

Entity-Relationship Model

Chapter 2



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, department, event, plants, specific book, specific course
- 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, books, students.



Entity Sets *customer* and *loan*

customer-id customer- customer- customer-
 name street 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- amount
number

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.

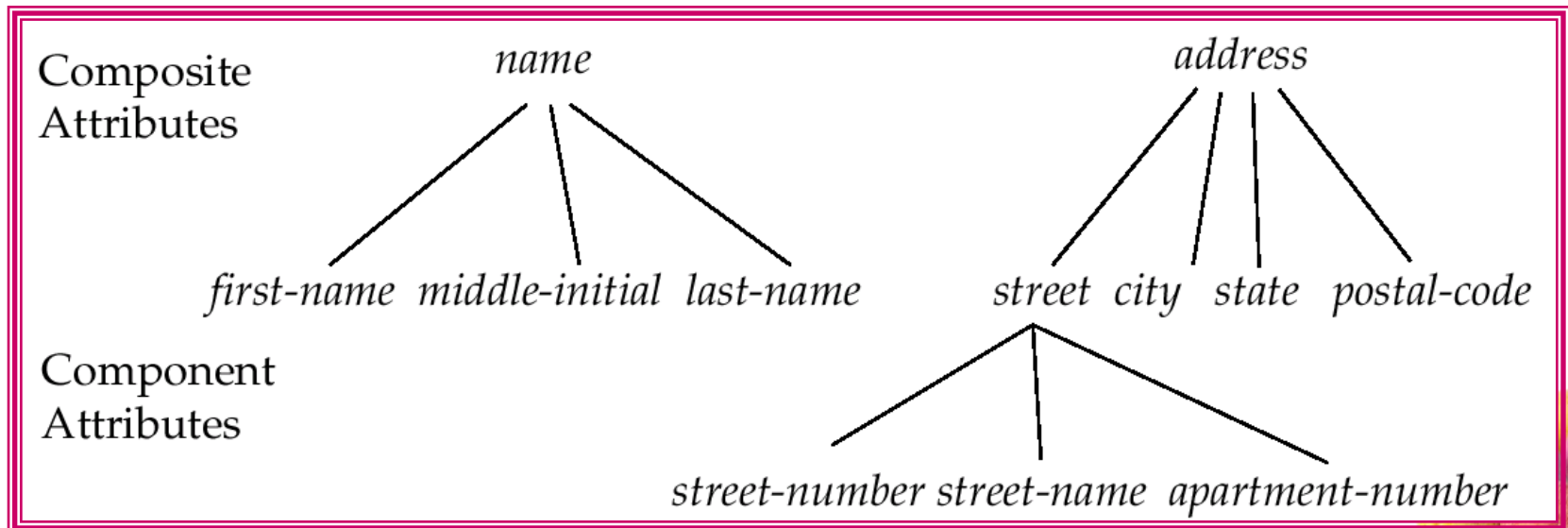
Example:

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

- *Domain* – the set of permitted values for each attribute . Its is also called **value set**.
- Database includes a collection of entity sets.
- Attribute types:



- 
- *Simple and composite attributes* :
 - The attributes is simple if it is not divided into subpart.
 - Composite attributes → can be divided into subpart. E.g name → firstname , middle name , last name



- *Single-valued and multi-valued attributes*
 - Attribute have a single value at a time.
 - Multi valued attribute can have multiple value at a time for same instance.
 - E.g. *phone-numbers*
- *Derived attributes*
 - Can be computed from other attributes
 - E.g. *age*, given date of birth
 - An attribute can take a null value. That is the value does not exist for an entity instance.
 - Sometime null value consider as unknown value. Unknown value may be either missing or not known.



Relationship Sets

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

Example: have
Sharma 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

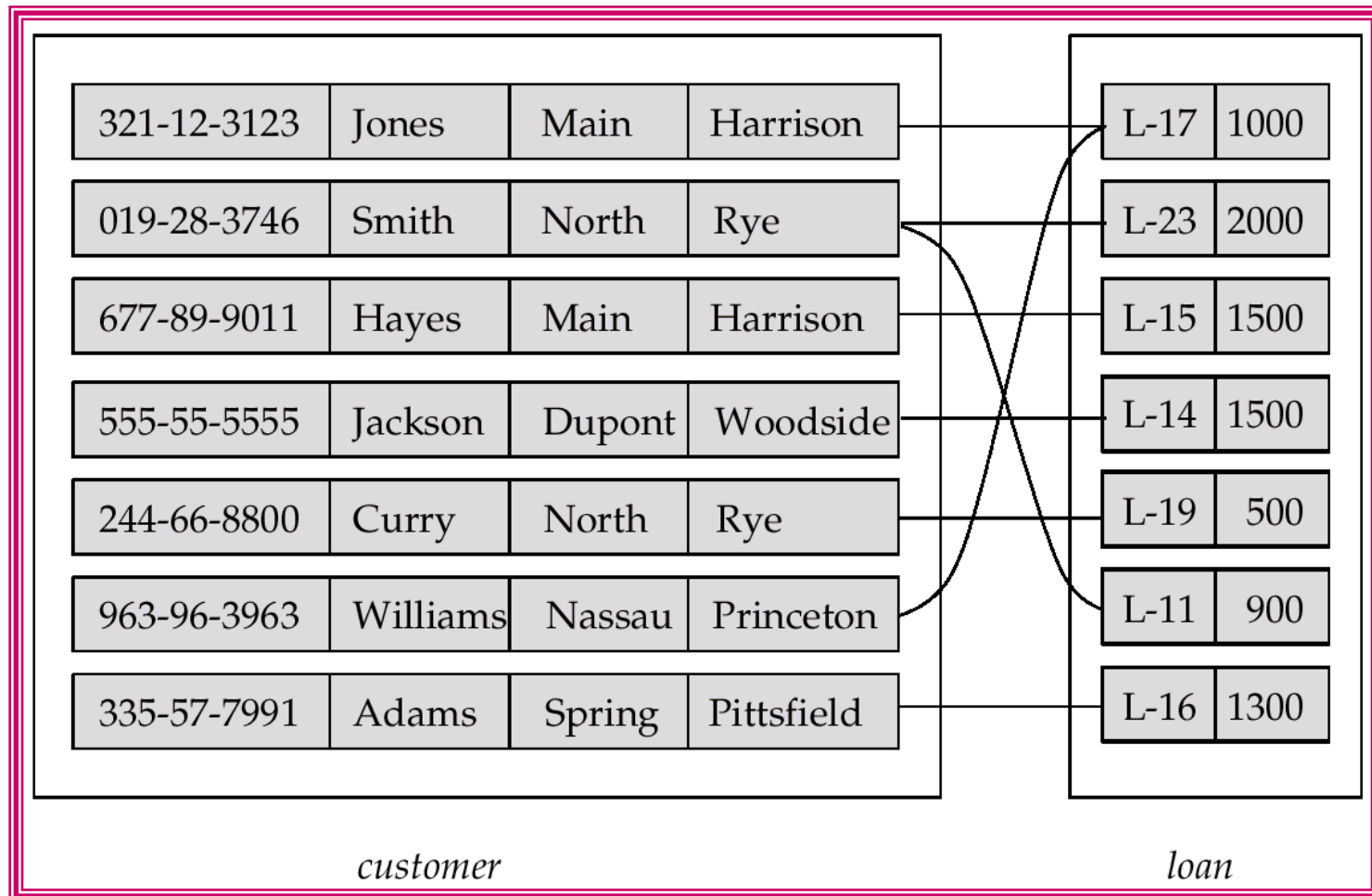
$\{(e_1, e_2, \dots, e_n) \mid e_1 \in E_1, e_2 \in E_2, \dots, e_n \in E_n\}$
where (e_1, e_2, \dots, e_n) is a relationship

- Example:

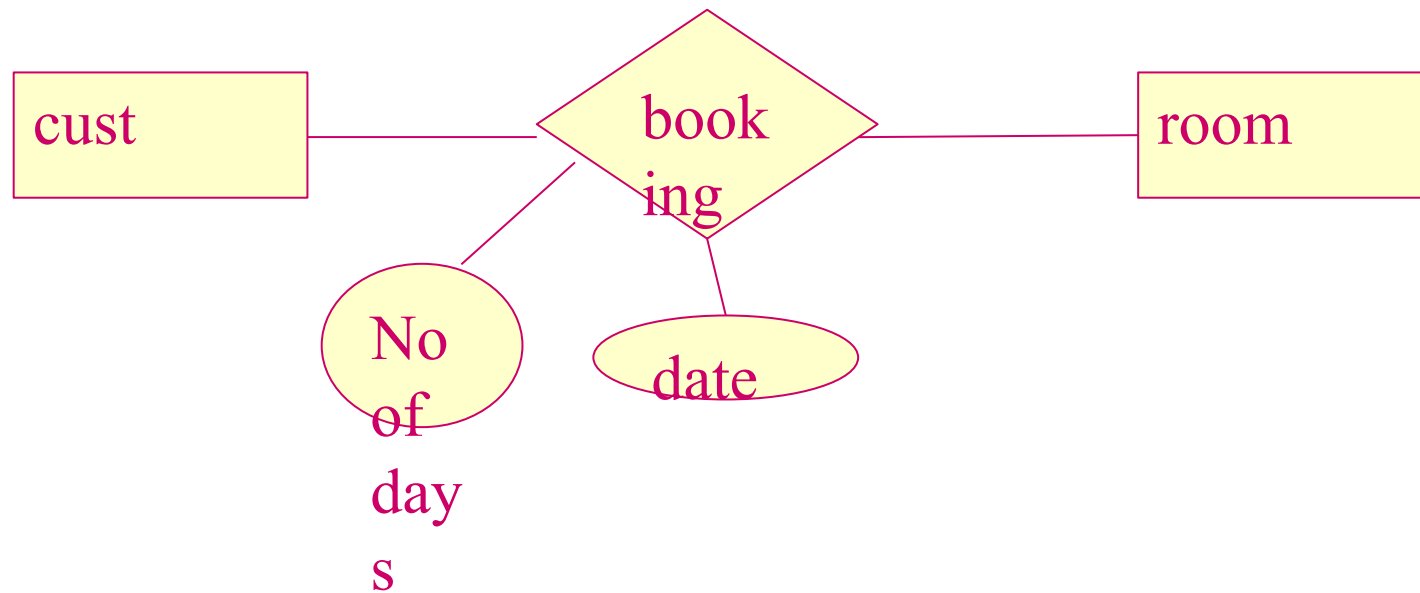
$(\text{Hayes}, \text{A-102}) \in \text{depositor}$



Relationship Set *borrower*



- Relationship may also have attributes → called **descriptive attributes**.
- E.g depositer has access-date
- Relationship to specify the most recent date on DEFAULT which customer accessed his account.



