Enhanced Entity-Relationship Modelling

Watch video: https://youtu.be/w-NHKN94g9A?t=39m28s

Enhanced Entity-Relationship Modelling

- The basic concepts of ER modelling are often insufficient to represent the requirements of more complex applications.

- Many different semantic data models have been proposed and some of the most important semantic concepts have been successfully incorporated into the original ER model.

- The ER model supported with additional semantic concepts is called the Enhanced Entity-Relationship (EER) model.

Topics List

- Specialization/Generalization
- Constraints on Specialization / Generalization

Specialization/Generalization

- The concept of Specialization/Generalization is associated with special types of entity types known as *superclasses* and *subclasses*, and the process of *attribute inheritance*.

Specialization/Generalization

- Superclass
  - An entity type that includes one or more distinct subgroupings of its occurrences, which must be represented in a data model.
- Subclass
  - A distinct subgrouping of occurrences of an entity type, which must be represented in a data model.

Specialization/Generalization

- Entity types that have distinct subclasses are called superclasses.

- For example, the entities that are members of the *Staff* entity type may be classified as *Manager, SalesPersonnel,* and *Secretary*.

- The *Staff* entity type is referred to as the **superclass** of the *Manager, SalesPersonnel,* and *Secretary* ***subclasses***.

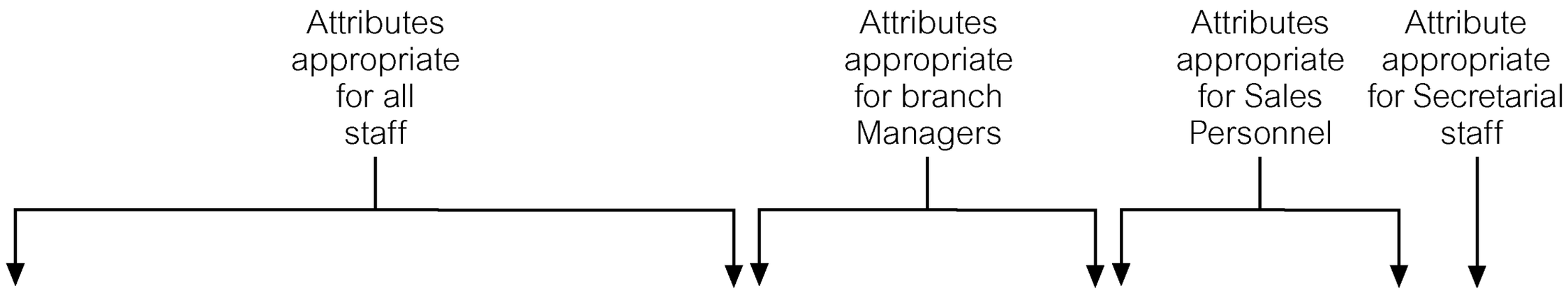
Specialization/Generalization

- Each member of a subclass is also a member of the superclass. The entity in the subclass is the same entity in the superclass, but has a distinct role. So the Superclass/subclass relationship is one-to-one (1:1).

- Superclass may contain overlapping or distinct subclasses.

- Not all members of a superclass need be a member of a subclass.

All Staff relation holding details of all staff



staffNo	name	position	salary	mgrStartDate	bonus	sales Area	car Allowance	typing Speed
SL21	John White	Manager	30000	01/02/95	2000			
SG37	Ann Beech	Assistant	12000					
SG66	Mary Martinez	Sales Manager	27000			SA1A	5000	
SA9	Mary Howe	Assistant	9000					
SL89	Stuart Stern	Secretary	8500					100
SL31	Robert Chin	Snr Sales Asst	17000			SA2B	3700	
SG5	Susan Brand	Manager	24000	01/06/91	2350			

Specialization/Generalization

- Attribute Inheritance
  - An entity in a subclass represents the same ‘real world’ object as in the superclass, and may possess subclass-specific attributes, as well as those associated with the superclass.
  - For example, a member of the *SalesPersonnel* subclass inherits all the attributes of the *Staff* superclass, such as *staffNo, name, position* and *salary* together with those specifically associated with the *SalesPersonnel* subclass, such as *salesArea* and *carAllowance.*

Specialization/Generalization

- Specialization
  - Process of maximizing differences between members of an entity type by identifying their distinguishing characteristics.
  - Specialization is a top-down approach to defining a set of superclasses and their related subclasses.
  - The set of subclasses is defined on the basis of some distinguishing characteristics of the entity types in the superclass.

