Lab 3. Knowledge Graphs

Sayyor Yusupov
Tianheng Zhou

B Ontology creation

B.1 TBOX definition

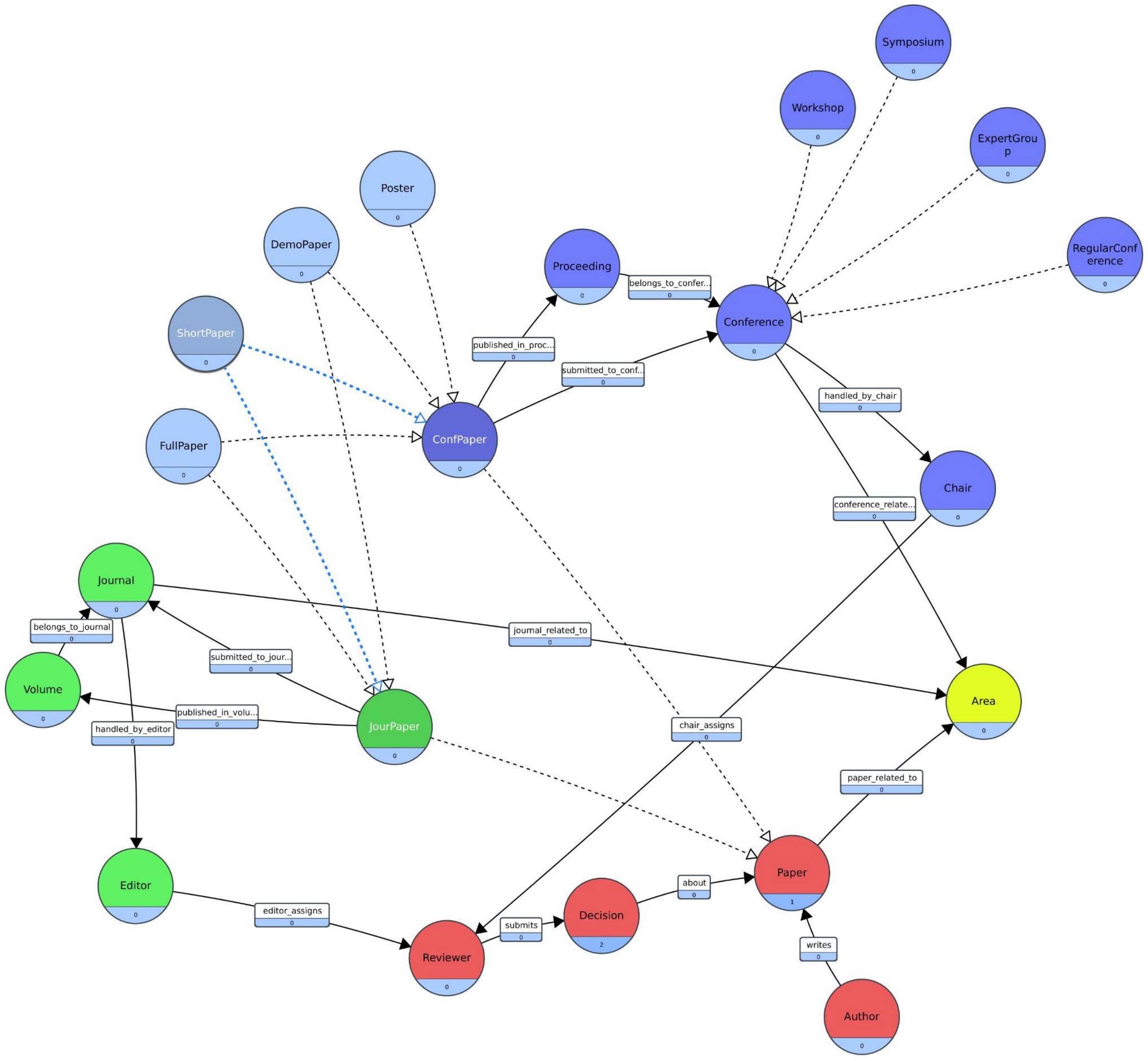


Figure 1. Graphical Representation of TBOX in Gra.Fo

To create the TBOX, we used the knowledge graph visualization tool GraFo (Fig. 1). You can see our graph in more detail, including the nodes' attributes here. By "attributes" Gra.Fo refers to "Resource to Literal" triples that define the characteristics of the resource. We will use this definition of the word "attributes" for the rest of the report. We are listing our design decisions and their justifications as follows.

To generate the tbox file, we export it from Gra.Fo as a .ttl file. We add it as *"BDMA_11C-B1_tbox-ZhouYusupov.ttl"* in our submission.

Decision 1. Replacing edge "Reviews" between "Reviewer" and "Paper" with another node "Decision".
This is an instance of reification. According to the requirements, we need attributes to indicate whether a paper is accepted or rejected ("is_accepted") and "review_text" and they are different for each "Reviews" edge. However, knowledge graphs are fully-normalized and edges cannot have attributes. Therefore, we turn the edge into the node "Decision" and set "is_accepted" and "review_text" as properties of the node "Decision".

