CSE 215: Database

Department of CSE
Bangladesh University of Engineering & Technology

What is it?

- The ER data model employs three basic concepts:
  - entity sets
  - relationship sets
  - attributes
- The ER model also has an associated diagrammatic representation, which can express the overall logical structure of a database graphically
  - ER diagram

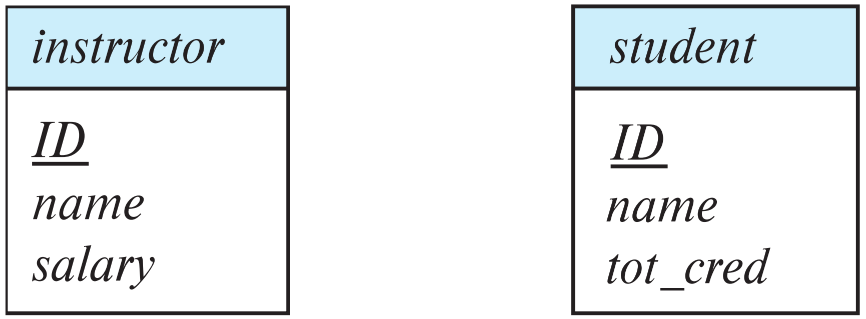
Entity Sets

- An **entity** is an object that exists and is distinguishable from other objects
  - Example: specific person, company, plant
- An **entity set** is a set of entities of the same type that share the same properties
  - Example: set of all persons, companies, trees,
- An entity is represented by a set of attributes; i.e., descriptive properties possessed by all members of an entity set
  - Example:

    instructor = (ID, name, salary)
    course= (course_id, title, credits)
- A subset of the attributes form a **primary key** of the entity set; i.e., uniquely identifying each member of the set

Entity Sets – ER Diagram

- Entity sets can be represented graphically as follows:
  - Rectangles represent entity sets.
  - Attributes listed inside entity rectangle
  - Underline indicates primary key attributes



instructor

ID
name
salary

student

ID
name
tot_cred

Relationship Sets

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

  Example:

44553 (Peltier)	*advisor*	22222 (Einstein)
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:

  $$(44553,22222) \in \textit{advisor}$$

Relationship Sets

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

  Example:

44553 (Peltier)	*advisor*	22222 (Einstein)
student entity	relationship set	*instructor* entity

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

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

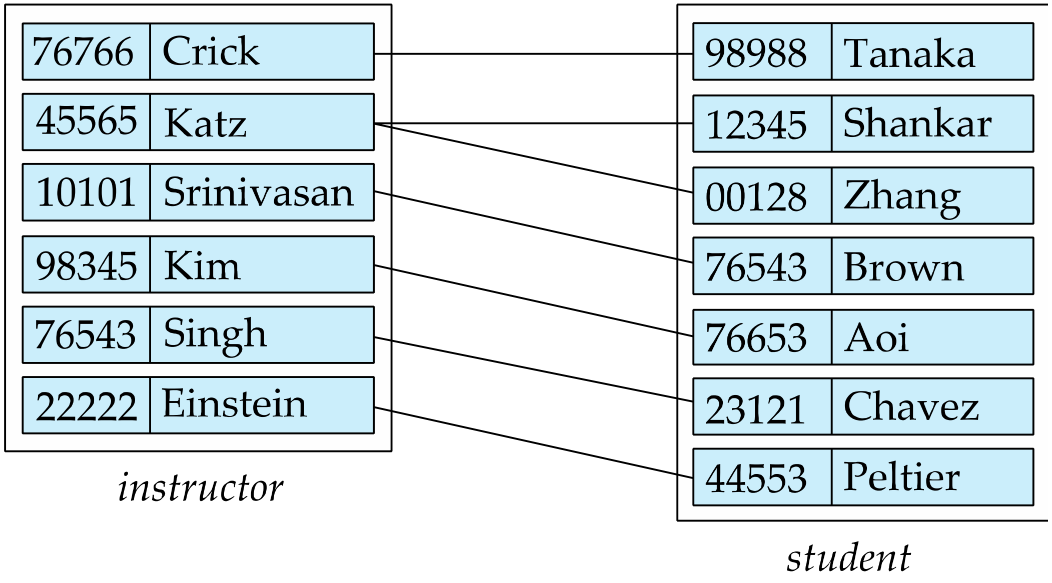
  where $(e_1, e_2, \ldots, e_n)$ is a relationship

  - Example:

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

Relationship Sets

- Example: we define the relationship set advisor to denote the associations between students and the instructors who act as their advisors
- Pictorially, we draw a line between related entities



