Accessing and Integrating Data through Ontologies

Diego Calvanese

KRDB Research Centre for Knowledge and Data
Free University of Bozen-Bolzano, Italy

Department of Computing Science
Umeå University, Sweden

European Autonomous Network Forum 2021

14–15 October 2021 – Online

Data integration

Databases are great!
They let us manage efficiently huge amounts of data ...

... assuming you have put all data into your schema.

However, the reality is much more complicated and **heterogeneous**:

- Data sets were created independently.
- Data are often stored across different sources.
- Data sources are controlled by different people / organizations.

Goal of data integration

To put together different data sources,
created for different purposes,
and controlled by different people,
making them accessible in a uniform way.

Why heterogeneity?

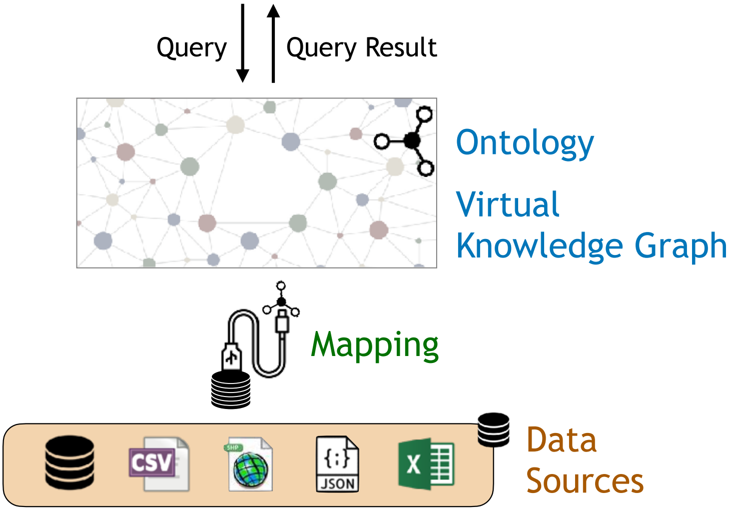
- **Data model heterogeneity**: Relational data, graph data, xml, json, csv, text files, . . .
- **System heterogeneity**: Even when systems adopt the same data model, they are not always fully compatible.
- **Schema heterogeneity**: Different people see things differently, and design schemas differently!
- **Data-level heterogeneity**: e.g., ‘IBM’ vs. ‘Int. Business Machines’ vs. ‘International Business Machines’.

How to address heterogeneity?

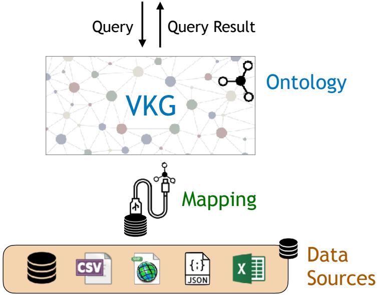
We combine three key ideas:

1. Use a global (or integrated) schema and **map the data sources to the global schema**.
2. Adopt a very flexible data model for the global schema
   $\leadsto$ **Knowledge Graph** whose vocabulary is expressed in an **ontology**.
3. Exploit **virtualization**, i.e., the KG is not materialized, but kept virtual.

Virtual Knowledge Graph (VKG) architecture



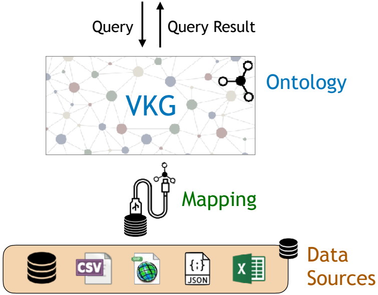
Why an ontology?



An ontology is a structured formal representation of concepts and their relationships that are relevant for the domain of interest.

- In the VKG setting, the ontology has a twofold purpose:
  - It defines a vocabulary of terms to denote classes and properties that are familiar to the user.
  - It extends the data in the sources with background knowledge about the domain of interest, and this knowledge is machine processable.
- One can make use of custom-built domain ontologies.
- In addition, one can rely on standard ontologies, which are available for many domains.

Why a Knowledge Graph for the global schema?

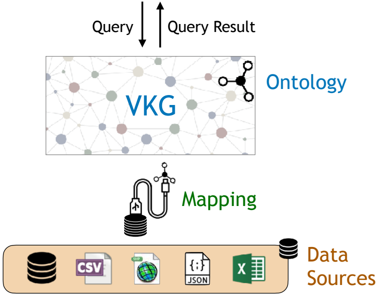


The traditional approach to data integration adopts a relational global schema.

A Knowledge Graph, instead:

- Does not require to commit early on to a specific structure.
- Can better accommodate heterogeneity.
- Can better deal with missing / incomplete information.
- Does not require complex restructuring operations to accommodate new information or new data sources.

