

Importare i dati in RDF caso di studio: Ontop

Panoramica delle principali funzionalità

Ontop

- Strumento di mapping da basi di dati relazionali a ontologie:
 - <https://ontop.inf.unibz.it>
- Disponibile anche come plugin per Protégé
- Tutorial: <https://github.com/ontop/ontop/wiki>
(Start here, Using Ontop)
- Mapping esportabili in formato R2RML

R2RML



RDB to RDF Mapping Language



Recommendation del W3C per esprimere mappings da database relazionali a datasets RDF



In pratica, i mapping definiti in R2RML permettono di accedere ai dati contenuti nello schema relazionale per mezzo del formato target, così da:

Interrogare i dati via SPARQL

Esportarli in formato RDF

R2RML: esempio (dal documento di specifiche)

@prefix rr: <http://www.w3.org/ns/r2rml#>.

@prefix ex: <http://example.com/ns#>.

<#TriplesMap1>

rr:logicalTable [rr:tableName "EMP"];

rr:subjectMap [

rr:template "http://data.example.com/employee/{EMPNO}";

rr:class ex:Employee;];

rr:predicateObjectMap [rr:predicate ex:name;

rr:objectMap [rr:column "ENAME"];].

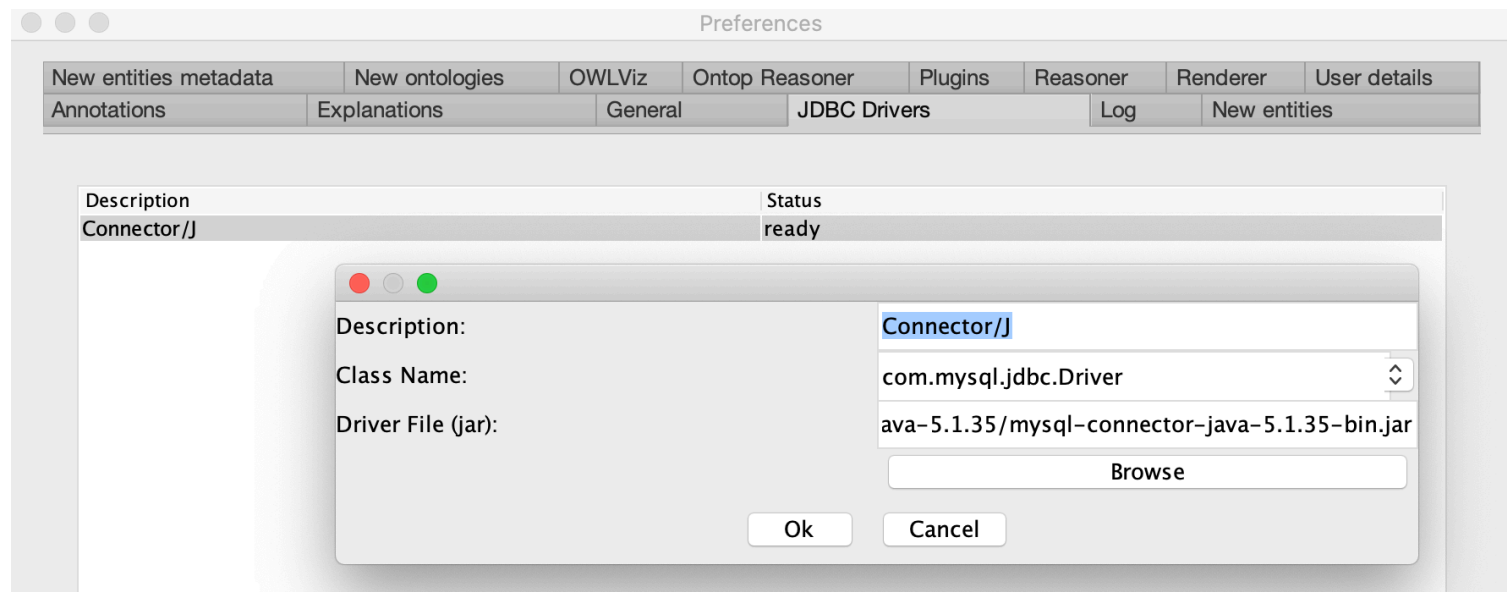
Preleva il dato dalla tabella EMP e crea le triple seguenti

<http://data.example.com/employee/7369> rdf:type ex:Employee.

