2. Entity Relationship (ER) Model

- **Conceptual Modeling** is a very important phase in designing a successful database application

- **Database Application = database + associated application programs**

e.g. bank database application

→ bank database + Programs that update bank database

- These application programs provide user-friendly graphical user interfaces (GUls). Hence, part of the database application will require the design, implementation, and testing of these application programs.

2. **Entity Relationship (ER) Model**

- Database design methodologies and software design methodologies are strongly related.

- As an example for Conceptual modeling (High-level data model) , we present **ER modeling concepts** , for specifying database structures and constraints during database design.

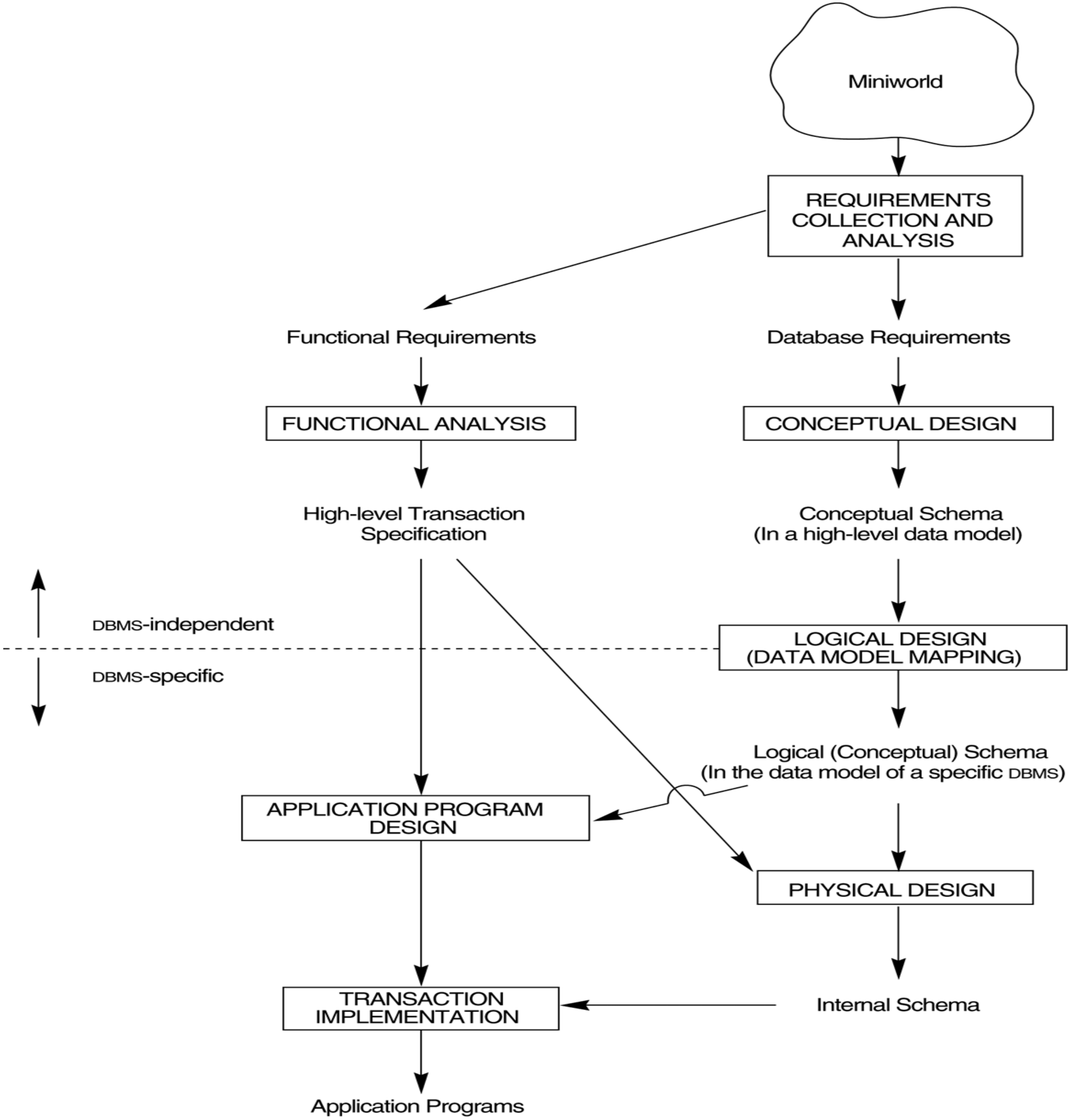
- This model and its variations are frequently used for the conceptual design of database applications, and many database design tools employ its concepts.

Topics to be discussed

1. Using High Level Conceptual Data models for database design
2. Entity types, entity sets ,Attributes and Keys
3. Relationship types, Relationship sets
4. Roles and structural constraints
5. Weak entity types
6. ER diagrams
7. An example of database application
8. Refining the ER design for company database
9. Naming conventions and design Issues.

1. **Using High Level Conceptual Data models for database design**

The main phases in Database design process



Miniworld
REQUIREMENTS COLLECTION AND ANALYSIS
Functional Requirements
Database 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

The main phases in Database design process :

Step - 1 : Requirement Collections and analysis

- Database designers interview perspective database users to understand and document their data requirements.
- These requirements are specified in as detailed and complete a form as possible.
- In parallel with this, functional requirements of the application are also specified .

The main phases in Database design process :

Step - 2 : Conceptual design

- Conceptual schema is a concise description of **data requirements** of the user and detailed description of the **entity types, relationships and constraints**
- These description are expressed using a high level data model.
- The model does not include any implementation details and so it is meant for non -technical users.
- The high-level user operations are specified using basic data model operations.

The main phases in Database design process :

Step - 3 : Logical design (data model mapping)

- In this step the database is actually implemented using a commercial DBMS.
- Most current commercial DBMSs use an implementation data model such as the relational or the object-relational database model
- So the conceptual schema is transformed from high level data model into the implementation data model.

The main phases in Database design process :

Step - 4 : Physical design

- **Physical design specifies :**
  - the internal storage structures,
  - indexes,
  - access paths
  - and file organization for the database files

- **In parallel, the application programs are designed and implemented as database transactions**

Basic Concepts of Entity Relationship (E-R) Model

Entity types, Entity Sets, Attributes and Keys

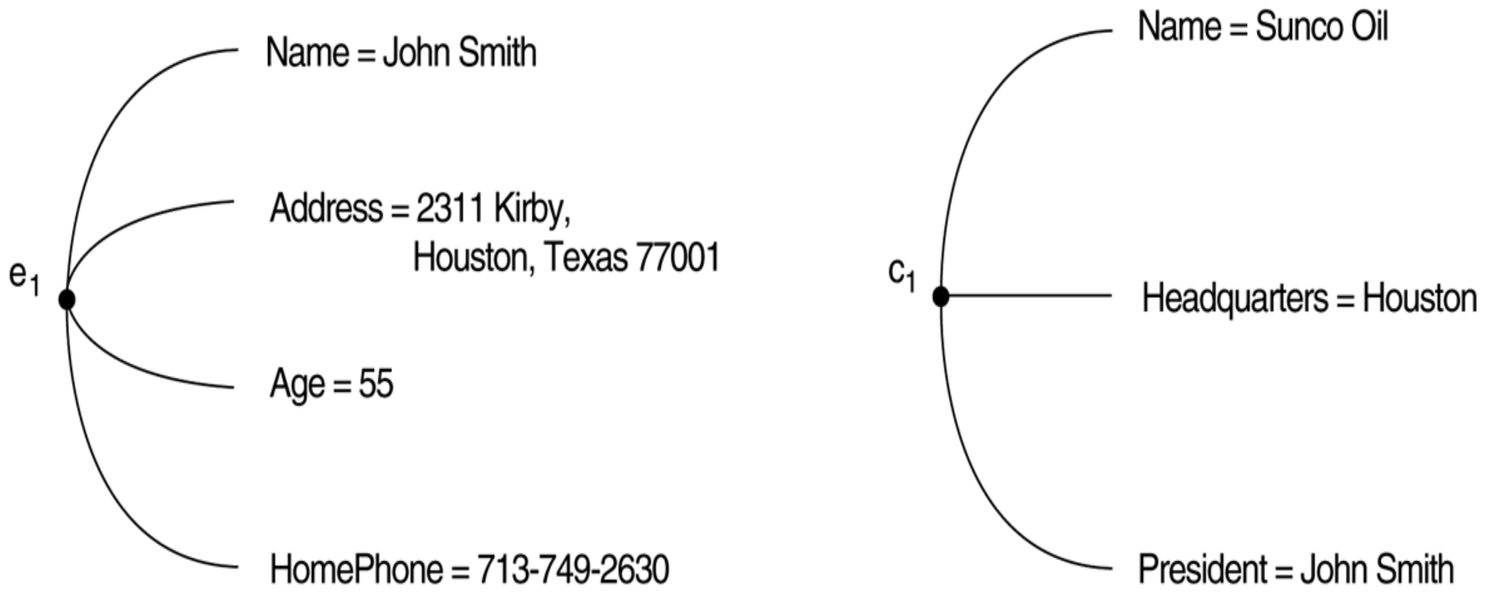
Entities and Attributes:

Entity : An entity may be an object with a physical /Conceptual existence.

E.g.: a Person, car , house , Course , Job , Company

Attributes : The particular properties that describe an entity .

E.g. : employee and Company entities and their attributes



Types of attributes.

