

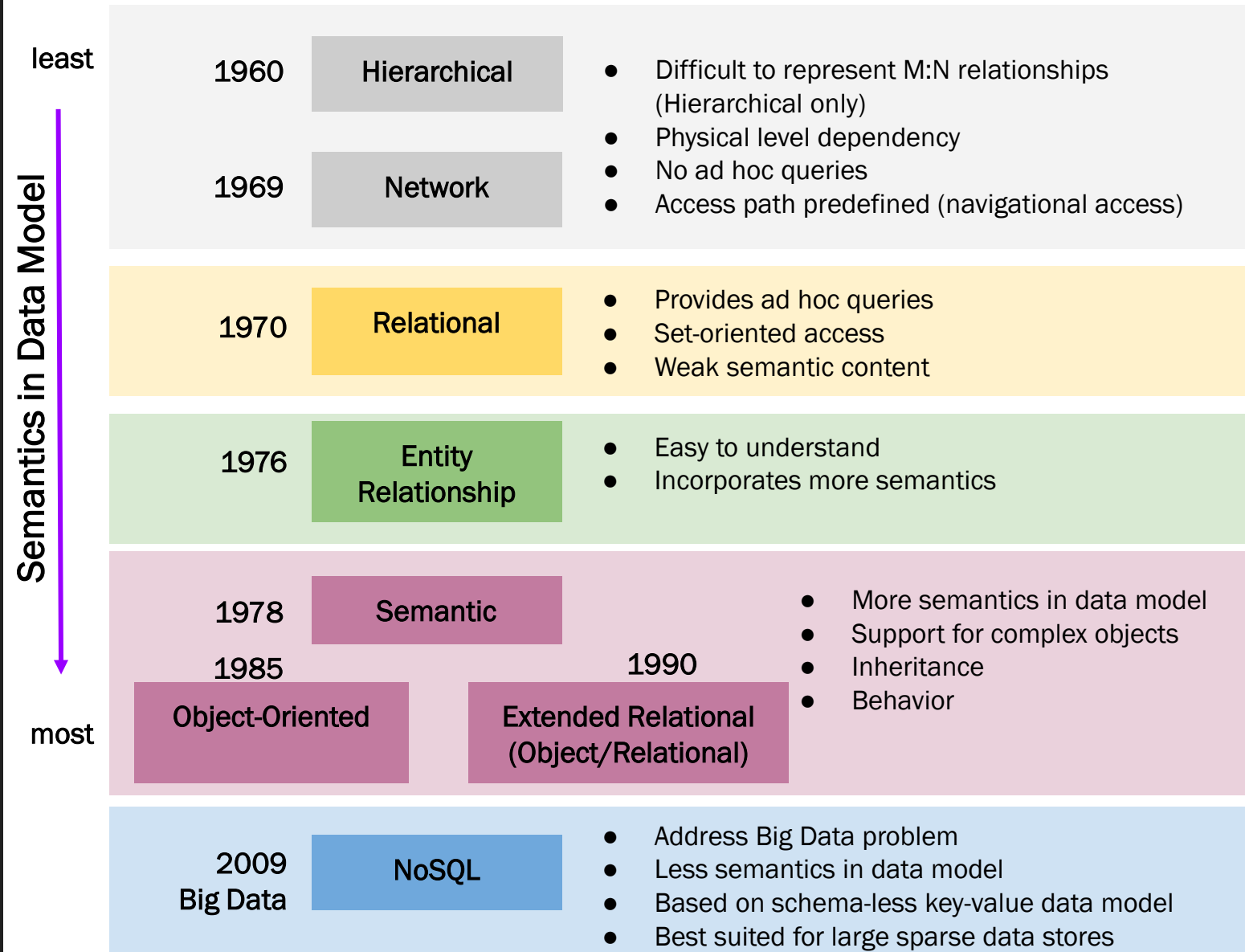
AT82.02

DATA MODELING AND MANAGEMENT

UNIT 1-4: ENTITY RELATIONSHIP (ER) MODEL

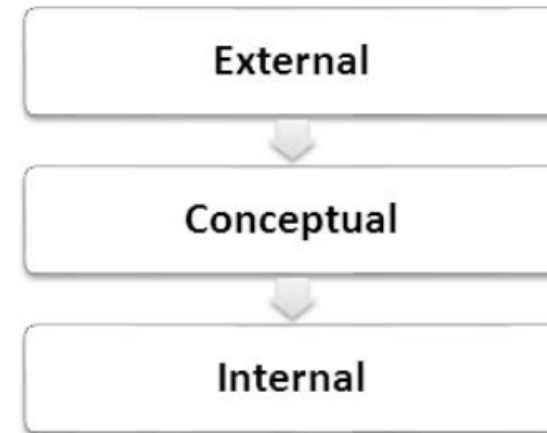
CHUTIPORN ANUTARIYA (CHUTI AT AIT DOT AC DOT TH)

Evolution of data models



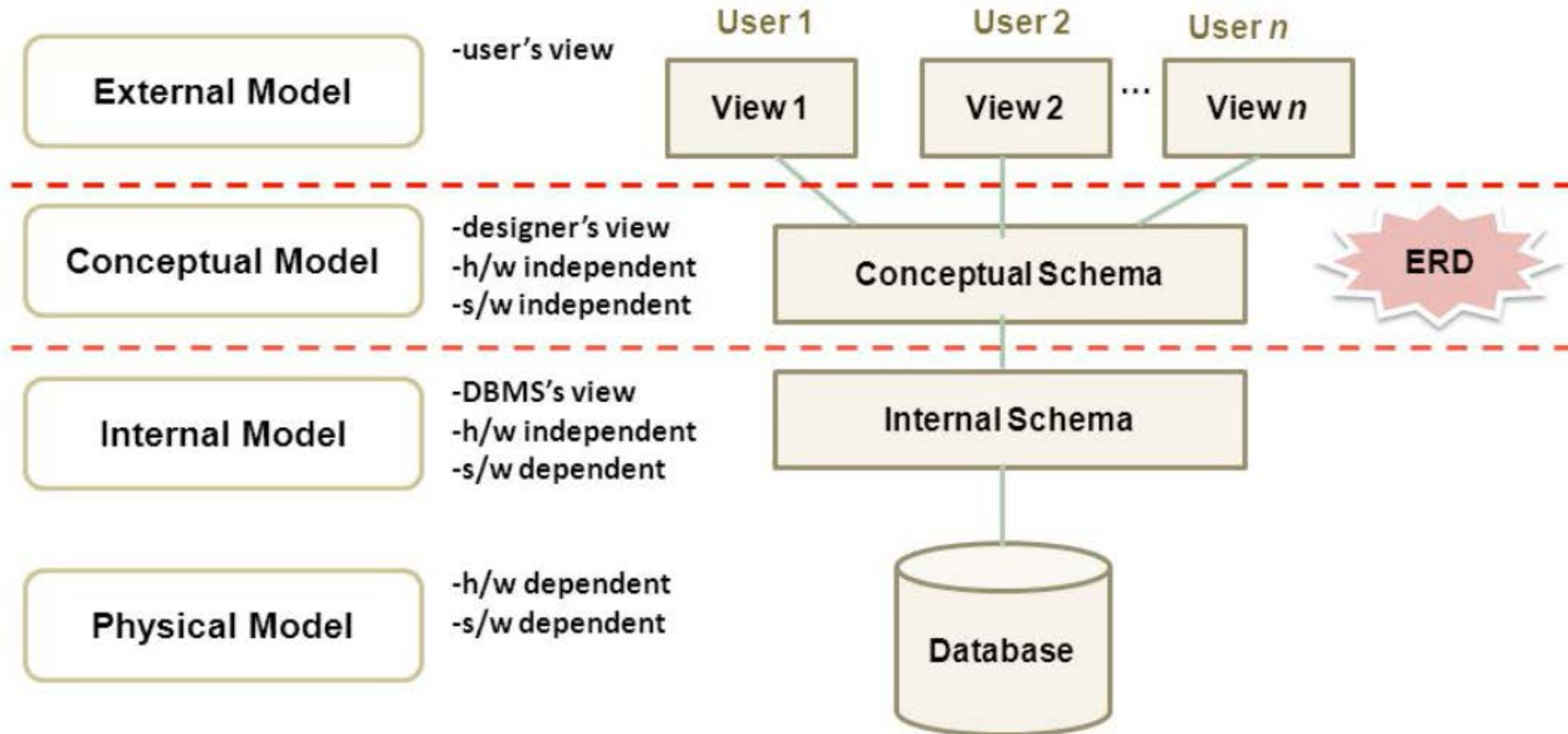
Degree of Data Abstraction

- American National Standards Institute (**ANSI**) Standards Planning and Requirements Committee (**SPARC**)
- **Three Level ANSI-SPARC Architecture**
 - Defined a framework for data modeling based on degrees of data abstraction(1970s):



Data Abstraction

Three Level ANSI-SPARC Architecture



Data Abstraction Level

Model	Degree of Abstraction	Focus	Independent of
External	High	End-user views	HW and SW
Conceptual		Global view of data (database model independent)	HW and SW
Logical		Specific database model	HW
Physical	Low	Storage and access methods	Neither HW nor SW

