Query answering and query abstraction through ontologies

Maurizio Lenzerini

Dipartimento di Ingegneria Informatica
Automatica e Gestionale Antonio Ruberti

SAPIENZA
UNIVERSITÀ DI ROMA

EROSS 2020

ER Online Summer Seminars

October 14, 2020

ER Online Summer Seminars: very interesting event

EROSS

The KRDB Research Centre for Knowledge and Data at th
Summer Seminars" (EROSS-2020), an interdisciplinary se
such as Ontology, Data Semantics, Artificial Intelligence, S

ER Online Summer Seminars: very, very interesting event

EROSS

The KRDB Research Centre for Knowledge and Data at th
Summer Seminars" (EROSS-2020), an interdisciplinary se
such as Ontology, Data Semantics, Artificial Intelligence, S

concepts *carve reality at its joints*

ER Online Summer Seminars: very, very, very interesting event

EROSS

The KRDB Research Centre for Knowledge and Data at th
Summer Seminars" (EROSS-2020), an interdisciplinary se
such as Ontology, Data Semantics, Artificial Intelligence, S

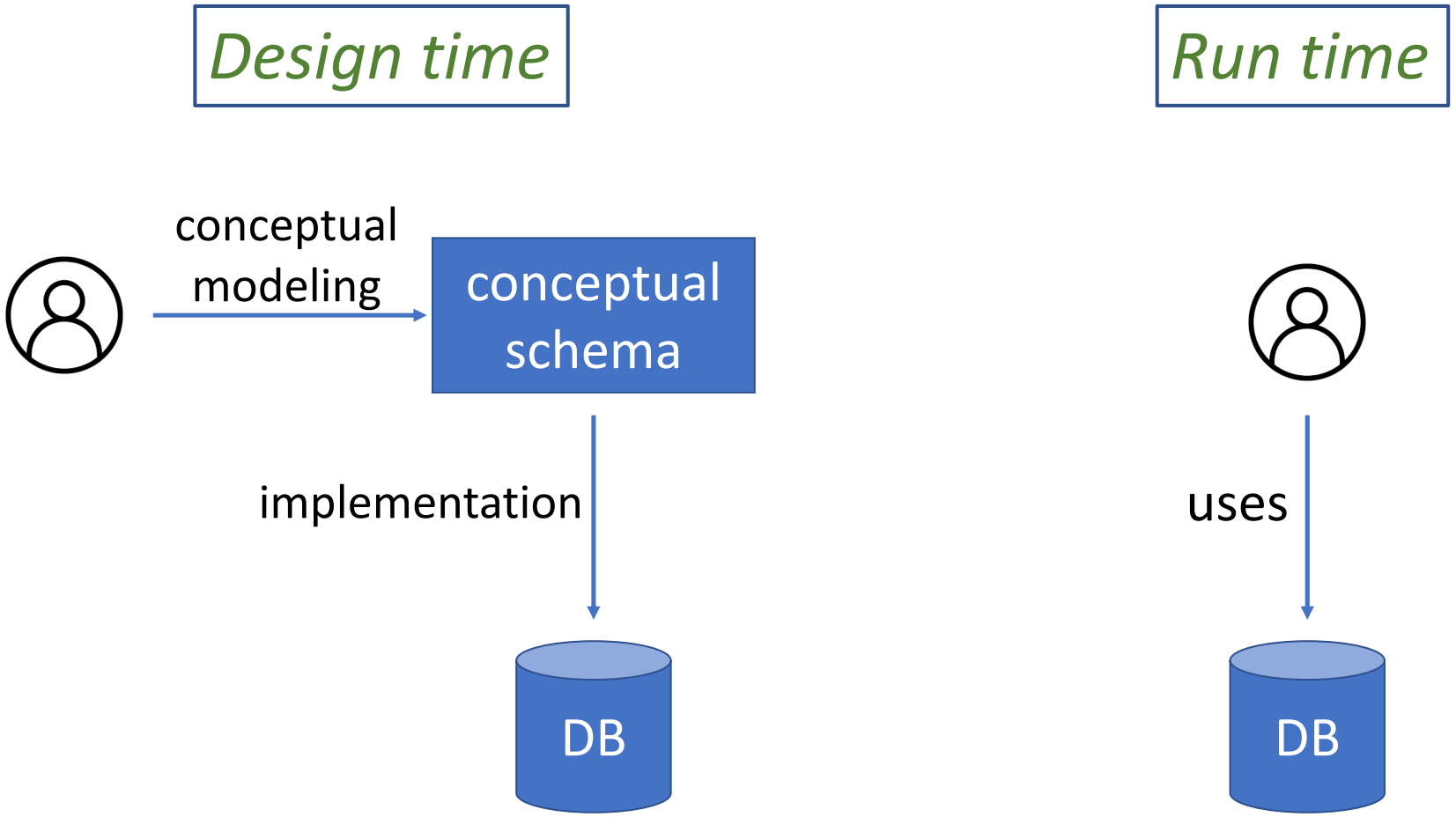
concepts *carve reality at its joints*

Next steps (joint wor

Tobias Brock and roll **Artem**

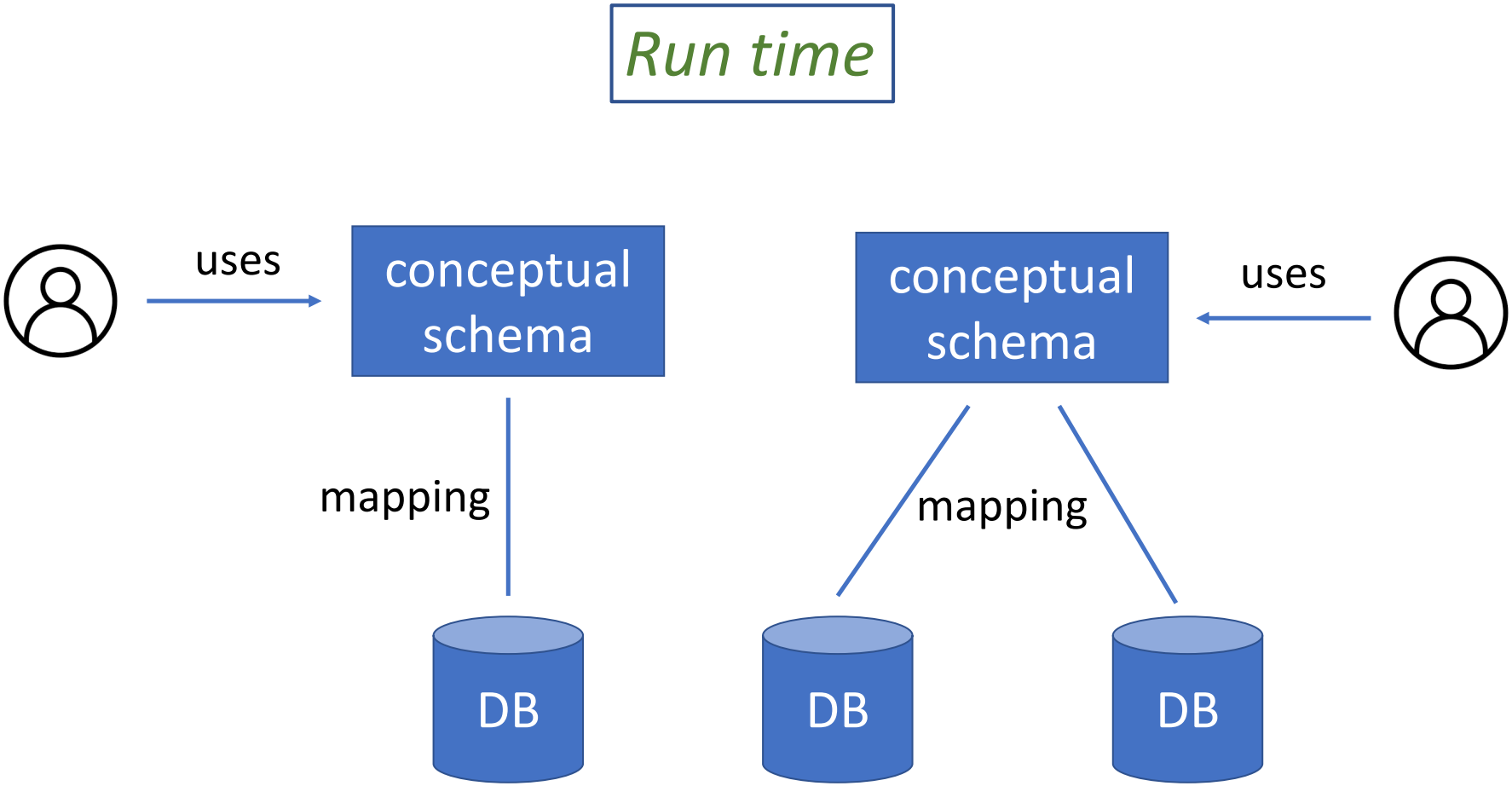
- Approach solves many of th
  conformance checking (e.g.
- Why consider frequencies ir
- Challenges currently being a

Conceptual modeling in the '70s: top-down data design



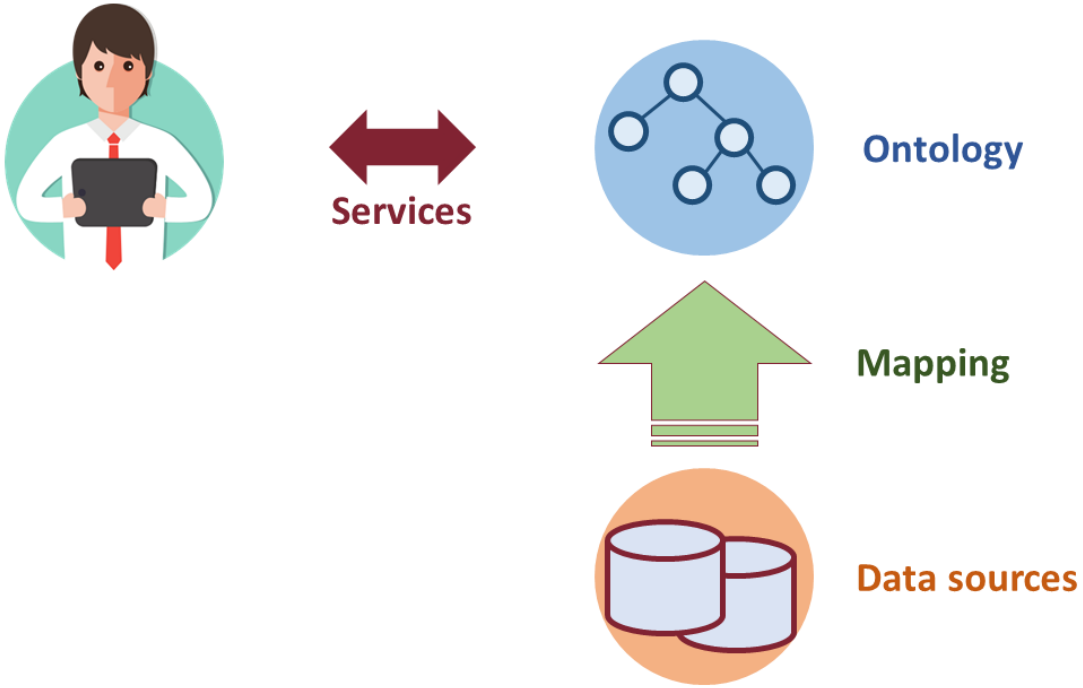
Why does the conceptual disappear at run time?

