

Entity Relationship Model

Chapter - 2

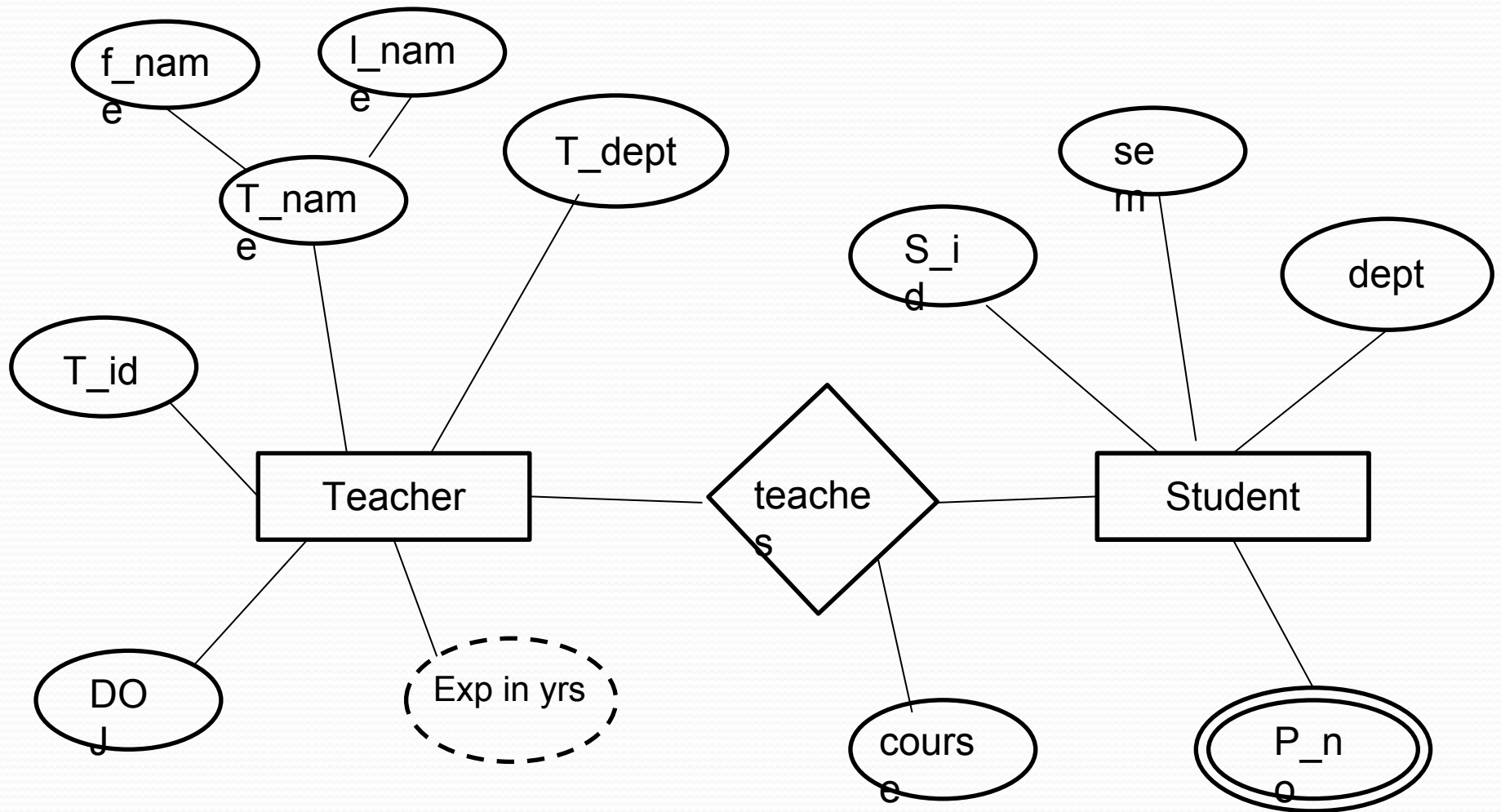
Overview

- Introduction and basics
- Entity and Entity Set
- Attributes and types
- Relationship
- Strong and weak entity set
- Traps
- Conversion

ER - Model

- Introduced by Dr. Peter Chen in 1976
- A non technical design method works on conceptual level based on the perception of real world
- Consists of collection of basic objects called entities and of relationships among these objects and attributes which defines their properties
- Free from ambiguities and provides a standard and logical way of visualizing data
- Basically it is a diagrammatic representation easy to understand even by non technical user

ER - Model



Entities

- 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*, *PAN no*, *Aadhar no*, *etc.*
- 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

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*
 - *Stored* and *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

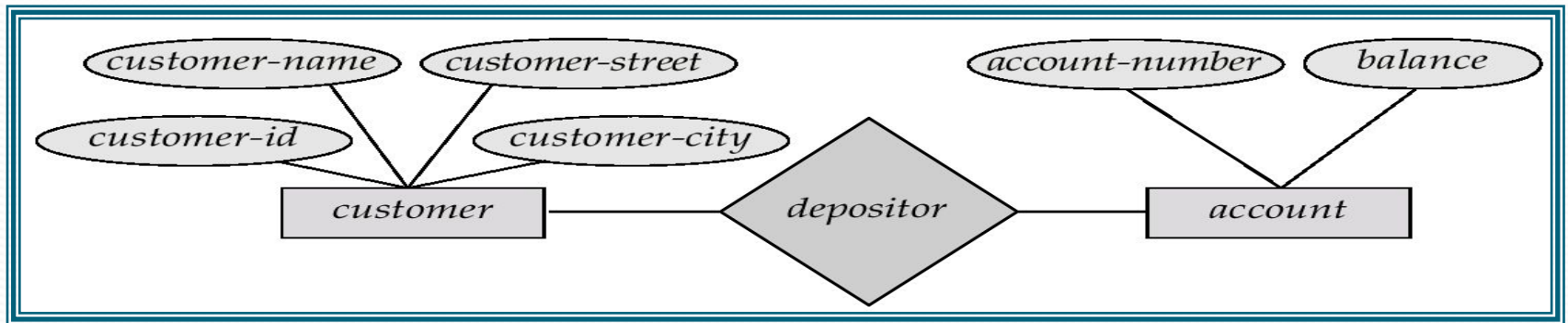
street-number street-name apartment-number

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

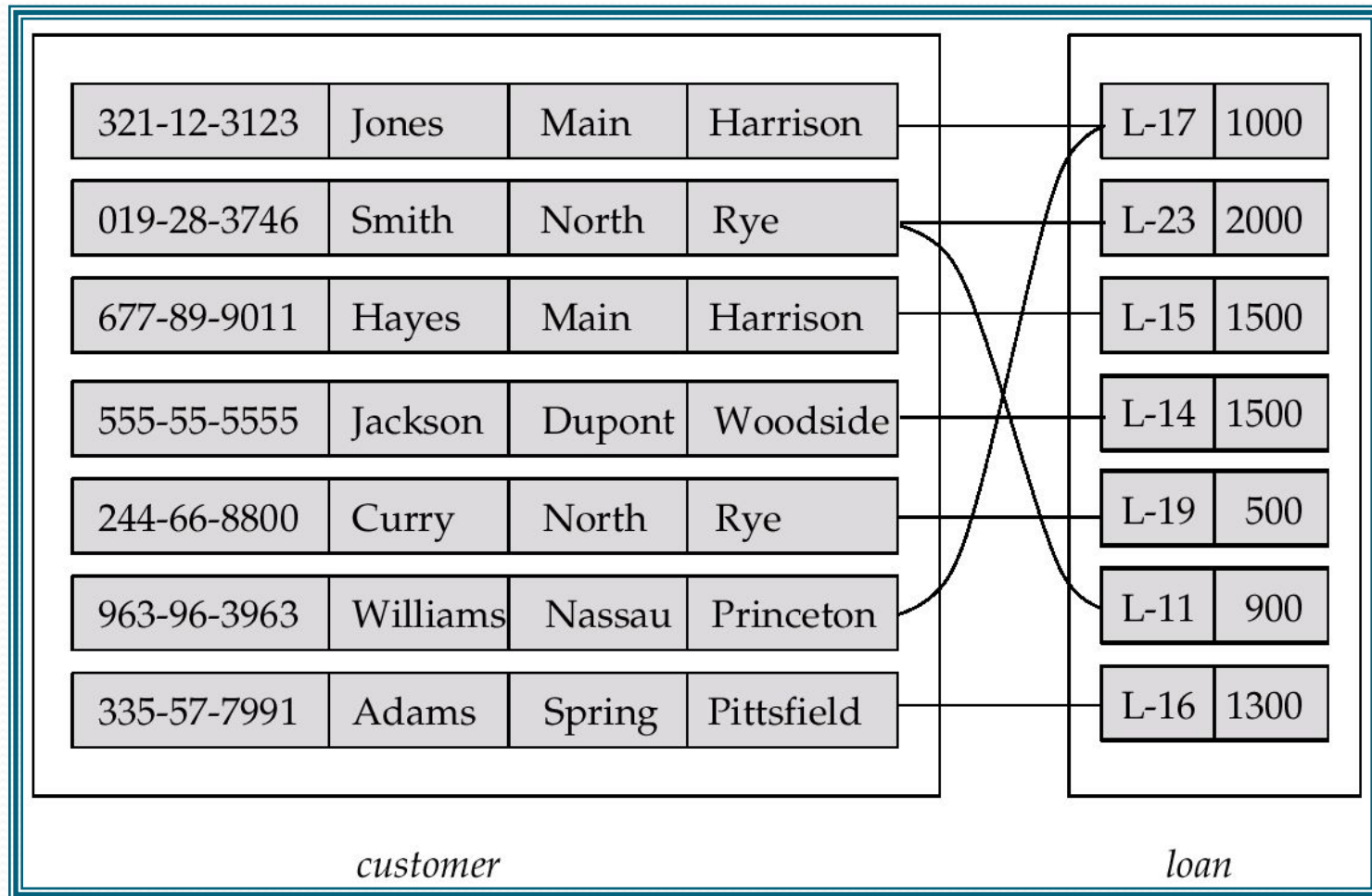
$$\{(e_1, e_2, \dots, e_n) \mid e_1 \in E_1, e_2 \in E_2, \dots, e_n \in E_n\}$$

where (e_1, e_2, \dots, e_n) is a relationship

- Example:

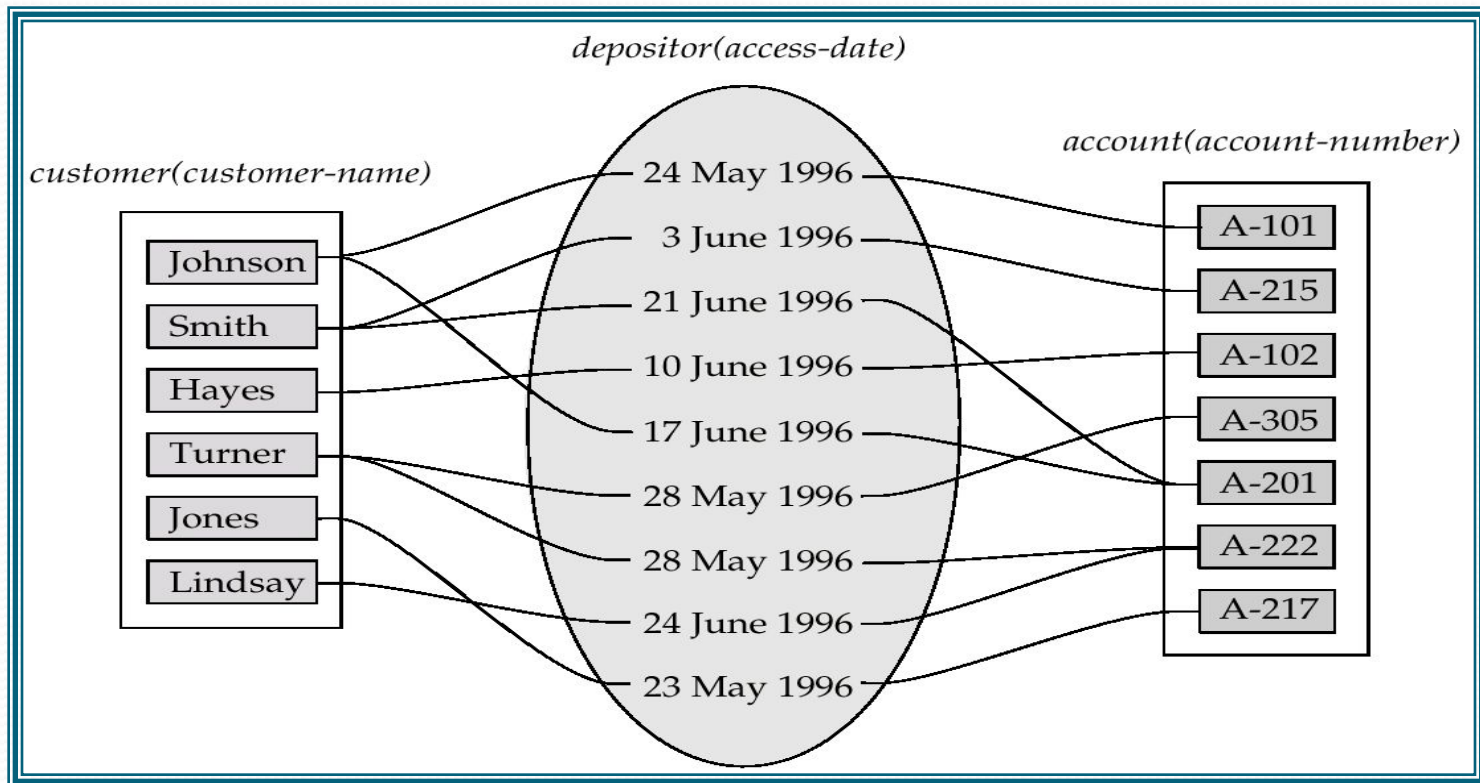
$(\text{Hayes}, \text{A-102}) \in \text{depositor}$