Why mappings?

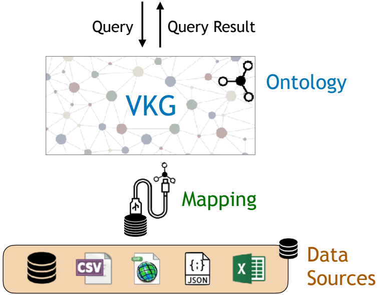


The traditional approach to data integration relies on mediators, which are specified through complex code.

Mappings, instead:

- Provide a declarative specification, and not code.
- Are easier to understand, and hence to design and to maintain.
- Support an incremental approach to integration.
- Are machine processable, hence can be used for query optimization.

Why virtualization?



Materialized data integration relies on extract-transform-load (ETL) operations, to load data from the sources into an integrated data store / data warehouse / materialized KG.

In the virtual approach, instead:

- The data stays in the sources and is only accessed at query time.
- No need to construct a large and potentially costly materialized data store and keep it up-to-date.
- Hence the data is always fresh wrt the latest updates at the sources.
- One can rely on the existing data infrastructure and expertise.
- There is better support for an incremental approach to integration.

Outline

1. Ontology-Based Data Integration
2. Applications of the VKG Approach
3. The VKG Framework
4. The Ontop System
5. Conclusions

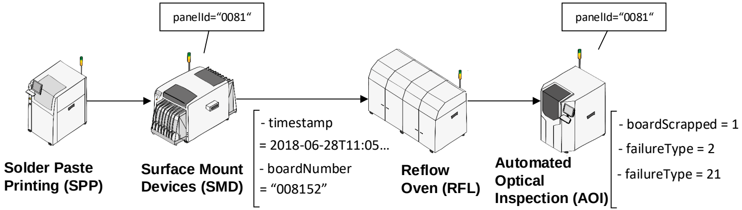
Outline

Applications of the VKG approach

- Adopted in many academic and industrial use cases.[1]
- Some application areas:
  - Industry 4.0
    - Ability to deal with data coming from different vendors, or with historical heterogeneous data.
    - Examples: Equinor, Siemens, Bosch
  - Analytical processing / BI
    - Combine internal data, manual processes (e.g., Excel), and external data
    - Data privacy issues / GDPR: we need to avoid data copies
    - Examples: Toscana Open Research, a large European university, a large TLC company
  - Geospatial data
    - GeoSPARQL over PostGIS
    - Examples: LinkedGeoData.org, South Tyrolean Open Data Hub

[1] G. Xiao, L. Ding, B. Cogrel, and D. Calvanese. Virtual knowledge graphs: An overview of systems and use cases. *Data Intelligence*, 1:201–223, 2019.

Failure detection for Surface Mounting Process pipeline in Bosch[2]



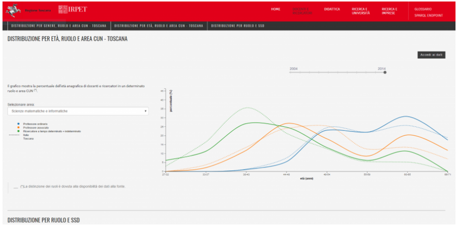
- Failure detection fundamentally relies on the integration and analysis of data generated in different phases of the process.
- The involved machines come from different suppliers and rely on distinct formats.

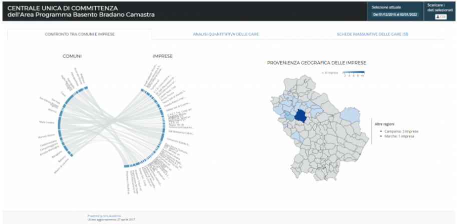
[2] E. Güzel Kalaycı, I. Grangel Gonalez, F. Lösch, G. Xiao, A. ul Mehdi, E. Kharlamov, and D. Calvanese. Semantic integration of Bosch manufacturing data using virtual knowledge graphs. In *Proc. of ISWC*, 2020.

