CS & IT College
2020/2021 Semester 1

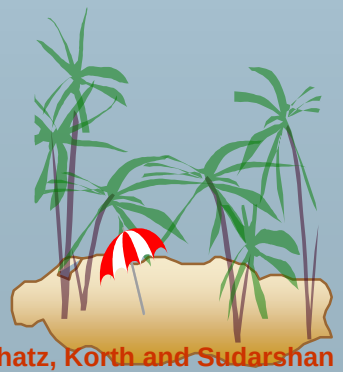
IS203 Database Principals

Chapter 2: Entity-Relationship Model

Asst Prof. Asaad Alhijaj

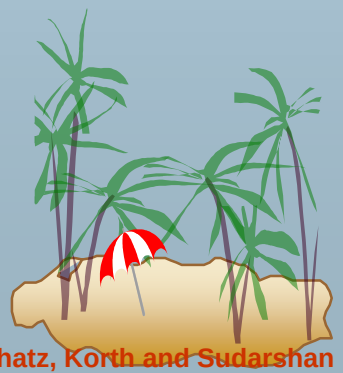
Reference:

“Database System Concepts Fourth Edition” by Abraham Silberschatz Henry F. Korth S. Sudarshan , McGraw-Hill ISBN 0-07-255481-9



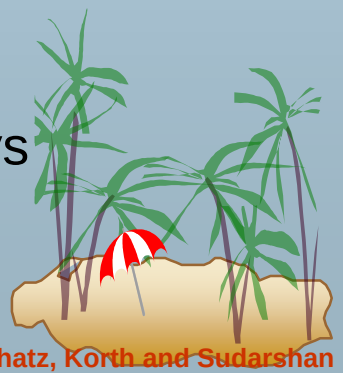
Chapter 2: Entity-Relationship Model

- Entity Sets
- Relationship Sets
- Design Issues
- **Mapping** Constraints
- **Keys**
- **E-R** Diagram
- Design of an E-R Database Schema
- **UML**: Unified Modeling Language
- Reduction of an E-R Schema to **Tables**



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

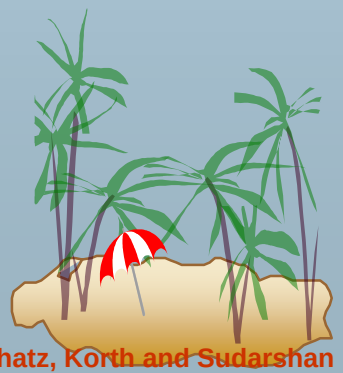
Attributes

- An entity is represented by a set of ***attributes***, that is descriptive properties possessed by all members of an entity set.

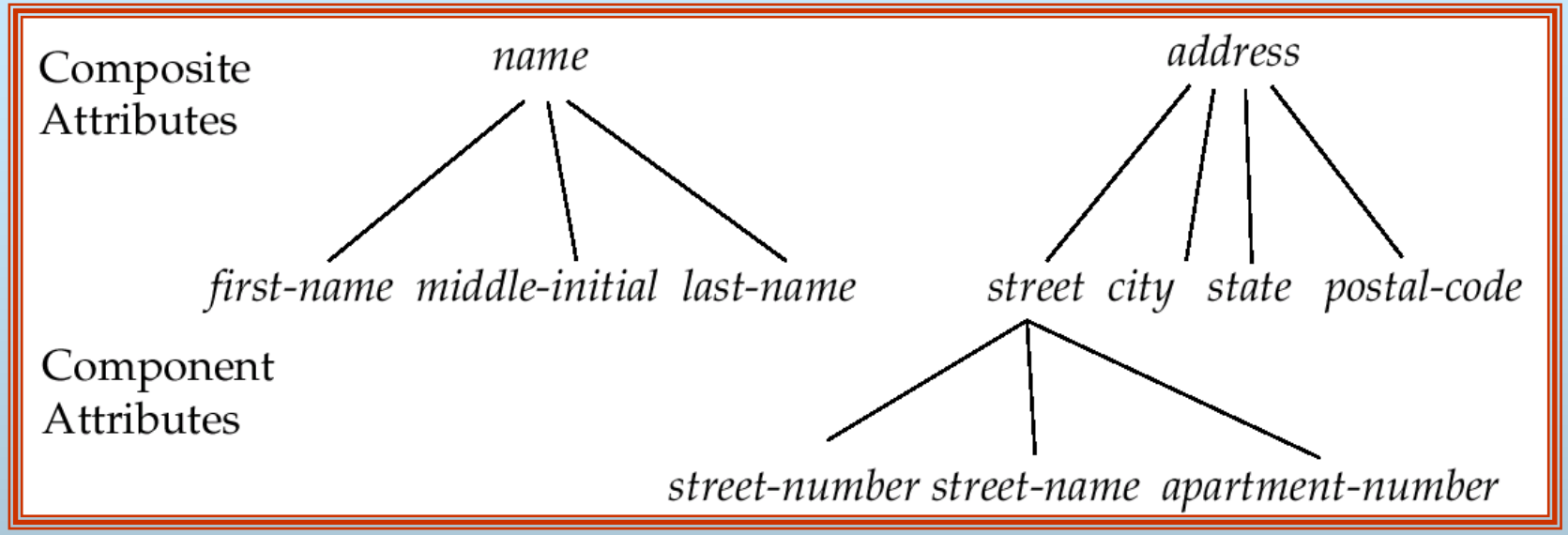
  Example:

  customer = (customer-id, customer-name, customer-street, customer-city)
  loan = (loan-number, amount)

- ***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



Relationship Sets

- A ***relationship*** is an association among several entities

  Example:

Hayes	*depositor(يودع)*	A-102
customer entity	relationship set	*account* entity

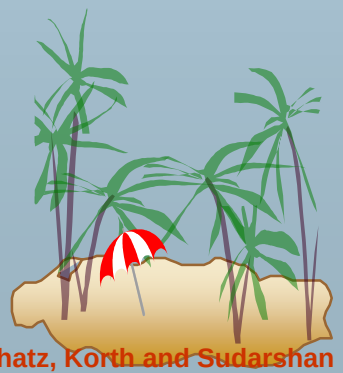
- A ***relationship set*** is a mathematical relation among $n \geq 2$ entities, each taken from entity sets علاقة رياضية بين كينونتين على الاقل

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

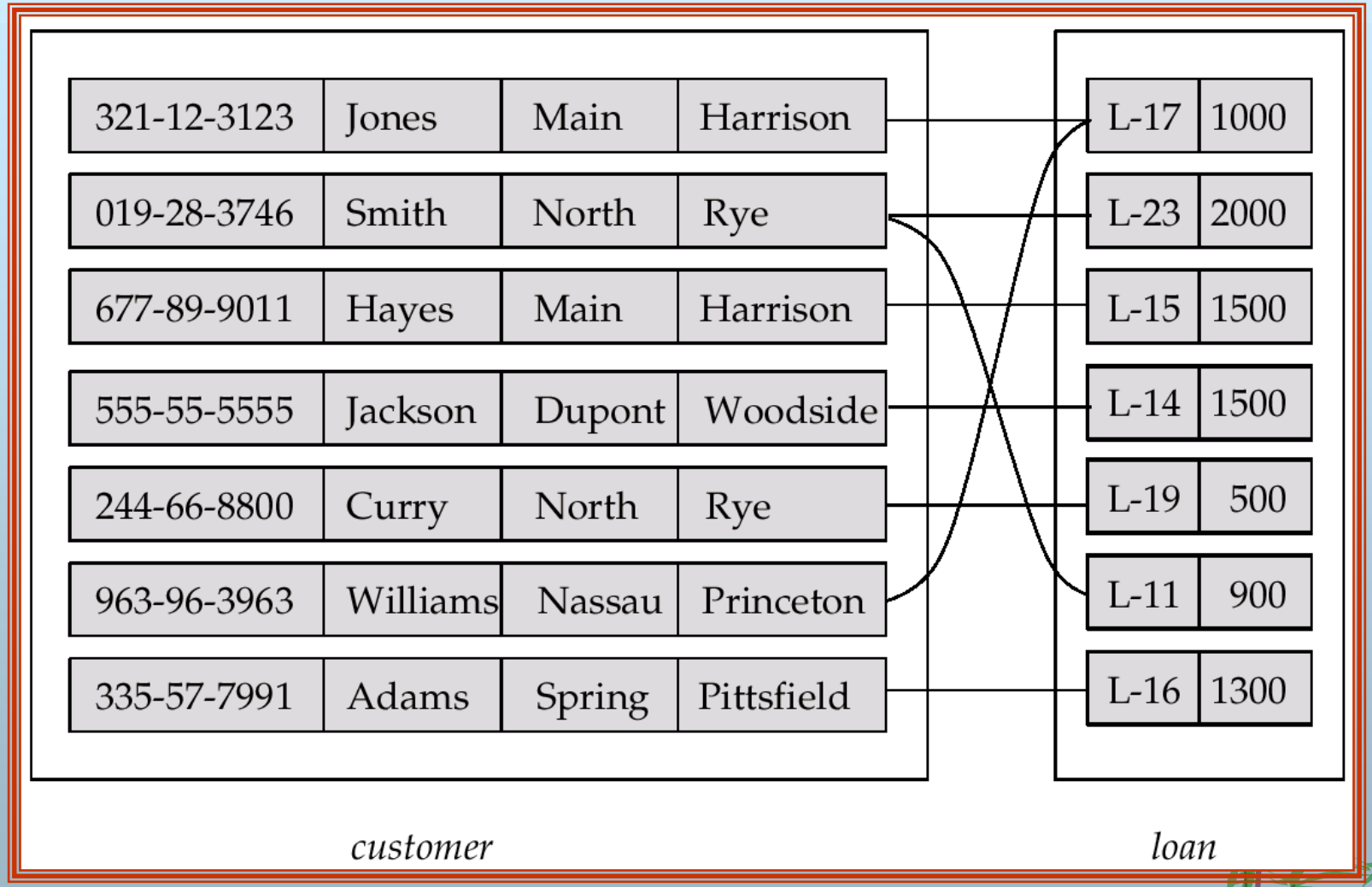
where $(e_1, e_2, \ldots, e_n)$ is a relationship

- Example:

$$(\text{Hayes, A-102}) \in \textit{depositor}$$

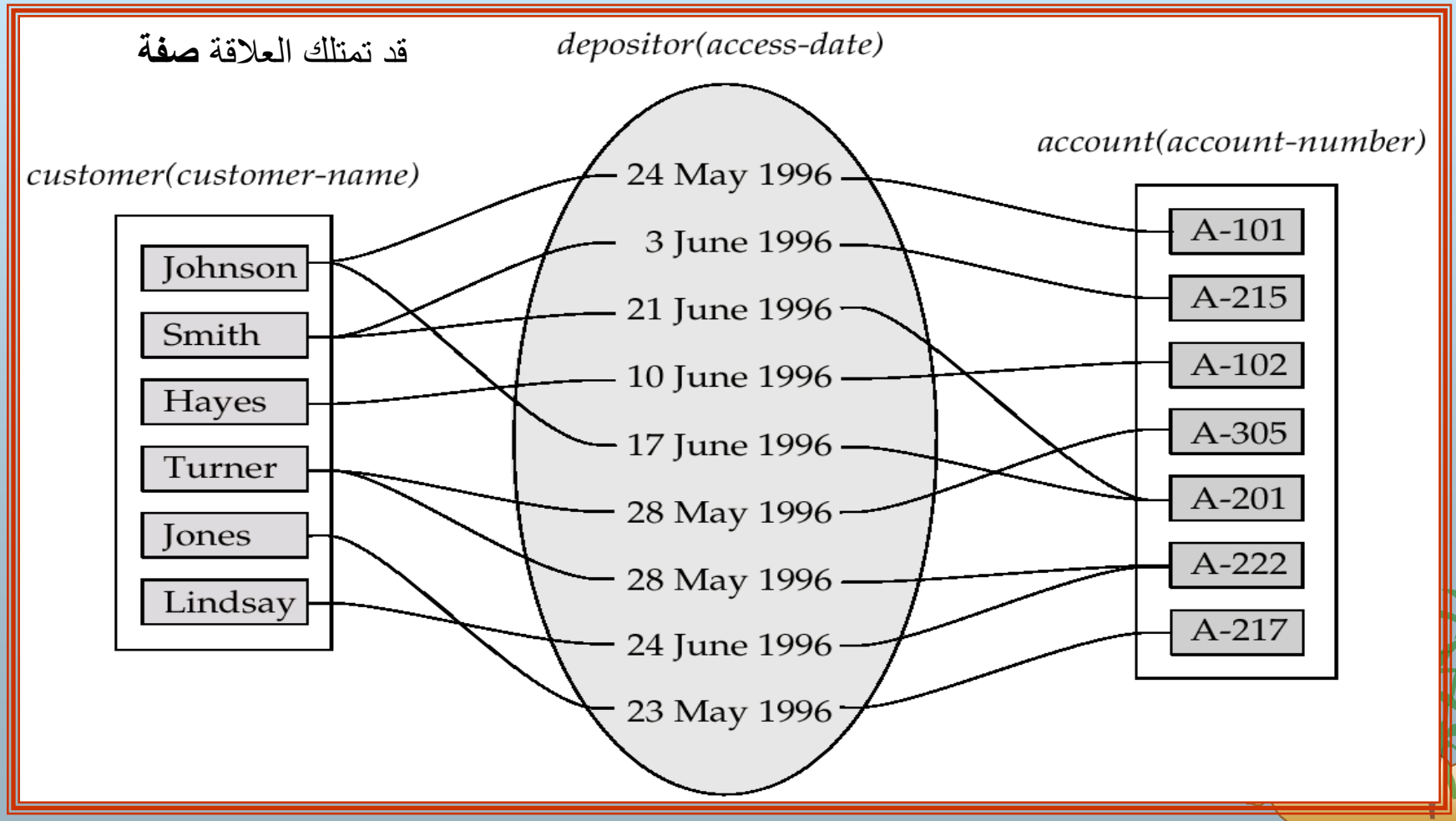


Relationship Set *borrower* (يقترض)



Relationship Sets (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***



Degree of a Relationship Set

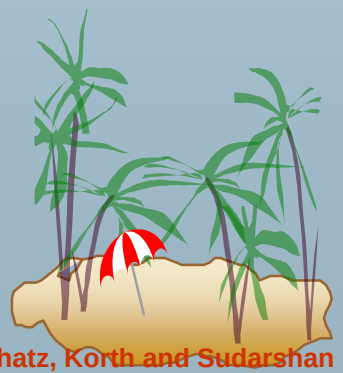
- Refers to **number of entity sets** that participate in a relationship set.
- Relationship sets that involve two entity sets are ***binary*** (or ***degree two***). Generally, most relationship sets in a database system are binary.
- Relationship sets may involve more than two entity sets.

