop up.
7. Select **Class expression editor** (it should be automatically select).
   - **a.** Begin typing **"Jet"** only, then press Ctrl + Spacebar (or Cmd + Spacebar on Mac) for the program to autocomplete the class name as **JetEngine** (Figure 16 '3').
   - **b.** Press **OK** to complete the class assignment for the individual.
8. **REPEAT** the process to create an individual for the ***Compressor***, Name: **c1**. *Note** Adjust the process as needed.*
9. **REPEAT AGAIN** to create an individual for ***Thrust***, Name: **th1-720**.



JetEngine

Figure 16: Class Assignment for an Individual.

1.5 *Object Property Assertions*

According to an example sentence:

<u>"Jet Engine 1 has a Compressor 1 as its component. The engine has a Thrust of 720 kN."</u>

1. Access the Property assertion window while in the "Individuals" tab, located at the bottom right corner of the interface.

> **Tip:** If it's not visible on the interface, navigate through the top menu by selecting: <u>Window → Views → Individuals Views → Property Assertions.</u> Then, position it on the screen where you prefer.

2. The object properties to be added are ***"has component part at some time"*** and ***"has quality":***
   a. First, select the **je1** individual *(Figure 17 '1')*. A blue tab will indicate your selection.
   b. Click on the **plus sign** ⊕ next to the **Object property assertions** as indicated in *Figure 17 '2'* (found within the *"Property assertions"* view).
   c. In the dialog box that appears, object property name, type **"has component part at some time"** (*Figure 17* '3').
   d. For individual name, type **c1** (*Figure 17* '4'), then click **OK** (*Figure 17* '5') to finalize adding the object property.
3. This achieves the representation: **"Jet Engine 1 has a Compressor 1 as its component."**
4. Follow a similar process for the statement: "The engine has a Thrust of 720 kN." *Hint The object property name is **has quality**. The individual name is **th1-720**.*

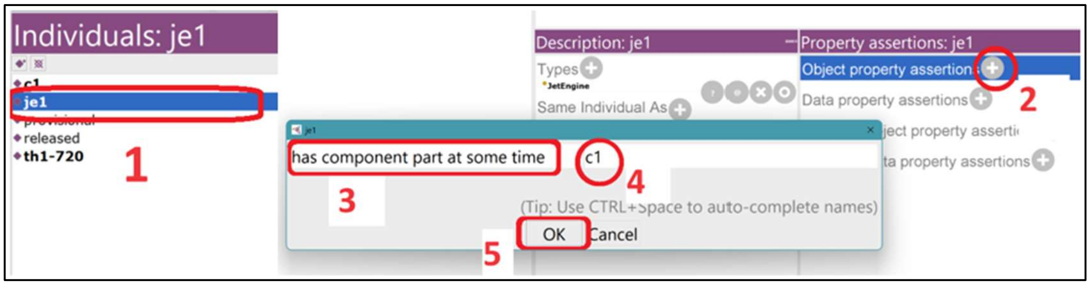


Figure 17: Object Property Assertions.

1.6 *Practice on Protégé*

1.6.1 Using the Practice File

1. While in Protégé, navigate to File (located at the top left corner of the application) and select Open… *Note** If there is a dialog box called **"Open in current window"** appear, select **"No"** for this scenario.*
2. Navigate to the tutorial folder and locate the file named **"jet-engine-CR-forPractice.rdf"**
   a. For this example, it is found in → IOF-DigitalThread-Tutorial-main (this is the folder extracted from the ZIP file downloaded from GitHub) → **IOF-DigitalThread-Tutorial-main** → **examples** → **jet-engine** (Figure 18).
   b. You can either double-click on the file **"jet-engine-CR-forPractice.rdf"** or select the file, then click **"Open".**

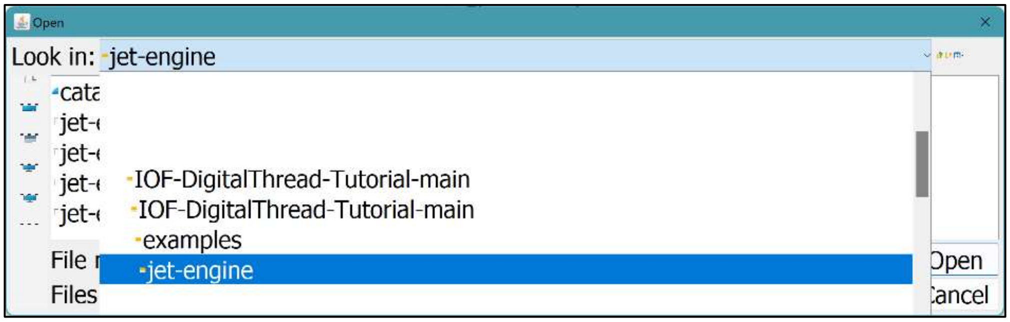


Figure 18: Open a Practice Ontology File on Protégé.

3. Assignments (refer back to sections 1.4 and 1.5 if needed for guidance):
   a. Add an individual named **"c-DS1"** (for compressor's design specification)
   b. Assign the Types in the Description View to **c-DS1** as **DesignSpecification**, which is a class from the IOF Core ontology. *Note** c-DS1 is an acronym for compressor's design specification number 1.*
   c. Add multiple Object property assertions:
      i. *prescribesProducedEntity* **c1**
      ii. *prescribesProducedEntity* **c2**
      iii. *prescribesDesignedEntity* **c-d1**

*Note** the properties CR:prescribesProducedEntity and CR:prescribesDesignedEntity are part of new Object properties currently under proposal.*

1.6.2 Use HermiT Reasoner in Protégé

1. While on the same ontology file **"jet-engine-CR-forPractice.rdf",** open, navigate to the Reasoner tab and select HermiT as your reasoning option.
2. Then, go back to the Reasoner tab, but this time select Start reasoner.

Tip: To start reasoner simply press Ctrl + R (or Cmd + R on Mac).

IMPORTANT: This reasoning process is a fundamental aspect of working with ontologies in Protégé, enabling users to develop robust, logically consistent ontologies and to discover implicit knowledge and relationships within

Part 2: Get to Know GraphDB (OntoText)

2.1 *Introduction to GraphDB*

Ontotext GraphDB is a highly efficient and robust graph database with RDF and SPARQL support (graphdb.ontotext.com).

