Entity-Relationship Diagram

Truong Tuan Anh
CSE-HCMUT

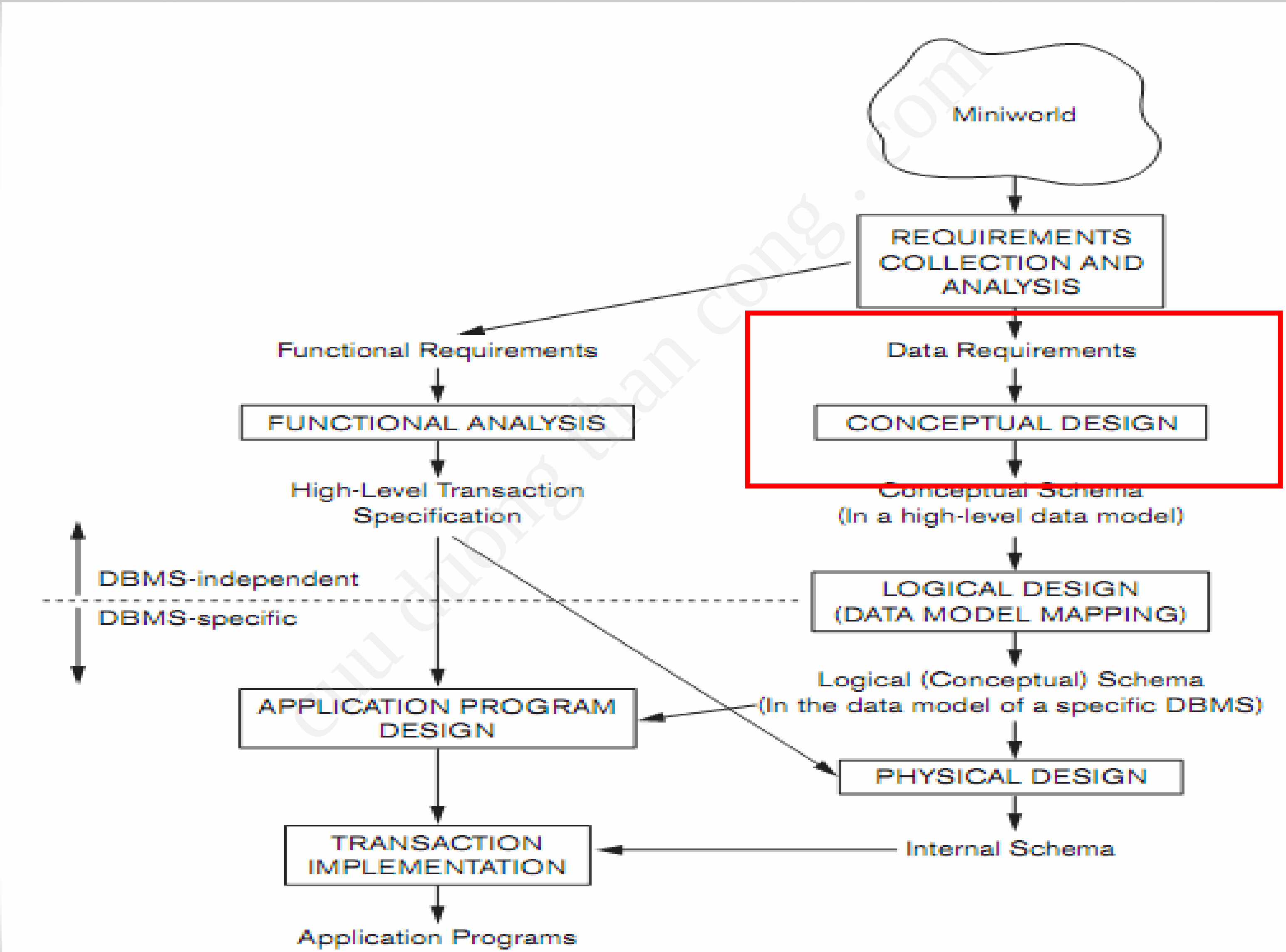
Contents

- Database design process
- ER Model

Database Design Process

- Two main activities:
  - Database design
  - Applications design
- Database design
  - To design the conceptual schema for a database application
- Applications design
  - Programs and interfaces that access the database
  - Generally considered part of software engineering

Database Design Process



Database Design Process

- Collect and Analyze requirements
  - Database designers interview prospective database users to understand and document data requirements
  - Outputs:
    - Data requirements
    - Functional requirements

Database Design Process

- Conceptual design
  - Create a conceptual schema for the database.
    - Description of data requirements
    - Uses the concepts provided by the high-level data model
    - Includes detailed descriptions of the entity types, relationships, and constraints
    - Independent of storage and implementation details.

Database Design Process

- Logical design or data model mapping
  - Output is a database schema in implementation data model of DBMS
- Physical design phase
  - Internal storage structures, file organizations, indexes, access paths, and physical design parameters for the database files specified

ER Model

What is ER Model?

- Entity-Relationship (ER) model
  - Popular high-level conceptual data model
  - A logical organisation of data within a database system
- ER diagrams:
  - Diagrammatic notation associated with the ER model

Why ER Model?

- User requirements can be specified formally & unambiguously
- The conceptual data model is independent of any particular DBMS
- It does not involve any physical or implemental details
- It can be easily understood by ordinary users.
- It provides an effective bridge between user requirements and logical database design and implementation

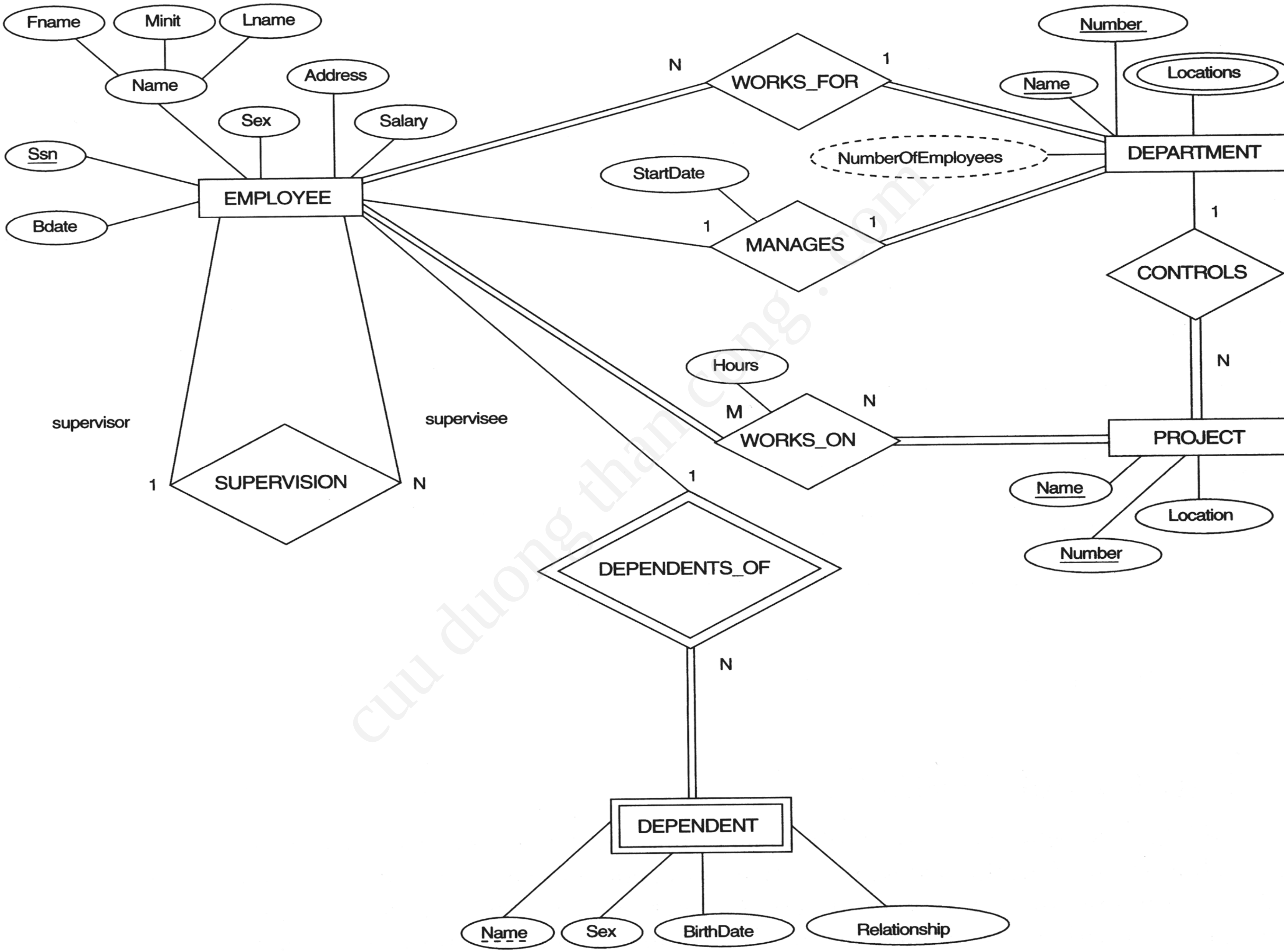
A Sample Database Application

- The **COMPANY** database: keeps track of **employees, departments, and projects.**

- The company is organized into DEPARTMENTs.
  - Each department has a unique name, a unique number, and a particular employee who *manages* the department.
  - We keep track of the start date when that employee began managing the department.
  - A department may have several locations.
- A department controls a number of PROJECTs
  - Each of which has a unique name, a unique number, and a single location.

A Sample Database Application

- We store EMPLOYEE's name, Social Security number, address, salary, sex, and birth date.
  - *An employee is assigned to one department, but may work on several projects*, which are not necessarily controlled by the same department. We keep track of the current number of hours per week that an employee works on each project.
  - We also keep track of the direct supervisor of each employee.
- We want to keep track of the DEPENDENTs of each employee, including first name, sex, birth date, and relationship to the employee.

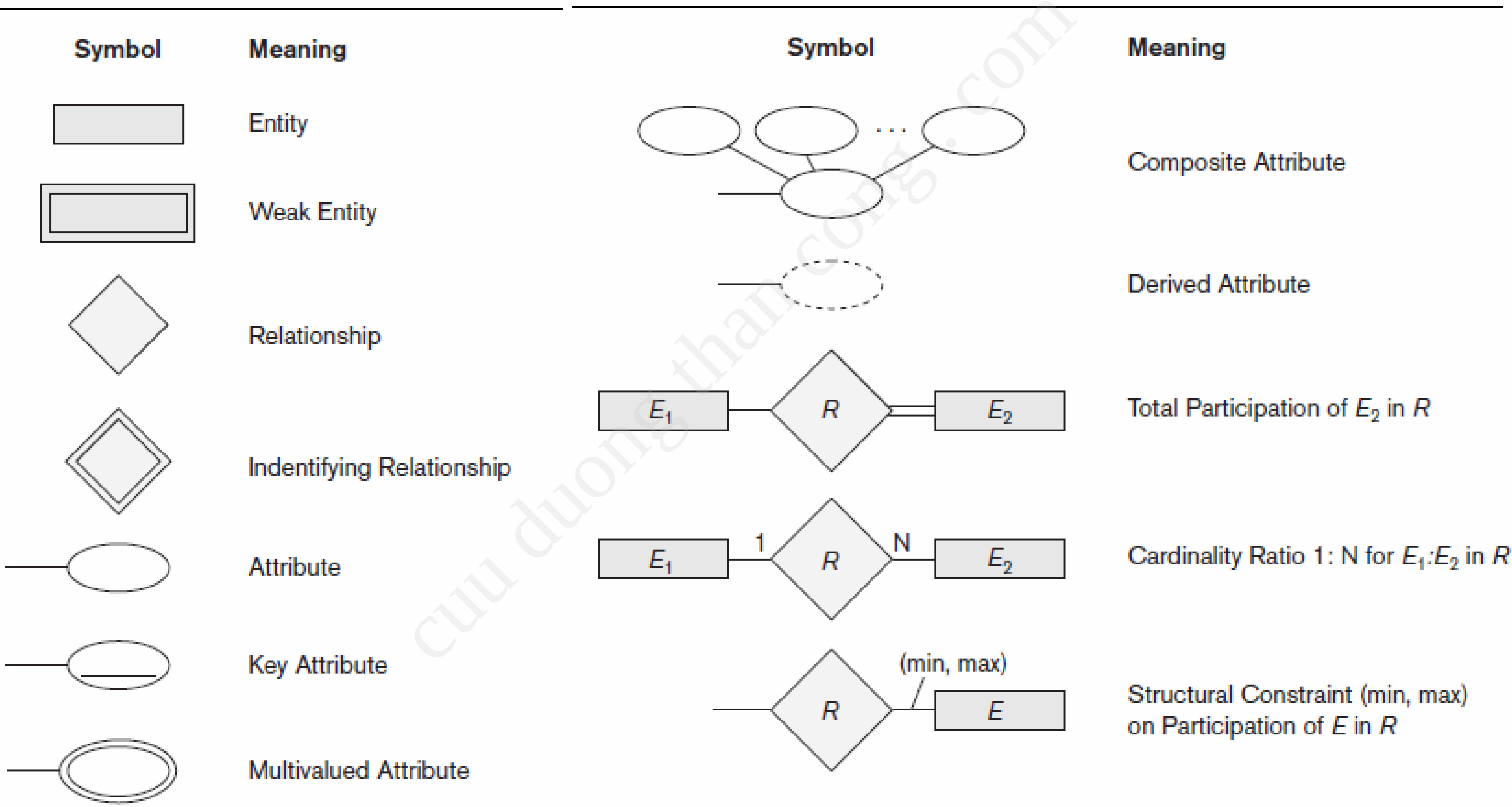

Fname
Minit
Lname
Name
Address
Sex
Salary
Ssn
Bdate
EMPLOYEE
WORKS_FOR
N
1
Number
Name
Locations
DEPARTMENT
NumberOfEmployees
StartDate
MANAGES
1
1
1
CONTROLS
N
Hours
WORKS_ON
M
N
PROJECT
Name
Location
Number
supervisor
supervisee
SUPERVISION
1
N
1
DEPENDENTS_OF
N
DEPENDENT
Name
Sex
BirthDate
Relationship

