Comp 115: Databases

The Entity-Relationship Model

Instructor: Manos Athanassoulis

http://www.cs.tufts.edu/comp/115/
The Entity-Relationship Model

Basic ER modeling concepts
Readings: Chapters 2.1-2.3

Constraints

Complex relationships

Conceptual Design
Databases Model the Real World

“Data Model” allows us to translate real world things into structures that a computer can store.

Many models: Relational, ER, O-O, Network, Hierarchical, etc.

Relational

Rows & Columns

Keys & Foreign Keys to link Relations

<table>
<thead>
<tr>
<th>sid</th>
<th>cid</th>
<th>grade</th>
</tr>
</thead>
<tbody>
<tr>
<td>53666</td>
<td>Carnatic101</td>
<td>5</td>
</tr>
<tr>
<td>53666</td>
<td>Reggae203</td>
<td>5.5</td>
</tr>
<tr>
<td>53650</td>
<td>Topology112</td>
<td>6</td>
</tr>
<tr>
<td>53666</td>
<td>History105</td>
<td>5</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>sid</th>
<th>name</th>
<th>login</th>
<th>age</th>
<th>gpa</th>
</tr>
</thead>
<tbody>
<tr>
<td>53666</td>
<td>Jones</td>
<td>jones@cs</td>
<td>18</td>
<td>5.4</td>
</tr>
<tr>
<td>53688</td>
<td>Smith</td>
<td>smith@eecs</td>
<td>18</td>
<td>4.2</td>
</tr>
<tr>
<td>53650</td>
<td>Smith</td>
<td>smith@math</td>
<td>19</td>
<td>4.8</td>
</tr>
</tbody>
</table>
Database Design

Requirements Analysis
  user needs; what must database do?

Conceptual Design
  high level description (often done w/ ER model)

Logical Design
  translate ER into DBMS data model

Schema Refinement
  consistency, normalization

Physical Design
  indexes, disk layout

Security Design
  who accesses what
Database Design

Requirements Analysis
user needs; what must database do?

Conceptual Design
high level description (often done w/ ER model)

Logical Design
translate ER into DBMS data model

Schema Refinement
consistency, normalization

Physical Design
indexes, disk layout

Security Design
who accesses what
Conceptual Design

*entities* and *relationships*

what should we store for each?

what are the *integrity constraints* that hold?

a database “schema” in the ER Model can be represented pictorially (*ER diagrams*)

ER diagrams are mapped to relational schemata
**Entity**: real-world object, described (in DB) using a set of **attributes**

**Entity Set**: a collection of similar entities (all employees)

- entities in an entity set have the **same attributes**
- each entity set has a **key**
- each attribute has a **domain**
Relationship: association among two or more entities: “Fred works in Pharmacy department”
relationships can have their own attributes

Relationship Set: collection of (similar) relationships
entity set can participate in different relationship sets

or

in different “roles” in the same set
The Entity-Relationship Model

Basic ER modeling concepts

Constraints
Readings: Chapters 2.4-2.4.3, 2.5.3

Complex relationships

Conceptual Design
Key Constraints

An employee can work in many departments; a department can have many employees.

In contrast, each department has at most one manager, according to the key constraint on Manages.
Participation Constraints

does every employee work in a department?

If so, this is a **participation constraint**: the participation of Departments in Manages is said to be **total (vs. partial)**

Basically means “at least one”

Means: “exactly one”
A **weak entity** can be identified uniquely by the primary key of another (*owner*) entity (+ some of its attributes)

- 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

Weak entities have only a “partial key” (dashed underline)
More Elaborate (and Realistic) Example

Should we add any additional constraints?

every employee a policy?

every policy at least one dependent?
Ternary Relationships

in general, n-ary relationships
S “can-supply” P, D “needs” P, and D “deals-with” S. Does it imply that D has agreed to buy P from S? If so, how do we record qty?
Now you try

University database schema

**Entities:** Courses, Students, Teachers

Each course has id, title, time, room #

Make up suitable attributes for students, teachers

Each course has **exactly one** teacher

Students have a grade for each course
Now ... keep track of multiple semesters!

- each course for at most once each semester
- each course offering has exactly one teacher
- track student transcripts across entire enrollment period
- track history of courses taught by each teacher
The Entity-Relationship Model

Basic ER modeling concepts

Constraints

Complex relationships
Readings: Chapters 2.4.4-2.4.5

Conceptual Design
ISA (‘is a’) Hierarchies

as in C++, or other PLs, attributes are inherited

if we declare A ISA B, every A entity is also considered to be a B entity.
ISA (‘is a’) Hierarchies

Overlap constraints: Can Joe be an Hourly_Emps as well as a Contract_Emps entity? *(Allowed/Disallowed)*

Covering constraints: Does every Employees entity also have to be an Hourly_Emps or a Contract_Emps entity? *(Yes/No)*

Reasons for using ISA:
- to add descriptive attributes specific to a subclass
  - we do not keep “hours worked” for everybody
- to identify entities that participate in a particular relationship
  - manager can be only a “contract employee”
Aggregation

used for a relationship involving another relationship set

[for purposes of participation in (other) relationships]

Monitors is a distinct relationship, with a descriptive attribute

Also, can say that each sponsorship is monitored by at most one employee
The Entity-Relationship Model

Basic ER modeling concepts

Constraints

Complex relationships

Conceptual Design

Readings: Chapter 2.5
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: binary or ternary? 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 (related to Employees)?

**Depends** upon how we want to use address information, and the semantics of the data:

If we have **several addresses per employee**, *address* must be an entity (since attributes cannot be set-valued)

If the **structure** (city, street, etc.) is **important**, *address* must be modeled as an entity (since attribute values are atomic)
Entity vs. Attribute (Cont.)

Works_In2 does not allow an employee to work in a department for two or more periods.

why?

Approach: Similar to the problem of wanting to record several addresses for an employee: we want to record several values of the descriptive attributes for each instance of this relationship.
OK as long as a manager gets a separate discretionary budget \((dbudget)\) for each department

What if manage’s \(dbudget\) covers all managed departments? (can repeat value, but such redundancy is problematic)
Summary of Conceptual Design

*Conceptual design* follows *requirements analysis*

Yields a high-level description of data to be stored

ER model popular for conceptual design

Constructs are expressive, close to the way people think about their applications

Originally proposed by Peter Chen, 1976

Note: there are many variations on ER model

Basic constructs: *entities, relationships,* and *attributes* (of entities and relationships)

Some additional constructs: *weak entities, ISA hierarchies,* and *aggregation*
Notes on the ER design

ER design is *subjective*

many “correct” ways to model a given scenario!

analyzing alternatives can be tricky

*common dilemmas*: entity vs. attribute, entity vs. relationship, binary or n-ary relationship, whether or not to use ISA hierarchies, aggregation

many types of *constraints cannot be expressed* (notably, *functional dependencies*)

[although constraints play an important role in determining the best database design for an enterprise]
Context: Overall Database Design Process

Today

Requirements Analysis
user needs; what must database do?

Conceptual Design
high level description (often done w/ER model)

Next time: Logical Design
translate ER into DBMS data model

Later: Schema Refinement
consistency, normalization

Physical Design
indexes, disk layout

Security Design
who accesses what