Entity-Relationship Model
HOW TO BUILD A DB APPLICATION

• Pick an application
• Figure out what to model (ER model)
  – Output: ER diagram
• Transform the ER diagram to a relational schema
• Refine the relational schema (normalization)
• Now ready to implement the schema and load the data!
Running Example

We want to store information about:

• companies and employees
  – Each company has a name, an address, and a CEO
  – Each company has a list of employees

• products manufactured by these companies
  – Each product has a name and a description
THE ER MODEL

• Gives us a language to specify
  – what information the DB must hold
  – what are the relationships among components of that information

• Proposed by Peter Chen in 1976

• What we will cover
  – basic stuff: entities, attributes, relationships
  – constraints
  – weak entity sets
  – design principles
Entities & Attributes

- **Entities**
  - objects distinguishable from other objects
  - described using a set of attributes

- **Attributes**
  - each has an atomic domain: string, integer, ...

- **Entity set**: a collection of similar entities
ENTITIES & ATTRIBUTES

Product
- price
- name
- category

Company
- name
- stockprice

Person
- name
- address
- ssn
RELATIONS

• A mathematical definition:
  – if $A, B$ are sets, then a relation $R$ is a subset of $A \times B$

• Example
  – $A = \{1, 2, 3\}$, $B = \{a, b, c, d\}$
  – $R = \{(1, a), (1, c), (2, c), (3, b)\}$
makes is a subset of Product \times Company
RELATIONSHIPS

Product

makes

Company

buys

employs

Person

name

address

ssn

price

category

name

name

stockprice

makes

employs

buys

name

address

ssn
MULTIPLICITY OF RELATIONSHIPS

• one-one

• many-one

• many-many
MULTIPLICITY OF RELATIONSHIPS

- Product
  - price
  - name
  - category
  - makes
    - Company
      - name
      - stockprice
      - employs
        - Person
          - name
          - address
          - ssn
      - buys
**NOTATION DIFFERENCE**

- We use:

- The cow book uses (page 33):

You should use the notation in the slides!
How do we model a purchase relation between buyers, products and stores?

We can still model this as a mathematical set (ternary relation)
ARROWS IN MULTI-WAY RELATIONSHIPS

What does the arrow mean here?

A given person can purchase a given product from at most one store!
ARROWS IN MULTI-WAY RELATIONSHIPS

What about here?

A given person can **purchase** a given **product** from at most one **store** AND a given **store** sells to a given **person** at most one **product**
How can we say that a given person buys from at most one store?

Not possible, we can only approximate!
What if we need an entity set twice in a relationship?

- Product
- Purchase
- Store
- Person (salesperson)
- Person (buyer)
Roles in Relationships

- Label the edges to indicate the roles
- Collapse the two entity sets into one

Diagram:

- Product
- purchase
- Store
- Person
- salesperson
- buyer
Why do we need the arrows?
**Relationships: Recap**

- Modeled as a *mathematical set*
- **Binary** and *multi-way* relationships
- Converting a multi-way one into many binary ones
- **Constraints** on the degree of the relationship
  - many-one, one-one, many-many
  - limitations of arrows
- **Attributes** of relationships
  - not necessary, but useful
ADDITIONAL FEATURES
**SUBCLASSES**

subclass = specialized case

= fewer entities

= more properties

• **Example:** Products
  – Software products
  – Educational products
Attributes are inherited by the subclasses!
**CONSTRAINTS**

**constraint** = an assertion about the database that must be true at all times

- part of the database schema
- central in database design

When creating the ER diagram, you need to find as many constraints as possible!
**Types of Constraints**

- **keys**: SSN *uniquely* identifies a person
- **single-value**: a person can have *only one* father
- **referential integrity**: if you work for a company, it *must exist* in the database
- **domain**: age is between 0 and 150
- **other**: e.g. at most 80 students enroll in a class
Why do we need constraints?

• Give more semantics to the data
  – help us better understand it
• Prevent wrong data entry
• Allow us to refer to entities (e.g. using keys)
• Enable efficient storage and data lookup
Key Constraints
**Key Constraints**

- *Every entity set must have a key*
- A key can consist of more than one attribute
- There can be more than one key for an entity set
  - one key will be designated as primary key
- No formal way to specify multiple keys in an ER diagram
SINGLE-VALUE CONSTRAINTS

An entity may have at most one value for a given attribute or relationship

- an attribute of an entity set has a single value
- a many-one relation implies a single value constraint
A relationship has one value and the value must exist
Entity sets are **weak** when their key attributes come from other classes to which they are related. Entities of an entity set need “help” to identify them uniquely!
DESIGN PRINCIPLES
1. Be Faithful to the App!

What is wrong here?

- Product
  - purchase
  - Person

- Country
  - president
  - Person
2. AVOID REDUNDANCY!

What is wrong here?

Product -> purchase -> Store

- personName
- personAddress
- date
2. **Avoid Redundancy!**

- Redundancy occurs when we say the same thing in two different ways
- Redundancy wastes space and encourages inconsistency
  - The two instances of the same fact may become inconsistent if we change one and forget to change the other
2. **Avoid Redundancy!**

What is wrong here?

![UML Diagram]

- Beer
- manfBy
- Manufacturer
  - address
  - name
  - manufacturer
What is wrong here?
4. Attributes Over Entities

An entity set should satisfy at least one of the following conditions

- it is more than the name of something; it has at least one non-key attribute
- it is the “many” in a many-one or many-many relationship

What is wrong here?
5. Don’t Overuse Weak Entity Sets

• Beginning database designers often doubt that anything could be a key by itself
  – They make all entity sets weak, supported by all other entity sets to which they are linked

• In reality, we create unique IDs for entity sets
  – Examples: SSN, ISBN, ...
ER Model: Recap

Key concepts:
• entity, attribute, entity set
• relation: binary, multi-way
• relationship roles, attributes on relationships
• subclasses (ISA)
• weak entity sets
• constraints
  – many-one, one-one, many-many
  – keys, single-valued, referential integrity
• design principles