Database design

Logical Design

DBMG

Logical Design (1/2)

- Introduction
- Restructuring of the Entity-Relationship schema
- Removing generalizations
- Partitioning of concepts
- Removing multivalued attributes
- Removing composed attributes
- Selection of primary identifiers

Logical Design (2/2)

- Translation into the relational model
  - entity and many-to-many relationships
  - one-to-many relationships
  - one-to-one relationships
  - entity with external identifier
  - ternary relationships

Logical Design

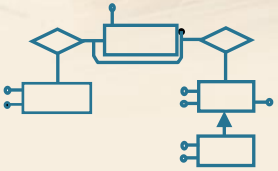
Introduction

Logical Design

- Select a logical model
  - in our case, the relational model
- Goal
  - build a relational schema that correctly and efficiently represents all the information described by the ER schema
- Not just a simple translation
  - simplification of the scheme to make it compatible with the relational model
  - optimization to increase the efficiency of queries

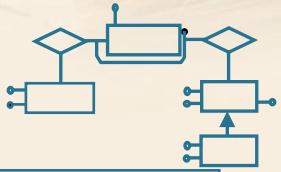
Logical design steps

ER Schema



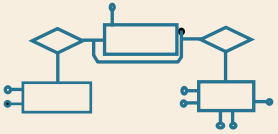
Logical design steps

ER Schema



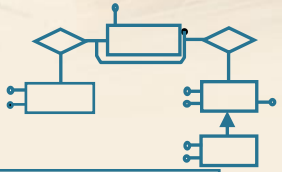
Restructuring the ER Schema

Restructured ER schema



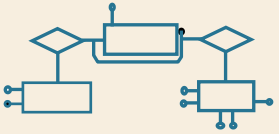
Logical design steps

Schema ER



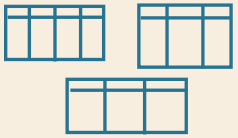
Restructuring the ER Schema

Restructured ER schema



Translation

Logical schema



Logical Design

Restructuring an ER schema

DBMG

ER scheme restructuring

- Implementation aspects
  - This is not a conceptual schema
- Goals
  - Removing costructs for which there is no direct representation in the relational model
  - Optimize data access

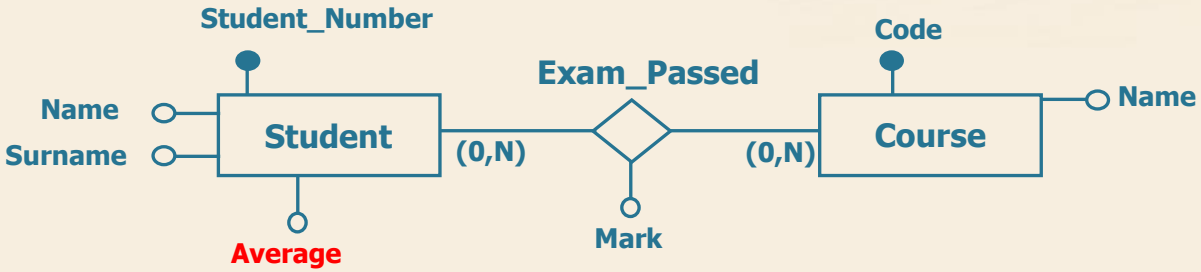
Restructuring tasks

- Analysis of redundancies
- Removing generalizations
- Partitioning and merging of entities and relationships
- Selection of primary identifiers

Analysis of redundancies

- Issue
  - To represent informations that can be derived from other data
  - Decide whether to keep or remove them
- Advantages
  - Speed up and simplify queries
- Disadvantages
  - increased complexity of updates
  - slowing down of updates
  - more storage space required

Redundant attribute: example



- In this schema the attribute Average is redundant
  - It is useful for speeding up queries to calculate studendt's average.
  - if kept, the redundancy indication must be added in the relational schema.

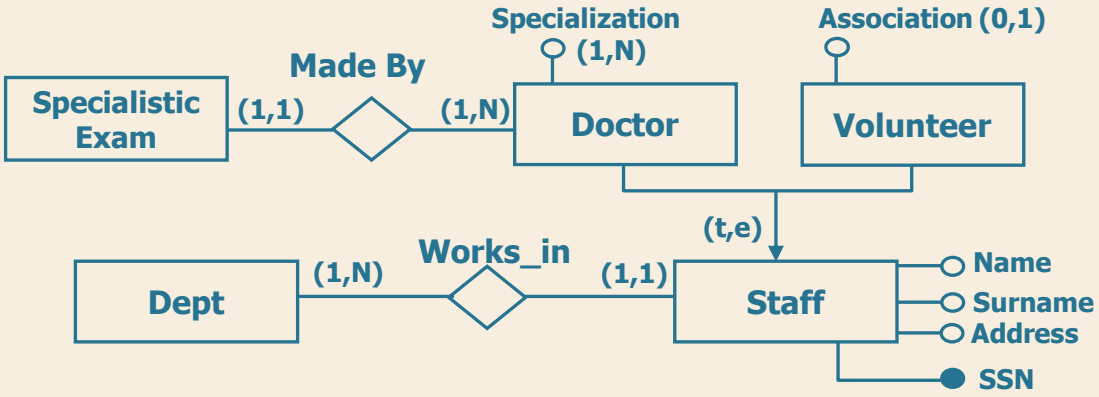
Logical Design

Removing generalizations

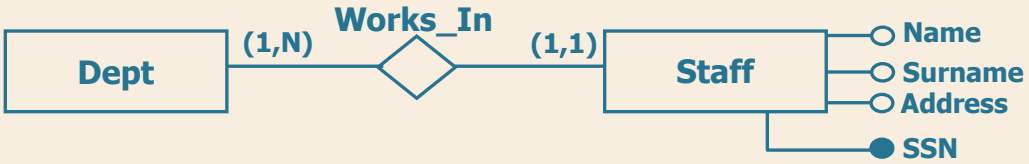
Removing Generalization

- The relational model does not allow the direct representation of generalizations of the ER model
- We need, therefore, to trasform these into entities and relationships
- Possible restructurings methods:
  - Child entities merged into parent entity
  - Parent entity merged into child entities
  - Generalization translated into relatioships

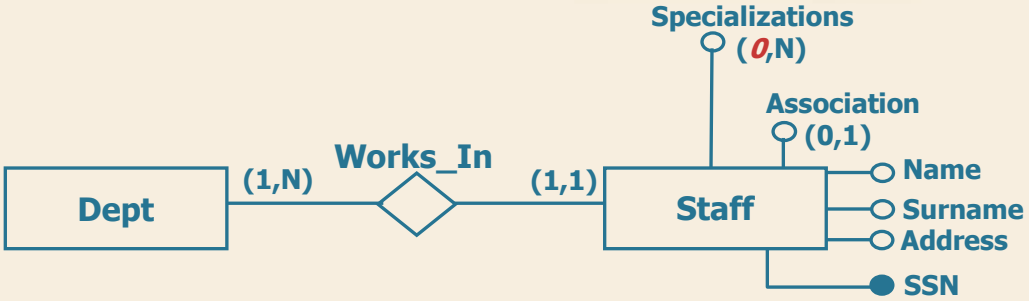
Example


Specialization
(1,N)
Association (0,1)
Made By
Specialistic Exam
(1,1)
(1,N)
Doctor
Volunteer
(t,e)
Works_in
Dept
(1,N)
(1,1)
Staff
Name
Surname
Address
SSN

