Lesson 3:
Data Modeling Using Entity-Relationship Model

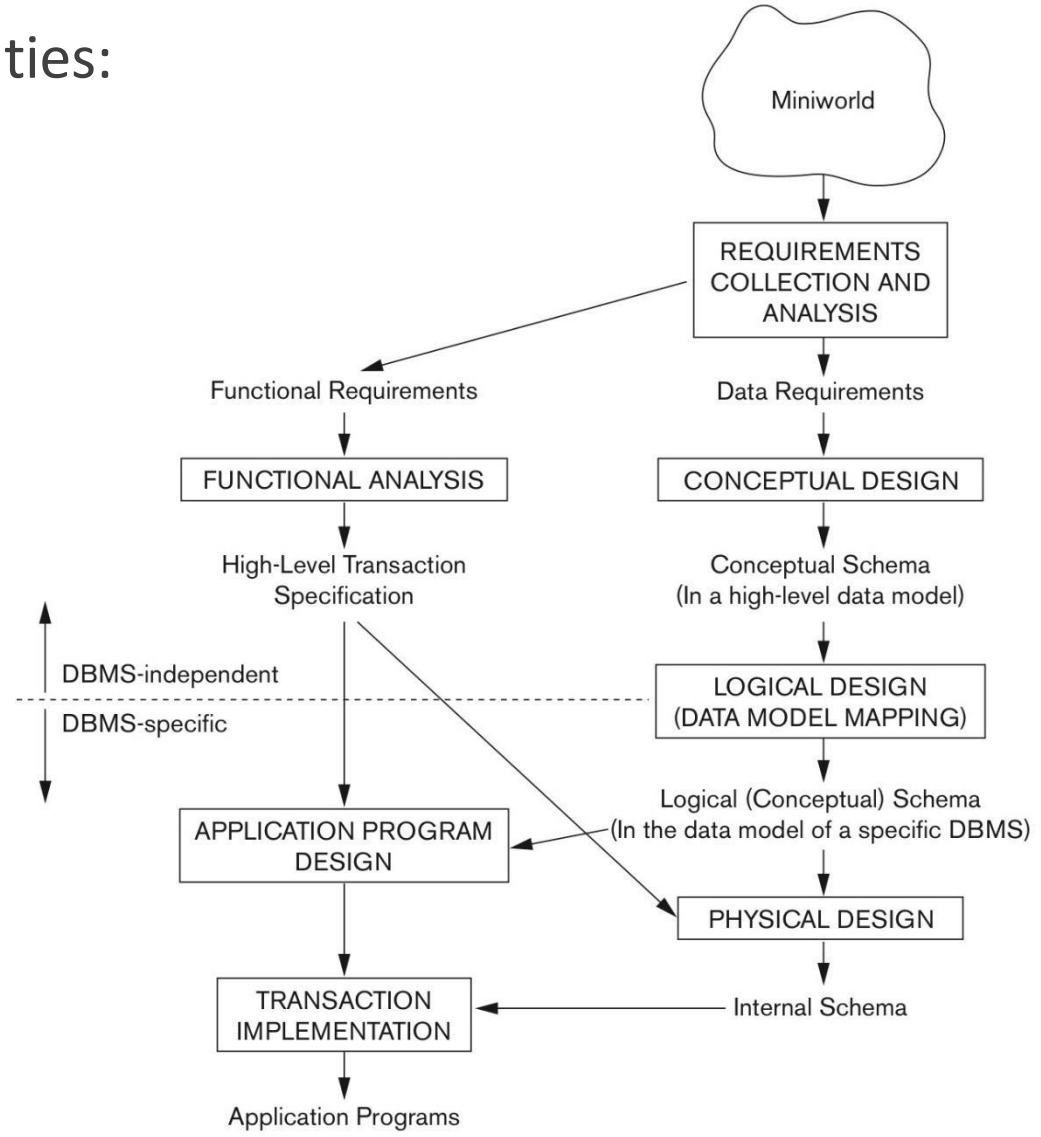
CSC430/530 – DATABASE MANAGEMENT SYSTEMS

OUTLINE

- Database design process.
- Entity-relationship model basic concepts.
- Entities.
- Attributes.
- Relationships.

OVERVIEW OF DATABASE DESIGN PROCESS (1)

- **Database system design** is divided into two main activities:
  - **Database design**.
    - Conceptual design.
    - Logical design.
    - Physical design.
  - **Applications design**.
    - Functional analysis.
    - Application program design.
    - Transaction implementation.



Main phases of database system design

OVERVIEW OF DATABASE DESIGN PROCESS (2)

- **Requirement collection & analysis**.
  - Database designer interviews database users to understand and document detailed **requirements**.
  - Result = **data** and **functional requirements**.
- **Conceptual design**.
  - Creation of the **conceptual schema** using high-level conceptual data model.
    - Conceptual schema is a concise **description** of data requirements (**entity types**, **relationships** & **constraints)**.
- **Functional analysis**.
  - High-level user **queries** and **operations** are specified using basic data model operations.
  - Result = **transaction specification**.

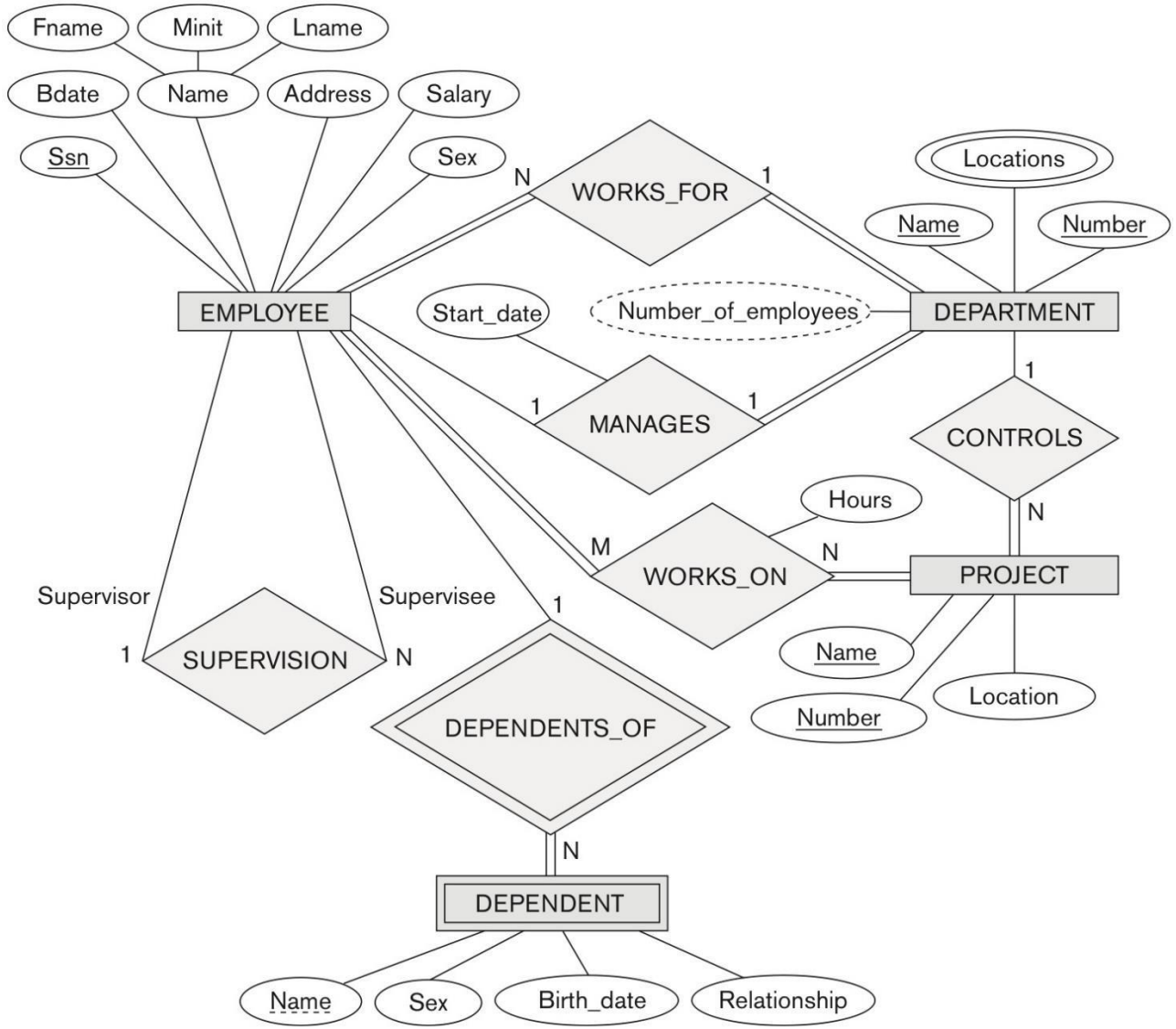
OVERVIEW OF DATABASE DESIGN PROCESS (3)