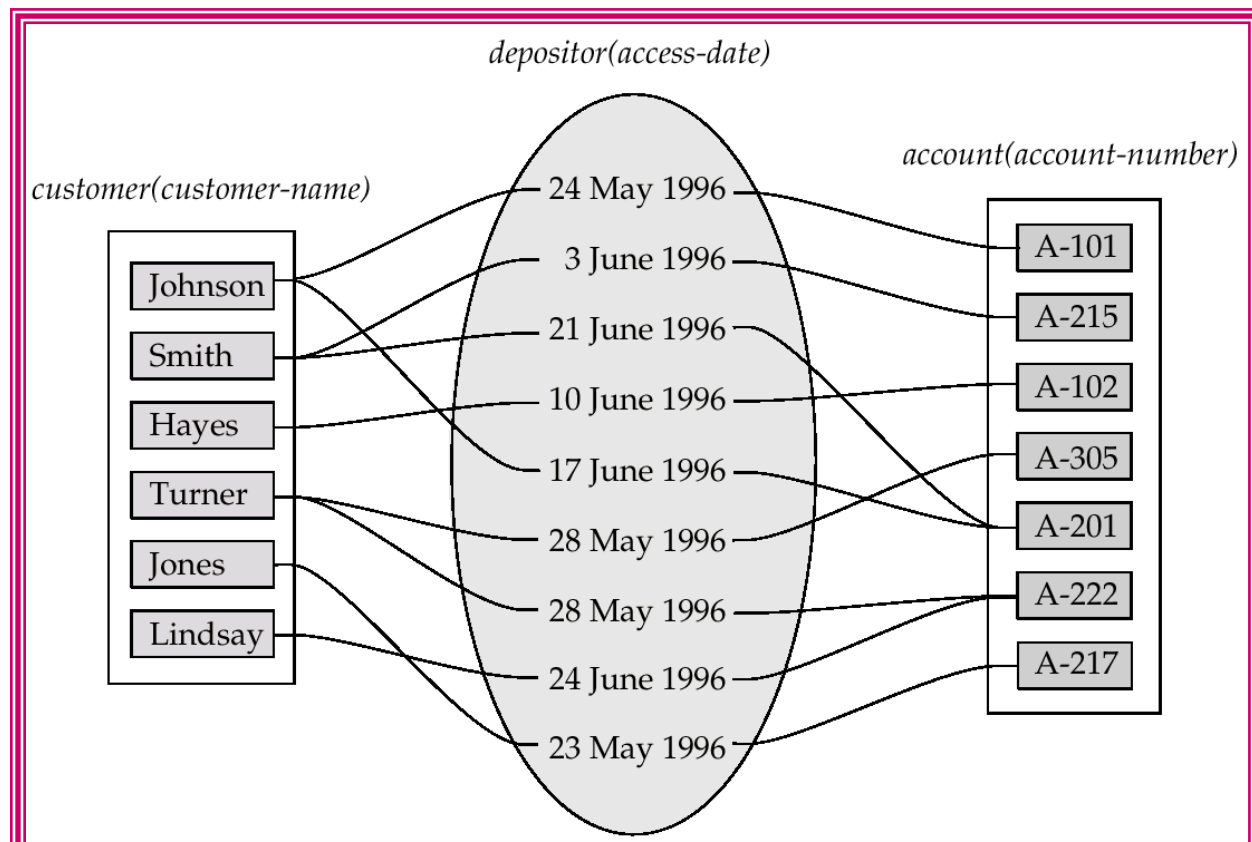


- Custid , custname,
- 1 xxx
- Custid,roomno, date, noday
- 1 101 8-12-2019 3
- 1 201 8-12-2020 2

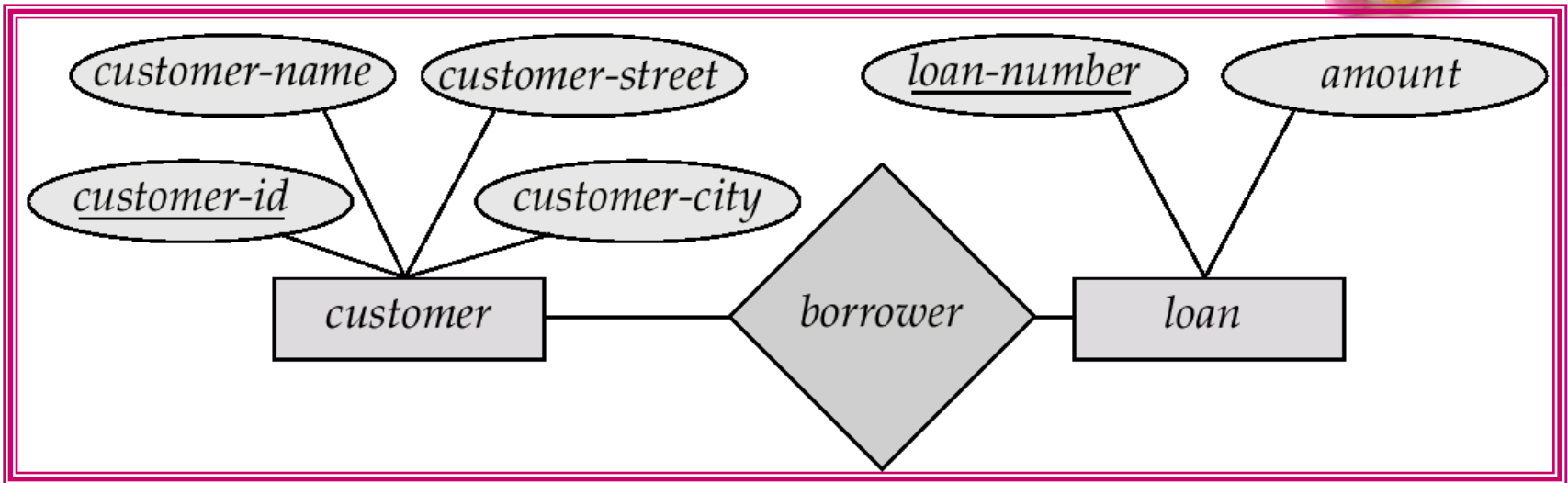


Relationship Sets (Cont.)

- 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*

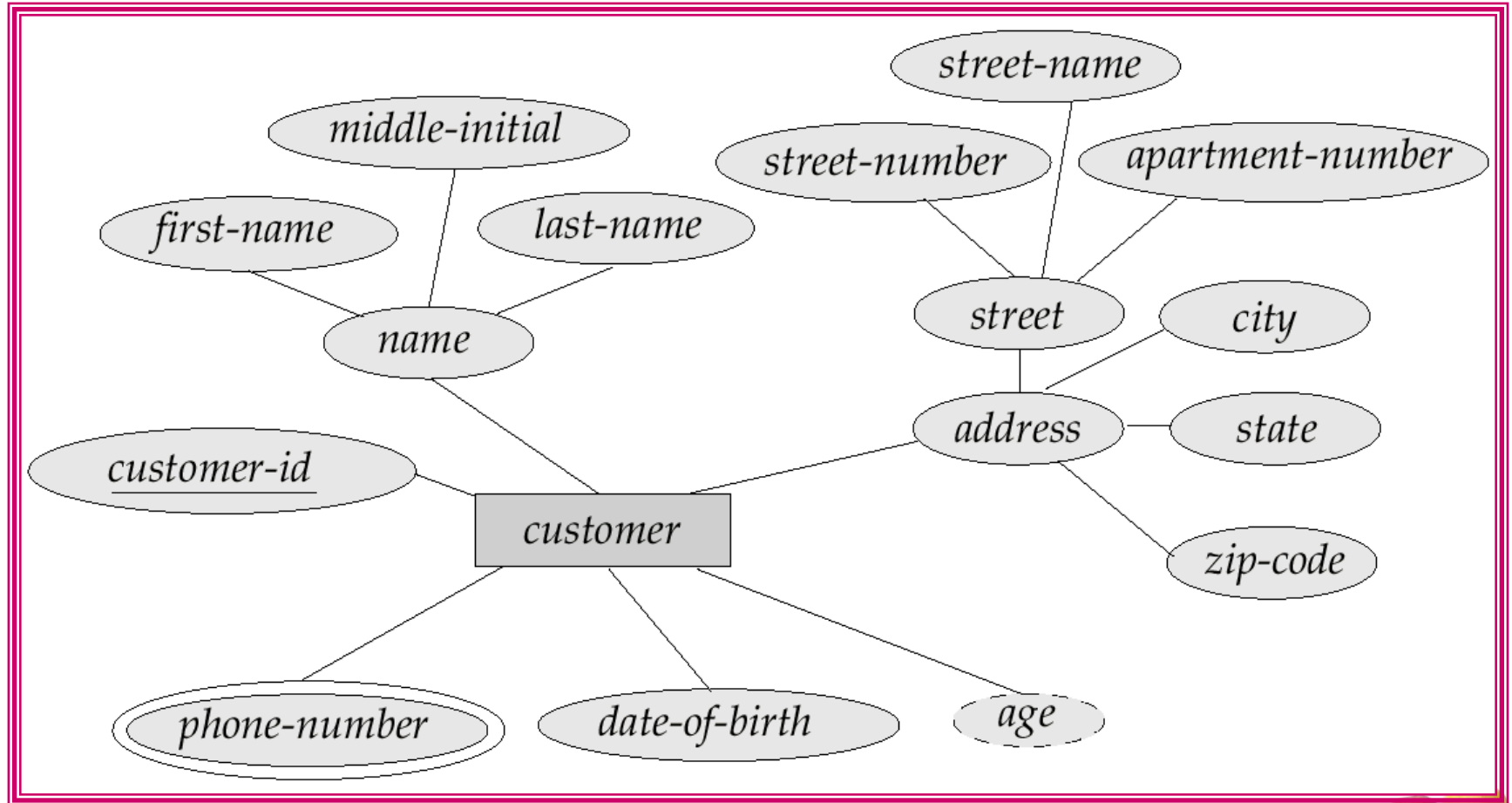


E-R Diagrams

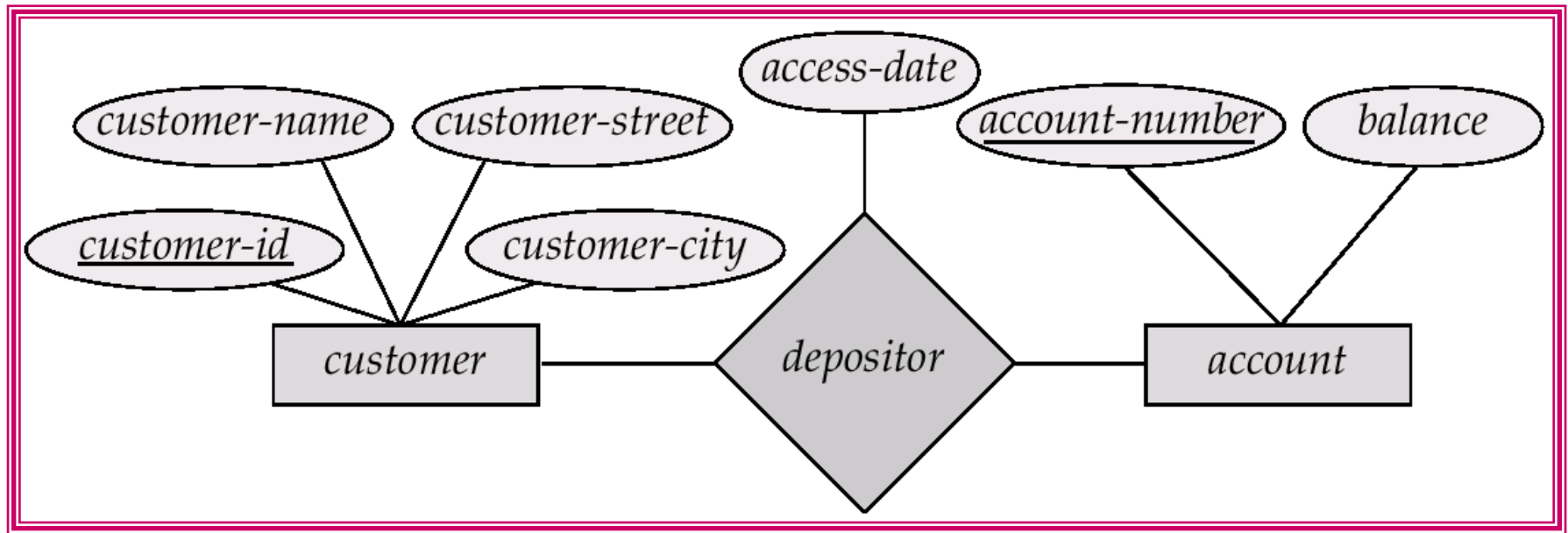


- **Rectangles** represent entity sets.
- **Diamonds** represent relationship sets.
- **Lines** link attributes to entity sets and entity sets to relationship sets.
- **Ellipses** represent attributes
 - **Double ellipses** represent multivalued attributes.
 - **Dashed ellipses** denote derived attributes.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

