TECHNISCHE UNIVERSITÄT DRESDEN

International Center for Computational Logic

Computational Logic ∴ Group

Sebastian Rudolph
International Center for Computational Logic
TU Dresden

Existential Rules – Lecture 1

Adapted from slides by Andreas Pieris and Michaël Thomazo
Summer Term 2024

Lecture Information

- Lecturer: Sebastian Rudolph

  sebastian.rudolph@tu-dresden.de
- Lecture: Mondays, 11:10 – 12:40, APB0005
- Tutorial: Mondays, 9:00 – 10:50, APB0005
- Dates, slides, and other info on course webpage:

  https://iccl.inf.tu-dresden.de/web/Introduction_to_Existential_Rules_(SS2024)
- Oral examintation

Learning Outcomes and Prerequisites

Outcomes: A good understanding of

- The fundamentals of ontology-based query answering
- The complexity of the problem and the main techniques
- Possible research directions

Preferable prerequisites: Basic knowledge of

- First-order logic (syntax and semantics)
- Databases (relational model)
- Complexity theory (complexity classes, reductions)

Today

- Ontology-based Data Access
- Ontology-based Query Answering
- Ontology and Query Languages

Accessing Big Data: A New Challenge

"Data is stored in various heterogeneous formats over many differently structured databases. As a result, the gathering of only relevant data spread over disparate sources becomes a very time consuming task"

Jim Crompton, W3C Workshop on Semantic Web in Oil & Gas Industry, 2008

Accessing Big Data: A New Challenge

Experts in geology and geophysics develop stratigraphic models of unexplored areas on the basis of data acquired from previous operations at nearby geographical locations

Statoil

Facts:

- 1000 TB of relational data
- Using diverse schemata
- Spread over 2000 tables, over multiple individual data bases

Data Access for Exploration:

- 900 experts in Statoil Exploration
- Up to 4 days for new data access queries - assistance from IT experts
- 30 - 70% of time spent on data gathering

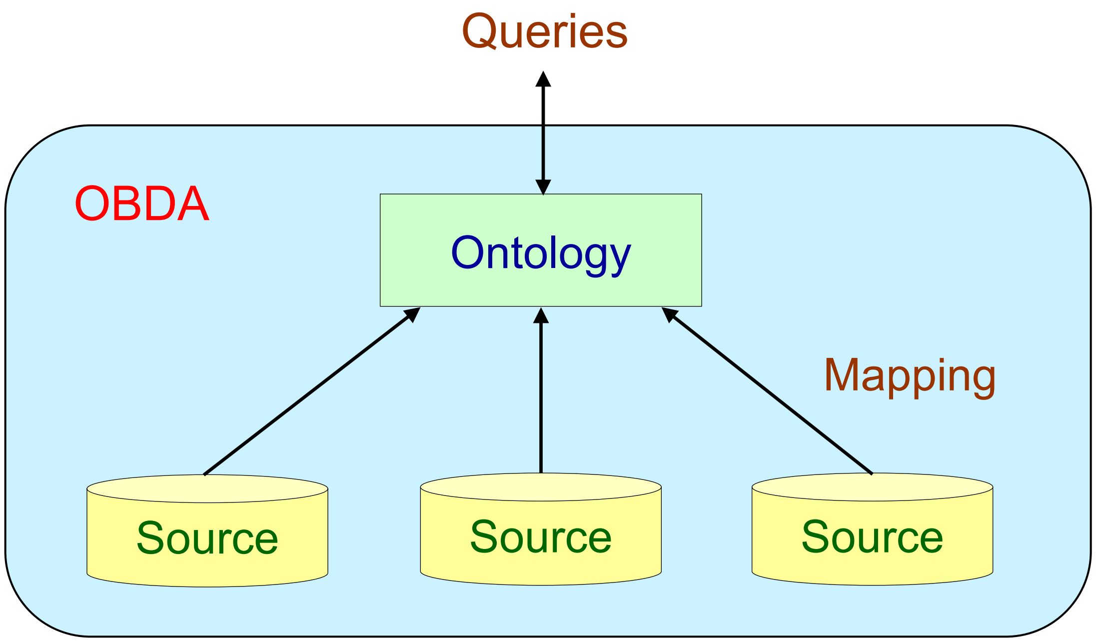
based on slides by Diego Calvanese

Ontology-Based Data Access (OBDA)

- Achieve transparency in accessing data using… logic
- Manage data by exploiting Knowledge Representation techniques

- Key principles underlying OBDA:
  - Conceptual, high level representation of the domain of interest in terms of an ontology (i.e., a logical theory)
  - Map the ontology to the data sources - do not migrate the data
  - Specify all information requests to the data in terms of the ontology

Ontology-Based Data Access: Architecture



- Ontology: provides a unified conceptual "global view" of the data
- Data Sources: external and independent (possibly multiple and heterogeneous)
- Mapping: semantically link data at the sources with the ontology

Ontology-Based Data Access: Formalization

Syntax: An OBDA system is a triple $\mathcal{O} = \langle \Sigma, D, M \rangle$, where:

- Σ is an ontology expressed in some logical language (first-order logic)
- D is a (federated) relational database representing the sources
- M is a set of mappings of the form $Q_D(\mathbf{X}) \subseteq Q_\Sigma(\mathbf{X})$

Q_D: first-order query over D — Q_Σ: first-order query over Σ

Semantics: We assign to $\mathcal{O}$ a first-order logic semantics – an instance is a model of $\mathcal{O}$ if it satisfies Σ and M w.r.t. D

