Mapping from ER Models to Relation Models

陆伟

College of Software

Database Systems-Design and Application

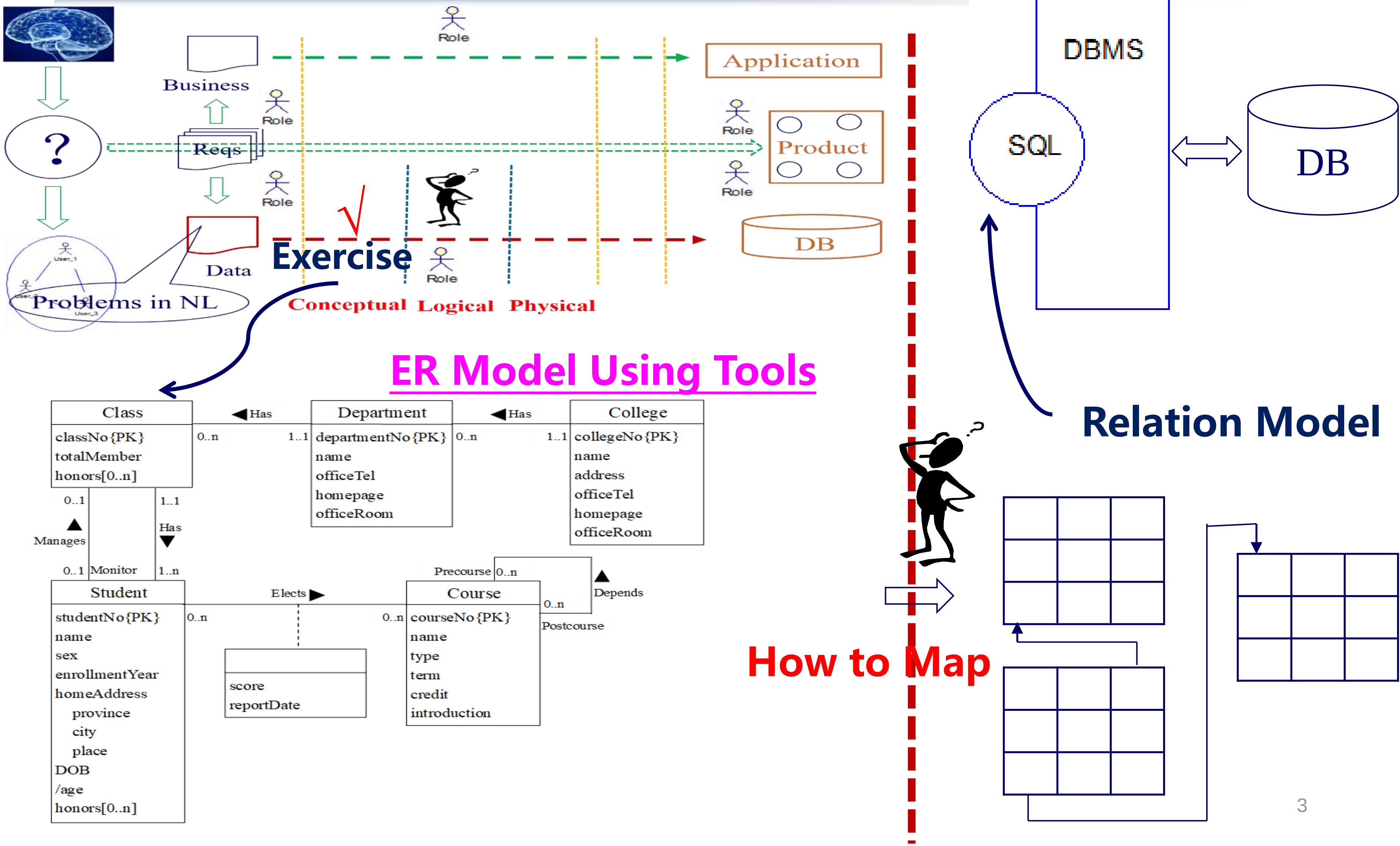
April 29, 2021

Northwestern Polytechnical University

Outline

- Objective and Task
- Elements Comprare between ER Model and Relation Model
- Mapping Algorithm
- Brief Summary of Mappings
- Example
- Summary

Objective and Task



Business

Reqs

Data

Role

Role

Role

Role

Role

Role

Role

?

