



KDI • Knowledge and Data Integration

GraphDB

W10.L19.M6.T19.2.2

Contents

1 Graph Databases

2 Demo

Graph Databases

- In computing, a graph database (GDB) is a database that uses graph structures for semantic queries with nodes, edges, and properties to represent and store data.
- A key concept of the system is the graph (or edge or relationship). The graph relates the data items in the store to a collection of nodes and edges, the edges representing the relationships between the nodes.
- The relationships allow data in the store to be linked together directly and, in many cases, retrieved with one operation.
- Querying relationships is fast because they are perpetually stored in the database. Relationships can be intuitively visualized using graph databases, making them useful for heavily inter-connected data.

Graph Databases [Contd.]

- Graph databases are a type of NoSQL database, created to address the limitations of relational databases.
- While the graph model explicitly lays out the dependencies between nodes of data, the relational model and other NoSQL database models link the data by implicit connections.
- In other words, relationships are a first-class citizen in a graph database and can be labelled, directed, and given properties. This is compared to relational approaches where these relationships are implied and must be reified at run-time.
- Retrieving data from a graph database requires a (graph pattern-matching based) query language other than SQL (which was designed for the manipulation of data in a relational system). There are many such graph query languages.

Graph Databases [Contd.]

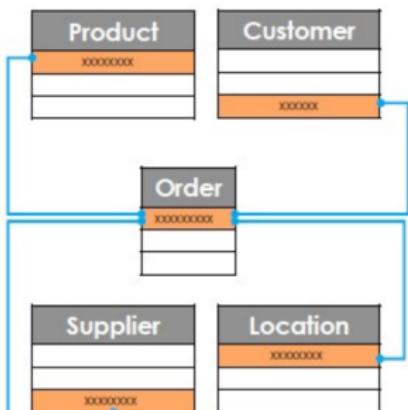
A (not exhaustive) list of graph databases (many have a free/community edition):

- GraphDB
- Neo4j
- Amazon Neptune
- AnzoGraph DB
- AllegroGraph (etc.)

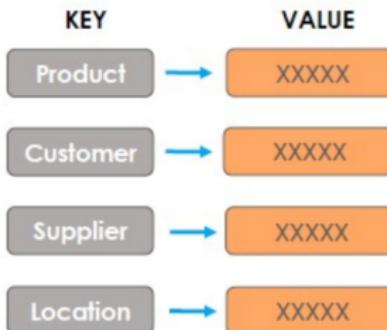
They often differ in their underlying graph data models, and accordingly their graph query languages also differ.

