Laboratory: Rule Discovery

Pablo Mollá Chárlez

February 4, 2025

Contents

1 How to Evaluate Rules Using an SQL Interface

The goal of this section is to understand how to translate the computation of the support and confidence metrics of a rule using an SQL interface. We consider a generic rule R of the form:

$$R : B(\alpha, \beta) \implies r(\alpha, \beta)$$

where:

- B(α, β) represents the body of the rule, which can include one or more conditions.
- r(α, β) corresponds to the head relation of the rule.

1.1 Metric Formulas

The two main metrics used to evaluate a rule are defined as follows:

- Support: The support of a rule R measures the number of facts in the knowledge base $\mathcal{K}$ that validate the rule R. Mathematically, it is expressed by the following formula:

$$\text{support(R)} = |\{p : (\mathcal{K} \wedge R \vDash p) \wedge p \in \mathcal{K}\}|$$

- Confidence: The confidence of a rule R represents the proportion of correct predictions among all predictions made by the rule. It is given by:

$$\text{confidence(R)} = \frac{\text{support(R)}}{\text{support(R)}+|\text{cex(R)}|}$$

  where:

  - **support(R)** is the number of correct predictions (true positives).
  - **cex(R)** represents the counterexamples, i.e., predictions made by R that are not valid in the knowledge base K.

  Note: Confidence metrics vary depending on the adopted hypothesis. Therefore, we define:

  - **owa-conf:** Confidence based on the $\boxed{\text{open-world assumption}}$
  - **cwa-conf:** Confidence based on the $\boxed{\text{closed-world assumption}}$
  - **pca-conf:** Confidence based on the $\boxed{\text{partial completeness assumption}}$

1.2 Exercise: Evaluating a Rule on Wikidata

Objective: Apply the concepts of support and confidence to evaluate a rule based on data from Wikidata. Consider the following rule:

$$R : \text{positionHeld(x, UKPrimeMinister)} \implies \text{memberOf(x, BullingdonClub)}$$

SPARQL Query: UK Prime Ministers Who Were Not Members of the Bullingdon Club

```
SELECT DISTINCT ?primeMinister ?primeMinisterName
WHERE {
?primeMinister p:P39 ?statement.
?statement ps:P39 wd:Q14211. # Position:UK Prime Minister
FILTER NOT EXISTS {
    primeMinister wdt:P463 wd:Q469039. #Not a member of Bullingdon Club
}
OPTIONAL {
    ?primeMinister rdfs:label ?primeMinisterName.
    FILTER (LANG(?primeMinisterName ) = "en"). #Retrieve labels in English
    }
}
```

Testing Platforms: The Wikidata and YAGO platforms provide online SPARQL endpoints:

★ Wikidata SPARQL Endpoint ★ YAGO SPARQL Endpoint

To explore the predicates available on Wikidata, consult the complete list of properties: Wikidata Property List.

1.3 Solution

We have the following rule:

$$R: \text{positionHeld(x, UKPrimeMinister)} \implies \text{memberOf(x, BullingdonClub)}$$

and we want to know how well that rule holds in Wikidata. This rule basically states:

"All UK Prime Ministers are members of the Bullingdon Club."

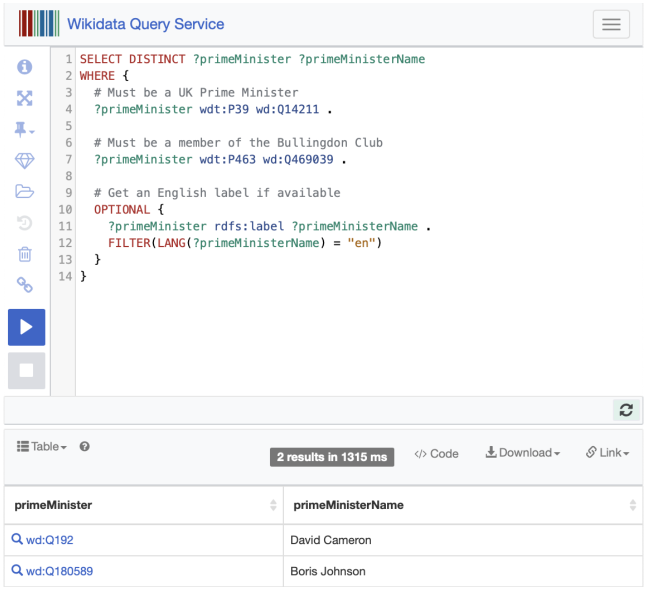
We want to measure how true that rule actually is in Wikidata, and to do so, we need to apply the concepts of support and confidence. To find the support, we need to SPARQL query the true predictions, and we can use the following query:

```
SELECT DISTINCT ?primeMinister ?primeMinisterName
WHERE {
  # Must be a UK Prime Minister
  ?primeMinister wdt:P39 wd:Q14211 .

  # Must be a member of the Bullingdon Club
  ?primeMinister wdt:P463 wd:Q469039 .

  # Get an English label if available
  OPTIONAL {
    ?primeMinister rdfs:label ?primeMinisterName .
    FILTER(LANG(?primeMinisterName) = "en")
  }
}
```

