Ontology-Based Data Access with Ontop

Benjamin Cogrel
benjamin.cogrel@unibz.it

KRDB Research Centre for Knowledge and Data
Free University of Bozen-Bolzano, Italy

Freie Universität Bozen
Libera Università di Bolzano
Free University of Bozen-Bolzano

JDEV,
Marseille, 06 July 2017

Ontology-Based Data Access (OBDA)

Outline

1. SQL queries over tables can be hard to write manually
2. RDF and other Semantic Web standards
3. Ontology-Based Data Access
4. Optique platform
5. Conclusion

unibz

Toy example: University Information System

Relational source

uni1.student

s_id	first_name	last_name
1	Mary	Smith
2	John	Doe

uni1.academic

a_id	first_name	last_name	position
1	Anna	Chambers	1
2	Edward	May	9
3	Rachel	Ward	8

uni1.course

c_id	title
1234	Linear algebra

uni1.teaching

c_id	a_id
1234	1
1234	2

Information need	SQL query
1. First and last names of the students	`SELECT DISTINCT "first_name", "last_name"` `FROM "uni1"."student"`

2. First and last names of the persons	`SELECT DISTINCT "first_name", "last_name"`
	`FROM "uni1"."student"`
	`UNION`
	`SELECT DISTINCT "first_name", "last_name"`
	`FROM "uni1"."academic"`

3. Course titles and teacher names	`SELECT DISTINCT co."title", ac."last_name"`
	`FROM "uni1"."course" co,`
	`     "uni1"."academic" ac,`
	`     "uni1"."teaching" teach`
	`WHERE co."c_id" = teach."c_id"`
	`   AND ac."a_id" = teach."a_id"`

4. All the teachers	`SELECT DISTINCT "a_id"`
	`FROM "uni1"."teaching" UNION`
	`SELECT DISTINCT "a_id"`
	`FROM "uni1"."academic"`
	`WHERE "position" BETWEEN 1 AND 8`

Integration of a second source

Fusion of two universities

`uni2.person`

pid	fname	lname	status
1	Zak	Lane	8
2	Mattie	Moses	1
3	Céline	Mendez	2

`uni2.course`

cid	lecturer	lab_teacher	topic
1	1	3	Information security

Translation of information needs I

<table>
<tr><th>Information need</th><th>SQL query</th></tr>
<tr><td>1. First and last names of the students</td><td><pre>SELECT DISTINCT "first_name", "last_name"
FROM "uni1"."student"
UNION
SELECT DISTINCT "fname" AS "first_name",
                "lname" AS "last_name"
FROM "uni2"."person"
WHERE "status" BETWEEN 1 and 2</pre></td></tr>
<tr><td>2. First and last names of the persons</td><td><pre>SELECT DISTINCT "first_name", "last_name"
FROM "uni1"."student"
UNION
SELECT DISTINCT "first_name", "last_name"
FROM "uni1"."academic"
UNION
SELECT DISTINCT "fname" AS "first_name",
                "lname" AS "last_name"
FROM "uni2"."person"</pre></td></tr>
</table>

Translation of information needs II

Information need	SQL query
3. Course titles and teacher names	`SELECT DISTINCT co."title", ac."last_name" FROM "uni1"."course" co, "uni1"."academic" ac, "uni1"."teaching" teach WHERE co."c_id" = teach."c_id" AND ac."a_id" = teach."a_id" UNION SELECT DISTINCT co."topic" AS "title", pe."lname" AS "last_name" FROM "uni2"."person" pe, "uni2"."course" co WHERE pe."pid" = co."lecturer" OR pe."pid" = co."lab_teacher"`

```sql
SELECT DISTINCT co."title", ac."last_name"
FROM "uni1"."course" co,
     "uni1"."academic" ac,
     "uni1"."teaching" teach
WHERE co."c_id" = teach."c_id"
      AND ac."a_id" = teach."a_id"
UNION
SELECT DISTINCT co."topic" AS "title",
                pe."lname" AS "last_name"
FROM "uni2"."person" pe,
     "uni2"."course" co
WHERE pe."pid" = co."lecturer"
      OR pe."pid" = co."lab_teacher"
```

Translation of information needs III

Information need	SQL query

4. All the teachers	`SELECT DISTINCT 'uni1/' || "a_id" AS "id"`
	`FROM "uni1"."teaching"`
	`UNION`
	`SELECT DISTINCT 'uni1/' || "a_id" AS "id"`
	`FROM "uni1"."academic"`
	`WHERE "position" BETWEEN 1 AND 8`
	`UNION`
	`SELECT DISTINCT 'uni2/' || "lecturer" AS "id"`
	`FROM "uni2"."course"`
	`UNION`
	`SELECT DISTINCT 'uni2/' || "lab_teacher" AS "id"`
	`FROM "uni2"."course"`
	`UNION`
	`SELECT DISTINCT 'uni2/' || "pid" AS "id"`
	`FROM "uni2"."person"`
	`WHERE "status" BETWEEN 6 AND 9`

