Chapter 3: Data Modeling Using the Entity-Relationship (ER) Model

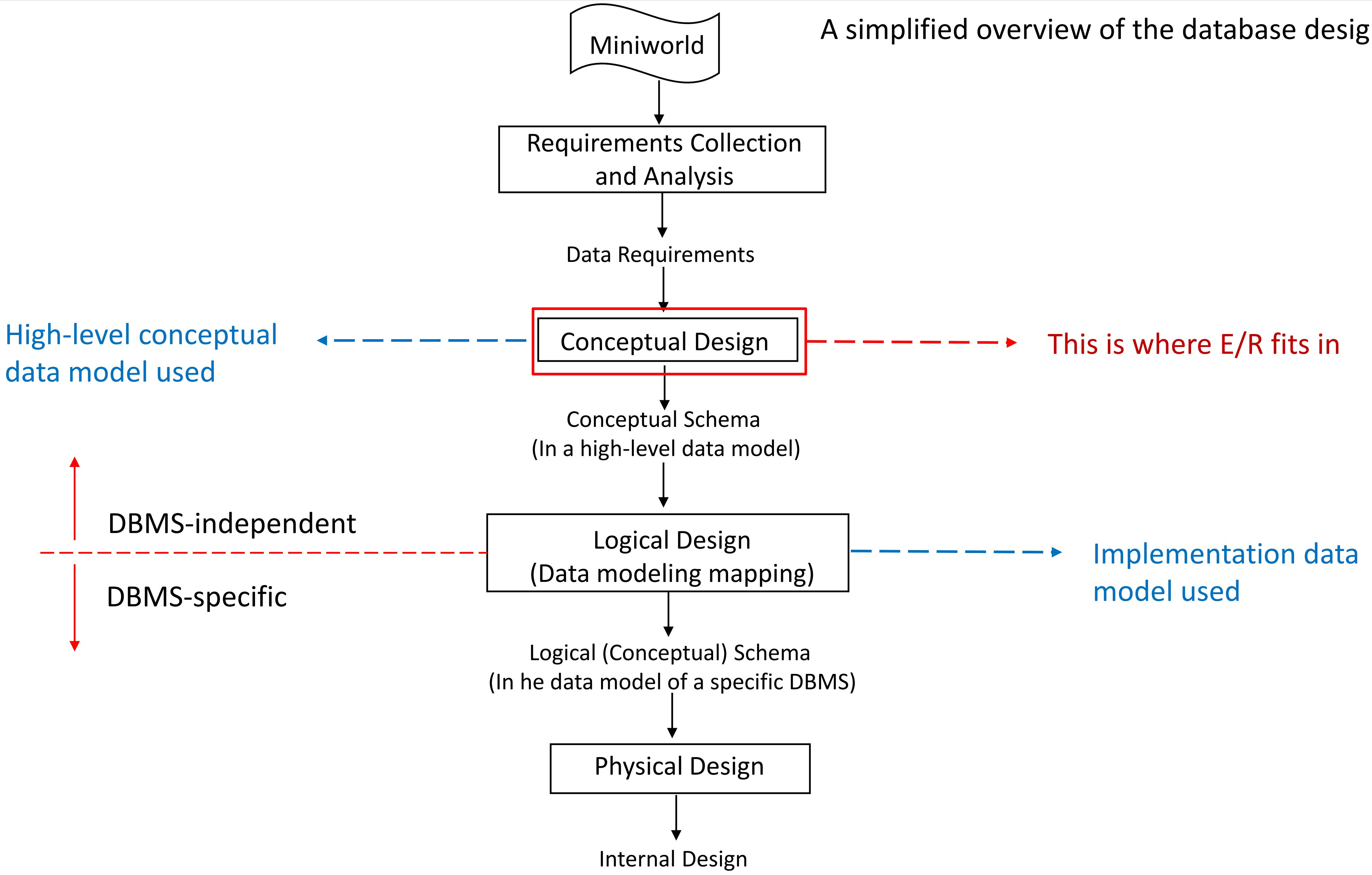
Database Design

- **Database design: Why do we need it?**
  - Agree on structure of the database before deciding on a particular implementation

- **Consider issues such as:**
  - What entities to model
  - How entities are related
  - What constraints exist in the domain
  - How to achieve <u>good</u> designs

- **Several formalisms exist**
  - We discuss one flavor of E/R diagrams

A simplified overview of the database design process



Miniworld

Requirements Collection and Analysis

Data Requirements

Conceptual Design

High-level conceptual data model used

This is where E/R fits in

Conceptual Schema (In a high-level data model)

DBMS-independent

DBMS-specific

Logical Design (Data modeling mapping)

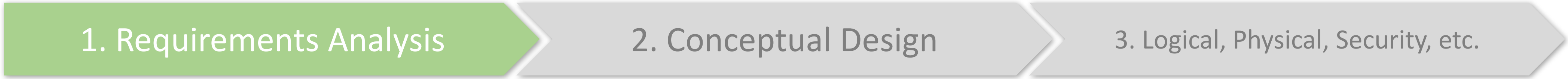
Implementation data model used

Logical (Conceptual) Schema (In he data model of a specific DBMS)

Physical Design

Internal Design

Database Design Process



1. Requirements Analysis

2. Conceptual Design

3. Logical, Physical, Security, etc.

1. Requirements analysis

- What is going to be stored?
- How is it going to be used?
- What are we going to do with the data?
- Who should access the data?

Technical and non-technical people are involved

Database Design Process (cont.)

1. Requirements Analysis

2. Conceptual Design

3. Logical, Physical, Security, etc.

2. Conceptual Design

- A high-level description of the database
- Sufficiently precise that technical people can understand it
- But, not so precise that non-technical people can't participate

This is where E/R fits in.

Database Design Process (cont.)

1. Requirements Analysis

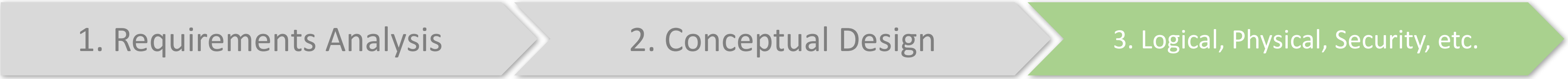
2. Conceptual Design

3. Logical, Physical, Security, etc.

3. More:

- Logical Database Design
- Physical Database Design
- Security Design

Database Design Process (cont.)



1. Requirements Analysis

2. Conceptual Design

3. Logical, Physical, Security, etc.

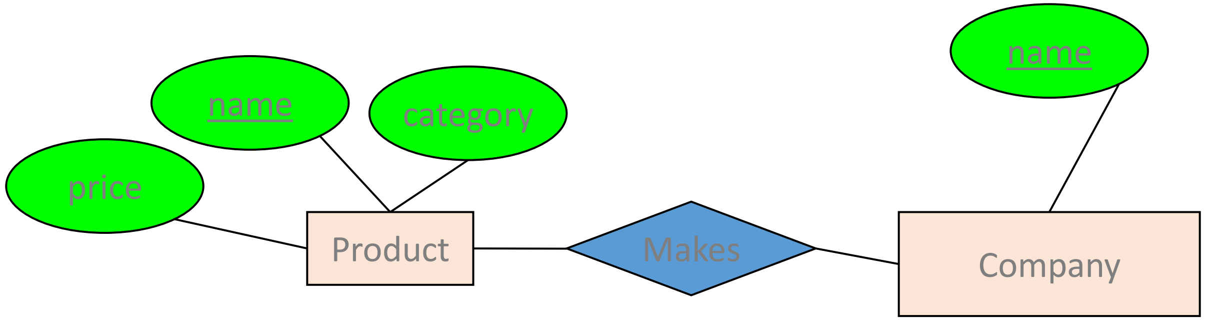
- A primary goal of database design is to decide what tables to create. Usually, there are two principles:
  - Capture all the information that needs to be captured by the underlying application.
  - Achieve the above with little redundancy

- The first principle is enforced with an entity relationship (ER) diagram, while the second with normalization.

This lecture focuses on the ER diagram.

Database Design Process (cont.)

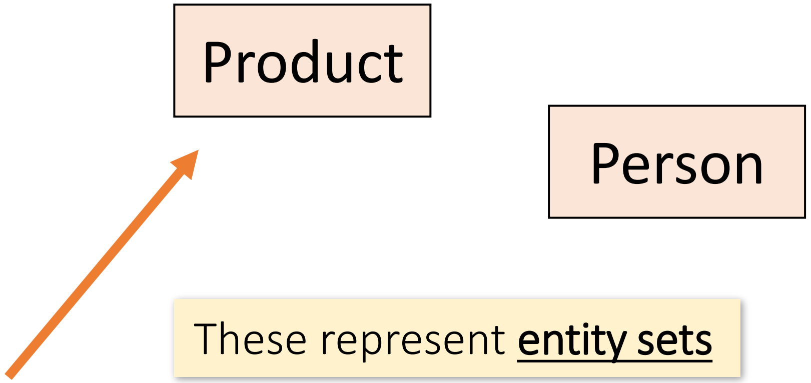
- An ER diagram is a pictorial representation of the information that can be captured by a database. Such a “picture” serves two purposes:
    - It allows database professionals to describe an overall design concisely yet accurately.
    - (Most of) it can be easily transformed into the relational schema.