(My) research question



Principles and tools for "using" the conceptual schema at run time

Ontology-based data management

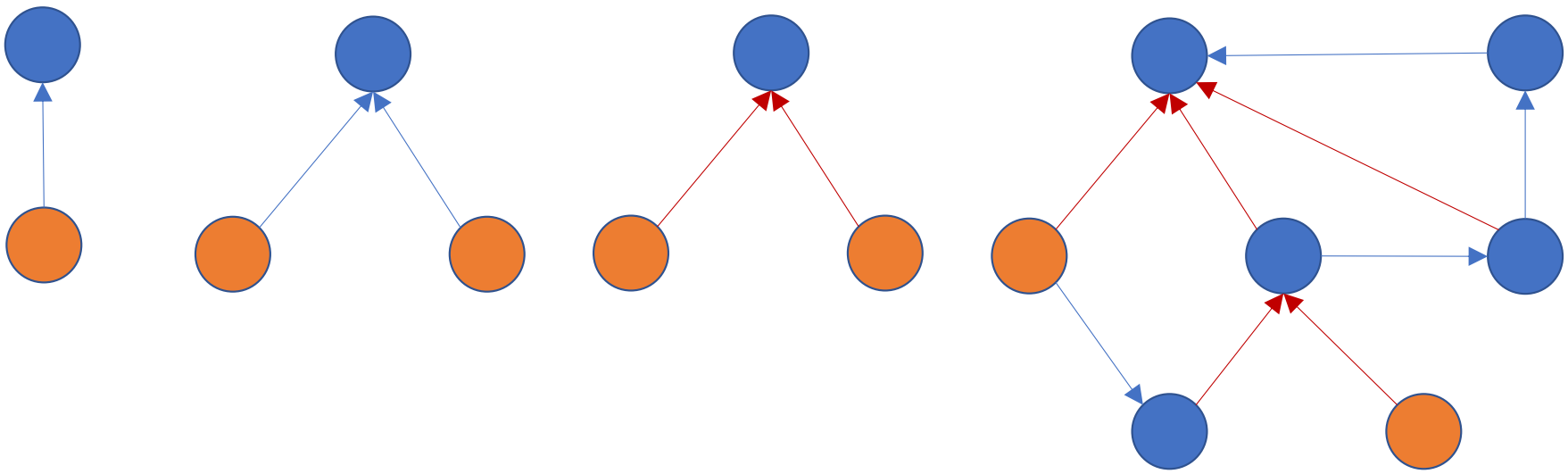


Based on three main components:

- Ontology, a declarative, logic-based specification of the domain of interest, used as a unified, conceptual view for clients
- Data sources, representing external, independent, heterogeneous, storage (or, more generally, computational) structures
- Mappings, used to semantically link data at the sources to the ontology

Abstraction for several data management and interoperability scenarios

Data interoperability architectures



Data independence [Date et al.]

Data integration [Doan et al. 2012]

Data exchange [Arenas et al. 2014]

Collaborative data sharing [Karvounarakis et al.2013]

- Data independence at run time (data access through ontology)
- (Virtual) data integration (data federation)
- Data exchange/consolidation (materialized data integration, ETL, ELT)
- Collaborative data sharing (P2P data integration, Peer data management)

Mappings

The relationship between two nodes of the data interoperability network is specified by means of a set of mapping assertions.

Syntax of mapping assertion

Each mapping assertions from (source) node $\mathcal{S}$ to (target) node $\mathcal{G}$ has the form

$$\forall \vec{x}\ (\Phi(\vec{x}) \rightarrow \Psi(\vec{x}))$$

where

- $\Phi(\vec{x})$ is a query over $\mathcal{S}$, whose free variables are $\vec{x}$
- $\Psi(\vec{x})$ is a query over $\mathcal{G}$, whose free variables are from $\vec{x}$.

Intuitive semantics of mapping assertion

The data items in $\mathcal{S}$ satisfying the pattern expressed as $\Phi(\vec{x})$ "match" the data items in $\mathcal{G}$ satisfying the pattern expressed as $\Psi(\vec{x})$.

Types of mappings

Taxonomy from data integration [L. 2002]

- Global as view (GAV) – $\Psi(\vec{x})$ is an **atom** $v(\vec{x})$

$$\forall \vec{x}\ (\Phi(\vec{x}) \rightarrow v(\vec{x}))$$

  the element v of the target node is associated with a view over the source node

- Local as view (LAV) – $\Phi(\vec{x})$ is an **atom** $v(\vec{x})$

$$\forall \vec{x}\ (v(\vec{x}) \rightarrow \Psi(\vec{x}))$$

  the element v of the source node is associated with a view over the target node

- GLAV

$$\forall \vec{x}\ (\Phi(\vec{x}) \rightarrow \Psi(\vec{x}))$$

  a view over the source node is associated to a view over the target node

Example of mapping

Consider the following mapping relating Node 1 to Node 2:

$$\forall x \forall y \forall z \ \mathsf{Student}(x) \wedge \mathsf{Grade}(x, y, z) \rightarrow \exists w \ \mathsf{Teaches}(w, y) \wedge \mathsf{Enrolled}(x, y)$$

- Data at Node 1:
  { Student(Mario), Grade(Mario,IS,B), Student(Anna), Grade(Anna,AI,A) }

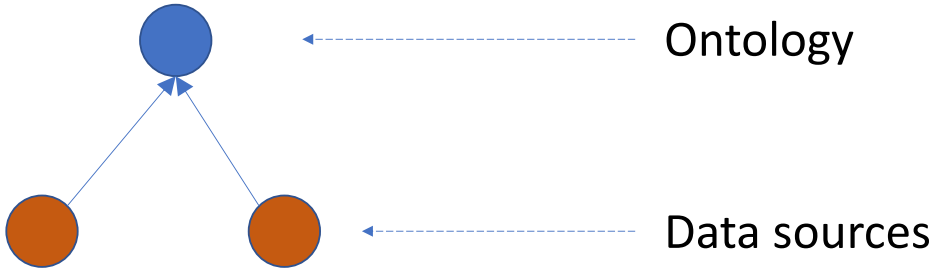
After data exchange:

- Data at Node 2 after data exchange:
  { Teaches(w_1,IS), Enrolled(Mario,IS), Teaches(w_2,AI), Enrolled(Anna,AI) }

This talk: how to use mappings for processing queries in ontology-based data management

- Compile time
  - Discovery
  - Analysis (equivalence, redundancy, optimization)
  - Reasoning (inverse, composition, etc.)

- **Run time**
  - Exchanging data
  - Update propagation
  - Translation
  - **Processing queries in ontology-based data management**



Outline

Outline

Formal framework of ontology-based data management

An ontology-based data management (OBDM) **specification** [Poggi et al. 2008, L. 2018] is a triple $\mathcal{J} = \langle \mathcal{O}, \mathcal{M}, \mathcal{S} \rangle$, where

- $\mathcal{O}$ is the ontology, expressed as a logical theory – here, as usual, a TBox in a **Description Logic** (DL)
- $\mathcal{S}$ is a data schema representing the data sources – here, a (federated) relational schema
- $\mathcal{M}$ is a set of mapping assertions, each of the form

$$\forall \vec{x} \; (\Phi(\vec{x}) \rightarrow \Psi(\vec{x}))$$

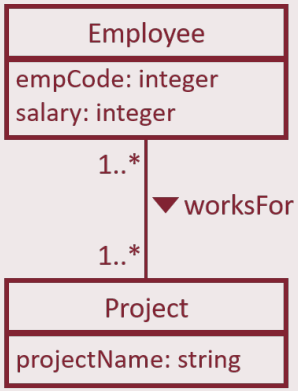
  where (in this talk we focus on conjunctive queries, but we could be more general)
  - $\Phi(\vec{x})$ is a conjunctive query (SPJ query) over $\mathcal{S}$, with free variables $\vec{x}$
  - $\Psi(\vec{x})$ is a conjunctive query (SPJ query) over $\mathcal{O}$, with free variables from $\vec{x}$.

An OBDM **system** is a pair $(\mathcal{J}, D)$, where $\mathcal{J} = \langle \mathcal{O}, \mathcal{M}, \mathcal{S} \rangle$ is an OBDM specification, and D is an $\mathcal{S}$-database, i.e., a source database that is legal for $\mathcal{S}$.

Ontology-based data management specification – Example

Ontology $\mathcal{O}$ (TBox)

$\text{Employee} \sqsubseteq \exists \text{worksFor}$
$\text{Employee} \sqsubseteq \exists \text{empCode}$
$\text{Employee} \sqsubseteq \exists \text{salary}$
$\text{Project} \sqsubseteq \exists \text{worksFor}^-$
$\text{Project} \sqsubseteq \exists \text{projectName}$
$\exists \text{worksFor} \sqsubseteq \text{Employee}$
$\exists \text{worksFor}^- \sqsubseteq \text{Project}$



Federated source schema $\mathcal{S}$

$D_1[SSN\text{: String}, PrName\text{: String}]$
Employees and Projects they work for

$D_2[Code\text{: String}, Salary\text{: Int}]$
Employee's Code with salary

$D_3[Code\text{: String}, SSN\text{: String}]$
Employee's Code with SSN

...

Mapping $\mathcal{M}$

M_1:
```
SELECT SSN,PrName
FROM D1
```
$\rightarrow$ Employee(**pe**(*SSN*)), Project(**pr**(*PrName*)), projectName(**pr**(*PrName*), *PrName*), workFor(**pe**(*SSN*), **pr**(*PrName*))

M_2:
```
SELECT Code,Salary
FROM D2,D3
WHERE D2.Code = D3.Code
```
$\rightarrow$ Employee(**pe**(*SSN*)), salary(**pe**(*SSN*), *Salary*)

M_3:
```
SELECT Code,Salary
FROM D2,D3
WHERE D2.Code = D3.Code
   AND SSN NOT IN
   (SELECT SSN FROM D1)
```
$\rightarrow$ Employee(**pe**(*SSN*)), salary(**pe**(*SSN*), *Salary*)

Ontology-based data management: Semantics

Let $\mathcal{J} = \langle \mathcal{O}, \mathcal{M}, \mathcal{S} \rangle$ be an OBDM specification, and let $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ be an interpretation for the ontology $\mathcal{O}$.

Def.: Semantics

The semantics of $\mathcal{J}$ is given with respect to an $\mathcal{S}$-database D. $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ is a model of $(\mathcal{J}, D)$ if:

- $\mathcal{I}$ satisfies all axioms of $\mathcal{O}$, i.e., is a model of $\mathcal{O}$;
- $\mathcal{I}$ satisfies $\mathcal{M}$ wrt D, i.e., satisfies every assertion $\Phi(\vec{x}) \rightarrow \Psi(\vec{x})$ in $\mathcal{M}$ wrt D, which means that the sentence $\forall \vec{x}\ (\Phi(\vec{x}) \rightarrow \Psi(\vec{x}))$ is true in $\mathcal{I} \cup D$.

$(\mathcal{J}, D)$ is satisfiable or consistent, if it admits at least one model.

Semantics of queries

The semantics of a query $q(\vec{x})$ posed to the OBDM system $(\mathcal{J}, D)$ is given in terms of the set $cert(q, \mathcal{J}, D)$ of **certain answers**, i.e., those answers that hold in **all the models** of $(\mathcal{J}, D)$.

Ontology-mediated query answering

A DL-knowledge base is a pair $\langle \mathcal{T}, \mathcal{A} \rangle$, where $\mathcal{T}$ is a TBox (also called ontology, i.e., general axioms on concepts and relations) and $\mathcal{A}$ is an ABox (specific axioms on individuals)

Ontology-mediated query answering, or query answering over DL-KBs

Given a DL-knowledge base $\langle \mathcal{T}, \mathcal{A} \rangle$, a query q over such KB and a tuple a of constants in $\mathcal{A}$, check whether a is a certain answer to q over $\langle \mathcal{T}, \mathcal{A} \rangle$, i.e., if $q(a)$ is true in every model of $\mathcal{T} \cup \mathcal{A}$.

Let us denote by $\mathcal{M}(D)$ the ABox obtained by "applying" the mapping $\mathcal{M}$ to D (i.e., by (i) computing the answer to the left-hand-side query of each mapping assertion m, and (ii) for each such answer adding into the ABox the "corresponding" tuple satisfying the right-hand-side query of m)

Proposition

If q is a query posed to $\mathcal{J} = \langle \mathcal{O}, \mathcal{M}, \mathcal{S} \rangle$ and D is an $\mathcal{S}$-database, then $a \in cert(q, \mathcal{J}, D)$ if and only if a is a certain answer to q over $\langle \mathcal{O}, \mathcal{M}(D) \rangle$.

