Introduction to Database Systems
CSE 444

Lecture 5: E/R Diagrams
Outline

• **E/R diagrams**
  – Sec. 4.1- 4.4

• **From E/R diagrams to relations**
  – Sec. 4.5 and 4.6
Database Design

• Why do we need it?
  – Need a way to model real world entities in terms of relations
  – Not easy to go from real-world entities to a database schema

• 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 exists
  – We discuss E/R diagrams
Database Design Process

Today
Data Modeling

Next lecture
Refinement
SQL Tables
Files

E/R diagrams
Conceptual Schema
Physical Schema
Relations
Conceptual Schema Design

Conceptual Model:

Relational Model: plus FD’s (FD = functional dependency)

Normalization: Eliminates anomalies
Entity / Relationship Diagrams

Objects → entities
Classes → entity sets

Attributes:
- address

Relationships:
- buys

- first class citizens (not associated with classes)
- not necessarily binary

This is an entity set
Product
Keys in E/R Diagrams

- Every entity set must have a key
- May be a multi-attribute key:
What is a Relation?

- A mathematical definition:
  - if A, B are sets, then a relation R is a subset of $A \times B$
- $A=\{1, 2, 3\}, \ B=\{a, b, c, d\}$,
  - $A \times B = \{(1, a), (1, b), \ldots, (3, d)\}$
  - $R = \{(1, a), (1, c), (3, b)\}$

- **makes** is a subset of **Product $\times$ Company**:

CSE 444 - Summer 2010
Multiplicity of E/R Relations

• one-one:

• many-one

• many-many
Multi-way Relationships

Product

Purchase

date

Person

Store
Converting Multi-way Relationships to Binary

Arrows are missing: which ones?
Design Principles

What's wrong?

Moral: be faithful to reality
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 → relation
- Relationship → relation
Entity Set to Relation

**Product** *(prod-ID, category, price)*

<table>
<thead>
<tr>
<th>prod-ID</th>
<th>category</th>
<th>price</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gizmo55</td>
<td>Camera</td>
<td>99.99</td>
</tr>
<tr>
<td>Pokemn19</td>
<td>Toy</td>
<td>29.99</td>
</tr>
</tbody>
</table>
CREATE TABLE Product ( 
  prod-ID CHAR(30) PRIMARY KEY, 
  category VARCHAR(20), 
  price double 
)
Relationships to Relations

Shipment (prod-ID, cust-ID, name, date)

<table>
<thead>
<tr>
<th>prod-ID</th>
<th>cust-ID</th>
<th>name</th>
<th>date</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gizmo55</td>
<td>Joe12</td>
<td>UPS</td>
<td>4/10/2010</td>
</tr>
<tr>
<td>Gizmo55</td>
<td>Joe12</td>
<td>FEDEX</td>
<td>4/9/2010</td>
</tr>
</tbody>
</table>
CREATE TABLE Shipment(
    name CHAR(30)
    REFERENCES Shipping-Co,
    prod-ID CHAR(30),
    cust-ID VARCHAR(20),
    date DATETIME,
    PRIMARY KEY (name, prod-ID, cust-ID),
    FOREIGN KEY (prod-ID, cust-ID)
    REFERENCES Orders
)
Relationships to Relations

\[
\text{Makes}(\text{product-name}, \text{product-category}, \text{company-name}, \text{year})
\]

<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)
No need here 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
- prod-ID
- price

Purchase

Person
- ssn
- name

Store
- name
- address

How do we represent that in a relation?
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

ISA

Software Product

ISA

Educational Product

name

category

price

platforms

Age Group

CSE 444 - Summer 2010
Understanding Subclasses

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

Product

<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>

Sw.Product

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

Ed.Product

<table>
<thead>
<tr>
<th>Name</th>
<th>Age Group</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gizmo</td>
<td>todler</td>
</tr>
<tr>
<td>Toy</td>
<td>retired</td>
</tr>
</tbody>
</table>

Other ways to convert are possible
See book sec 4.6

CSE 444 - Summer 2010
Difference between OO and E/R inheritance

- **OO**: classes are disjoint (same for Java, C++)

```
Product
  p1  p2  p3
SoftwareProduct
  sp1  sp2
EducationalProduct
  ep1  ep2  ep3
```
Difference between OO and E/R inheritance

- E/R: entity sets overlap
Difference between OO and E/R inheritance

No need for multiple inheritance in E/R

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

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: More faithful

![UML diagram]

- Person isa Owner
- Company isa Owner
- FurniturePiece ownedBy Owner
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:

- name
- category
- price

Product

Multi-attribute key v.s. Multiple keys

Not possible in E/R
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

![Diagram]

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.
Handling Weak Entity Sets

Convert to a relational schema

University(name)
Team(number,universityName,sport)
No need to represent affiliation separately