Visual Data and Schema Queries over Knowledge Graphs

Sergejs Rikačovs[1], Kārlis Čerāns[1,*]

[1]Institute of Mathematics and Computer Science, University or Latvia, Raiņa bulvāris 29, Rīga, LV-1459, Latvia

Abstract

We demonstrate the concept of presenting a data schema of a knowledge graph, used to support visual schema presentation and visual queries over the data, as a knowledge graph itself. We demonstrate, how the visual schema analysis methods (visual schema presentation and visual query creation) can be applied to the meta-level schema, as well, including creation of queries involving both the data and its schema levels.

Keywords

RDF, Knowledge graph schema, Visual schema diagram, Visual queries

1. Introduction

Visual methods of data analysis, including visual data schema presentation and visual queries over the data hold a promise of enabling direct work with the data for a wider group of custom domain experts, as well as they can ease the work of data professionals by allowing to exploit their visual perception capabilities in their work.

Various methods and tools for visual data structure presentation exist, including OWL ontology visualization tools, as VOWL [1], OntoDia [2] and OWLGrEd [3], as well as RDFShape [4] for RDF data shape presentation. The visual data structure presentation methods exist in commercial knowledge graph management environments, such as Metaphactory and TopBraid Composer, as well. There are also early tools for on-the-fly extracting and visualizing the actual data schema, such as LD-VOWL [5] and LODSight [6].

The tools for visual data exploration and queries involve *OptiqueVQs* [7], *RDF Explorer* [8], *QueryVOWL* [9], *LinDA* [10] and *ViziQuer* [11].

Out of these tools, *ViziQuer* provides an option to combine the data schema presentation and visual query facilities into a single visual environment [12]. Its existing technological pipeline for visual schema presentation and schema-backed query creation is based on a dedicated structure of a data schema that involves the description of the schema entities (classes and properties) and their connections (e.g., what properties can apply to instances of what classes, or what class-to-class and property-to-property relations are possible). The frequencies of entity and their connection appearance and other factors, like property domain/range information and property cardinalities can be relevant elements in the data schema structure, as well.

The current solution enabling the shared visual data schema and visual query experience in the *ViziQuer* tool context foresees storing the data schema in a relational database (RDB). This solves the performance issues, even when the schemas of large data sets, such as *DBpedia* and *Wikidata* are stored. However, RDB storage is little helpful when it comes to explaining what data actually are stored in the database. The RDB format also makes it difficult to share the data schemas among the different users and to integrate their data into larger data ecosystems, typically offered by the knowledge graphs.

In this work we outline the concept of creating a knowledge graph out of the visual data schemas themselves to enable them to be acted upon by various KG and semantic technology tools, including but not limited to the visual schema presentation and visual queries in the *ViziQuer* tool itself. We

EKAW 2024: EKAW 2024 Workshops, Tutorials, Posters and Demos, 24th International Conference on Knowledge Engineering and Knowledge Management (EKAW 2024), November 26-28, 2024, Amsterdam, The Netherlands.

[*] Corresponding author.

sergejs.rikacovs@lumii.lv (S. Rikačovs); karlis.cerans@lumii.lv (K. Čerāns)
00009-0003-8989-0942 (S. Rikačovs); 0000-0002-0154-5294 (K. Čerāns)

