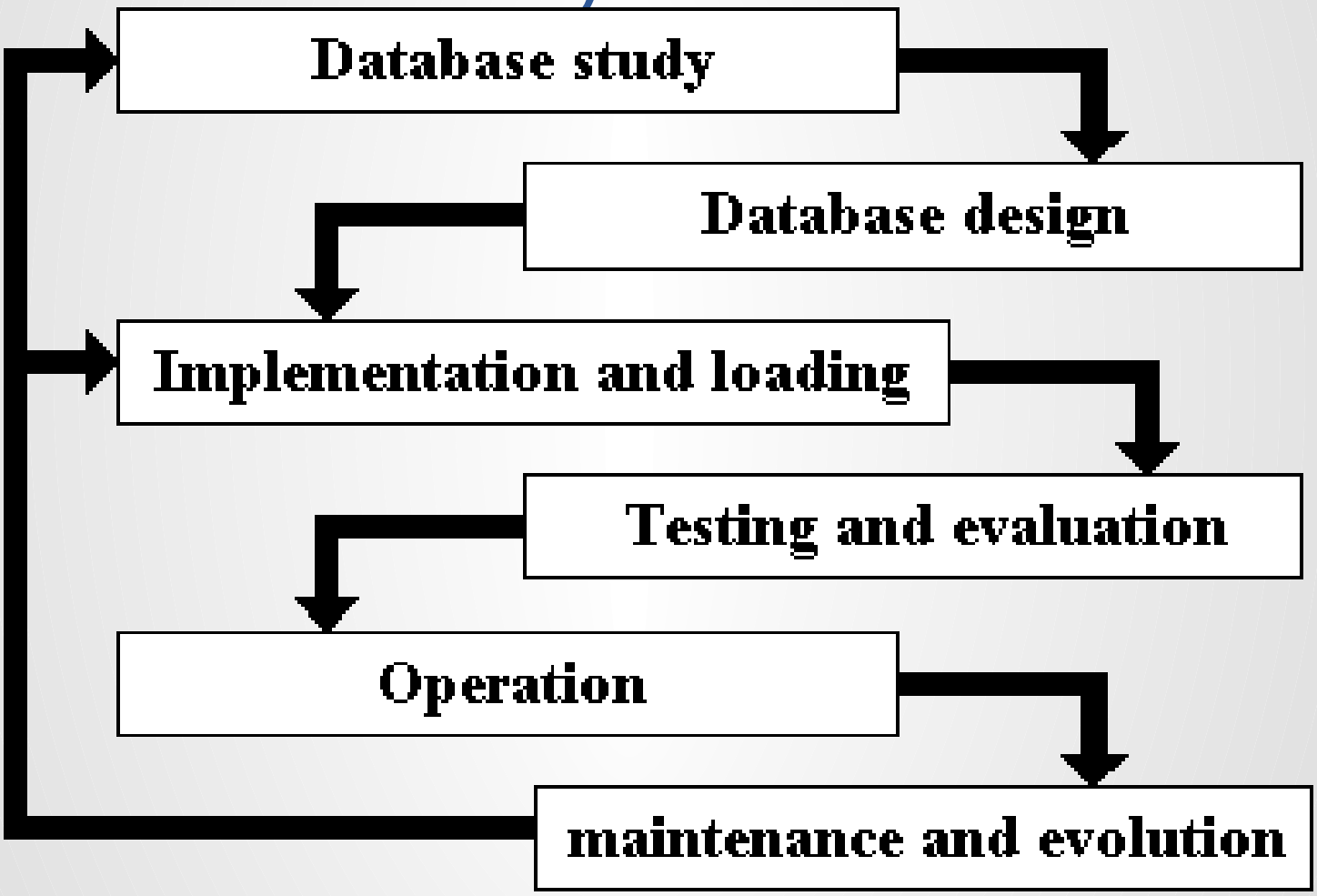
E R diagram

Database Analysis Life Cycle

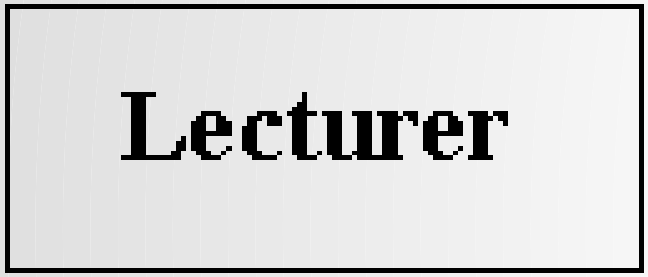


Basics

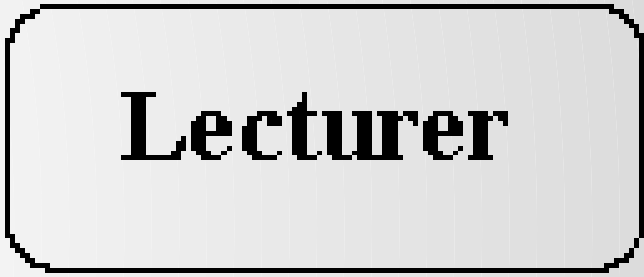
Entity Relationship (ER) modelling

- is a design tool
- is a graphical representation of the database system
- provides a high-level conceptual data model
- supports the user's perception of the data
- is DBMS and hardware independent
- had many variants
- is composed of entities, attributes, and relationships

Entities



Chen's notation



other notations

Entities

- An entity is any object in the system that we want to model and store information about
- Individual objects are called entities
- Groups of the same type of objects are called entity types or entity sets
- Entities are represented by rectangles (either with round or square corners)

Entity types

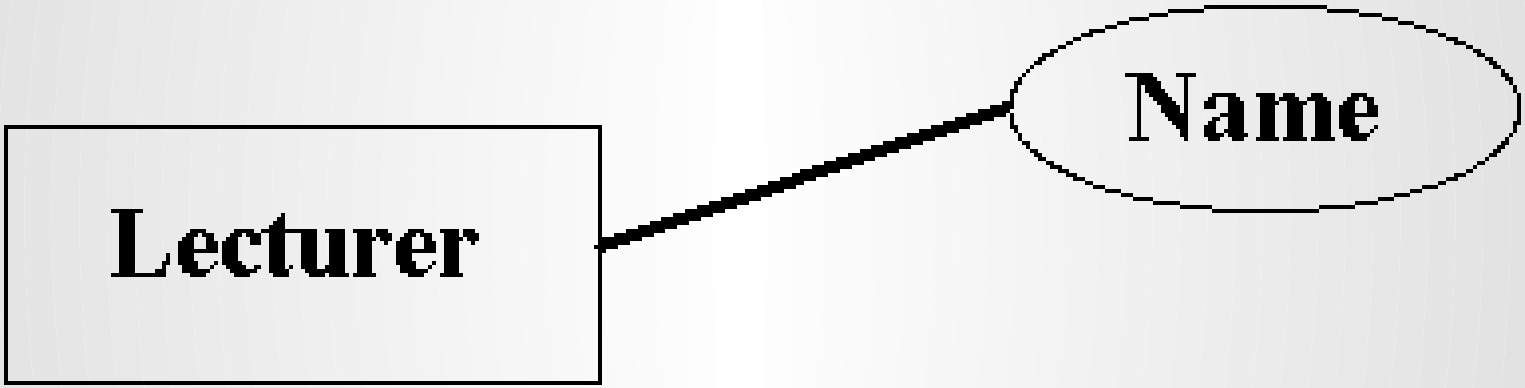
There two types of entities;

1. weak entity

2.strong entity

Attribute

- All the data relating to an entity is held in its attributes.
- An attribute is a property of an entity.
- Each attribute can have any value from its domain.
- Each entity within an entity type:
  - May have any number of attributes.
  - Can have different attribute values than that in any other entity.
  - Have the same number of attributes.
- Attributes can be
  - simple or composite
  - single-valued or multi-valued
- Attributes can be shown on ER models
- They appear inside ovals and are attached to their entity.


Lecturer
Name

Keys

- **A key is a data item that allows us to uniquely identify individual occurrences or an entity type.**
- **A candidate key is an attribute or set of attributes that uniquely identifies individual occurrences or an entity type.**
- **An entity type may have one or more possible candidate keys, the one which is selected is known as the primary key.**
- **A composite key is a candidate key that consists of two or more attributes**
- **The name of each primary key attribute is underlined.**

Relationships

- **A *relationship type* is a meaningful association between entity types**
- **A *relationship* is an association of entities where the association includes one entity from each participating entity type.**
- **Relationship types are represented on the ER diagram by a series of lines.**
- **In the original Chen notation, the relationship is placed inside a diamond, e.g. managers manage employee**

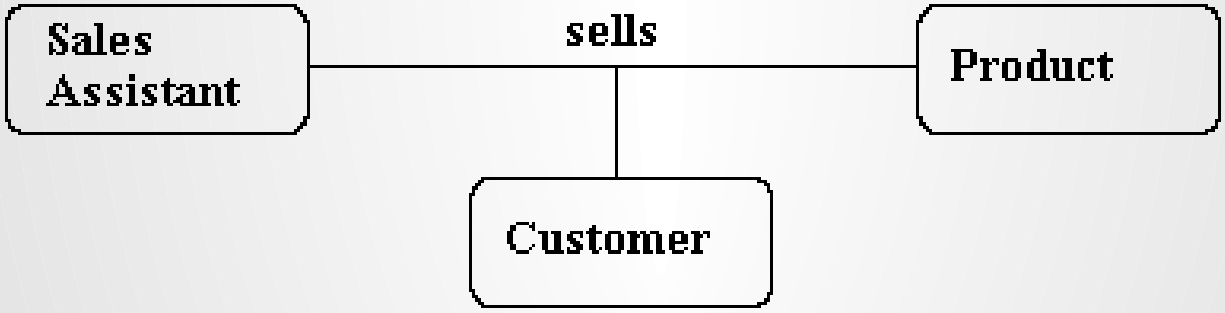

Manager
manages
Employee


Manager
manages
Employee

Degree of a Relationship

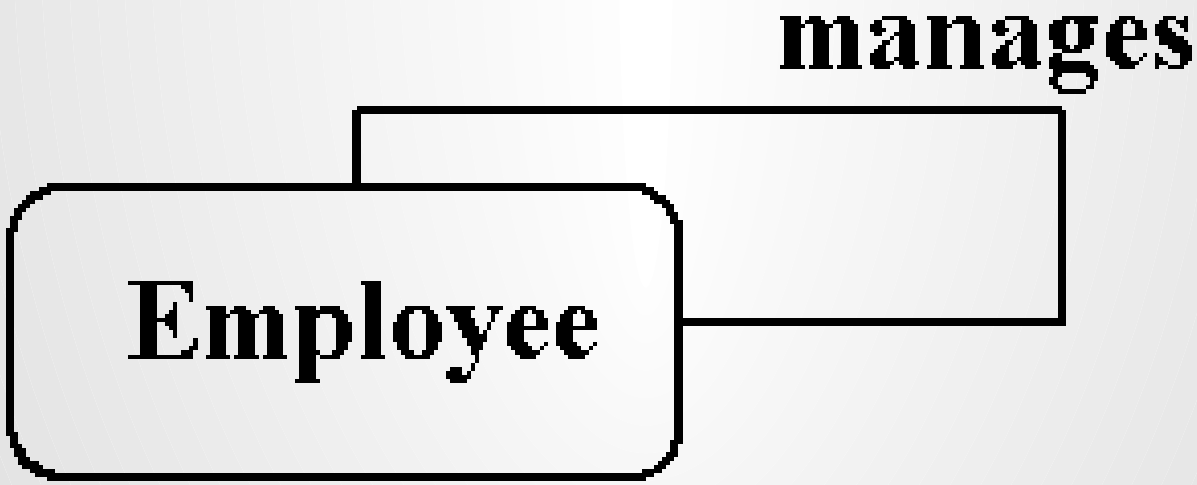
- The number of participating entities in a relationship is known as the **degree of the relationship.**
- If there are two entity types involved it is a ***binary* relationship type**

