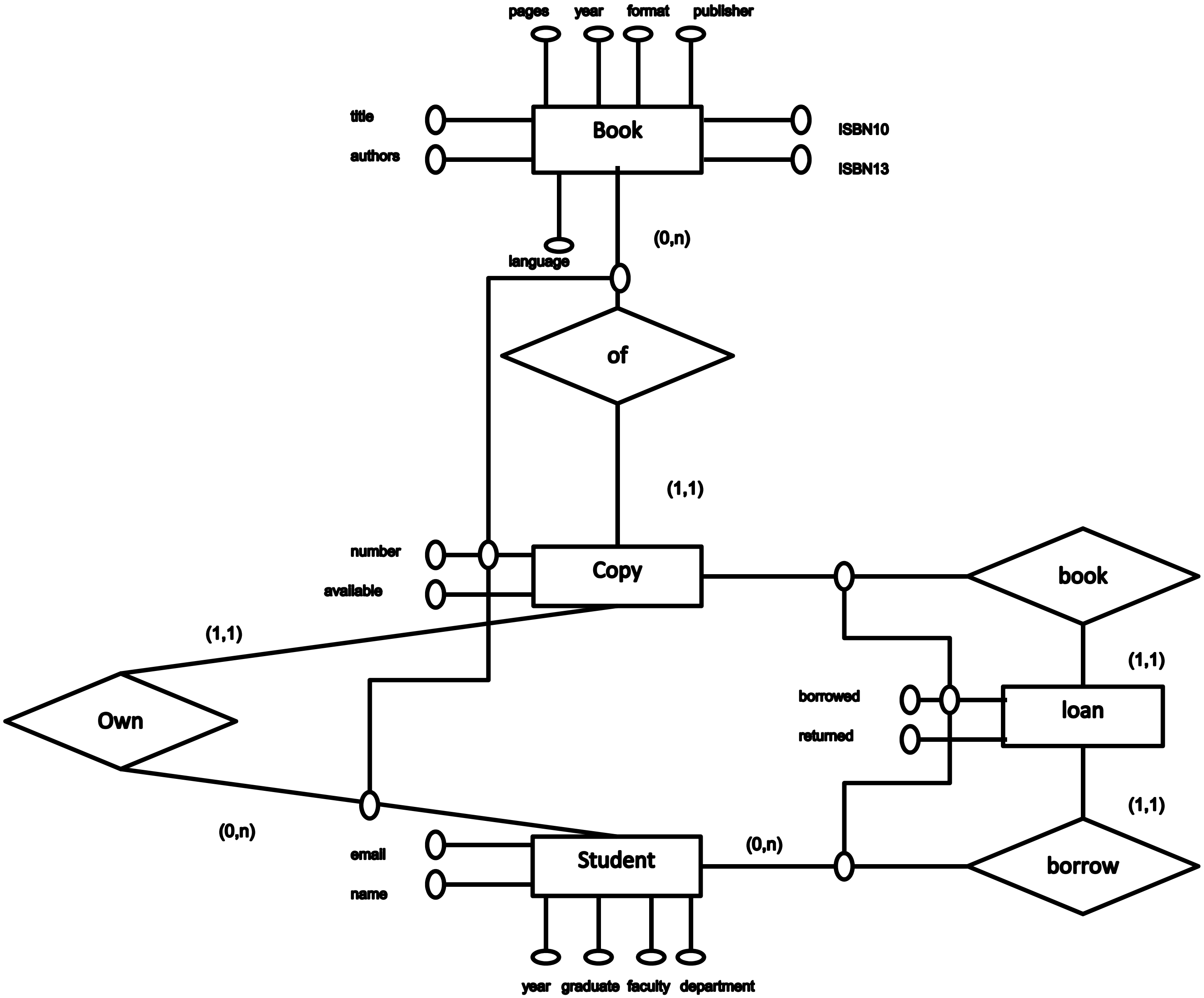
In the Lecture Series Introduction to Database Systems

Conceptual Modeling

Presented by Stéphane Bressan

Entity-relationship Diagram from theTutorial



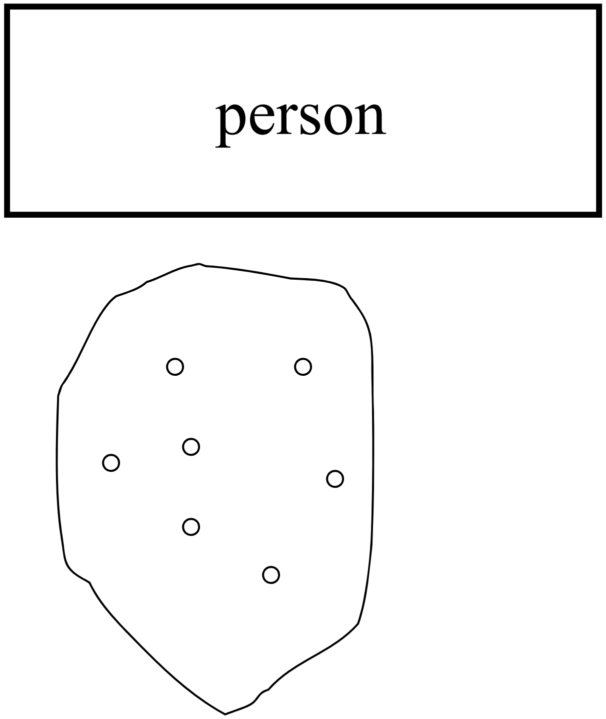
Introduction to Database Systems

Entity Relationship

- The Entity-relationship model is a graphical model for representing the conceptual model for the data centric design of an application

Entities and Entity Sets

- Entities are identifiable “things”
- The named box represents a set of entities or entity set

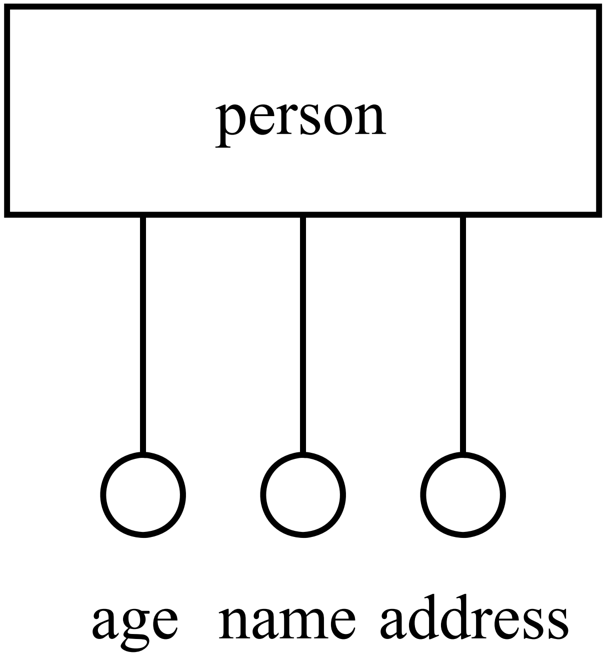


Attributes,Values and Value Sets

- The E-R model is value-oriented
- Values can be integer, strings, or atoms

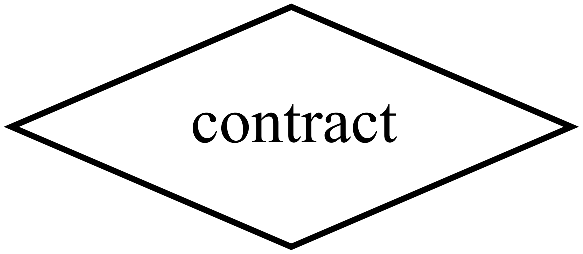
Attributes of Entities

- Entities can have attributes
- All entities in one entity set have the same attributes
- However the attributes take different values for each entities



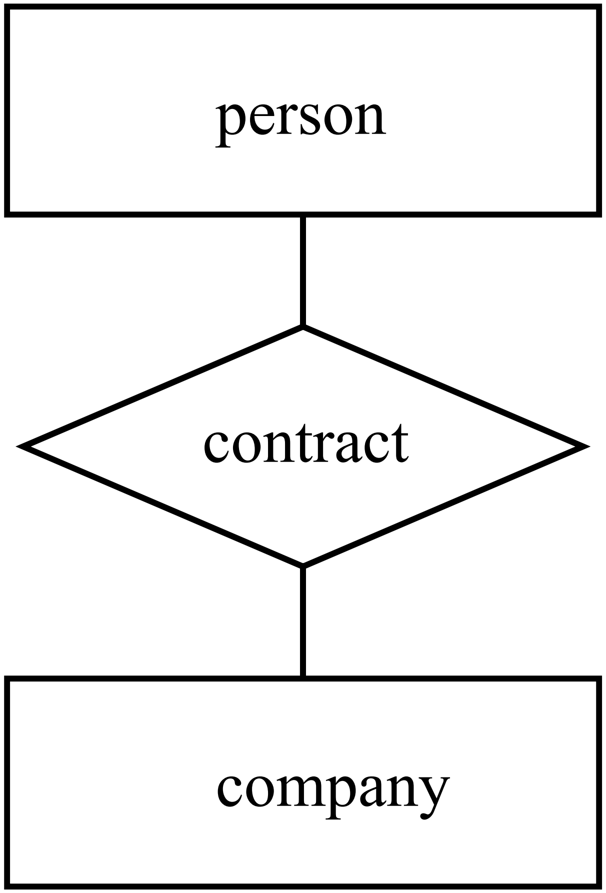
Relationships and Relationship Sets

- Relationships
- A lozenge represents a set of relationships or a relationship set



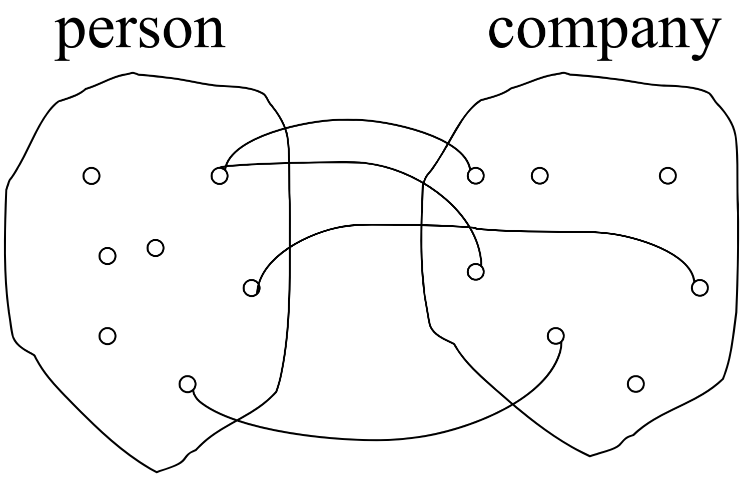
Relationships and Relationship Sets

- A relationship associates 2 or more entities
- A relationship set is a set of relationships associating entities from the same entity sets



