

# Database Management Systems (CSN-351)

## Overview and Entity Relationship Model

**BTech 3rd Year (CS) + Minor**

Instructor: **Ranita Biswas**

Department of Computer Science and Engineering

Indian Institute of Technology Roorkee

Roorkee, Uttarakhand - 247 667, India



# Database Management Systems (DBMS)

- Interrelated data about a particular enterprise
- Set of programs/applications to access the data
- Examples in real life:
  - Banking: transactions
  - Airlines: reservations, schedules
  - Universities: registration, grades
  - Sales: customers, products, purchases
  - Online retailers: order tracking, customized recommendations
  - Manufacturing: production, inventory, orders, supply chain
  - Human resources: employee records, salaries, tax deductions

# Why not File Systems?

- Data redundancy and inconsistency — Multiple file formats, duplication of information in different files.
- Difficulty in accessing data — Need to write a new program to carry out each new task.
- Data isolation — Multiple files and formats.
- Integrity problems — Integrity constraints are not stated explicitly.  
Hard to add new constraints or change existing ones.

# Why not File Systems? (contd.)

- Atomicity of updates — Failures may leave database in an inconsistent state with partial updates carried out.
- Concurrent access by multiple users — Uncontrolled concurrent accesses can lead to inconsistencies.
- Security problems — Hard to provide user access to some, but not all, data.

# Levels of Abstraction

- Physical level — describes how a record is stored.
- Logical level — describes data stored in database, and the relationships among the data.
- View level — application programs hide details of data types.

# Schema and Instance

- Logical Schema — the overall logical structure of the database
- Physical Schema — the overall physical structure of the database
- Instance — the actual content of the database at a particular point in time
- Physical Data Independence — the ability to modify the physical schema without changing the logical schema

# Data Models

- A collection of tools for describing Data, Data relationships, Data semantics, Data constraints
- Relational model
- Entity-Relationship data model (mainly for database design)
- Object-based data models (Object-oriented and Object-relational)
- Semistructured data model (XML)
- Other older models: Network model, Hierarchical model

# Modeling

- 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

|                       |                  |                          |
|-----------------------|------------------|--------------------------|
| 44553 (Peltier)       | <i>advisor</i>   | 22222 (Einstein)         |
| <i>student</i> entity | relationship set | <i>instructor</i> entity |

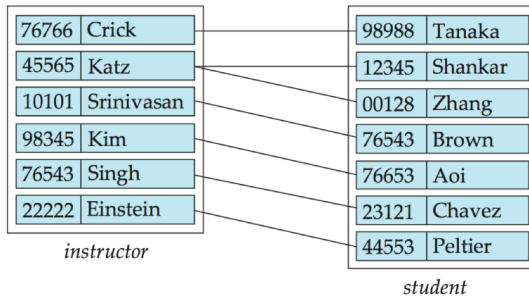
- A relationship set is a mathematical relation among  $n \geq 2$  entities, each taken from entity sets

$$\{(e_1, e_2, \dots, e_n) | e_1 \in E_1, e_2 \in E_2, \dots, e_n \in E_n\}$$

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

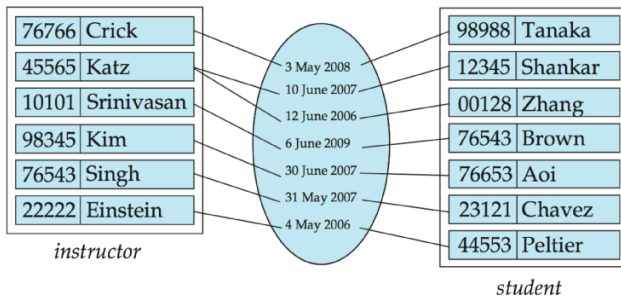
Example:  $(44553, 22222) \in \text{advisor}$

# Relationship Set: *advisor*



# Relationship Sets (contd.)

- An *attribute* can also be property of a relationship set.