E/R is a *visual syntax* for DB design which is ***precise enough*** for technical points, but ***abstracted enough*** for non-technical people

Entities and Entity Sets

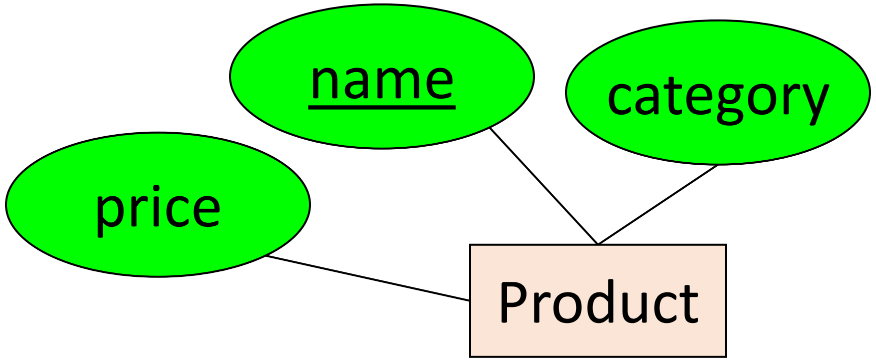
- **Entities** & **entity sets** are the primitive unit of the E/R model
  - Entities are the individual objects, which are members of entity sets
    - Ex: A specific person or product
  - Entity sets are the *classes* or *types* of objects in our model
    - Ex: Person, Product
    - *These are what is shown in E/R diagrams - as rectangles*
    - *Entity sets represent the sets of all possible entities*



These represent **entity sets**

Entities and Entity Sets (cont.)

- An entity set has **attributes**
  - Represented by ovals attached to an entity set

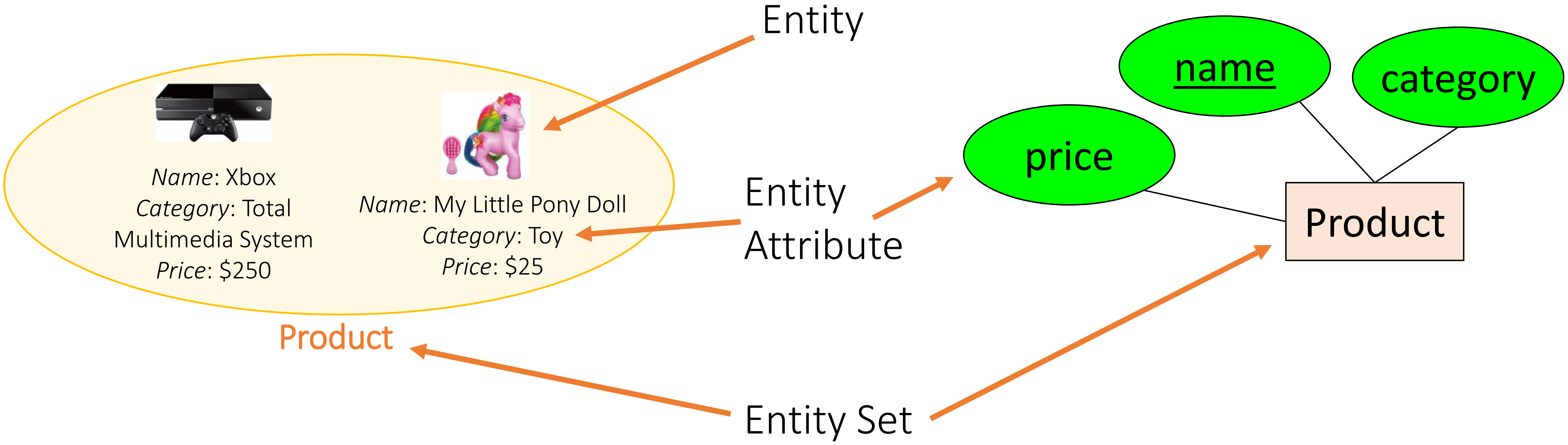


Shapes **are** important.
Colors **are not.**

Entities vs. Entity Sets (cont.)

Example:

Entities are **<u>not</u>** explicitly represented in E/R diagrams!



Name: Xbox
Category: Total Multimedia System
Price: $250

Name: My Little Pony Doll
Category: Toy
Price: $25

Product

Entity

Entity Attribute

Entity Set

<u>name</u>

category

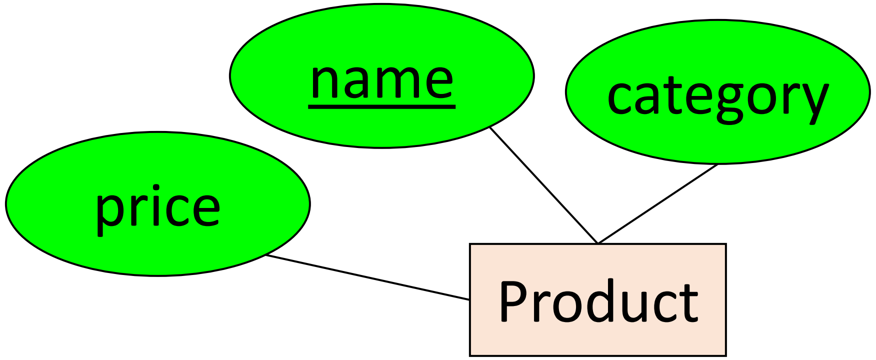
price

Product

Keys

- A *key* is a **minimal** set of attributes that uniquely identifies an entity.

Denote elements of the primary key by underlining.



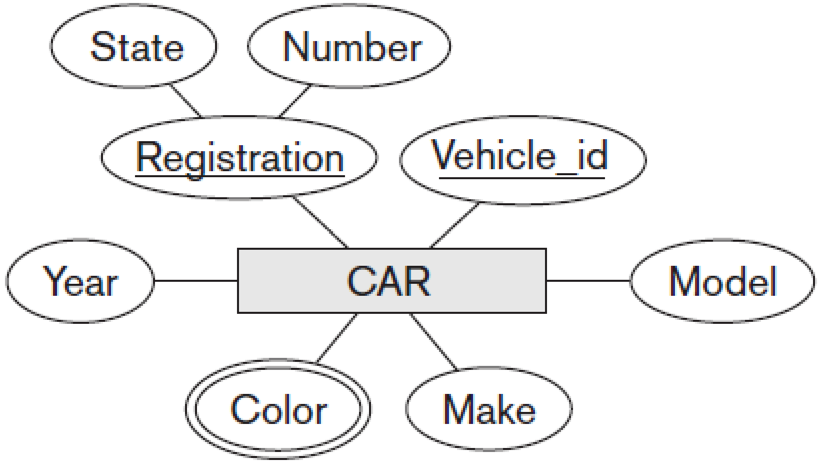
Here, {name, category} is **not** a key (it is not *minimal*).

If it were, what would it mean?

The E/R model forces us to designate a single **primary** key, though there may be multiple candidate keys

Keys (cont.)

Example:



CAR
Registration (Number, State), Vehicle_id, Make, Model, Year, {Color}

CAR$_1$
((ABC 123, TEXAS), TK629, Ford Mustang, convertible, 2004 {red, black})

CAR$_2$
((ABC 123, NEW YORK), WP9872, Nissan Maxima, 4-door, 2005, {blue})

CAR$_3$
((VSY 720, TEXAS), TD729, Chrysler LeBaron, 4-door, 2002, {white, blue})

⋮

Keys (cont.)

- Sometimes several attributes together form a key.

Student_number	Section_identifier	Grade
17	112	B
17	119	C
17	102	A
8	85	A
8	92	B
8	102	A
8	135	B

Initial Conceptual Design: Requirements Collection—COMPANY

- The company is organized into ***departments***
  - A unique name, a unique number
  - A particular ***employee*** who manages the department
  - The start date when that manager began managing the department
  - Have several locations

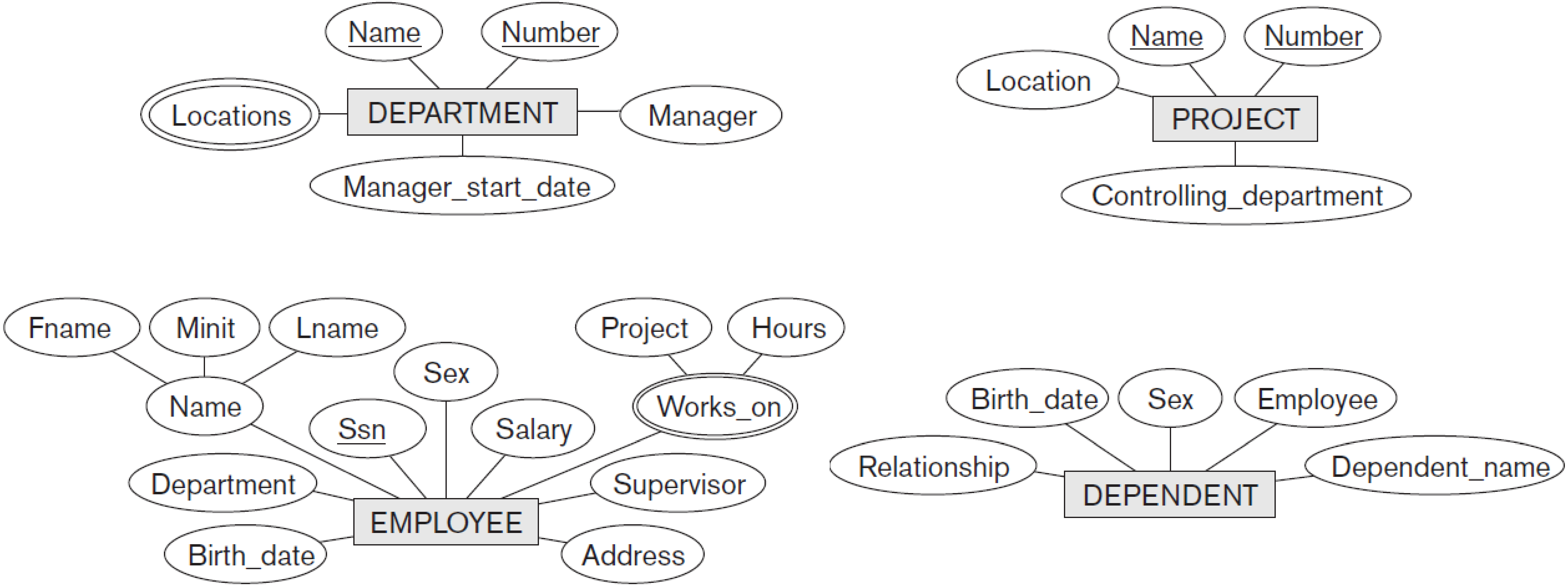
- A ***department*** controls a number of ***projects***
  - A unique name, a unique number, and a single location

Initial Conceptual Design: Requirements Collection—COMPANY (cont.)