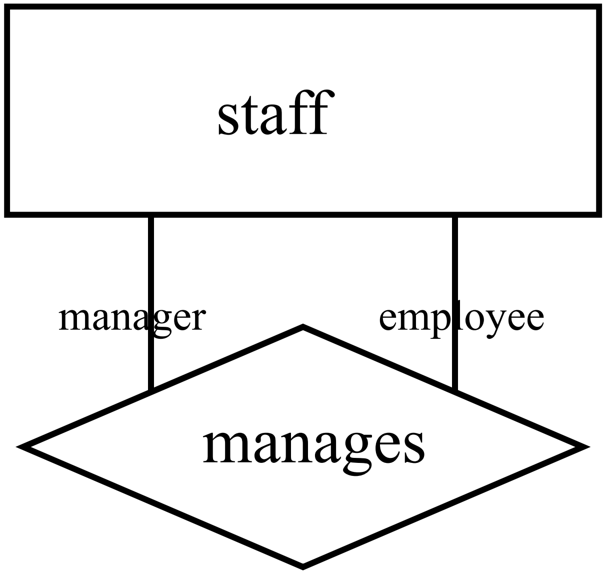
Relationships and Relationship Sets

- A relationship associates 2 or more entities
- A relationship set is a set of relationships associating entities from the same entity sets



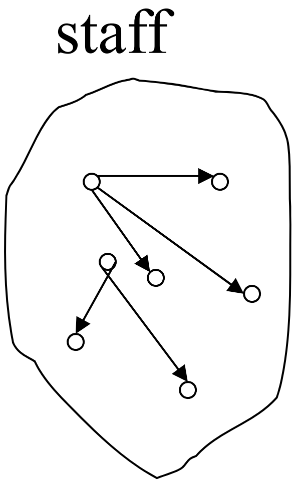
Relationships and Relationship Sets

- Relationships can associate entities from the same entity set
- In this case and in general, participation, or role, in the relationship can be named



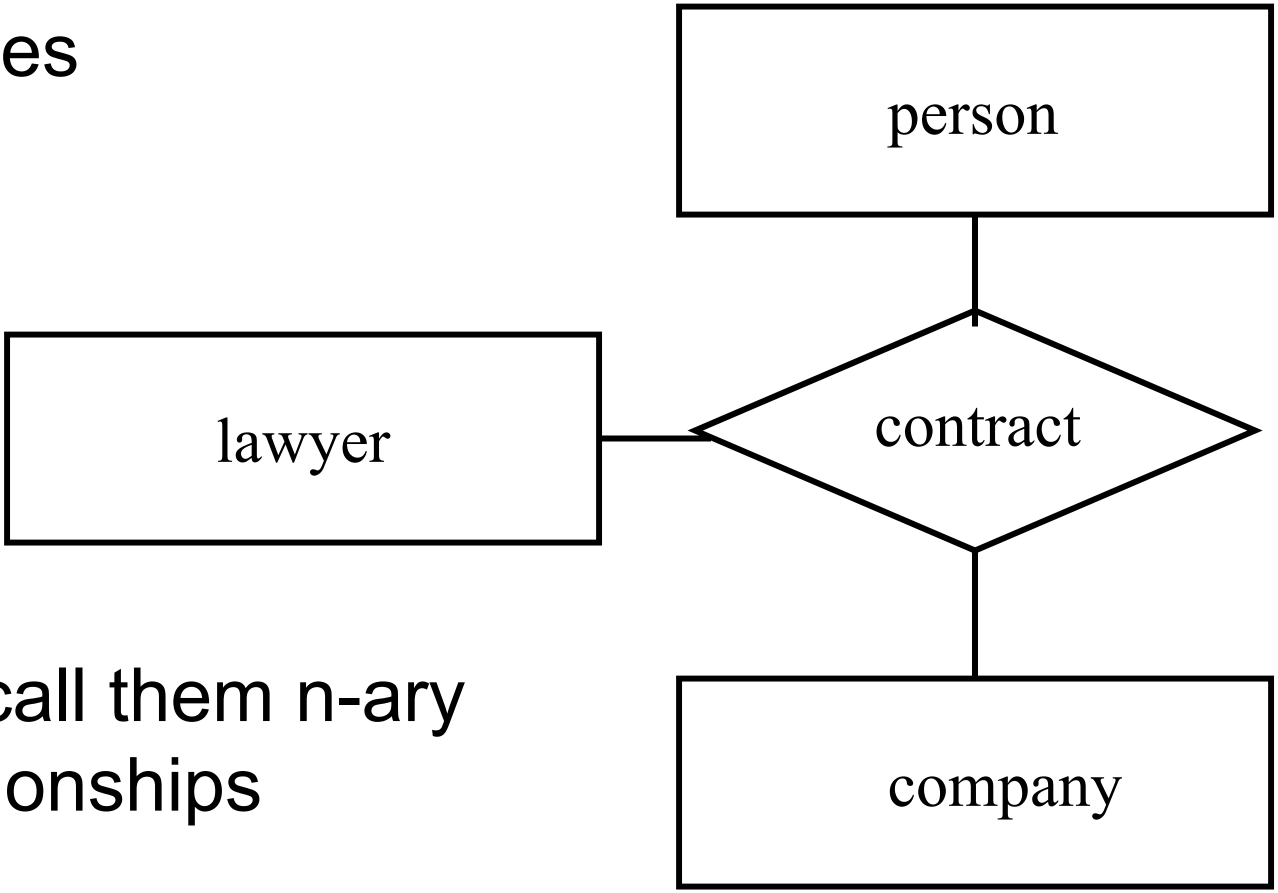
Relationships and Relationship Sets

- Relationships can associate entities from the same entity set
- In this case and in general, participation, or role, in the relationship can be named



Relationships and Relationship Sets

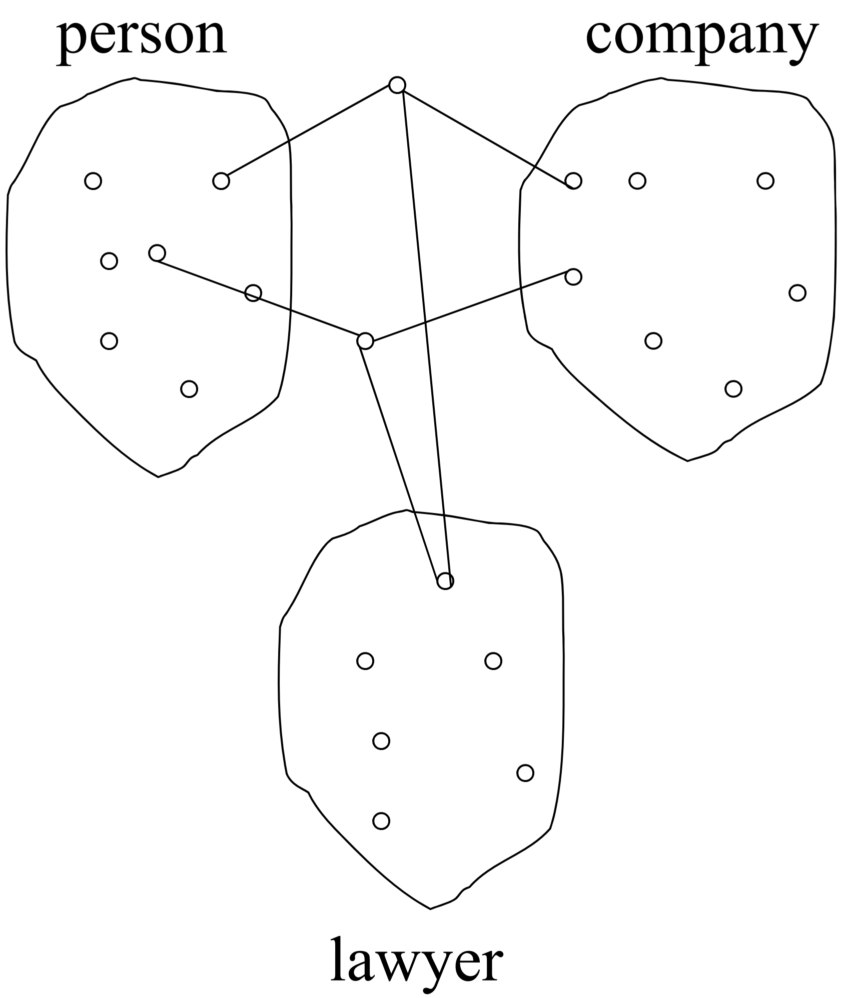
- A relationship can associate more than 2 entities



- We call them n-ary relationships

Relationships and Relationship Sets

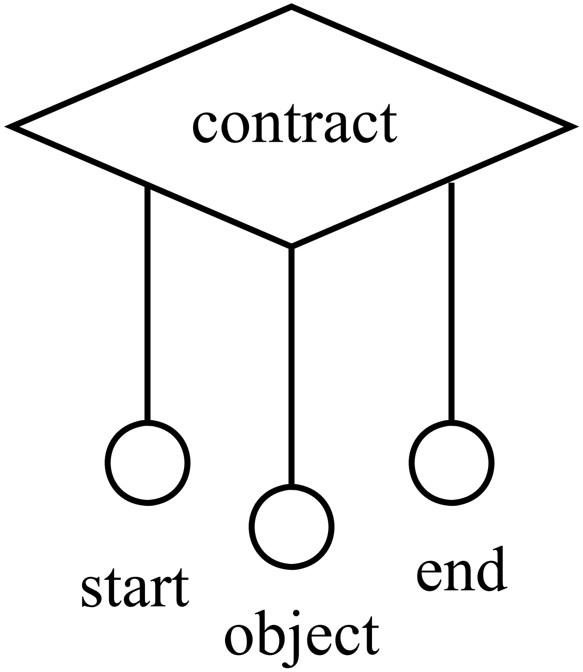
- A relationship can associate more than 2 entities



- We call them n-ary relationships

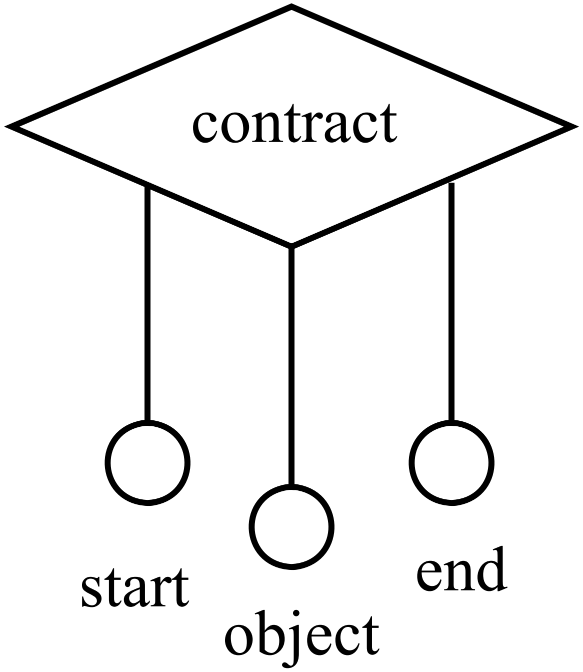
Attributes of Relationships

- Relationship can have attributes
- All relationships in one relationship set have the same attributes

