SPHN RDF Training Primer

Vasundra Touré and Philip Krauss

Data interoperability team, Personalized Health Informatics Group, SIB Swiss Institute of Bioinformatics

A project of SAMW ASSM Schweizerische Akademie der Medizinischen Wissenschaften, Académie Suisse des Sciences Médicales, Accademia Svizzera delle Scienze Mediche, Swiss Academy of Medical Sciences

SIB Swiss Institute of Bioinformatics

Agenda

1. Strategy
2. Basic Components
   - Ontology
   - RDF
   - SPARQL

Wishlist : F.A.I.R. and more

Schema Softness

- I want to be able to describe data but allow certain levels of freedom — Ontologies
- I don’t want to be limited by tables

Accessibility

- I want that all information is stored and queried in open formats and can be read also in 10+ years — RDF + SPARQL

Interoperability

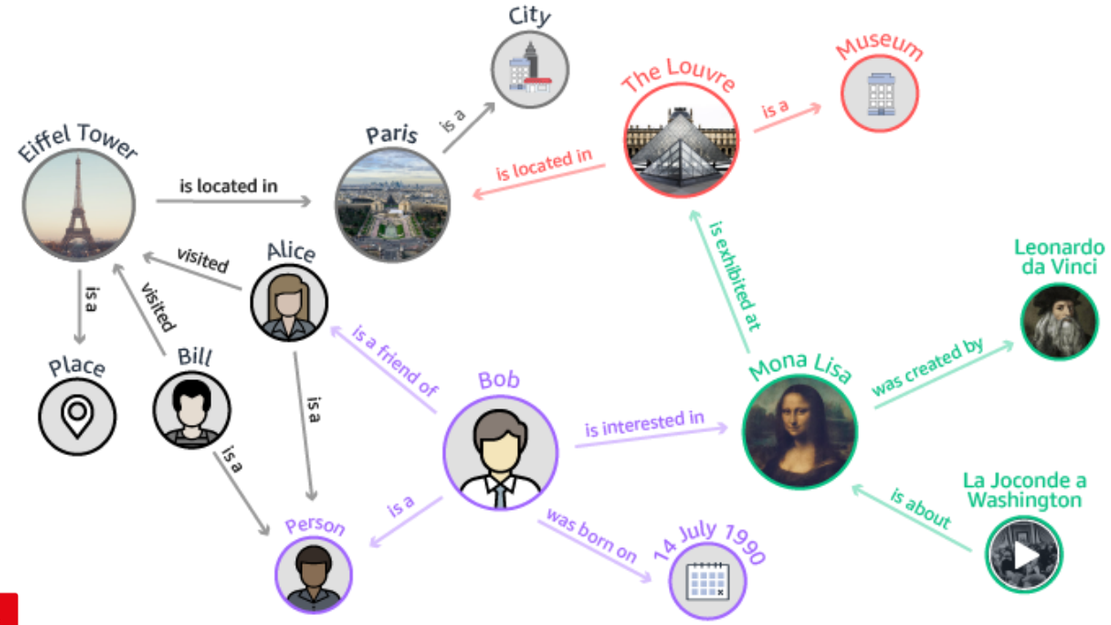
- I want that information can be created, stored, exchanged, merged, read and queried by different systems, leading to the same information — Semantic Web
- I want that information that means the same thing is the same thing
- I want make use of information that is already available

Semantic Web – a vision in 2001

Semantic Web, a visionary concept proposed in 2001 by Sir Tim Berners-Lee, the inventor of the World Wide Web

"I have a dream for the Web [in which computers] become capable of analyzing all the data on the Web – the content, links, and transactions between people and computers. .."

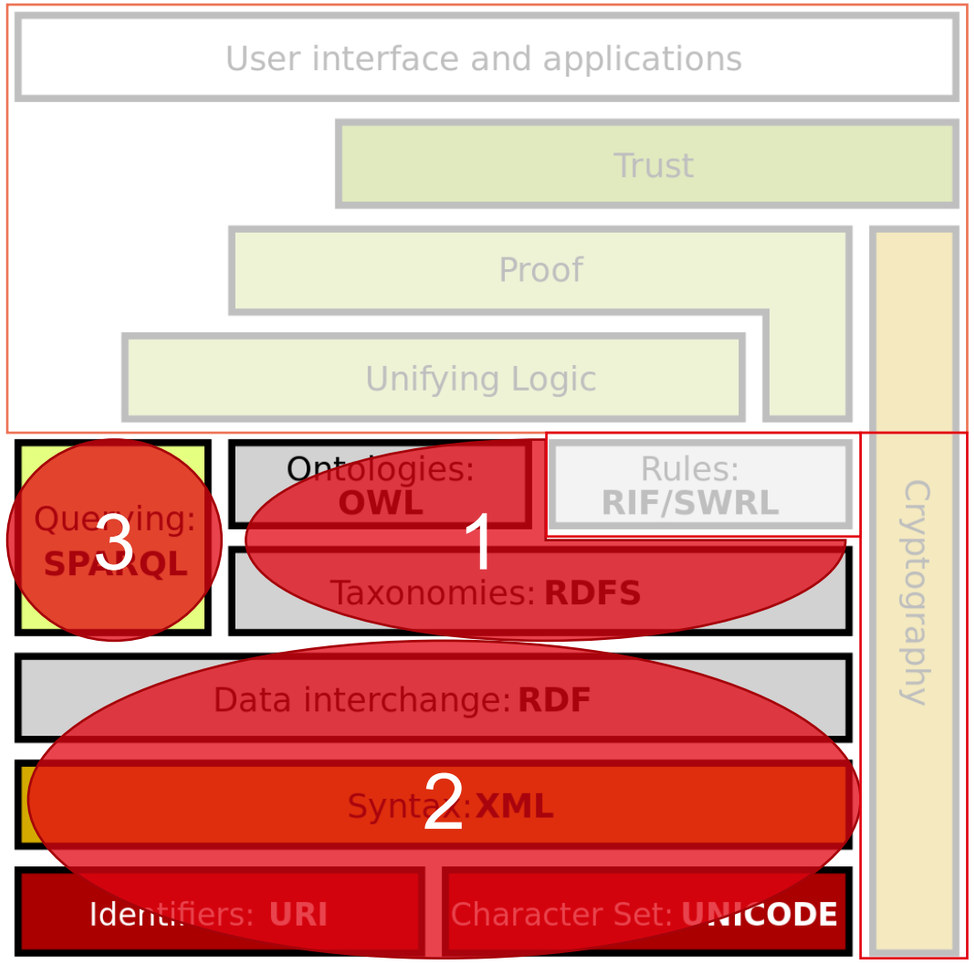
Sir Tim Berners-Lee, fossbytes.com, 2016



Amazon illustrates Knowledge Graph in Neptune

Semantic Web Layer Cake

1. Ontologies
2. RDF
3. SPARQL

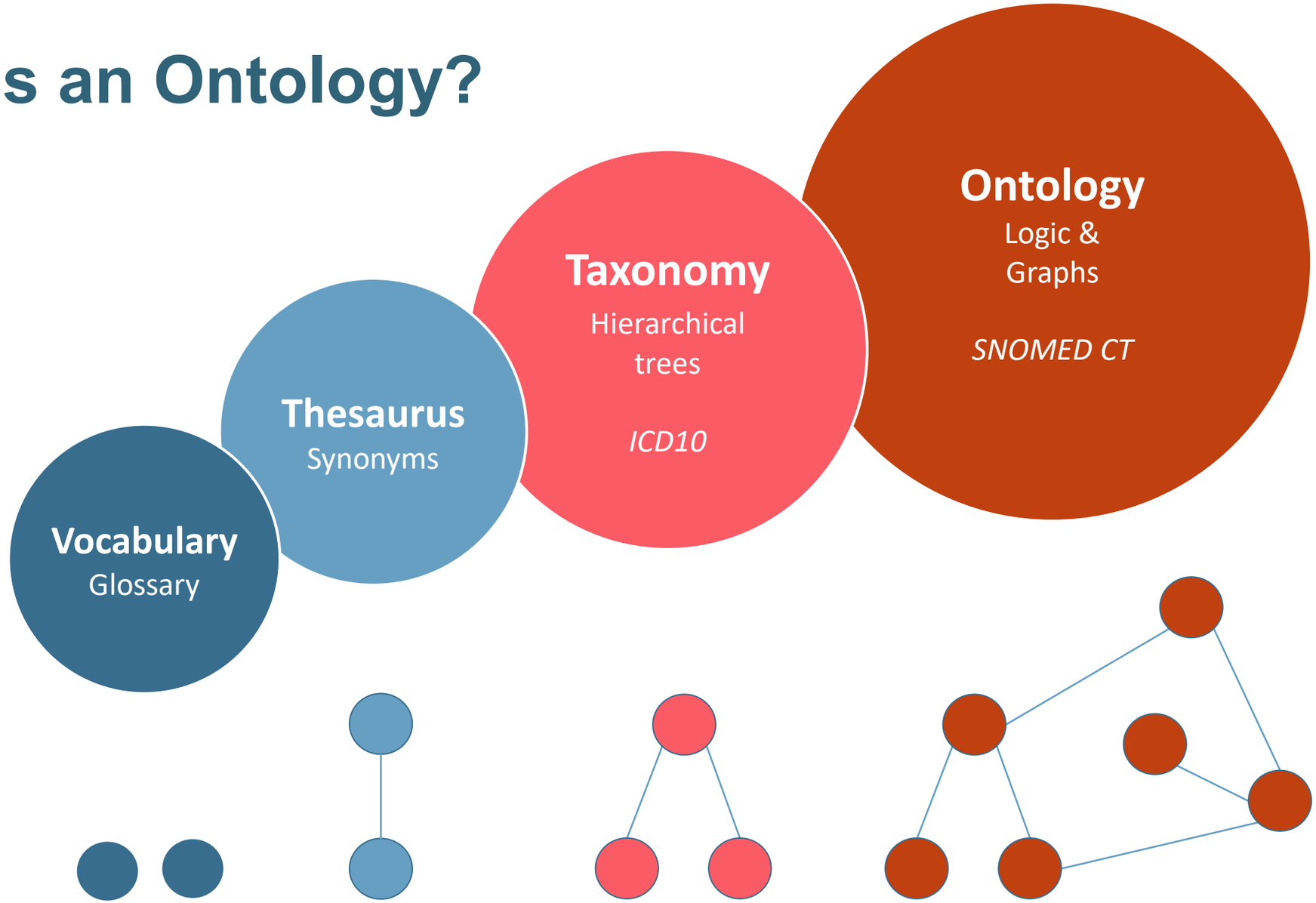


Ontologies

e.g. OWL Ontology Web Language
A W3C Standard

“How to formally describe Meaning of Information”

What is an Ontology?



What is an Ontology?