- The database will store each ***employee***'s:
  - Name, SSN, address, salary, gender, birth date
  - An ***employee*** is assigned to one ***department***
  - An ***employee*** may work on several ***projects***, which are not necessarily controlled by the same ***department***
  - Required to track the current number of hours per week that an employee works on each ***project***
  - Required to track the direct supervisor of each employee (who is another employee)

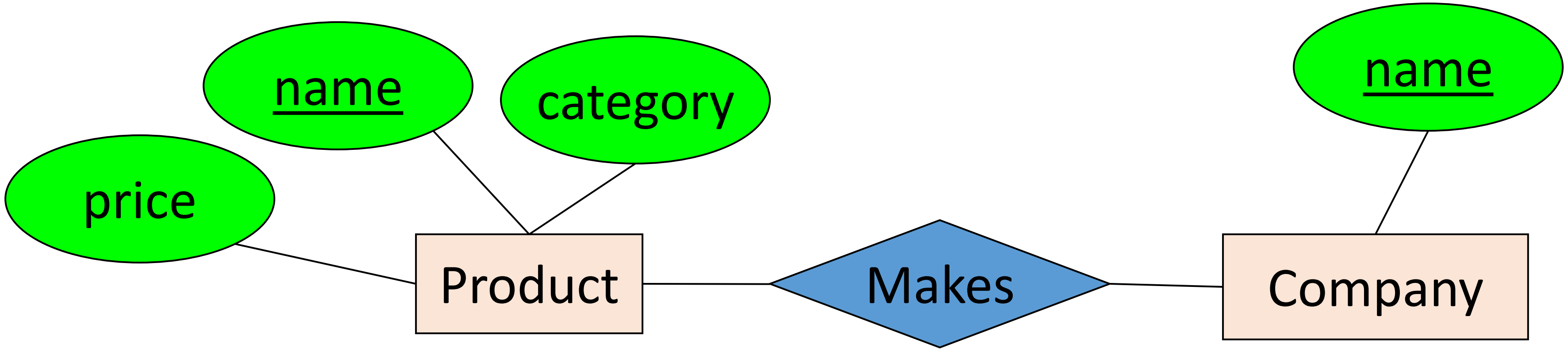
- The database will keep track of the **dependents** of each **employee**
  - First name, gender, birth date, and relationship to the employee

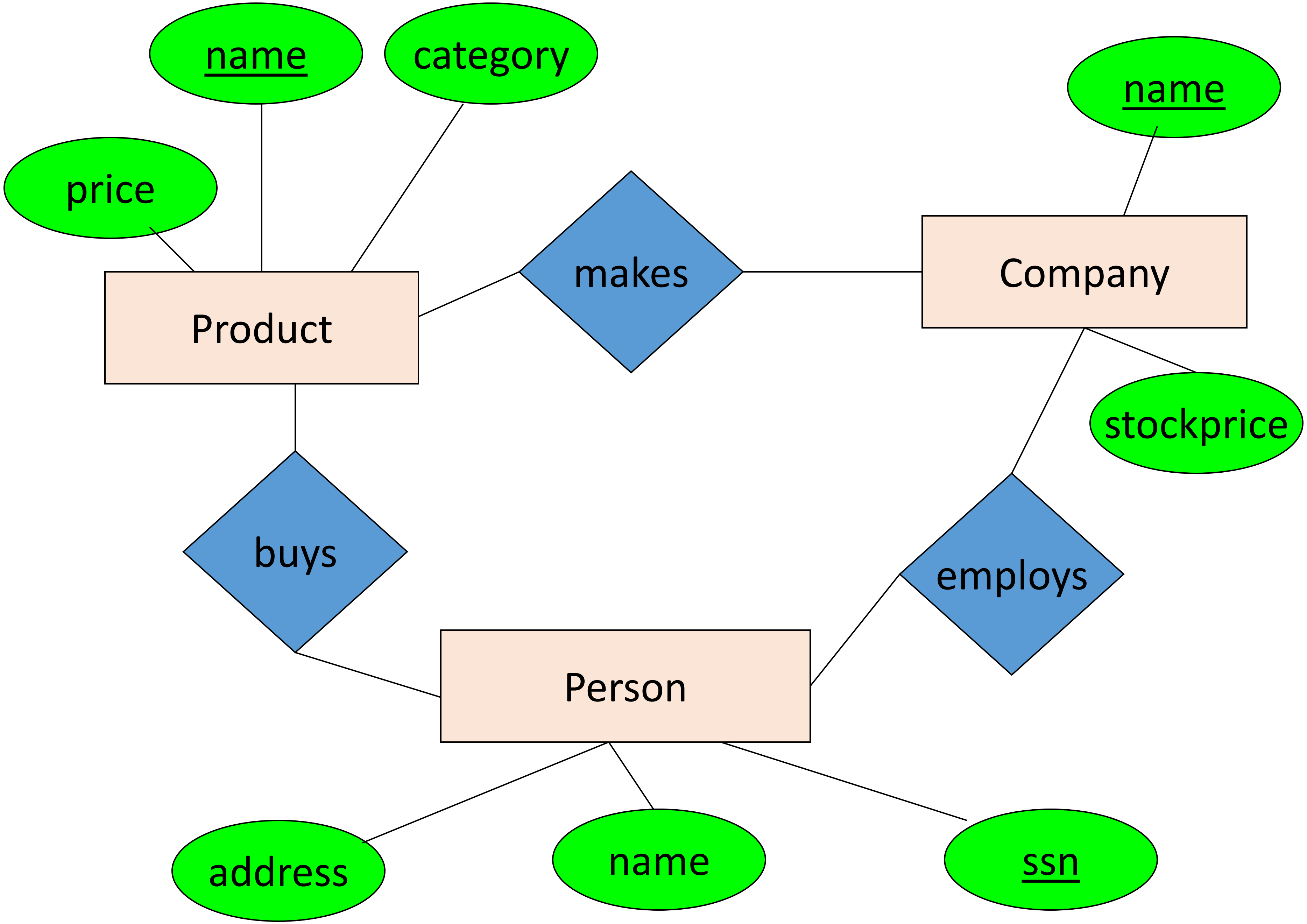
Initial Conceptual Design: Preliminary Design of Entity Sets



The R in E/R: **Relationships**

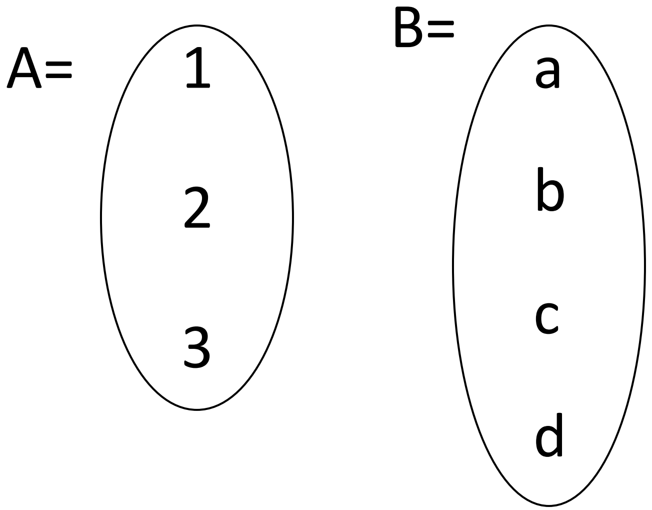
- A **relationship** is between two entities




name
category
price
name
Product
makes
Company
stockprice
buys
employs
Person
address
name
ssn

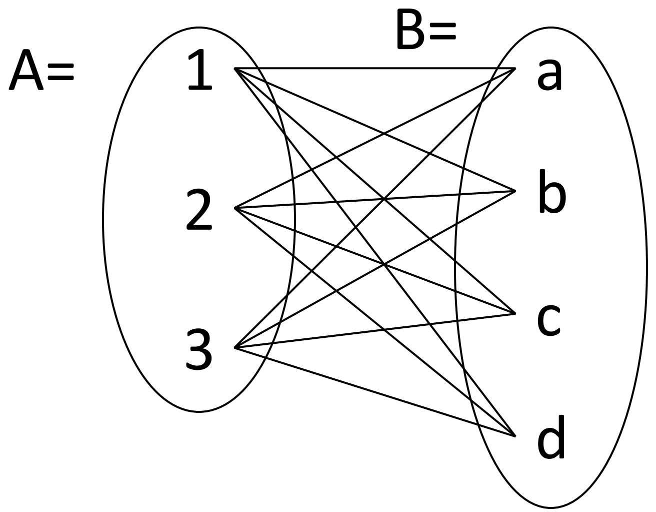
What is a Relationship?