Ontology-Based Data Access: Example

Ontology Σ - high level representation of the domain of interest

$\forall X\ (\mathit{Researcher}(X) \rightarrow \exists Y\ (\mathit{worksFor}(X,Y) \wedge \mathit{Project}(Y)))$

$\forall X\ (\mathit{Project}(X) \rightarrow \exists Y\ (\mathit{worksFor}(Y,X) \wedge \mathit{Researcher}(Y)))$

$\forall X \forall Y\ (\mathit{worksFor}(X,Y) \rightarrow \mathit{Researcher}(X) \wedge \mathit{Project}(Y))$

$\forall X\ (\mathit{Project}(X) \rightarrow \exists Y\ (\mathit{PrName}(X,Y)))$

Ontology-Based Data Access: Example

Relational database *D* - a single database that represents the sources

worksIn	SSN	Name
	100	AAA
	200	BBB
	300	CCC

Ontology-Based Data Access: Example

Relational database *D* - a single database that represents the sources

worksIn	SSN	Name
	100	AAA
	200	BBB
	300	CCC

the researcher with SSN 100 works for the project with name “AAA”

Ontology-Based Data Access: Example

Mapping *M* - semantically link data at the sources with the ontology

SELECT SSN, Name
FROM *worksIn*

$\subseteq$

Researcher(person(SSN)) $\wedge$
Project(proj(Name)) $\wedge$
worksFor(person(SSN), proj(Name)) $\wedge$
PrName(proj(Name), Name)

Ontology-Based Data Access: Example

Mapping *M* - semantically link data at the sources with the ontology

SELECT SSN, Name FROM *worksIn*	⊆	*Researcher*(person(SSN)) ∧ *Project*(proj(Name)) ∧ *worksFor*(person(SSN), proj(Name)) ∧ *PrName*(proj(Name), Name)

- Constructors to create objects of the ontology from tuples of values in the database - solution to the impedance mismatch problem
- The constructors are simply Skolem functions

Ontology-Based Data Access: Example

An instance is a model of $\mathcal{O}$ if it satisfies Σ and M w.r.t. D

Ontology-Based Data Access: Example

An instance is a model of $\mathcal{O}$ if it satisfies Σ and *M* w.r.t. *D*

worksIn	SSN	Name
	100	AAA
	200	BBB
	300	CCC

SELECT SSN, Name FROM *worksIn*	⊆	*Researcher*(person(SSN)) ∧ *Project*(proj(Name)) ∧ *worksFor*(person(SSN), proj(Name)) ∧ *PrName*(proj(Name), Name)

Researcher(person(100)), *Project*(proj(AAA)), *worksFor*(person(100), proj(AAA)),
PrName(proj(AAA), AAA),

Ontology-Based Data Access: Example

An instance is a model of $\mathcal{O}$ if it satisfies Σ and *M* w.r.t. *D*

worksIn	SSN	Name
	100	AAA
	200	BBB
	300	CCC

SELECT SSN, Name FROM *worksIn*	⊆	*Researcher*(person(SSN)) ∧ *Project*(proj(Name)) ∧ *worksFor*(person(SSN), proj(Name)) ∧ *PrName*(proj(Name), Name)

Researcher(person(100)), *Project*(proj(AAA)), *worksFor*(person(100), proj(AAA)),
PrName(proj(AAA), AAA),
Researcher(person(200)), *Project*(proj(BBB)), *worksFor*(person(200), proj(BBB)),
PrName(proj(BBB), BBB),

Ontology-Based Data Access: Example

An instance is a model of $\mathcal{O}$ if it satisfies Σ and *M* w.r.t. *D*

worksIn	SSN	Name
	100	AAA
	200	BBB
	300	CCC

SELECT SSN, Name
FROM *worksIn*

⊆

Researcher(person(SSN)) ∧
Project(proj(Name)) ∧
worksFor(person(SSN), proj(Name)) ∧
PrName(proj(Name), Name)

Researcher(person(100)), *Project*(proj(AAA)), *worksFor*(person(100), proj(AAA)),
PrName(proj(AAA), AAA),
Researcher(person(200)), *Project*(proj(BBB)), *worksFor*(person(200), proj(BBB)),
PrName(proj(BBB), BBB),
Researcher(person(300)), *Project*(proj(CCC)), *worksFor*(person(300), proj(CCC)),
PrName(proj(CCC), CCC)

Ontology-Based Data Access: Example

An instance is a model of $\mathcal{O}$ if it satisfies Σ and *M* w.r.t. *D*

$$\forall X\ (\mathit{Researcher}(X) \rightarrow \exists Y\ (\mathit{worksFor}(X,Y) \wedge \mathit{Project}(Y)))$$