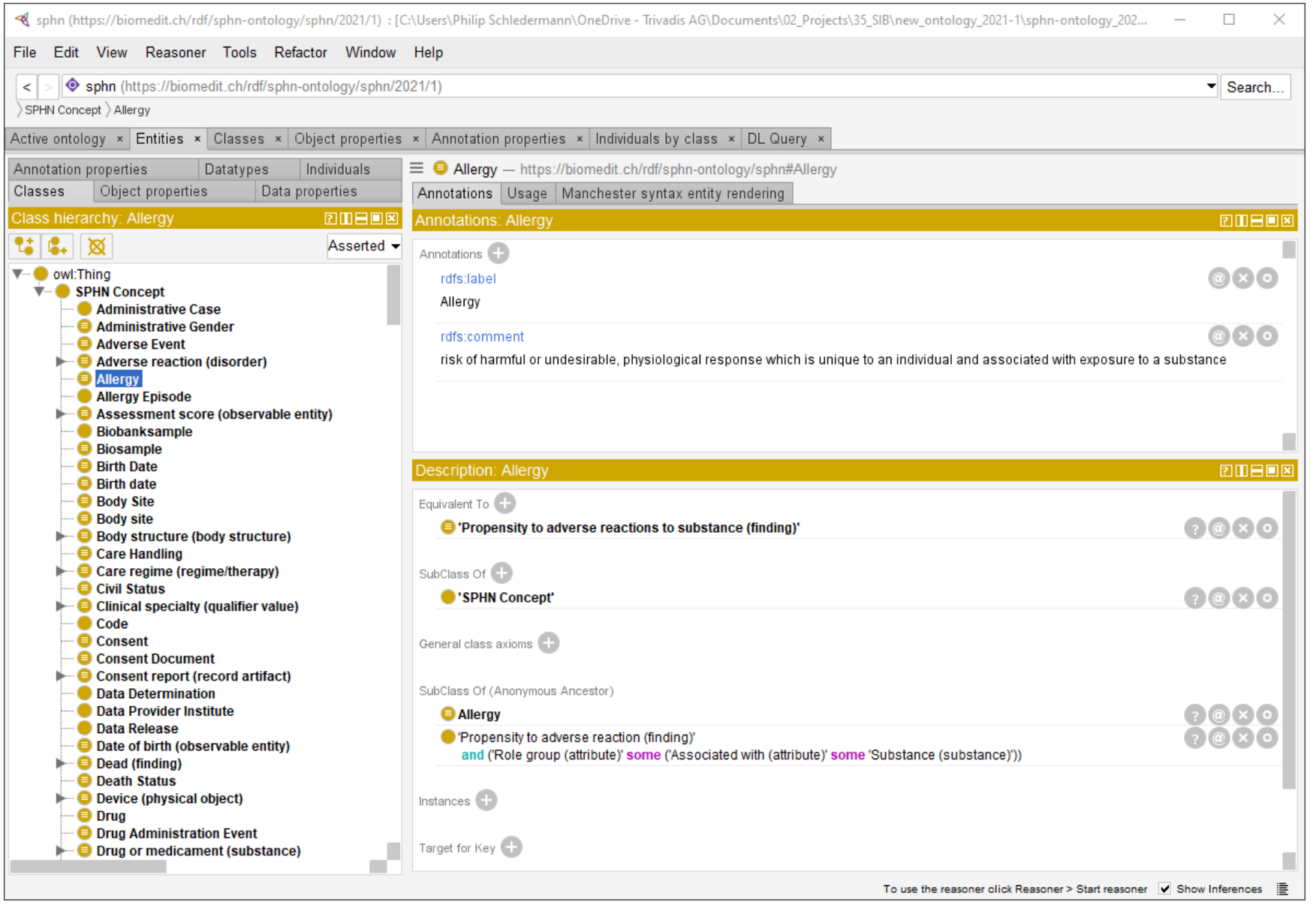
Descriptions + Rules how a thing is connected to another thing

Similar to a table definition but on graphs and

- Potentially describing a formal interpretation of a thing
- Not necessarily defining everything completely

For all the Relational Database people, the ontology can be compared to an ER (Entity Relationship) Model with some extensions.

Let’s show it by browsing the SPHN Ontology


sphn (https://biomedit.ch/rdf/sphn-ontology/sphn/2021/1) : [C:\Users\Philip Schledermann\OneDrive - Trivadis AG\Documents\02_Projects\35_SIB\new_ontology_2021-1\sphn-ontology_202...
File Edit View Reasoner Tools Refactor Window Help
sphn (https://biomedit.ch/rdf/sphn-ontology/sphn/2021/1)
Search...
SPHN Concept Allergy
Active ontology Entities Classes Object properties Annotation properties Individuals by class DL Query
Annotation properties Datatypes Individuals
Classes Object properties Data properties
Class hierarchy: Allergy
Asserted
owl:Thing
SPHN Concept
Administrative Case
Administrative Gender
Adverse Event
Adverse reaction (disorder)
Allergy
Allergy Episode
Assessment score (observable entity)
Biobanksample
Biosample
Birth Date
Birth date
Body Site
Body site
Body structure (body structure)
Care Handling
Care regime (regime/therapy)
Civil Status
Clinical specialty (qualifier value)
Code
Consent
Consent Document
Consent report (record artifact)
Data Determination
Data Provider Institute
Data Release
Date of birth (observable entity)
Dead (finding)
Death Status
Device (physical object)
Drug
Drug Administration Event
Drug or medicament (substance)
Allergy — https://biomedit.ch/rdf/sphn-ontology/sphn#Allergy
Annotations Usage Manchester syntax entity rendering
Annotations: Allergy
Annotations
rdfs:label
Allergy
rdfs:comment
risk of harmful or undesirable, physiological response which is unique to an individual and associated with exposure to a substance
Description: Allergy
Equivalent To
'Propensity to adverse reactions to substance (finding)'
SubClass Of
'SPHN Concept'
General class axioms
SubClass Of (Anonymous Ancestor)
Allergy
'Propensity to adverse reaction (finding)'
and ('Role group (attribute)' some ('Associated with (attribute)' some 'Substance (substance)'))
Instances
Target for Key
To use the reasoner click Reasoner > Start reasoner
Show Inferences

What we can learn from the Allergy definition

An Allergy

- is connected to a patient using the hasSubjectPseudoIdentifier relation
- is connected to a hospital using the hasDataProviderInstitute relation
- is connected to a substance using the hasSubstance relation
- Substances defined in SPHN are the same as substances as described SNOMED CT

This ontology does not say anything about

- Serialization/Datatypes
- Cardinalities

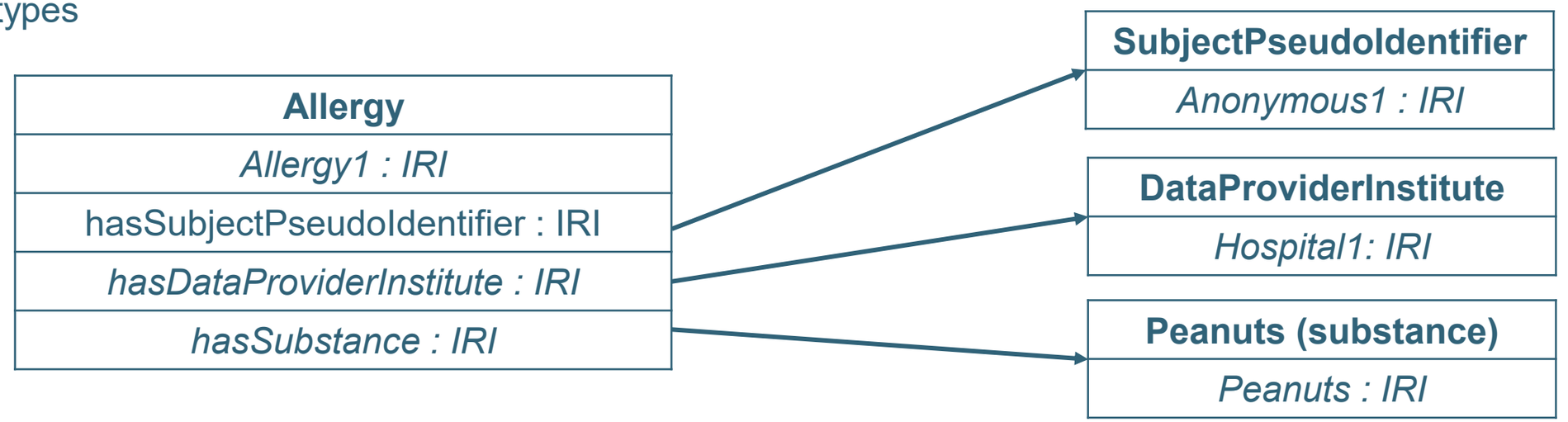
Especially for the Relational Database people

An Allergy

- is connected to a patient using the hasSubjectPseudoIdentifier relation
- is connected to a hospital using the hasDataProviderInstitute relation
- is connected to a substance using the hasSubstance relation
- Substances defined in SPHN are the same as substances as described SNOMED CT

This ontology does not say anything about

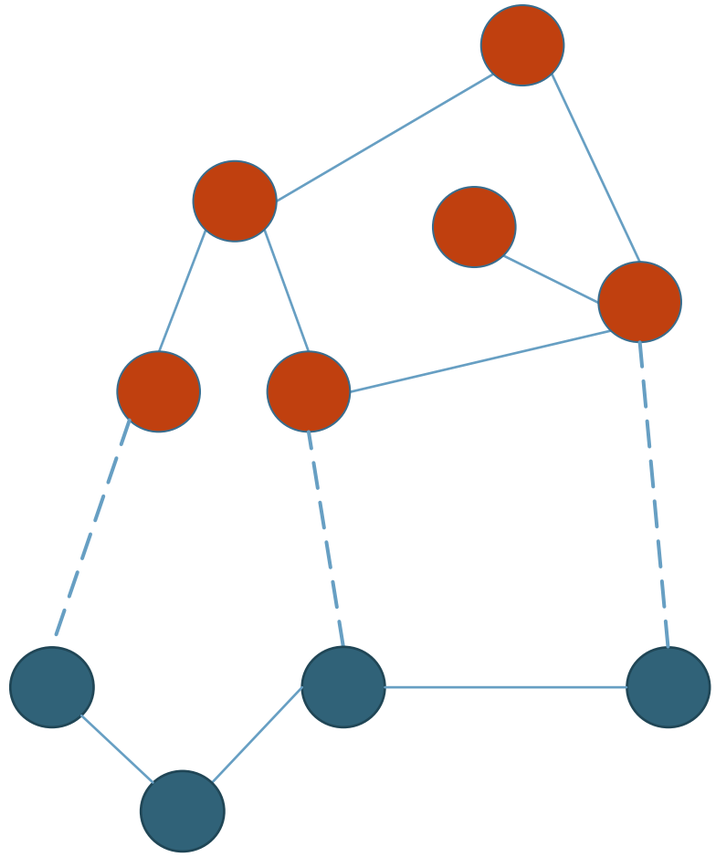
- Serialization/Datatypes
- Cardinalities



Ontology vs Instance data

Ontology
- describes how objects relates to each other
- formalizes the meaning
- stored in RDF or OWL
- SPHN
  - Provided by SPHN or adapted by projects
  - Also called “SPHN RDF schema”

Instance data
- instantiates the objects from the ontology
- Modeled data according to the ontology
- SPHN
  - Hospitals produce instance data
  - Stored in RDF and transferred to project environment



Spoiler: You will see in a few slides what the dashed lines mean…

RDF

Resource Description Framework

A W3C Standard

"How to represent and exchange Information"

Classical Tables to express information

Allergy	has substance
Allergy1	*Peanuts*

Peanuts (substance)
Peanuts

Allergy1 is an instance of the **Allergy**

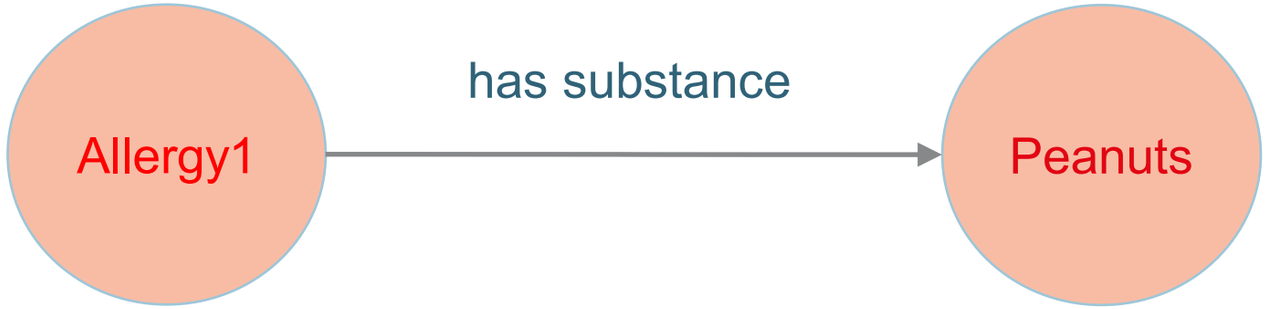
Allergy1 **has substance** *Peanuts*

Peanuts is an instance of the **Peanuts substance**

But we can also use Graphs

Allergy	has substance
Allergy1	*Peanuts*

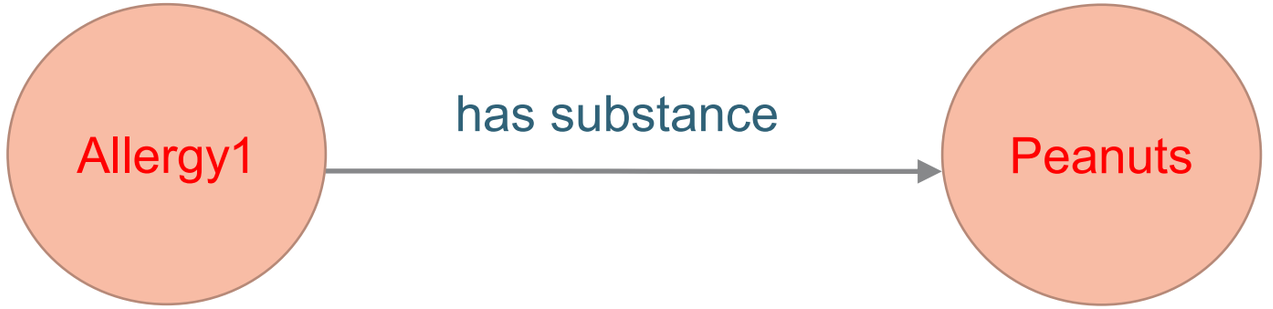
Peanuts (substance)
Peanuts



RDF Elements <-> How to build a Graph

IRIs	*<http://sib.swiss/allergies/Allergy1>*	IRIs (Internationalized Resource Identifiers) are a more generic form of URIs. They identify things like keys, and not need to be resolvable.
Literals	"Peanut"^^*<http://www.w3.org/2001/XMLSchema#string>*	Everything that is not a thing but a describing element. We use the XML Schema namespace for that.
Blank Nodes	_:alice	A key that is only valid inside the data. You can typically not query by a blank node itself.

