ENTITY-RELATIONSHIP (ER) MODELING

Dr. Mohammad Eshtay
Fadia Ala’eddin

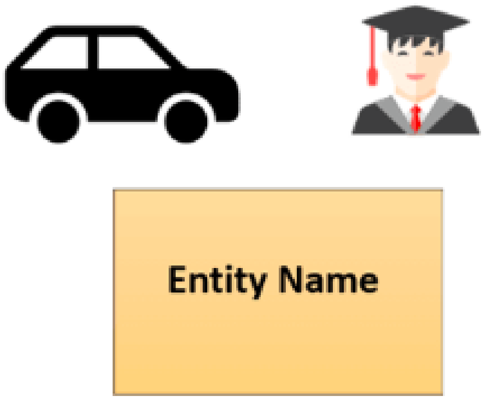
Definition

- **ENTITY RELATIONAL (ER) MODEL** is a high-level conceptual data model diagram. ER modeling helps you to analyze data requirements systematically to produce a well-designed database. The Entity-Relation model represents real-world entities and the relationship between them.

Entity Relationship Model (ER)

- ER model was proposed by **Peter Chen** in 1971
- ER model has become the standard tool for *conceptual schema* design
- ER model consists of three basic constructs: entities, attributes and relationships.
- ER model has three main concepts:
  - *Entities (and their entity types and entity sets)*
  - *Attributes (simple, composite, single valued, multivalued, stored, derived)*
  - *Relationships (and their relationship types and relationship sets)*

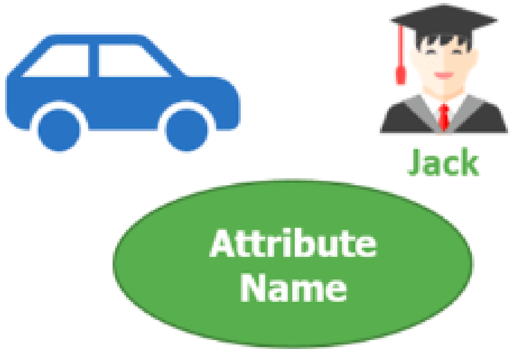
Components



Entity

Person,place,object,event or concept about which data is to be maintained

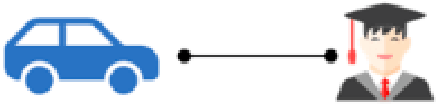
Example: Car, Student



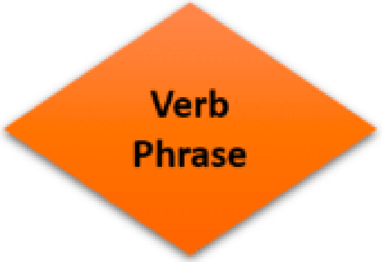
Attribute

Property or characteristic of an entity

Example: Color of car Entity Name of Student Entity



Relation



Association between the instances of one or more entity types

Example: Blue Car Belongs to Student Jack

What Is An Entity ?

- An **entity** is a “thing” in the real world with an independent existence, Or
- **Entities** are specific objects or things in the mini-world that are represented in the database.
- It may be an object with **physical existence** (e.g. person, car, house, **employee**), or it may be an object with **conceptual existence** (job, **course**).

Entity and Entity Set

- Two types of entities:
  - *Strong entity: can exist independently (or can uniquely identify itself)*
  - *Weak entity: existence depends on the existence of other (strong) entity or entities*

- Examples:
  - *An employee is a strong entity but the dependents of the employee could be weak entities*
  - *An account in a bank is a strong entity but a transaction could be a week entity*

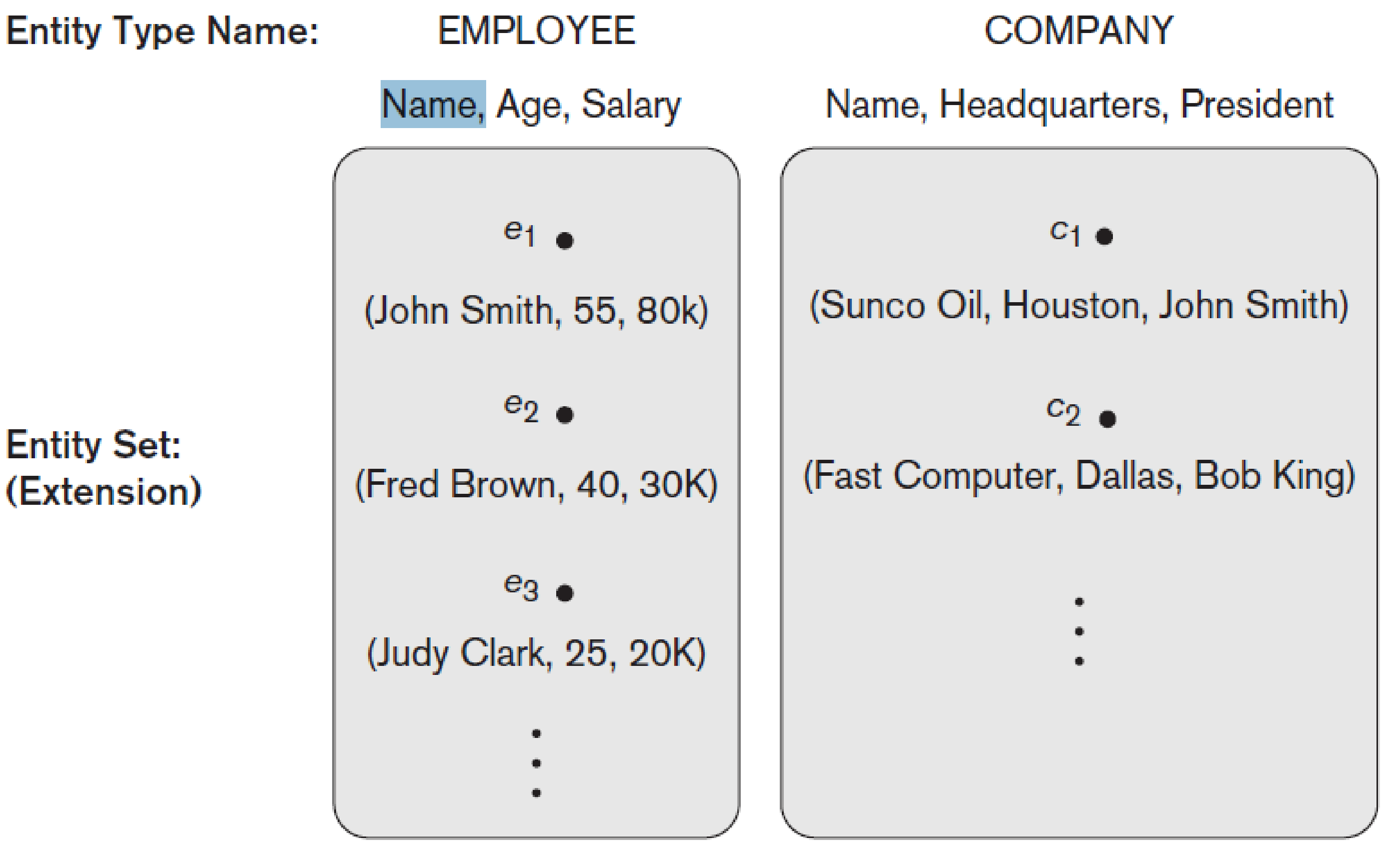
Entity Type and Entity Set

- An **entity type** defines a set of entities that have the same attributes.
  - ***STUDENT*** *is an* ***entity type (Schema)***

- An **entity set** is a collection of entities of the same entity type

- Examples:
  - ***Rema, Ali, Amal, Samer, Rana*** *are entity set of an entity type* ***STUDENT***

Entity type and Entity Set - Example



Entity Type Name: EMPLOYEE COMPANY

Name, Age, Salary

Name, Headquarters, President

Entity Set:
(Extension)

e_1 ●

(John Smith, 55, 80k)

e_2 ●

(Fred Brown, 40, 30K)

e_3 ●

(Judy Clark, 25, 20K)

⋮

c_1 ●

(Sunco Oil, Houston, John Smith)

c_2 ●

(Fast Computer, Dallas, Bob King)

⋮

Figure 3.6
Two entity types, EMPLOYEE and COMPANY, and some member entities of each.

Attributes

- An **entity** has a set of **attribute**s that describes it.
- **Attributes** are **properties** used to describe an entity.

  Person(SSN, Name, Address, Job-description, Salary).
- An entity will have a value for each of its attributes

  (999-010-201, John Smith, '20 Alebany Rd, Cardiff, UK', 'Manager', 2500)

- The properties of an entity set are called attributes of the entity set.

  –Students: SSN, Name, Address, GPA, Status, …

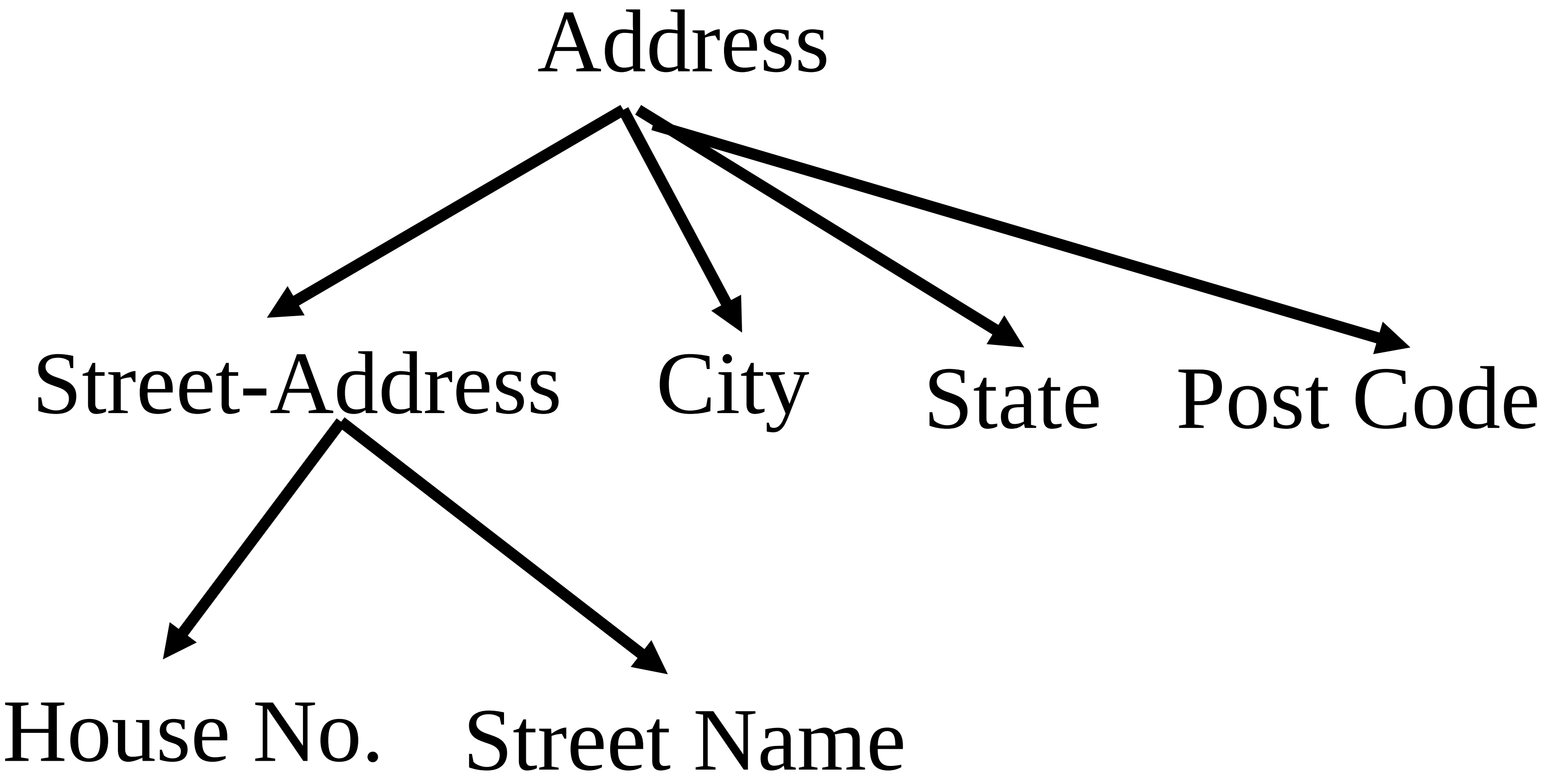
- For a given application, only a limited number of attributes of an entity set are of interest

Types of Attributes

- **Simple** (or ***atomic***) attribute is a one which cannot be divided into smaller parts.
  - *Examples: SSN, GPA, Salary, Sex.*

- **Composite** attribute is an attribute which can be divided into smaller subparts, these subparts represent more basic attributes with independent meanings of their own.
  - *Examples: Name: First_Name, Middle_Name, Last_Name*
  - *Address: Street_Address, City, State, Zip code*