Outline

Two fundamental problems regarding queries

Given OBDM specification $\mathcal{J} = \langle \mathcal{O}, \mathcal{M}, \mathcal{S} \rangle$

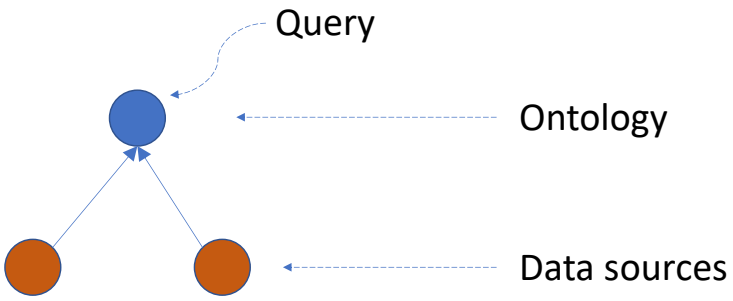
1. Query answering:
   given a query $q_{\mathcal{O}}$ over the ontology $\mathcal{O}$, compute a query $q_{\mathcal{S}}$ that captures the certain answers to $q_{\mathcal{O}}$ at best under $\mathcal{J}$

   $\longrightarrow$ Direct rewriting (or, Ontology-to-Source rewriting – top-down)

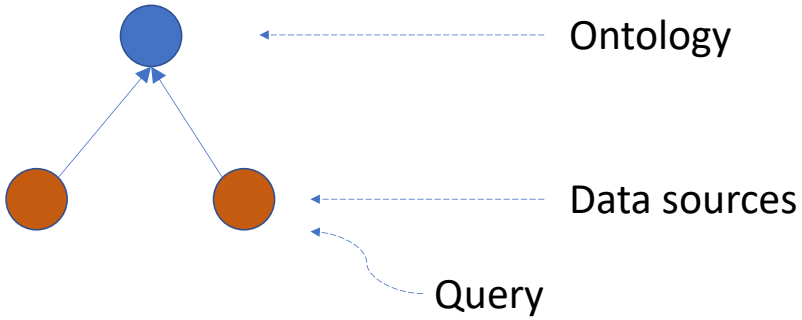
2. Query abstraction:
   given a query $q_{\mathcal{S}}$ over the source schema $\mathcal{S}$, compute a query $q_{\mathcal{O}}$ whose certain answers capture $q_{\mathcal{S}}$ at best under $\mathcal{J}$

   $\longrightarrow$ Reverse rewriting (or, Source-to-Ontology rewriting – bottom-up)

Usages of query answering and abstraction



1. Query answering:
   - Compute the certain answers of a query expressed over the ontology
   - Look for source data that are inconsistent/incomplete with respect to the ontology
   - Tell me which sources store data corresponding to instances of relevant concepts/relationships, or relevant views of such ontology elements
2. Query abstraction:
   - Explain the content of a data source in terms of the ontology
   - Verify if a given data service expressed over the data sources can be expressed in terms of the ontology
   - Automatically associate semantics to open data sets



Outline

Computing certain answers

Problem

Given an OBDM specification $\mathcal{J} = \langle \mathcal{O}, \mathcal{M}, \mathcal{S} \rangle$ and a query $q_{\mathcal{O}}$ over the ontology $\mathcal{O}$, compute a query $q_{\mathcal{S}}$ over the source schema $\mathcal{S}$ that best characterizes $q_{\mathcal{O}}$ under $\mathcal{J}$.

The best characterization is obtained by the so-called perfect rewriting.

Formal definition of perfect rewriting

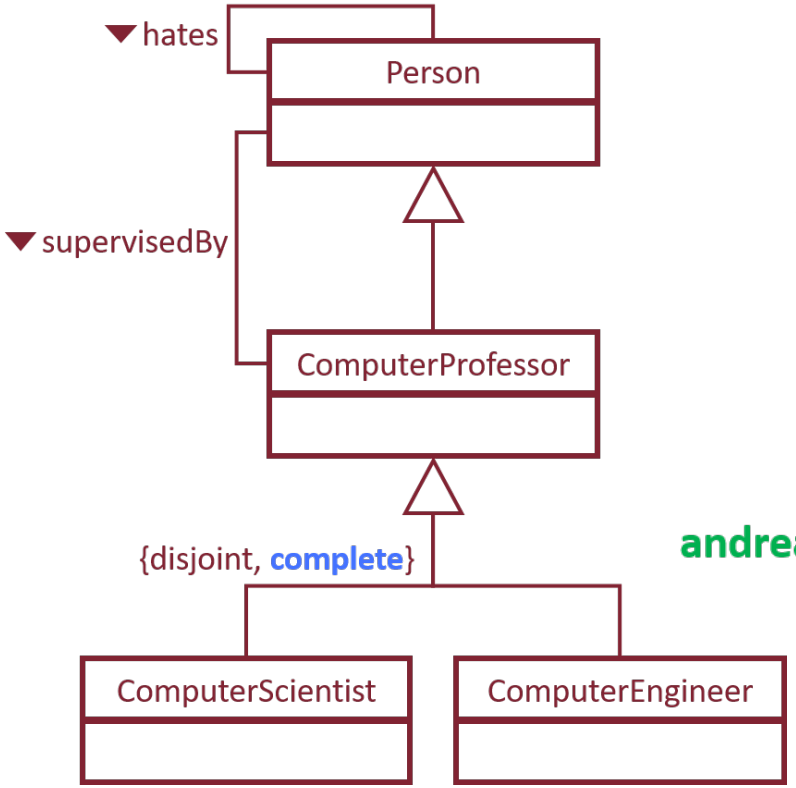
A query $q_{\mathcal{S}}$ over $\mathcal{S}$ is a **perfect $\mathcal{O}$-to-$\mathcal{S}$ $\mathcal{J}$-rewriting of** $q_{\mathcal{O}}$ if for every $\mathcal{S}$-database D, $q_{\mathcal{S}}^{D}$ coincides with the certain answers $cert(q_{\mathcal{O}}, \mathcal{J}, D)$.

Two basic computational problems:

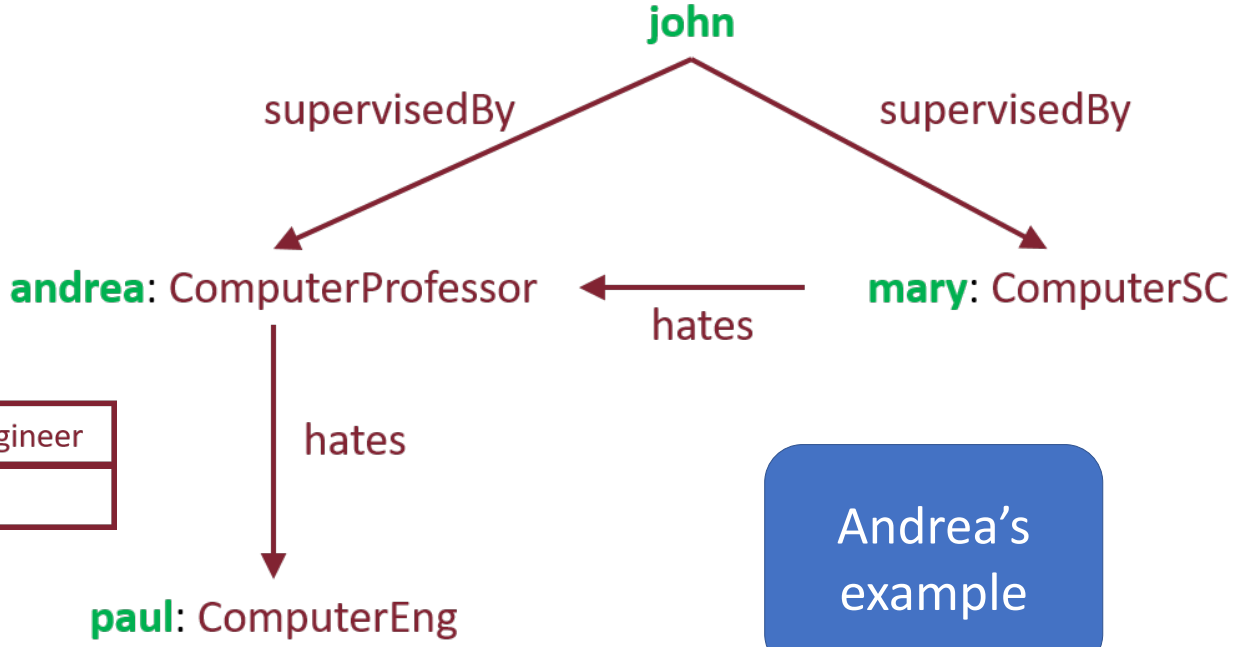
- Verification (check if $q_{\mathcal{S}}$ is a perfect $\mathcal{O}$-to-$\mathcal{S}$ $\mathcal{J}$-rewriting of $q_{\mathcal{O}}$)
- Computation (compute the perfect $\mathcal{O}$-to-$\mathcal{S}$ $\mathcal{J}$-rewriting of $q_{\mathcal{O}}$)

Starting from [Calvanese et al. 2007], computing perfect $\mathcal{O}$-to-$\mathcal{S}$ $\mathcal{J}$-rewritings in several scenarios is one of the most studied problems in KR&R in recent years.

Direct rewriting for computing certain answer



Note that ComputerProfessor is partitioned into ComputerScientist and ComputerEngineer.
Here is the ABox corresponding to $\mathcal{M}(D)$, rendered as a "knowledge graph":



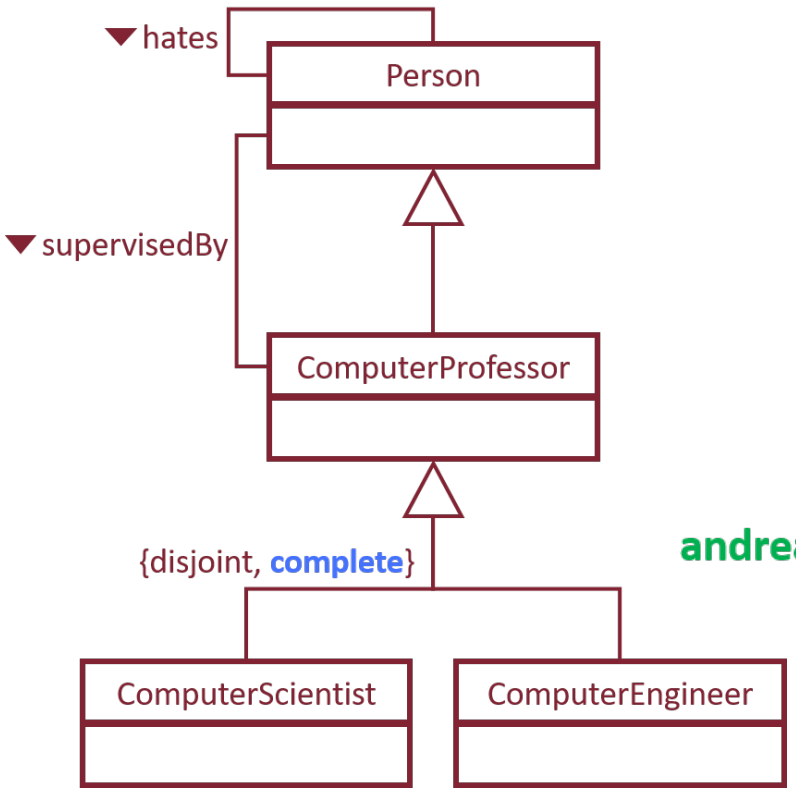
Andrea's example

Query:

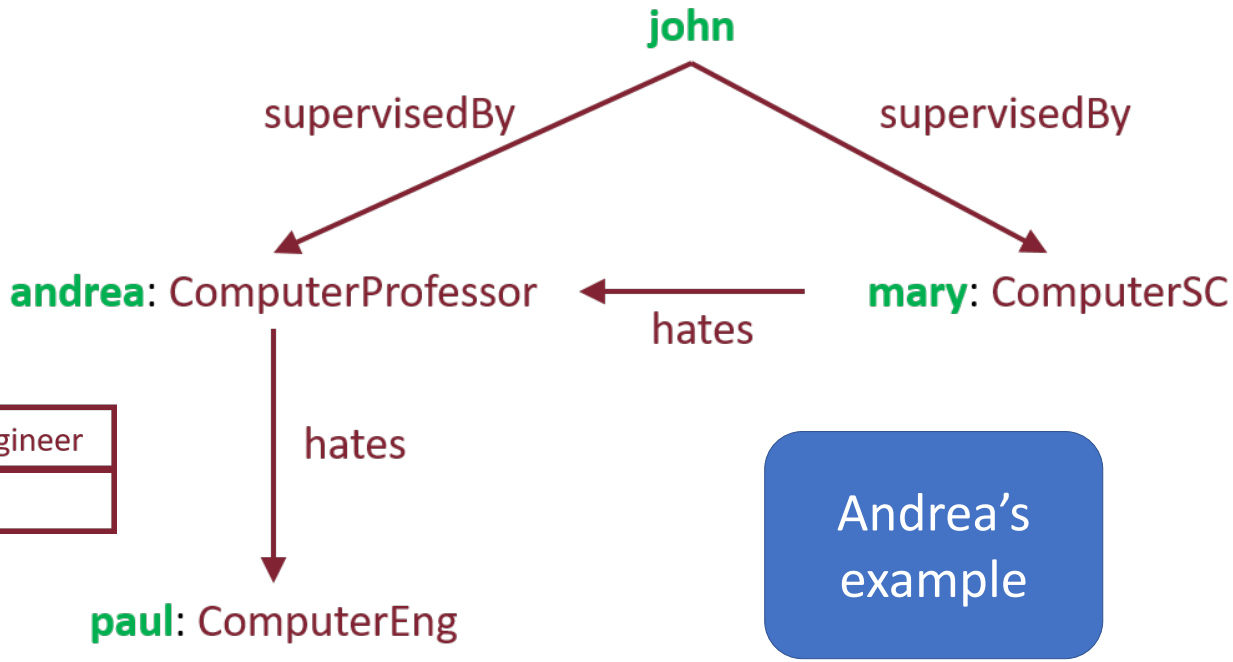
$\{\ (x) \mid \exists y, z.\, \mathsf{supervisedBy}(x, y), \mathsf{ComputerSC}(y), \mathsf{hates}(y, z), \mathsf{ComputerEng}(z)\ \}$

Answer: ???

