CS425 – Summer 2016
Jason Arnold
Chapter 7: Entity-Relationship Model

Partially taken from

Klaus R. Dittrich

modified from:

Database System Concepts, 6th Ed.

Requirement Analysis Example Zoo

- The zoo stores information about animals, cages, and zoo keepers.
- Animals are of a certain species and have a name. For each animal we want to record its weight and age.
- Each cage is located in a section of the zoo. Cages can house animals, but there may be cages that are currently empty. Cages have a size in square meter.
- Zoo keepers are identified by their social security number. We store a first name, last name, and for each zoo keeper. Zoo keepers are assigned to cages they have to take care of (clean, …). Each cage that is not empty has a zoo keeper assigned to it. A zoo keeper can take care of several cages. Each zoo keeper takes care of at least one cage.

Modeling – ER model

- A *database* can be modeled as:
  - a collection of entities,
  - relationship among entities.
- An **entity** is an object that exists and is distinguishable from other objects.
  - Example: specific person, company, event, plant
- Entities have **attributes**
  - Example: people have *names* and *addresses*
- An **entity set** is a set of entities of the same type that share the same properties.
  - Example: set of all persons, companies, trees, holidays

Entity Sets *instructor* and *student*

instructor_ID	instructor_name
76766	Crick
45565	Katz
10101	Srinivasan
98345	Kim
76543	Singh
22222	Einstein

instructor

student-ID	student_name
98988	Tanaka
12345	Shankar
00128	Zhang
76543	Brown
76653	Aoi
23121	Chavez
44553	Peltier

student

Relationship Sets

- A **relationship** is an association among several entities

  Example:

44553 (Peltier)	<u>*advisor*</u>	22222 (<u>Einstein</u>)
student entity	relationship set	*instructor* entity

- A **relationship set** is a mathematical relation among $n \geq 2$ entities, each taken from entity sets

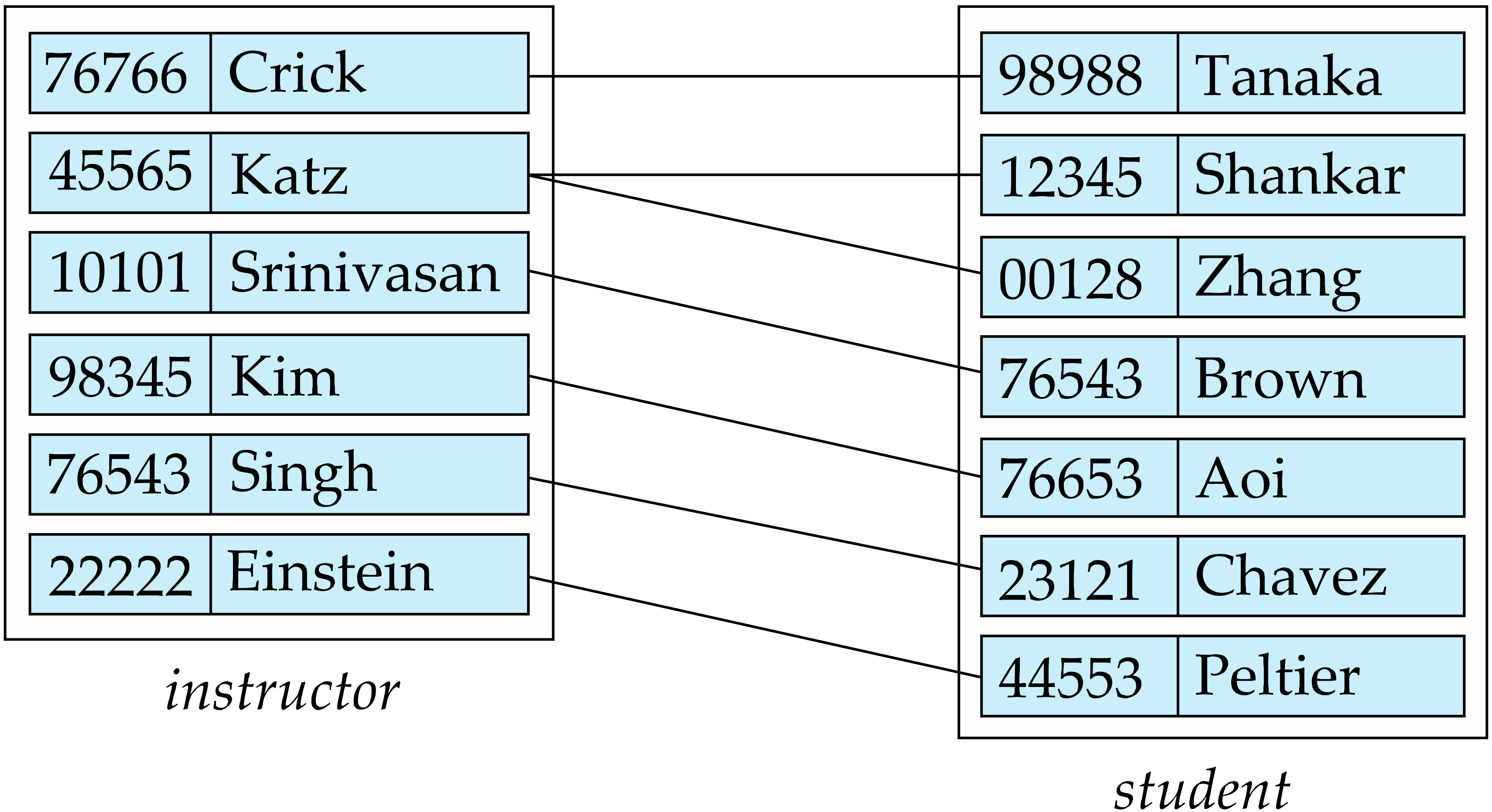
  $$\{(e_1, e_2, \ldots e_n) \mid e_1 \in E_1, e_2 \in E_2, \ldots, e_n \in E_n\}$$

  where $(e_1, e_2, \ldots, e_n)$ is a relationship

  - Example:

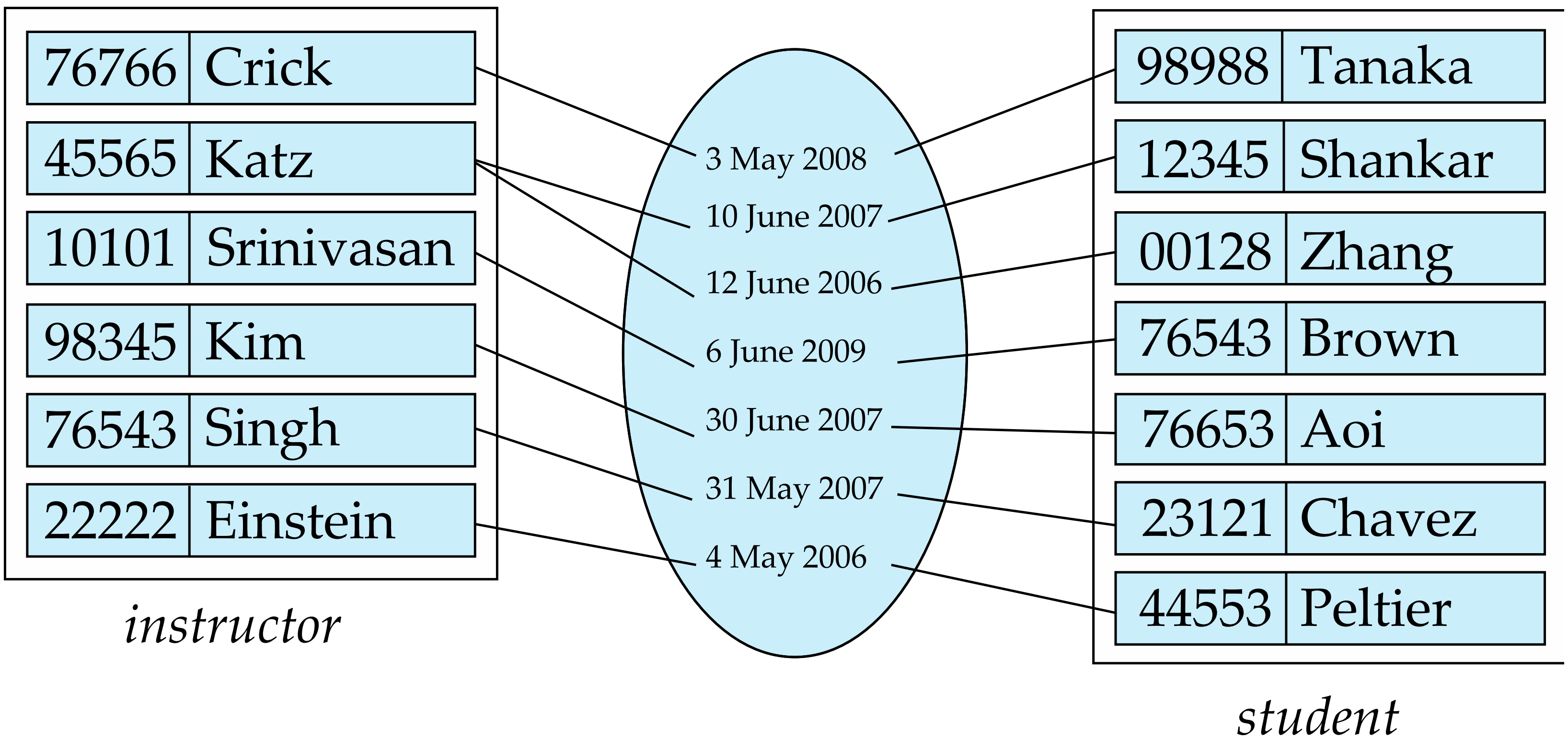
    $(44553, 22222) \in$ *advisor*

Relationship Set *advisor*



Relationship Sets (Cont.)

- An **attribute** can also be property of a relationship set.
- For instance, the *advisor* relationship set between entity sets *instructor* and *student* may have the attribute *date* which tracks when the student started being associated with the advisor



Degree of a Relationship Set

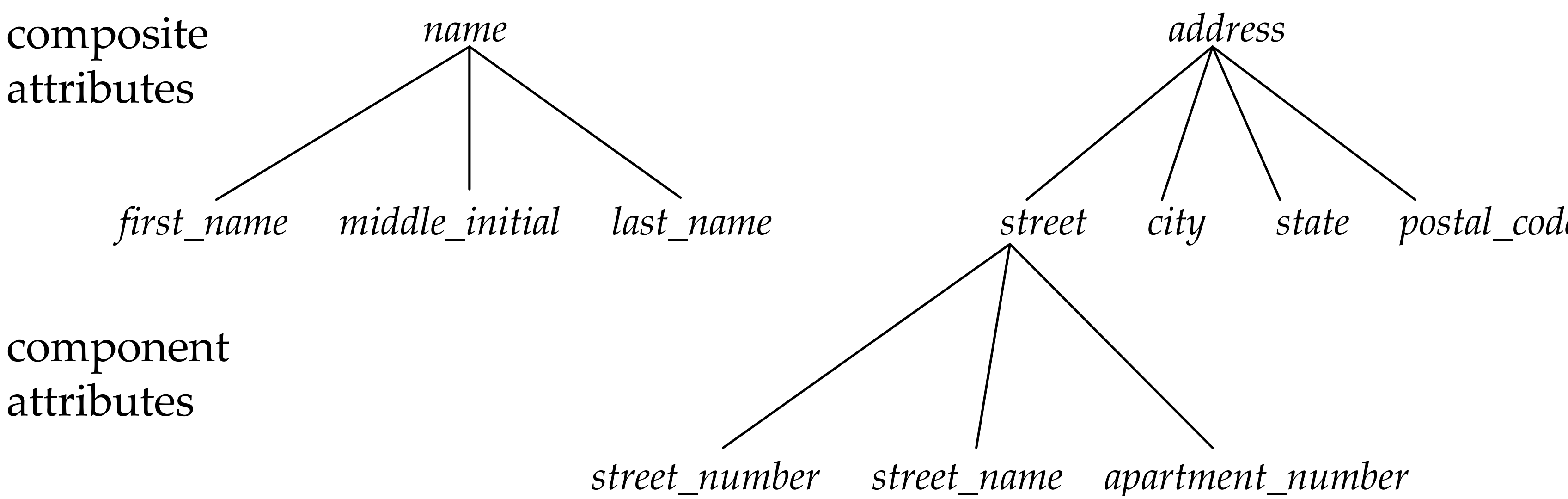
- **binary relationship**
  - involve two entity sets (or degree two).
- Relationships between more than two entity sets are rare. Most relationships are binary. (More on this later.)
  - Example: *students* work on research *projects* under the guidance of an *instructor*.
  - relationship *proj_guide* is a ternary relationship between *instructor, student,* and *project*