  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*

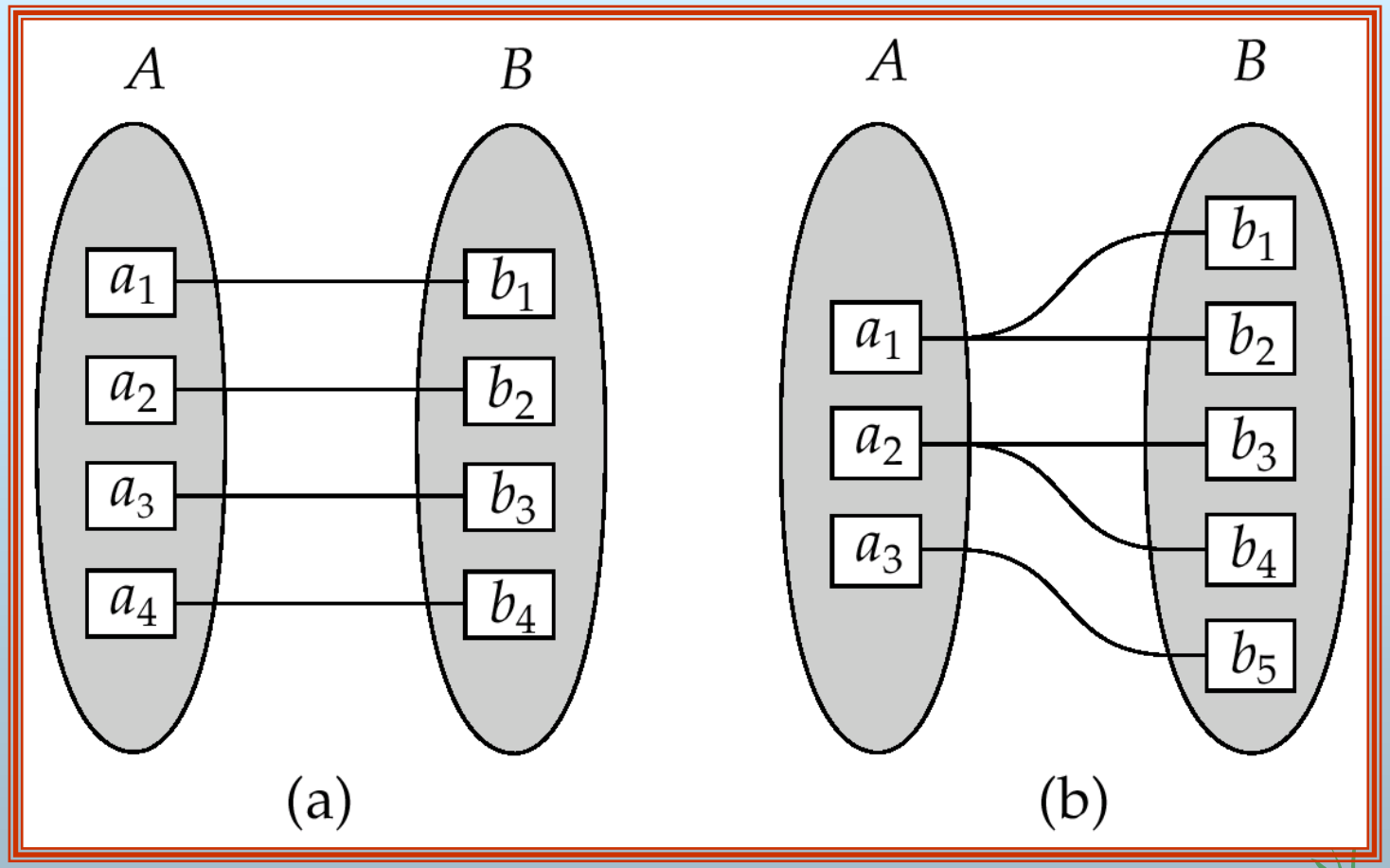
- Relationships between more than two entity sets are rare (نادر). Most relationships are binary. (More on this later.)