Direct rewriting for computing certain answer



Note that ComputerProfessor is partitioned into ComputerScientist and ComputerEngineer.
Here is the ABox corresponding to $\mathcal{M}(D)$, rendered as a "knowledge graph":



Andrea's example

Query:

$\{ (x) \mid \exists y, z. \mathsf{supervisedBy}(x, y), \mathsf{ComputerSC}(y), \mathsf{hates}(y, z), \mathsf{ComputerEng}(z) \}$

Answer: $\{$ `john` $\}$ To obtain this answer, we need to reasoning by cases

$\longrightarrow$ **No first-order query is a perfect ontology-to-source rewriting of q**

Complexity of conjunctive query answering in DLs

Studied extensively for (unions of) CQs and various ontology languages:

	Combined complexity	Data complexity
Plain databases	NP-complete	AC^0 (1)
OWL 2	???	coNP-hard (2)

(1) This makes it possible to scale with the data.

(2) Already for a TBox with a single disjunction (see example above).

Research question

Can we find interesting DLs for which we can always rewrite the query over the ontology into a FOL query over the source? A lot of research work/answers in the last decade!

Complexity of conjunctive query answering in DLs

Studied extensively for (unions of) CQs and various ontology languages:

	Combined complexity	Data complexity
Plain databases	NP-complete	AC^0 (1)
OWL 2	???	coNP-hard (2)

(1) This makes it possible to scale with the data.

(2) Already for a TBox with a single disjunction (see example above).

Research question

Can we find interesting DLs for which we can always rewrite the query over the ontology into a FOL query over the source? A lot of research work/answers in the last decade!

The *DL-Lite* family

DL-Lite is a family [Calvanese et al 2007] of DLs optimized according to the **tradeoff** between **expressive power** and **complexity** of query answering, with emphasis on **data**.

- The same complexity as relational databases.
- In fact, query answering is FOL-rewritable and hence can be delegated to a relational DB engine.

Nevertheless they have the right expressive power to capture the essential features of conceptual modeling formalisms.

DL-Lite provides robust foundations for **Ontology-Based Data Management**.

The *DL-Lite* family is at the basis of the OWL 2 QL profile of the W3C standard Web Ontology Language OWL.

Capturing basic ontology constructs in *DL-Lite*

Modeling construct	*DL-Lite*
ISA between classes	$Student \sqsubseteq Person$
... and or relations	$isMatherOf \sqsubseteq isParentOf$
Disjointness between classes	$Student \sqsubseteq \neg Professor$
... and or relations	$isMatherOf \sqsubseteq \neg isFatherOf$
Domain of relations	$\exists livesIn \sqsubseteq Person$
Range of rleaitons	$\exists livesIn^- \sqsubseteq City$
Mandatory participation *(min card = 1)*	$Person \sqsubseteq \exists livesIn$ $City \sqsubseteq \exists livesIn^-$
Functionality of relations *(max card = 1)* *but a functional relation cannot be specialized*	(**funct** $livesIn$) (**funct** $livesIn^-$)

Note: *DL-Lite* distinguishes between abstract objects and data values as well (**we can represent concept attributes**) (ignored here).

Query answering in *DL-Lite*-based OBDM systems

In a *DL-Lite*-based OBDM specification $\langle \mathcal{O}, \mathcal{M}, \mathcal{S} \rangle$

- $\mathcal{O}$ is expressed in *DL-Lite*
- $\mathcal{M}$ is a set of GAV mapping assertions (the right-hand side Ψ is a conjunctive query (CQ) without existential variables).
- queries over $\mathcal{O}$ are unions of conjunctive queries (UCQs).

Query answering is performed through ontology-to-source rewriting:

Given a (U)CQ q, OBDM specification $\mathcal{J} = \langle \mathcal{O}, \mathcal{M}, \mathcal{S} \rangle$, $\mathcal{S}$-database D, we compute $cert(q, \mathcal{J}, D)$ as follows:

1. we compute the **ontology rewriting** $r_{q,\mathcal{O}}$ of q using $\mathcal{O}$;
2. we compute the **mapping rewriting** r of $r_{q,\mathcal{O}}$ using $\mathcal{M}$;
3. evaluate the UCQ r directly over D.

Correctness of this procedure shows that r is the perfect ontology-to-source rewriting of q wrt $\mathcal{J}$, and shows **FOL-rewritability** of query answering in *DL-Lite*.

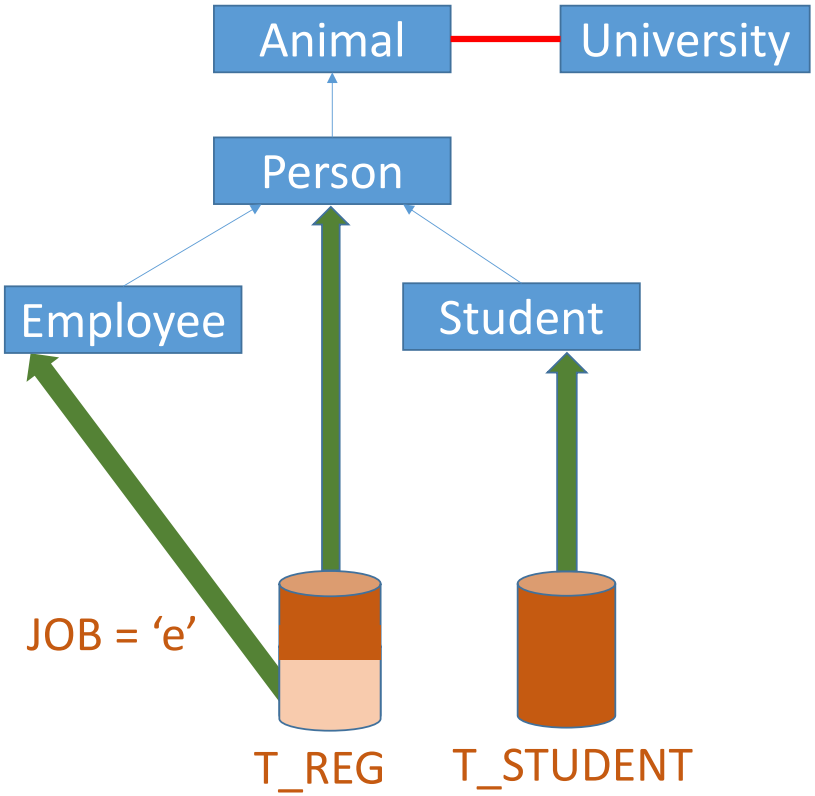
SAPIENZA UNIVERSITÀ DI ROMA

Query answering in *DL-Lite*-based OBDM systemse

$\mathcal{S}$ contains two tables

- T_REG(ID,JOB)
- T_STUDENT(ID,UNIVERSITY)

Ontology query $q_{\mathcal{O}}$:
$\{ (x) \mid \text{Person}(x) \}$



Ontology rewriting: $\{ (x) \mid \text{Person}(x) \vee \text{Employee}(x) \vee \text{Student}(x) \}$

Mapping rewriting: $\{ (x) \mid \text{T\_REG}(x, y) \vee \text{T\_STUDENT}(x, z) \}$

Computational complexity of query answering

Proposition

Query answering on a *DL-Lite* ontology-based data management system $(\langle \mathcal{O}, \mathcal{M}, \mathcal{S} \rangle, D)$ of the kind considered so far is

1. PTIME in the size of the ontology $\mathcal{O}$ and the mappings $\mathcal{M}$.
2. AC^0 in the size of the database D, in fact FOL-rewritable.
3. Exponential in the size of the query, more precisely NP-complete.

Precisely the complexity of evaluating CQs in plain relational DBs.

Can we go beyond *DL-Lite* and remain in AC^0?

The DLs of the *DL-Lite* family are essentially the maximally expressive DLs enjoying these nice computational properties.

Outline

Query abstraction

Problem

Given an OBDM specification $\mathcal{J} = \langle \mathcal{O}, \mathcal{M}, \mathcal{S} \rangle$ and a query $q_{\mathcal{S}}$ over the sources schema $\mathcal{S}$, compute a query $q_{\mathcal{O}}$ over the ontology $\mathcal{O}$ whose certain answers best characterize $q_{\mathcal{S}}$ under $\mathcal{J}$.

Sort of reverse engineering problem, that is relevant is several tasks, e.g.,

- providing the semantics of the various data sources;
- providing the semantics of data services expressed over the sources;
- checking whether the ontology and the mapping assertions are "adequate" for answering source queries through the OBDM system;
- automatically documenting the semantics of open data sets.

The notion of abstraction (sometimes called realization or reverse rewriting) aims at addressing this problem [Cima 2017, Lutz et al. 2018, Cima et al. 2019, Cima 2020].

Perfect abstraction

Let $\mathcal{J} = \langle \mathcal{O}, \mathcal{M}, \mathcal{S} \rangle$ be an OBDM specification, and let $q_{\mathcal{S}}$ be a query over $\mathcal{S}$. The best characterization is obtained by the co-called perfect abstraction.

Definition (Formal definition of abstraction)

A query $q_{\mathcal{O}}$ over $\mathcal{O}$ is a **perfect** $\mathcal{J}$-abstraction of $q_{\mathcal{S}}$, if for every $\mathcal{S}$-database D such that $\langle \mathcal{J}, D \rangle$ is consistent, we have that

$$q_{\mathcal{S}}^{D} = cert(q_{\mathcal{O}}, \mathcal{J}, D)$$

$q_{\mathcal{S}}^{D} = cert(q_{\mathcal{O}}, \mathcal{J}, D)$	
Query answering Ontology-to-Source rewriting (top-down)	**Query abstraction** Source-to-Ontology rewriting (bottom-up)
input: $q_{\mathcal{O}}$ **output**: $q_{\mathcal{S}}$	**input**: $q_{\mathcal{S}}$ **output**: $q_{\mathcal{O}}$

