Stony Brook University

CSE 305 / CSE532

Lecture 04 (Chapter 4)

Conceptual Modeling of Databases with E-R Model and UML

Lecturer: Sael Lee

Slide adapted from the author's and Dr. Ilchul Yoon's slides.

SUNY Korea

Database Design

- Goal: specification of database schema
- Methodology
  - Use **E-R model** to get a high-level graphical view of essential components of enterprise and how they are related
  - Convert E-R diagram to DDL – use rules

- **E-R** Model: enterprise is viewed as a set of
  - Entities
  - Relationships among entities

Entities

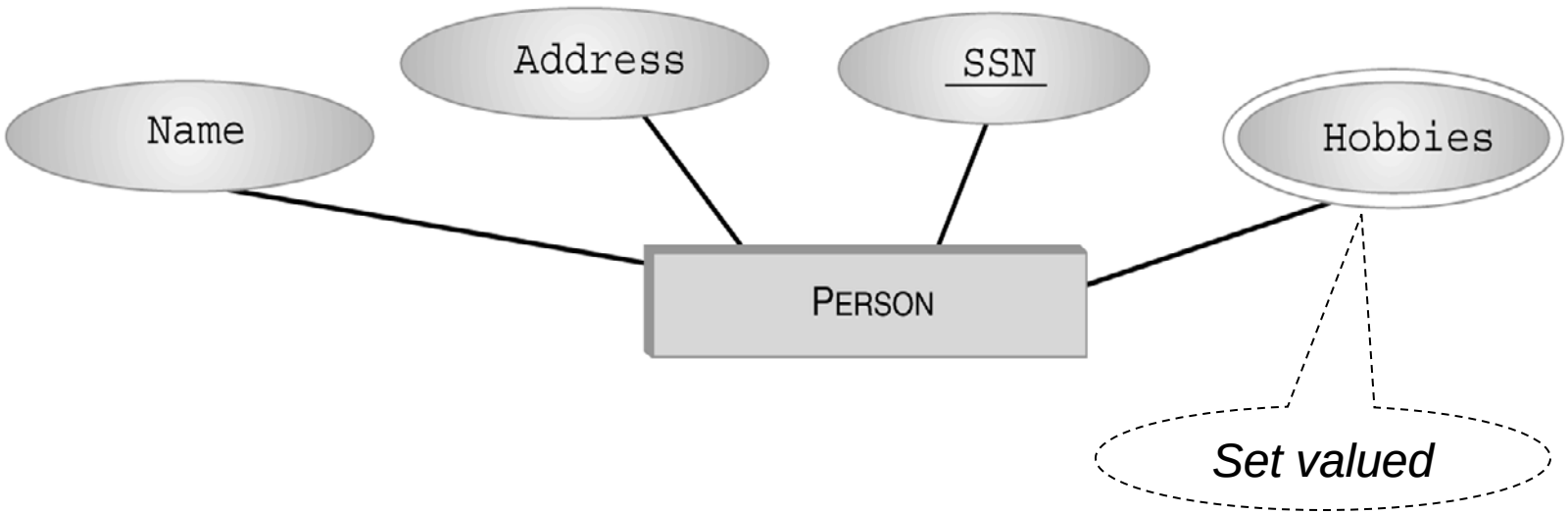
- Entity (instance): an object involved in the enterprise
  - Ex: John, CSE532
- Entity Type: set of similar objects
  - Ex: students, courses
- Attribute: describes one aspect of an entity type
  - Ex: name, maximum enrollment

Entity Type

- Entity type described by set of attributes
  - Person: Id, Name, Address, Hobbies
- **Domain**: possible values of an attribute
  - Value can be a set (in contrast to relational model)
    - (111111, John, 123 Main St, {stamps, coins})
- **Key**: minimum set of attributes that <u>uniquely identifies</u> an entity
- Entity Schema: entity type name, attributes (and associated domain), key constraints

Entity Type (con't)

- Graphical Representation in E-R diagram:



Relationships

- Relationship: relates two or more entities
  - John majors in Computer Science
- Relationship Type: set of similar relationships
  - Student (entity type) related to Department (entity type) by MajorsIn (relationship type).
- Distinction:
  - relation (relational model) - set of tuples
  - **relationship** (E-R Model) – describes relationship between entities of an enterprise
  - Both entity types and relationship types (E-R model) may be represented as relations (in the relational model)

Attributes and Roles

- **Attribute** of a relationship type describes the relationship
  - e.g., John majors in CS since 2000
    - John and CS are related
    - 2000 describes relationship - value of SINCE attribute of MajorsIn relationship type
- **Role** of a relationship type names one of the related entities
  - e.g., John is value of Student role, CS value of Department role of MajorsIn relationship type
  - (John, CS; 2000) describes a relationship

SUNY Korea

Relationship Type

- Described by set of attributes and roles
  - e.g., MajorsIn: Student, Department, Since
  - Here we have used as the role name (Student) the name of the entity type (Student) of the participant in the relationship, but …

SUNY Korea

Roles

- Problem: relationship can relate elements of same entity type
  - e.g., ReportsTo relationship type relates two elements of Employee entity type:
    - Bob reports to Mary since 2000
  - We do not have distinct names for the roles
  - It is not clear who reports to whom

SUNY Korea

Roles (con't)

- Solution: role name of relationship type need not be same as name of entity type from which participants are drawn
  - ReportsTo has roles *Subordinate* and *Supervisor* and attribute *Since*
  - Values of *Subordinate* and *Supervisor* both drawn from entity type Employee

SUNY Korea

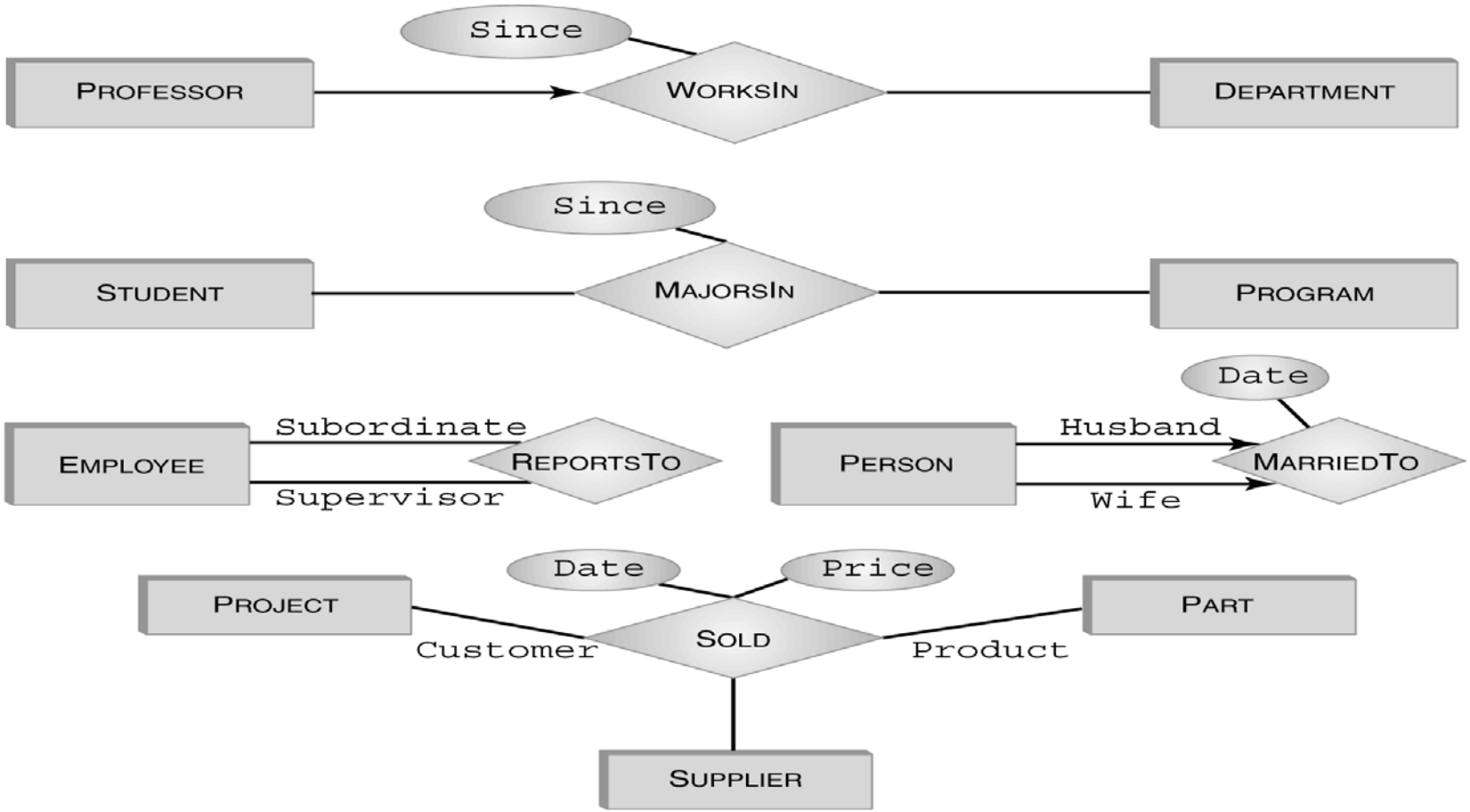
Schema of a Relationship Type

- Role names, R_i, and their corresponding entity sets
  - Roles must be single valued
  - number of roles = degree of relationship
- Attribute names, A_j, and their corresponding domains
  - Attributes may be set valued
- Key: Minimum set of roles and attributes that <u>uniquely identify a relationship</u>

- Relationship: $<e_1, \ldots e_n; a_1, \ldots a_k>$
  - e_i is an entity, a value from R_i's entity set
  - a_j is a set of attribute values with elements from domain of A_j

Graphical Representation

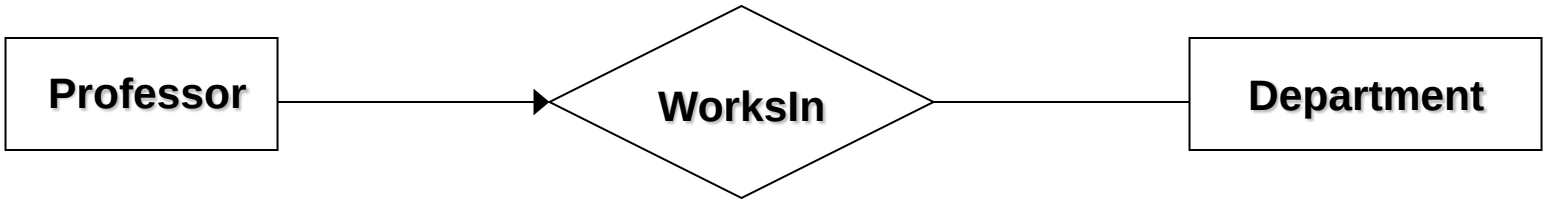
- Roles are edges labeled with role names
  - Omitted if role name = name of entity set
  - Most attributes have been omitted.



Single-role Key Constraint

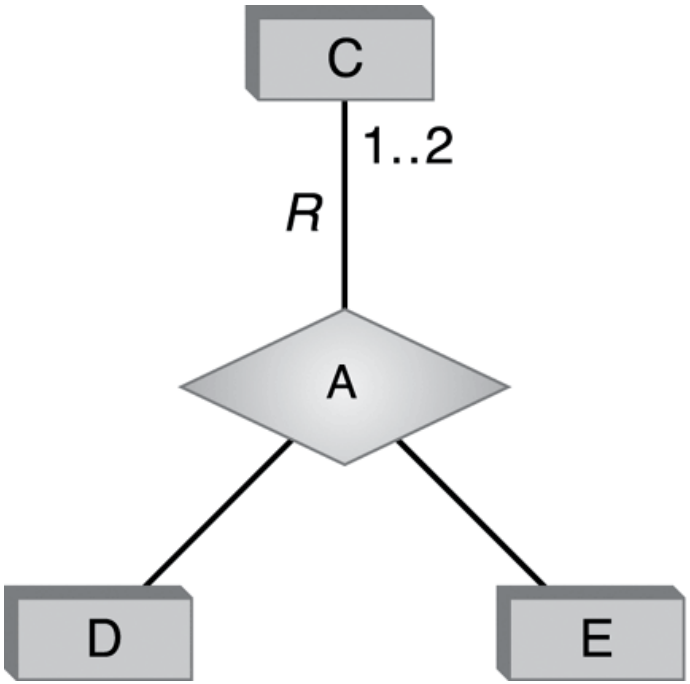
- If, for a particular participant entity type, <u>each entity</u> (instance) participates in at most one relationship (instance), corresponding role is a key of relationship type
  - E.g., Professor role is unique in WorksIn

