Knowledge Representation and Reasoning

Course Project Report

PKG2020 Knowledge Graph

A Linked Data Approach to Bibliometric Research Data

Team Members

Zain Ul Abdeen	2023773
Omer Khan	2023578
Muhammad Umar	2023535

Instructor

Dr. Khurram Jadoon

Fall 2025

December 24, 2025

Contents

1 Introduction to the Domain 4
1.1 Domain Overview 4
1.2 Motivation 4
1.3 Target Application Use Cases 4

2 Dataset Description 5
2.1 Data Source 5
2.2 Key Data Fields 5
2.3 Evidence of Non-RDF Status 5

3 Competency Questions 5

4 Conceptual Model 6
4.1 Conceptual Model Diagram 6
4.2 T-Box Schema 6

5 Ontology Design 7
5.1 Classes (20+ as Required) 7
5.2 Enumeration Class 8
5.3 Cardinality Restrictions 8
5.4 Intersection, Union, and Complement Classes 8
5.5 Object Properties 9
5.6 Data Properties 9
5.7 Functional and Inverse Functional Properties 9

6 Graph Generation using Python 10
6.1 Tools Used 10
6.2 Pipeline Scripts 10
6.3 Sample Code: Populating Authors 10

7 External Linking (5-Star Linked Data) 11
7.1 Linking Strategy 11
7.2 Linking Implementation 11
7.3 Proof: Dataset Not Previously Available as Linked Data 11

8 Reasoning Scenarios 13
8.1 Reasoning Implementation 13
8.2 Reasoning Results 13
8.3 SWRL Rules 14

9 Hand-Annotated Individuals 14
9.1 Reasoning Verification 14

10 SPARQL Queries 15
10.1 Author Queries (CQ1-CQ3) 15
10.1.1 CQ1: Authors with Multiple Institutions 15
10.1.2 CQ2: Most Prolific Authors 15
10.1.3 CQ3: Author Collaboration Network 16

10.2 Article & Bio-Entity Queries (CQ4-CQ7) 16
10.2.1 CQ4: Articles Mentioning Genes 16
10.2.2 CQ5: Articles Mentioning Species 16
10.2.3 CQ6: Gene-Mutation Correlations 16
10.2.4 CQ7: Bio-Entity Type Distribution 17
10.3 Organization & Affiliation Queries (CQ8-CQ9) 17
10.3.1 CQ8: Top Organizations by Author Count 17
10.3.2 CQ9: Affiliations by Country 17
10.4 Employment & Education Queries (CQ10-CQ12) 18
10.4.1 CQ10: Top Education Institutions 18
10.4.2 CQ11: Employment Timeline 18
10.4.3 CQ12: Authors with Education Records 18
10.5 NIH Project Queries (CQ13-CQ14) 19
10.5.1 CQ13: Authors with NIH Funding 19
10.5.2 CQ14: Principal Investigators 19
10.6 Complex Analytical Query (CQ15) 19
10.6.1 CQ15: Complete Author Profile 19
10.7 Federated Query to External Knowledge Bases 20

11 Web Application **20**
11.1 Application Overview 20
11.2 Entity Types Searchable 20
11.3 Running the Application 21

12 Results and Statistics **21**
12.1 Generated Data 21
12.2 OWL Files Generated 21
12.3 Ontology Files Summary 22

13 Visualization **22**
13.1 Complete Ontology Schema (All 23 Classes) 22
13.2 RDF Triple Pattern (Subject-Predicate-Object) 22
13.3 Sample Knowledge Graph Visualization 23
13.4 Knowledge Graph Statistics 23
13.5 Tools Used 23
13.6 Web Application Features 24
13.7 Live GraphDB Endpoint 24

14 Reflection **24**
14.1 Learning Outcomes 24
14.2 Added Value of Linked Data 25
14.3 Challenges Faced 25

15 Conclusion **25**

16 References **25**

17 Appendix: Project Repository **26**
17.1 Repository Structure 26

List of Figures

1 Conceptual Model of PKG2020 Ontology 6
2 T-Box: Class Hierarchy and Object Properties 7
3 Wikidata Search: No results for PKG2020 dataset 12
4 DBpedia Databus Search: PKG2020 not found 12
5 Linked Open Data Cloud: Dataset not registered 13
6 Complete PKG2020 Ontology Schema with 23 Classes and Object Properties 22
7 RDF Triple Pattern Examples: Subject → Predicate → Object 22
8 Sample Knowledge Graph: Article with Authors, Bio-Entities, and Affiliations 23
9 Distribution of Entity Types in Knowledge Graph (2.1M+ triples) 23
10 Web Application Architecture 24

List of Tables

1 Dataset Files Overview 5
2 15 Competency Questions for PKG2020 Ontology 6
3 Ontology Classes (T-Box) 7
4 Object Properties 9
5 Data Properties 9
6 Python Scripts Pipeline 10
7 Hand-Annotated Individuals 14
8 Ontology Statistics 21
9 Generated OWL Files 21

1 Introduction to the Domain

1.1 Domain Overview

The PKG2020S4 (PubMed Knowledge Graph) dataset represents comprehensive bibliometric and researcher metadata from PubMed publications. This domain encompasses:

- **Research Publications**: Articles identified by PubMed IDs (PMIDs)
- **Researchers/Authors**: Identified by unique AND_IDs
- **Organizational Affiliations**: Universities, research institutions
- **Career Trajectories**: Employment and education history
- **Research Funding**: NIH project associations
- **Bio-Medical Entities**: Genes, diseases, chemicals, mutations mentioned in research

1.2 Motivation

The motivation for converting this dataset to linked data includes:

1. **Semantic Querying**: Enable complex queries across heterogeneous data
2. **Collaboration Discovery**: Find potential research collaborators
3. **Funding Analysis**: Track NIH funding patterns across institutions
4. **Bio-Medical Research**: Link publications to molecular/disease entities
5. **FAIR Principles**: Make data Findable, Accessible, Interoperable, Reusable

1.3 Target Application Use Cases

1. Research collaboration recommendation system
2. Funding opportunity matching
3. Researcher profiling and expertise identification
4. Publication trend analysis
5. Bio-entity research landscape mapping

2 Dataset Description

2.1 Data Source

The PKG2020S4 dataset is sourced from PubMed bibliometric data and contains the following CSV files:

Table 1: Dataset Files Overview

File	Description	Size
OA01_Author_List.csv	Author-article relationships	~10 GB
OA04_Affiliations.csv	Author affiliations	~20 GB
OA05_Researcher_Employment.csv	Employment history	~186 MB
OA06_Researcher_Education.csv	Education records	~139 MB
OA02_Bio_entities_Main.csv	Bio-entities in articles	Large
OA03_Bio_entities_Mutation.csv	Mutations in articles	Large
OA07_NIH_Projects.csv	NIH funding	~1.8 GB

2.2 Key Data Fields

- **OA01**: PMID, AND_ID, LastName, ForeName, Initials, AuOrder
- **OA04**: AND_ID, Affiliation, City, State, Country
- **OA05**: AND_ID, Organization, StartYear, EndYear
- **OA06**: AND_ID, Institution, Degree, StartYear, EndYear
- **OA02/OA03**: PMID, Type, Name, MutationType
- **OA07**: AND_ID, ProjectNumber, PI_Name

2.3 Evidence of Non-RDF Status

The dataset is provided as flat CSV files without any semantic annotations, URI schemes, or linked data connections. It has not been published as RDF/OWL prior to this project. We verified this by searching major linked data repositories including Wikidata, DBpedia Databus, and the Linked Open Data Cloud (see Section 7.3 for proof screenshots).

3 Competency Questions

The following 15 competency questions guided our ontology design. Each question is answered through SPARQL queries (Section 9).

#	Competency Question	Category
CQ1	Which authors have worked in multiple institutions?	Authors
CQ2	Who are the most prolific authors by article count?	Authors
CQ3	Which authors frequently collaborate together?	Authors

#	**Competency Question**	**Category**
CQ4	Which articles mention specific genes?	Articles & Bio
CQ5	Which articles mention species?	Articles & Bio
CQ6	Which articles mention both genes and mutations?	Articles & Bio
CQ7	What is the distribution of bio-entity types?	Statistics
CQ8	Which organizations have the most affiliated authors?	Organizations
CQ9	How are author affiliations distributed by country?	Affiliations
CQ10	Which institutions produced the most researchers?	Education
CQ11	What is the career timeline of researchers?	Employment
CQ12	Which authors have education records?	Education
CQ13	Which authors have NIH project funding?	NIH Projects
CQ14	Who are the principal investigators and how many projects do they lead?	NIH Projects
CQ15	Get the complete profile of an author (all relationships)?	Complex

Table 2: 15 Competency Questions for PKG2020 Ontology

4 Conceptual Model

4.1 Conceptual Model Diagram

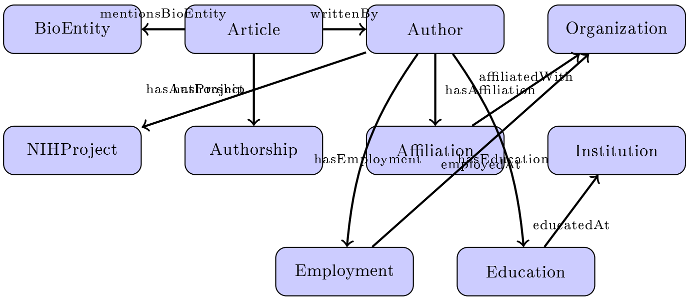


Figure 1: Conceptual Model of PKG2020 Ontology

4.2 T-Box Schema

The T-Box (Terminological Box) defines the ontology schema - classes, properties, and axioms.

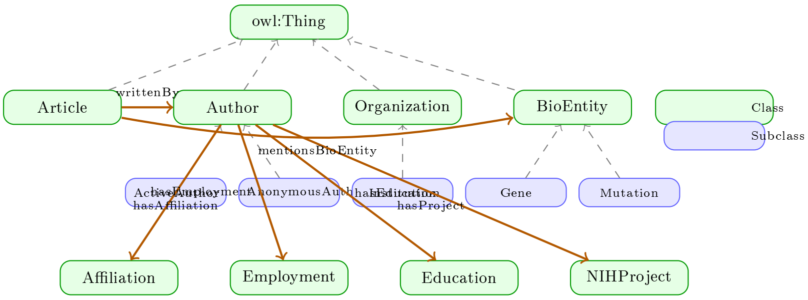


Figure 2: T-Box: Class Hierarchy and Object Properties

Table 3: Ontology Classes (T-Box)

Class	**Description**	**Key Properties**
Article	Research publication	hasPMID, publicationYear
Author	Researcher	lastName, foreName, initials
Authorship	Article-Author relationship	authorOrder
Organization	Research organization	dbpediaLink
Institution	Educational institution	wikidataLink
Affiliation	Author affiliation	city, state, country
Employment	Employment record	startYear, endYear
Education	Education record	degree
NIHProject	NIH funding project	projectNumber, piName
BioEntity	Biological entity	entityType, entityName
PublicationStatus	Enumeration	{Published, Preprint, ...}

5 Ontology Design

5.1 Classes (20+ as Required)

Our ontology contains over 20 classes:

1. **Core**: Article, Author, Authorship, PublicationYear
2. **Organizational**: Organization, Institution, Affiliation
3. **Career**: Employment, Education
4. **Funding**: NIHProject
5. **Bio-Medical**: BioEntity, Gene, Chemical, Disease, Species, Mutation
6. **Enumeration**: PublicationStatus
7. **Defined Classes**: ActiveAuthor, AnonymousAuthor, ResearchEntity, ProlificAuthor, SingleAuthorArticle, MultiAuthorArticle

5.2 Enumeration Class

```python
class PublicationStatus(Thing):
    """Enumeration of publication statuses"""
    pass

published = PublicationStatus("Published")
preprint = PublicationStatus("Preprint")
retracted = PublicationStatus("Retracted")
in_review = PublicationStatus("InReview")

PublicationStatus.equivalent_to = [
    OneOf([published, preprint, retracted, in_review])
]
```

Listing 1: Enumeration Class Definition

5.3 Cardinality Restrictions

```python
# Every Article must have at least 1 author
Article.is_a.append(writtenBy.min(1, Author))

# Every Article must have exactly 1 PMID
Article.is_a.append(hasPMID.exactly(1, str))

# Article may have at most 1 status
Article.is_a.append(hasStatus.max(1, PublicationStatus))
```