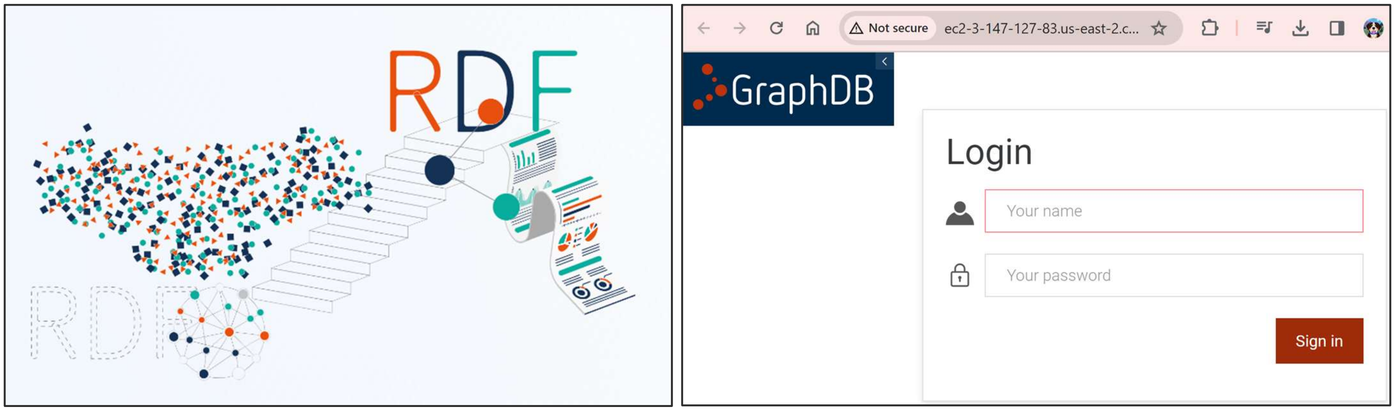


Figure 19: Illustration From graphdb.ontotext.com (left), and Login Page for This Tutorial (right).

2.2 *Log In to the GraphDB*

1. To access the remote location provided by OntoText, connect using the following details:
   a. **URL**: http://ec2-3-147-127-83.us-east-2.com@rPY8Hi-kMN9A70*Fpute.amazonaws.com/
   b. **Shortened URL:** https://bit.ly/iof2402
   c. This GraphDB port availability is thru 2024-07-09.
   d. Username: **<last name>** (all in lower cases)
   e. Password: **<first name>** (all in lower cases)
2. Ensure to verify the URLs and credentials provided for accuracy and security compliance before attempting to connect (Figure 19).

Want to install it on your computer?

If you're interested in installing GraphDB on your computer, here's how you can get started with the free version and access resources for assistance:

1. **Download GraphDB:** Use this link to navigate to the download page for GraphDB, where you can access the free version https://www.ontotext.com/products/graphdb/download/
2. **Installation Guide:** Need help with the installation? Click here for detailed instructions: https://graphdb.ontotext.com/documentation/10.5/how-to-install-graphdb.html
3. **About GraphDB:** To understand what GraphDB is and its capabilities, visit: https://graphdb.ontotext.com/documentation/10.5/index.html
4. **Tutorials from GraphDB:** For hands-on learning, check out the available tutorials: https://graphdb.ontotext.com/documentation/10.5/interactive-guides.html
5. **Documentation:** Access the complete GraphDB Documentation for Version 10.5 here: https://graphdb.ontotext.com/documentation/10.5/pdf/GraphDB.pdf

These resources should help you successfully download, install, and start using GraphDB, as well as provide you with the necessary information and guides to effectively utilize this powerful database for your projects.

2.3 *Setup a Repository and Import Data*

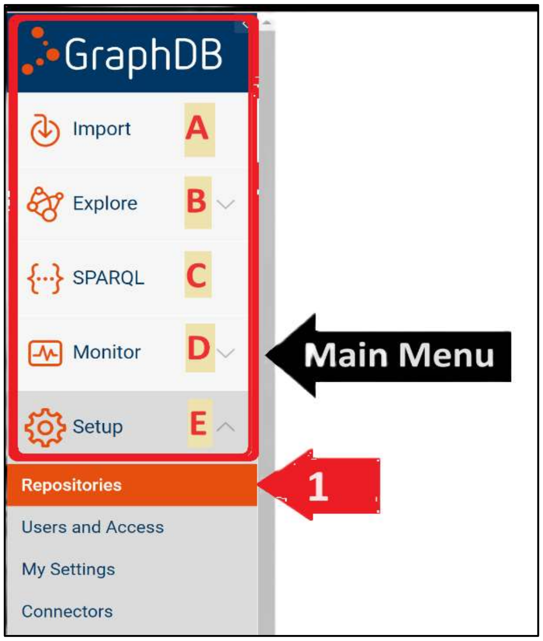


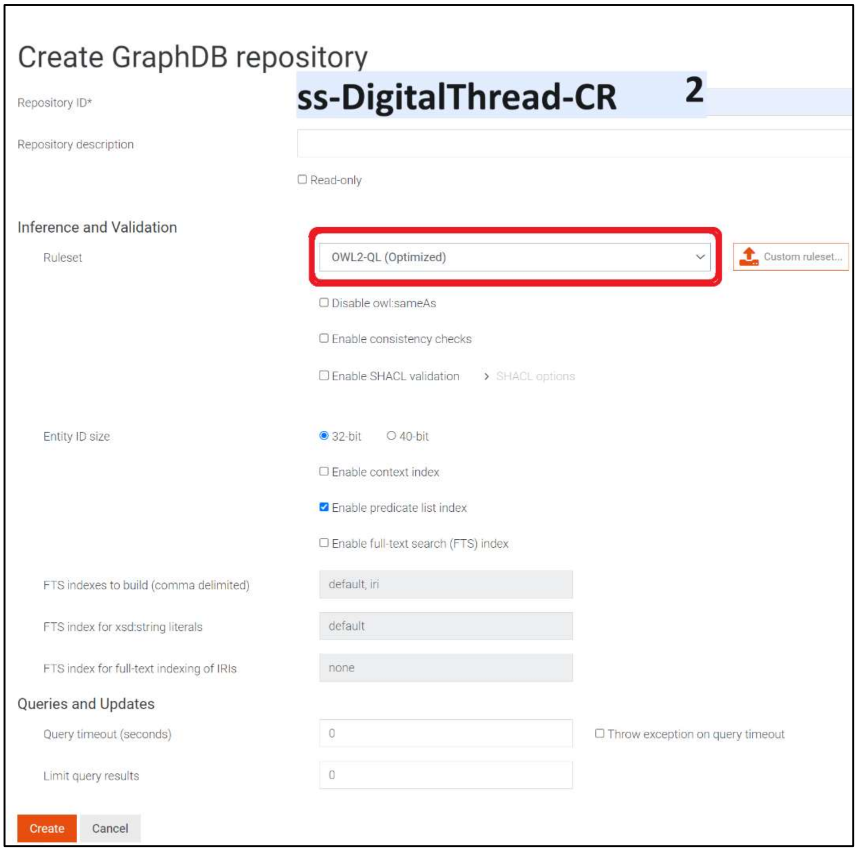
Figure 20: GraphDB Main Menu.