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

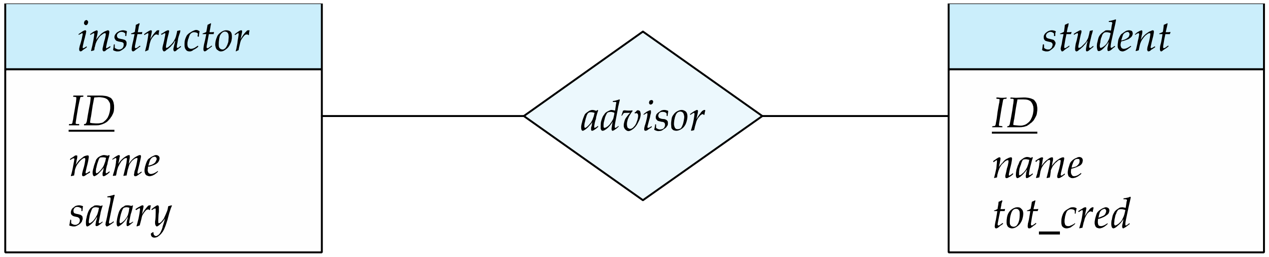
instructor

98988	Tanaka
12345	Shankar
00128	Zhang
76543	Brown
76653	Aoi
23121	Chavez
44553	Peltier

student

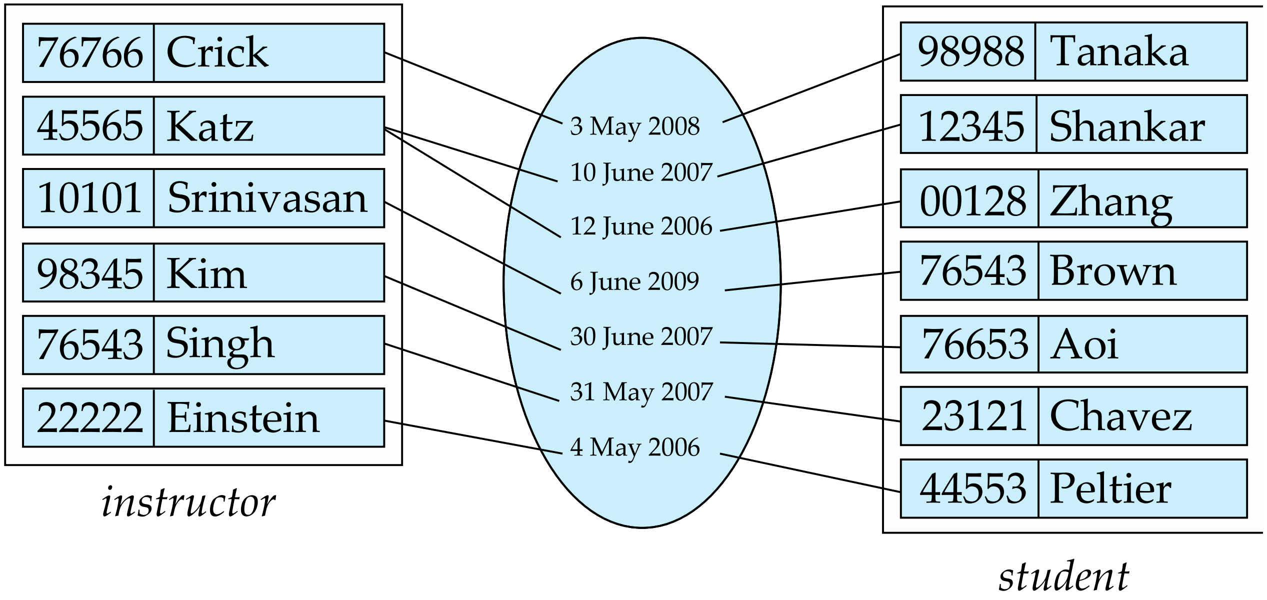
Relationship Sets-ER Diagram

- Diamonds represent relationship sets



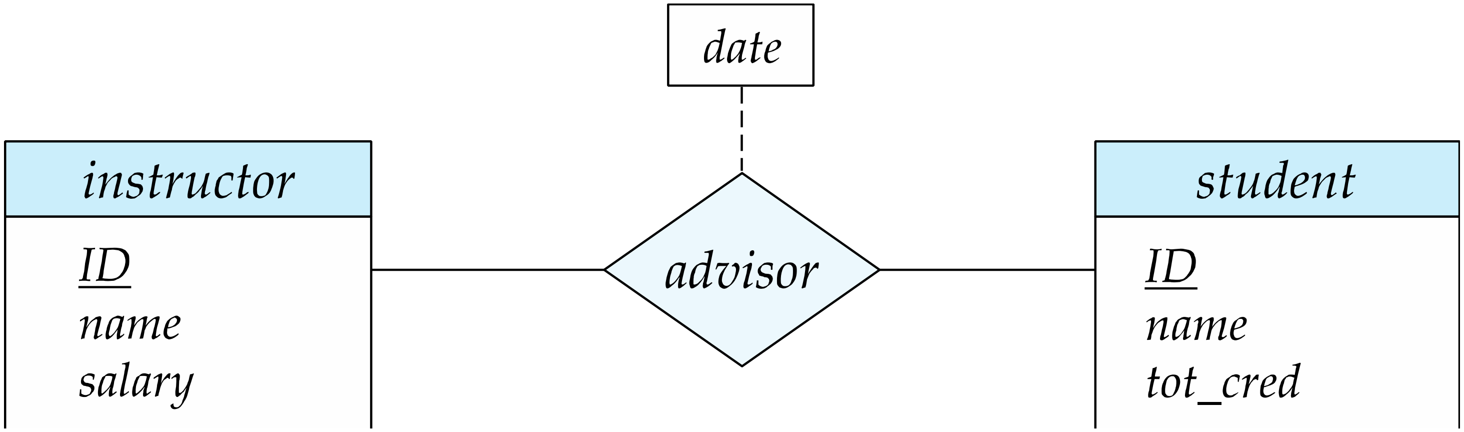
Relationship Sets

- An attribute can also be associated with 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