Relationship Set Borrower



Relationship Set

- 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

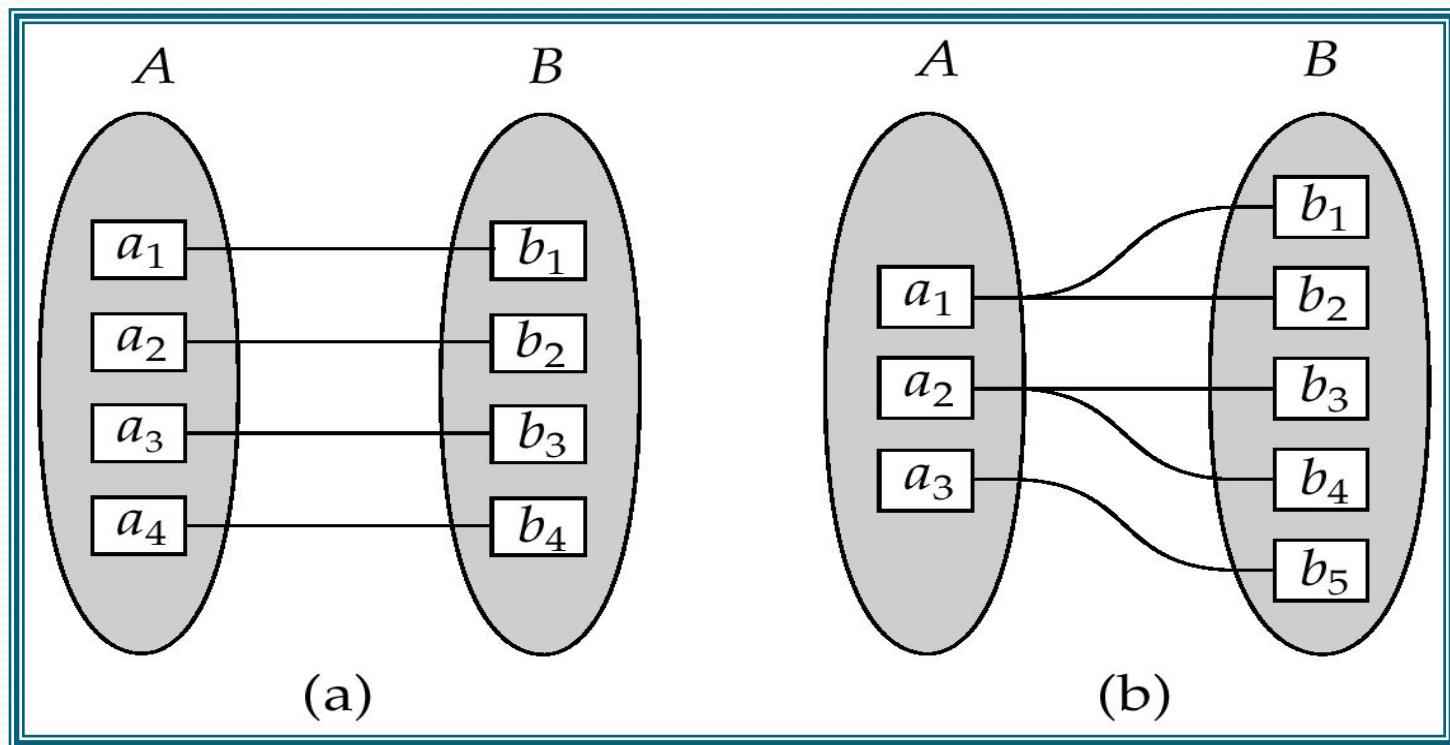
- Refers to number of entity sets that participate in a relationship set.
- Relationship sets that involve two entity sets are *binary* (or degree two). Generally, most relationship sets in a database system are binary.
- Relationship sets may involve more than two entity sets.
 - E.g. Suppose employees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

