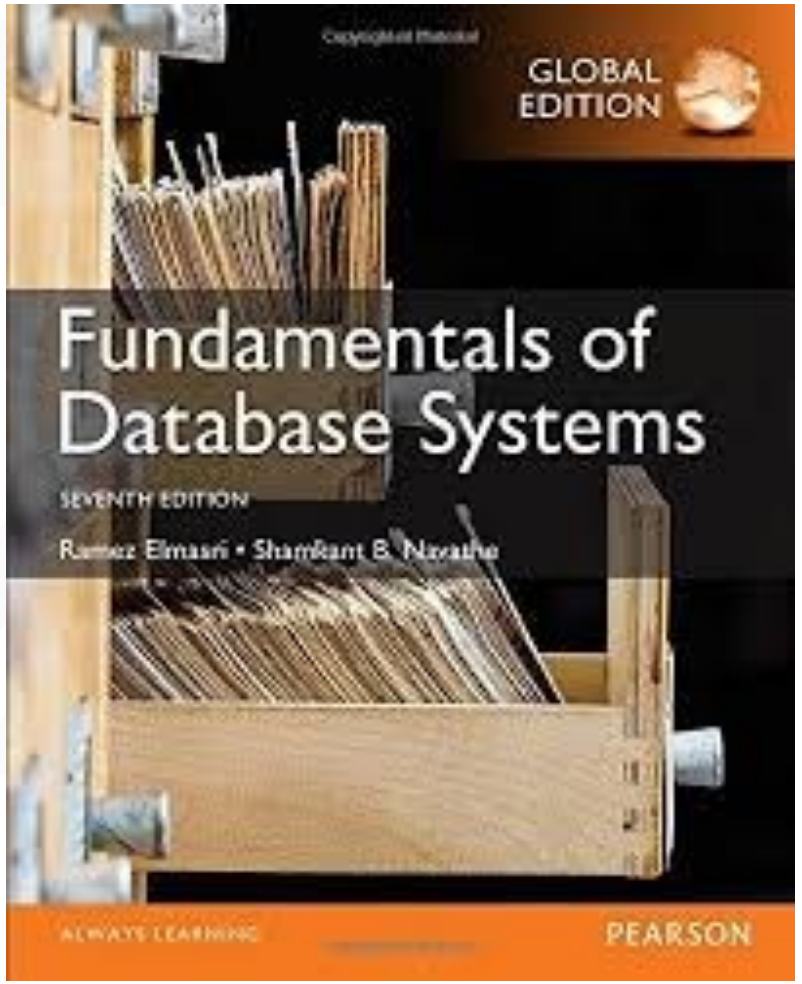
Database Systems

Program in Computer Engineering
Faculty of Engineering

King Mongkut's Institute of Technology Ladkrabang

Text

- Ramez Elmasri and Shamkant B. Navathe. "Fundamentals of Database Systems" 7th Edition., Pearson, 2017



Chapter 3

Data Modeling Using the Entity-Relationship (ER) Model

Data Models (from Ch 2)

- A set of concepts to describe the ***structure*** of a database, the ***operations*** for manipulating these structures, and certain ***constraints*** that the database should obey.

Overview of Database Design Process

- Two main activities:
  - Database design
  - Applications design
- Focus in this chapter on conceptual database design
  - To design the conceptual schema for a database application
- Applications design focuses on the programs and interfaces that access the database
  - Generally considered part of software engineering

- User defined **operations** (or **transactions**) including **retrievals** and **updates**.
- Common tools
  - Data flow diagram
  - Sequence diagram
  - Scenarios etc.

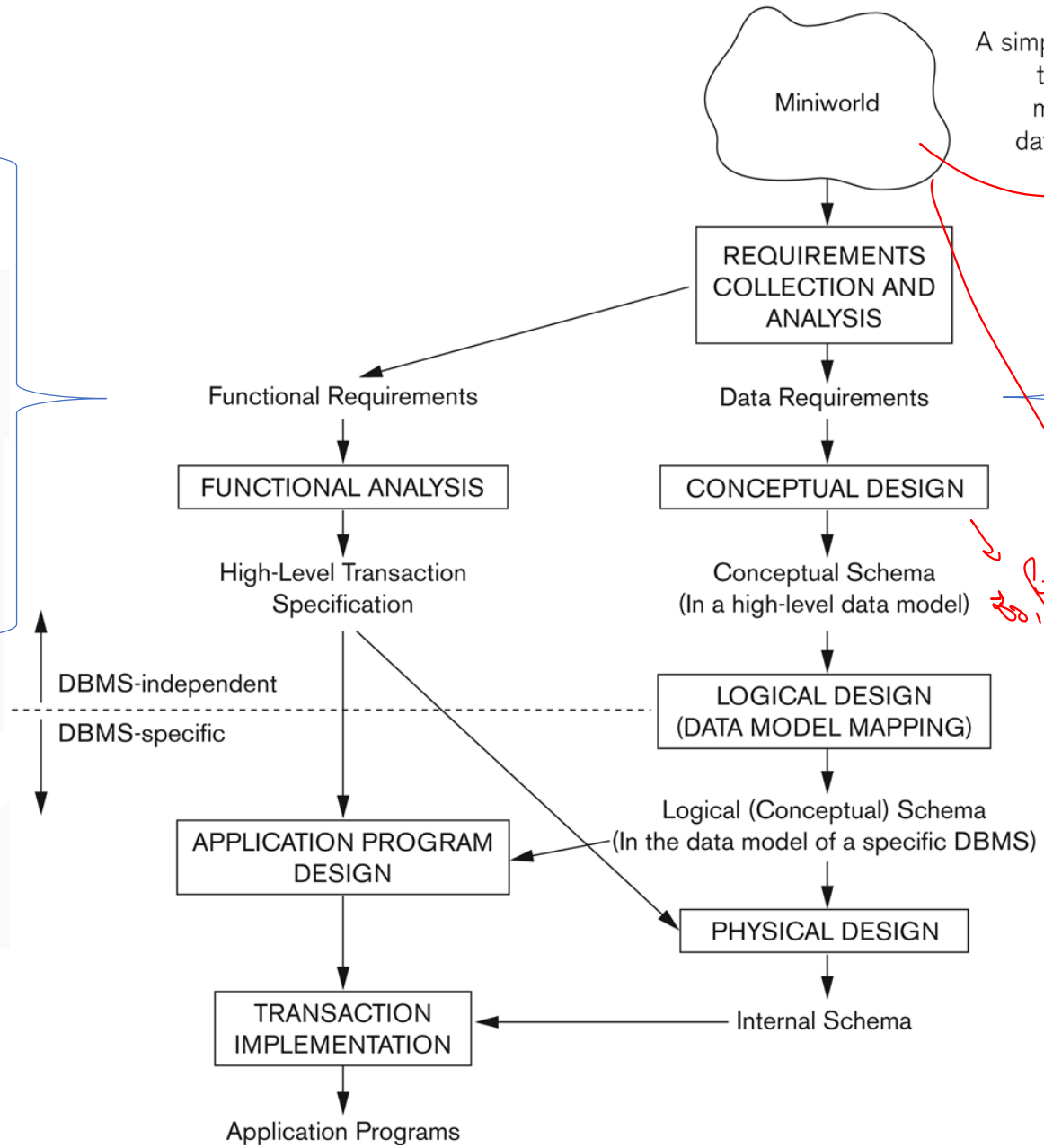


Figure 3.1
A simplified diagram to illustrate the main phases of database design.

- Detailed description
  - Entity types
  - Relationships
  - Constraints

Example COMPANY Database

- Create a database schema design based on the following (simplified) **requirements** of the COMPANY Database:
  - The company is organized into DEPARTMENTs.
    Each department has a name, number and an employee who *manages* the department.
    We keep track of the start date of the department manager.
    A department may have several locations.
  - Each department *controls* a number of PROJECTs.
    Each project has a unique name, unique number and is located at a single location.

- The database will store each EMPLOYEE's social security number, address, salary, sex, and birthdate.
  - Each employee *works for* one department but may *work on* several projects.
  - The DB will keep track of the number of hours per week that an employee currently works on each project.
  - It is required to keep track of the *direct supervisor* of each employee

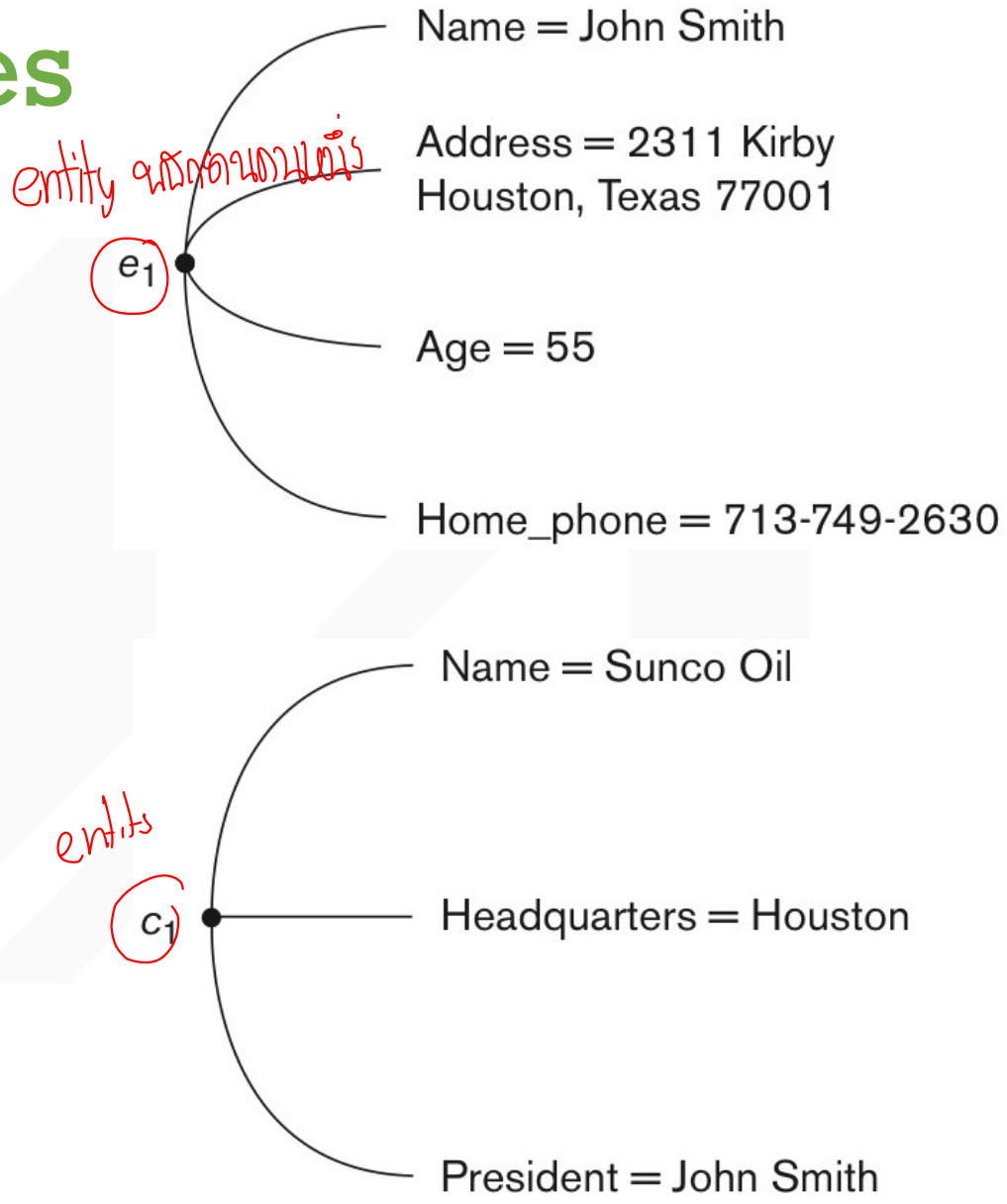
- Each employee may *have* a number of DEPENDENTs.
  - For each dependent, the DB keeps a record of name, sex, birthdate, and relationship to the employee

Entities and Attributes

- **Entity** is a basic concept for the ER model.
  Entities are specific things or objects in the mini-world that are represented in the database.
  - For example the EMPLOYEE John Smith, the Research DEPARTMENT, the ProductX PROJECT
- **Attributes** are properties used to describe an entity.
  - For example an EMPLOYEE entity may have the attributes Name, SSN, Address, Sex, BirthDate

Entities and Attributes

- A specific entity will have a value for each of its attributes.
  - For example a specific employee entity may have Name='John Smith', SSN='123456789', Address ='731, Fondren, Houston, TX', Sex='M', BirthDate='09-JAN-55'
- Each attribute has a *value set* (or data type) associated with it – e.g. integer, string, date, enumerated type, …



Types of Attributes

- **Simple**
  - Each entity has a single atomic value for the attribute.
  - For example, SSN or Sex.
- **Composite** → ใช้ใน entity → เป็นการรวมของ attributes → หลายๆ → ชื่อนามสกุล, ถนน, ซอย
  - The attribute may be composed of several components.
  - For example:
    - Address(Apt#, House#, Street, City, State, ZipCode, Country), or
    - Name(FirstName, MiddleName, LastName).
    - Composition may form a hierarchy where some components are themselves composite.
- **Multi-valued**
  - An entity may have multiple values for that attribute.
  - For example,
    - Color of a CAR or
    - PreviousDegrees of a STUDENT.
  - Denoted as {Color} or {PreviousDegrees}.