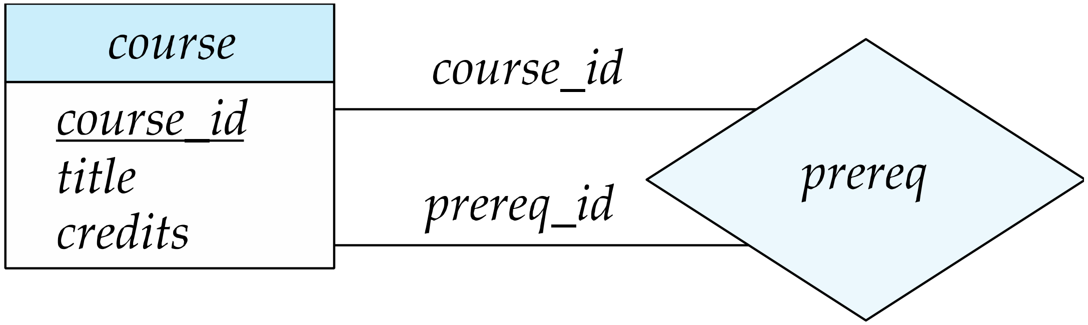
Relationship Sets-ER Diagram

- Diamonds represent relationship sets



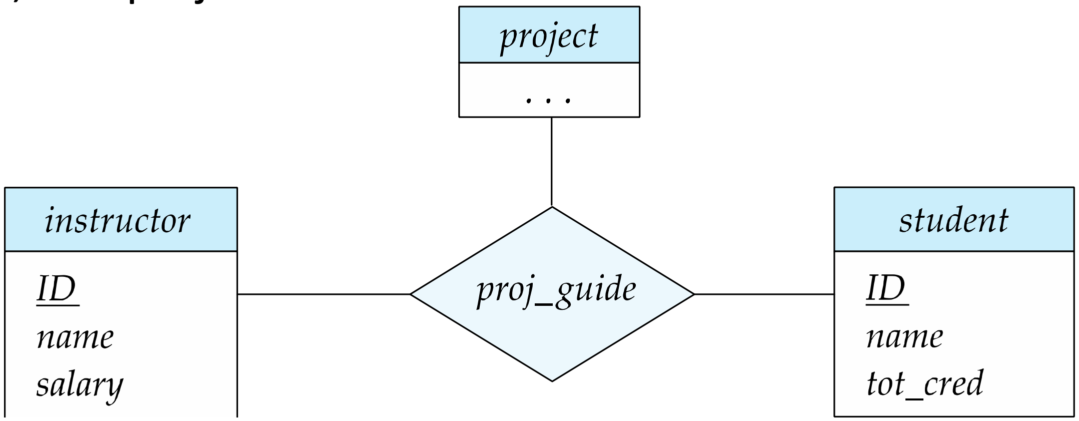
Roles

- 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.



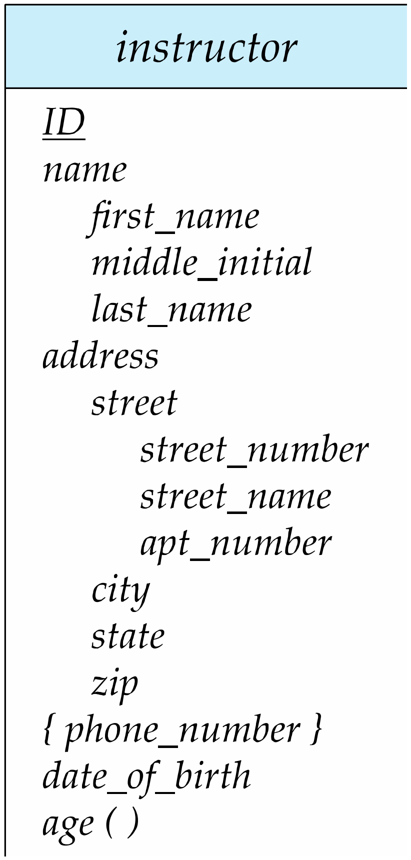
Relationship Degree

- Binary relationship
  - Involve two entity sets (or degree two)
- Relationships between more than two entity sets are rare
  - 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

- Attribute types:
  - **Simple** and **composite** attributes
    - Composite attributes allow us to divided attributes into subparts
  - **Single-valued** and **multivalued** attributes
    - Example: multivalued attribute: *phone_numbers*
  - **Derived** attributes
    - Can be computed from other attributes
    - Example: age, given date_of_birth
- **Domain** – the set of permitted values for each attribute