Problems in NL

User_1

User_2

User_3

Application

Product

DB

√

Exercise

Conceptual Logical Physical

ER Model Using Tools

Class
classNo{PK}
totalMember
honors[0..n]

◀Has 0..n 1..1

Department
departmentNo{PK}
name
officeTel
homepage
officeRoom

◀Has 0..n 1..1

College
collegeNo{PK}
name
address
officeTel
homepage
officeRoom

0..1 ▲ Manages 0..1 Monitor

1..1 Has ▼ 1..n

Student
studentNo{PK}
name
sex
enrollmentYear
homeAddress
province
city
place
DOB
/age
honors[0..n]

Elects ▶ 0..n 0..n

score
reportDate

Course
courseNo{PK}
name
type
term
credit
introduction

Precourse 0..n

▲ Depends

0..n Postcourse

DBMS

SQL

DB

Relation Model

How to Map

Objective and Task

- Objective
  - To careate a set of relation schema and constraints to represent the elements in ER model and satisfy the user's requirement.
- Task
  - How to do? Mapping
  - The steps.

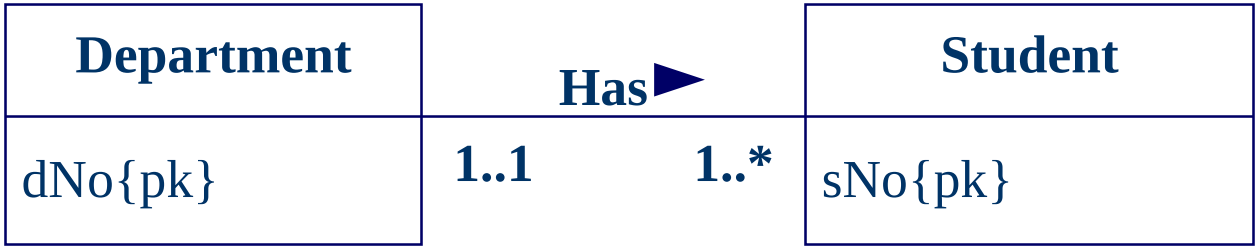
Elements Comprare between ER Model and Relation Model

Elements in ER Model
entity
strong, weak
attribute
simple, composite, mutli-value
relationship
1:1, 1:N, M:N
N-ary
recursive
value set
key attribute

Elements in Relation Model
relation(table)
attribute(column)
foreign key
domain
primary key (alternate key)

Elements Comprare between ER Model and Relation Model

- Discuss every element in ER model



D-S(dNo,dName, officeRoom, homepage,sNo, sName, sex, age)

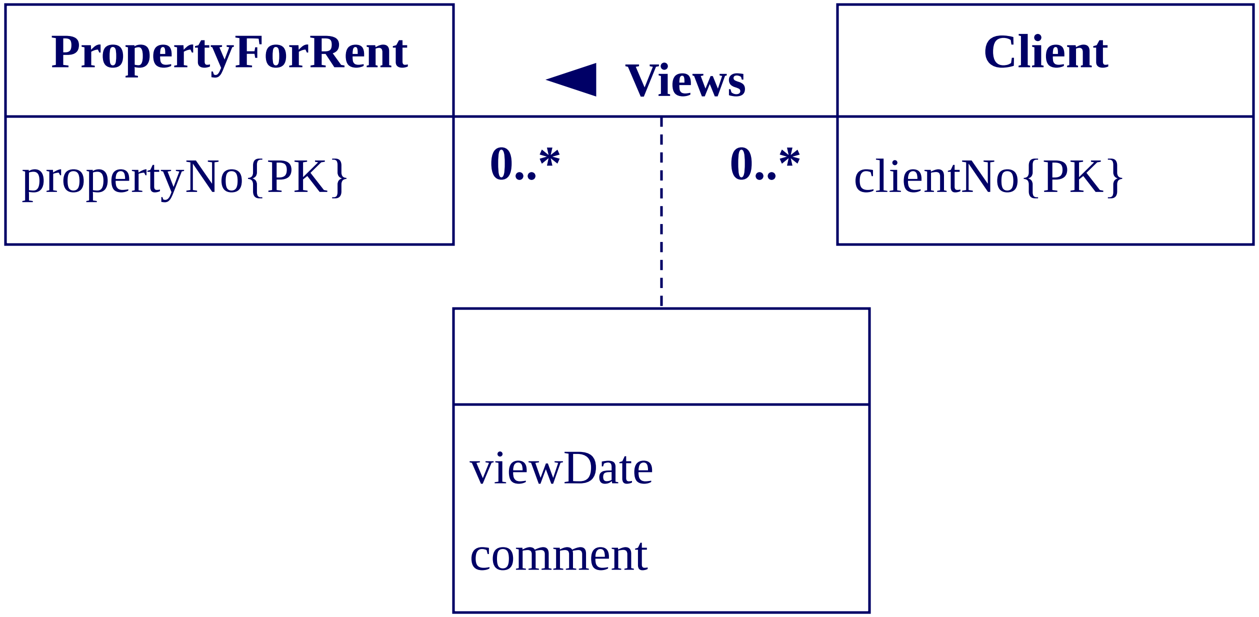
D(dNo,dName, officeRoom, homepage)
S(sNo, sName, sex, age,dNo)

Mapping Algorithm

- (Ⅰ) Remove features difficult to mapping directly
    - (1) Remove many-to-many (*:*) binary relationship types;
    - (2) Remove many-to-many (*:*) recursive relationship types;
    - (3) Remove complex relationship types;
    - (4) Remove multi-valued attributes.

