Lecture 1: Entity-Relationship (ER) Model

CS3402 Database Systems

ER Model

- An entity–relationship model (ER model for short) describes interrelated things of interest in a specific domain of knowledge. A basic ER model is composed of entity types (which classify the things of interest, e.g., employee and department) and specifies relationships that can exist between entities (instances of those entity types).

- An ER model is commonly formed to represent things that a business needs to remember in order to perform business processes. Consequently, the ER model becomes an abstract data model, that defines a data or information structure which can be implemented in a database, typically a relational database.

Entity, Entity Type and Entity Set (1/2)

- An **entity** is defined as a thing capable of an independent existence that can be uniquely identified and exists either physically or logically. For example, a person exists physically while an order transaction exists logically; they both can be uniquely identified. Entities can be thought of as nouns. In relational databases, an entity refers to a single tuple.

- An **entity type** defines a collection of entities that have the same attributes.

- The collection of all entities of a particular entity type in the database at any point in time is called an **entity set**.

3

Entity, Entity Type and Entity Set (2/2)

- The CAR entity with two key attributes, Registration and Vehicle_id, one composite attribute, Registration, three simple attributes, Make, Model and Year, and one multivalued attribute,
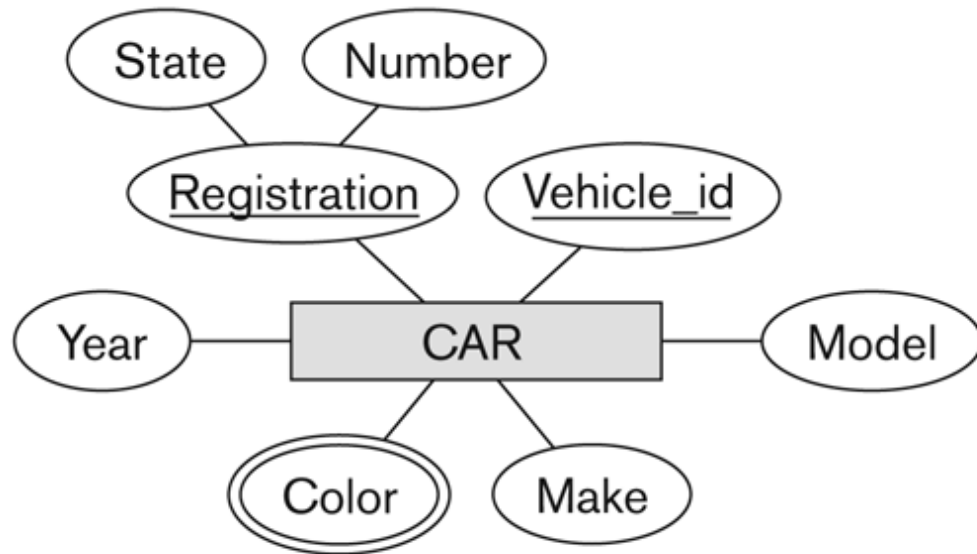
Entity type

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

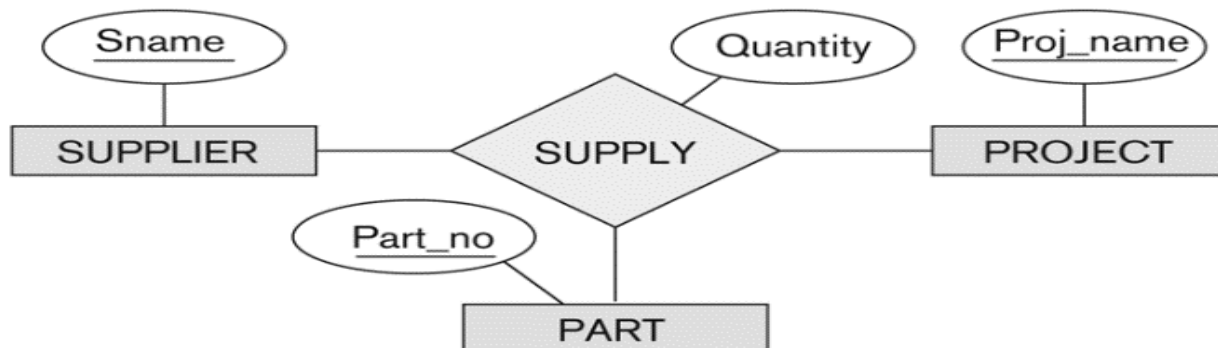# Relationship, Relationship Type and Relationship Set (1/2)

- A **relationship** among two or more entities represents an association among two or more entities, for example, a works-on relationship between an employee and a project.

- A **relationship type** among entity types defines a set of associations among entities from these entity types.

- A **relationship set** is a collection of relationship instances all belonging to one relationship type represented in the database. For example, if a relationship type is registration, each enrolment of a student in a course is an instance of registration and appears is in the relationship set.

Relationship, Relationship Type and Relationship Set (2/2)

- Binary relationship WORK_FOR between EMPLOYEE and DEPARTMENT entities

- Ternary relationship SUPPLY among SUPPLIER, PROJECT and PART entities
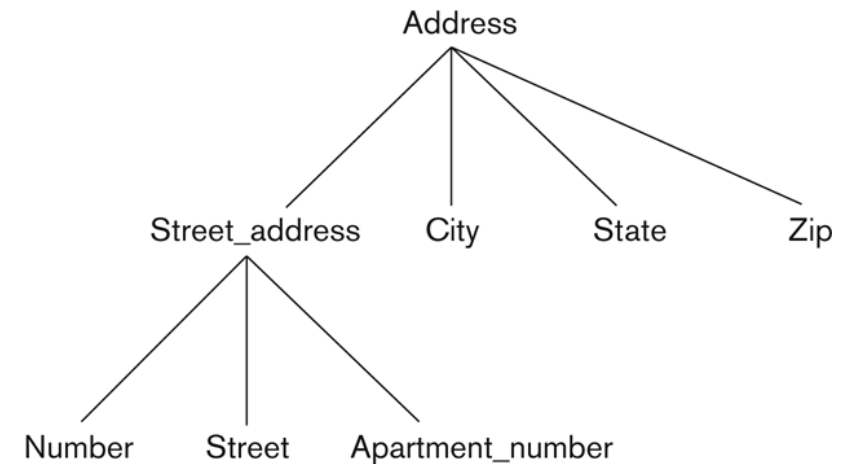
Attributes

- An **attribute** represents some property of interest that further describes an entity, such as the employee's name or salary.

- Both entities and relationships can have attributes. For example, an employee entity has a Social Security Number (SSN) (like HK card ID number) attribute, while a work_for relationship has a start date attribute.

- A **key attribute** is a set of attributes (one or more attributes) that uniquely identify an entity (i.e., no two entities have the same value(s) in the key attribute).

