QuOnto: Querying Ontologies

Marco Ruzzi

Sapienza University of Rome
Roma, 26 ottobre 2011

Summary

Introduction

- Expressive Description Logics: high computational complexity problems
- Need for tractable DLs
- *DL-Lite*
- ... tailored for knoledge 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:

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

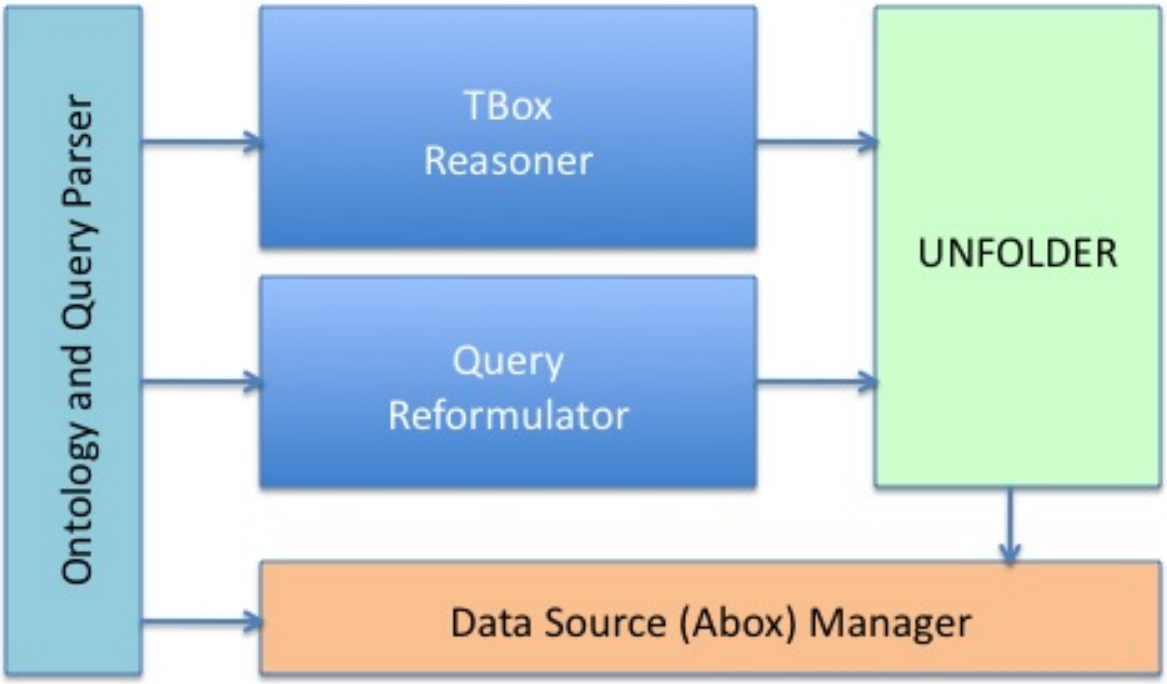
TBox Assertions:

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

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

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

QuOnto: Architecture of the system



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)

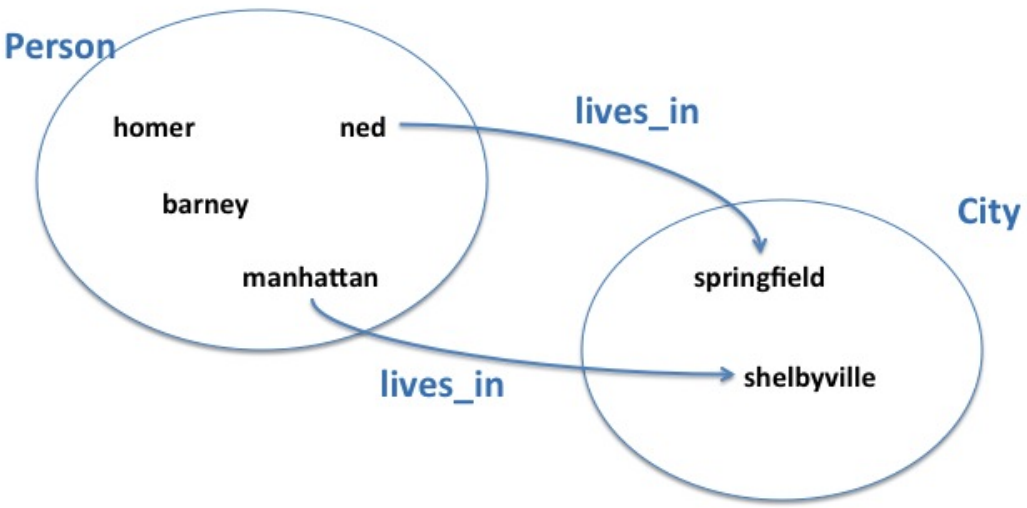
An example ontology: the ABox

Person

Object
homer
barney

lives_in

Domain	Range
ned	springfield
manhattan	shelbyville



Queries

Query: ask for all the LivingBeing represented in the ontlogy

FOL:

