Database I

(60012301-1)

Lecture 4: Relationships in (ER) Diagram

Dr. Obead Alhadreti

كلية الحاسب الآلي بالقنفذة
Computing College at AlQunfudah

Review of ER Concepts

- Entity: is a *thing* or *object* in the real world with an *independent existence.*
- Attributes: are properties used to describe an entity.
- Entity Type: a collection of entities that have the same attributes.
- Entity Instance: An entity instance is a single occurrence of an entity.
- Entity Set: The collection of all entities of a particular entity type in the database at any point in time.
- Key Attributes: are attributes whose values are distinct for each individual entity in the entity set.

Identifying Entities

- First, we need to work out what the database needs to store.

- We do this by analyzing the requirements of our database and **working out the nouns** in the sentences.

- In a classroom situation, you will be given the description or requirements by your lecturer. In real life, you need to start your work by talking to your customers, to the potential users of the system, and to domain experts to identify requirements or description of intended database.

Identifying Entities

- Don't worry if you have a hard time identifying entities and attributes.
- When building a data model, it’s common to go through multiple versions before you can achieve the desired model.
- Some nouns start as entities and end up as attributes or vice versa.

Identifying Entities

- For instance, you could be given the following short description of your client's blog:
  We publish many posts. Each post has a specific author and each author has a nickname.

- In our short blog description, we can identify the following nouns: post, author, nickname. These nouns can become separate entities: Post, author, or attributes (nickname can become an attribute for author).

Exercise 1: Identifying Entities

- Let's imagine that university authorities have asked us to prepare a simple system for class management. They gave us the following general description:

  There are many students who take up various courses. All courses are run by lecturers. We have ID numbers for students, courses and lecturers. When a new student comes, we need to know their first name, last name and date of birth. When a new lecturer comes, we write down their first name, last name and their date of employment. All course have names and descriptions so that students know what they cover. We also store the date when the course starts.

Exercise 2: Identifying Entities

- Let's imagine that some authorities have asked us to prepare a census system. They gave us the following general description:

  Find the right entities and their attributes to represent all the countries in the world, their interior regions (which can be called states, provinces, or regions) and their cities. We want to represent each country's name, continent, date of independence, type of government and population. For each region (or province, state, etc.) we wish to store the capital city, the name of the governor, and the population. Finally, for each city we want to have the name, founding date, and population,.

Attribute Types

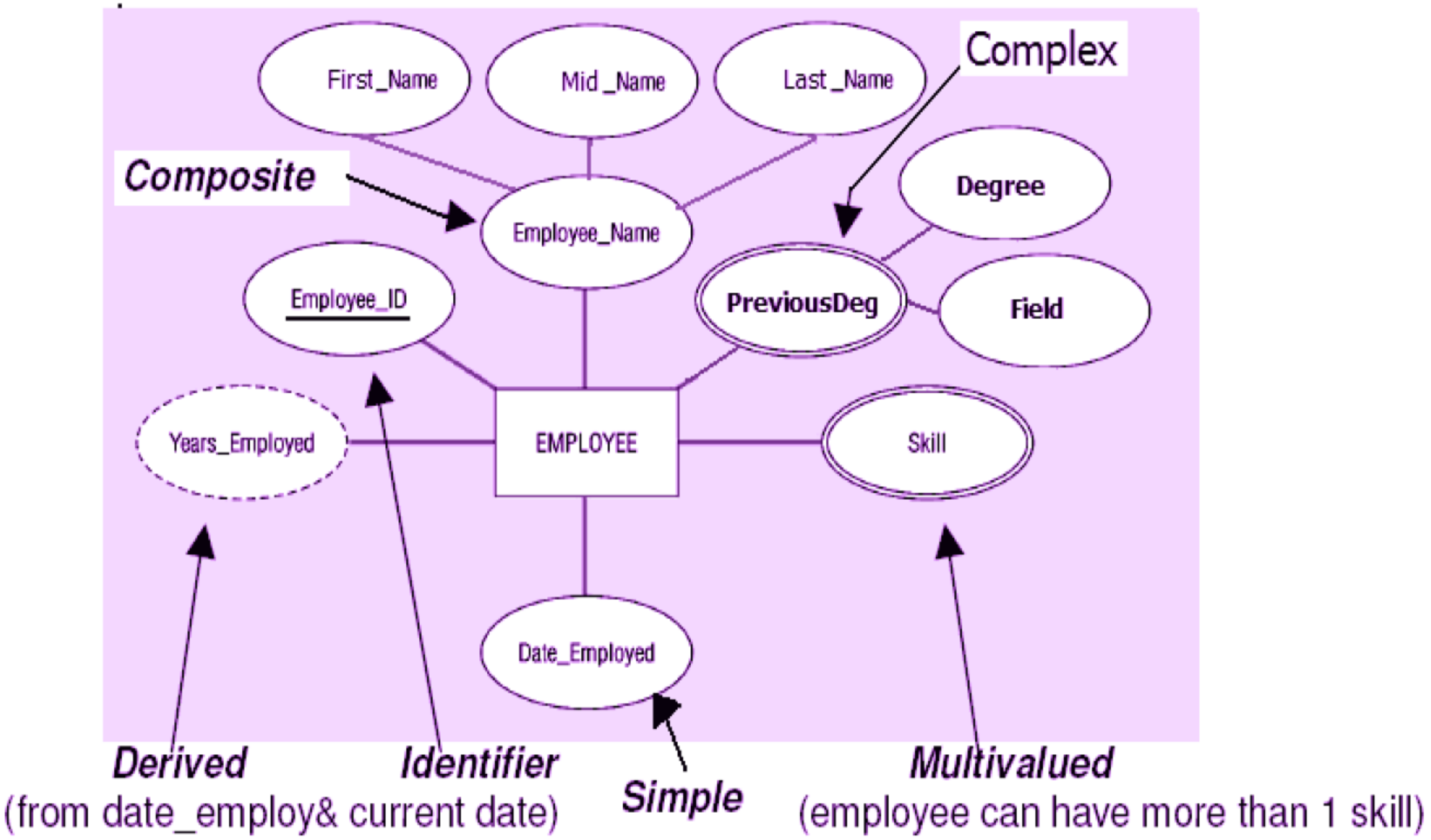
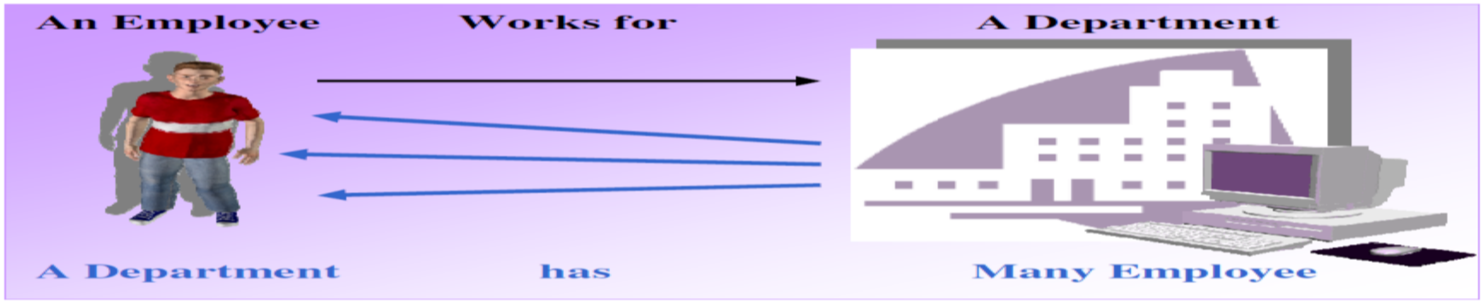


