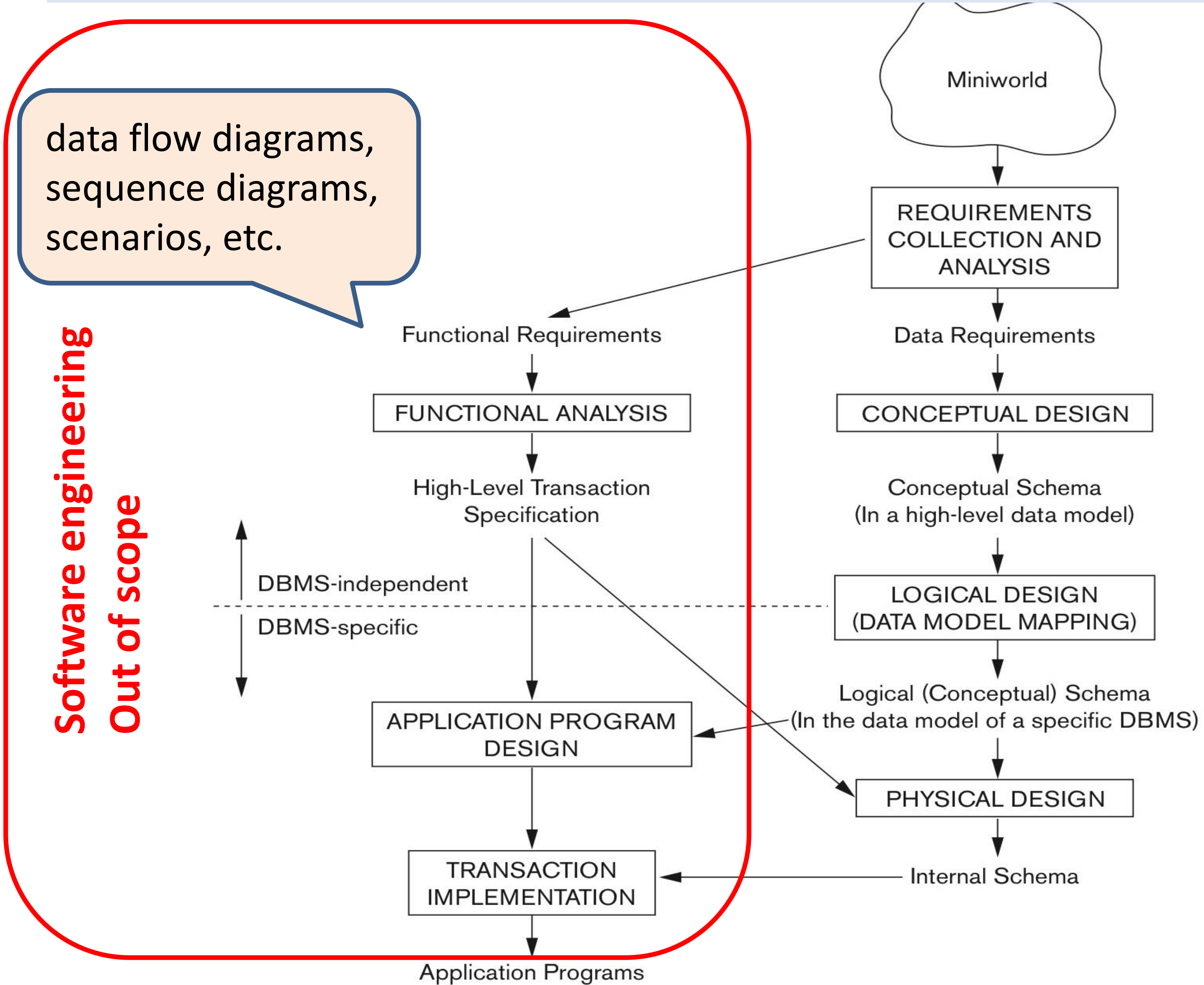
CMPE 138/180B
Database System I
Entity-Relationship (ER) Model

Instructor: Kong Li

Outline

- High-Level Conceptual Data Models for Database Design
- A Sample Database Application
- Entity Types, Entity Sets, Attributes, and Keys
- Relationship Types, Relationship Sets, Roles, and Structural Constraints
- Weak Entity Types
- Refining the ER Design for the COMPANY Database
- ER Diagrams, Naming Conventions, and Design Issues
- Other Notations: (min, max) and UML Class Diagrams
- Relationship Types of Degree Higher than Two

Steps for Database Design



data flow diagrams, sequence diagrams, scenarios, etc.

Software engineering

Out of scope

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

Steps for Database Design (cont'd)

- Requirements collection and Analysis
  - Input: Talk to the right people to collect/analyze requirements
  - Output: data and functional requirements
- Conceptual design
  - Input: data requirements
  - Map data requirements to Entity Relationship Diagram (ERD)
  - Output: conceptual schema - ERDs
- Logical design
  - Input: conceptual schema - ERDs
  - Map ERDs to DB tables and constraints (DDL)
  - Output: logical schema (DBMS specific)
- Physical design
  - Internal storage structures, file organizations, indexes, access paths, and physical design parameters for the database files specified

Why is Data Modeling Important

- Leverage
  - Small change to data model may have big impact on the system
  - Well-designed data model → significant savings in total programming cost
  - Poor data modeling → expensive to fix
- Conciseness
  - Implicitly define a whole set of how to retrieve/update/delete data, what can be done, and what cannot be done
- Data quality
  - Capture "valid" data, based on business requirements
  - Data format, e.g., mm/dd/yyyy or dd/mm/yyyy
  - Data type, e.g., integer, string, etc
  - Integrity constraints

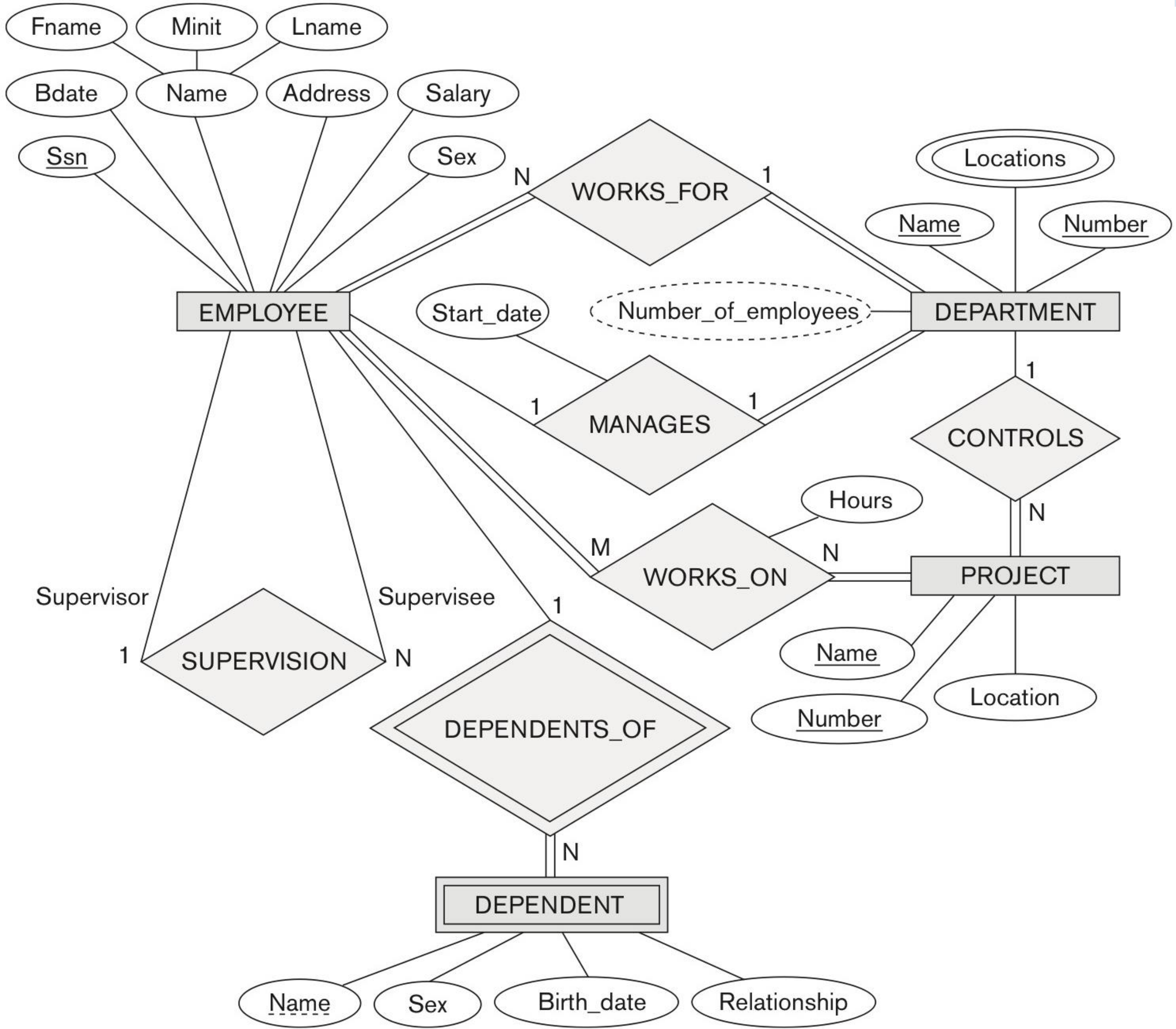
Sample database: COMPANY

- Miniworld:
  - Purpose: why
  - Actors: who
    - roles: what
    - interactions with other actors: how, where, when, whom
  - Operations: data requirements, functional requirements
- Data requirements for the company miniworld
  - Company: multiple departments
  - Department
    - Unique name, unique number
    - One head of the department (employee), w/ start date
    - Several locations
    - Must have one or more employees
    - May control certain projects

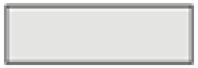
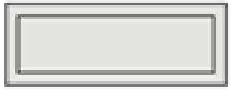
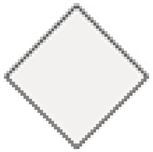
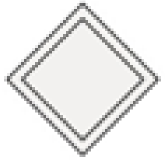
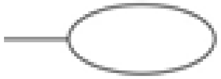
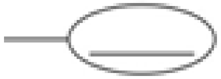
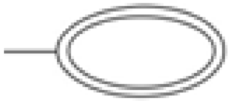
Sample database: COMPANY

- Project
  - Unique name, unique number
  - Must be controlled by a single department
  - Single location
  - Must have one or more workers (employees)
- Employee
  - Name, ssn, address, salary, gender, b-day
  - Must work for a single department
  - Must work on one or more projects (controlled by own or other departments)
    - Number of hours/week/project
  - May report to a supervisor (another employee)
    - May supervise a few employees
  - May have dependents
    - Each dependent: first name (unique for a given employee only), gender, b-day, relationship
  - May be manager (manage up to one department)

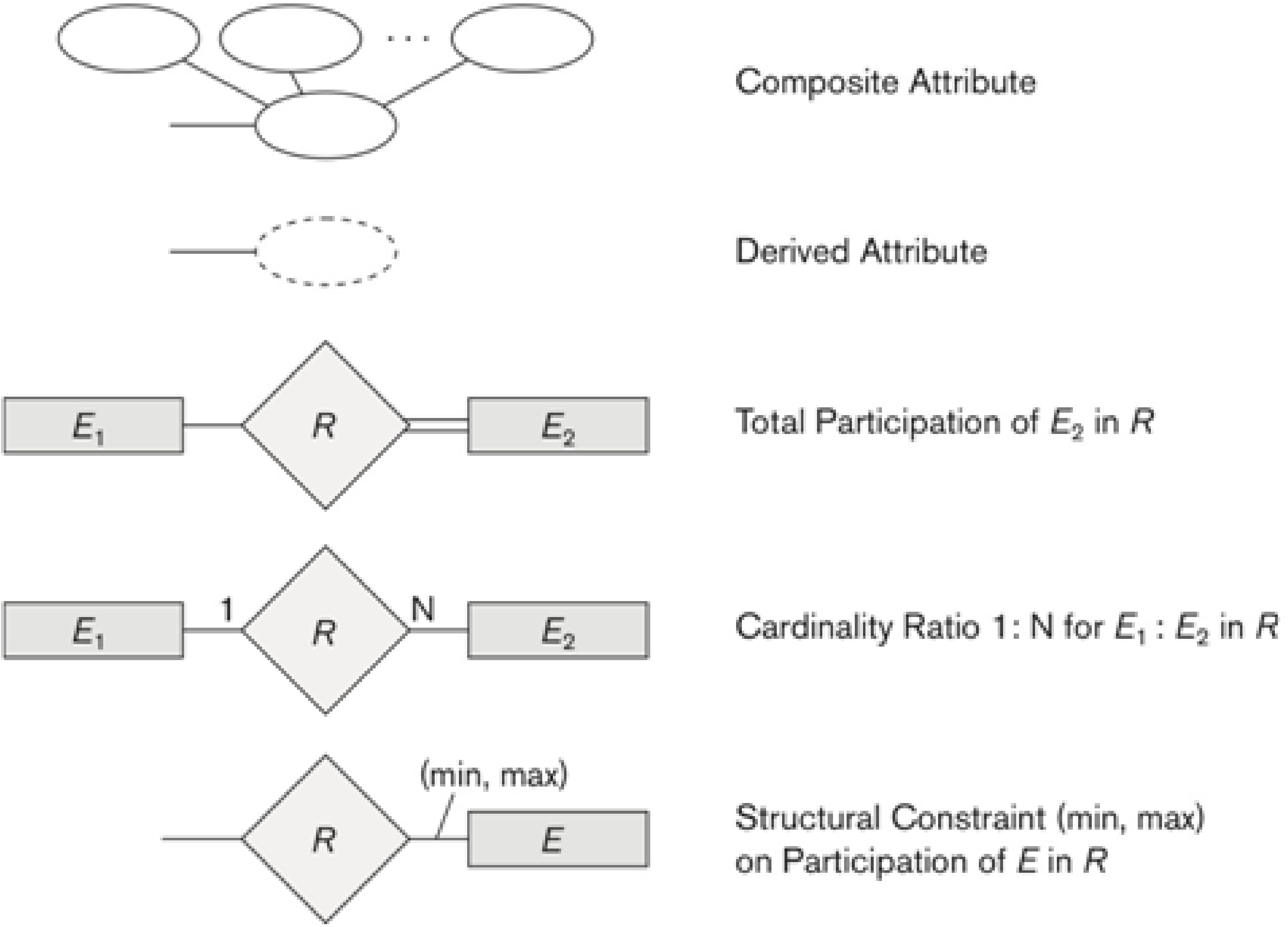
ER Diagram of COMPANY



ER Diagram Notations – Peter Chen

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

ER Diagram Notations – Peter Chen (cont'd)