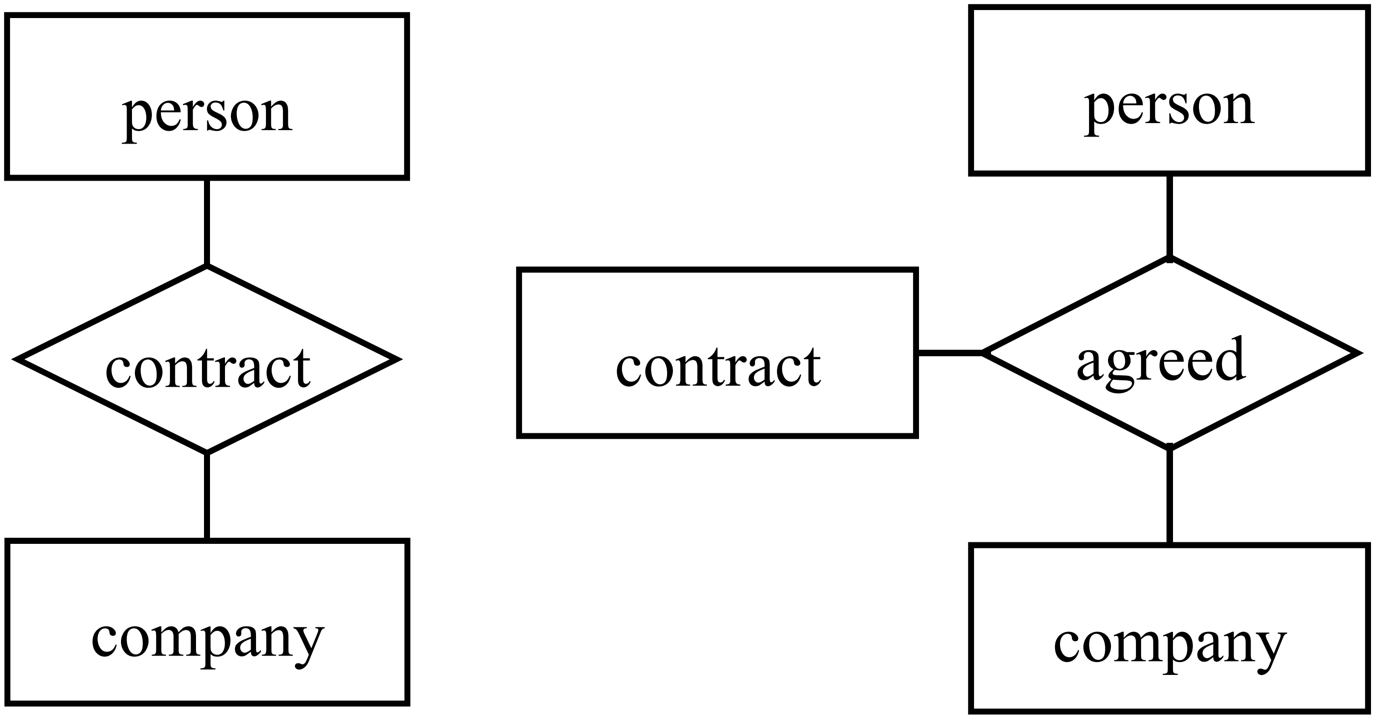


Attributes of Relationships

- Relationships are distinguished not by their attributes but by their participating entities

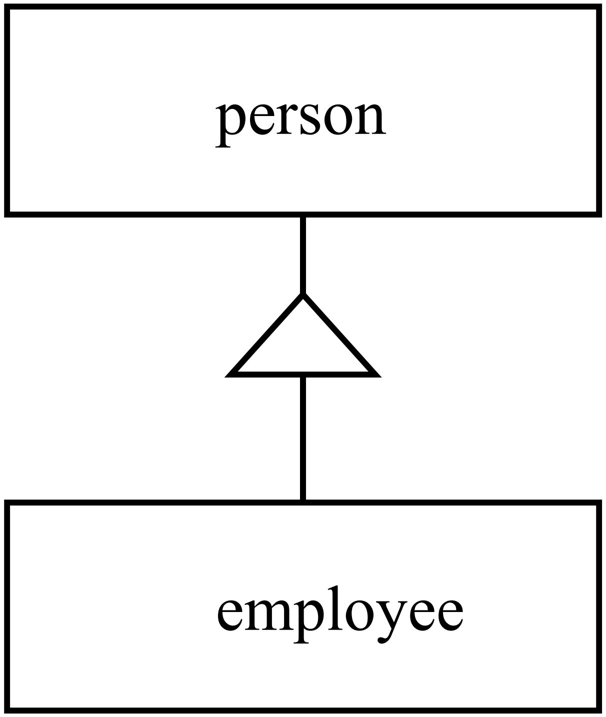


Entity or Relationship?



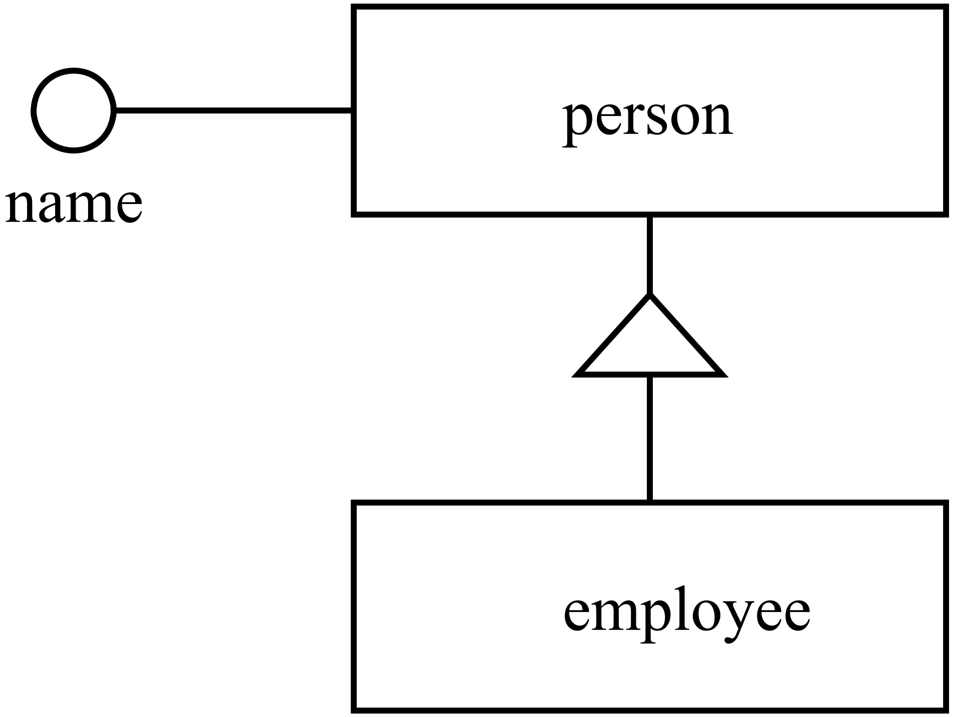
Hierarchies: Subset/Supersets

- It is possible to require that an entity set be a subset of another entity set (its superset)