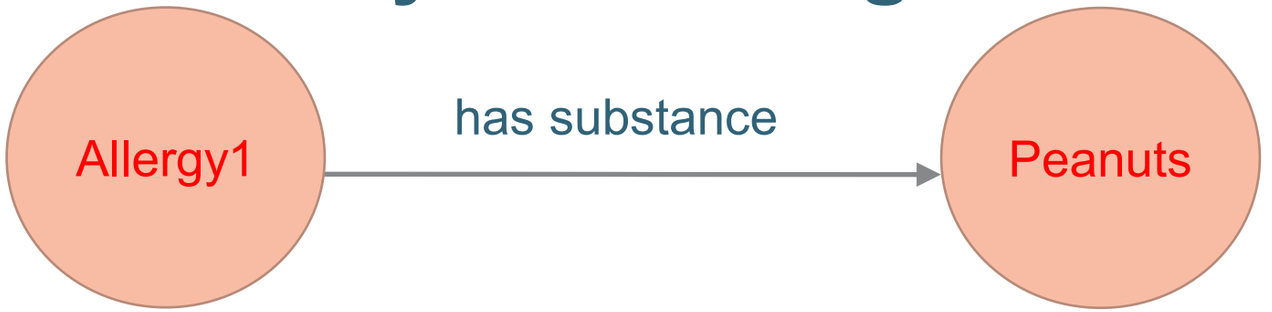
How do we combine them?



Subject	Predicate	Object
IRI or Blank Node	*IRI*	*IRI, Blank Node or Literal*
<http://sib.swiss/allergies/Allergy1>	<https://biomedit.ch/rdf/sphn-ontology/sphn#hasSubstance>	<http://sib.swiss/substances/Peanuts>

If you want to dig into the specification: https://www.w3.org/TR/rdf11-concepts/

Is there a nicer way?
Namespaces as Syntactic Sugar



≡

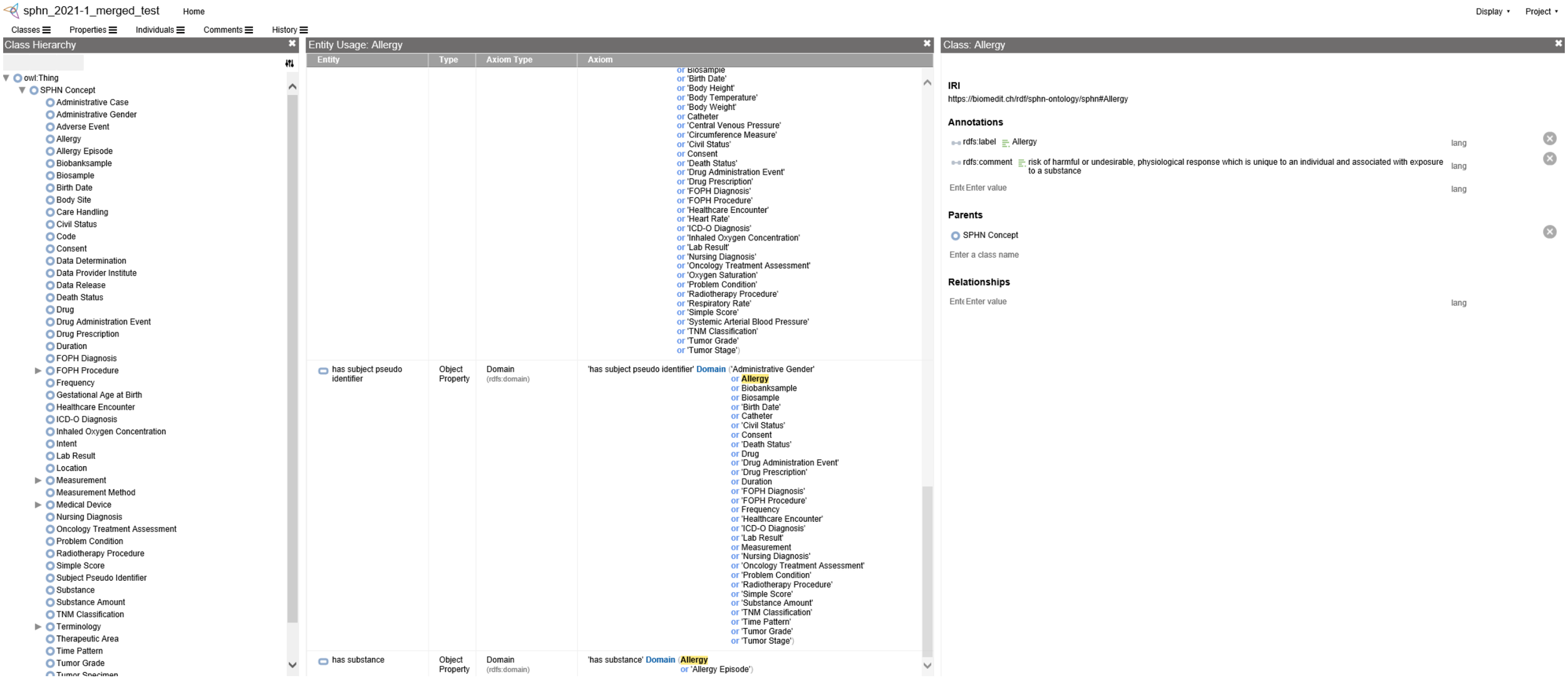
	Subject	Predicate	Object
	[prefixed] IRI or Blank Node	*[prefixed] IRI*	*[prefixed] IRI, Blank Node or Literal*
1	<http://sib.swiss/allergies/Allergy1>	<https://biomedit.ch/rdf/sphn-ontology/sphn#hasSubstance>	<http://sib.swiss/substances/Peanuts>
2	PREFIX **substances** : <http://sib.swiss/substances/>		
3	PREFIX **sphn**: <https://biomedit.ch/rdf/sphn-ontology/sphn#>		
4	<http://sib.swiss/allergies/Allergy1>	**sphn:**hasSubstance	**substances:**Peanuts

Just be aware not all serializations support it. Turtle (ttl) for example does.

Start creating some data

We want to model the following information:

- A patient (“anonymous1”) that came to our fictive hospital (“hospital1”)
- “anonymous1” has an allergy against peanuts
- Peanuts are a substance
- We want to declare that our peanut substance is an instance of SNOMED CT Code “762952008”


sphn_2021-1_merged_test
Home
Display
Project
Classes
Properties
Individuals
Comments
History
Class Hierarchy
owl:Thing
SPHN Concept
Administrative Case
Administrative Gender
Adverse Event
Allergy
Allergy Episode
Biobanksample
Biosample
Birth Date
Body Site
Care Handling
Civil Status
Code
Consent
Data Determination
Data Provider Institute
Data Release
Death Status
Drug
Drug Administration Event
Drug Prescription
Duration
FOPH Diagnosis
FOPH Procedure
Frequency
Gestational Age at Birth
Healthcare Encounter
ICD-O Diagnosis
Inhaled Oxygen Concentration
Intent
Lab Result
Location
Measurement
Measurement Method
Medical Device
Nursing Diagnosis
Oncology Treatment Assessment
Problem Condition
Radiotherapy Procedure
Simple Score
Subject Pseudo Identifier
Substance
Substance Amount
TNM Classification
Terminology
Therapeutic Area
Time Pattern
Tumor Grade
Tumor Specimen
Entity Usage: Allergy
Entity
Type
Axiom Type
Axiom
has subject pseudo identifier
Object Property
Domain (rdfs:domain)
'has subject pseudo identifier' Domain ('Administrative Gender' or Allergy or Biobanksample or Biosample or 'Birth Date' or Catheter or 'Civil Status' or Consent or 'Death Status' or Drug or 'Drug Administration Event' or 'Drug Prescription' or Duration or 'FOPH Diagnosis' or 'FOPH Procedure' or Frequency or 'Healthcare Encounter' or 'ICD-O Diagnosis' or 'Lab Result' or Measurement or 'Nursing Diagnosis' or 'Oncology Treatment Assessment' or 'Problem Condition' or 'Radiotherapy Procedure' or 'Simple Score' or 'Substance Amount' or 'TNM Classification' or 'Time Pattern' or 'Tumor Grade' or 'Tumor Stage')
has substance
Object Property
Domain (rdfs:domain)
'has substance' Domain (Allergy or 'Allergy Episode')
Class: Allergy
IRI
https://biomedit.ch/rdf/sphn-ontology/sphn#Allergy
Annotations
rdfs:label Allergy lang
rdfs:comment risk of harmful or undesirable, physiological response which is unique to an individual and associated with exposure to a substance lang
Enter value lang
Parents
SPHN Concept
Enter a class name
Relationships
Enter value lang

Have a look at the ontology first

The Allergy seems to be the central point

- connects to the Person via the hasSubjectPseudoIdentifier
- connects to the Hospital via the hasDataProviderInstitute
- connects to the Substance via the hasSubstance

Create IRIs for the instances

	Name	IRI
Patient	anonymous1	http://sib.swiss/fictivePatients/**anonymous1**
Hospital	hospital1	http://sib.swiss/**hospital1**
Allergy	allergy1	http://sib.swiss/allergies/**allergy1**
Peanuts	peanuts1	http://sib.swiss/substances/**peanuts1**

anonymous1

hospital1

allergy1

peanuts1

What are we missing?

Connecting elements! -> Predicates

Name	IRI
hasSubstance	https://biomedit.ch/rdf/sphn-ontology/sphn#**hasSubstance**
hasSubjectPseudoIdentifier	https://biomedit.ch/rdf/sphn-ontology/sphn#**hasSubjectPseudoIdentifier**
hasDataProviderInsitute	https://biomedit.ch/rdf/sphn-ontology/sphn#**hasDataProviderInstitute**

hasSubstance →

hasSubjectPseudoIdentifier →

hasDataProviderInstitute →

We are still missing something…

Semantic types:
Instantiating Things of the Ontology

short form	IRI
rdf:type	http://www.w3.org/1999/02/22-rdf-syntax-ns#**type**

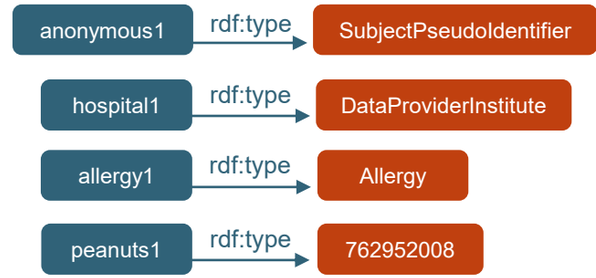
rdf:type →

That's the dashed lines you saw earlier!