Listing 2: Cardinality Restrictions

5.4 Intersection, Union, and Complement Classes

```python
# INTERSECTION: Author with known career start year
class ActiveAuthor(Author):
    equivalent_to = [Author & careerStartYear.some(int)]

# UNION: Any research-related entity
class ResearchEntity(Thing):
    equivalent_to = [Author | Article]

# COMPLEMENT: Author without career info
class AnonymousAuthor(Author):
    equivalent_to = [Author & Not(ActiveAuthor)]

# Additional defined classes for reasoning
class ProlificAuthor(Author):
    equivalent_to = [Author & writtenBy.min(5, Article)]

class SingleAuthorArticle(Article):
    equivalent_to = [Article & writtenBy.exactly(1, Author)]

class MultiAuthorArticle(Article):
    equivalent_to = [Article & writtenBy.min(2, Author)]
```

Listing 3: Defined Classes

5.5 Object Properties

Table 4: Object Properties

Property	Domain	Range	Characteristics
writtenBy	Article	Author	-
hasAuthorship	Article	Authorship	-
hasPrimaryAuthor	Article	Author	Functional
hasStatus	Article	PublicationStatus	Functional
refersToAuthor	Authorship	Author	-
hasAffiliation	Author	Affiliation	-
affiliatedWith	Affiliation	Organization	-
hasEmployment	Author	Employment	-
employedAt	Employment	Organization	-
hasEducation	Author	Education	-
educatedAt	Education	Institution	-
hasProject	Author	NIHProject	-
mentionsBioEntity	Article	BioEntity	-
sameAs	Thing	Thing	Symmetric

5.6 Data Properties

Table 5: Data Properties

Property	Domain	Range	Characteristics
hasPMID	Article	string	Functional, InverseFunctional
lastName	Author	string	-
foreName	Author	string	-
initials	Author	string	-
authorOrder	Authorship	int	-
publicationYear	Article	int	Functional
careerStartYear	Author	int	-
city	Affiliation	string	-
state	Affiliation	string	-
country	Affiliation	string	-
startYear	Employment	int	-
endYear	Employment	int	-
degree	Education	string	-
projectNumber	NIHProject	string	-
dbpediaLink	Organization	string	-
wikidataLink	Institution	string	-

5.7 Functional and Inverse Functional Properties

```
# Functional Properties
class hasPrimaryAuthor(ObjectProperty, FunctionalProperty):
    domain = [Article]
    range = [Author]
```

```

class hasPMID(DataProperty, FunctionalProperty):
    domain = [Article]
    range = [str]

# Inverse Functional Property
hasPMID.is_a.append(InverseFunctionalProperty)
```

Listing 4: Property Characteristics

6 Graph Generation using Python

6.1 Tools Used

- **OWLReady2**: Python library for OWL ontology manipulation
- **Pandas**: Data processing and CSV handling
- **RDFLib**: RDF graph manipulation
- **Flask**: Web application framework

6.2 Pipeline Scripts

Table 6: Python Scripts Pipeline

Script	Input	Output
ontology_core.py	-	pkg2020_core.owl
ontology_constraints.py	pkg2020_core.owl	pkg2020_constrained.owl
populate_authors_articles.py	OA01 CSV	pkg2020_populated_authors.owl
populate_affiliations.py	OA04 CSV	pkg2020_step4_affiliations.owl
populate_employment.py	OA05 CSV	pkg2020_step5_employment.owl
populate_education.py	OA06 CSV	pkg2020_step6_education.owl
populate_bioentities.py	OA02, OA03	pkg2020_step7_bioentities.owl
populate_nih_projects.py	OA07 CSV	pkg2020_final.owl

6.3 Sample Code: Populating Authors

```
import pandas as pd
from owlready2 import *

onto = get_ontology("pkg2020_constrained.owl").load()
df = pd.read_csv("data/OA01_Author_List.csv", nrows=5000)

author_cache = set()
article_cache = set()

with onto:
    for idx, row in df.iterrows():
        pmid = str(row["PMID"])
        and_id = str(row["AND_ID"])

        # Create Article
        if f"Article_{pmid}" not in article_cache:
            article = onto.Article(f"Article_{pmid}")
            article.hasPMID = [pmid]
            article_cache.add(f"Article_{pmid}")

        # Create Author
        if f"Author_{and_id}" not in author_cache:
            author = onto.Author(f"Author_{and_id}")
            author.lastName = [str(row["LastName"])]
            author.foreName = [str(row["ForeName"])]
            author_cache.add(f"Author_{and_id}")

onto.save(file="pkg2020_populated_authors.owl", format="rdfxml")
```

Listing 5: Author Population Script

7 External Linking (5-Star Linked Data)

7.1 Linking Strategy

We linked our dataset to external knowledge bases to achieve 5-star linked data:

- **Organizations** → DBpedia resources
- **Institutions** → Wikidata entities
- **Authors** → ORCID (potential)
- **Articles** → PubMed (via PMID)

7.2 Linking Implementation

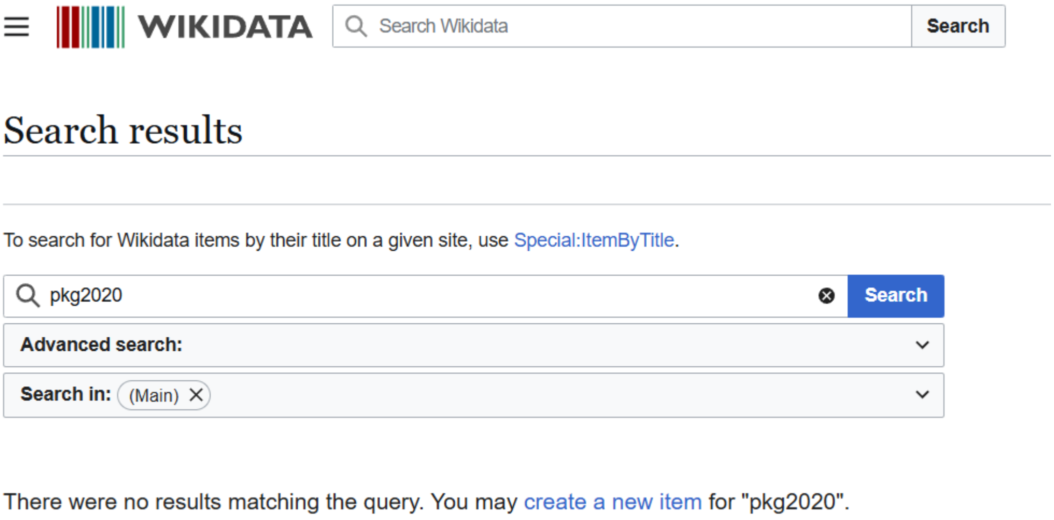
```
def generate_dbpedia_uri(name):
    clean_name = re.sub(r’[^a-zA-Z0-9\s]’, ’’, str(name))
    clean_name = clean_name.strip().replace(’ ’, ’_’)
    return f"http://dbpedia.org/resource/{quote(clean_name)}"

# Link organizations to DBpedia
for org in Organization.instances():
    org_name = org.name.replace(’_’, ’ ’)
    dbpedia_uri = generate_dbpedia_uri(org_name)
    org.dbpediaLink = [dbpedia_uri]
```