Decision 2. Creating nodes "ConfPaper" and "JourPaper" that inherit from the node "Paper".
Three paper types "DemoPaper", "ShortPaper" and "FullPaper" can either belong to a conference or a journal. But "Poster" can only belong to a conference. Therefore, we decided to have a node "Paper" that has basic relations, such as those with nodes "Reviewer", "Author" and "Area". The nodes "ConfPaper" and "JourPaper" are then subclasses of "Paper" and they connect to "Conferences" and "Journals respectively". Then, "Poster" is only a subclass of "ConfPaper", while other paper types are subclasses of both "ConfPaper" and "JourPaper".

Decision 3. Separate Nodes for concepts Author and Reviewer.
Although an author can also be a reviewer, "Reviewer" has different relations and a different semantic meaning and is not necessarily an "Author". Therefore, we set different nodes for each of them and not connect them in any way. We modified our dataset accordingly.

Assumption 1. We assume that the input does not contain posters that are set to journals. We also check this condition in section B2 and return an error otherwise.

For most of the nodes, we have 1 property, as required. But some of them, "ConfPaper" and "JourPaper", do not have properties since they inherit them from "Paper". Several nodes have 2 properties. In particular, here is a list of properties we have for each node and their XSD datatypes:

Journal: StartingYear (integer)
Volume: Number of Pages (integer)
Editor: Age (integer)
Reviewer: Age (integer)
Decision: is_accepted (boolean); Review_Text (string)

Author: Affiliation (string)
Paper: PaperYear (integer)
Area: AreaID (integer)
Chair: Age (integer)
Conference: StartingYear (integer)
Proceeding: Number of Pages (integer)

The titles and names of the concepts (e.g.: Author Name, Paper Title) are usually expressed with rdfs:label, therefore we do not create an additional property for them. Also, we change the name mentioned here to avoid duplication.

B.2 ABOX definition

Before loading the dataset from Lab 1, we had to modify it to fit the schema. We did data preprocessing so that we have all the columns needed for creating the ABOX.

Data Preprocessing

Here are the columns we have in our previous dataset, from Lab 1:

1. "Cite_fake.csv": Each row is a unique paper.
   TitleID (Row ID starting from 0), Authors (Multiple authors in one column), Title (Paper title), Year (The submitted year), Source title (Conference or Journal name), Volume (For both conference and journal), DOI, Affiliations (Multiple authors' affiliation in one column), Authors with affiliations (Combine authors with their affiliations in one column), Abstract (Abstract of the paper), Author Keywords, Keywords (Keywords of the paper in one column), Page, Type (Whether the paper in one row belongs to a conference or a journal), Edition (The edition of the conference or journal), Co-Author (The co-author of a paper), City (The place of the conference), Cite (The paper cites which papers inside the dataset)
2. "Review.csv": Each row is a unique (paper, reviewer) pair. We assume that for each conference or journal, there will always be three reviewers. So the number of rows is three times the number in "Cite_fake.csv".
   Source title, Year, Type, Edition, Authors_list (Python list representation of Authors in Cite_fake file), TitleID, Title, Vote (The reviewer in this row votes for 1 (Agree) or 0 (Disagree)), Approval (The total approval status of one paper, if two out of three or three out of three authors vote for 1 then "Yes", otherwise, "No"), Reviewer (Reviewer name)

In order to make the data look real, here are the transformations we did:

1. Store the optional columns in the Cite_fake file to another file, "Redundant_Columns.csv".
2. Add an Approval column for the Cite_fake file.
3. First, fill in all the NAs in Volume in the Cite_fake file with a reasonable random number; For those rejected papers, set their Volume as NAs; Then divide the Volume into two

columns Proceeding and Volume. The Proceeding will have a value only if this row represents a conference; The Volume will have a value only if this row represents a journal.

4. In the Cite_fake file, specify for each conference whether it is a workshop, symposium, expert group, or regular conference. Add the column “Conference Type”.
5. In the Cite_fake file, add a column “Paper Type”. If the paper in one row belongs to a conference, then draw with a probability distribution from ["Full Paper", "Short Paper", "Demo Paper", "Poster"]; If the paper in one row belongs to a journal, then draw with a probability distribution from ["Full Paper", "Short Paper", "Demo Paper"].
6. In the Cite_fake file, We assume there are two chairs or editors for each conference or journal. We add two columns, Chair_or_Editor1 and Chair_or_Editor2.
7. We change the name of column Keywords in the Cite_fake file to Paper Area. We add areas “Databases”, “Machine learning”, and “Natural language processing” to the previous Paper Area with a probability distribution.
8. We count to get the most frequent keywords (areas) for each conference or journal; then, in these sets, we delete some of the less frequent “Databases”, “Machine learning”, and “Natural language processing”, then we create a separate file Conf_Jour_Area.csv, and make these sets as the area for conference or journal.
9. For the Review file, we create a column “Review_text” for those reviewers' comments. If the vote is 1, then we draw the words from ["Great", "Awesome", "Excellent", "Fantastic", "Sound Work", "Innovative Idea", "Nice Findings"]; If the vote is 0, then we draw the words from ["Need Improvement", "Exist Some Problems", "Not Rigorous Experiment", "Need Major Revision", "Old Idea"].
10. For the Review file, we add a column “Who_Assign”. This is drawn from the Chair_or_Editor1 and Chair_or_Editor2 to determine who assigned this reviewer.
11. Some papers have the same Title, so we added “_NumberXXX” at the end to make the paper title unique in both Cite_fake and Review.