Entity with Composite, Multivalued, and Derived Attributes



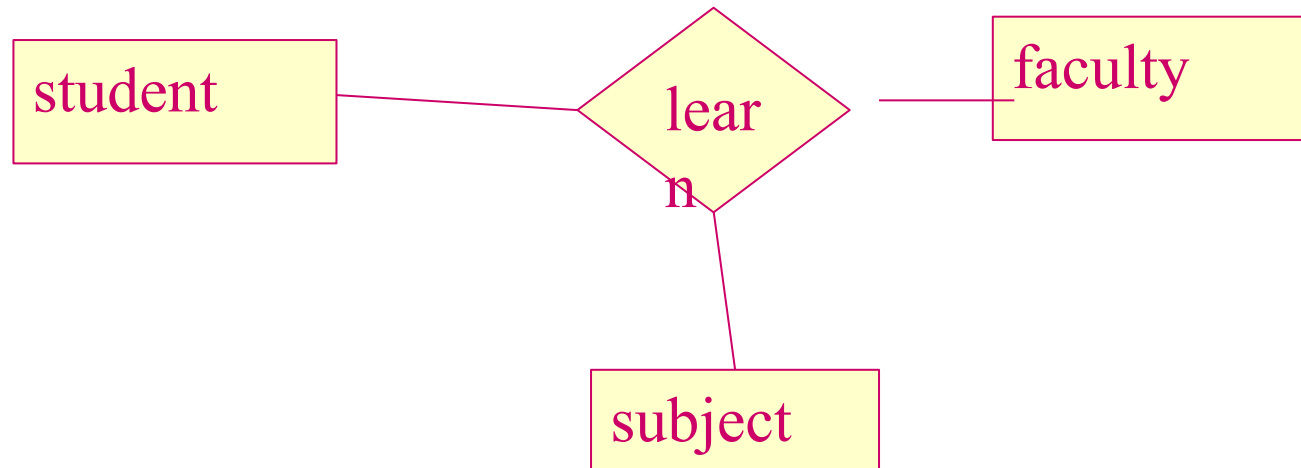
Relationship Sets with Attributes



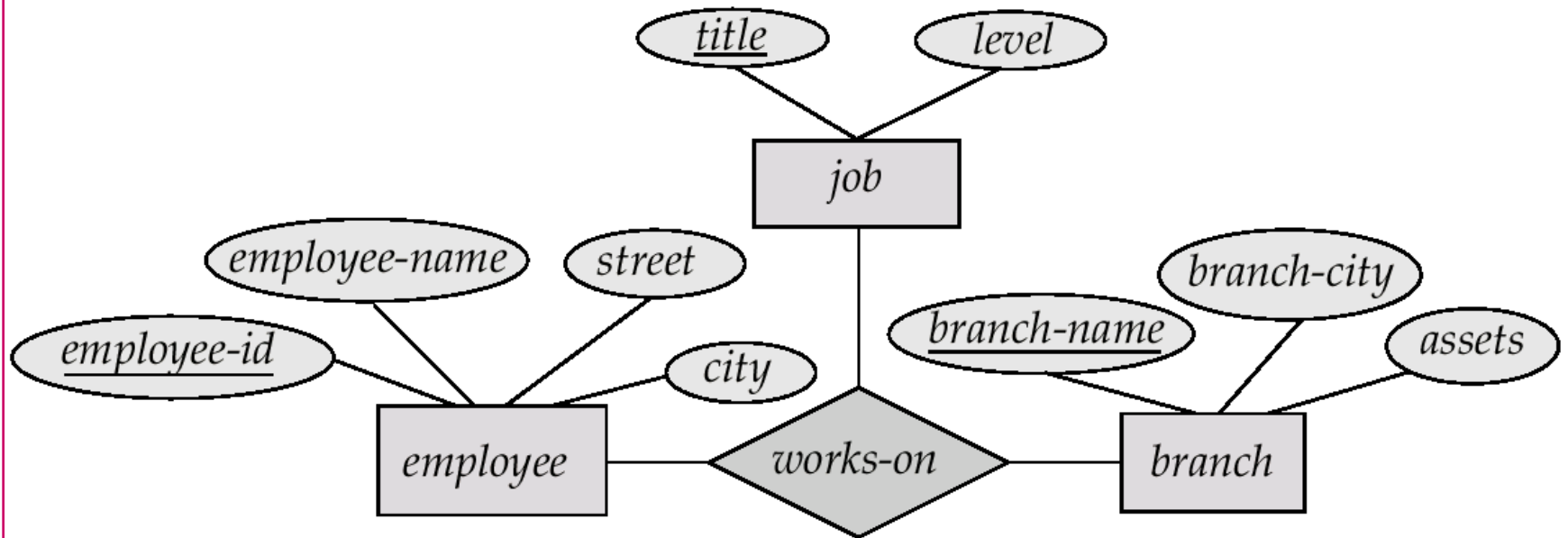
Degree of a Relationship Set

- 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.

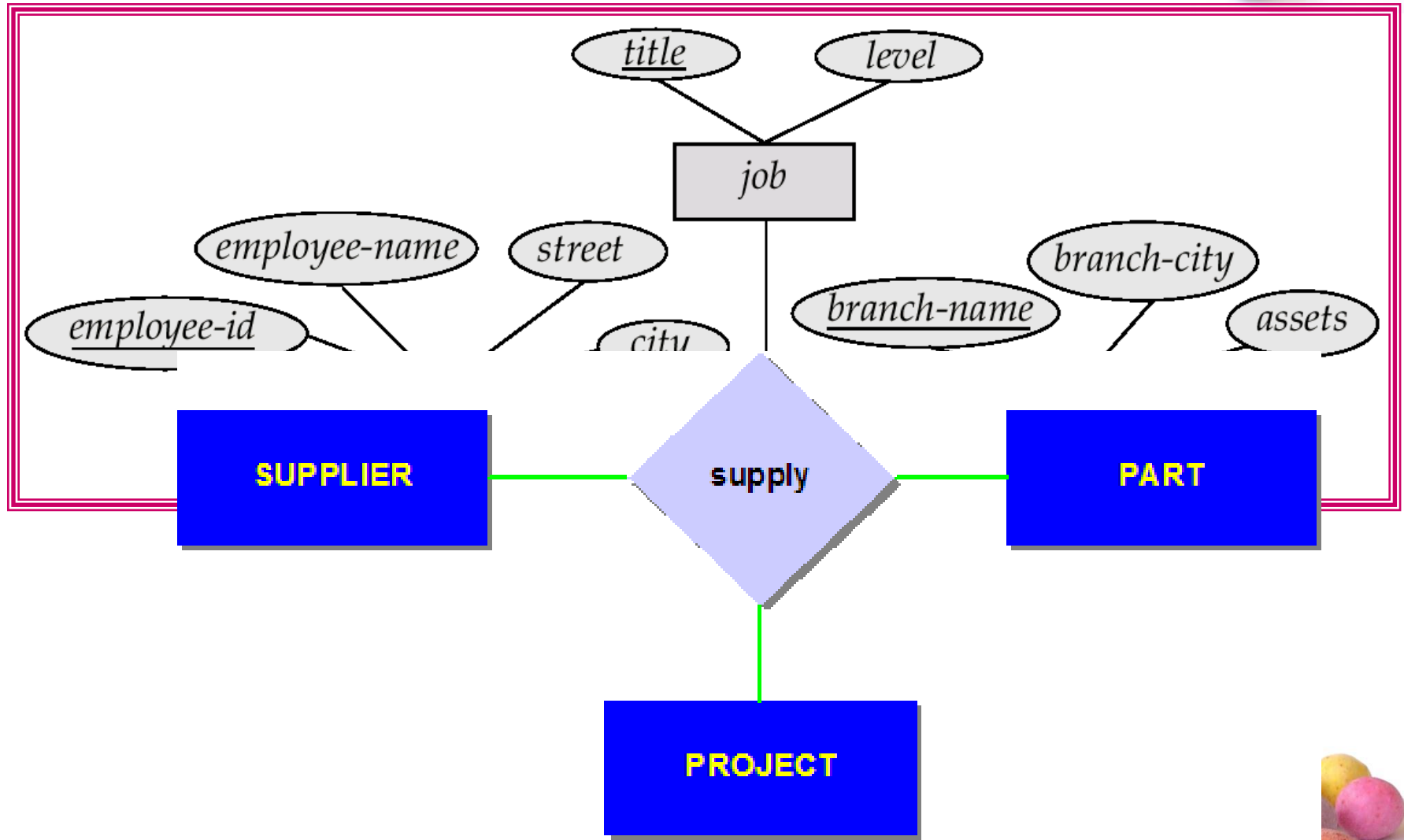
- Turnery relationship n-ary relationship



E-R Diagram with a Ternary Relationship

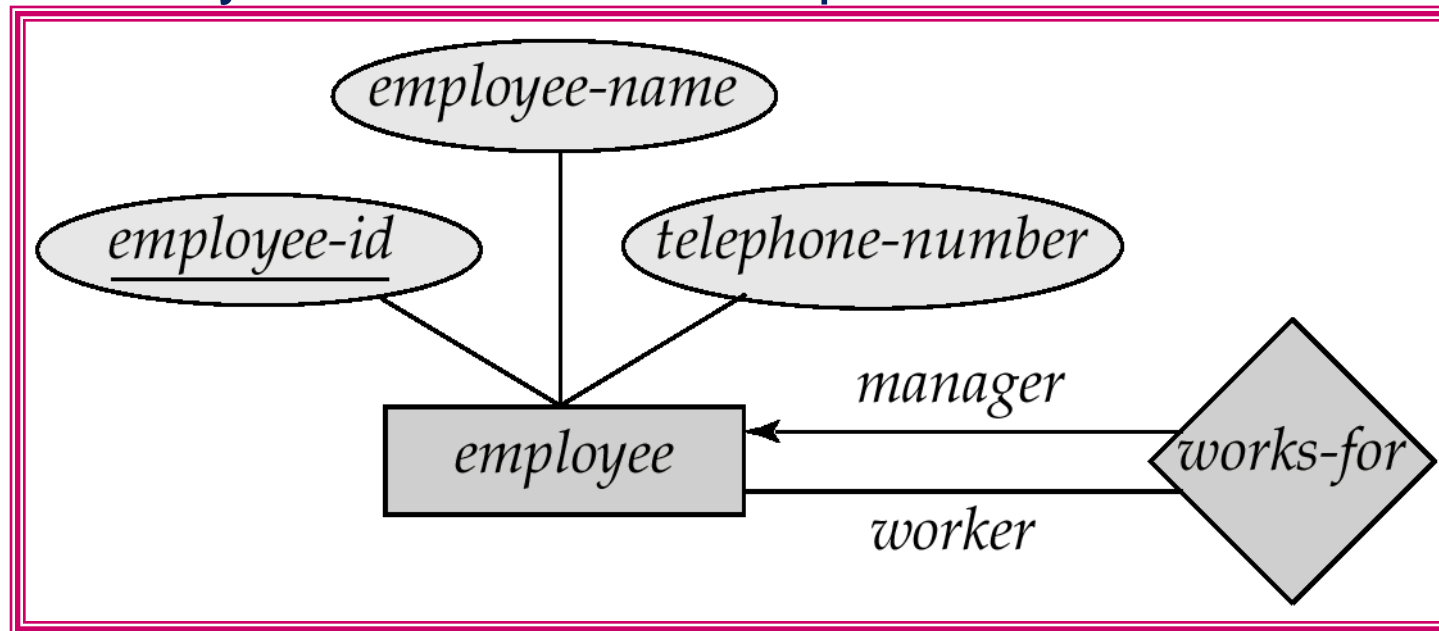


E-R Diagram with a Ternary Relationship



Roles

- Role is a function , that entity plays in relationship.
- 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
- Entity sets of a relationship need not be distinct