Mapping Algorithm

- (1) Remove many-to-many (*:*) binary relationship types

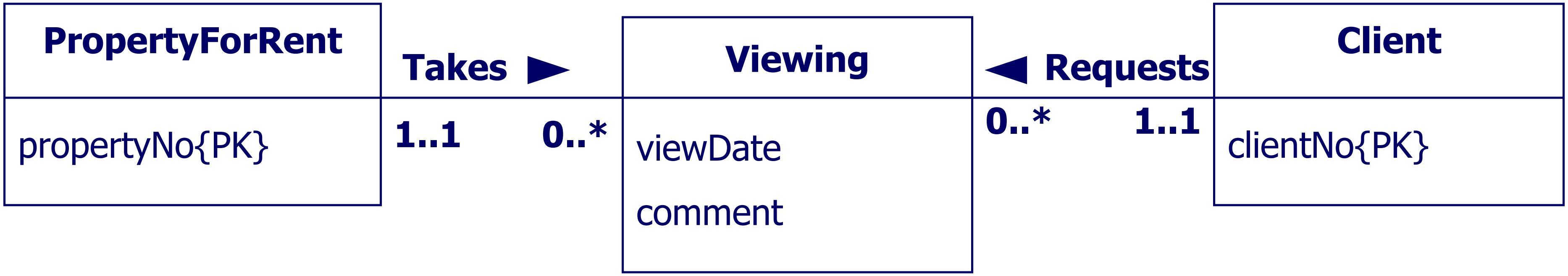


Mapping Algorithm

- (1) Remove many-to-many (*:*) binary relationship types
  - If a many-to-many (*:*) relationship is present in the conceptual data model, we could decompose this relationship to identify an intermediate entity (a weak entity).
  - The *:* relationship is replaced with two one-to-many (1:*) relationship to the newly identified entity.

Mapping Algorithm

- (1) Remove many-to-many (*:*) binary relationship types



PropertyForRent

propertyNo{PK}

Takes ►

1..1 0..*

Viewing

viewDate

comment

◄ Requests

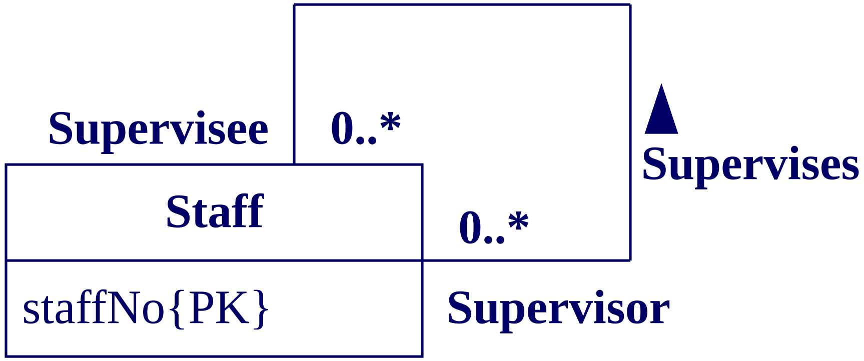
0..* 1..1

Client

clientNo{PK}

Mapping Algorithm

- (2) Remove many-to-many (*:*) recursive relationship types

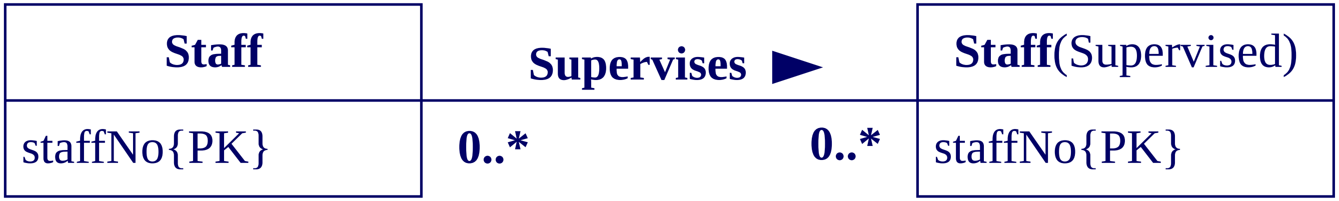


Mapping Algorithm

- (2) Remove many-to-many (*:*) recursive relationship types
  - If a many-to-many (*:*) recursive relationship is represented in the conceptual data model, we could decompose this relationship to identify an intermediate entity (a weak entity).
  - The *:* relationship is replaced with two one-to-many (1:*) relationship to the newly identified entity.

