Dept. of Computer Science and Engineering
University of Rajshahi
www.ru.ac.bd

Dr. Shamim Ahmad

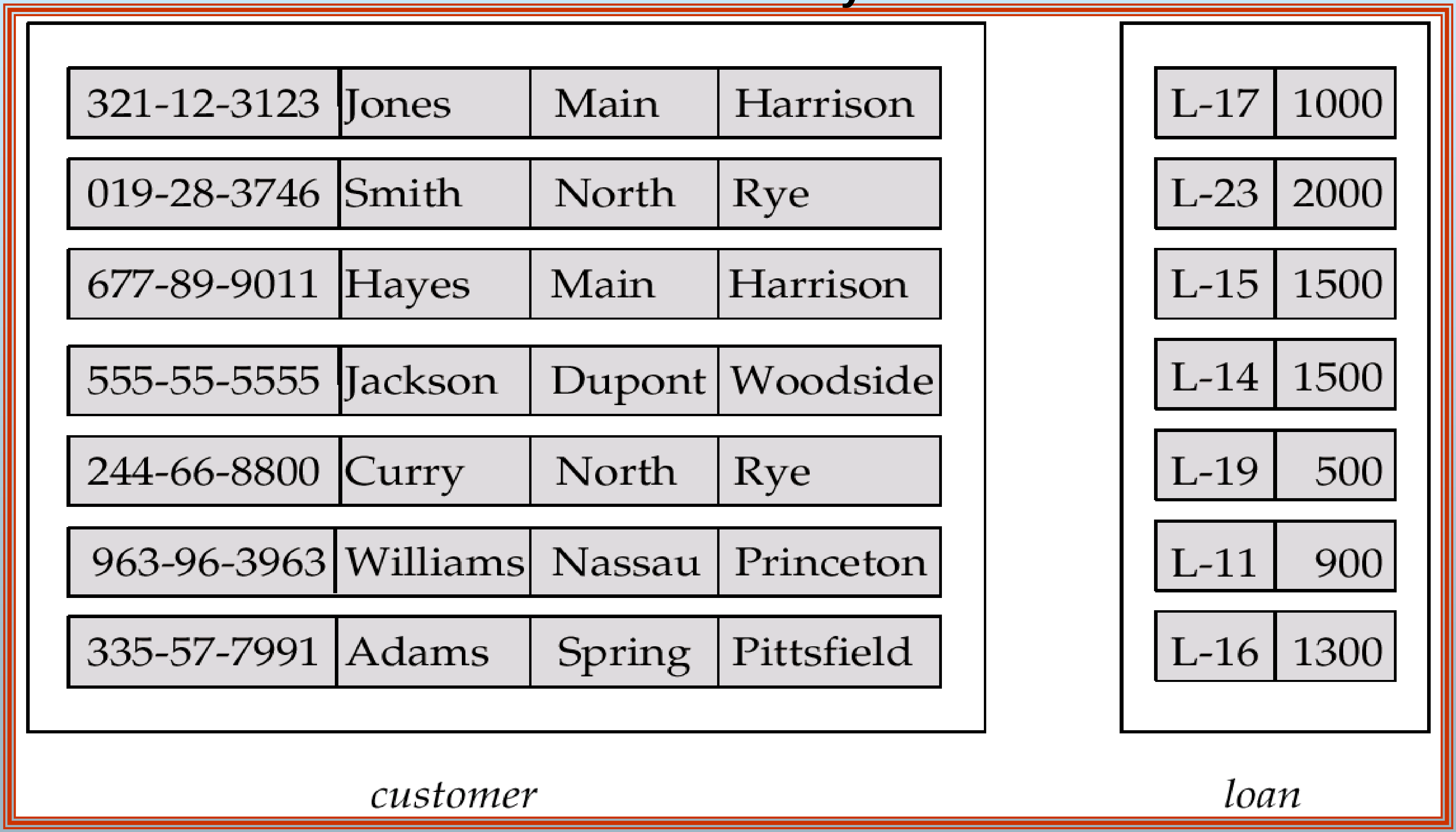
Chapter 2: Entity-Relationship Model

- Basic Concepts
- Constraints
- Keys
- Design Issues
- E-R Diagram

Basic Concepts

- Entity Sets
  - A *database* can be modeled as:
    - a collection of entities,
    - relationship among entities.
  - An ***entity*** is an object that exists and is distinguishable from other objects.
    - Example: specific person, company, event, plant
  - Entities have *attributes*
    - Example: people have *names* and *addresses*
  - An ***entity set*** is a set of entities of the same type that share the same properties.
    - Example: set of all persons, companies, trees, holidays

Entity Sets *customer* and *loan*



customer-id	customer-name	customer-street	customer-city
321-12-3123	Jones	Main	Harrison
019-28-3746	Smith	North	Rye
677-89-9011	Hayes	Main	Harrison
555-55-5555	Jackson	Dupont	Woodside
244-66-8800	Curry	North	Rye
963-96-3963	Williams	Nassau	Princeton
335-57-7991	Adams	Spring	Pittsfield

customer

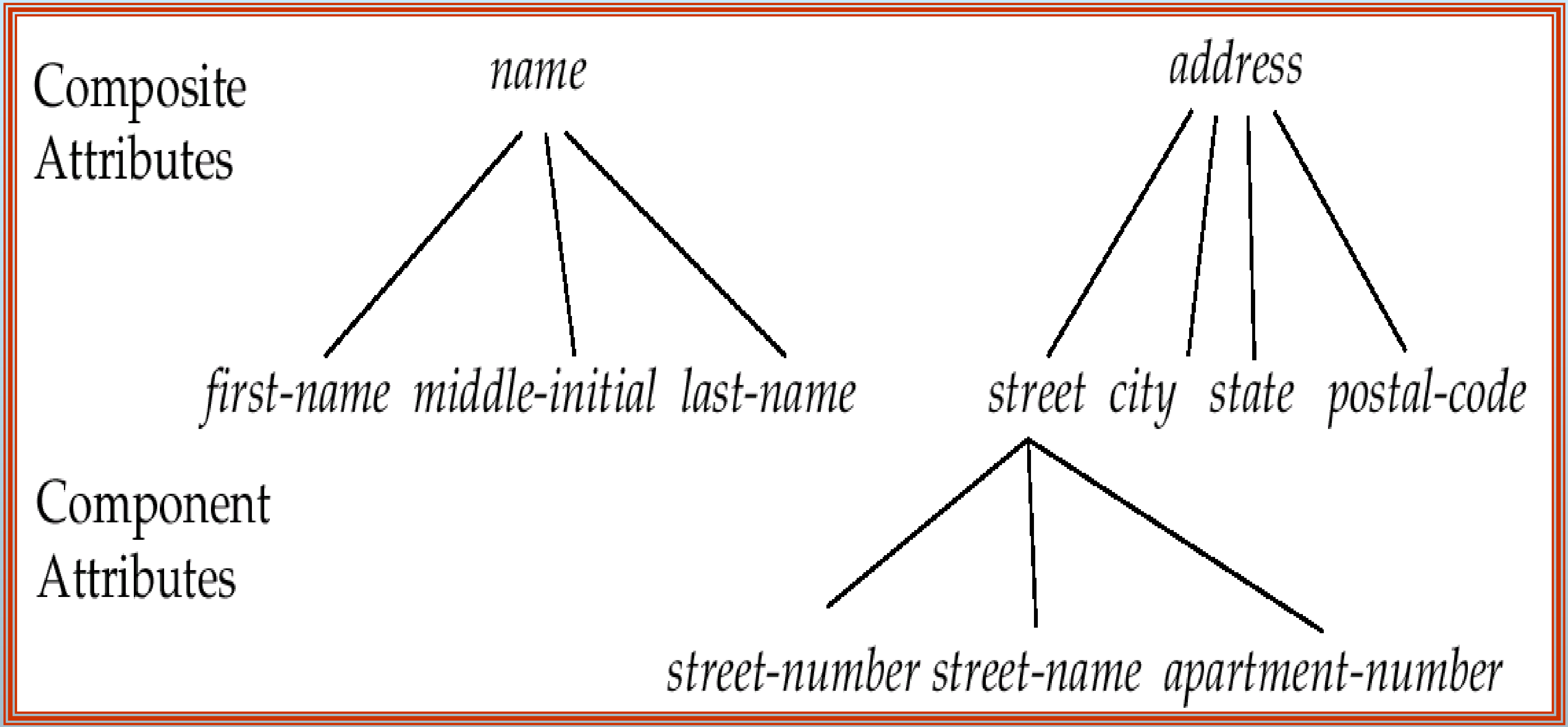
loan-number	amount
L-17	1000
L-23	2000
L-15	1500
L-14	1500
L-19	500
L-11	900
L-16	1300

loan

Basic Concepts (Cont.)

- Attributes
  - An entity is represented by a set of attributes, that is descriptive properties possessed by all members of an entity set.
  - *Domain* – the set of permitted values for each attribute
  - Attribute types:
    - *Simple* and *composite* attributes.
    - *Single-valued* and *multi-valued* attributes
      - E.g. multivalued attribute: *phone-numbers*
    - *Derived* attributes
      - Can be computed from other attributes
      - E.g. *age*, given date of birth

Composite Attributes



Composite Attributes

name

first-name *middle-initial* *last-name*

address

street *city* *state* *postal-code*

Component Attributes

street-number *street-name* *apartment-number*

Basic Concepts (Cont.)

- Relationship Sets
  - A relationship is an association among several entities

Example:

Hayes *depositor* A-102

customer entity relationship set *account* entity

  - A *relationship set* is a mathematical relation among $n \geq 2$ entities, each taken from entity sets

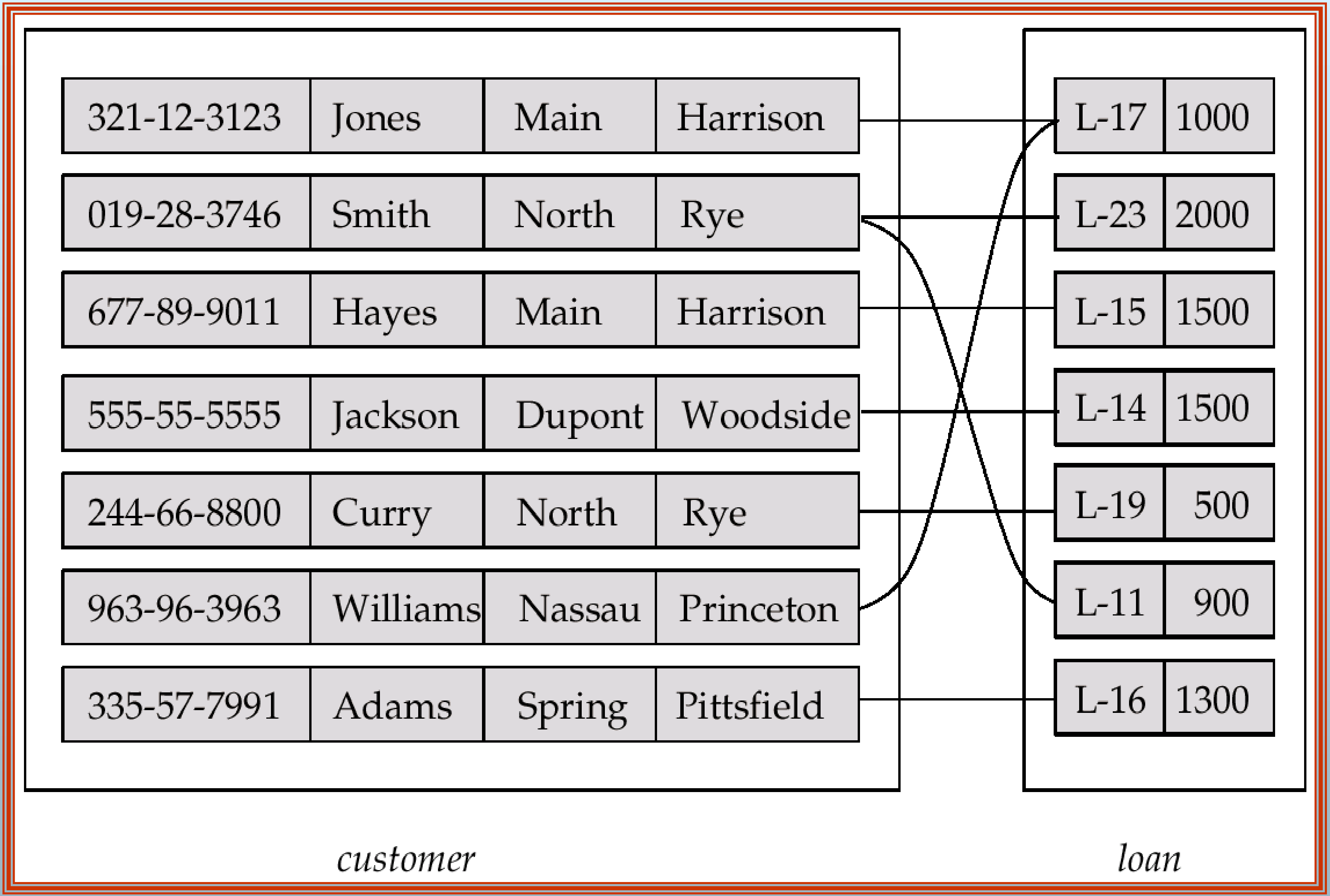
$\{(e_1, e_2, \ldots e_n) \mid e_1 \in E_1, e_2 \in E_2, \ldots, e_n \in E_n\}$

where $(e_1, e_2, \ldots, e_n)$ is a relationship

  - Example:

(Hayes, A-102) $\in$ *depositor*

Relationship Set *borrower*

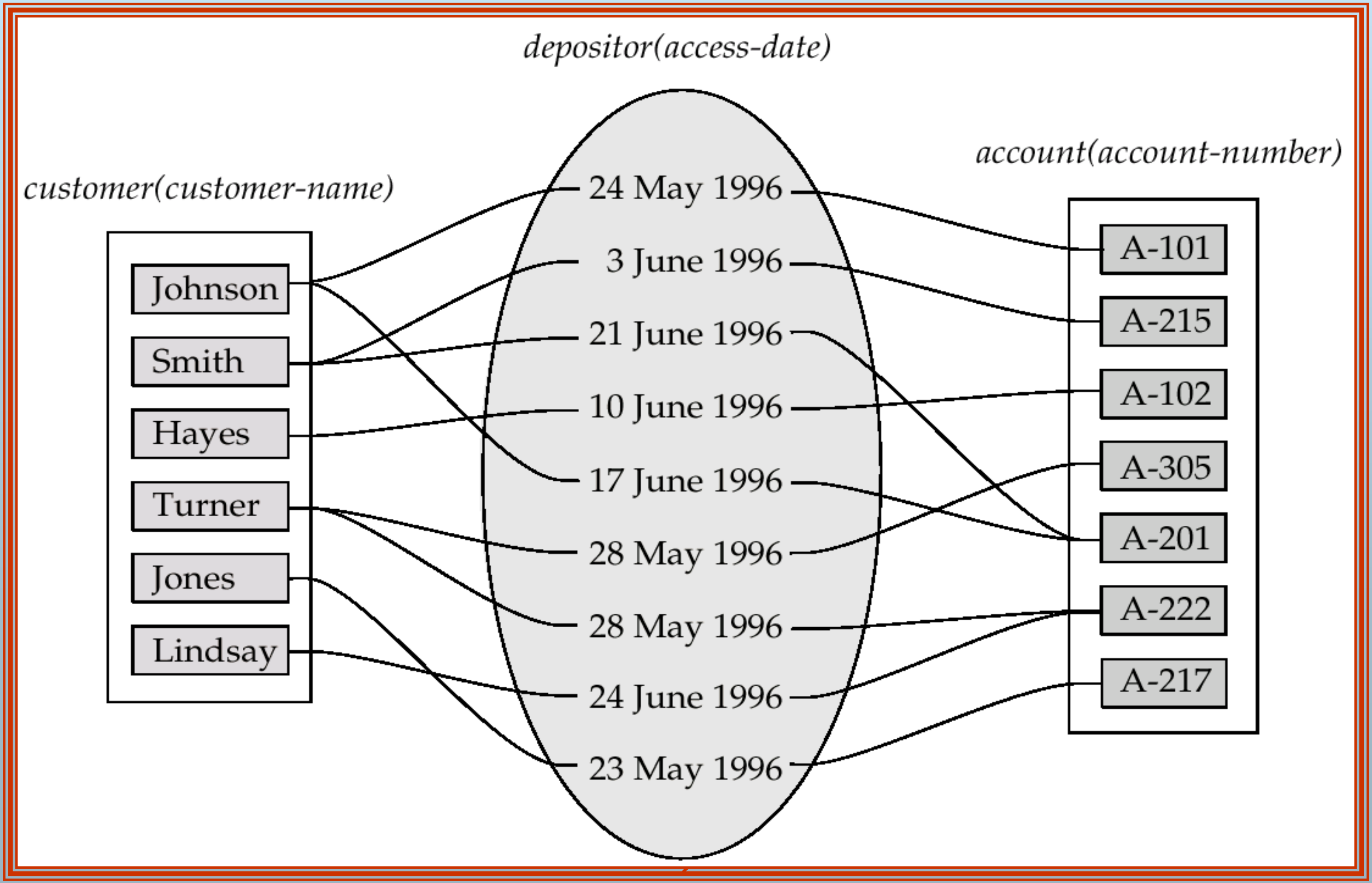


321-12-3123	Jones	Main	Harrison
019-28-3746	Smith	North	Rye
677-89-9011	Hayes	Main	Harrison
555-55-5555	Jackson	Dupont	Woodside
244-66-8800	Curry	North	Rye
963-96-3963	Williams	Nassau	Princeton
335-57-7991	Adams	Spring	Pittsfield

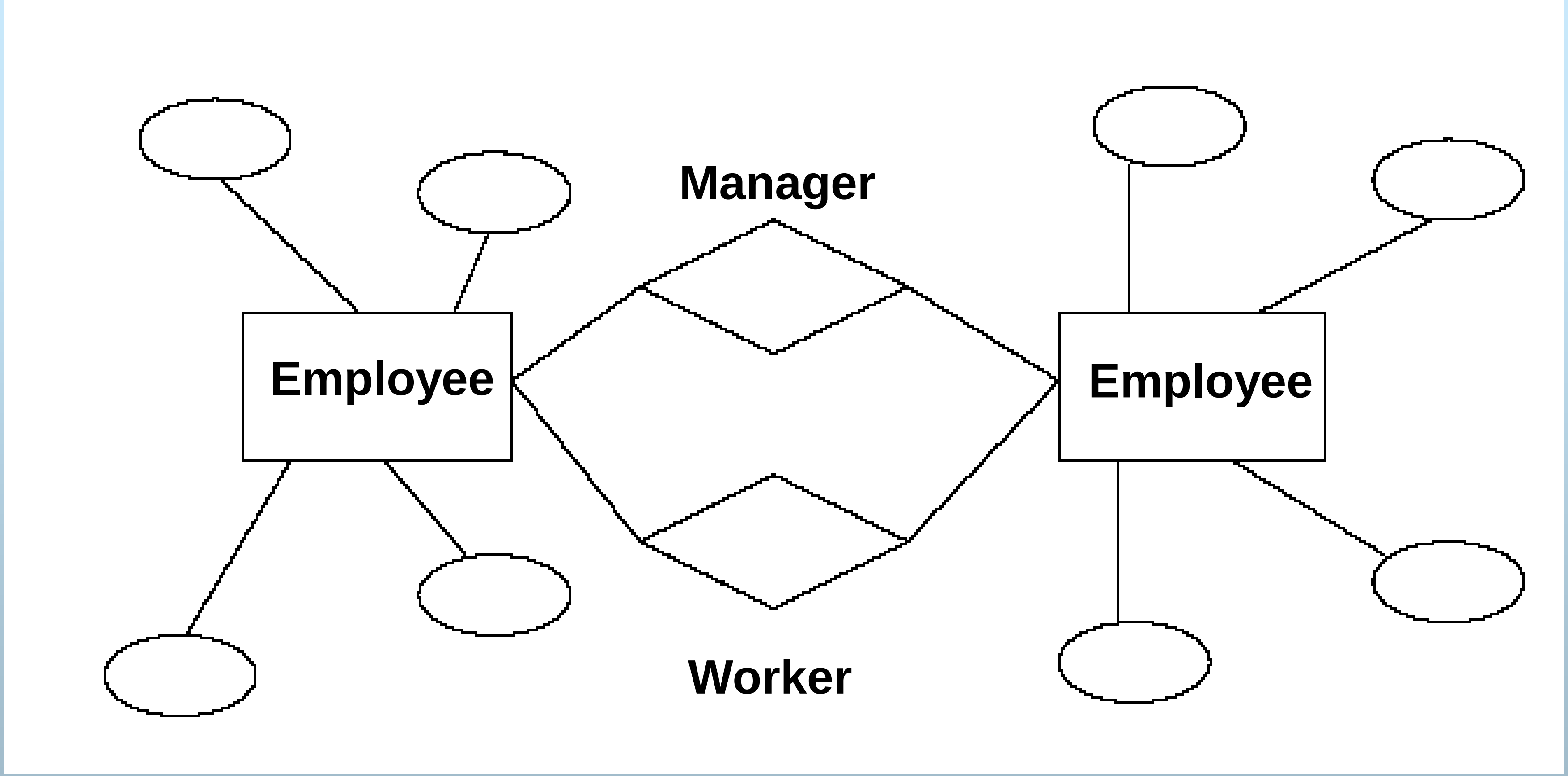
L-17	1000
L-23	2000
L-15	1500
L-14	1500
L-19	500
L-11	900
L-16	1300

Basic Concepts(Cont.)

- **An *attribute*** can also be property of a **relationship set**.
- For instance, the *depositor* relationship set between entity sets *customer* and *account* may have the attribute ***access-date***



Recursive Relationship



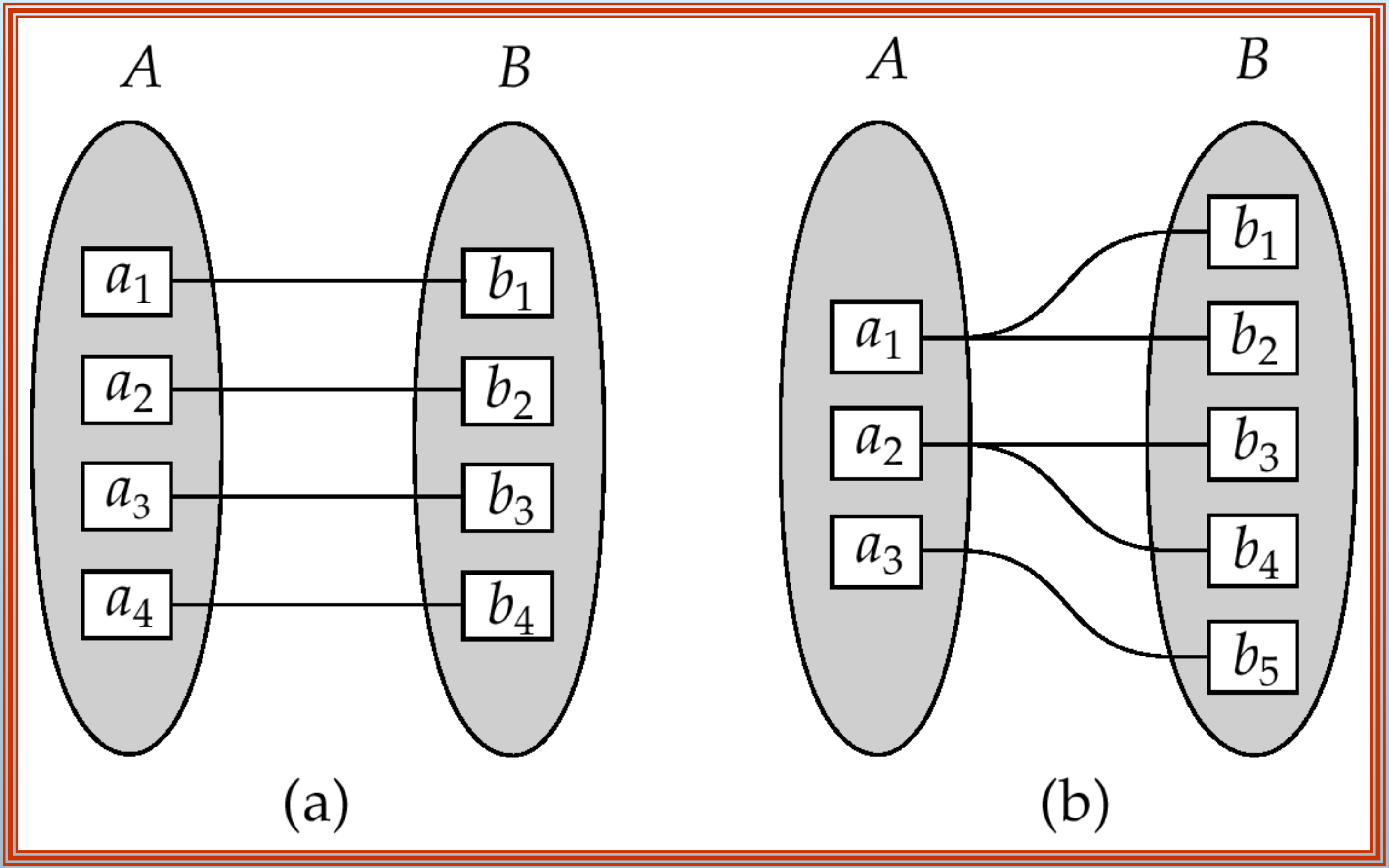
Basic Concepts(Cont.)