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



Mapping Cardinalities

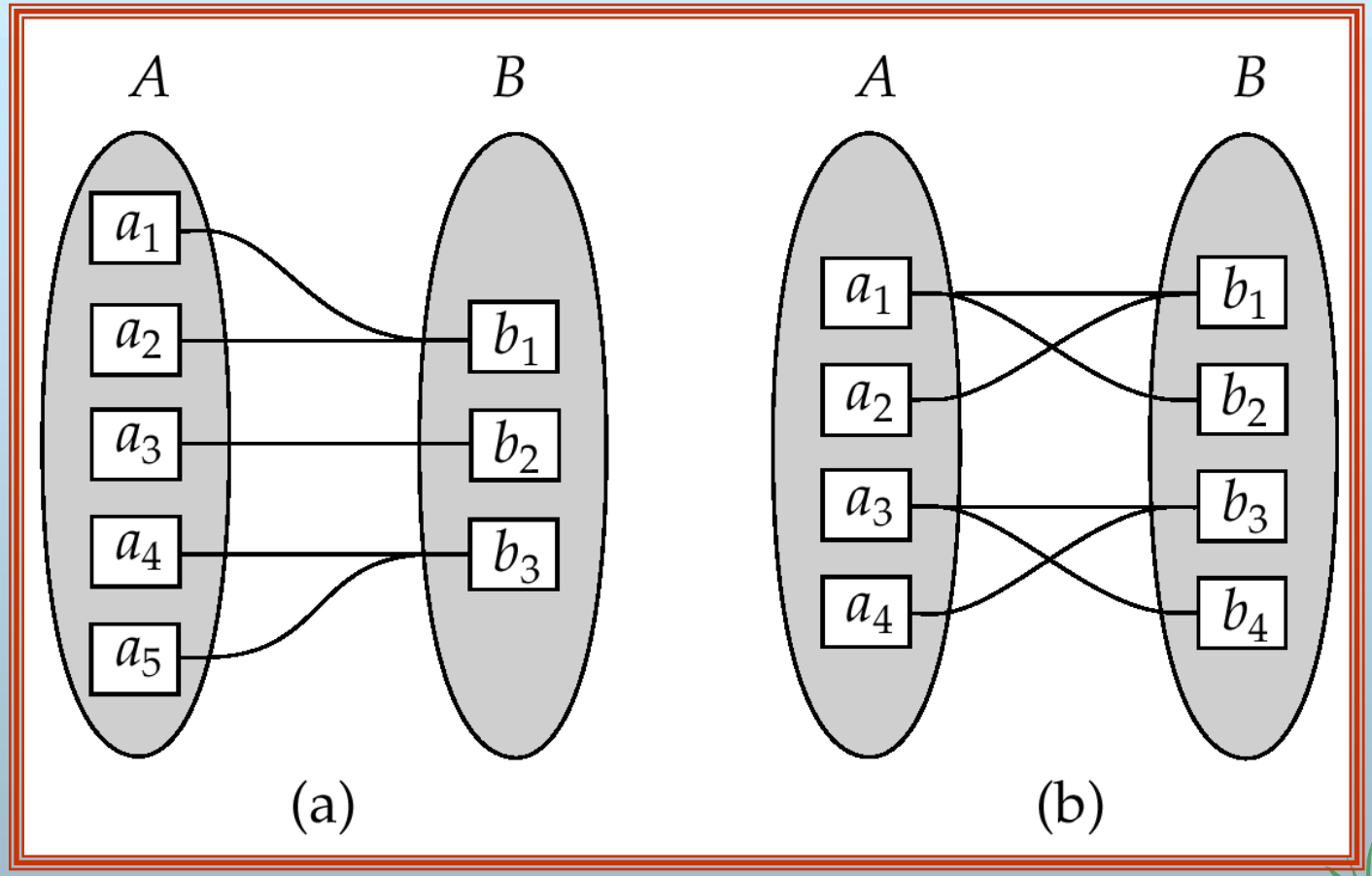


One to one

One to many

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

Mapping Cardinalities



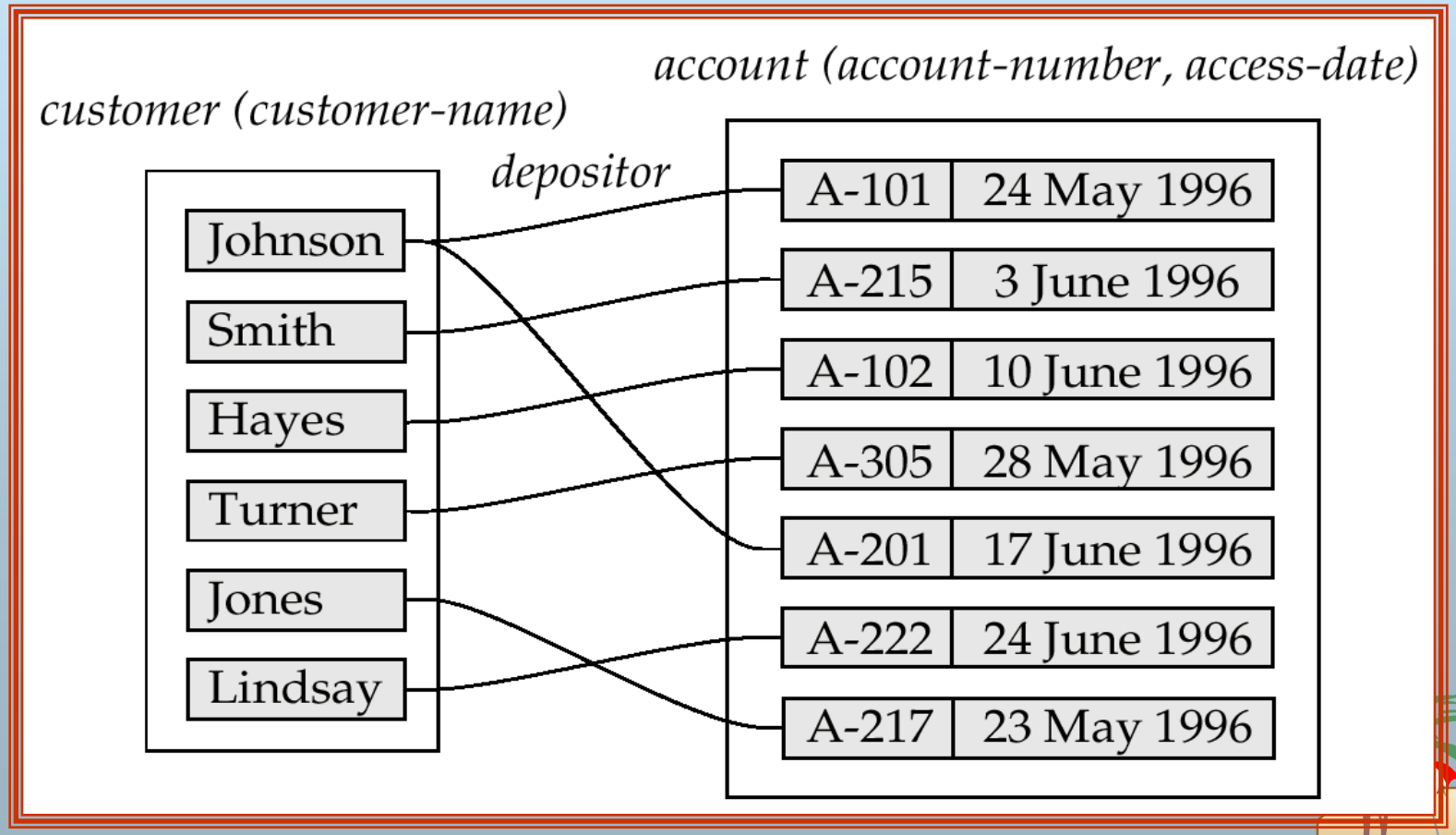
Many to one

Many to many

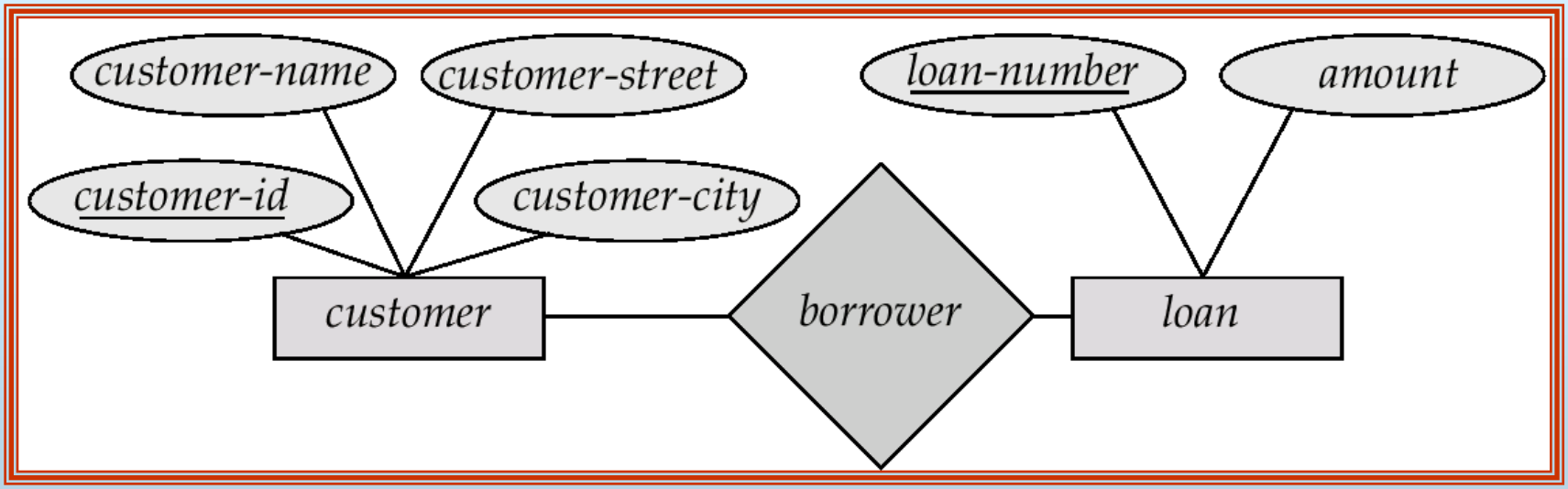
Note: Some elements in A and B may not be mapped to any elements in the other set

Mapping Cardinalities affect ER Design

- 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**

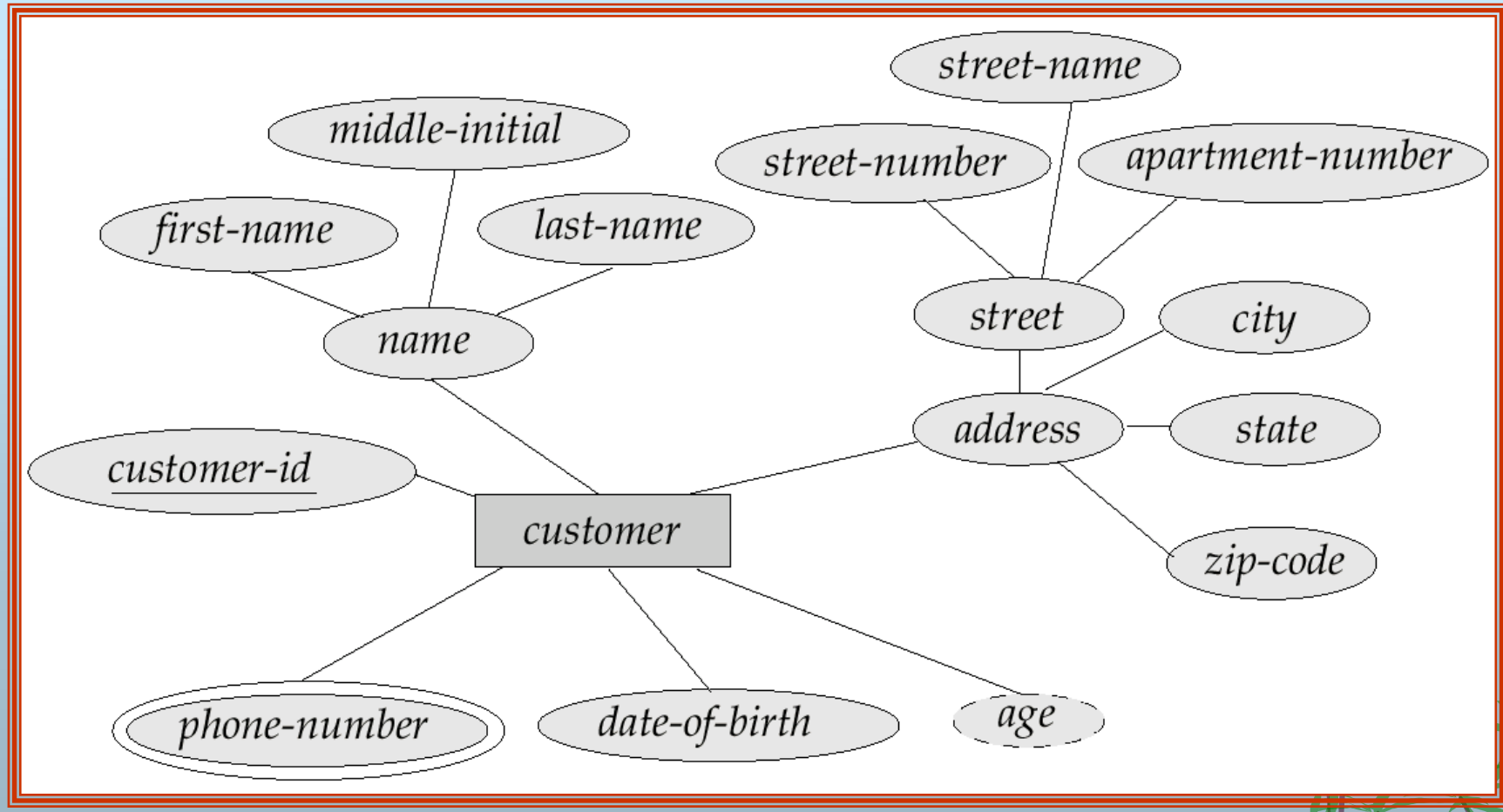


E-R Diagrams

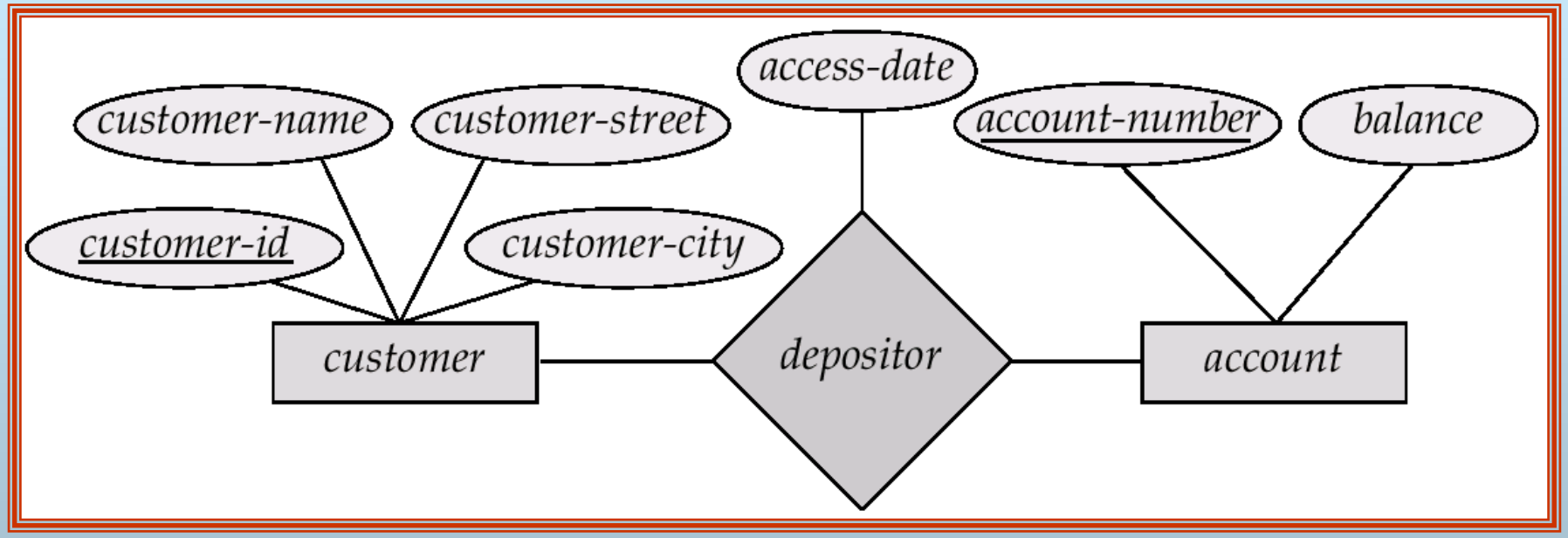


- **Rectangles** represent entity sets.
- **Diamonds** represent relationship sets.
- **Lines** link attributes to entity sets and entity sets to relationship sets.
- **Ellipses** represent attributes
  - **Double ellipses** represent multivalued attributes.
  - **Dashed ellipses** denote derived attributes.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

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

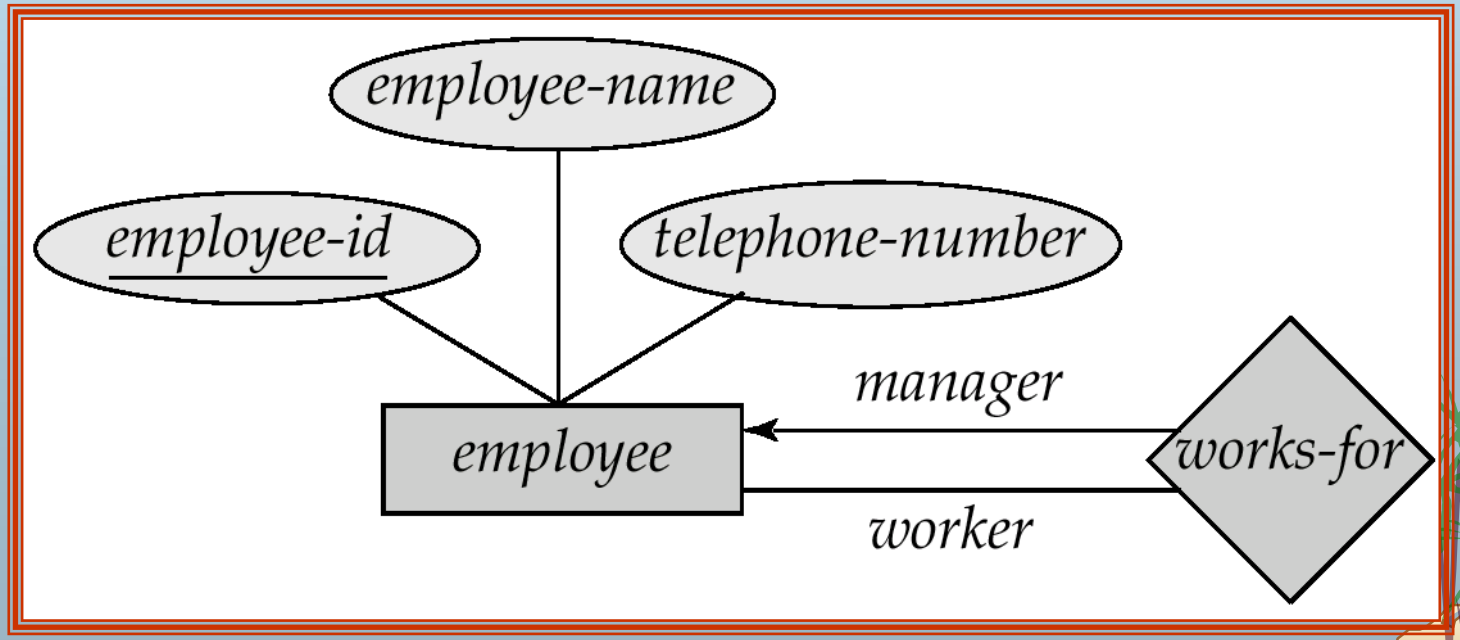


Relationship Sets with Attributes



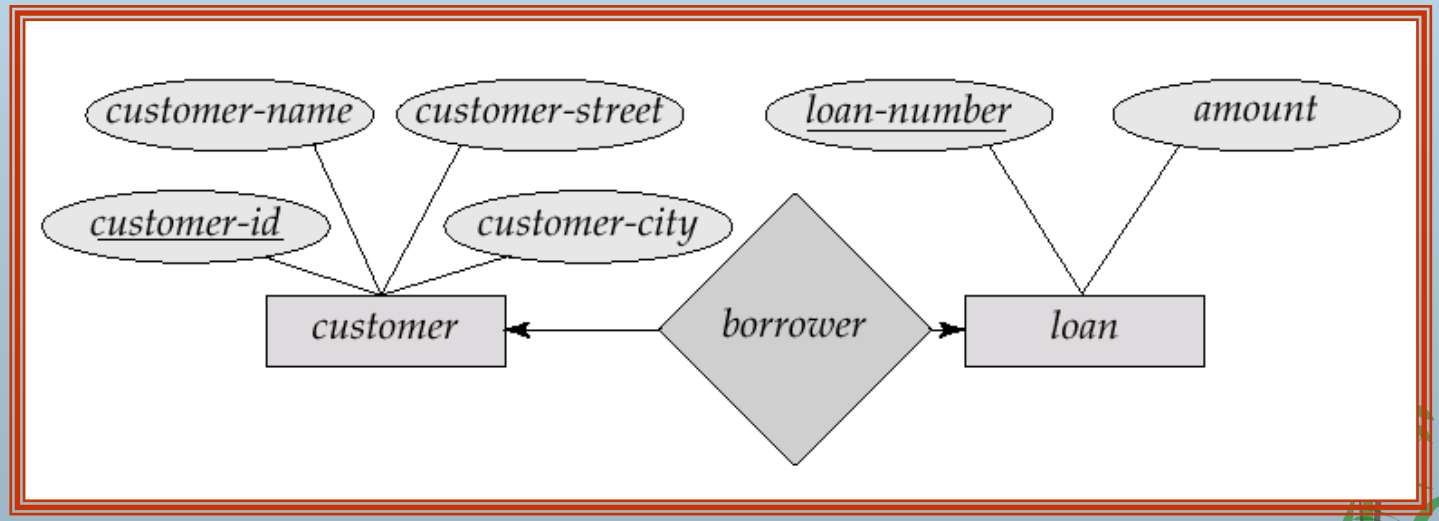
Roles

- Entity sets of a relationship need not be distinct (واضحة)
- 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