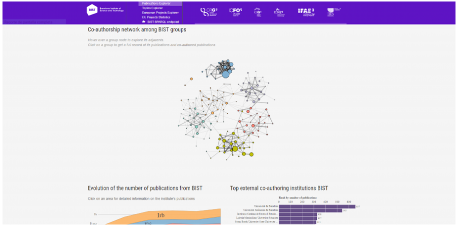
Applications of the VKG approach

- Adopted in many academic and industrial use cases.
- Some application areas:
  - Industry 4.0
    - Ability to deal with data coming from different vendors, or with historical heterogeneous data.
    - Examples: Equinor, Siemens, Bosch
  - Analytical processing / BI
    - Combine internal data, manual processes (e.g., Excel), and external data
    - Data privacy issues / GDPR: we need to avoid data copies
    - Examples: Toscana Open Research, a large European university, a large TLC company
  - Geospatial data
    - GeoSPARQL over PostGIS
    - Examples: LinkedGeoData.org, South Tyrolean Open Data Hub

Toscana Open Research







http://www.toscanaopenresearch.it/en/

A large European university

- Internal data
  - Research funding, HR, teaching, etc.
  - Redundant applications due to the merge of several universities.
  - Operational data store and data warehouse.
  - Many processes are still using Excel.
- External data
  - Open Data (from the ministry, EU commission and public initiatives).
  - Commercial bibliometric data.
  - Mainly for benchmarking.

VKG over scientific documentation for a large TLC company

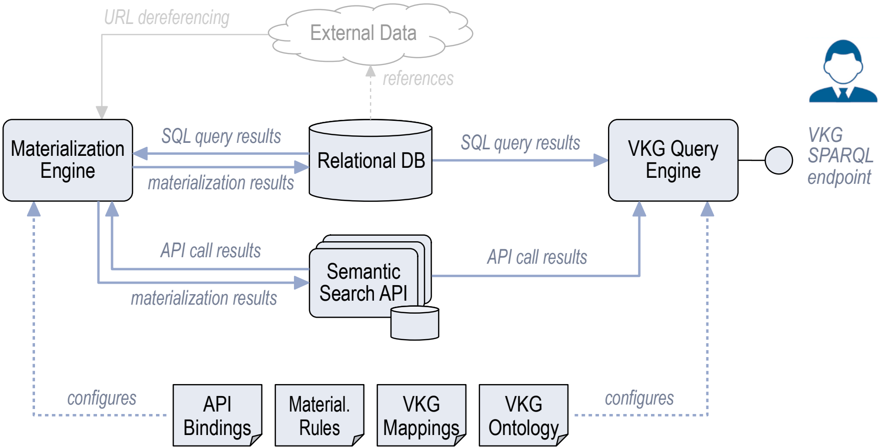
Goal: build a **virtual knowledge graph** integrating structured data in a proprietary platform and the results of information extraction from related semi-structured data.

- Structured data is provided by a relational database.
- **Semi-structured data** consisting of text with little (if any) structure or markup, such as natural language text, HTML documents, PDF files.
- **Information extraction (IE)** aims at extracting structured information from semi-structured data, possibly leveraging natural language processing (NLP) techniques.

Motivations:

- Provide an unambiguous formalization of the knowledge in the platform, to ease exploitation.
- Provide an integrated, queryable, up-to-date view over all available information.
- Enable more advanced services, such as intelligent search and intelligent recommendation.

High-Level architecture of VKG over semi-structured data

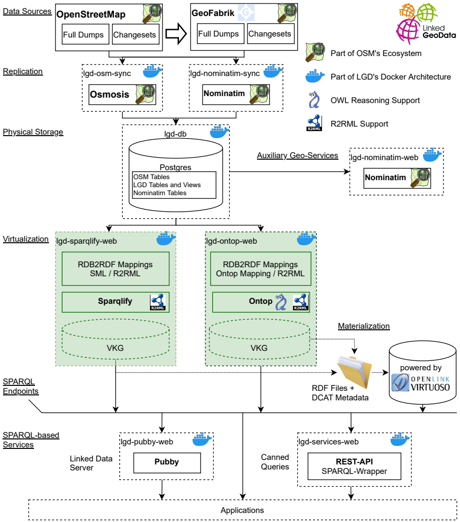


Applications of the VKG approach

- Adopted in many academic and industrial use cases.
- Some application areas:
  - Industry 4.0
    - Ability to deal with data coming from different vendors, or with historical heterogeneous data.
    - Examples: Equinor, Siemens, Bosch
  - Analytical processing / BI
    - Combine internal data, manual processes (e.g., Excel), and external data
    - Data privacy issues / GDPR: we need to avoid data copies
    - Examples: Toscana Open Research, a large European university, a large TLC company
  - Geospatial data
    - GeoSPARQL over PostGIS
    - Examples: LinkedGeoData.org, South Tyrolean Open Data Hub

LinkedGeoData.org

- LGD converts OpenStreetMap to RDF
- one of the most important Geospatial Knowledge Graphs
- The next version of LGD will be based on Ontop
- ... in collaboration with University of Leipzig



