# Database Systems (CSF212)
# Lecture 11 - 12

BITS Pilani
Pilani Campus

innovate achieve lead

BITS Pilani
Pilani Campus

# E-R Diagrams

# Overview of Database Design

- Conceptual design
  - Use ER diagrams
    - Pictorial representation of DB schema
  - What are the *entities* and *relationships* in the enterprise?
    - E.g. customer & account entity; deposit relationship
  - What information about the entities and relationships should we store in DB?
  - What are the *integrity constraints* or *business rules* that hold?
- Logical design
  - Transform conceptual schema into implementation model
  - e.g., map an ER diagram into a relational schema
- Physical design and database tuning

# Entity-Relationship Model

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

# Entity and Entity Sets

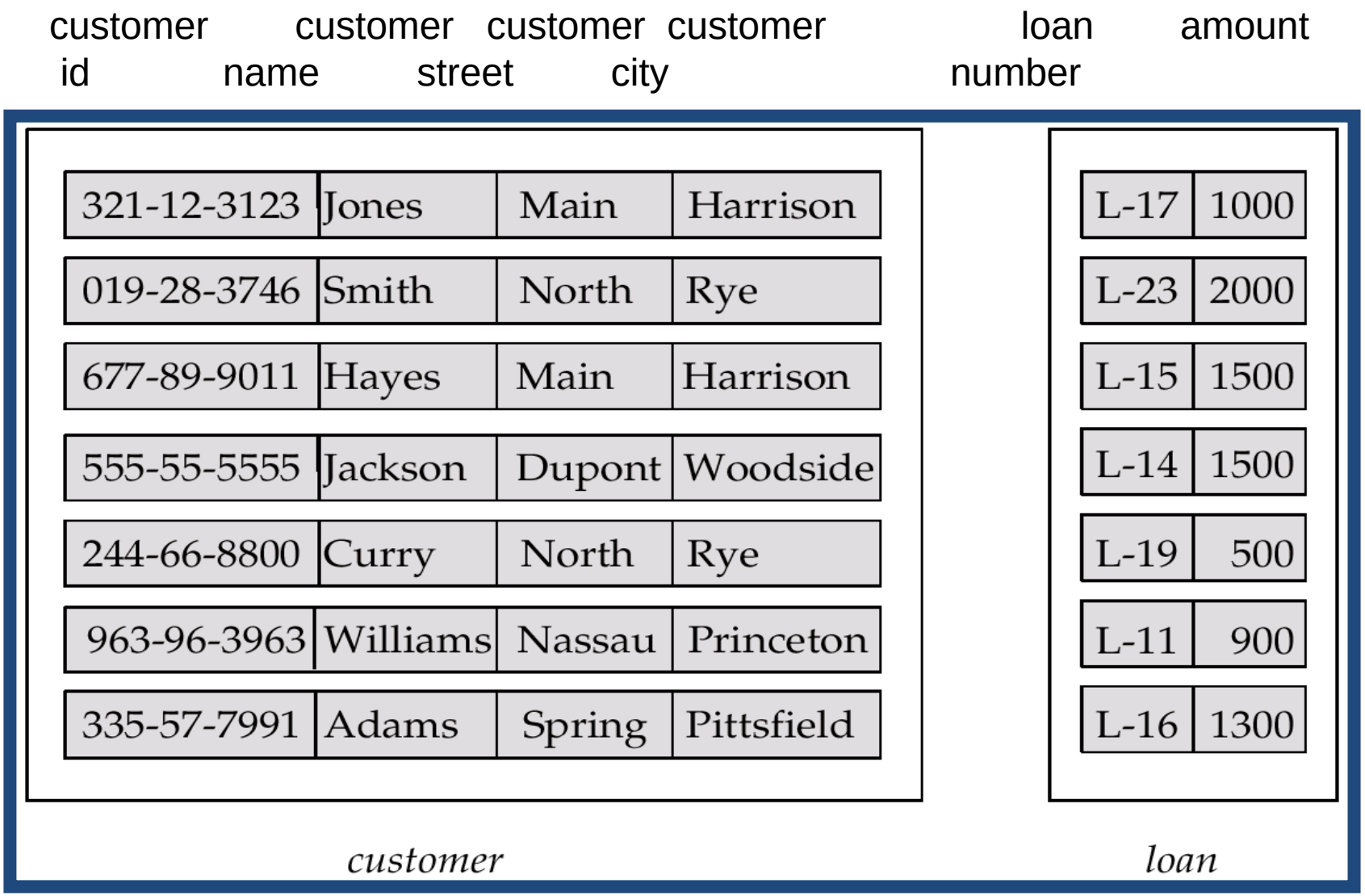
- A *database* can be modeled as:
  - a collection of entity sets each of which contains any number of entities,
  - relationship among entities.
- An *entity set* is a set of entities of the same type that share the same properties.
  - Example: set of all customers, accounts
- An *entity* is an object that exists and is distinguishable from other objects e.g., set of all customers of a bank, set of all account numbers
  - Are associated with a set of *attributes* e.g. *name, ID, dept_name* etc

# Entity and Entity Sets

- Attributes are associated with a set of permitted values known as *domain* of that attribute
- Every entity is described by a set of (attribute, value) pairs, one pair for each attribute of the entity set

If an entity set *customer* is associated with attributes name, ID, phone-no, city then {(name, John), (ID, 123456), (phone-no, 9999988888), (city, New york)} describes a particular entity in the set

# Entity Sets *customer* and *loan*



# Attributes

- Attribute types:
  - ***Simple*:** Atomic valued attribute which cannot be divided further. E.g., phone-no
  - ***Composite:*** Made up of more then one atomic valued attribute. E.g., Name attribute having first name and last name
  - ***Single-valued:*** Attribute that can have only a single value
    - they can be either simple or composite

# Attributes

- ***Multi-valued:*** Attributes that can have many values E.g. multi-valued attribute: *phone-numbers*
- ***Derived:*** Can be computed from other attributes
    - Are not stored in the database
    - E.g. *age*, given date of birth

# Relationship Sets

see book, to improve understanding. A relationship set, is basically a mapping between various Tables.

- Relationship: Association among several entities

  Example:

  Hayes — *depositor* — A-102

  ***customer* entity** — relationship set — ***account* entity**

- A *relationship* set is a mathematical relation among entities $\{(e_1, e_2, \ldots e_n) \mid e_1 \in E_1, e_2 \in E_2, \ldots, e_n \in E_n\}$ where $(e_1, e_2, \ldots, e_n)$ is a relationship
  - Example:

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

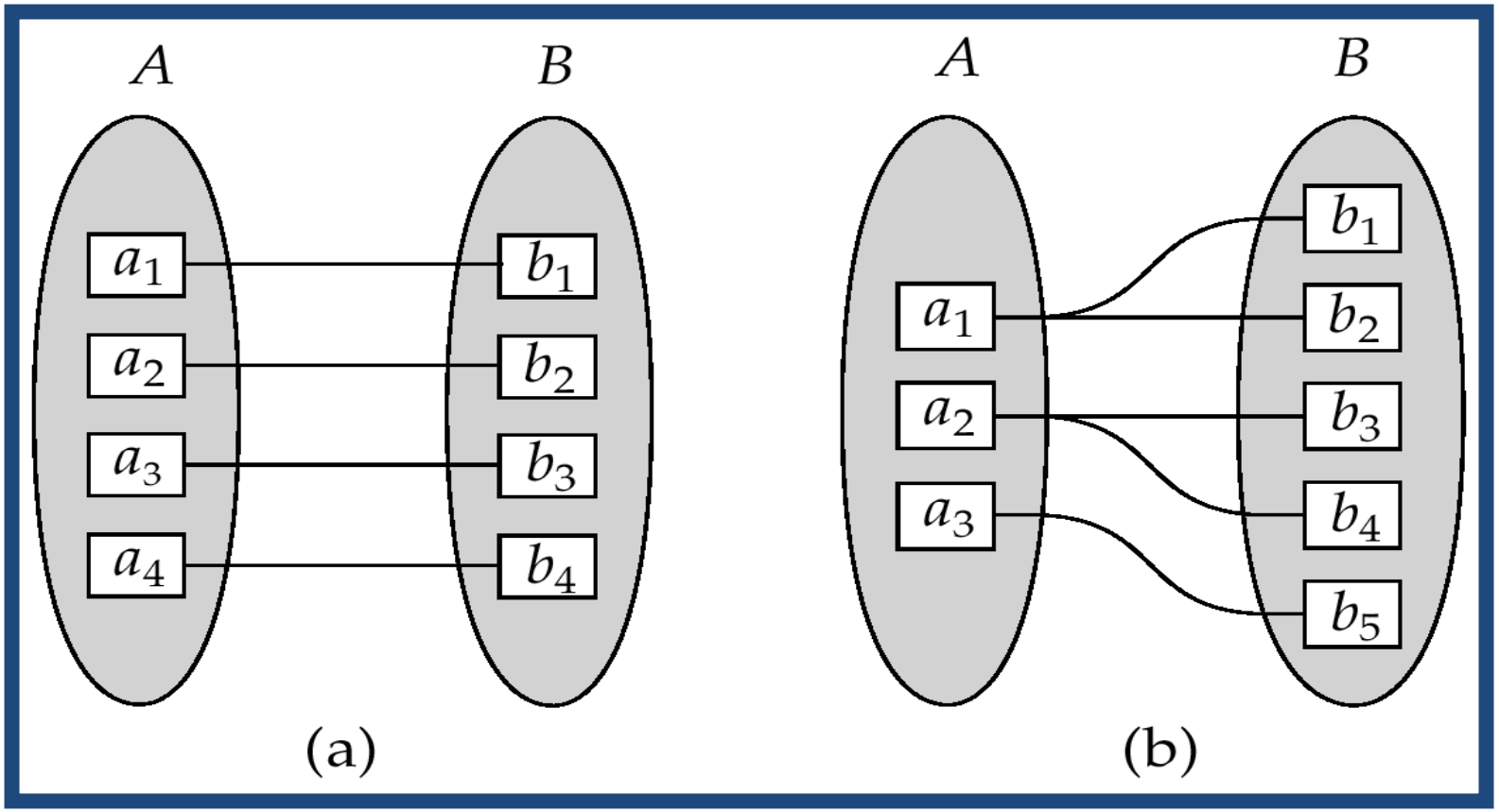
# Degree of a Relationship Set

- Refers to number of entity sets that participate in a relationship set.
- Binary Relationship sets
  - Involve two entity sets
  - Most relationship sets in DB
- N-ary Relationships (N >=1)
  - Can be converted to binary relations
    - E.g. Two binary relationships, mother and father, relating a child to her farther and mother vs. one ternary relationship parent

# Constraints: Mapping Cardinalities

- Express the number of entities to which another entity can be associated via a relationship set
- Represented by drawing either a directed line (→), signifying "one," or an undirected line (—), signifying "many," between the relationship set and the entity set
- For a binary relationship set the mapping cardinality must be one of the following types:
  - One to One
  - Many to One
  - Many to Many