- For example, SSN is a single key attribute for employees. University ID and student ID are composite attributes to identify a university student in a country because student IDs may not be unique among universities in a country.

Types of Attributes (1/2)

- A **<u>simple attribute</u>** has a single atomic value that does not contain any smaller meaningful components. For example, SSN and gender.

- A **<u>composite attribute</u>** is composed of several components. For example, address contains flat, block, street, city and country. Composition may form a hierarchy where some components are themselves composite.

- A **<u>multi-valued attribute</u>** has multiple values. For example, color of a product (i.e., red and white) and major of a student (i.e., computer science and mathematics).

The hierarchy of composite attributes.

Types of Attributes (2/2)

- In general, composite and multi-valued attributes may be nested to any number of levels although this is rare. For example, a person can have multiple postal addresses (i.e., composite multi-valued attribute).

- A **derived attribute** is an attribute who value is calculated from other attributes. The derived attribute need not be physically stored within the database; instead, it can be derived by using an algorithm. For example, the number of employees of a department can be calculated by counting the number of employees associated with the department.

Value Sets (Domains) of Attributes

- Each simple attribute is associated with a value set (or domain).

- The value set specifies the set of values associated with an attribute. For example, date has a value of MM-DD-YYYY, where each letter is an integer, course grade has a value of {A+, A, A-, B+, B, B-, C+, C, C-, D, F}, and name is a string up to 100 characters.

- Value sets are similar to data types in most programming languages, e.g., integer, characters, float, double, and boolean.

Participation Constraints on Relationships

- Participation constraint indicates the minimum number of relationship instances that an entity can participate in.

- A **total participation** requires that each entity is involved in the relationship. In other words, an entity must exist related to another entity, i.e., existence dependency. Total participation is represented by double lines in ER model. For example, every employee must work for a department, i.e., the participation of employee in work_for relationship is total.

- A **partial participation** means that not all entities are involved in the relationship. Partial participation is represented by single lines in ER model. For example, some employees manage departments, i.e., the participation of employee in manage relationship is partial.

Cardinality Constraints on Relationships (1/5)

- Cardinality ratio indicates the maximum number of relationship instances that an entity can participate in. Consider two entity sets S and T,
  - A 1:1 or **one-to-one relationship** from S to T is a cardinality type in which an entity from S is related to at most one entity from T and vice versa.
  - An N:1 or **many-to-one relationship** from S to T is a cardinality type in which two or more entities from S can be related to an entity from T.
  - A 1:N or **one-to-many relationship** from S to T is a cardinality type in which an entity from S can be related to two or more entities from T.
  - An N:M or **many-to-many relationship** from S to T is a cardinality type in which an entity from S can be related to two or more entities from T, and an entity from T can be related to two or more entities from S.

Cardinality Constraints on Relationships (2/5)

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

- WORKS_FOR N:1 relationship between EMPLOYEE and DEPARTMENT, i.e., many employees work for the same department.
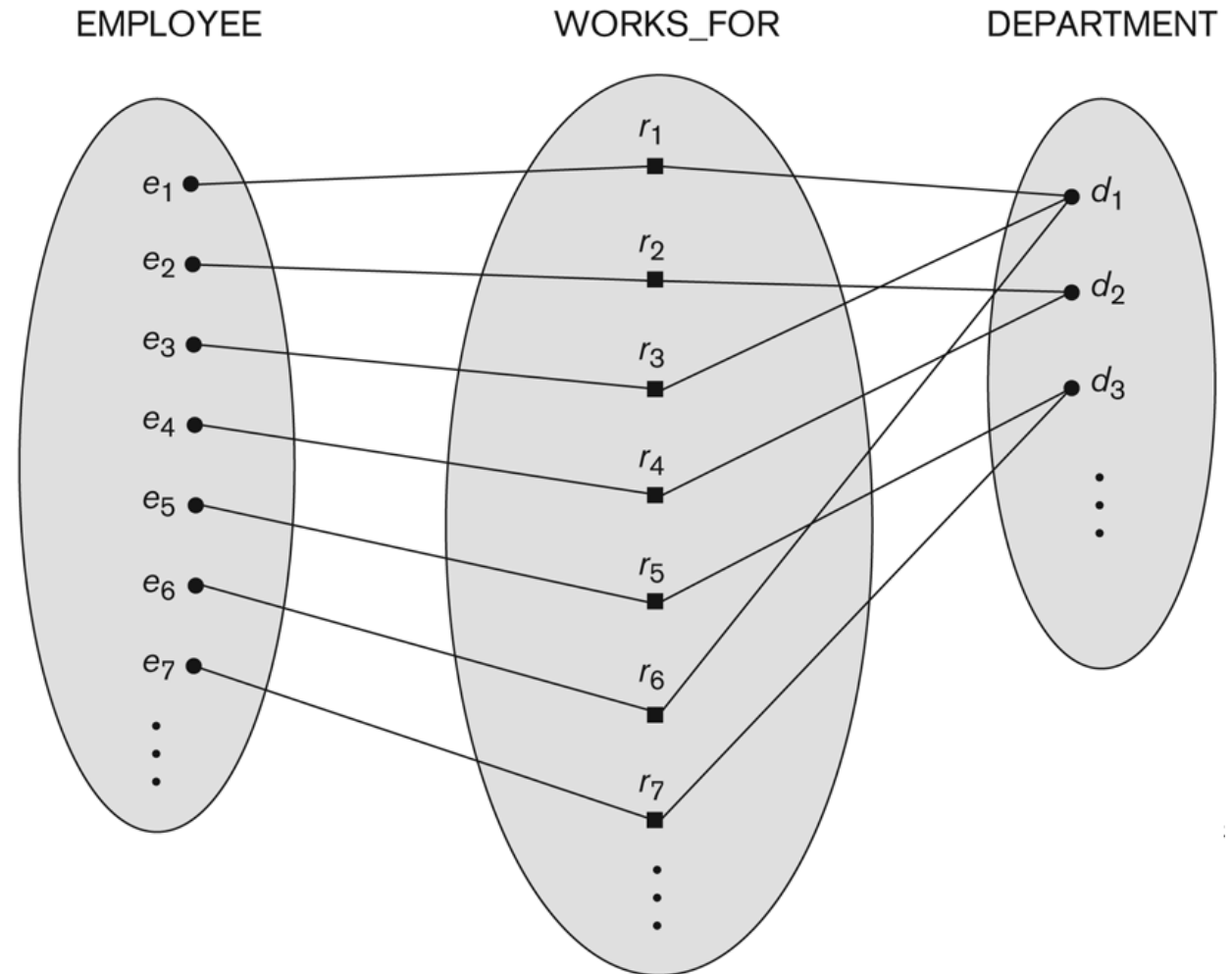
Cardinality Constraints on Relationships (3/5)

- Some instances in the WORKS_ON relationship set, which represents a relationship type WORKS_ON between EMPLOYEE and PROJECT.