Then, we generate those additional files for satisfying “at least one property per node” requirement:

1. Proceeding_Mapping.csv: Unique proceedings mapping to their number of pages.
2. Volume_Mapping.csv: Unique Volumes mapping to their number of pages.
3. Chair_Age.csv: Unique Chairs mapping to their age.
4. Editor_Age.csv: Unique Editors mapping to their age.
5. Reviewer_Age.csv: Unique Reviewers mapping to their age.
6. Jour_Starting_Year.csv: Unique Journals mapping to their starting year.
7. Conf_Starting_Year.csv: Unique Conferences mapping to their starting year.
8. Area_ID.csv: Unique Areas (A large set of keywords) mapping to their area id.
9. Affiliation_Mapping.csv: Unique Authors mapping to their affiliation.

Finally, these are what we have in the new dataset, beside those additional files:

1. Cite_fake_New.csv: TitleID, Authors, Title, Source title, Proceeding, Volume, Paper Area, Type, Conference Type, Paper Type, Approval, Chair_or_Editor1, Chair_or_Editor2

2. review_New.csv: Source title, Year, Type, Edition, Authors_list, Title, Decision (The previous Vote column, just change the name), Approval, Reviewer, Review_text, Who_Assign
3. Redundant_Columns.csv: Year, DOI, Affiliations, Authors with affiliations, Abstract, Author Keywords, Page, Edition, Co-Author, City, Cite
4. Conf_Jour_Area.csv: Source title, Type, Conf_or_Jour_Area (All areas in one column, separated by semicolons)

The final data satisfied all the constraints that we can come up with in the problem description and the real world. You can see it in our GitHub repository here.

Defining ABOX

We used the Python library RDFLib to create the ABOX. Other libraries used included Pandas to load and preprocess our dataset and IRIBaker that converts strings into IRIs.

We first loaded the TBOX schema to our RDFLib object and set the required Namespaces. We then loaded each .csv file and went through all the rows one by one. We generated triples to both define the concepts and link the concepts together, by using the same names for properties and resources as we did in TBOX.

The dataset for papers includes a column “Approval” to indicate if the paper is approved by the reviewers. To ensure that this information is correct and the resources “Paper” link with resources “Proceeding” or “Volume” only when the decision is positive, we are confirming it with reviewers' decisions in the file with reviews “review_New.csv”. If they do not match, we return an exception.

We had to define only the lowest subclass of a concept. For example, for a paper that is a demo paper and in a conference, we only defined a node “Demo Paper” and connected it with a “Conference” node. The fact that it is also a “ConfPaper” node and a “Paper” node could be inferred.

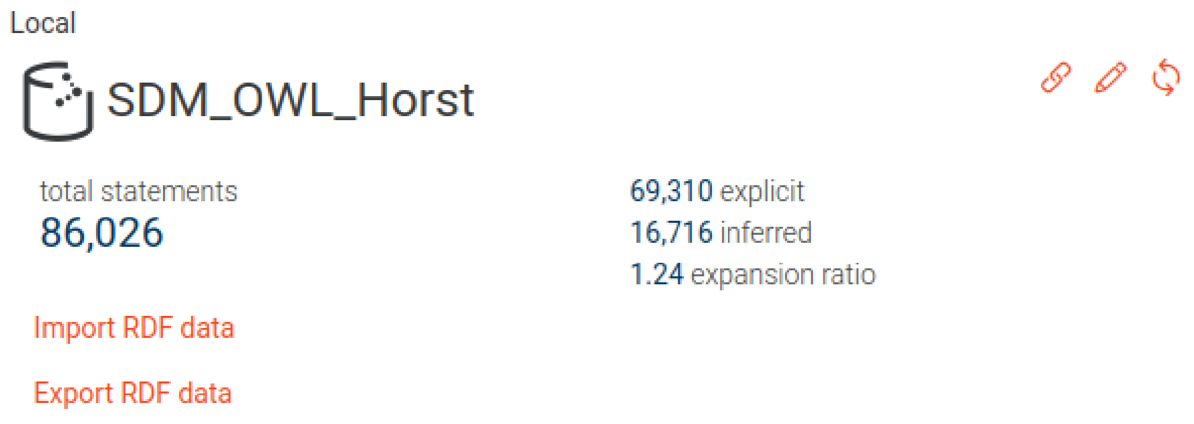
Finally, we saved the ABOX as .ttl file.

The code for loading ABOX can be seen in *“BDMA_11C-B2-ZhouYusupov.py”* and our final abox is stored in *“BDMA_11C-B2_abox-ZhouYusupov.ttl”*.

B.3 Create the final ontology

As mentioned in B2, we loaded TBOX into the graph before setting up ABOX using the function dataset.default_context.parse(). Also, we manually set the same Namespaces for ABOX and defined the rdfs:type for each resource. Therefore, we did not have to further link TBOX with ABOX. The definition of types and Namespaces can be seen in the code for loading ABOX. With this setup, we could successfully execute all the queries in section B4. And there is no problem running queries across the ABOX and TBOX, after our testing.