Attributes

- An entity is represented by a set of attributes, that are descriptive properties possessed by all members of an entity set.
  - Example:

    instructor = (*ID, name, street, city, salary*)
    course= (*course_id, title, credits*)
- **Domain** – the set of permitted values for each attribute
- Attribute types:
  - **Simple** and **composite** attributes.
  - **Single-valued** and **multivalued** attributes
    - Example: multivalued attribute: *phone_numbers*
  - **Derived** attributes
    - Can be computed from other attributes
    - Example: age, given date_of_birth

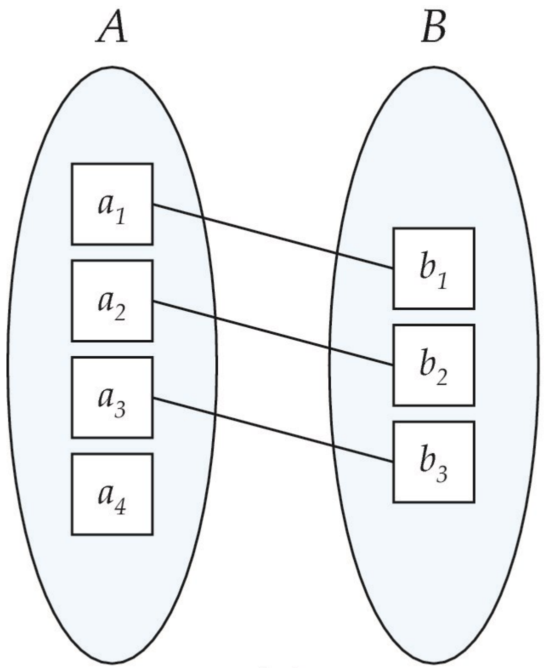
Composite Attributes



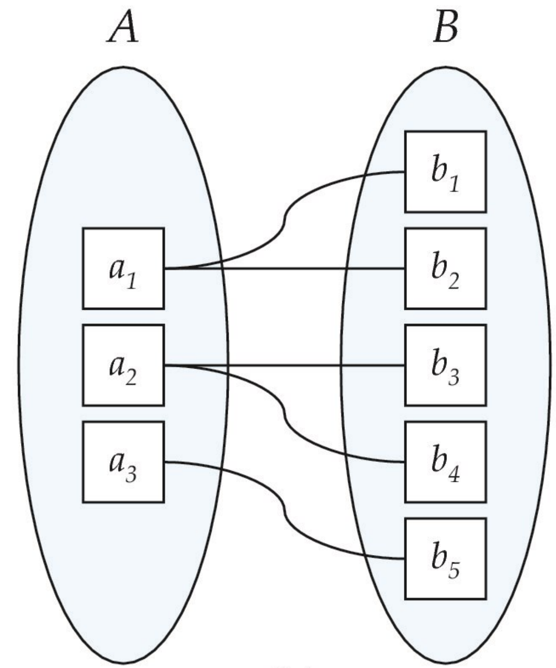
Mapping Cardinality Constraints

- Express the number of entities to which another entity can be associated via a relationship set.
- For a binary relationship set the mapping cardinality must be one of the following types:
  - One to one (1-1)
  - One to many (1-N)
  - Many to one (N-1)
  - Many to many (N-M)

Mapping Cardinalities



(a)



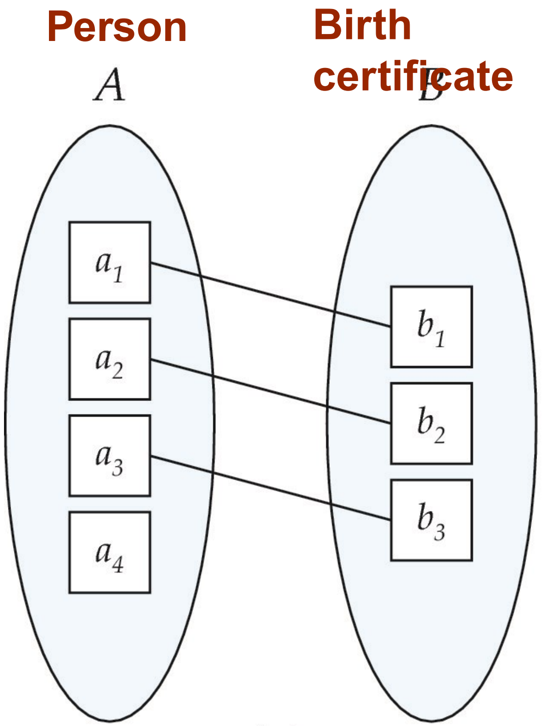
(b)

One to one

One to many

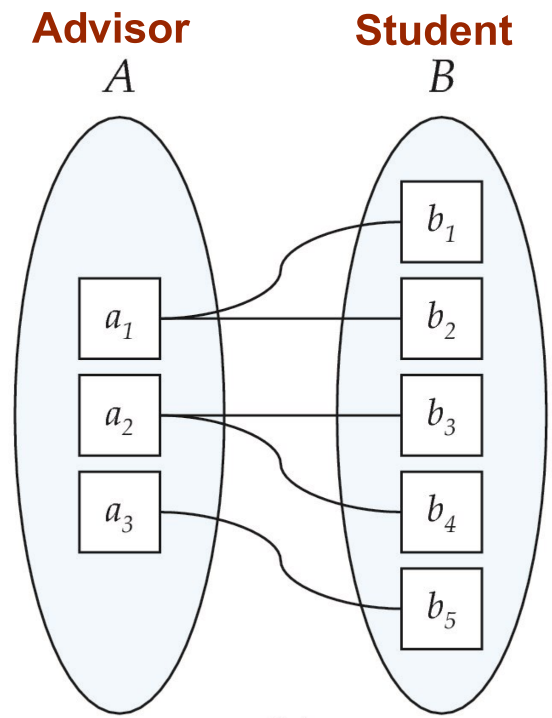
Note: Some elements in *A* and *B* may not be mapped to any elements in the other set

Mapping Cardinalities Example



(a)

One to one



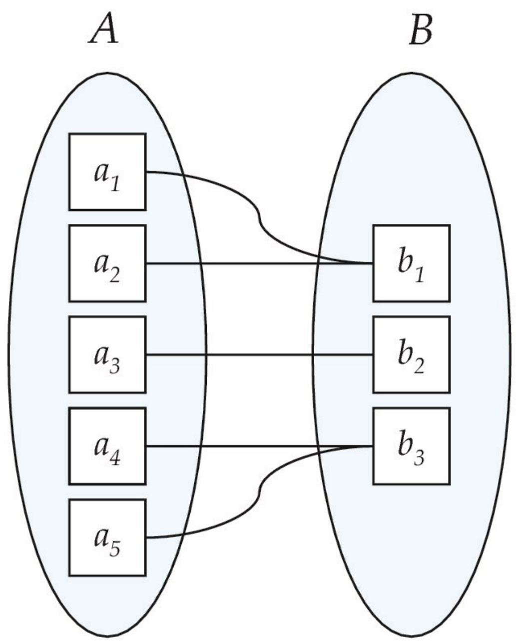
(b)

One to many

Note: Some elements in *A* and *B* may not be mapped to any elements in the other set

Mapping Cardinalities Example

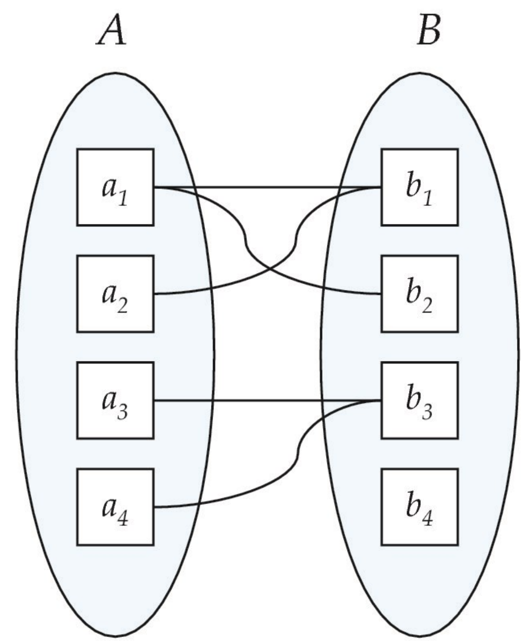
Employee **Department**



(a)

Many to one

Student **Course**



(b)

Many to many

Note: Some elements in A and B may not be mapped to any elements in the other set

Mapping Cardinality Constraints Cont.

- What if we allow some elements to not be mapped to another element?
  - E.g., 0:1 – 1
- For a binary relationship set the mapping cardinality must be one of the following types:

- **1-1**
  - 1-1
  - 0:1-1
  - 1-0:1
  - 0:1-0:1
- **1-N**
  - 0:1-N
  - 0:1-0:N
  - 1-N
  - 1-0:N
- **N-1**
  - N-1
  - N-0:1
  - 0:N-1
  - 0:N-0:1
- **N-M**
  - N-M
  - N-0:M
  - 0:N-M
  - 0:N-0:M

Mapping Cardinality Constraints Cont.

- Typical Notation
  - (0:1) – (1:N)

Keys

- A **super key** of an entity set is a set of one or more attributes whose values uniquely determine each entity.
- A **candidate key** of an entity set is a minimal super key
  - *ID* is candidate key of *instructor*
  - *course_id* is candidate key of *course*
- Although several candidate keys may exist, one of the candidate keys is selected to be the **primary key**.
- **Note: Basically the same as for relational model**

Keys for Relationship Sets

- The combination of primary keys of the participating entity sets forms a super key of a relationship set.
  - (*s_id, i_id*) is the super key of *advisor*
  - *NOTE: this means* ***a pair of entities can have at most one relationship in a particular relationship set.***
    - Example: if we wish to track multiple meeting dates between a student and her advisor, we cannot assume a relationship for each meeting. We can use a multivalued attribute though or model meeting as a separate entity
- Must consider the mapping cardinality of the relationship set when deciding what are the candidate keys
- Need to consider semantics of relationship set in selecting the *primary key* in case of more than one candidate key

Keys for Relationship Sets Cont.

