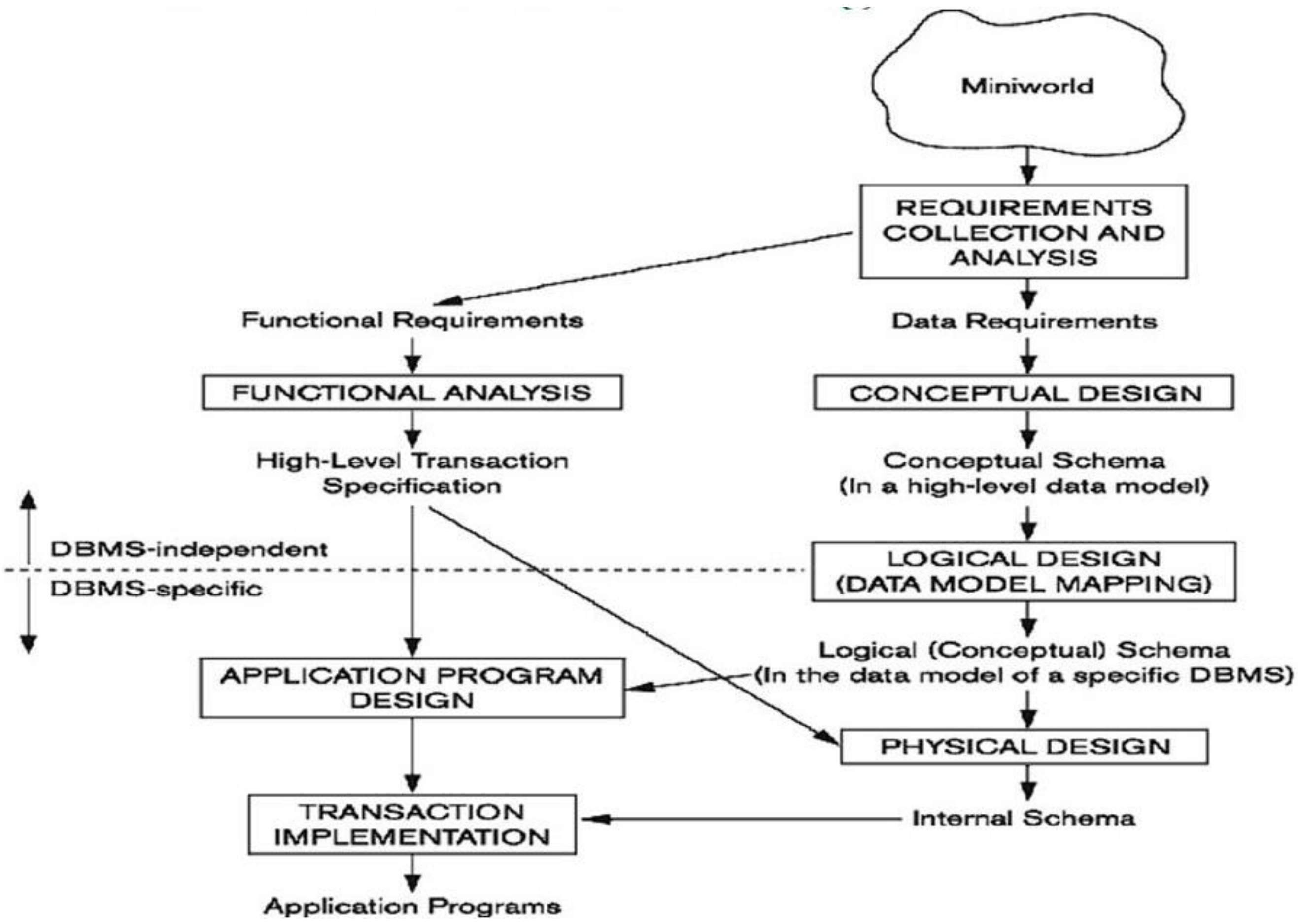
Unit II: Data Modeling and Database Design