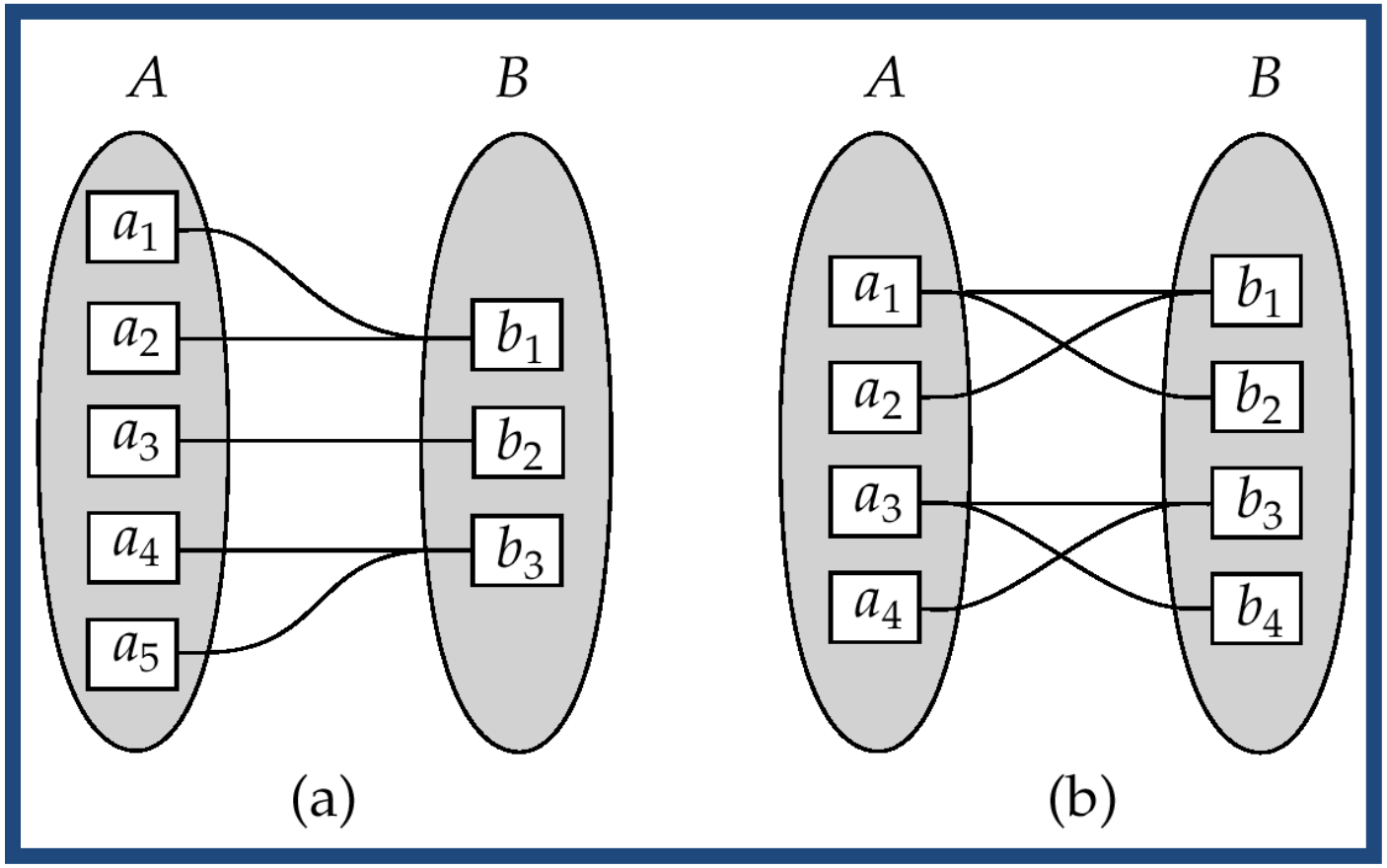
# Mapping Cardinalities




One to one

One to many

# Mapping Cardinalities



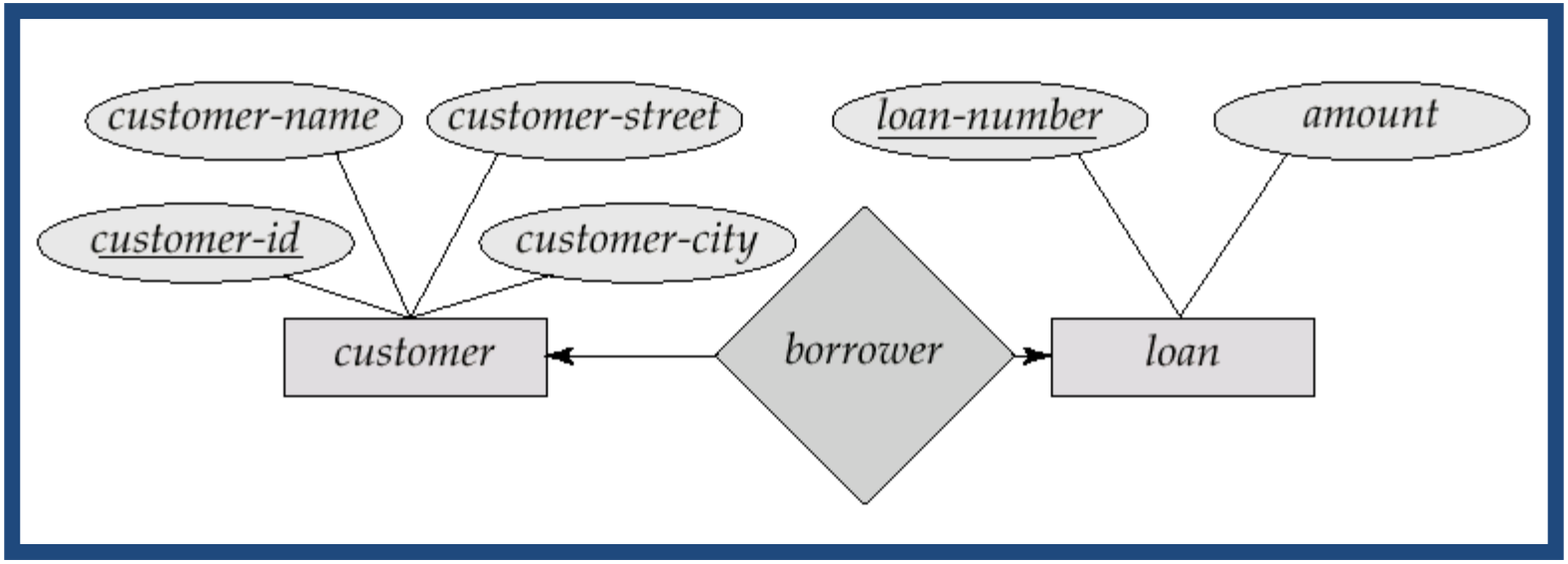

Many to One

Many to Many

# One to One Cardinality Constraint

Example

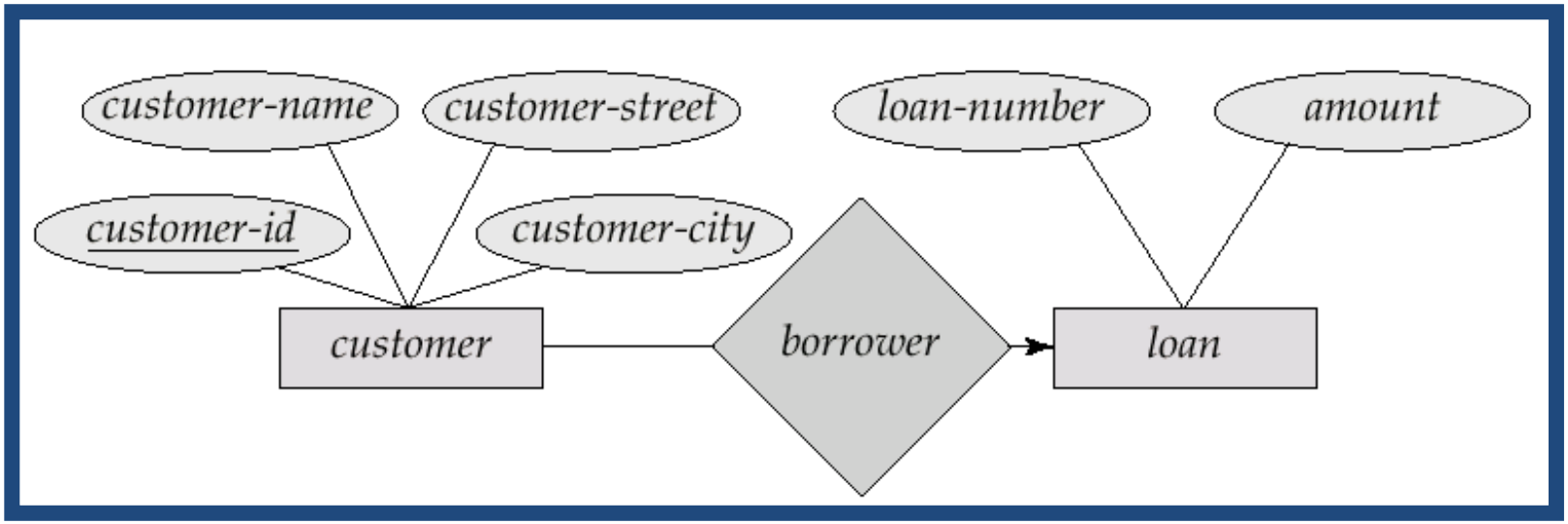
- A customer is associated with at most one loan via the relationship *borrower*
- A loan is associated with at most one customer via *borrower*



