PhUSE EUConnect18

Linked Data for Clinical Trials

An Interactive Hands-on Workshop

EXERCISES

Version 3.1

2018-11-05

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  |  |  |  |

Contents

[Introduction 2](#_Toc527635216)

[Server Login and Preparation 2](#_Toc527635217)

[Exercises 3](#_Toc527635218)

[1. Create a Study Graph 3](#_Toc527635219)

[1.1 Open the Graph Editor 3](#_Toc527635220)

[1.2 Add a Study, Study Phase, and Treatment Arms 4](#_Toc527635221)

[1.3 Add Persons and Treatments 5](#_Toc527635222)

[1.4 Add Demographics 6](#_Toc527635223)

[1.5 Add another Person 7](#_Toc527635224)

[1.6 Review and Export to TTL 8](#_Toc527635225)

[1.7 Validate the Data 8](#_Toc527635226)

[1.8 Upload to Database 10](#_Toc527635227)

[2. Query Graph Data 10](#_Toc527635228)

[2.1 Show all triples 10](#_Toc527635229)

[2.2 How many people participateIn the study? 11](#_Toc527635230)

[2.3 Find the names of people who participate in the study. 11](#_Toc527635231)

[2.4 List the names of the people randomized to each treatment type. 12](#_Toc527635232)

[3. Ontology and Inference 12](#_Toc527635233)

[3.1 Review the Ontology 12](#_Toc527635234)

[3.2 Add the Ontology to the LDWStudy database 12](#_Toc527635235)

[3.3 Find the names of HumanStudySubjects in your study. 13](#_Toc527635236)

[3.4 Write a reasoner-based query to find all types (classes) of Persons 14](#_Toc527635237)

[4. Merge Studies 14](#_Toc527635238)

[4.1 Create the Data Pool 14](#_Toc527635239)

[4.2 List Drug1 studies by phase. 15](#_Toc527635240)

[4.3 How many HumanStudySubjects participated in all Drug1 studies? 15](#_Toc527635241)

[4.4 How many women received active treatment (non placebo) across all Drug1 studies combined? 15](#_Toc527635242)

[4.5 List the Types of Persons in the studies. 15](#_Toc527635243)

[Appendix 1: Course Resources 17](#_Toc527635244)

[Appendix 2: Update Files using TortoiseGit 17](#_Toc527635245)

# Introduction

Instructions in this document are specific to the PhUSE cloud server environment, Graph Editor application and workshop methods. The exercises represent one of many possible approaches to the material and make no claim to be best or recommended method. An instructor will complete the exercises at the same time as the class.

Your feedback is welcomed and encouraged. Please send your comments to: **tim.williams@PhUSE.eu**

**Materials**

Printed copies of the following will be supplied by the instructor:

* Exercises (this document)
* Info Sheet (unique to each person)
* Graph Editor Reference
* SPARQL Reference Sheet

All course content is available on GitHub at:

https://github.com/phuse-org/LinkedDataWorkshop/tree/master/**EUConnect18**

**Symbols**

|  |  |
| --- | --- |
|  | Stop and wait. Do not proceed to the next section until instructed. |
| **!** | Cautions and warnings. Failure to follow instructions may lead to unanticipated problems. |
|  | Presentation by the instructor. |
| IdeaIcon_clean_20mm | Helpful tips and advice. |
| Image result for information icon | Additional information. |

# Server Login and Preparation

Instructions assume MS Windows operating system.

1. On your laptop, ensure your screen resolution is set to 1600 x 900 or higher.
2. In the search box on the taskbar on your laptop, type **remote desk,** then select **Remote Desktop Connection** from the Programs section.﻿
3. If needed, select **Show Options** to see the tabs in the next step.
4. Select the **Display** tab. Ensure the Display Configuration slider is all the way to the right for **Large,** **Full Screen**:



1. On the **General** tab, enter values in the **Computer:** and **User name:** supplied on your **Info Sheet**.
2. Click **Connect**.
3. When prompted, enter the password value from the **Info Sheet** and click **OK**.

|  |  |  |
| --- | --- | --- |
|  | Stop here and wait for the instructor.  Presentation follows |  |

# Exercises

# Create a Study Graph

The PhUSE Cloud Server environment is used for all exercises.