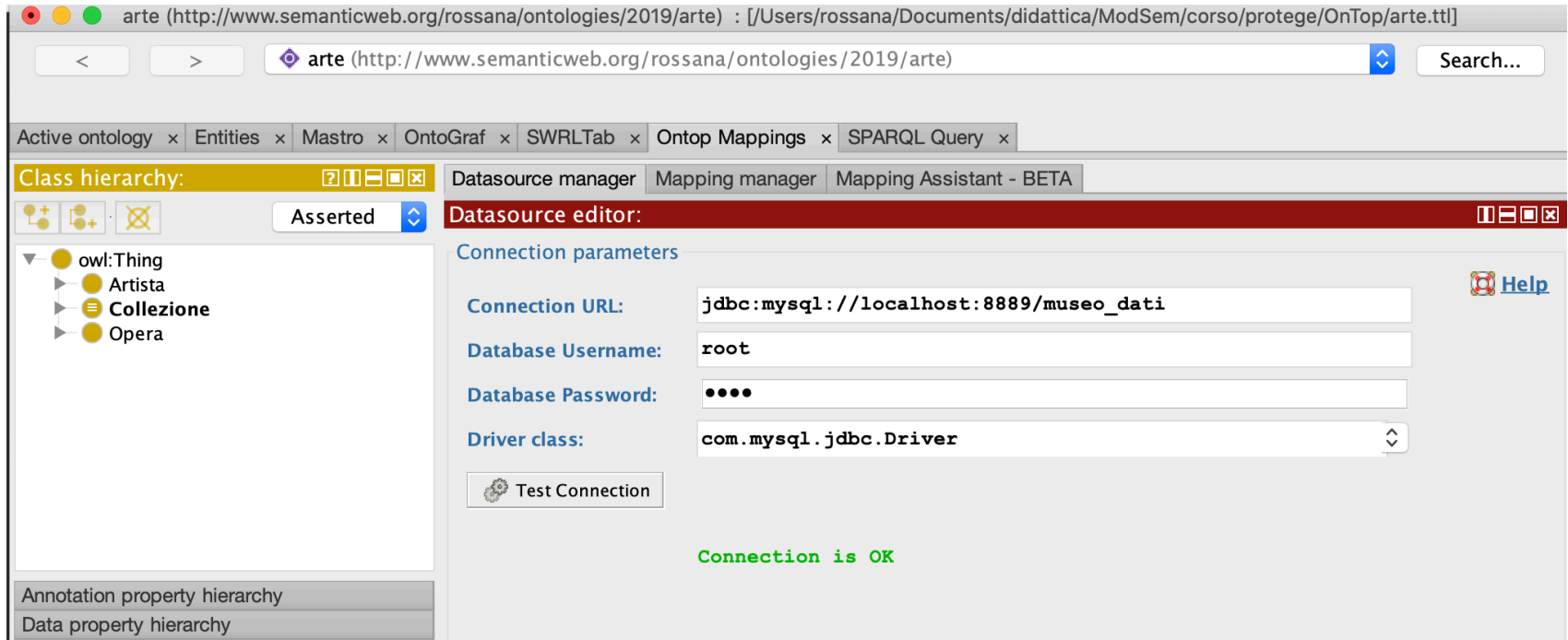
<http://data.example.com/employee/7369> ex:name "SMITH".

Ontop: configurazione

- Avviare il database server
- Configurare il plugin di Protégé
- In Preferenze | JDBC Drivers, settare il driver per il proprio database server