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

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

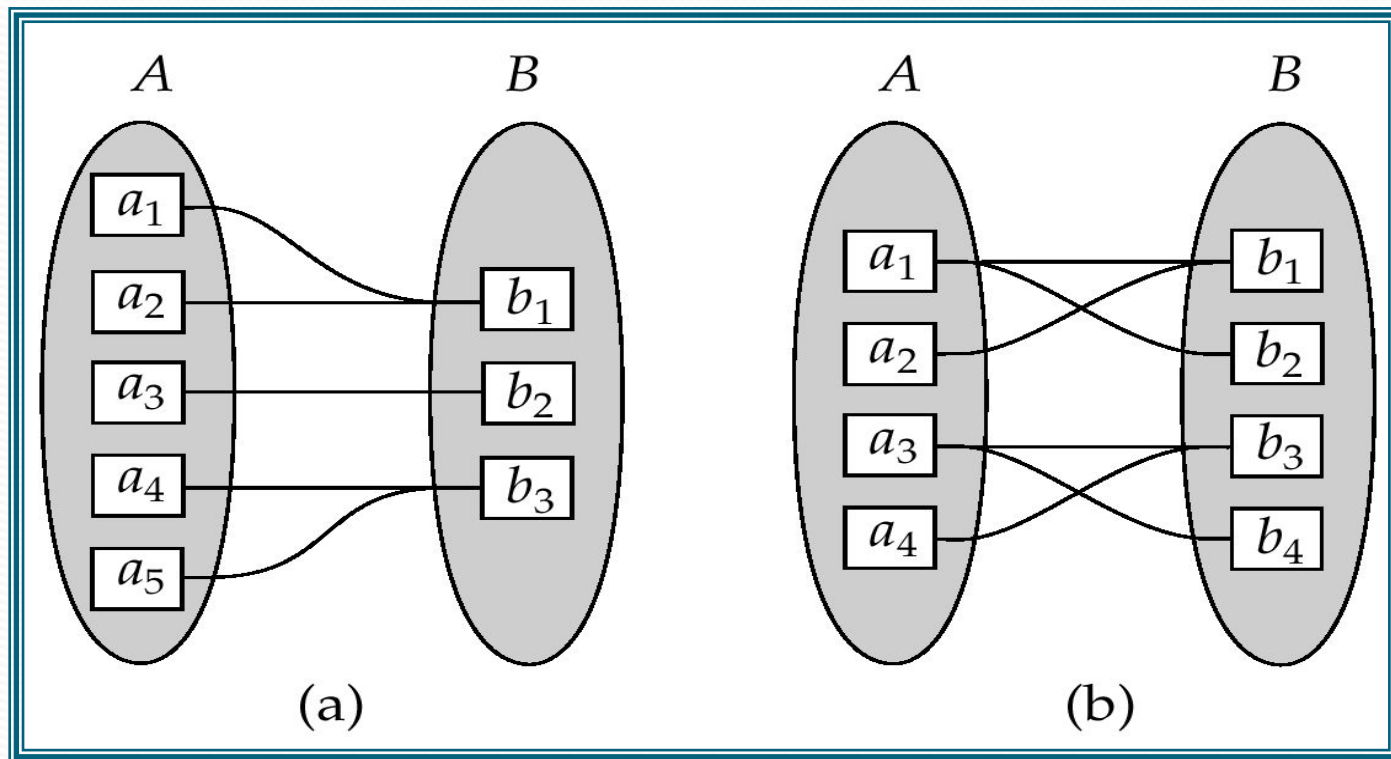


One to one

One to many

Mapping Cardinalities

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

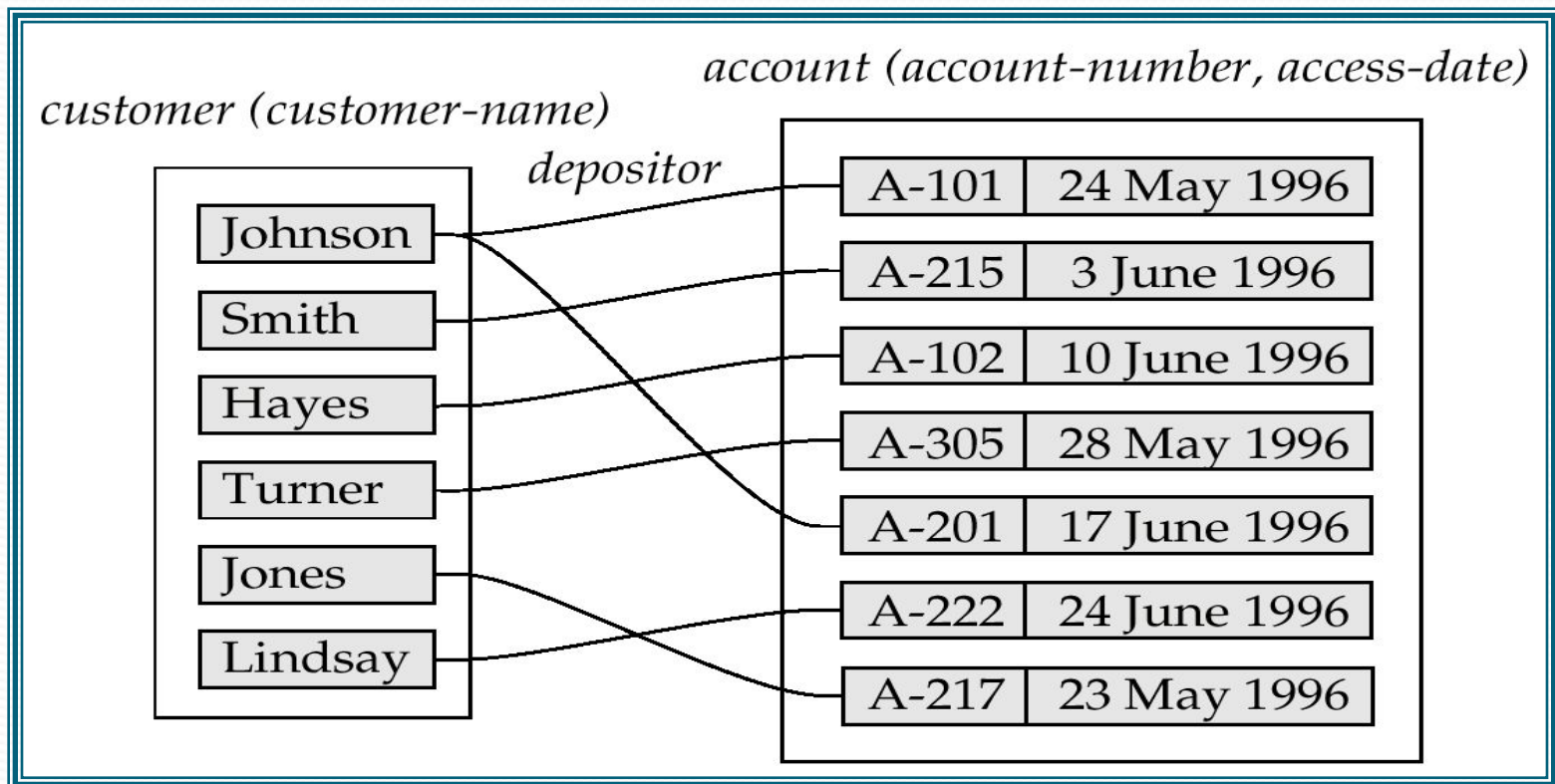


Many to one

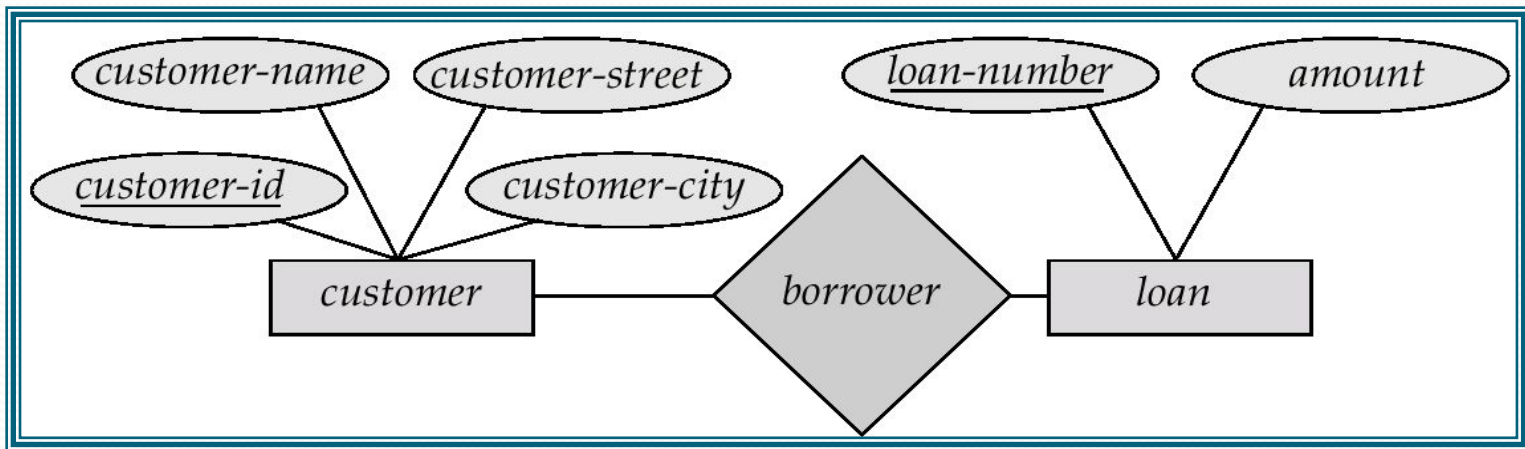
Many to many

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

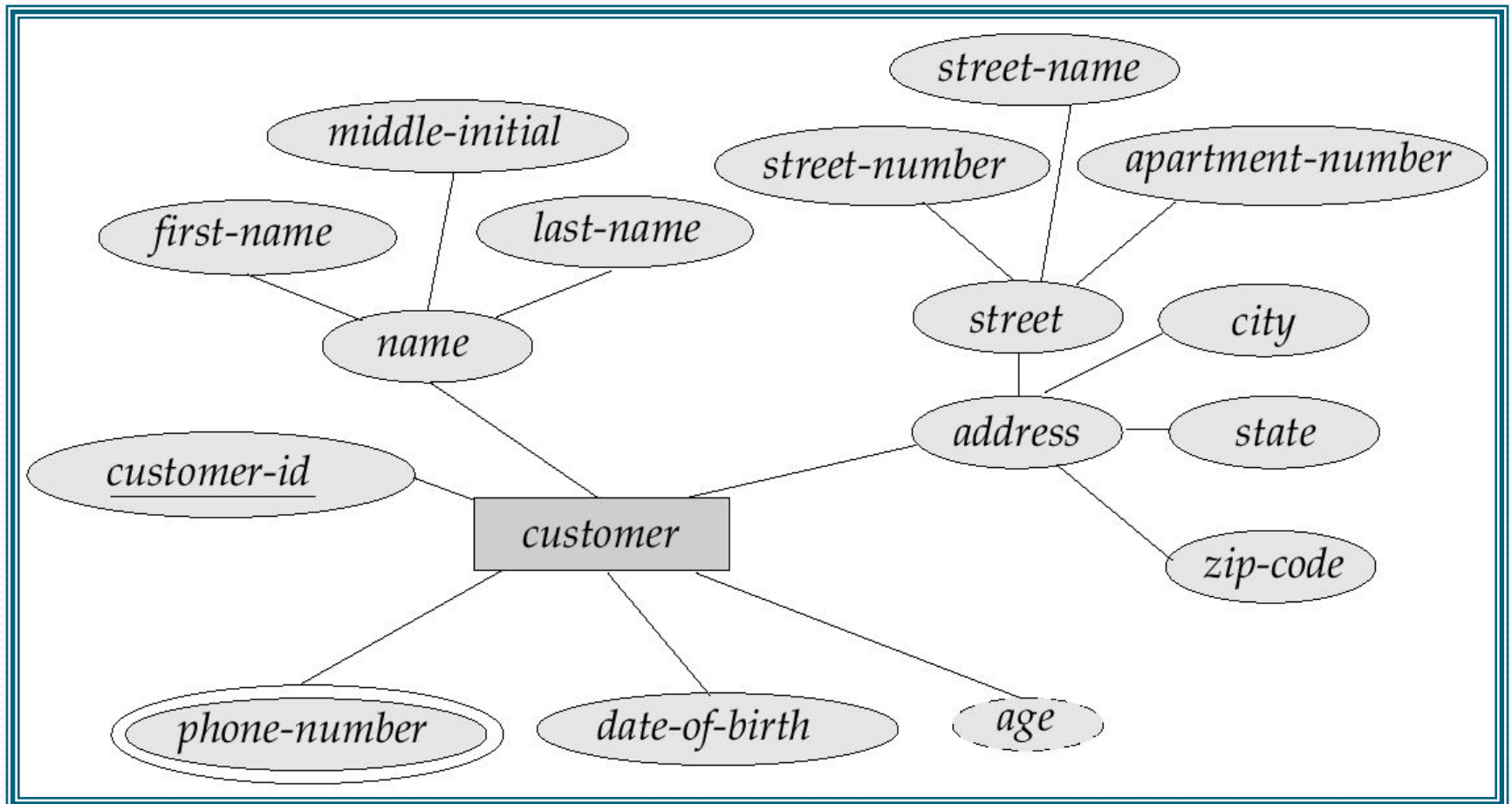


E-R Diagrams

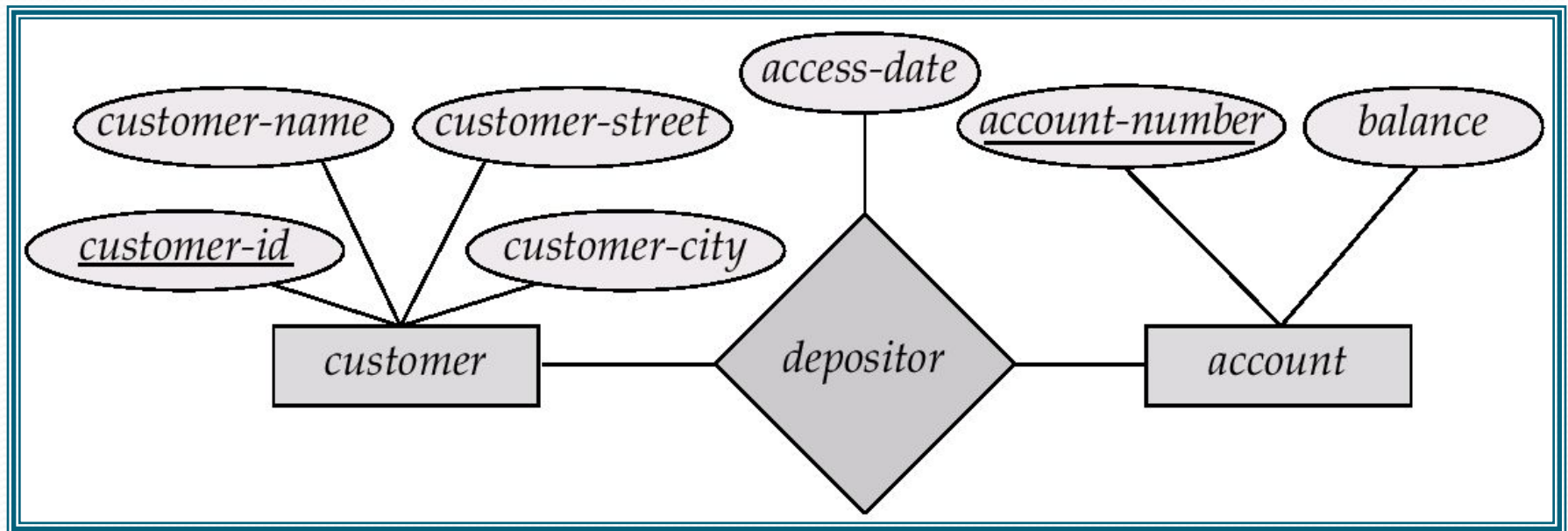


- **Rectangles** represent entity sets.
- **Diamonds** represent relationship sets.
- **Lines** link attributes to entity sets and entity sets to relationship sets.
- **Ellipses** represent attributes
 - **Double ellipses** represent multivalued attributes.
 - **Dashed ellipses** denote derived attributes.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

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

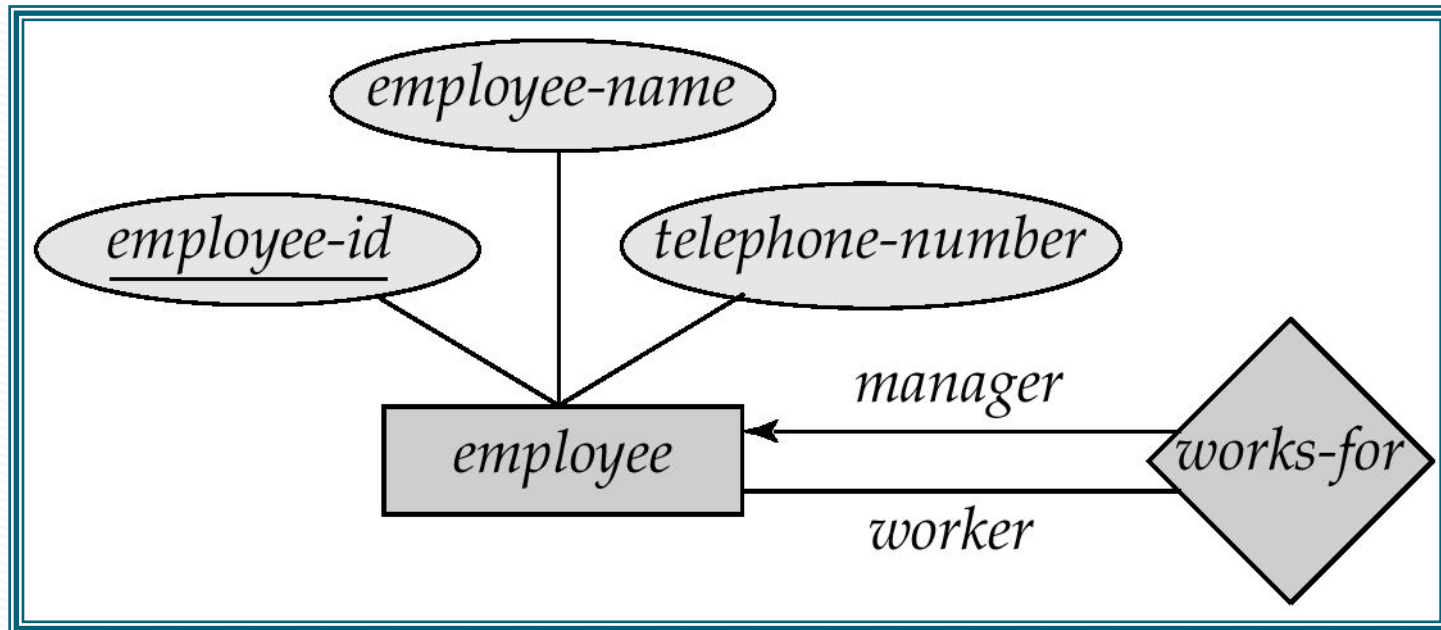


Relationship Sets with Attributes



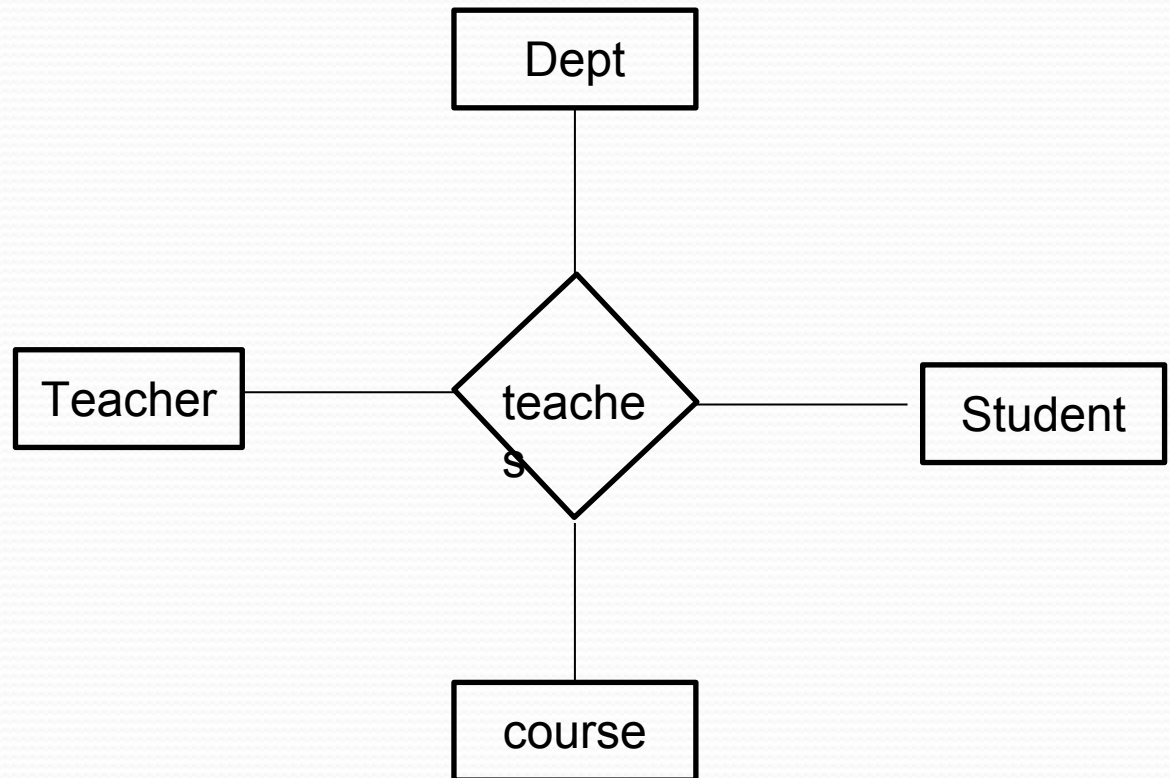
Roles

- Entity sets of a relationship need not be distinct
- 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