# Many to One Cardinality Constraint

Example

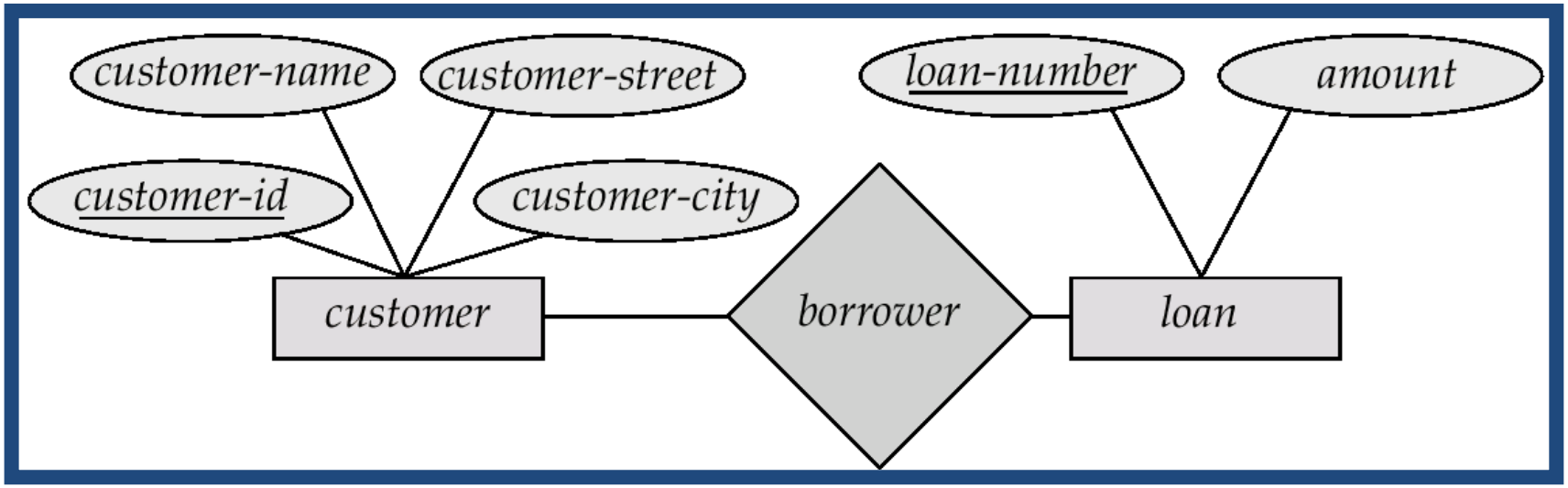
- In a many-to-one relationship a loan is associated with several (including 0) customers via *borrower*,
- A customer is associated with at most one loan via *borrower*



