ER Model

By :
Dr. Rinkle Rani
Associate Professor, CSED
TIET, Patiala

Keys of a Relation

ÅSuper Key

ÅCandidate Key

ÅPrimary Key

ÅAlternate Key

ÅArtificial Key

ÅForeign Key

Keys of a Relation

ÅSuper Key

ÅCandidate Key

ÅPrimary Key

ÅAlternate Key

ÅArtificial Key

ÅForeign Key

Super Key

A super key has the uniqueness property but not necessarily the irreducibility property.

For example if Roll_number is unique in relation STUDENT then, the set of attributes (Roll_number, Name, Class) is a super key for a relation STUDENT, these set of attributes are also
unique, but this combination of keys (composite key) is not having the property of irreducibility because Roll_number which is one subset of the composite key is also unique itself.
Thus, this composite key is called as super key because it has the property of uniqueness but not the irreducibilty.

Candidate Key

Candidate keys are those attributes of a relation, which have the properties of uniqueness and irreducibility.

Irreducibility: No proper subset of K has the uniqueness property.

Primary Key

Primary key is a candidate key choose by the designer for unique identification of records of a relation.

Primary key cannot contain any Null value because we cannot uniquely identify multiple Null values.

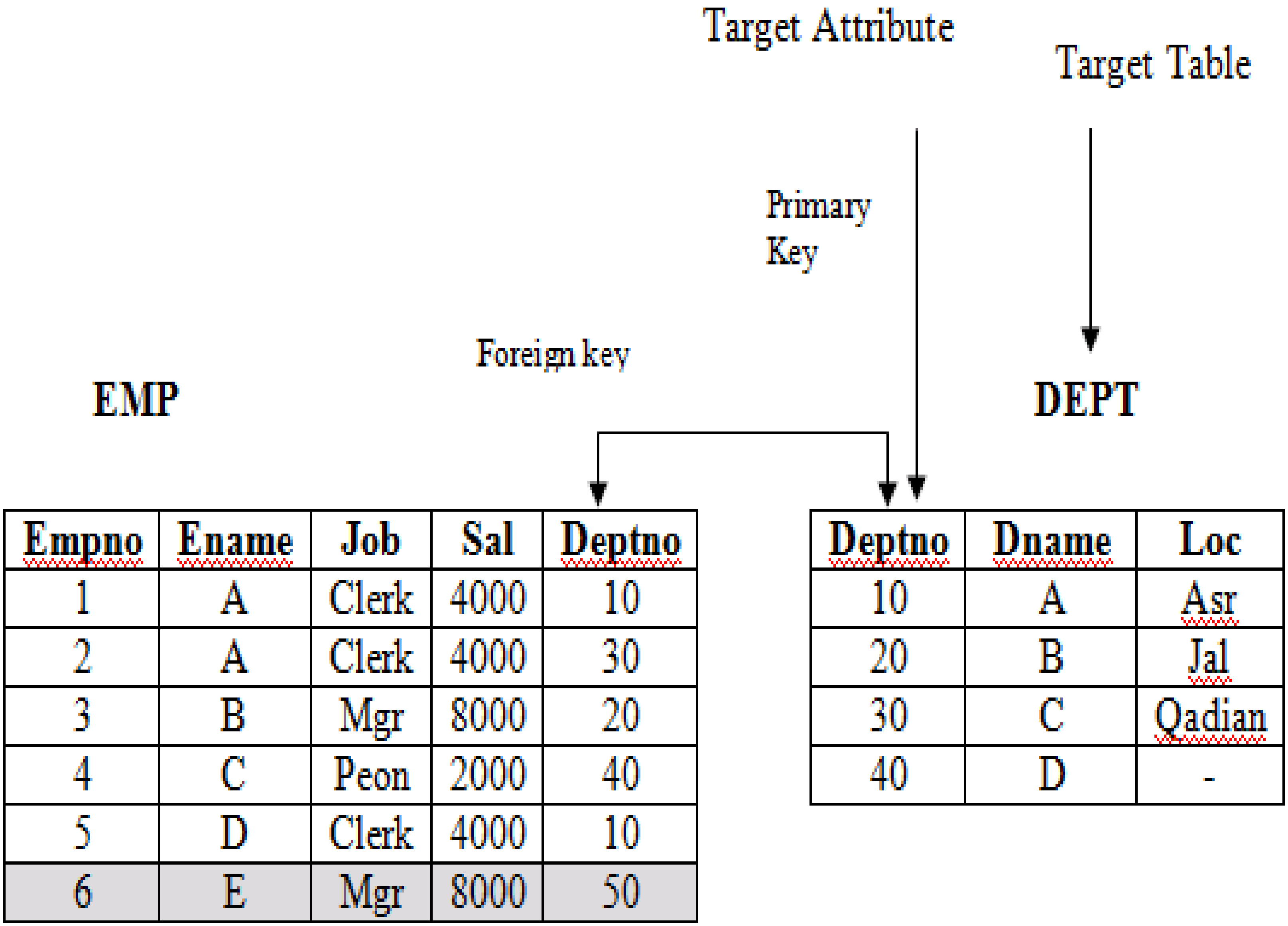
Alternate Key

The **alternate keys of any table are** simply those candidate keys, which are not currently selected as the primary key.

Foreign Key

Foreign keys are the attributes of a table, which refers to the primary key of some another table. Foreign Keys permit only those values, which appears in the primary key of the table to which it refers or may be null. Foreign keys are used to link together two or more different tables which have some form of relationship with each other. The foreign key is a reference to the tuple of a table from which it was taken, this tuple being called the **Referenced or Target tuple. The table containing the** referenced tuple will be called as **Target table.**

The matter of integrity of foreign keys is referred to as **Referential Integrity.**



Target Attribute

Target Table

Primary Key

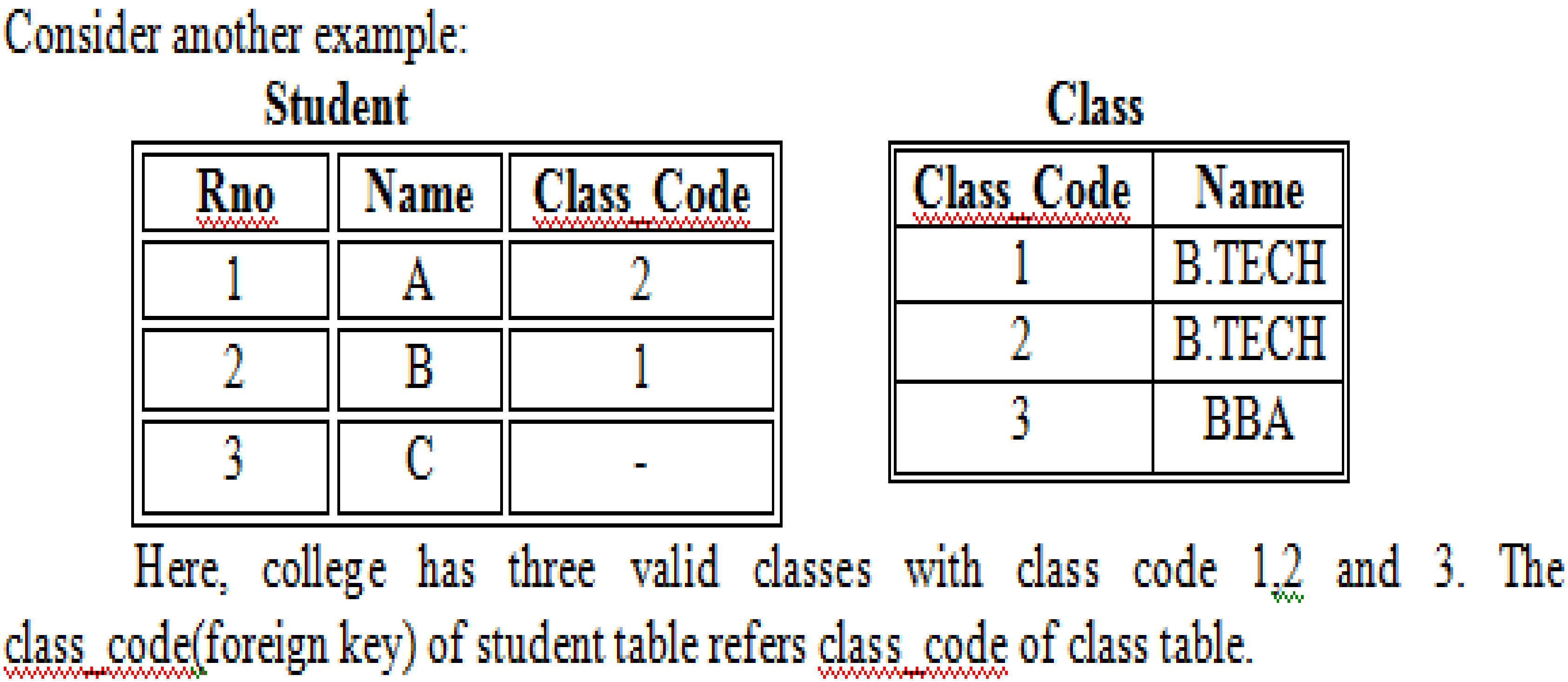
Foreign key

EMP

Empno	Ename	Job	Sal	Deptno
1	A	Clerk	4000	10
2	A	Clerk	4000	30
3	B	Mgr	8000	20
4	C	Peon	2000	40
5	D	Clerk	4000	10
6	E	Mgr	8000	50

DEPT

Deptno	Dname	Loc
10	A	Asr
20	B	Jal
30	C	Qadian
40	D	-



Consider another example:

Student

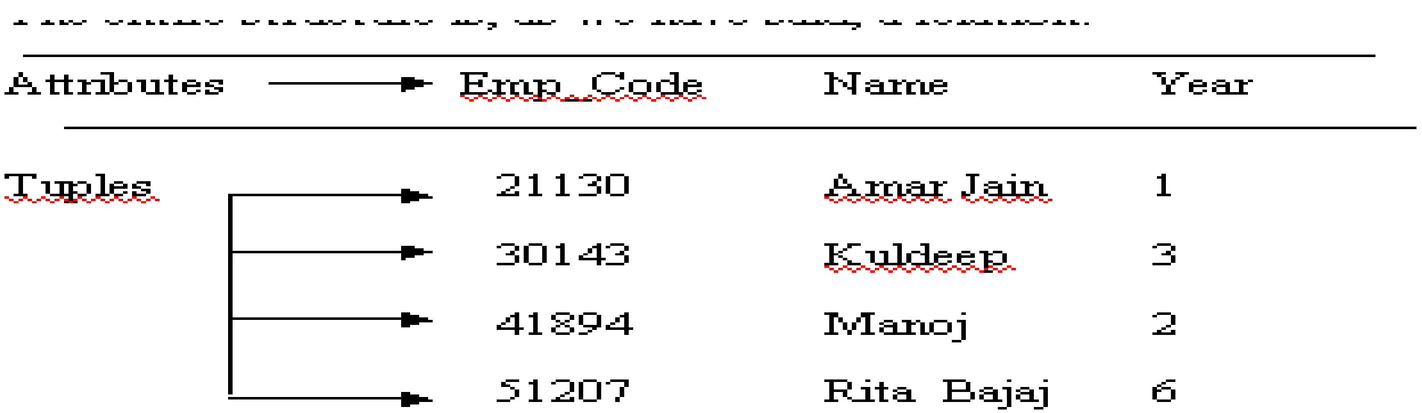
Rno	Name	Class_Code
1	A	2
2	B	1
3	C	-

Class

Class_Code	Name
1	B.TECH
2	B.TECH
3	BBA

Here, college has three valid classes with class code 1,2 and 3. The class_code(foreign key) of student table refers class_code of class table.

Other Terminologies



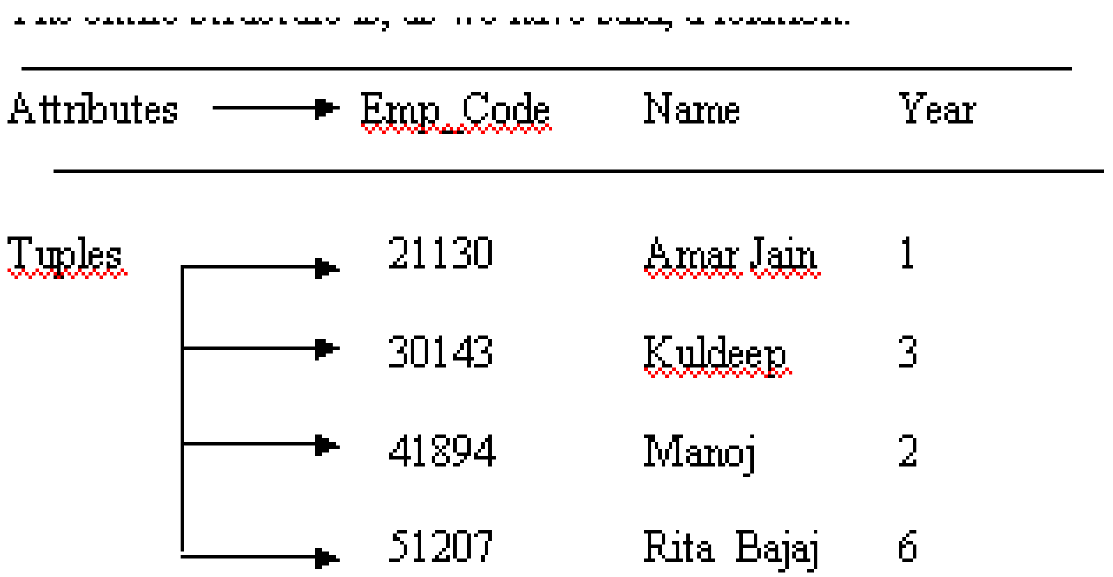
Attributes	Emp_Code	Name	Year
Tuples	21130	Amar Jain	1
	30143	Kuldeep	3
	41894	Manoj	2
	51207	Rita Bajaj	6

Cardinality of relation

The number of tuples in a relation determines its cardinality. In this case, the relation has a cardinality of 4.

