Chapter 7: Entity-Relationship Model

Edited by Radhika Sukapuram

Database System Concepts, 7th Ed.

Reduction to Relation Schemas

Reduction to Relation Schemas

- Entity sets and relationship sets can be expressed uniformly as *relation schemas*.
- A database which conforms to an E-R diagram can be represented by a collection of schemas.
- For each entity set and relationship set there is
  - a unique schema that is assigned the name of the corresponding entity set or relationship set.
- Each schema has a number of columns (generally corresponding to attributes), which have unique names.

Representing Entity Sets

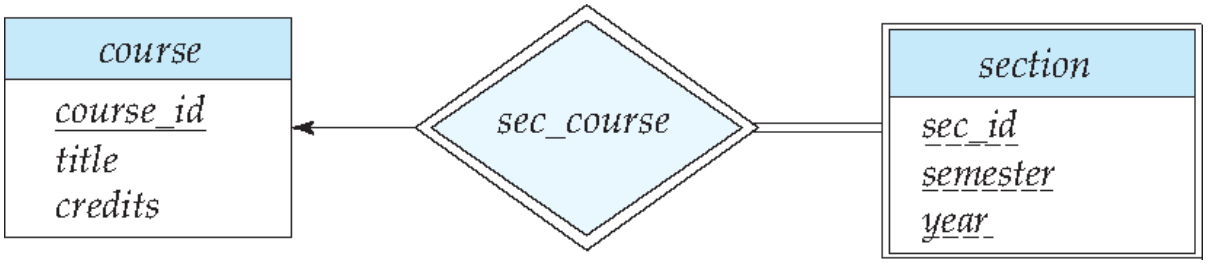
- A strong entity set reduces to a schema with the same attributes

  student(<u>ID</u>, name, tot_cred)

  and the same primary keys

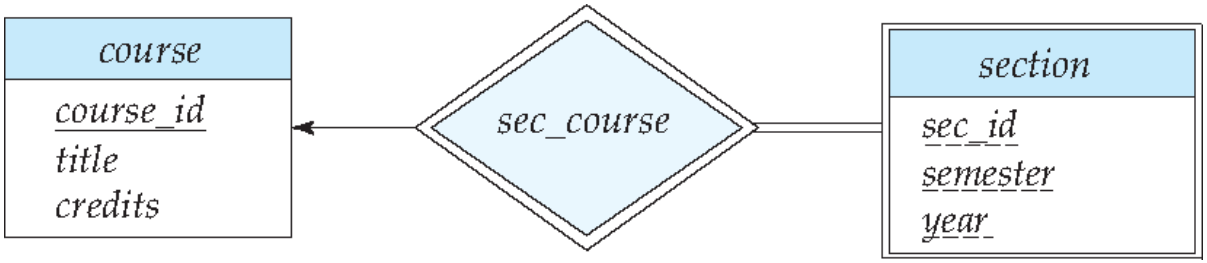
- A weak entity set becomes a table that includes a column for the primary key of the identifying strong entity set

  section (<u>course_id, sec_id, sem, year</u>)

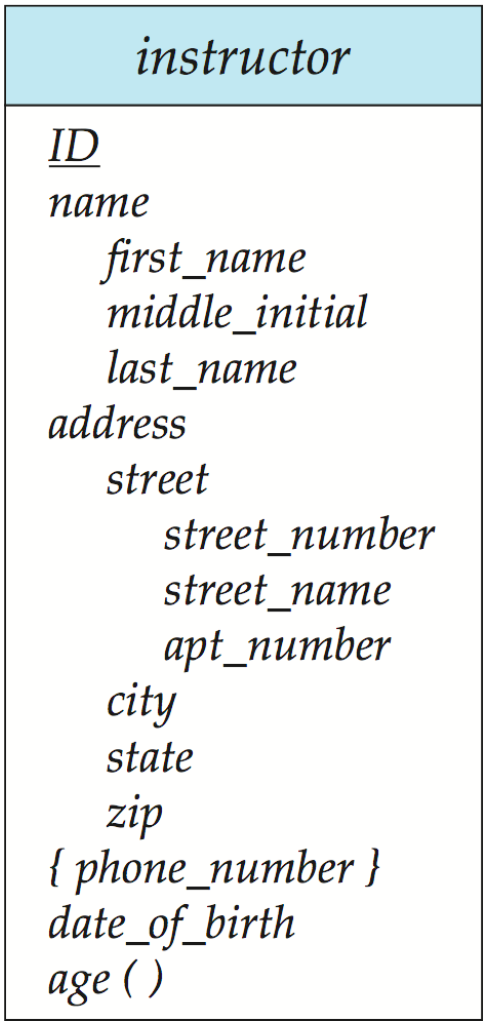


Representing entity sets cont.