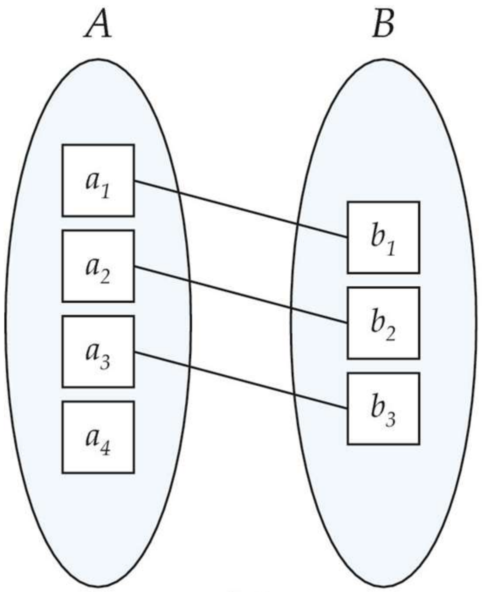
instructor

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

Mapping Cardinalities

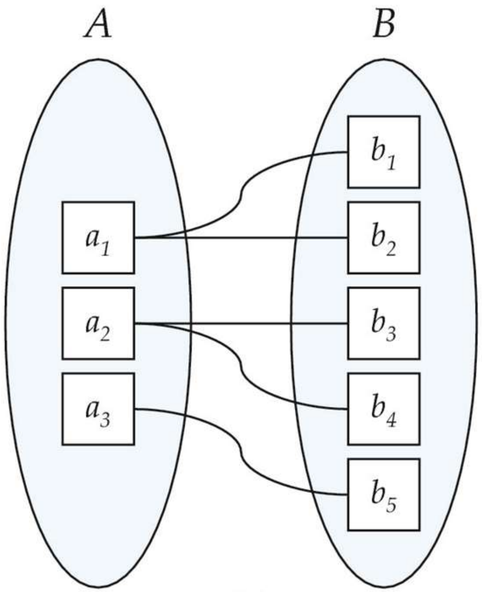
- Express the number of entities to which another entity can be associated via a relationship set
- **One to one**
  - An entity in A is associated with at most one entity in B, An entity in B is associated with at most one entity in A
- **One to many**
  - An entity in A is associated with any number (0 or more) of entities in B, An entity in B is associated with at most one entity in A
- **Many to one**
  - An entity in A is associated with at most one entity in B, An entity in B is associated with any number (0 or more) of entities in A
- **Many to many**
  - An entity in A is associated with any number (0 or more) of entities in B, An entity in B is associated with any number (0 or more) of entities in A

Mapping Cardinalities



(a)

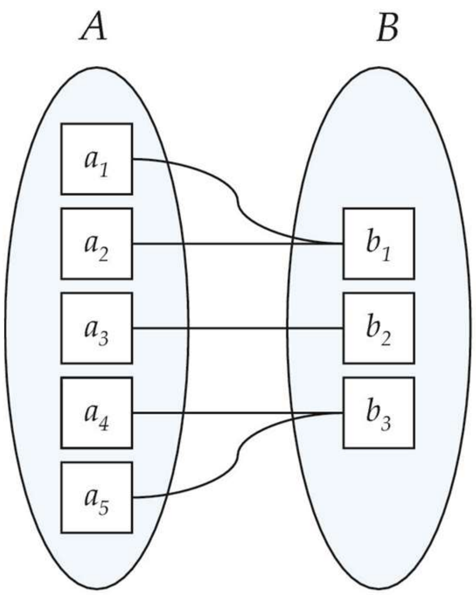
One to one



(b)

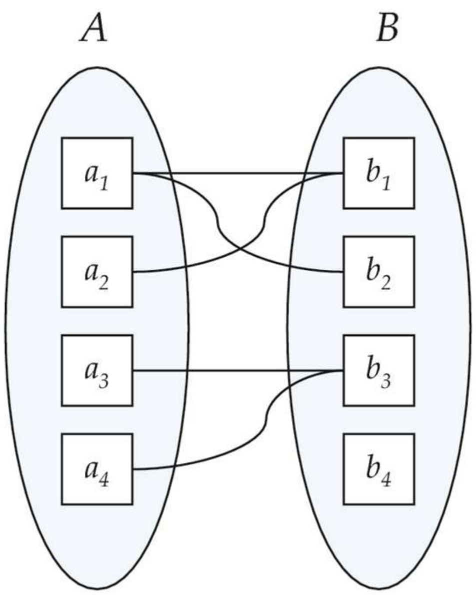
One to many

Mapping Cardinalities



(a)

Many to one

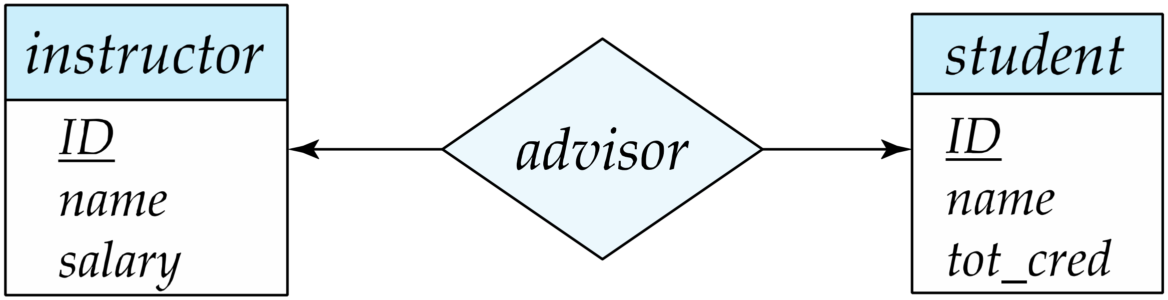


(b)

Many to many

Cardinality Constraints-ER Diagram

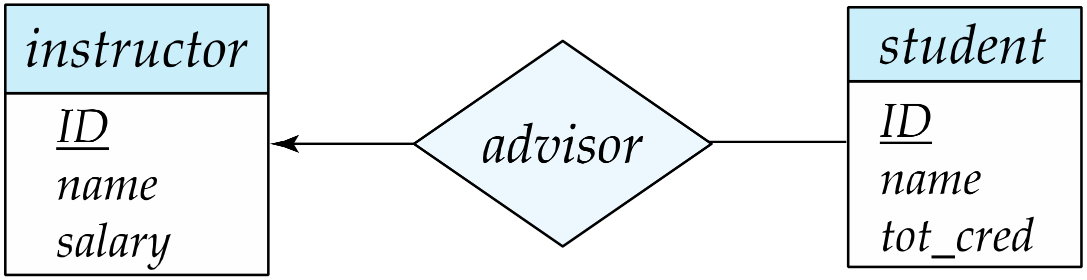
- We express cardinality constraints by drawing either a directed line (⍰), signifying “one,” or an undirected line (—), signifying “many,” between the relationship set and the entity set