- WORKS_ON M:N relationship between EMPLOYEE and PROJECT, i.e., many employees work on many projects.
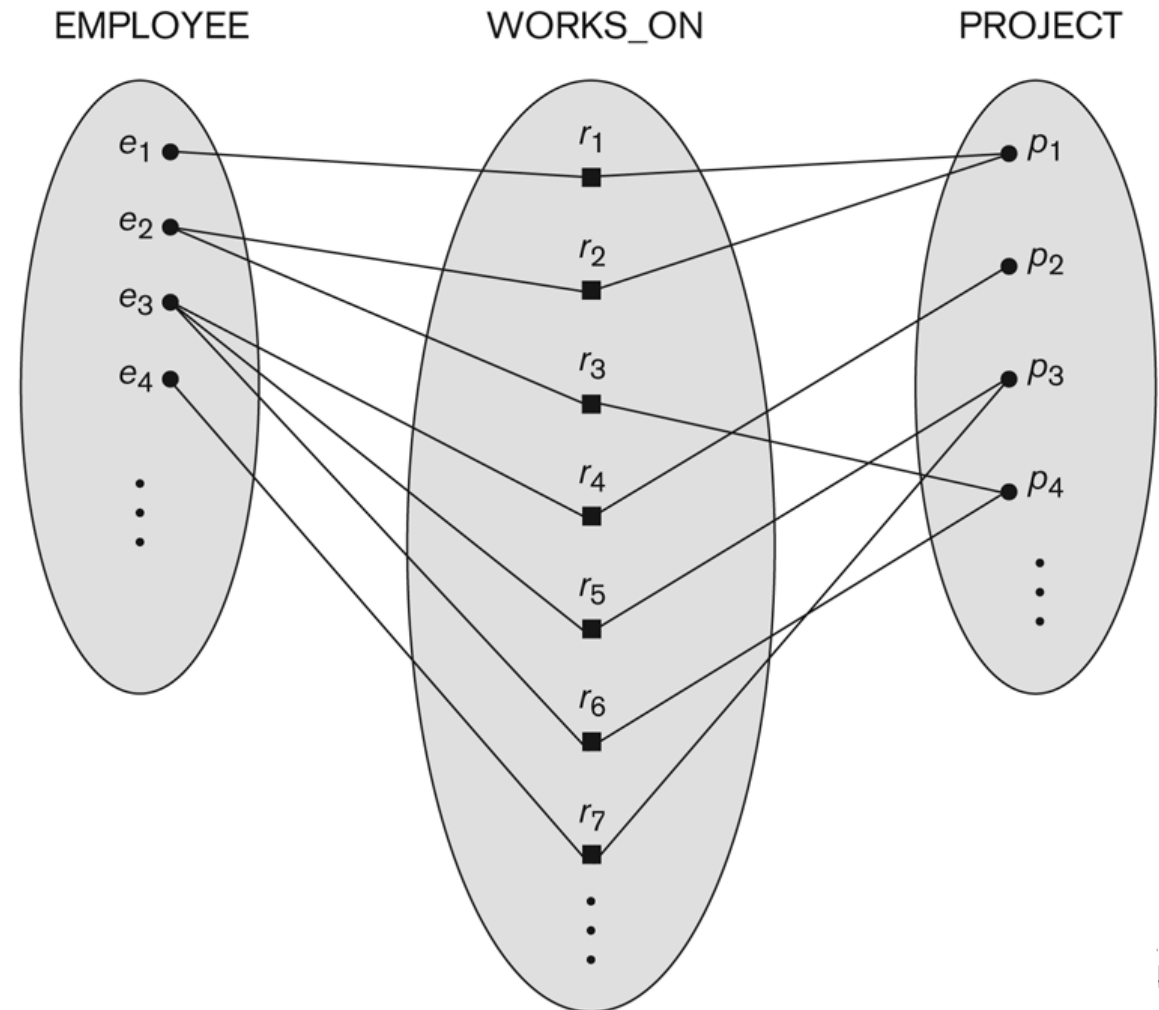
Cardinality Constraints on Relationships (4/5)

- (min, max) notation for relationship structural constraints
  - This notation specifies that each entity participates in at least min and at most max relationship instances in a relationship.
  - min must be at least 0 and at most max (0 <= min and min <= max)
  - max must be at least 1 (max >= 1)

Cardinality Constraints on Relationships (5/5)

One EMPLOYEE may manage one DEPARTMENT at most

One DEPARTMENT is managed by one EMPLOYEE

One EMPLOYEE works for one DEPARTMENT

One DEPARTMENT has 1 to many EMPLOYEEs

Recursive Relationship Type (1/2)

- A recursive relationship is one in which the same entity participates more than once in the relationship. The relationship should be marked by the role that an entity takes in the participation.

- It is also called a self-referencing relationship type.

- Suppose one employee is assigned the task of supervising the other employees. The supervision relationship is a recursive relationship because the same entity, a particular employee, participates more than once in the relationship, as a supervisor and as a supervisee.
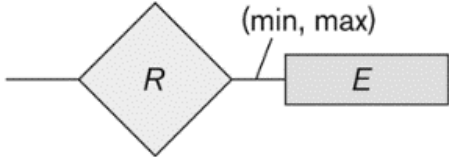
Recursive Relationship Type (2/2)
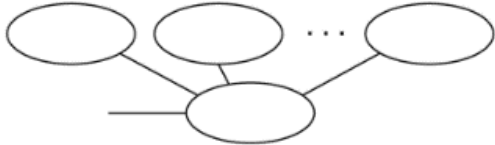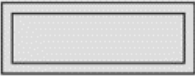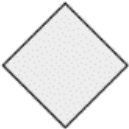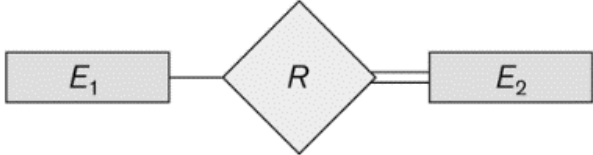
- A recursive relationship SUPERVISION between EMPLOYEE in the supervisor role (1) and EMPLOYEE in the supervisee role (2).