- A foreign key constraint on section is created
  - *course_id* of *section* references the primary key *course_id* of *course*
  - with on delete cascade



Representation of Entity Sets with Composite Attributes



- Composite attributes are flattened out by creating a separate attribute for each component attribute
  - the schema corresponding to the entity set *instructor* has three attributes *name_first_name* , *name_middle_initial* and *name_last_name*
    - Prefix omitted if there is no ambiguity (*name_first_name* could be *first_name)*
- Ignoring multivalued attributes, extended *instructor* schema is
  - *instructor(ID, first_name, middle_initial, last_name, street_number, street_name, apt_number, city, state, zip_code, date_of_birth)*

Representation of Entity Sets with Multivalued Attributes

- Example: Multivalued attribute *phone_number* of *instructor* is represented by a schema:
  inst_phone= (*ID*, *phone_number*)
  - Why is *phone_number* a part of the primary key ?
- A multivalued attribute *M* of an entity *E* is represented by a separate schema *EM*
- Schema *EM* has attributes corresponding to the primary key of *E* and an attribute corresponding to multivalued attribute *M*

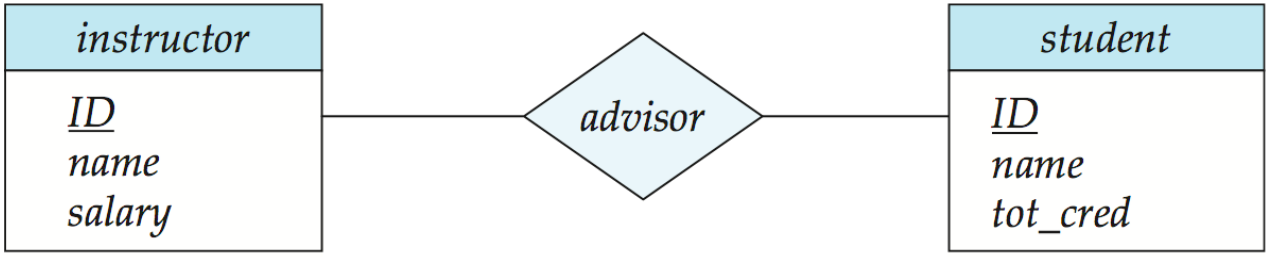
Representation of Entity Sets with Multivalued Attributes cont.

- Each value of the multivalued attribute maps to a separate tuple of the relation on schema *EM*
  - For example, an *instructor* entity with primary key 22222 and phone numbers 456-7890 and 123-4567 maps to two tuples: (22222, 456-7890) and (22222, 123-4567)
- Create a foreign key constraint from EM to E
  - primary key derived from E of EM referencing E
  - *id* of *inst_phone* references *instructor*
- Exercise: What if E has only two attributes : one simple attribute which is a primary key and a multivalued attribute ?