Main Phases of Database Design

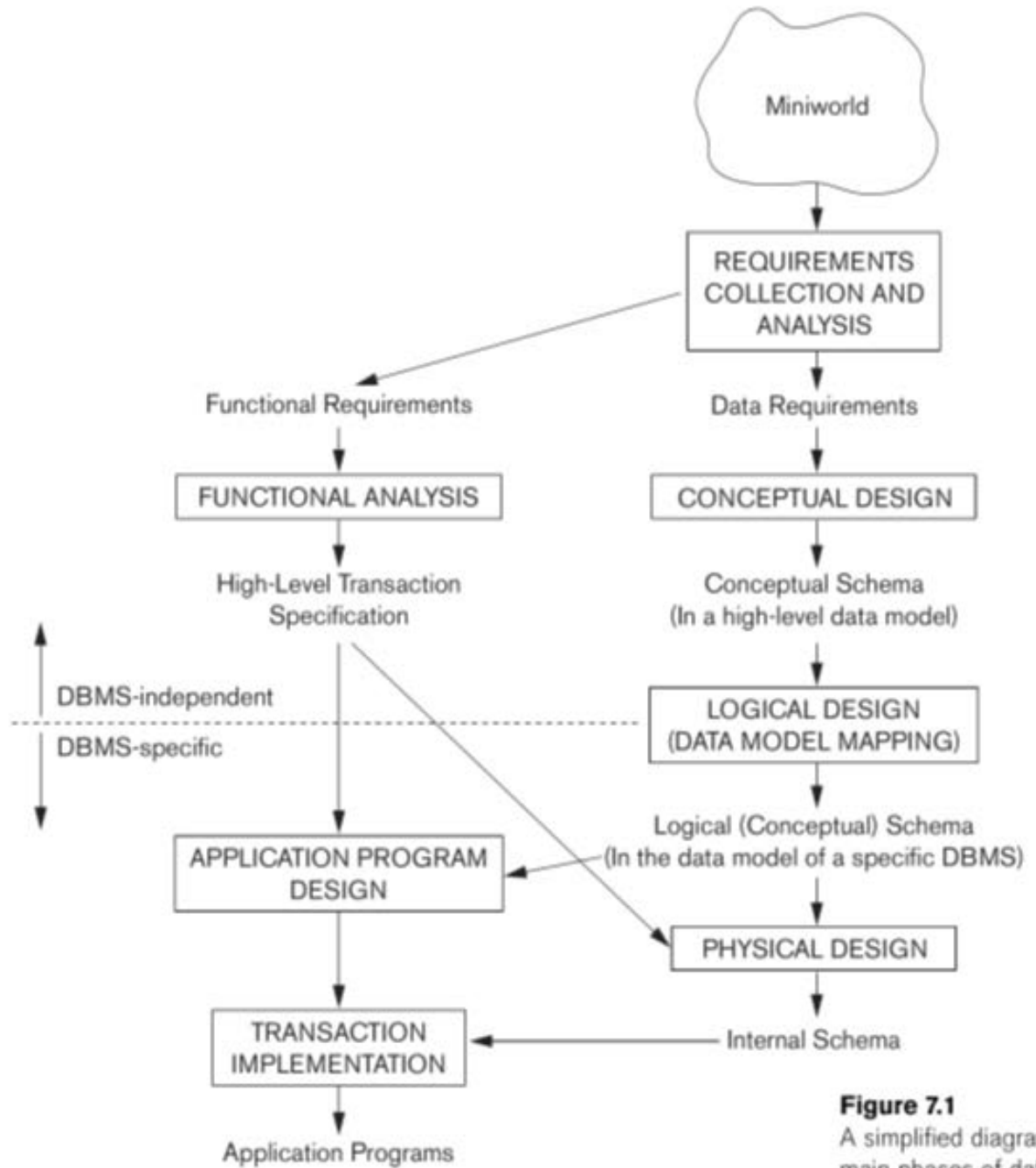
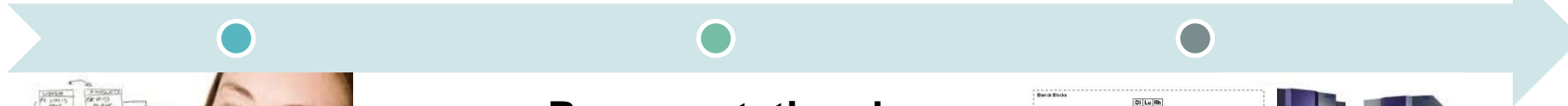


Figure 7.1
A simplified diagram to illustrate the main phases of database design.

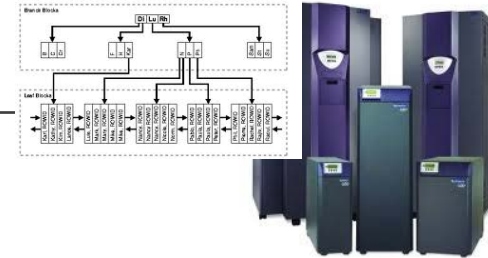
Using High-Level Conceptual Data Models for Database Design

**[High-level]
Conceptual Data
Models**

**[Low-level] Physical
Data Models**



**Representational
Data Models**



Requirements Collection and Analysis

Database designers interview prospective database users to understand and document data requirements

Result: data requirements

Functional requirements of the application

Conceptual Design

Conceptual schema

Description of data requirements

Includes detailed descriptions of the entity types, relationships, and constraints

Transformed from high-level data model into implementation data model

Logical Design or Data Model Mapping

Result is a database
schema in implementation
data model of DBMS

Physical Design

Internal storage structures, file organizations, indexes, access paths, and physical design parameters for the database files specified

Entity-Relationship Model

A Sample Database Application

COMPANY

- Employees, departments, and projects
- Company is organized into departments
- Department controls a number of projects
- Employee: store each employee's name, Social Security number, address, salary, sex (gender), and birth date
- Keep track of the dependents of each employee

Company Database

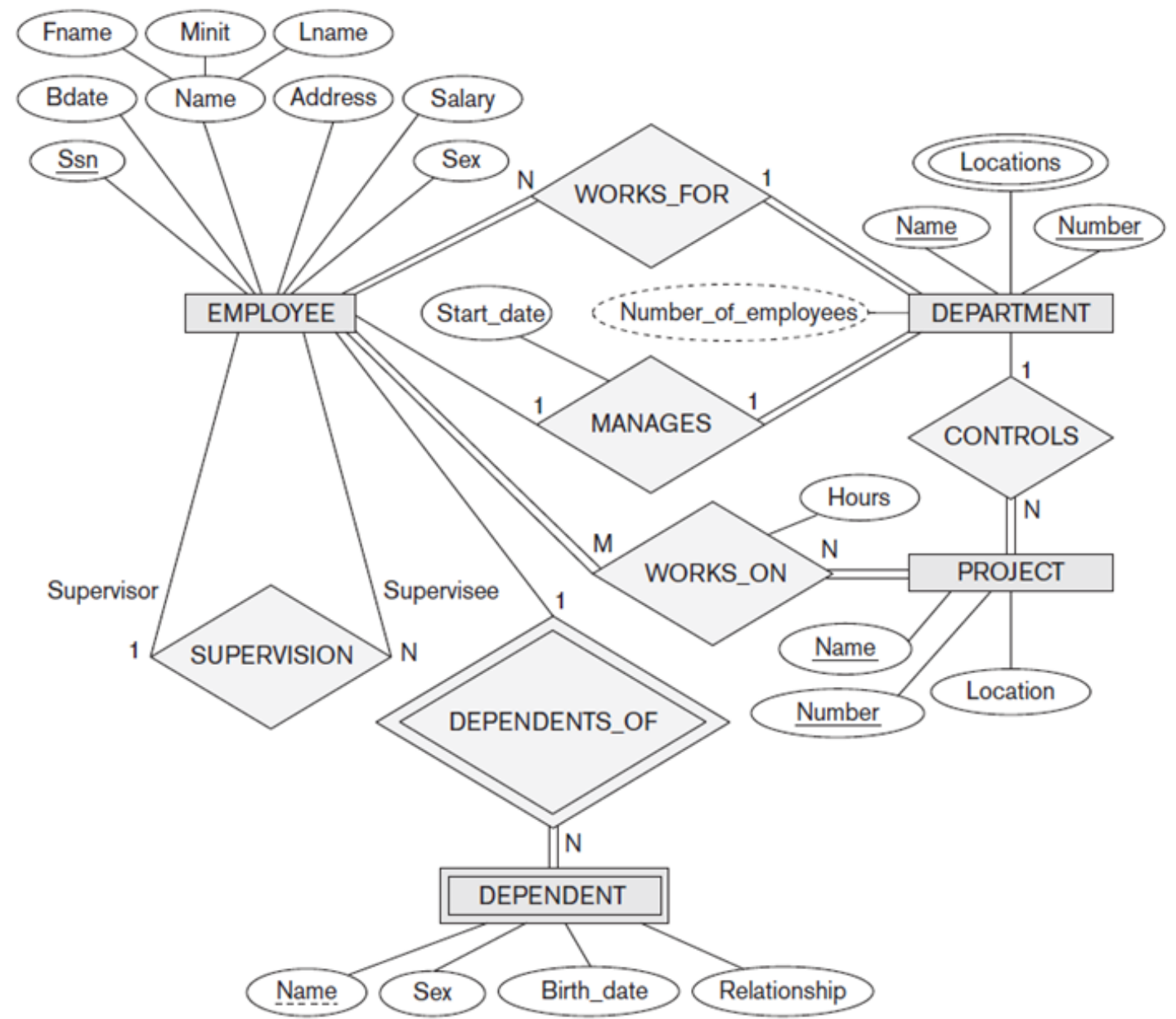


Figure 7.2

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

ER Model

ER model describes data as:

- Entities
- Relationships
- Attributes

Entities

An **entity** is a representation of a class of object.

It can be a person, place, thing, etc. Entities usually have attributes that describe them.

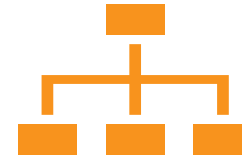
Attributes



Attribute:

An **attribute** is a property that describes a particular entity.

The attribute(s) that uniquely distinguishes an instance of the entity is the **identifier**.



Types of attributes:

Composite versus **simple (atomic)** attributes

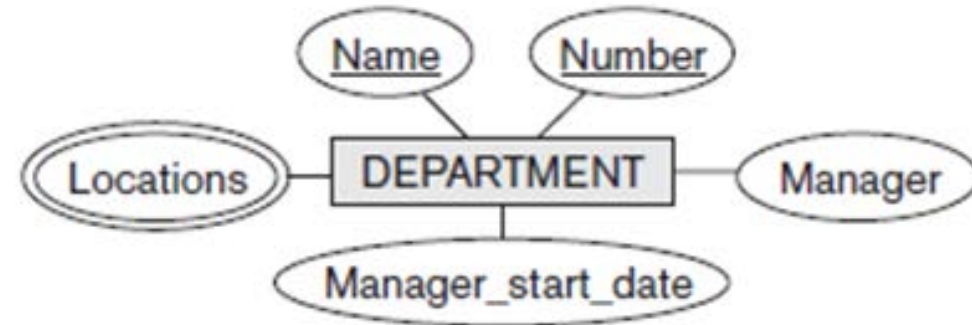
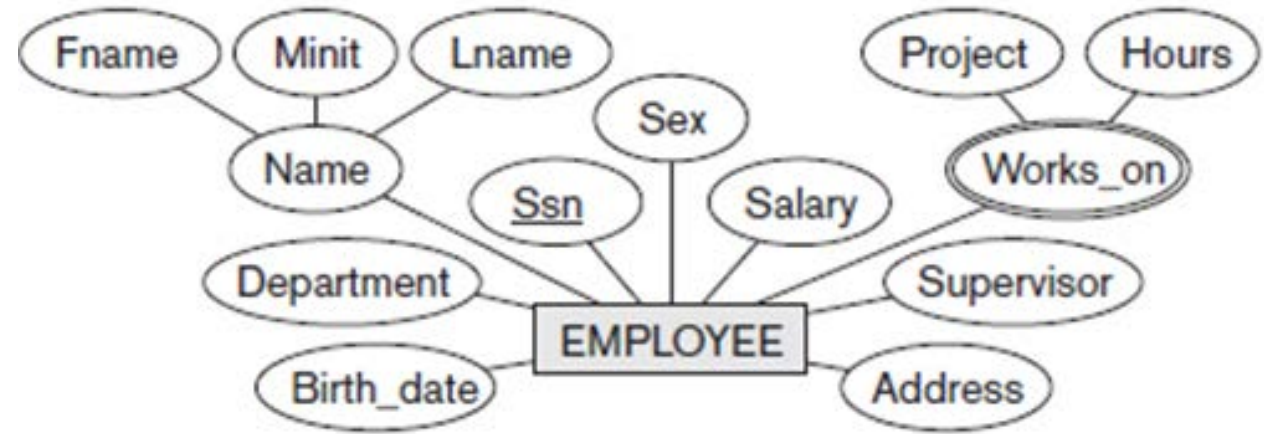
Single-valued versus **multivalued** attributes

