CS 4604: Introduction to Database Management Systems

B. Aditya Prakash

Lecture #6: Entity/Relational Models---Part 1
NOT IN BOOK!

- IMPORTANT:
  - Follow only lecture slides for this topic!
  - Differences from the book:
    - More details
    - Slightly different notation
Database Design

- Requirements Analysis
  - user’s needs
- Conceptual Design
  - high level (E/R)
- Logical Design
  - tables (schema)
- Schema Refinement
  - normalization
- Physical Design
  - indices etc.
- Security Design
  - access controls
Database Design

- Requirements Analysis
  - user’s needs
- Conceptual Design
  - high level (E/R)
- Logical Design
  - tables (schema)
- Schema Refinement
  - normalization
- Physical Design
  - indices etc.
- Security Design
  - access controls
Basic Database Terminology

- **Data model**: describes high-level conceptual structuring of data
  - Example: Data is set of student records, each with ID, name, address, and courses
  - Example: Data is a graph where nodes represent people and edges represent friendship relations

- **Schema** describes how data is to be structured and stored in a database
  - Defined during creation of the database
  - Schemas rarely change

- **Data** is actual “instance” of database
  - Updated continuously
  - Changes rapidly
Why Learn About Database Modeling?

- The way in which data is stored is very important for subsequent access and manipulation by SQL.

- Properties of a good data model:
  - It is easy to write correct and easy to understand queries.
  - Minor changes in the problem domain do not change the schema.
  - Major changes in the problem domain can be handled without too much difficulty.
  - Can support efficient database access.
Purpose of E/R Model

- The E/R model allows us to sketch the design of a database informally.
  - Represent different types of data and how they relate to each other
- Designs are drawings called *entity-relationship diagrams*.
- Fairly mechanical ways to convert E/R diagrams to real implementations like relational databases exist.
When designing E/R diagrams,
– forget about relations/tables!
– only consider how to model the information you need to represent in your database.
Example

- Professors advising students, Students taking courses, Students taught by professors
Tools

- Entities (‘entity sets’)
- Relationships (‘rel. sets’) and mapping constraints
- Attributes
Example

- Professors advising students, Students taking courses, Students taught by professors

Nouns $\rightarrow$ entity sets
Verbs $\rightarrow$ relationship sets
Entity Sets

- **Entity** = “thing” or objects
- **Entity set** = collection of similar entities.
  - Similar to a class in object-oriented languages.
- **Attribute** = property of an entity set.
  - Generally, all entities in a set have the same properties.
  - Our convention is to use ‘atomic attributes’ e.g. integers, character strings etc.
  - FYI: there exist
    - **multivalued** or set-valued attributes (eg., ‘dependents’ for EMPLOYEE)
    - **derived** attributes (eg., 15% tip)
E/R Diagrams

- In an entity-relationship diagram, each entity set is represented by a rectangle.
- Each attribute of an entity set is represented by an oval, with a line to the rectangle representing its entity set.
Example: Entity Sets

- **Students**
  - PID
  - Name
  - Address

- **Courses**
  - Name
  - DeptName
  - Classroom

**Prakash 2014**
A relationship connects two or more entity sets.

It is represented by a diamond, with lines to each of the entity sets involved.

Don’t confuse ‘Relationships’ with ‘Relations’!
Example: Relationships

Students
- PID
- Name
- Address
- Take
  - Courses
    - Number
    - Name
    - DeptName
    - Classroom
- Advisor
- Professors
  - Name
  - Age
- Teach
  - Professors
    - Advise
    - Students

Students Take Courses
Professors Teach Courses
Professors Advise Students

Prakash 2014
Instance of an E/R Diagram

- An E/R is NOT an implementation of the DB
  – Just a notation for specifying structure

- Still useful to think of instance of an E/R Diagram === the particular data stored in a database
Instance of an Entity Set

- For each entity set, the instance stores a specific set of entities
- Each entity is a tuple containing specific values for each attribute
- Example: Instance of Entity set Students

