

Chapter 4

Data Modeling Using the Entity-Relationship (ER) Model

Data Modeling Using the Entity-Relationship (ER) Model

- **Entity-Relationship (ER) model**
 - Popular high-level conceptual data model
- **ER diagrams**
 - Diagrammatic notation associated with the ER model

Using High-Level Conceptual Data Models for Database Design

- **Requirements collection and analysis**
 - Database designers interview prospective database users to understand and document data requirements
 - Result: **data requirements**
 - **Functional requirements** of the application

Using High-Level Conceptual Data Models (cont'd.)

■ **Conceptual schema**

- Conceptual design
- Description of data requirements
- Includes detailed descriptions of the entity types, relationships, and constraints
- Transformed from high-level data model into implementation data model

Entity Types, Entity Sets, Attributes, and Keys

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

Entities and Attributes

■ Entity

- Thing in real world with independent existence about which we want to store data
- An entity may be an object with a physical existence (student, car, house, or product) or it may be an object with a conceptual existence (a company, a job, or a university course)
- Entity is represented by rectangle

■ Entity Occurrence

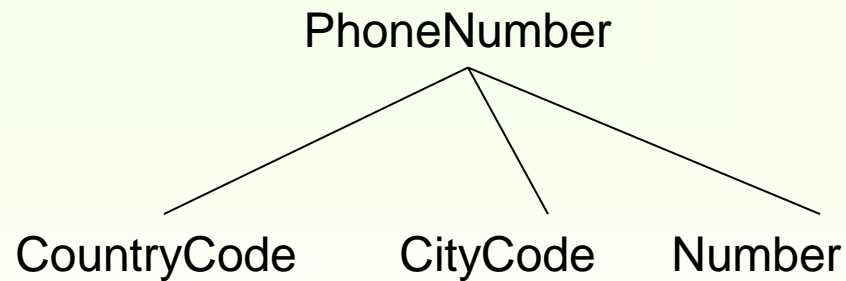
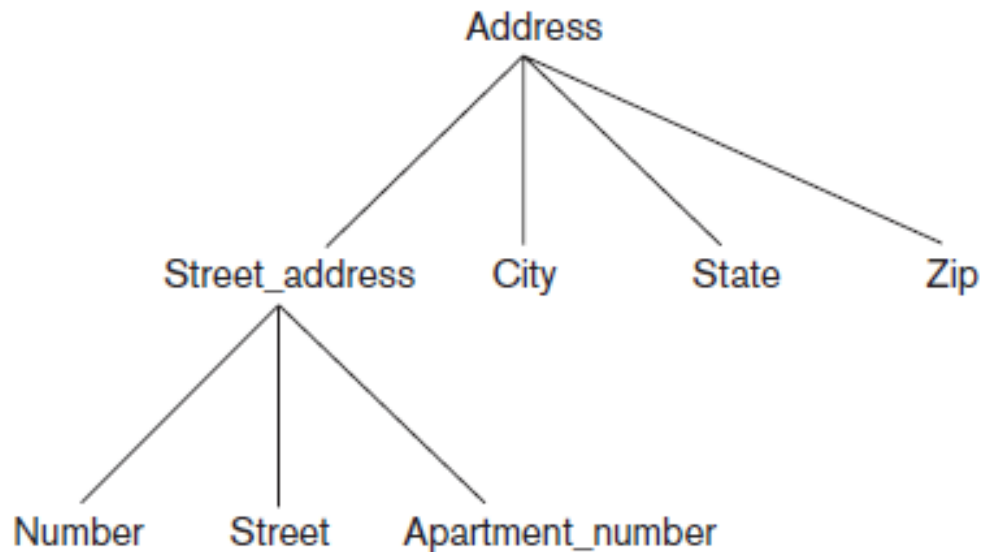
- A uniquely identifiable object of an entity
- Simply saying, entity occurrence is the single tuple or record of an entity

Entities and Attributes (Cont.)

- **Attributes**
 - Particular properties that describe entity
 - Types of attributes:
 - **Composite** versus **simple** (atomic) *attributes*
 - **Single-valued** versus **multivalued** attributes
 - **Stored** versus **derived** attributes
 - **Complex** attributes
- **Attributes are represented by oval**

Attribute Types

- **Composite vs Simple Attributes:**



Attribute Types(cont.)

▪ Multi-valued vs Single valued Attributes:

Attributes that accepts multiple values.

Multi-valued attribute is represented by **double oval**

Example:

- A person may have different phone numbers or may have many e-mails
- Car may have different color
- A person may own multiple cars

Attribute Types(cont.)

▪ **Stored and Derived Attributes:**

Derived attribute means that the value of this attribute can be calculated from another attributes. It is represented by *dotted oval*. The values that are entered and stored in the database in an attribute is called stored attribute

Example:

- A person age may be calculated through the date of birth attribute.
- Number of employees working in a department can be calculated from counting the number of rows of employees working in each department

Attribute Types(cont.)

- **Complex Attributes:**

Complex attribute is both composite and multi-valued attribute

Example:

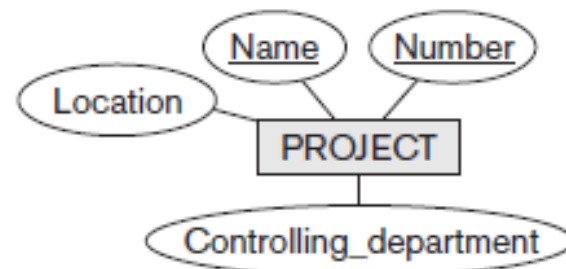
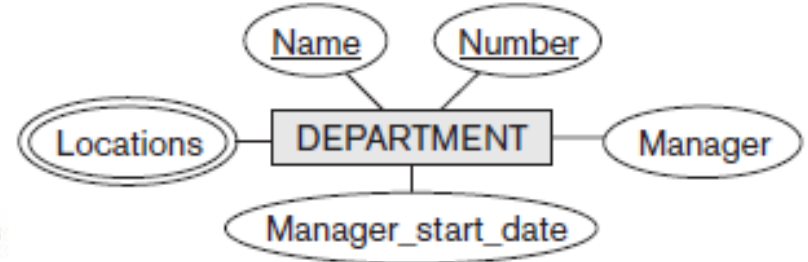
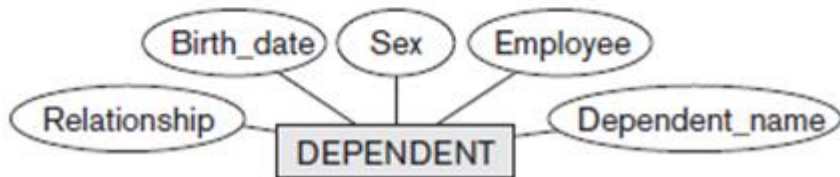
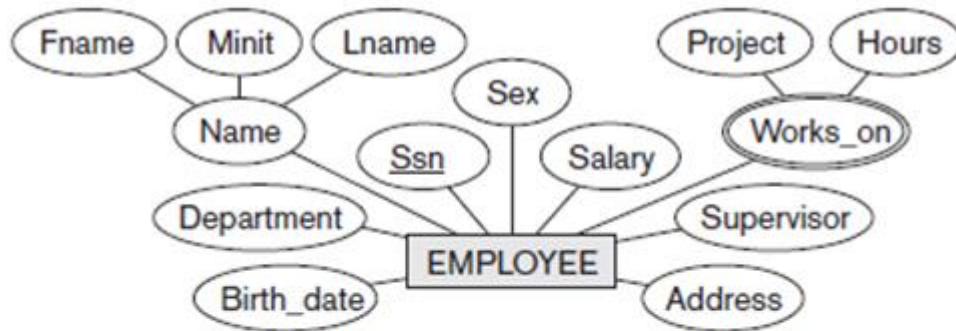
- Address attribute may be composite and multi-valued
- Phone attribute may also be composite and multi-valued

Key Attribute of an Entity

- An entity type usually has one or more attributes whose values are distinct for each individual entity in the entity set. Such an attribute is called a **key attribute**
- An entity may contain **one or more** key attribute
- Key attributes are illustrated by solid line under the attribute



Initial Conceptual Design of the COMPANY Database

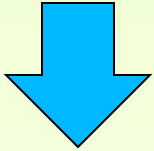


Weak Entity Types

- Do not have key attributes of their own
 - Identified by being related to specific entities from another entity type
- Always has a total participation constraint
- **Represented by:** double rectangle
- *Example:* Hotel/room
Course/section
Parent/Child

Weak Entity Types (Cont.)

