Chapter 2: Entity-Relationship Model

- Entity Sets
- Relationship Sets
- Design Issues
- Mapping Constraints
- Keys
- E–R Diagram
- Extended E-R Features
- Design of an E-R Database Schema
- Reduction of an E-R Schema to Tables

Entity Sets

- A *database* can be modeled as:
  - a collection of entities,
  - relationships among entities.
- An *entity* is an object that exists and is distinguishable from other objects.

  Example: specific person, company, event, plant
- 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-name*, *social-security*,
  customer-street, *customer-city*)

  account = (*account-number*, *balance*)

- *Domain* – the set of permitted values for each attribute

- Attribute types:
  - *Simple* and *composite* attributes.
  - *Single-valued* and *multi-valued* attributes.
  - *Null* attributes.
  - *Derived* attributes.

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

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

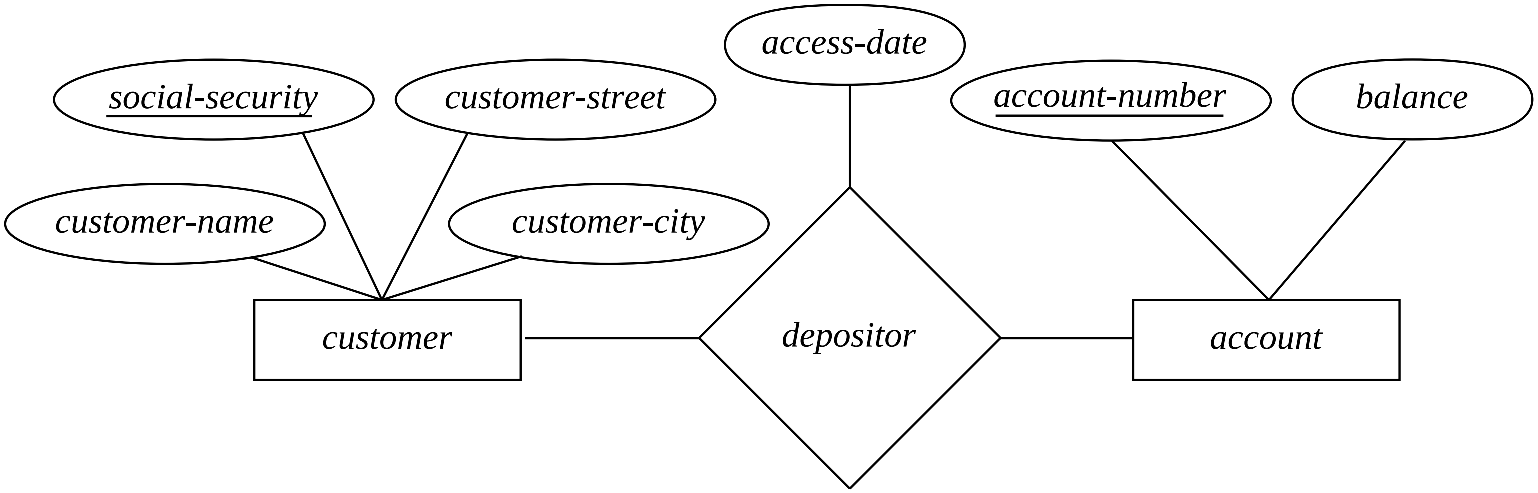
  where $(e_1, e_2, ..., e_n)$ is a relationship

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

Relationship Sets (Cont.)

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

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



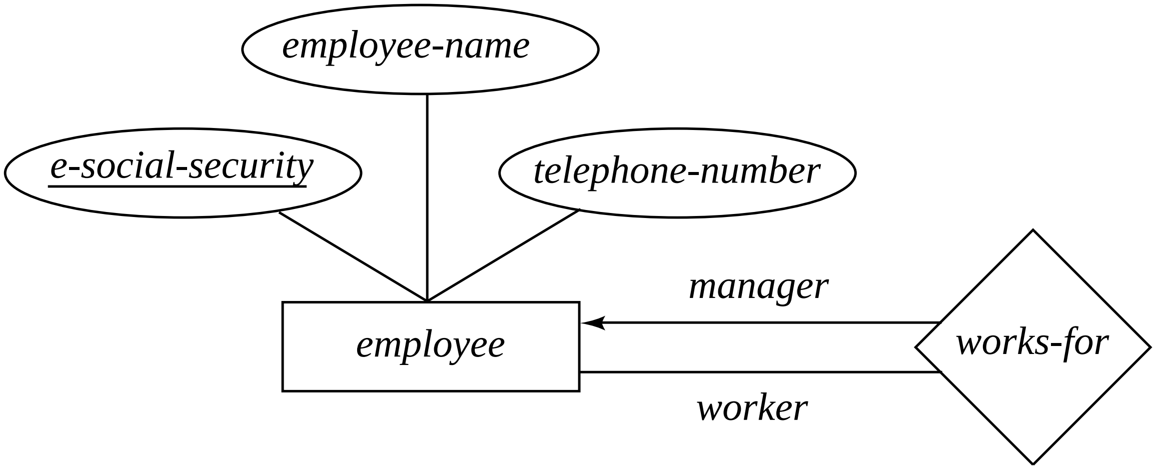
Degree of a Relationship Set

- 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.

