Extended Entity Relationship (EER) Model

- ER model has been widely used but have some shortcomings.
- Difficult to represent cases where an entity may have varying attributes dependent upon some property.
- ER model has been extended into Extended Entity Relationship model which includes more semantics such as generalization, categorization and aggregation.

Cardinality: One-to-one relationship

A one-to-one relationship between set A and set B is defined as: For all a in A, there exists at most one b in B such that a and b are related, and vice versa.

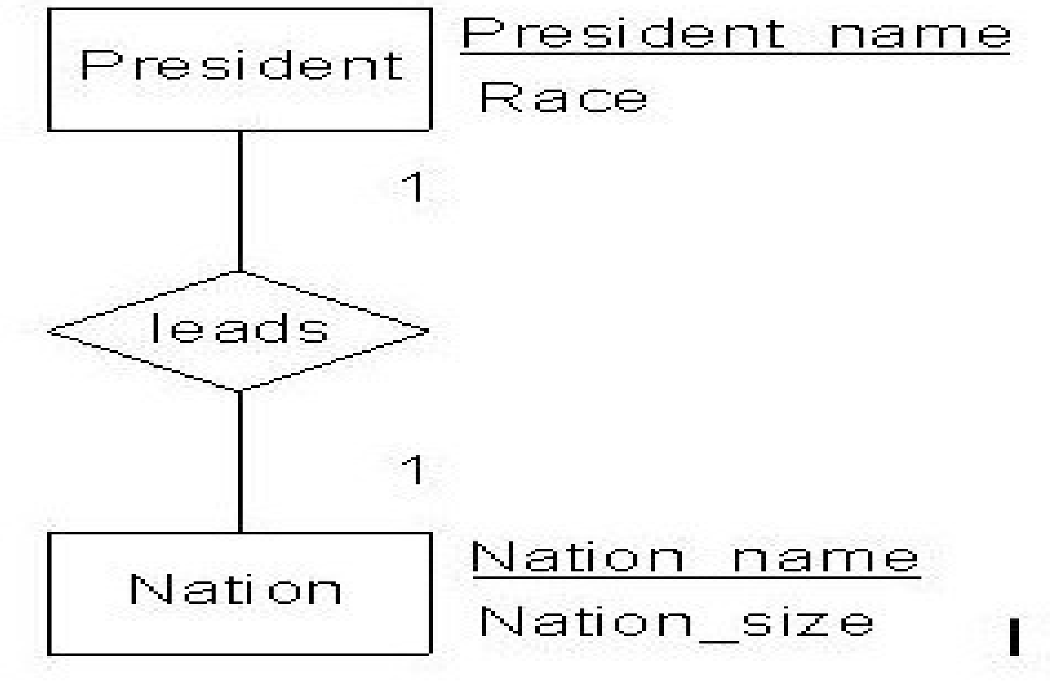
Example

A president leads a nation.

Relational Model:

Relation President (<u>President_name</u>, Race, *Nation_name)
Relation Nation (<u>Nation_name</u>, Nation_size)
Where underlined are primary keys and "*" prefixed are foreign keys

Extended Entity Relationship mod



Cardinality: Many-to-one relationship

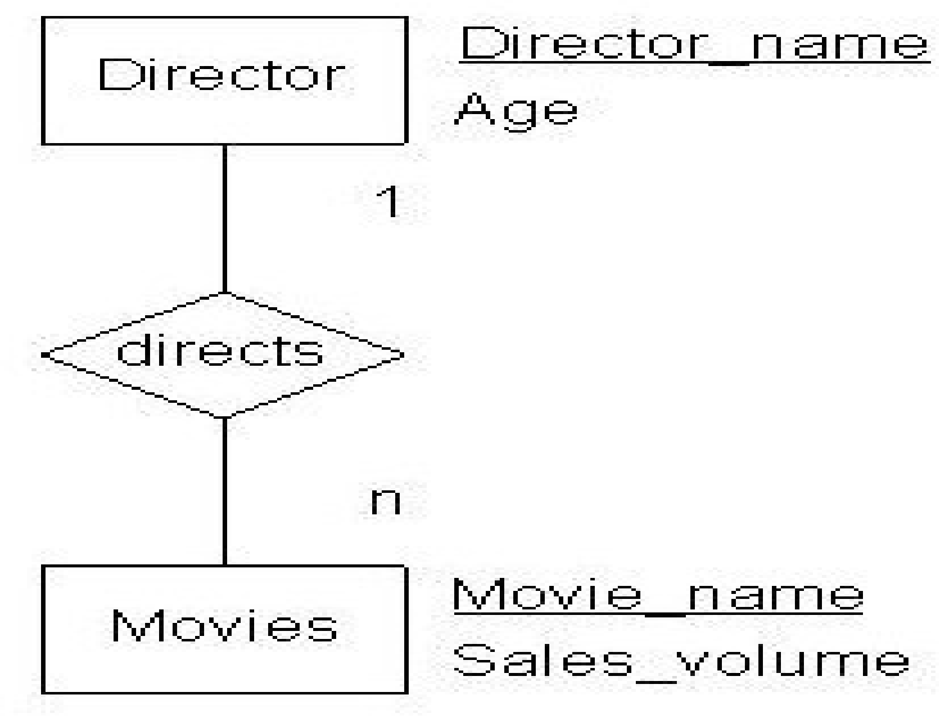
A many-to-one relationship from set A to set B is defined as: For all a in A, there exists at most one b in B such that a and b are related, and for all b in B, there exists zero or more a in A such that a and b are related.

Example A director directs many movies.

Relational Model:

Relation Director (Director_name, Age)
Relation Movies (Movie_name, Sales_volume, *Director_name)

Extended entity relationship model:



Cardinality: Many-to-many relationship

A many-to-many relationship between set A and set B is defined as: For all a in A, there exists zero or more b in B such that a and b are related, and vice versa.

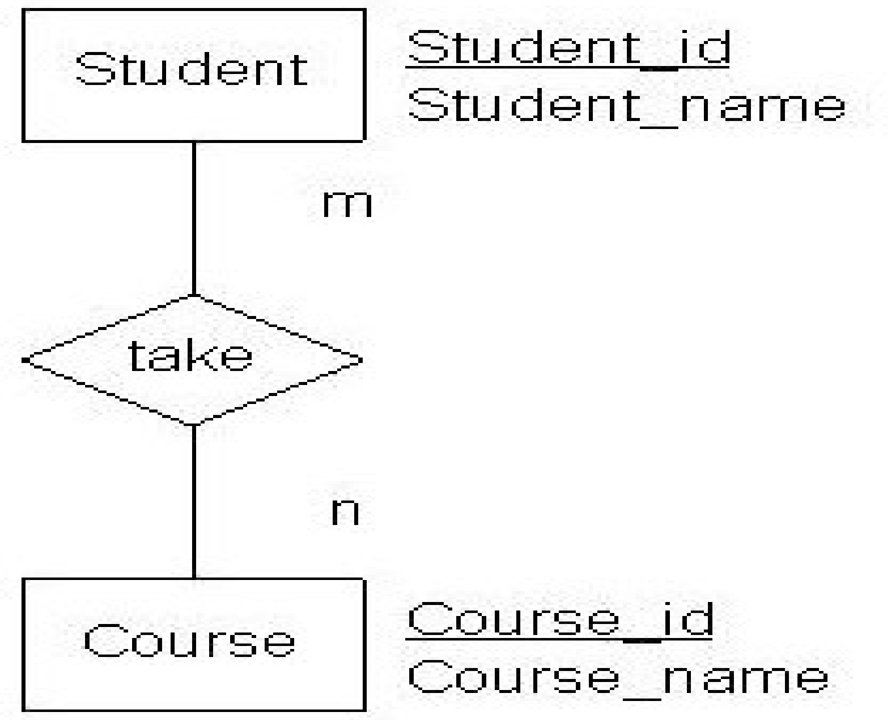
Example

Many students take many courses such that a student can take many courses and a course can be taken by many students.

Relational Model:

Relation Student (Student_id, Student_name)
Relation Course (Course_id, Course_name)
Relation take (*Student_id, *Course_id)

Extended entity relationship model:



Data Semantic: Is-a (Subtype) relationship

The relationship A isa B is defined as: A is a special kind of B.

Example

Father is Male.

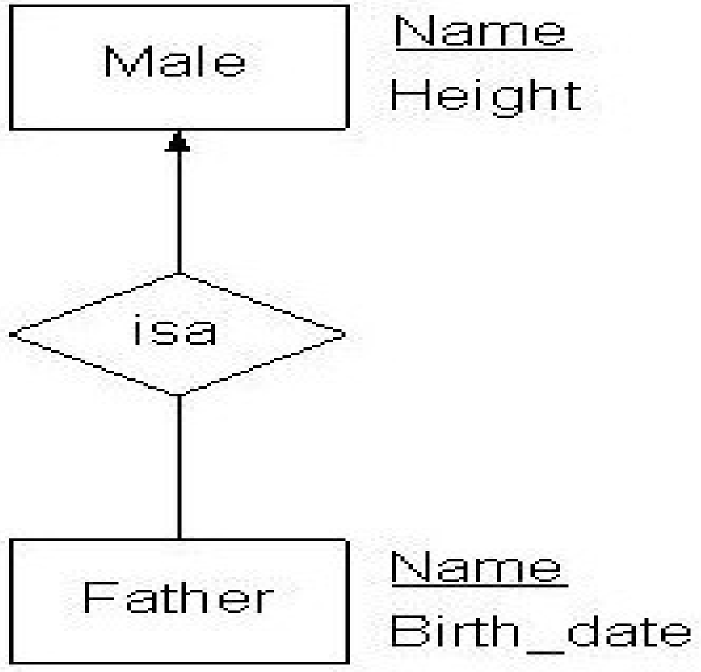
Relational Model:

Relation Male (Name, Height)

Relation Father (*Name, Birth_date)

Chú ý rằng ở đây vẫn là quan hệ 1 - 1

Extended entity relationship model



Data Semantic: Disjoint Generalization

-Generalization is to classify similar entities into a single entity. More than one is-a relationship can form data abstraction (i.e. super-class and subclasses) among entities.

