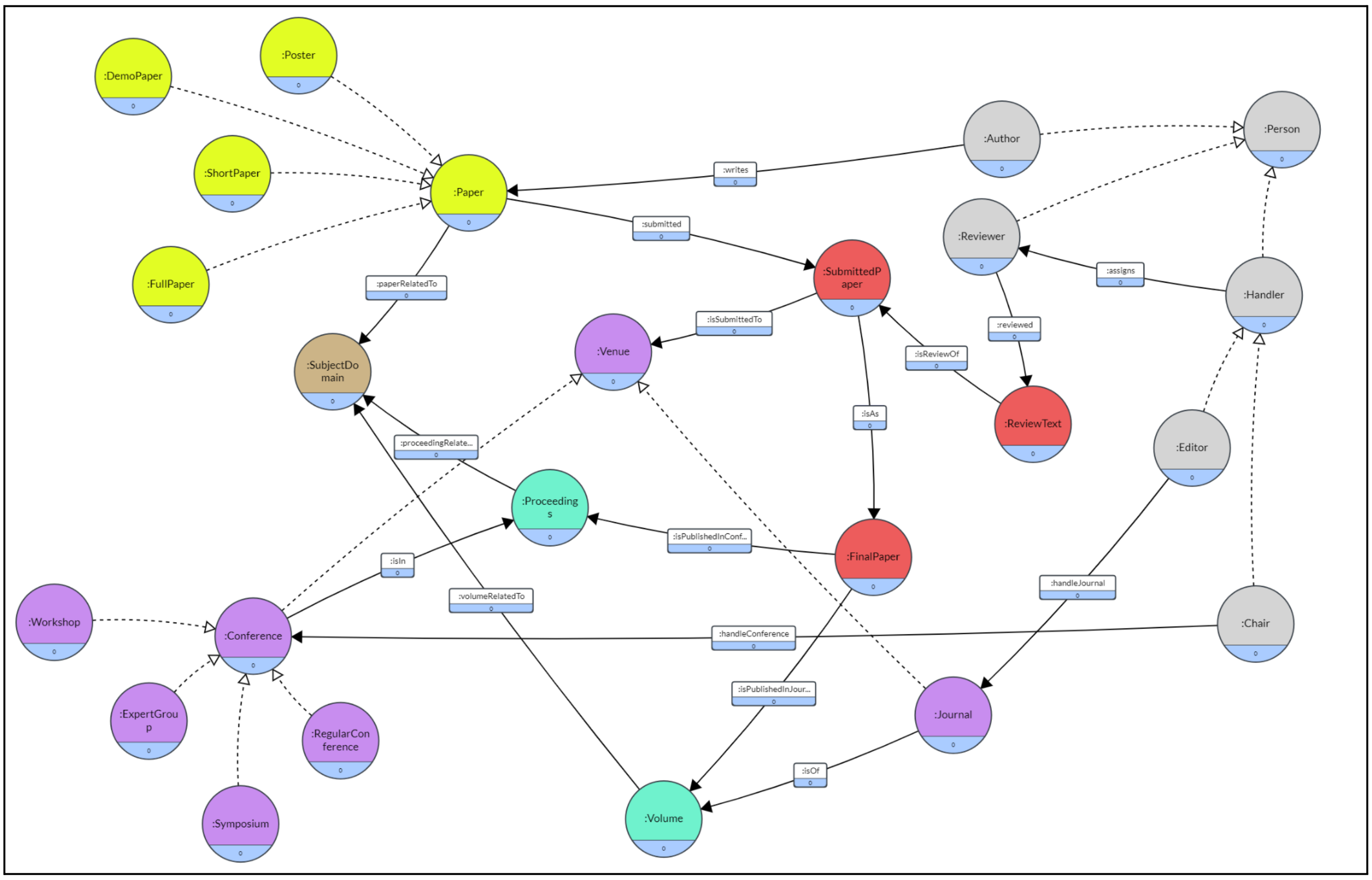
B. ONTOLOGY CREATION

B1. TBOX Definition



Note: Please find a clearer representation of our graph here.

This TBOX was created using Python's RDFS library.