- In general, composite and multi-valued attributes may be nested arbitrarily to any number of levels, although this is rare.
  - For example,
    - PreviousDegrees of a STUDENT is a composite multi-valued attribute denoted by {PreviousDegrees (College, Year, Degree, Field)}
  - Multiple PreviousDegrees values can exist
  - Each has four subcomponent attributes:
    - College, Year, Degree, Field

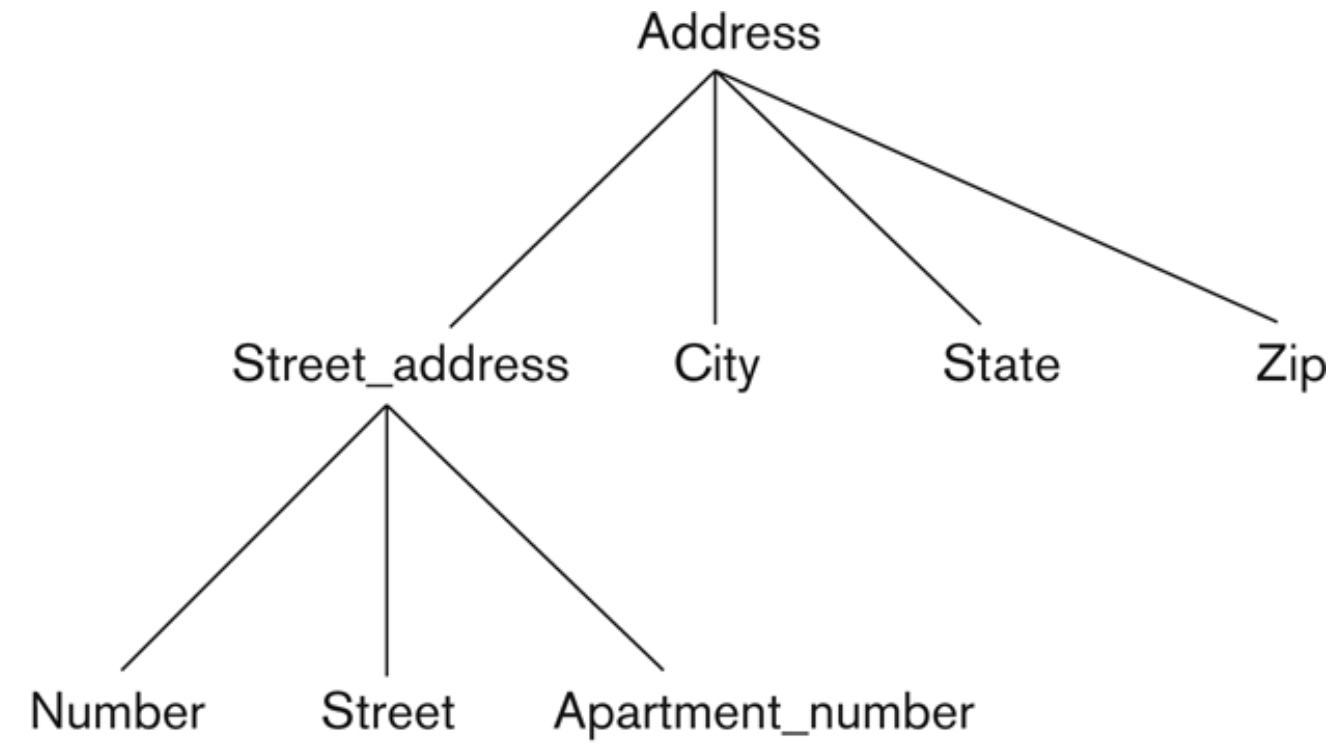


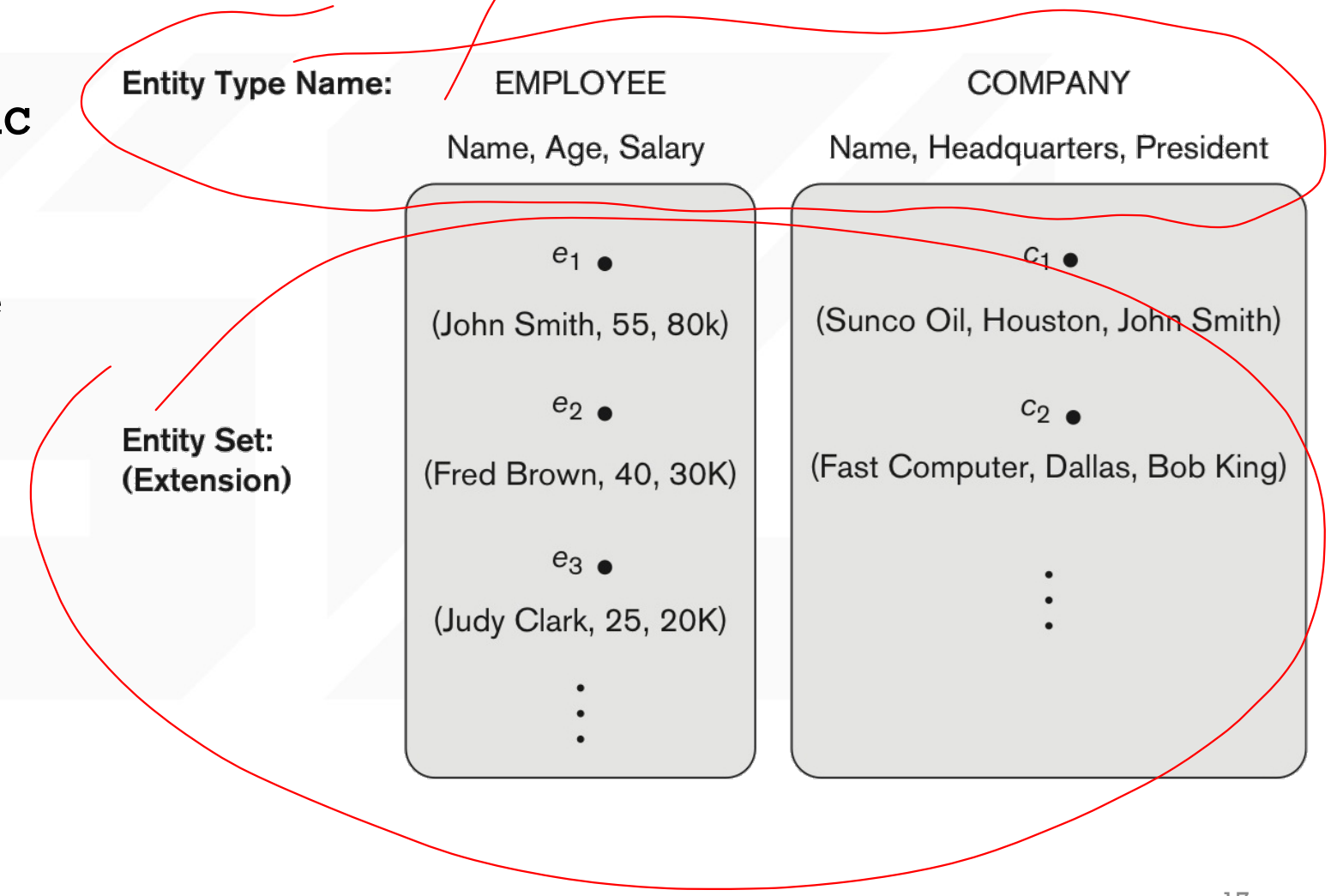
Figure 3.4
A hierarchy of composite attributes.

- **Stored vs. Derived Attributed**
  - Two or more attribute values are related
    - E.g., Birth_date vs. Age
      - The Age attribute is derived from Birth_date.
      - Age is called a derived attribute and is derivable from the Birth_date.
  - Some attribute values can be derived from related entities
    - E.g., Number_of_employees of a DEPARTMENT entity
      - Can be derived by counting the number of employee related to (working for) that department.

- **NULL value**
  - Not applicable
  - Unknown
    - The attribute value is missing or
    - Not known

Entity Types

- Entities with the same basic attributes are grouped or typed into an entity type.
  - For example, the entity type EMPLOYEE and PROJECT.



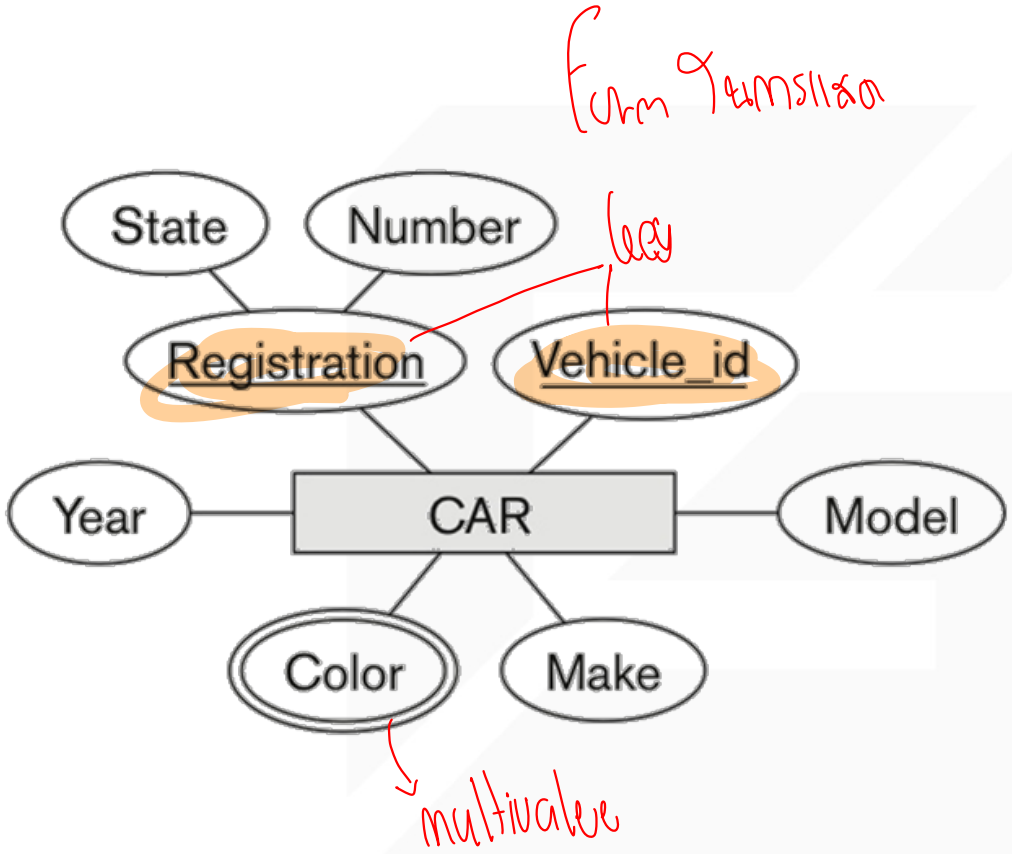
Entity Sets

- Each entity type will have a collection of entities stored in the database
  - Called the **entity set** or sometimes **entity collection**
- Same name used to refer to both the **entity type** and the **entity set**
- However, entity type and entity set may be given different names
- Entity set is the current ***state*** of the entities of that type that are stored in the database

Key Attribute → คุณสมบัติของเอนทิตี → คีย์

- An attribute of an entity type for which each entity must have a unique value is called a key attribute of the entity type.
  - For example, SSN of EMPLOYEE.
- A key attribute may be composite.
  - VehicleTagNumber is a key of the CAR entity type with components (Number, State).