If there are **three entity types** involved it is a ***ternary* relationship type**



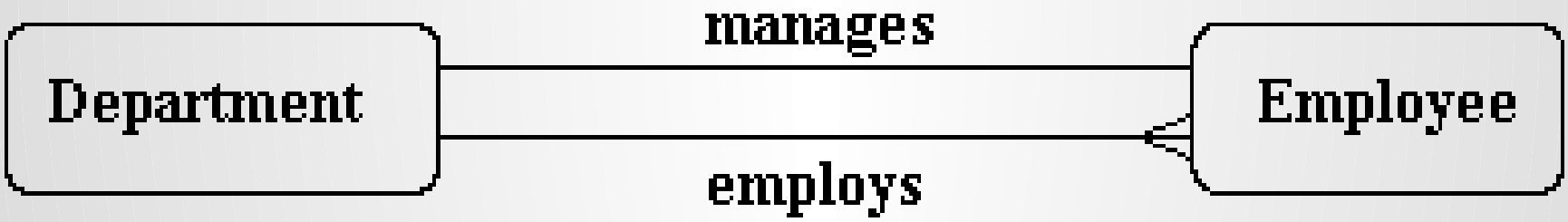
Recursive relationship

- It is possible to have a n-ary relationship (e.g. quaternary or unary).
- **Unary relationships are also known as a *recursive* relationship.**



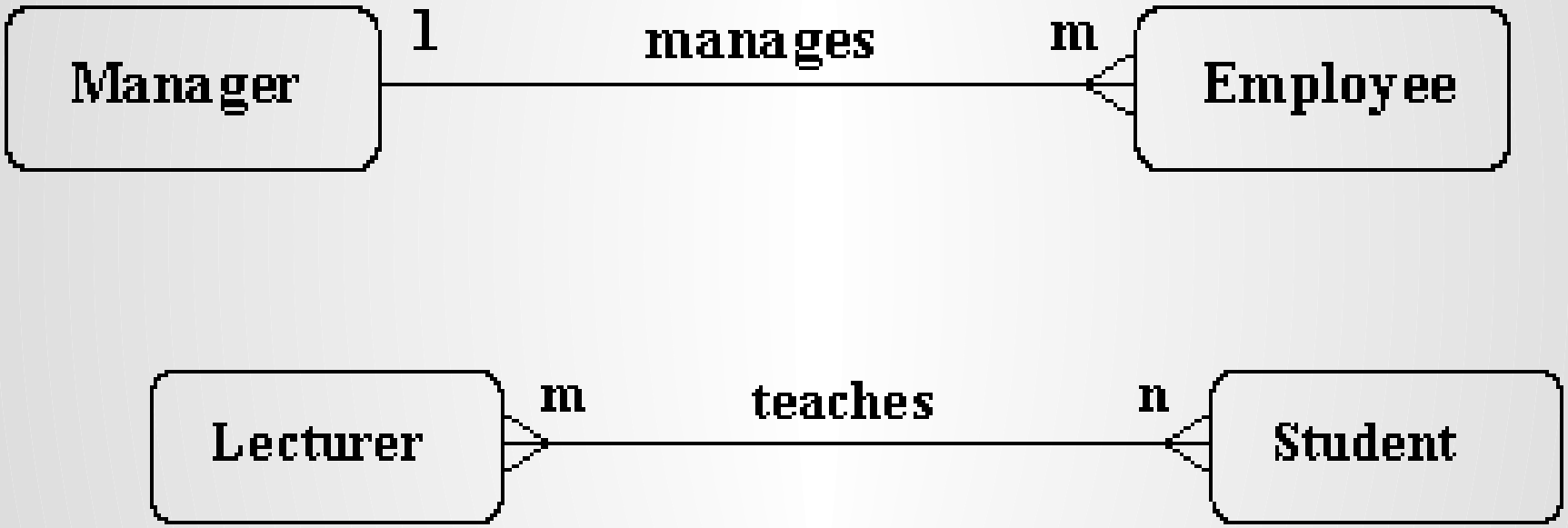
Degree of a Relationship

- It is also possible to have entities associated through two or more distinct relationships.



Cardinality

- Relationships are rarely one-to-one
- For example, a manager usually manages more than one employee
- This is described by the *cardinality* of the relationship, for which there are four possible categories.
  - **One to one (1:1) relationship**
  - **One to many (1:m) relationship**
  - **Many to one (m:1) relationship**
  - **Many to many (m:n) relationship**
- On an ER diagram, if the end of a relationship is straight, it represents 1, while a "crow's foot" end represents many.


Manager
1
manages
m
Employee
Lecturer
m
teaches
n
Student

Country Bus Company

A Country Bus Company owns a number of busses.

- Each bus is allocated to a particular route, although some routes
- may have several busses.
- Each route passes through a number of towns.
- One or more drivers are allocated to each stage of a route, which corresponds to a journey through some or all of the towns on a route.
- Some of the towns have a garage where busses are kept and each of the busses are identified by the registration number and can carry different numbers of passengers, since the vehicles vary in size and can be single or

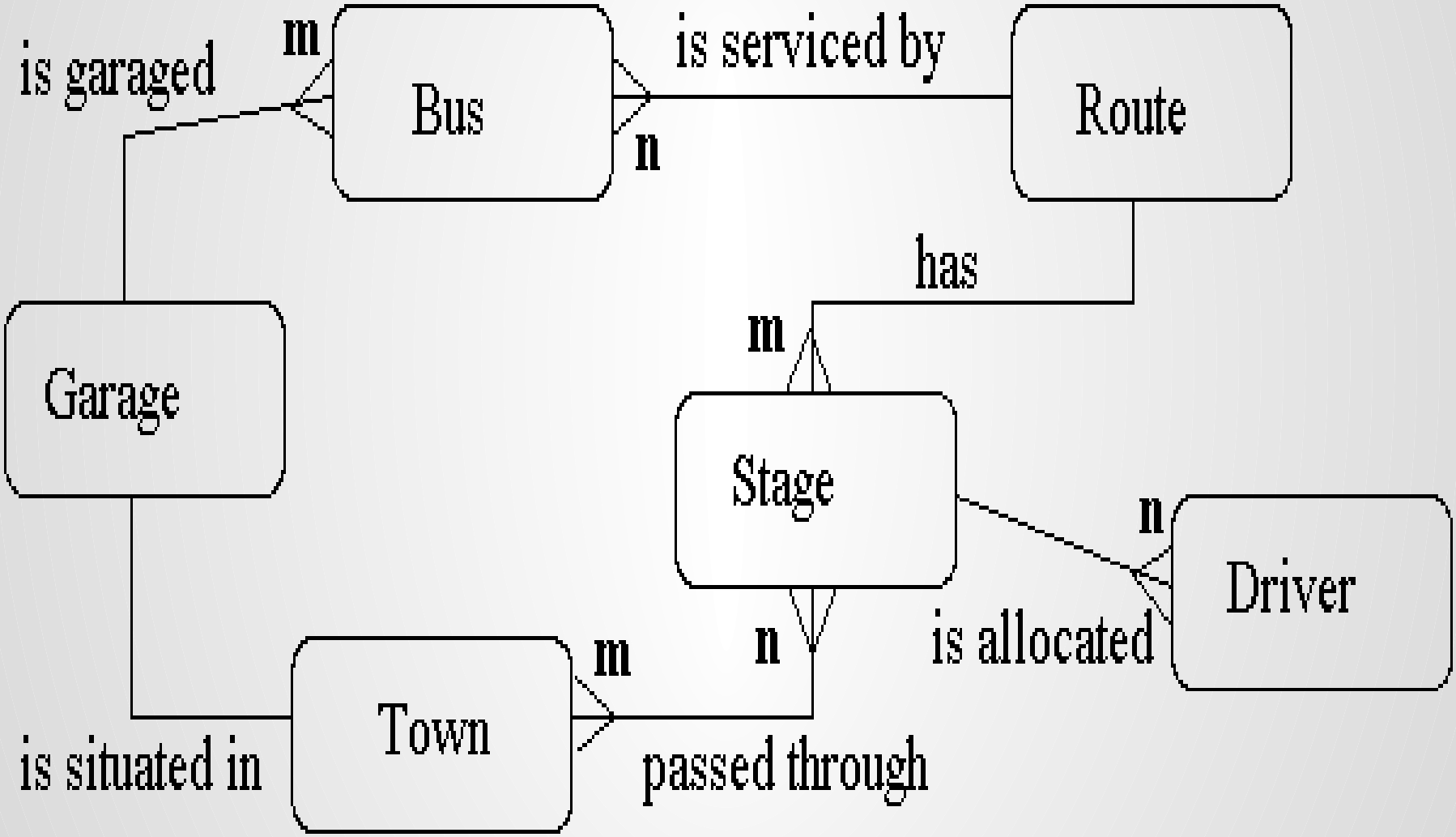
  double-decked.
- Each route is identified by a route number
- and information is available on the average number of passengers

  carried per day for each route.
- Drivers have an employee number, name, address, and sometimes a telephone number.

Entities

- Bus - Company owns busses and will hold information about them.
- Route - Buses travel on routes and will need described.
- Town - Buses pass through towns and need to know about them
- Driver - Company employs drivers, personnel will hold their data.
- Stage - Routes are made up of stages
- Garage - Garage houses buses, and need to know where they are.

E-R diagram


Bus
Route
Garage
Stage
Driver
Town
is garaged
m
n
is serviced by
has
m
n
n
is allocated
m
n
is situated in
passed through

Attributes

Bus (reg-no,make,size,deck,no-pass)
Route (route-no,avg-pass)
Driver (emp-no,name,address,tel-no)
Town (name)
Stage (stage-no)
Garage (name,address)