The previous query produces 2 results, therefore the support is: $Support(R) = 2$.



```
SELECT DISTINCT ?primeMinister ?primeMinisterName
WHERE {
  # Must be a UK Prime Minister
  ?primeMinister wdt:P39 wd:Q14211 .

  # Must be a member of the Bullingdon Club
  ?primeMinister wdt:P463 wd:Q469039 .

  # Get an English label if available
  OPTIONAL {
    ?primeMinister rdfs:label ?primeMinisterName .
    FILTER(LANG(?primeMinisterName) = "en")
  }
}
```

primeMinister	primeMinisterName
wd:Q192	David Cameron
wd:Q180589	Boris Johnson

Figure 1: Support Query

Now, we need to compute the confidence of such rule, thus we need to determine the **cex(R)**, which is the **counterexamples** of the rule **R**, meaning the cases where the rule's prediction fails. The confidence is given by the formula:

$$\text{confidence(R)} = \frac{\text{support(R)}}{\text{support(R)}+|\text{cex(R)}|}$$

In order to determine the **counterexamples**, we need to decide which prime ministers are considered as **counterexamples**. In other words, "which prime ministers does the rule claim should be members of the Bullingdon Club, but in fact are not members, according to the knowledge base in Wikidata". We need to consider 3 different assumptions:

1. **Closed-World Assumption (CWA)**

   Under CWA, if the person is not stated to be a member, then he definitely is not a member. So every prime minister which is not in our support query is not a member. We can find all "non-members" simply by looking for prime ministers where there is no wdt:P463 (member of) = Q469039 (Bullingdon Club) statement. Then, the count of these results will be the **counterexamples cex(R)** under CWA.

```
SELECT DISTINCT ?primeMinister ?primeMinisterName
WHERE {
  # Must be a UK Prime Minister
  ?primeMinister wdt:P39 wd:Q14211 .

  # FILTER NOT EXISTS => "No membership in Bullingdon"
  FILTER NOT EXISTS {
    ?primeMinister wdt:P463 wd:Q469039 .
  }

  # Get an English label if available
  OPTIONAL {
    ?primeMinister rdfs:label ?primeMinisterName .
    FILTER(LANG(?primeMinisterName) = "en")
  }
}
```

   The previous SPARQL query produces 79 results, meaning that there are 79 prime ministers who are not members of the Bullingdon Club. Then,

$$\text{confidence}_{\text{CWA}}(R) = \frac{\text{support}(R)}{\text{support}(R) + \text{cex}(R)} = \frac{2}{2+79} \approx 0.0246.$$

2. **Open-World Assumption (OWA)**

   Under OWA, if a prime minister is not stated to be a member, that does not automatically mean "not a member" - it might just be unknown. So we only count as a counterexample if the data explicitly says the primer minister is not a member. In Wikidata, we do not usually have "negative" statements such as "X is definitely not in the Bullingdon Club." So in practice, your cex(R) under OWA may well be zero, unless Wikidata has some specific "excluded" membership property.

```
SELECT DISTINCT ?primeMinister ?primeMinisterName
WHERE {
  ?primeMinister wdt:P39 wd:Q14211 .   # is a UK Prime Minister

  # -- Hypothetical: Some kind of explicit "excluded membership" statement
  # e.g. a specialized property or a "no value" statement for wdt:P463 = Q469039

  ?primeMinister p:P463 ?membershipStatement .
  ?membershipStatement ps:P463 wd:Q469039 .
  ?membershipStatement wikibase:rank  wd:deprecatedRank .

  OPTIONAL {
    ?primeMinister rdfs:label ?primeMinisterName .
    FILTER(LANG(?primeMinisterName) = "en")
  }
}
```