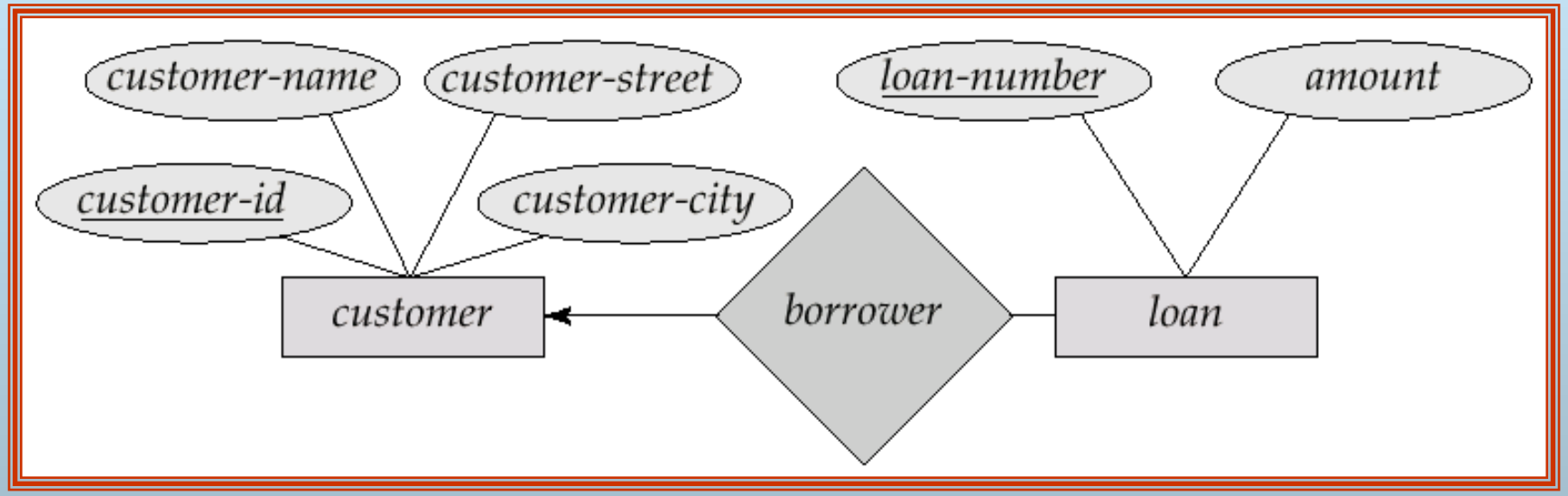
Cardinality Constraints

- 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*



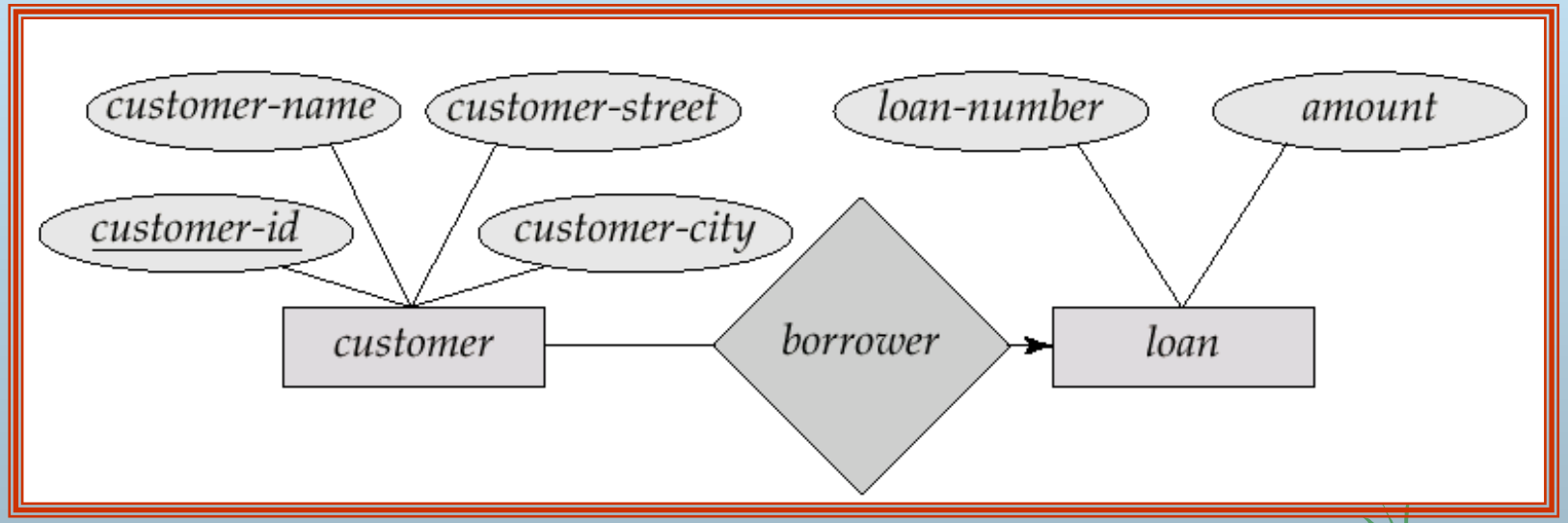
One-To-Many Relationship

- 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*

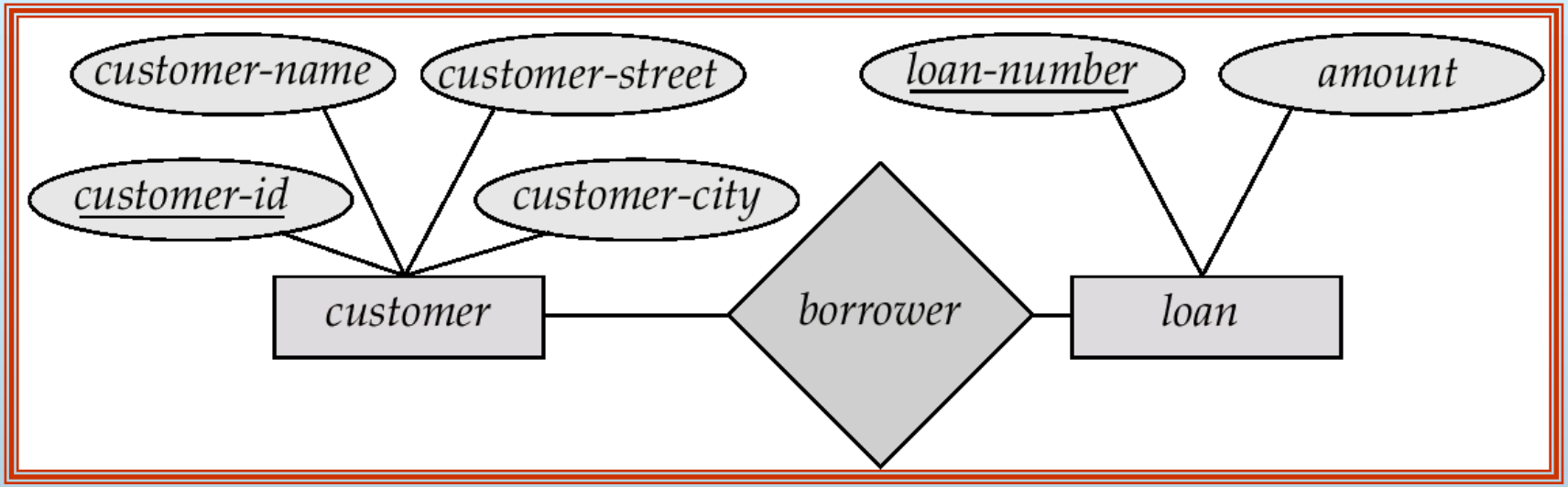


Many-To-One Relationships

- 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*



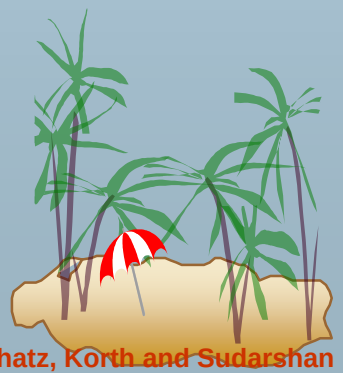
Many-To-Many Relationship



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

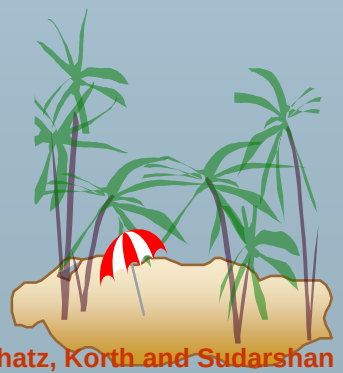
Keys

- 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***.

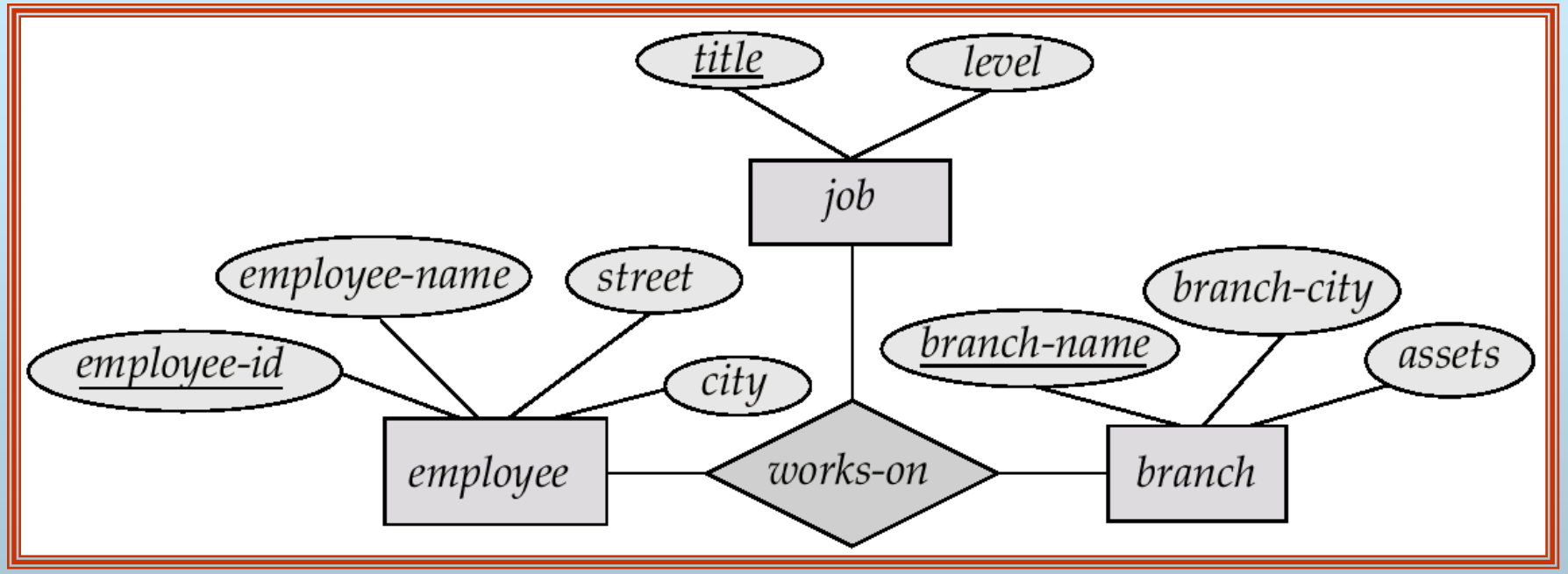


Keys for Relationship Sets

- The combination of primary keys of the participating entity sets forms a **super key** of a relationship set.
  - (*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