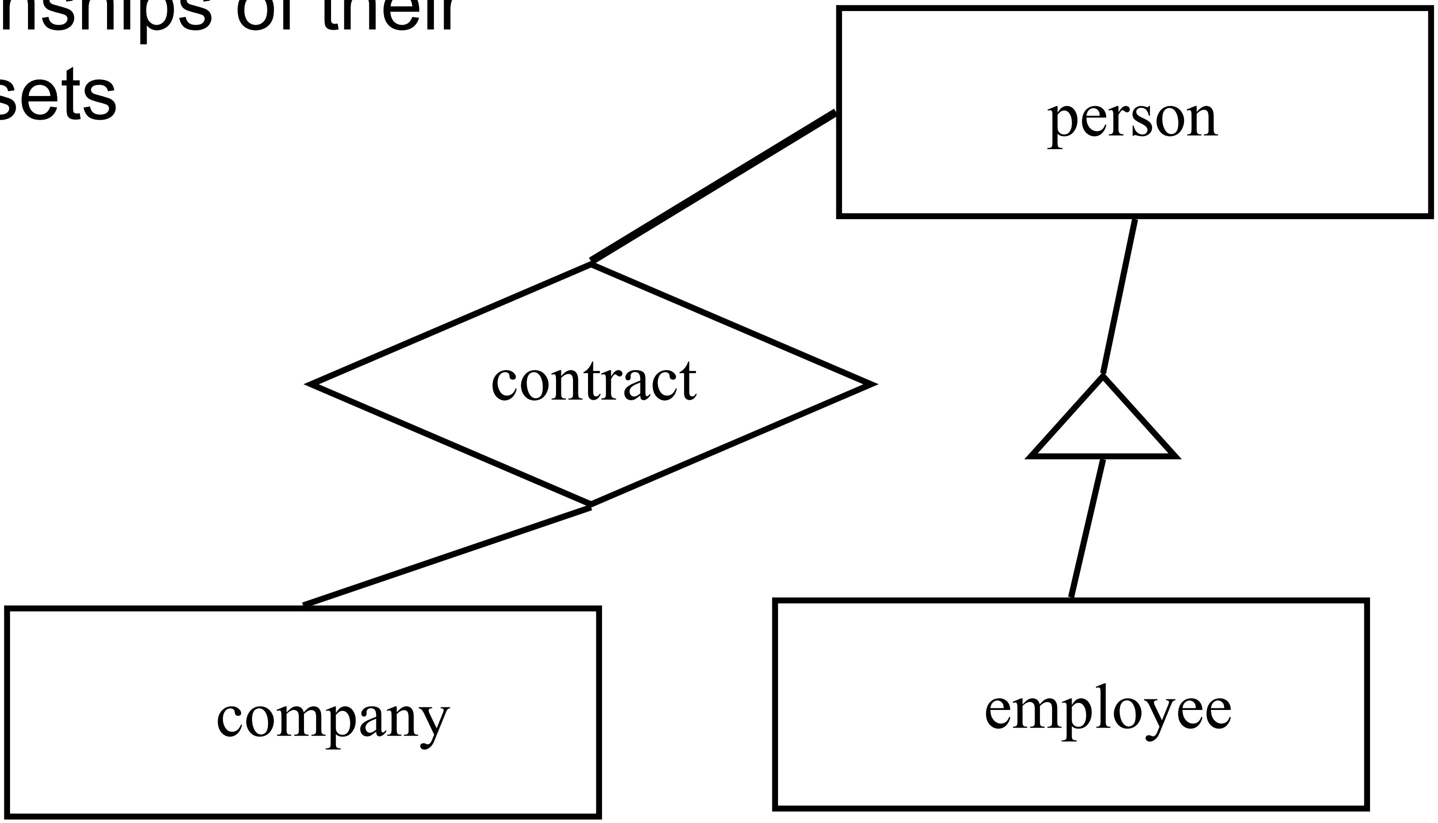
Hierarchies: Inheritance

- Entities in the subsets inherit the attributes of their supersets



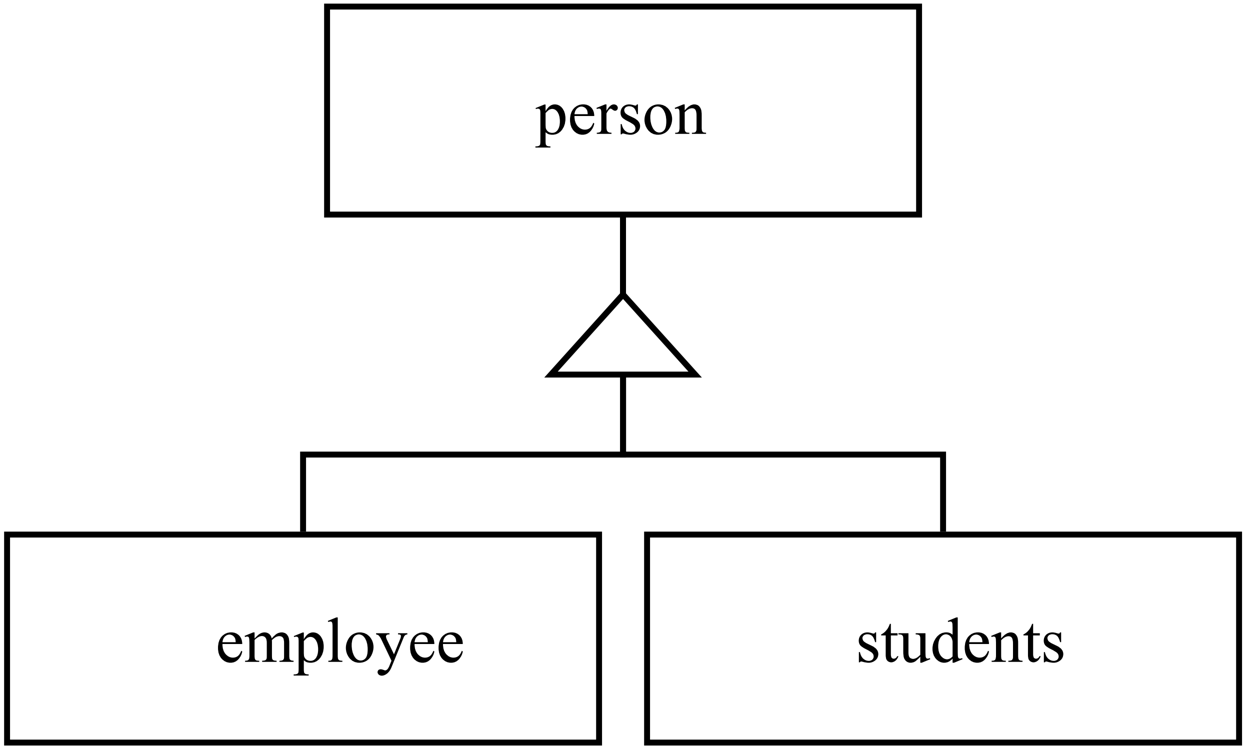
Hierarchies: Inheritance

- Entities in the subsets can participate in the relationships of their supersets



Hierarchies: Generalization/Specialization

- This construction can help design specialization hierarchies

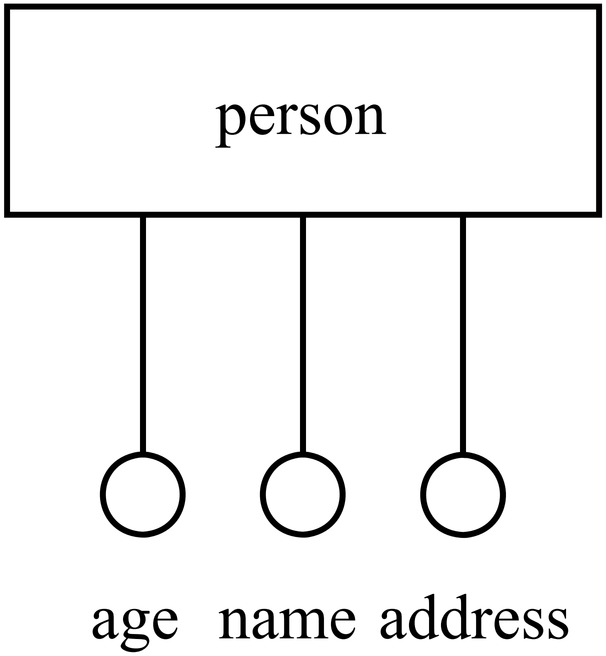


Conceptual Modelling

Integrity

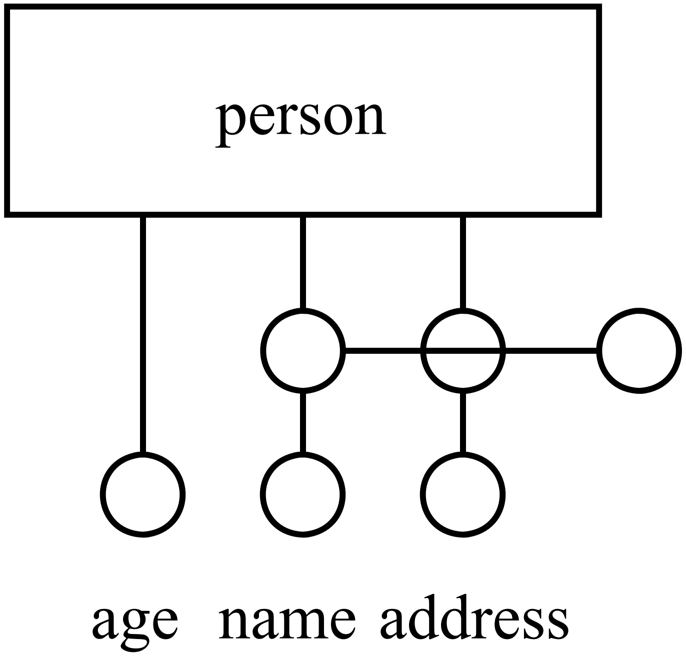
Entities' Identity

- One attribute can identify the entity
- This is a property of all entities in an entity set
- *Notice: at least all attributes identify the entity*



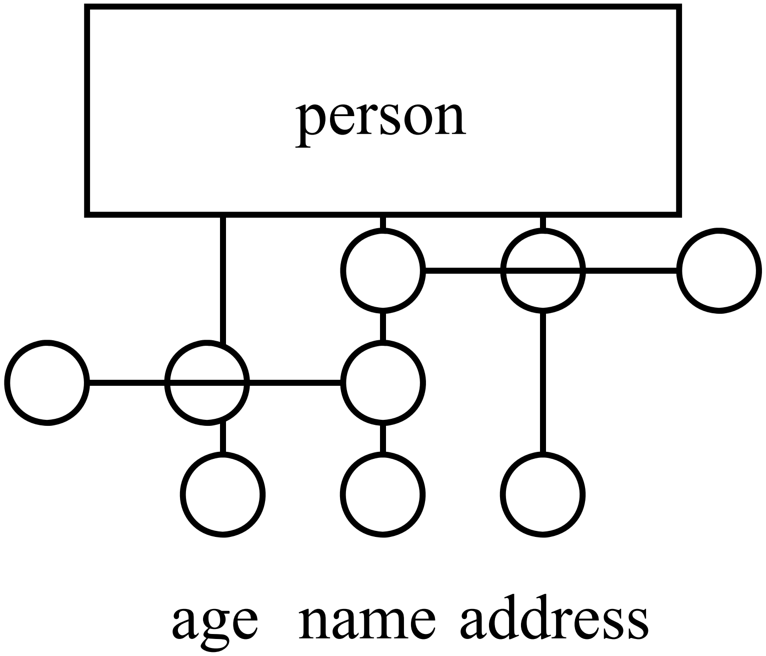
Entities' Identity

- A combination of attributes can identify the entity



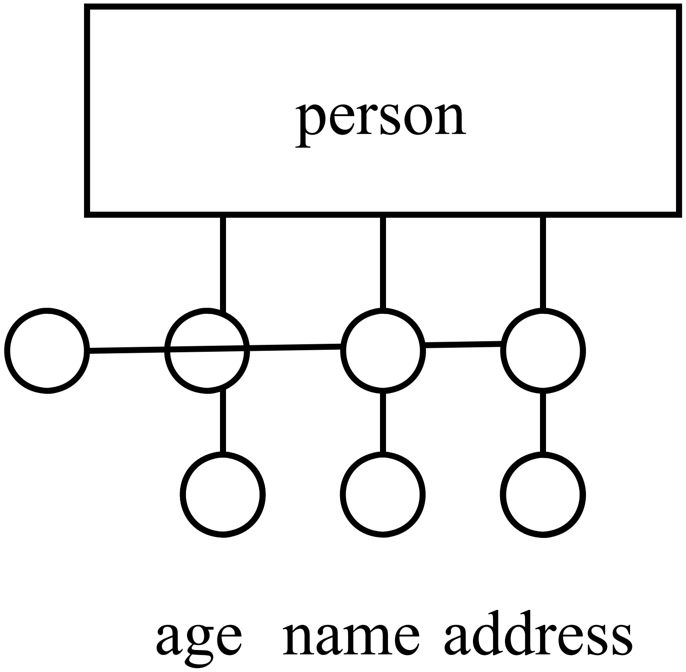
Entities' Identity

- There might be several possible combination of attributes to identify an entity

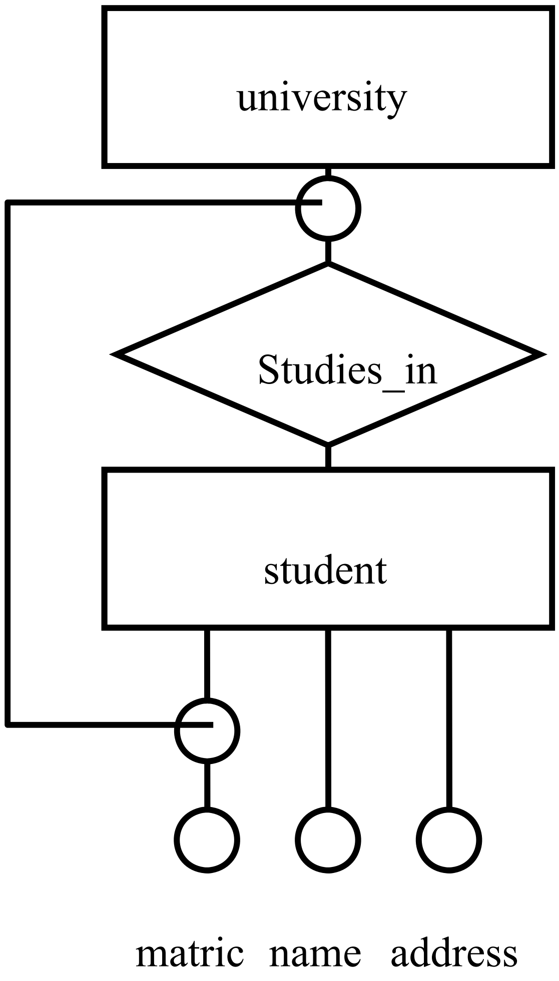


Entities’ Identity

- *Notice: at least all attributes identify the entity*
- *But we might prefer a minimum set of attributes*