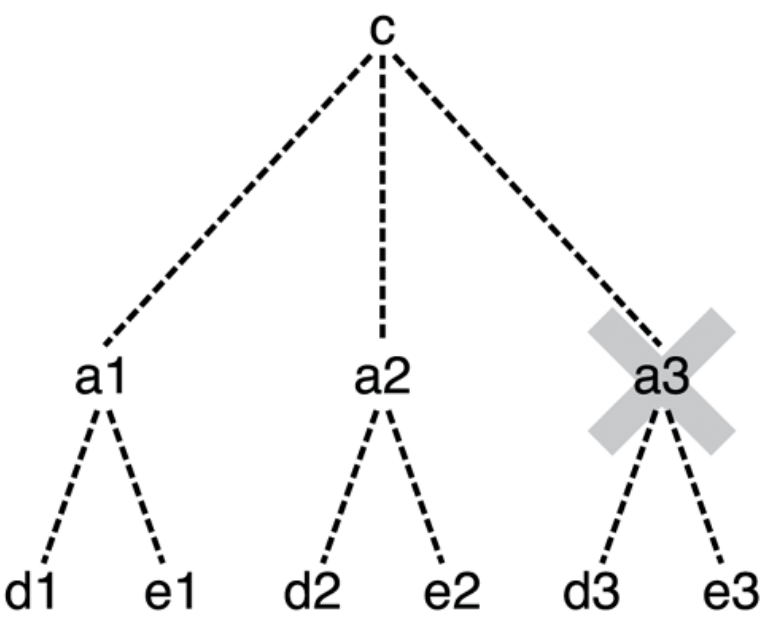
- Representation in E-R diagram: arrow



Cardinality Constraints

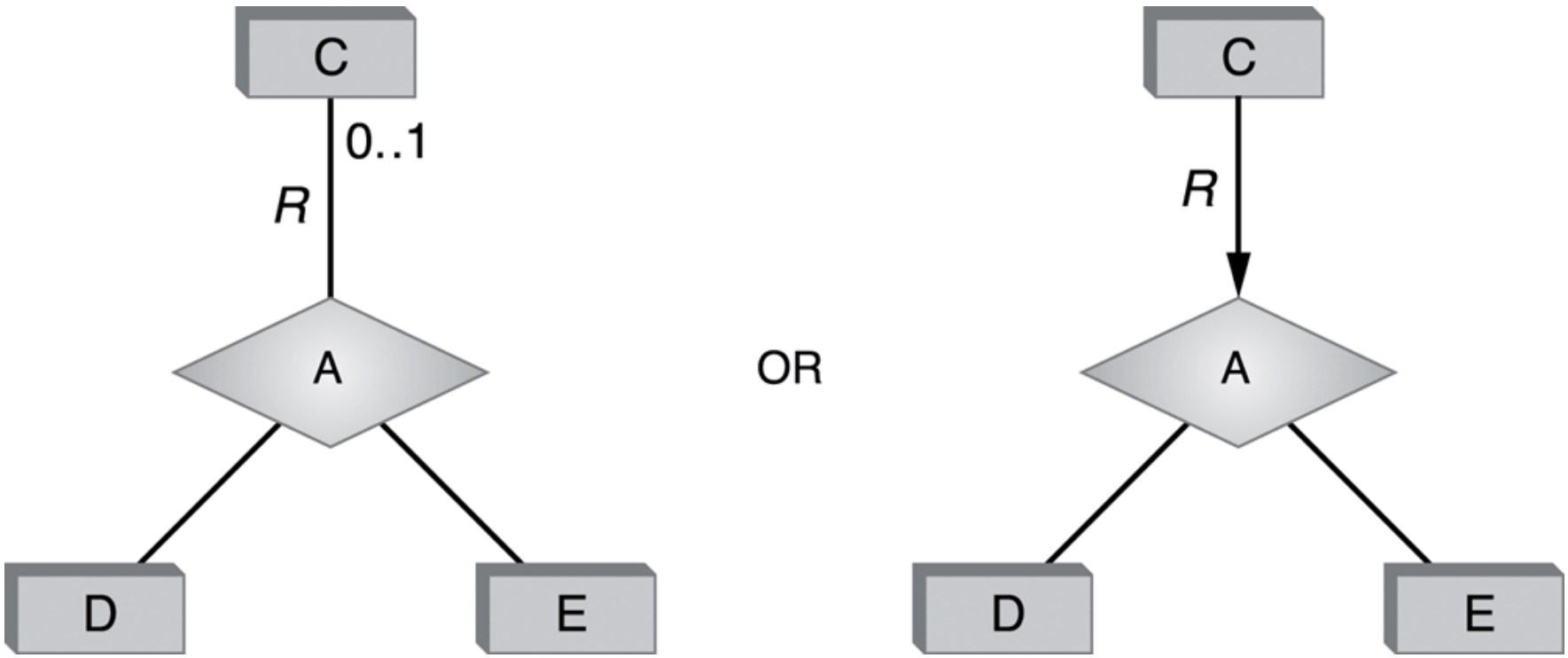
- Defined on role and restricts the number of relationship instances in which an instance of connected entity type can participate.



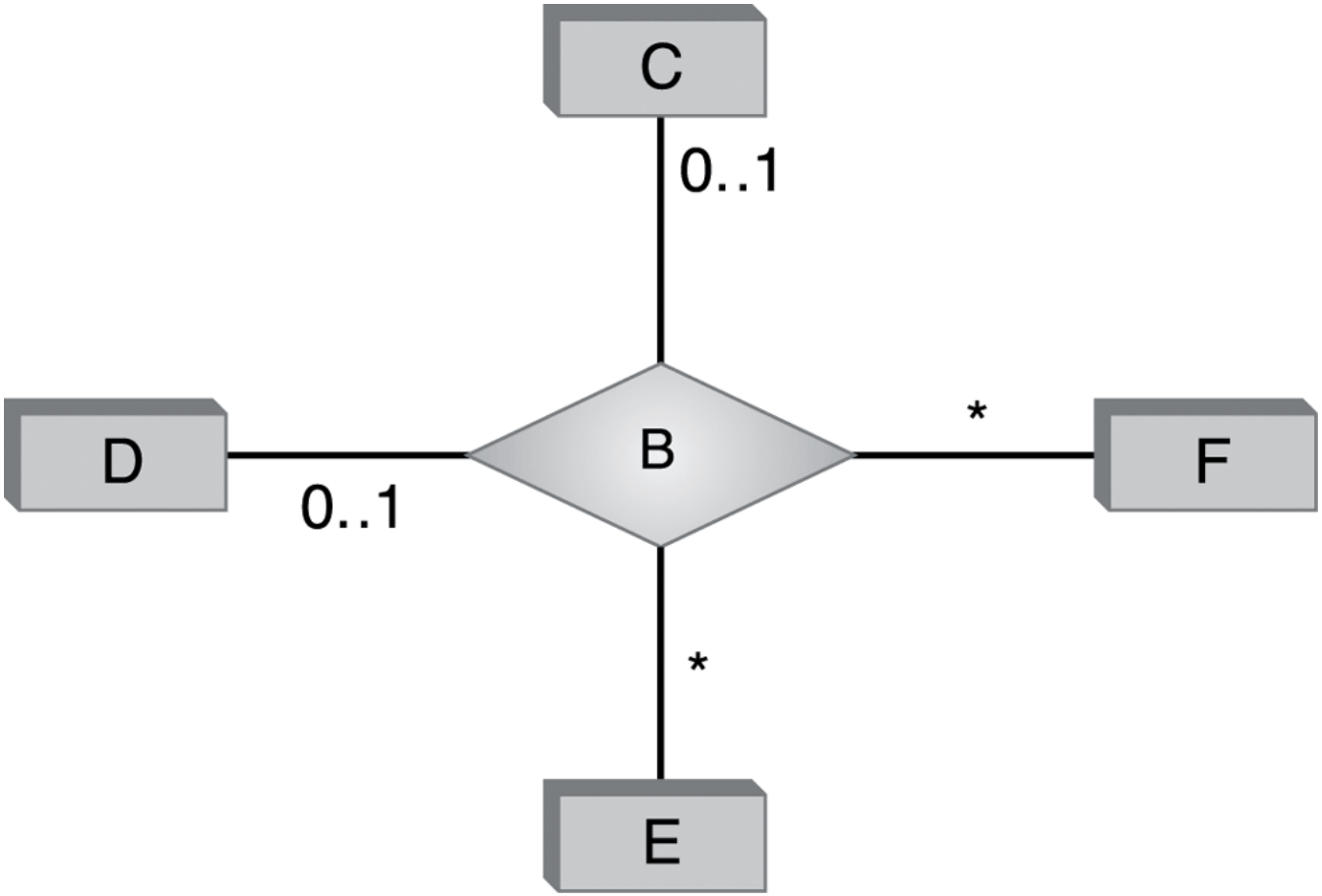


Cardinality Constraints and Single-Role Key Constraints

- Two ways of representation



Many-to-One, One-to-One, and Many-to-Many Correspondence

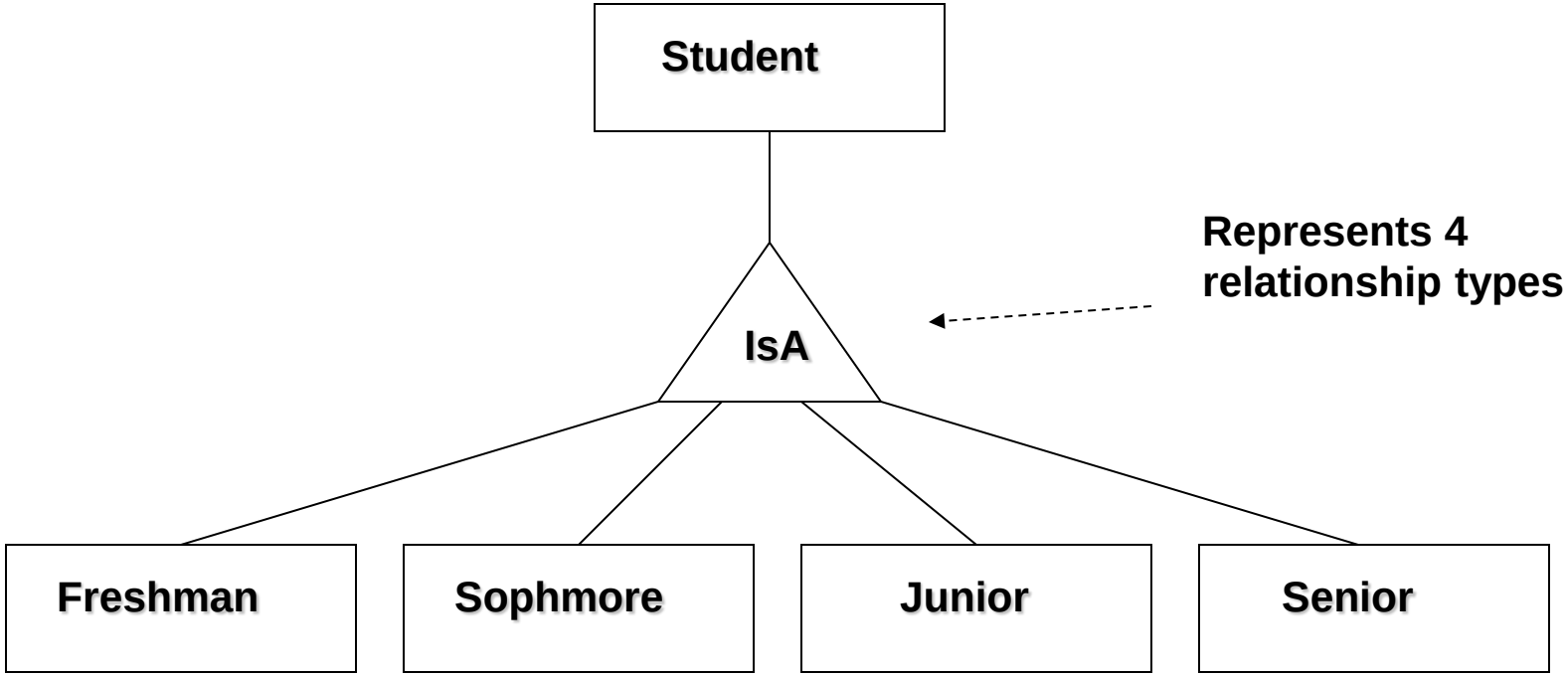


Entity Type Hierarchies

- One entity type might be subtype of another
  - Freshman is a subtype of Student
- A relationship exists between a Freshman entity and the corresponding Student entity
  - e.g., Freshman John is related to Student John
- This relationship is called IsA
  - Freshman IsA Student
  - The two entities related by IsA are always descriptions of the same real-world object

SUNY Korea

IsA


Student
IsA
Represents 4 relationship types
Freshman
Sophmore
Junior
Senior

Properties of IsA

- Inheritance - Attributes of supertype apply to subtype.
  - E.g., GPA attribute of Student applies to Freshman
  - Subtype inherits all attributes of supertype.
  - Key of supertype is key of subtype
- Transitivity - Hierarchy of IsA
  - Student is subtype of Person, Freshman is subtype of Student, so Freshman is also a subtype of Person

