Databases

From Entity-Relationship (ER) model to Enhanced ER model

Dr. George Mertzios
Michaelmas Term

george.mertzios@durham.ac.uk
Room 2066, MCS Building
Tel: 42 429

Course Outline

- Enhanced Entity-Relationship (EER) Model
- Semistructured Databases - XML
- XML Data Manipulation - XPath, XQuery
- Transactions and Concurrency Control
- Distributed Transactions
- Distributed Concurrency Control

Data models

- Data Model:
  - collection of intuitive concepts describing *data*, their *relationships* and *constraints*
- We have seen:
  - Relational Data Model
    - relations are tables (columns + rows),
    - attributes are columns, tuples are rows
  - Sometimes too low level for big companies:
    - designers, programmers, end users understand data and its use in different ways
  - We need a model of communication that is non-technical and free of ambiguities

    $\Longrightarrow$ Entity-Relationship (ER) model

Entity-Relationship (ER) model

- Top-down approach to database design
  - graphical description of the DB
- Basic concepts:
  - the important data objects (entities)
  - the important properties of the entities (attributes)
  - the associations between the entities (relationships)
- Furthermore:
  - constraints on the entities, relationships, and attributes
- Several notations for representing the ER model
  - Unified Modeling Language (UML)
    (most popular diagrammatic notation)

Entities

- Entity type: group of objects with the same properties, identified as having independent existence, e.g.: ‘Client’

- Entity occurrence: a uniquely identifiable instance of an entity type, e.g.: a specific Client called ‘James Smith’
  - We use ‘entity’ when the meaning is clear from the context

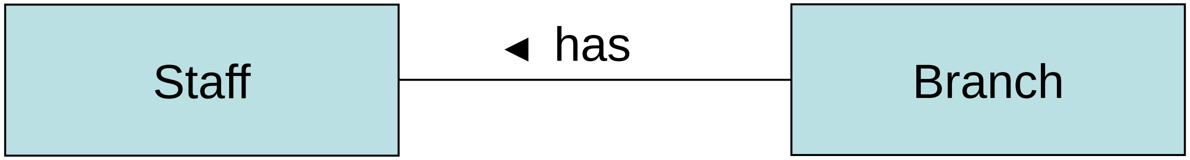
- Diagrammatic representation of entities:
  - a rectangle labeled by the *name* of the entity type

Staff

Branch

Relationships

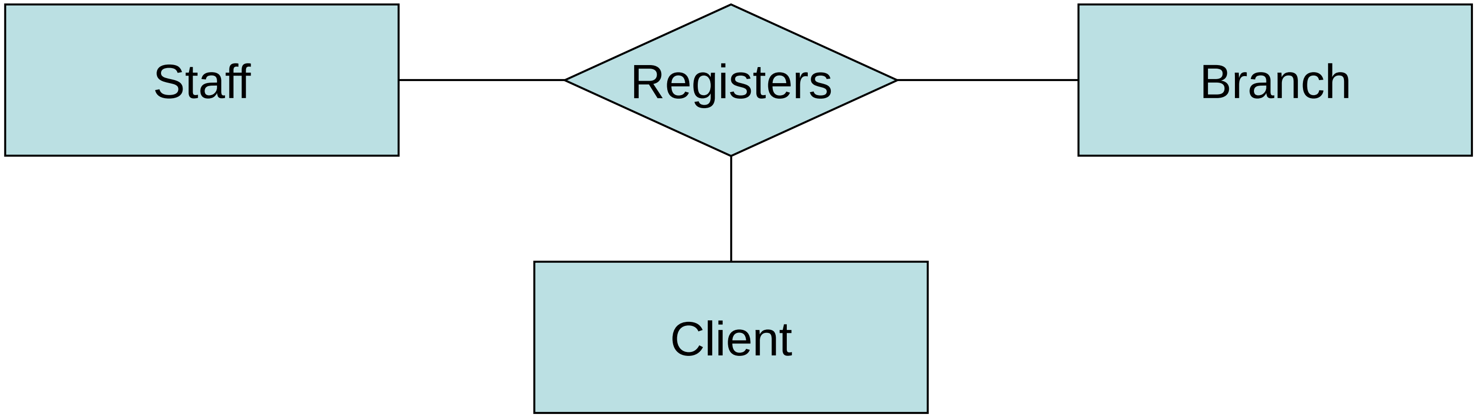
- <u>Relationship (type):</u> a meaningful association between two (or more) entity types
- <u>Degree</u> of a relationship: the number of entities that participate in a relationship
  - degree = n ⇒ 'n-ary' relationship
  - degree = 2 ⇒ 'binary'
  - degree = 3 ⇒ 'ternary' – degree = 4 ⇒ 'quaternary'
- Diagrammatic representation of *binary* relationships:
  - a line connecting the participating entities, labeled by the *name* of the relationship (has also a *direction*)



'Branch **<u>has</u>** Staff'

Relationships

- Diagrammatic representation of *n-ary* relationships (where $n \geq 3$):
  - a diamond labeled by the *name* of the relationship, connecting the participating entities