Weak Entities

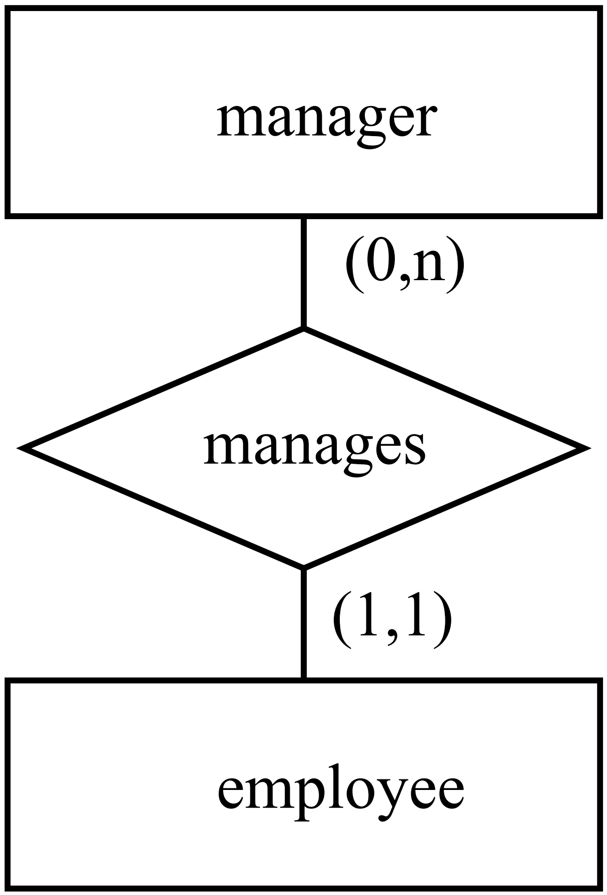


Weak Entities

- Some entities can only be identify within the scope of a relationship with another entity set
- Notice that the relationship must exist and be unique for each entity in the set

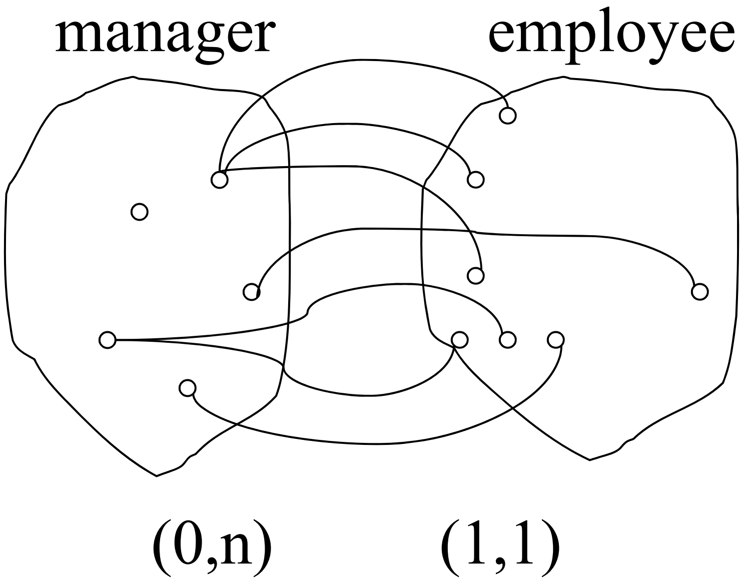
Relationships' Cardinality

- The cardinality of the participation in a relationship can be constrained by a minimum and maximum value:
  (1,1)
  (0, n)
  (2, 5)



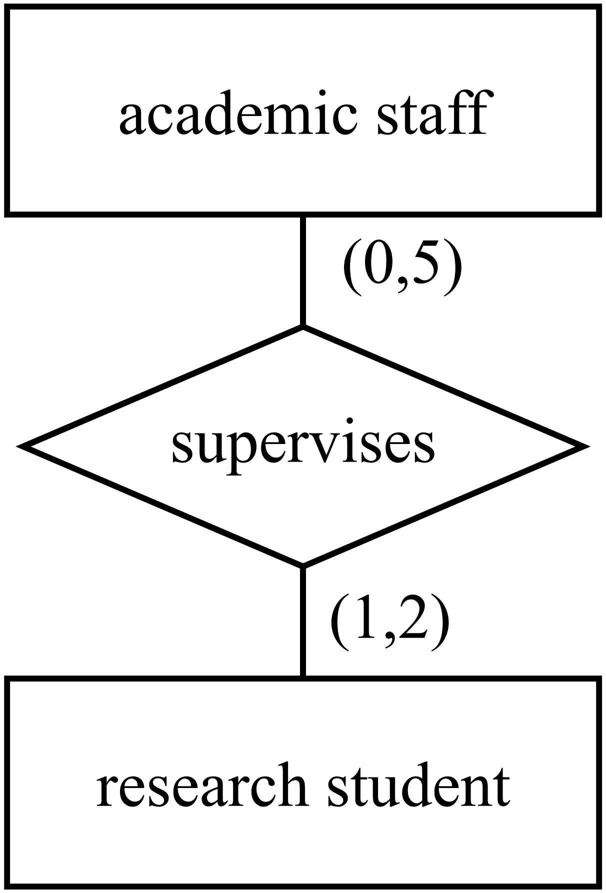
Relationships' Cardinality

- The cardinality of the participation in a relationship can be constrained by a minimum and maximum value:
  - (1,1)
  - (0, n)
  - (2, 5)



Relationships' Cardinality

- Another example



Relationships’ Cardinality

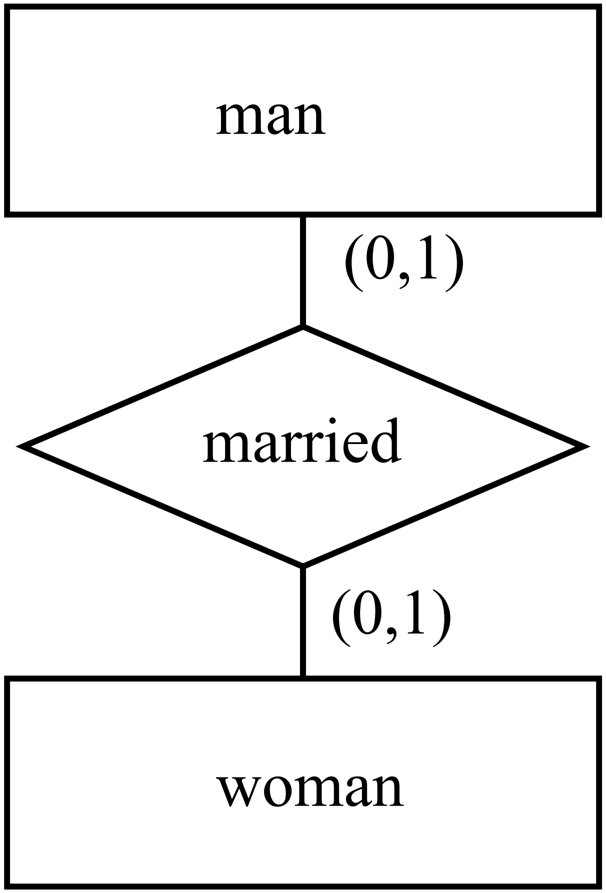
- (1, x) mandatory participation
- (0, x) optional participation

Relationships' Cardinality

- (x, 1) for all entities involved characterizes a one-to-one relationship
- (x, 1) for one entity involved and (x, N) or (x, y) y > 1 for the others characterizes a one-to-many relationship
- (x, N) or (x, y) y > 1 for all entities involved characterizes a many-to-many relationship

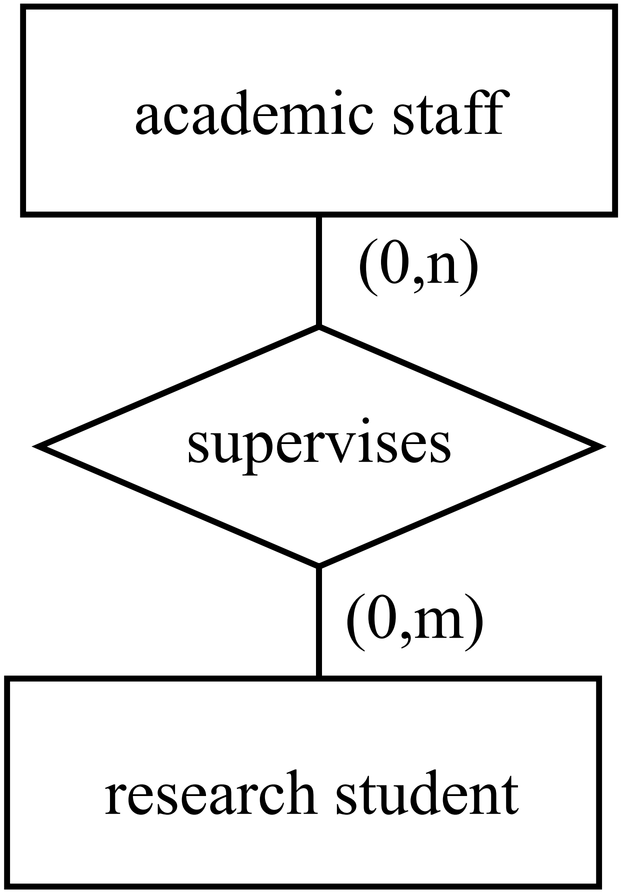
Relationships’ Cardinality

- Example of a one-to-one relationship



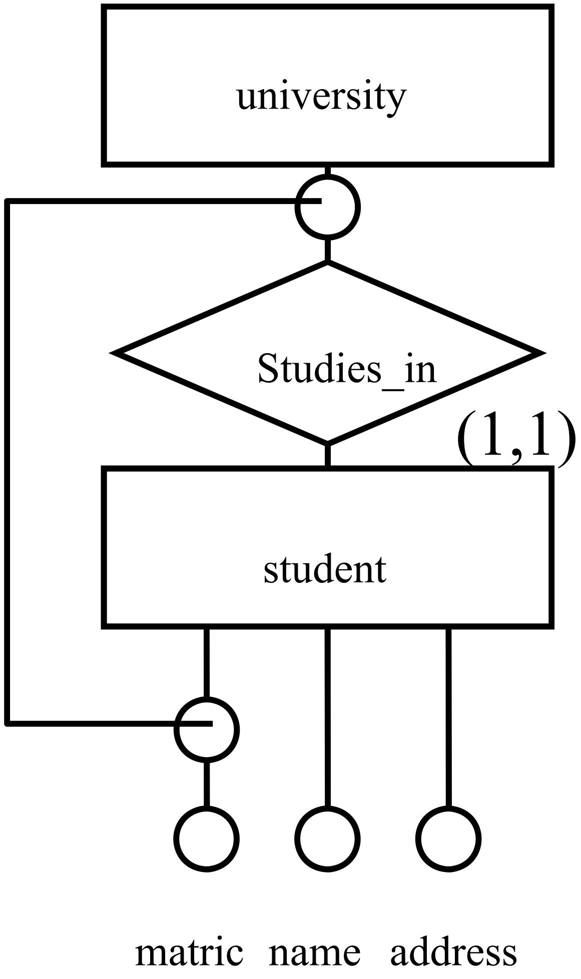
Relationships' Cardinality

- By default we have many-to-many relationships



Weak Entities

- Weak entities can only be define for a participation constrained by (1,1) cardinalities
- Also called mandatory one-to-many relationships

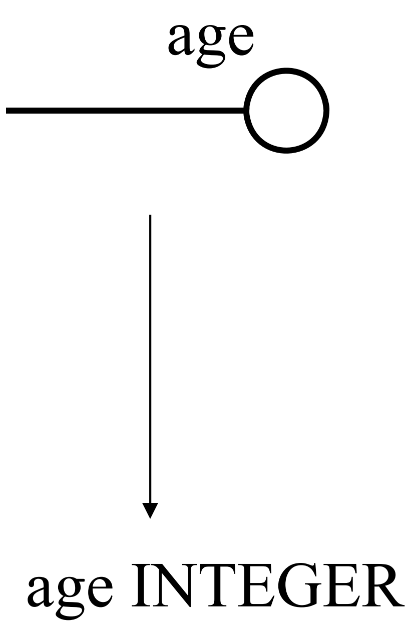


Conceptual to Logical Design

From E-R to Relational
Textbook Section 3.5

Value Sets

- Value sets are mapped to domains
- *In practice this is a first step towards the physical design*
- E-R attributes are mapped to attributes of relations