Mapping Algorithm

- (2) Remove many-to-many (*:*) recursive relationship types



Staff

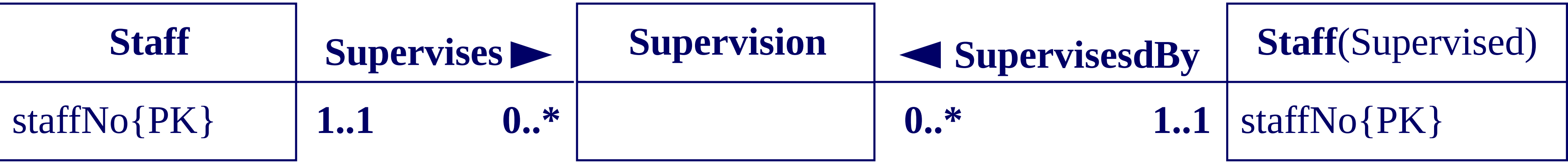
staffNo{PK}

Supervises ►

0..* **0..***

Staff(Supervised)

staffNo{PK}



Staff

staffNo{PK}

Supervises ►

1..1 **0..***

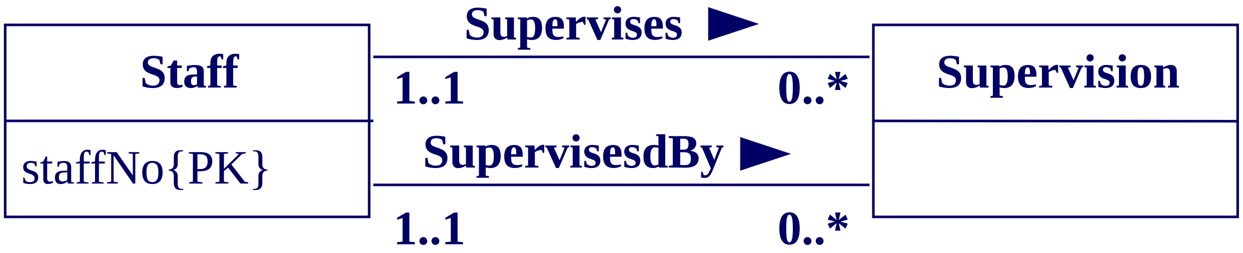
Supervision

◄ **SupervisesdBy**

0..* **1..1**

Staff(Supervised)

staffNo{PK}



Staff

staffNo{PK}

Supervises ►

1..1 **0..***

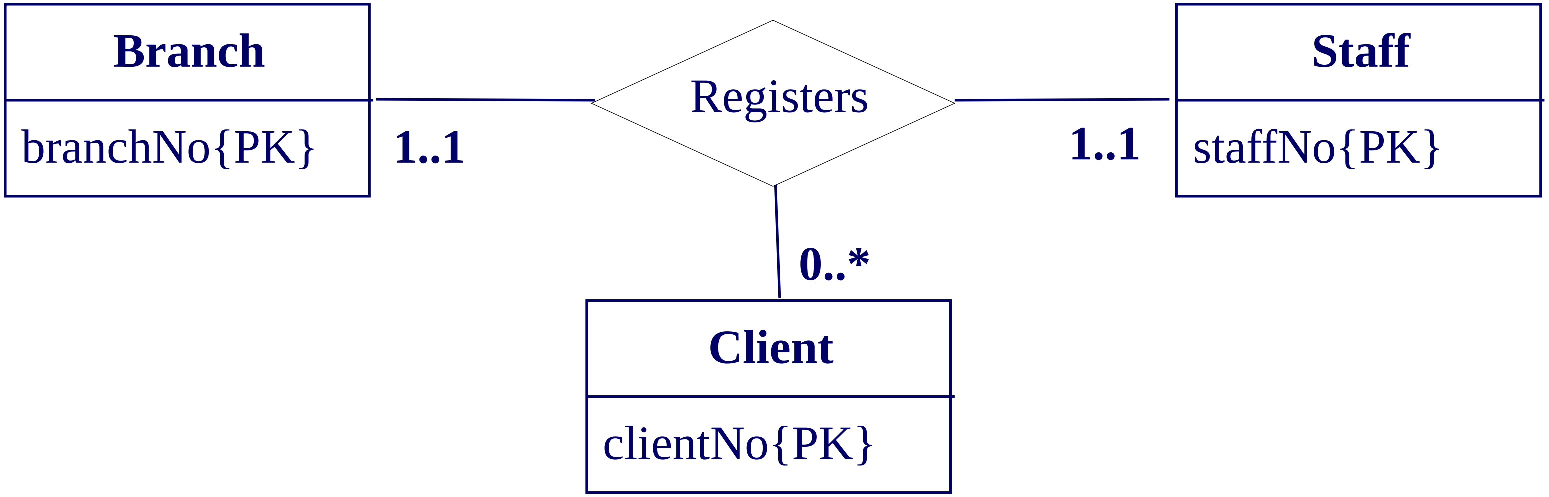
SupervisesdBy ►

1..1 **0..***

Supervision

Mapping Algorithm

- (3) Remove complex relationship types
  - A complex relationship is a relationship between three or more entity types.

