

QuOnto: Querying Ontologies

Marco Ruzzi

Sapienza University of Rome
Roma, 26 ottobre 2011

Summary

- Introduction
- *DL-Lite*
- Architecture of the system
- Languages adopted
- Sample Ontology
- Queries
- The Demo

Introduction

- Expressive Description Logics: high computational complexity problems
- Need for tractable DLs
- *DL-Lite*
- ... tailored for knowledge bases with a huge ABox
- ... nice computational complexity: AC_0 in the size of ABox
- ... preserves enough expressive power to represent almost completely ER and UML class diagram.

DL-Lite syntax

Given the concept symbol C , the role symbol P and the attribute symbol U , we define the following concept expressions:

$$B ::= A \mid \exists Q \mid \delta(U), \quad C ::= B \mid \neg B \mid \exists Q.C$$

Role expressions:

$$Q ::= P \mid P^-, \quad R ::= Q \mid \neg Q$$

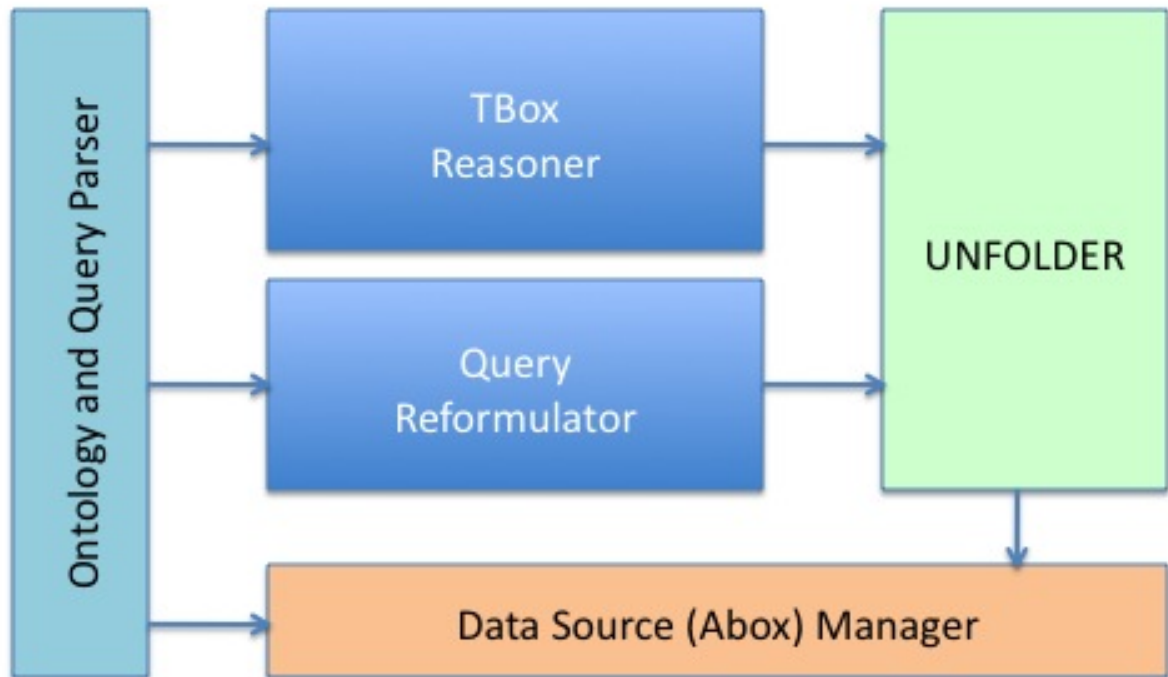
TBox Assertions:

$$B \sqsubseteq C \mid Q \sqsubseteq R \mid (\text{funct } R) \mid (\text{funct } U)$$

