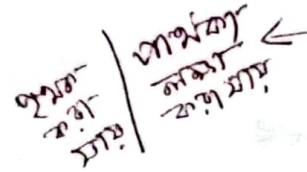
* Database model is of (2) types :

1) Collection of entities
2) Relationship among entities

* Entity : An entity is an (object) that exists and is distinguishable from other objects. (person, company, event, etc)



* Entity sets : set of entitities of the
  - same type that share the
  - same properties

Instructor Entity sets

instructor ID	name
101	Raihan Ullah
102	Pantha Protim Paul

} Ex.

* Relationship: association among several entities

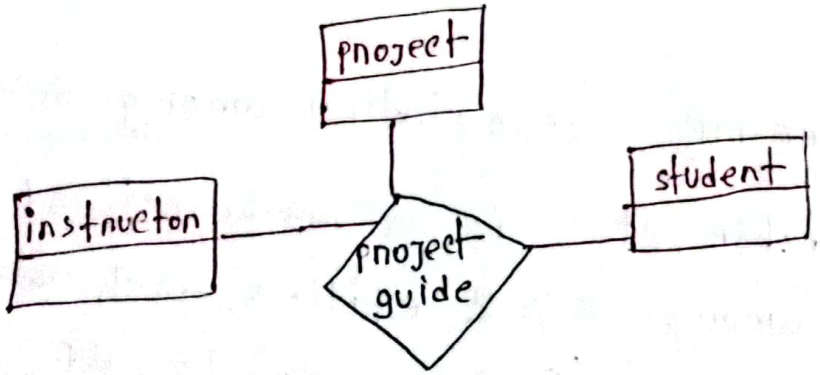
* Relationship sets: is a mathematical relation among $n \geq 2$ entities, each taken from entity sets $\{(e_1, e_2, \ldots e_n) \mid e_1 \in E_1 \ldots e_n \in E_n\}$ where $(e_1, e_2 \ldots e_n)$ is a relationship.

Degnee of nelationship set:
no. of entity sets involved in the nelationship.

Binany nelationship: 2 degnee
→ nelationships betwen 2 entity sets.

Tennenу nelationship: 3 degnee

Thene ane 3 entity sets involved in the nelationship, and each entity in one entity set can be associated with one on mone entities in the othen 2 entity sets thnough tennenу nelationship.



4 types: 1) one to one ↔
2) one to many ←
3) many to one →
4) many to many —

Mapping cardinality of relationship sets:

— refers to the number of entities in one entity set that can be associated with a specific no. of entities in another entity set through a relationship set.
/via

Semantics of a relationship sets:

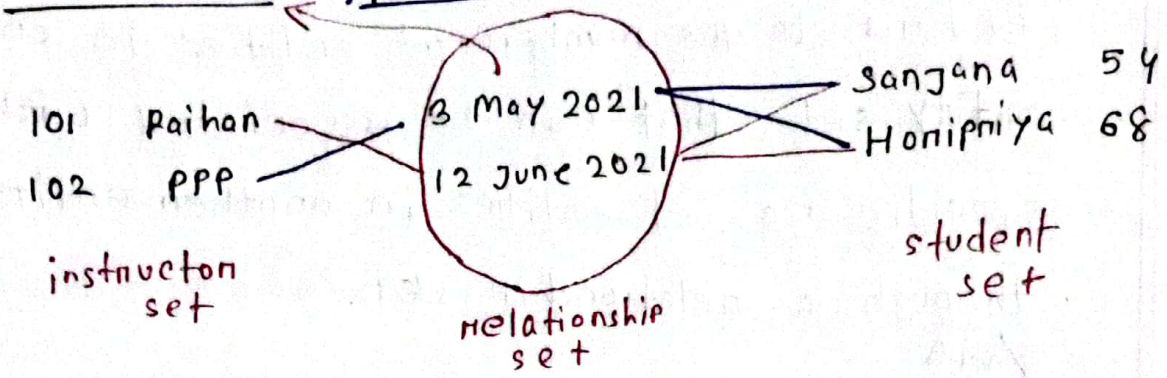
- describes the nature of the association between the entities in the entity sets.

Here, manages — semantics

"employee" and "project" entity set has mapping cardinality (0, N), which means an employee can manage 0 or more projects.

Relationship sets with attributes.

* Attribute: property of a relationship set



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.