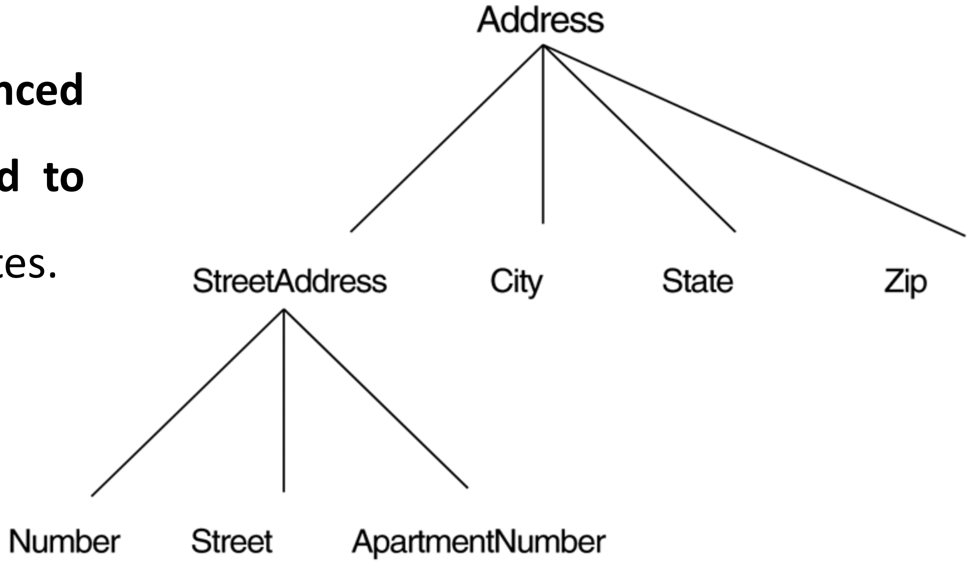
1. Simple (atomic) v/s Composite:

- A simple attribute can not be divided into smaller subparts
- Composite attributes can be divided into smaller parts

E.g. : Address → (StreetAddress, City, State, and Zip)

Composite attribute can form a hierarchy

If the composite attribute is referenced only as a whole, there is no need to subdivide it into component attributes.



Types of attributes

2. Single Valued v/s Multivalued attributes

Single Valued → has a single value for an enity

e.g. → age of a person

Multivalued → has multiple values

e.g. → Color of a car or College degree for a person.

3. Stored v/s Derived attributes

Stored attribute : BirthDate of a person

Derived attributes : Age can be derived from BirthDate

Types of attributes

3. Stored v/s Derived attributes

- An attribute can be derived from its related attribute

E.g. **Stored attribute :** BirthDate of a person

Derived attributes : Age can be derived from BirthDate

- Some attribute values can be derived from related entities

E.g.: **Entity :** CSE Department

Derived attribute : NoOfEmployees (in CSE department)

Types of attributes

4. Null values

- Null value is used when there is no applicable value(not applicable) for an attribute

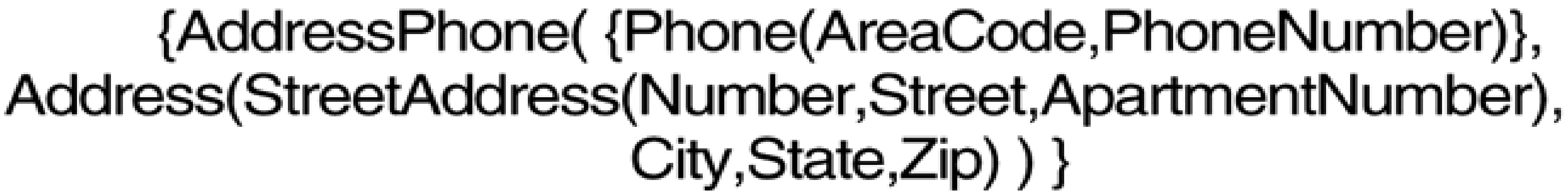
  e.g. : a person may not have college degree.

- Null values are also used when we do not know (unknown) the value of an attribute.
  - Unknown values can be
    - existing but missing (height of a person)
    - Not known whether it is existing (home phone number of some person)

Types of attributes.

5. Complex attributes :

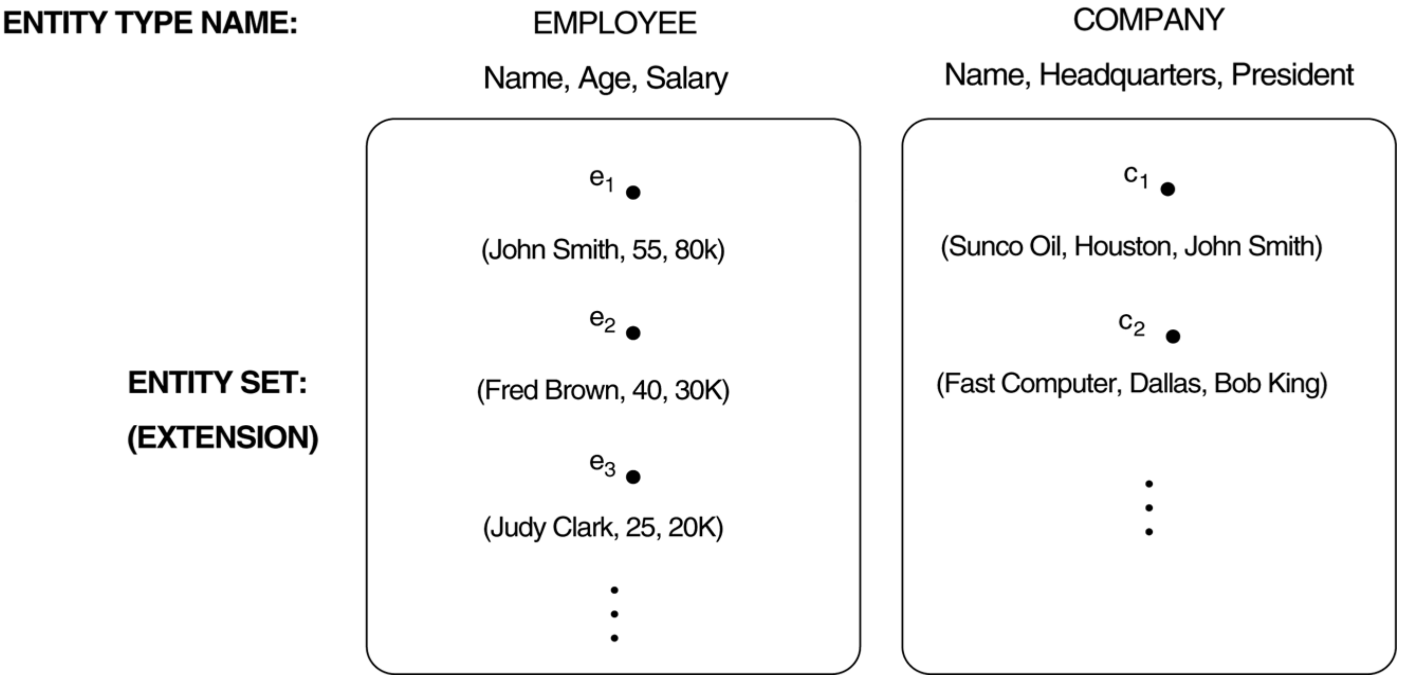
- Composite and multi valued attributes can be nested in an arbitrary way to obtain a complex attribute .
- Composite attributes are grouped between parentheses () and separating them with coma.
- Multivalued attributes are displayed between braces { }



{AddressPhone({Phone(AreaCode,PhoneNumber)},
Address(StreetAddress(Number,Street,ApartmentNumber),
City,State,Zip)) }

Entity types, Entity Set

- **Entity type :** A Collection of entities that have the same attributes. Each entity type is described by Name and attributes.
- **Entity Set :** Collection of all entities of a particular entity type in the database at any point in time.



Entity types, Entity Set

- **An entity type is represented in ER diagrams**
- **Notations in ER diagrams**
  - An entity type is represented by a rectangular box enclosing the entity type name.
  - Attribute names are enclosed in ovals and are attached to their entity type by straight lines.
  - Composite attributes are attached to their component attribute by straight lines.
  - Multi valued attributed are displayed in double ovals.

Key attributes

- An attribute whose values are distinct for each individual entity in the entity set is called a **key attribute**

  Example : Register Numbers of students.

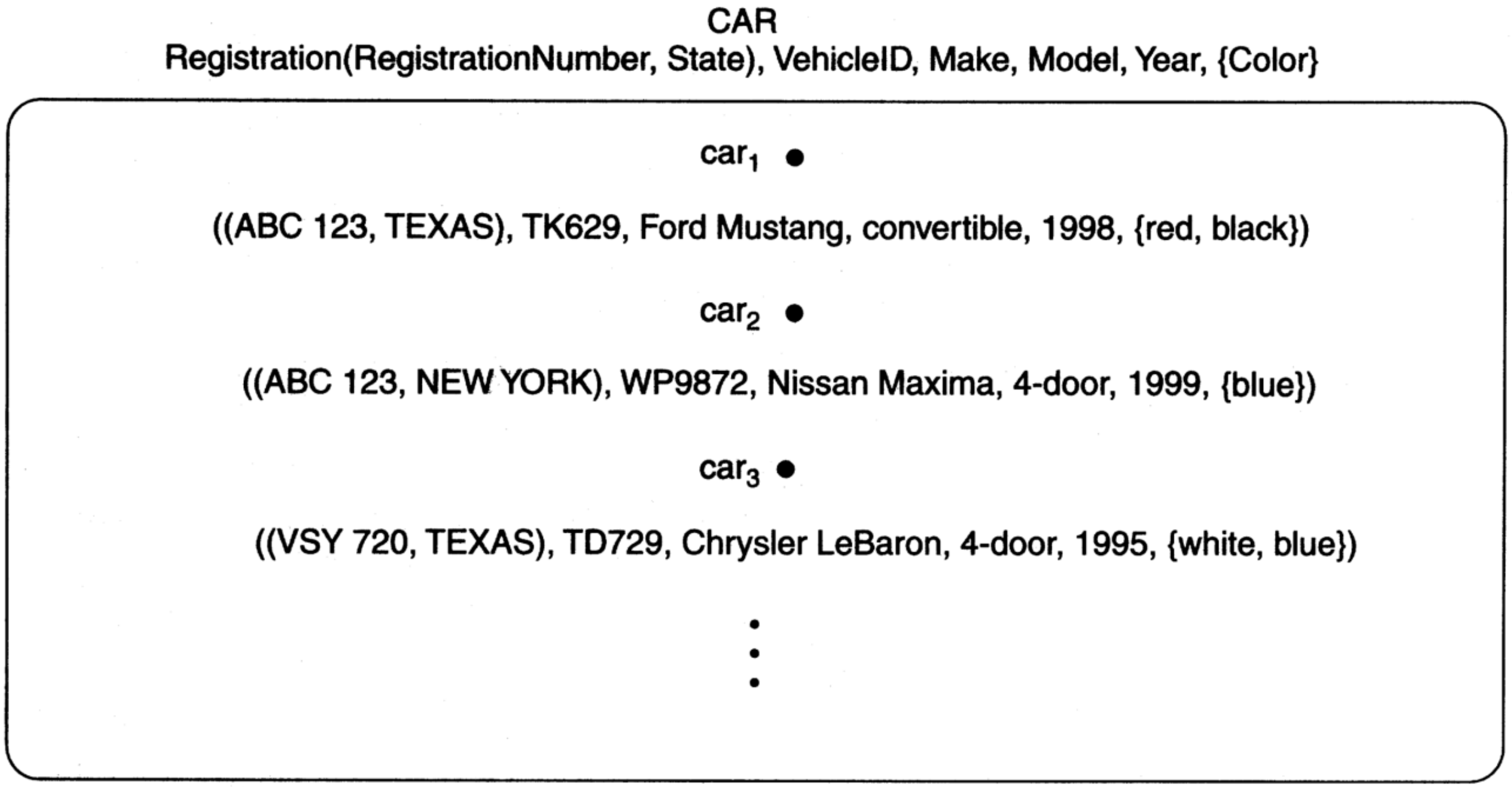
- A key attribute can be used to identify each entity uniquely. In ER diagram, each key attribute has its name underlined inside the oval