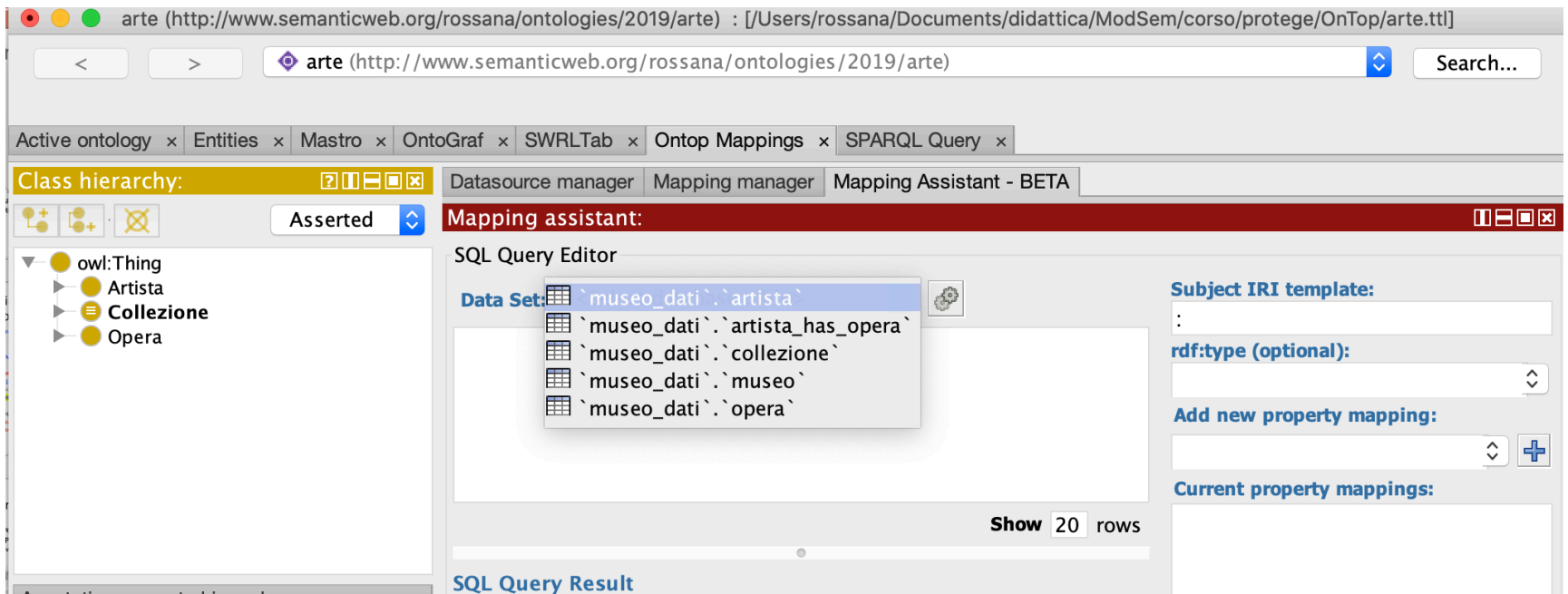
Ontop: collegamento al database



- URL dello schema di partenza (con nome schema): jdbc:mysql://<host>[:port]/[database]
- Credenziali per il database
- Driver

Il target è l'ontologia caricata in Protégé

Mapping Assistant: creazione dei mapping



Selezione tabella di partenza

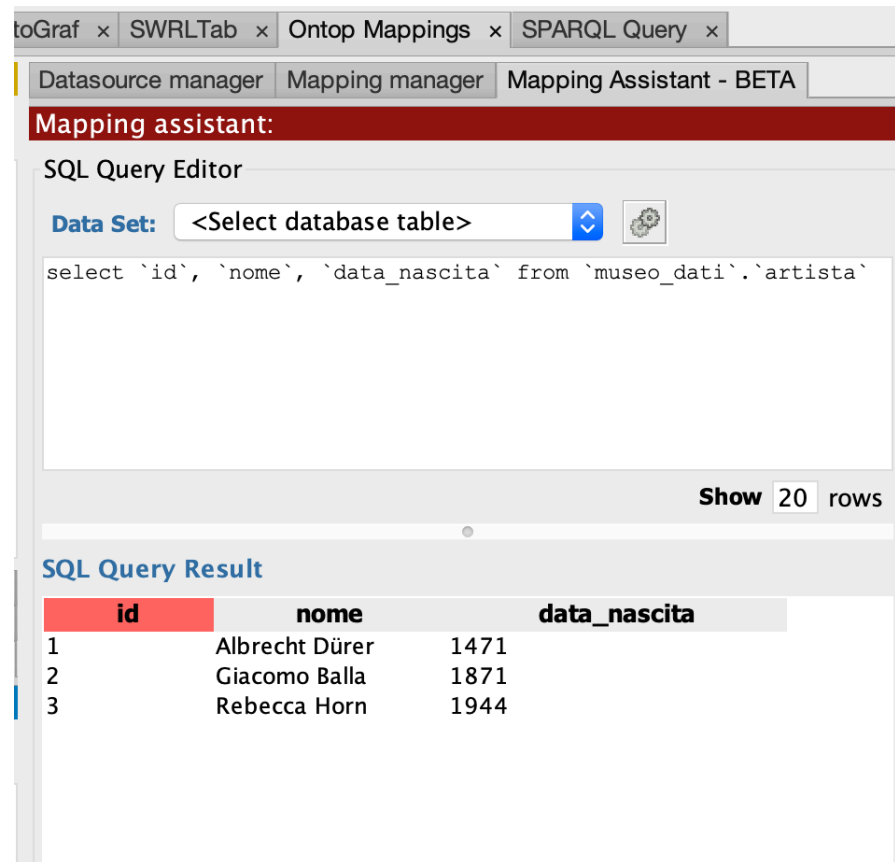
Selezione dei dati nello schema

- Selezionare la tabella
- Eseguire la query
- Verificare i risultati visualizzati