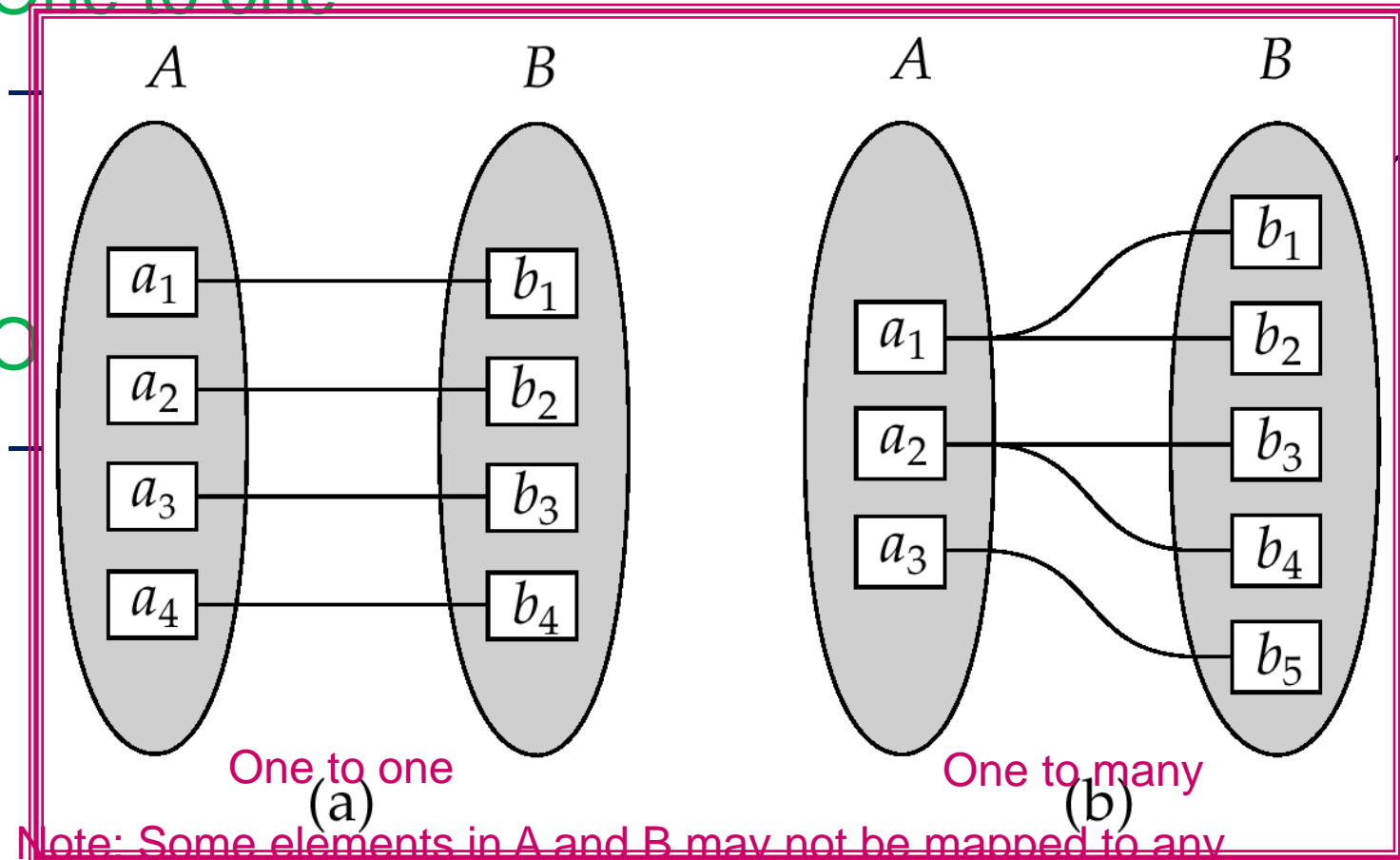
Mapping Cardinalities

- It Expresses 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



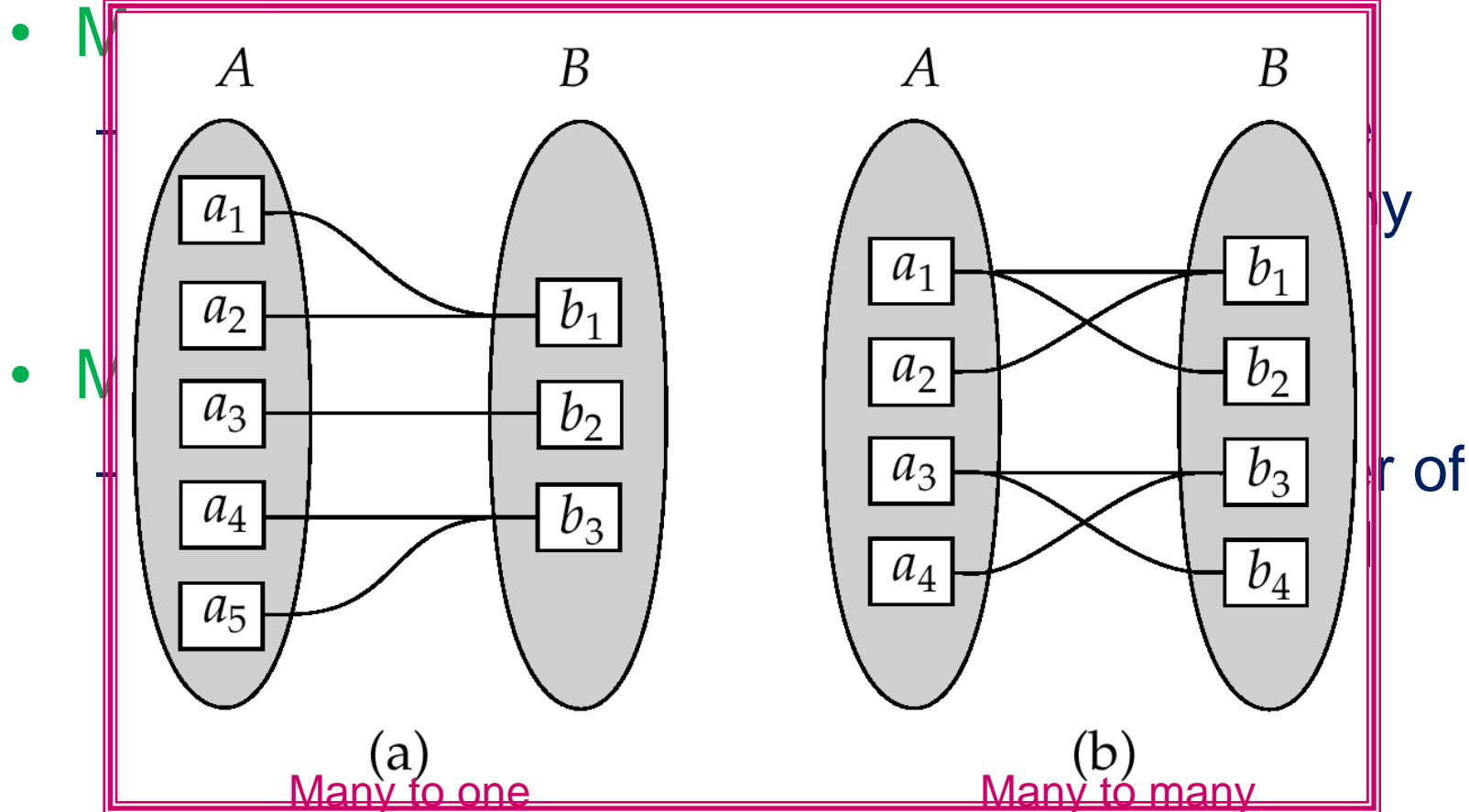
Mapping Cardinalities

- One to one



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

Mapping Cardinalities

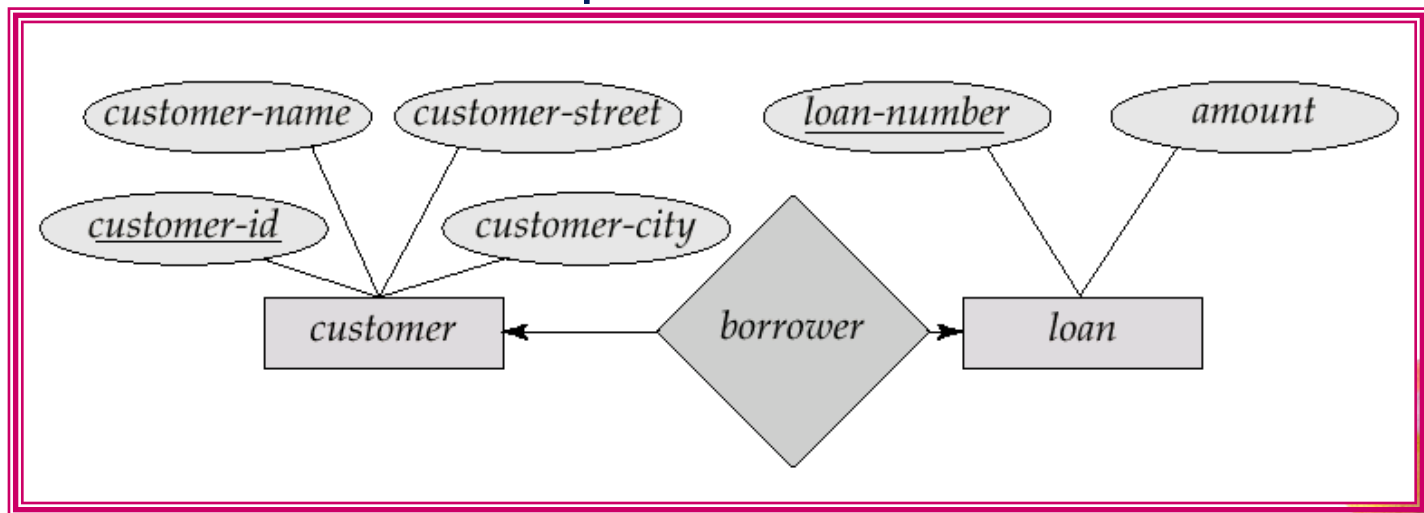


Note: Some elements in A and B may not be mapped to any elements in the other set



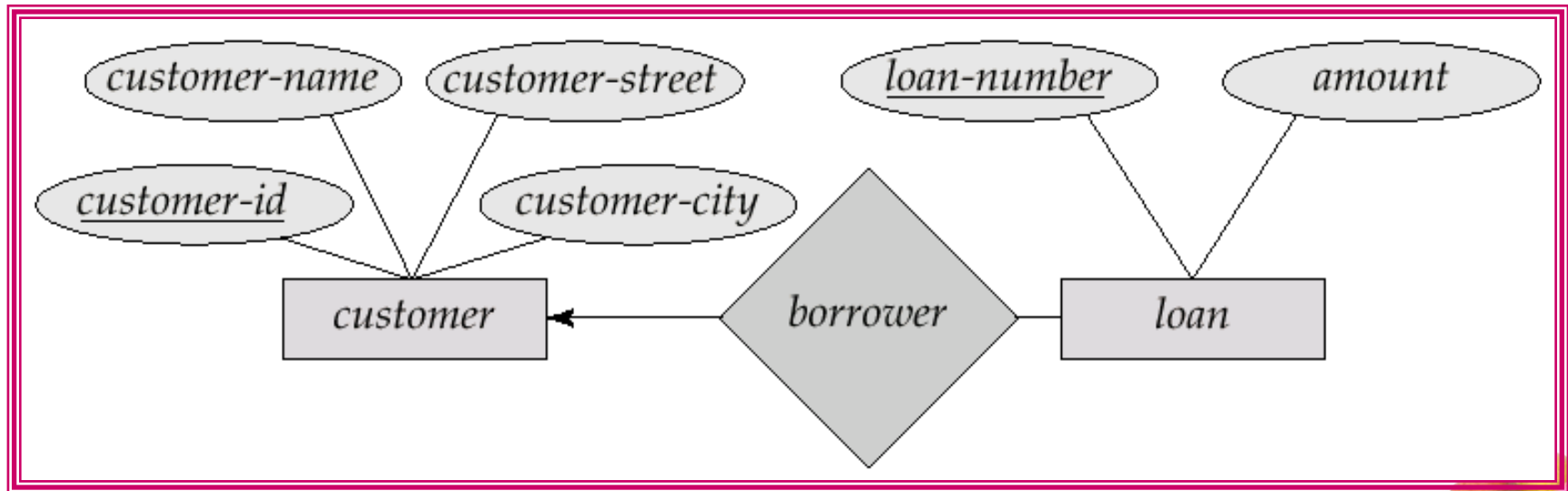
Cardinality Constraints

- We express cardinality constraints by drawing either a directed line (\rightarrow), 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*