Composite Attribute

Derived Attribute

Total Participation of E_2 in R

Cardinality Ratio 1: N for E_1 : E_2 in R

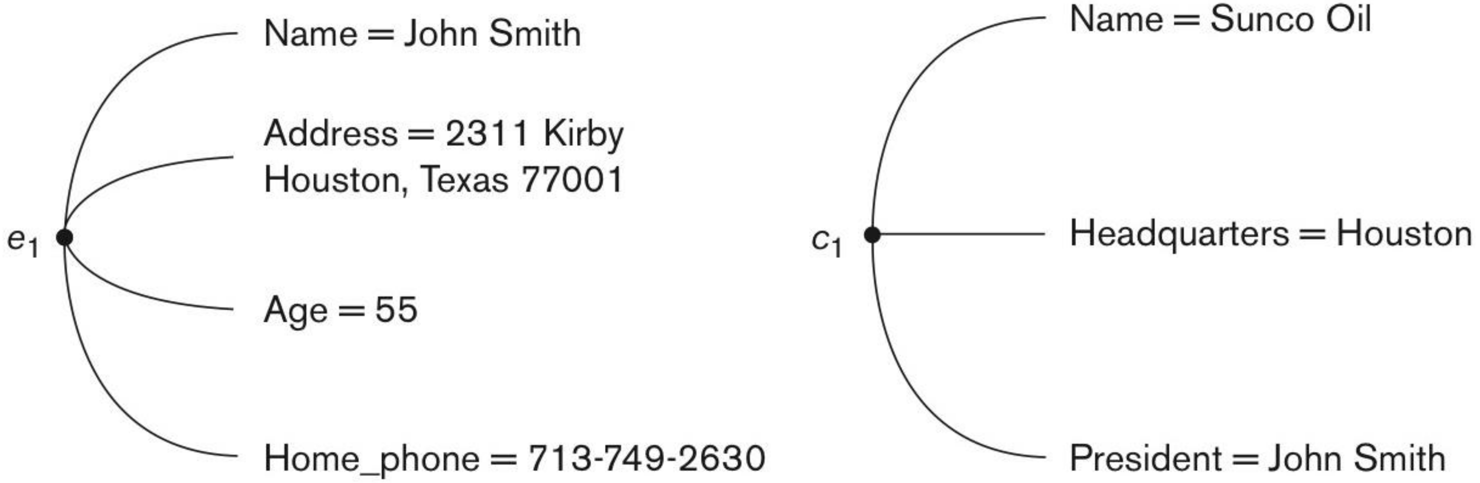
Structural Constraint (min, max) on Participation of E in R

ER Model

- ER model describes data as
  - Entities
    - Objects
  - Attributes
    - Properties of objects
  - Relationships
    - Data association among objects

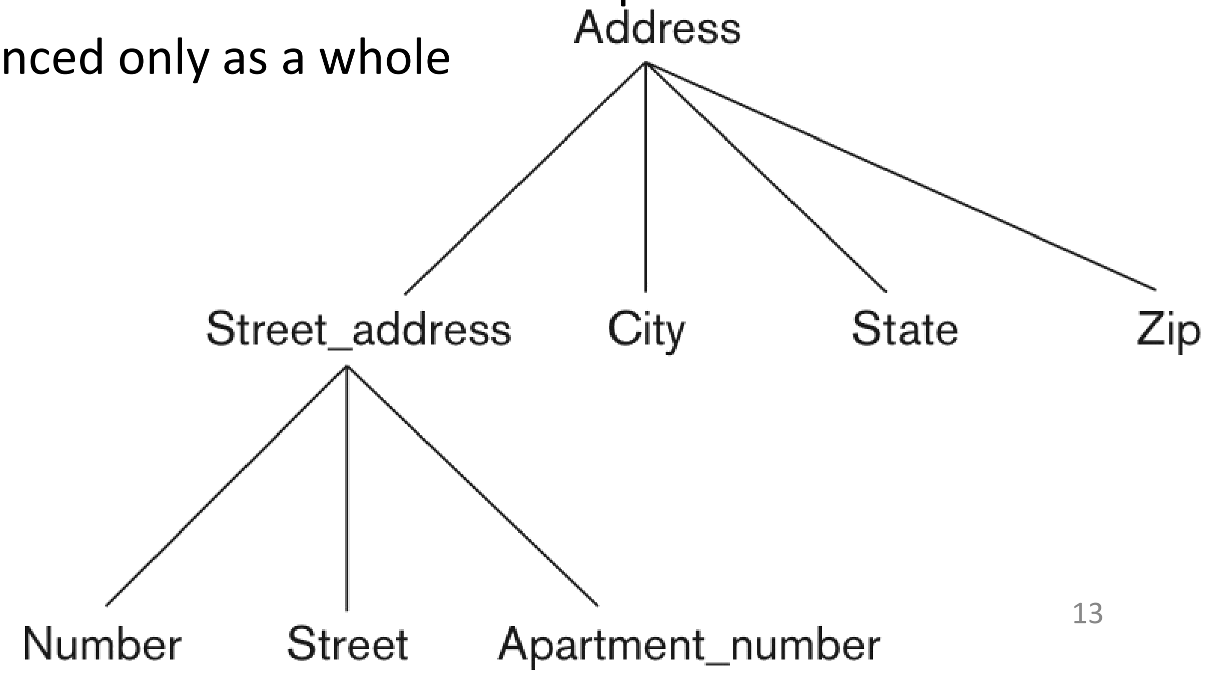
Entities and Attributes

- **Entity**
  - – Thing in real world with independent existence
    - Object w/ physical existence, e.g., car, person, house, employee
    - Object w/ conceptual existence, e.g., job, course, company, univ
- **Attributes**
  - – Particular properties that describe entity
  - – Each attribute has value(s)



Attribute Types

- attribute type in ERD: oval
- simple vs composite attributes
  - Simple attr: single atomic value; referenced as a whole
  - Composite attr:
    - referenced as a unit or referenced to its components
    - Useless if referenced only as a whole



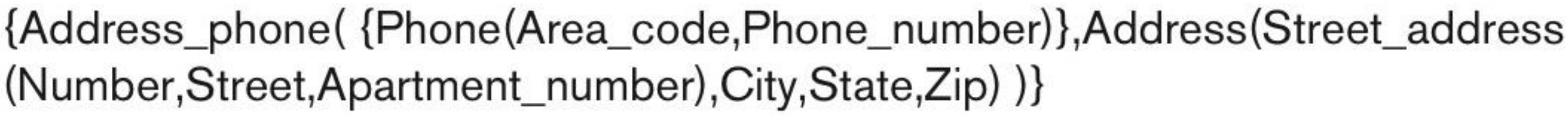
Attribute Types (cont'd)

- Single-valued vs multi-valued attributes
  - single-valued: age, b-day

    oval
  - multi-valued: {phone}, {college-degree}

    Double-lined oval
  - "NY, LA, SF" vs "NY", "LA", "SF"
- Stored vs derived attributes

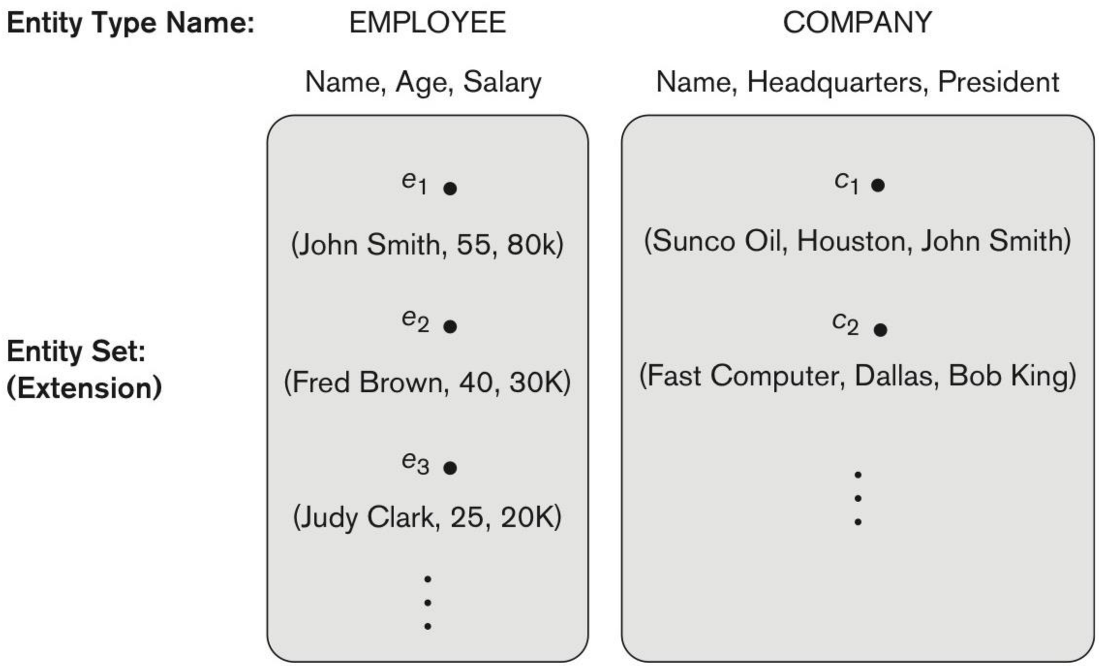
  Dashed oval
  - E.g., b-day → age
- NULL value: Unknown, N/A, missing
- Complex Attributes
  - Composite and multi-valued attributes can be nested



{Address_phone({Phone(Area_code,Phone_number)},Address(Street_address (Number,Street,Apartment_number),City,State,Zip))}

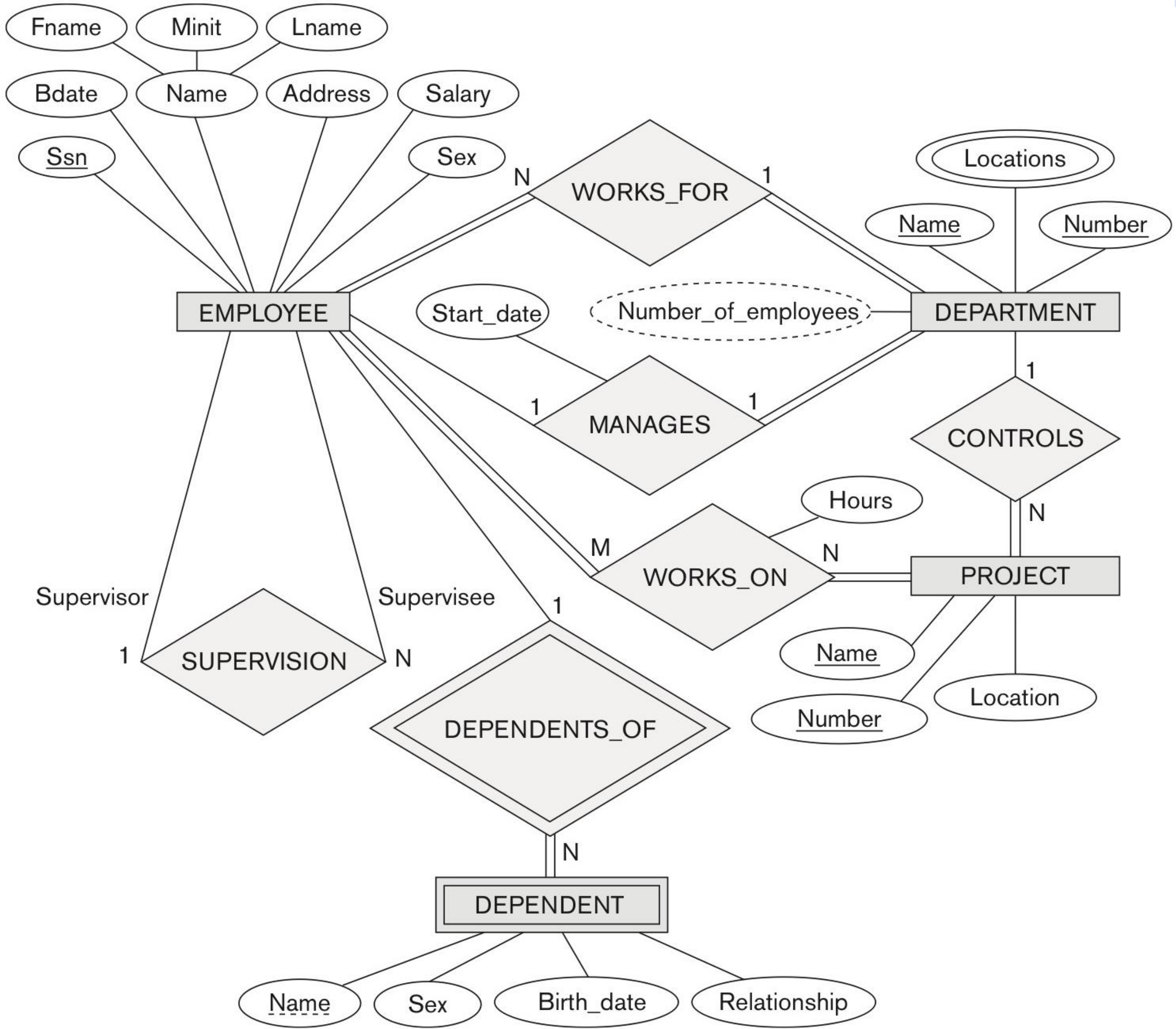
Entity Types, Entity Sets

- Entity type
  - Collection (or set) of entities w/ the same set of attr
  - Each such entity has own value(s) for each attr



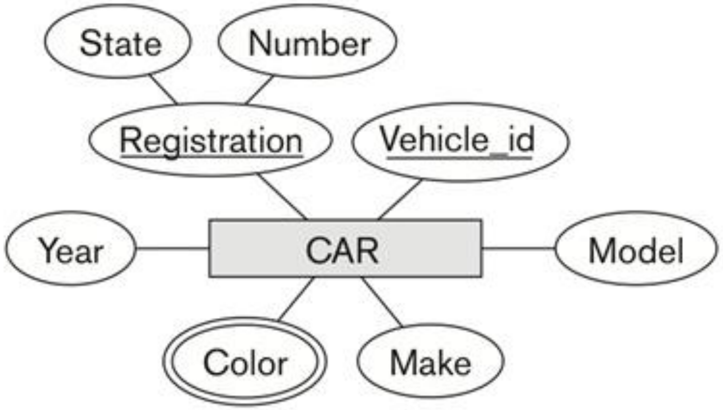
  - E.g., EMPLOYEE is both entity type and entity set
  - Entity type in ERD: rectangle

ER Diagram of COMPANY



Attribute: Key, Value Sets

- Key attribute or uniqueness constraint
  - Attribute w/ distinct value for each individual entity in entity set
  - Key attribute in ERD: underlined oval
  - A entity type may have more than one key attr