Comparison of Databases

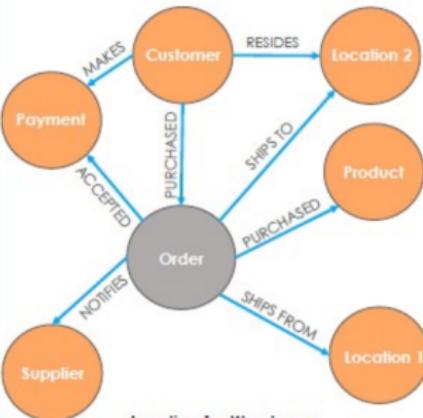
Relational Database



Key-Value Database



Graph Database



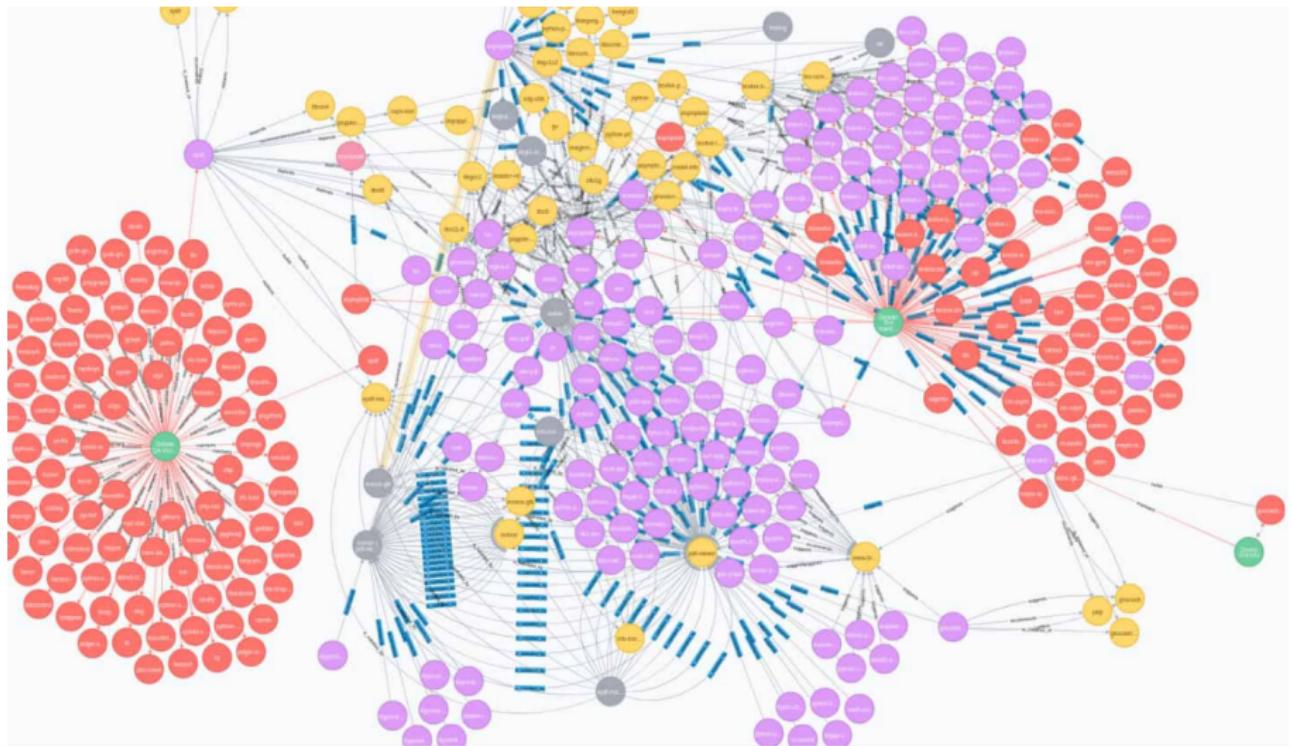
- Rigid Schema
- High Performance for transactions
- Poor performance for deep analytics

- Highly fluid schema/no schema
- High performance for simple transactions
- Poor performance for deep analytics

- Flexible schema
- High performance for complex transactions
- High performance for deep analytics

Location 1 = Warehouse
Location 2 = Delivery Location

Visualization in Graph Databases



Graph Database: Usage in iTelos Methodology

- By now, we know that RDF is the underlying graph data model in our iTelos KG development methodology
- In the Data Integration phase, the output of KarmaLinker is an RDF file defining the Data Knowledge Graph (DKG) which can be represented, visualized and queried upon using graph databases.
- Our choice of Graph Database for this phase is **GraphDB Free**.
- GraphDB Free is a highly efficient, robust, and scalable Semantic Graph Database (native RDF), providing the core infrastructure for solutions where modelling agility, data integration and relationship exploration are important.

GraphDB Free: Features

GraphDB Free is one of the few triplestores that can perform semantic inferencing at scale, allowing users to derive new semantic facts from existing facts. Some of its important features are as follows:-

- Free to use
- Manages tens of billions of RDF statements on a single server.
- Performs query and reasoning operations.
- Scalability both in terms of data volume and loading and inferencing speed
- Does not require a license file. It is, however, not open source
- Fully W3C standard-compliant [RDF, RDFS, OWL, SPARQL].

Contents

1 Graph Databases

2 Demo

Installation

- The easiest way to set up and run GraphDB is to use the native installations provided for the GraphDB Free edition.
- This kind of installation is the best option for your laptop/desktop computer, and does not require the use of a console, as it works in a graphic user interface (GUI).
- For this distribution, you do not need to download Java, as it comes pre-configured.
- Go to [GraphDB Free](#) and request your copy.
- You will receive an email with the download link and instructions (as per OS).

