Database Applications (15-415)

The Entity Relationship Model
Lecture 2, January 15, 2014

Mohammad Hammoud
Today...

- **Last Session:**
  - Course overview and a brief introduction on databases and database systems

- **Today’s Session:**
  - Main steps involved in designing databases
  - Constructs of the entity relationship (ER) model
  - Integrity constrains that can be expressed in the ER model
  - Conceptual design choices

- **Announcements:**
  - The first Problem Solving Assignment (PS1) is now posted
    - Deadline is Jan 23, 2014 by midnight
  - Tomorrow is the first recitation
    - A case study on the ER model will be solved
Outline

- Database Design
- ER Model: Constructs and Constraints
- ER Model: Conceptual Design Choices
Database Design

- Requirements Analysis
  - Users needs

- Conceptual Design
  - A high-level description of the data (e.g., using the ER model)

- Logical Design
  - The conversion of an ER design into a relational database schema

- Schema Refinement
  - Normalization (i.e., restructuring tables to ensure some desirable properties)

- Physical Design
  - Building indexes and clustering some tables

- Security Design
  - Access controls
Outline

Database Design

ER Model: Constructs and Constraints

ER Model: Conceptual Design Choices
Entities and Entity Sets

- **Entity:**
  - A real-world object distinguishable from other objects in an enterprise (e.g., University, Students and Faculty)
  - Described using a set of attributes

- **Entity set:**
  - A collection of similar entities (e.g., *all* employees)
  - All entities in an entity set have the same set of attributes (until we consider ISA hierarchies, anyway!)
  - Each entity set has a *key*
  - Each attribute has a *domain*
“ssn” is the primary key, hence, underlined.
Relationship and Relationship Sets

- **Relationship:**
  - Association among two or more entities (e.g., Mohammad is teaching 15-415)
  - Described using a set of attributes

- **Relationship set:**
  - Collection of similar relationships
  - Same entity set could participate in different relationship sets, or in different “roles” in the same set
More Tools and ER Diagrams

Relationships (‘rel. sets’) and mapping constraints

A Binary Relationship

A Self-Relationship
Suppose that departments have offices at different locations and we want to record the locations at which each employee works.

Consequently, we must record an association between an employee, a department and a location.

This is referred to as a “Ternary Relationship” (vs. Self & Binary Relationships).
Key Constraints

- Consider the “Employees” and “Departments” entity sets with a “Manages” relationship set
  - An employee can work in many departments
  - A department can have many employees
  - This restriction is an example of a “key constraint”

Key constraints are denoted by thin arrows.
Entities can be related to one another as “one-to-one”, “one-to-many” and “many-to-many”.

This is said to be the cardinality of a given entity in relation to another.
Cardinalities: Examples

- **COUNTRY** has **1 CAPITAL**
- **PERSON** owns **N CAR**
- **STUDENT** takes **M SECTION**
Cardinalities: Examples

**Book’s notation:**

- **COUNTRY** has **CAPITAL**
- **PERSON** owns **CAR**
- **STUDENT** takes **SECTION**
Cardinalities: Examples

Book’s notation vs 1 to N notation
A Working Example

- **Requirements:** Students, taking courses, offered by instructors; a course may have multiple sections; one instructor per section

- **How to start?**
  - Nouns -> entity sets
  - Verbs -> relationship sets
Primary key = unique identifier
But: sections of a course (with different instructors)?
But: s-id is not unique... (see later)
Q: how to record that students take courses?
Participation Constraints

- Consider again the “Employees” and “Departments” entity sets as well as the “Manages” relationship set
  - Should every department have a manager?
  - If so, this is a participation constraint
  - Such a constraint entails that every Departments entity must appear in an instance of the Manages relationship
  - The participation of Departments in Manages is said to be total (vs. partial)

Total participation is denoted by a **thick line**

Total participation + key constraint are denoted by a **thick arrow**
Total vs. Partial Participations

Total, Total

- **COUNTRY** has **CAPITAL**

??

- **PERSON** owns **CAR**

??

- **STUDENT** takes **SECTION**
Total vs. Partial Participation

Total, Total

COUNTRY has CAPITAL

Partial, Total

PERSON owns CAR

??

STUDENT takes SECTION
Total vs. Partial Participation

Total, Total
- COUNTRY has CAPITAL

Partial, Total
- PERSON owns CAR

Partial, Total
- STUDENT takes SECTION
Weak Entities

- A **weak entity** can be identified uniquely only by considering the primary key of another (owner) entity
  - Owner entity set and weak entity set must participate in a one-to-many relationship set (one owner, many weak entities)
  - Weak entity set must have total participation in this identifying relationship set

- The set of attributes of a weak entity set that uniquely identify a weak entity for a given owner entity is called **partial key**
Weak Entities: An Example

- “Dependents” has no unique key of its own
  - “Dependents” is a weak entity with partial key “pname”
  - “Policy” is an identifying relationship set
  - “pname” + “ssn” are the primary key of “Dependents”

Partial keys are underlined using broken lines

Weak entities and identifying relationships are drawn using thick lines
ISA (‘is a’) Hierarchies

- Entities in an entity set can sometimes be classified into subclasses (this is “kind of similar” to OOP languages)

- If we declare B ISA A, every B entity is also considered to be an A entity

“Employees” is specialized into subclasses

“Hourly_Emps” and “Contract_Emps” are generalized by “Employees”
Overlap and Covering Constraints

- **Overlap constraints**
  - Can an entity belong to both ‘B’ and ‘C’?