We tried two inference regime entailments for our dataset, *“RDFS-Plus (Optimized)”* and *OWL-Horst (Optimized)*. For OWL-Horst, we did not have to define the types of nodes for the query below to run. It inferred this information from the properties used. 16,716 statements were inferred out of a total of 86,026 statements (Figure 2).



Local

SDM_OWL_Horst

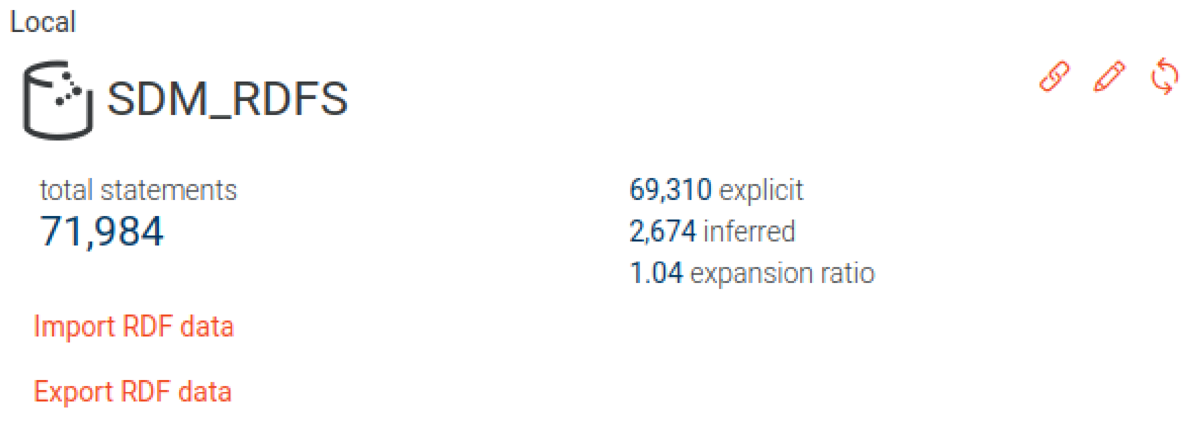
total statements
86,026

69,310 explicit
16,716 inferred
1.24 expansion ratio

Figure 2. Statistics of the repository from GraphDB.

```
PREFIX gf: <http://www.gra.fo/schema/untitled-ekg#>
SELECT ?s ?p WHERE {
    {?s ?p gf:Author}
}
```

However with RDFS-Plus, this query returns nothing if the types are not defined, but returns all the nodes after defining them. With types defined, 2,674 statements were inferred by the system out of a total of 71,984 statements (Figure 3).



Local

SDM_RDFS

total statements
71,984

69,310 explicit
2,674 inferred
1.04 expansion ratio

Figure 3. Statistics of the repository from GraphDB.

Summary Statistics
We used SPARQL to retrieve the statistics.

Statistic	Value
Number of classes	32
Number of properties	39
Number of instances	13926
Number of triples	71979

The Number of instances and their percentage of the parent class of different types of concepts:

Class	Number of Instances	Percentage of the Parent Class (%)
Paper	890	100
Poster	188	21
Demopaper	216	24
Shortpaper	193	22
FullPaper	293	33
Paper in Conference	638	72
Paper in Journal	252	28
Conference	5	100
Workshop	1	20
Symposium	1	20
Expertgroup	1	20
Regularconference	2	40
Proceeding	238	100
Chair	10	100
Journal	4	100
Volume	143	100