ABox Assertions (a , b and c are constants):

$$A(a) \mid P(a, b) \mid U(a, c)$$

QuOnto: Architecture of the system



5 / 22

Languages adopted by QuOnto

TBox: OWL2QL in Functional-style Syntax

- <http://www.w3.org/TR/owl2-syntax/>
- <http://www.w3.org/TR/owl-profiles/>

ABox: XML proprietary syntax
(*an example will be shown*)

Query: Conjunctive Queries in SPARQL

- <http://www.w3.org/TR/rdf-sparql-query/>

An example ontology: the TBox

Declaration(Class(Person)) (a class is a concept)
Declaration(Class(LivingBeing))
Declaration(Class(Pet))
Declaration(Class(Man))
Declaration(Class(Woman))
Declaration(Class(City))
Declaration(ObjectProperty(lives_in)) (...a role)
Declaration(ObjectProperty(has_mother))
Declaration(ObjectProperty(has_father))
Declaration(ObjectProperty(has_owner))
Declaration(DataProperty(name)) (... an attribute)

An example ontology: the TBox (2)

SubClassOf(Person LivingBeing) (**Person** \sqsubseteq **LivingBeing**)
SubClassOf(Pet LivingBeing)
SubClassOf(Man Person)
SubClassOf(Woman Person)
DisjointClasses(Man Woman) (**Man** \sqsubseteq \neg **Woman**)
DisjointClasses(Pet Person)
SubClassOf(Person DataSomeValuesFrom(name rdfs:Literal))
SubClassOf(Pet DataSomeValuesFrom(name rdfs:Literal))
SubClassOf(City DataSomeValuesFrom(name rdfs:Literal))
ObjectPropertyDomain(lives_in Person) (\exists **lives_in** \sqsubseteq **Person**)
ObjectPropertyRange(lives_in City) (\exists **lives_in** \sqsubseteq **City**)
ObjectPropertyDomain(has_mother Person)
ObjectPropertyRange(has_mother Woman)
ObjectPropertyDomain(has_father Person)
ObjectPropertyRange(has_father Man)

An example ontology: the ABox

Concept ABox specification

```
SELECT object FROM Person  
→ Person(pers($object))
```

Role ABox specification

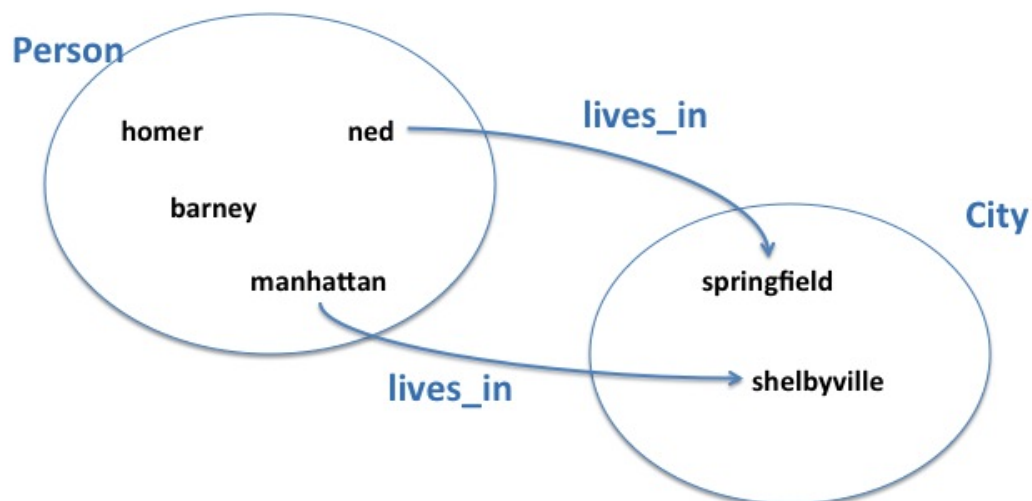
```
SELECT domain, range FROM lives_in  
→ lives_in(pers($domain), city($range))
```