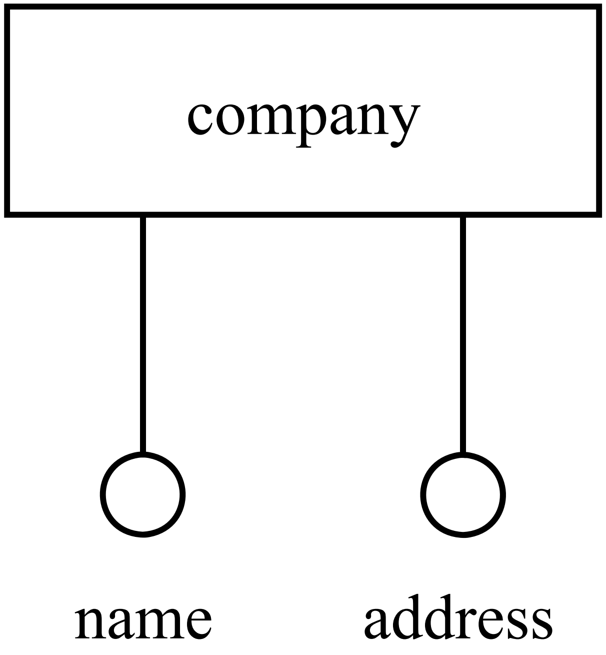
age

age INTEGER

Entity Sets

- Entity sets are mapped to relations
- The entity set attributes are mapped to attributes of the relation
- The keys are mapped to primary key

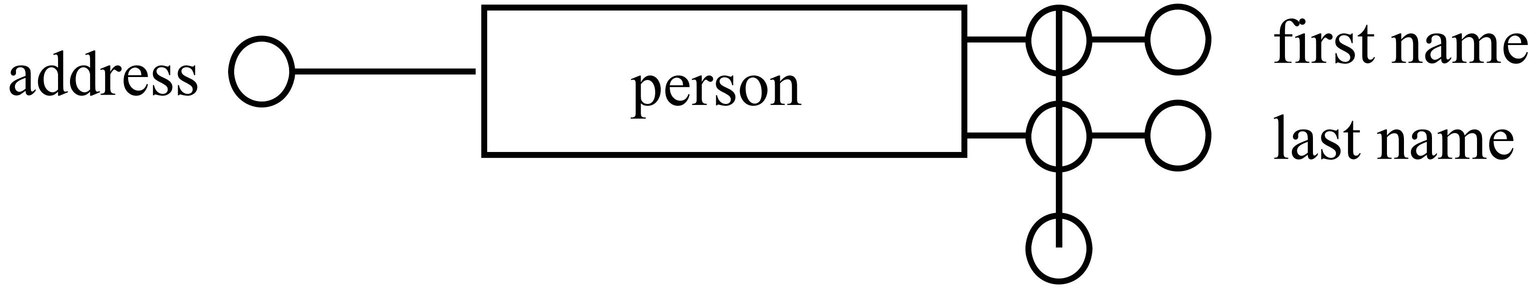
Entity Sets



Entity Sets

```
CREATE TABLE company
   (
   name VARCHAR(64) PRIMARY KEY,
   address VARCHAR(128),
   )
```

Entity Sets



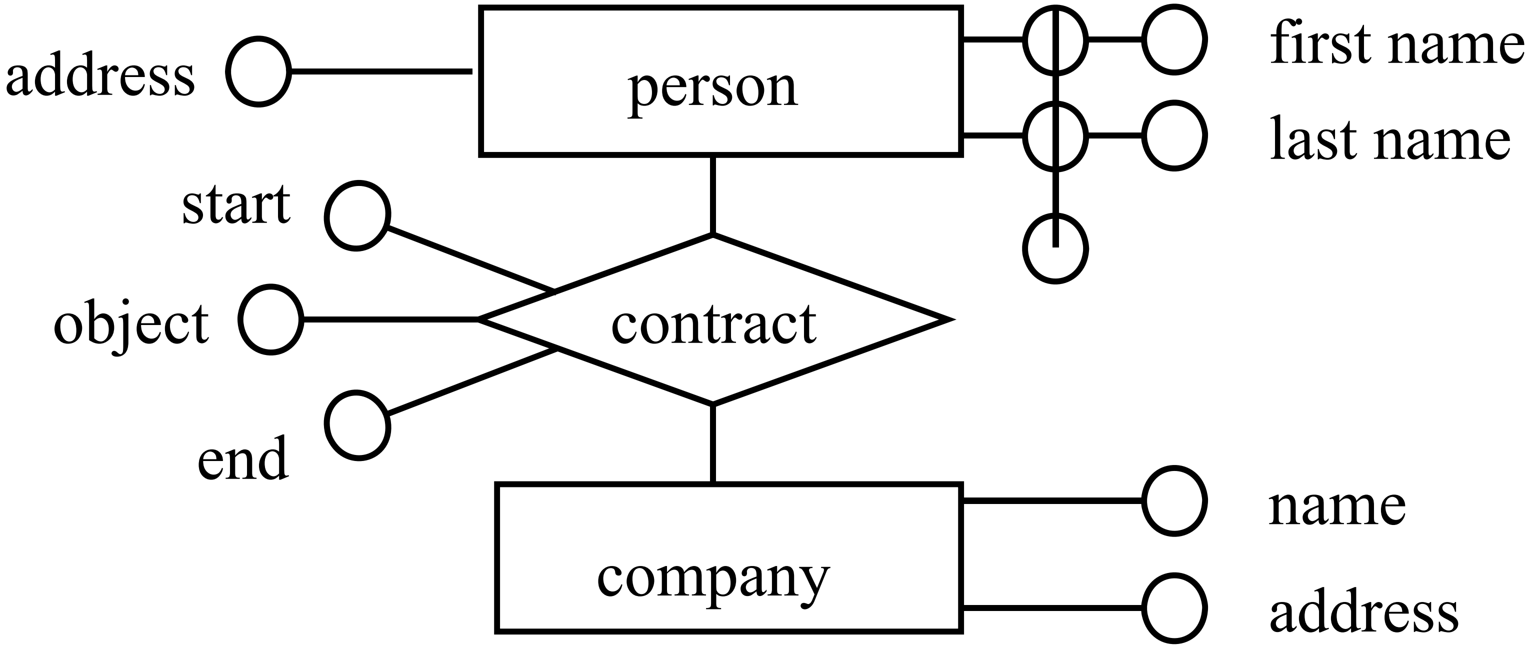
Entity Sets

```
CREATE TABLE person
    (
    first name VARCHAR(32),
    last_name VARCHAR(32),
    address VARCHAR(128),
    PRIMARY KEY (first_name, last_name))
```

Relationship Sets

- Relationship sets are mapped to relations
- The attributes of the relation consist of the attributes of the relationship set
- As well as of the keys of the participating entities

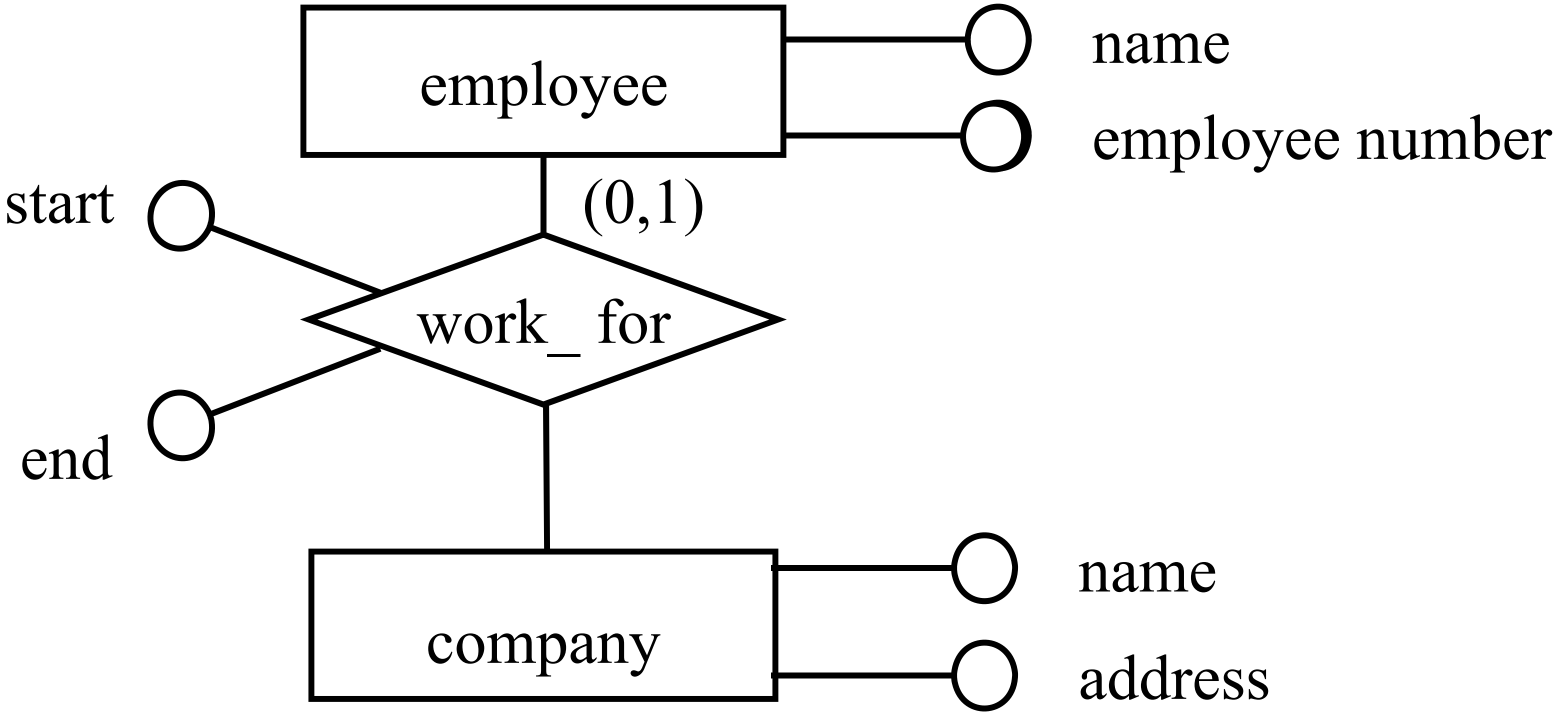
Relationship Sets



Relationship Sets

```
CREATE TABLE contract
    (
    start DATE,
    end DATE,
    object VARCHAR(128),
    pfirst_name VARCHAR(32),
    plast_name VARCHAR(32),
    cname VARCHAR(64),
    PRIMARY KEY (pfirst_name, plast_name, cname),
    FOREIGN KEY (pfirst_name , plast_name ) REFERENCES
    person(first_name, last_name),
    FOREIGN KEY (cname ) REFERENCES company(name)
    )
```

