Database Management Systems

Module 20: Entity-Relationship Model/3

Partha Pratim Das

Department of Computer Science and Engineering
Indian Institute of Technology, Kharagpur

ppd@cse.iitkgp.ac.in

Module Recap

- ER Diagram for ER Models
- Translation of ER Models to Relational Schema

Module Objectives

- To understand extended features of ER Model
- To discuss various design issues

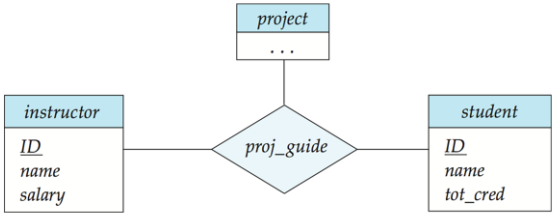
Module Outline

- Extended ER Features
- Design Issues

Extended ER Features

Non-binary Relationship Sets

- Most relationship sets are binary
- There are occasions when it is more convenient to represent relationships as non-binary
- ER Diagram with a Ternary Relationship



Cardinality Constraints on Ternary Relationship

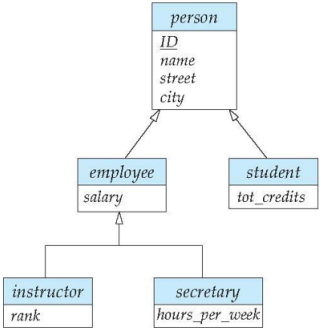
- We allow at most one arrow out of a ternary (or greater degree) relationship to indicate a cardinality constraint
- For example, an arrow from *proj_guide* to *instructor* indicates each student has at most one guide for a project
- If there is more than one arrow, there are two ways of defining the meaning.
  - For example, a ternary relationship R between A, B and C with arrows to B and C could mean
    a) Each A entity is associated with a unique entity from B and C or
    b) Each pair of entities from (A, B) is associated with a unique C entity, and each pair (A ,C) is associated with a unique B
  - Each alternative has been used in different formalisms
  - To avoid confusion we outlaw more than one arrow

Specialization: ISA

- **Top-down design process**: We designate sub-groupings within an entity set that are distinctive from other entities in the set
- These sub-groupings become lower-level entity sets that have attributes or participate in relationships that do not apply to the higher-level entity set
- Depicted by a *triangle* component labeled ISA (e.g., *instructor* "is a" *person*)
- **Attribute inheritance**: A lower-level entity set inherits all the attributes and relationship participation of the higher-level entity set to which it is linked

Specialization: ISA (2)

- **Overlapping**: *employee* and *student*
- **Disjoint**: *instructor* and *secretary*
- Total and Partial



Representing Specialization via Schema

- Method 1:
  - Form a schema for the higher-level entity
  - Form a schema for each lower-level entity set, include primary key of higher-level entity set and local attributes

schema	attributes
person	ID, name, street, city
student	ID, tot_cred
employee	ID, salary

  - Drawback: Getting information about, an *employee* requires accessing two relations, the one corresponding to the low-level schema and the one corresponding to the high-level schema

Representing Specialization as Schema (2)

- Method 2:
  - Form a schema for each entity set with all local and inherited attributes

schema	attributes
person	ID, name, street, city
student	ID, name, street, city, tot_cred
employee	ID, name, street, city, salary

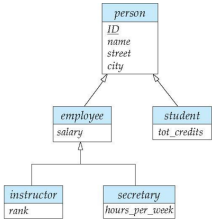
  - Drawback: *name, street* and *city* may be stored redundantly for people who are both students and employees

Generalization

- **Bottom-up design process**: Combine a number of entity sets that share the same features into a higher-level entity set
- Specialization and generalization are simple inversions of each other; they are represented in an ER diagram in the same way
- The terms specialization and generalization are used interchangeably

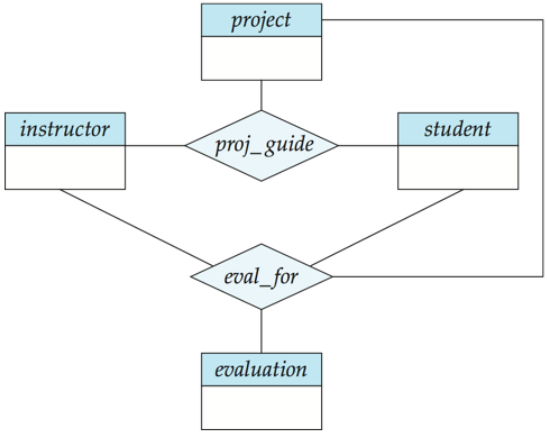
Design Constraints on a Specialization / Generalization

- **Completeness constraint**: Specifies whether or not an entity in the higher-level entity set must belong to at least one of the lower-level entity sets within a generalization
  - **total**: an entity must belong to one of the lower-level entity sets
  - **partial**: an entity need not belong to one of the lower-level entity sets
- Partial generalization is the default. We can specify total generalization in an ER diagram by adding the keyword **total** in the diagram and drawing a dashed line from the keyword to the corresponding hollow arrow-head to which it applies (for a total generalization), or to the set of hollow arrow-heads to which it applies (for an overlapping generalization).
- The *student* generalization is total. All student entities must be either graduate or undergraduate. Because the higher-level entity set arrived at through generalization is generally composed of only those entities in the lower-level entity sets, the completeness constraint for a generalized higher-level entity set is usually total.