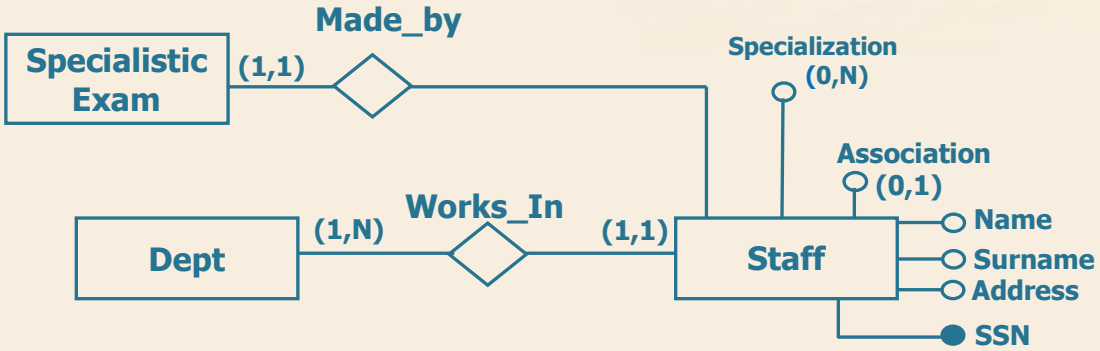
Child->Parent



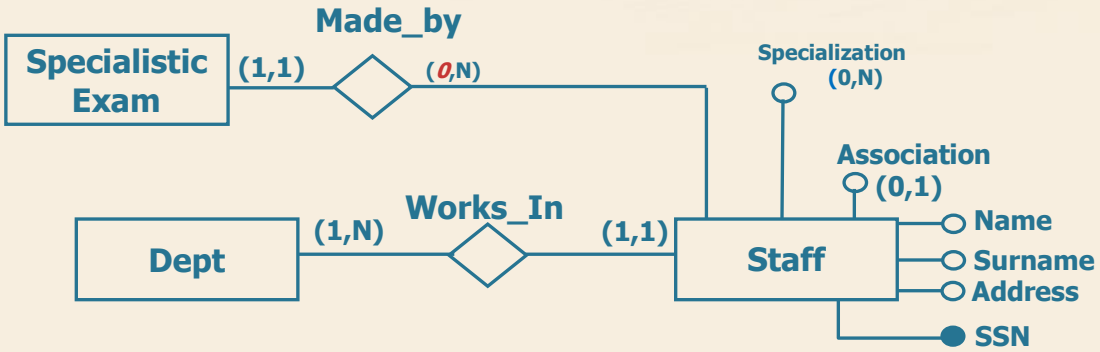
Child entities' attributes



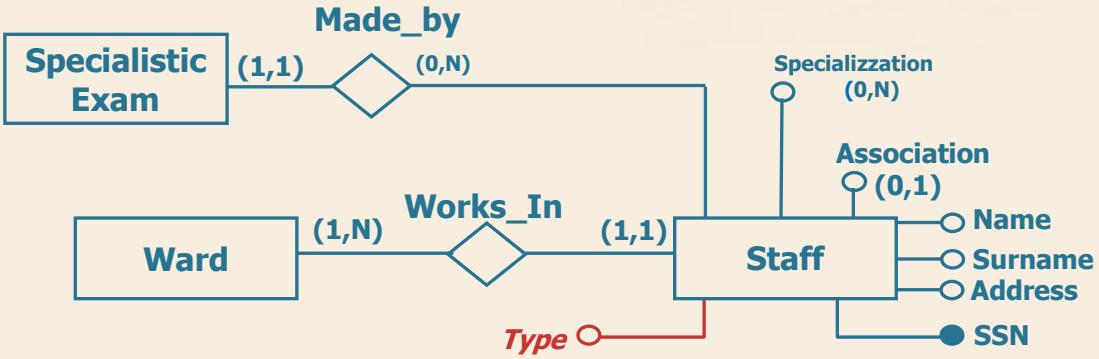
Relations with child entities



Relations with child entities

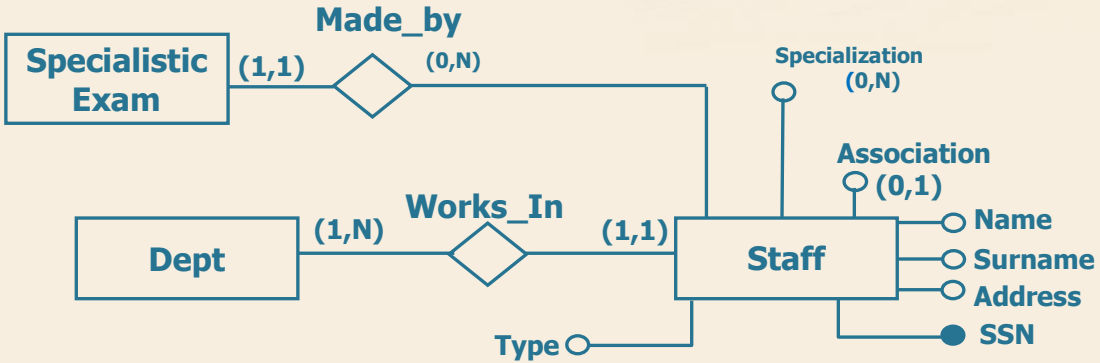


The «Type» attribute



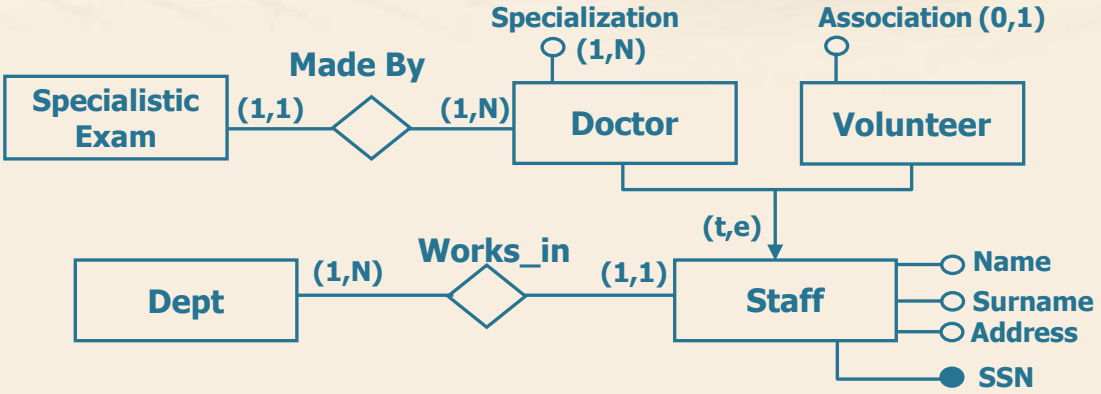
- «Type» indicates the original entity: doctor or volunteer

Child->Parent

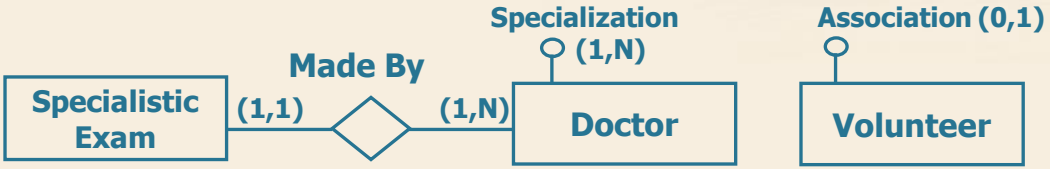


- Always usable
  - in case of overlapped entities, many combinations are possible as Type values, e.g., skier and sailor

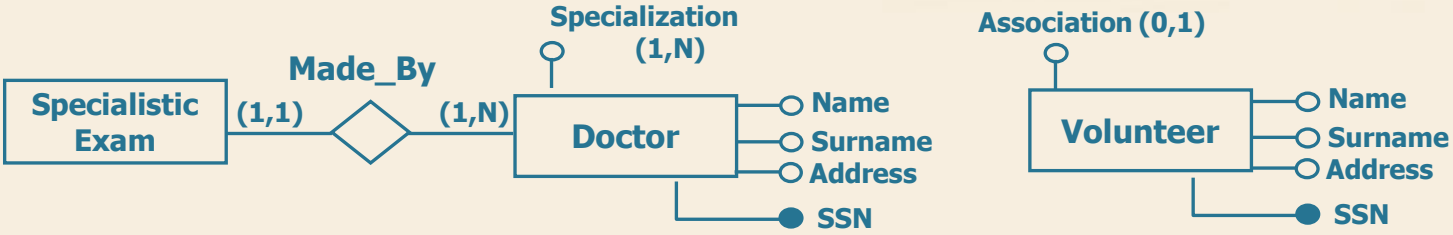
Example


Specialization
(1,N)
Association (0,1)
Made By
Specialistic Exam
(1,1)
(1,N)
Doctor
Volunteer
(t,e)
Dept
(1,N)
Works_in
(1,1)
Staff
Name
Surname
Address
SSN