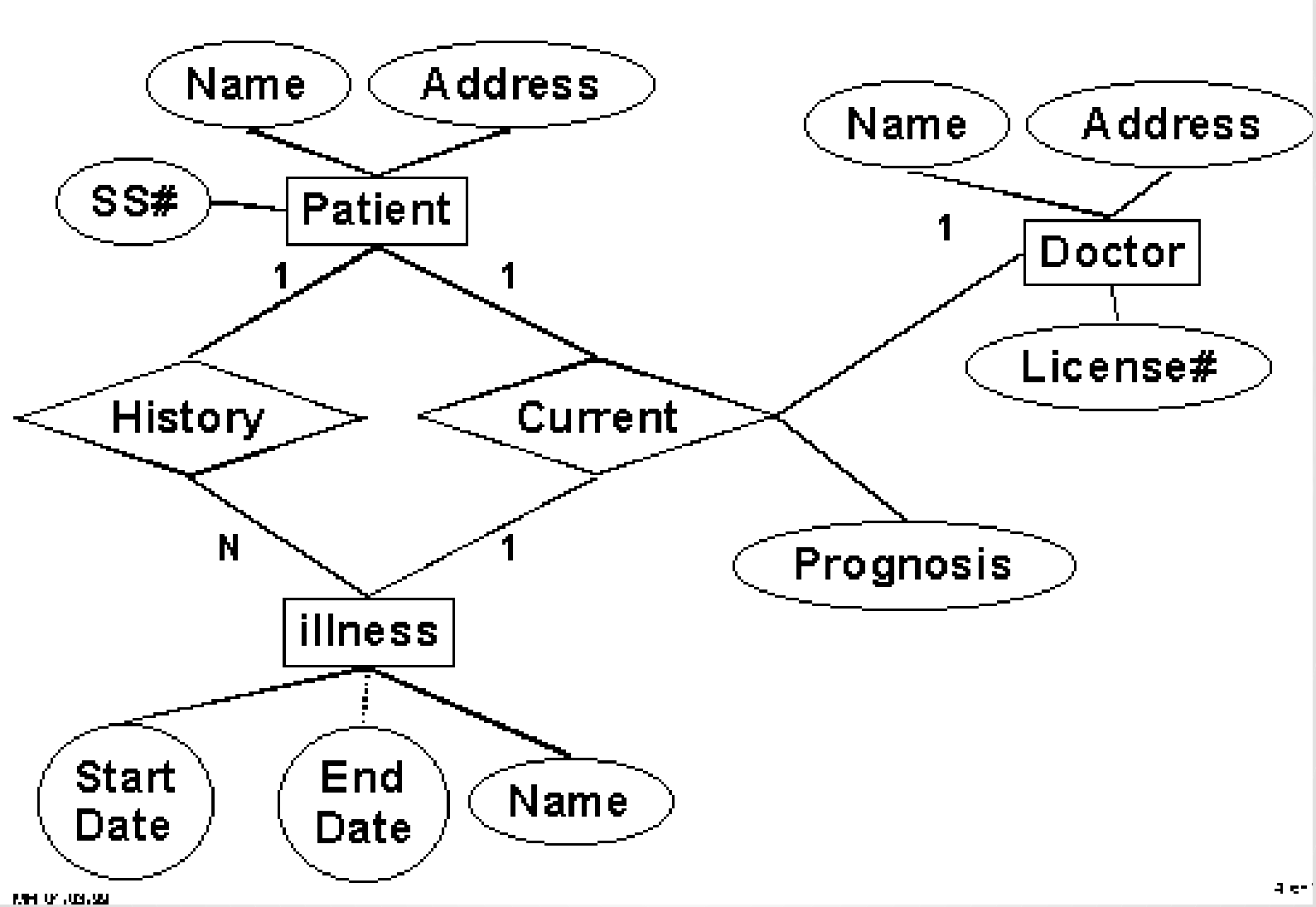
Relationships

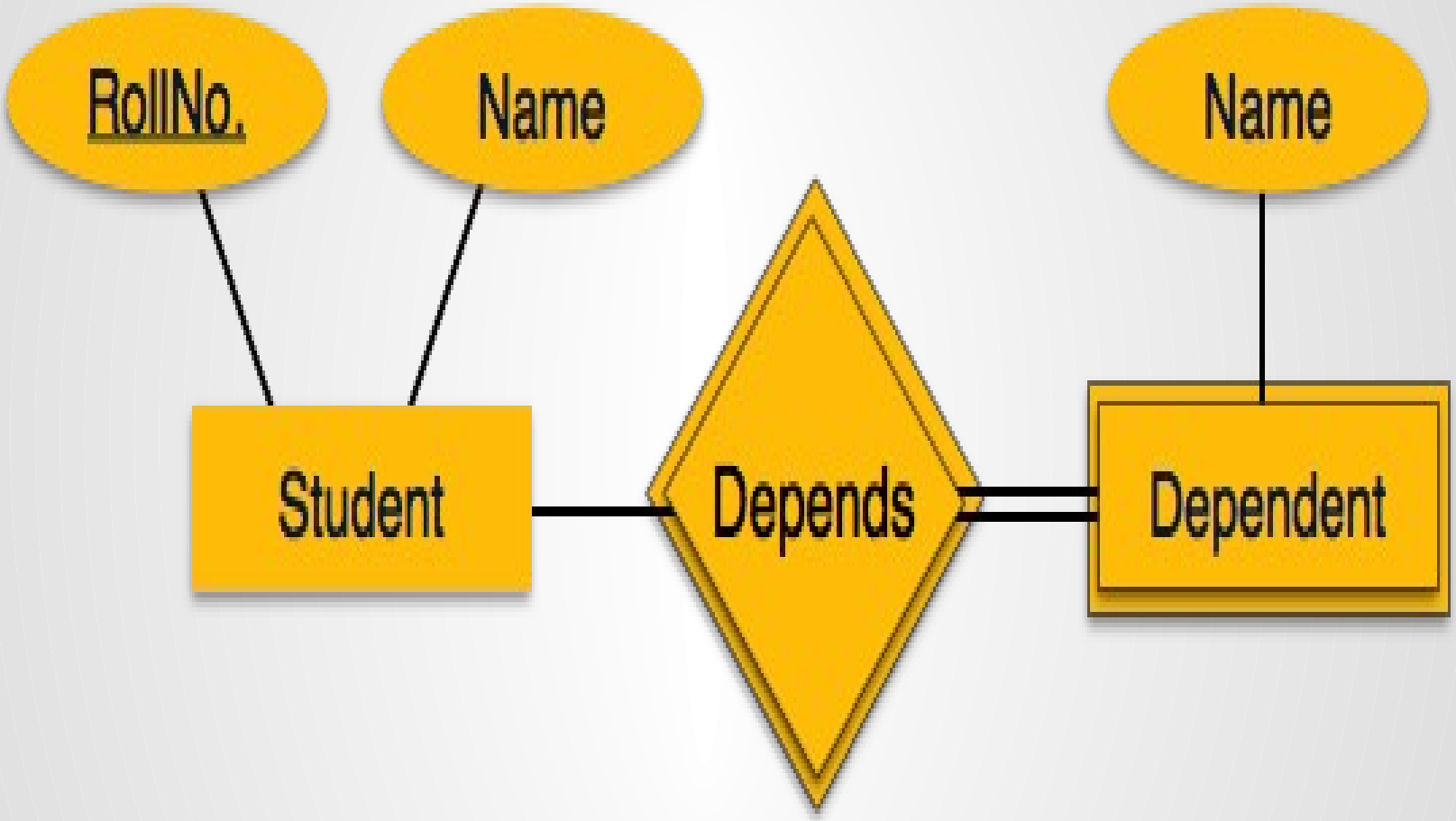
- A bus is allocated to a route and a route may have several buses.
- Bus-route (m:1) is serviced by
- A route comprises of one or more stages.
- route-stage (1:m) comprises
- One or more drivers are allocated to each stage.
- driver-stage (m:1) is allocated
- A stage passes through some or all of the towns on a route.
- stage-town (m:n) passes-through
- A route passes through some or all of the towns
- route-town (m:n) passes-through
- Some of the towns have a garage
- garage-town (1:1) is situated
- A garage keeps buses and each bus has one `home' garage
- garage-bus (m:1) is garaged

Steps to draw ER diagram

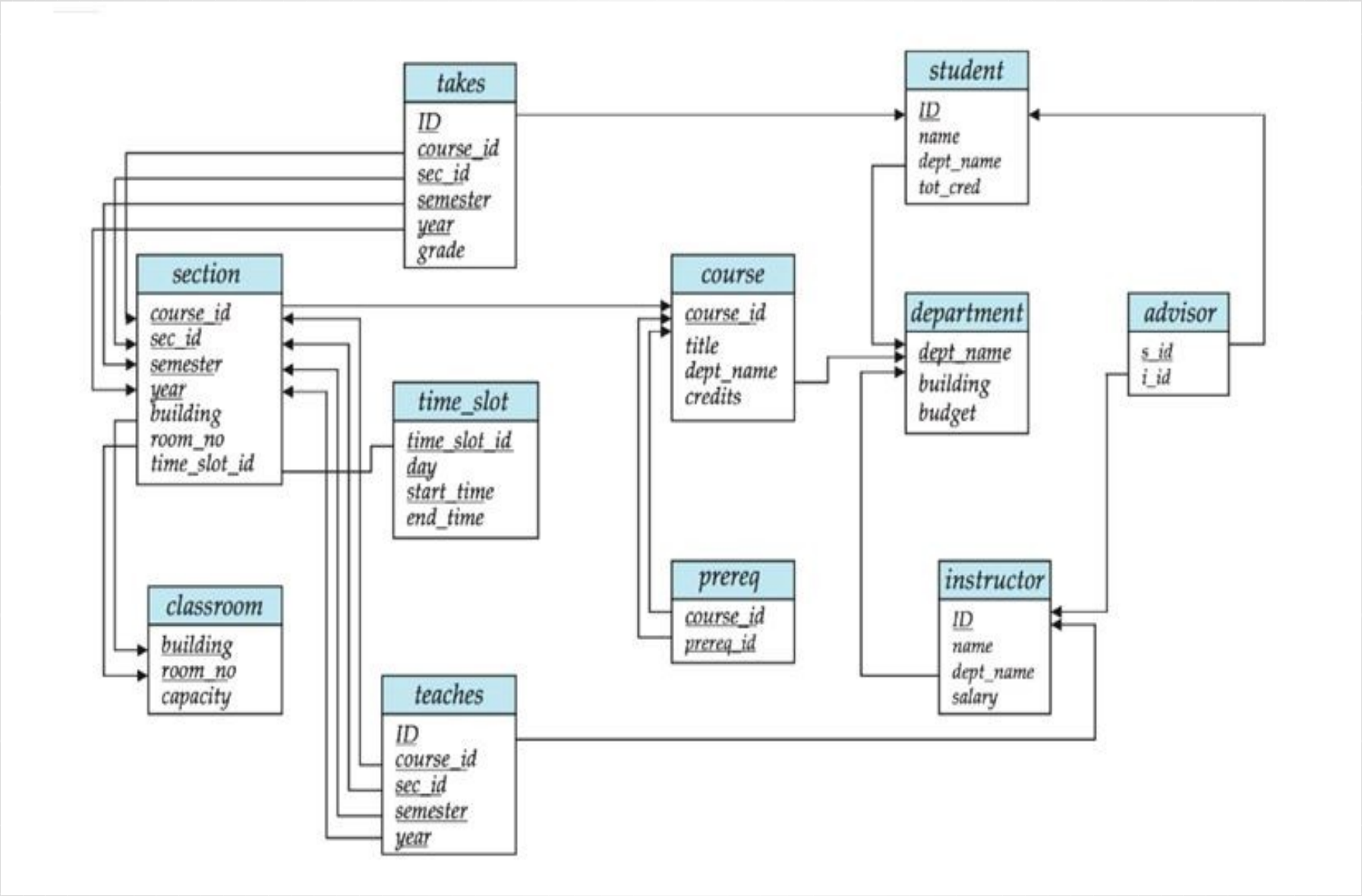
- **Identify entities** - list all potential entity types. These are the object of interest in the system. It is better to put too many entities in at this stage and them discard them later if necessary.
- **Remove duplicate entities** - Ensure that they really separate entity types or just two names for the same thing.
  - Also do not include the system as an entity type
  - e.g. if modeling a library, the entity types might be books, borrowers, etc.
  - The library is the system, thus should not be an entity type.
- **List the attributes of each entity** (all properties to describe the entity which are relevant to the application).
  - Ensure that the entity types are really needed.
  - are any of them just attributes of another entity type?
  - if so keep them as attributes and cross them off the entity list.
  - Do not have attributes of one entity as attributes of another entity!
- **Mark the primary keys.**
  - Which attributes uniquely identify instances of that entity type?
  - This may not be possible for some weak entities.
- **Define the relationships**
  - Examine each entity type to see its relationship to the others.
- **Describe the cardinality of the relationships**
  - Examine the constraints between participating entities.
- **Remove redundant relationships**
  - Examine the ER model for redundant relationships.