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

SPARQL:

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

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

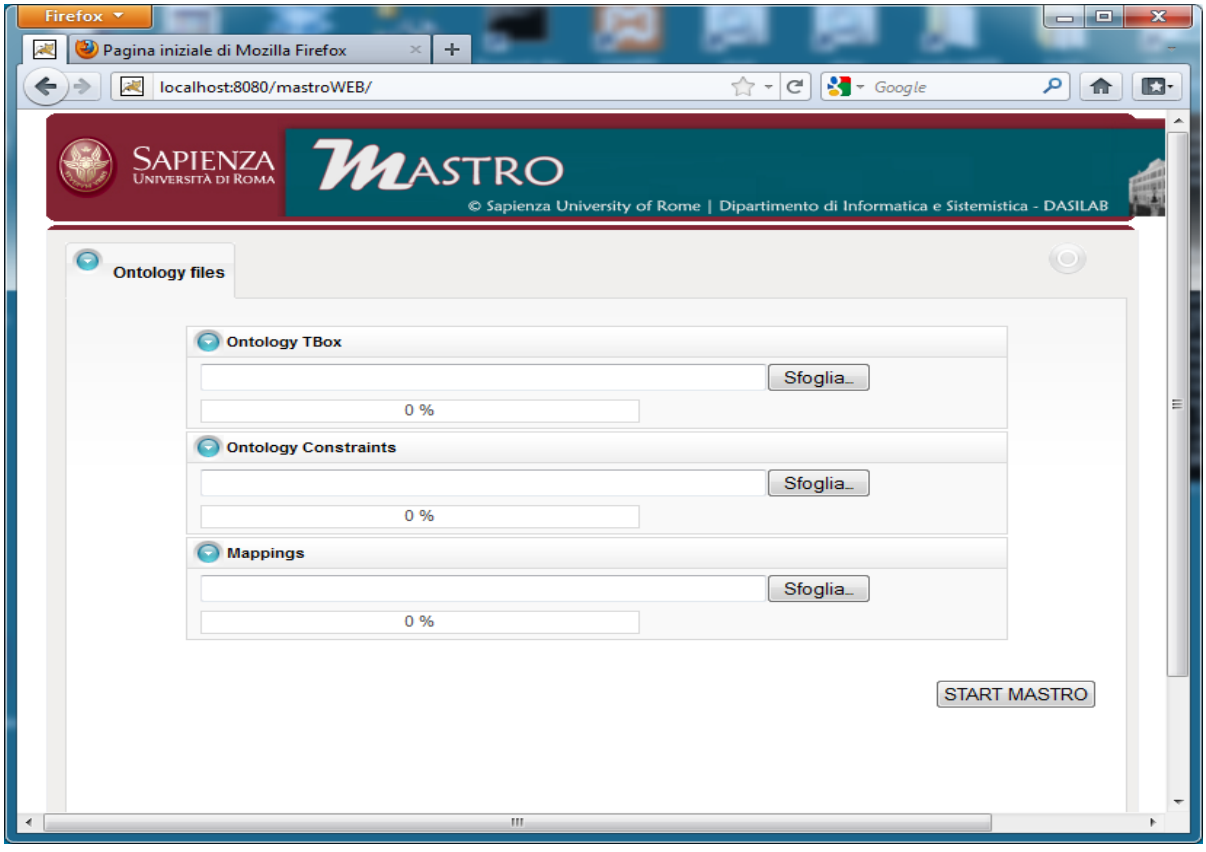
FOL:

$$q = \{x, y \mid Person(x) \wedge lives\_in(x, y)\}$$

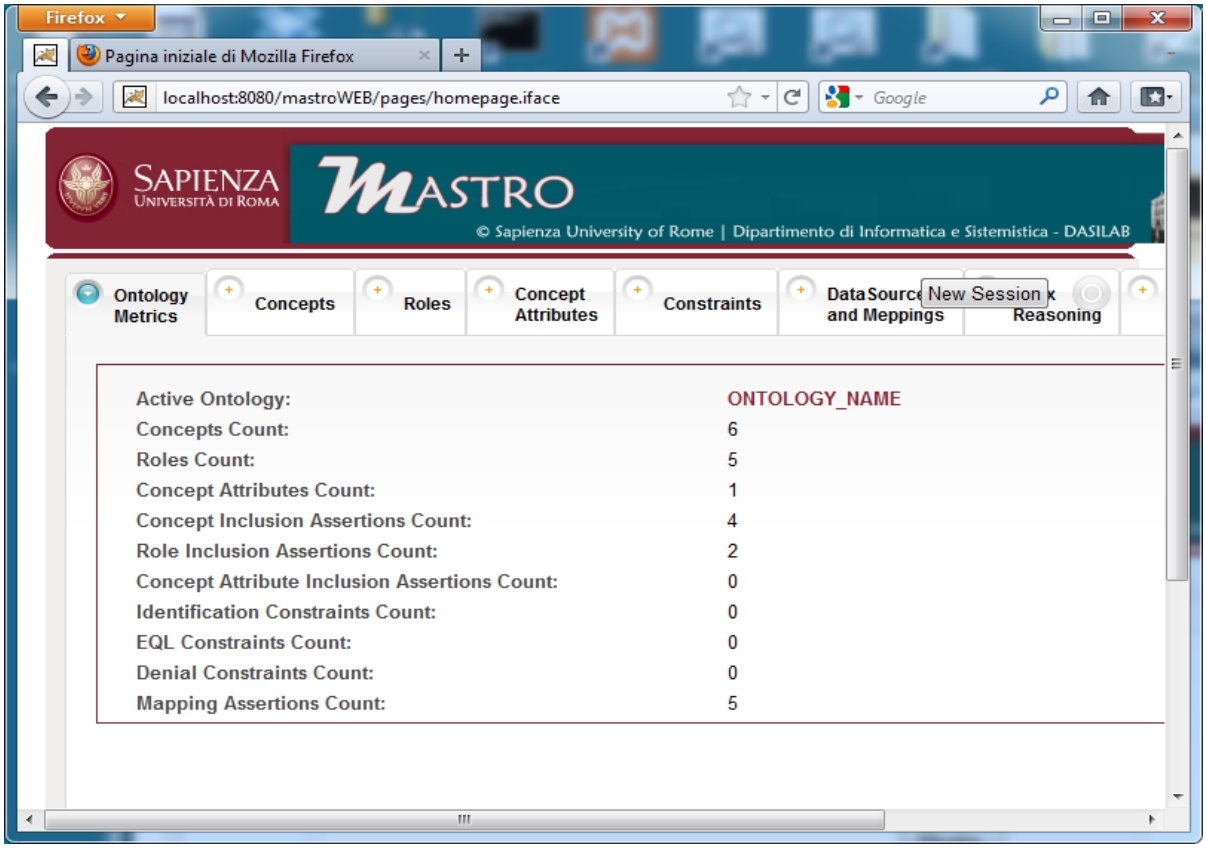
SPARQL:

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

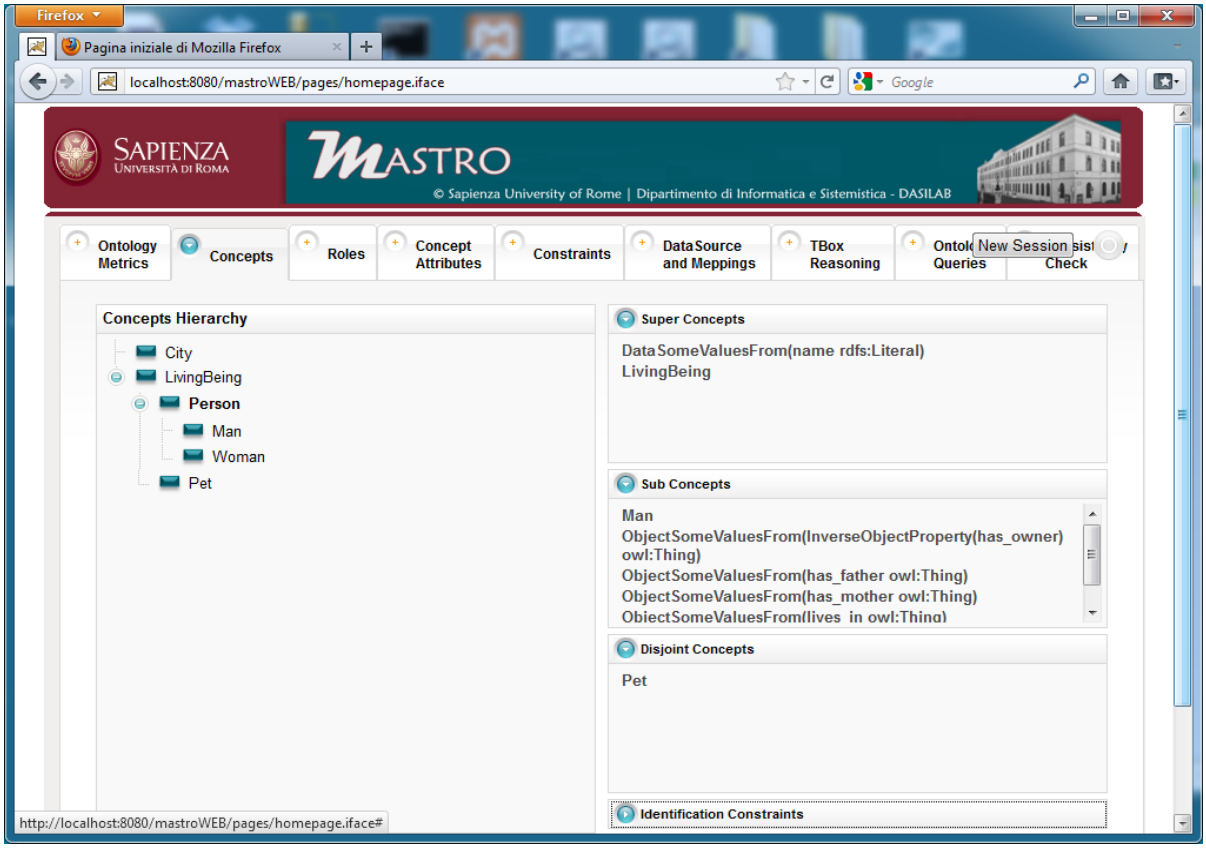
Screenshots: the home page of the system