Putting it all together

Subjects or Objects:

Name	IRI	Type
anonymous1	http://sib.swiss/fictivePatients/**anonymous1**	sphn:**SubjectPseudoIdentifier**
hospital1	http://sib.swiss/**hospital1**	sphn:**DataProviderInstitute**
peanuts1	http://sib.swiss/allergies/**allergy1**	sphn**:Allergy**
peanuts1	http://sib.swiss/substances/**peanuts1**	snomed:**762952008**



Predicates:

Name	IRI
hasSubstance	sphn:**hasSubstance**
hasSubjectPseudoIdentifier	sphn:**hasSubjectPseudoIdentifier**
hasDataProviderInstitute	sphn:**hasDataProviderInstitute**
type	rdf:**type**

hasSubstance →

hasSubjectPseudoIdentifier →

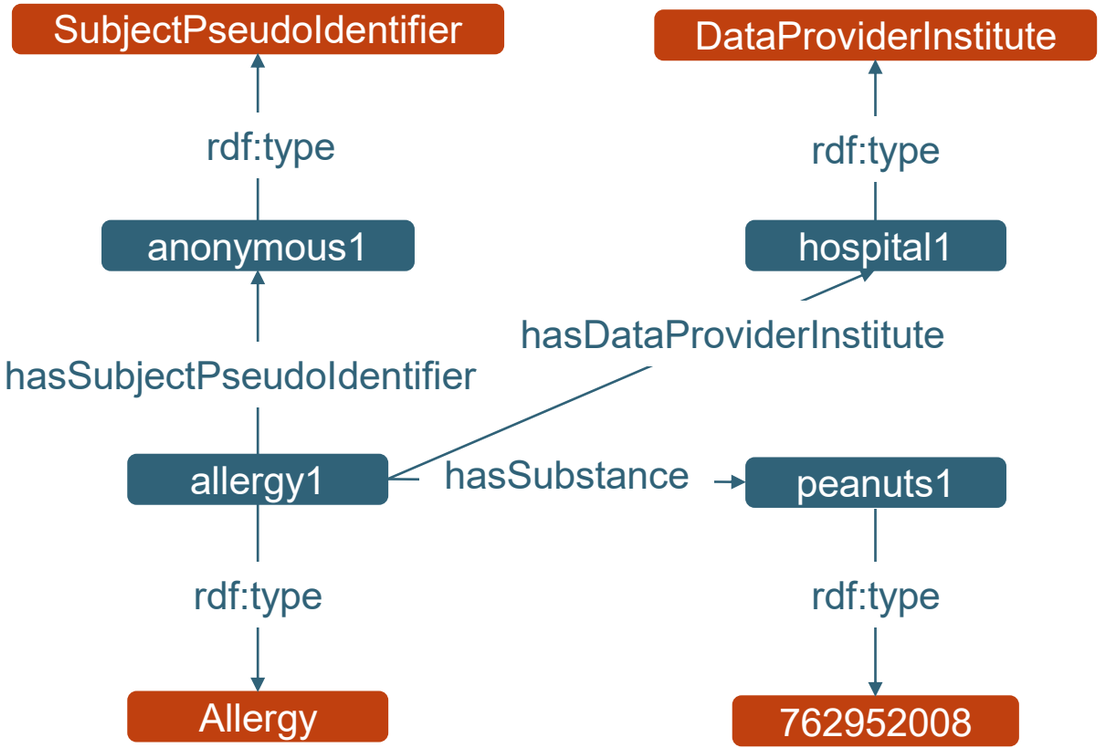
hasDataProviderInstitute →

rdf:type →

sphn, snomed, rdf Prefixes are omitted on this page

A valid Turtle file

```
@prefix allergies: <http://sib.swiss/allergies/> .
@prefix patients: <http://sib.swiss/fictivePatients/> .
@prefix substances: <http://sib.swiss/substances/> .
@prefix sib: <http://sib.swiss/> .
@prefix sphn: <https://biomedit.ch/rdf/sphn-ontology/sphn#> .
@prefix snomed: <http://snomed.info/id/> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .

# types
patients:anonymous1 rdf:type sphn:SubjectPseudoIdentifier .
sib:hospital1 rdf:type sphn:DataProviderInstitute .
allergies:allergy1 rdf:type sphn:Allergy .
substances:peanuts1 rdf:type snomed:762952008 .

# relations to the allergy
allergies:allergy1 sphn:hasSubjectPseudoIdentifier patients:anonymous1 .
allergies:allergy1 sphn:hasDataProviderInstitute sib:hospital1 .
allergies:allergy1 sphn:hasSubstance substances:peanuts1 .
```



Loading into a DB

DBs for storing RDF are triplestores

Examples of Triplestores

Stardog

ORACLE

GraphDB

AllegroGraph
Franz Inc.