ER Model Concepts

ER Model Concepts

- ER model describes data as:
  - Entities
  - Relationships
  - Attributes

ER Diagram: Summary

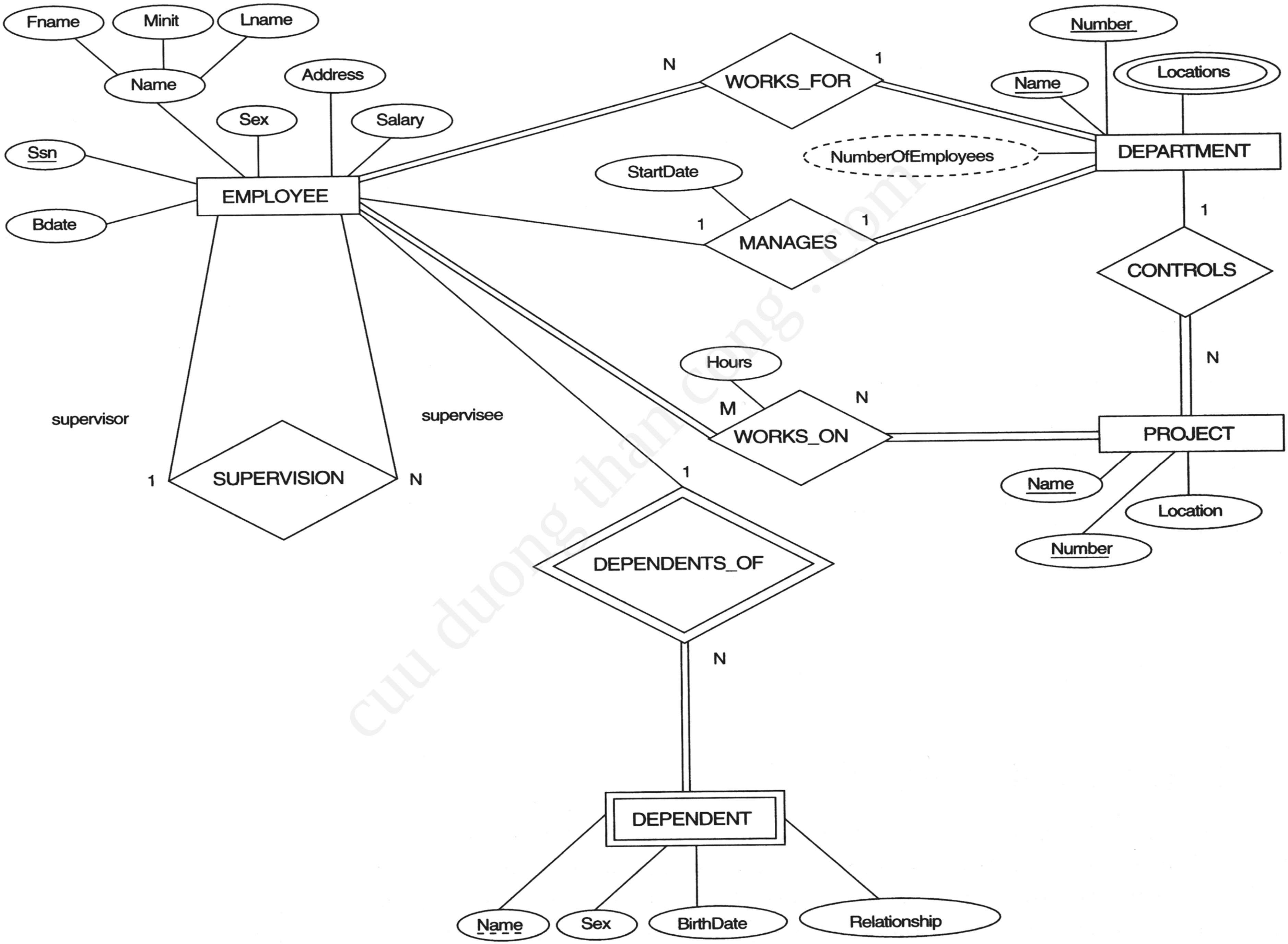


Entities and Attributes

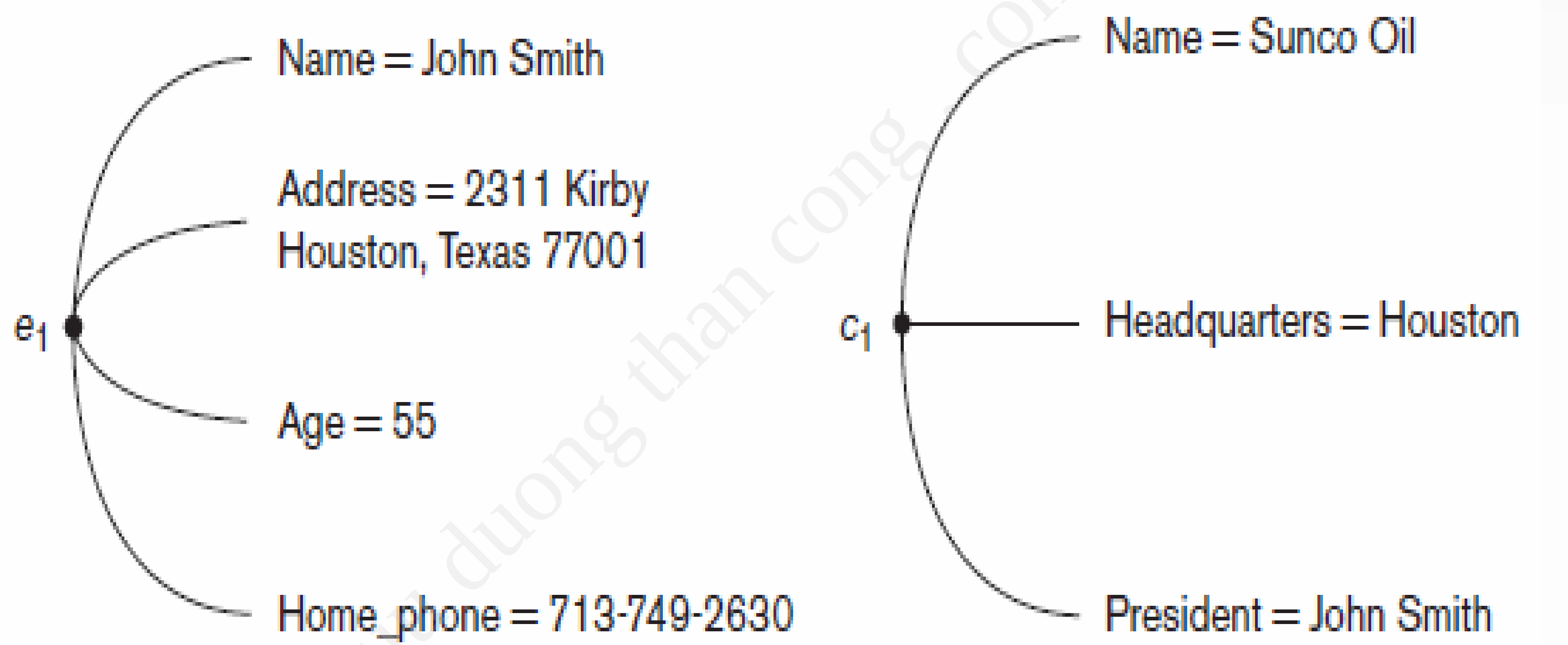
- **Entity** is a thing in the real world with an independent existence.
  - Ex: the EMPLOYEE John Smith, the Research DEPARTMENT, the ProductX PROJECT
- **Attributes** are properties describing an entity.
  - Ex: an EMPLOYEE entity may have Name, SSN, Address, Sex, BirthDate
- A specific entity will have a value for each of its attributes
- Each attribute has a **value set** (or data type) associated with it.

Entities and Attributes

- **Types of Attributes**
  - **Simple attributes:** each entity has a single atomic value for the attribute.
  - **Composite attributes:** attribute may be composed of several components.
  - **Multi-valued attributes:** an entity may have multiple values for that attribute.
  - **Derived:** attribute represents a value that is derivable from value of a related attribute, set of attributes, or relationships.
  - **Complex attributes:** composite and multivalued attributes can be nested arbitrarily


Fname
Minit
Lname
Name
Address
Sex
Salary
Ssn
EMPLOYEE
Bdate
N
WORKS_FOR
1
Number
Name
Locations
DEPARTMENT
NumberOfEmployees
StartDate
1
MANAGES
1
1
CONTROLS
N
Hours
M
WORKS_ON
N
PROJECT
Name
Location
Number
supervisor
supervisee
1
SUPERVISION
N
1
DEPENDENTS_OF
N
DEPENDENT
Name
Sex
BirthDate
Relationship

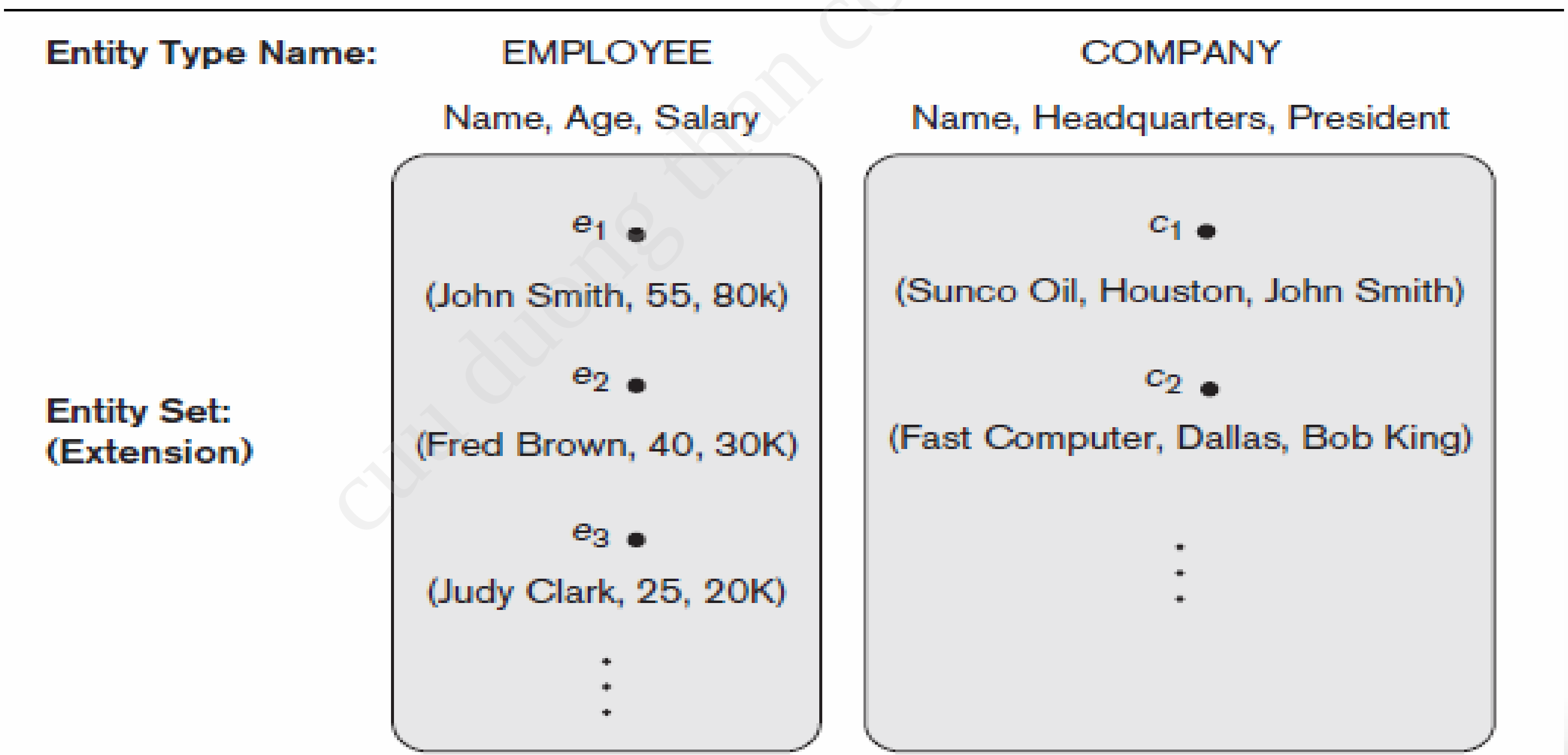
Entities and Attributes



Two entities, EMPLOYEE e1, and COMPANY c1, and their attributes.