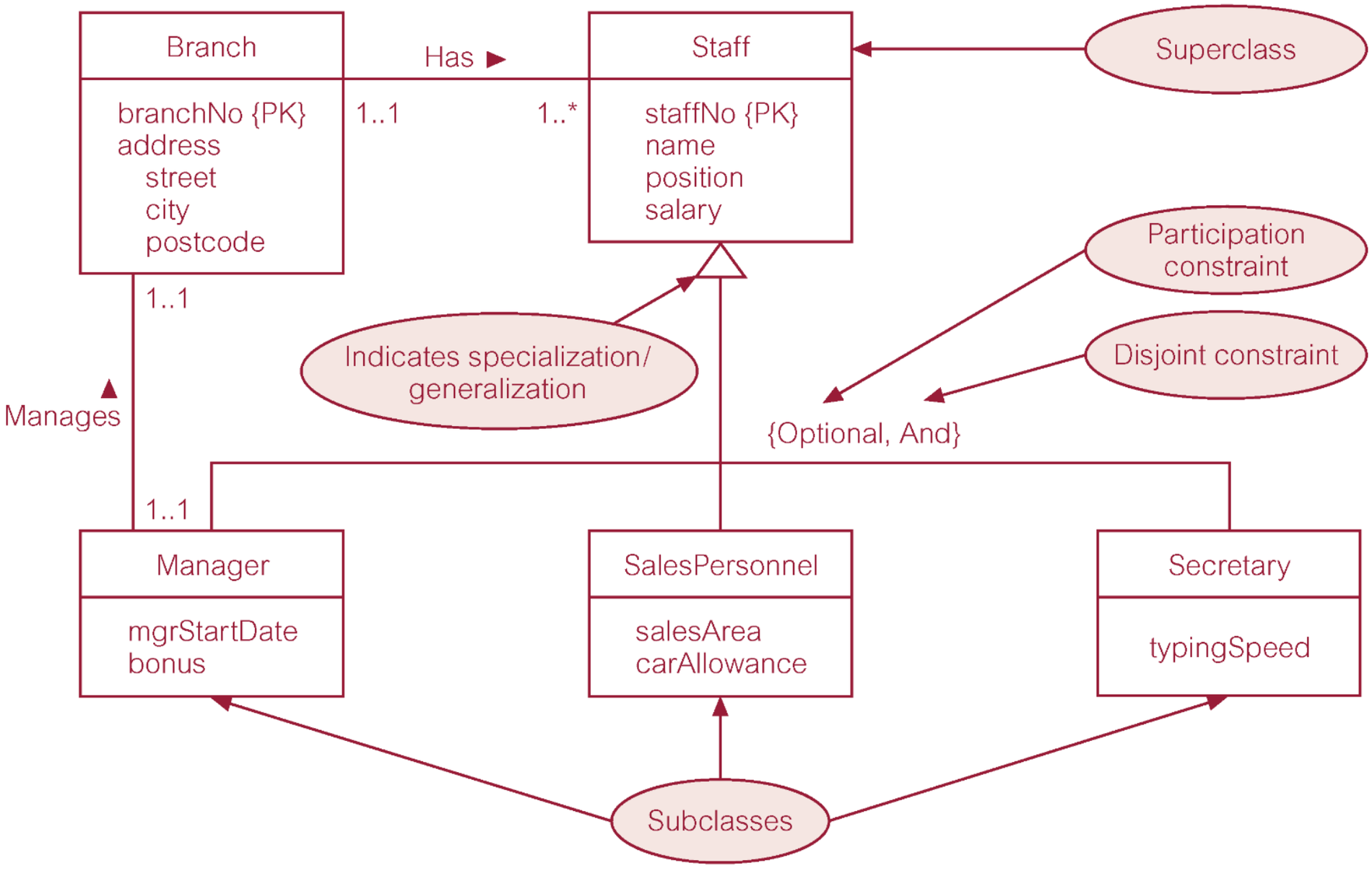
Specialization/Generalization

- Specialization
  - When we identify a set of subclasses of an entity type, we then associate attributes specific to each subclass, and also identify any relationships between each subclass and other entity types or subclasses.
  - For example, consider a model where all members of staff are represented as an entity called *Staff*. If we apply the process of specialization on the *Staff* entity type, we attempt to identify differences between members of this entity type, such as members with distinctive attributes and/or relationships.