  The entity sets *customer*, *loan*, and *branch* may be linked by the ternary (degree three) relationship set *CLB*.

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

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

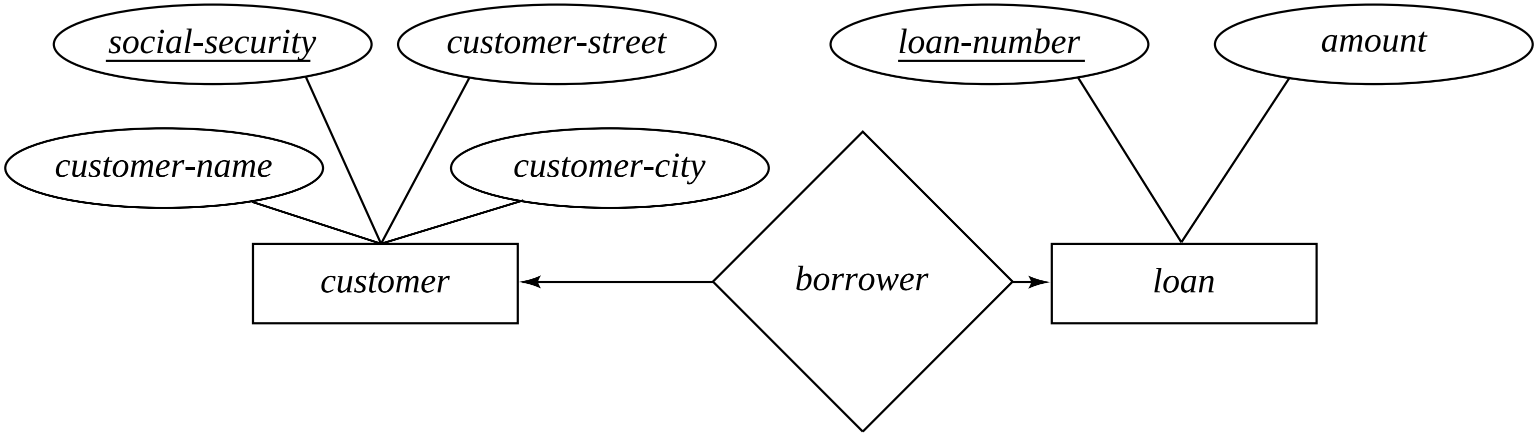
- Binary versus *n*-ary relationship sets

  Although it is possible to replace a nonbinary (*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.

Mapping Cardinalities

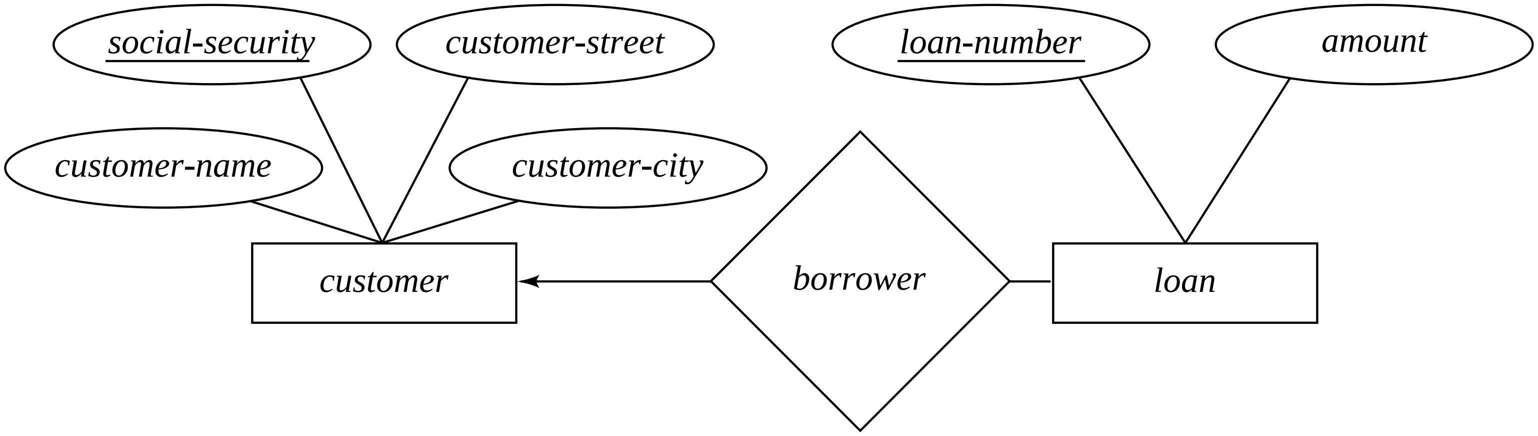
- Express the number of entities to which another entity can be associated via a relationship set.
- Most useful in describing binary relationship sets.
- For a binary relationship set the mapping cardinality must be one of the following types:
  - One to one
  - One to many
  - Many to one
  - Many to many
- We distinguish among these types 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