- **Logical design** (*data model mapping*).
  - Transformation of conceptual schema from **high-level** data model into **implementation** data model.
    - **Implementation** of the database using DBMS software.
  - Result = **logical** (conceptual) **schema**, expressed as a data model of chosen DBMS.
- **Physical design**.
  - Specification of **internal storage structures**, **file organization**, **indexes**, **access paths**, etc.
  - Result = **internal schema**.
- **Application program design** and **transaction implementation**.
  - Design and implementation of application programs as database transactions corresponding to the high-level transaction specifications.
  - Result = **application programs**.

EXAMPLE COMPANY DATABASE (1)

- The aim is to design a **database schema** based on the following **requirements**:
  - The company is organized into departments. Each department has unique name, 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 unique name, 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 one 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 name, sex (gender), birth date, and relationship to the employee.

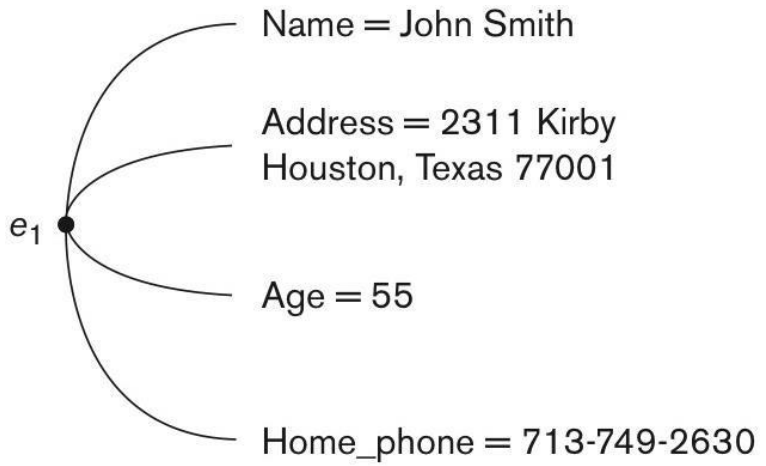
EXAMPLE COMPANY DATABASE (2)



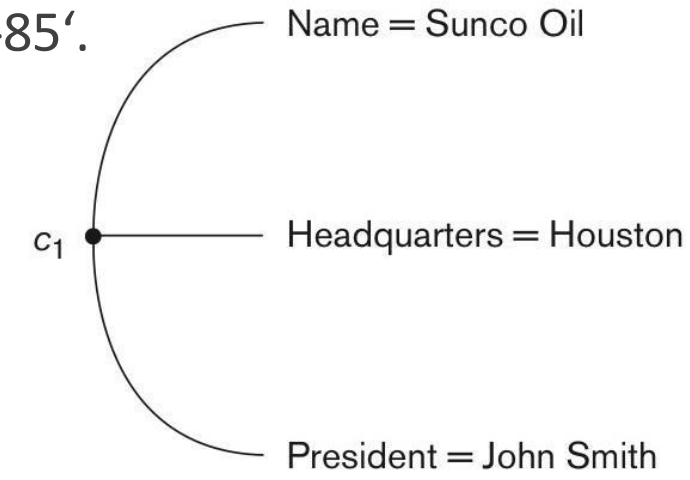
ER diagram for company database

BASIC CONCEPTS OF ENTITY-RELATIONSHIP MODEL

- Two basic **concepts** of **entity-relationship** (ER) model: **entity** & **attribute**.
- **Entity** - **object** of the mini-world, represented in the database.
  - *EMPLOYEE* John Smith, Research *DEPARTMENT*, ProjectX *PROJECT*.
- **Attribute** - particular **property** that describes an entity.
  - Name, SSN, address, sex, date of birth of an EMPLOYEE.

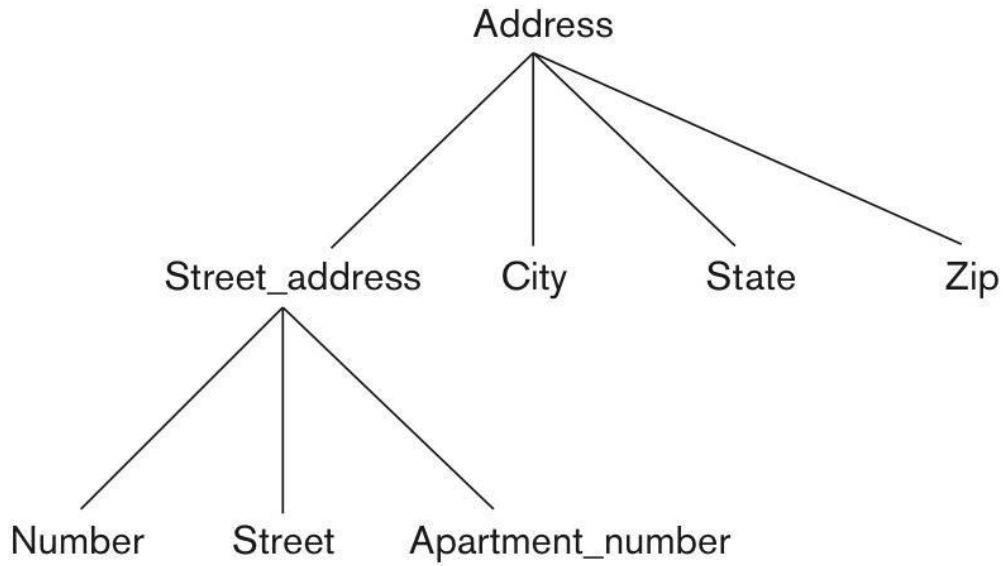


- **Entity** has a **value** for each **attribute**.
  - Name='John Smith', SSN='123456789',
    Address ='600, Dan Reneau Dr, Ruston, LA', Sex='M', BirthDate='09-JAN-85'.
- Each **attribute** has a specific **value set**.
  - *Name* – character string of up to 15 characters.
  - *SSN* – list of nine integers.
- In **ER diagram**: entity – rectangle, attribute – oval .



TYPES OF ATTRIBUTES (1)

- **Attributes** are grouped by following **types**:
  - **Simple** (*atomic*).
    - Each entity has a **single** atomic value for the attribute.
      - Ex.: SSN or Sex.
  - **Composite.**
    - Divided into **smaller subparts** - more basic attributes with independent meaning.
      - Address {Apt#, House#, Street, City, State, ZipCode, Country}
      - Name {FirstName, MiddleName, LastName}
    - Represented by **{ }**.
  - **Multi-valued**.
    - Entity may have **multiple** values for an attribute.
      - *Colors* of a CAR or *PreviousDegrees* of a STUDENT.
    - Represented by **( )**.
    - Illustrated as **double ovals** in ER diagram.