Degree of a relation

Each column in the tuple (row) is called an attribute. The number of attributes in a relation determines its degree. The relation in has a degree of 3.



Attributes →	Emp_Code	Name	Year
Tuples →	21130	Amar Jain	1
→	30143	Kuldeep	3
→	41894	Manoj	2
→	51207	Rita Bajaj	6

Domains

A domain definition specifies the kind of data represented by the attribute. More particularly, a domain is the set of all possible values that an attribute may validly contain.

Designing of Database

Entity-Relationship Model

How to design the database?

There are two approaches

1. E-R Modeling: Identifying entity and relations
2. Normalization: Refinement of database designing

E-R Model

Å The Entity-Relationship (ER) model was originally proposed by Peter in 1976

Å The ER model is a conceptual data model that views the real world as entities and relationships.

Å A basic component of the model is the Entity Relationship diagram, which is used to visually represent data objects.

Basic Constructs of E-R Modeling

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 etc.

Entity Sets customer and loan

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

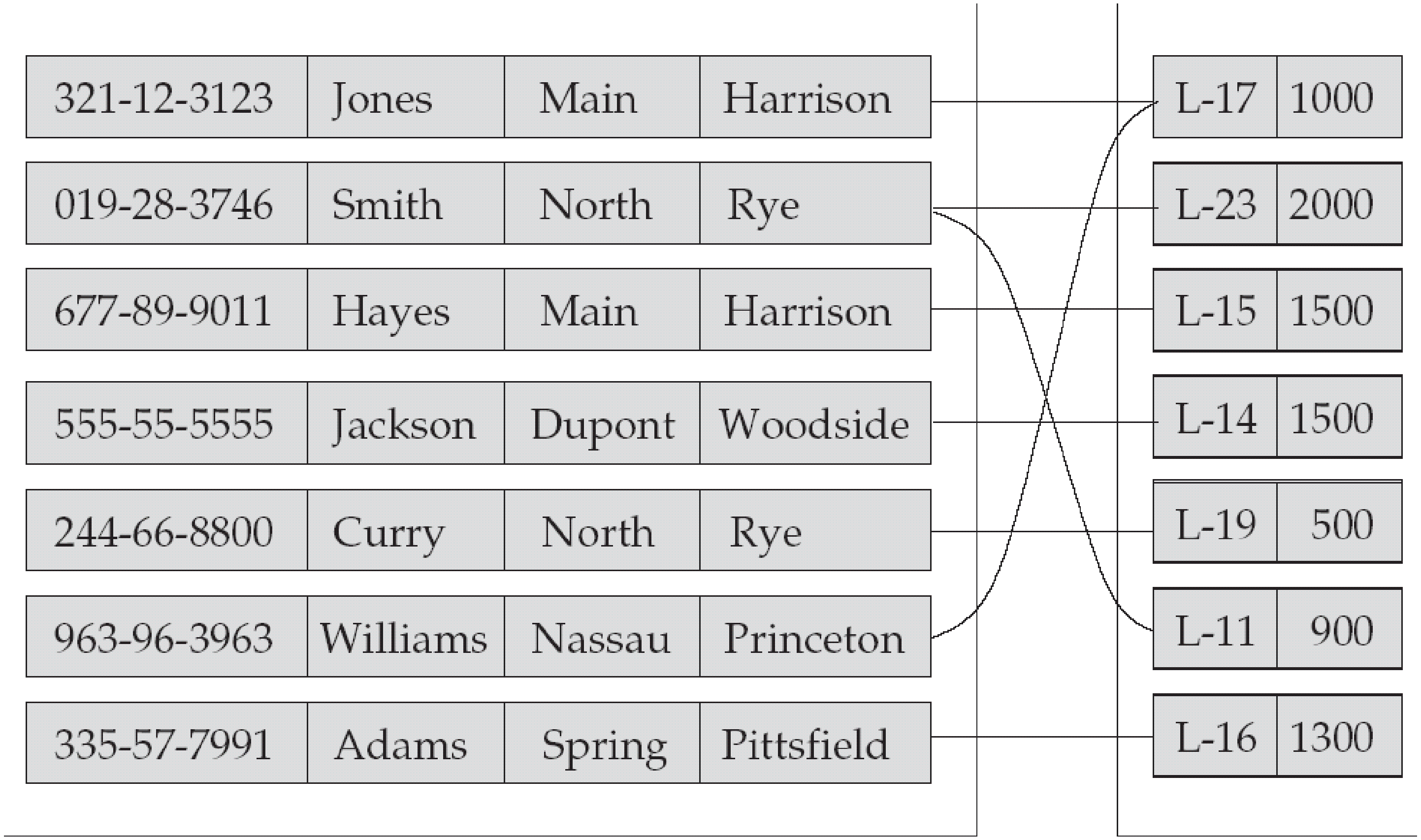
customer

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

loan

Relationship Set borrower

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



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

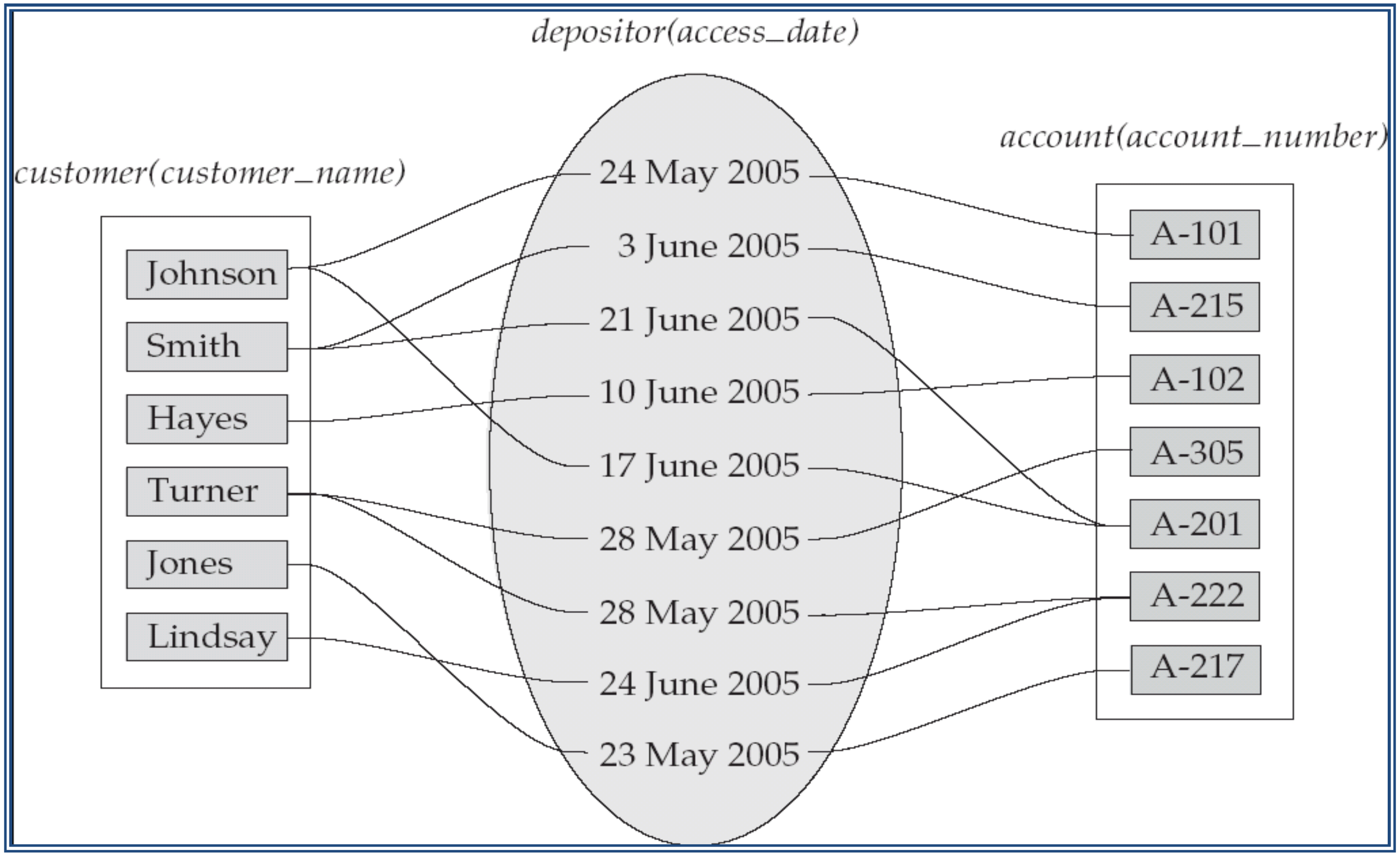
customer

loan

Relationship Sets (Cont.)

Å An **attribute** can also be property of a relationship set.

Å For instance, the *depositor* relationship set between entity sets *customer* and *account* may have the attribute *access-date*



Degree of a Relationship Set

Å Refers to number of entity sets that participate in a relationship set.

Å Relationship sets that involve two entity sets are **binary** (or degree two). Generally, most relationship sets in a database system are binary.

Å Relationship sets may involve more than two entity sets.

Å Relationships between more than two entity sets are rare. Most relationships are binary.

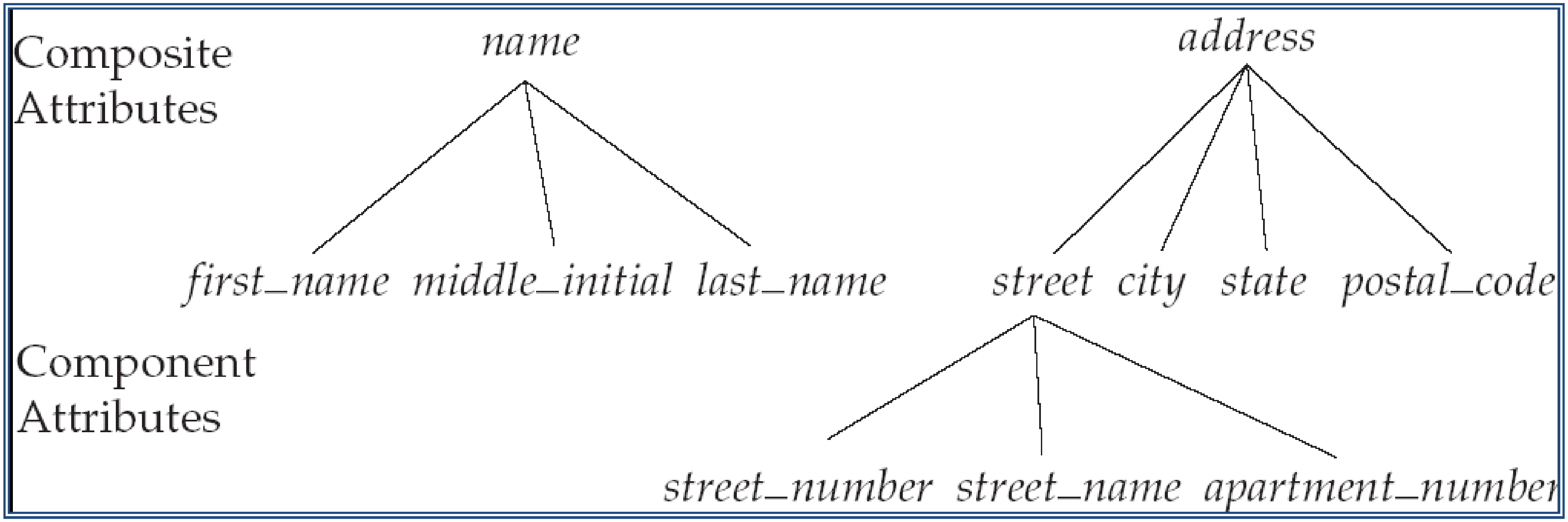
- Example: Suppose employees of a bank may have jobs (responsibilities) at multiple branches, with different jobs at different branches. Then there is a ternary relationship set between entity sets *employee, job, and branch*

Attributes

Attributes describe the properties of the entity of which they are associated. We can classify attributes as following:

Attribute types:

- *Simple* and *composite* attributes.
- *Single-valued* and *multi-valued* attributes
  - Example: multivalued attribute: *phone_numbers*
- *Derived* attributes
  - Can be computed from other attributes
  - Example: age, given date_of_birth



Composite Attributes

name

first_name *middle_initial* *last_name*

address

street *city* *state* *postal_code*

Component Attributes

street_number *street_name* *apartment_number*

Mapping Cardinality Constraints

Å Express the number of entities to which another entity can be associated via a relationship set.

Å Most useful in describing binary relationship sets.