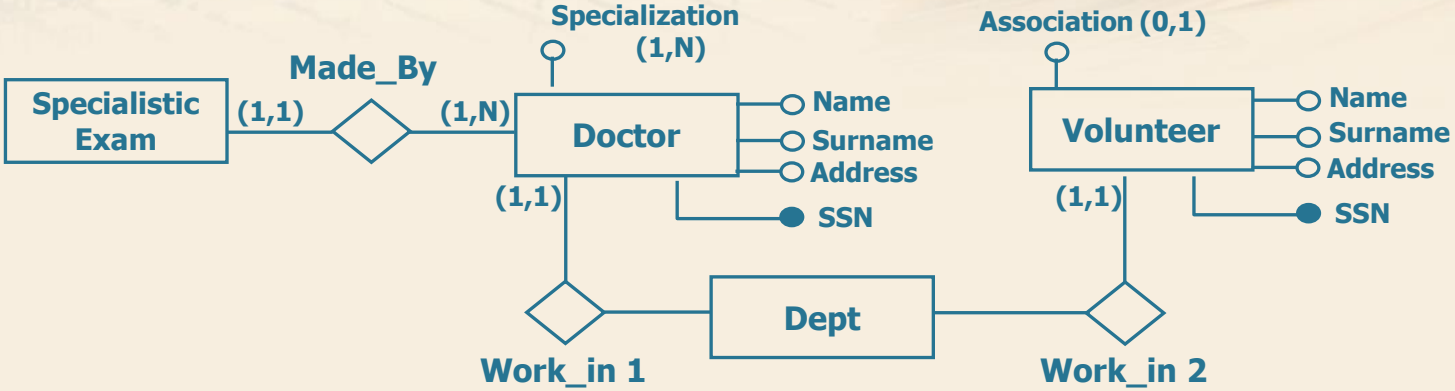
Parent->Child



Parent's attributes

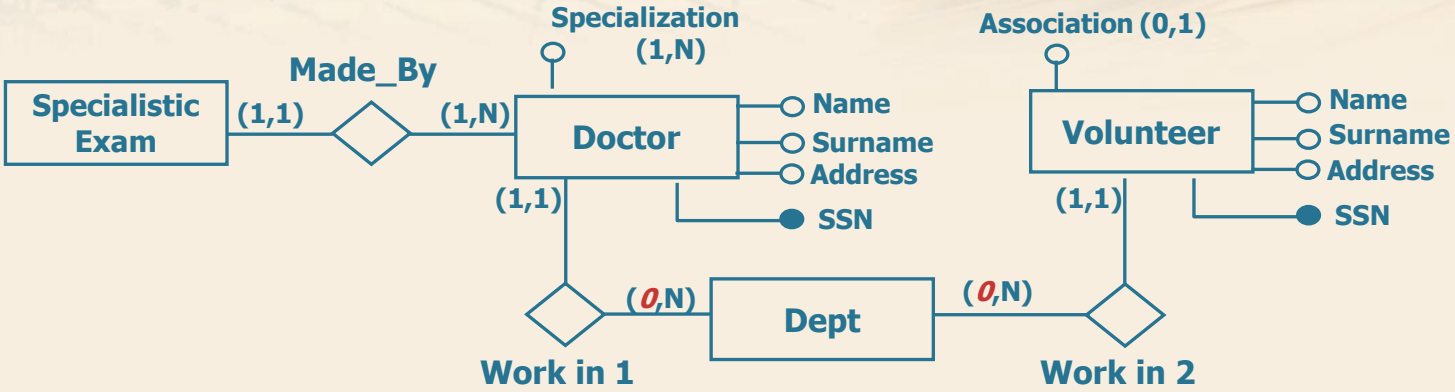


Relationships with parent



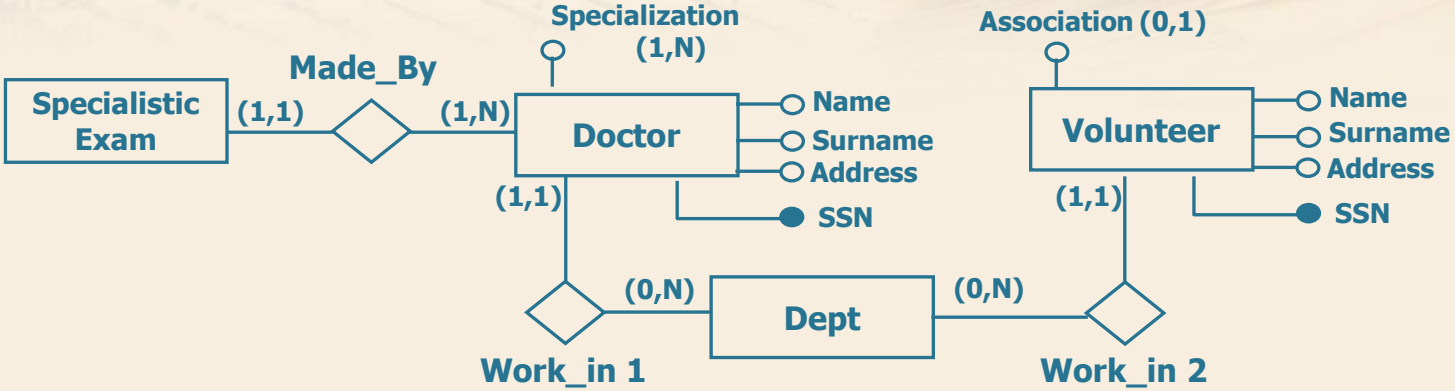
- Relations with the parent entity need to be split

Cardinality of «work in»



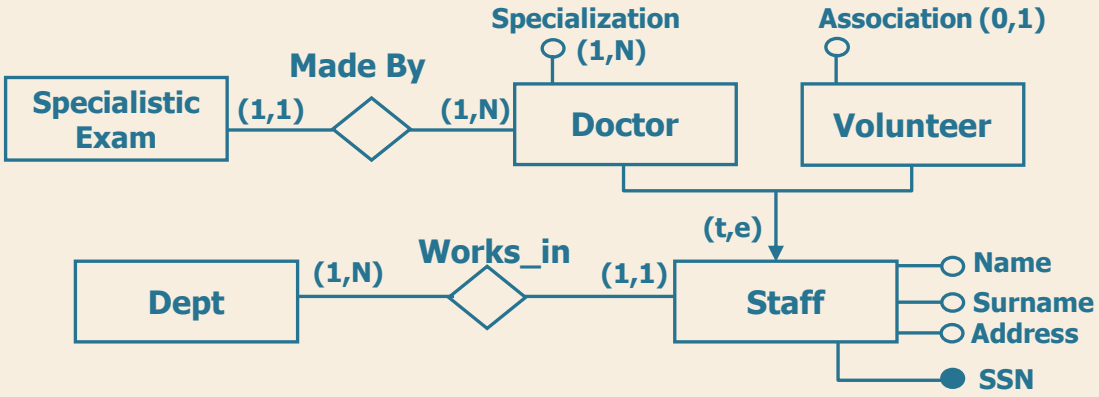
- Relations with the parent entity need to be split

Parent -> Child

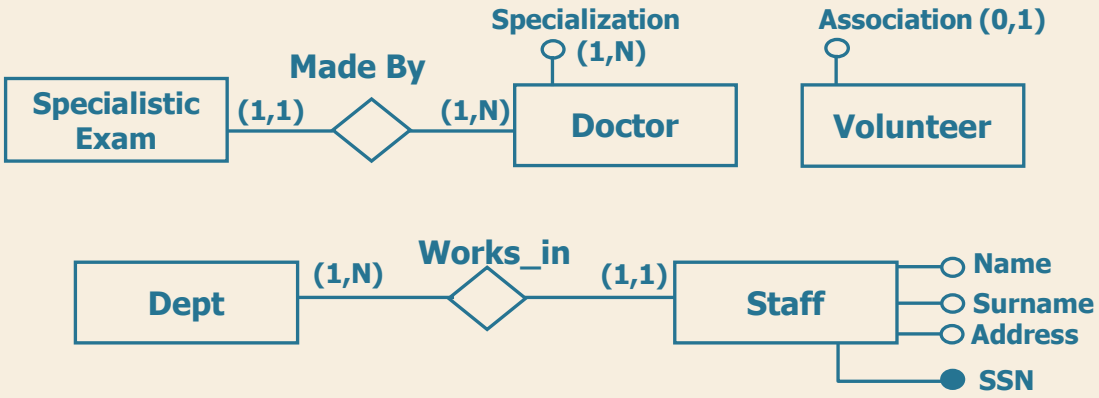


- Cannot be used for **partial** generalizzation
  - However, we can trasform partial generalizzation into total, adding a new entity called «Others»
- Cannot be used for **overlapping** generalizzation
  - Duplicate identifiers cause problems.

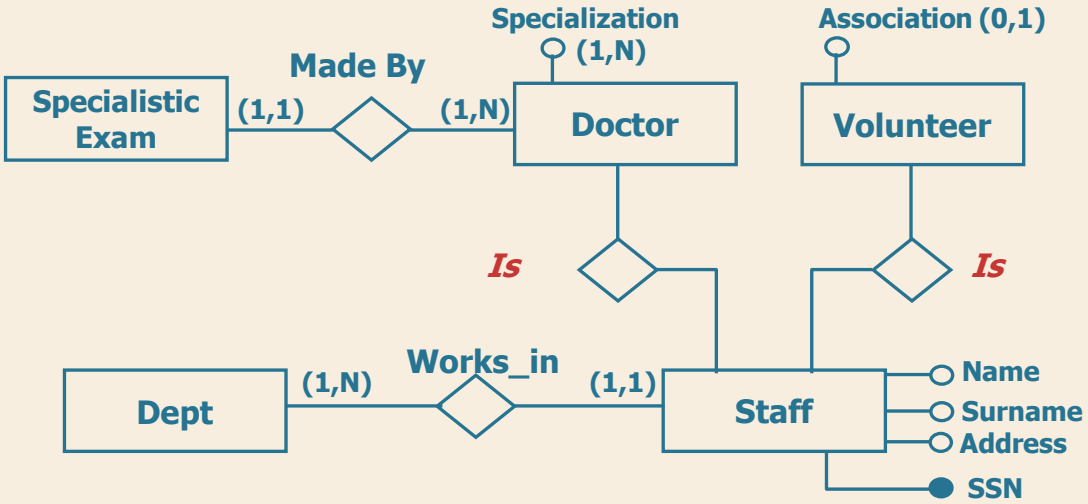
Example


Specialization
(1,N)
Association (0,1)
Made By
Specialistic Exam
(1,1)
(1,N)
Doctor
Volunteer
(t,e)
Works_in
Dept
(1,N)
(1,1)
Staff
Name
Surname
Address
SSN

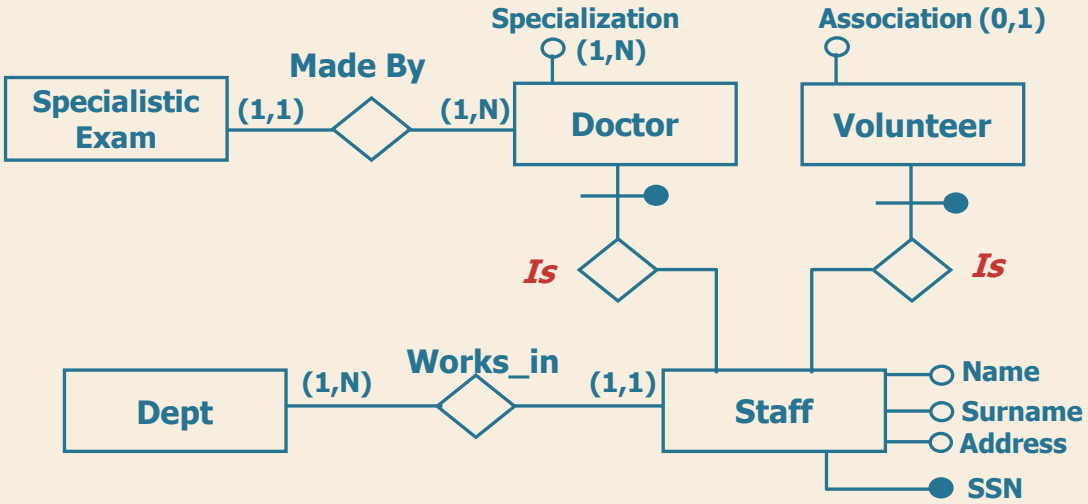
Example


Specialization
(1,N)
Association (0,1)
Made By
Specialistic Exam
(1,1)
(1,N)
Doctor
Volunteer
Works_in
Dept
(1,N)
(1,1)
Staff
Name
Surname
Address
SSN