- An entity type may have more than one key.
  - The CAR entity type may have two keys:
    - VehicleIdentificationNumber (popularly called VIN)
    - VehicleTagNumber (Number, State), aka license plate number
- Each key is **underlined**
  - Note: this is different from the relational schema where only one "primary key" is underlined.



The CAR entity type with two key attributes; Registration and Vehicle_id.

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})

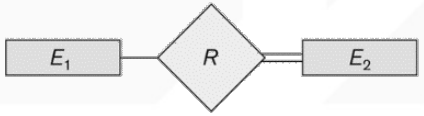
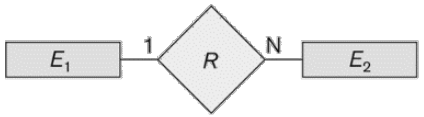
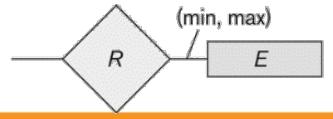
CAR_3
((VSY 720, TEXAS), TD729, Chrysler LeBaron, 4-door, 2002, {white, blue})

Entity set with three entities

Displaying an Entity type

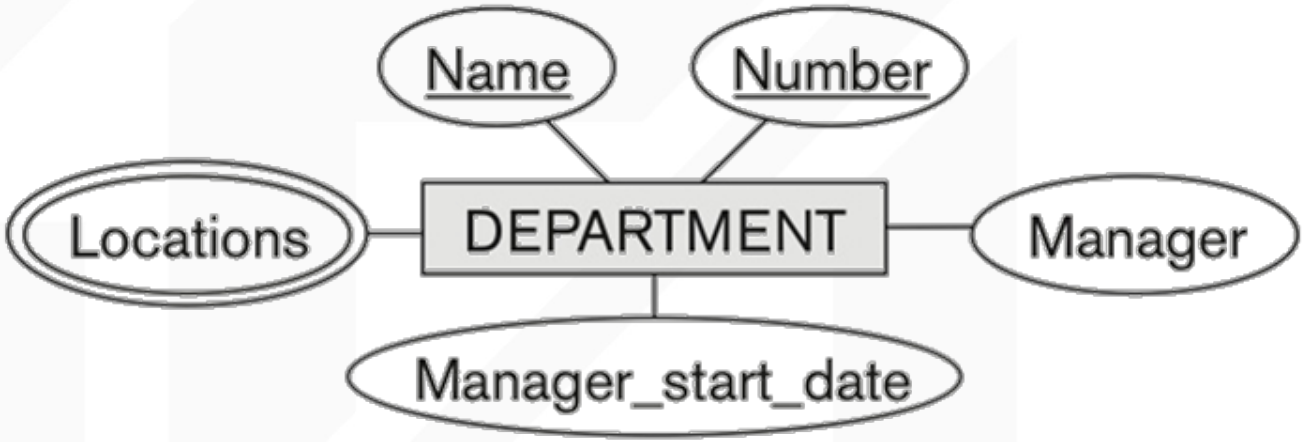
- In ER diagrams, an entity type is displayed in a rectangular box
- Attributes are displayed in ovals
  - Each attribute is connected to its entity type
  - Components of a composite attribute are connected to the oval representing the composite attribute
  - Each key attribute is underlined
  - Multivalued attributes displayed in double ovals

Figure 3.14
Summary of the notation for ER diagrams.

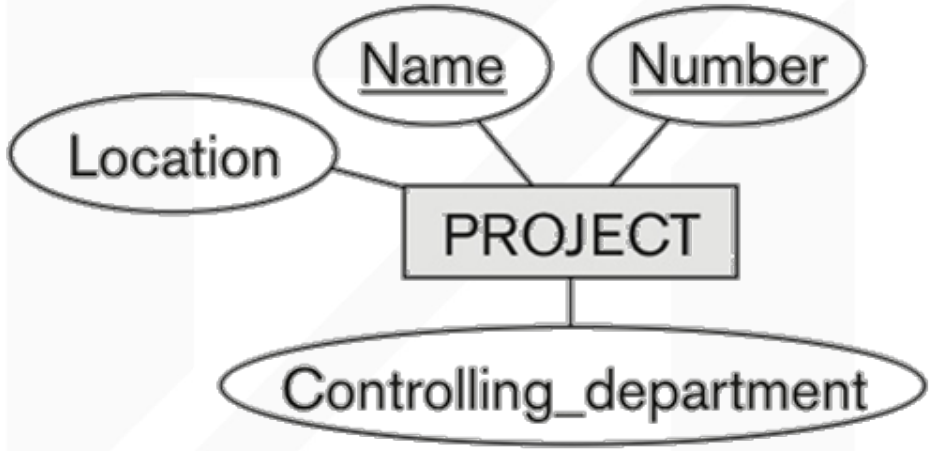
Symbol	Meaning
	Entity
	Weak Entity
	Relationship
	Indentifying Relationship
	Attribute
	Key Attribute
	Multivalued Attribute
	Composite Attribute
	Derived Attribute
	Total Participation of E_2 in R
	Cardinality Ratio 1: N for E_1:E_2 in R
	Structural Constraint (min, max) on Participation of E in R

Initial Conceptual Design

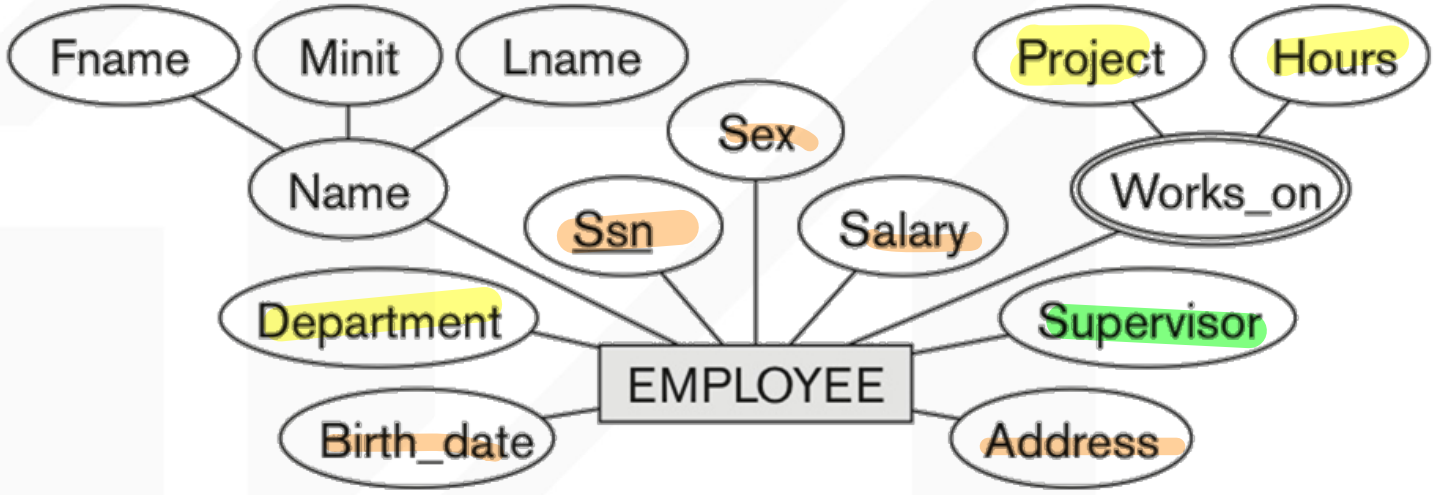
- The company is organized into DEPARTMENTs.
  Each department has a name, number and an employee who *manages* the department.
  We keep track of the start date of the department manager.
  A department may have several locations.



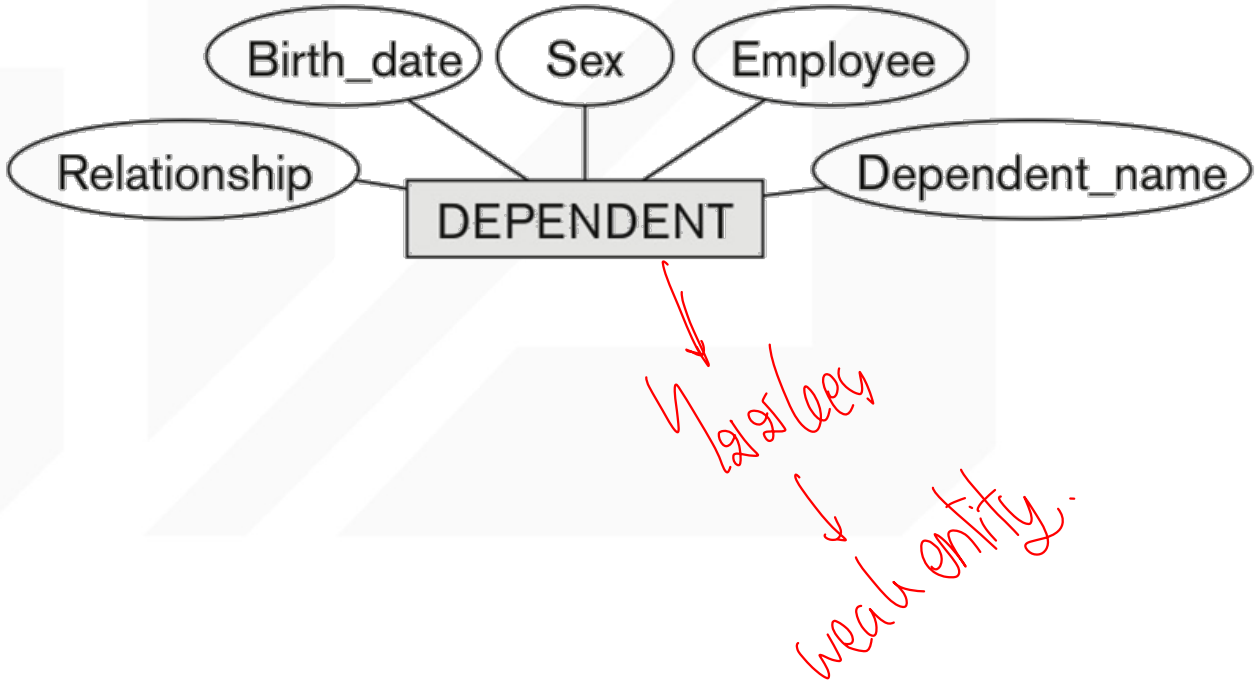
- Each department controls a number of PROJECTs
- Each project has a unique name, unique number and is located at a single location.