- ***A mathematical definition:***
  - Let A, B be sets
    - *A={1,2,3},   B={a,b,c,d}*



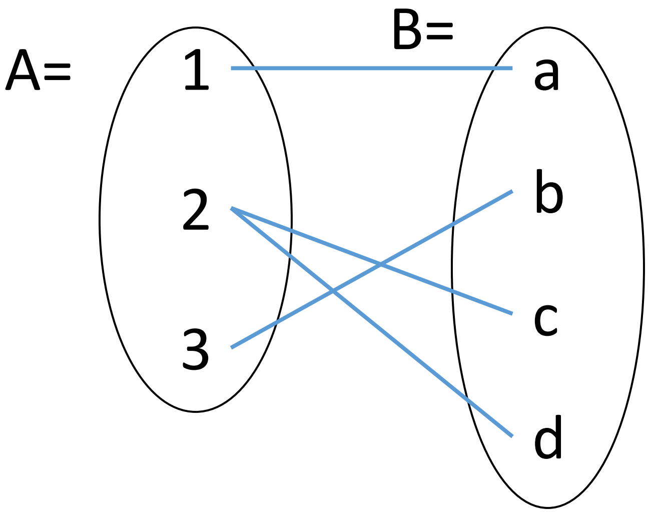
What is a Relationship?

- ***A mathematical definition:***
  - Let A, B be sets
    - *A={1,2,3}, B={a,b,c,d}*
  - A x B (the ***cross-product***) is the set of all pairs (a,b)
    - *A $\times$ B = {(1,a), (1,b), (1,c), (1,d), (2,a), (2,b), (2,c), (2,d), (3,a), (3,b), (3,c), (3,d)}*



What is a Relationship?

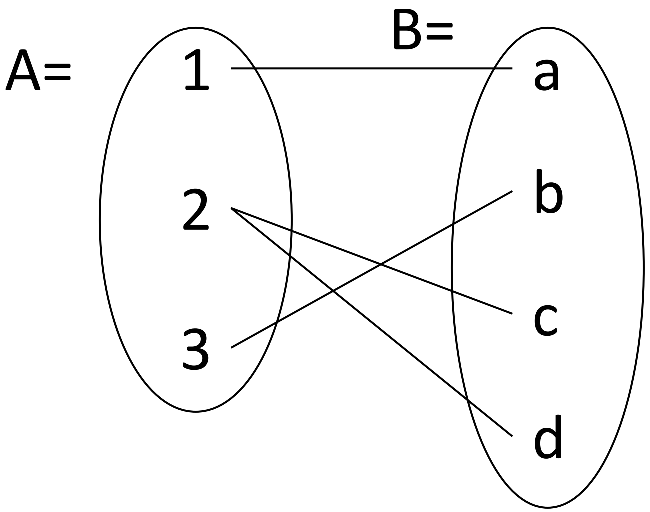
- ***A mathematical definition:***
  - Let A, B be sets
    - *A={1,2,3}, B={a,b,c,d},*
  - A x B (the ***cross-product***) is the set of all pairs (a,b)
    - $A \times B$ *= {(1,a), (1,b), (1,c), (1,d), (2,a), (2,b), (2,c), (2,d), (3,a), (3,b), (3,c), (3,d)}*
  - **We define a <u>relationship</u> to be a subset of A x B**
    - *R = {(1,a), (2,c), (2,d), (3,b)}*

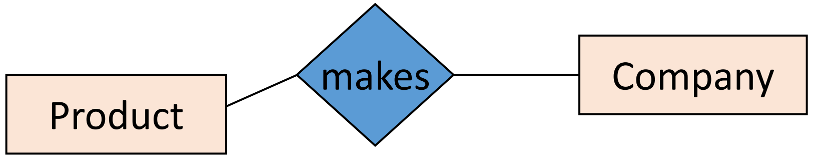


What is a Relationship?

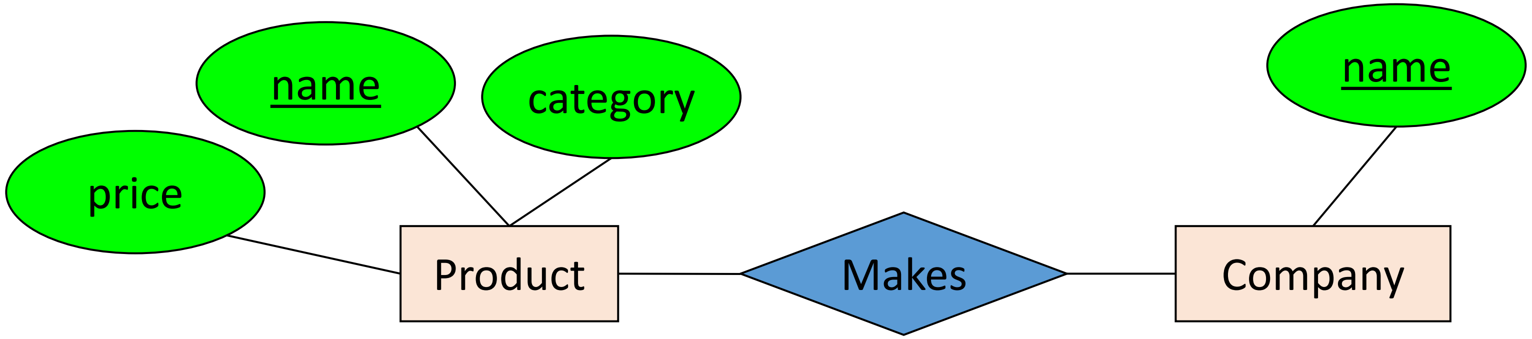
- ***A mathematical definition:***
  - Let A, B be sets
  - A x B (the ***cross-product***) is the set of all pairs
  - A relationship is a subset of A x B

- **Makes** is relationship- it is a ***subset*** of **Product × Company**:





What is a Relationship?



A <u>**relationship**</u> between **entity sets P and C** is a ***subset of all possible pairs of entities in P and C***, with tuples uniquely identified by ***P and C's keys***

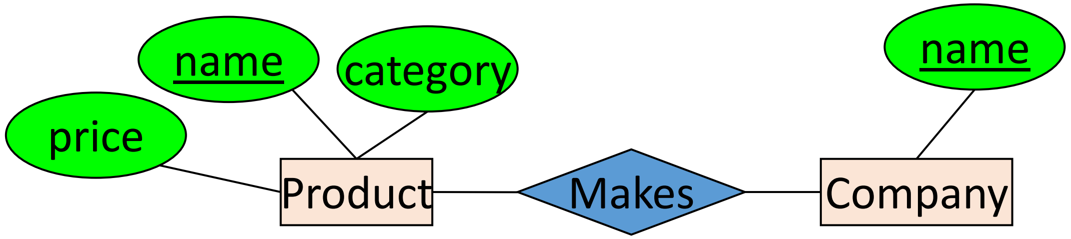
What is a Relationship?

Company

name
GizmoWorks
GadgetCorp

Product

name	category	price
Gizmo	Electronics	$9.99
GizmoLite	Electronics	$7.50
Gadget	Toys	$5.50



A <u>relationship</u> between **entity sets P and C** is a ***subset of all possible pairs of entities in P and C***, with tuples uniquely identified by ***P and C's keys***

What is a Relationship?

Company

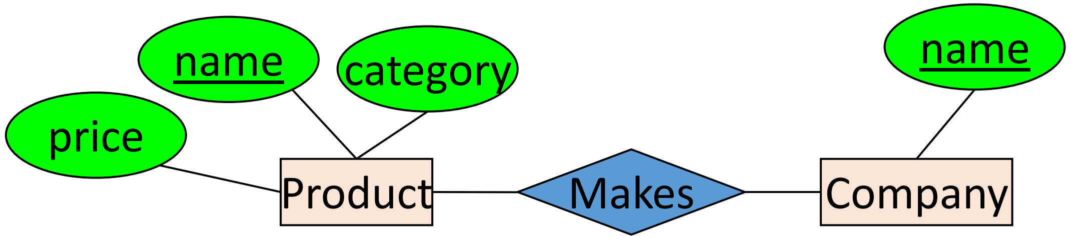
name
GizmoWorks
GadgetCorp

Product

name	category	price
Gizmo	Electronics	$9.99
GizmoLite	Electronics	$7.50
Gadget	Toys	$5.50

Company C × Product P

C.name	P.name	P.category	P.price
GizmoWorks	Gizmo	Electronics	$9.99
GizmoWorks	GizmoLite	Electronics	$7.50
GizmoWorks	Gadget	Toys	$5.50
GadgetCorp	Gizmo	Electronics	$9.99
GadgetCorp	GizmoLite	Electronics	$7.50
GadgetCorp	Gadget	Toys	$5.50



A relationship between **entity sets P and C** is a ***subset of all possible pairs of entities in P and C***, with tuples uniquely identified by ***P and C's keys***

What is a Relationship?

Company

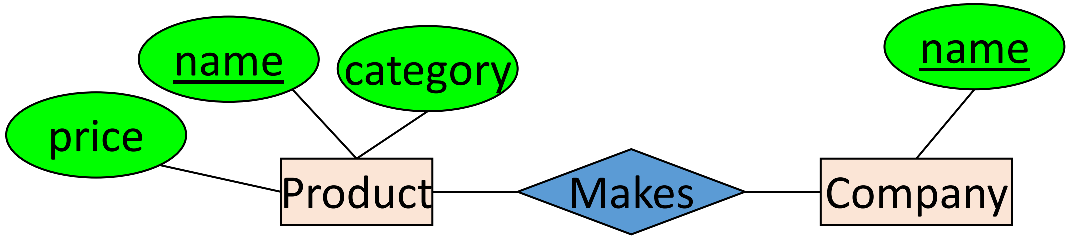
name
GizmoWorks
GadgetCorp

Product

name	category	price
Gizmo	Electronics	$9.99
GizmoLite	Electronics	$7.50
Gadget	Toys	$5.50

Company C × Product P

C.name	P.name	P.category	P.price
GizmoWorks	Gizmo	Electronics	$9.99
GizmoWorks	GizmoLite	Electronics	$7.50
GizmoWorks	Gadget	Toys	$5.50
GadgetCorp	Gizmo	Electronics	$9.99
GadgetCorp	GizmoLite	Electronics	$7.50
GadgetCorp	Gadget	Toys	$5.50



Makes

C.name	P.name
GizmoWorks	Gizmo
GizmoWorks	GizmoLite
GadgetCorp	Gadget

A relationship between **entity sets P and C** is a ***subset of all possible pairs of entities in P and C***, with tuples uniquely identified by ***P and C's keys***

What is a Relationship?