Å For a binary relationship set the mapping cardinality must be one of the following types:

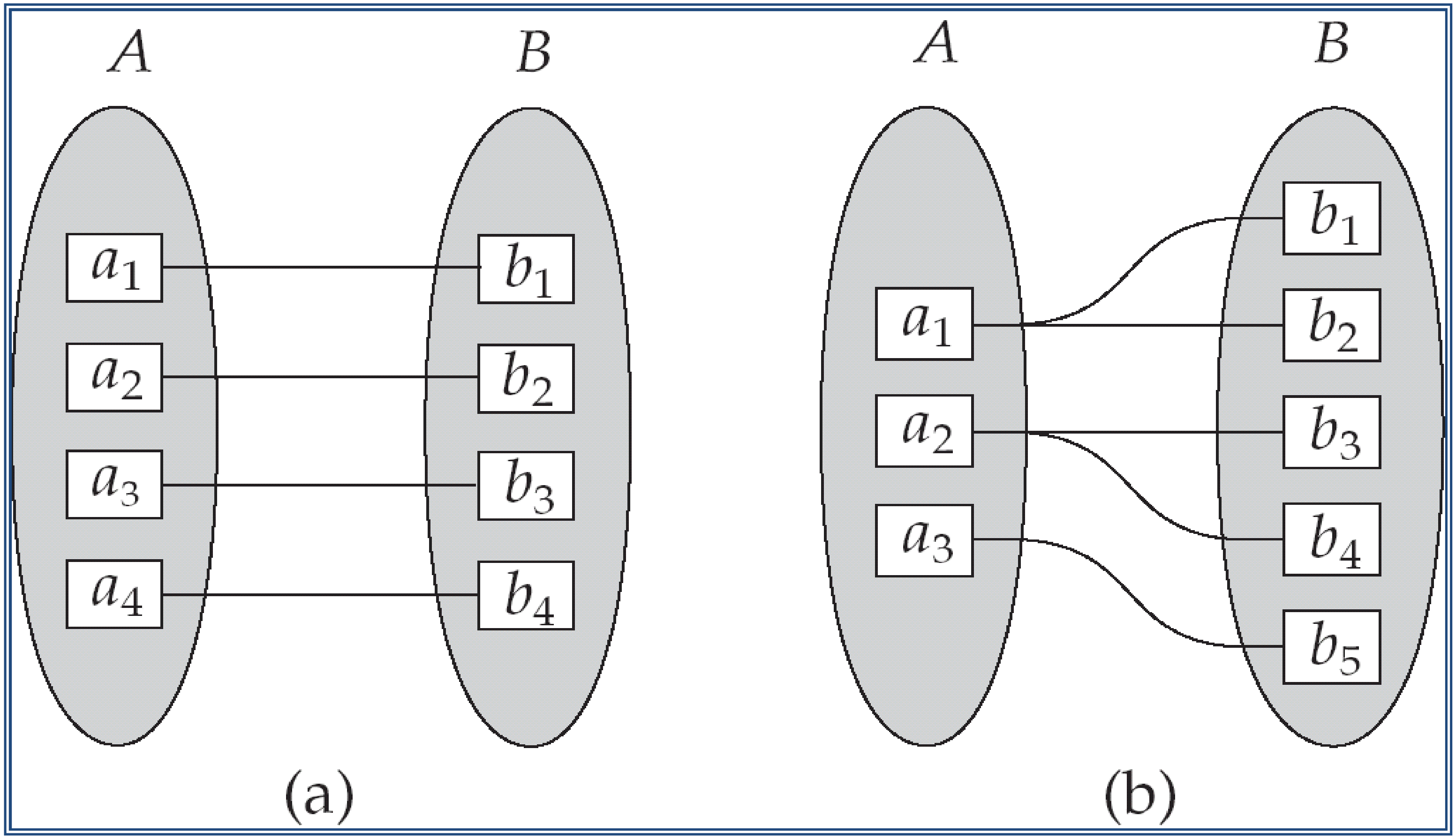
ï One to one

ï One to many

ï Many to one

ï Many to many

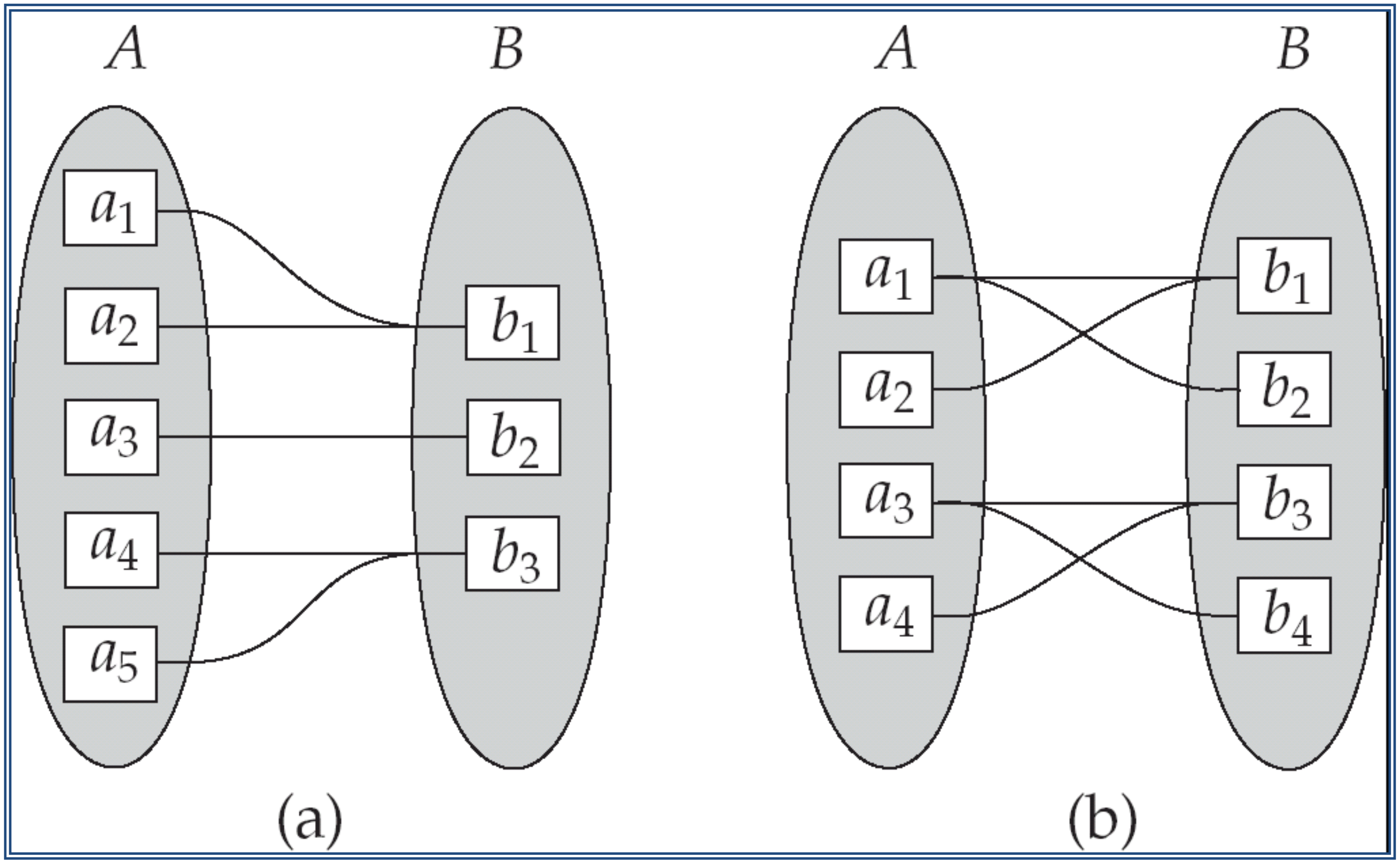
Mapping Cardinalities



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

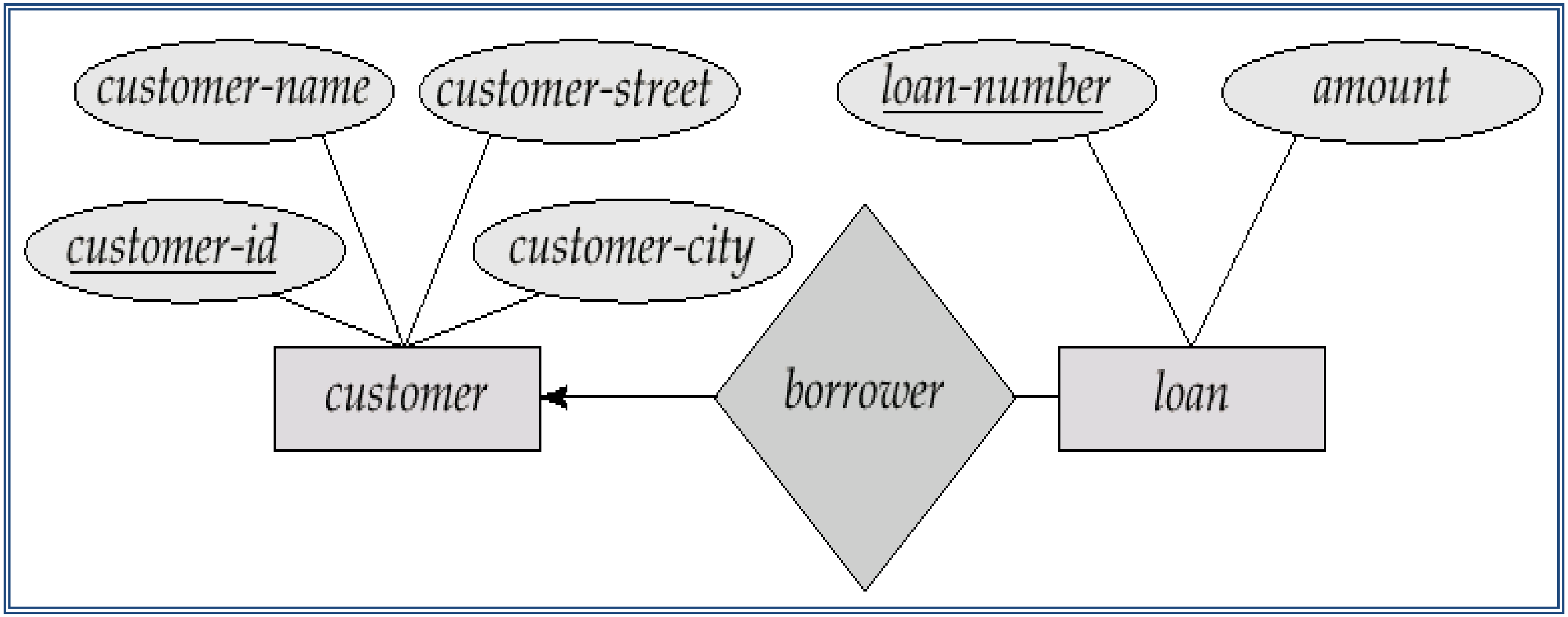
Cardinality Constraints

- 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.