- Sometimes, several attributes together form a key. Such attributes are represented using a composite attribute in ER - diagram. **Example :** (RegistrationNumber, State) **of a car**

Key attributes

- Some entity types have more than one key attributes.

  Example : Registration and VehicleId of a car



CAR

Registration(RegistrationNumber, State), VehicleID, Make, Model, Year, {Color}

car_1 ●

((ABC 123, TEXAS), TK629, Ford Mustang, convertible, 1998, {red, black})

car_2 ●

((ABC 123, NEW YORK), WP9872, Nissan Maxima, 4-door, 1999, {blue})

car_3 ●

((VSY 720, TEXAS), TD729, Chrysler LeBaron, 4-door, 1995, {white, blue})

⋮

Value Sets (Domains) of attributes :

- Specifies the set of values that may be assigned to that attribute for each individual entity.

  Example : Age of an employee is in between 18 and 50.

- **Value sets are not represented in ER diagram.**

- Mathematically , an attribute A of entity type E whose value set is V can defined as a function from E to the power set P(V) of V.

  i .e $A : E \rightarrow P(V)$.

Relationship types, Relationship sets , roles and Structural constraints

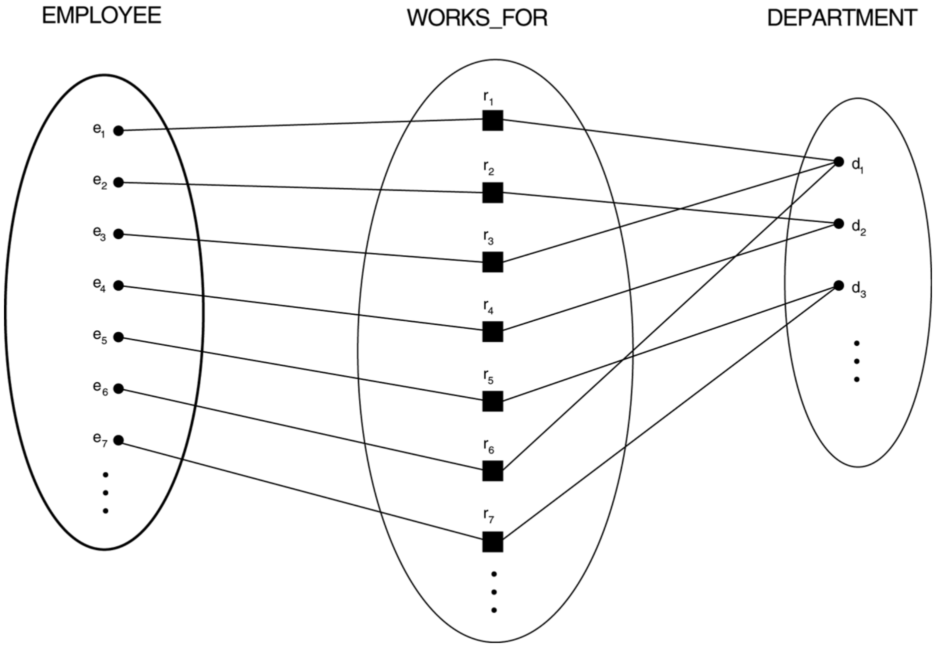
Relationship types

- A relationship type R among n entity types E_1, E_2, E_3.. E_n defines a set of associations among entities from these entity types.

E.g. ; An Employee Works For a Department

Entity Types → Employee and Department

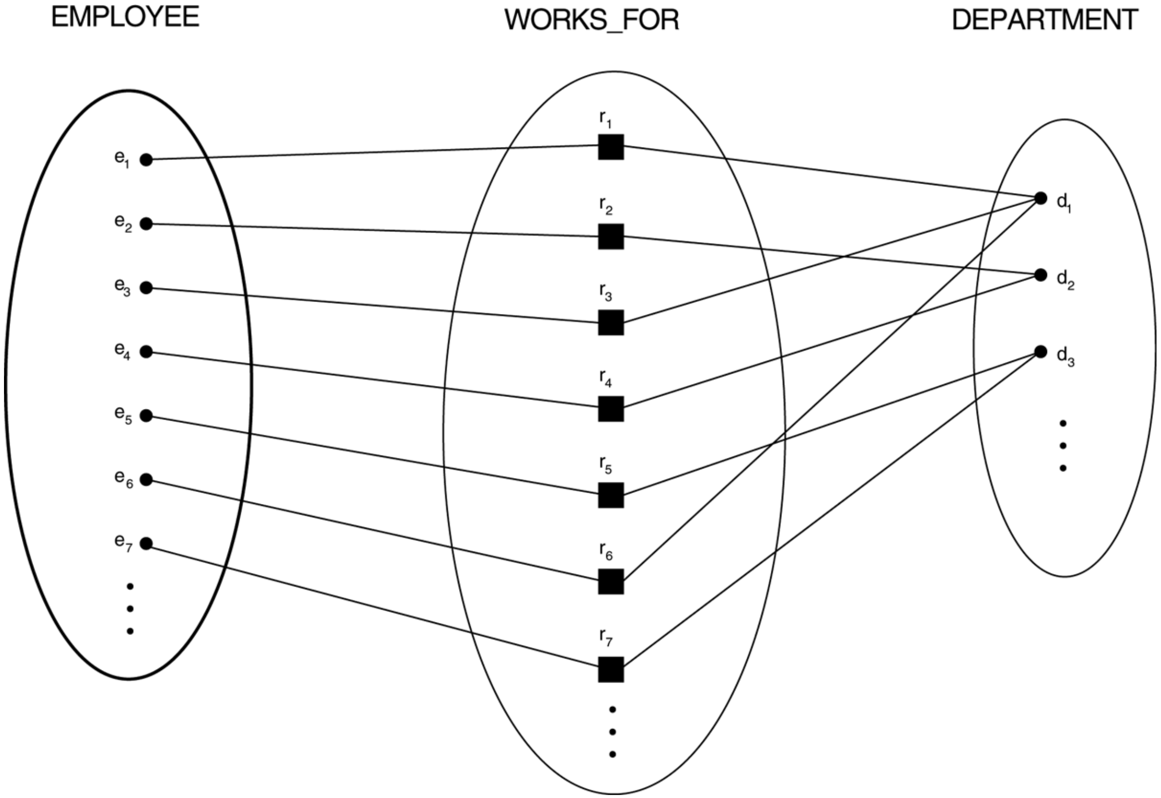
Relationship → Works_For



In ER – diagram , a relationship type represented by a diamond shaped box, which are connected by straight lines to the rectangular boxes representing the participating entity type.

Relationship sets

The relationship set R is a set of relationships instances r_i , where each r_i associates n individual entities $(e_1,e_2…e_n)$ and each entity e_j in r_i is a member of entity type E_j, $1<=j<=n$.