- e_1 is the supervisee of e_5 through relationship instance r_1 and is the supervisor of e_2 and e_3 through r_2 and r_3, respectively.
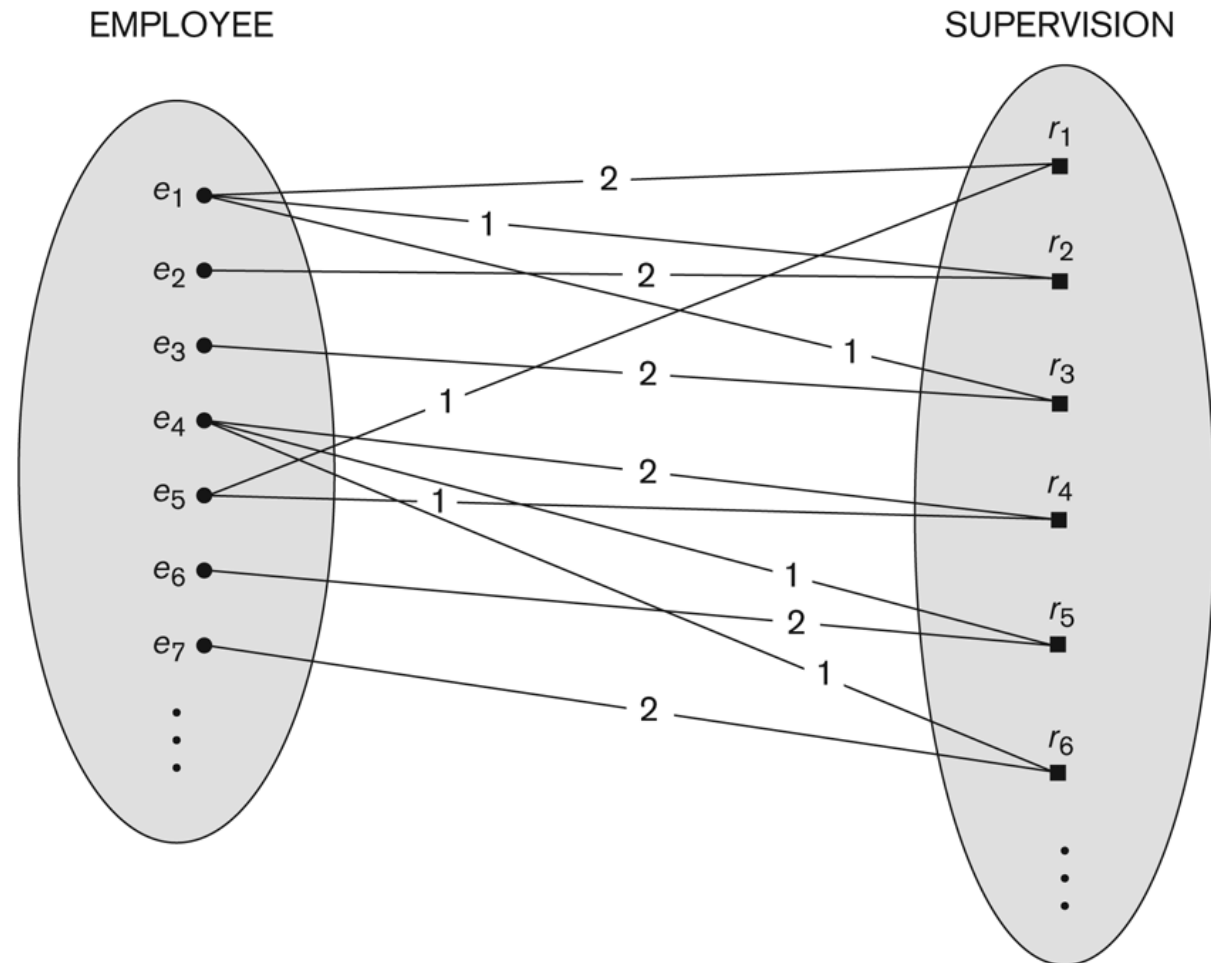
Weak Entity Types

- A weak entity that does not have a key attribute and is identification-dependent on another entity type. It must participate in an identifying relationship type with an owner or identifying entity type. In other words, weak entity type must be owned by some owner entity type.

- A weak entity is identified by the combination of: (1) its partial key and (2) the identifying entity type related to the identifying relationship type.

- For example,
  - Ada Chan is an employee. She has a dependent Cindy Chan.
  - Bob Chan is an employee. He has a dependent Cindy Chan.
  - The two dependent entities are identical.
  - The EMPYLOEE entity type owns the DEPENDENT entity type.

Summary of Notations for ER Diagrams

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

Composite Attribute

Derived Attribute

E_1 — R = E_2 Total Participation of E_2 in R

E_1 — 1 — R — N — E_2 Cardinality Ratio 1: N for E_1:E_2 in R

R — (min, max) — E Structural Constraint (min, max) on Participation of E in R
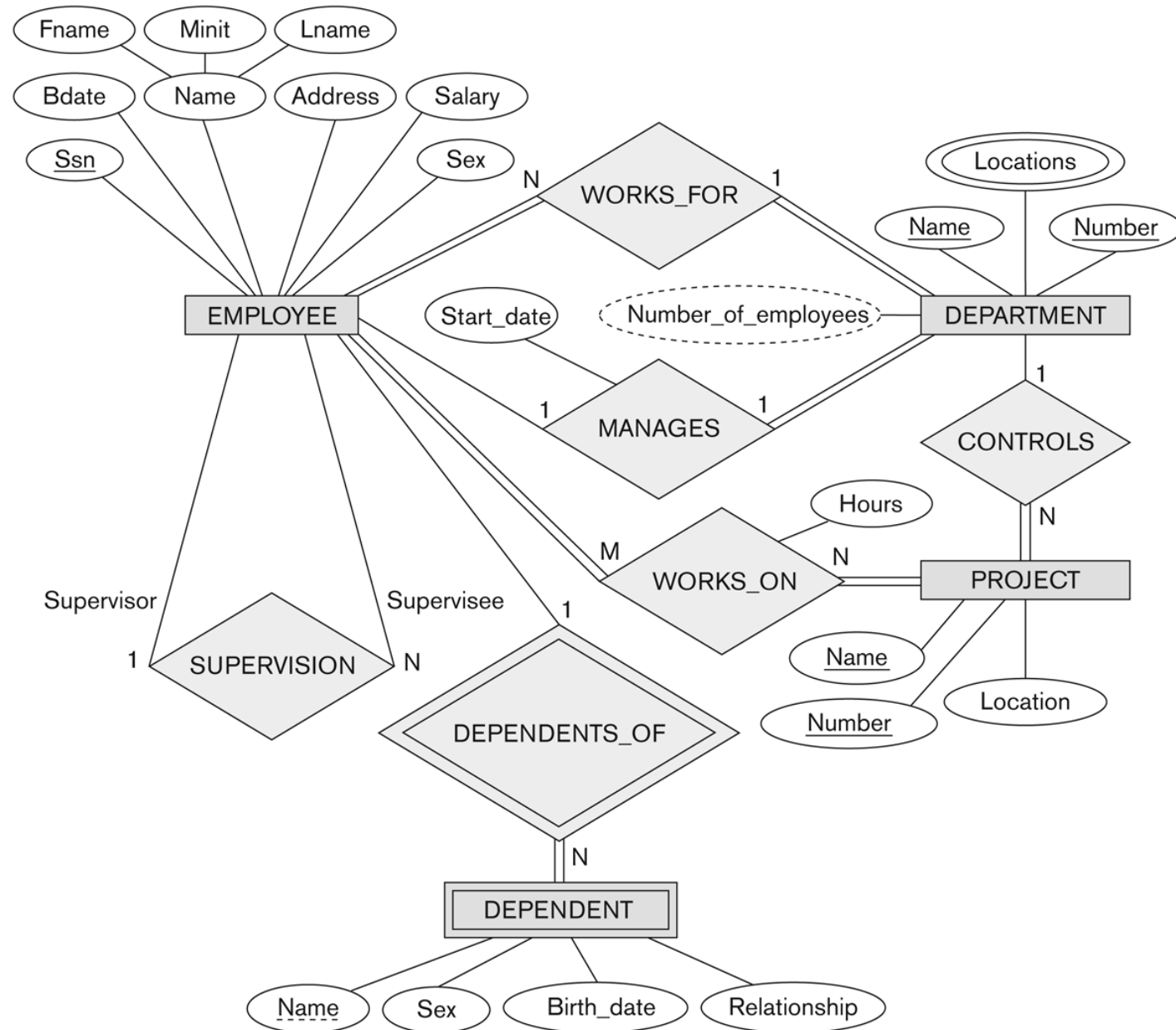
Case Study 1 (1/5)