- 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*

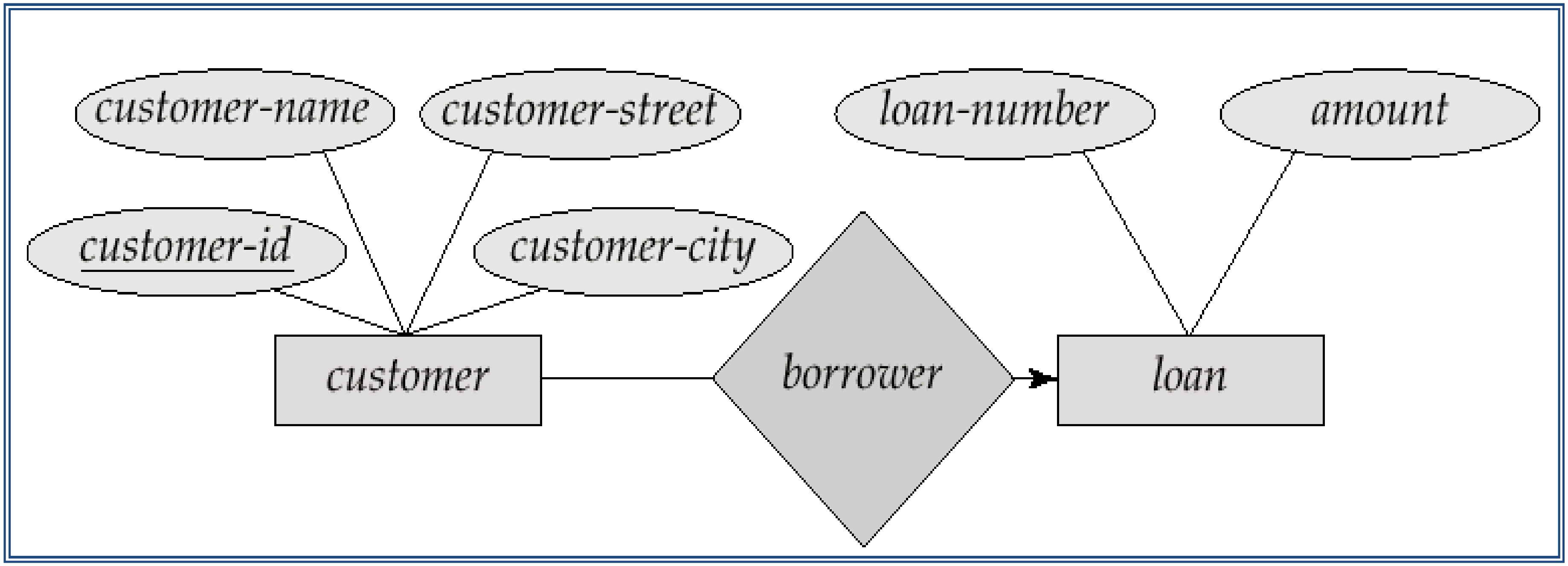
One-To-Many Relationship

- Å 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

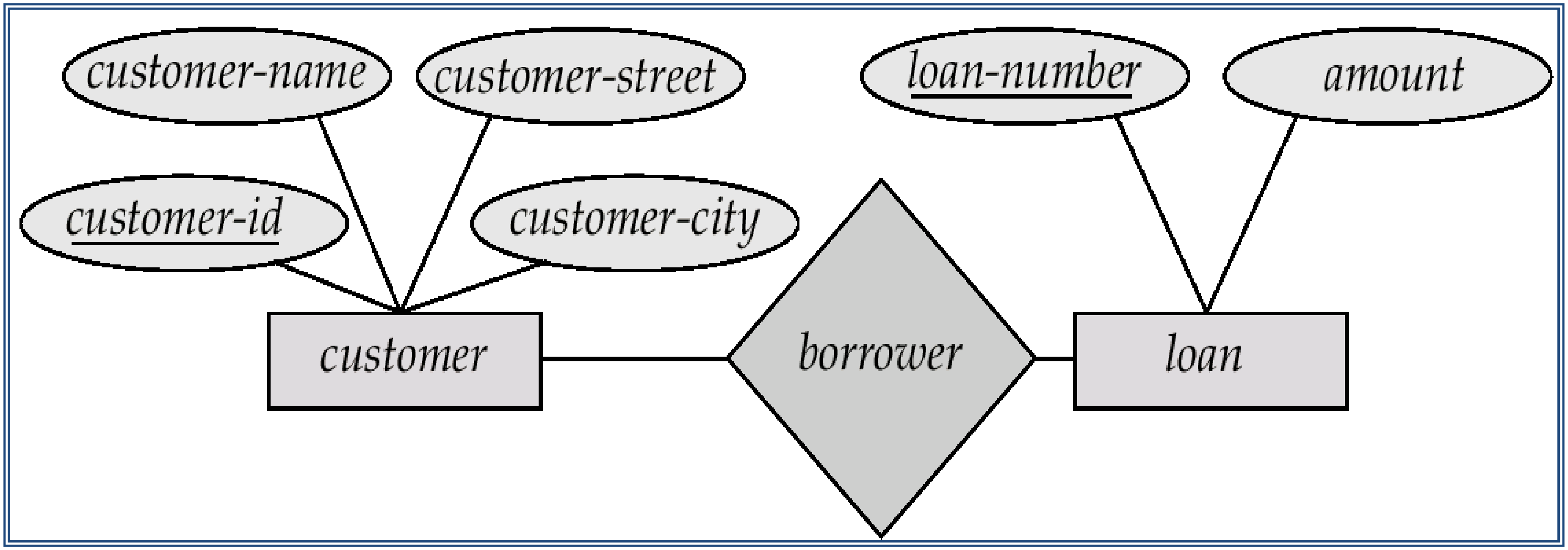
Å In a many-to-one relationship a loan is associated with several (including 0) customers via *borrower*, a customer is associated with at most one loan via *borrower*



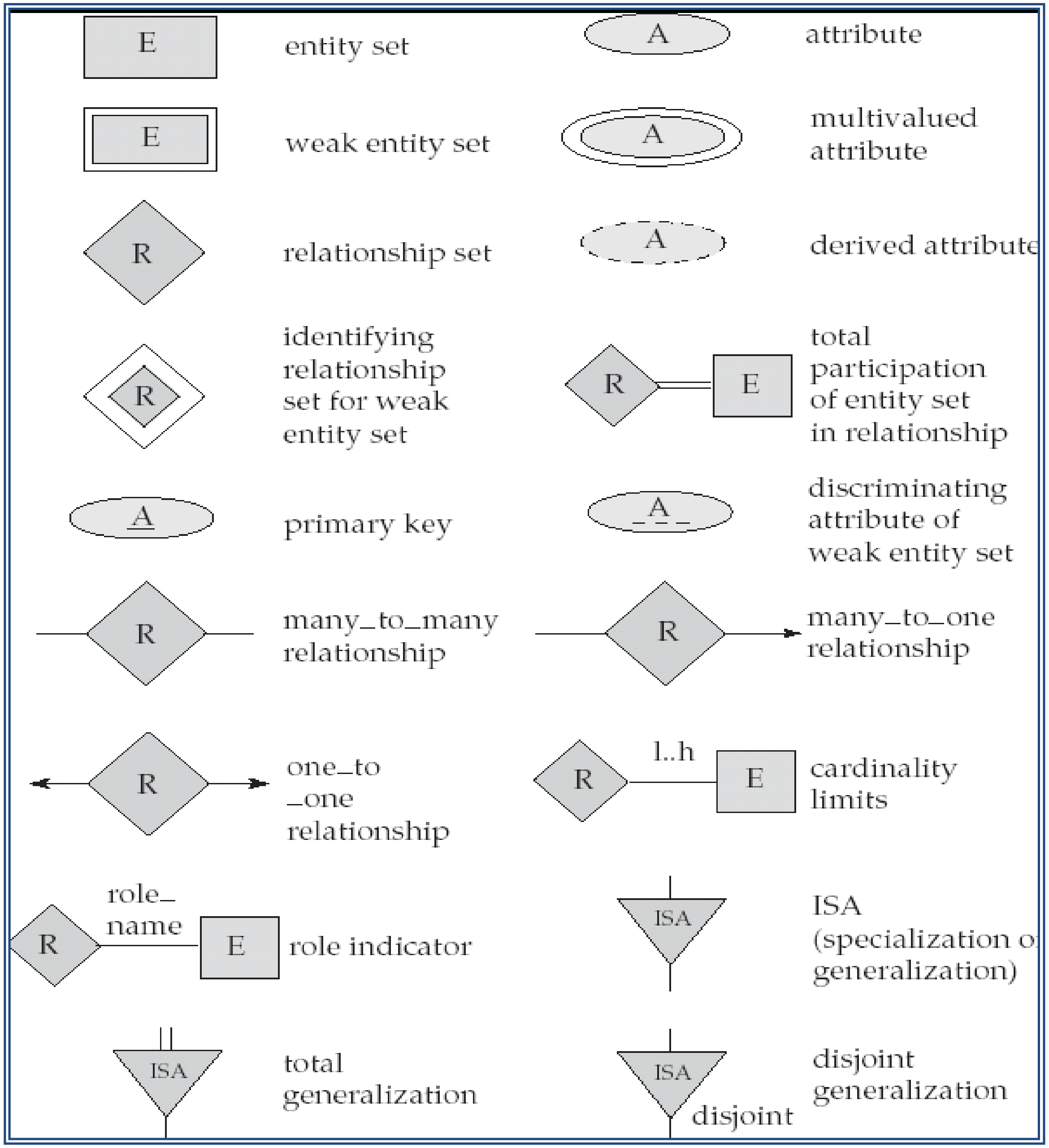
Many-To-Many Relationship

Å A customer is associated with several (possibly 0) loans via borrower

Å A loan is associated with several (possibly 0) customers via borrower



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

many_to_many relationship

many_to_one relationship

one_to_one relationship

cardinality limits

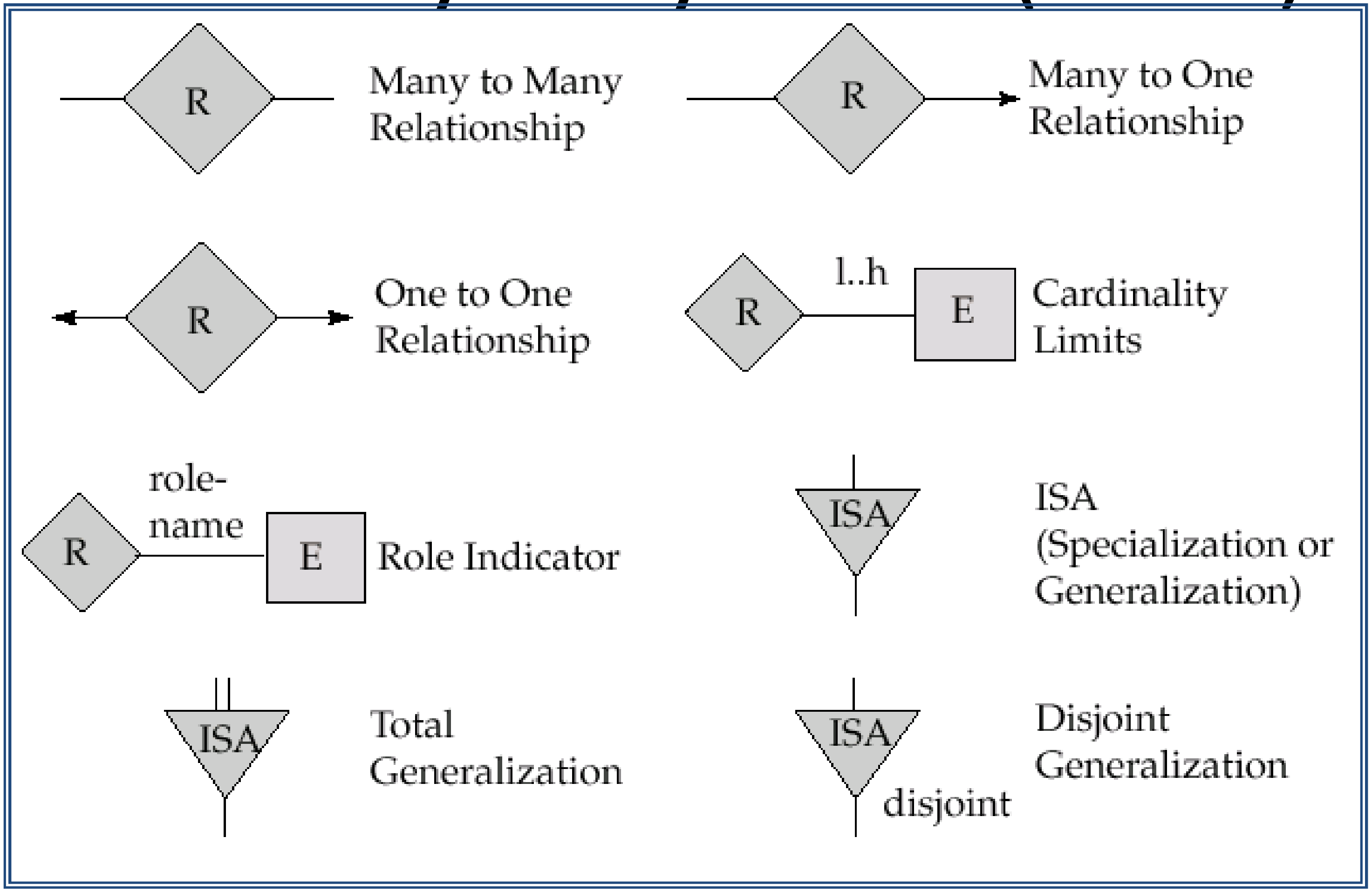
role indicator

ISA (specialization o generalization)

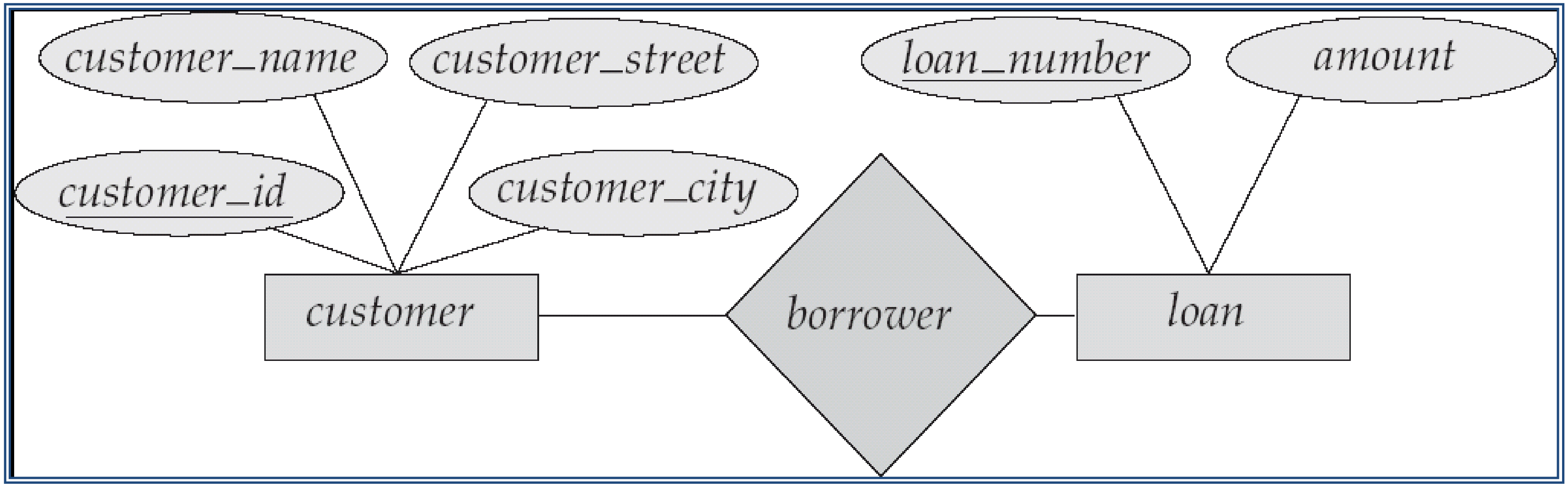
total generalization

disjoint generalization

Summary of Symbols (Cont.)