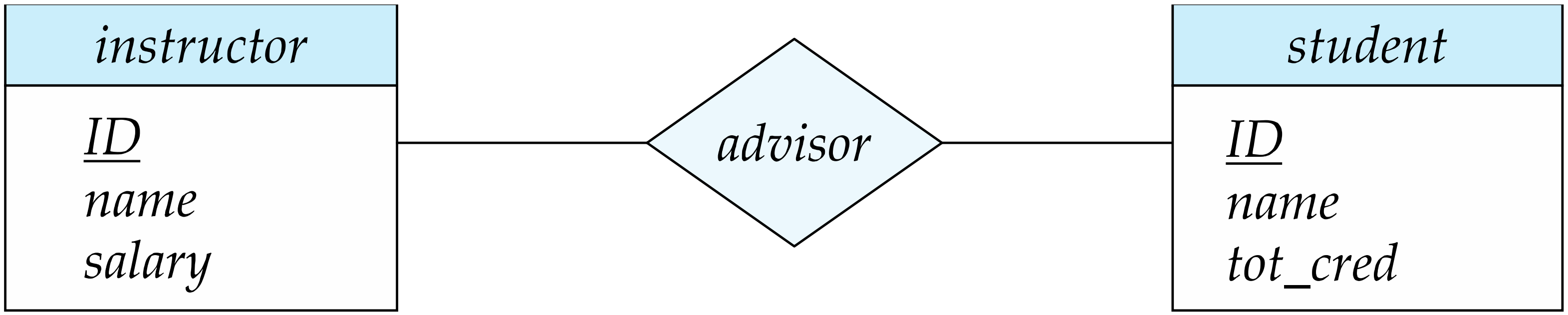
- Must consider the mapping cardinality of the relationship set when deciding what are the candidate keys
  - 1-1: both primary keys are candidate keys
    - Example: **hasBc**: (Person-Birthcertificate)
  - N-1: the N side is the candidate key
    - Example: **worksFor**: (Instructor-Department)
  - N-M: the combination of both primary keys
    - Example: **takes**: (Student-Course)

Redundant Attributes

- Suppose we have entity sets
  - *instructor*, with attributes including *dept_name*
  - *department*

  and a relationship
  - *inst_dept* relating *instructor* and *department*
- Attribute *dept_name* in entity *instructor* is redundant since there is an explicit relationship *inst_dept* which relates instructors to departments
  - The attribute replicates information present in the relationship, and should be removed from *instructor*
  - BUT: when converting back to tables, in some cases the attribute gets reintroduced, as we will see.

E-R Diagrams



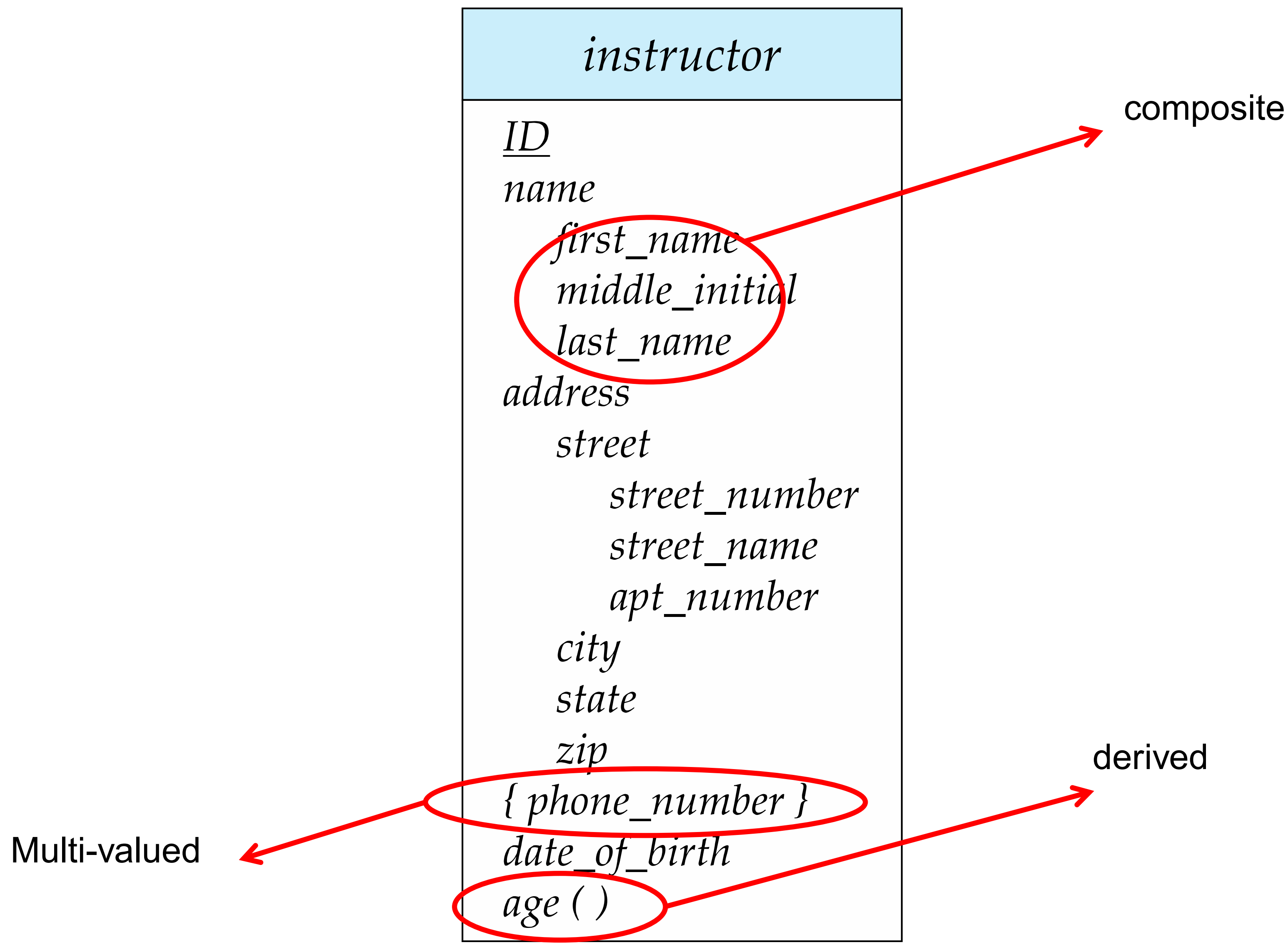
- Rectangles represent entity sets.
- Diamonds represent relationship sets.
- Attributes listed inside entity rectangle
- Underline indicates primary key attributes

Entity With Composite, Multivalued, and Derived Attributes

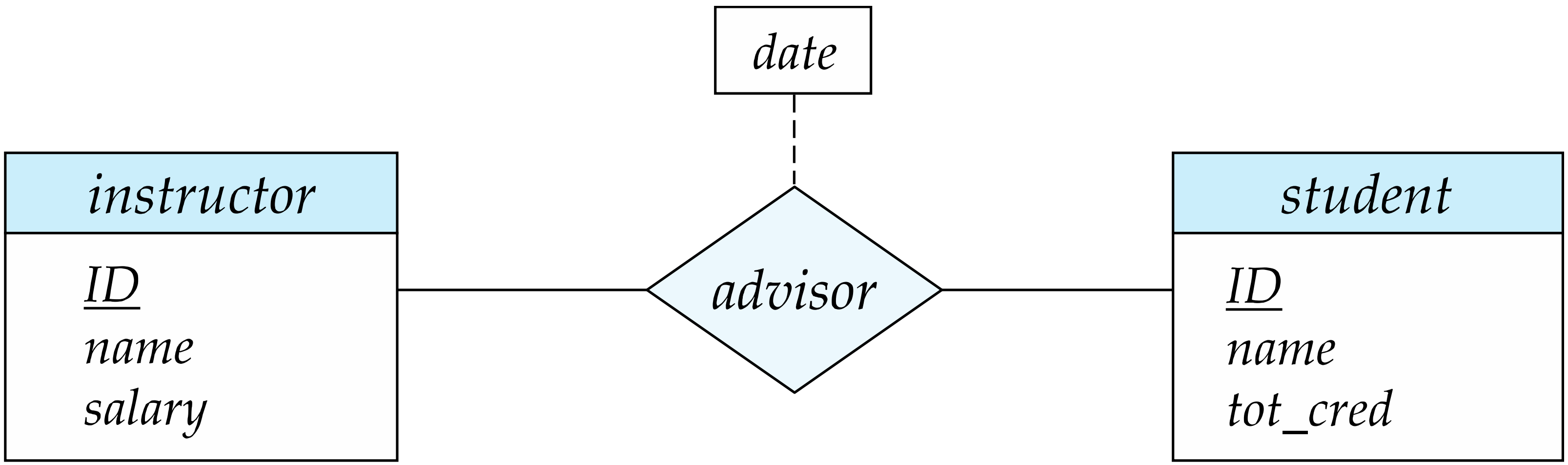
instructor

- <u>ID</u>
- *name*
  - *first_name*
  - *middle_initial*
  - *last_name*
- *address*
  - *street*
    - *street_number*
    - *street_name*
    - *apt_number*
  - *city*
  - *state*
  - *zip*
- *{ phone_number }*
- *date_of_birth*
- *age ( )*

Entity With Composite, Multivalued, and Derived Attributes

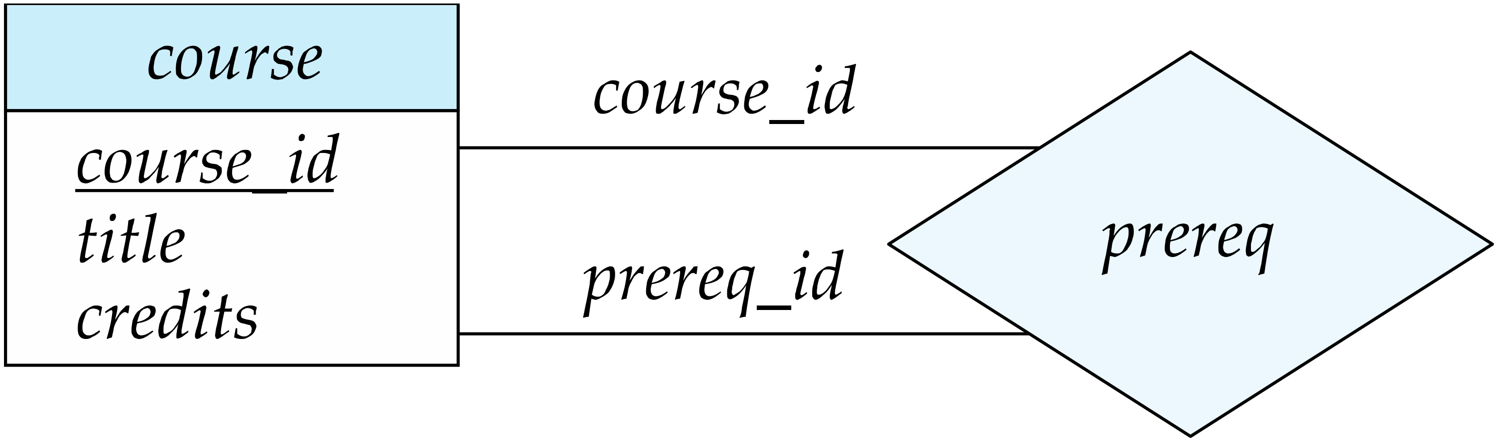


Relationship Sets with Attributes



Roles

- Entity sets of a relationship need not be distinct
  - Each occurrence of an entity set plays a "role" in the relationship
- The labels "*course_id*" and "*prereq_id*" are called **roles**.