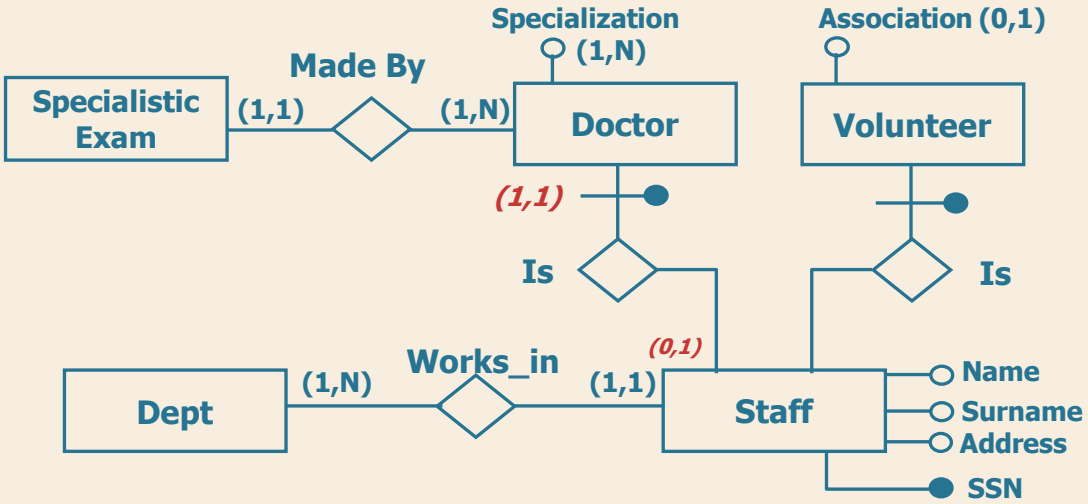
Relations between parent and child entities



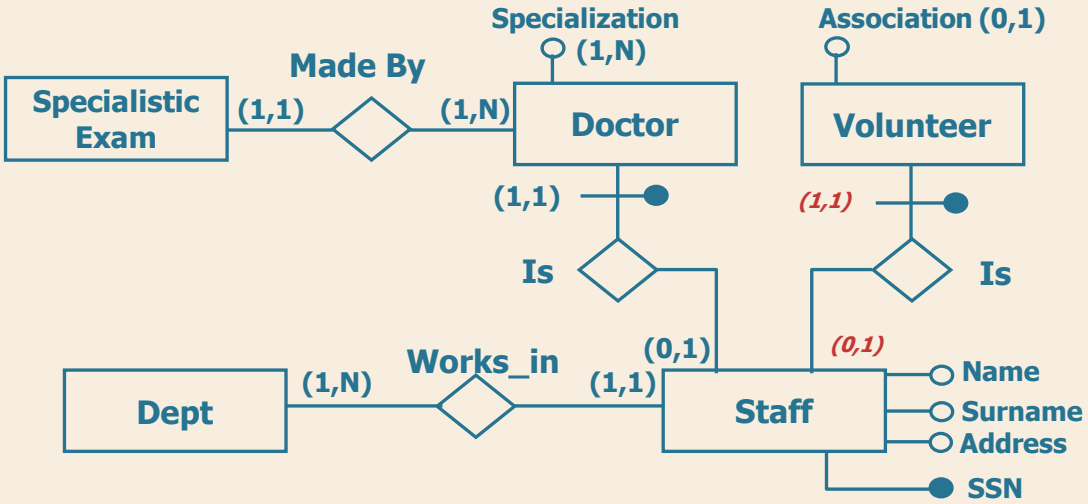
Child entities' identification



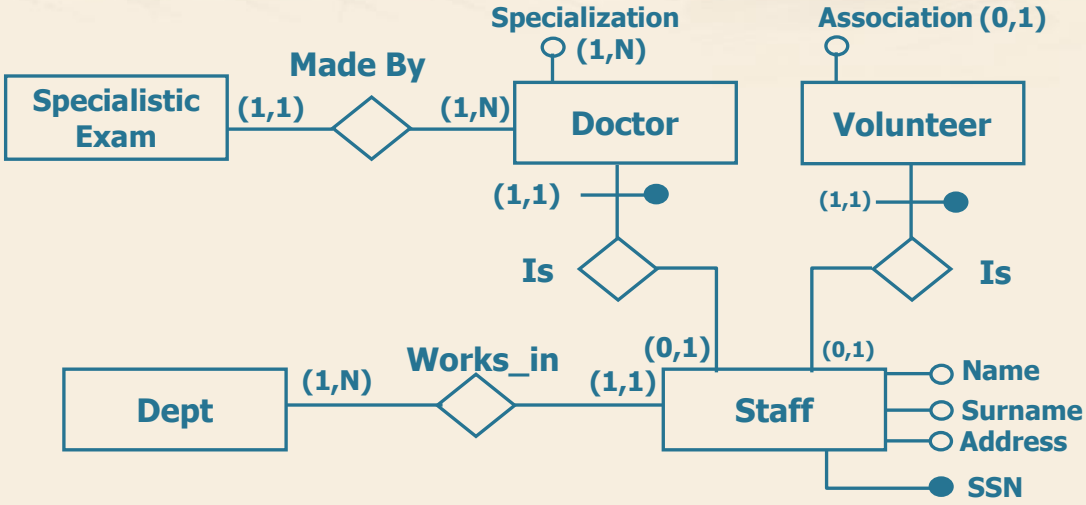
Cardinality of «is»



Cardinality of «is»



Generalization translated into relatioships



- This is the general solution.
- This kind of solution is always usable.
  - può essere dispendiosa per ricostruire l'informazione di partenza
  - But the restructuring process could be expensive...

Assessment of alternatives

- Merging child entities into parent entity is useful when:
  - The operations involve the occurrences and the attributes of child and parent entitites more or less in the same way (optimize data access).

Assessment of alternatives

- Merging parent entity into child entities is useful when:
  - The generalizzation is «total»
  - there are operations that refer only to occurrences of child entities and so they make distinctions between these entities (optimize data access).

Assessment of alternatives

- The various options can be combined
  - there are operations that refer only to occurrences of some child entities (optimize data access).

Assessment of alternatives

- In presence of hieracy:
  - Procede in the same way
  - Start from the lower levels.

Logical Design

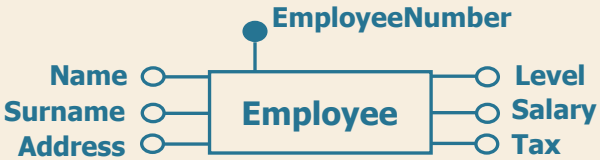
Partitioning of concepts

DBMG

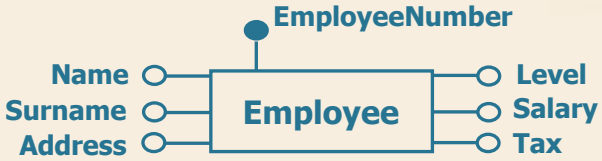
Partitioning of concepts

- Partitioning of entities and relationships
  - Best representation of different concepts
  - Separating attributes of the same concept that are accessed by different operation
  - Improve the efficency of the operations.

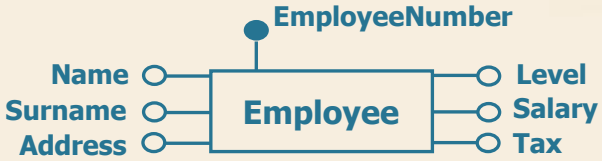
Partizionamento di entità



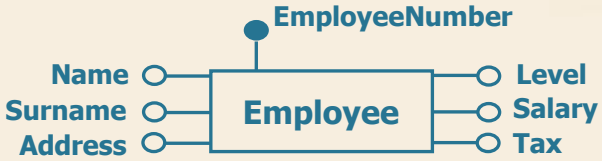
Partizionamento di entità

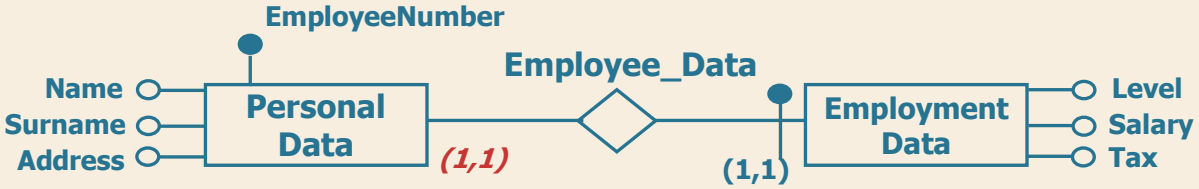


Cardinality of "Employment Data"

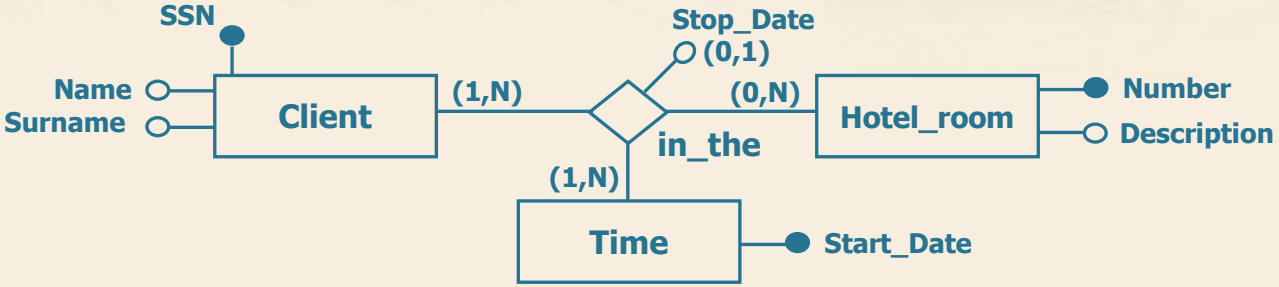


Cardinality of «Employement Data»