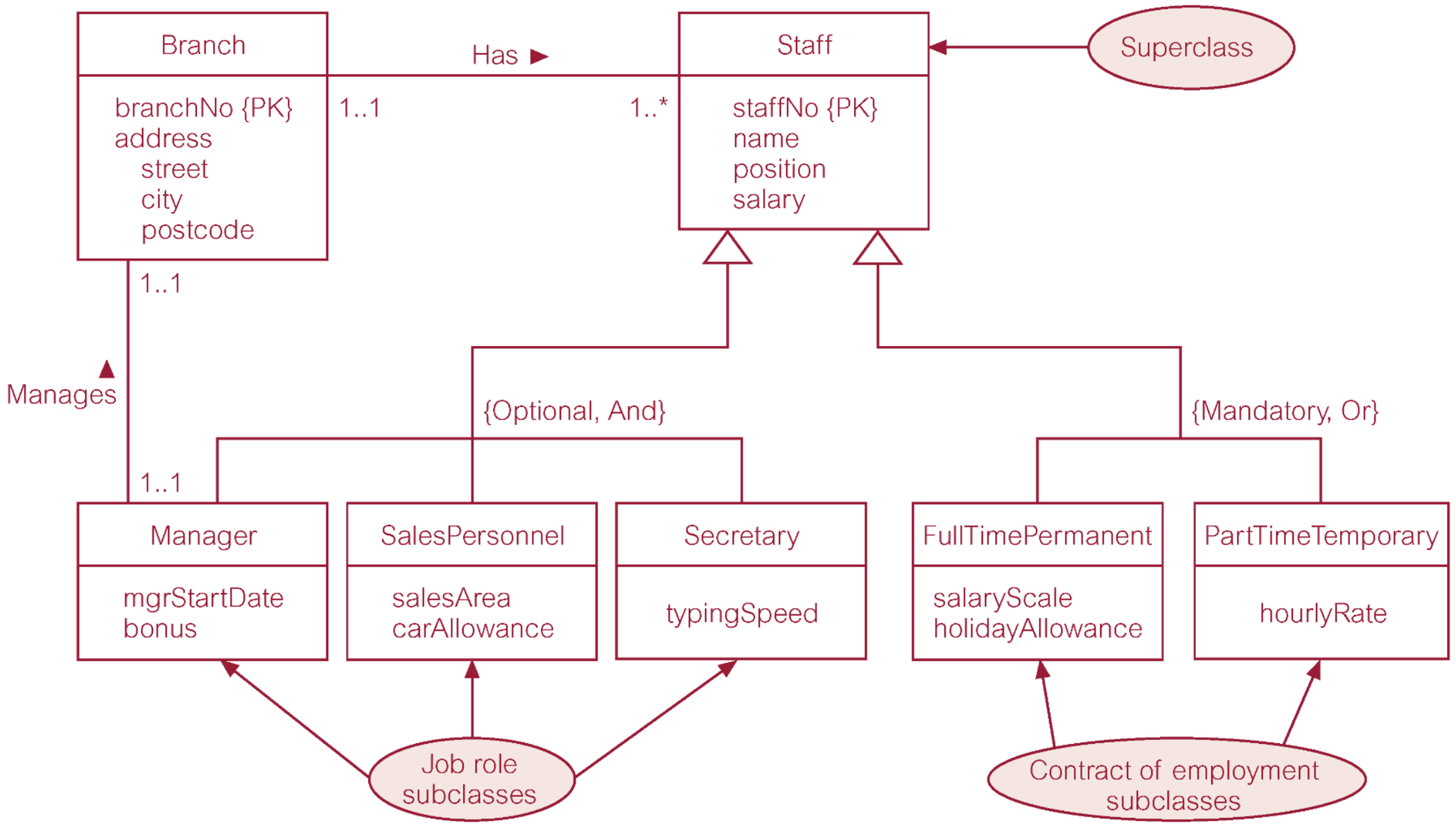
Specialization/Generalization

- Generalization
  - Process of minimizing differences between entities by identifying their common characteristics.
  - The process of generalization is a bottom-up approach, that results in the identification of a generalized superclass from the original entity types.
  - For example, consider a model where *Manager, SalesPersonnel,* and *Secretary* are represented as distinct entity types. If we apply the process of generalization on these entity types, we identify similarities between them, such as common attributes and relationships.