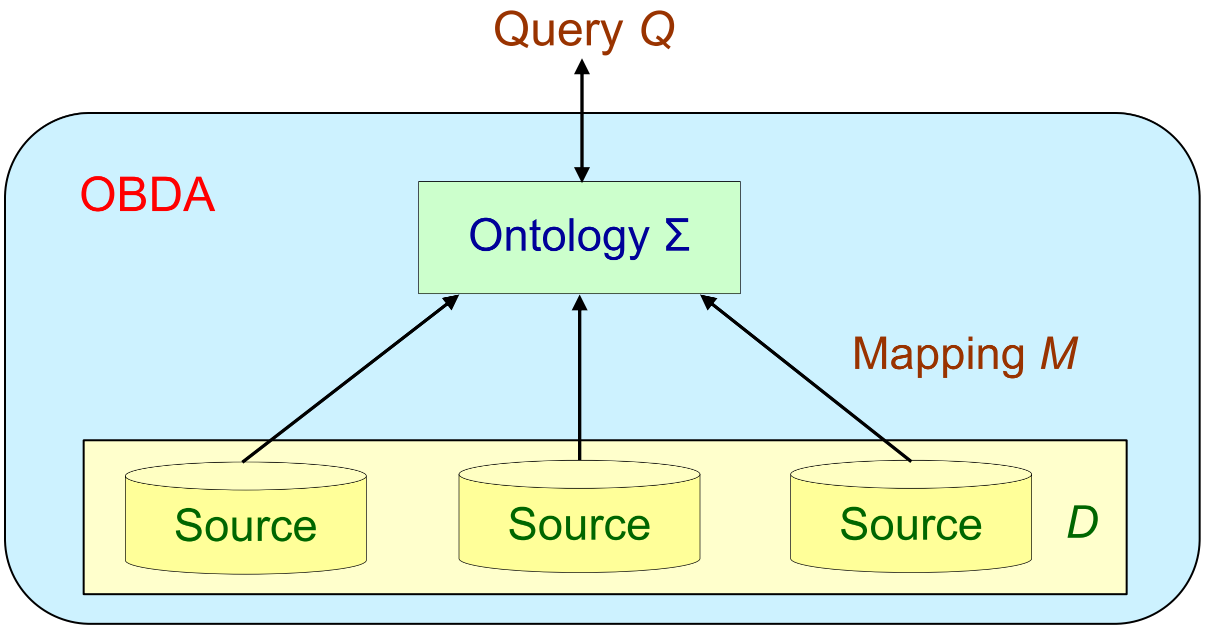
$$\forall X\ (\mathit{Project}(X) \rightarrow \exists Y\ (\mathit{worksFor}(Y,X) \wedge \mathit{Researcher}(Y)))$$

$$\forall X \forall Y\ (\mathit{worksFor}(X,Y) \rightarrow \mathit{Researcher}(X) \wedge \mathit{Project}(Y))$$

$$\forall X\ (\mathit{Project}(X) \rightarrow \exists Y\ (\mathit{PrName}(X,Y)))$$

Researcher(person(100)), *Project*(proj(AAA)), *worksFor*(person(100), proj(AAA)),
PrName(proj(AAA), AAA),
Researcher(person(200)), *Project*(proj(BBB)), *worksFor*(person(200), proj(BBB)),
PrName(proj(BBB), BBB),
Researcher(person(300)), *Project*(proj(CCC)), *worksFor*(person(300), proj(CCC)),
PrName(proj(CCC), CCC)

Query Answering in OBDA

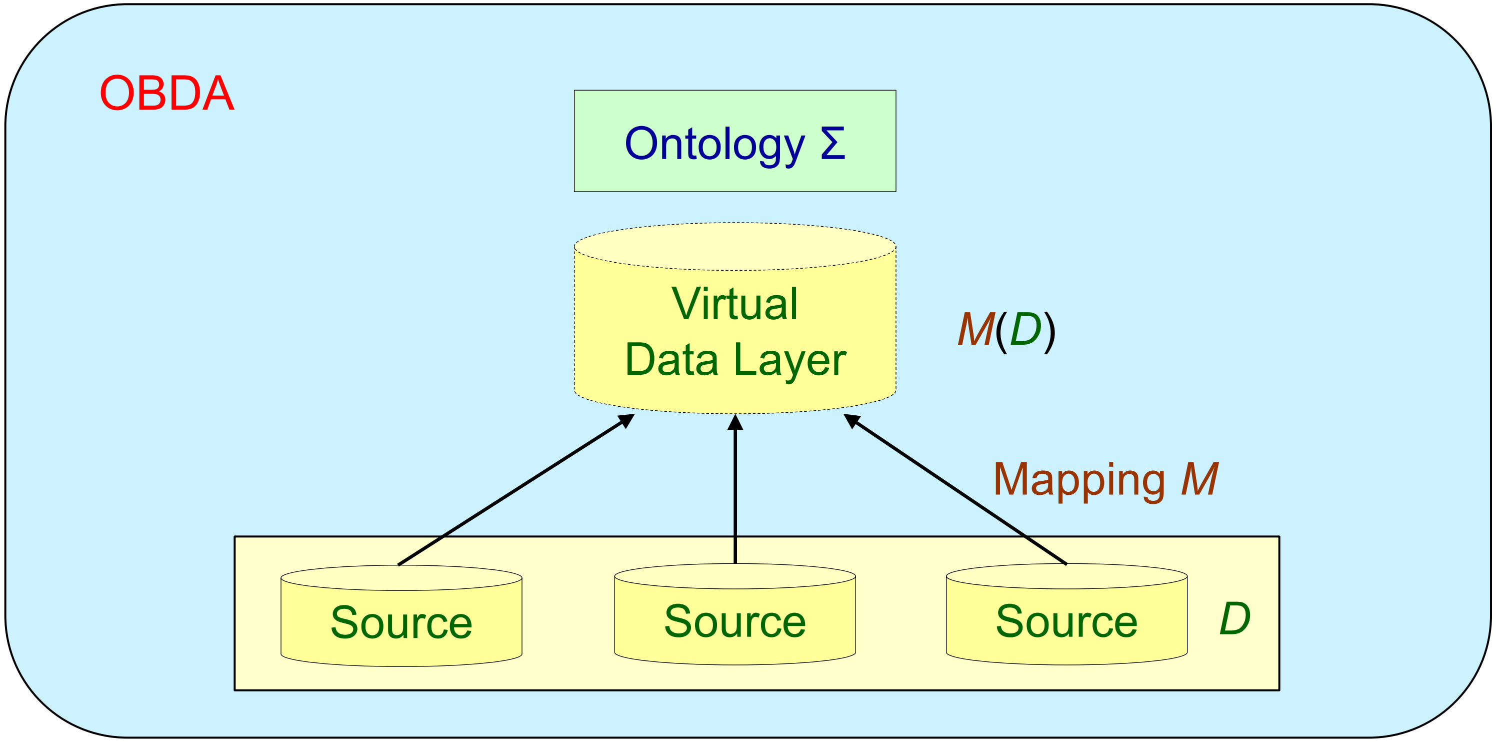


$\mathcal{O} = \langle \Sigma, D, M \rangle$

- We adopt the certain answer semantics
- Find those answers that hold in all models of the OBDA system