For modeling the concepts of paper, author, publication and review, we have considered the following attributes other than the triples in which each one participates:

Belongs to the concept (Class)	**Attribute name (Property)**	**rdfs: domain**	**rdfs: range[1]**	**Our comments**
Paper	1) paperAbstract	lab:Paper	xsd:string	Paper abstract and title are some attributes (as for any paper in the normal world, with which a paper can be identified. Note that, for the purpose of this lab, we have limited to these attributes only, but we can use doi, url, etc.
	2) paperTitle	lab:Paper	xsd:string	
Author	1) authorName	lab:Author	xsd:string	Similar to paper, author attributes are restricted to only authorName, but we can add age, date of birth, university, etc.
Review	1) comment	lab:ReviewText	xsd:string	For this lab, the default value of 'content of the reviewed text goes here' is assumed for each review. We can generate different reviews using

				the faker library of Python.
	2) decision	lab:ReviewText	xsd:boolean	We have assumed that the decisions (accepted or rejected) are assigned decisions randomly with weights of 0.8 (accept) and 0.2 (reject) to each submittedPaper.
Publication	1) paperAcceptanceDate	lab:FinalPaper	xsd:date	Since publication is basically accepted papers (in conferences / journals) and in our case we have considered the concept of FinalPaper as Publication, the attribute considered for the same is paperAcceptanceDate. For this, we have extracted the year of proceedings and volume (from proceedingYear and volumeYear respectively) and an acceptance date is generated as a random in that particular year of publication.
Related Areas	1) keywords	lab:SubjectDomain	xsd:string	In our model, we have assumed this as the SubjectDomain Class and keywords have been assigned accordingly.
Journal	1) journalTitle	lab:Journal	xsd:string	For this lab, we have considered only one attribute for journal and conference i.e. title.
Conference	1) conferenceTitle	lab:Conference	xsd:string	
Proceedings	1) proceedingName	lab:Proceedings	xsd:string	For this lab, proceedings have been assumed to have 2 attributes: name and year, wherein years are assigned any year between 2001 and 2022.
	2) proceedingYear	lab:Proceedings	xsd:int	
Volumes	1) volumeName	lab:Volume	xsd:string	Similarly, for volume we have assumed 2 attributes: name and year, wherein years are assigned any year between 2001 and 2022
	2) volumeYear	lab:Volume	xsd:int	

Some assumptions related to the TBOX modeling are mentioned below:

- All Papers are considered Submitted Papers. Therefore, the number of instances for both classes is the same. However, the number of Final Papers is smaller, since not all of them are accepted. Note that, the rejected papers don't proceed to any conference proceedings or journal volumes.
- Since posters (a type of Paper Class) can only be submitted to conferences, it is assumed that the data obtained for papers are submitted to conferences, they are of type Poster and if submitted to journals they can be of any type - shortPaper, demoPaper or fullPaper, hence no constraints added on TBOX/ABOX (This constraint can be viewed in the preprocessing file here).
- Different types of conferences - workshops, symposiums, expert groups and regular conferences are considered as subclasses of the Conference Class. These types of conferences are randomly assigned while generating data (as in the preprocessing file) and later handled with conditional statements while connecting TBOX with ABOX in Part B.3.
- A venue can be either a conference or a journal, hence modeled as the main concept (super class) with conference and journal as their subclasses.
- Properties: venueRelatedTo, paperRelatedTo and publicationRelatedTo are created to connect conferences, journals, papers and/or publications respectively to the concept of SubjectDomain.

- Since an edition of a Conference or a Journal where a Paper is published is related to a SubjectDomain, we connect Proceedings with SubjectDomain by ProceedingRelatedTo, and Volume to SubjectDomain by volumeRelatedTo.