Degrees of A Relationship set

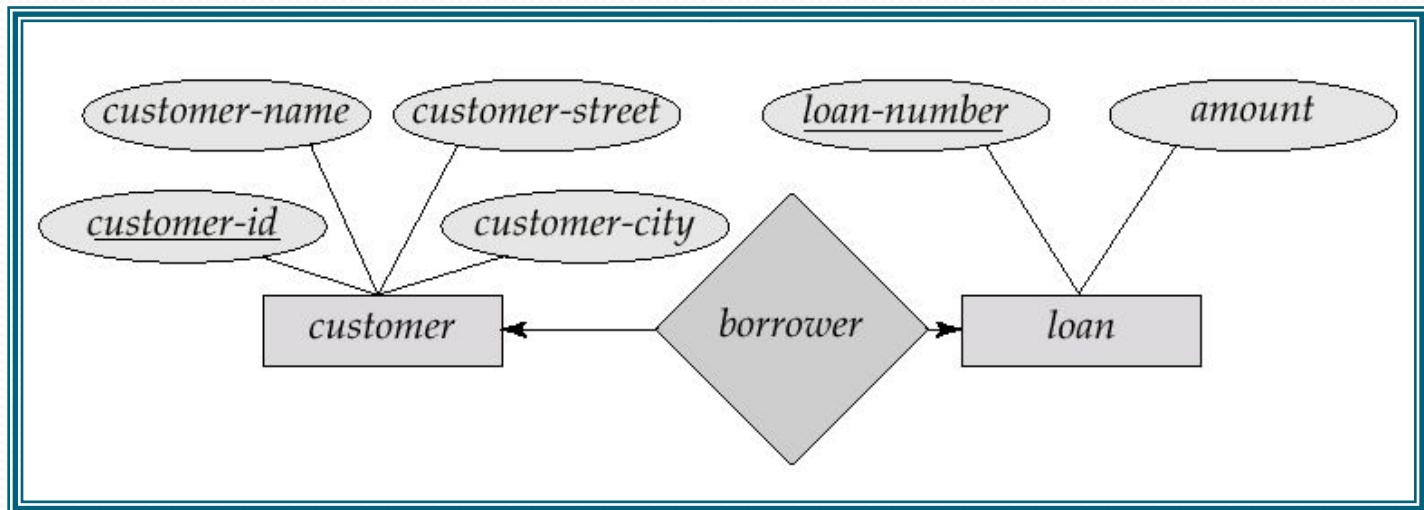
- Unary
- Binary
- Ternary
- Quaternary
- N-ary



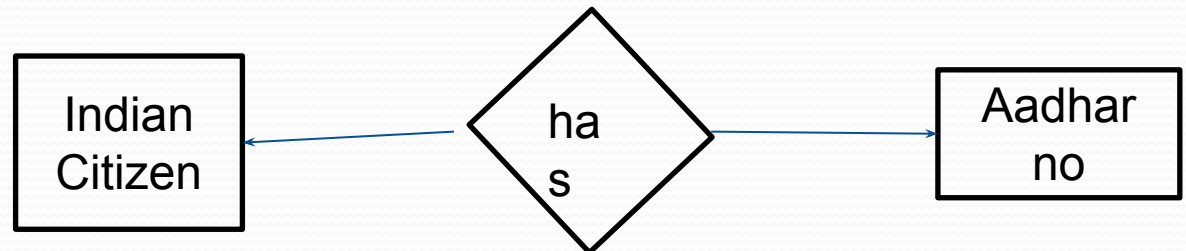
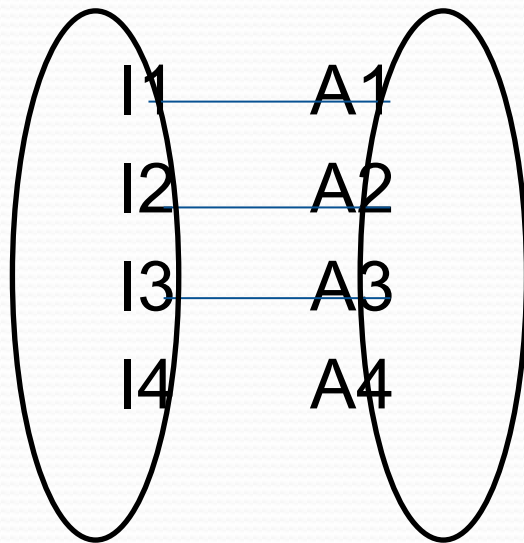
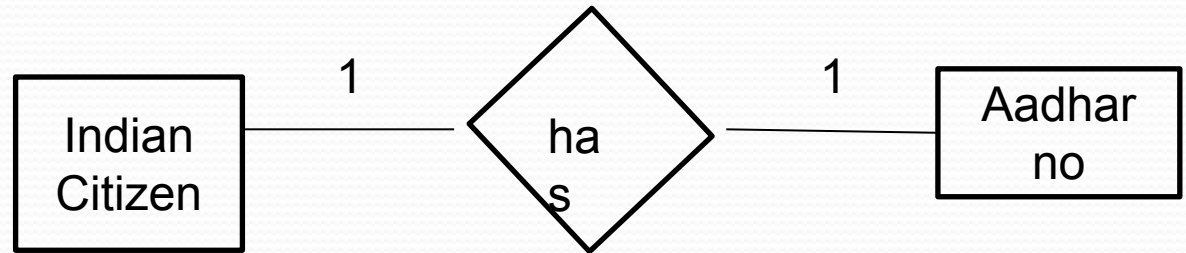
- Note: Most of relationship sets in ER- Diagram are binary, occasionally however relationship sets involve more than two entity sets.

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*
 - A loan is associated with at most one customer via *borrower*

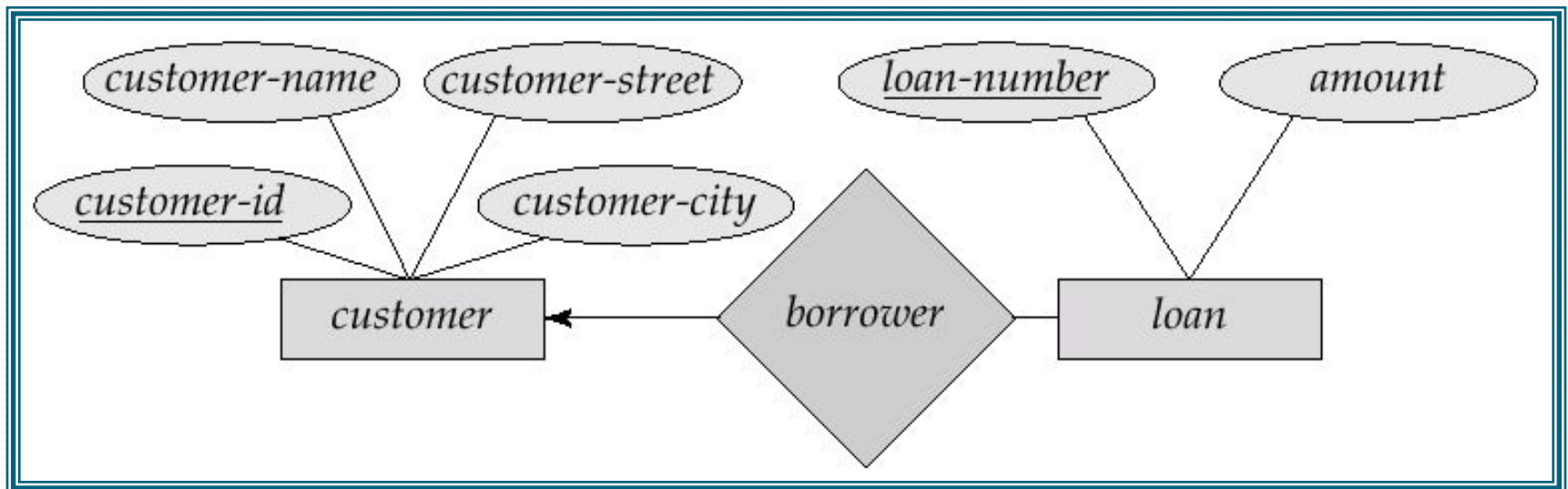


One to One

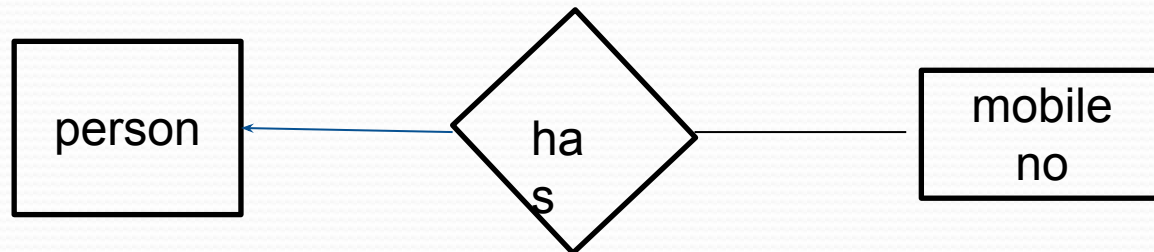
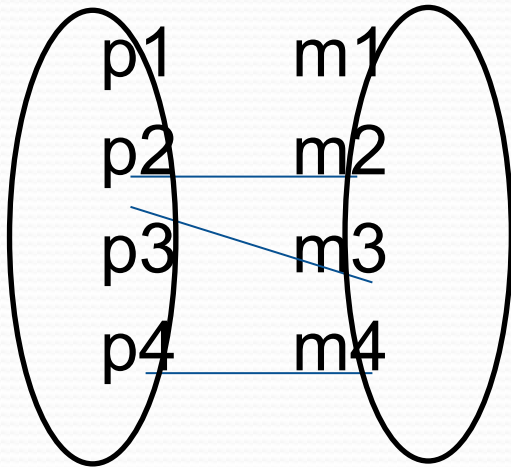
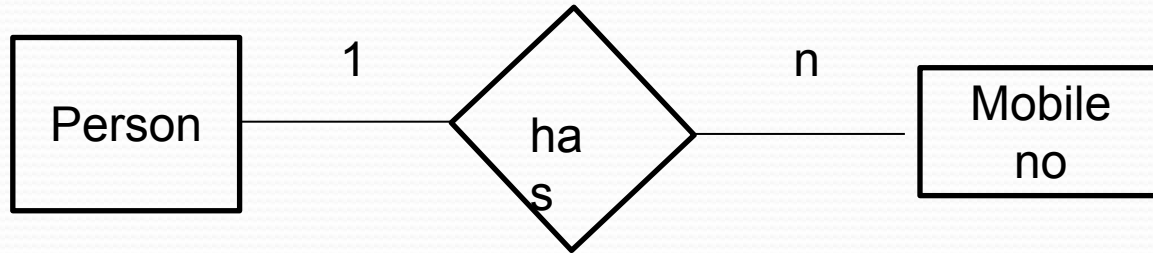