Relationship sets

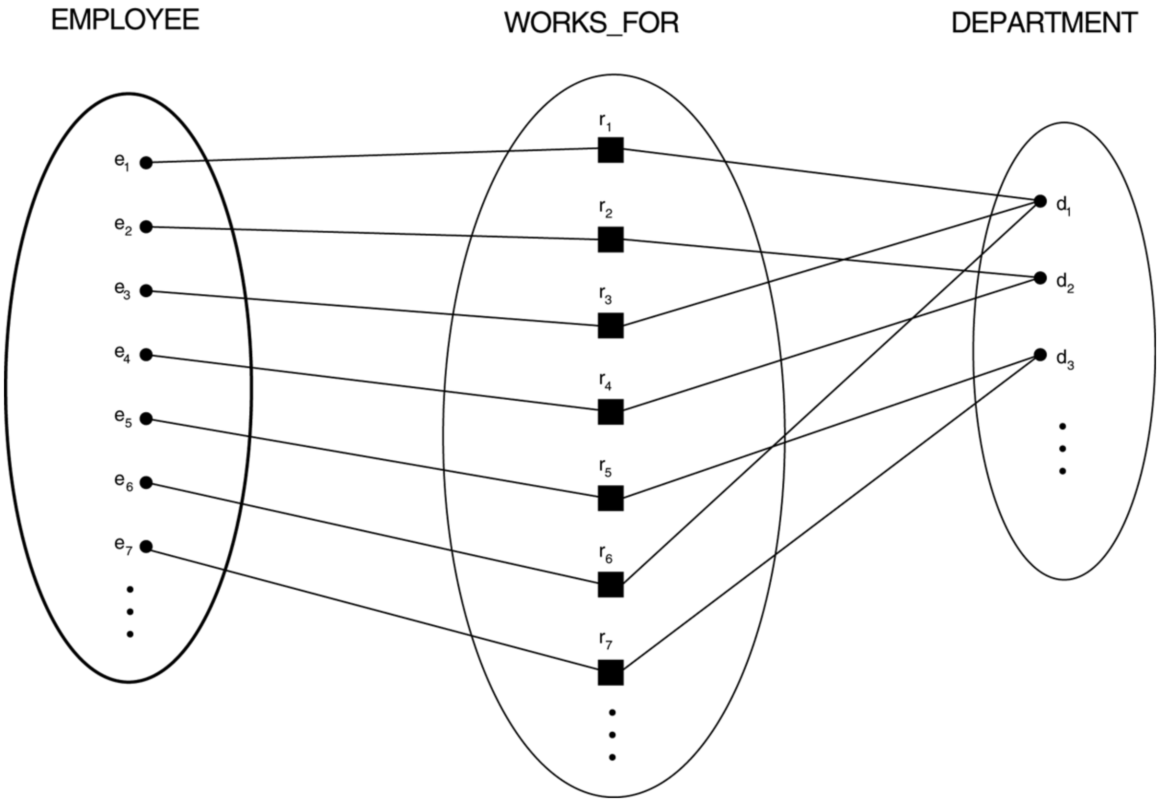
- Informally, each relationship instance r_i in R is an association of entities, where the association includes exactly one entity from each participating entity type.

for e.g. Each relationship instance in the relationship set WORKS_FOR associates one employee entity and one department entity.

Relationship Degree :

Number of participating entity types in a relationship.

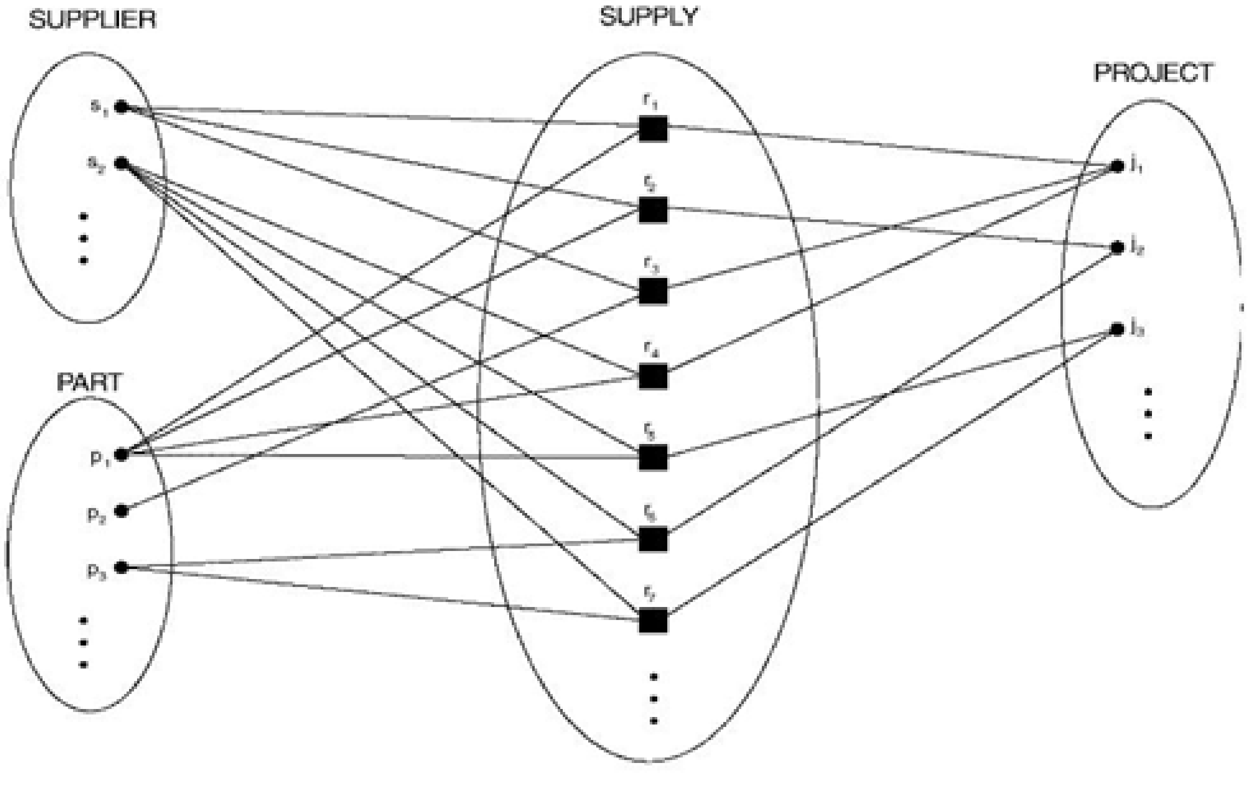
An example for binary relationship



Relationship Degree :

An example for ternary relationship

Each relationship instance r_i associates three entities-a supplier s, a part *p, and a project j whenever* s supplies part p *to project j.*



Role names :

It signifies the role played by each participating entity type and helps to explain what the relationship means.

Example : Employee **works for** Department

The relationship type **works_for** signifies that

- employee plays the role of an en *employee* or *worker*
- and Department plays the role of *department* or *employer*

- Role names are not technically necessary in relationship types where all the participating entity types are distinct,

Recursive Relationships :

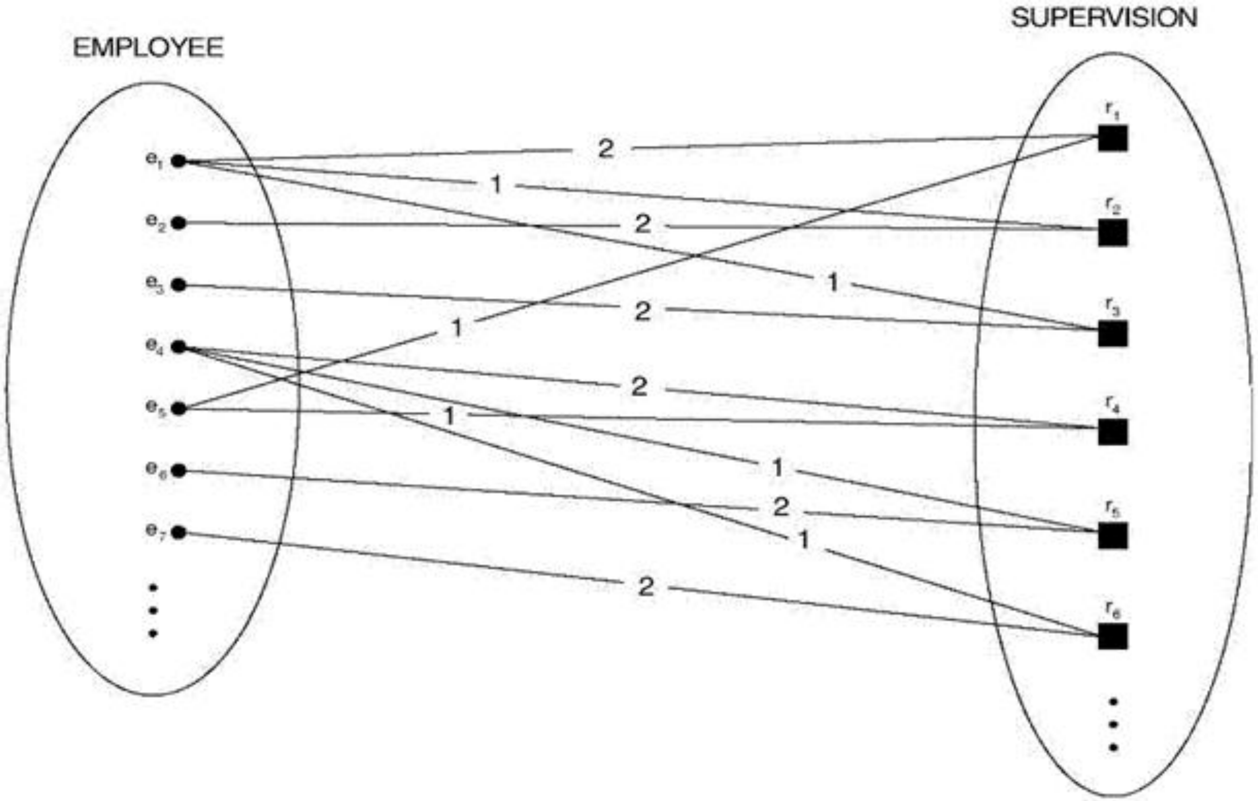
However ,In some cases the same entity type participates more than once in a relationship in different role.