1. Login to GraphDB (session 2.2 procedure 1 Log In to the GraphDB).
2. In the Main Menu as shown in Figure 20, click on **"Setup"** (labeled as 'E'), then choose **"Repositories"**.
3. From the main interface select the option to **"Create new repository"** ⊕ Create new repository
4. Choose **"GraphDB Repository"** as the type of repository for this tutorial.

GraphDB Repository

GraphDB repositories store data, answer queries and execute updates.

5. Enter a name for your repository. It's recommended to use a *combination of your initials or unique letters and/or numbers,* followed by **"-DigitalThread-CR",** to make it easily identifiable.
6. When setting up Inference and Validation under the Ruleset section, select **"OWL2-QL (Optimized)"** which is recommended for the purposes of this tutorial.
7. Then at the bottom of the page click **'Create'** Create. This will finalize the creation of your new repository configured specifically for this tutorial.



Create GraphDB repository

Repository ID* ss-DigitalThread-CR 2

Repository description

□ Read-only

Inference and Validation

Ruleset: OWL2-QL (Optimized) | Custom ruleset...

□ Disable owl:sameAs

□ Enable consistency checks

□ Enable SHACL validation › SHACL options

Entity ID size: ● 32-bit ○ 40-bit

□ Enable context index

☑ Enable predicate list index

□ Enable full-text search (FTS) index

FTS indexes to build (comma delimited): default, iri

FTS index for xsd:string literals: default

FTS index for full-text indexing of IRIs: none

Queries and Updates

Query timeout (seconds): 0 □ Throw exception on query timeout

Limit query results: 0

Create Cancel

Figure 21: Create GraphDB Repository.

8. In the **Repositories view,** you can connect to your repository by clicking on the plug sign 🔌 (Figure 22 'Connected') *Alternatively, you can select your repository from the drop-down menu located at the top right corner of the interface (Figure 22 'Drop-down to select a repository')*

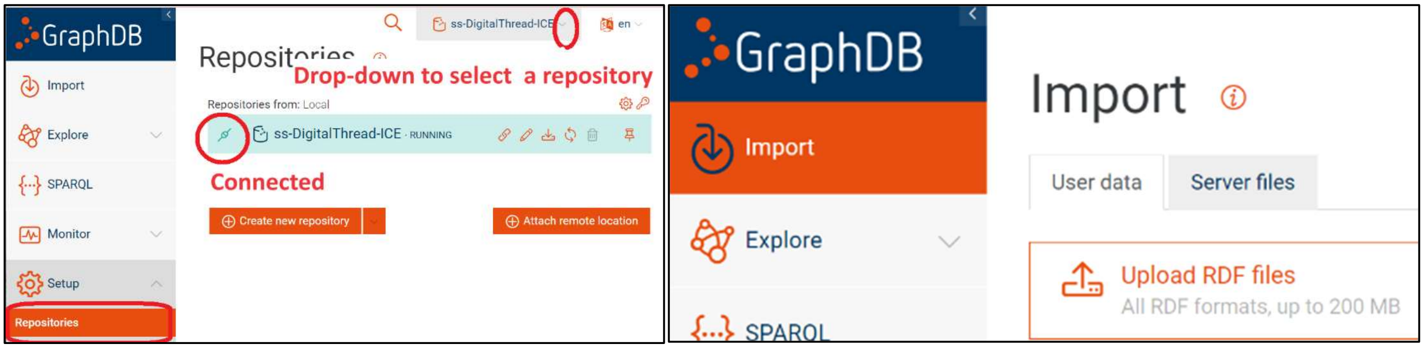


Figure 22: Repositories View (left) and Import View (right).

9. Once connected to the repository, click on **"Import"** Import at the Main Menu.
10. In the **"User data"** tab, choose the **"Upload RDF files"** option.
11. Navigate to the folder where you downloaded and unzipped the tutorial session materials:
    a. Path: *IOF-DigitalThread-Tutorial-main > ontologies > **import***
    b. Import <u>**ALL**</u> the files in the import folder, including:
        i. Core.rdf
        ii. bfo-2020.owl
        iii. AnnotationVocabulary.rdf
        iv. cr.rdf.

> **Tip:** To import multiple files from the same folder, press and hold Ctrl (or Cmd on Mac) while selecting all the files you wish to import.

12. Repeat a similar process to *upload 2 files* located in IOF-DigitalThread-Tutorial-main > examples > **jet-engine:**
    a. Jet-engine-Base.rdf
    b. jet-engine-cr.rdf
13. Verify, then import all the items you've uploaded or click **"Select all"** (Figure 23 ' 1'), then click **"Import"**(Figure 23 ' 2') to import all the files at once.
14. An **"Import settings"** dialog box will appear. For this tutorial, you *can simply select* **"Import"** to complete the process.
15. Your repository is now ready for further analysis.