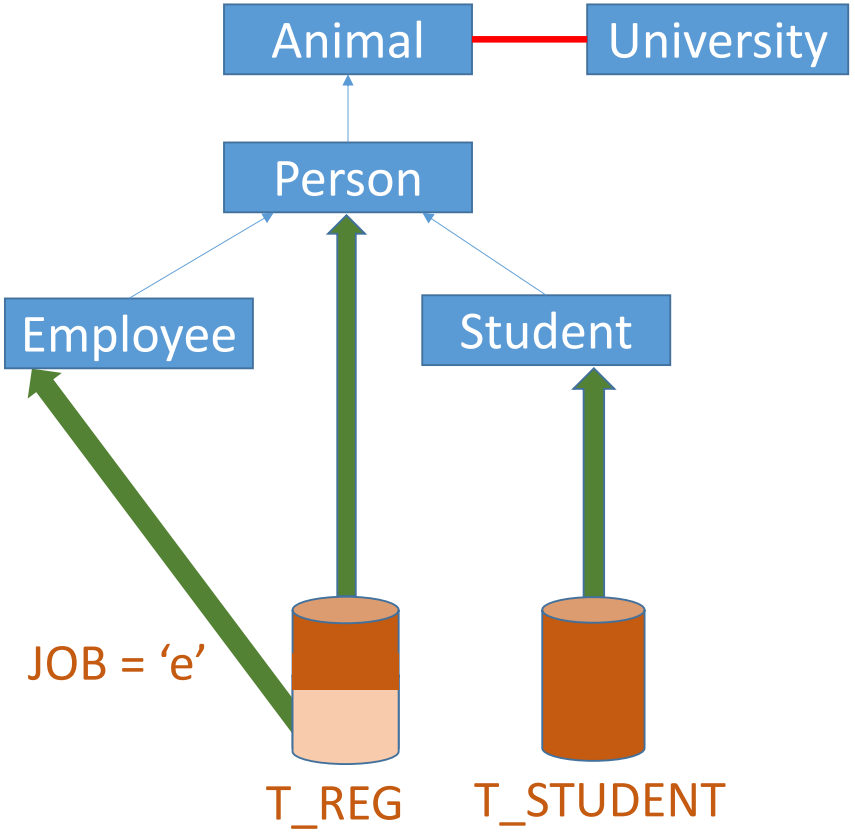
Two basic computational problems:

- Verification (check if $q_{\mathcal{O}}$ is a perfect $\mathcal{J}$-abstraction of $q_{\mathcal{S}}$)
- Computation (compute the perfect $\mathcal{J}$-abstraction of $q_{\mathcal{S}}$)

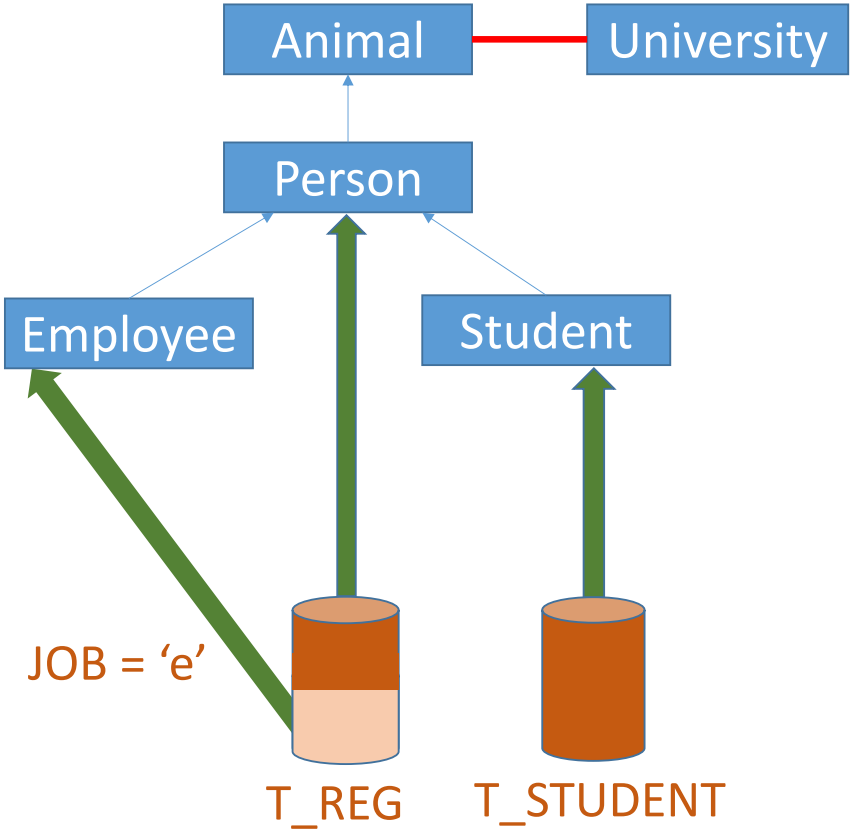
Perfect abstraction - Example

Source query $q_{\mathcal{S}}$:
select ID as x from T_STUDENT

Source query $q'_{\mathcal{S}}$:
select ID as x from T_REG



Perfect abstraction - Example

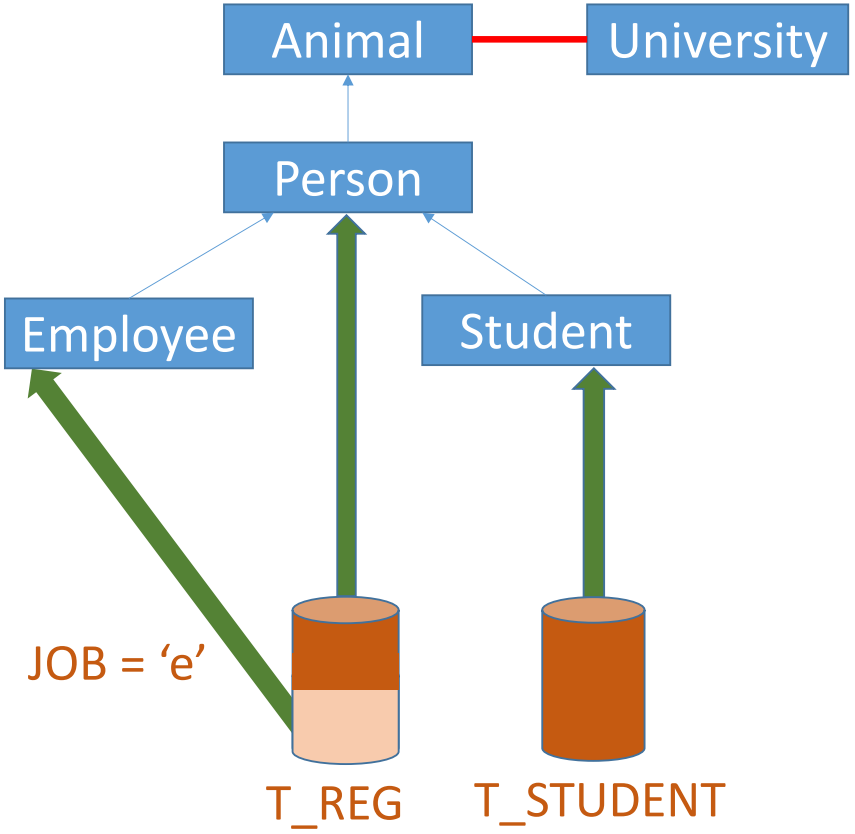


Source query $q_{\mathcal{S}}$:
select ID as x from T_STUDENT

Source query $q'_{\mathcal{S}}$:
select ID as x from T_REG

- perfect $\mathcal{J}$-abstraction of $q_{\mathcal{S}}$: **???**

Perfect abstraction - Example

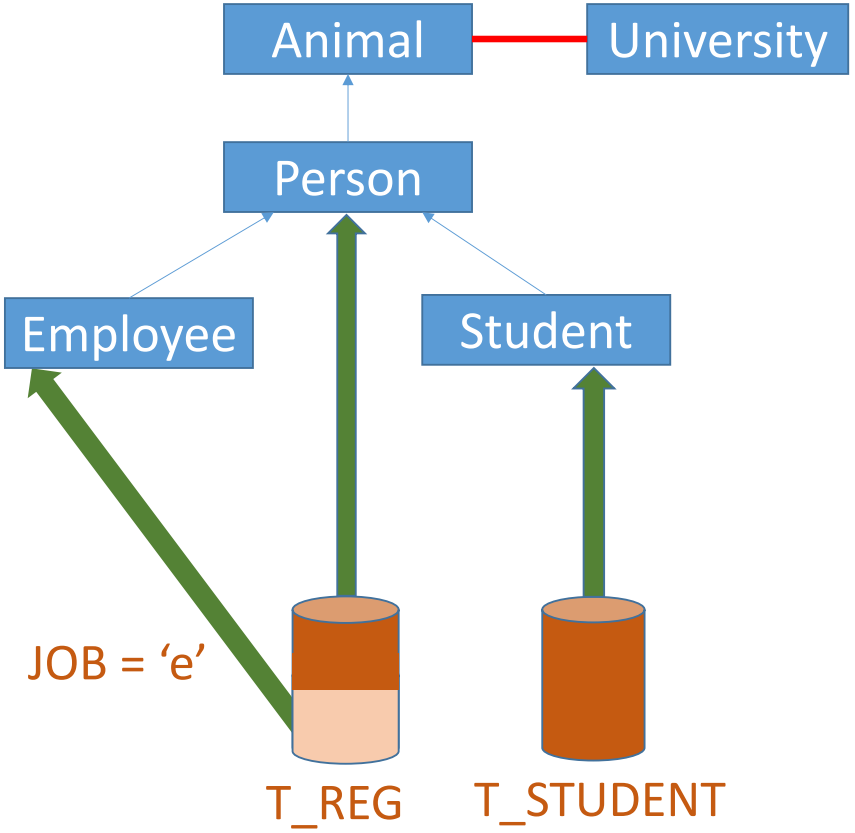


Source query $q_{\mathcal{S}}$:
select ID as x from T_STUDENT

Source query $q'_{\mathcal{S}}$:
select ID as x from T_REG

- perfect $\mathcal{J}$-abstraction of $q_{\mathcal{S}}$: Student(x)

Perfect abstraction - Example

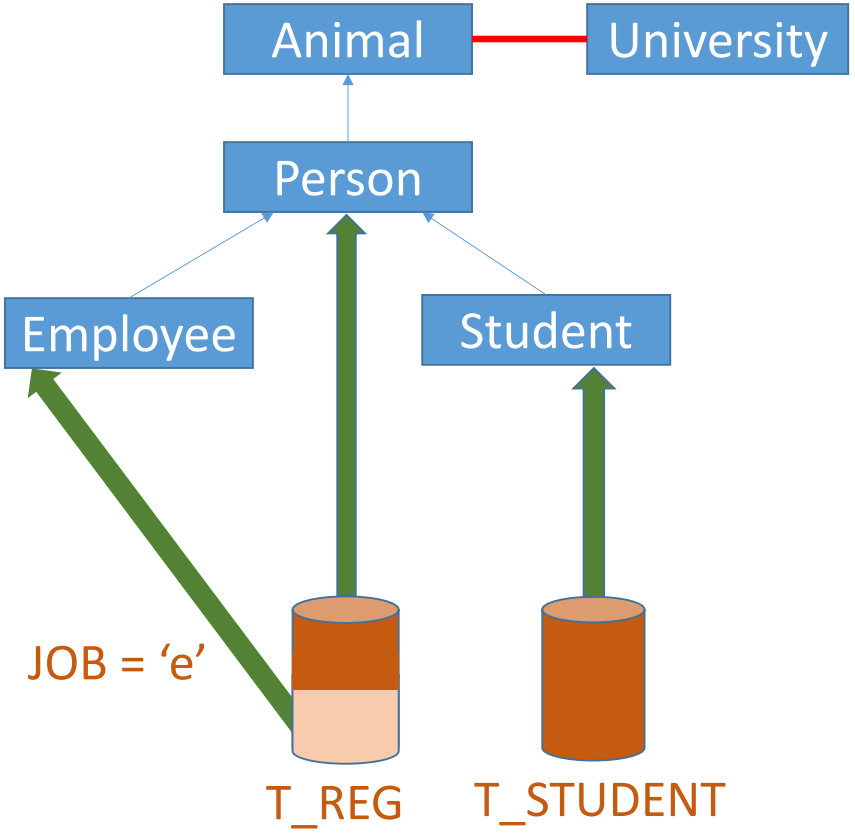


Source query $q_{\mathcal{S}}$:
select ID as x from T_STUDENT

Source query $q'_{\mathcal{S}}$:
select ID as x from T_REG

- perfect $\mathcal{J}$-abstraction of $q_{\mathcal{S}}$: Student(x)
- perfect $\mathcal{J}$-abstraction of $q'_{\mathcal{S}}$: **???**

Perfect abstraction - Example



Source query $q_{\mathcal{S}}$:
select ID as x from T_STUDENT

Source query $q'_{\mathcal{S}}$:
select ID as x from T_REG

- perfect $\mathcal{J}$-abstraction of $q_{\mathcal{S}}$: Student(x)
- perfect $\mathcal{J}$-abstraction of $q'_{\mathcal{S}}$: none

Sound and complete abstractions

The perfect abstraction for a source query may not exist.

Let $\mathcal{J} = \langle \mathcal{O}, \mathcal{M}, \mathcal{S} \rangle$ be an OBDM specification, and let $q_{\mathcal{S}}$ be a query over $\mathcal{S}$.