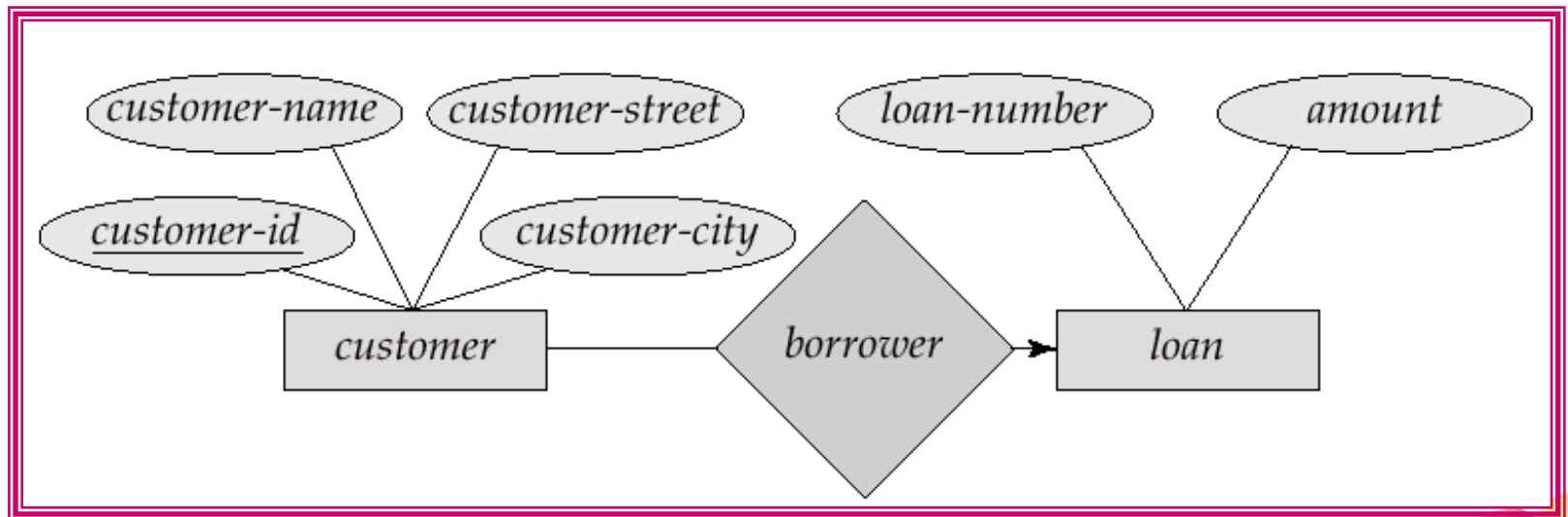
One-To-Many Relationship

- 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

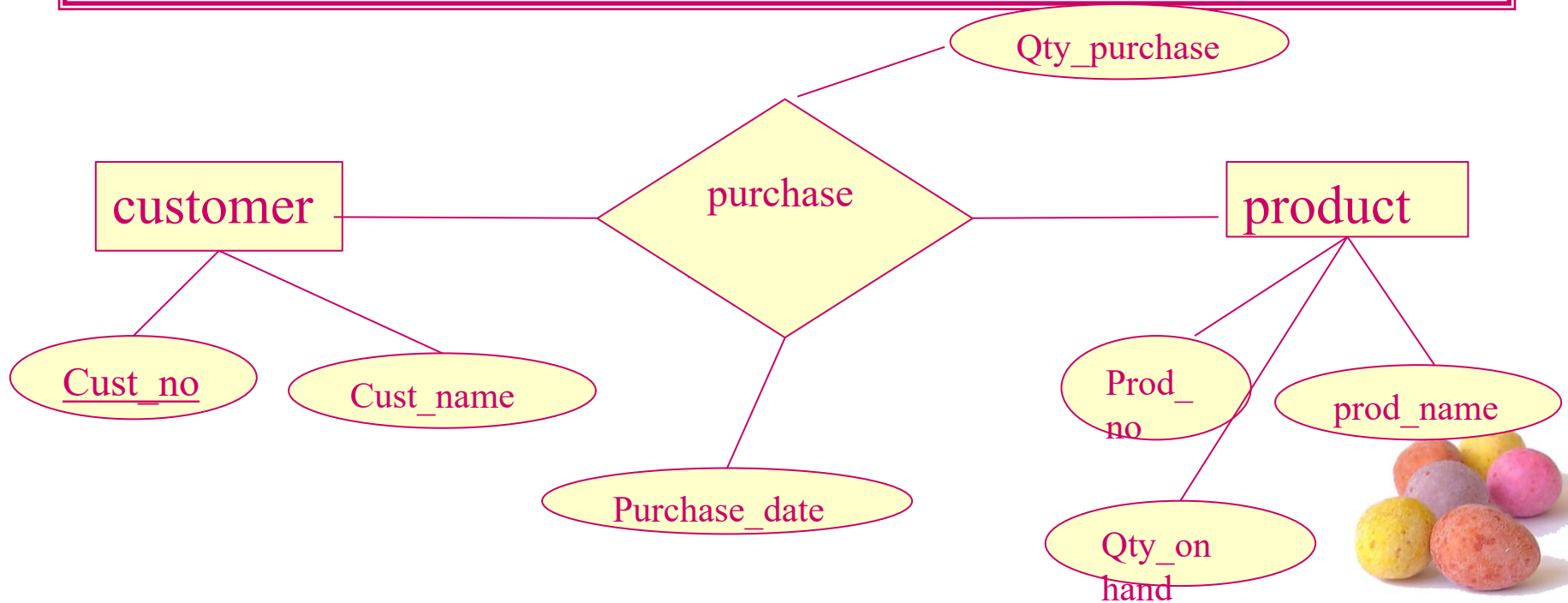
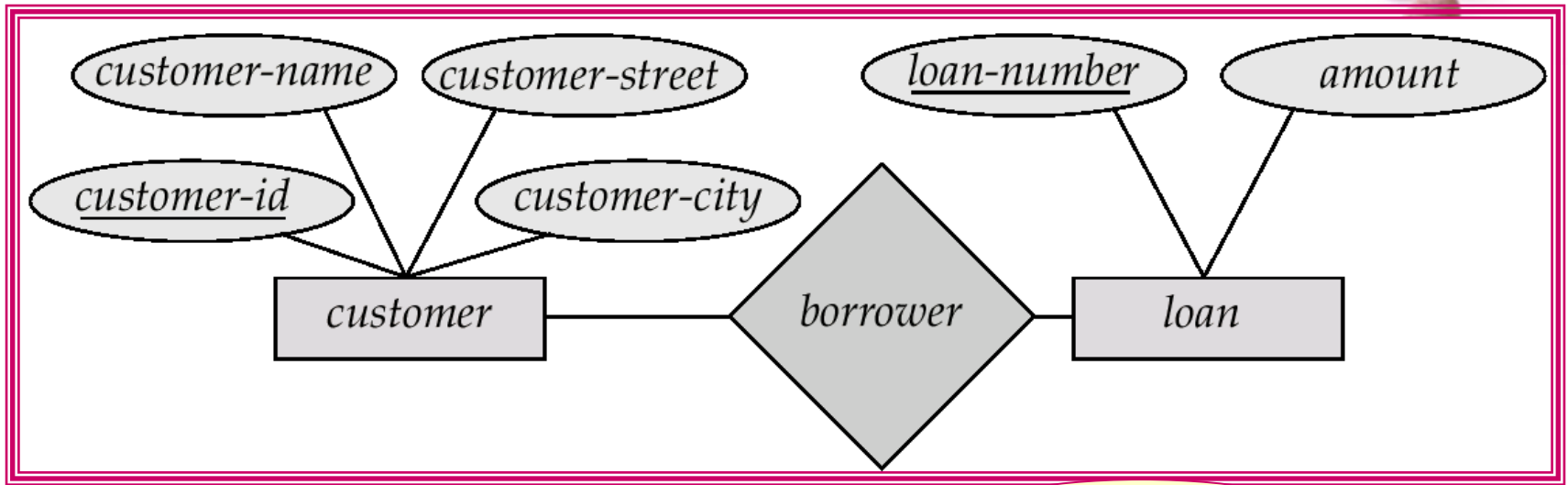
- In a many-to-one relationship a loan is associated with several (including 0) customers via *borrower*, a customer is associated with at most one loan via *borrower*



- Two entity students and college.
- one students can joined in one college
- One college can make joined in many students.



Many-To-Many Relationship





- One customer can purchase many products. (t-shirt, jeans, mobile)
- One product can be purchased by many customer. [t-shirt can be purchased by cust1 and cust2]
- Purchase_date and Qty_purchase are descriptive attribute.



Participation of an Entity Set in a Relationship Set

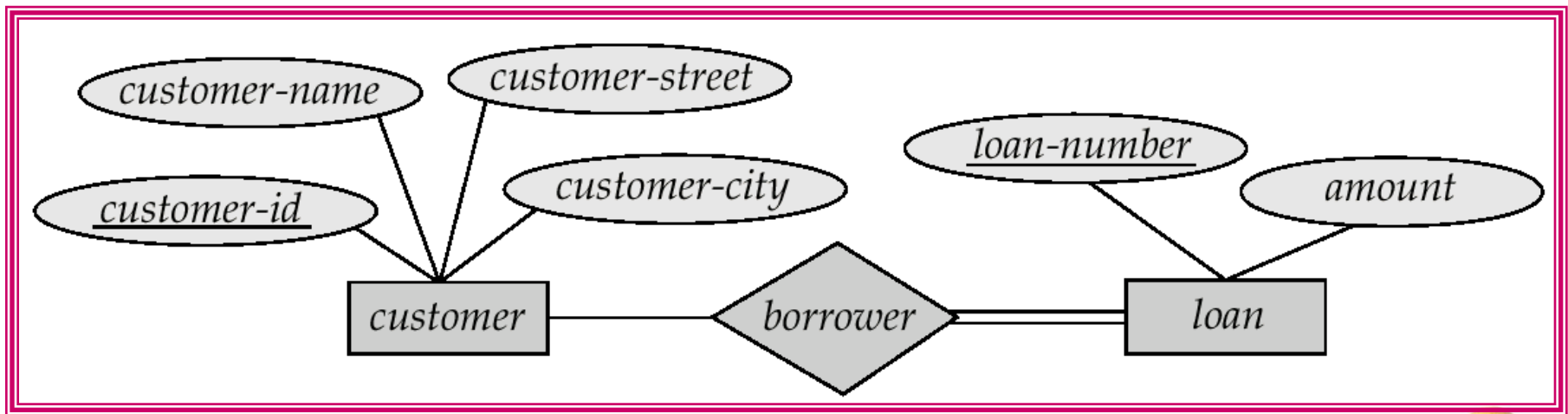
Total participation (indicated by double line): every entity in the entity set participates in at least one relationship instance in the relationship set

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



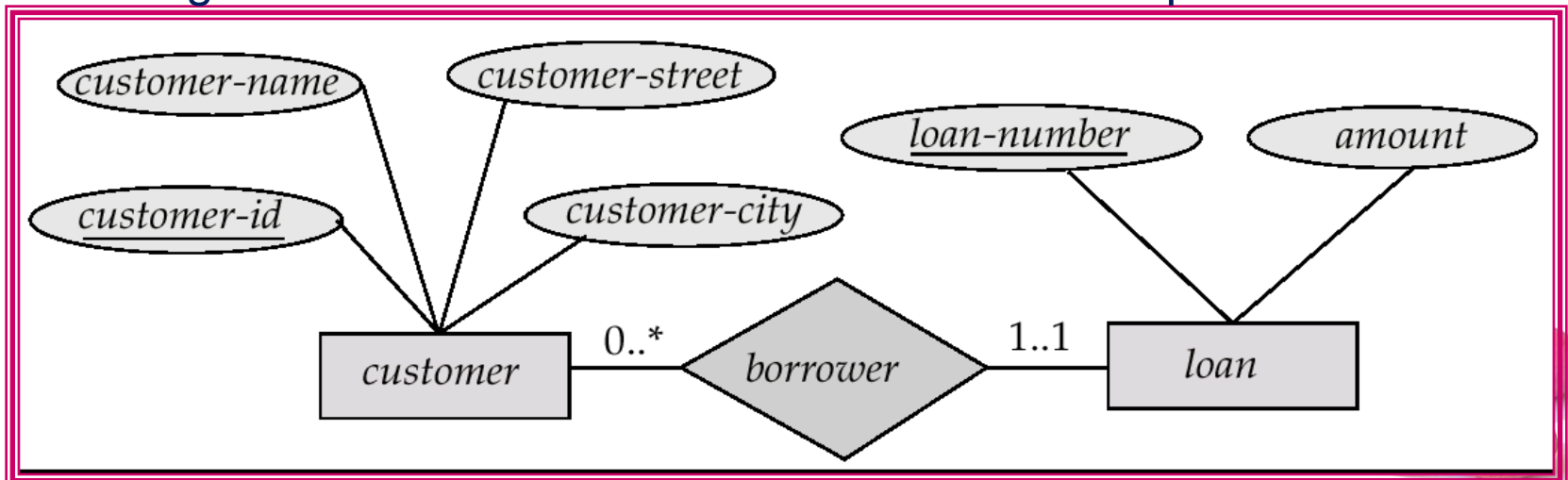
- Double line between the entity set “Student” and relationship set “Enrolled in” signifies total participation.
- It specifies that each student must be enrolled in at least one course.