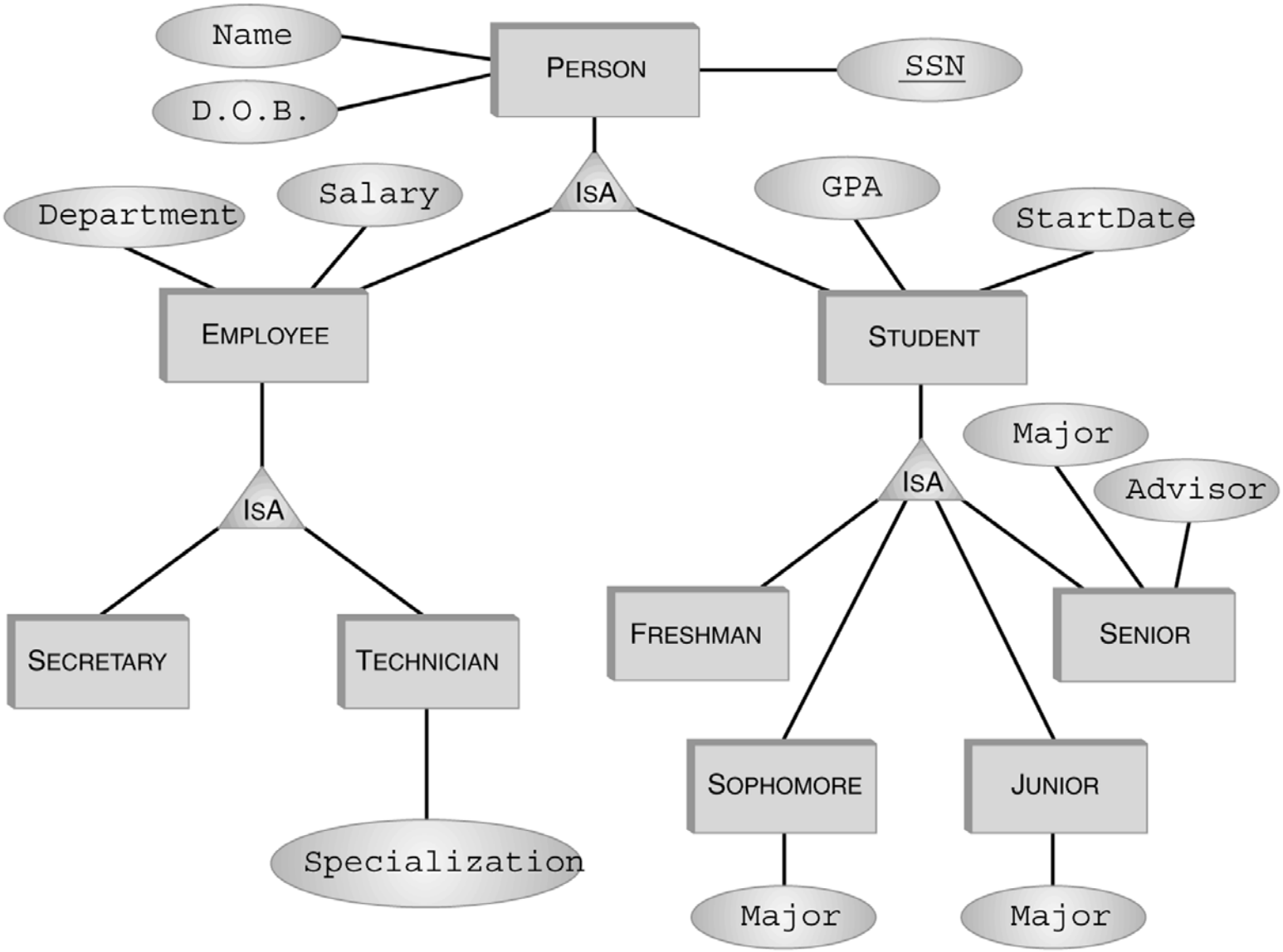
SUNY Korea

Advantages of IsA

- Can create a more concise and readable E-R diagram
  - Attributes common to different entity sets need not be repeated
  - They can be grouped in one place as attributes of supertype
  - Attributes of (sibling) subtypes can be different

SUNY Korea

IsA Hierarchy - Example



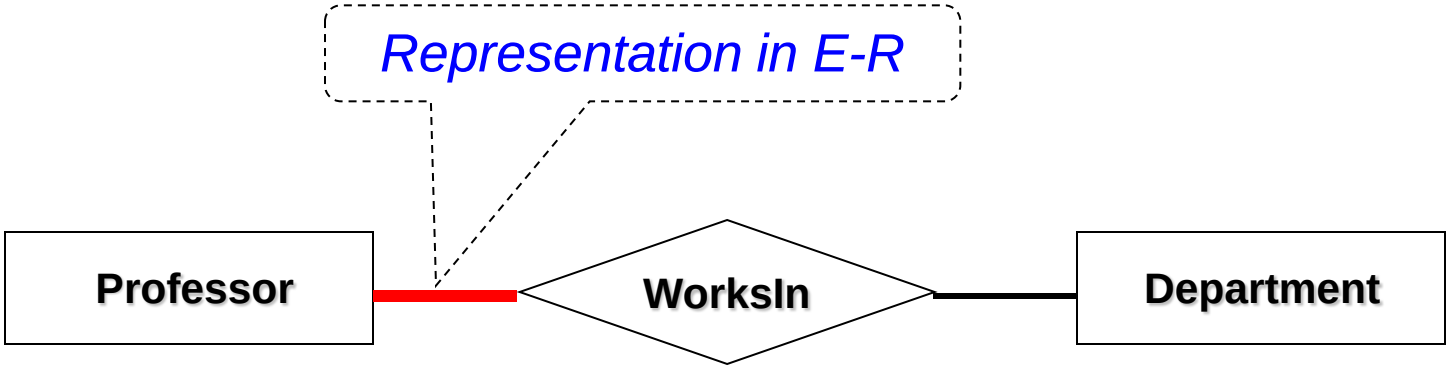
Constraints on Type Hierarchies

- Might have associated constraints:
  - **Covering** constraint: Union of subtype entities is equal to set of supertype entities
    - Employee is either a secretary or a technician (or both)
  - **Disjointness** constraint: Sets of subtype entities are disjoint from one another
    - Freshman, Sophomore, Junior, Senior are disjoint set

SUNY Korea

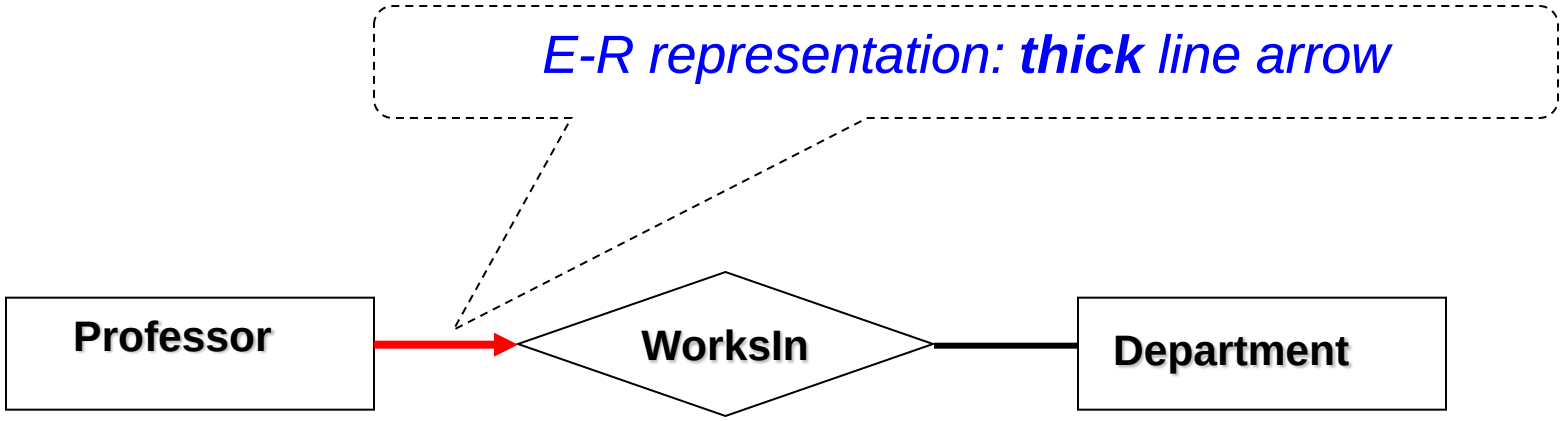
Participation Constraint

- If every entity participates in at least one relationship, a **participation constraint** holds:
  - A participation constraint of entity type E having role ρ in relationship type R states that for e in E there is an r in R such that ρ(r) = e.
  - e.g., every professor works in at least one department



Participation and Key Constraint

- If every entity participates in exactly one relationship, both a **participation** (at least one) and a **key constraint** (at most one) hold:
  - e.g., every professor works in exactly one department



Representing Entity Types in Relational Model

- An entity type corresponds to a relation
- Relation's attributes = entity type's attributes
  - Problem: entity type can have set valued attributes, e.g.,
  - Person: Id, Name, Address, Hobbies
  - Solution: Use several rows to represent a single entity
    - (111111, John, 123 Main St, stamps)
    - (111111, John, 123 Main St, coins)
  - Problems with this solution:
    - Redundancy
    - Key of entity type (Id) not key of relation
    - Hence, the resulting relation must be further transformed (Chapter 6)

SUNY Korea

Representing Relationship Types in Relational Model

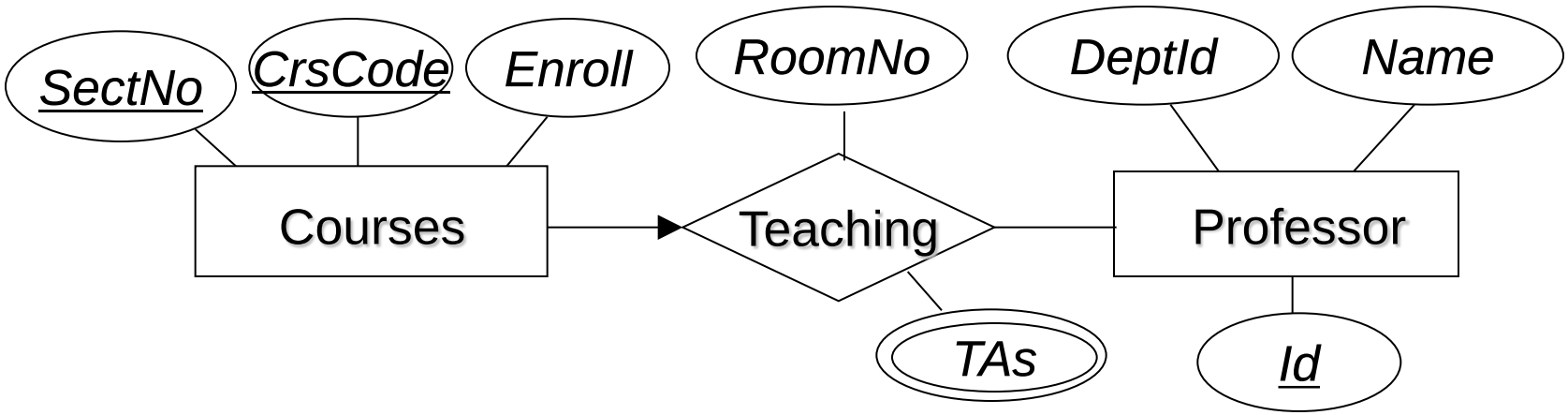
- Typically, a relationship becomes a relation in the relational model

- Attributes of the corresponding relation are
  - Attributes of relationship type
  - For each role, the primary key of the entity type associated with that role are also included

SUNY Korea

Representing Relationship Types in Relational Model (cont'd)

- Example:



- Courses (CrsCode, SectNo, Enroll)
- Professor (Id, DeptId, Name)
- Teaching (CrsCode, SecNo, Id, RoomNo, TAs)

Not in the key

Representing Relationship Types in Relational Model (cont'd)

- Candidate key of corresponding table = candidate key of relationship
  - Except when there are set valued attributes
  - Example: Teaching (<u>CrsCode, SectNo</u>, Id, RoomNo, TAs)
    - Key of relationship type = (CrsCode, SectNo)
    - Key of relation = (CrsCode, SectNo, TAs)

Set valued

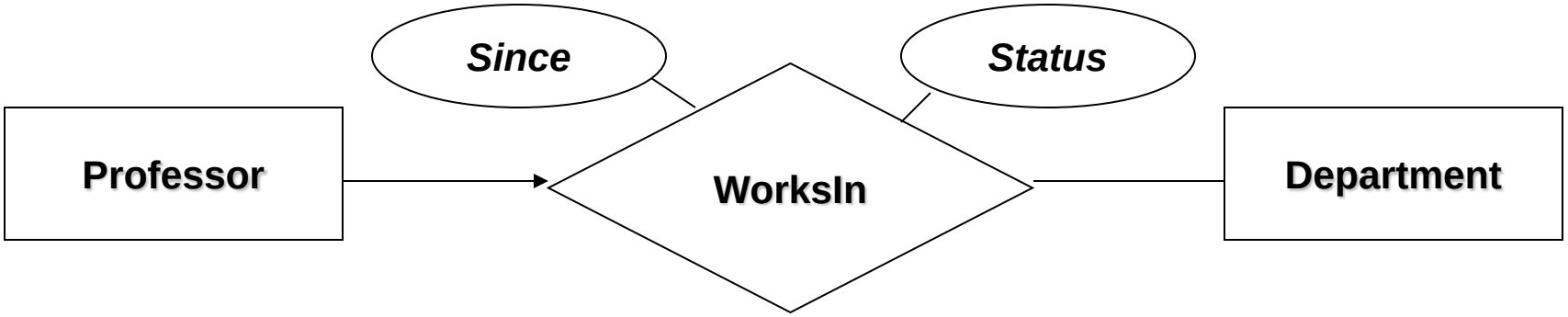
CrsCode	*SectNo*	*Id*	*RoomNo*	*TAs*
CSE305	**1**	**1234**	**Hum 22**	**Joe**
CSE305	**1**	**1234**	**Hum 22**	**Mary**

Representation of Role in SQL

- Each role of relationship type produces a foreign key in corresponding relation
  - Foreign key references table corresponding to entity type from which role values are drawn

