## Knowledge Bases and Databases

Part 1: First-Order Queries

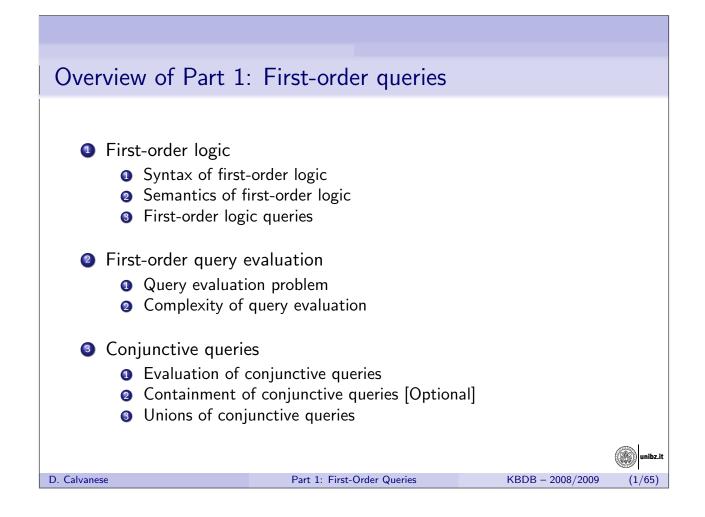
#### Diego Calvanese

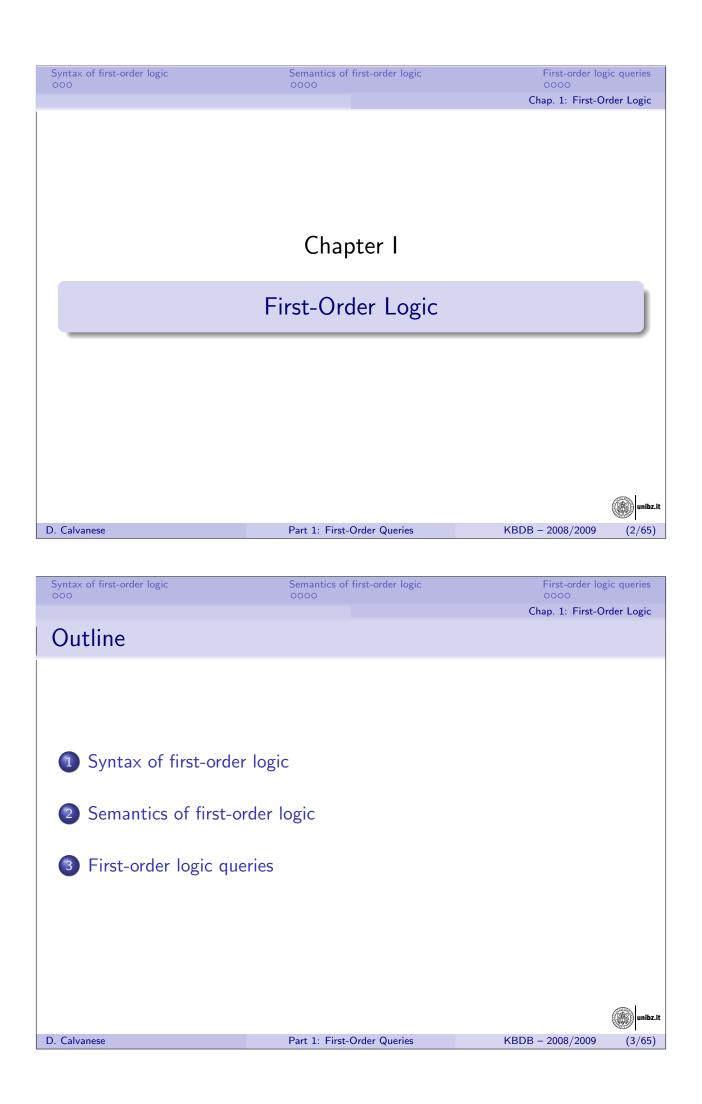
Faculty of Computer Science Master of Science in Computer Science

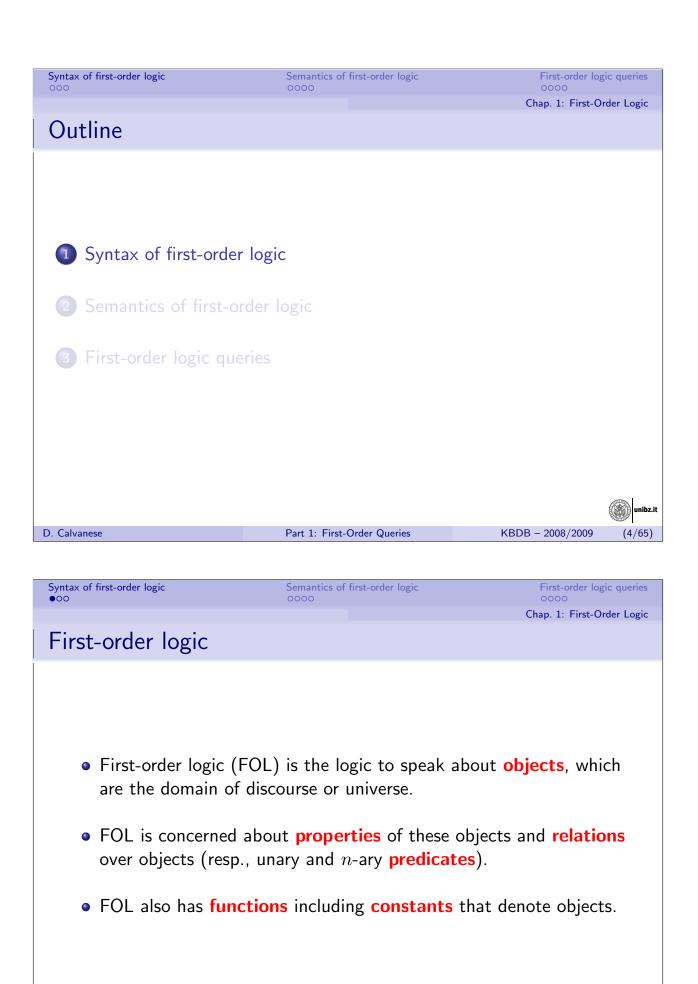
A.Y. 2008/2009



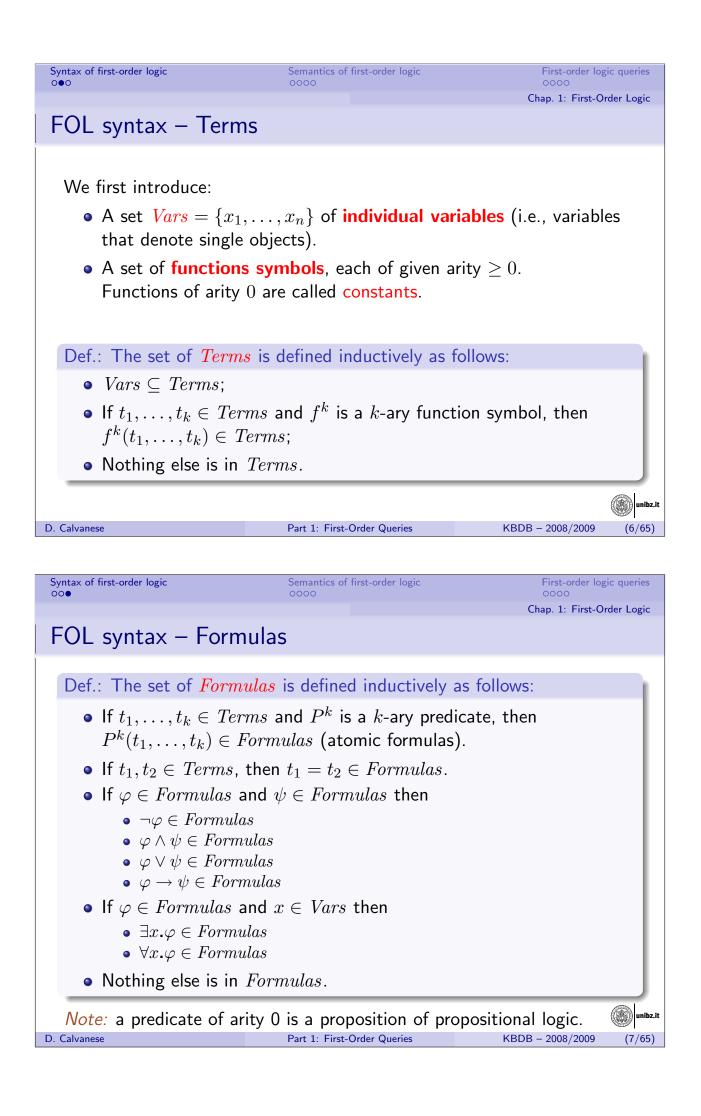
Freie Universität Bozen Libera Università di Bolzano Free University of Bozen · Bolzano