Industrial case: stratigraphic model design

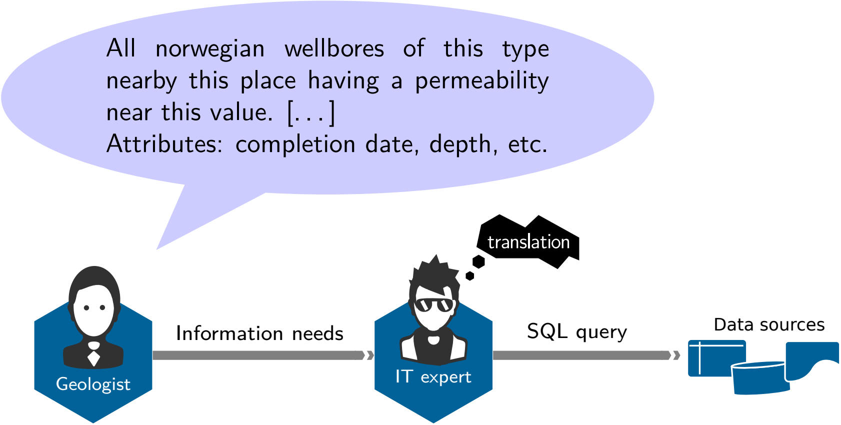
Users: domain experts

- $\sim$ 900 geologists et geophysicists
- Data collecting: 30-70% of their time

Sources

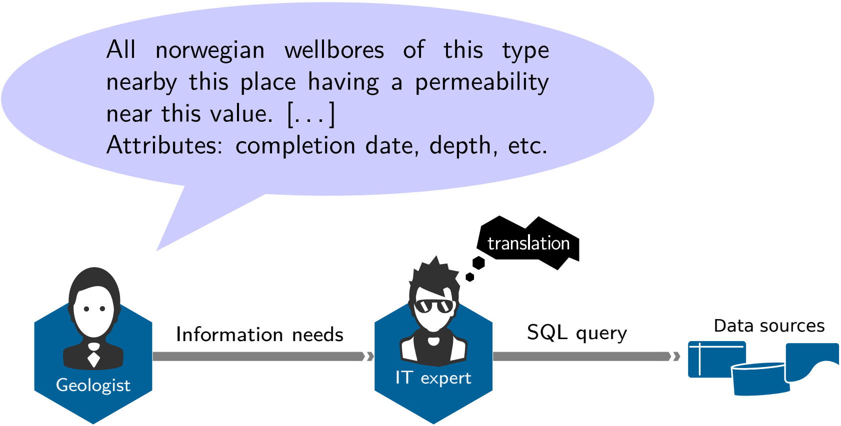
- *Exploitation and Production Data Store*: $\sim$ 1500 tables (100s GB)
- *Norwegian Petroleum Directorate FactPages*
- *OpenWorks*

Designing a new (ad-hoc) query



NB: Simplified information needs

Designing a new (ad-hoc) query



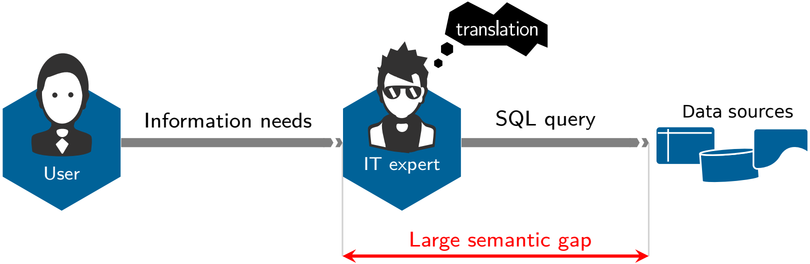
Takes 4 days in average (with EPDS only)