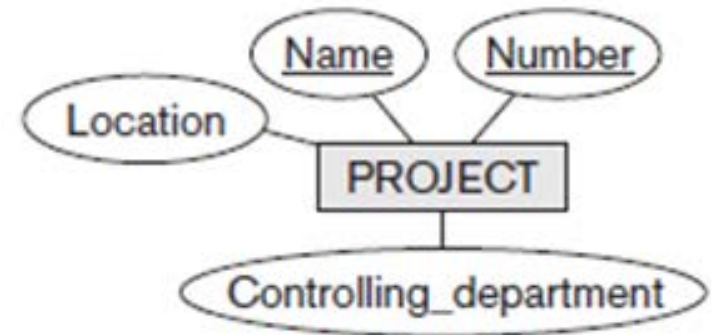
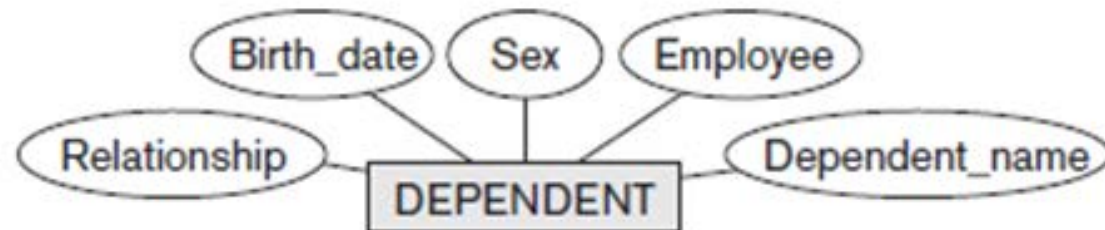
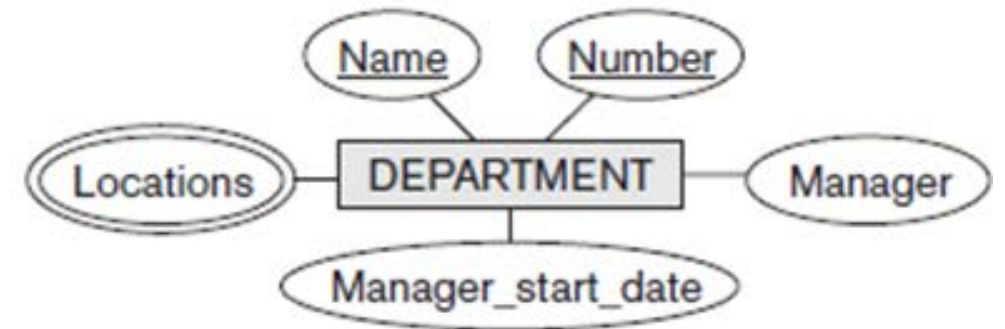
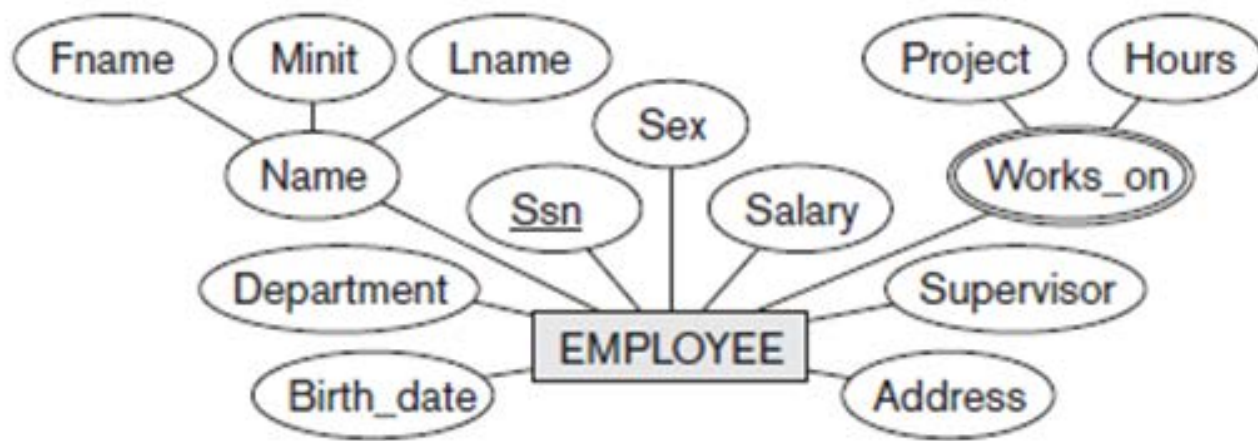
Stored versus **derived** attributes

NULL values

Complex attributes

Composite & Multivalued Attributes





Initial Conceptual Design COMPANY Database

Relationships



When an attribute of one entity type refers to another entity type



Represent references as relationships not attributes



Relationships illustrate the association between two entities.



Usually, each relationship has a name, expressed as a verb. This describes what kind of relationship connects the objects.

Role Names and Recursive Relationships

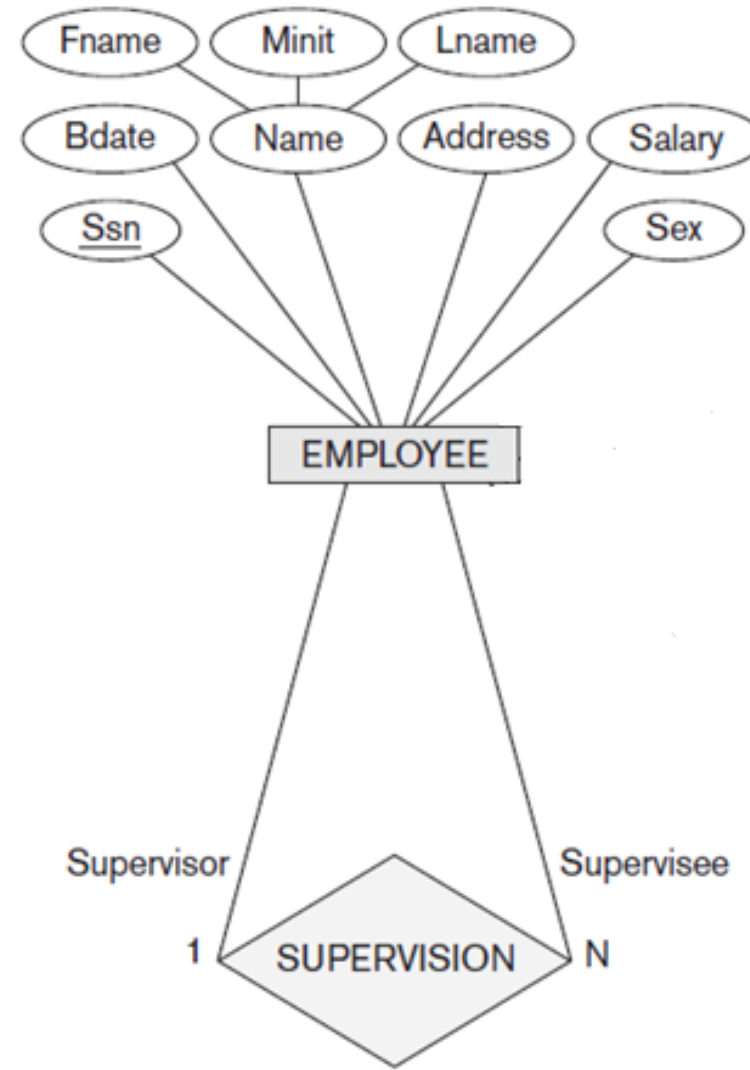
Role names and recursive relationships

- Role name signifies role that a participating entity plays in each relationship instance

Recursive relationships

- Same entity type participates more than once in a relationship type in different roles
- Must specify role name

Example: Recursive Relationship & Role



Cardinality/Multiplicity

Multiplicity refers to the **maximum** number of times that an instance of one entity can be associated with instances in the related entity. It can be one or many.



One to one (1:1)



One to many (1:N)

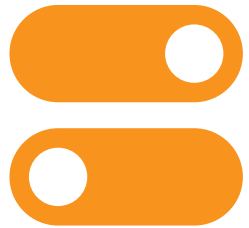


Many to one (N:1)



Many to many (M:N)