A screenshot of a database schema viewer showing the structure of the 'artista' table. The table has three columns: 'id' of type INT, 'nome' of type VARCHAR(45), and 'data_nascita' of type INT. There is an 'Indexes' section at the bottom with a right-pointing arrow.

artista	
id	INT
nome	VARCHAR(45)
data_nascita	INT
Indexes ▶	



The screenshot shows the 'Mapping Assistant - BETA' window. It has tabs for 'Datasource manager', 'Mapping manager', and 'Mapping Assistant - BETA'. The 'SQL Query Editor' section contains a 'Data Set' dropdown set to '<Select database table>' and a text area with the SQL query: `select `id`, `nome`, `data_nascita` from `museo_dati`.`artista``. Below the editor, it says 'Show 20 rows'. The 'SQL Query Result' section displays a table with three columns: 'id', 'nome', and 'data_nascita'. The results show three rows of data.

id	nome	data_nascita
1	Albrecht Dürer	1471
2	Giacomo Balla	1871
3	Rebecca Horn	1944

Mapping Assistant: creazione template

The screenshot shows the 'Mapping Assistant - BETA' window. The 'SQL Query Editor' has a 'Data Set' dropdown set to '<Select database table>' and a text area containing the query: `select `id`, `nome`, `data_nascita` from `museo_dati`.`artista``. Below the editor, the 'SQL Query Result' displays a table with 3 rows and 3 columns: **id**, **nome**, and **data_nascita**. The results are: 1 Albrecht Dürer 1471, 2 Giacomo Balla 1871, and 3 Rebecca Horn 1944. On the right, the 'Subject IRI template' is ':artista{id}'. The 'rdf:type (optional)' is 'Artista'. Under 'Add new property mapping:', 'dataNascita' is added. The 'Current property mappings:' section shows 'nome' mapped to '<Undefined...>' and 'dataNascita' mapped to 'data_nascita' (highlighted with a red box).

Datasource manager Mapping manager Mapping Assistant - BETA

Mapping assistant:

SQL Query Editor

Data Set: <Select database table>

```
select `id`, `nome`, `data_nascita` from `museo_dati`.`artista`
```

Show 20 rows

SQL Query Result

id	nome	data_nascita
1	Albrecht Dürer	1471
2	Giacomo Balla	1871
3	Rebecca Horn	1944

Subject IRI template:
:artista{id}

rdf:type (optional):
Artista

Add new property mapping:
dataNascita

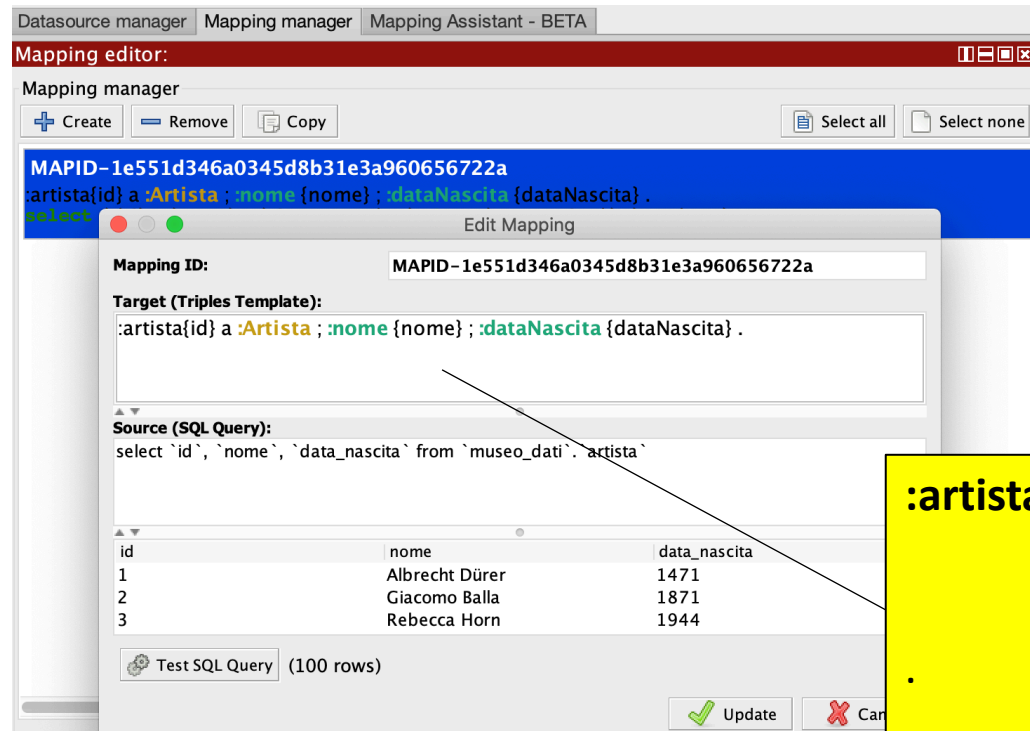
Current property mappings:

- nome <Undefined...>
- dataNascita <Undefined...>
data_nascita

Si inseriscono nello schema

- Il template per il soggetto
- La classe del soggetto
- I mapping delle colonne sulle singole proprietà

Verifica del template



Nel mapping manager è possibile visualizzare e modificare il template creato in formato Turtle

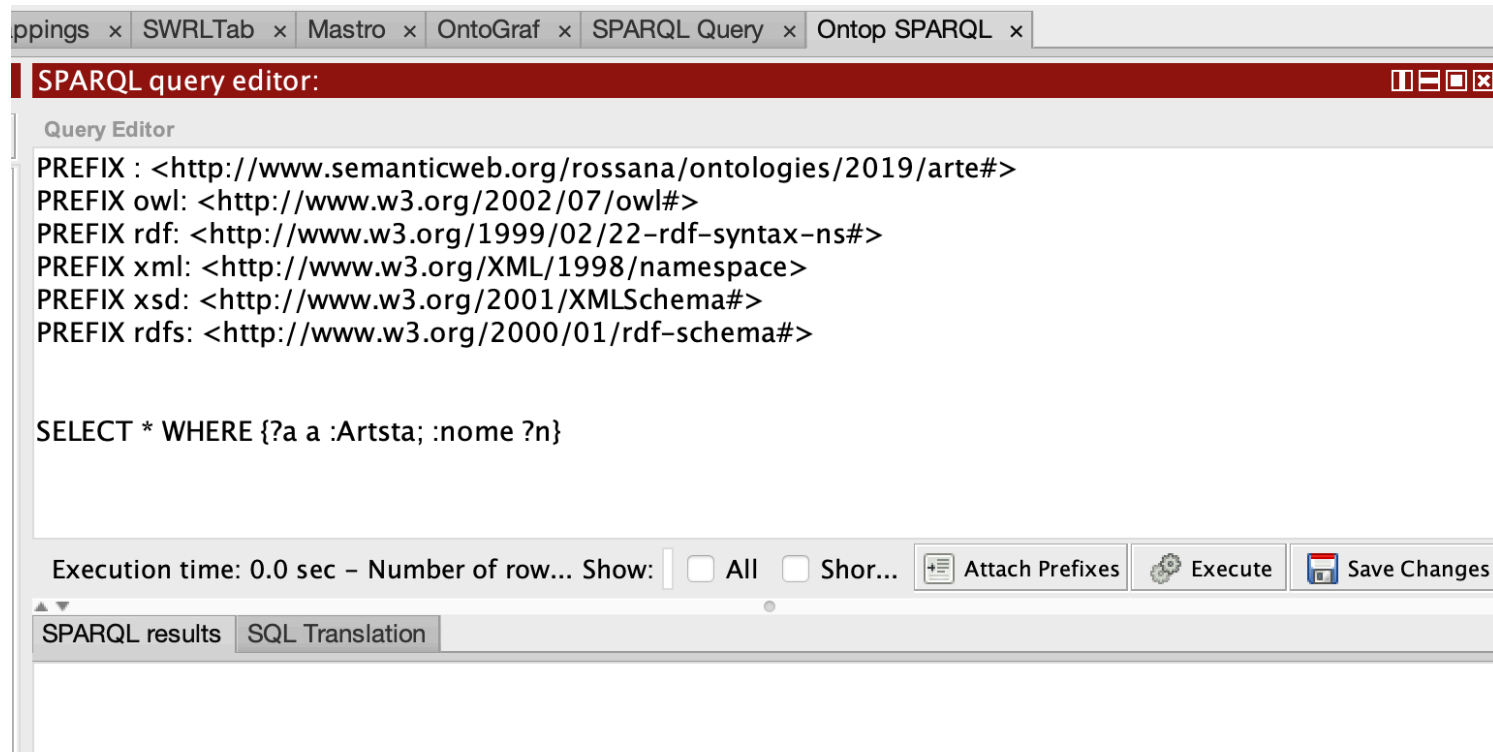
```
:artista{id} a :Artista ;  
:nome {nome} ;  
:dataNascita {dataNascita} .
```