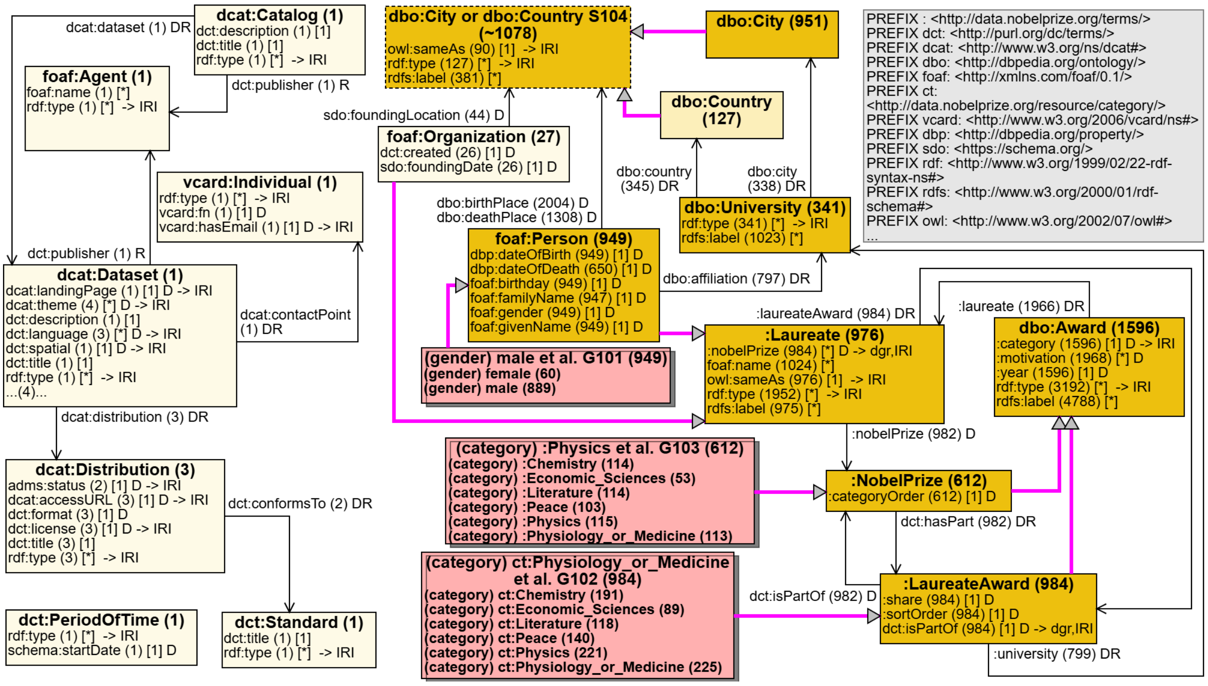

dcat:Catalog (1)
dct:description (1) [1]
dct:title (1) [1]
rdf:type (1) [*] -> IRI
dcat:dataset (1) DR
dct:publisher (1) R
foaf:Agent (1)
foaf:name (1) [*]
rdf:type (1) [*] -> IRI
vcard:Individual (1)
rdf:type (1) [*] -> IRI
vcard:fn (1) [1] D
vcard:hasEmail (1) [1] D -> IRI
dct:publisher (1) R
dcat:Dataset (1)
dcat:landingPage (1) [1] D -> IRI
dcat:theme (4) [*] D -> IRI
dct:description (1) [1]
dct:language (3) [*] D -> IRI
dct:spatial (1) [1] D -> IRI
dct:title (1) [1]
rdf:type (1) [*] -> IRI
...(4)...
dcat:contactPoint (1) DR
dcat:distribution (3) DR
dcat:Distribution (3)
adms:status (2) [1] D -> IRI
dcat:accessURL (3) [1] D -> IRI
dct:format (3) [1] D
dct:license (3) [1] D -> IRI
dct:title (3) [1]
rdf:type (3) [*] -> IRI
dct:conformsTo (2) DR
dct:PeriodOfTime (1)
rdf:type (1) [*] -> IRI
schema:startDate (1) [1] D
dct:Standard (1)
dct:title (1) [1]
rdf:type (1) [*] -> IRI
dbo:City or dbo:Country S104 (~1078)
owl:sameAs (90) [1] -> IRI
rdf:type (127) [*] -> IRI
rdfs:label (381) [*]
sdo:foundingLocation (44) D
foaf:Organization (27)
dct:created (26) [1] D
sdo:foundingDate (26) [1] D
dbo:birthPlace (2004) D
dbo:deathPlace (1308) D
foaf:Person (949)
dbp:dateOfBirth (949) [1] D
dbp:dateOfDeath (650) [1] D
foaf:birthday (949) [1] D
foaf:familyName (947) [1] D
foaf:gender (949) [1] D
foaf:givenName (949) [1] D
(gender) male et al. G101 (949)
(gender) female (60)
(gender) male (889)
dbo:City (951)
dbo:Country (127)
dbo:country (345) DR
dbo:city (338) DR
dbo:University (341)
rdf:type (341) [*] -> IRI
rdfs:label (1023) [*]
dbo:affiliation (797) DR
PREFIX : <http://data.nobelprize.org/terms/>
PREFIX dct: <http://purl.org/dc/terms/>
PREFIX dcat: <http://www.w3.org/ns/dcat#>
PREFIX dbo: <http://dbpedia.org/ontology/>
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
PREFIX ct: <http://data.nobelprize.org/resource/category/>
PREFIX vcard: <http://www.w3.org/2006/vcard/ns#>
PREFIX dbp: <http://dbpedia.org/property/>
PREFIX sdo: <https://schema.org/>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
...
:laureateAward (984) DR
:laureate (1966) DR
:Laureate (976)
:nobelPrize (984) [*] D -> dgr,IRI
foaf:name (1024) [*]
owl:sameAs (976) [1] -> IRI
rdf:type (1952) [*] -> IRI
rdfs:label (975) [*]
dbo:Award (1596)
:category (1596) [1] D -> IRI
:motivation (1968) [*] D
:year (1596) [1] D
rdf:type (3192) [*] -> IRI
rdfs:label (4788) [*]
:nobelPrize (982) D
(category) :Physics et al. G103 (612)
(category) :Chemistry (114)
(category) :Economic_Sciences (53)
(category) :Literature (114)
(category) :Peace (103)
(category) :Physics (115)
(category) :Physiology_or_Medicine (113)
:NobelPrize (612)
:categoryOrder (612) [1] D
dct:hasPart (982) DR
(category) ct:Physiology_or_Medicine et al. G102 (984)
(category) ct:Chemistry (191)
(category) ct:Economic_Sciences (89)
(category) ct:Literature (118)
(category) ct:Peace (140)
(category) ct:Physics (221)
(category) ct:Physiology_or_Medicine (225)
dct:isPartOf (982) D
:LaureateAward (984)
:share (984) [1] D
:sortOrder (984) [1] D
dct:isPartOf (984) [1] D -> dgr,IRI
:university (799) DR