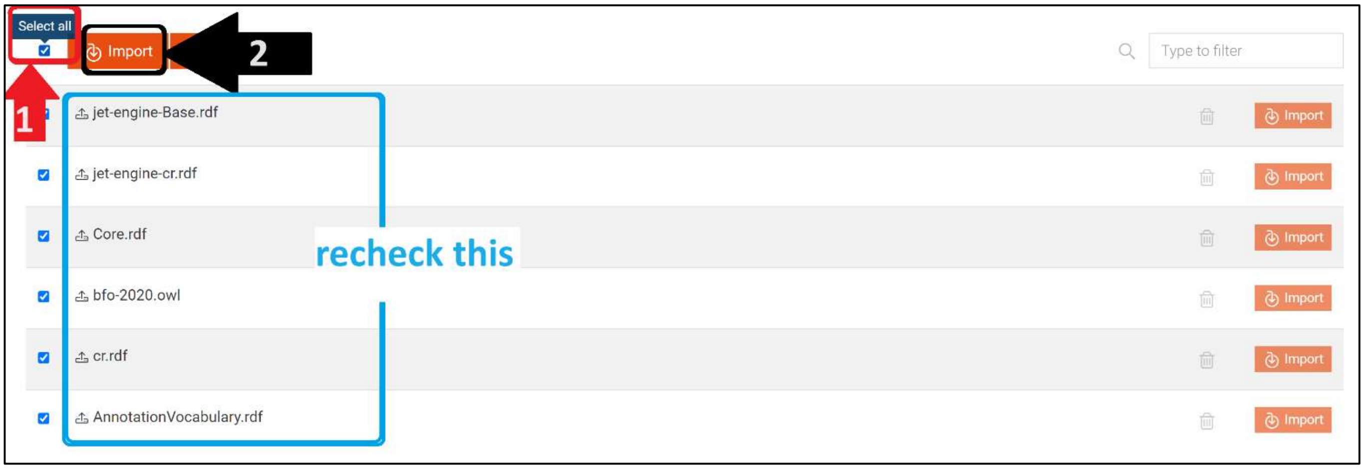
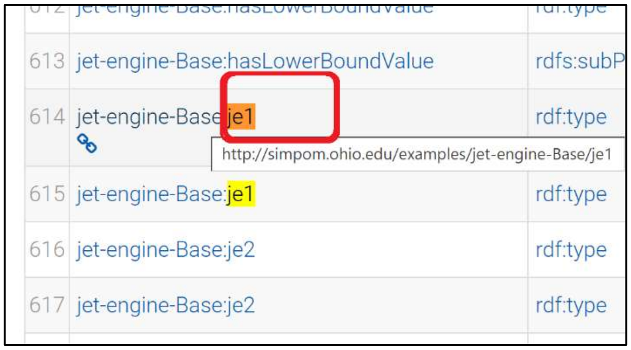


Figure 23: Import Ontology Files.

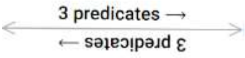
2.4 Explore Knowledge Graphs

1. On the Main Menu, navigate to **"Explore"** (labeled as 'B' in Figure 20) → **"Graphs Overview".**
2. Within the main interface, choose **"The default graph".**
3. Using *Ctrl+ F (or Cmd + F on Mac)* to activate normal search, type in **je1** to locate the specific individual.
4. Click on the term ***jet-engine-Base:je1*** located in <u>subject</u> column.
5. Then select **"Visual graph"** Visual graph , located near the top on the right side of the interface.



612	jet-engine-Base:hasLowerBoundValue	rdf:type
613	jet-engine-Base:hasLowerBoundValue	rdfs:subP
614	jet-engine-Base:je1	rdf:type
	http://simpom.ohio.edu/examples/jet-engine-Base/je1	
615	jet-engine-Base:je1	rdf:type
616	jet-engine-Base:je2	rdf:type
617	jet-engine-Base:je2	rdf:type

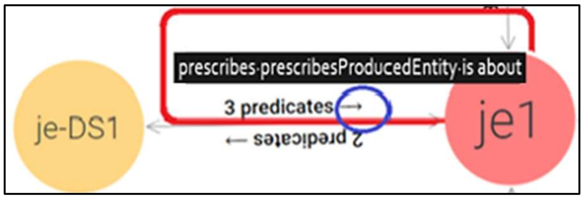
Figure 24: Locate a Specific Class or Individual.

6. Noticed the multitude of nodes that appeared? These are classes and individuals related to je1. This is due to the Ruleset of OWL2-QL (optimized) acting as an inference (reasoner), revealing a complex network of relations and inferences.
7. To focus on the individual entities first, you can adjust the visual settings:
   a. Click on the settings **gear icon** ⚙ (top right) to access the **graph view configuration** options (Figure 25).
   b. In the Graph Settings panel on the right side (see Figure 25:), navigate to the **Types** tab and in the field of Preferred **types**, enter **"owl:NamedIndividual"** and **hit enter**. This will be converted automatically to (http://www.w3.org/2002/07/owl#NamedIndividual).
   c. Enable the **"Show preferred types only"** option and click **"Save"** to apply your changes.
   d. Now, the visual graph should only display the individuals related to the individual je1, excluding any classes.
   e. You'll observe various numbers and predicates with arrows indicating the object properties linking each subject individual to its object individual. 

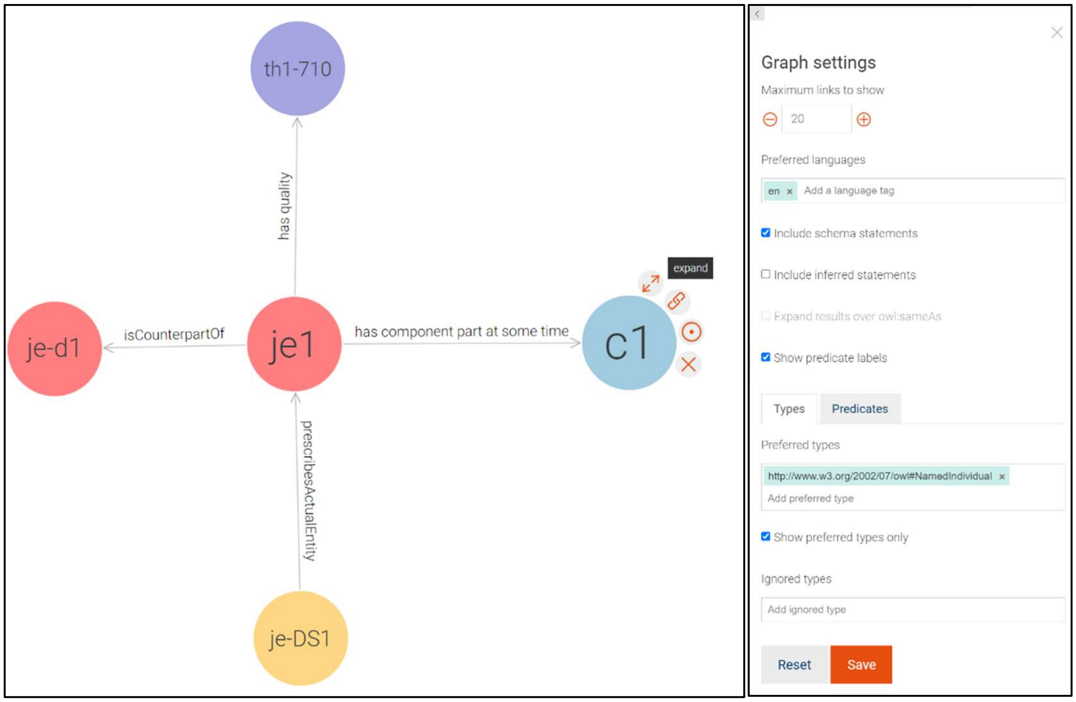
      - Displayed properties include both asserted and inferred statements generated by the OWL2-QL reasoner.