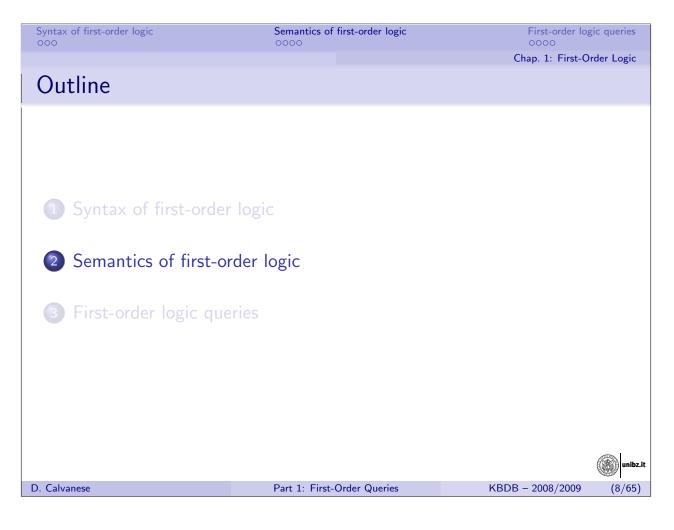




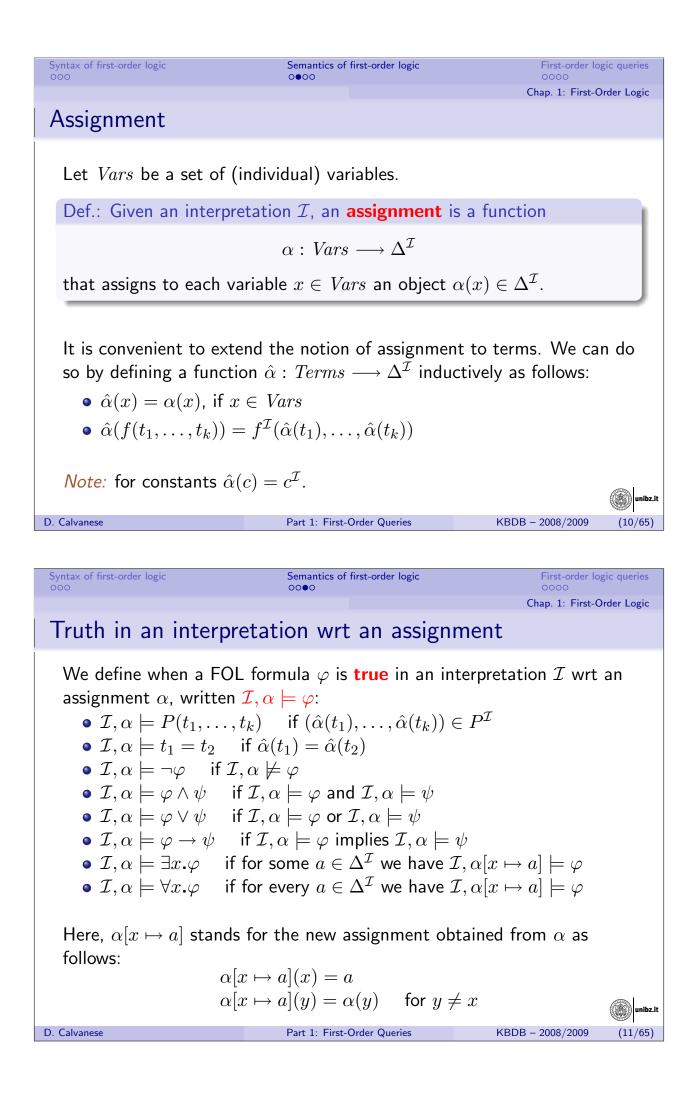


(5/65)

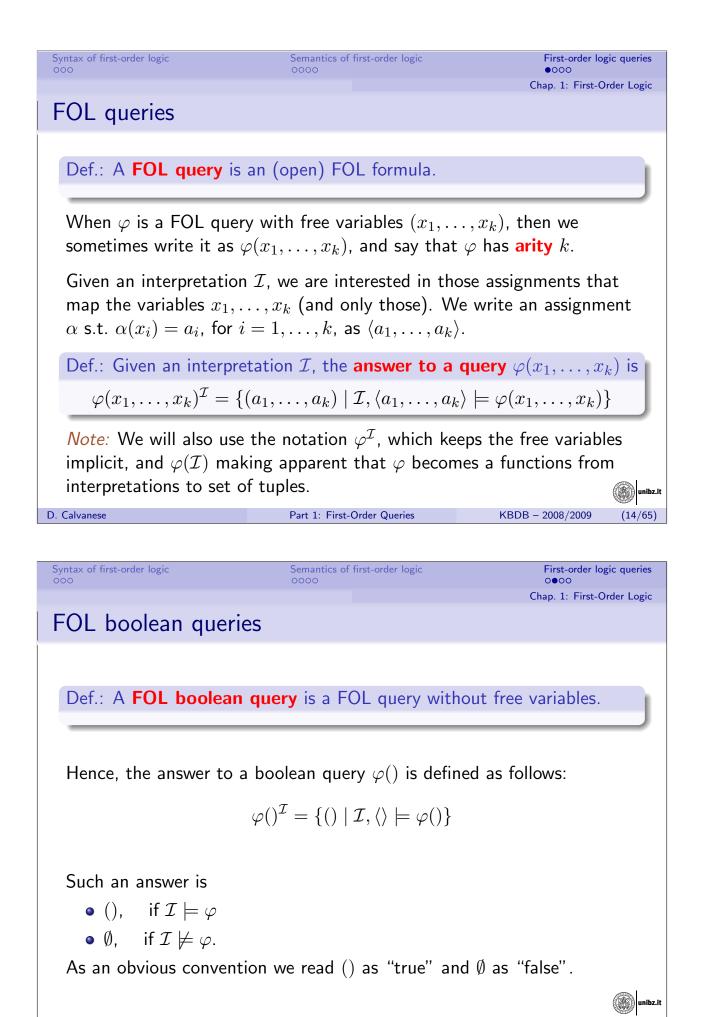




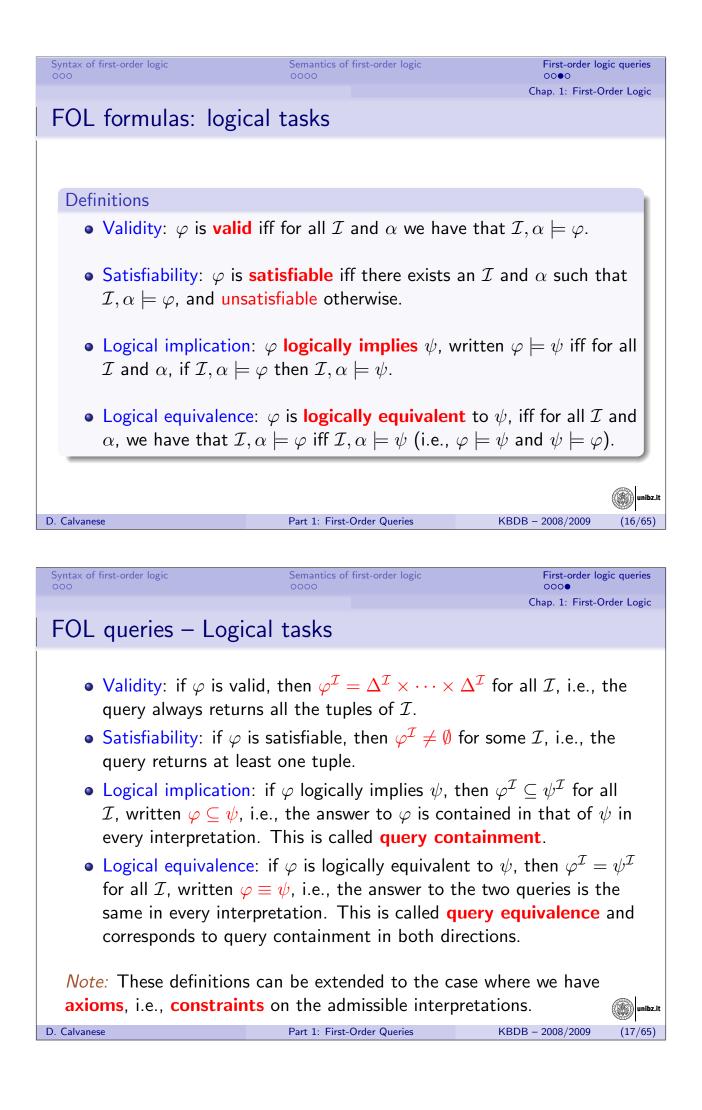
Syntax of first-order logic	Semantics of first-order logic	First-order logic queries
		Chap. 1: First-Order Logic
Interpretations		
-	predicates $P_1, P_2, \dots$ and f d arity, a FOL interpretati	• • • •
$\mathcal{I}$ =	$= (\Delta^{\mathcal{I}}, P_1^{\mathcal{I}}, P_2^{\mathcal{I}}, \dots, f_1^{\mathcal{I}}, f_2^{\mathcal{I}},$	)
where:		
$ullet$ $\Delta^{\mathcal{I}}$ is the domain	(a set of objects)	
• if $P_i$ is a $k$ -ary pre-	edicate, then $P_i^\mathcal{I} \subseteq \Delta^\mathcal{I}  imes \cdot$	$\cdots  imes \Delta^{\mathcal{I}}$ ( $k$ times)
• if $f_i$ is a $k$ -ary fur	ction, then $f_i^\mathcal{I}:\Delta^\mathcal{I} imes\cdots imes$	$<\Delta^{\mathcal{I}} \longrightarrow \Delta^{\mathcal{I}}$ (k times)
• if $f_i$ is a constant	(i.e., a 0-ary function), the	en $f_i^\mathcal{I}:()\longrightarrow \Delta^\mathcal{I}$
(i.e., $f_i$ denotes e	xactly one object of the do	main)
		() unibz.it
D. Calvanese	Part 1: First-Order Queries	KBDB – 2008/2009 (9/65)