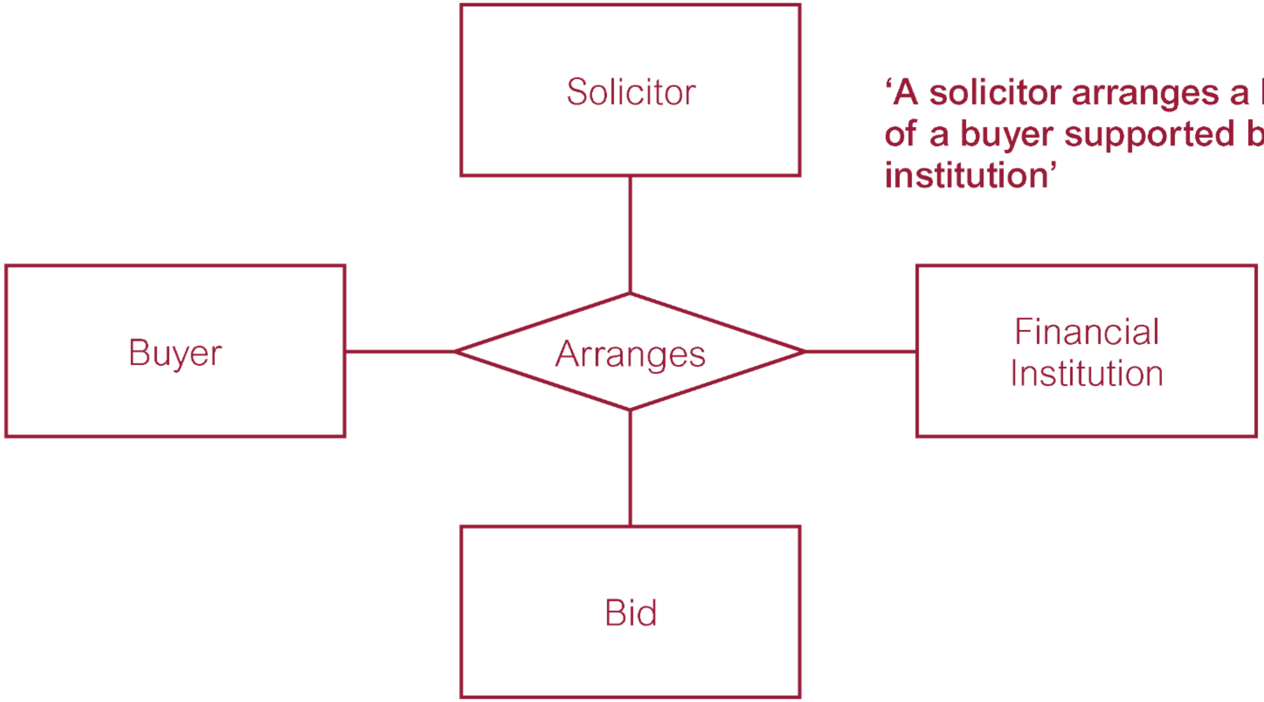


‘Staff **registers** a client at a Branch’

  - not necessarily a direction

Relationships

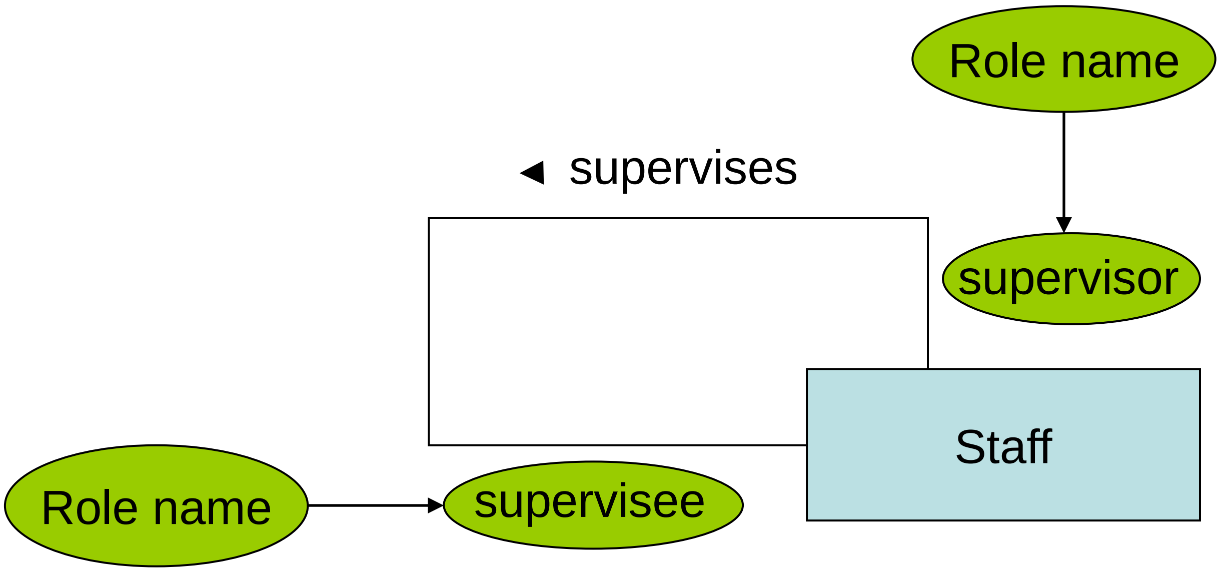
- Diagrammatic representation of *n-ary* relationships (where $n \geq 3$):
  - – a quaternary example (n=4):



'A solicitor arranges a bid on behalf of a buyer supported by a financial institution'

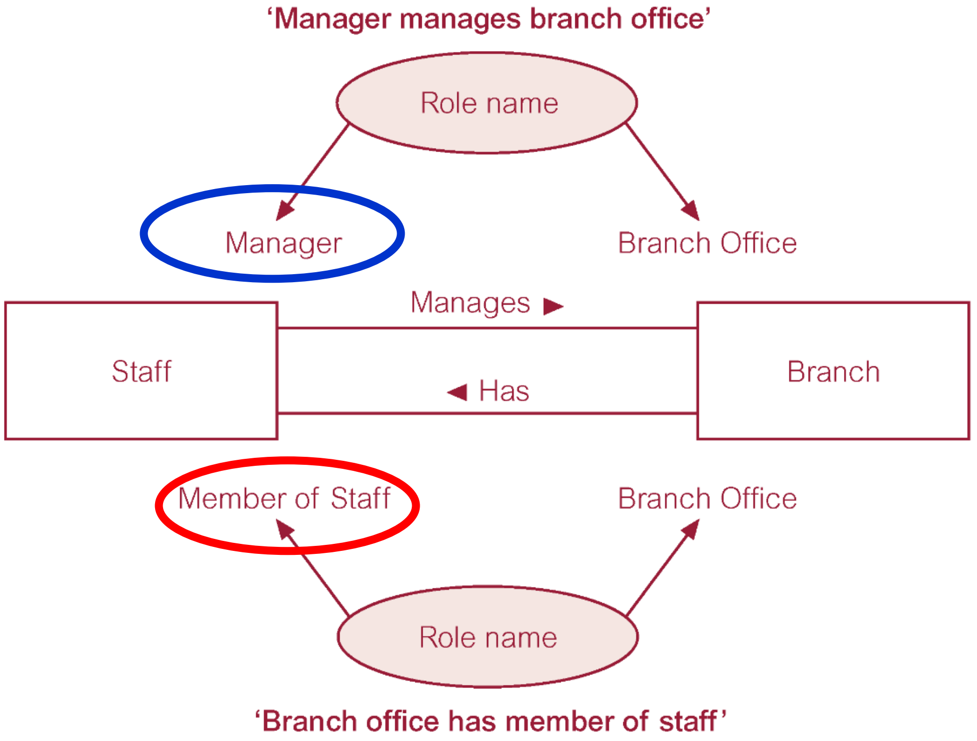
Relationships

- Relationships with degree = 1:
  - ‘unary’ or ‘recursive’ relationships
- Example: ‘Staff (supervisor) supervises Staff (supervisee)’
  - we need role names (‘supervisor’, ‘supervisee’) to indicate the purpose of each entity occurrence in the relationship



