Towards Ontology-based OLAP

Datalog-based Reasoning over Multidimensional Ontologies

Bernd Neumayr, Stefan Anderlik, Michael Schrefl

DOLAP 2012, Maui, Hawaii, USA, November 2nd, 2012

Supported by the Austrian Ministry of Transport, Innovation, and Technology in program FIT-IT Semantic Systems and Services under grant FFG-829594: Semantic Cockpit: an ontology-driven, interactive business intelligence tool for comparative data analysis.
Agenda

■ General Idea
  □ Problem/Motivation
  □ Multidimensional Ontologies (MDOs)
  □ Architecture: Ontology-based OLAP

■ Technical Details (very short, see paper)
  □ MDO Metamodel
  □ Mapping to Datalog
  □ Datalog-based Reasoning over MDO Concepts

■ Ongoing and Future Work
Problem

How to

reuse,
maintain,
understand,
share and
organize

large sets of multidimensional queries?

(we focus on boolean predicates of selection clauses)
**Problem**

Running Example

**Query 1:**
Treatment(insurant, prescribingDoctor, treatingDoctor;
insurant.age > 65, insurant.cityOfResidence.inhabitants > 50000).costs

**Query 2:**
Treatment(insurant, prescribingDoctor, treatingDoctor;
insurant.age > 65, insurant.cityOfResidence.inhabitants > 100000, treatingDoctor.cityOfPractice.inhabitants > 50000).costs
Partial Solution

- Define business terms and use them for querying
- Implement business terms as
  - SQL: Views
  - MDX: Named Sets
- Lack of:
  - reasoning support: consistency checking, subsumption checking
  - organization of business terms
Partial Solution

**Running Example**

```
Query1:  Treatment(insurant, prescribingDoctor, treatingDoctor,;
         insurant.old, insurant.cityOfResidence.mediumToLargeCity)
         .costs

Query2:  Treatment(insurant, prescribingDoctor, treatingDoctor,;
         insurant.old, insurant.cityOfResidence.largeCity, treatingDoctor.cityOfPractice.mediumToLargeCity)
         .costs
```

- mediumToLargeCity
  - == inhabitants > 50000
- largeCity
  - == inhabitants > 100000
- old
  - == age > 65
Ontologies to the Rescue

■ Ontologies:
  - formal conceptual domain models with
  - richly defined concepts organized in subsumption hierarchies

■ use ontologies ‘as high-level, conceptual view over data repositories’ for querying, as in OBDA:

Multidimensional Ontologies (MDOs)

- Conceptual Multidimensional Model (Schema + Individuals)
  + Defined Concepts (Views)
  + Subsumption Reasoning over Defined Concepts

- Separation between Primitive Concepts (Schema) and Defined Concepts (Views) as in:
  
Open-world MDO vs. closed-world DWH

Other than in OBDA we assume - as a prerequisite for correct aggregation results - that data in a data warehouse is always complete.

A Grounded MDO can be regarded as a model (valid interpretation) of the MDO. Some of the Datalog rules and constraints defining structure and semantics of MDOs only apply to Grounded MDOs.
Architecture: Ontology-based OLAP

- Business Analyst
- Ontology Based OLAP Frontend
  - Multi-Dimensional Ontology (MDO) Engine
    - MDO Reasoner
    - MDO Manager
  - MDO-DWH Mapper
- DWH
- BI Analysis Graphs
  - MDO Measures
    - MDO
      - Multidimensional
        - Hierarchical
        - Flat
    - Mapping & Grounding
Why don’t we use OWL for defining MDOs?

■ Worldview of business analysts difficult to represent in OWL:
  □ Business analyst: focus on highly-aggregated numbers describing some abstract points in a multidimensional space.
  □ Blurring between ABox and TBox: dimension members (nodes) and cells are individuals and classes, thus, classes of dimension members or cells are metaclasses

■ OWL does not exploit structural specificities of MDOs

■ Future work: MDOs in OWL
  Extend/Restrict OWL for MDOs or build MDOs on top of OWL?

■ OWL ontologies in the SemCockpit project:
Duality of Nodes

Node as Individual

provinceOfResidence

cityOfResidence

insurant

upperAustria

tyrol

vienna
Duality of Nodes

Node as Class

provinceOfResidence

cityOfResidence

insurant

upperAustria

tyrol

vienna
Layered Approach to MDOs

- Decompose multidimensional schema to reusable chunks (primitive concepts) at different layers:
  - Flat: Datatype, Entity types
  - Hierarchical: Hierarchies with Levels
  - Multidimensional: MD-Base, MD-Space

- Different kinds of defined concepts (defined local to primitives)
  - Flat: Datarange, Entity Concept
  - Hierarchical: H-Concept
  - Multidimensional: MD-Concept

- Interpreted by different kinds of individuals
  - Flat: Value, Entity
  - Hierarchical: Node
  - Multidimensional: Point
DFM as starting point

- **provinceOfPractice**
- **inhabitants**
- **cityOfPractice**
- **prescribing**
- **Doctor**
- **popDensity**
- **provinceOfPractice**
- **inhabitants**
- **cityOfPractice**
- **treating**
- **Doctor**