LinkedGeoData.org

LinkedGeoData.org

endpoint address: http://localhost:8080/sparql | ontop v4.1.0-beta-1-SNAPSHOT

Playground | Example Queries

Query x | Query 1 x | Query 2 x | Query 3 x | Query 4 x | Query 6 x | Query 7 x | road segment x | isHostedBy x | Query 11 x | Query 10 x | Query 12 x | Query 13 x | Query 5 x | Query 8 x | Query 9 x | Query 14 x | +

```
SELECT ?x ?wkt ?wktLabel ?wktColor WHERE {
  {  ?x a lgdo:University ; geo:asWKT ?wkt . OPTIONAL {?x rdfs:label ?wktLabel . FILTER (LANG(?wktLabel) = '')}
    BIND('red' AS ?wktColor)
  }
  UNION {
    ?u a lgdo:University ; geo:asWKT ?uWkt . OPTIONAL {?u rdfs:label ?uktLabel . FILTER (LANG(?uktLabel) = '')}
    ?x a lgdo:Restaurant ; geo:asWKT ?wkt ; rdfs:label ?wktLabel . FILTER (LANG(?wktLabel) = '')
    FILTER(geof:distance(?wkt, ?uWkt, uom:metre) < 200)
    BIND('blue' AS ?wktColor)
  }
  UNION {
    ?u a lgdo:University ; geo:asWKT ?uWkt . OPTIONAL {?u rdfs:label ?uktLabel . FILTER (LANG(?uktLabel) = '')}
    BIND(geof:buffer(?uWkt, 200, uom:metre) AS ?wkt)  BIND('red' AS ?wktColor)
}
```

Table | Response | Pivot Table | Google Chart | Geo | </>

VKG over the South Tyrolean Open Data Hub (ODH)

`https://sparql.opendatahub.bz.it/`

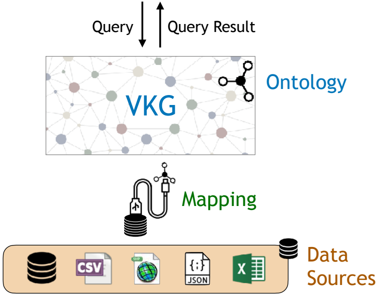
- ODH publishes tourism, mobility, and weather data from different providers through a JSON-based Web API.
- The backend relies on PostgreSQL databases.
- Joint project between Ontopic and NOI Techpark on extending ODH with a VKG.

Outline

Components of the VKG framework

We consider now the main components that make up the VKG framework, and the languages used to specify them.

In defining such languages, we need to consider the **tradeoff between expressive power and efficiency**, where the key point is efficiency with respect to the data.

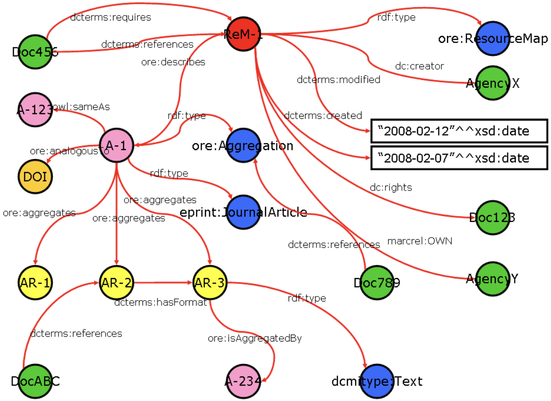


The W3C has standardized languages that are suitable for VKGs:

1. Knowledge graph: expressed in **RDF** [W3C Rec. 2014] (v1.1)
2. Ontology $\mathcal{O}$: expressed in **OWL 2 QL** [W3C Rec. 2012]
3. Mapping $\mathcal{M}$: expressed in **R2RML** [W3C Rec. 2012]
4. Query: expressed in **SPARQL** [W3C Rec. 2013] (v1.1)

RDF – Data is represented as a graph

The graph consists of a set of **subject-predicate-object triples**:



Object property:

`<A-1> ore:describes <ReM-1> .`

Data property:

`<ReM-1> :created "2008-02-07" .`

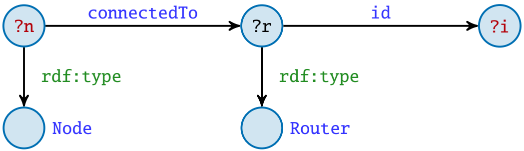
Class membership:

`<ReM-1> rdf:type ore:ResourceMap .`

SPARQL query language

- Is the standard query language for RDF data. [W3C Rec. 2008, 2013]
- Core query mechanism is based on **graph matching**.

```
SELECT ?n ?i
WHERE { ?n rdf:type Node .
        ?n connectedTo ?r .
        ?r rdf:type Router .
        ?r id ?i .
      }
```