# Open the Graph Editor

1. Open Google Chrome using the shortcut on the server's desktop.
2. Click on  in the shortcuts bar.
3. The Graph Editor opens and shows nodes that include drug, drug name, gender, clinical trial phase, and others.

**Ask for assistance if you do not see these nodes.**



|  |  |
| --- | --- |
| IdeaIcon_clean_20mm | If the Graph Editor does not load, click "X" to cancel, then reload the page. Ask for assistance if the graph editor still fails to load. |

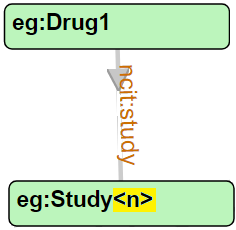
# Add a Study, Study Phase, and Treatment Arms

|  |  |
| --- | --- |
| **!** | Values unique to your graph are found on your **Info Sheet** and referenced in the exercises as <Info Sheet:[field]> . |

1. Create a **Study*n*** node. Follow along with the instructor to create a study node using the value from your Info Sheet.

Node values:

|  |  |  |
| --- | --- | --- |
| **Type:** | IRI |  |
| **Prefix:** | eg |  |
| **Label:** | *<Info Sheet: Study>* | Image result for information icon*See Info Sheet for value* |



1. Create a predicate link ***from*** **Drug1** ***to*** the **Study*n*** node. You must edit the new link by double-clicking it and selecting the correct value from the list of available predicates, then click Update/Hide.

|  |  |
| --- | --- |
| **Predicate:** | ncit:study |

1. Create a link from your **Study*n*** node ***to*** a **Phase** node using the predicate **ncit:phase** . You may pick any one of the three available Phase nodes (Phase2, Phase3, Phase4) to assign a trial phase to your study. ***Link to only one of the three available nodes.***

|  |  |
| --- | --- |
| **Predicate:** | ncit:phase |

1. Create two Treatment Arm nodes for **Study*n*** using values from your **Info Sheet**. Match the upper and lowercase for the labels exactly.

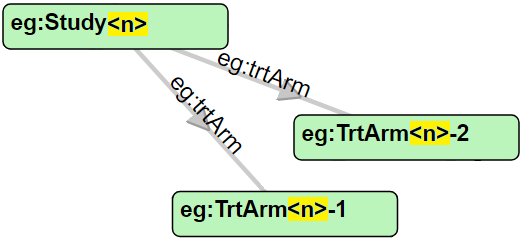
Node 1 values:

|  |  |  |
| --- | --- | --- |
| **Type:** | IRI |  |
| **Prefix:** | eg |  |
| **Label:** | *<Info Sheet: Treat Arm One>* | Image result for information icon*See Info Sheet for value.* |

Node 2 values:

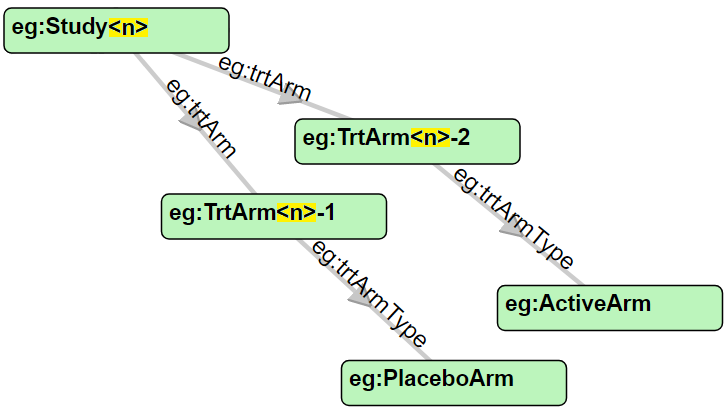
|  |  |  |
| --- | --- | --- |
| **Type:** | IRI |  |
| **Prefix:** | eg |  |
| **Label:** | *<Info Sheet: Treat Arm Two>* | Image result for information icon*See Info Sheet for value.* |

Ensure you have two Treatment Arm (**TrtArm***n-n*) nodes in your graph.