Hospital Management System

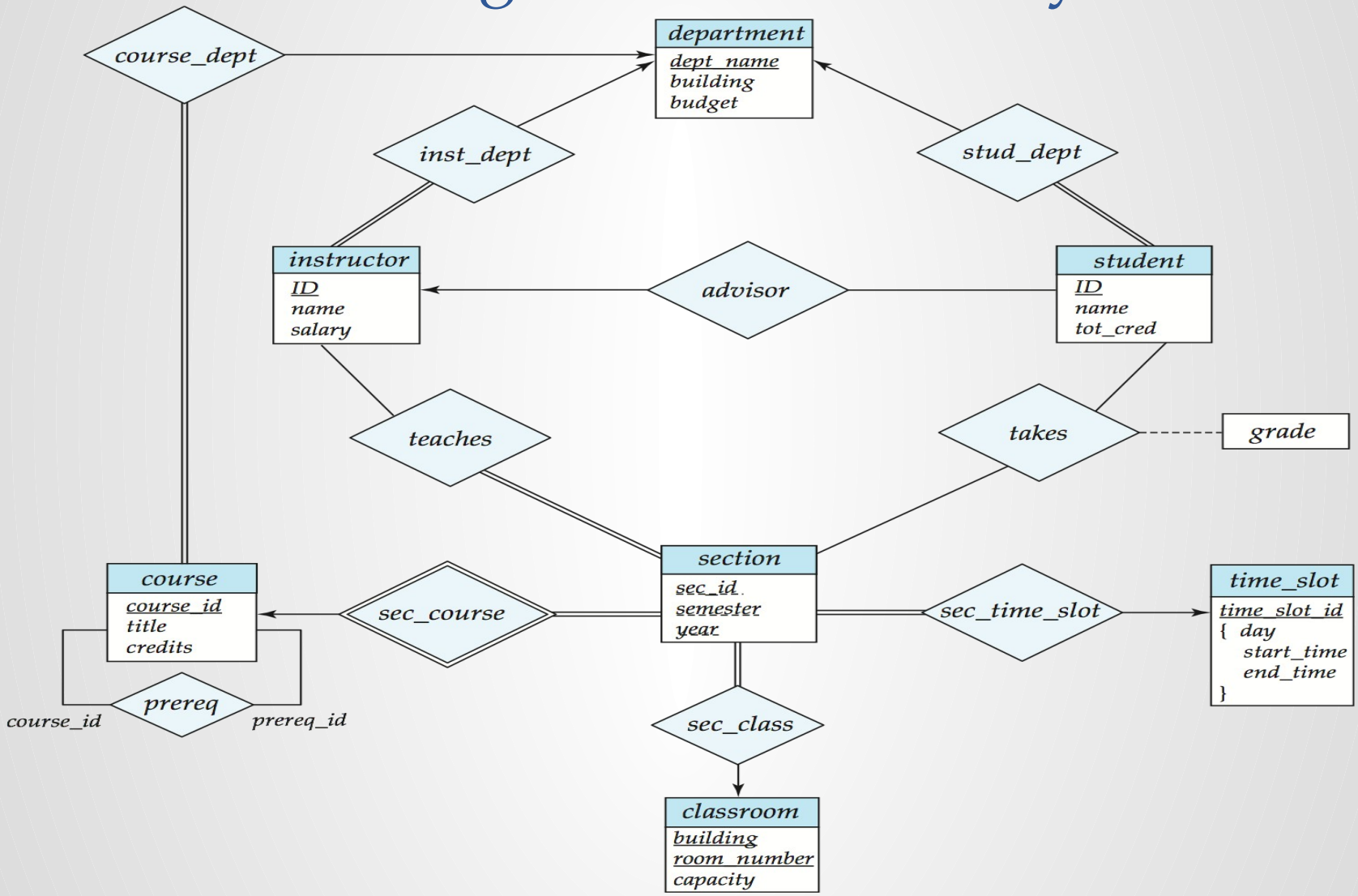



RollNo.
Name
Name
Student
Depends
Dependent

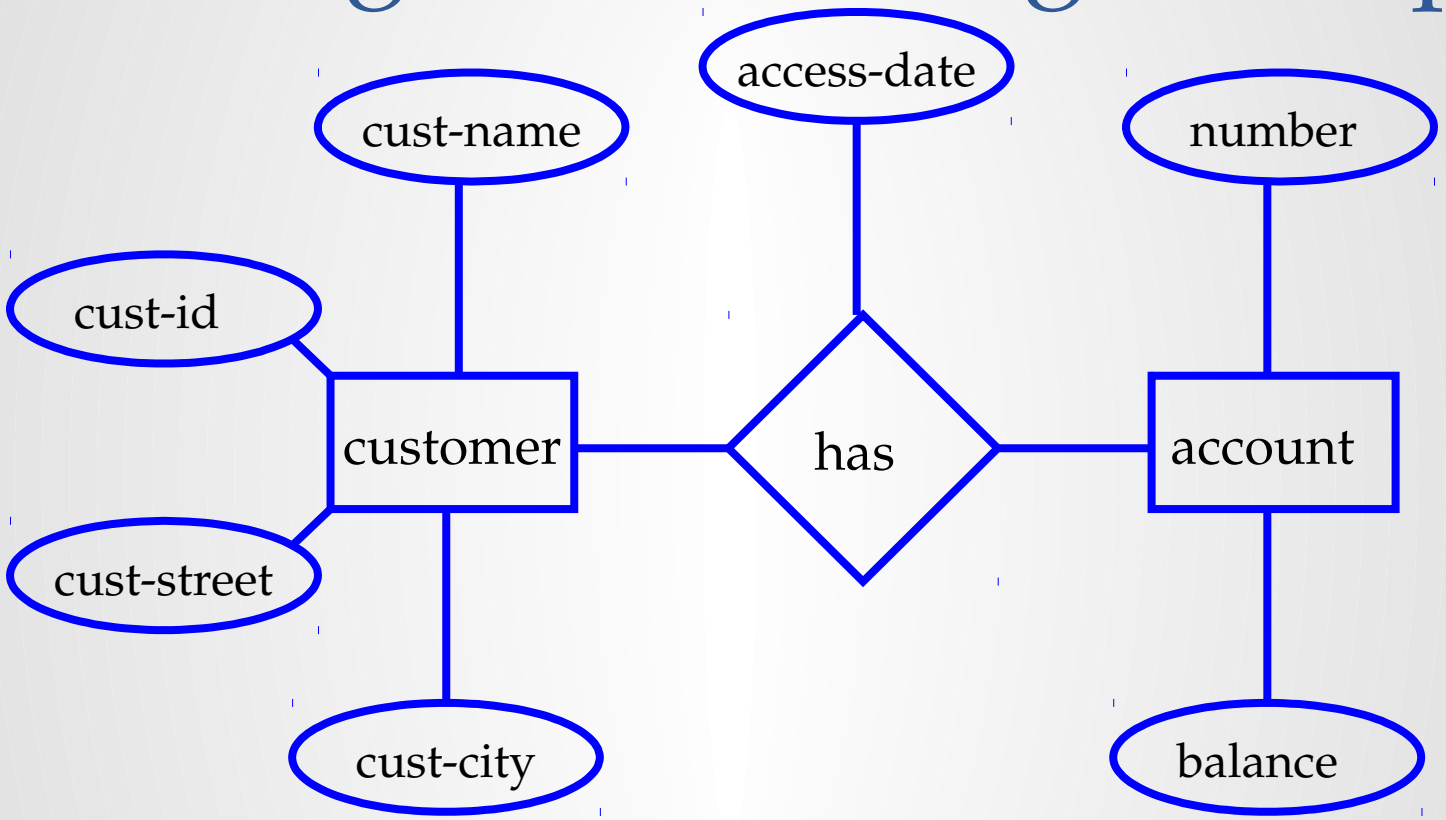
Schema DIAGRAM OF University database



ER diagram for University



ER Diagram: Starting Example



- Rectangles: entity sets
- Diamonds: relationship sets
- Ellipses: attributes

Slides Courtesy: Amol Deshpande

Next: Relationship Cardinalities

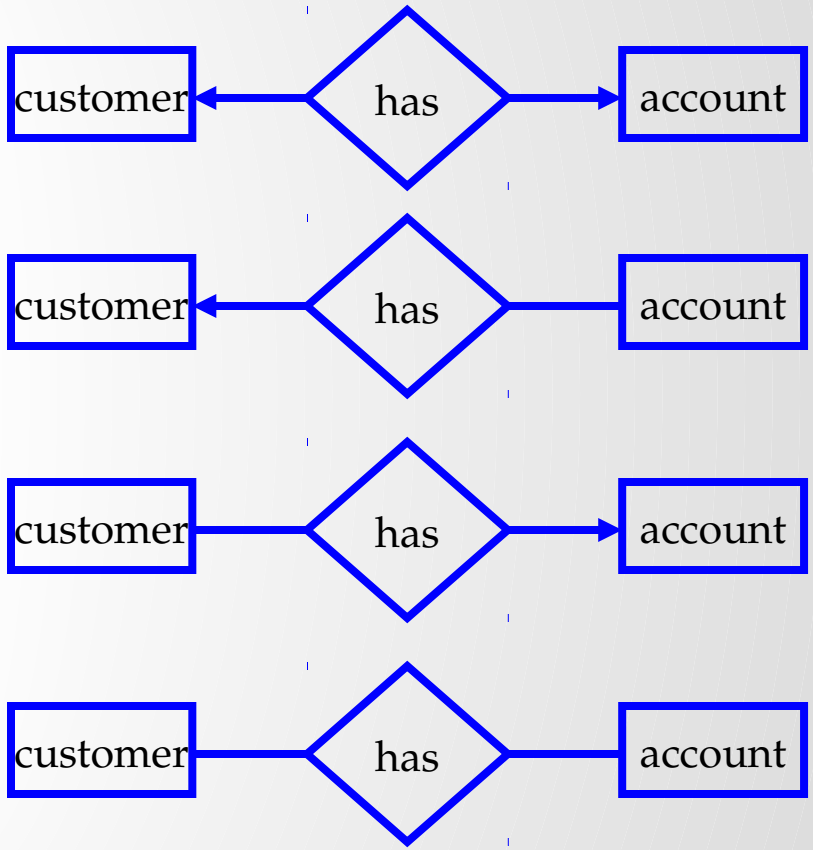
- We may know:
  - One customer can only open one account
  - *OR*
  - One customer can open multiple accounts
- Representing this is important
- Why ?
  - Better manipulation of data
  - Can enforce such a constraint
  - Remember: If not represented in conceptual model, the domain knowledge may be lost

Mapping Cardinalities

- Express the number of entities to which another entity can be associated via a relationship set
- Most useful in describing binary relationship sets