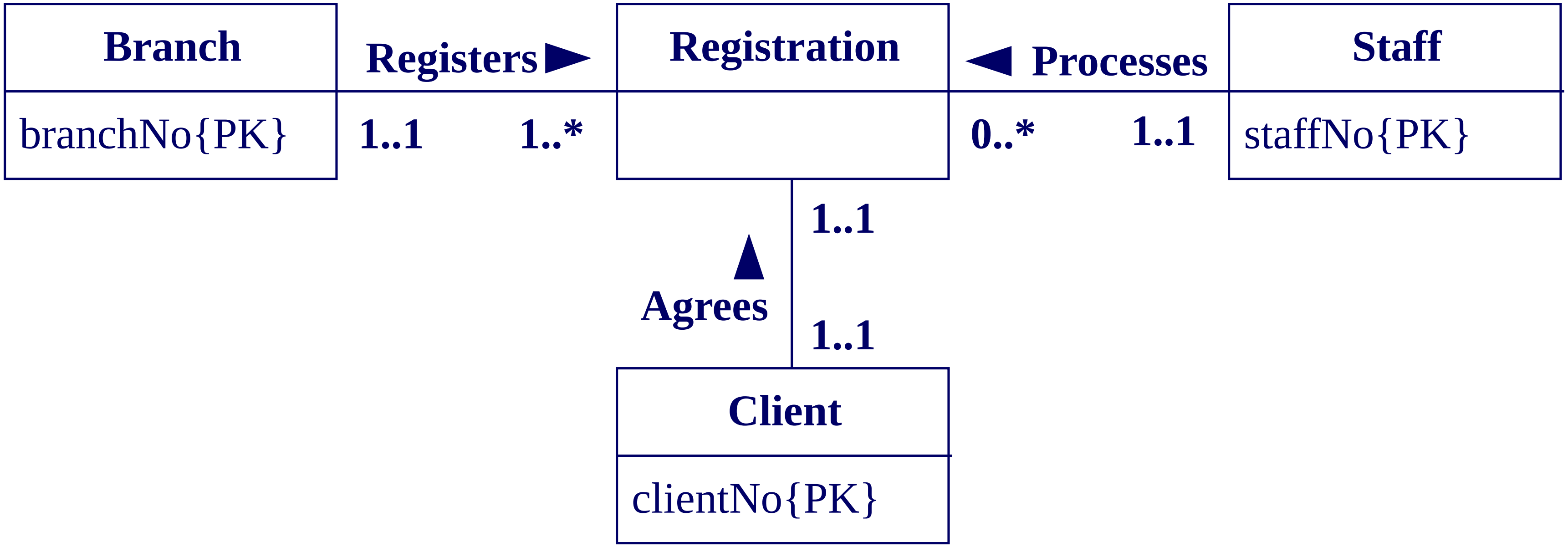


Mapping Algorithm

- (3) Remove complex relationship types
  - If a complex relationship is represented in the conceptual data model, we could decompose this relationship to identify an intermediate entity.
  - The complex relationship is replaced with the required number of 1:* (binary) relationships to the newly identified entity.

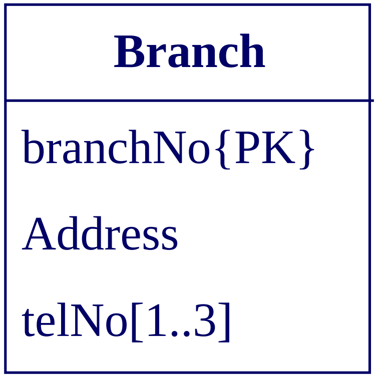
Mapping Algorithm

- (3) Remove complex relationship types



Mapping Algorithm

- (4) Remove multi-valued attributes



Branch

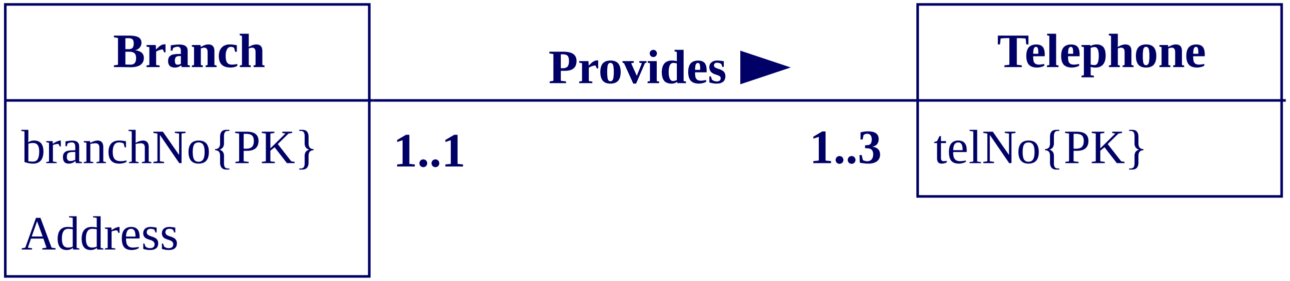
branchNo{PK}

Address

telNo[1..3]

Mapping Algorithm

- (4) Remove multi-valued attributes
  - If a multi-valued attribute is present in the conceptual data model, we could decompose this attribute to identify an entity.