- A subclass entity is a subset of its super-class entity. There are two kinds of generalization.
- The first is disjoint generalization such that subclass entities are mutually exclusive.
- The second is overlap generalization such that subclass entities can overlap each other.

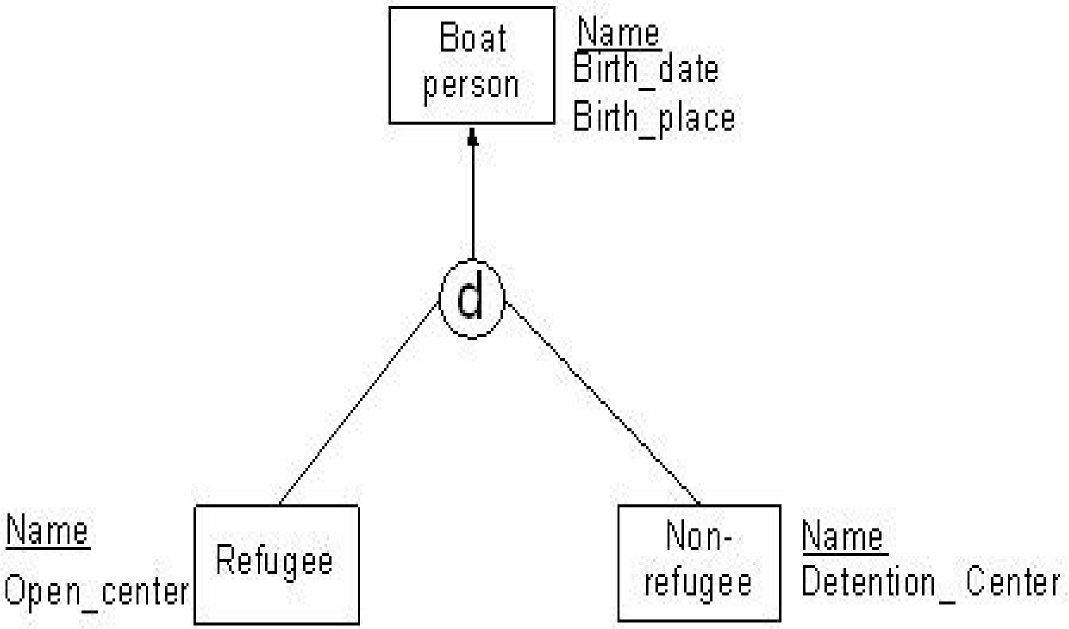
Example of Disjoint Generalization

A refugee and a non-refugee can both be a boat person, but a refugee cannot be a non-refugee, and vice versa.

***Relational Model**:*

Relation Boat_person (Name, Birth_date, Birth_place)
Relation Refugee (*Name, Open_center)
Relation Non-refugee (*Name, Detention_center)

***Extended entity relationship model**:*



Data Semantic: Overlap Generalization

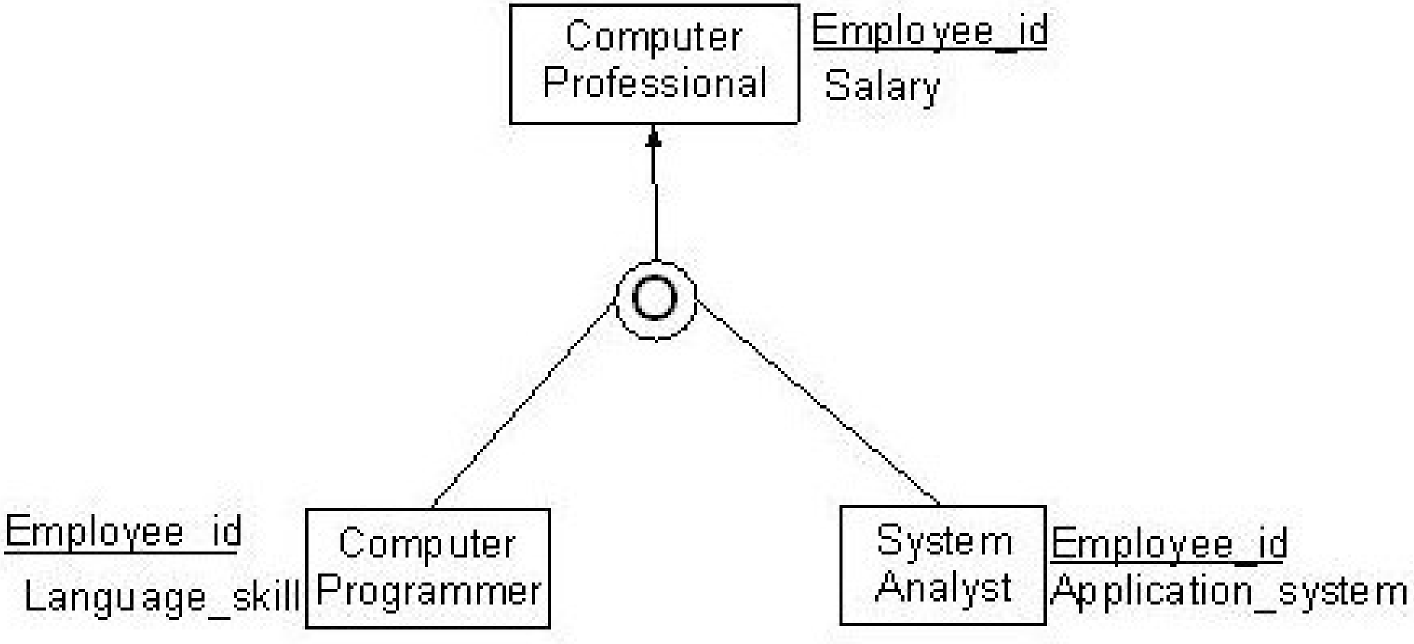
Example of Overlap Generalization

A computer programmer and a system analyst can both be a computer professional, and a computer programmer can also be a system analyst, and vice versa.

Relational Model:

Relation Computer_professional (Employee_id, Salary)
Relation Computer_programmer (*Employee_id, Language_skill)
Relation System_analyst (*Employee_id, Application_system)

Extended entity relationship model:



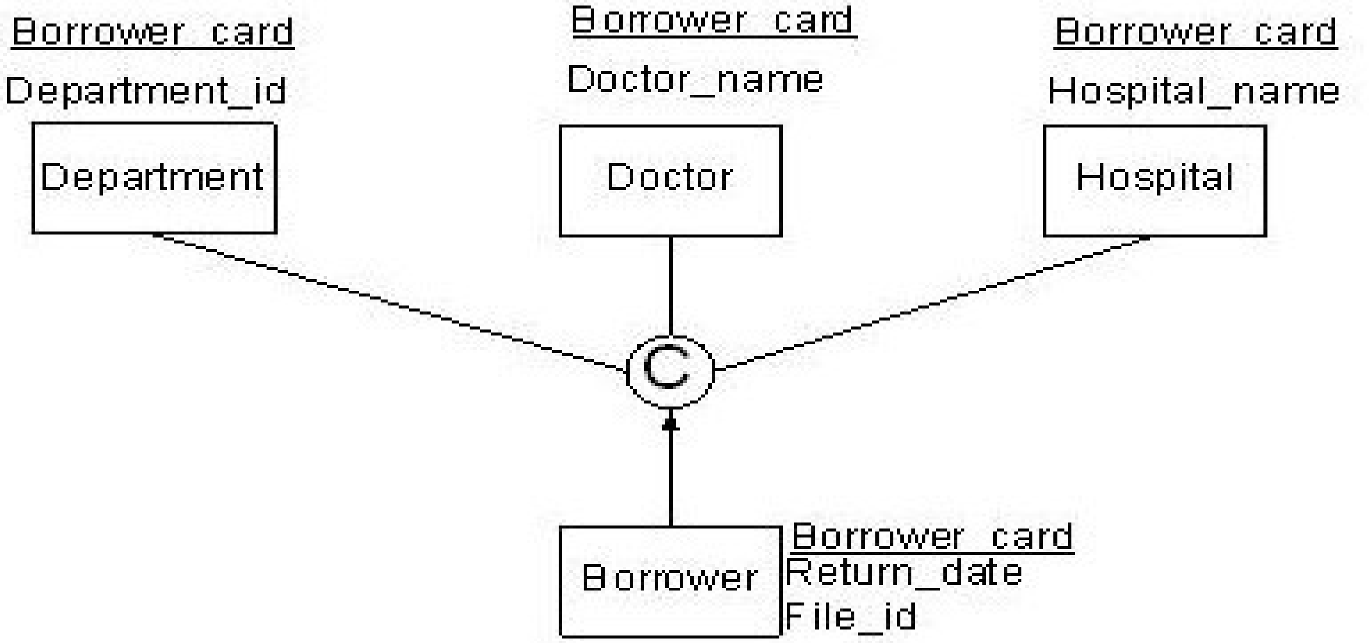
Data Semantic: Categorization Relationship

In cases the need arises for modeling a single super-class/subclass relationship with more than one super-class (es), where the super-classes represent different entity types. In this case, we call the subclass a category.

Relational Model:

Relation Department (Borrower_card, Department_id)
Relation Doctor (Borrower_card, Doctor_name)
Relation Hospital (Borrower_card, Hospital_name)
Relation Borrower (*Borrower_card, Return_date, File_id)

Extended Entity Relationship Model



Data Semantic: Aggregation Relationship

Aggregation is a method to form a composite object from its components. It aggregates attribute values of an entity to form a whole entity.