An Example of A Composite Attribute



Types of Attributes

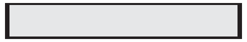
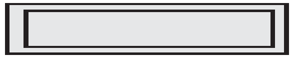
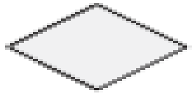
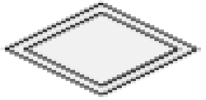
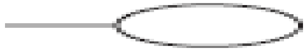
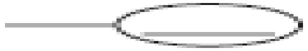
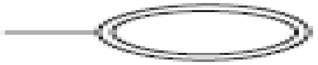
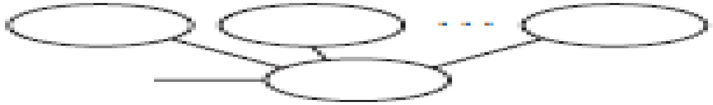
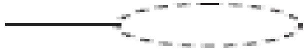
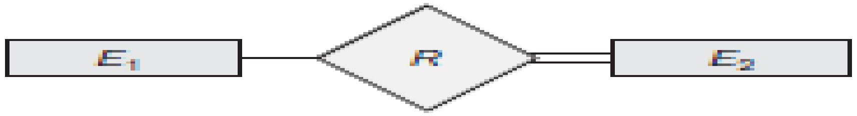
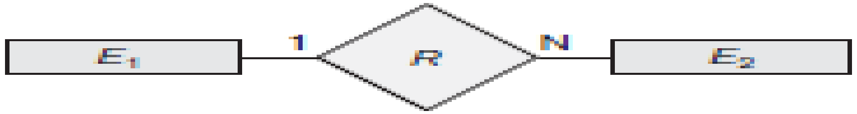
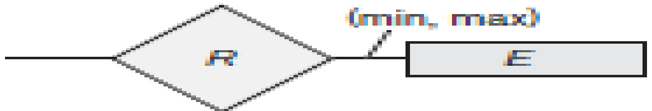
- **A single-valued attribute** is a one which has one (single) value for a particular entity.
  - *Example: Age, **BirthDate***

- **A multi-valued attribute** is a one which may have one or more values for the same entity.
  - *Authors of Books*
  - *Phone Number*

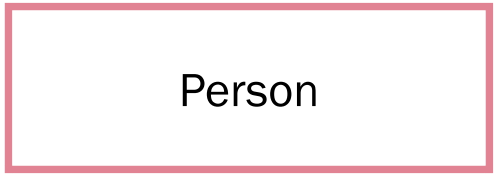
Types of Attributes

- **A stored attribute** is a one whose value is explicitly stored in the database.
  - *e.g. name, birth-date.*

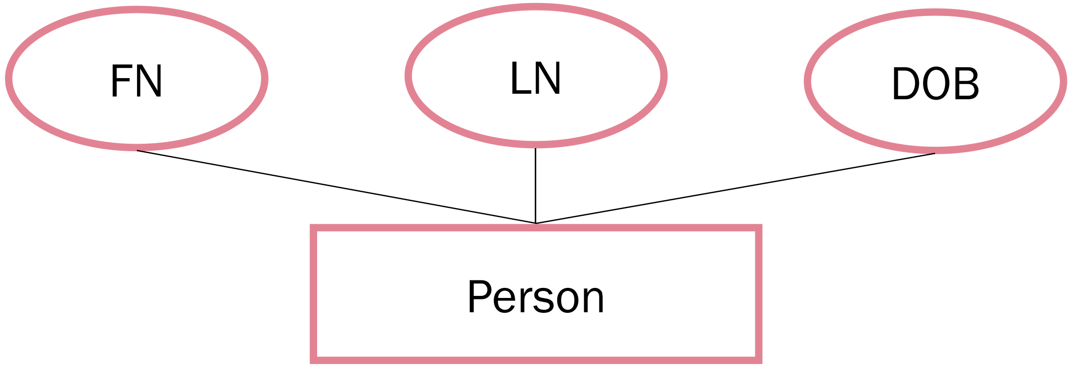
- **Derived-attributes:** whose values are computed from other attributes.
  - *Age from Birthdate*
  - *Annual Salary from Monthly Salary*
  - *NoOfEmployees ==> Count number of employees in the Employee table.*

Symbol	Meaning
	Entity
	Weak Entity
	Relationship
	Indentifying 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

- Pictorially, an entity set is denoted by a rectangle with its type written inside □
- By convention, singular noun, though we may not adhere to this convention if not adhering to it makes things clearer
- By convention, capitalized, or all capitals, if acronym



- An entity may have a set of zero or more attributes, which are some properties
- Attributes of an entity are written in ellipses (for now solid lines) connected to the entity
  - *Example: FN: "First Name." LN: "Last Name." DOB: "Date of Birth."*



Example of different attributes

To have a simple example of a person with attributes

» Child: Bob

» Child: Carol

» FN: Alice

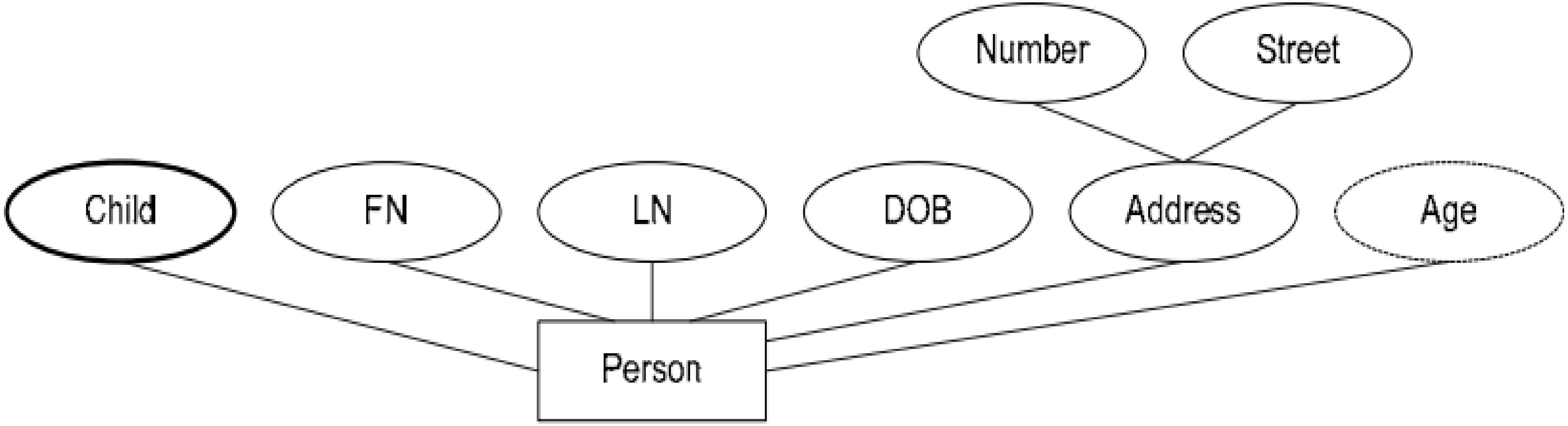
» LN: Xie

» DOB: 1980-01-01

» Address.Number: 100

» Address.Street: Mercer

» Age: Current Date minus DOB specified in years (rounded down)



Example: ER diagram of the CAR entity type

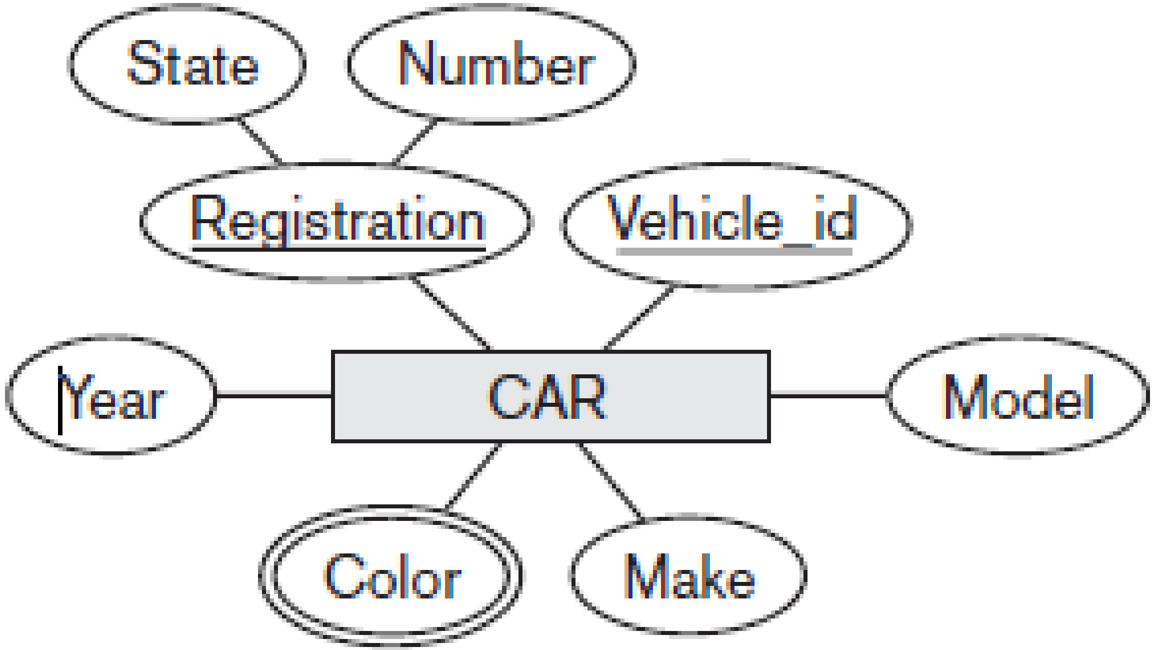


Figure 7.7
The CAR entity type with two key attributes, Registration and Vehicle_id. (a) ER diagram notation.

Relationships

- A **relationship** relates two or more distinct entities with a specific meanin
  - *For example: EMPLOYEE Ahmed* **works on** *PROJECT x*
- Degree of a relationship: number of participating entity types
  - *Binary, ternary*