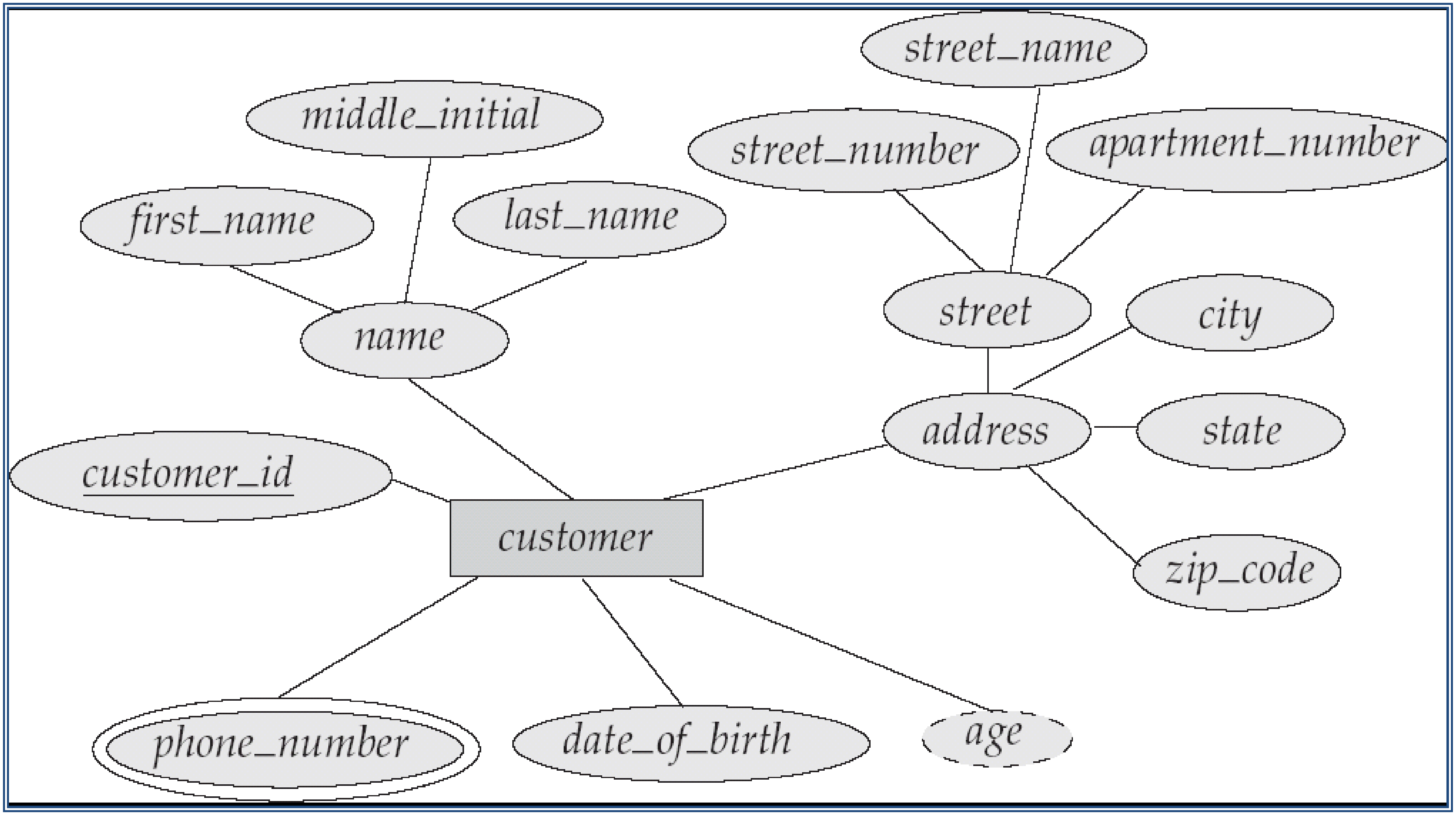


E-R Diagrams

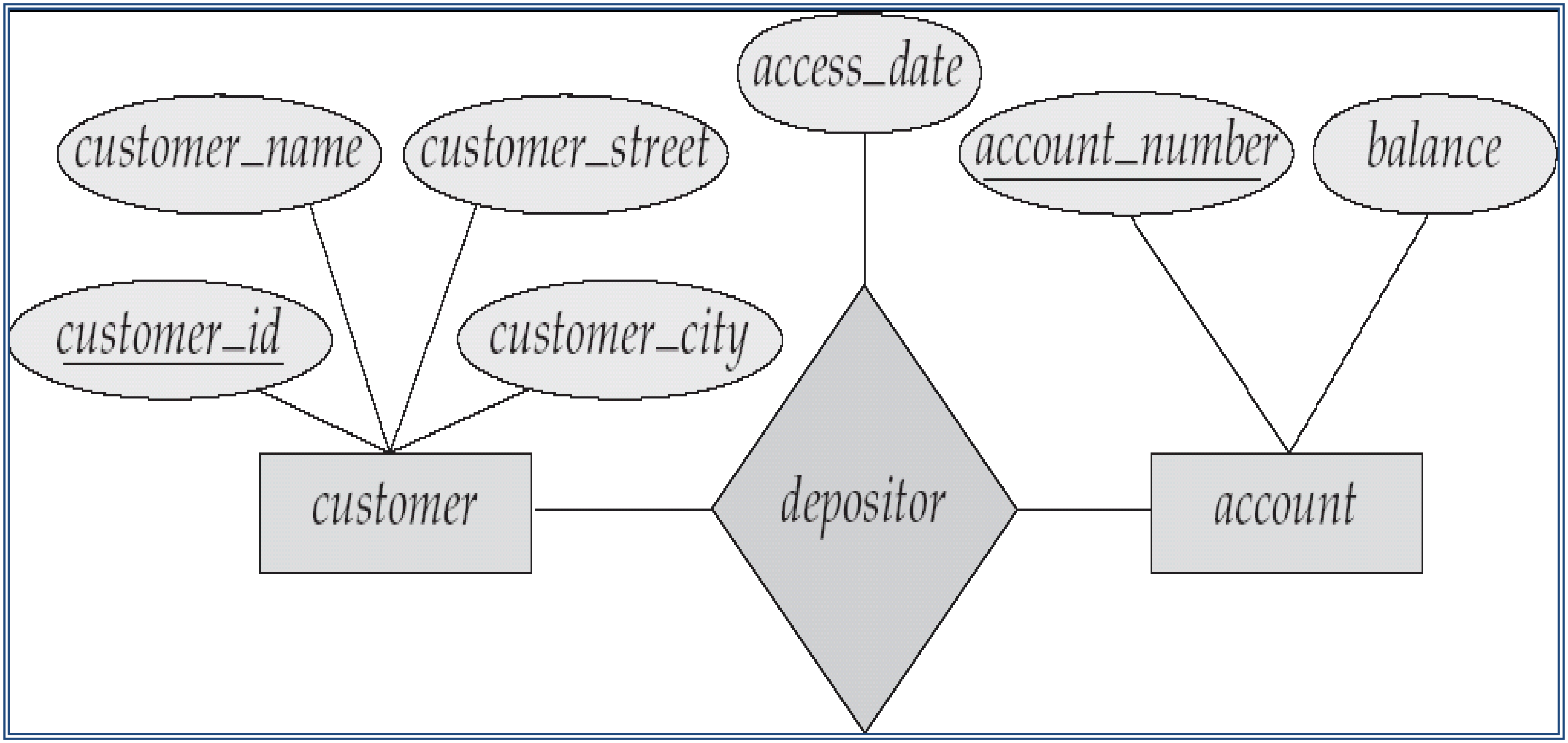


- Rectangles represent entity sets.
- Diamonds represent relationship sets.
- Lines link attributes to entity sets and entity sets to relationship sets.
- Ellipses represent attributes
  - Double ellipses represent multi valued attributes.
  - Dashed ellipses denote derived attributes.
- Underline indicates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

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



Relationship Sets with Attributes



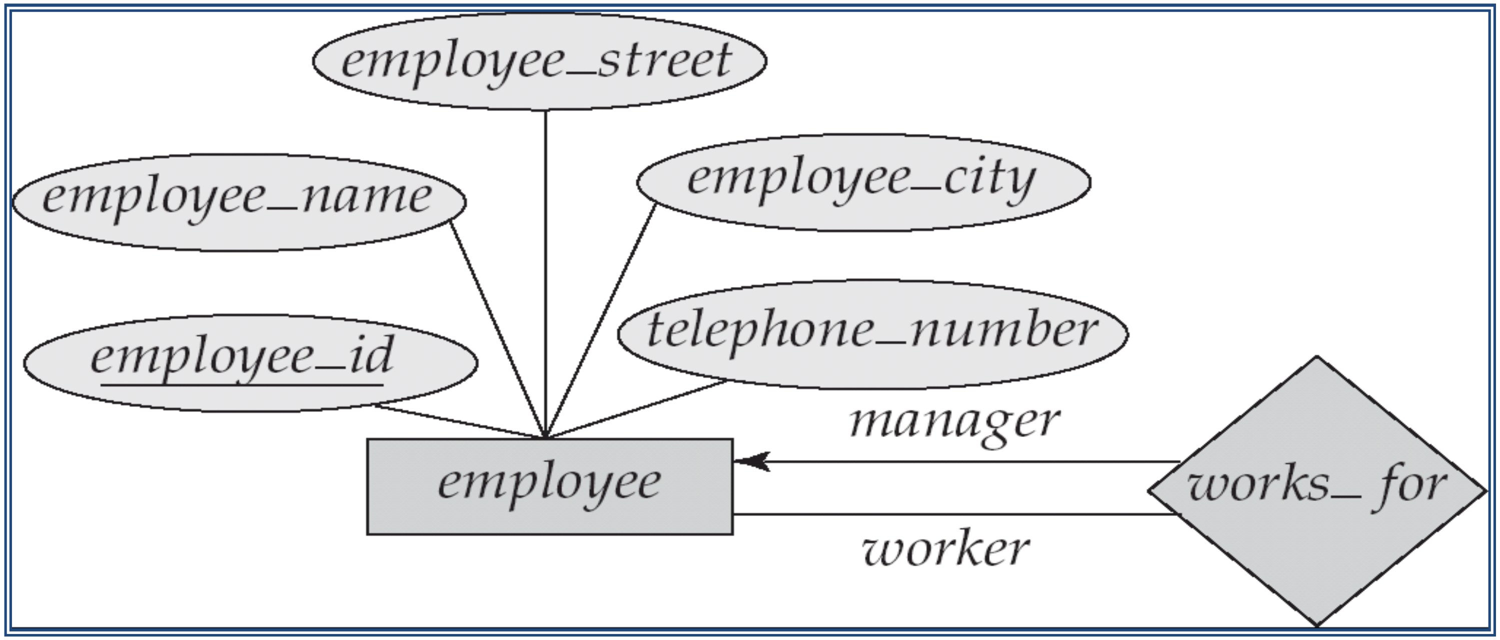
Roles

Å Entity sets of a relationship need not be distinct

Å The labels “manager” and “worker” are called **roles**; they specify how employee entities interact via the works_for relationship set.

Å Roles are indicated in E-R diagrams by labeling the lines that connect diamonds to rectangles.

Å Role labels are optional, and are used to clarify semantics of the relationship



Weak Entity Sets

Å An entity set that does not have a primary key is referred to as a **weak entity set**.

Å The existence of a weak entity set depends on the existence of a **identifying entity set**

ï it must relate to the identifying entity set via a total, one-to-many relationship set from the identifying to the weak entity set

ï Identifying relationship depicted using a double diamond

Å 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.

Å 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.

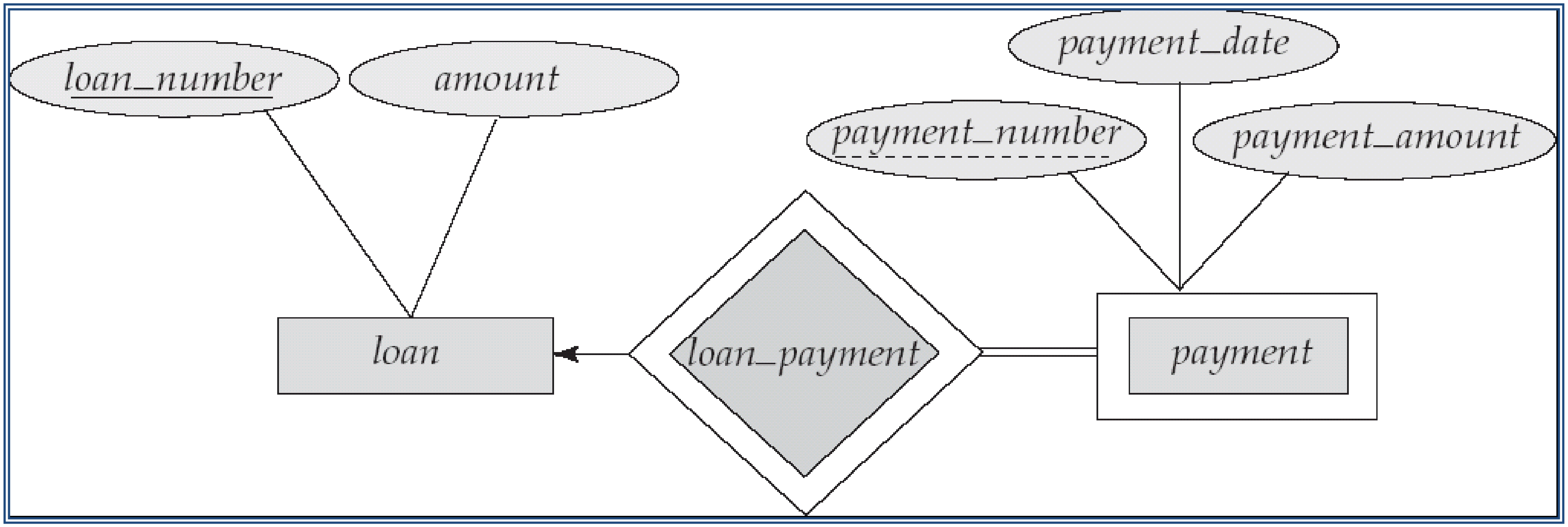
Weak Entity Sets (Cont.)

Å We depict a weak entity set by double rectangles.

Å We underline the discriminator of a weak entity set with a dashed line.