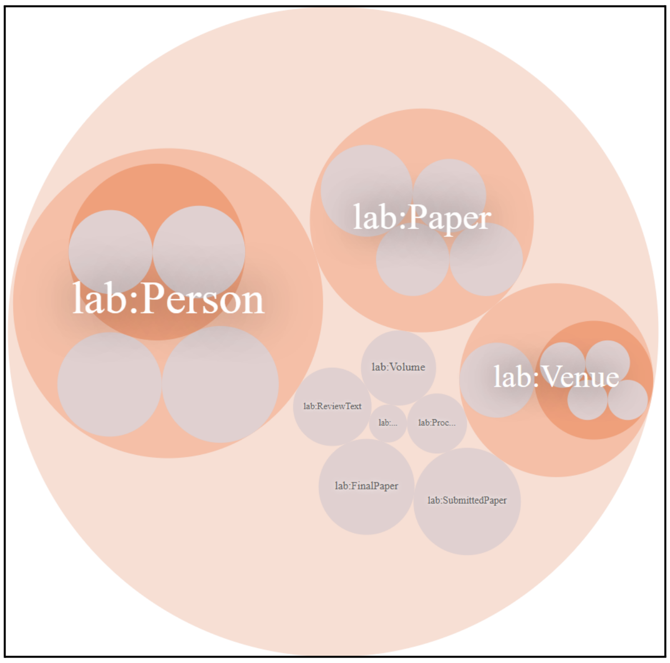
- Editors handle conferences while chairs handle journals and since both of these are responsible to assign reviewers to papers, they can be modeled as a subclass of Handler (which is a subclass of Person) and in this way Handler (either editors/chairs) assign reviewers to the submitted papers. The total number of reviewers per paper is set to be 2 as mentioned in the data preprocessing stage found here.

- Since in rdfs properties cannot have the same name, conferences are linked to proceedings using :isIn while journals are linked to volumes using :isOf, but both try to model the semantic that the published papers from conferences are collected together to form proceedings while that from journal form volumes.

- All the papers are submitted to only conferences and journals, nowhere other than that.

- ConferenceProceedings are unique, for each conference a unique proceeding is present. Same is the case for JournalVolumes.

- In our case, we have modeled 3 subclasses of Person Class: Reviewer, Author and Handler. Also, reviewer and author classes don't have any subclasses.



B2. ABOX Definition

Some assumptions and details related to the ABOX modeling are mentioned below:

- For this laboratory, inference is on (Some triples are still stated explicitly).

- For the Person hierarchy, the reviewers and handlers (editors and chairs) are chosen from the author instances as shown in the preprocessing file here. Also, for this lab 2 reviewers are selected for each paper from the set of authors (who have not authored the paper) from the dataset, like in the real world (Reviewers are generally people who have many papers referenced and published in top conferences or journals).

- We have not assumed a majority vote in case of 2 reviewers for each paper. Instead, both reviewers of every paper provide the same decision either accepted or rejected.
- A venue can be either a conference or a journal, hence modeled as the main concept (super class) with conference and journal as their subclasses. For this lab, we have assumed only instances of conference and journal are created. Thanks to inference, when viewing the instances for venue we can inherit from the subclasses of conference and journal.
- We have created ids like conferenceId (which has data like c0, c1, etc.) for classes (in this case - conference) that makes it simpler and unique to add to the URIs when creating Knowledge Graphs. Additionally, it is also used to join when required during preparation of data.
- Note that further comments and details related to data creation and preprocessing have been added in respective files as present on GitHub.

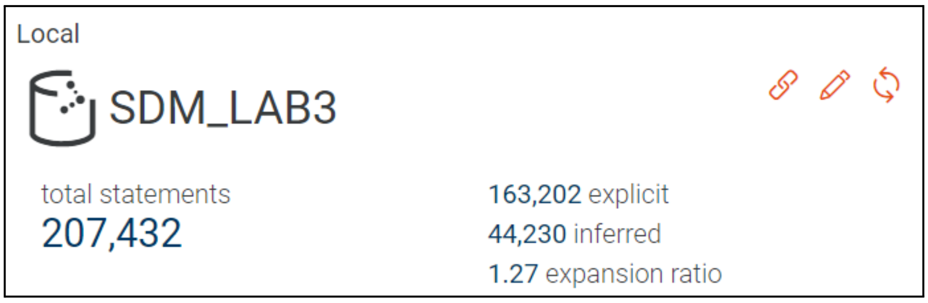
B3. Create the final ontology

To define the ontology and connect ABOX and TBOX we use the same tools of RDFlib as for ABOX creation. With the *graph.add()* function we add instances by iterating over the datasets and in every iteration a triple is passed. *Type* from the RDF namespace indicates that an added instance is a type of a class from namespace LAB. For example, adding submittedPapers:

```
g.add((URIRef(LAB+submittedPapersDF['paperId'][k]), RDF.type, LAB.SubmittedPaper))
```

In this part, we only connect *RDF.type* to pure classes, i.e. the classes that don't have subclasses (like SubjectDomain, Author) and subClasses (like Poster, ShortPaper, DemoPaper, FullPaper for Paper class). Literals are not part of this section, since they are defined during the ABOX creation.