Hierarchy of composite attributes

TYPES OF ATTRIBUTES (2)

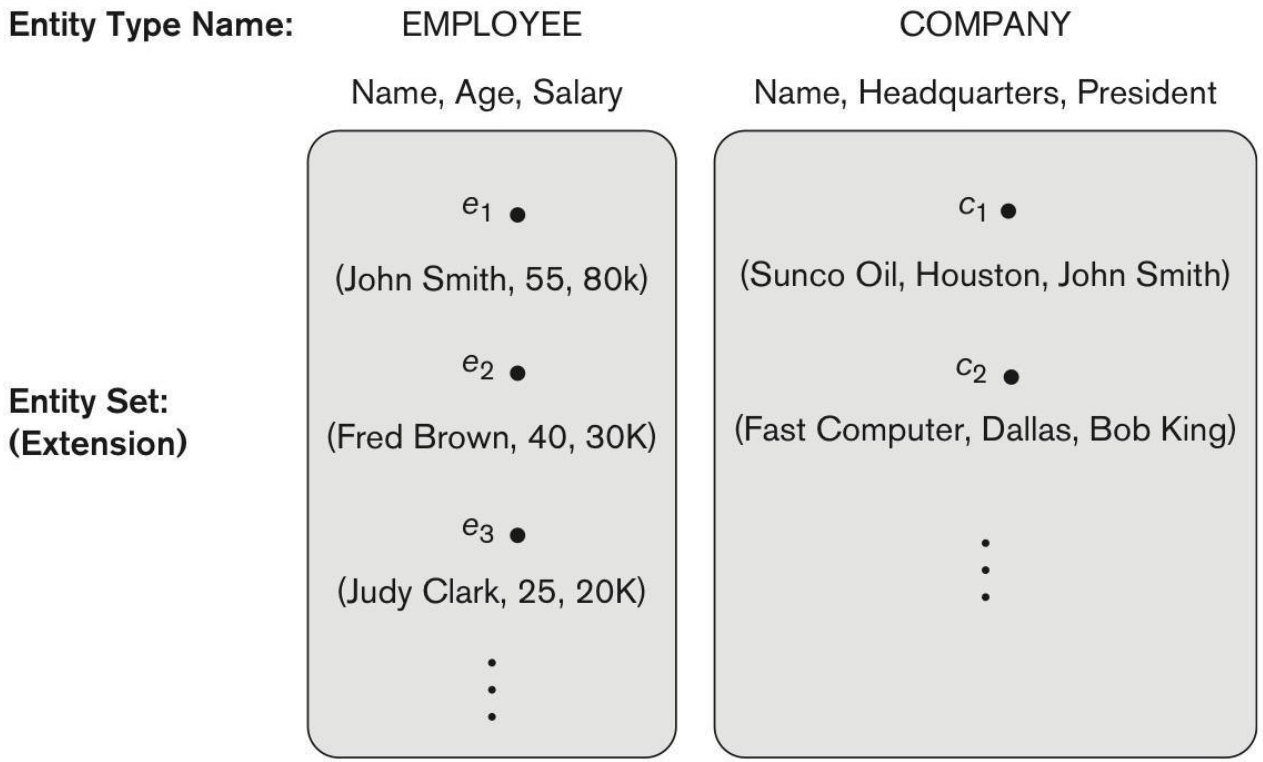
- **Composite** & **multi-valued** attributes are **complex** attributes and could be nested.
  - *PreviousDegrees* of a STUDENT is a composite multi-valued attribute.
    - (PreviousDegrees {College, Year, Degree, Field})
      - Multiple PreviousDegrees values can exist.
      - Each has four subcomponent attributes: College, Year, Degree, Field.
- **Derived attributes**.
  - Can be **derived** by using a **value** of another attribute.
    - Ex.: *Age* can be derived from current date and the value of *DateOfBirth* attribute.
  - Some attribute values can be derived from **related entities**.
    - *NumberOfEmployees* of DEPARTMENT can be derived by counting number of employees related to that department.

TYPES OF ATTRIBUTES (3)

- **NULL values**.
  - Used when a particular entity **does not have** an applicable value for an attribute.
    - **Not applicable**.
      - *ApartmentNumber* will be NULL if person lives in a house.
      - *CollegeDegree* will be NULL if employee does not have a degree.
    - **Unknown**.
      - Exists, but missing.
        - *Height* for PERSON.
      - Not known if exists.
        - *HomePhone* for EMPLOYEE.

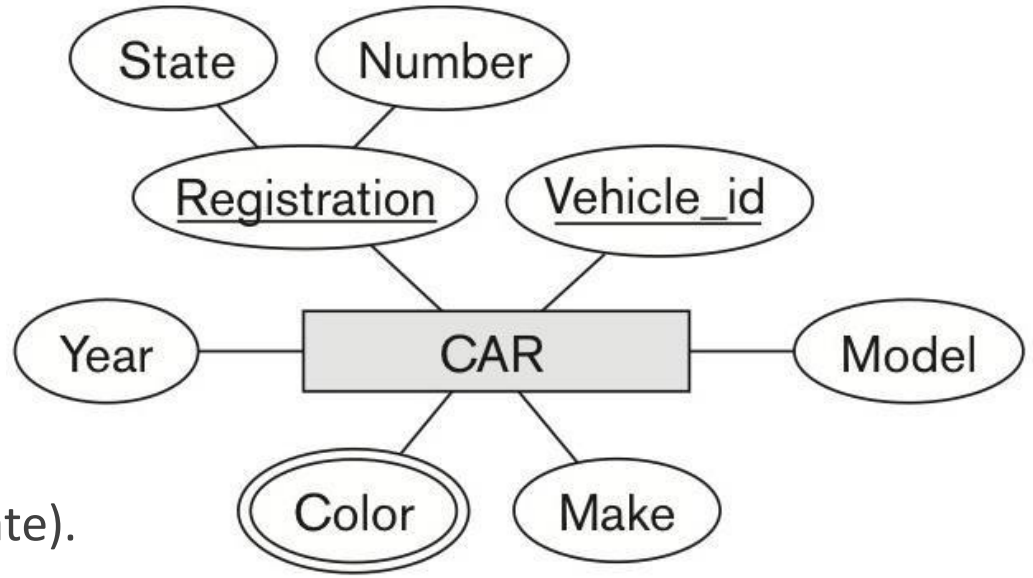
ENTITY TYPES & ENTITY SETS

- **Entity type.**
  - Defines a **collection** of entities with same **attributes**.
  - Each entity type is described by the **name** and **attributes**.
  - Displayed as a **rectangular box** in ER diagrams.
- **Entity set.**
  - The **collection** of **all** entities of a particular type.



KEY ATTRIBUTE OF ENTITY TYPE

- Each **entity type** must have a **key attribute***.
- **Key attribute**.
  - One or more attributes whose values are **distinct** for each individual entity in the entity set.
    - Ex.: SSN of EMPLOYEE.
- **Key attribute** may be **composite**.
  - *Registration* - key of CAR entity type composed of (Number, State).
- **Entity type** may have **more than one** key.
  - Ex.: CAR entity type has two keys:
    - *VehicleIdentificationNumber* - VIN numbers.
    - *VehicleTagNumber* (Number, State) - license plate number.
- **Key attribute** of an entity type must follow **uniqueness** property for every entity in a set.



CAR entity type with two key attributes

INITIAL DESIGN OF COMPANY DATABASE (1)

- Company database 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 one 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 name, sex, birth date, and relationship to the employee.

INITIAL DESIGN OF COMPANY DATABASE (1)