Figure 3.9: Entity Type with all attributes types.

Relationships

Relationships

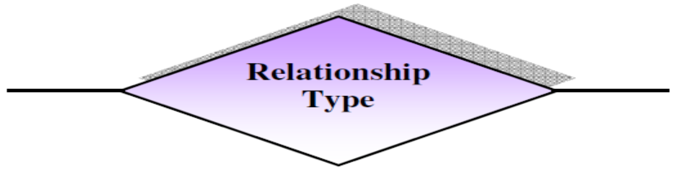
- A relationship: is a relation (association) among at least two entities instances belonging to one or more entity sets. When an attribute of one entity type refers to another entity type, some relationship exist.
- For example, EMPLOYEE Sami works for IT DEPARTMENT.
- Here, "WORKS FOR" is called a relationship.



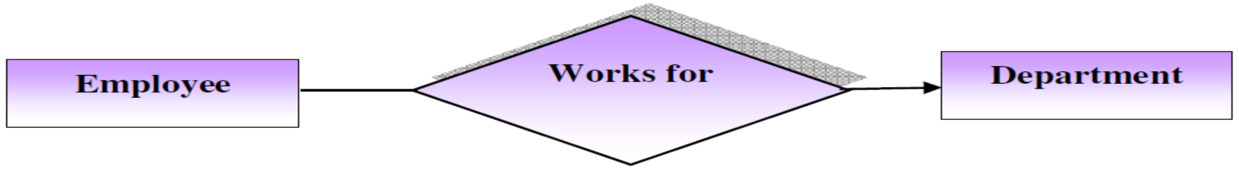
Relationship Type

- **Relationship Type (R):** associates entity types. For example, the "WORKS FOR" is a relationship type in which EMPLOYEEs and DEPARTMENTs participate.

- In ER diagrams, relationship types are displayed as diamond-shaped boxes, which are connected by straight lines to the rectangular boxes representing the participating entity types.