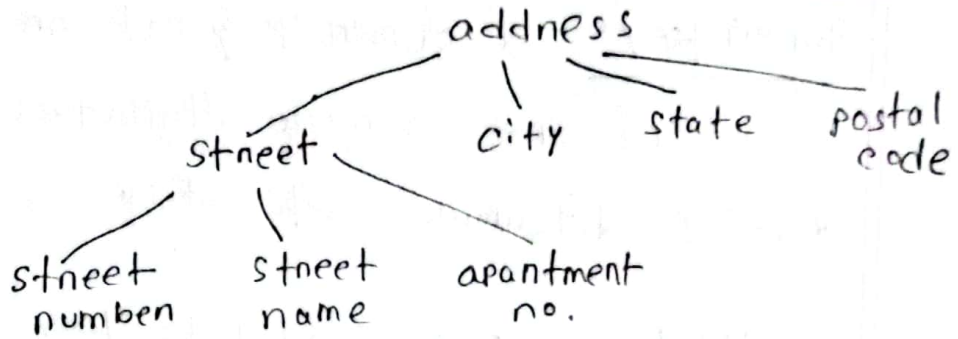
Types of attribute:

1) simple and composite attributes

simple → can't be divided into subparts, single piece of info about an entity

composite → can be divided into smallen subparts, each of which represents a distinct piece of info. about an entity.

Ex:



2) Single-valued and multivalued attributes

Single-valued
- have only 1 value for the given entity
- Ex: Age

multivalued
- can have multiple values for the given entity
- ex: mobile no.
  - can have multiple phone numbers

3) Derived attributes:

Unlike regular attributes, a derived attribute is not directly stored in the database but is calculated based on other attributes at the time of a query on retrieval.

Ex: (age), calculated based on date of birth

Redundant attributes:

can be derived, still present in DB, leading problems like - higher storage requirements, slower query performance, data inconsistencies

Ex: 1st and last name stored, (name) is then redundant as it's derived from 1st and last

Keys

Super key: a super key of an entity set is a set of one on more attributes whose values uniquely determine each entity.

Candidate key: A candidate key is a minimal super key without any redundant attributes.

Primary key: is the selected candidate key that is used to uniquely identify each entity in the entity set.

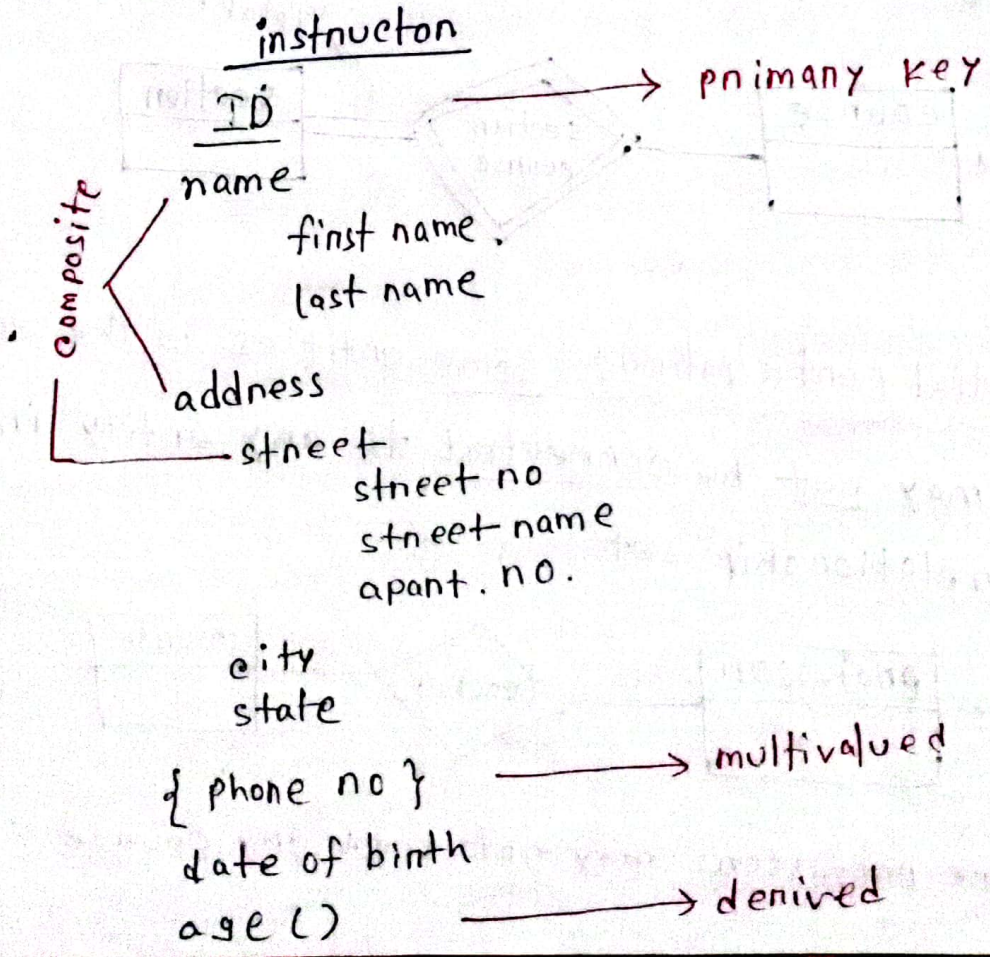
Ex: 'student ID' can alone be a super key because each student has a unique ID.
Name + Email can also be a super key because this combination of attributes can uniquely identify each student.

Name + Email can't be a candidate key because it's not minimal as it contains 2 attributes, instead of 1.

student ID is a candidate key because it's the smallest combination of attnibutes to uniquely identify each student.

student ID → primany key, as it's simplest and the most efficient.