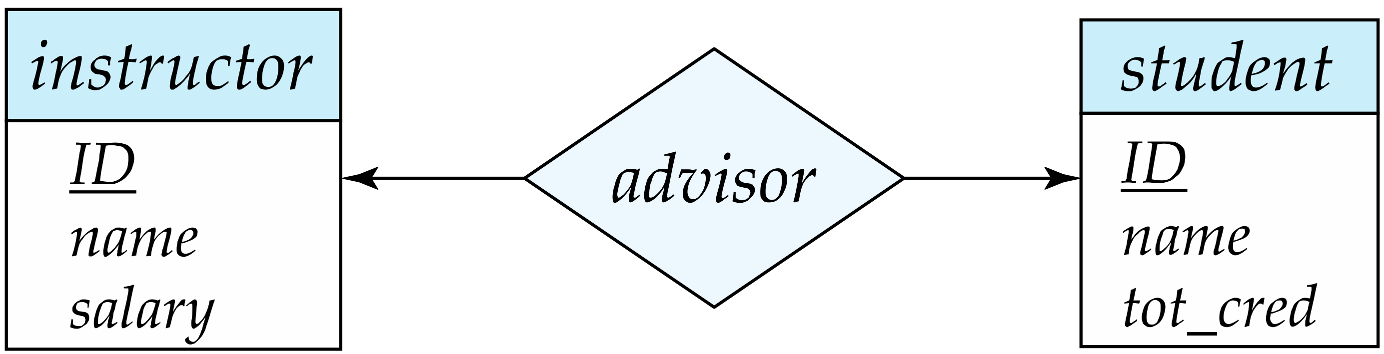


Cardinality Constraints

- We express cardinality constraints by drawing either a directed line ($\rightarrow$), signifying "one," or an undirected line (—), signifying "many," between the relationship set and the entity set.
- One-to-one relationship:
  - A student is associated with at most one *instructor* via the relationship *advisor*
  - A *student* is associated with at most one *department* via *stud_dept*

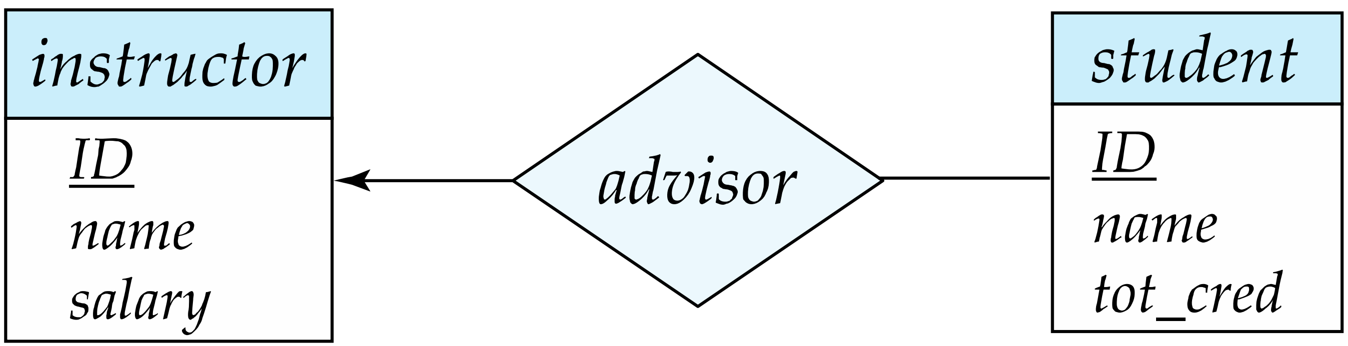
One-to-One Relationship

- one-to-one relationship between an *instructor* and a *student*
  - an instructor is associated with at most one student via *advisor*
  - and a student is associated with at most one instructor via *advisor*



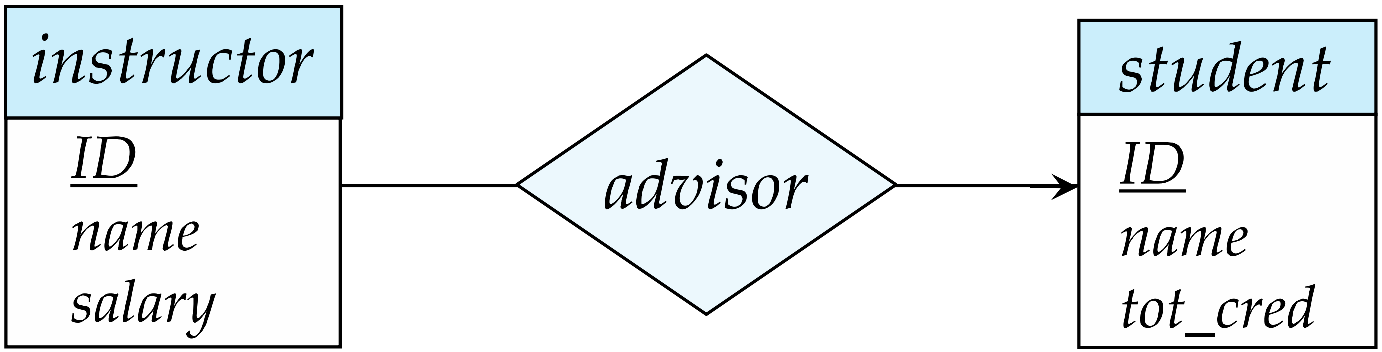
One-to-Many Relationship

- one-to-many relationship between an *instructor* and a *student*
  - an instructor is associated with several (including 0) students via *advisor*
  - a student is associated with at most one instructor via advisor,



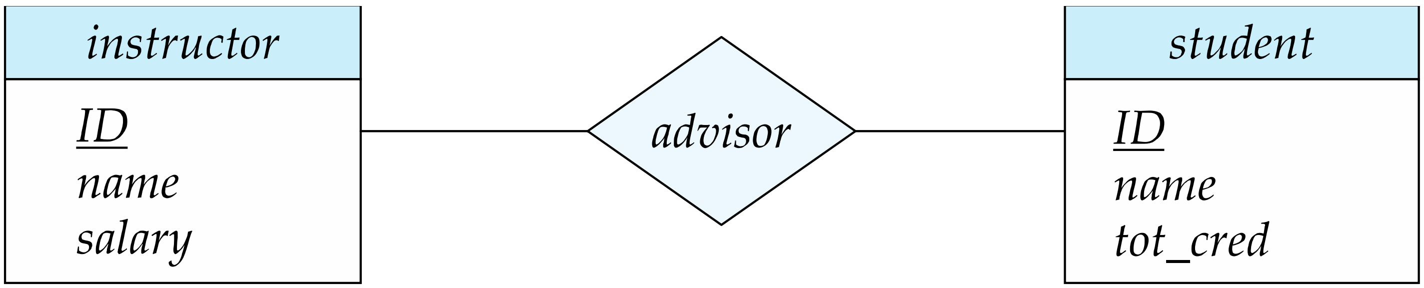
Many-to-One Relationships

- In a many-to-one relationship between an *instructor* and a *student,*
  - an instructor is associated with at most one student via *advisor*,
  - and a student is associated with several (including 0) instructors via *advisor*



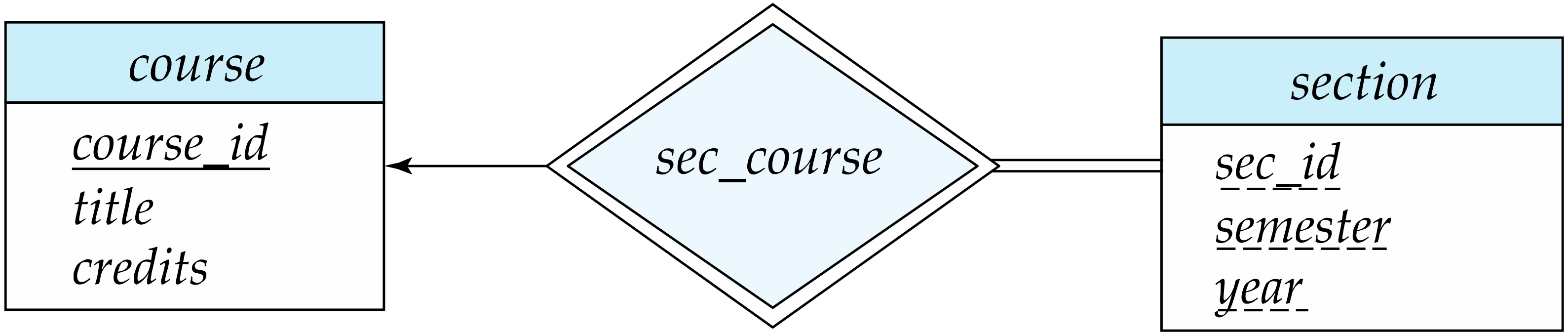
Many-to-Many Relationship

- An instructor is associated with several (possibly 0) students via *advisor*
- A student is associated with several (possibly 0) instructors via *advisor*



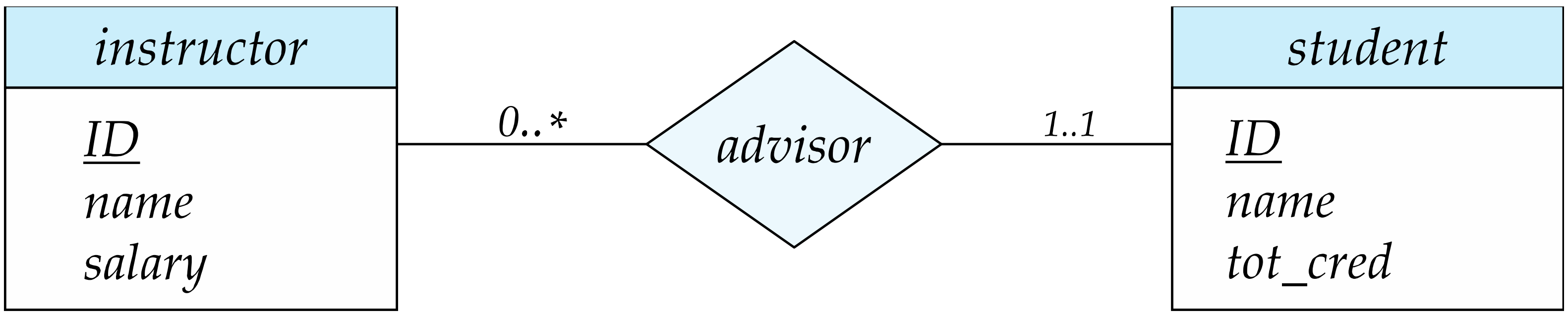
Participation of an Entity Set in a Relationship Set

- Total participation (indicated by double line): every entity in the entity set participates in at least one relationship in the relationship set
  - E.g., participation of *section* in *sec_course* is total
    - every *section* must have an associated course
- Partial participation: some entities may not participate in any relationship in the relationship set
  - Example: participation of *instructor* in *advisor* is partial



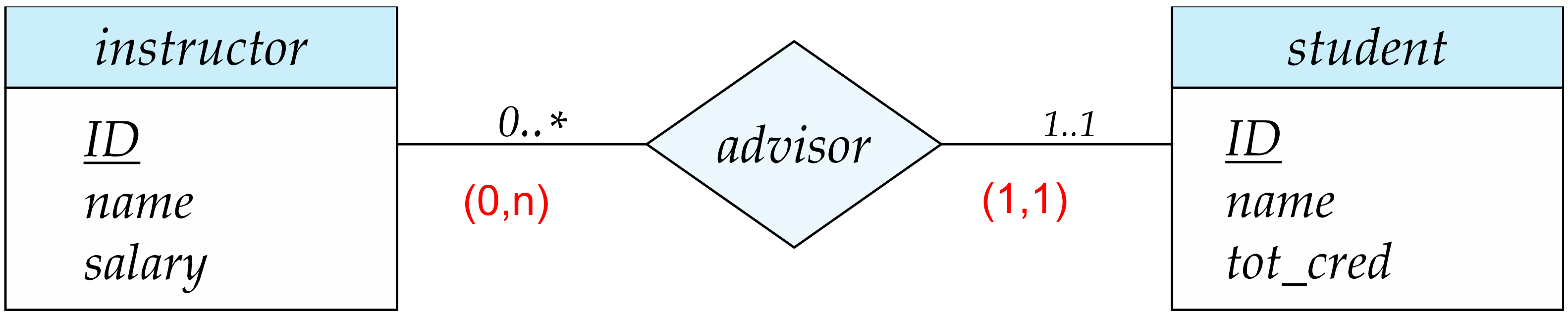
Alternative Notation for Cardinality Limits