Entity Types and Keys

- Entity type
  - Collection (or set) of entities that have the same attributes



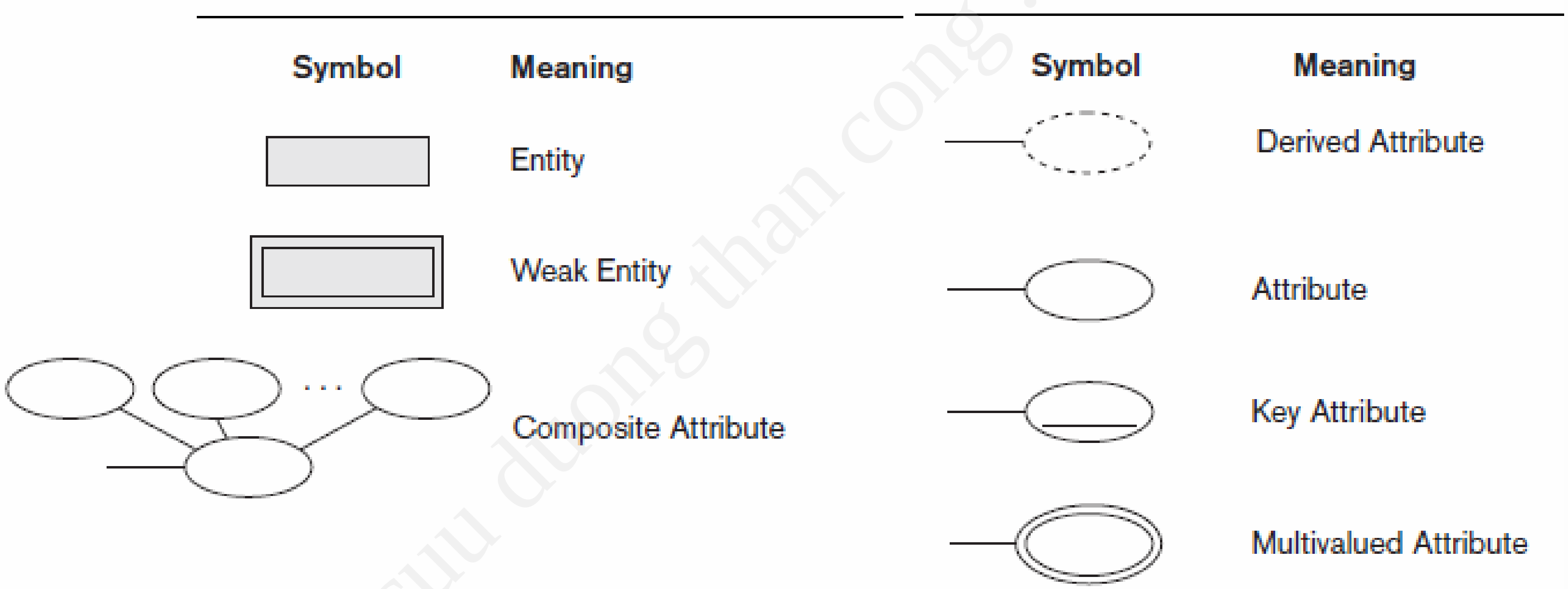
Entity Types and Keys

- **Key** or **uniqueness constraint**
  - Attributes whose values are distinct for each individual entity in entity set
  - Uniqueness property must hold for *every entity set* of the entity type
  - Ex: SSN of EMPLOYEE
- An entity type may have more than one key
  - Ex: the STUDENT entity type may have two keys (in university context):
    - Citizen ID and
    - Student ID

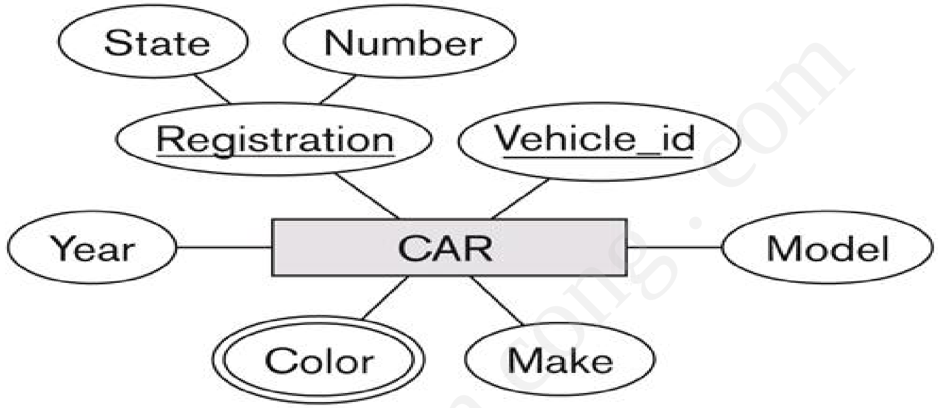
Entity Types and Keys

- **Super key:** A set of attributes (one or more) that together define an entity in an entity set
- **Candidate key:** A minimal super key, meaning it has the least possible number of attributes to still be a super key. An entity set may have more than one candidate key
- **Primary key:** A candidate key chosen by the database designer to uniquely identify the entity set

ERD: Entity


Symbol
Meaning
Entity
Weak Entity
Composite Attribute
Symbol
Meaning
Derived Attribute
Attribute
Key Attribute
Multivalued Attribute

Entity Type: Example



CAR
Registration (Number, State), Vehicle_id, Make, Model, Year, {Color}

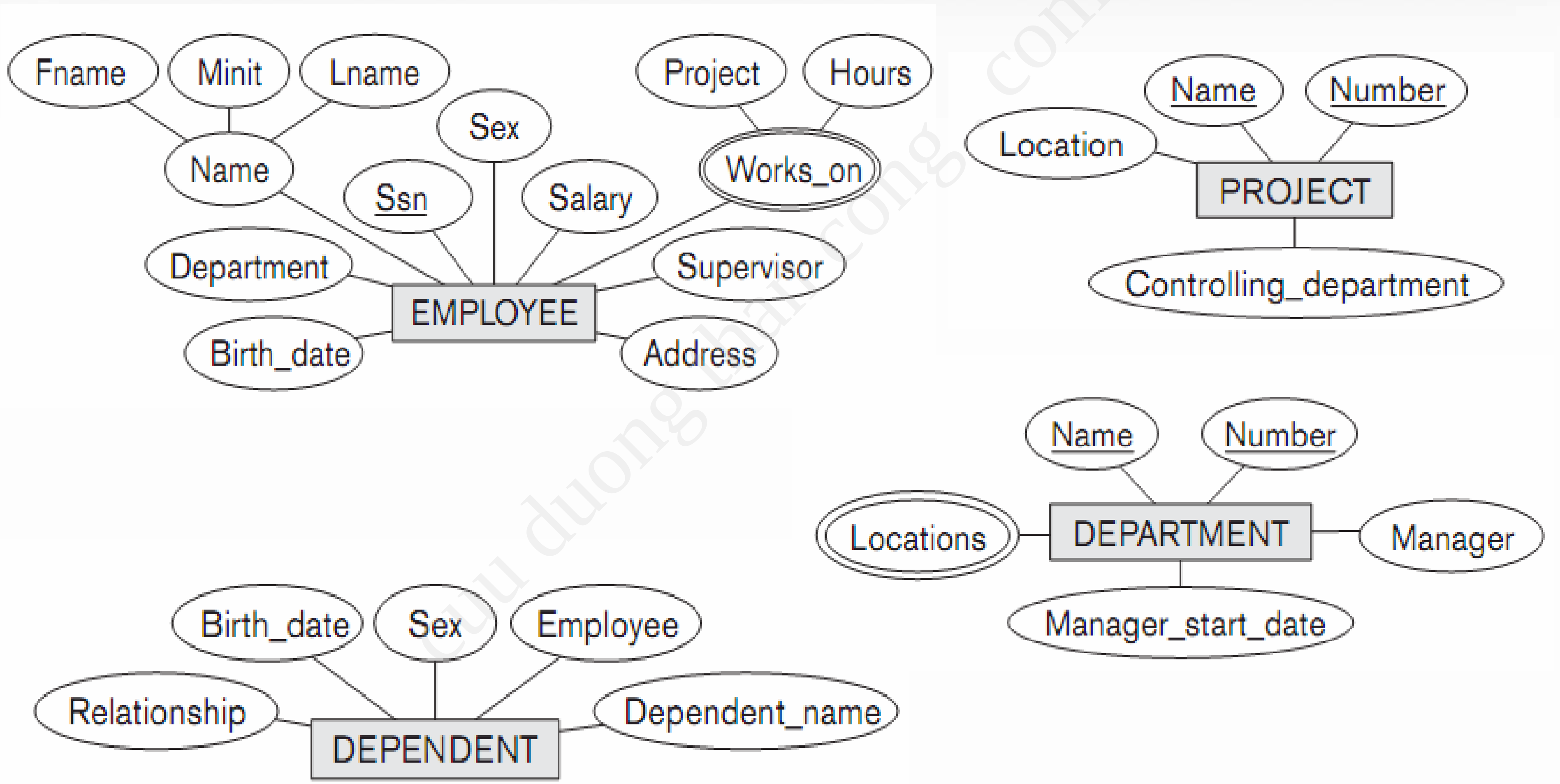
CAR_1
((ABC 123, TEXAS), TK629, Ford Mustang, convertible, 2004 {red, black})

CAR_2
((ABC 123, NEW YORK), WP9872, Nissan Maxima, 4-door, 2005, {blue})

CAR_3
((VSY 720, TEXAS), TD729, Chrysler LeBaron, 4-door, 2002, {white, blue})

⋮

Example: COMPANY Database



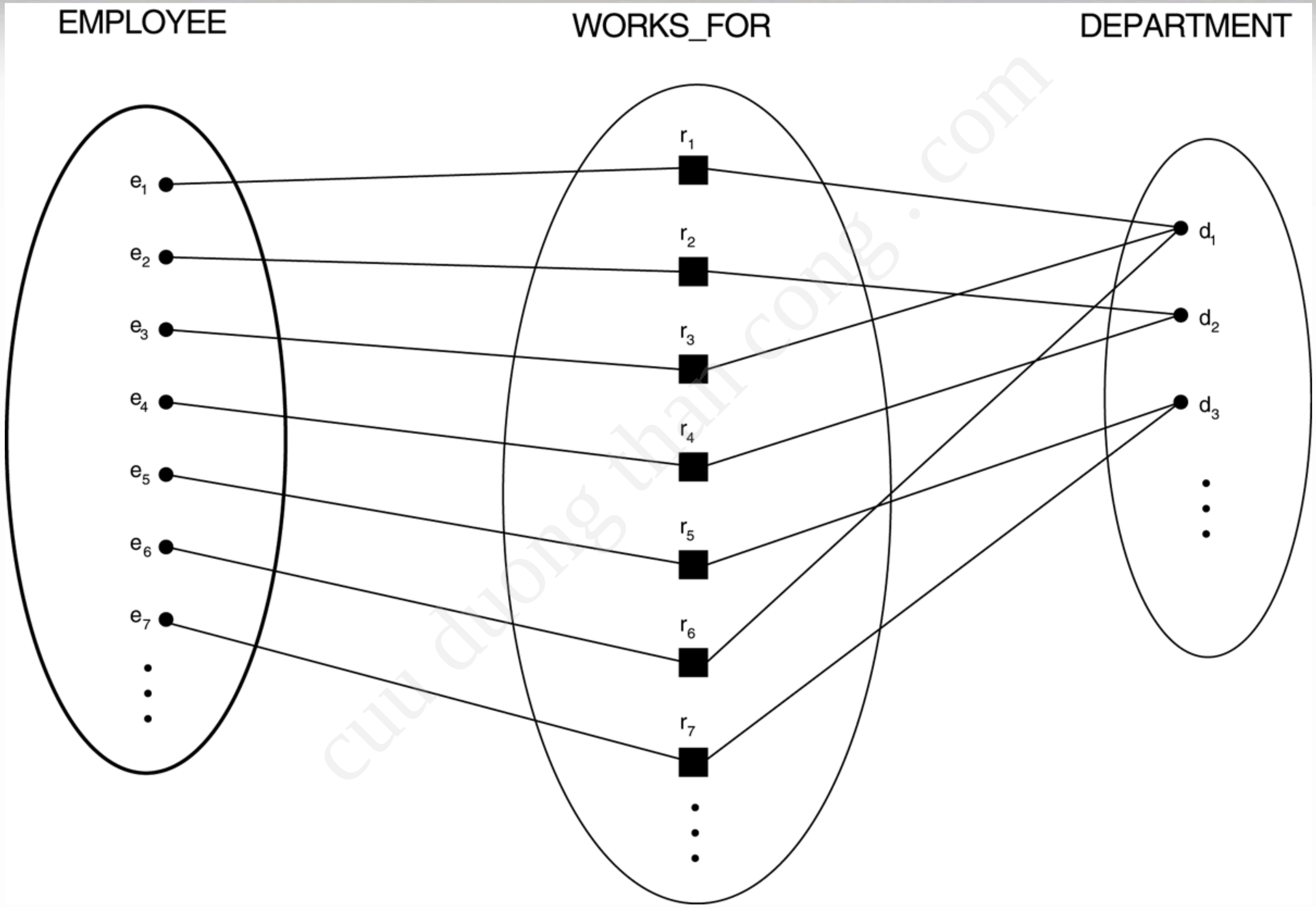
Relationships

- **Relationship type** R among n entity types E_1, E_2, ..., E_n
  - Defines a set of associations among entities from these entity types
  - Ex: the WORKS_FOR relationship type between EMPLOYEEs and DEPARTMENTs
- **Relationship instances** r_i
  - Each r_i associates n individual entities (e_1, e_2, ..., e_n). Each entity e_j in r_i is a member of entity set E_j
  - Ex: EMPLOYEE John Smith works on the ProductX PROJECT

Relationships

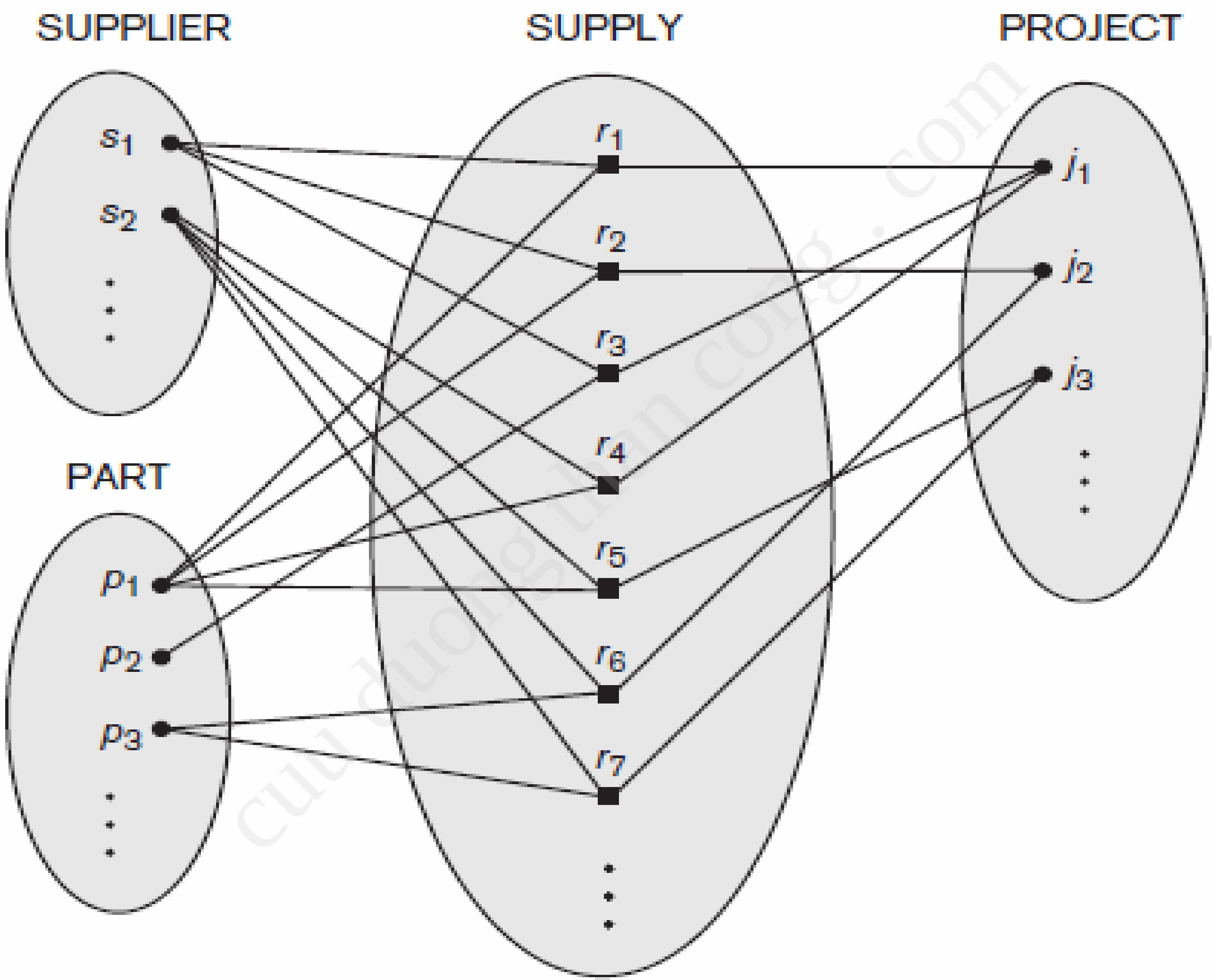
- **Degree** of a relationship type
  - Number of participating entity types
  - Binary (degree 2), ternary (degree 3), and n-ary (degree n)
- More than one relationship type can exist with the same participating entity types
  - Ex: MANAGES and WORKS_FOR are distinct relationships between EMPLOYEE and DEPARTMENT, but with different meanings and different relationship instances