Chapter 03: Data Modeling Using the Entity-Relationship(ER) model

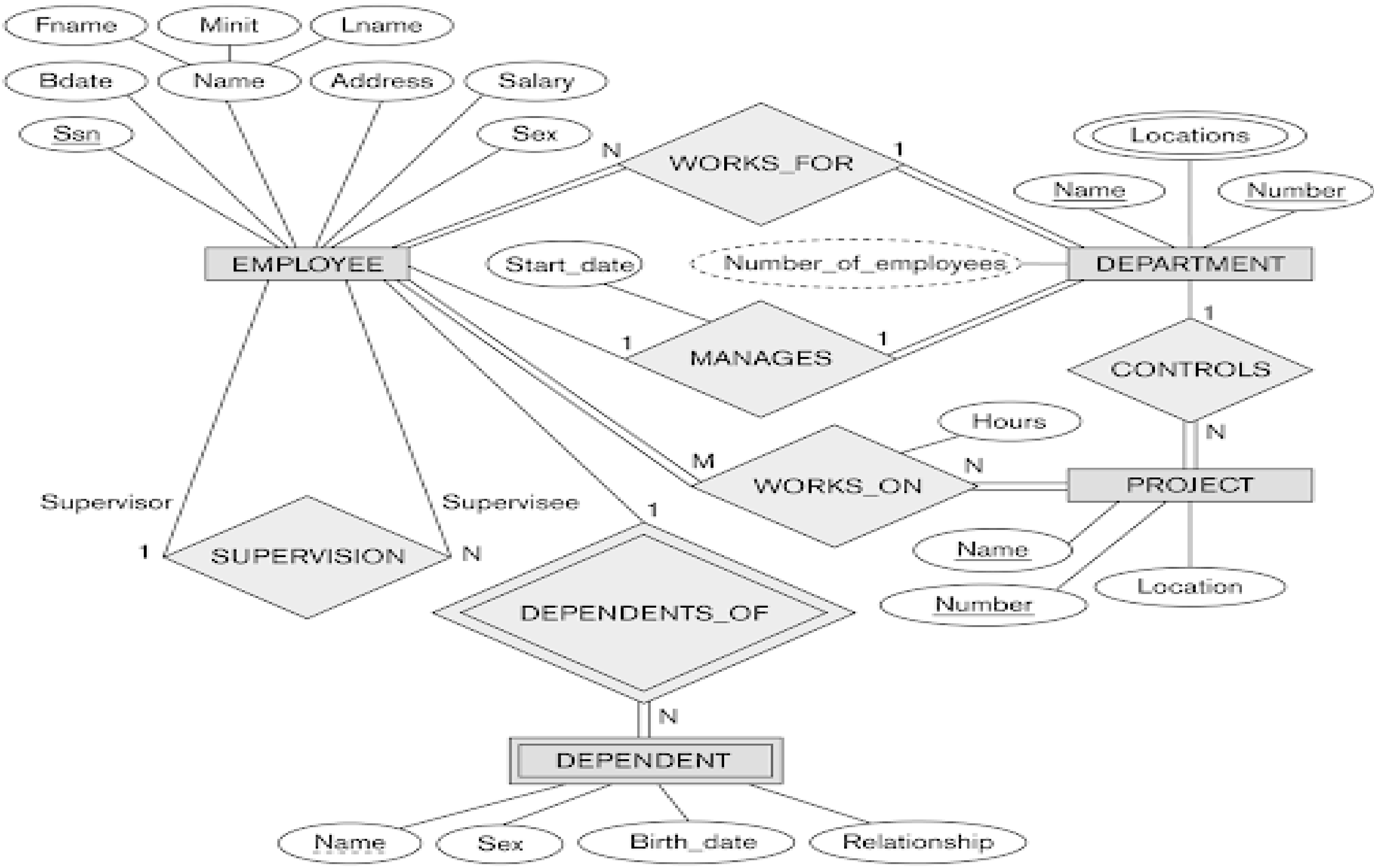
1. Database Application Design: General Process


Miniworld
REQUIREMENTS COLLECTION AND ANALYSIS
Functional Requirements
Data Requirements
FUNCTIONAL ANALYSIS
CONCEPTUAL DESIGN
High-Level Transaction Specification
Conceptual Schema (In a high-level data model)
DBMS-independent
DBMS-specific
LOGICAL DESIGN (DATA MODEL MAPPING)
Logical (Conceptual) Schema (In the data model of a specific DBMS)
APPLICATION PROGRAM DESIGN
PHYSICAL DESIGN
TRANSACTION IMPLEMENTATION
Internal Schema
Application Programs

