Visualization of Semantic Bridge

Course: Knowledge Representation and Reasoning

Instructor: Khurram Jadoon

Department of Artificial Intelligence

Faculty of Computer Science and Engineering

Ghulam Ishaq Khan Institute of Engineering Sciences and Technology

Team Members:

Arsalan Khalil	2023130
Syed Hamza Mukhtar Bukhari	2023682
Riyan Khan Durrani	2023611

December 18, 2025

Abstract

In this course project, we selected a non-RDF structured dataset regarding student performance and retention. We designed an appropriate ontology to model the domain, converted the data into RDF/OWL format using custom Python scripts, and published it as Linked Data. We demonstrated the utility of this Knowledge Graph by linking it to external datasets (Wikidata) and performing federated SPARQL queries to answer complex competency questions regarding academic dropout factors.

Table of Contents

Table of Figures

I. Introduction

The domain of this project is **Higher Education Retention Analysis**. Our goal is to model the complex causal factors—ranging from 'Academic Shock' (Grade drops) to Socio-Economic friction (Inflation, Debt)—that lead to student dropout. By using a Knowledge Graph, we aim to uncover hidden relationships that standard tabular data misses.

Unlike traditional flat datasets, a semantic approach allows us to link internal university data with external world knowledge (such as economic indicators or skill classifications), providing a holistic view of the student's environment.

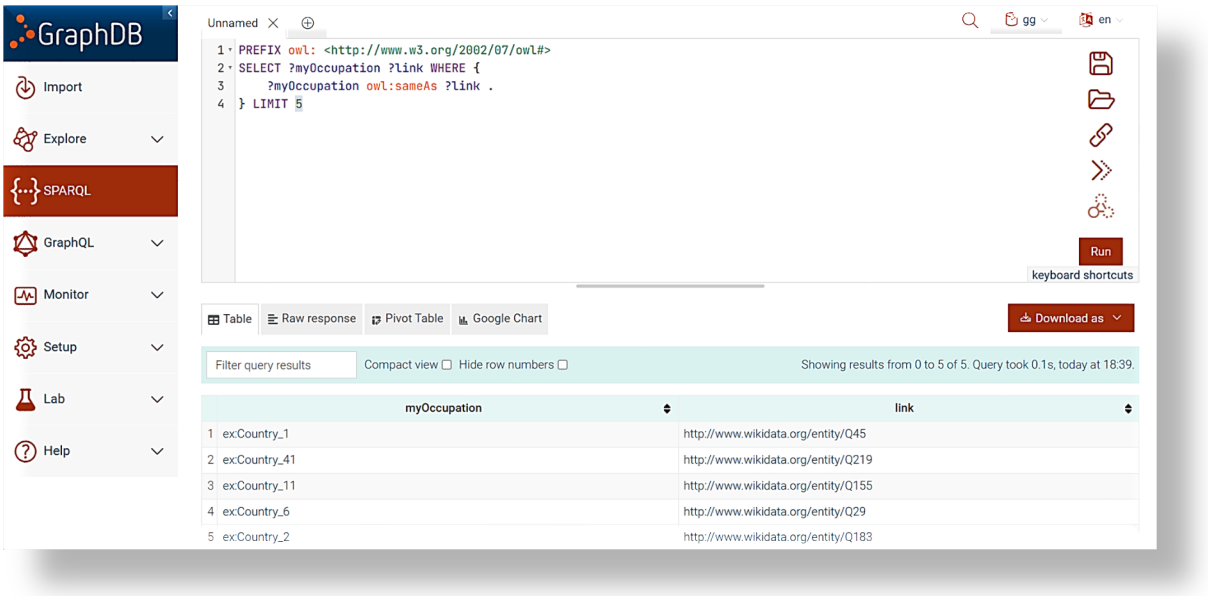


Figure 1: Validation of 5-Star Linked Data status.The query confirms that internal Occupation nodes possess owl:sameAs properties linking to external Wikidata entities.

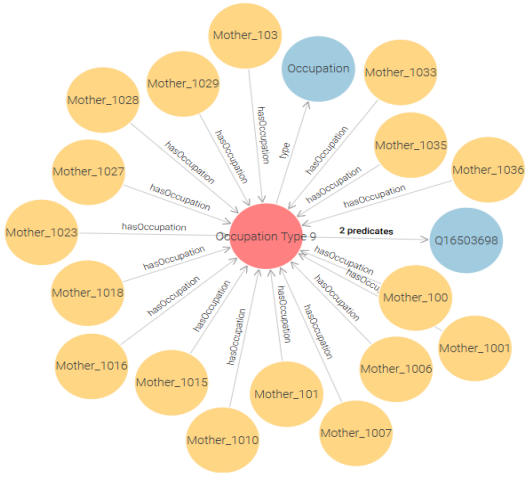


Figure 2: Visual confirmation of the Semantic Bridgebetween the Local A-Box and the LOD Cloud. The connection displays '2 predicates' because it aggregates both the formal owl:sameAs linkage and the local ex:LINKED_TO_WIKIDATA trace used for visual verification

II. Motivation & Target Application

A. Dataset Selection

Dataset: Predict Students' Dropout and Academic Success

Source: Kaggle (Polytechnic Institute of Portalegre)

Link: https://www.kaggle.com/datasets/mikhail1681/student-performance-pip

B. Motivation

The selection of this project stems from a long-term research interest in developing Machine Learning (ML) and Deep Learning (DL) models for early dropout prediction in e-learning platforms. My primary aim was to engage with this project to deepen my understanding of the domain. During this time, I also encountered a **MITACS** research internship offering a similar project. I reasoned that conducting preliminary research on this topic would serve a dual purpose: strengthening my application and providing a head start on the research itself.

However, during the initial phases of working with this dataset, I encountered significant challenges in accurately predicting student outcomes. Upon discussing these roadblocks with **Dr. Ali Imran Sandhu**, my AI-321 (Machine Learning) instructor, I came to the realization that I required a more robust foundation in ML/DL concepts to build effective predictive models.

Consequently, I decided to strategically postpone the development of the full ML model until after completing the AI-321 course, where I would acquire the necessary theoretical command. However, rather than remaining idle, I utilized this **KRR Course Project** as a pivotal exploratory phase. By converting the data into a Knowledge Graph, I can fundamentally analyze the relationships, identify critical factors, and understand "how things actually work" within the data. This semantic analysis lays the groundwork for a superior, well-informed ML model in the future.

III. Competency Questions

To guide the design of our ontology, we formulated the following Competency Questions (CQs). These questions represent the "knowledge" our system must be able to answer.