NB: Simplified information needs

Anonymized extract of a typical query

```
SELECT [...]
FROM
db_name.table1 table1,
db_name.table2 table2a,
db_name.table2 table2b,
db_name.table3 table3a,
db_name.table3 table3b,
db_name.table3 table3c,
db_name.table3 table3d,
db_name.table4 table4a,
db_name.table4 table4b,
db_name.table4 table4c,
db_name.table4 table4d,
db_name.table4 table4e,
db_name.table4 table4f,
db_name.table5 table5a,
db_name.table5 table5b,
db_name.table6 table6a,
db_name.table6 table6b,
db_name.table7 table7a,
db_name.table7 table7b,
db_name.table8 table8,
db_name.table9 table9,
db_name.table10 table10a,
db_name.table10 table10b,
db_name.table10 table10c,
db_name.table11 table11,
db_name.table12 table12,
db_name.table13 table13,
db_name.table14 table14,
db_name.table15 table15,
db_name.table16 table16
WHERE [...]
table2a.attr1='keyword' AND
table3a.attr2=table10c.attr1 AND
table3a.attr6=table6a.attr3 AND
table3a.attr9='keyword' AND
table4a.attr10 IN ('keyword') AND
table4a.attr1 IN ('keyword') AND
table5a.kinds=table4a.attr13 AND
table5b.kinds=table4c.attr74 AND
table5b.name='keyword' AND
(table6a.attr19=table10c.attr17 OR
(table6a.attr2 IS NULL AND
table10c.attr4 IS NULL)) AND
table6a.attr14=table5b.attr14 AND
table6a.attr2='keyword' AND
(table6b.attr14=table10c.attr8 OR
(table6b.attr4 IS NULL AND
table10c.attr7 IS NULL)) AND
table6b.attr19=table5a.attr55 AND
table6b.attr2='keyword' AND
table7a.attr19=table2b.attr19 AND
table7a.attr17=table15.attr19 AND
table4b.attr11='keyword' AND
table8.attr19=table7a.attr80 AND
table8.attr19=table13.attr20 AND
table8.attr4='keyword' AND
table9.attr10=table16.attr11 AND
table3b.attr19=table10c.attr18 AND
table3b.attr22=table12.attr63 AND
table3b.attr66='keyword' AND
table10a.attr54=table7a.attr8 AND
table10a.attr70=table10c.attr10 AND
table10a.attr16=table4d.attr11 AND
table4c.attr99='keyword' AND
table4c.attr1='keyword' AND
table11.attr10=table5a.attr10 AND
table11.attr40='keyword' AND
table11.attr50='keyword' AND
table2b.attr1=table1.attr8 AND
table2b.attr9 IN ('keyword') AND
table2b.attr2 LIKE 'keyword'% AND
table12.attr9 IN ('keyword') AND
table7b.attr1=table2a.attr10 AND
table3c.attr13=table10c.attr1 AND
table3c.attr10=table6b.attr20 AND
table3c.attr13='keyword' AND
table10b.attr16=table10a.attr7 AND
table10b.attr11=table7b.attr8 AND
table10b.attr13=table4b.attr89 AND
table13.attr1=table2b.attr10 AND
table13.attr20=''keyword'' AND
table13.attr15='keyword' AND
table3d.attr49=table12.attr18 AND
table3d.attr18=table10c.attr11 AND
table3d.attr14='keyword' AND
table4d.attr17 IN ('keyword') AND
table4d.attr19 IN ('keyword') AND
table16.attr28=table11.attr56 AND
table16.attr16=table10b.attr78 AND
table16.attr5=table14.attr56 AND
table4e.attr34 IN ('keyword') AND
table4e.attr48 IN ('keyword') AND
table4f.attr89=table5b.attr7 AND
table4f.attr45 IN ('keyword') AND
table4f.attr1='keyword' AND
table10c.attr2=table4e.attr19 AND
(table10c.attr78=table12.attr56 OR
(table10c.attr55 IS NULL AND
table12.attr17 IS NULL))
```

Semantic gap



Querying over tables

Requires a lot of knowledge about:

1. Magic numbers
   (e.g. 1 → full professor)
2. Cardinalities and normal forms
3. Spreading of closely-related information across many tables

Data integration

- Exacerbates these issues
- Variety: **challenge #1** for most Big Data initiatives

unibz

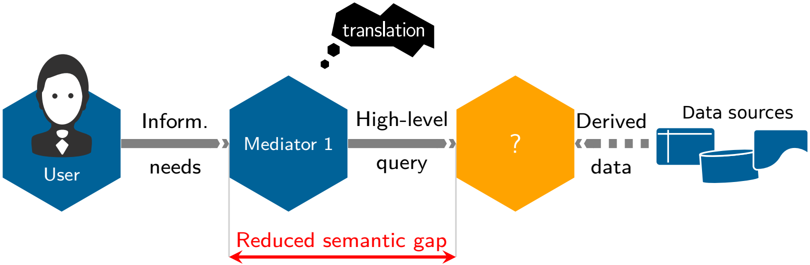
High-level translation

Main bottleneck: translation

- of the information needs
- . . . into a **formal query**

Goal

Make such a translation easy
(Ideally: IT expertise not required)



Mediator 1 could be a user, an IT expert or a GUI

General approach: two steps

1. Translate the information needs into a **high-level query**
2. Answer the high-level query **automatically**

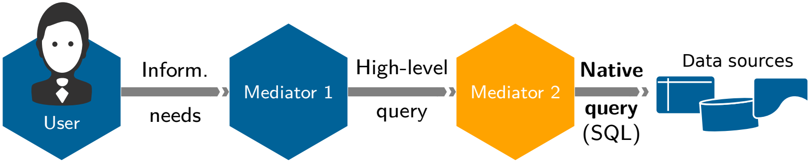
Choice 1: How to derive data from the data sources

Extract Transform Load (ETL) process

E.g. relational data warehouse, triplestore

Virtual views

E.g. virtual databases (Teiid, Apache Drill, Exareme), **OBDA** (Ontop)



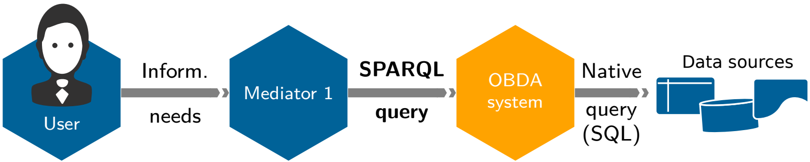
unibz

Choice 2: New representation of the data

New representation	Corresponding query language
Relational schema	SQL
JSON document	Mongo Aggregate, SQL (with e.g. Drill or Teiid)
XML document	XPath, XQuery, SQL (with e.g. Teiid)
RDF graph	SPARQL

unibz

Ontology-Based Data Access (OBDA)