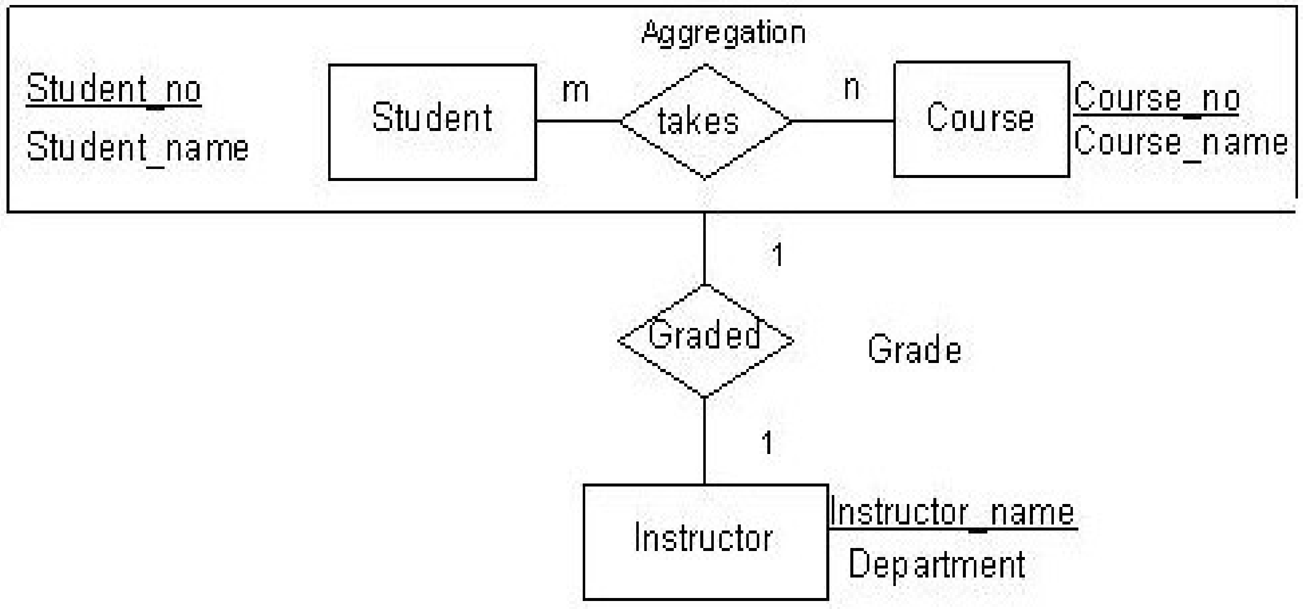
Example

The process of a student taking a course can form a composite entity (aggregation) that may be graded by an instructor if the student completes the course.

Relational Model:

Relation Student (Student_no, Student_name)
Relation Course (Course_no, Course_name)
Relation Takes (*Student_no, *Course_no, *Instructor_name)
Relation Instructor (Instructor_name, Department)

Extended Entity Relationship Model



Data Semantic: Total Participation

An entity is in total participation with another entity provided that all data occurrences of the entity must participate in a relationship with the other entity.

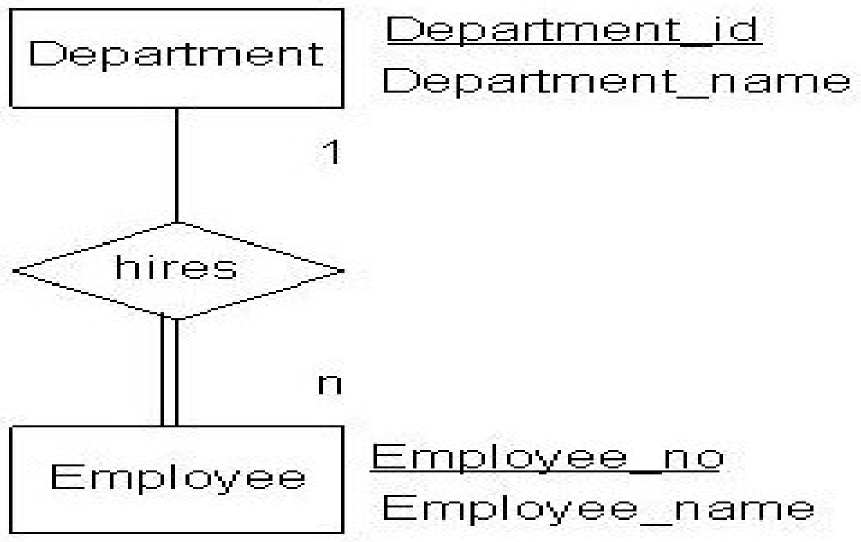
Example

An employee must be hired by a department.

Relational Model:

Relation Department (Department_id, Department_name)
Relation Employee (Employee_id, Employee_name, *Department_id)

Extended entity relationship model:



Mọi nhân viên đều phải thuộc 1 Department nào đấy

Data Semantic: Partial Participation

An entity is in partial participation with another entity provided that the data occurrences of the entity are not totally participate in a relationship with the other entity.

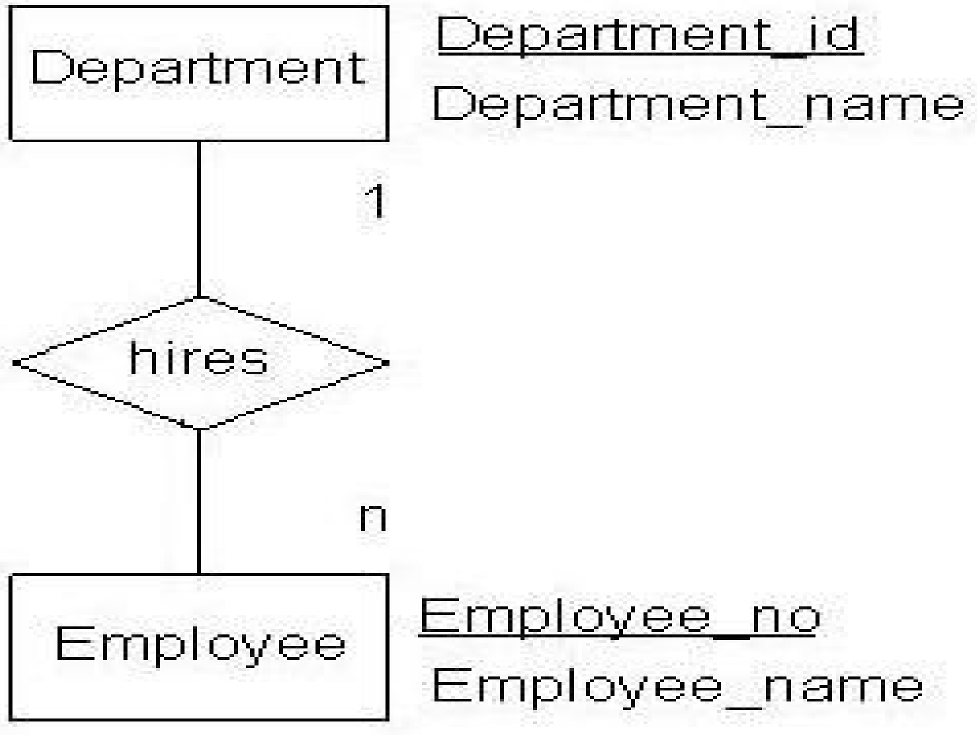
Example

An employee may be hired by a department.

Relational Model:

Relation Department (Department_id, Department_name)
Relation Employee (Employee_no, Employee_name, &Department_id)
Where & means that null value is allowed

Extended entity relationship model:



Data Semantic: Weak Entity

The existence of a weak entity depends on its strong entity.

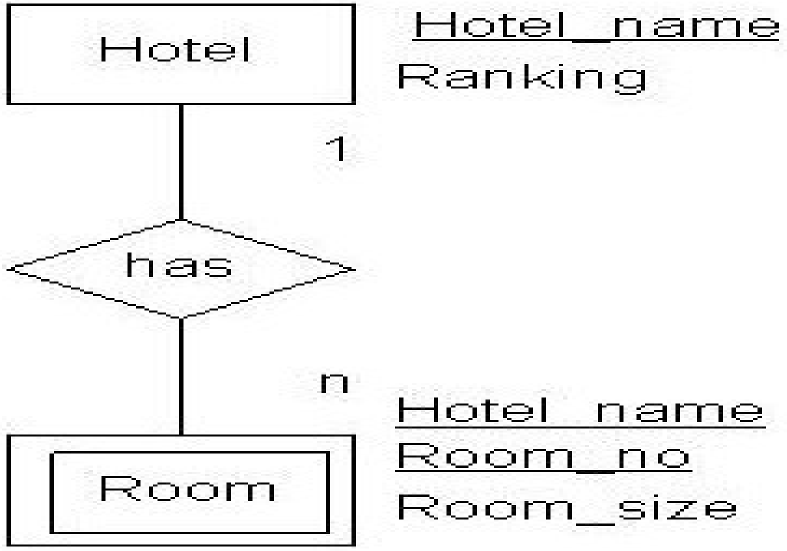
Example:A hotel room must concatenate hotel name for identification.