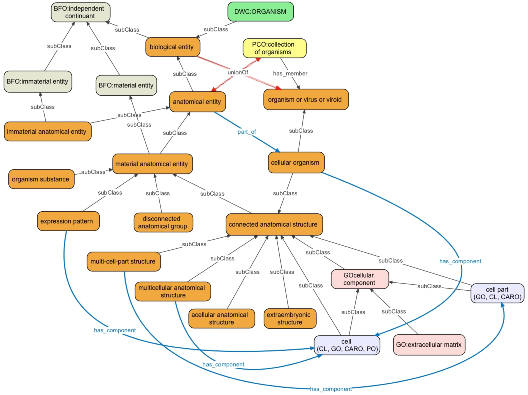


Additional language features (SPARQL 1.1):

- `UNION`: matches one of alternative graph patterns
- `OPTIONAL`: produces a match even when part of the pattern is missing
- complex `FILTER` conditions
- `GROUP BY`, to express aggregations
- `MINUS`, to remove possible solutions
- property paths (regular expressions)

What is an ontology?

- An ontology conceptualizes a domain of interest in terms of **concepts/classes**, (binary) **relations**, and their **properties**.
- It typically organizes the concepts in a hierarchical structure.
- Ontologies are often represented as graphs.



What is an ontology?

- An ontology conceptualizes a domain of interest in terms of **concepts/classes**, (binary) **relations**, and their **properties**.
- It typically organizes the concepts in a hierarchical structure.
- Ontologies are often represented as graphs.
- However, an ontology is actually a **logical theory**, expressed in a suitable fragment of first-order logic

$\forall x.\,\mathsf{Pressure}(x) \rightarrow \mathsf{Measurement}(x)$
$\forall x.\,\mathsf{Porosity}(x) \rightarrow \mathsf{Measurement}(x)$
$\forall x.\,\mathsf{Permeability}(x) \rightarrow \mathsf{Measurement}(x)$
$\forall x.\,\mathsf{Temperature}(x) \rightarrow \mathsf{Measurement}(x)$
$\forall x.\,\mathsf{Pressure}(x) \rightarrow \neg\mathsf{Porosity}(x) \land \neg\mathsf{Permeability}(x) \land \neg\mathsf{Temperature}(x)$
$\forall x.\,\mathsf{Porosity}(x) \rightarrow \neg\mathsf{Permeability}(x) \land \neg\mathsf{Temperature}(x)$
$\forall x.\,\mathsf{Permeability}(x) \rightarrow \neg\mathsf{Temperature}(x)$

$\forall x.\,\mathsf{HydrostaticPressure}(x) \rightarrow \mathsf{Pressure}(x)$
$\forall x.\,\mathsf{FormationPressure}(x) \rightarrow \mathsf{Pressure}(x)$
$\forall x.\,\mathsf{PorePressure}(x) \rightarrow \mathsf{Pressure}(x)$
$\forall x.\,\mathsf{HydrostaticPressure}(x) \rightarrow \neg\mathsf{FormationPressure}(x) \land \neg\mathsf{PorePressure}(x)$
$\forall x.\,\mathsf{FormationPressure}(x) \rightarrow \neg\mathsf{PorePressure}(x)$

$\forall x,y.\,\mathsf{hasFormationPressure}(x,y) \rightarrow \mathsf{Wellbore}(x) \land \mathsf{FormationPressure}(y)$
$\forall x,y.\,\mathsf{hasDepth}(x,y) \rightarrow \mathsf{FormationPressure}(x) \land \mathsf{Depth}(y)$
$\forall x.\,\mathsf{FormationPressure}(x) \rightarrow \exists y.\,\mathsf{hasDepth}(x,y)$

$\forall x,y.\,\mathsf{hasFormationPressure}(x,y) \rightarrow \mathsf{hasMeasurement}(x,y)$

$\forall x,y.\,\mathsf{completionDate}(x,y) \rightarrow \mathsf{Wellbore}(x) \land \mathsf{xsd{:}dateTime}(y)$
$\forall x.\,\mathsf{Wellbore}(x) \rightarrow (\sharp\{y \mid \mathsf{completionDate}_{\mathsf{wb}}(x,y)\} \leq 1)$
$\forall x,y.\,\mathsf{wellboreTrack}_{\mathsf{wb}}(x,y) \rightarrow \mathsf{Wellbore}(x) \land \mathsf{xsd{:}string}(y)$
$\forall x.\,\mathsf{Wellbore}(x) \rightarrow (\sharp\{y \mid \mathsf{wellboreTrack}_{\mathsf{wb}}(x,y)\} \leq 1)$

$\forall x,y.\,\mathsf{hasCoreSample}(x,y) \rightarrow \mathsf{Core}(x) \land \mathsf{CoreSample}(y)$
$\forall x.\,\mathsf{CoreSample}(x) \rightarrow \exists y.\,\mathsf{hasCoreSample}(y,x) \land \mathsf{Core}(y)$
$\ldots$

What is an ontology?