Choice 1: How to derive data from the DBs

1. Extract Transform Load (ETL) process
2. **Virtual views**

Choice 2: How to represent the derived data

1. New relational schema, JSON or XML documents
2. **Resource Description Framework (RDF)**

unibz

Resource Description Framework (RDF)

RDF provides a description of the domain in terms of **triples**:

Subject → `<http://example.org/uni2/person/3>`

Predicate → `<http://xmlns.com/foaf/0.1/firstName>`

Object → "Céline"`^^xsd:string`

Triple elements: resources denoted by global identifiers (IRIs)

1. Subject: IRI of the described resource
2. Predicate: IRI of the property
3. Object: attribute value or IRI of another resource

Resource Description Framework (RDF)

RDF provides a description of the domain in terms of **triples**:

Subject Predicate Object

<http://example.org/uni2/person/3>
<http://xmlns.com/foaf/0.1/firstName>
“Céline”^^xsd:string

Triple elements: resources denoted by global identifiers (IRIs)

1. Subject: IRI of the described resource
2. Predicate: IRI of the property
3. Object: attribute value or IRI of another resource

Prefixes: useful abbreviations and/or references to external information

```
@prefix foaf: <http://xmlns.com/foaf/0.1/>
```

Resource Description Framework (RDF)

RDF provides a description of the domain in terms of **triples**:

Subject → `<http://example.org/uni2/person/3>`

Predicate → `foaf:firstName`

Object → "Céline"`^^xsd:string`

Triple elements: resources denoted by **global identifiers** (IRIs)

1. Subject: IRI of the described resource
2. Predicate: IRI of the property
3. Object: attribute value or IRI of another resource

Prefixes: useful abbreviations and/or references to external information

```
@prefix foaf: <http://xmlns.com/foaf/0.1/>
```

Resource Description Framework (RDF)

RDF provides a description of the domain in terms of **triples**:

Subject → `<http://example.org/uni2/person/3>`

Predicate → `foaf:firstName`

Object → "Céline"`^^xsd:string`

Triple elements: resources denoted by **global identifiers** (IRIs)

1. Subject: IRI of the described resource
2. Predicate: IRI of the property
3. Object: attribute value or IRI of another resource

Prefixes: useful abbreviations and/or references to external information

```
@prefix foaf: <http://xmlns.com/foaf/0.1/>
@prefix : <http://example.org/voc#>
```

Resource Description Framework (RDF)

RDF provides a description of the domain in terms of **triples**:

Subject → `<http://example.org/uni2/person/3>`

Predicate → `foaf:firstName`

Object → "Céline"`^^xsd:string`

Triple elements: resources denoted by global identifiers (IRIs)

1. Subject: IRI of the described resource
2. Predicate: IRI of the property
3. Object: attribute value or IRI of another resource

Prefixes: useful abbreviations and/or references to external information

```
@prefix foaf: <http://xmlns.com/foaf/0.1/>
@prefix : <http://example.org/voc#>
```

Resource Description Framework (RDF)

RDF provides a description of the domain in terms of **triples**:

Subject	Predicate	Object
`<uni2/person/3>`	`foaf:firstName`	"Céline"`^^xsd:string`

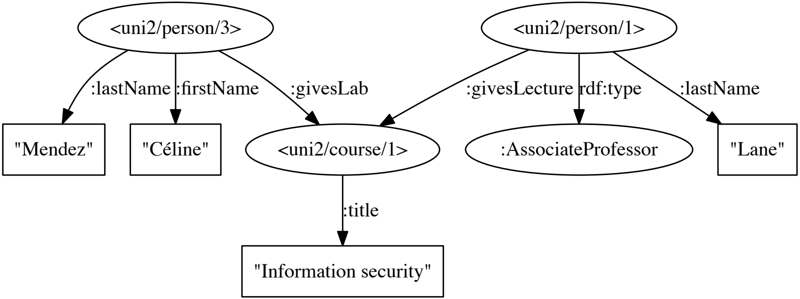
Triple elements: resources denoted by global identifiers (IRIs)

1. Subject: IRI of the described resource
2. Predicate: IRI of the property
3. Object: attribute value or IRI of another resource

Prefixes: useful abbreviations and/or references to external information

```
@prefix foaf: <http://xmlns.com/foaf/0.1/>
@prefix : <http://example.org/voc#>
```

RDF graph


<uni2/person/3>
<uni2/person/1>
:lastName
:firstName
:givesLab
:givesLecture
rdf:type
:lastName
"Mendez"
"Céline"
<uni2/course/1>
:AssociateProfessor
"Lane"
:title
"Information security"

SPARQL

SPARQL Protocol and RDF Query Language

Title of courses taught by a professor and professor names

```
PREFIX : <http://example.org/voc#>
# Other prefixes omitted

SELECT ?title ?fName ?lName {

  ?teacher rdf:type :Professor .
  ?teacher :teaches ?course .
  ?teacher foaf:lastName ?lName .

  ?course :title ?title .

  OPTIONAL {
    ?teacher foaf:firstName ?fName .
  }
}
```