GraphDB Free GUI

The screenshot shows the GraphDB Free GUI interface. On the left is a sidebar with the following items:

- Import
- Explore
- SPARQL
- Monitor
- Setup
- Help

The main area has the following sections:

- View resource**: A search bar labeled "Search RDF resources..." and buttons for "Text" and "Visual".
- Active repository**: Shows a "Local" repository named "myrepo" with 2,566 total statements. It includes explicit (1,839), inferred (727), and expansion ratio (1.40) statistics. Buttons for Import RDF data, Import tabular data with OntoRefine, and Export RDF data are also present.
- Saved SPARQL queries**: A list of query templates:
 - Add statements: `PREFIX dc: <http://purl.org/dc/elements/1.1/> INSERT DATA { GRAPH...`
 - Clear graph: `CLEAR GRAPH <http://example>`
 - Remove statements: `PREFIX dc: <http://purl.org/dc/elements/1.1/> DELETE DATA { GRAPH...`
 - SPARQL Select template: `SELECT ?s ?p ?o WHERE { ?s ?p ?o . } LIMIT 100`

Creating a Repository

- The first step is to create a repository.
- Go to *Setup -> Repositories*.
- Click *Create new repository*.
- Enter a name (as per choice) as a *Repository ID* and leave all other optional configuration settings at their default values.
- Click the *Connect* button to set the newly created repository as the repository for this location.
- Use the pin to select it as the default repository.

Creating a Repository: Illustration

Repositories i

Repositories from: Local ⚙️ 🔑

The screenshot shows a user interface for managing repositories. At the top, there is a header with the text "Repositories from: Local" and two small icons: a gear and a key. Below this is a list of repositories, starting with "SYSTEM · System configuration repository". To the right of this list is a red "Delete" icon. A large orange button at the bottom left contains the text "Create new repository" with a plus sign icon. Below this button is another orange button labeled "Create from file".

Creating a Repository: Illustration

[Contd.]

Repositories from: Local ⚙️ 🔒

Connect repository

myrepo SYSTEM • System configuration repository

+ Create new repository

Creating a Repository: Illustration

[Contd.]



Loading a DKG through the GraphDB Workbench

- Go to *Import -> RDF*.
- Open the *User data* tab and click the *Upload RDF files* to upload the file
- Click the *Import* button.
- Enter the *Import* settings in the pop-up window.
- Start importing by clicking the *Import* button.

Loading a DKG through the GraphDB Workbench: Illustration

Import ⓘ

User data Server files ⓘ Help

Upload RDF files All RDF formats, up to 200 MB

Get RDF data from a URL All RDF formats

Import RDF text snippet Type or paste RDF data

Import

graphdb-news-dataset.nt Imported successfully in less than a second.	
pub-ontology-types.ttl Imported successfully in less than a second.	
pub-ontology.ttl Imported successfully in less than a second.	
pub-properties.ttl Imported successfully in less than a second.	
publishing-ontology.ttl Imported successfully in less than a second.	

Loading a DKG through the GraphDB Workbench: Illustration [Contd.]

Import settings X

Base IRI ⓘ

Target graph ⓘ From data The default graph Named graph

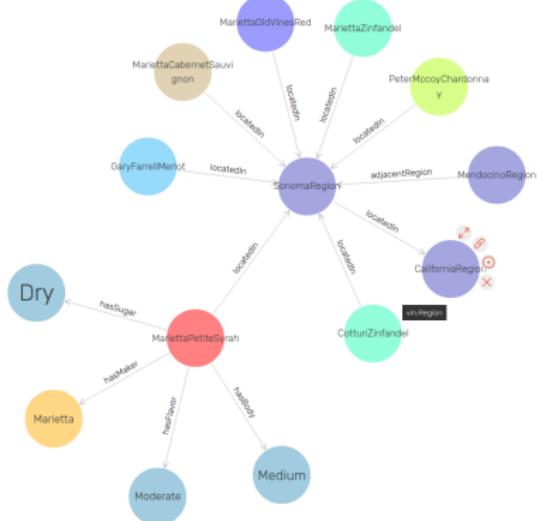
Enable replacement of existing data

[Show advanced settings ▾](#)

[Restore defaults](#) Cancel Import

Exploring Instances

To explore instances and their relationships, navigate to *Explore -> Visual graph*, and find an instance of interest through the *Easy graph* search box.



Exploring Instances [Contd.]