Definition (Sound abstraction)

A query $q_{\mathcal{O}}$ over $\mathcal{O}$ is a **sound** $\mathcal{J}$-abstraction of $q_{\mathcal{S}}$ if for every $\mathcal{S}$-database D such that $\langle \mathcal{J}, D \rangle$ is consistent, we have that

$$cert(q_{\mathcal{O}}, \mathcal{J}, D) \subseteq q_{\mathcal{S}}^{D}$$

Definition (Complete abstraction)

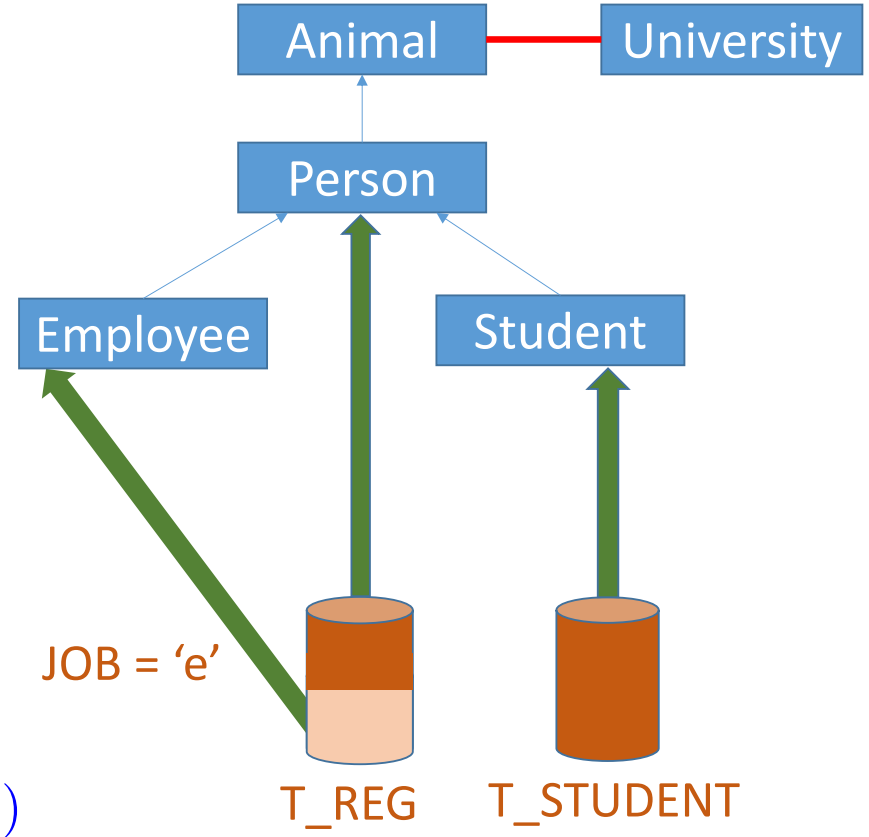
A query $q_{\mathcal{O}}$ over $\mathcal{O}$ is a **complete** $\mathcal{J}$-abstraction of $q_{\mathcal{S}}$ if for every $\mathcal{S}$-database D such that $\langle \mathcal{J}, D \rangle$ is consistent, we have that

$$q_{\mathcal{S}}^{D} \subseteq cert(q_{\mathcal{O}}, \mathcal{J}, D)$$

Sound and complete abstractions - Example

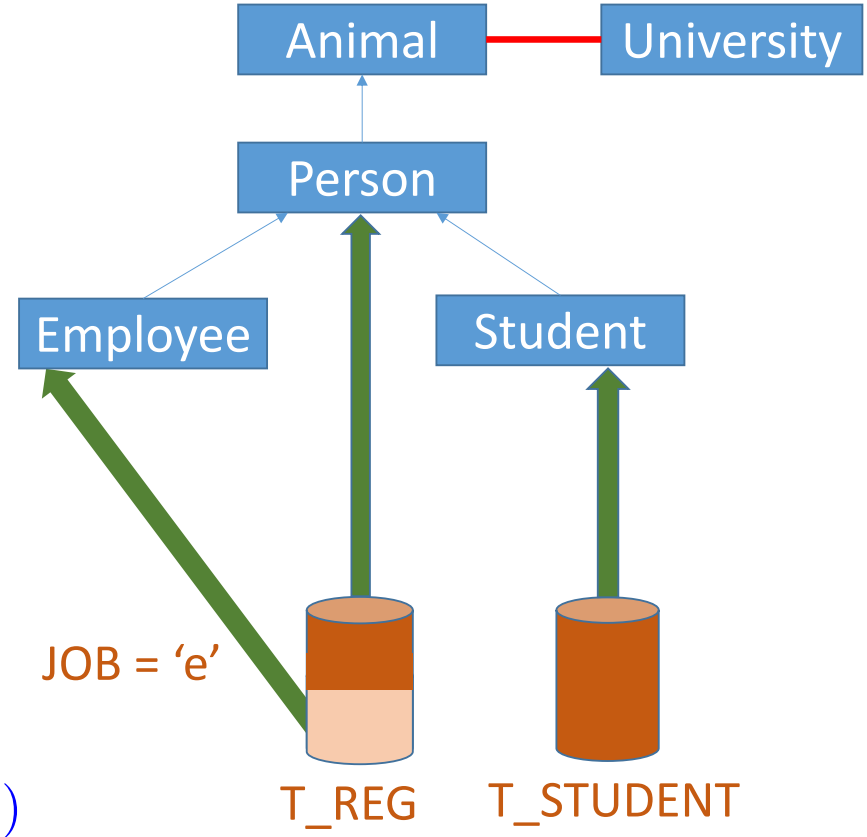
Source query $q_{\mathcal{S}}$:
select ID as x from T_STUDENT

Source query $q'_{\mathcal{S}}$:
select ID as x from T_REG



- perfect $\mathcal{J}$-abstraction of $q_{\mathcal{S}}$: Student(x)
- perfect $\mathcal{J}$-abstraction of $q'_{\mathcal{S}}$: none

Sound and complete abstractions - Example



Source query $q_{\mathcal{S}}$:
select ID as x from T_STUDENT

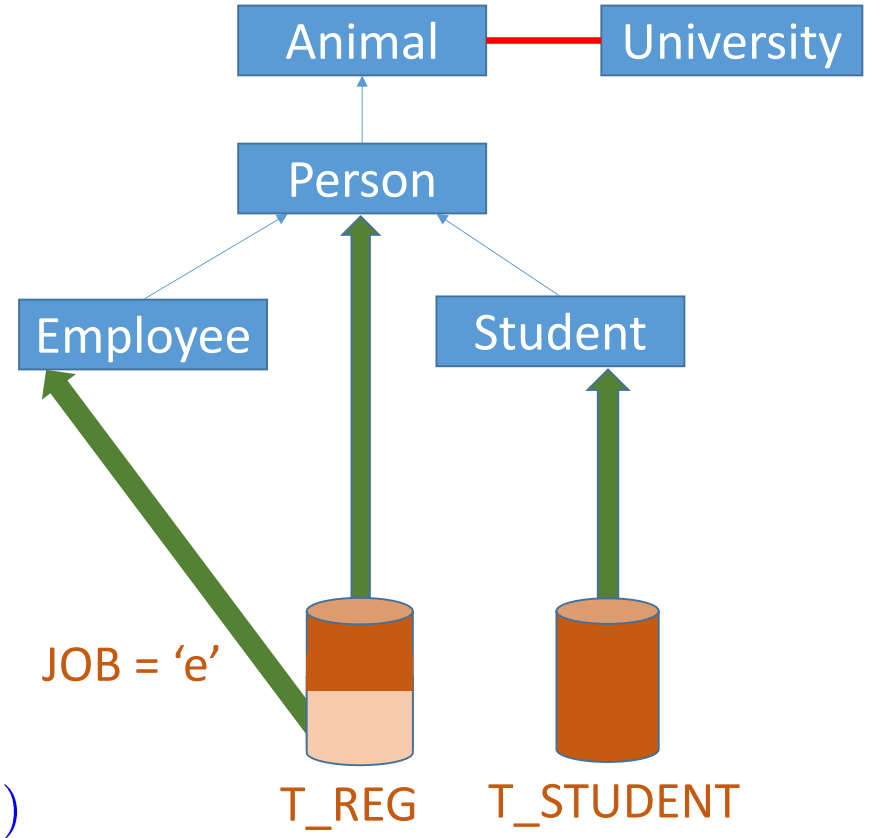
Source query $q'_{\mathcal{S}}$:
select ID as x from T_REG

- perfect $\mathcal{J}$-abstraction of $q_{\mathcal{S}}$: Student(x)
- perfect $\mathcal{J}$-abstraction of $q'_{\mathcal{S}}$: none
- complete $\mathcal{J}$-abstraction of $q'_{\mathcal{S}}$: **???**
- complete $\mathcal{J}$-abstraction of $q'_{\mathcal{S}}$: **???**

Sound and complete abstractions - Example

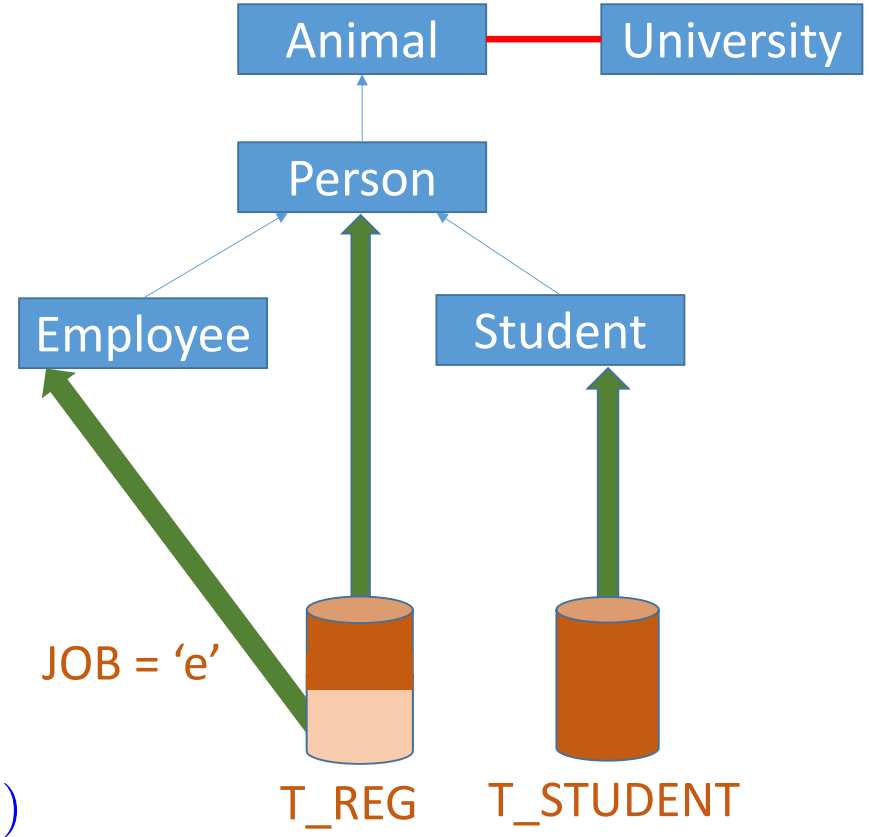
Source query $q_{\mathcal{S}}$:
select ID as x from T_STUDENT

Source query $q'_{\mathcal{S}}$:
select ID as x from T_REG



- perfect $\mathcal{J}$-abstraction of $q_{\mathcal{S}}$: Student(x)
- perfect $\mathcal{J}$-abstraction of $q'_{\mathcal{S}}$: none
- complete $\mathcal{J}$-abstraction of $q'_{\mathcal{S}}$: Animal(x)
- complete $\mathcal{J}$-abstraction of $q'_{\mathcal{S}}$: Person(x)