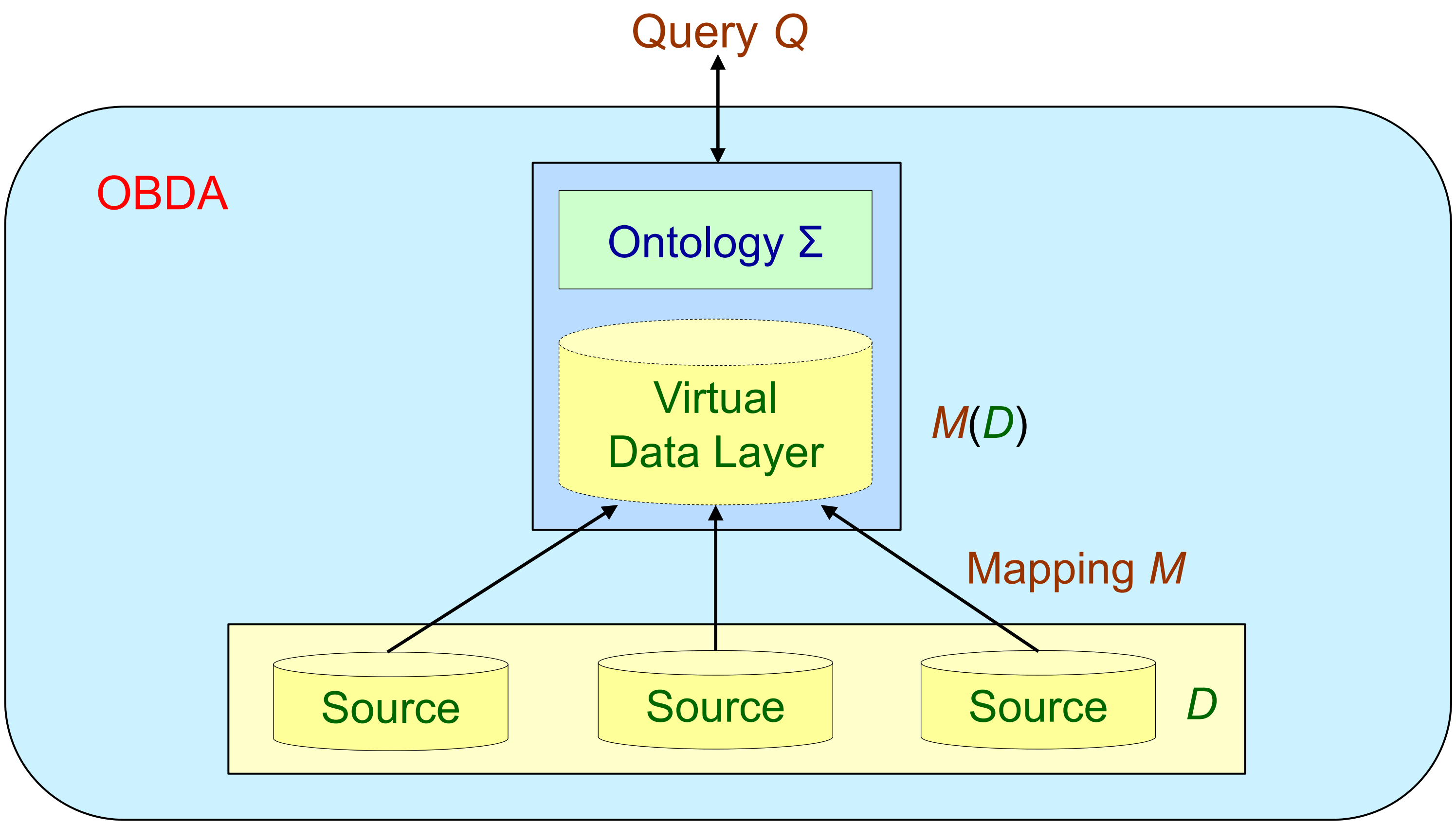
$$\text{certain}(Q,\mathcal{O}) = \bigcap_{J \in \text{models}(\mathcal{O})} Q(J)$$