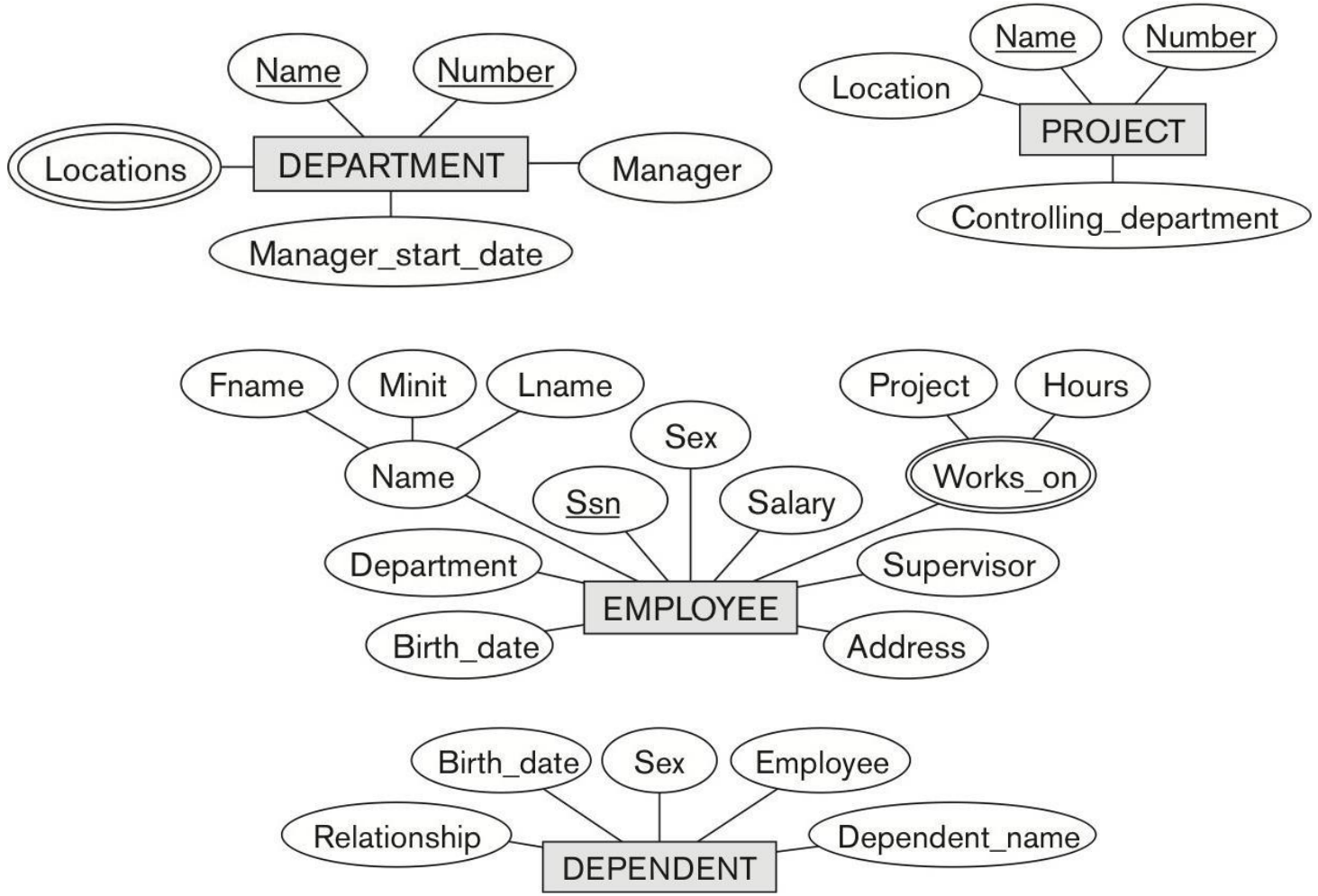
- Company database requirements:
  - The company is organized into departments. Each department has a unique name, a unique number, and a particular employee who manages the department. We keep track of the start date when that employee began managing the department. A department may have *several* locations.
  - 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 *one* 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 name, sex, birth date, and relationship to the employee.

INITIAL DESIGN OF COMPANY DATABASE (2)

- Based on the requirements, we can identify four initial **entity types** in the COMPANY database:
  - DEPARTMENT
    - Name (key1), Number (key2), Manager, Locations (multi-valued), Manager_start_date
  - PROJECT
    - Name (key1), Number (key2), Location, Controlling_department
  - EMPLOYEE
    - Name (composite), SSN (key), Sex, Address, Salary, Birth_date, Department, Supervisor
  - DEPENDENT
    - Employee, Dependent_name, Sex, Birth_date, Relationship

INITIAL DESIGN OF COMPANY DATABASE (2)

- Based on the requirements, we can identify four initial **entity types** in the COMPANY database:
  - DEPARTMENT
    - Name (key1), Number (key2), Manager, Locations (multi-valued), Manager_start_date
  - PROJECT
    - Name (key1), Number (key2), Location, Controlling_department
  - EMPLOYEE
    - Name (composite), SSN (key), Sex, Address, Salary, Birth_date, Department, Supervisor
  - DEPENDENT
    - Employee, Dependent_name, Sex, Birth_date, Relationship



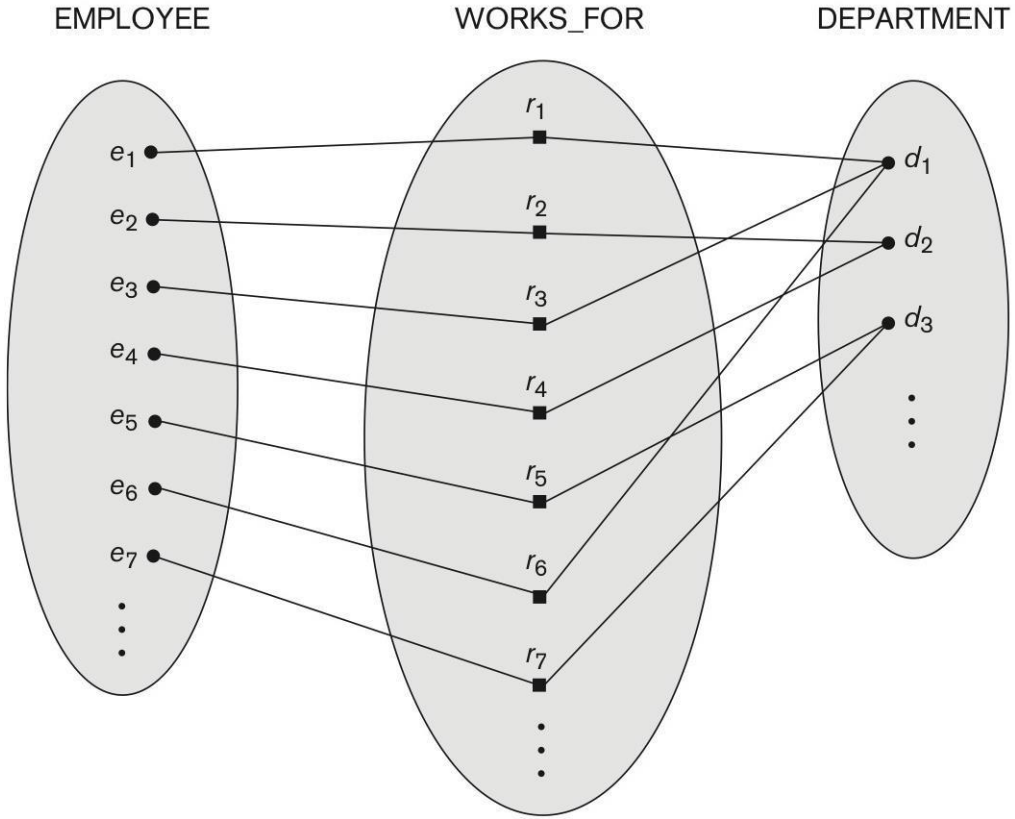
Initial ER diagram for company database

RELATIONSHIP TYPES & RELATIONSHIP SETS (1)

- Another crucial **concept** of ER model is a **relationship**.
- **Relationship** relates two or more **distinct entities** with a specific **meaning**.
  - Ex.: EMPLOYEE John Smith *works on* the ProductX PROJECT, or EMPLOYEE Franklin White *manages* the Research DEPARTMENT.
- **Relationship type** defines a collection of **relationships** among **entities** from entity types.
  - *WORKS_ON* relationship type between EMPLOYEE and PROJECT entity types.
  - *MANAGES* relationship type between EMPLOYEE and DEPARTMENT entity types.
- **Relationship set** - current **set** of **relationship instances** represented in the database.
  - The current **state** of a relationship type.
- In ER diagrams, relationship types are **diamond-shaped boxes** connected by straight lines to **rectangular boxes** (participating entity types).