Sound and complete abstractions - Example

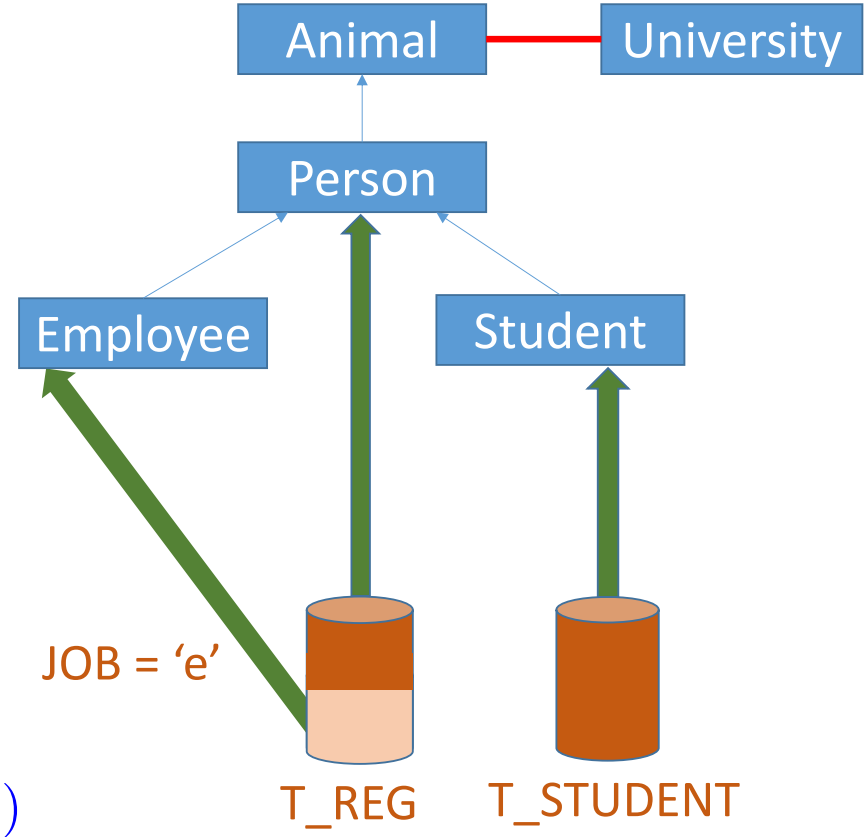


Source query $q_{\mathcal{S}}$:
select ID as x from T_STUDENT

Source query $q'_{\mathcal{S}}$:
select ID as x from T_REG

- perfect $\mathcal{J}$-abstraction of $q_{\mathcal{S}}$: Student(x)
- perfect $\mathcal{J}$-abstraction of $q'_{\mathcal{S}}$: none
- complete $\mathcal{J}$-abstraction of $q'_{\mathcal{S}}$: Animal(x)
- complete $\mathcal{J}$-abstraction of $q'_{\mathcal{S}}$: Person(x)
- sound $\mathcal{J}$-abstraction of $q'_{\mathcal{S}}$: **???**
- sound $\mathcal{J}$-abstraction of $q'_{\mathcal{S}}$: **???**

Sound and complete abstractions - Example



Source query $q_{\mathcal{S}}$:
select ID as x from T_STUDENT

Source query $q'_{\mathcal{S}}$:
select ID as x from T_REG

- perfect $\mathcal{J}$-abstraction of $q_{\mathcal{S}}$: Student(x)
- perfect $\mathcal{J}$-abstraction of $q'_{\mathcal{S}}$: none
- complete $\mathcal{J}$-abstraction of $q'_{\mathcal{S}}$: Animal(x)
- complete $\mathcal{J}$-abstraction of $q'_{\mathcal{S}}$: Person(x)
- sound $\mathcal{J}$-abstraction of $q'_{\mathcal{S}}$: Person(x), University(x)
- sound $\mathcal{J}$-abstraction of $q'_{\mathcal{S}}$: Employee(x)

Maximally sound and minimally complete abstractions

Let $\mathcal{L}$ be a class of queries.

Definition

If $q_{\mathcal{O}} \in \mathcal{L}$ is a sound $\mathcal{J}$-abstraction of $q_{\mathcal{S}}$, then $q_{\mathcal{O}}$ is **$\mathcal{L}$-maximally sound** if no $q' \in \mathcal{L}$ exists such that

(i) q' is a sound $\mathcal{J}$-abstraction of $q_{\mathcal{S}}$,

(ii) $\forall$ $\mathcal{S}$-database D $cert(q_{\mathcal{O}}, \mathcal{J}, D) \subseteq cert(q', \mathcal{J}, D)$, and

(iii) there exists an $\mathcal{S}$-database D s.t. $cert(q_{\mathcal{O}}, \mathcal{J}, D) \subset cert(q', \mathcal{J}, D)$.

Definition

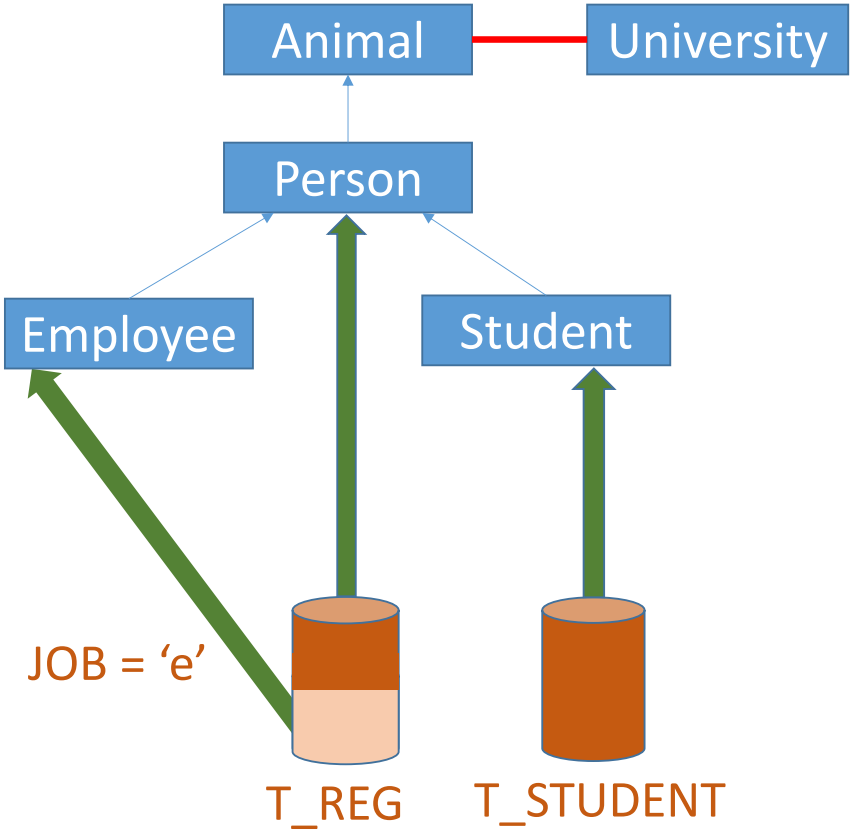
If $q_{\mathcal{O}} \in \mathcal{L}$ is a complete $\mathcal{J}$-abstraction of $q_{\mathcal{S}}$, then $q_{\mathcal{O}}$ is **$\mathcal{L}$-minimally complete** if no $q' \in \mathcal{L}$ exists such that

(i) q' is a complete $\mathcal{J}$-abstraction of $q_{\mathcal{S}}$,

(ii) $\forall$ $\mathcal{S}$-database D $cert(q', \mathcal{J}, D) \subseteq cert(q_{\mathcal{O}}, \mathcal{J}, D)$, and

(iii) there exists an $\mathcal{S}$-database s.t. $cert(q', \mathcal{J}, D) \subset cert(q_{\mathcal{O}}, \mathcal{J}, D)$.

Maximal and minimal abstractions - Example

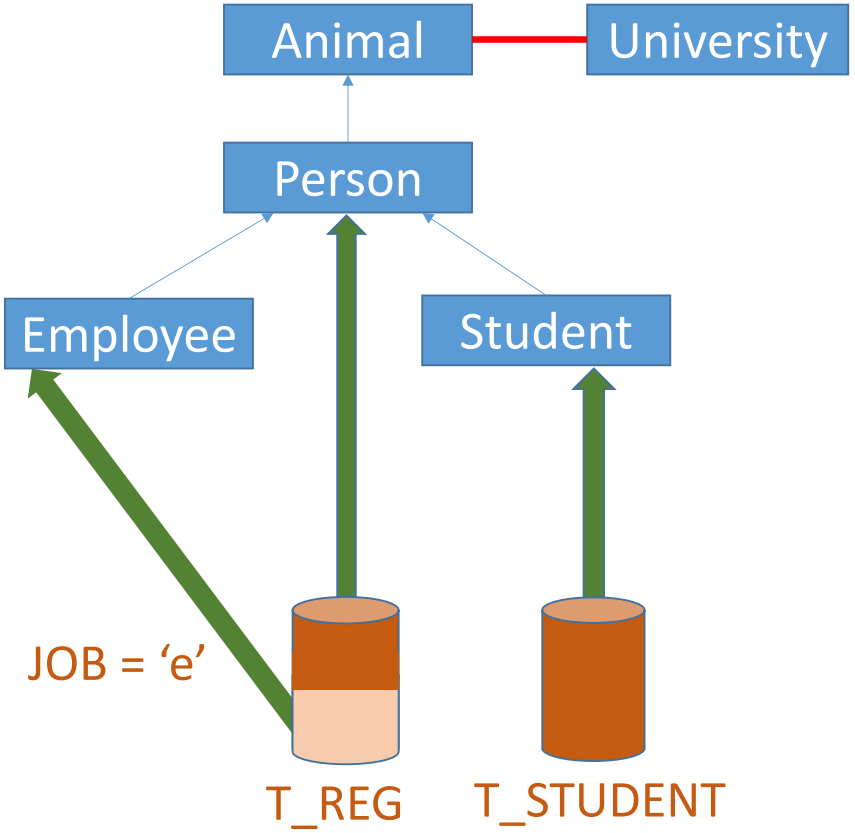


Source query $q_{\mathcal{S}}$:
select ID as x from T_STUDENT

Source query $q'_{\mathcal{S}}$:
select ID as x from T_REG

- perfect $\mathcal{J}$-abstraction of $q_{\mathcal{S}}$: Student(x)
- perfect $\mathcal{J}$-abstraction of $q'_{\mathcal{S}}$: none
- UCQ-minimally complete $\mathcal{J}$-abstraction of $q'_{\mathcal{S}}$: **???**
- UCQ-maximally sound $\mathcal{J}$-abstraction of $q'_{\mathcal{S}}$: **???**

Maximal and minimal abstractions - Example



Source query $q_{\mathcal{S}}$:
select ID as x from T_STUDENT

Source query $q'_{\mathcal{S}}$:
select ID as x from T_REG

- perfect $\mathcal{J}$-abstraction of $q_{\mathcal{S}}$: Student(x)
- perfect $\mathcal{J}$-abstraction of $q'_{\mathcal{S}}$: none
- UCQ-minimally complete $\mathcal{J}$-abstraction of $q'_{\mathcal{S}}$: Person(x)
- UCQ-maximally sound $\mathcal{J}$-abstraction of $q'_{\mathcal{S}}$: Employee(x)

Can we compute the UCQ-maximally complete abstractions?

Theorem

The verification problem for complete abstractions is NP-complete.

What about the computation problem? We now focus on the problem of computing the **UCQ-maximally complete abstraction** of a $q_{\mathcal{S}}$ over the source schema $\mathcal{S}$ with respect to an OBDM specification $\mathcal{J} = \langle \mathcal{O}, \mathcal{M}, \mathcal{S} \rangle$, where

- $q_{\mathcal{S}}$ is a **CQ**
- $\mathcal{O}$ a **DL-Lite**$_{\mathcal{R}}$ (therefore, no functionality assertions) ontology, and
- $\mathcal{M}$ a set of **GLAV** mapping assertions of the form

$$\forall \vec{x}\ \forall \vec{y}\ (\Phi_{\mathcal{S}}(\vec{x}, \vec{y}) \ \rightarrow\ \exists \vec{z}\ \Psi_{\mathcal{O}}(\vec{x}, \vec{z}))$$

  where $\Phi_{\mathcal{S}}(\vec{x})$ is a CQ over $\mathcal{S}$ and $\Psi_{\mathcal{O}}$ is a CQ over the ontology $\mathcal{O}$.

Note that computing such rewriting solves also the problem of computing the perfect abstraction (it is sufficient to check the result for soundness)

How to compute it: basic idea

There is a strict correlation between CQs and relational databases. Given a CQ q over a schema $\mathcal{S}$ it is possible to construct in linear time an $\mathcal{S}$-database D_q that fully captures q, and **viceversa**:

- every constant in q becomes a value in D_q;
- every variable in q becomes a *labeled null* in D_q;
- every atom $R_i(\vec{u}) \in q$ becomes a fact (tuple) in D_q.