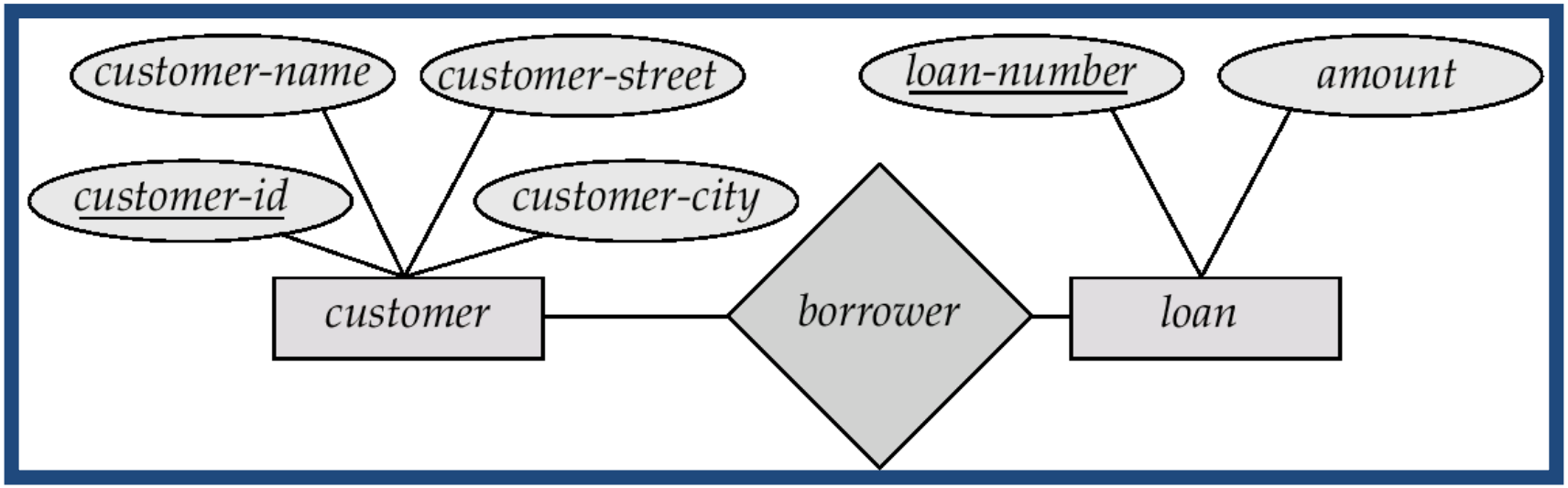
# Many to Many Cardinality Constraint

Example

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

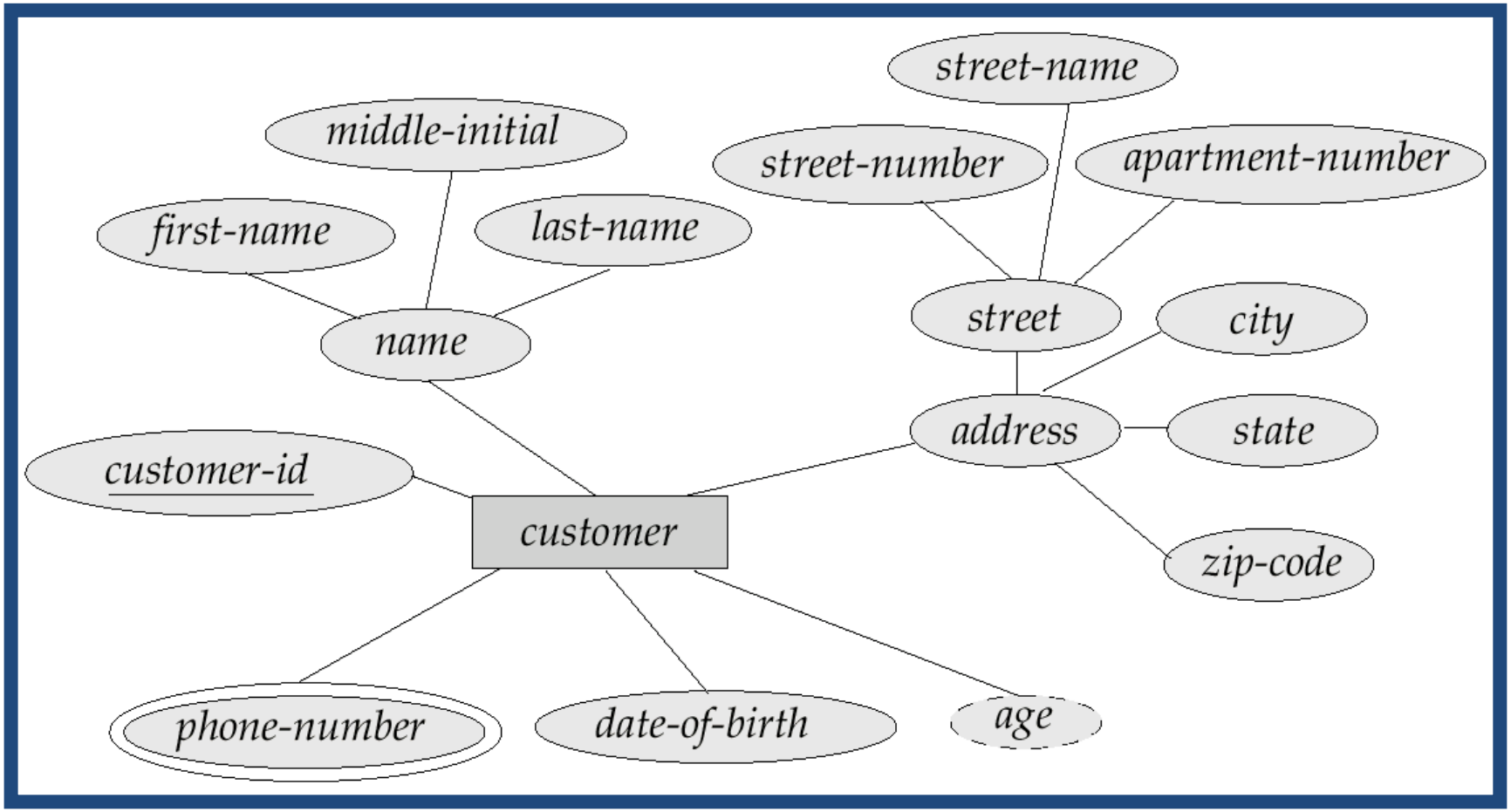


# E-R Diagram

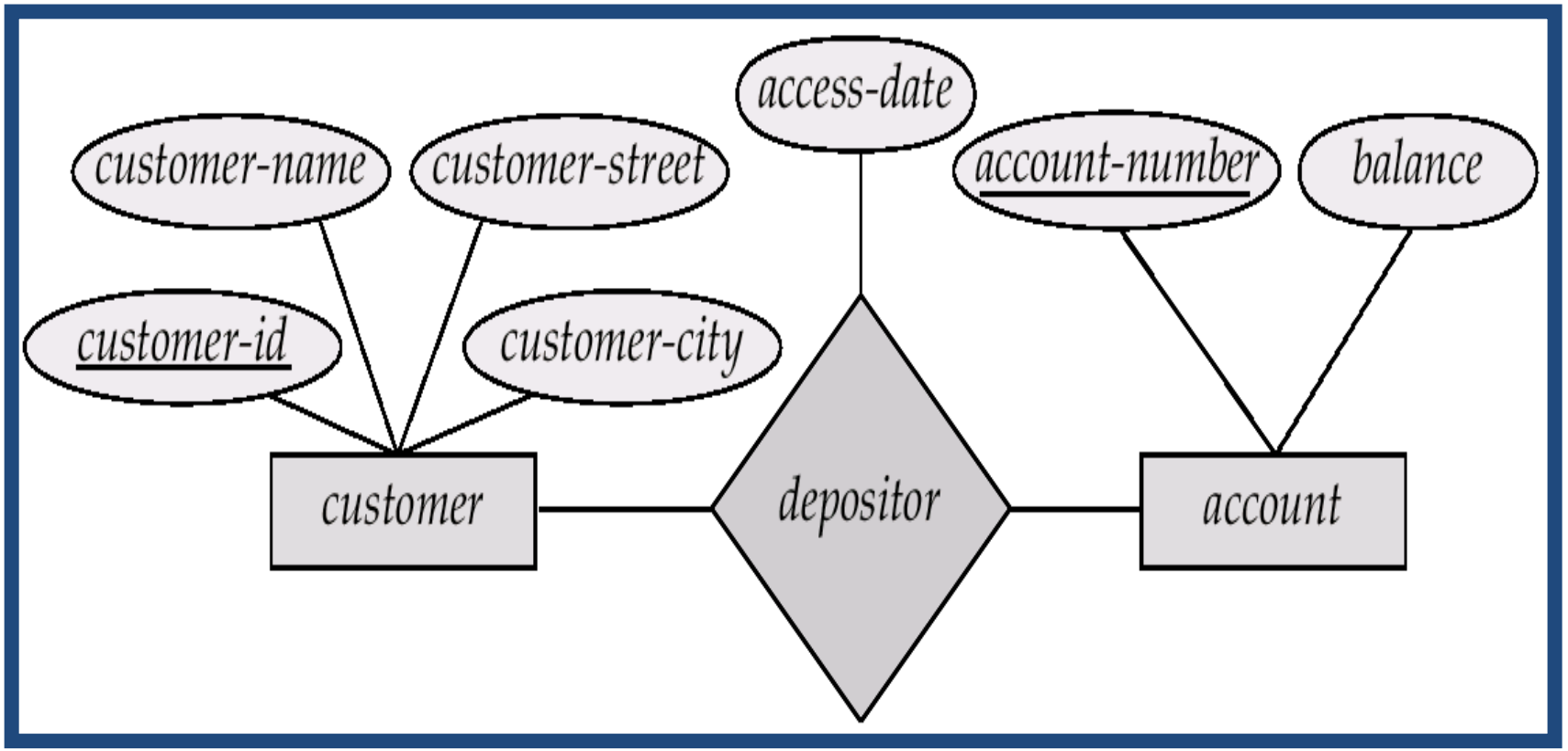



- **Rectangles** represent entity sets.
- **Diamonds** represent relationship sets.
- **Lines** link attributes to entity sets and entity sets to relationship sets.
- **Ellipses** represent attributes
  - **Double ellipses** represent multivalued attributes.
  - **Dashed ellipses** denote derived attributes.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