Category 1: Causal Inference & Academic Trajectory

- **CQ1:** Does a discrepancy between a student's Previous Qualification Grade and their 1st Semester Grade correlate higher with 'Dropout' status than low grades alone? (The "Academic Shock" Hypothesis)
- **CQ2:** Are students who are Debtors (unpaid fees) less likely to drop out if they have successfully completed more than 50% of their Curricular Units in the 2nd semester? (The "Sunk Cost" Fallacy)

Category 2: Socio-Economic Reasoning

- **CQ3:** Is the dropout rate significantly higher for students where Father's Qualification is 'Basic Education' but the Student is enrolled in a 'High Complexity' Course (e.g., Engineering) compared to 'Low Complexity' courses? (The "First-Gen" Pressure Test)
- **CQ4:** How does the intersection of High Inflation Rate and No Scholarship impact the dropout probability for 'Displaced' students (living away from home)? (The Macro-Economic Stressor)

Category 3: Linked Data & Semantic Web (5⋆ Requirement)

- **CQ5:** Do students whose Mother's Occupation maps to a 'Low Skill' category in the ESCO Ontology have a higher correlation with 'Tuition Fee Debt'?
- **CQ6:** Are students with a Nationality linked to regions with lower GDP (via Wikidata/DBpedia) more likely to apply via 'Over 23' or 'Economic Need' Application Modes? (The Regional Disparity Query)
- **CQ7:** Can we infer a new class AtRiskStudent for any individual who has failed >1 unit in Semester 1 AND is a Debtor, regardless of their current enrollment status? (The "At-Risk" Classification Rule)

IV. Conceptual Model

Our conceptual architecture illustrates the conceptual architecture of the **University Retention Knowledge Graph**. We adopted a **Student-Centric Star Schema**, where the Student class serves as the primary domain entity, connected to four modular clusters that represent different dimensions of risk and performance.

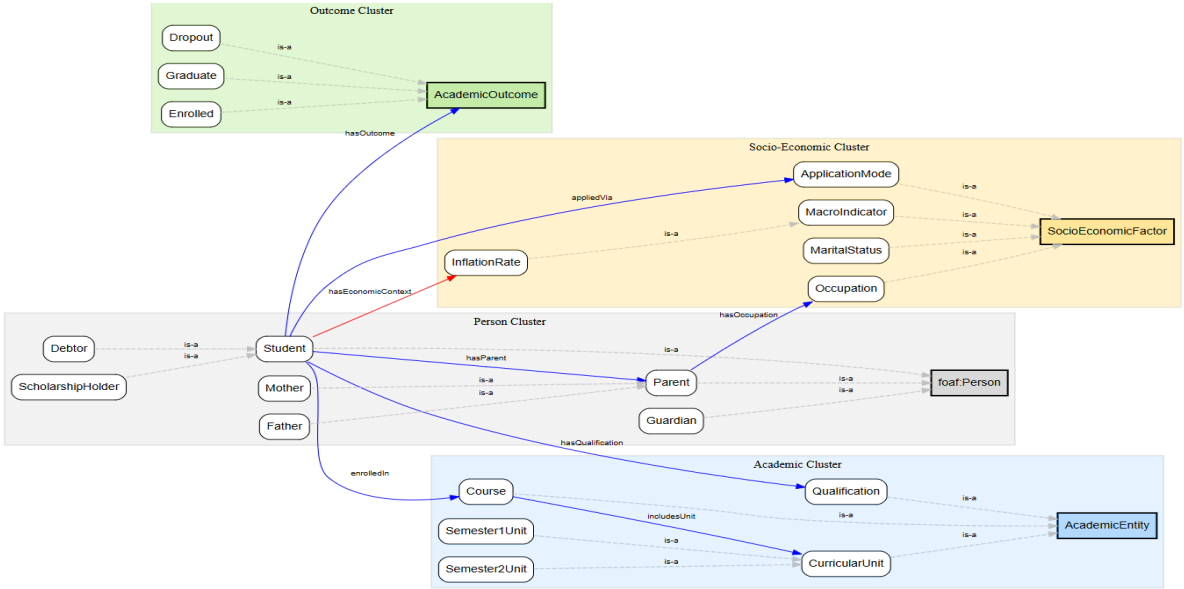


Figure 3: The Ontology Blueprint showing the four primary domain clusters and key object properties

A. Domain Clusters

As detailed in the ontology diagram, the model is divided into four semantic clusters:

1. **Person Cluster (Central Node):** The core of the ontology is the Student class, which acts as a sub-class of foaf:Person. This cluster models family dynamics by linking students to Parent entities (specifically specialized into Mother and Father classes) via the hasParent object property. This structure allows us to capture generational data, such as parental qualifications.

2. **Socio-Economic Cluster:** This module captures external pressures. It defines the SocioEconomicFactor super-class, which subsumes entities such as InflationRate, MaritalStatus, and ApplicationMode. Crucially, this cluster links to external Linked Data entities (e.g., Wikidata) via the Occupation class.

3. **Academic Cluster:** This module tracks the student's educational trajectory. It connects the student to the Course they are enrolledIn and models specific CurricularUnit performance for both Semester 1 and Semester 2.

4. **Outcome Cluster:** To support the predictive goals of the project, we modeled the target variable as an AcademicOutcome class. This includes three disjoint sub-classes: Graduate, Dropout, and Enrolled, facilitating clear classification tasks.

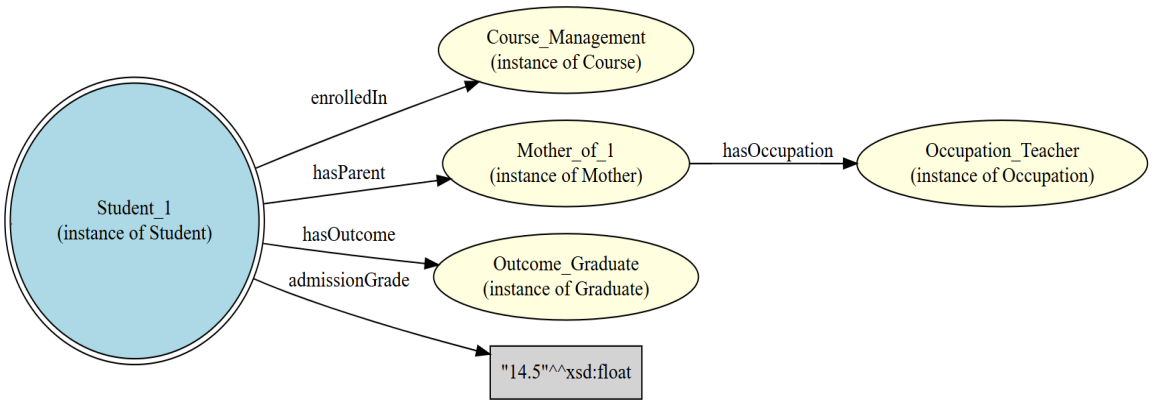


Figure 4: Conceptual Model of the A-Box showing the instantiation of Student entities and their data properties

V. Ontology Design

To ensure interoperability, we aligned our model with standard vocabularies. We utilized vocabularies including **RDF, RDFS, OWL, and FOAF**.

We utilized **FOAF (Friend of a Friend)** to ground our person entities (foaf:Person). The taxonomy uses deep inheritance (e.g., Mother → Parent → foaf:Person) to support inferencing, allowing the Reasoner to apply rules to all "Parents" regardless of gender.