Esportazione mapping

- Ontop | Export R2RML mappings
- Risultato: .ttl contenente i mapping

```
<urn:MAPID-1e551d346a0345d8b31e3a960656722a_1>
  a rr:TriplesMap ;
  rr:logicalTable [ a rr:R2RMLView ;
                    rr:sqlQuery "select `id`, `nome`, `data_nascita` from `museo_dati`.`artista`"
                    ] ;
  rr:predicateObjectMap [ a rr:PredicateObjectMap ;
                          rr:objectMap [ a rr:TermMap , rr:ObjectMap ;
                                          rr:column "nome" ;
                                          rr:termType rr:Literal
                                          ] ;
                          rr:predicate :nome
                          ] ;
  rr:predicateObjectMap [ a rr:PredicateObjectMap ;
                          rr:objectMap [ a rr:TermMap , rr:ObjectMap ;
                                          rr:column "dataNascita" ;
                                          rr:termType rr:Literal
                                          ] ;
                          rr:predicate :dataNascita
                          ] ;
  rr:subjectMap [ a rr:SubjectMap , rr:TermMap ;
                 rr:class :Artista ;
                 rr:template "http://www.semanticweb.org/rossana/ontologies/2019/arte#artista{id}" ;
                 rr:termType rr:IRI
                 ] .
```

Accesso al database via SPARQL



- Lanciare il reasoner Ontop
- Premere Execute

Non dimenticare i prefissi!

Accesso al database via SPARQL

The screenshot shows the Ontop SPARQL query editor interface. The title bar includes tabs for 'Ontop SPARQL', 'Ontop Mappings', 'SWRLTab', 'OntoGraf', and 'SPARQL Query'. The main window is titled 'SPARQL query editor:' and contains a 'Query Editor' section with the following prefixes and query:

```
PREFIX : <http://www.semanticweb.org/rossana/ontologies/2019/arte#>
PREFIX g: <http://www.semanticweb.org/rossana/ontologies/2019/arte/mappings>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX xml: <http://www.w3.org/XML/1998/namespace>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX obda: <https://w3id.org/obda/vocabulary#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>

SELECT * WHERE {?a a :Artista; :nome ?n}
```

Below the query editor, the execution status is shown: 'Execution time: 0.147 sec - Number of r...'. There are buttons for 'Attach Prefixes', 'Execute', and 'Save Changes'. The 'Execute' button is highlighted.

The results are displayed in a table with two columns: 'a' and 'n'. The results are:

a	n
<http://www.semanticweb.org/rossana/ontologies/2019/arte...>	"Albrecht Dürer"^^xsd:string
<http://www.semanticweb.org/rossana/ontologies/2019/arte...>	"Giacomo Balla"^^xsd:string
<http://www.semanticweb.org/rossana/ontologies/2019/arte...>	"Rebecca Horn"^^xsd:string

Accesso al database via SPARQL

Ontop SPARQL x Ontop Mappings x SWRLTab x OntoGraf x SPARQL Query x

SPARQL query editor:

Query Editor

```
PREFIX : <http://www.semanticweb.org/rossana/ontologies/2019/arte#>
PREFIX g: <http://www.semanticweb.org/rossana/ontologies/2019/arte/mappings>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX xml: <http://www.w3.org/XML/1998/namespace>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX obda: <https://w3id.org/obda/vocabulary#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
```

```
SELECT * WHERE {?a a :Artista; :dataNascita ?n}
```

Execution time: 0.061 sec – Number of r... Show: ☐ All ☐ Shor... Attach Prefixes Execute Save Changes

SPARQL results SQL Translation

a	n
<http://www.semanticweb.org/rossana/ontologies/2019/arte#artista1>	"1471"^^xsd:integer
<http://www.semanticweb.org/rossana/ontologies/2019/arte#artista2>	"1871"^^xsd:integer
<http://www.semanticweb.org/rossana/ontologies/2019/arte#artista3>	"1944"^^xsd:integer

Materializzazione triple

Ontop | Materialize triples ...