- An ER diagram for the company database.

- 3 entities: EMPLOYEE, DEPARTMENT, and PROJECT

- 1 weak entity: DEPENDENT

- 4 relationships: WORKS_FOR, MANAGES, WORKS_ON, and CONTROLS

- 1 identifying relationship: DEPENDENTS_OF

- 1 recursive relationship: SUPERVISION
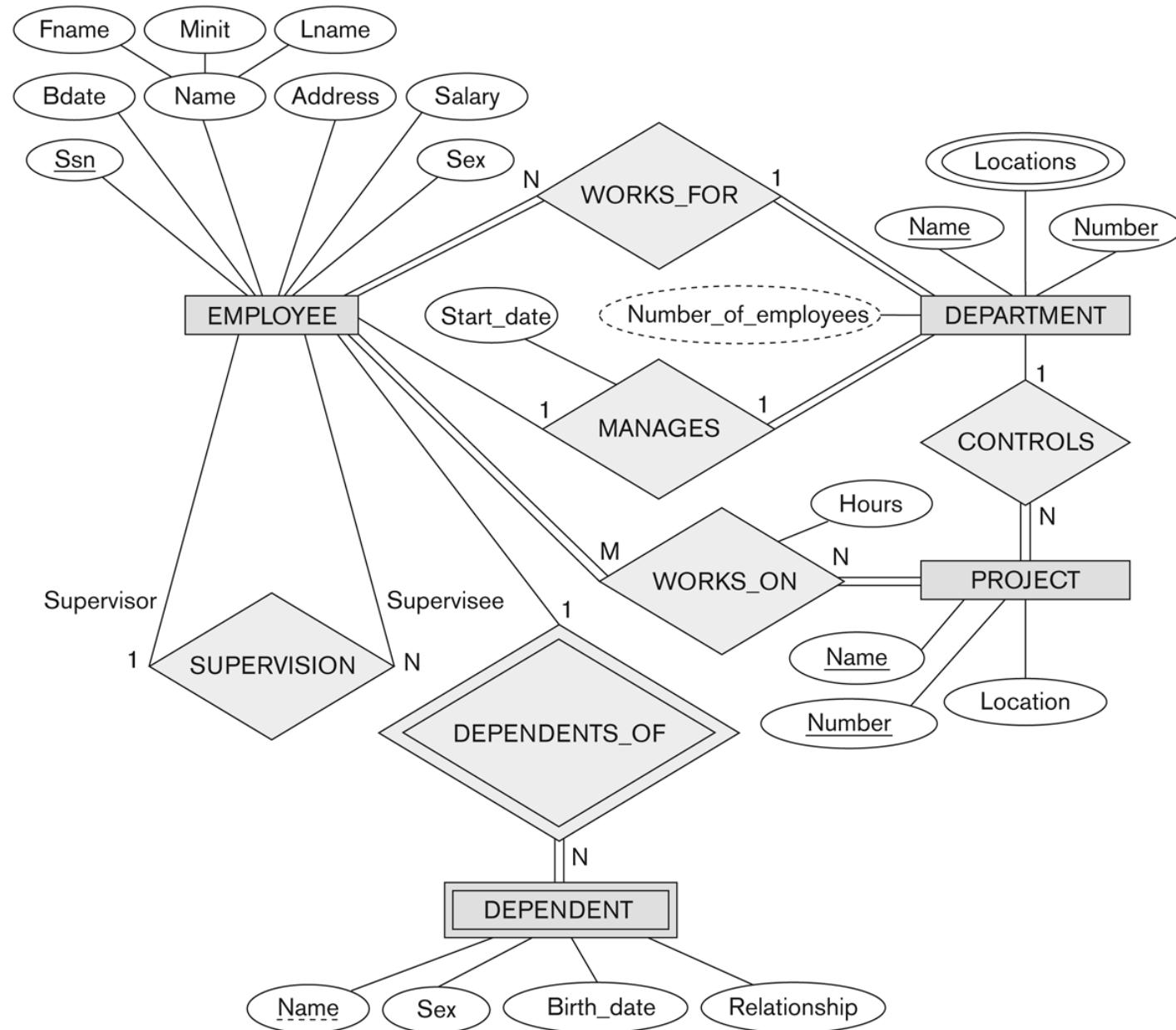
Case Study 1 (2/5)

- The company is organized into DEPARTMENTs

- Each DEPARTMENT has a unique name, unique number, many EMPLOYEEs and an EMPLOYEE who manages the DEPARTMENT.

- A DEPARTMENT may have several locations.