Query Answering in OBDA



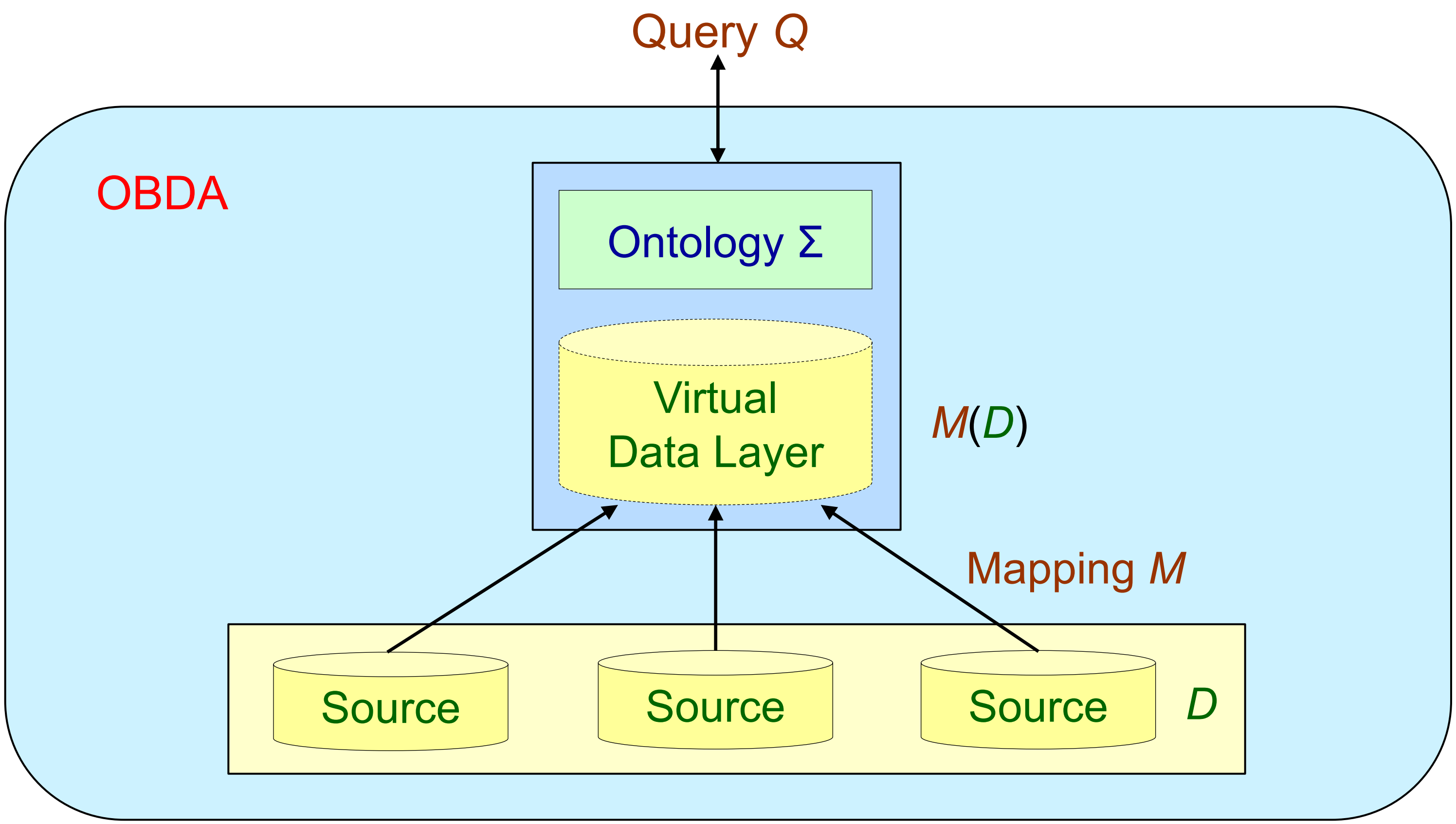
- The sources and the mapping define a virtual data layer $M(D)$

Query Answering in OBDA



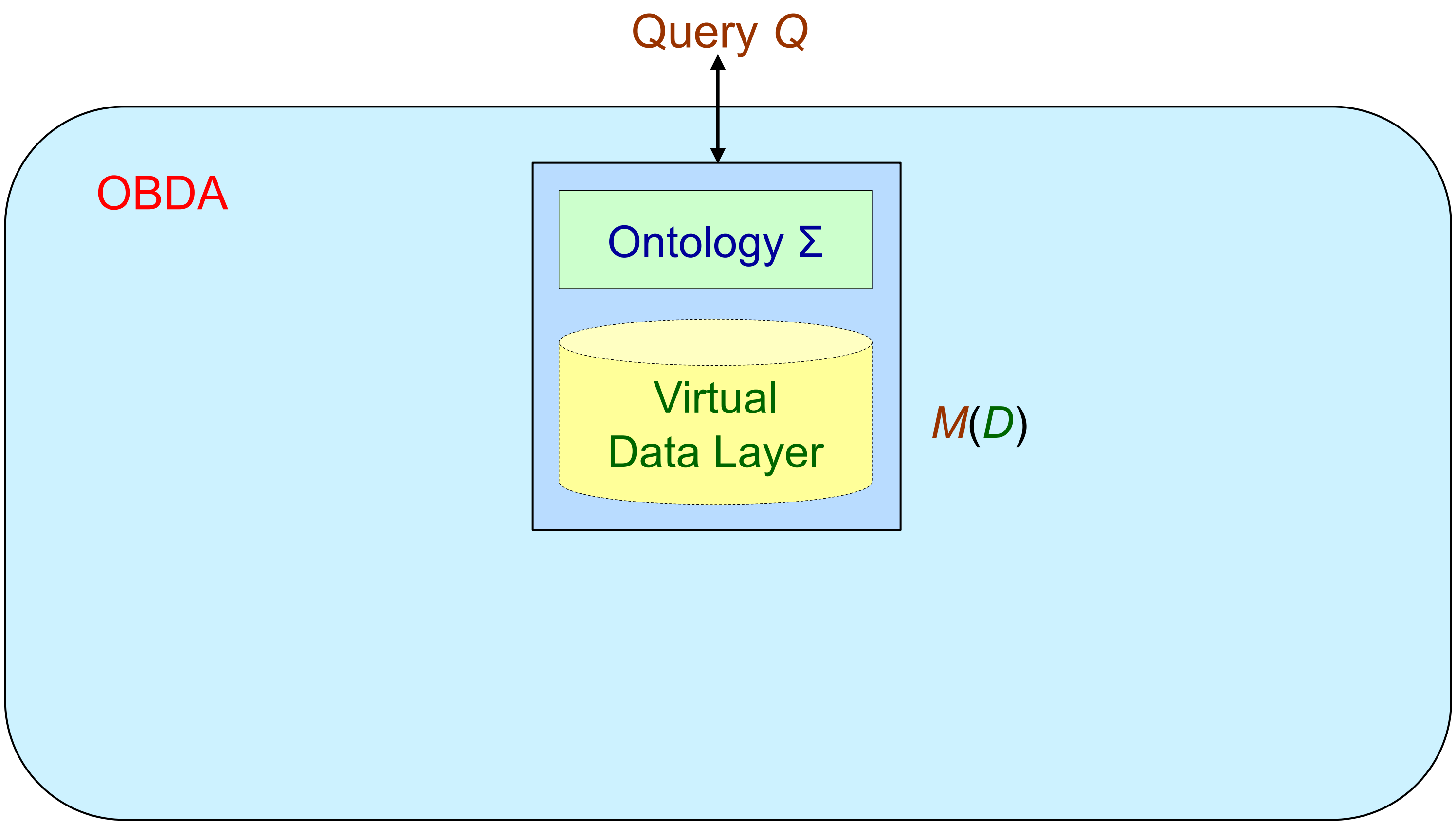
- The sources and the mapping define a virtual data layer $M(D)$
- Queries are answered against the knowledge base $\langle M(D), \Sigma \rangle$