One to one

Cardinality Constraints-ER Diagram

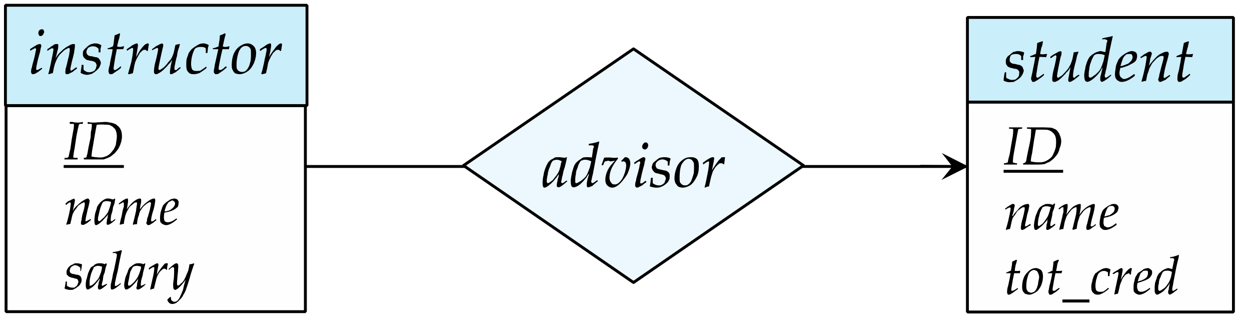
- We express cardinality constraints by drawing either a directed line (⍰), signifying “one,” or an undirected line (—), signifying “many,” between the relationship set and the entity set



One to many

Cardinality Constraints-ER Diagram

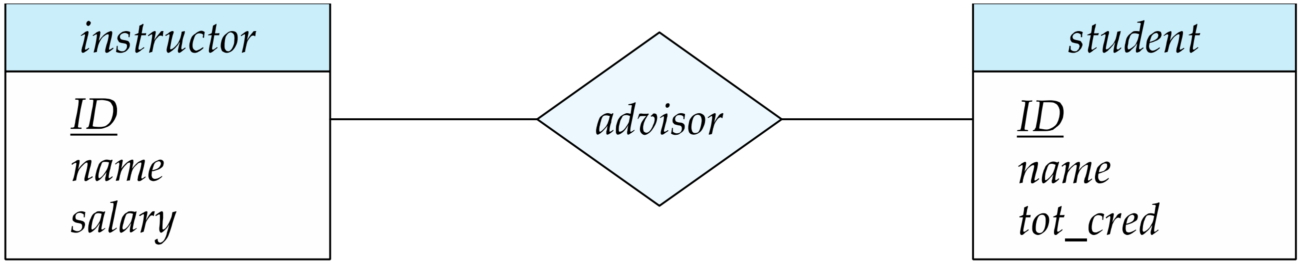
- We express cardinality constraints by drawing either a directed line (⍰), signifying “one,” or an undirected line (—), signifying “many,” between the relationship set and the entity set



Many to one

Cardinality Constraints-ER Diagram

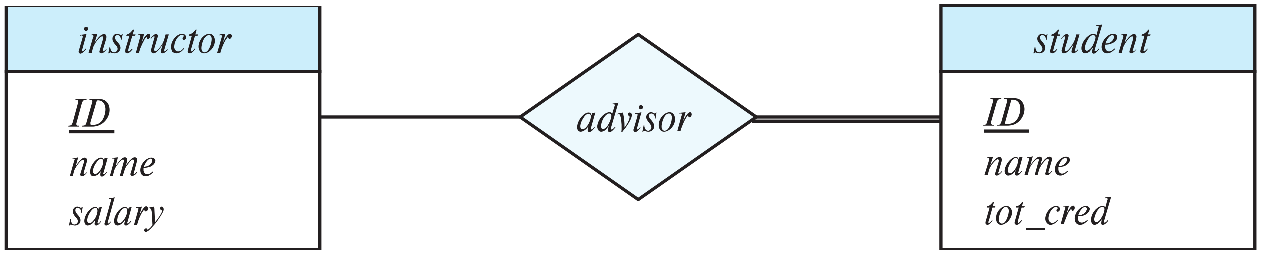
- We express cardinality constraints by drawing either a directed line (), signifying “one,” or an undirected line (—), signifying “many,” between the relationship set and the entity set



Many to many

Cardinality Constraints-ER Diagram

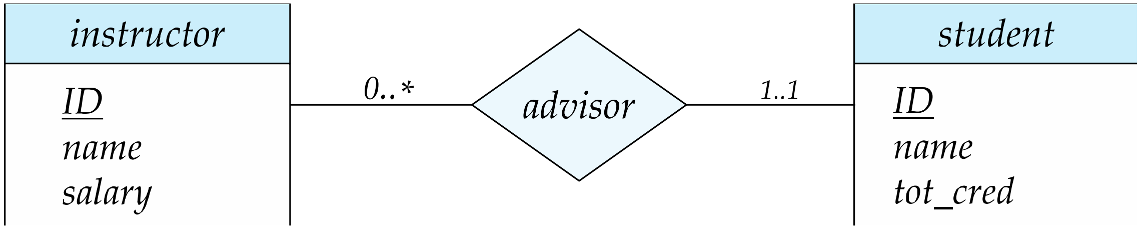
- **Total participation** (indicated by double line): every entity in the entity set participates in at least one relationship in the relationship set
  - Example: participation of student in advisor relation is total, every student must have an associated instructor



- **Partial participation**: some entities may not participate in any relationship in the relationship set
  - Example: participation of instructor in advisor is partial

Cardinality Constraints-ER Diagram

- A line may have an associated minimum and maximum cardinality, shown in the form l..h, where l is the minimum and h the maximum cardinality
  - A minimum value of 1 indicates total participation
  - A maximum value of 1 indicates that the entity participates in at most one relationship
  - A maximum value of * indicates no limit



Instructor can advise 0 or more students

A student must have 1 advisor; cannot have multiple advisors

Cardinality Constraints-ER Diagram

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

Primary Key for Entity Sets

- By definition, individual entities are distinct.
- From database perspective, the differences among them must be expressed in terms of their attributes.