**Treatment**
- **quantity**
- **costs**

- **insurant**
- **age**
- **cityOfResidence**
- **inhabitants**
- **provinceOfResidence**
- **popDensity**
Decompose DFM to reusable chunks: MDO Schema

- **Province**
  - popDensity : Number

- **Doctor**

- **Municipality**
  - inhabitants : Number

- **Insurant**
  - age : Number

Diagram:
- CityOfPractice
  - Province
  - Doctor
  - Municipality
  - Insurant
  - Treatment
    - prescribingDoctor
    - treatingDoctor
    - insurant

- CityOfResidence
  - Province

- ProvinceOfPractice
  - Doctor

- ProvinceOfResidence
  - Municipality

- Insurant
  - Doctor
MDO Schema
Language Constructs

Attribute

Entitytype

Datatype

Level

Hierarchy

Dimension

MDBase

Granularity

Running Example

Bernd Neumayr et al.
Towards Multidimensional Ontologies
MDO Individuals

Running Example

<table>
<thead>
<tr>
<th>Entity</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>linz:Municipality</td>
<td>inhabitants = 191000</td>
</tr>
</tbody>
</table>

Entity

Point

Value

Node
Flat Concepts
Definition

ENTITYTYPE municipality

ENTITYCONCEPT largeCity
== (inhabitants => (from:100000))

ENTITYCONCEPT mediumToLargeCity
== (inhabitants => (from:50000))
Hierarchical Concepts

**Definition**

HIERARCHY insurant

HCONCEPT livingInRuralArea
   == (provinceOfResidence=> rural)

HCONCEPT adultInMediumToLargeCity
   == (cityOfResidence => mediumToLargeCity, 
      insurant => adult )

HCONCEPT oldInLargeCity
   == (cityOfResidence=> largeCity, 
      insurant => old)

HCONCEPT oldInLargeCityInRuralArea
   == ( provinceOfResidence=>rural, 
      cityOfResidence=> largeCity, 
      insurant => old)
Interpretation of Hierarchical Concepts

provinceOfResidence

cityOfResidence

insurant

livingInRuralArea

oldInLargeCityInRuralArea

upperAustria

tyrol

vienna
Hierarchical Concepts
Subsumption Hierarchies

Running Example

- Doctor
- cityOfPractice
- provinceOfPractice
- popDensity
- inhabitants
- treating
- Doctor
- provinceOfPractice
- popDensity
- provinceOfPractice
- cityOfPractice
- doctor
- age
- Municipality
- mediumToLargeCity
- smallVillage
- largeCity
- Province
- urban
- rural
- Insurant
- T
- adult
- child
- age
- cityOfResidence
- provinceOfResidence
- popDensity
- municipality
- mediumToLargeCity
- smallVillage
- largeCity
- provinceOfResidence
- popDensity
- provinceOfResidence
- cityOfPractice
- prescribing
- Doctor
- treating
- Doctor
- provinceOfPractice
- popDensity
- provinceOfPractice
- cityOfPractice
- quantity
- costs
- Insurant
- T
- adult
- livingInRuralArea
- old
- child
- InRuralArea
- smallVillage
- mediumToLargeCity
- largeCity
- oldInLargeCity
- adultInMediumToLargeCity
- adultInMediumToLargeCityInRuralArea
- oldInLargeCityInRuralArea
- urban
- rural
- mediumToLargeCity
Multidimensional Concepts

Definition

MDBASE Treatment

MDCONCEPT forAdultInsurantsLivingInMediumToLargeCities

== (insurant => adultInMediumToLargeCity)

MDCONCEPT forOldInsurantsLivingInLargeCitiesBy-DoctorsPractisingInMediumToLargeCities

== (insurant => oldInLargeCity,
   treatingDoctor => practisingInMediumToLargeCity)
Multidimensional Concepts
Subsumption Hierarchy

Running Example

- Doctor
  - practisingIn
    - MediumToLargeCity

- Province
  - T
    - urban
    - rural
- Municipality
  - T
    - mediumToLargeCity
    - smallVillage
    - largeCity
- provinceOfPractice
- cityOfPractice
- townOfPractice
- provinceOfResidence
- cityOfResidence
- townOfResidence
- insurant
- age
- adult
- child
- old
- quantity
- costs
- Treatment
- forAdultInsurantsLivingInMediumToLargeCities
- forOldInsurantsLivingInLargeCities
- ByDoctorsPractisingInLargeCities
- popDensity
- inhabitants
- provinceOfPractice
- cityOfPractice

- smallVillage
- mediumToLargeCity
- largeCity

- oldInLargeCity
- livingInRuralArea
- oldInLargeCityInRuralArea
Agenda

■ General Idea
  □ Problem/Motivation
  □ Multidimensional Ontologies (MDOs)
  □ Architecture: Ontology-based OLAP

■ Technical Details (very short, see paper)
  □ MDO Metamodel
  □ Mapping to Datalog
  □ Datalog-based Reasoning over MDO Concepts