That query would return all prime ministers that are explicitly flagged as "not members". In most Wikidata contexts, though, the database does not store such explicit negatives. You'll probably get zero results, meaning **cex(R)** under OWA = 0. Then:

$$\text{confidence}_{\text{OWA}}(R) = \frac{\text{support}(R)}{\text{support}(R) + \text{cex}(R)} = \frac{\text{support}(R)}{\text{support}(R) + 0} = 1.0 \quad \text{(if no explicit negation is found)}$$

3. **Partial-Completeness Assumption (PCA)**

   Under PCA, we only consider an entity as "negative" if Wikidata does list some membership(s) for that entity, but not the Bullingdon Club. If Wikidata is completely silent about all clubs for that prime minister, we skip them (we say "unknown," not negative). Thus we need two pieces:

   (a) Who are the prime ministers that have some wdt:P463 membership statement?

   (b) Among those, which do not mention **wd:Q469039**?

   Here's a query that returns prime ministers who definitely have membership data for at least one organization, but not Q469039:

```
SELECT DISTINCT ?primeMinister ?primeMinisterName
WHERE {
  # must be a UK Prime Minister
  ?primeMinister wdt:P39 wd:Q14211 .

  # has at least one membership in some org
  ?primeMinister wdt:P463 ?someOrg .

  # but that org is not Bullingdon
  FILTER(?someOrg != wd:Q469039)

  # Now, we also want to ensure that the PM is never stated
  # to be in Bullingdon. So we do a NOT EXISTS check:
  FILTER NOT EXISTS {
    ?primeMinister wdt:P463 wd:Q469039 .
  }

  OPTIONAL {
    ?primeMinister rdfs:label ?primeMinisterName .
    FILTER(LANG(?primeMinisterName) = "en")
  }
}
```

   The count of these results corresponds to the **cex(R)** under PCA. Using Wikidata, we obtain 39 prime ministers satisfying the previous conditions. They are prime ministers for whom we know at least one membership (so presumably the membership data is "partially complete"), yet we see no membership in Bullingdon. Then:

$$\text{confidence}_{\text{PCA}}(R) = \frac{\text{support}(R)}{\text{support}(R) + \text{cex}(R)} = \frac{2}{2 + 39} = 0.0488.$$

   Finally, note that any prime ministers who has no membership statements at all for **wdt:P463** simply does not appear in that PCA counterexample list. That is exactly how we avoid penalizing "unknown membership" under PCA.

2 Computing Metrics on a Given Graph

Consider the relations created and createdBy, along with the following knowledge graph: The thick red line

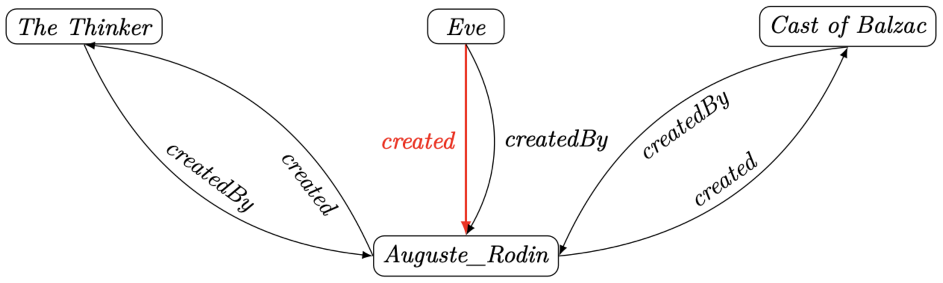


Figure 2: Relations **created** and **createdBy**

represents an erroneous fact, extracted incorrectly instead of its inverse. The domains of the two relations should normally be disjoint, as these relations are inverses of each other and are not symmetric. However, due to this error, the domains of **createdBy** and **created** share the element **Eve** in common.

1. Compute:

   - **pca-conf(createdBy(x, y) $\Longrightarrow$ created(x, y)) =?**
   - **pca-conf(created(x, y) $\Longrightarrow$ createdBy(x, y)) =?**

2. What do you observe?

   A concise way to see what happens is to list exactly which “created” and “createdBy” facts appear in the little graph (including the one erroneous triple). Then we can count support and counterexamples under the **partial-completeness assumption**.