1. Create a link ***from*** **Study*n*** ***to*** each of the two Treatment Arm nodes using the predicate **eg:trtArm** ..

|  |  |
| --- | --- |
| **Predicate:** | eg:trtArm |

There should now be one **eg:trtArm** link from the Study to each of the Treatment Arm nodes.

1. Now join ***from*** each Treatment Arm node ***to*** one of the Treatment Type nodes (eg:ActiveArm, eg:PlaceboArm) using the predicate **eg:trtArmType**. It does not matter which arm is linked to the **PlaceboArm** or the **ActiveArm**.

|  |  |
| --- | --- |
| **Predicate:** | eg:trtArmType |

The Study and Treatment Arm portion of your graph should look similar to:

|  |  |
| --- | --- |
| IdeaIcon_clean_20mm | Click  to save your graph in case anything goes wrong in the next steps. |
| Image result for information icon | You now have two Treatment Arms that are unique to your study, assigned to one of two Treatment Arm Types: PlaceboArm, ActiveArm. These "arm type" nodes are identical across all Drug1 studies in the workshop and will allow identification of all persons who received active Drug1 treatment. In real life, the Treatment Arm would be further extended to include dosage and administration information unique to that study, resulting in a graph that would be too complex for the workshop. |

# Add Persons and Treatments

1. Add two Person nodes using values from your **Info Sheet**.

Person 1 node:

|  |  |  |
| --- | --- | --- |
| **Type:** | IRI |  |
| **Prefix:** | eg |  |
| **Label:** | *<Info Sheet: Person One>* | Image result for information icon*See Info Sheet for values.* |

Person 2 node:

|  |  |  |
| --- | --- | --- |
| **Type:** | IRI |  |
| **Prefix:** | eg |  |
| **Label:** | *<Info Sheet: Person Two>* | Image result for information icon*See Info Sheet for values.* |

1. Link ***from*** each of the two Person nodes ***to*** the **Study*n*** node using the **eg:participatesIn** predicate.

|  |  |
| --- | --- |
| **Predicate:** | eg: participatesIn |

1. Link ***from*** one of the Person nodes ***to*** one of the Treatment Arm (**trtArm*n-n***) nodes using the **eg:randomizedTo** predicate.

|  |  |
| --- | --- |
| **Predicate:** | eg:randomizedTo |

1. Link ***from*** the second Person node ***to*** the other Treatment Arm (**trtArm*n-n***) node using another **eg:randomizedTo** predicate.

|  |  |
| --- | --- |
| **Predicate:** | eg:randomizedTo |

Now each Person **participatesIn**the **Study*n*** and is **randomizedTo** a Treatment Arm.

# Add Demographics

1. Link ***from***the two Person nodes ***to*** a Gender node using the **ncit:gender** predicate.

|  |  |
| --- | --- |
| **Predicate:** | ncit:gender |

1. Create one Given Name(First name) for eachPersonnode in the graph. Names have Type: STRING because they are string literals.

Node values:

|  |  |  |
| --- | --- | --- |
| **Type:** | STRING |  |
| **Prefix:** | *NOT APPLICABLE* | Image result for information iconPrefix is not applicable to literals and will be ignored. |
| **Label:** | *<Any name you want>* |  |

1. Link ***from*** each Person node ***to*** their Given Name using the **schema:givenName** predicate.

|  |  |
| --- | --- |
| **Predicate:** | schema:givenName |

1. Create **Age** nodes for each Person as INT (integer) literals:

Node values:

|  |  |  |
| --- | --- | --- |
| **Type:** | INT |  |
| **Prefix:** | *NOT APPLICABLE* | Image result for information iconPrefix is not applicable to literals and will be ignored. |
| **Label:** | *<Any integer age value you want>* |  |

1. Link ***from***each Person ***to*** their age value using the **eg:age** predicate.

|  |  |  |
| --- | --- | --- |
| **Predicate:** | eg:age |  |

# Add another Person

1. Add another Person node to the graph:

Node values:

|  |  |  |
| --- | --- | --- |
| **Type:** | IRI |  |
| **Prefix:** | eg |  |
| **Label:** | *<Info Sheet: Person Three>* |  |

***This node will represent you and your role in this study.***

1. Create a new node for your Given Nameas a string literal.

Node values:

|  |  |  |
| --- | --- | --- |
| **Type:** | STRING |  |
| **Prefix:** | NOT APPLICABLE |  |
| **Label:** | *<Your given (first) name>* |  |