Query Answering in OBDA



$$\text{certain}(Q, \langle \Sigma, D, M \rangle) = \text{certain}(Q, \langle M(D), \Sigma \rangle) = \bigcap_{J \in \text{models}(M(D) \wedge \Sigma)} Q(J)$$

Query Answering in OBDA

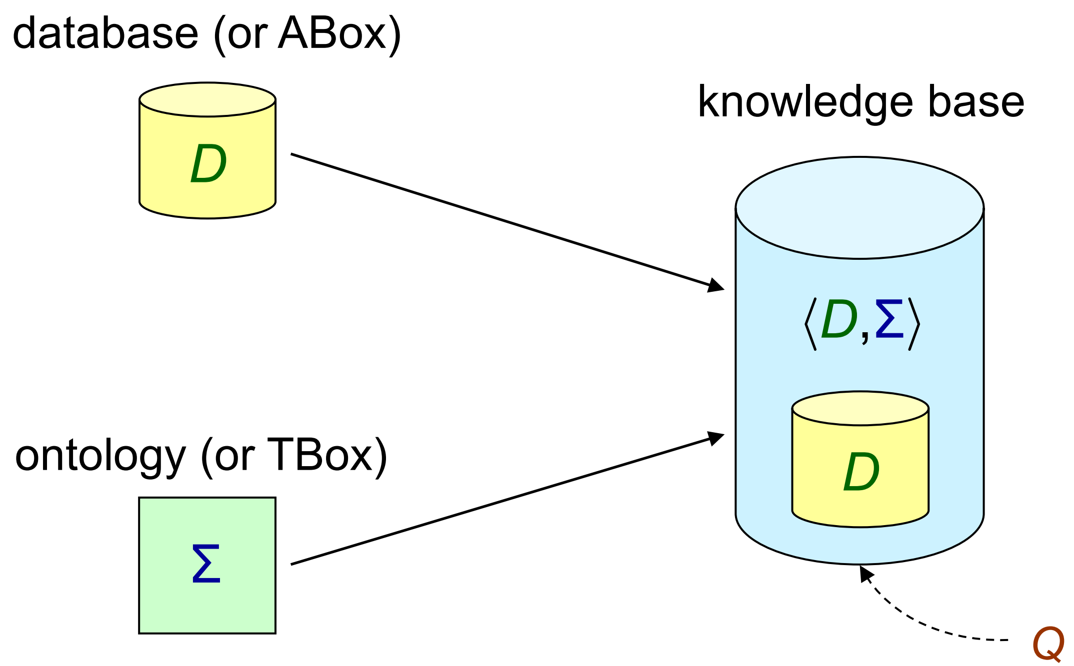


Ontology-based Query Answering

Up to Now

- Ontology-based Data Access
- Ontology-based Query Answering
- Ontology and Query Languages

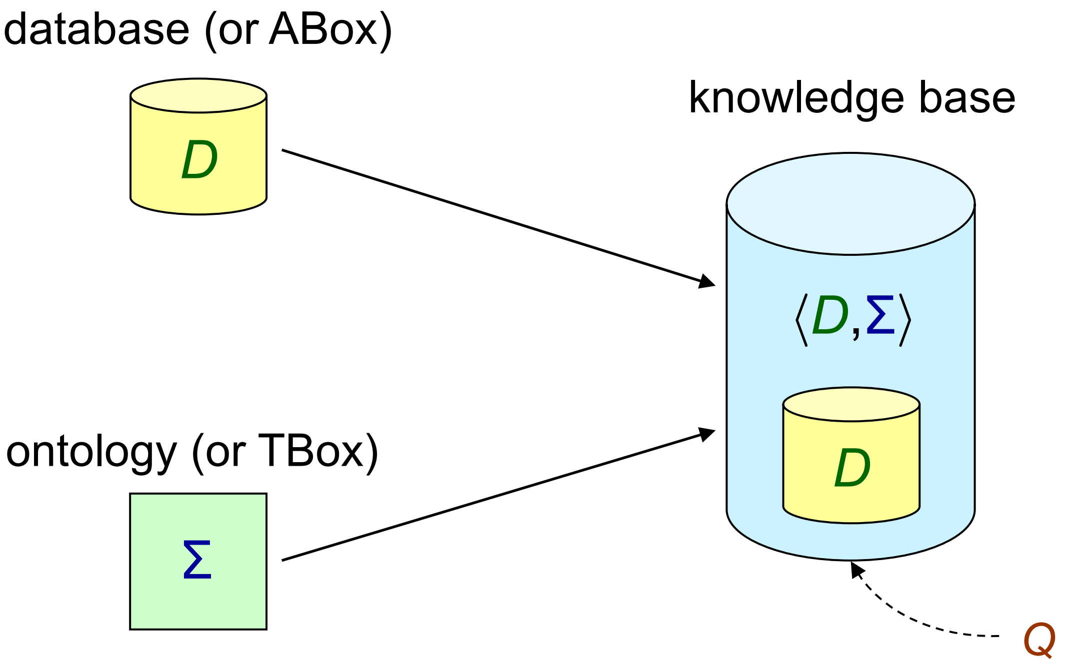
Ontology-Based Query Answering (OBQA)