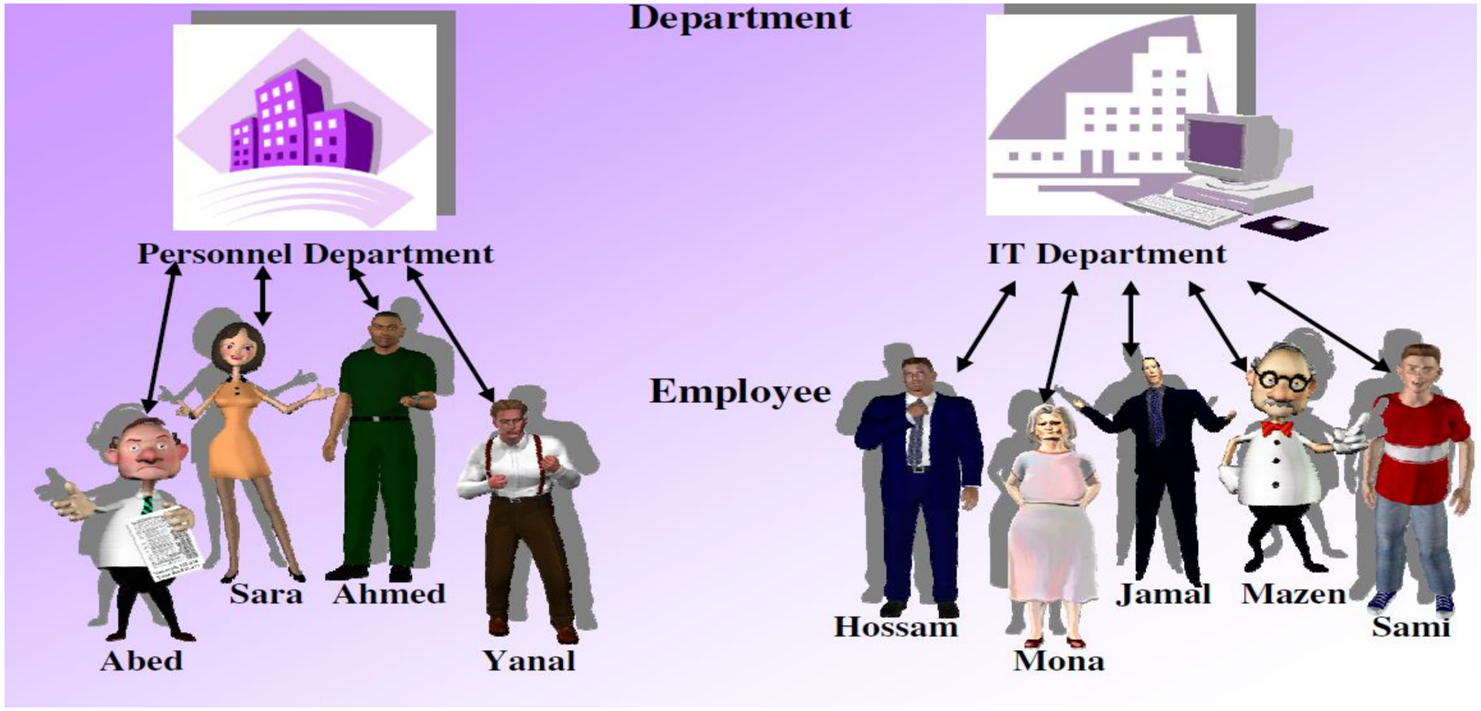


- *Example:*



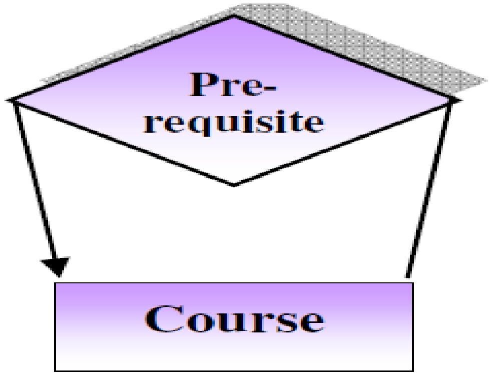
Relationship Set

- Relationship Set: A collection of **relationship instances** (relationships between entities) represented by a relationship type.



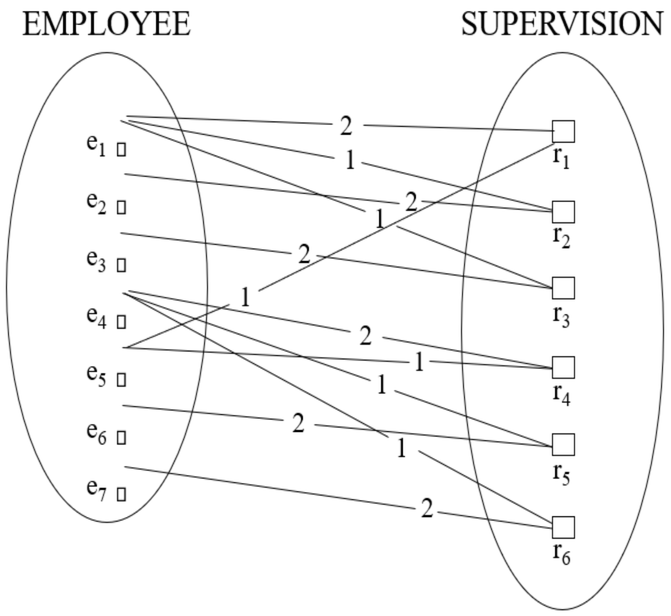
Recursive Relationships

- Relationship types may associate an entity type with itself.
- Both participations are same entity type in different roles.
- In such a case, the **roles** of the entity types in the relationship type are listed on the edges, and the relationship is said to be **recursive.**



Recursive Relationships

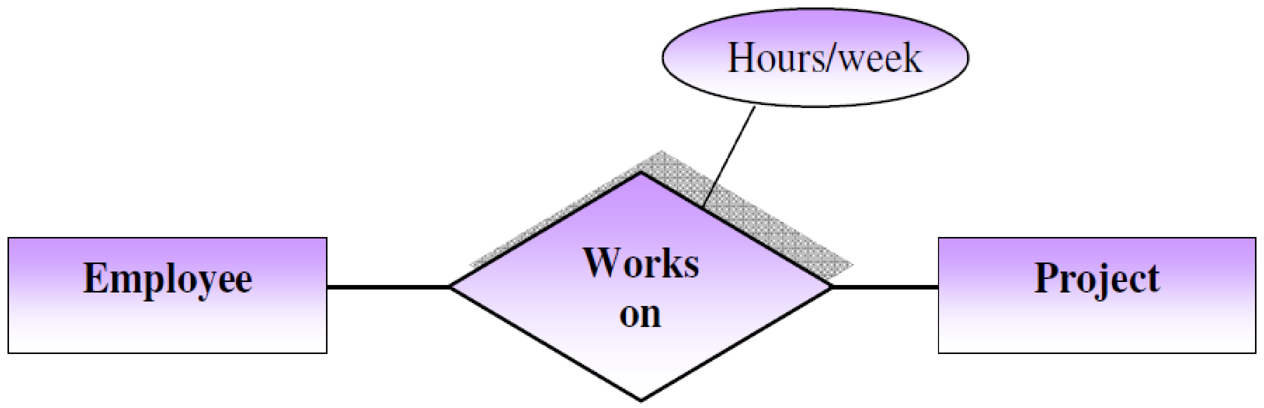
- For example, SUPERVISION relationships between EMPLOYEE (in role of supervisor or boss) and (another) EMPLOYEE (in role of subordinate or worker). In ER diagram, need to display role names to distinguish participations.



Relationship Attribute

- A relationship type can have attributes called *descriptive* attributes.
- For example, HoursPerWeek of WORKS_ON
- Its value for each relationship instance describes the number of hours per week that an EMPLOYEE works on a PROJECT.
- A value of HoursPerWeek depends on a particular (employee, project) combination