Mapping Cardinalities

- One-to-One
- One-to-Many
- Many-to-One
- Many-to-Many



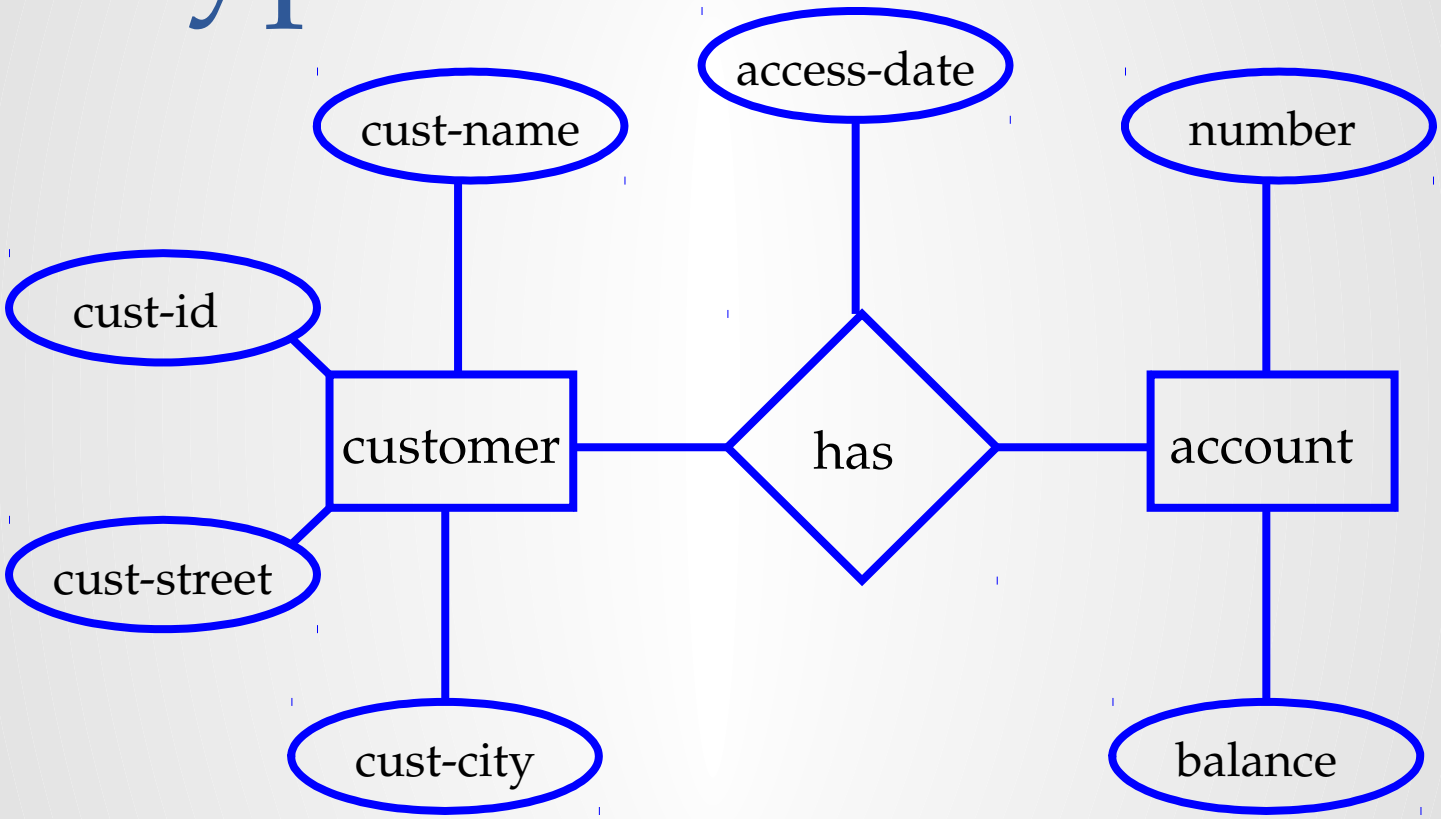
Mapping Cardinalities

- Express the number of entities to which another entity can be associated via a relationship set
- Most useful in describing binary relationship sets
- N-ary relationships ?

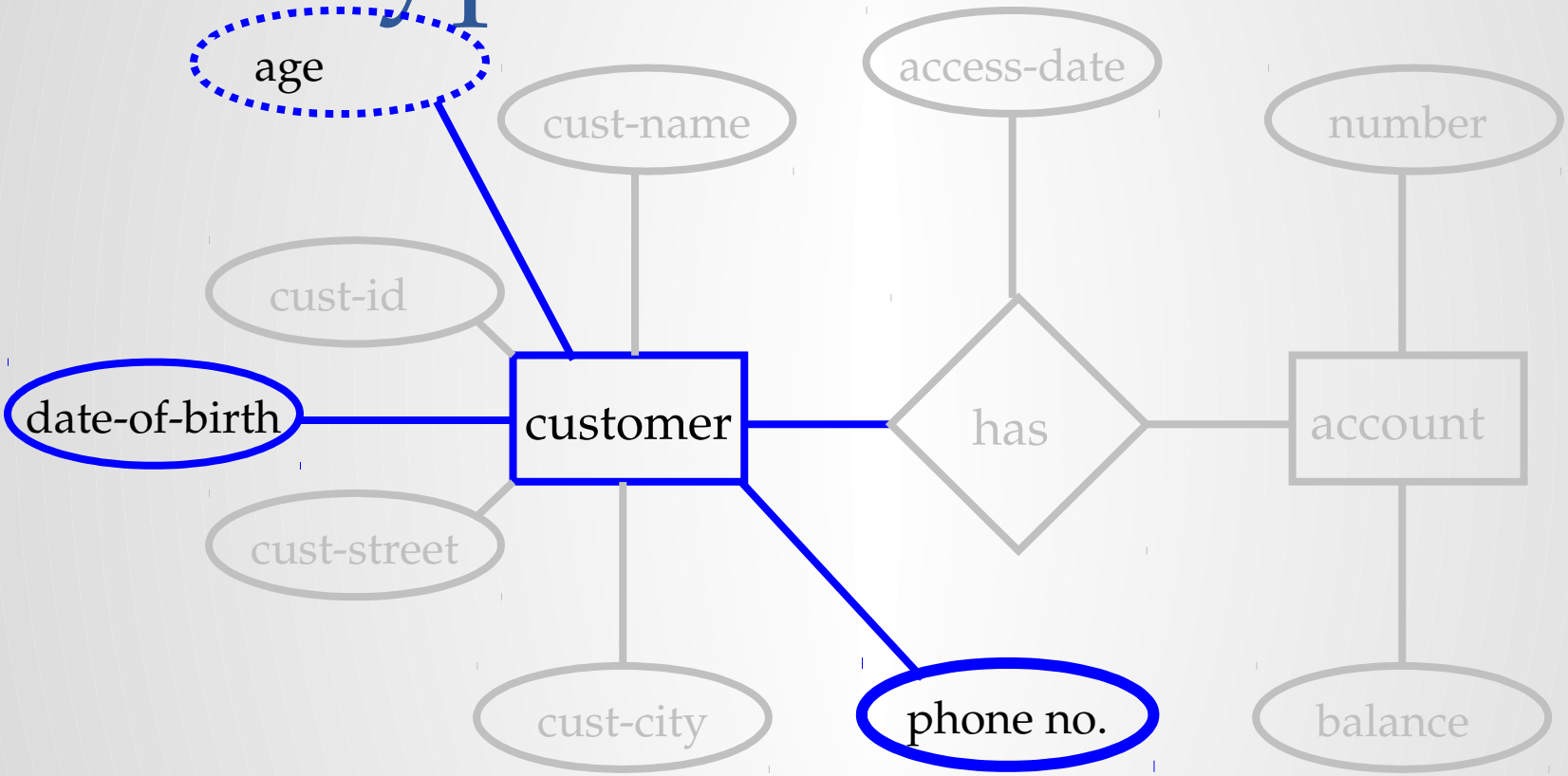
Next: Types of Attributes

- Simple vs Composite
  - Single value per attribute ?
- Single-valued vs Multi-valued
  - E.g. Phone numbers are multi-valued
- Derived
  - If date-of-birth is present, age can be derived
  - Can help in avoiding redundancy, enforcing constraints etc…