- **Degree of a Relationship Set**
  - Refers to number of entity sets that participate in a relationship set.
  - Relationship sets that **involve two entity sets** are ***binary*** (or degree two). Generally, most relationship sets in a database system are binary.
  - Relationship sets may **involve more than two** entity sets.
    - E.g. Suppose employees of a bank may have jobs (responsibilities) at multiple branches, with different jobs at different branches. Then there is **a ternary** relationship set between entity sets *employee, job and branch*
  - Relationships between more than two entity sets are rare. Most relationships are binary. (More on this later.)

Constraints

- **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.**
  - **For a binary relationship set the mapping cardinality must be one of the following types:**
    - **One to one**
    - **One to many**
    - **Many to one**
    - **Many to many**

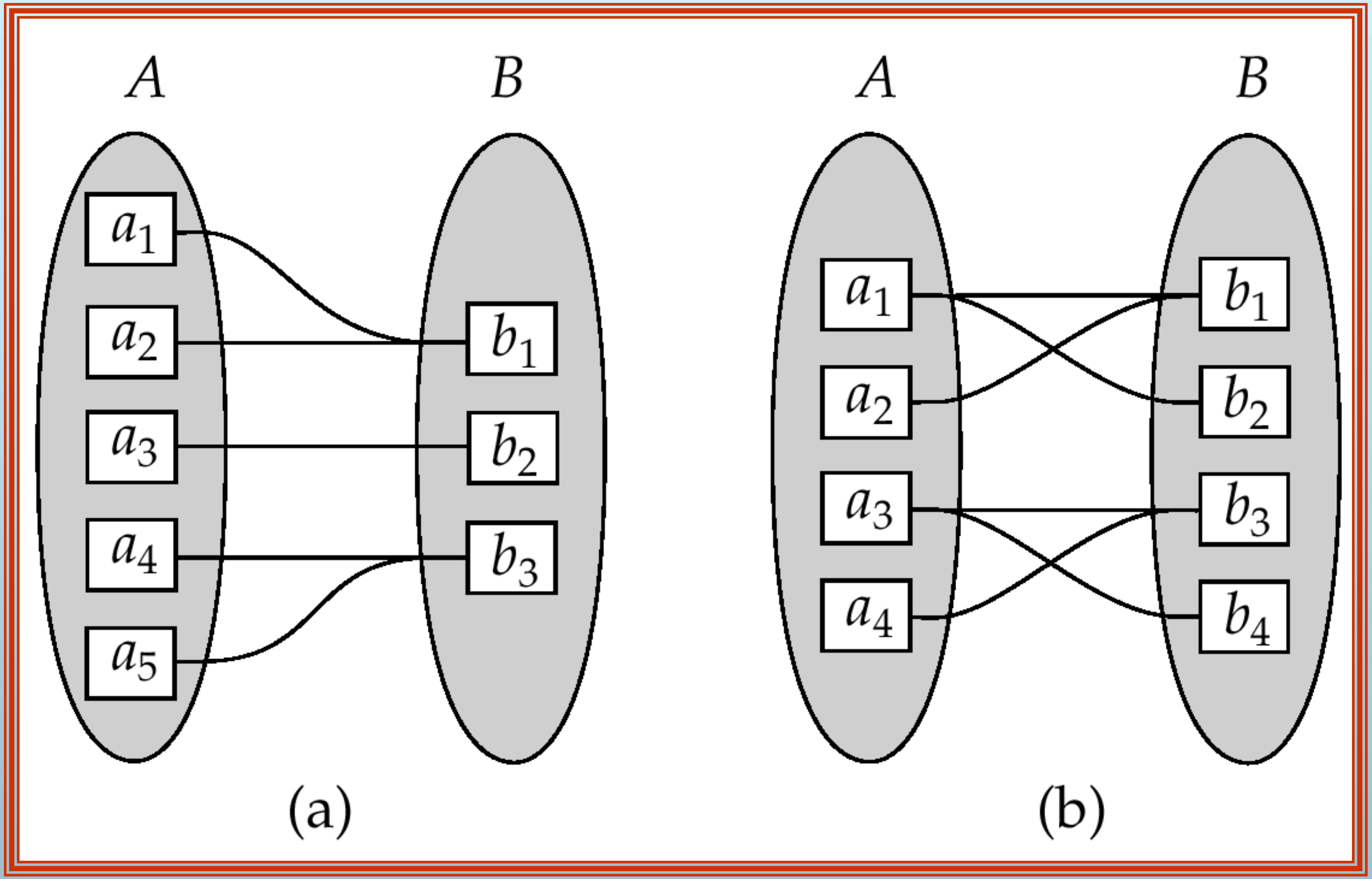
Constraints (Cont.)



One to one One to many

Note: Some elements in A and B may not be mapped to any elements in the other set

Constraints (Cont.)



Many to one   Many to many

Note: Some elements in A and B may not be mapped to any elements in the other set

Constraints (Cont.)

- **Participation Constraints**
  - **The participation of an entity set *E* in a relationship set *R* is said to be total if every entity in *E* participates in at least one relationship in *R*.**
    - **For example, we expect every loan entity to be related to at least one customer through the borrower relationship.**
  - **If only some entities in E participate in relationship in R, the participation of entity set E in relationship R is said to be partial.**
    - **For example, the participation of customer in the borrower relationship set is therefore set is therefore partial.**

Keys

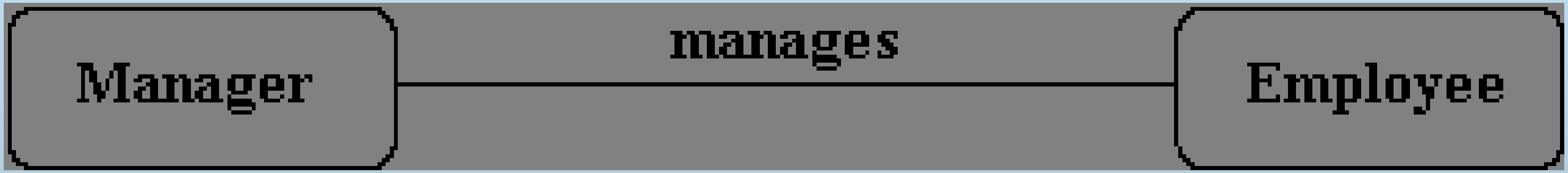
- **Entity Sets**
  - A ***super key*** of an entity set is a set of one or more attributes whose values uniquely determine each entity.
  - A ***candidate key*** of an entity set is a minimal super key
    - *Customer-id* is candidate key of *customer*
    - *account-number* is candidate key of *account*
  - Although several candidate keys may exist, one of the candidate keys is selected to be the *primary key*.
- **Relationship Sets**
  - The combination of primary keys of the participating entity sets forms a super key of a relationship set.

Keys (Cont.)

- (*customer-id, account-number*) is the super key of *depositor*
- *NOTE: this means a pair of entity sets can have at most one relationship in a particular relationship set.*
  - E.g. if we wish to track all access-dates to each account by each customer, we cannot assume a relationship for each access. We can use a multivalued attribute though
- Must consider the mapping cardinality of the relationship set when deciding the what are the candidate keys
- Need to consider semantics of relationship set in selecting the *primary key* in case of more than one candidate key

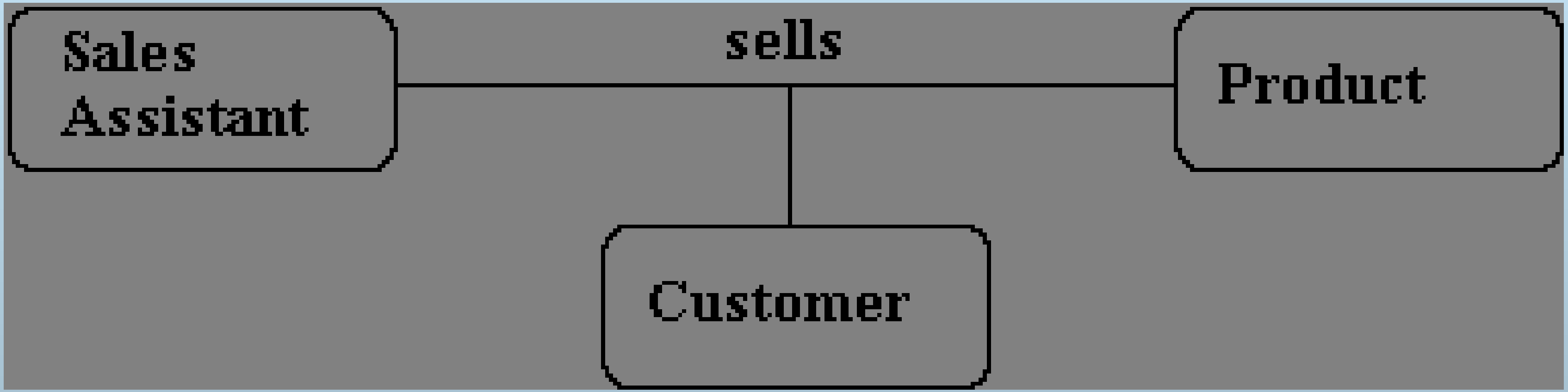
Binary Relationships

- If there are two entity types involved it is a *binary* relationship type



Ternary relationship

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



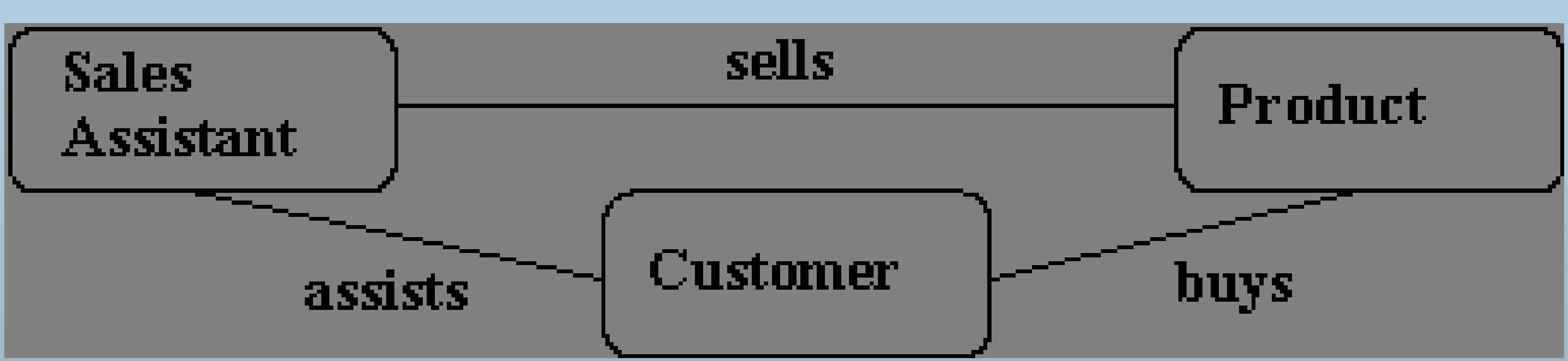
Replacing ternary relationships

- When ternary relationships occurs in an ER model they should always be removed before finishing the model.
- Sometimes the relationships can be replaced by a series of binary relationships that link pairs of the original ternary relationship.