- An ontology conceptualizes a domain of interest in terms of **concepts/classes**, (binary) **relations**, and their **properties**.
- It typically organizes the concepts in a hierarchical structure.
- Ontologies are often represented as graphs.
- However, an ontology is actually a **logical theory**, expressed in a suitable fragment of first-order logic, or better, in **description logics**.

$$
\begin{aligned}
\text{Pressure} &\sqsubseteq \text{Measurement}\\
\text{Porosity} &\sqsubseteq \text{Measurement}\\
\text{Permeability} &\sqsubseteq \text{Measurement}\\
\text{Temperature} &\sqsubseteq \text{Measurement}\\
\text{Pressure} &\sqsubseteq \neg\text{Porosity} \sqcap \neg\text{Permeability} \sqcap \neg\text{Temperature}\\
\text{Porosity} &\sqsubseteq \neg\text{Permeability} \sqcap \neg\text{Temperature}\\
\text{Permeability} &\sqsubseteq \neg\text{Temperature}\\
\text{HydrostaticPressure} &\sqsubseteq \text{Pressure}\\
\text{FormationPressure} &\sqsubseteq \text{Pressure}\\
\text{PorePressure} &\sqsubseteq \text{Pressure}\\
\text{HydrostaticPressure} &\sqsubseteq \neg\text{FormationPressure} \sqcap \neg\text{PorePressure}\\
\text{FormationPressure} &\sqsubseteq \neg\text{PorePressure}\\
\exists\text{hasFormationPressure} &\sqsubseteq \text{Wellbore}\\
\exists\text{hasFormationPressure}^- &\sqsubseteq \text{FormationPressure}\\
\exists\text{hasDepth} &\sqsubseteq \text{FormationPressure}\\
\exists\text{hasDepth}^- &\sqsubseteq \text{Depth}\\
\text{FormationPressure} &\sqsubseteq \exists\text{hasDepth}\\
\text{hasFormationPressure} &\sqsubseteq \text{hasMeasurement}\\
\exists\text{completionDate}_{wb} &\sqsubseteq \text{Wellbore}\\
\exists\text{completionDate}_{wb}^- &\sqsubseteq \text{xsd:dateTime}\\
\text{Wellbore} &\sqsubseteq (\leq 1\ \text{completionDate}_{wb})\\
\exists\text{wellboreTrack}_{wb} &\sqsubseteq \text{Wellbore}\\
&\cdots
\end{aligned}
$$

The OWL 2 QL ontology language

- **OWL 2 QL** is one of the three standard sub-languages of the very expressive standard ontology language OWL 2. [W3C Rec. 2012]
- It is considered a lightweight ontology language:
  - controlled expressive power
  - efficient inference
- Optimized for accessing large amounts of data
  - Queries over the ontology can be rewritten into SQL queries over the underlying relational database (First-order rewritability).
  - Logical consistency of ontology and data can also be checked by executing SQL queries over the underlying database.

Constructs of OWL 2 QL

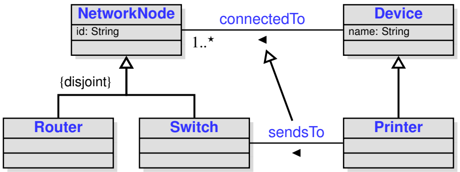
In an OWL 2 QL ontology, one can express knowledge about the classes and properties in the domain of interest by means of various types of assertions.

- Subclass assertions — Router `rdfs:subClassOf` NetworkNode
- Class disjointness — NetworkNode `owl:disjointWith` User
- Domain of a property — connectedTo `rdfs:domain` User
- Range of a property — connectedTo `rdfs:range` NetworkNode
- Subproperty assertions — sendsTo `rdfs:subPropertyOf` connectedTo
- Inverse properties — accesses `owl:inverseOf` isAccessedBy
- Mandatory participation to a property — ... `owl:someValuesFrom` ...

Representing OWL 2 QL ontologies as UML class diagrams

There is a close correspondence between OWL 2 QL and conceptual modeling formalisms, such as UML class diagrams and ER schemas.