Figure 1: An example Nobel Prizes data schema

demonstrate the possibility of creating distributed visual queries in *ViziQuer* simultaneously over the original data and the data schema, as well.

In what follows, Section 2 reviews the visual schemas and queries in the *ViziQuer* environment context. Section 3 presents the data schemas as Knowledge graphs and Section 4 concludes the paper.

The material supporting the demonstration is available at https://github.com/LUMII-Syslab/viziquer-tools-kg .

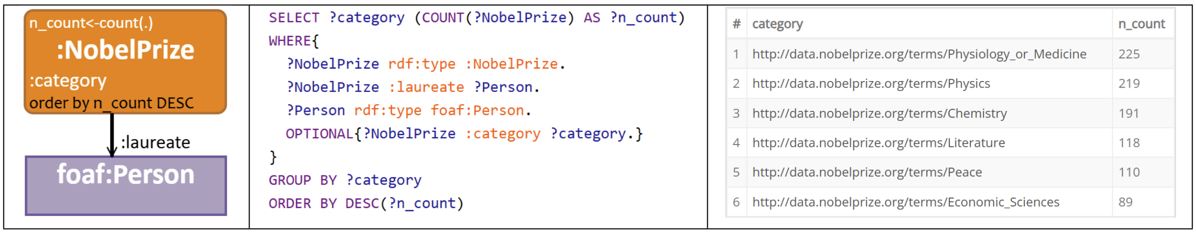
2. Visual Schemas and Queries

Figure 1 provides an example data set schema for a snapshot of the Nobel Prizes data set[1]. The schema lists the data classes as nodes and the properties available at each of them as attributes, or as links connecting the property source and target classes. There is frequency information for classes, attributes (in the context of their ascription class) and links (in the context of link source and target classes), as well as attribute and link cardinalities in their respective contexts. Note that the properties *:hasPart* and *:isPartOf* connect just the *nobel:nobelPrize* and *nobel:LaureateAward* classes, their applicability to the instances of the *dbo:Award* classes is covered by their ascription its subclasses (the schema provides the information about the "essential" ascription points of the properties). The D and R markers at the attributes and links depicting a property inform that the particular property visualization context corresponds to its domain (D) or range (R).

The data schema, as stored in the visual tool, allows creating visual queries for inspecting the data set contents, for instance, by asking for statistics of Nobel Prizes received by organizations across different prize categories, as shown in Figure 2.

A visual query is a rooted directed graph with data nodes corresponding to query variables (with an option to assign class information, as well), the data links establishing their connections and attributes building up the selection list. There are means for further query structuring as control nodes (unit or union) and non-data edges (non-link edges and same-data edges). The reader may consult [13] for a more detailed visual query notation explanation.