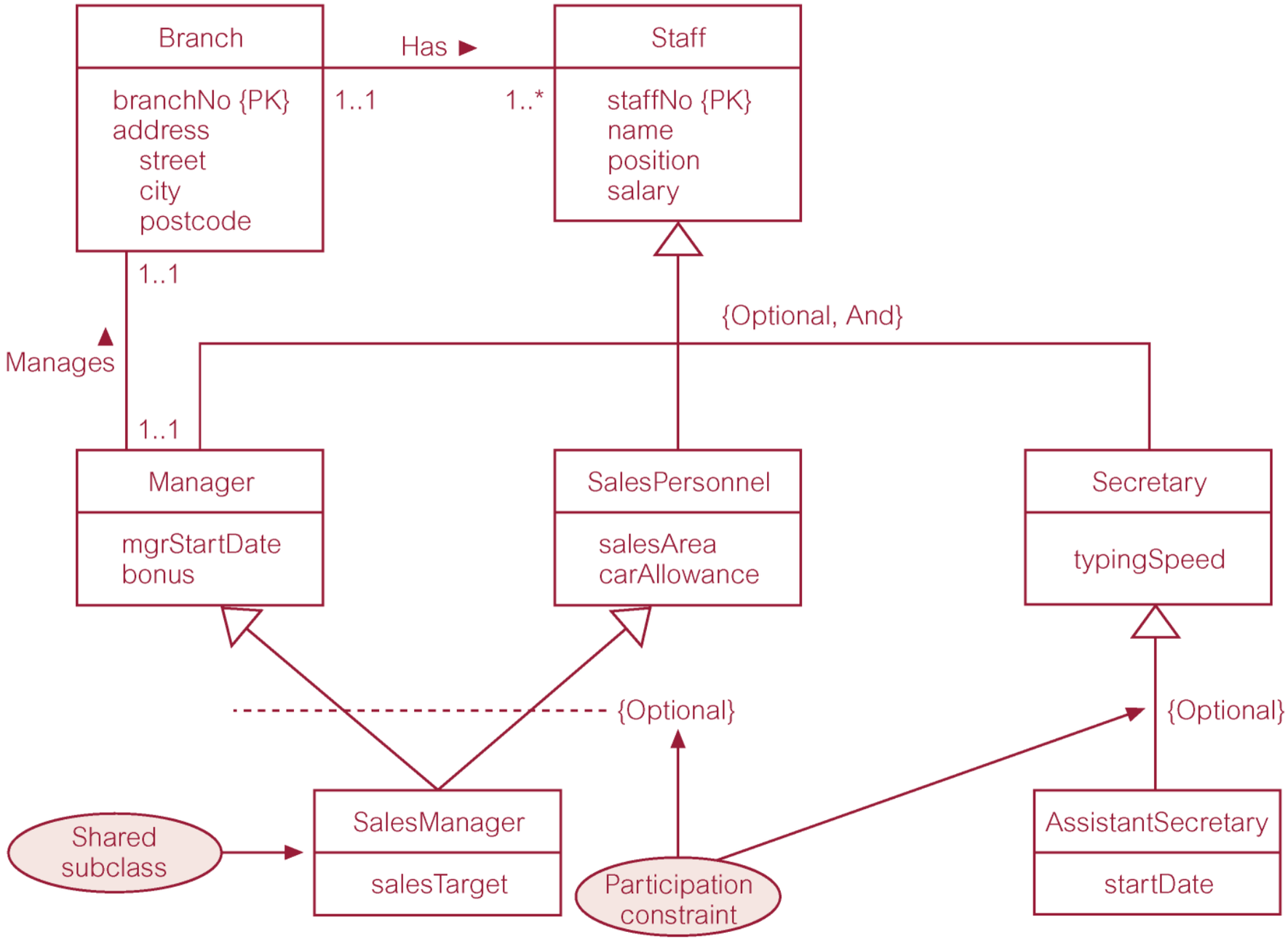
Specialization/generalization of Staff entity type into subclasses representing job roles



Specialization/generalization of Staff entity type into job roles and contracts of employment



EER diagram with shared subclass and subclass with its own subclass



Topics List

- Specialization/Generalization
- Constraints on Specialization/Generalization

Constraints on Specialization/Generalization

- Two constraints that may apply to a specialization/generalization:
  - Participation constraints
  - Disjoint constraints.

Constraints on Specialization/Generalization

- Participation constraint
  - Determines whether every member in superclass must participate as a member of a subclass.
  - May be *mandatory* or *optional*.
  - A superclass/subclass relationship with *mandatory* participation specifies that each member in the superclass must also be a member of a subclass.
  - A superclass/subclass relationship with *optional* participation specifies that a member of a superclass need not belong to any of its subclasses.
  - To represent the participation constraint, either *mandatory* or *optional* is placed in curly brackets below the triangle that points towards the superclass.

Constraints on Specialization/Generalization

- Disjoint constraint
  - Describes relationship between members of the subclasses and indicates whether member of a superclass can be a member of one, or more than one, subclass.
  - May be *disjoint* or *nondisjoint*.
  - The disjoint constraint only applies when a superclass has more than one subclass.