Hover over a node to see a menu for the following actions:

- Expand a node to show its relationships or collapse to hide them if already expanded. You can also expand the node by double-clicking on it.
- Copy a node's IRI to the clipboard.
- Focus on a node to restart the graph with this instance as the central one. Note that you will lose the current state of your graph.
- Delete a node to hide its relationships and hide it from the graph.
- Click on a node to see more info about it: a side panel opens on the right, including a short description (`rdfs:comment`), labels (`rdfs:label`), image (`foaf:depiction`) if present, and all `Data` `Type` properties.

Class Hierarchy

- To explore your data, navigate to *Explore -> Class hierarchy*.
- You can see a diagram depicting the hierarchy of the imported RDF classes by number of instances.
- The biggest circles are the parent classes and the nested ones are their children.

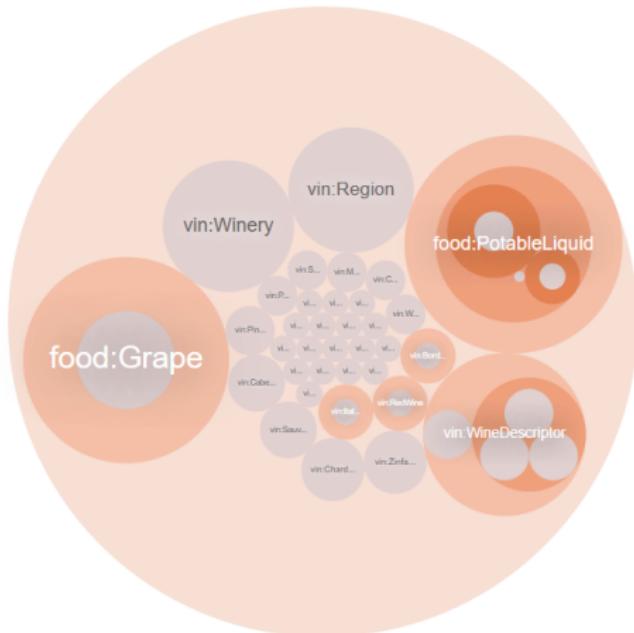
Class Hierarchy: Illustration

Class hierarchy i

Class Count i



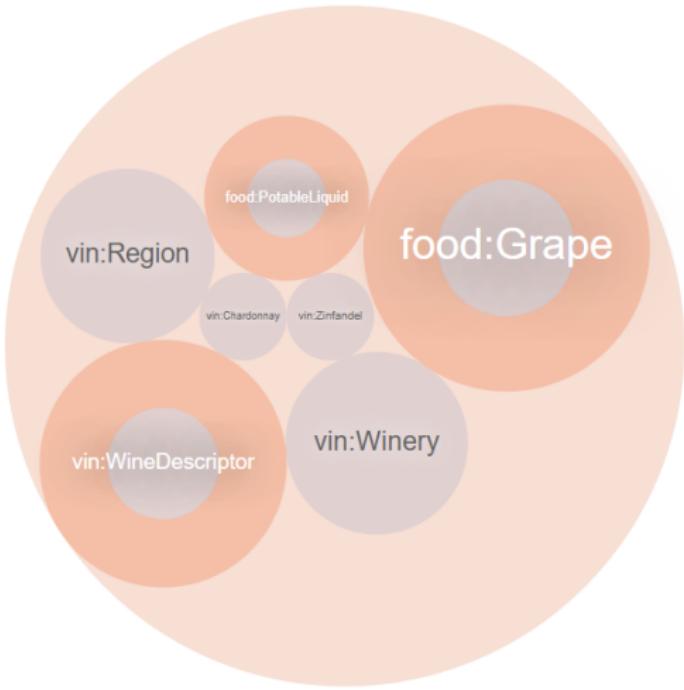
1



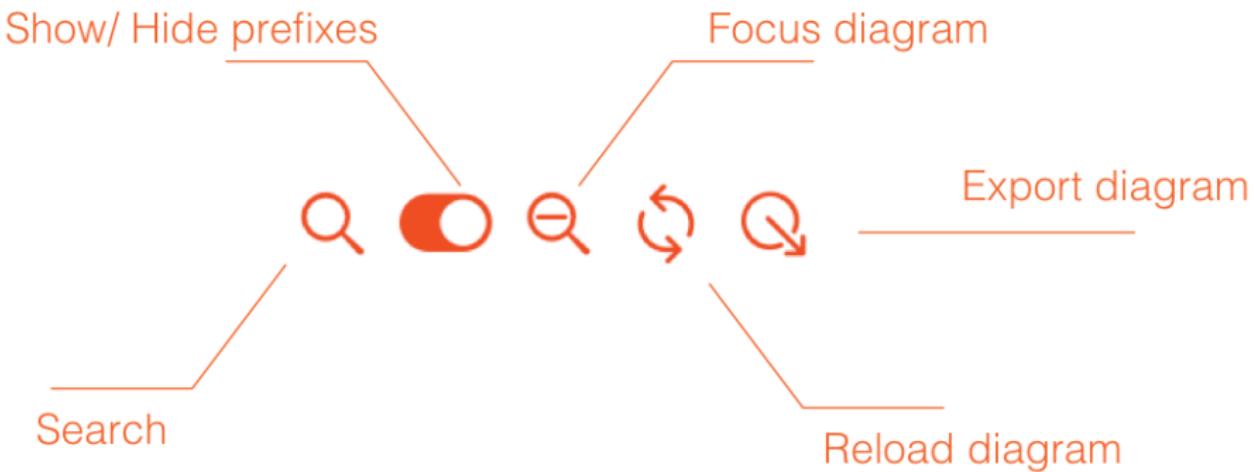
Class Hierarchy: Illustration [Contd.]

Class hierarchy (i)

Class Count (i)

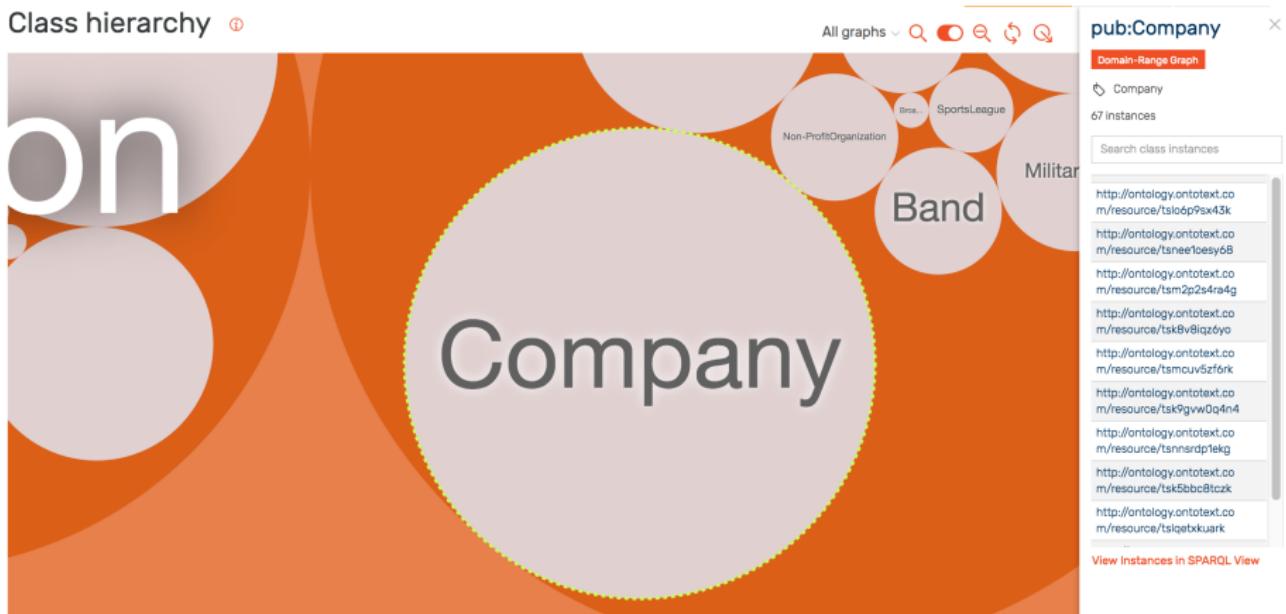


Class Hierarchy: Illustration [Contd.]



Class Hierarchy: Illustration [Contd.]