- Cardinality limits can also express participation constraints

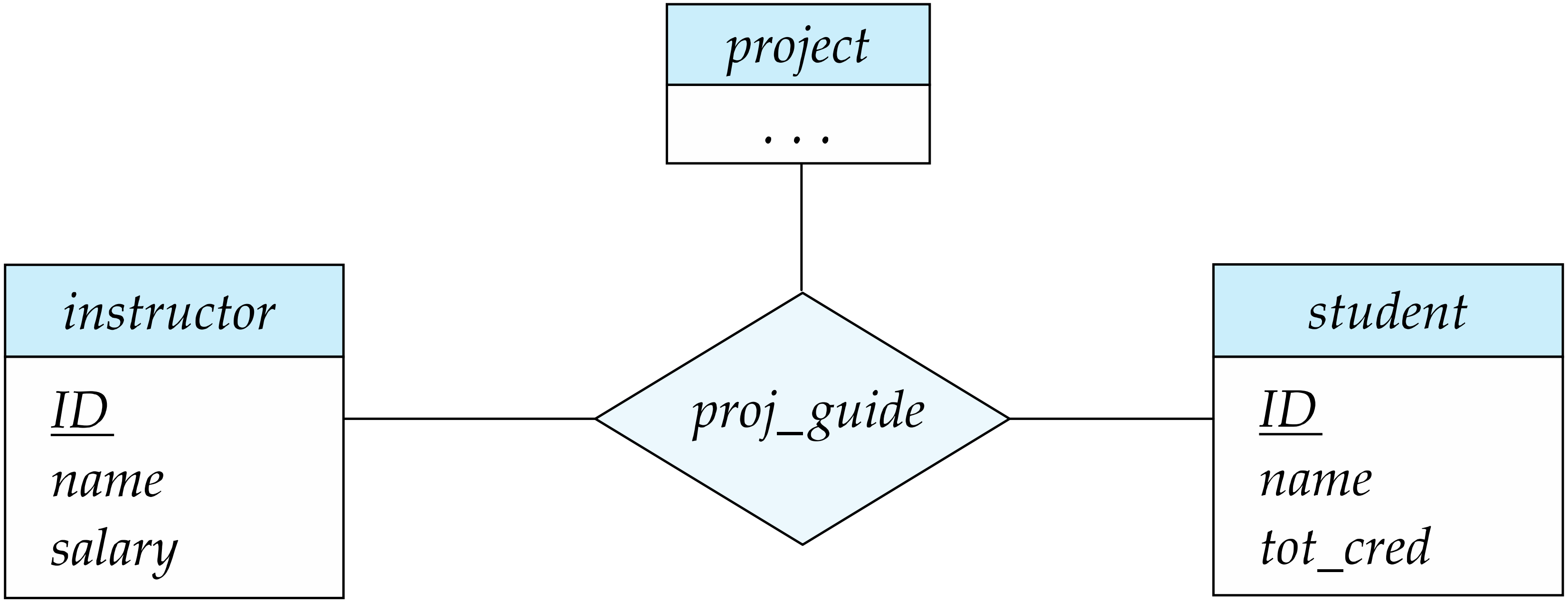


Alternative Notation for Cardinality Limits

- Alternative Notation



E-R 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
- E.g., 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.
  - E.g., 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

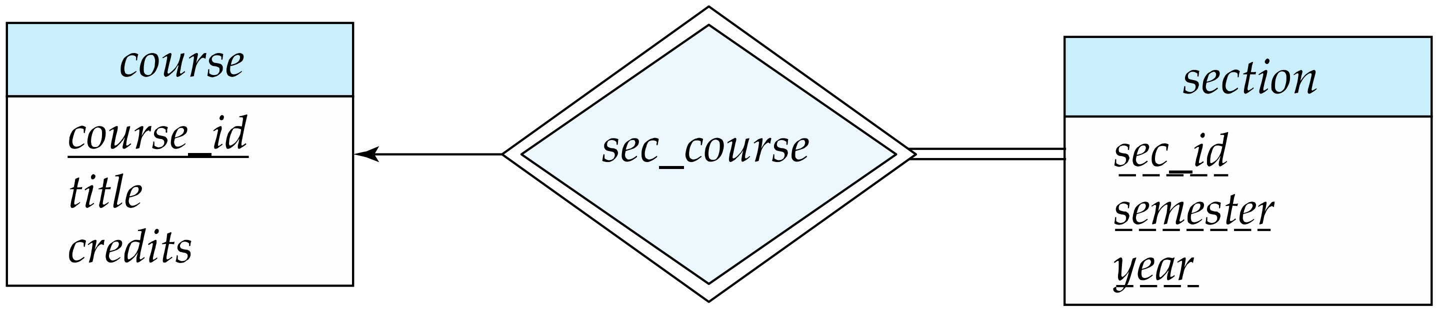
    2. each pair of entities from (*A, B*) is associated with a unique *C* entity, and each pair (*A, C*) is associated with a unique *B*
  - Each alternative has been used in different formalisms
  - To avoid confusion we outlaw more than one arrow
- Better to use cardinality constraints such as (0,n)

Weak Entity Sets

- An entity set that does not have a primary key is referred to as a **weak entity set**.
- The existence of a weak entity set depends on the existence of a **identifying entity set**
  - It must relate to the identifying entity set via a total, one-to-many relationship set from the identifying to the weak entity set
  - **Identifying relationship** depicted using a double diamond
- The **discriminator** (*or partial key)* of a weak entity set is the set of attributes that distinguishes among all the entities of a weak entity set that are associated with the same entity of the identifying entity set
- The primary key of a weak entity set is formed by the primary key of the strong entity set on which the weak entity set is existence dependent, plus the weak entity set's discriminator.

Weak Entity Sets (Cont.)

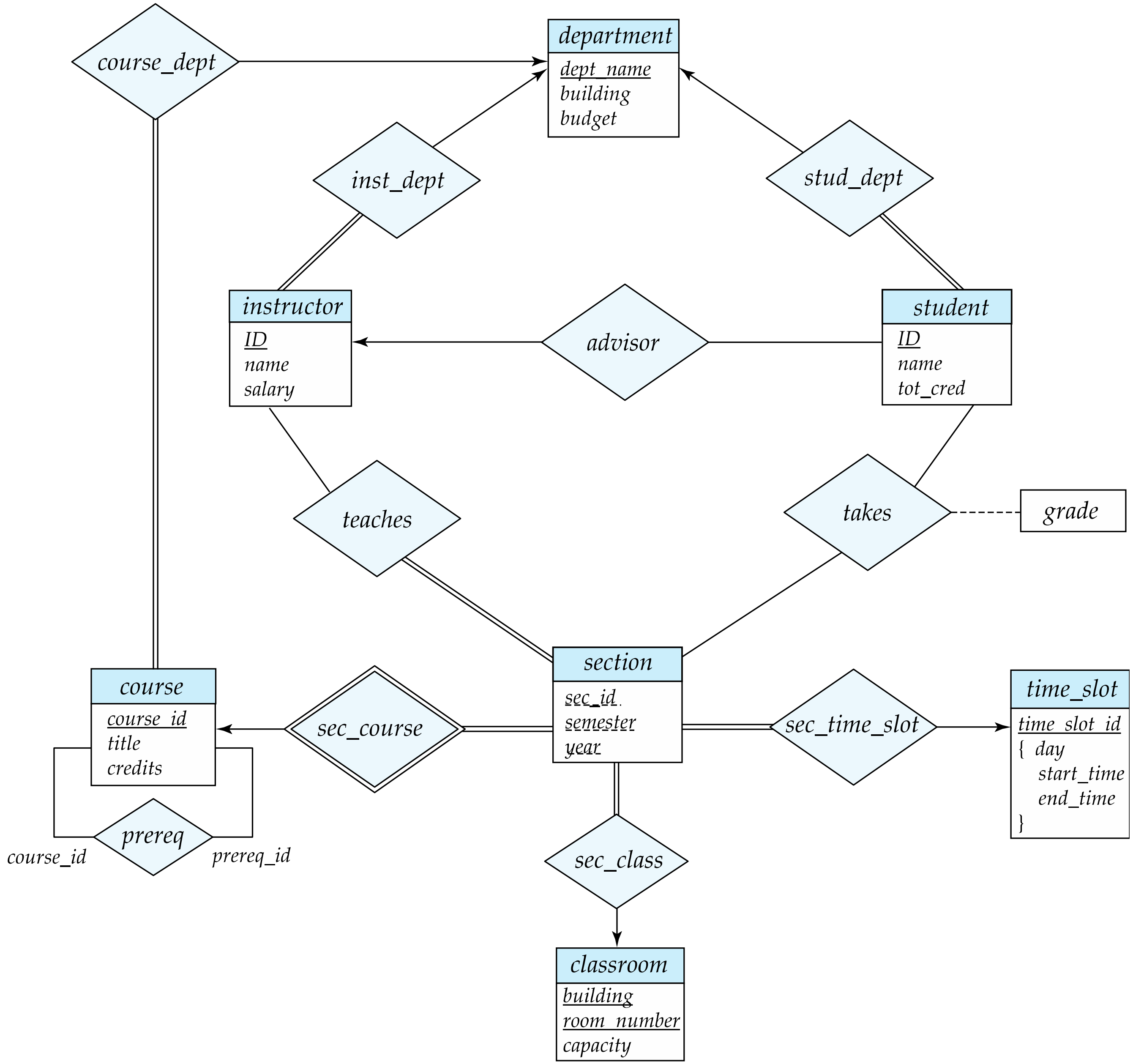
- We underline the discriminator of a weak entity set with a dashed line.
- We put the identifying relationship of a weak entity in a double diamond.
- Primary key for *section* – (*course_id, sec_id, semester, year*)



Weak Entity Sets (Cont.)