000	Semantics of first-order logic	First-order logic queries 0000
Open vs. closed fo	ormulas	Chap. 1: First-Order Logic
of any quantifier, An open formul	formula $\varphi$ is <b>free</b> if $x$ does r , otherwise it is <b>bound</b> . a is a formula that has some a, also called <b>sentence</b> , is a	free variable.
means to be <b>true in</b> a mentioning the assign role in verifying $\mathcal{I}, \alpha$	(but not for open formulas) $\varphi$ an interpretation, written $\mathcal{I}$ ment, since the assignment $\varphi$ = $\varphi$ . Is are strongly related to <b>que</b>	$\varphi \models \varphi$ , without $\alpha$ does not play any
D. Calvanese	Part 1: First-Order Queries	KBDB – 2008/2009 (12/65)
Syntax of first-order logic	Semantics of first-order logic	First-order logic queries
Syntax of first-order logic	Semantics of first-order logic 0000	First-order logic queries 0000 Chap. 1: First-Order Logic
		0000
000		0000
000	0000	0000
Outline	oooo	0000
Outline Syntax of first-orde	er logic order logic	0000
<ul> <li>Outline</li> <li>Syntax of first-orde</li> <li>Semantics of first-orde</li> </ul>	er logic order logic	0000
Outline  Syntax of first-orde  Semantics of first-orde	er logic order logic	0000
<ul> <li>Outline</li> <li>Syntax of first-orde</li> <li>Semantics of first-orde</li> </ul>	er logic order logic	0000



(15/65)



Chap. 2: First-Order Query Evaluation

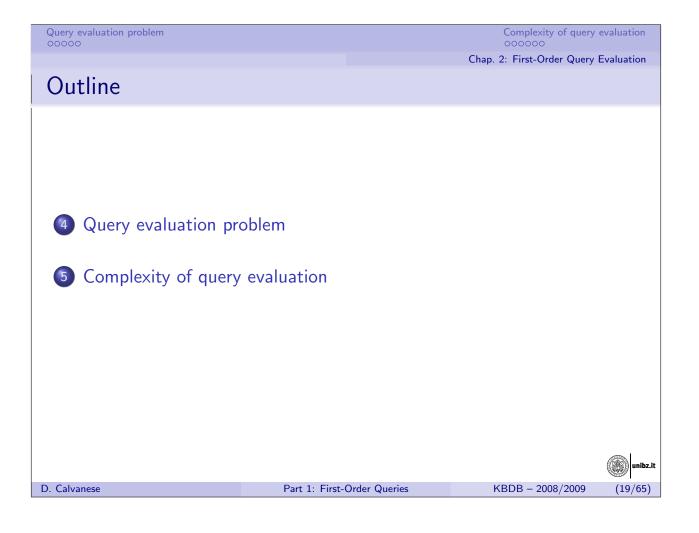
# Chapter II

## First-Order Query Evaluation

D. Calvanese

Part 1: First-Order Queries

KBDB – 2008/2009 (18/65)



Query evaluation problem		Complexity of query	evaluation
		Chap. 2: First-Order Query	Evaluation
Outline			
Query evaluation pr	oblem		
5 Complexity of query	v evaluation		
			unibz.it
D. Calvanese	Part 1: First-Order Queries	KBDB - 2008/2009	(20/65)
Query evaluation problem		Complexity of query	evaluation
		Chap. 2: First-Order Query	Evaluation

Query evaluation

Let us consider:

- a **finite alphabet**, i.e., we have a finite number of predicates and functions, and
- a finite interpretation  $\mathcal{I}$ , i.e., an interpretation (over the finite alphabet) for which  $\Delta^{\mathcal{I}}$  is finite.

Then we can consider query evaluation as an algorithmic problem, and study its computational properties.

*Note:* To study the **computational complexity** of the problem, we need to define a corresponding decision problem.



Chap. 2: First-Order Query Evaluation

#### Query evaluation problem

#### Definitions

• Query answering problem: given a finite interpretation  $\mathcal{I}$  and a FOL query  $\varphi(x_1, \ldots, x_k)$ , compute

$$\varphi^{\mathcal{I}} = \{ (a_1, \dots, a_k) \mid \mathcal{I}, \langle a_1, \dots, a_k \rangle \models \varphi(x_1, \dots, x_k) \}$$

• Recognition problem (for query answering): given a finite interpretation  $\mathcal{I}$ , a FOL query  $\varphi(x_1, \ldots, x_k)$ , and a tuple  $(a_1, \ldots, a_k)$ , with  $a_i \in \Delta^{\mathcal{I}}$ , check whether  $(a_1, \ldots, a_k) \in \varphi^{\mathcal{I}}$ , i.e., whether

 $\mathcal{I}, \langle a_1, \ldots, a_k \rangle \models \varphi(x_1, \ldots, x_k)$ 

*Note:* The recognition problem for query answering is the decision problem corresponding to the query answering problem.

```
D. Calvanese
```

Part 1: First-Order Queries

KBDB – 2008/2009

(22/65)

Query evaluation problem

Complexity of query evaluation 000000 Chap. 2: First-Order Query Evaluation

#### Query evaluation algorithm

We define now an algorithm that computes the function  $\mathtt{Truth}(\mathcal{I}, \alpha, \varphi)$ in such a way that  $\mathtt{Truth}(\mathcal{I}, \alpha, \varphi) = \mathtt{true}$  iff  $\mathcal{I}, \alpha \models \varphi$ .

We make use of an auxiliary function  $\text{TermEval}(\mathcal{I}, \alpha, t)$  that, given an interpretation  $\mathcal{I}$  and an assignment  $\alpha$ , evaluates a term t returning an object  $o \in \Delta^{\mathcal{I}}$ :

```
\begin{array}{l} \Delta^{\mathcal{I}} \; \operatorname{TermEval}(\mathcal{I}, \alpha, t) \; \{ \\ & \text{ if } (t \; \text{ is } x \in Vars) \\ & \text{ return } \alpha(x); \\ & \text{ if } (t \; \text{ is } f(t\_1, \ldots, t\_k)) \\ & \text{ return } f^{\mathcal{I}}(\operatorname{TermEval}(\mathcal{I}, \alpha, t\_1), \ldots, \operatorname{TermEval}(\mathcal{I}, \alpha, t\_k)); \\ \} \end{array}
```

Then, Truth $(\mathcal{I}, \alpha, \varphi)$  can be defined by structural recursion on  $\varphi$ .

(23/65)

(24/65)

