Chapter 4: Ontologies and Knowledge Graphs

Leopoldo Bertossi

Ontologies and Connections

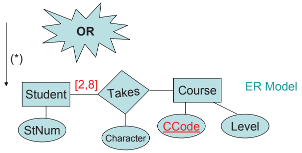
- Ontologies have their origin in AI, and a long tradition
- With some roots and inspirations in data management, e.g. Conceptual Modelling (think of ER models)
- Many applications, in particular in Business Intelligence
- Many spin-offs: Semantic Web Languages, Graph Databases, Knowledge Graphs, ...
- General idea: Represent knowledge in restricted languages

  Symbolic (logic-based) or graphical languages

  For a good balance between expressive power and computational performance
- In general, Ontologies are Knowledge Bases that describe a domain in terms of Concepts (or Entities) and Relationships between Concepts

ER Models and Ontologies

- The common route of an Entity/Relationship (ER) Model
  1. Want to model an outside reality (OR), e.g. a business environment
  2. Identify relevant concepts (entities, classes) and relationships
  3. Draw a diagram depicting them
  4. Analyze the (graphical) model
  5. Identify some key elements
  6. Transform ER model into a relational model (think of tables)
  7. Etc.



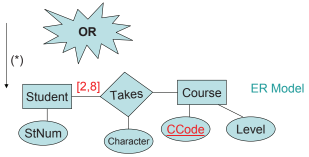
- Here: *Students*, *Course* are entities, *Takes* is relationship, *StNum*, etc. are attributes
- The elements in red are semantic constraints, that help keep the correspondence (*) between the OR and the model

- The model tells us that:

  (a) Students take courses,
  and courses are taken by students

  (b) Students have student numbers

  (c) Courses have codes and levels



- The attribute *Character* hanging from *Takes* is about the relationship

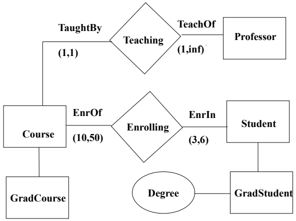
  A student takes a course as mandatory, elective, extracurricular, ...

- The label `[2,8]` on the link tells us that "every student takes between 2 and 8 courses": A cardinality constraint

- The underlined course code, `CCode`, indicates that the code fully identifies the course: A key constraint

  More precisely, any two courses that have the same code must have the same values for all the attributes (for courses)

Exercise: Discuss the (extended) ER model below



The link between *GradCourse* and *Course* is an IS-A link (subclass, subconcept, subentity link)

IS-A links have a predefined and fixed meaning, and enables inheritance of attributes, e.g attributes of *Course* by *GradCourse*

For possible extensions of ER models, we added Links or *Roles*: *TeachOf* to tealationships

We can say, e.g. *TeachOf(bertossi,kr4ai)*, and *TaughtBy(kr4ai,bertossi)*

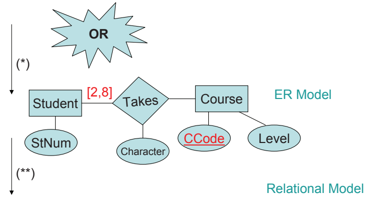
Exercise: Consider the following situation:

The FunToys company produces and sells a large number of toys in a chain of stores, over a wide territory. A main business goal for this company could be to understand the impact of promotions on sales, that is, how promotions influence product sales and to what extent promotions are profitable. Another important business goal could be the analysis of the warehouse process, where inventory levels should be measured monthly, for each product and warehouse controlled by the company. It follows that possible dimensions of the FunToys data warehouse application are Product, Store, Warehouse, Time, and Promotion. The Product dimension may be organized into levels such as item (whose members are products such as Disneys Dinosaur and Duplo Pooh), product-line (containing members like Mattels Disney and Lego Duplo), brand (Mattel and Lego), category (Popular Characters and Blocks), and department (Action Figures and Blocks). The elements of the Time dimension describe days over a period of time; this dimension may be organized into the levels day, month, quarter, year, and season. A member of the level day might be Feb 27, 2001. Members of the level day can be grouped to members of the level month, but also to members of the level season (e.g., Carnival). Descriptions of the item level might be its name and code.