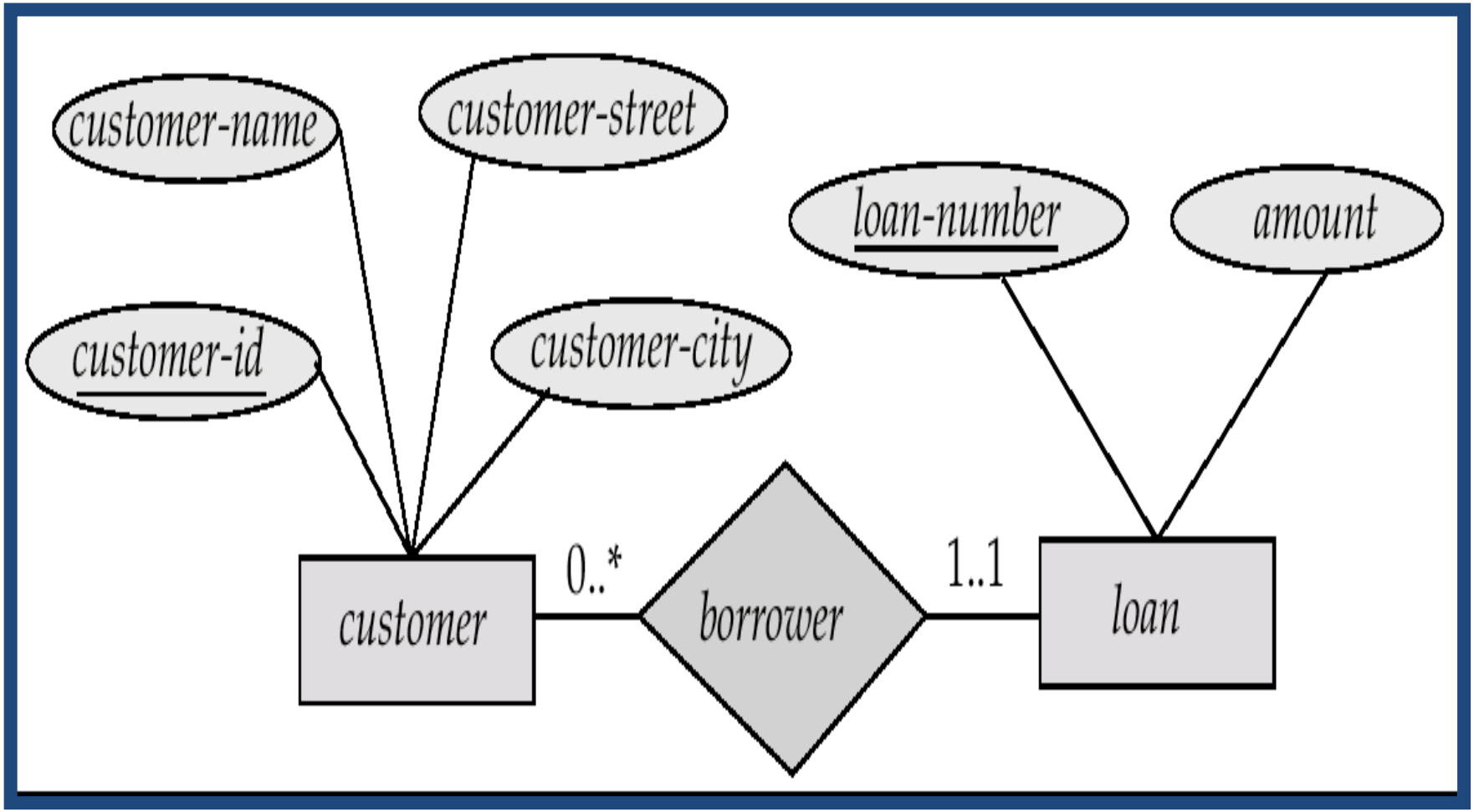
# E-R Diagram With Composite, Multi-valued, and Derived Attributes



# Relationship Sets with Attributes



# Alternative Notation for Cardinality Limits



# Existence dependency constraints

- Existence-dependent: Existence of entity x depends on the existence of entity y
  - y is said to be ***dominant entity*** and x is said to be ***subordinate entity***
  - if *y* is deleted, so is *x.* Vice-versa is not mandatory
  - e.g., ***account*** and ***transaction***.

# Keys

# Keys in Entity Sets

- A ***super key*** of an entity set
  - A set of one or more attributes whose values uniquely determine each entity
  - E.g. *customer_id* & *customer_name*

- A ***candidate key***
  - A minimal super key
  - Customer id is candidate key of customer
  - *Account-number* is candidate key of account

# Keys in Entity Sets

- A ***super key*** of an entity set
  - A set of one or more attributes whose values uniquely determine each entity
- A ***candidate key***
  - A minimal super key
  - Customer id is candidate key of customer

**E**mployee(ID, name, res_phone_no).

super key = {{ID}, {ID, name}, {ID, res_phone_no}, {name, res_phone_no}, {ID, name, res_phone_no}}

candidate key = {{ID}, {name, res_phone_no}}

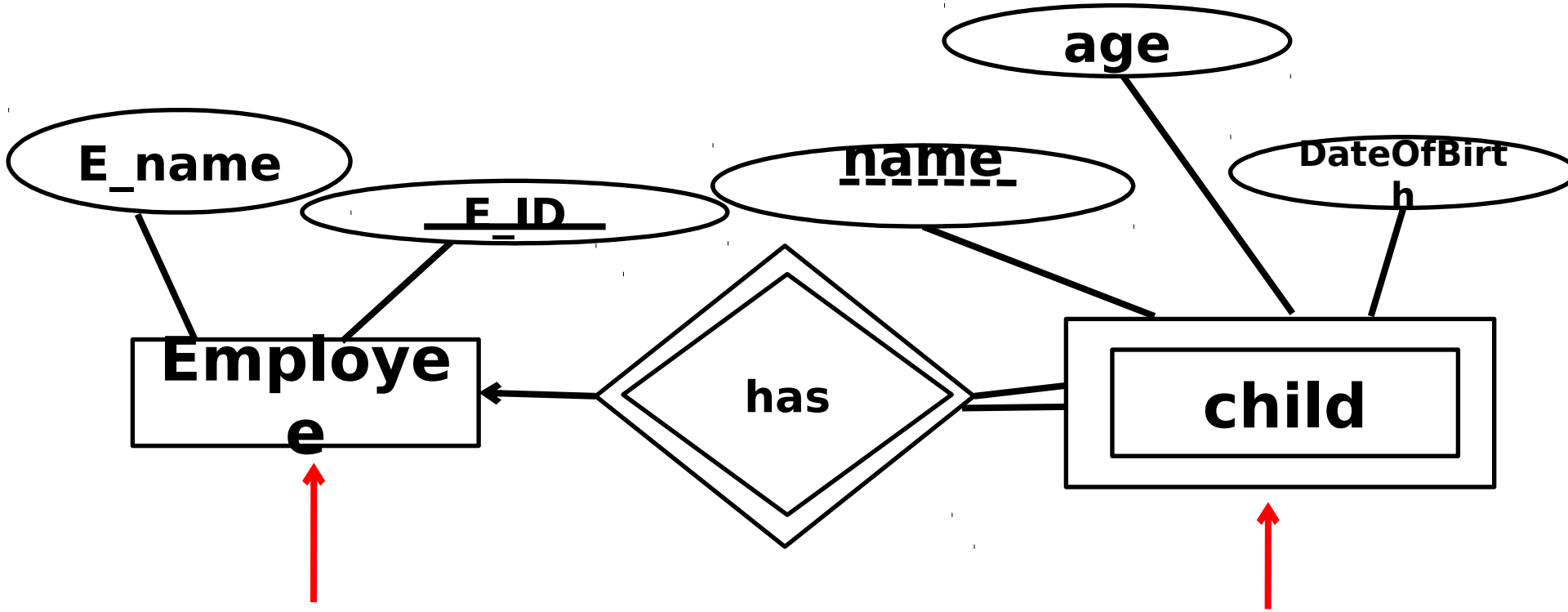
# Keys in Entity Sets

- Primary key: Candidate key chosen to uniquely identify a record in a relation
  - weak entity set: Entity set does not have sufficient attributes to form a primary key
  - strong entity set: Entity set having a primary key

  - E.g., consider the transaction entity set having three attributes: ***transaction-number***, ***date*** **and** ***amount***