Scegliere il formato

<http:// www.semanticweb.org/arte#artista1> a

<http:// www.semanticweb.org/arte #Artista> .

<http://www.semanticweb.org/arte#artista1>

<http:// www.semanticweb.org/arte#dataNascita>

"1471"^^<http://www.w3.org/2001/XMLSchema#integer> .

<http:// www.semanticweb.org/arte#artista1>

<http://www. www.semanticweb.org/arte#nome>

"Albrecht Dürer" .

Mapping di relazioni su object properties

SELECT Q1.nome, Q1.artid AS aid, opera.titolo AS tit

FROM

(SELECT artista.nome AS nome,
artista_has_opera.artista_id AS artid,
artista_has_opera.opera_id AS opid

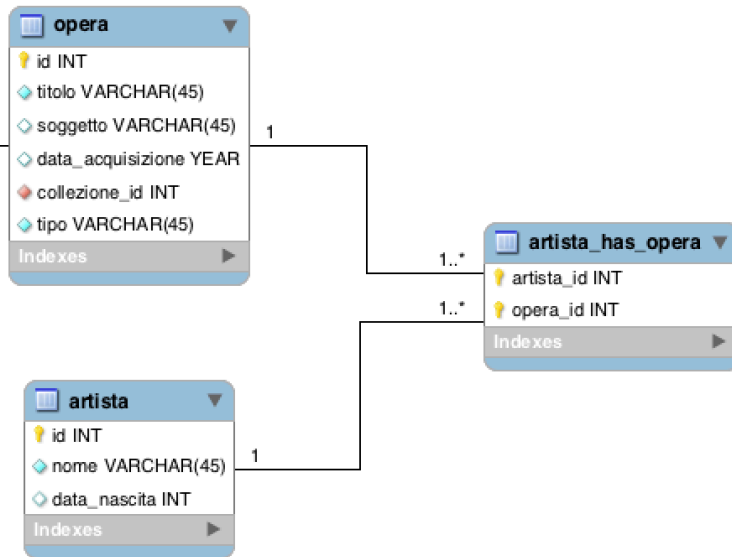
FROM artista JOIN artista_has_opera

ON artista.id = artista_has_opera.artista_id) as Q1

JOIN opera ON opid = opera.id

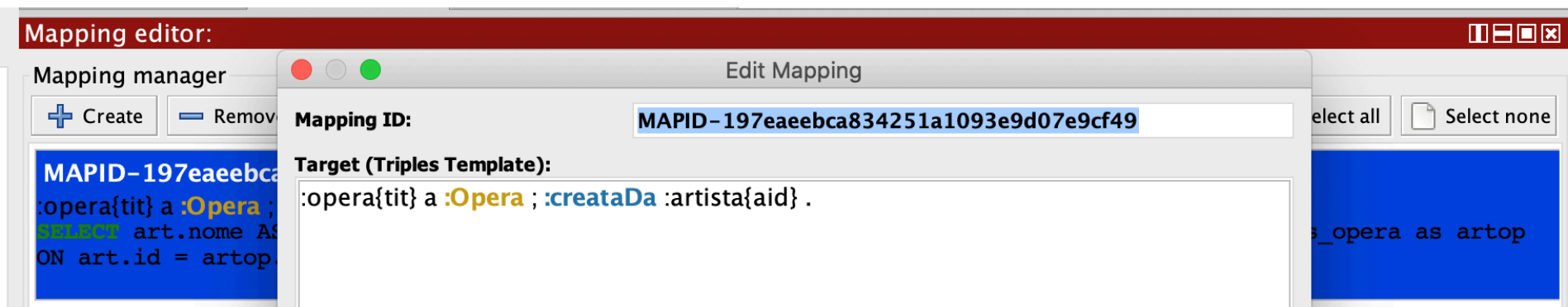


	nome	aid	tit
►	Albrecht Dürer	1	Autoritratto
	Giacomo Balla	2	Feu d'artifice
	Rebecca Horn	3	Miroir du Lac
	Rebecca Horn	3	Cutting through the past



*Ontop non supporta notazione qualified (usare AS)
Le subqueries devono essere rinominate*

Mapping verso RDF



	nome	aid	tit
▶	Albrecht Dürer	1	Autoritratto
	Giacomo Balla	2	Feu d'artifice
	Rebecca Horn	3	Miroir du Lac
	Rebecca Horn	3	Cutting through the past