* Entity with primany key, composite, multivalued and denived attnibutes:

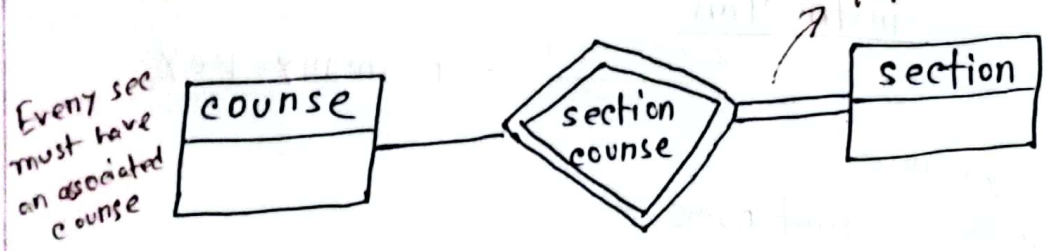


* Participation of an entity set in a relationship set:

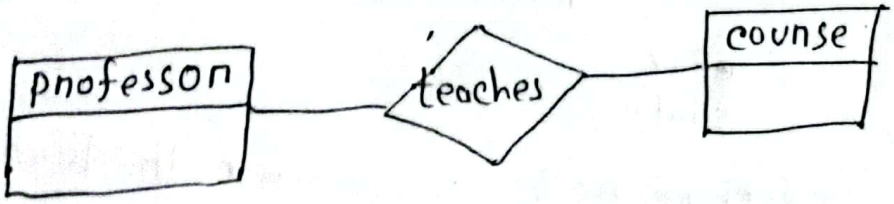
- refers to whether each entity in the entity set must participate in the relationship set or not

2 types:

1) Total participation: Every entity in the entity set must be connected to at least one entity in the relationship set.



2) Partial participation: Some entities in the entity set may not be connected to any entity in the relationship set.



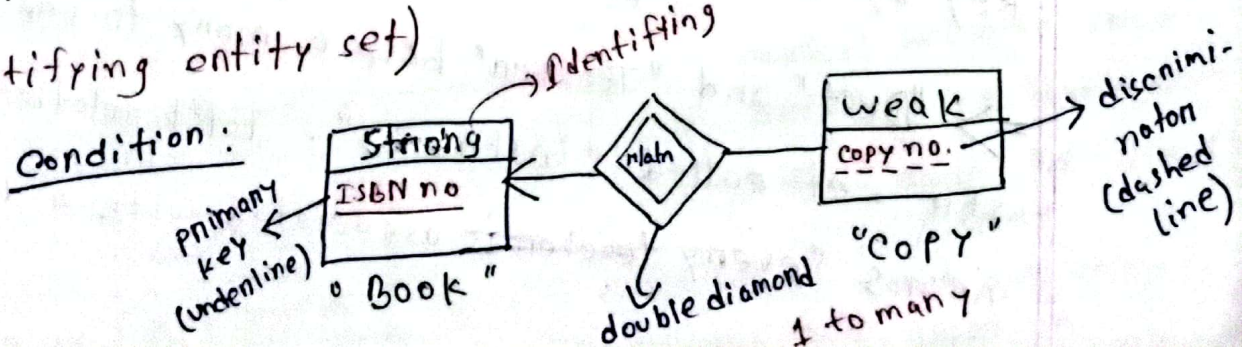
some professons may not teach any counse.

* Weak entity set: An entity set that doesn't have a primary key is referred to as a weak entity set.

→ "The existence of a weak entity set depends on the existence of an identifying entity set" means that a weak entity set can only exist if it's associated with an identifying entity set.

For example, "Book" entity set is identified by its unique ISBN number, but "copy" entity set can't be uniquely identified by its attributes alone because multiple copies of the same book can exist. So, each copy should be identified by combination of "copy number" and the "ISBN number" (primary key of "Book"). So, "copy" (weak entity set)'s existence depends on the existence of "Book" (the identifying entity set)

Condition:



-* primary key (weak entity) = primary key (strong) + discrimination (weak)

book's ISBN | "copy" — copy no.

* The discrimination of a weak entity set is a set of one or more attributes that, together with the primary key of the identifying entity set, uniquely identifies each entity in the weak entity set. Ex: copy number.

— Why called partial key?

→ (it) discrimination doesn't uniquely identify the entity on its own, it only identifies the entity in combination with the identifying entity set.

Redundancy of schemas and solutions:

1) If there is any "many to one" or "one to many" total relationship sets, the many-side can be represented by adding extra attribute (primary key of one-side) to the many-side.