<table>
<thead>
<tr>
<th>Name</th>
<th>PID</th>
<th>Address</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hermione Grainger</td>
<td>HG</td>
<td>Gryffindor Tower</td>
</tr>
<tr>
<td>Draco Malfoy</td>
<td>DM</td>
<td>Slytherin Tower</td>
</tr>
<tr>
<td>Harry Potter</td>
<td>HP</td>
<td>Gryffindor Tower</td>
</tr>
<tr>
<td>Ron Weasley</td>
<td>RW</td>
<td>Gryffindor Tower</td>
</tr>
</tbody>
</table>
Instance of a Relationship

- Example: Instance of relationship Takes (no DeptName)

<table>
<thead>
<tr>
<th>Student</th>
<th>PID</th>
<th>Address</th>
<th>CourseName</th>
<th>Enrollment</th>
<th>Grade</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hermione Grainger</td>
<td>HG</td>
<td>Gryffindor</td>
<td>Potions</td>
<td>∞</td>
<td>A-</td>
</tr>
<tr>
<td>Draco Malfoy</td>
<td>DM</td>
<td>Slytherin</td>
<td>Potions</td>
<td>∞</td>
<td>B</td>
</tr>
<tr>
<td>Harry Potter</td>
<td>HP</td>
<td>Gryffindor</td>
<td>Potions</td>
<td>∞</td>
<td>A</td>
</tr>
<tr>
<td>Ron Weasley</td>
<td>RW</td>
<td>Gryffindor</td>
<td>Potions</td>
<td>∞</td>
<td>C</td>
</tr>
</tbody>
</table>

- Relationship R between (entity sets) E and F
  - Relates some *entities* in E to some *entities* in F
Instance of a Relationship

- Instance is a set of pairs of tuples (e; f ) where e is in E and f is in F
  - Instance need not relate every tuple in E with every tuple in F
  - Relationship set for R: the pairs of tuples (e; f ) related by R
- (Conceptually) An instance of R is simply the ‘concatentation’ of the attribute lists for all pairs of tuples (e; f ) in the relationship set for R
- ‘Tuples’ in R have two components, one from E and one from F
Attributes for a Relationship

- Question: What is Grade an attribute of?
- Such an attribute is a property of the entity-pairs in the relationship
Many-Many Relationships

- In a *many-many* relationship, an entity of either set can be connected to many entities of the other set.
Many-One Relationships

- Some binary relationships are *many-one* from one entity set to another.
- Each entity of the first set is connected to at most one entity of the second set.
- But an entity of the second set can be connected to zero, one, or many entities of the first set.
One-One Relationships

- In a one-one relationship, each entity of either entity set is related to at most one entity of the other set.

- The schema defines the multiplicity of relationships. Don't use the instances of the schema to determine multiplicity.
Representing “Multiplicy”

- Show a many-one relationship by an arrow entering the “one” side.
- Show a one-one relationship by arrows entering both entity sets.
Different kinds of relationships

- many-many
- many-one
- one-one

Profs. Advise Students

Students Take Courses

Profs. Office Rooms

Prakash 2014

VT CS 4604
Exactly one

- In some situations, we can also assert “exactly one,” i.e., each entity of one set must be related to exactly one entity of the other set. To do so, we use a rounded arrow.
Example: Exactly One

- Consider *Best-course* between *Profs* and *Courses*.
- Some courses are not the best-course of any professor, so a rounded arrow to *Profs* would be inappropriate.
- But a professor has to have a best-course
Roles in Relationships

- Can the same entity set appear more than once in the same relationship?
- Prerequisite relationship between two Courses

- But which course is the pre-req?
Roles in Relationships

- Label the connecting lines with the *role* of the entity
Parallel Relationships

- Can there be more than one relationship between the same pair of entities?
- TA and Take relationship between Students and Classes
Are Attributes on Relationships Needed

- Attribute on relationship → Attribute to an entity and make relationship multi-way
Entity vs. attribute