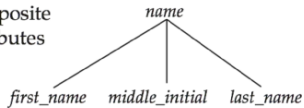


# Attributes

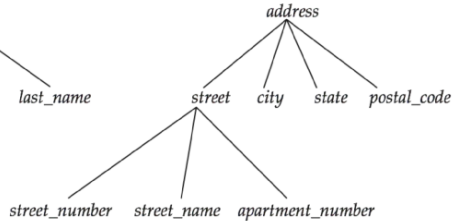
- An entity is represented by a set of attributes, that is descriptive properties possessed by all members of an entity set.
- *Domain* — the set of permitted values for each attribute
- Attribute types: *Simple* and *composite* attributes, *Single-valued* and *multivalued* attributes, *Derived* attributes

# Composite Attributes

composite  
attributes



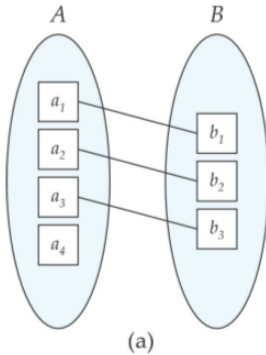
component  
attributes



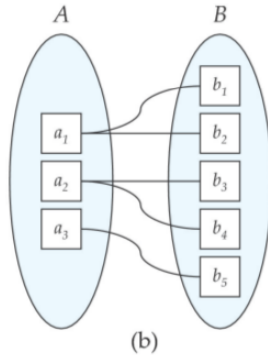
# Mapping Cardinality Constraints

- 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



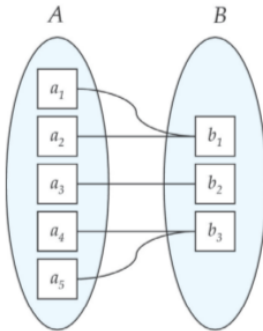
One to one



One to many

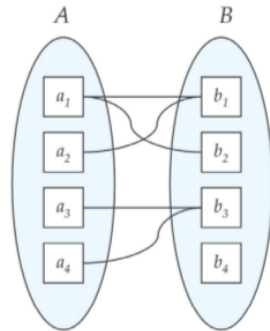


# Mapping Cardinalities (contd.)



(a)

Many to  
one



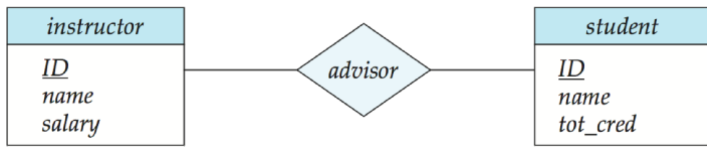
(b)

Many to many

# Keys

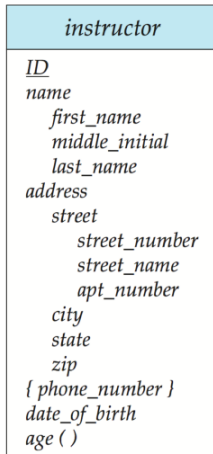
- A *super key* of an entity set is a set of one or more attributes whose values uniquely determine each entity.
- A *candidate key* of an entity set is a minimal super key.
- Although several candidate keys may exist, one of the candidate keys is selected to be the *primary key*.
- *Keys for Relationship Sets* — The combination of primary keys of the participating entity sets forms a super key of a relationship set.

# E-R Diagrams

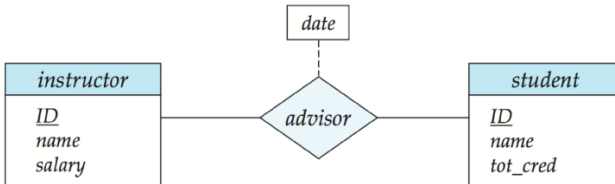


- Rectangles represent entity sets
- Diamonds represent relationship sets
- Attributes listed inside entity rectangle
- Underline indicates primary key attributes