Cardinality / Multiplicity

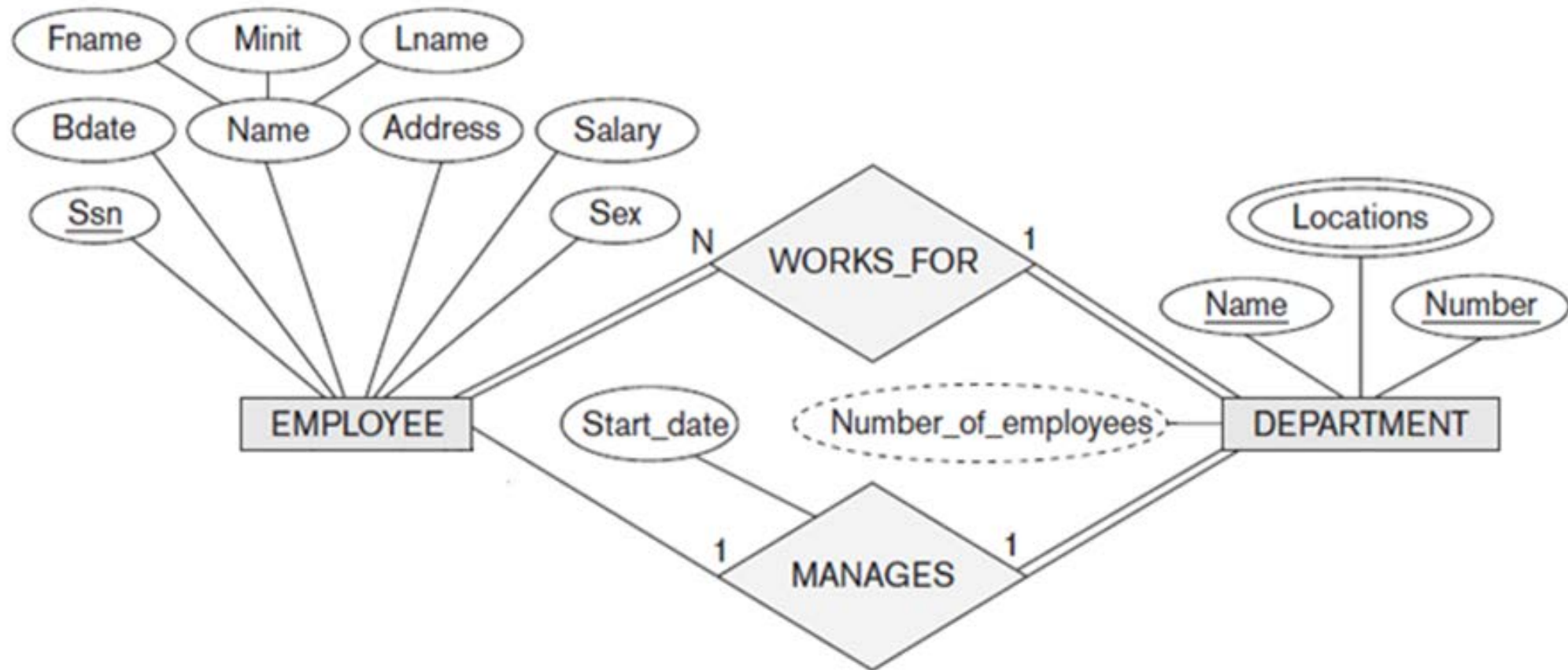


The **minimum** number of times one instance can be related to others. It can be zero or one, and accordingly describes the relationship as **optional** or **mandatory**.

Mandatory vs Optional

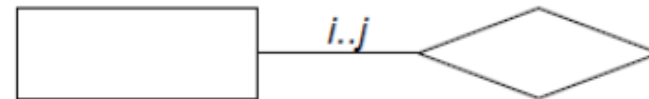
- **Participation constraint**
 - Specifies whether existence of entity depends on its being related to another entity
 - Types: **total** and **partial**
- **Total Participation:** every entity in the entity set participates in at least one relationship in the relationship set
- **Partial Participation:** some entities may not participate in any relationship in the relationship set

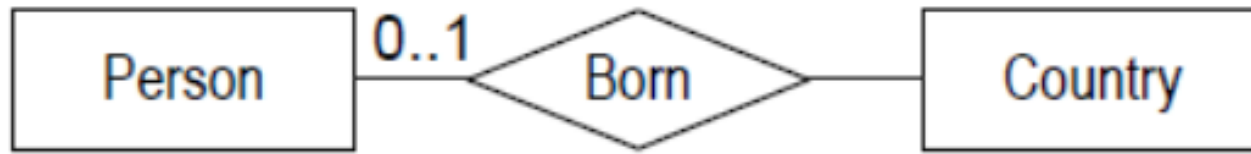
Ex: Cardinality & Participation Constraints



Alternate Notation

- We can specify how many times each entity from some entity set can participate in some relationship, in every instance of the database
- In general we can say that
 - » This number is in the interval $[i, j]$, $0 \leq i \leq j$, with i and j integers, denoted by $i..j$; or
 - » This number is at the interval $[i, \infty)$, denoted by $i..*$
- $0..*$ means no constraint
- No constraint can also be indicated by not writing out anything

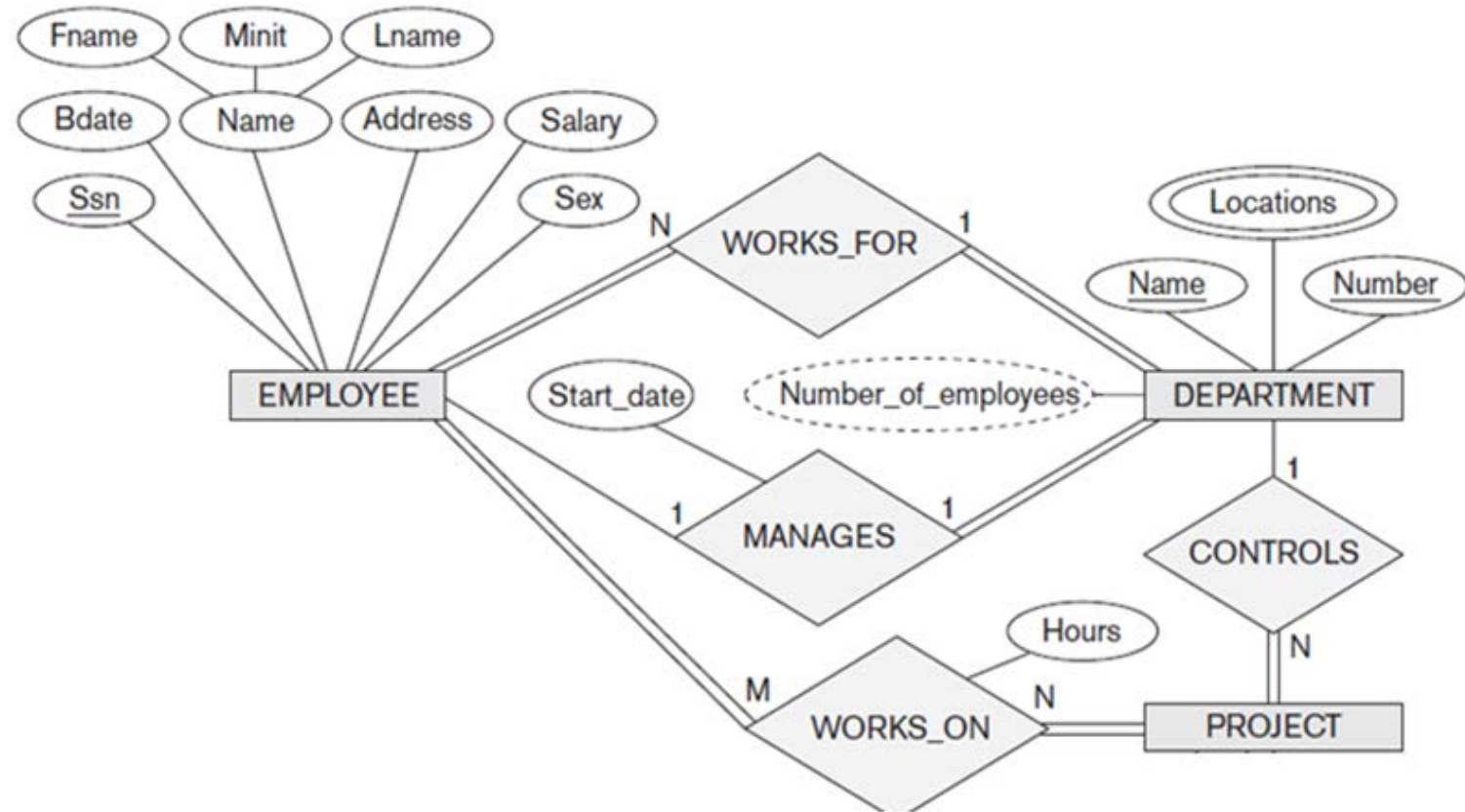




Ex: Alternate Notation

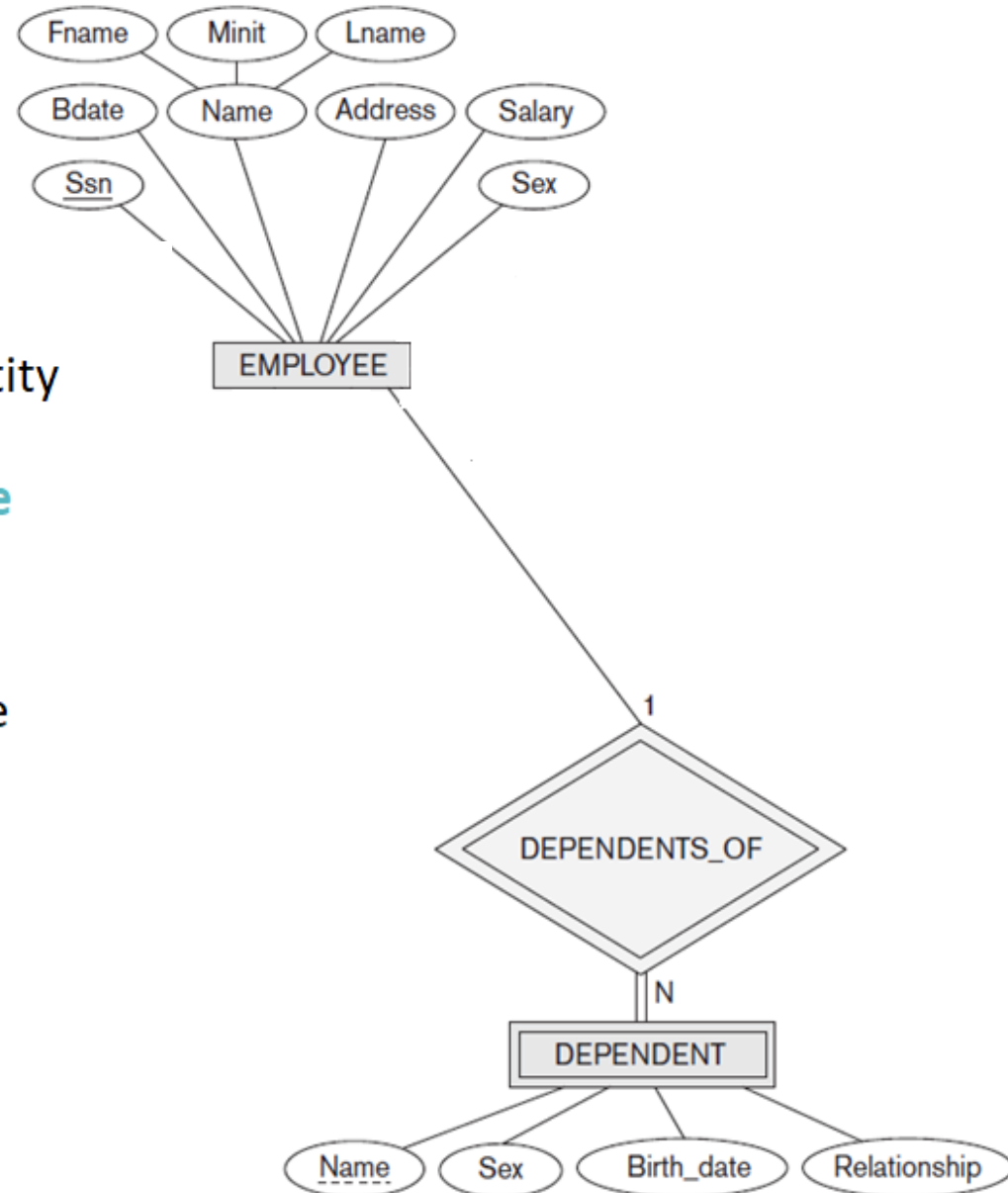
Attributes of Relationship

Relationship can also have attributes



Weak Entity

- Do not have key attributes of their own
 - Identified by being related to specific entities from another entity type.
 - This other entity type is called **identifying** or **owner entity type**
- **Identifying relationship**
 - Relates a weak entity type to its owner
 - We put the identifying relationship of a weak entity in a double diamond.
- Normally has a partial key (or discriminator), which is the attribute that can uniquely identify weak entities that are related to the same owner entity. We underline the discriminator of a weak entity set with a dashed line.
- Always has a total participation constraint