Binary Relationship

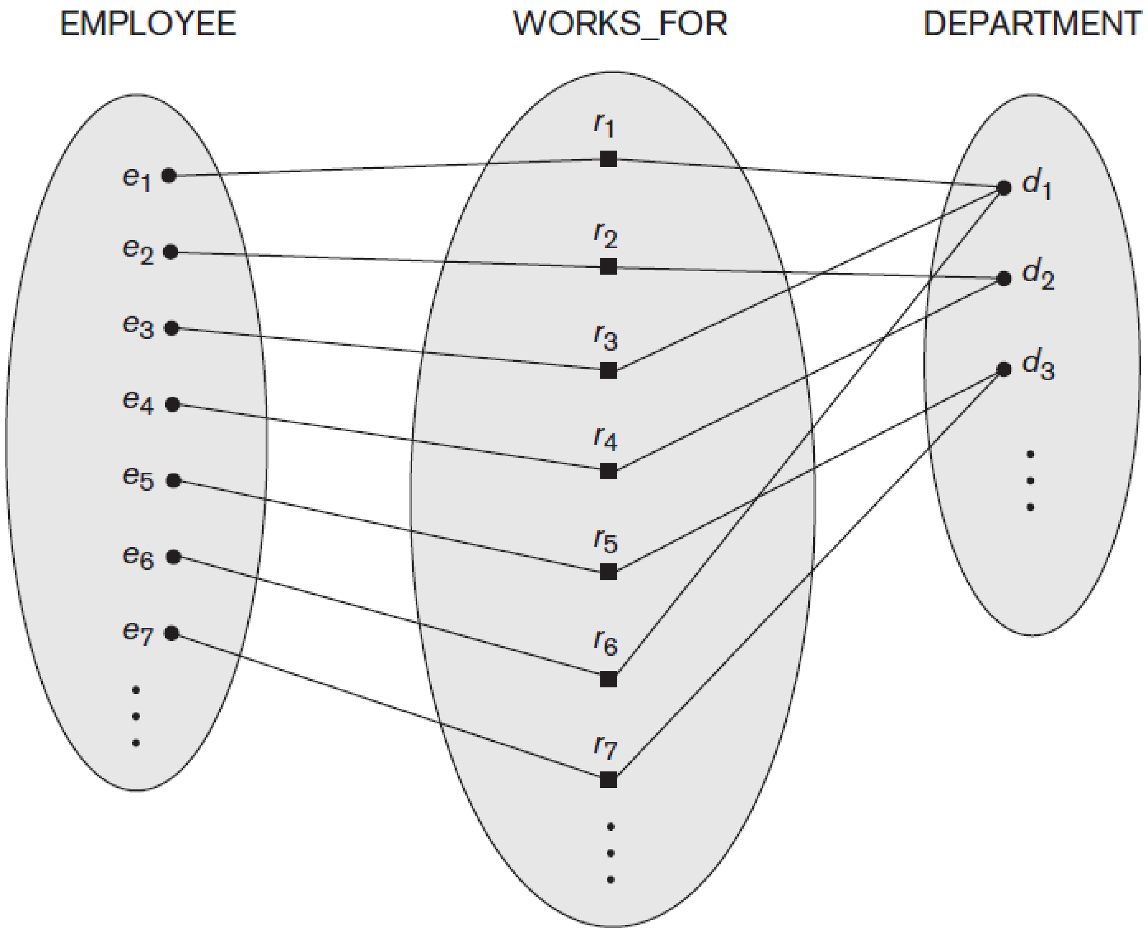
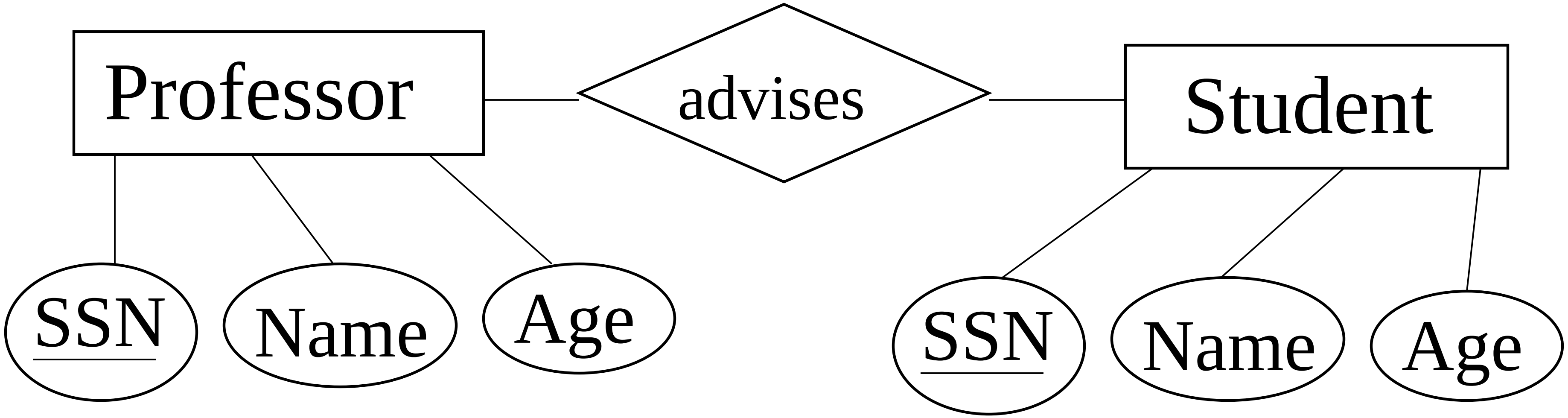
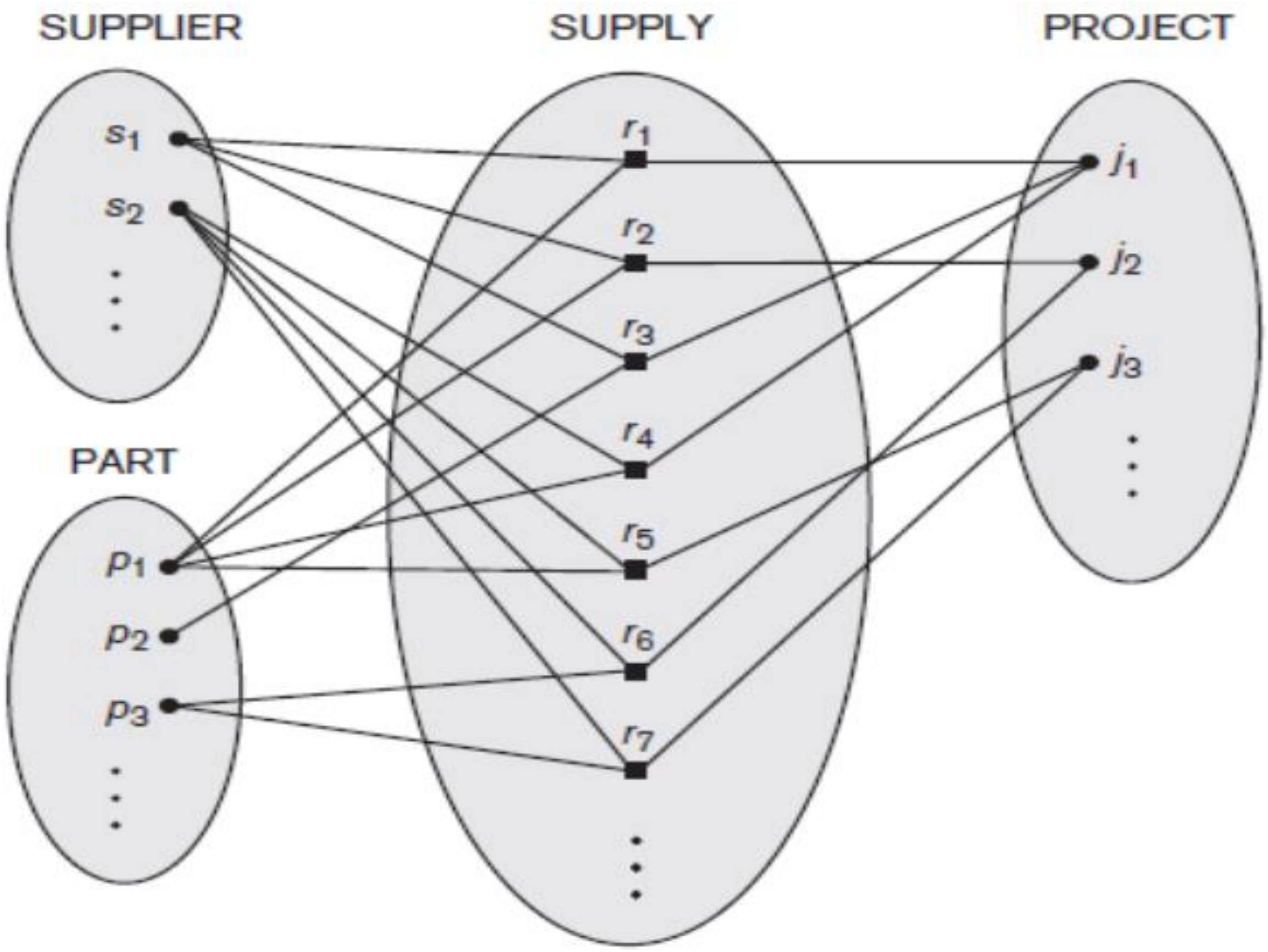


Figure 3.9
Some instances in the WORKS_FOR relationship set, which represents a relationship type WORKS_FOR between EMPLOYEE and DEPARTMENT.