1. Link ***from*** your Person node ***to*** your Given Name using the **schema:givenName** predicate.

|  |  |  |
| --- | --- | --- |
| **Predicate:** | schema:givenName |  |

1. Link ***from*** your Person node ***to*** your gender using the **ncit:gender** predicate.

|  |  |
| --- | --- |
| **Predicate:** | ncit:gender |

*The polite instructor will not ask you to create an age node and link. :)*

1. Your role in this study is Linked Data Expert. Link ***from*** your Person node ***to*** the **Study*n*** node using the **eg:LDExpert** predicate.

|  |  |  |
| --- | --- | --- |
| **Predicate:** | eg:LDExpert |  |

1. Congratulate yourself for becoming a Linked Data Expert, then Click  to save your graph.

# Review your graph

1. Use this checklist to confirm proper spelling and upper/lower case for node names in your graph. Make any necessary changes.

|  |  |  |
| --- | --- | --- |
| **Node Name** |  | √ |
| **eg:Study*<n>*** | ***<n>*** *is the same number for Study, TrtArm, and Person* |  |
| **eg:TrtArm*<n>*-*<x>*** | ***<x>*** = Number 1, or 2 |  |
| **eg:Person*<n>* <y>** | **<y>** = Number 1, 2 or 3 |  |

1. Review your **predicates** by following the relationship ***from*** the Subject ***to*** the Object as indicated by the arrows in the graph. For example: "**Person21** **participatesIn** **Study2**" is correct. "**Study2** **particpatesIn** **Person21**" is wrong and the direction of the link must be corrected.
2. Save your graph if you make any changes to the values.

|  |  |  |
| --- | --- | --- |
|  | Stop here and wait for the instructor.  Ask the instructor to review your graph so the next steps will go smoothly. |  |

# Export to TTL

1. Click on  to export the graph to TTL format.
2. Click "OK" to confirm.
3. If a "Download multiple files" message appears, click **Allow**.
4. Two files will appear at the bottom of the Chrome window: **WhiteBoardTriples.TTL** and **whiteboard.json.**
5. Click the dropdown arrow beside **WhiteBoardTriples.TTL** and select **Open.**
6. Follow along with the instructor's explanation of the TTL file content.
7. Find the various ***Subjects, Predicates,*** and ***Objects*** you created in your graph.

|  |  |
| --- | --- |
| Image result for information icon | If you create the TTL more than once, you will see multiple files that include a number in their name:  WhiteBoardTriples(1).TTL  WhiteBoardTriples(2).TTL  etc.  Always use the highest numbered (most recent) file. |

1. If you find any errors, go back to the Graph Editor to fix the nodes or links, then Create TTL again using the steps in this section.

|  |  |
| --- | --- |
| **!** | Leave the graph open in Chrome. You will refer to it later when creating queries. Ask for assistance if you accidentally close or refresh the Graph Editor window. |

# Validate the Data

1. Click on the **RScripts** shortcut  on the desktop to open the ...scripts/R folder.
2. Double-click the file **TTLValidation.R** to open it into RStudio.
3. ****Run the code by clicking the Run App icon 
4. Click **Browse...** under **.TTL File**
5. **** Scroll up in the left pane to find the **Quick access** links and select **Downloads**.
6. Double-click on **WhiteBoardTriples.TTL** to load the file into the app.

|  |  |
| --- | --- |
| **!** | Remember to load the most recent version of your TTL file, which may have a number in the name : WhiteBoardTriples(1).TTL, WhiteBoardTriples(2).TTL , etc. Ask for assistance if needed. |

1. A default query appears in the app. Click **Run query** to execute the query and view the result in the **Query Result:** area.
2. Review the values in the **Query Result** and compare them to your graph.
3. Click on the  tab. If the message "All QC Checks Passed", proceed to **Step 10**. ***If not all checks pass:***
   1. Review the values listed and compare them to your graph. Do the values or their prefixes need to change to match the values given in the exercises?
   2. Ask for assistance if the problem is not obvious.
   3. Return the Graph Editor to make corrections as needed.
   4. Export to TTL again using the steps in section **1.7 Export to TTL**, then re-run the validation steps in this section until you receive the message "All QC Checks Passed" or the instructor approves any items displayed on the QC Check page.