=> "Dept" and "Teacher" have a many to one relationship set called "instructs". Total relationship means "every teacher is associated with a dept"

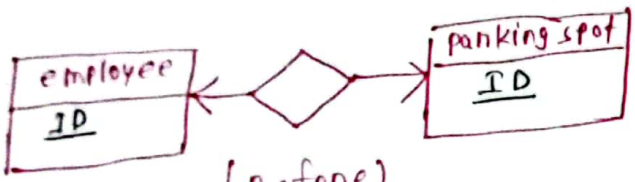
Now, add an extra attribute "dept_name" to the "teacher". So, relationship set "instructs" is not needed anymore.

(Lessen) → no. of relationship set

2) For 1 to 1 relation,

→ add extra attribute (primary key of one set) to the other entity set

→ less relationship



(Before)

(After)

3) many to one / one to many

→ many-side is partial

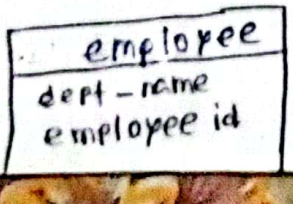
→ instead of adding extra attribute, add relationship. (more appropriate)

Ex: employee (many) → dept (one)

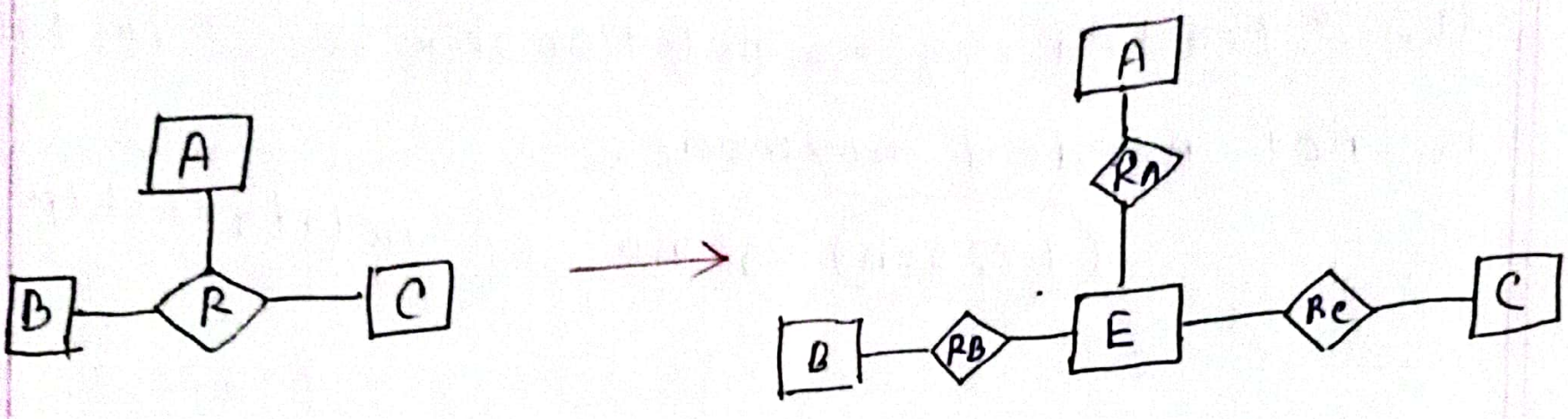
partial because some dept may not have any employee.