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

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

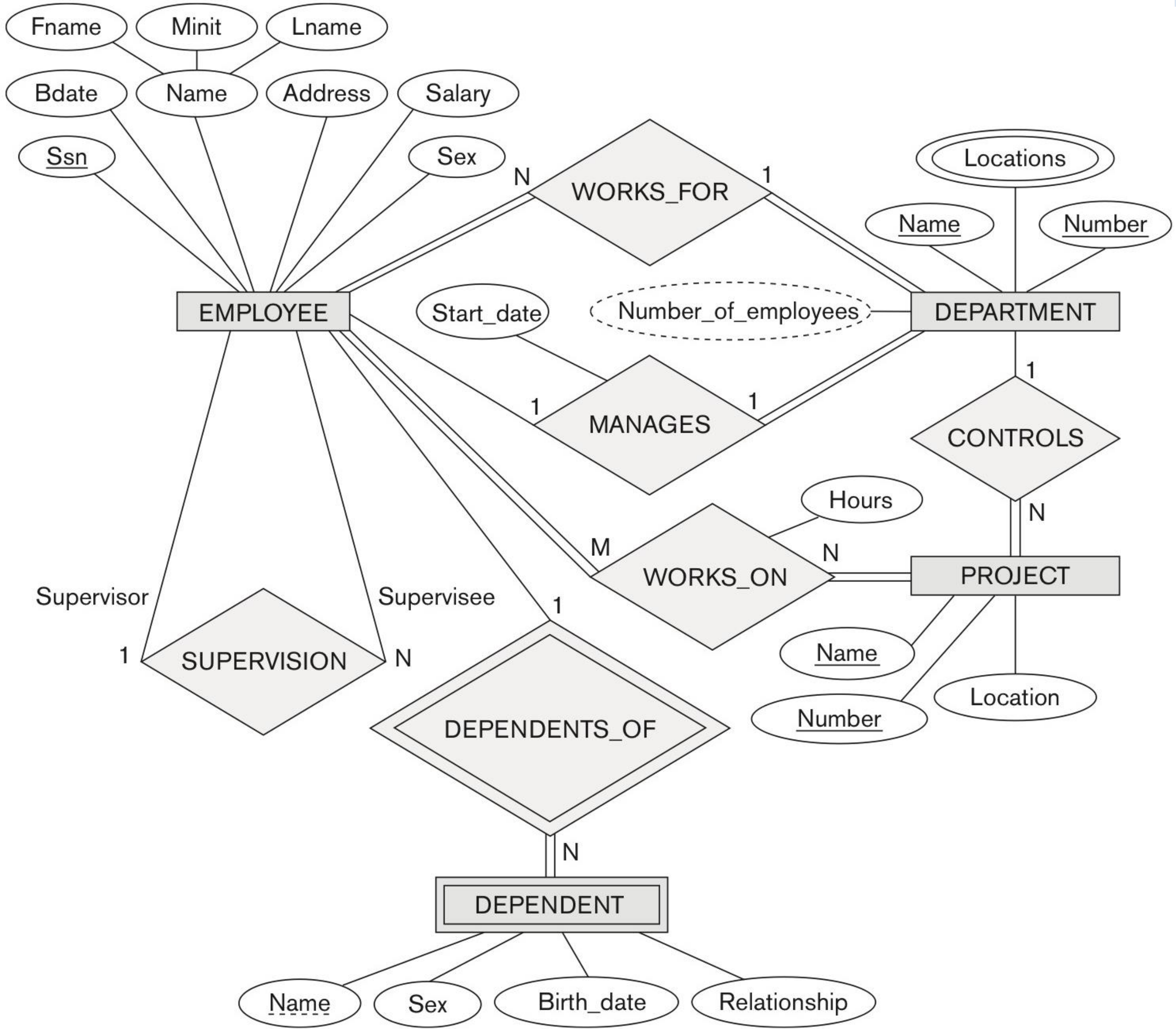
CAR_2
((ABC 123, NEW YORK), WP9872, Nissan Maxima, 4-door, 2005, {blue})

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

⋮

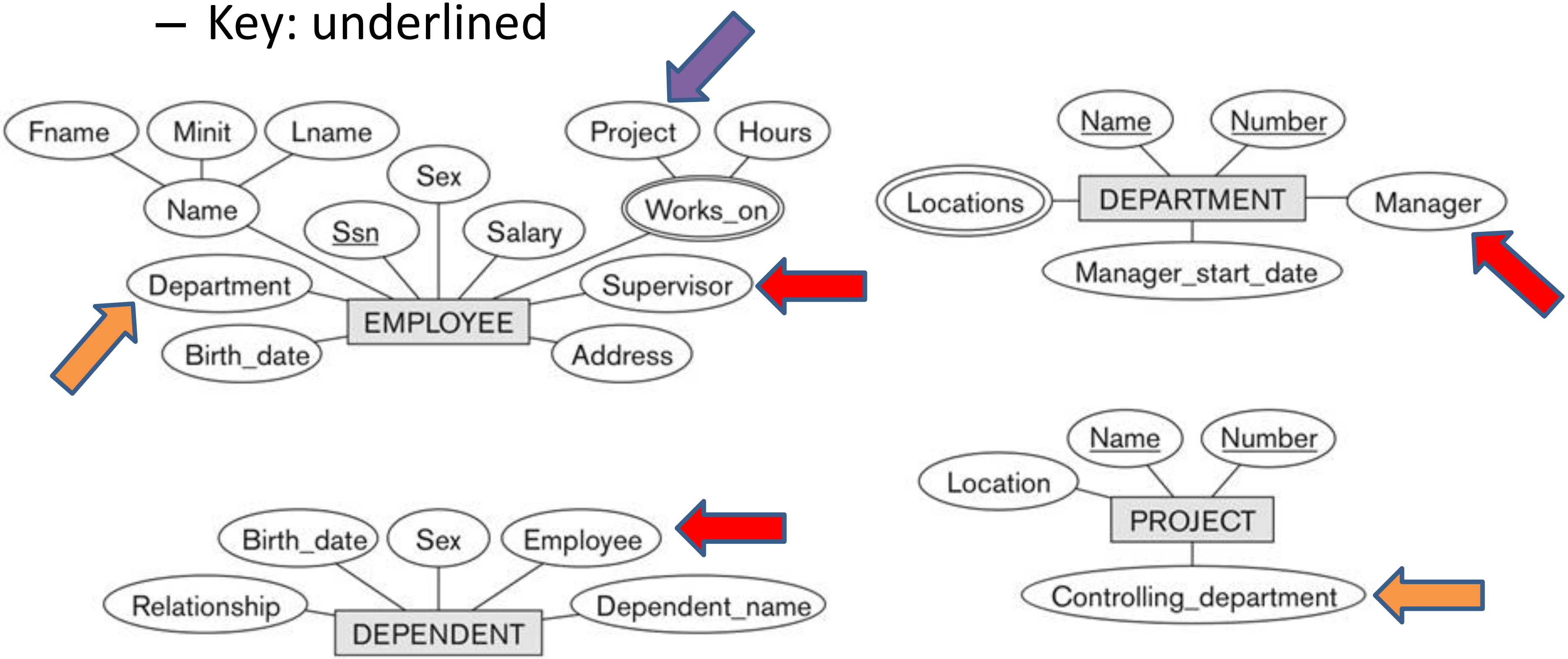
- Value sets (domain) of attribute: all possible values

ER Diagram of COMPANY



Initial Conceptual Design of the COMPANY database

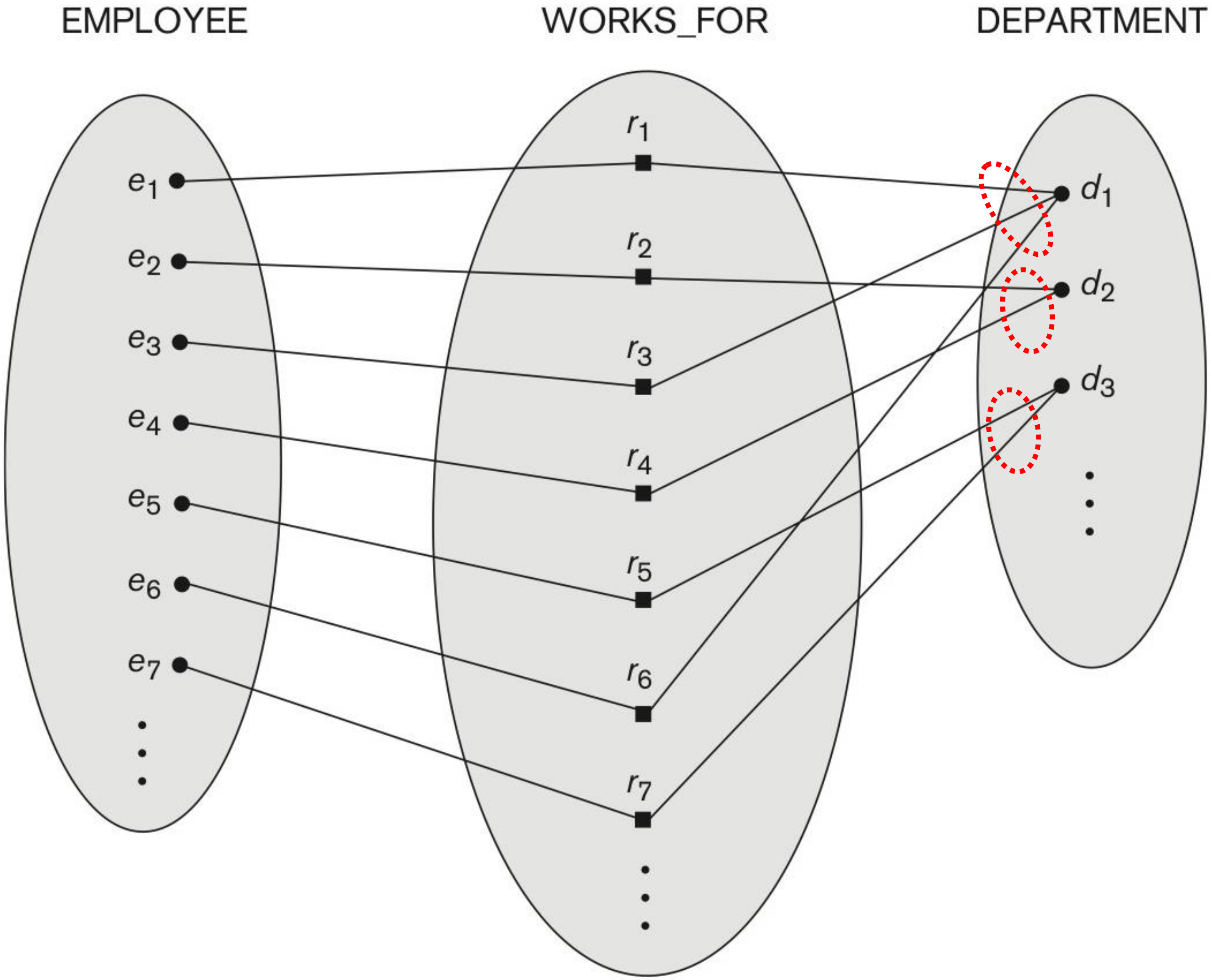
- Entity type: rectangle
- Attribute type: oval, double-lined, dashed-lined
    - Key: underlined



Relationship Types, Sets, Instances

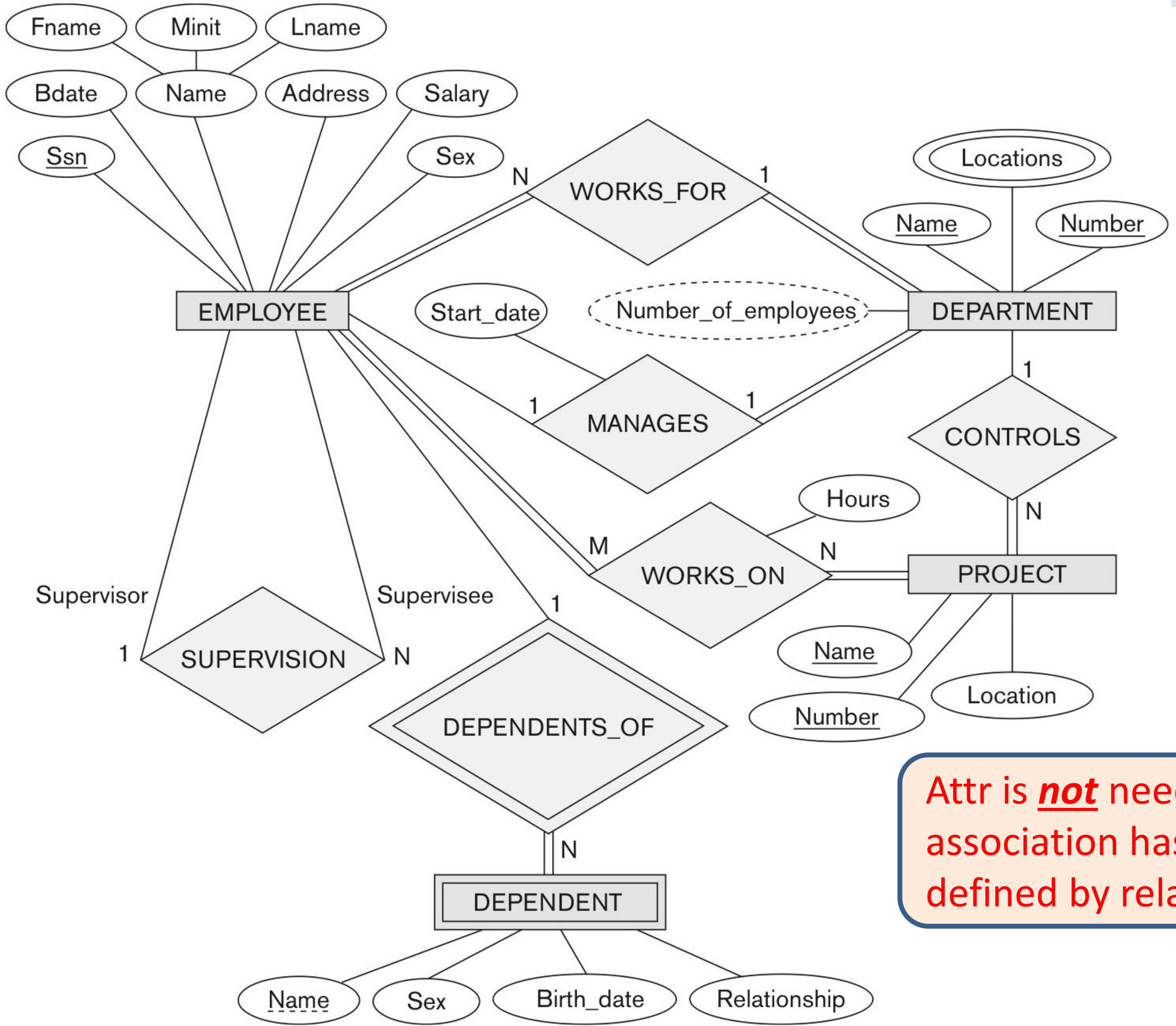
- Relationship
  - Data association: one entity type refers to another entity type
  - represent such references as relationships, not as attrs
- Relationship type R among n entity types E_1, E_2, ..., E_n
  - Defines a set of associations among entities from these entity types
- Relationship set R == set of relationship instances r_i
  - r_i : associates n individual entities (e_1, e_2, ..., e_n)
  - e_j in r_i : member of entity set E_j
- relationship type in ERD: diamond