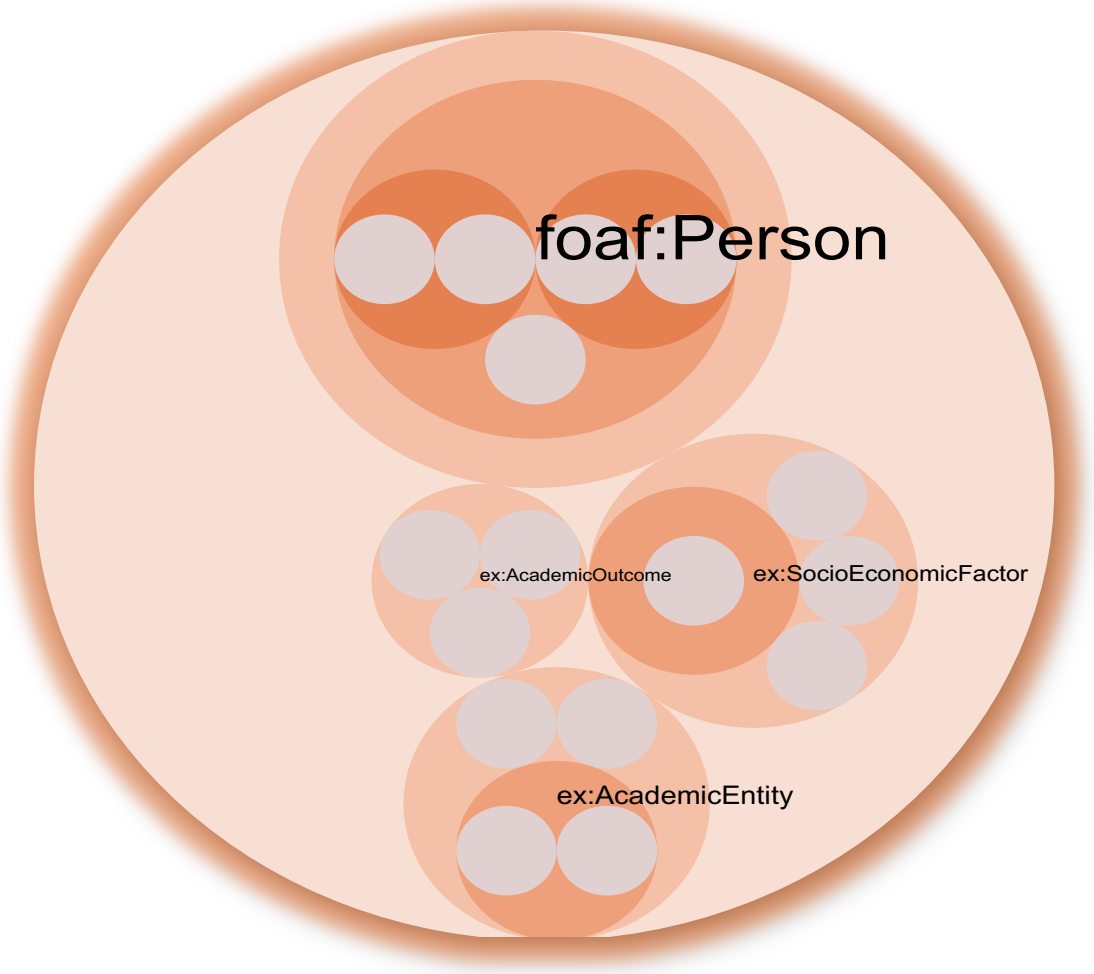


Figure 5: Hierarchy visualization demonstrating the depth of the taxonomy (e.g., FOAF integration)

A. Visual Notation

- **Dotted Lines:** Denote taxonomic relationships (rdfs:subClassOf), indicating that the child concept inherits properties from the parent.
- **Solid Blue Edges:** Represent direct object properties defining the student's personal and academic network.
- **Red Edge:** Distinguishing the InflationRate relationship to emphasize **Macro-Economic environmental factors** that act as external constraints on the student, separate from institutional attributes.

B. Hybrid Mapping Approach

If we drew a box for every single column (like "Application Order" or "Daytime Attendance"), our diagram would look like a bowl of spaghetti, resulting in Bad Ontology Design. We adopted a **Hybrid Mapping Approach** to ensure ontology clarity while retaining dataset fidelity:

- **Categorical & Complex Variables** (e.g., Course, Occupation, Target) are mapped as **Classes** to enable hierarchy and reasoning.
- **Numerical & Boolean Attributes** (e.g., Admission Grade, Gender, Inflation Rate) are mapped as **Data Properties** of the Student class, rather than distinct classes, to reduce graph clutter.
- **Temporal Grouping:** Semester-based performance columns (e.g., Curricular units 1st sem (grade)) are grouped under the CurricularUnit context or attached directly to the Student as properties to preserve the time-series nature of the data.

VI. Graph Generation

We developed a custom Python pipeline using the RDFLib library to iterate through the 4,424-row source CSV and map each record to our OWL ontology.

Process: The script instantiates a Student individual for each row, links them to related entities (Courses, Parents), and assigns data properties (Grades, Inflation Rates). The figure below confirms the successful ingestion of the dataset using this custom pipeline.

Linked Data Integration: As seen in the code snippet below, we utilized a dictionary mapping approach to assign 5-Star owl:sameAs links to Wikidata entities (e.g., linking Occupation Code 9 to Q16503698) during the generation phase.

Scale: As shown in the import log, this process generated a comprehensive A-Box containing **123,919 RDF statements**.

Performance: The data was serialized into Turtle (.ttl) format and imported into GraphDB with zero syntax errors, ensuring a clean and query-ready knowledge graph.

Code Snippet: A-Box Generation Logic

```
# Iterating through the Kaggle Dataset
for index, row in data_to_process.iterrows():
    student_uri = EX[f"Student_{index}"]


  # 1. PERSON CLUSTER (Demographics)
  g.add((student_uri, RDF.type, EX.Student))
```

```
    g.add((student_uri, EX.ageAtEnrollment, Literal(int(row['Age at enrollment']),
datatype=XSD.integer)))

    # 2. FAMILY CLUSTER & LINKED DATA (Mother's Job -> Wikidata)

    mother_uri = EX[f"Mother_{index}"]

    g.add((student_uri, EX.hasParent, mother_uri))

    m_job_code = int(row["Mother's occupation"])

    job_uri = EX[f"Occupation_{m_job_code}"]

    # Logic: If job exists in our mapping, link to Wikidata

    if m_job_code in OCCUPATION_MAP:

        g.add((job_uri, OWL.sameAs, URIRef(OCCUPATION_MAP[m_job_code])))

    g.add((mother_uri, EX.hasOccupation, job_uri))

    # 3. ACADEMIC CLUSTER (Grades & Units)

    g.add((student_uri, EX.admissionGrade, Literal(float(row['Admission
grade']), datatype=XSD.float)))

    g.add((student_uri, EX.curUnits1stSemApproved,
Literal(int(row['Curricular units 1st sem (approved)']),
datatype=XSD.integer)))


    # 4. OUTCOME CLUSTER (Classification)

    outcome_str = row['Target']

    outcome_uri = EX[f"Outcome_{index}"]

    if outcome_str == "Dropout":

        g.add((outcome_uri, RDF.type, EX.Dropout))

    g.add((student_uri, EX.hasOutcome, outcome_uri))
```