Relationships

- Role names:
  - can also be used when entities are associated through more than one relationships
  - for clarifying the purpose of each relationship
- Example:



Attributes

- Attribute: a descriptive property of an entity or relationship
- Attribute domain: the set of allowable attribute values
- Attributes may be:
  - simple / composite (e.g.: ‘salary’ / ‘address’)
  - single-valued / multi-valued (e.g.: ‘staffNo’ / ‘telNumbers’)
  - sometimes derived (e.g.: ‘age’ is derived by ‘dateOfBirth’)
  - Null valued
- Candidate key:
  - a minimal set of attributes, whose values uniquely identify an entity occurrence

  ⇒ cannot be null
- Primary key:
  - we choose exactly one candidate key

Attributes

- Simple (composite) key:
  - a candidate key that consists of one (many) attribute(s)

- Factors for the choice of primary key:
  - number of attributes (preferably smaller)
  - attribute length (preferably smaller)
  - future certainty of uniqueness

Example:

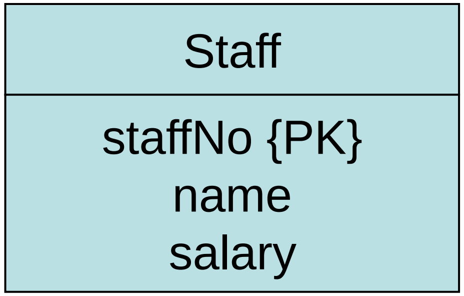
- the company-defined StaffNo: max. 5 characters (SG14)
- the NIN: max 9 characters (WL220658D)

⟹ StaffNo is preferable

Attributes on Entities

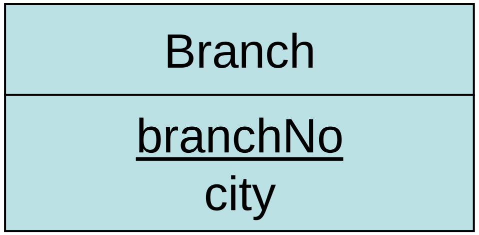
Diagrammatic representation:

- Divide the rectangle of the entity into two parts:
  - the upper part has the entity name
  - the lower part has a list of the attributes
- The primary key is usually:
  - underlined, or
  - labeled with the tag {PK}


Staff

staffNo {PK}
name
salary

or:

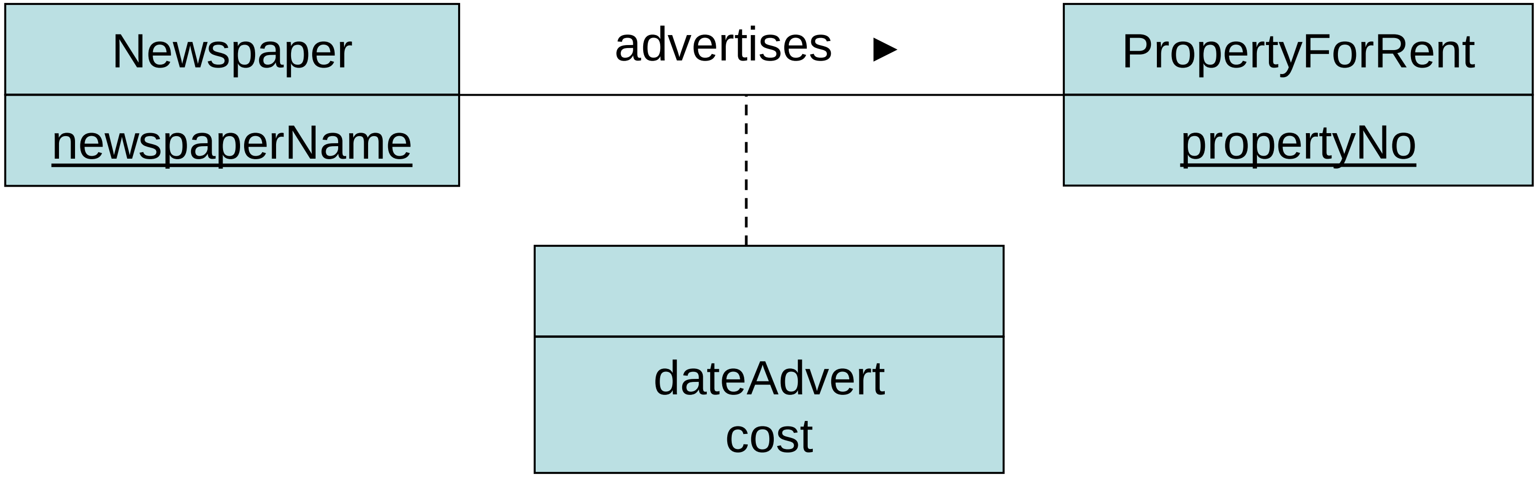

Branch

branchNo
city

Attributes on Relationships

Diagrammatic representation:

- a labeled rectangle with two parts (as for entities)
  - the upper part is empty
  - the lower part has a list of the relationship attributes
- the rectangle is connected by a dashed line with the relationship



Multiplicity

- Multiplicity of a relationship (type):

  Number of entity occurrences, to which another entity can be associated with this relationship