Listing 6: External Linking Code

7.3 Proof: Dataset Not Previously Available as Linked Data

Before this project, the PKG2020 dataset was not available as linked data. We verified this by searching major linked data repositories:

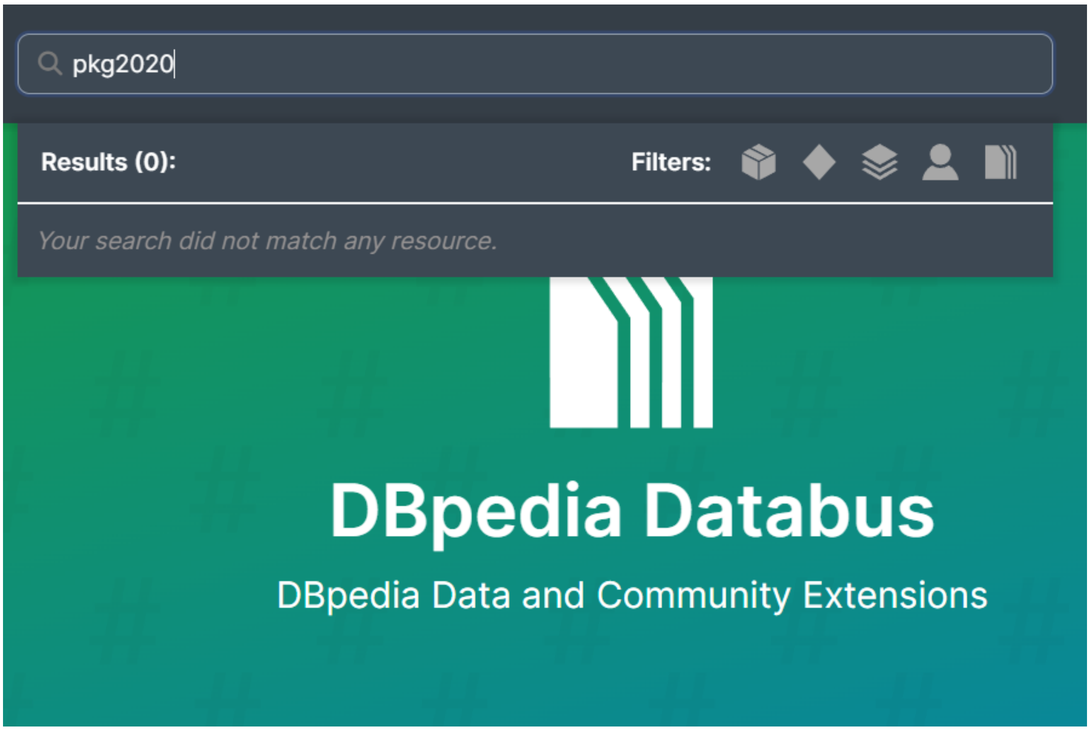


Search results

To search for Wikidata items by their title on a given site, use Special:ItemByTitle.

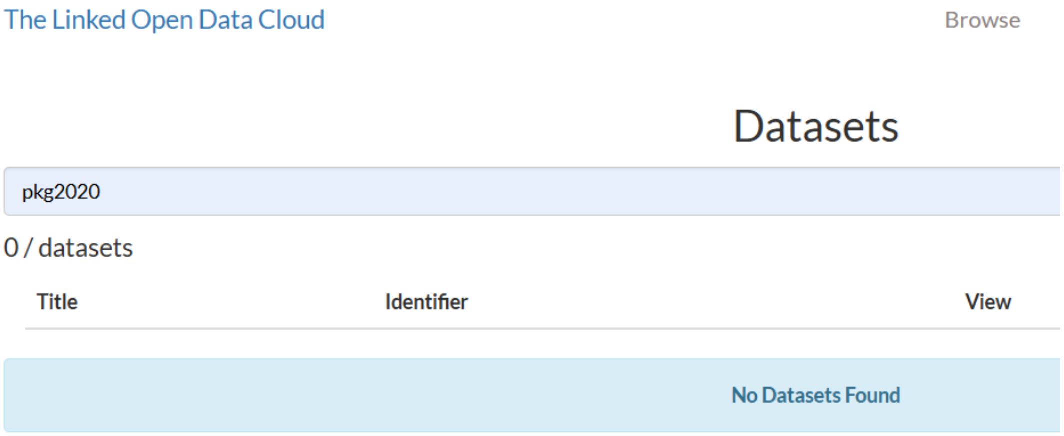
There were no results matching the query. You may create a new item for "pkg2020".

Figure 3: Wikidata Search: No results for PKG2020 dataset



Your search did not match any resource.