Replacing ternary relationships (Cont)

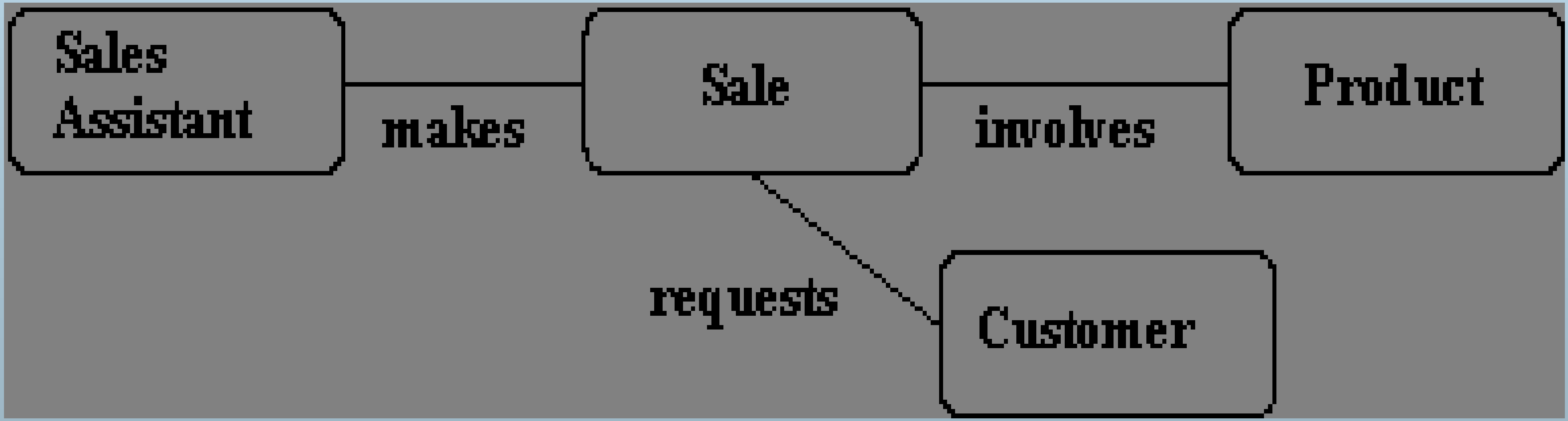
This can result in the loss of some information - It is no longer clear which sales assistant sold a customer a particular product.

Try replacing the ternary relationship with an entity type and a **set of binary** relationships.

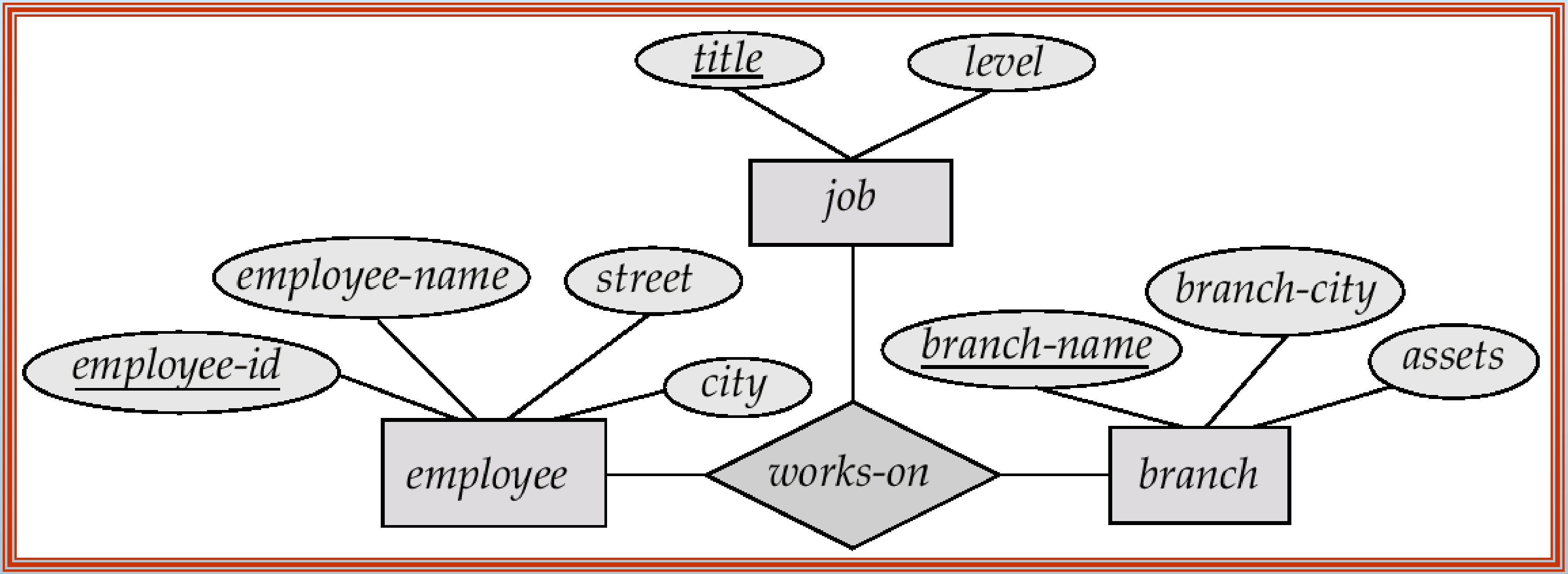


Replacing ternary relationships (Cont)

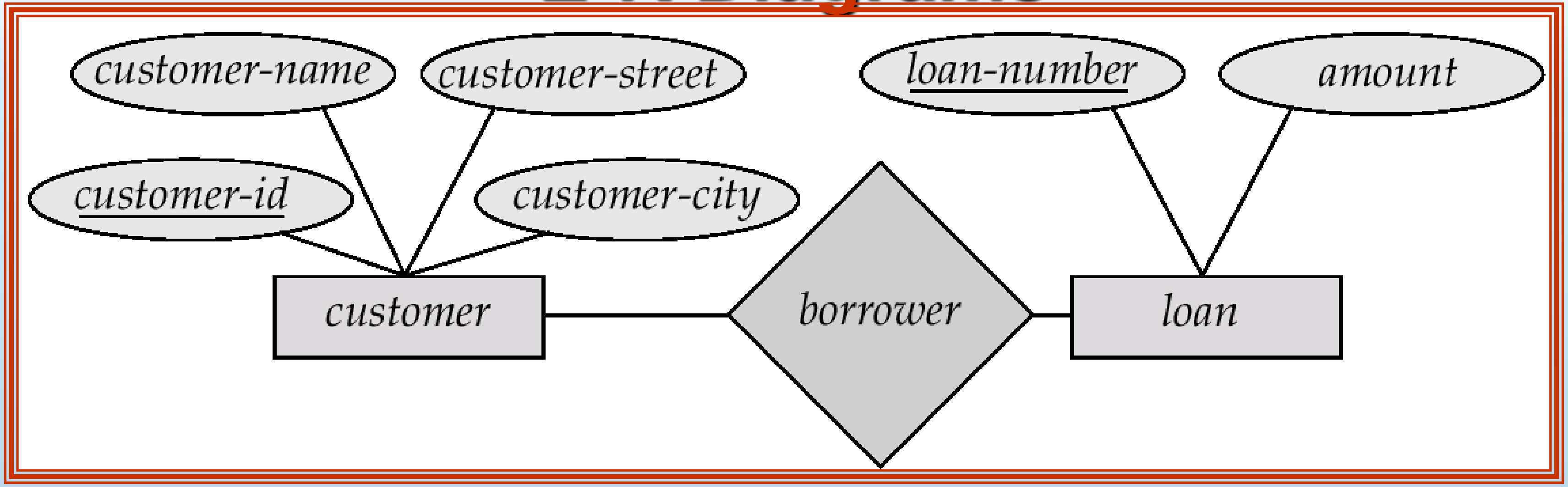
- Relationships are usually verbs, so name the new entity type by the relationship verb rewritten as a noun.
- The relationship *sells* can become the entity type *sale*.
- So a sales assistant can be linked to a specific customer and both of them to the sale of a particular product.
- This process also works for higher order relationships.