Relationship Instances



Derived attr: number of employees

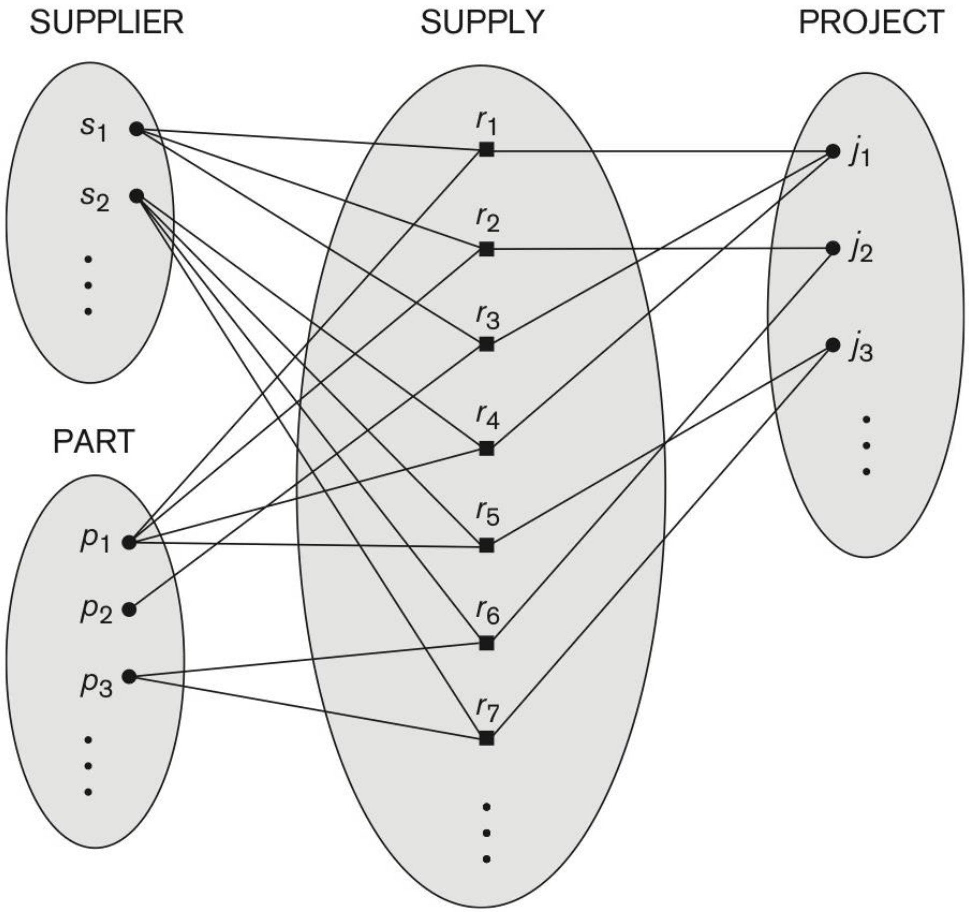
ER Diagram of COMPANY



Attr is ***not*** needed if data association has been defined by relationship

Relationship Degree

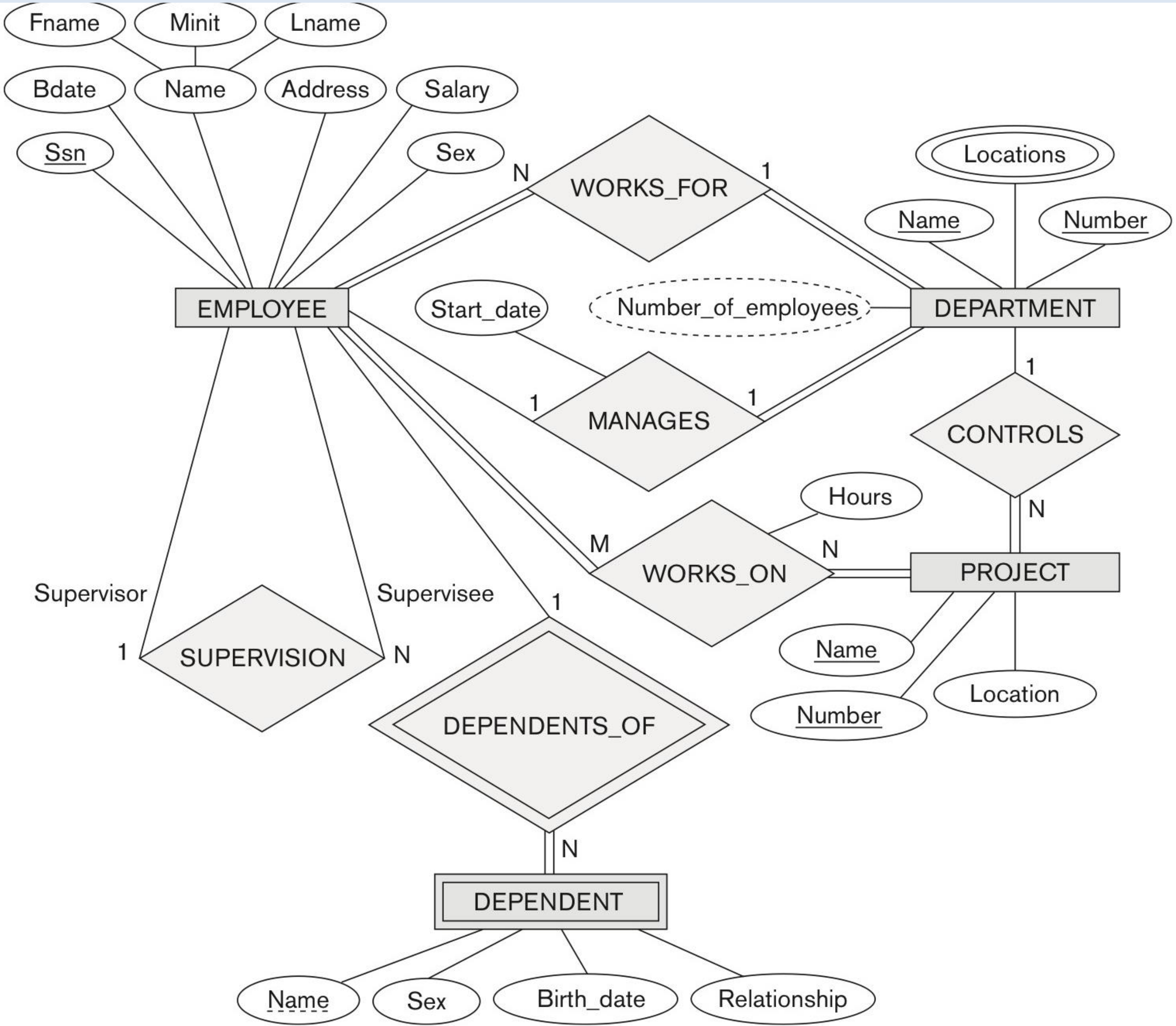
- Degree of a relationship type
  - Number of participating entity types
  - Binary, ternary, etc



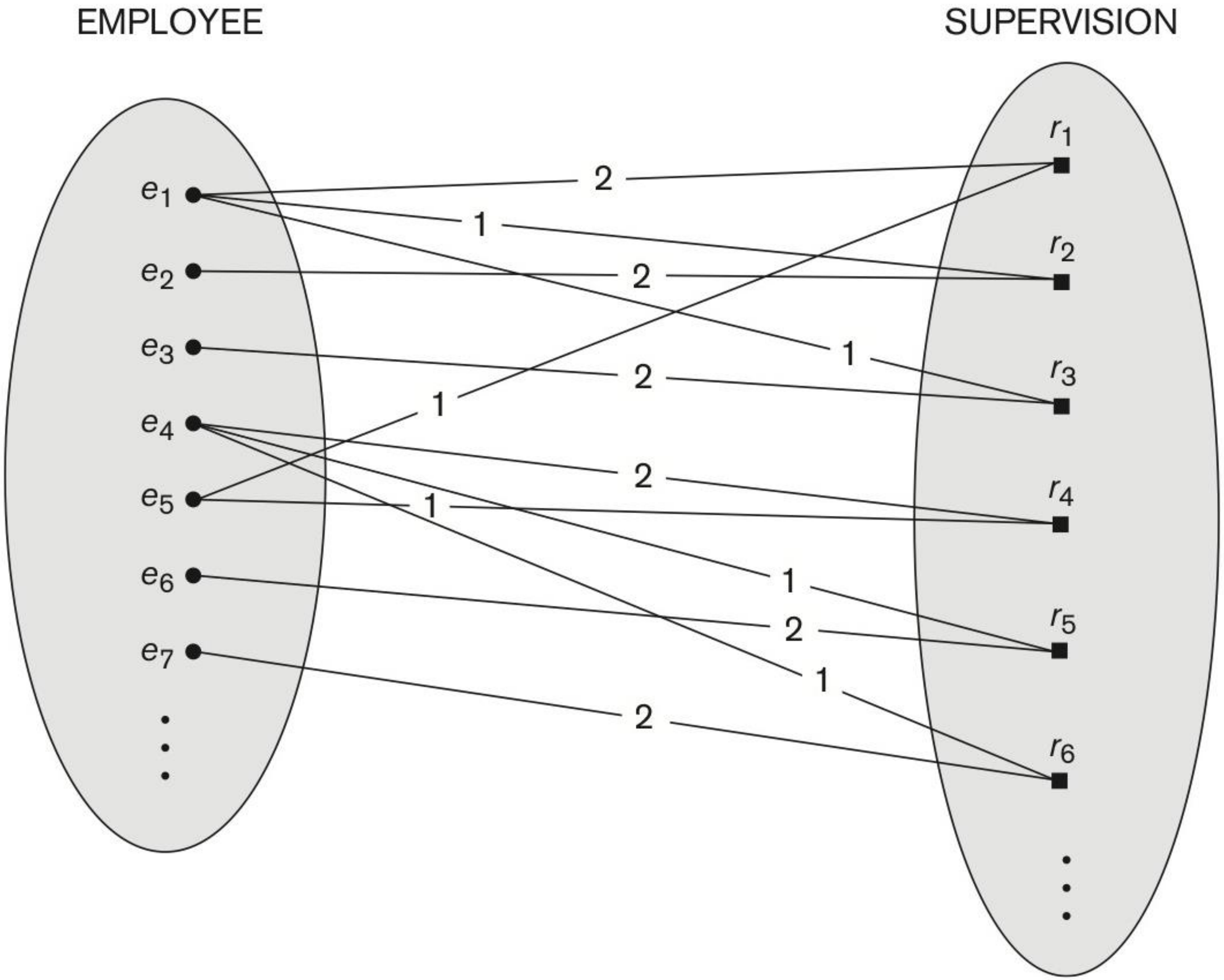
Role and Recursive Relationship

- Role: Help to explain the relationship
  - WORKS_FOR relationship:
    - EMPLOYEE: role - employee or worker
    - DEPARTMENT: role - employer or department
- Role name
  - defines the role that an entity type plays in relationship
  - Optional but essential for recursive relationship
- Recursive relationship
  - same entity type participates more than once in a relationship type in different roles
  - E.g., supervision

ER Diagram of COMPANY



Recursive Relationship



1: supervisor role
2: supervisee role

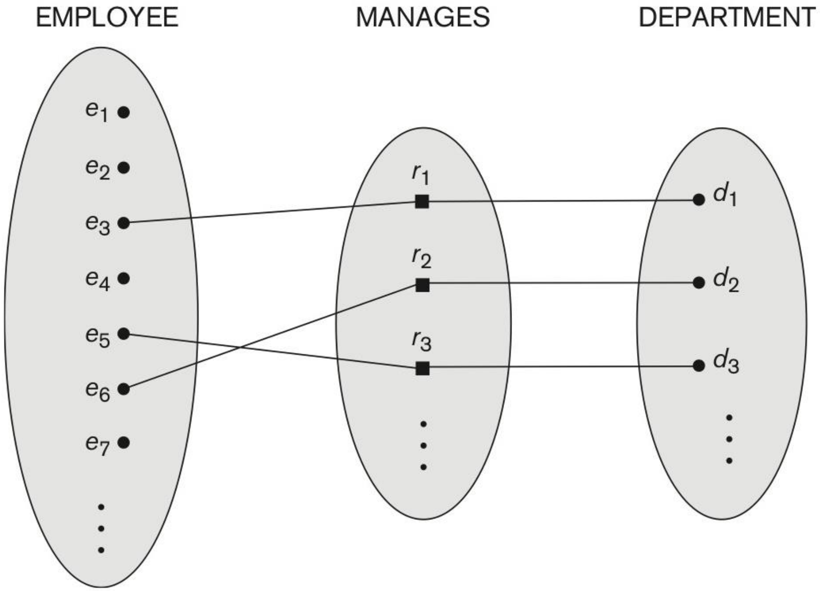
e_1 supervises e_2 and e_3

Relationship: Structural Constraints

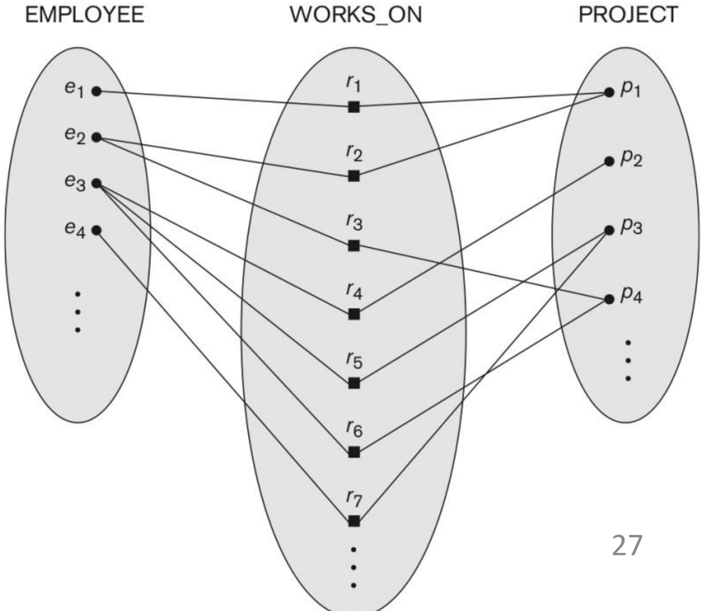
- **Cardinality ratio** for a binary relationship
  - Specifies *max* number of relationship instances that entity can participate in
  - Types: 1:1, 1:N, N:1, M:N
  - ER diagram: display 1, M, or N on diamond