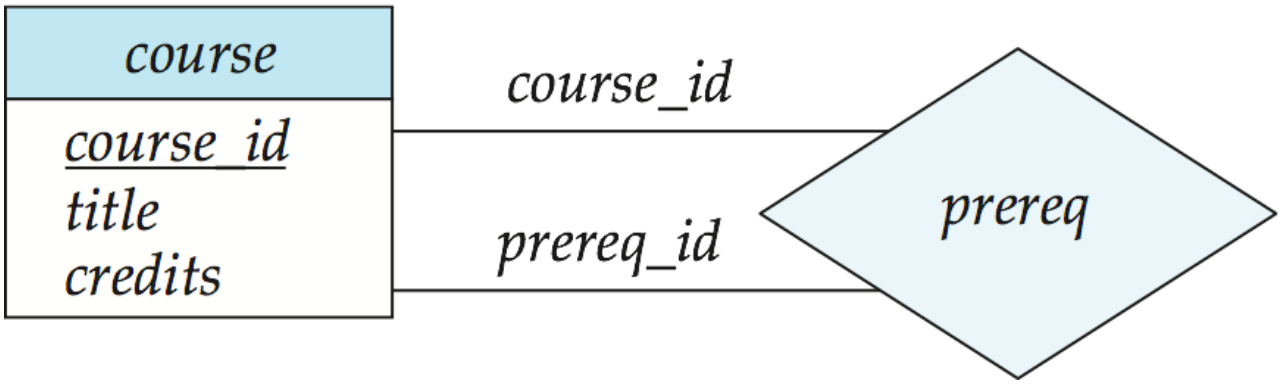
Representing Relationship Sets

- A many-to-many relationship set is represented as
  - a schema with attributes for the primary keys of the two participating entity sets
  - and any descriptive attributes of the relationship set.
- Example: schema for relationship set *advisor*

  advisor = (*s_id, i_id*)
- Two foreign keys, with *s_id* referencing student and *i_id* referencing *instructor*



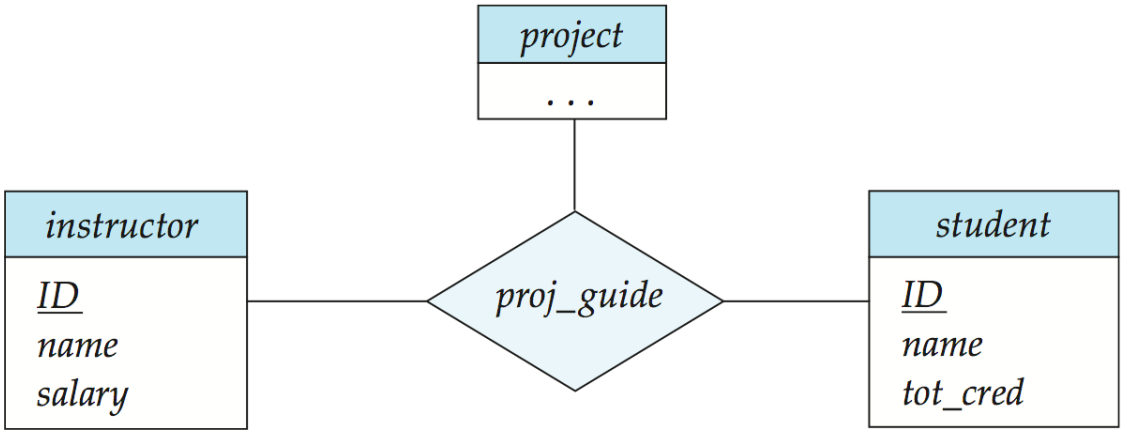
Representing Relationship Sets



prereq (course_id, prereq_id)