rdf4j

Jena

OPENLINK VIRTUOSO

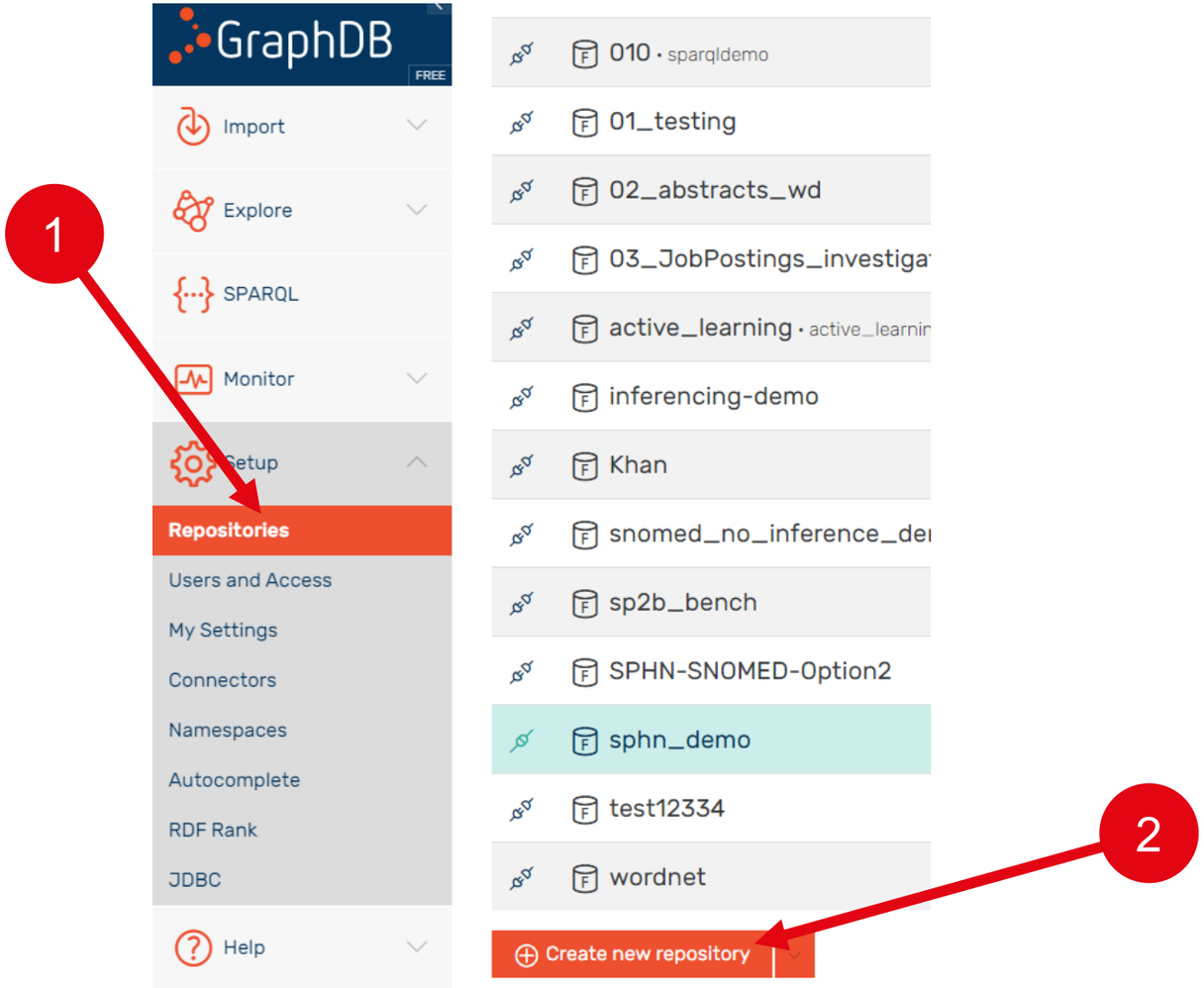
MarkLogic

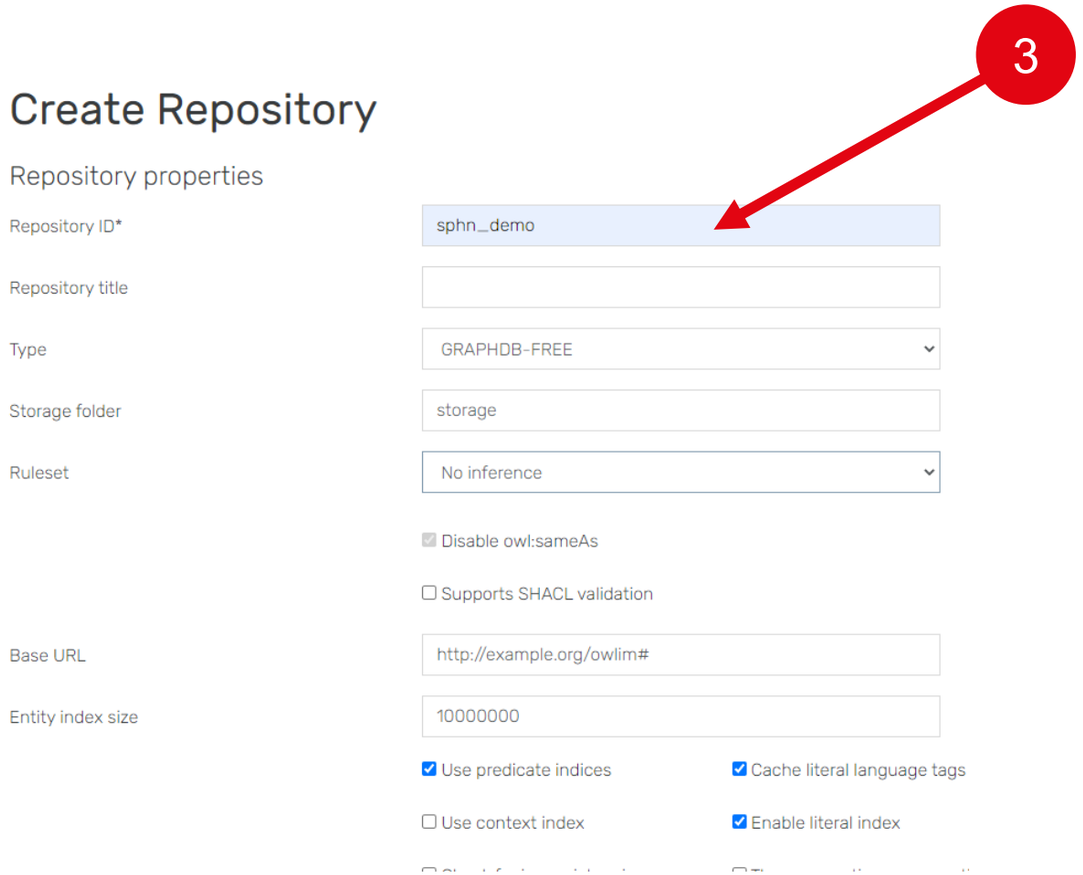
aws Amazon Neptune

RDFOX

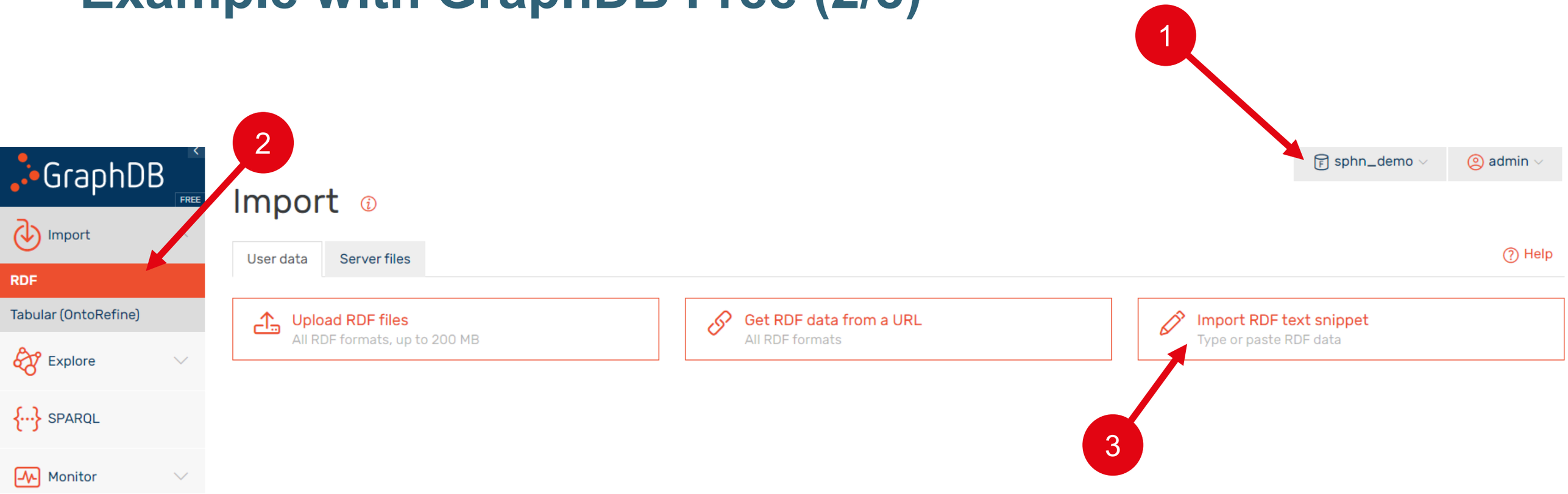
+ some more

Example with GraphDB Free (1/8)



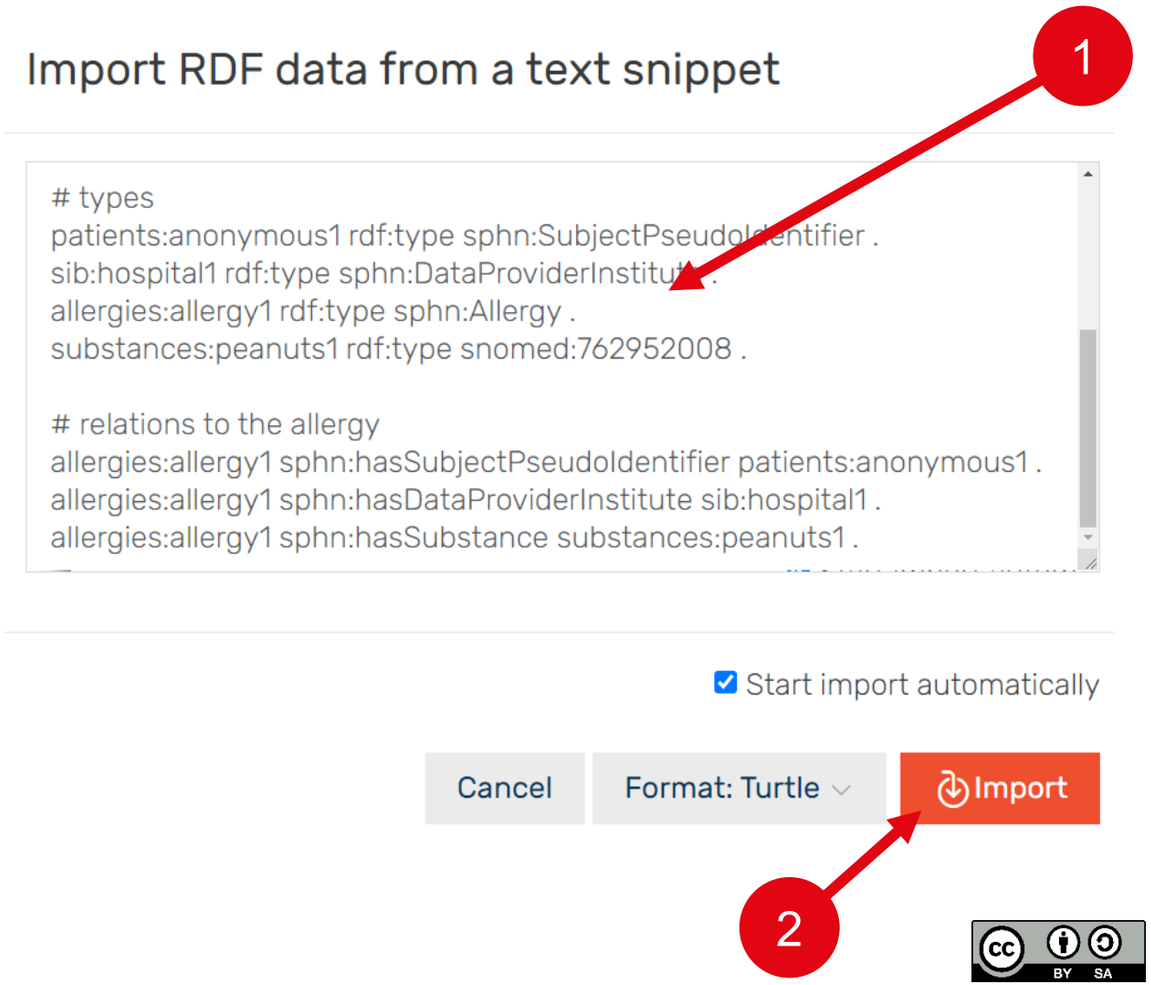


Example with GraphDB Free (2/8)

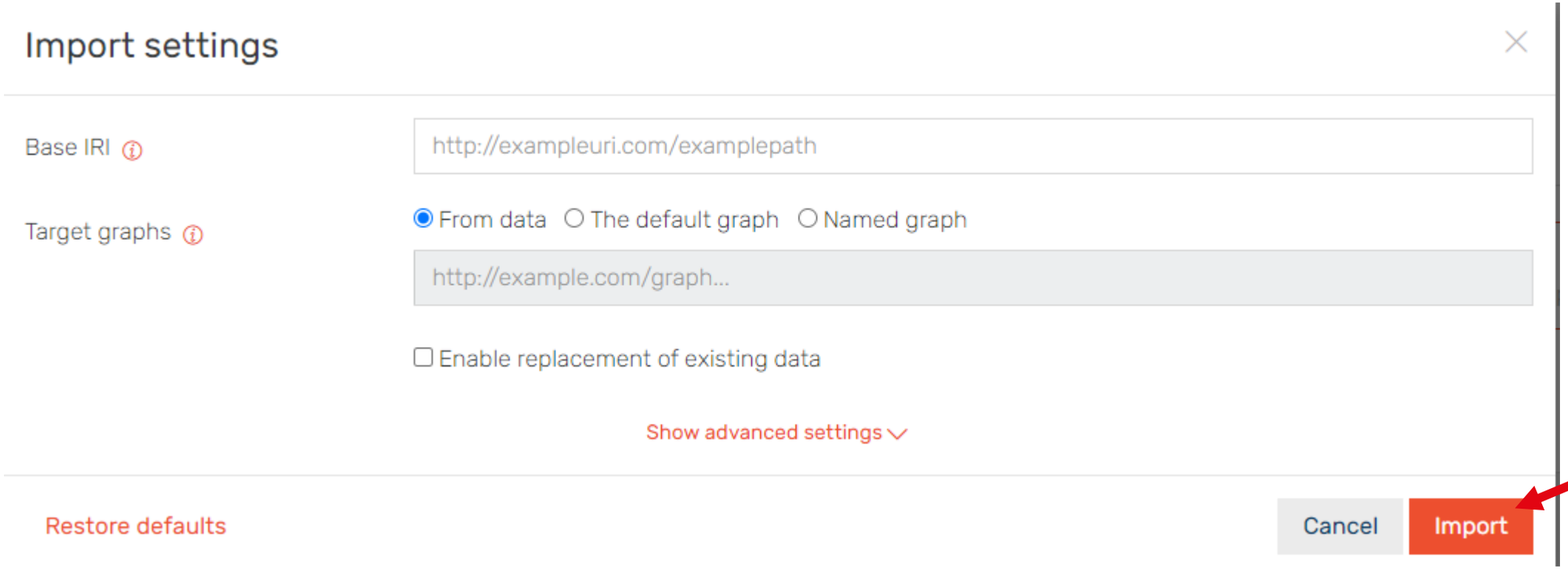


Example with GraphDB Free (3/8)

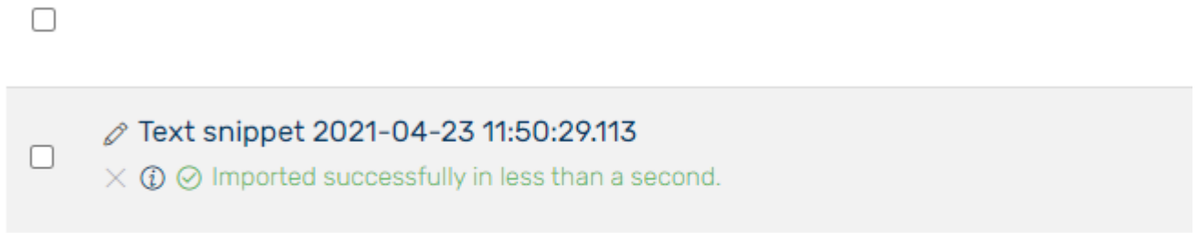
```
@prefix allergies: <http://sib.swiss/allergies/> .
@prefix patients: <http://sib.swiss/fictivePatients/> .
@prefix substances: <http://sib.swiss/substances/> .
@prefix sib: <http://sib.swiss/> .
@prefix sphn: <https://biomedit.ch/rdf/sphn-ontology/sphn#> .
@prefix snomed: <http://snomed.info/id/> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .

# types
patients:anonymous1 rdf:type sphn:SubjectPseudoIdentifier .
sib:hospital1 rdf:type sphn:DataProviderInstitute .
allergies:allergy1 rdf:type sphn:Allergy .
substances:peanuts1 rdf:type snomed:762952008 .

# relations to the allergy
allergies:allergy1 sphn:hasSubjectPseudoIdentifier patients:anonymous1 .
allergies:allergy1 sphn:hasDataProviderInstitute sib:hospital1 .
allergies:allergy1 sphn:hasSubstance substances:peanuts1 .
```