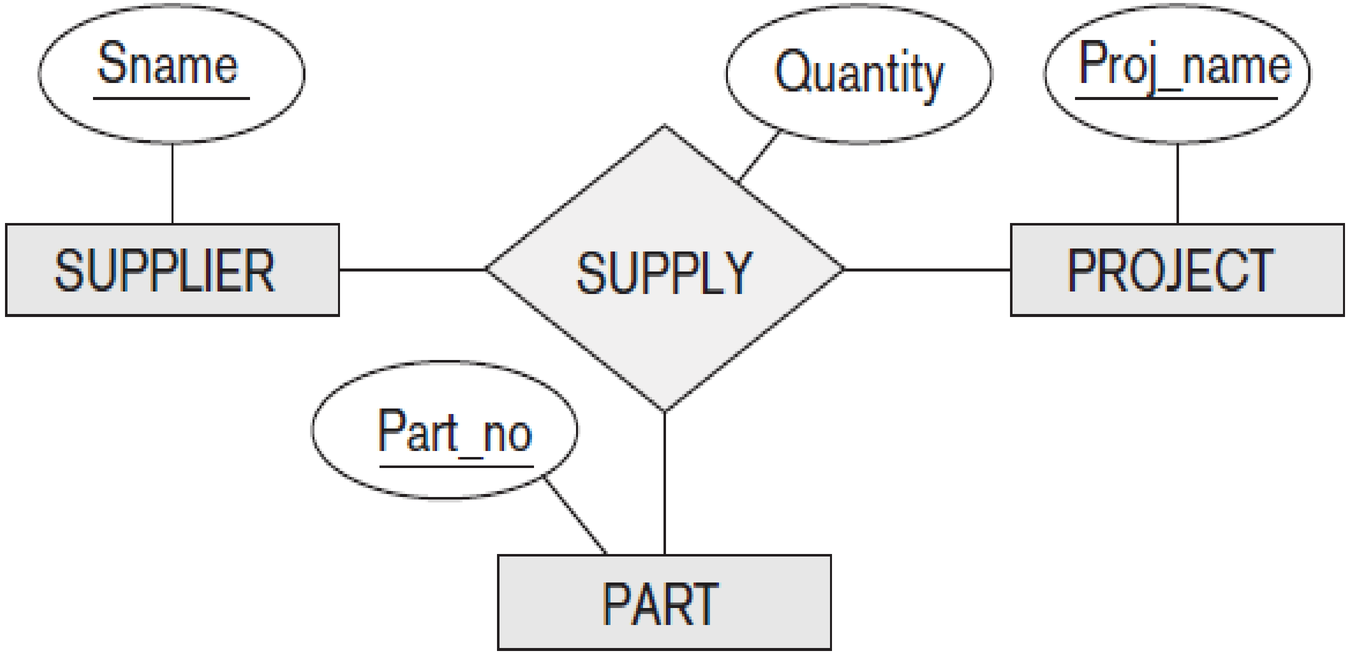
Binary relationship



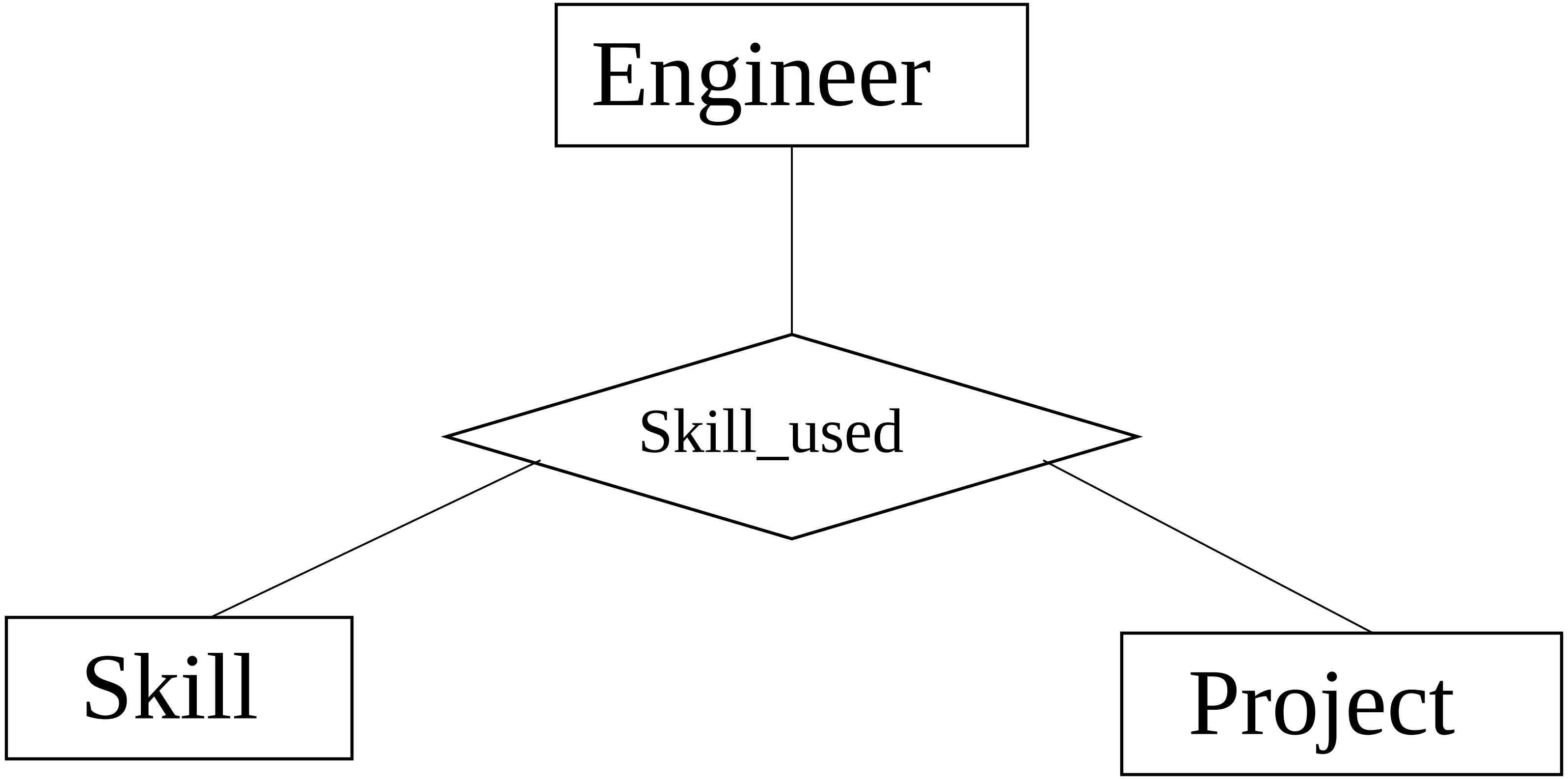
Ternary Relationship



Ternary Relationship



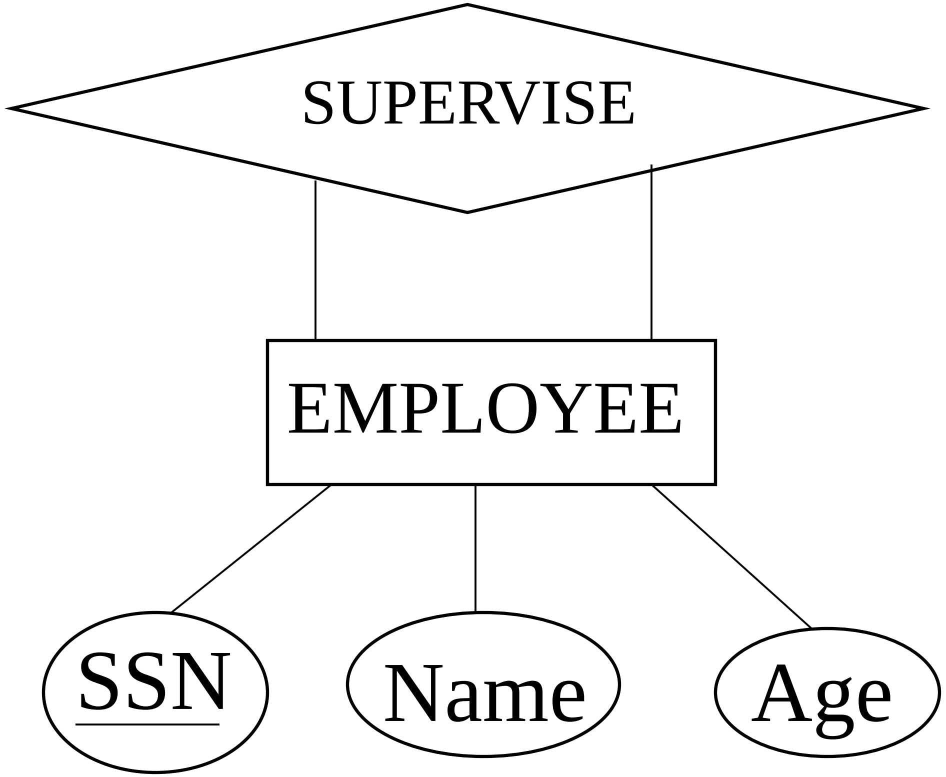
Ternary Relationship - Example



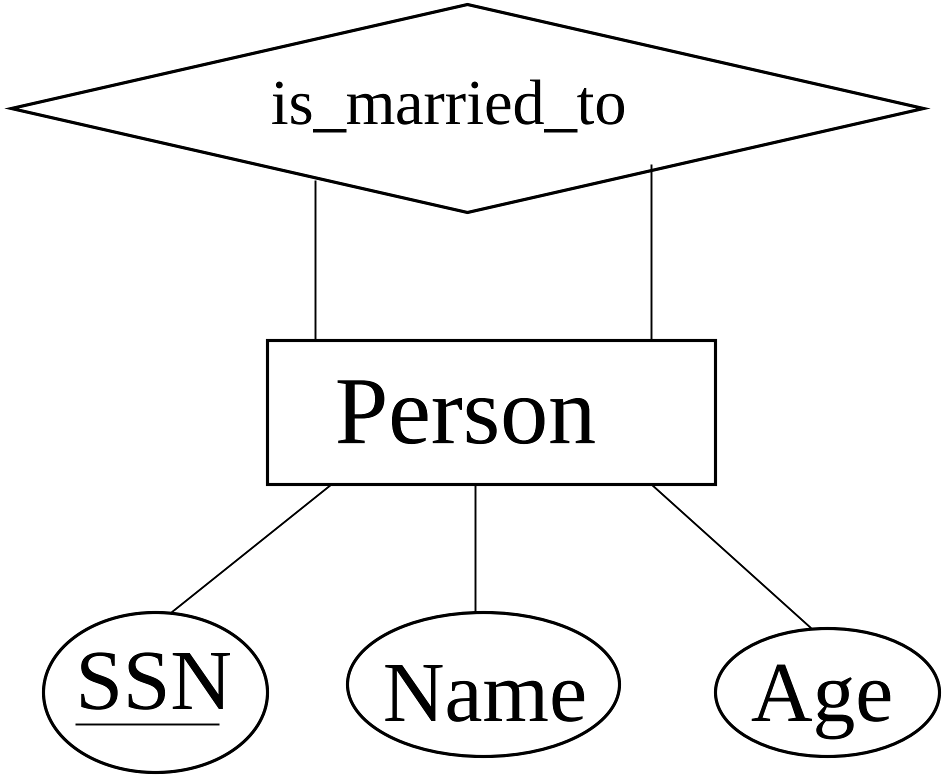
Recursive Relationship

- The SUPERVISION relationship type relates an employee to a supervisor, where both employee and supervisor entities are members of the same EMPLOYEE entity set

Recursive Relationship - Example



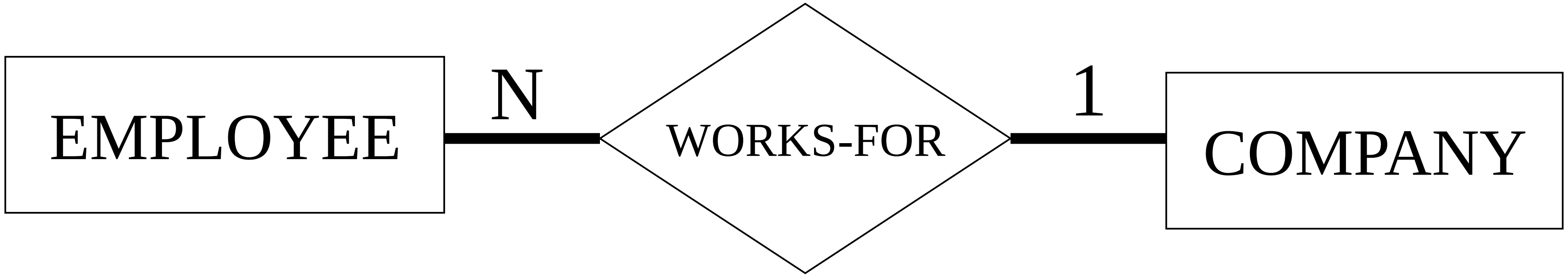
Recursive Relationship - Example



Constraints on Relationships

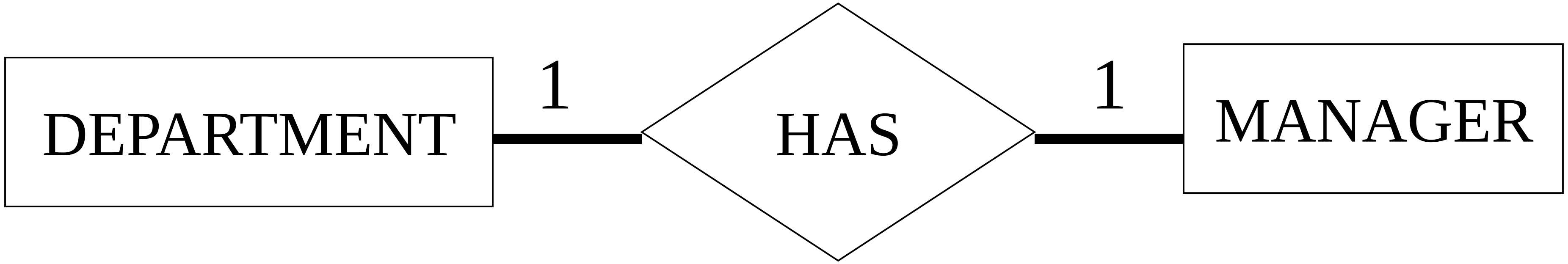
- Constraints on Relationship Types
  - ***Cardinality Ratio*** *(specifies maximum participation)*
    - One-to-one (1:1)
    - One-to-many (1:N) or Many-to-one (N:1)
    - Many-to-many (M:N)
  - *Existence Dependency Constraint (specifies minimum participation) (also called* ***participation constraint****)*
    - zero (optional participation, not existence-dependent)
    - one or more (mandatory participation, existence-dependent

One-to-many (1:N) or Many-to-one (N:1)



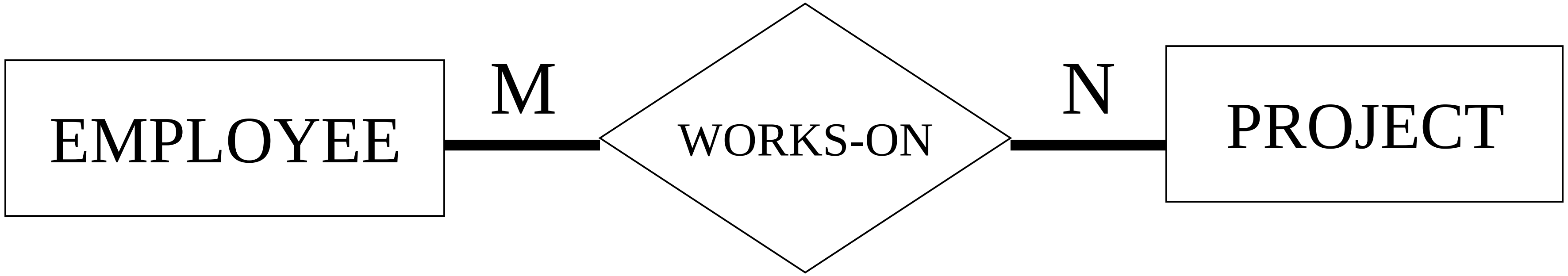
An employee works for one company, and a company has many employees working for it.

One-to-one (1:1)



A department has one manager and a manager manages one department.

Many-to-many (M:N)



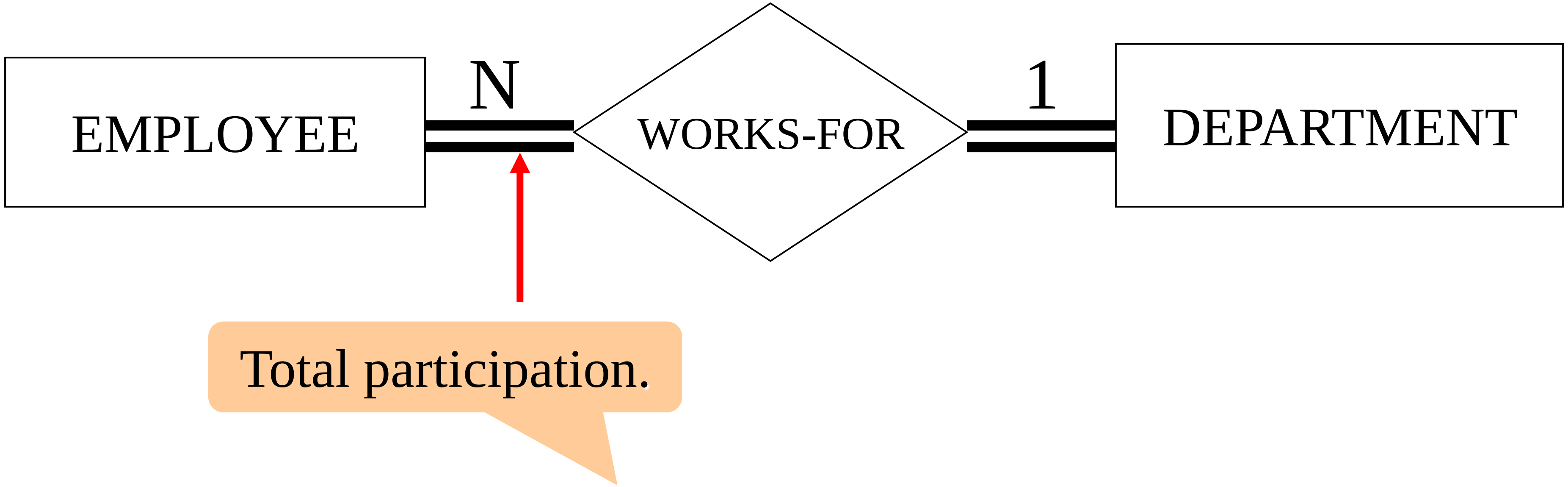
An employee works on many projects, and a project has many employees working on it.

Participation Constraints

- Specifies whether the ***existence*** of an entity depends on its being related to another entity via the relationship type.

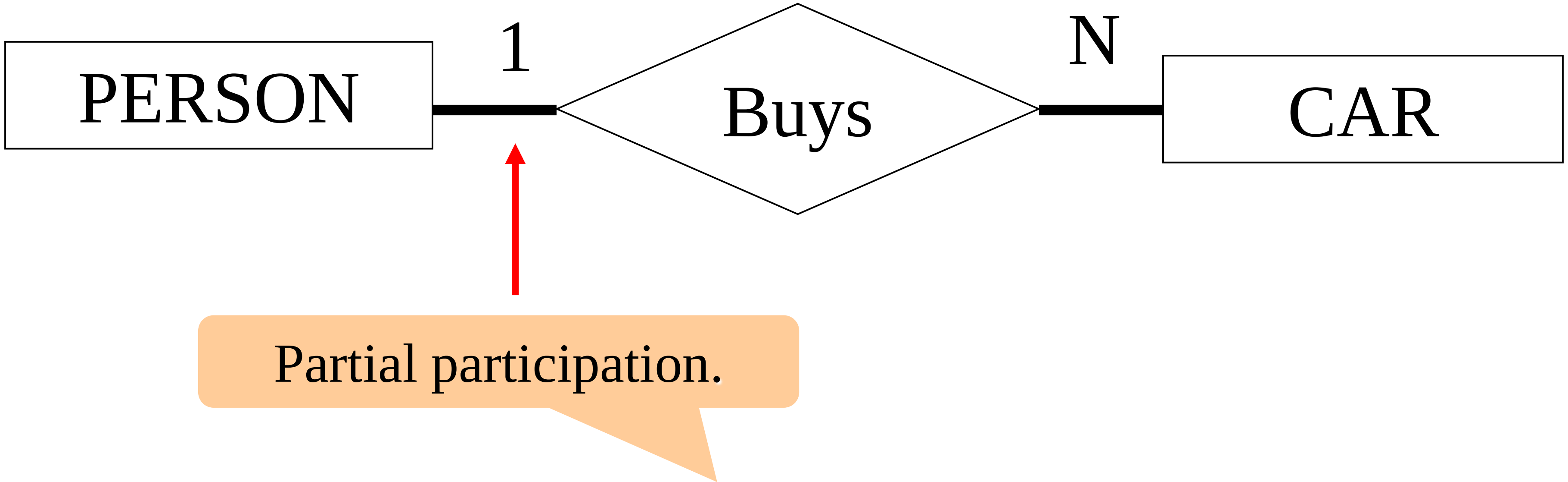
- There is total and partial participation.