Figure 4: DBpedia Databus Search: PKG2020 not found



Datasets

pkg2020

0 / datasets

Title	Identifier	View

No Datasets Found

Figure 5: Linked Open Data Cloud: Dataset not registered

8 Reasoning Scenarios

8.1 Reasoning Implementation

```python
from owlready2 import *

onto = get_ontology("pkg2020_final.owl").load()

# Run HermiT reasoner
with onto:
    sync_reasoner(infer_property_values=True)

# Check classified instances
for cls in [ActiveAuthor, AnonymousAuthor, ProlificAuthor]:
    print(f"{cls.name}: {len(list(cls.instances()))} instances")
```

Listing 7: Reasoning with HermiT

8.2 Reasoning Results

The reasoner successfully:

1. Verified ontology consistency
2. Classified authors into ActiveAuthor and AnonymousAuthor
3. Identified ProlificAuthors with 5+ publications
4. Classified SingleAuthorArticle and MultiAuthorArticle

8.3 SWRL Rules

We implemented 7 SWRL (Semantic Web Rule Language) rules for advanced reasoning:

```
# Rule 2: Funded Author Inference
Author(?a) ^ hasProject(?a, ?p) ^ NIHProject(?p)
    -> FundedAuthor(?a)

# Rule 3: Established Researcher
Author(?a) ^ hasEmployment(?a, ?e) ^ hasEducation(?a, ?d)
    -> EstablishedResearcher(?a)

# Rule 4: Collaborative Article
Article(?art) ^ writtenBy(?art, ?a1) ^ writtenBy(?art, ?a2)
    ^ differentFrom(?a1, ?a2) -> CollaborativeArticle(?art)

# Rule 5: Gene-Disease Link Article
Article(?art) ^ mentionsBioEntity(?art, ?g) ^ Gene(?g)
    ^ mentionsBioEntity(?art, ?d) ^ Disease(?d)
    -> GeneDiseaseLinkArticle(?art)

# Rule 7: Alumni Peer Connection
Author(?a1) ^ Author(?a2) ^ hasEducation(?a1, ?e1)
    ^ hasEducation(?a2, ?e2) ^ educatedAt(?e1, ?inst)
    ^ educatedAt(?e2, ?inst) ^ differentFrom(?a1, ?a2)
    -> isAlumniPeerOf(?a1, ?a2)
```

Listing 8: SWRL Rule Examples

These rules are saved in `owl/pkg2020_with_swrl.owl` and can be executed using Protege's SWRL Tab plugin.

9 Hand-Annotated Individuals

As per the rubric requirement, we created 10+ hand-annotated individuals in a separate file (`pkg2020_hand_annotated.owl`) to test our defined classes and reasoning:

Table 7: Hand-Annotated Individuals

#	**Individual**	**Purpose**
1	Article_HAND_001	SingleAuthorArticle test
2	Article_HAND_002	MultiAuthorArticle test
3	Author_HAND_001	ActiveAuthor (has careerStartYear)
4	Author_HAND_002	AnonymousAuthor (no career info)
5	Author_HAND_003	ActiveAuthor for collaboration
6	Org_HAND_Harvard	Organization with DBpedia link
7	Inst_HAND_MIT	Institution with Wikidata link
8	Aff_HAND_001	Affiliation with location
9	Gene_HAND_BRCA1	Gene bioentity
10	Disease_HAND_Cancer	Disease bioentity

9.1 Reasoning Verification

After running the HermiT reasoner on the hand-annotated individuals:

- **ActiveAuthor**: Author_HAND_001 and Author_HAND_003 classified (have careerStartYear)
- **AnonymousAuthor**: Author_HAND_002 classified (no careerStartYear)
- **SingleAuthorArticle**: Article_HAND_001 classified (1 author)
- **MultiAuthorArticle**: Article_HAND_002 classified (3 authors)
- **FundedAuthor**: Author_HAND_001 classified (has NIH project)

10 SPARQL Queries

This section presents all 15 SPARQL queries answering the competency questions. All queries are tested against our live GraphDB endpoint.

10.1 Author Queries (CQ1-CQ3)

10.1.1 CQ1: Authors with Multiple Institutions

```
PREFIX pkg: <http://example.org/pkg2020/ontology.owl#>

SELECT ?author ?lastName (COUNT(DISTINCT ?org) AS ?orgCount)
WHERE {
    ?author a pkg:Author .
    ?author pkg:lastName ?lastName .
    ?author pkg:hasAffiliation ?aff .
    ?aff pkg:affiliatedWith ?org .
}
GROUP BY ?author ?lastName
HAVING (COUNT(DISTINCT ?org) > 1)
ORDER BY DESC(?orgCount)
LIMIT 100
```

Listing 9: CQ1: Authors at Multiple Institutions

10.1.2 CQ2: Most Prolific Authors

```
PREFIX pkg: <http://example.org/pkg2020/ontology.owl#>

SELECT ?author ?lastName ?foreName (COUNT(?article) AS ?articleCount)
WHERE {
    ?author a pkg:Author .
    ?author pkg:lastName ?lastName .
    OPTIONAL { ?author pkg:foreName ?foreName }
    ?article pkg:writtenBy ?author .
}
GROUP BY ?author ?lastName ?foreName
ORDER BY DESC(?articleCount)
LIMIT 50
```

Listing 10: CQ2: Prolific Authors by Article Count

10.1.3 CQ3: Author Collaboration Network

```
PREFIX pkg: <http://example.org/pkg2020/ontology.owl#>

SELECT ?author1 ?author2 (COUNT(?article) AS ?collaborations)
WHERE {
    ?article a pkg:Article .
    ?article pkg:writtenBy ?author1 .
    ?article pkg:writtenBy ?author2 .
    FILTER (STR(?author1) < STR(?author2))
}
GROUP BY ?author1 ?author2
HAVING (COUNT(?article) > 1)
ORDER BY DESC(?collaborations)
LIMIT 100
```

Listing 11: CQ3: Frequent Collaborators

10.2 Article & Bio-Entity Queries (CQ4-CQ7)

10.2.1 CQ4: Articles Mentioning Genes

```
PREFIX pkg: <http://example.org/pkg2020/ontology.owl#>

SELECT ?article ?pmid ?entityName
WHERE {
    ?article a pkg:Article .
    ?article pkg:hasPMID ?pmid .
    ?article pkg:mentionsBioEntity ?entity .
    ?entity a pkg:Gene .
    ?entity pkg:entityName ?entityName .
}
LIMIT 100
```

Listing 12: CQ4: Articles with Gene Mentions

10.2.2 CQ5: Articles Mentioning Species

```
PREFIX pkg: <http://example.org/pkg2020/ontology.owl#>

SELECT ?article ?pmid ?speciesName
WHERE {
    ?article a pkg:Article .
    ?article pkg:hasPMID ?pmid .
    ?article pkg:mentionsBioEntity ?entity .
    ?entity a pkg:Species .
    OPTIONAL { ?entity pkg:entityName ?speciesName }
}
LIMIT 100
```