1: max
M or N: no max
Explicit value if known

1:1

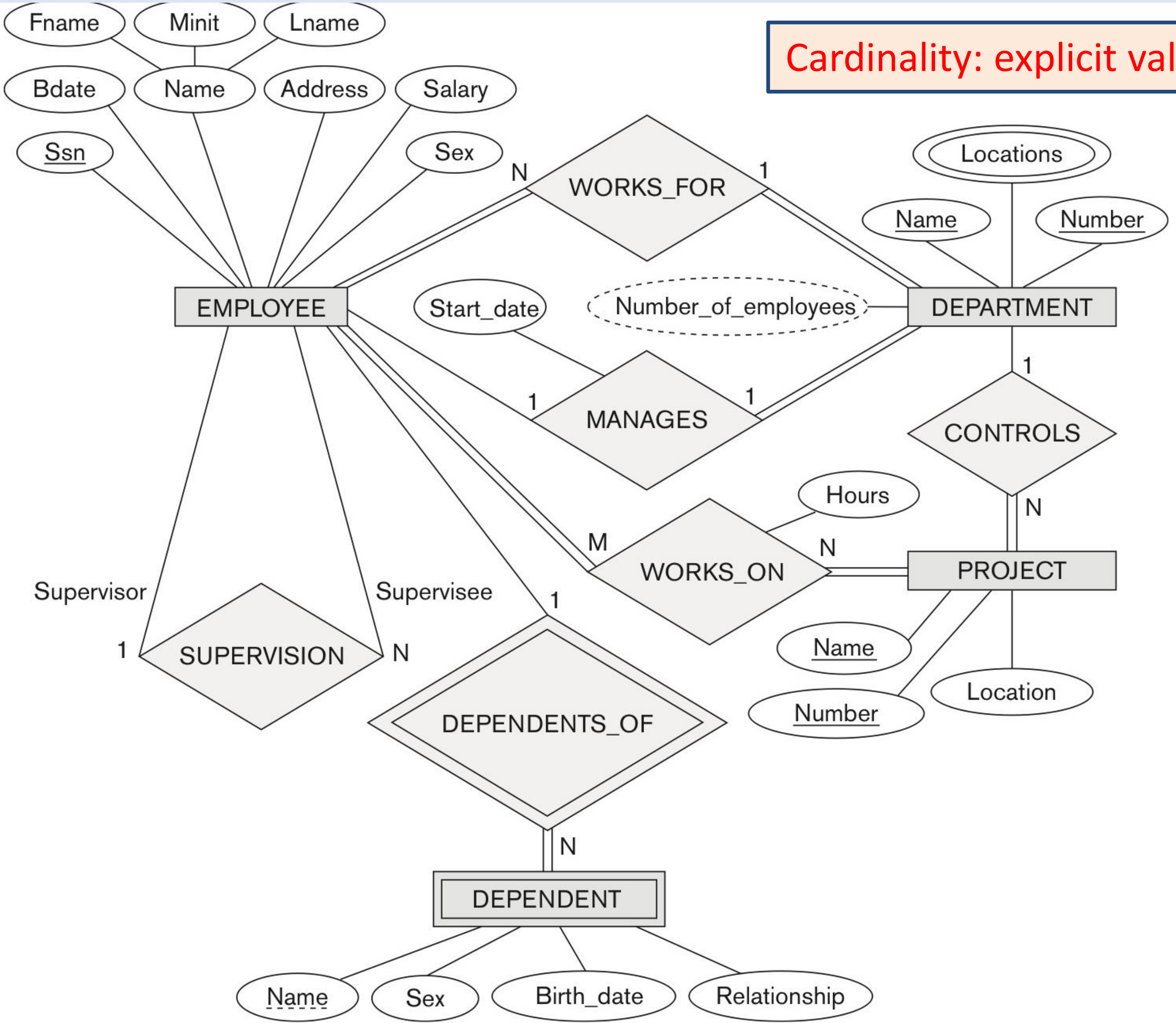


M:N



ER Diagram of COMPANY

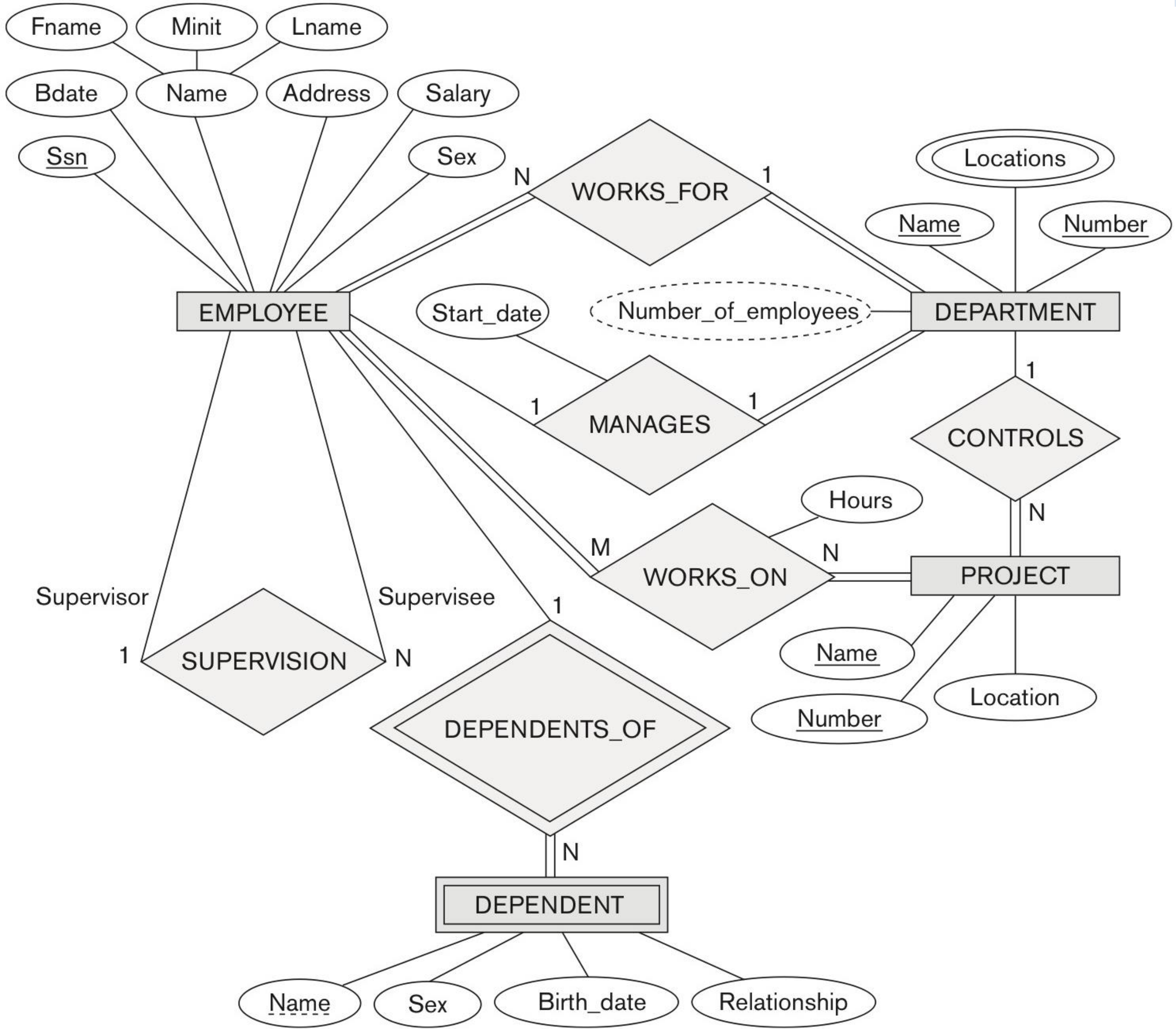
Cardinality: explicit value if known



Relationship: Structural Constraints (cont’d)

- **Participation constraint (*minimum* cardinality constraint)**
  - Specifies whether existence of entity depends on its being related to another entity
  - Types: total and partial
- **Total participation (existence dependency)**
  - every employee must work for a department
    - an employee entity can exist only if it participates in at least one WORKS_FOR relationship instance
  - ERD: double line connecting entity type and relationship type
- **Partial participation**
  - Some, not all, employees manage some departments
  - Some, not all, employees have dependents
  - ERD: single line connecting entity type and relationship type

ER Diagram of COMPANY

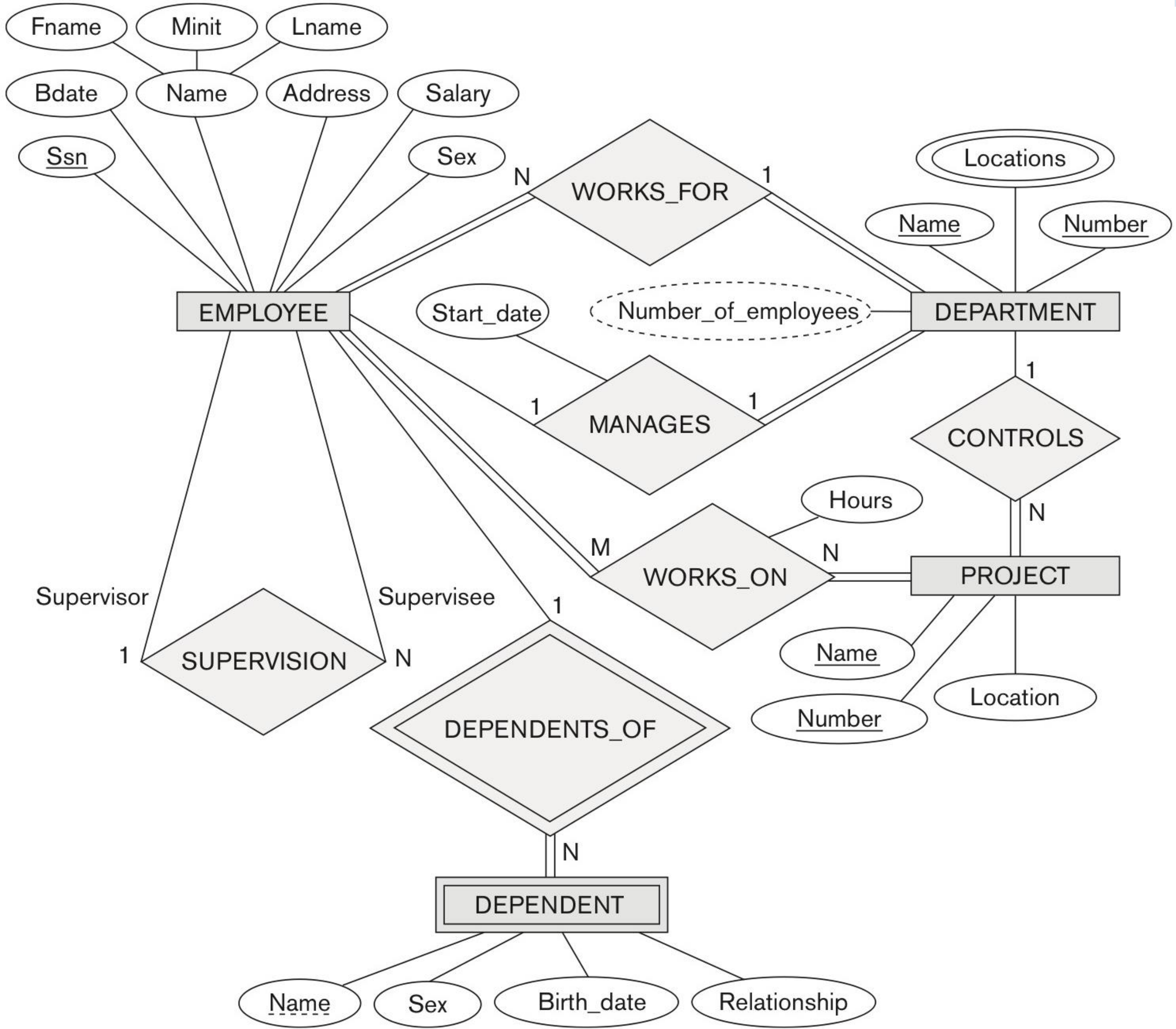


Attributes of Relationship Types

- E.g., attr Hours for the WORKS_ON relationship
- E.g., attr Start_date for the MANAGES relationship
- Migrating attrs of relationship to entity type
  - 1:1 relationship type:
    - Can be migrated to either one entity type
    - e.g., Start_date attr for MANAGES relationship → attr of either EMPLOYEE or DEPARTMENT
  - 1:N relationship type:
    - Can be migrated *only* to entity type on N-side of relationship
    - E.g., Start_date attr for WORKS_FOR relationship → attr of EMPLOYEE, not DEPARTMENT
  - M:N relationship type:
    - Some attrs may be determined by combination of participating entities → Must be specified as relationship attrs
    - E.g., hours for WORKS_ON relationship