Attribute ABox specification

```
SELECT domain, range FROM name  
→ name(pers($domain), $range)
```

An example ontology: the ABox

Person	lives_in	
Object	Domain	Range
homer	ned	springfield
barney	manhattan	shelbyville



Queries

Query: ask for all the LivingBeing represented in the ontology

FOL:

$$q = \{x \mid \text{LivingBeing}(x)\}$$

SPARQL:

```
select $X where {$X rdf:type :LivingBeing}
```

Query: ask for all persons and the cities they live in

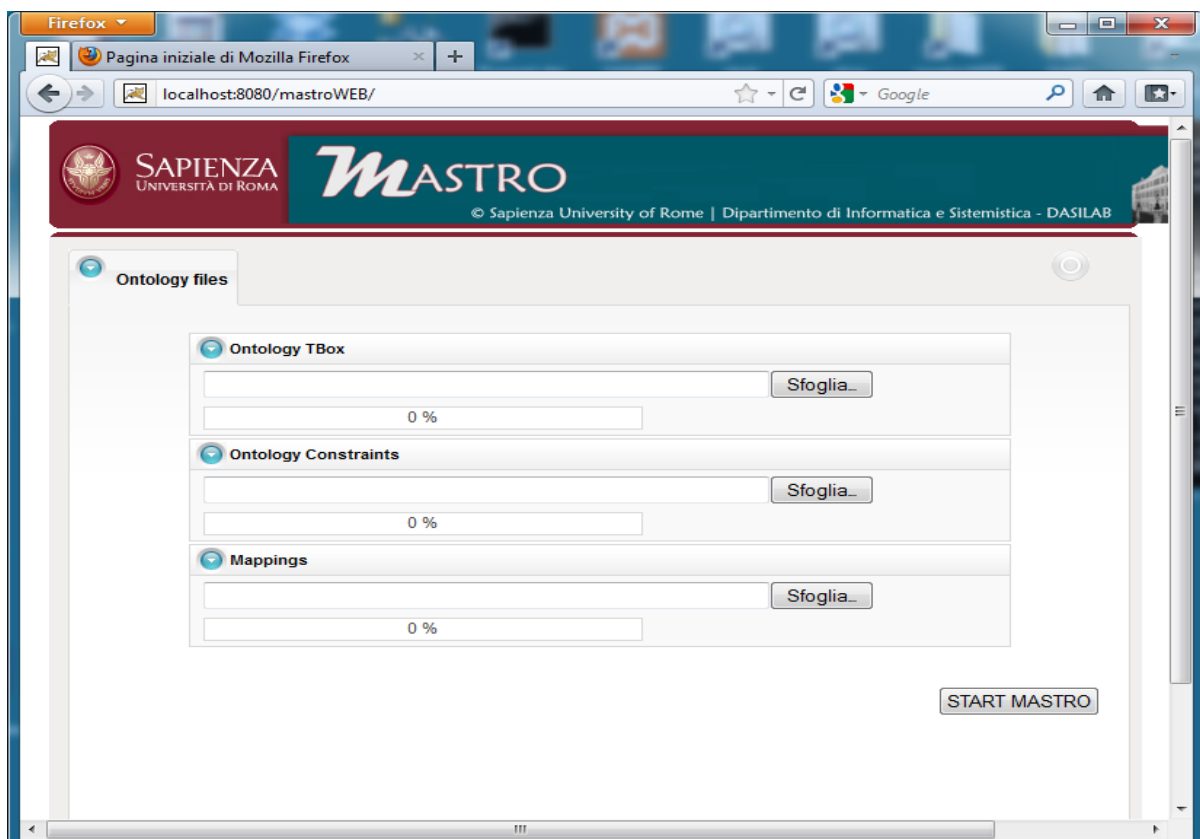
FOL:

$$q = \{x, y \mid \text{Person}(x) \wedge \text{lives_in}(x, y)\}$$

SPARQL:

```
select $X $Y where {$X rdf:type :Person . $X :lives_in $Y}
```

Screenshots: the home page of the system



The main page: ontology metrics

The screenshot shows the MASTRO web application interface. The header includes the Sapienza University of Rome logo and the MASTRO logo. Below the header, there is a navigation menu with tabs for Ontology Metrics, Concepts, Roles, Concept Attributes, Constraints, DataSource and Meppings, New Session, and Reasoning. The main content area displays a table of ontology metrics:

Active Ontology:	ONTOLOGY_NAME
Concepts Count:	6
Roles Count:	5
Concept Attributes Count:	1
Concept Inclusion Assertions Count:	4
Role Inclusion Assertions Count:	2
Concept Attribute Inclusion Assertions Count:	0
Identification Constraints Count:	0
EQL Constraints Count:	0
Denial Constraints Count:	0
Mapping Assertions Count:	5

Concepts hierarchy

The screenshot shows the MASTRO web application interface with the 'Concepts' tab selected. The main content area displays a 'Concepts Hierarchy' tree on the left and a list of concepts on the right. The hierarchy tree shows the following structure:

- City
- LivingBeing
 - Person
 - Man
 - Woman
 - Pet

The right panel shows the following concepts:

- Super Concepts**
 - DataSomeValuesFrom(name rdfs:Literal) LivingBeing
- Sub Concepts**
 - Man
 - ObjectSomeValuesFrom(InverseObjectProperty(has_owner) owl:Thing)
 - ObjectSomeValuesFrom(has_father owl:Thing)
 - ObjectSomeValuesFrom(has_mother owl:Thing)
 - ObjectSomeValuesFrom(lives in owl:Thing)
- Disjoint Concepts**
 - Pet
- Identification Constraints**

Roles hierarchy

Firefox
Pagina iniziale di Mozilla Firefox
localhost:8080/mastroWEB/pages/homepage.iface

SAPIENZA UNIVERSITÀ DI ROMA
MASTRO
© Sapienza University of Rome | Dipartimento di Informatica e Sistemistica - DASILAB

Ontology Metrics Concepts Roles Concept Attributes Constraints DataSource and Meppings TBox Reasoning Ontology Queries New Session System Check

Roles Hierarchy

- has_owner
- has_parent
- has_father
- has_mother
- lives_in

Functional: Inverse Functional:

Domain

Range

Super Roles

Sub Roles

Disjoint Roles

ABox specification through data mappings

Firefox
Pagina iniziale di Mozilla Firefox
localhost:8080/mastroWEB/pages/homepage.iface

SAPIENZA UNIVERSITÀ DI ROMA
MASTRO
© Sapienza University of Rome | Dipartimento di Informatica e Sistemistica - DASILAB

Ontology Metrics Concepts Roles Concept Attributes Constraints DataSource and Meppings TBox Reasoning Ontology Queries New Session System Check

Mapping Assertions RDBMS Source Schema Inspector RDBMS Queries

Mapping Head Filter:

Mapping Names	Head	Body
M1:Person M2:lives_in M3:has_mother M4:has_father M4:has_owner	Person(pers(object))	SELECT object FROM Person

javascript: |||

Data source schema inspector

Database Name: jdbc:sqlserver://localhost:1433;DatabaseName=Person
User name: sa

TABLE_CAT	TABLE_SCHEM	TABLE_NAME	TABLE_TYPE	REMARKS
Person	dbo	has_father	TABLE	
Person	dbo	has_mother	TABLE	
Person	dbo	has_owner	TABLE	
Person	dbo	lives_in	TABLE	
Person	dbo	Person	TABLE	
Person	INFORMATION_SCHEMA	CHECK_CONSTRAINTS	VIEW	
Person	INFORMATION_SCHEMA	COLUMN_DOMAIN_USAGE	VIEW	
Person	INFORMATION_SCHEMA	COLUMN_PRIVILEGES	VIEW	
Person	INFORMATION_SCHEMA	COLUMNS	VIEW	
Person	INFORMATION_SCHEMA	CONSTRAINT_COLUMN_USAGE	VIEW	