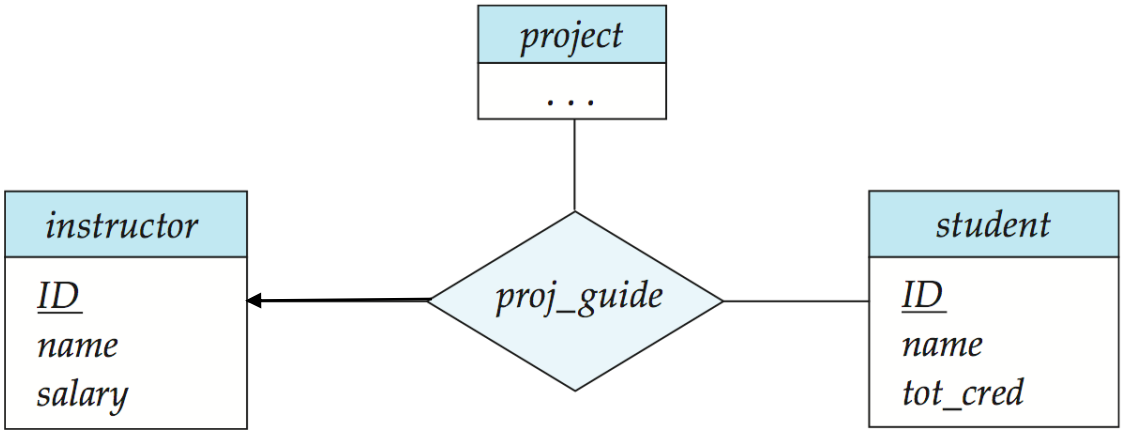
Non-binary Relationship Sets

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



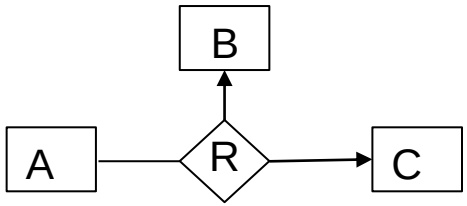
Cardinality Constraints on Ternary Relationships

- 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



Cardinality Constraints on Ternary Relationships cont.

- 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
    1. Each *A* entity is associated with a unique entity from *B* and *C* or
    2. 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



Representing Relationship Sets cont.

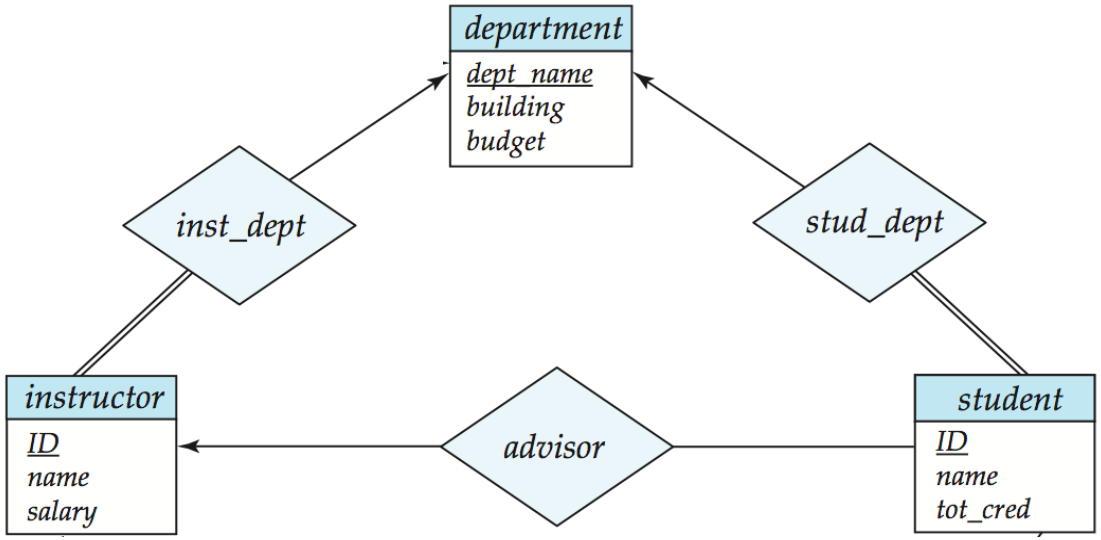
- Exercise: What are the primary keys for schemes translated from binary relationships with various cardinality mappings ?
- n-ary relationship without any arrows on the edges
  - Union of the primary key attributes of the participating relationships
- n-ary relationship with an arrow on one of its edges (only 1 arrow is allowed)
  - Union of primary key of the entity sets not on the arrow side of the relationship

Representing Relationship Sets cont.

- Foreign keys must also be created from relation R
  - For each entity set E_i related to R
  - Create a foreign key from R
  - With the attributes of R that were derived from the primary key attributes of E_i
  - Referencing the primary key of the relation representing E_i
- Exercise : Are foreign keys always created from relations created from relationship sets to the relations created from the entity sets they were derived from ?

Combination of Schemas

- Instead of creating a schema for relationship set *inst_dept*
  - add an attribute *dept_name* to the schema arising from entity set *instructor*
- Many-to-one and one-to-many relationship sets that are total on the many-side
  - can be represented by adding extra attributes to the “many” side
  - containing the primary key of the “one” side
  - Add a foreign key to the “many” side referencing the “one” side
  - Primary key: the primary key of entity set into whose schema the merge took place