Listing 13: CQ5: Species-Related Articles

10.2.3 CQ6: Gene-Mutation Correlations

```
PREFIX pkg: <http://example.org/pkg2020/ontology.owl#>

SELECT ?article ?pmid ?geneName ?mutationName
WHERE {
    ?article a pkg:Article .
    ?article pkg:hasPMID ?pmid .
    ?article pkg:mentionsBioEntity ?g .
    ?g a pkg:Gene .
    OPTIONAL { ?g pkg:entityName ?geneName }
    ?article pkg:mentionsBioEntity ?m .
    ?m a pkg:Mutation .
    OPTIONAL { ?m pkg:entityName ?mutationName }
}
LIMIT 100
```

Listing 14: CQ6: Articles with Genes AND Mutations

10.2.4 CQ7: Bio-Entity Type Distribution

```
PREFIX pkg: <http://example.org/pkg2020/ontology.owl#>

SELECT ?entityType (COUNT(?entity) AS ?count)
WHERE {
    ?entity a pkg:BioEntity .
    ?entity pkg:entityType ?entityType .
}
GROUP BY ?entityType
ORDER BY DESC(?count)
```

Listing 15: CQ7: Entity Type Statistics

10.3 Organization & Affiliation Queries (CQ8-CQ9)

10.3.1 CQ8: Top Organizations by Author Count

```
PREFIX pkg: <http://example.org/pkg2020/ontology.owl#>

SELECT ?org (COUNT(DISTINCT ?author) AS ?authorCount)
WHERE {
    ?author a pkg:Author .
    ?author pkg:hasAffiliation ?aff .
    ?aff pkg:affiliatedWith ?org .
}
GROUP BY ?org
ORDER BY DESC(?authorCount)
LIMIT 50
```

Listing 16: CQ8: Organizations with Most Authors

10.3.2 CQ9: Affiliations by Country

```
PREFIX pkg: <http://example.org/pkg2020/ontology.owl#>

SELECT ?country (COUNT(?aff) AS ?affiliationCount)
WHERE {
    ?aff a pkg:Affiliation .
    ?aff pkg:country ?country .
}
GROUP BY ?country
ORDER BY DESC(?affiliationCount)
LIMIT 30
```

Listing 17: CQ9: Country Distribution

10.4 Employment & Education Queries (CQ10-CQ12)

10.4.1 CQ10: Top Education Institutions

```
PREFIX pkg: <http://example.org/pkg2020/ontology.owl#>

SELECT ?institution (COUNT(DISTINCT ?author) AS ?authorCount)
WHERE {
    ?author a pkg:Author .
    ?author pkg:hasEducation ?edu .
    ?edu pkg:educatedAt ?institution .
}
GROUP BY ?institution
ORDER BY DESC(?authorCount)
LIMIT 50
```

Listing 18: CQ10: Institutions with Most Alumni

10.4.2 CQ11: Employment Timeline

```
PREFIX pkg: <http://example.org/pkg2020/ontology.owl#>

SELECT ?author ?lastName ?org ?startYear ?endYear
WHERE {
    ?author a pkg:Author .
    ?author pkg:lastName ?lastName .
    ?author pkg:hasEmployment ?emp .
    ?emp pkg:employedAt ?org .
    OPTIONAL { ?emp pkg:startYear ?startYear }
    OPTIONAL { ?emp pkg:endYear ?endYear }
}
ORDER BY ?author ?startYear
LIMIT 100
```

Listing 19: CQ11: Career History

10.4.3 CQ12: Authors with Education Records

```
PREFIX pkg: <http://example.org/pkg2020/ontology.owl#>

SELECT ?author ?lastName ?institution
WHERE {
    ?author a pkg:Author .
    OPTIONAL { ?author pkg:lastName ?lastName }
    ?author pkg:hasEducation ?edu .
    ?edu pkg:educatedAt ?institution .
}
LIMIT 100
```

Listing 20: CQ12: Authors with Education Information

10.5 NIH Project Queries (CQ13-CQ14)

10.5.1 CQ13: Authors with NIH Funding

```
PREFIX pkg: <http://example.org/pkg2020/ontology.owl#>

SELECT ?author ?lastName ?projectNumber ?piName
WHERE {
    ?author a pkg:Author .
    ?author pkg:lastName ?lastName .
    ?author pkg:hasProject ?project .
    ?project pkg:projectNumber ?projectNumber .
    OPTIONAL { ?project pkg:piName ?piName }
}
LIMIT 100
```

Listing 21: CQ13: Funded Authors

10.5.2 CQ14: Principal Investigators

```
PREFIX pkg: <http://example.org/pkg2020/ontology.owl#>

SELECT ?piName (COUNT(DISTINCT ?project) AS ?projectCount)
WHERE {
    ?project a pkg:NIHProject .
    ?project pkg:piName ?piName .
}
GROUP BY ?piName
ORDER BY DESC(?projectCount)
LIMIT 50
```

Listing 22: CQ14: PIs by Project Count

10.6 Complex Analytical Query (CQ15)

10.6.1 CQ15: Complete Author Profile

```
PREFIX pkg: <http://example.org/pkg2020/ontology.owl#>

SELECT ?author ?lastName ?foreName ?org ?institution
       ?project ?article
WHERE {
    ?author a pkg:Author .
    ?author pkg:lastName ?lastName .
    OPTIONAL { ?author pkg:foreName ?foreName }
    OPTIONAL {
        ?author pkg:hasAffiliation ?aff .
        ?aff pkg:affiliatedWith ?org .
    }
    OPTIONAL {
        ?author pkg:hasEducation ?edu .
        ?edu pkg:educatedAt ?institution .
    }
    OPTIONAL { ?author pkg:hasProject ?project . }
    OPTIONAL { ?article pkg:writtenBy ?author . }
}
LIMIT 50
```