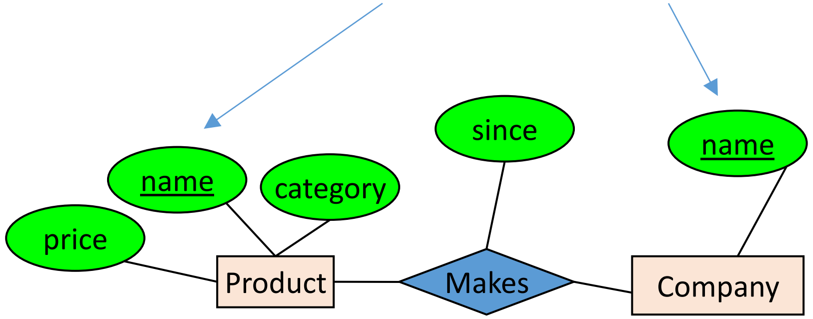
- There can only be **one relationship for every unique combination of entities**

This follows from our mathematical definition of a relationship- it's a SET!

- This also means that **the relationship is uniquely determined by the keys of its entities**

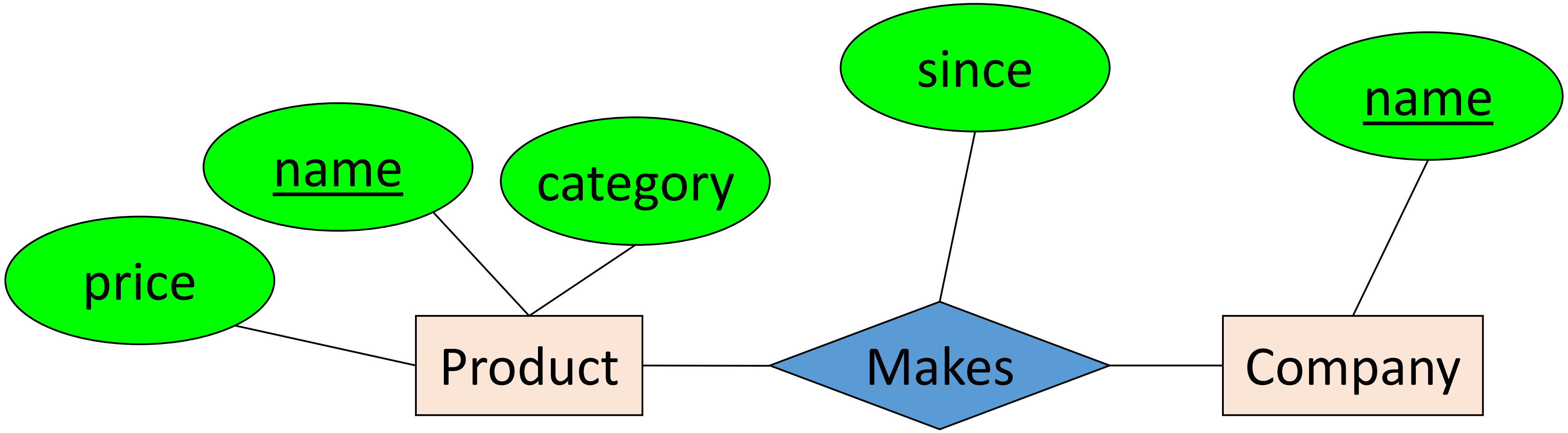
- *Example: the "key" for Makes (to right) is {Product.name, Company.name}*

$\text{Key}_{\text{Makes}} = \text{Key}_{\text{Product}} \cup \text{Key}_{\text{Company}}$



Relationships and Attributes

- Relationships may have attributes as well.

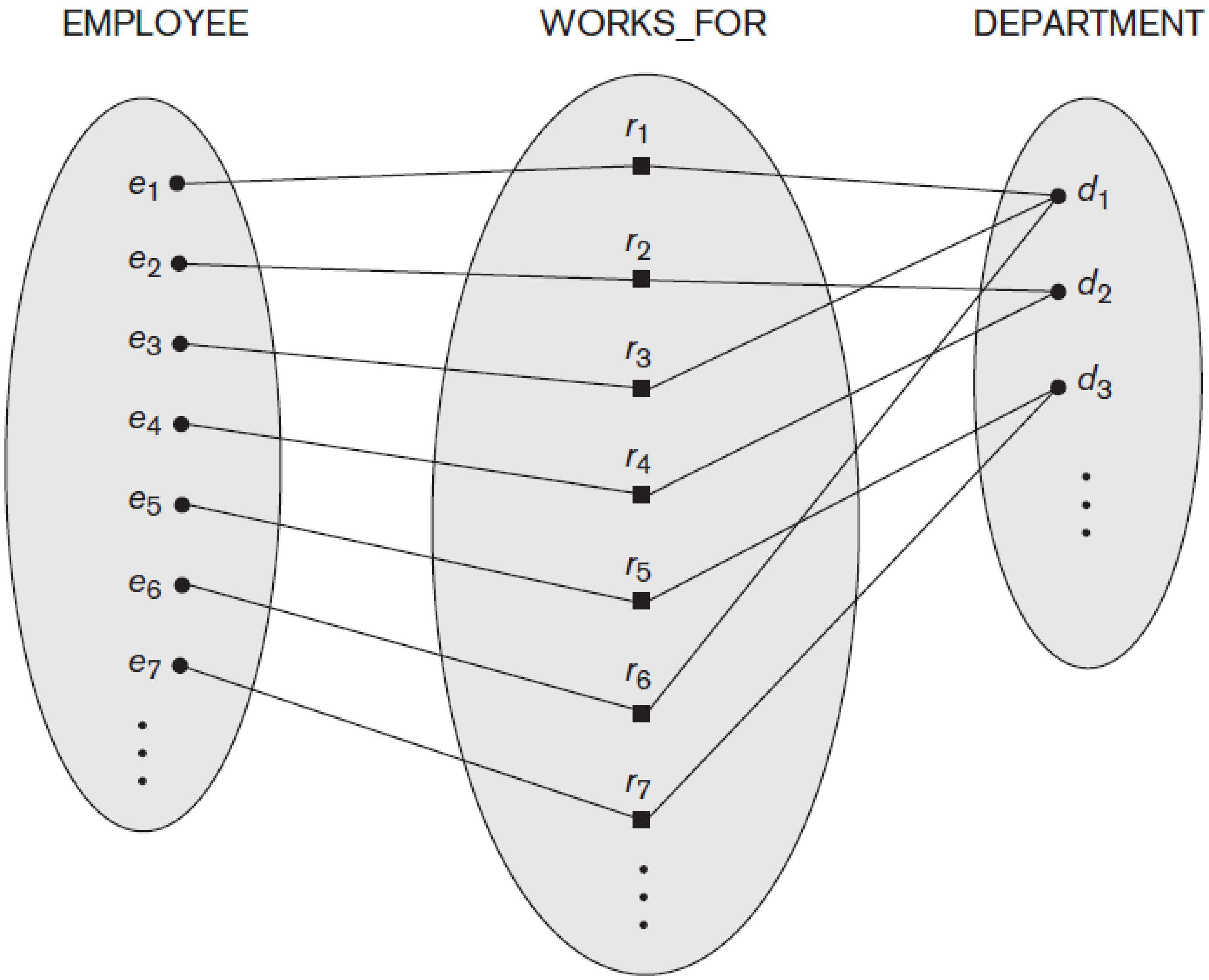


For example: "since" records when company started making a product

Constraints on Relationships

- Constraints
  - To limit the possible combination of entities that may participate in the corresponding relationship set
  - Determined from the miniworld situation

Constraints on Relationships (cont.)