- Relationships can be:
  - one-to-one (1:1),

    e.g.: ‘Staff manages Branch’
  - one-to-many (1:*),

    e.g.: ‘MathTeacher teaches Student’
  - many-to-many (*:*),

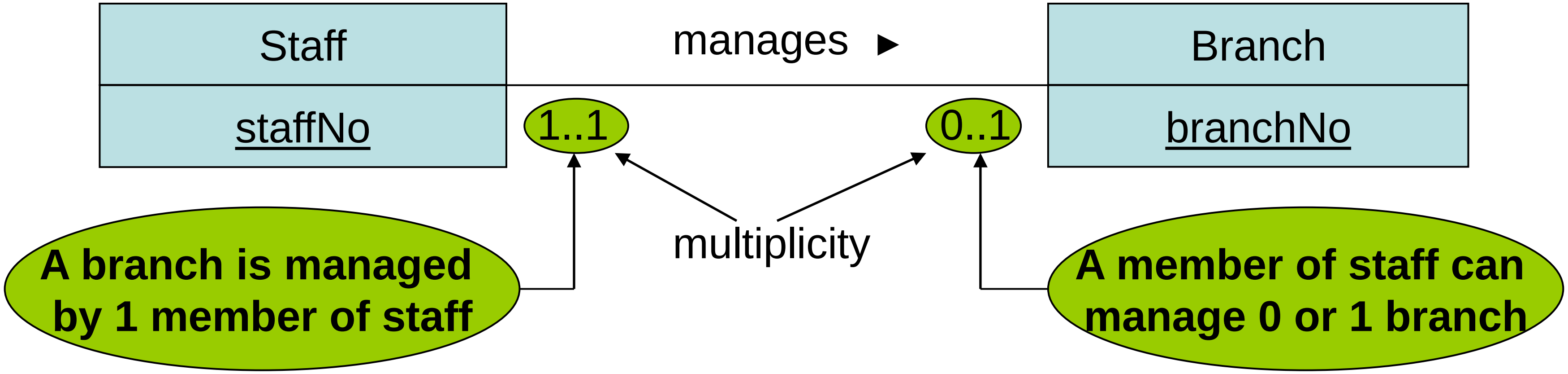
    e.g.: ‘Newspaper advertises PropertyForRent’

- Do we need many many-to-one (*:1) relationships?

Multiplicity

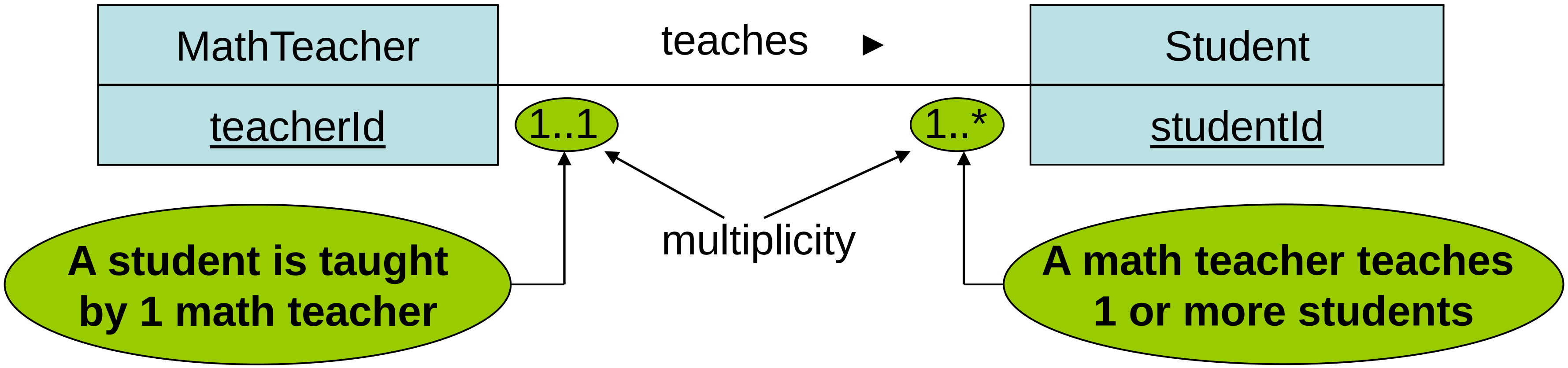
Diagrammatic representation:

- we write the minimum / maximum number of occurrences of each entity in the relation
- example (1:1 relationship):


Staff
staffNo
manages ▶
Branch
branchNo
1..1
0..1
multiplicity
A branch is managed by 1 member of staff
A member of staff can manage 0 or 1 branch

Multiplicity

- example: (1:*) relationship:



MathTeacher

teacherId

teaches ▸

Student

studentId

1..1

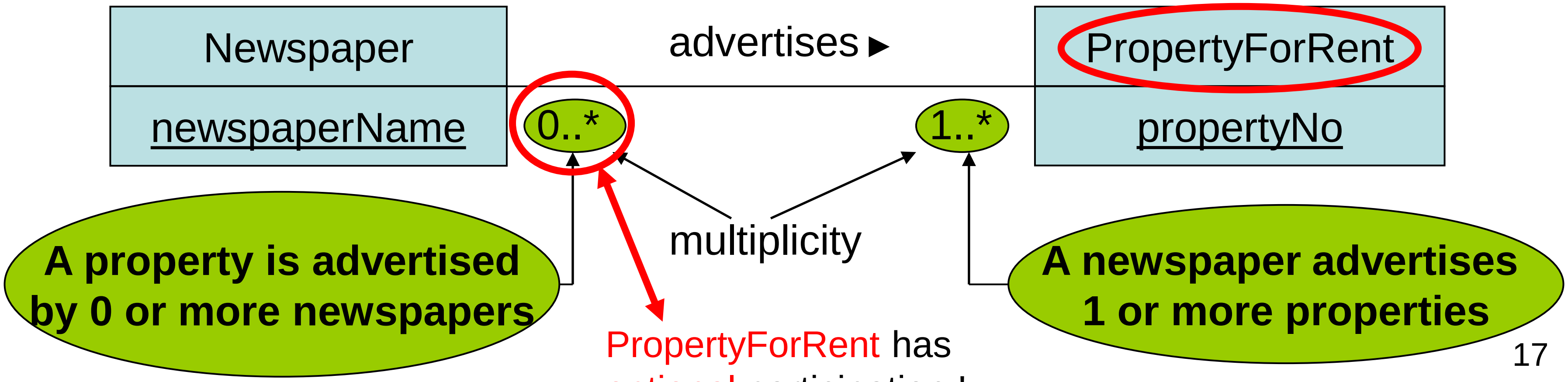
1..*

multiplicity

A student is taught by 1 math teacher

A math teacher teaches 1 or more students

- example: (*:*) relationship:



Newspaper

newspaperName

advertises ▸

PropertyForRent

propertyNo

0..*

1..*

multiplicity

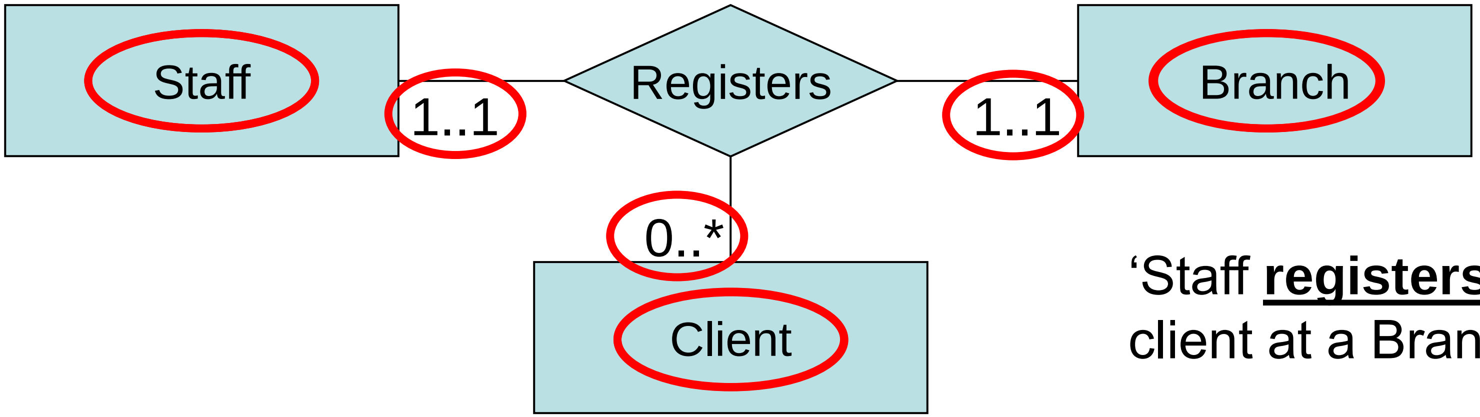
A property is advertised by 0 or more newspapers

A newspaper advertises 1 or more properties

PropertyForRent has optional participation !

Multiplicity of *n*-ary relationships

- Multiplicity of an *n*-ary relationship:
  - number of entity occurrences, when the *(n-1)* values are fixed for the other entity types

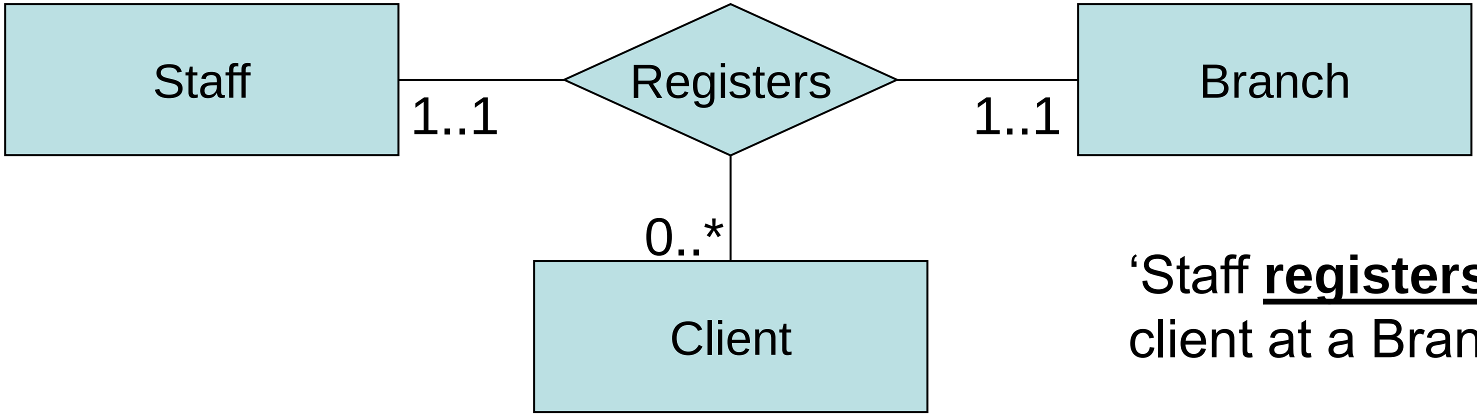


'Staff **registers** a client at a Branch'

- fixed values StaffNo / BranchNo: ⇒ zero or more ClientNo values
- fixed values StaffNo / ClientNo: ⇒ exactly one BranchNo value
- fixed values ClientNo / BranchNo: ⇒ exactly one StaffNo value

Multiplicity of *n*-ary relationships

- Multiplicity of an *n*-ary relationship:
  - number of entity occurrences, when the *(n-1)* values are fixed for the other entity types



'Staff **registers** a client at a Branch'

The multiplicity of the relationship Registers is:

– with respect to Staff and Branch: 0..* ("many")

– with respect to Staff and Client: 1..1 ("one")

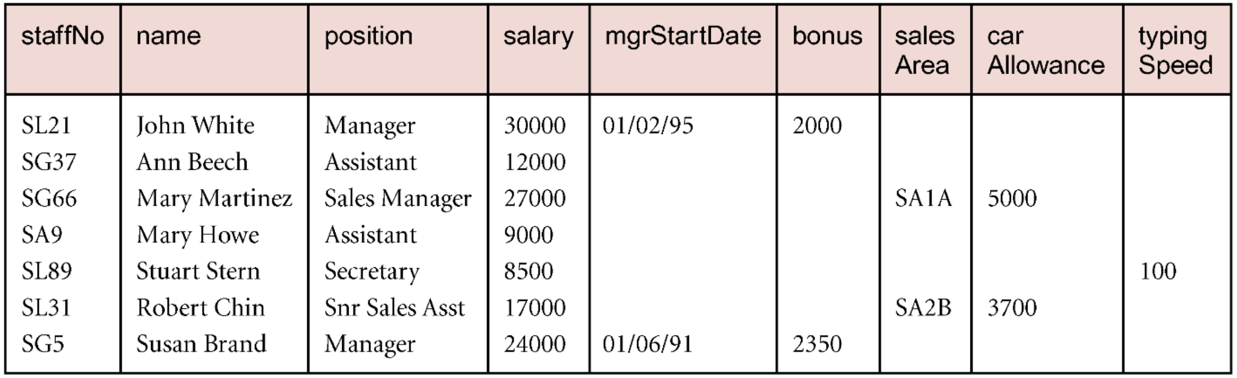
– with respect to Client and Branch: 1..1 ("one")