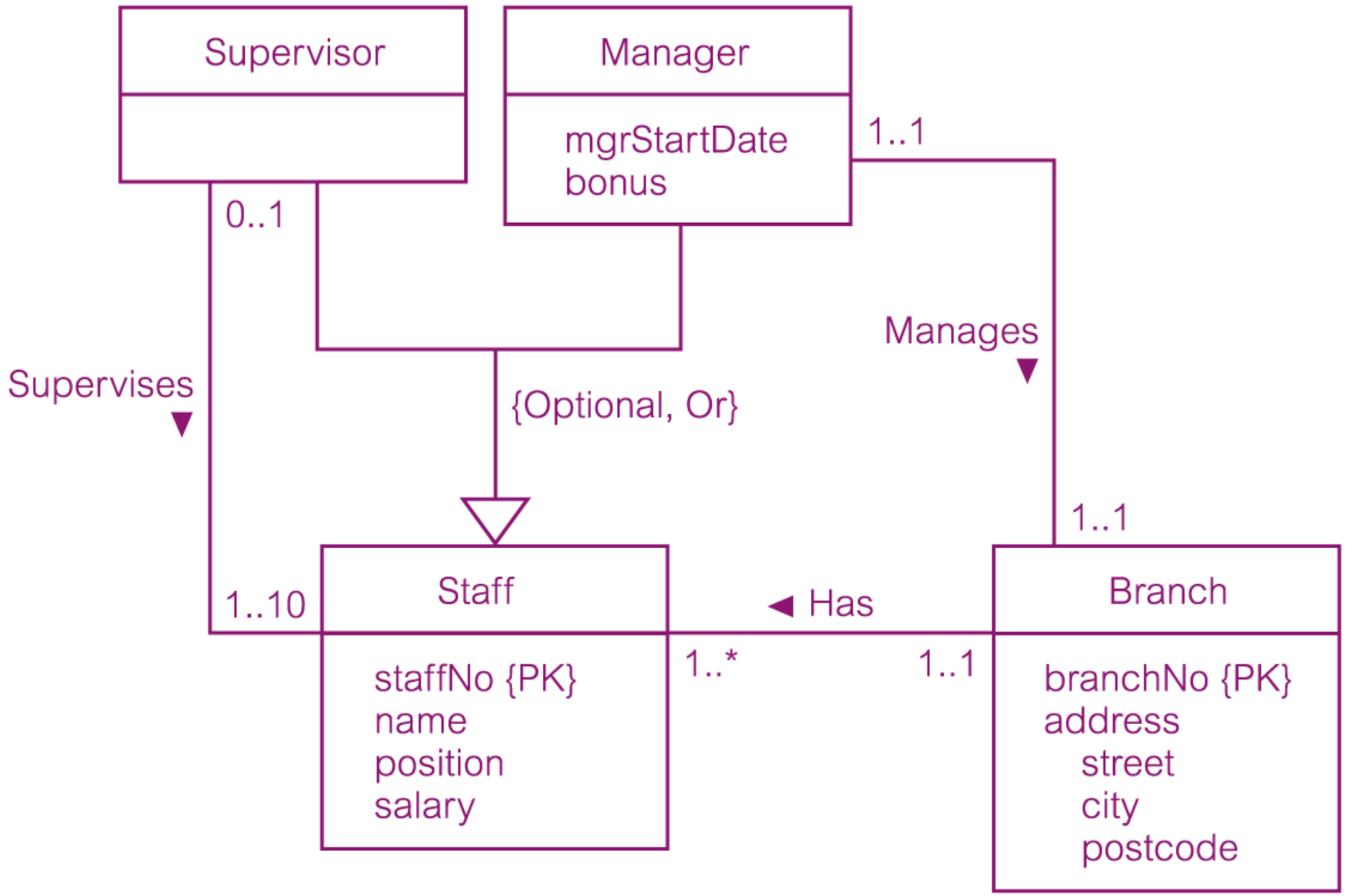
Constraints on Specialization/Generalization

- Disjoint constraint
  - If the subclasses are *disjoint*, then an entity occurrence can be a member of only one of the subclasses.
  - If the subclasses are *nondisjoint*, then an entity occurrence may be a member of more than one subclass.
  - To represent the disjoint constraint, either *Or (disjoint)* or *And (nondisjoint)* is placed next to the participation constraint within curly brackets below the triangle that points towards the superclass.