Total Participation



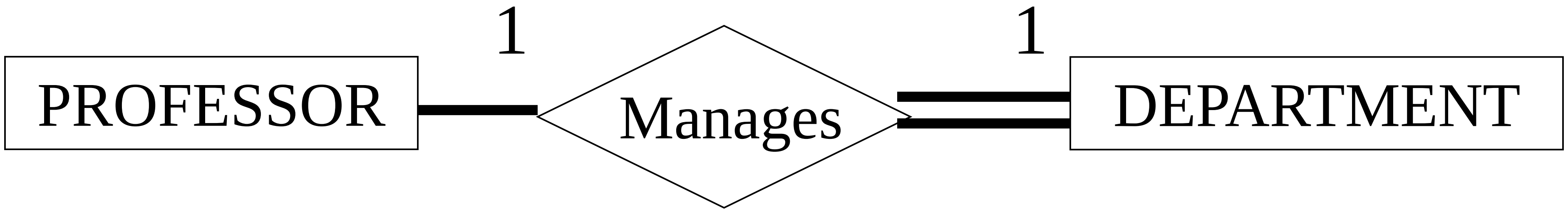
Every employee must be related to a department via WORKS-FOR relationship. A department must have at least one employee.

Partial Participation



A person may buy a car and car may be bought by a person

Total & Partial Participation

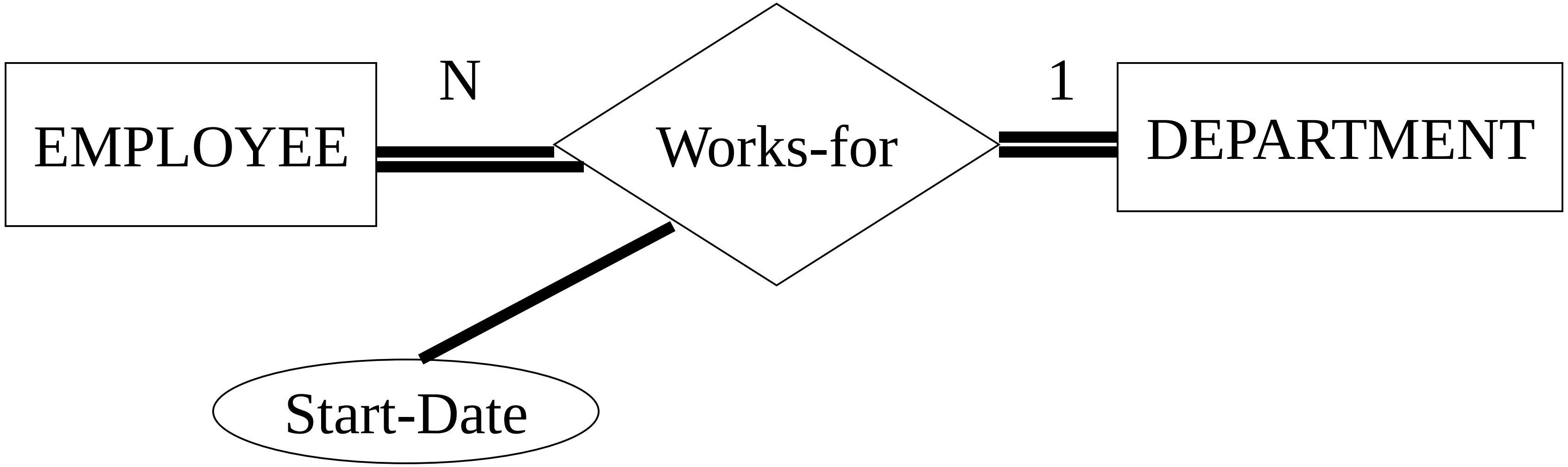


A professor may manage a department (*partial participation*), but a department must be managed by a professor (*total participation*).

Total & Partial Participation

- Total
  - *if every **<u>employee</u>** **must** **work for** a **<u>department</u>**, the entity EMPLOYEE in WORKS_FOR is called total sometimes called **<u>existence dependency</u>***
- Partial
  - ***some** or "part" of the set of **<u>employee entities</u>** are related to a **<u>department entity</u>** via MANAGES, but not necessarily all*

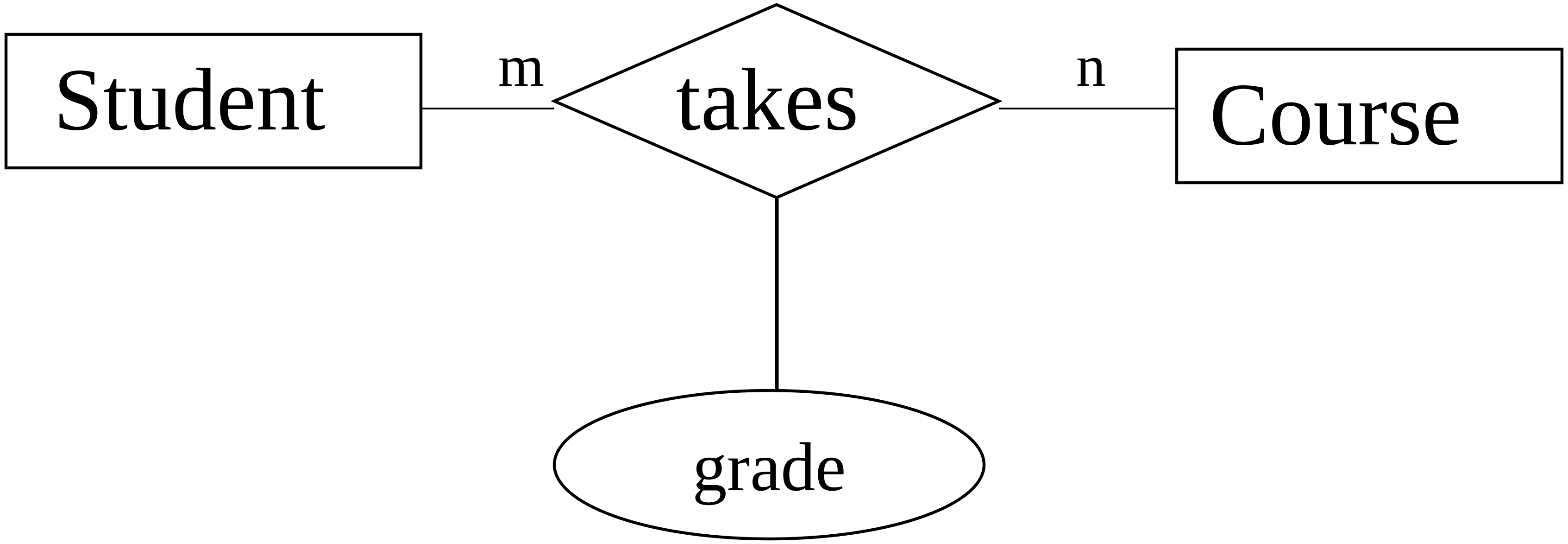
Attributes of Relationship Types



We may keep a *start date* attribute to record for each employee the date he/she started *work for* a certain department.

Attribute of Relationship

Where to keep the grade information?

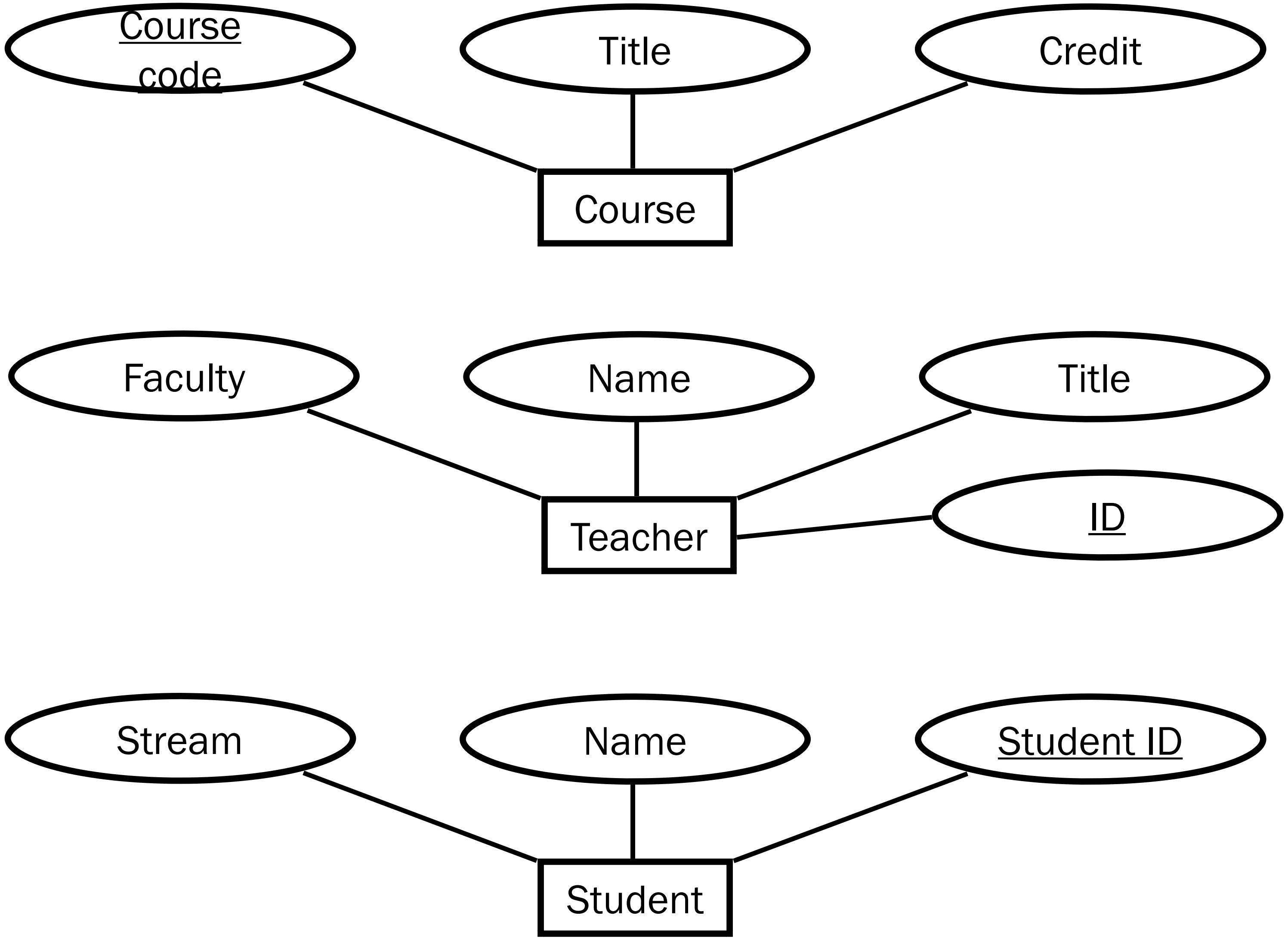


E-R Diagram with Relationships Example

College Registration System:

- College Courses: contain course code, title and credits.
- Course offering: contain course code, year, semester, section number, teacher, time and class room.
- Student : contain student id, student name, stream.
- Teacher: contain teacher ID, teacher name, faculty, title.

In addition, the enrollment of student in any courses and mark that display to them must be appropriately model.


Course code
Title
Credit
Course
Faculty
Name
Title
ID
Teacher
Stream
Name
Student ID
Student

Binary relationship

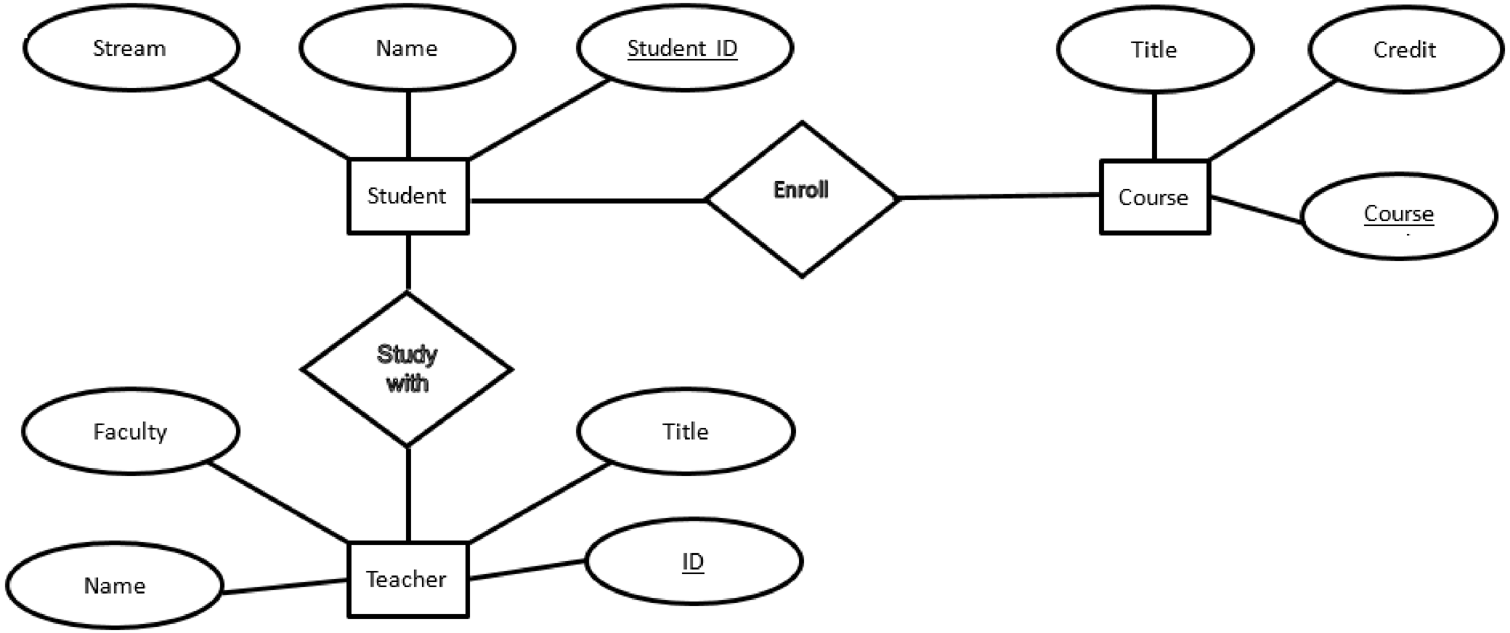


Figure 4: College ER diagrams

Weak Entity Types

- *An entity that* ***does not have a key attribute,*** *A Weak entity type has a* ***partial key.***
- A weak entity type always has a **total participation** with its identifying entity type