2. Example company database

- **Requirements of the Company** (oversimplified for illustrative purposes)
  - The company is organized into **DEPARTMENT**s. Each department has a name, number and an employee who ***manages*** the **department**. We keep track of the start date of the **department** manager.
  - The company is organized into **DEPARTMENT**s. Each department has a name, number and an employee who ***manages*** the **department**. We keep track of the start date of the **department** manager.
  - Each **department *controls*** a number of **PROJECTs**. Each **project** has a name, number and is located at a single location.
  - We store each **EMPLOYEE's** social security number, address, salary, sex, and birth-date. Each **employee *works*** *for* **one department** but may ***work on*** **several projects.** We keep track of the number of hours per week that an **employee** currently **works on** each **project**. We also keep track of the ***direct supervisor*** of each **employee**.
  - Each **employee** may *have* a number of **DEPENDENTs**. For each **dependent**, we keep track of their name, sex, birth date, and relationship to **employee.**

ER diagram of the company database



3. ER Model Concepts

Entities and Attributes

- **Entities** are specific objects or things in the mini-world that are represented in the database.

  e.g. the EMPLOYEE, DEPARMENT, PROJECT

- **Attributes** are properties used to describe an entity.

  For example an EMPLOYEE entity may have a Name, SSN, Address, Sex, BirthDate

- Each attribute has a ***value set*** (or data type) associated with it

  e.g. integer, string, subrange, enumerated type, …

Types of Attributes

1. **Simple/atmoic Attribute:** the attribute that has a single atomic value.

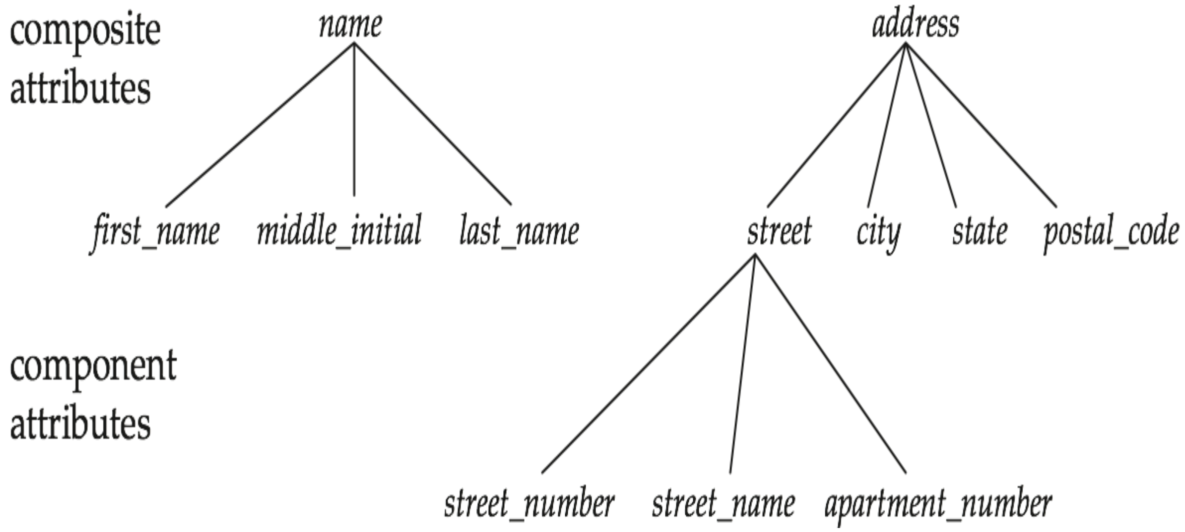
   Each entity has a single atomic value for the attribute.

   e.g. SSN or Sex

2. **Composite Attribute:** the attribute that may be composed of several components. Attribute that can be divided into smaller subparts.

   e.g. Address, Name

   Composition may form a hierarchy where some components are themselves composite.



3. Single valued versus Multivalued Attributes

Single valued attributes: attributes having single value, e.g. age

Multivalued attributes: attributes that has multiple values

e.g. college degree

Entity Types and Key Attributes

- **Entity Type:** refers to the **group of Entities** which have the same basic attributes. e.g. the EMPLOYEE entity type or the PROJECT entity type.