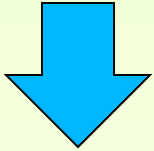
Invoice#	InvoiceDate	EmployeeName	TotalAmount
5092	12/11/2013	Ahmed	230
5093	12/11/2013	Ashraf	780



Invoice#	ProductID	ProductName	Quantity	Unit_price
5092	1121	We need an attribute from the related table to contribute in the key attribute		90
5092	7812			70
5093	1121			90
5093	9021	LMN	3	100
5093	7812	ABC	3	70

Weak Entity Types (Cont.)

Course#	CourseName	No_of_hours	Location
IS123	Database	3	18T
CS389	Networks	3	Hall 1



Course#	Section#	InstructorName	Location	Time
IS123	1	Nermine	Lab 12	9-11
IS123	2	Mai	Lab11	9-11
IS123	3	Laila	Lab 12	12-2
IS123	4	Sarah	Lab 2	2-4
CS389	1	Bishoy	Lab 3	12-2
CS389	2	Nermine	Lab 4	12-2

Relationship Types, Relationship Sets, Roles, and Structural Constraints

■ **Relationship**

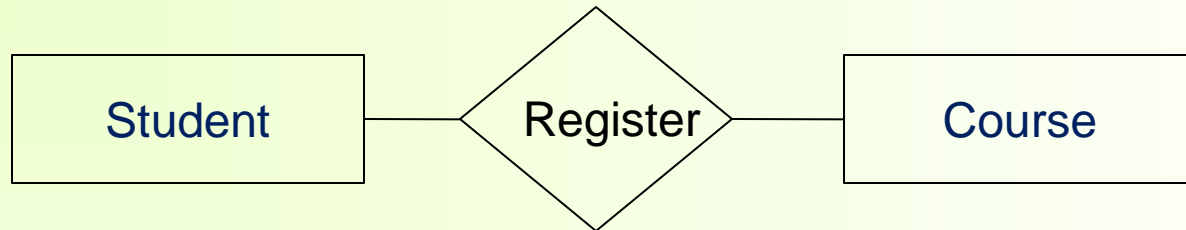
- When an attribute of one entity type refers to another entity type
- Represent references as relationships not attributes
- Represented by diamond

Relationship Degree

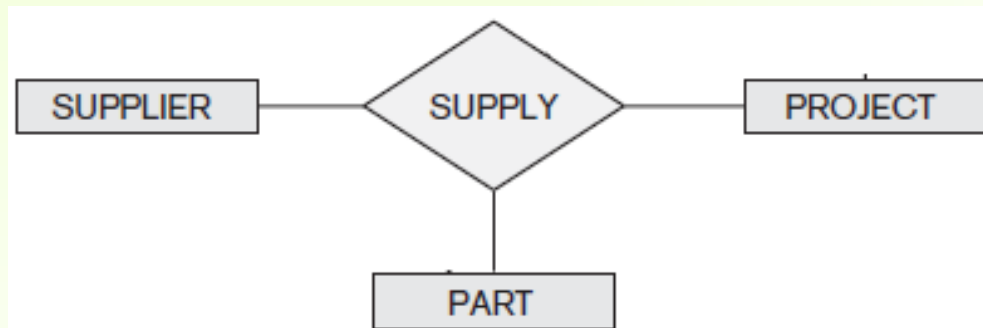
- **Degree** of a relationship type
 - Number of participating entity types
 - **Binary, ternary**
 - **Binary relationship:** relationship between 2 entities
 - **Ternary relationship:** relationship between 3 entities
 - **Remember:** degree of relation ?

Relationship Degree (Cont.)

Binary
relationship

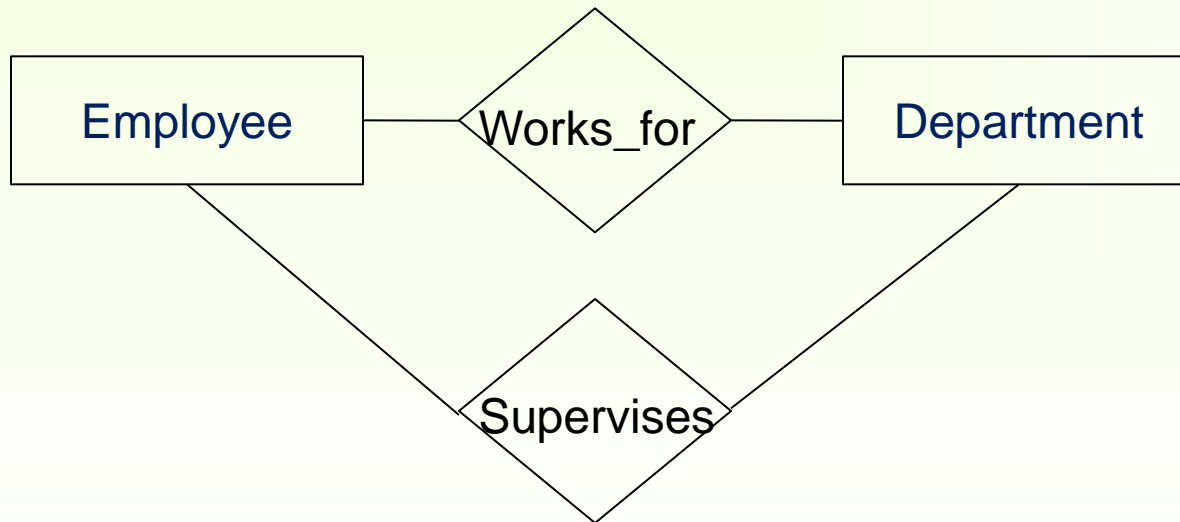


Ternary
relationship



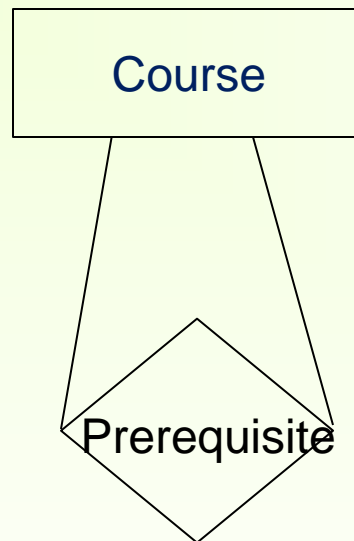
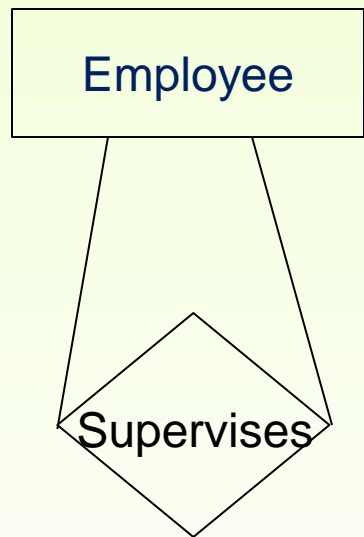
Role Names

- Role name signifies role that a participating entity plays in each relationship
 - Same entity type participates more than once in a relationship type in different roles
 - Must specify role name



Recursive Relationship

- A relationship type where the *same entity type participates more than once in different roles*.