- In such case role names are essential for distinguishing the meaning of each .
- Such relationship is called ***Recursive Relationships.***

Recursive Relationships : An Example

An Employee **Supervises** another Employee

- lines marked “1" represent the supervisor role,
- lines marked "2“ represent the supervisee role;



2.4 Relationship types, Relationship sets , roles and Structural constraints

Structural constraints:

Relationship Types have certain constraint that limit the possible combination of entities that may participate in the corresponding relationship set.

e.g. : If company has a rule that Each employee must work for exactly one department

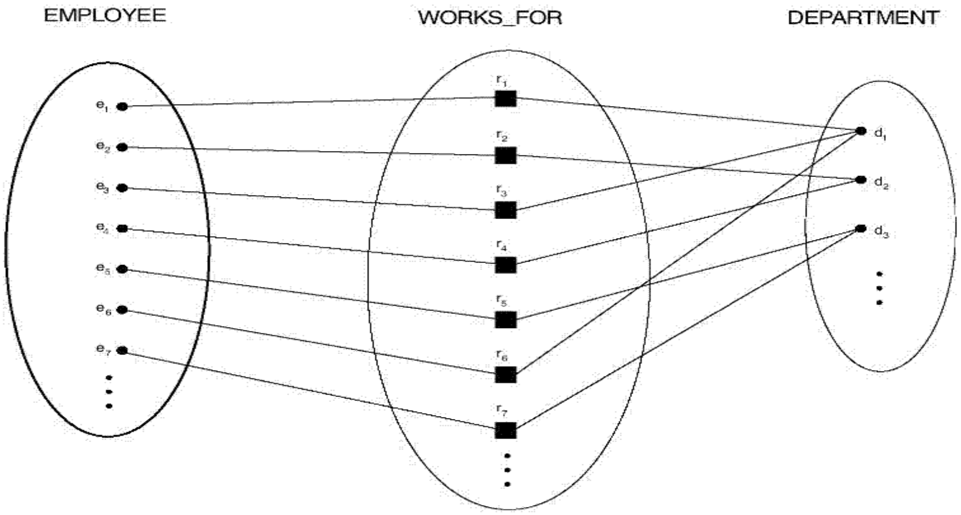


Figure 03.09
Some instances of the WORKS_FOR relationship between EMPLOYEE and DEPARTMENT.

2.4 Relationship types, Relationship sets , roles and Structural constraints

Two types of Structural constraints:

1. **Cardinality Ratio for binary relationship:**
2. **Participation Constraint :**

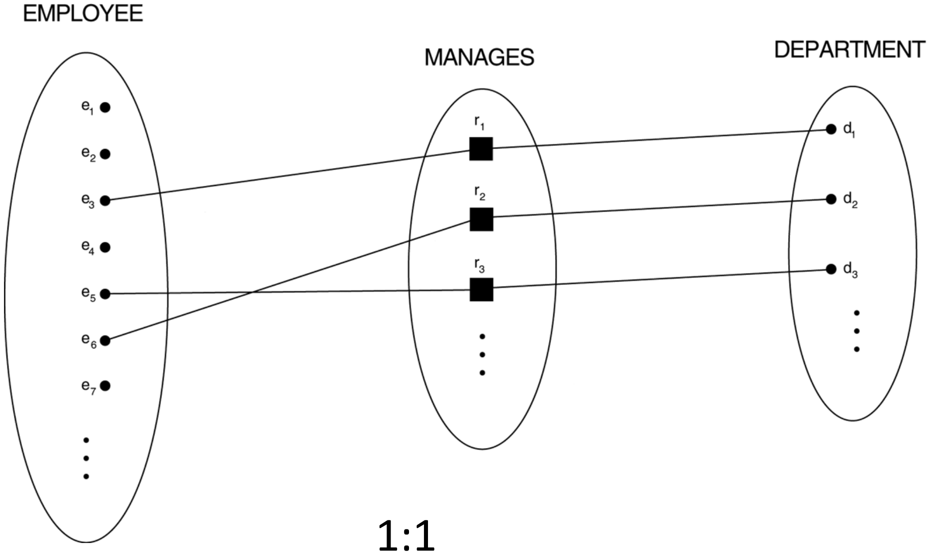
Cardinality Ratio for binary relationship:

- Specifies the maximum number of relationship instances that an entity can participate in.
- The possible ratios for binary ratios for binary relationship types are :

1:1, 1 : N, N:1, M:N

1:1

Employee Manages Department

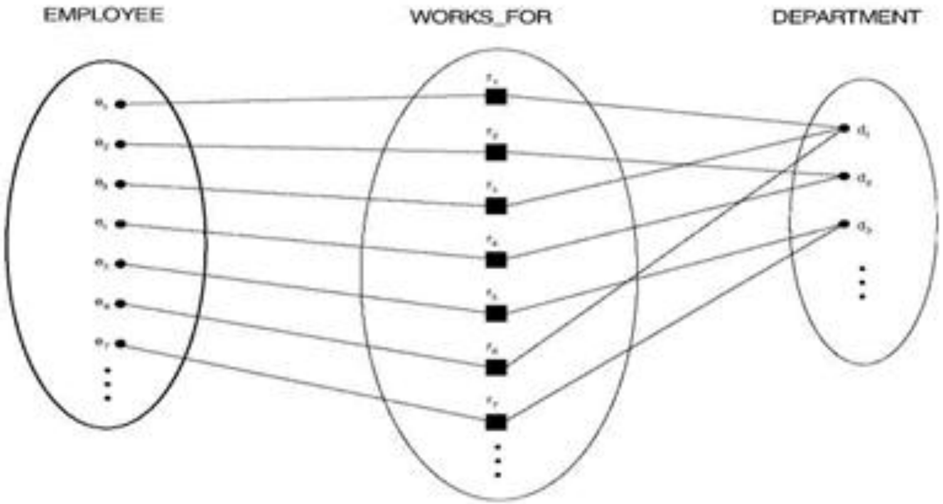


1:1

2.4 Relationship types, Relationship sets , roles and Structural constraints

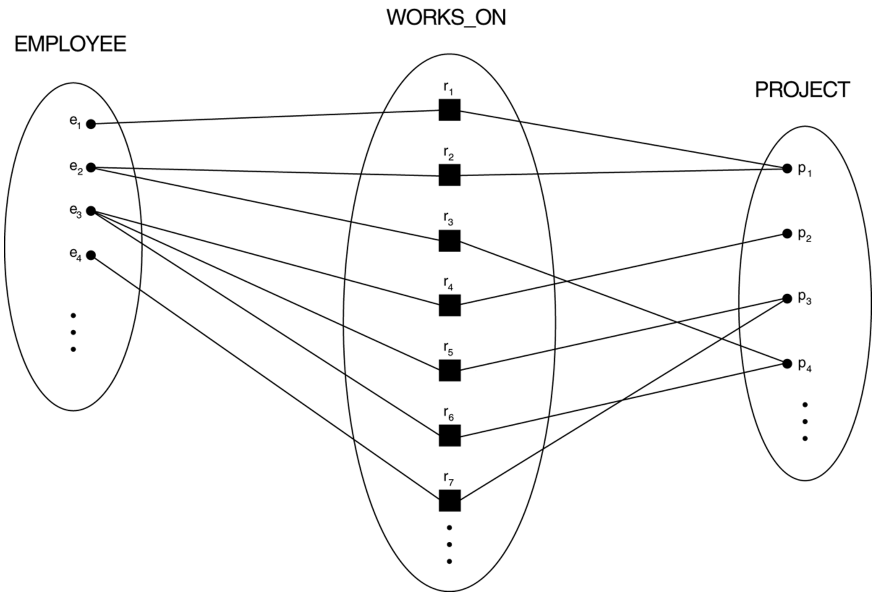
1:N

Employee works for department



M:N

Employee works on Project



2.4 Relationship types, Relationship sets , roles and Structural constraints

Participation Constraint : It specifies whether the existence of an entity depends on its being related to another entity via relationship type. It is also called minimal cardinality constraint.

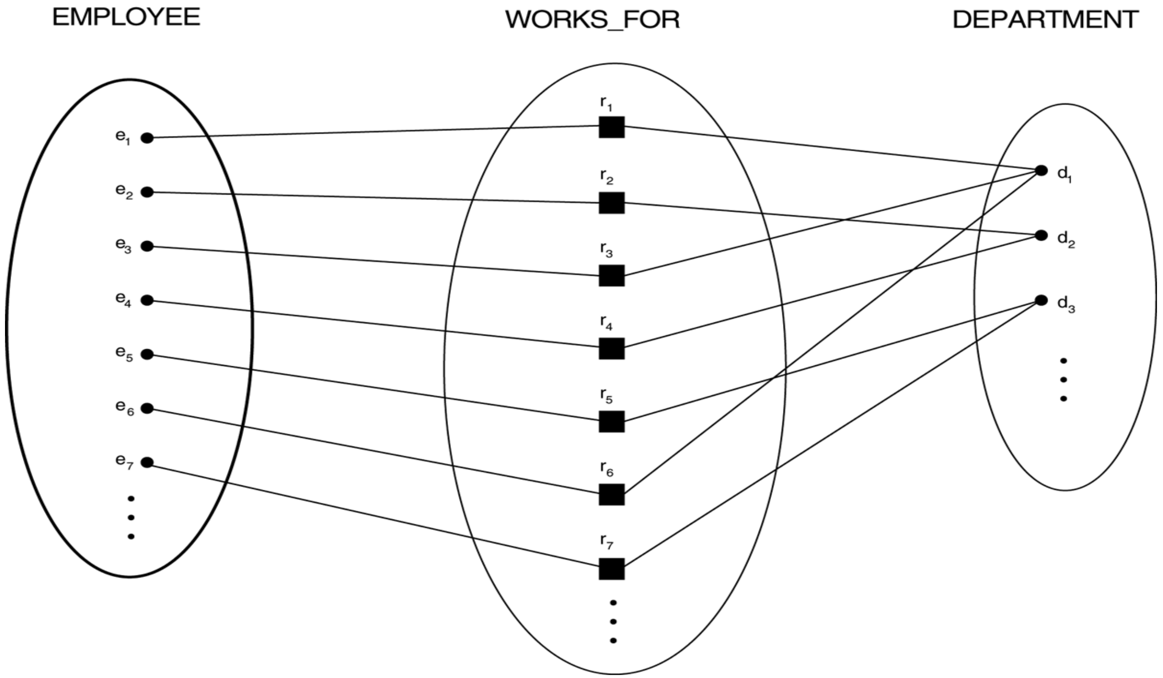
There are two types of participation constraints