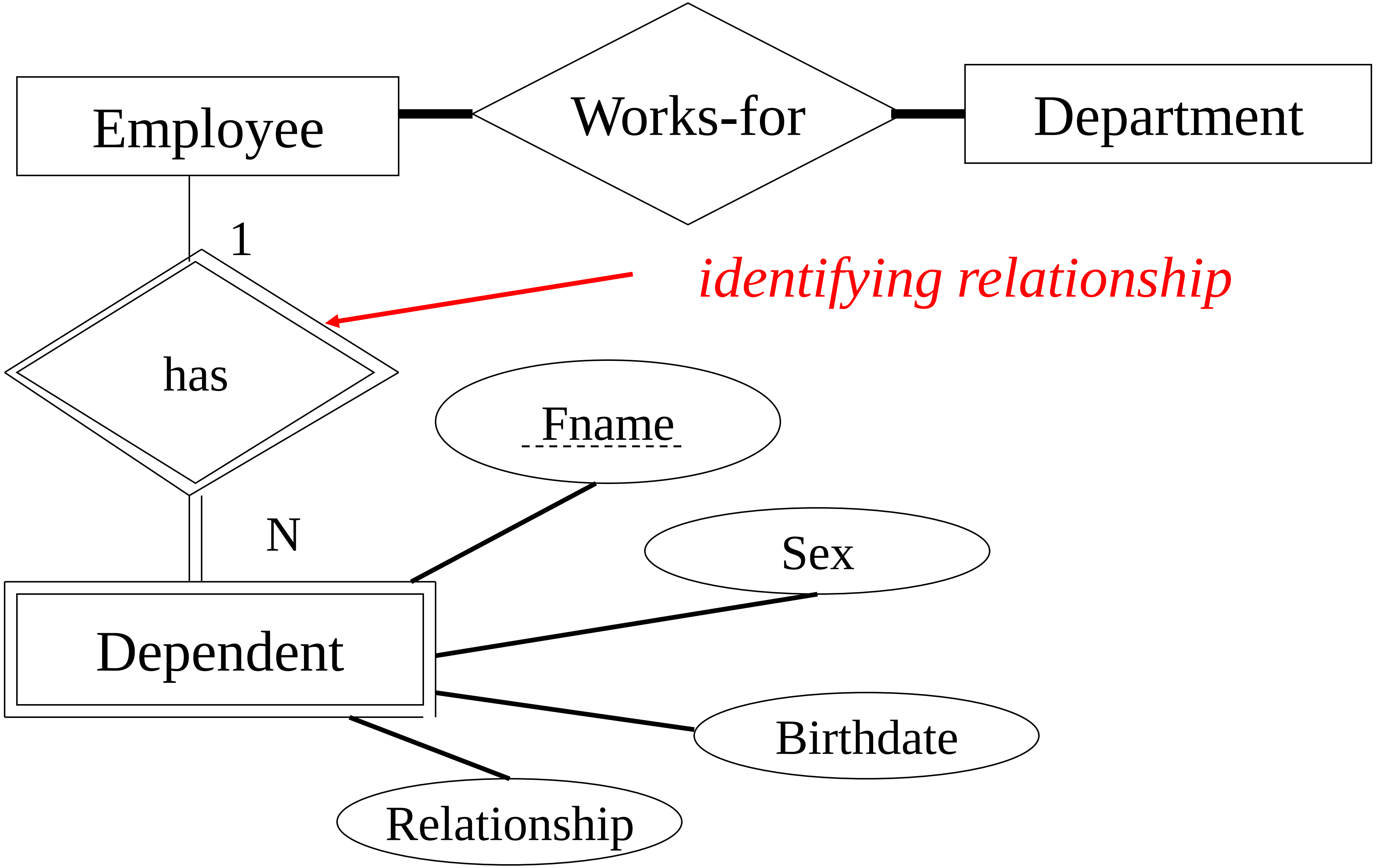
- **Example:**
  - *A DEPENDENT entity is identified by the dependent’s first name, and the specific EMPLOYEE with whom the dependent is related*
  - ***Name*** *of DEPENDENT is the* ***partial key***
  - *DEPENDENT is a* ***weak entity type***
  - *EMPLOYEE is its identifying entity type via the identifying relationship type DEPENDENT_OF*

Weak Entity

- A weak entity is one that can only exist when owned by another one. For example: a *ROOM* can only exist in a *BUILDING*.
- On the other hand, a *TIRE* might be considered as a strong entity because it also can exist without being attached to a *CAR*.

A weak entity type is an entity which does not have any key attributes

Weak Entity - Example

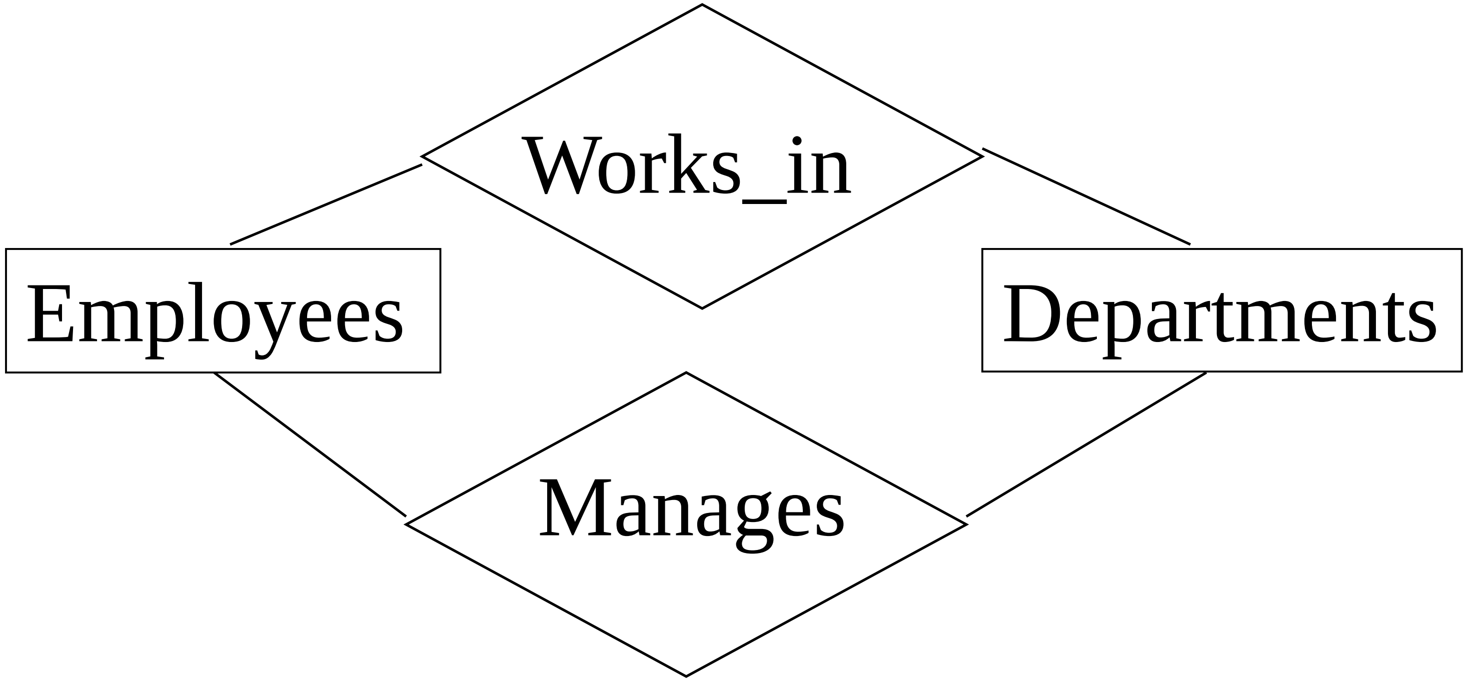


Weak Entity Types

- In the previous example, the **first name** is enough to identify kids within a single family, but is not enough to identify entities as stand alone entities (two families may use identical names for their kids)

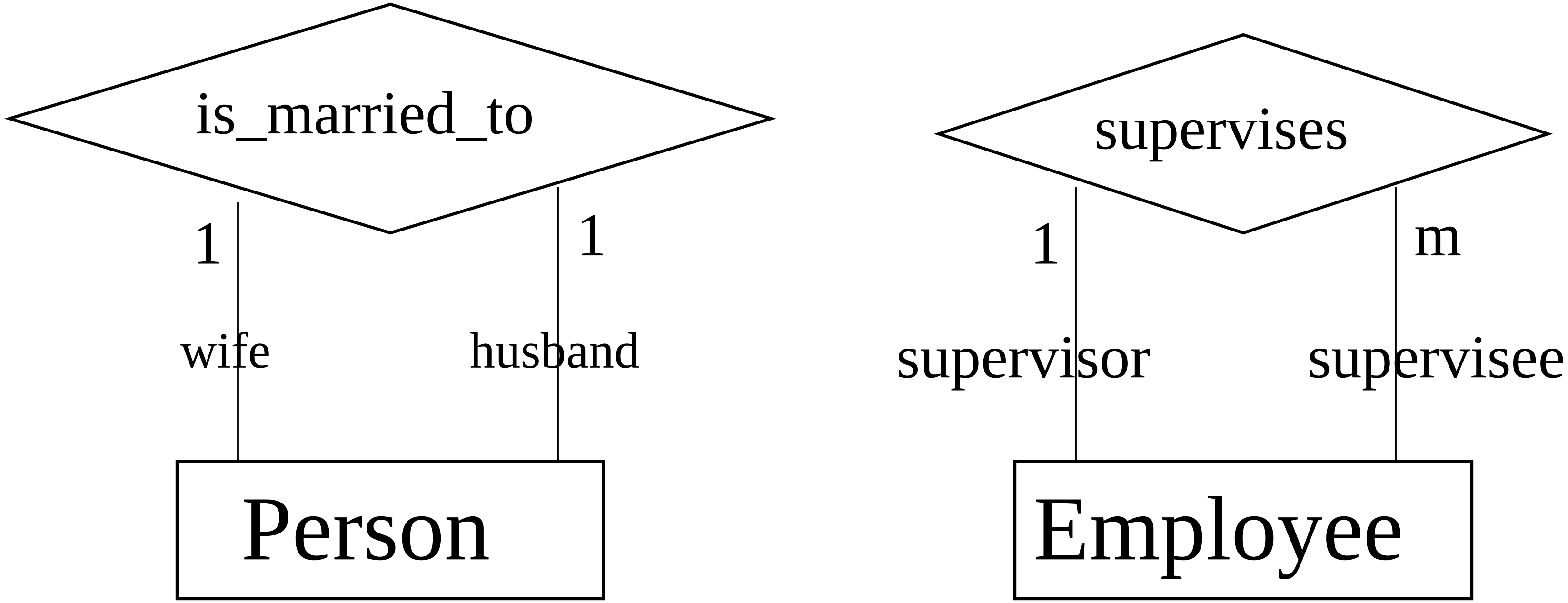
Relationships

- Several relationships may exist among the same set of entity sets.

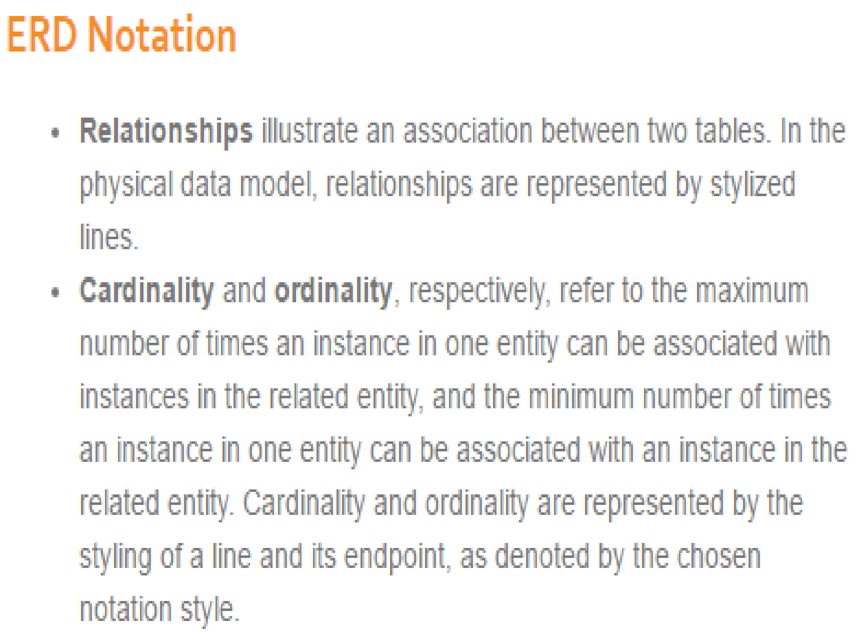


Role of an Entity Set

Roles need to be explicitly given.