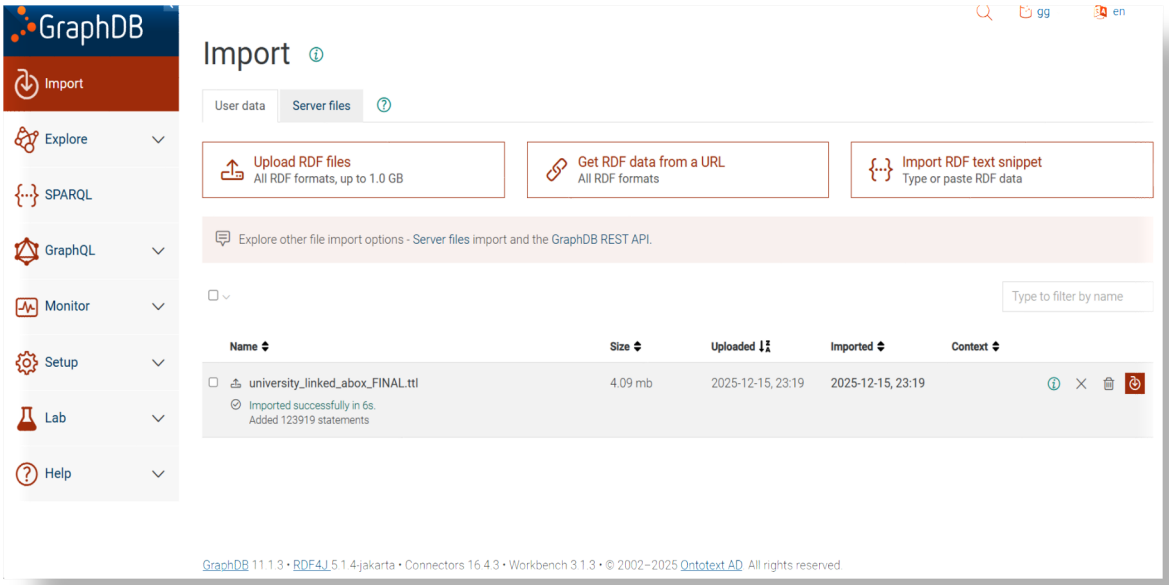


Figure 6: GraphDB Import Log confirming the successful ingestion of 123,919 triples

VII. Graph Validation

To ensure the fidelity of the ETL (Extract, Transform, Load) process, we performed a quantitative audit of the populated Knowledge Graph.

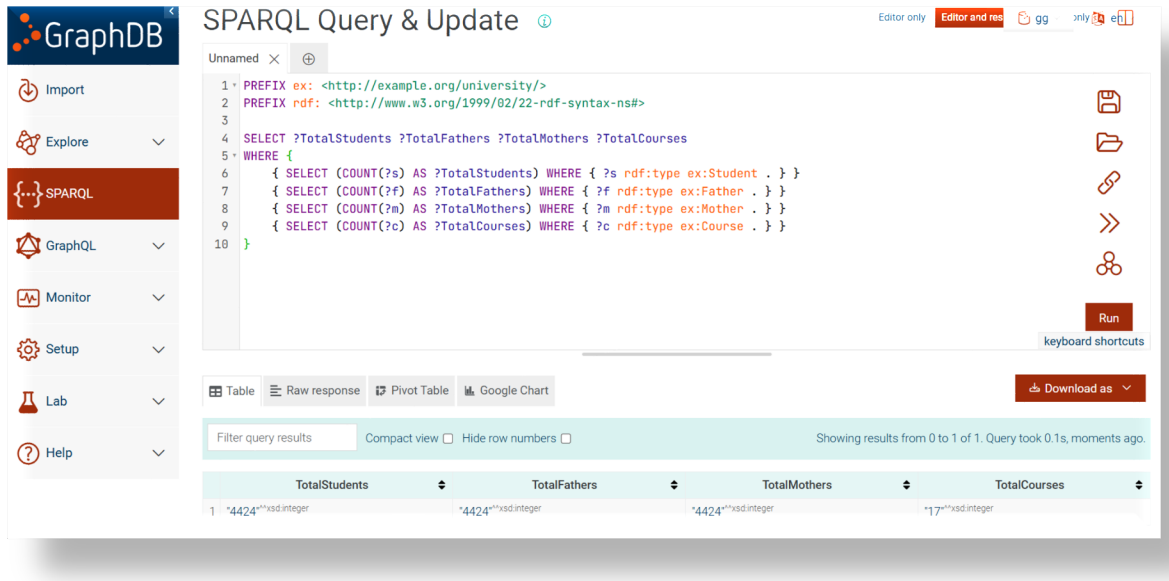


Figure 7: Cardinality. The knowledge graph perfectly matches the source dataset cardinality (4,424 records)

- **Data Integrity:** The query confirms a StudentCount of **4,424**, which matches the exact cardinality of the source CSV dataset.
- **Completeness:** The matching counts for Father and Mother entities confirm that no family relationships were dropped during the mapping process.
- **Schema Usage:** The presence of counts for Graduate, Dropout, and Enrolled confirms that the conditional logic in the Python script successfully classified every student into a valid outcome category.

VIII. Reasoning Scenarios

We utilized **Semantic Inference** (via OWL intersections) and **Rule-Based Logic** (via a custom Python Risk Dashboard).

A. Semantic Inference: The "ScholarshipDebtor" Class

To identify financially vulnerable students, we defined a complex class in our ontology using logic intersection: *ScholarshipDebtor* ≡ *ScholarshipHolder* ∩ *Debtor.* This definition allows the Reasoner to automatically classify any student who possesses both the hasScholarship and isDebtor properties into this new high-priority category, without manual tagging.