E-R Diagram with a Ternary Relationship

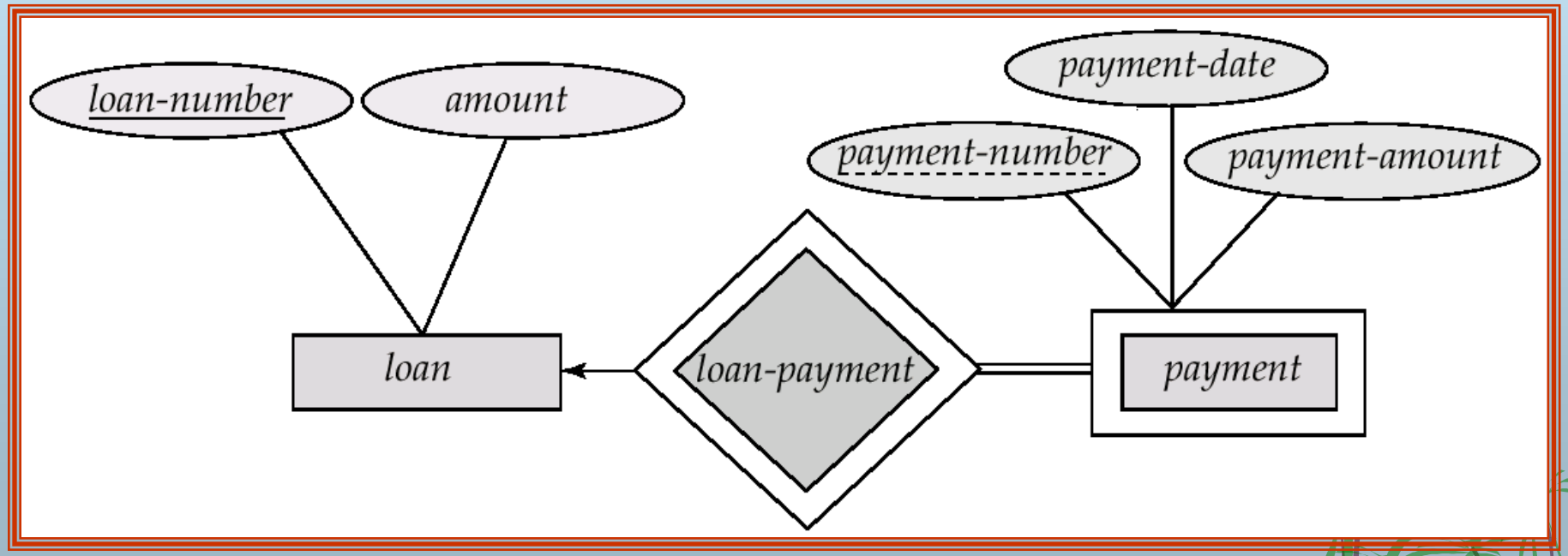


Weak Entity Sets

- An entity set that does not have a primary key is referred to as a ***weak entity set***.
- The existence of a weak entity set depends on the existence of a *identifying entity set*
  - it must relate to the identifying entity set via a total, one-to-many relationship set from the identifying to the weak entity set
  - Identifying relationship depicted using a double diamond
- The ***discriminator*** *(or partial key)* of a weak entity set is the set of attributes that distinguishes among all the entities of a weak entity set.
- The primary key of a weak entity set is formed by the primary key of the strong entity set on which the weak entity set is existence dependent, plus the weak entity set's discriminator.

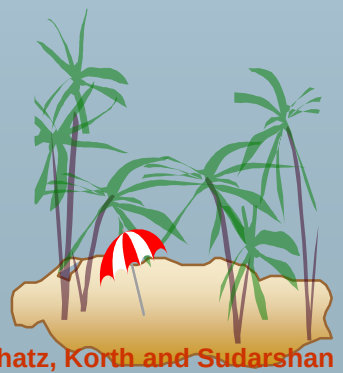
Weak Entity Sets (Cont.)

- We depict a weak entity set by double rectangles.
- We underline the discriminator of a weak entity set with a dashed line.
- *payment-number* – discriminator of the *payment* entity set
- Primary key for *payment* – (*loan-number, payment-number*)



Weak Entity Sets (Cont.)

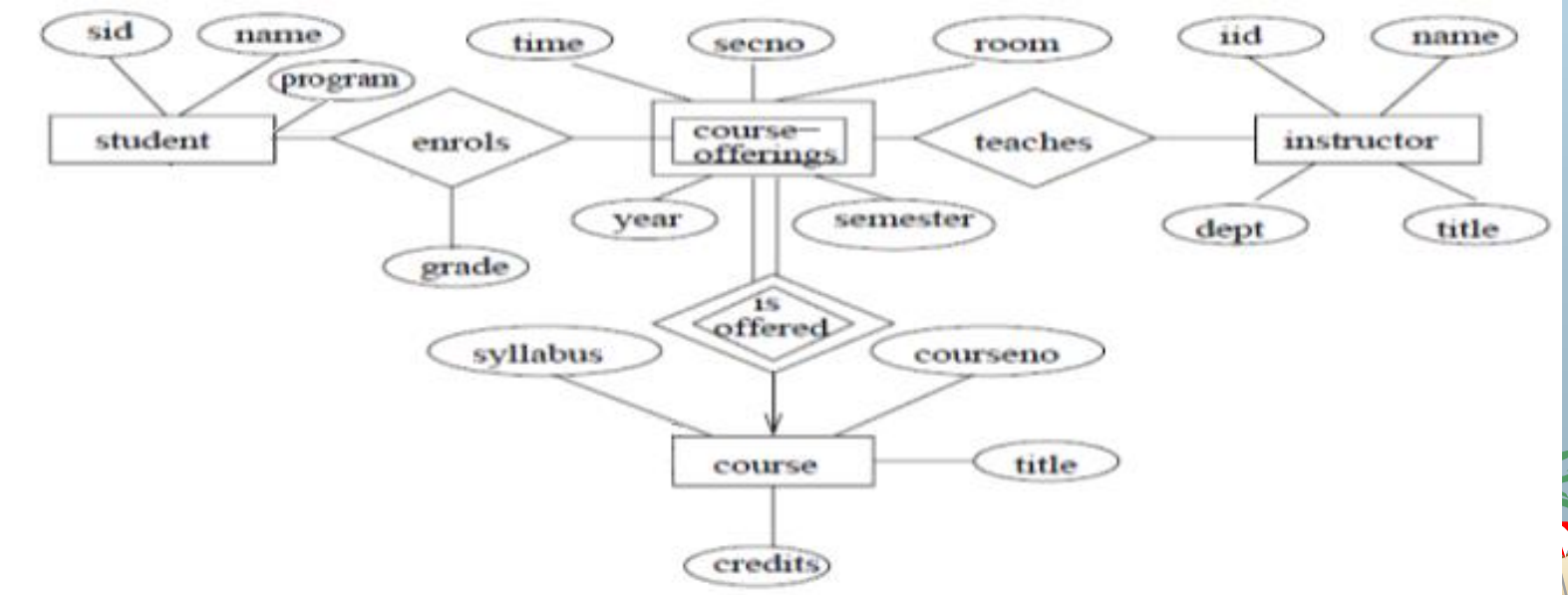
- Note: the primary key of the strong entity set is not explicitly stored with the weak entity set, since it is implicit in the identifying relationship.
- If *loan-number* were explicitly stored, *payment* could be made a strong entity, but then the relationship between *payment* and *loan* would be duplicated by an implicit relationship defined by the attribute *loan-number* common to *payment* and *loan*



More Weak Entity Set Examples

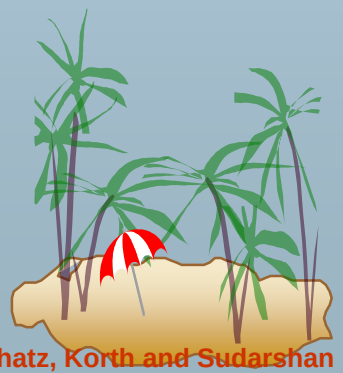
- In a university, a *course* is a strong entity and a *course-offering* can be modeled as a weak entity
- The discriminator of *course-offering* would be *semester* (including year) and *section-number* (if there is more than one section)
- If we model *course-offering* as a strong entity we would model *course-number* as an attribute.

  Then the relationship with *course* would be implicit in the *course-number* attribute

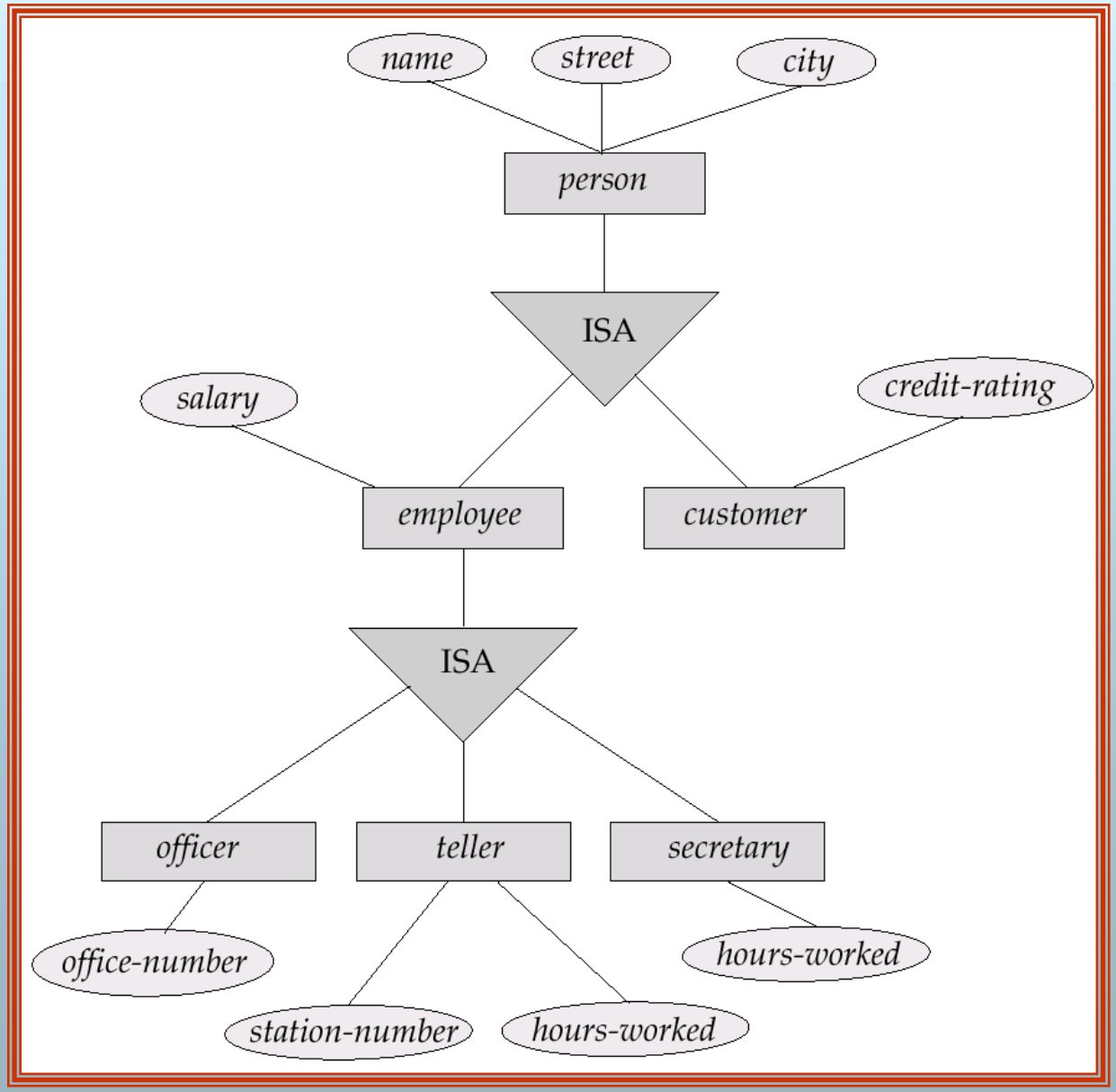


Specialization

- Top-down design process; we designate subgroupings within an entity set that are distinctive from other entities in the set.
- These subgroupings become lower-level entity sets that have attributes or participate in relationships that do not apply to the higher-level entity set.
- Depicted by a *triangle* component labeled ISA (E.g. *customer* "is a" *person*).
- **Attribute inheritance** – a lower-level entity set inherits all the attributes and relationship participation of the higher-level entity set to which it is linked.