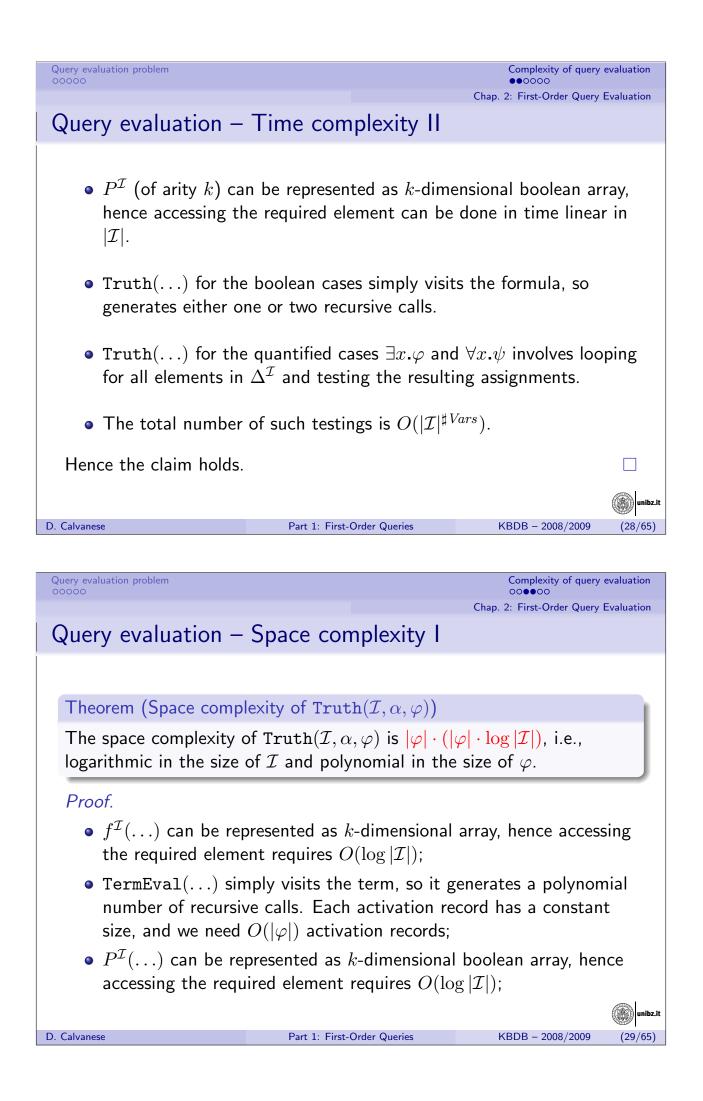
Chap. 2: First-Order Query Evaluation

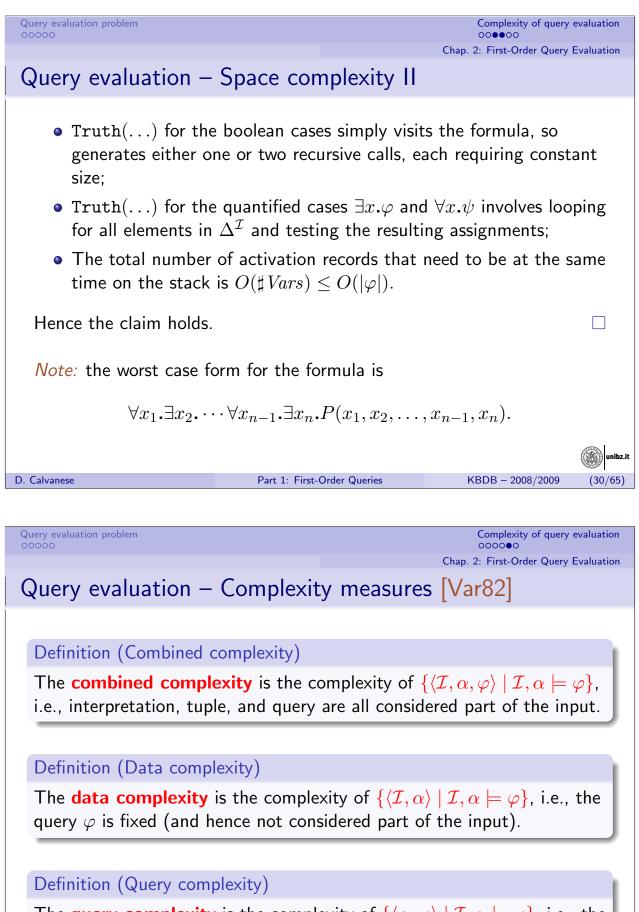
### Query evaluation algorithm (cont'd)

```
boolean Truth(\mathcal{I}, \alpha, \varphi) {
        if (\varphi is t_{-1} = t_{-2})
            return TermEval(\mathcal{I}, \alpha, t_{-1}) = TermEval(\mathcal{I}, \alpha, t_{-2});
        if (\varphi is P(t_1, \ldots, t_k))
            return P^{\mathcal{I}} (TermEval(\mathcal{I}, \alpha, t_{-1}),..., TermEval(\mathcal{I}, \alpha, t_{-k}));
        if (\varphi is \neg \psi)
           return \negTruth(\mathcal{I}, \alpha, \psi);
        if (\varphi \text{ is } \psi \circ \psi')
           return Truth(\mathcal{I}, \alpha, \psi) \circ \text{Truth}(\mathcal{I}, \alpha, \psi');
        if (\varphi is \exists x.\psi) {
           boolean b = false;
            for all ( a \in \Delta^{\mathcal{I}} )
                 b = b \vee Truth(\mathcal{I}, \alpha[x \mapsto a], \psi);
            return b;
        }
        if (\varphi is \forall x.\psi) {
            boolean b = true;
            for all ( a \in \Delta^{\mathcal{I}} )
                 b = b \wedge Truth(\mathcal{I}, \alpha[x \mapsto a], \psi);
           return b;
        }
     }
                                                                                                                   KBDB - 2008/2009
D. Calvanese
                                                             Part 1: First-Order Queries
```

Query evaluation problem Complexity of query evaluation 00000 Chap. 2: First-Order Query Evaluation Query evaluation – Results Theorem (Termination of Truth $(\mathcal{I}, \alpha, \varphi)$ ) The algorithm Truth terminates. Proof. Immediate. Theorem (Correctness) The algorithm Truth is sound and complete, i.e.,  $\mathcal{I}, \alpha \models \varphi$  if and only if  $Truth(\mathcal{I}, \alpha, \varphi) = true.$ *Proof.* Easy, since the algorithm is very close to the semantic definition of  $\mathcal{I}, \alpha \models \varphi$ . D. Calvanese Part 1: First-Order Queries KBDB - 2008/2009 (25/65)

Query evaluation problem	Complexity of query	evaluation
	Chap. 2: First-Order Query	Evaluation
Outline		
4 Query evaluation problem		
5 Complexity of query evaluation		
		<i>.</i> .
D. Calvanese Part 1: First-Order Queries	KBDB – 2008/2009	(26/65)
Query evaluation problem	Complexity of query ●●0000	evaluation
	••0000	
	••0000	
Query evaluation – Time complexity I	••0000	
Query evaluation – Time complexity I Theorem (Time complexity of $Truth(\mathcal{I}, \alpha, \varphi)$ )	••0000 Chap. 2: First-Order Query	
Query evaluation – Time complexity I	•••••••••••••••••••••••••••••••••••••	
Query evaluation – Time complexity I Theorem (Time complexity of Truth( $\mathcal{I}, \alpha, \varphi$ )) The time complexity of Truth( $\mathcal{I}, \alpha, \varphi$ ) is ( $ \mathcal{I} $ +	•••••••••••••••••••••••••••••••••••••	
Query evaluation – Time complexity I Theorem (Time complexity of Truth( $\mathcal{I}, \alpha, \varphi$ )) The time complexity of Truth( $\mathcal{I}, \alpha, \varphi$ ) is ( $ \mathcal{I} $ +	•••••••••••••••••••••••••••••••••••••	
<b>Query evaluation</b> – Time complexity I Theorem (Time complexity of Truth( $\mathcal{I}, \alpha, \varphi$ )) The time complexity of Truth( $\mathcal{I}, \alpha, \varphi$ ) is ( $ \mathcal{I} $ + polynomial in the size of $\mathcal{I}$ and exponential in the size of $\mathcal{I}$ an	Chap. 2: First-Order Query $ \alpha  +  \varphi )^{ \varphi }$ , i.e., he size of $\varphi$ .	Evaluation
<b>Query evaluation</b> – Time complexity I Theorem (Time complexity of Truth( $\mathcal{I}, \alpha, \varphi$ )) The time complexity of Truth( $\mathcal{I}, \alpha, \varphi$ ) is ( $ \mathcal{I} $ + polynomial in the size of $\mathcal{I}$ and exponential in the size of $\mathcal{I}$ an	Chap. 2: First-Order Query $ \alpha  +  \varphi )^{ \varphi }$ , i.e., he size of $\varphi$ .	Evaluation
<b>Query evaluation</b> – Time complexity I Theorem (Time complexity of $Truth(\mathcal{I}, \alpha, \varphi)$ ) The time complexity of $Truth(\mathcal{I}, \alpha, \varphi)$ is $( \mathcal{I}  + polynomial in the size of \mathcal{I} and exponential in theProof.• f^{\mathcal{I}} (of arity k) can be represented as k-dimension$	Chap. 2: First-Order Query Chap. 2: First-Order Query for $ \alpha  +  \varphi )^{ \varphi }$ , i.e., he size of $\varphi$ .	Evaluation e
<b>Query evaluation</b> – Time complexity I Theorem (Time complexity of Truth( $\mathcal{I}, \alpha, \varphi$ )) The time complexity of Truth( $\mathcal{I}, \alpha, \varphi$ ) is ( $ \mathcal{I}  + \phi$ ) polynomial in the size of $\mathcal{I}$ and exponential in the <b>Proof.</b> • $f^{\mathcal{I}}$ (of arity $k$ ) can be represented as $k$ -dimensional constraints of the required element can be done • TermEval() visits the term, so it general	Chap. 2: First-Order Query Chap. 2: First-Order Query for $ \alpha  +  \varphi )^{ \varphi }$ , i.e., he size of $\varphi$ .	Evaluation e





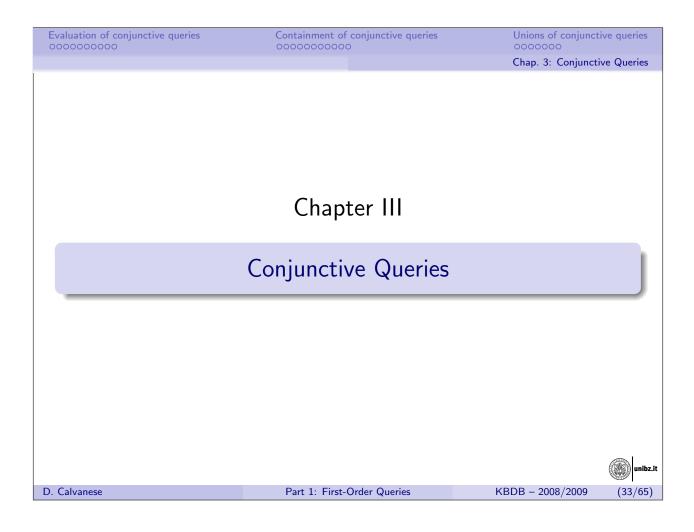
The **query complexity** is the complexity of  $\{\langle \alpha, \varphi \rangle \mid \mathcal{I}, \alpha \models \varphi\}$ , i.e., the interpretation  $\mathcal{I}$  is fixed (and hence not considered part of the input).