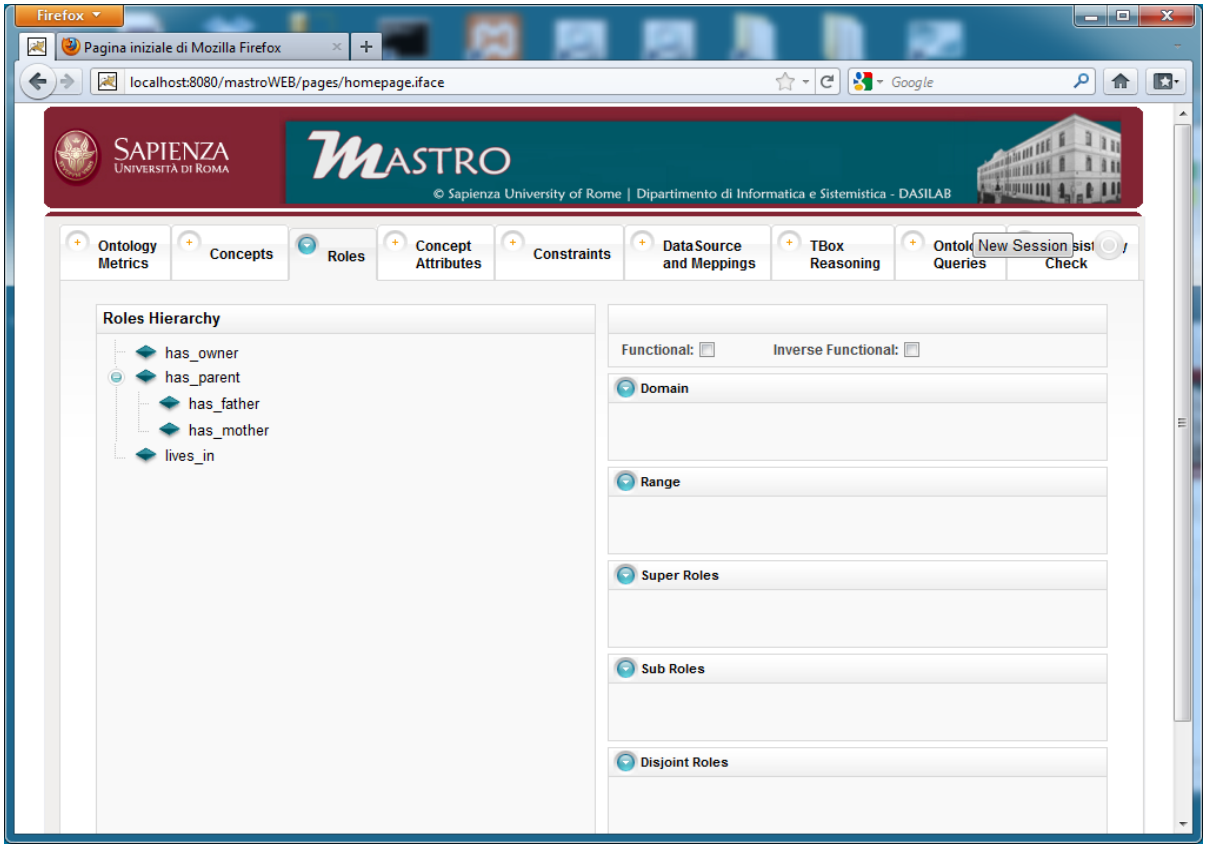
The main page: ontology metrics