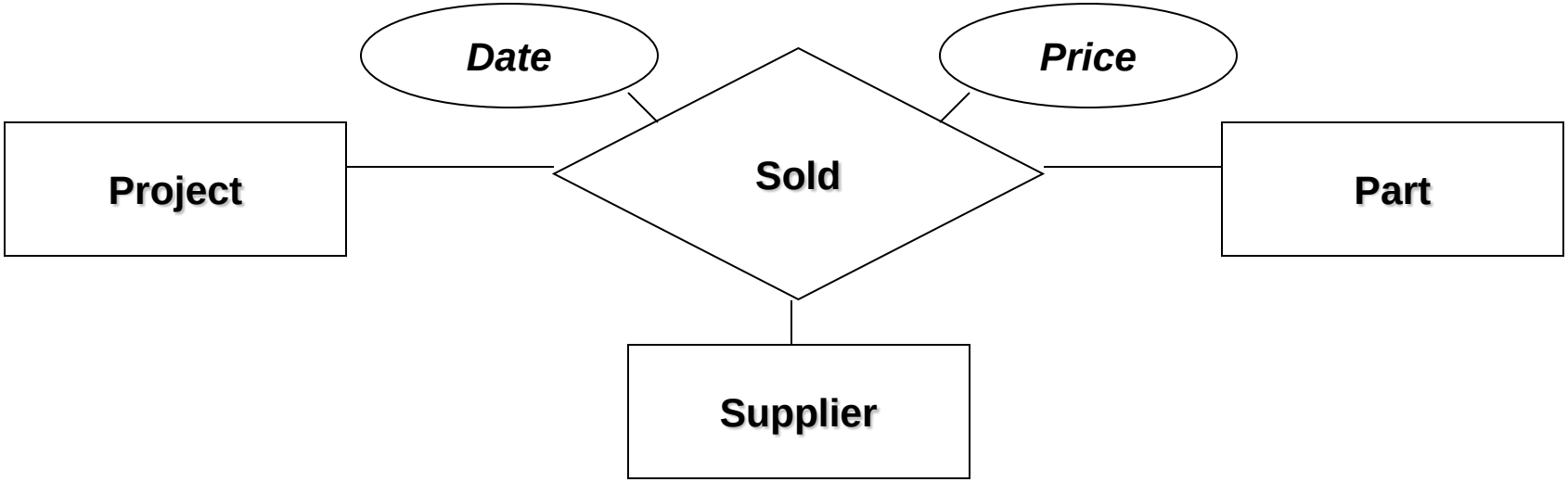
SUNY Korea

Example 1



CREATE TABLE WorksIn (
Since DATE, -- _attribute_
Status CHAR (10), -- _attribute_
ProfId INTEGER, -- _role (key of_ Professor_)_
DeptId CHAR (4), -- _role (key of_ Department_)_
PRIMARY KEY (_ProfId_), -- _since a professor works in at most one department_
FOREIGN KEY (_ProfId_) REFERENCES Professor (_Id_),
FOREIGN KEY (_DeptId_) REFERENCES Department)

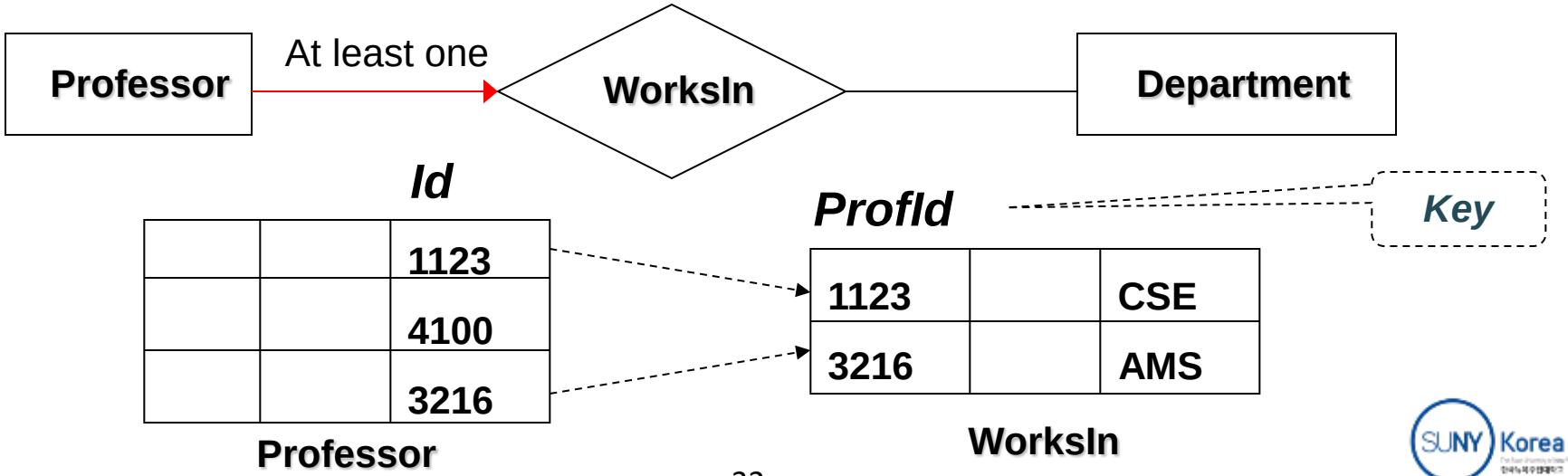
Example 2




```
CREATE TABLE  Sold  (
   Price  INTEGER,                -- attribute
   Date  DATE,                    -- attribute
   ProjId  INTEGER,               -- role
   SupplierId  INTEGER,           -- role
   PartNumber  INTEGER,           -- role
   PRIMARY KEY (ProjId, SupplierId, PartNumber, Date),
   FOREIGN KEY (ProjId) REFERENCES Project(Id),
   FOREIGN KEY (SupplierId) REFERENCES Supplier (Id),
   FOREIGN KEY (PartNumber) REFERENCES Part (Number)  )
```

Representing Single-Role Key Constraints in Relational Model

- *Key of the relation corresponding to the entity type is key of the relation corresponding to the relationship type*
  - *Id* is PK of **Professor**; *ProfId* is key of **WorksIn**
  - Cannot use FK in **Professor** to refer to **WorksIn** since some professors (e.g., one with Id 4100) may not work in any dept
    - But *ProfId* is a foreign key in **WorksIn** that refers to **Professor**



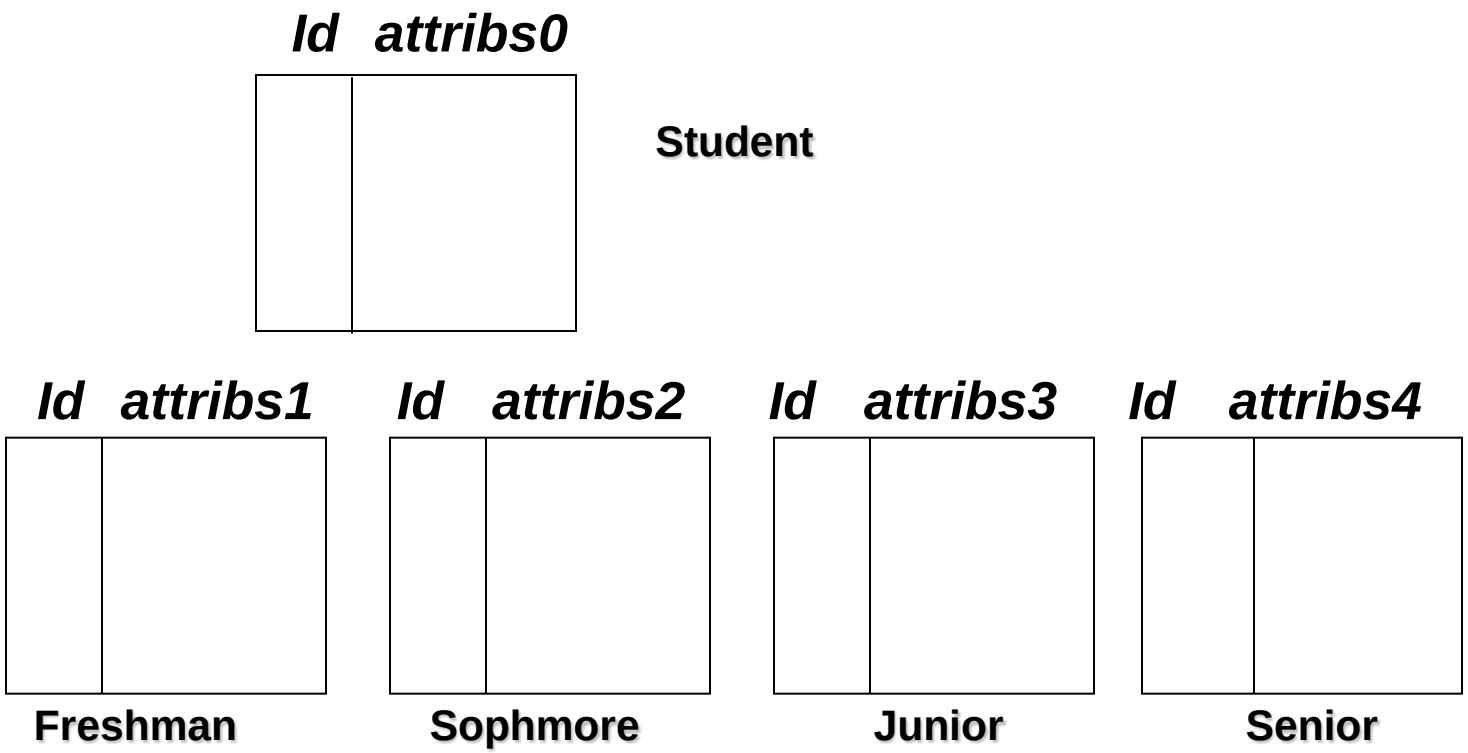
Representing Type Hierarchies in the Relational Model

- Supertypes and subtypes can be realized as separate relations
  - Need a way of identifying subtype entity with its (unique) related supertype entity
    - Choose a CK and make it an attribute of all entity types in hierarchy

SUNY Korea

Type Hierarchies and the Relational Model

- Translated by adding the PK of supertype to all subtypes
- Make FK from subtypes to the supertype



FOREIGN KEY *Id* REFERENCES Student

in Freshman, Sophomore, Sunior, Senior

Type Hierarchies and Relational Model

- Redundancy eliminated if **IsA** is not disjoint
  - For individuals who are both employees and students, Name and DOB are stored only once

Person

SSN	*Name*	*DOB*
1234	Mary	1950

Employee

SSN	*Department*	*Salary*
1234	Accounting	35000

Student

SSN	*GPA*	*StartDate*
1234	3.5	1997

Type Hierarchies and Relational Model

- If all disjoint,
  - Use a single relation (i.e., union of subtypes + a special column)
    - An example: UGRAD_YEAR with the domain of {‘freshmen’, ‘sophomores’, ‘juniors’, ‘seniors’}

- If covering,
  - Create one relation per each subtype
  - Attributes of the relation
    - the union of attributes of the subtype and those of supertype

SUNY Korea

Discussed So Far...

- Entity instance, Entity Type
  - Attribute, domain, key,…
- Relationship instance, Relationship Type
  - Relationship attribute, role, relationship key
  - Single-role key constraint (at most one, arrow)
  - Participation constraint (at least one, thick line)
- Entity type hierarchy (IsA relaionship)
  - Covering, disjoint
- Representation of E-R model into relational data model
  - Multi-valued attribute
  - Single-role key constraint
  - IsA relationship

SUNY Korea

CSE 305 / CSE532

Lecture 05 (Chapter 4)

Conceptual Modeling of Databases with E-R Model and UML

Lecturer: Sael Lee

Slide adapted from the author's and Dr. Ilchul Yoon's slides.

Representing Participation Constraints in the Relational Model

- Inclusion dependency: Every professor works in at least one dept.
  - in the relational model: (easy)
    - Professor (*Id*) references WorksIn (*ProfId*) - means simply "references"
  - in SQL:
    - *Simple case* – *ProfId* is a key in WorksIn (i.e., ***if* every** professor works in <u>exactly one</u> department) then it is easy:
      - Solution 1: FOREIGN KEY Professor (*Id*) REFERENCES WorksIn (*ProfId*)
      - Solution 2: We can combine the tables Professor and WorksIn
    - *General case* – *ProfId* is not a key in WorksIn, so can't use foreign key constraint. Then must use **ASSERTION** statement – more complex (shown later)

Participation and Key Constraint in the Relational Model – Simple Case

- Example:

	Id
xxxxxx	1123
yyyyyy	4100
zzzzzzz	3216

Professor

ProfId		
1123		CSE
4100		ECO
3216		AMS

WorksIn

Representing Participation and Key Constraint in SQL – Simple Case, Solution 1

- **If both participation and key constraints are specified**, then we can use foreign key constraint in entity table

```
CREATE TABLE Professor  (
   Id   INTEGER,
      ……
   PRIMARY KEY (Id),      --  Id can't be null
   FOREIGN KEY (Id) REFERENCES WorksIn  (ProfId)
                                  --  all professors participate
)
```

Participation and Key Constraint in Relational Model – Simple Case, Solution 2

- Alternative solution if both key and participation constraints apply
  - Merge the tables representing the entity and relationship types
    - Can be done since the 1:1 and onto relationship between the rows of the entity set and the rows in the relationship sets allows us to put all attributes in one table

SUNY Korea

Participation and Key Constraint in Relational Model – Simple Case, Solution 2

- Example

```
CREATE TABLE  Prof_WorksIn  (
    Id   INTEGER,
    Name CHAR(40),
    DeptId  CHAR(4),
    PRIMARY KEY (Id)
)
```

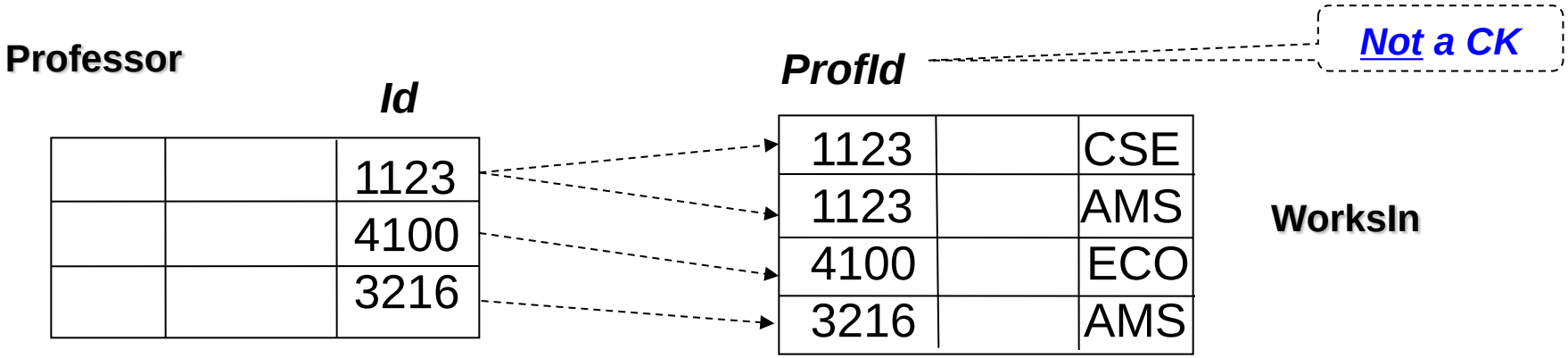
Name	***Id***		***DeptId***
xxxxxxx	1123		CSE
yyyyyyy	4100		ECO
zzzzzzzz	3216		AMS