One-To-Many Relationship

- 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*

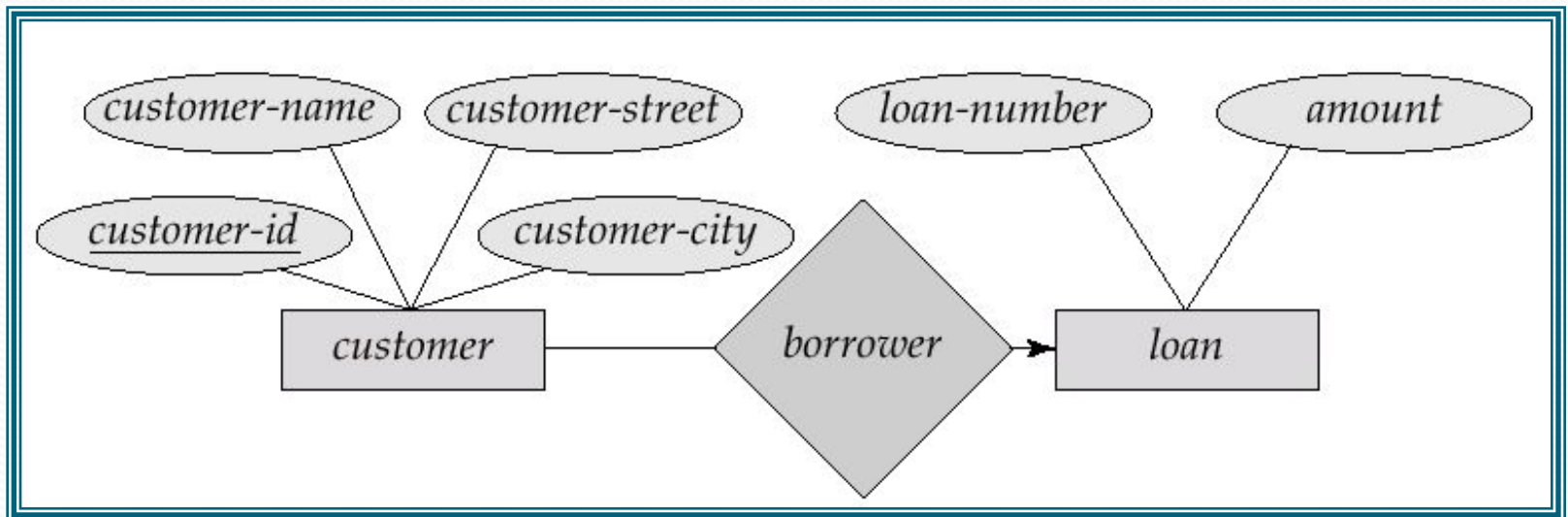


One to Many

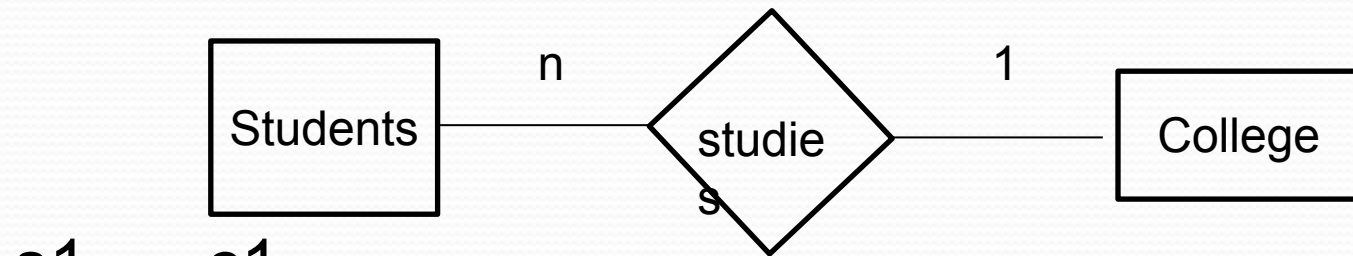


Many-To-One Relationships

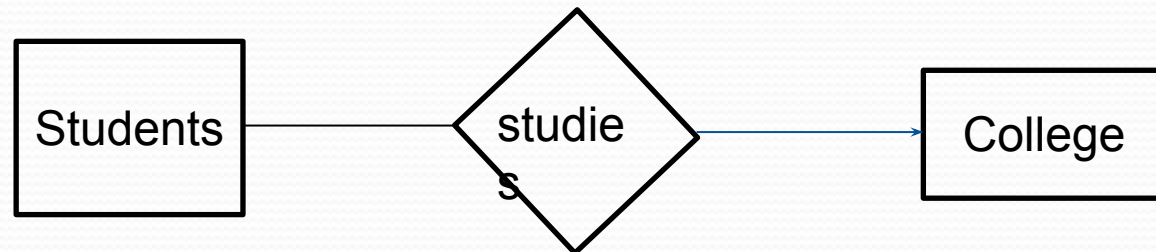
- In a many-to-one relationship a loan is associated with several (including 0) customers via *borrower*, a customer is associated with at most one loan via *borrower*



Many to One

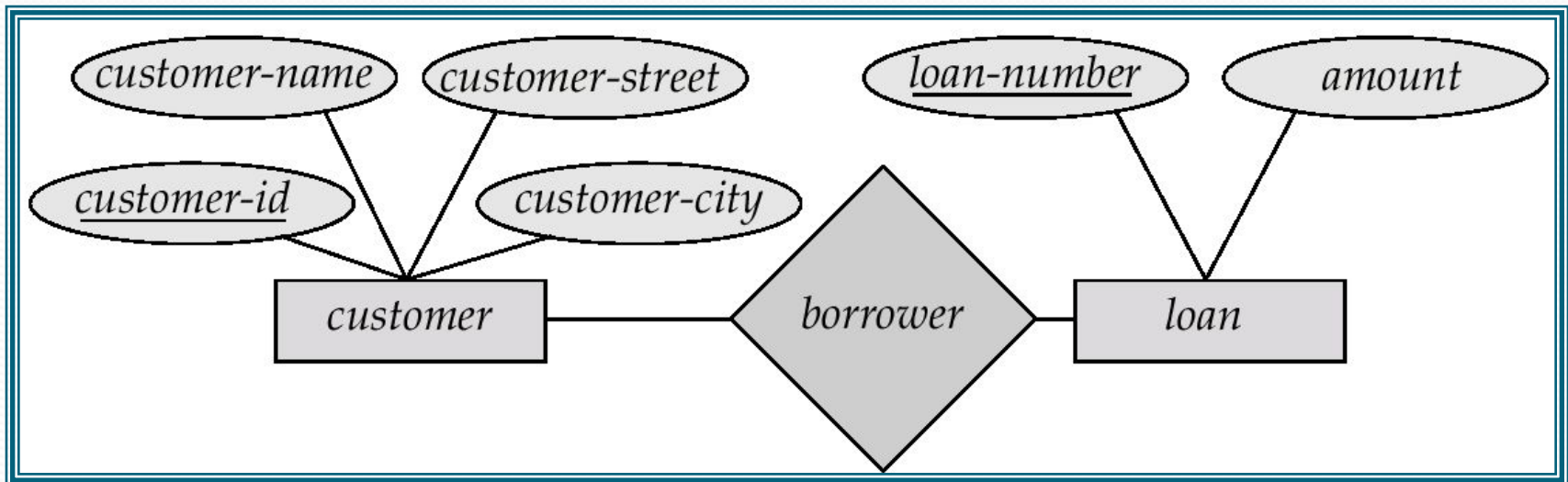


s1 c1
s2 c2
s3 c3
s4 c4

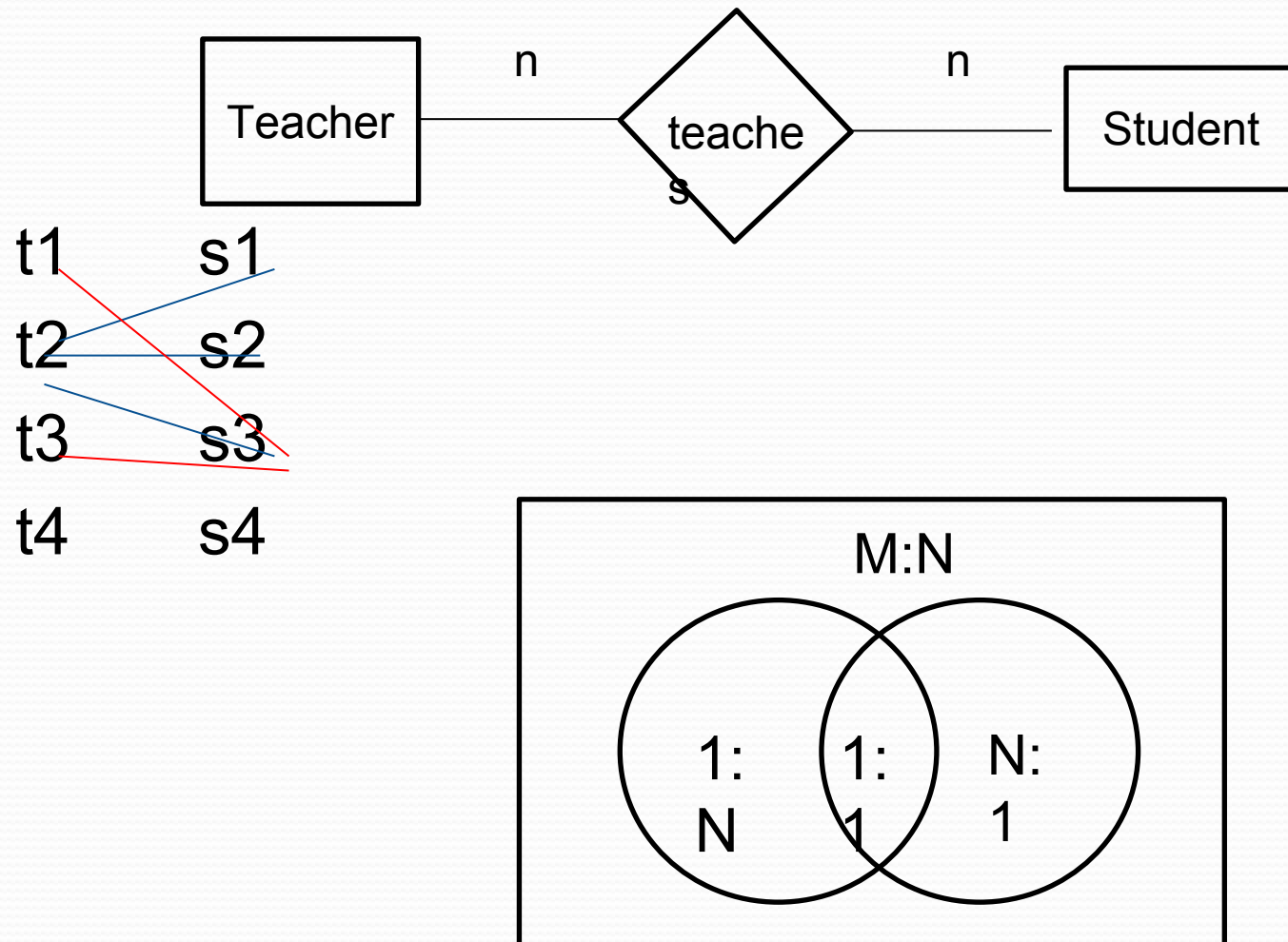


Many-To-Many Relationship

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

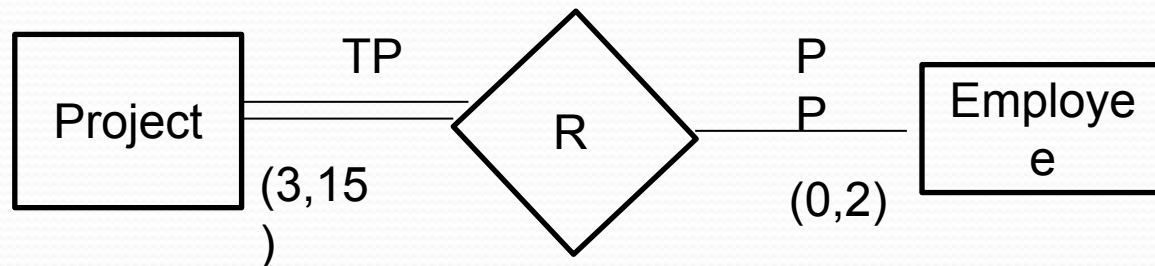


Many to Many



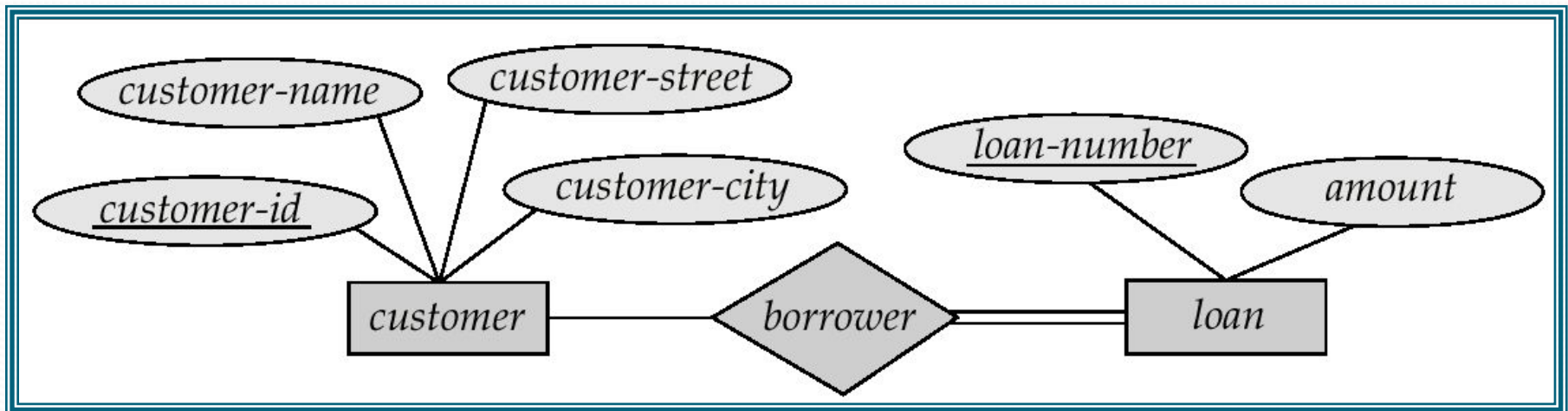
Participation Constraints

- Specifies whether the existence of an entity depends on its being related to another entity via a relationship type
- These constraints specify the minimum and maximum number of relationship instances that each entity can /must participate in.
- Max Cardinality : It defines maximum number of times an entity occurrence participated in relationship.
- Min Cardinality : It defines minimum number of times an entity occurrence participated in relationship.