- The values of the attribute values of an entity must be such that they can uniquely identify the entity.
  - No two entities in an entity set are allowed to have exactly the same value for all attributes.

- A **key** for an entity is a set of attributes that suffice to distinguish entities from each other

Keys

- Let K is a subset of attributes
- K is a superkey if values for K are sufficient to identify a unique tuple of each possible relation r(R)
  - Example: {ID} and {ID,name} are both superkeys of instructor
- Superkey K is a candidate key if K is minimal
  - Example: {ID} is a candidate key for Instructor
- One of the candidate keys is selected to be the primary key

Primary Key for Relationship Sets

- To distinguish among the various relationships of a relationship set we use the individual primary keys of the entities in the relationship set

- Let R be a relationship set involving entity sets E1, E2, .. En
- The primary key for R is consists of the union of the primary keys of entity sets E1, E2, ..En
  - If the relationship set R has attributes a1, a2, .., am associated with it, then the primary key of R and attributes a1, a2, .., am describe the relationship in set R
- Example: relationship set “advisor”
  - The primary key consists of instructor.ID and student.ID

Primary Key for Relationship Sets

- The choice of the primary key for a relationship set depends on the mapping cardinality of the relationship set
- Many-to-Many relationships
  - The preceding union of the primary keys is a minimal superkey and is chosen as the primary key
- One-to-Many relationships
  - The primary key of the “Many” side is a minimal superkey and is used as the primary key
- Many-to-one relationships
  - The primary key of the “Many” side is a minimal superkey and is used as the primary key
- One-to-one relationships
  - The primary key of either one of the participating entity sets forms a minimal superkey, and either one can be chosen as the primary key

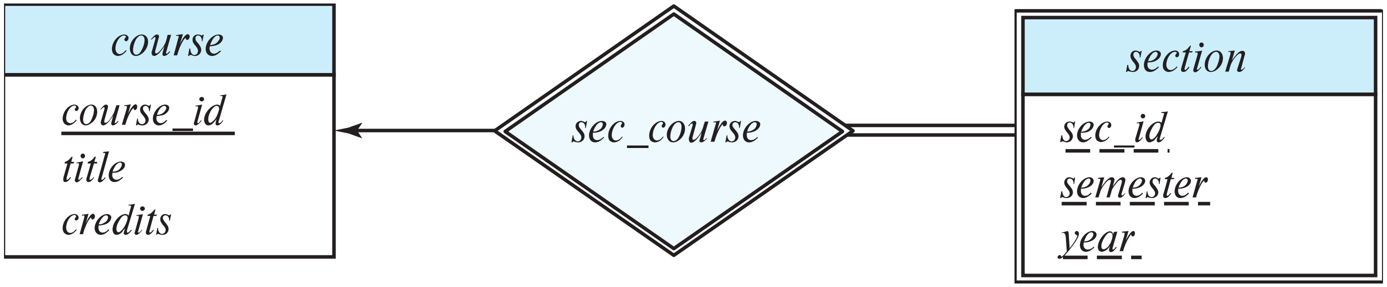
Weak/Strong Entity Sets

- A **weak entity set** is one whose existence is dependent on another entity, called its **identifying entity**
- Instead of associating a primary key with a weak entity, we use the primary key of identifying entity, along with extra attributes called **discriminator** to uniquely identify a weak entity
- The identifying entity set is said to **own** the weak entity set that it identifies
- The relationship associating the weak entity set with the identifying entity set is called the **identifying relationship**

- An entity set that is not a weak entity set is termed a **strong entity set**.

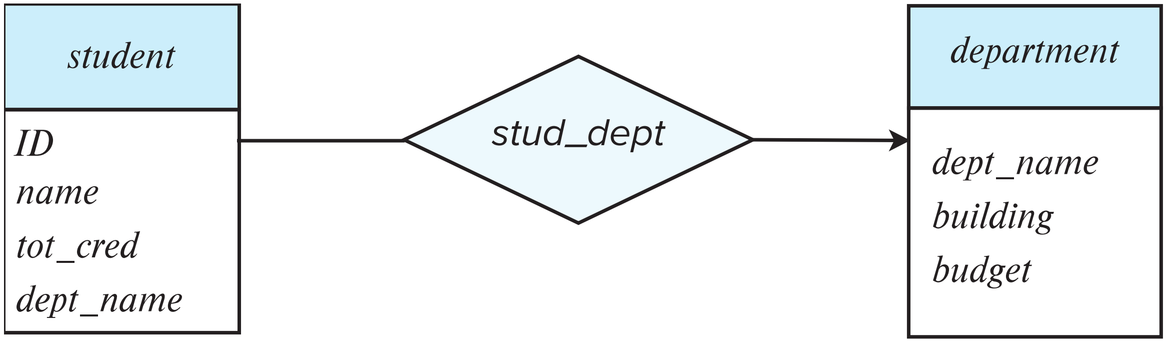
Weak Entity Sets-ER Diagram

- In E-R diagrams, a weak entity set is depicted via a double rectangle.
- We underline the discriminator of a weak entity set with a dashed line.
- The relationship set connecting the weak entity set to the identifying strong entity set is depicted by a double diamond
- Primary key for section – (course_id, sec_id, semester, year)