Tip: To understand the predicates represented on each line, hover your mouse over the one you're interested in for more details (as shown in Figure 25).

<u>For example</u>



Subject	Predicate	Object
je-DS1	prescribes	je1
je-DS1	prescribesProducedEntity	je1
je-DS1	is about	je1
je1	prescribed by	je-DS1
je1	is subject of	je-DS1



Graph settings

Maximum links to show

20

Preferred languages

en × Add a language tag

☑ Include schema statements

☐ Include inferred statements

☐ Expand results over owl:sameAs

☑ Show predicate labels

Types | Predicates

Preferred types

http://www.w3.org/2002/07/owl#NamedIndividual ×

Add preferred type

☑ Show preferred types only

Ignored types

Add ignored type

Reset Save

Figure 25: Adjusting Graph Settings.

f. To simplify the graph for better understanding and to minimize distractions from the reasoner, we'll opt out of including the inferred statements.

g. Return to the Graph Settings, then **uncheck** the **"Include inferred statements"** option, which will automatically disable it. Remember to click **"Save"** to *apply your changes*.

h. You will now see only the object properties (predicates) that have been explicitly asserted within the project, making the graph more focused and easier to interpret on the input data.

i. Hovering over a node in the visual graph offers a menu for several actions to enhance your exploration and analysis:

- **Expand/ Collapse:** You can expand a node to reveal its relationships or collapse it to hide them if it's already expanded. Double-clicking on a node also expands it.
- **Copy IRI:** This option allows you to copy the node's IRI (Internationalized Resource Identifier) directly to your clipboard, facilitating reference or further investigation.
- **Focus:** Selecting this will reorient the graph to center around the chosen instance as the main focus. Be mindful that this action resets the current state of your graph to concentrate on the selected node.
- **Remove:** Use this to remove a node's visual representation and its relationships from the graph. This action helps declutter the view by hiding less relevant information. *Note**you can get the individual back by collapsing it subject/object individual and then expand it again.*

NOTE: Clicking on a node reveals a side panel with the node's details, including a description (rdfs:comment), labels (rdfs:label), RDF rank, and an image (foaf:depiction) if present, plus all DataType properties. You can search for specific DataType property values. Click the node again to close the panel and return to the graph.

2.5 SPARQL Query

Answering competency questions is a common approach when working with Semantic Web technologies. Competency questions are typically formulated during the ontology development process to validate the ontology's ability to retrieve the desired information.

For this session we will try to answer the following competency question:

"What is the difference of real engine thrust and designed engine thrust?"

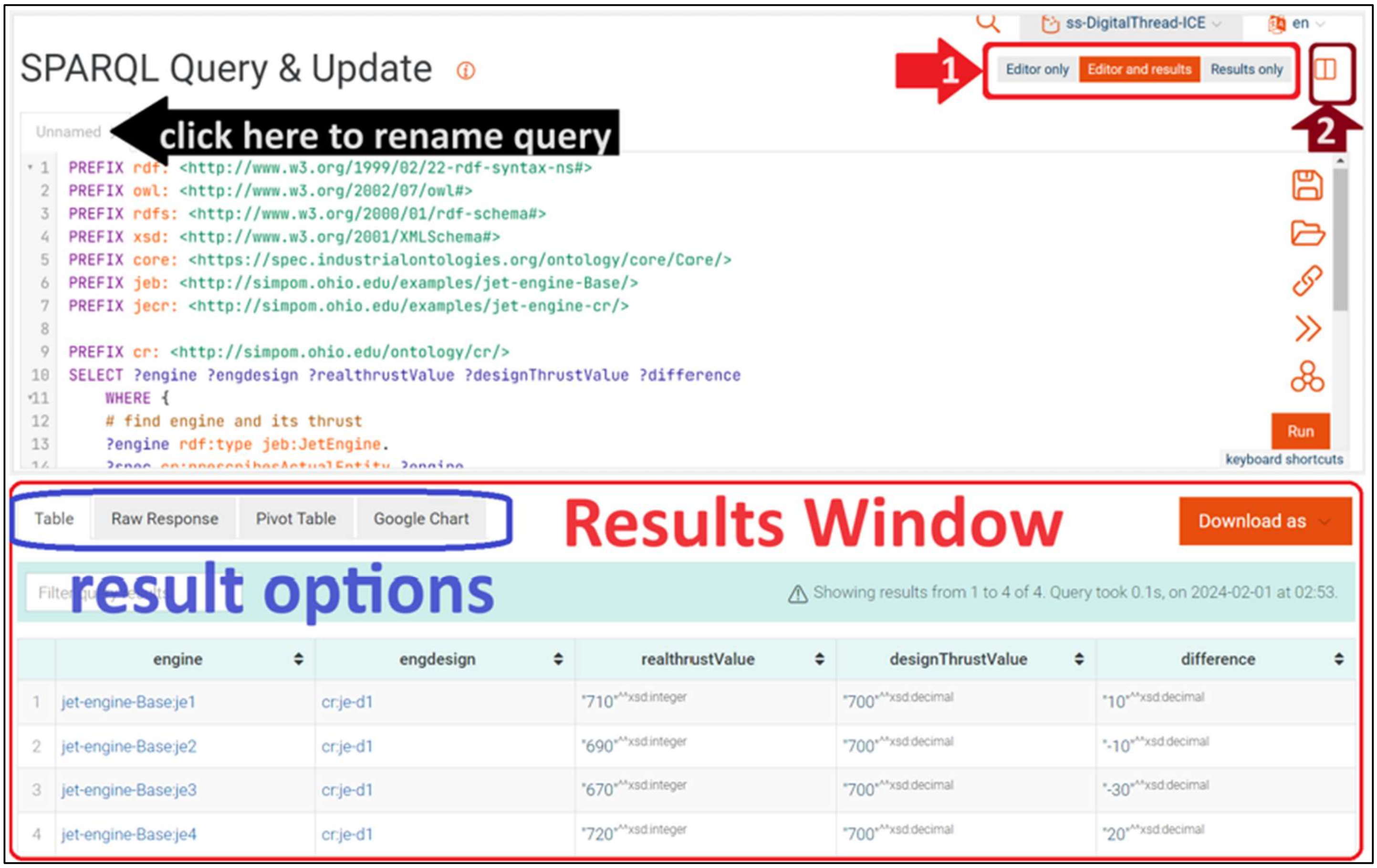
```
#What is the difference of real engine thrust and designed engine thrust?
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX core: <https://spec.industrialontologies.org/ontology/core/Core/>
PREFIX jeb: <http://simpom.ohio.edu/examples/jet-engine-Base/>
PREFIX jecr: <http://simpom.ohio.edu/examples/jet-engine-cr/>
PREFIX cr: <http://simpom.ohio.edu/ontology/cr/>

SELECT ?engine ?realThrustValue ?designThrustValue ?difference
WHERE {
# find engine and its thrust
        ?engine rdf:type jeb:JetEngine.
        ?spec cr:prescribesProducedEntity ?engine.
        ?engine core:hasQuality ?thrust.
# find the thrust value
        ?thrust core:hasValueExpressionAtAllTimes ?thrustValueExpr.
        ?thrustValueExpr core:hasSimpleExpressionValue ?realThrustValue.
# find the engine design from associated spec
        ?spec cr:prescribesDesignedEntity ?engdesign.
        ?engdesign core:hasQuality ?thrustdesign.
# find design thrust value
        ?thrustdesign core:hasValueExpressionAtAllTimes ?thrustDesignValueExpr.
        ?thrustDesignValueExpr jeb:hasLowerBoundValue ?designThrustValue.
# compare them
BIND((?realThrustValue - ?designThrustValue) as ?difference)
        }
```