   - **The Facts in the Graph**

     From the figure, we have the following triples (the thick red one is erroneous but does appear in the Knowledge Base):

$$\begin{aligned}
&TheThinker \Longrightarrow createdBy \Longrightarrow AugusteRodin\\
&CastofBalzac \Longrightarrow createdBy \Longrightarrow AugusteRodin\\
&Eve \Longrightarrow createdBy \Longrightarrow AugusteRodin\\
&AugusteRodin \Longrightarrow created \Longrightarrow TheThinker\\
&AugusteRodin \Longrightarrow created \Longrightarrow CastofBalzac\\
&Eve \Longrightarrow created \Longrightarrow AugusteRodin \text{ (erroneous but present)}
\end{aligned}$$

     Hence the domain of “createdBy” includes three works (“The Thinker,” “Cast of Balzac,” and “Eve”), and the domain of “created” (artist $\Longrightarrow$ artwork) should have been just Auguste Rodin, but because of the error, “Eve” also appears as a creator. Next, we will study once again the 3 assumptions as practice:

     – $\boxed{R_1 : \textbf{createdBy(x, y)} \Longrightarrow \textbf{created(x, y)}}$

       (a) Open-World Assumption (OWA):

       ∗ **Support**: The support of the rule corresponds to the number of (x, y) for which both createdBy(x,y) and created(x,y) appear in the Knowledge Base, which is 1, because we have for **x** = Eve, **y** = Auguste Rodin:

         · createdBy(Eve, Auguste_Rodin) and

         · created(Eve, Auguste_Rodin) (the erroneous triple for us but not for the system).

  - ∗ **Counterexample**: If we call $B \vDash \textit{createdBy}$ (body) and $H \vDash \textit{created}$ (head), the counterexample corresponds to: cex_OWA = $|B \cap \neg H|$, then it's easy to deduce that there are no relations stating **not created**, therefore cex_OWA = $|B \cap \varnothing| = |\varnothing| = 0$.
  - ∗ **Confidence:** Confidence_OWA = $\frac{\text{support}(R_1)}{\text{support}(R_1)+\text{cex}_{OWA}(R_1)} = \frac{1}{1+0} = 1.0$

(b) Closed-World Assumption (CWA):
  - ∗ **Support**: The support of the rule still is 1.
  - ∗ **Counterexample**: If we call $B \vDash \textit{createdBy}$ (body) and $H \vDash \textit{created}$ (head), the counterexample corresponds to: cex_CWA = $|\{(x,y) \in B : (x,y) \notin H\}|$, then it's easy to deduce that there are 2 relations satisfying it, therefore cex_CWA = 2.
  - ∗ **Confidence:** Confidence_CWA = $\frac{\text{support}(R_1)}{\text{support}(R_1)+\text{cex}_{CWA}(R_1)} = \frac{1}{1+2} = 0.33$

(c) Partial-Completeness Assumption (PCA):
  - ∗ **Support**: The support of the rule still is 1.
  - ∗ **Counterexample**: If we call $B \vDash \textit{createdBy}$ (body) and $H \vDash \textit{created}$ (head), the counterexample corresponds to: cex_PCA = $|\{(x,y) \in B : x \in H \wedge (x,y) \notin H\}|$, then it's easy to deduce that there is no relation satisfying it (from the system point of view, the erroneous triple is not wrong even if we know it is), therefore cex_CWA = 0.
  - ∗ **Confidence:** Confidence_CWA = $\frac{\text{support}(R_1)}{\text{support}(R_1)+\text{cex}_{CWA}(R_1)} = \frac{1}{1+0} = 1.0$

– $\boxed{R_2 : \textbf{created(x, y)} \implies \textbf{createdBy(x, y)}}$

(a) Open-World Assumption (OWA):
  - ∗ **Support**: The support of the rule corresponds to the number of (x,y) for which both created(x,y) and createdBy(x,y) appear in the Knowledge Base, which is 1, because we have for **x** = Eve, **y** = Auguste Rodin:
    - · created(Eve, Auguste_Rodin) (the erroneous triple for us but not for the system) and
    - · createdBy(Eve, Auguste_Rodin)
  - ∗ **Counterexample**: If we call $B \vDash \textit{created}$ (body) and $H \vDash \textit{createdBy}$ (head), the counterexample corresponds to: cex_OWA = $|B \cap \neg H|$, then it's easy to deduce that there are no relations stating **not createdBy**, therefore cex_OWA = $|B \cap \varnothing| = |\varnothing| = 0$.
  - ∗ **Confidence:** Confidence_OWA = $\frac{\text{support}(R_1)}{\text{support}(R_1)+\text{cex}_{OWA}(R_1)} = \frac{1}{1+0} = 1.0$