Algebra

- Basic Graph Patterns
- OPTIONAL
- UNION
- GROUP BY
- MINUS
- FILTER NOT EXISTS

RDF Schema (RDFS)

Lightweight ontology

rdfs:subClassOf

	:AssociateProfessor rdfs:subClassOf :Professor . <uni1/academic/1> rdf:type :AssociateProfessor .
⟹	<uni1/academic/1> rdf:type :Professor .

rdfs:subPropertyOf

	:givesLecture rdfs:subPropertyOf :teaches . <uni2/academic/2> :givesLecture <uni2/course/1> .
⟹	<uni2/academic/2> :teaches <uni2/course/1> .

rdfs:domain

	:teaches rdfs:domain :Teacher . <uni2/academic/2> :teaches <uni2/course/1> .
⟹	<uni2/academic/2> rdf:type :Teacher .

rdfs:range

	:teaches rdfs:range :Course . <uni2/academic/2> :teaches <uni2/course/1> .
⟹	<uni2/course/1> rdf:type :Course .

Web Ontology Language (OWL)

Some constructs

owl:inverseOf

	`:isTaughtBy owl:inverseOf :teaches .` `<uni2/academic/2> :teaches <uni2/course/1> .`
$\Longrightarrow$	`<uni2/course/1> :isTaughtBy <uni2/academic/2> .`

owl:disjointWith

	`:Student owl:disjointWith :Professor .`
	`<uni1/academic/19> rdf:type Professor .`
	`<uni1/academic/19> rdf:type Student .`

$\Longrightarrow$	Inconsistent RDF graph

owl:sameAs

	`<uni2/person/2> :sameAs <uni1/academic/21> .` `<uni2/person/2> :teaches <uni2/course/1> .`
$\Longrightarrow$	`<uni1/academic/21> :teaches <uni2/course/1> .`

Full OWL 2 is very expressive

- Many more constructs
- Computation costs become easily prohibitive

Profile OWL 2 QL

Based on the Description Logic *DL-Lite*$_{\mathcal{R}}$

Supported constructs

- Class and property hierarchies (`rdfs:subClassOf` and (`rdfs:subPropertyOf`)
- Property domain and range (`rdfs:domain`, `rdfs:range`)
- Inverse properties (`owl:inverseOf`)
- Class disjunction (`owl:disjointWith`)
- Mandatory participation (advanced)

Not supported

- Individual identities (`owl:sameAs`)
- Cardinality constraints (functional property, etc.)
- Many other constructs

Summary

- Lightweight ontologies
- A bit more than RDFS
- First-order rewritability (rewritable into a SQL query)

unibz

Mappings RDB-RDF

Ontop native format (similar to the R2RML standard)

Source (SQL)

```
SELECT s_id, firstName, lastName
FROM uni1.student
```

Target (RDF, Turtle-like)

```
ex:uni1/student/{s_id} a :Student ;
          foaf:firstName "{firstName}"^^xsd:string ;
          foaf:lastName "{lastName}"^^xsd:string .
```

Result

- DBs unified into one RDF graph
- This graph can be queried with SPARQL

Mappings RDB-RDF
Other mappings

Object property (:teaches)

Target (RDF)	`ex:uni1/academic/{a_id} :teaches ex:uni1/course/{c_id} .`
Source	`SELECT * FROM "uni1"."teaching"`

Mappings RDB-RDF

Other mappings

Object property (`:teaches`)

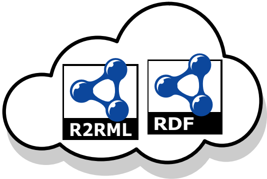
Target (RDF)	`ex:uni1/academic/{a_id} :teaches ex:uni1/course/{c_id} .`
Source	`SELECT * FROM "uni1"."teaching"`

Magic number

Target (RDF)	`ex:uni1/academic/{a_id} a :FullProfessor .`
Source	`SELECT * FROM "uni1"."academic" WHERE "position" = 1`

Querying the saturated RDF graph

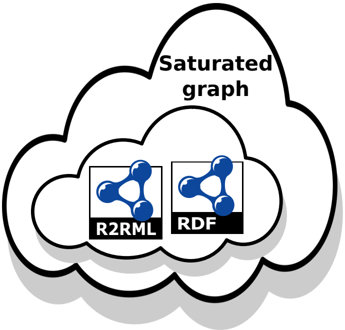
With SPARQL



Querying the saturated RDF graph
With SPARQL

Saturated RDF graph

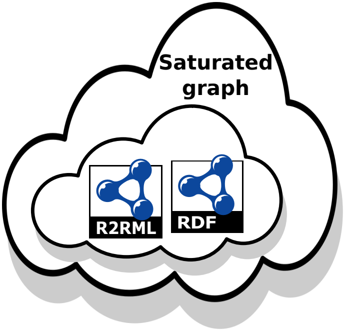
- Saturation of the RDF graph derived from the mappings
- According to the ontology constraints
- Usually much bigger graph!



Querying the saturated RDF graph
With SPARQL

Saturated RDF graph

- Saturation of the RDF graph derived from the mappings
- According to the ontology constraints
- Usually much bigger graph!