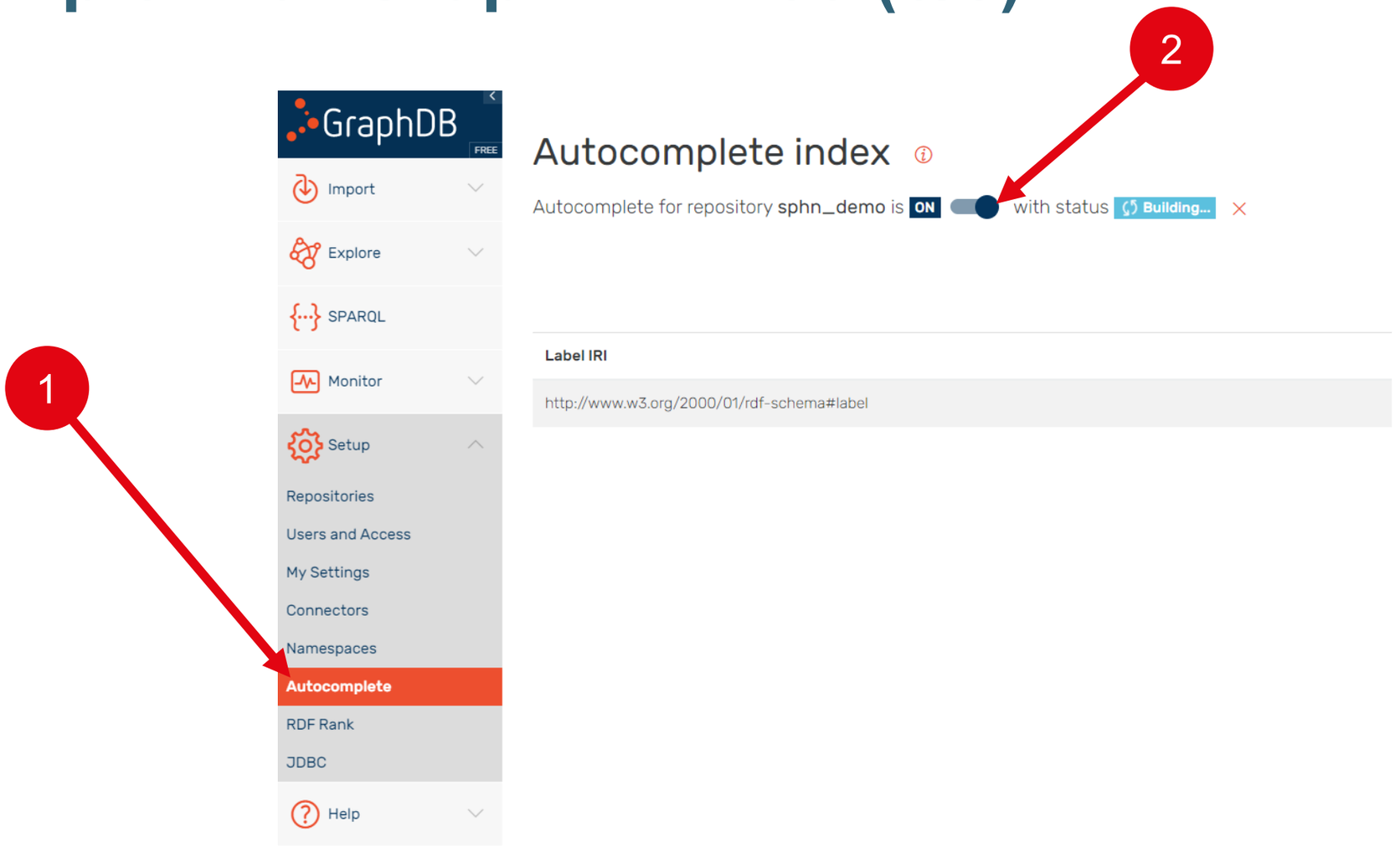
Example with GraphDB Free (4/8)



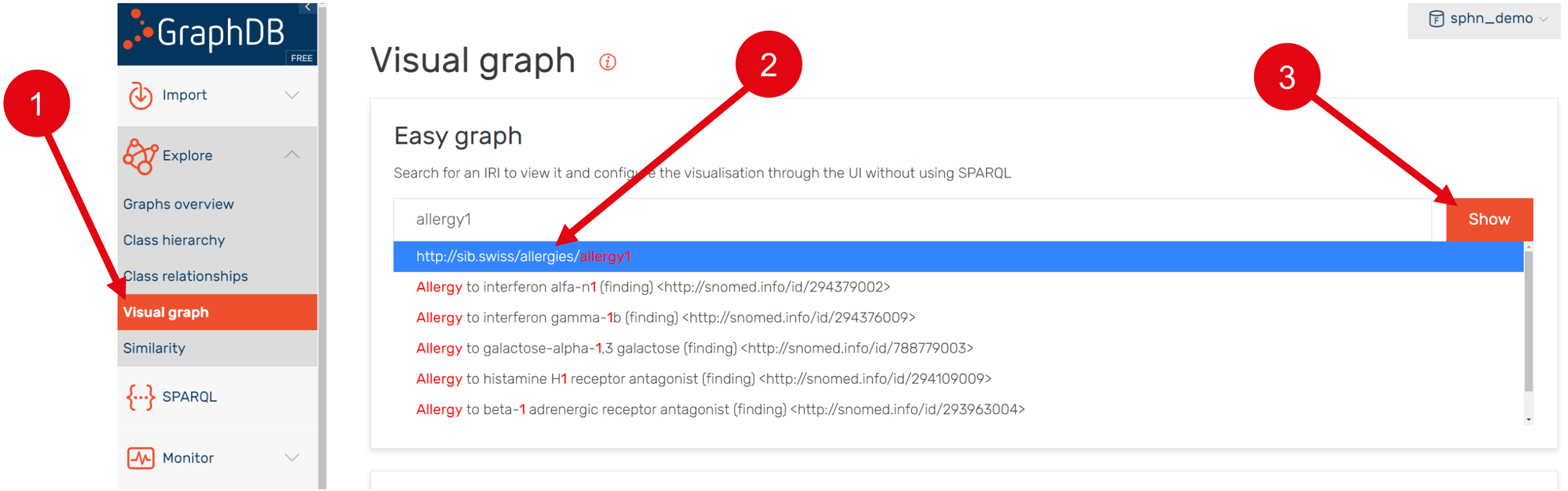
Example with GraphDB Free (5/8)



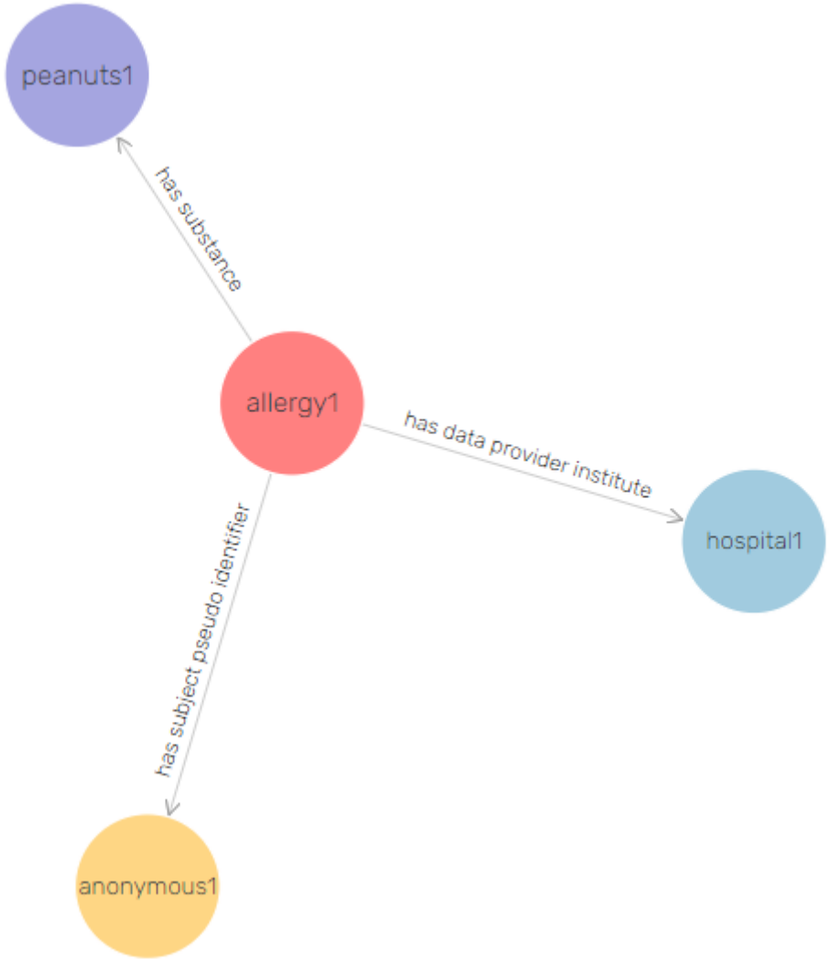
Example with GraphDB Free (6/8)



Example with GraphDB Free (7/8)



Example with GraphDB Free (8/8)



Some Information

I loaded the ontology beforehand ;-) You should do that as well in your projects so you will be able to get all the information about types etc!

We have shown the easiest way for the demo to get data inside a nearly fresh GraphDB instance here. This is mainly for testing. There exist other options.

Have a look at the documentation of your database e.g. https://graphdb.ontotext.com/documentation/free/index.html

GraphDB Free is a Free of use for personal usage. It is not open-source. You can apply for a license using https://www.ontotext.com/products/graphdb/graphdb-free/

SPARQL

SPARQL Protocol and RDF Query Language
A W3C Standard

“How to query Information”

The SPARQL Syntax

```
prefix dc: <...>
prefix uni: <...>
```
← Prefix declarations

```
SELECT ...
```
← Declare type of query: `SELECT`, `ASK`, `DESCRIBE`, `CONSTRUCT`

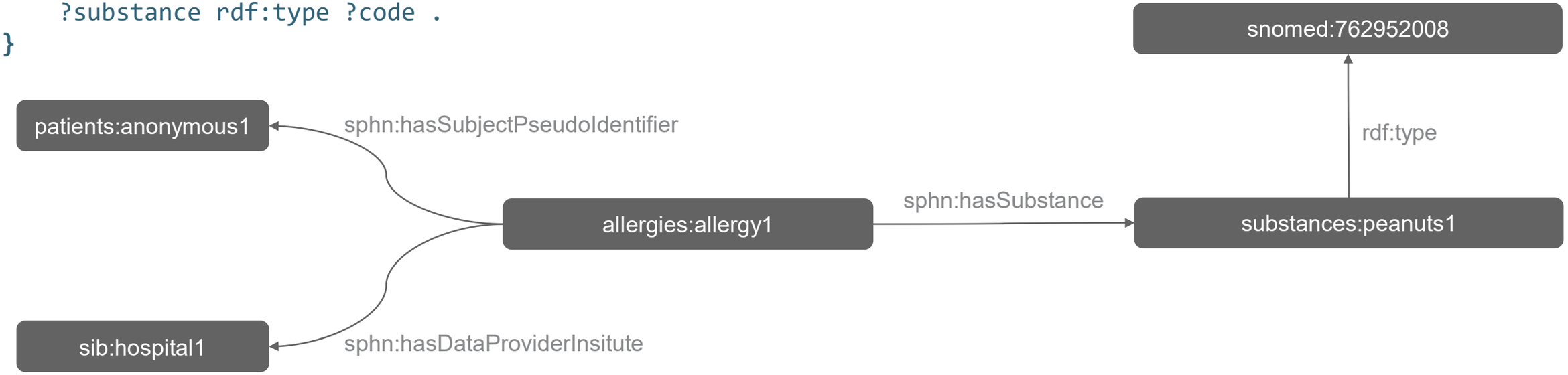
```
FROM <...>
FROM NAMED <...>
```
← Define dataset

```
WHERE { ... }
```
← Graph pattern

```
ORDER BY ...
HAVING ...
GROUP BY ...
LIMIT ...
OFFSET ...
BINDINGS ...
```
← Query modifiers

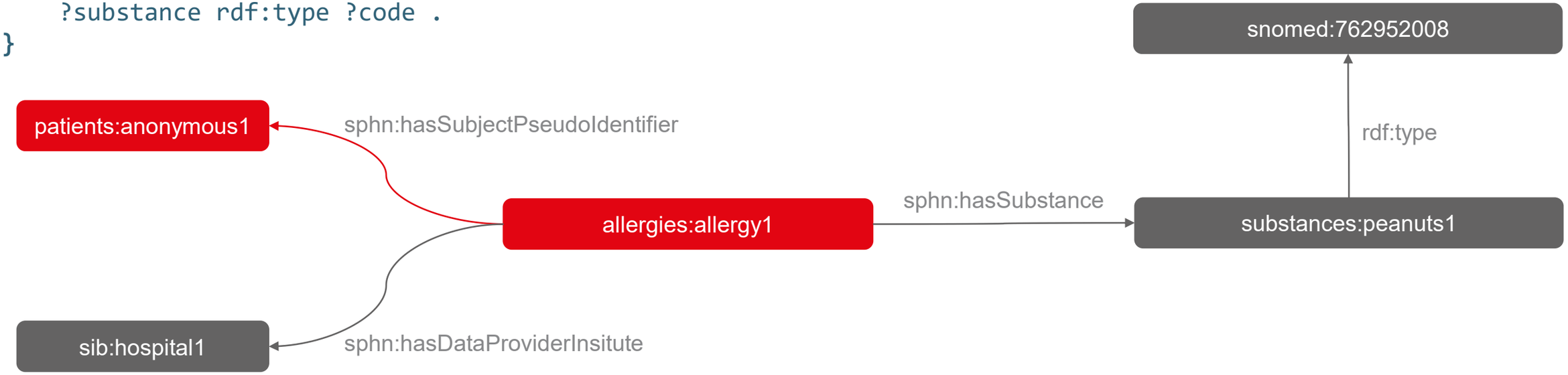
Example Query Evaluation (1/5)

```
PREFIX patients: <http://sib.swiss/fictivePatients/>
PREFIX sphn: <https://biomedit.ch/rdf/sphn-ontology/sphn#>
PREFIX sib: <http://sib.swiss/>

SELECT ?code
WHERE {
    ?allergy   sphn:hasSubjectPseudoIdentifier patients:anonymous1 .
    ?allergy   sphn:hasSubstance ?substance .
    ?substance rdf:type ?code .
}
```