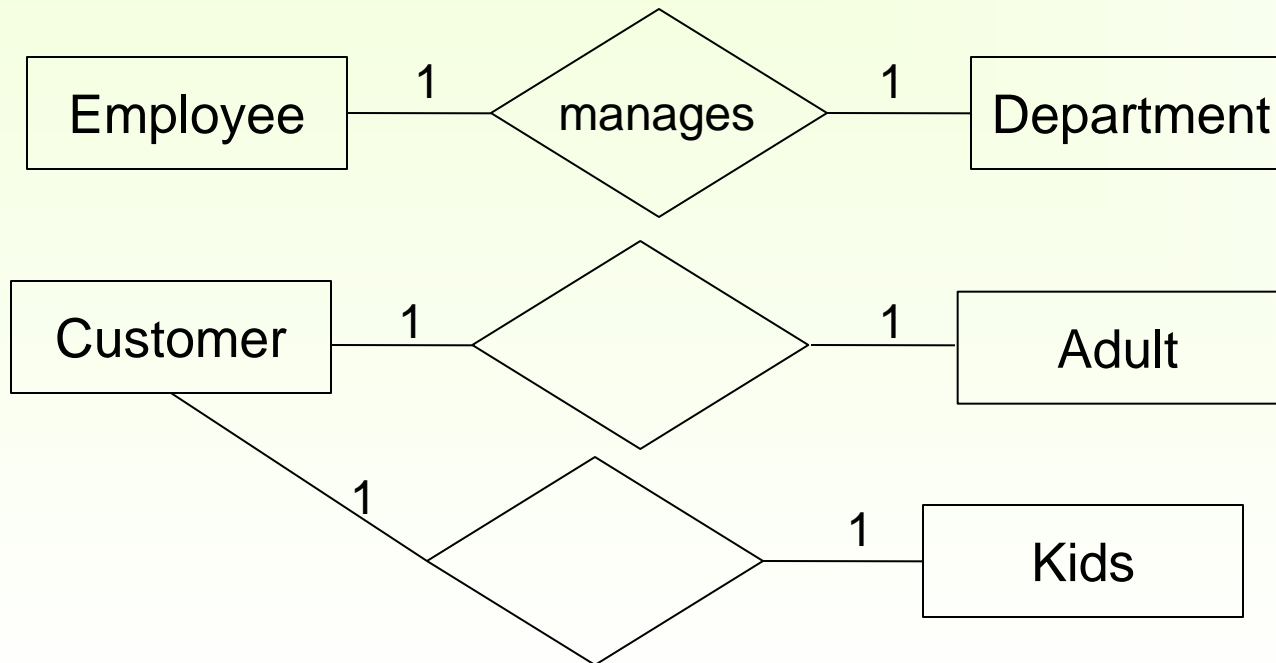


Constraints on Binary Relationship Types

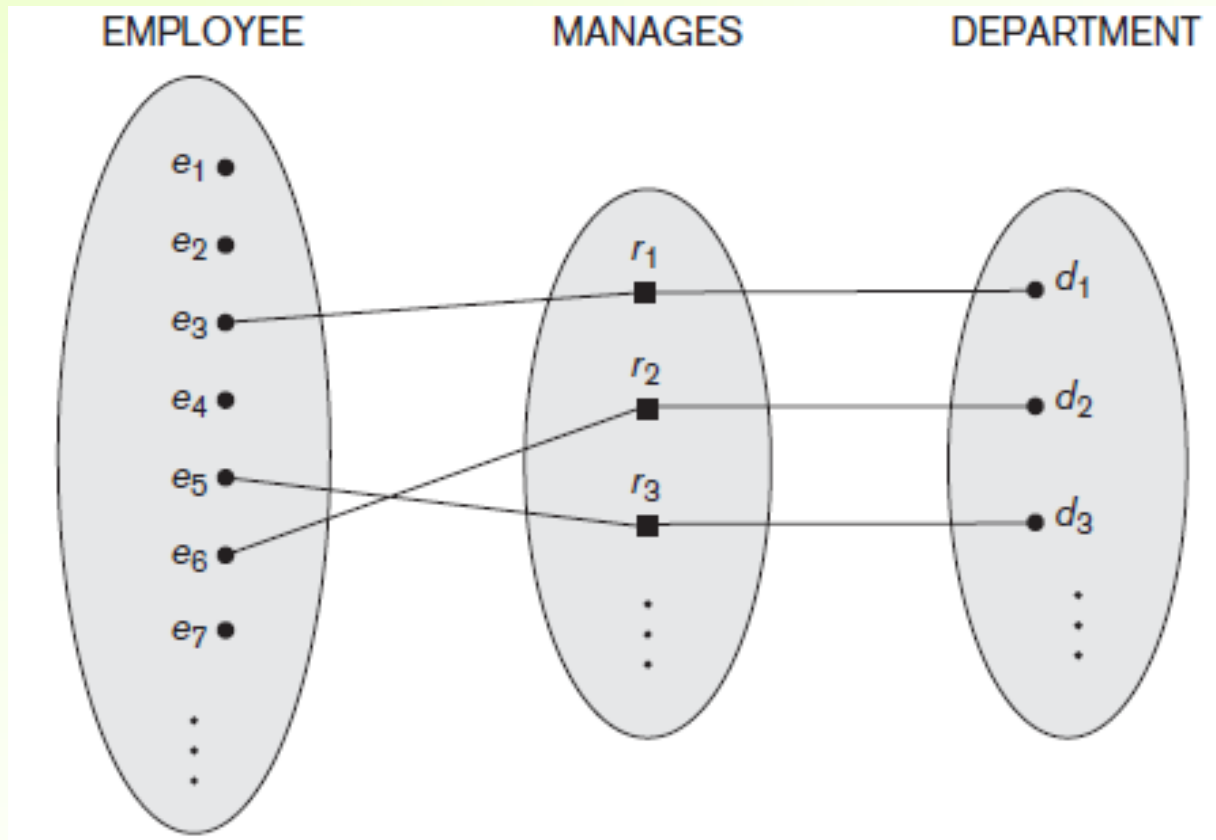
- **Cardinality ratio** for a binary relationship
 - Specifies maximum number of relationship instances that entity can participate in
- **Participation constraint**
 - Specifies whether existence of entity depends on its being related to another entity
 - Types: **total** and **partial**

Cardinality Ratio

- 1:1 Relationship: one entity instance in an entity is related to one entity instance in the other entity in the relationship
 - Ex: employee manages department