Branch

branchNo{PK}

Address

Provides ►

1..1

1..3

Telephone

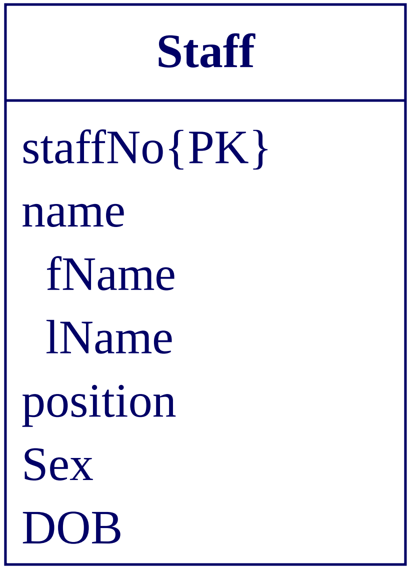
telNo{PK}

Mapping Algorithm

- (Ⅱ) Mapping simple elements remain in ER model to relation sechma and constraints
  - (1) strong entity types
  - (2) weak entity types
  - (3) one-to-many (1:*) binary relationship types
  - (4) one-to-one (1:1) binary relationship types
  - (5) one-to-one (1:1) recursive relaitonship types
  - (6) superclass/subclass relationship types

Mapping Algorithm

- (1) strong entity types
  - For each strong entity in the data model, create a relation that includes all the simple attributes of that entity. For composite attributes, include only the constituent simple attributes.


Staff

staffNo{PK}
name
  fName
  lName
position
Sex
DOB

Staff (staffNo, fName, lName, position, sex, DOB)

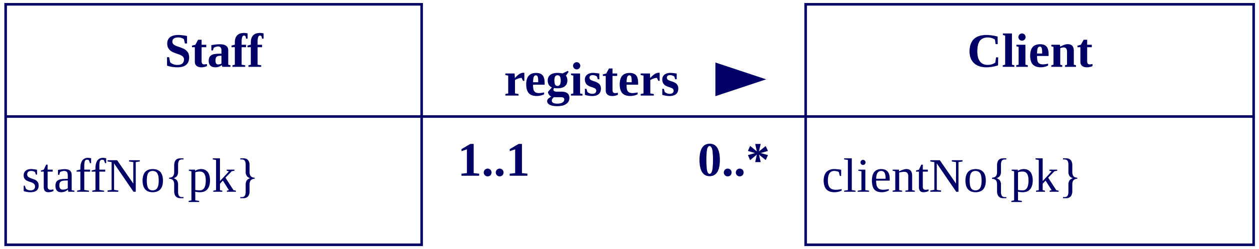
Primary Key： staffNo

Mapping Algorithm

- (2) weak entity types
  - For each weak entity in the data model, create a relation that includes all the simple attributes of that entity.
  - The primary key of a weak entity is partially or fully derived from each owner entity and so the identification of the primary key of a weak entity cannot be made until after all the relationships with the owner entities have been mapped.

Mapping Algorithm

- (3) one-to-many (1:*) binary relationship types



Staff

staffNo{pk}

registers ►

1..1

0..*

Client

clientNo{pk}

Staff (staffNo, fName, lName, position, sex, DOB)

Primary Key： staffNo

Client(clientNo,fName,lName,telNo,staffNo)

Primary key: clientNo

Alternate key: telNo

Foreign key: staffNo references staff(staffNo)

Mapping Algorithm

- (3) one-to-many (1:*) binary relationship types
  - For each 1:* binary relationship, the entity on the ‘one side’ of the relationship is designated as the parent entity and the entity on the ‘many side’ is designated as the child entity.
  - Post a copy of the primary key attribute(s) of the parent entity into the relation representing the child entity, to act as a foreign key.