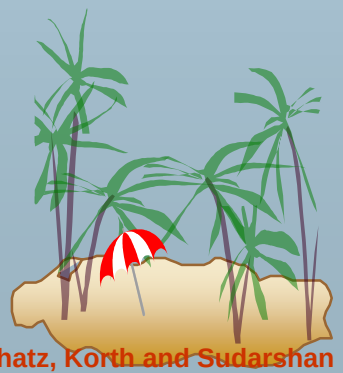


Specialization Example



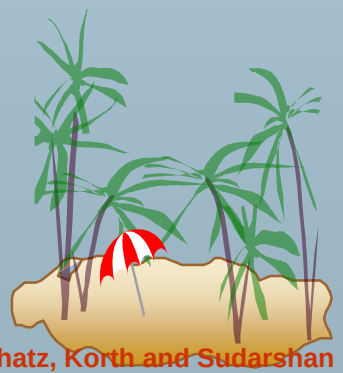
Generalization

- A bottom-up design process – combine a number of entity sets that share the same features into a higher-level entity set.
- Specialization and generalization are simple inversions of each other; they are represented in an E-R diagram in the same way.
- The terms specialization and generalization are used interchangeably.



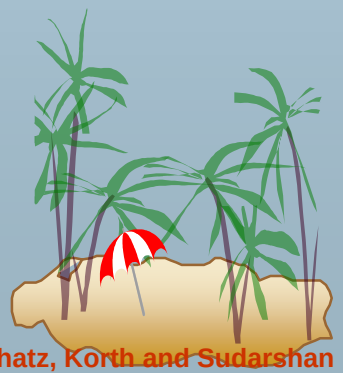
Specialization and Generalization (Contd.)

- Can have multiple specializations of an entity set based on different features.
- E.g. *permanent-employee* vs. *temporary-employee*, in addition to *officer* vs. *secretary* vs. *teller*
- Each particular employee would be
  - a member of one of *permanent-employee* or *temporary-employee*,
  - and also a member of one of *officer*, *secretary*, or *teller*
- The ISA relationship also referred to as **superclass - subclass** relationship

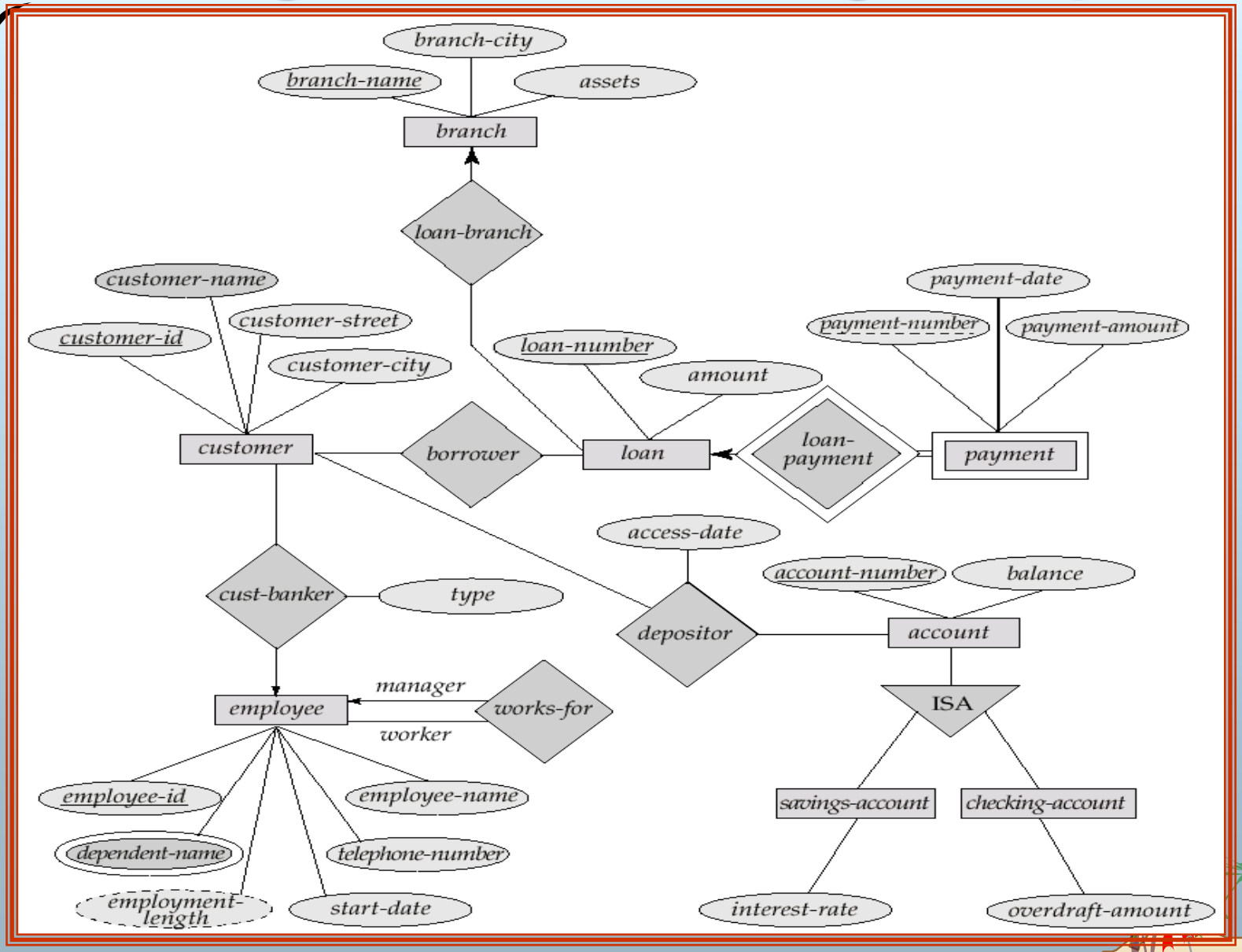


E-R Design Decisions

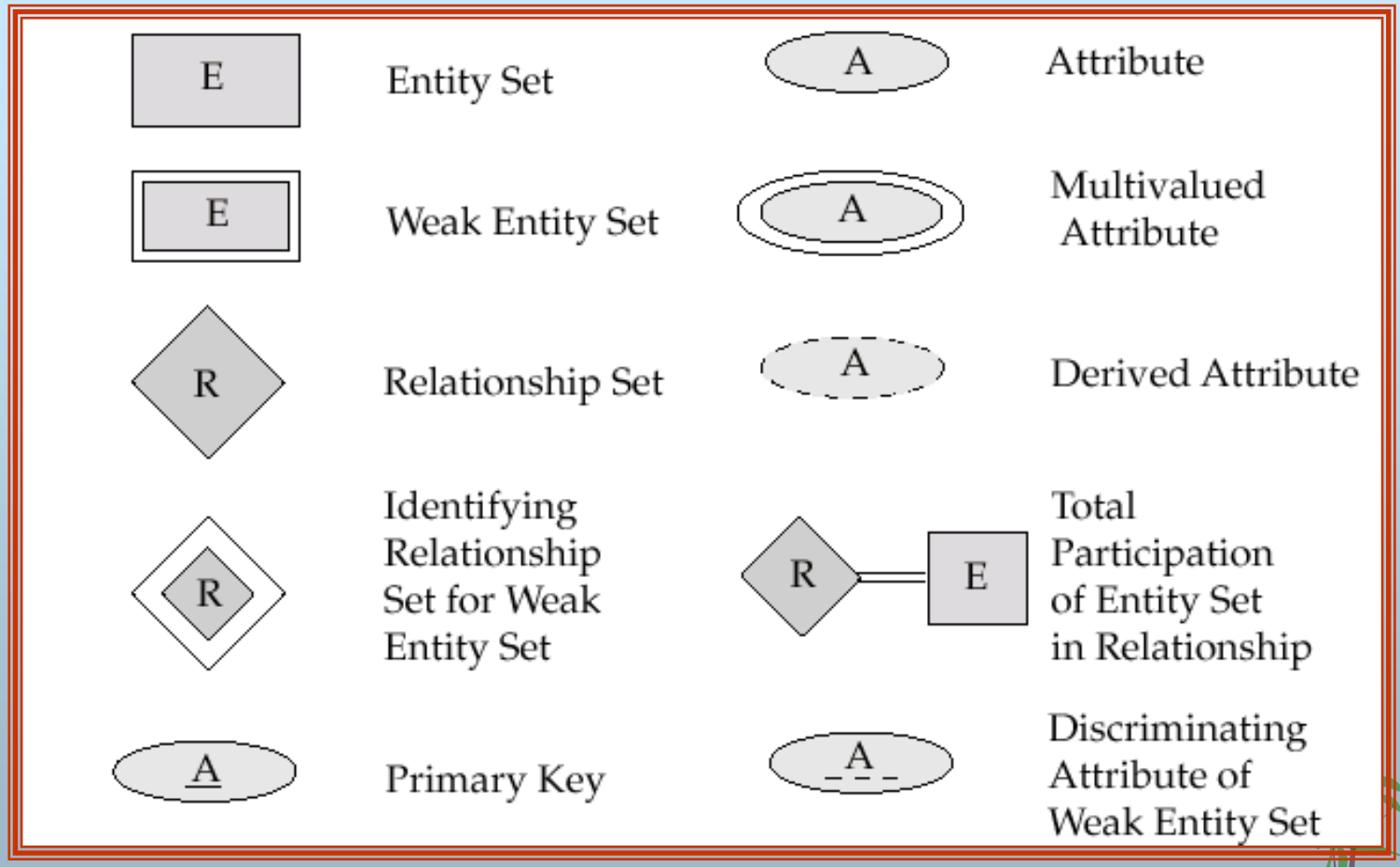
- The use of an attribute or entity set to represent an object.
- Whether a real-world concept is best expressed by an entity set or a relationship set.
- The use of a ternary relationship versus a pair of binary relationships.
- The use of a strong or weak entity set.
- The use of specialization/generalization – contributes to modularity in the design.
- The use of aggregation – can treat the aggregate entity set as a single unit without concern for the details of its internal structure.



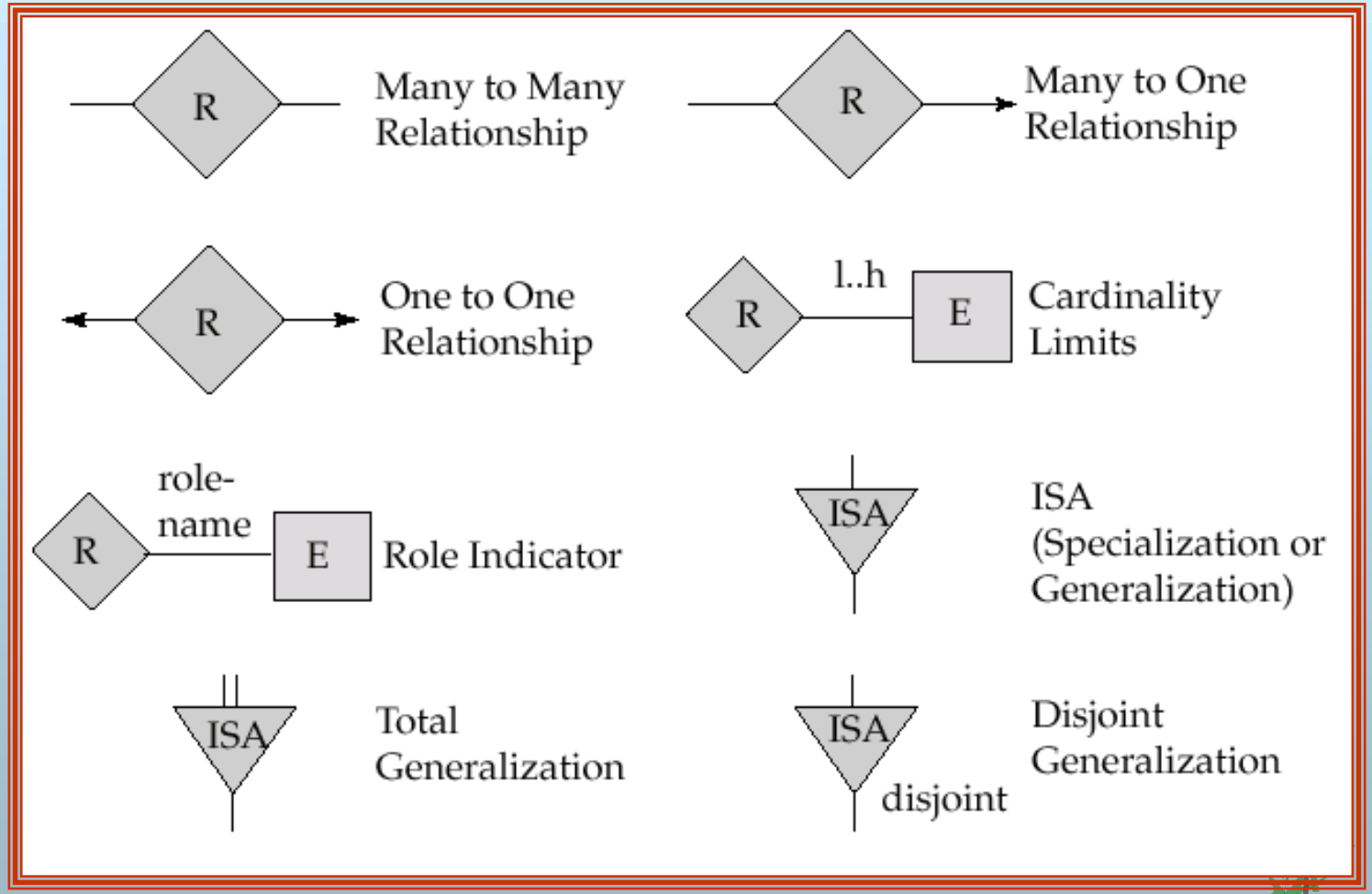
E-R Diagram for a Banking Enterprise



Summary of Symbols Used in E-R Notation

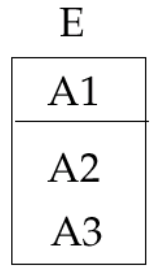


Summary of Symbols (Cont.)