- Note: the primary key of the strong entity set is not explicitly stored with the weak entity set, since it is implicit in the identifying relationship.
- If *course_id* were explicitly stored, *section* could be made a strong entity, but then the relationship between *section* and *course* would be duplicated by an implicit relationship defined by the attribute *course_id* common to *course* and *section*

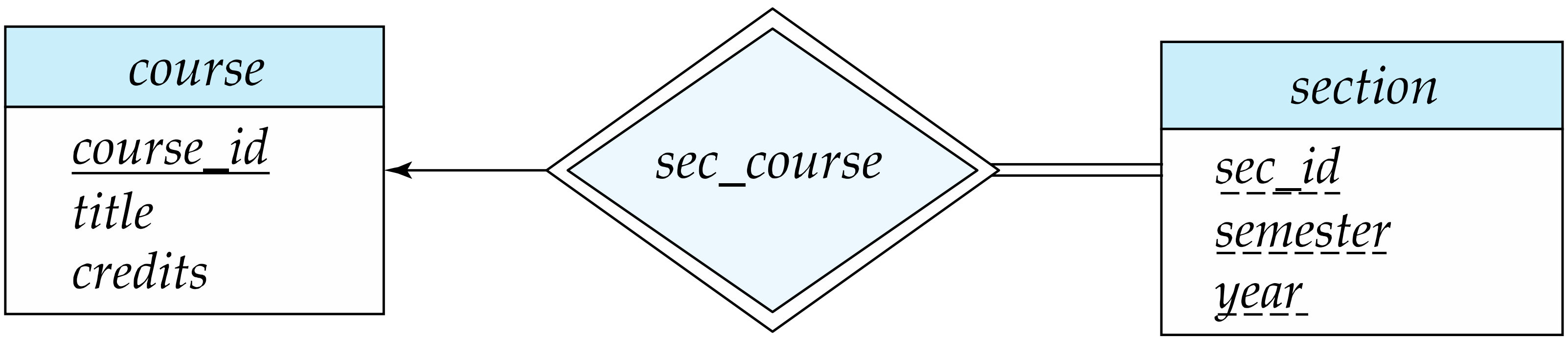
E-R Diagram for a University Enterprise



Reduction to Relational Schemas

Representing Entity Sets With Simple Attributes

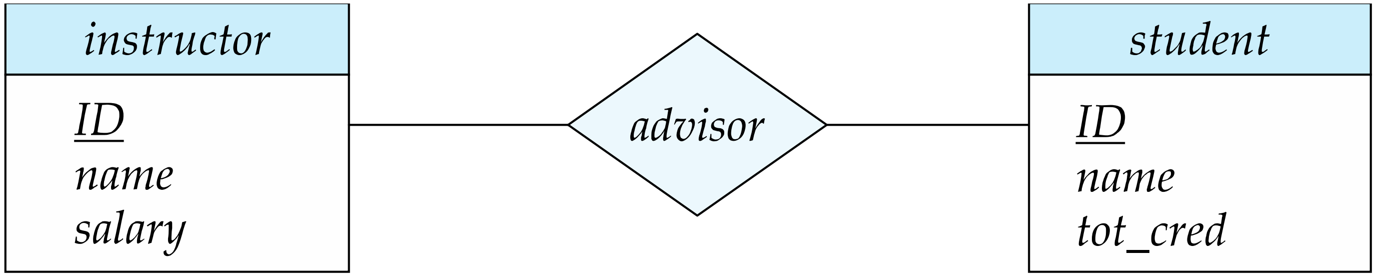
- A strong entity set reduces to a schema with the same attributes
  student(ID, name, tot_cred)
- A weak entity set becomes a table that includes a column for the primary key of the identifying strong entity set
  section (course_id, sec_id, sem, year)



Representing Relationship Sets

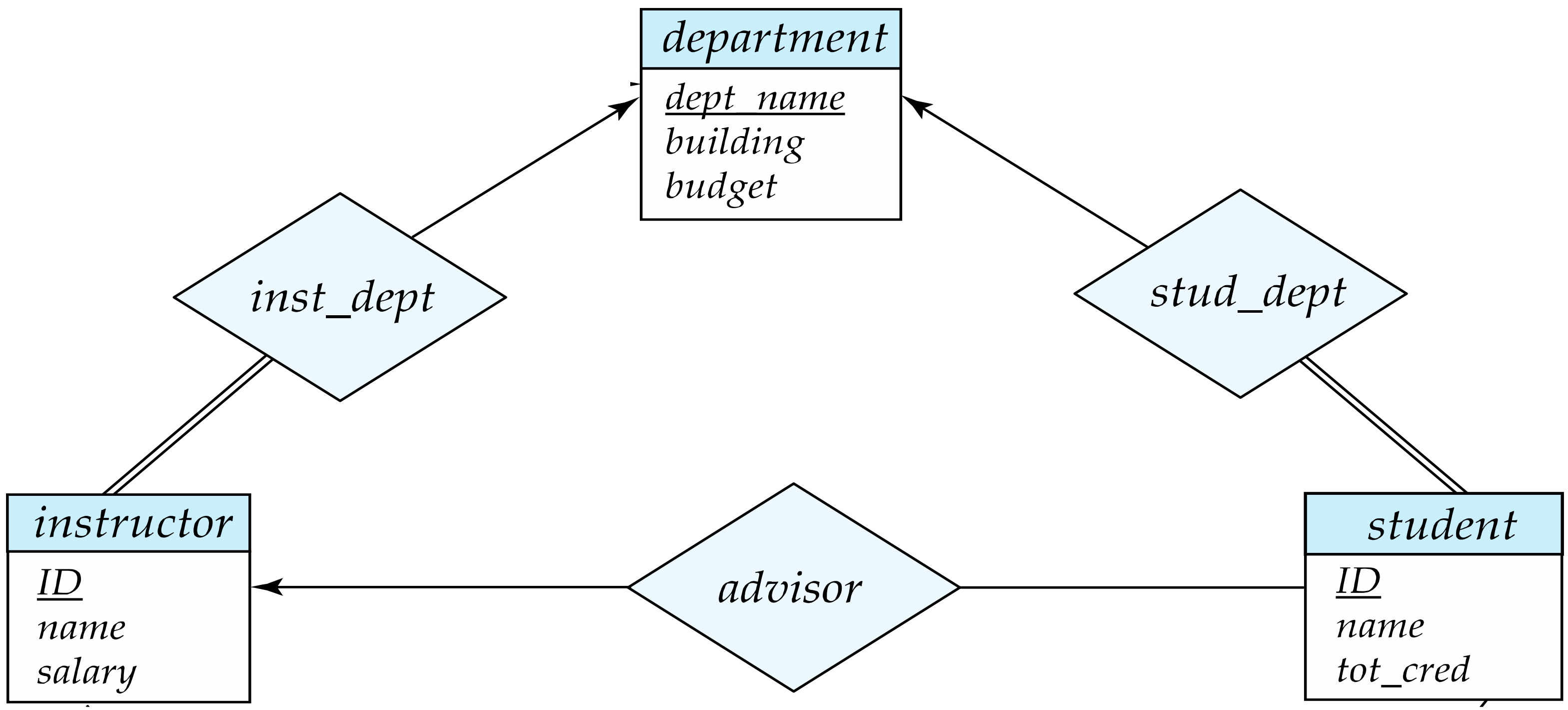
- A many-to-many relationship set is represented as a schema with attributes for the primary keys of the two participating entity sets, and any descriptive attributes of the relationship set.
- Example: schema for relationship set *advisor*

  advisor = (*s_id, i_id*)



Redundancy of Schemas

- Many-to-one and one-to-many relationship sets that are total on the many-side can be represented by adding an extra attribute to the "many" side, containing the primary key of the "one" side
- Example: Instead of creating a schema for relationship set *inst_dept*, add an attribute *dept_name* to the schema arising from entity set *instructor*



Redundancy of Schemas (Cont.)

- For one-to-one relationship sets, either side can be chosen to act as the "many" side
  - That is, extra attribute can be added to either of the tables corresponding to the two entity sets
  - If the relationship is total in both sides, the relation schemas from the two sides can be merged into one schema
- If participation is *partial* on the "many" side, replacing a schema by an extra attribute in the schema corresponding to the "many" side could result in null values
- 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

Composite and Multivalued Attributes

instructor
<u>*ID*</u> *name*   *first_name*   *middle_initial*   *last_name* *address*   *street*   *street_number*   *street_name*   *apt_number*   *city*   *state*   *zip* *{ phone_number }* *date_of_birth* *age ( )*

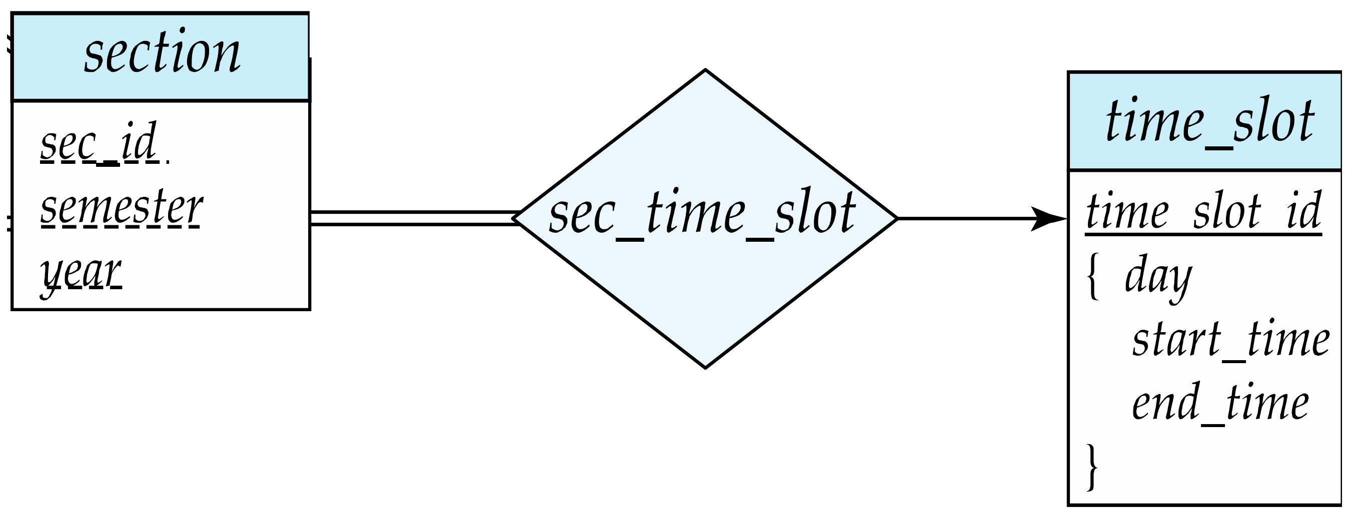
- Composite attributes are flattened out by creating a separate attribute for each component attribute
  - Example: given entity set *instructor* with composite attribute *name* with component attributes *first_name* and *last_name* the schema corresponding to the entity set has two attributes *name_first_name* and *name_last_name*
    - *Prefix omitted if there is no ambiguity*
- 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)

Composite and Multivalued Attributes

- 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*
  - Example: Multivalued attribute *phone_number* of *instructor* is represented by a schema:
    inst_phone= (*<u>ID</u>*, *<u>phone_number</u>*)
  - 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)

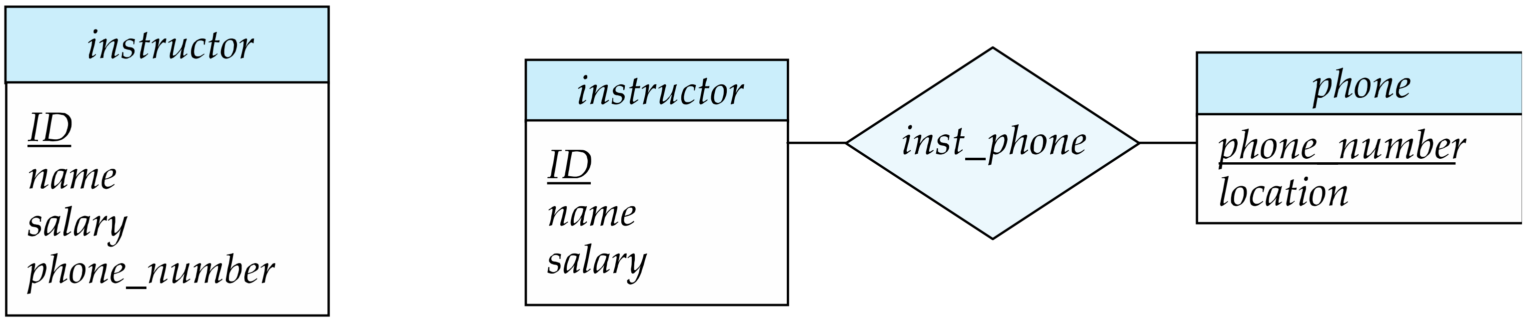
Multivalued Attributes (Cont.)