Editor	8	100
Reviewer	11	100
Author	2275	100
Area	7695	100

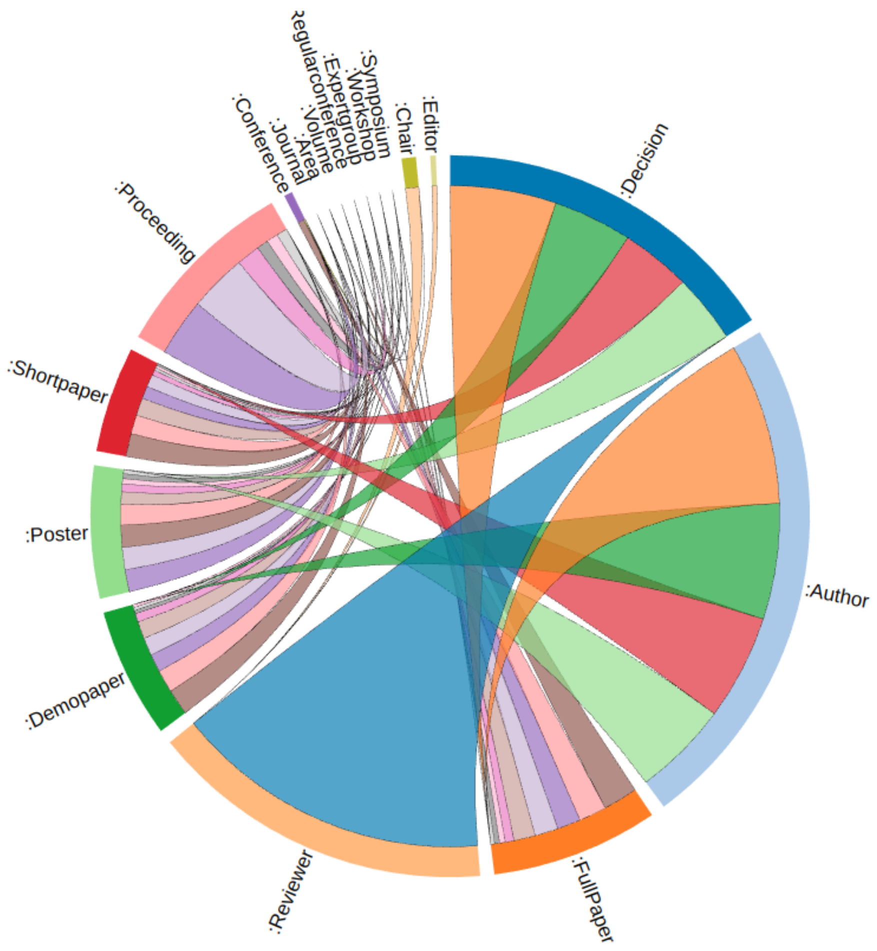
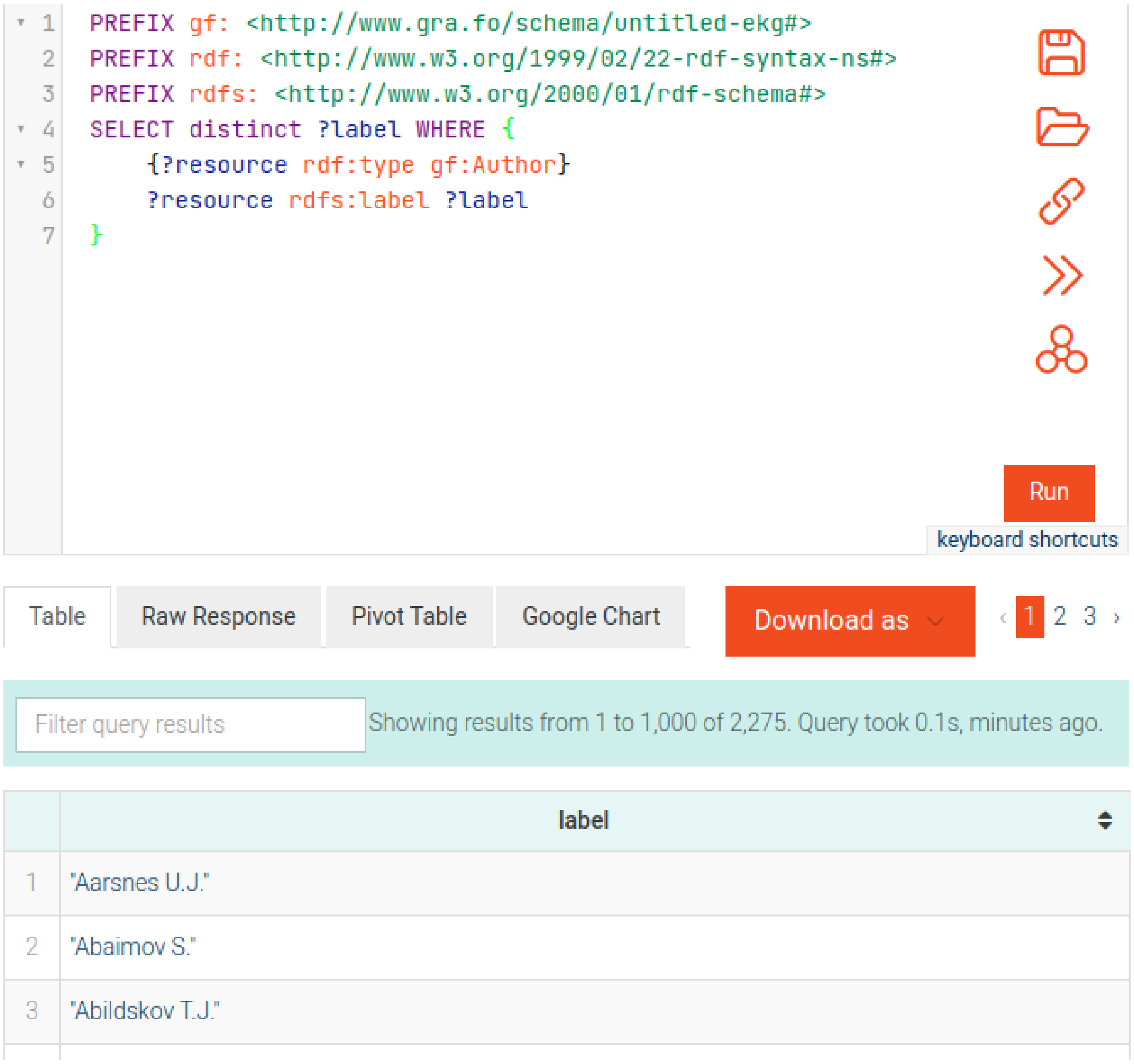


Figure 4. Edges between concepts.

B.4 Querying the ontology

The resulting queries can also be seen in textual form in our GitHub repo here.