$$\text{certain}(Q, \langle D,\Sigma\rangle) \quad = \quad \bigcap_{J \in \text{models}(D \wedge \Sigma)} Q(J)_{\downarrow}$$

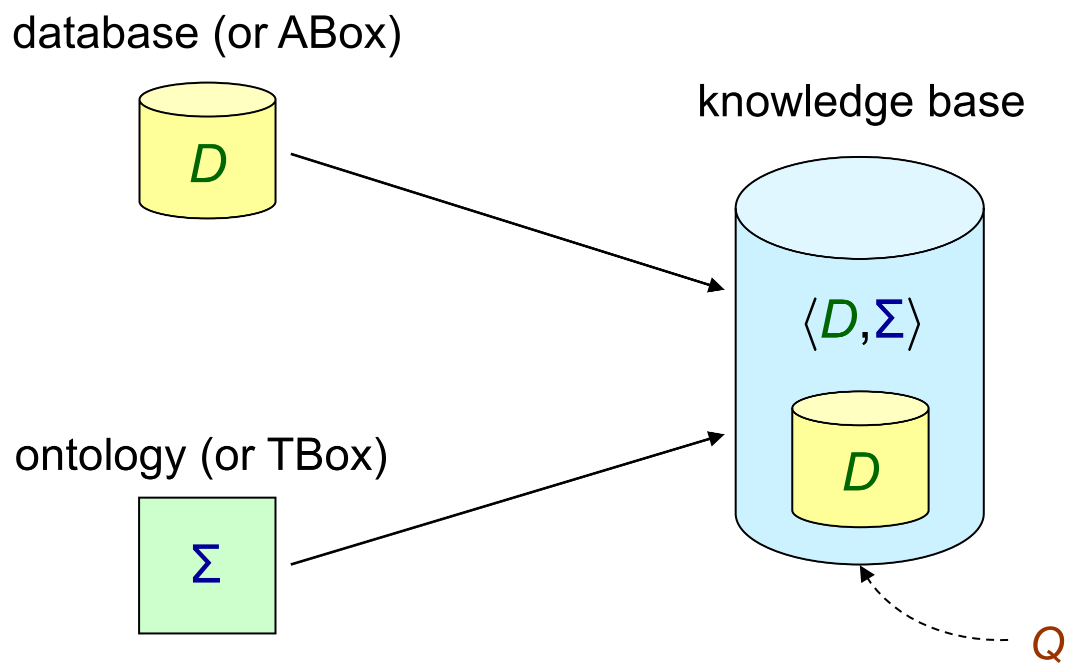
$\downarrow$ - we are interested only on ground answers that contain values from *D*

Ontology-Based Query Answering (OBQA)



ATTENTION: OBQA is not OBDA, but a crucial task in OBDA. We should talk about OBDA only in the presence of external sources and mappings

Ontology-Based Query Answering (OBQA)



This lecture is about Ontology-based Query Answering.

Issues in Ontology-Based Query Answering

What is the right ontology language?

- A wide spectrum of languages that differ in expressive power and computational complexity (e.g., description logics, existential rules)
- Scalability to very large amounts of data is a key

What is the right query language?

- Well-known languages from database theory (e.g., conjunctive queries)

Few Words on Description Logics (DLs)

- DLs are well-behaved fragments of first-order logic
- Several DL-based languages exist (from lightweight to very expressive logics)
- Strongly influenced the W3C standard Web Ontology Language OWL

- Syntax: We start from a vocabulary with
  - Concept names: atomic classes, unary predicates, e.g., *Parent*, *Person*
  - Role names: atomic relations, binary predicates, e.g., *hasParent*

  and we build axioms
  - $Person \sqsubseteq \exists hasParent.Parent$ – each person has a parent
  - $Parent \sqsubseteq Person$ – *each parent is a person*

- Semantics: via first-order interpretations

DL-Lite Family

DL-Lite: Popular family of DLs - at the basis of the OWL 2 QL profile of OWL

DL-Lite Axioms	First-order Representation
$A \sqsubseteq B$	$\forall X\ (A(X) \rightarrow B(X))$
$A \sqsubseteq \exists R$	$\forall X\ (A(X) \rightarrow \exists Y\ R(X,Y))$
$\exists R \sqsubseteq A$	$\forall X \forall Y\ (R(X,Y) \rightarrow A(X))$
$\exists R \sqsubseteq \exists P$	$\forall X \forall Y\ (R(X,Y) \rightarrow \exists Z\ P(X,Z))$
$A \sqsubseteq \exists R.B$	$\forall X\ (A(X) \rightarrow \exists Y\ (R(X,Y) \wedge B(Y)))$
$R \sqsubseteq P$	$\forall X \forall Y\ (R(X,Y) \rightarrow P(X,Y))$
$A \sqsubseteq \neg B$	$\forall X\ (A(X) \wedge B(X) \rightarrow \bot)$