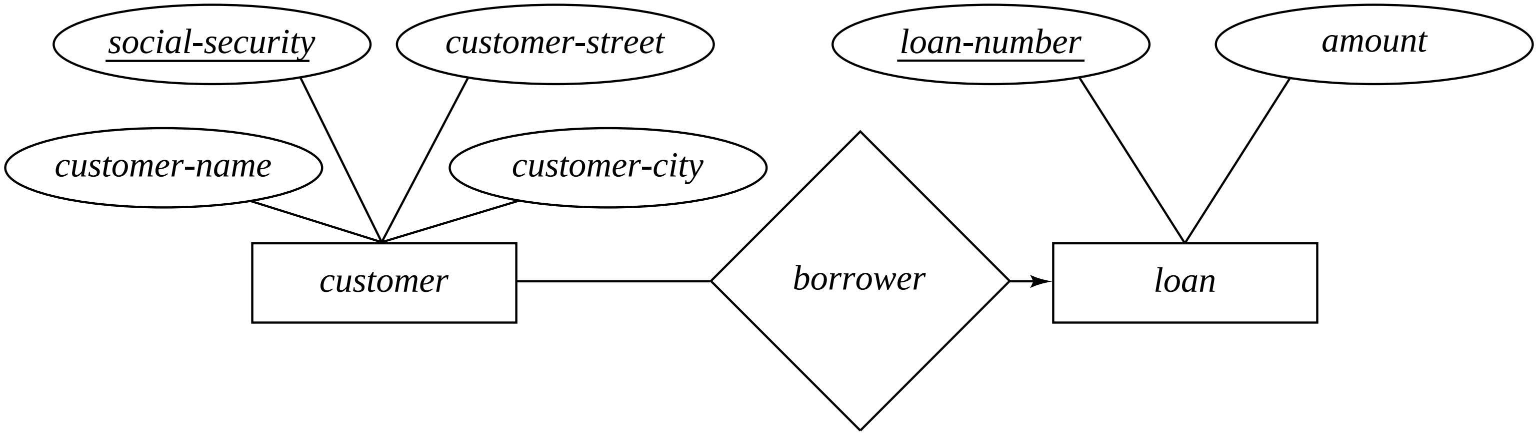


- A customer is associated with at most one loan via the relationship *borrower*
- A loan is associated with at most one customer via *borrower*

One-To-Many and Many-to-One Relationships



(a)



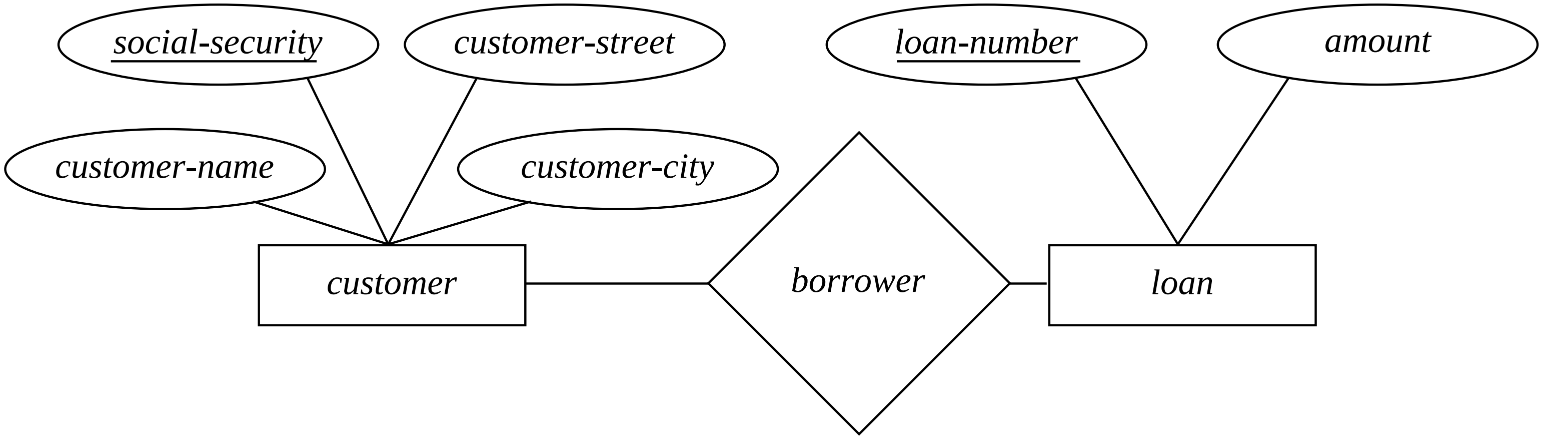
(b)

One-To-Many and Many-to-One (Cont.)

