# Principles of Database Systems (CS213)

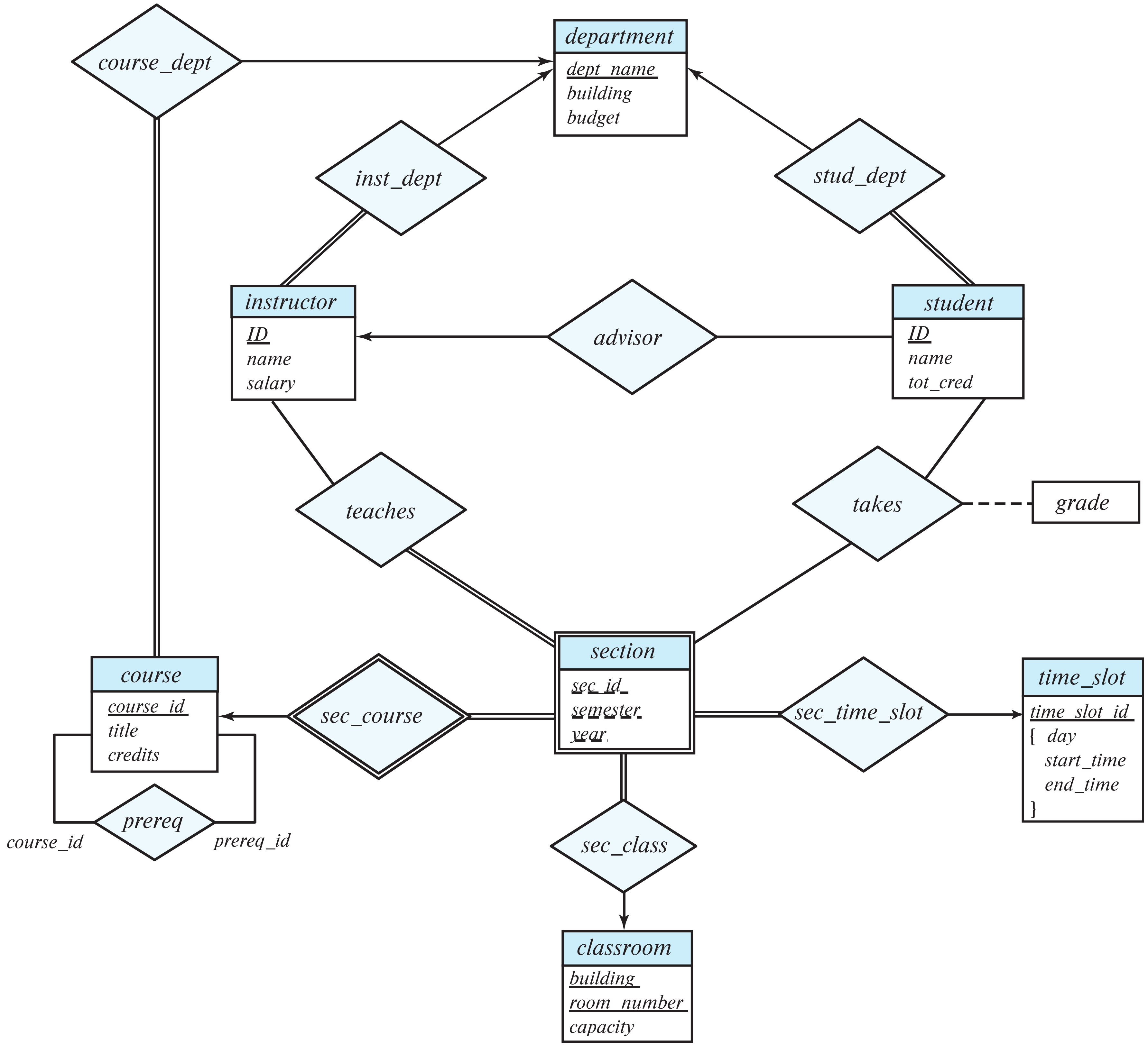
## Lecture 8: Database Design Using the E-R Model

**Yuxin Ma, Shiqi Yu**

Department of Computer Science and Engineering
Southern University of Science and Technology

- Most contents are from slides made by Stéphane Faroult and the authors of Database System Concepts (7th Edition).
- Their original slides have been modified to adapt to the schedule of CS307 at SUSTech.

# The New Running Example



# Design Phases

- Initial phase: characterize fully the data needs of the prospective database users.
- Second phase: choosing a data model
  - Applying the concepts of the chosen data model
  - Translating these requirements into a conceptual schema of the database
  - A fully developed conceptual schema indicates the functional requirements of the enterprise
    - Describe the kinds of operations (or transactions) that will be performed on the data

# Design Phases

- Final Phase: Moving from an abstract data model to the implementation of the database
  - Logical Design – Deciding on the database schema.
    - Database design requires that we find a “good” collection of relation schemas.
    - Business decision – What attributes should we record in the database?
    - Computer Science decision – What relation schemas should we have and how should the attributes be distributed among the various relation schemas?
  - Physical Design – Deciding on the physical layout of the database

# Design Alternatives

- In designing a database schema, we must ensure that we avoid two major pitfalls:
  - **Redundancy**: a bad design may result in <u>repeat information</u>
    - Redundant representation of information may lead to data inconsistency among the various copies of information
  - **Incompleteness**: a bad design may <u>make certain aspects</u> of the enterprise <u>difficult or impossible to model</u>
- Avoiding bad designs is not enough
  - There may be a large number of good designs from which we must choose

# Entity-Relationship Model (E-R Model)
# Entity-Relationship Diagram (E-R Diagram)

# Design Approaches

- Entity Relationship Model (covered in this chapter)
  - Models an enterprise as <u>a collection of</u> entities and relationships
    - **Entity**: a “thing” or “object” in the enterprise that is distinguishable from other objects
      - Described by a set of attributes
    - **Relationship**: an association among several entities
  - Represented diagrammatically by an entity-relationship diagram (E-R diagram)

- Normalization Theory (coming in the next few weeks)
  - Formalize what designs are bad, and test for them

# Entity Sets

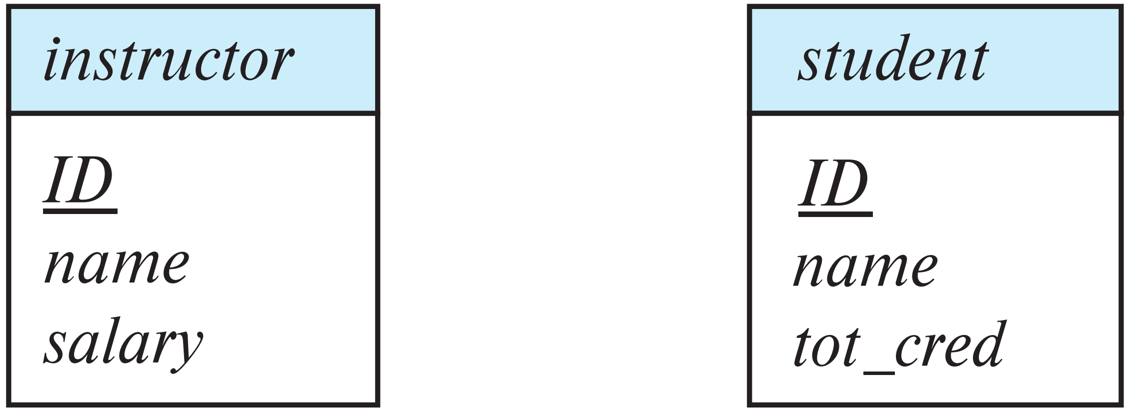
- An entity is an object that exists and is distinguishable from other objects
  - Example: specific person, company, event, plant
- An entity set is a set of entities of the same type that share the same properties
  - Example: set of all persons, companies, trees, holidays
- An entity is represented 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.