- Special case:entity *time_slot* has only one attribute other than the primary-key attribute, and that attribute is multivalued
  - Optimization: Don't create the relation corresponding to the entity, just create the one corresponding to the multivalued attribute
  - *time_slot*(*time_slot_id, day, start_time*, *end_time*)
  - Caveat: *time_slot* attribute of *section* (from *sec_time_slot*) cannot be a foreign key due to this optimization



Design Issues

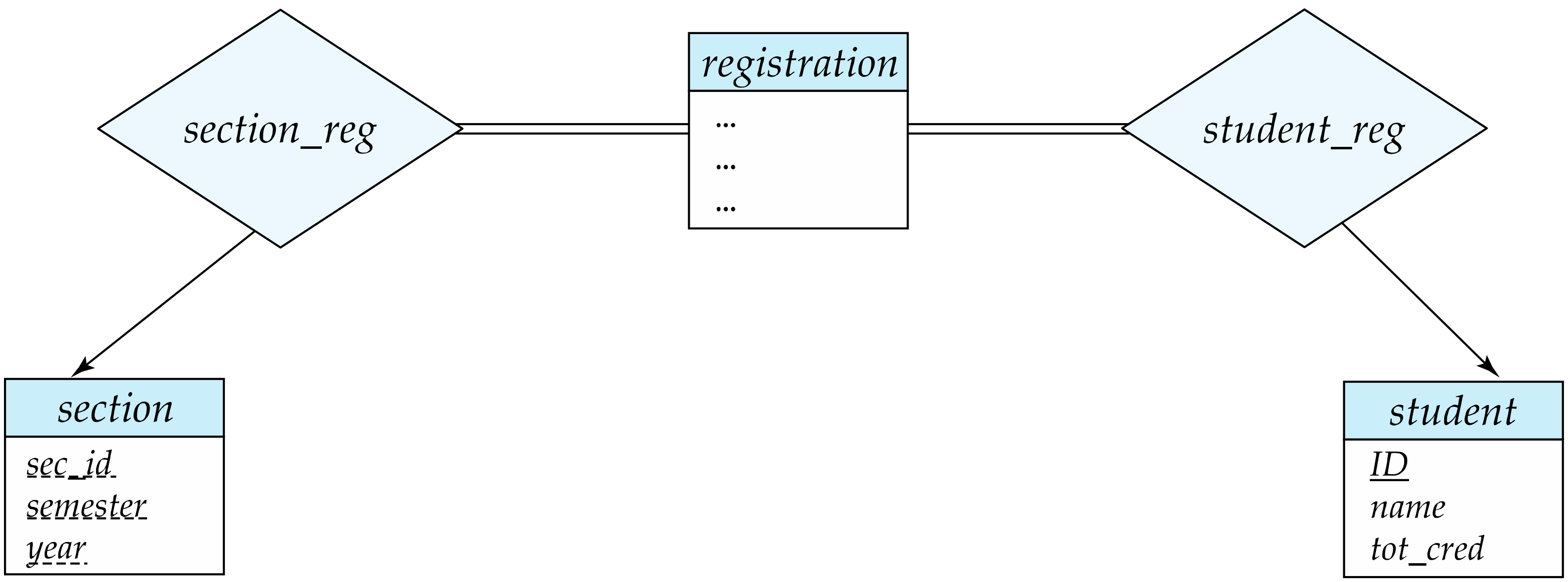
- **Use of entity sets vs. attributes**



- Designing phone as an entity allow for primary key constraints for phone
- Designing phone as an entity allow phone numbers to be used in relationships with other entities (e.g., student)
- Use of phone as an entity allows extra information about phone numbers

Design Issues

- **Use of entity sets vs. relationship sets**
  - Possible guideline is to designate a relationship set to describe an action that occurs between entities
  - Possible hint: the relationship only relates entities, but does not have an existence by itself. E.g., hasAddress: (department-address)



Design Issues

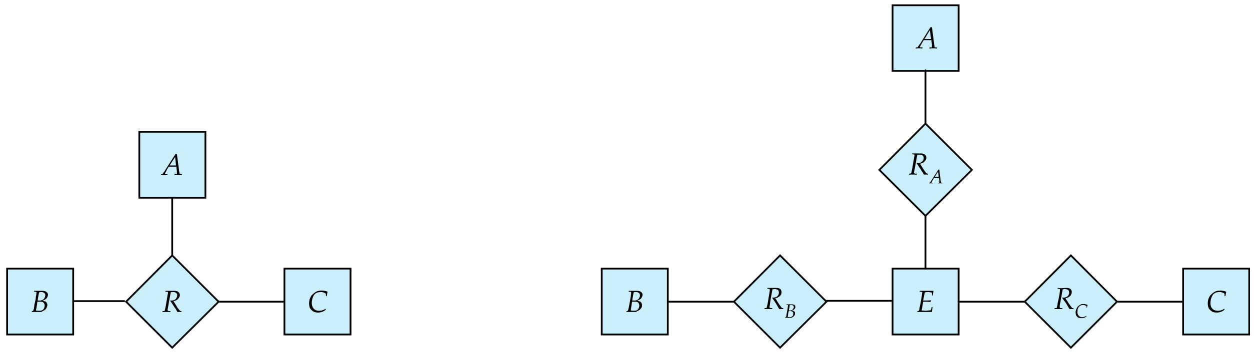
- **Binary versus n-ary relationship sets**
  - Although it is possible to replace any nonbinary (*n*-ary, for *n* > 2) relationship set by a number of distinct binary relationship sets + an aritifical entity set, a *n*-ary relationship set shows more clearly that several entities participate in a single relationship.

Binary Vs. Non-Binary Relationships

- Some relationships that appear to be non-binary may be better represented using binary relationships
  - E.g., A ternary relationship *parents*, relating a child to his/her father and mother, is best replaced by two binary relationships, *father* and *mother*
  - 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 a special identifying attribute for *E*
  - 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



(a) (b)

ER-model to Relational Summary

- ***Rule 1)*** *Strong entity E*
  - *Create relation with attributes of E*
  - *Primary key is equal to the PK of E*
- ***Rule 2)*** *Weak entity W identified by E through relationship R*
  - *Create relation with attributes of W and R and PK(E).*
  - *Set PK to discriminator attributes combined with PK(E). PK(E) is a foreign key to E.*
- ***Rule 3)*** *Binary relationship R between A and B: one-to-one*
  - *If no side is total add PK of A to as foreign key in B or the other way around. Add any attributes of the relationship R to A respective B.*
  - *If one side is total add PK of the other-side as foreign key. Add any attributes of the relationship R to the total side.*
  - *If both sides are total merge the two relation into a new relation E and choose either PK(A) as PK(B) as the new PK. Add any attributes of the relationship R to the new relation E.*

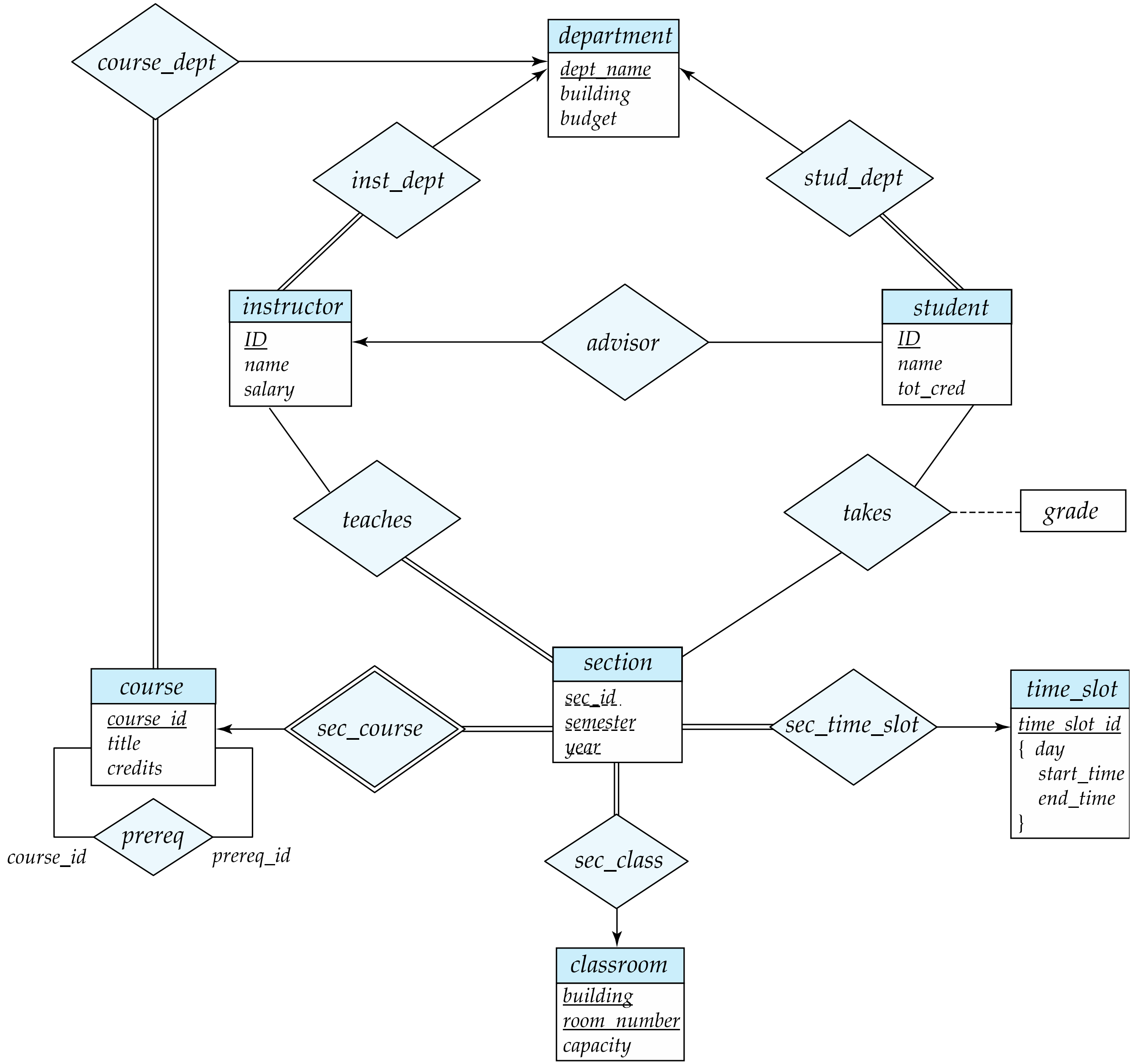
ER-model to Relational Summary (Cont.)

- ***Rule 4)*** *Binary relationship R between A and B: one-to-many/many-to-one*
  - *Add PK of the "one" side as foreign key to the "many" side.*
  - *Add any attributes of the relationship R to the "many" side.*
- ***Rule 5)*** *Binary relationship R between A and B: many-to-many*
  - *Create a new relation R.*
  - *Add PK's of A and B as attributes + plus all attributes of R.*
  - *The primary key of the relationship is PK(A) + PK(B). The PK attributes of A/B form a foreign key to A/B*
- ***Rule 6)*** *N-ary relationship R between E_1 … E_n*
  - *Create a new relation.*
  - *Add all the PK's of E_1 … E_n. Add all attributes of R to the new relation.*
  - *The primary key or R is PK(E_1) … PK(E_n). Each PK(E_i) is a foreign key to the corresponding relation.*

ER-model to Relational Summary (Cont.)