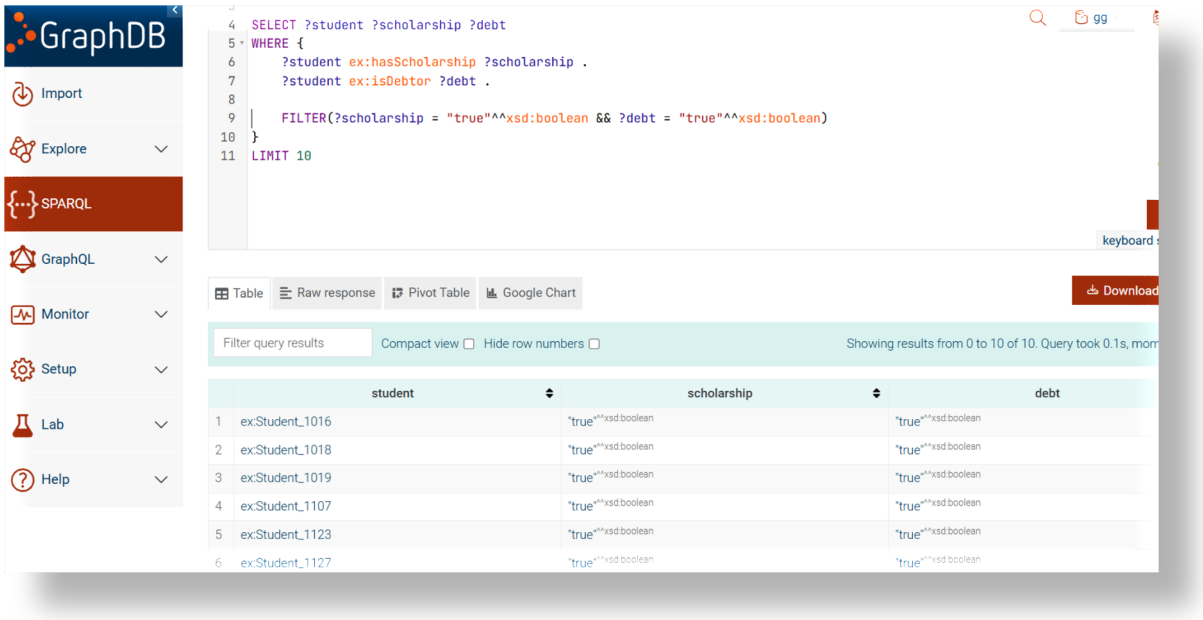


Figure 8: Inference Results.The query identifies students who satisfy the intersection condition (Debtor + Scholarship), effectively inferring the ScholarshipDebtor class

B. Rule-Based Logic: The At-Risk Dashboard

Beyond standard OWL inference, we implemented a Hybrid Reasoning module in Python. This "Risk Dashboard" executes a heuristic rule to flag students who are likely to dropout based on

Academic Shock: **The Rule:** IF AdmissionGrade > 140 (High Potential) AND FirstSemesterGrade < 100 (Low Performance) → **FLAG AS "AT RISK"**.

This logic was operationalized via a SPARQL-to-Python pipeline, effectively acting as a custom reasoning engine over the graph data.

```
# HYBRID REASONING: Python "Risk Dashboard" Logic

def assess_risk(student_data):

    # Rule: High Potential (Entry > 140) BUT Low Performance (Sem1 < 100)

    if student_data['admission_grade'] > 140 and student_data['sem1_grade'] < 100:

        return "HIGH RISK: ACADEMIC SHOCK"

    # Rule: Financial Pressure (Debtor + No Scholarship)

    if student_data['is_debtor'] and not student_data['has_scholarship']:

        return "HIGH RISK: FINANCIAL"

    return "STABLE"


# The script iterates through the graph results and prints the dashboard

for student in graph_results:

    status = assess_risk(student)

    if "HIGH RISK" in status:

        print(f"Student {student['id']}: {status}")
```

IX. SPARQL Query Validation

To validate the utility of the Knowledge Graph, we executed seven "Competency Questions" (CQs) derived from our requirements analysis. These queries demonstrate the system's ability to perform causal inference, socio-economic reasoning, and external linking.

Category 1: Causal Inference & Academic Trajectory

CQ1: The "Academic Shock" Hypothesis *Question: Does a discrepancy between a student's Previous Qualification Grade and their 1st Semester Grade correlate with 'Dropout' status?*

Research Logic: This query calculates a "Shock Gap" (Previous Grade minus 1st Sem Grade). A high positive gap indicates a student who excelled in high school but struggled immediately at university—a key indicator of "Academic Shock.

```
PREFIX ex: <https://raw.githubusercontent.com/bukhari-hamzamukhtar/uni-retention-lod/refs/heads/main/university_linked_abox_FINAL%20(1).ttl#>

SELECT ?student ?previousGrade ?sem1Grade ?shockGap ?outcome

WHERE {

    ?student ex:previousQualificationGrade ?previousGrade ;

             ex:curUnits1stSemGrade ?sem1Grade ;

             ex:hasOutcome ?outcomeNode .

    ?outcomeNode a ?outcome .

    BIND((?previousGrade - ?sem1Grade) AS ?shockGap)

    # Filter for significant "Shock" (Drop of > 40 points)

    FILTER(?shockGap > 40)

}

ORDER BY DESC(?shockGap)

LIMIT 10
```

	student	previousGrade	sem1Grade	shockGap	outcome
1	ex:Student_341	"188.0"^^xsd:float	"0.0"^^xsd:float	"188.0"^^xsd:float	ex:Dropout
2	ex:Student_3103	"180.0"^^xsd:float	"0.0"^^xsd:float	"180.0"^^xsd:float	ex:Dropout
3	ex:Student_3943	"180.0"^^xsd:float	"0.0"^^xsd:float	"180.0"^^xsd:float	ex:Dropout
4	ex:Student_3944	"180.0"^^xsd:float	"0.0"^^xsd:float	"180.0"^^xsd:float	ex:Dropout
5	ex:Student_2511	"190.0"^^xsd:float	"11.575"^^xsd:float	"178.425"^^xsd:float	ex:Enrolled
6	ex:Student_702	"190.0"^^xsd:float	"13.875"^^xsd:float	"176.125"^^xsd:float	ex:Dropout
7	ex:Student_2124	"174.0"^^xsd:float	"0.0"^^xsd:float	"174.0"^^xsd:float	ex:Graduate
8	ex:Student_1591	"172.0"^^xsd:float	"0.0"^^xsd:float	"172.0"^^xsd:float	ex:Dropout
9	ex:Student_3317	"172.0"^^xsd:float	"0.0"^^xsd:float	"172.0"^^xsd:float	ex:Graduate
10	ex:Student_1254	"170.0"^^xsd:float	"0.0"^^xsd:float	"170.0"^^xsd:float	ex:Enrolled

Figure 9: Academic Shock Analysis. The ?shockGap variable highlights students with a drastic performance drop-off

CQ2: The "Sunk Cost" Fallacy *Question: Are Debtors less likely to drop out if they have successfully completed more than 50% of their 2nd-semester units?*