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

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

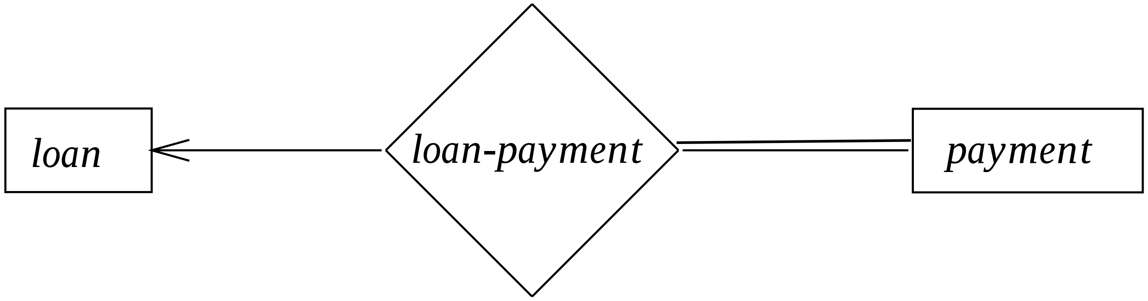
Many-To-Many Relationship



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

Existence Dependencies

- If the existence of entity *x* depends on the existence of entity *y*, then *x* is said to be *existence dependent* on *y*.
  - *y* is a *dominant entity* (in example below, *loan*)
  - *x* is a *subordinate entity* (in example below, *payment*)



- If a *loan* entity is deleted, then all its associated *payment* entities must be deleted also.

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
  - *social-security* 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*.
- The combination of primary keys of the participating entity sets forms a candidate key of a relationship set.
  - must consider the mapping cardinality and the semantics of the relationship set when selecting the *primary key*.
  - (*social-security, account-number*) is the primary key of *depositor*

E-R Diagram Components

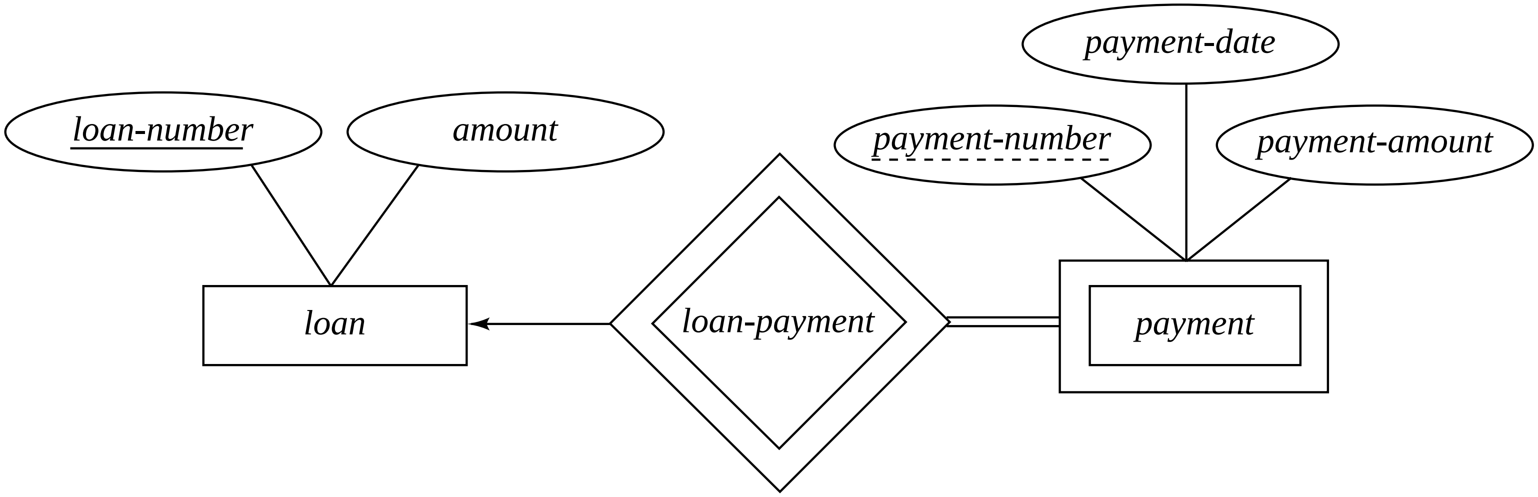
- **Rectangles** represent entity sets.
- **Ellipses** represent attributes.
- **Diamonds** represent relationship sets.
- **Lines** link attributes to entity sets and entity sets to relationship sets.
- **Double ellipses** represent multivalued attributes.
- **Dashed ellipses** denote derived attributes.
- Primary key attributes are underlined.