- **Covering constraints**
  - Can an ‘A’ entity belong to neither ‘B’ nor ‘C’?
Overlap Constraints: Examples

- **Overlap constraints**
  - Can John be in Hourly_Emps and Contract_Emps? Intuitively, *no*

  - Can John be in Contract_Emps and in Senior_Emps? Intuitively, *yes* ➞ “Contract_Emps OVERLAPS Senior_Emps”
Covering Constraints: Examples

- Covering constraints
  - Does every one in Employees belong to a one of its subclasses? Intuitively, *no*
  
  - Does every Motor_Vehicles entity have to be either a Motorboats entity or a Cars entity? Intuitively, *yes* ➞ “Motorboats AND Cars COVER Motor_Vehicles”
More Details on ISA Hierarchies

- Attributes are *inherited* (i.e., if B ISA A, the attributes defined for a B entity are the attributes for A plus B)

- We can have *many* levels of an ISA hierarchy

- Reasons for using ISA:
  - To add descriptive attributes specific to a subclass
  - To identify entities that participate in a relationship
Aggregation

- Aggregation allows indicating that a relationship set (identified through a *dashed box*) participates in another relationship set.

```
Employees

Monitors

Projects

Sponsors

Departments

name

ssid

lot

started_on

since

did

dname

budget

pid

pbudget

until


```
Outline

Database Design

ER Model: Constructs and Constraints

ER Model: Conceptual Design Choices
Conceptual Design Choices

- Should a concept be modeled as an *entity* or an *attribute*?
- Should a concept be modeled as an *entity* or a *relationship*?
- How should we identify relationships?
  - *Binary* or *ternary*?
  - *Ternary* or *aggregation*?
- Constraints in the ER Model:
  - A lot of data semantics can (and should) be captured
  - But some constraints cannot be captured in ER diagrams
Entity vs. Attribute

- Should *address* be an attribute of Employees or an entity (connected to Employees by a relationship)?

- This depends upon the use we want to make of address information, and the semantics of the data
  - If we have several addresses per an employee, *address* must be an entity (since attributes cannot be set-valued)
  - If the structure (city, street, etc.) is important (e.g., we want to retrieve employees in a given city), *address* must be modeled as an entity
Entity vs. Attribute (Cont’d)

- Consider the following ER diagram:

- **A problem**: Works_In4 does not allow an employee to work in a department for two or more periods

- **Solution**: introduce “Duration” as a new entity set
Entity vs. Relationship

- Consider the following ER diagram whereby a manager gets a separate discretionary budget for each department.

- What if a manager gets a discretionary budget that covers all managed departments?
  - Redundant data
  - Misleading

This fixes the problem!
Binary vs. Ternary Relationships

If each policy is owned by just 1 employee:
Binary vs. Ternary Relationships

If each policy is owned by just 1 employee:

---

Key constraint on Policies would mean policy can only cover 1 dependent!
If each policy is owned by just 1 employee:

Better design!

What are the additional constraints?
Binary vs. Ternary Relationships

- But sometimes ternary relationships cannot be replaced by a set of binary relationships
Binary vs. Ternary Relationships

- But sometimes ternary relationships cannot be replaced by a set of binary relationships.

Diagram:

- Contract
  - qty
  - Suppliers
  - Parts

VS.

- Suppliers
  - can-supply
    - Parts
  - deals-with
    - Departments
  - Departments
Binary vs. Ternary Relationships

- But sometimes ternary relationships cannot be replaced by a set of binary relationships
But sometimes ternary relationships cannot be replaced by a set of binary relationships.
Binary vs. Ternary Relationships

- But sometimes ternary relationships cannot be replaced by a set of binary relationships

- $S$ “can-supply” $P$, $D$ “needs” $P$, and $D$ “deals-with” $S$ does not imply that $D$ has agreed to buy $P$ from $S$

- How do we record $qty$?

why is it bad?
Ternary vs. Aggregation Relationships

- When to use aggregation?
  - If we want to attach a relationship to a relationship

- What if we do not want to record the *until* attribute of Monitors relationship?
We might reasonably use a ternary relationship instead of an aggregation

What if each sponsorship (of a project by a department) is to be monitored by at most one employee?
Summary

- **Conceptual design follows requirements analysis**
  - Yields a high-level description of data to be stored

- The ER model is popular for conceptual design
  - Its constructs are expressive, close to the way people think about their applications

- The basic constructs of the ER model are:
  - *Entities, relationships, and attributes* (of entities and relationships)
Summary

- Some additional constructs of the ER model are:
  - Weak entities, ISA hierarchies, and aggregation

- Several kinds of integrity constraints can be expressed in the ER model
  - Key constraints, participation constraints, and overlap/covering constraints for ISA hierarchies

- Note: there are many variations on the ER model
Summary

- ER design is *subjective*
  - There are often many ways to model a given scenario!

- Analyzing alternatives can be tricky, especially for a large enterprise

- Common choices include:
  - Entity vs. attribute
  - Entity vs. relationship
  - Binary or *n-ary* relationship (e.g., ternary)
  - Whether or not to use ISA hierarchies
  - Whether or not to use aggregation
Next Class

The relational Model