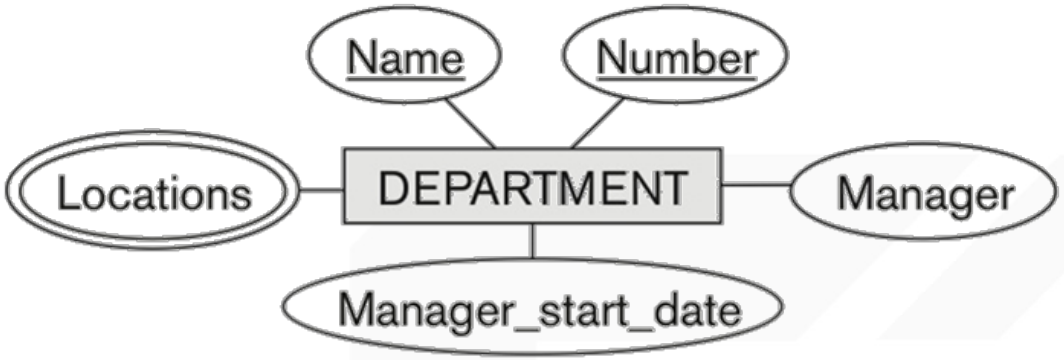


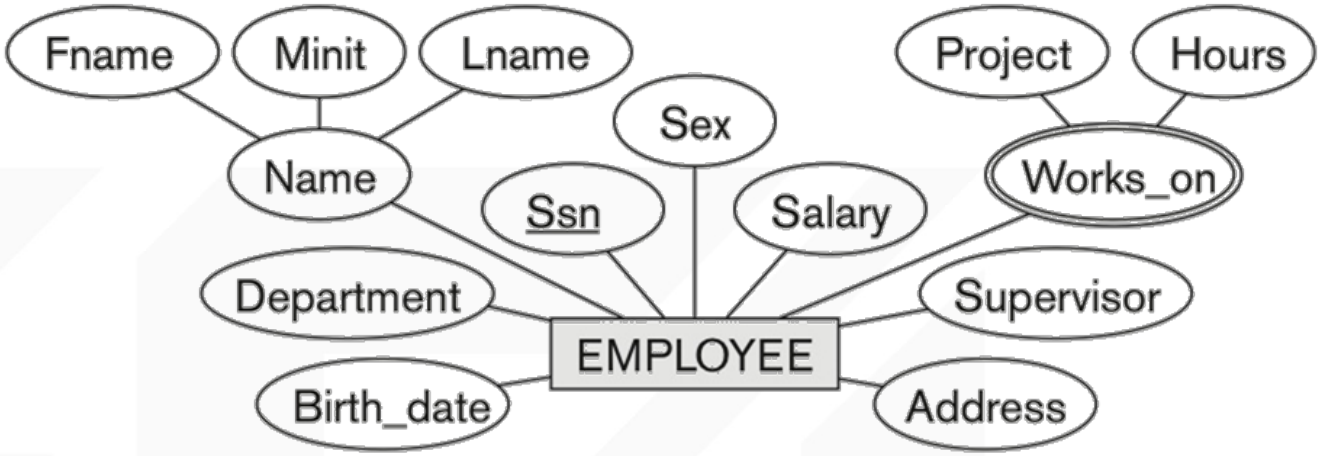
- The database will store each EMPLOYEE's social security number, address, salary, sex, and birthdate.
- Each employee works for one department but may work on several projects.
- The DB will keep track of the number of hours per week that an employee currently works on each project.
- It is required to keep track of the direct supervisor of each employee

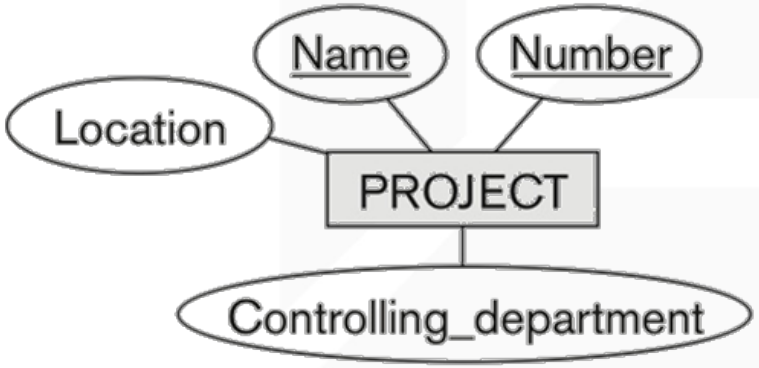


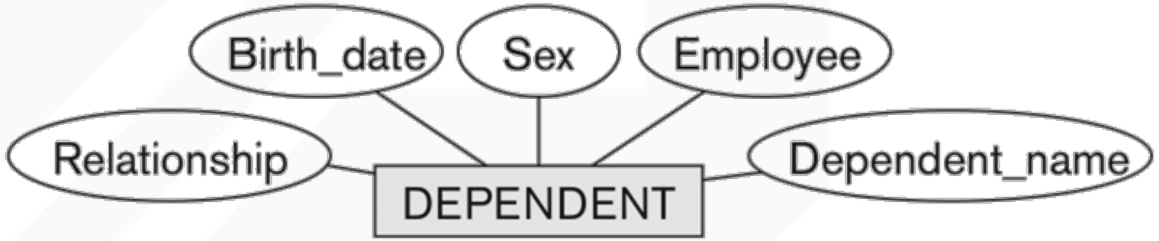
- Each employee may have a number of DEPENDENTs.
- For each dependent, the DB keeps a record of name, sex, birthdate, and relationship to the employee




Name
Number
Locations
DEPARTMENT
Manager
Manager_start_date


Fname
Minit
Lname
Name
Sex
Ssn
Salary
Project
Hours
Works_on
Department
EMPLOYEE
Supervisor
Birth_date
Address


Name
Number
Location
PROJECT
Controlling_department


Birth_date
Sex
Employee
Relationship
DEPENDENT
Dependent_name

- ER model has three main concepts:
  - Entities (and their entity types and entity sets)
  - Attributes (simple, composite, multivalued)
  - Relationships (and their relationship types and relationship sets

Relationships and Relationship Types

- A **relationship** relates two or more distinct entities with a specific meaning.
  - For example,
    EMPLOYEE John Smith *works on* the ProductX PROJECT, or
    EMPLOYEE Franklin Wong *manages* the Research DEPARTMENT.

- Relationships of the same type are grouped or typed into a **relationship type**.
  - For example,
    - The WORKS_ON relationship type in which EMPLOYEEs and PROJECTs participate
    - The MANAGES relationship type in which EMPLOYEEs and DEPARTMENTs participate.

- The degree of a relationship type is the number of participating entity types.
  - Both MANAGES and WORKS_ON are binary relationships

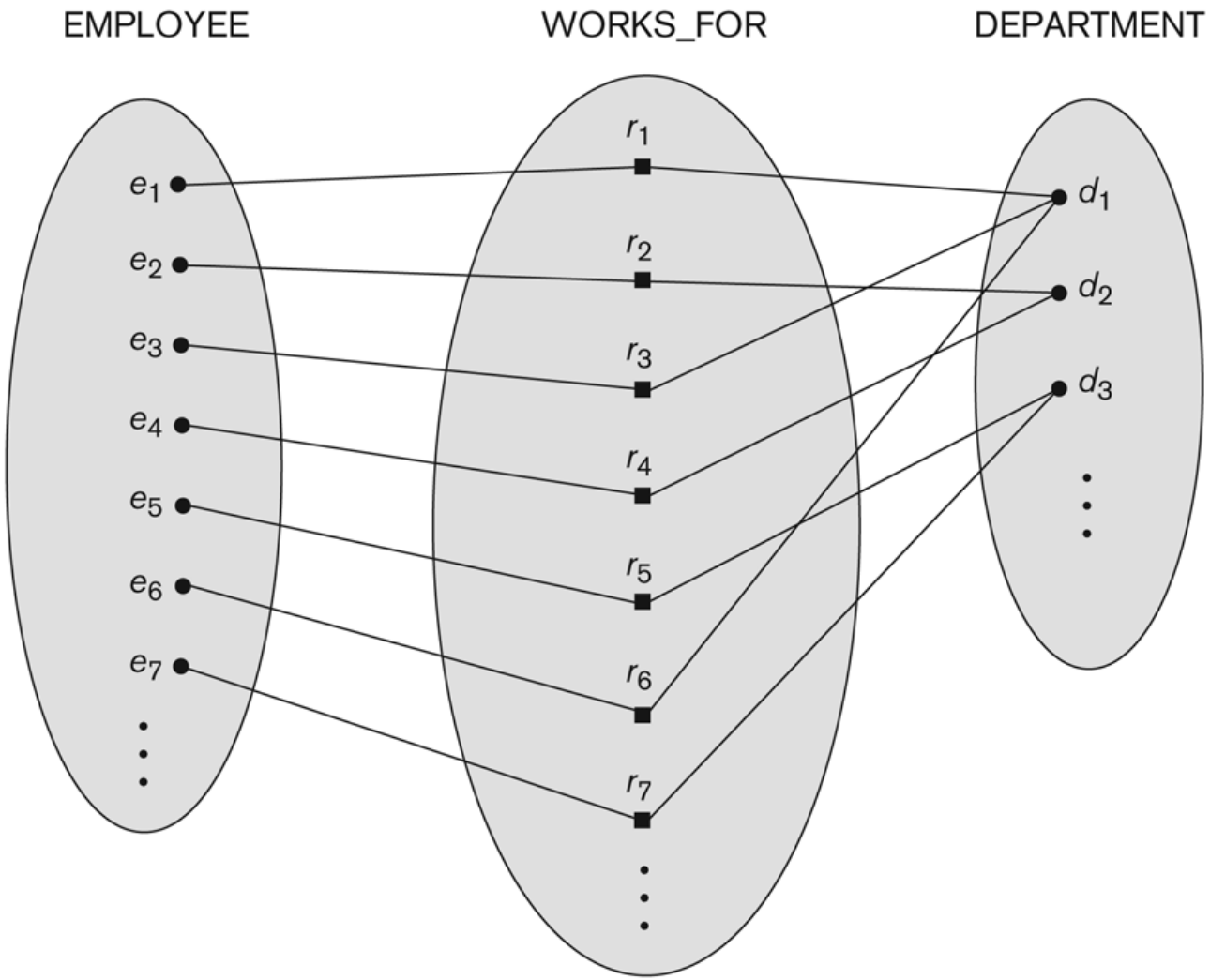
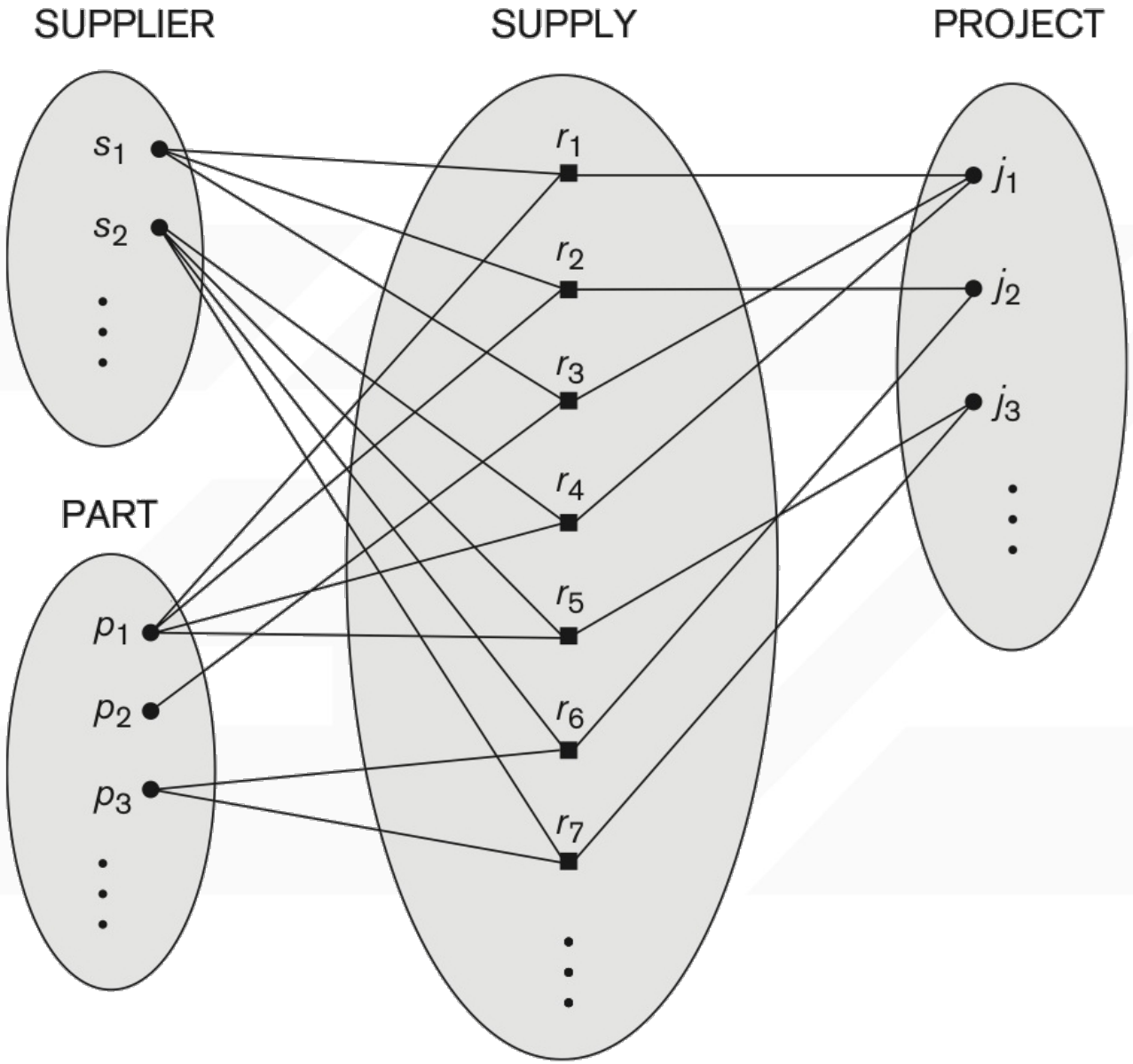