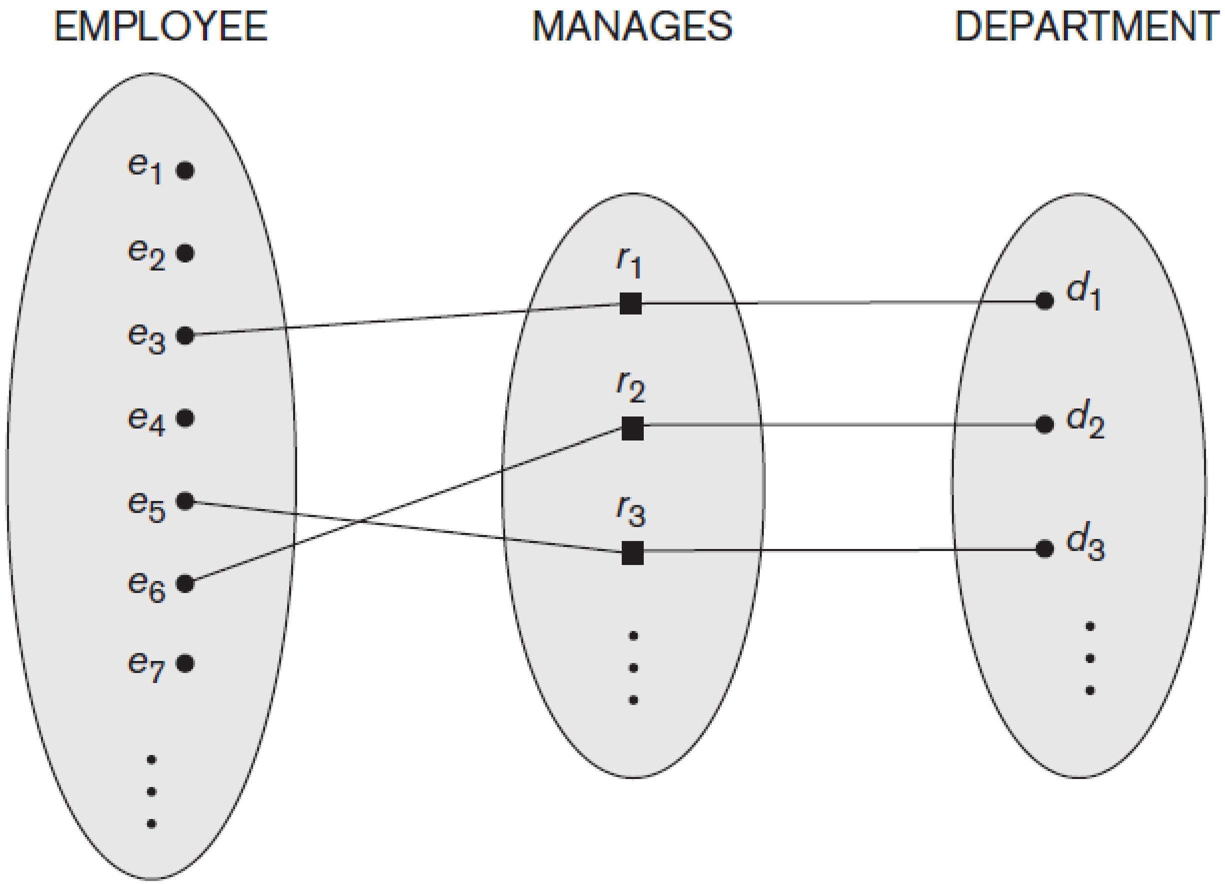
Constraint: each employee must work for exactly one department

Two Types of Relationship Constraints (1)

- Cardinality ratios
  - Specifies the maximum number of relationship instances that an entity can participate in
  - E.g., in WORKS_FOR relationship, DEPARTMENT : EMPLOYEE is of cardinality ratio 1 : N
  - Means what?

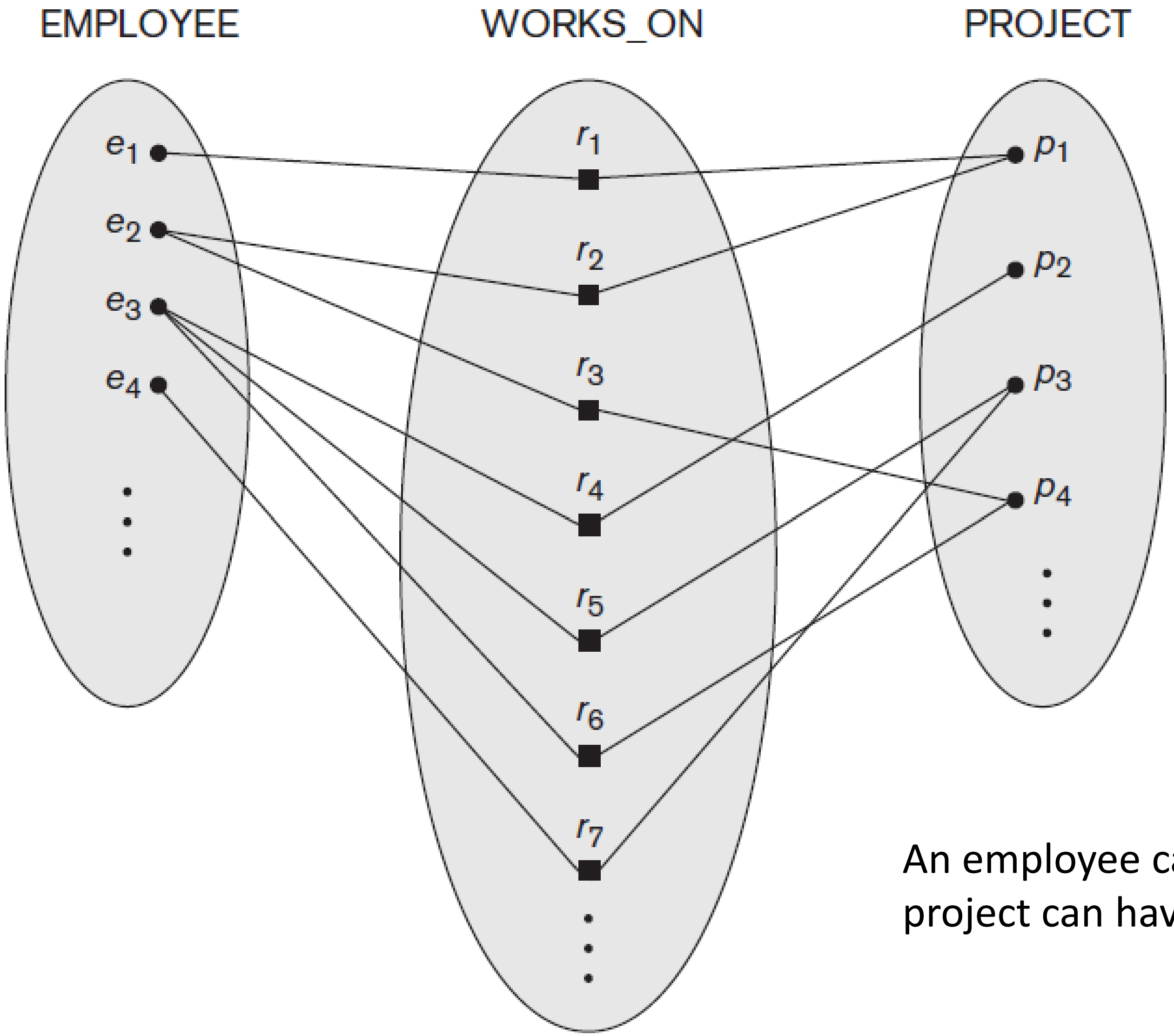
- Possible cardinality ratios
  - 1 : 1
  - 1 : N
  - N : 1
  - M : N

A Running Example of 1 : 1



An employee can manage at most one department and a department can have at most one manager.

A Running Example of M : N



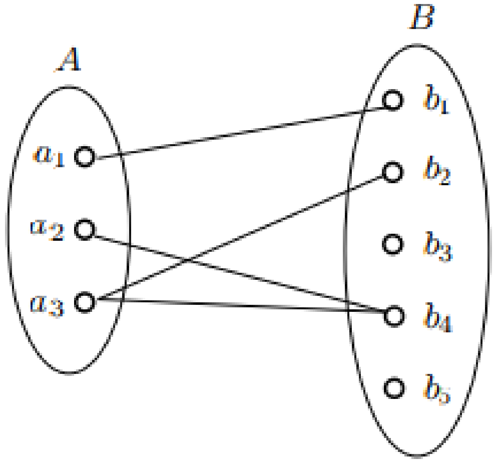
An employee can work on several projects and a project can have several employees

Two Types of Relationship Constraints (2)

- Participation constraints
  - Specifies the **minimum** number of relationship instances that each entity can participate in (also called minimum cardinality constraint)
    - *Total participation*
    - *Partial participation*

Two Types of Relationship Constraints (2) (cont.)

- Let *R* be a relationship set between entity sets *A* and *B*.
- The participation of *A* is **total** if every entity of A must participate in at least one relationship in *R*
- Otherwise, the participation of A is **partial**

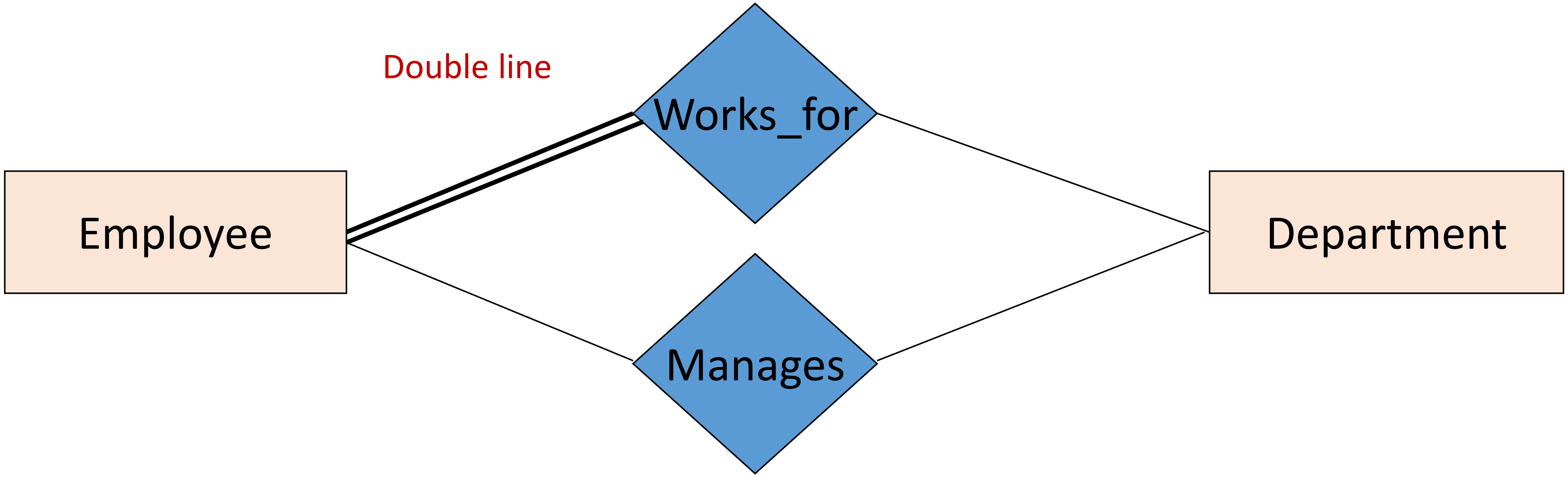


What's the participation of A?