- We keep track of the start date of the department manager and the number of employees for each DEPARTMENT.
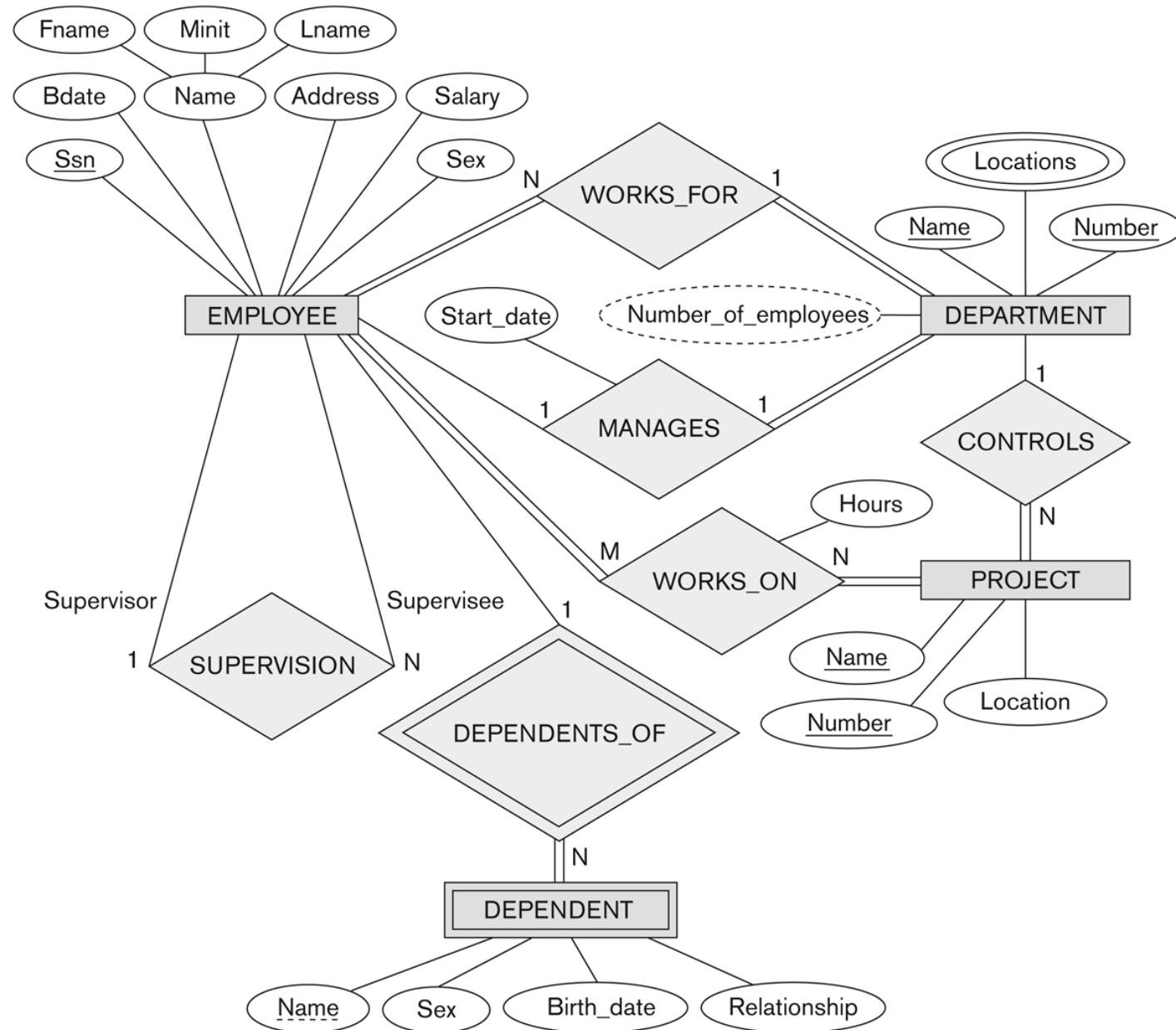
Case Study 1 (3/5)

- A DEPARTMENT controls a number of PROJECTs.

- Each PROJECT has a unique name, unique number and is located at a single location and is controlled by a DEPARTMENT.