- **Key attribute:** the attribute of an entity type that ***uniquely identify each of the entities***.

- Each entity must have **at least one key attribute** whose value must be unique. e.g. SSN of EMPLOYEE
- A key attribute may be composite.

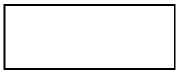
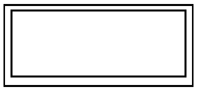
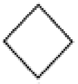
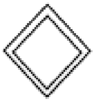
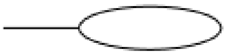
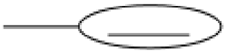
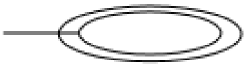
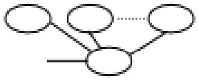
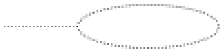
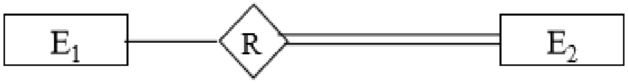
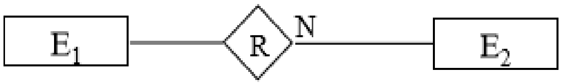
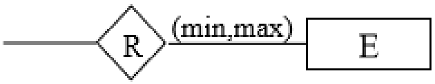
e.g. VehicleNumber is a key of the CAR entity type with components (Number, State)

- An entity type may have more than one key.

  e.g. the CAR entity type may have two keys:

  - VehicleEngineeNumber and
  - VehicleNumber (Number, State), also known as license_plate number.

4. Summary of the ER-Diagram Notation

Symbol	**Meaning**
	--ENTITY TYPE
	--WEAK/IDENTIFYING ENTITY TYPE
	--RELATIONSHIP TYPE
	--IDENTIFYING/WEAK RELATIONSHIP TYPE
	--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 PARTI OF E IN R

5. Weak Entity Types

- An entity that does not have a key attribute
- 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 entity type **Example:**

Suppose that a DEPENDENT entity is identified by the dependent's first name and birthdate, *and* the specific EMPLOYEE that the dependent is related to.

DEPENDENT is a weak entity type with EMPLOYEE as its identifying entity type via the identifying relationship type DEPENDENT_OF

6. Relationships and Relationship Types(1)

- A **relationship** relates two or more distinct entities with a specific meaning.

  E.g. EMPLOYEE e1 **works on** the p1 PROJECT

- Relationships of the same type are grouped or typed into a **relationship type.**

  E.g. 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.

  E.g. Both MANAGES and WORKS_ON are binary relationships.

Example, relationship instances of the WORKS_FOR relationship between EMPLOYEE and DEPARTMENT:

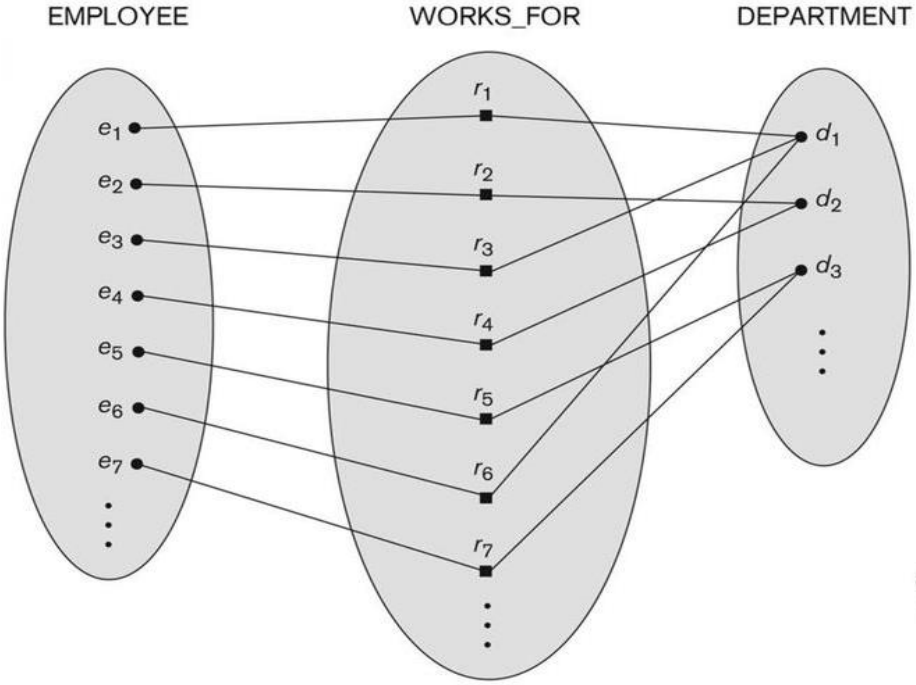
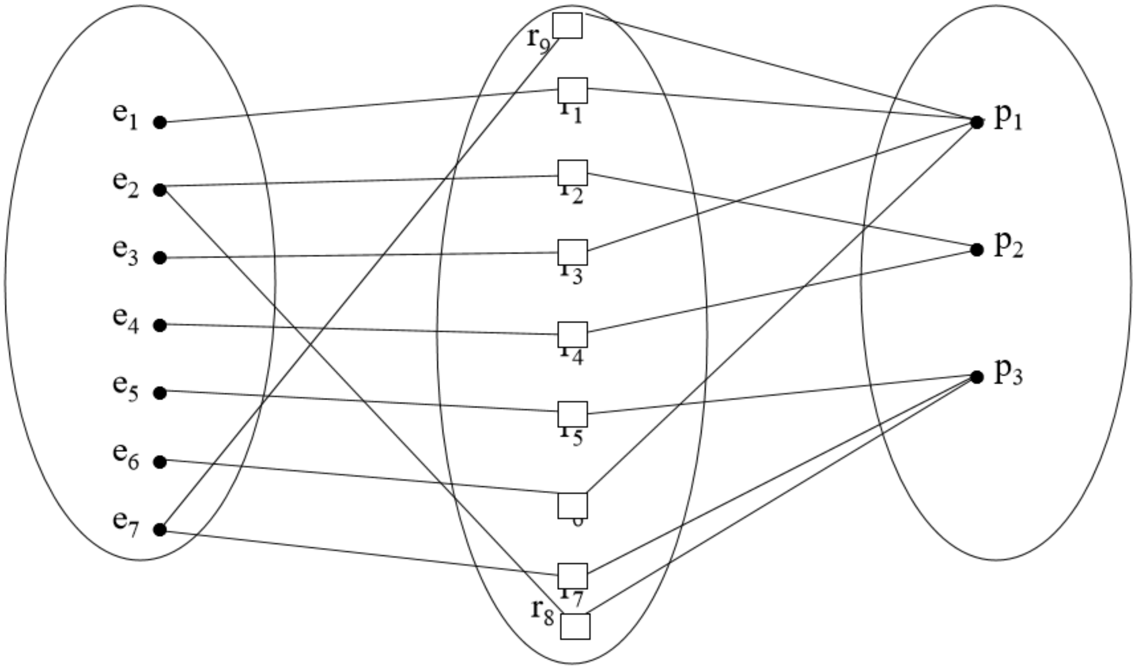


Figure 3.9
Some instances in the WORKS_FOR relationship set, which represents a relationship type WORKS_FOR between EMPLOYEE and DEPARTMENT.

Example, relationship instances of the WORKS_ON relationship between EMPLOYEE and PROJECT:

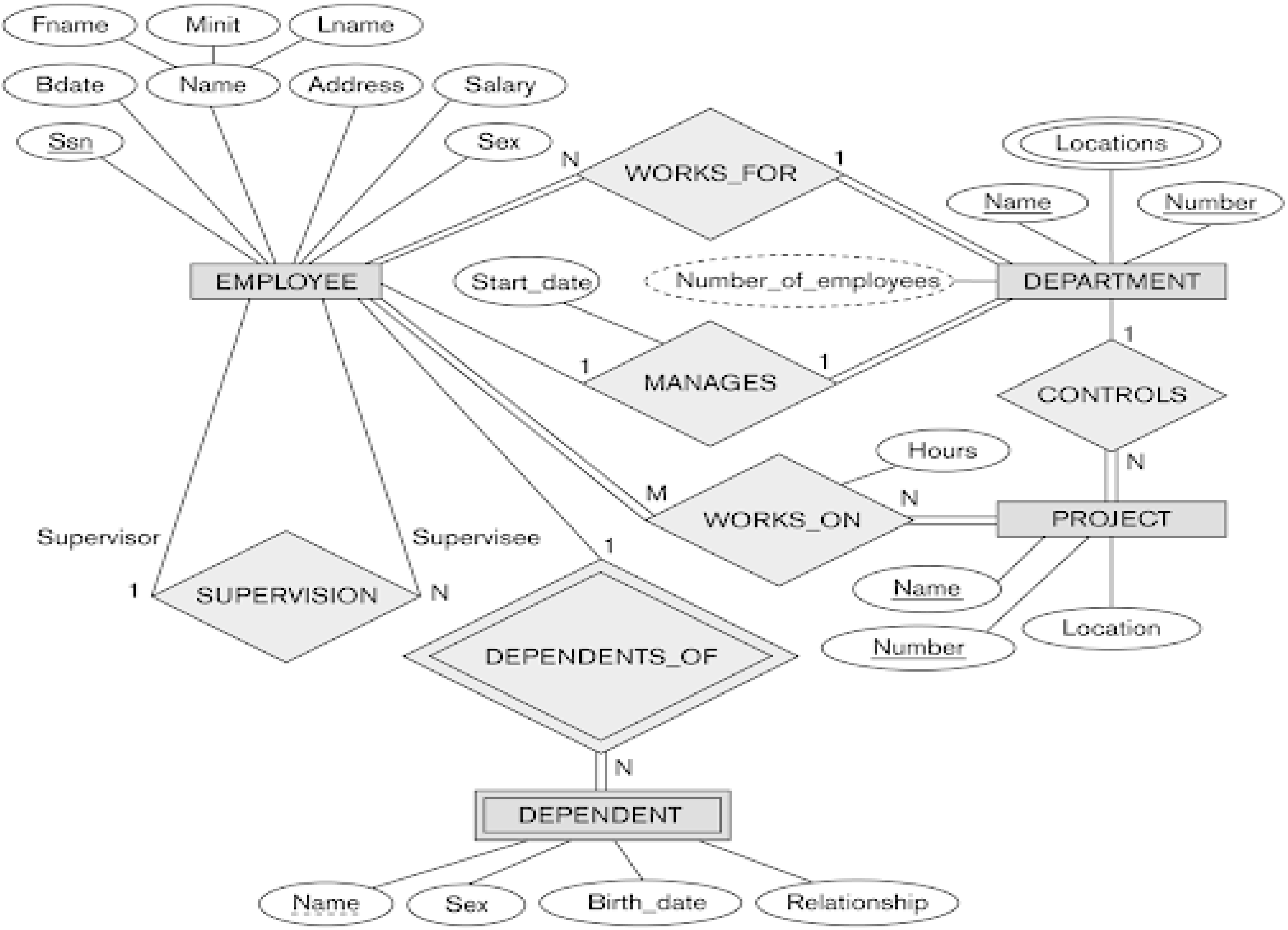


7. Relationships and Relationship Types(2)

✓ More than one relationship type can exist with the same participating entity types.

E.g. MANAGES and WORKS_FOR are distinct relationships between EMPLOYEE and DEPARTMENT, but with different meanings and different relationship instances.

ER DIAGRAM – Relationship Types are: WORKS_FOR, MANAGES, WORKS_ON, CONTROLS, SUPERVISION, DEPENDENTS_OF