Refining the ER Design for the COMPANY Database



Change attributes that represent relationships into relationship types



Determine cardinality ratio and participation constraint of each relationship type

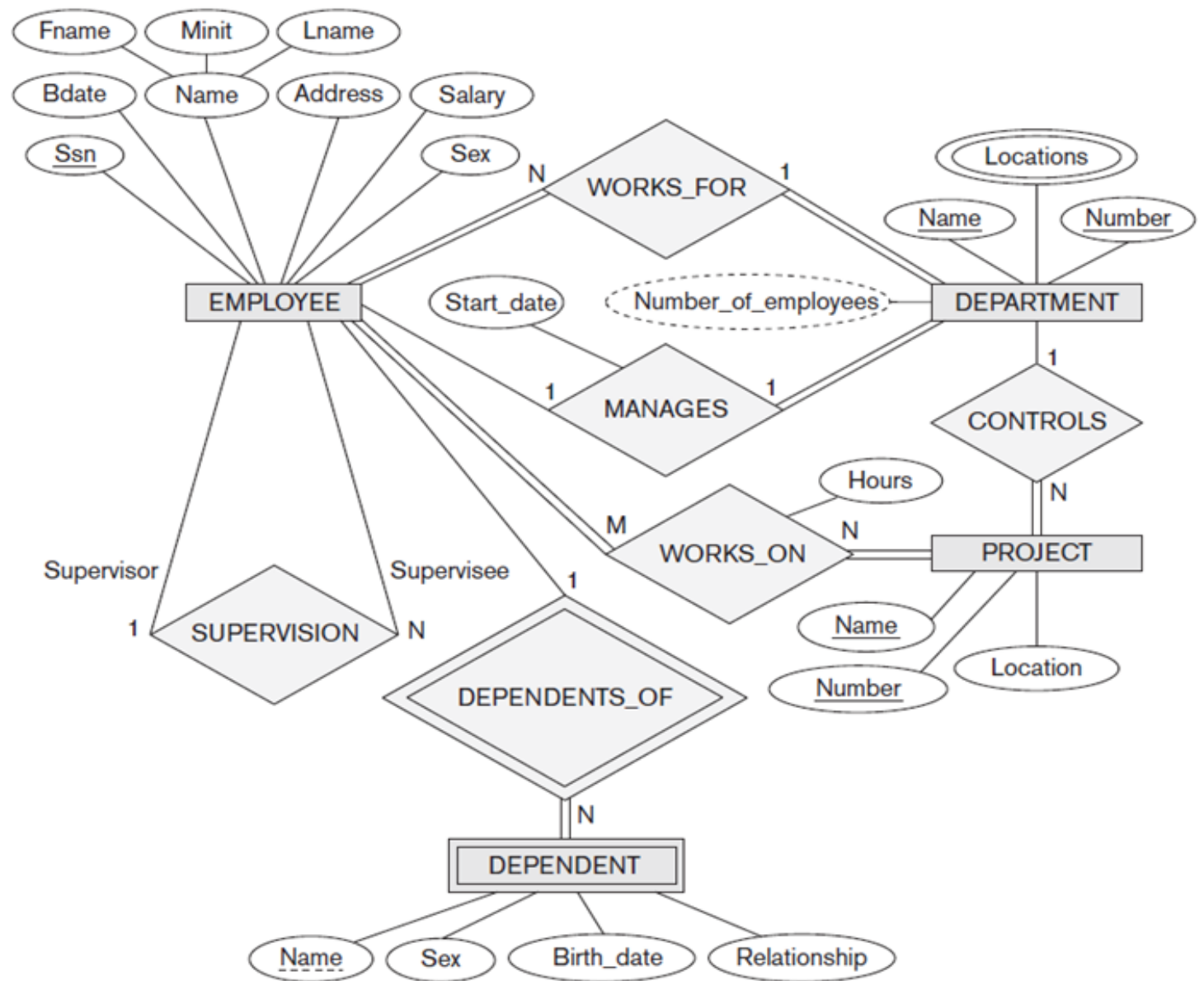


Figure 7.2

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

Relationship Types of Degree Higher than Two

Degree of a relationship type

- Number of participating entity types

Binary

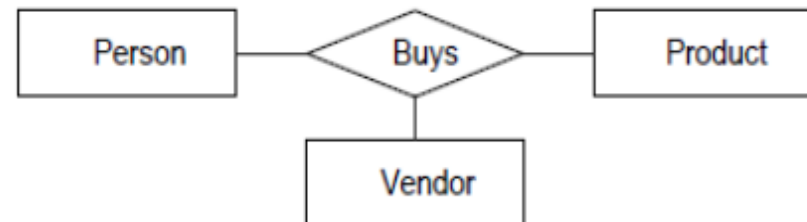
- Relationship type of degree two

Ternary

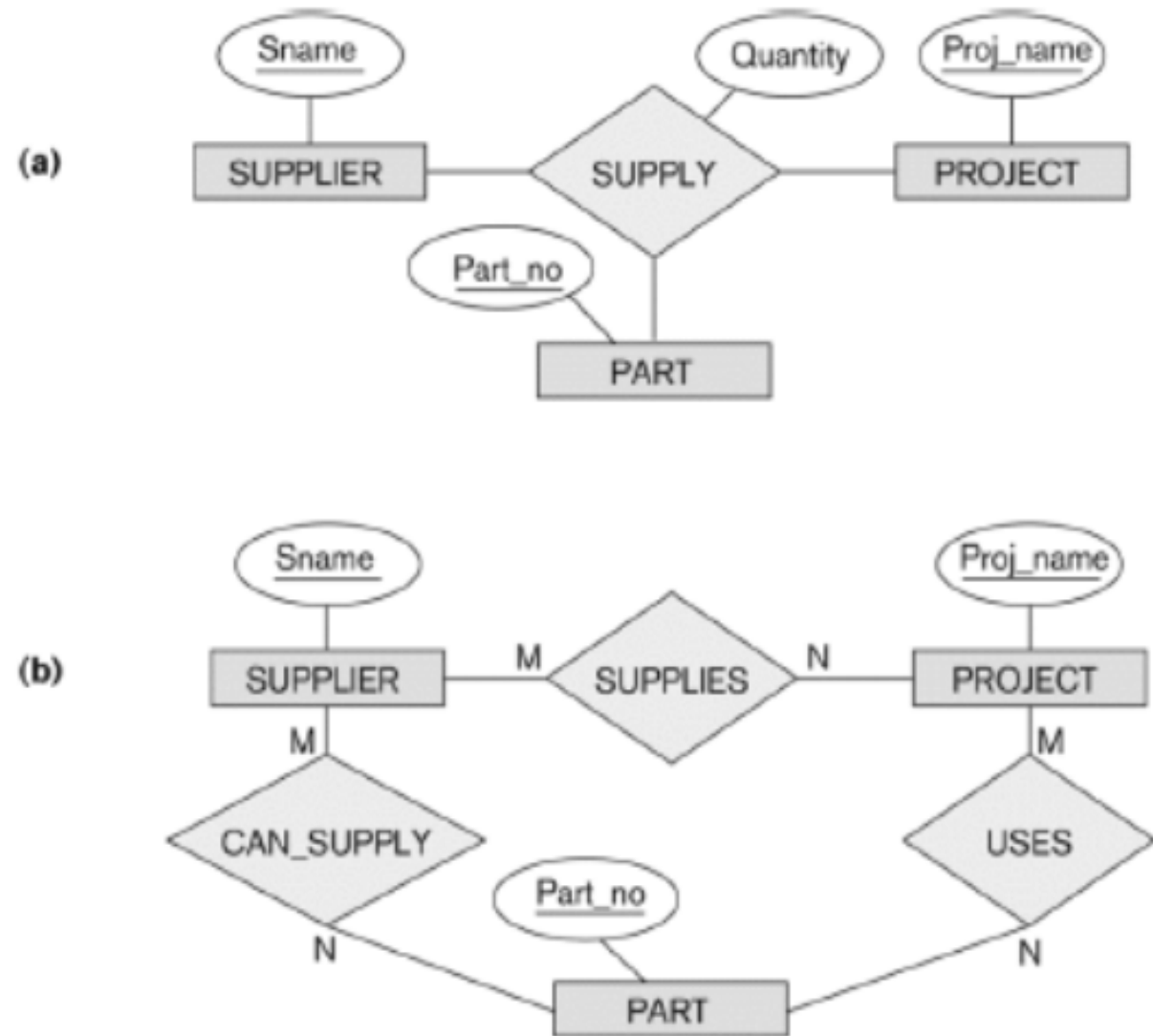
- Relationship type of degree three

Ex: Ternary Relationship

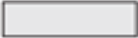











- Let's look at Buys listing all tuples of (x,y,z) where person x Buys product y from vendor z
- Let us just state it informally:
 - » Joe buys computer from IBM
 - » Joe buys computer from Dell
 - » Tom buys computer from Dell
 - » Tom buys monitor from Apple
 - » Joe buys monitor from IBM
 - » Max buys computer from IBM
 - » Max buys monitor from Dell



Ex: Ternary Relationships



Model Notation Summary

Symbol	Meaning	Figure 7.14 Summary of the notation for ER diagrams.
	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	

ER Modeling Practices

Practice Time: Motor Vehicle Insurance Policy Management

An insurance company writes policies for drivers. One policy can cover many drivers and also many vehicles, but a vehicle can be related to only one policy. Drivers can share one or more vehicles (e.g. a husband and wife own one vehicle and they both drive the same vehicle or a family can have multiple vehicles).

The company gets a master list of violations from the Department of Motor Vehicles. These violations are then input into the system and used to determine the price of the policy. A driver may commit more than one violation. One or more drivers can commit the same violation. The system should keep a track of all customers - active (with insurance) and inactive (held in an archive – for cancelled customers). All customers should be able to get a quote, insurance or cancel the insurance.