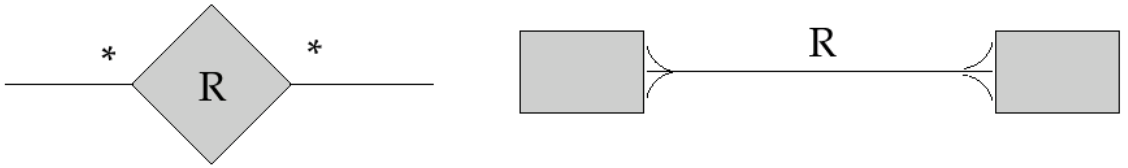


Alternative E-R Notations

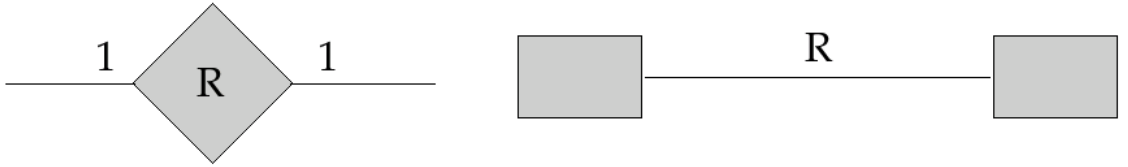
Entity set E with attributes A1, A2, A3 and primary key A1



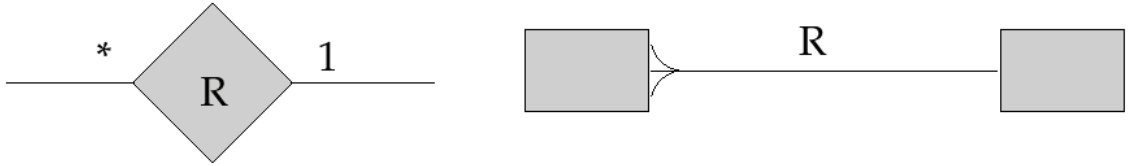
Many to Many Relationship



One to One Relationship

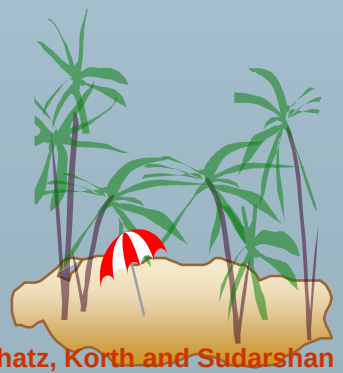


Many to One Relationship



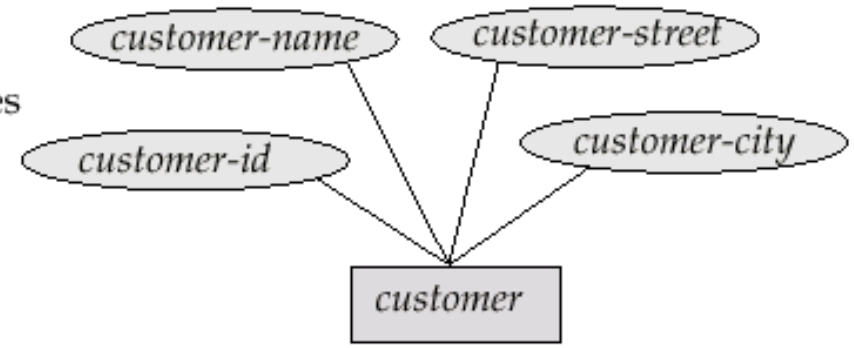
UML

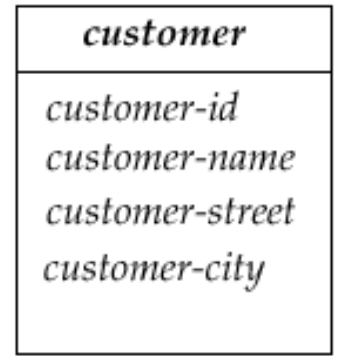
- UML: Unified Modeling Language
- UML has many components to graphically model different aspects of an entire software system
- UML Class Diagrams correspond to E-R Diagram, but several differences.



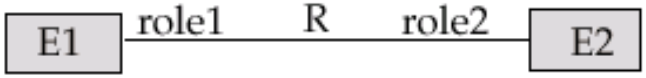
Summary of UML Class Diagram Notation

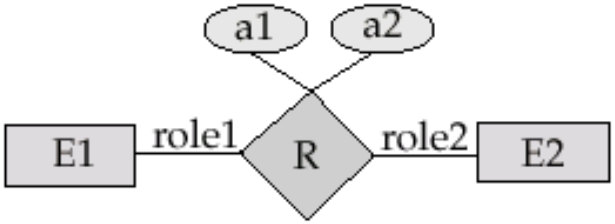
1. Entity sets and attributes

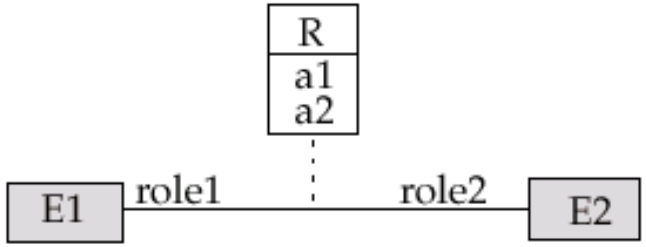




2. Relationships

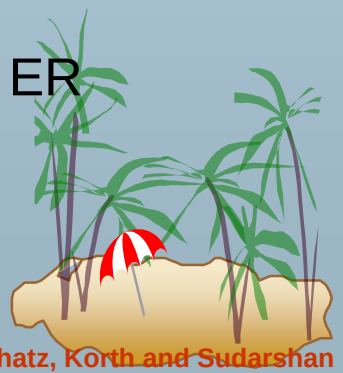




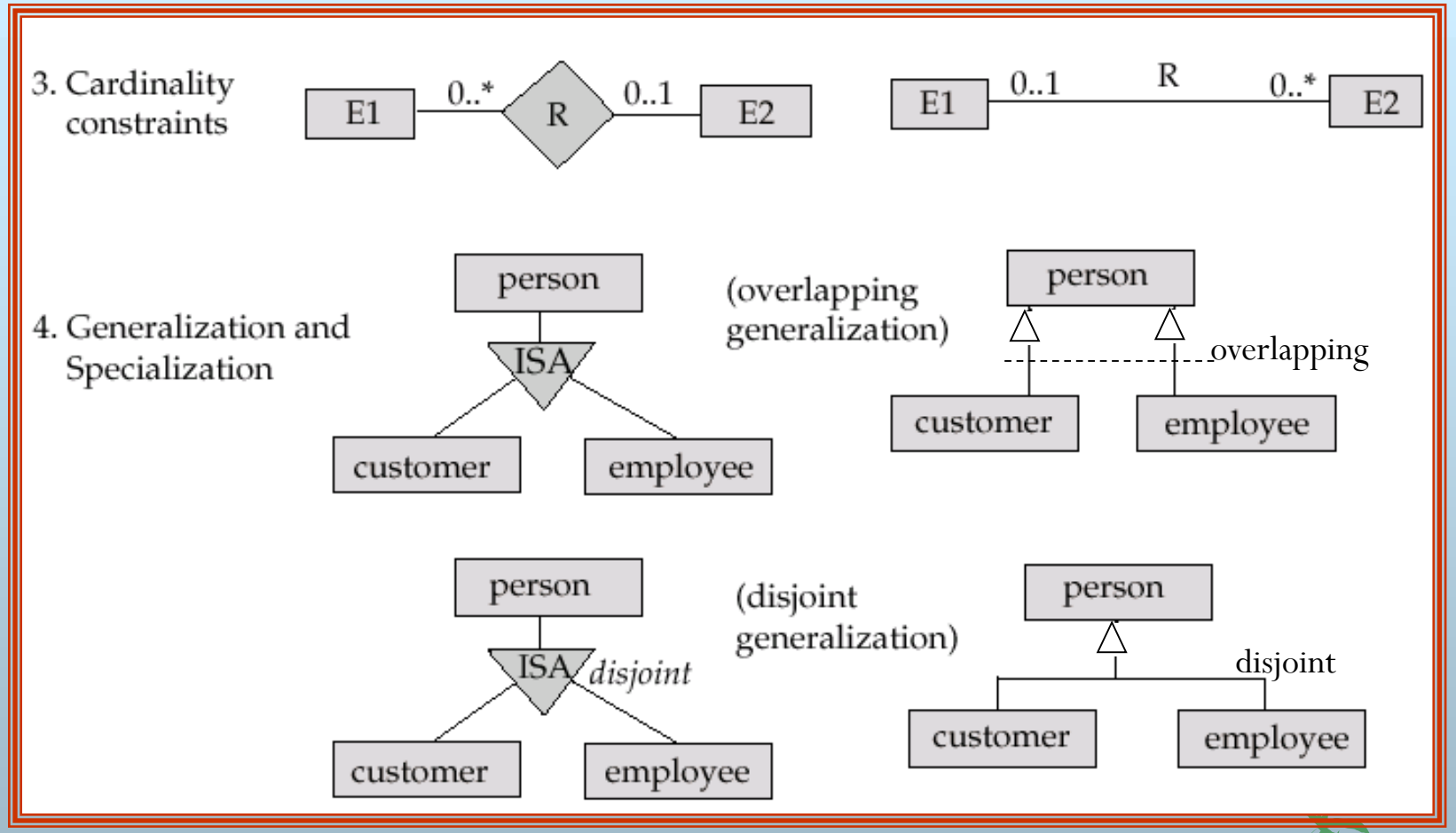


UML Class Diagrams (Contd.)

- Entity sets are shown as boxes, and attributes are shown within the box, rather than as separate ellipses in E-R diagrams.
- Binary relationship sets are represented in UML by just drawing a line connecting the entity sets. The relationship set name is written adjacent to the line.
- The role played by an entity set in a relationship set may also be specified by writing the role name on the line, adjacent to the entity set.
- The relationship set name may alternatively be written in a box, along with attributes of the relationship set, and the box is connected, using a dotted line, to the line depicting the relationship set.
- Non-binary relationships drawn using diamonds, just as in ER diagrams



UML Class Diagram Notation (Cont.)

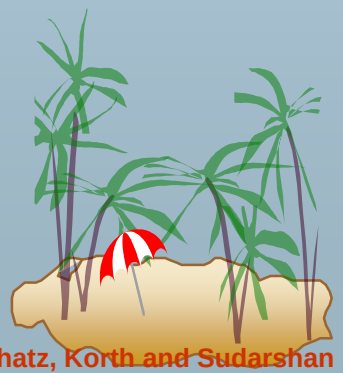


*Note reversal of position in cardinality constraint depiction

*Generalization can use merged or separate arrows independent of disjoint/overlapping

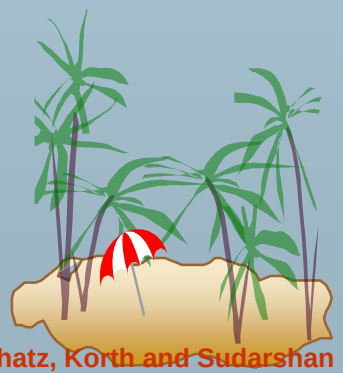
UML Class Diagrams (Contd.)

- Cardinality constraints are specified in the form *l..h*, where *l* denotes the minimum and *h* the maximum number of relationships an entity can participate in.
- Beware: the positioning of the constraints is exactly the reverse of the positioning of constraints in E-R diagrams.
- The constraint 0..* on the *E*2 side and 0..1 on the *E*1 side means that each *E*2 entity can participate in at most one relationship, whereas each *E*1 entity can participate in many relationships; in other words, the relationship is many to one from *E*2 to *E*1.
- Single values, such as 1 or * may be written on edges; The single value 1 on an edge is treated as equivalent to 1..1, while * is equivalent to 0..*.



Reduction of an E-R Schema to Tables

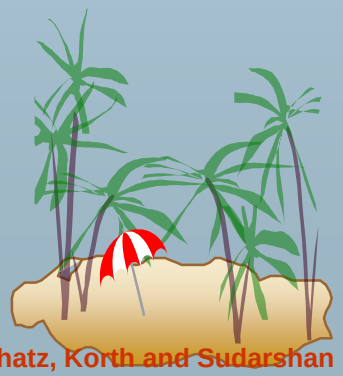
- Primary keys allow entity sets and relationship sets to be expressed uniformly as *tables* which represent the contents of the database.
- A database which conforms to an E-R diagram can be represented by a collection of tables.
- For each entity set and relationship set there is a unique table which is assigned the name of the corresponding entity set or relationship set.
- Each table has a number of columns (generally corresponding to attributes), which have unique names.
- Converting an E-R diagram to a table format is the basis for deriving a relational database design from an E-R diagram.