# Representing Entity sets in ER Diagram

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



# Relationship Sets

- A relationship is an association among several entities

```
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

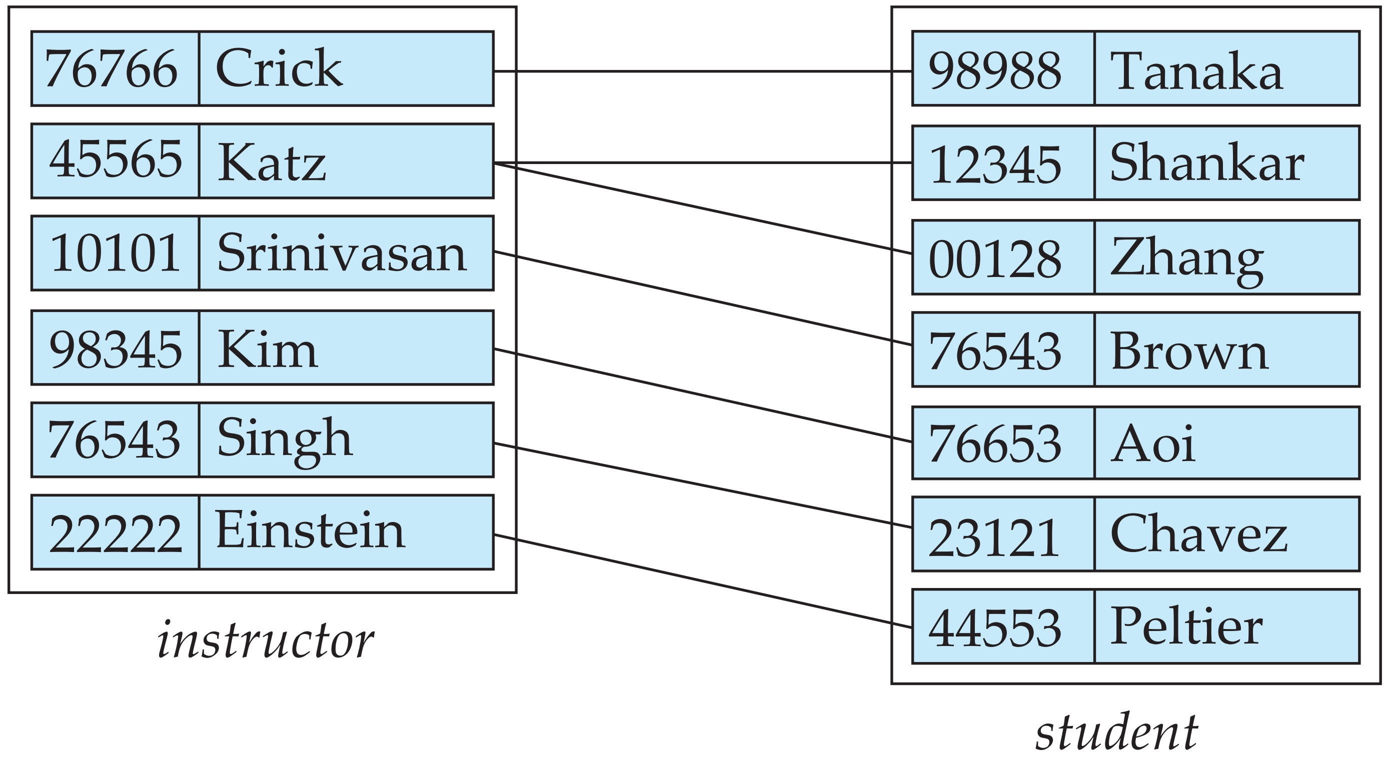
$$\{(e_1, e_2, \ldots e_n) \mid e_1 \in E_1, e_2 \in E_2, \ldots, e_n \in E_n\}$$

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

- Example: $(44553,22222) \in advisor$

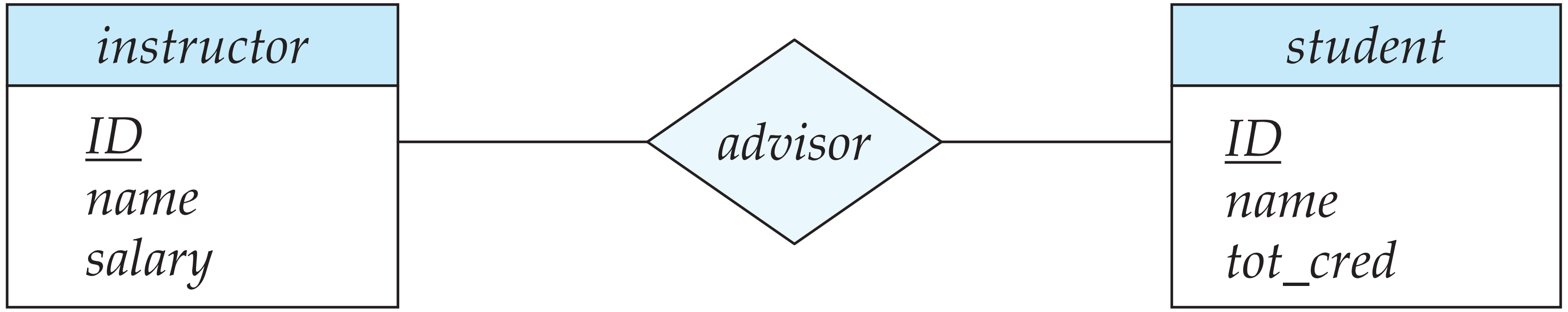
# Relationship Sets

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



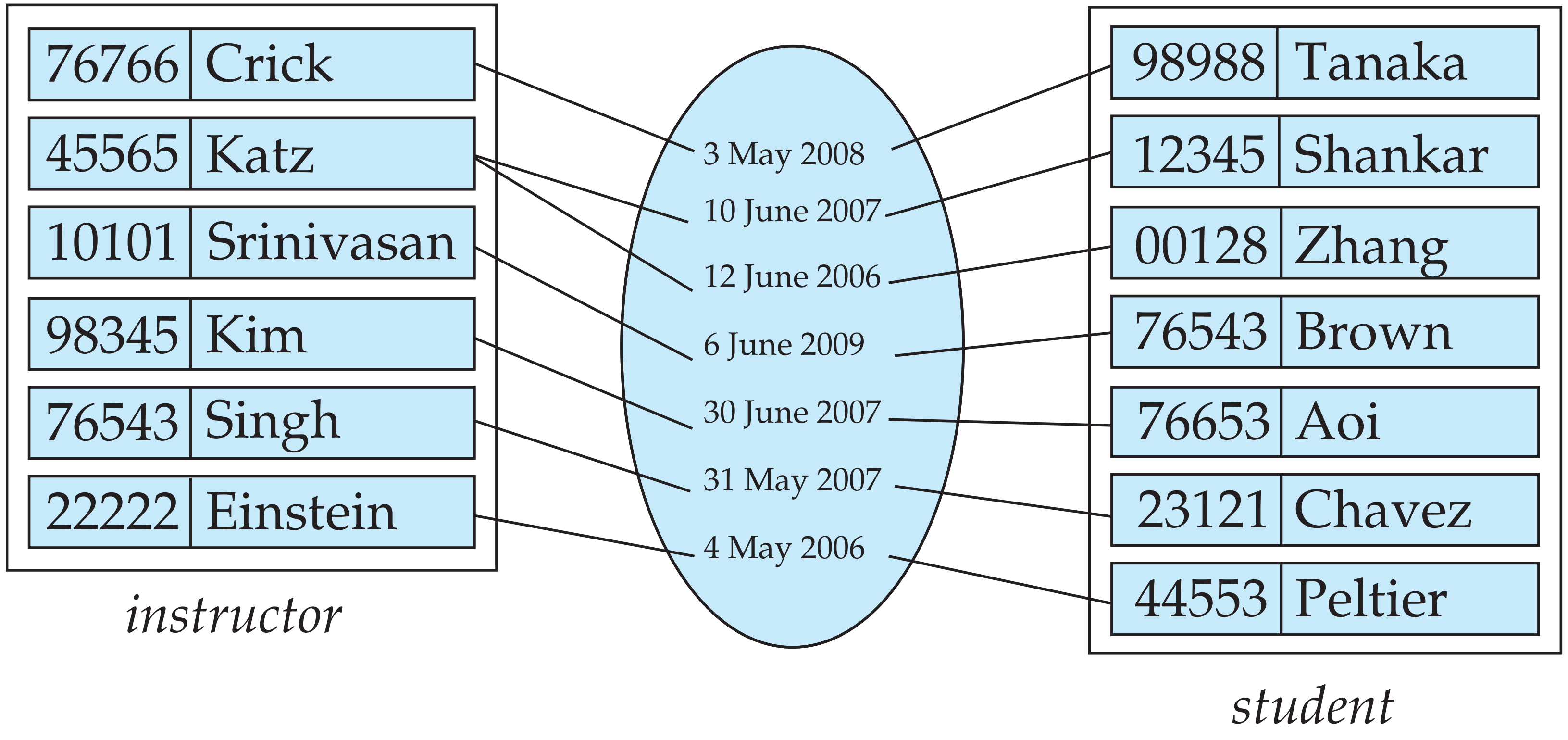
# Representing Relationship Sets via E-R Diagrams

- Diamonds represent relationship sets

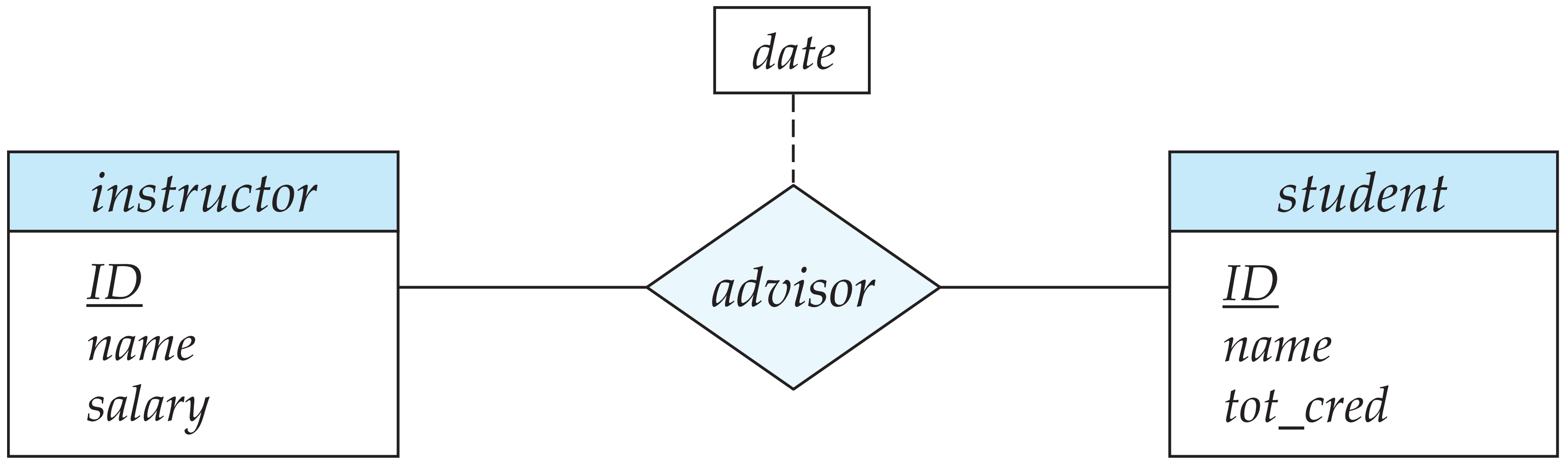


# Relationship Sets (Cont.)

- 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



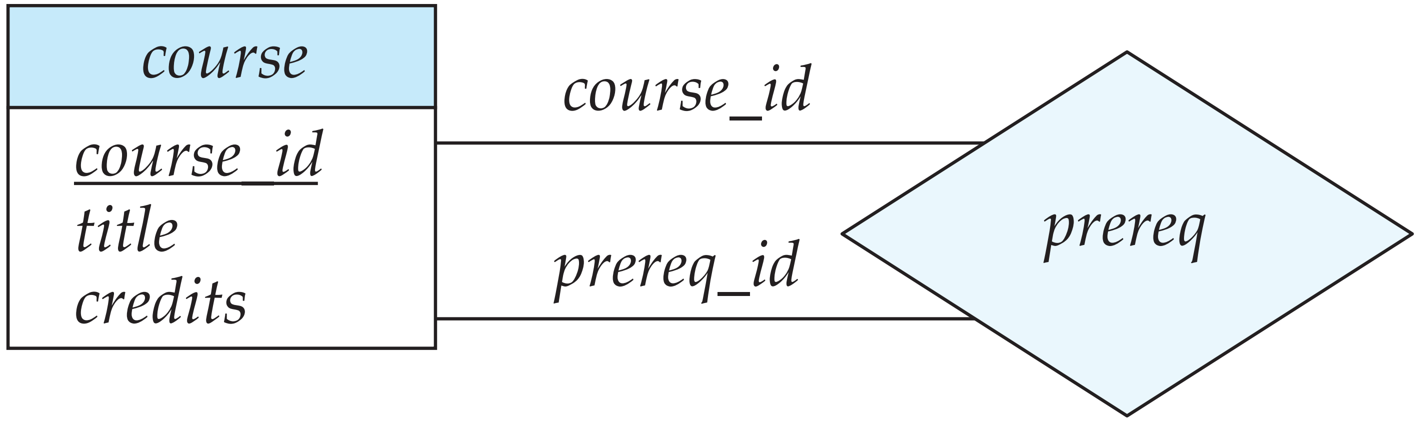
# Relationship Sets with Attributes



# Roles

- Entity sets of a relationship need not be distinct
  - That is to say, we can create self-pointing relationships for an entity set
  - Each occurrence of an entity set plays a "role" in the relationship

  - Example: A relationship set to represent the prerequisites of a course
    - E.g., Data Structure depends on Introduction to Programming
    - The labels "`course_id`" and "`prereq_id`" are called roles

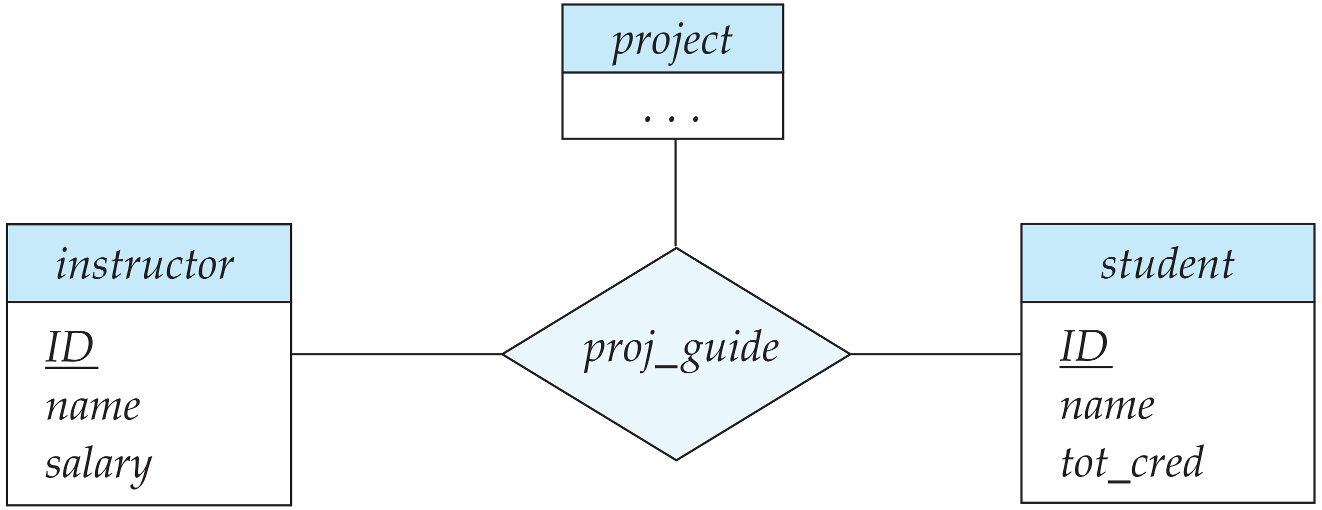


# Degree of a Relationship Set

- Binary relationship
  - Involve **two** entity sets (or degree two).
  - Most relationship sets in a database system are binary
- 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

# 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



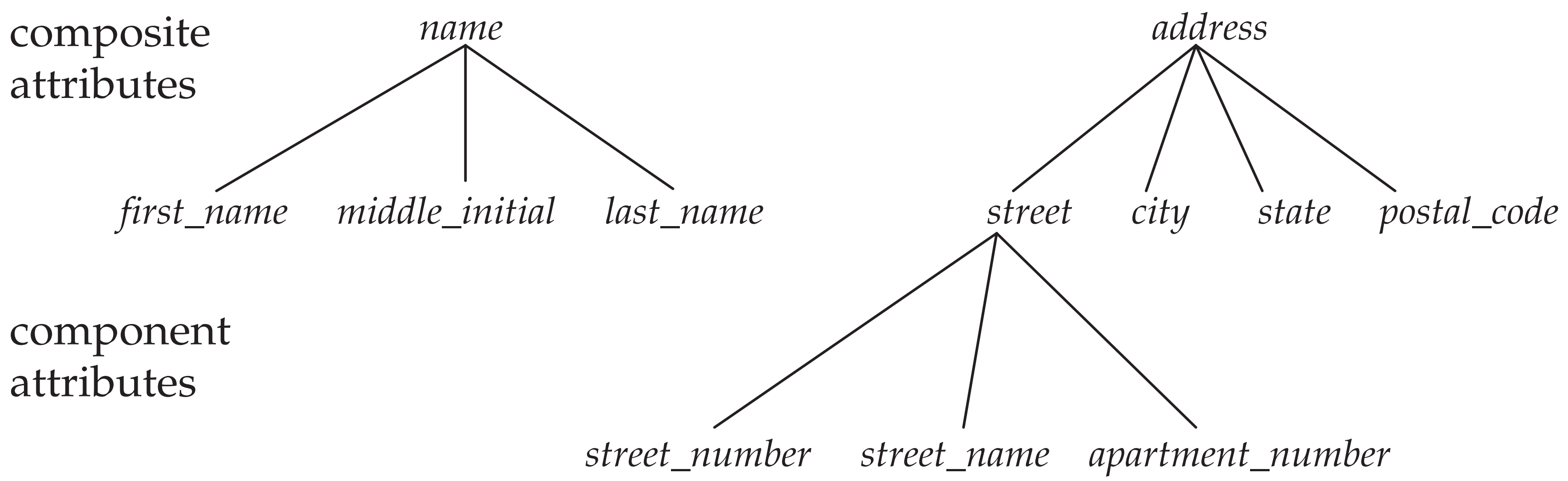
# Complex Attributes

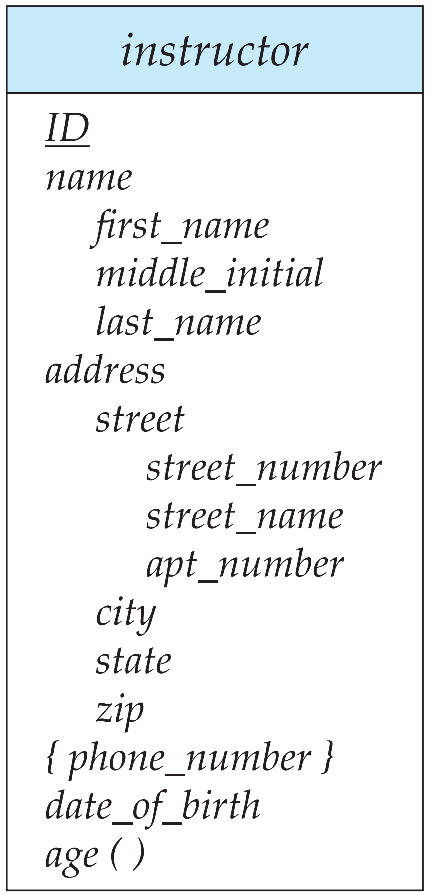
- Attribute types:
  - Simple and composite attributes.
  - Single-valued and multivalued attributes
    - Example: multivalued attribute: phone_numbers
      - A person can have 1 or more phone numbers at the same time
  - Derived attributes
    - Can be computed from other attributes
    - Example:  age, given date_of_birth

- **Domain**: The set of permitted values for each attribute

# Composite Attributes

- Composite attributes allow us to divided attributes into subparts (other attributes)
  - Sometimes we may only use part of the attributes, where the composite attribute is a good design choice

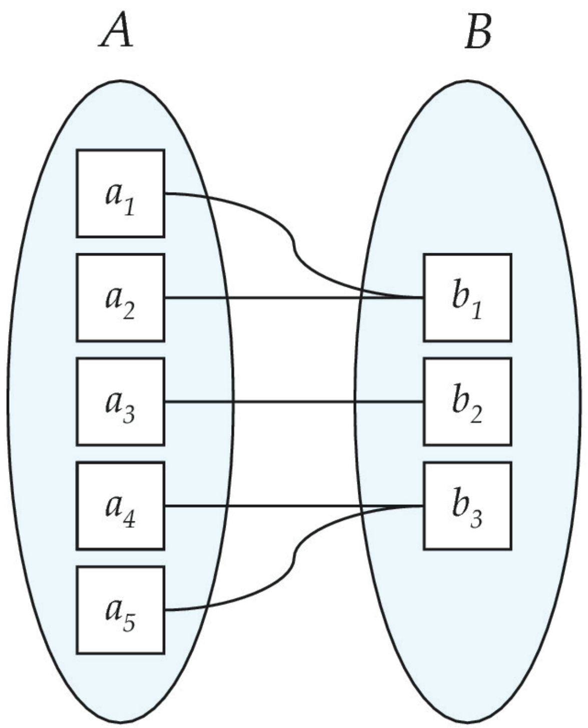


# Mapping Cardinality Constraints

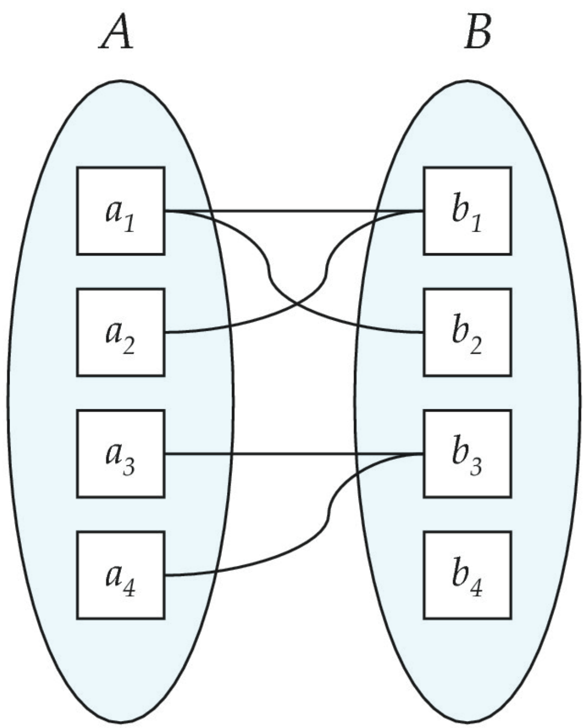
- Mapping Cardinality（映射基数）
  - Express the number of entities to which another entity can be associated via a relationship set.
    - Most useful in describing binary relationship sets
- For a binary relationship set, the mapping cardinality must be one of the following types:
  - One to one
  - One to many
  - Many to one
  - Many to many

# Mapping Cardinalities




(a)

Many to one

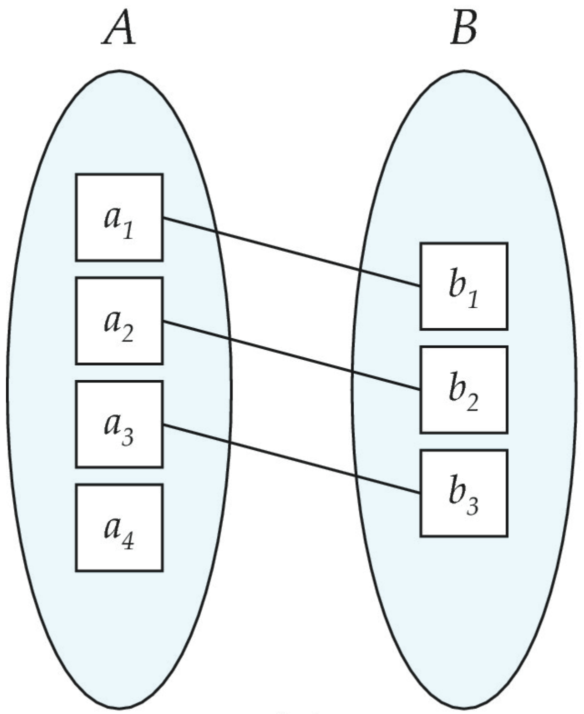



(b)

Many to many

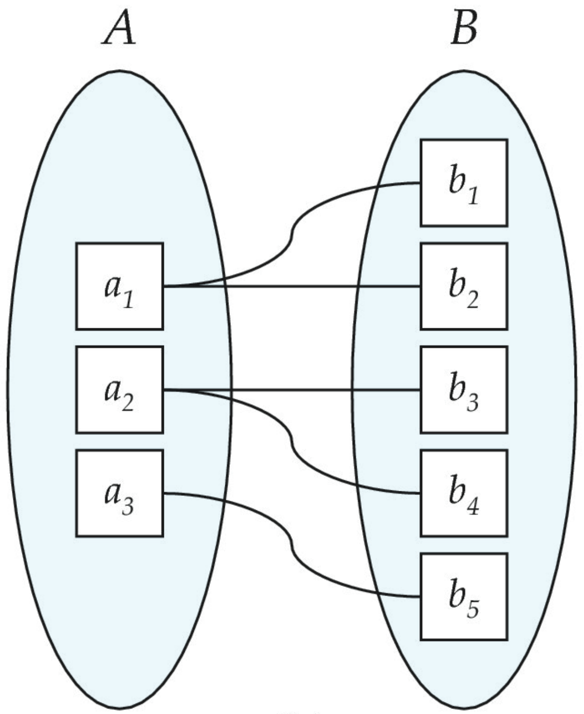
Note: Some elements in *A* and *B* may not be mapped to any elements in the other set

# Mapping Cardinalities




(a)

One to one




(b)

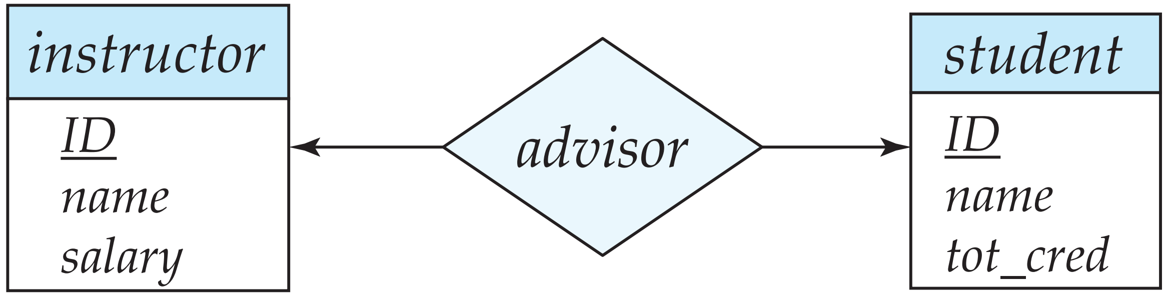
One to many

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

# Representing Cardinality Constraints in ER Diagram

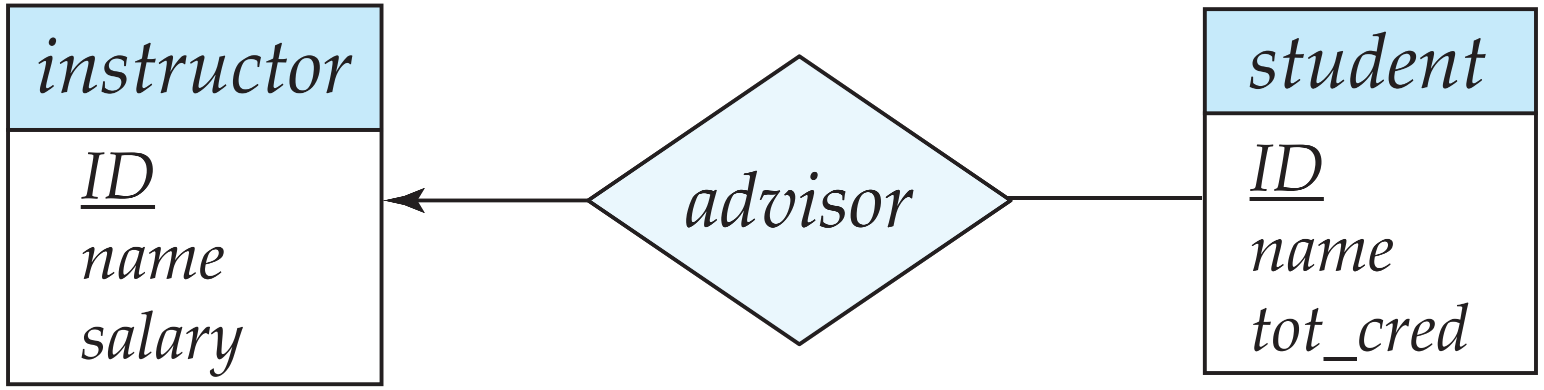