RELATIONSHIP TYPES & RELATIONSHIP SETS (2)

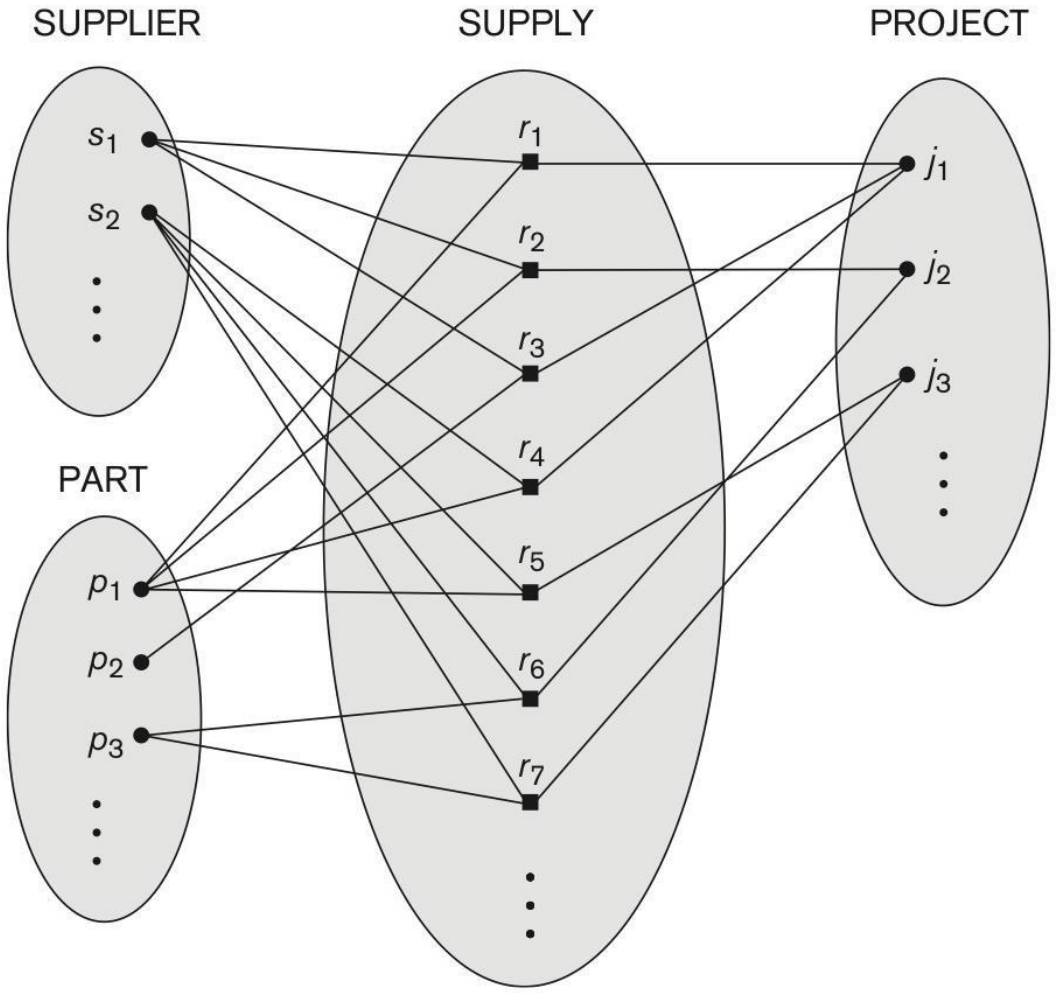
- *WORKS_FOR* relationship between EMPLOYEE and DEPARTMENT entity types.
  - **Associates** each employee with the department for which employee works.
  - Each **relationship instance** in the **relationship set** WORKS_FOR associates one EMPLOYEE entity and one DEPARTMENT entity.



WORKS_FOR relationship

RELATIONSHIP DEGREE

- Each **relationship type** is characterized by a **degree**.
  - **Degree - number** of **participating** entity types.
    - Degree of **two** = **binary** relationship.
    - Degree of **three** = **ternary** relationship.



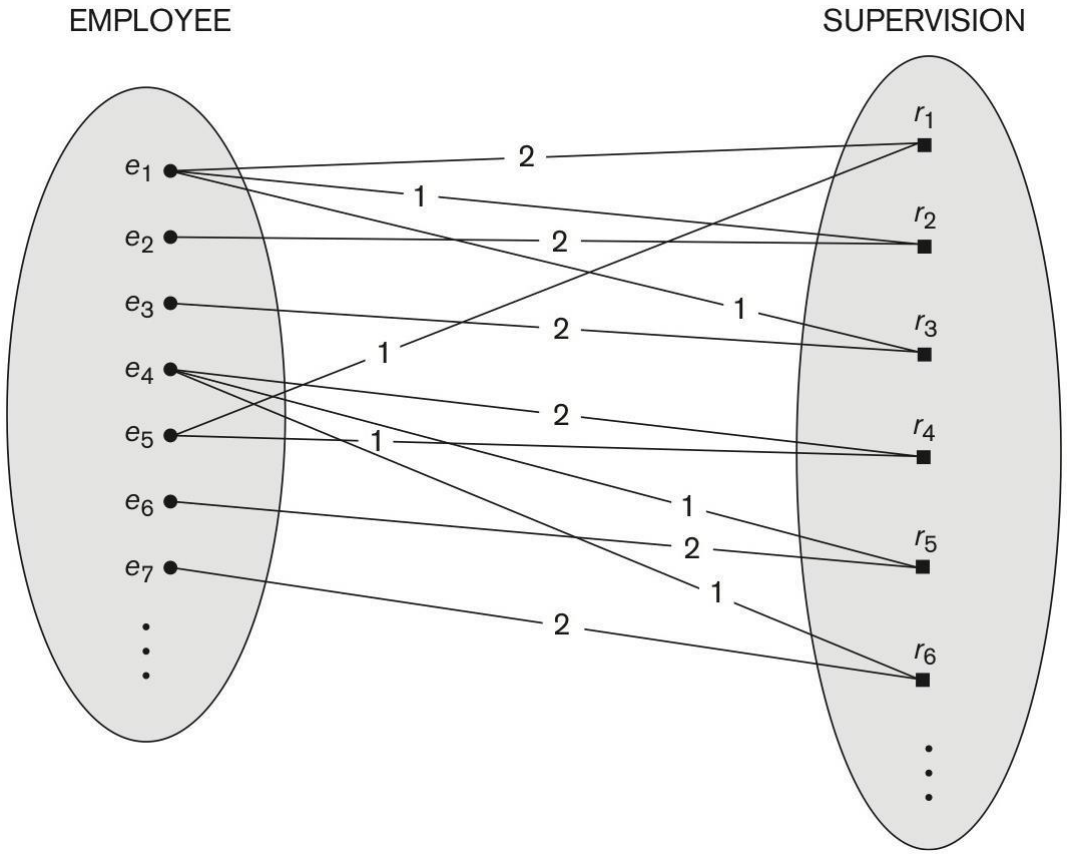
SUPPLY ternary relationship

RELATIONSHIPS AS ATTRIBUTES

- **Binary relationships** can be represented as **attributes** of entity types.
- *WORKS_FOR* relationship between EMPLOYEE and DEPARTMENT entity types can be expressed as an **attribute**:
  - Attribute *Department* of EMPLOYEE entity type.
    - References to the DEPARTMENT entity for which employee works.
      - Values – set of all DEPARTMENT entities (entity set).
  - Multi-valued attribute *Employees* of DEPARTMENT entity type.
    - Reference to the EMPLOYEE entities that work for a department.
      - Values – set of all EMPLOYEE entities who work for that department.
- Either of attributes (*Department* of EMPLOYEE / *Employee* of DEPARTMENT) can represent WORKS_FOR relationship type.
  - If both are presented they are constrained to be **inverse** of each other.

ROLE NAMES & RECURSIVE RELATIONSHIPS

- **Recursive relationship**.
  - **Same** entity type can participate more than once in the relationship type in **different roles**.
  - **Role names** are essential in recursive relationships to distinguish a meaning of the role.
- **Role names**.
  - **Signify** the **role** that a participating entity from the entity type plays in each relationship instance.
  - Helps explaining the **meaning** of a relationship.