Prof_WorksIn

***Merge of* Professor *and* WorksIn**

Representing Participation Constraint in the Relational Model – General Case

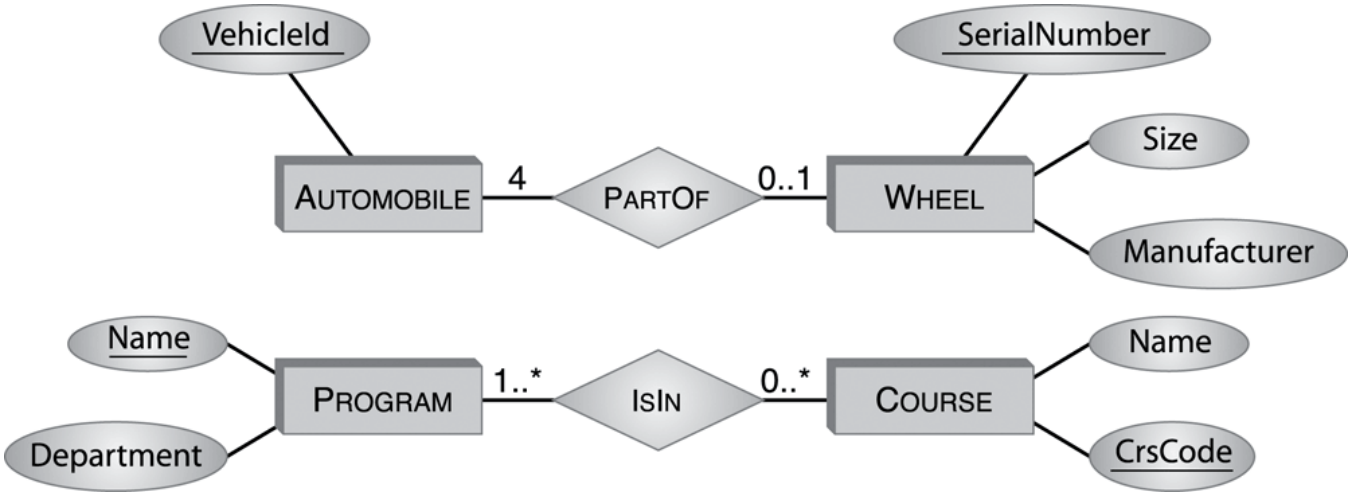
- Cannot use foreign key in **Professor** <u>if *ProfId* is not a CK</u> in **WorksIn** – must use assertions




```
CREATE ASSERTION  ProfsInDepts
   CHECK ( NOT EXISTS (
      SELECT  *  FROM  Professor P
      WHERE NOT EXISTS (
         SELECT  *  FROM  WorksIn W
         WHERE   P.Id = W.ProfId ) ) )
```

Representing Part-Of Constraint in the Relational Model

- **Non-exclusive** part-of
  - Case 1: Subpart can exist independently but can be part of at most one whole
    - Make one relation from subpart entity and relationship
  - Case 2: Subpart can exist independently and be shared between different wholes
    - Translate into separate relations

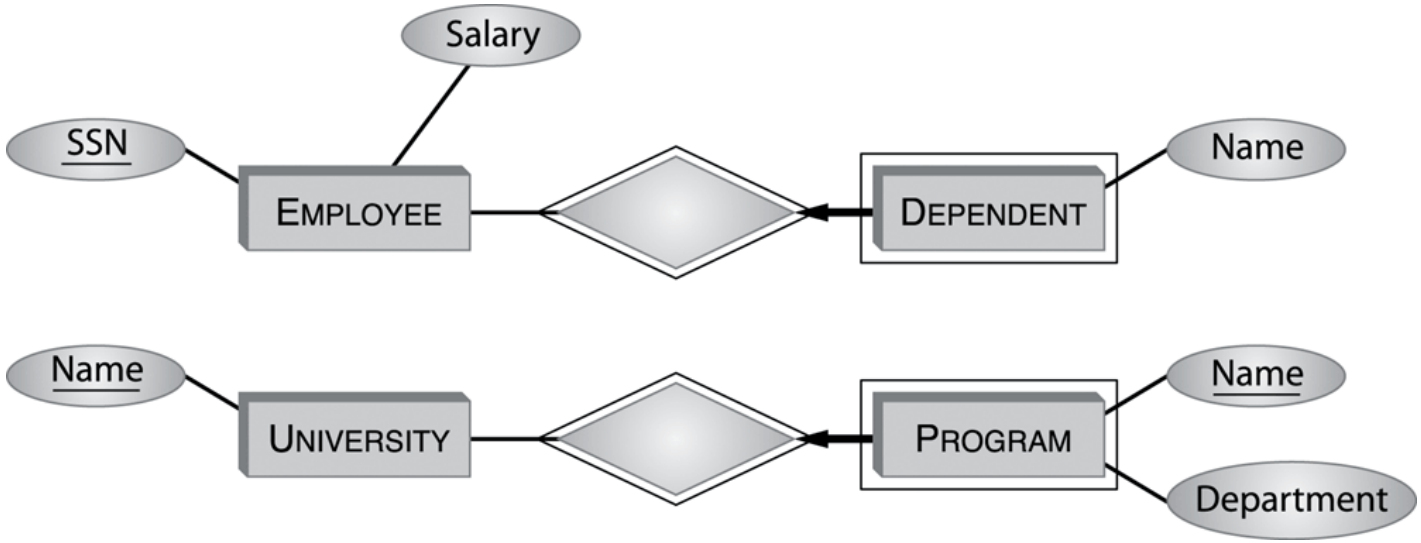


Representing Part-Of Constraint in the Relational Model

```
CREATE TABLE WheelMergedWithPartOf
    SerialNumber        INTEGER,
    Size                CHAR(10),
    Manufacturer        CHAR(20),
    VehicleID           CHAR(20),
    PRIMARY KEY         (SerialNumber),
    FOREIGN KEY         (VehicleID) REFERENCES Automobile
```

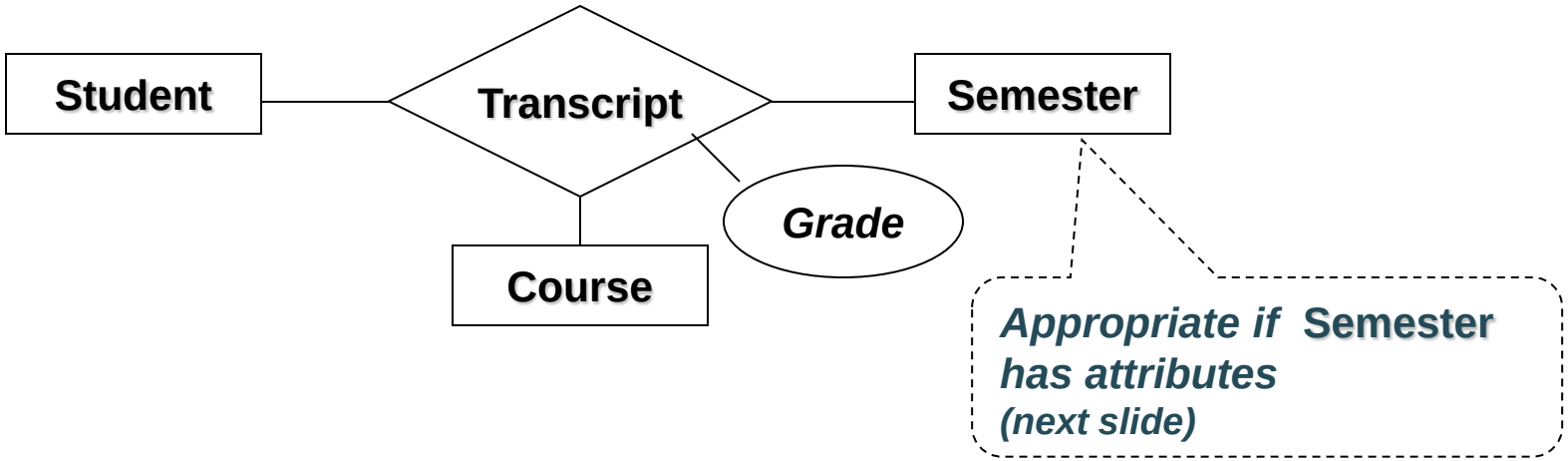
Representing Part-Of Constraint in the Relational Model

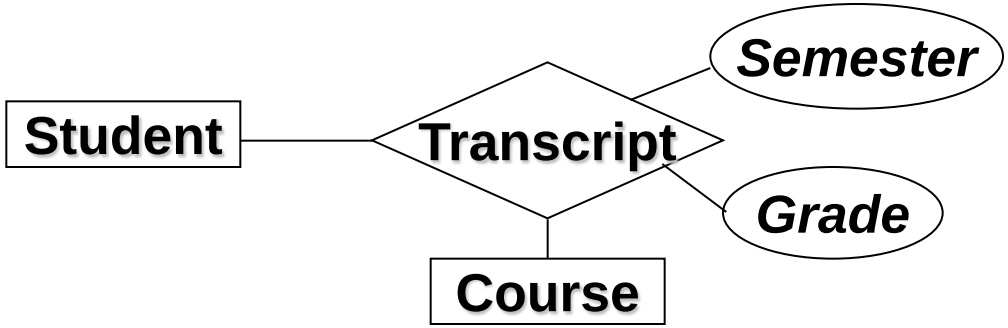
- Exclusive part-of
  - Subpart is a weak entity (one and only one participation)
    - Use the rule for participation and key constraints



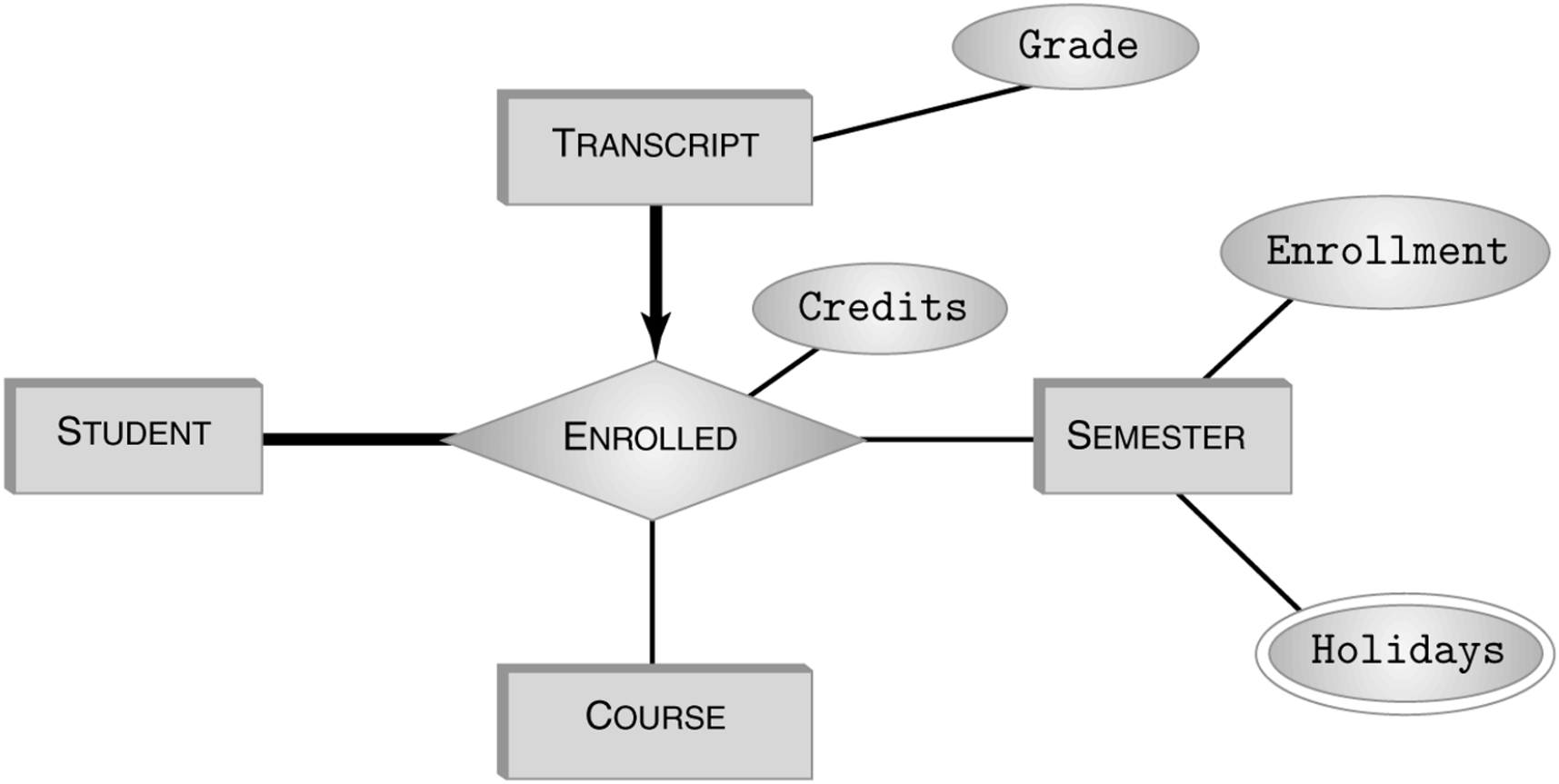
Entity or Attribute?