Figure 3.9
Some instances in the WORKS_FOR relationship set, which represents a relationship type WORKS_FOR between EMPLOYEE and DEPARTMENT.

- Degree of a Relationship Type
  - The number of participating entity type.
    - E.g., the WORKS_FOR relationship is of degree two
  - Degree of two is called binary
  - Degree of three is called ternary.


SUPPLIER
SUPPLY
PROJECT
s_1
s_2
PART
p_1
p_2
p_3
r_1
r_2
r_3
r_4
r_5
r_6
r_7
j_1
j_2
j_3

- Recursive Relationship Type
  - A relationship type between the same participating entity type in **distinct roles**
  - Also called a **self-referencing** relationship type.
  - Example: the SUPERVISION relationship

- Example: the SUPERVISION relationship
  - EMPLOYEE participates twice in two distinct roles:
    - supervisor (or boss) role
    - supervisee (or subordinate) role
  - Each relationship instance relates two distinct EMPLOYEE entities:
    - One employee in *supervisor* role
    - One employee in *supervisee* role

- In a recursive relationship type.
  - Both participations are same entity type in different roles.
  - For example,
    SUPERVISION relationships between EMPLOYEE
    (in role of supervisor or boss)
    and
    (another) EMPLOYEE
    (in role of subordinate or worker).

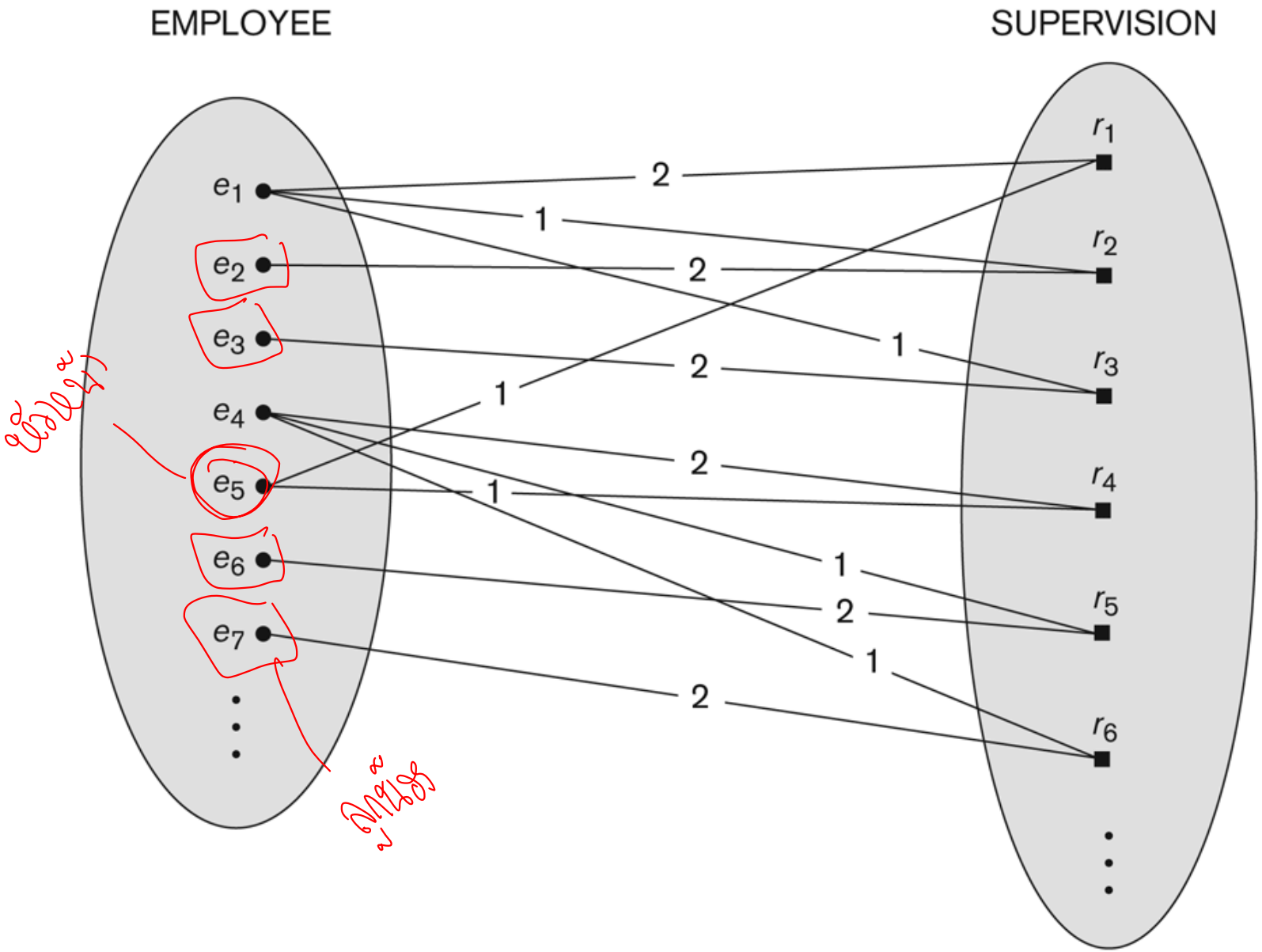


Figure 3.11
A recursive relationship SUPERVISION between EMPLOYEE in the *supervisor* role (1) and EMPLOYEE in the *subordinate* role (2).

Discussion on Relationship Types

- In the refined design, some attributes from the initial entity types are refined into relationships:
  - Manager of DEPARTMENT -> MANAGES
  - Works_on of EMPLOYEE -> WORKS_ON
  - Department of EMPLOYEE -> WORKS_FOR
  - etc

- In general, more than one relationship type can exist between the same participating entity types
  - MANAGES and WORKS_FOR are distinct relationship types between EMPLOYEE and DEPARTMENT
  - Different meanings and different relationship instances