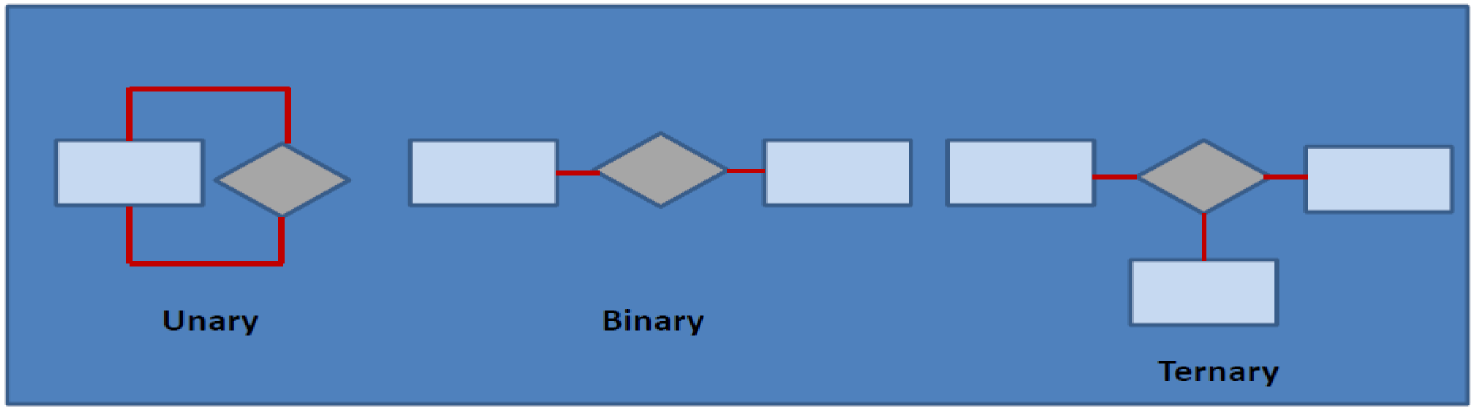
Relationship Attribute



Degree of a Relationship Type

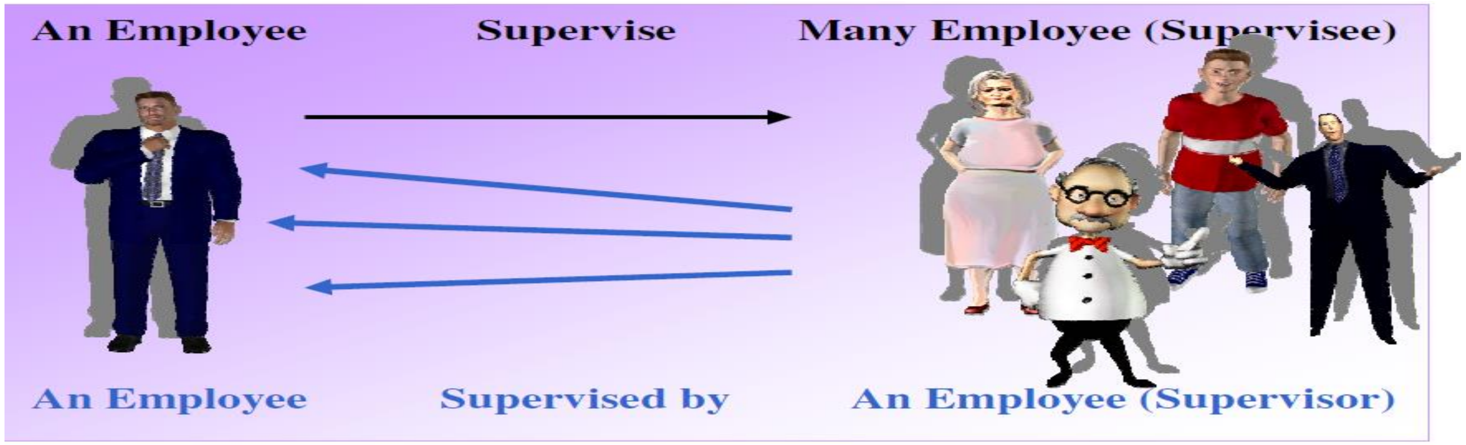
- It is the number of participating entity types.

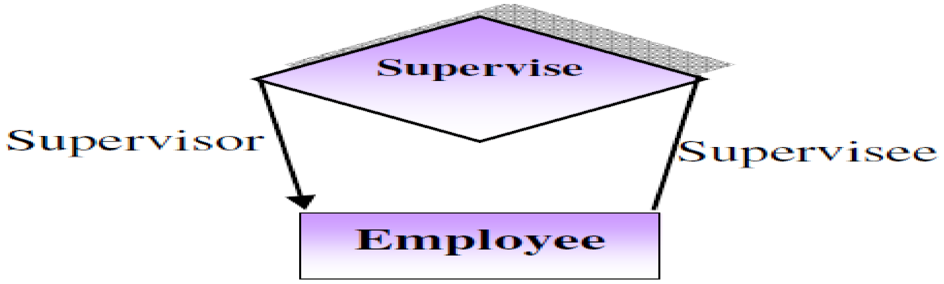
- There are four degrees of a relationship type:
  1. *Unary*
  2. *Binary*
  3. *Ternary*
  4. *n-ary*



Unary Relationship

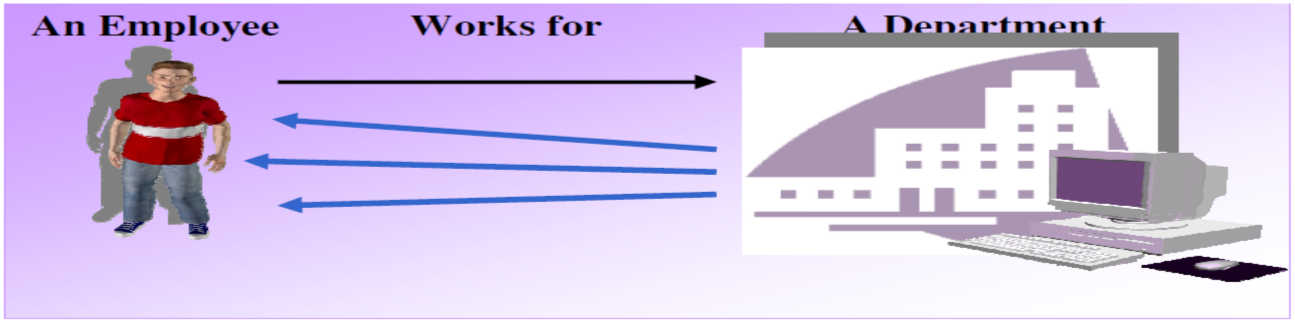
- **Unary:** related to another of the same entity type. Also called recursive relationships.