→ then dept id / name = null

→ so avoid null values, add relation



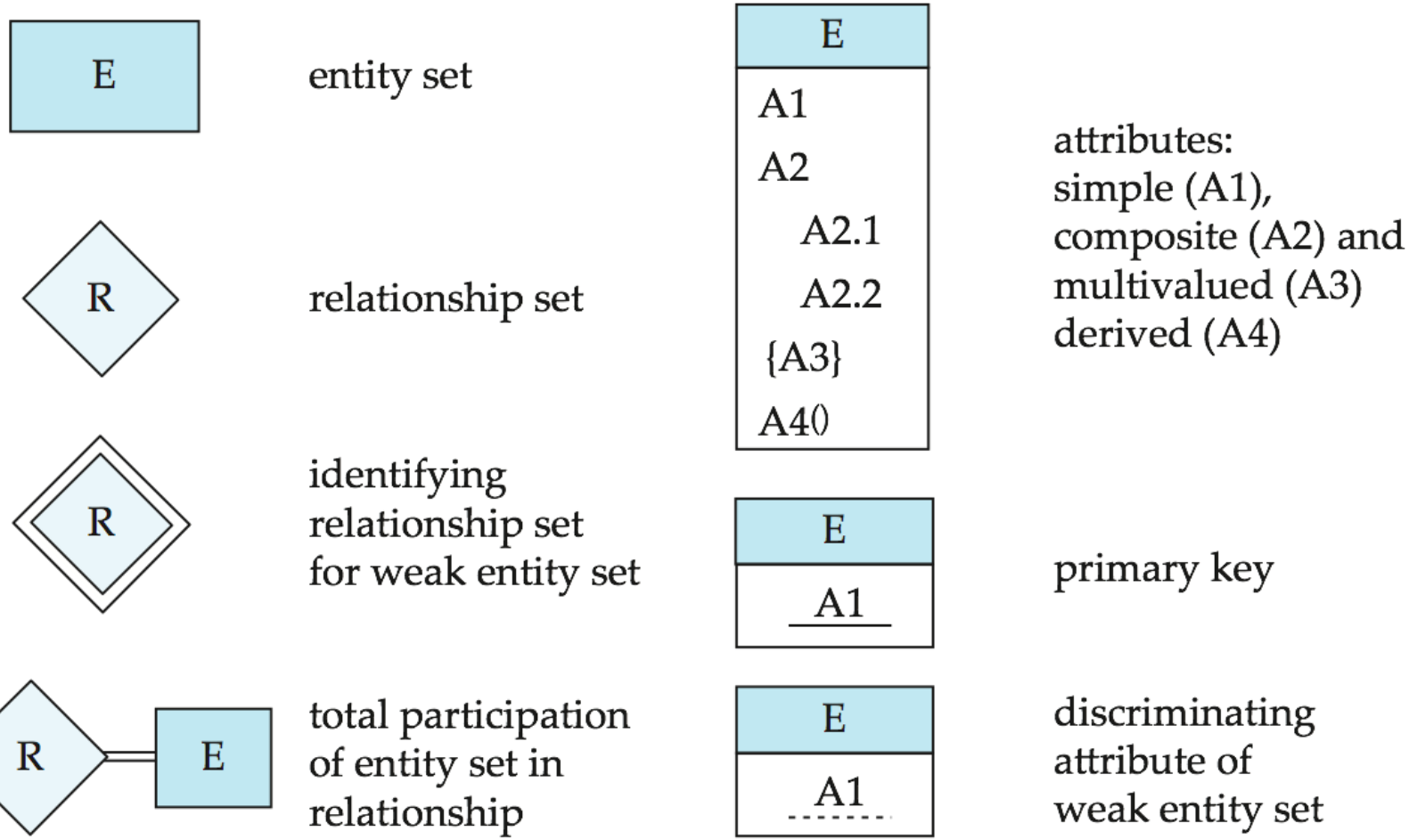
↳ Converting non-binary → binary relation



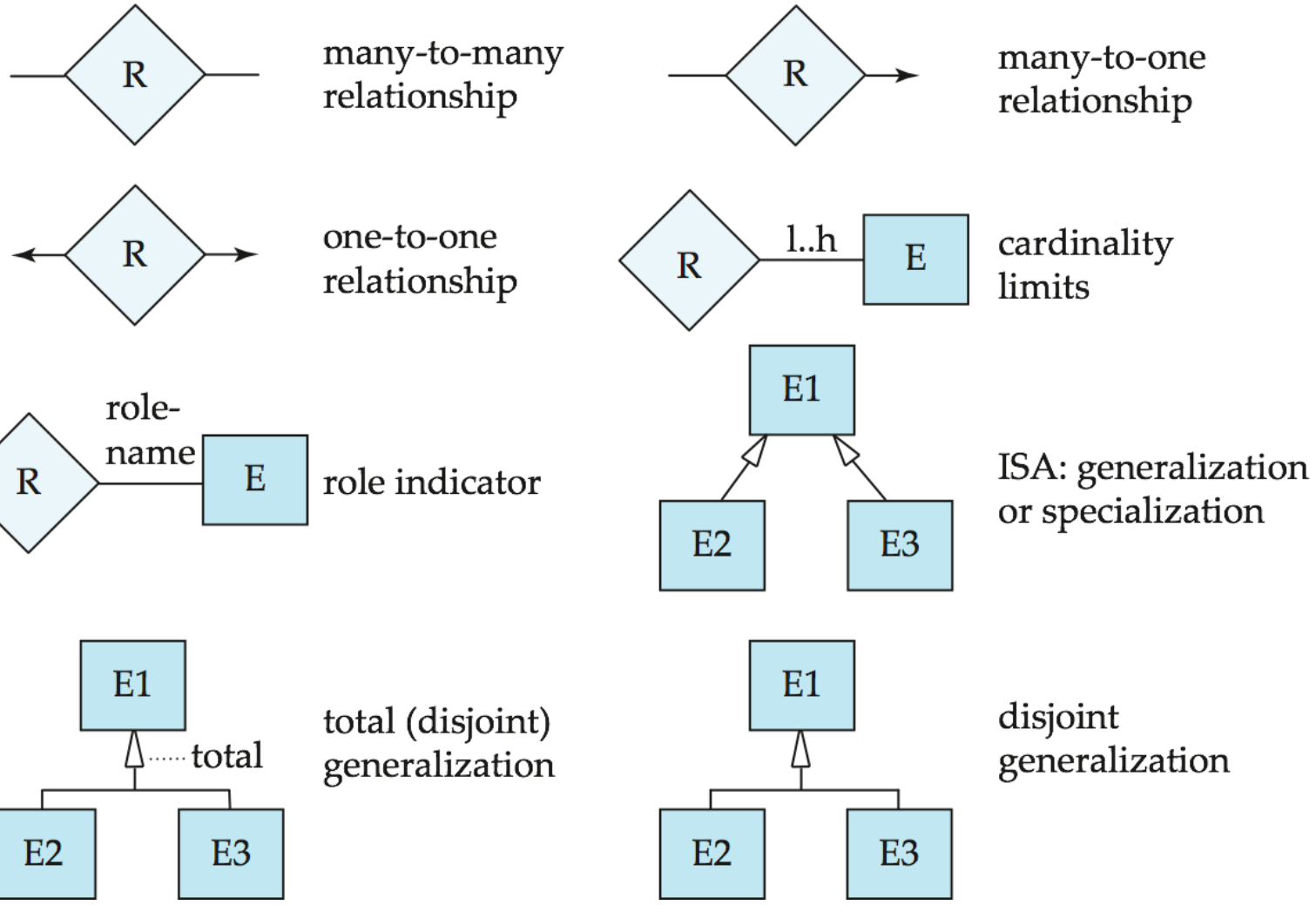
— add new entity [RA]

