Advanced Data Modeling

Steffen Staab
with
Simon Schenk

UNIVERSITÄT KOBLENZ · LANDAU

Organizational Issues

Start of Excercise: Tuesday 15.04.08, 8.15 hrs. Room

No lecture on 2008-04-21 and no excercise on 2008-04-22 .

Slides on the web page/Klips.

Lecture INSS02 is Part of the „Schwerpunkt“ Data & Knowledge Engineering in the Master‘s Programme of Computer Science

Also eligible as Wahl- / Wahlpflicht in the Bachelor/Master

Examinations

Admittance to examination:

Present three times in the excerises (Übungen)

Exam:

Oral exam.

Contact the secretary Ms Werger end of June.

Structure of the lecture

- Relational data model;
- Deductive data model;
- Recursive definitions and their semantics;
- Query answering;
- Integrity constraints;
- Complex values;
- Object-oriented and object-relational data model;
- Simple deductive object-oriented data model;
- Unpredictable.

Deductive Databases

- evolved during the 1980s, based on the ideas developed in **relational databases** and **logic programming**.

- developed with the aim of **increasing the expressive power** of relational query languages, and in particular in connection with the inability of the latter to **express recursive queries**.

Query languages

- navigational (early DBMS);
- declarative (relational DBMS).

Logic tried to solve problems similar to those arising in foundations of databases:

- how to formalize the application world (**language**);
- How to express its properties (**semantics, model theory**);
- How to reason about these properties (**proof theory**).

Logic can handle in a **uniform framework**

- recursive definitions;
- integrity constraint;
- deduction, induction and abduction;
- Models for complex values . . .

Informal overview of deductive databases

- Extentionally defined relations.
- Intentionally defined relations.
- Integrity constraints.
- Recursion.
- Complex values.

Extensionally defined relations

Extensional definition:

by explicit enumeration of all tuples in the relation.

("Maier", "Mozartstrasse", 678);

…

("Schmidt", "Raiffeisenstrasse", 857);

…

Extensionally defined relation

In deductive databases we use the language of first-order logic. and represent this relation by a set of **facts**:

entry(”Maier”, ”Mozartstrasse”, 678);

…

entry(”Schmidt”, ”Raiffeisenstrasse”, 857);

…

Extensional database

The **extensional database** defines relations by sets of facts, for example

hasHighestDegree(”Maier”, BSc);

hasHighestDegree(”Schmidt”, MSc);

…

higherDegree(MSc,BSc);

…

Analogue of **tables** in relational databases.

Suppose we want to define a relation **personWithHigherDegree** among persons:

Person *A* has higher degree than person *B* **if** the highest degree of *A* is higher than the highest degree of *B*.

Extensional definition

personWithHigherDegree(”Schmidt”,”Maier”).

personWithHigherDegree(”Maier”,”Kunz”).

…

is dangerous

(**too large**, may become **inconsist** after updates).

Rephrase

For each pair of people *A*, *B*,
A has higher degree than *B* **if**
the highest degree of *A* is *DA* **and**
the highest degree of *B* is *DB* **and**
DA is a higher degree than *DB*.

Clause (rule)

```
personWithHigherDegree(A,B) :-      % head of the clause

hasHighestDegree(A,DA),             % body
hasHighestDegree(B,DB),             % of the
higherDegree(DA,DB).                % clause
```


Steffen Staab
staab@uni-koblenz.de

```
SELECT
    D1.person, D2.person
FROM
    hasHighestDegree D1,
    hasHighestDegree D2,
    higherDegree
WHERE
    D1.degree = higherDegree.higher AND
    D2.degree = higherDegree.lower
```

The relation personWithHigherDegree holds between objects
A, *B*
if
the relation hasHighestDegree holds between objects *A*, *DA*
and
the relation asHighestDegree holds between objects *B*, *DB*
and
the relation higherDegree holds between objects *RA*,*RB*.

For all objects **A, B, DA, DB**

the relation personWithHigherDegree holds between objects

A, B

if

the relation hasHighestDegree holds between objects ***A, DA***

and

the relation asHighestDegree holds between objects ***B, DB***

and

the relation higherDegree holds between objects ***RA,RB***.

Constants

subordinate(*O*, president) :- officer(*O*).

Here *O* is a variable, while president is a **constant**.

How to say this syntactically?

- Different conventions:
  - Possibility 1: All variables are explicitly quantified
  - Possibility 2: Variables are implicitly quantified (universally or existentially – needs to be agreed by convention)

    Sets of variables and constants are defined as such

Disjunction

How to express **every human is either a woman or a man**?

human(*A*) :-
    man(*A*).

human(*A*) :-
    woman(*A*).

How to express that every doctor has the same qualification as Doctor No, with the exception of Doctor No himself.

```
sameAs(A,A) :- Object(A).

sameQualification(A,B) :-
    hasHighestDegree(A, D),
    hasHighestDegree(B, D),
    notSameAs(A,B).

hasHighestDegree(DrNo, PhD).
```

Use negation:

```
sameQualification(A,B) :-
    hasHighestDegree(A, D),
    hasHighestDegree(B, D),
    not SameAs(A,B).
```

Negation is handled used the closed world assumption.

likes(*x*, *y*), not likes(*y*, *x*).

sameQualification(DrNo, *y*).