- We express cardinality constraints by:
  - drawing either a directed line (→), signifying "one,"
  - or an undirected line (—), signifying "many,"
- … between the relationship set and the entity set.

- <u>One-to-one relationship</u> between an instructor and a student :
  - A student is associated with at most one instructor via the relationship advisor



# Representing Cardinality Constraints in ER Diagram

- 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

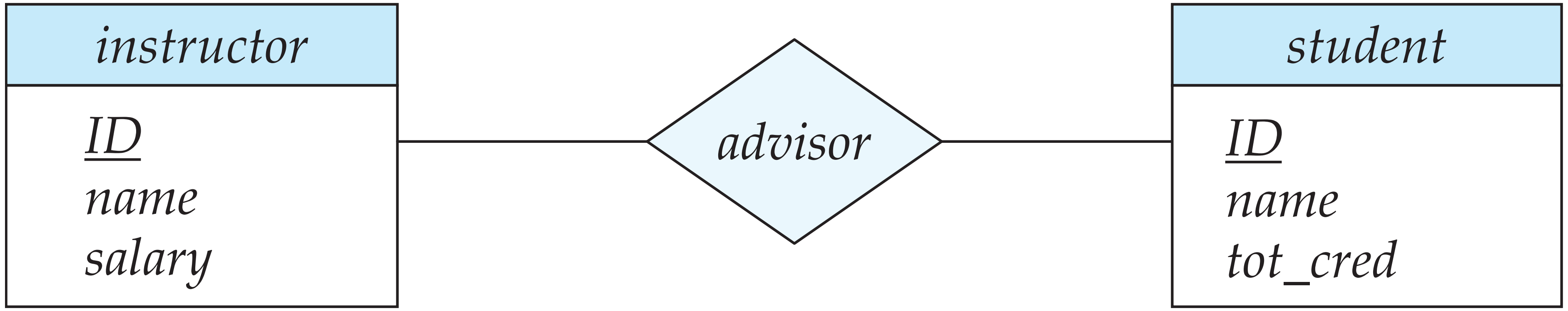


# Representing Cardinality Constraints in ER Diagram

- In a <u>many-to-one relationship</u> 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

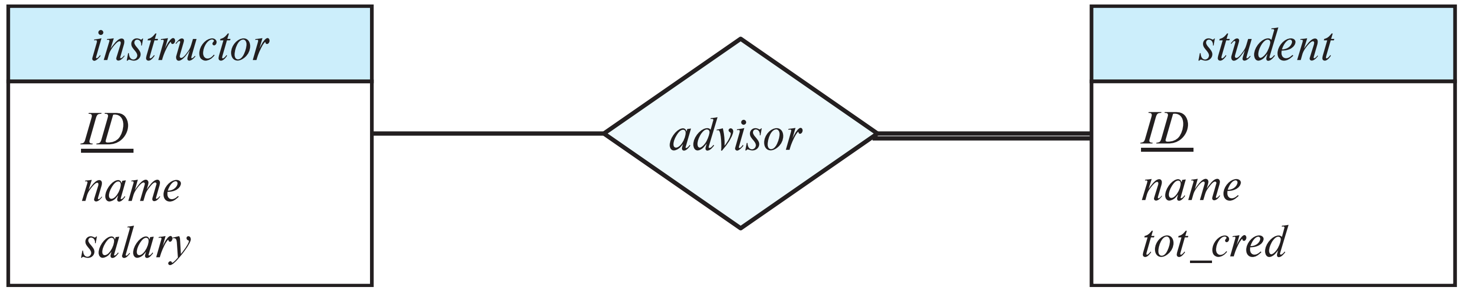
# Representing Cardinality Constraints in ER Diagram

- 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



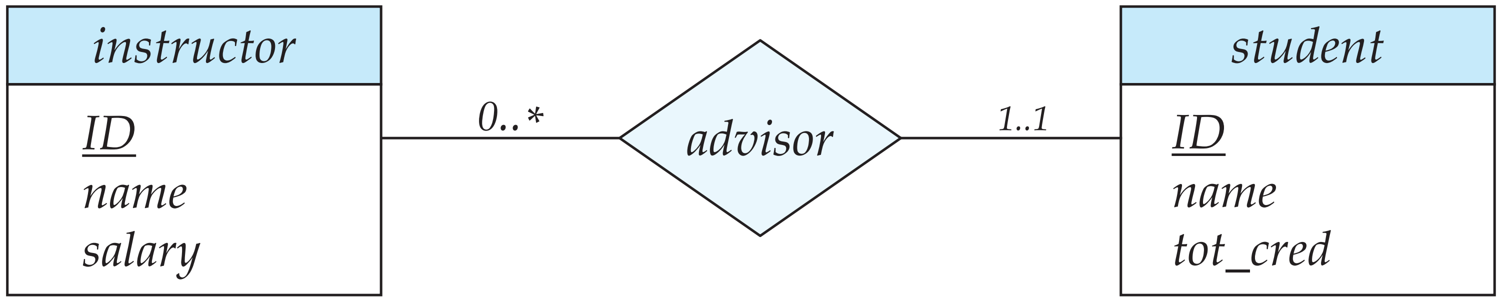
# Total and Partial Participation

- 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
    - i.e., 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



# Notation for Expressing More Complex Constraints

- 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.




- Example
  - Instructor can advise 0 or more students
  - A student must have 1 advisor; cannot have multiple advisors

# Primary Key

- Primary keys provide a way to specify how entities and relations are distinguished

# 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 <u>suffice to distinguish entities from each other</u>

# 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 consists of the union of the primary keys of entity sets E1, E2, ..En
  - If the relationship set R has attributes $a_1, a_2, .., a_m$ associated with it, the primary key of R also includes the attributes $a_1, a_2, .., a_m$
- Example: relationship set “advisor”.
  - The primary key consists of instructor.ID and student.ID

- The choice of the primary key for a relationship set depends on the mapping cardinality of the relationship set.

# Choice of Primary key for Binary Relationship

- Many-to-Many relationships
  - The preceding union of the primary keys is a minimal superkey and is chosen 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.

* *K* is a **superkey** of *R* 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*.

# Choice of Primary key for Binary Relationship

- 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.

# Weak Entity Sets

- Consider a section entity, which is uniquely identified by a course_id, semester, year, and sec_id.
  - Clearly, section entities are related to course entities. Suppose we create a relationship set sec_course between entity sets section and course.
  - Note that the information in sec_course is redundant, since section already has an attribute course_id, which identifies the course with which the section is related.
  - One option to deal with this redundancy is to get rid of the relationship sec_course; however, by doing so the relationship between section and course becomes implicit in an attribute, which is not desirable.

# Weak Entity Sets

- An alternative way to deal with this redundancy is to not store the attribute course_id in the section entity and to only store the remaining attributes section_id, year, and semester.