E-R Diagram with a Ternary Relationship

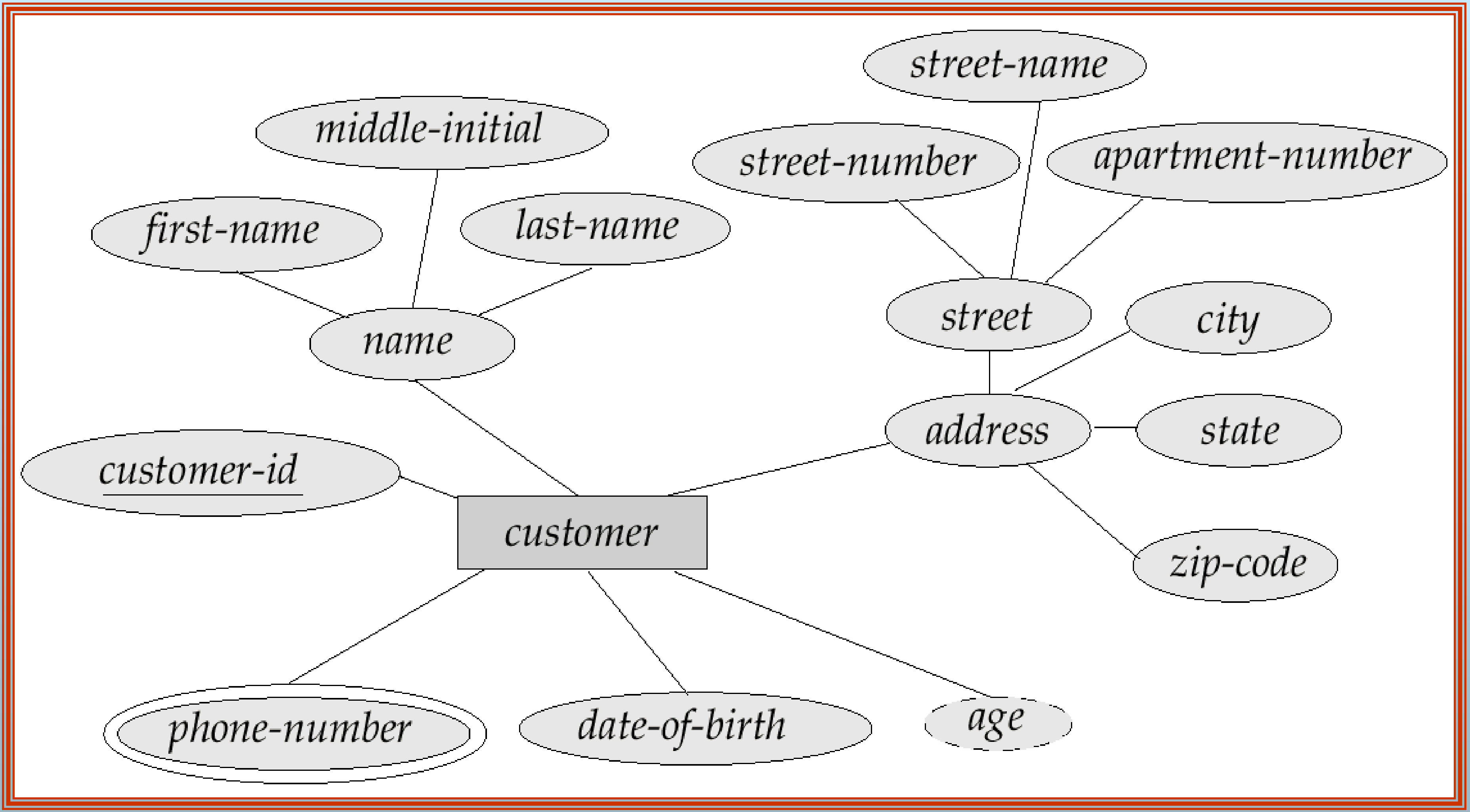


E-R Diagrams

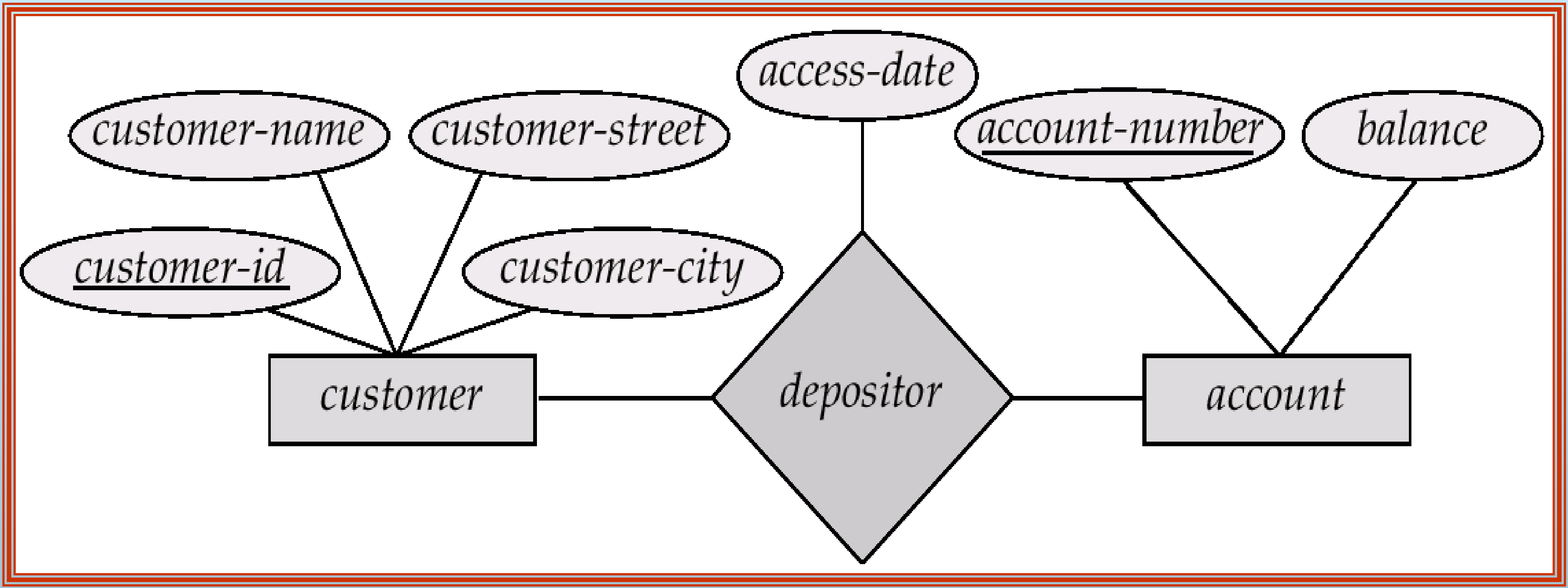


- **Rectangles** represent entity sets.
- **Diamonds** represent relationship sets.
- **Lines** link attributes to entity sets and entity sets to relationship sets.
- **Ellipses** represent attributes
  - **Double ellipses** represent multivalued attributes.
  - **Dashed ellipses** denote derived attributes.
- **Underline** indicates primary key attributes

E-R Diagram With Composite, Multivalued, and Derived Attributes

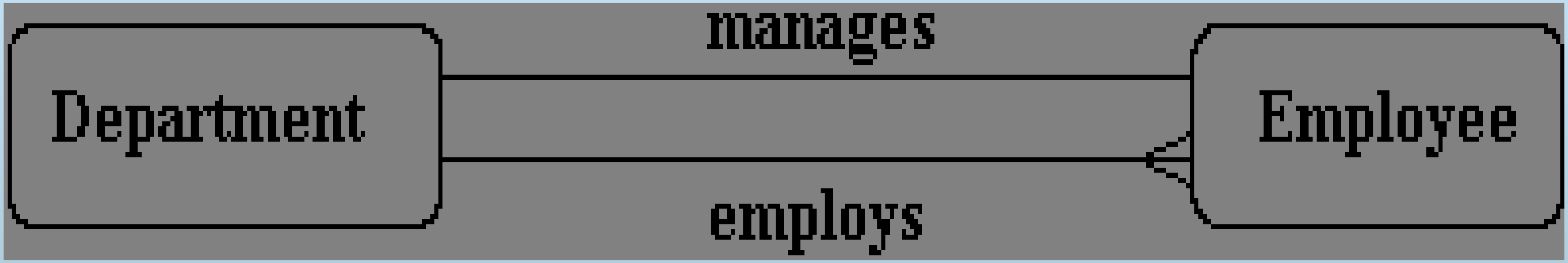


Relationship Sets with Attributes



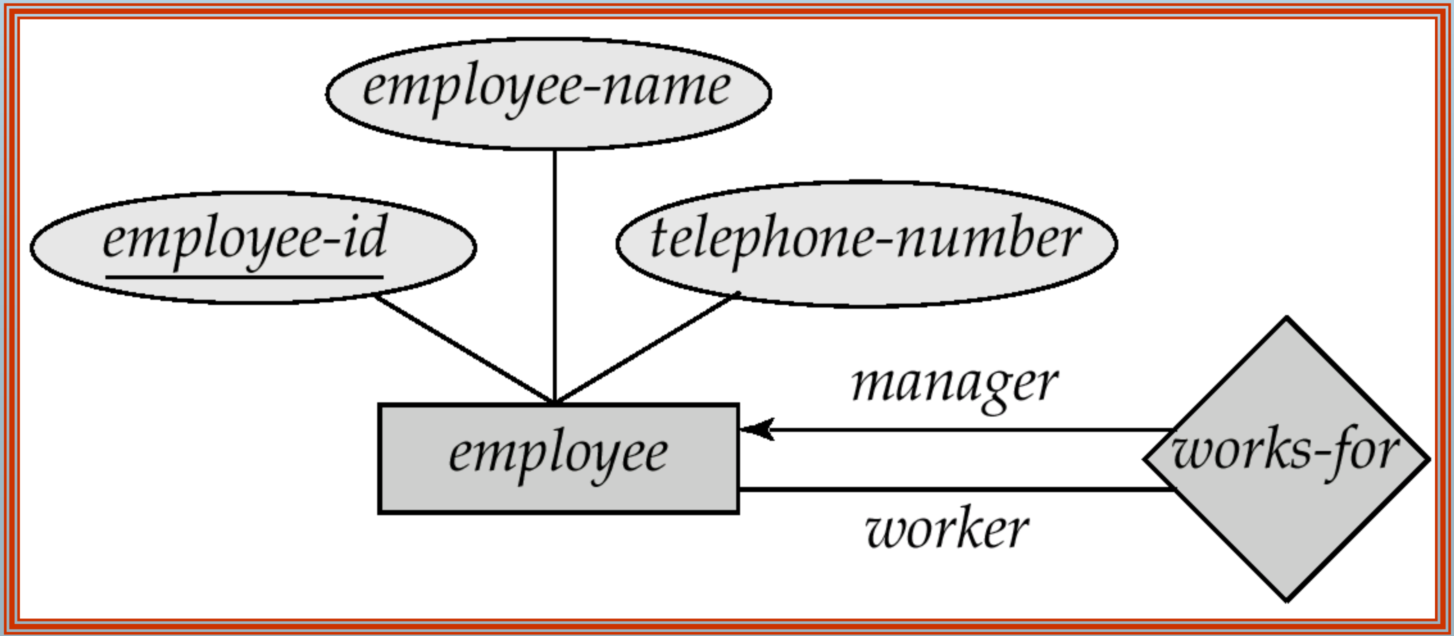
Degree of a Relationship

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



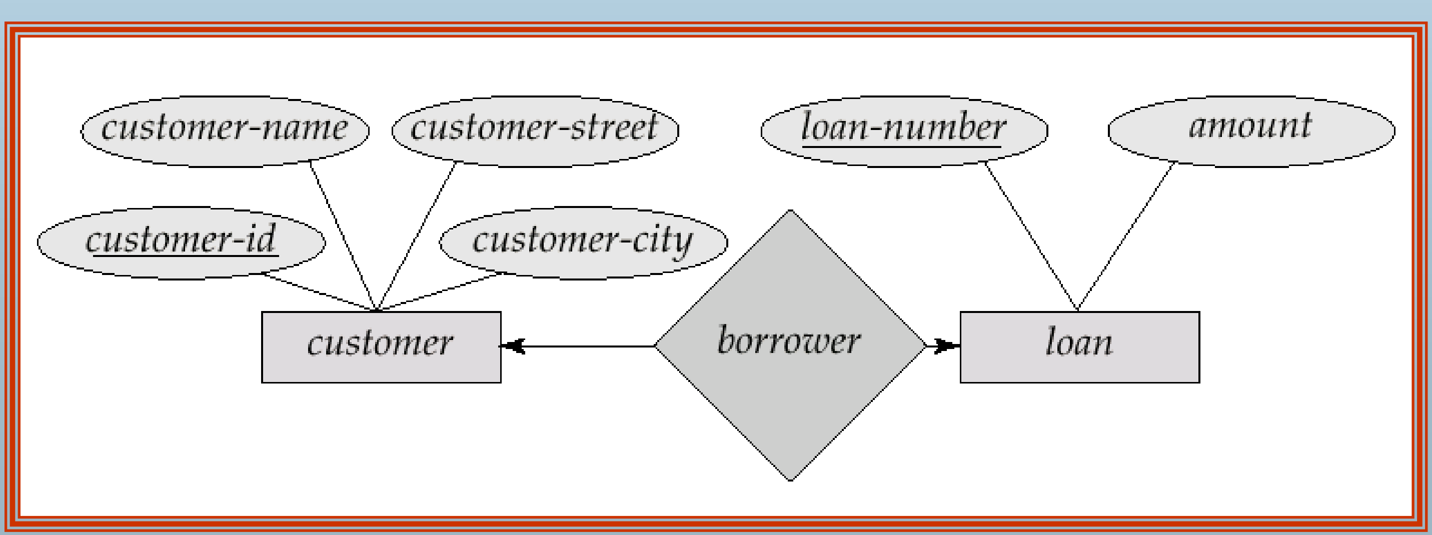
Roles: Degree of a Relationship

- The labels “manager” and “worker” are called roles; they specify how employee entities interact via the works-for relationship set.
- Roles are indicated in E-R diagrams by labeling the lines that connect diamonds to rectangles.
- Role labels are optional, and are used to clarify semantics of the relationship



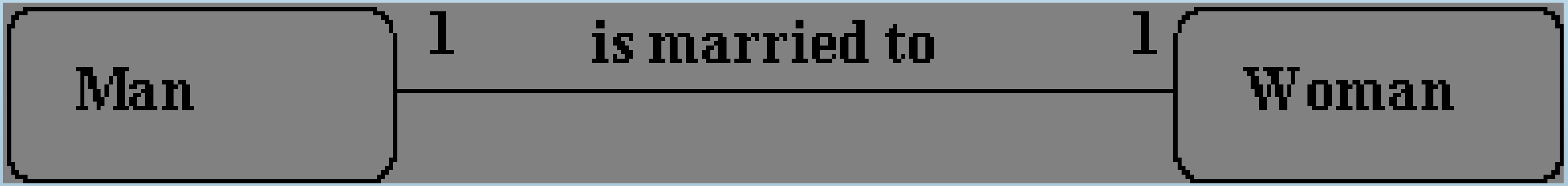
Cardinality Constraints

- We express cardinality constraints by drawing either a directed line (→), signifying "one," or an undirected line (—), signifying "many," between the relationship set and the entity set.
- E.g.: One-to-one relationship:
  - A customer is associated with at most one loan via the relationship *borrower*
  - A loan is associated with at most one customer via *borrower*