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 strong entity set; it must relate to the strong set via a one-to-many relationship set.
- 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.)

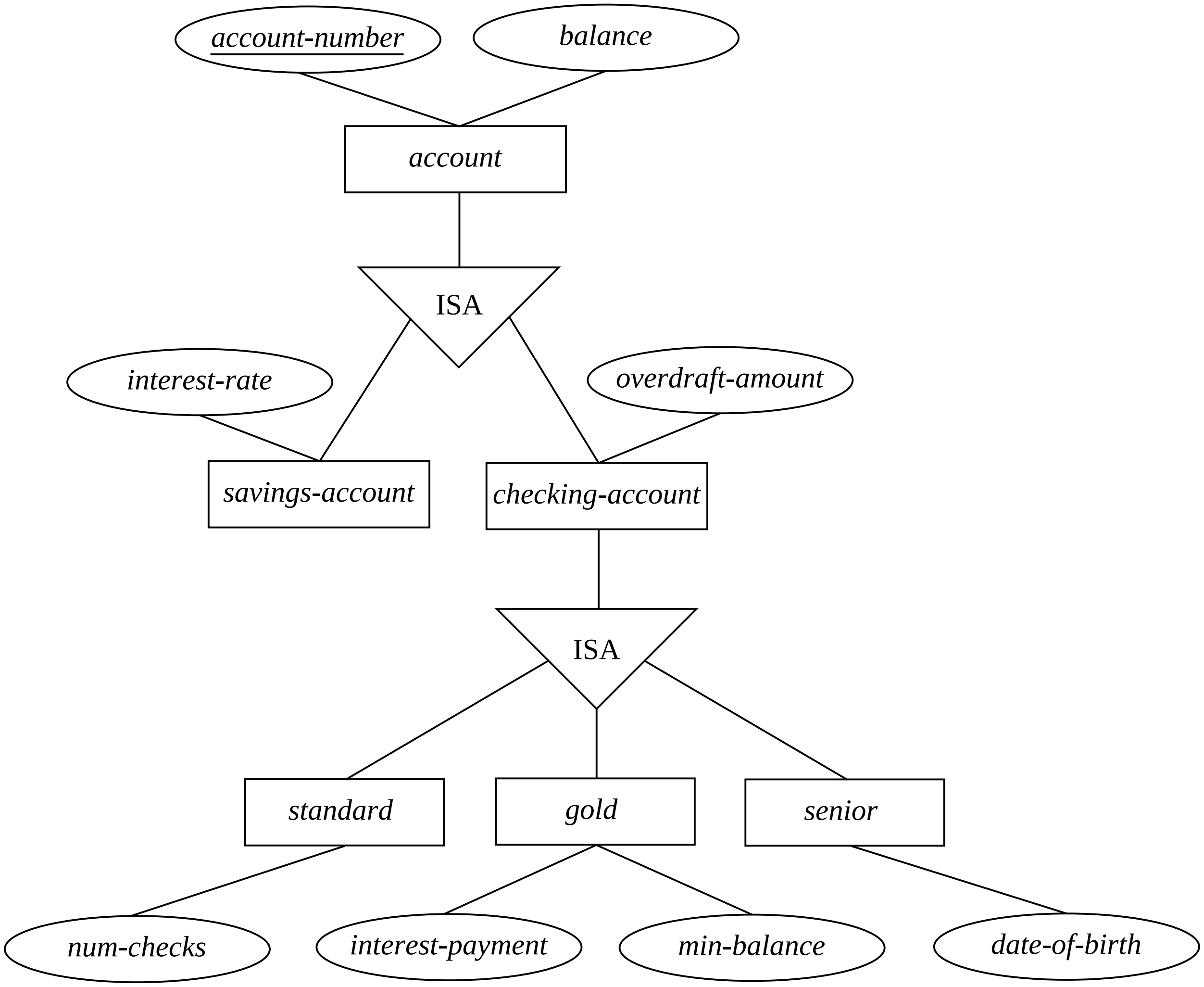
- We depict a weak entity set by double rectangles.
- We underline the discriminator of a weak entity set with a dashed line.
- *payment-number* – discriminator of the *payment* entity set
- Primary key for *payment* – (*loan-number, payment-number*)



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 (i.e., *savings-account* “is an” *account*)

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.