— new 3 relation

Summary of Symbols Used in E-R Notation



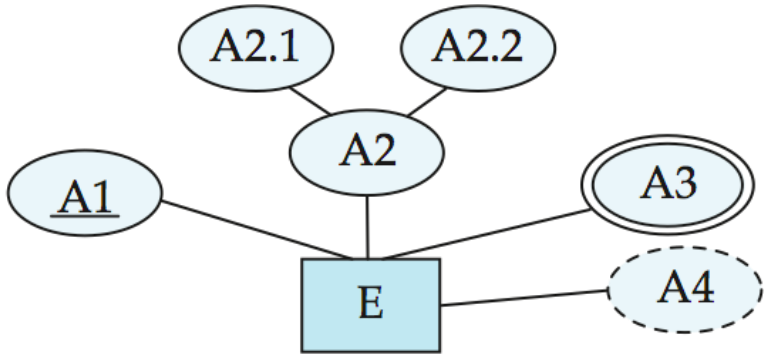
Symbols Used in E-R Notation (Cont.)



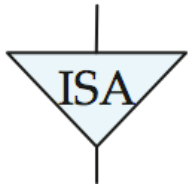
Alternative ER Notations

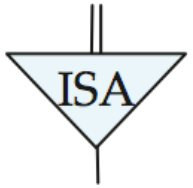
- Chen, IDE1FX, …

entity set E with
simple attribute A1,
composite attribute A2,
multivalued attribute A3,
derived attribute A4,
and primary key A1