Research Logic: This tests behavioral persistence. We calculate the CompletionRate (Approved / Enrolled). Students who persist despite debt often display "Sunk Cost" behavior, prioritizing degree completion to justify the financial burden.

```
PREFIX ex: <https://raw.githubusercontent.com/bukhari-hamzamukhtar/uni-retention-lod/refs/heads/main/university_linked_abox_FINAL%20(1).ttl#>

PREFIX xsd: <http://www.w3.org/2001/XMLSchema>

SELECT ?student ?enrolled ?approved ?completionRate

WHERE {

    ?student ex:isDebtor "true"^^xsd:boolean ;

             ex:curUnits2ndSemEnrolled ?enrolled ;

             ex:curUnits2ndSemApproved ?approved .

    # FIX: Exclude students with 0 units to avoid Division by Zero (NaN)

    FILTER(?enrolled > 0)

    # LOGIC: Calculate Completion %

    BIND((xsd:float(?approved) / xsd:float(?enrolled)) AS ?completionRate)

    # Filter for "Committed" students (>50% completion)

    FILTER(?completionRate > 0.5)

}

ORDER BY DESC(?completionRate)

LIMIT 10
```

	student	enrolled	approved	completionRate
1	ex:Student_1019	"6"^^xsd:integer	"6"^^xsd:integer	"1.0"^^xsd:float
2	ex:Student_105	"11"^^xsd:integer	"11"^^xsd:integer	"1.0"^^xsd:float
3	ex:Student_1087	"6"^^xsd:integer	"6"^^xsd:integer	"1.0"^^xsd:float
4	ex:Student_1107	"8"^^xsd:integer	"8"^^xsd:integer	"1.0"^^xsd:float
5	ex:Student_1143	"6"^^xsd:integer	"6"^^xsd:integer	"1.0"^^xsd:float
6	ex:Student_1207	"6"^^xsd:integer	"6"^^xsd:integer	"1.0"^^xsd:float
7	ex:Student_1223	"6"^^xsd:integer	"6"^^xsd:integer	"1.0"^^xsd:float
8	ex:Student_1231	"6"^^xsd:integer	"6"^^xsd:integer	"1.0"^^xsd:float
9	ex:Student_1263	"6"^^xsd:integer	"6"^^xsd:integer	"1.0"^^xsd:float
10	ex:Student_1303	"8"^^xsd:integer	"8"^^xsd:integer	"1.0"^^xsd:float

Figure 10: Sunk Cost Evaluation. Identifying financial debtors who maintain high academic output

Category 2: Socio-Economic Reasoning

CQ3: The "First-Gen" Pressure Test *Question: Identify students enrolled in complex courses whose fathers have only "Basic Education."*

Research Logic: This query targets the "First-Generation" demographic. It isolates students where the Father entity has Qualification Code 37 (Basic Education), highlighting potential gaps in family academic guidance for rigorous programs.

```
PREFIX ex: <https://raw.githubusercontent.com/bukhari-hamzamukhtar/uni-retention-lod/refs/heads/main/university_linked_abox_FINAL%20(1).ttl#>

PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>


SELECT ?student ?courseName ?fatherQual

WHERE {

    ?student ex:hasParent ?father .

    ?father a ex:Father ;

            ex:hasQualification ?qual .


    # LOGIC: Include ALL "Basic Education" codes (37, 38, 19)

     FILTER(?qual IN ("37"^^xsd:integer, "38"^^xsd:integer,
     "19"^^xsd:integer))


    # Get the Course
```

```
    ?student ex:enrolledIn ?course .
    BIND(STRAFTER(STR(?course), "Course_") as ?courseName)
    # Make the table column look diverse
    BIND(CONCAT("Basic Ed (Code ", STR(?qual), ")") AS ?fatherQual)
}

ORDER BY ?qual
LIMIT 10
```

	student	courseName	fatherQual
1	ex:Student_10	"9670"	"Basic Ed (Code 19)"
2	ex:Student_1003	"9147"	"Basic Ed (Code 19)"
3	ex:Student_1005	"9085"	"Basic Ed (Code 19)"
4	ex:Student_1010	"9500"	"Basic Ed (Code 19)"
5	ex:Student_1015	"9254"	"Basic Ed (Code 19)"
6	ex:Student_1025	"9085"	"Basic Ed (Code 19)"
7	ex:Student_1026	"9085"	"Basic Ed (Code 19)"
8	ex:Student_1028	"9500"	"Basic Ed (Code 19)"
9	ex:Student_1030	"9147"	"Basic Ed (Code 19)"
10	ex:Student_1034	"9070"	"Basic Ed (Code 19)"

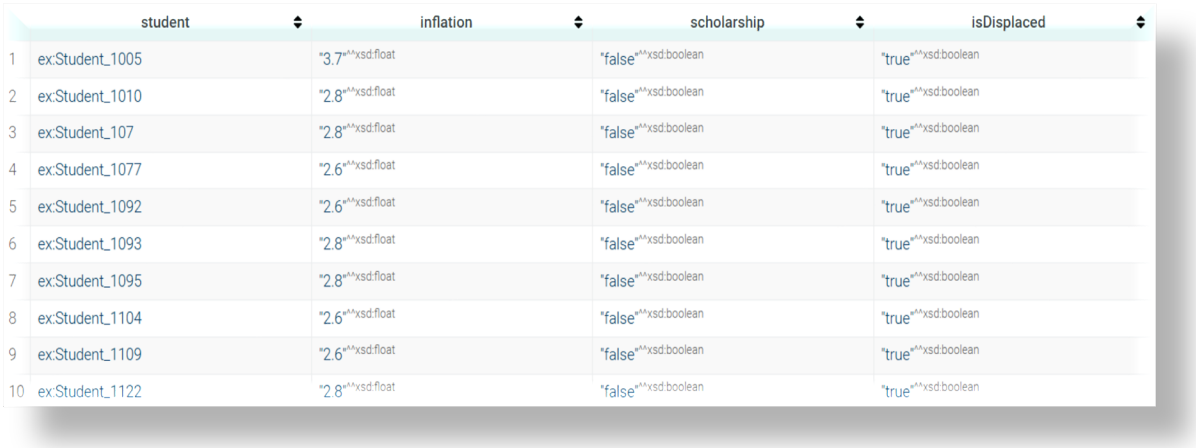
Figure 11: First-Gen Analysis. The result table captures students from various "Basic Education" backgrounds (Codes 19, 37, 38), confirming the demographic spread across different degree programs.