Å payment_number – discriminator of the *payment* entity set

Å Primary key for *payment* – (*loan_number, payment_number*)



Extended E-R Features: Specialization

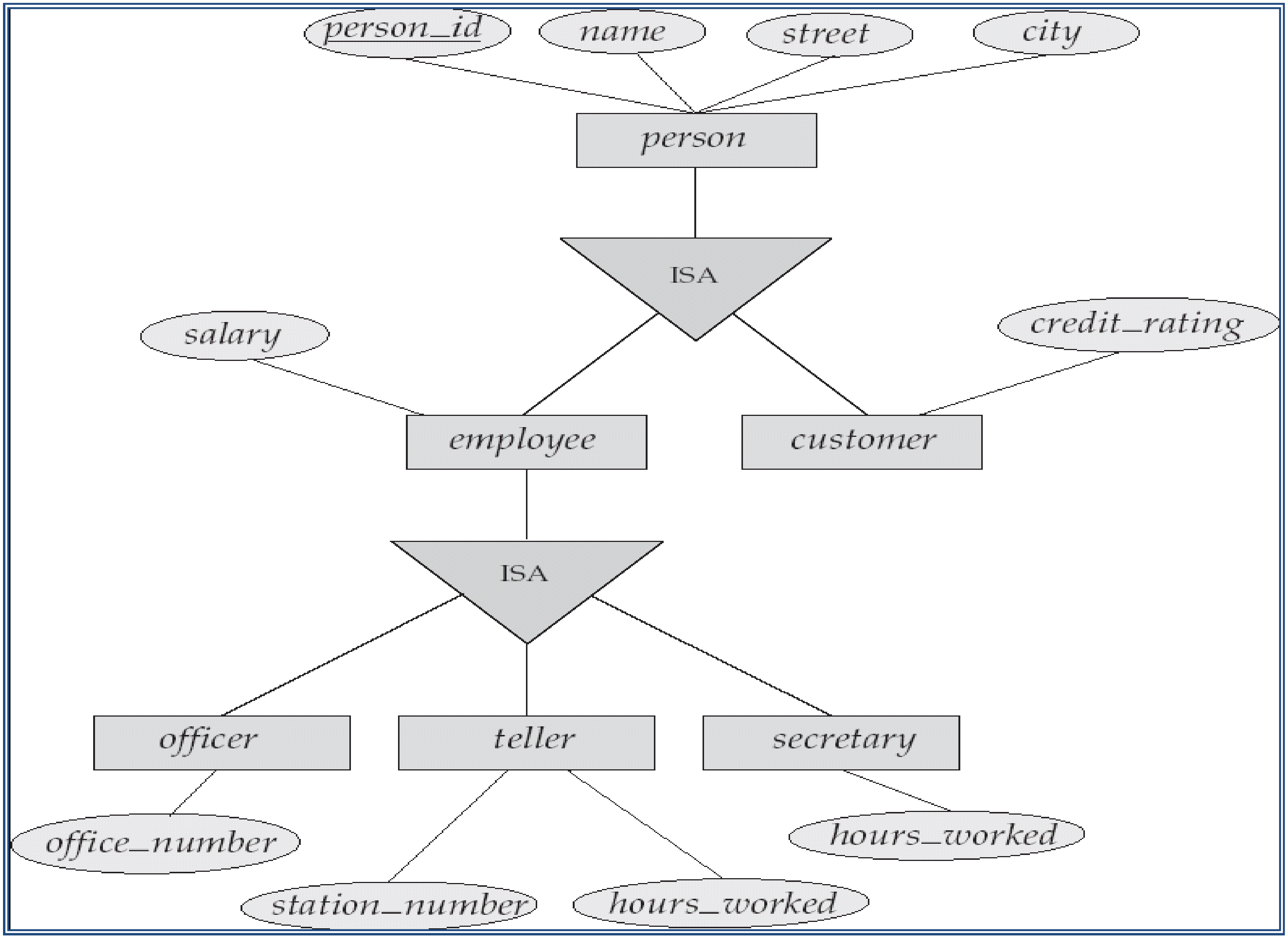
Å Top-down design process; we designate subgroupings within an entity set that are distinctive from other entities in the set.

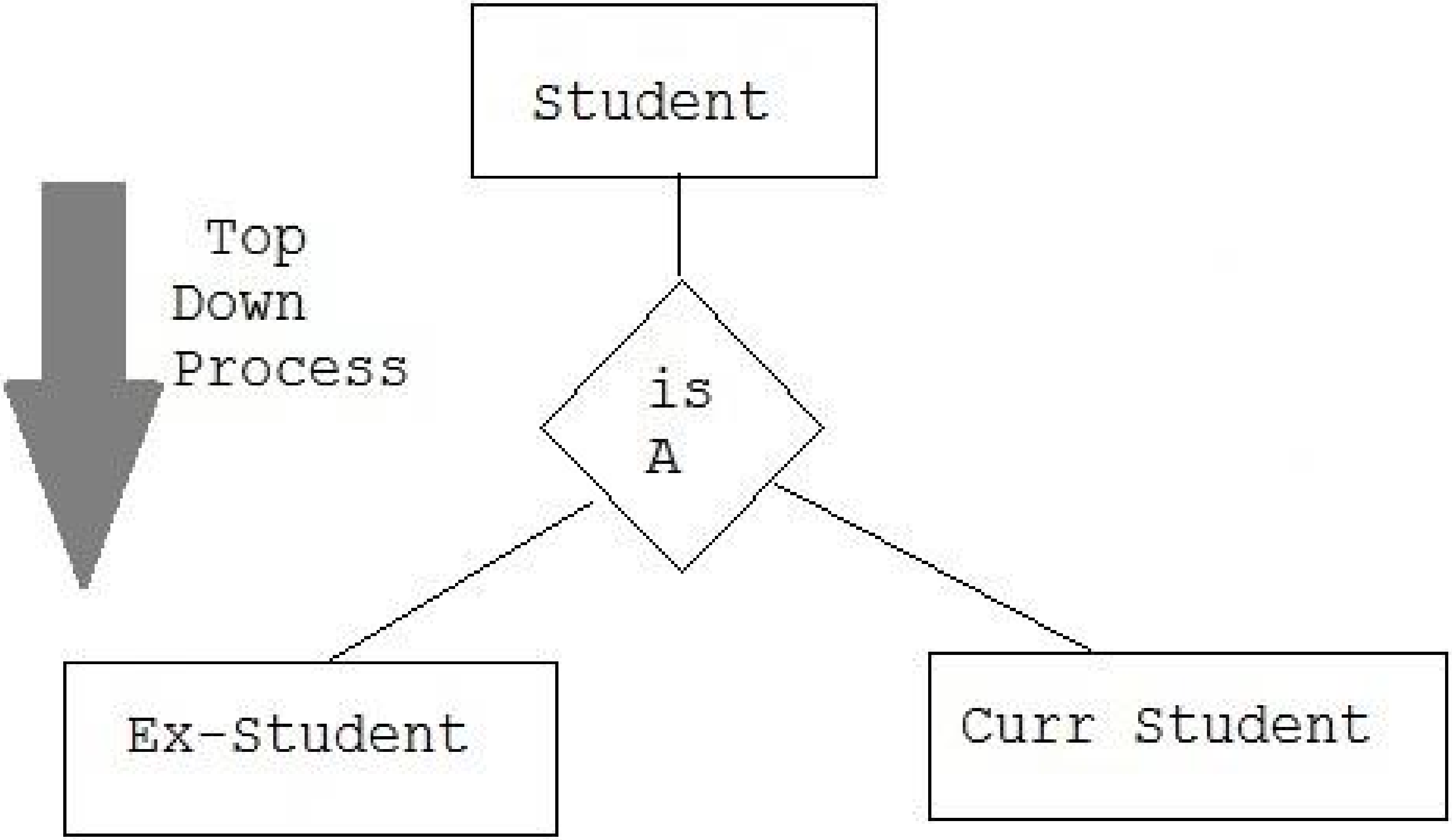
Å These subgroupings become lower-level entity sets that have attributes or participate in relationships that do not apply to the higher-level entity set.

Å Depicted by a *triangle* component labeled ISA (E.g. *customer* "is a" *person*).

Å **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



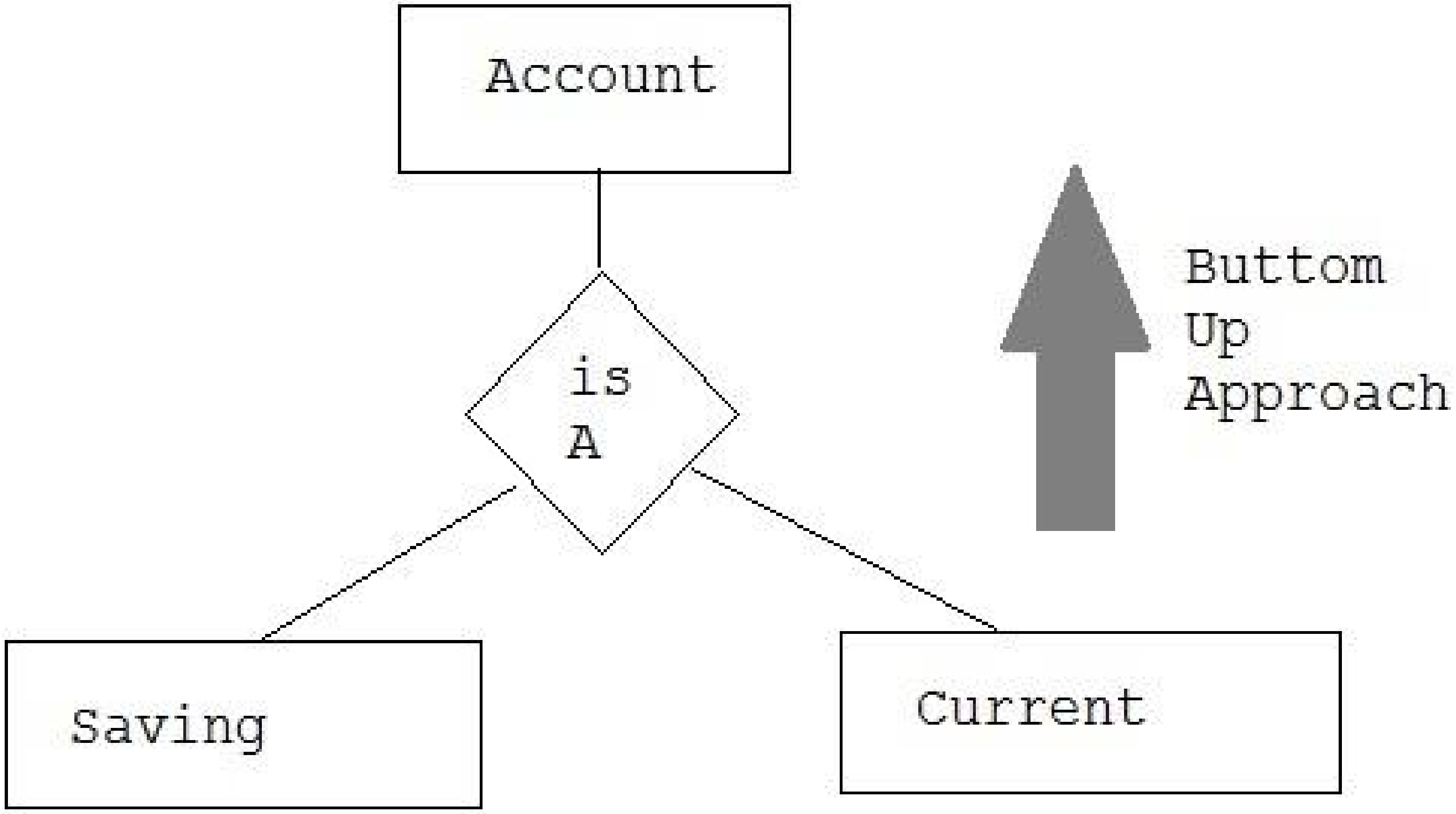

Student
Top
Down
Process
is
A
Ex-Student
Curr Student

Extended ER Features: Generalization

Å **A bottom-up design process** – combine a number of entity sets that share the same features into a higher-level entity set.

Å Specialization and generalization are simple inversions of each other; they are represented in an E-R diagram in the same way.

Å The terms specialization and generalization are used interchangeably.


Account
is
A
Saving
Current
Buttom
Up
Approach

Specialization and Generalization (Cont.)

Å Can have multiple specializations of an entity set based on different features.

Å E.g. *permanent_employee* vs. *temporary_employee*, in addition to *officer* vs. *secretary* vs. *teller*

Å Each particular employee would be

  ï a member of one of *permanent_employee* or *temporary_employee*,

  ï and also a member of one of *officer*, *secretary*, or *teller*

Å The ISA relationship also referred to as **superclass - subclass** relationship

Design Constraints on a Specialization/Generalization

- Constraint on which entities can be members of a given lower-level entity set.
  - condition-defined
    - Example: all customers over 65 years are members of *senior-citizen* entity set; *senior-citizen* ISA *person*.
  - user-defined
- Constraint on whether or not entities may belong to more than one lower-level entity set within a single generalization.
  - **Disjoint**
    - an entity can belong to only one lower-level entity set
    - Noted in E-R diagram by writing *disjoint* next to the ISA triangle
  - **Overlapping**
    - an entity can belong to more than one lower-level entity set

Design Constraints on a Specialization/Generalization (Cont.)