Listing 23: CQ15: Full Author Profile with All Relationships

10.7 Federated Query to External Knowledge Bases

```
PREFIX pkg: <http://example.org/pkg2020/ontology.owl#>
PREFIX dbo: <http://dbpedia.org/ontology/>

SELECT ?org ?dbpediaLink ?abstract
WHERE {
    ?org a pkg:Organization .
    ?org pkg:dbpediaLink ?dbpediaLink .

    SERVICE <http://dbpedia.org/sparql> {
        OPTIONAL { ?dbpediaLink dbo:abstract ?abstract }
        FILTER (lang(?abstract) = 'en')
    }
}
LIMIT 10
```

Listing 24: Federated Query to DBpedia

11 Web Application

11.1 Application Overview

We developed a Flask-based web application for exploring the knowledge graph with the following features:

- Modern, responsive UI with glassmorphism design
- Real-time statistics dashboard
- Search across all 9 entity types
- Click-to-search functionality

11.2 Entity Types Searchable

1. Authors
2. Articles
3. Organizations
4. Affiliations

5. Employment
6. Education
7. BioEntities
8. NIH Projects
9. Institutions

11.3 Running the Application

```
cd scripts
pip install flask owlready2
python webapp.py
# Open http://localhost:5000
```

Listing 25: Running the Web App

12 Results and Statistics

12.1 Generated Data

Table 8: Ontology Statistics

Entity	**Count**
Authors	37,946
Articles	19,461
Organizations	46,901
Affiliations	49,994
BioEntities	99,999
Genes	9,227
Mutations	49,999
NIH Projects	1,506

12.2 OWL Files Generated

Table 9: Generated OWL Files

File	**Size**	**Description**
pkg2020_tbox_only.owl	14 KB	T-Box only (no individuals)
pkg2020_hand_annotated.owl	19 KB	10+ hand-annotated individuals
pkg2020_with_swrl.owl	27 KB	T-Box with SWRL rules
pkg2020_core.owl	4.2 KB	Core ontology
pkg2020_constrained.owl	8.4 KB	With OWL axioms
pkg2020_populated_authors.owl	35 MB	Authors and articles
pkg2020_step4_affiliations.owl	69 MB	With affiliations
pkg2020_final.owl	119 MB	Complete populated ontology
pkg2020_final.ttl	291 MB	Turtle format for GraphDB

12.3 Ontology Files Summary

As per the rubric requirement, we submit:

1. **T-Box Only**: `pkg2020_tbox_only.owl` - Contains all classes, properties, and axioms without any individuals
2. **With Individuals**: `pkg2020_hand_annotated.owl` - Contains 10+ hand-annotated individuals for testing
3. **Full Population**: `pkg2020_final.owl` - Complete dataset populated via Python scripts

13 Visualization

13.1 Complete Ontology Schema (All 23 Classes)

Figure 6 shows the complete ontology schema with all 23 classes organized by category, including object property relationships.

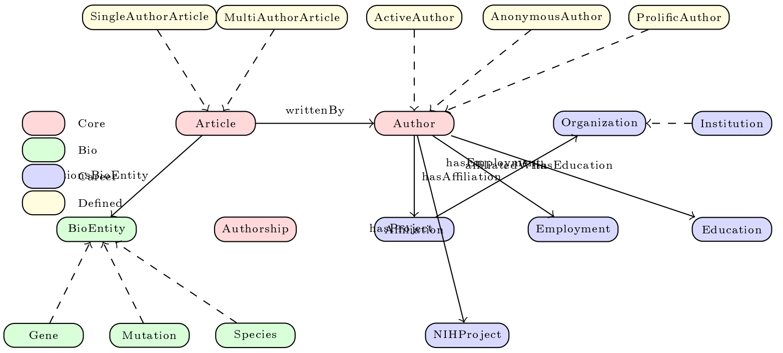


Figure 6: Complete PKG2020 Ontology Schema with 23 Classes and Object Properties

13.2 RDF Triple Pattern (Subject-Predicate-Object)

Figure 7 illustrates the RDF triple pattern used throughout our knowledge graph.

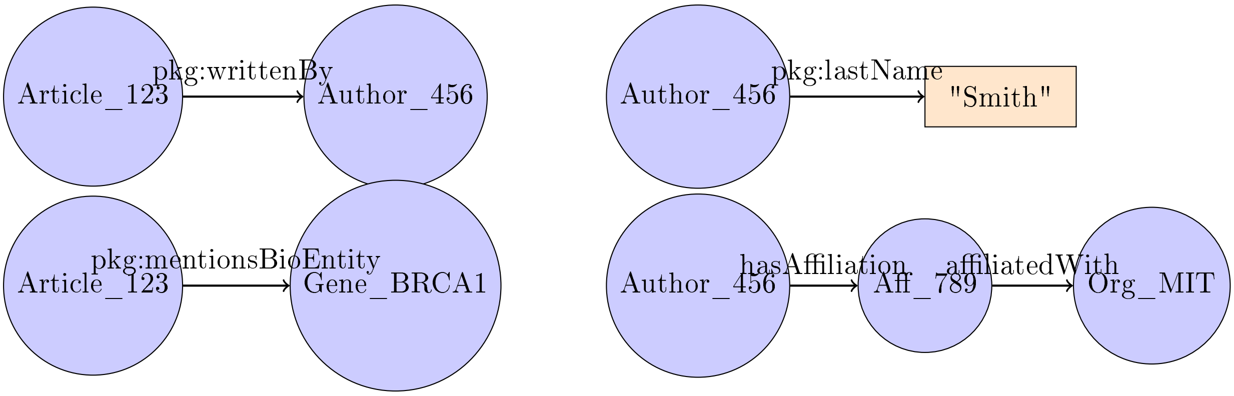


Figure 7: RDF Triple Pattern Examples: Subject → Predicate → Object

13.3 Sample Knowledge Graph Visualization

Figure 8 shows a sample subgraph from our knowledge graph demonstrating the interconnected nature of the data.