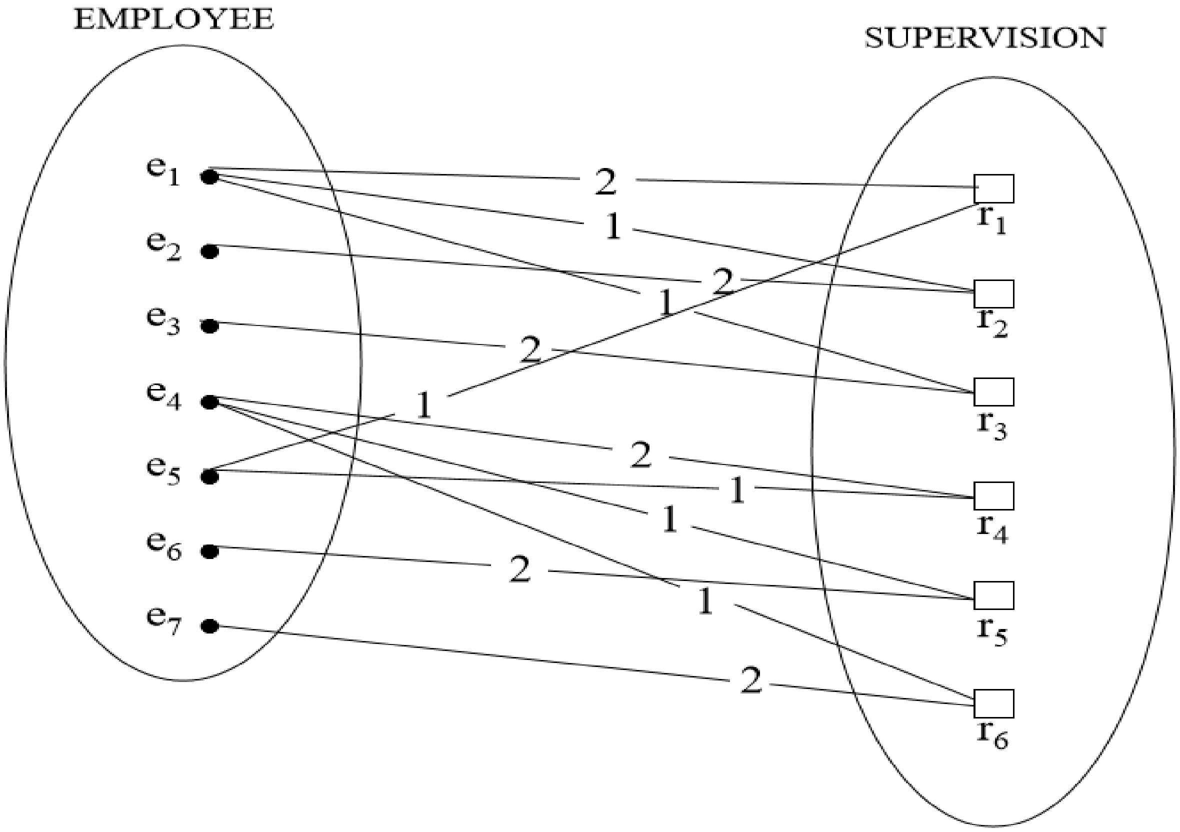
8. Relationships and Relationship Types (3)

We can also have a **recursive relationship type**.

- Both participations are same entity type in different roles. e.g SUPERVISION relationships between EMPLOYEE (in role of supervisor or boss) and (another) EMPLOYEE (in role of subordinate or worker).

- In following figure, first role participation labeled with 1 and second role participation labeled with 2.
- In ER diagram, need to display role names to distinguish participations.

A RECURSIVE RELATIONSHIP:- SUPERVISION



9. Constraints on Relationships

- **Relationship Constraints:** refers to some rules of mini world that specify the limit to the number of relationship instances that an entity can participate in.
- Two types of Relationship Constraints:
  1. Maximum Cardinality (also called Cardinality Ratio) Constraint: specifies the maximum number of relationship instances that an entity can participate in.

  - Possible Cardinality Ratio for an binary relationship types :
    - ✓ **One-to-one** (1:1) binary Relationship :

      E.g MANAGES – an employee can manage only one department at a time and a departments has only one manager
    - ✓ **One-to-many** (1:N) or Many-to-one (N:1) binary Relationship:

      e.g. WORKS_FOR- an employee can work for Only one department and a department employs any numbers of employee.
    - ✓ **Many-to-many(M:N)** binary Relationship:

      e.g. WORKS_ON-an employee can work on several projects and a project can employs many employee.

2. Minimum Cardinality Constraint (also called participation constraint or existence dependency constraints): - Specifies the minimum number of relationship instances that each entity can participate in. i.e this constraint specify whether the existence of an entity depends on it being related to another entity via the relationship type.

Two types of Participation Constraints:

- ✓ Partial Participation (not existence-dependent)
- ✓ Total Participation (mandatory, existence-dependent)

E.g. If a Company rule states that every employee must work for a department, then an employee entity can exist only if it participates in atleast one WORKS_FOR relationship instance. Thus, the Participation of EMPLOYEE in WORKS_FOR is total pariticipation

-Since we do not expect every employee to manage a department, the participation of EMPLOYEE in MANAGES is partial participation.