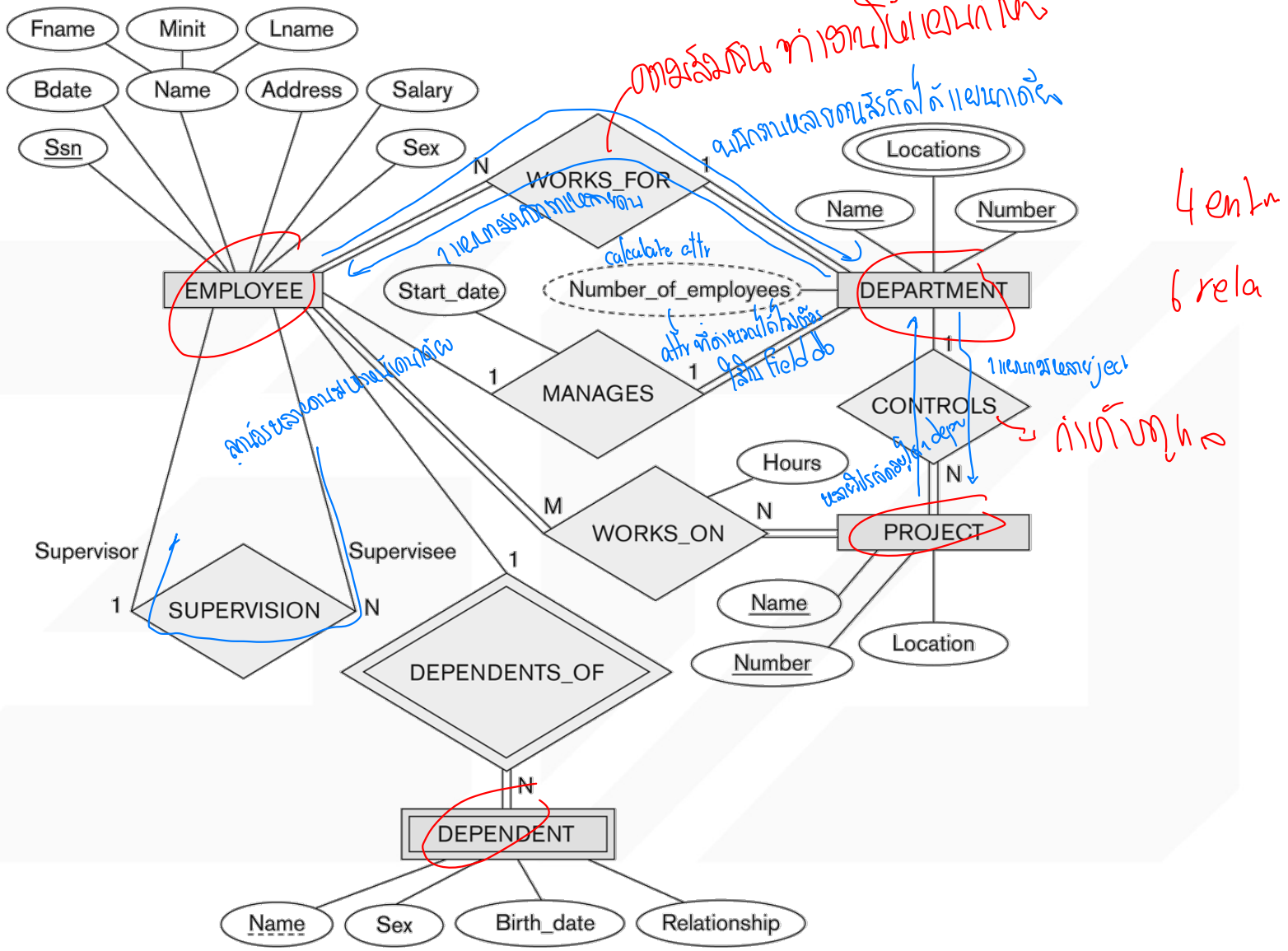
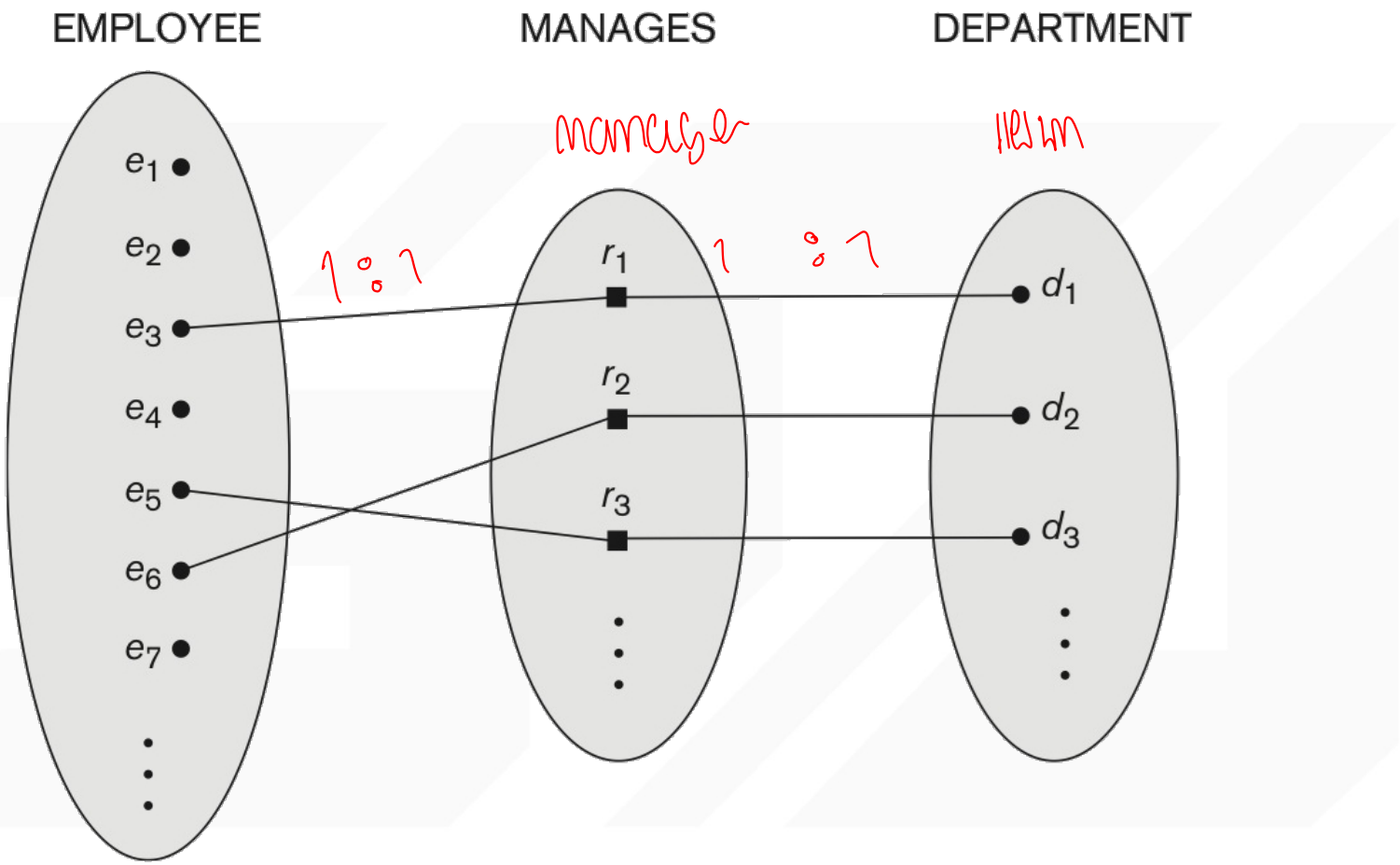


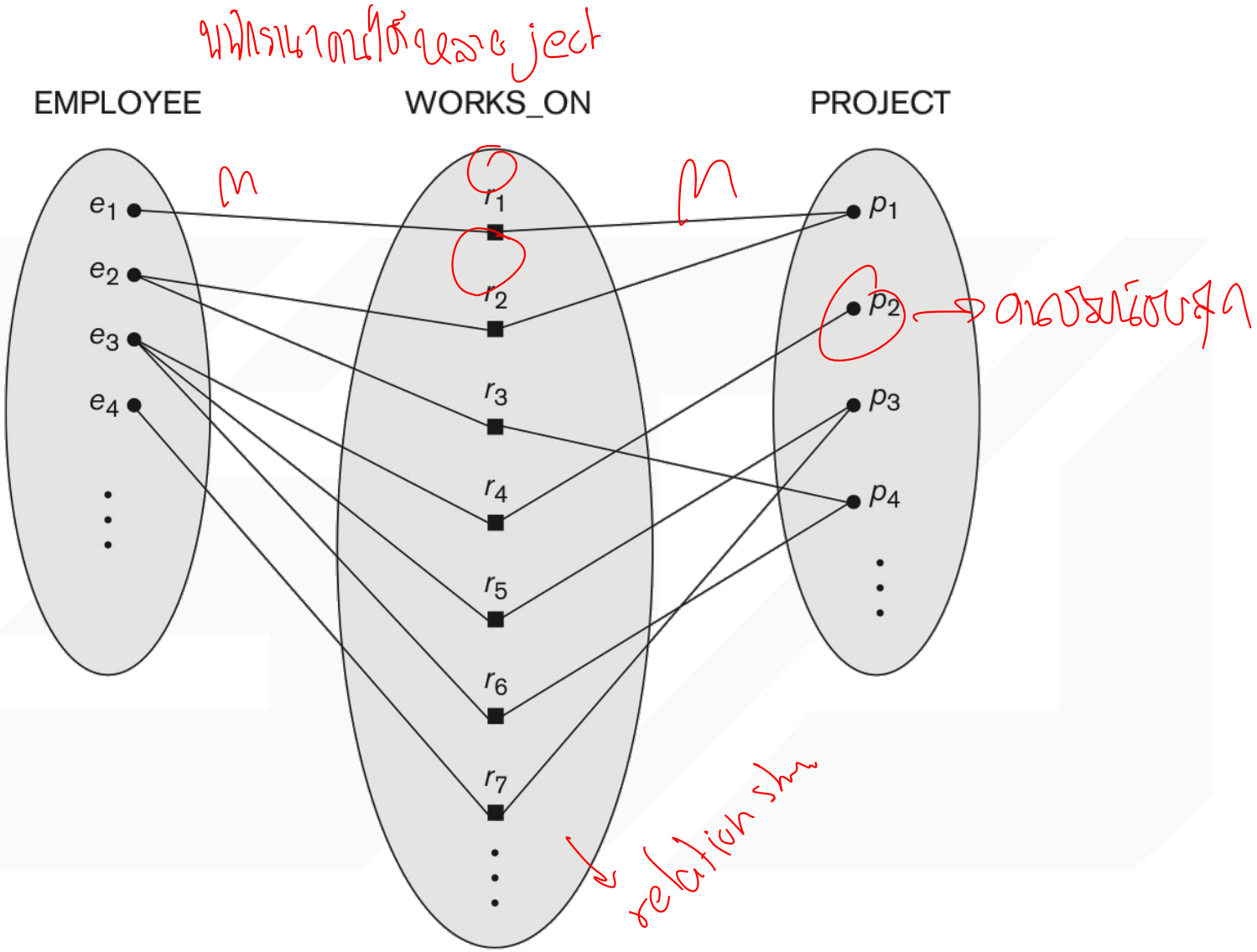
Figure 3.2
An ER schema diagram for the COMPANY database. The diagrammatic notation is introduced gradually throughout this chapter.

Constraints on Relationships

- Constraints on Relationship Types
  - Also known as ratio constraints
  - Cardinality Ratio (specifies *maximum* participation)
    - One-to-one (1:1)
    - One-to-many (1:N) or Many-to-one (N:1)
    - Many-to-many (M:N)
  - Existence Dependency Constraint (specifies *minimum* participation) (also called participation constraint)
    - zero (optional participation, not existence-dependent)
    - one or more (mandatory participation, existence-dependent)



A 1:1 relationship, MANAGES



An M:N relationship, WORKS_ON

Weak Entity Types

- An entity that does not have a key attribute and that is identification-dependent on another entity type.
- A weak entity must participate in an identifying relationship type with an owner or identifying entity type
- 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 relationship type

- Example:
  - A DEPENDENT entity is identified by the dependent's first name, and the specific EMPLOYEE with whom the dependent is related
  - Name of DEPENDENT is the partial key
  - DEPENDENT is a weak entity type
  - EMPLOYEE is its identifying entity type via the identifying relationship type DEPENDENT_OF

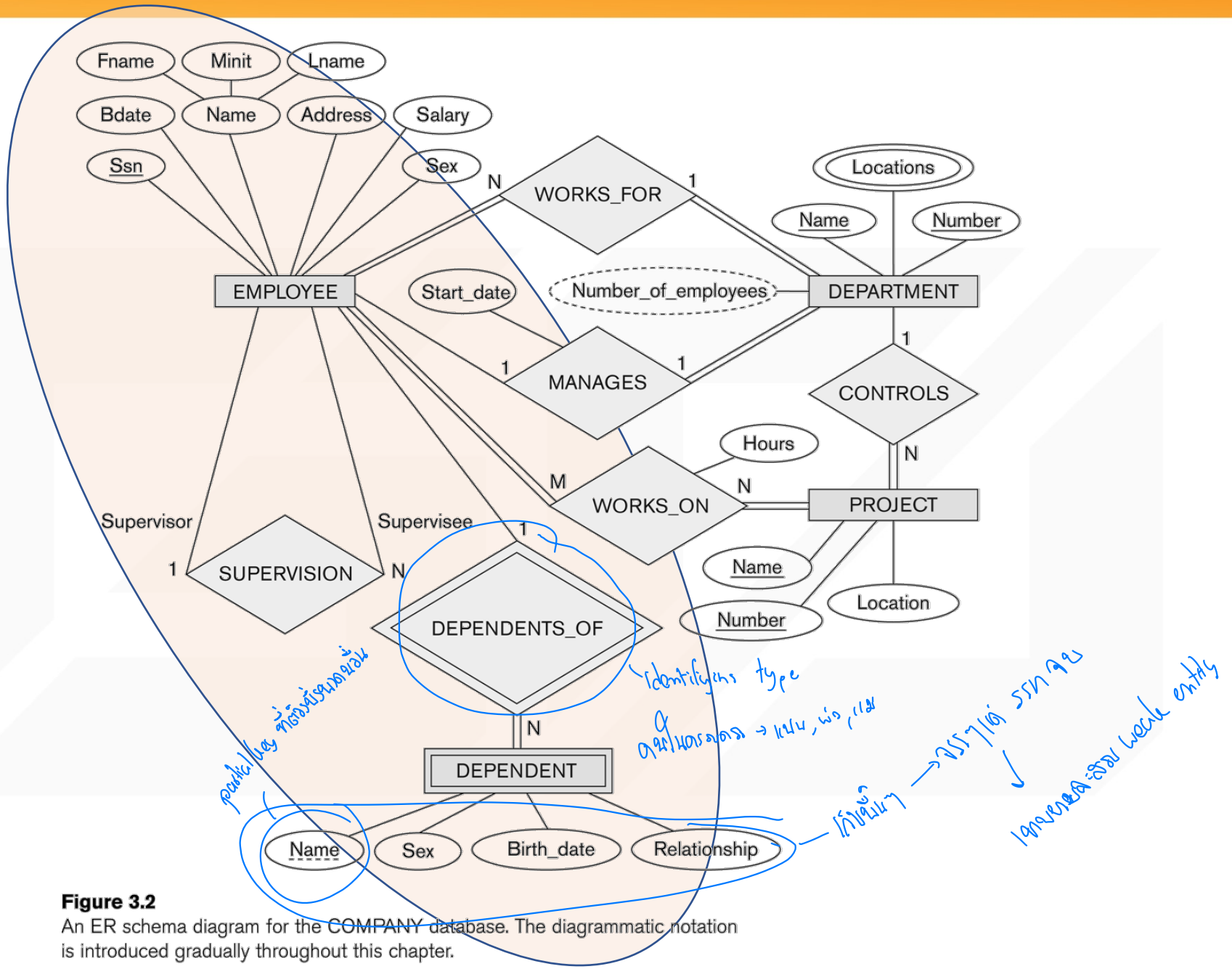


Figure 3.2
An ER schema diagram for the COMPANY database. The diagrammatic notation is introduced gradually throughout this chapter.