attr of relationship: can never be key attr

ER Diagram of COMPANY



Weak Entity Types

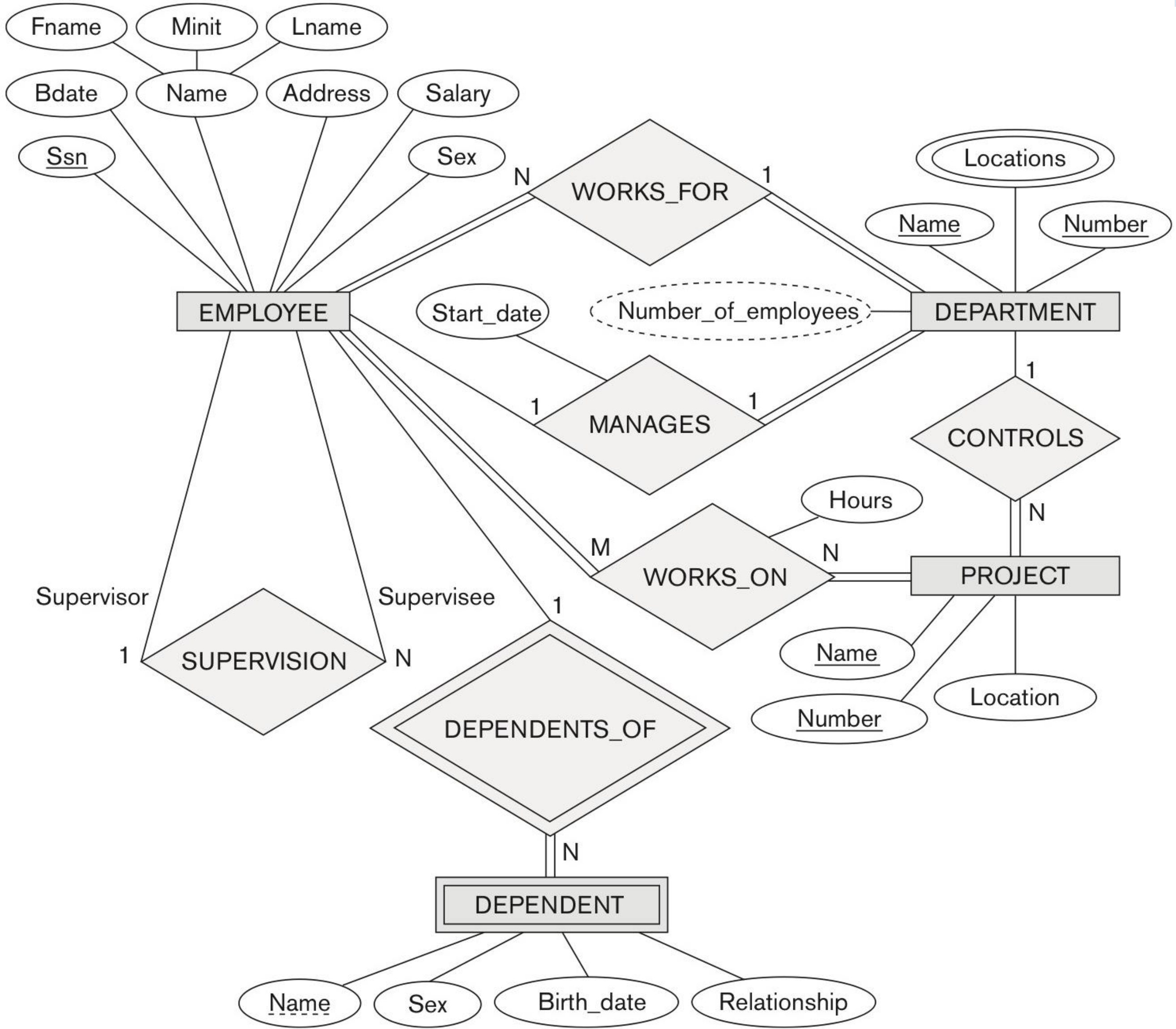
- Do not have key attrs of their own
  - Identified by being related to specific entities from another entity type
  - E.g., DEPENDENT: weak entity type wrt EMPLOYEE
- Usually have **partial key**
  - Uniquely distinguish weak entities related to the same owner entity
  - May have duplicated value among all (weak) entities
  - E.g., Name attr in DEPENDENT
- w/ one or more **Identifying (or owner or parent) entity type**
- **Identifying relationship**
  - Relates a weak entity type to its owner (or parent) entity type(s)
  - A weak entity type: at least one, may be more, identifying entity types
  - Cardinality ratio: usually 1:1 or 1:N (where 1 is on owner entity side)
- ***Always*** has a total participation constraint (existence dependency) b/w identifying relationship and weak entity

Other weak entity example?

Weak Entity Types (cont'd)

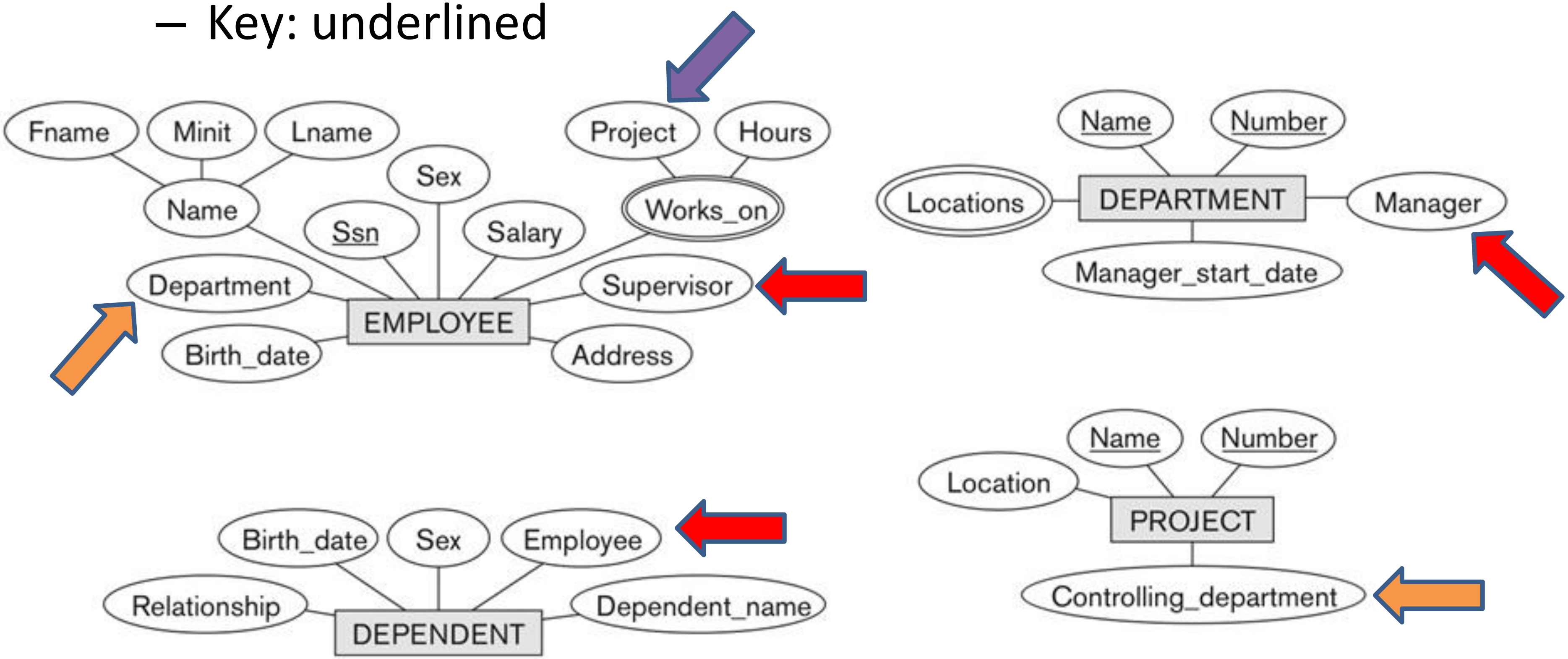
- Must have identifying relationship to identify parent(s)
- Can optionally have (regular) relationships with entities
- Not every existence dependency results in a weak entity type
  - E.g., DRIVER_LICENSE entity cannot exist unless it is related to a PERSON entity, even though it has its own key (License_number) and hence is not a weak entity
- ER diagram:
  - Weak entity type: double-lined rectangle
  - Identifying relationship: double-lined diamond
  - Partial key: dashed underline or dotted underline
- Alternative to weak entity types: complex (composite, multi-valued) attrs

ER Diagram of COMPANY



Initial Conceptual Design of the COMPANY database

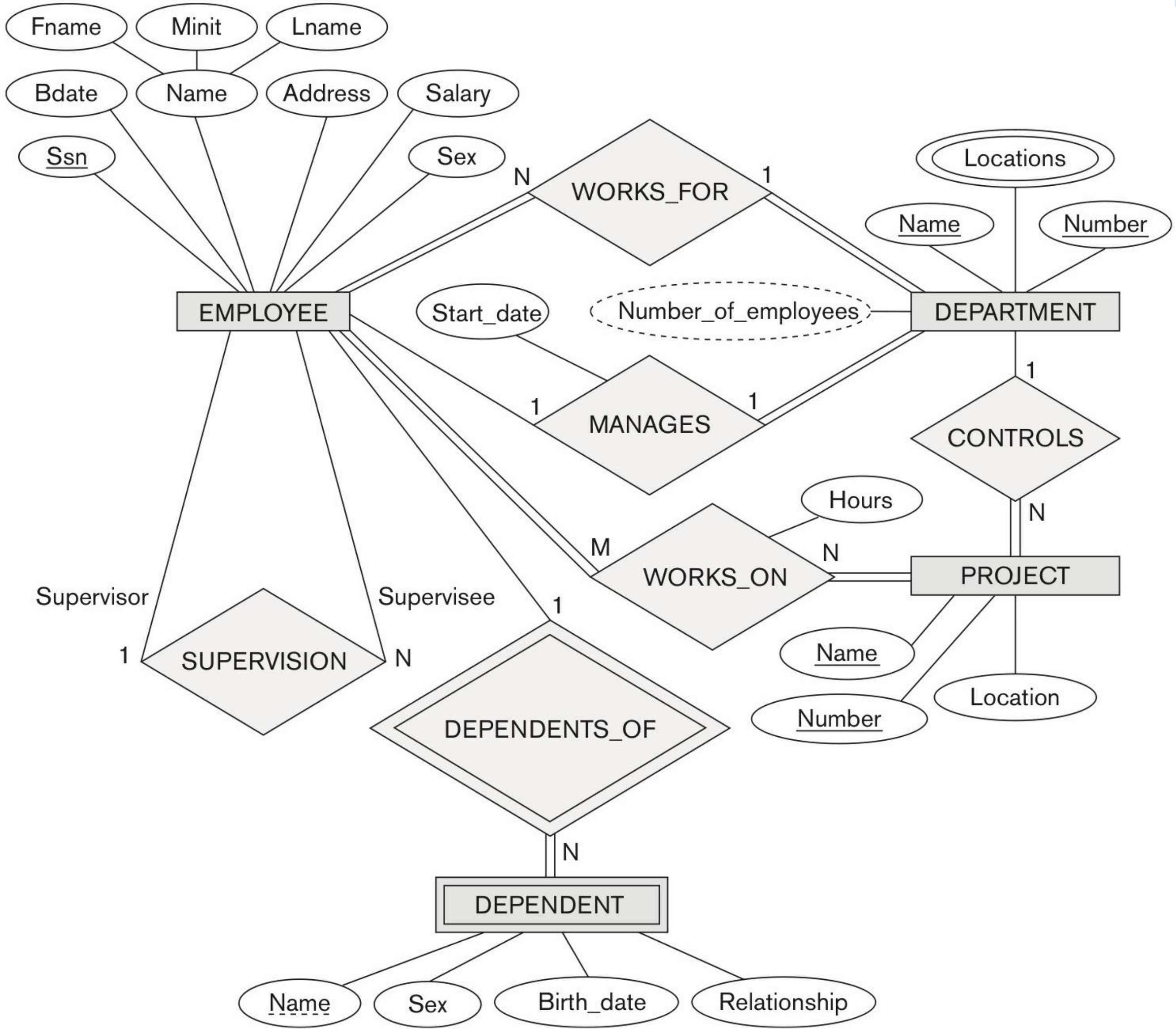
- Entity type: rectangle
- Attribute type: oval, double-lined, dashed-lined
  - Key: underlined



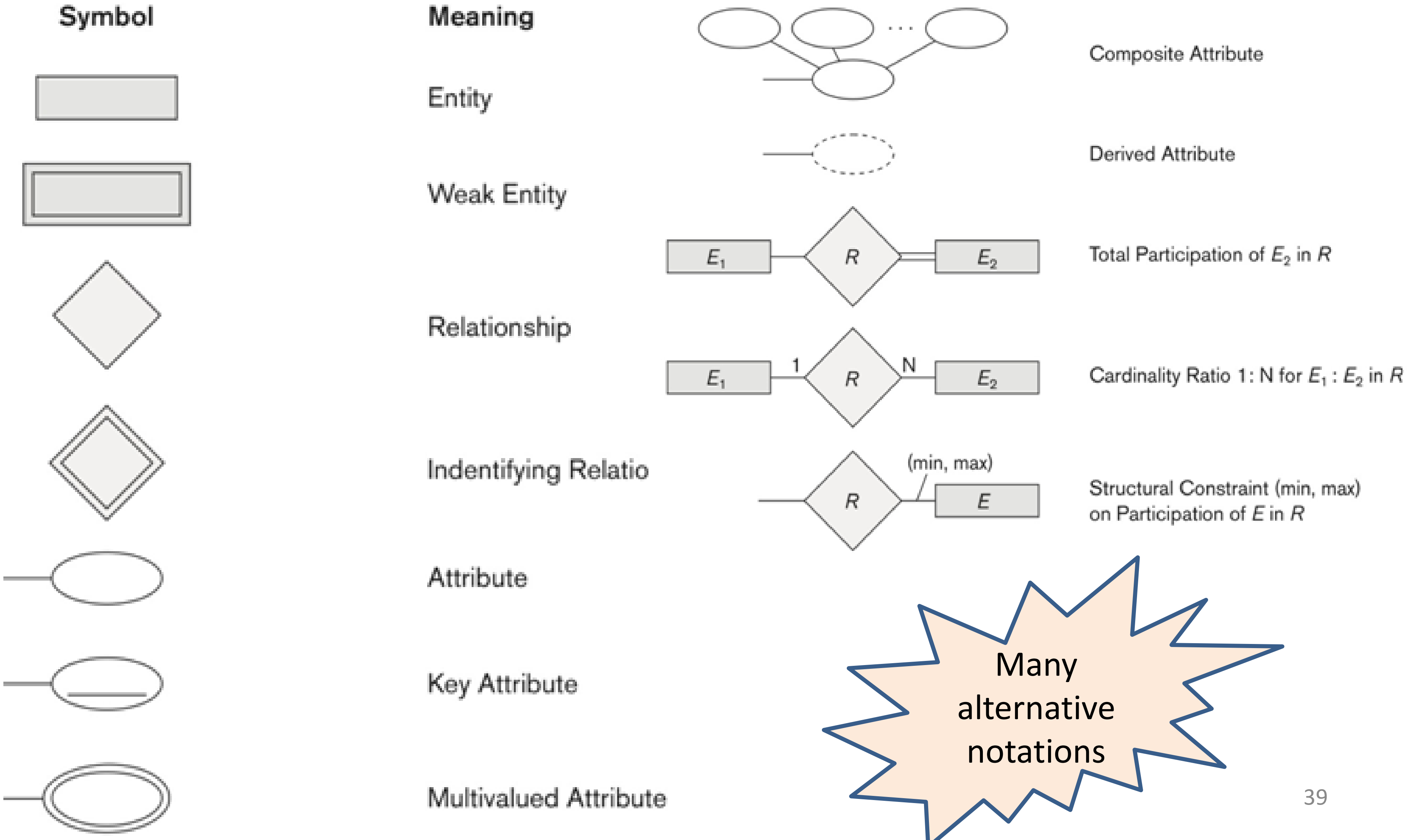
Revising Attributes to Relationships in Initial Design of COMPANY