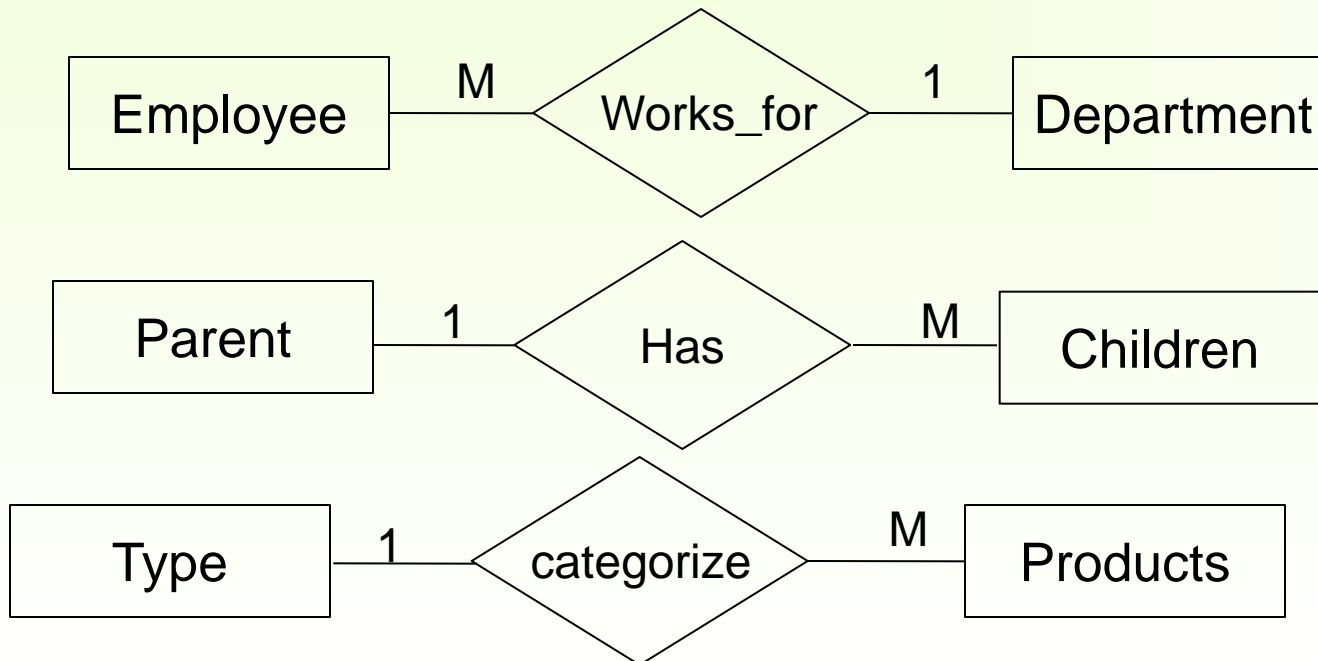


1:1 Relationship

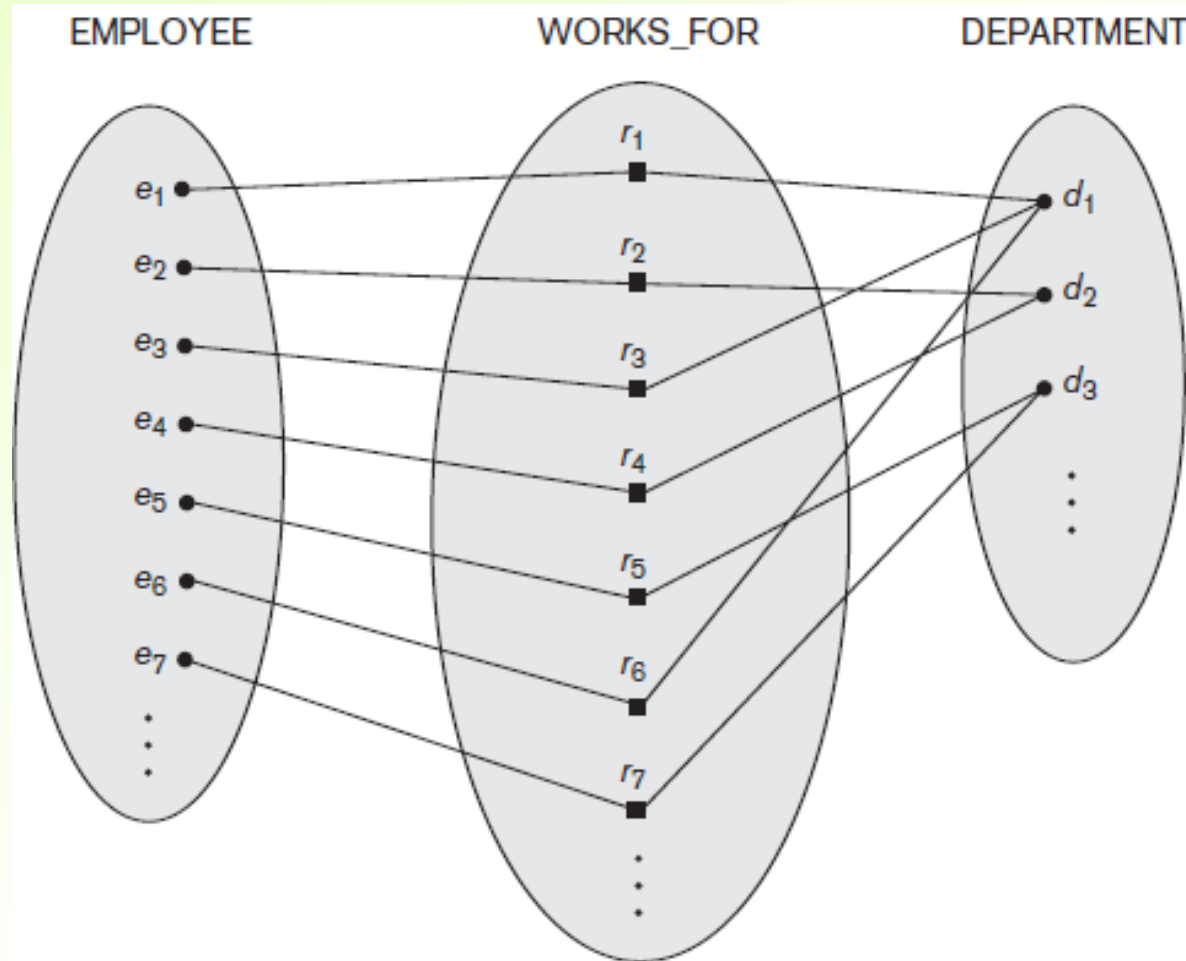


Cardinality Ratio (Cont.)

- 1:M Relationship: one entity instance in an entity is related to many entity instances in the other entity in the relationship
 - Ex: employee works for department