# Entity With Composite, Multivalued, and Derived Attributes

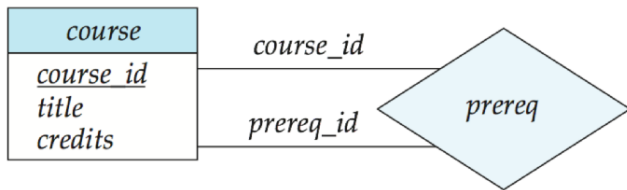


# Relationship Sets with Attributes



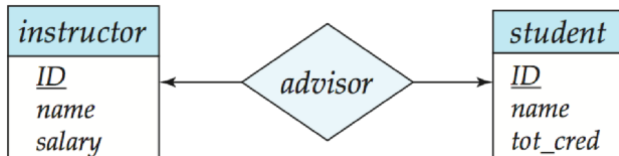
# Roles

- 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



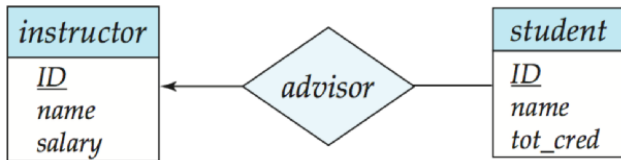
# One-to-One Relationship

- An instructor is associated with at most one student via *advisor*
- A student is associated with at most one instructor via *advisor*



# One-to-Many Relationship

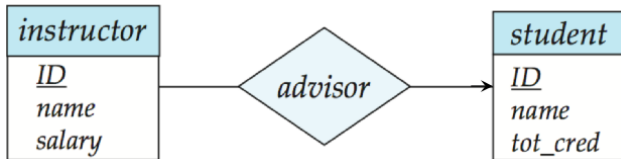
- An instructor is associated with several (including 0) students via *advisor*
- A student is associated with at most one instructor via *advisor*





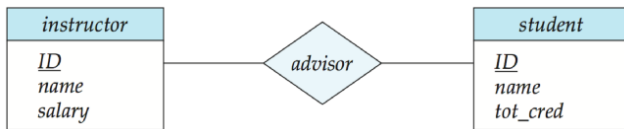
# Many-to-One Relationship

- An instructor is associated with at most one student via *advisor*
- A student is associated with several (including 0) instructors via *advisor*



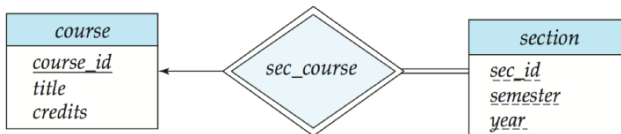
# Many-to-Many Relationship

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



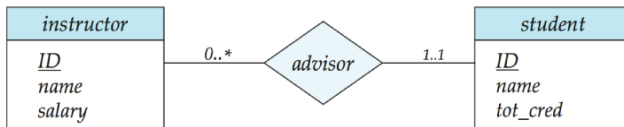
# Participation of an Entity Set in a Relationship Set

- *Total participation* (indicated by double line): every entity in the entity set participates in at least one relationship in the relationship set
- *Partial participation*: some entities may not participate in any relationship in the relationship set



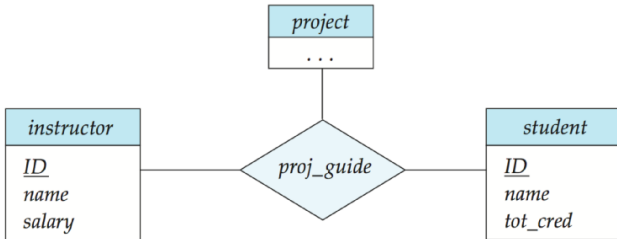
# Alternative Notation for Cardinality Limits

- Cardinality limits can also express participation constraints.



Each instructor advises 0 or more students and each student is advised by exactly 1 instructor.

# E-R Diagram with a Ternary Relationship



# Cardinality Constraints on Ternary Relationship

- At most one arrow is allowed out of a ternary (or greater degree) relationship to indicate a cardinality constraint.
- If there is more than one arrow, there are two ways of defining the meaning.