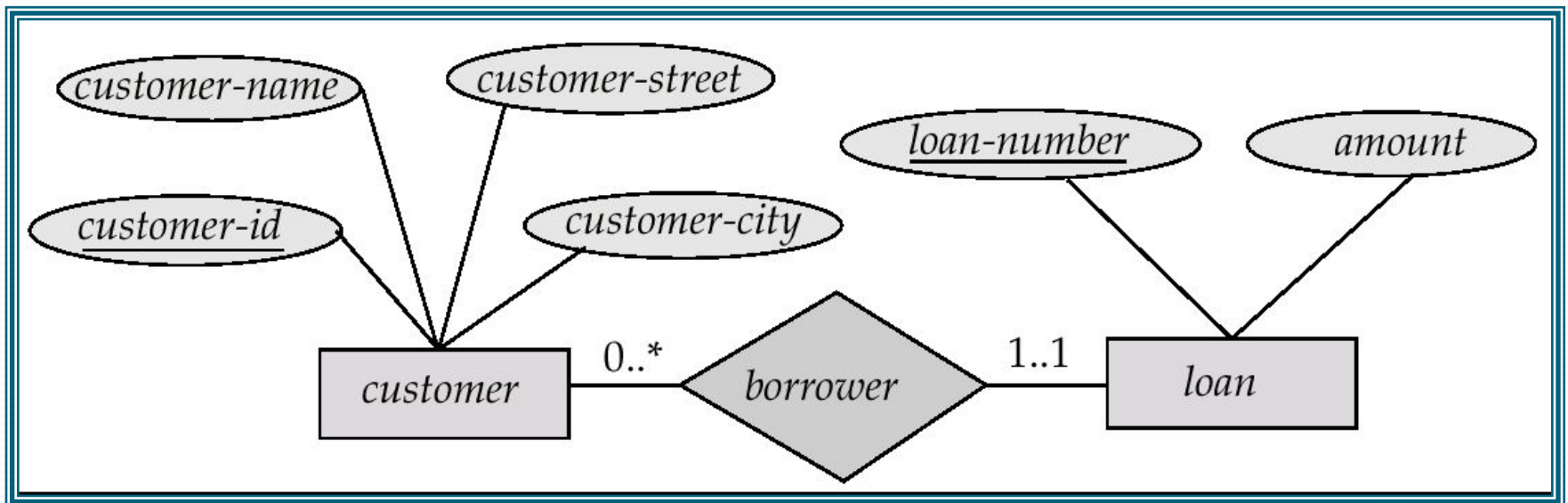
Participation Constraints

- 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



Keys

- **KEYS in DBMS** is an attribute or set of attributes which helps you to identify a row(tuple) in a relation(table)

Emp id	First Name	Last Name	Dept.
101	abc	xyz	A
102	def	xyz	A
103	abc	pqr	A

- Key -> Emp id
- Key - >(First name, Last name)

Types of Keys

- A *super key* of an entity set is a set of one or more attributes whose values uniquely determine each row in a table.

1) R(ABCD)

$A \rightarrow BC$

$(A) \Rightarrow (ABC) \nexists$

Note : Here A cannot be a super key.

2) R(ABCD)

$ABC \rightarrow D \quad (ABC) \Rightarrow ABCD$

$AB \rightarrow CD \quad (AB) \Rightarrow ABCD$

$A \rightarrow BCD \quad (A) \Rightarrow ABCD$

Note: Here all three are super key

Types of Keys

- A *candidate key* of an entity set is a minimal super key
- R(ABCD)
ABC \rightarrow D (ABC) \Rightarrow ABCD
AB \rightarrow CD (AB) \Rightarrow ABCD
A \rightarrow BCD (A) \Rightarrow ABCD
- Here A is a Candidate key
- R(ABCD)
B \rightarrow ACD (B) \Rightarrow ACD
ACD \rightarrow B (ACD) \Rightarrow B
- Here both B , ACD are **super keys** as well as **candidate keys**

Types of Keys

- Although several candidate keys may exist, one of the candidate keys is selected to be the *primary key*.
- R(ABCD)
B \rightarrow ACD (B) \Rightarrow ACD
ACD \rightarrow B (ACD) \Rightarrow B
- Candidate Keys \Rightarrow B
ACD
- Primary Key \Rightarrow either (B) or (ACD)

Types of Keys

- Identify Super Key, Candidate Key, Primary key from below :-
- R(ABCD)
 - AB \rightarrow C
 - C \rightarrow BD
 - D \rightarrow A

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

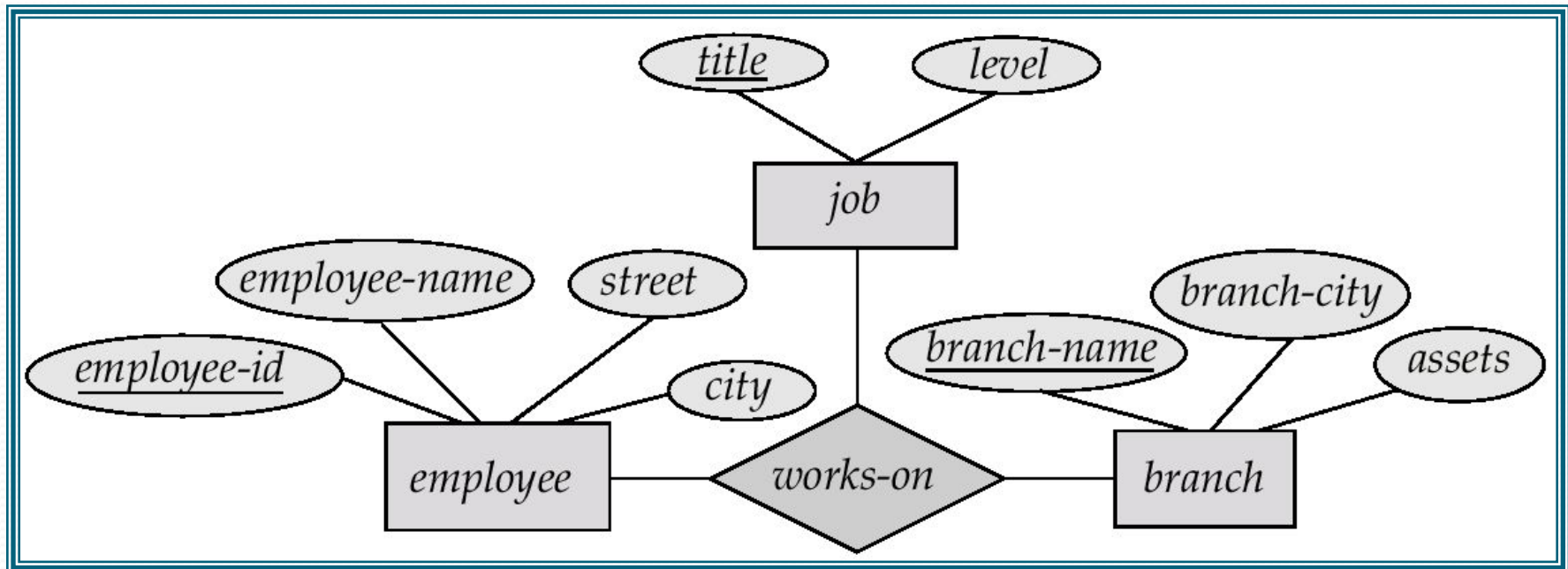
Cust_id	Acc_no	Access Date
C101	A10	1/1/20
C102	A11	1/1/20
C101	A10	1/2/20
C101	A10	1/3/20

Customer

Keys for Relationship Sets

- Must consider the mapping cardinality of the relationship set when deciding , 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

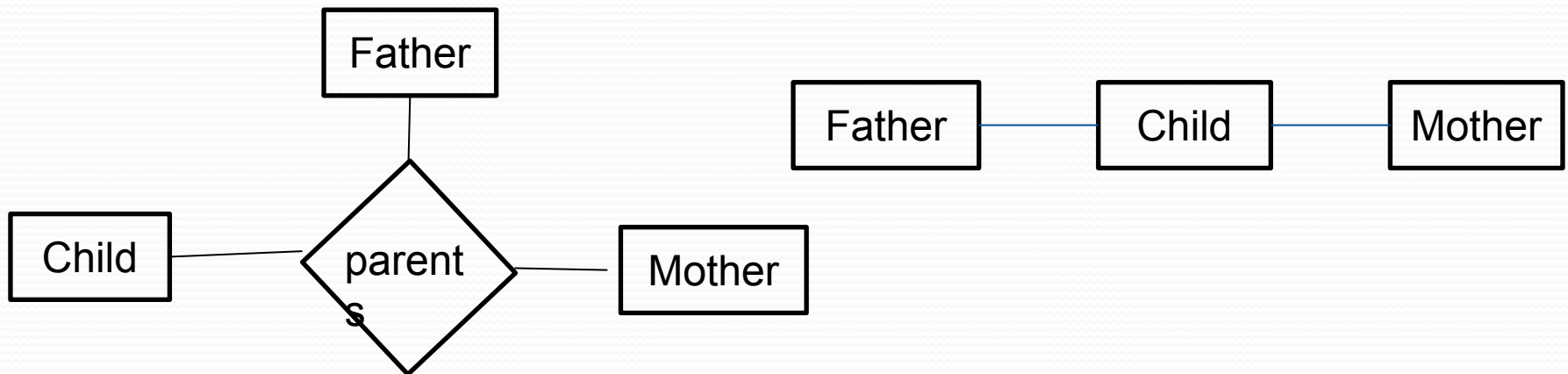


Cardinality Constraints on Ternary Relationship

- We allow at most one arrow out of a ternary (or greater degree) relationship to indicate a cardinality constraint
 - E.g. an arrow from works-on to job indicates each employee works on at most one job at any branch.
- If there is more than one arrow, there are two ways of defining the meaning.
 - E.g a ternary relationship R between A, B and C with arrows to B and C could mean
 1. each A entity is associated with a unique entity from B and C or
 2. each pair of entities from (A, B) is associated with a unique C entity, and each pair (A, C) is associated with a unique B
- Each alternative has been used in different formalisms

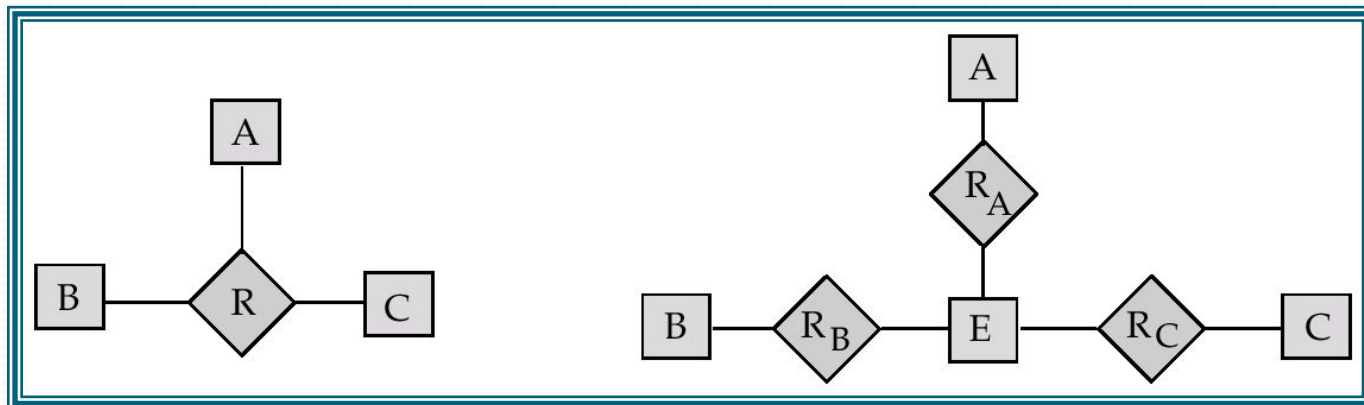
Binary Vs. Non-Binary Relationships

- 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)
- But there are some relationships that are naturally non-binary
 - E.g. works-on



Converting Non-Binary Relationships to Binary Form

- 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
 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



Converting Non-Binary Relationships

- Also need to translate constraints
 - Translating all constraints may not be possible
 - There may be instances in the translated schema that cannot correspond to any instance of R
- Exercise: add constraints to the relationships RA, RB and RC to ensure that a newly created entity corresponds to exactly one entity in each of entity sets A, B and C
- We can avoid creating an identifying attribute by making E a weak entity set identified by the three relationship sets