Relationship Instances: Example

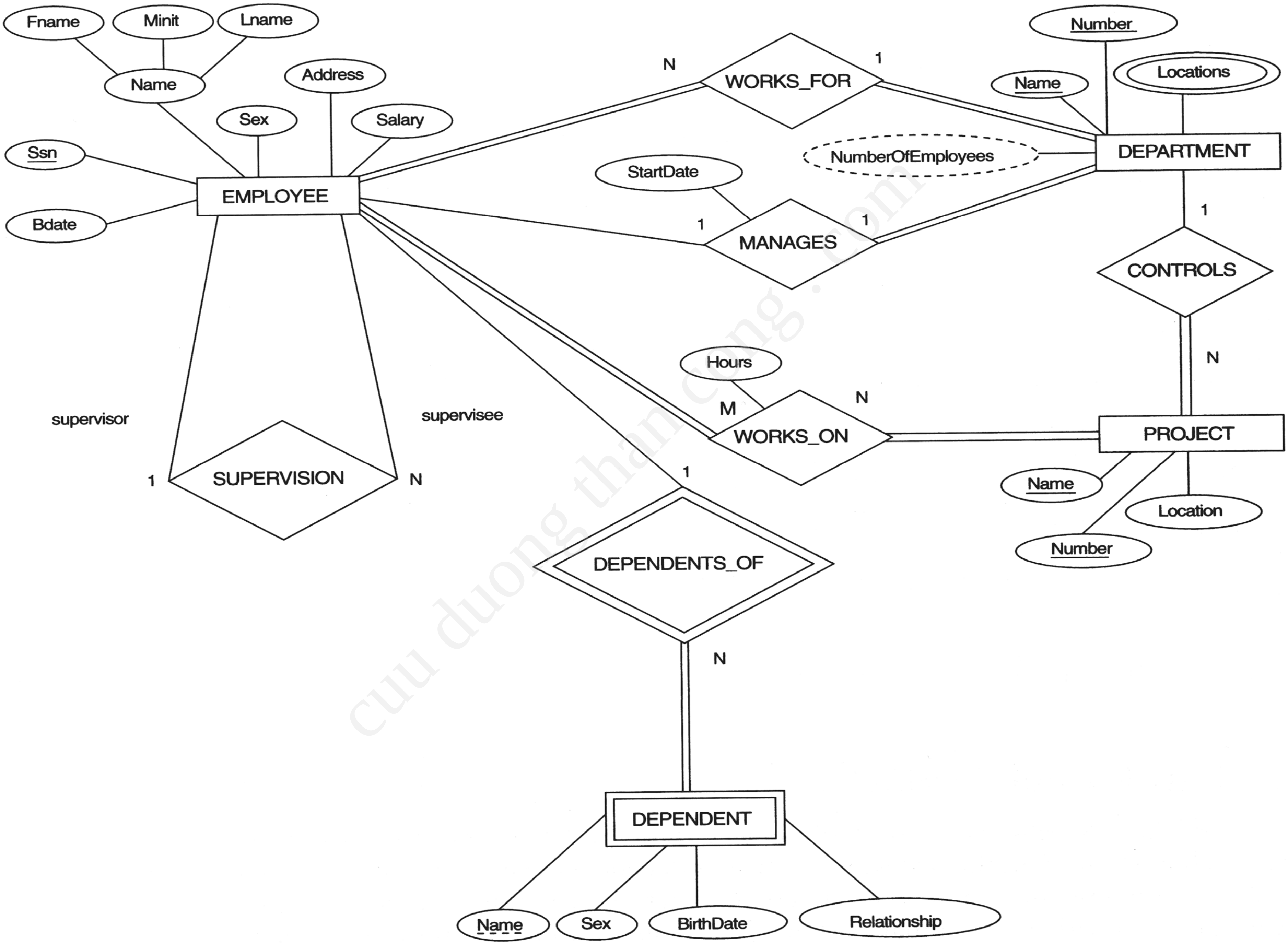


A binary relationship

Relationship Instances: Example



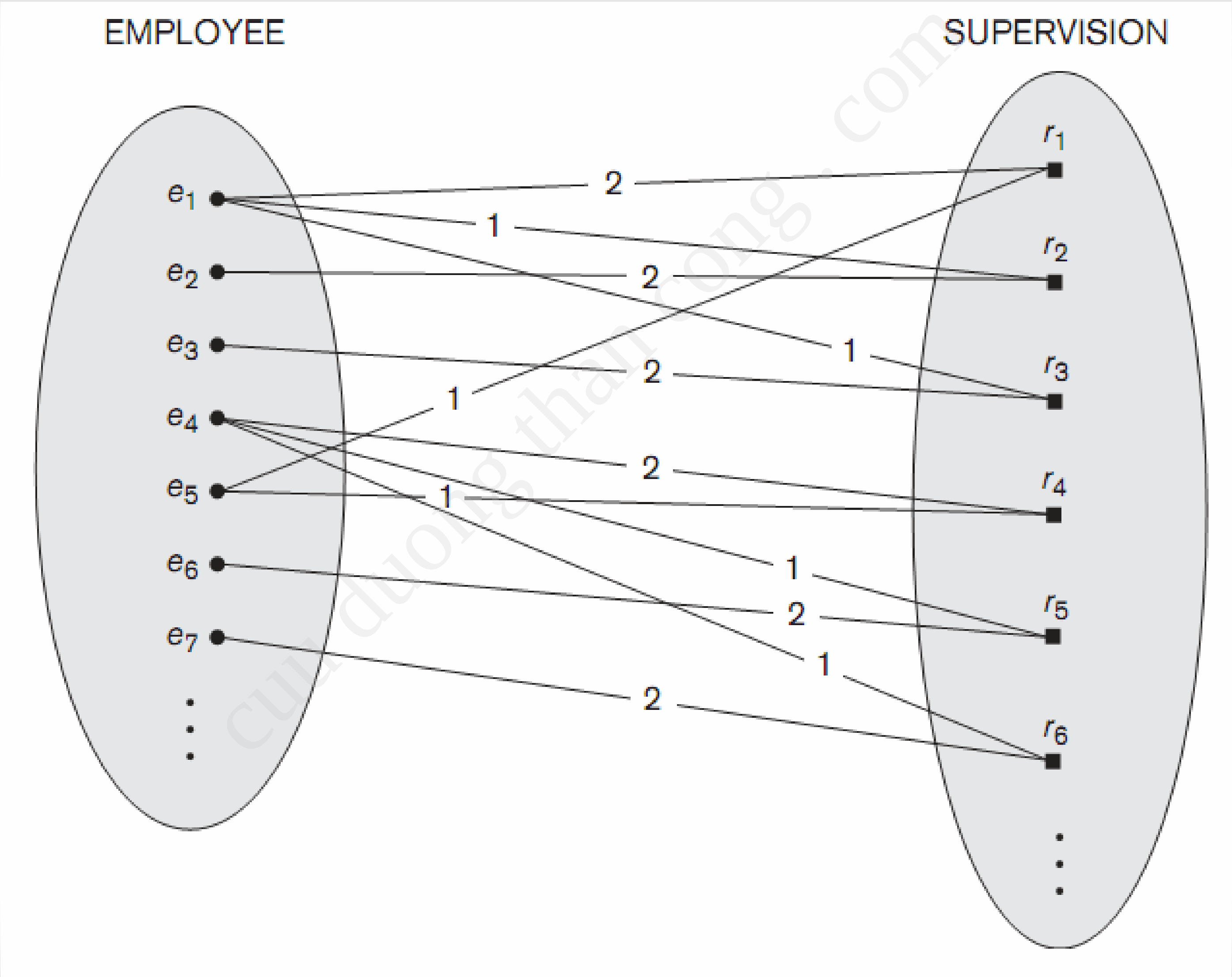
A ternary relationship


Fname
Minit
Lname
Name
Address
Sex
Salary
Ssn
Bdate
EMPLOYEE
N
WORKS_FOR
1
Number
Name
Locations
DEPARTMENT
NumberOfEmployees
StartDate
1
MANAGES
1
1
CONTROLS
N
Hours
M
WORKS_ON
N
PROJECT
Name
Location
Number
supervisor
supervisee
1
SUPERVISION
N
1
DEPENDENTS_OF
N
DEPENDENT
Name
Sex
BirthDate
Relationship

Relationships

- **Recursive** relationships
  - Same entity type participates more than once in a relationship type in different roles
  - Must specify role that a participating entity plays in each relationship instance
  - Ex: SUPERVISION relationships between EMPLOYEE (in role of supervisor or boss) and (another) EMPLOYEE (in role of subordinate or worker)