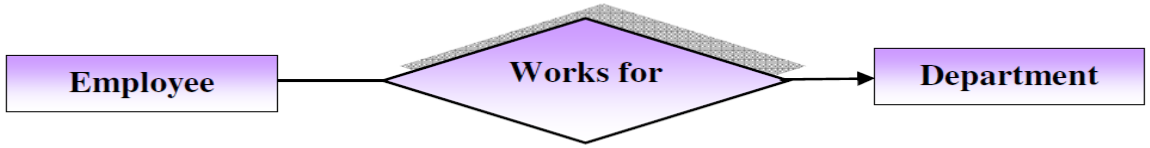




Binary Relationship

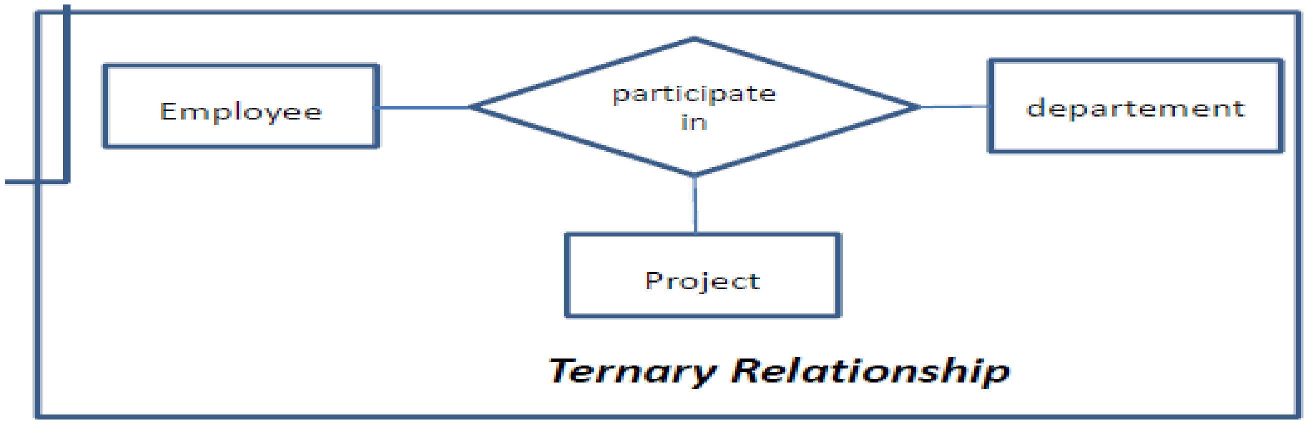
- **Binary:** entities of two different types related to each other. (Two participating entities).





Ternary and *n-ary* Relationships

- **Ternary:** entities of three different types related to each other. (three participating entities).



- ***n-ary:*** entities of more than three different types related to each other.

Structural Constraints on relationship

- Cardinality ratio express the number of entities to which another entity can be associated via a relationship set. (specifies maximum participation).

- There are four types of cardinality: 1:1, 1:N, N:1, or M:N. Shown by placing appropriate numbers on the relationship edges. This is called Chen Style.

- **One-to-One (1:1)** cardinality ratio, an entity in one set is associated with at most one entity in another

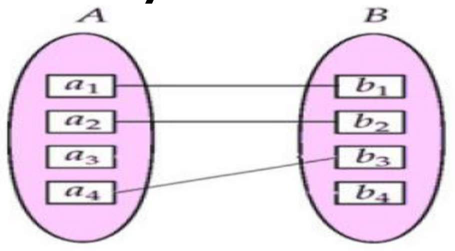
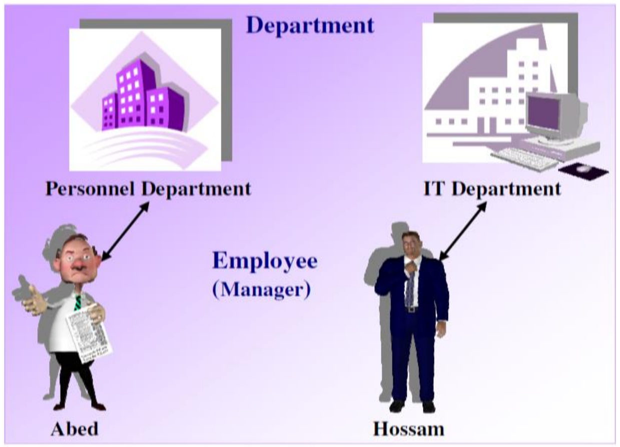


Figure 3.22: One-to-One relationship set.

One-to-One (1:1) cardinality ratio



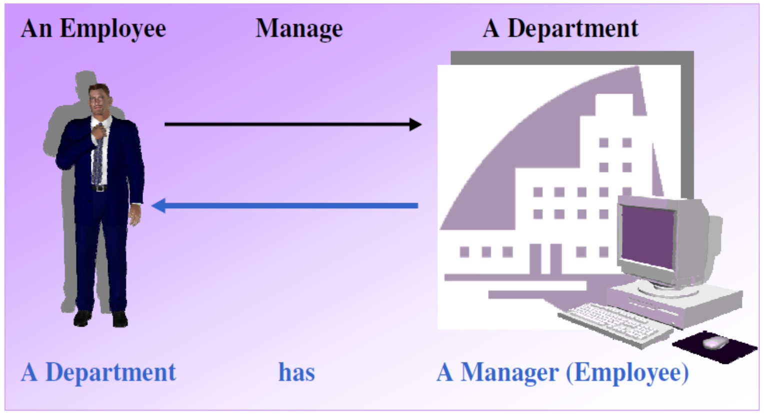
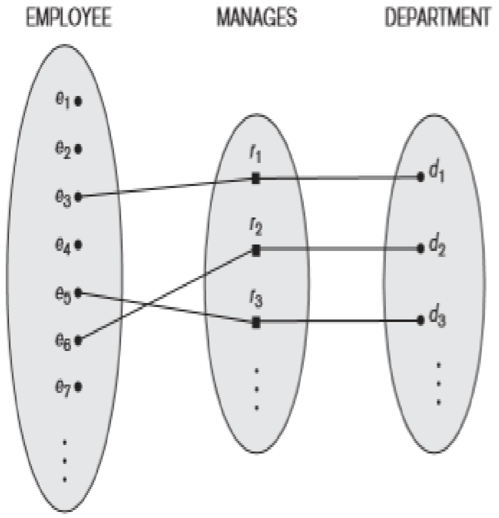
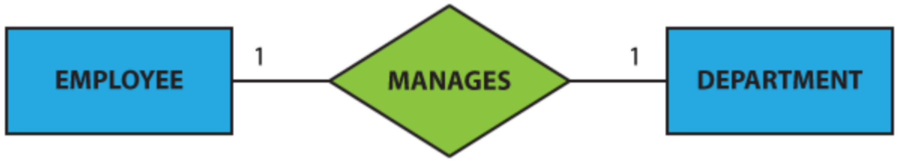


Figure 7.12
A 1:1 relationship, MANAGES.