Materialized RDF graph

- ETL + saturation
- – Maintenance
- OWL 2 RL

Virtual RDF graph

- Query reformulation
- \+ No materialization (mapping saturation instead)
- OWL 2 QL

Mapping saturation example

TBox, user-defined mapping assertions and foreign key

$\mathsf{Student} \sqcup \mathsf{PostDoc} \sqcup \mathsf{AssociateProfessor} \sqcup \exists\mathsf{teaches} \sqsubseteq \mathsf{Person}$

$$\mathsf{Student}(\mathrm{URI}_1(p)) \leftarrow \texttt{uni1-student}(p, f, l) \quad (1)$$

$$\mathsf{PostDoc}(\mathrm{URI}_2(a)) \leftarrow \texttt{uni1-academic}(a, f, l, s), s = 9 \quad (2)$$

$$\mathsf{AssociateProfessor}(\mathrm{URI}_2(a)) \leftarrow \texttt{uni1-academic}(a, f, l, s), s = 2 \quad (3)$$

$$\mathsf{FacultyMember}(\mathrm{URI}_2(a)) \leftarrow \texttt{uni1-academic}(a, f, l, s) \quad (4)$$

$$\mathsf{teaches}(\mathrm{URI}_2(a), \mathrm{URI}_3(c)) \leftarrow \texttt{uni1-teaching}(c, a) \quad (5)$$

FK: $\exists y_1.\texttt{uni1-teaching}(y_1, x) \rightarrow \exists y_2 y_3 y_4.\texttt{uni1-academic}(x, y_2, y_3, y_4)$

Mapping saturation example

TBox, user-defined mapping assertions and foreign key

$\text{Student} \sqcup \text{PostDoc} \sqcup \text{AssociateProfessor} \sqcup \exists\text{teaches} \sqsubseteq \text{Person}$

$$\text{Student}(\mathrm{URI}_1(p)) \leftarrow \texttt{uni1-student}(p, f, l) \quad (1)$$

$$\text{PostDoc}(\mathrm{URI}_2(a)) \leftarrow \texttt{uni1-academic}(a, f, l, s), s = 9 \quad (2)$$

$$\text{AssociateProfessor}(\mathrm{URI}_2(a)) \leftarrow \texttt{uni1-academic}(a, f, l, s), s = 2 \quad (3)$$

$$\text{FacultyMember}(\mathrm{URI}_2(a)) \leftarrow \texttt{uni1-academic}(a, f, l, s) \quad (4)$$

$$\text{teaches}(\mathrm{URI}_2(a), \mathrm{URI}_3(c)) \leftarrow \texttt{uni1-teaching}(c, a) \quad (5)$$

FK: $\exists y_1.\texttt{uni1-teaching}(y_1, x) \rightarrow \exists y_2 y_3 y_4.\texttt{uni1-academic}(x, y_2, y_3, y_4)$

Non-optimized saturated mapping assertions for Person

$$\text{Person}(\mathrm{URI}_1(p)) \leftarrow \texttt{uni1-student}(p, f, l) \quad (6)$$

$$\text{Person}(\mathrm{URI}_2(a)) \leftarrow \texttt{uni1-academic}(a, f, l, s), s = 9 \quad (7)$$

$$\text{Person}(\mathrm{URI}_2(a)) \leftarrow \texttt{uni1-academic}(a, f, l, s), s = 2 \quad (8)$$

$$\text{Person}(\mathrm{URI}_2(a)) \leftarrow \texttt{uni1-academic}(a, f, l, s) \quad (9)$$

$$\text{Person}(\mathrm{URI}_2(a)) \leftarrow \texttt{uni1-teaching}(c, a) \quad (10)$$

Mapping saturation example

TBox, user-defined mapping assertions and foreign key

$\mathsf{Student} \sqcup \mathsf{PostDoc} \sqcup \mathsf{AssociateProfessor} \sqcup \exists \mathsf{teaches} \sqsubseteq \mathsf{Person}$

$$\mathsf{Student}(\mathrm{URI}_1(p)) \leftarrow \texttt{uni1-student}(p, f, l) \quad (1)$$

$$\mathsf{PostDoc}(\mathrm{URI}_2(a)) \leftarrow \texttt{uni1-academic}(a, f, l, s), s = 9 \quad (2)$$

$$\mathsf{AssociateProfessor}(\mathrm{URI}_2(a)) \leftarrow \texttt{uni1-academic}(a, f, l, s), s = 2 \quad (3)$$

$$\mathsf{FacultyMember}(\mathrm{URI}_2(a)) \leftarrow \texttt{uni1-academic}(a, f, l, s) \quad (4)$$

$$\mathsf{teaches}(\mathrm{URI}_2(a), \mathrm{URI}_3(c)) \leftarrow \texttt{uni1-teaching}(c, a) \quad (5)$$

FK: $\exists y_1.\texttt{uni1-teaching}(y_1, x) \rightarrow \exists y_2 y_3 y_4.\texttt{uni1-academic}(x, y_2, y_3, y_4)$