Practice Time: Fix It

The repair company “FixIT” has hired you to design a database system to keep track of its operations. FixIT gives services to repair any IT computing devices such as computer desktops, laptops and smartphones of any models and makes. The system should keep track of the repair jobs, the items (or parts) used for each repair job, the labor costs for each repair job, the repairmen performing each repair job, and the total cost of each repair job.

When a customer bring a device in to be repaired, he/she describes the device’s problem, makes a deposit on the repair job, and is given an estimated date to return and uplift their devices. Repairmen then perform repairs based on the repair job, and detail the labor costs and the items (parts) used for each repair job.

Each customer may check the status and details of his/her repair job from the online system by entering the job ID or customer name. When a customer returns, he/she pays the total cost of the repair job less the deposit, collects a receipt for the payment, and uplifts the repaired device.

ER Diagrams, Naming Conventions, and Design Issues

Proper Naming of Schema Constructs

Choose names that convey meanings attached to different constructs in schema

Nouns give rise to entity type names

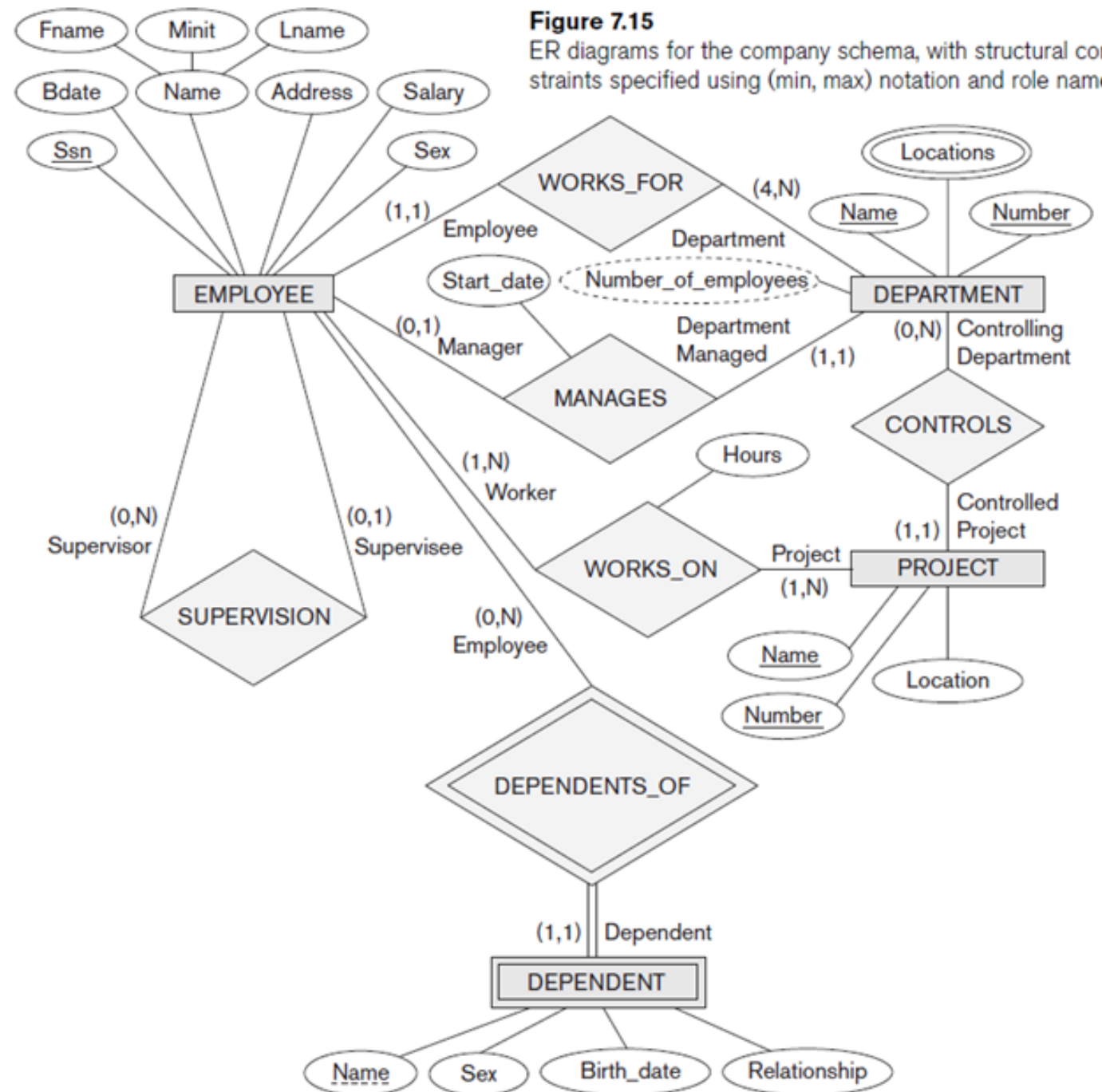
Verbs indicate names of relationship types

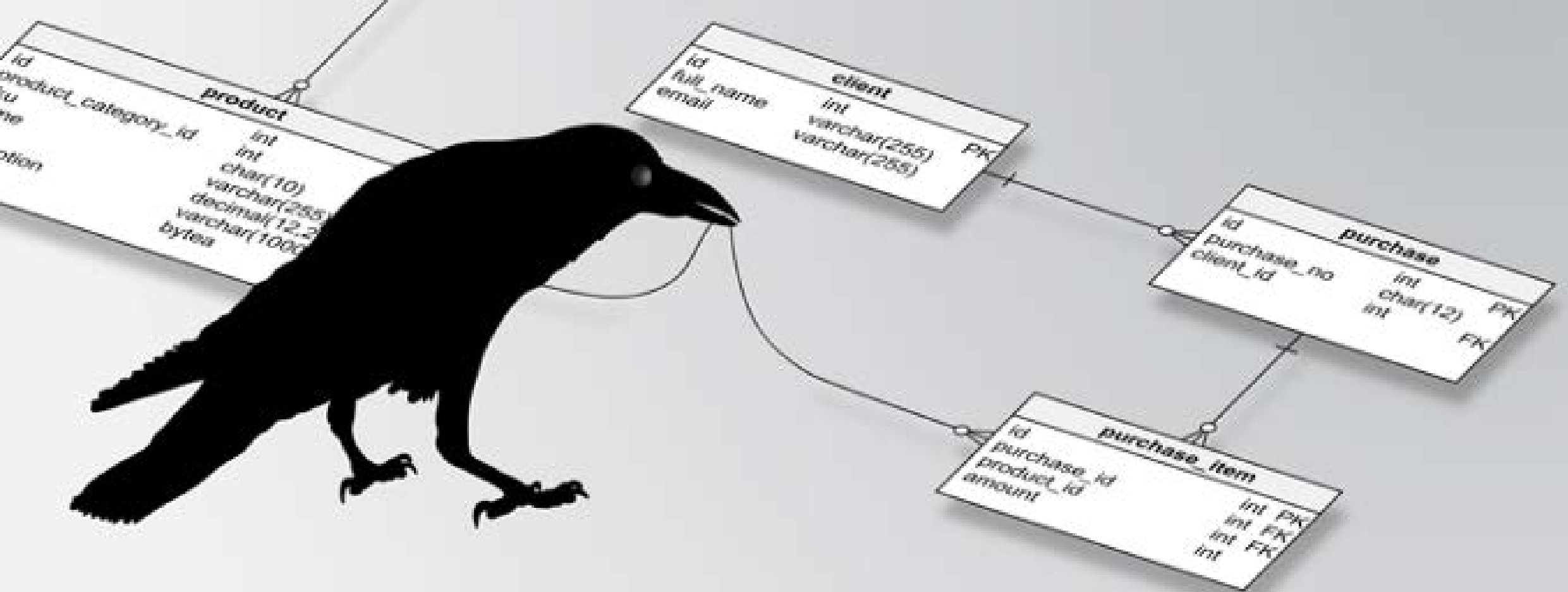
Choose binary relationship names to make ER diagram readable from left to right and from top to bottom

Alternative Notations for ER Diagrams

Specify structural constraints on relationships

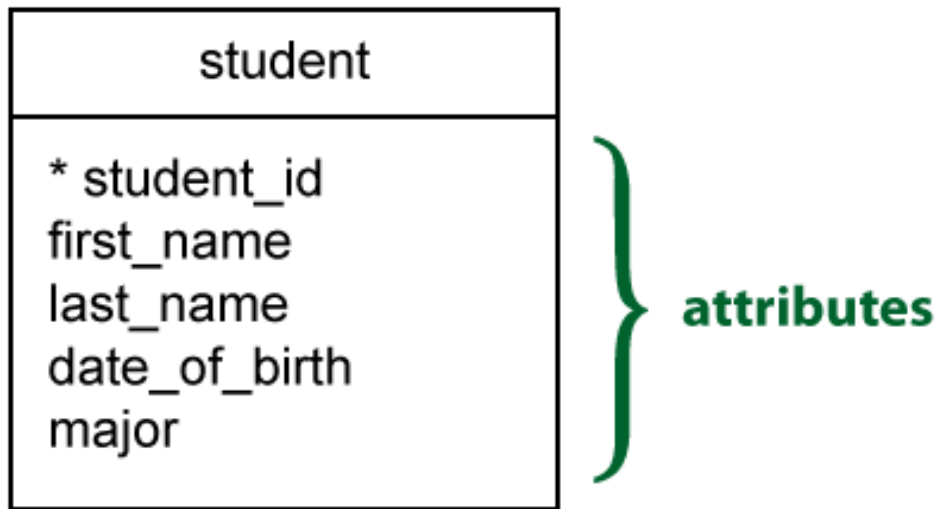
- Replaces cardinality ratio (1:1, 1:N, M:N) and single/double line notation for participation constraints
- Associate a pair of integer numbers (min, max) with each participation of an entity type E in a relationship type R, where $0 \leq \min \leq \max$ and $\max \geq 1$





Entity-Relationship Model Using Crow's Foot Notation

Entities and Attributes



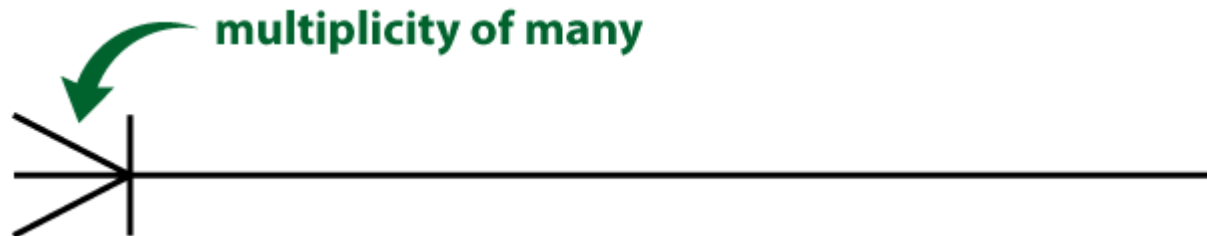
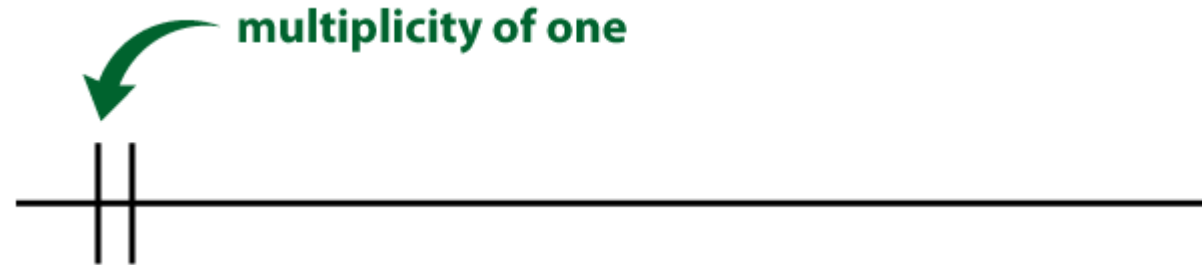
An **entity** is a representation of a class of object. It can be a person, place, thing, etc. Entities usually have attributes that describe them.