[1] Originally at http://data.nobelprize.org; a snapshot has been created created and stored locally (see the supporting resources page for details).



```
SELECT ?category (COUNT(?NobelPrize) AS ?n_count)
WHERE{
  ?NobelPrize rdf:type :NobelPrize.
  ?NobelPrize :laureate ?Person.
  ?Person rdf:type foaf:Person.
  OPTIONAL{?NobelPrize :category ?category.}
}
GROUP BY ?category
ORDER BY DESC(?n_count)
```

#	category	n_count
1	http://data.nobelprize.org/terms/Physiology_or_Medicine	225
2	http://data.nobelprize.org/terms/Physics	219
3	http://data.nobelprize.org/terms/Chemistry	191
4	http://data.nobelprize.org/terms/Literature	118
5	http://data.nobelprize.org/terms/Peace	110
6	http://data.nobelprize.org/terms/Economic_Sciences	89

Figure 2: An example visual query, its translation into SPARQL and results

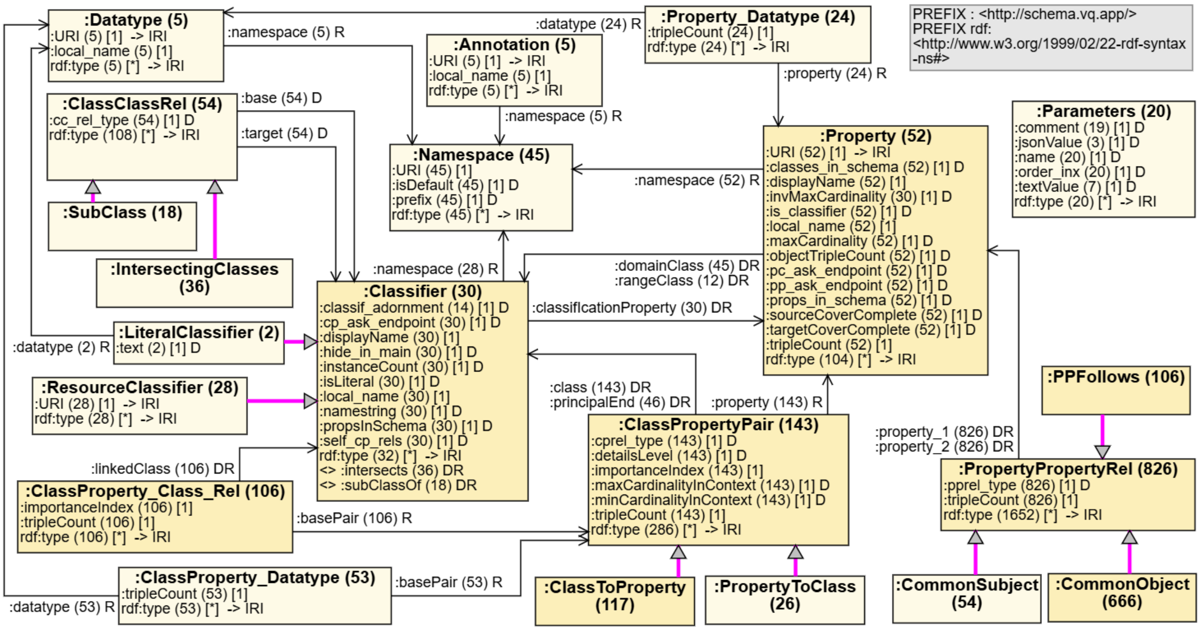


Figure 3: A visual presentation of meta-data schema (with Nobel Prizes schema statistics)

3. Data Schemas as Knowledge Graphs

The data schemas (used for visual schema presentation and visual query support) in the ViziQuer environment[2] are handled by a dedicated open-source Data Shape Server[3] that can be adapted also to serve the data schema information to other clients (e.g., if the context-sensitive auto-completion of class and property names were to be implemented in a textual SPARQL editor). The Data shape server stores the data schemas in a relational *PostgreSQL* database, ensuring their technical availability, paying less attention to the ease of explainability of the schema structure.

For the data schema explanation and their integration into the knowledge graph landscape, the data schemas can be mapped into the Knowledge graph (KG) format. We propose an initial data schema conceptualization structure and manually create custom mappings for generating KG resources (together with their class assignment) from database records of different tables, as well as creating of the data triples in KG from the contents of these records. The mappings are then interpreted by a custom code that creates the actual data triples[4].