Attributes of Relationship Types

- A relationship type can have attributes:
  - For example, HoursPerWeek of WORKS_ON
  - Its value for each relationship instance describes the number of hours per week that an EMPLOYEE works on a PROJECT.
    - A value of HoursPerWeek depends on a particular (employee, project) combination
  - Most relationship attributes are used with M:N relationships
    - In 1:N relationships, they can be transferred to the entity type on the N-side of the relationship

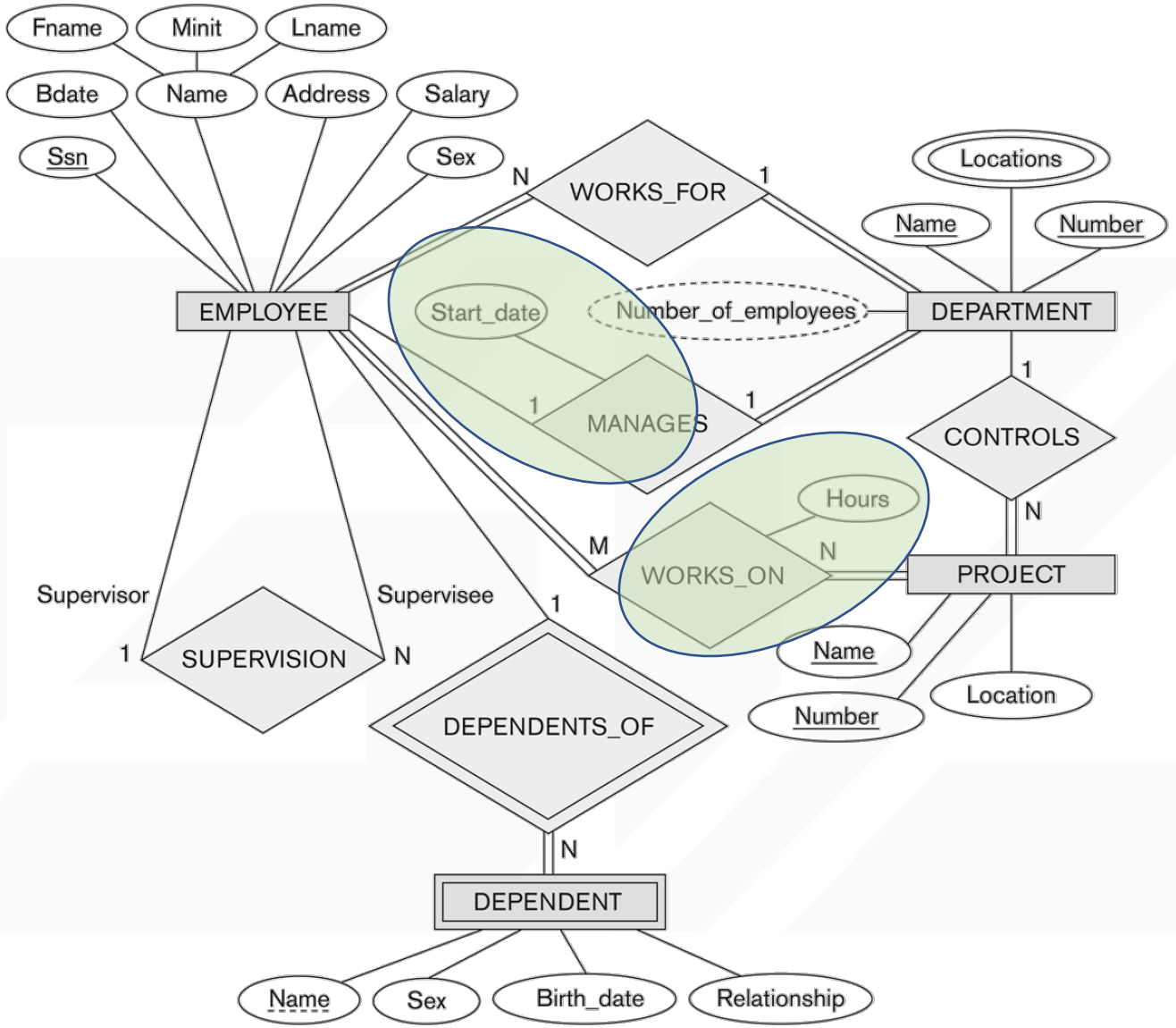
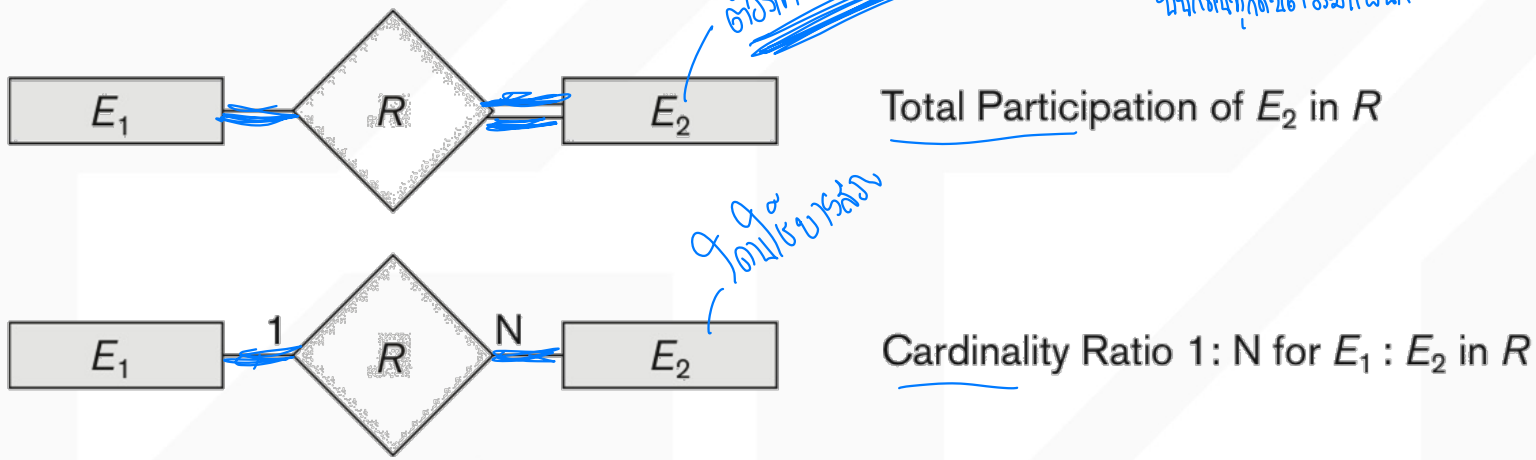


Figure 3.2
An ER schema diagram for the COMPANY database. The diagrammatic notation is introduced gradually throughout this chapter.

Notation for Constraints on Relationships



- Cardinality ratio (of a binary relationship): 1:1, 1:N, N:1, or M:N
  - Shown by placing appropriate numbers on the relationship edges.
- Participation constraint (on each participating entity type): total (called existence dependency) or partial.
  - Total shown by double line, partial by single line.

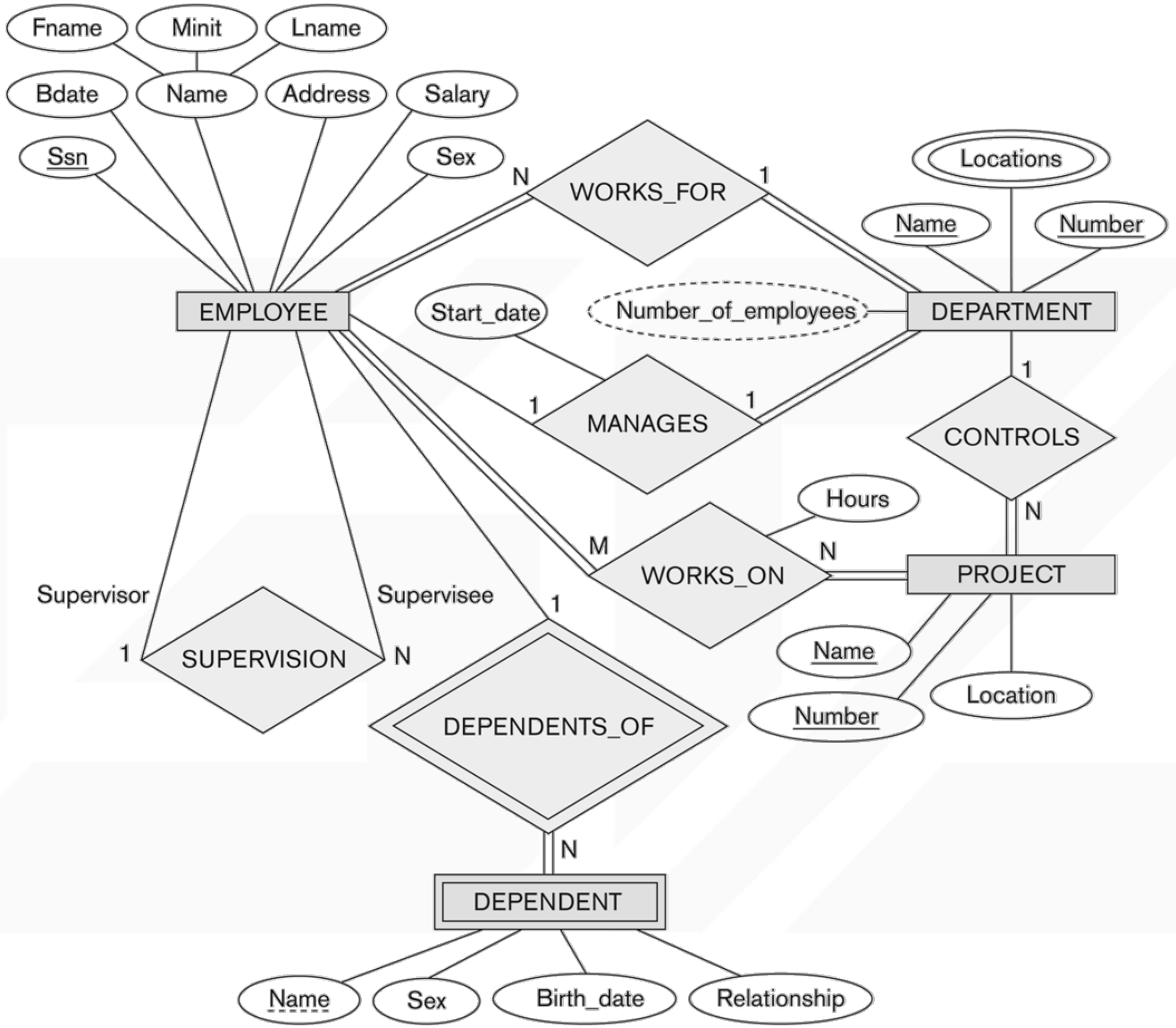
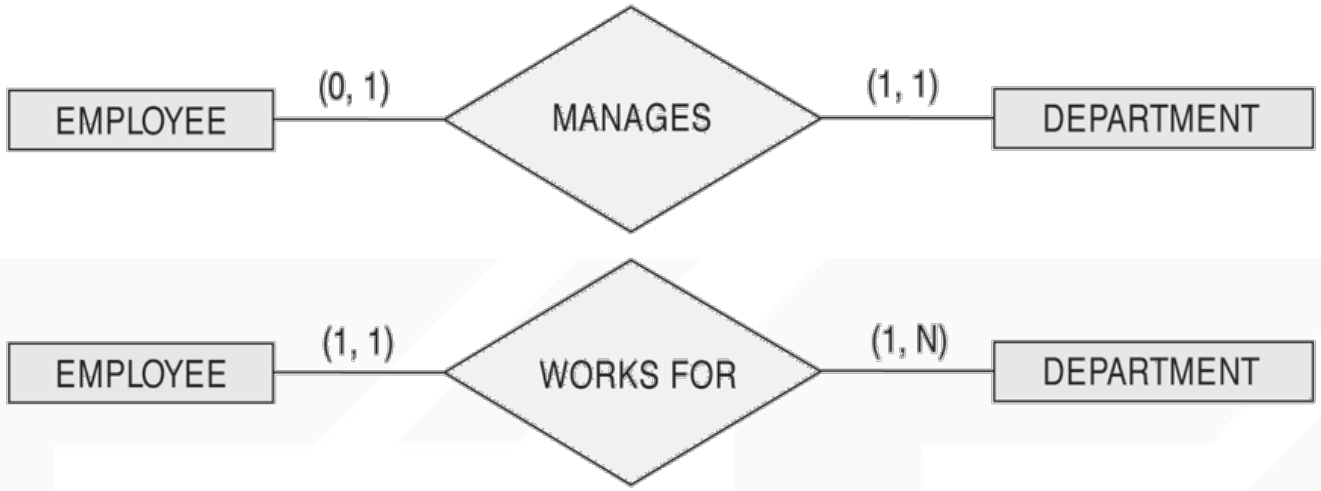