***Relational Model**:*

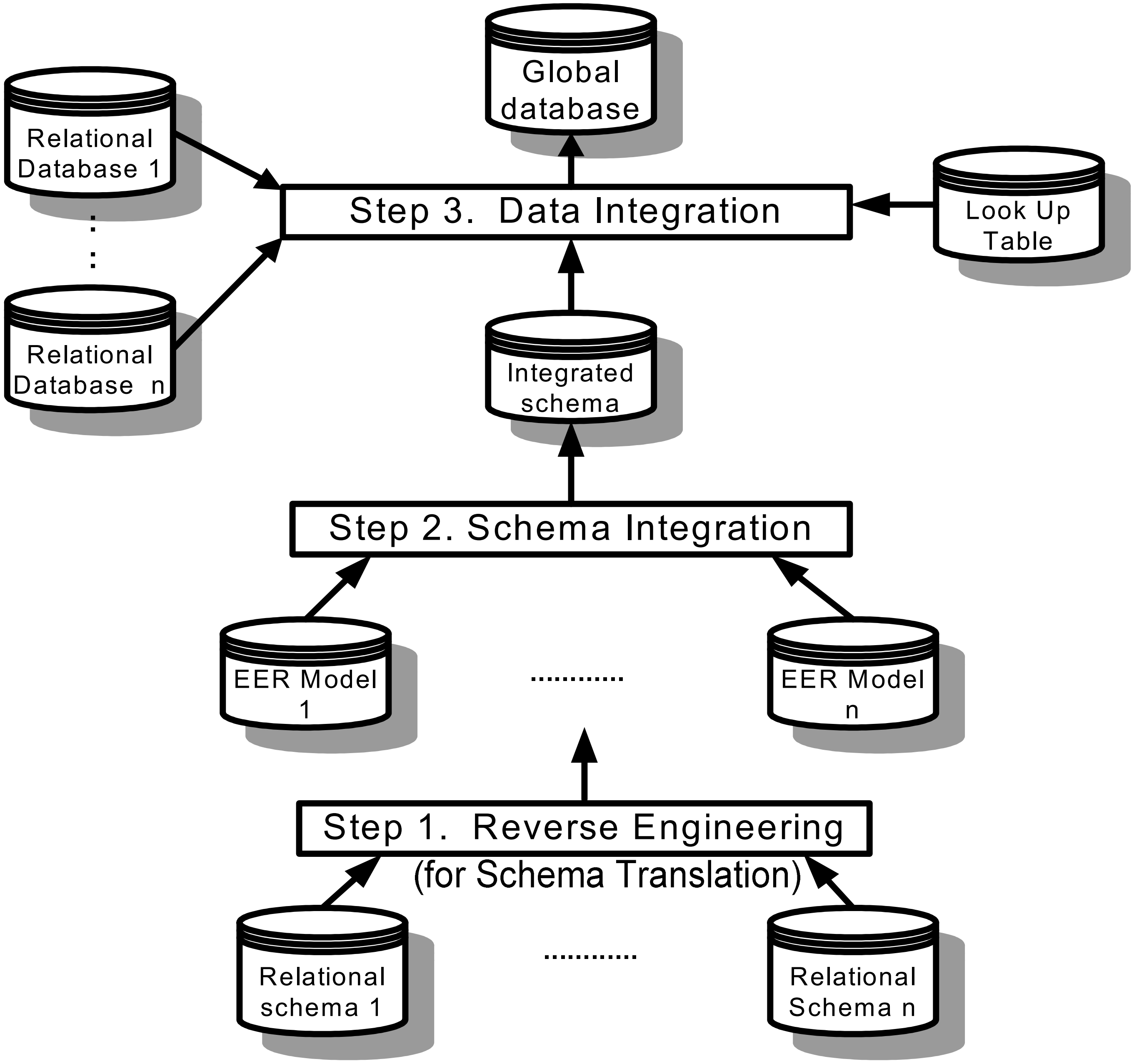
Relation Hotel (Hotel_name, Ranking)
Relation Room (*Hotel_name, Room_no, Room_size)

Khóa chính của thực thể yếu = thuộc tính phân biệt (Room_no)
+ thuộc tính khóa ngoại (Hotel_name)

Extended entity relationship model



Architecture of multiple databases integration



Reverse engineering relational schema into EER model

Step 1 Defining each relation, key and field:

1) The relations are preprocessed by making any necessary candidate key substitutions as follows:

- *Primary relation: describing entities.*
  - *Primary relation - Type 1 (PR1). Đây là quan hệ mà khóa chính không chứa khóa của quan hệ khác.*
  - *Primary relation - Type 2 (PR2). Là quan hệ mà khóa chính chứa khóa của **MỘT** quan hệ khác.*

- Secondary relation: *là quan hệ mà khóa chính được hình thành đầy đủ hoặc một phần từ khóa chính của các quan hệ khác*.
  - *S*econdary relation - Type 1 (SR1). Nếu khóa của SR được hình thành từ khóa chính của quan hệ PR thì là SR1 (thường khóa chính của SR được cấu thành đầy đủ từ khóa chính các quan hệ khác)
  - Secondary relation - Type 2 (SR2). Secondary relations that are not of Type 1.
- Key attribute - Primary (KAP): Là một thuộc tính trong khóa chính của một quan hệ và cũng là khóa ngoại của quan hệ khác.
- Key attribute - General (KAG): Tất cả thuộc tính khóa chính khác trong quan hệ SR mà không phải loại KAP.

- Foreign key attribute (FKA): đây là non-primary-key attribute của một quan hệ chính (PR) và đồng thời là khóa ngoại của một quan hệ khác.
- Nonkey attribute (NKA). The rest of the non-primary-key attribute.

Step 2 Map each PR1 into entity

- For each Type 1 primary relation (PR1), define a corresponding entity type and identify it by the primary key. Its nonkey attributes map to the attributes of the entity types with the corresponding domains.

Step 3 Map each PR2 into a subclass entity or weak entity

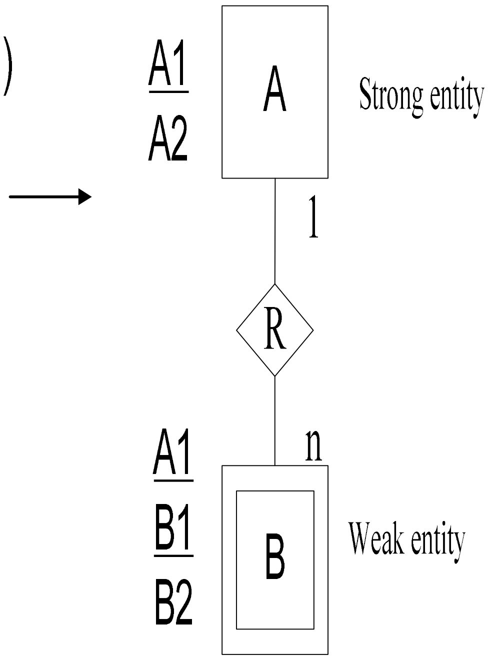
Case 1:

Relational schema

PR1: Relation A (A1, A2)

PR2: Relation B (*A1, B1, B2)

EER model



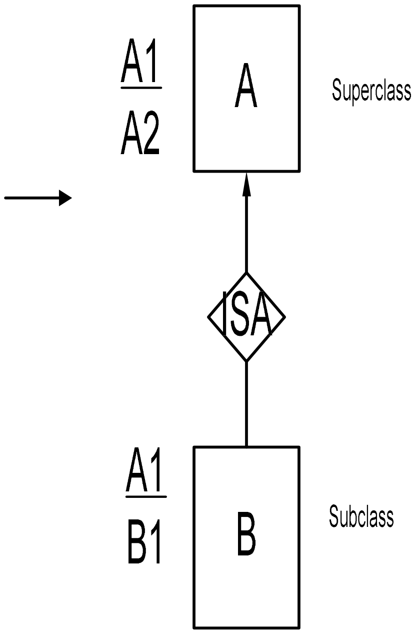
CASE 2:

Relational schema

PR1: Relation A (A1, A2)

PR2: Relation B (*A1, B1)

EER model



Step 4 Map SR1 into binary/n-ary relationship

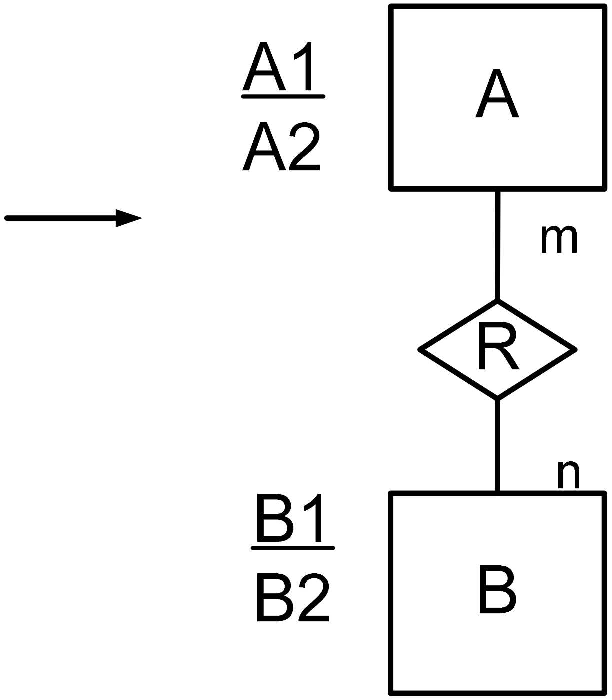
Relational schema

PR1: Relation A (A1, A2)

PR1: Relation B (B1, B2)

SR1: Relation AB (*A1, *B1)

EER model

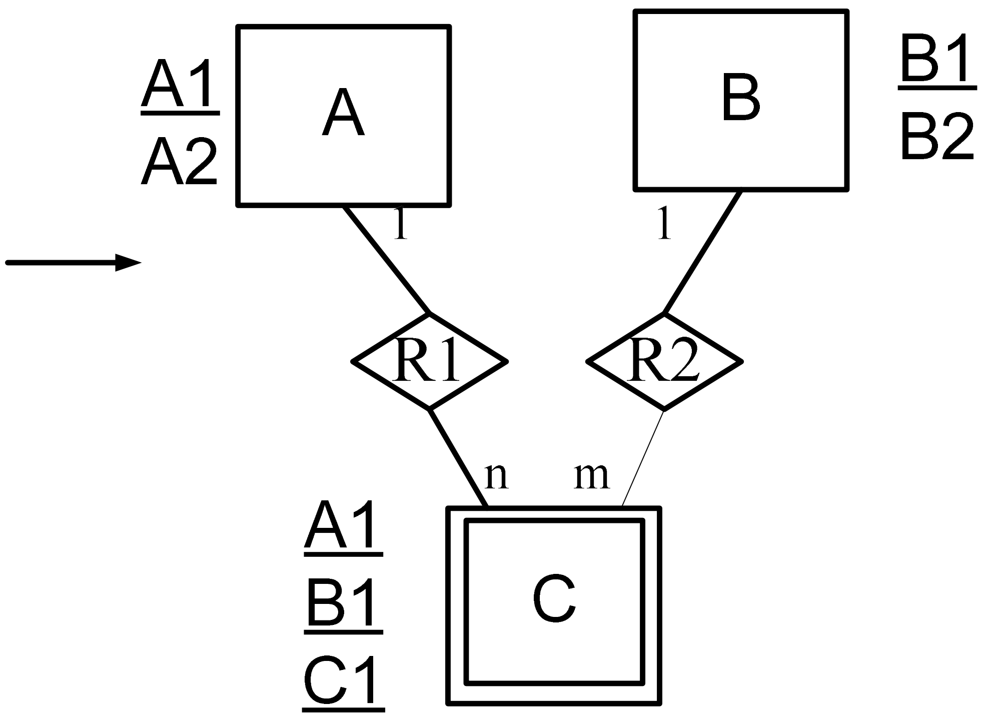


Step 5 Map SR2 into binary/n-ary relationship

Relational schema

PR1: Relation A (A1, A2)
PR1: Relation B (B1, B2)
SR2: Relation C (*A1, *B1, C1)