Figure 3 contains the visualization of the meta-schema obtained from the analysis of the KG containing the Nobel Prizes data endpoint (the classes in the data schema would be shared for schemas of different endpoints; the entity frequency statistics would be different, though).

The central concepts of the data schema are *:Classifier* for presenting the data classes (and values in

[2] https://github.com/LUMII-Syslab/viziquer, see also https://viziquer.app

[3] https://github.com/LUMII-Syslab/data-shape-server

[4] Should a more standardized approach be required, a R2RML [14] mapping for the data schema KG generation can be created. The ad hoc mapping creation approach has been, however, the easiest approach to solve the practical data mapping task.

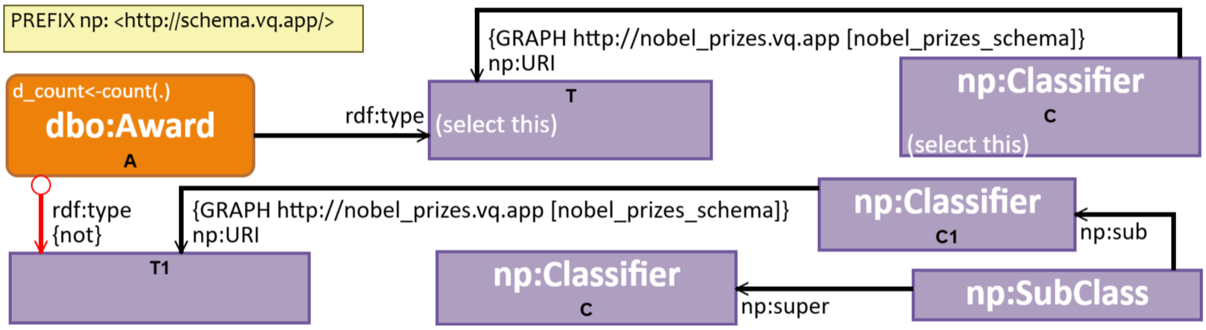


Figure 4: An example of a query over the data and schema instances

other classifiers, if used) and *:Property* for describing the properties. The *:ClassPropertyPair* describes the connections of classes and properties, including what classes can be sources and what classes can be targets for a property. Further on, the class *:CPC_Rel* describes the target classes for a property in a source class context and source classes for a property in a target class context (including the respective frequency/triple counts). The *:ClassClassRel* accounts for subclass relation encoding (other class-to-class relations, as equivalent classes or overlapping classes can be encoded here, as well). The *:PropertyPropertyRel* class describes the patterns of different properties appearing together, either as one property following the other, or two properties having a common subject or common object; these patterns are essential for query auto-completion in a situation when the class information is not available within the query fragment built so far.

We note also the *:importanceIndex* attribute in the *:ClassPropertyPair* and *:ClassProperty_Class_Rel* classes. It informs (if the value is above 0) about when the class is important in the context of a property (or in the context of the property in the context of the "other end" class), so that the attribute or link holding the property name is to be ascribed in the schema diagram at the respective class.

Although the queries involving the meta-data level are available also on the data-level schemas and the *ViziQuer* visual notation has means to support these (using, e.g., the explicit variables in the class and property name positions in a schema diagram, cf. [13], the availability of explicit metadata makes the meta-level queries more convenient. Figure 4 presents an involved multi-graph query asking for the statistics of the instances in the class *dbo:Award* in the *nobel_prizes* data schema, according to their "most specific" classes.

The "most specific" class for an instance *?A* is looked up in the named graph *nobel_prizes_schema* holding the schema information as an instance *?C* of the class *np:Classifier*. The negation condition states that there is no other type *?T1* of the instance *?A* such that it would be a subclass of the classifier instance *?C* in the *nobel_prizes_schema* named graph[5].

The availability of the data schema as a knowledge graph allows creating a wide range of meta-analysis queries over the data. The queries working over meta-schemas of different data sets are possible, as well.

4. Conclusions and Future Work

This work has demonstrated the natural concept of presenting a knowledge graph data schema, as stored in a visual schema and query tool, itself as a data schema. Such a presentation has provided basis of the explanation of the data schema concepts, making it easier re-usable, as well as it allows using the knowledge graph analysis tools over the data schema representation.