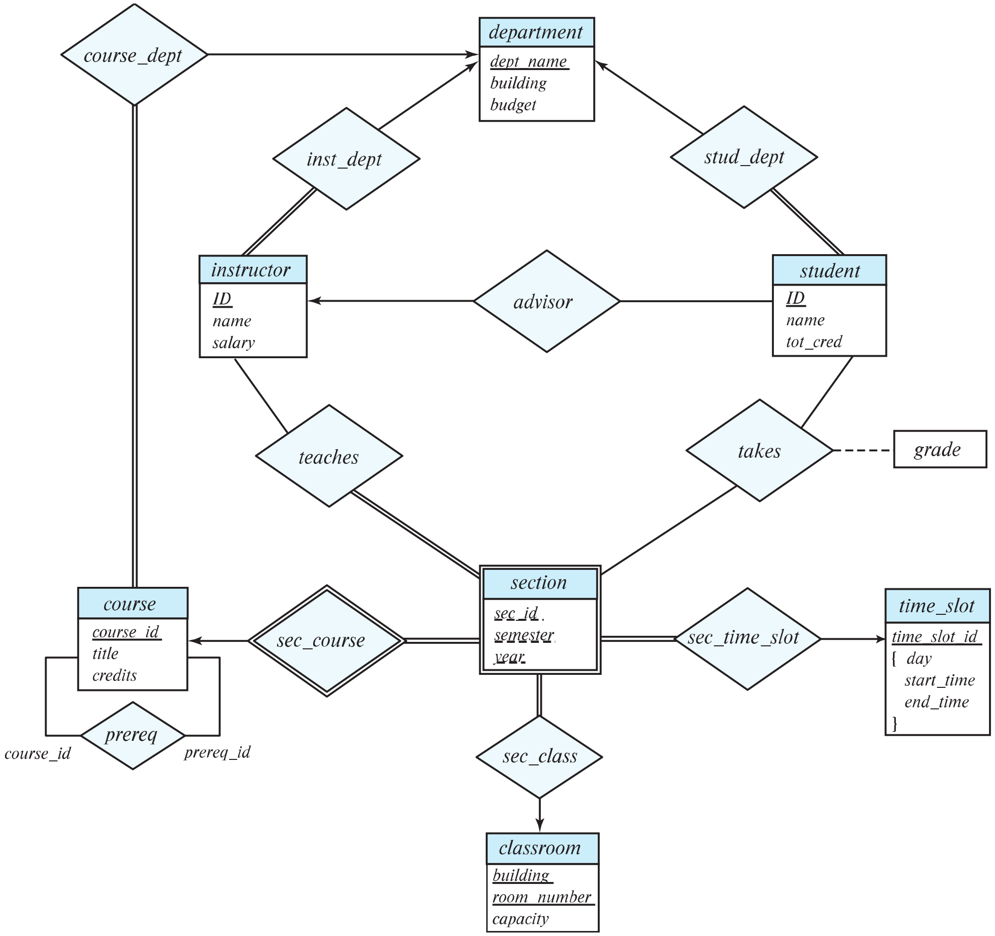
Redundant Attributes

- We model the fact that each student has an associated department using a relationship set stud_dept
- The attribute dept_name in student below replicates information present in the relationship and is therefore redundant and needs to be removed.
- BUT: when converting back to tables, in some cases the attribute gets reintroduced, as we will see later



(a) Incorrect use of attribute

ER Diagram of University



Reduction to Relation Schemas

- Entity sets and relationship sets can be expressed uniformly as *relation schemas* that represent the contents of the database
- 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(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

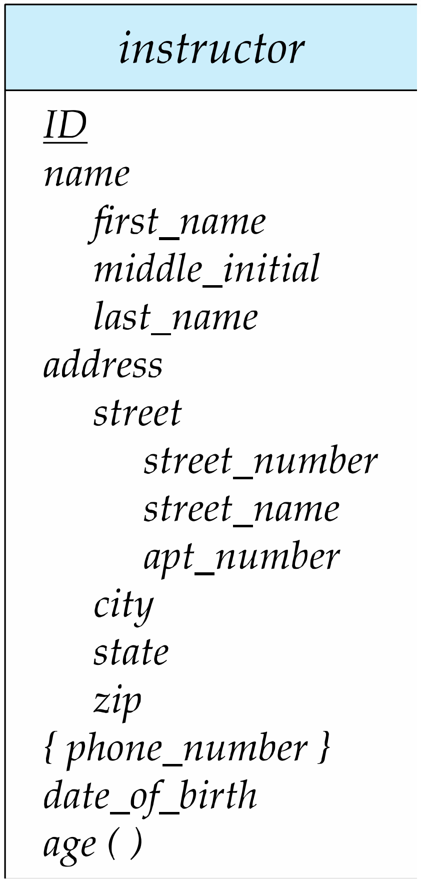
section (course_id, sec_id, sem, year)

ER Model to Relation Schema

Composite Attributes

- Composite attributes are flattened out by creating a separate attribute for each component attribute

- 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)



instructor

ID
name
  first_name
  middle_initial
  last_name
address
  street
    street_number
    street_name
    apt_number
  city
  state
  zip
{ phone_number }
date_of_birth
age ( )