Problems with the ER model

- The basic concepts of the ER model (entities, attributes, relationships) are normally adequate for building data models in traditional DBs

- However: **limiting when modeling <u>modern</u> and <u>complex</u> DB applications with <u>large</u> amounts of data**

Example:



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			

Problems with the ER model

- The basic concepts of the ER model (entities, attributes, relationships) are normally adequate for building data models in traditional DBs

- However: **limiting when modeling modern and complex DB applications with large amounts of data**

- Solution: additional semantic modeling concepts
  - reduce the complexity of the ER model
  - more intuitive
  - $\Longrightarrow$ Enhanced Entity Relationship (EER) model

- The main concepts of the EER model are:
  - specialization
  - generalization

The Enhanced ER (EER) model

- Subclass: a subgrouping of occurrences of an entity type, which requires to be represented separately
- Superclass: an entity type that has two or more distinct subclasses

- Example: superclass: *Staff*, subclasses: *Manager*, *Secretary*, …

- Superclass/subclass relationship:
  - each member of a subclass is also a member of the superclass

- Attribute inheritance:
  - all attributes of the superclass are also attributes of the subclasses
  - a subclass has additional attributes than its superclass

The Enhanced ER (EER) model

- Type hierarchy: an entity with its subclasses
  and their subclacces etc.
- Type hierarchy is also known as:
  – Specialization hierarchy
    e.g.: Manager is specialization of Staff
  – Generalization hierarchy
    e.g.: Staff is generalization of Manager
  – IS-A hierarchy
    e.g.: Manager IS-A (member of) Staff
- Main advantages of the EER model:
  – avoid describing similar concepts more than once
  – have relations that include a subclass but not the superclass
  – more semantic information to the design:
    manager IS-A member of Staff, van IS-A type of vehicle

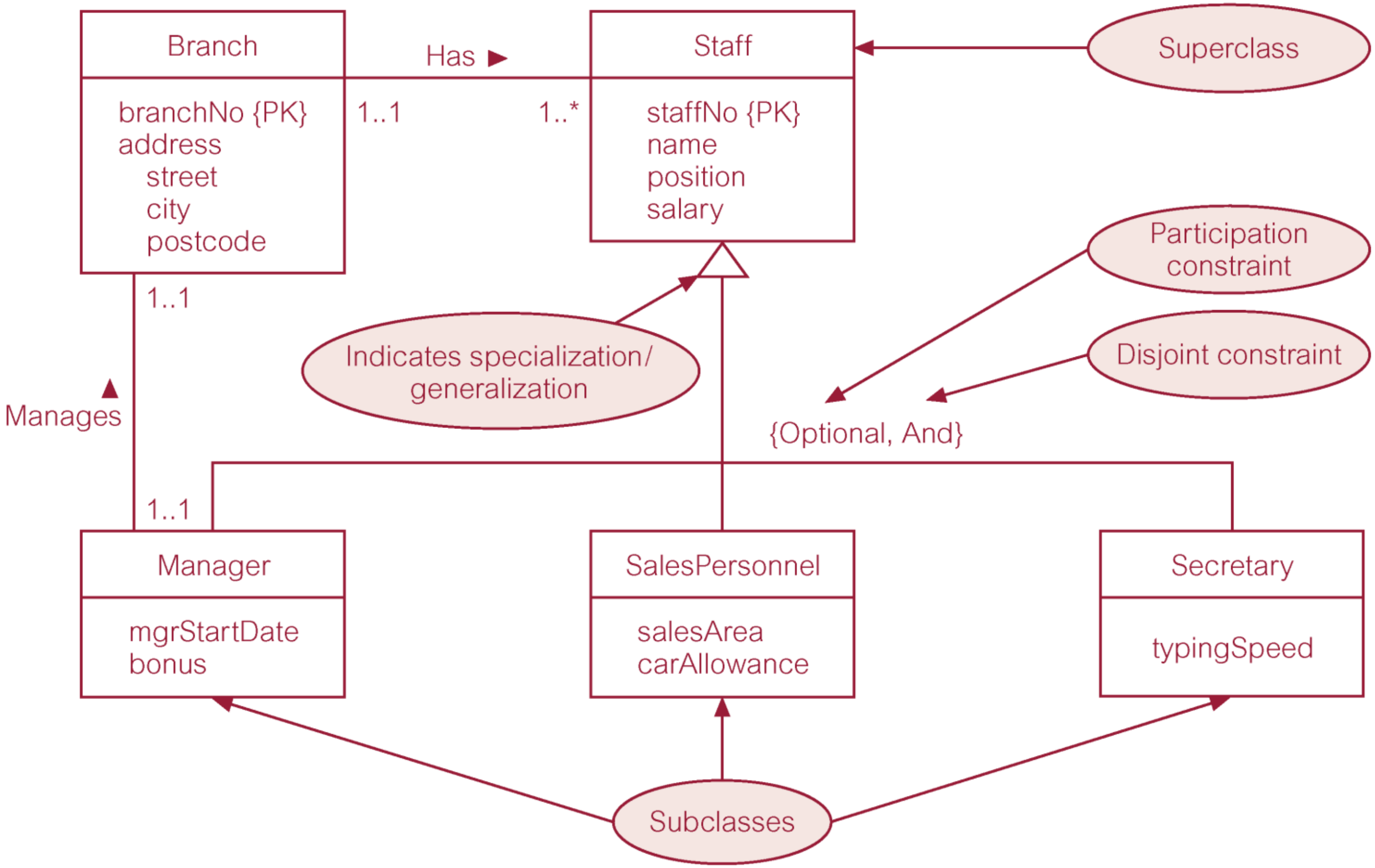
Specialization / Generalization

- Specialization:
  - The top-down process of maximizing the differences between entity occurrences, by identifying their distinguishing characteristics
  - Given superclass(es), it leads to identifying subclasses

- Generalization:
  - The bottom-up process of minimizing the differences between entity occurrences, by identifying their common characteristics
  - Given subclasses, it leads to identifying superclass(es)