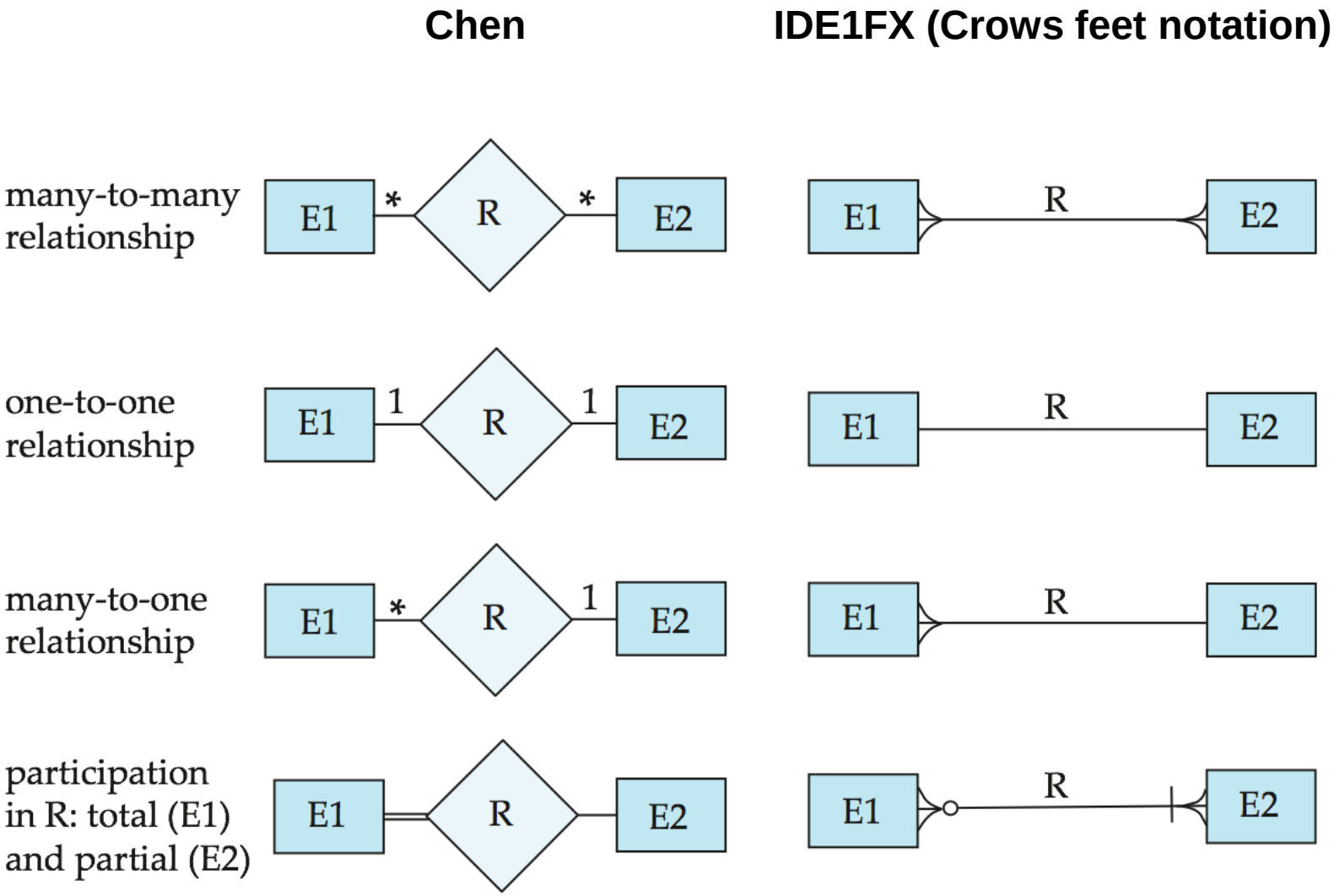


weak entity set 

generalization 

total generalization 

Alternative ER Notations

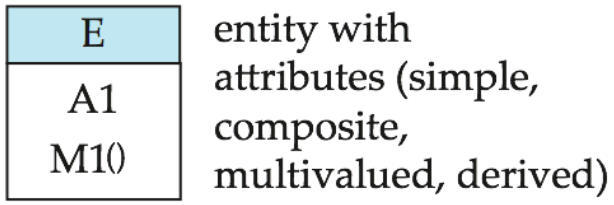


UML

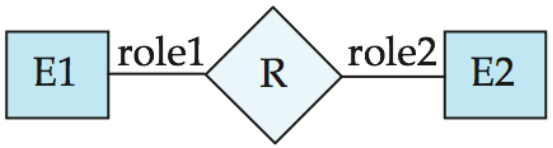
- **UML**: Unified Modeling Language
- UML has many components to graphically model different aspects of an entire software system
- UML Class Diagrams correspond to E-R Diagram, but several differences.

ER vs. UML Class Diagrams

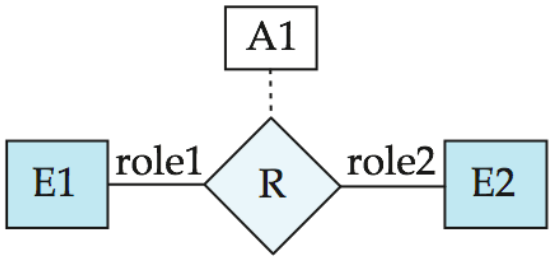
ER Diagram Notation



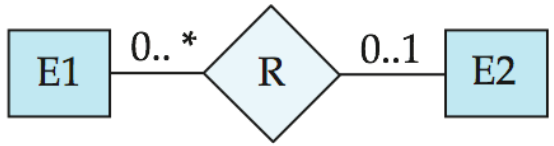
entity with attributes (simple, composite, multivalued, derived)



binary relationship

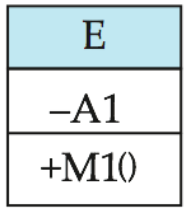


relationship attributes

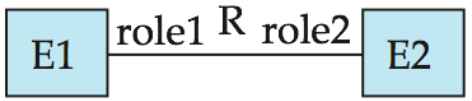


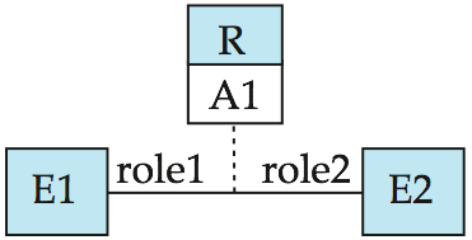
cardinality constraints

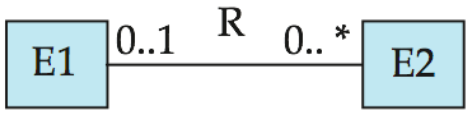
Equivalent in UML



class with simple attributes and methods (attribute prefixes: + = public, – = private, # = protected)