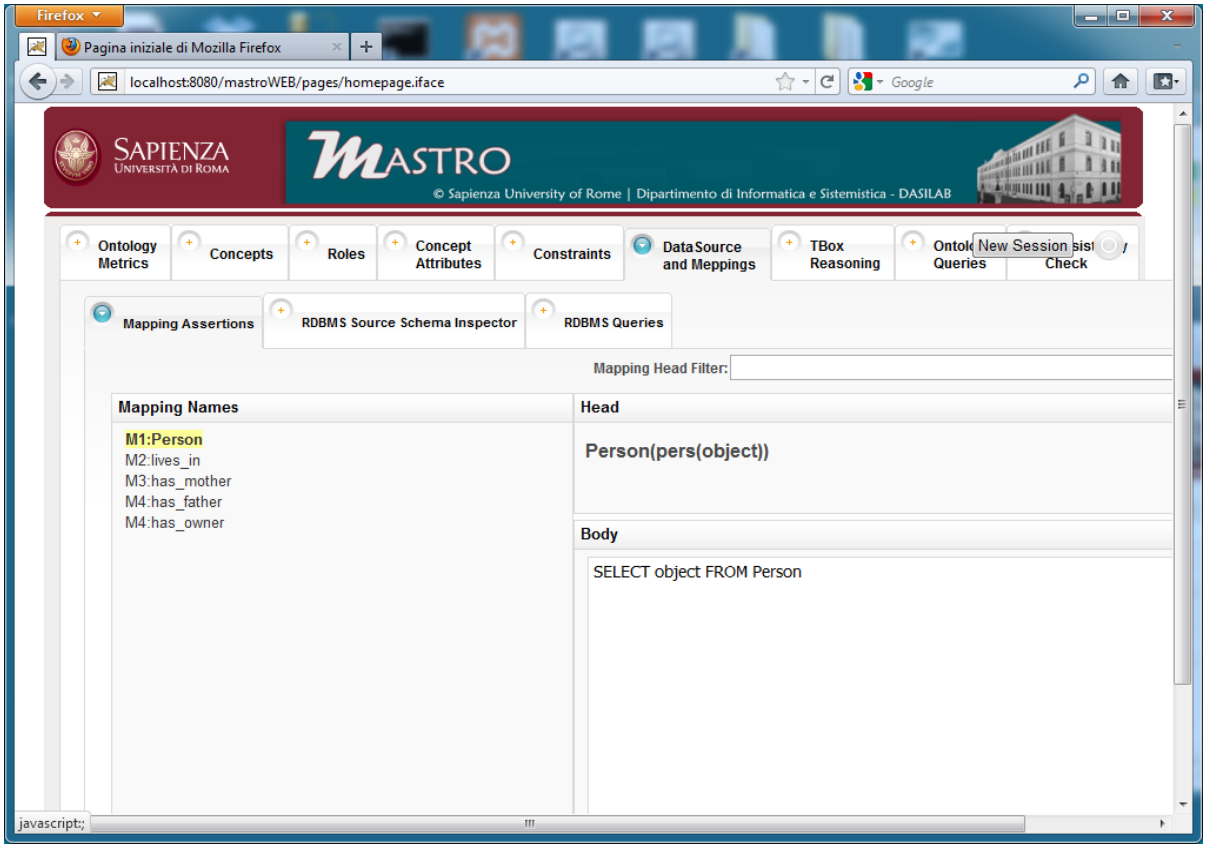
Concepts hierarchy



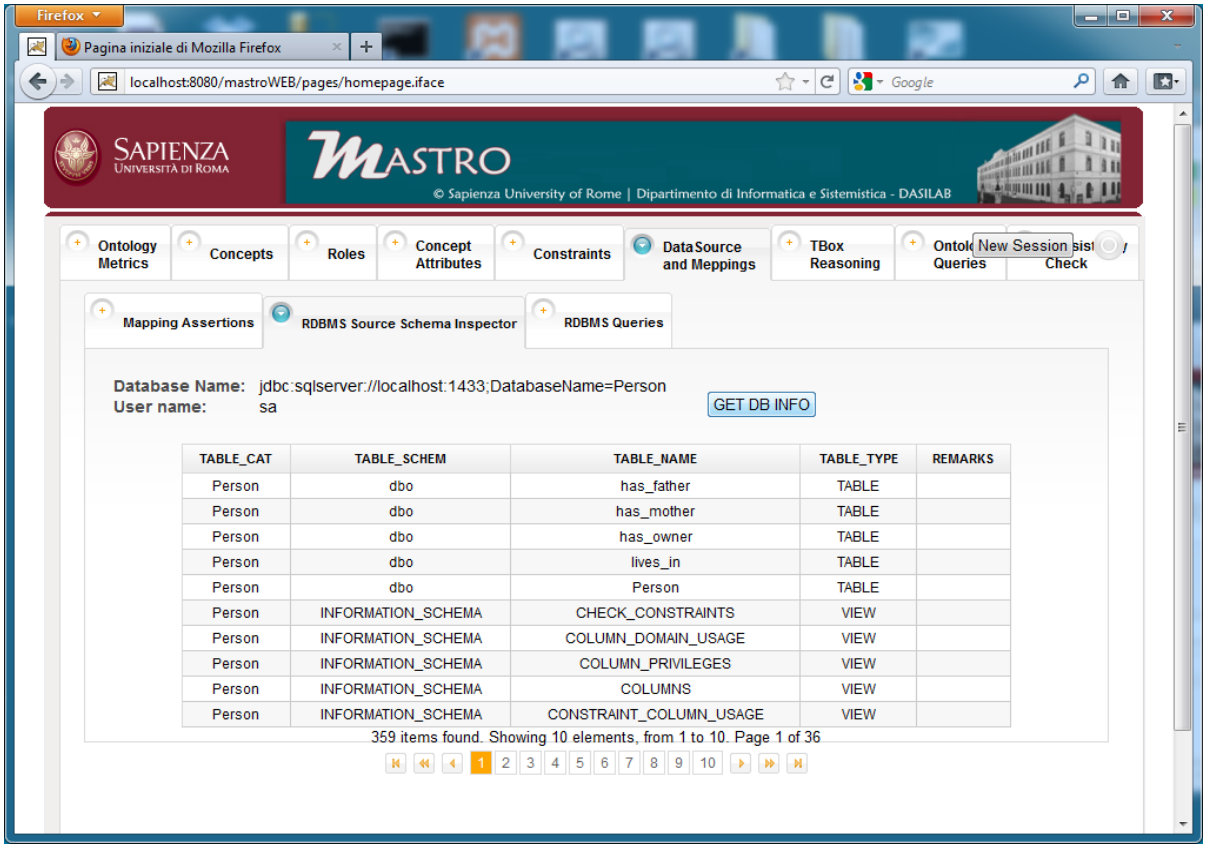