    - However, the entity set section then does not have enough attributes to identify a particular section entity uniquely
- To deal with this problem, we treat the relationship sec_course as a special relationship that provides extra information, in this case, the course_id, required to identify section entities uniquely.

- A weak entity set is one whose existence is dependent on another entity, called its <u>identifying entity</u>
- Instead of associating a primary key with a weak entity, we use the identifying entity, along with extra attributes called discriminator to uniquely identify a weak entity.

# Weak Entity Sets

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

- Every weak entity must be associated with an identifying entity; that is, the weak entity set is said to be existence dependent on the identifying entity set.
  - 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

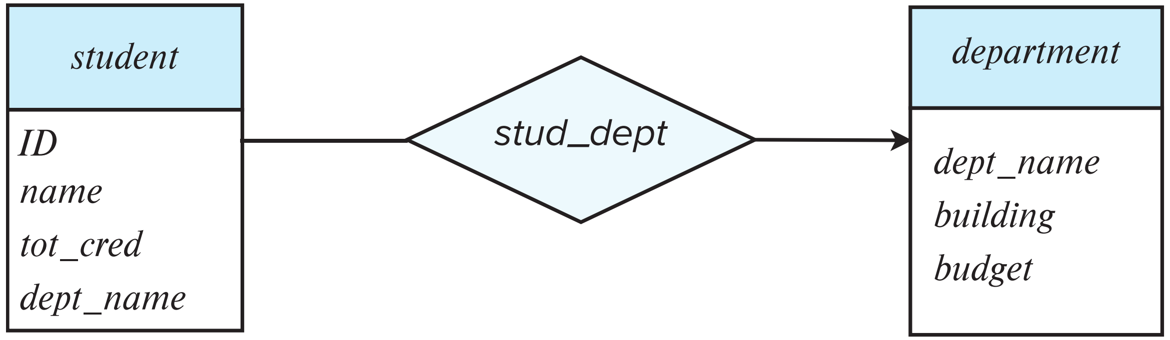
- Note that the relational schema we eventually create from the entity set section does have the attribute course_id, for reasons that will become clear later, even though we have dropped the attribute course_id from the entity set section.

# Expressing Weak Entity Sets

- 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

- Suppose we have entity sets:
  - student, with attributes: ID, name, tot_cred, dept_name
  - department, with attributes: dept_name, building, budget
- 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.




(a) Incorrect use of attribute

- BUT: when converting back to tables, in some cases the attribute gets reintroduced.

# Reduction to Relation Schemas

# 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

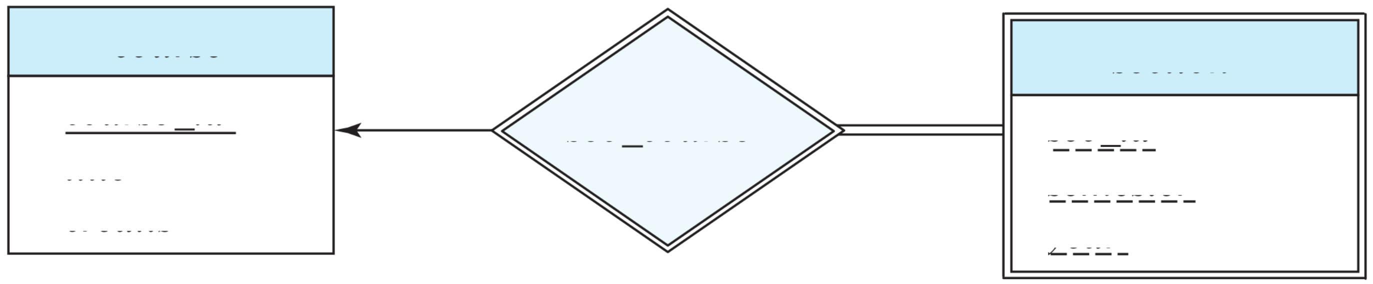
- A strong entity set reduces to a schema with the same attributes

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

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

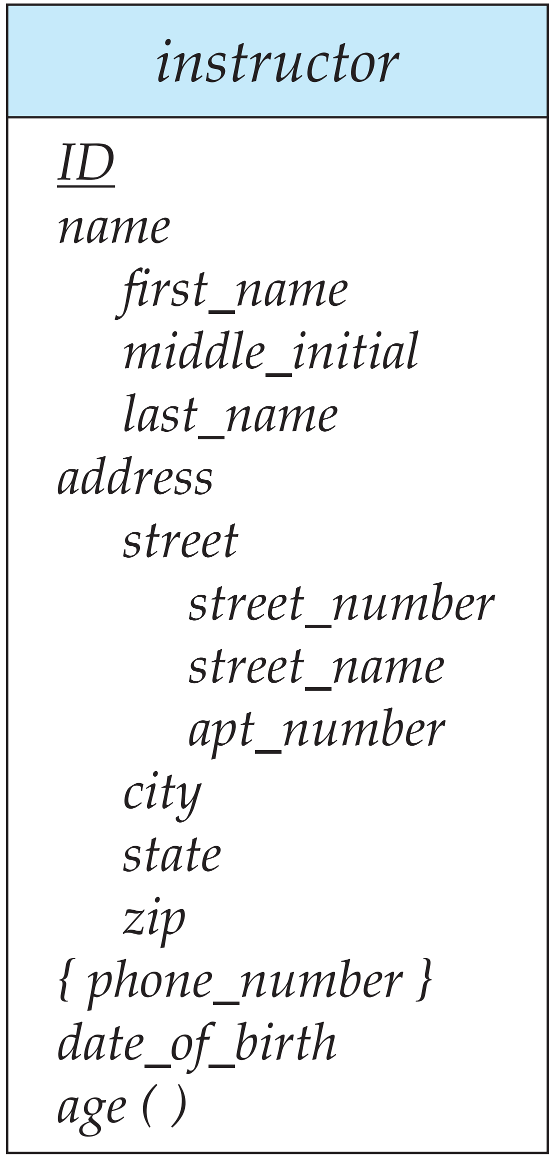
  *section (<u>course_id, sec_id, sem, year</u>)*

- Example



# Representation of Entity Sets with Composite Attributes

- 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 (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

- 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>*)

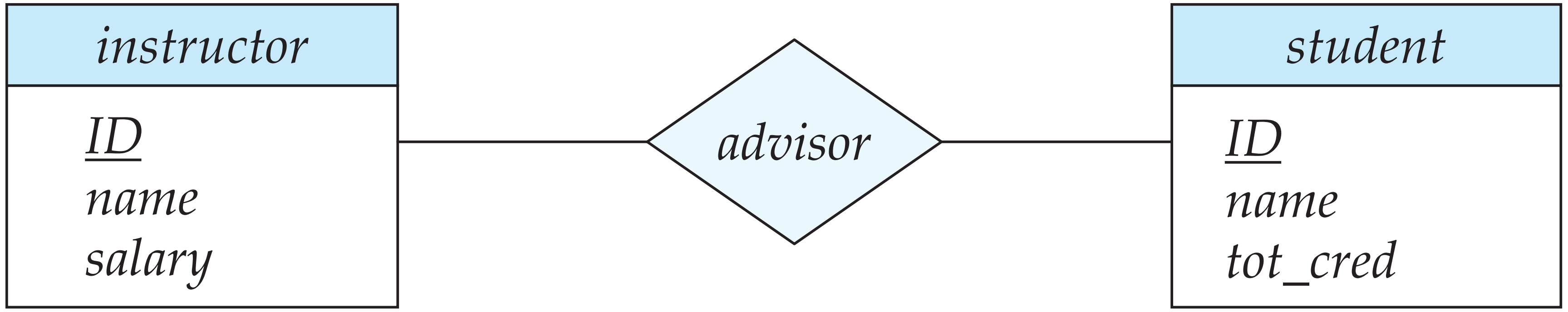
  - Each value of the multivalued attribute maps to a separate tuple of the relation on schema EM
  - For example, an instructor entity with primary key 22222 and phone numbers 456-7890 and 123-4567 maps to two tuples:

    *(22222, 456-7890)* and *(22222, 123-4567)*

# Representing Relationship Sets

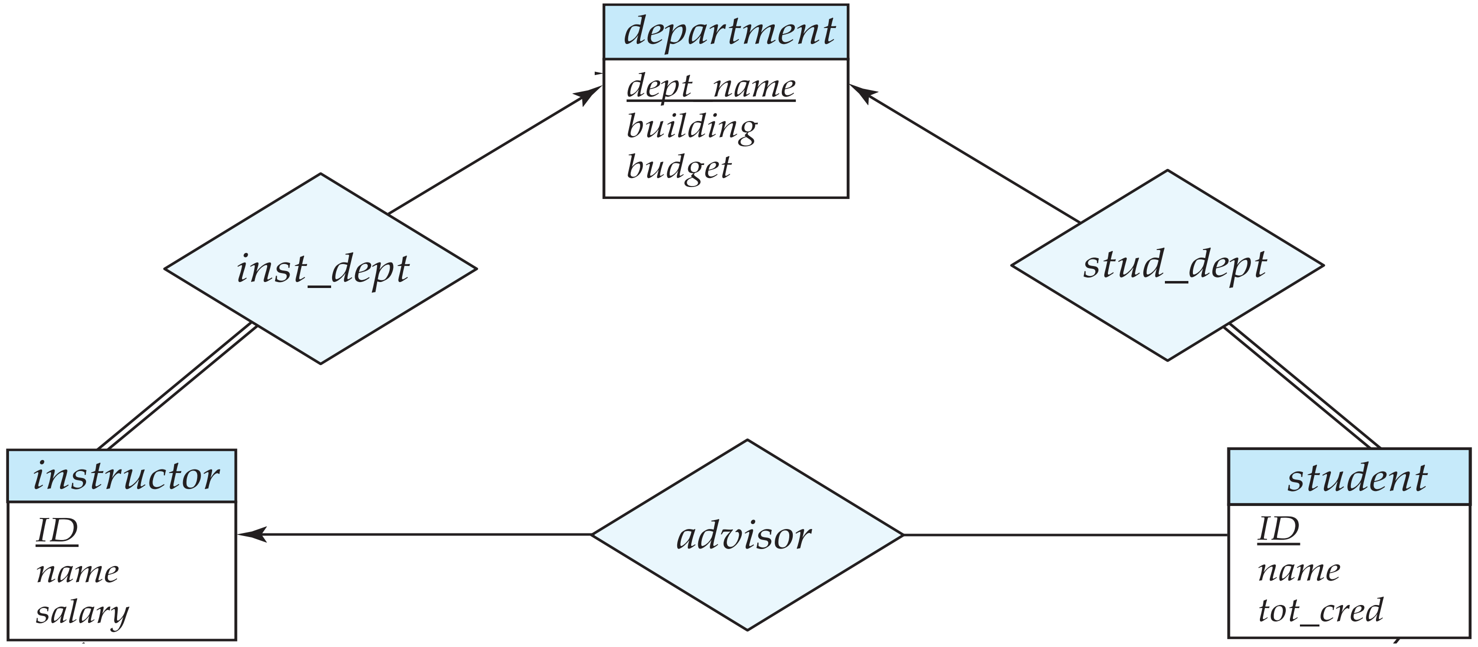
- A many-to-many relationship set is represented as a schema with attributes for the primary keys of the two participating entity sets, and any descriptive attributes of the relationship set.
  - Example: schema for relationship set advisor

*advisor* = (<u>*s_id*</u>, <u>*i_id*</u>)



# 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*
  - Example



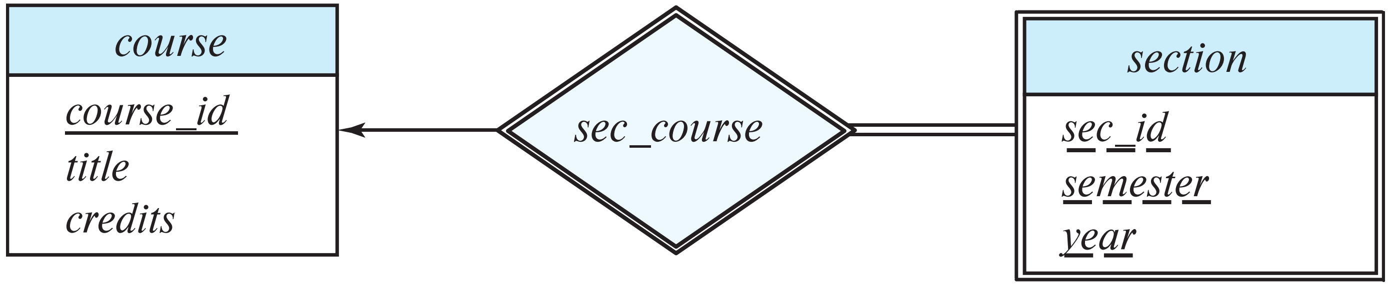
# Redundancy of Schemas

- 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

# 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



# Normalization

# Recall: Design Alternatives

- In designing a database schema, we must ensure that we avoid two major pitfalls:
  - **Redundancy**:  a bad design may result in repeat information
    - Redundant representation of information may lead to data inconsistency among the various copies of information
  - **Incompleteness**: a bad design may make certain aspects of the enterprise difficult or impossible to model
- Avoiding bad designs is not enough
  - There may be a large number of good designs from which we must choose

# Recall: Design Alternatives

- In designing a database schema, we must ensure that we avoid two major pitfalls:
  - **Redundancy**: a bad design may result in repeat information
    - Redundant representation of information may lead to data inconsistency among the various copies of information
  - **Incompleteness**: a bad design may make certain aspects of the enterprise difficult or impossible to model
- Avoiding bad designs is not enough
  - There may be a large number of good designs from which we must choose
- Do we have any guidelines on how to get a good design?
  - Normal Forms!

# First Normal Form (1NF)

- A relational schema R is in **first normal form** if the domains of all attributes of R are atomic
  - Domain is atomic if its elements are considered to be indivisible units
    - Examples of non-atomic domains:
      - Set of names, composite attributes
      - Identification numbers like CS101 that can be broken up into parts
        - » However, in practice, we can also consider it atomic
  - Non-atomic values complicate storage and encourage redundant (repeated) storage of data

# First Normal Form (1NF)

- Example: Non-atomic attribute

| station_id | name | location |
|---|---|---|
| 1 | Luohu(罗湖) | 114.11833 , 22.53111 |
| 2 | Guomao(国贸) | 114.11889 , 22.54 |
| 3 | Laojie(老街) | 114.11639 , 22.54444 |
| 4 | Grand Theater(大剧院) | 114.10333 , 22.54472 |
| 5 | Science Museum(科学馆) | 114.08972 , 22.54333 |
| 6 | Huaqiang Rd(华强路) | 114.07889 , 22.54306 |
| 7 | Gangxia(岗厦) | 114.06306 , 22.53778 |
| 8 | Convention and Exhibition Center Station(会展中心) | 114.05472 , 22.5375 |
| 9 | Shopping Park(购物公园) | 114.05472 , 22.53444 |
| 10 | Xiangmihu(香蜜湖) | 114.034 , 22.5417 |

# First Normal Form (1NF)

- Another example: Starring
  - Problems: 1) Redundant names; 2) difficulties in updating/deleting a specific person; 3) extra cost in splitting names; 4) difficulties in making statistics

| Movie ID | Movie Title | Country | Year | Director | Starring |
|---|---|---|---|---|---|
| 0 | Citizen Kane | US | 1941 | welles, o. | Orson Welles, Joseph Cotten |
| 1 | La règle du jeu | FR | 1939 | Renoir, J. | Roland Toutain, Nora Grégor, Marcel Dalio, Jean Renoir |
| 2 | North By Northwest | US | 1959 | HITCHCOCK, A. | Cary Grant, Eva Marie Saint, James Mason |
| 3 | Singin' in the Rain | US | 1952 | Donen/Kelly | Gene Kelly, Debbie Reynolds, Donald O'Connor |
| 4 | Rear Window | US | 1954 | Alfred Hitchcock | James Stewart, Grace Kelly |

# First Normal Form (1NF)

- Fix it by splitting the names into two columns

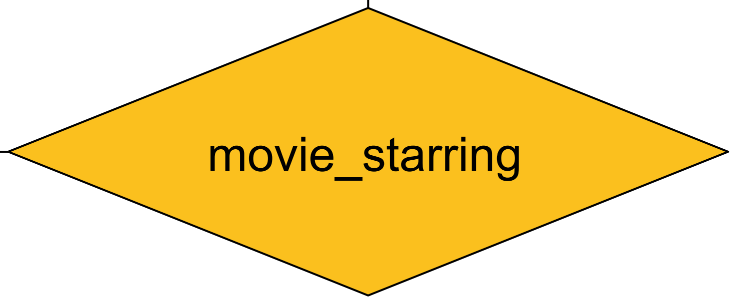
| station_id | english_name | chinese_name | longitude | latitude |
|---|---|---|---|---|
| 1 | Luohu | 罗湖 | 114.11833 | 22.53111 |
| 2 | Guomao | 国贸 | 114.11889 | 22.54 |
| 3 | Laojie | 老街 | 114.11639 | 22.54444 |
| 4 | Grand Theater | 大剧院 | 114.10333 | 22.54472 |