IMPORTANT: These questions should cover the range of knowledge that your ontology is expected to represent. For example, if your ontology deals with a library system, a competency question might be, "Which books are authored by 'J.K. Rowling'?"

2.5.1 Running SPARQL Query

1. On the menu bar on the left side of your screen, select **"SPARQL"** to navigate to the SPARQL View.
2. Copy the provided code in the previous page (15), and paste it in the Editor section.



```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX core: <https://spec.industrialontologies.org/ontology/core/Core/>
PREFIX jeb: <http://simpom.ohio.edu/examples/jet-engine-Base/>
PREFIX jecr: <http://simpom.ohio.edu/examples/jet-engine-cr/>

PREFIX cr: <http://simpom.ohio.edu/ontology/cr/>
SELECT ?engine ?engdesign ?realthrustValue ?designThrustValue ?difference
    WHERE {
    # find engine and its thrust
    ?engine rdf:type jeb:JetEngine.
    ?spec cr:prescribesActualEntity ?engine
```

Showing results from 1 to 4 of 4. Query took 0.1s, on 2024-02-01 at 02:53.

	engine	engdesign	realthrustValue	designThrustValue	difference
1	jet-engine-Base:je1	cr:je-d1	"710"^^xsd:integer	"700"^^xsd:decimal	"10"^^xsd:decimal
2	jet-engine-Base:je2	cr:je-d1	"690"^^xsd:integer	"700"^^xsd:decimal	"-10"^^xsd:decimal
3	jet-engine-Base:je3	cr:je-d1	"670"^^xsd:integer	"700"^^xsd:decimal	"-30"^^xsd:decimal
4	jet-engine-Base:je4	cr:je-d1	"720"^^xsd:integer	"700"^^xsd:decimal	"20"^^xsd:decimal

Figure 26: SPARQL Window View.

3. Once you've entered the code, click **"Run"** or use Ctrl + Enter (or Cmd + Enter on Mac) to execute the query and display the results.
4. Various result formats are available, as shown in Figure 26 (result options), allowing you to choose how you would like the results to be presented.

Tip: GraphDB offers two viewing/editing modes - horizontal and vertical (Figure 26 '2'). In both horizontal and vertical modes, you have the flexibility to hide either the editor or the results (Figure 26 '1'). This allows you to concentrate on either crafting your query or examining the results. Use the "Editor only," "Editor and results," or "Results only" buttons to toggle between these different views according to your needs.

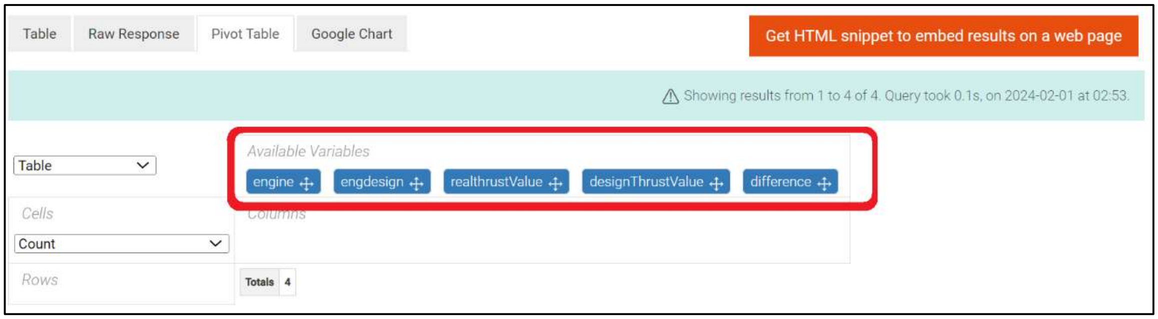
2.5.2 Regular Table Result

1. Table result (regular table result) is an option that Protégé also provides us. The difference in GraphDB is a better looking interface and the query Editor is easy to follow since it offers syntax highlighting and namespace autocompletion for easy reading and writing.
2. The result displaying by the Table tab is shown above in Figure 26.