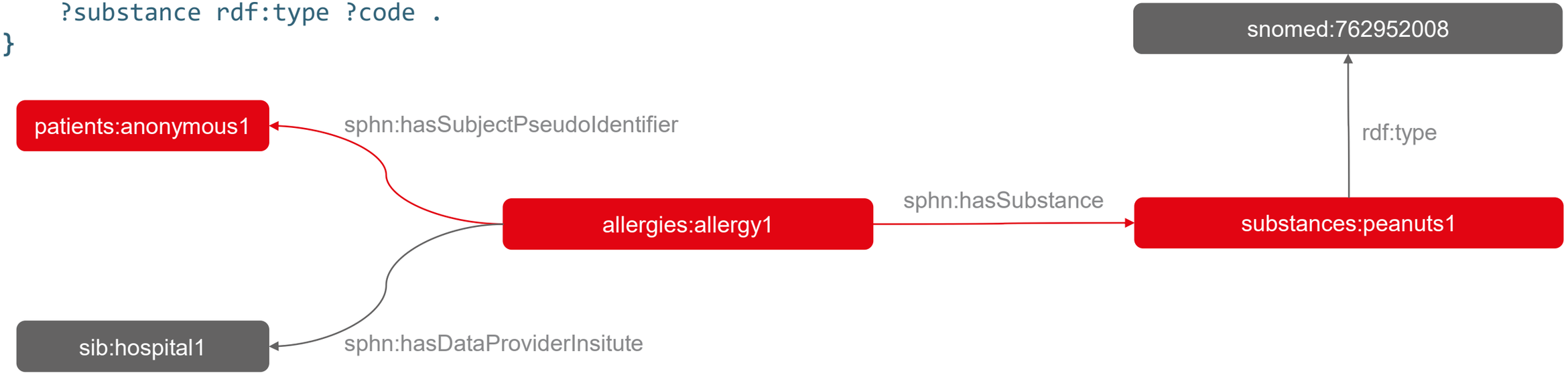
Example Query Evaluation (2/5)

```
PREFIX patients: <http://sib.swiss/fictivePatients/>
PREFIX sphn: <https://biomedit.ch/rdf/sphn-ontology/sphn#>
PREFIX sib: <http://sib.swiss/>

SELECT ?code
WHERE {
    ?allergy   sphn:hasSubjectPseudoIdentifier patients:anonymous1 .
    ?allergy   sphn:hasSubstance ?substance .
    ?substance rdf:type ?code .
}
```



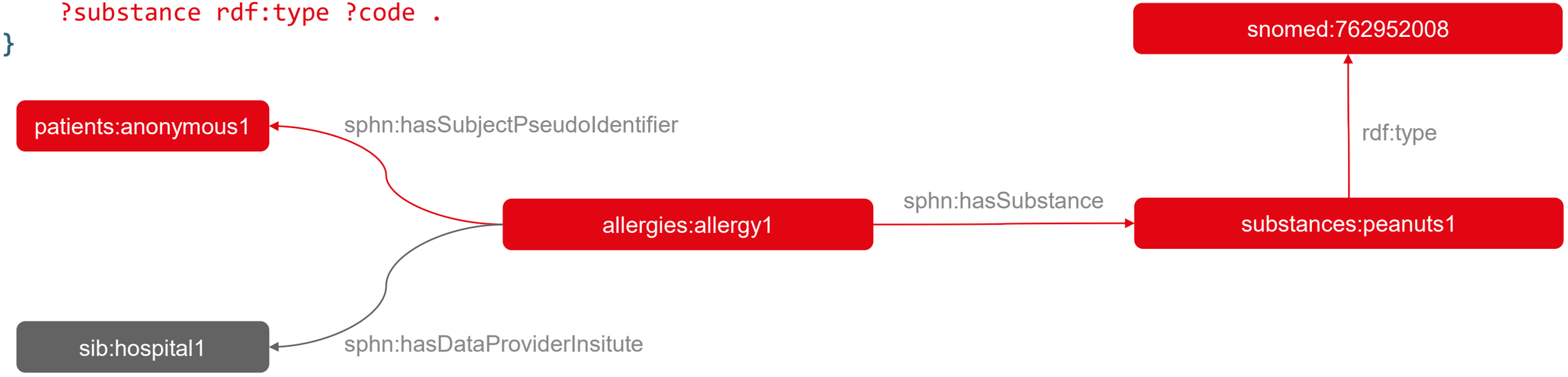
Example Query Evaluation (3/5)

```
PREFIX patients: <http://sib.swiss/fictivePatients/>
PREFIX sphn: <https://biomedit.ch/rdf/sphn-ontology/sphn#>
PREFIX sib: <http://sib.swiss/>

SELECT ?code
WHERE {
    ?allergy   sphn:hasSubjectPseudoIdentifier patients:anonymous1 .
    ?allergy   sphn:hasSubstance ?substance .
    ?substance rdf:type ?code .
}
```



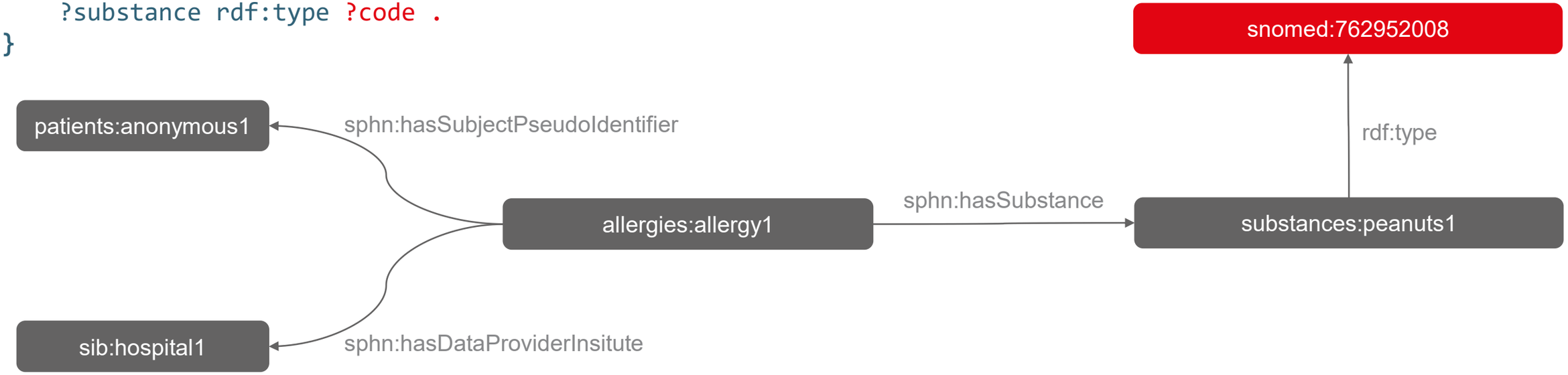
Example Query Evaluation (3/5)

```
PREFIX patients: <http://sib.swiss/fictivePatients/>
PREFIX sphn: <https://biomedit.ch/rdf/sphn-ontology/sphn#>
PREFIX sib: <http://sib.swiss/>

SELECT ?code
WHERE {
    ?allergy   sphn:hasSubjectPseudoIdentifier patients:anonymous1 .
    ?allergy   sphn:hasSubstance ?substance .
    ?substance rdf:type ?code .
}
```

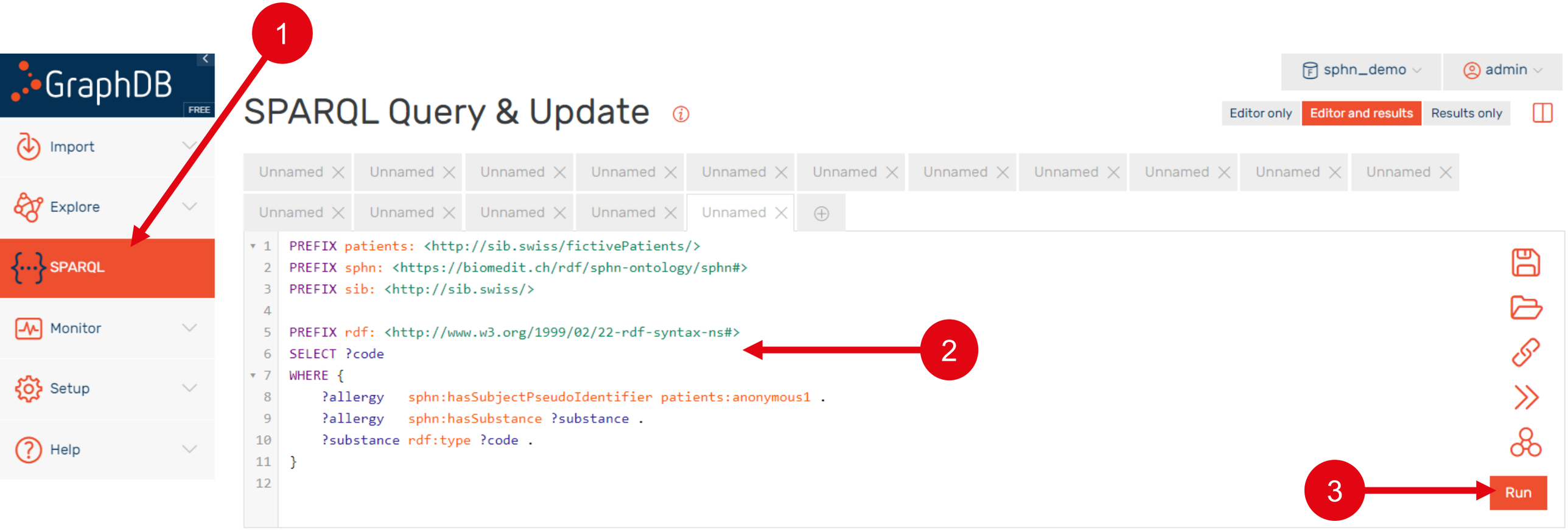


Example Query Evaluation (3/5)

```
PREFIX patients: <http://sib.swiss/fictivePatients/>
PREFIX sphn: <https://biomedit.ch/rdf/sphn-ontology/sphn#>
PREFIX sib: <http://sib.swiss/>

SELECT ?code
WHERE {
    ?allergy   sphn:hasSubjectPseudoIdentifier patients:anonymous1 .
    ?allergy   sphn:hasSubstance ?substance .
    ?substance rdf:type ?code .
}
```



Let's do it live (1/2)



Let’s do it live (2/2)

	code
1	snomed:762952008

Test

What do we need to change in the previous query to get the IRI of the Hospital?