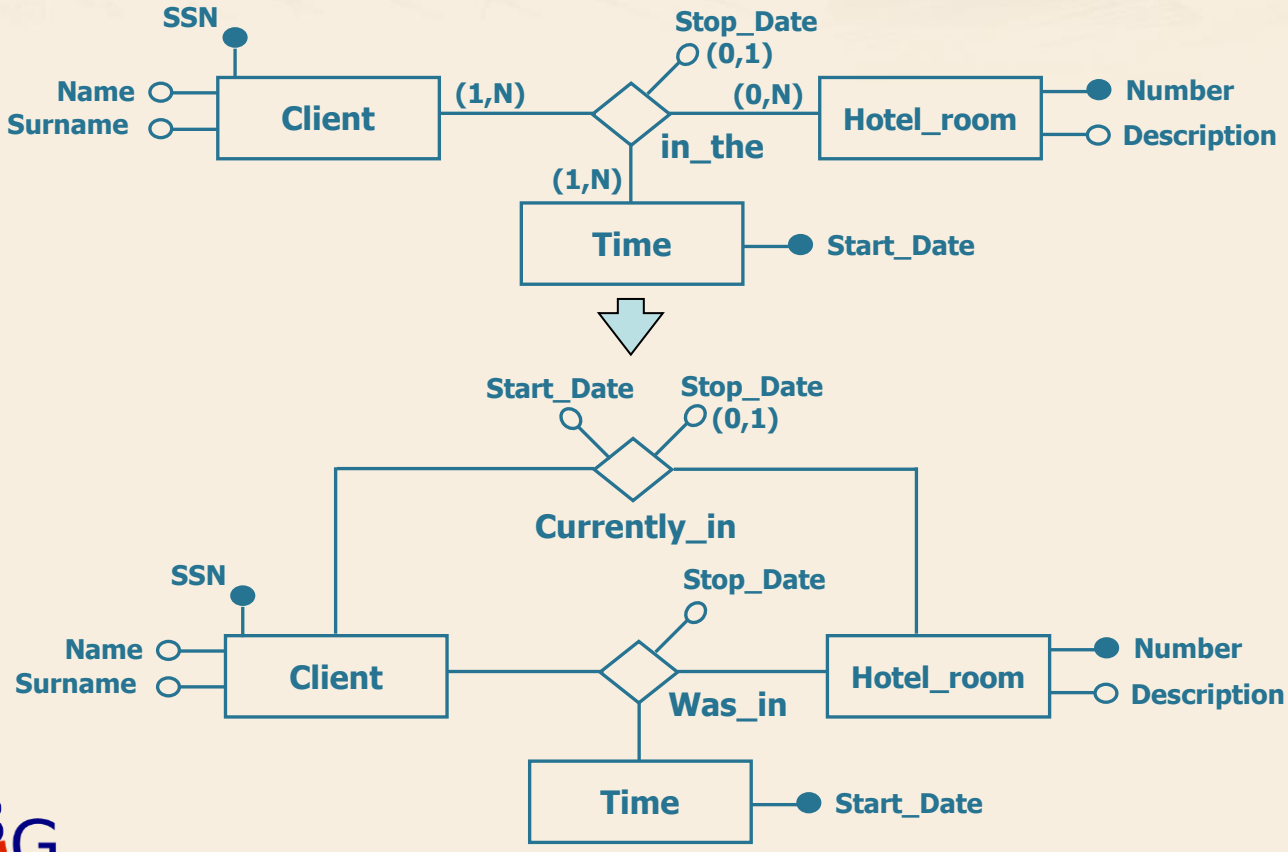




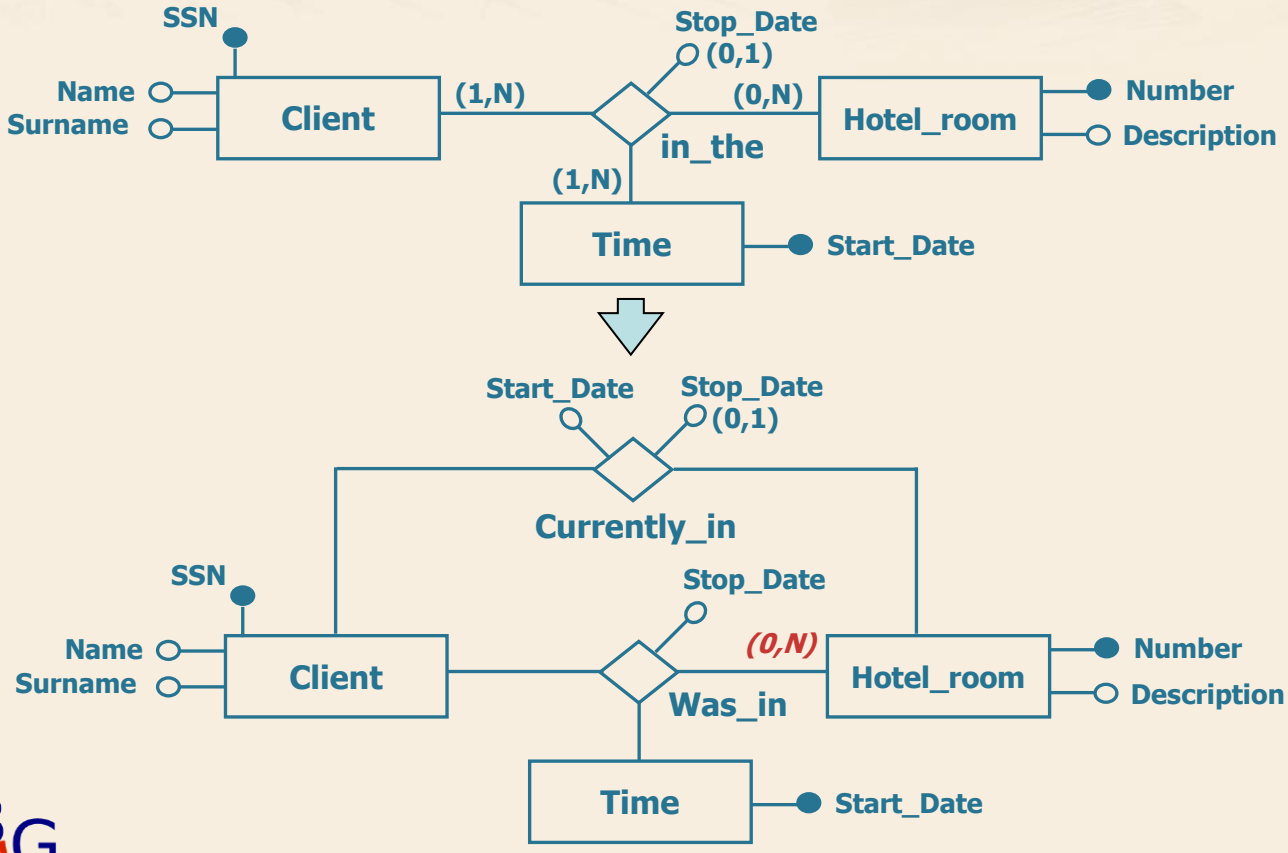
Relationships' partitioning



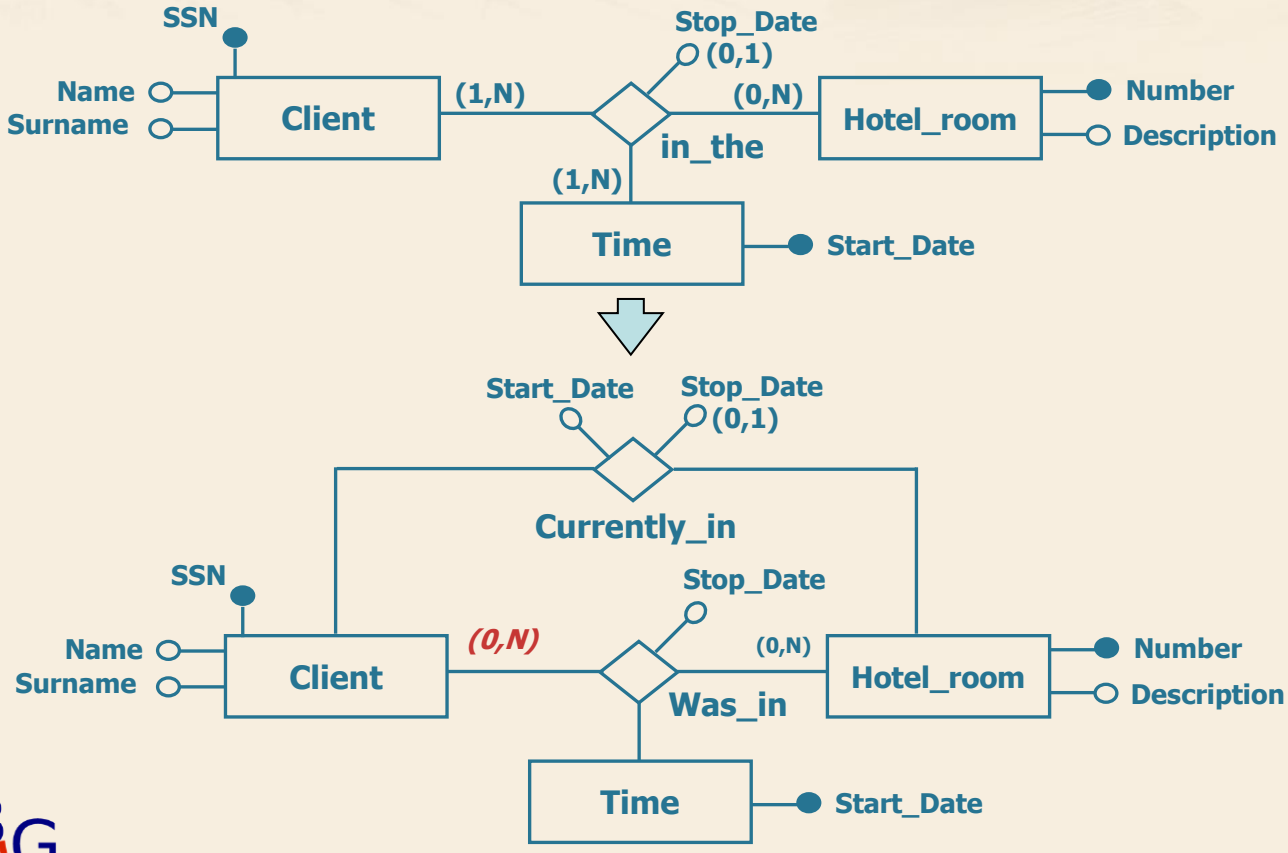