Note that for this laboratory, we have used RDFS-Plus (Optimized) entailment regiment with inference as on.


Local
SDM_LAB3
total statements
207,432
163,202 explicit
44,230 inferred
1.27 expansion ratio

Summary Statistics

The below table shows the statistics of the graph obtained after linking TBOX and ABOX:

Measure:	Total number of Classes	Total number of Properties	Total number of Instances	Total number of Triples
Value:	24	34	30643	163202

Also, the below tables mention the total number of instances by classes:

Class:	Person	Author	Reviewer	Handler	Editor	Chair	Volume	Proceedings	SubjectDomain
Total instances:	10684	10684	3984	2229	790	1555	384	115	19

Class:	Paper	SubmittedPaper	FinalPaper	Poster	FullPaper	DemoPaper	ShortPaper	ReviewText
Total instances:	2500	5000	2015	1468	343	335	338	499

Class:	Venue	Conference	Symposium	Workshop	ExpertGroup	RegularConference	Journal
Total instances:	499	115	23	23	36	33	384

Note that you can further find the statistical queries used for the same here in our GitHub repo.

B4. Querying the ontology

1. Find all authors

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX lab: <http://SDM_LAB3.org/>

SELECT ?author ?authorName
WHERE
{
   ?author rdf:type lab:Author .
   ?author lab:authorName ?authorName .
}
```

As mentioned in TBOX and ABOX creation, we have “Author” rdfs:Class, so finding all authors was pretty straight forward; just search all data instances which have the rdf:type of “Author”. In this query, we have also obtained authorName. You can find the query and output.

2. Find all properties whose domain is Author

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX lab: <http://SDM_LAB3.org/>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>

SELECT ?propertyName
WHERE
{
   ?propertyName rdfs:domain lab:Author .
}
```

As detailed in TBOX creation, we have the “Author” rdfs:Class and there are properties associated with the Class namely - authorName and writes and in this query we are finding these properties (since Author is the domain). Note that, in our TBOX (as defined before), we don’t have any subclasses of Author, hence we don’t obtain any properties of Author (as if it were the superClass). You can find the query and output.

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

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX lab: <http://SDM_LAB3.org/>

SELECT ?propertyName
WHERE
{
   {?propertyName rdfs:domain lab:Conference}
   UNION
   {?propertyName rdfs:domain lab:Journal}
}
```

Similar to query 2, “Conference” and “Journal” classes don’t have any properties wherein they are in the domain as a superClass, hence only direct properties’ domain is found. You can find the query and output.

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

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX lab: <http://SDM_LAB3.org/>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>

SELECT (?pTitle as ?paper_title) (?procName as ?proceeding_name)
WHERE
{
   ?paper     rdf:type       lab:Paper ;
           lab:submitted   ?submittedPaper;
           lab:paperTitle  ?pTitle .

   ?author     rdf:type       lab:Author ;
           lab:authorName   "C. Groom" ;
           lab:writes      ?paper .

   ?submittedPaper rdf:type    lab:SubmittedPaper ;
              lab:isAs    ?finalPaper .

   ?finalPaper lab:isPublishedInConference    ?proceedings .

   ?proceedings  lab:proceedingName ?procName ;
             lab:proceedingRelatedTo   ?subject.

   ?subject rdf:type lab:SubjectDomain .

   ?subject lab:keywords "Database" .
}
```

For this query, the given author is chosen as “C. Groom”. Firstly, we are finding the papers written by this author, then further narrowing it down to finding submitted papers and final papers (accepted in conference proceedings) wherein the conference proceedings are related to the subject domain of Database. *Assumption*: *For this question, papers are supposed to be from a conference of “databases”. With respect to our dataset, the subject domain of the database is modeled as Database (first letter as capital), hence the query fetches papers relevant to “Databases”.* You can find the query and output.

References:

[1] https://www.w3.org/2011/rdf-wg/wiki/XSD_Datatypes
[2] Our GitHub repository: https://github.com/risg99/SDM_Lab_3