What's the participation of B?

Two Types of Relationship Constraints (2) (cont.)

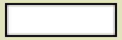
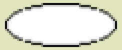
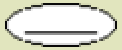
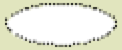
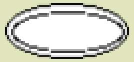
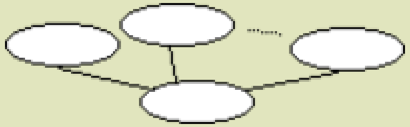
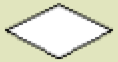
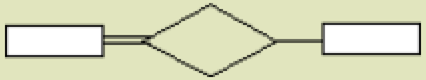
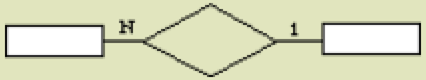
- E.g., “every employee must work for a department”



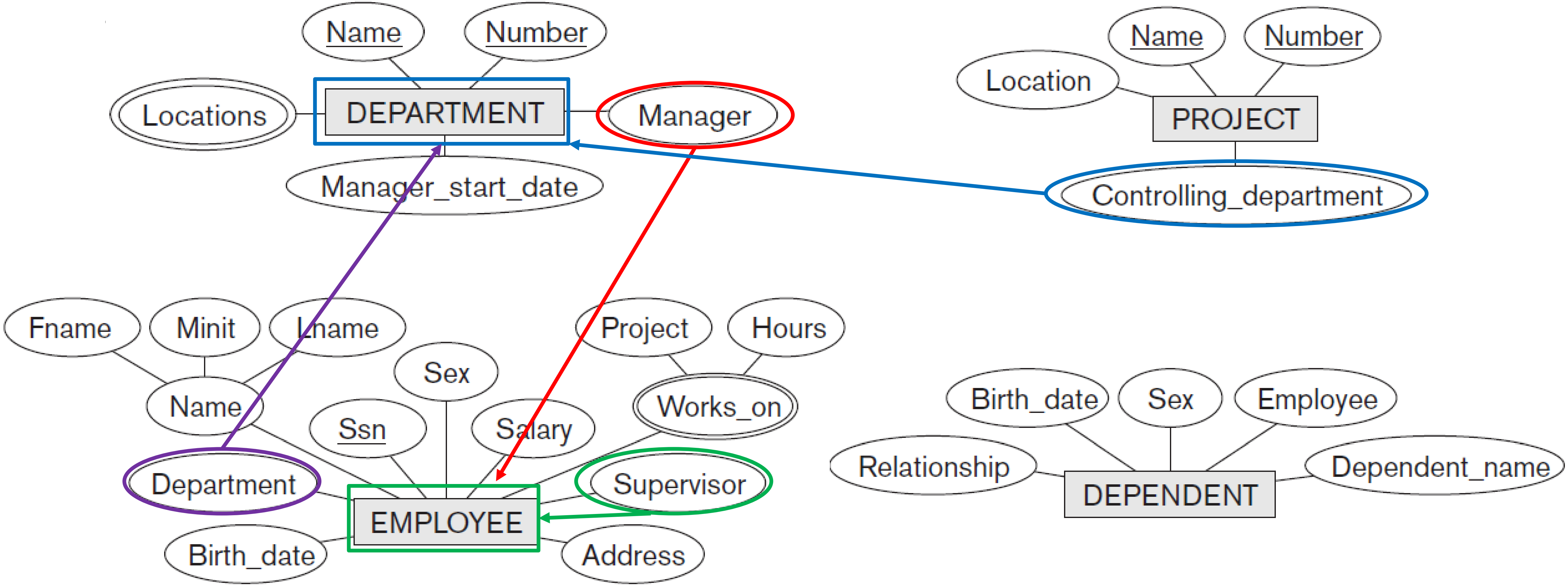
- We do not expect every employee to mange a department

We can include in an ER diagram a participation constraint in which participation of Employee in Works_for is **total**

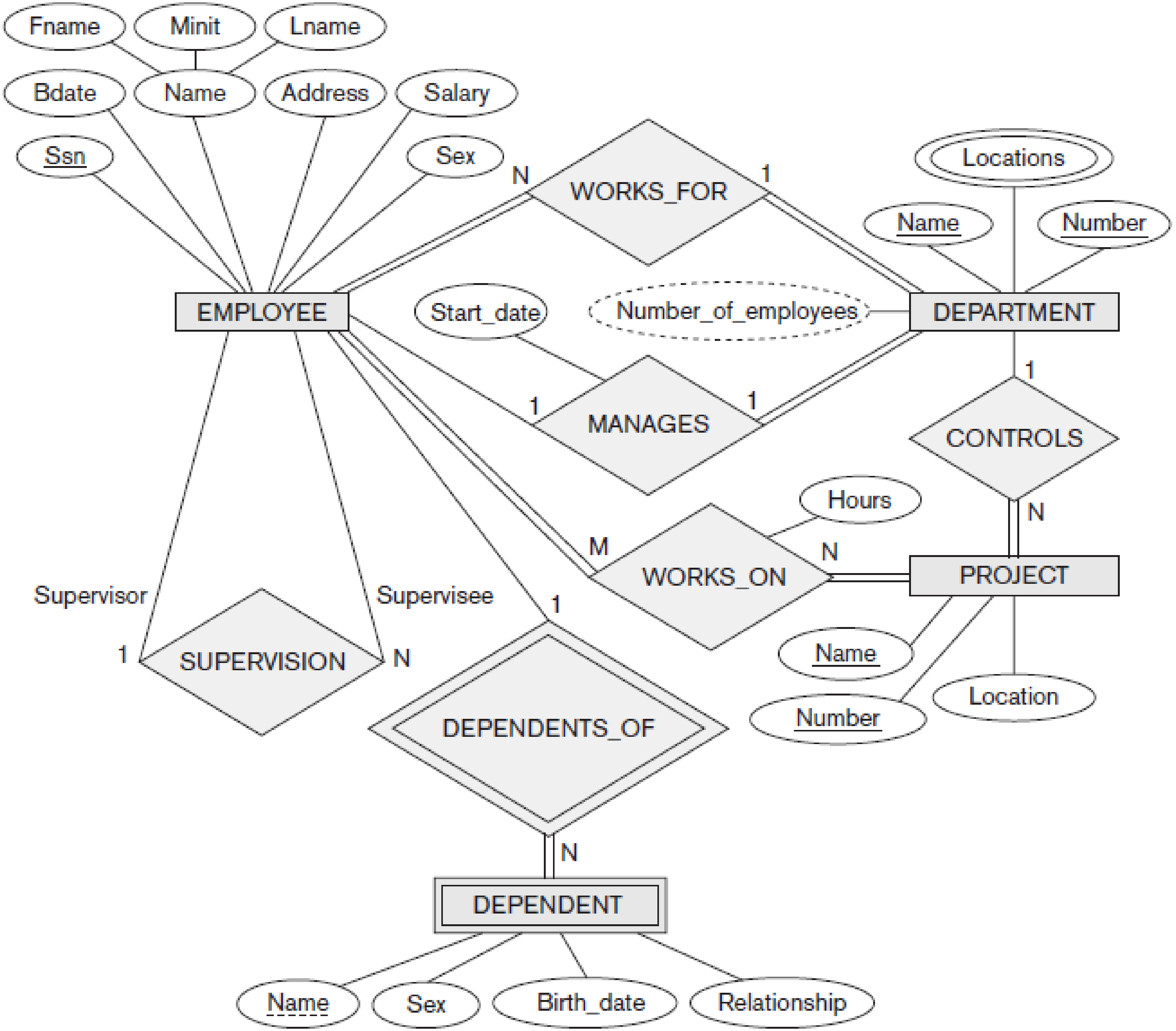
Summary of the ER Diagram Notation

Notation	**Meaning**
	Entity type
	Attribute
	Key attribute
	Derived attribute
	Multivalued attribute
	Composite attribute
	Relationship type
	Total participation
	Many-to-one relationship
	Weak entity type with identifying relationship

Initial Conceptual Design: Refine It

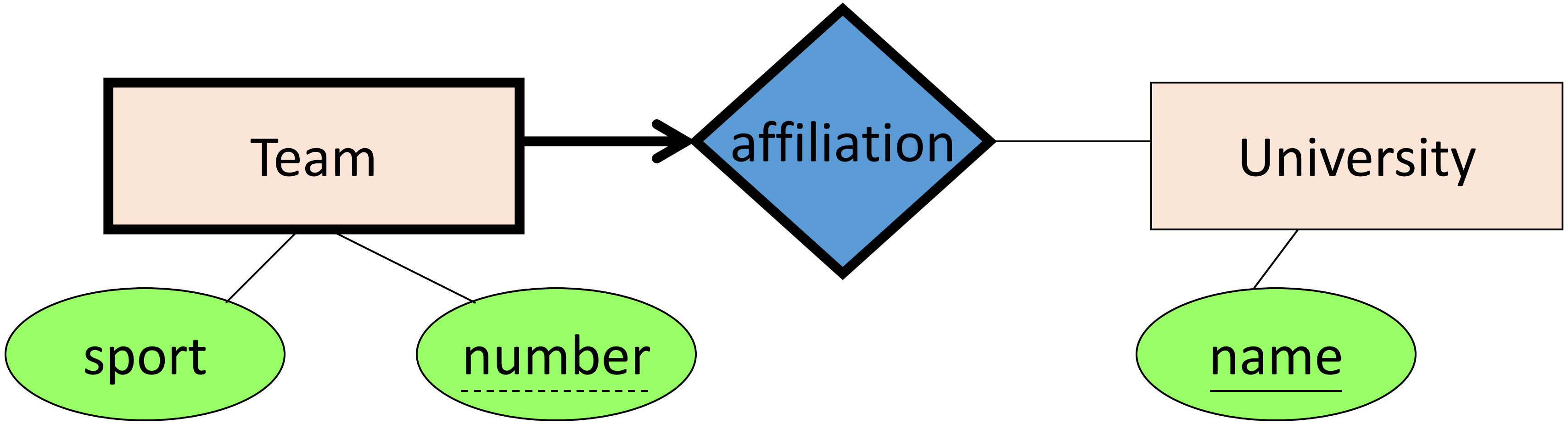


In the **ER model**, these references should not be represented as **attributes** but as **relationships**


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
Location
Number
DEPENDENTS_OF
N
DEPENDENT
Name
Sex
Birth_date
Relationship

Weak Entity Sets

Entity sets are *weak* when their key comes from other classes to which they are related.