Roles hierarchy



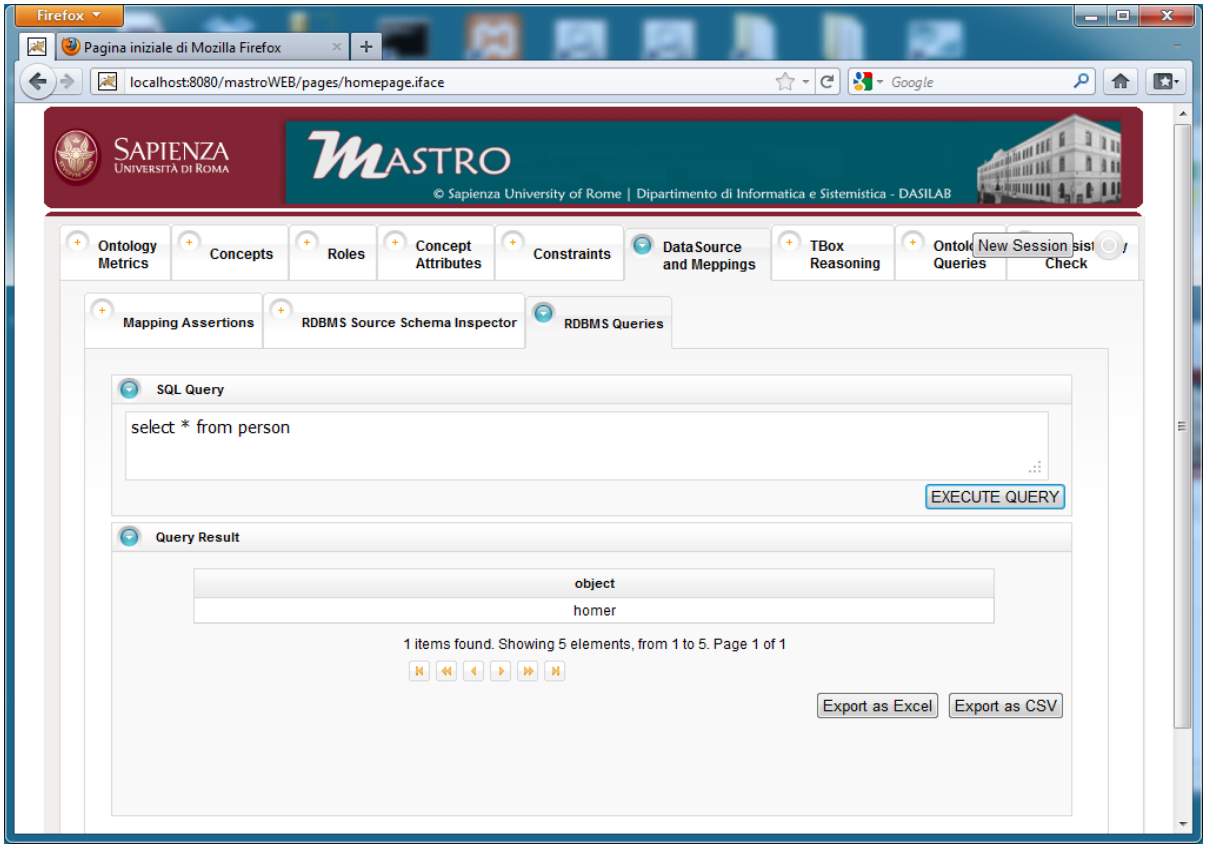
ABox specification through data mappings