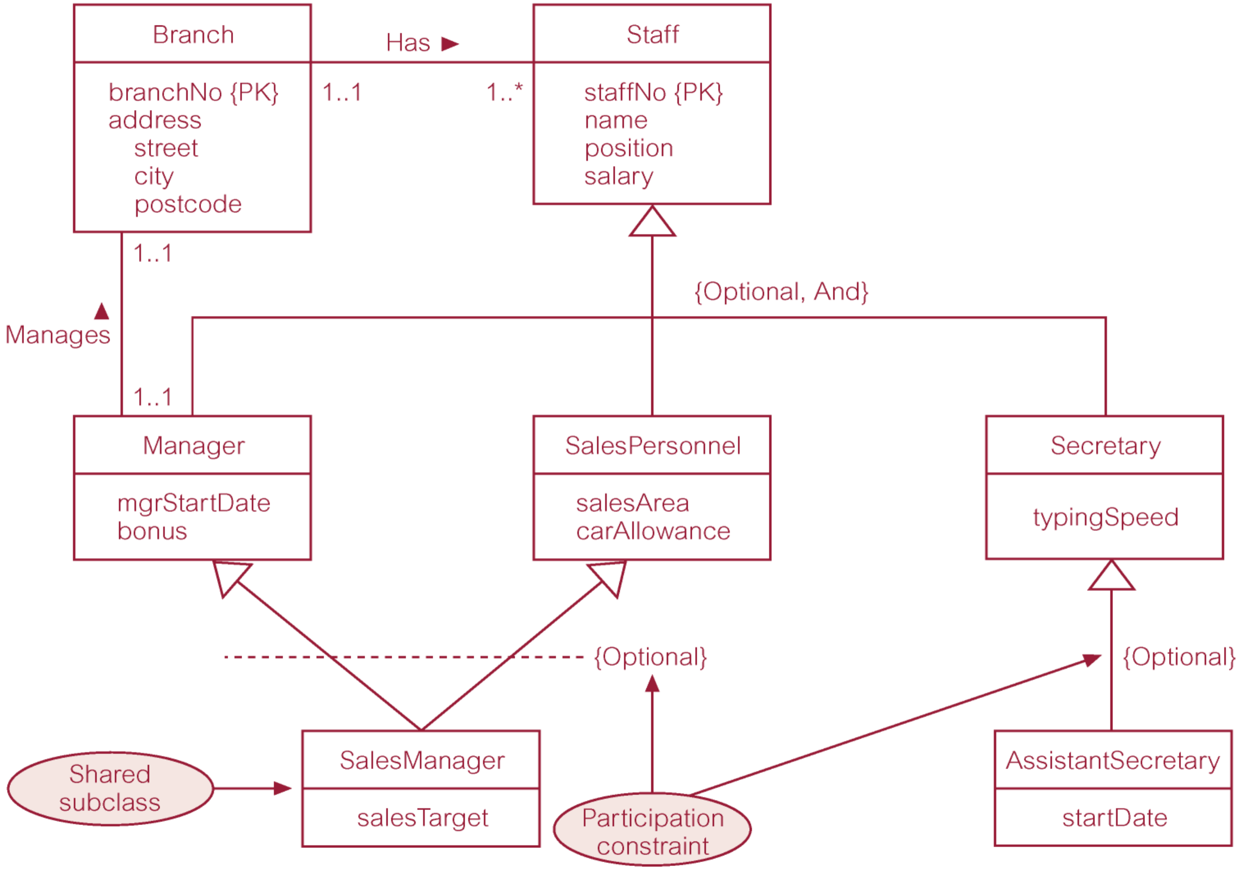
Specialization / Generalization

- Diagrammatic representation:
  - the subclasses are attached by lines to a triangle that points toward the superclass



Specialization / Generalization

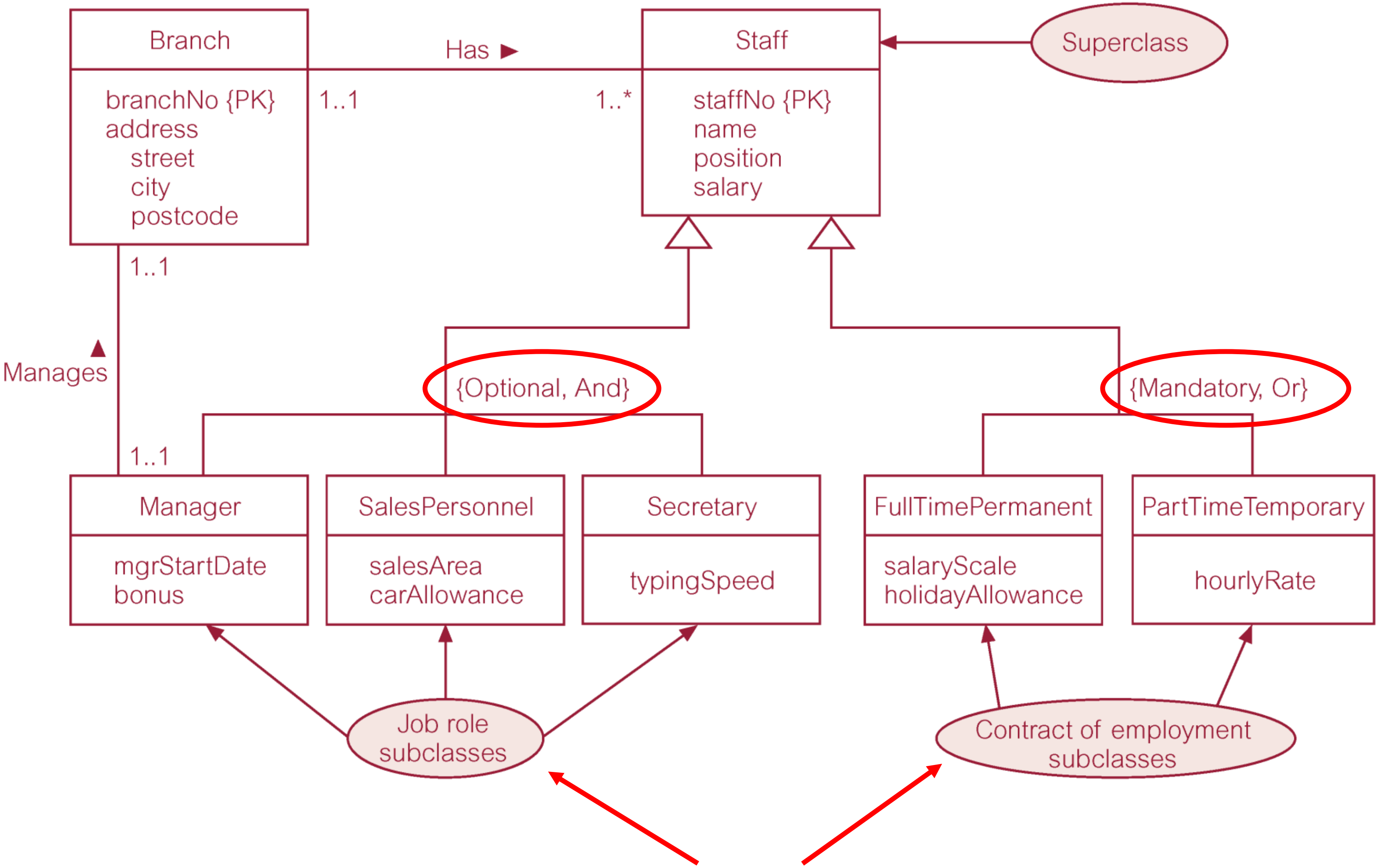
- An extended version:
  - a shared subclass
  - a subclass with its own subclass