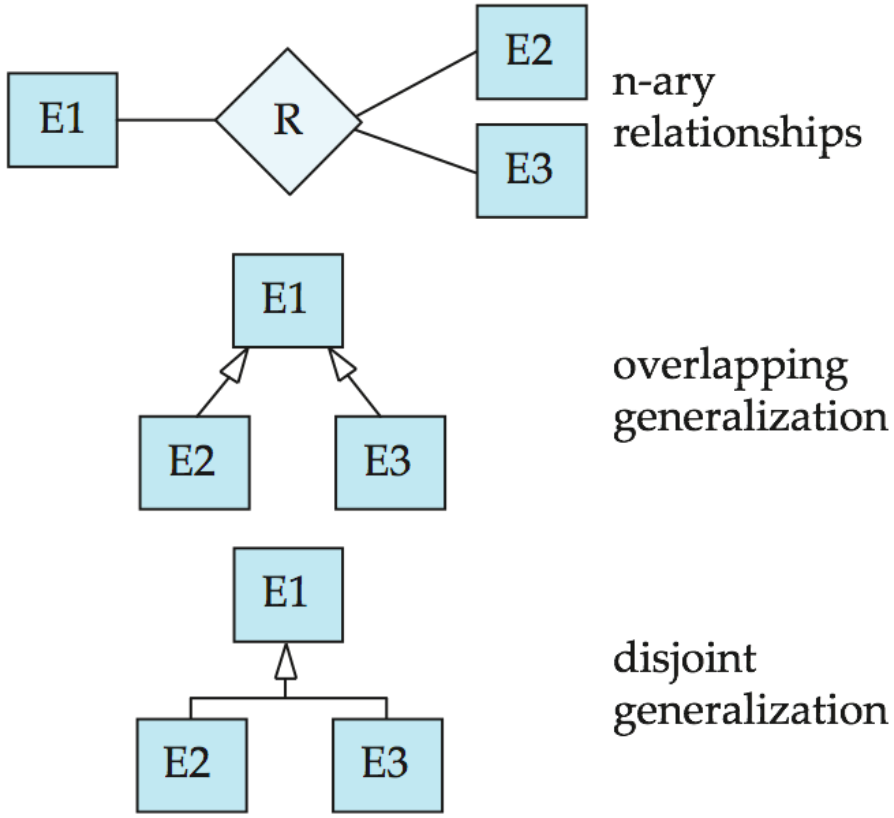




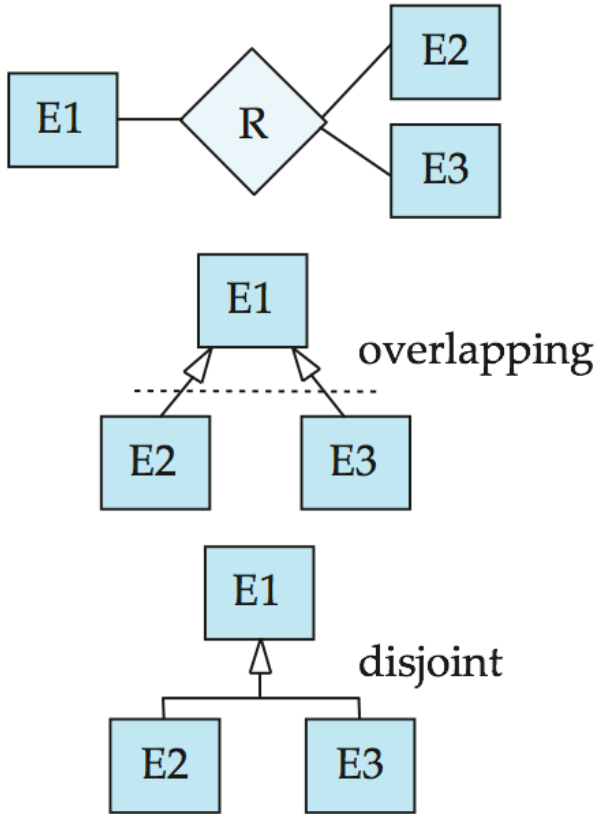
*Note reversal of position in cardinality constraint depiction

ER vs. UML Class Diagrams

ER Diagram Notation



Equivalent in UML



*Generalization can use merged or separate arrows independent of disjoint/overlapping

Super	Candidate	Primary
Uniquely identify an entity	Uniquely identify an entity	Uniquely identify an entity
Includes additional attributes that is not necessary for identifying	Doesn’t include	Doesn’t include
	Unique, not null	Unique, not null
Contains redundant attribute	Contains redundant attribute	Contains redundant attribute
There can be multiple super keys	There can be multiple candidate keys	There will be only 1 primary key
Not main	Not main	Main identifier of entity

Let's consider a table named "Employees" with the following columns:

EmployeeID (primary key)

FirstName

LastName

Email

PhoneNumber

DateOfBirth

In this example, "EmployeeID" is the primary key for the "Employees" table, which means that it uniquely identifies each row in the table. This column cannot contain null values, and it must be unique across all rows in the table.

However, there are other columns in the "Employees" table that could also serve as unique identifiers. For example, the combination of "Email" and "PhoneNumber" could also uniquely identify each employee. This combination of columns is a candidate key because it satisfies the requirements of a key (i.e., it is unique and not null), but it is not the chosen primary key for the table.

Finally, a superkey in the "Employees" table could be the combination of "EmployeeID", "FirstName", "LastName", and "DateOfBirth". This combination of columns uniquely identifies each row in the table, but it contains additional columns that are not necessary for uniqueness. Therefore, this combination of columns is a superkey but not a candidate key, as it includes more information than needed to identify each row.