Types of Attributes

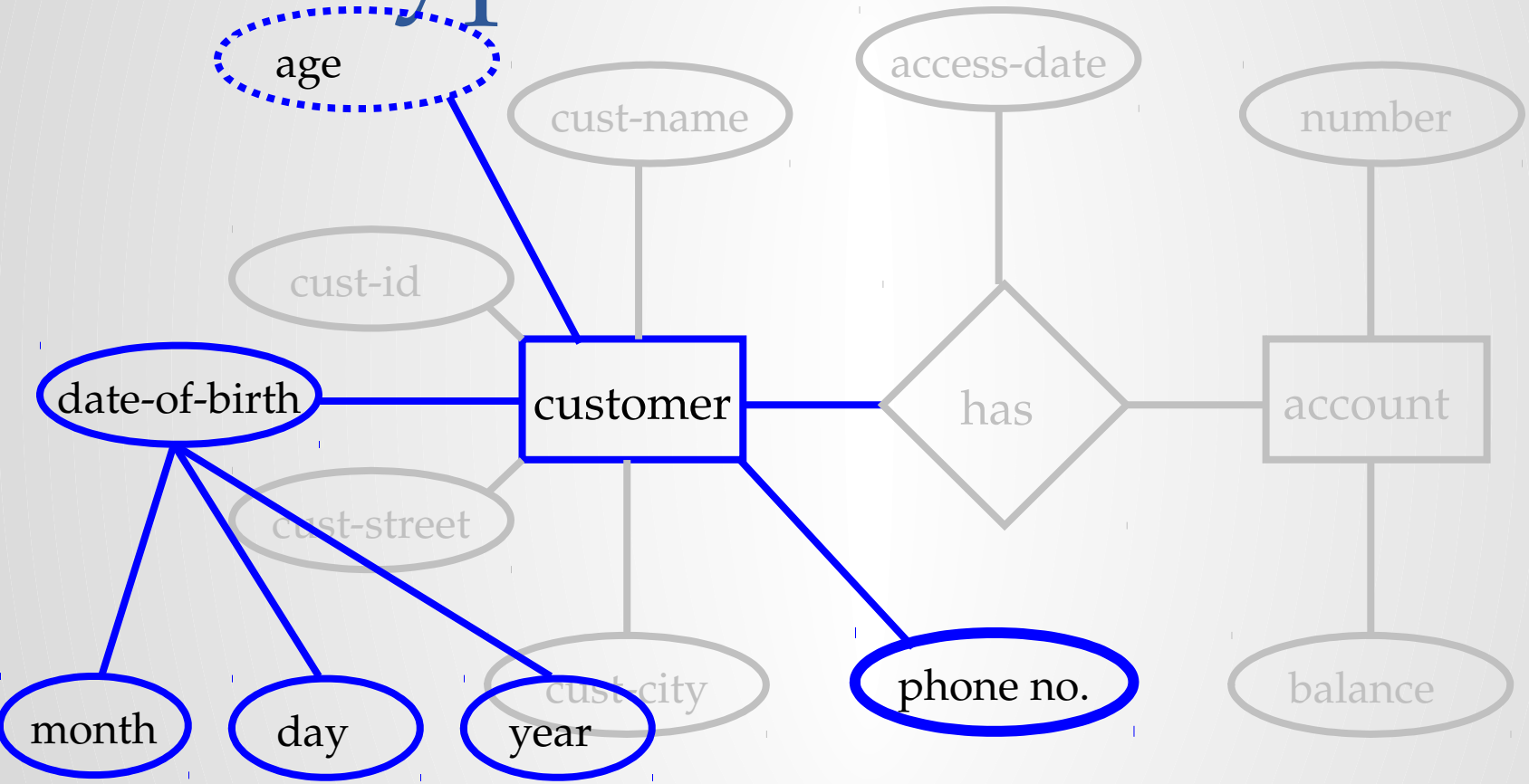


Types of Attributes



- **multi-valued (double ellipse)**
- **derived (dashed ellipse)**

Types of Attributes

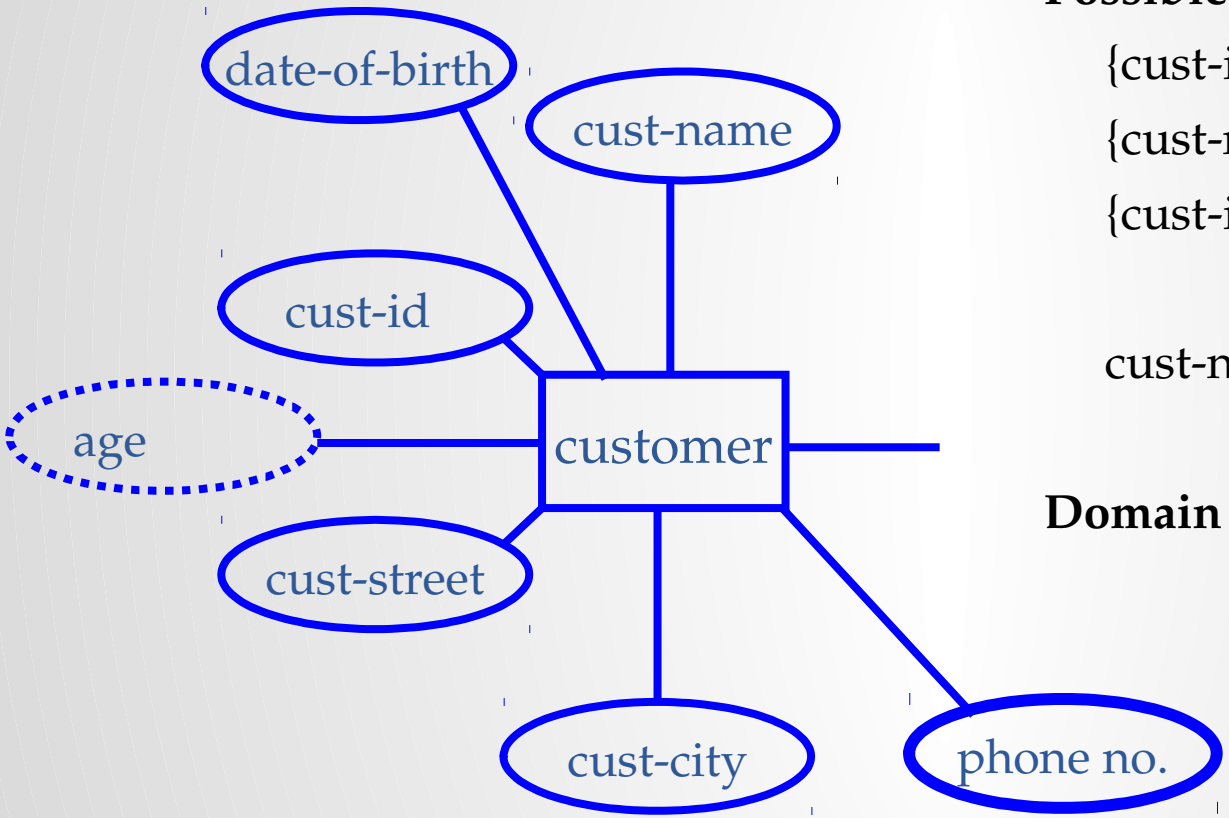


Composite Attribute

Next: Keys

- Key = set of attributes identifying individual entities or relationships

Entity Keys



Possible Keys:

{cust-id}

{cust-name, cust-city, cust-street}

{cust-id, age}

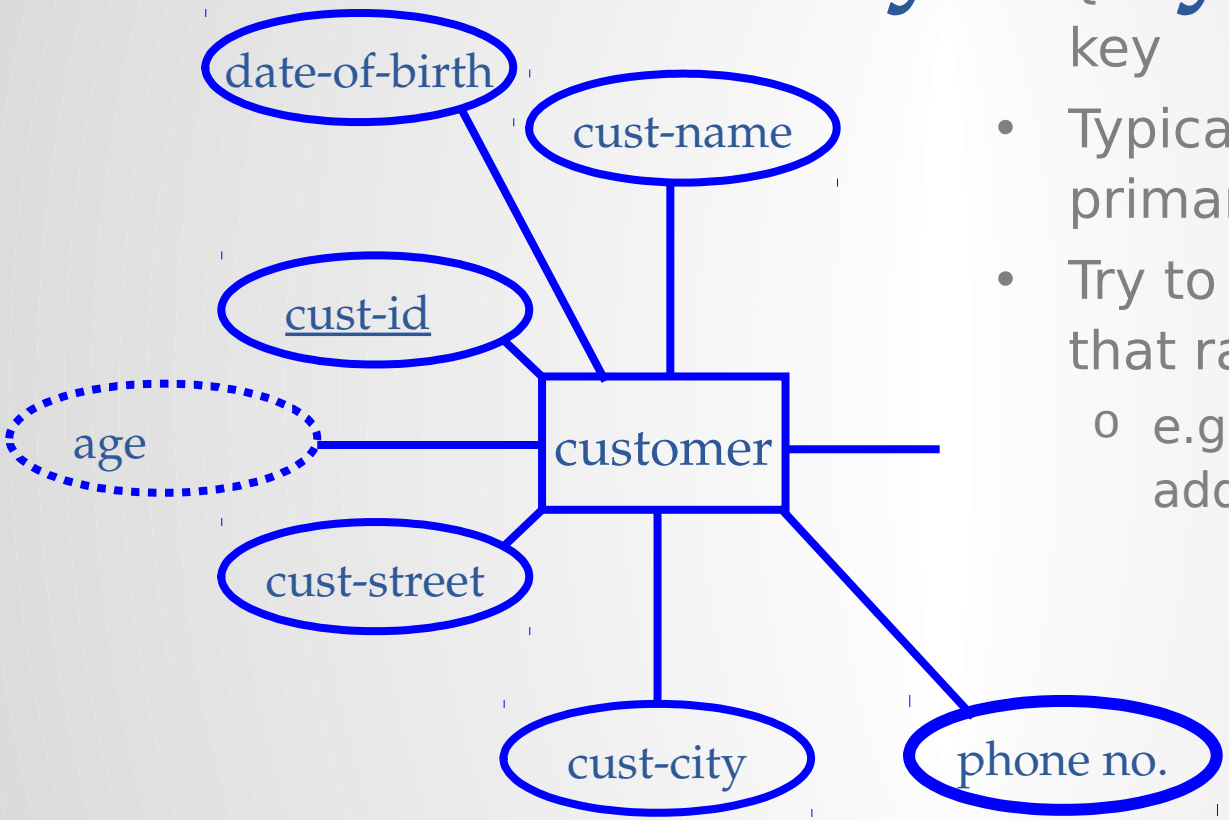
cust-name ?? Probably not.

Domain knowledge dependent !!

Entity Keys

- *Superkey*
  - any attribute set that can distinguish entities
- *Candidate key*
  - a minimal superkey
    - Can't remove any attribute and preserve key-ness
      - {cust-id, age} not a superkey
      - {cust-name, cust-city, cust-street} is
        - assuming cust-name is not unique
- *Primary key*
  - Candidate key chosen as *the* key by DBA
  - Underlined in the ER Diagram

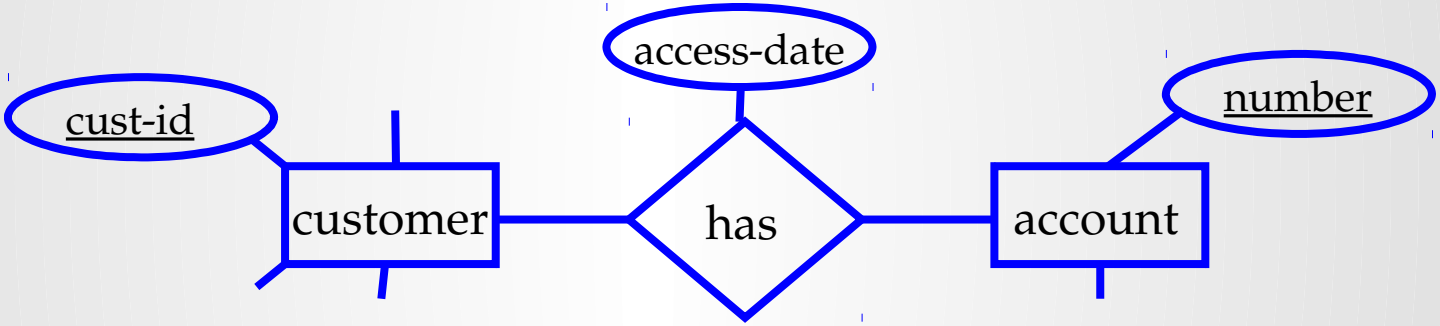
Entity Keys



- {*cust-id*} is a natural primary key
- Typically, SSN forms a good primary key
- Try to use a candidate key that rarely changes
  - e.g. something involving address not a great idea

Relationship Set Keys

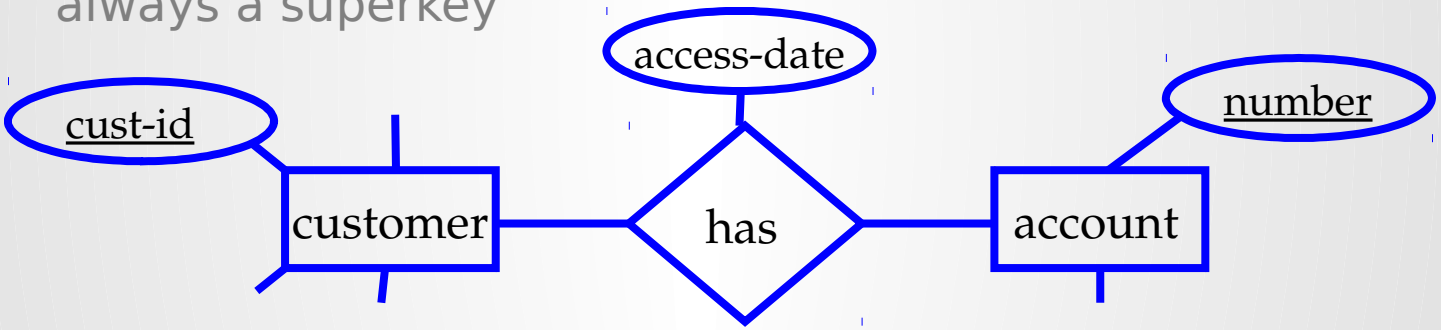
- What attributes are needed to represent a relationship completely and uniquely ?
  - Union of primary keys of the entities involved, and relationship attributes