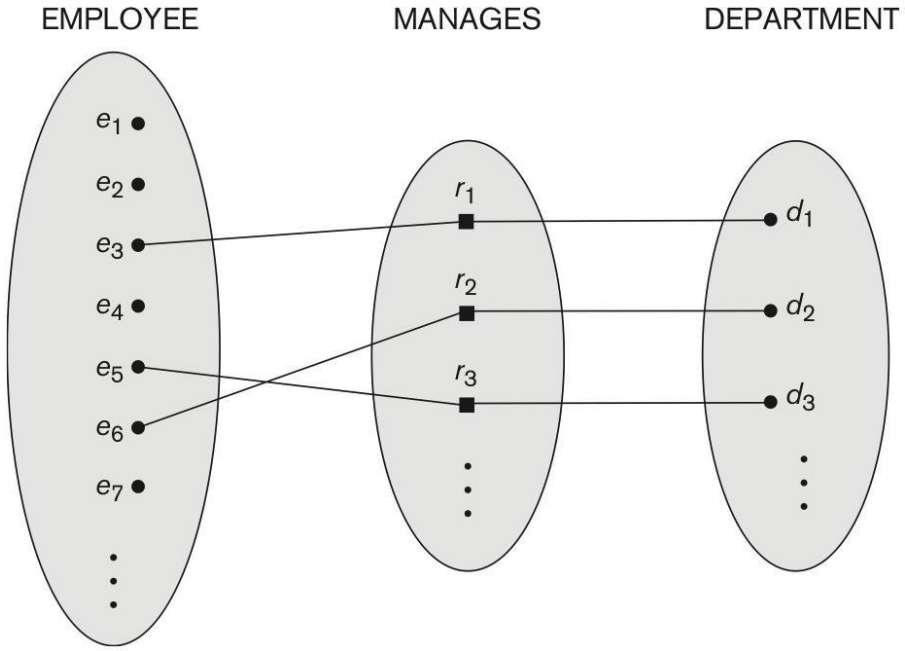
Recursive relationship SUPERVISION

CONSTRAINTS ON BINARY RELATIONSHIP TYPES

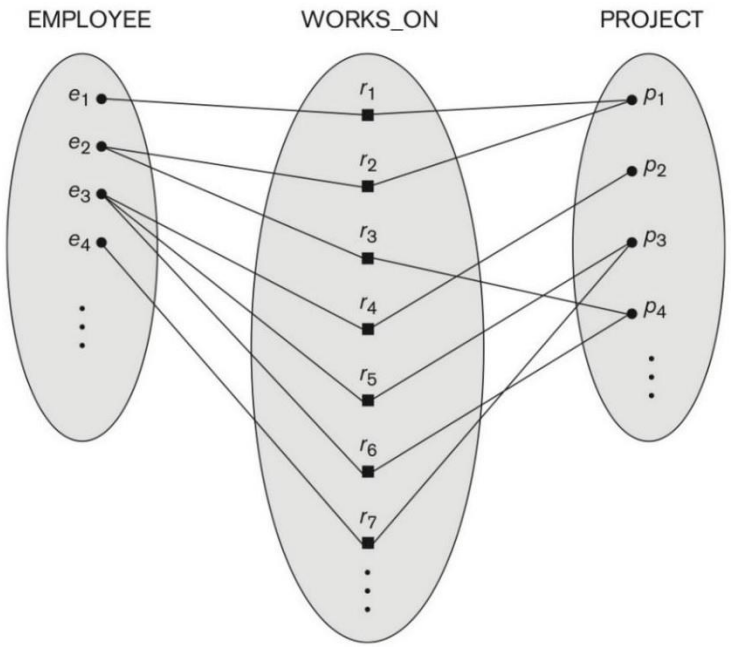
- **Constraints** limit the possible **combination** of entities that may participate in the corresponding relationship set.
  - Derived from **business rules** of the mini-world.
- **Relationships** must comply with two **types** of **constraints**:
  - **Cardinality ratio constraint**.
  - **Participation constraint** (*existence dependency*).
- **Cardinality ratio** & **participation constraint** together represent **structural constraints** of a relationship type.

CARDINALITY RATIO CONSTRAINT

- **Cardinality ratio**.
  - Specifies **maximum** number of relationship instances that an entity can **participate** in.
    - **One-to-one** (1:1).
    - **One-to-many** (1:N) or **many-to-one** (N:1).
    - **Many-to-many** (M:N).
  - Represented as **1**, **M**, or **N** on the diamonds in ER diagram.



MANAGES relationship



WOKRS_ON relationship

PARTICIPATION CONSTRAINT

- **Participation constraint** (*existence dependency*).
  - Specifies **minimum** number of relationship instances that an entity can participate in.
    - **Existence** of an entity depends on its being **related** to another entity.
  - **Participation** can be **total** or **partial**.
    - **Total** is **mandatory** participation.
      - Displayed as a **double line** connecting the participating entity type to the relationship in ER diagram.
    - **Partial** is **optional** participation.
      - Displayed as a **single line** connecting the participating entity type to the relationship in ER diagram.

ATTRIBUTES OF RELATIONSHIP TYPES

- Relationship types can have **attributes** just like entity types.
  - *Hours* attribute for WORKS_ON relationship type.
  - *StartDate* attribute for MANAGES relationship type.
- In 1:1 and 1:N relationship types the attribute can **migrate** to one of the participating entities.
  - In 1:1 to **either** of the entities.
  - In 1:N only to entity type on the **N-side**.
- In M:N some attributes may be determined by the **combination** of participating entities in a relationship instance.
  - Such attributes must be specified as **relationship attributes**.

WEAK ENTITY TYPES

- **Weak entity types** - entity types that **do not** have **key attribute** of their own.
  - Identified by being **related** to specific entity from another entity type and a **partial key**.
- **Identifying relationship relates** a weak entity type to its owner.
- Example:
  - DEPENDENT entity is identified by the **dependent's name**, and the **specific EMPLOYEE** to whom the dependent is related.
    - *Name* of DEPENDENT is the **partial key**.
    - EMPLOYEE is its **identifying entity** type via the **identifying relationship** type DEPENDENT_OF.
- Displayed as a **double-lined boxes** and **double-lined diamonds** in ER diagram.

REFINING ER DIAGRAM FOR COMPANY DATABASE (1)

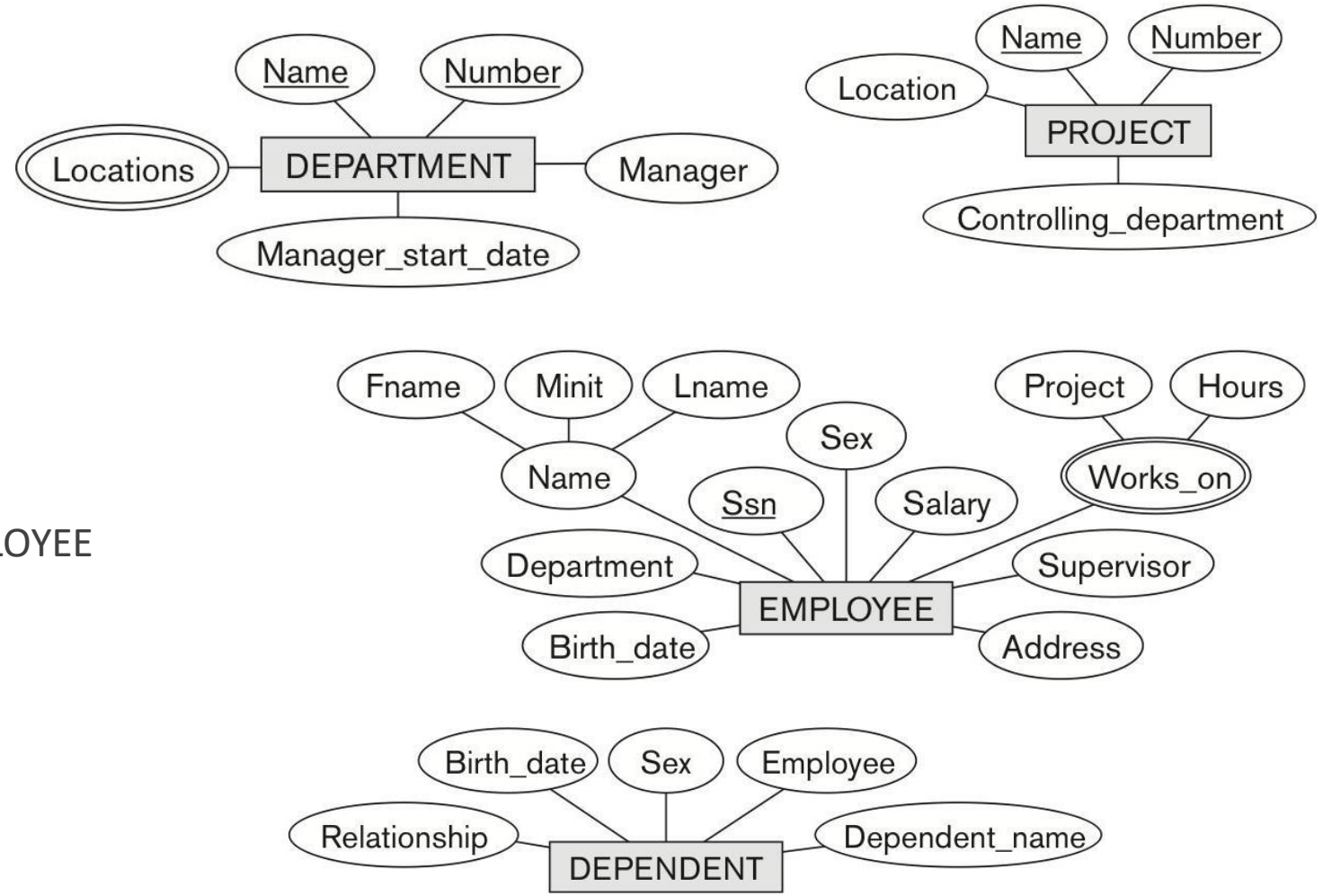
- Company database requirements:
  - The company is organized into departments. Each department has a unique name, a unique number, and a particular employee who manages the department. We keep track of the start date when that employee began managing the department. A department may have *several* locations.
  - 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 *one* 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 name, sex, birth date, and relationship to the employee.