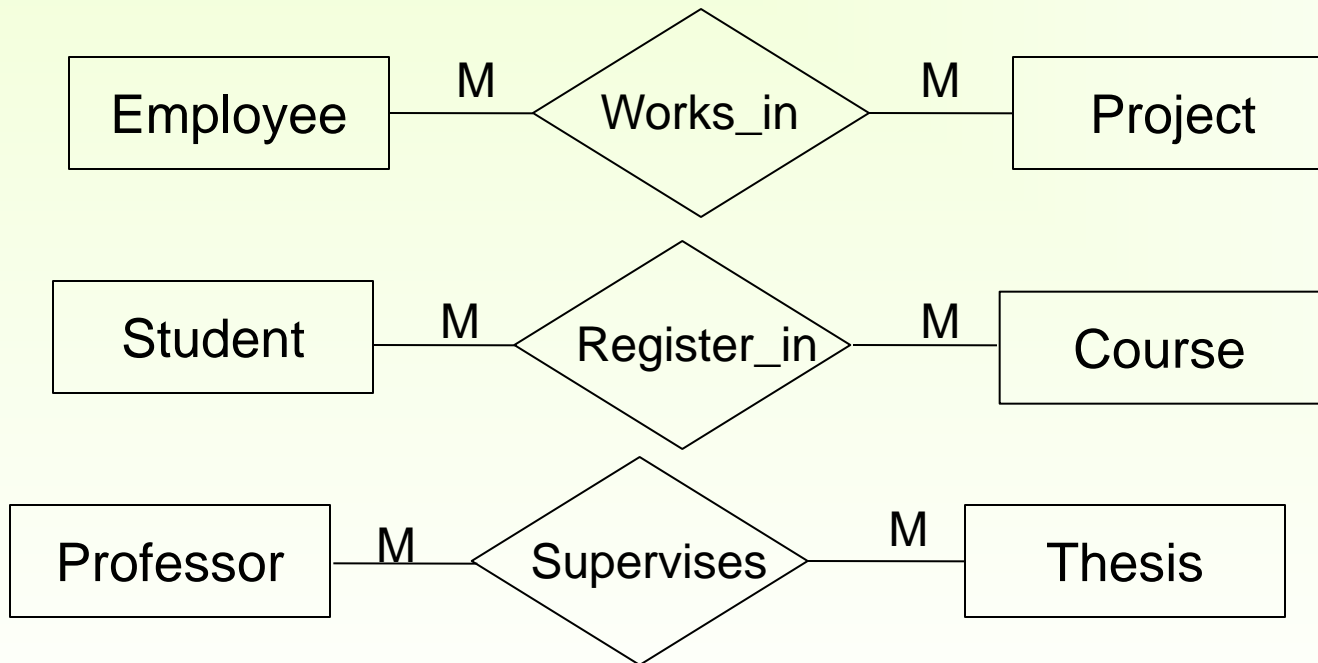


1:M Relationship

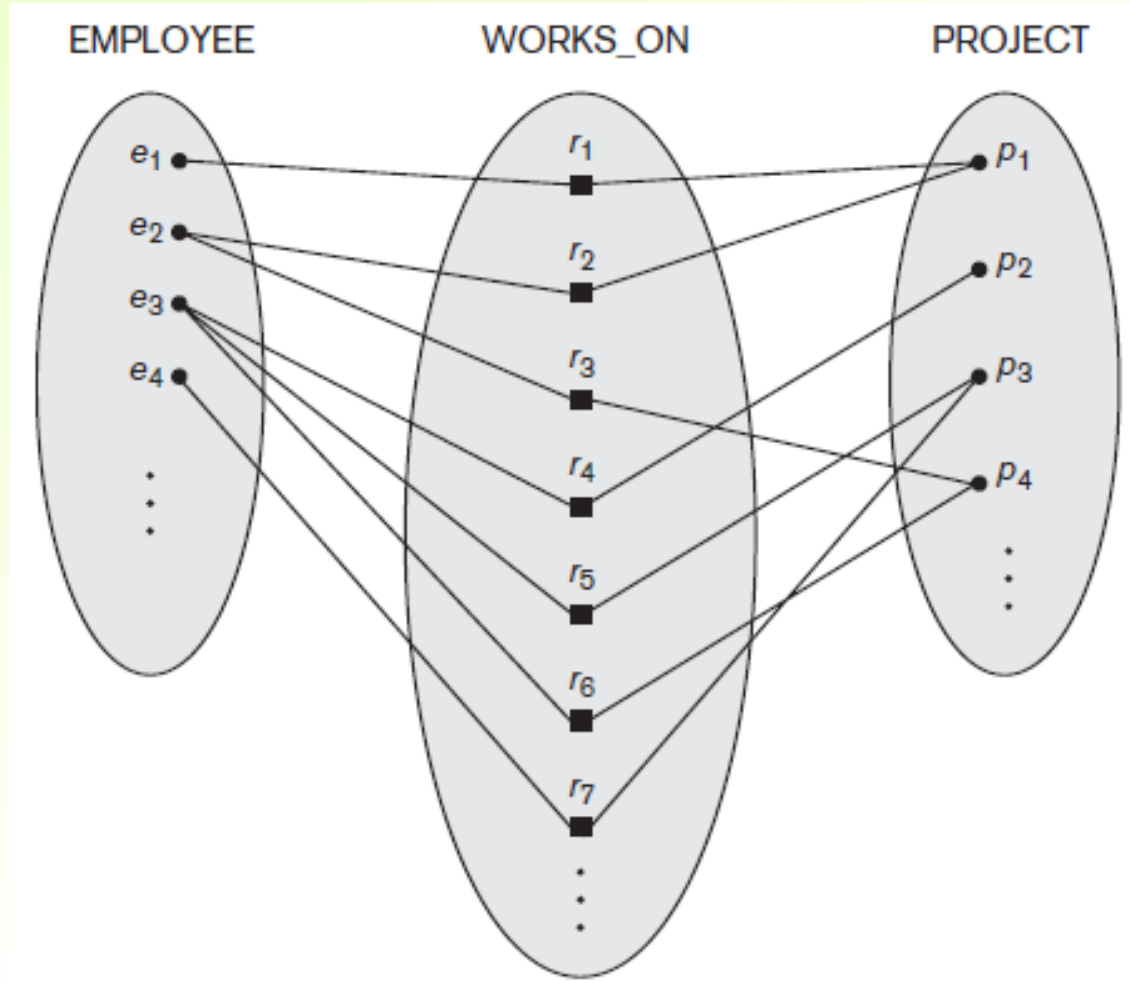


Cardinality Ratio (Cont.)

- M:M Relationship: Many entity instances in an entity are related to many entity instances in the other entity in the relationship

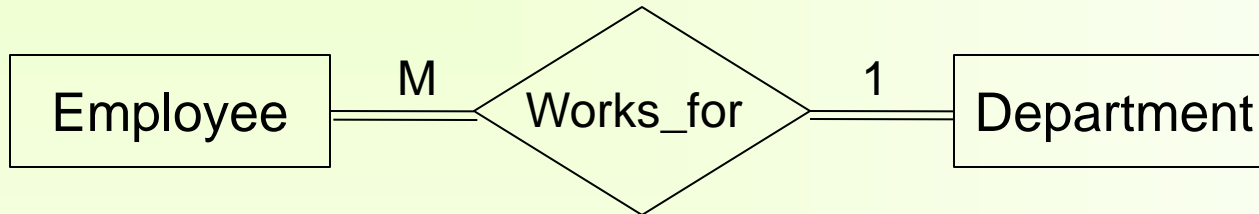


M:M Relationship

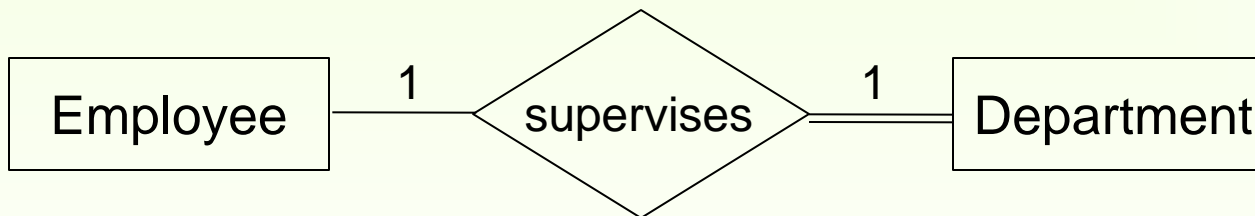


Total and Partial Participation

- Each Employee should work in a department, Each department should hire employees