Key Constraints (one-to-many relationships)



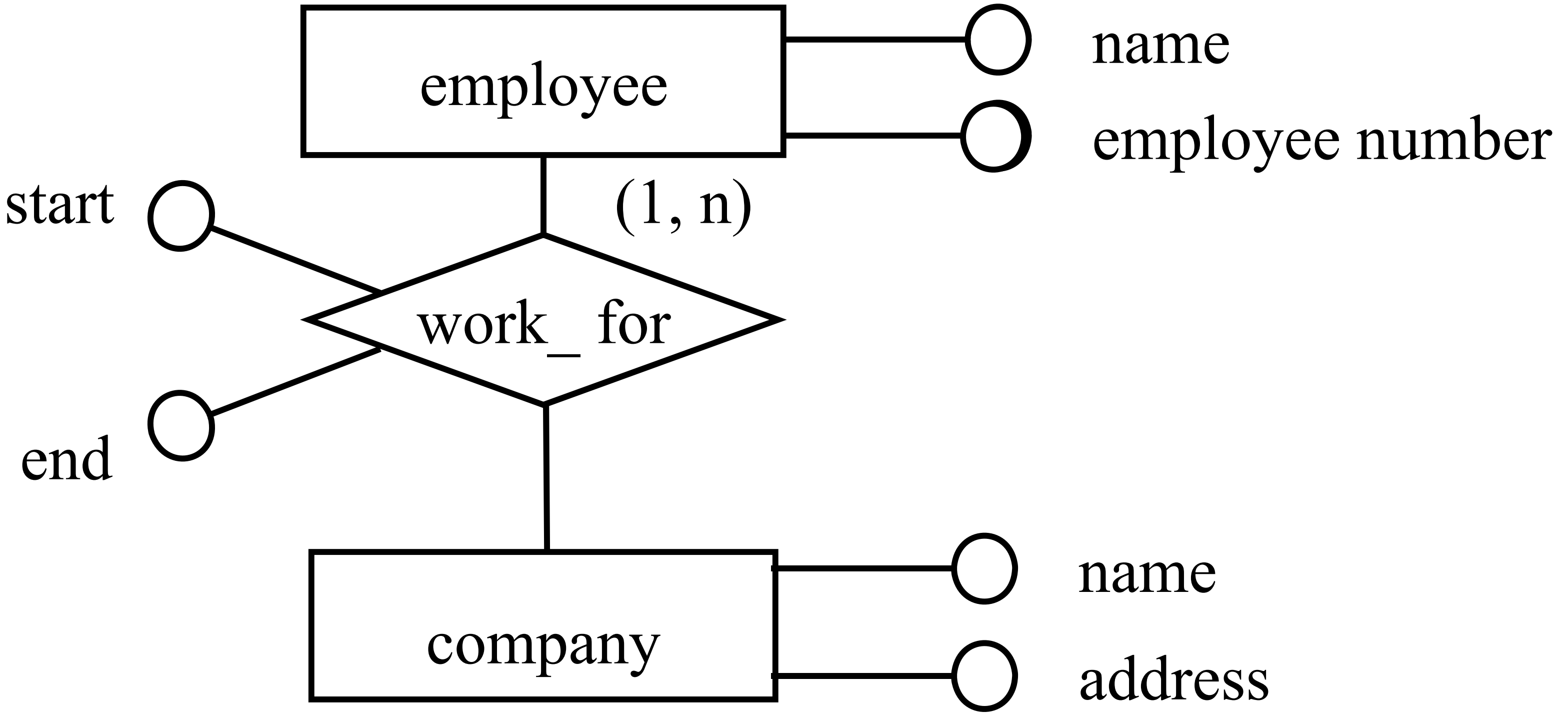
Key Constraints (one-to-many relationships)

```
CREATE TABLE work_for
   (
   start DATE,
   end DATE,
   enumber CHAR(8),
   cname VARCHAR(32),
   PRIMARY KEY (enumber, cname),
   FOREIGN KEY (enumber) REFERENCES
   employee(number),
   FOREIGN KEY (cname) REFERENCES
   company(name)
   )
```

Key Constraints (one-to-many relationships)

```
CREATE TABLE work_for
   (
   start DATE,
   end DATE,
   enumber CHAR(8) PRIMARY KEY,
   cname VARCHAR(32),
   FOREIGN KEY (enumber) REFERENCES
   employee(number),
   FOREIGN KEY (cname) REFERENCES
   company(name)
   )
```

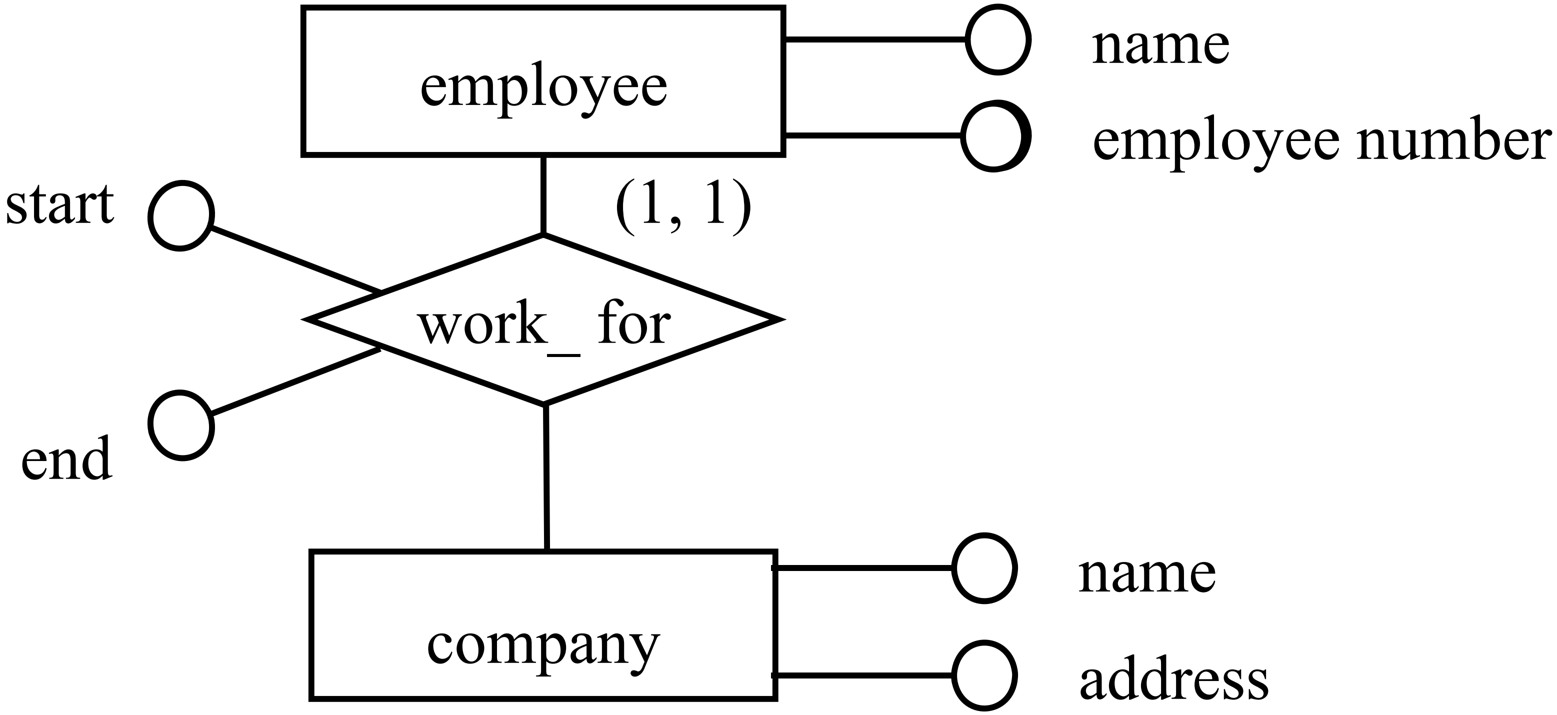
Participation Constraints



Participation Constraints

```
CREATE TABLE work_for
   (
   start DATE,
   end DATE,
   enumber CHAR(8),
   cname VARCHAR(32),
   PRIMARY KEY (enumber, cname),
   FOREIGN KEY (enumber) REFERENCES
   employee(number),
   FOREIGN KEY (cname) REFERENCES
   company(name)
   )
```

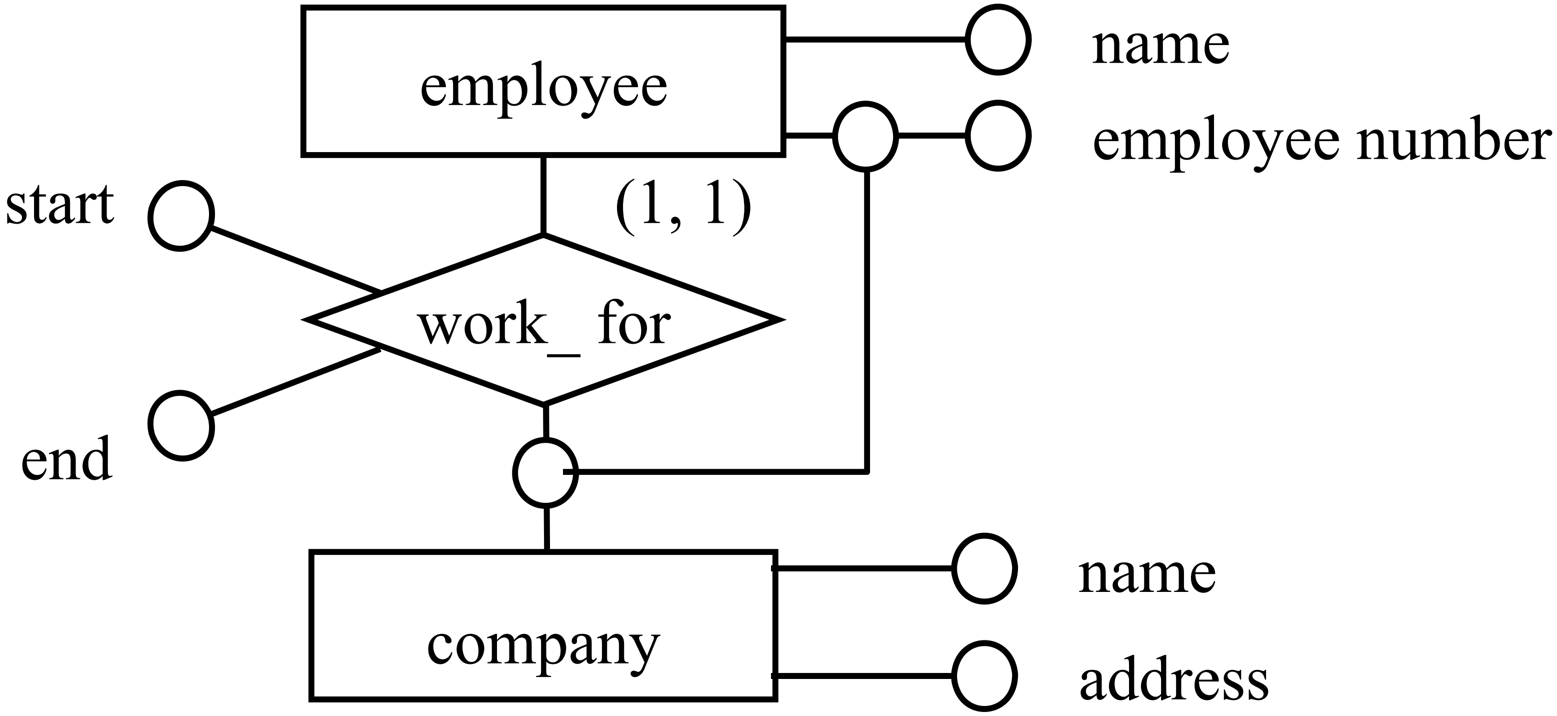
Key and Participation Constraints



Key and Participation Constraints

```
CREATE TABLE employee_work_for
   (
   start DATE,
   end DATE,
   enumber CHAR(8) PRIMARY KEY,
   ename CHAR(32),
   cname VARCHAR(32),
   FOREIGN KEY (cname) REFERENCES
   company(name)
   )
```