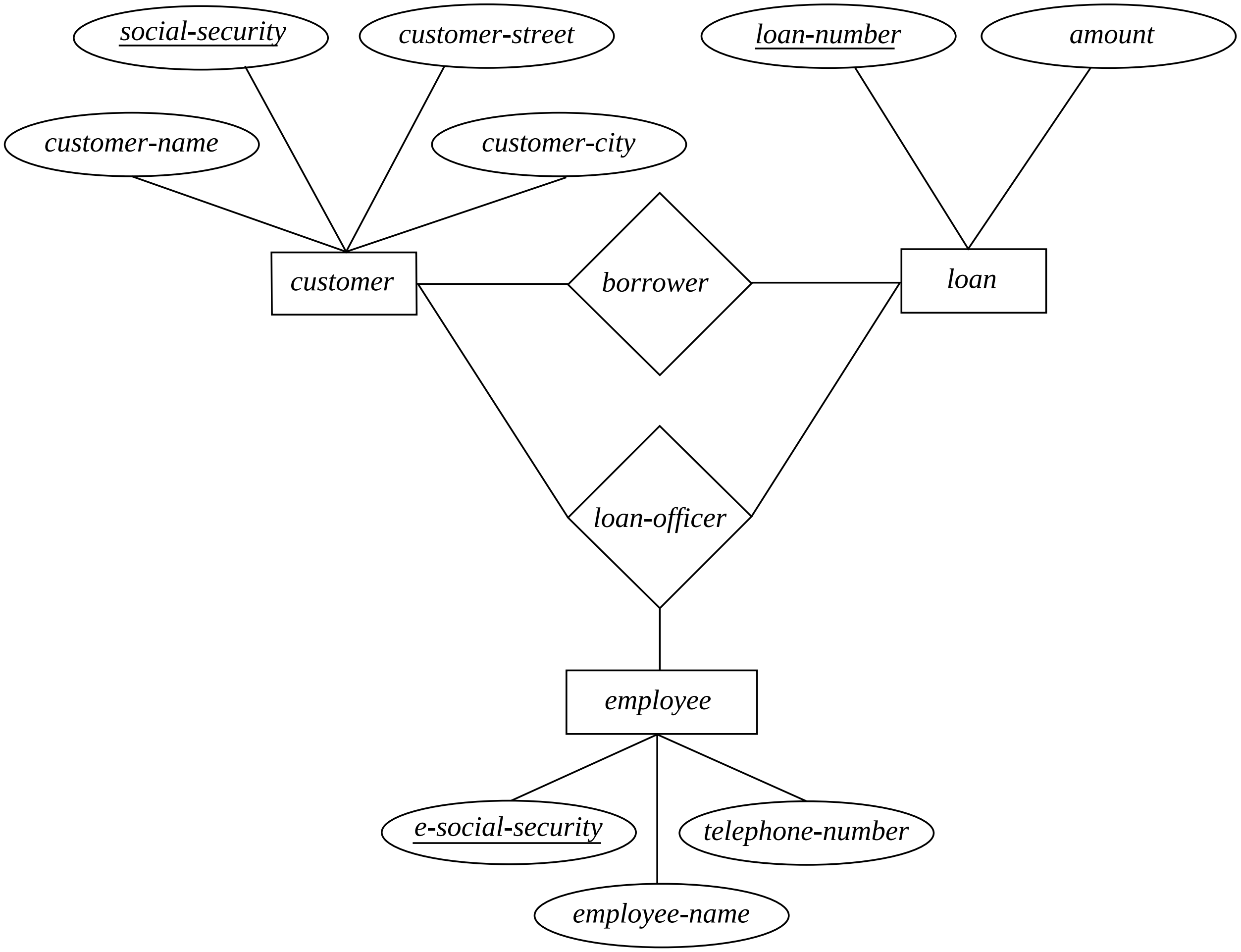
- **Attribute Inheritance** – a lower-level entity set inherits all the attributes and relationship participation of the higher-level entity set to which it is linked.

Design Constraints on a Generalization

- Constraint on which entities can be members of a given lower-level entity set.
  - condition-defined
  - user-defined
- Constraint on whether or not entities may belong to more than one lower-level entity set within a single generalization.
  - disjoint
  - overlapping
- 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
  - partial