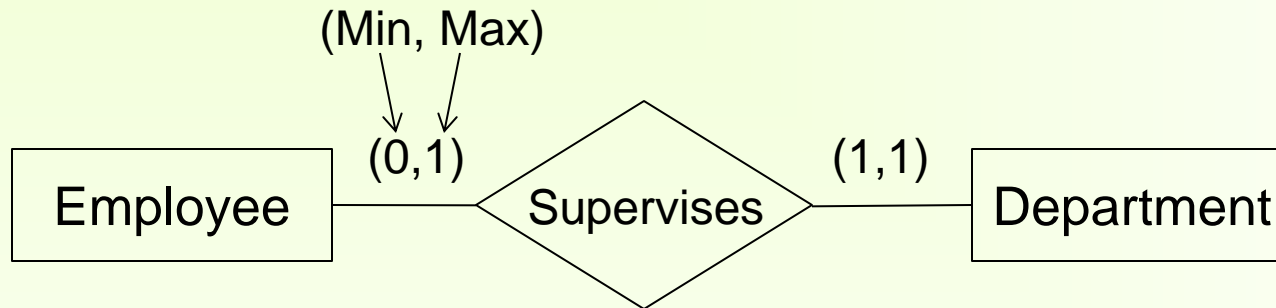
- Each department should have a manager, i.e. total participation. But not all employees are managers, i.e. partial participation



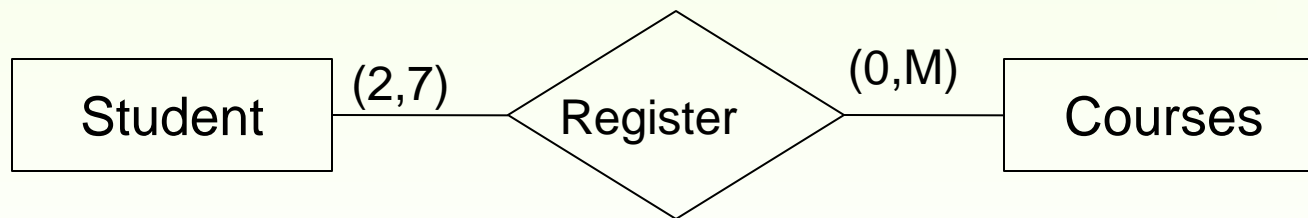
Alternative Notation for Cardinality

This notation involves associating a pair of integer numbers (min, max). This indicates that each entity object in entity E should appear minimum time and maximum time in relationship R.

Note: 0 means *partial participation*, 1 or more means *total participation*



Students should register at minimum 2 courses and at maximum 7 courses



ER Diagrams, Naming Conventions, and Design Issues












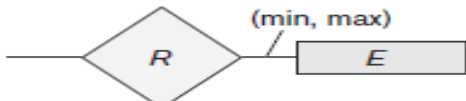
Symbol	Meaning
	Entity
	Weak Entity
	Relationship
	Identifying Relationship
	Attribute
	Key Attribute
	Multivalued Attribute
	Composite Attribute
	Derived Attribute
	Total Participation of E_2 in R
	Cardinality Ratio 1 : N for $E_1:E_2$ in R
	Structural Constraint (min, max) on Participation of E in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

Figure 7.14
Summary of the notation for ER diagrams.

Sample Database Application

- We would like to create a database for a company. The COMPANY database keeps track of a company's *employees*, *departments*, and *projects*.
- Suppose that after the requirements collection and analysis phase, the database designers provide the following constraints and business rules:

Business Rules for Company Database

- 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**. (1)
- A department may control several *projects*, each of which has a **unique name**, a **unique number**, and a **single location** (2)

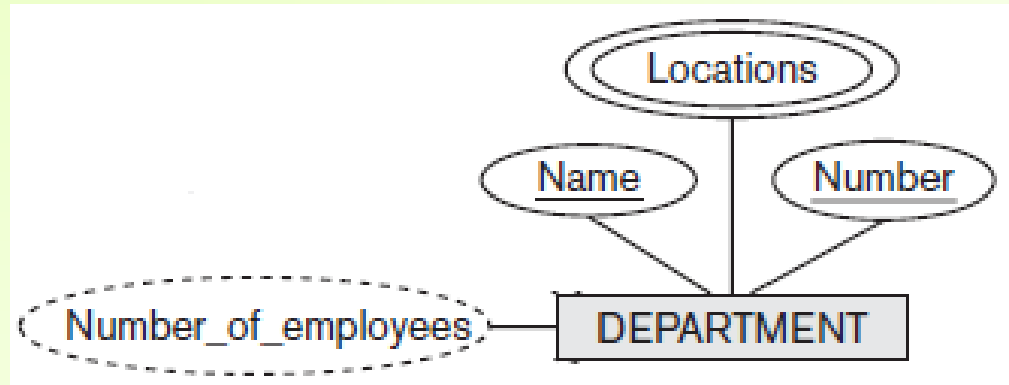
Business Rules for Company Database (Cont.)

- We store each employee's name, Social Security number, address, salary, sex(gender), 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 (who is another employee). (3)

Business Rules for Company Database (Cont.)

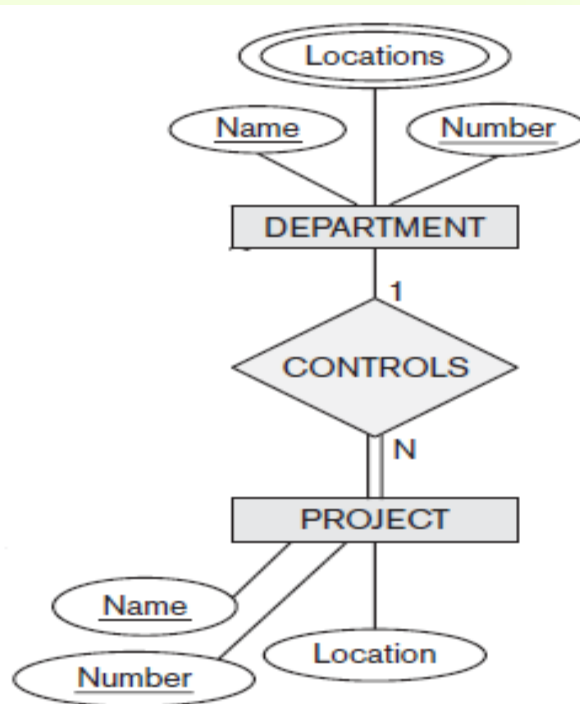
- We want to keep track of the dependents of each employee for insurance purposes. We keep each dependent's first name, sex, birth date, and relationship to the employee (4)

ER Schema Step by Step

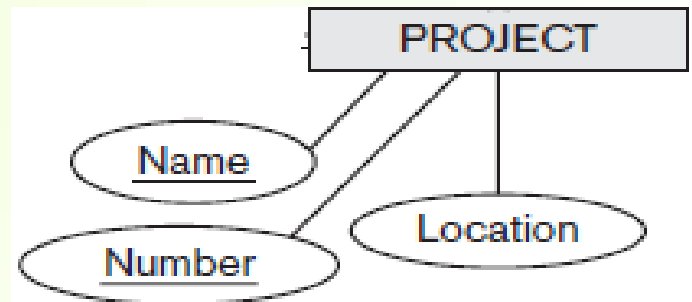


(1)