# Weak Entity Sets

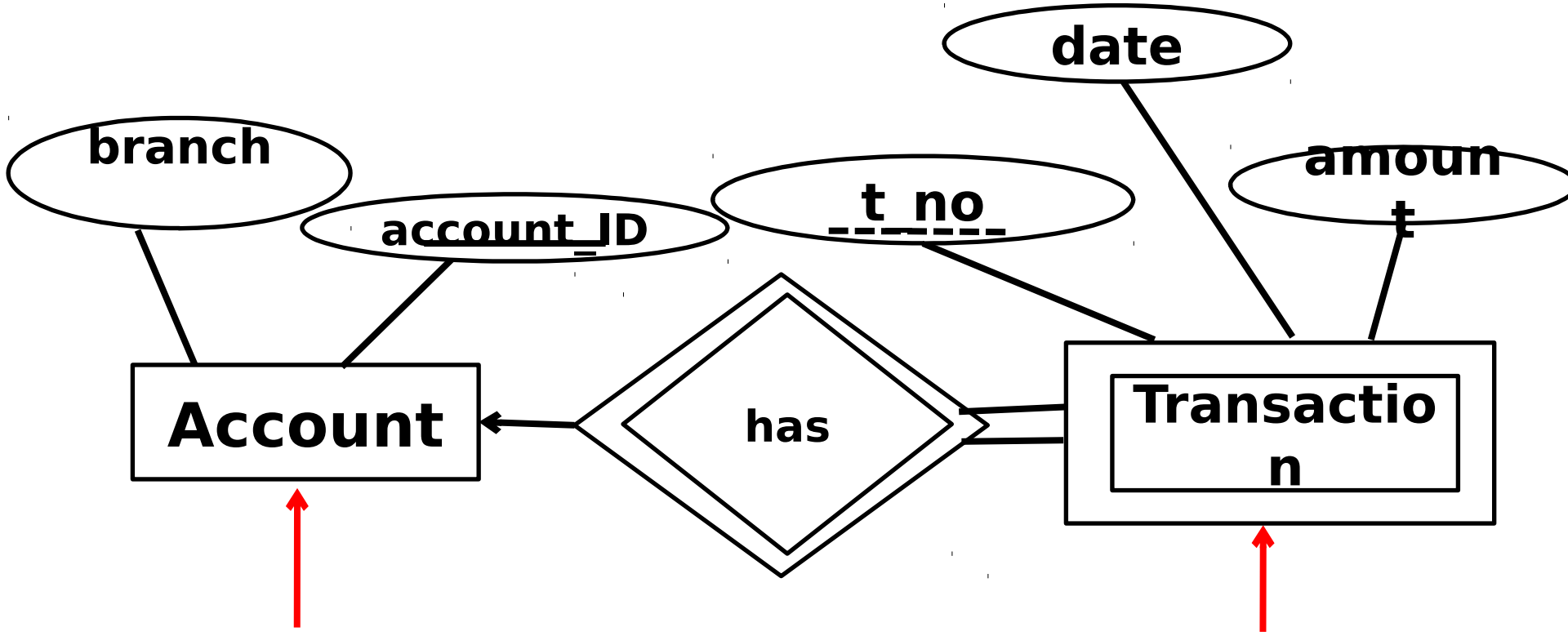
- An entity set that **does not have a primary key**
- The existence of a weak entity set depends on the existence of a *identifying entity set*
  - It must relate to the identifying entity set via a **total**, **one-to-many** relationship set from the identifying to the weak entity set
  - Identifying relationship depicted using a double diamond
- Di*scriminator (or partial key):* May not uniquely identify entities of a weak entity set, but can be combined with the primary key of a strong entity set to uniquely identify entities of the weak entity set
- The primary key of a weak entity set
  - primary key of the strong entity set + weak entity set's discriminator.




nary Key(Employee) = {E_ID} // represented by solid l

Partial Key(child) = {name} // Represented by dashed lines. Two employees can have child of same name, age and DateOfBirth

rimary Key(child) = {E_ID, name}




imary Key(Account) = {account_ID}

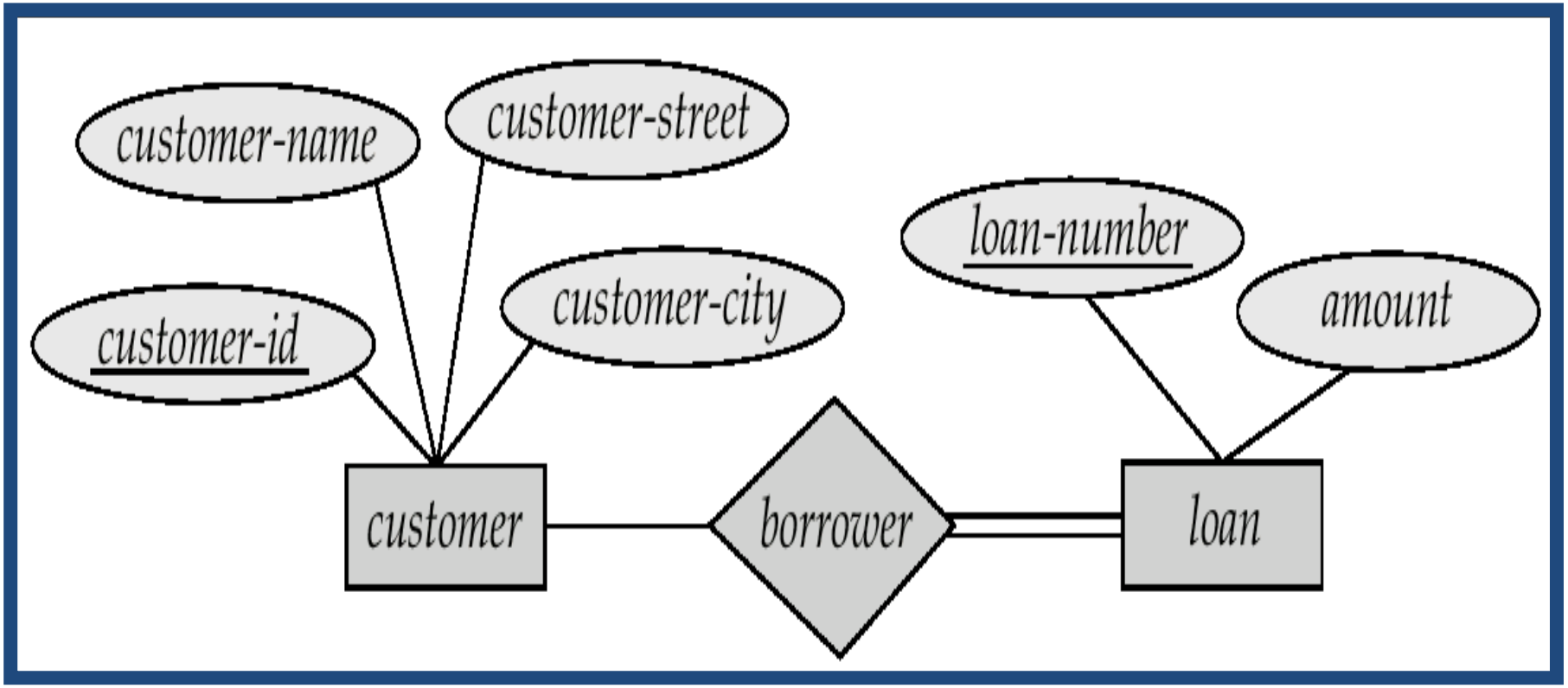
Partial Key(Transaction) = {t_no} // Although t_no can uniquely identify the records in transaction entity set, transaction entity set does not have any existence without account entity set.

Primary Key(Transaction) = {account_ID, t_no}

# 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 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*

# Participation of an Entity Set in a Relationship Set




Every entity in *loan* should be associated with some entity in *customer* through *borrower* relationship, but vice-versa may not be *true*.