- Single line between the entity set “Course” and relationship set “Enrolled in” signifies partial participation.
- It specifies that there might exist some courses for which no enrollments are made.

Alternative Notation for Cardinality Limits

- Cardinality limits can also express participation constraints.
- An edge between an entity set and relationship set can have an associated minimum and maximum cardinality.
- **l .. h** l is lower limit and h is higher limit.
- A minimum value 1 indicate a total participation of entity set into the relationship.
- in example each loan must have atleast one associated customer.
- Maximum /higher limit decide the cardinality of entity set. Like if both the side higher limit is 1 then it is one to one relationship





- Student enrolled in atleast one course. And can join more than one course So 1..*
- Some course dose not have any student enrollment.



- So minimum is 0 and maximux is * as for one course maximum many student can enrolled.



Keys

- A *super key* of an entity set is a set of one or more attributes whose values uniquely determine each entity instance. E.g cust_no is superkey for customer.
- Superkey may contains extra/more attributes.
- (e.g customer_name , cust_no)
- (customer_name,cust_no,cust_address)
- (cust_no,cust_address) are super keys





- 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*.



- E.g Employee have attribute
ssn,emp_id,emp_name

- So superkeys will be

- {ssn}

- {emp_id}

- {ssn,emp_id}

- {emp_id,emp_name}

- {ssn,emp_name}

- {ssn,emp_id,emp_name}

- **Candidate Key**

- {SSN}

- {Emp_id}

- Primary key can be any of the one from candidate key. i.e
ssn,emp_id can be a 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(relationship)*
 - *(Stud_id, course_id)* is super key of relationship set *enrolled-in*
 - *NOTE: this means a pair of entity sets can have at most one relationship in a particular relationship set.*





- 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



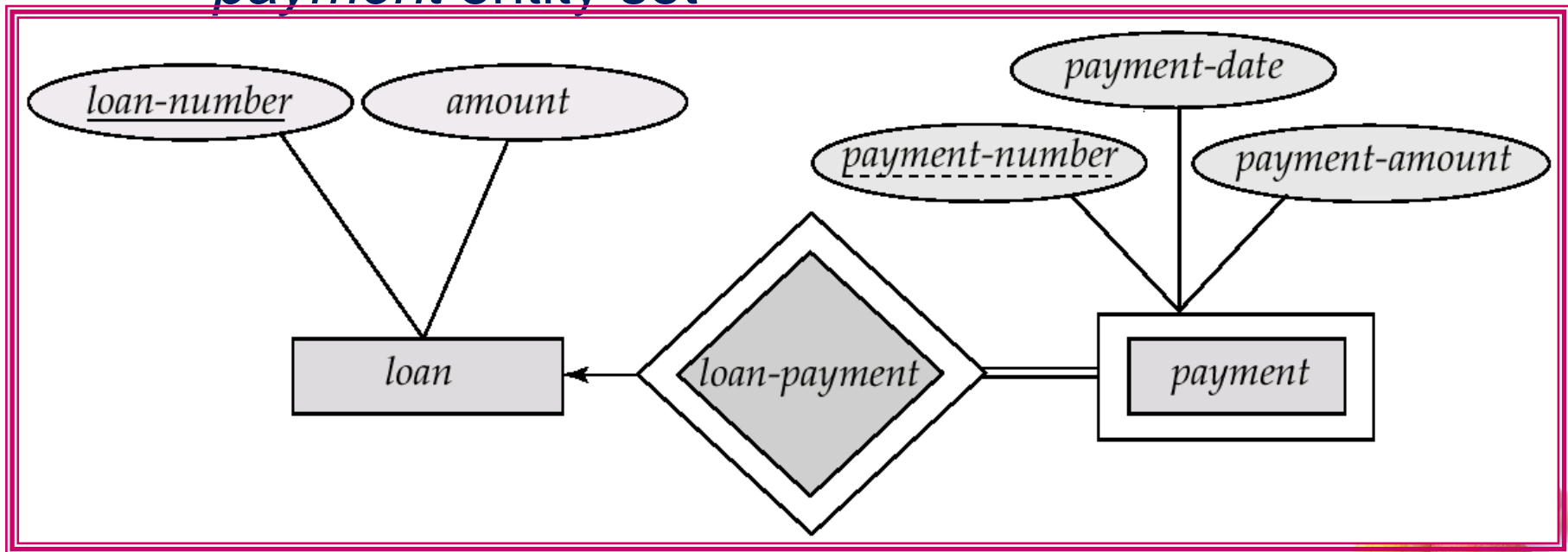
Weak Entity Sets

- 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/strong entity set*
 - *Identifying relationship* depicted using a double diamond
 - Weak entity depicted using double rectangle.
- 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.
- underline the discriminator of a weak entity set with a dashed line.
- *payment-number* – **discriminator** of the *payment* entity set





- loan_number, pay_number

1

1

1

2

2

1

sdfssf

2

2



Weak Entity Sets (Cont.)

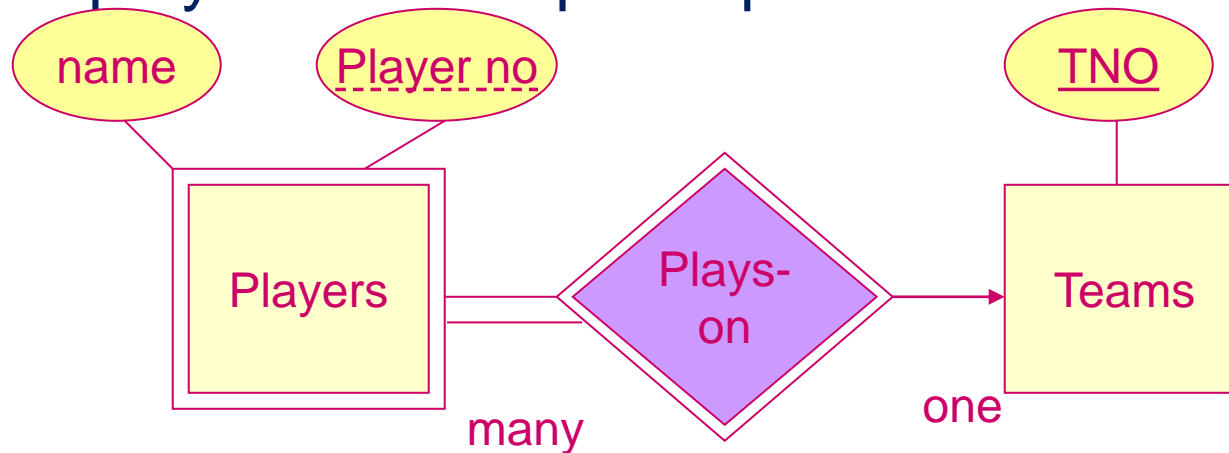


- Loan is strong entity
- Payment is weak entity. As every payment is is exists if loan is exists.
- Payment number is discriminator so strong entity's primary key loan_number and payment_number both combinedly find the uniq instance
- Payment entity have total participation has every loan have atleast minimum one payment installment.
- 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.



More Weak Entity Set Examples

- Team is strong entity having team number.
- Player is weak entity.
- Without team player does not exist. [player does not play independently without team]
- At least minimum one player for play. So player to play-on is total participation