EER model



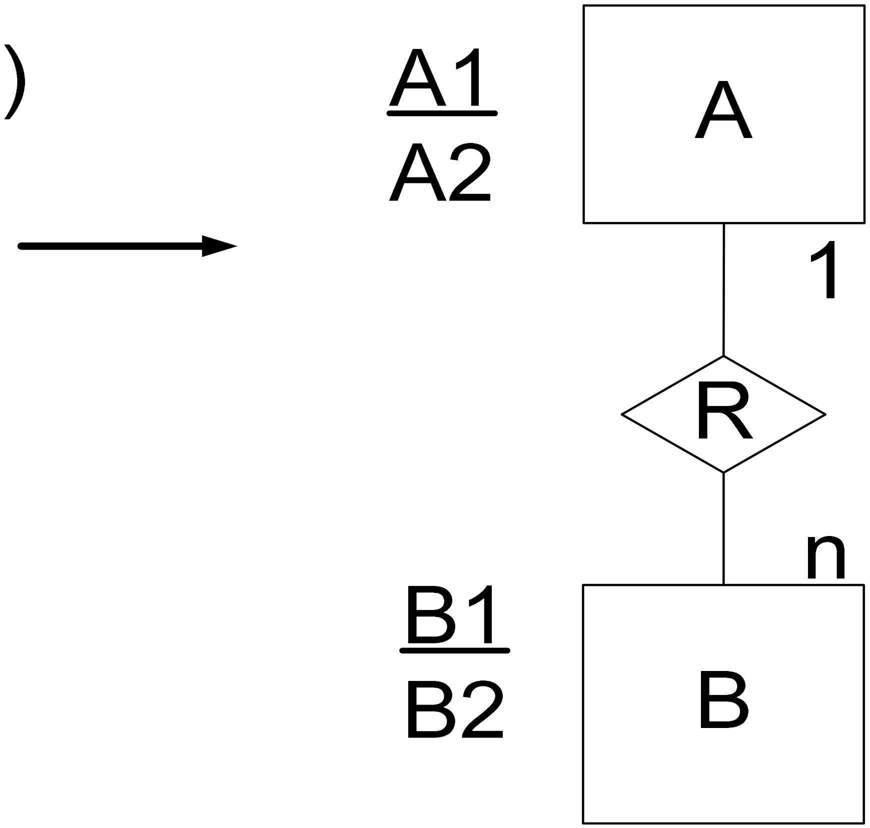
Step 6 Map each FKA into relationship

Relational schema

EER model

PR1: Relation A (A1, A2)

PR1: Relation B (B1, B2, *A1)



Step 7 Map inclusion dependency into semantic

If IDs have been derived between two relations, relation A with a as primary key and b’ as foreign key, relation B with b as primary key and a’ as foreign key, then

Case 1. Given ID: $a' \subseteq a$, then entity A is in 1:n relationship with entity B.

Case 2. Given ID: $a' \subseteq a$, and $b' \subseteq b$, then entity A is in 1:1 relationship with entity B.

Case 3. Given ID: $a' \subseteq a$, and $b' \subseteq b$, and a’b’ is a composite key, then entity A is in m:n relationship with entity B.

Step 8 Draw EER model.

Draw the derived EER model as a result of above steps.

An university enrollment system:

Relations:

Department (Dept#, Dept_name,)	Khoa
Instructor (*Dept#, Inst_name, Inst_addr)	Giảng viên
Course (Course#, Course_location)	Khóa học
Prerequisite (Prer#, Prer_title, *Course#)	Điều kiện
Student (Student#, Student_name)	Sinh viên
Section (*Dept#, *Inst_name,*Course#, Section#)	Lớp
Grade (*Dept#, *Inst_name, *Course#, *Section#, *Student#, Grade)	Điểm

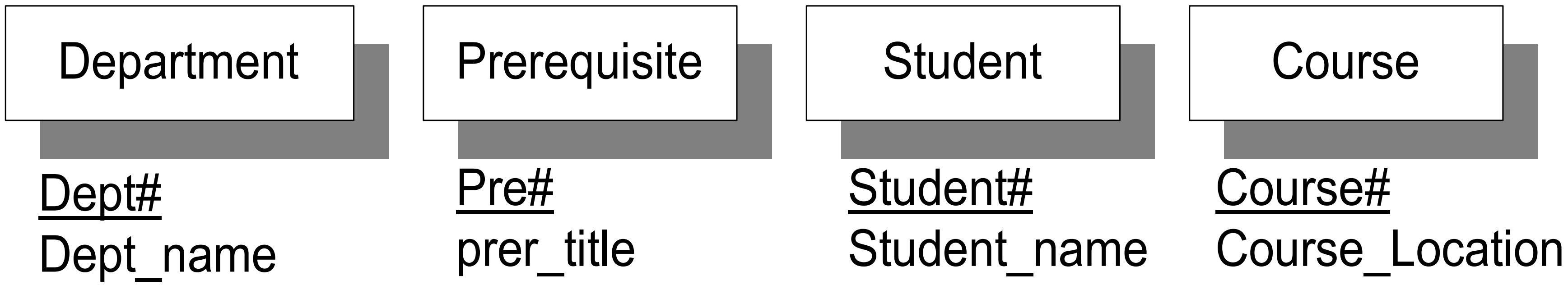
Relations and attributes classification table

Relation	Rel	Primary-	KAP	KAG	FKA	NKA
Name	Type	Key				
DEPT	PR1	Dept#				Dept_name
INST	PR2	Dept# Inst_name	Dept#	Inst_name		Inst_addr
COUR	PR1	Course#				Course_location
STUD	PR1	Student#				Stud_name
PREP	PR1	Prer			Course#	Prer_title
SECT	SR2	Course# Dept# Section# Inst_name	Course# Dept# Inst_name	Section#		Inst_name
GRADE	SR2	Inst_name Course# Student# Dept# Section#	Inst_name Course# Student# Dept# Section#			Grade

Giải thích các quan hệ PR, SR:

- Department, Course, Prerequisite, Student là PR1 vì primary key không chứa khóa của quan hệ khác.
- Instructor là PR2 vì primary key có chứa khóa của quan hệ khác (*Dept#)*
- Section là SR vì khóa chính được hình thành từ khóa chính của các quan hệ Instructor và Course. Không phải SR1 vì có cả thuộc tính Section#, không phải khóa chính của các quan hệ PR trên => Section là SR2
- Grade là SR vì khóa chính được hình thành từ khóa chính của các quan hệ Section và Student. Không phải SR1 vì có chứa khóa chính từ quan hệ Section không phải PR mà là SR => Grade là SR2

Step 2. Map each PR1 into entity



Department

Dept#
Dept_name

Prerequisite

Pre#
prer_title

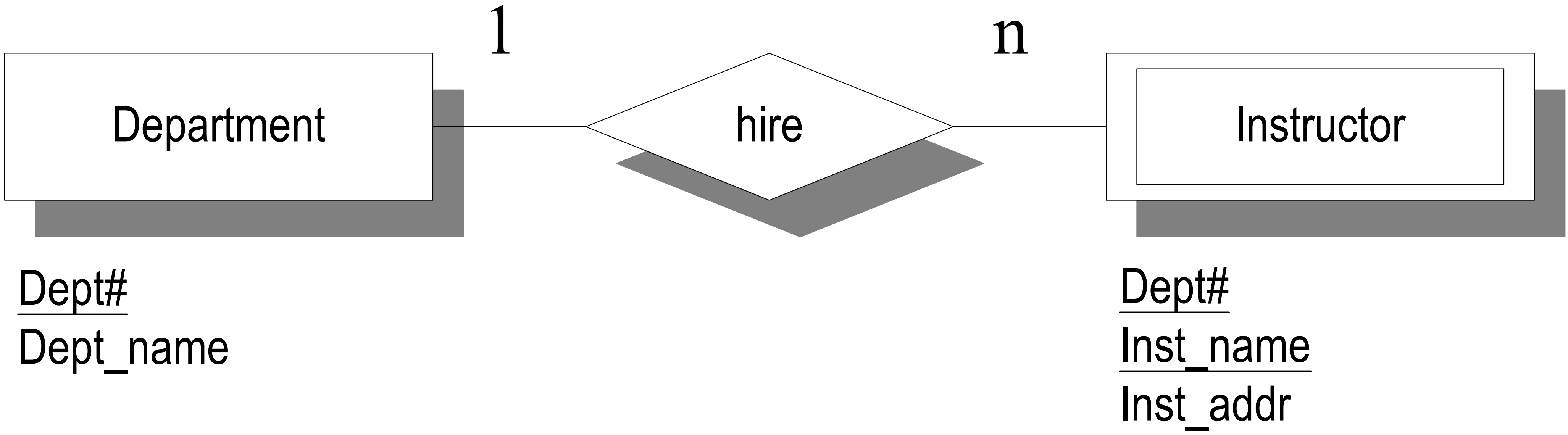
Student

Student#
Student_name

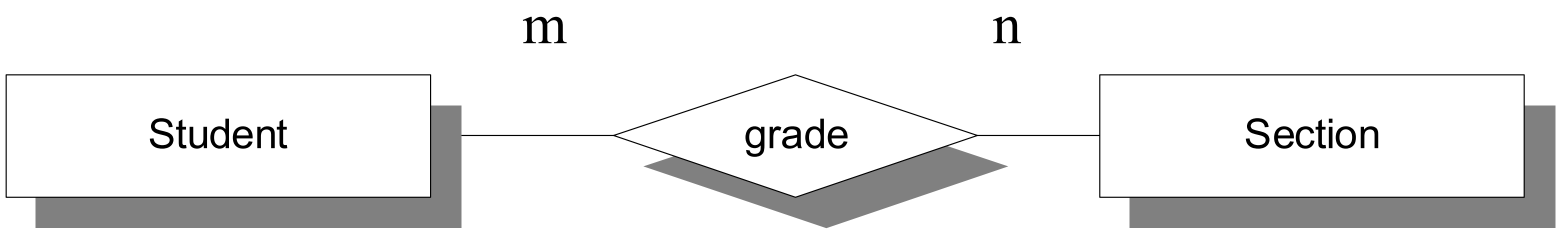
Course

Course#
Course_Location

Step 3. Map each PR2 into weak entity.