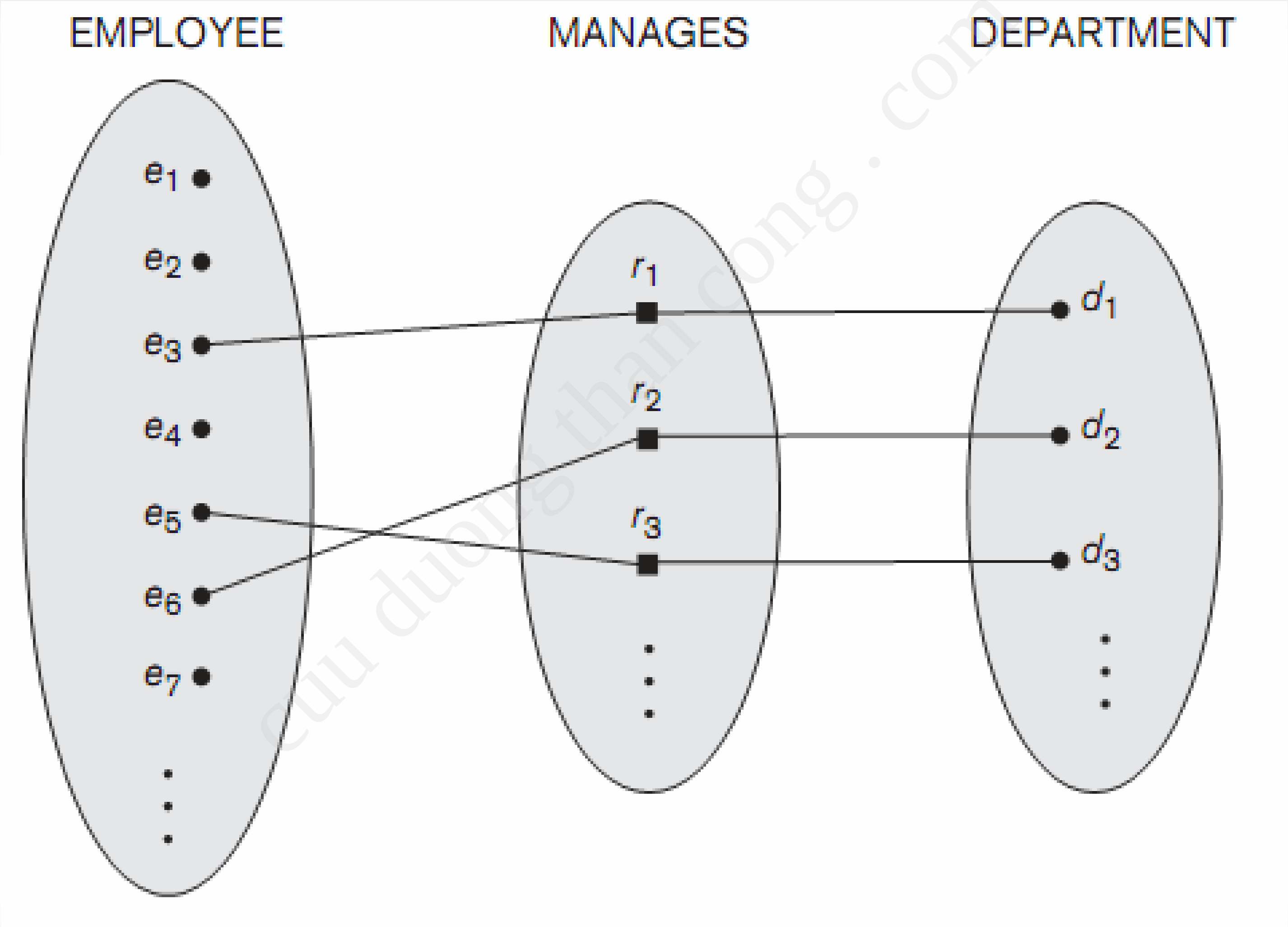
Recursive Relationship: Example



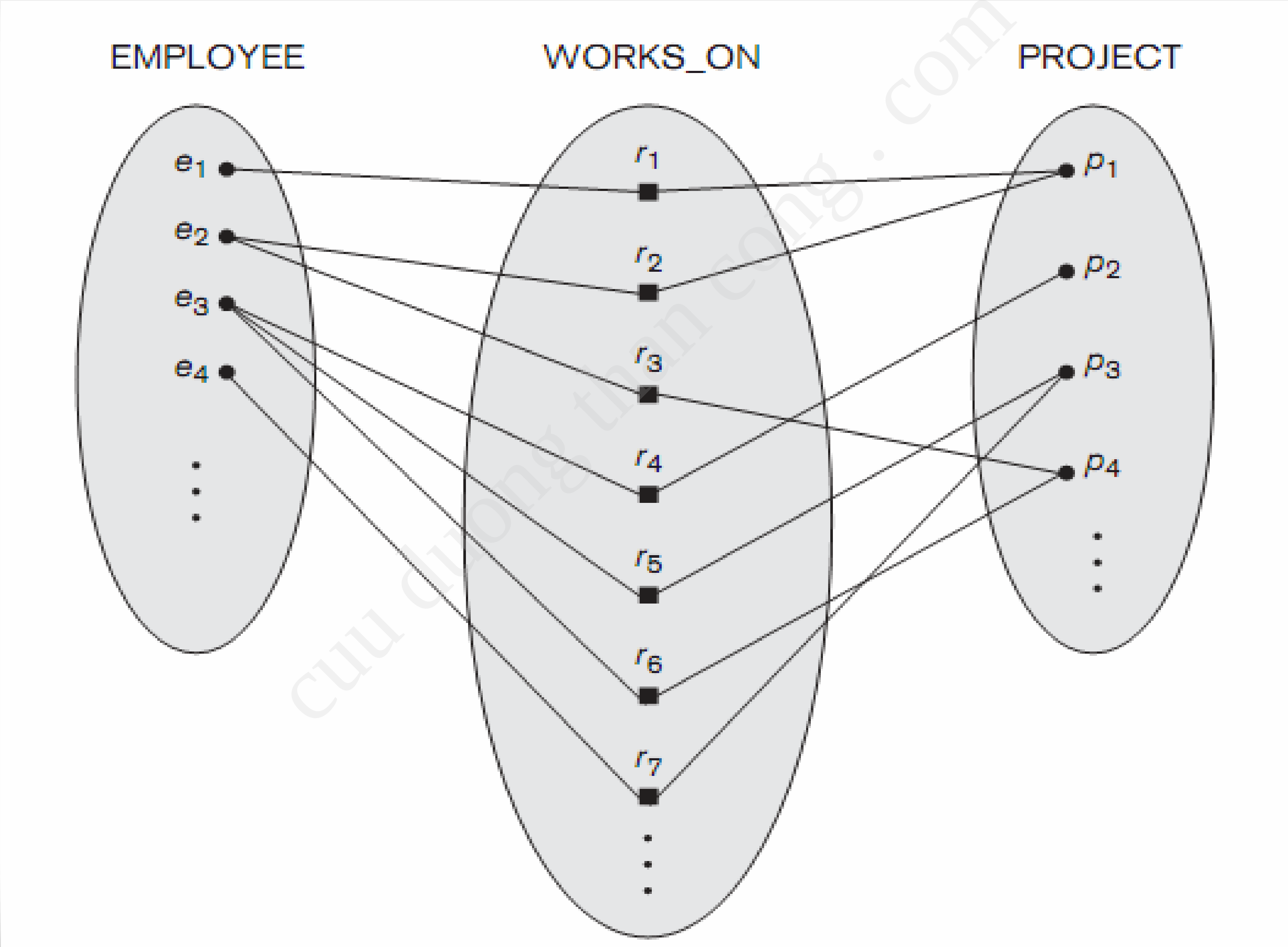
Constraints on Binary Relationship Type

- **Structural constraints**: one way to express semantics of relationship: cardinality ratio and membership class
- **Cardinality ratio**: specifies maximum number of relationship instances that entity can participate in a binary relationship.
  - one-to-one (1:1)
  - one-to-many (1:M) or many-to-one (M:1)
  - many-to-many (M:N)

One-to-one (1:1) Relationship

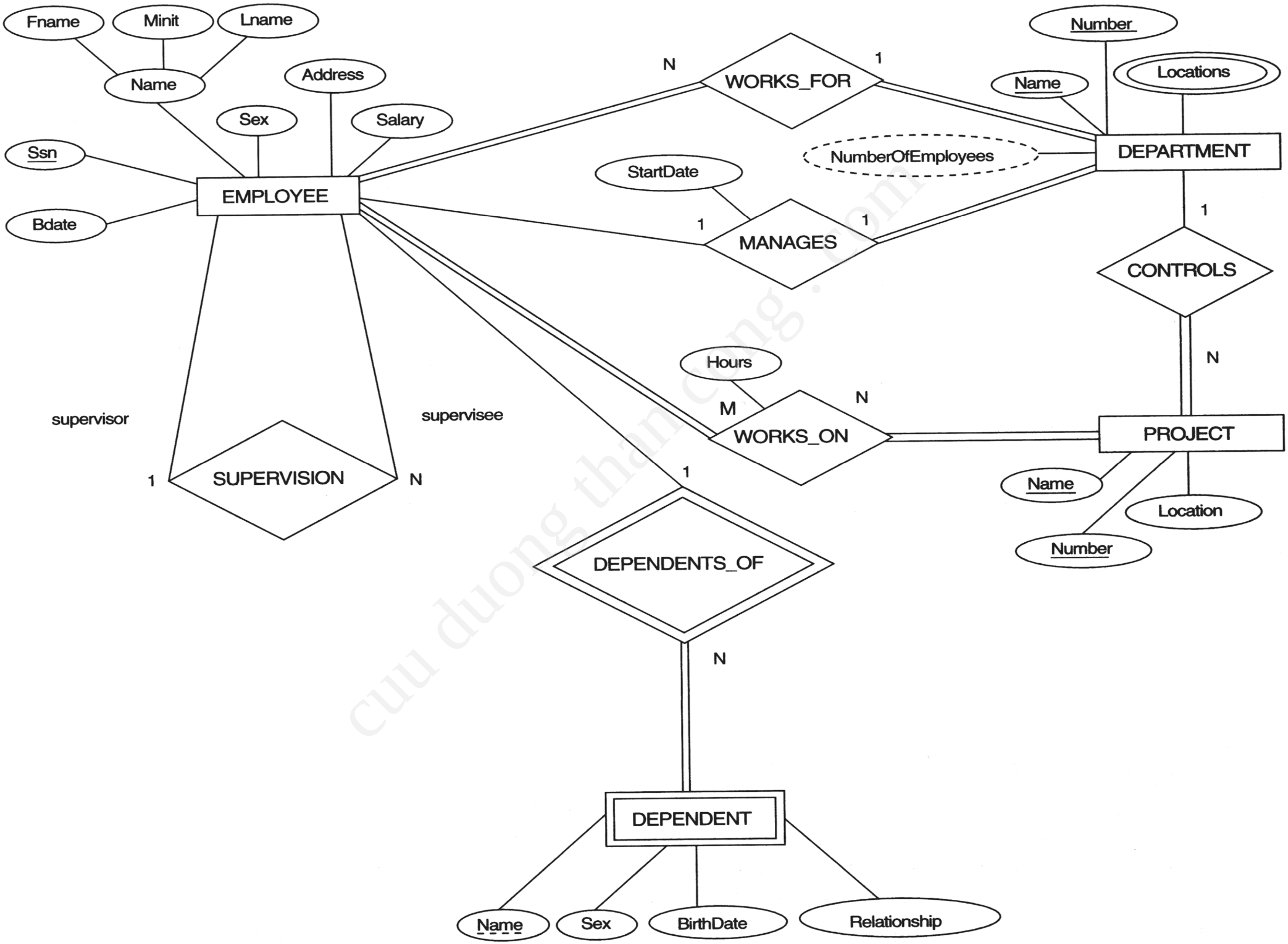


Many-to-many (M:N) Relationship



Constraints on Binary Relationship Type

- **Membership class (or participation constraint):** specifies if existence of entity depends on its being related to another entity
  - **Mandatory (total participation)** - every instance of a participating entity type must participate in the relationship. (double line)
  - **Optional (partial participation)** - not every instance of a participating entity type must participate in the relationship. (single line)

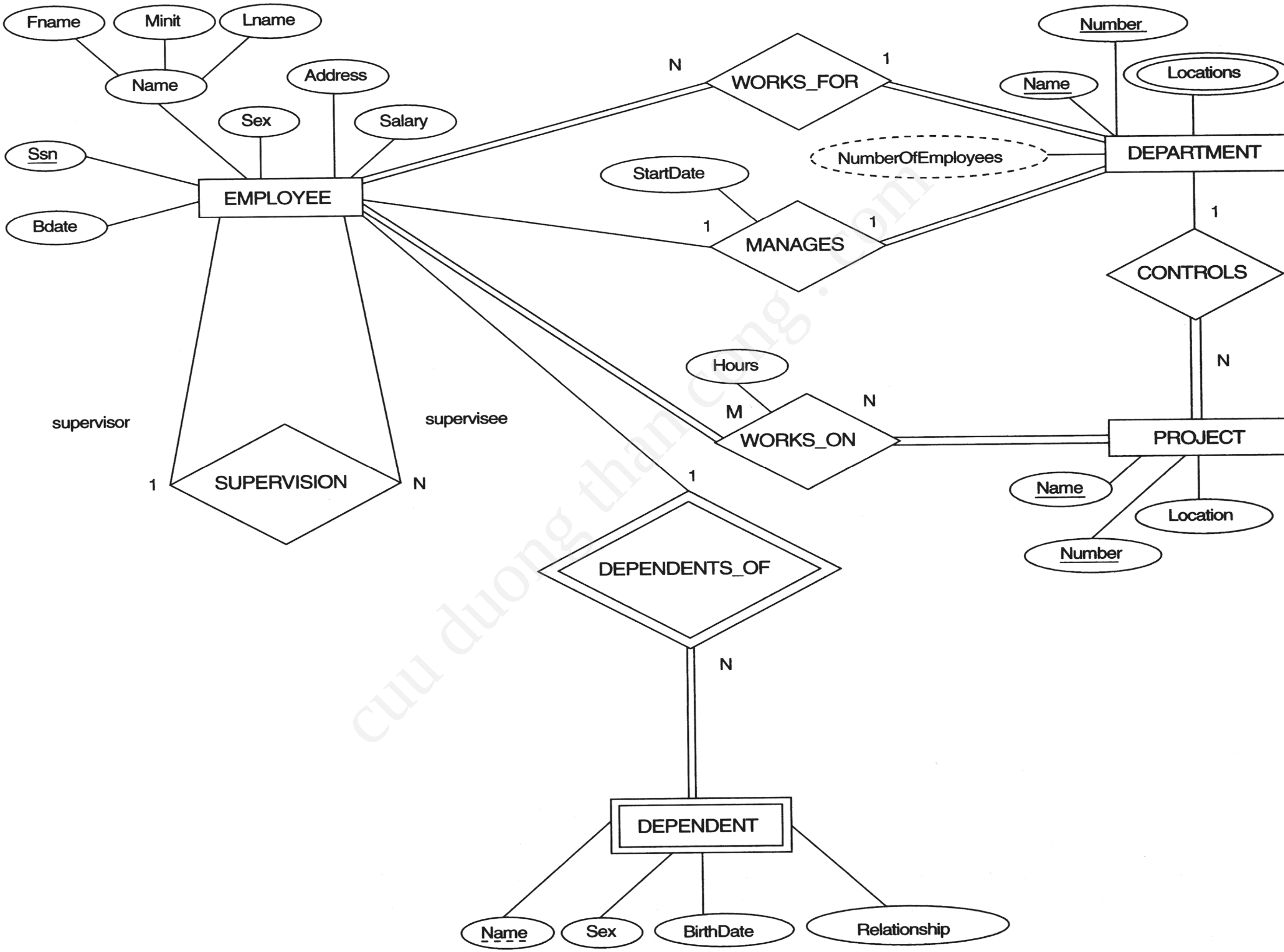

Fname
Minit
Lname
Name
Address
Sex
Salary
Ssn
EMPLOYEE
Bdate
N
WORKS_FOR
1
Number
Name
Locations
DEPARTMENT
NumberOfEmployees
StartDate
1
MANAGES
1
1
CONTROLS
N
Hours
M
N
WORKS_ON
PROJECT
Name
Location
Number
supervisor
supervisee
1
SUPERVISION
N
1
DEPENDENTS_OF
N
DEPENDENT
Name
Sex
BirthDate
Relationship

Attributes of Relationship Types

- A relationship type can have attributes.
  - Ex: HoursPerWeek of WORKS_ON
- Attributes of 1:1 or 1:N relationship types can be migrated to one entity type
  - For a 1:N relationship type: relationship attribute can be migrated only to entity type on N-side of relationship
  - For M:N relationship types: must be specified as relationship attributes

Weak Entity Types

- Do not have key attributes of their own
  - Identified by being related to specific entities from another entity type
- **Identifying relationship**
  - Relates a weak entity type to its owner
- Always has a total participation constraint
- Entities are identified by the combination of:
  - A partial key of the weak entity type
  - The particular entity they are related to in the identifying entity type



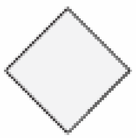
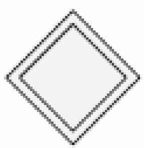
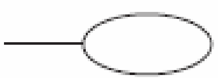
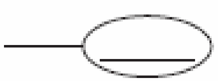
Fname
Minit
Lname
Name
Address
Sex
Salary
Ssn
EMPLOYEE
Bdate
supervisor
supervisee
1
SUPERVISION
N
N
WORKS_FOR
1
Number
Name
Locations
NumberOfEmployees
DEPARTMENT
StartDate
1
MANAGES
1
1
CONTROLS
N
Hours
M
N
WORKS_ON
PROJECT
Name
Location
Number
1
DEPENDENTS_OF
N
DEPENDENT
Name
Sex
BirthDate
Relationship