The Description Logic EL

EL: Popular DL for biological applications - at the basis of OWL 2 EL profile

EL Axioms	First-order Representation
$A \sqsubseteq B$	$\forall X\ (A(X) \rightarrow B(X))$
$A \sqcap B \sqsubseteq C$	$\forall X\ (A(X) \wedge B(X) \rightarrow C(X))$
$A \sqsubseteq \exists R.B$	$\forall X\ (A(X) \rightarrow \exists Y\ (R(X,Y) \wedge B(Y)))$
$\exists R.B \sqsubseteq A$	$\forall X \forall Y\ (R(X,Y) \wedge B(Y) \rightarrow A(X))$

…several other, more powerful, description logics exist

…but, this lecture is about existential rules

an alternative way for representing ontologies

Recall our Example

Ontology Σ - high level representation of the domain of interest

$\forall X\ (\mathit{Researcher}(X) \rightarrow \exists Y\ (\mathit{worksFor}(X,Y) \wedge \mathit{Project}(Y)))$

$\forall X\ (\mathit{Project}(X) \rightarrow \exists Y\ (\mathit{worksFor}(Y,X) \wedge \mathit{Researcher}(Y)))$

$\forall X \forall Y\ (\mathit{worksFor}(X,Y) \rightarrow \mathit{Researcher}(X) \wedge \mathit{Project}(Y))$

$\forall X\ (\mathit{Project}(X) \rightarrow \exists Y\ (\mathit{PrName}(X,Y)))$

Some Notation

- Our basic vocabulary:
    - A countable set **C** of constants - domain of a database
    - A countable set **N** of (labeled) nulls - globally ∃-quantified variables
    - A countable set **V** of (regular) variables - used in rule and queries
- A term is a constant, null or variable
- An atom has the form $P(t_1, \ldots, t_n)$ where P is an n-ary predicate and each t_i is a term
- Sets of atoms are typically understood as the conjunction over their elements

Syntax of Existential Rules

An existential rule is an expression

$$\forall \mathbf{X} \forall \mathbf{Y} \, (\underbrace{\varphi(\mathbf{X},\mathbf{Y})}_{\text{body}} \rightarrow \exists \mathbf{Z} \, \underbrace{\psi(\mathbf{X},\mathbf{Z})}_{\text{head}})$$

- **X**,**Y** and **Z** are tuples of variables of **V**
- φ(**X**,**Y**) and ψ(**X**,**Z**) are (constant-free) conjunctions of atoms

…a.k.a. tuple-generating dependencies, and Datalog$^{\pm}$ rules

Homomorphism

- Semantics of existential rules via the key notion of homomorphism

- A substitution from a set of symbols **S** to a set of symbols **T** is a function $h : \mathbf{S} \rightarrow \mathbf{T}$, i.e., a set of assignments of the form $s \mapsto t$, with $s \in \mathbf{S}$ and $t \in \mathbf{T}$

- A homomorphism from a set of atoms **A** to a set of atoms **B** is a substitution $h : \mathbf{C} \cup \mathbf{N} \cup \mathbf{V} \rightarrow \mathbf{C} \cup \mathbf{N} \cup \mathbf{V}$ such that:

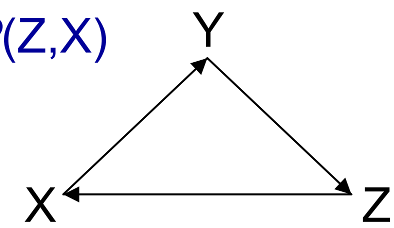
  (i) $t \in \mathbf{C} \Rightarrow h(t) = t$ (cf. unique name assumption)

  (ii) $P(t_1, \ldots, t_n) \in \mathbf{A} \Rightarrow h(P(t_1, \ldots, t_n)) := P(h(t_1), \ldots, h(t_n)) \in \mathbf{B}$

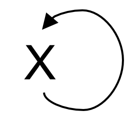
- Can be naturally extended to sets (and thus conjunctions) of atoms

Exercise: Find the Homomorphisms

$\varphi_1 = P(X,Y) \land P(Y,Z) \land P(Z,X)$



$\varphi_2 = P(X,X)$



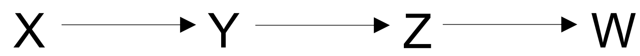
$\varphi_3 = P(X,Y) \land P(Y,X) \land P(Y,Y)$

$\varphi_4 = P(X,Y) \land P(Y,X)$

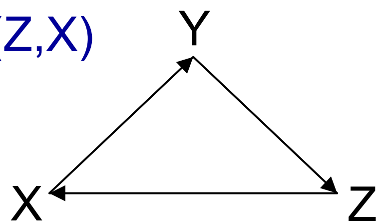
$\varphi_5 = P(X,Y) \land P(Y,Z) \land P(Z,W)$

Exercise: Find the Homomorphisms

$\varphi_5 \quad = \quad P(X,Y) \wedge P(Y,Z) \wedge P(Z,W)$

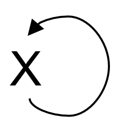


$\varphi_1 \quad = \quad P(X,Y) \wedge P(Y,Z) \wedge P(Z,X)$



$\varphi_4 \quad = \quad P(X,Y) \wedge P(Y,X)$

$\varphi_2 \quad = \quad P(X,X)$



$\varphi_3 \quad = \quad P(X,Y) \wedge P(Y,X) \wedge P(Y,Y)$