359 items found. Showing 10 elements, from 1 to 10. Page 1 of 36

Direct data source queries

SQL Query

```
select * from person
```

EXECUTE QUERY

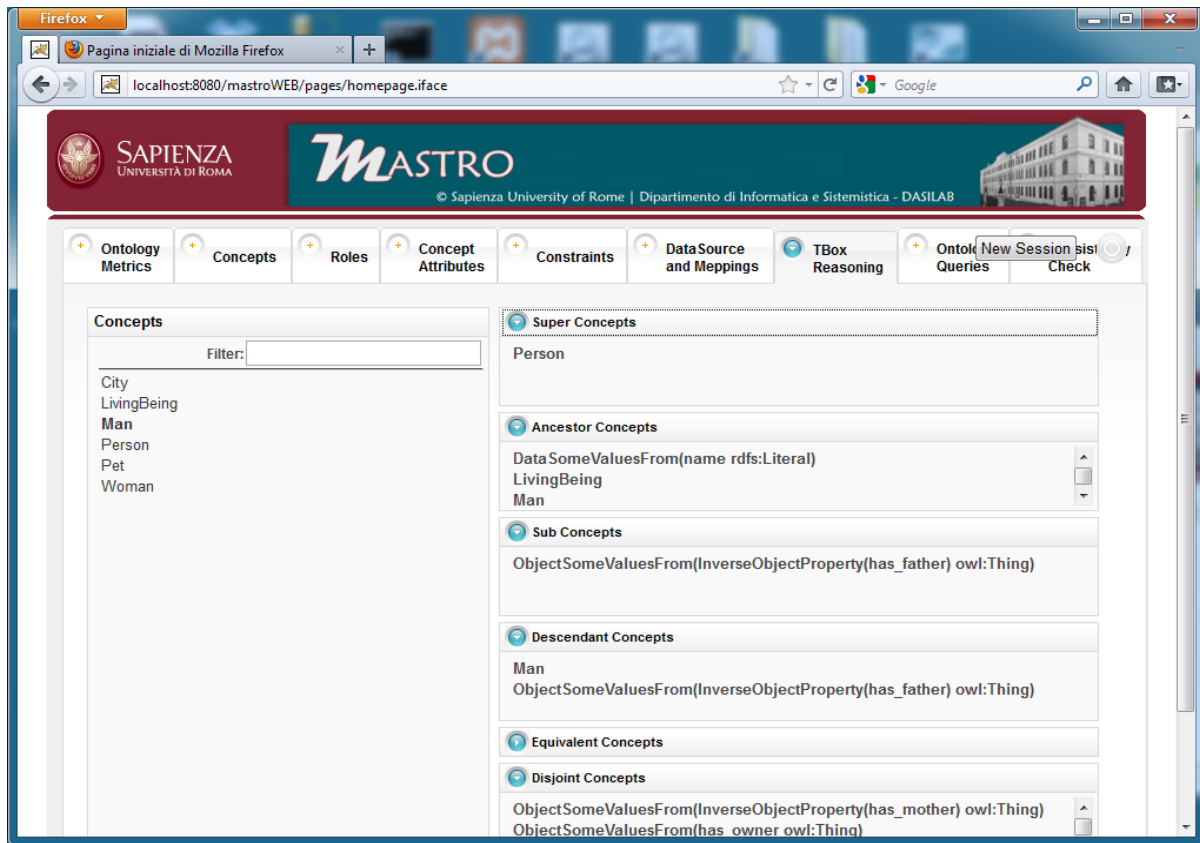
Query Result

object
homer

1 items found. Showing 5 elements, from 1 to 5. Page 1 of 1