2.5.3 Pivot Table

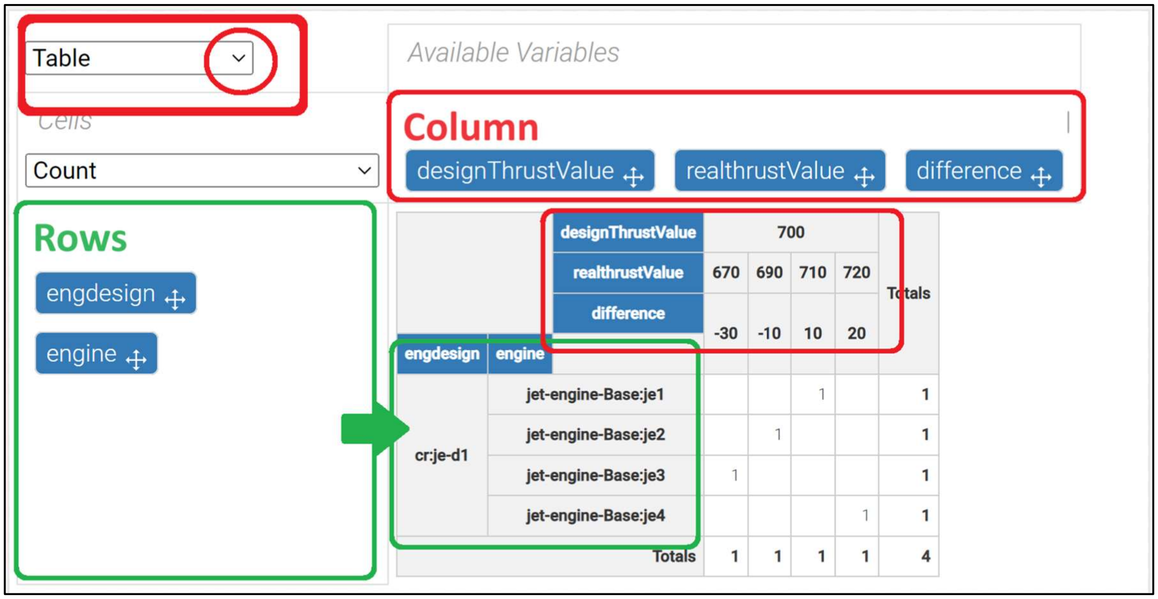
1. Choose the "Pivot Table" tab to view the pivot table representation of the query results.
2. Given that the query specifies **SELECT** *?engine ?engdesign ?realthrustValue ?designThrustValue ?difference*, the results will return the available variables: *engine, engdesign, realthrustValue, designThrustValue, and difference.*

NOTE: **1)** The **SELECT** clause in SPARQL is utilized to define which variables in a query should be returned in the results.
2) The total count of results and the query execution time are indicated in the header of the query results.
3) You can drag and drop each of the available variables to organize the pivot table according to your preferences, allowing you to view the data in a specific arrangement.



⚠ Showing results from 1 to 4 of 4. Query took 0.1s, on 2024-02-01 at 02:53.

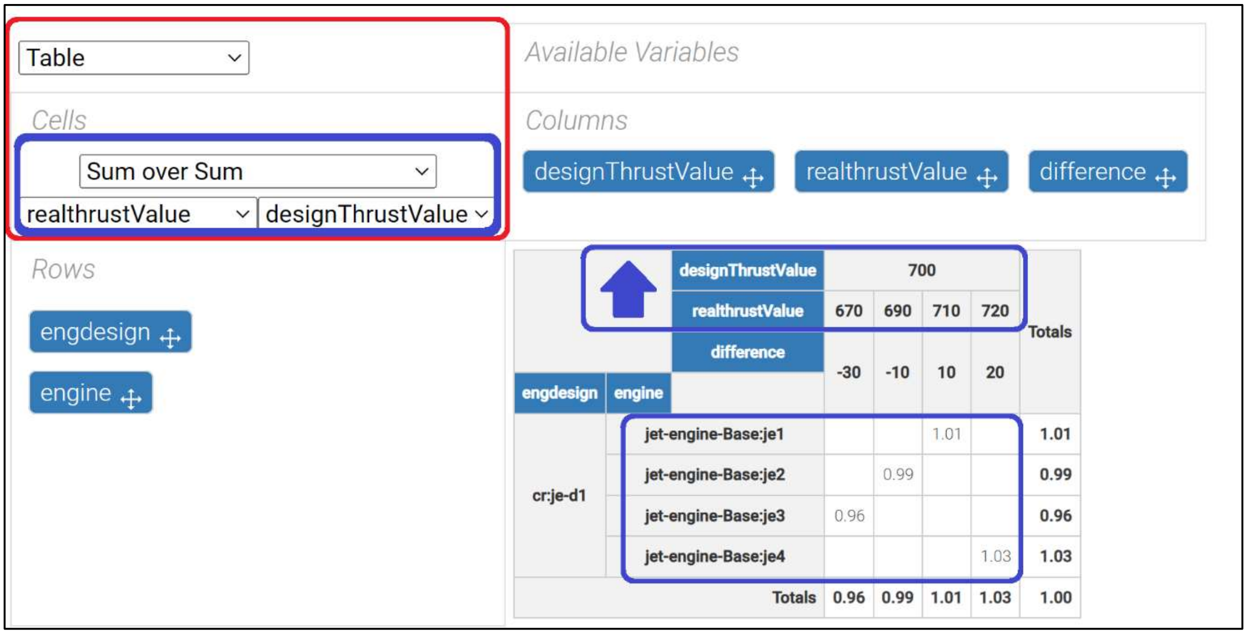
Figure 27: Pivot Table (Available Variables).



	designThrustValue	700				Totals
	realthrustValue	670	690	710	720	
	difference	-30	-10	10	20	
engdesign	engine					
cr:je-d1	jet-engine-Base:je1			1		1
	jet-engine-Base:je2		1			1
	jet-engine-Base:je3	1				1
	jet-engine-Base:je4				1	1
	Totals	1	1	1	1	4

Figure 28: Pivot Table Details.

3. For instance, you might arrange ***'engdesign'* and *'engine'* as the rows data**, with the **remaining variables set as columns**, as depicted above. *Note** The order of variable matters.*
4. Additionally, you can tailor the output format of the pivot table to better suit your data analysis needs by selecting from the available options in the dropdown menu circled in red. The options include:
    a. Table, Table Barchart, Heatmap, Row Heatmap, Col Heatmap, Treemap, Line Chart, Bar Chart, Stacked Bar Chart, Area Chart, and Scatter Chart.
5. Another useful feature is the ability to perform minor statistical analyses and calculations on your results. This can be accessed by selecting the dropdown arrow below **"Cells"** (located under the red circle in the figure mentioned above).