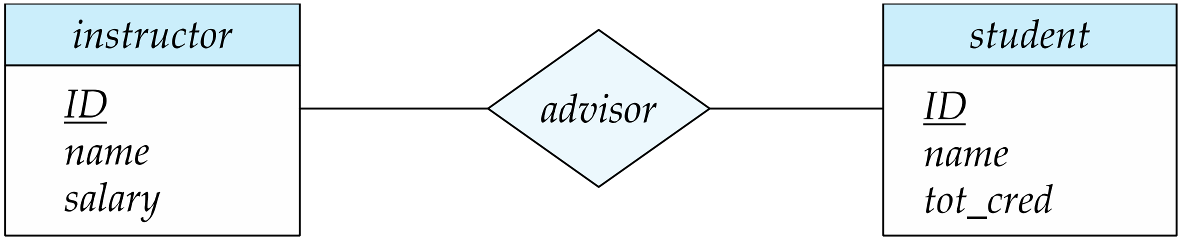
Multivalued Attributes

- 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= (ID, phone_number)
- 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)

Reduction to Relation Schemas

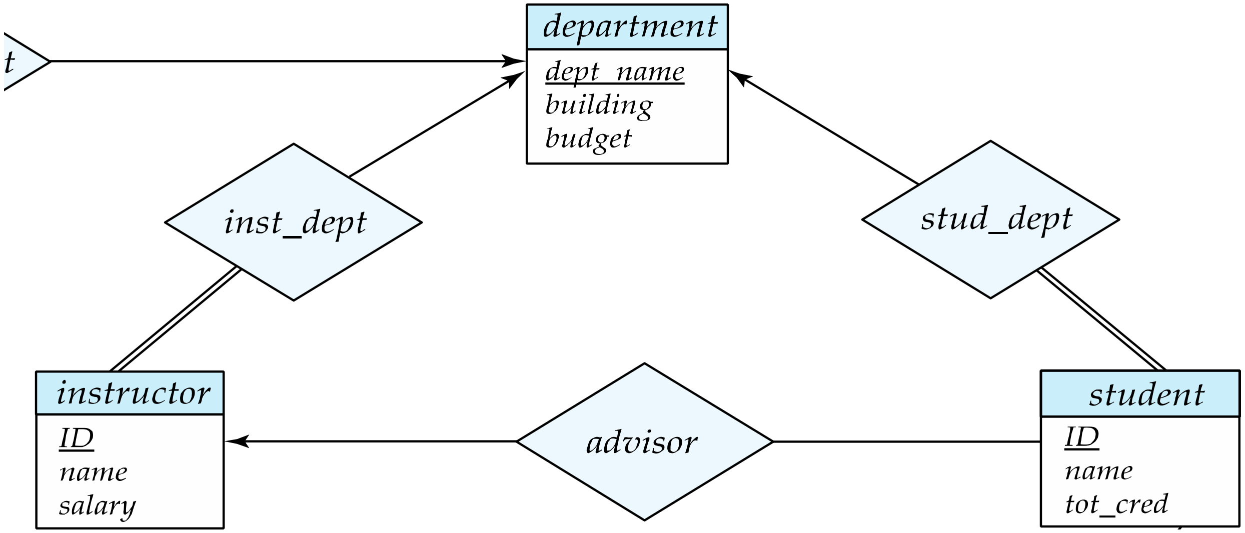
- A many-to-many relationship set is represented as a schema with attributes for the primary keys of the two participating entity sets, and any descriptive attributes of the relationship set
- Example: schema for relationship set *advisor*

advisor = (*s_id, i_id*)



Reduction to Relation 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*



Reduction to Relation Schemas

- 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, replacing a schema by an extra attribute in the schema corresponding to the “many” side could result in null values

Reduction to Relation Schemas

- For one-to-one relationship sets, either side can be chosen to act 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

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*

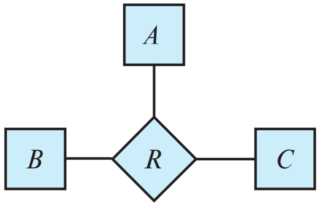
Binary Vs. Non-Binary Relationships

- 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:

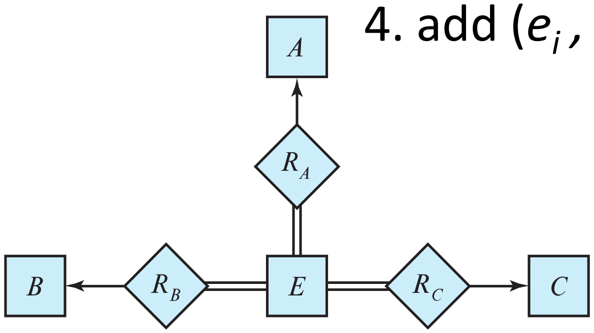
    1. R_A, relating *E* and *A* 2. R_B, relating *E* and *B*
    3. R_C, relating *E* and *C*
  - Create an identifying attribute for *E and* add any attributes of *R* to *E*
  - For each relationship (a_i , b_i , c_i) in *R,* create

    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



(a)



(b)

Binary Vs. Non-Binary Relationships

- Contracts(Supplier, Part, Project)
- A supplier supplies a particular part to a particular project

Supplier	Part	Project
S1	P1	J1
S1	P2	J2
S2	P1	J1

- SUPPLIER — supplies — CONTRACT
- PART — provides — CONTRACT
- PROJECT — used_for — CONTRACT

Binary Vs. Non-Binary Relationships

- CONTRACT Table

ContractID
C1
C2
C3

Binary Vs. Non-Binary Relationships

- SUPPLIER–CONTRACT relationship

Supplier	ContractID
S1	C1
S1	C2
S2	C3

Binary Vs. Non-Binary Relationships

- PART–CONTRACT relationship

Part	ContractID
P1	C1
P2	C2
P1	C3

Binary Vs. Non-Binary Relationships

- PROJECT–CONTRACT relationship

Project	ContractID
J1	C1
J2	C2
J1	C3

Acknowledgement

- Some slides and figures are collected from
  - https://www.db-book.com/