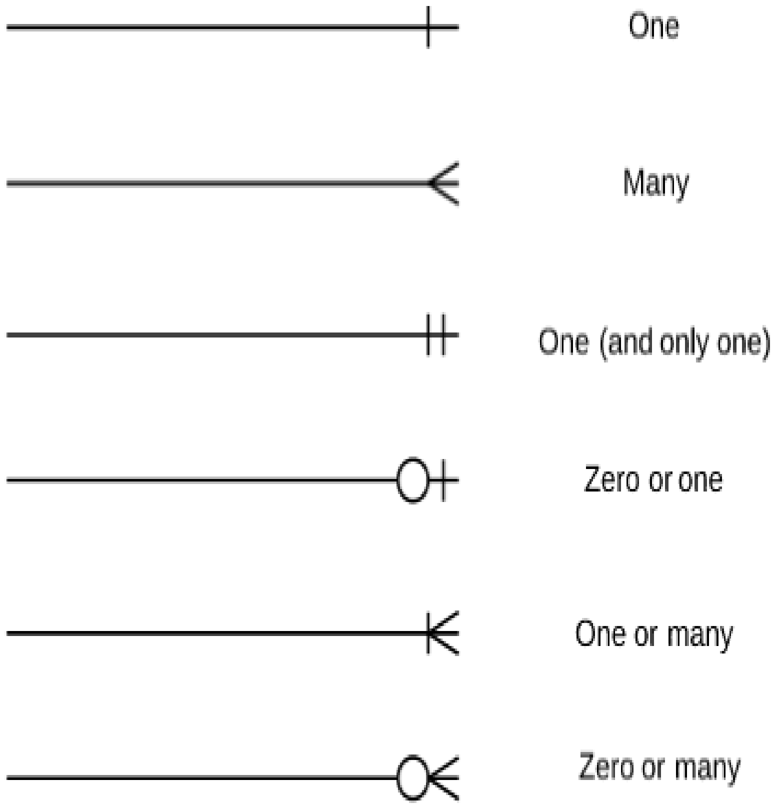


ERD Notations



ERD Notation

- **Relationships** illustrate an association between two tables. In the physical data model, relationships are represented by stylized lines.
- **Cardinality** and **ordinality**, respectively, refer to the maximum number of times an instance in one entity can be associated with instances in the related entity, and the minimum number of times an instance in one entity can be associated with an instance in the related entity. Cardinality and ordinality are represented by the styling of a line and its endpoint, as denoted by the chosen notation style.



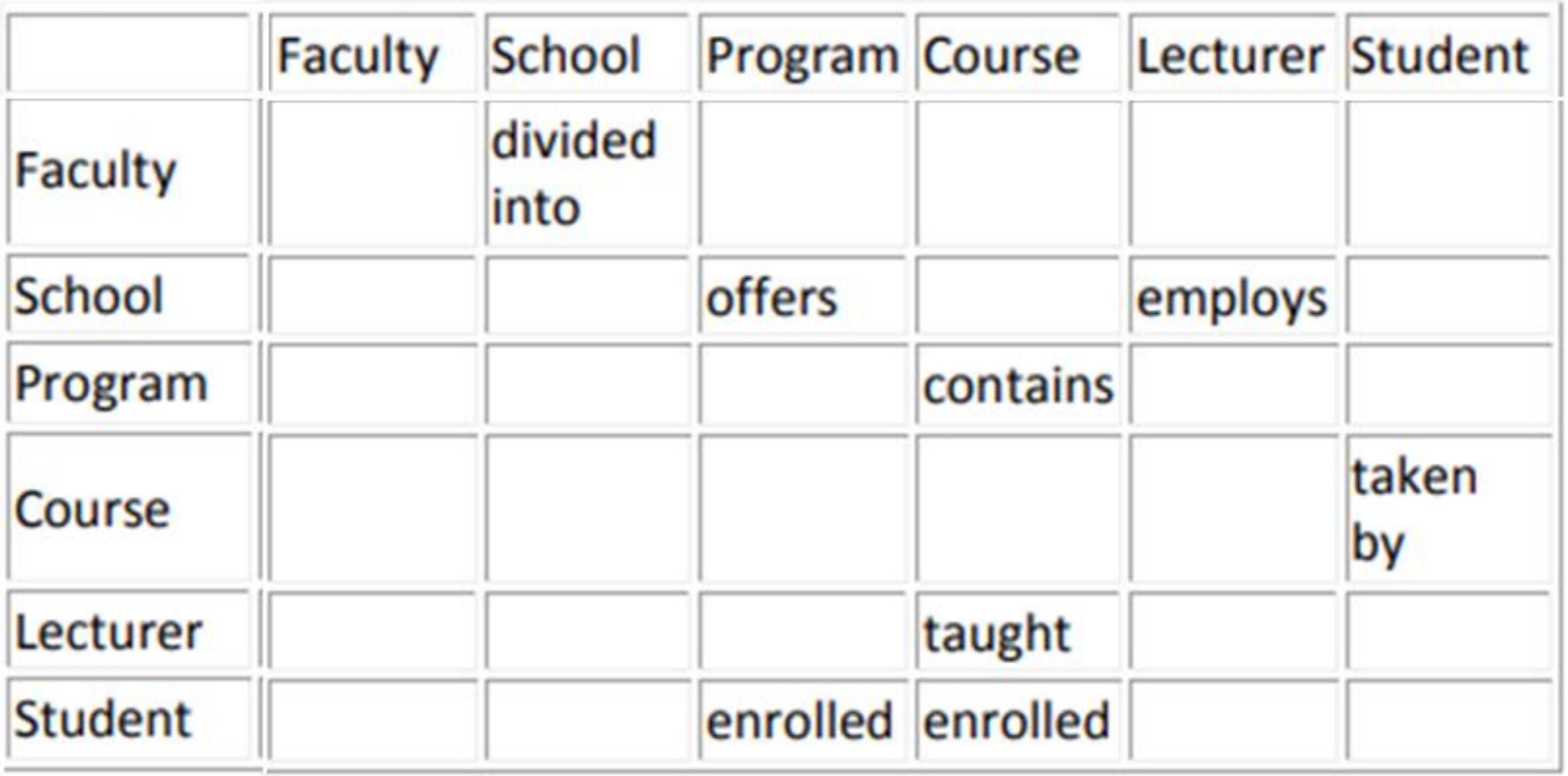
Example

A University contains many Faculties. The Faculties in turn are divided into several Schools. Each School offers numerous programs and each program contains many courses. Lecturers can teach many different courses and even the same course numerous times. Courses can also be taught by many lecturers. A student is enrolled in only one program but a program can contain many students. Students can be enrolled in many courses at the same time and the courses have many students enrolled.

Step 1 - Identify Entities

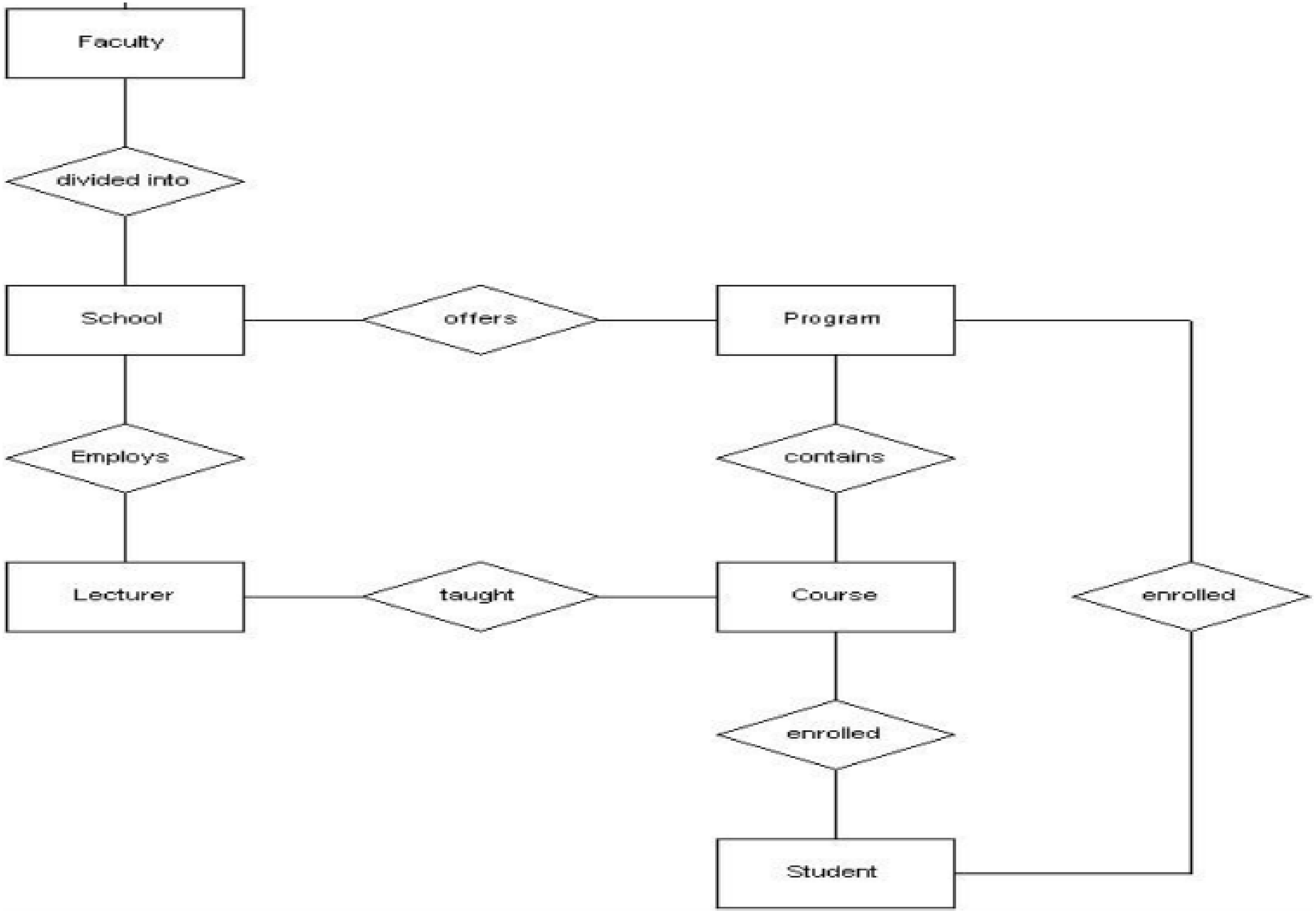
1. University
2. Faculty
3. School
4. Program
5. Course
6. Lecturer
7. Student

Step 2 - Find Relationships



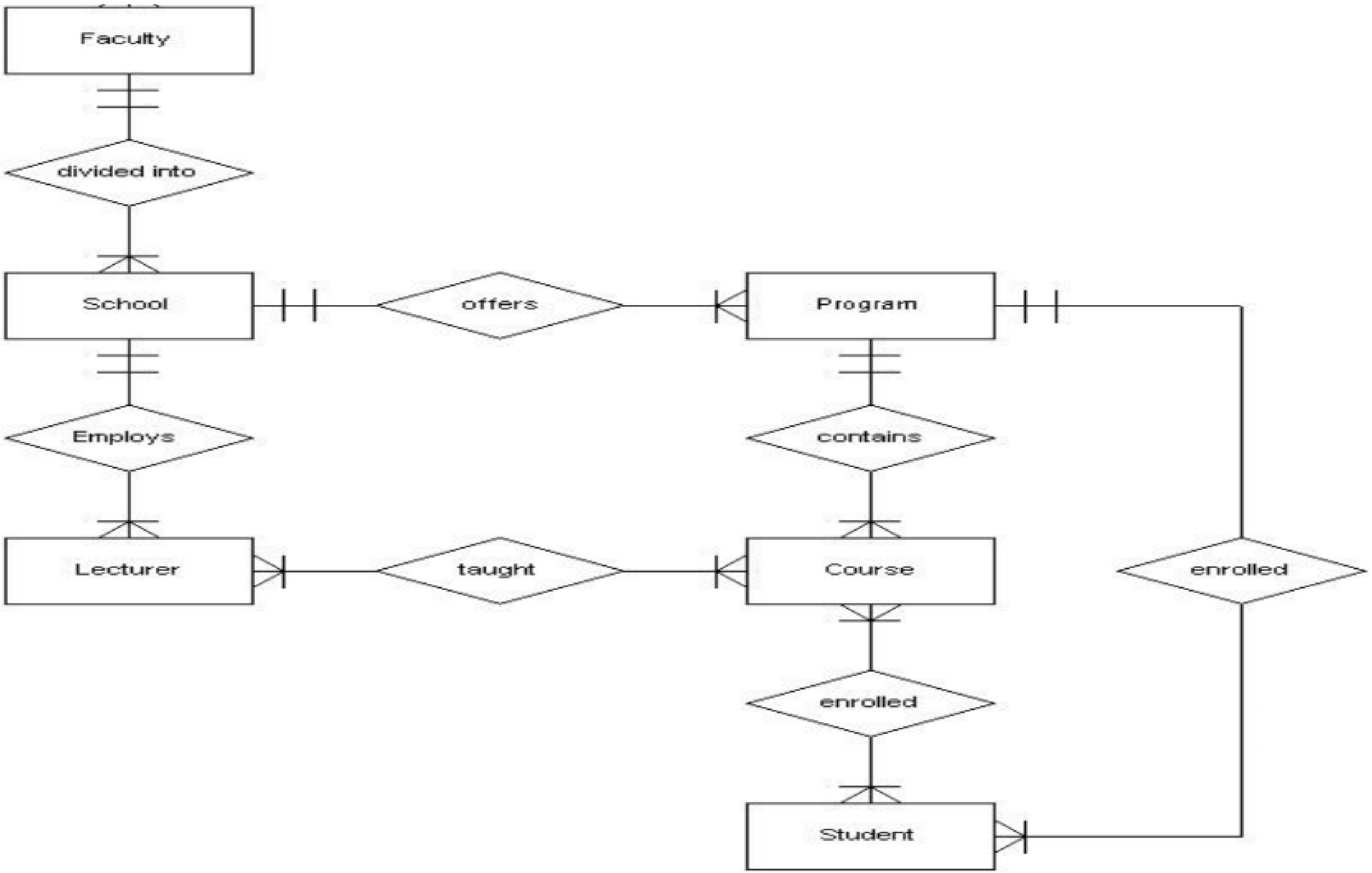
	Faculty	School	Program	Course	Lecturer	Student
Faculty		divided into				
School			offers		employs	
Program				contains		
Course						taken by
Lecturer				taught		
Student			enrolled	enrolled		