- Entity EMPLOYEE (w/ emp#, name, job_code, ...)
- Q: How about ‘spouse’ - entity or attribute?
- Q: How about ‘dependents’?
Entity vs. attribute

- Entity EMPLOYEE (w/ emp#, name, job_code, ...)
- Q: How about ‘spouse’ - entity or attribute?
  - A: probably, ‘attribute’ is enough
- Q: How about ‘dependents’?
  - A: Entity - we may have many dependents
Multi-way Relationships

- Relationships may connect more than 2 entity sets
- \( \geq 1 \) professor can teach a course but each student evaluates each professor separately
- Three-way Evaluation relationship between Students, Professors, and Classes
Multi-way Relationships

- \( \geq 1 \) professor can teach a course but each student taught by at most one professor, and each student only evaluates that professor
- Add arrow directed towards Professors
### Multiplicity in Multiway Relationships

- An arrow pointing to an entity set $E \Rightarrow$ if we select an entity from each of the other entity sets, the selected entities are related to at most one entity in $E$.

- E/R diagram forbids connections between “Hermione Grainger”, “Potions” and two different professors.

<table>
<thead>
<tr>
<th>Student</th>
<th>Course</th>
<th>Professor</th>
<th>Grade</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hermione Grainger</td>
<td>Potions</td>
<td>Snape</td>
<td>F-</td>
</tr>
<tr>
<td>Draco Malfoy</td>
<td>Potions</td>
<td>Snape</td>
<td>A*</td>
</tr>
<tr>
<td>Harry Potter</td>
<td>Potions</td>
<td>Lupin</td>
<td>A+</td>
</tr>
<tr>
<td>Ron Weasley</td>
<td>Potions</td>
<td>Lupin</td>
<td>B+</td>
</tr>
</tbody>
</table>
Binary vs Ternary Rel.

- Can a ternary rel. be replaced by binary rels?
Attempt 1

Is this OK?
– 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?
Attempt 2
Attempt 2: contd
Converting Multiway to Binary

- It is easy to convert a multiway relationship to multiple binary relationships
  - Create a new connecting entity set. Think of its entities as the tuples in the relationship set for the multiway relationship
  - Introduce relationships from the connecting entity set to each of the entities in the original relationship
  - If an entity set plays > 1 role, create a relationship for each role
Converting Multiway to Binary

Equivalent:
Not exactly equivalent, but can be made so by additional FDs.
Example of the Conversion

- Instance of Evaluation (ternary) relationship before conversion:

<table>
<thead>
<tr>
<th>Student</th>
<th>Course</th>
<th>Professor</th>
<th>Grade</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hermione Grainger</td>
<td>Potions</td>
<td>Snape</td>
<td>F-</td>
</tr>
<tr>
<td>Draco Malfoy</td>
<td>Potions</td>
<td>Snape</td>
<td>A*</td>
</tr>
<tr>
<td>Harry Potter</td>
<td>Potions</td>
<td>Lupin</td>
<td>A+</td>
</tr>
<tr>
<td>Ron Weasley</td>
<td>Potions</td>
<td>Lupin</td>
<td>B+</td>
</tr>
</tbody>
</table>
Example of the Conversion

- Instance of Evaluation (ternary) relationship before conversion:

- After Evaluation entity set

<table>
<thead>
<tr>
<th>Eval_Id</th>
<th>Grade</th>
</tr>
</thead>
<tbody>
<tr>
<td>e1</td>
<td>F-</td>
</tr>
<tr>
<td>e2</td>
<td>A*</td>
</tr>
<tr>
<td>e3</td>
<td>A+</td>
</tr>
<tr>
<td>e4</td>
<td>B+</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Student</th>
<th>Course</th>
<th>Professor</th>
<th>Grade</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hermione Grainger</td>
<td>Potions</td>
<td>Snape</td>
<td>F-</td>
</tr>
<tr>
<td>Draco Malfoy</td>
<td>Potions</td>
<td>Snape</td>
<td>A*</td>
</tr>
<tr>
<td>Harry Potter</td>
<td>Potions</td>
<td>Lupin</td>
<td>A+</td>
</tr>
<tr>
<td>Ron Weasley</td>
<td>Potions</td>
<td>Lupin</td>
<td>B+</td>
</tr>
</tbody>
</table>
Details of the Conversion