For the FunToys company, a possible fact is a daily sale. This fact can be analyzed with respect to the day of the sale, the product sold, the store of the sale, and the promotion applied to the daily sale. The measurements made for each daily sale could include the number of units sold, the income and the cost. Thus, a data cube Sales can be used to describe daily information about the items sold by the stores of the chain. An instance of this data cube can state the fact that on Feb 27, 2001 the store Colosseum has sold 2 pieces of Duplo Pooh, applying a Carnival 2001 Promotion, for a corresponding gross income of 19.98 Euros against a cost of 14.98 Euros. In the warehouse process, measurable facts are the inventory levels, to be measured, for instance, monthly, for each product and warehouse. They can be modeled by means of a data cube Inventory. The measurements made for each monthly inventory could include the inventory level (the quantity in stock at the end of the month), the quantity shipped during the month, and the value at cost of the quantity in stock.

Produce an ER model, including a diagram. In this case, the data model could consist of two ER models with shared components

- Usual next step in data management: produce a relational model from the ER model, also a model of OR; a logical model



Relational Schema: Student(StNum,Address), Course(<u>CCode</u>,Level,MaxReg), Takes(StNum,CCode,Character) + ICs

- We can use predicate logic
- Relational ICs become part of the model

  With some coming from the ER model, and other new, e.g. referential constraints from *Takes* to the other two
- The ER model can be (and commonly is) discarded after the relational DB is created and populated
- But the ER model contains much semantic information (meaning)
- It could become metadata: data about data
- A semantic layer that is closer to OR and what users understand To be used in combination with the DB

- ER model is a diagram

  Not suitable for reasoning or query answering (QA)

- How to combine a diagrammatic model with a logical model?

  How to realize the integration?

- The idea is to reconstruct the ER model as a knowledge-base, more precisely, a formal ontology

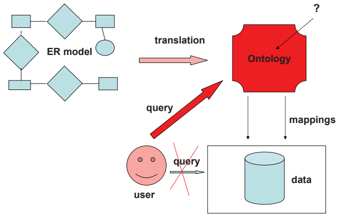
  Written in some restricted language of predicate logic

- After that a user can interact directly with the ontology

  The latter interacts with the DB (the data source)

- Query the ontology

  The ontology interacts with DB

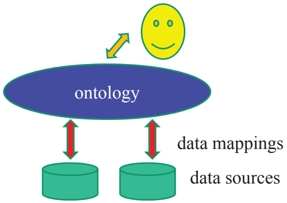


Ontologies for Data Integration

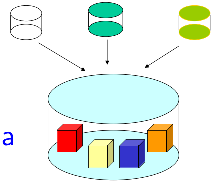
- Similar idea can be used for virtual data integration

  Access several data sources via a single encompassing layer
- Creating mappings between sources and common semantic layer

  With a common language
- The user interacts with the ontology
- The "implementation" takes care of accessing underlying data



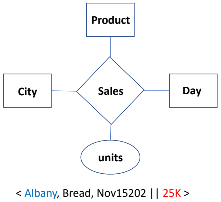
- This is all part of OBDA: Ontology-Based Data Access
- Alternative to materialized data integration
- New integrated DB created, with new schema
- Typically a data warehouse (DWH)
- Common in business applications



- Example: Multidimensional Databases (Data Warehouses)
- Quite common and useful in Business Intelligence
- For materialized data integration, business understanding, business analytics, and decision support
- Start with a ER model

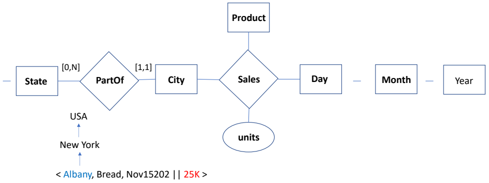
  Three dimensions: City, Product, Time

  Numerical attribute: units sold



- Dimensions can be developed into hierarchies of categories
- Different levels of abstraction

  Aggregation along dimensions



- Can be represented as an ontology