Non-optimized saturated mapping assertions for Person

$$\mathsf{Person}(\mathrm{URI}_1(p)) \leftarrow \texttt{uni1-student}(p, f, l) \quad (6)$$

$$\mathsf{Person}(\mathrm{URI}_2(a)) \leftarrow \texttt{uni1-academic}(a, f, l, s), s = 9 \quad (7)$$

$$\mathsf{Person}(\mathrm{URI}_2(a)) \leftarrow \texttt{uni1-academic}(a, f, l, s), s = 2 \quad (8)$$

$$\mathsf{Person}(\mathrm{URI}_2(a)) \leftarrow \texttt{uni1-academic}(a, f, l, s) \quad (9)$$

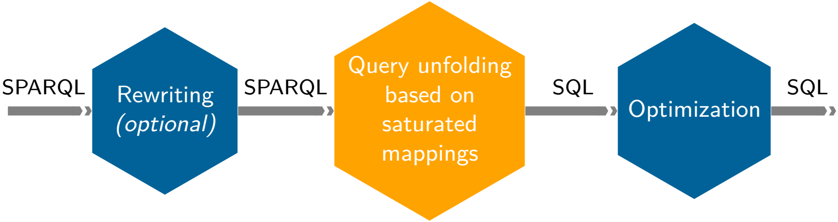
$$\mathsf{Person}(\mathrm{URI}_2(a)) \leftarrow \texttt{uni1-teaching}(c, a) \quad (10)$$

Mapping assertions for Person after optimization (query containment)

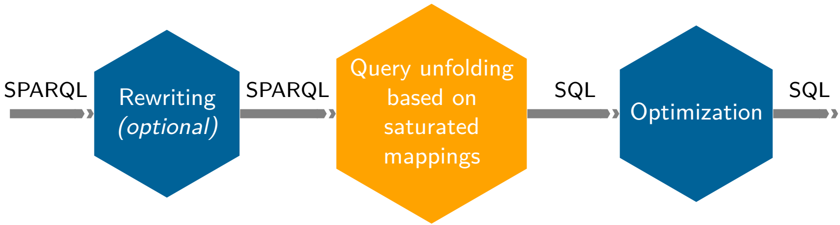
$$\mathsf{Person}(\mathrm{URI}_1(p)) \leftarrow \texttt{uni1-student}(p, f, l) \quad (11)$$

$$\mathsf{Person}(\mathrm{URI}_2(p)) \leftarrow \texttt{uni1-academic}(p, f, l, s) \quad (12)$$

Query reformulation



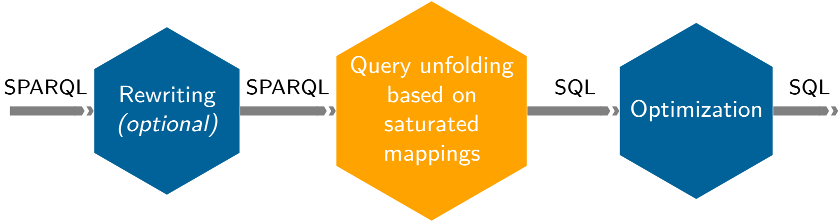
Query reformulation



Role of the OWL 2 QL ontology

- Minor: SPARQL query rewriting *(very specific cases)*
- Main: mapping saturation *(offline)*

Query reformulation



Role of the OWL 2 QL ontology

- Minor: SPARQL query rewriting *(very specific cases)*
- Main: mapping saturation *(offline)*

Mapping saturation

- Query containment optimization
- Not only OWL 2 QL:
  - Horn fragment of OWL 2 [Botoeva *et al.*, 2016]
  - SWRL with linear recursion [Xiao *et al.*, 2014]

SQL query optimization

Objective : produce a SQL query. . .

- Similar to manually written ones
- Adapted to existing query planners

SQL query optimization

Objective : produce a SQL query. . .

- Similar to manually written ones
- Adapted to existing query planners

Structural optimization

- From Join-of-unions to union-of-joins
- IRI decomposition for pruning and improving joining perf.

SQL query optimization

Objective : produce a SQL query. . .

- Similar to manually written ones
- Adapted to existing query planners

Structural optimization

- From Join-of-unions to union-of-joins
- IRI decomposition for pruning and improving joining perf.

Semantic optimization

- Redundant join elimination
- Redundant union elimination
- Using DB constraints

SQL query optimization

Objective : produce a SQL query. . .

- Similar to manually written ones
- Adapted to existing query planners

Structural optimization

- From Join-of-unions to union-of-joins
- IRI decomposition for pruning and improving joining perf.

Semantic optimization

- Redundant join elimination
- Redundant union elimination
- Using DB constraints

DB constraints

- Unique constraints (e.g. primary keys)
- Inclusion dependencies (foreign keys)
- Vital for query reformulation!