Relationships' partitioning



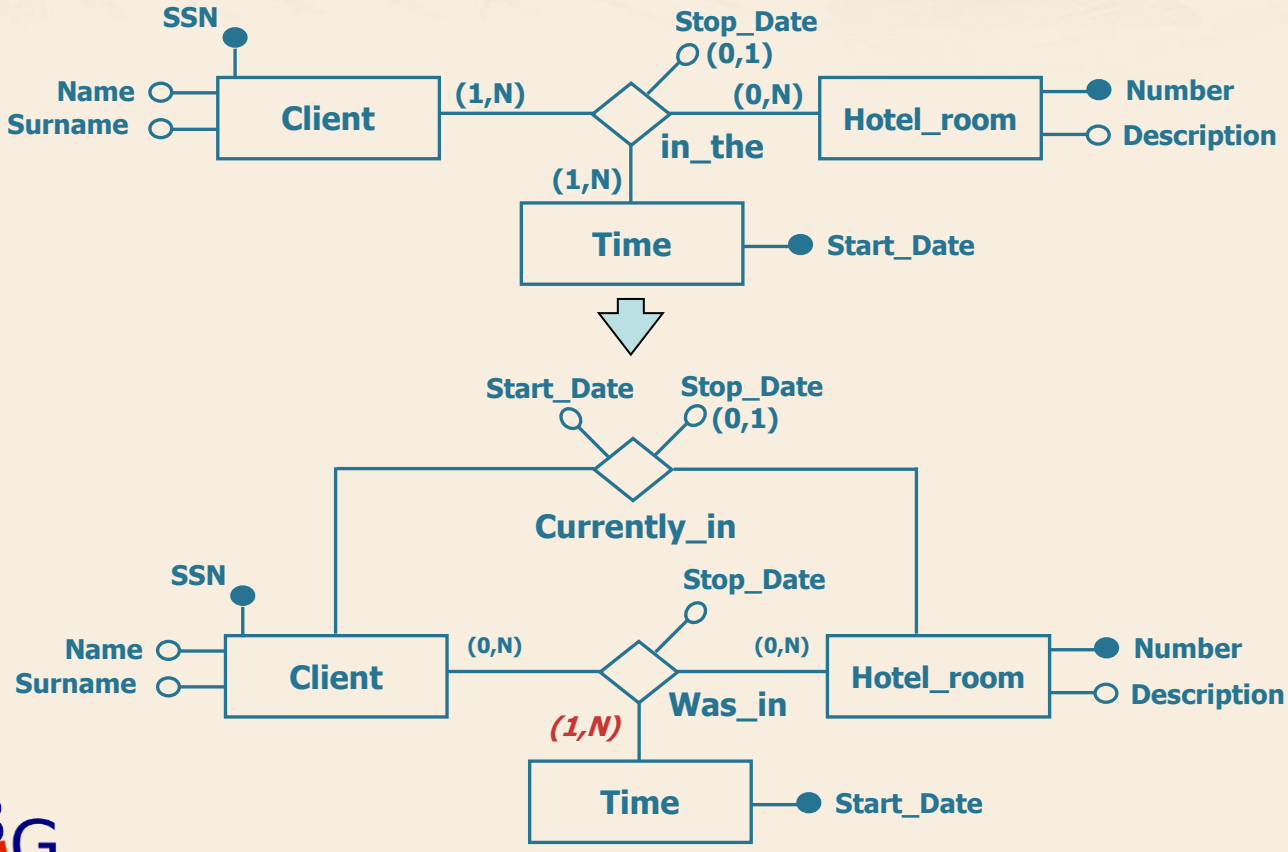
Cardinality of «Was in»



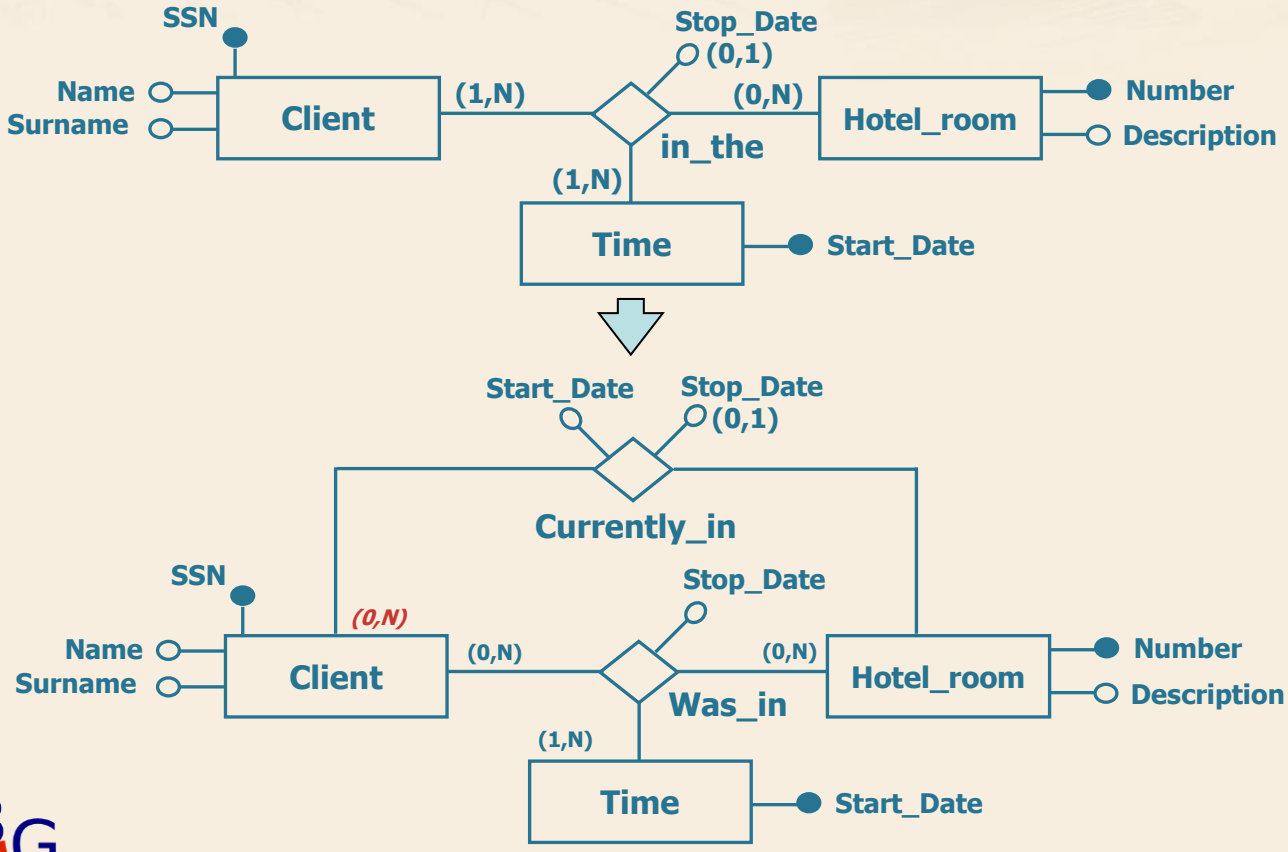
Cardinality of «Was in»