- Each EMPLOYEE has social security number (Ssn), address, salary, sex, and birthdate. Ssn is a key attribute and name is an composite attribute.
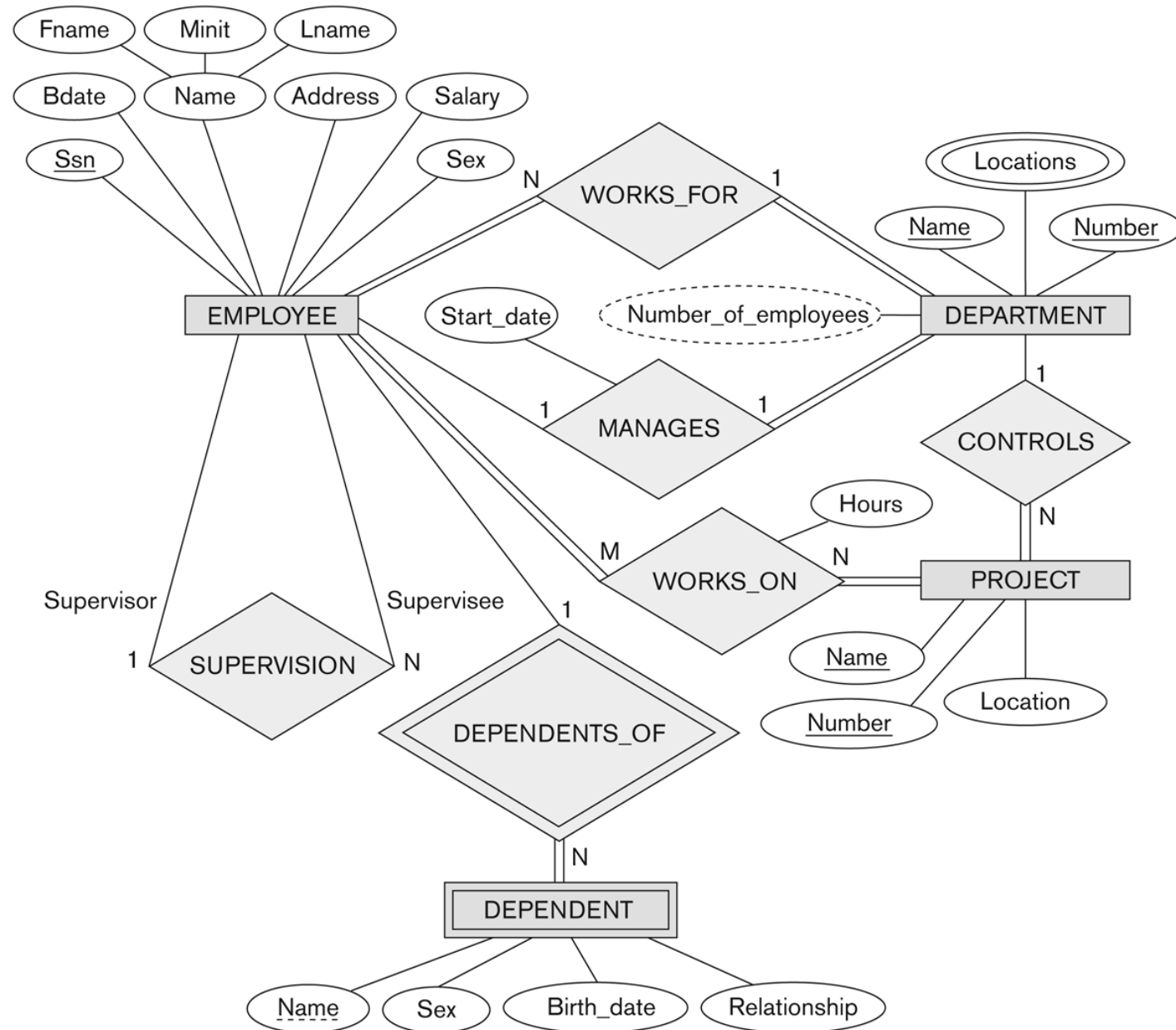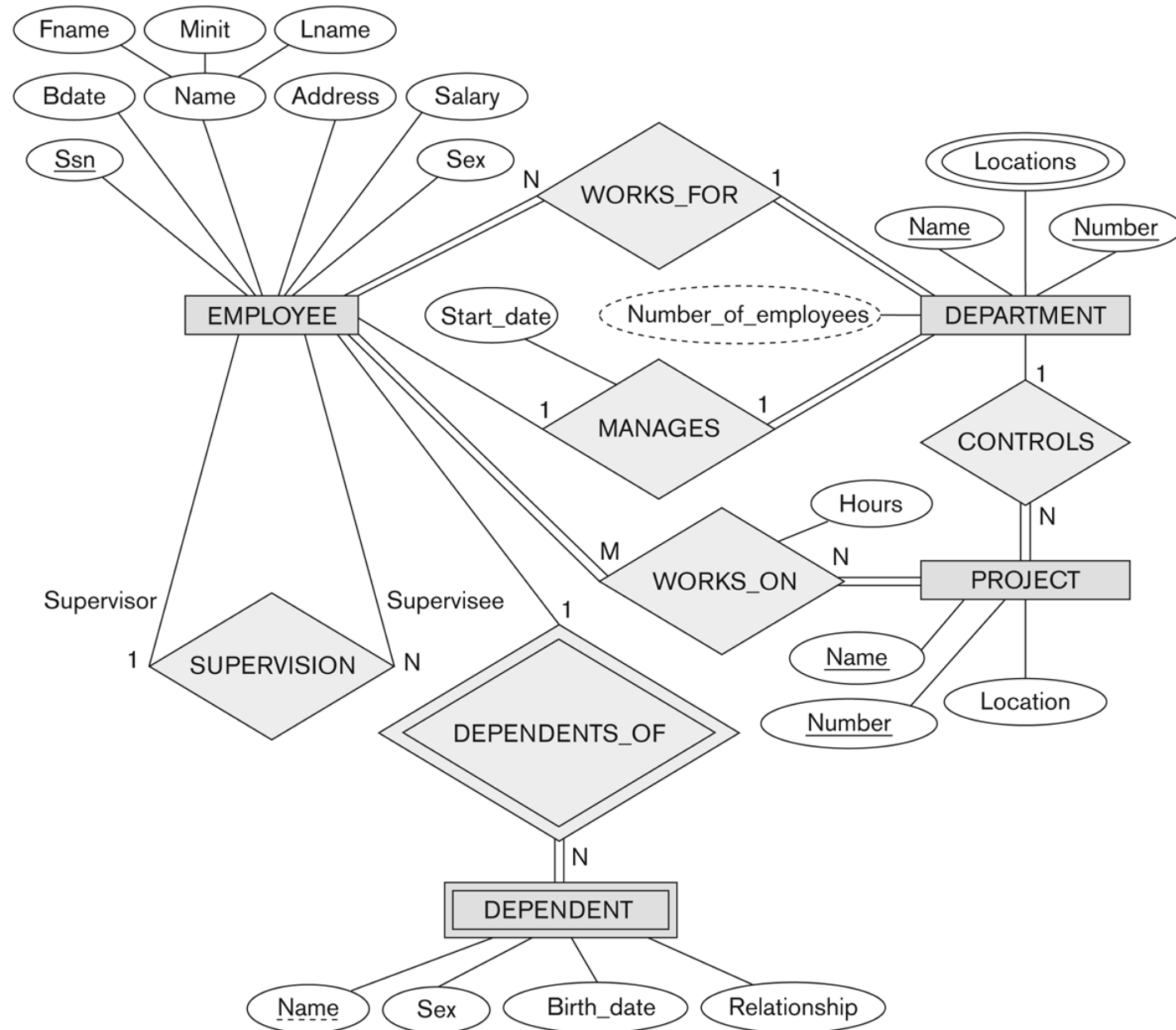
Case Study 1 (4/5)

- Each EMPLOYEE works for one DEPARTMENT. Many EMPLOYEEs work for the same DEPARTMENT.

- Each EMPLOYEE may work on several PROJECTs.

- Many EMPLOYEEs work on the same PROJECT.

- An EMPLOYEE manages at most one DEPARTMENT.

Case Study 1 (5/5)

- It is required to keep track the number of hours per week that each EMPLOYEE currently works on each PROJECT and the direct supervisor of each EMPLOYEE.

- A supervisor can supervise many EMPLOYEEs.