| 5 | Science Museum | 科学馆 | 114.08972 | 22.54333 |
| 6 | Huaqiang Rd | 华强路 | 114.07889 | 22.54306 |
| 7 | Gangxia | 岗厦 | 114.06306 | 22.53778 |
| 8 | Convention and Exhibition Cent… | 会展中心 | 114.05472 | 22.5375 |
| 9 | Shopping Park | 购物公园 | 114.05472 | 22.53444 |
| 10 | Xiangmihu | 香蜜湖 | 114.034 | 22.5417 |

# First Normal Form (1NF)

- Fix it by treating the column as a multi-valued attribute

| Movie ID | Movie Title | Country | Year | Director |
|---|---|---|---|---|
| 0 | Citizen Kane | US | 1941 | welles, o. |
| 1 | La règle du jeu | FR | 1939 | Renoir, J. |
| 2 | North By Northwest | US | 1959 | HITCHCOCK, A. |
| 3 | Singin' in the Rain | US | 1952 | Donen/Kelly |
| 4 | Rear Window | US | 1954 | Alfred Hitchcock |

| Star ID | Firstname | Lastname | Born | Died |
|---|---|---|---|---|
| 1 | | | | |
| 2 | | | | |
| 3 | | | | |



# Second Normal Form (2NF)

- A relation satisfying 2NF must:
  - be in 1NF
  - not have any non-prime attribute that is dependent on any proper subset of any candidate key of the relation
    - A non-prime attribute of a relation is an attribute that is not a part of any candidate key of the relation.

# Second Normal Form (2NF)

- Example: Consider this table with the composite primary key (*station_id, line_id*)

| station_id | english_name | chinese_name | district | line_id | line_color | operator |
|---|---|---|---|---|---|---|
| 1 | Luohu | 罗湖 | Luohu | 1 | Green | Shenzhen Metro Corporation |
| 2 | Guomao | 国贸 | Luohu | 1 | Green | Shenzhen Metro Corporation |
| 3 | Laojie | 老街 | Luohu | 1 | Green | Shenzhen Metro Corporation |
| 4 | Grand Theater | 大剧院 | Luohu | 1 | Green | Shenzhen Metro Corporation |
| 4 | Grand Theater | 大剧院 | Luohu | 11 | Purple | Shenzhen Metro Corporation |
| 4 | Grand Theater | 大剧院 | Luohu | 2 | Orange | Shenzhen Metro Corporation |
| 3 | Laojie | 老街 | Luohu | 3 | DeepSkyBlue | Shenzhen Metro No.3 Line |

- The columns *line_color* and *operator* are not related to station_id
  - They are only related to *line_id*, which is only part of (a subset of) the primary key
- Similarly, *english_name*, *chinese_name*, and *district* are not related to *line_id*
  - They are only related to *station_id*, which is only part of (a subset of) the primary key

# Second Normal Form (2NF)

- Example: Consider this table with the composite primary key *(station_id, line_id)*

| station_id | english_name | chinese_name | district | line_id | line_color | operator |
|---|---|---|---|---|---|---|
| 1 | Luohu | 罗湖 | Luohu | 1 | Green | Shenzhen Metro Corporation |
| 2 | Guomao | 国贸 | Luohu | 1 | Green | Shenzhen Metro Corporation |
| 3 | Laojie | 老街 | Luohu | 1 | Green | Shenzhen Metro Corporation |
| 4 | Grand Theater | 大剧院 | Luohu | 1 | Green | Shenzhen Metro Corporation |
| 4 | Grand Theater | 大剧院 | Luohu | 11 | Purple | Shenzhen Metro Corporation |
| 4 | Grand Theater | 大剧院 | Luohu | 2 | Orange | Shenzhen Metro Corporation |
| 3 | Laojie | 老街 | Luohu | 3 | DeepSkyBlue | Shenzhen Metro No.3 Line |

  - Problem when not meeting 2NF: Insertion and deletion anomaly
    - We cannot insert a new station with no lines assigned yet (unless using NULLs)
    - If we delete a line, all stations associated with this line will be deleted as well

# Second Normal Form (2NF)

- Fix it by