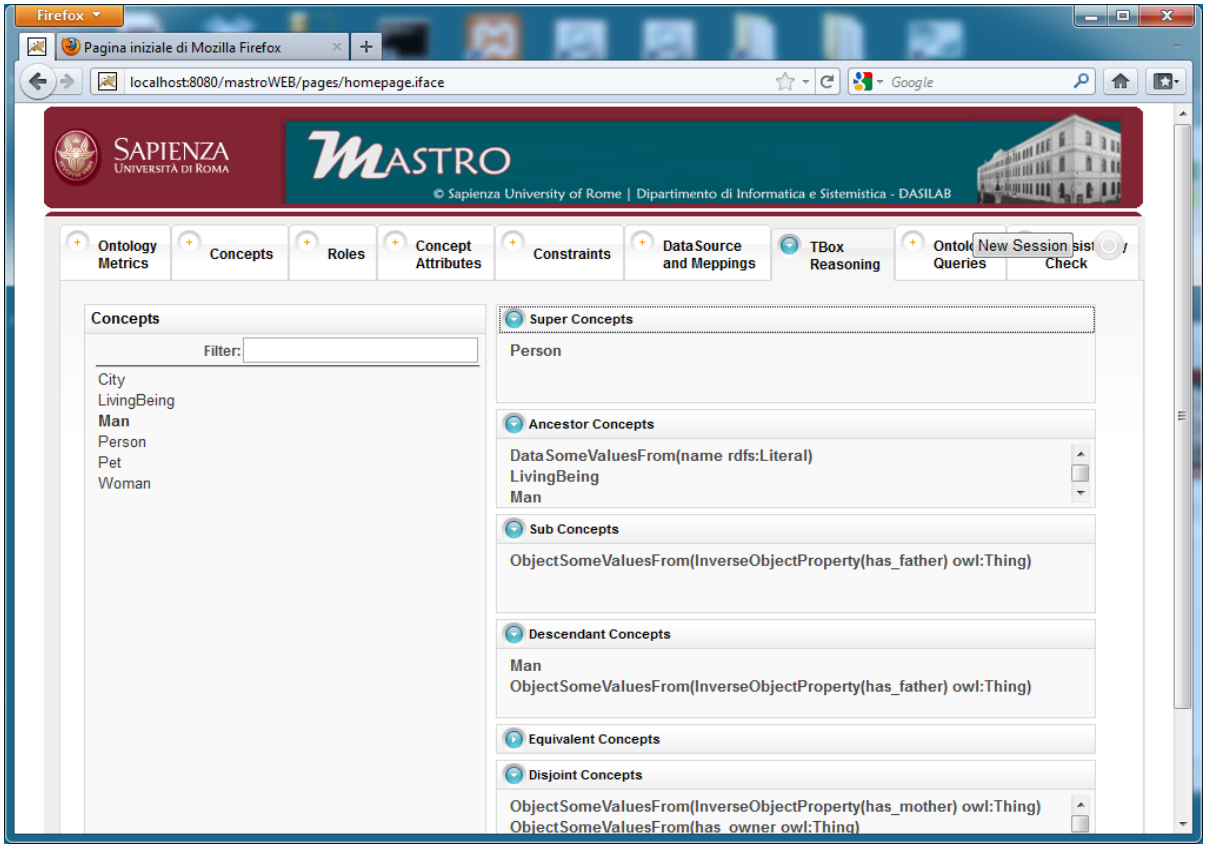
Data source schema inspector



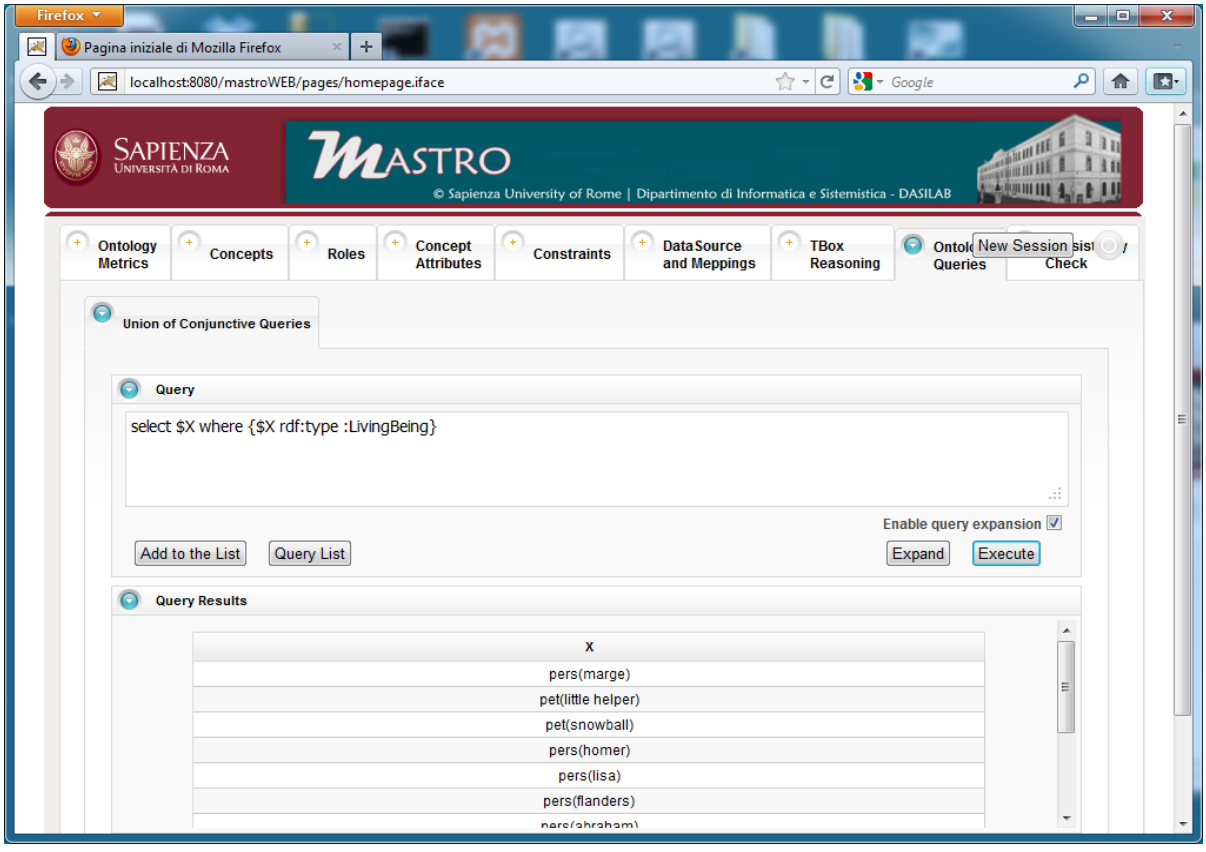
Direct data source queries