  - *{cust-id, access-date, account number}* describes a relationship completely

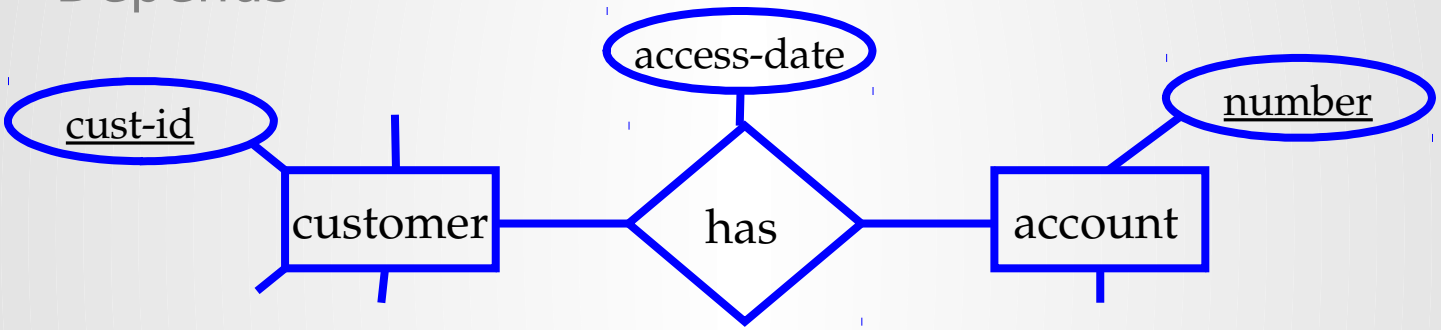
Relationship Set Keys

- Is *{cust-id, access-date, account number}* a candidate key *?*
  - No. Attribute *access-date* can be removed from this set without losing key-ness
  - In fact, union of primary keys of associated entities is always a superkey



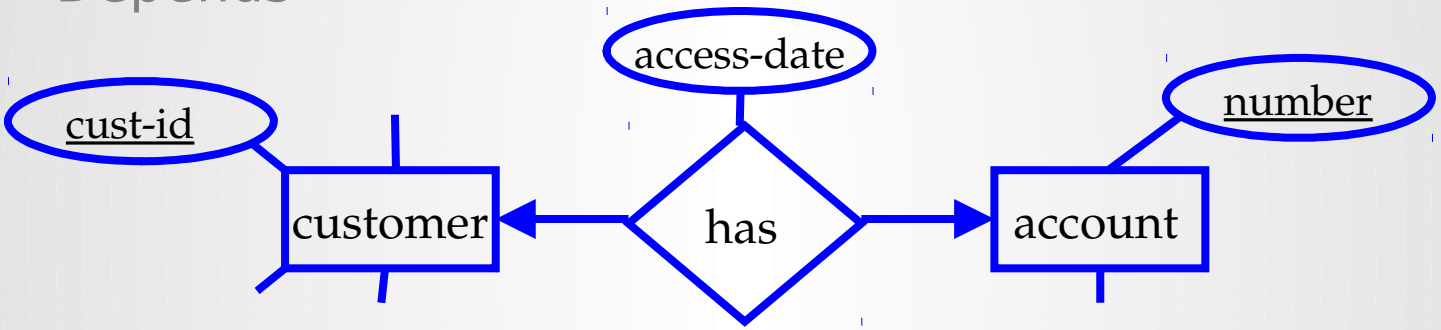
Relationship Set Keys

- Is {cust-id, account-number} a candidate key ?
  - Depends



Relationship Set Keys

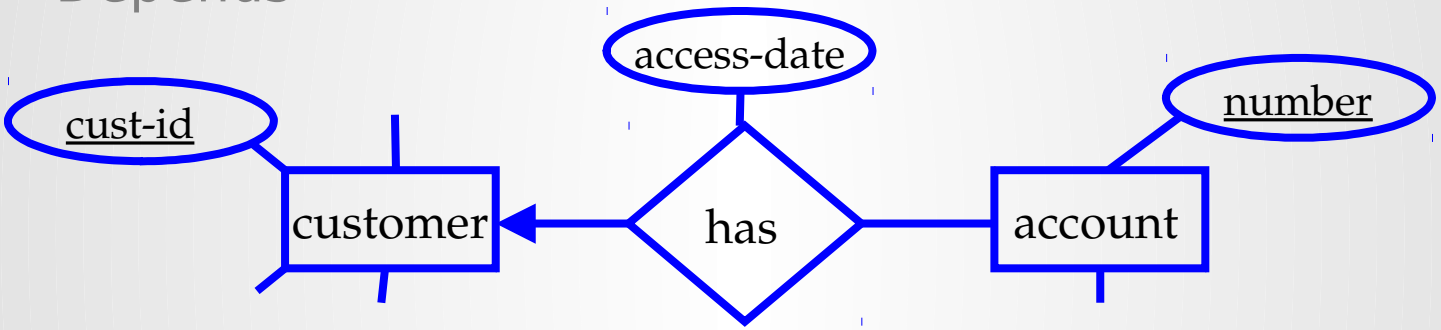
- Is {cust-id, account-number} a candidate key ?
  - Depends



- If one-to-one relationship, either *{cust-id}* or *{account-number}* sufficient
  - Since a given *customer* can only have one *account*, she can only participate in one relationship
  - Ditto *account*

Relationship Set Keys

- Is {cust-id, account-number} a candidate key ?
  - Depends



- If one-to-many relationship (as shown), *{account-number}* is a candidate key
  - A given customer can have many accounts, but at most one account holder per account allowed

Relationship Set Keys

- General rule for binary relationships
  - one-to-one: primary key of either entity set
  - one-to-many: primary key of the entity set on the many side
  - many-to-many: union of primary keys of the associate entity sets
- n-ary relationships
  - More complicated rules

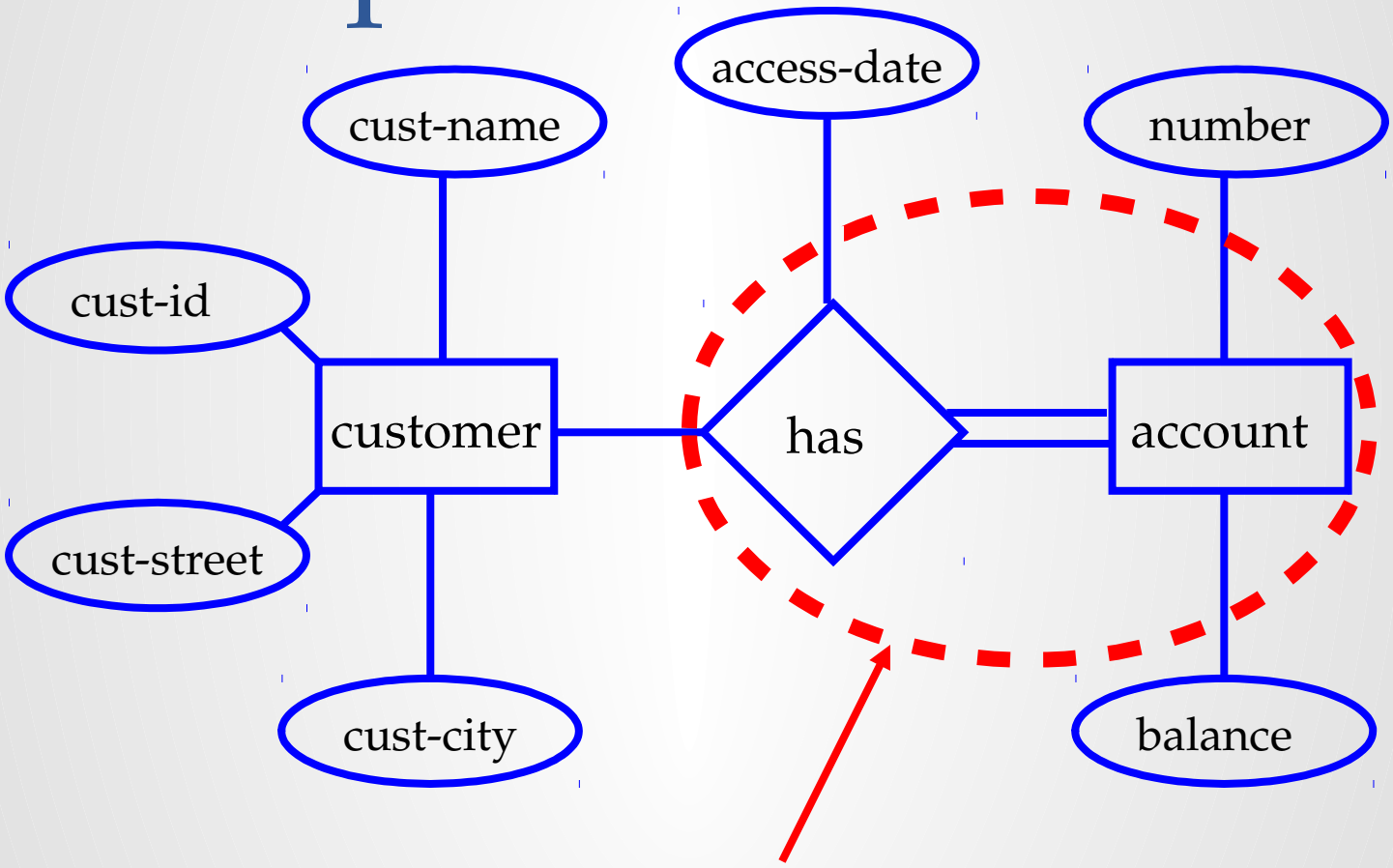
Next: Data Constraints

- Representing semantic data constraints
  - We already saw constraints on relationship cardinalities

Participation Constraint

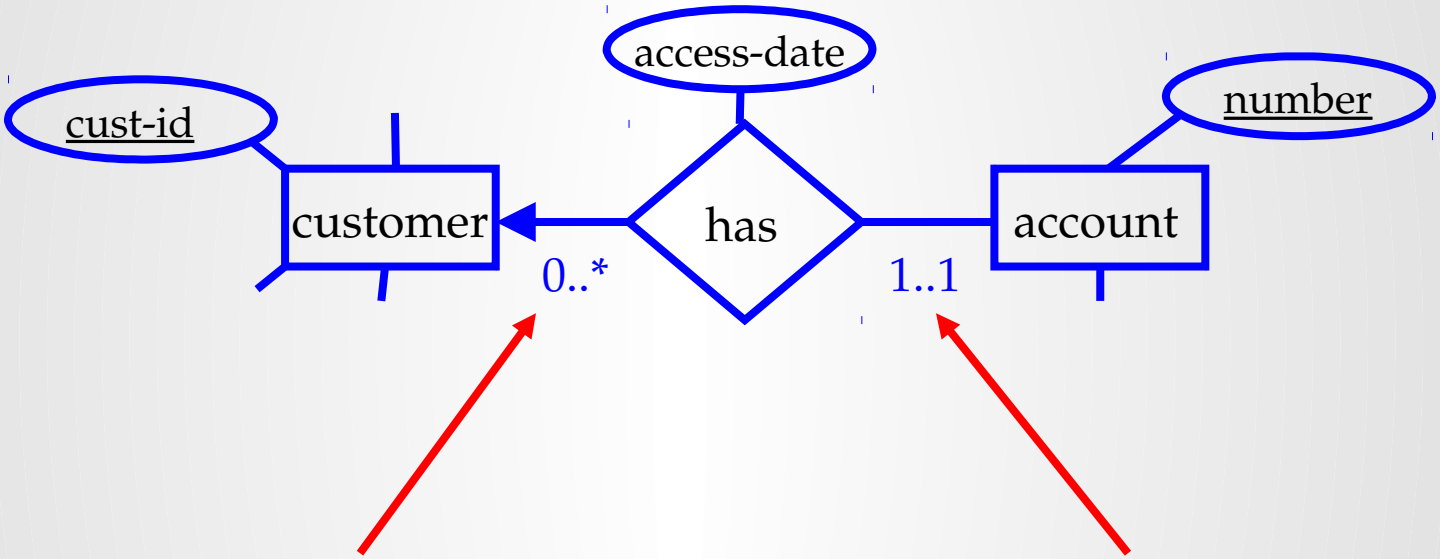
- Given an entity set E, and a relationship R it participates in:
  - If every entity in E participates in at least one relationship in R, it is total participation
  - partial otherwise

Participation Constraint



Cardinality Constraints

How many relationships can an entity participate in ?