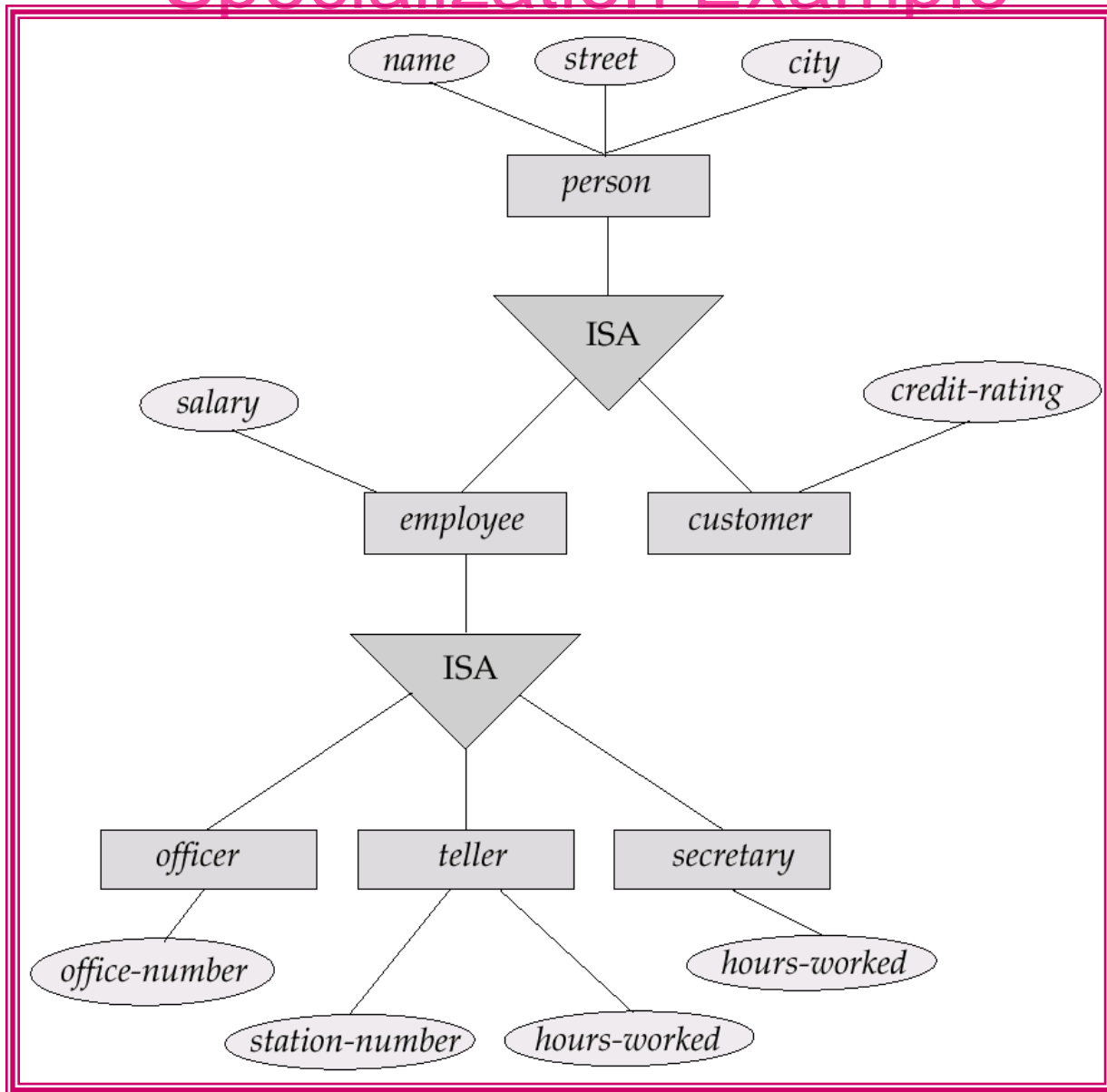


Specialization

- 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



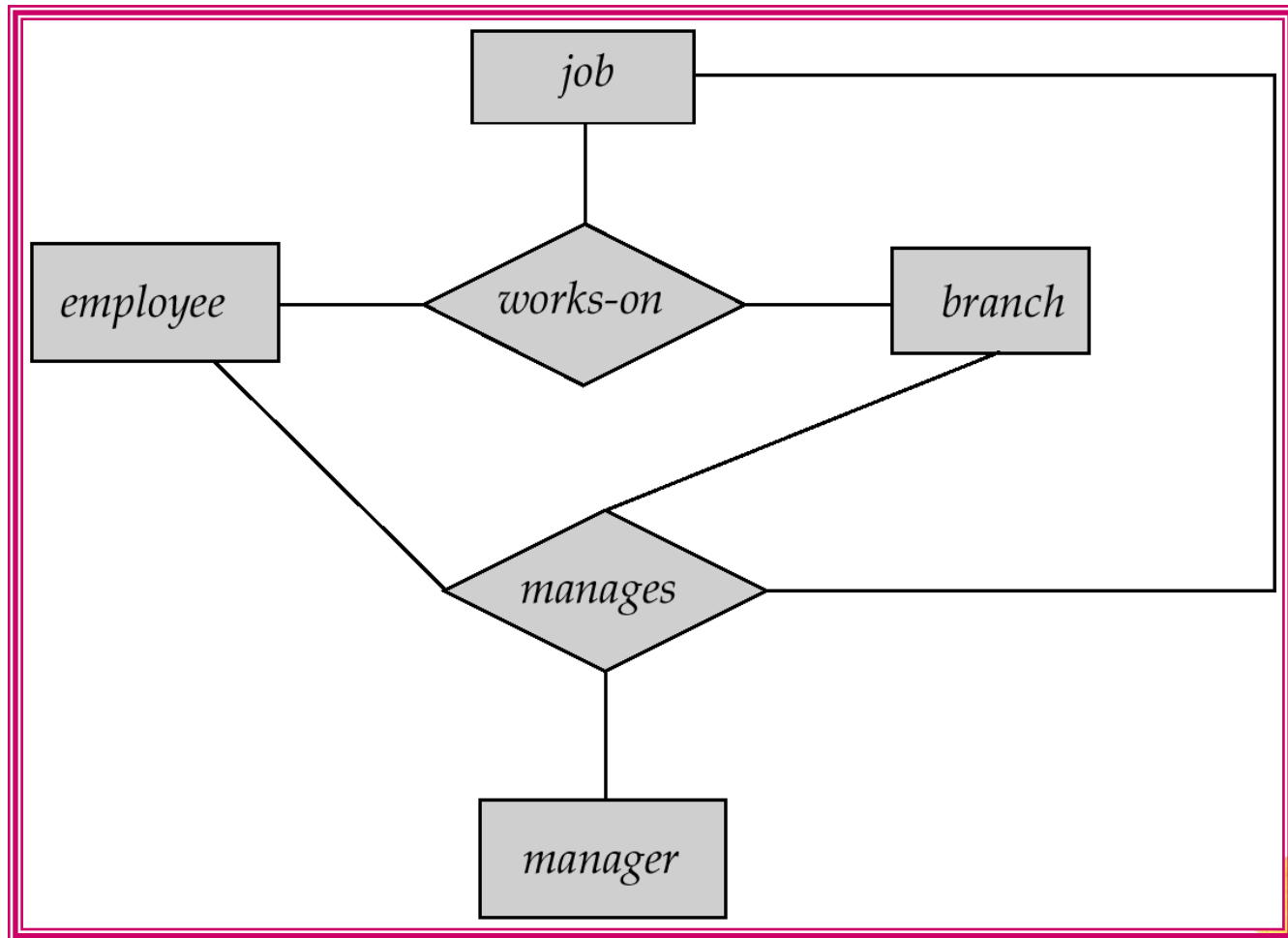
- A bottom-up design process – combine a number of entity sets that share the same features into a higher-level entity set.
- Specialization and generalization are simple inversions of each other; they are represented in an E-R diagram in the same way.
- The terms specialization and generalization are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



Aggregation

Consider the ternary relationship *works-on*, which we saw earlier

Suppose we want to record managers for tasks performed by an employee at a branch

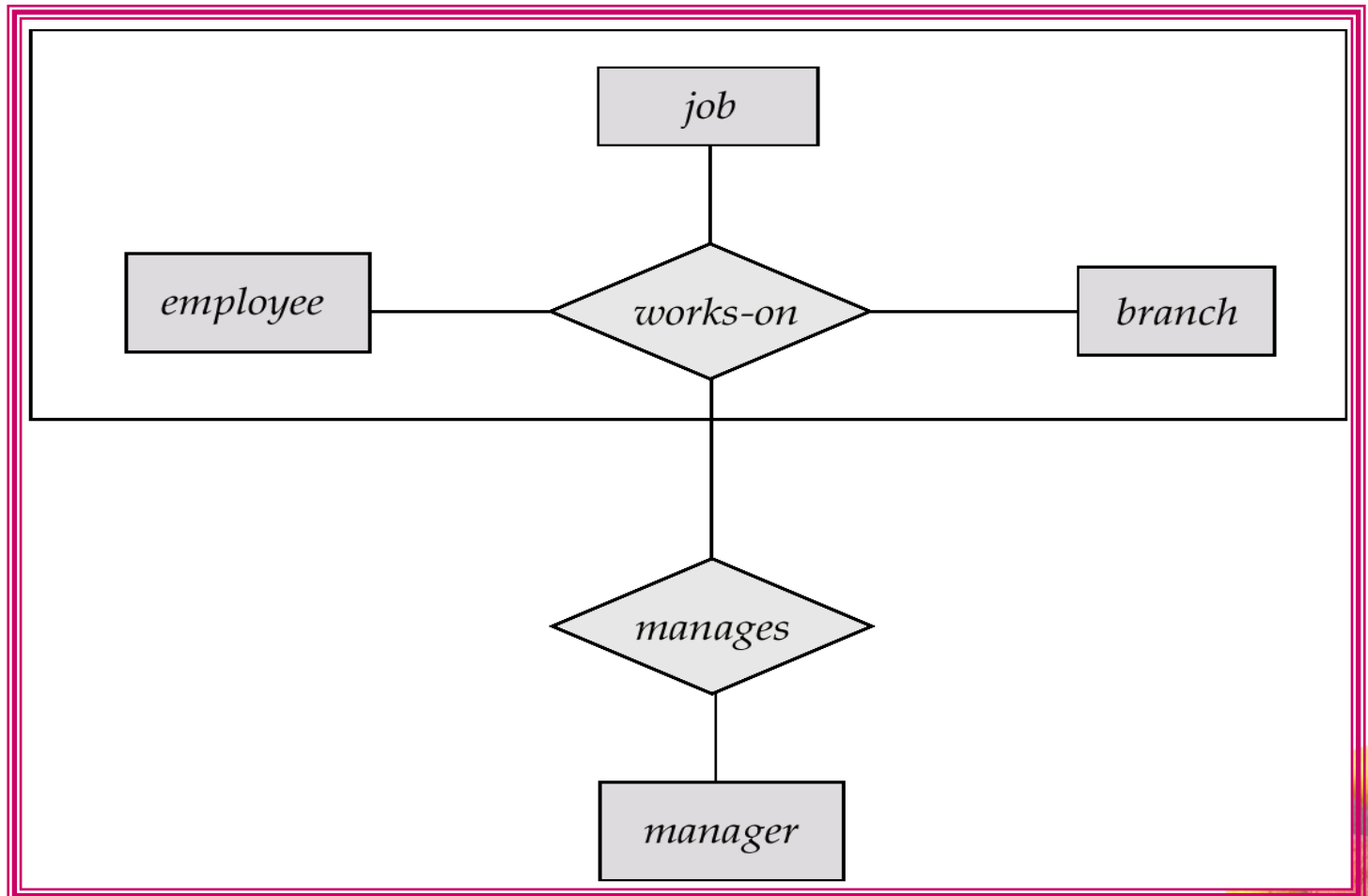


Aggregation (Cont.)

- Relationship sets *works-on* and *manages* represent overlapping information
 - Every *manages* relationship corresponds to a *works-on* relationship
 - However, some *works-on* relationships may not correspond to any *manages* relationships
 - So we can't discard the *works-on* relationship
- Eliminate this redundancy via *aggregation*
 - Treat relationship as an abstract entity
 - Abstraction of relationship into new entity

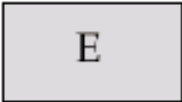


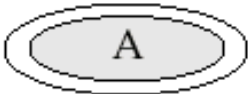



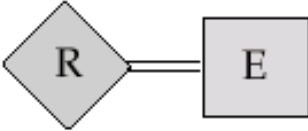
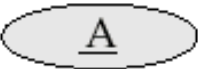
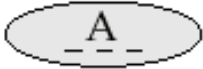


- An employee works on a particular job at a particular branch
- An employee, branch, job combination may have an associated manager



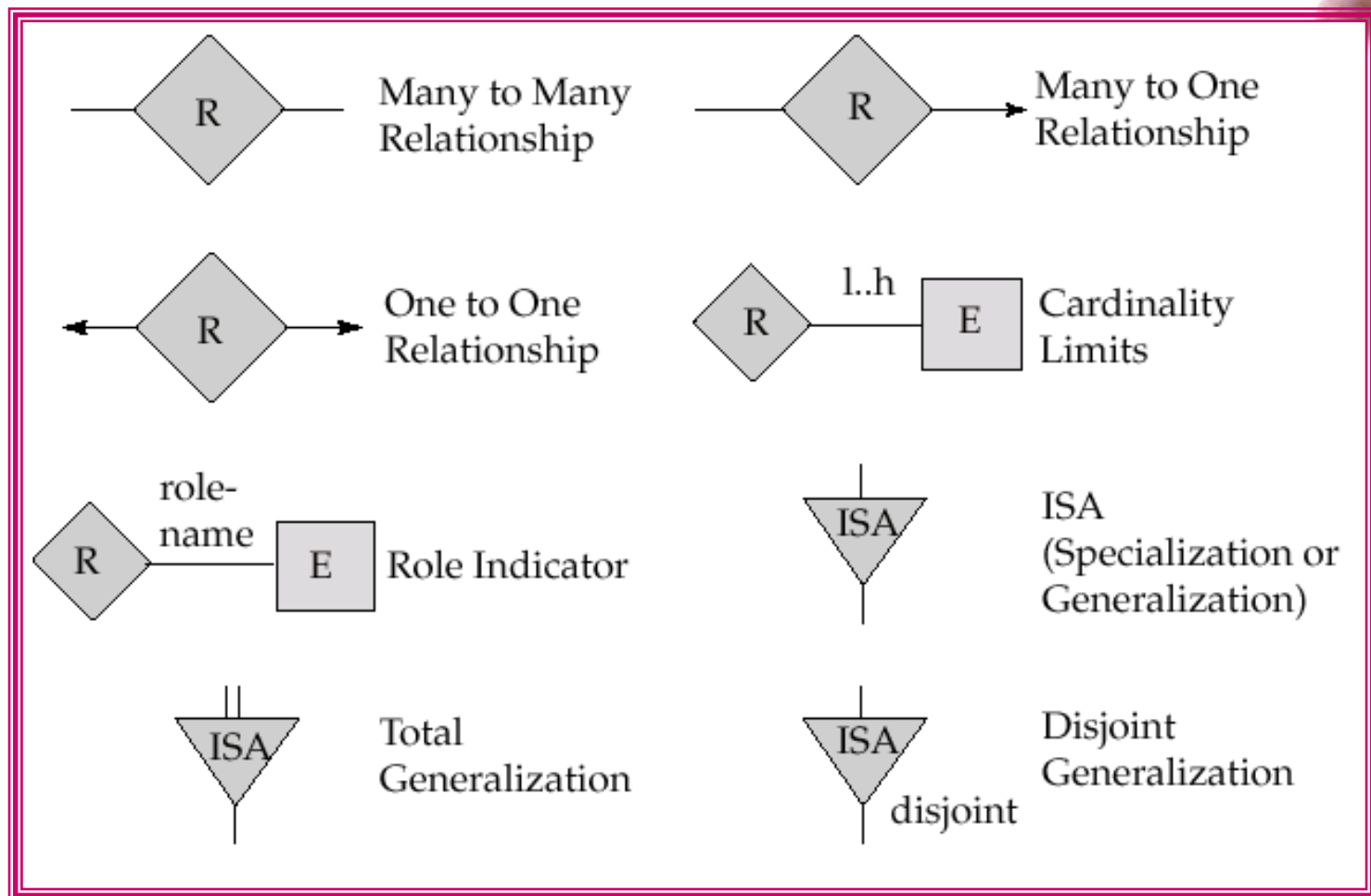
Summary of Symbols Used in E-R Notation



	Entity Set		Attribute
	Weak Entity Set		Multivalued Attribute
	Relationship Set		Derived Attribute
	Identifying Relationship Set for Weak Entity Set		Total Participation of Entity Set in Relationship
	Primary Key		Discriminating Attribute of Weak Entity Set



Summary of Symbols (Cont.)