Design Issues

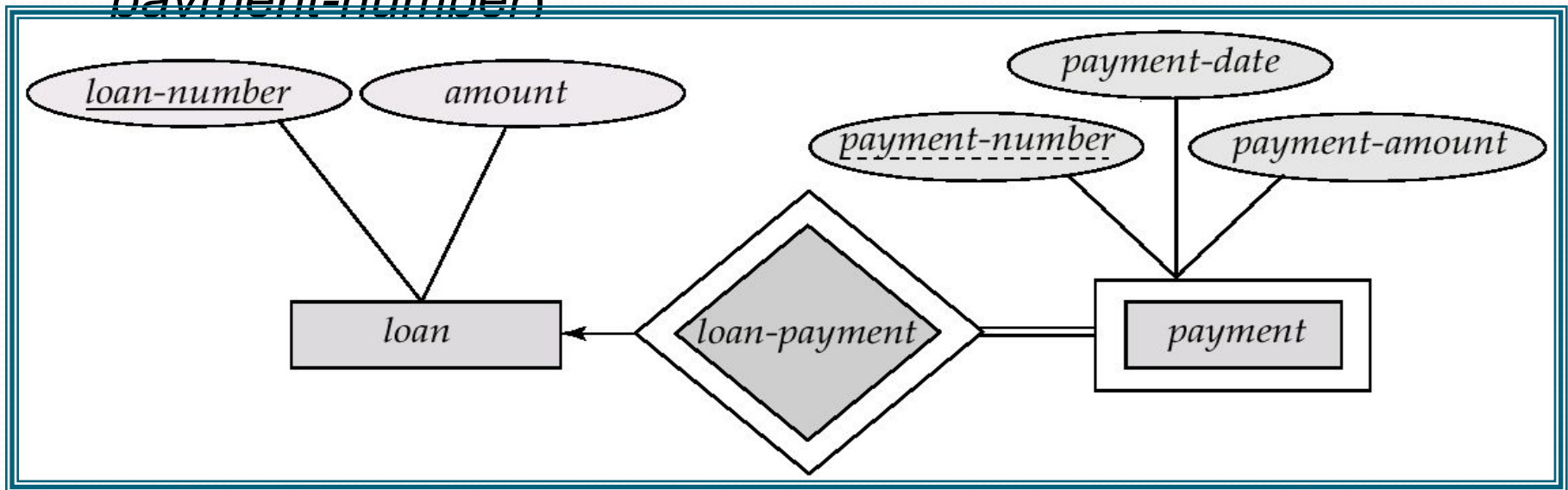
- Use of entity sets vs. attributes
 - Choice mainly depends on the structure of the enterprise being modeled, and on the semantics associated with the attribute in question.
- Use of entity sets vs. relationship sets
 - Possible guideline is to designate a relationship set to describe an action that occurs between entities
- Binary versus n -ary relationship sets
 - Although it is possible to replace any non binary (n -ary, for $n > 2$) relationship set by a number of distinct binary relationship sets, a n -ary relationship set shows more clearly that several entities participate in a single relationship.
- Placement of relationship attributes

Weak Entity Set

- 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 an *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's existence is dependent, plus the weak entity set's discriminator.

Weak Entity Set

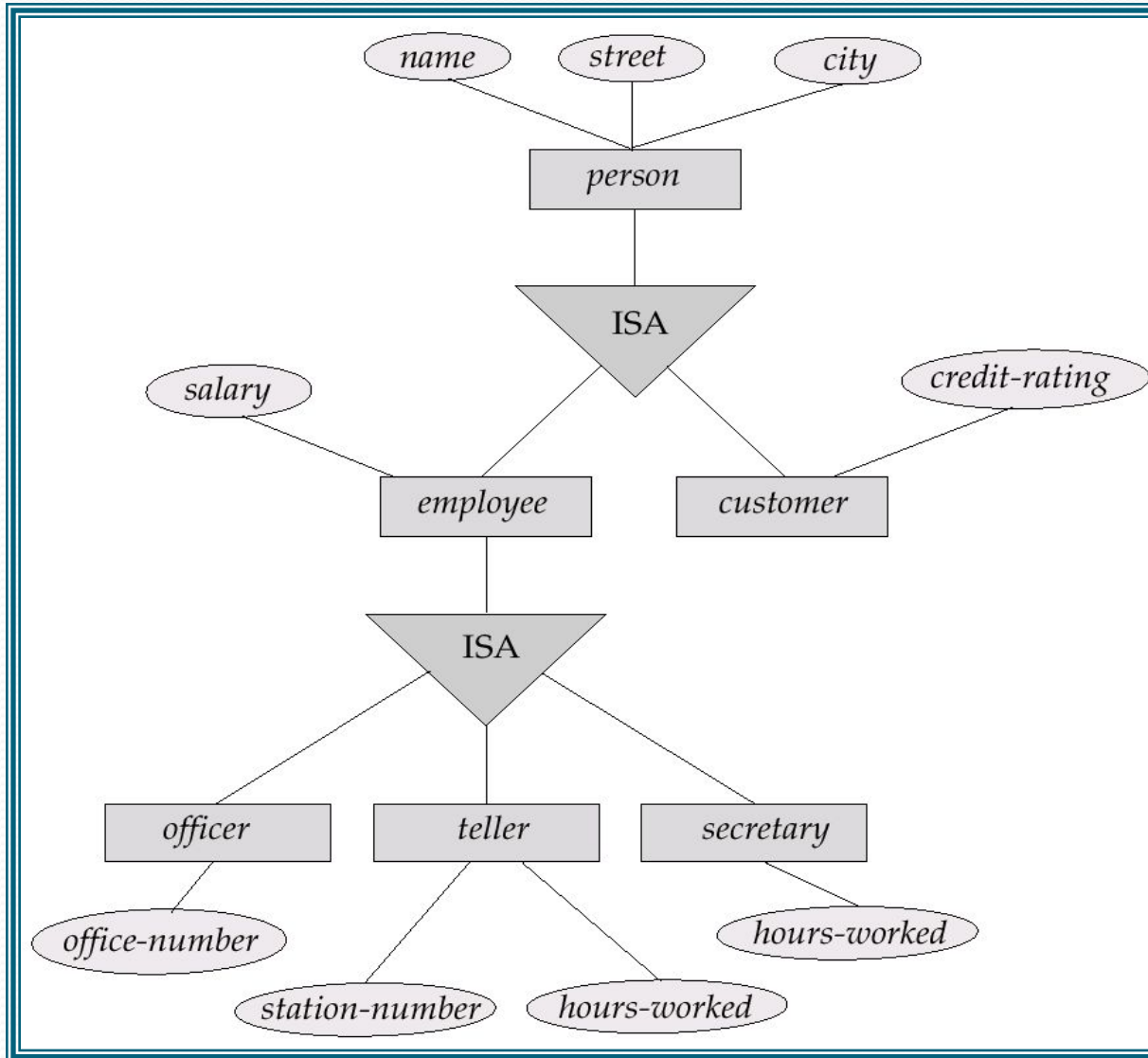
- We depict a weak entity set by double rectangles.
- We underline the discriminator of a weak entity set with a dashed line.
- *payment-number* – discriminator of the *payment* entity set
- Primary key for *payment* – (*loan-number*, *payment-number*)



Specialization

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

- 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

Design Constraints on a Specialization/Generalization

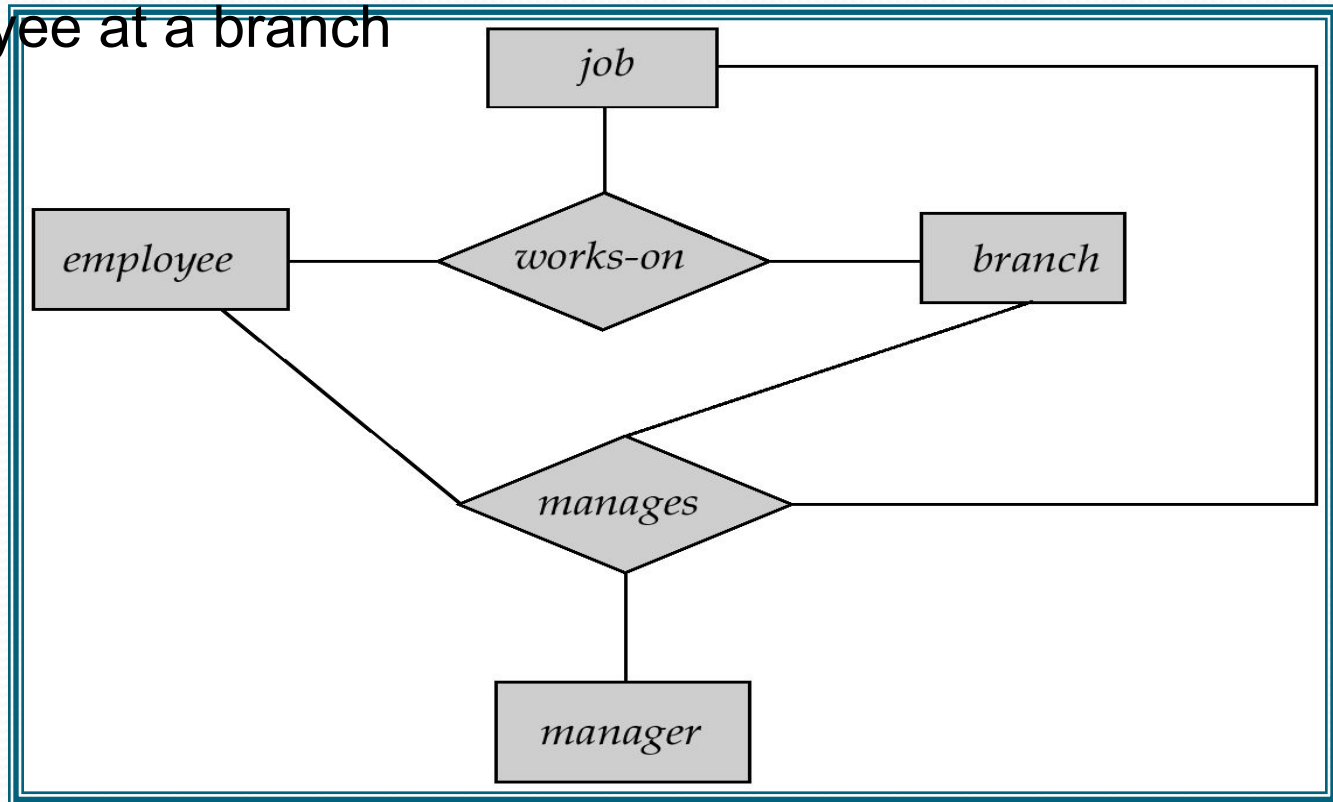
- Constraint on which entities can be members of a given lower-level entity set.
 - condition-defined
 - E.g. all customers over 65 years are members of *senior-citizen* entity set; *senior-citizen* ISA *person*.
 - user-defined
- Constraint on whether or not entities may belong to more than one lower-level 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