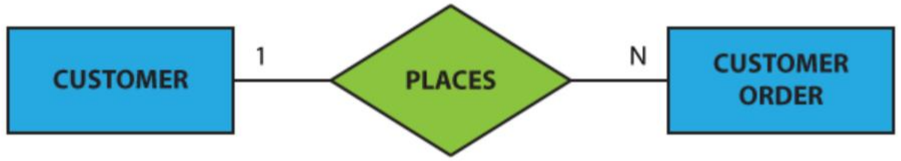


Look at the ERD carefully. Arrow head is used in either side of the relationship Manage

[9]N stands for *any number* of related entities (zero or more).

One-to-many (1: N)

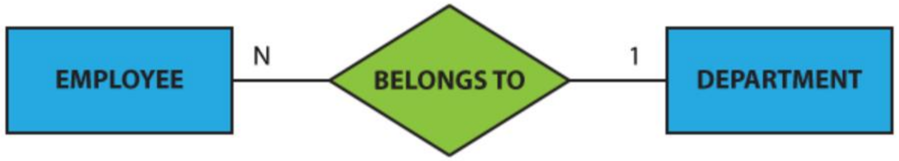
- **One-to-many (1: N)** cardinality ratio, an entity in the first set is associated with 0 or more entities in the second set. However, those entities in the second set can be associated with at most one entity in the first.
- For example: The customer may place many orders, but each order can be placed by one customer only:



Many-to-one (N: 1)

- **Many-to-one (N: 1)** cardinality ratio is just the reverse of the 1:N cardinality ratio.

- For example, many employees may belong to one department, but one particular employee can belong to one department only.



Many-to-many (N:M)

- **Many-to-many (N:M)** cardinality ratio, entities of either set may be associated with any number of entities in the other.

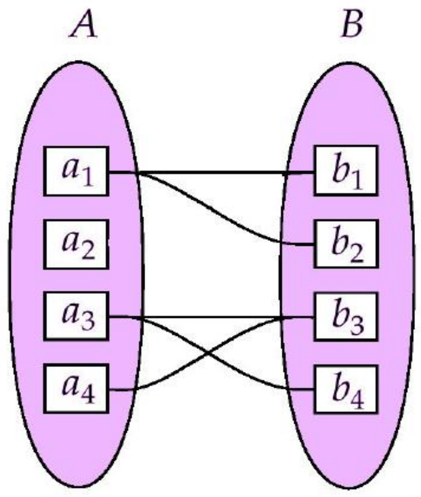
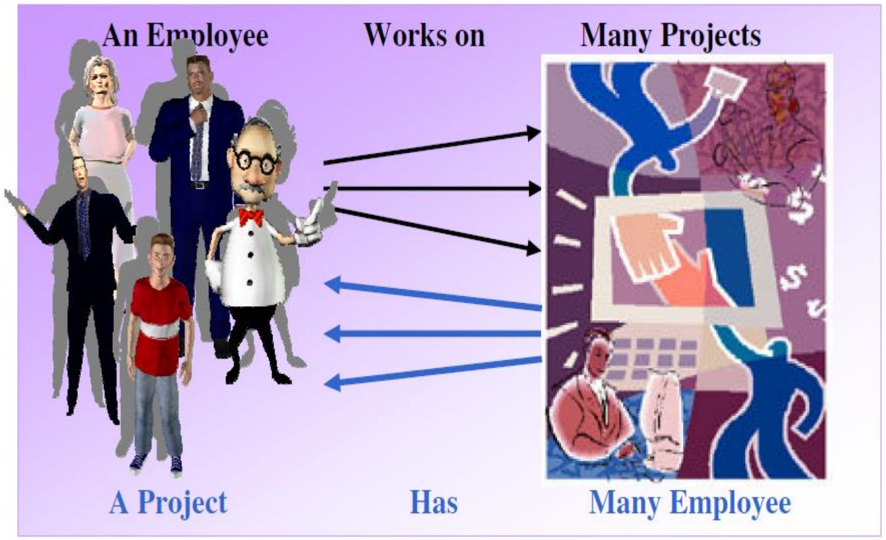
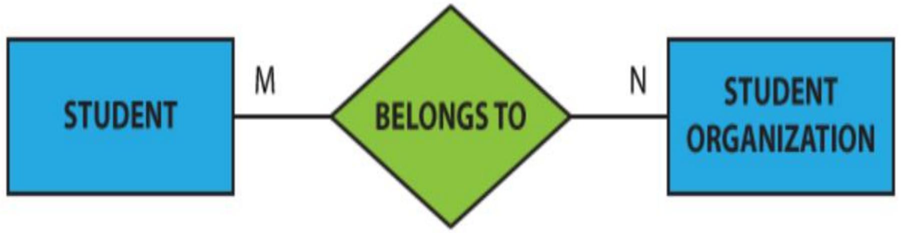


Figure 3.28: Many-to-Many relationship set.



Example

- For example: One student may belong to more than one student organizations, and one organization can admit more than one student:



Selecting Cardinality Ratio

- The cardinality ratio depends on the real-world situation.
- **Example:** there are two situation of the borrower relationship between customer and loan:
  - If a loan can belong to only one customer, and a customer can have several loans, then the relationship type from customer to loan is one-to-many.
  - If a loan can belong to several customers, the relationship type is many-to-many.

Participation Constraints

- Specifies minimum participation

- The participation of an entity set in a relationship set is said to be:

1. **Partial**: if only some entities in entity type participate in relationships in relationship type.
   - Each entity ***can*** participate in a relationship set.
   - Optional participation, not existence-dependent.
   - The minimum value is zero.
   - Shown by a single line.