Aggregation

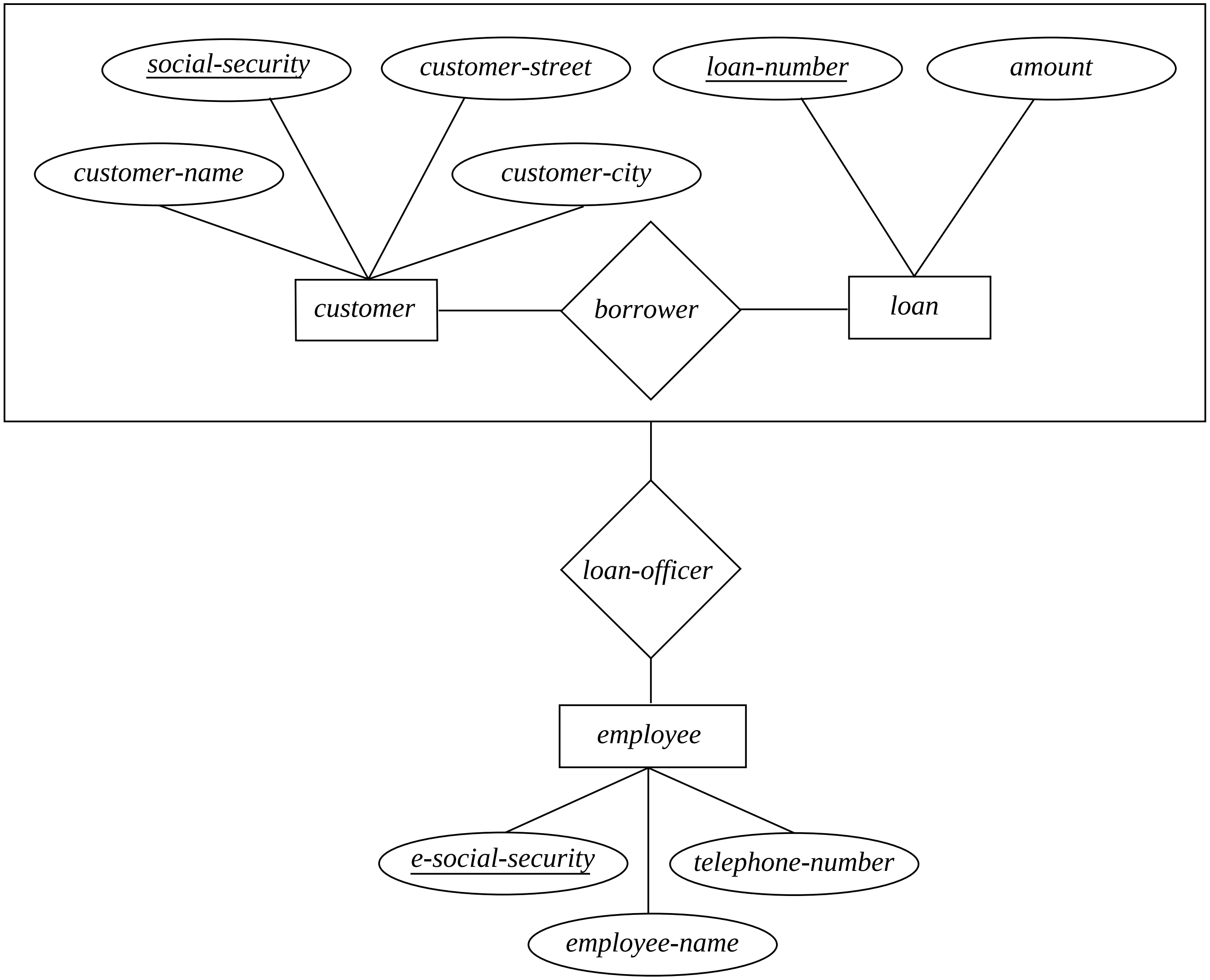
- Loan customers may be advised by a loan-officer.



Aggregation (Cont.)

- Relationship sets *borrower* and *loan-officer* represent the same information
- 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 that:
  - A customer takes out a loan
  - An employee may be a loan officer for a *customer-loan* pair