Class hierarchy



Domain-Range Graph

- To explore the connectedness of a given class, double click the class circle or the *Domain-Range Graph* button from the side panel.
- You can see a diagram that shows this class and its properties with their domain and range, where domain refers to all subject resources and range - to all object resources.
- You can also further explore the class connectedness by clicking:
 - the green nodes (object property class)
 - the labels - they lead to the View resource page, where you can find more information about the current class or property

Domain-Range Graph: Illustration

Domain-Range graph ⓘ



- main class node
- class node
- datatype node
- explicit property
- implicit property
- collapsed property

Class relationships

- To explore the relationships between the classes, navigate to *Explore -> Class relationships*.
- You can see a complicated diagram showing only the top relationships, where each of them is a bundle of links between the individual instances of two classes.
- Each link is an RDF statement, where the subject is an instance of one class, the object is an instance of another class, and the link is the predicate.
- Depending on the number of links between the instances of two classes, the bundle can be thicker or thinner and gets the color of the class with more incoming links.
- These links can be in both directions. To control which classes to display in the diagram, use the add/remove icon next to each class.

Class relationships: Illustration

GraphDB FREE

Import

Explore

Graphs overview

Class hierarchy

Class relationships

Visual graph

Similarity

SPARQL

Monitor

Setup

Class relationships ⓘ

Showing the dependencies between 10 classes

Filter classes

All Incoming Outgoing

Class	Links
vin:Region	193
vin:Winery	104
vin:WineFlavor	86
vin:WineSugar	92
vin:WineBody	82
vin:Zinfandel	56

The sunburst chart visualizes the dependency structure between 10 wine-related classes. The inner ring represents primary dependencies, and the outer ring represents secondary or tertiary dependencies. The classes shown are vin:Region, vin:Winery, vin:WineFlavor, vin:WineSugar, vin:WineBody, vin:Zinfandel, vin:SauvignonBlanc, vin:CabernetSauvignon, vin:Merlot, and vin:Chardonnay.

Class relationships: Illustration

GraphDB FREE

Import

Explore

Graphs overview

Class hierarchy

Class relationships

Visual graph

Similarity

SPARQL

Monitor

Setup

Help

Class relationships ⓘ

Showing the dependencies between 10 classes

Filter classes

All Incoming Outgoing

Class	Links
vin:Region	140 ←
vin:WineFlavor	86 ←
vin:WineSugar	86 ←
vin:WineBody	82 ←
vin:Winery	52 ←
vin:Zinfandel	5 ←
vin:Chardonnay	5 ←

vin:Region

vin:WineFlavor

vin:WineSugar

vin:WineBody

vin:Winery

vin:Zinfandel

vin:Chardonnay

vin:CabernetSauvignon

vin:SauvignonBlanc

vin:Merlot

vin:PinotNoir

vin:Riesling

vin:Rhone

vin:WhiteWines

myrepo

X Q Q

Class relationships: Illustration

GraphDB FREE

Import

Explore

Graphs overview

Class hierarchy

Class relationships

SPARQL

Monitor

Setup

Help

myrepo

Filter classes

All Incoming Outgoing

Class	Links
vin:Region	53
vin:Winery	52
vin:Zinfandel	51
vin:Chardonnay	49
vin:CabernetSauvignon	40
vin:SauvignonBlanc	32

The sunburst chart visualizes the dependencies between 10 classes. The inner ring represents the primary distribution of links, while the outer ring represents secondary or tertiary relationships. The classes are color-coded and labeled on the chart:

- vin:Region (purple)
- vin:Winery (blue)
- vin:Zinfandel (orange)
- vin:Chardonnay (green)
- vin:PinotNoir (light green)
- vin:Riesling (yellow)
- vin:Merlot (pink)
- vin:CabernetSauvignon (red)
- vin:SauvignonBlanc (dark green)
- vin:PetiteSirah (dark red)

Relationships are shown as lines connecting the segments of the sunburst chart.