Figure 3.2
An ER schema diagram for the COMPANY database. The diagrammatic notation is introduced gradually throughout this chapter.

Alternative (min, max) notation for relationship structural constraints:

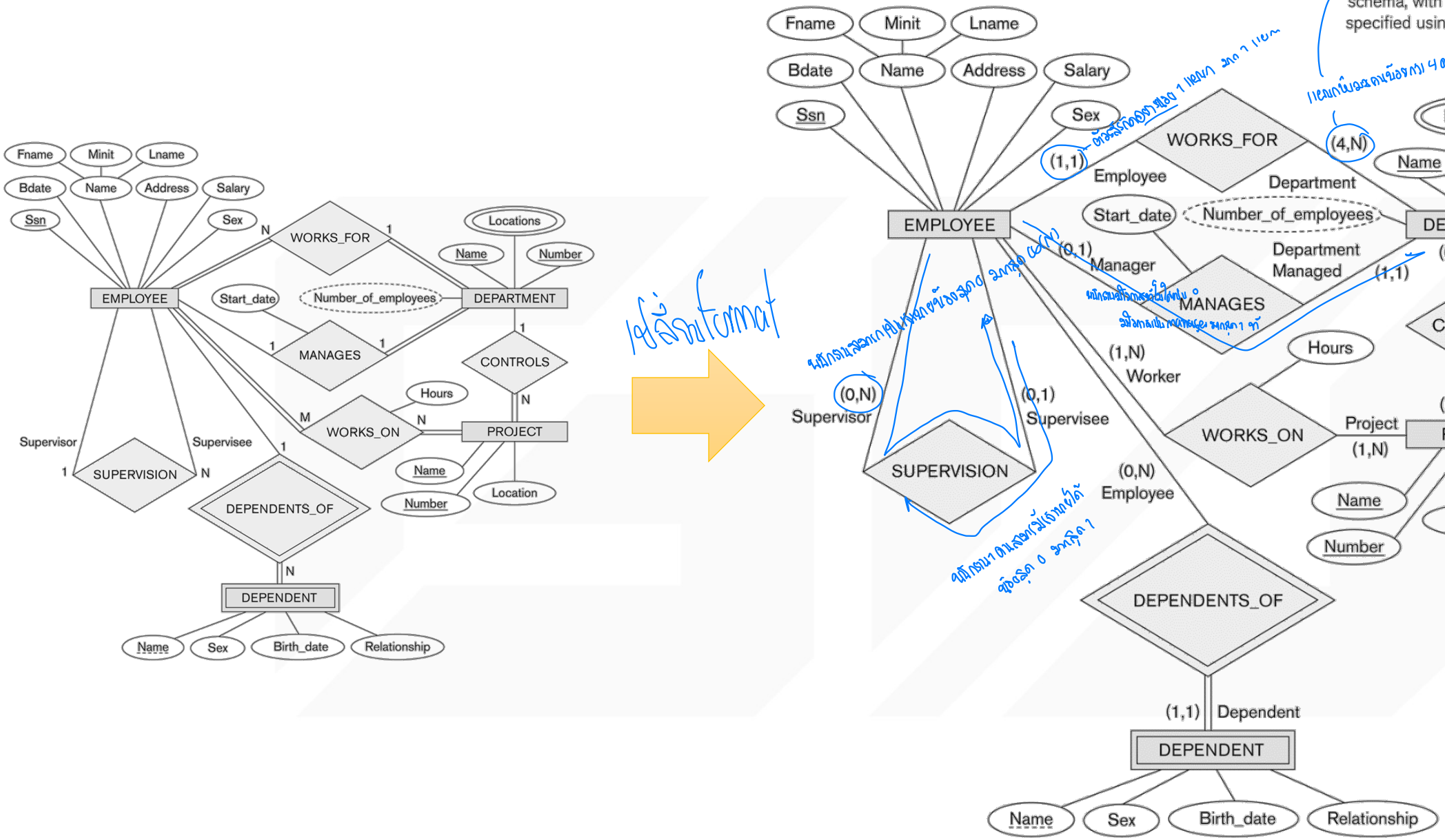
- Specified on each participation of an entity type E in a relationship type R
- Specifies that each entity e in E participates in at least *min* and at most *max* relationship instances in R
- Default(no constraint): min=0, max=n (signifying no limit)
- Must have min ≤ max, min ≥ 0, max ≥ 1
- Derived from the knowledge of mini-world constraints

( ) → min max notation



- Examples:
  - A department has exactly one manager and an employee can manage at most one department.
    - Specify (0,1) for participation of EMPLOYEE in MANAGES
    - Specify (1,1) for participation of DEPARTMENT in MANAGES
  - An employee can work for exactly one department but a department can have any number of employees.
    - Specify (1,1) for participation of EMPLOYEE in WORKS_FOR
    - Specify (0,n) for participation of DEPARTMENT in WORKS_FOR

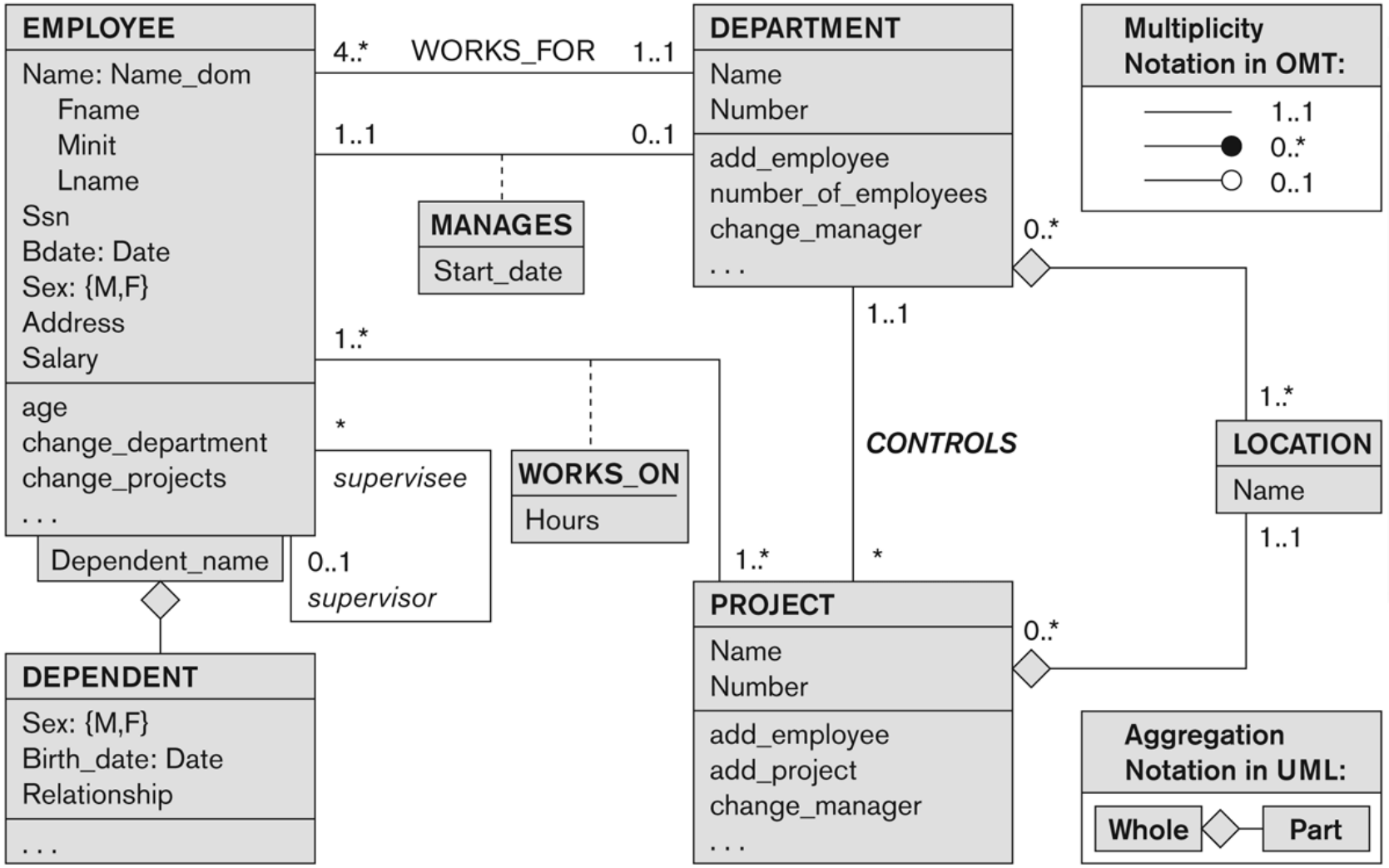
Figure 3.15
ER diagrams for the company schema, with structural constraints specified using (min, max) notation and role names.



UML class diagrams

- Represent classes (similar to entity types) as large rounded boxes with three sections:
  - Top section includes entity type (class) name
  - Second section includes attributes
  - Third section includes class operations (operations are not in basic ER model)
- Relationships (called associations) represented as lines connecting the classes
  - Other UML terminology also differs from ER terminology
- Used in database design and object-oriented software design
- UML has many other types of diagrams for software design

UML class diagram for COMPANY database schema




?
Q
&
A
!