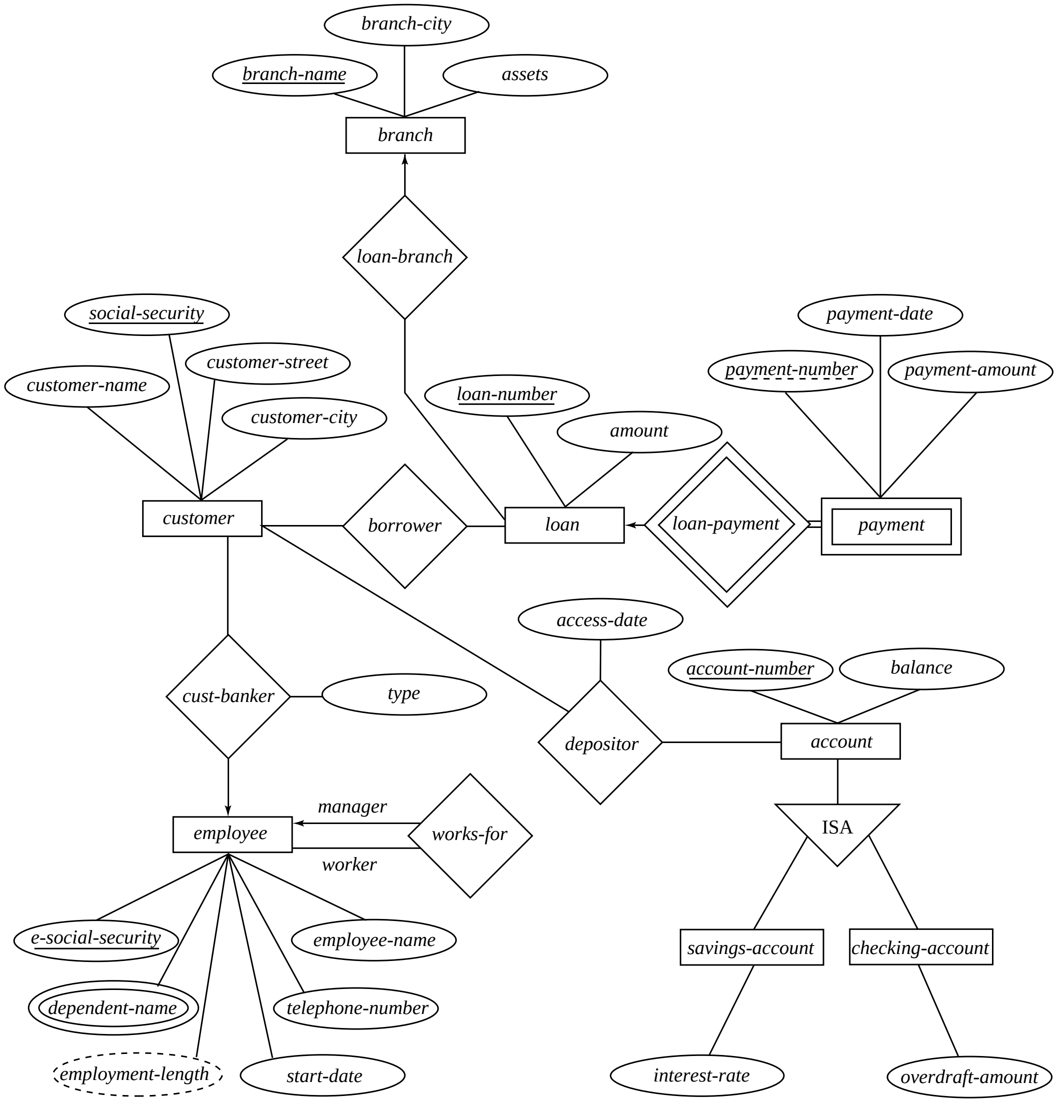
Aggregation Example



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 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 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.
- 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-name	*social-security*	*c-street*	*c-city*
Jones	321-12-3123	Main	Harrison
Smith	019-28-3746	North	Rye
Hayes	677-89-9011	Main	Harrison

The *customer* table

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

loan-number	*payment-number*	*payment-date*	*payment-amount*
L-17	5	10 May 1996	50
L-23	11	17 May 1996	75
L-15	22	23 May 1996	300

The *payment* table

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.

social-security	*account-number*	*access-date*
...	...	...

The *depositor* table

- The table corresponding to a relationship set linking a weak entity set to its identifying strong entity set is redundant.

  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 Generalization as Tables

- Method 1: Form a table for the generalized entity *account*

  Form a table for each entity set that is generalized (include primary key of generalized entity set)

table	table attributes
account	*account-number, balance, account-type*
savings-account	*account-number, interest-rate*
checking-account	*account-number, overdraft-amount*

- Method 2: Form a table for each entity set that is generalized

table	table attributes
savings-account	*account-number, balance, interest-rate*
checking-account	*account-number, balance, overdraft-amount*

Method 2 has no table for generalized entity *account*

Relations Corresponding to Aggregation

customer

customer-name	*cust-social-security*	*customer-street*	*customer-city*

loan

loan-number	*amount*

borrower

cust-social-security	*loan-number*

employee

emp-social-security	*employee-name*	*phone-number*

loan-officer

emp-social-security	*cust-social-security*	*loan-number*

Determining Keys from E-R Sets

- **Strong entity set.** The primary key of the entity set becomes the primary key of the relation.
- **Weak entity set.** The primary key of the relation consists of the union of the primary key of the strong entity set and the discriminator of the weak entity set.
- **Relationship set.** The union of the primary keys of the related entity sets becomes a super key of the relation.

  For binary many-to-many relationship sets, above super key is also the primary key.

  For binary many-to-one relationship sets, the primary key of the "many" entity set becomes the relation's primary key.

  For one-to-one relationship sets, the relation's primary key can be that of either entity set.