Class relationships: Illustration

GraphDB FREE

Import

Explore

Graphs overview

Class hierarchy

Class relationships

Visual graph

Similarity

SPARQL

Monitor

Setup

Help

Class relationships ⓘ

Showing the dependencies between 10 classes

Filter classes

All Incoming Outgoing

Class	Links
vin:Region	193 ↔
vin:Winery	104 ↔
vin:WineFlavor	86 ←
vin:WineSugar	92 ↔
vin:WineBody	82 ←
vin:Zinfandel	56 ↔
vin:Chardonnay	54 ↔

vin:Region

vin:Winery

vin:WineFlavor

vin:WineSugar

vin:WineBody

vin:Zinfandel : vin:Region 15 →

vin:locatedIn : 15 →

vin:SauvignonBlanc

vin:CabernetSauvignon

Rouge

Blanc

SPARQL

SPARQL is a query language for RDF graph databases with the following types:

- **SELECT** - returns tabular results;
- **CONSTRUCT** - creates a new RDF graph based on query results;
- **ASK** - returns YES if the query has a solution, otherwise “NO”;
- **DESCRIBE** - returns RDF data about a resource;
- **INSERT** - inserts triples into a graph;
- **DELETE** - deletes triples from a graph (etc.)

You have to use SPARQL to formalize CQs and explore the KG.

SPARQL

For learning SPARQL, please visit the following resources (**highly recommended**):

- [SPARQL Introductory Tutorial](#)
- [How to Query RDFS SPARQL](#)
- [Complex Queries with SPARQL](#)
- [SPARQL 1.1 Query Language](#)
- [SPARQL playground](#)

Query data through the Workbench: Illustration

The screenshot shows the GraphDB Workbench interface. On the left is a sidebar with icons for Import, Explore, SPARQL (highlighted in red), Monitor, Setup, and Help. The main area is titled "SPARQL Query & Update". It displays a query editor with the following SPARQL code:

```
1 SELECT ?s ?p ?o
2 WHERE {
3   ?s ?p ?o .
4 } LIMIT 100
```

Below the code are several orange action icons: a save icon, a folder icon, a link icon, a copy icon, and a refresh icon. A "Run" button is at the bottom right of the editor. To the right of the editor are tabs for Table, Raw Response, Pivot Table, and Google Chart. The "Table" tab is selected. The results table shows the following data:| | s | p | o |
| --- | --- | --- | --- |
| 1 | rdf:type | rdf:type | rdf:Property |
| 2 | rdfs:subPropertyOf | rdf:type | rdf:Property |
| 3 | rdfs:subPropertyOf | rdf:type | owl:TransitiveProperty |
| 4 | rdfs:subClassOf | rdf:type | rdf:Property |
| 5 | rdfs:subClassOf | rdf:type | owl:TransitiveProperty |
| 6 | rdfs:domain | rdf:type | rdf:Property |

At the top right of the results area, there are buttons for "Editor only", "Editor and results" (which is highlighted in red), "Results only", and a download icon. A status message at the top right says "Showing results from 1 to 100 of 100. Query took 0.1s, yesterday at 20:59." and a "Filter query results" input field.

Query data through the Workbench: Illustration [Contd.]

The screenshot shows the GraphDB Workbench interface. On the left is a sidebar with navigation links: Import, Explore, SPARQL (highlighted in red), Monitor, Setup, and Help. The main area is titled "SPARQL Query & Update". It features a query editor with tabs for Unnamed, SPARQL Select template, and vin:Winery. The current tab contains the following SPARQL query:

```
PREFIX wine: <http://www.w3.org/TR/2003/PR-owl-guide-20031209/wine#>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
SELECT ?x
WHERE {
    ?x wine:hasFlavor wine:Strong .
    ?x wine:locatedIn wine>NewZealandRegion .
}
```

To the right of the query are several icons: a save icon, a folder icon, a link icon, a double arrow icon, and a refresh icon. Below the query is a "Run" button and a note: "Press Alt+Enter to autocomplete". To the right of the query editor is a results panel with tabs for Table, Raw Response, Pivot Table, and Google Chart. The "Table" tab is selected. The results table shows two rows of data:| | x |
| --- | --- |
| 1 | vin:CorbansPrivateBinSauvignonBlanc |
| 2 | vin:CorbansSauvignonBlanc |

Query data through the Workbench: Illustration [Contd.]