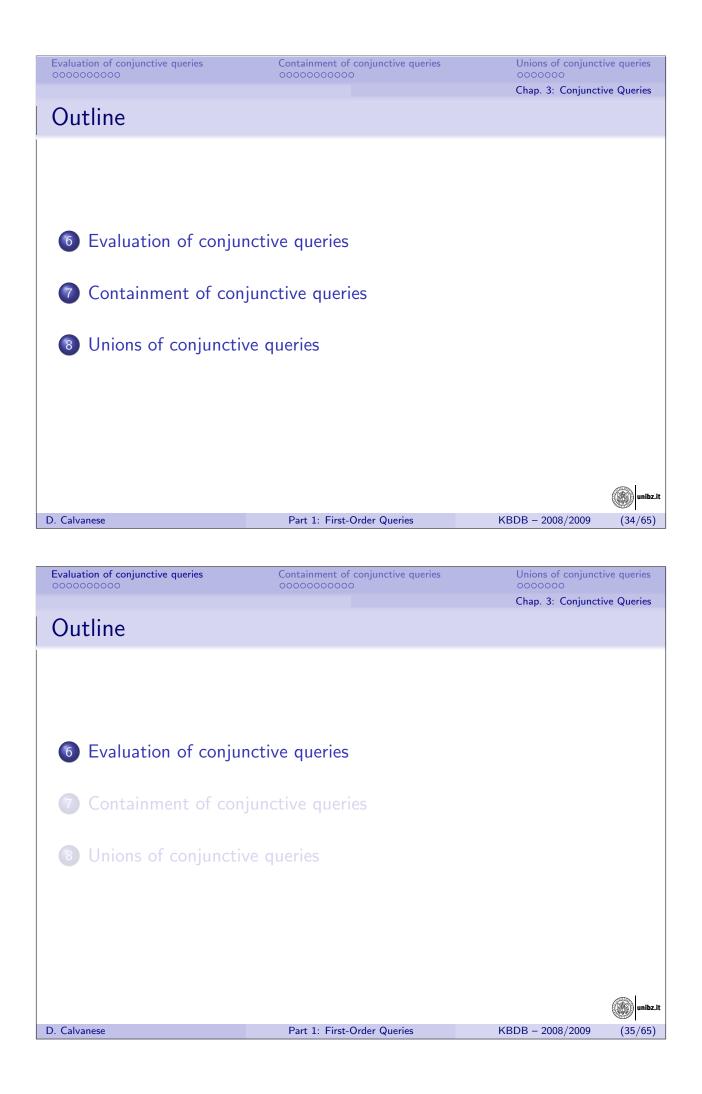
unibz.it

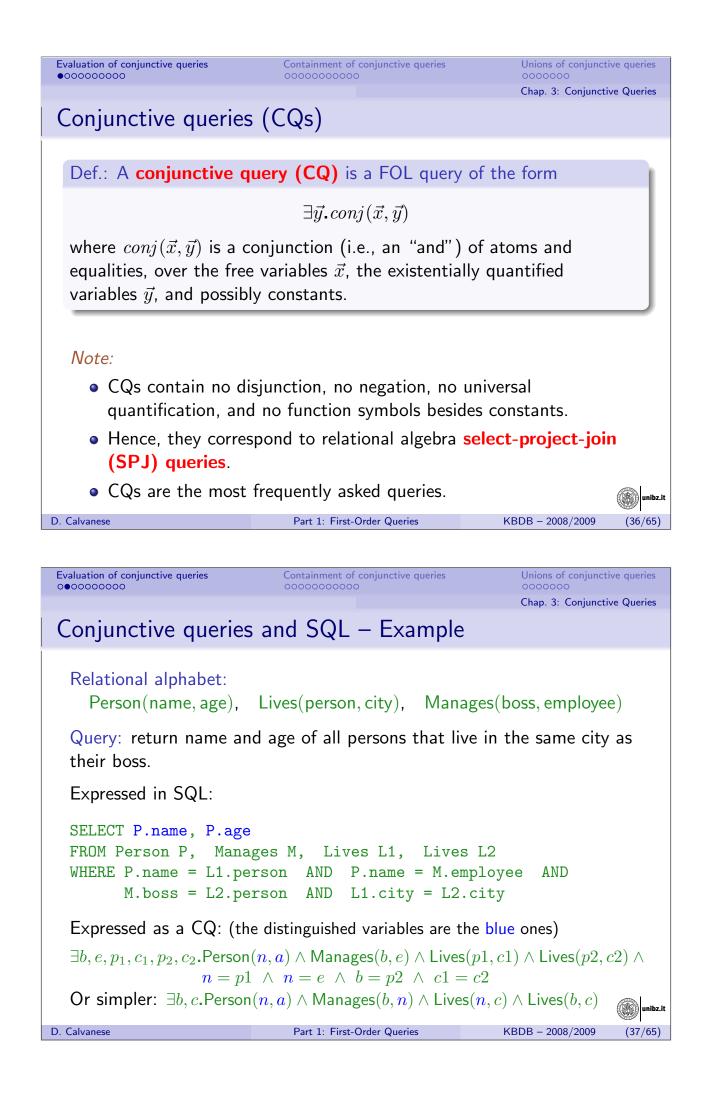
(31/65)

KBDB - 2008/2009

Query evaluation problem		Complexity of query o	evaluation
		Chap. 2: First-Order Query I	Evaluation
Query evaluation – Co	ombined, data, que	ery complexity	
Theorem (Combined comp		on)	
<ul> <li>The complexity of { (<i>I</i>, α,</li> <li>time: exponential</li> </ul>	$\left  arphi  ight angle \left  arphi ,lpha \models arphi  ight brace$ is:		
• space: PSPACE-comp	olete — see [Var82] for	hardness	
Theorem (Data complexity	y of query evaluation)		
The complexity of $\{ \langle \mathcal{I}, \alpha \rangle$	$\mid \mathcal{I}, \alpha \models \varphi \}$ is:		
• time: polynomial			
• space: in LOGSPACE			
Theorem (Query complexi	ty of query evaluation)		
The complexity of $\{\langle lpha, arphi  angle$	$\mid \mathcal{I}, \alpha \models \varphi \}$ is:		
• time: exponential			
• space: PSPACE-comp	olete — see [Var82] for	hardness	z.it
D. Calvanese	Part 1: First-Order Queries	KBDB – 2008/2009	(32/65)







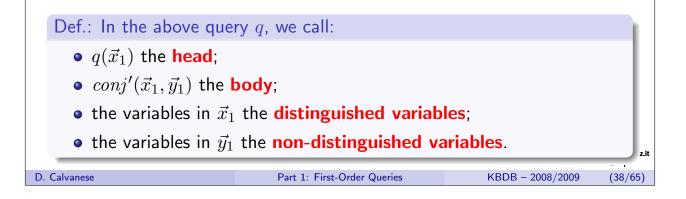
#### Datalog notation for CQs

A CQ  $q = \exists \vec{y}.conj(\vec{x},\vec{y})$  can also be written using datalog notation as

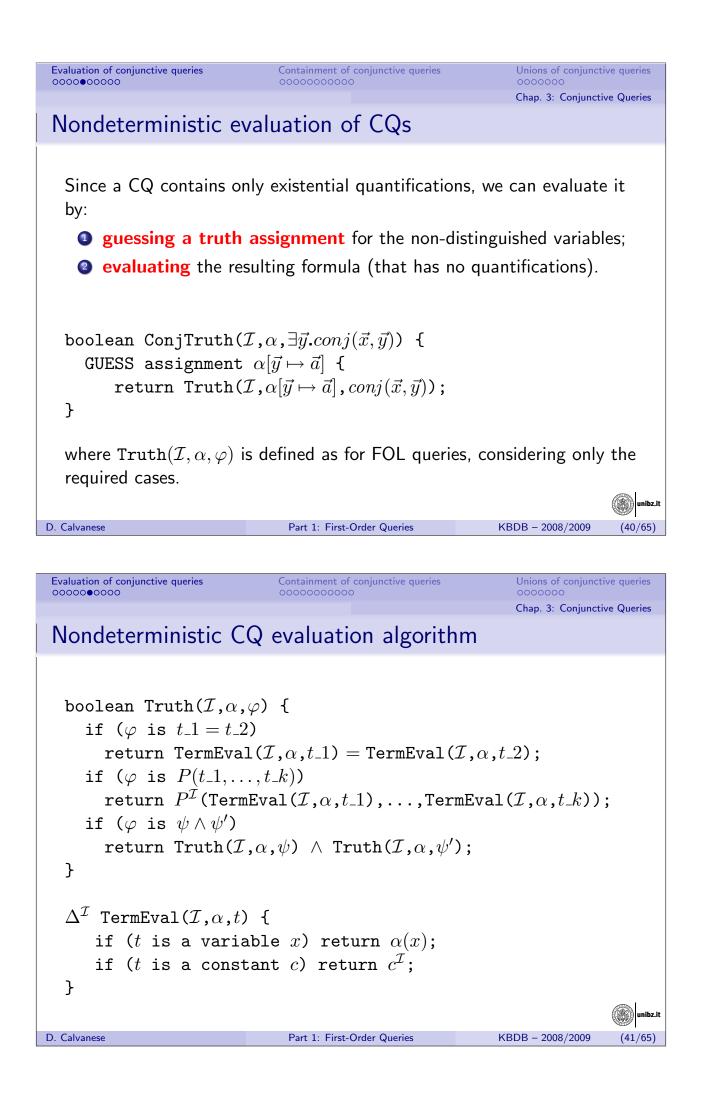
 $q(\vec{x}_1) \leftarrow conj'(\vec{x}_1, \vec{y}_1)$ 