Mapping Algorithm

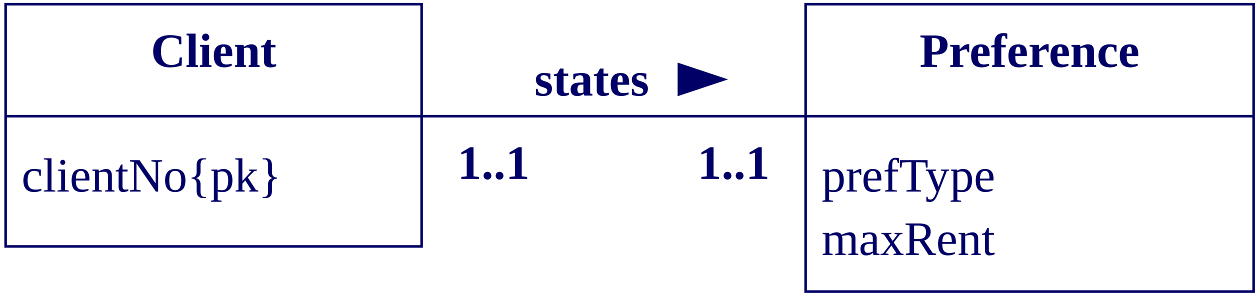
- (4) one-to-one (1:1) binary relationship types
  - Creating relations to represent a 1:1 relationship is slightly more complex as the cardinality cannot be used to help identify the parent and child entities in a relationship.
  - Instead, the participation constraints are used to help decide how to do.

Mapping Algorithm

- (4) one-to-one (1:1) binary relationship types
  - (a) mandatory participation on both sides
    - In this case we should combine the entities involved into one relation and choose one of the primary keys of the original entities to be the primary key of the new relation, while the other (if one exists) is used as an alternate key.
    - If the 1:1 relationship has one or more attributes, these attributes should also be included in the merged relation.

Mapping Algorithm

- (4) one-to-one (1:1) binary relationship types
  - (a) mandatory participation on both sides



Client

clientNo{pk}

states ►

1..1 **1..1**

Preference

prefType
maxRent

Client(clientNo,fName,lName,telNo,prefType,maxRent,staffNo)

Primary key: clientNo

Alternate key: telNo

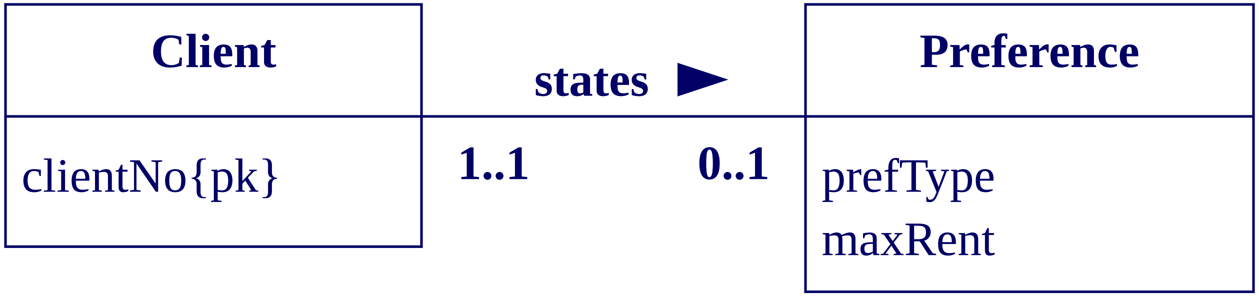
Foreign key: staffNo references staff(staffNo)

Mapping Algorithm

- (4) one-to-one (1:1) binary relationship types
  - (b) mandatory participation on one side
    - The entity that has optional participation in the relationship is designated as the parent entity, and the entity that has mandatory participation in the relationship is designated as the child entity.
    - If the relaitonship has one or more attributes, these attributes should follow the posting of the primary key to the child relation.

Mapping Algorithm

- (4) one-to-one (1:1) binary relationship types
  - (b) mandatory participation on one side



Client

clientNo{pk}

states ►

1..1

0..1

Preference

prefType

maxRent

Client(clientNo,fName,lName,telNo,staffNo)

Primary key: clientNo

Alternate key: telNo

Foreign key: staffNo references staff(staffNo)

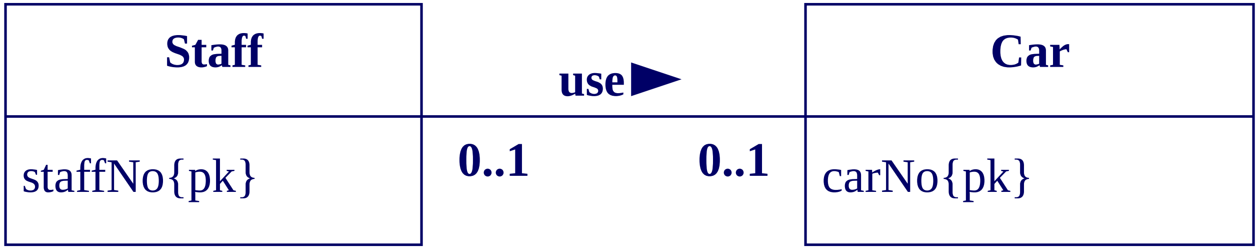
Preference(clientNo,prefType,maxRent)