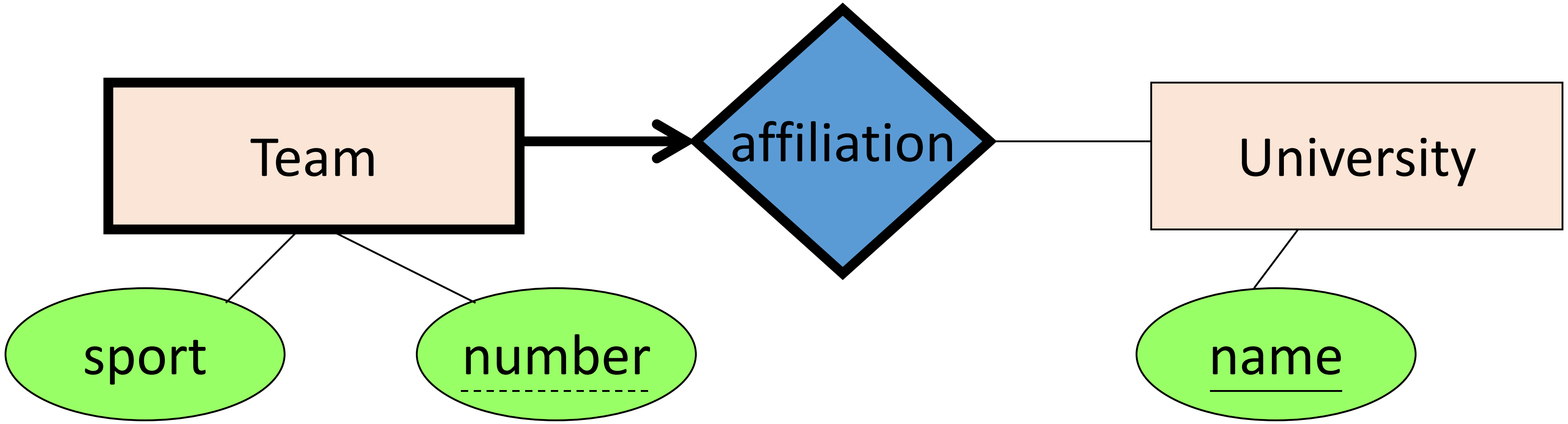


"Football team" v. "***The GSU*** Football team" *(E.g., GT has a football team too, sort of)*

Entity types that do not have key attributes of their own are called weal entity types.

Weak Entity Sets (cont.)

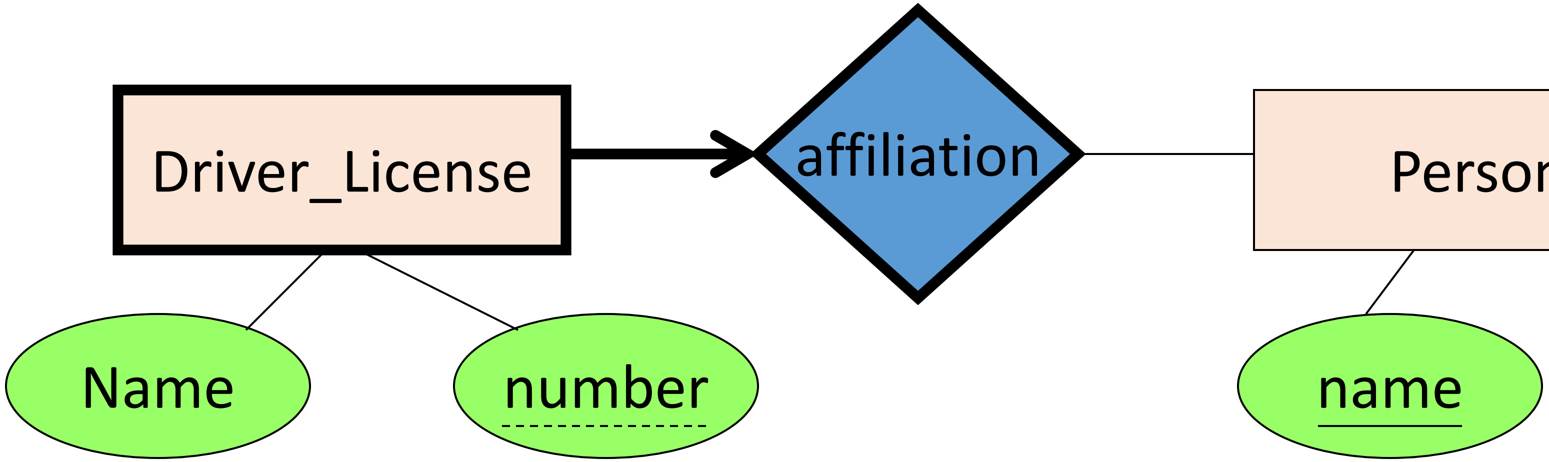
Entity sets are *weak* when their key comes from other classes to which they are related.



- *number* is a *partial key*. (denote with dashed underline).
- University is called the *identifying owner*.
- Participation in affiliation must be total. Why?

Weak Entity Sets (cont.)

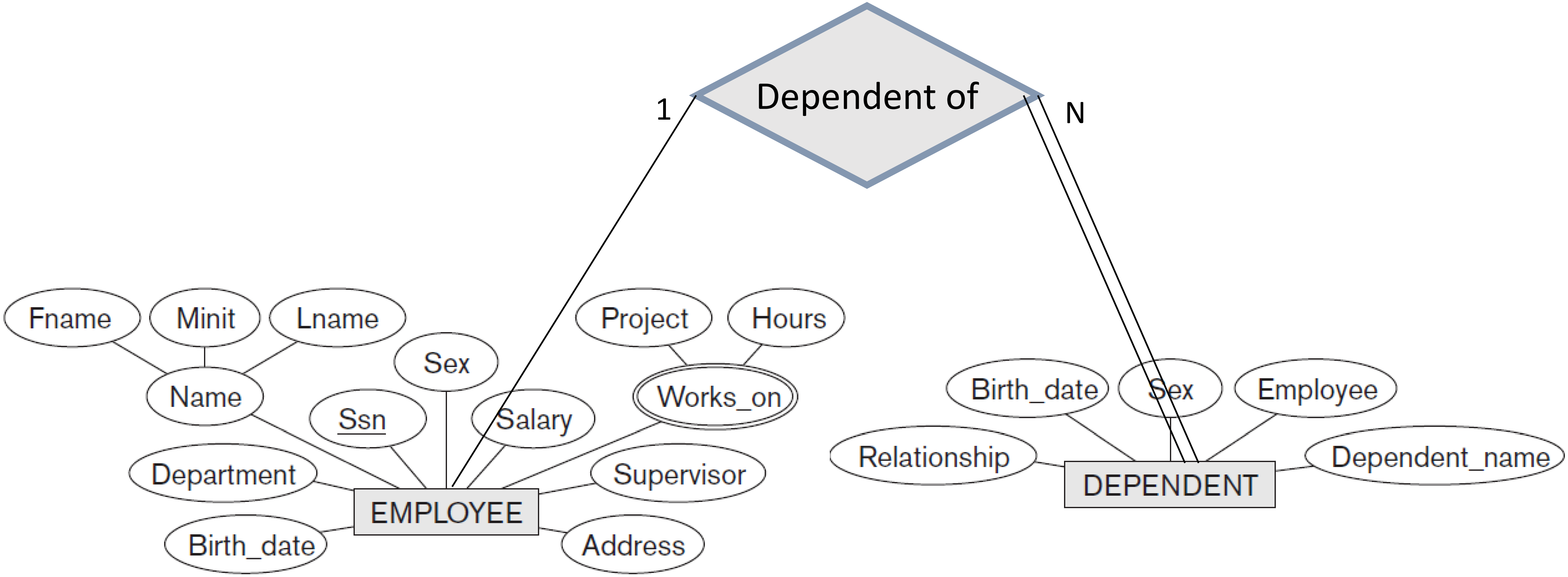
- However, **not every** existence dependency results in a weak entity type



ADRIVER_LICENSE entity **cannot** exist **unless** it is related to a PERSON entity.
Even though it has its own key (License_number) and hence is not a weak entity.

Weak Entity Sets (cont.)

- Dependents with same values are identified as distinct entities only after determining the particular related employee


Dependent of
1
N
Fname
Minit
Lname
Name
Sex
Ssn
Salary
Project
Hours
Works_on
Department
Supervisor
EMPLOYEE
Birth_date
Address
Birth_date
Sex
Employee
Relationship
Dependent_name
DEPENDENT