- **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

- 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

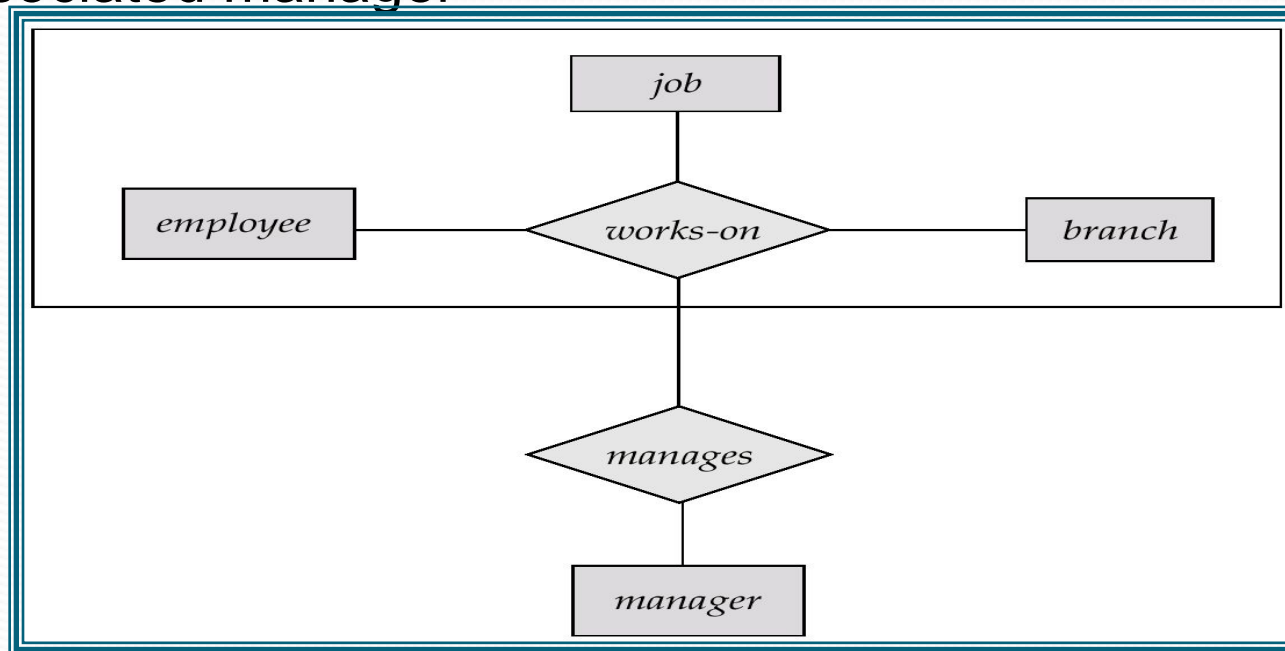


Aggregation

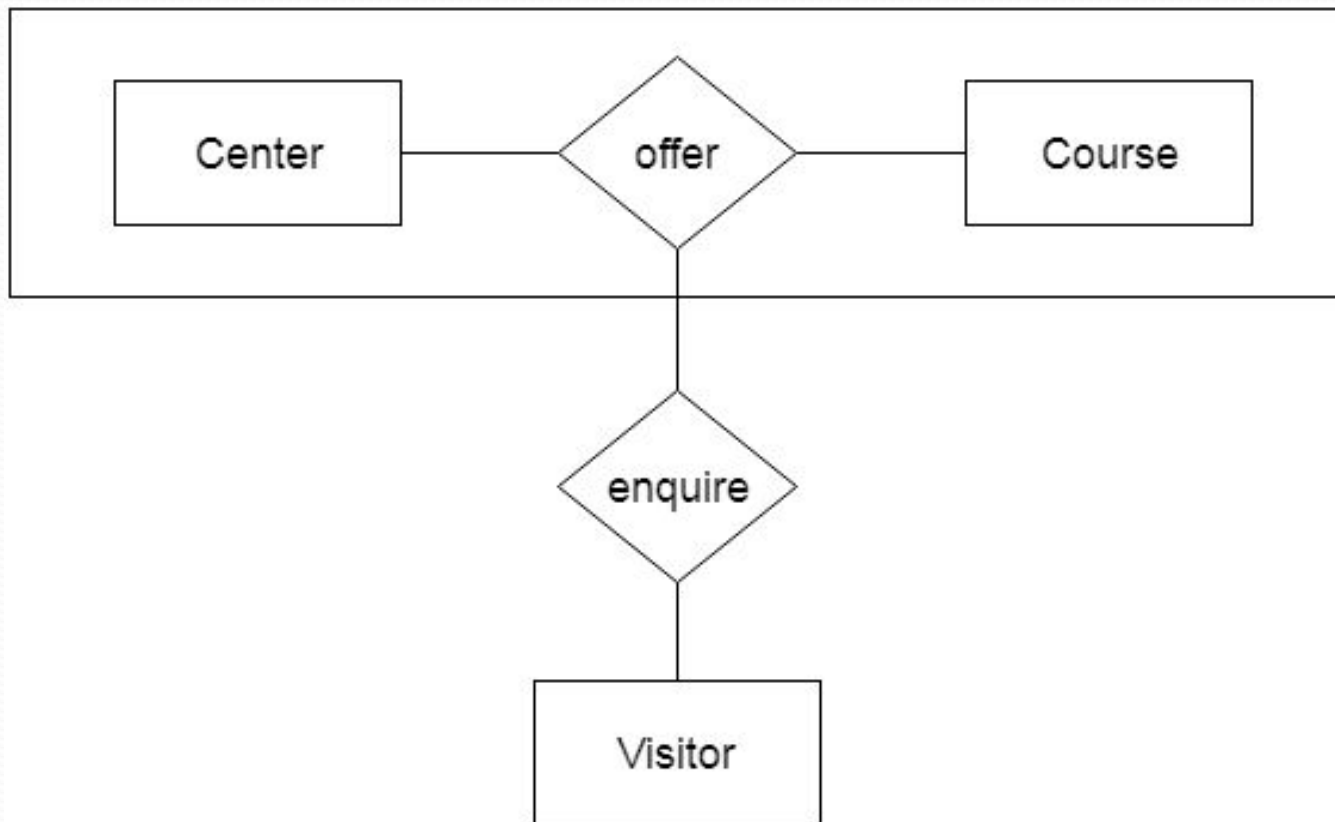
- Relationship sets *works-on* and *manages* represent overlapping information
 - Every *manages* relationship corresponds to a *works-on* relationship
 - However, some *works-on* relationships may not correspond to any *manages* relationships
 - So we can't discard the *works-on* relationship
- Eliminate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

E-R Diagram With Aggregation

- 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



E-R Design Decisions

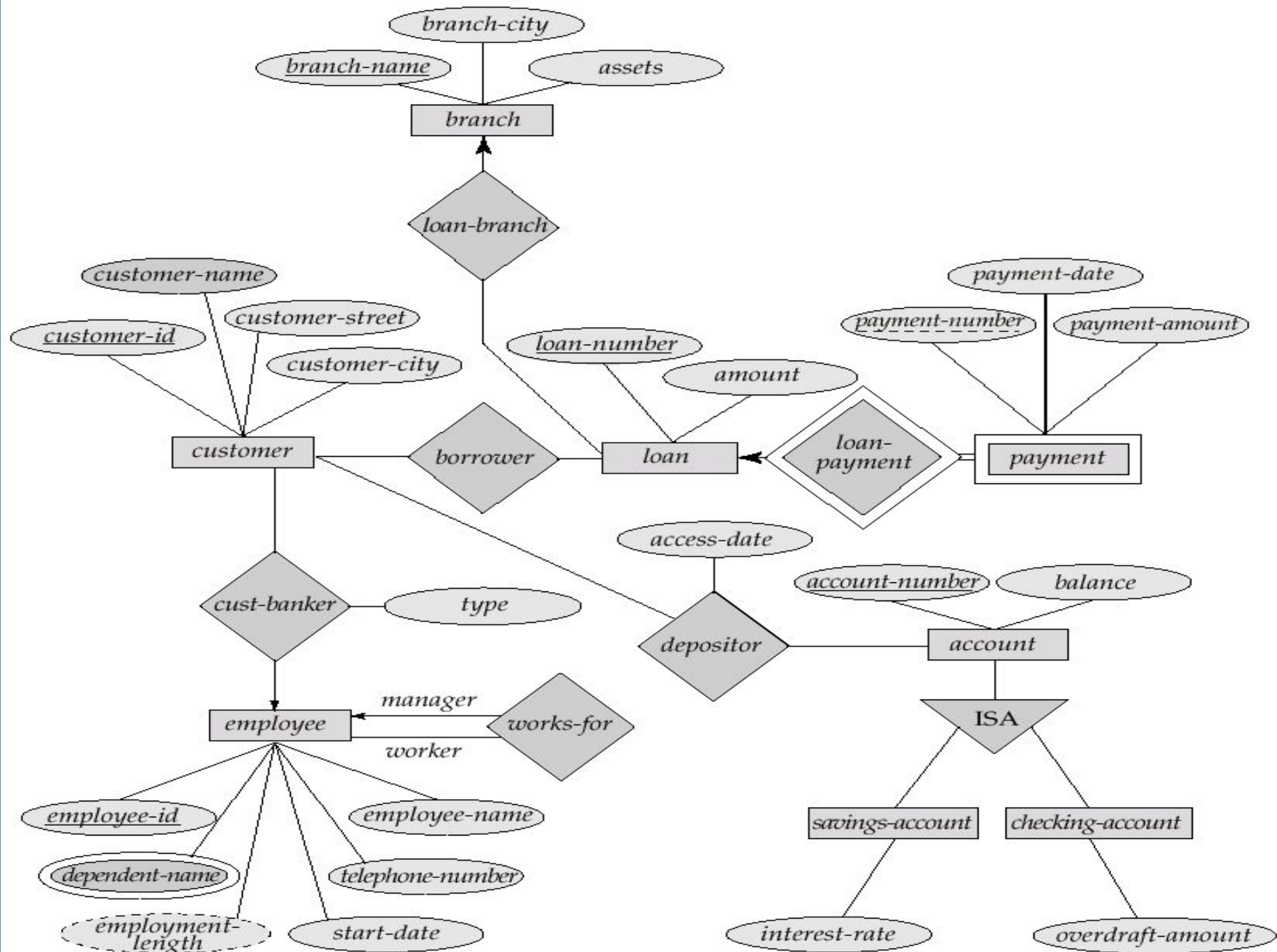
- The use of an attribute or entity set to represent an object.
- Whether a real-world concept is best expressed by an entity set or a relationship set.
- The use of a ternary relationship versus a pair of binary relationships.
- The use of a strong or weak entity set.
- The use of specialization/generalization – 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.

ER-model for Banking Enterprise





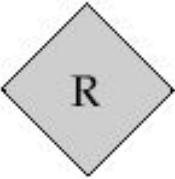
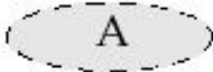

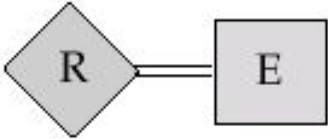

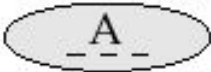
- Customer (cust_id, cust_nm, cust_addr, cust_mo)
- Employee(e_id, e_nm, e_addr)
- Account(acc_no, acc_bal)
- Saving Acc, Current Acc
- Branch(branch_nm, branch_city)
- Loan(loan_no, loan_amt)



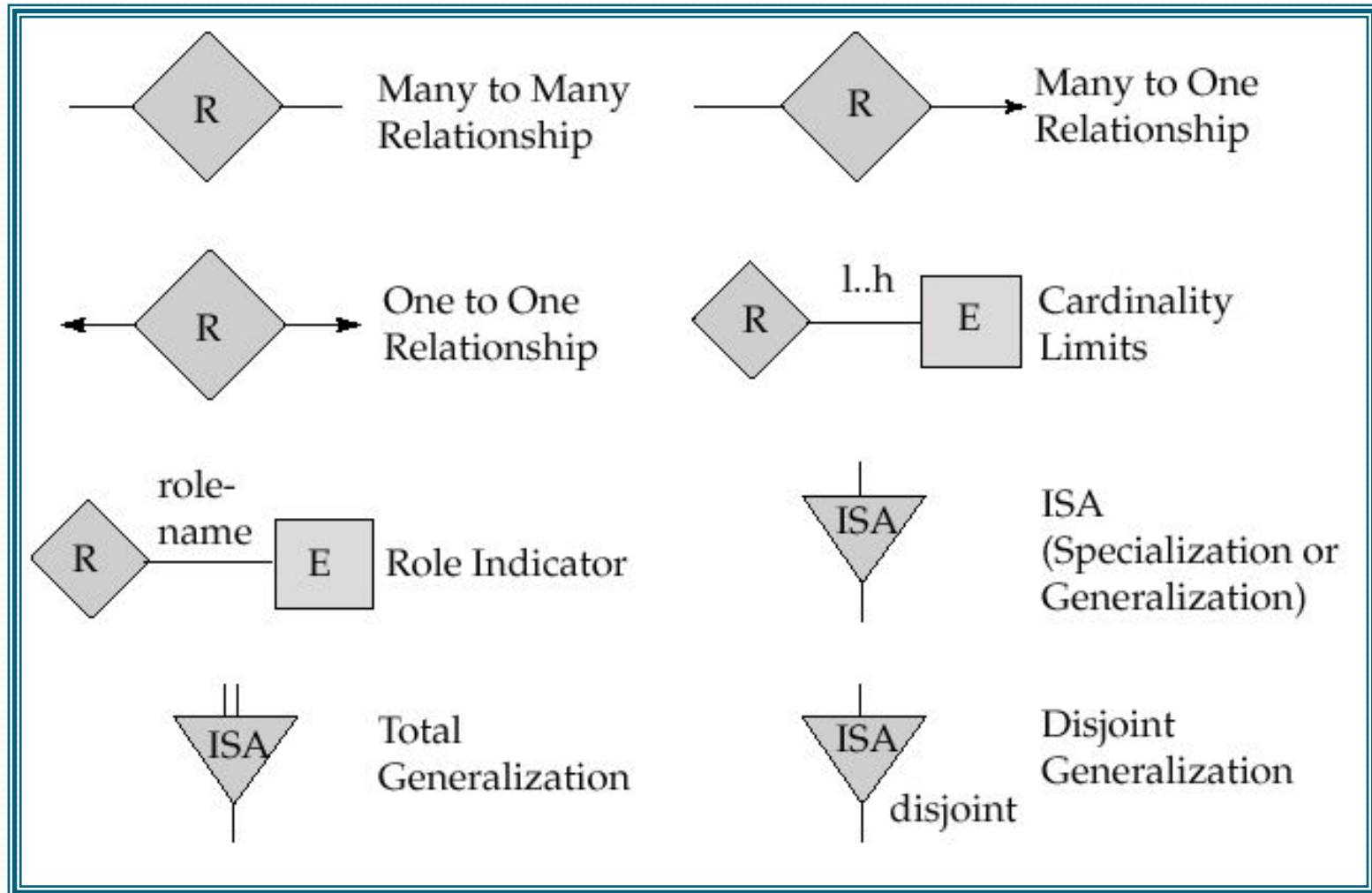
E-R Diagram for a Banking Enterprise



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

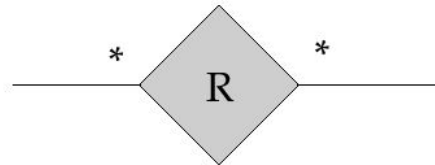


Alternative E-R Notations

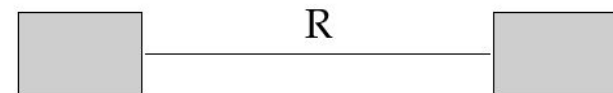
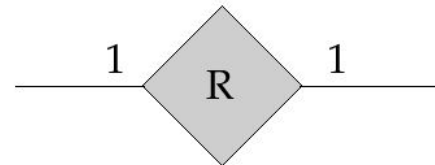
Entity set E with
attributes A1, A2, A3
and primary key A1

E	
A1	
A2	
A3	

Many to Many
Relationship



One to One
Relationship



Many to One
Relationship