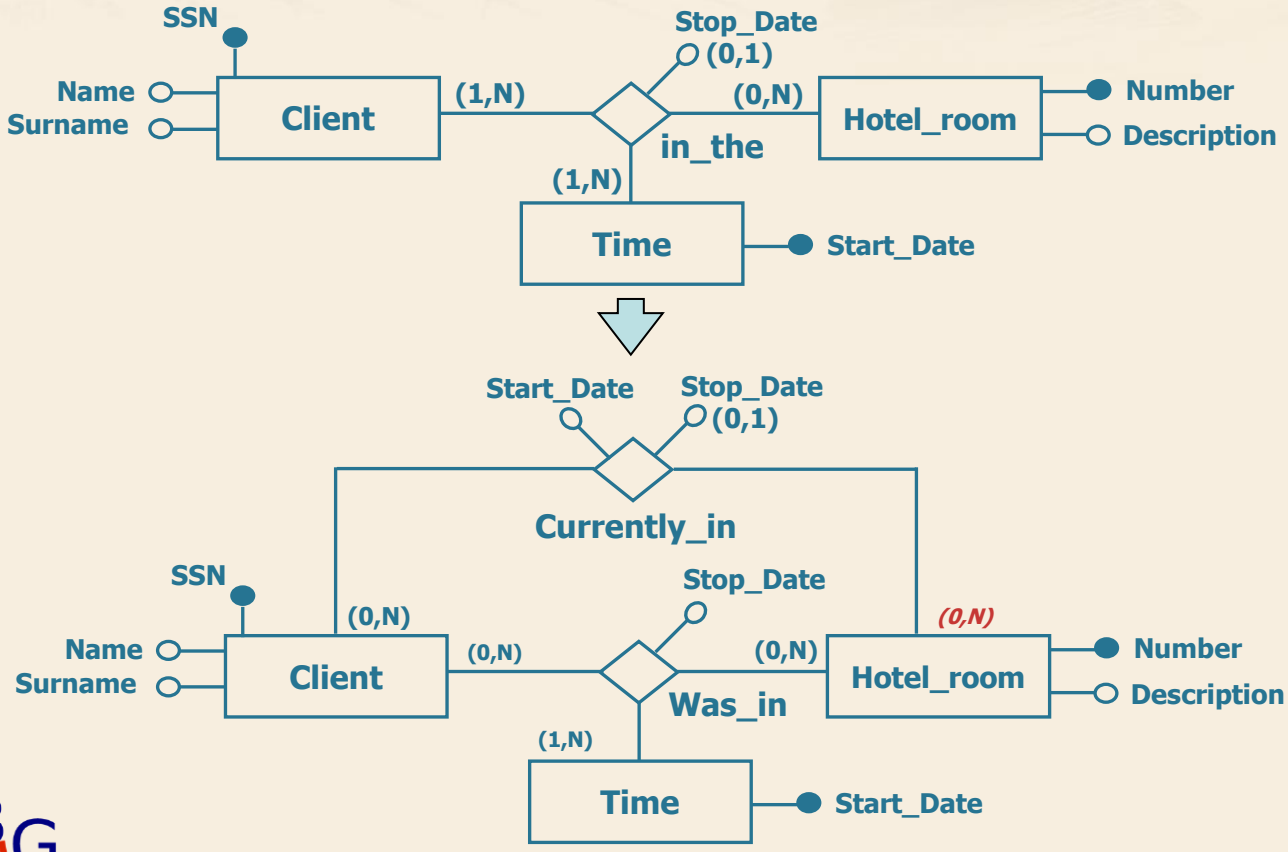
Cardinality of «Was in»



Cardinality of «Currently in»



Cardinality of «Currently in»



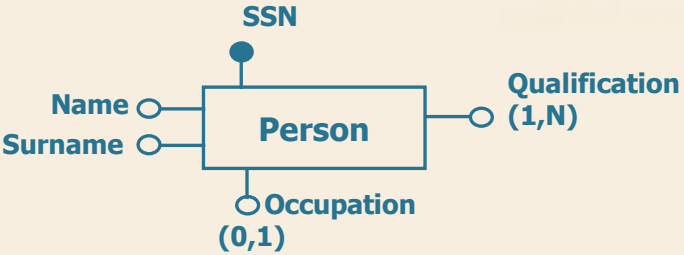
Logical Design

Removing multivalued attributes

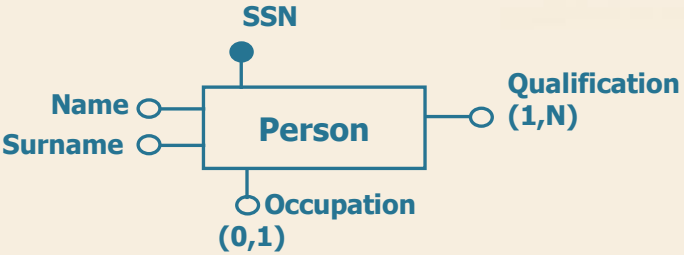
Removing multivalued attributes

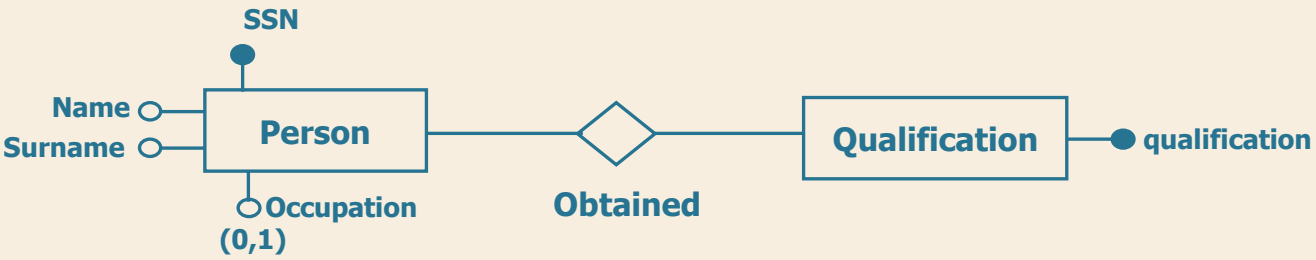
- Multi-valued attributes are not representable in the relational model
- A multi-valued attribute is represented by a relationship between
  - the original entity
  - a new entity
- Pay attention to the cardinality of the new relationship

Removing multivalued attributes

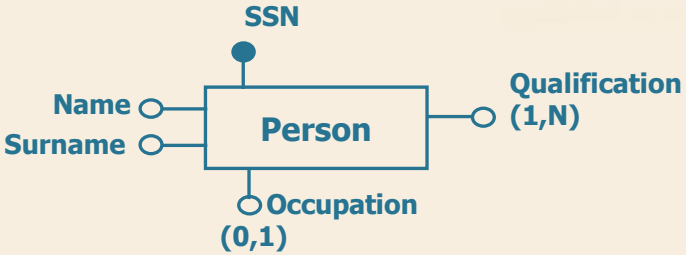


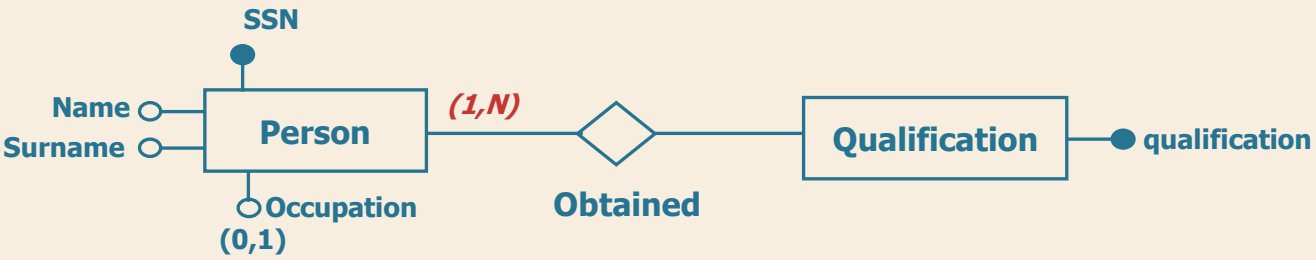
Removing multivalued attributes



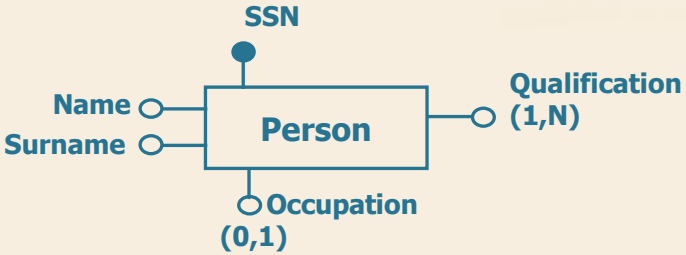


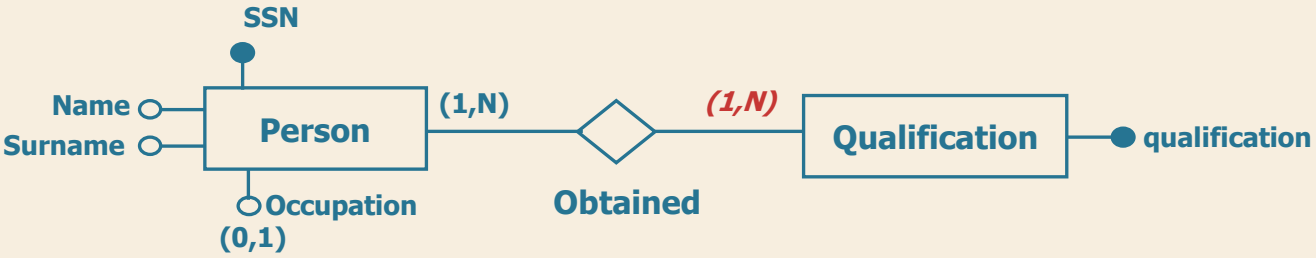
Cardinality of «Obtained»



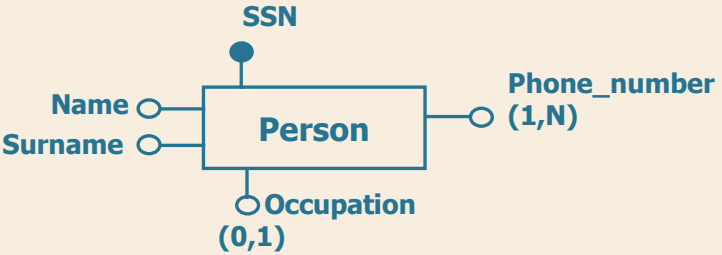


Cardinality of «Obtained»