|  |  |
| --- | --- |
| **!** | Unless advised by the instructor, do not proceed past this step until the QC Check page displays "All QC checks passed". Failure to validate your graph may cause errors in later exercises. |

1. Click on the  tab at the top of the app to view a network graph of the query result.
2. Explore the visualization:

|  |  |  |
| --- | --- | --- |
|  | **Mouse** | **Touchpad** |
| Reposition graph | Hold down the left mouse button and drag | Hold finger on touchpad, drag |
| Zoom | Scroll wheel up/down | Move two fingers on touchpad together to zoom in, apart to zoom out |
| View labels | Position cursor over node or link | Position cursor over node or link |
| Node Connections | Click on node | Tap on node |

1. Close both the **RShiny app** and **RStudio** when you are done browsing your graph. If prompted to "Save workspace image...", Click "Don't Save".

|  |  |  |
| --- | --- | --- |
|  | Stop here and wait for the instructor. |  |

# Upload to Database

1. Open a new tab in Google Chrome, **keeping your graph editor tab open.**
2. Click the **Stardog** shortcut on the Chrome bookmarks bar.
3. In the Stardog application window, click **Databases** on the top menu.Under **Databases**, click the row for **LDWStudy.**



1. In the upper left, click 

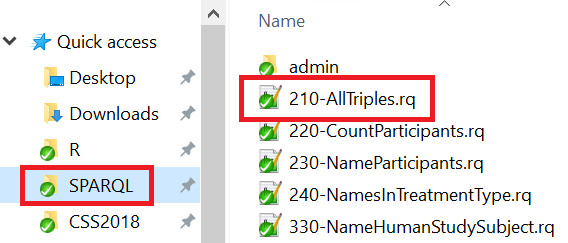


1. Select **Data | Add** from the Stardog menu:
2. Click **Browse.**
3. Select the **Quick access | Downloads** folder and double-click **WhiteBoardTriples.ttl** to select the file.
4. Click **Upload**.
5. You should see the message: ***Success! Data added successfully.***

# Query Graph Data

# Show all triples

Find all the Subjects, Predicates, and Objects in your graph.

1. If the "Query Panel" is not displayed, click on the **>\_Query** menu item at the top of the Stardog display.
2. Click the Open Folder icon  in the Query Panel.
3. Select the **Quick Access | SPARQL** folder and double-click the file **210-AllTriples.rq** to load it into the query window.
4. With the query loaded in to the window, click the  button. Scroll down to review the results.
5. Compare this view to what you expected after viewing the TTL file in an earlier exercise.

# How many people *participateIn* the study?

Find the number persons participating in your study by counting the number of **participatesIn**predicates.

1. Load the SPARQL query **220-CountParticipants.rq**.
2. Examine the query and refer back to your graph in the Graph Editor while the instructor explains the query.
3. Execute the query.
4. Is this this result you expected, based on your graph? Why are you (the Linked Data Expert) not included in the graph?

# Find the names of people who participate in the study.

In this exercise you want to obtain the givenName values of the Persons who participate in the study. Refer back to your graph in the Graph Editor to create the query using the steps below, or skip down to the SOLUTION: to load the query file.

Try modifying the query from **Exercise 2.2** using the next steps as a guide.

1. Return the person's name (givenName) in the SELECT statement:

SELECT ?person\_name

1. In the WHERE statement, identify Person nodes that have the **participatesIn** predicate:

?person eg:participatesIn ?study ;

1. Then follow from that same Person node using the **givenName** predicate to find their name.

?person eg:participatesIn ?study ;

schema:givenName ?person\_name .

1. Execute your query and view the results. If you have trouble writing the query, load and execute the file:

SOLUTION: **230-NameParticipants.rq**

# List the names of the people randomized to each treatment type.

1. Modify the SELECT statement from **Exercise 2.3** to select treatment type in addition to person name variables.

SELECT ?trt\_type ?person\_name

1. Change the predicate value in the first line of the WHERE statement to use **eg:randomizedTo** , as follows:
2. ?person eg:randomizedTo ?trt\_arm ;

schema:givenName ?person\_name .