  - Splitting the two unrelated parts into two different tables of entities
  - And create a relationship set (if it is the many-to-many relationship between the two entities)

- By the way…
  - A relation with a single-attribute primary key is automatically in 2NF once it meets 1NF.

stations

| station_id | english_name | chinese_name | district |
|---|---|---|---|
| 1 | Luohu | 罗湖 | Luohu |
| 2 | Guomao | 国贸 | Luohu |
| 3 | Laojie | 老街 | Luohu |
| 4 | Grand Theater | 大剧院 | Luohu |

Foreign key

primary key(line_id, station_id)

line_detail

| line_id | station_id | num | dist |
|---|---|---|---|
| 1 | 1 | 1 | 0 |
| 1 | 2 | 2 | 1 |
| 1 | 3 | 3 | 1 |
| 1 | 4 | 4 | 1 |
| 11 | 4 | 21 | <null> |
| 2 | 4 | 26 | 2 |
| 3 | 3 | 10 | 2 |

Foreign key

lines

| line_id | line_color | operator |
|---|---|---|
| 1 | Green | Shenzhen Metro Corporation |
| 2 | Orange | Shenzhen Metro Corporation |
| 3 | DeepSkyBlue | Shenzhen Metro No.3 Line |
| 11 | Purple | Shenzhen Metro Corporation |

# Third Normal Form (3NF)

- A relation satisfying 3NF must:
  - be in 2NF
  - all the attributes in a table are determined only by the candidate keys of that relation and not by any non-prime attributes

# Third Normal Form (3NF)

- Example: Consider this table which describes the bus lines and their stops
  - Primary key (*bus_line*)

| bus_line | station_id | chinese_name | english_name | district |
|---|---|---|---|---|
| B796 | 21 | 鲤鱼门 | Liyumen | Nanshan |
| M343 | 21 | 鲤鱼门 | Liyumen | Nanshan |
| M349 | 21 | 鲤鱼门 | Liyumen | Nanshan |
| M250 | 26 | 坪洲 | Pingzhou | Bao'an |
| 374 | 61 | 安托山 | Antuo Hill | Futian |
| B733 | 61 | 安托山 | Antuo Hill | Futian |
| B828 | 120 | 临海 | Linhai | Nanshan |

- The column *station_id* depends on the primary key (bus_line)
- However, the columns *chinese_name*, *english_name*, and *district* depend on *station_id*, which is not the primary key.
  - They only have "indirect/transitive" dependence on the primary key
- Problem: Data redundancy

# Third Normal Form (3NF)

- Example: Consider this table which describes the bus lines and their stops
  - Primary key (*bus_line*)

| bus_line | station_id | chinese_name | english_name | district |
|---|---|---|---|---|
| B796 | 21 | 鲤鱼门 | Liyumen | Nanshan |
| M343 | 21 | 鲤鱼门 | Liyumen | Nanshan |
| M349 | 21 | 鲤鱼门 | Liyumen | Nanshan |
| M250 | 26 | 坪洲 | Pingzhou | Bao'an |
| 374 | 61 | 安托山 | Antuo Hill | Futian |
| B733 | 61 | 安托山 | Antuo Hill | Futian |
| B828 | 120 | 临海 | Linhai | Nanshan |

- Problem when not meeting 3NF:
  - Data redundancy: as you can see in the table, the attributes for a station have been stored multiple times
  - Insertion and deletion anomaly: inserting a new bus line with no station becomes impossible without NULLs; deleting a station/bus line may also delete corresponding bus lines/stations.

# Third Normal Form (3NF)

- Fix it by:
  - <u>Create a new table</u> with *station_id* as the primary key
    - i.e., the column which *chinese_name*, *english_name*, and *district* depend on
  - Move all columns which depend on the new primary key into the new table
    - And, only leave the primary key of the new table (*station_id*) in the original table
  - (*In practice, if necessary) Add a foreign-key constraint
    - Not related to relational database modeling, only in implementations