Example

Let $\mathcal{S}$ contain the tables **Tab1(id,city)** and **Tab2(id,city)**, and consider the following CQ $q_{\mathcal{S}}$ over $\mathcal{S}$

$$q_{\mathcal{S}}(x) \leftarrow \mathsf{Tab1}(x, y), \mathsf{Tab2}(\text{'sara'}, y)$$

The $\mathcal{S}$-database $D_{q_{\mathcal{S}}}$ associated to the query $q_{\mathcal{S}}$ is:

$$\mathsf{Tab1}(\mathbf{x},\mathbf{y}),\ \mathsf{Tab2}(\mathbf{'sara'},\mathbf{y})$$

where $\mathbf{x}, \mathbf{y}$ are labeled nulls.

How to compute it: basic idea (cont'd)

Roughly speaking, the $\mathcal{S}$-database $D_{q_\mathcal{S}}$ associated to $q_\mathcal{S}$ is representative of those instances of $\mathcal{S}$ on which the evaluation of $q_\mathcal{S}$ is non-empty.

Given the OBDM specification $\mathcal{J} = \langle \mathcal{O}, \mathcal{M}, \mathcal{S} \rangle$, we denote by $\mathcal{M}(D_{q_\mathcal{S}})$ the set of $\mathcal{O}$-facts (ABox, in DL terminology) obtained by chasing $D_{q_\mathcal{S}}$ with the mapping assertions in $\mathcal{M}$.

Intuition

The query $q_\mathcal{O}$ over the ontology corresponding to the UCQ-minimally complete $\mathcal{J}$-abstraction of $q_\mathcal{S}$ is the query corresponding to $\mathcal{M}(D_{q_\mathcal{S}})$.

$$q_\mathcal{S} \xrightarrow{\text{compute the associated DB}} D_{q_\mathcal{S}} \xrightarrow{\text{compute the chase}} \mathcal{M}(D_{q_\mathcal{S}}) \xrightarrow{\text{compute the corresponding query}} q_\mathcal{O}$$

How to compute it: basic idea (cont'd)

Example

Suppose to have the following mapping:

m_1: Tab1(x,y) $\rightsquigarrow$ Person(x), livesIn(x, y)
m_2: Tab2(x,y) $\rightsquigarrow$ Person(x), worksIn(x, y)

and consider the instance $D_{q_\mathcal{S}}$ of $\mathcal{S}$ associated to the query $q_\mathcal{S}$:

Tab1(**x**,**y**), Tab2(**'sara'**,**y**),

It is easy to see that $\mathcal{M}(D_{q_\mathcal{S}})$ is the **ABox** containing the following assertions:

Person(**x**), livesIn(**x**, **y**), Person(**'sara'**), worksIn(**'sara'**, **y**)

From $\mathcal{M}(D_{q_\mathcal{S}})$ we can construct the following query over the ontology, denoting *all the Persons living in the same city where the person 'sara' works*

$q_\mathcal{O}(x) \leftarrow$ Person(x), livesIn(x, y), Person('sara'), worksIn('sara', y)

UCQ-minimally complete abstraction: algorithm

We denote by $\top$ the special conjunctive query over $\mathcal{O}$ that is true for every tuple.

Algorithm FindMinimallyCompleteAbstraction

Input: a *DL-Lite*$_{\mathcal{R}}$ OBDM specification $\langle \mathcal{O}, \mathcal{M}, \mathcal{S} \rangle$,
UCQ $q_{\mathcal{S}} = q^1_{\mathcal{S}}(\vec{t_1}) \cup \ldots \cup q^n_{\mathcal{S}}(\vec{t_n})$ over $\mathcal{S}$
Output: UCQ $q_{\mathcal{O}}$ over $\mathcal{O}$
begin

1. $q_{\mathcal{O}} := \{ \vec{t_1} \mid \mathcal{M}(q^1_{\mathcal{S}}), \top(\vec{t_1}) \} \cup \ldots \cup \{ \vec{t_n} \mid \mathcal{M}(q^n_{\mathcal{S}}), \top(\vec{t_n}) \}$
2. **return** $q_{\mathcal{O}}$

end

Informally, the algorithm computes the output query as union of CQs obtained by simply applying the mapping $\mathcal{M}$ to each CQ $q^i_{\mathcal{S}}$ in $q_{\mathcal{S}}$, using $\top$ to bind the distinguished variables of the output query that do not appear in $\mathcal{M}(q^i_{\mathcal{S}})$.

UCQ-minimally complete abstraction: properties

Let $\mathcal{J} = \langle \mathcal{O}, \mathcal{M}, \mathcal{S} \rangle$ be a *DL-Lite*$_{\mathcal{R}}$ OBDM specification, and let $q_{\mathcal{S}}$ be a CQ over $\mathcal{S}$.

- **FindMinimallyCompleteAbstraction**$(\mathcal{J}, q_{\mathcal{S}})$ terminates and runs in:
  - (i) PTIME in the size of $q_{\mathcal{S}}$;
  - (ii) PTIME in the size of $\mathcal{O}$;
  - (iii) EXPTIME in the size of $\mathcal{M}$.
- The query returned by **FindMinimallyCompleteAbstraction**$(\mathcal{J}, q_{\mathcal{S}})$ is a **UCQ-minimally complete $\mathcal{J}$-abstraction** of $q_{\mathcal{S}}$.
- The UCQ-minimally complete $\mathcal{J}$-abstraction of $q_{\mathcal{S}}$ is **unique** up to logical equivalence, and it can be expressed as a CQ if $q_{\mathcal{S}}$ is a CQ.
- A perfect $\mathcal{J}$-abstraction of $q_{\mathcal{S}}$ exists in the class of positive queries if and only if the query $q_{\mathcal{O}}$ returned by **FindMinimallyCompleteAbstraction**$(\mathcal{J}, q_{\mathcal{S}})$ is a sound $\mathcal{J}$-abstraction of $q_{\mathcal{S}}$, in which case $q_{\mathcal{O}}$ is the perfect $\mathcal{J}$-abstraction of $q_{\mathcal{S}}$.

UCQ-maximally sound abstraction

Computing UCQ-maximally sound $\mathcal{J}$-abstractions is more involved.

Theorem

The verification problem for sound abstractions is Π_2^p-complete.

Theorem

UCQ-maximally sound $\mathcal{J}$-abstractions of a query $q_{\mathcal{S}}$ may not exist if the OBDM specification $\mathcal{J}$ has one of the following features:

1. disjointness axioms in the ontology;
2. inclusion axioms with $\exists R$ as right-hand side in the ontology;
3. LAV mapping assertions, even without joins involving existential variables in the right-hand side;
4. non-pure GAV mapping assertions;
5. $q_{\mathcal{S}}$ has joins on existential variables.

We have devised an algorithm for computing UCQ-maximally sound $\mathcal{J}$-abstractions in the restricted setting with none of the above features.

UCQ-maximally sound abstraction with non-pure GAV mappings

A GAV mapping is pure if for each mapping assertion, the variables in the right-hand-side query are different. Consider the following OBDA specification $\mathcal{J} = \langle \mathcal{O}, \mathcal{S}, \mathcal{M} \rangle$ with non-pure GAV mapping:

- $\mathcal{O} = \emptyset$,
- $\mathcal{S} = \{ s_1, s_2 \}$,
- $\mathcal{M} = \{ s_1(x_1, x_2) \rightarrow R(x_1, x_2), \quad s_2(x) \rightarrow R(x, x) \}$,

and the following $q_{\mathcal{S}} = \{(x_1, x_2) \mid s_1(x_1, x_2)\}$.

$q'_{\mathcal{O}} = \{(x_1, x_2) \mid R(x_1, x_2)\}$ is the UCQ-minimally complete $\mathcal{J}$-rewriting of $q_{\mathcal{S}}$, but is not sound. Intuitively, the infinite query:

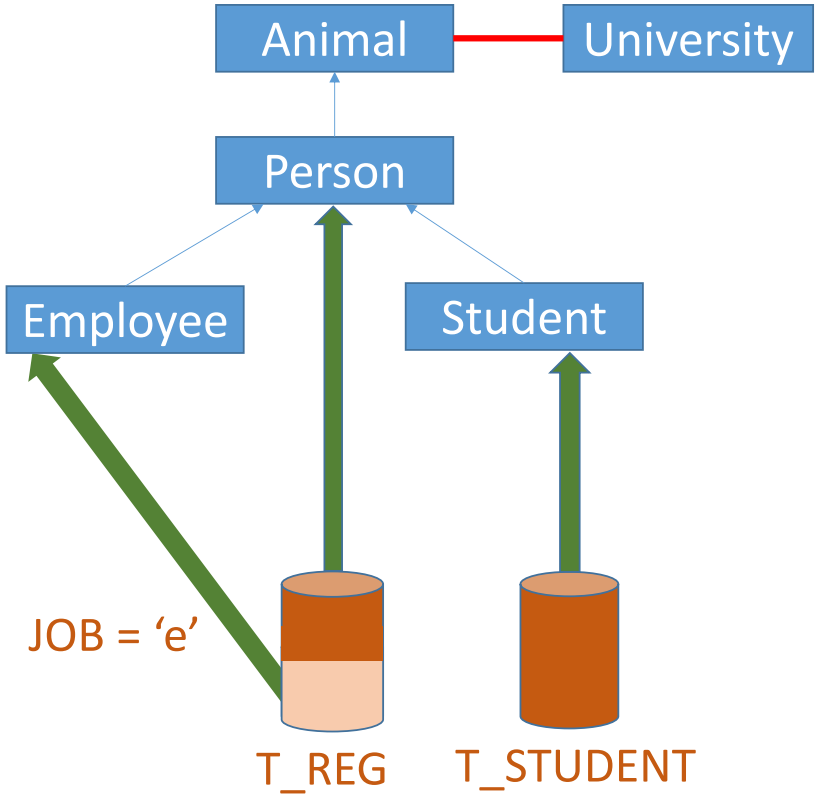
$$q_{\mathcal{O}} = \bigcup_{\substack{a,b \in const \\ \text{with } a \neq b}} \{(a, b) \mid R(a, b)\}$$

is the maximally sound $\mathcal{J}$-abstraction of $q_{\mathcal{S}}$ in the class of positive queries, but is not a UCQ $\rightarrow$ no UCQ-maximally sound $\mathcal{J}$-abstraction of $q_{\mathcal{O}}$ exists.

Outline

Future work on query abstraction

A notable challenge: can we obtain better sound abstractions if we express abstractions with languages going beyond UCQ?



Source query $q'_{\mathcal{S}}$:
select ID as x from T_REG

- UCQ-maximally sound $\mathcal{J}$-abstraction of $q'_{\mathcal{S}}$: Employee(x)
- a better sound $\mathcal{J}$-abstraction of $q'_{\mathcal{S}}$: Person(x), $\neg$ **K**Student(x), corresponding to the non-monotonic query in EQL-lite asking for all persons that are not known to be students.

Future work on query abstraction

More challenges:

- The existence problem.
- Investigate ontology query languages beyond UCQ (such as EQL-lite)
- Single out more restricted settings guaranteeing existence of UCQ-maximally sound abstractions.
- Going beyond *DL-Lite*$_{\mathcal{R}}$.
- We have assumed that the query language used to express $q_{\mathcal{S}}$ is the language of CQs. This is too limited: real applications often requires aggregation, negation, universal quantification, etc. Can we compute abstractions of queries with such operators?
- Study specific applications of abstraction, and try to specialize the corresponding techniques.
- Investigate possibile use of query abstraction in explanation of classifiers

Research themes in HOPE, a national MIUR-PRIN project with Sapienza University of Rome, Free University of Bozen/Bolzano, Polytechnic University of Milan, University of Milan, and University of Cagliari.

SAPIENZA UNIVERSITÀ DI ROMA

Thanks

Joint work with(*)

- Diego Calvanese, Giuseppe De Giacomo, Domenico Lembo, Antonella Poggi, Riccardo Rosati, *and many others* (*query answering*)
- *Gianluca Cima, Antonella Poggi* (*query abstraction*)

Thank you for your attention!

(*) *My heart is not weary, it's light and it's free; I've got nothin' but affection for all those who've sailed with me – Bob Dylan*