(b) Closed-World Assumption (CWA):
  - ∗ **Support**: The support of the rule still is 1.
  - ∗ **Counterexample**: If we call $B \vDash \textit{created}$ (body) and $H \vDash \textit{createdBy}$ (head), the counterexample corresponds to: cex_CWA = $|\{(x,y) \in B : (x,y) \notin H\}|$, then it's easy to deduce that there are 2 relations satisfying it, therefore cex_CWA = 2.
  - ∗ **Confidence:** Confidence_CWA = $\frac{\text{support}(R_1)}{\text{support}(R_1)+\text{cex}_{CWA}(R_1)} = \frac{1}{1+2} = 0.33$

(c) Partial-Completeness Assumption (PCA):
  - ∗ **Support**: The support of the rule still is 1.
  - ∗ **Counterexample**: If we call $B \vDash \textit{created}$ (body) and $H \vDash \textit{createdBy}$ (head), the counterexample corresponds to: cex_PCA = $|\{(x,y) \in B : x \in H \wedge (x,y) \notin H\}|$, then it's easy to deduce that there is no relation satisfying it (from the system point of view, the erroneous triple is not wrong even if we know it is), therefore cex_CWA = 0.
  - ∗ **Confidence:** Confidence_CWA = $\frac{\text{support}(R_1)}{\text{support}(R_1)+\text{cex}_{CWA}(R_1)} = \frac{1}{1+0} = 1.0$

In this toy example, both inverse rules end up having exactly the same confidence under all three assumptions:

- OWA and PCA each yield a confidence of 1.0 for both rules, because only the single (erroneous) pair (Eve, Rodin) is in both relations, and there are no "explicit negatives" or "partially complete subjects" that force counterexamples.

- CWA yields a confidence of 1/3 for both rules, since each rule has the same number of "body-only" pairs (2) and one "body+head" pair (1), giving $1/(1+2) = 1/3$.

So even though the two rules are truly inverse (and one is obviously incorrect), they end up with the same confidence values under each assumption given the particular facts and error in this KB.

3 Detection of Subproperty Relationships

In the context of description logic, the concept of a subproperty is essential for modeling hierarchical inclusion between properties. If a property r_1 is a subproperty of r_2 (denoted $r_1 \subseteq r_2$), it means that all instances of r_1 are also instances of r_2. In other words, the subproperty relationship expresses that r_1 is a specialization of r_2, thereby formalizing an inheritance relationship between the two properties. The subproperty relationship, denoted $r_1 \subseteq r_2$, can be expressed as a Horn clause as follows:

$$r_1(x, y) \implies r_2(x, y)$$

This rule states that if a pair (x, y) satisfies the relation $r_1(x, y)$, then it must also satisfy the relation $r_2(x, y)$. In other words, every instance of the property r_1 implies a corresponding instance of the property r_2, thus modeling the hierarchical inclusion between the two properties. Consider the following property:

Property: Let r_1 and r_2 be two functional relations with $r_1 \subseteq r_2$. Then,

$$\text{pca-conf}(r_1 \subseteq r_2) = 1 \text{ and pca-conf}(r_2 \subseteq r_1) = 1.$$

4 Query Expressiveness

4.1 Objective: Comparing SPARQL and SQL

The goal of this section is to analyze the SQL equivalents of the **OPTIONAL** clause, **FILTER NOT EXIST** clause, and path patterns found in SPARQL queries. We assume that the knowledge base is stored in a single table, **KB**, with three attributes:

S (subject), **P (predicate)**, and **O (object)**.

This schema, which may seem inefficient, has proven to be highly performant. In particular, it is the schema adopted by the RDF3X system, which has demonstrated superiority compared to a partitioned schema where each predicate is stored in a separate table **P(S, O)—P for predicate**, **S for subject**, and **O for object**.

4.2 Exercises

Translate the following SPARQL queries into SQL equivalents for the **OPTIONAL** clause and path queries.