Aggregation

- Consider the ternary relationship *proj_guide*, which we saw earlier
- Suppose we want to record evaluations of a student by a guide on a project

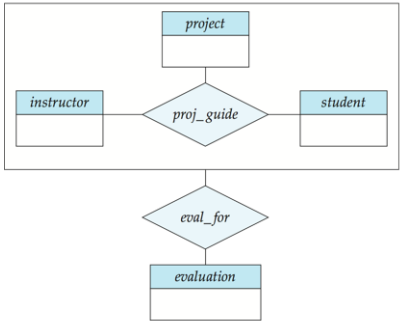


Aggregation (2)

- Relationship sets *eval_for* and *proj_guide* represent overlapping information
  - Every *eval_for* relationship corresponds to a *proj_guide* relationship
  - However, some *proj_guide* relationships may not correspond to any *eval_for* relationships
    - So we cannot discard the *proj_guide* relationship
- Eliminate this redundancy via *aggregation*
  - Treat relationship as an abstract entity
  - Allows relationships between relationships
  - Abstraction of relationship into new entity

Aggregation (3)

- Eliminate this redundancy via *aggregation* without introducing redundancy, the following diagram represents:
  - A student is guided by a particular instructor on a particular project
  - A student, instructor, project combination may have an associated evaluation



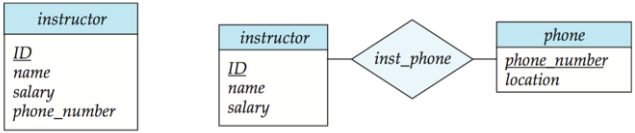
Representing Aggregation via Schema

- To represent aggregation, create a schema containing
  - Primary key of the aggregated relationship,
  - The primary key of the associated entity set
  - Any descriptive attributes
- In our example:
  - The schema
    textiteval_for is:
    eval_for (s_ID, project_id, i_ID, evaluation_id)
  - The schema *proj_guide* is redundant

Design Issues

Entities vs. Attributes

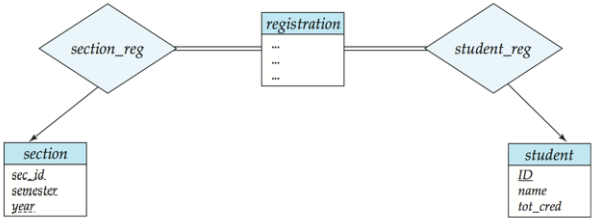
- Use of entity sets vs. attributes



- Use of phone as an entity allows extra information about phone numbers (plus multiple phone numbers)

Entities vs Relationship Sets

- **Use of entity sets vs. relationship sets**
  Possible guideline is to designate a relationship set to describe an action that occurs between entities



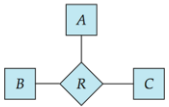
- **Placement of relationship attributes**
  For example, attribute date as attribute of advisor or as attribute of student

Binary vs Non-Binary Relationships

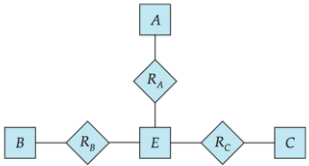
- Although it is possible to replace any non-binary (n-ary, for n $>$ 2) relationship set by a number of distinct binary relationship sets, a n-ary relationship set shows more clearly that several entities participate in a single relationship
- Some relationships that appear to be non-binary may be better represented using binary relationships
  - For example, a ternary relationship *parents*, relating a child to his/her father and mother, is best replaced by two binary relationships, *father* and *mother*
    - Using two binary relationships allows partial information (e.g., only mother being known)
  - But there are some relationships that are naturally non-binary
    - Example: *proj_guide*

Binary vs Non-Binary Relationships (2): Conversion

- In general, any non-binary relationship can be represented using binary relationships by creating an artificial entity set.
  - Replace R between entity sets A, B and C by an entity set E, and three relationship sets:
    1. R_A, relating E and A
    2. R_B, relating E and B
    3. R_C, relating E and C
  - Create an identifying attribute for E and add any attributes of R to E
  - For each relationship (a_i, b_i, c_i) in R, create
    a) a new entity e_i in the entity set E
    b) add (e_i, a_i) to R_A
    c) add (e_i, b_i) to R_B
    d) add (e_i, c_i) to R_C