10. Attributes of Relationship types

A relationship type can have attributes;

e.g. HoursPerWeek of WORKS_ON;

describes the number of hours per week that an EMPLOYEE works on a PROJECT.

11. Structural Constraints – one way to express semantics of relationships

Structural constraints on relationships:

- **Cardinality ratio** (of a binary relationship): 1:1, 1:N, N:1, or M:N

SHOWN BY PLACING APPROPRIATE NUMBER ON THE LINK.

- **Participation constraint** (on each participating entity type): total (called *existence dependency*) or partial.

SHOWN BY DOUBLE LINING THE LINK

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
- Must have min<= max, min>=0, max >=1

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

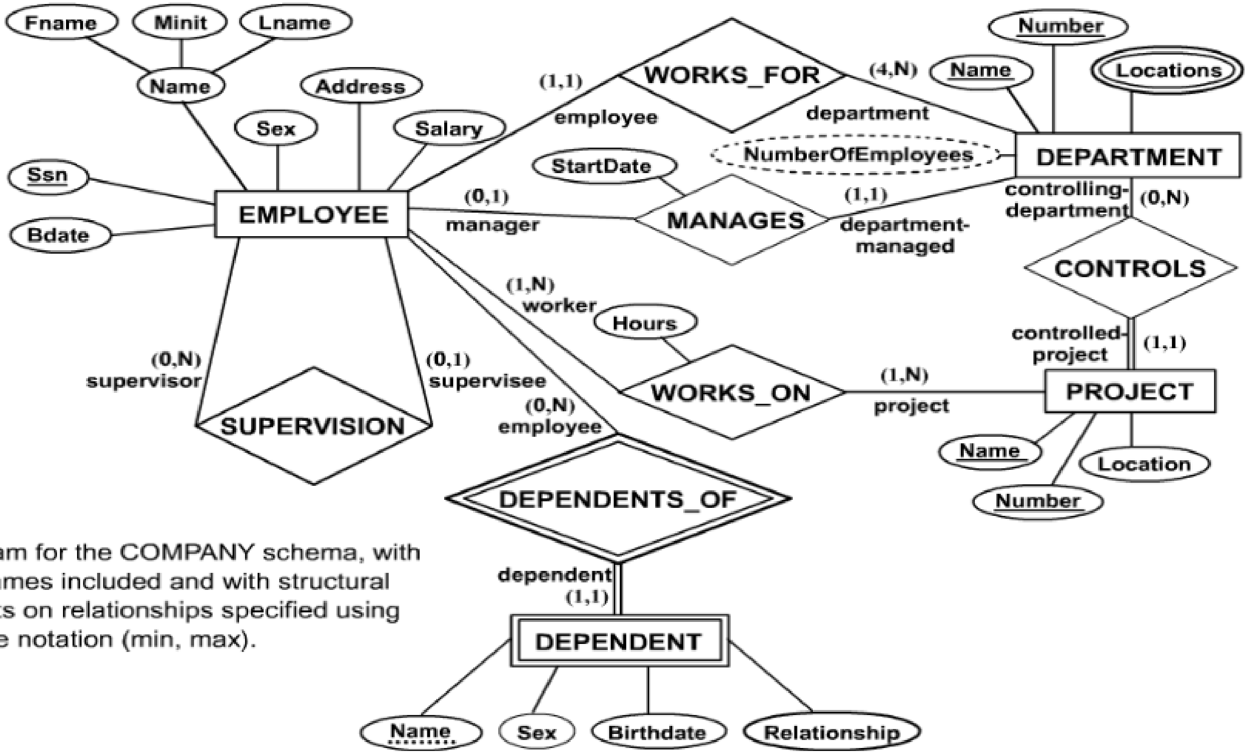
- An employee can work for *exactly one* department but a department can have *any number of employees*.

Specify (1,1) for participation of EMPLOYEE in WORKS_FOR

Specify (0,n) for participation of DEPARTMENT in WORKS_FOR

ER diagram for the company schema:

Alternative ER Notations



ER diagram for the COMPANY schema, with all role names included and with structural constraints on relationships specified using alternative notation (min, max).