- MANAGES
  - 1:1 relationship between EMPLOYEE and DEPARTMENT
  - Participation: EMPLOYEE: partial, DEPARTMENT: total
- WORKS_FOR
  - 1:N relationship type between DEPARTMENT and EMPLOYEE
  - Total participations for both
- CONTROLS
  - 1:N relationship type between DEPARTMENT and PROJECT
  - Participation: PROJECT: total, DEPARTMENT: partial
- SUPERVISION
  - 1:N relationship type between EMPLOYEE (supervisor role) and EMPLOYEE (supervisee)
  - Partial participations for both
- WORKS_ON
  - M:N relationship type with attribute Hours between EMPLOYEE and PROJECT
  - Total participations for both
- DEPENDENTS_OF
  - 1:N relationship type between EMPLOYEE and DEPENDENT
  - Identifying relationship for weak entity type DEPENDENT
  - Participation: EMPLOYEE: partial, DEPENDENT: total

ER Diagram of COMPANY



Notation of ER Diagram – Peter Chen



Many alternative notations

Proper Naming of Schema Constructs

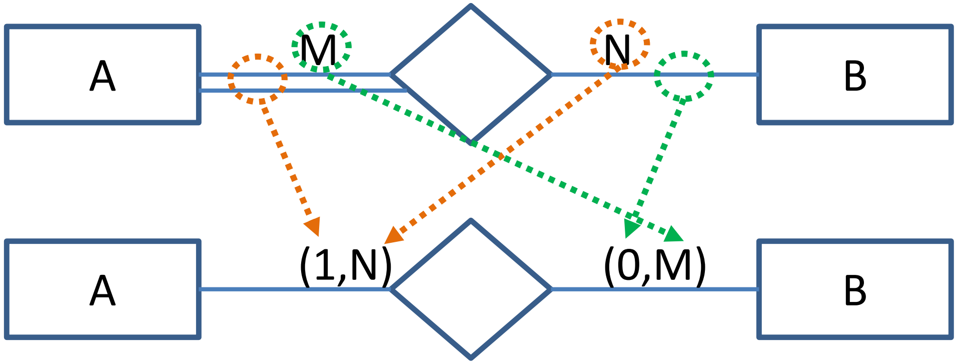
- Name - convey meanings attached to different constructs in schema
  - Entity type: noun
  - Relationship type: verb
- ER diagram readable from left to right and from top to bottom

Design Choices for ER Conceptual Design

- Model concept first as an attr
  - Refined into a relationship if attr is a reference to another entity type
- Attr exists in several entity types → elevated to independent entity type
  - E.g., UNIVERSITY has STUDENT, INSTRUCTOR, and COURSE, each has an attr Department → Promote to entity type DEPARTMENT
- An independent entity type may be demoted to attr
  - E.g, an entity type DEPARTMENT (w/ single attr Dept_name) is related to only one other entity type STUDENT → Demote the entity type to attr

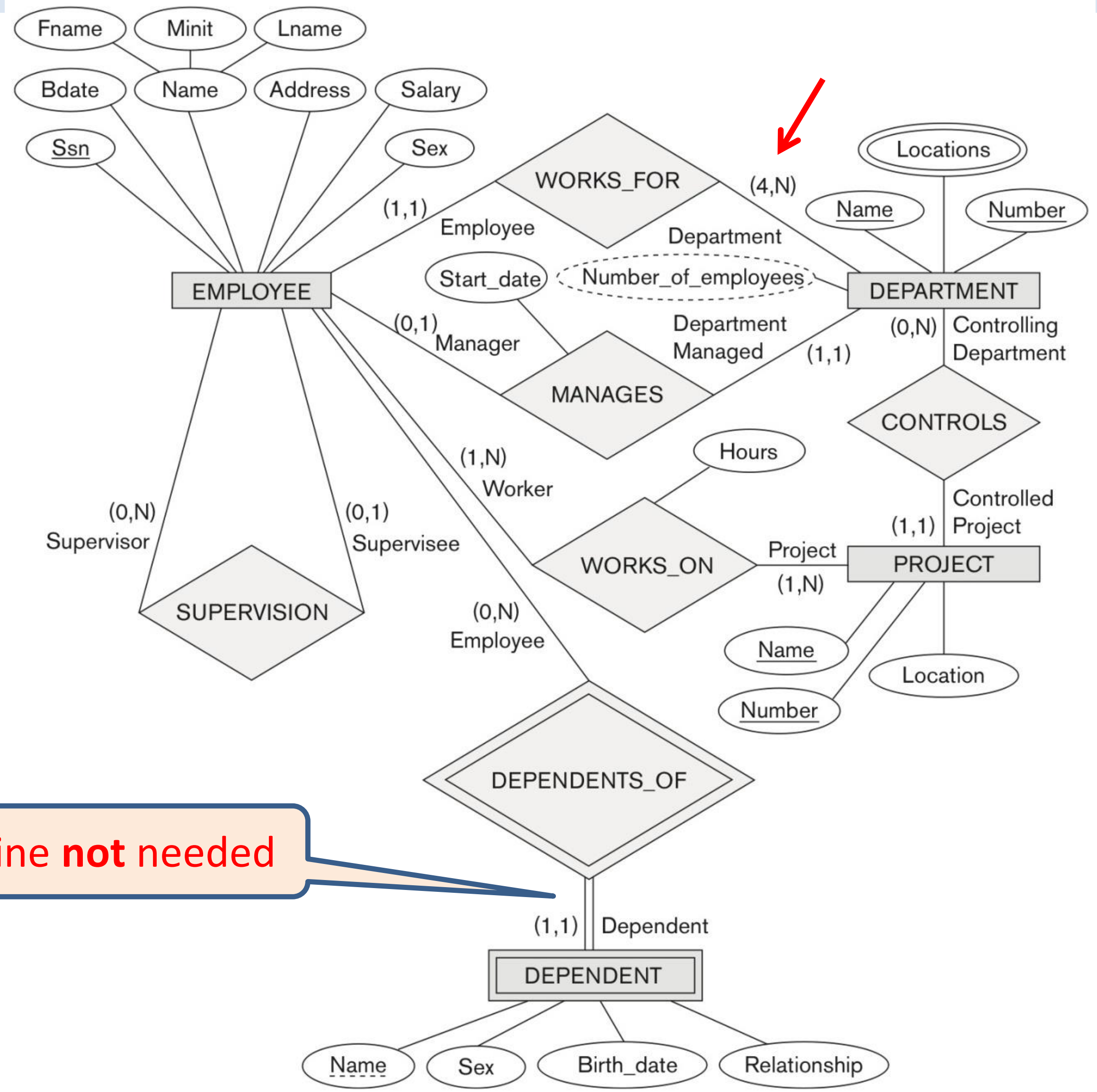
Alternative Notations for ERDs

- **(min, max) notation** on relationship type
  - Replaces cardinality ratio (1:1, 1:N, M:N) and single/double line (participation constraints)
  - Conversion between these two notations



  - (min, max) on the "other side" of entity
    - UML: same side of entity
  - Expressive capability of requirements vs regular ERD
- ERD: ***single*** consistent notation (regular or min-max)

COMPANY: (min, max) Notation



Double line **not** needed

Other Notation: Unified Modeling Language (UML)

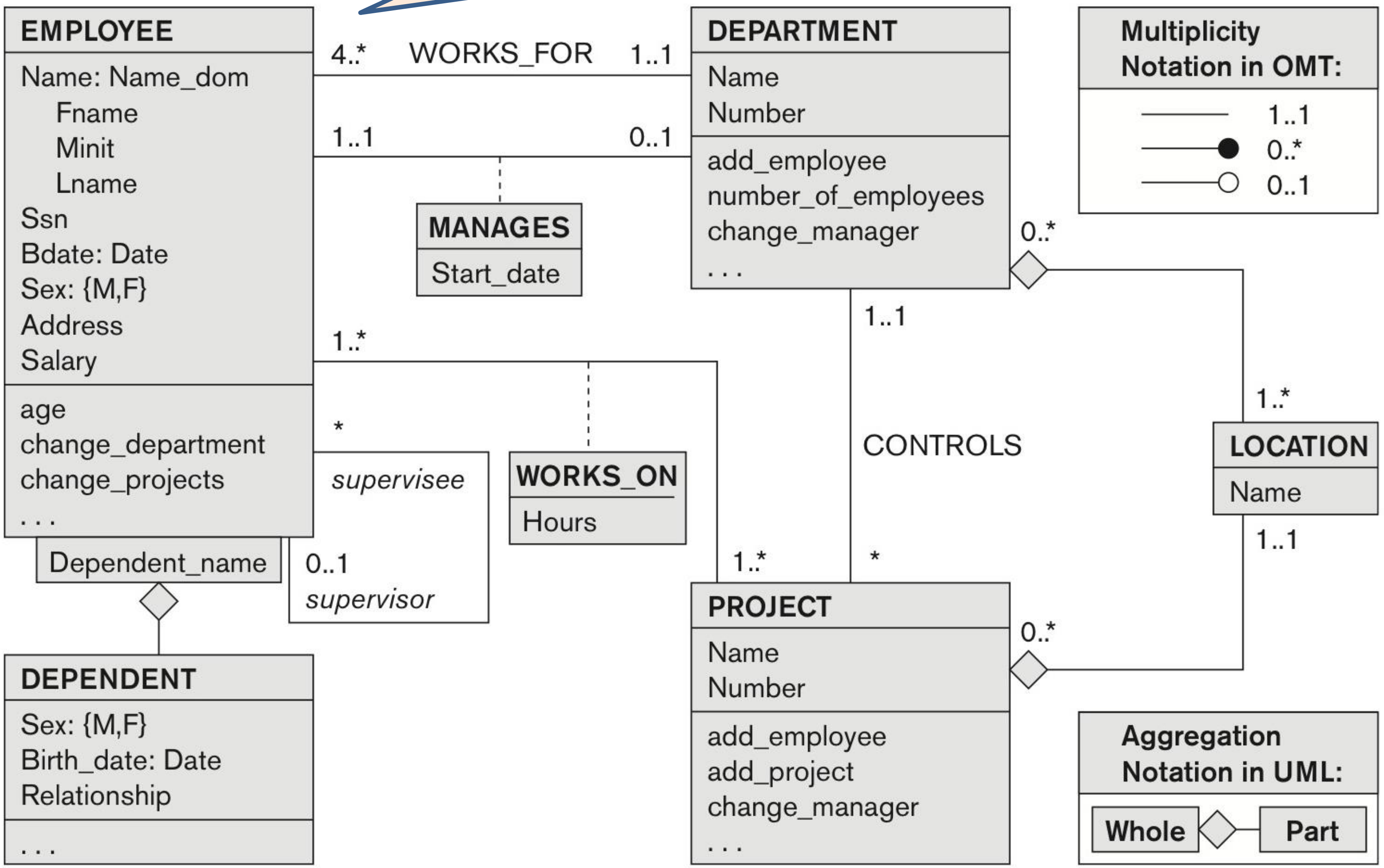
- UML methodology
  - Used extensively in software design
  - Many types of diagrams for various software design purposes
- ER diagram ➔UML class diagrams
  - Entity ➔ class (top: name, middle: attributes, bottom: operations)
  - Relationship ➔ association, aggregation
  - Relationship instance ➔ link
  - Relationship attribute ➔ link attribute
  - (min, max) ➔ multiplicities min..max
    - Multiplicities (and role names) are placed *on the opposite ends of the relationship*
  - Recursive relationship ➔ reflexive association
  - **unidirectional** and **bidirectional** associations/aggregations
  - Weak entity ➔ qualified association (or aggregation)

Functional requirements

*: no max limit on participation
: 0..
1: 1..1

UML Class Diagram for COMPANY

vs (min,max) ➔ on different side

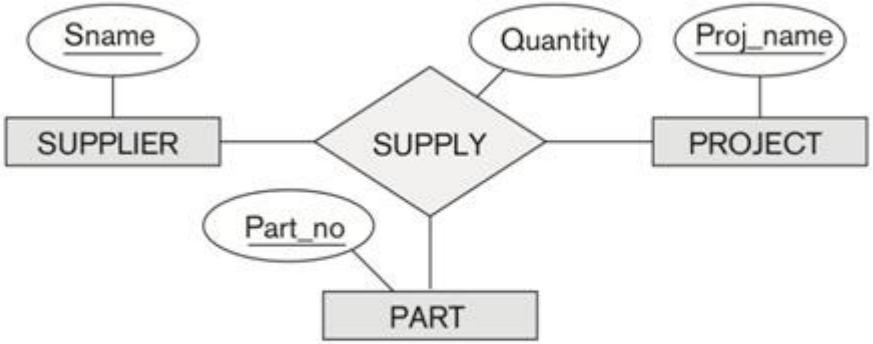


Relationship Types of Degree > 2

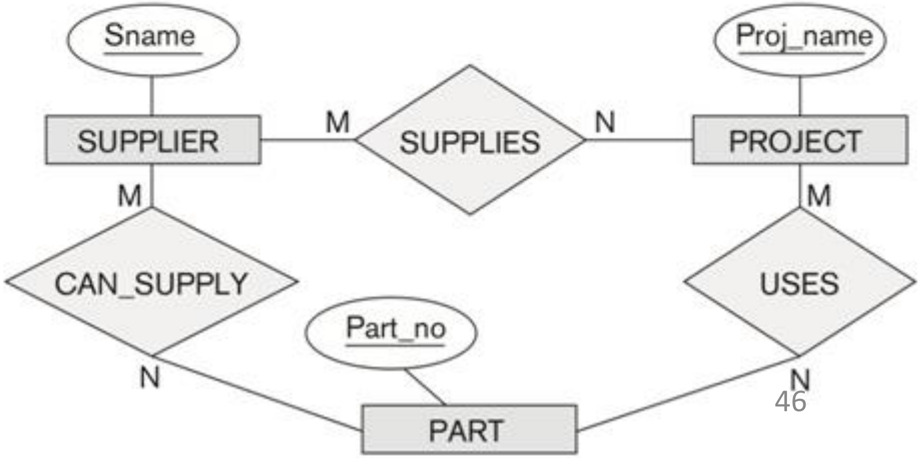
- **Degree** == # of participating entity types
- 2-way vs 3-way: How to choose?