Step 4. Map SR1 into binary/n-ary relationship.



Student#
Student_name

Grade

Section#

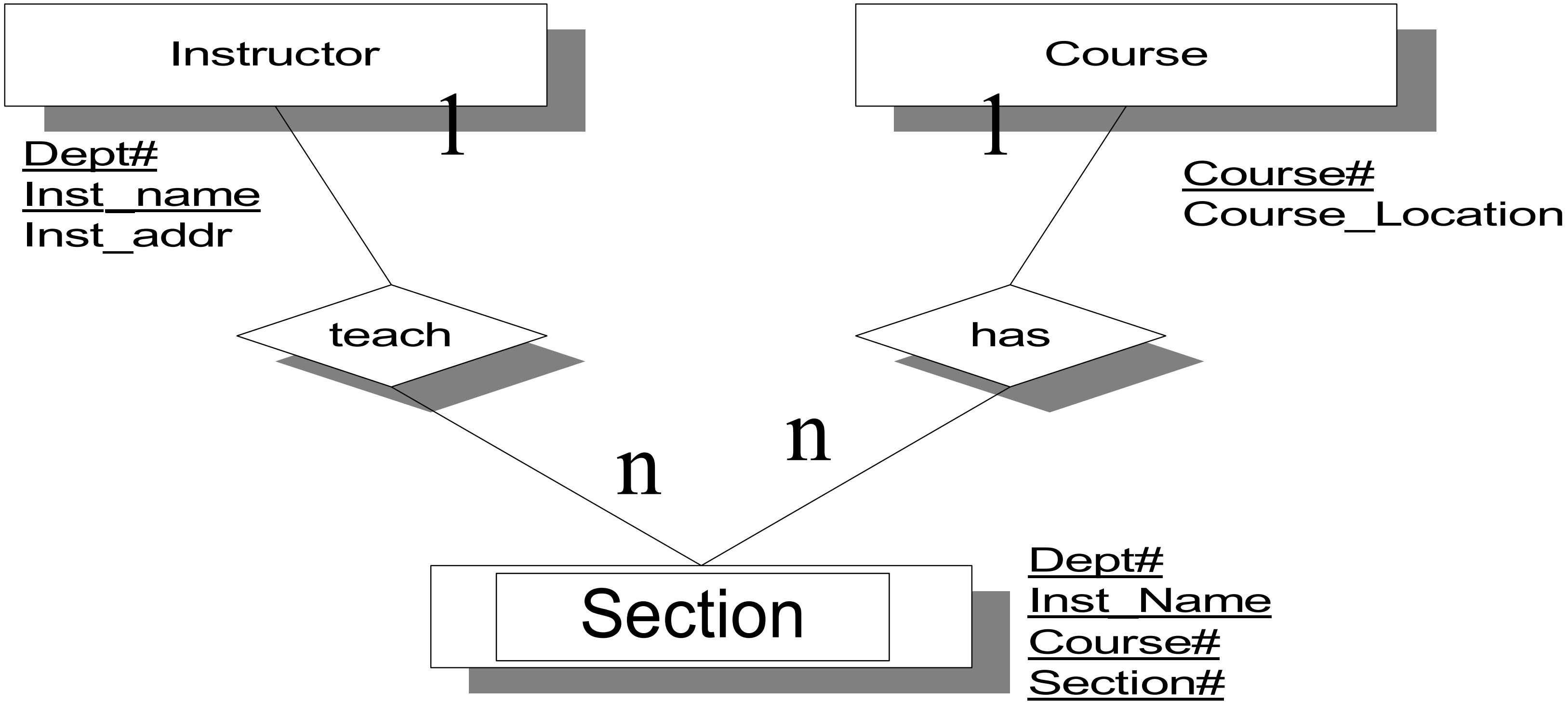
Dept#

Inst_name

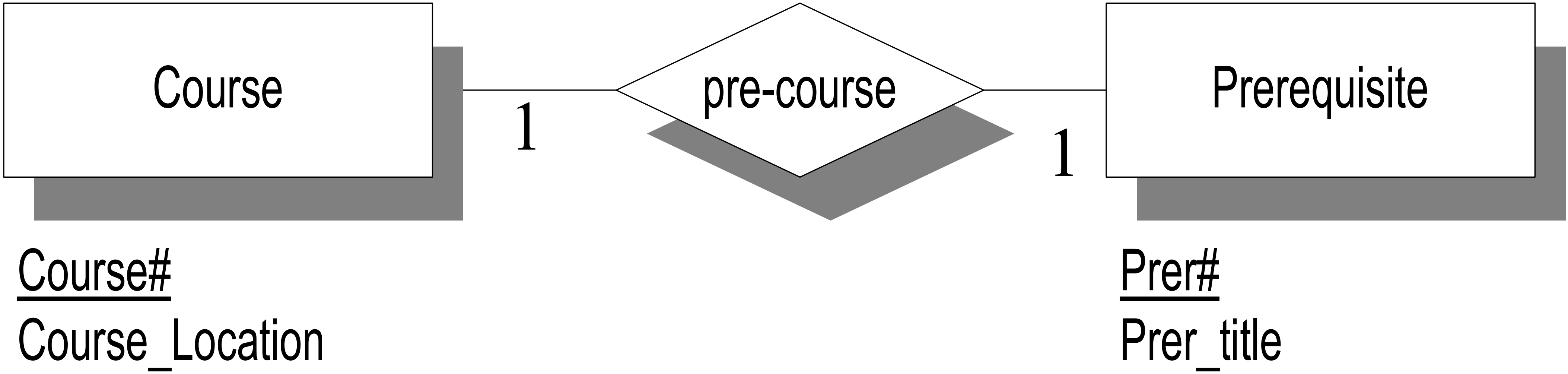
Course#

Section#

Step 5. Map SR2 into binary/n-ary relationship



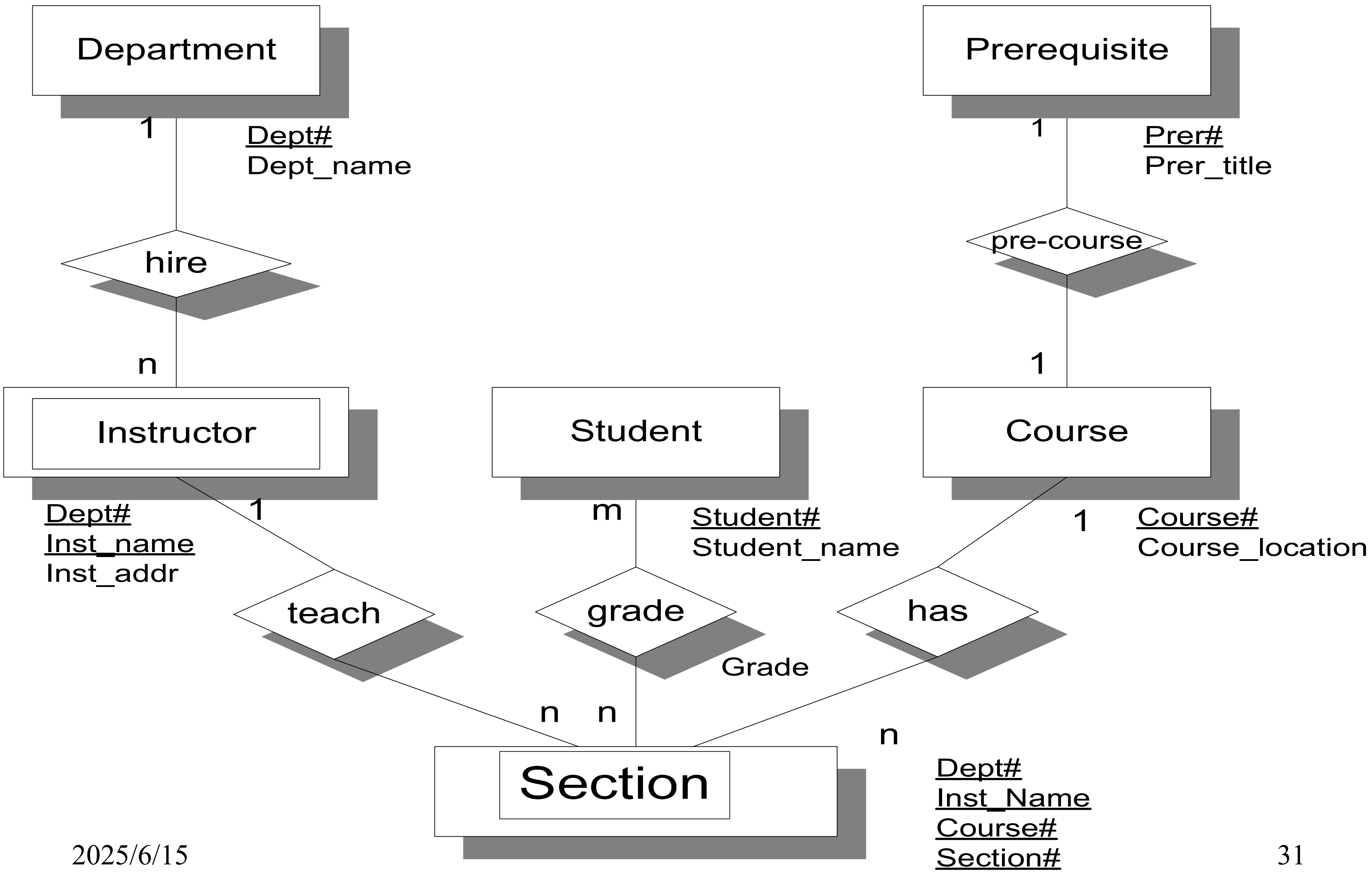
Step 6. Map each FKA into relationship



Step 7. Map each inclusion dependency into semantics (binary/n-ary relationship)