Description: creataDa

- Equivalent To +
- SubProperty Of +
- Inverse Of +
crea
- Domains (intersection) +
Opera
- Ranges (intersection) +
Artista

il nome dell'autore è mappato sull'id dell'autore in un template separato

Chiavi esterne nel mapping

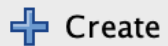
- Si mappa la relazione tra opere e autori sulla object property *creataDa*
- Il mapping delle proprietà dell'autore (data di nascita e nome) è gestito mediante un template separato
- Utilizzando per l'oggetto della tripla (autore) lo stesso mapping utilizzato per mappare nome e data di nascita degli autori sull'id dell'autore si mantiene il collegamento tra autori e opere

Mapping multipli

Mapping editor:



Mapping manager



Create

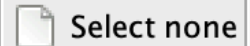
Remove



Copy



Select all



Select none

MAPID-18162ecf3081452ca25cee9724d14cff

:opera{tit} a :Opera ; :creataDa :artista{aid} .

```
SELECT Q1.nome, Q1.artid AS aid, opera.titolo AS tit
FROM (SELECT artista.nome AS nome, artista_has_opera.artista_id
AS artid, artista_has_opera.opera_id AS opid FROM artista JOIN
artista_has_opera ON artista.id = artista_has_opera.artista_id) as Q1
JOIN opera ON opid = opera.id
```

MAPID-b92724d573e647cabbf2912575a2752f

:artista{id} a :Artista ; :nome {nome} .

```
select `id`, `nome`, `data_nascita` from `museo_dati`.`artista`
```

Query SPARQL su mapping multipli

SPARQL query editor: ⏏ ⏏ ⏏ ⏏

Query Editor

```
PREFIX : <http://www.semanticweb.org/rossana/ontologies/2019/arte#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX xml: <http://www.w3.org/XML/1998/namespace>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX obda: <https://w3id.org/obda/vocabulary#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>

SELECT ?o ?a ?n
WHERE {
    ?o a :Opera ; :creataDa ?a .
    ?a :nome ?n .
}
```

Execution time: 0.198 sec – Number of rows retrieved: 4

Show: ☐ All ☐ Short IRI

SPARQL results SQL Translation

o	a	n
<http://www.semanticweb.org/rossana/ontologies/2019/arte#operaAutoritratto>	<http://www.semanticweb.org/rossana/ontologi...	"Albrecht Dürer"^^xsd:string
<http://www.semanticweb.org/rossana/ontologies/2019/arte#operaFeu%20d%27artifice>	<http://www.semanticweb.org/rossana/ontologi...	"Giacomo Balla"^^xsd:string
<http://www.semanticweb.org/rossana/ontologies/2019/arte#operaMiroir%20du%20Lac>	<http://www.semanticweb.org/rossana/ontologi...	"Rebecca Horn"^^xsd:string
<http://www.semanticweb.org/rossana/ontologies/2019/arte#operaCutting%20through%20the%20past>	<http://www.semanticweb.org/rossana/ontologi...	"Rebecca Horn"^^xsd:string

- Mantenendo consistente l'IRI della stessa entità nei diversi template si mantengono le chiavi esterne del database
- È creare queries SPARQL che coinvolgono più mapping contemporaneamente (artista e opera, attributi di artista (qui: nome))

Triple materializzate (con inferenze)

<http://www.semanticweb.org/arte#artista1> a
<http://www.semanticweb.org/arte#Artista> .

<http://www.semanticweb.org/arte#artista1>
<http://www.semanticweb.org/arte#nome> "Albrecht Dürer" .

<http://www.semanticweb.org/arte#operaAutoritratto> a
<http://www.semanticweb.org/arte#Opera> .

<http://www.semanticweb.org/arte#operaAutoritratto>
<http://www.semanticweb.org/arte#creataDa>
<http://www.semanticweb.org/arte#artista1>

<http://www.semanticweb.org/arte#artista1>
<http://www.semanticweb.org/arte#crea>
<http://www.semanticweb.org/arte#operaAutoritratto> .

Ontop reasoner e inferenze

- Il reasoner è in grado di fare inferenze su RDF
- Dato che creatoDa(opera, artista) ha come proprietà inversa crea(artista, opera), il reasoner inferisce che la vale la proprietà inversa per ogni coppia artista-opera
- Si ottiene lo stesso binding effettuando la query sulla proprietà inferita:

```
SELECT ?o ?a ?n
```

```
WHERE {
```

```
    ?o a :Opera.
```

```
    ?a :crea ?o .
```

```
    ?a :nome ?n .
```

```
}
```