Weak Entity Sets



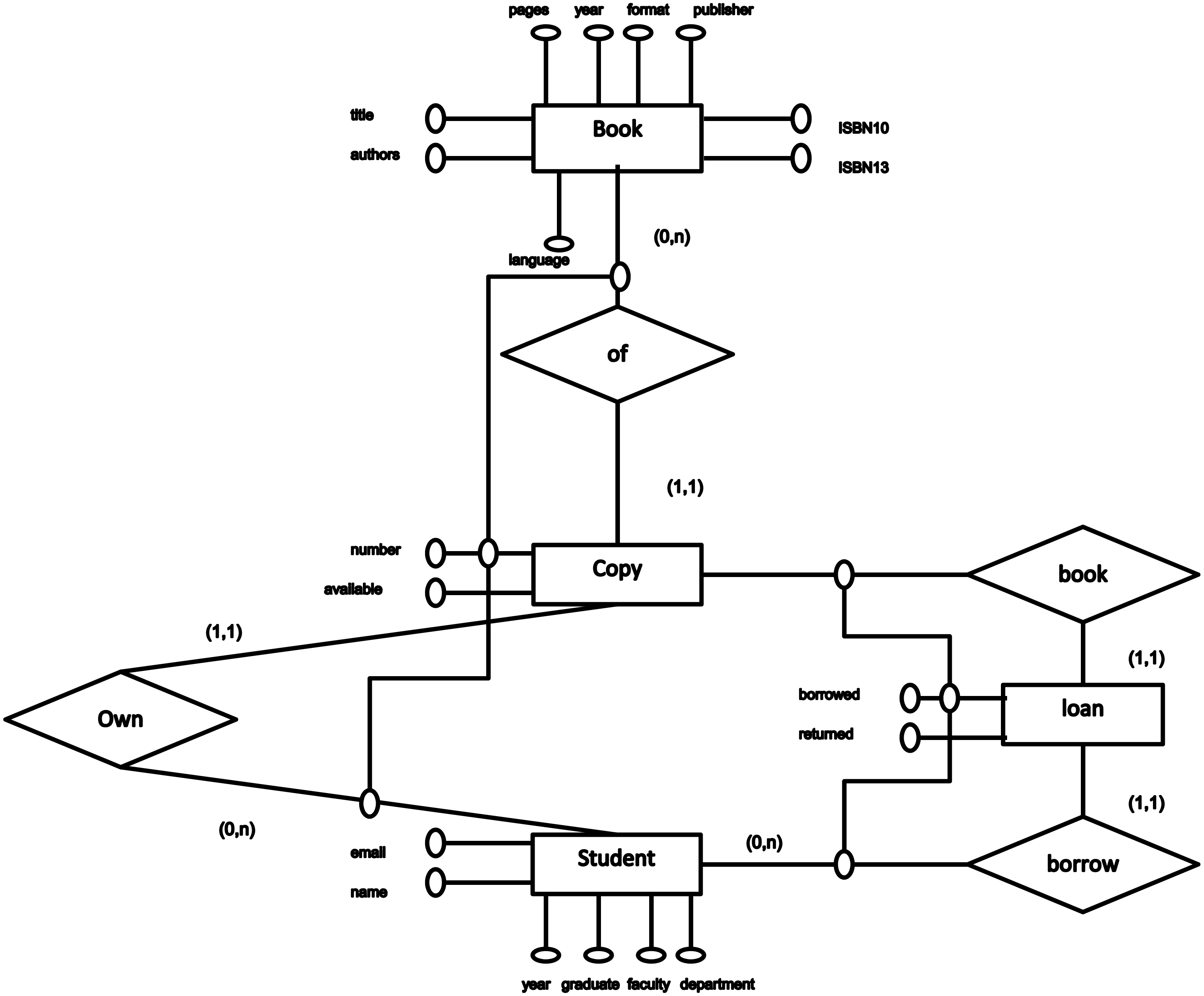
Weak Entity Sets

```
CREATE TABLE employee_work_for
    (
    start DATE,
    end DATE,
    enumber CHAR(8) PRIMARY KEY,
    ename CHAR(32),
    cname VARCHAR(32),
    FOREIGN KEY (cname) REFERENCES
    company(name)
    )
```

Weak Entity Sets

```
CREATE TABLE employee_work_for
   (
   start DATE,
   end DATE,
   enumber CHAR(8),
   ename CHAR(32),
   cname VARCHAR(32),
   PRIMARY KEY (enumber, cname),
   FOREIGN KEY (cname) REFERENCES
   company(name)
   )
```

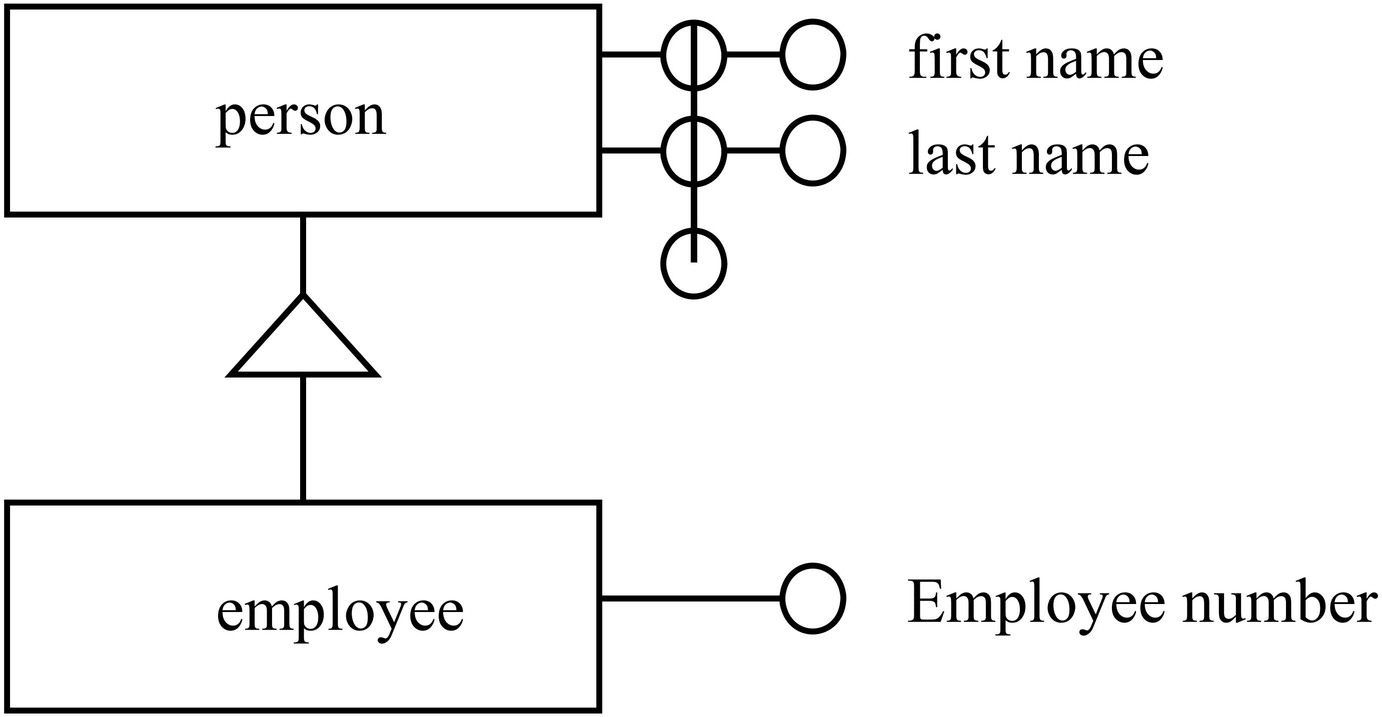
Entity-relationship Diagram from theTutorial



Hierarchies

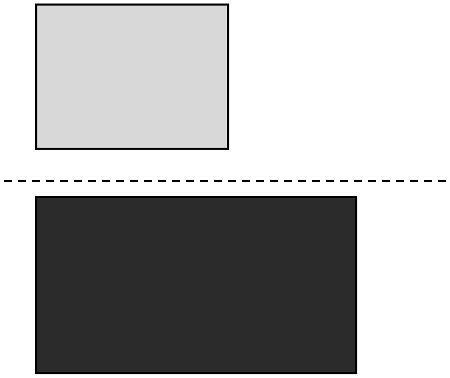
- Hierarchies can be mapped to relations in different ways, e.g.:
  - Horizontal mapping
  - Vertical mapping

Hierarchies



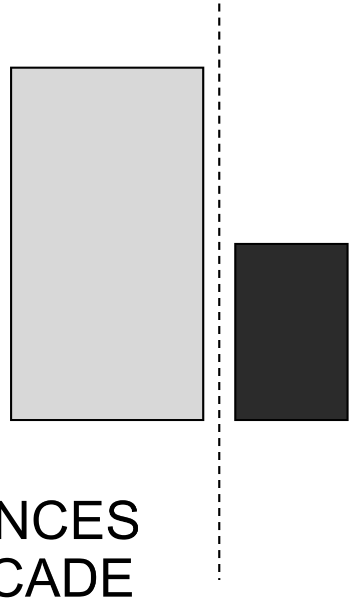
Hierarchies: Horizontal Mapping

```
CREATE TABLE person
    (
    first name VARCHAR(32),
    last_name VARCHAR(32),
    address VARCHAR(128),
    PRIMARY KEY (first_name, last_name)
    )
CREATE TABLE employee
    (
    first name VARCHAR(32),
    last_name VARCHAR(32),
    address VARCHAR(128),
    employee_number INTEGER PRIMARY KEY
    )
```



Hierarchies: Vertical Mapping

```
CREATE TABLE person
    (
    first name VARCHAR(32),
    last_name VARCHAR(32),
    address VARCHAR(128),
    PRIMARY KEY (first_name, last_name)
    )
CREATE TABLE employee
    (
    first name VARCHAR(32),
    last_name VARCHAR(32),
    employee_integer PRIMARY KEY,
    FOREIGN KEY (first_name, last_name) REFERENCES
    person(first_name, last_name) ON DELETE CASCADE
    )
```



Credits

The content of this lecture is based on chapter 7 of the book “Introduction to database Systems”
By
S. Bressan and B. Catania,
McGraw Hill publisher

Clipart and media are licensed from Microsoft Office Online Clipart and Media