1. Find all Authors.



```
PREFIX gf: <http://www.gra.fo/schema/untitled-ekg#>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
SELECT distinct ?label WHERE {
    {?resource rdf:type gf:Author}
    ?resource rdfs:label ?label
}
```

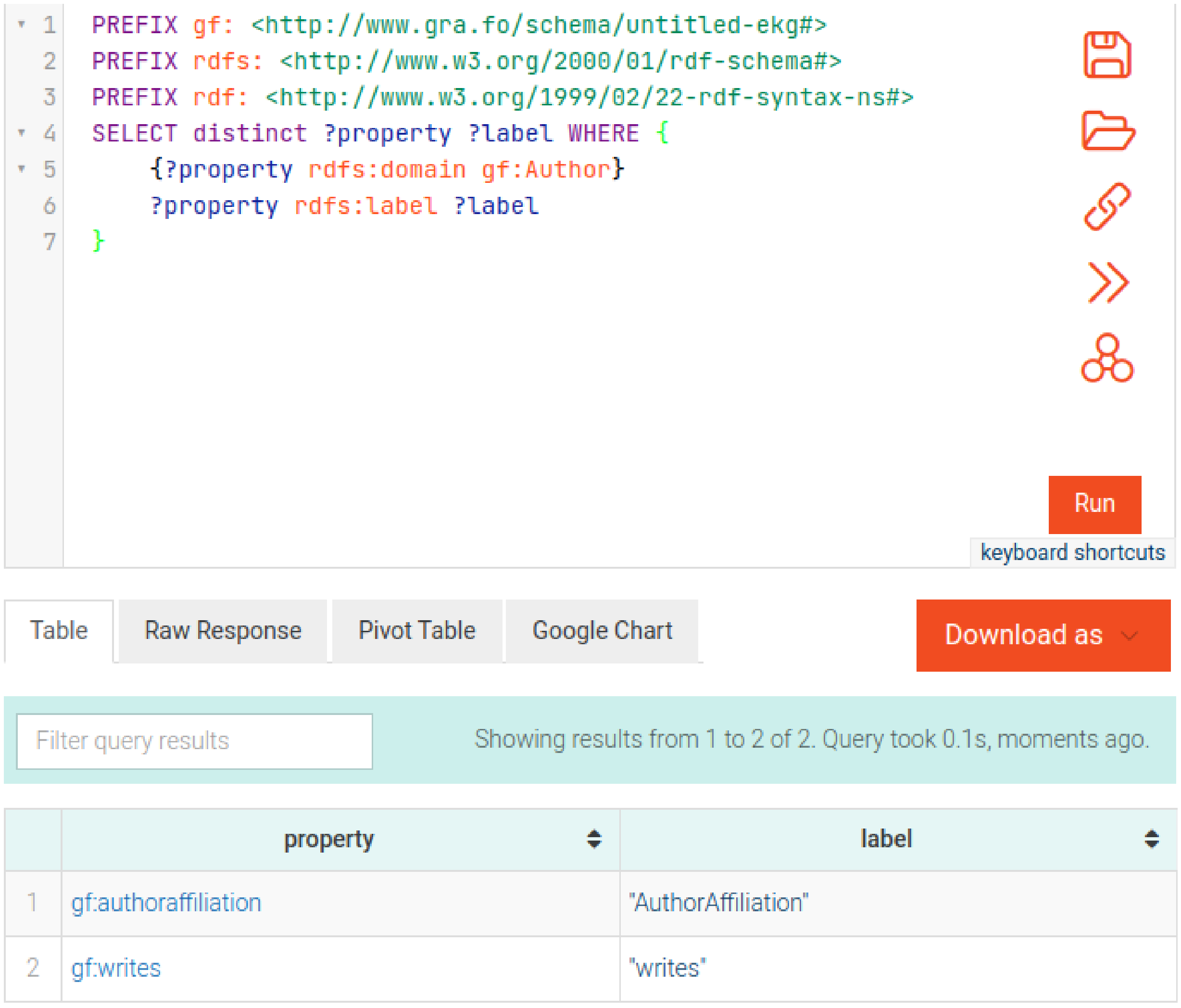
Showing results from 1 to 1,000 of 2,275. Query took 0.1s, minutes ago.

	label
1	"Aarsnes U.J."
2	"Abaimov S."
3	"Abildskov T.J."

Figure 5. Query 1 on GraphDB.

We used *rdf:type* to find Authors and then returned their labels.

2. Find all properties whose domain is Author.



```
PREFIX gf: <http://www.gra.fo/schema/untitled-ekg#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
SELECT distinct ?property ?label WHERE {
    {?property rdfs:domain gf:Author}
    ?property rdfs:label ?label
}
```