- Sometimes information can be represented as either an entity or an attribute.



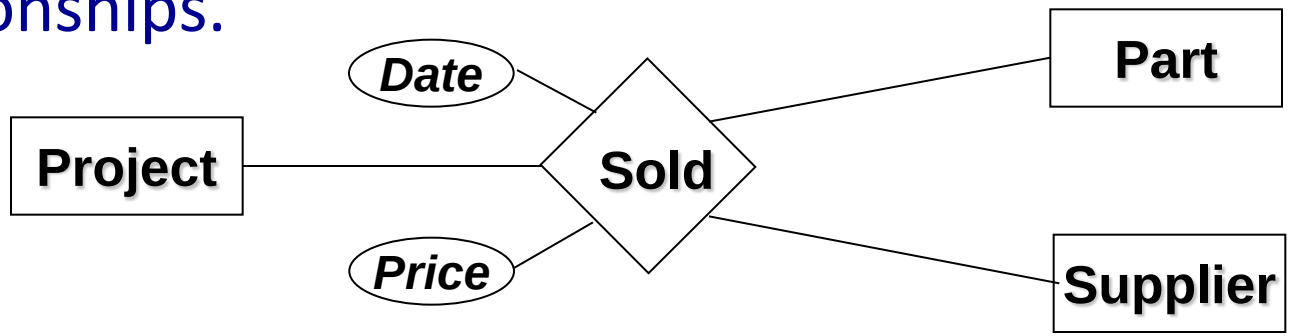


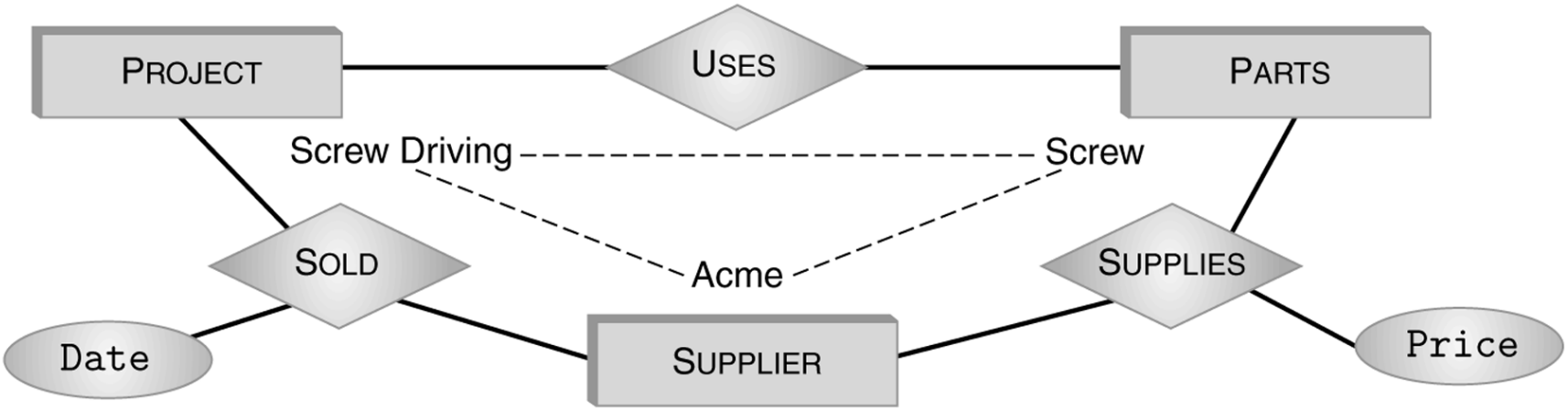
Entity or Relationship?



(Non-) Equivalence of Diagrams

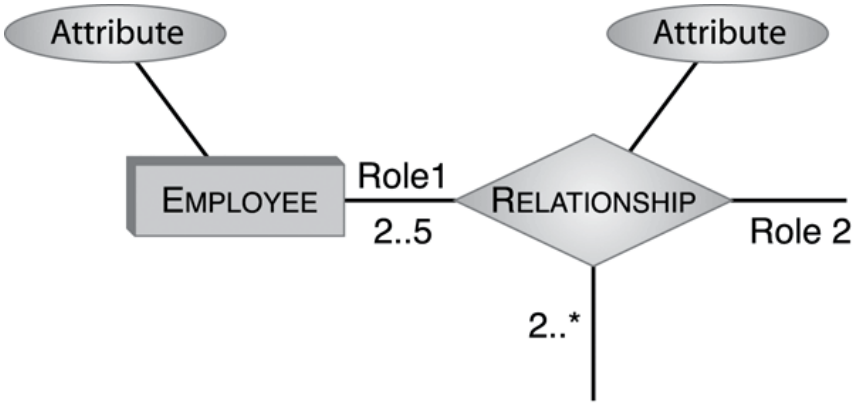
- Transformations between binary and ternary relationships.

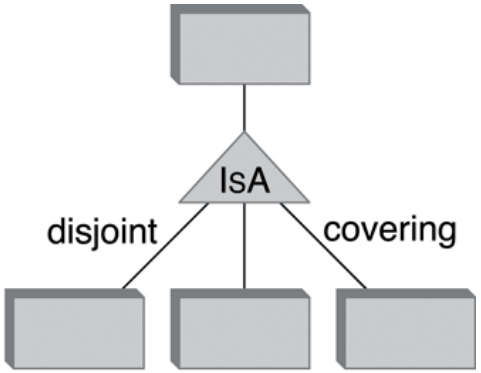


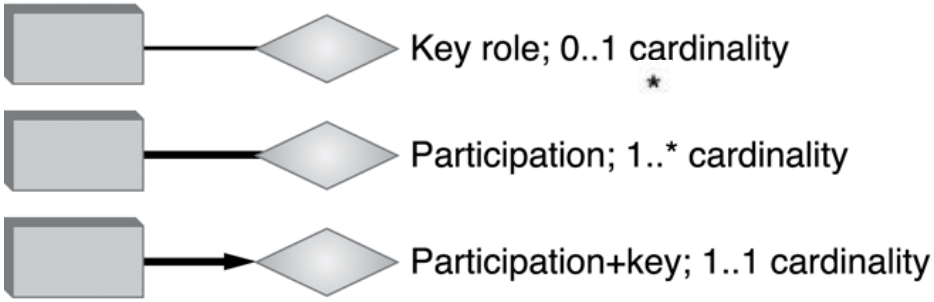


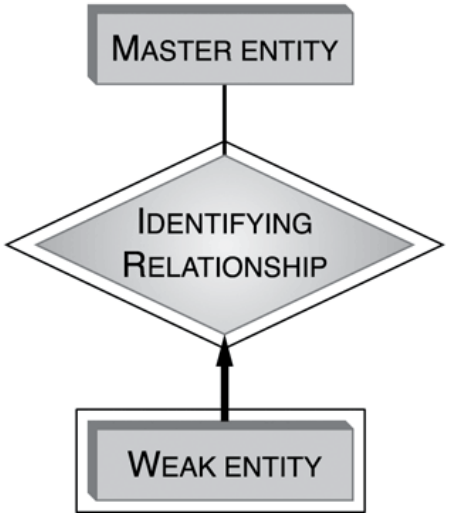
Information Lost!!!

Summary of E-R Notations



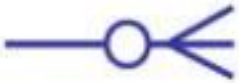






Crow's Foot Notation for Cardinality

from Zero to Many



from One to Many

from One to One

i.e., one and only one

from Zero to One

Use of **UML** to represent E-R Model

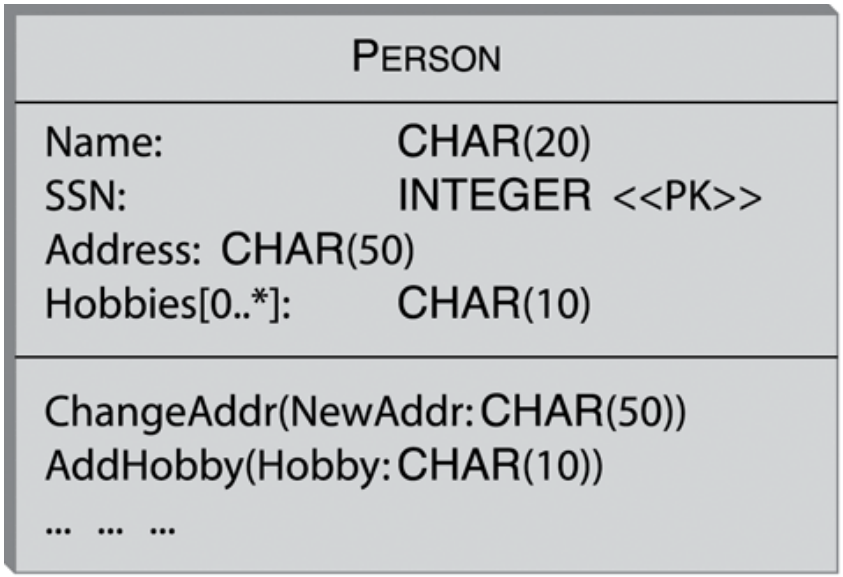
- Unified Modeling Language
  - Unified representation for software design
  - Class diagram will be used for E-R model

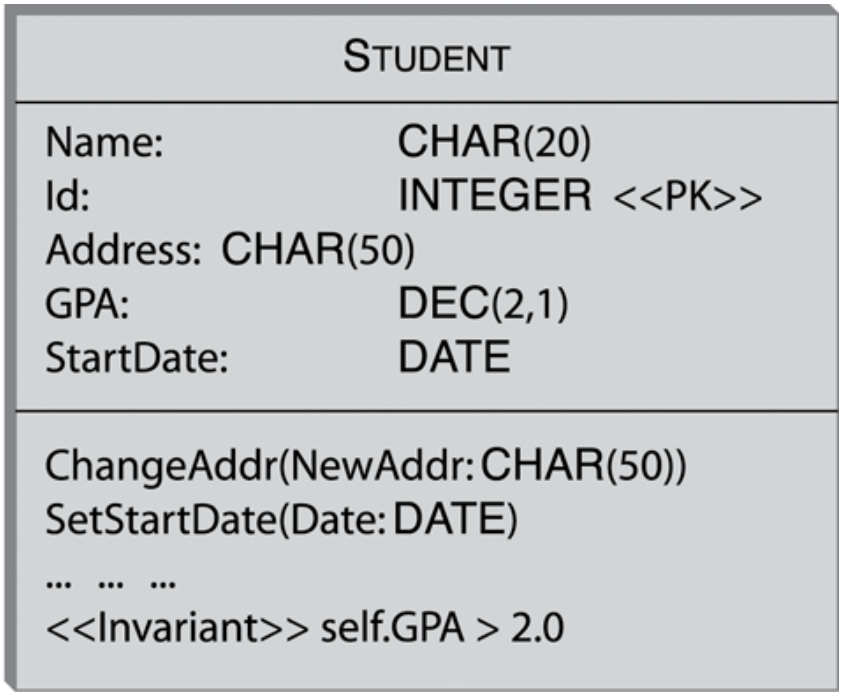
- Other diagrams in UML
  - Use-case diagram: user-interaction
  - Sequence diagram: dynamic interaction between objects
  - State diagram: state transition of complex objects
  - Activity diagram: flowchart or workflow
  - Communication diagram (formerly, collaboration diagram): message flow between objects
  - ….

Representing Entities in UML

- Modeled as classes
  - Attributes
  - Methods represent transactions associated with entities
- Object Constraint Language (OCL) for specifying constraints
  - Can be used for specifying keys, CHECK and ASSERTION
- Extensibility mechanism
  - Use of stereotype for PK and FK
    - *SSN*: **INT <<PK>>**
    - **<<FK Professor** (*Id*)**>>** *ProfID*: **INT**

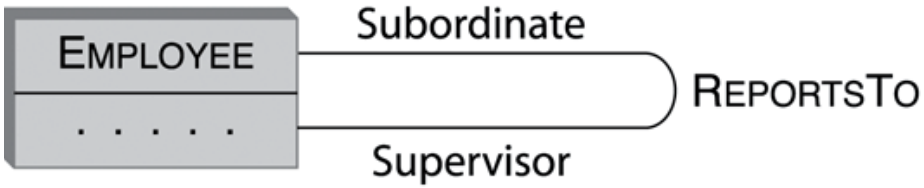
Representing Entities in UML

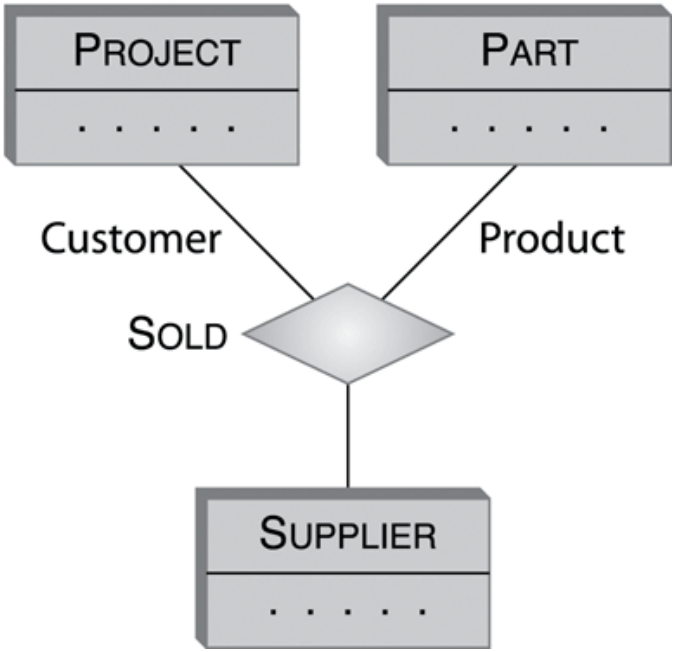




Representing Relationship in UML

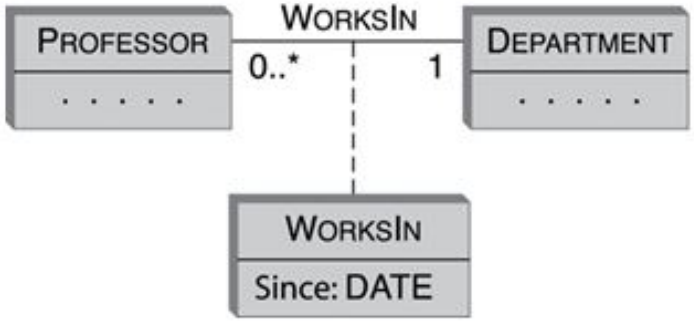
- Relationships are called <u>associations</u>
  - For binary association types, UML uses a line to connect classes
  - For 3+ association types, uses diamond symbol