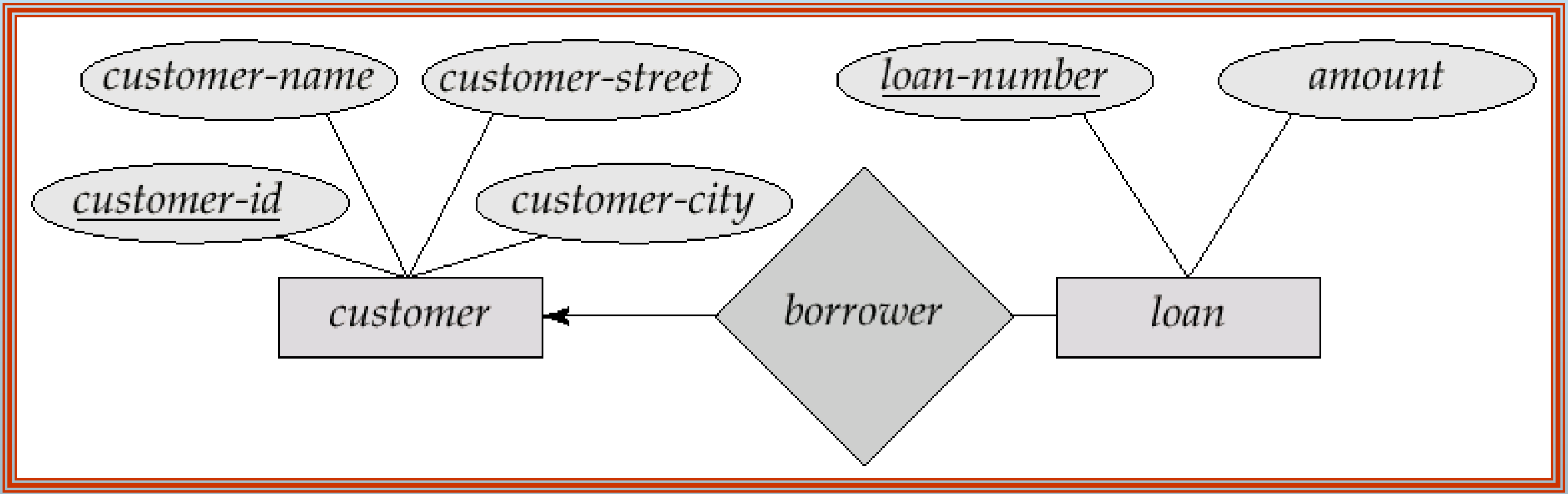
One-To-One Relationship

- A man can only marry one woman, and a woman can only marry one man, so it is a one to one (1:1) relationship



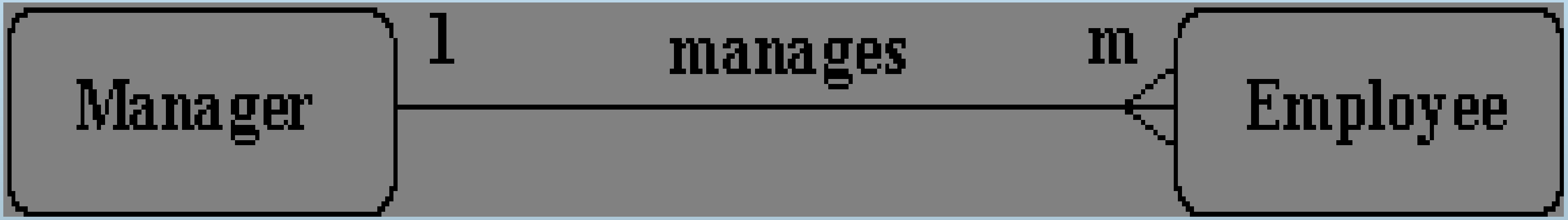
One-To-Many Relationship

- In the one-to-many relationship a loan is associated with at most one customer via *borrower*, a customer is associated with several (including 0) loans via *borrower*



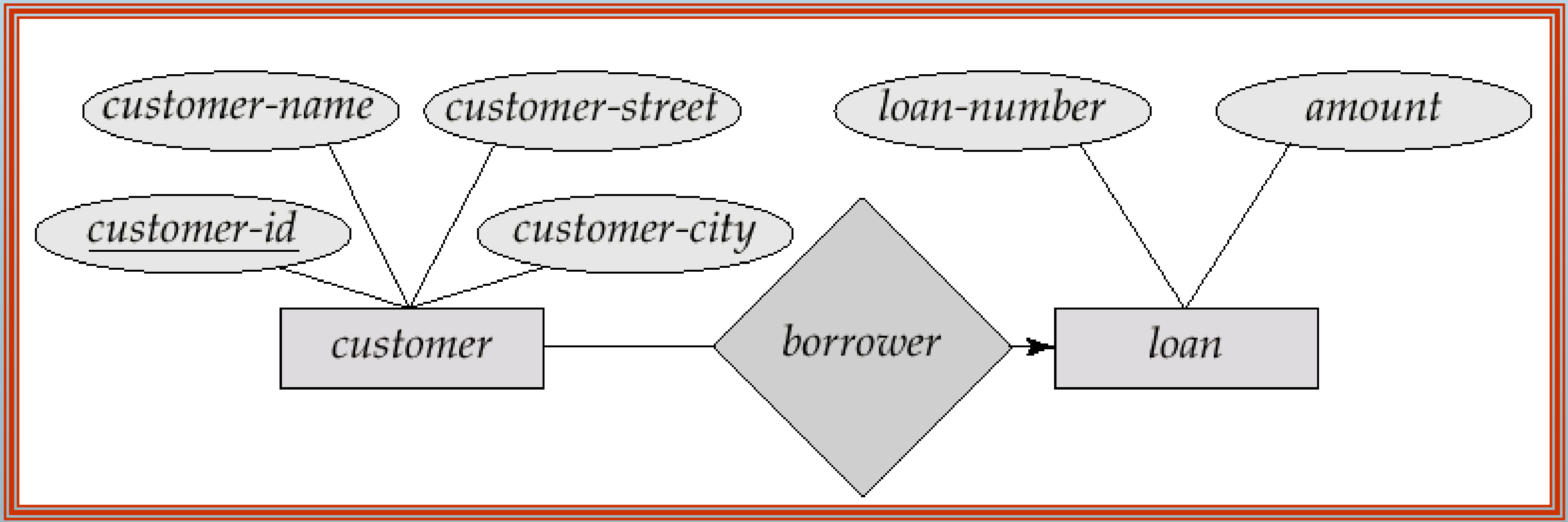
One-To-Many Relationship

- One manager manages many employees, but each employee only has one manager, so it is a one to many (1:n) relationship



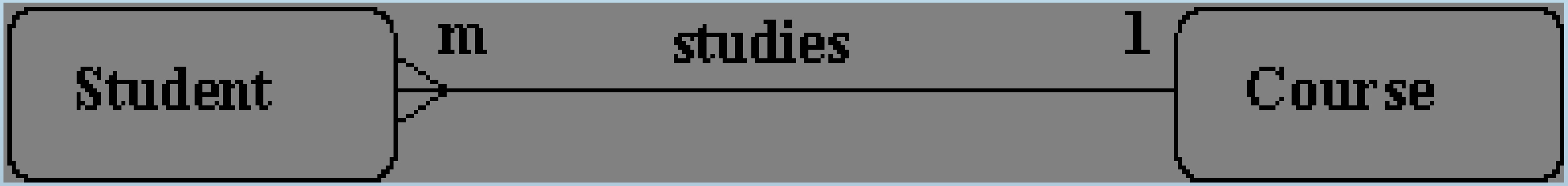
Many-To-One Relationships

- In a many-to-one relationship a loan is associated with several (including 0) customers via *borrower*, a customer is associated with at most one loan via *borrower*

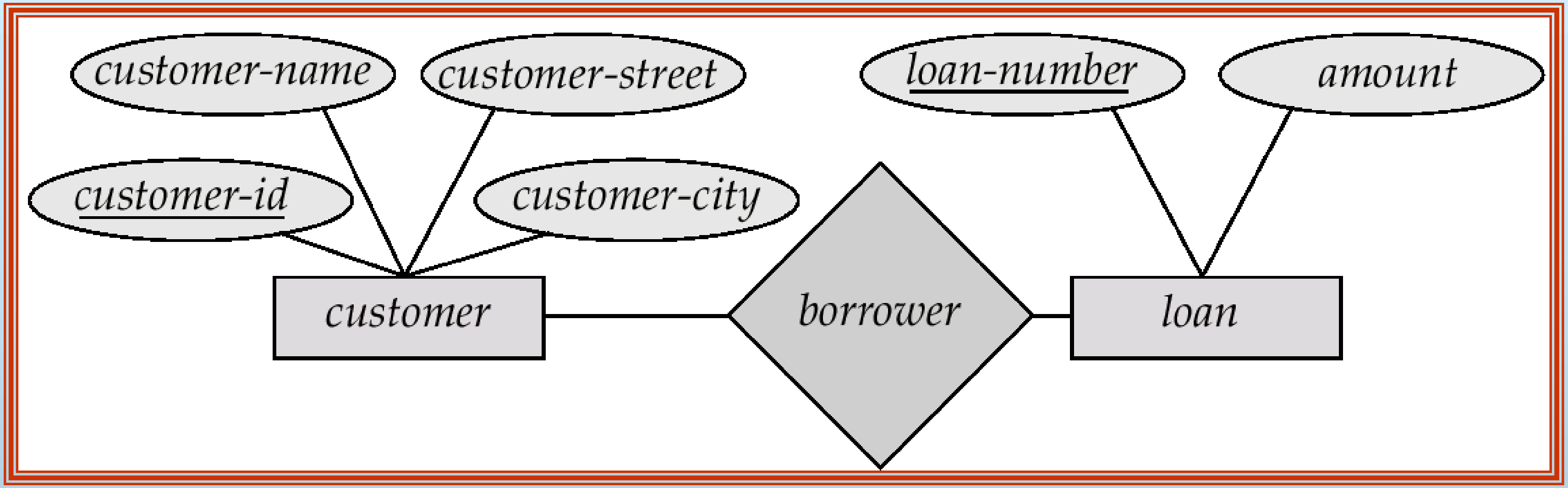


Many-To-One Relationships

- many students study one course. They do not study more than one course, so it is a many to one (m:1) relationship



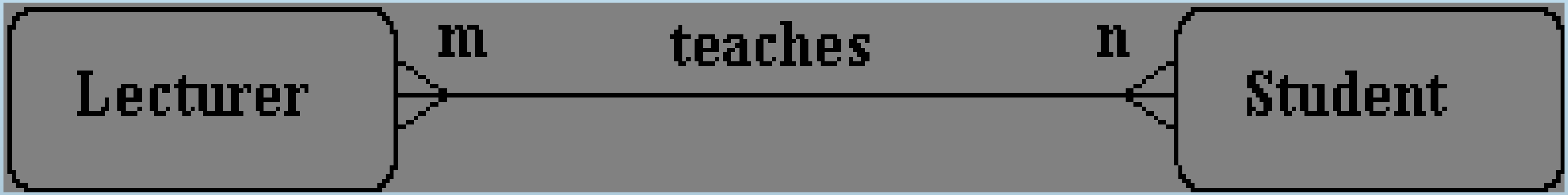
Many-To-Many Relationship



- A customer is associated with several (possibly 0) loans via borrower
- A loan is associated with several (possibly 0) customers via borrower