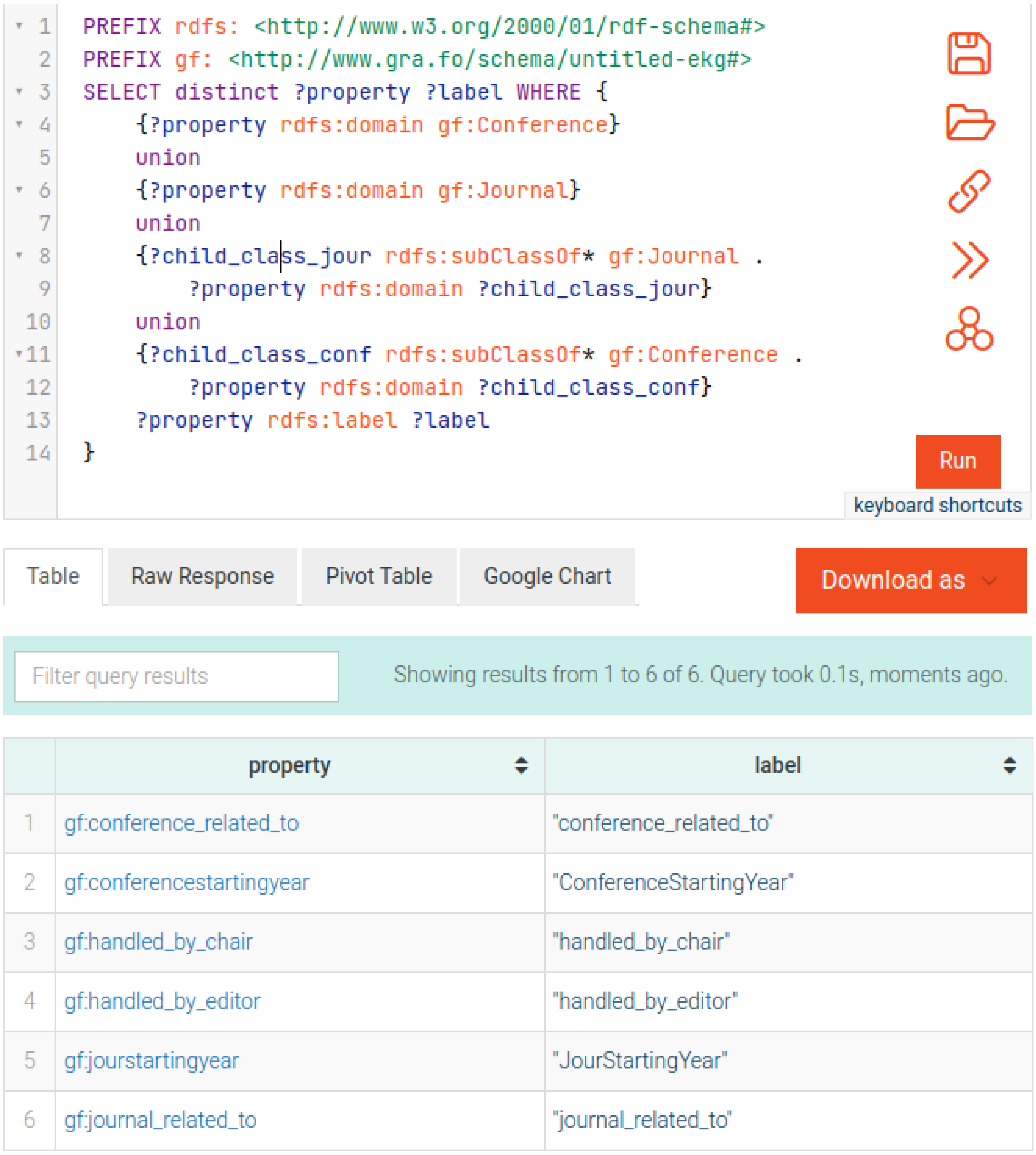
Showing results from 1 to 2 of 2. Query took 0.1s, moments ago.

	property	label
1	gf:authoraffiliation	"AuthorAffiliation"
2	gf:writes	"writes"

Figure 6. Query 2 on GraphDB.

This query is also straightforward. Since “Author” has no parent classes, we only need to find domains of the node “Author”.

3. Find all properties whose domain is either Conference or Journal.



```
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX gf: <http://www.gra.fo/schema/untitled-ekg#>
SELECT distinct ?property ?label WHERE {
    {?property rdfs:domain gf:Conference}
    union
    {?property rdfs:domain gf:Journal}
    union
    {?child_class_jour rdfs:subClassOf* gf:Journal .
        ?property rdfs:domain ?child_class_jour}
    union
    {?child_class_conf rdfs:subClassOf* gf:Conference .
        ?property rdfs:domain ?child_class_conf}
    ?property rdfs:label ?label
}
```