The relation
between
project and
department

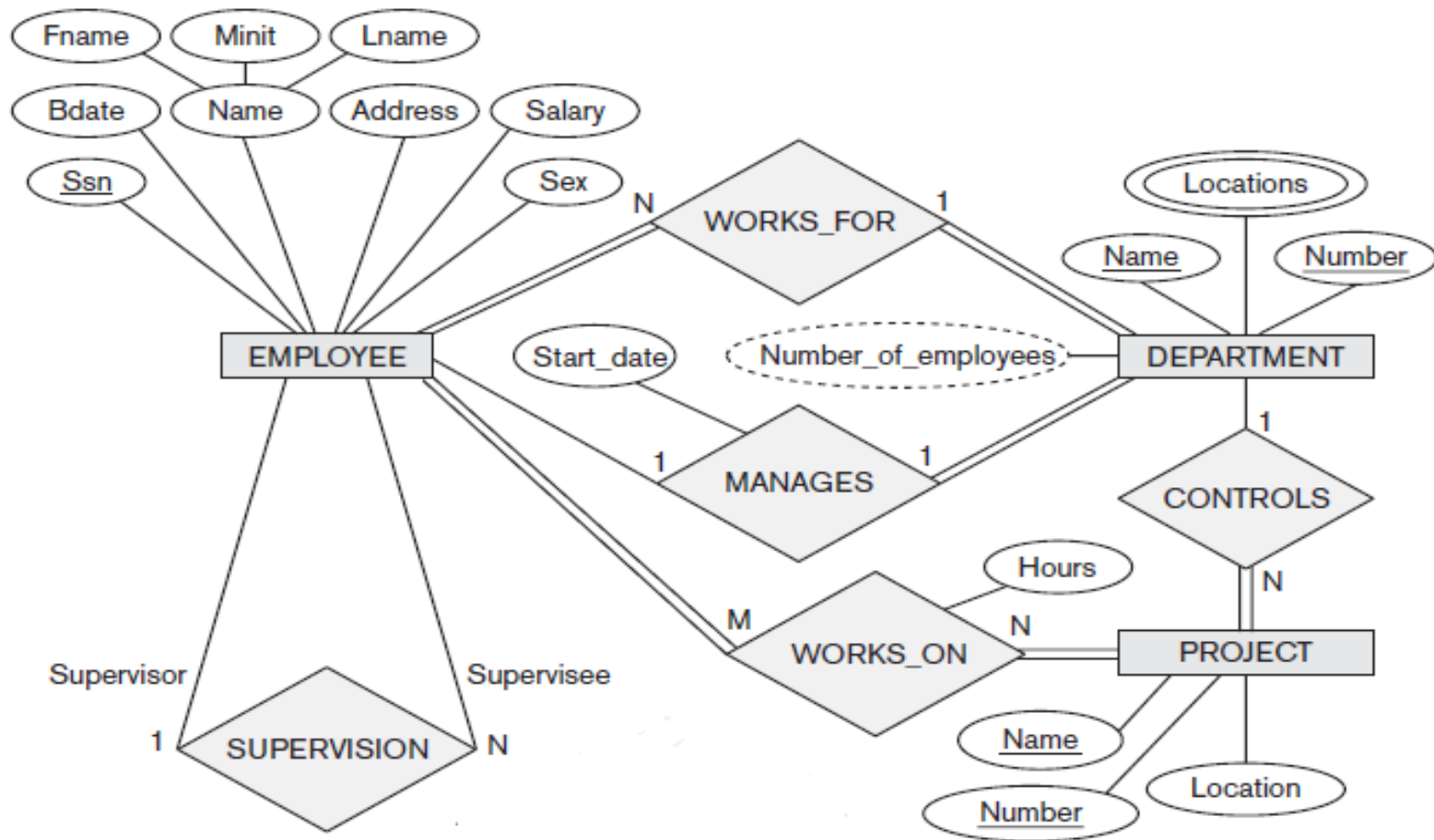


(2)



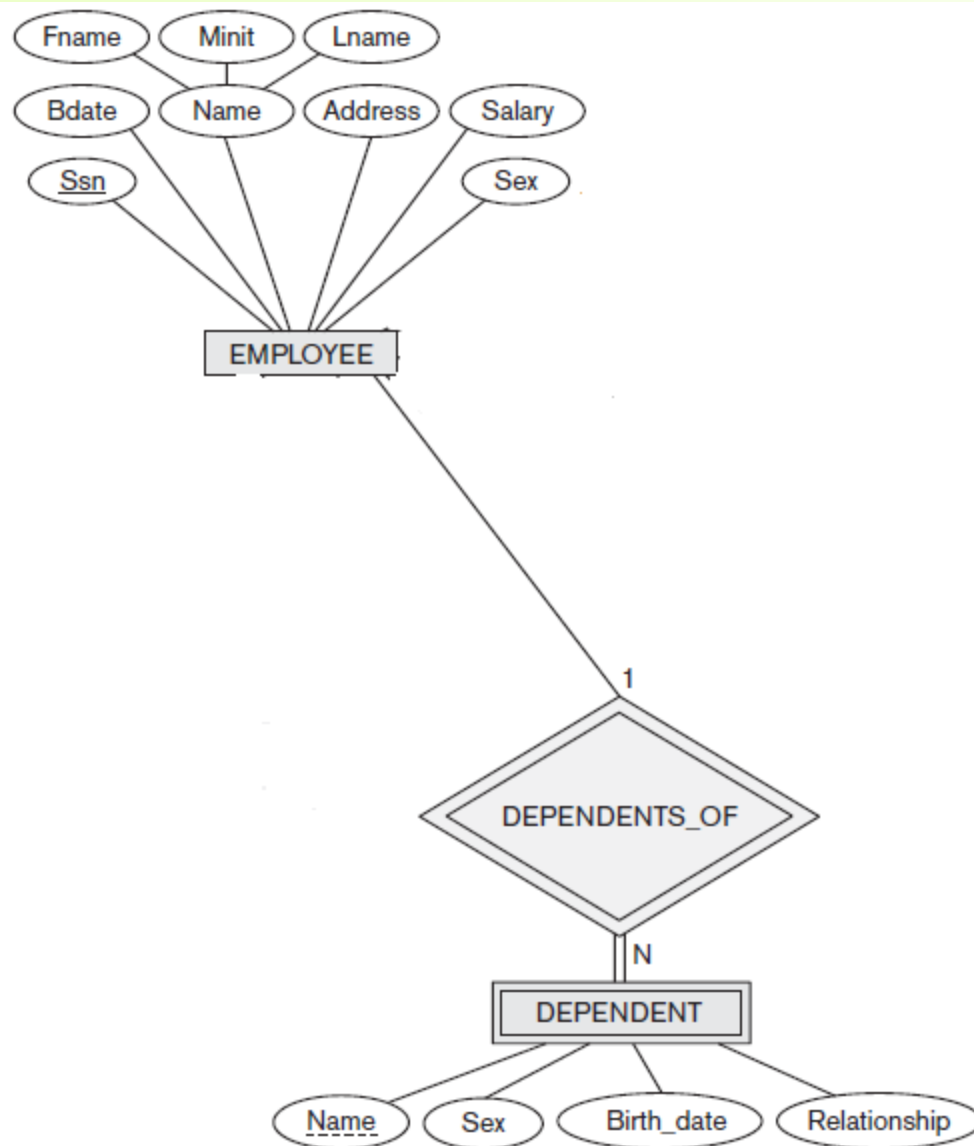
ER Schema Step by Step (Cont.)

(3)



ER Schema Step by Step (Cont.)