REFINING ER DIAGRAM FOR COMPANY DATABASE (1)

- Company database 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 one *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 name, sex, birth date, and relationship to the employee.

REFINING ER DIAGRAM FOR COMPANY DATABASE (2)

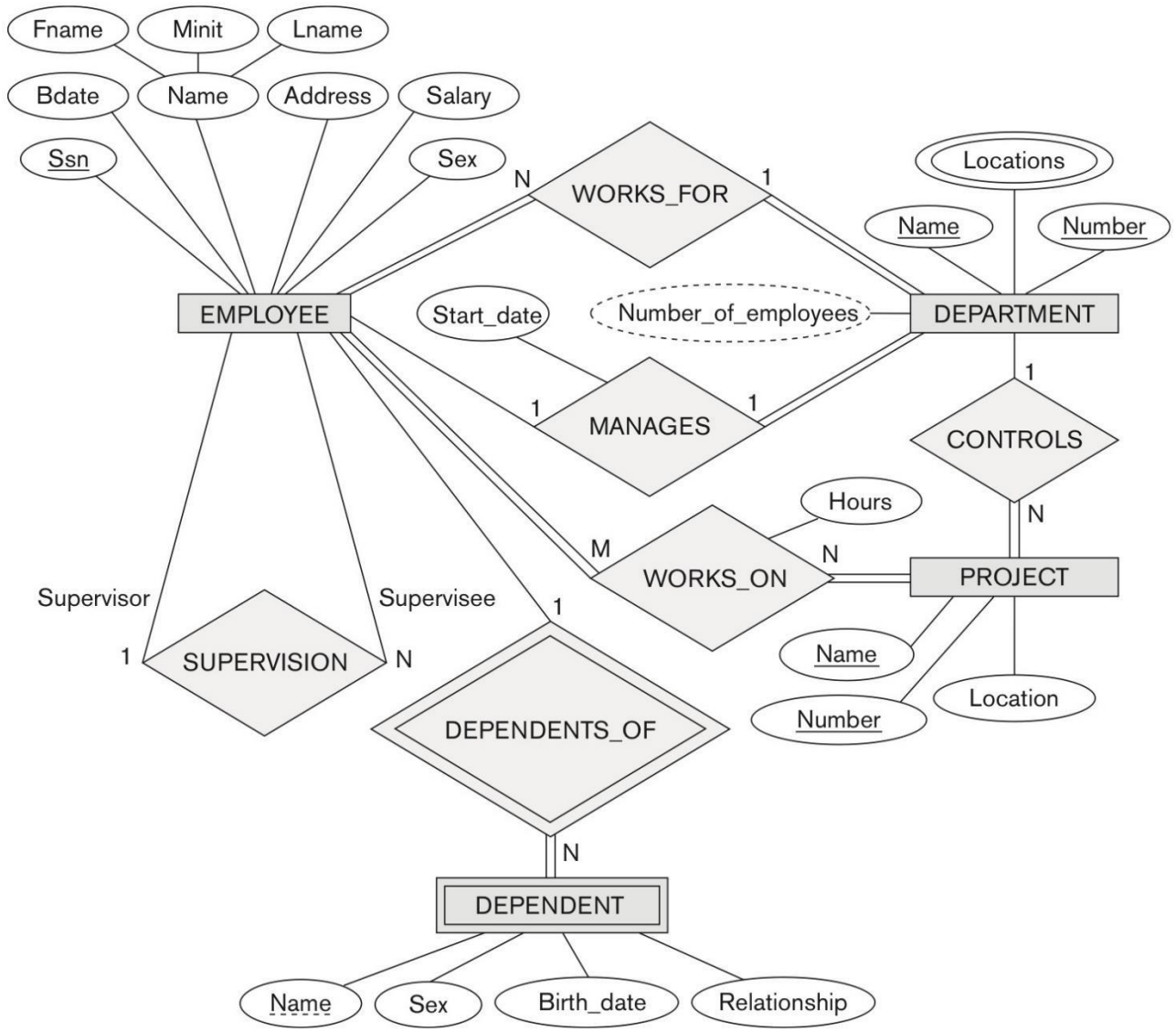
- By examining the requirements, **six binary relationship** types are identified:
  - MANAGES
    - 1:1 between EMPLOYEE and DEPARTMENT.
    - Relationship attribute - Start_date.
  - WORKS_FOR
    - 1:N between DEPARTMENT and EMPLOYEE.
  - CONTROLS
    - 1:N between DEPARTMENT and PROJECT.
  - SUPERVISION
    - 1:N between EMPLOYEE (supervisor) and EMPLOYEE (supervisee).
  - WORKS_ON
    - M:N between EMPLOYEE and PROJECT.
    - Relationship attribute – Hours.
  - DEPENDENTS_OF
    - 1:N between EMPLOYEE and DEPENDENT.
    - Identifying relationship for weak entity DEPENDENT.



Initial ER diagram for company database

REFINING ER DIAGRAM FOR COMPANY DATABASE (2)

- By examining the requirements, **six binary relationship** types are identified:
  - MANAGES
    - 1:1 between EMPLOYEE and DEPARTMENT.
    - Relationship attribute - Start_date.
  - WORKS_FOR
    - 1:N between DEPARTMENT and EMPLOYEE
  - CONTROLS
    - 1:N between DEPARTMENT and PROJECT
  - SUPERVISION
    - 1:N between EMPLOYEE (supervisor) and EMPLOYEE (supervisee).
  - WORKS_ON
    - M:N between EMPLOYEE and PROJECT.
    - Relationship attribute – Hours.
  - DEPENDENTS_OF
    - 1:N between EMPLOYEE and DEPENDENT.
    - Identifying relationship for weak entity DEPENDENT.