Export as Excel Export as CSV

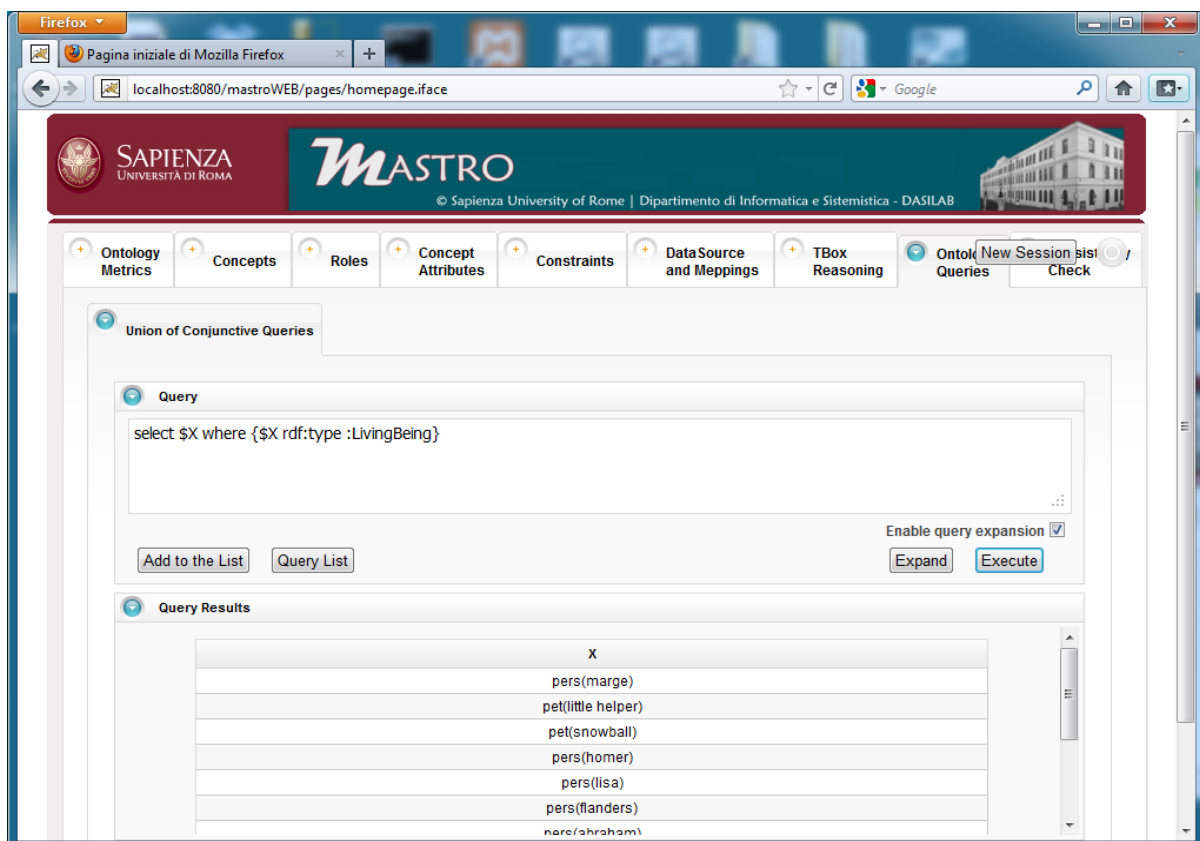
Tbox Reasoning



The screenshot shows the MASTRO web interface with the 'TBox Reasoning' tab selected. The left sidebar lists concepts: City, LivingBeing, Man, Person, Pet, Woman. The main content area shows a hierarchy of concepts:

- Super Concepts:** Person
- Ancestor Concepts:** DataSomeValuesFrom(name rdfs:Literal), LivingBeing, Man
- Sub Concepts:** ObjectSomeValuesFrom(InverseObjectProperty(has_father) owl:Thing)
- Descendant Concepts:** Man, ObjectSomeValuesFrom(InverseObjectProperty(has_father) owl:Thing)
- Equivalent Concepts:**
- Disjoint Concepts:** ObjectSomeValuesFrom(InverseObjectProperty(has_mother) owl:Thing), ObjectSomeValuesFrom(has_owner owl:Thing)

Ontology queries



The screenshot shows the MASTRO web interface with the 'Ontology Queries' tab selected. A query is entered in the 'Query' field:

```
select $X where {$X rdf:type :LivingBeing}
```

The 'Execute' button is highlighted. Below the query field, there are buttons for 'Add to the List', 'Query List', 'Expand', and 'Execute'. The 'Query Results' section shows a table with one column 'X' and several rows of instance names:

X
pers(marge)
pet(little helper)
pet(snowball)
pers(homer)
pers(lisa)
pers(flanders)
pers(abraham)

Query reformulation 1: expansion

The screenshot shows the MASTRO web interface in a Firefox browser. The page title is "Pagina iniziale di Mozilla Firefox" and the URL is "localhost:8080/mastroWEB/pages/homepage.iframe". The interface features a navigation bar with tabs for "Ontology Metrics", "Concepts", "Roles", "Concept Attributes", "Constraints", "DataSource and Meppings", "TBox Reasoning", "Ontol. Queries", "New Session", and "Sist. Check". The main content area is titled "Union of Conjunctive Queries" and contains a "Query" section with the text "select \$X where {\$X rdf:type :LivingBeing}". Below this is an "Expanded Query" section containing a list of SQL-like queries: "SELECT \$x0 WHERE {\$x0 rdf:type :Woman}", "SELECT \$x0 WHERE {\$X_UB_0 :has_mother \$x0}", "SELECT \$x0 WHERE {\$x0 :has_owner \$X_UB_1}", "SELECT \$x0 WHERE {\$x0 :has_father \$X_UB_2}", "SELECT \$x0 WHERE {\$x0 rdf:type :Man}", "SELECT \$x0 WHERE {\$x0 :lives_in \$X_UB_3}", "SELECT \$x0 WHERE {\$x0 rdf:type :Pet}", "SELECT \$x0 WHERE {\$X_UB_4 :has_father \$x0}", "SELECT \$x0 WHERE {\$X_UB_5 :has_owner \$x0}", and "SELECT \$x0 WHERE {\$x0 :lives_in \$X_UB_6}". There are buttons for "Add to the List", "Query List", "Expand", and "Execute". A checkbox for "Enable query expansion" is checked.

Query reformulation 2: unfolding

The screenshot shows the MASTRO web interface in a Firefox browser, similar to the previous one. The main content area is titled "Union of Conjunctive Queries" and contains a "Query" section with the text "select \$X where {\$X rdf:type :LivingBeing}". Below this is an "Unfolded Query" section containing a list of SQL-like queries: "Query 1: SELECT DISTINCT TABLE0.range FROM (SELECT domain, range FROM has_mother) AS TABLE0", "Query 2: SELECT DISTINCT TABLE0.domain FROM (SELECT domain, range FROM has_owner) AS TABLE0", "Query 3: SELECT DISTINCT TABLE0.domain FROM (SELECT domain, range FROM has_father) AS TABLE0", "Query 4: SELECT DISTINCT TABLE0.domain FROM (SELECT domain, range FROM lives_in) AS TABLE0", "Query 5: SELECT DISTINCT TABLE0.range FROM (SELECT domain, range FROM has_father) AS TABLE0", "Query 6: SELECT DISTINCT TABLE0.range FROM (SELECT domain, range FROM has_owner) AS TABLE0", "Query 7: SELECT DISTINCT TABLE0.domain FROM (SELECT domain, range FROM has_mother) AS TABLE0", and "Query 8: SELECT DISTINCT TABLE0.object FROM (SELECT object FROM Person) AS TABLE0".