- **Completeness constraint** -- specifies whether or not an entity in the higher-level entity set must belong to at least one of the lower-level entity sets within a generalization.
  - **total** : an entity must belong to one of the lower-level entity sets
  - **partial**: an entity need not belong to one of the lower-level entity sets

Aggregation

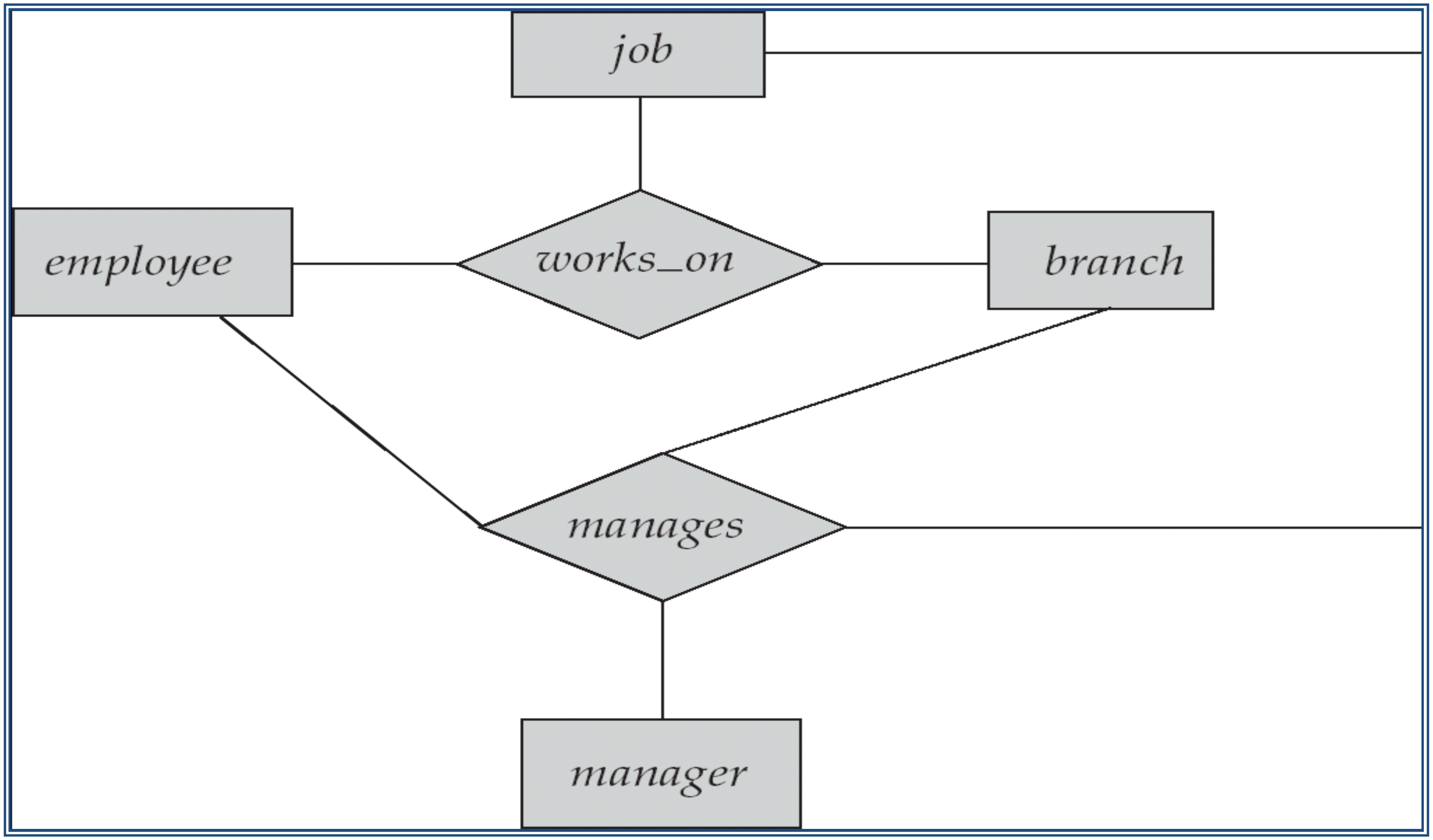
One limitation of the E-R model is that it cannot express relationships among relationships.

The best way to model a situation like this is by the use of Aggregation

Aggregation

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

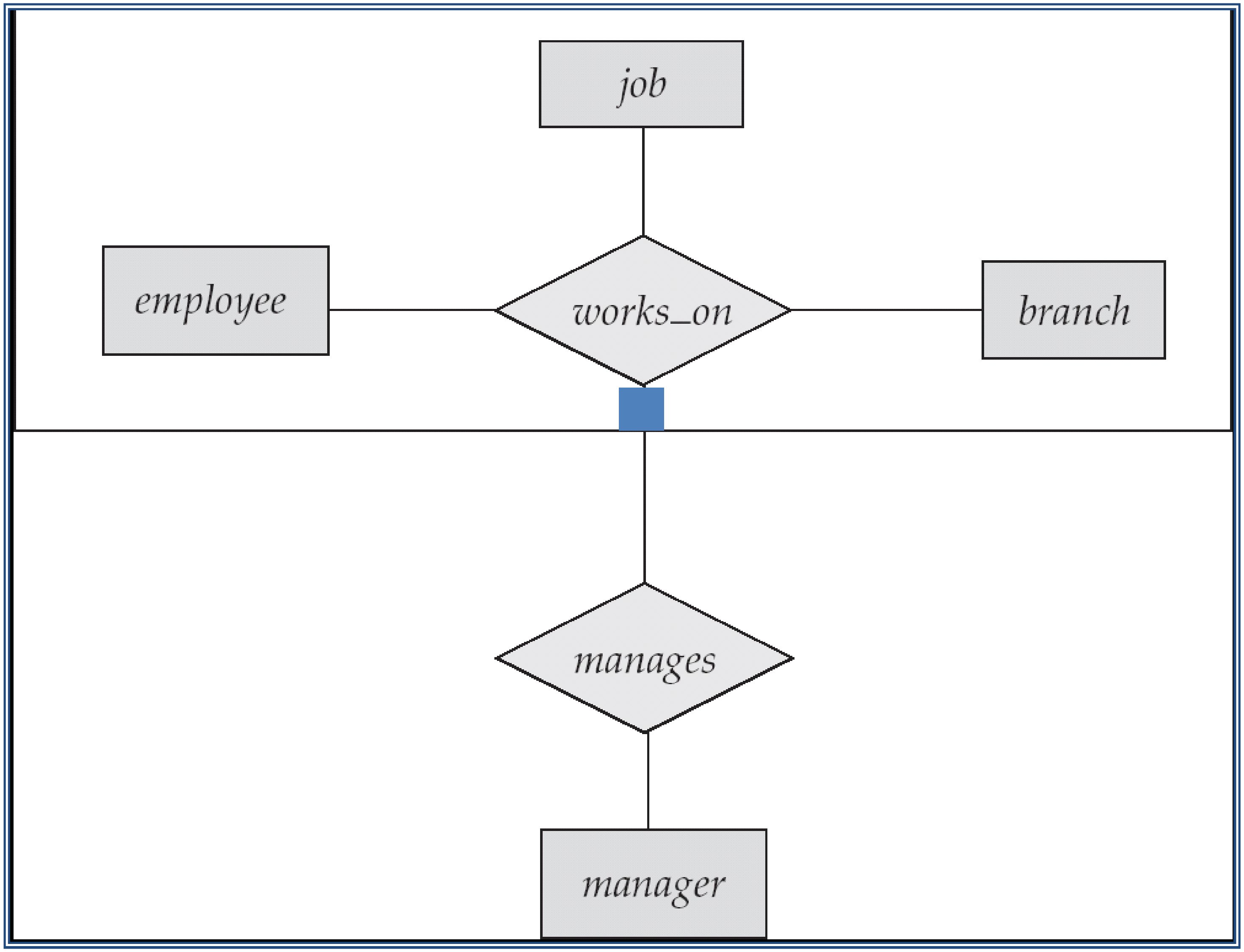
Suppose we want to record managers for tasks performed by an employee at a branch



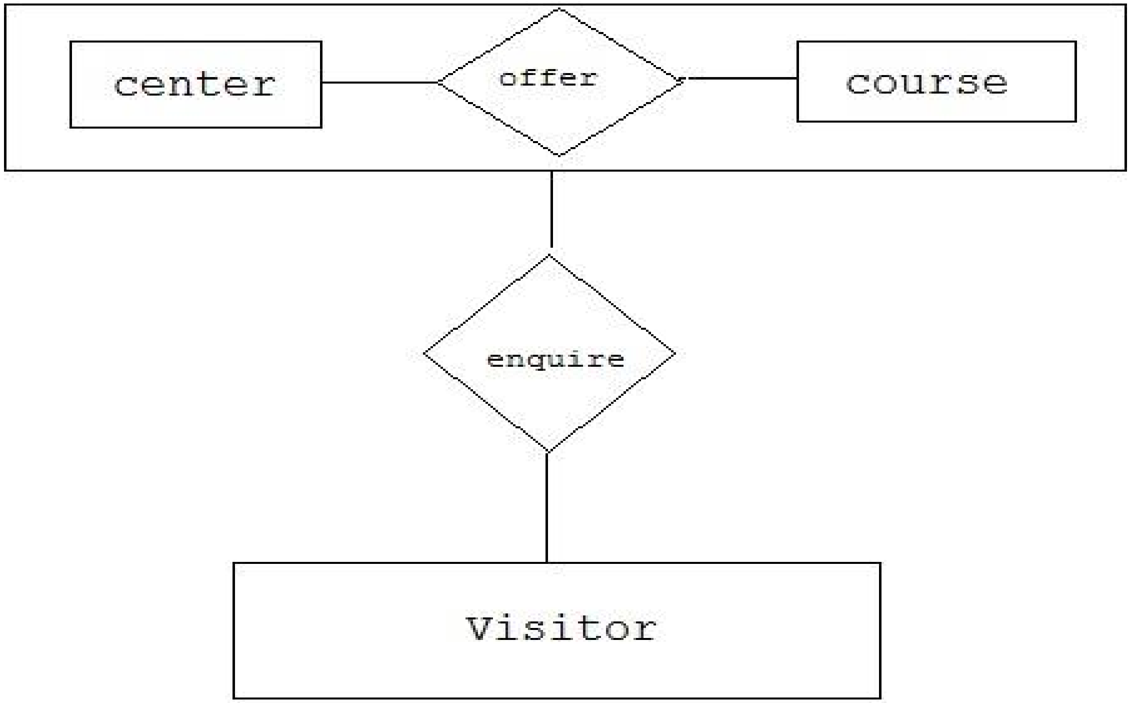
Aggregation (Cont.)

- 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
  - Allows relationships between relationships
  - Abstraction of relationship into new entity
- Without introducing redundancy, the following diagram represents:
  - An employee works on a particular job at a particular branch
  - An employee, branch, job combination may have an associated manager