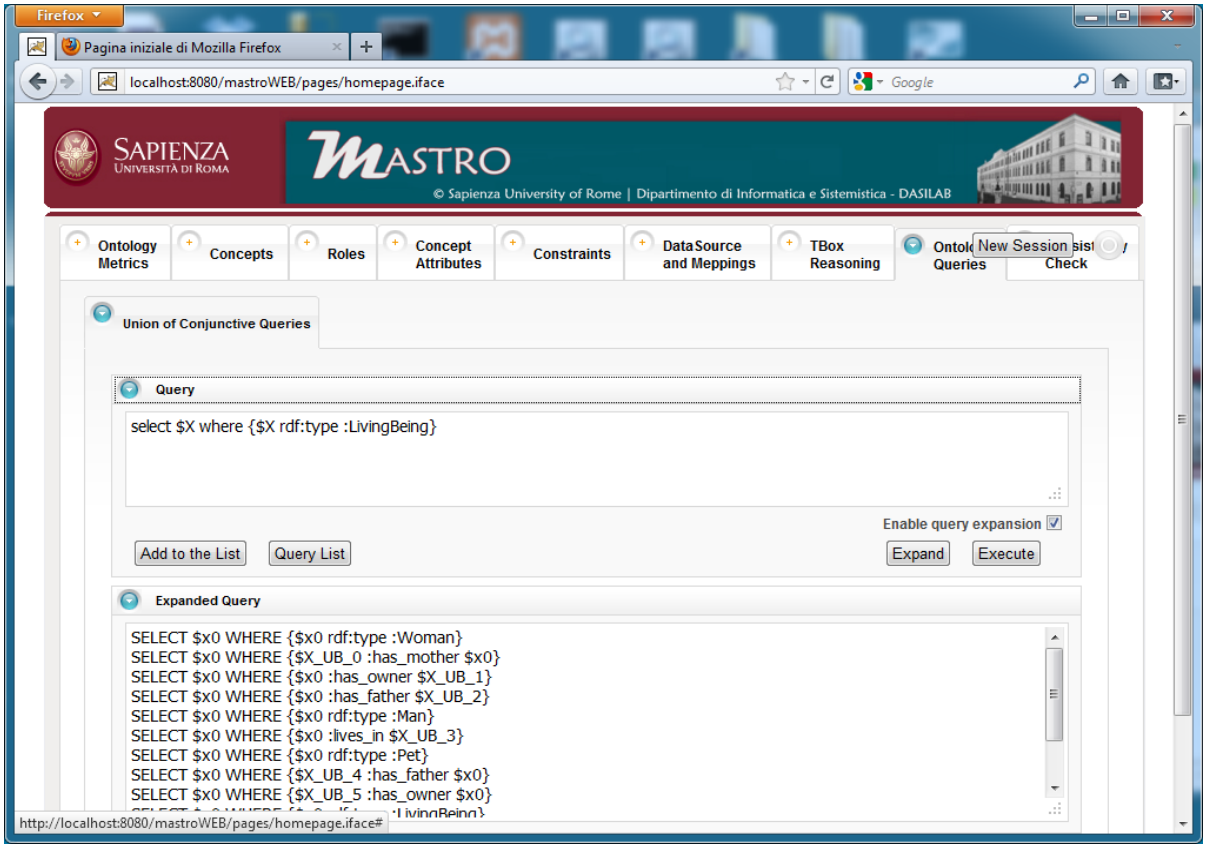
Tbox Reasoning



Ontology queries



Query reformulation 1: expansion



Query reformulation 2: unfolding