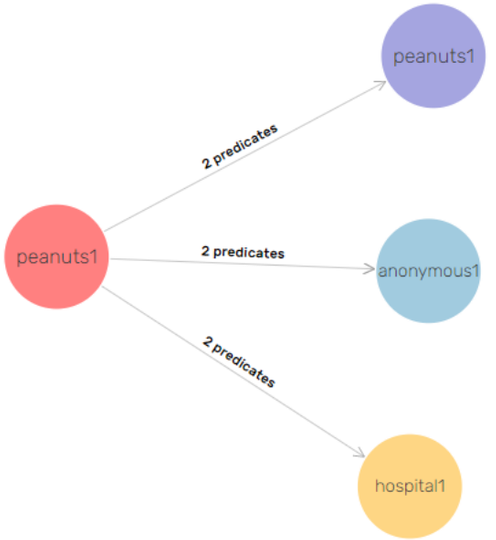
```
PREFIX patients: <http://sib.swiss/fictivePatients/>
PREFIX sphn: <https://biomedit.ch/rdf/sphn-ontology/sphn#>
PREFIX sib: <http://sib.swiss/>

SELECT ?code
WHERE {
    ?allergy   sphn:hasSubjectPseudoIdentifier patients:anonymous1 .
    ?allergy   sphn:hasSubstance ?substance .
    ?substance rdf:type ?code .
}
```

What do we need to change in the previous query to get the IRI of the Hospital?

```
PREFIX patients: <http://sib.swiss/fictivePatients/>
PREFIX sphn: <https://biomedit.ch/rdf/sphn-ontology/sphn#>
PREFIX sib: <http://sib.swiss/>

SELECT ?code
WHERE {
    ?allergy   sphn:hasSubjectPseudoIdentifier patients:anonymous1 .
    ?allergy   sphn:hasSubstance ?substance .
    ?substance rdf:type ?code .
}
```

Possible relations:

rdf:type
sphn:hasSubjectPseudoIdentifier
sphn:hasDataProviderInstitute
sphn:hasSubstance



Slide to edit live in the session

This one

```
PREFIX patients: <http://sib.swiss/fictivePatients/>
PREFIX sphn: <https://biomedit.ch/rdf/sphn-ontology/sphn#>
PREFIX sib: <http://sib.swiss/>

SELECT ?hospital
WHERE {
    ?allergy   sphn:hasSubjectPseudoIdentifier patients:anonymous1 .
    ?allergy  sphn:hasDataProviderInstitute ?hospital .
}
```

```
PREFIX patients: <http://sib.swiss/fictivePatients/>
PREFIX sphn: <https://biomedit.ch/rdf/sphn-ontology/sphn#>
PREFIX sib: <http://sib.swiss/>

SELECT ?hospital
WHERE {
    ?allergy   sphn:hasSubjectPseudoIdentifier patients:anonymous1 .
    ?allergy  sphn:hasDataProviderInstitute ?hospital .
}
```

Run

Table | Raw Response | Pivot Table | Google Chart

Download as

Filter query results

Showing results from 1 to 1 of 1. Query took 0.1s, moments ago.

	hospital
1	sib:hospital1

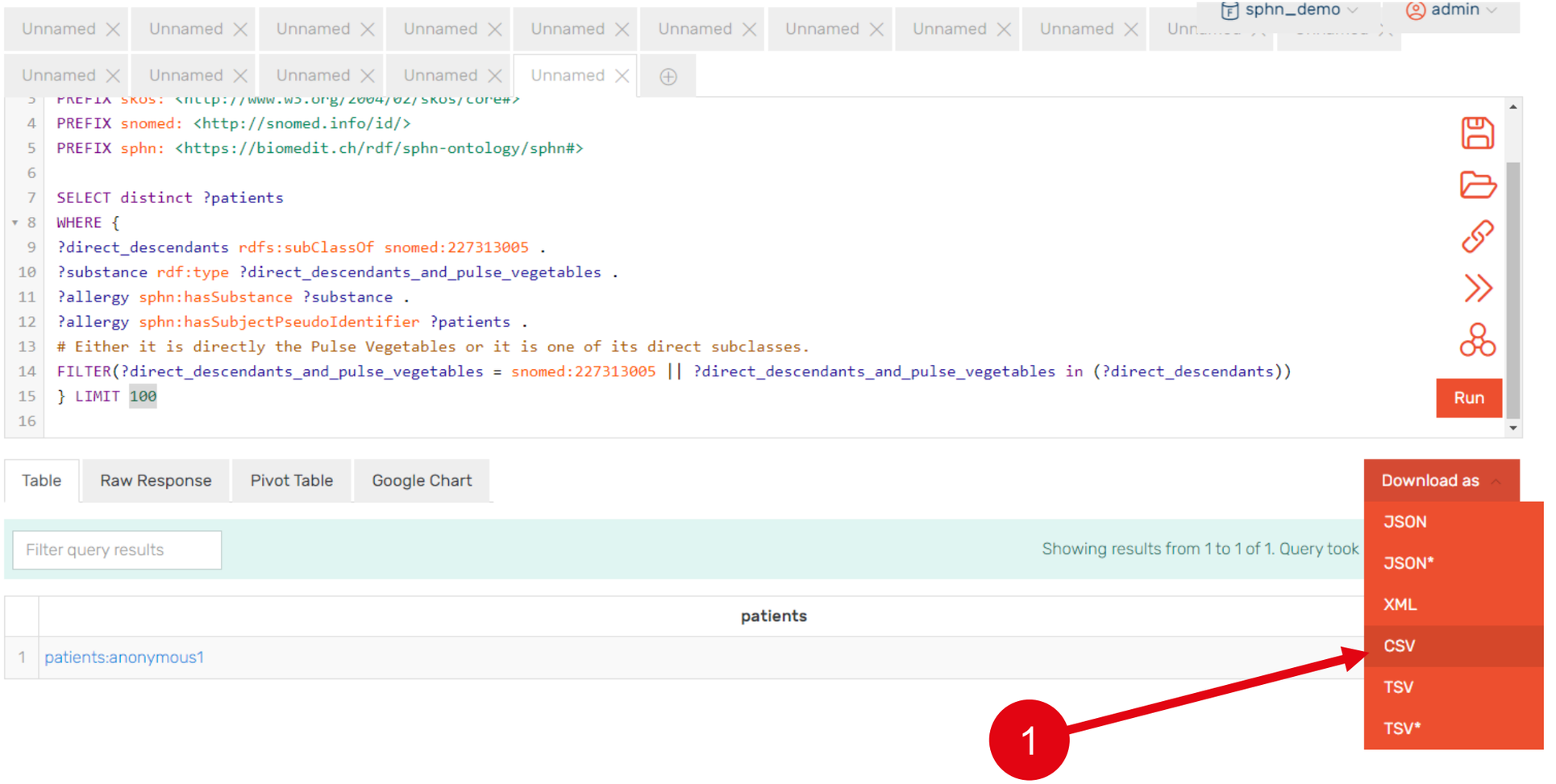
Let’s do something fancy:

Give me all patients that have an allergy against something in family of Pulse vegetables

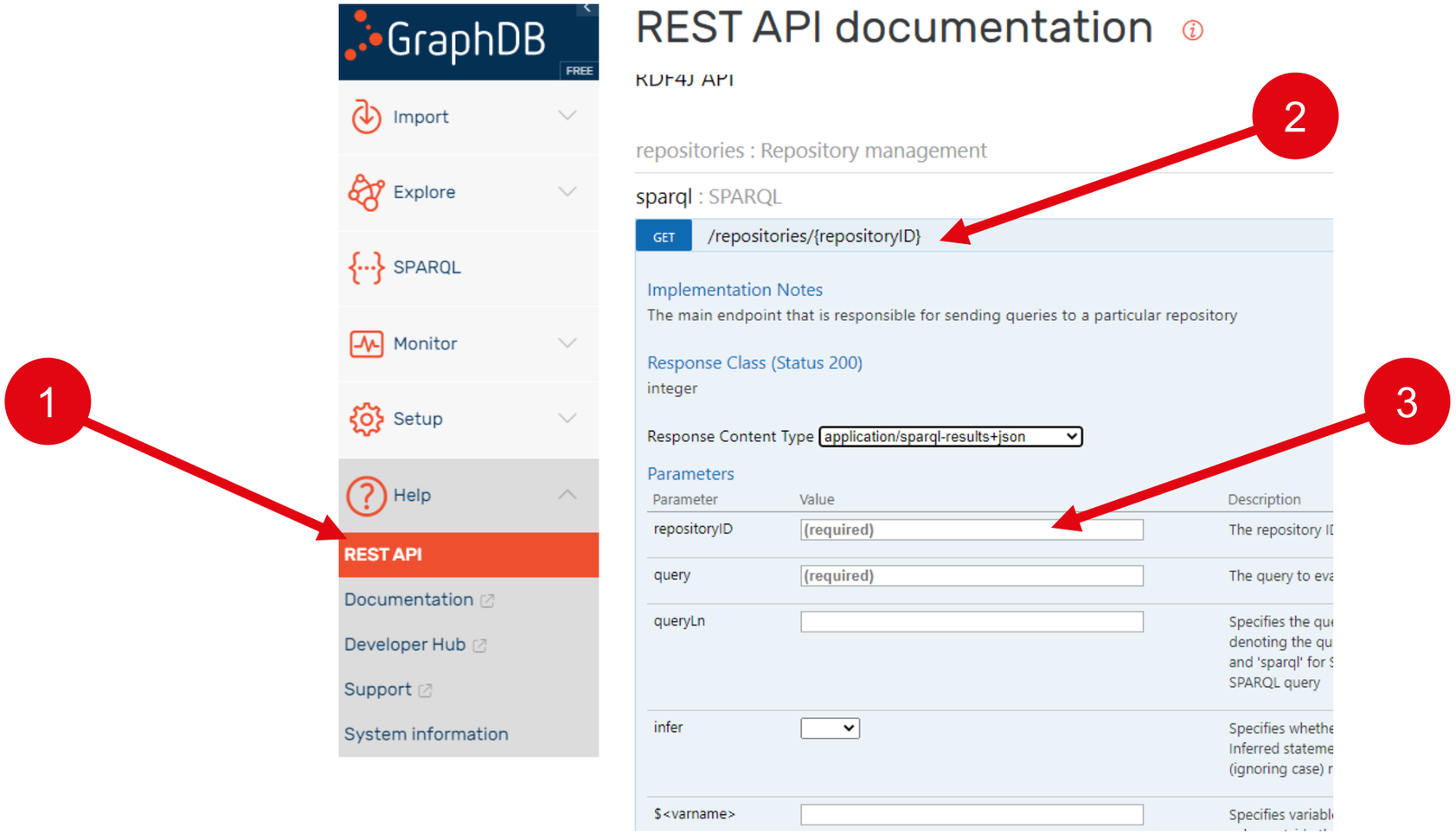
```
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX skos: <http://www.w3.org/2004/02/skos/core#>
PREFIX snomed: <http://snomed.info/id/>
PREFIX sphn: <https://biomedit.ch/rdf/sphn-ontology/sphn#>

SELECT ?patients
WHERE {
     ?direct_descendants rdfs:subClassOf snomed:227313005 .
     ?substance rdf:type ?direct_descendants .
     ?allergy sphn:hasSubstance ?substance .
     ?allergy sphn:hasSubjectPseudoIdentifier ?patients .
}
```

How do you download as flat data?

For “short” queries



For “long” queries



Further Information

W3C References

- SPARQL Reference : https://www.w3.org/TR/sparql11-query/
- RDF Primer : https://www.w3.org/TR/rdf11-primer/
- RDF Concepts & Abstract Syntax : https://www.w3.org/TR/rdf11-concepts/

Books

- Bob DuCharme : **Learning SPARQL : Querying and Updating with SPARQL 1.1** : ISBN: 978-1449371432

  > Getting your Hands dirty with SPARQL and RDF
- Andreas Blumenauer, Helmut Nagy (SWC) : **The Knowledge Graph Cookbook** : ISBN: 978-3-902796-70-7

  > More modern and business focused
- Jeffrey T. Pollok : **Semantic Web for Dummies** : ISBN: 978-0-470-39679-7

  > "old" but covers all necessary concepts needed, quite good beginner book

Internet Pages or Papers

- SPARQL 1.1. Cheat Sheet : http://www.iro.umontreal.ca/~lapalme/ift6281/sparql-1_1-cheat-sheet.pdf
- Knowledge Graphs : https://arxiv.org/pdf/2003.02320.pdf

Thank you

We’d value feedback on your training experience:

https://www.sib.swiss/form/view.php?id=89324