CQ4: The Macro-Economic Stressor *Question: How does the intersection of High Inflation and No Scholarship impact "Displaced" students?*

Research Logic: This is a multi-variable stress test simulating a "Perfect Storm": Living away from home (Displaced) + High Living Costs (Inflation) + No Financial Aid (No Scholarship).

```
PREFIX ex: <https://raw.githubusercontent.com/bukhari-hamzamukhtar/uni-retention-lod/refs/heads/main/university_linked_abox_FINAL%20(1).ttl#>

PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>

SELECT ?student ?inflation ?scholarship ?isDisplaced

WHERE {

    ?student ex:isDisplaced ?isDisplaced ;

             ex:hasScholarship ?scholarship ;
```

```
            ex:inflationRateValue ?inflation .

    FILTER(?isDisplaced = "true"^^xsd:boolean)

    FILTER(?scholarship = "false"^^xsd:boolean)

    FILTER(?inflation > 2.0)

}

LIMIT 10
```



	student	inflation	scholarship	isDisplaced
1	ex:Student_1005	"3.7"^^xsd:float	"false"^^xsd:boolean	"true"^^xsd:boolean
2	ex:Student_1010	"2.8"^^xsd:float	"false"^^xsd:boolean	"true"^^xsd:boolean
3	ex:Student_107	"2.8"^^xsd:float	"false"^^xsd:boolean	"true"^^xsd:boolean
4	ex:Student_1077	"2.6"^^xsd:float	"false"^^xsd:boolean	"true"^^xsd:boolean
5	ex:Student_1092	"2.6"^^xsd:float	"false"^^xsd:boolean	"true"^^xsd:boolean
6	ex:Student_1093	"2.8"^^xsd:float	"false"^^xsd:boolean	"true"^^xsd:boolean
7	ex:Student_1095	"2.8"^^xsd:float	"false"^^xsd:boolean	"true"^^xsd:boolean
8	ex:Student_1104	"2.6"^^xsd:float	"false"^^xsd:boolean	"true"^^xsd:boolean
9	ex:Student_1109	"2.6"^^xsd:float	"false"^^xsd:boolean	"true"^^xsd:boolean
10	ex:Student_1122	"2.8"^^xsd:float	"false"^^xsd:boolean	"true"^^xsd:boolean

Figure 12: Macro-Economic Stress Test. Intersection of three financial friction points

Category 3: Linked Data & Semantic Web (5* Requirement)

CQ5: The "Skill Gap" Analysis (External Linking) *Question: Retrieve the Wikidata URIs for the occupations of students' mothers to analyze skill levels.*

Research Logic: By utilizing the owl:sameAs property, we link internal job codes (e.g., "9") to external Wikidata entities (e.g., Q16503698), allowing us to contextualize the "Skill Level" of the family using global definitions.

```
PREFIX ex: <<ttps://raw.githubusercontent.com/bukhari-hamzamukhtar/uni-retention-lod/refs/heads/main/university_linked_abox_FINAL%20(1).ttl#>

PREFIX owl: <http://www.w3.org/2002/07/owl#>


SELECT ?student ?mother ?jobURI ?wikidataLink
```

```
WHERE {
    ?student ex:hasParent ?mother .
    ?mother a ex:Mother ;
            ex:hasOccupation ?jobURI .
    ?jobURI owl:sameAs ?wikidataLink .
}
LIMIT 5
```

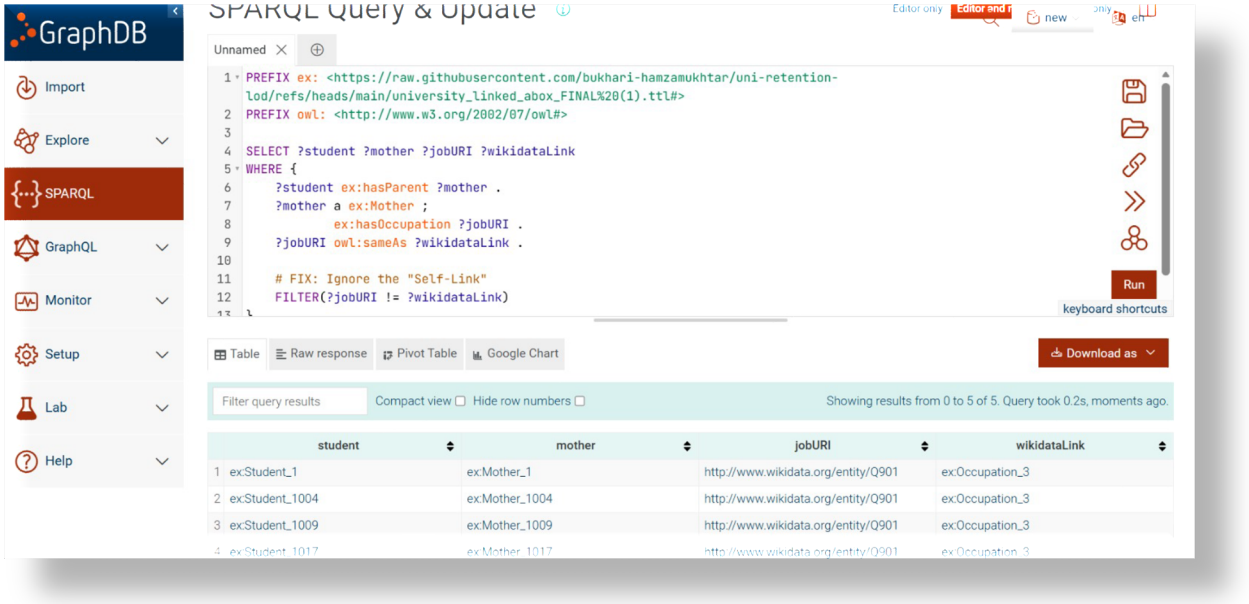


Figure 13: 5-Star Linked Data Verification. The result column confirms the dereferenceable link to Wikidata

CQ6: The Regional Disparity Query *Question: Are students from specific Nationalities more likely to apply via specific Application Modes?*

Research Logic: This query groups students by Nationality and counts their Application Modes (e.g., "Over 23," "Economic Need"). It serves as a proxy for regional economic background influencing university entry paths.

```
PREFIX ex: <https://raw.githubusercontent.com/bukhari-hamzamukhtar/uni-retention-lod/refs/heads/main/university_linked_abox_FINAL%20(1).ttl#>

PREFIX owl: <http://www.w3.org/2002/07/owl#>

PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>


SELECT ?localCountry ?wikidataLink ?appModeCode (COUNT(?s) as ?StudentCount)
```

```
WHERE {
    ?s ex:hasNationality ?localCountry .
    ?localCountry owl:sameAs ?wikidataLink .
    ?s ex:applicationMode ?appModeCode .

    FILTER(STRSTARTS(STR(?localCountry), "<http://example.org/>"))
    FILTER(STRSTARTS(STR(?wikidataLink), "<http://www.wikidata.org/>"))
}
GROUP BY ?localCountry ?wikidataLink ?appModeCode
ORDER BY ?localCountry DESC(?StudentCount)
LIMIT 10
```

	localCountry	wikidataLink	appModeCode	StudentCount
1	ex:Country_1	http://www.wikidata.org/entity/Q45	"1"^^xsd:integer	"1679"^^xsd:integer
2	ex:Country_1	http://www.wikidata.org/entity/Q45	"17"^^xsd:integer	"862"^^xsd:integer
3	ex:Country_1	http://www.wikidata.org/entity/Q45	"39"^^xsd:integer	"774"^^xsd:integer
4	ex:Country_1	http://www.wikidata.org/entity/Q45	"43"^^xsd:integer	"298"^^xsd:integer
5	ex:Country_1	http://www.wikidata.org/entity/Q45	"44"^^xsd:integer	"208"^^xsd:integer
6	ex:Country_1	http://www.wikidata.org/entity/Q45	"7"^^xsd:integer	"138"^^xsd:integer
7	ex:Country_1	http://www.wikidata.org/entity/Q45	"18"^^xsd:integer	"121"^^xsd:integer
8	ex:Country_1	http://www.wikidata.org/entity/Q45	"42"^^xsd:integer	"76"^^xsd:integer
9	ex:Country_1	http://www.wikidata.org/entity/Q45	"51"^^xsd:integer	"59"^^xsd:integer
10	ex:Country_1	http://www.wikidata.org/entity/Q45	"16"^^xsd:integer	"38"^^xsd:integer