1. **Query 1:**

```
SELECT DISTINCT ?scientist
WHERE {
    ?scientist rdf:type yago:Scientist.
    FILTER NOT EXISTS {
        ?scientist schema:award yago:NobelPrize.
    }
}
```

SQL Version of Query 1:

```
SELECT DISTINCT t1.subject AS scientist
FROM Triples AS t1
WHERE t1.predicate = 'rdf:type'
  AND t1.object = 'yago:Scientist'
  AND NOT EXISTS (
    SELECT 1
    FROM Triples AS t2
    WHERE t2.subject = t1.subject
      AND t2.predicate = 'schema:award'
      AND t2.object = 'yago:NobelPrize'
  );
```

2. **Query 2:**

```
SELECT ?pName ?date
WHERE {
    ?p schema:award yago:NobelPrize.
    ?p rdfs:label ?pName .
    OPTIONAL { ?p schema:birthDate ?date.}}
```

SQL Version of Query 2:

```
SELECT DISTINCT lbl.object AS pName, bdate.object AS date
FROM Triples AS aw
JOIN Triples AS lbl
  ON lbl.subject = aw.subject
  AND lbl.predicate = 'rdfs:label'
LEFT JOIN Triples AS bdate
  ON bdate.subject = aw.subject
  AND bdate.predicate = 'schema:birthDate'
WHERE aw.predicate = 'schema:award'
AND aw.object = 'yago:NobelPrize';
```

3. **Query 3:**

```
SELECT ?Nname
WHERE {
    ?p rdfs:label ?pName.
    ?p rdf:type yago:Scientist.}
```

SQL Version of Query 3:

```
SELECT DISTINCT label.object AS Nname
FROM Triples AS t
JOIN Triples AS label
  ON t.subject = label.subject
  AND label.predicate = 'rdfs:label'
WHERE t.predicate = 'rdf:type'
  AND t.object    = 'yago:Scientist';
```

4. **Query 4:**

```
SELECT ?pName
WHERE {
    ?p rdfs:label ?pName.
    ?p rdf:type/rdfs:subClassOf yago:Scientist.}
```

SQL Version of Query 4:

```
SELECT DISTINCT lbl.object AS pName
FROM Triples AS type
JOIN Triples AS lbl
   ON lbl.subject = type.subject
   AND lbl.predicate = 'rdfs:label'
JOIN Triples AS sc
   ON sc.subject = type.object
   AND sc.predicate = 'rdfs:subClassOf'
   AND sc.object    = 'yago:Scientist'
WHERE type.predicate = 'rdf:type';
```

4.3 Testing the Solutions

We suggest using the online SQL database tool: https://sqliteonline.com/.

Creating and Dropping the Table: Before inserting data, we need to create the table. To avoid duplication errors, drop the table if it already exists using:

- **Dropping the Table:**

```
DROP TABLE IF EXISTS KB;
```

- **Creating the KB Table:**

```
CREATE TABLE KB (
S TEXT,
P TEXT,
O TEXT):
```

- **Inserting Data:** Insert the following data into the table.

```
INSERT INTO KB (S, P, O) VALUES
('yago:Marie_Curie', 'rdfs:label', '"Marie Curie"@en'),
('yago:Max_Planck', 'rdfs:label', '"Max Planck"@en'),
('yago:Marie_Curie', 'rdf:type', 'yago:Scientist'),
('yago:Dmitri_Mendeleev', 'rdf:type', 'yago:Scientist'),
('yago:Max_Planck', 'rdf:type', 'yago:TheoreticalPhysicist'),
('yago:TheoreticalPhysicist', 'rdfs:subClassOf', 'yago:Physicist'),
('yago:Physicist', 'rdfs:subClassOf', 'yago:Scientist'),
('yago:Marie_Curie', 'schema:birthDate', '"1867-11-07"^^xsd:date'),
('yago:Marie_Curie', 'rdfs:label', '"Marie Curie"@fr'),
('yago:Max_Planck', 'rdfs:label', '"Max Planck"@de'),
('yago:Marie_Curie', 'schema:award', 'yago:NobelPrize'),
('yago:Marie_Curie', 'schema:award', 'yago:AlbertMedal'),
('yago:Marie_Curie', 'schema:award', 'yago:NobelPrize'),
('yago:Max_Planck', 'schema:award', 'yago:NobelPrize');
```

  These rows represent **RDF** facts expressed as triples (subject, predicate, object).

- **Verifying the Data:** After insertion, you can verify the table's content using:

```
SELECT  FROM KB;
```