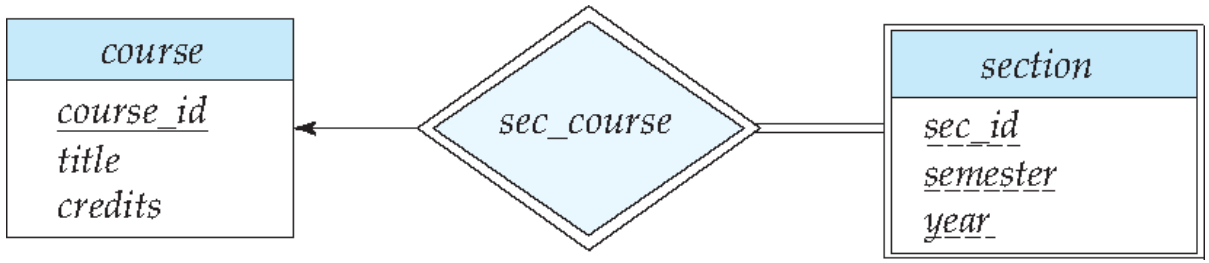
Combination of Schemas (Cont.)

- For one-to-one relationship sets, either side can be chosen to act as the “many” side
  - That is, an extra attribute can be added to either of the tables corresponding to the two entity sets
- If participation is *partial* on the “many” side (*instructor*),
  - replacing a schema by an extra attribute in the schema corresponding to the “many” side , containing the primary key of the “one” side (*dept_name)* could result in null values (for *dept_name*)

Redundancy of Schemas

- The schema corresponding to a relationship set linking a weak entity set to its identifying strong entity set is redundant.

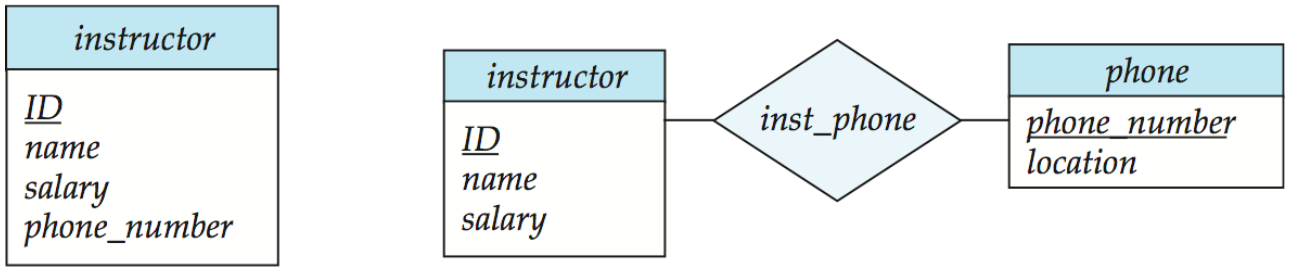
- Example: The *section* schema already contains the attributes that would appear in the *sec_course* schema



Design Issues

Entities vs. Attributes

- Use of entity sets vs. attributes



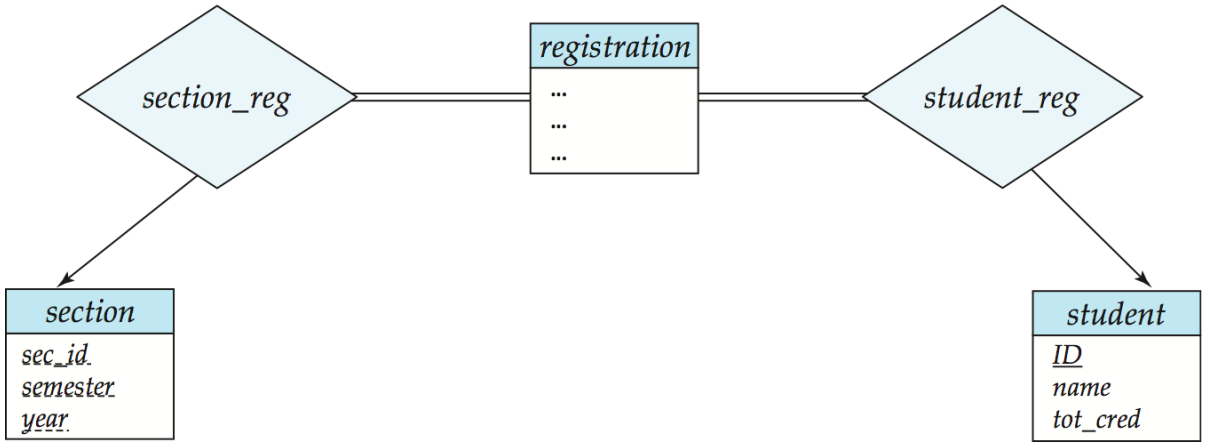
- Use of phone as an entity allows extra information about phone numbers (plus multiple phone numbers)
- An entity is more general than an attribute
- Use an entity when generality is more useful