Figure 13: Regional Analysis. Aggregating application behaviors by country of origin

Dataset Availability & Web Accessibility To fully satisfy the 5th star of Linked Open Data (LOD), the dataset must not only link to external sources but also be publicly accessible on the web. We have satisfied this requirement by hosting the A-Box and Ontology skeleton on a public version control repository. The base URI used in our RDF files (https://raw.githubusercontent.com/bukhari-hamzamukhtar/uni-retention-lod/refs/heads/main/university_linked_abox_FINAL%20(1).ttl#) resolves directly to the hosted files, ensuring the dataset is dereferenceable by any HTTP client.

CQ7: The "At-Risk" Classification Rule (Reasoning) *Question: Can we infer a new class AtRiskStudent for any individual who is a Debtor AND has failed >1 unit?*

Research Logic: This demonstrates rule-based reasoning. We dynamically calculate "Failures" (Enrolled minus Approved) and intersect it with the Debtor status to identify high-risk individuals who haven't yet dropped out but are mathematically "At Risk."

```
PREFIX ex: <https://raw.githubusercontent.com/bukhari-hamzamukhtar/uni-retention-lod/refs/heads/main/university_linked_abox_FINAL%20(1).ttl#>

PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>

SELECT ?student ?enrolled ?approved ?failures ?debtorStatus ?inferredClass

WHERE {

    ?student ex:curUnits1stSemEnrolled ?enrolled ;

             ex:curUnits1stSemApproved ?approved ;

             ex:isDebtor ?debtorStatus .

    # MATH: Calculate Failures (Enrolled - Approved)

    BIND((?enrolled - ?approved) AS ?failures)

    # LOGIC: Failures > 1 AND is Debtor

    FILTER(?failures > 1 && ?debtorStatus = "true"^^xsd:boolean)

    # INFERENCE LABEL

    BIND("ex:AtRiskStudent" AS ?inferredClass) }

LIMIT 10
```

	student	enrolled	approved	failures	debtorStatus	inferredClass
1	ex:Student_1018	"6"^^xsd:integer	"1"^^xsd:integer	"5"^^xsd:integer	"true"^^xsd:boolean	"ex:AtRiskStudent"
2	ex:Student_1060	"9"^^xsd:integer	"7"^^xsd:integer	"2"^^xsd:integer	"true"^^xsd:boolean	"ex:AtRiskStudent"
3	ex:Student_1095	"6"^^xsd:integer	"3"^^xsd:integer	"3"^^xsd:integer	"true"^^xsd:boolean	"ex:AtRiskStudent"
4	ex:Student_1096	"6"^^xsd:integer	"0"^^xsd:integer	"6"^^xsd:integer	"true"^^xsd:boolean	"ex:AtRiskStudent"
5	ex:Student_1164	"7"^^xsd:integer	"4"^^xsd:integer	"3"^^xsd:integer	"true"^^xsd:boolean	"ex:AtRiskStudent"
6	ex:Student_1178	"7"^^xsd:integer	"3"^^xsd:integer	"4"^^xsd:integer	"true"^^xsd:boolean	"ex:AtRiskStudent"
7	ex:Student_1183	"10"^^xsd:integer	"8"^^xsd:integer	"2"^^xsd:integer	"true"^^xsd:boolean	"ex:AtRiskStudent"
8	ex:Student_1202	"5"^^xsd:integer	"3"^^xsd:integer	"2"^^xsd:integer	"true"^^xsd:boolean	"ex:AtRiskStudent"
9	ex:Student_1206	"14"^^xsd:integer	"12"^^xsd:integer	"2"^^xsd:integer	"true"^^xsd:boolean	"ex:AtRiskStudent"
10	ex:Student_1209	"6"^^xsd:integer	"4"^^xsd:integer	"2"^^xsd:integer	"true"^^xsd:boolean	"ex:AtRiskStudent"

Figure 14: Validation of Reasoning Rule. The query successfully flags students for the new 'AtRiskStudent' class based on computed performance metrics

X. Bonus: End User Demo

To demonstrate the practical utility of the Knowledge Graph beyond raw SPARQL queries, we developed a prototype **"Student Retention Dashboard."**

Figure 16 illustrates the output of this Python-based application. The tool connects to the GraphDB endpoint, executes the complex "At-Risk" reasoning rules (defined in CQ7 and CQ1), and presents Academic Advisors with a prioritized, human-readable list of students requiring immediate intervention.

Architecture: Python RDFLib Wrapper → SPARQL Endpoint → Pandas DataFrame.

Value: This dashboard bridges the gap between Semantic Web technology and end-user decision-making, allowing non-technical staff to act on the "Academic Shock" and "Socio-Economic" insights derived from the ontology.

```
Loading Knowledge Graph... (This may take a few seconds)
Graph Loaded! 123919 statements found.

======================================
   STUDENT RETENTION DASHBOARD
======================================
Analyzing At-Risk Candidates based on Rule Q7...

STUDENT              | DEBT   | FAILURES   | STATUS
-----------------------------------------------------
Student 1494         | YES    | 7          | 🔴 HIGH RISK
Student 1568         | YES    | 7          | 🔴 HIGH RISK
Student 1828         | YES    | 7          | 🔴 HIGH RISK
Student 2407         | YES    | 7          | 🔴 HIGH RISK
Student 2854         | YES    | 7          | 🔴 HIGH RISK
Student 2876         | YES    | 7          | 🔴 HIGH RISK
Student 3116         | YES    | 7          | 🔴 HIGH RISK
Student 3175         | YES    | 7          | 🔴 HIGH RISK
Student 35           | YES    | 7          | 🔴 HIGH RISK
Student 3526         | YES    | 7          | 🔴 HIGH RISK

======================================
```

Figure 15: CLI Output of the Student Retention Dashboard. The application successfully flags students like Student_192 (Academic Shock) and Student_882 (Debt Risk) for advisor review.

XI. Tools & Libraries

We employed the following stack to achieve the project milestones:

Python Libraries:

- **kagglehub:** For programmatic dataset ingestion.
- **pandas:** For initial data manipulation and cleaning.
- **rdflib:** For Ontology creation, A-Box generation, and RDF serialization.
- **os:** For file handling and path management.

Software Tools:

- **WebVOWL:** Utilized for ontology structure visualization and logic validation.
- **GraphDB:** Served as the Triple Store and SPARQL endpoint hosting.
- **Python:** Used for the ETL pipeline and the End-User Dashboard logic.