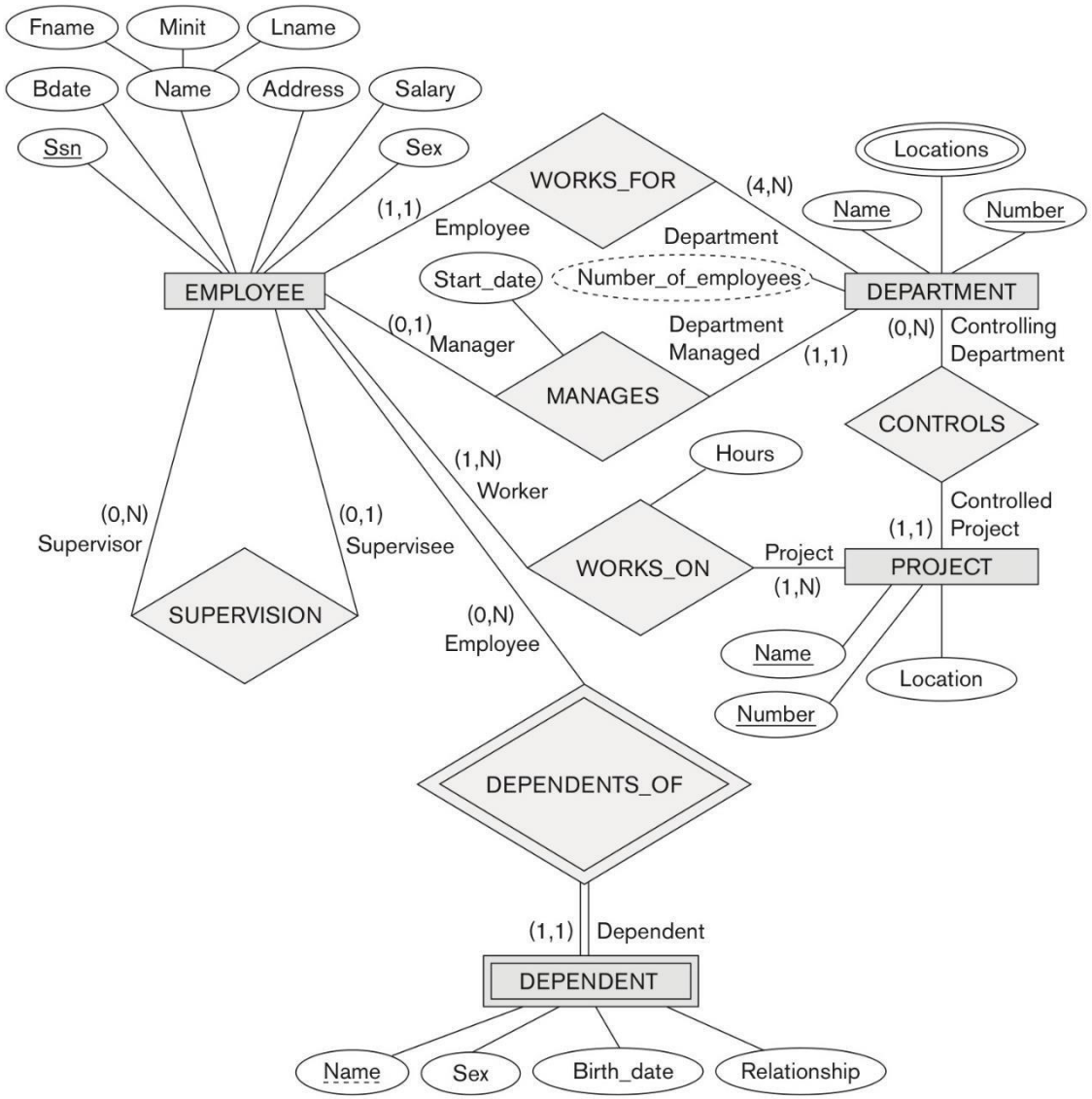
Refined ER Diagram For Company Database

ALTERNATIVE NOTATION OF RELATIONSHIP STRUCTURAL CONSTRAINTS (1)

- **Cardinality ratio** *(1:1, 1:N, M:N)* and single/double lines are replaced with *(min, max)* **participation constraints**.
- The numbers mean that each entity must participate in at least **min** and at most **max** relationship instances at any point in time.
  - min = 0, implies **partial** participation
  - min > 0, implies **total** participation
- Examples:
  - A *department* has **exactly one** *manager* and an *employee* can manage **at most one** *department*.
    - Specifies **(0,1)** for participation of EMPLOYEE in MANAGES.
    - Specifies **(1,1)** for participation of DEPARTMENT in MANAGES.

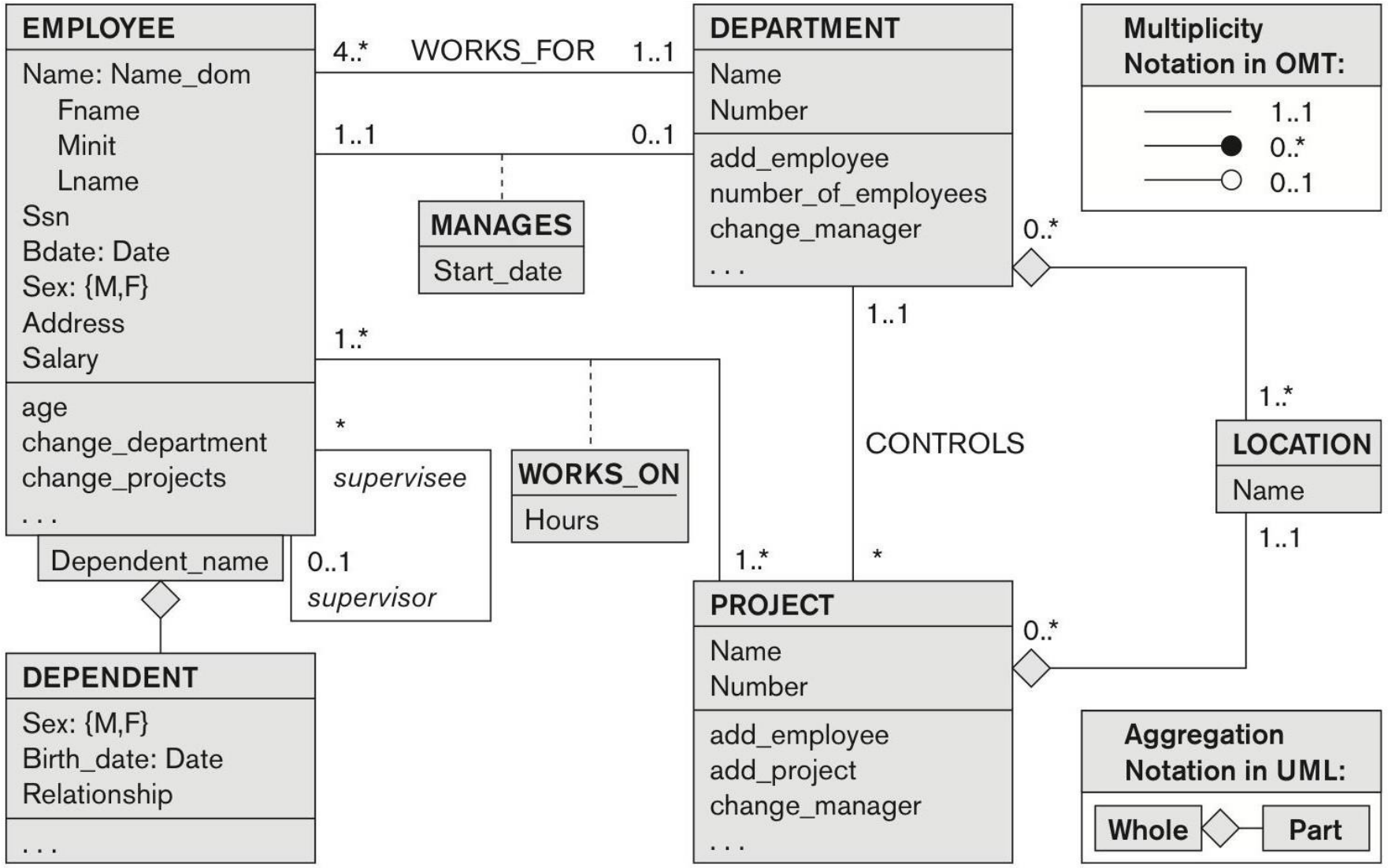
  - An *employee* can work for **exactly one** *department* but a *department* can have **any number** of *employees*.
    - Specifies (1,1) for participation of EMPLOYEE in WORKS_FOR.
    - Specifies (1,N) for participation of DEPARTMENT in WORKS_FOR.

ALTERNATIVE NOTATION OF RELATIONSHIP STRUCTURAL CONSTRAINTS (2)



ER diagram for company database with (min, max) notation and role names

ALTERNATIVE CONCEPTUAL SCHEMA DIAGRAM



COMPANY conceptual schema in UML class diagram notation

SUMMARY

- Overview of database system design process.
- ER model basic concepts.
  - Entity.
    - Entity types, entity sets.
  - Attribute.
    - Types of attributes, key attributes.
  - Relationship.
    - Relationship types.
    - Relationship degrees.
    - Recursive relationships & role names.
    - Relationship constraints.
    - Attributes of relationship types.
    - Weak entity types.