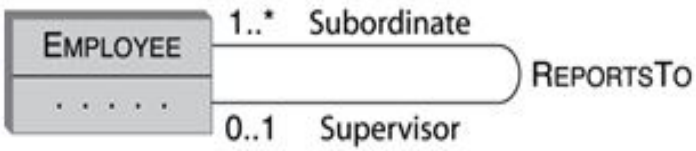


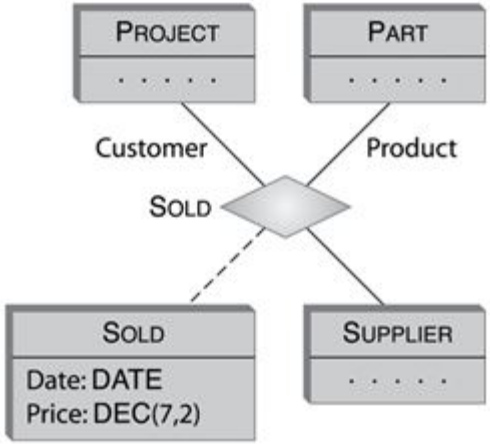
Representing Relationship in UML

- Association class
  - Previously, relationship attributes
  - Connected to an association using a dashed line



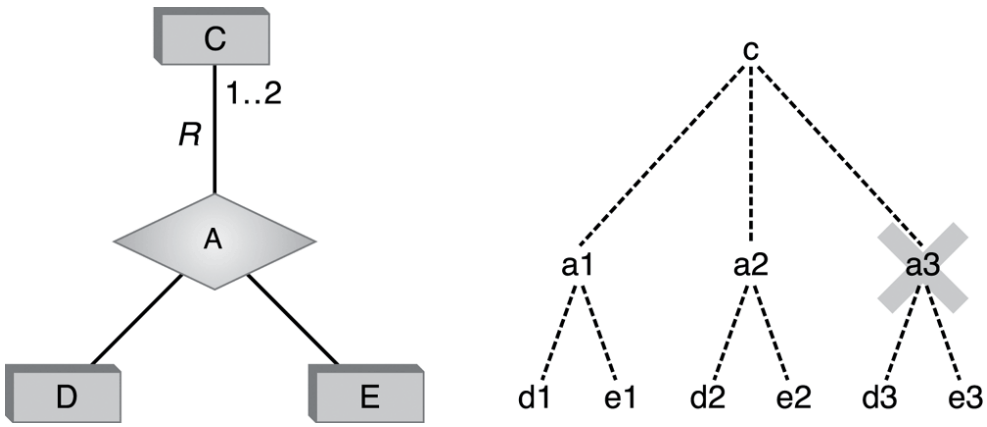
- Multiplicity constraints on roles
  - Range specification: *n..m*
    - *Lower and upper bounds on the number of objects of a class that can be connected by means of an association type to any given set (or combination) of objects attached to the other ends of the association.*



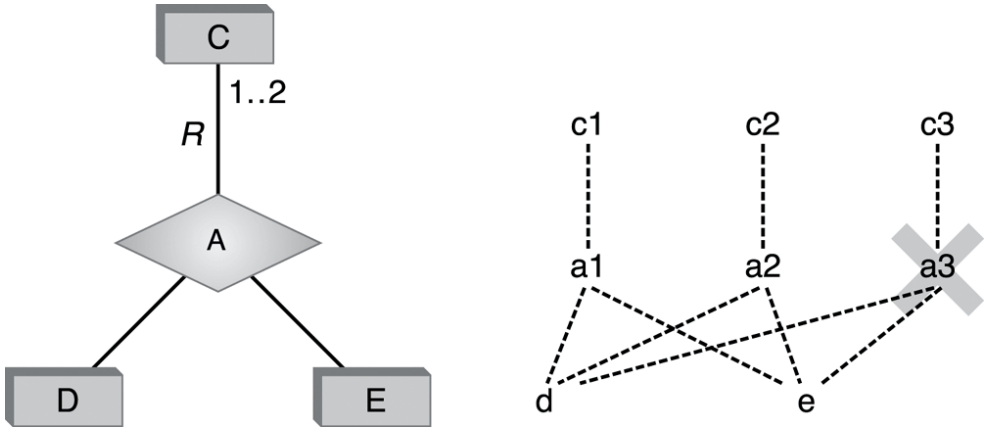


UML Multiplicity vs. E-R Cardinality Constraint

- Looks similar but has “opposite meaning”
- In E-R model,

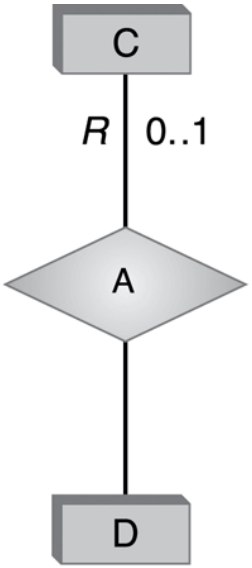


- In UML,

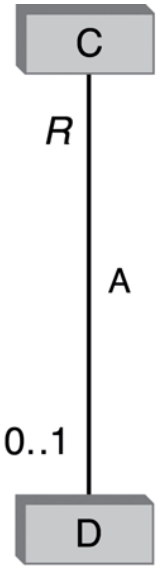


UML Multiplicity vs. E-R Cardinality Constraint

- Certainly, opposite meaning and same expression power (including key constraints) for binary associations



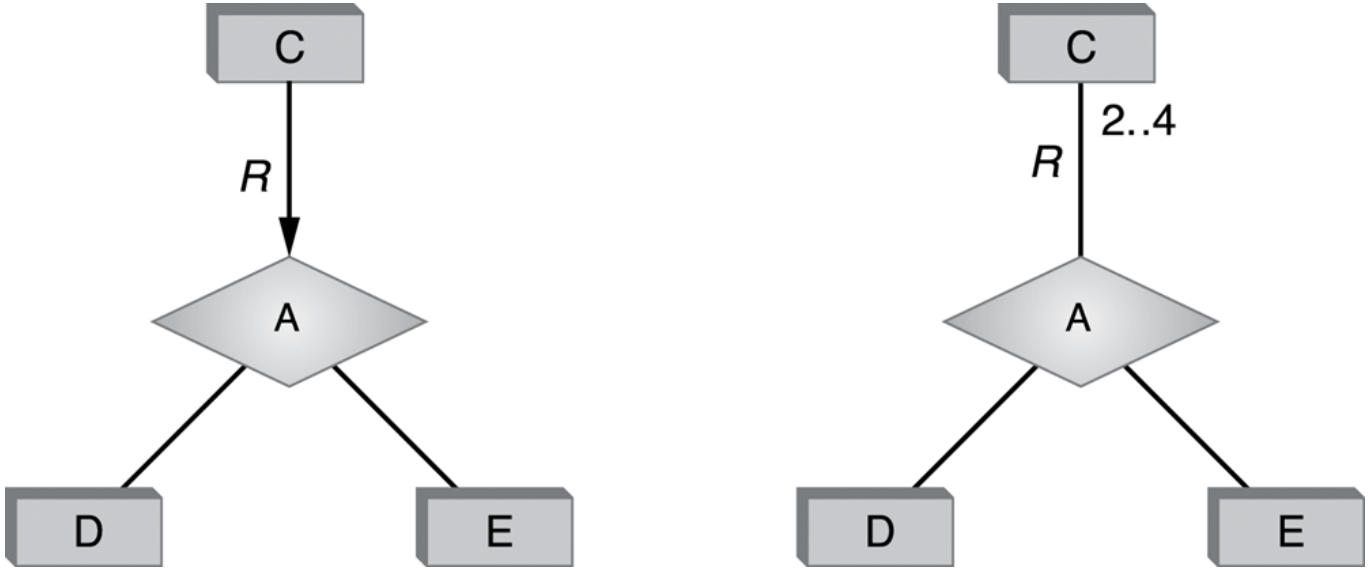
Cardinality constraint in E-R



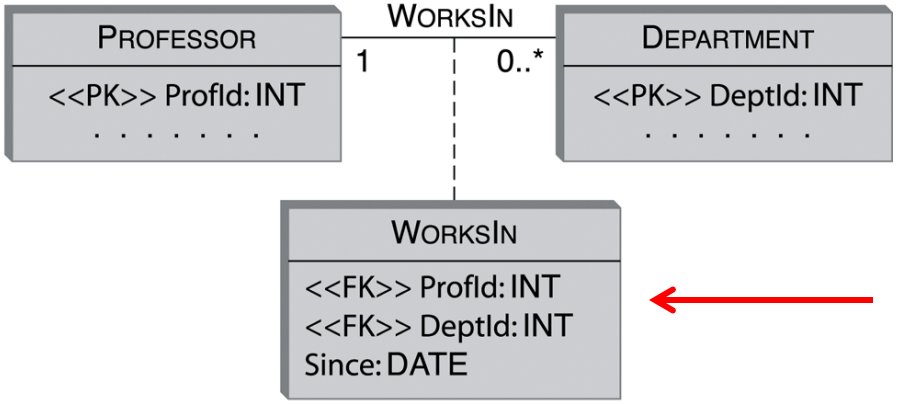
Equivalent multiplicity constraint in UML

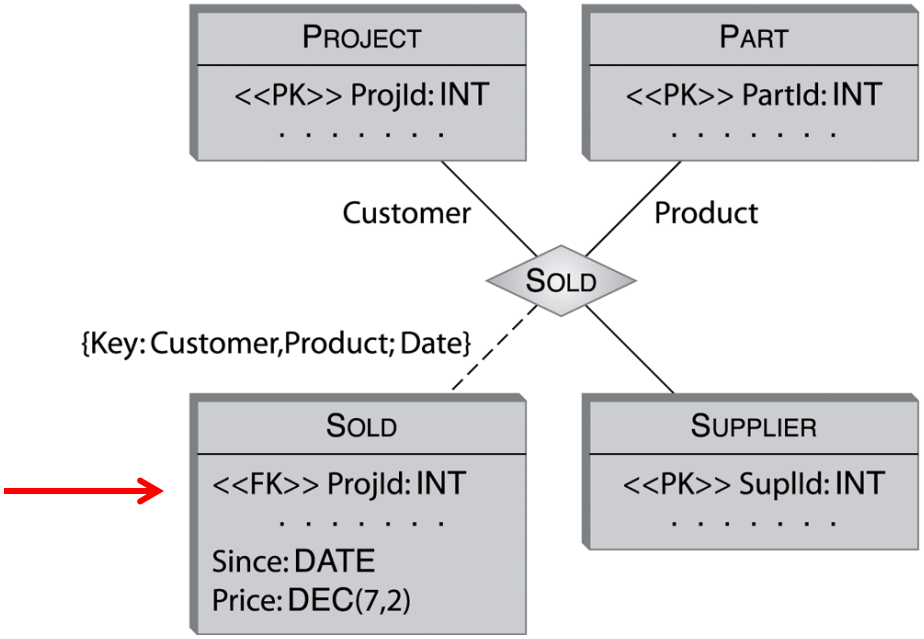
UML Multiplicity vs. E-R Cardinality Constraint

- For ternary and higher-degree, the difference is greater
  - Incomparable expression power
  - Some cardinality constraints cannot be represented in multiplicity in UML