Table

Cells

Sum over Sum

realthrustValue | designThrustValue

Available Variables

Columns

designThrustValue | realthrustValue | difference

Rows

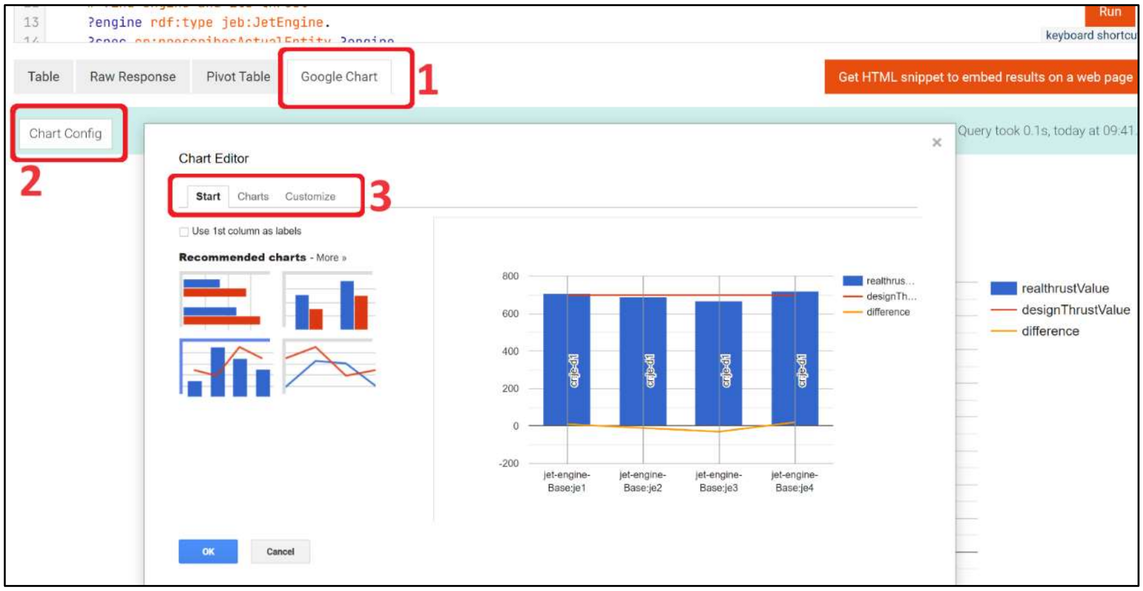
engdesign | engine

		designThrustValue	700				Totals
		realthrustValue	670	690	710	720	
		difference	-30	-10	10	20	
engdesign	engine						
cr:je-d1	jet-engine-Base:je1				1.01		1.01
	jet-engine-Base:je2			0.99			0.99
	jet-engine-Base:je3		0.96				0.96
	jet-engine-Base:je4					1.03	1.03
		Totals	0.96	0.99	1.01	1.03	1.00

Figure 29: Using Pivot Table for Simple Analyses.

2.5.4 Google Chart

1. An important feature to note is that Google Chart is designed to enhance the visualization of query results directly from SPARQL queries.
2. The Google Chart function takes the query results and dynamically creates a chart to visualize this data. This chart can be displayed within the GraphDB interface or on a web page, enabling users to instantly observe patterns, trends, and insights within their dataset.
3. With Google Chart, you can determine how the query results are visualized by selecting the type of chart (such as a bar chart, line chart, or pie chart), configuring data series, axes, labels, and other visualization parameters to tailor the display to your needs.
4. To begin customizing, click on the **"Google Chart"** tab (marked as '1'), then choose **"Chart Config"** (marked as '2'). From there, you can select your customization options from the tab in the third section.



```
?engine rdf:type jeb:JetEngine.
```

Figure 30: Google Chart Configurations.

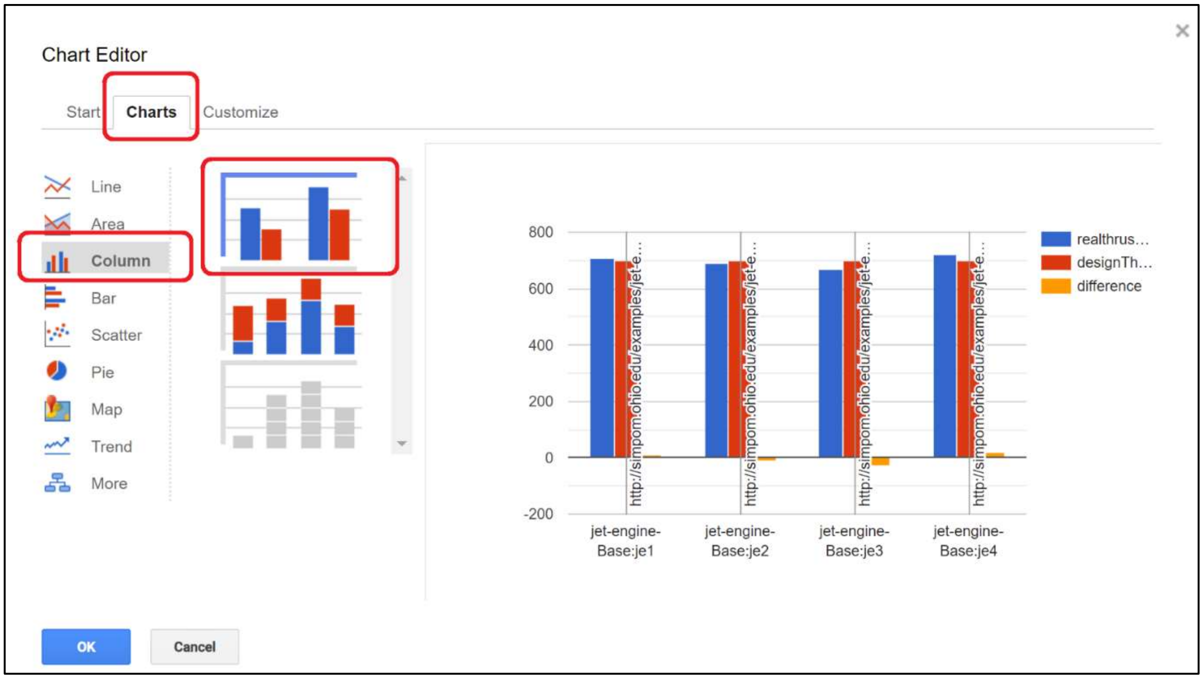


Figure 31: Google Chart Configurations (2).

2.6 Exercise on GraphDB

1. **Create a New Repository:** Start by creating a new repository. Name it something relevant to ICE (Information Content Entity) to reflect the focus of this exercise.
2. **Import Files:** Import all necessary files into your newly created repository. Instead of importing ~~jet-engine-CR.rdf~~, import jet-engine-ICE.rdf. This change targets a different aspect of the ontology, focusing on Information Content Entities.
3. **Knowledge Graph Differences:** Once you've imported the files, explore the Knowledge Graph. Observe the differences compared to the *CR (Counterpart Relation)* model. Consider how Information Content Entities are represented and related within this new context.
4. **Edit SPARQL Queries:** Adjust your SPARQL queries to align with the new ontology. This may involve changing PREFIX definitions and other necessary components to ensure your queries are compatible with the ICE-focused ontology.
5. **Explore Result Functions:** With your updated SPARQL query, run and explore the different result functions available in GraphDB. Pay attention to how the results differ from those related to the CR ontology and what insights they might offer regarding Information Content Entities.