stations

| station_id | chinese_name | english_name | district |
|---|---|---|---|
| 21 | 鲤鱼门 | Liyumen | Nanshan |
| 26 | 坪洲 | Pingzhou | Bao'an |
| 61 | 安托山 | Antuo Hill | Futian |
| 120 | 临海 | Linhai | Nanshan |
| 121 | 宝华 | Baohua | Bao'an |

Foreign key

bus_lines

| station_id | bus_line |
|---|---|
| 21 | B796 |
| 21 | M343 |
| 21 | M349 |
| 26 | M250 |
| 61 | 374 |
| 61 | B733 |
| 120 | B828 |
| 121 | B828 |
| 121 | M235 |

# Normalization

- In practice, we usually just satisfy 1NF, 2NF and 3NF

| | UNF (1970) | 1NF (1970) | 2NF (1971) | 3NF (1971) | EKNF (1982) | BCNF (1974) | 4NF (1977) | ETNF (2012) | 5NF (1979) | DKNF (1981) | 6NF (2003) |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Primary key (no duplicate tuples)[4] | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| Atomic columns (cells cannot have tables as values)[5] | ✗ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| Every non-trivial functional dependency either does not begin with a proper subset of a candidate key or ends with a prime attribute (no partial functional dependencies of non-prime attributes on candidate keys)[5] | ✗ | ✗ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| Every non-trivial functional dependency either begins with a superkey or ends with a prime attribute (no transitive functional dependencies of non-prime attributes on candidate keys)[5] | ✗ | ✗ | ✗ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| Every non-trivial functional dependency either begins with a superkey or ends with an elementary prime attribute | ✗ | ✗ | ✗ | ✗ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | N/A |
| Every non-trivial functional dependency begins with a superkey | ✗ | ✗ | ✗ | ✗ | ✗ | ✓ | ✓ | ✓ | ✓ | ✓ | N/A |
| Every non-trivial multivalued dependency begins with a superkey | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✓ | ✓ | ✓ | ✓ | N/A |
| Every join dependency has a superkey component[8] | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✓ | ✓ | ✓ | N/A |
| Every join dependency has only superkey components | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✓ | ✓ | N/A |
| Every constraint is a consequence of domain constraints and key constraints | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✓ | ✗ |
| Every join dependency is trivial | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✓ |

https://en.wikipedia.org/wiki/Database_normalization

# Normalization

**Every non key attribute must provide a fact about the key, the whole key, and nothing but the key.**

William Kent (1936 – 2005)

William Kent. "A Simple Guide to Five Normal Forms in Relational Database Theory", Communications of the ACM 26 (2), Feb. 1983, pp. 120–125.

# More to Read

- A formal definition of NFs
  - Chapter 7, Relational Database Design, “Database System Concepts”

- The original research papers on normal forms
  - E. F. Codd. 1970. A relational model of data for large shared data banks. Commun. ACM 13, 6 (June 1970), 377–387. DOI:https://doi.org/10.1145/362384.362685
  - Codd, E. F.. “Further Normalization of the Data Base Relational Model.” Research Report / RJ / IBM / San Jose, California RJ909 (1971).
  - Armstrong, William Ward. “Dependency Structures of Data Base Relationships.” IFIP Congress (1974).