Many-To-Many Relationship

- One lecturer teaches many students and a student is taught by many lecturers, so it is a many to many (m:n) relationship



Deriving the relationship parameters

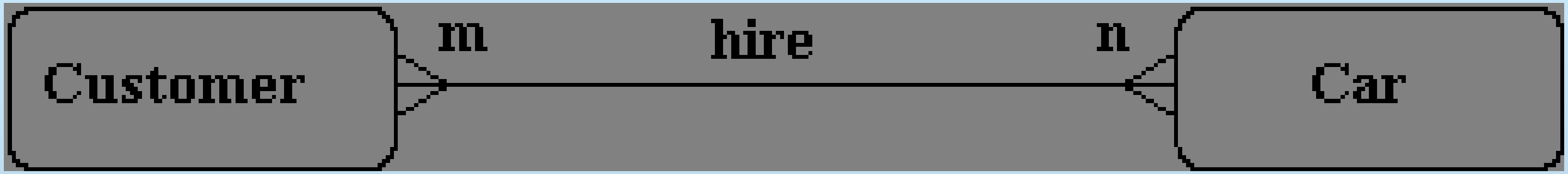
- To check we have the correct parameters (sometimes also known as the **degree**) of a **relationship**, ask two questions:
- **One course** is studied by **how many** students? **Answer = `zero or more'**
  - This gives us the **degree at the `student' end**
  - **The `more'** part means that the cardinality **is `many**
  - The `**zero'** part means that the relationship is `**optional'**. (denoted y ‘O”
  - If the answer was `**one or more'**, then the relationship would be `**mandatory'.**
- **One student** studies **how many** courses? **Answer = `One'**
  - This gives us **the degree at the `course' end** of the relationship**.**
  - The answer `**one'** means that the cardinality of this relationship is 1, and is `**mandatory'**

Splitting n:m Relationships

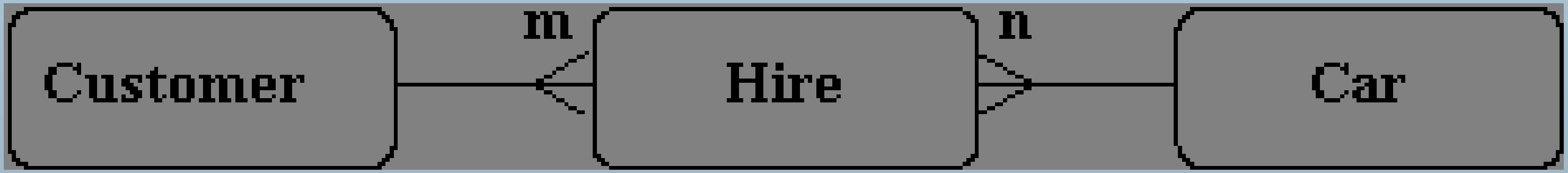
- A many to many relationship in an ER model is not necessarily incorrect. They can be replaced using an intermediate entity. This should only be done where:
  - The m:n relationship hides an entity
  - the resulting ER diagram is easier to understand.

- **Example**: Consider the case of a car hire company.
  - Customers hire cars
  - One customer hires many card and
  - A car is hired by many customers.

Splitting n:m Relationships

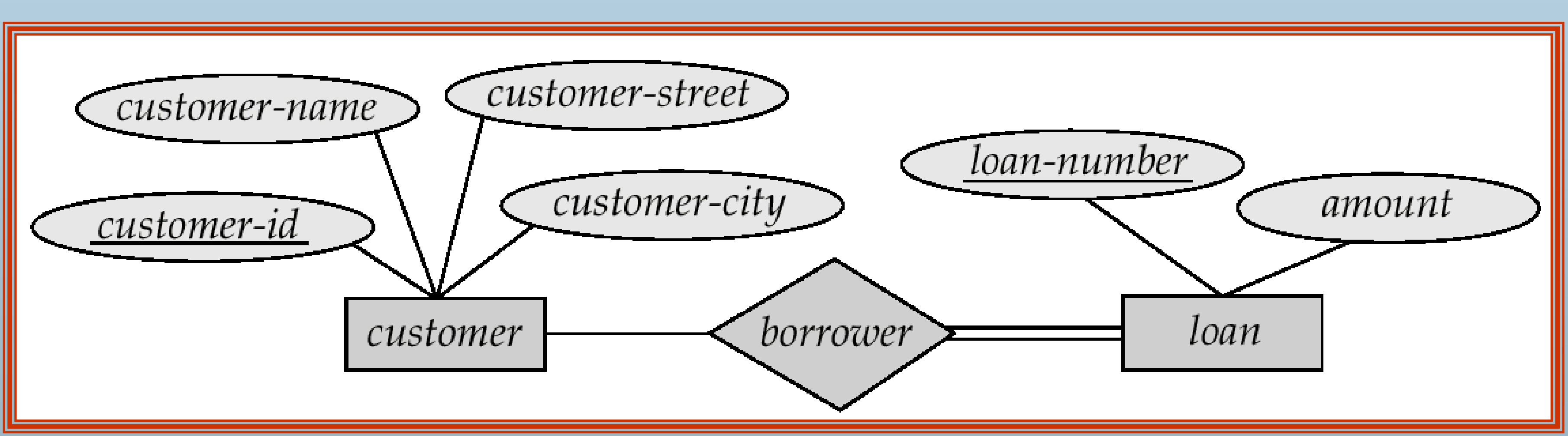


- The many to many relationship can be broken down to reveal a **`hire' entity**, which contains an attribute **`date of hire'.**



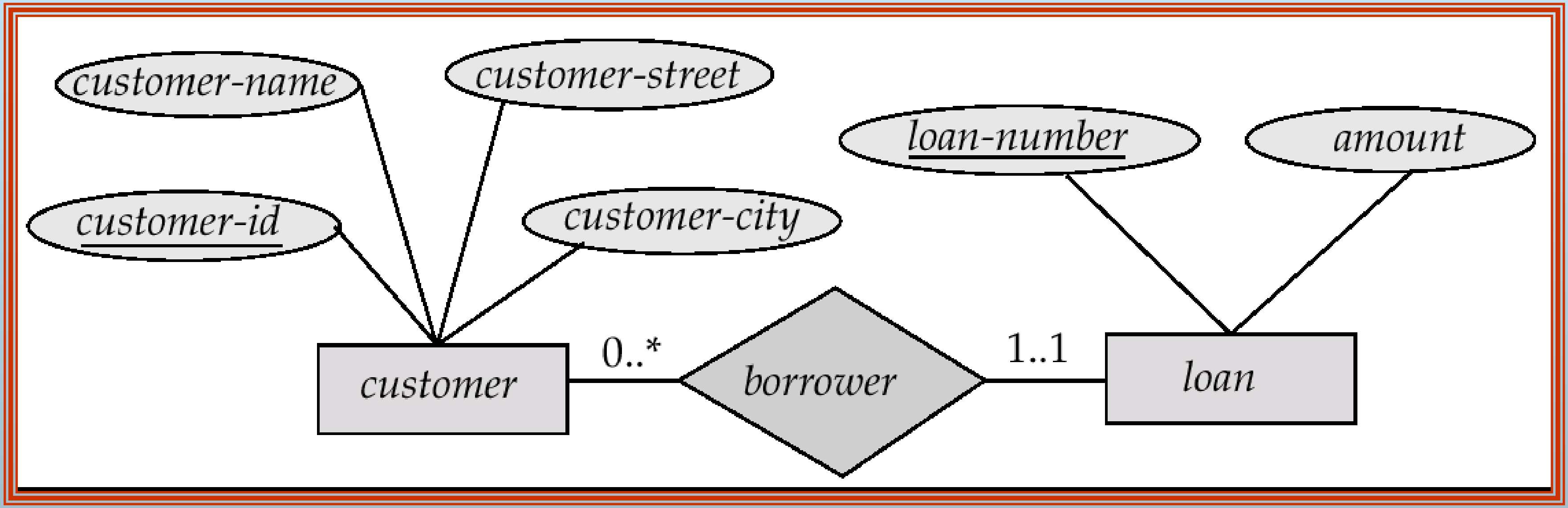
Participation of an Entity Set in a Relationship Set

- **Total participation** (indicated by double line): every entity in the entity set participates in at least one relationship in the relationship set
  - E.g. participation of *loan* in *borrower* is total
    - every loan must have a customer associated to it via borrower
- **Partial participation**: some entities may not participate in any relationship in the relationship set
  - E.g. participation of *customer* in *borrower* is partial



Alternative Notation for Cardinality Limits

- Cardinality limits can also express participation constraints



Design Issues

- Use of entity sets vs. attributes
  - A common mistake is to use the **primary key of an entity set as another entity set, instead of using a relationship**.
  - Another related mistake that people sometimes make is to designate the **primary key attributes** of the related entity sets as **attributes of the relationship set.**

Design Issues (Cont.)

- Use of entity sets vs. relationship sets
  - We assumed that a **bank loan** is modeled as an **entity**.
  - An alternative is to model a loan not as **an entity**,
    - but rather as a relationship between **customers** and **branches,**
    - with ***loan-number*** and ***amount* as descriptive attributes.**

Design Issues (Cont.)

- Binary versus *n*-ary relationship sets

  Some relationships that appear to be non-binary may be better represented using binary relationships

  - E.g. A **ternary** relationship
    - *parents*, relating a child to his/her **father and mother**
    - is best replaced by **two binary relationships**, *father* and *mother*
    - Using two binary relationships allows partial information (e.g. only mother being know)

Design Issues (Cont.)

- But there there are some relationships that are naturally non-binary
  - E.g. *works-on*
- In general, any non-binary relationship can be represented using **binary relationships** by creating an **artificial entity set**.
- Replace ***R*** between **entity sets A, B and C** by an entity **set *E***, and three relationship sets:

1. R_A, relating E and A

2. R_B, relating E and B

3. R_C, relating E and C

- Create a special identifying attribute for *E*
- Add any attributes of *R* to *E*
- For each relationship (a_i , b_i , c_i) in *R*, create e_i

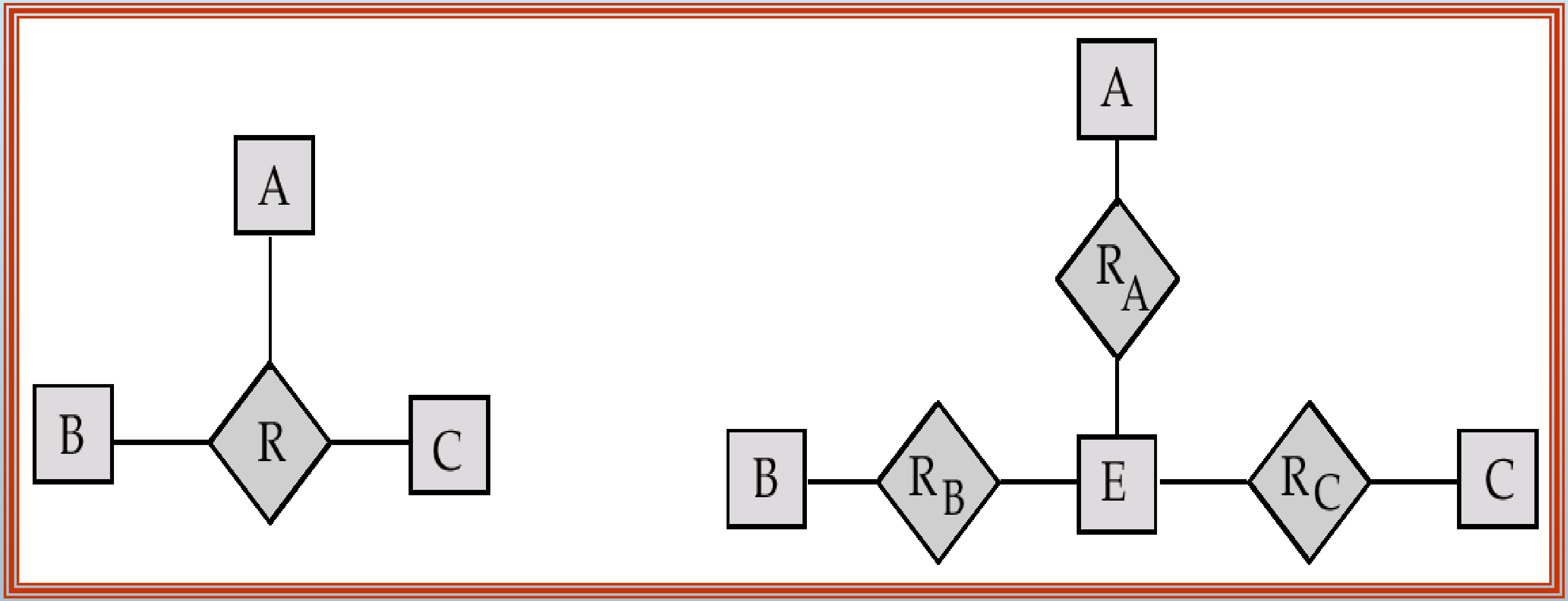
Design Issues (Cont.)