1. **Total Participation** (every entity of an entity type must participate in at least one relationship instance)
2. **Partial participation** (every entity of an entity type need not participate a relationship instance)

2.4 Relationship types, Relationship sets , roles and Structural constraints

Example for Total Participation:

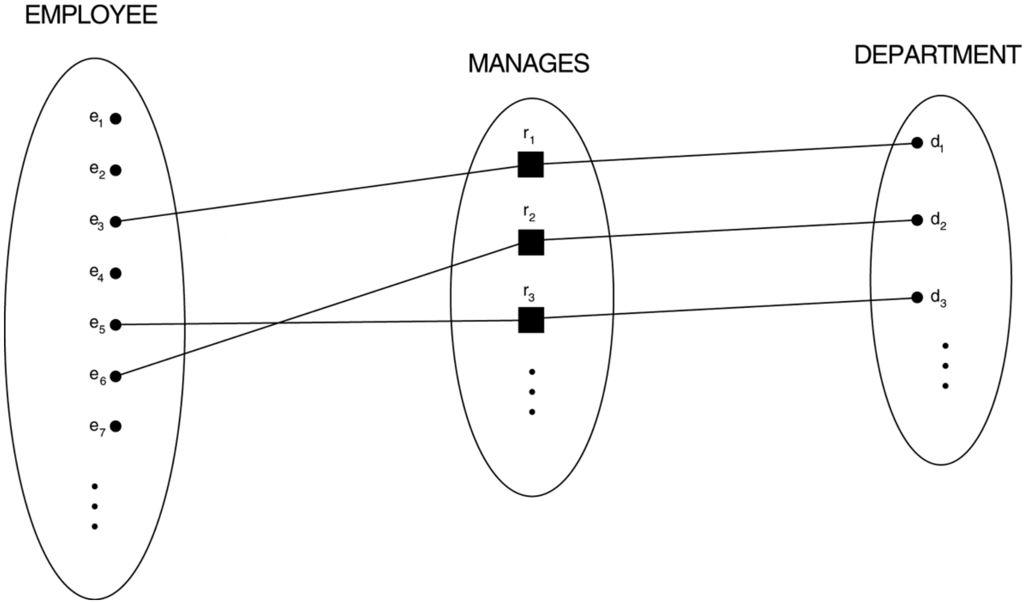
If a company rule states that every employee must work for a department , then an employee can exist only if participates in at least in one **works_for** relationship instance. Therefore this is also called **existence dependency**



2.4 Relationship types, Relationship sets , roles and Structural constraints

Example for Partial Participation:

We do not expect every employee to manage a department , So the participation of EMPLOYEE in MANAGES relationship type is partial.



In ER diagram Total participation is displayed as double line connecting the participating entity type to the relationship , whereas partial participation is displayed by a single line.

Weak Entity Types

- **Entity Types that do not have key attributes of their own are called weak entity types .**

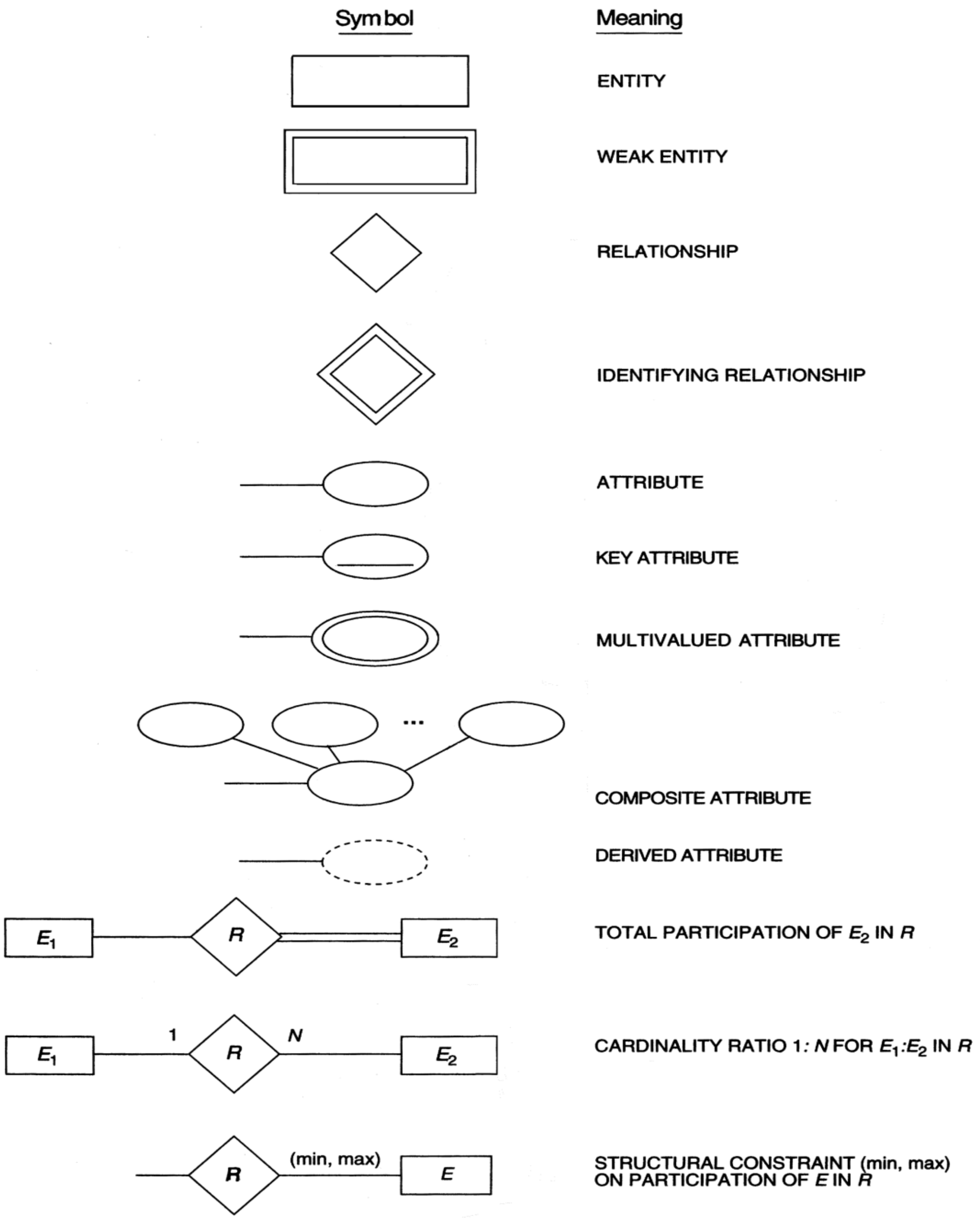
- Their existence depends on the existence of other entity type called **identifying or owner entity type** of the weak entity type.

- we call the relationship type that relates a weak entity type to its owner the **identifying relationship.**

Weak Entity Types

- A weak entity type always has a **total participation .**
- A weak entity type normally has a **partial key.**
- *Both weak entity type and owner entity type is displayed by surrounding their boxes and diamonds with double lines.*
- *Partial key is underlined with a dashed or dotted line.*

Notations in ER-diagrams



Symbol	Meaning
	ENTITY
	WEAK ENTITY
	RELATIONSHIP
	IDENTIFYING 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

2.2 An Example DATABASE Application

The database designer has provided the following Description of a company .

1. 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 that start date when that employee began managing the department. A department may have several location.

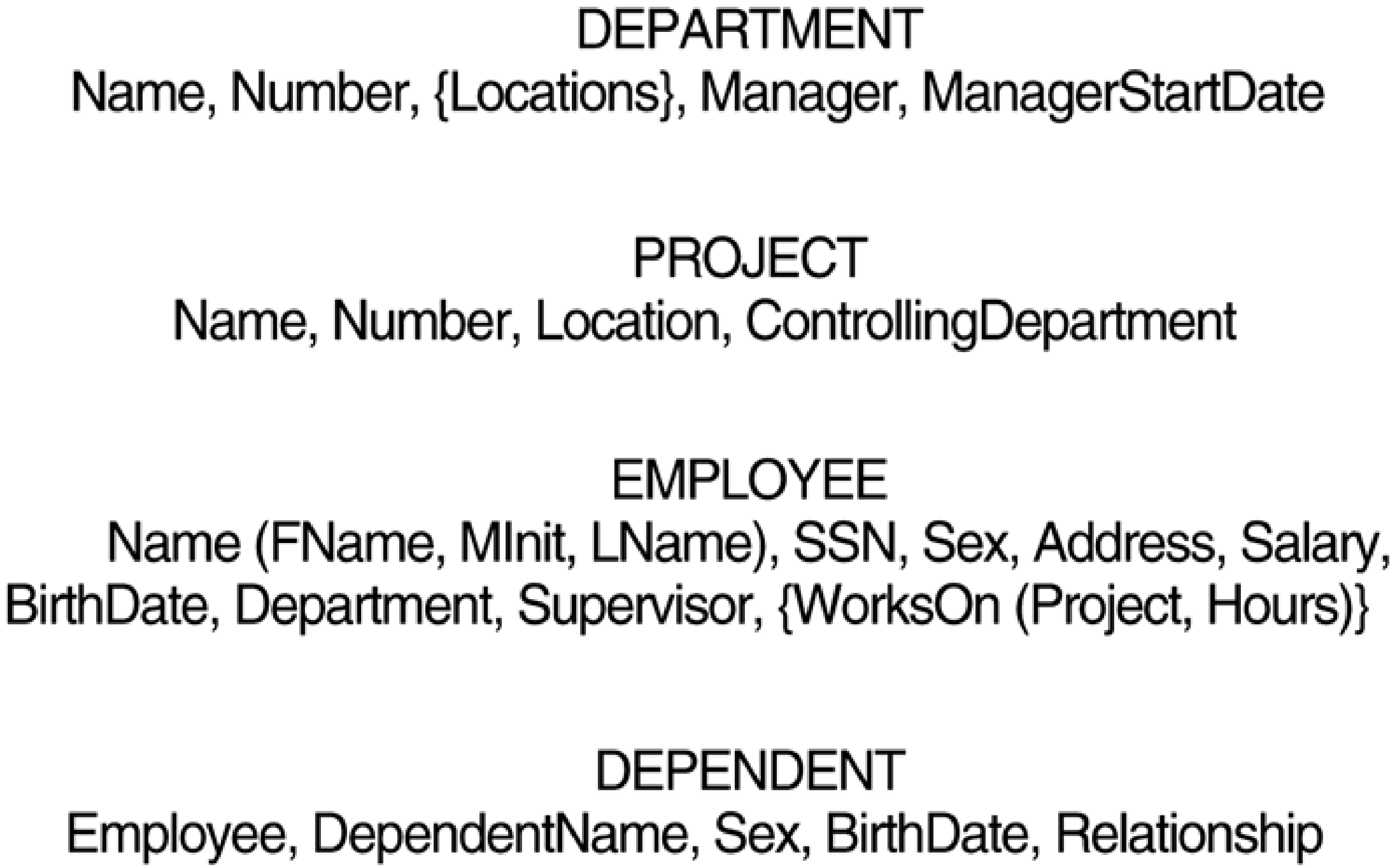
2 . A department controls a number of projects , each of which has a unique name , a unique number and a single location.