The screenshot shows the GraphDB Workbench interface. On the left, there's a sidebar with icons for Import, Explore, SPARQL (which is highlighted in red), Monitor, Setup, and Help. The main area is titled "SPARQL Query & Update". It has tabs for Unnamed, SPARQL Select template, and vin:Winery. Below these tabs, there's a query editor with the following SPARQL code:

```
1 PREFIX wine: <http://www.w3.org/TR/2003/PR-owl-guide-20031209/wine#>
2 PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
3 SELECT ?x
4
5 WHERE {
6   ?x wine:hasFlavor wine:Strong .
7   ?x wine:locatedIn wine:NewZealandRegion .
8 }
```

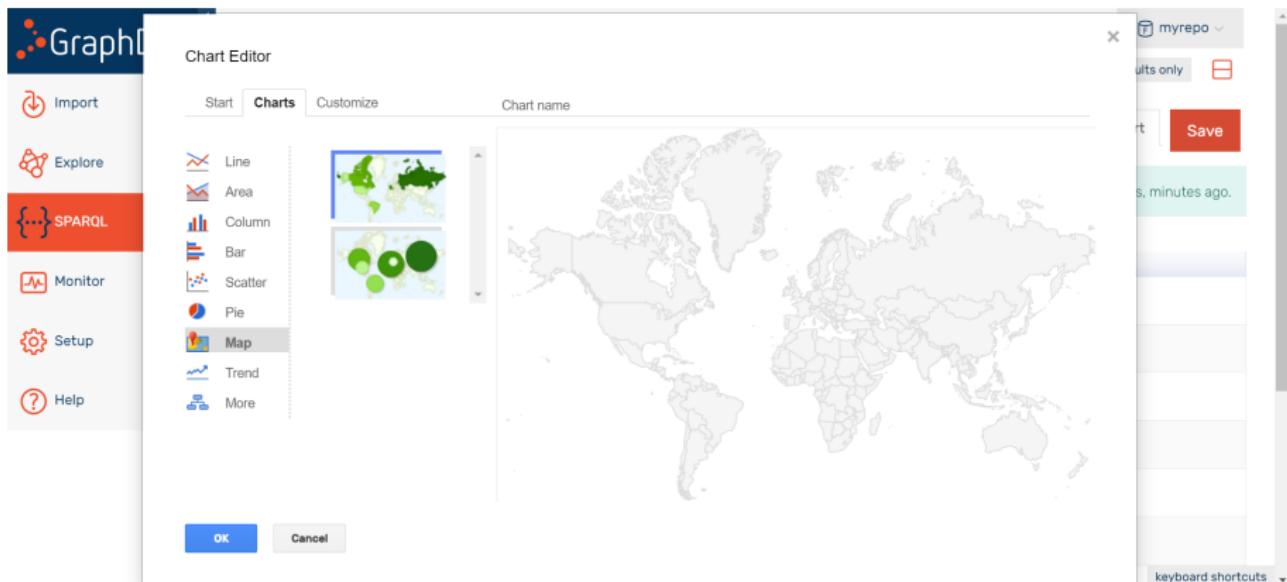
On the right side of the query editor, there are several icons: a save icon, a folder icon, a link icon, a copy icon, and a refresh icon. A "Run" button is located at the bottom right of the query editor. Below the editor, a message says "Press Alt+Enter to autocomplete".

At the top right of the main window, there's a dropdown menu set to "myrepo". Below it, there are three tabs: "Editor only" (selected), "Editor and results", and "Results only". To the right of these tabs is a "Save" button. At the bottom right of the main window is a "Save" button.

The results section shows the following JSON-like output:

```
1 { "head": {
2   "vars": [
3     "x"
4   ]
5 },
6 "results": {
7   "bindings": [
8     {
9       "x": {
10         "type": "uri",
11         "value": "http://www.w3.org/TR/2003/PR-owl-guide-20031209/wine#CorbansPrivateBinSauvignonBlanc"
12       }
13     },
14     {
15       "x": {
16         "type": "uri",
17         "value": "http://www.w3.org/TR/2003/PR-owl-guide-20031209/wine#CorbansSauvignonBlanc"
18       }
19     }
20   ]
21 }
```

Query data through the Workbench: Illustration [Contd.]



References

Please visit the following reference resources (**highly recommended**):

- [GraphDB Free Documentation](#)
- [GraphDB Fundamentals](#)
- [SPARQL in 11 minutes](#)
- [GraphDB Workbench Tutorials](#)

Thank you for listening!



KDI • Knowledge and Data Integration



W10.L19.M6.T19.2.2



GraphDB