The visual query environment allows building queries consuming data from different named graphs, corresponding to different schemas; we have shown an example of creating a query looking simultaneously both into the level of the data themselves, and the data schema.

[5] The global subquery construct is used in the visual notation to enforce the usage of the SPARQL MINUS construct.

A further work would be to expand the visual data schema visualization and visual query software to allow it to work directly with the schema stored as a knowledge graph, as well (without the need to store the schema in a relational database). Such an extension would ease re-using and sharing of schemas of data sets among different users (as the schema would be just a knowledge graph itself).

4.0.1. Acknowledgements

This work has been partially supported by a Latvian Science Council Grant lzp-2021/1-0389 “Visual Queries in Distributed Knowledge Graphs”.

References

[1] S. Lohmann, S. Negru, F. Haag, T. Ertl, Visualizing ontologies with vowl, Semantic Web 7 (2016) 399–419.

[2] D. Mouromtsev, D. Pavlov, Y. Emelyanov, A. Morozov, D. Razdyakonov, M. Galkin, The simple, web-based tool for visualization and sharing of semantic data and ontologies, in: ISWC P&D 2015, CEUR, volume 1486, 2015. URL: http://ceur-ws.org/Vol-1486/paper_77.pdf.

[3] J. Bārzdiņš, K. Čerāns, R. Liepiņš, A. Sproģis, Uml style graphical notation and editor for owl 2, in: Proc. of BIR’2010, LNBIP, volume 64, Springer, 2010, pp. 102–113.

[4] J. E. L. Gayo, D. Fernández-Álvarez, H. Garcıa-González, Rdfshape: An rdf playground based on shapes, in: CEUR Workshop Proceedings, volume 2180, 2018.

[5] M. Weise, S. Lohmann, F. Haag, Ld-vowl: Extracting and visualizing schema information for linked data, in: Voila!2016, 2016, pp. 120–127.

[6] M. Dudáš, V. Svátek, J. Mynarz, Dataset summary visualization with lodsight, in: The 12th Extended Semantic Web Conference (ESWC2015), 2015. URL: http://lod2-dev.vse.cz/lodsight/lodsight-eswc2015-demopaper.pdf.

[7] A. Soylu, E. Kharlamov, D. Zheleznyakov, E. J. Ruiz, M. Giese, M. Skjaeveland, D. Hovland, R. Schlatte, S. Brandt, H. Lie, I. Horrocks, Optiquevqs: a visual query system over ontologies for industry, Semantic Web 9 (2018) 627–660.

[8] H. Vargas, C. Buil-Aranda, A. Hogan, C. López, Rdf explorer: A visual sparql query builder, in: The Semantic Web – ISWC 2019. Lecture Notes in Computer Science, volume 11778, Springer, Cham, 2019.

[9] F. Haag, S. Lohmann, S. Siek, S. Ertl, Queryvowl: Visual composition of sparql queries, in: The Semantic Web: ESWC 2015 Satellite Events, LNCS, volume 9341, Springer, 2015, pp. 62–66. URL: http://vowl.visualdataweb.org/queryvowl/.

[10] K. Thellmann, F. Orlandi, S. Auer, Linda - visualising and exploring linked data, in: SEMANTiCS 2014 (Posters & Demos), 2014, pp. 39–42.

[11] K. Čerāns, A. Šostaks, U. Bojārs, J. Ovčiņņikova, L. Lāce, M. Grasmanis, A. Romāne, A. Sproģis, J. Bārzdiņš, Viziquer: A web-based tool for visual diagrammatic queries over rdf data, in: ESWC 2018 Satellite Events, LNCS, volume 11155, Springer, 2018, pp. 158–163. doi:10.1007/978-3-319-98192-5_30.

[12] K. Čerāns, U. Bojārs, J. Ovčiņņikova, L. Lāce, M. Grasmanis, A. Sprogis, Towards visual federated sparql queries, in: SEMANTiCS-PDWT 2024, CEUR Workshop Proceedings, volume 3759, 2024. URL: https://ceur-ws.org/Vol-3759/paper24.pdf.

[13] J. Ovčiņņikova, A. Šostaks, K. Čerāns, Visual diagrammatic queries in viziquer: Overview and implementation, Baltic Journal of Modern Computing 11 (2023) 317–350. doi:10.22364/bjmc.2023.11.2.07.

[14] R2rml: Rdb to rdf mapping language, https://www.w3.org/TR/r2rml/, 2012.