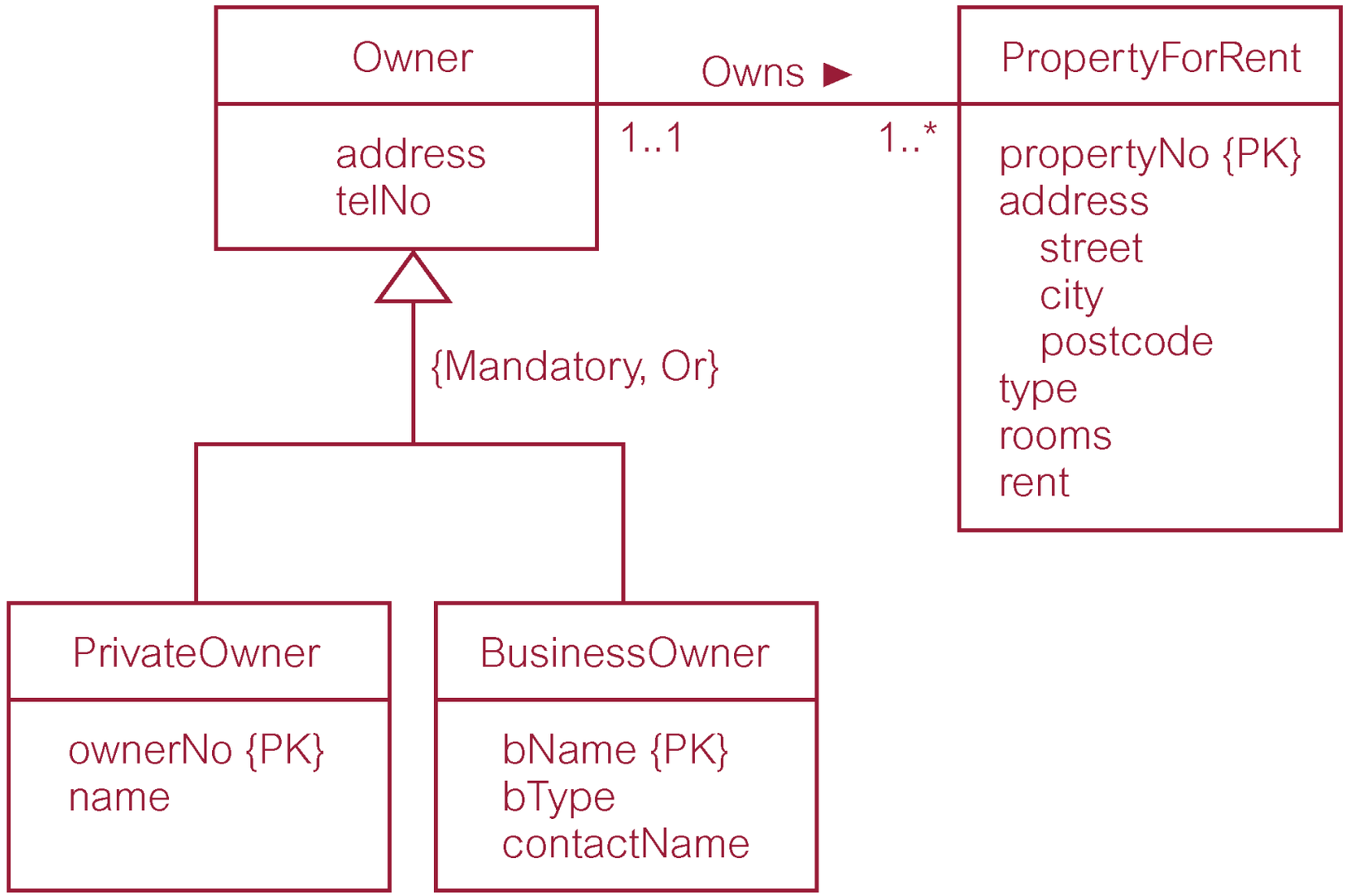
Constraints on Specialization/Generalization

- There are four categories of constraints of specialization and generalization:
  - mandatory and disjoint
  - optional and disjoint
  - mandatory and nondisjoint
  - optional and nondisjoint.