Router `rdfs:subClassOf` NetworkNode	subclass
Router `owl:disjointWith` Switch	disjointness
connectedTo `rdfs:domain` Device	domain
connectedTo `rdfs:range` NetworkNode	range
sendsTo `rdfs:subPropertyOf` connectedTo	sub-association
... `owl:someValuesFrom` ...	mandatory participation



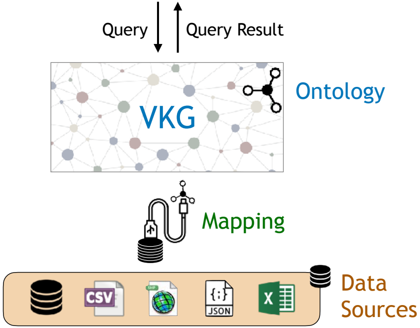
In fact, to visualize an OWL 2 QL ontology, we can use standard UML class diagrams.

Use of mappings

In the VKG framework, the mapping encodes how the data in the sources should be used to create the Virtual Knowledge Graph, which is formulated in the vocabulary of the ontology.

VKG defined from the mapping and the data.

- Queries are answered with respect to the ontology and the data of the VKG.
- The data of the VKG is not materialized (it is virtual!).
- Instead, the information in the ontology and the mapping is used to translate queries over the ontology into queries formulated over the sources.



Note: The graph is **always up to date** wrt the data sources.

Mapping language

The mapping consists of a set of assertions of the form

SQL Query $\rightsquigarrow$ Class membership assertion

SQL Query $\rightsquigarrow$ Property membership assertion

Intuition behind the mapping

The answers returned by the SQL Query in the left-hand side are used to create the objects (and values) that populate the Class / Property in the right-hand side.

Note: The mapping contains also a mechanism to transform values retrieved from the database into objects of the VKG (thus solving the so-called impedance mismatch).

Query answering in VKGs

In VKGs, we want to answer queries formulated over the ontology, by using the data provided by the data sources through the mapping.

- The ontology contains **domain knowledge** that can be used to enrich answers.

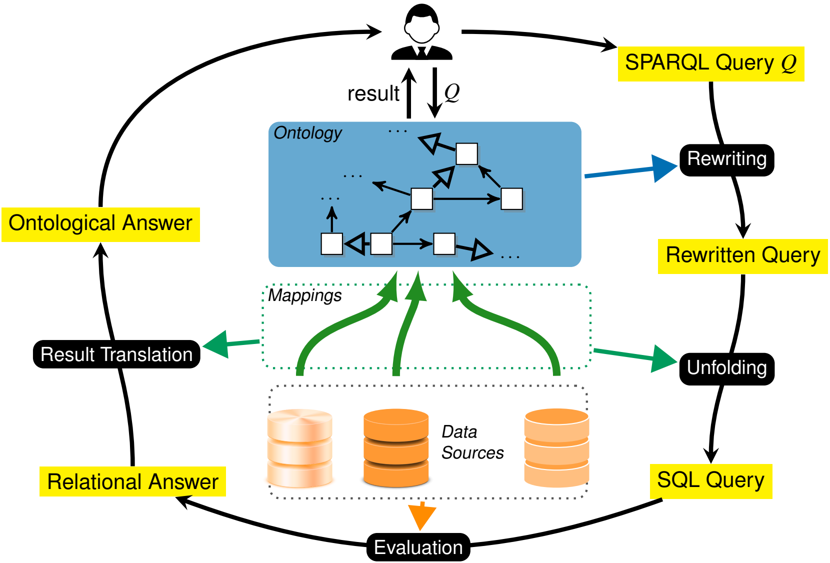
  Example: Suppose that our data contains LJ-2025 among the Printers, and that the ontology states that each Printer is a NetworkDevice.
  If we ask for all NetworkDevices, we should return also LJ 2025, considering both the data and the knowledge in the ontology.

- The **mapping** encodes the information of how to translate a query over the ontology into a query over the **database**.

A VKG query answering engine has to take into account all these types of information.