(4)



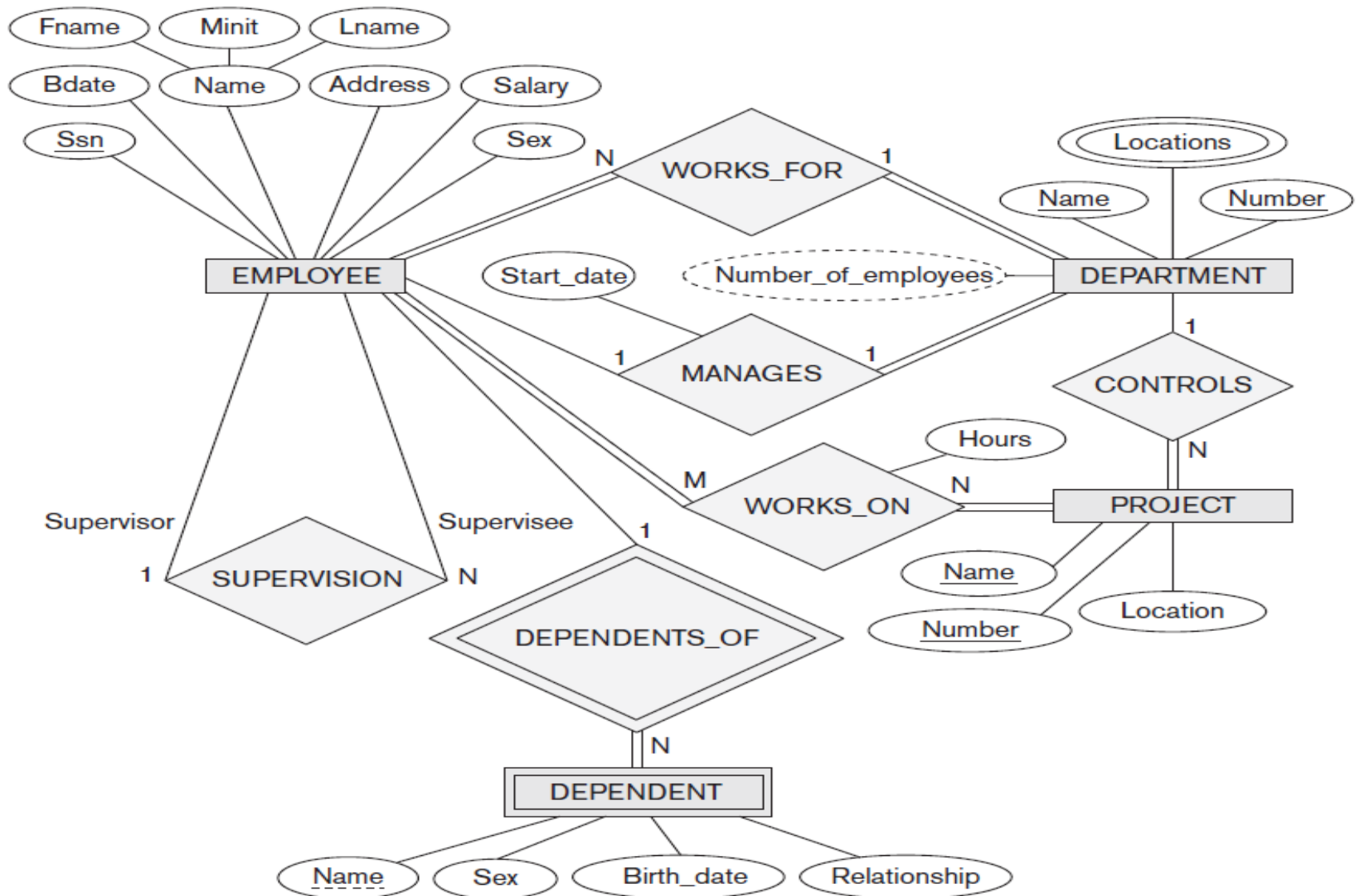
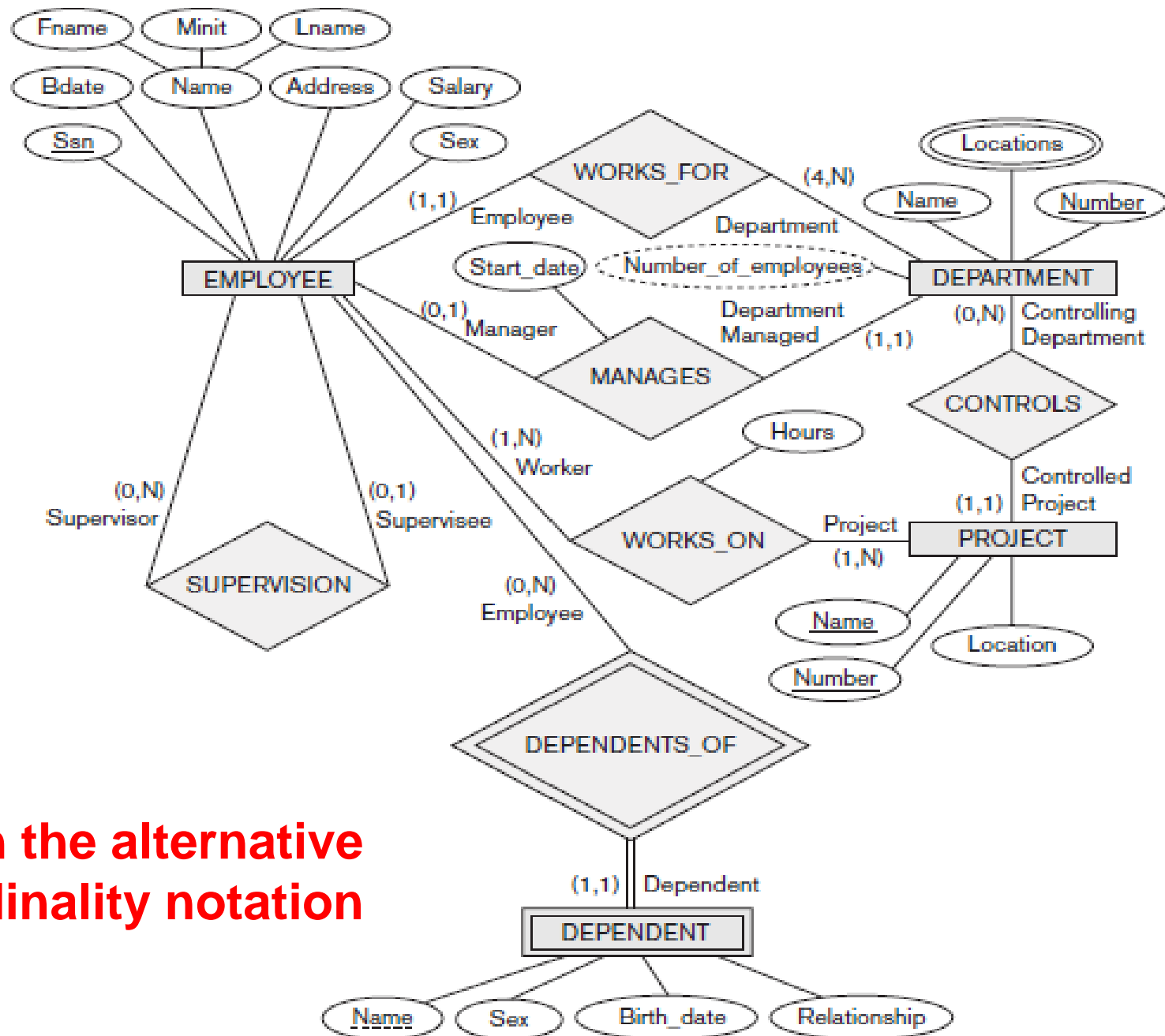


Figure 7.2

An ER schema diagram for the COMPANY database. The diagrammatic notation is introduced gradually throughout this chapter and is summarized in Figure 7.14.



With the alternative cardinality notation