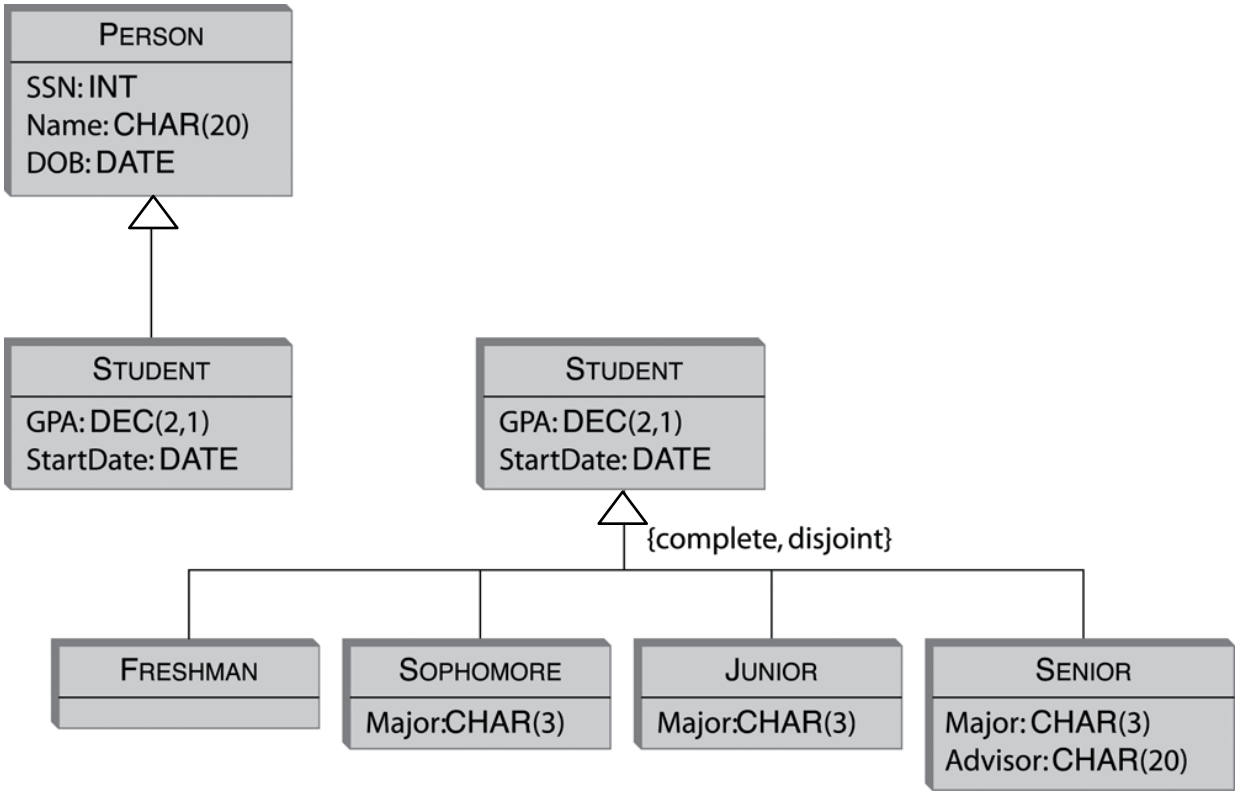
Foreign Keys in UML





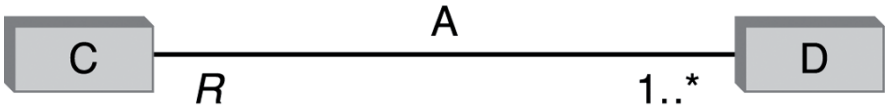
IsA Hierarchy in UML

- IsA relationship is called generalization
  - Solid arrow with a large hollow head leading from a subclass to a superclass

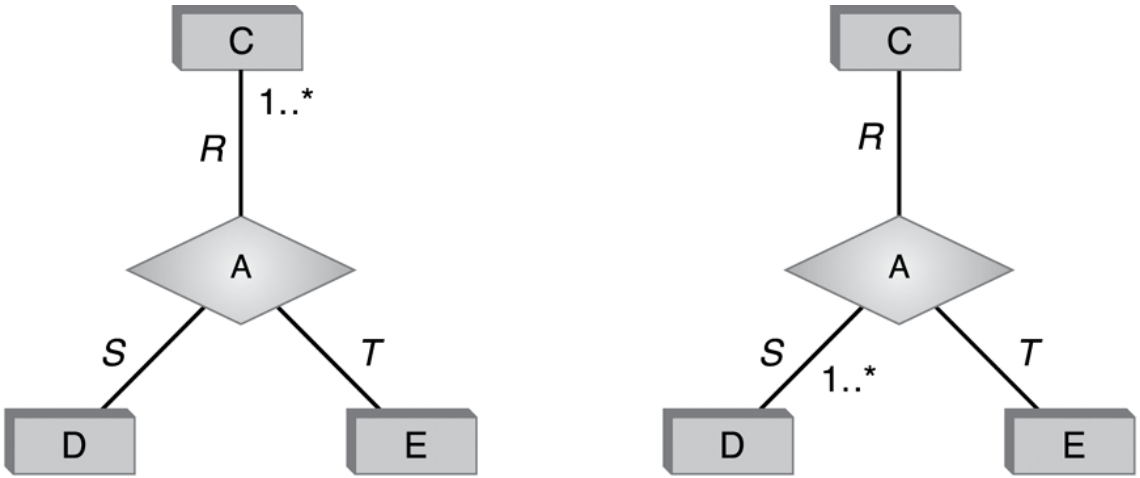


Participation Constraint in UML

- Obvious for binary associations
  - Use multiplicity lower bound greater than or equal to 1



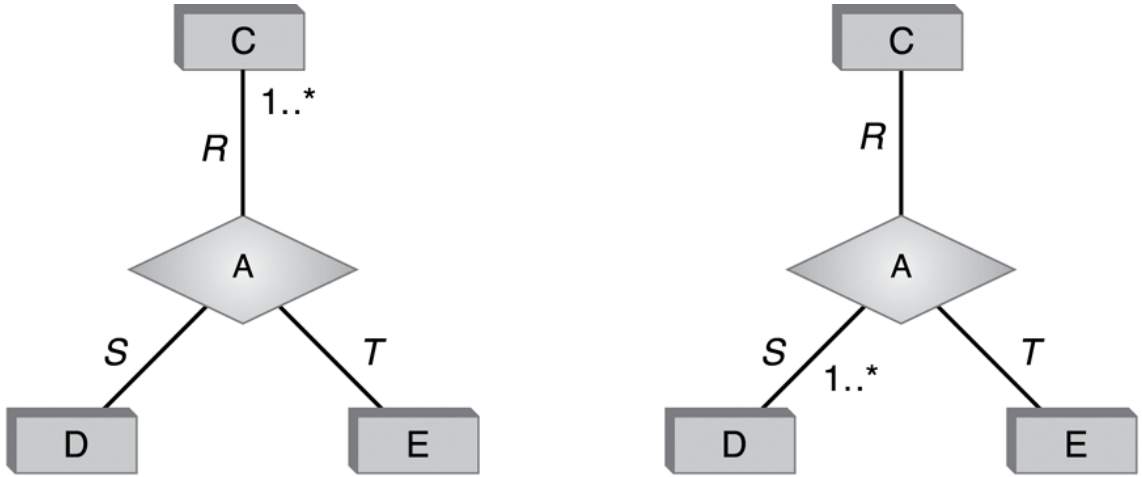
- For ternary and higher-degree associations?
  - Difficult to represent identical semantics



(a) Participation in a ternary relationship in E-R (b) Partial simulation of the same in UML

Participation Constraint in UML

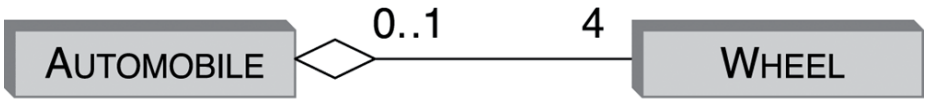
- In ER,
  - *for every entity $c \in C$ there are entities $d \in D$ and $e \in E$ that participate in a relationship $a \in A$ with c.*
- In UML,
  - *for every pair of objects $c \in C$ and $e \in E$, there is at least one object $d \in D$ that participates in a relationship $a \in A$ with c and e.*



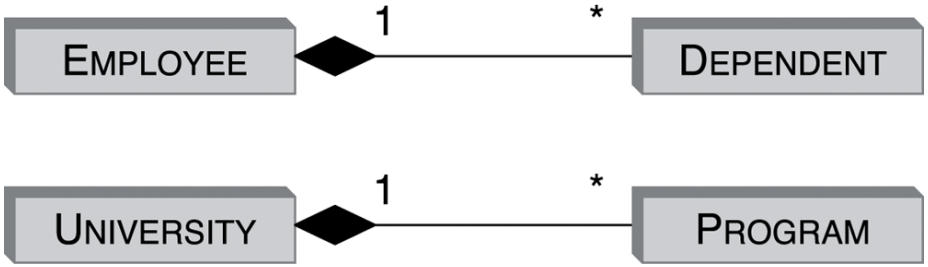
(a) Participation in a ternary relationship in E-R (b) Partial simulation of the same in UML

Part-Of Relationship in UML

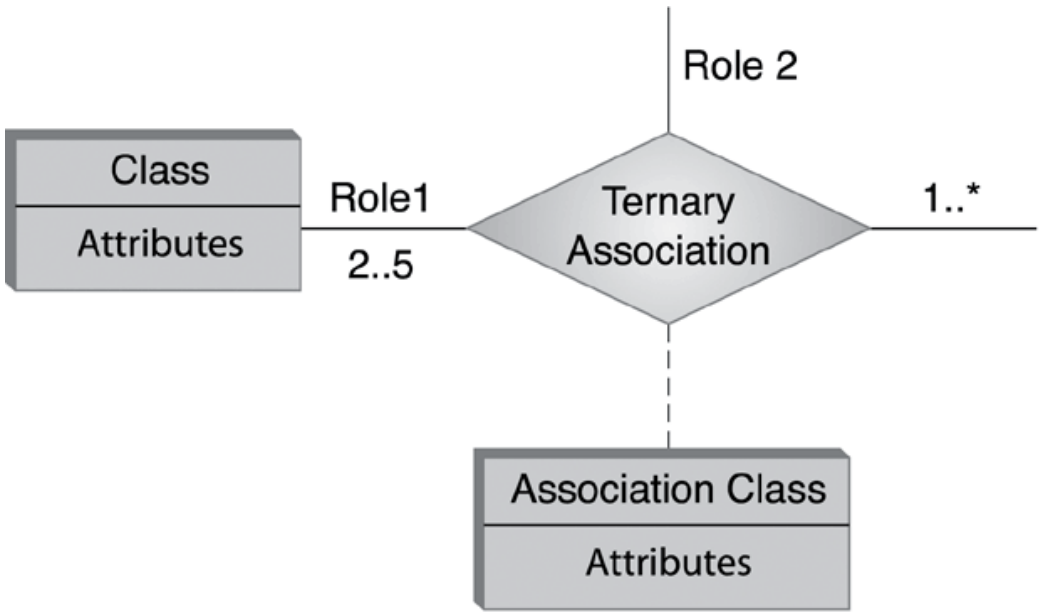
- Non-exclusive part-of relationship is called aggregation
  - Often accompanied by appropriate multiplicity constraints

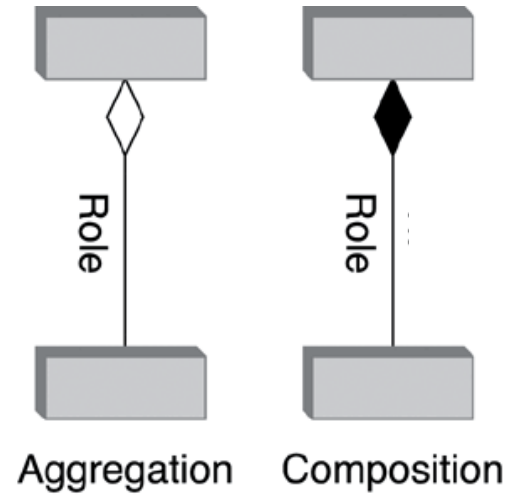


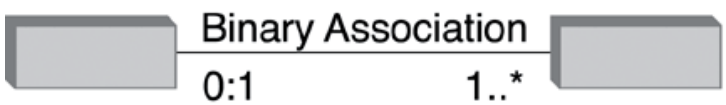
- Exclusive part-of relationship is called composition
  - Viewed as a special kind of aggregation

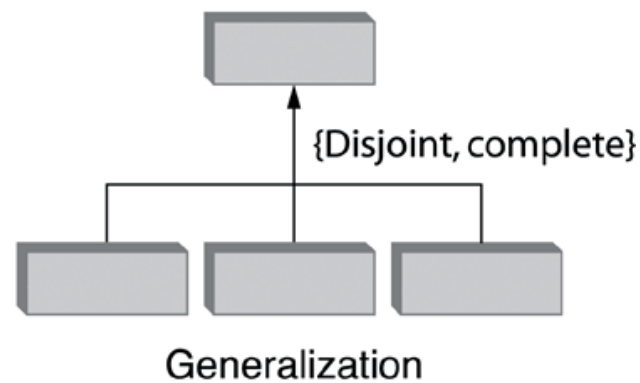


Summary of UML Notations









Company Schema

- Employ information to keep in the schema: first name, last name, SSN, birthdate, address, sex, salary
- Department information: department number, department name, location
- Project information: project number, name, location
- Dependent information: dependent name, sex, birth date, relationship

- Each employee work for a single department
- There is a manager for each department and the company records the work start date of the manager
- A department can operate multiple projects
- A department can be located in multiple locations
- An employee may have multiple dependents
- An employee can be assigned to multiple projects, and the work hours can be different between participating projects.
- Multiple departments can participate in a project

Company Schema
Business Rules and Referential integrity constraints

- Additional business constraints

- [R1] When an employee is fired, then all information about the dependents of that employee are deleted from the database.
- [R2] When an employee is fired, then all information about the employee working on a project can be deleted from the database.
- [R3] When a project is deleted, then all information about employees working on that project can be deleted. Caution: This does not mean that we delete employee information altogether!
- [R4] Employee information cannot be removed from the database if the employee is the manager of a department. In this case, the department needs to be assigned a new manager before the employee is deleted.
- [R5] An employee who is supervising another employee should not be deleted.
- [R6] A department cannot be deleted if there are current projects in the department.
- [R7] When a department is deleted all employees in that department are deleted.

Foreign key/ Relation name	Primary key/ Relation name	Integrity Constraint	Business Rule
DNO in Employee	DNUMBER in Department	ON DELETE CASCADE	R7
SuperSSN in Employee	SSN in Employee	ON DELETE NO ACTION	R5
MgrSSN in Department	SSN in Employee	ON DELETE NO ACTION	R4
DNUMBER in Dept_Location	DNUMBER in Department	ON DELETE CASCADE	-
DNUM in Project	DNUMBER in Department	ON DELETE NO ACTION	R6
ESSN in WORKS_ON	SSN in Employee	ON DELETE CASCADE	R2
PNO in WORKS_ON	PNUMBER in Project	ON DELETE CASCADE	R3
ESSN in DEPENDENT	SSN in Employee	ON DELETE CASCADE	R1