Query answering by query rewriting

Query answering by query rewriting



Outline

The *Ontop* system

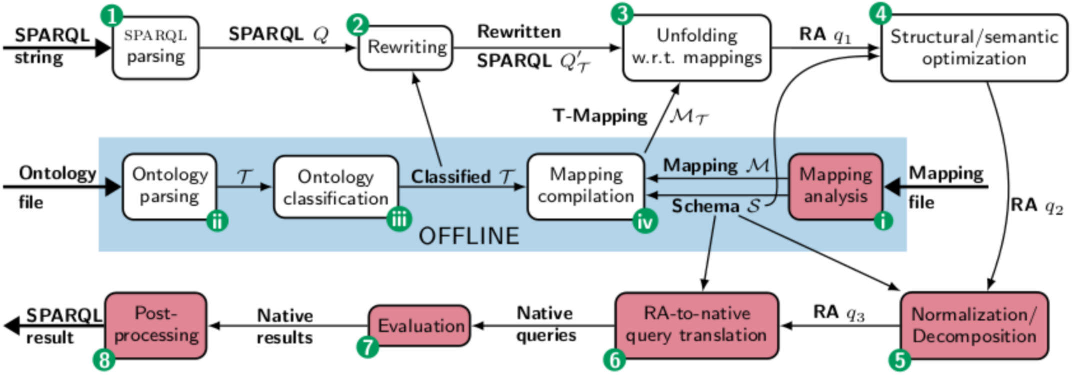
ontop

https://ontop-vkg.org/

- State-of-the-art VKG system.
- Addresses the key challenges in query answering of scalability and performance.
- Compliant with all relevant Semantic Web standards:
  RDF, RDFS, OWL 2 QL, R2RML, SPARQL, and GeoSPARQL.
- Supports all major relational DBMSs:
  Oracle, DB2, MS SQL Server, Postgres, MySQL, Teiid, Dremio, Denodo, etc.
- Open-source and released under Apache 2 license.

unibz

Query answering in *Ontop*



Developer community

unibz

ONTOPIC

Birkbeck UNIVERSITY OF LONDON

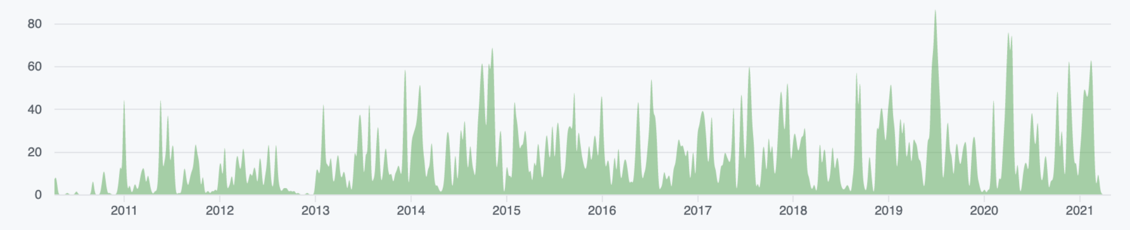
UiO : University of Oslo

HELLENIC REPUBLIC
National and Kapodistrian
University of Athens

UNIVERSITÄT LEIPZIG

ontotext

POLITECNICO MILANO 1863



Ontop downloads

Downloads

51,084

2015-05-09 to 2021-07-11

Countries

Top: **US**, at 15%

Operating Systems

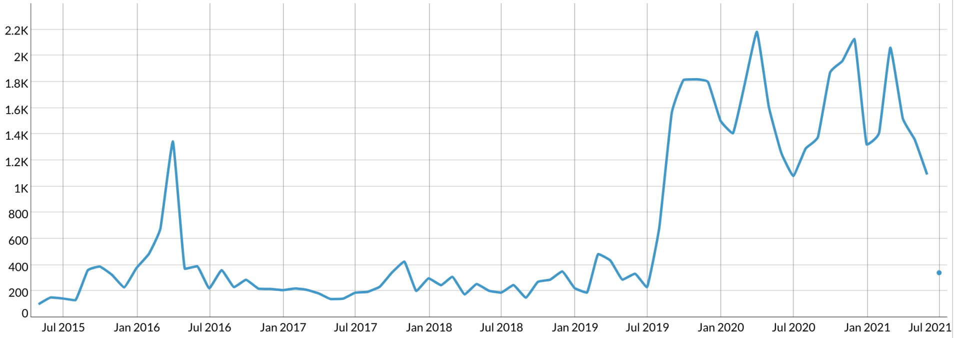
Top: **Other**, at 68%

Download Statistics

All Files (Change File)

Date Range: 2015-05-09 to 2021-07-11

Daily Weekly Monthly



Outline

Conclusions

- VKGs are by now a mature technology to address the challenges related to data access and integration.
- It has been well-investigated and applied in many different scenarios mostly for the case of relational data sources.
- The technology is general purpose, but the bio-medical domain is very well suited for its application.
- Performance and scalability w.r.t. larger datasets (volume), larger and more complex ontologies (variety, veracity), and multiple heterogeneous data sources (variety, volume) is a challenge.
- Recently VKGs have been investigated for alternative types of data, such as **temporal data**, **noSQL** and tree structured data, **linked open data**, and **geo-spatial data**.
- Performance and scalability are even more critical for these more complex domains.

Thank you!

- E: calvanese@inf.unibz.it
- H: `http://www.inf.unibz.it/~calvanese/`

- *Ontop* website: `https://ontop-vkg.org/`
- Github: `http://github.com/ontop/ontop/`
- Facebook: `https://www.facebook.com/obdaontop/`
- Twitter: @ontop4obda
- *Ontopic* website: `https://ontopic.biz/`