3 . We store each employee’s name, social security number, address, salary, sex and birthdates. An employee is assigned to one department but may work on several projects , which are not necessarily controlled by the same department. We keep track of the number of hours per week that an employee works on each projects. We also keep track of the direct supervisor of each employee .

4. We want to keep track of the dependents of each employee for insurance purposes. We keep each dependent’s first name , sex , birthdates and relation ship to the employee.

2.2 An Example DATABASE Application

Initial Conceptual Design



DEPARTMENT

Name, Number, {Locations}, Manager, ManagerStartDate

PROJECT

Name, Number, Location, ControllingDepartment

EMPLOYEE

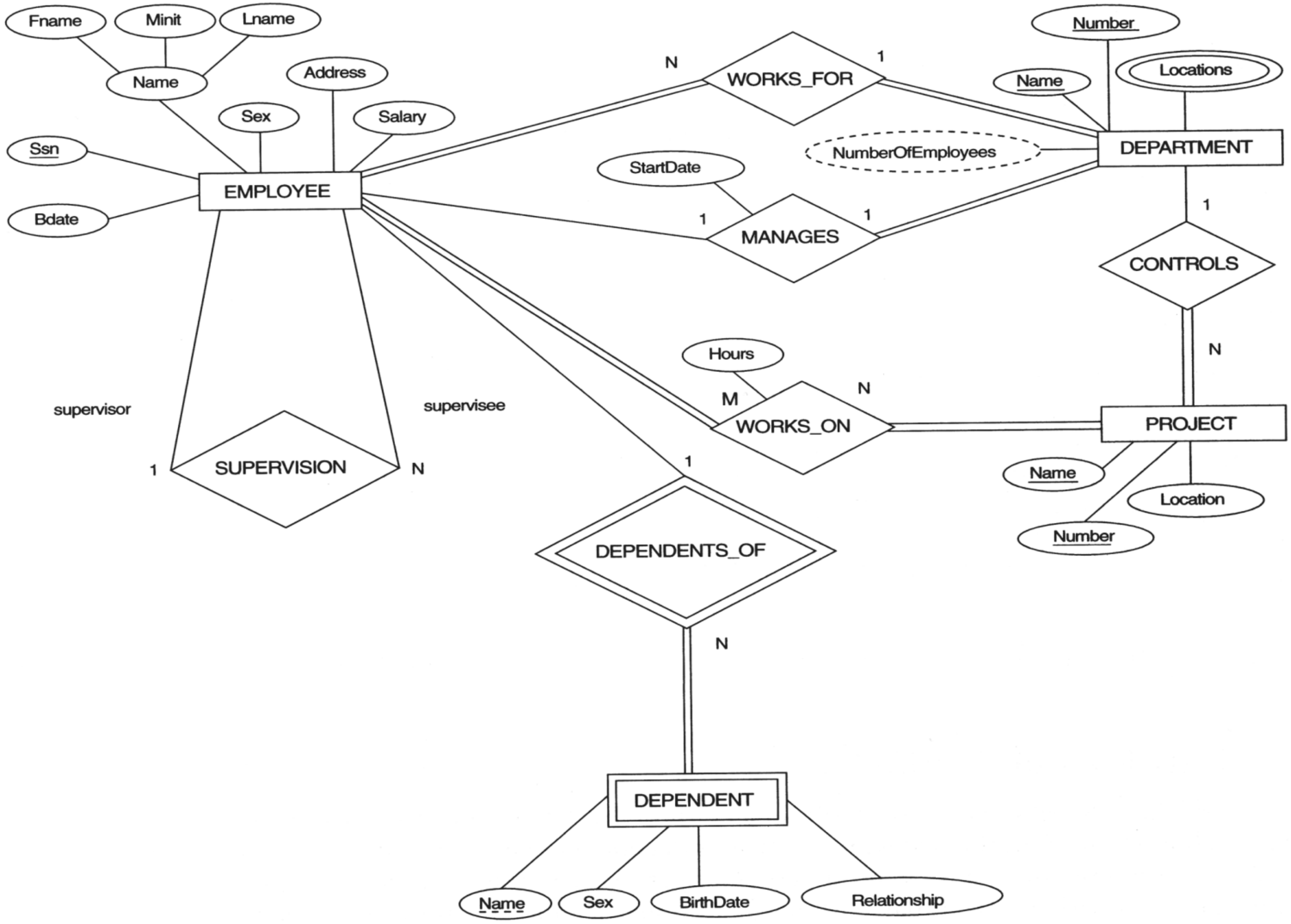
Name (FName, MInit, LName), SSN, Sex, Address, Salary, BirthDate, Department, Supervisor, {WorksOn (Project, Hours)}

DEPENDENT

Employee, DependentName, Sex, BirthDate, Relationship

2.2 An Example DATABASE Application

An ER schema diagram for the COMPANY database



ER - DIAGRAM

NAMING CONVENTIONS

AND DESIGN ISSUES

Naming Convention in ER diagram

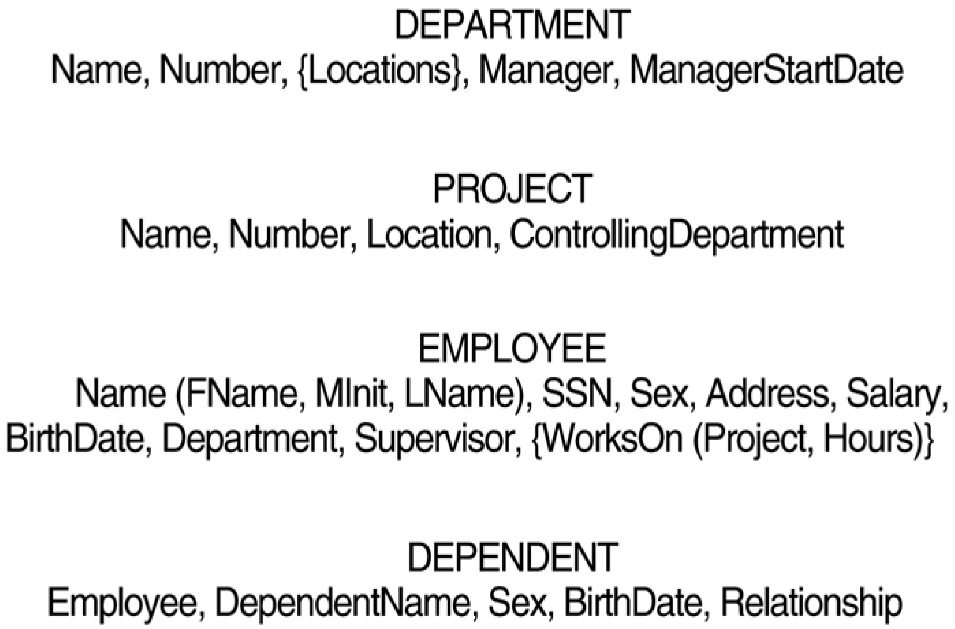
1. Choose names that convey , as much as possible, the meaning attached to the different construct in the schema.

2. Choose singular names for entry types, rather than plural ones.
3. Use uppercase letters for entity types and relationship type names.
4. Attribute names are capitalized. Role names are in lower case letters.
5. Choose binary relationship names to make ER diagram of the schema readable from left to right or top to bottom.

Design Choices for ER

A schema design process is an iterative refinement process. Some of the refinements include the following:

- A concept may be first modeled as an attribute and then refined into relationship because it is determined that the attribute is a reference to another entity type.