An **attribute** is a property that describes a particular entity.

The attribute(s) that uniquely distinguishes an instance of the entity is the **identifier**.

Cardinality / Multiplicity

Multiplicity refers to the **maximum** number of times that an instance of one entity can be associated with instances in the related entity. It can be one or many.



Relationships

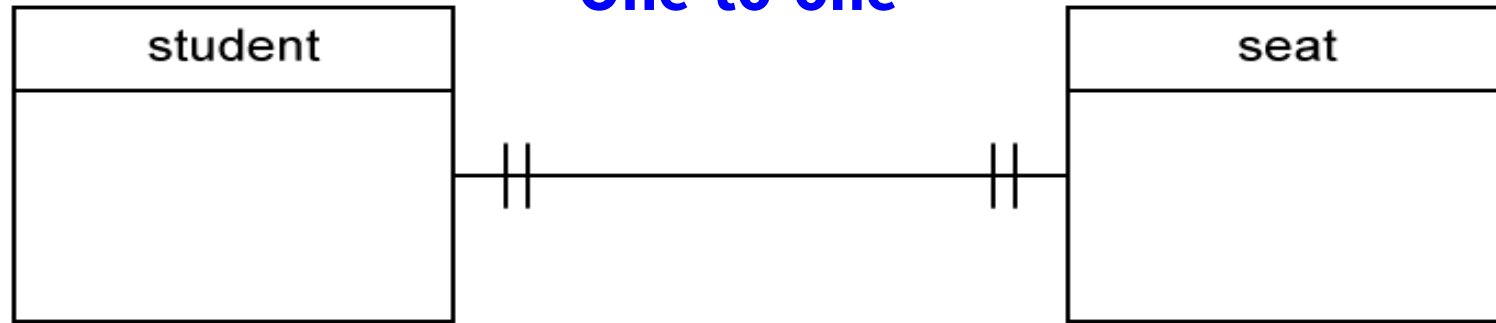


Relationships illustrate the association between two entities. They are presented as a straight line.

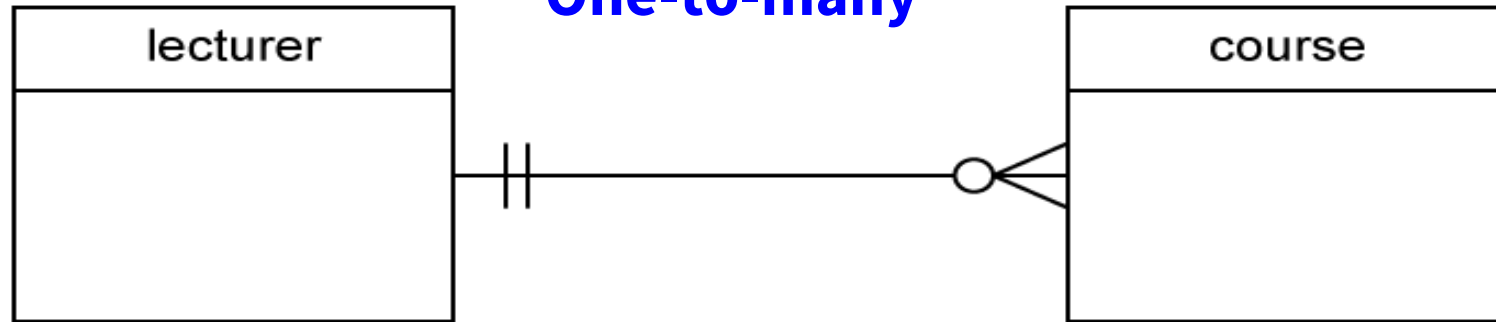


Usually, each relationship has a name, expressed as a verb, written on the relationship line. This describes what kind of relationship connects the objects.

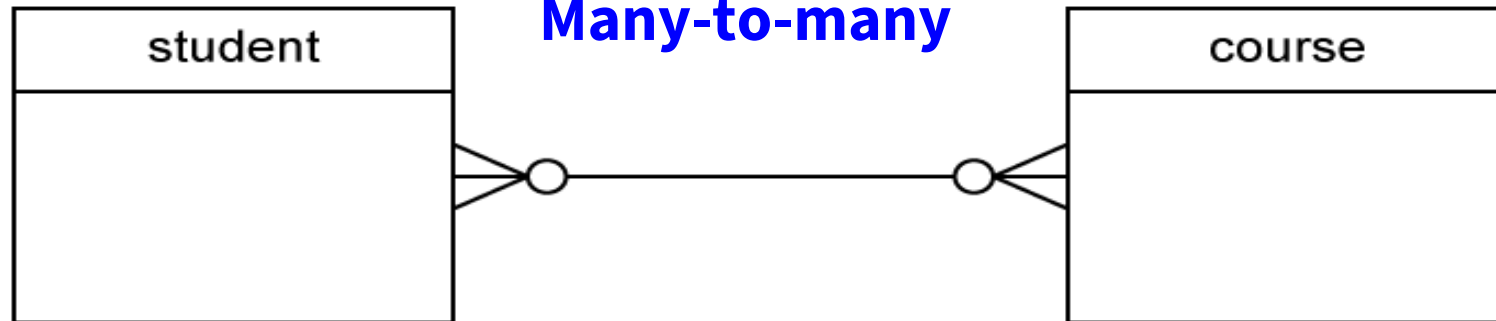
One-to-one



One-to-many



Many-to-many



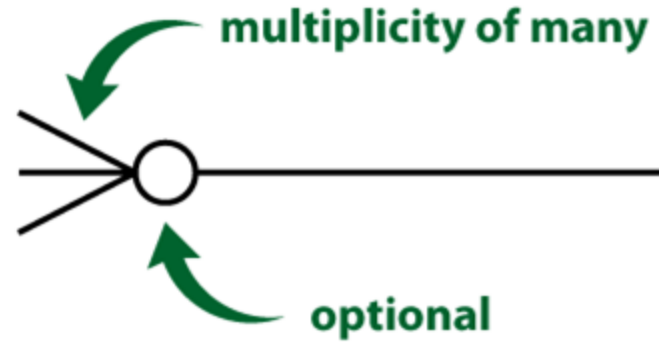
Cardinality / Multiplicity

The **minimum** number of times one instance can be related to others. It can be zero or one, and accordingly describes the relationship as **optional** or **mandatory**.