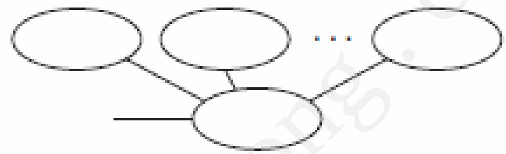
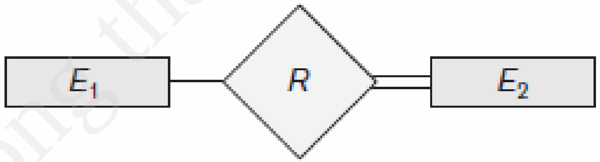
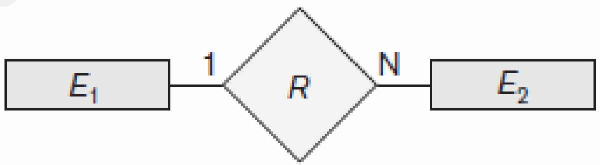
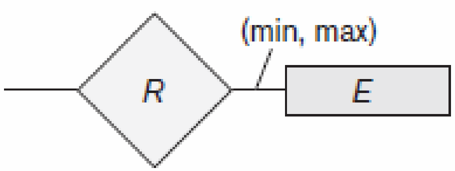
ER Diagram and Naming Conventions

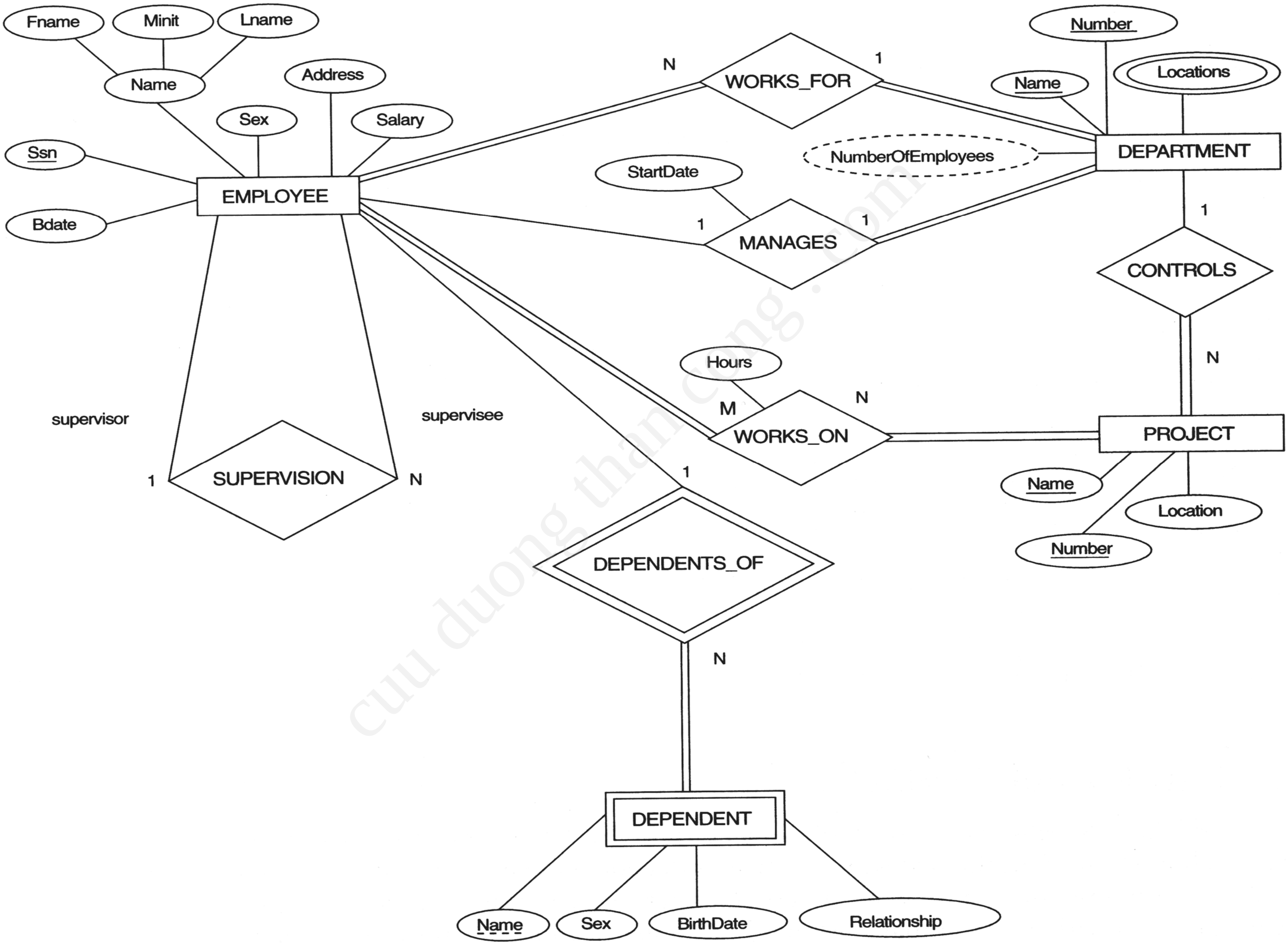
ER Diagram and Naming Conventions

- An ER model can be expressed in the form of the ER diagram.
- Proper Naming of Schema Constructs:
  - Choose names that convey meanings attached to different constructs in schema
  - Nouns give rise to entity type names
  - Verbs indicate names of relationship types
  - Choose binary relationship names to make ER diagram readable from left to right and from top to bottom

ER Diagram: Summary

Symbol	Meaning
	Entity
	Weak Entity
	Relationship
	Indentifying Relationship
	Attribute
	Key Attribute
	Multivalued Attribute

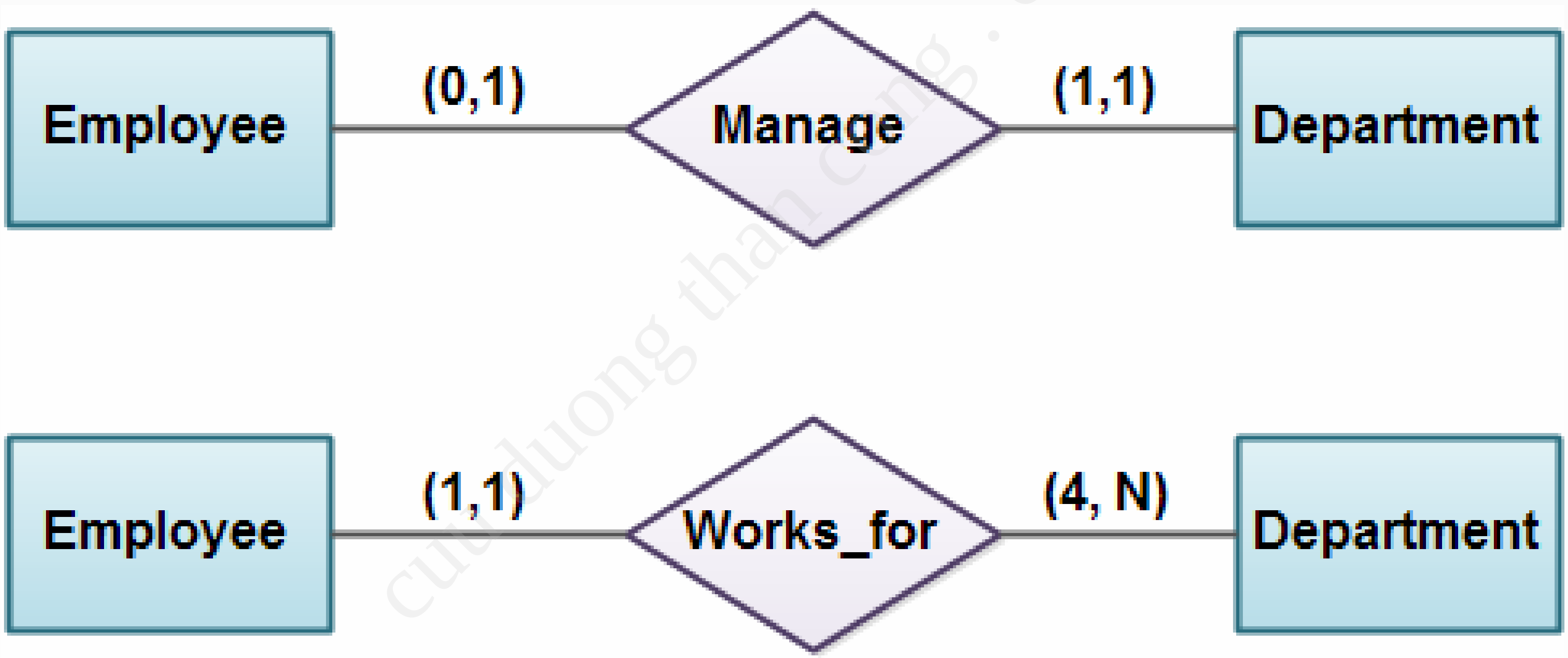
Symbol	Meaning
	Composite Attribute
	Derived Attribute
	Total Participation of E_2 in R
	Cardinality Ratio 1: N for E_1:E_2 in R
	Structural Constraint (min, max) on Participation of E in R


Fname
Minit
Lname
Name
Address
Sex
Salary
Ssn
Bdate
EMPLOYEE
N
WORKS_FOR
1
Number
Name
Locations
NumberOfEmployees
DEPARTMENT
StartDate
1
MANAGES
1
1
CONTROLS
N
Hours
M
WORKS_ON
N
PROJECT
Name
Location
Number
supervisor
supervisee
1
SUPERVISION
N
1
DEPENDENTS_OF
N
DEPENDENT
Name
Sex
BirthDate
Relationship

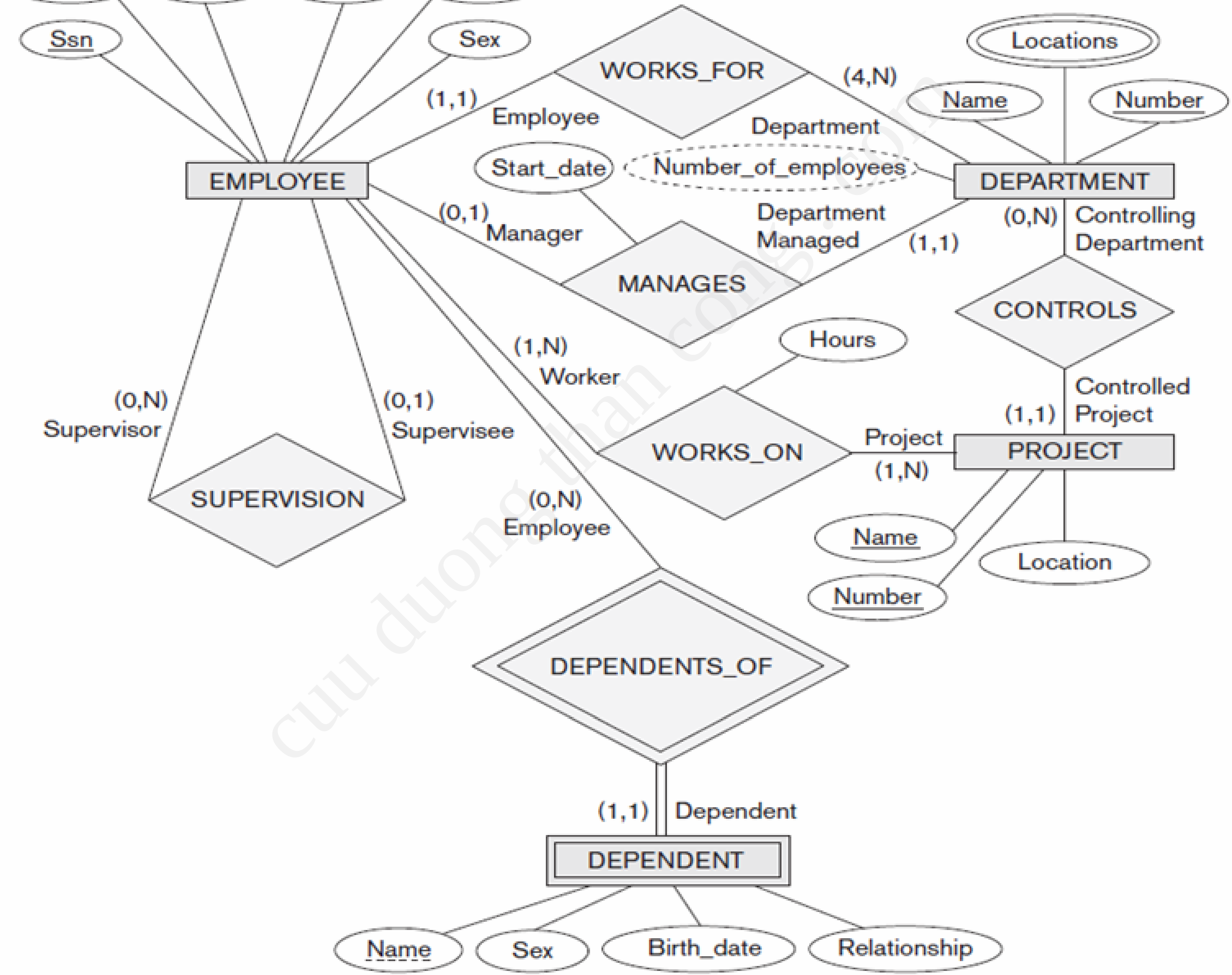
Alternative Diagrammatic Notations

- **(Min-max) notation for relationships**
  - Specify structural constraints on relationships
  - Replaces cardinality ratio (1:1, 1:N, M:N) and single/double line notation for participation constraints
  - Associate a pair of integer numbers (min, max) with each participation of an entity type *E* in a relationship type *R*, where $0 \le min \le max$ and $max \ge 1$