DEPARTMENT
Name, Number, {Locations}, Manager, ManagerStartDate

PROJECT
Name, Number, Location, ControllingDepartment

EMPLOYEE
Name (FName, MInit, LName), SSN, Sex, Address, Salary,
BirthDate, Department, Supervisor, {WorksOn (Project, Hours)}

DEPENDENT
Employee, DependentName, Sex, BirthDate, Relationship

Initial conceptual design

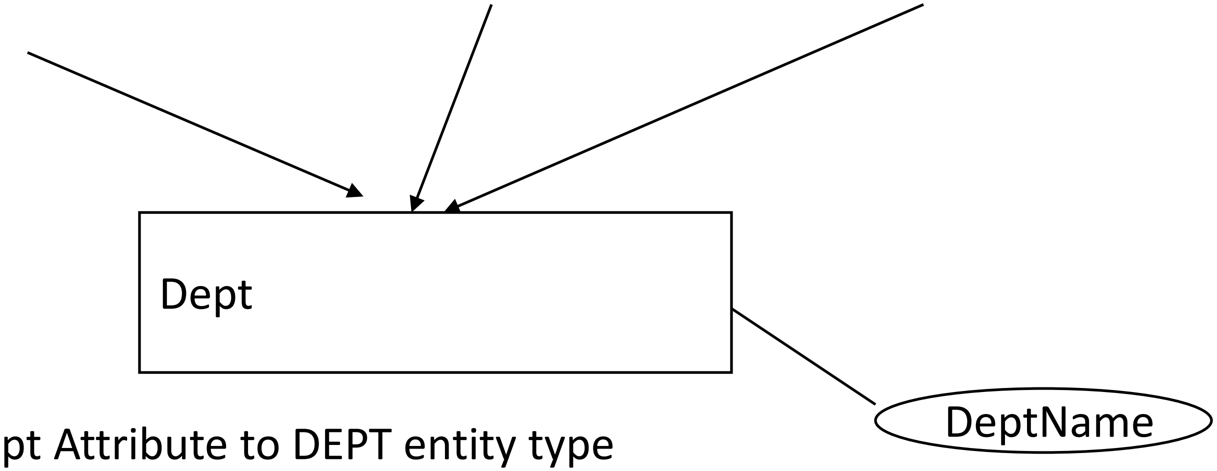
Design Choices for ER

- An attribute that exists in several entity type may be elevated or promoted to an independent a entity type.

Eg :

STUDENT, INSTRUCTOR , COURCE

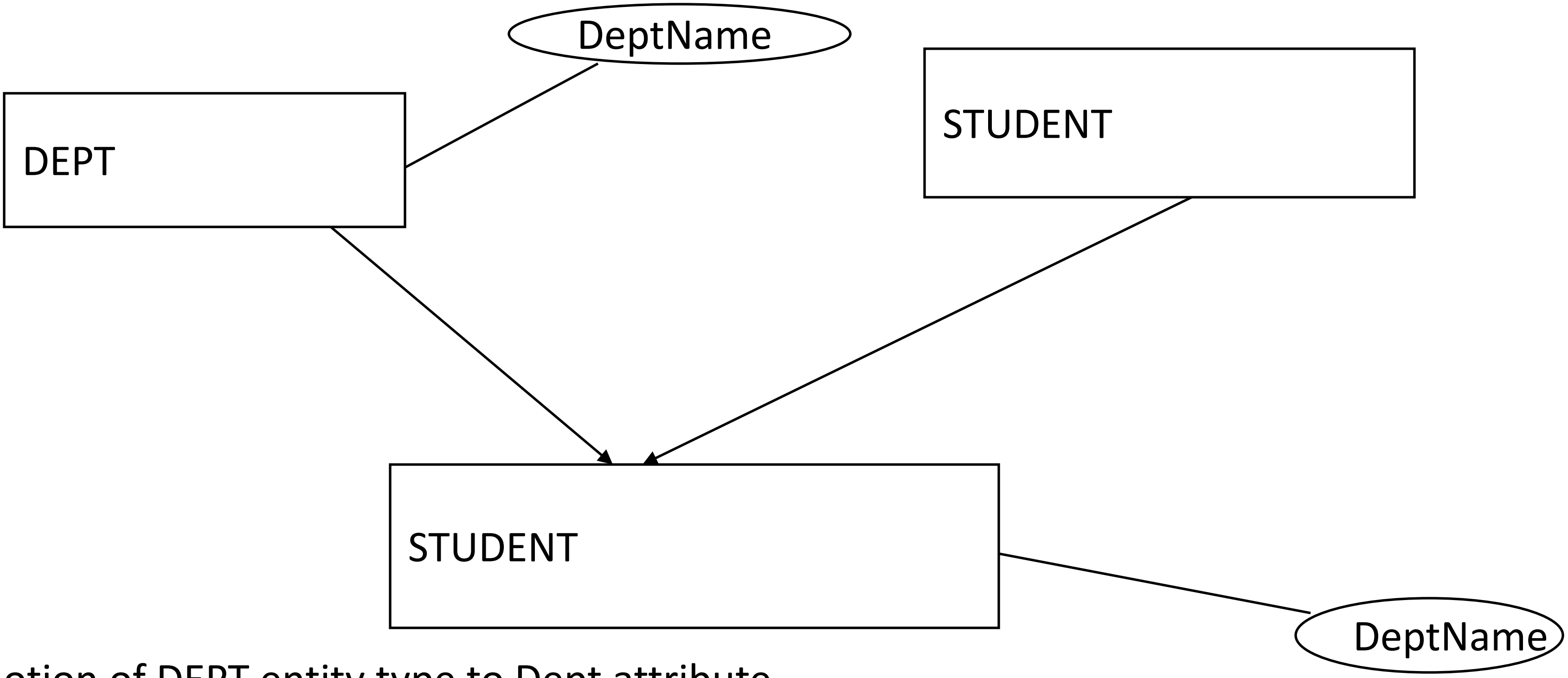
(…Dept) (…Dept) (..Dept)



Promotion of Dept Attribute to DEPT entity type

Design Choices for ER

- A inverse refinement to the previous case may be applied.



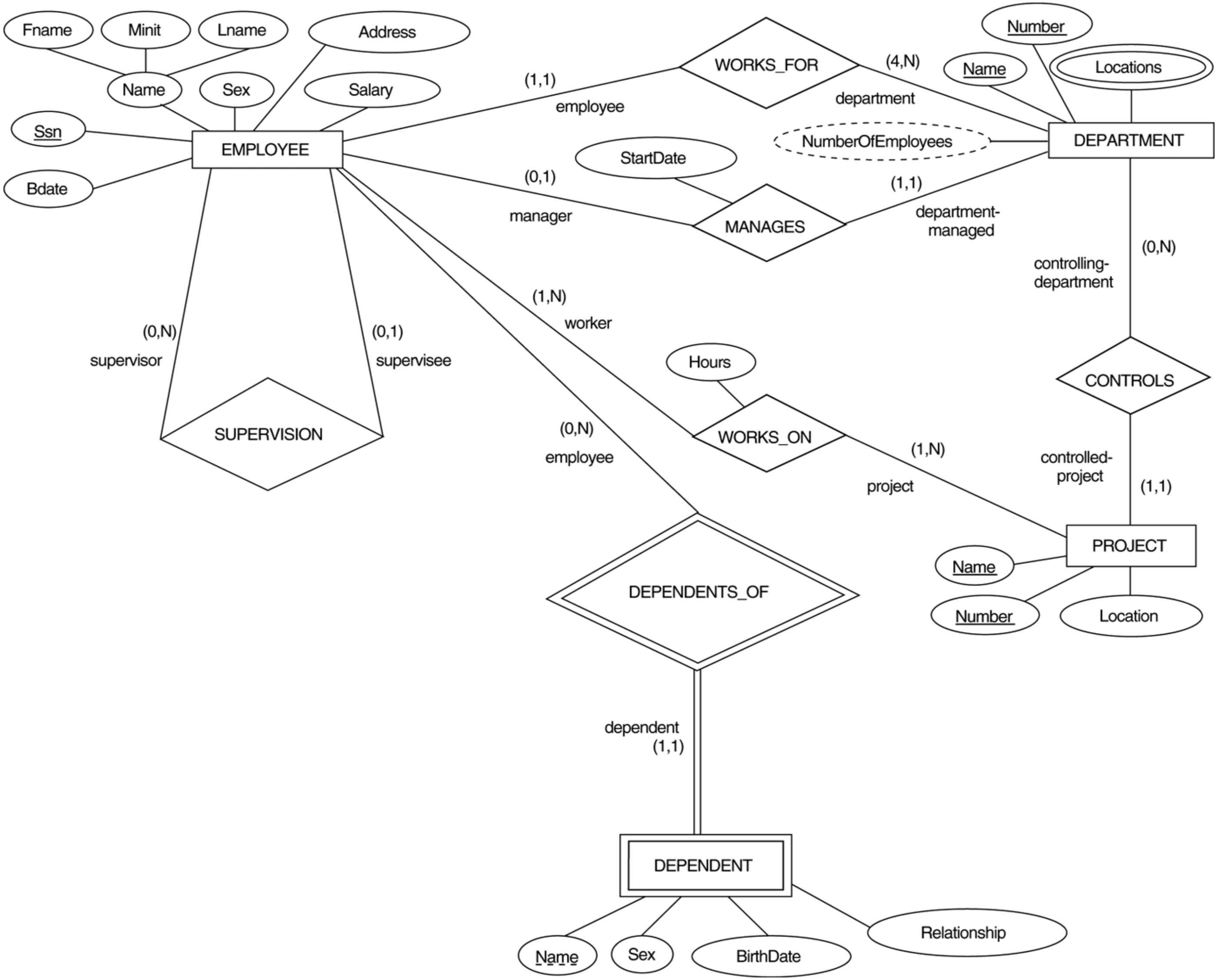
Demotion of DEPT entity type to Dept attribute

Alternative Notations for ER diagram

- Alternative design for specifying structural constraints.
- We associate a pair of integer numbers (min, max) with each participation of an entity type in a relationship type.
- min → minimum number of participation
- max→ maximum number of participation
- If min=0 , it is partial participation and min>0 total participation

ER diagram Design issues

Alternative ER diagram for the COMPANY database


Fname
Minit
Lname
Address
Name
Sex
Salary
Ssn
EMPLOYEE
Bdate
WORKS_FOR
(1,1)
employee
(4,N)
department
Number
Name
Locations
DEPARTMENT
NumberOfEmployees
StartDate
(0,1)
manager
MANAGES
(1,1)
department-managed
(0,N)
controlling-department
(1,N)
worker
(0,N)
supervisor
(0,1)
supervisee
SUPERVISION
Hours
WORKS_ON
(1,N)
project
CONTROLS
(0,N)
employee
controlled-project
(1,1)
PROJECT
Name
Number
Location
DEPENDENTS_OF
dependent
(1,1)
DEPENDENT
Name
Sex
BirthDate
Relationship