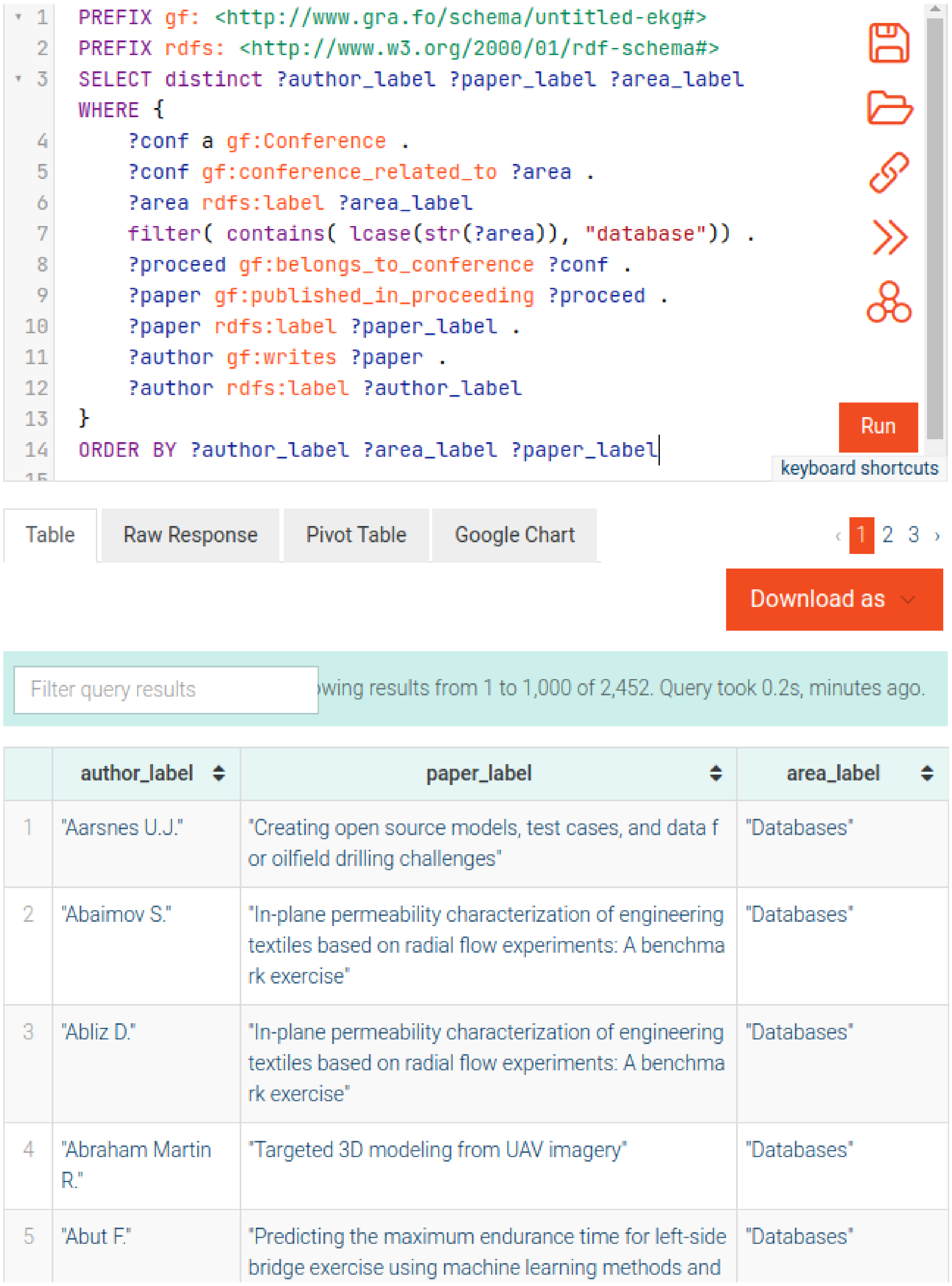
Showing results from 1 to 6 of 6. Query took 0.1s, moments ago.

	property	label
1	gf:conference_related_to	"conference_related_to"
2	gf:conferencestartingyear	"ConferenceStartingYear"
3	gf:handled_by_chair	"handled_by_chair"
4	gf:handled_by_editor	"handled_by_editor"
5	gf:jourstartingyear	"JourStartingYear"
6	gf:journal_related_to	"journal_related_to"

Figure 7. Query 3 on GraphDB.

First, we retrieved all properties of Conference and Journals. Then, we retrieved any existing properties of subclasses of “Conference” and “Journal” nodes. Even though the subclasses do not have unique properties, we would check them as well for other repositories.

4. Find all the papers written by a given author that were published in database conferences.



```
PREFIX gf: <http://www.gra.fo/schema/untitled-ekg#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
SELECT distinct ?author_label ?paper_label ?area_label
WHERE {
    ?conf a gf:Conference .
    ?conf gf:conference_related_to ?area .
    ?area rdfs:label ?area_label
    filter( contains( lcase(str(?area)), "database")) .
    ?proceed gf:belongs_to_conference ?conf .
    ?paper gf:published_in_proceeding ?proceed .
    ?paper rdfs:label ?paper_label .
    ?author gf:writes ?paper .
    ?author rdfs:label ?author_label
}
ORDER BY ?author_label ?area_label ?paper_label
```

wing results from 1 to 1,000 of 2,452. Query took 0.2s, minutes ago.

	author_label	paper_label	area_label
1	"Aarsnes U.J."	"Creating open source models, test cases, and data f or oilfield drilling challenges"	"Databases"
2	"Abaimov S."	"In-plane permeability characterization of engineering textiles based on radial flow experiments: A benchma rk exercise"	"Databases"
3	"Abliz D."	"In-plane permeability characterization of engineering textiles based on radial flow experiments: A benchma rk exercise"	"Databases"
4	"Abraham Martin R."	"Targeted 3D modeling from UAV imagery"	"Databases"
5	"Abut F."	"Predicting the maximum endurance time for left-side bridge exercise using machine learning methods and	"Databases"

Figure 8. Query 4 on GraphDB.

First, we retrieved all resources “Conference” and “Area” that are linked to them. Then we filtered the areas to check if they contained the string “database” in them. To ensure that all the right areas matched, we put names of the Area resources into lowercase. After that, we found all papers that were published in the conferences through Proceedings and also retrieved all authors who wrote those papers. For the nodes Author, Area, Paper we also retrieved their labels and grouped by the results by Author and then by Paper Title and Area, to get papers of all authors in one place.