Constraints

- Participation constraint:
  - determines whether *every* member in the superclass must participate as a member of a subclass or not
  - can be **mandatory** or **optional**
- Disjoint constraint:
  - determines whether a member of a superclass can be a member of *one* or *more* subclasses
  - only applies in case of at least two subclasses
    - **or** (i.e. disjoint: it can belong to only one subclass)
    - **and** (i.e. non-disjoint: it can belong to more subclasses)
- Diagrammatically:
  - we write the values of the above constraints as a label below the specialization / generalization triangle

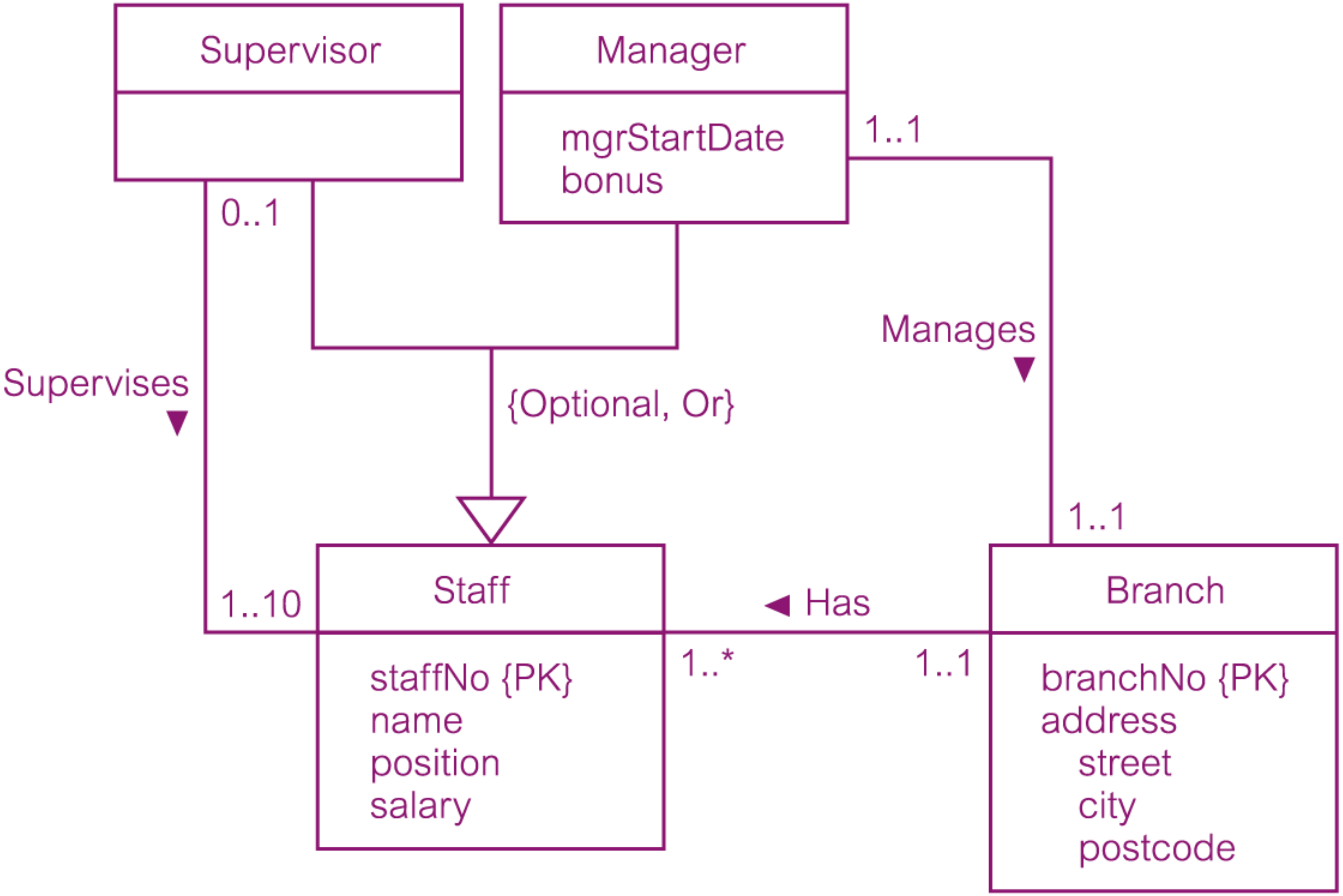
Constraints



two different specializations

Further example

- Relationship between subclass and superclass:



Summary of the Lecture

- Overview of the Entity-Relationship (ER) model
  - Entities
  - Attributes
  - Relationships
  - Diagrammatic representation
  - Multiplicity of relations

- Additional semantic concepts to the ER model: the Enhanced ER (EER) model
  - Subclasses / Superclasses
  - Attribute inheritance
  - Type hierarchy
  - Specialization / Generalization
  - Participation and Disjoint constraints
  - Diagrammatic representation

Next time

- Enhanced Entity-Relationship (EER) Model
- **Semistructured Databases - XML**
- XML Data Manipulation - XPath, XQuery
- Transactions and Concurrency Control
- Distributed Transactions
- Distributed Concurrency Control