1. Refer to your graph in the Graph Editor. You want to follow the path from **?trt\_arm** along the **eg:trtArmType** predicate to the type of treatment ( **?trt\_type** ) .

?person eg:randomizedTo ?trt\_arm ;

schema:givenName ?person\_name .

?trt\_arm eg:trtArmType ?trt\_type .

1. Add an ORDER BY statement after the WHERE to sort your results. ORDER BY appears after the } closing bracket:

ORDER BY ?trt\_type ?person\_name

SOLUTION: **240-NamesInTreatmentType.rq**

|  |  |  |
| --- | --- | --- |
|  | Stop here and wait for the instructor.  Presentation on Ontologies and Reasoning |  |

# Ontology and Inference

In this section you will add an ontology to your graph data and employ a reasoner.

# Review the Ontology

1. Click on the **Data** shortcut  on the desktop to open the .../data folder.
2. Double-click on the file **StudyOntology.TTL** to open it into a text editor.
3. Follow along with the instructor's explanation of the file.

# Add the Ontology to the LDWStudy database

1. Return to the Stardog application in Chrome.
2. Select **Data | Add** from the Stardog menu.
3. Click **Browse.**
4. Select the **Quick access | data** folder and double-click **StudyOntology.TTL** to select the file.
5. Click **Upload**.
6. You should see the message: **Success! Data added successfully.**

# Find the names of HumanStudySubjects in your study.

*Your original data contains no definition of a HumanStudySubject.*

|  |  |
| --- | --- |
|  | Follow along with the instructor as they describe how HumanStudySubject can be inferred by the ontology using the links and entities in your original data. |

|  |  |
| --- | --- |
| IdeaIcon_clean_20mm | In SPARQL queries, the **rdf:type** link can be shorted to **a** and interpreted as "*is a*".  Example:  You read: eg:Person11 rdf:type eg:HumanStudySubject  as: "Person11 ***is a*** HumanStudySubject."  Instead of querying using **participatesIn** , as in:  ?person eg:participatesIn ?study ;  With a Reasoner you can query using the *inferred* entity **eg:HumanStudySubject**.  ?person a eg:HumanStudySubject; |

1. Open the query file **230-NameParticipants.rq** that you used in **Exercise 2.3 .** Using the information in the advice box IdeaIcon_clean_20mmabove this step to modify the queryto find ?person "is a" HumanStudySubject.
2. Turn on the reasoner by clicking the **Reasoning** button.
3. **Execute** the query.
4. Refer back to your graph in the Graph Editor and be amazed! There is no "Human Study Subject" defined in your original graph. Using an ontology and reasoner *inferred* new information.
5. What happens if you turn OFF the reasoner and execute the query? Don't forgot to turn the reasoner back ON for the next exercise.

SOLUTION: **330-NameHumanStudySubject.rq** (found under Quick access , SPARQL)

# Write a reasoner-based query to find all types (classes) of Persons

Query to find all persons associated with the study, not just *participants*.

1. Write a query that uses the ***superclass*** of the **HumanStudySubject** and **DataExpert** classes.

CAUTION: The superclass prefix is not "eg" because it is defined in an external ontology!

1. Return the following values: Person IRI, Given Name of the Person, and the Person Type

HINT: Use the "is a" predicate: ?person a ?person\_type

SOLUTION: **340-NamesPersonTypes.rq**

1. The query result returns all the inferred "types of things" for Person nodes, not just their role in the study. You can return just the roles in the study by filtering on the http name space used for the Study graph.

FILTER ( strstarts(str(?person\_type), "http://example.org/LDWorkshop#") )

SOLUTION: **341-PersonStudyTypes.rq**

|  |  |  |
| --- | --- | --- |
|  | Stop here and wait for the instructor.  Presentation on Merging Data |  |

# Merge Studies

# Create the Data Pool

|  |  |
| --- | --- |
| **!** | The instructor will perform these steps first and provide instruction on how the class should proceed to create the pool. |

1. Click on **Admin Console** at the top of the Stardog application. 



1. Click on **Drug1Pool** in the Databases panel.
2. In the upper left, click 
3. Open the query **410-PoolAllStudies.rq** into the query window an execute it. A blue progress line will display briefly at the top of the application while the query is executing.
4. After the query has completed, execute the query **415-PoolTripleCount.rq**. Compare your triple count to the number provided by the instructor to ensure you have the correct data.