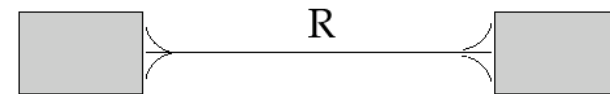
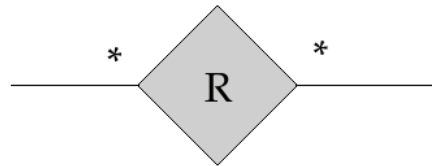


Alternative E-R Notations

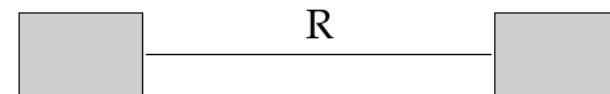
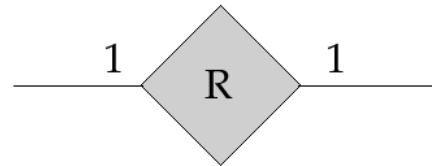
Entity set E with
attributes A1, A2, A3
and primary key A1

E	
A1	
A2	
A3	

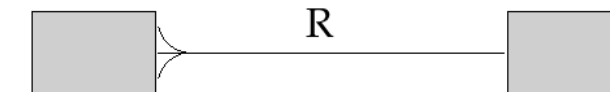
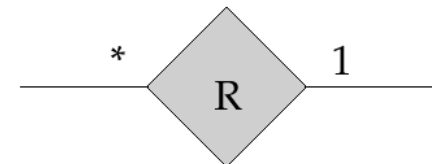
Many to Many
Relationship



One to One
Relationship



Many to One
Relationship



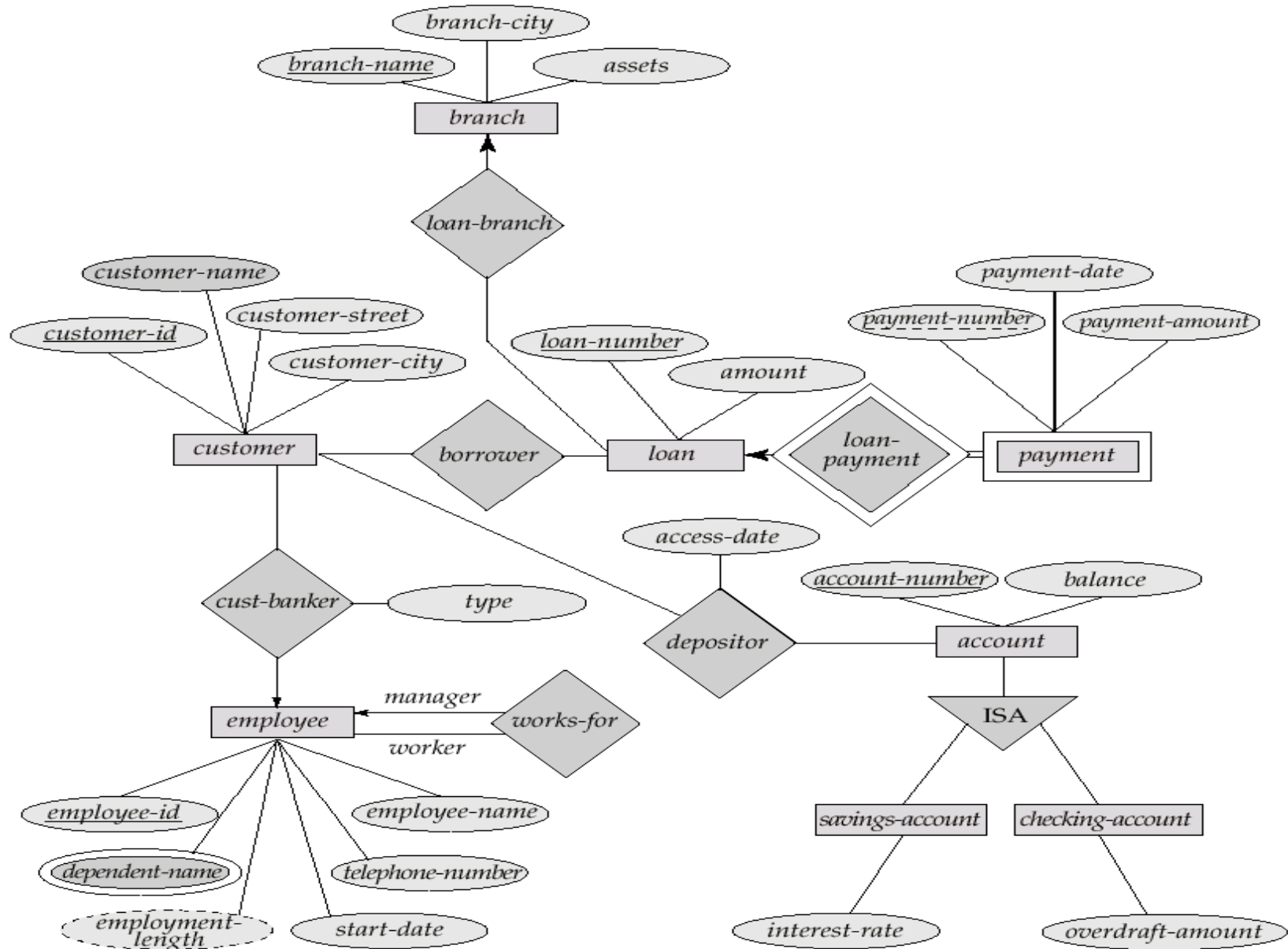
E-R Design Decisions



- 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



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.
- 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.

<i>customer-id</i>	<i>customer-name</i>	<i>customer-street</i>	<i>customer-city</i>
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



Composite and Multivalued Attributes

- Composite attributes are flattened out by creating a separate attribute for each component attribute
 - E.g. given entity set *customer* with composite attribute *name* with component attributes *first-name* and *last-name* the table corresponding to the entity set has two attributes
name.first-name and *name.last-name*
- A multivalued attribute M of an entity E is represented by a **separate table EM**
 - Table EM has attributes corresponding to the primary key of E and an attribute corresponding to multivalued attribute M





- E.g. Multivalued attribute *dependent-names* of *employee* is represented by a table *employee-dependent-names(employee-id, dname)*
- Each value of the multivalued attribute maps to a separate row of the table EM



Representing Weak Entity Sets

- A weak entity set becomes a table that includes a column for the primary key of the identifying strong entity set

<i>loan-number</i>	<i>payment-number</i>	<i>payment-date</i>	<i>payment-amount</i>
L-11	53	7 June 2001	125
L-14	69	28 May 2001	500
L-15	22	23 May 2001	300
L-16	58	18 June 2001	135
L-17	5	10 May 2001	50
L-17	6	7 June 2001	50
L-17	7	17 June 2001	100
L-23	11	17 May 2001	75
L-93	103	3 June 2001	900
L-93	104	13 June 2001	200

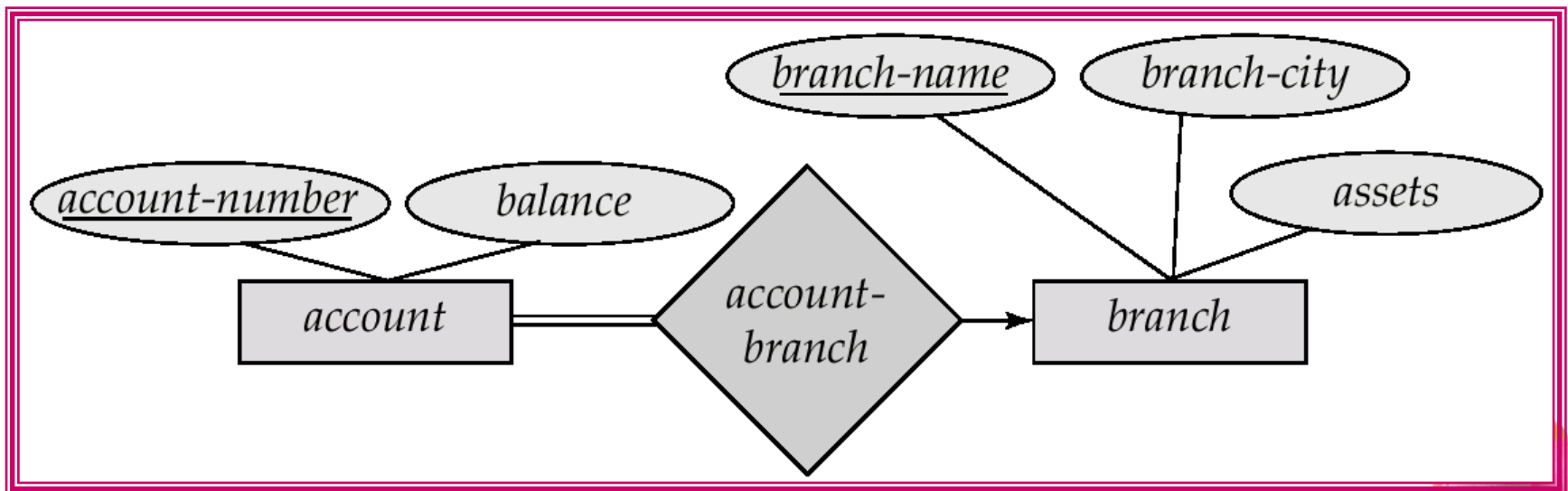
Representing Relationship Sets as Tables

- 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*

<i>customer-id</i>	<i>loan-number</i>
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

Redundancy of Tables

- Many-to-one and one-to-many relationship sets that are total on the many-side can be represented by adding an extra attribute to the many side, containing the primary key of the one side
- E.g.: Instead of creating a table for relationship *account-branch*, add an attribute *branch* to the entity set *account*



Redundancy of Tables (Cont.)



- For one-to-one relationship sets, either side can be chosen to act as the “many” side
 - That is, extra attribute can be added to either of the tables corresponding to the two entity sets
- If participation is *partial* on the many side, replacing a table by an extra attribute in the relation corresponding to the “many” side could result in null values
- The table corresponding to a relationship set linking a weak entity set to its identifying strong entity set is redundant.
 - E.g. The *payment* table already contains the information that would appear in the *loan-payment* table (i.e., the columns *loan-number* and *payment-number*).



Representing Specialization as Tables



- Method 1:
 - Form a table for the higher level entity
 - Form a table for each lower level entity set, include primary key of higher level entity set and local attributes

table	table attributes
<i>person</i>	<i>name, street, city</i>
<i>customer</i>	<i>name, credit-rating</i>
<i>employee</i>	<i>name, salary</i>

- Drawback: getting information about, e.g., *employee* requires accessing two tables



Representing Specialization as Tables (Cont.)



- Method 2:

- Form a table for each entity set with all local and inherited attributes

table	table attributes
<i>person</i>	<i>name, street, city</i>
<i>customer</i>	<i>name, street, city, credit-rating</i>
<i>employee</i>	<i>name, street, city, salary</i>

- If specialization is total, table for generalized entity (*person*) not required to store information
 - Can be defined as a “view” relation containing union of specialization tables
 - But explicit table may still be needed for foreign key constraints
- Drawback: street and city may be stored redundantly for persons who are both customers and employees