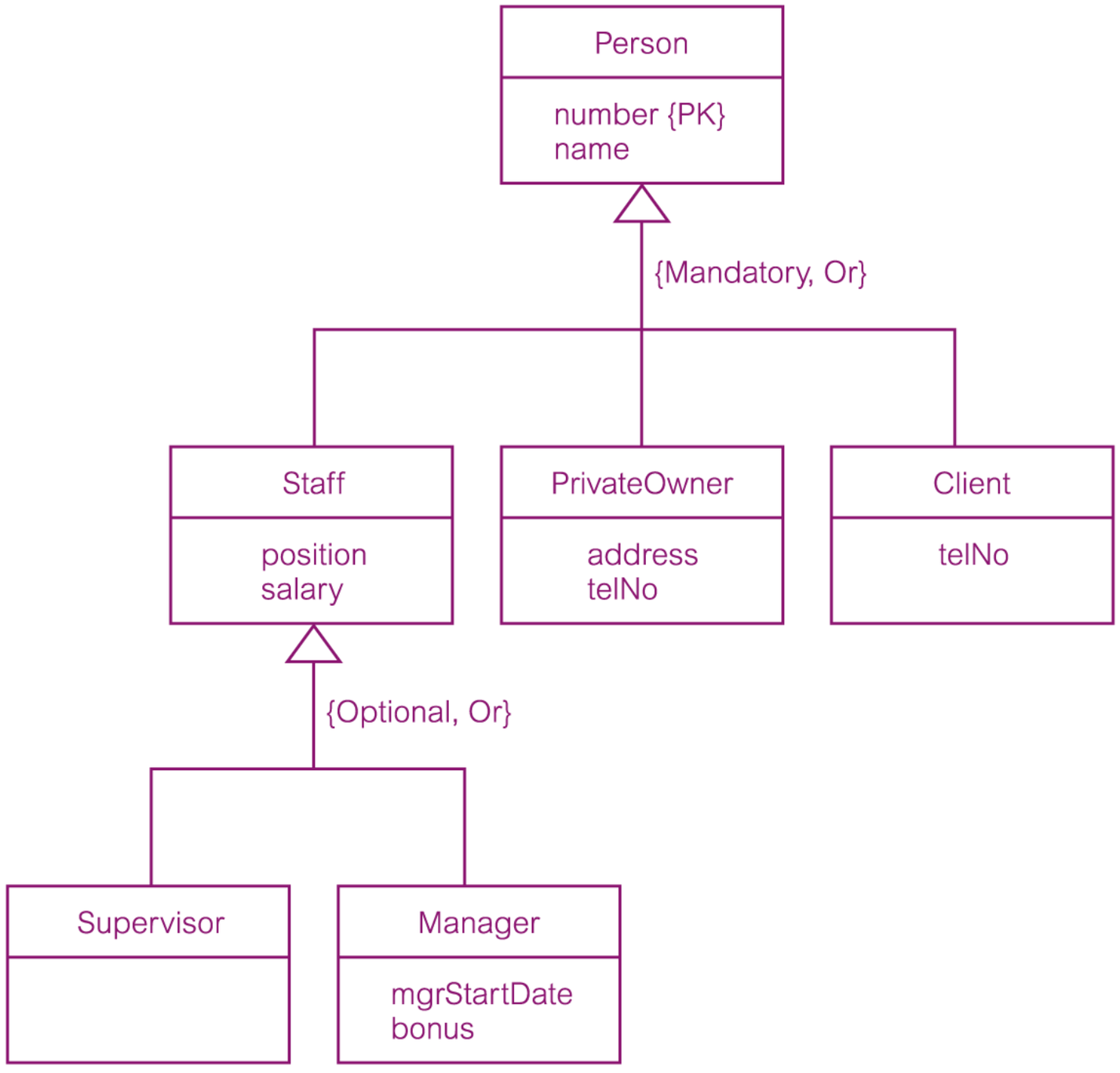
DreamHome worked example - Staff Superclass with Supervisor and Manager subclasses



DreamHome worked example - Owner Superclass with PrivateOwner and BusinessOwner subclasses



DreamHome worked example - Person superclass with Staff, PrivateOwner, and Client subclasses