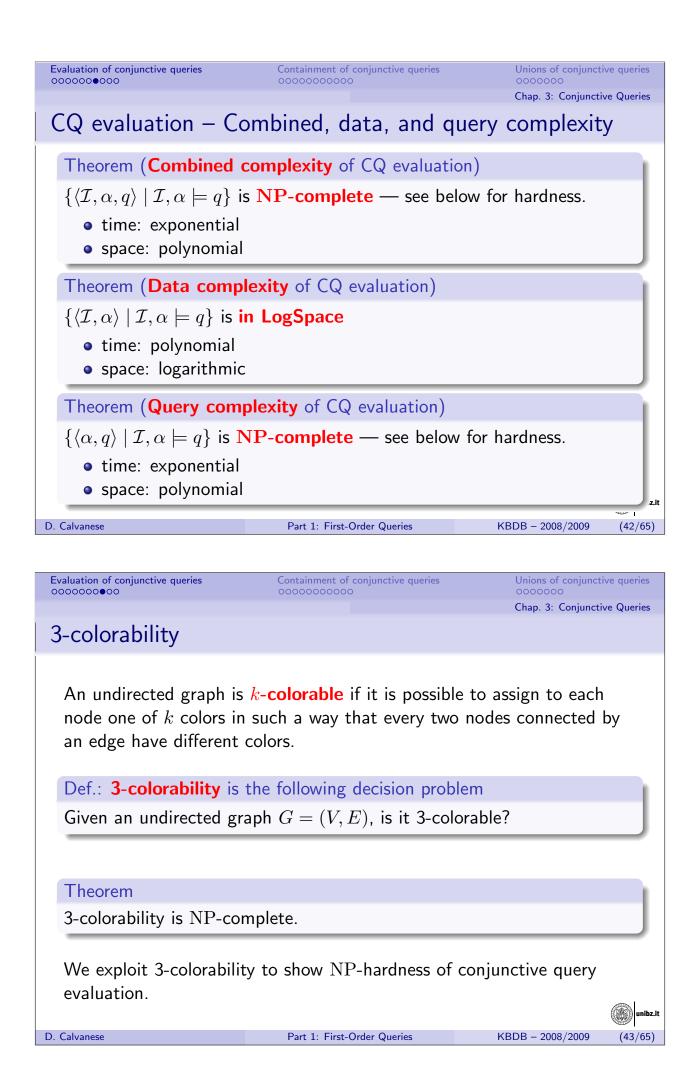
where  $conj'(\vec{x}_1, \vec{y}_1)$  is the list of atoms in  $conj(\vec{x}, \vec{y})$  obtained by equating the variables  $\vec{x}$ ,  $\vec{y}$  according to the equalities in  $conj(\vec{x}, \vec{y})$ .

As a result of such an equality elimination, we have that  $\vec{x}_1$  and  $\vec{y}_1$  can contain constants and multiple occurrences of the same variable.



Evaluation of conjunctive queries	Containment of conjunctive queries	Unions of conjunctive	queries
		Chap. 3: Conjunctive (	Queries
Conjunctive queries	– Example		
relation – note that	pretation $\mathcal{I} = (\Delta^{\mathcal{I}}, E^{\mathcal{I}})$ , when the such interpretation is a (dimensional dimension of the state of the sta	rected) graph.	0
in the graph:	q returns an nodes that part		e
Ξ	$y, z.E(x, y) \wedge E(y, z) \wedge E(z, y)$	,x)	
• The query $q$ in da	talog notation becomes:		
Ģ	$q(\boldsymbol{x}) \leftarrow E(\boldsymbol{x}, y), E(y, z), E(z, y)$	x)	
SELECT E1.f	L is (we use Edge(f,s) for	E(x,y):	
-	2.f AND E2.s = E3.f AND	E3.s = E1.f	unibz.it
D. Calvanese	Part 1: First-Order Queries	KBDB - 2008/2009	(39/65)