Given derived inclusion dependency	Derived Semantics
Instructor.Dept# ⊆ Department.Dept#	n:1 relationship between entities Instructor and Department
Section.Dept# ⊆ Department.Dept#	
Section.Inst_name⊆ Instructor.Inst_name	
Section.Course# ⊆ Course.Course#	1:n relationship between entities Instructor and Section and between Course and Section.
Grade.Dept# ⊆ Section.Dept#	
Grade.Inst_name ⊆ Section.Inst_name	
Grade.Course# ⊆ Section.Course#	
Grade.Student# ⊆ Student.Student#	m:n relationship between relationship Section and entity Student.
Prerequisite.Course# ⊆ Course.Course#	
Course.Prer# ⊆ Prerequisite.Prer#	1:1 relationship between Course and Prerequisite

Step 8. Draw EER model.



CHÚ Ý

- Về lược đồ EER bên trên và cách phân tích từ Step 4 đang theo hướng cho rằng Grade là SR1.
- Để đúng theo lý thuyết thì nên để Grade là SR2, do đó phải chỉnh lại tất cả từ Step 4, bài trên chỉ mang tính tham khảo.

Reading assignment

Chapter 3 Schema Translation in “Information Systems Reengineering and Integration” by Joseph Fong, published by Springer Verlag, 2006, pp. 115-121.

Review question 2

What are the major differences between Generalization and Categorization in terms of data volume (data occurrences) in their related superclass entity/entities and subclass entity/entities?

Is there any special case such that these two data semantics can be overlapped?

Tutorial Question 2

In a reverse engineering approach, translate the following relational schema to an Entity-relationship model.

Order (Order_code, Order_type, Our_reference, Order_date, Approved_date, *Head, *Supplier_code)
Supplier (Supplier_code, Supplier_name)
Product (Product_code, Product_description)
Department (Department_code, Department_name)
Head (Head, *Department_code, Title)
Order_Product (*Order_code, *Product_code, Qty, Others, Amount)
Note (*Order_code, Sequence#, Note)

where underlined are primary keys and prefixed with '*' are foreign keys.

Relations and attributes classification table

Relation Name	Rel Type	Primary key	KAP	KAG	FKA	NKA
ODR	PR1	Order_code			Head Supplier_code	Order_type, Our_reference, Order_date, Approved_date
SUPP	PR1	Supplier_code				Supplier_name
PRD	PR1	Product_code				Product_description
DPM	PR1	Department_code				Department_name
HEAD	PR1	Head			Department_code	Title
ODR_PRD	SR1	Order_code Product_code	Order_code Product_code			Qty Others Amount
NOTE	PR2	Order_code Sequence#	Order_code	Sequence#		Note

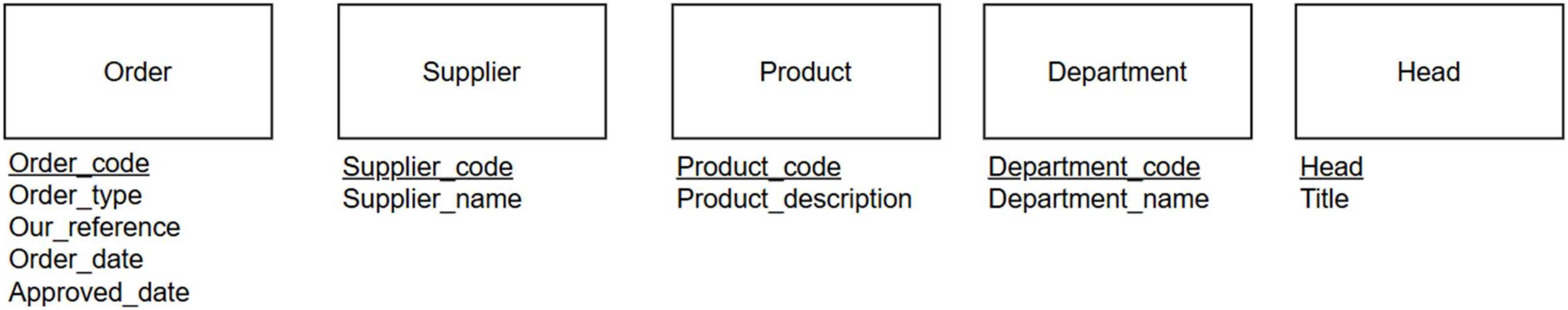
Giải thích các quan hệ PR, SR:

- Order, Supplier, Product, Department, Head là PR1 vì primary key không chứa khóa của quan hệ khác.

- Note là PR2 vì primary key có chứa khóa của quan hệ khác (*Order_code*)*

- Order_Product là SR vì khóa chính được hình thành từ khóa chính của các quan hệ Order và Product => Là SR1 vì khóa của quan hệ này chỉ hình thành từ khóa chính của các quan hệ PR trên.

Step 2. Map each PR1 into entity



Order

Order_code
Order_type
Our_reference
Order_date
Approved_date

Supplier

Supplier_code
Supplier_name

Product

Product_code
Product_description

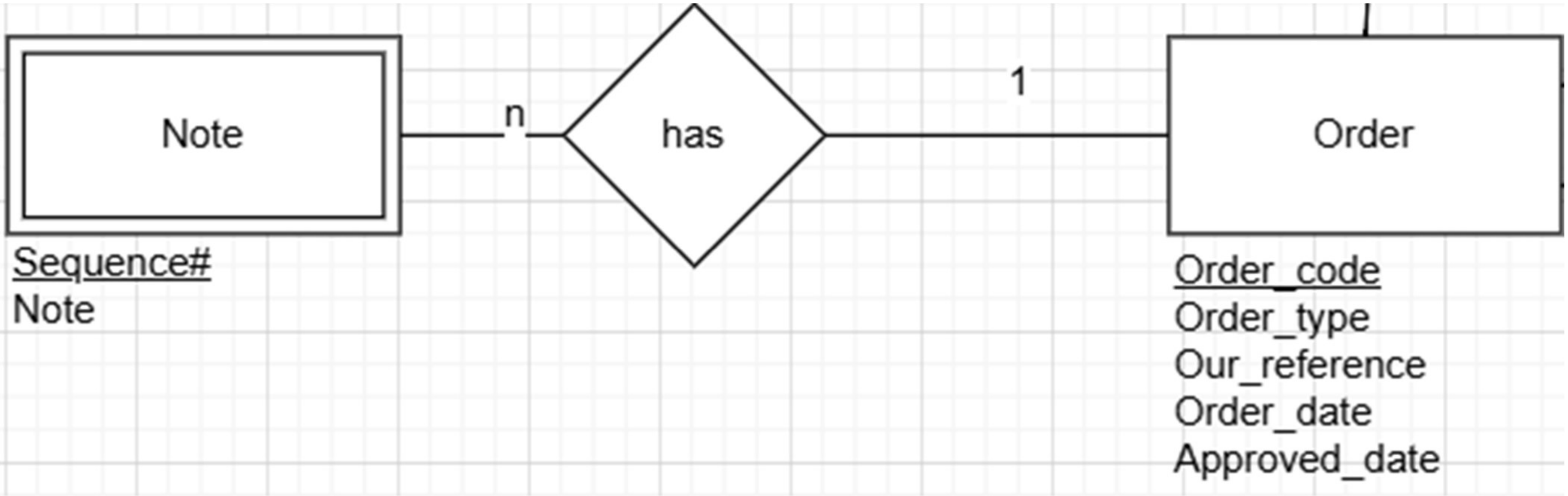
Department

Department_code
Department_name

Head

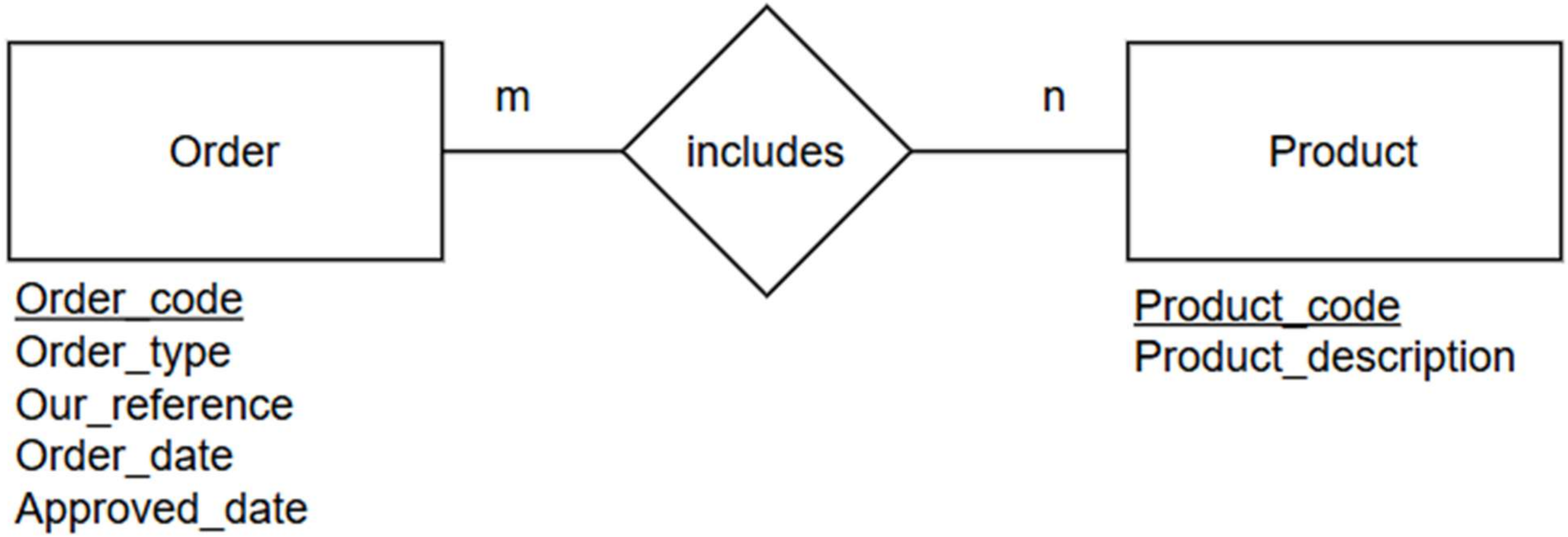
Head
Title

Step 3. Map each PR2 into weak entity.