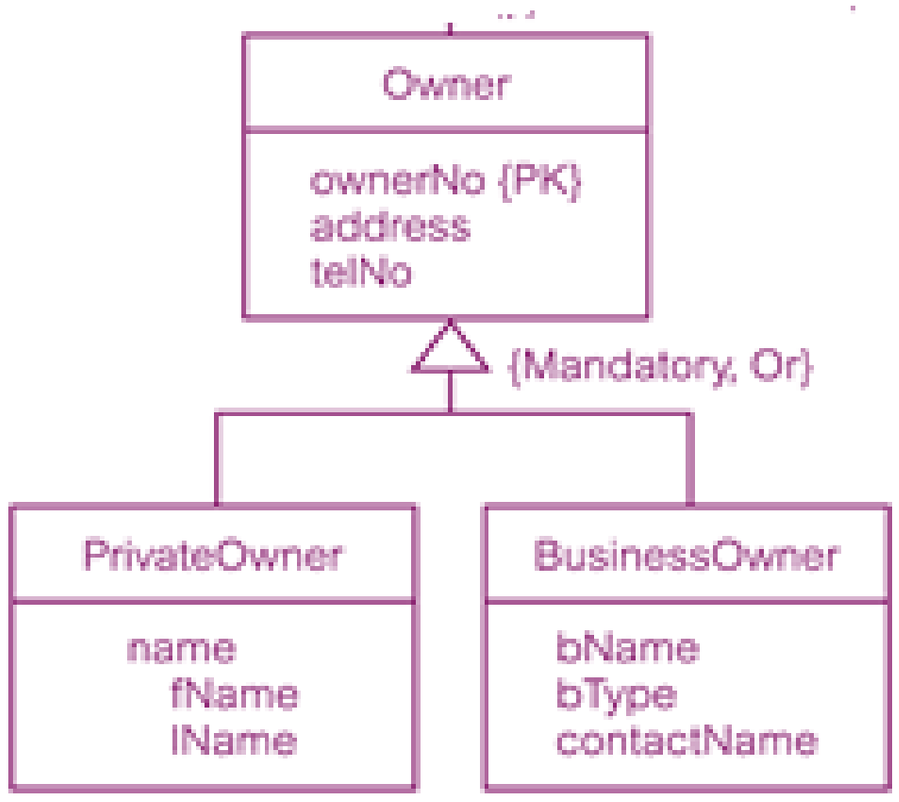
Derive relations for logical data model for superclass/subclass relationship types

- Identify superclass entity as parent entity and subclass entity as the child entity. There are various options on how to represent such a relationship as one or more relations.
- The selection of the most appropriate option is dependent on a number of factors such as the **disjointness** and **participation** constraints on the superclass/subclass relationship, whether the subclasses are involved in distinct relationships, and the number of participants in the superclass/subclass relationship.

Guidelines for representation of superclass / subclass relationship

Participation constraint	Disjoint constraint	Relations required
Mandatory	Nondisjoint {And}	Single relation (with one or more discriminators to distinguish the type of each tuple)
Optional	Nondisjoint {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)
Mandatory	Disjoint {Or}	Many relations: one relation for each combined superclass/subclass
Optional	Disjoint {Or}	Many relations: one relation for superclass and one for each subclass

Guidelines for representation of superclass / subclass relationship



Guidelines for representation of superclass / subclass relationship

- Consider the *Owner* superclass/subclass relationship type (previous slide). As we have seen there are various ways to represent this relationship type as one or more relations.

- The options range from placing all the attributes into one relation with 2 discriminators *pOwnerFlag* and *bOwnerFlag* indicating whether a record belongs to a particular subclass (Option 1) to dividing the attributes into three relations (Option 4).

Guidelines for representation of superclass / subclass relationship

- In this case the most appropriate representation of the superclass/subclass relationship type is determined by the constraints on this relationship type.

- The relationship type that the *Owner* superclass has with its subclasses is *mandatory* and *disjoint*, as each member of the *Owner* superclass must be a member of one of the subclasses but cannot belong to both.

- We therefore select Option 3 as the best representation of this relationship type and create a separate relation to represent each subclass, and include the attributes of the superclass in both.

Guidelines for representation of superclass / subclass relationship

Option 1 – Mandatory, nondisjoint

AllOwner (ownerNo, address, telNo, fName, lName, bName, bType, contactName, pOwnerFlag, bOwnerFlag)
Primary Key ownerNo

Option 2 – Optional, nondisjoint

Owner (ownerNo, address, telNo)
Primary Key ownerNo

OwnerDetails (ownerNo, fName lName, bName, bType, contactName, pOwnerFlag, bOwnerFlag)
Primary Key ownerNo
Foreign Key ownerNo **references** Owner(ownerNo)

Option 3 – Mandatory, disjoint

PrivateOwner (ownerNo, fName, Name, address, telNo)
Primary Key ownerNo

BusinessOwner (ownerNo, bName, bType, contactName, address, telNo)
Primary Key ownerNo

Option 4 – Optional, disjoint

Owner (ownerNo, address, telNo)
Primary Key ownerNo

PrivateOwner (ownerNo, fName, Name)
Primary Key ownerNo
Foreign Key ownerNo **references** Owner(ownerNo)

BusinessOwner (ownerNo, bName, bType, contactName)
Primary Key ownerNo
Foreign Key ownerNo **references** Owner(ownerNo)