Representing Entity Sets as Tables

- A strong entity set reduces to a table with the same attributes.

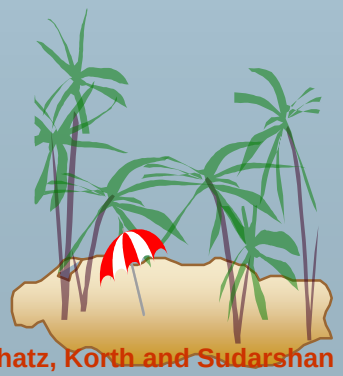
customer-id	*customer-name*	*customer-street*	*customer-city*
019-28-3746	Smith	North	Rye
182-73-6091	Turner	Putnam	Stamford
192-83-7465	Johnson	Alma	Palo Alto
244-66-8800	Curry	North	Rye
321-12-3123	Jones	Main	Harrison
335-57-7991	Adams	Spring	Pittsfield
336-66-9999	Lindsay	Park	Pittsfield
677-89-9011	Hayes	Main	Harrison
963-96-3963	Williams	Nassau	Princeton



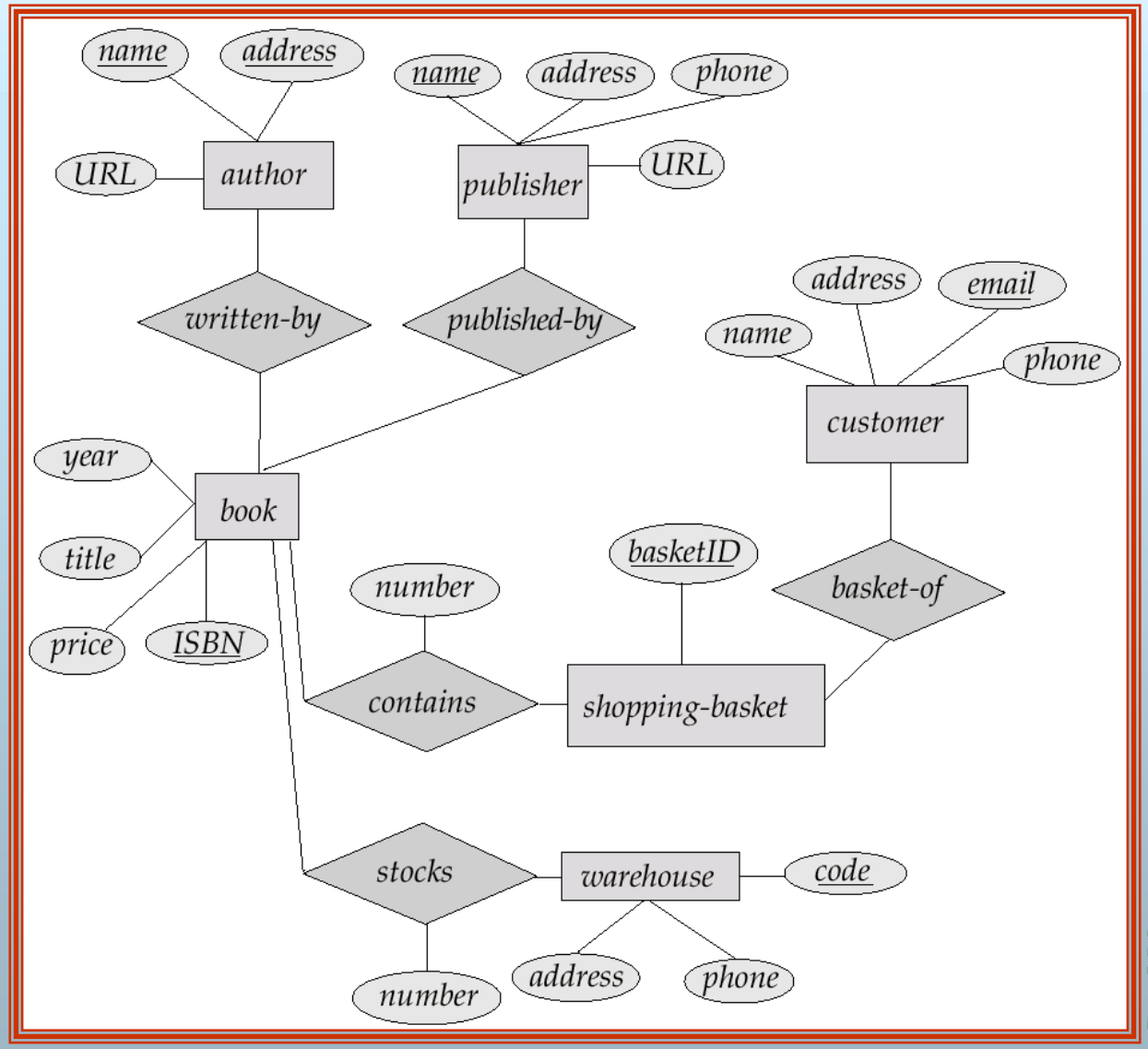
Representing Relationship Sets as Tables

- A many-to-many relationship set is represented as a table with columns for the primary keys of the two participating entity sets, and any descriptive attributes of the relationship set.
- E.g.: table for relationship set *borrower*

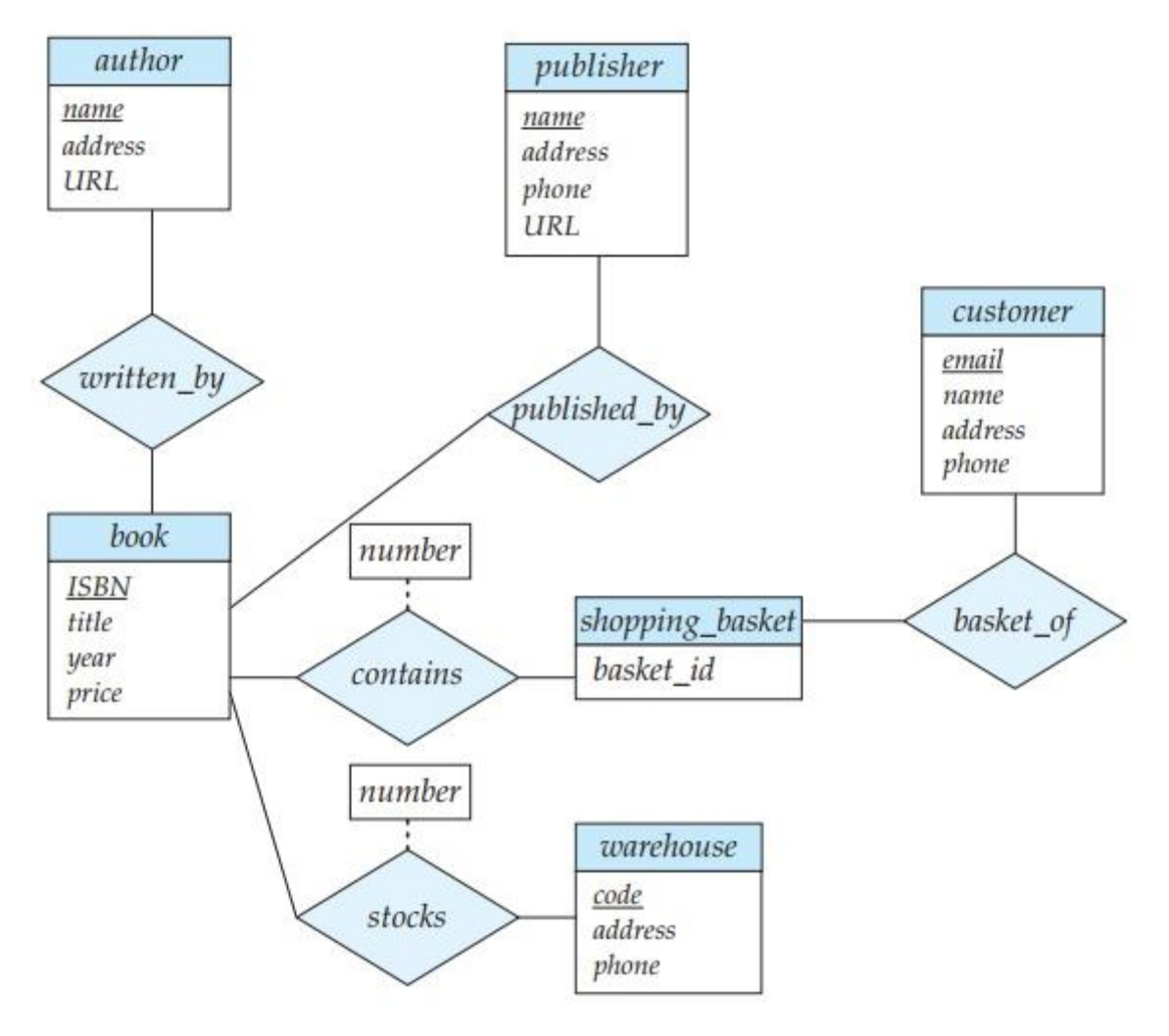
customer-id	*loan-number*
019-28-3746	L-11
019-28-3746	L-23
244-66-8800	L-93
321-12-3123	L-17
335-57-7991	L-16
555-55-5555	L-14
677-89-9011	L-15
963-96-3963	L-17



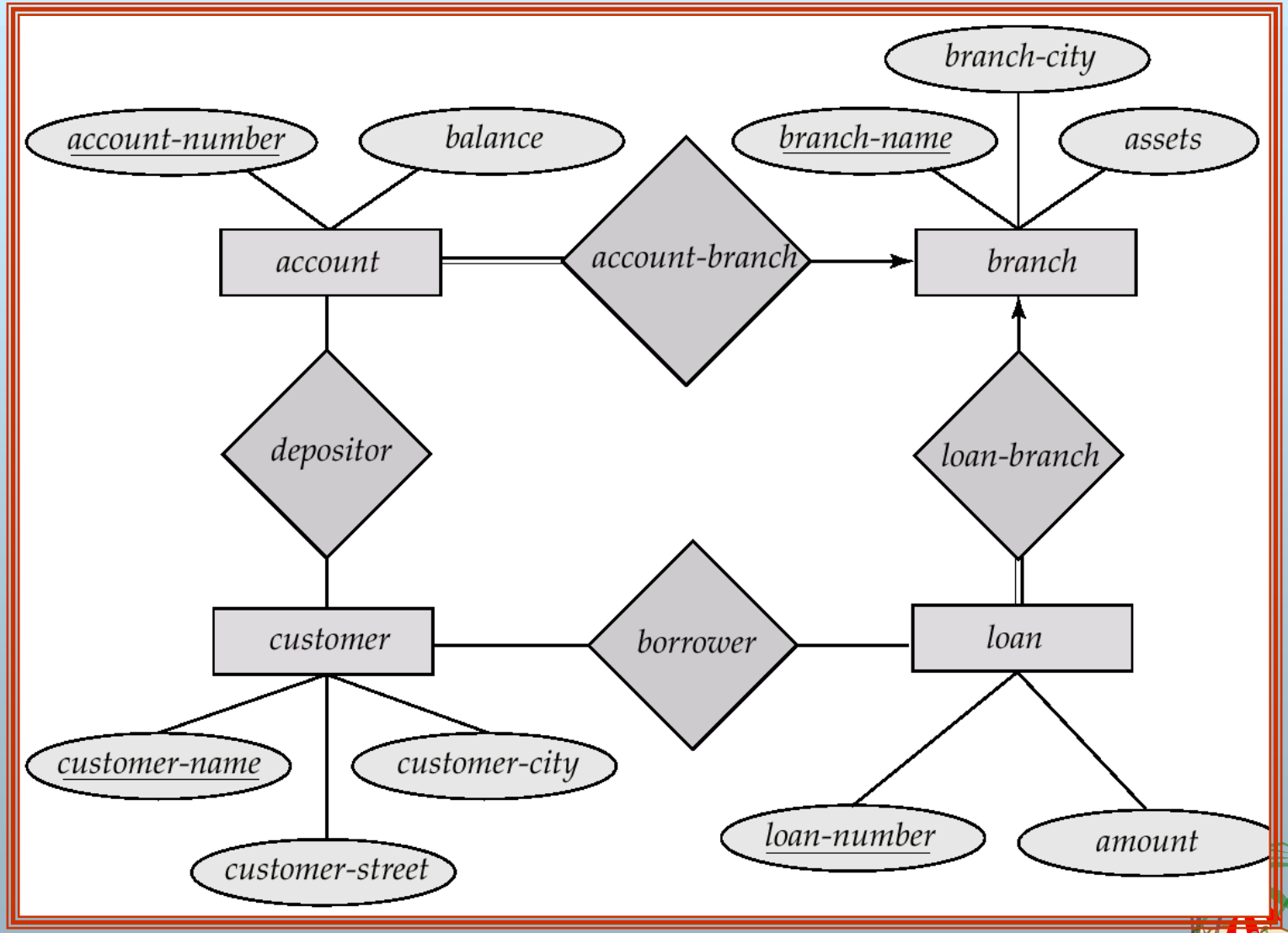
E-R Diagram for Book-Store Database



UML Diagram for Book-Store Database



E-R Diagram for the Banking Enterprise



Schema Diagram(UML) for the Banking Enterprise