Entities vs attributes

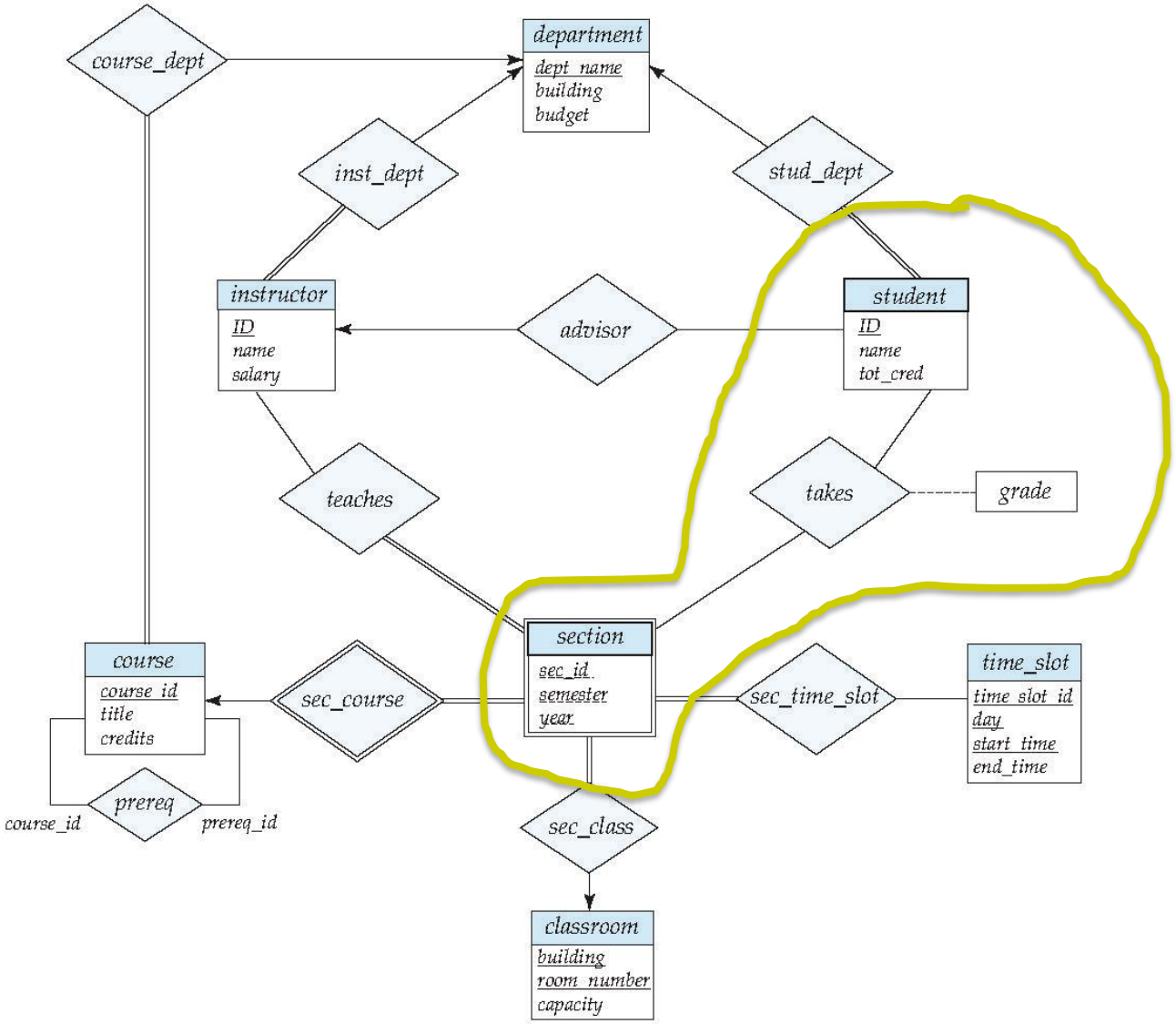
- Common mistakes
  - Instead of using a relationship (*advisor*)
    - using the primary key of an entity set (*id* of *student*) as an attribute of another entity set (*instructor*)
    - *id* of a *student* must not be an attribute of *instructor* even if the instructor advises only one student
    - Instead, use an *advisor* relationship
      - Makes the connection explicit
      - A better representation of reality
  - Primary key attributes of entity sets are made attributes of the relationship set
    - *id* of a *student* and *id* of *instructor* must not be attributes of *advisor*
      - Because the primary key attributes are implicit in the relationship!