Step 3 - Draw rough ERD

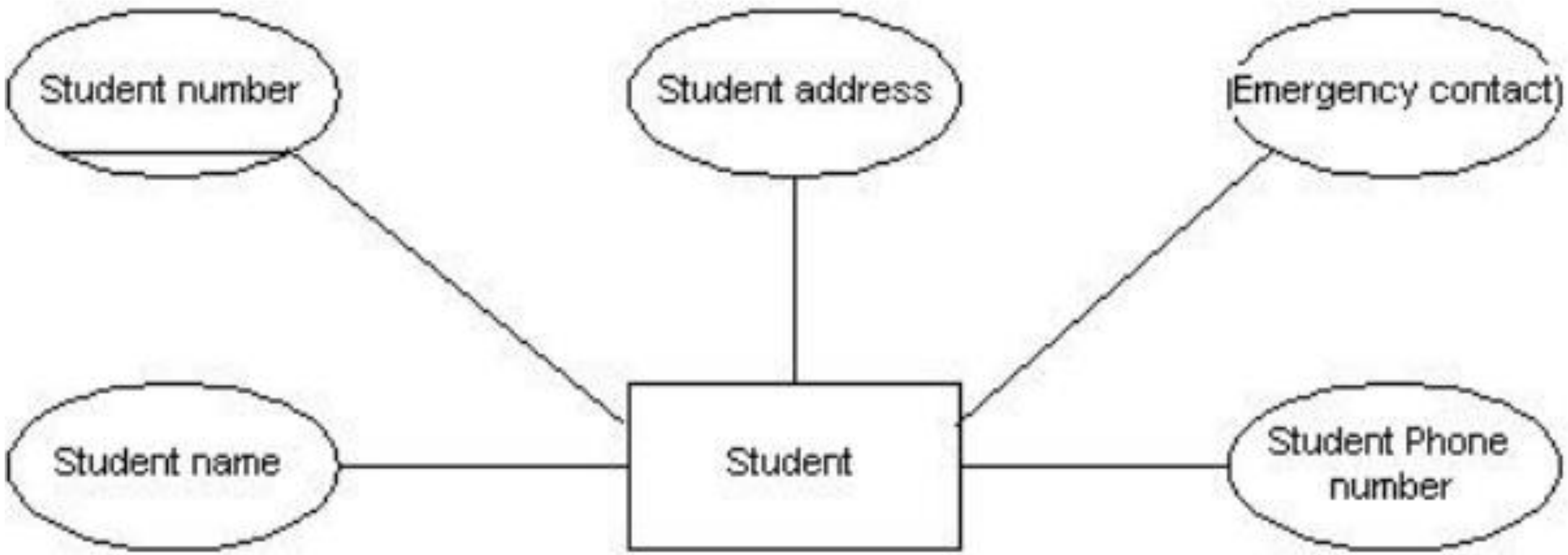


Step 4 - Fill in The Cardinality

- Each faculty is divided into several schools
- Each school offers numerous programs
- Each program contains many courses
- Each school employs many lecturers
- Lecturers can teach many courses
- Lecturers can teach the same course many times
- Courses can be taught by more than one lecturer
- A student is enrolled in only one program
- Students can be enrolled in many courses at the same time
- Courses have many students enrolled


Faculty
divided into
School
offers
Program
Employs
contains
Lecturer
taught
Course
enrolled
enrolled
Student

DRAW FULLY ATTRIBUTES ERD - Homework



Company Database – Example

Requirements:

- 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.
- A department controls a number of projects, each of which has a unique name, a unique number, and a single location.
- The database will store each employee’s name, Social Security number, address, salary, sex (gender), and birth date. An employee is assigned to on department, but may work on several projects, which are not necessarily controlled by the same department. It is required to keep track of the current number of hours per week that an employee works on each project, as well as the direct supervisor of each employee (who is another employee).
- The database will keep track of the dependents of each employee for insurance purposes, including each dependent’s first name, sex, birth date, and relationship to the employee

IdentifingEntities

1. An entity type DEPARTMENT with attributes Name, Number, Locations, Manager, and Manager_start_date. Locations is the only multivalued attribute. We can specify that both Name and Number are (separate) key attributes because each was specified to be unique.

2. An entity type PROJECT with attributes Name, Number, Location, and Controlling_department. Both Name and Number are (separate) key attributes.

3. An entity type EMPLOYEE with attributes Name, Ssn, Sex, Address, Salary, Birth_date, Department, and Supervisor. Both Name and Address may be composite attributes; however, this was not specified in the requirements. We must go back to the users to see if any of them will refer to the individual components of Name—First_name, Middle_initial, Last_name—or of Address. In our example, Name is modeled as a composite attribute, whereas Address is not, presumably after consultation with the users.

4. An entity type DEPENDENT with attributes Employee, Dependent_name, Sex, Birth_date, and Relationship (to the employee).

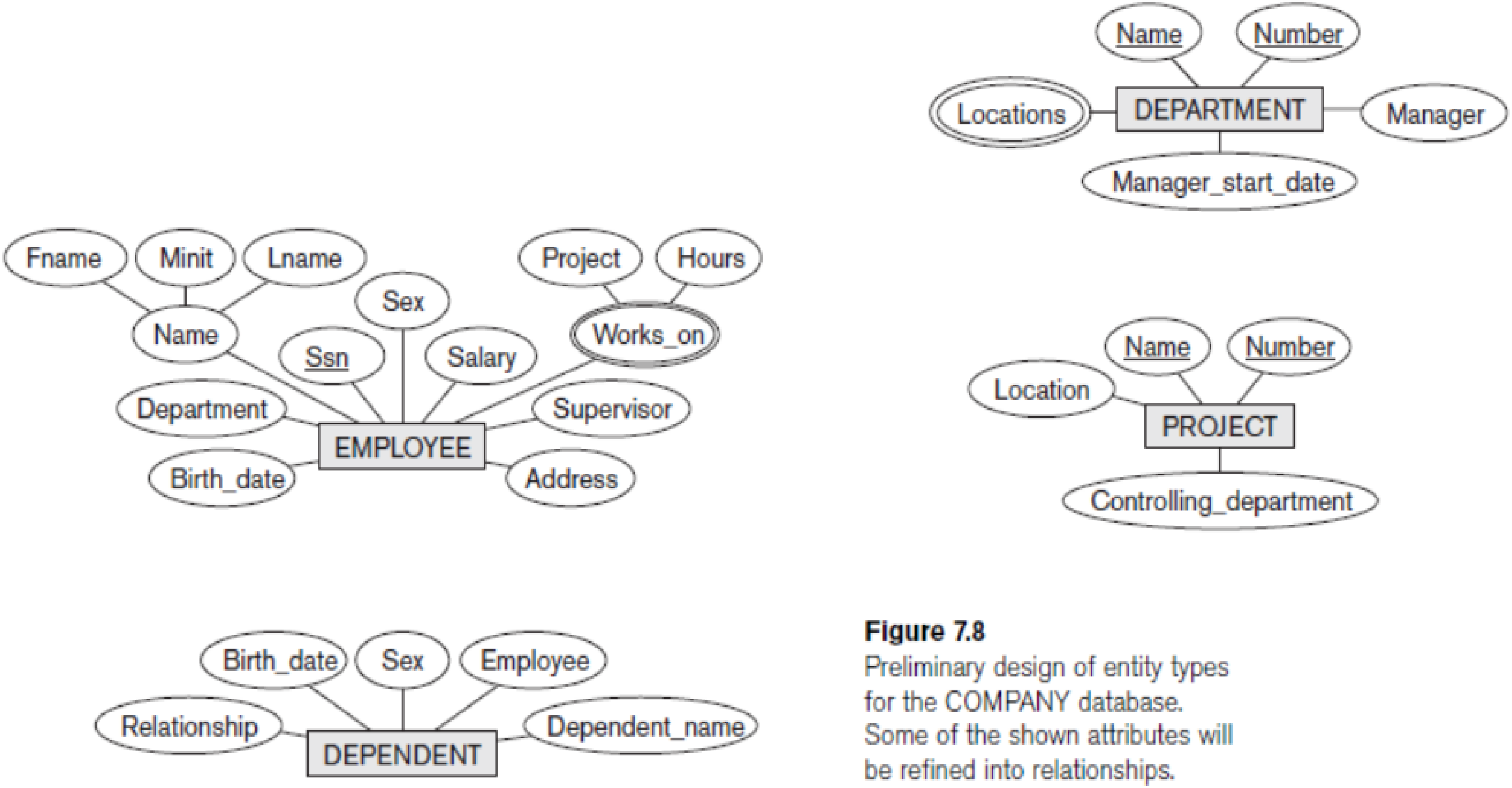
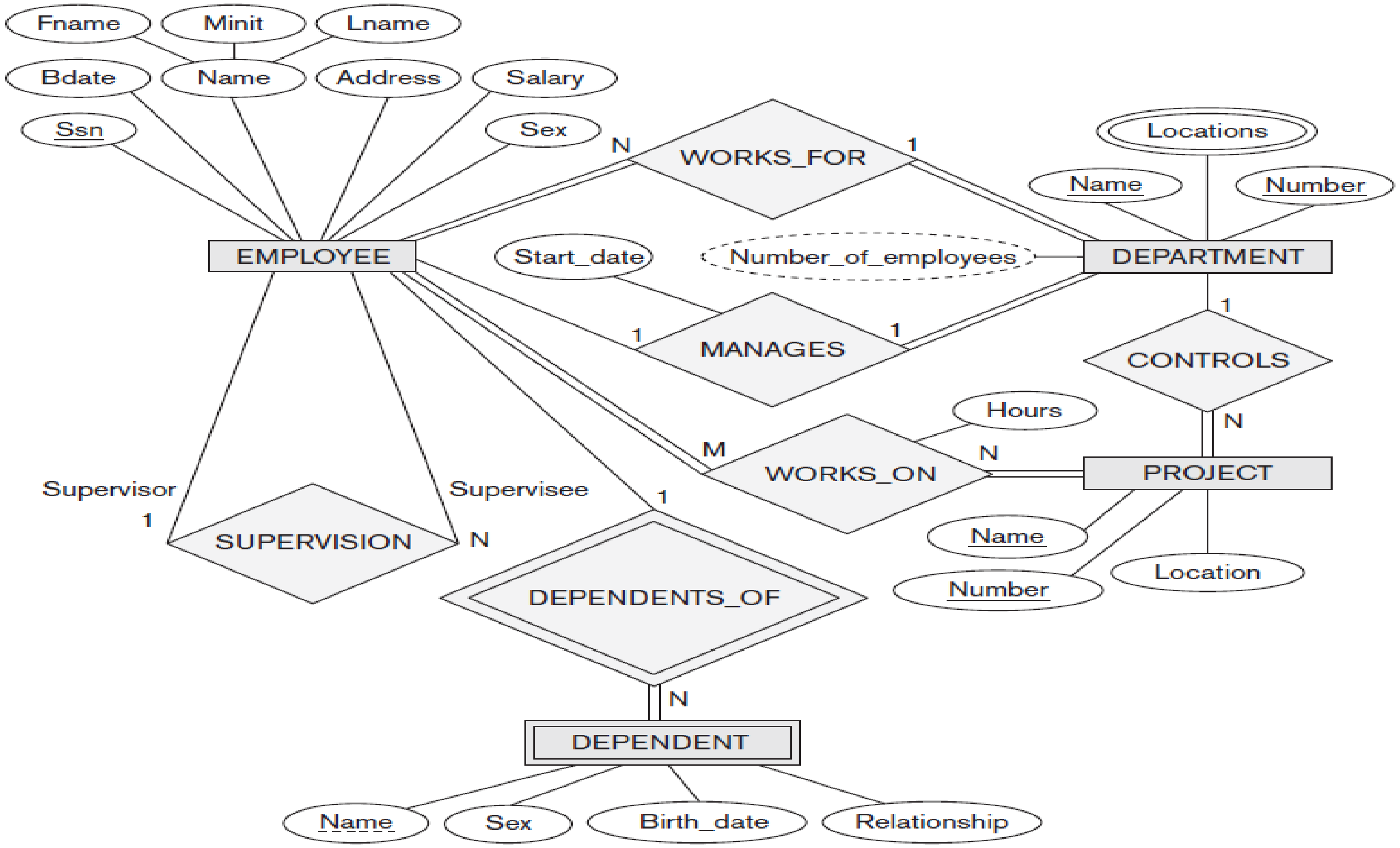


Figure 7.8
Preliminary design of entity types for the COMPANY database. Some of the shown attributes will be refined into relationships.

Company Database – Solution


Fname
Minit
Lname
Bdate
Name
Address
Salary
Ssn
Sex
N
WORKS_FOR
1
Locations
Name
Number
EMPLOYEE
Start_date
Number_of_employees
DEPARTMENT
1
1
MANAGES
1
CONTROLS
Hours
N
M
N
WORKS_ON
PROJECT
Supervisor
Supervisee
1
1
SUPERVISION
N
Name
DEPENDENTS_OF
Location
Number
N
DEPENDENT
Name
Sex
Birth_date
Relationship