CS 2441 Database Systems

Entity-Relationship (ER) Model

Part 1: Schedule

- Start with Data Models
  - Relational Model with a little ER model intro
- Formal query languages- Relational algebra
- SQL
- Database schema design: how to design a “good” schema, how to measure “good”?
  - Normal Forms (3NF, BCNF)
- Demonstrate concepts learnt on Commercial DBMS – Oracle, MySQL
How Does One Build a Database?

- Requirements Analysis: what data, apps, critical operations
  - Get from “client”
  - Typically expressed in some natural language
  - May require going back to the client for resolving questions

Building a Database and Application

1. Start with a conceptual model
   - “On paper” using certain techniques
     - E-R Model
     - Ignore low-level details – focus on logical representation
     - “step-wise refinement” of design with client input

2. Design & implement schema
   - Design and codify (in SQL) the relations/tables
   - Refine the schema – *normalization*
   - Do physical layout – indexes, etc.

3. Import the data

4. Write applications using DBMS and other tools
   - Many of the hard problems are taken care of by other people
     (DBMS, API writers, library authors, web server, etc.)
   - DBMS takes care of Query Optimization, Efficiency, etc.
**Conceptual Model: Why use a graphical language?**

- Convey database design and properties in simple but precise manner
  - Interpreted by any type of user
    - Does not need to know anything about CS
  - Capture the business rules of the application

- Picture is worth a thousand words

---

**An Example: “mini” banner**

- Database containing information about
  - Students
  - Faculty
  - Courses
- Students take courses
- Faculty teach courses
- How to ‘define’ student/faculty/course?
Entity Relationship Model

- Based on collection of real world objects or concept called *entities*; ex: employee, student
  - *attribute* represents properties of entity; s.s.num
- *relationship* represents interaction between entities
- overall logical structure represented by ER diagram representing entity sets, relationships, attributes

ER Model Basics

- **Conceptual design:**
  - What are the *entities* and *relationships* in the enterprise?
  - What information about these entities and relationships should we store in the database?
  - What are the *integrity constraints* or *business rules* that hold?
- Can map an ER diagram into a relational schema.
ER Model Definitions

- **Entity**: Real-world object distinguishable from other objects.
  - An entity is described (in DB) 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.

- Representation/Syntax:
  - Entity set represented by rectangle
  - Attribute represented by Oval
    - Key attribute underlined

ER Model Basics (Contd.)

- **Relationship**: Association among two or more entities. E.g., Dan takes Database Course; Attishoo works in Pharmacy department.
  - Relationship can also have attributes (that appear only for this relationship set)

- Representation/Syntax: a Diamond symbol
  - Attributes represented by Oval (same as before)

- **Relationship Set**: Collection of similar relationships.
  - An n-ary relationship set R relates n entity sets E1 ... En; each relationship in R involves entities e1 ∈ E1, ..., en ∈ En
    - Same entity set could participate in different relationship sets, or in different “roles” in same set.
Conceptual Design Process

- What are the entities being represented? **STUDENTS**
- What are the relationships? **Takes**
- What info (attributes) do we store about each? **name**, **exp-grade**
- What keys & integrity constraints do we have? **sid**

Student Entity

- **sid**: Student
- **name**: Student
- **exp-grade**: Takes

Diagram:
- Node: **Student**
- Node: **sid**
- Node: **name**
- Node: **exp-grade**
- Link: **Takes**
Connectivity in the E-R Diagram?

- Attributes can only be connected to entities or relationships
- Entities can only be connected via relationships
- As for the edges, let’s consider kinds of relationships and integrity constraints…

Entity-Relationship Diagram for the Example

*Underlined attributes are keys*

- **STUDENTS**: sid, name
- **COURSES**: serno, subj, cid
- **PROFESSORS**: fid, name
- **Teaches**: semester
- **Takes**: exp-grade

(entity set relationship set)

attributes (recall these have domains)
**Example: A Company Database**

- COMPANY database keeps track of Employees and Departments
  - Employees identified by SSN, Name, Location
  - Department specified by Department ID (did), Name, Budget
- Each department has a unique manager
  - Database must keep track of starting date
- Each employee works in a department
  - Database must keep track of starting date

**Constraints – Key and Participation**

- Capture properties of the relationship and entities
- Every entity set has a key attribute
  - No two elements can have the same value on this attribute
    - Example: Student ID
- Does every element in the entity set appear/participate in the relationship?
  - Must every student take a course?
- Define constraints based on properties of the mapping/relation between entity sets
Properties of relations

- Binary relationships can be classified as one-to-one, many-to-one, one-to-many, many-to-many
- What is the type of mapping/relation

![Diagram of mapping types]

Example: the Teaches relationship

- Want to model the info that each course is taught by one faculty.
  - Type of mapping ???
  - 1-to-1
    - Note: This is a Mapping and not a function!
- Every course must have an instructor
  - Each element in the Course entity set must participate/appear in the Teaches relationship
- A faculty may teach zero or more courses
Takes Relationship

- Student can be enrolled in many courses and each Course can have many students
  - Type of mapping:
    - Many to Many
- Want to model the condition that every student must take at least one course
  - Each student must appear in Takes relationship
- How many courses can a student take?
- How many students must be enrolled in a course?

Mapping Cardinality, Participation Constraints, Structural constraints

- Type of mapping (cardinality)
  - 1-1, 1-many, many-many, many-1
  - Provides some information on relationship sets
- Participation constraints
  - Total vs Partial
    - Total: Every student sid must appear in Takes relationship
    - Partial: All faculty need not appear in Teaches relationship
- Structural constraints:
  - Minimum and maximum times they can appear in relationship
  - Syntax ??
**Roles: Labeled Edges**

Sometimes a relationship connects the same entity, and the entity has more than one role:

![Diagram of Labeled Edges]

This often indicates the need for recursive queries.

---

**Roles vs. Separate Entities**

![Diagram of Roles vs. Separate Entities]

What is the difference between these two representations?
Weak Entity Sets

- 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.
  - If Student is deleted, then we MUST delete the Parent

ISA Relationships: Subclasses (Structurally)

- Inheritance states that one entity is a “special kind” of another entity: “subclass” should be member of “base class”
**Relationships: Binary or n-ary**

- Binary: Relationship between two entity sets
- N-ary: Relationship between any N entity sets
  - Not all n-ary can be converted to a set of binary relationships

**Conceptual Design Using the ER Model**

- **Design choices:**
  - Should a concept be modeled as an entity or an attribute?
  - Should a concept be modeled as an entity or a relationship?
  - Identifying relationships: constraints, type, participation

- **Constraints in the ER Model:**
  - A lot of data semantics can (and should) be captured.
  - But some constraints cannot be captured in ER diagrams.
Summary of Conceptual Design

- Conceptual design follows requirements analysis,
  - Yields a high-level description of data to be stored
  - Visual language – the diagram is the syntax!

- ER model popular for conceptual design
  - Constructs are expressive, close to the way people think about their applications.
  - There are additional constructs in a “real” ER model based tools.

- Can automate mapping of ER model to relational tables!