Entities vs. Relationship sets

- **Use of entity sets vs. relationship sets**
  - A student takes a section of a course
  - Instead, create a course registration record for each course that the student takes
- Possible guidelines
  - If more attributes are associated with registration, it is an entity in its own right
  - designate a relationship set to describe an action that occurs between entities



E-R Diagram for a University Enterprise



Entities vs. Relationship sets

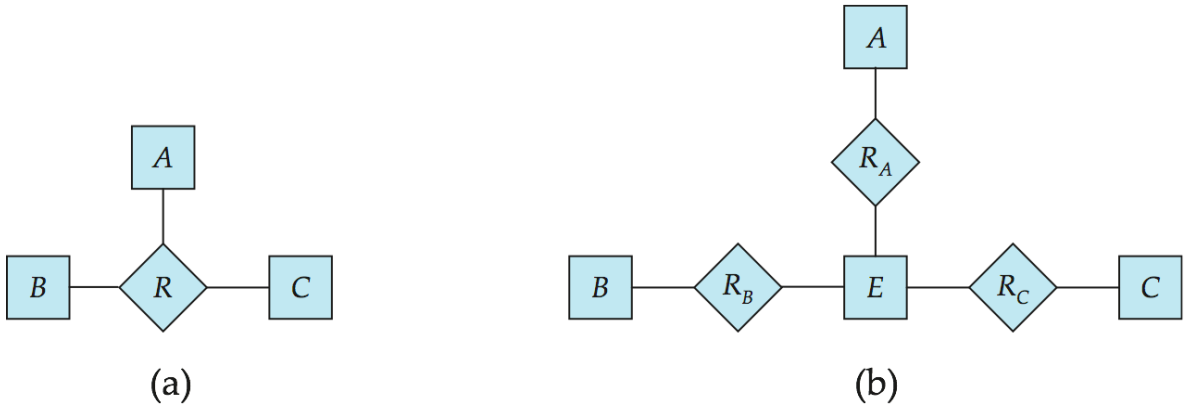
- **Placement of relationship attributes**
  - For example, attribute *date* (date on which an instructor starts advising a student) as attribute of *advisor* or as attribute of *student*
  - One-to-many from *instructor* - *student*: *date* may be associated with *student (*the entity on the many side)
  - One-to-one: either of the participating entities
  - Also depends on the characteristics of the enterprise being modelled
    - *date* may be associated with *advisor* to indicate that it is the date on which the advisor was associated with the student and not the student's date of entry into the university
  - Many-to-many: associated with the relationship set

Binary Vs. Non-Binary Relationships

- It is possible to replace any non-binary (*n*-ary, for *n* > 2) relationship set by a number of distinct binary relationship sets
  - Why ? some data models support only binary relationship sets
- But an *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 one parent being known)
    - Otherwise null values will need to be used
  - But there are some relationships that are naturally non-binary
    - Example: *proj_guide*

Converting Non-Binary Relationships to Binary Form

- 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
    1. a new entity e_i in the entity set *E*
    2. add (e_i , a_i) to R_A
    3. add (e_i , b_i) to R_B
    4. add (e_i , c_i) to R_C



Converting Non-Binary Relationships (Cont.)

- Also need to translate constraints
  - Translating all constraints may not be possible
    - Each pair of entities from A and B may be associated with at most one entity C – how to translate this ?

Restricting to binary relationship sets

- Restricting to binary relationship sets is not always desirable:
  - Increases space and complexity
  - Does not reflect reality intuitively
  - Translating all constraints may not be possible