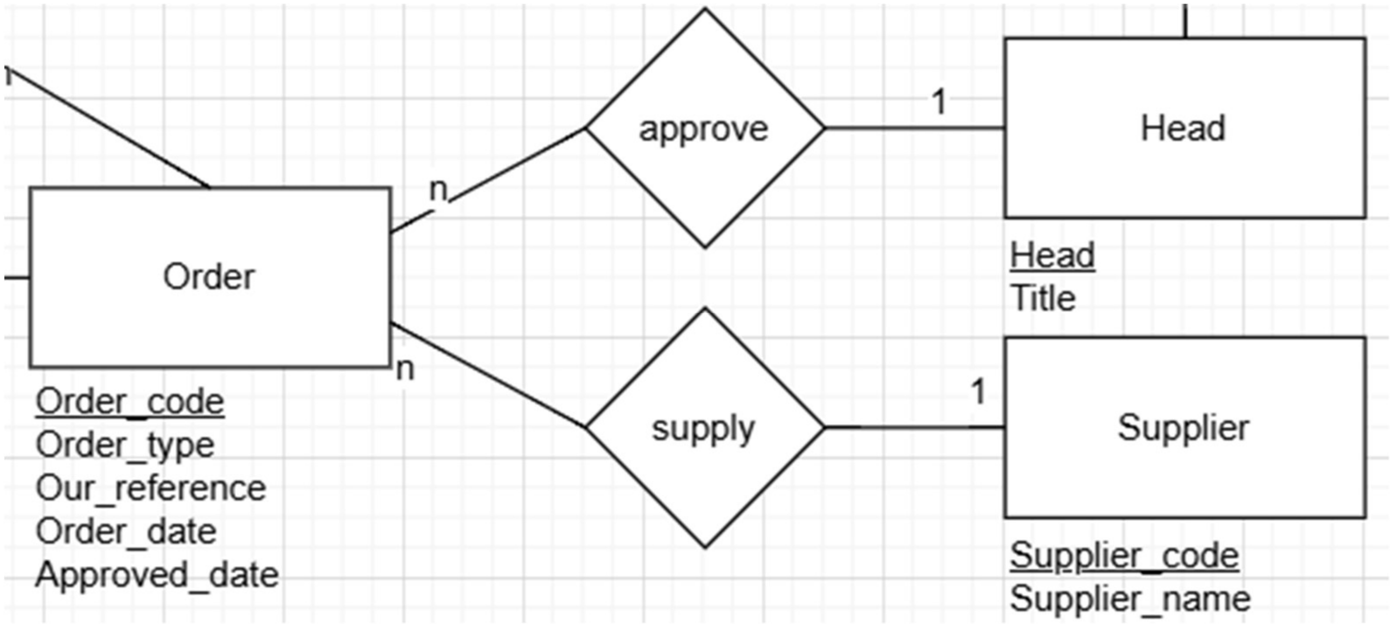


Order_code được bỏ trong Note vì đã được thể hiện qua quan hệ “has”

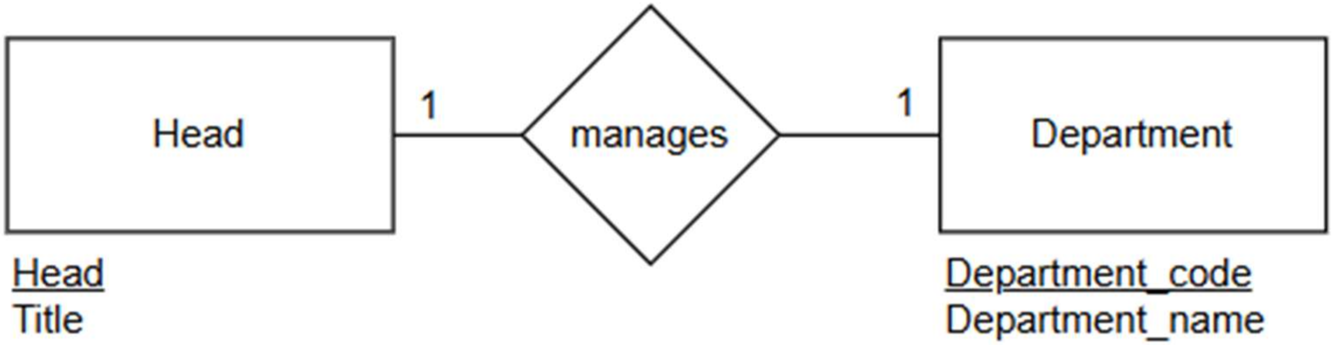
Step 4. Map SR1 into binary/n-ary relationship.



Step 5. Map each FKA into relationship



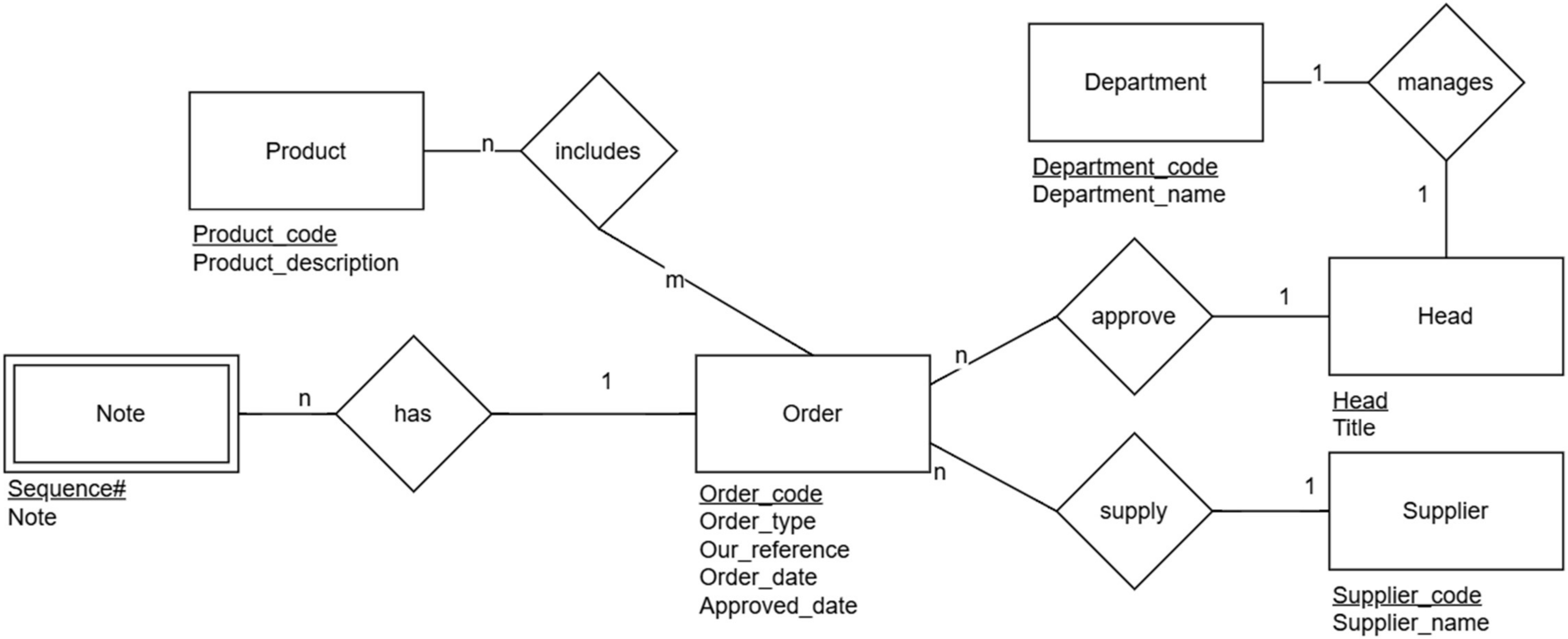
Head - Department về mặt ngữ nghĩa nên để 1-1



Step 6. Map each inclusion dependency into semantics (binary/n-ary relationship)

Given derived inclusion dependency	Derived Semantics
Note.Order_code ⊆ Order.Order_code	n:1 relationship between entities Note and Order
Order_Product.Order_code ⊆ Order.Order_code Order_Product.Product_code ⊆ Product.Product_code	m:n relationship between entity Product and entity Order
Order.Head ⊆ Head.Head	n:1 relationship between Order and Head
Order.Supplier_code ⊆ Head. Supplier_code	n:1 relationship between Order and Supplier
Order.Department_code ⊆ Department.Department_code	n:1 relationship between Head and Department (nhưng về ngữ nghĩa nên để 1:1)

Step 7. Draw EER model.



Chú ý mặc dù Deparment và Head quan hệ 1-1 nhưng không nhất thiết phải viết Dept_code trong Head vì đã được thể hiện trong "manages"