- Create an entity in the new Evaluation entity set for each instance (row) in the ternary Evaluation relationship.
- In the Student_of relationship, relate each entity in the Evaluation entity set with the corresponding student entity.
- How many students can the Student_of relationship relate an Evaluation entity to?
  - Only one!
- Therefore, the multiplicity of Student_of is many-to-one from Evaluation to Student.
Subclasses: Example

- University Employees, Handout 2 (will be released next week)

All employees have a unique ID. In addition to professors, universities also employ staff. The university pays all its employees a salary. Professors come in three flavors: 9-month appointees, calendar year appointees, and research professors. Each 9-month appointee and research professor has a grant that pays part of the employee’s salary. Calendar year and 9-month professors teach classes while research professors do not.
All employees have a unique ID. In addition to professors, universities also employ staff. The university pays all its employees a salary. Professors come in three flavors: 9-month appointees, calendar year appointees, and research professors. Each 9-month appointee and research professor has a grant that pays part of the employee’s salary. Calendar year and 9-month professors teach classes while research professors do not.

Someone from staff IS A employee
A Professor IS A employee
A Research Professor IS A Professor
A Teacher IS A Professor
A 9-month appointee IS A ??
Subclasses: Example

- University Employees, Handout 2 (will be released next week)

All employees have a unique ID. In addition to professors, universities also employ staff. The university pays all its employees a salary. Professors come in three flavors: 9-month appointees, calendar year appointees, and research professors. Each 9-month appointee and research professor has a grant that pays part of the employee’s salary. Calendar year and 9-month professors teach classes while research professors do not.

Someone from staff IS A employee
A Professor IS A employee
A Research Professor IS A Professor
A Teacher IS A Professor
A 9-month appointee IS A Teacher!
Subclasses: Example

- University Employees, Handout 2
Subclasses in the E/R Model

- A subclass of an entity set E is an entity set F such that
  - each entity in F is an entity in E
  - the entity set F must have at least one attribute or participate in at least one relationship that E does not

- Connect E to F using an *isa* relationship denoted by a triangle

- Convention is to draw E above F

- Each *isa* relationship is one-one but we do not draw the arrows.

- The set of *isa* relationships must form a tree.
Subclasses: Example

- University Students, Handout 2

Students enrolled in a university can be either undergraduates or graduates. Graduate students can be enrolled either in a Masters or a Ph.D. program. Each graduate student must submit a thesis. The thesis can be uniquely identified by its title. Each student can be a TA for at most one course. Furthermore, a course can have at most one graduate student as a TA (it may have multiple undergraduate TAs).
Subclasses: Example

- University Students, Handout 2

Diagram showing relationships between various subclasses related to students and courses.
E/R vs. OO Subclasses

- In object-oriented programming languages, each object is in only one class.
  - A subclass inherits variables and methods from the superclasses.

- In an E/R diagram, an entity has components in all the subclasses to which it belongs
  - If an entity \( e \) has a component in an subclass, then \( e \) has a component in the superclass
  - Does \( e \) have a component in the root?
  - The attributes of \( e \) are the union of the attributes of its components
  - \( e \) participates in all the relationships its components participate in
Components of an Entity

- Prof. Fingers InMany Pies has a 9-month appointment, teaches in one semester every year, and does not teach in the other semester.
- In the other semesters, his research grant pays his salary.
- Which entity sets does he have components in? (using a different isa hierarchy than before)
Components of an Entity

- How do we represent students enrolled in combined Bachelors-Masters programs?
- Such a student has components in multiple entity sets
Components of an Entity

- Such a student has components in multiple entity sets