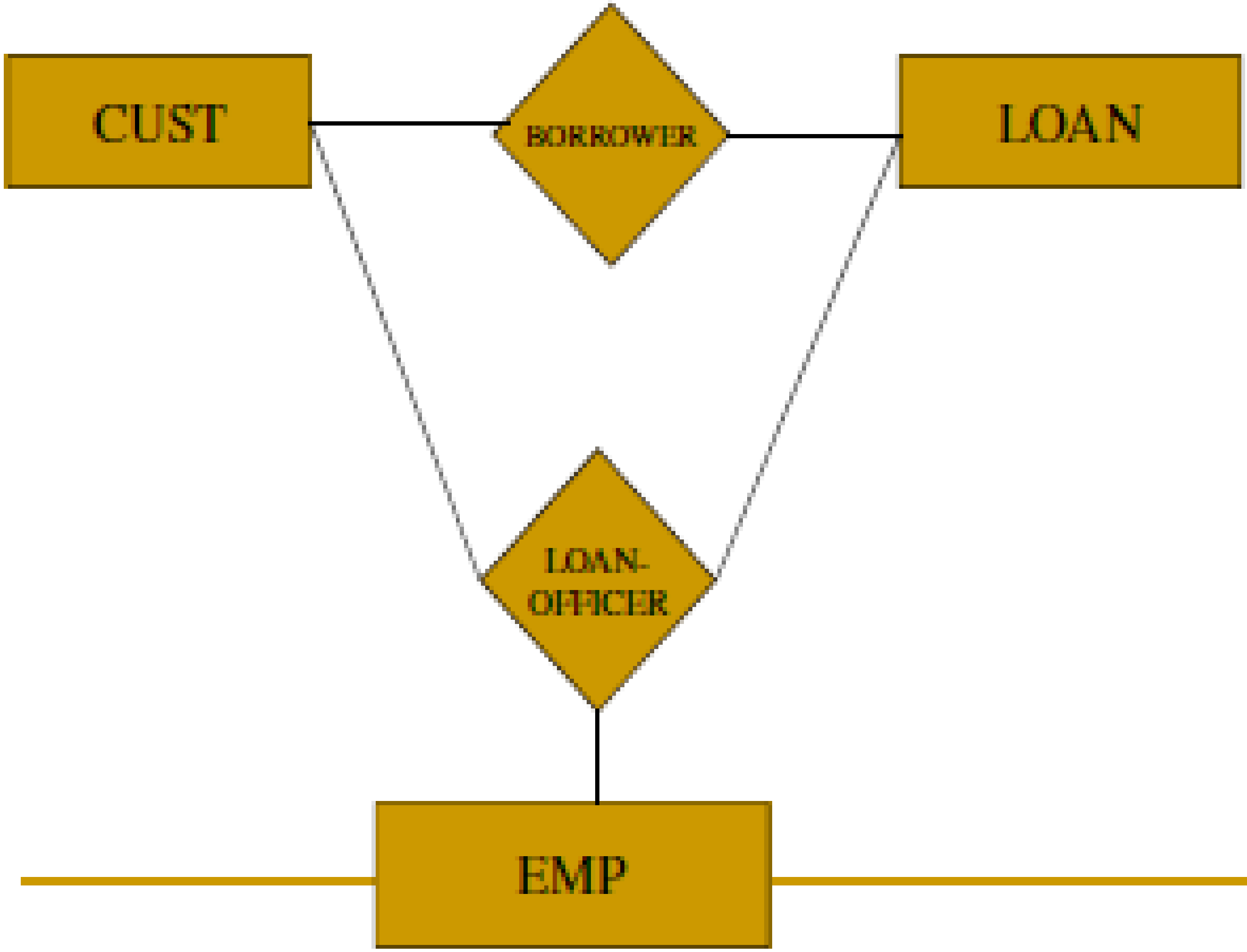
E-R Diagram With Aggregation



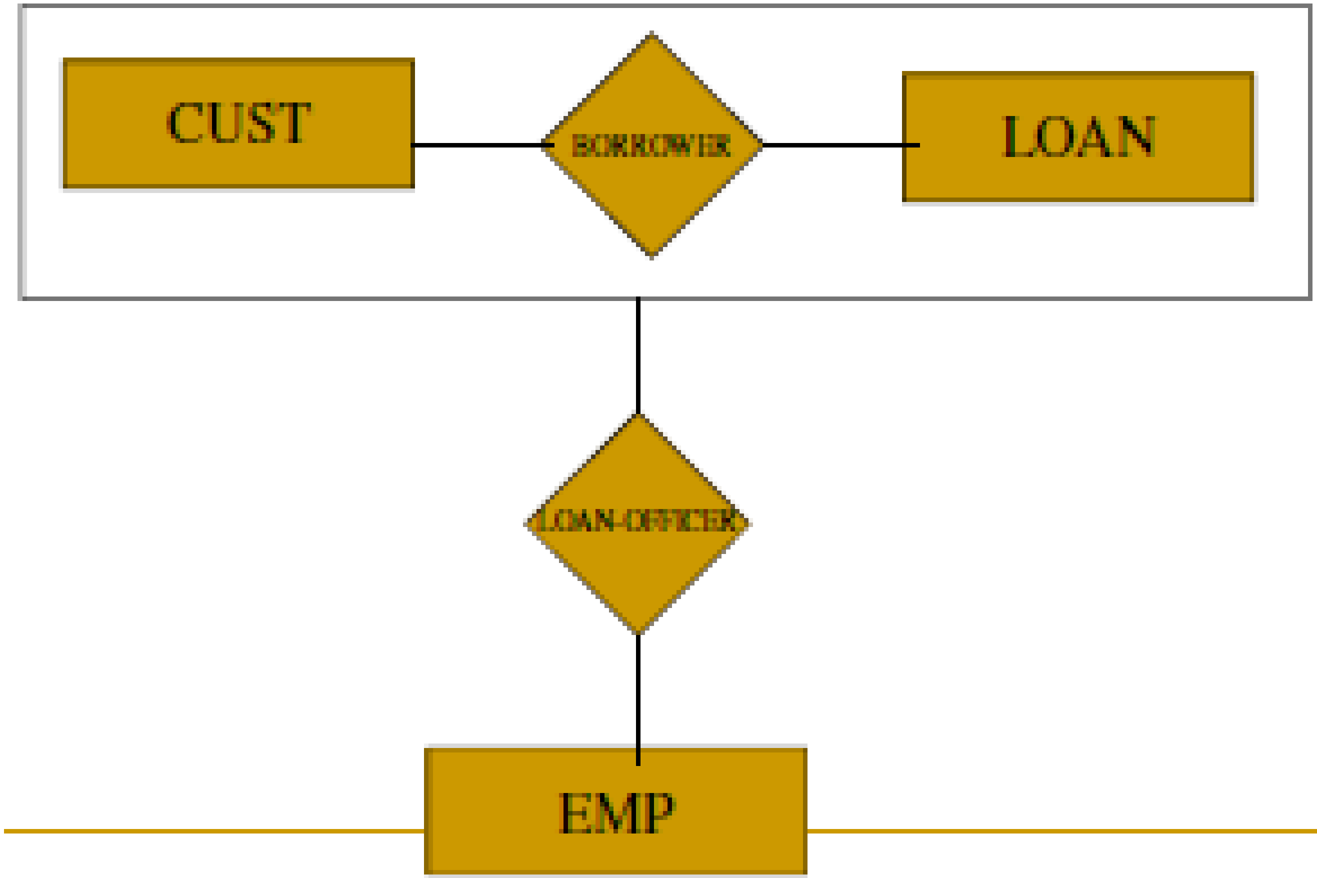
Aggregration is a process when relation between two entity is treated as a single entity. Here the relation between Center and Course, is acting as an Entity in relation with Visitor.



Suppose a customer loan pair may have a bank emp Who is a loan officer for that particular pair.

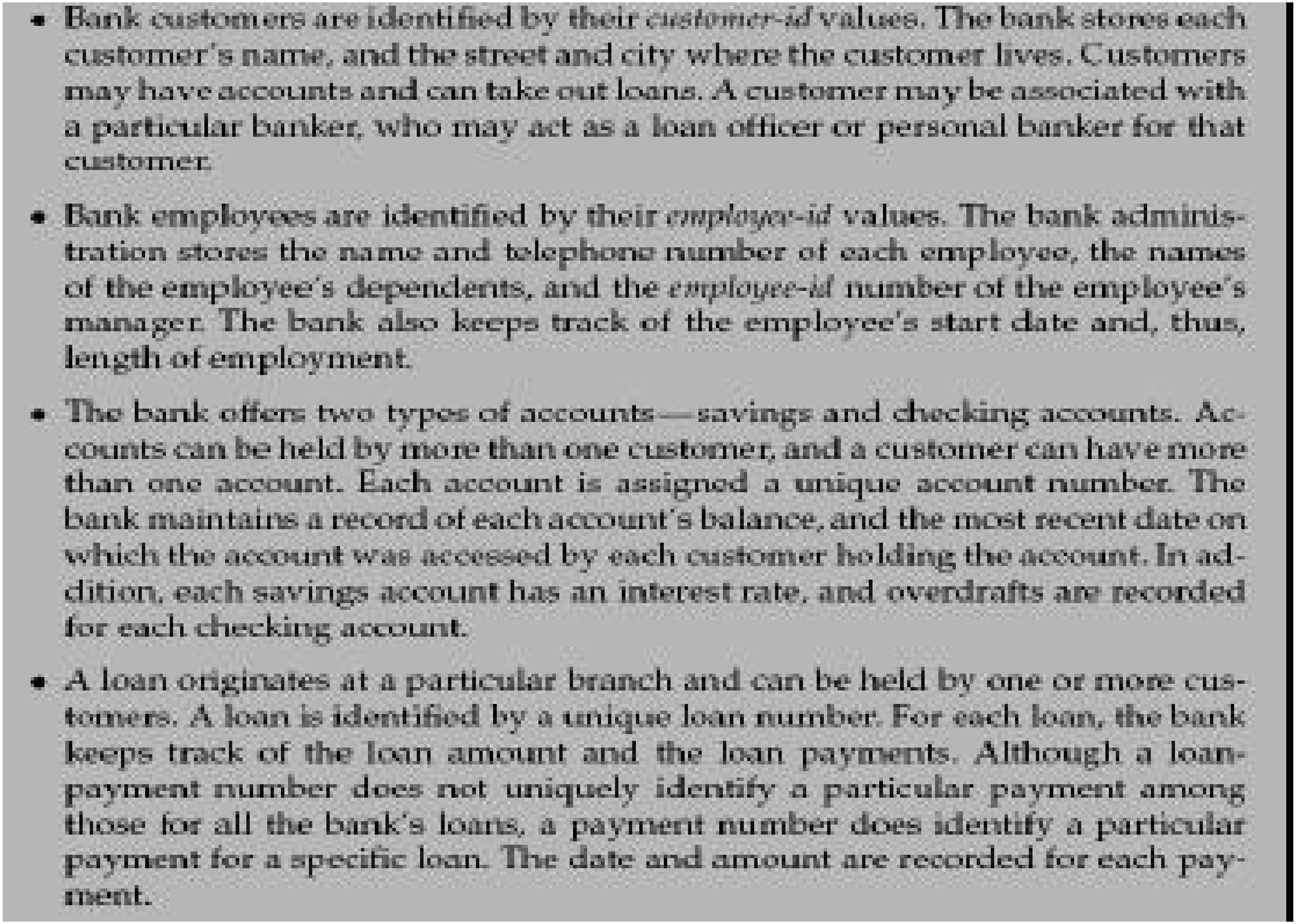


The best way to model the situation is to use aggregation. Aggregation is asn abstraction through which relationship are treated as higher level entities



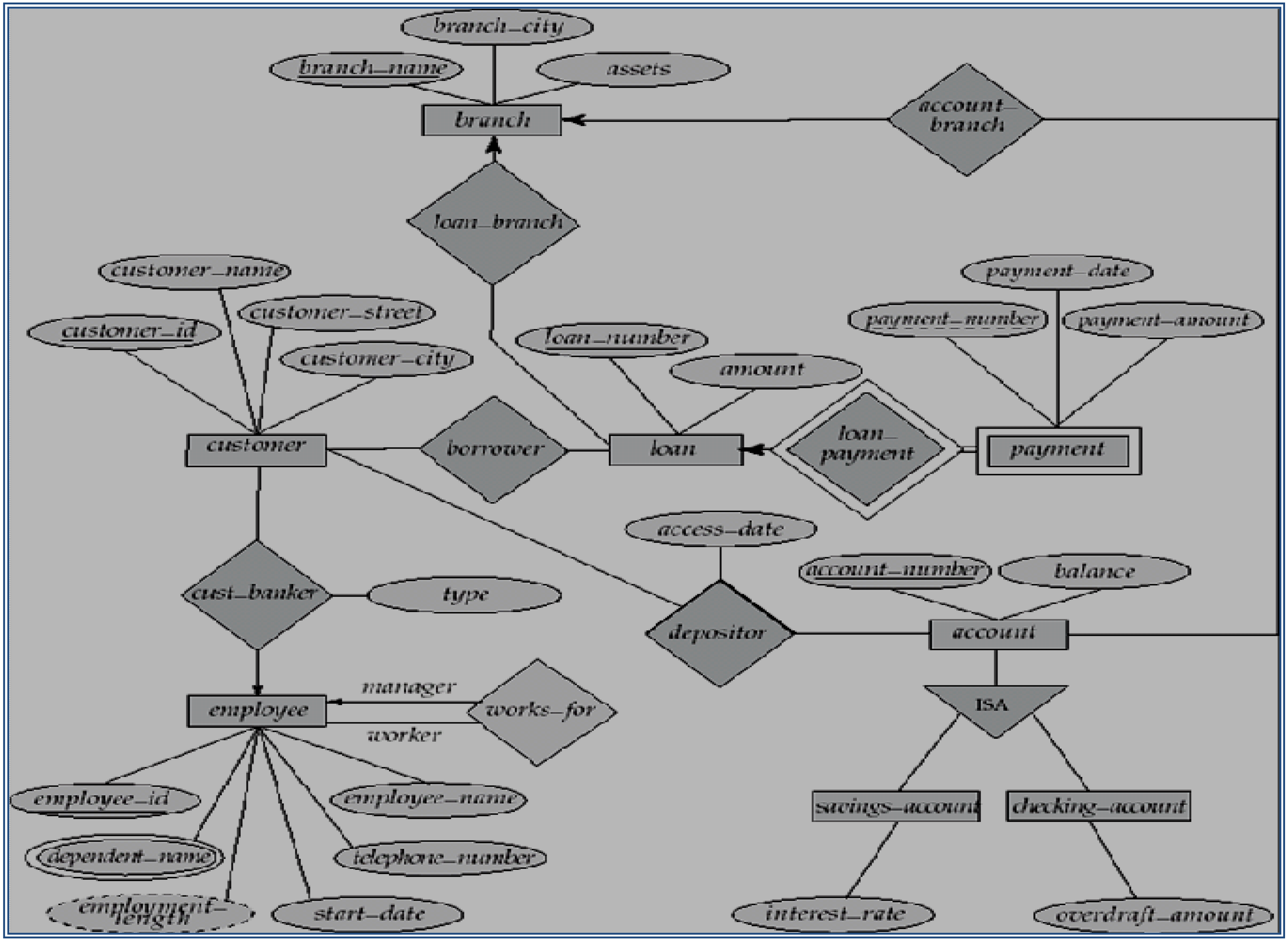
Case Study: Database Design for Banking Enterprise

- Here are the major characteristics of the banking enterprise.
- • The bank is organized into branches. Each branch is located in a particular city and is identified by a unique name. The bank monitors the assets of each branch.



- Bank customers are identified by their *customer-id* values. The bank stores each customer's name, and the street and city where the customer lives. Customers may have accounts and can take out loans. A customer may be associated with a particular banker, who may act as a loan officer or personal banker for that customer.
- Bank employees are identified by their *employee-id* values. The bank administration stores the name and telephone number of each employee, the names of the employee's dependents, and the *employee-id* number of the employee's manager. The bank also keeps track of the employee's start date and, thus, length of employment.
- The bank offers two types of accounts—savings and checking accounts. Accounts can be held by more than one customer, and a customer can have more than one account. Each account is assigned a unique account number. The bank maintains a record of each account's balance, and the most recent date on which the account was accessed by each customer holding the account. In addition, each savings account has an interest rate, and overdrafts are recorded for each checking account.
- A loan originates at a particular branch and can be held by one or more customers. A loan is identified by a unique loan number. For each loan, the bank keeps track of the loan amount and the loan payments. Although a loan-payment number does not uniquely identify a particular payment among those for all the bank's loans, a payment number does identify a particular payment for a specific loan. The date and amount are recorded for each payment.

E-R Diagram for a Banking Enterprise


branch_city
branch_name
assets
branch
account–branch
loan_branch
customer_name
customer–street
customer_id
customer–city
payment–date
payment_number
payment–amount
loan_number
amount
customer
borrower
loan
loan–payment
payment
access–date
account_number
balance
cust_banker
type
depositor
account
manager
employee
works–for
worker
ISA
employee_id
employee–name
savings–account
checking–account
dependent_name
telephone–number
employment–length
start–date
interest_rate
overdraft_amount