■ Ongoing and Future Work
MDO Metamodel

Canonical form of MDOs

Defined Concepts

Flat

Hierarchical

Multidimensional

Schema

Individuals

Datatype

Attribute

Entitytype

Entity

Datarrange

HConcept

Hierarchy

Dimension

Level

Node

Granularity

Point

MDConcept

 MDBase

MDSpace

Form

0

0.1

0

0.1

coarsest

finest

0.1

Parents

Canonical form of MDOs
MDO Metamodel: Derived Information
MDO in Datalog

- Why Datalog?
  - concise syntax
  - simple semantics
  - very mature reasoning procedures

- We use standard Datalog with
  - constraints (rules without heads)
  - stratified negation as failure
MDO in Datalog

- MDO language constructs $\rightarrow$ Datalog predicates
- Structure and Semantics of MDO $\rightarrow$ Datalog rules / constraints
  (some rules only apply to closed-world Grounded MDOs!)
- User-defined MDO $\rightarrow$ Datalog constants and facts

- Stratification along MDO layers: flat, hierarchical, multidim
**MDO in Datalog**

**Running Example**

**ENTITYTYPE** city

**ATTRIBUTE** inhabitants \(\rightarrow\) int

**ENTITY** linz

(inhabitants \(\rightarrow\) 191000)

```
et(et_city).
a(a_city_inhabitants, dt_int, et_city).
e(et_city_linz, et_city).
av(et_city_linz, a_city_inhabitants, 191000).
```
MDO Metamodel: Mapping to Datalog (EDB)
MDO Metamodel: Mapping to Datalog (EDB)

(order of single-valued properties)
MDO Metamodel: Mapping to Datalog (IDB)
Interpretation of Hierarchical Concepts (Concept Membership, Instance Checking)

A hierarchical concept is interpreted as a subset of the nodes of the hierarchy it belongs to. A node is member of a hierarchical concept if it is a descendant of the root node and for every ancestor level restriction it or one of its ancestors fulfills the restriction. If a node is member of a hierarchical concept, then all its descendant nodes are also members of the hierarchical concept.

\[
\text{not}_n\_\text{of} \ (N,HC) \ :- \ n(N,_,_), \ lr(HC,Lx,EC), \ n(Nx,Lx,Ex), \\
\hspace{1cm} \text{n}_h\_\text{tr} \ (N,Nx), \ \text{not} \ \text{e}_\text{of}(Ex,EC).
\]

\[
\text{not}_n\_\text{of} \ (N,HC) \ :- \ n(N,_,_), \ lr(HC,Lx,EC), \ \text{not} \ \text{n}_h\_\text{tr}_l \ (N,Lx).
\]

\[
\text{n}_h\_\text{tr}_l \ (N,L) \ :- \ \text{n}_h\_\text{tr} \ (N,Nx), \ n(Nx,L,\_).
\]

\[
\text{n}_\text{of}(N,HC) \ :- \ n(N,L,\_), \ l(L,\_,H), \ hc(HC,H,Nt), \ \text{n}_h\_\text{tr} \ (N,Nt), \\
\hspace{1cm} \text{not} \ \text{not}_n\_\text{of} \ (N,HC).
\]
Subsumption of Hierarchical Concepts

A hierarchical concept is subsumed by another hierarchical concept if they belong to the same hierarchy, the root node of the subsumed concept is the same or a descendant of the root node of the subsuming concept, and for every level restriction of the subsuming concept there is the same or a finer restriction in the subsumed concept or the restriction is fulfilled by the root node of the subsumed concept.

bernd Neumayr et al. Towards Multidimensional Ontologies

hc_s(HC,HCx) :- hc(HC,H,N), hc(HCx,H,Nx), n_h_tr(N,Nx),
                               not not_hc_s(HC,HCx).
not_hc_s(HC,HCx) :- hc(HC,_,_), lr(HCx,L,ECx),
                               not hc_s_lec(HC,L,ECx).
hc_s_lec(HC,L,ECx) :- lr(HC,L,EC), ec_s(EC,ECx).
hc_s_lec(HC,L,ECx) :- hc(HC,_,N), n(Ny,L,Ey), n_h_tr(N,Ny),
                               e_of(Ey,ECx).
Agenda

- General Idea
  - Problem/Motivation
  - Multidimensional Ontologies (MDOs)
  - Architecture: Ontology-based OLAP

- Technical Details (very short, see paper)
  - MDO Metamodel
  - Mapping to Datalog
  - Datalog-based Reasoning over MDO Concepts

- Ongoing and Future Work
Ongoing and Future Work

- Extending the MDO formalism
  - Concept definitions over (aggregated) measures
  - Built-in predicates (another kind of primitives)
- MDO in SQL, MDO in OWL
- Mapping between MDO and DWH
- Direct semantics for MDO and proof that our Datalog rules are sound & complete
- Apply MDO concepts for defining
  - Complex-derived measures
  - BI Analysis graphs (Neuböck, 2012)
  - Comparative data analysis

Thanks you for your attention!
References

- Bernd Neumayr, Michael Schrefl, Konrad Linner: *Semantic Cockpit: An Ontology-Driven, Interactive Business Intelligence Tool for Comparative Data Analysis*. ER Workshops 2011: 55-64