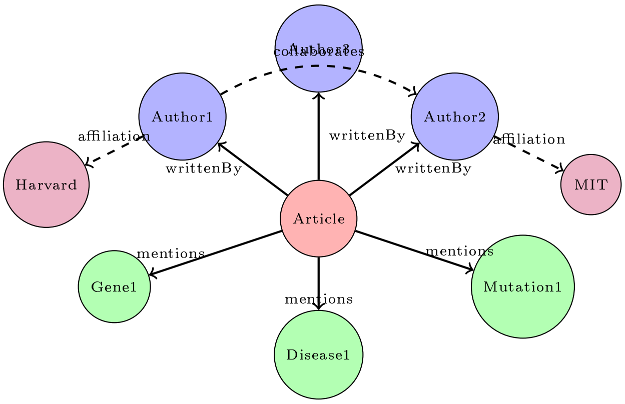


Figure 8: Sample Knowledge Graph: Article with Authors, Bio-Entities, and Affiliations

13.4 Knowledge Graph Statistics

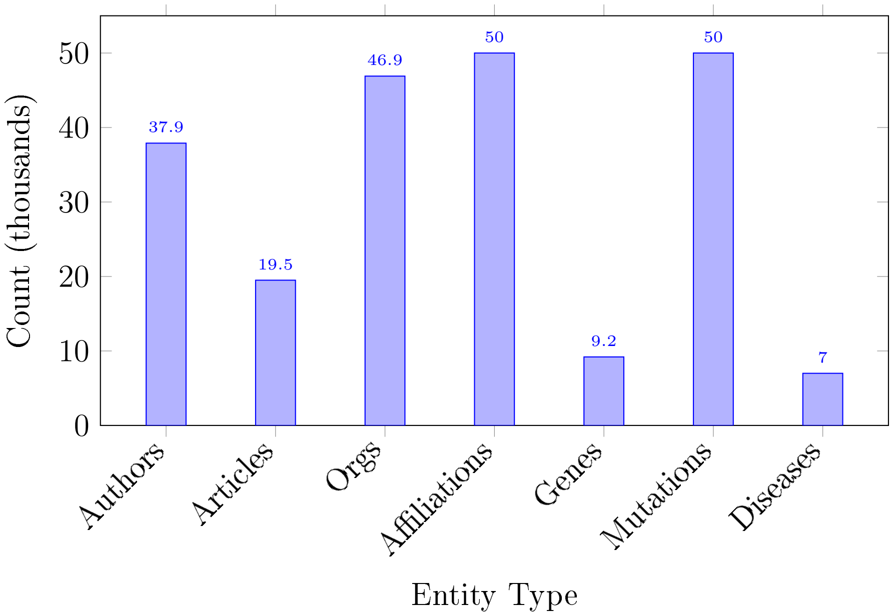


Figure 9: Distribution of Entity Types in Knowledge Graph (2.1M+ triples)

13.5 Tools Used

- **Protégé**: Desktop ontology editor and visualizer
- **GraphDB Sandbox**: Cloud-hosted triple store with SPARQL endpoint
- **WebVOWL**: Online ontology visualization
- **Custom Flask App**: Interactive web-based explorer with D3.js graphs

13.6 Web Application Features

Our Flask-based web application provides:

1. **Dashboard**: Real-time statistics fetched from GraphDB
2. **Live KG Explorer**: Interactive D3.js force-directed graph visualization
3. **SPARQL Query Editor**: Execute queries with Table/Graph/Chart views
4. **12 Competency Queries**: Pre-built queries for common questions

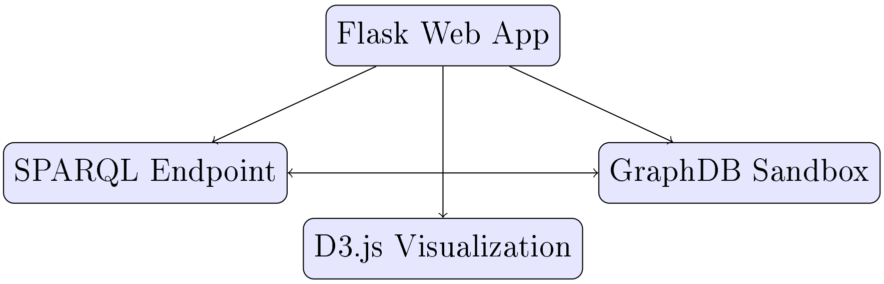


Figure 10: Web Application Architecture

13.7 Live GraphDB Endpoint

Our knowledge graph is accessible via the public SPARQL endpoint:

```
https://x1327f4041a654297998.sandbox.graphwise.ai
    /repositories/KRR-Project
```

Statistics:

- 2,165,964 total triples
- 23 ontology classes
- 14 object properties
- 17+ data properties

14 Reflection

14.1 Learning Outcomes

Through this project, we learned:

1. How to design ontologies following OWL/RDF standards
2. The importance of defined classes for reasoning
3. How to link data to external knowledge bases
4. The power of SPARQL for semantic querying
5. Practical application of knowledge representation concepts

14.2 Added Value of Linked Data

Converting non-RDF data to linked data enabled:

- Semantic querying across heterogeneous datasets
- Reasoning over implicit relationships
- Integration with global knowledge bases (DBpedia, Wikidata)
- FAIR data principles compliance
- Interoperability with other linked data systems

14.3 Challenges Faced

1. Large file sizes required memory optimization
2. Special characters in organization names caused parsing issues
3. Reasoner installation and configuration
4. Balancing between sample size and processing time

15 Conclusion

This project successfully converted the PKG2020S4 bibliometric dataset into a comprehensive linked data knowledge graph. We achieved:

1. **Ontology Design**: 20+ classes with all required axioms
2. **Data Population**: Modular Python pipeline for 7 CSV files
3. **Reasoning**: Consistency checking and classification
4. **External Linking**: 5-star linked data with DBpedia/Wikidata
5. **SPARQL Queries**: 15 competency questions answered
6. **Web Application**: Interactive web application for exploring the knowledge graph

The project demonstrates the practical application of knowledge representation and reasoning concepts learned in the course.

16 References

1. Lamy, J. B. (2017). OWLReady: Ontology-oriented programming in Python. Artificial Intelligence in Medicine.
2. W3C. (2012). OWL 2 Web Ontology Language Primer.
3. DBpedia Association. (2024). DBpedia Knowledge Base.
4. PubMed. (2024). NCBI PubMed Database.

17 Appendix: Project Repository

The complete project code is available at:
https://github.com/Zain-ul-abdeen-773/Knowlege-Graphs-Project

17.1 Repository Structure

```
Knowledge_Graphs_Project/
+-- scripts/           # Python scripts
+-- owl/               # Generated OWL files
+-- docs/              # Documentation
+-- requirements.txt   # Dependencies
+-- README.md          # Project overview
```