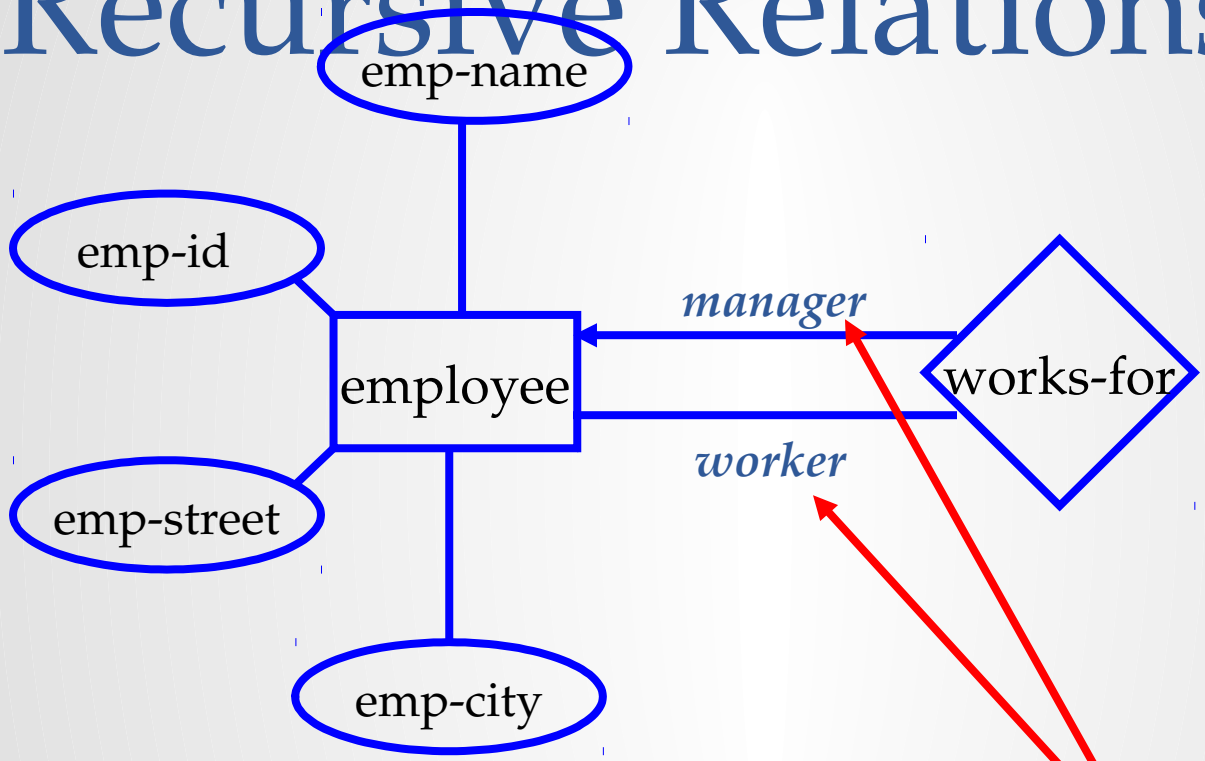
Minimum - 0
Maximum – no limit

Minimum - 1
Maximum - 1

Next: Recursive Relationships

- Sometimes a relationship associates an entity set to itself

Recursive Relationships



Must be declared with roles

Next: Weak Entity Sets

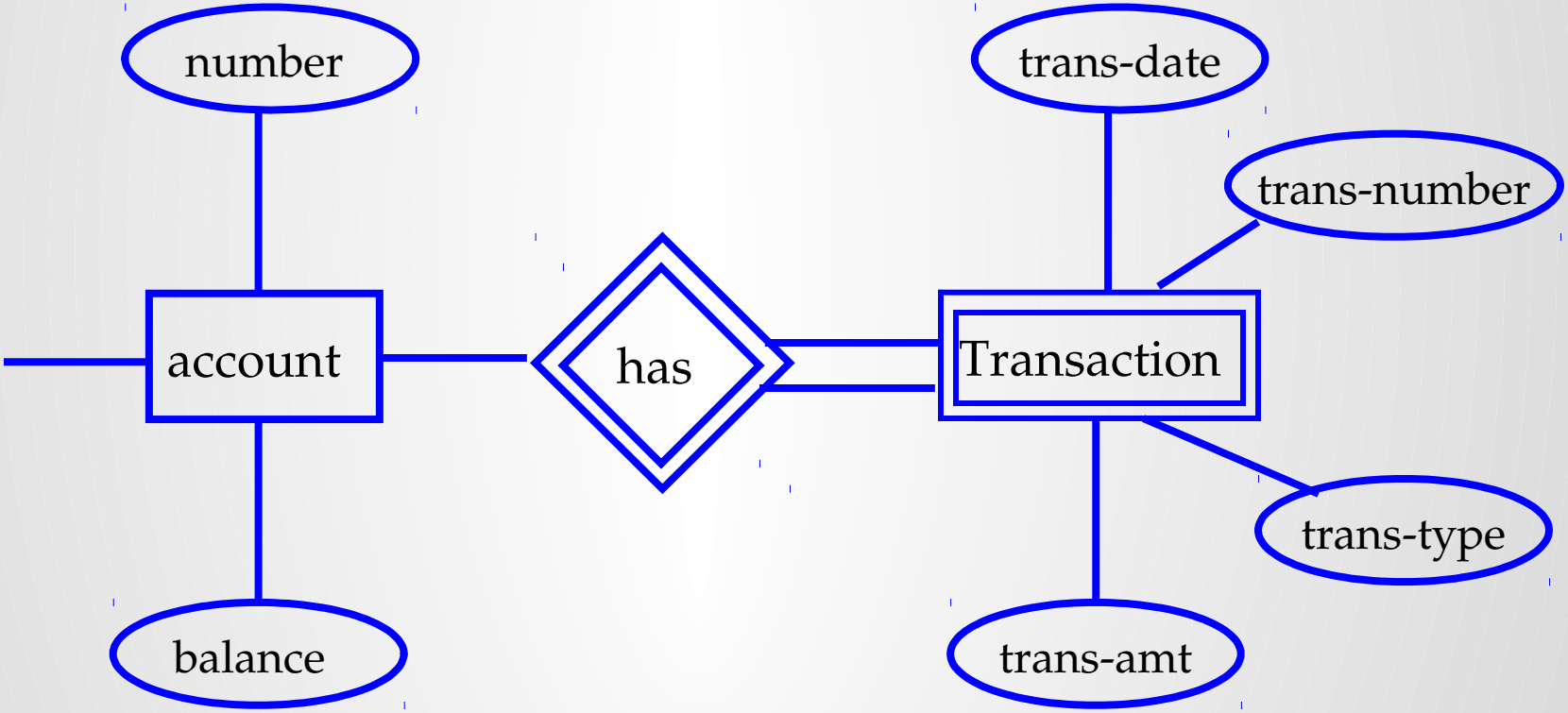
- An entity set without enough attributes to have a primary key
- E.g. Transaction Entity
  - Attributes:
    - transaction-number, transaction-date, transaction-amount, transaction-type
    - transaction-number: may not be unique across accounts

Weak Entity Sets

- A weak entity set must be associated with an identifying or owner entity set
- Account is the owner entity set for Transaction

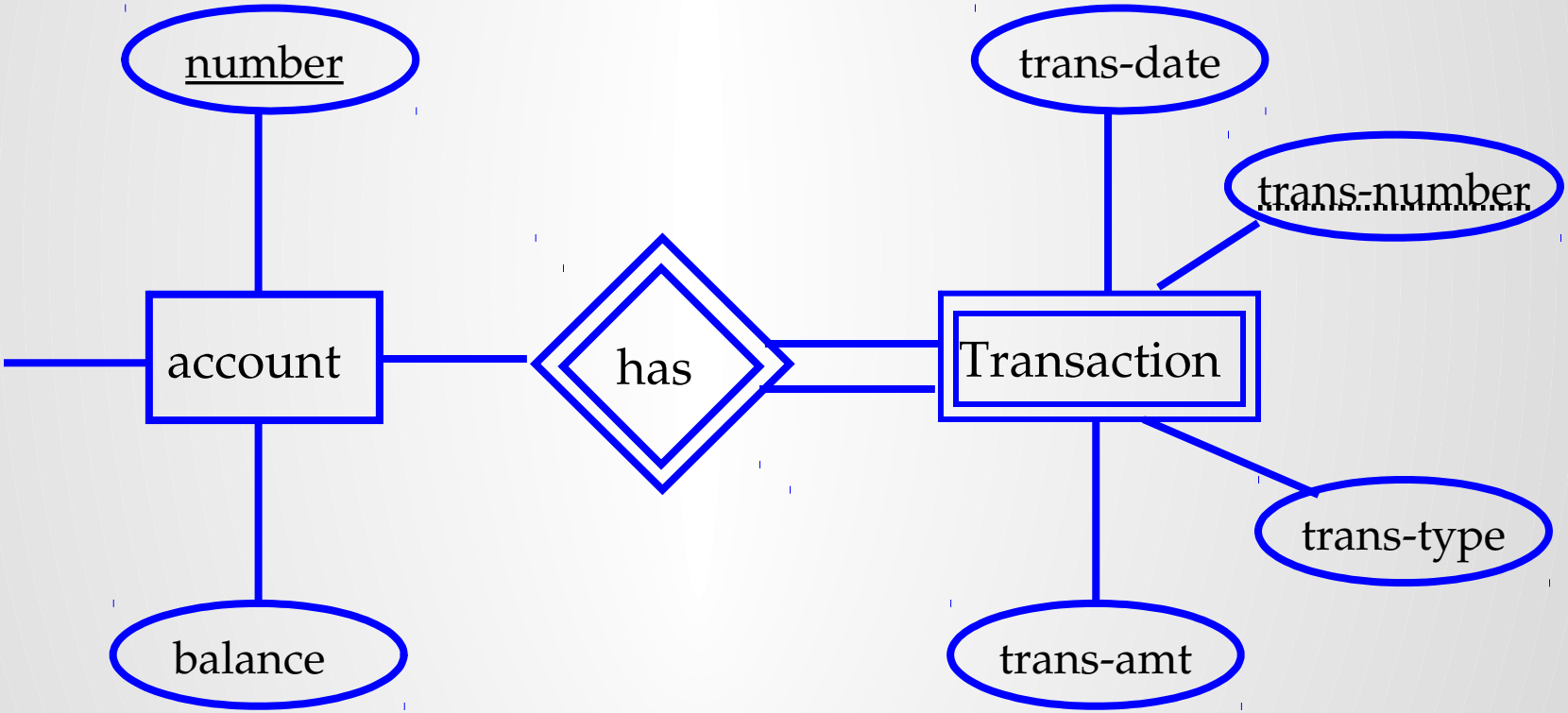
Weak Entity Sets

Still need to be able to distinguish between different weak entities associated with the same strong entity



Weak Entity Sets

Discriminator: A set of attributes that can be used for that



Weak Entity Sets

- Primary key:
  - Primary key of the associated strong entity + discriminator attribute set
  - For Transaction:
    - *{account-number, transaction-number}*

Next: Specialization

- Consider entity person:
  - Attributes: name, street, city
- Further classification:
  - customer
    - Additional attributes: customer-id, credit-rating
  - employee
    - Additional attributes: employee-id, salary
- Note similarities to object-oriented programming