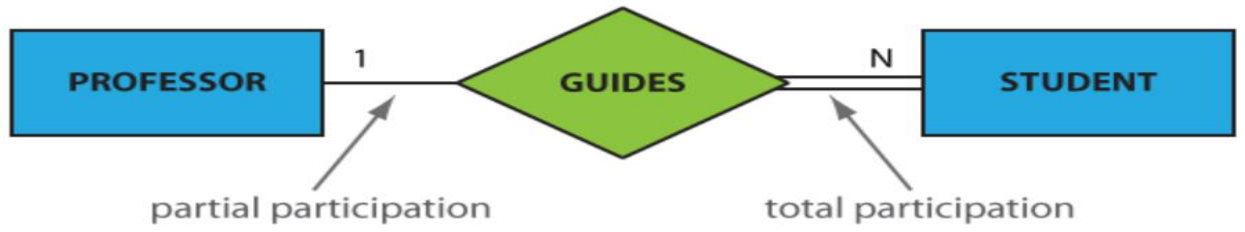
Participation Constraints

2. **Total**: if every entity in entity type participates in at least one relationship in relationship type.

- Each entity ***must*** participate in a relationship set.
- mandatory participation, existence-dependent
- Example: if a company policy states that every employee must work for a department, then an employee entity can exist only if it participates in a works-for relationship instance.

Participation Constraints

- The minimum value is one or more.
- Every entity in E participates in at least one relationship in R
- Shown by double line.

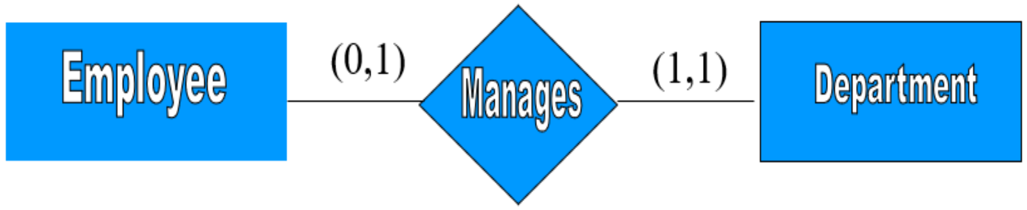


Alternative (min, max) notation for relationship structural constraints

- Specifies that each entity instance in entity type participates in *at least* min and *at most* max relationship instances in relationship type
- Default(no constraint): min=0, max=n
- Must have min≤max, min≥0, max ≥1
- If min=0 then entity participation in a relationship is optional
- If min=1 then entity participation in a relationship is mandatory.

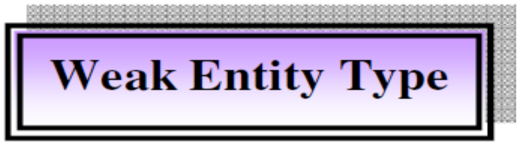
Alternative (min, max) notation for relationship structural constraints

- Example:
- A department has *exactly one* manager and an employee can manage *at most one* department.
  - Specify (0,1) for participation of EMPLOYEE in MANAGES
  - Specify (1,1) for participation of DEPARTMENT in MANAGES

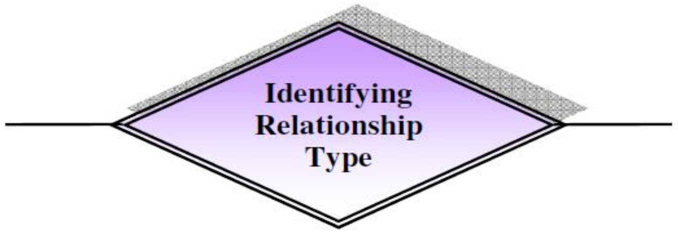


Weak Entity Types

- An entity type that does not have a key attribute.
- It is pictured by

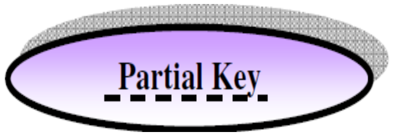


- A weak entity type must participate in an **identifying relationship type** with an **owner** or **identifying entity type**.
- **Identifying (Weak) relationship type** is pictured by diamond with double lines.



Weak Entity Types

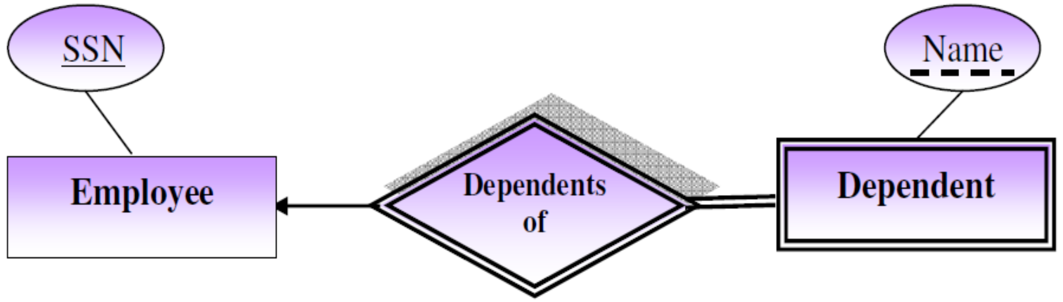
- Entities are identified by the combination of:
- A partial key of the weak entity type. It is underlined with a dashed or dotted line.



- An attribute is a Partial Key if a Key from a related entity type must be used in conjunction with the attribute in question to uniquely identify instances of a corresponding entity set.

- A weak entity type may have more than one identifying entity type and an identifying relationship type of degree higher than two.

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.