Ontop

http://ontop.inf.unibz.it

Ontop framework

- Started in 2010
- Open-source *(Apache 2)*
- W3C standard compliant *(SPARQL, OWL 2 QL, R2RML)*
- Supports all major relational DBs *(Oracle, DB2, Postgres, MySQL, etc.)* and some virtual DBs *(Teiid, Exareme)*

Components

- Java APIs
- Protégé extension (GUI)
- Sesame/RD4J endpoint

Integration

- Stardog 4.0 (virtual graphs)
- Fluidops Information Workbench
- Metaphacts semantic data management platform

Ontop

http://ontop.inf.unibz.it

ontop

Ontop framework

- Started in 2010
- Open-source *(Apache 2)*
- W3C standard compliant *(SPARQL, OWL 2 QL, R2RML)*
- Supports all major relational DBs *(Oracle, DB2, Postgres, MySQL, etc.)* and some virtual DBs *(Teiid, Exareme)*

Version 3

Beta in a few weeks

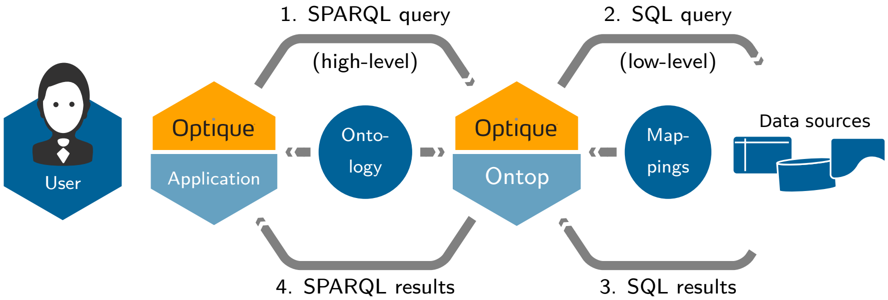
Components

- Java APIs
- Protégé extension (GUI)
- Sesame/RD4J endpoint

Integration

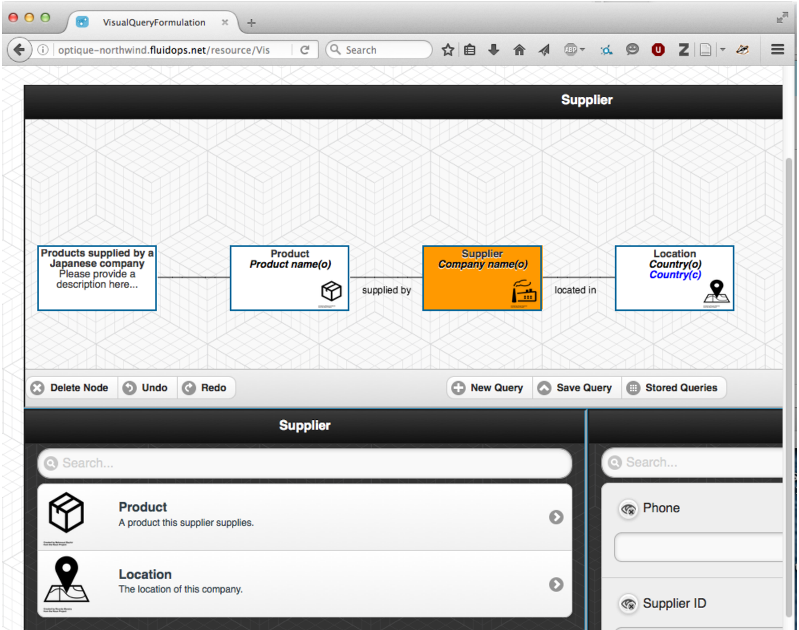
- Stardog 4.0 (virtual graphs)
- Fluidops Information Workbench
- Metaphacts semantic data management platform

Optique platform



Visual query formulation (Optique VQS)

`http://optique-northwind.fluidops.net` demo/demo



Conclusion

Main message: we need high-level access to data

1. SQL queries over tables can be difficult to write manually (low-level)
2. OBDA is a powerful solution for high-level data access
3. Ontop is an open-source OBDA framework

Work in progress

- Nested data (MongoDB)
- Streaming and temporal reasoning
- Better SPARQL `OPTIONAL`
- SPARQL aggregation
- SPARQL `MINUS`

Links

- Github : ontop/ontop
- ontop4obda@googlegroups.com
- Twitter : @ontop4obda
- `http://ontop.inf.unibz.it`

Ontop team

- Diego Calvanese
- Guohui Xiao
- Elena Botoeva
- Julien Corman
- Sarah Komla-Ebri
- Davide Lanti
- Elem Güzel Kalayci
- Vladislav Ryzhikov
- Roman Kontchakov (Birkbeck, London)
- Dag Hovland (Oslo)
- Mariano Rodriguez-Muro (now in IBM Research, NY)
- Martin Rezk (now in Rakuten, Tokyo)
- Me

References I

[Botoeva *et al.*, 2016] Elena Botoeva, Diego Calvanese, Valerio Santarelli, Domenico F. Savo, Alessandro Solimando, and Guohui Xiao.
Beyond OWL 2 QL in OBDA: Rewritings and approximations.
In *Proc. of the 30th AAAI Conf. on Artificial Intelligence (AAAI)*, 2016.

[Xiao *et al.*, 2014] Guohui Xiao, Martin Rezk, Mariano Rodriguez-Muro, and Diego Calvanese.
Rules and ontology based data access.
volume 8741 of *LNCS*, pages 157–172. Springer, 2014.