1. a new entity e_i in the entity set E

2. add (e_i , a_i) to R_A

3. add (e_i , b_i) to R_B

4. add (e_i , c_i) to R_C



Design Issues (Cont.)

- Placement of relationship attributes

Can make **access-date** an attribute of account, instead of a **relationship attribute**, if each account can have only one customer

- I.e., the relationship from **account to customer is many to one**, or equivalently, customer to **account is one to many**

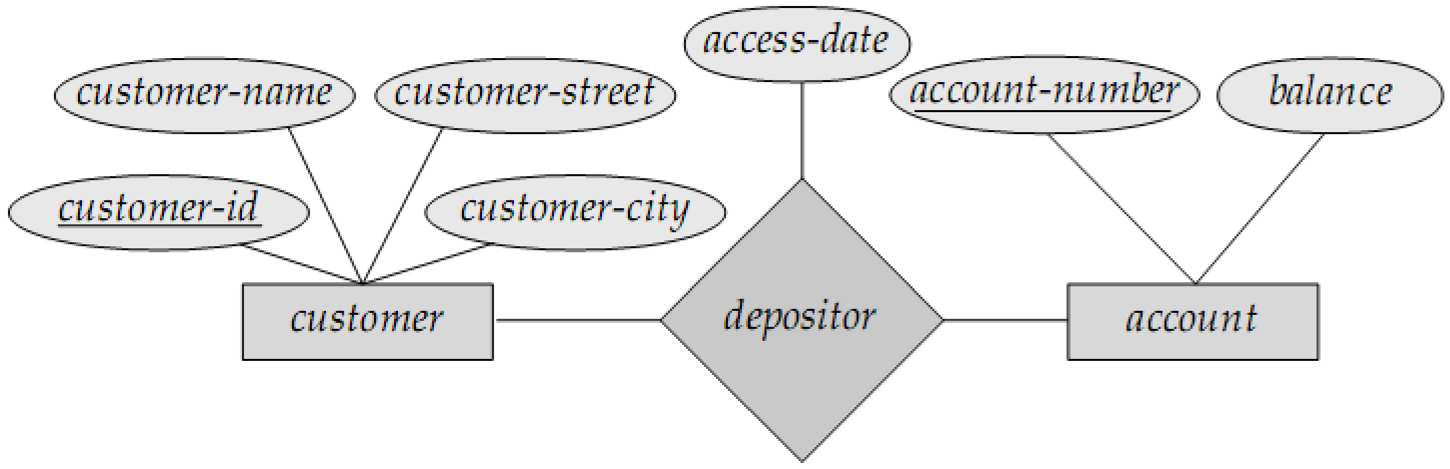


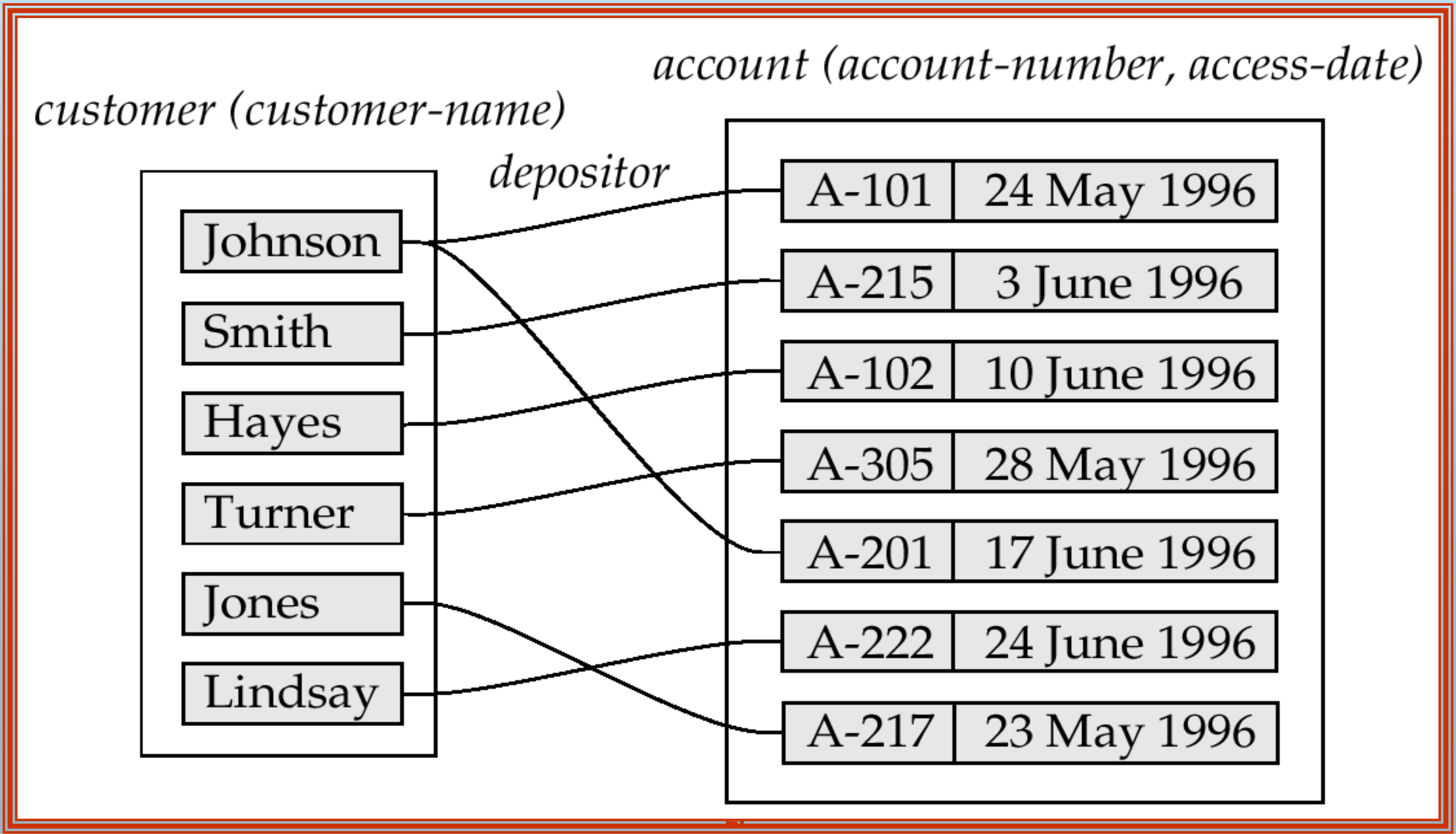
Figure 2.10 E-R diagram with an attribute attached to a relationship set.

Design Issues (Cont.)

- Placement of relationship attributes

  Can make **access-date** an attribute of account, instead of a **relationship attribute**, if each account can have only one customer

  - I.e., the relationship from **account to customer is many to one**, or equivalently, customer to **account is one to many**



account (account-number, access-date)

customer (customer-name)

depositor

Johnson
Smith
Hayes
Turner
Jones
Lindsay

A-101	24 May 1996
A-215	3 June 1996
A-102	10 June 1996
A-305	28 May 1996
A-201	17 June 1996
A-222	24 June 1996
A-217	23 May 1996

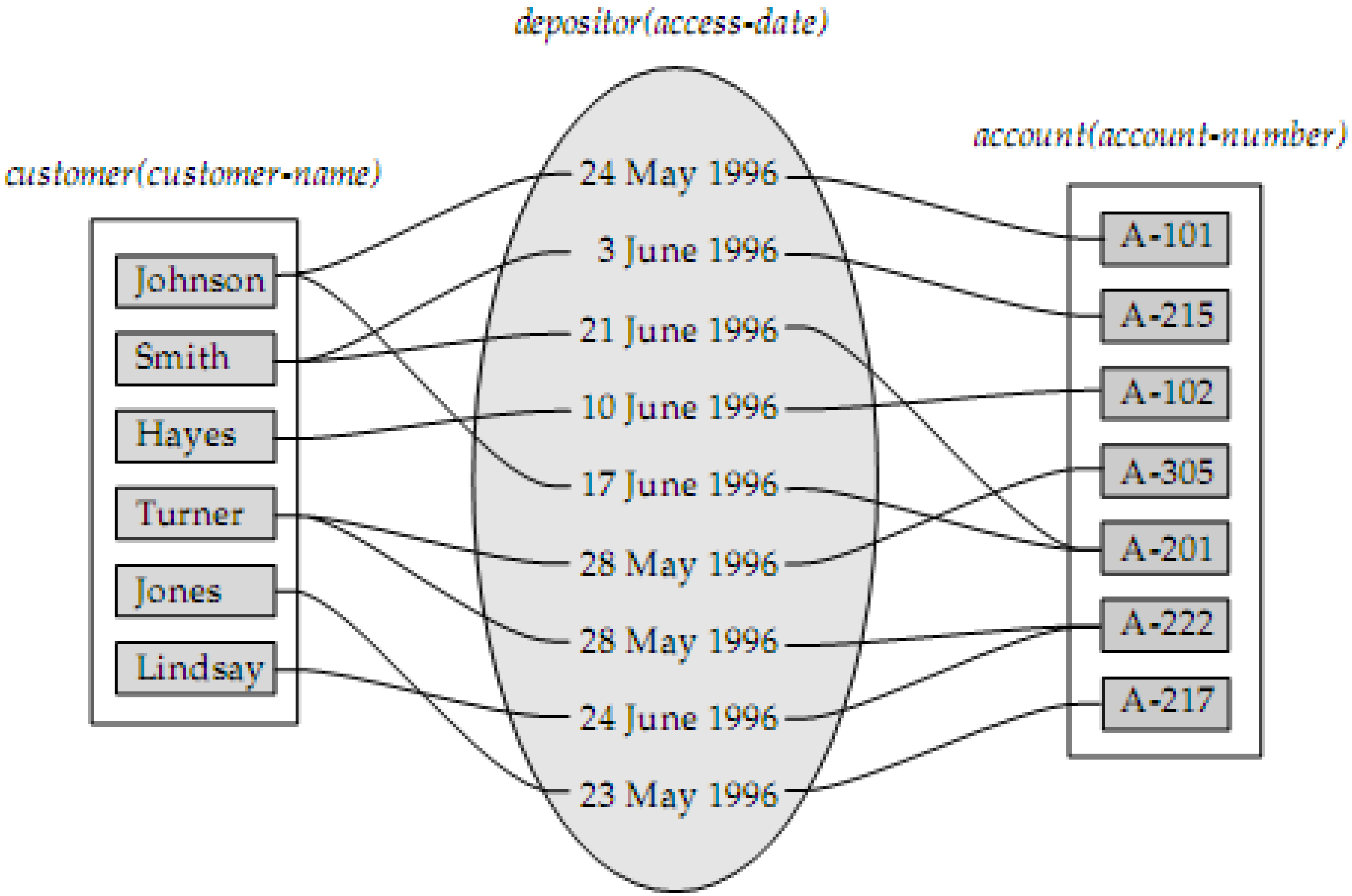


Figure 2.7 *Access-date* as attribute of the *depositor* relationship set.