- (a) != (b)
- 3-way relationship in (a) == subset of 3 binary relationships in (b)
- Designer decides based on semantics or meaning to be represented

(a)

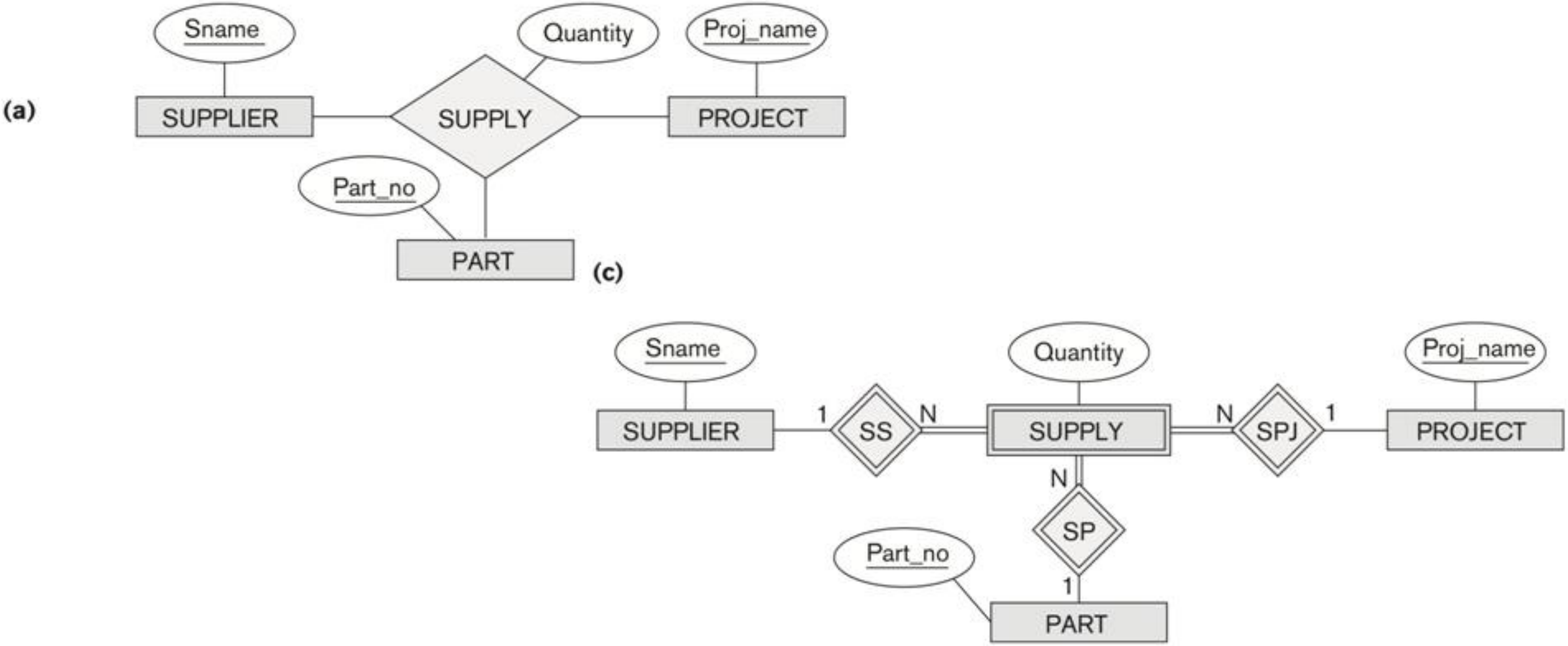


(b)



Relationship Types of Degree > 2 (cont’d)

- Some tools permit only binary relationships
  - Ternary relationship ➔ weak entity type, e.g. 3.17(c)
    - *Three* identifying relationships (no partial key, in this example)

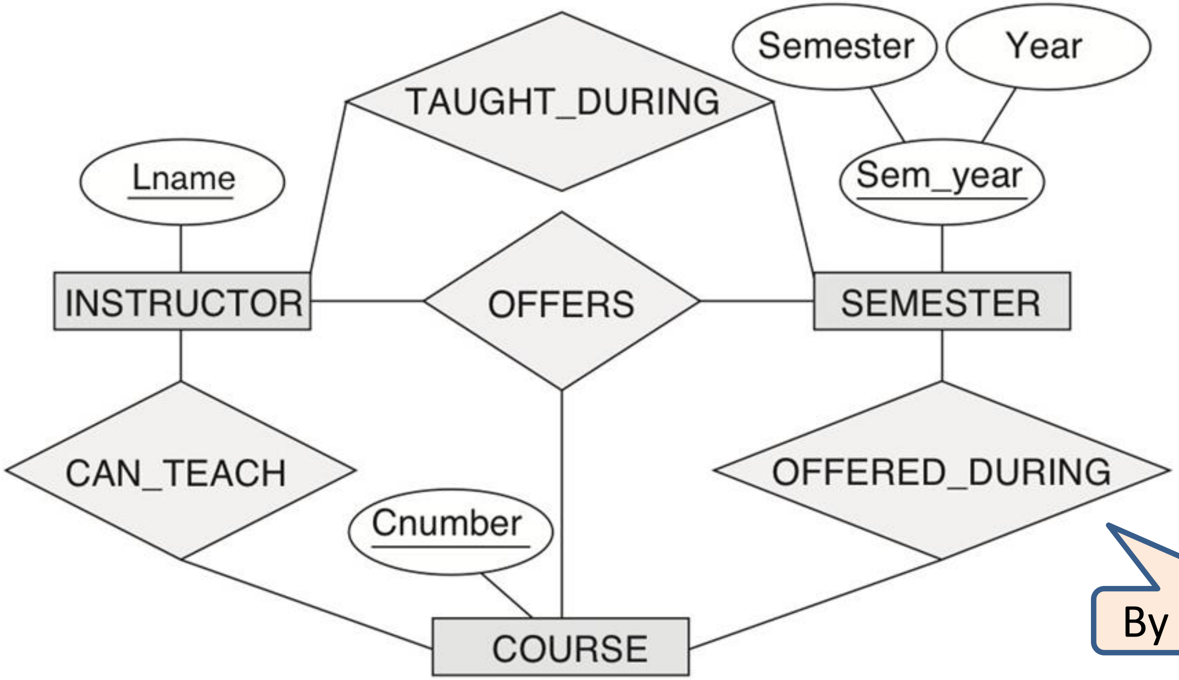


  - Ternary relationship ➔ regular entity type
    - Add a surrogate key, e.g., supply_id

Relationship Types of Degree > 2 (cont'd)

- Which one(s) are redundant?

Taught some course

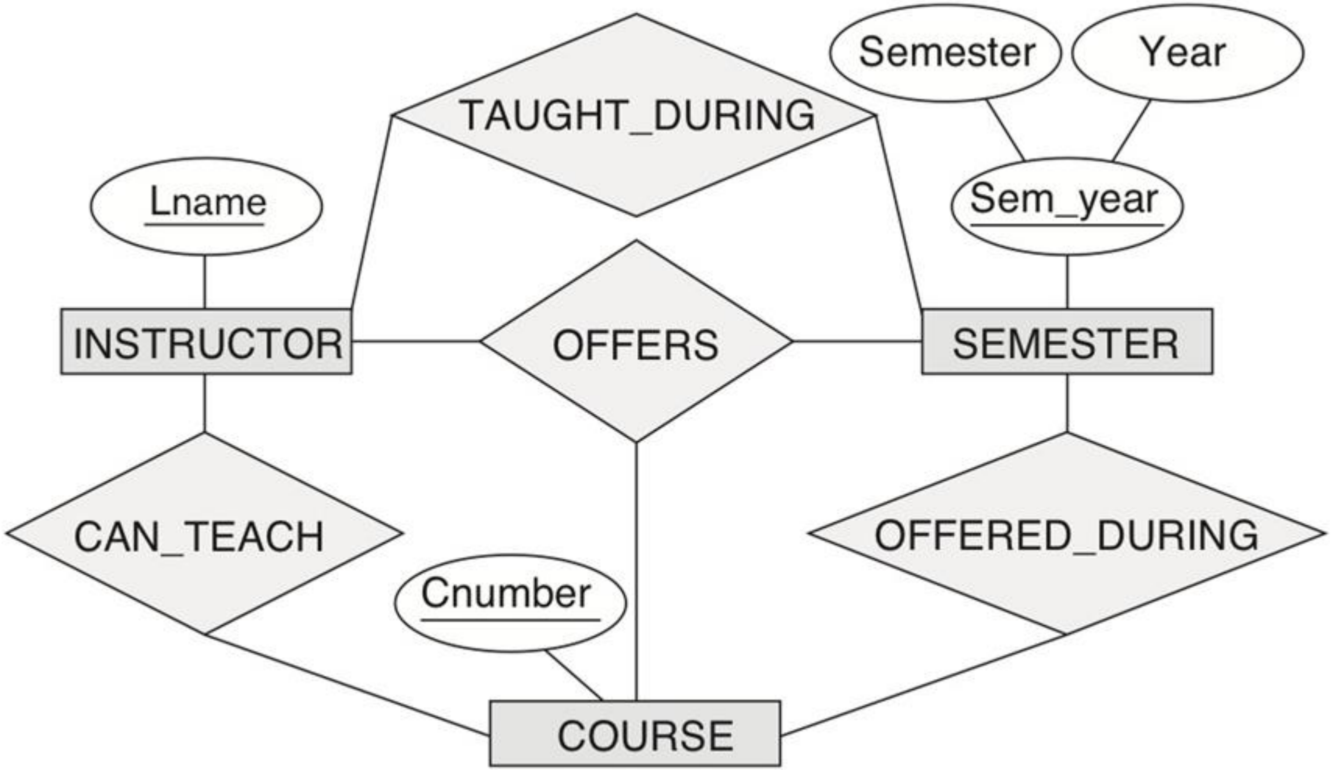


By any instructor

  - OFFERS vs three binary relationships
    - "OFFERS" is a subset, intersection of 3 binary relationships
    - Instances in OFFERS should not exist *unless* corresponding instance also exists in three binary relationships
      - The reverse may not be true

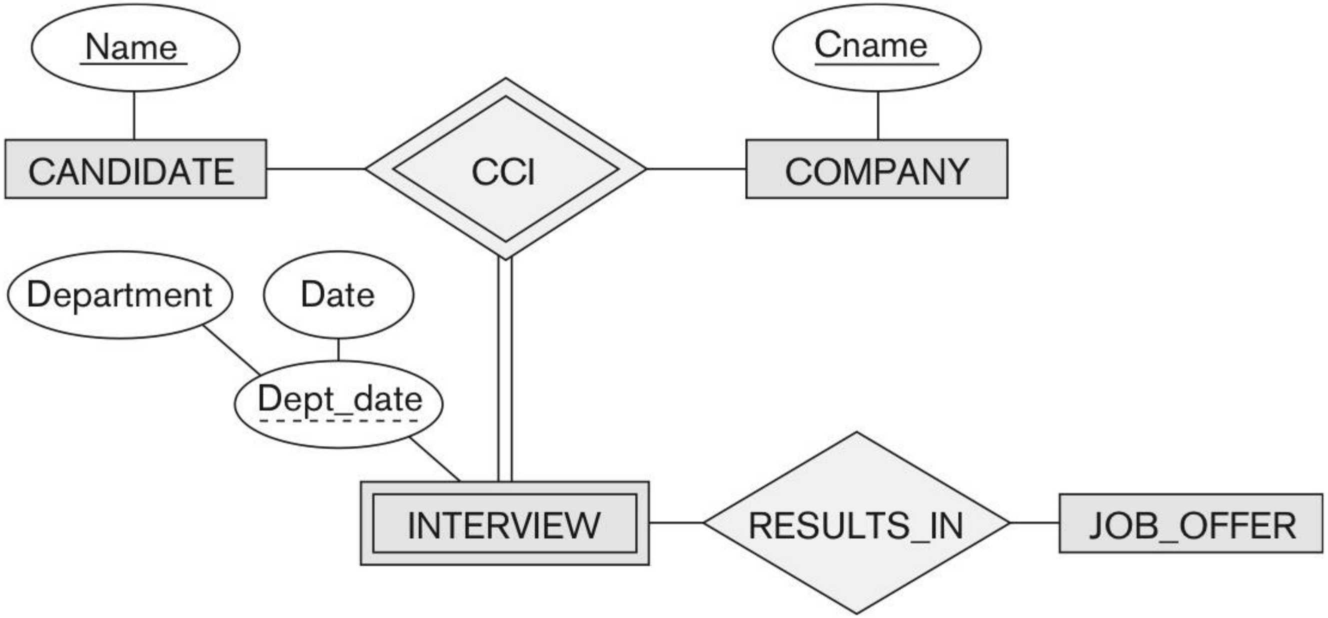
Relationship Types of Degree > 2 (cont'd)

- In general three binary relationships *cannot* replace a ternary relationship
  - they may do so under certain *additional constraints*
    - If CAN_TEACH relationship is 1:1, OFFERS can be left out



Relationship Types of Degree > 2 (cont’d)

- weak entity w/ a ternary (or n-ary) identifying relationship type



  - INTERVIEW: two owner entity types
    - Dept_date: partial key
  - A candidate can interview a company multiple times

Relationship != Operation

- Relationship in ERD specifies data association, e.g., how entities are related to one another
- Operation is specified in functional spec
- ERD specifies data requirements/aspect, not operations
  - ERD provides data model upon which operations are operated
  - Do not mix operations with relationships
    - Boundary may not be clear
    - No need to draw operations as relationships in ERD
  - Operations should be tracked separately, perhaps with the help from UML

ERD Tools: Peter Chen's Notation

free

- Dia (desktop: linux, windows, mac): choose "ER"
  - http://dia-installer.de/
  - https://www.youtube.com/watch?v=JoVwansiTkM&list=PLB65E97F582E33BFA

.dia file

- yED (desktop: linux, windows, mac): choose "Entity Relationship"
  - https://www.yworks.com/products/yed
  - "partial key" workaround: https://yed.yworks.com/support/qa/7885/update-please-need-this-feature-partial-key-representation

.graphml file

- https://www.draw.io (web-based): choose "Entity Relation", or "software/entity_relationship.xml"

.xml file

- https://www.lucidchart.com (web-based, sjsu): Shapes → "UML/UML Entity Relationship" (not "Entity Relationship"). Download as (not export)

Visio .vdx file

- Visio + www.visiocafe.com/downloads/various/DanielHarris/Chen_ER.zip

- Many tools supporting alternative ERD notations (crow's foot)

Common Issues w/ ERD

- Strong vs weak entity
- Missing identifying relationship
- Partial vs total participation
- Missing key or partial key
- Missing cardinality
- Explicit value in cardinality if value is known
- Flipped (min, max)
- No double line in (min, max) notation
- Redundant attr vs relationship

- Missing attr or entity
- Single vs multi-value attr
- Single consistent notation (regular or min-max) per ERD
- Avoid cross line

- Illegible or blurry screenshot of ERD
  - **Export** to JPG/BMP and then insert as picture to your answer file (actual ERD is viewable)
- Do not know how to view annotated comments on Canvas

Summary

- What are Data modeling (database design) steps?
- Entity: rectangle
  - Strong vs weak (double-lined rectangle)
- Attribute: oval
  - Key (underline oval)
  - Partial key (dashed underline)
  - Single vs multi-valued (double-lined oval)
  - Simple vs composite
- Relationship: diamond
  - Total vs partial
  - Cardinality
  - (min, max)
    - Expressive capability
  - Identifying (double-lined diamond)

Self Exercises

- 7/E: Exercise 3.16, 3.19, 3.21, 3.22, 3.24, 3.28
- 6/E: Exercise 7.16, 7.19, 7.21, 7.22, 7.24, 7.28