(min, max) Notation



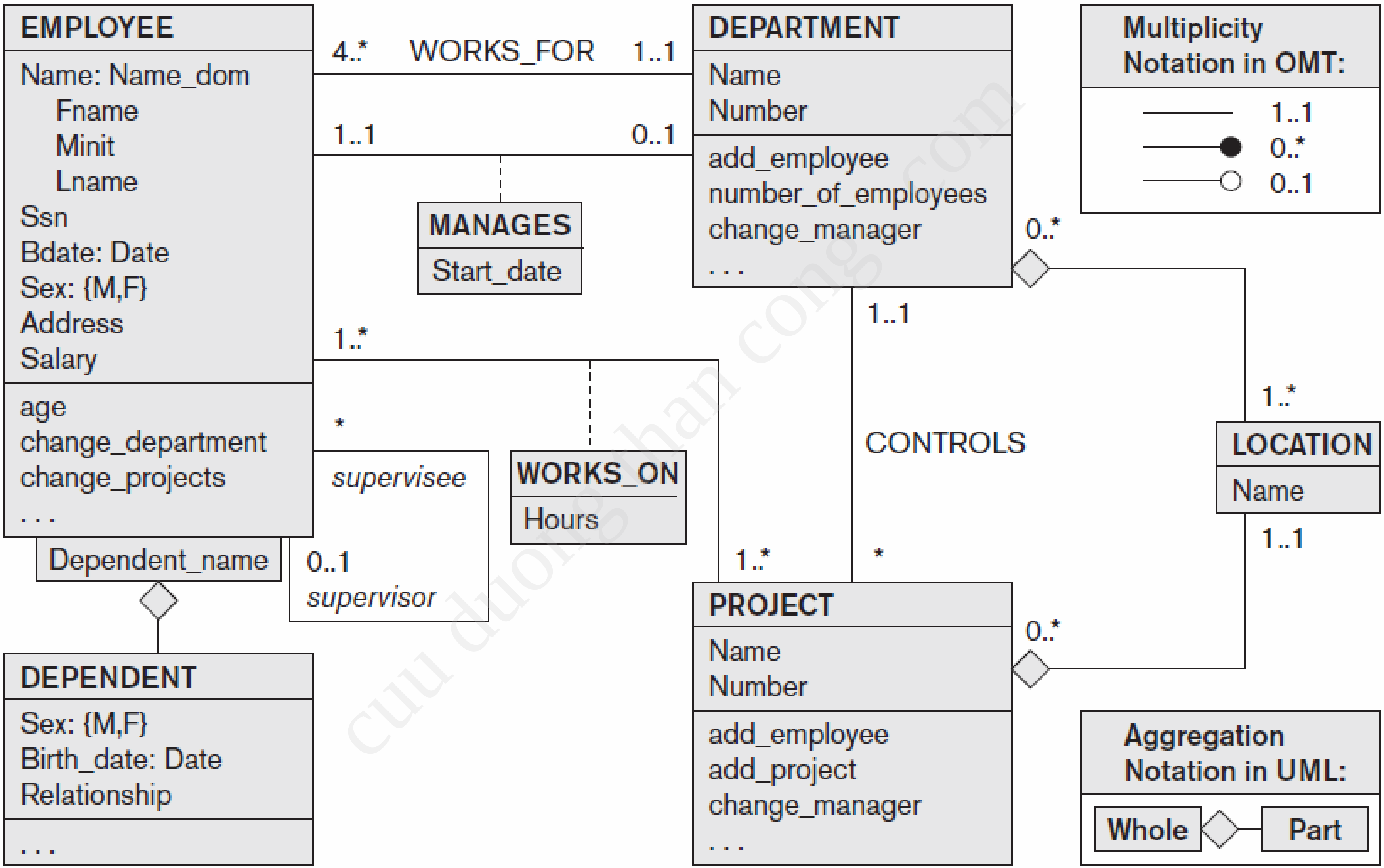
ER diagrams for the company schema, with structural constraints specified using (min, max) notation and role names.



Alternative Diagrammatic Notations

- UML methodology
  - Used extensively in software design
  - Many types of diagrams for various software design purposes
- **UML class diagrams**
  - Entity in ER corresponds to an object in UML

The COMPANY conceptual schema
in UML class diagram notation.



Alternative Diagrammatic Notations

- **UML class diagrams**
  - **Class** includes three sections:
    - Top section gives the class name
    - Middle section includes the attributes;
    - Last section includes operations that can be applied to individual objects
  - **Associations**: relationship types
  - **Relationship instances**: links

Alternative Diagrammatic Notations

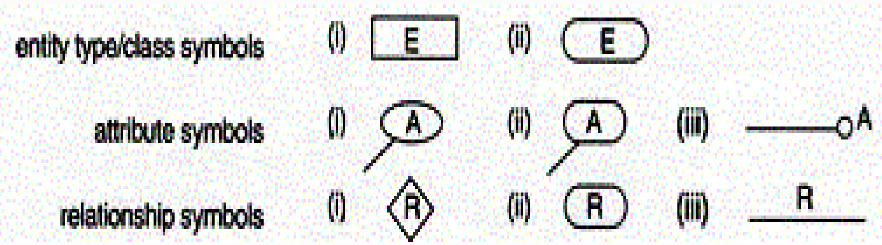
- **UML class diagrams**
  - Binary association
    - Represented as a line connecting participating classes
    - May optionally have a name
  - Link attribute
    - Placed in a box connected to the association's line by a dashed line

Alternative Diagrammatic Notations

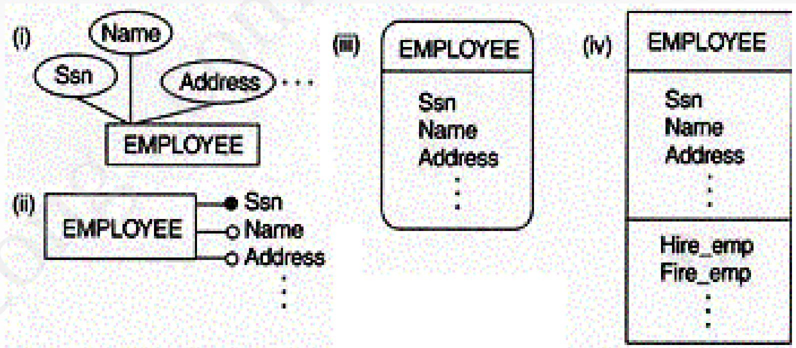
- **UML class diagrams**
  - **Multiplicities**: min..max, asterisk (*) indicates no maximum limit on participation
  - Types of relationships: **association** and **aggregation**
  - Distinguish between **unidirectional** and **bidirectional** associations
  - Model weak entities using **qualified association**

Alternative Diagrammatic Notations

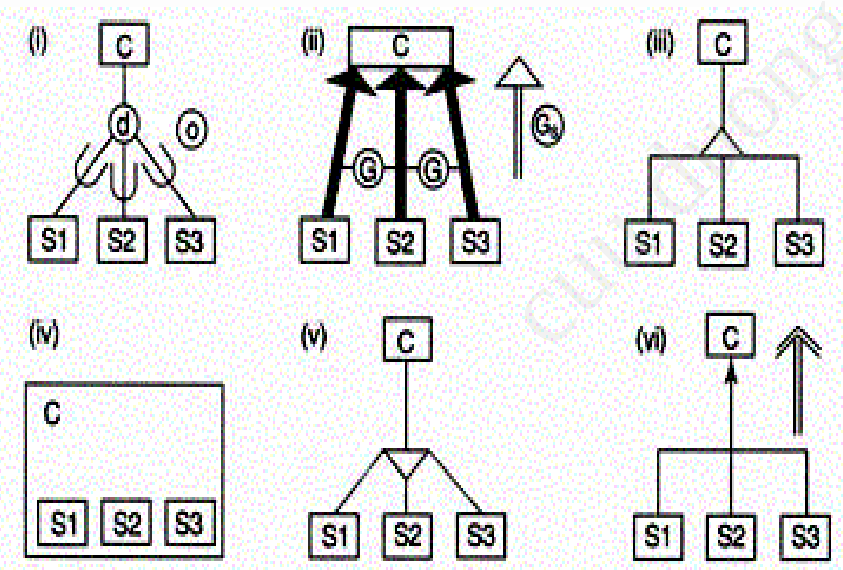
Symbols for entity type / class, attribute and relationship



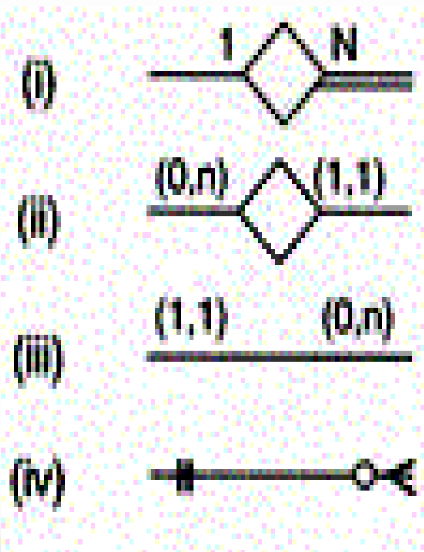
Displaying attributes



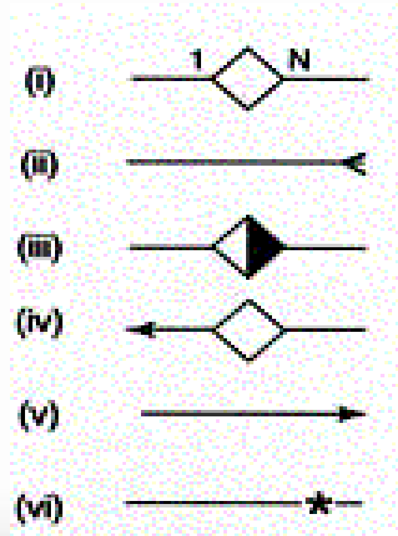
Notations for displaying specialization / generalization



Various (min, max) notations



Displaying cardinality ratios

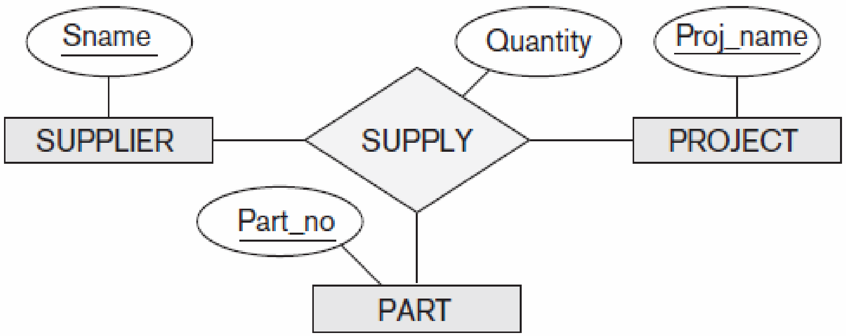


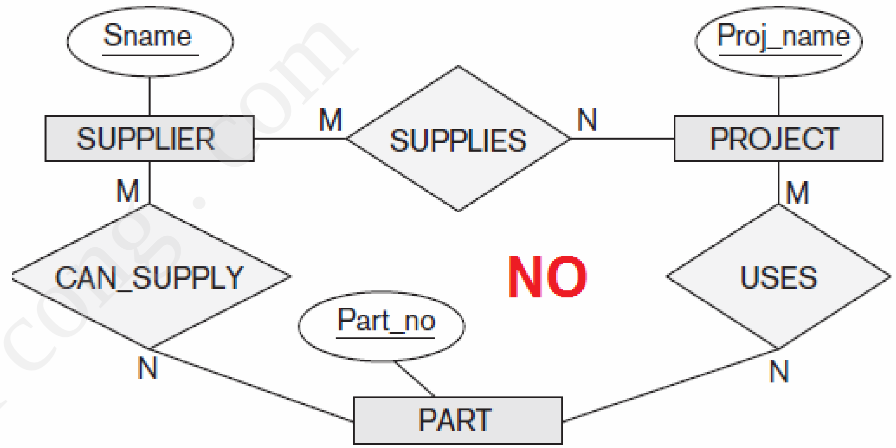
Higher Degree Relationship Types

Binary vs. Ternary Relationships

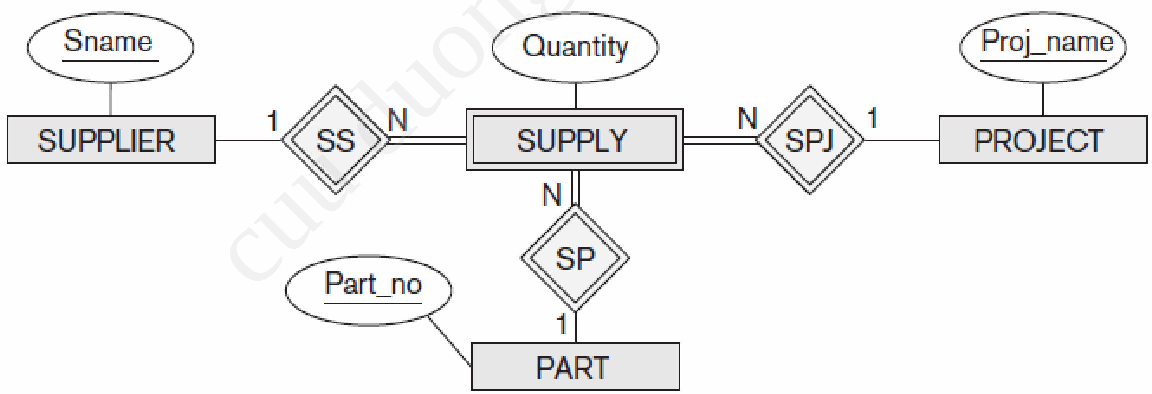
- Some database design tools permit only binary relationships
  - Ternary relationship must be represented as a weak entity type
  - No partial key and three identifying relationships
- Represent ternary relationship as a regular entity type
  - By introducing an artificial or surrogate key

Binary vs. Ternary Relationships





YES



Constraints on Higher-Degree Relationships

- Notations for specifying structural constraints on *n*-ary relationships
  - Should both be used if it is important to fully specify structural constraints