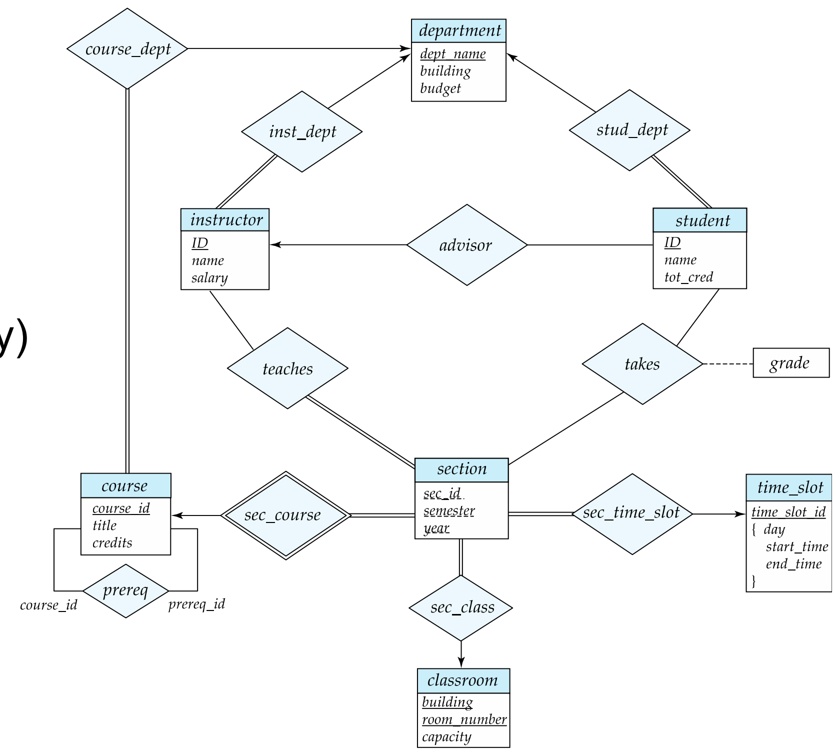
- ***Rule 7)*** *Entity E with multi-valued attribute A*
  - *Create new relation. Add A and PK(E) as attributes.*
  - *PK is all attributes. PK(E) is a foreign key.*

E-R Diagram for a University Enterprise

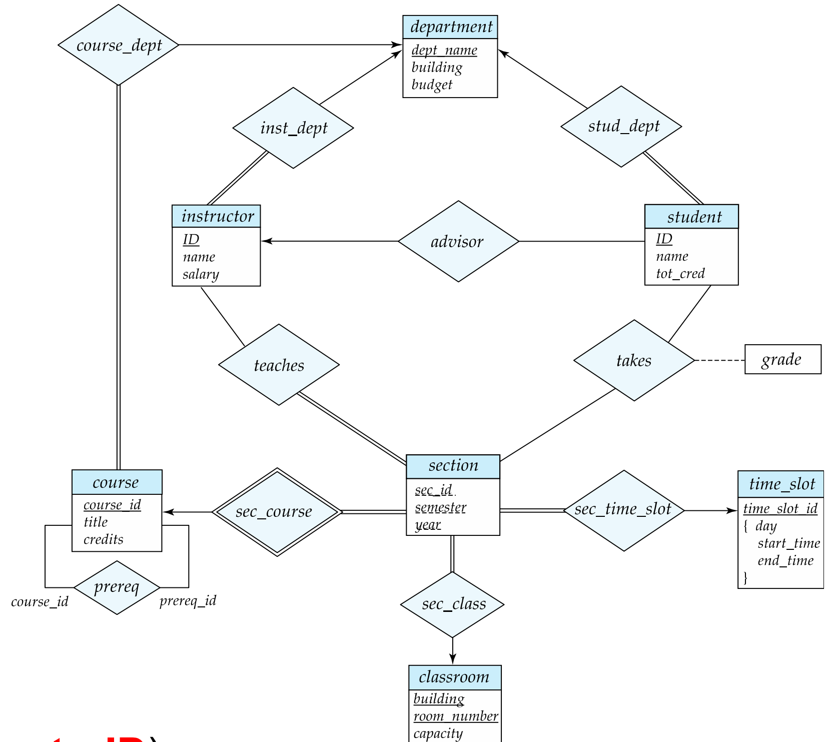


Translate the University ER-Model

- **Rule 1) Strong Entities**
  - **department**(dept_name, building, budget)
  - **instructor**(ID, name, salary)
  - **student**(ID, name, tot_cred)
  - **course**(course_id, title, credits)
  - **time_slot**(time_slot_id)
  - **classroom**(building,room_number, capacity)
- **Rule 2) Weak Entities**
  - **section**(course_id, sec_id, semester, year)



Translate the University ER-Model

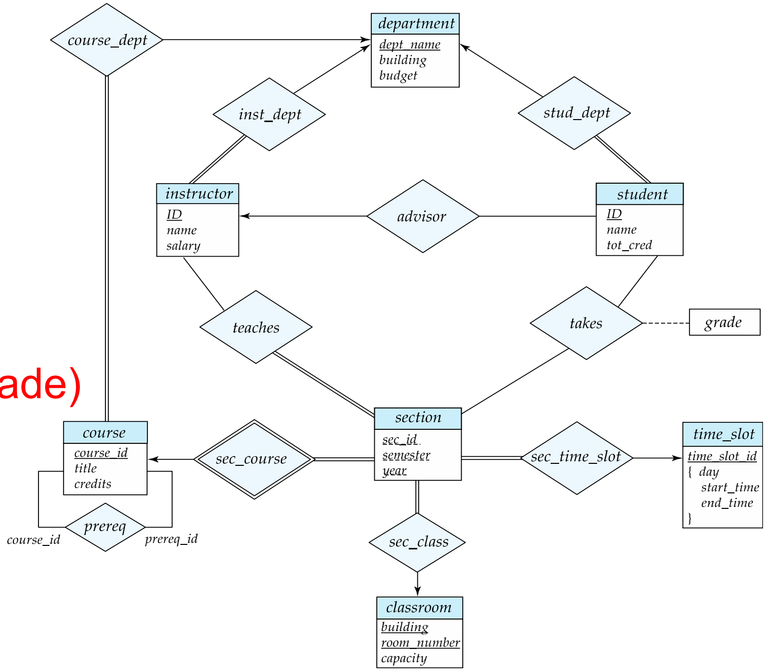


- **Rule 3) Relationships one-to-one**
  - None exist

- **Rule 4) Relationships one-to-many**
  - **department**(dept_name, building, budget)
  - **instructor**(ID, name, salary, **dept_name**)
  - **student**(ID, name, tot_cred, **dept_name**, **instr_ID**)
  - **course**(course_id, title, credits, **dept_name**)
  - **time_slot**(time_slot_id)
  - **classroom**(building,room_number, capacity)
  - **section**(course_id, sec_id, semester, year, **room_building**, **room_number**, **time_slot_id**)

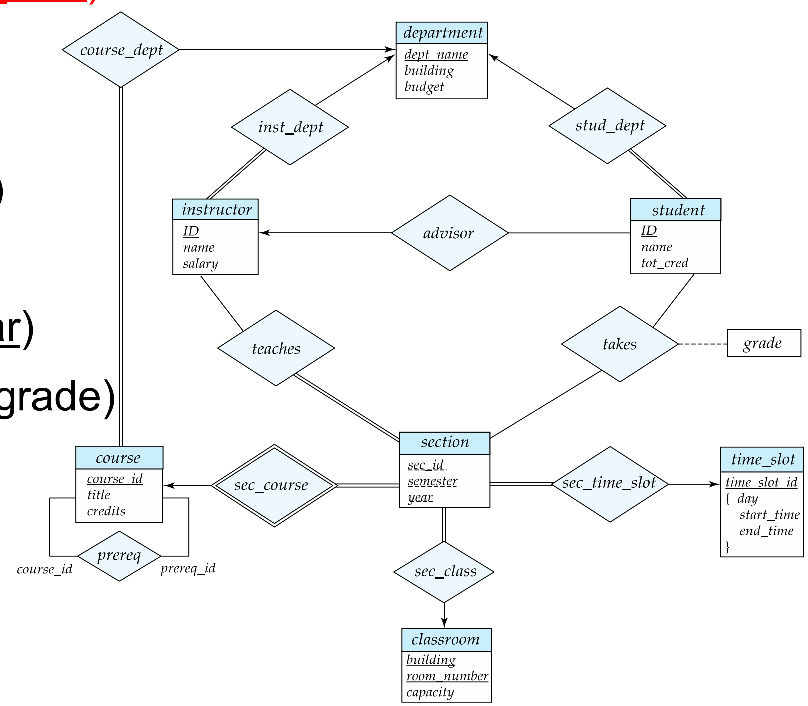
Translate the University ER-Model

- **Rule 5) Relationships many-to-many**
  - **department**(<u>dept_name</u>, building, budget)
  - **instructor**(<u>ID</u>, name, salary, dept_name)
  - **student**(<u>ID</u>, name, tot_cred, dept_name, instr_ID)
  - **course**(<u>course_id</u>, title, credits, dept_name)
  - **time_slot**(<u>time_slot_id</u>)
  - **classroom**(<u>building</u>,<u>room_number</u>, capacity)
  - **section**(<u>course_id</u>, <u>sec_id</u>, <u>semester</u>, <u>year</u>, room_building, room_number, time_slot_id)
  - **prereq**(<u>course_id</u>, <u>prereq_id</u>)
  - **teaches**(<u>ID</u>, <u>course_id</u>, <u>sec_id</u>, <u>semester</u>, <u>year</u>)
  - **takes**(<u>ID</u>, <u>course_id</u>, <u>sec_id</u>, <u>semester</u>, <u>year</u>, grade)
- **Rule 6) N-ary Relationships**
  - none exist



Translate the University ER-Model

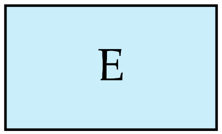
- **Rule 7) Multivalued attributes**
  - **department**(dept_name, building, budget)
  - **instructor**(ID, name, salary, dept_name)
  - **student**(ID, name, tot_cred, dept_name, instr_ID)
  - **course**(course_id, title, credits, dept_name)
  - **time_slot**(time_slot_id)
  - **time_slot_day**(time_slot_id, start_time, end_time)
  - **classroom**(building,room_number, capacity)
  - **section**(course_id, sec_id, semester, year, room_building, room_number, time_slot_id)
  - **prereq**(course_id, prereq_id)
  - **teaches**(ID, course_id, sec_id, semester, year)
  - **takes**(ID, course_id, sec_id, semester, year, grade)



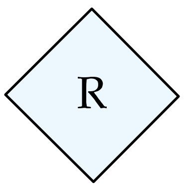
ER Design Decisions

- 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.

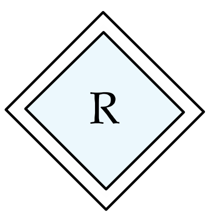
Summary of Symbols Used in E-R Notation



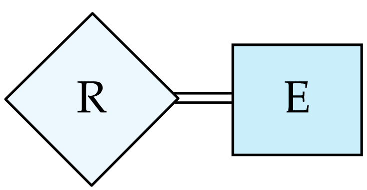
entity set



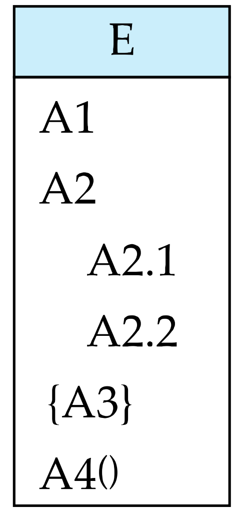
relationship set



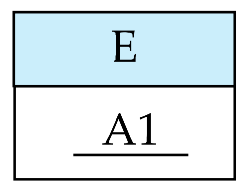
identifying relationship set for weak entity set



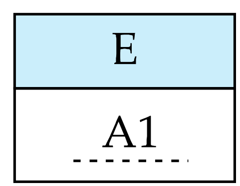
total participation of entity set in relationship



attributes: simple (A1), composite (A2) and multivalued (A3) derived (A4)



primary key



discriminating attribute of weak entity set

End of Chapter 7

Partially taken from

Klaus R. Dittrich

modified from:

Database System Concepts, 6th Ed.