(a)



(b)

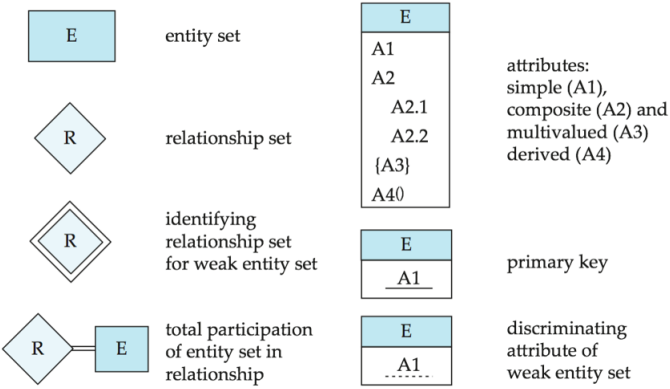
Binary vs Non-Binary Relationships (3): Conversion

- Also need to translate constraints
  - Translating all constraints may not be possible
  - There may be instances in the translated schema that cannot correspond to any instance of R.
    - Exercise: *add constraints to the relationships R_A, R_B and R_C* to ensure that a newly created entity corresponds to exactly one entity in each of entity sets —A, B and C
  - We can avoid creating an identifying attribute by making E, a weak entity set (described shortly) identified by the three relationship sets

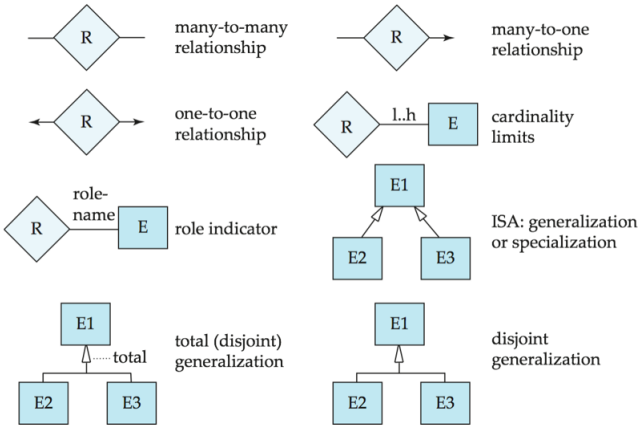
ER Design Decisions

- The use of an attribute or entity set to represent an object
- Whether a real-world concept is best expressed by an entity set or a relationship set
- The use of a ternary relationship versus a pair of binary relationships
- The use of a strong or weak entity set
- The use of specialization/generalization – contributes to modularity in the design
- The use of aggregation – can treat the aggregate entity set as a single unit without concern for the details of its internal structure

Symbols Used in ER Notation

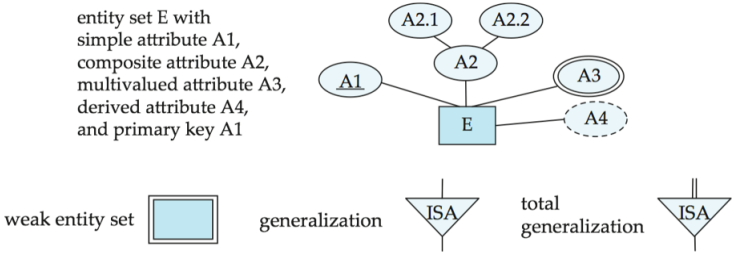


Symbols Used in ER Notation (2)

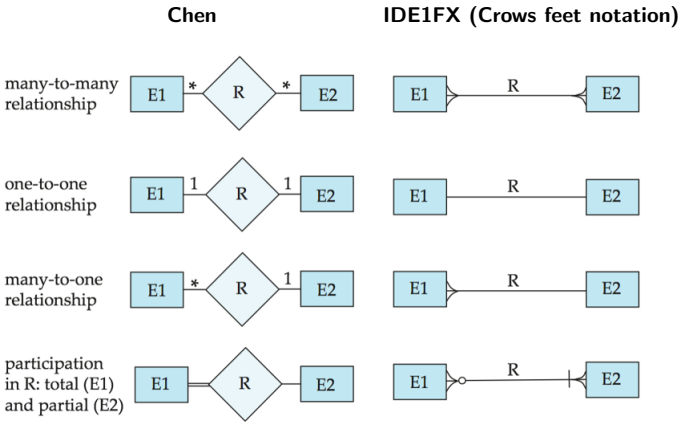


Symbols Used in ER Notation (3): Alternate

- Chen, IDE1FX,...



Symbols Used in ER Notation (4): Alternates



Module Summary

- Discussed the extended features of ER Model
- Deliberated on various design issues