ER Models: Problems

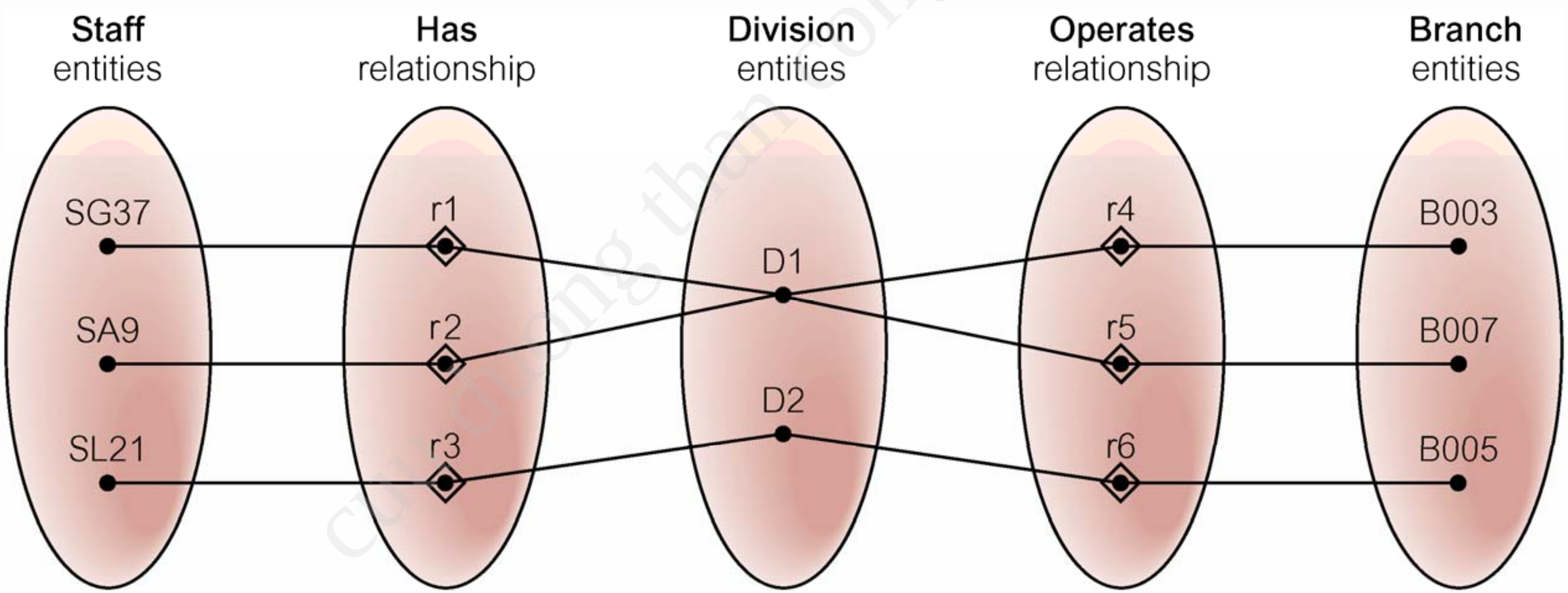
Problems with ER Models

- Problems may arise when designing a conceptual data model called **connection traps**
- Often due to a misinterpretation of the meaning of certain relationships
- Two main types of connection traps are called fan traps and chasm traps

Problems with ER Models

- Fan Trap
  - Where a model represents a relationship between entity types, but pathway between certain entity occurrences is ambiguous
  - Usually: two or more 1:N relationships fan out from the same entity
- Chasm Trap
  - Where a model suggests the existence of a relationship between entity types, but pathway does not exist between certain entity occurrences
  - Usually: optional participation

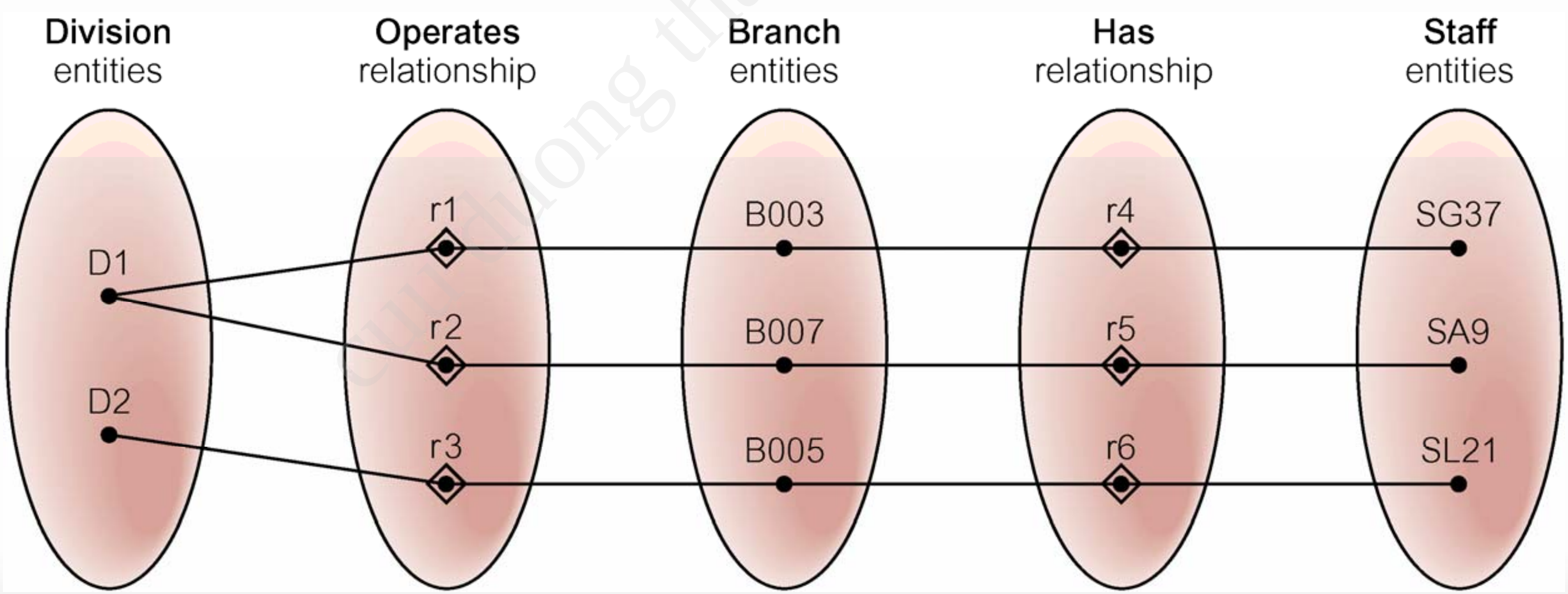
An Example of a Fan Trap



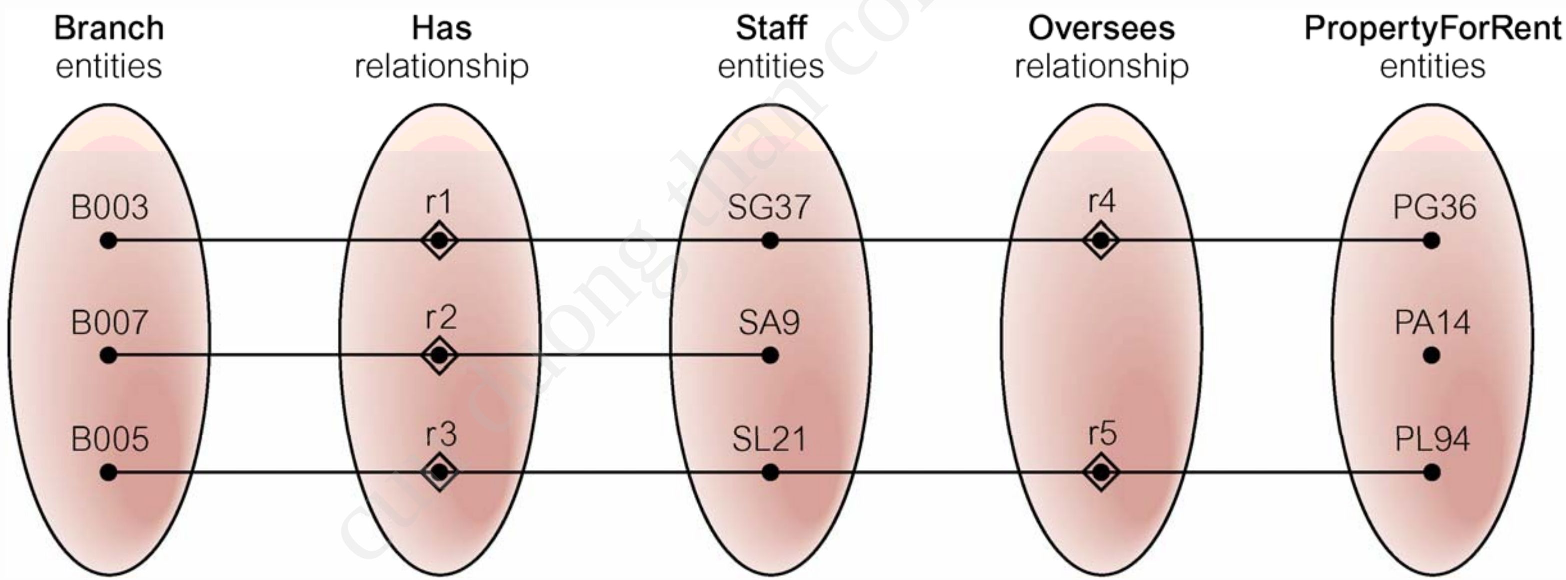
At which branch office does staff number SG37 work?

Restructuring ER Model to Remove Fan Trap

- SG37 works at branch B003

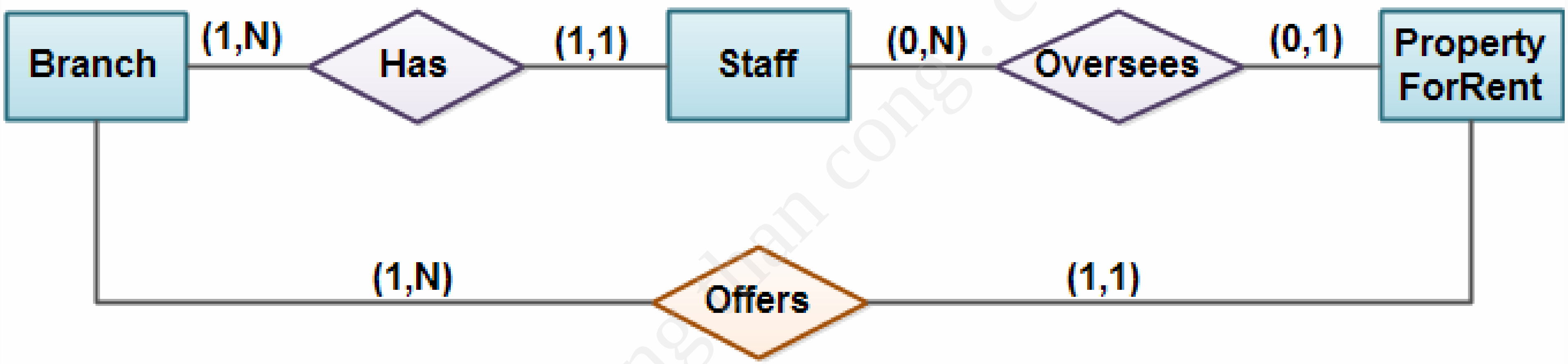


An Example of a Chasm Trap



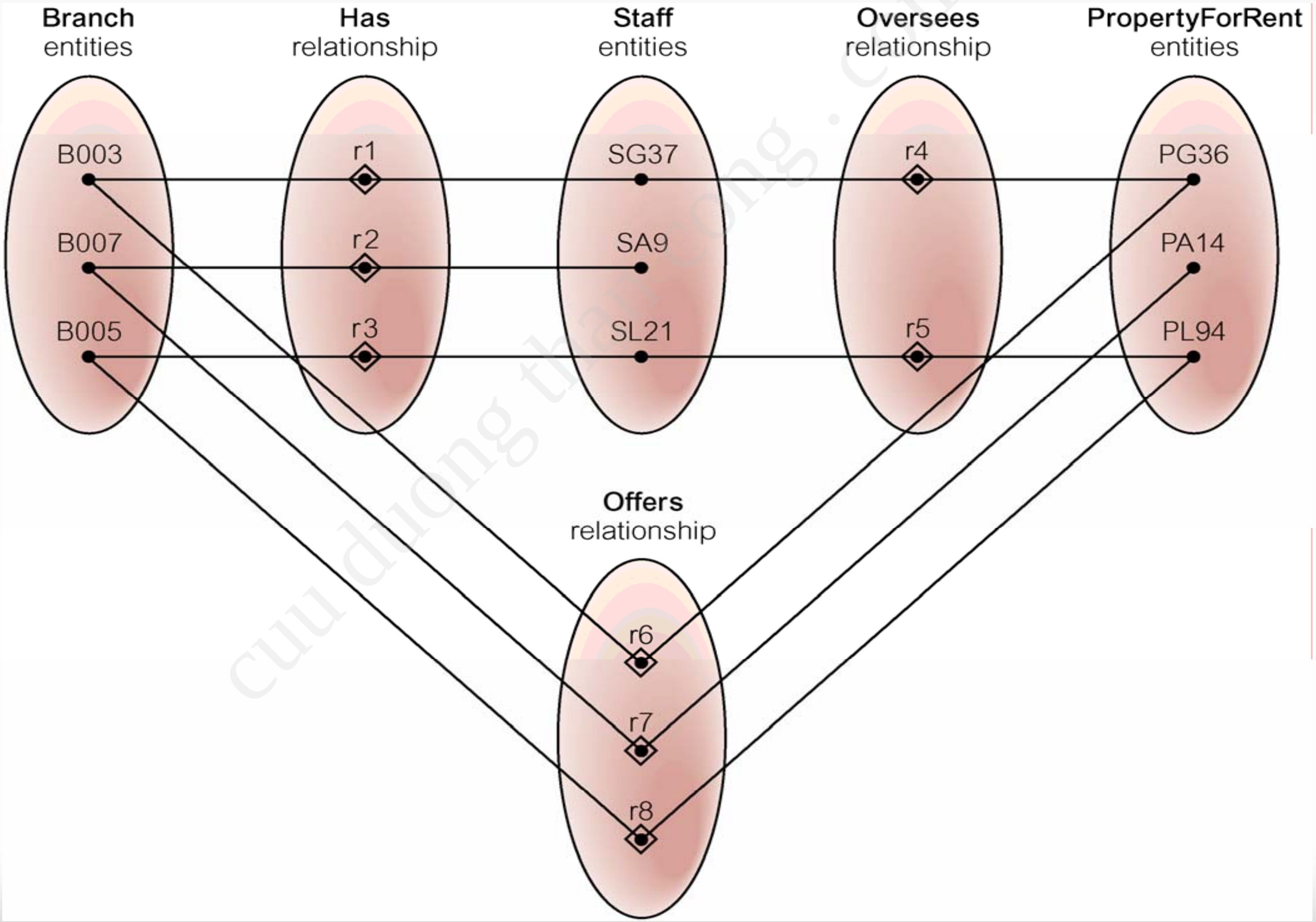
At which branch office is property PA14 available?

ER Model Restructured to Remove Chasm Trap



- Adding the ***Offers*** relationship resolves the chasm trap

ER Model Restructured to Remove Chasm Trap



Summary

1	Overview of Database Design Process
2	What is ER Model? And Why?
3	A Sample Database Application
4	ER Model Concepts
5	ER Diagram and Naming Conventions
6	Alternative Diagrammatic Notations
7	Relationship Types of Degree Higher than Two
8	Problems with ER Models