- An EMPLOYEE may have a number of DEPENDENTs. For each dependent, it is required to keep a record of name, sex, birthdate, and relationship to the EMPLOYEE.
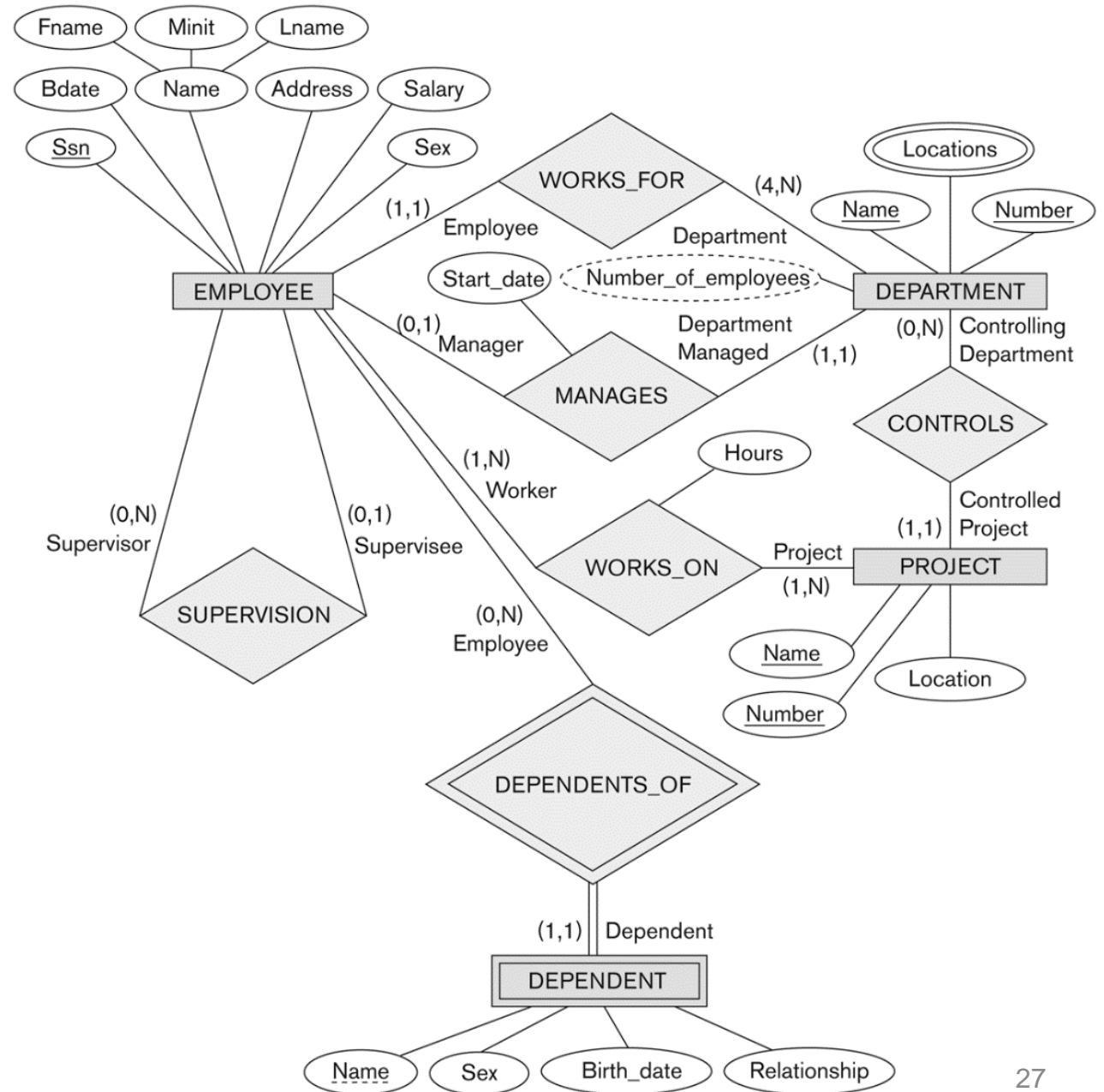
Case Study 2 (1/3)

- An ER diagram for the company database with structural constraints specified using (min, max) notation and role name.

- A DEPARTMENT has exactly one manager and an EMPLOYEE can manage at most one DEPARTMENT.

- An EMPLOYEE can work for exactly one DEPARTMENT but a DEPARTMENT has at least 4 EMPLOYEEs.
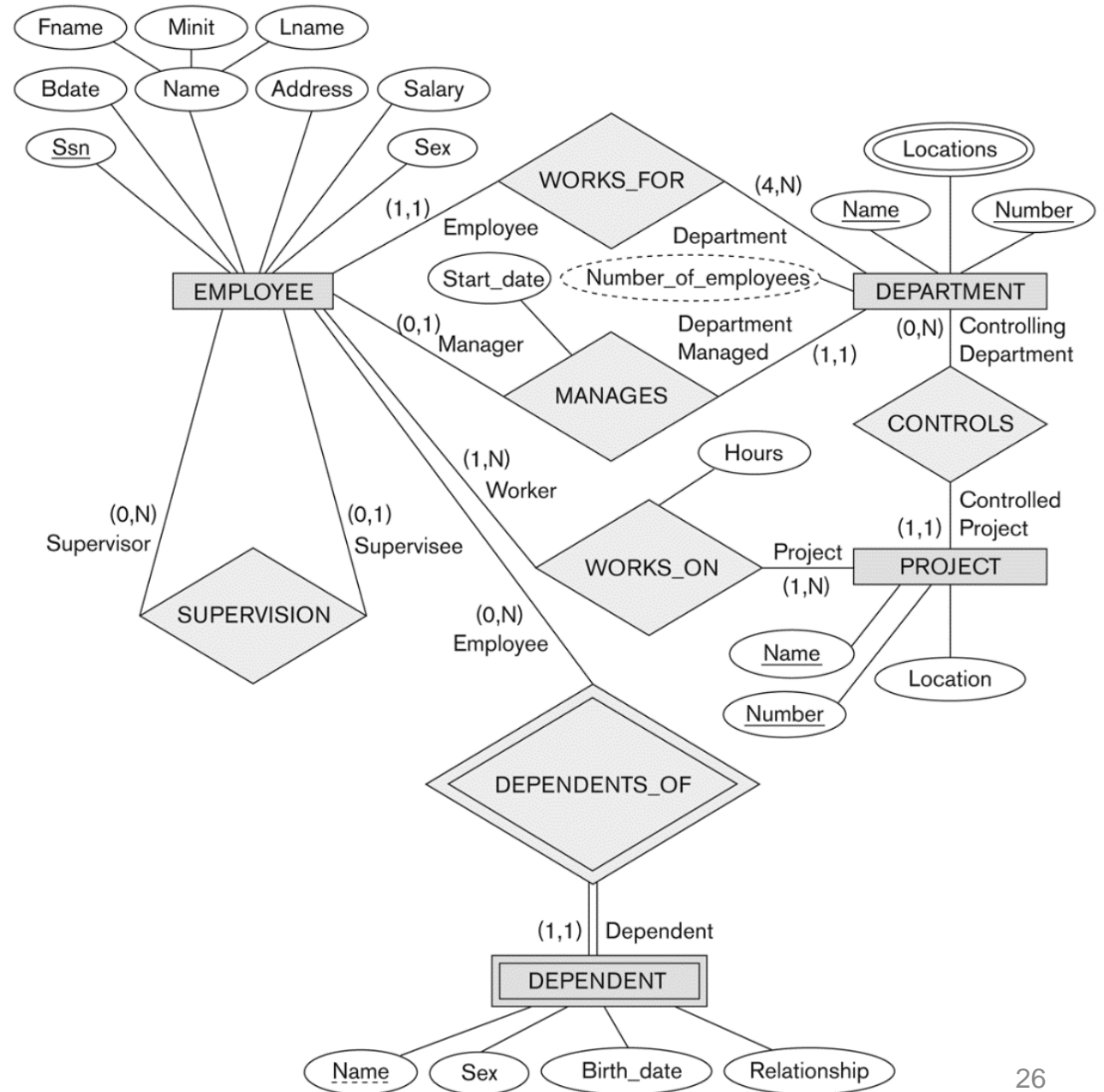
Case Study 2 (2/3)

- An EMPLOYEE works on at least one project. A PROJECT has at least one worker.

- A DEPARTMENT can control no PROJECT or any number of PROJECTs, but a PROJECT has exactly one controlling department.

- An EMPLOYEE can have no dependent or many dependents, but a dependent belongs to exactly one EMPLOYEE.

Case Study 2 (3/3)

- An EMPLOYEE has at most one supervisor and may be a supervisor supervising any number of supervisees.