# List Drug1 studies by phase.

1. Load the query **420-StudiesByPhase.rq** .
2. Execute the query and locate your study in the results.

BONUS Question: How would you alter the query to obtain the number of studies in each phase?

Hint: a) In the **SELECT** statement: **?phase** (**COUNT**(**?phase**) **AS** **?count**)

b) Use a **GROUP BY** after the **WHERE**.

SOLUTION to BONUS: **425-StudyCountByPhase.rq**

# How many HumanStudySubjects participated in all Drug1 studies?

1. Ensure the **Reasoner** is turned **On**.
2. Load the query **430-Drug1CountHSS.rq** and examine the query.
3. Execute the query.

|  |  |
| --- | --- |
| IdeaIcon_clean_20mm | If the count is 0, you may need to turn ON the Reasoner. |

BONUS Question: Alter the query to count **all people** involved in Drug1 Studies (not just HumanStudySubjects).

HINT: a) What is the superclass of **HumanStudySubject**?

b) Ensure you are using the correct prefix!

SOLUTION to BONUS: **435-Drug1CountPeople.rq**

# How many women received active treatment (non placebo) across all Drug1 studies combined?

1. Use the graph of your study in the Graph Editor to determine the paths that need to be followed to obtain the solution.

HINT : follow the pattern: ?person -- gender -- **Female** ;

-- randomizedTo --- ?trt\_arm .

?trt\_arm -- treatmentArmType -- **ActiveArm**.

SOLUTION: **440-Drug1WomenActiveTreat.rq**

# List the Types of Persons in the studies.

*You may be getting tired so consider skipping directly to the Solution to see one way of solving this question.*

1. List all Person IRIs, their givenName, their assigned Study, and their role in that study (as a **HumanStudySubject** or **DataExpert**). Ensure the Reasoner is turned ON.

HINT: a) Find **Person**s using the logical OR operator for their role in the study:

?person eg:participatesIn|eg:LDExpert ?study.

b) To return only "person types", you need to use the ontology's Person subclasses:

?personType rdfs:subClassOf schema:Person ;

1. Review the results. **Do you find anything surprising in the pooled results?**

SOLUTION: **450-Drug1PersonsAllStudies.rq**

Congratulations- You've made to the End of the Exercises!

Please Sign out from the server using **Windows | Person Icon | Sign out**.

|  |  |  |
| --- | --- | --- |
|  | Stop here and wait for the instructor.  Presentation follows |  |

# Appendix 1: Course Resources

* [Introduction to Semantic Web](https://www-stage.cambridgesemantics.com/semantic-university/introduction-semantic-web)

https://www-stage.cambridgesemantics.com/semantic-university/introduction-semantic-web

* [What is Linked Data?](https://www-stage.cambridgesemantics.com/semantic-university/what-linked-data)

https://www-stage.cambridgesemantics.com/semantic-university/what-linked-data

* [Introduction to Linked data](https://www-stage.cambridgesemantics.com/semantic-university/introduction-linked-data)

https://www-stage.cambridgesemantics.com/semantic-university/introduction-linked-data

* [SPARQL in 11 minutes](https://www.youtube.com/watch?v=FvGndkpa4K0)

https://www.youtube.com/watch?v=FvGndkpa4K0

* [Learning SPARQL](http://www.learningsparql.com/)

http://www.learningsparql.com/

-------------------------------------------------------------------------------------------------------

[Hands-on Workshop Github Repository](https://github.com/phuse-org/LinkedDataWorkshop/tree/master/CSS2018)

Course content is available at:

https://github.com/phuse-org/LinkedDataWorkshop/tree/master/EUConnect18

# Appendix 2: Update Files using TortoiseGit

Follow these steps if the instructor asks you to update the exercise files.

1. Click on the **LDWorkshop** shortcut  on the desktop.
2. Right-click on the **EUConnect18** folder and select **Git Sync** from the menu.



1. Click **Pull**
2. Click **Close** when the command completes. You have updated all content to the latest version as needed for the following exercises.