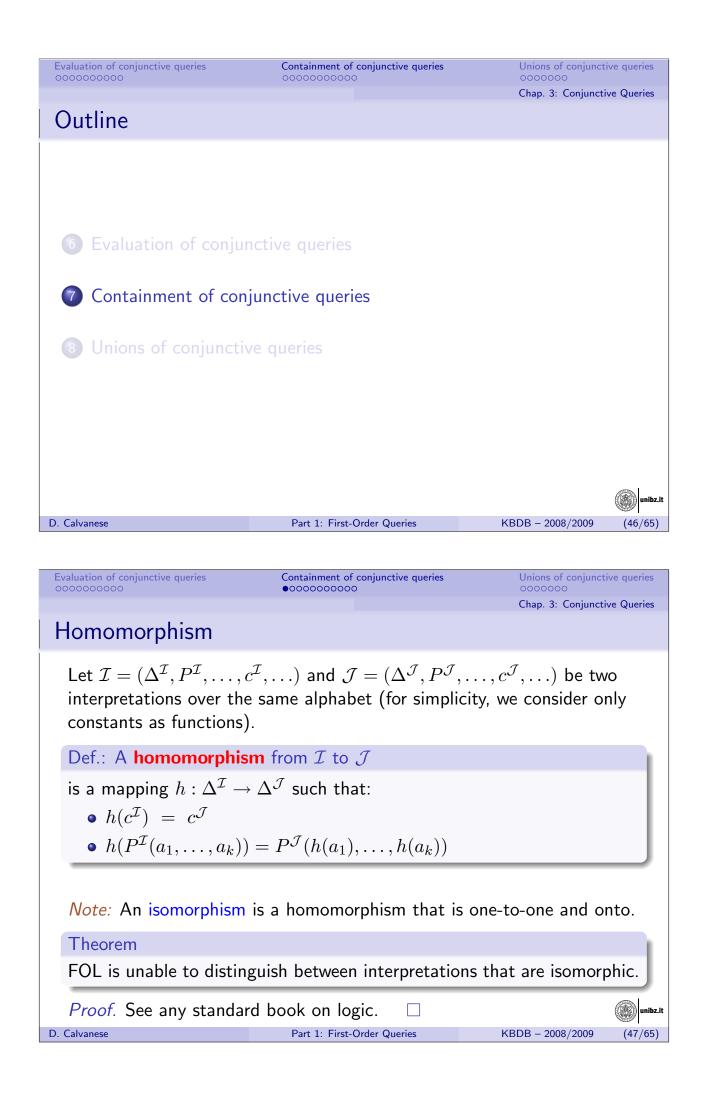


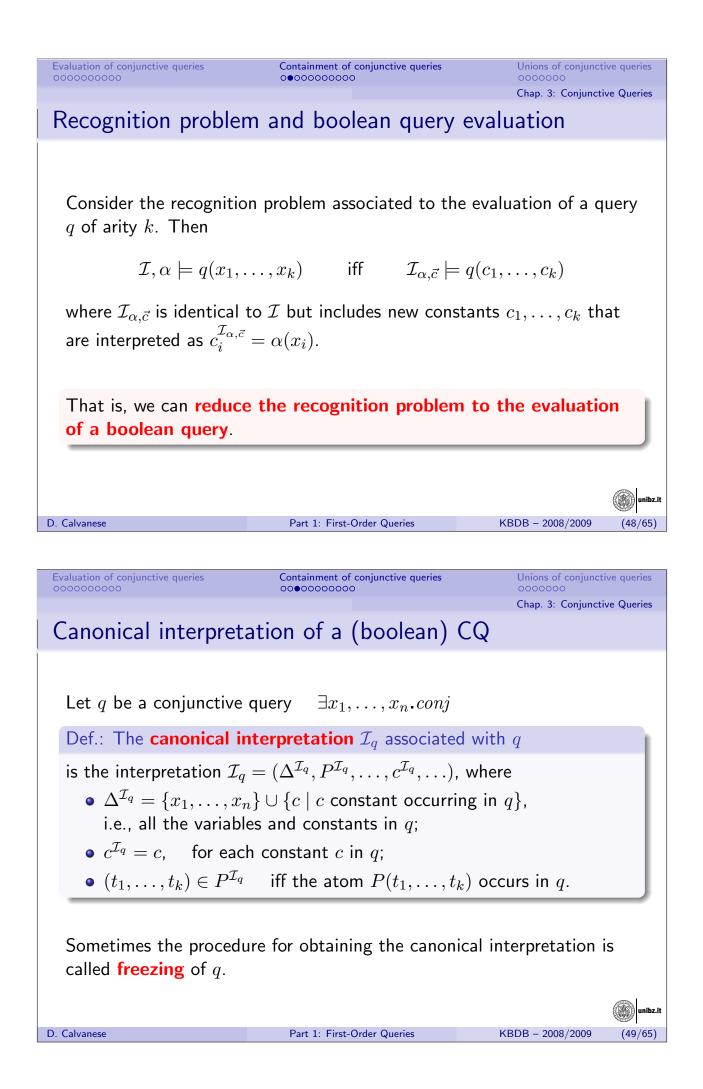


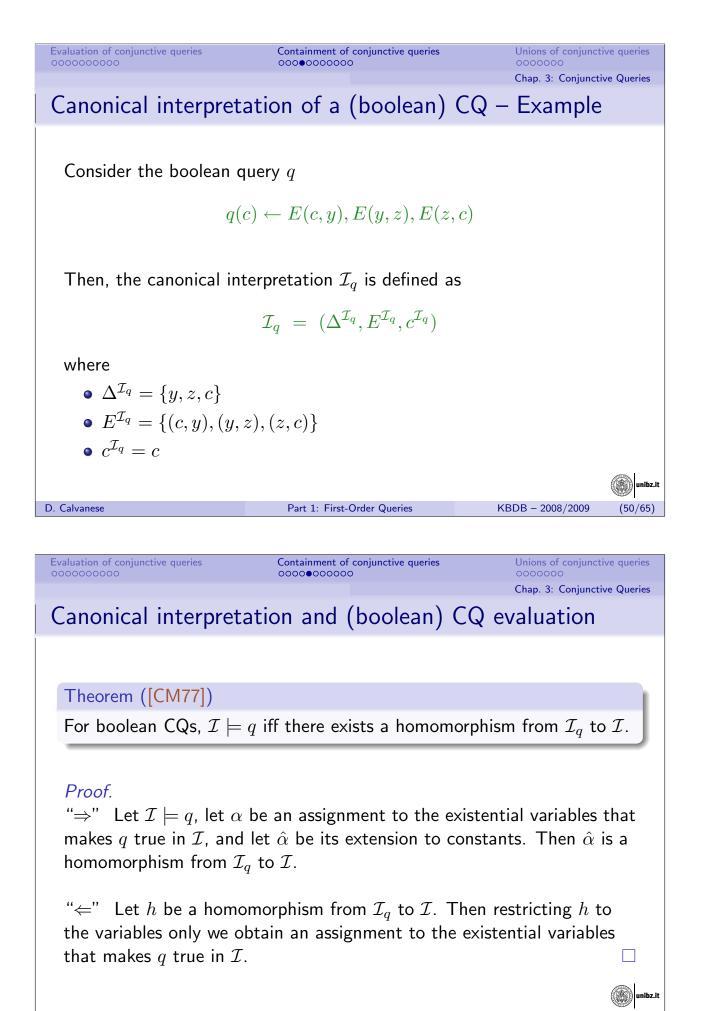
#### Reduction from 3-colorability to CQ evaluation

Let G = (V, E) be an undirected graph. We consider a relational alphabet consisting of a single binary relation Edge and define: • An Interpretation:  $\mathcal{I} = (\Delta^{\mathcal{I}}, \text{Edge}^{\mathcal{I}})$  where: •  $\Delta^{\mathcal{I}} = \{r, g, b\}$ • Edge<sup> $\mathcal{I}$ </sup> =  $\{(r, g), (g, r), (r, b), (b, r), (g, b), (b, g)\}$ • A conjunctive query: Let  $V = \{x_1, \dots, x_n\}$ , then consider the boolean conjunctive query defined as:  $q_G = \exists x_1, \dots, x_n$ .  $\bigwedge_{(x_i, x_j) \in E} \text{Edge}(x_i, x_j) \land \text{Edge}(x_j, x_i)$ Theorem *G* is 3-colorable iff  $\mathcal{I} \models q_G$ .

Evaluation of conjunctive queries	Containment of conjunctive queries	Unions of conjunctive que
		Chap. 3: Conjunctive Que
NP-hardness of	CQ evaluation	
The previous reduc	ction immediately gives us the har	dness for combined
Theorem		
CQ evaluation is	<b>NP-hard</b> in combined complexit	у.
<i>Note:</i> in the previo	ous reduction, the interpretation d Hence, the reduction provides also	oes not depend on
<i>Note:</i> in the previo the actual graph.	ous reduction, the interpretation d Hence, the reduction provides also	oes not depend on
<i>Note:</i> in the previous the actual graph. for query complexion <b>Theorem</b>	ous reduction, the interpretation d Hence, the reduction provides also	oes not depend on the lower-bound
<i>Note:</i> in the previous the actual graph. for query complexion <b>Theorem</b>	ous reduction, the interpretation d Hence, the reduction provides also ty.	oes not depend on the lower-bound



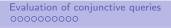




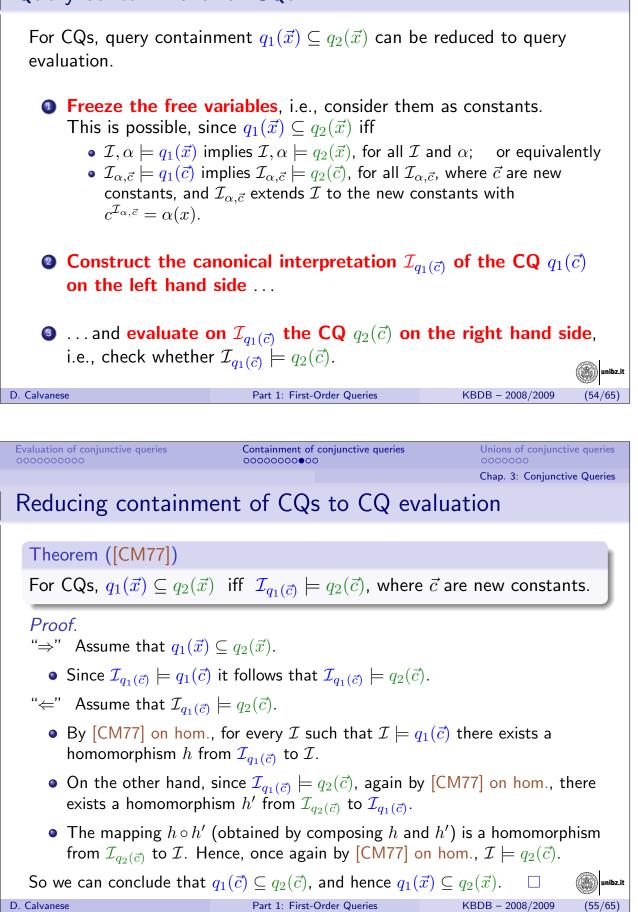
D. Calvanese

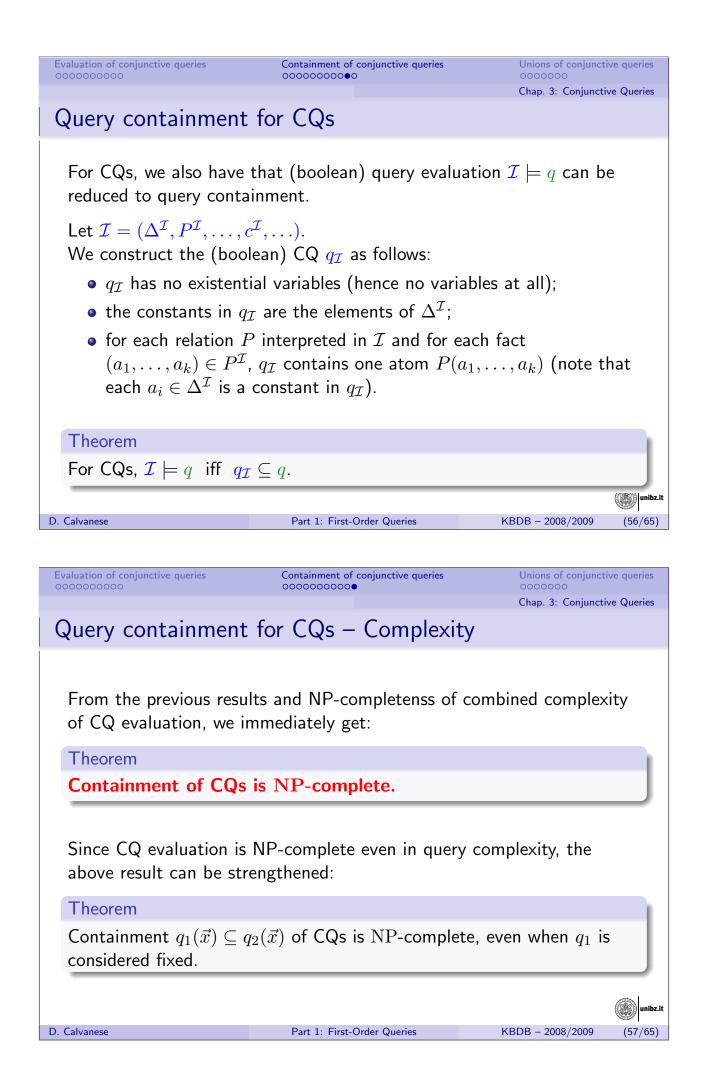
(51/65)

