Lecture 06:
E/R Diagrams

Monday, October 9, 2006
Outline

• E/R diagrams (Chapter 2)
• From E/R diagrams to relations (3.2, 3.3)

• Wednesday:
  – Project
• Friday:
  – Functional dependencies, normal forms:
  – Warning: this is hard, come to class
Database Design

• Why do we need it?
  – Agree on structure of the database before deciding on a particular implementation.

• Consider issues such as:
  – What entities to model
  – How entities are related
  – What constraints exist in the domain
  – How to achieve good designs

• Several formalisms exist
  – We discuss E/R diagrams
Entity / Relationship Diagrams

Objects → entities
Classes → entity sets

Attributes are like in ODL.

Relationships: like in ODL except

- first class citizens (not associated with classes)

- not necessarily binary
Keys in E/R Diagrams

• Every entity set must have a key
What is a Relation?

- A mathematical definition:
  - if A, B are sets, then a relation R is a subset of A × B

- A={1,2,3}, B={a,b,c,d},
  A × B = {(1,a),(1,b), . . ., (3,d)}
  R = {(1,a), (1,c), (3,b)}

- makes is a subset of Product × Company:
Multiplicity of E/R Relations

- one-one:

- many-one

- many-many
What does this say?
Multi-way Relationships

How do we model a purchase relationship between buyers, products and stores?

Can still model as a mathematical set (how?)
Arrows in Multiway Relationships

**Q:** what does the arrow mean?

**A:** a given person buys a given product from at most one store
Arrows in Multiway Relationships

Q: what does the arrow mean?

A: a given person buys a given product from at most one store AND every store sells to every person at most one product
Arrows in Multiway Relationships

Q: How do we say that every person shops at at most one store?

A: cannot. This is the best approximation. (Why only approximation?)
Converting Multi-way Relationships to Binary

- **Purchase**
  - **ProductOf**
    - **Product**
  - **StoreOf**
    - **Store**
  - **BuyerOf**
    - **Person**

- **date**
3. Design Principles

What’s wrong?

Product  Purchase  Person

Country  President  Person

Moral: be faithful!
Design Principles: What’s Wrong?

Moral: pick the right kind of entities.
Design Principles: What’s Wrong?

Moral: don’t complicate life more than it already is.
From E/R Diagrams to Relational Schema

- Entity set $\rightarrow$ relation
- Relationship $\rightarrow$ relation
**Entity Set to Relation**

**Product**

- **name**
- **category**
- **price**

**Product**(name, category, price)

<table>
<thead>
<tr>
<th>name</th>
<th>category</th>
<th>price</th>
</tr>
</thead>
<tbody>
<tr>
<td>gizmo</td>
<td>gadgets</td>
<td>$19.99</td>
</tr>
</tbody>
</table>
Relationships to Relations

**Makes** *(product-name, product-category, company-name, year)*

<table>
<thead>
<tr>
<th>Product-name</th>
<th>Product-Category</th>
<th>Company-name</th>
<th>Starting-year</th>
</tr>
</thead>
<tbody>
<tr>
<td>gizmo</td>
<td>gadgets</td>
<td>gizmoWorks</td>
<td>1963</td>
</tr>
</tbody>
</table>

*(watch out for attribute name conflicts)*
Relationships to Relations

No need for Makes. Modify **Product:**

<table>
<thead>
<tr>
<th>name</th>
<th>category</th>
<th>price</th>
<th>StartYear</th>
<th>companyName</th>
</tr>
</thead>
<tbody>
<tr>
<td>gizmo</td>
<td>gadgets</td>
<td>19.99</td>
<td>1963</td>
<td>gizmoWorks</td>
</tr>
</tbody>
</table>
Multi-way Relationships to Relations

Product
- name
- price

Purchase
- prodName
- stName
- ssn

Store
- name
- address

Person
- name
- ssn

Purchase(prodName, stName, ssn)
Modeling Subclasses

Some objects in a class may be special
• define a new class
• better: define a subclass

So --- we define subclasses in E/R
Subclasses

Product

- name
- category
- price

ISA:
- Software Product
  - platforms
- Educational Product
  - Age Group
Understanding Subclasses

• Think in terms of records:
  – Product
  – SoftwareProduct
  – EducationalProduct
Subclasses to Relations

<table>
<thead>
<tr>
<th>Name</th>
<th>Price</th>
<th>Category</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gizmo</td>
<td>99</td>
<td>gadget</td>
</tr>
<tr>
<td>Camera</td>
<td>49</td>
<td>photo</td>
</tr>
<tr>
<td>Toy</td>
<td>39</td>
<td>gadget</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Name</th>
<th>platforms</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gizmo</td>
<td>unix</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Name</th>
<th>Age Group</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gizmo</td>
<td>toddler</td>
</tr>
<tr>
<td>Toy</td>
<td>retired</td>
</tr>
</tbody>
</table>
Difference between OO and E/R inheritance

- **OO**: classes are disjoint (same for Java, C++)
Difference between OO and E/R inheritance

- E/R: entity sets overlap

![Diagram showing entity sets and overlap]

- E/R: entity sets overlap

- E/R: entity sets overlap

- E/R: entity sets overlap
No need for multiple inheritance in E/R

We have three entity sets, but four different kinds of objects.
Modeling UnionTypes With Subclasses

FurniturePiece

Person  Company

Say: each piece of furniture is owned either by a person, or by a company
Modeling Union Types with Subclasses

Say: each piece of furniture is owned either by a person, or by a company

Solution 1. Acceptable, imperfect (What’s wrong ?)
Modeling Union Types with Subclasses

Solution 2: better, more laborious
Constraints in E/R Diagrams

Finding constraints is part of the modeling process. Commonly used constraints:

**Keys:** social security number uniquely identifies a person.

**Single-value constraints:** a person can have only one father.

**Referential integrity constraints:** if you work for a company, it must exist in the database.

**Other constraints:** peoples’ ages are between 0 and 150.
Keys in E/R Diagrams

Underline:

No formal way to specify multiple keys in E/R diagrams
Single Value Constraints

makes

v. s.

makes
Referential Integrity Constraints

Each product made by at most one company. Some products made by no company.

Each product made by *exactly* one company.
Other Constraints

Product \( \rightarrow \) makes \( <100 \) \( \rightarrow \) Company

What does this mean?
Weak Entity Sets

Entity sets are weak when their key comes from other classes to which they are related.

Notice: we encountered this when converting multiway relationships to binary relationships (last lecture)
Handling Weak Entity Sets

Convert to a relational schema (in class)