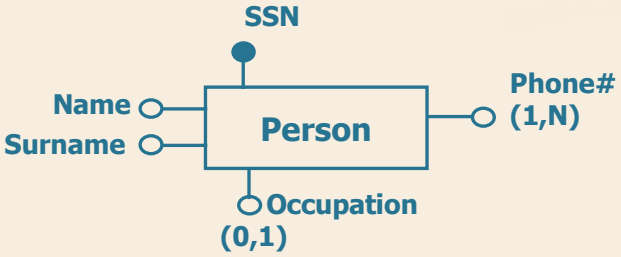


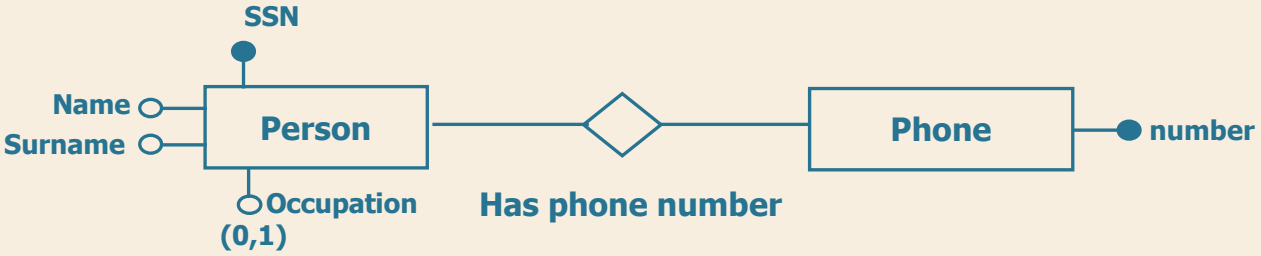


Removing multivalued attributes

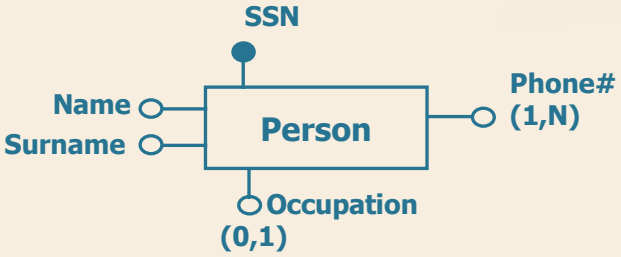


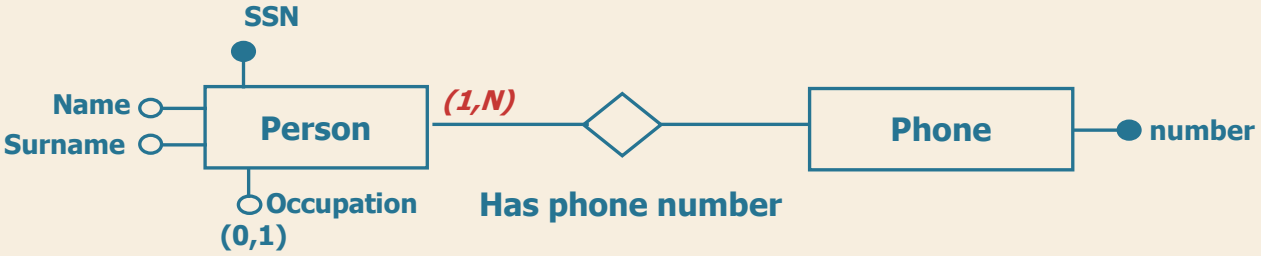
Removing multivalued attributes



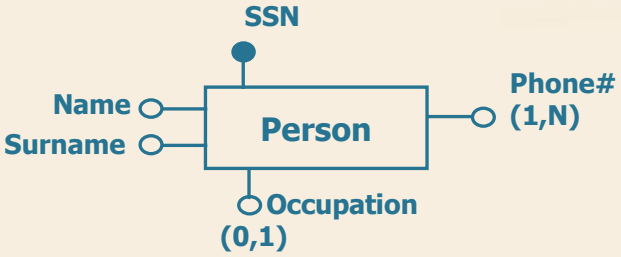


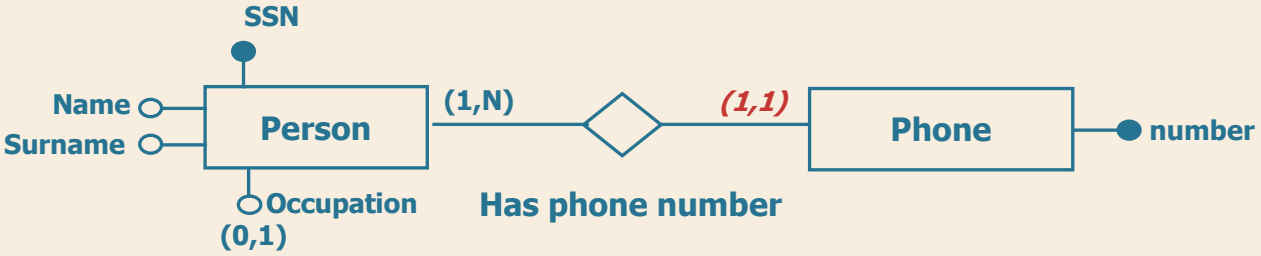
Cardinality of «Has phone number»





Cardinality of «Has phone number»





Logical Design

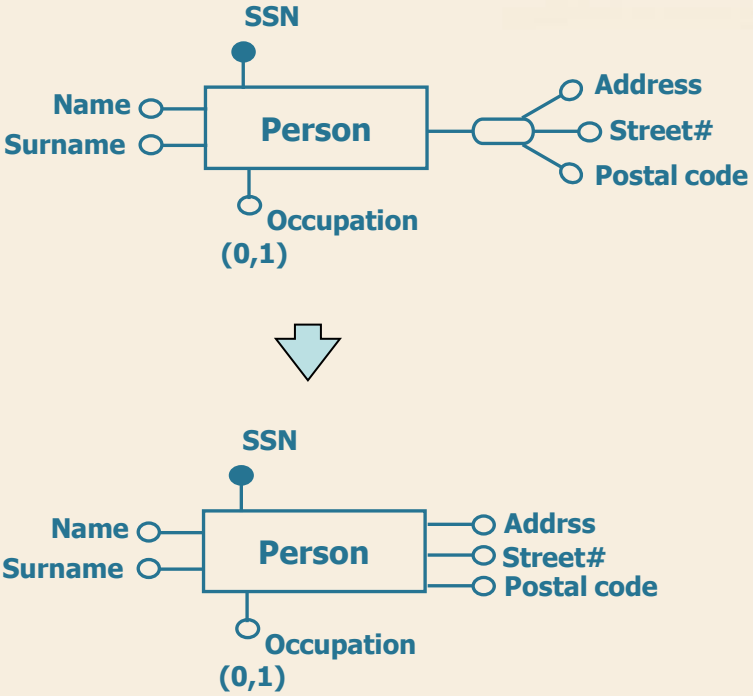
Removing composed attributes
Selection of primary identifiers

DBMG

Removing composed attributes

- Composed attributes are not representable in the relational model
- Two options
  - Split them in «individual» attributes
    - useful if you need to access each attribute separately

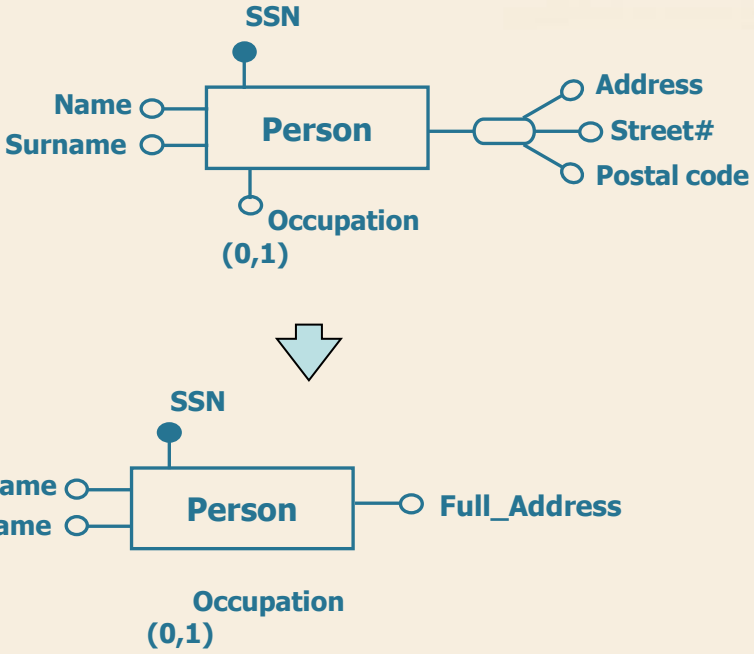
Split composed attributes



Removing composed attributes

- Composed attributes are not representable in the relational model.
- Two ways:
  - Split them in «individual» attributes.
    - useful if you need to access each attribute separately.
  - Use one attribute as a «link»
    - useful if access to comprehensive information is enough

Example


SSN
Name
Surname
Person
Address
Street#
Postal code
Occupation
(0,1)
SSN
Name
Surname
Person
Full_Address
Occupation
(0,1)

Selection of primary identifiers

- It is necessary to define the *primary key*
- The criteria for this decision are as follows
  - Attributes with **null** values **cannot** form primary identifiers.
  - Just **one** or **few** attributes
  - An **internal** identifier with few attributes is preferable to an external one
  - It is used by many operations to access the occurrences
- Possibly introduce a further (new) attribute to the entity, often called codes, e.g. «ProductCode»