Example COMPANY Database

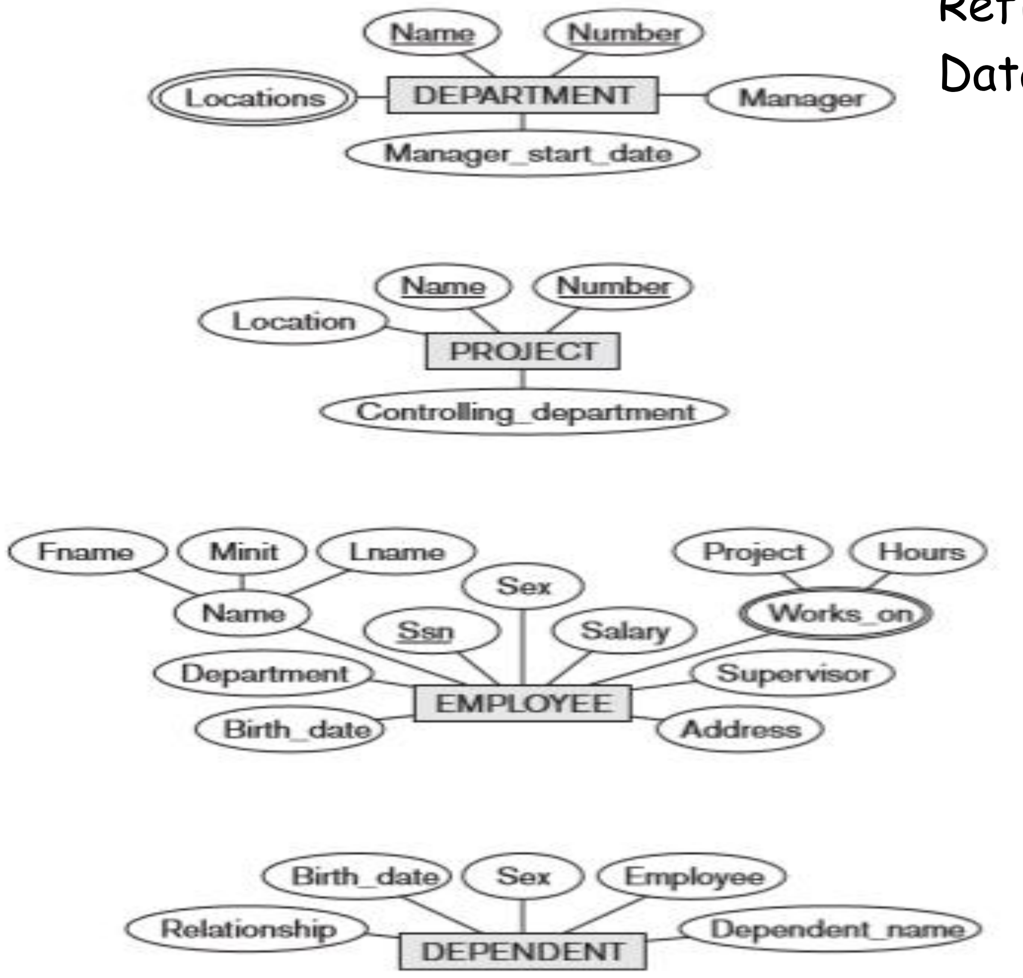
- We need to create a database schema design based on the following (simplified) **requirements** of the COMPANY Database:
  - The company is organized into DEPARTMENTs.
  - Each department has a name, number and an employee who *manages* the department. We keep track of the start date of the department manager. A department may have several locations.
  - Each department *controls* a number of PROJECTs.
  - Each project has a unique name, unique number and is located at a single location.

Example COMPANY Database (Contd.)

- We store each EMPLOYEE's social security number, address, salary, sex, and birthdate.
  - Each employee *works for* one department but may *work on* several projects.
  - We keep track of the number of hours per week that an employee currently works on each project.
  - We also keep track of the *direct supervisor* of each employee.

- Each employee may *have* a number of DEPENDENTs.
  - For each dependent, we keep track of their name, sex, birthdate, and relationship to the employee.

Refining the initial design by introducing **relationships**



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

Refining the initial design by introducing **relationships**

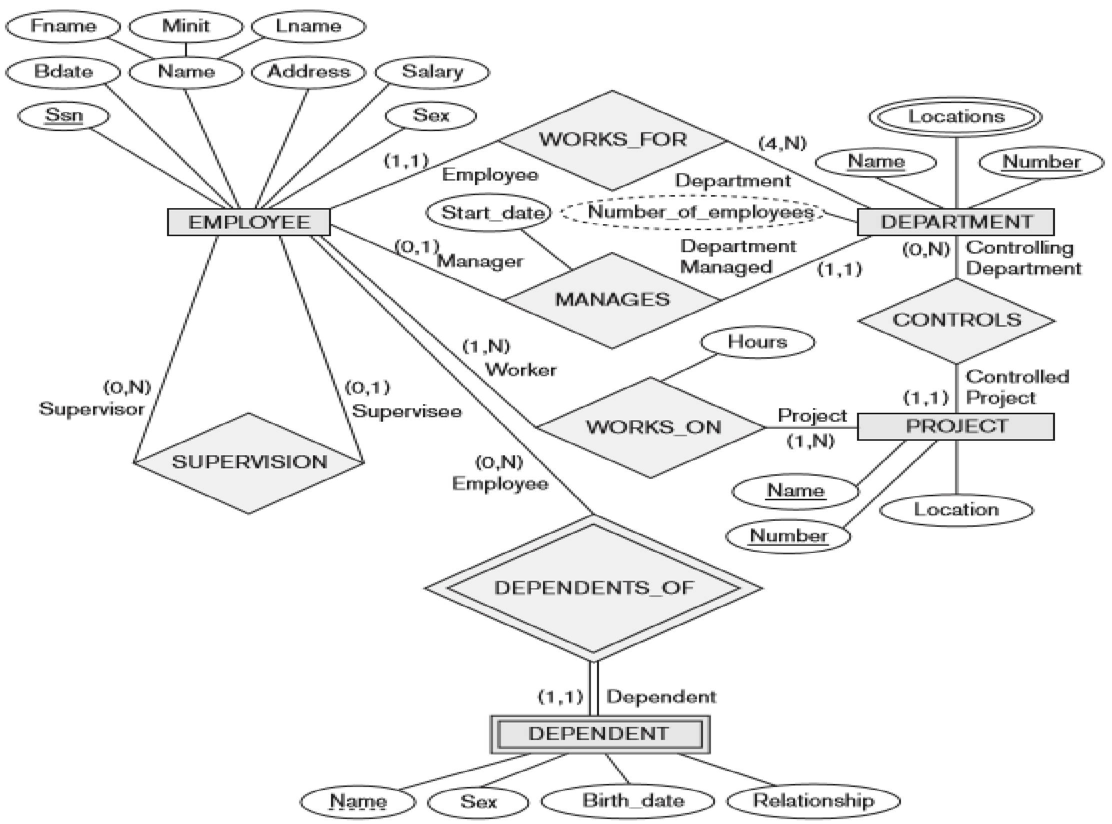


Figure 3.15
ER diagrams for the company schema, with structural constraints specified using (min, max) notation and role names.