Primary key: clientNo

Foreign key: clientNo references Client(clientNo)

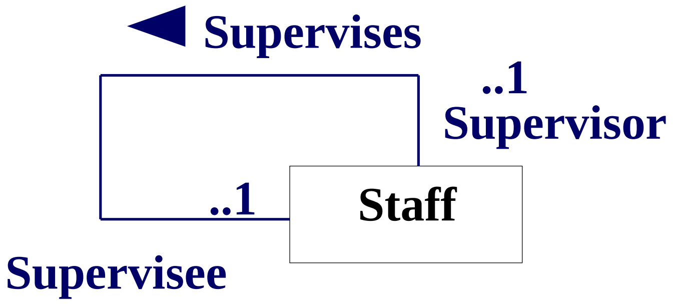
Mapping Algorithm

- (4) one-to-one (1:1) binary relationship types
    - (c) optional participation on both sides
        - In this case the designation of the parent and child entities is arbitrary unless we can find out more about the relationship that can help a decision to be made.
        - Discuss



Mapping Algorithm

- (5) one-to-one (1:1) recursive relaitonship types
  - For a 1:1 recursive relaitonship, follow the rules for participation as described above for a 1:1 relationship.

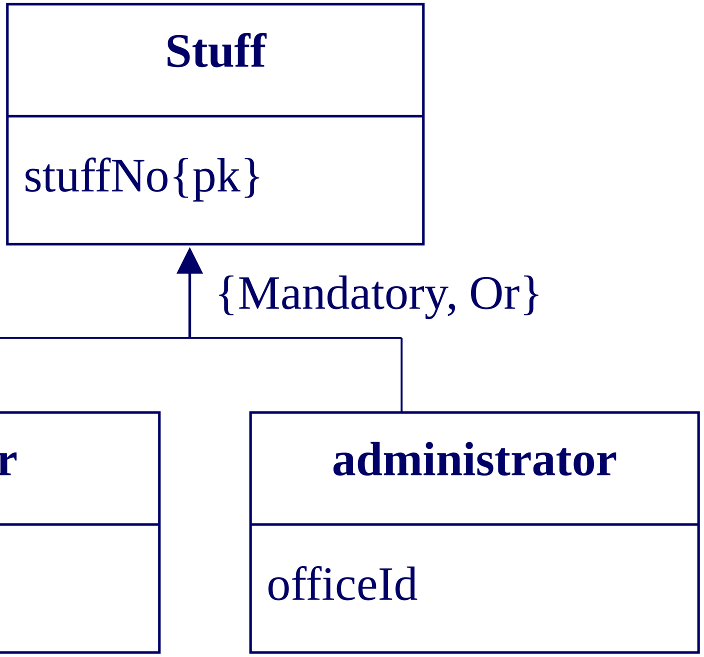


Mapping Algorithm

- (5) one-to-one (1:1) recursive relaitonship types
  - For a 1:1 recursive relationship with mandatory participation on both sides, represent the recursive relationship as a single relation with two copies of the primary key.
  - For a 1:1 recursive relaitonship with mandatory participation on only one side, we have the option to create a single relation with two copies of the primary key, or to create a new relation to represent the relation.
  - For a 1:1 recursive relationship with optional participation on both sides, create a new relation.

Mapping Algorithm

- (6) superclass/subclass relationship types



Mapping Algorithm

- (6) superclass/subclass relationship types

Participation constraint	**Disjoint constraint**	**Relations required**
M	And	Single relation (with one or more discriminators to distinguish the type of each tuple)
O	And	Two relations:one relation for superclass and one relation for all subclasses (with one or more discriminators to distinguish the type of each tuple)
M	Or	Many relations:one relation for each combined superclass/subclass
O	Or	Many relations:one relation for superclass and one for each subclass

Brief summary of mappings

实体/联系	映射
强实体	创建包含所有简单属性的的关系
弱实体	创建包含所有简单属性的关系（主关键字等到每个主实体的联系映射后再确定）
1：*二元联系	将一方实体的主关键字处理为表示多方实体关系的外部关键字
1：1 二元联系	
(a)双方强制参与	组合为一个实体
(b)一方强制参与	将“可选”方实体的主关键字处理为表示“强制”方实体关系的外部关键字
(c)双方可选参与	无进一步消息任选
超类/子类联系	参照超类/子类映射表
*：*二元联系、复杂联系	创建一个关系表示该联系，该关系包含该联系的所有属性。参与联系的所有实体的主关键字作为该关系的外部关键字
多值属性	创建一个新关系表示多值属性，并将主实体的主关键字作为该关系的外部关键字

Example

- Example(30-41页)

Summary

- In this chapter you should have learned:
  - Mapping ER Model to Relational Model
  - Rules for Mapping Different Elements in ER Model