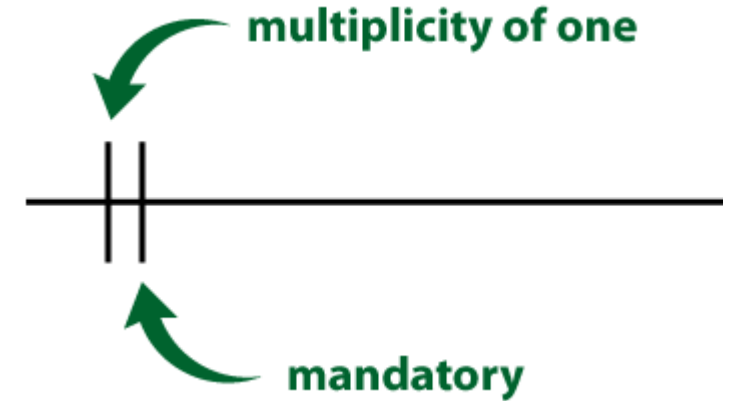


Four possible relationship cardinalities

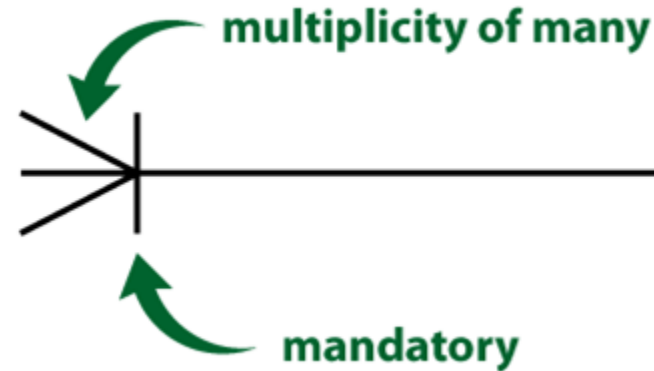
1. zero or many



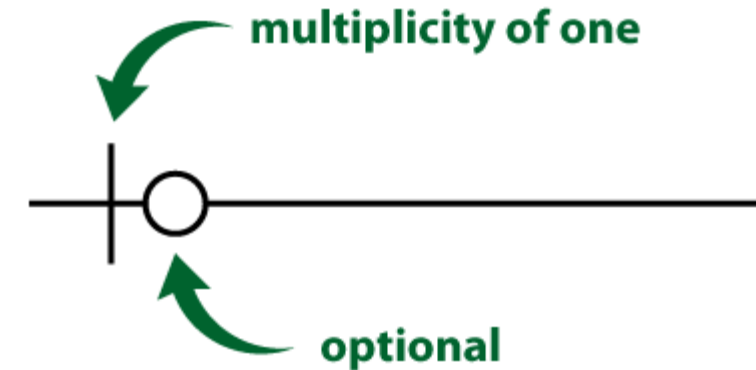
3. one and only one



2. one or many

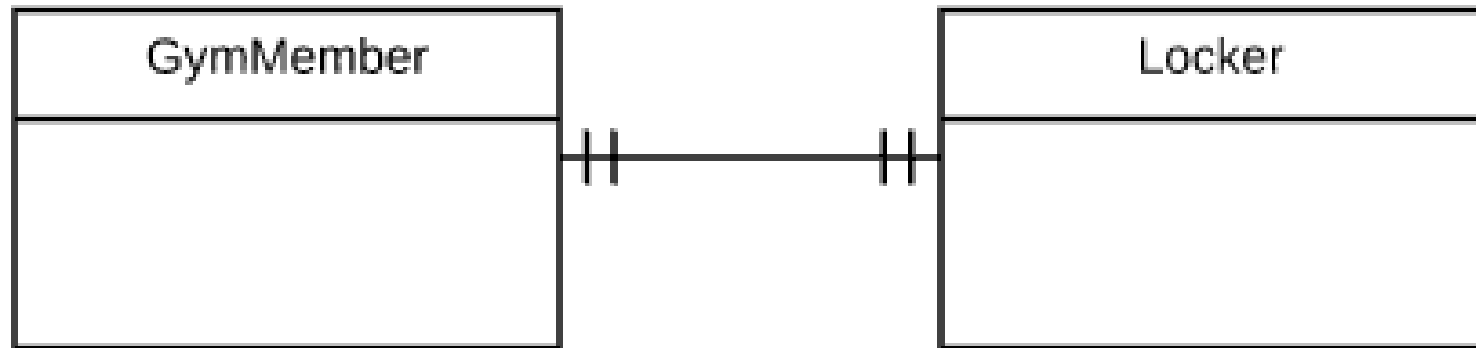


4. zero or one

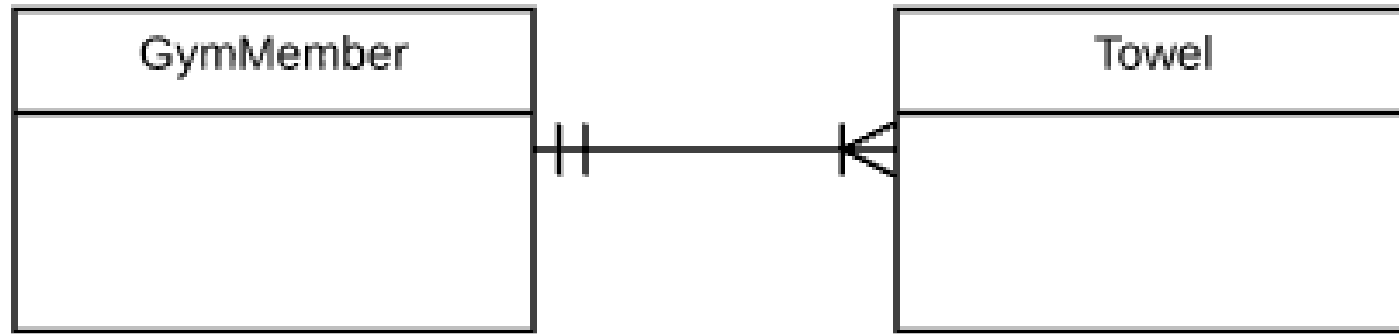


Example

A gym member has **one and only one** locker.

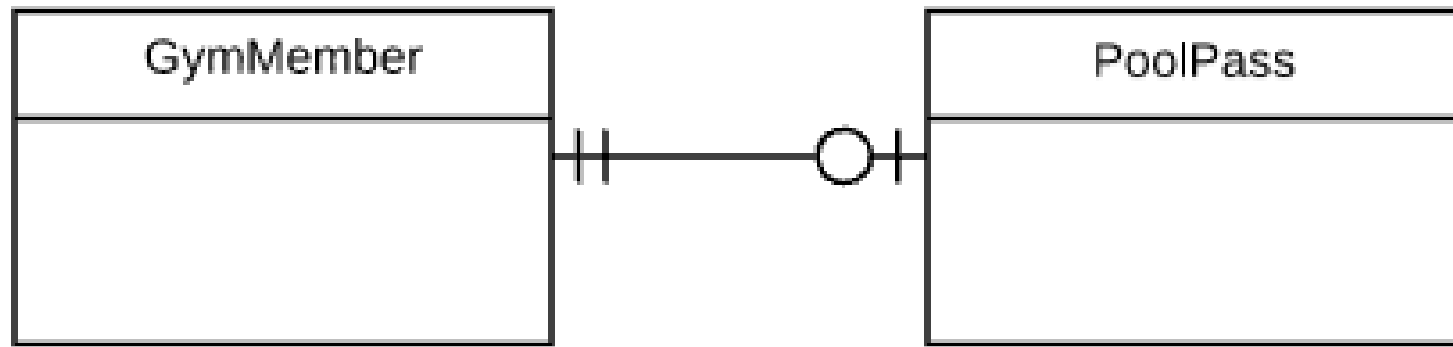


Example



A gym member can have **one or many** towels.

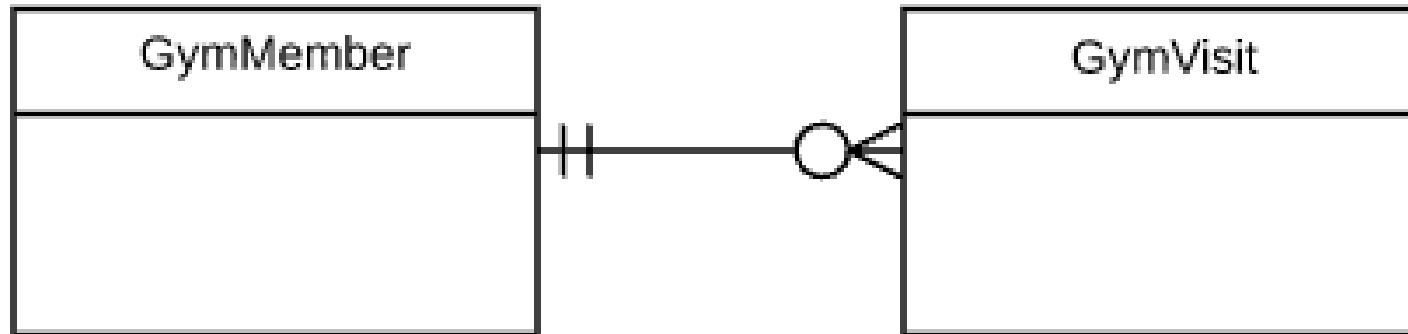
Example



A gym member can have **zero or one** pool pass.

Example

A gym member can have **zero or many** gym visits.



Example of Other Notation

UML CLASS DIAGRAMS

UML Class Diagrams

UML methodology

- Used extensively in software design
- Many types of diagrams for various software design purposes
- UML class diagrams

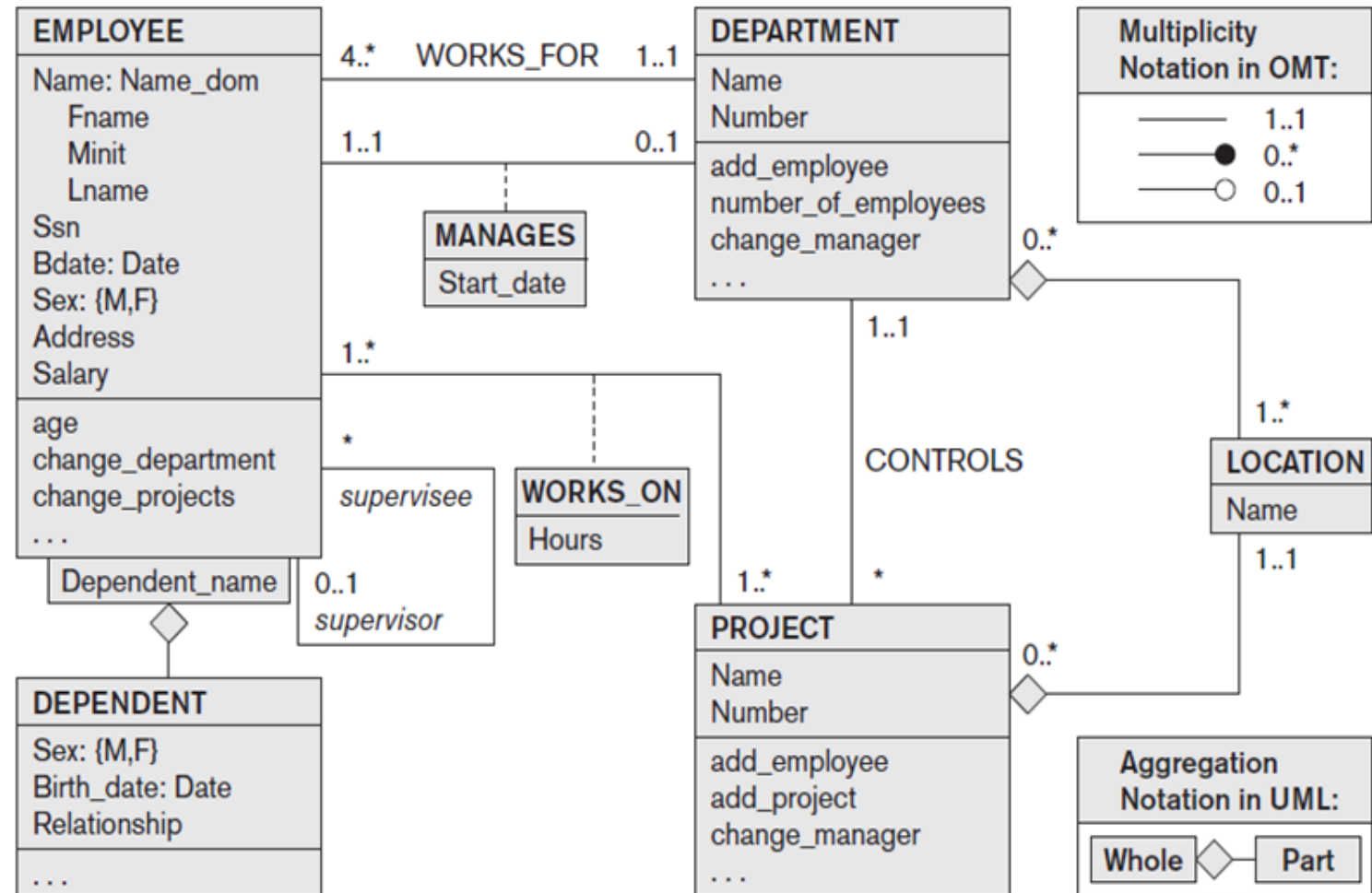
Entity in ER corresponds to an object in UML

Example

UML Class Diagram

Figure 7.16

The COMPANY conceptual schema
in UML class diagram notation.





Thank you.

Discuss 3 important
things / concepts
we have learned
today.

References

R. Elmasri and S. Navathe: Fundamentals of Database Systems, 7/E, Addison-Wesley, 2015