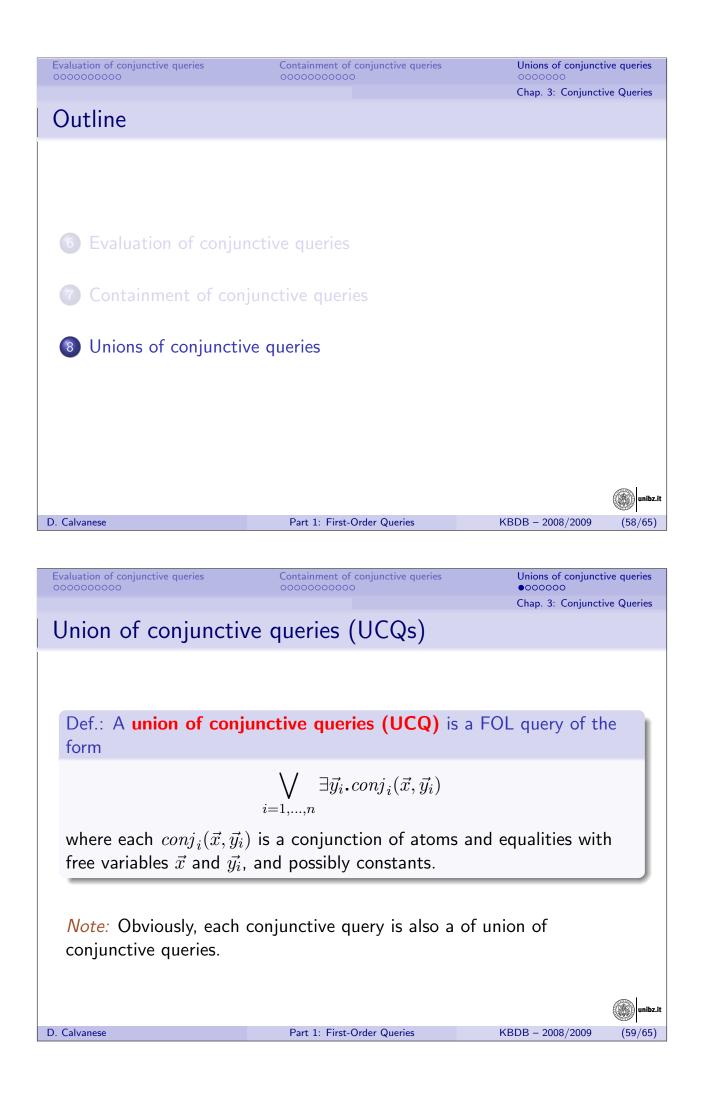
Evaluation of conjunctive queries	Containment of conjunctive queries	Unions of conjunctive queries
Canonical interpret	tation and (boolean) C	Chap. 3: Conjunctive Queries
Canonical interpret		Q CValuation
The previous result ca	n be rephrased as follows:	
(The recognition prob reduced to finding a	lem associated to) <b>query eva</b> <b>homomorphism</b> .	aluation can be
Finding a homomorph	icm botwoon two interpretation	one (aka relational
	ism between two interpretation wn as solving a <b>Constraint S</b>	(
Problem (CSP), a pro	oblem well-studied in AI – see	e also <mark>[KV98]</mark> .
		() unibz.it
D. Calvanese	Part 1: First-Order Queries	KBDB – 2008/2009 (52/65)
Evaluation of conjunctive queries	Containment of conjunctive queries	Unions of conjunctive queries
Query containmen	ł	Chap. 3: Conjunctive Queries
Query containmen	L	
Def.: Query contain	nent	
-	s $arphi$ and $\psi$ of the same arity,	$arphi$ is contained in $\psi$ ,
	all interpretations ${\mathcal I}$ and all a	ssignments $lpha$ we
have that	$\mathcal{I}, \alpha \models \varphi  implies  \mathcal{I}, \alpha \models \psi$	
(In logical terms: $\varphi \models$		
Note: Query containm	nent is of special interest in q	uery optimization.
Theorem		
For FOL queries, quer	y containment is undecidable	
Proof · Reduction from		
	n FOL logical implication.	unibz.it

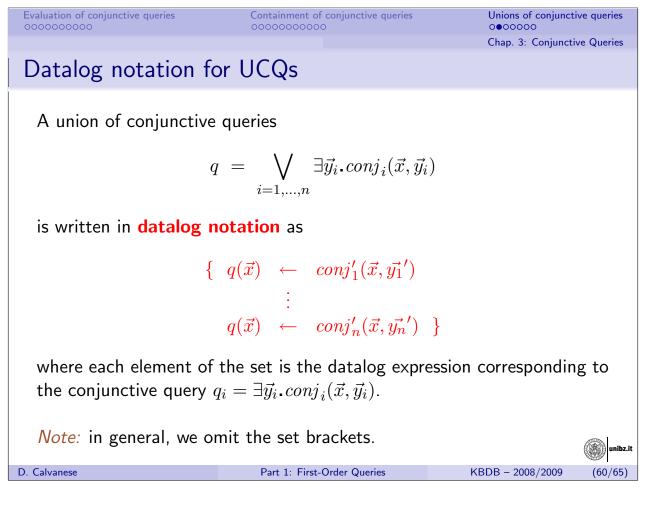


#### Query containment for CQs









Evaluation of conjunctive queries	Containment of conjunctive queries	Unions of conjunctive queries
		Chap. 3: Conjunctive Queries
Evaluation of UCQs		

From the definition of FOL query we have that:

$$\mathcal{I}, \alpha \models \bigvee_{i=1,\dots,n} \exists \vec{y_i}. conj_i(\vec{x}, \vec{y_i})$$

if and only if

$$\mathcal{I}, \alpha \models \exists \vec{y_i}. conj_i(\vec{x}, \vec{y_i}) \quad \text{for some } i \in \{1, \dots, n\}.$$

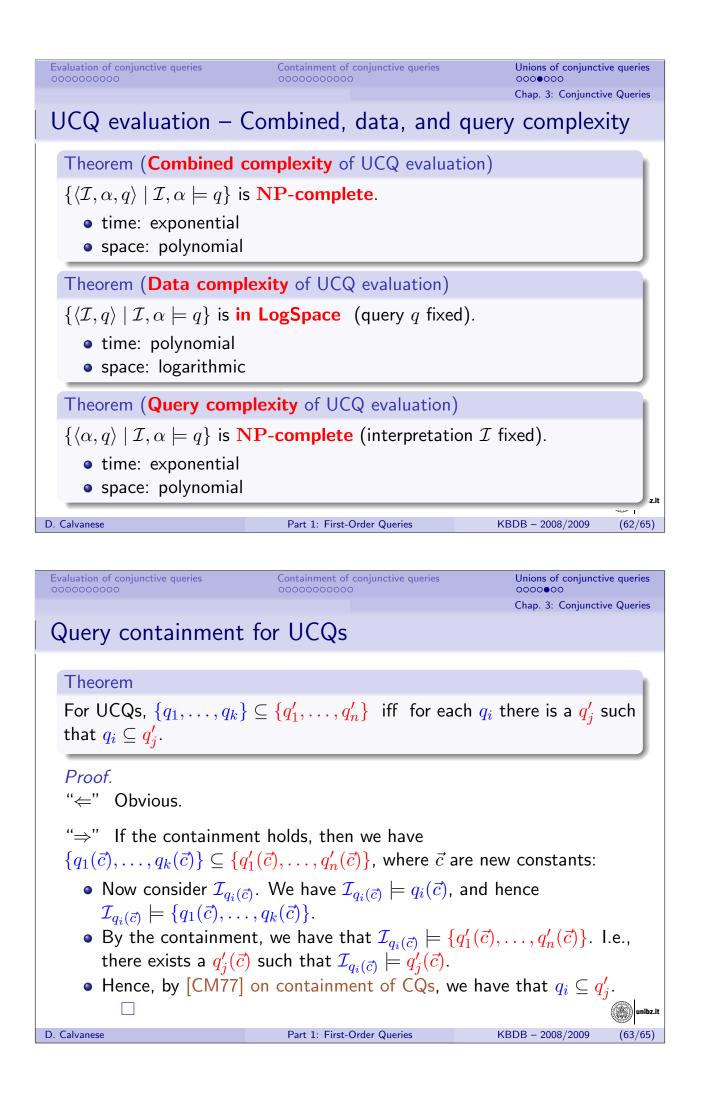
Hence to evaluate a UCQ q, we simply evaluate a number (linear in the size of q) of conjunctive queries in isolation.

Hence, evaluating UCQs has the same complexity as evaluating CQs.

D. Calvanese

KBDB - 2008/2009

(61/65)



Evaluation of	conjunctive queries Containment of conjunctive queries	Unions of conjunctive queries
Query	containment for UCQs – Complexity	Chap. 3: Conjunctive Queries
From t $\{q_1, \ldots$	the previous result, we have that we can check $\ldots, q_k\} \subseteq \{q_1', \ldots, q_n'\}$ by at most $k \cdot n$ CQ con mediately get:	
	inment of UCQs is NP-complete.	
		unibz.it
D. Calvanese	Part 1: First-Order Queries	KBDB – 2008/2009 (64/65)
Evaluation of	conjunctive queries         Containment of conjunctive queries           0         0000000000	Unions of conjunctive queries
Refere	ncoc	Chap. 3: Conjunctive Queries
Refere	lices	
[CM77]	A. K. Chandra and P. M. Merlin. Optimal implementation of conjunctive queries in rela In <i>Proc. of the 9th ACM Symp. on Theory of Compu</i> 77–90, 1977.	
[KV98]	P. G. Kolaitis and M. Y. Vardi.	
	Conjunctive-query containment and constraint satisface In Proc. of the 17th ACM SIGACT SIGMOD SIGART Database Systems (PODS'98), pages 205–213, 1998.	
[Var82]	M. Y. Vardi. The complexity of relational query languages. In <i>Proc. of the 14th ACM SIGACT Symp. on Theory</i> <i>(STOC'82)</i> , pages 137–146, 1982.	of Computing
		() unibz.it
D. Calvanese	Part 1: First-Order Queries	KBDB - 2008/2009 (65/65)