# Keys for Relationship Sets

- Let R be a relationship set involving entity sets E1, E2, … En.
- If R has **no attribute** of it's own then
  - attribute(R) = Primary-key(E1) U primary-key(E2) … U primary-key(En)
- If R has it's **own attribute set** {$a_1$, $a_2$,..$a_m$}
  - attribute(R) = Primary-key(E1) U primary-key(E2)

# Keys for Relationship Sets

- Primary key of relationship sets depends on:
  - Mapping cardinality
  - Structure of the attributes associated with the relationship set
- If the relationship set R has **no attribute**:
  - Superkey(R) = attribute(R)
  - If mapping cardinality is many-to-many, then Primarykey(R) = Superkey(R)
  - If mapping cardinality is many-to-one, then the primary key of the entity associated with *many* is

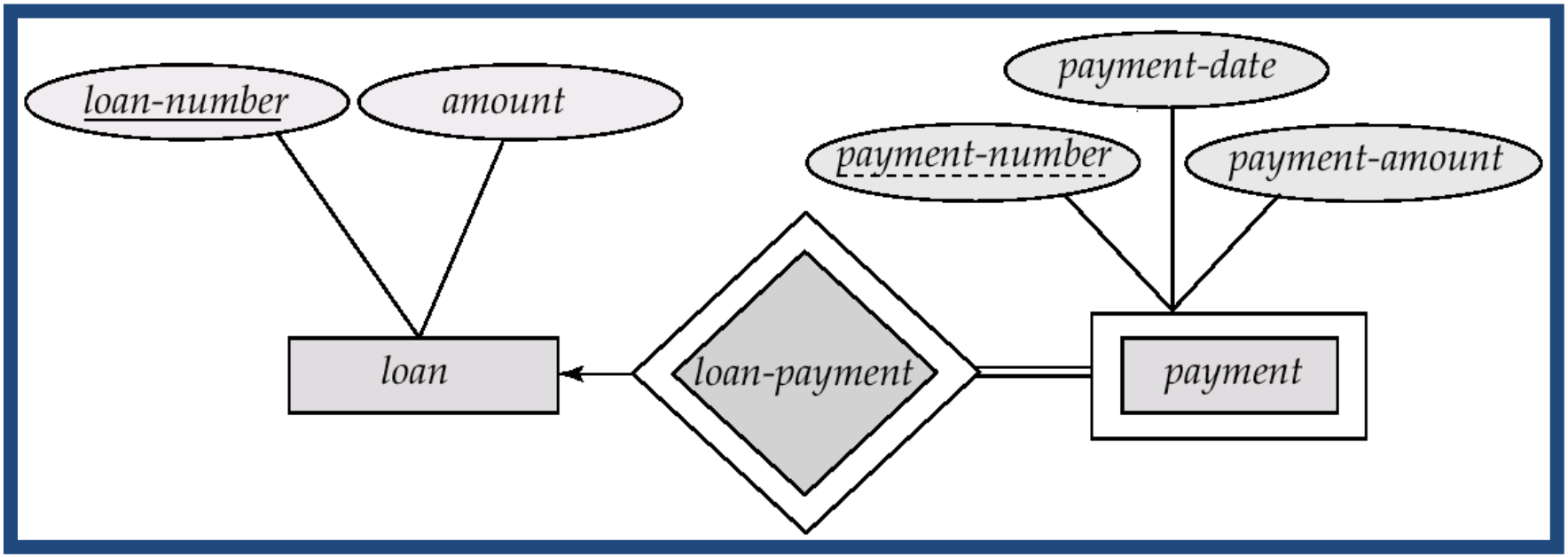
# Keys for Relationship Sets

- If the relationship set R has **it's own set of attributes** then super key is formed as before, with the possible addition of one or more of these attributes

# ER Diagrams to Relational Schema

# Representation of Strong Entity sets

- Let E be a strong entity set with attributes $a_1$, $a_2$, ... $a_n$ then:
  - E is represented by a table named as E with n distinct columns
  - Each of row of E represents one entity of the set



# Representation of Strong Entity sets

| **loan** | |
|---|---|
| Loan_number | Amount |

oan = {(L_1, 1000), (L_2, 2000), (L_3, 3000)}

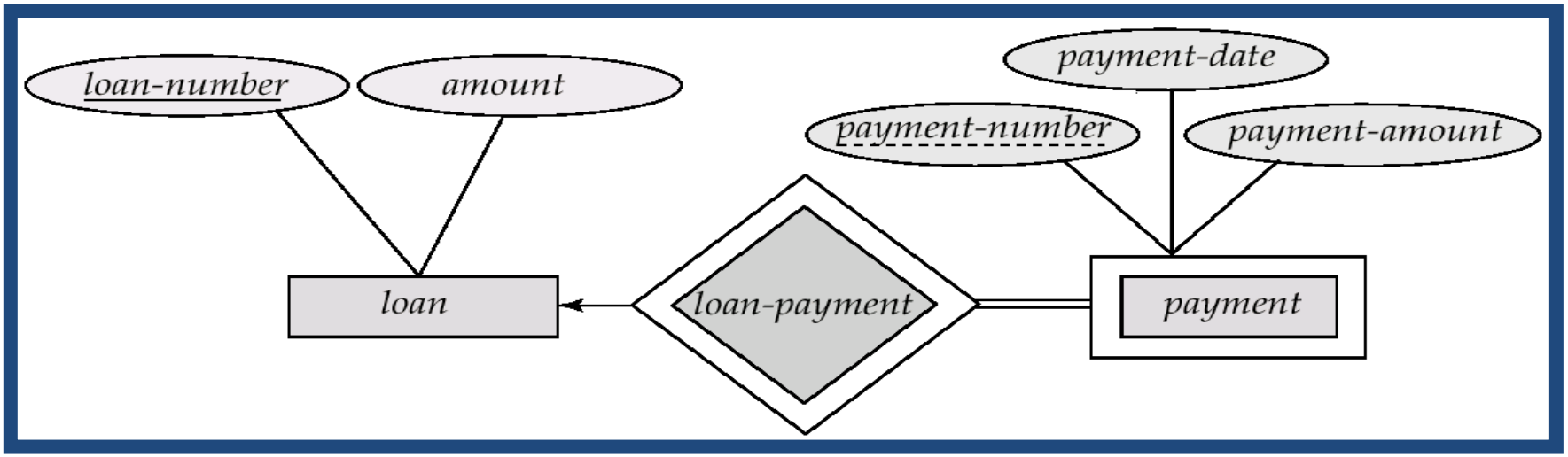
| **loan** | |
|---|---|
| loan_number | Amount |
| L_1 | 1000 |
| L_2 | 2000 |
| L_3 | 3000 |

# Representation of Weak Entity sets

- Let A be a weak entity set with attributes $a_1, a_2, \ldots a_n$ and B be the strong entity set on which A is dependent. Let the primary key of B be $\{b_1, b_2, \ldots b_m\}$, then:
  - A is represented by a table named as A with one column for each attribute of the set

    $\{a_1, a_2, \ldots a_n\} \cup \{b_1, b_2, \ldots b_m\}$
  - Each of row of E represents one entity of the set

# Representation of Weak Entity sets



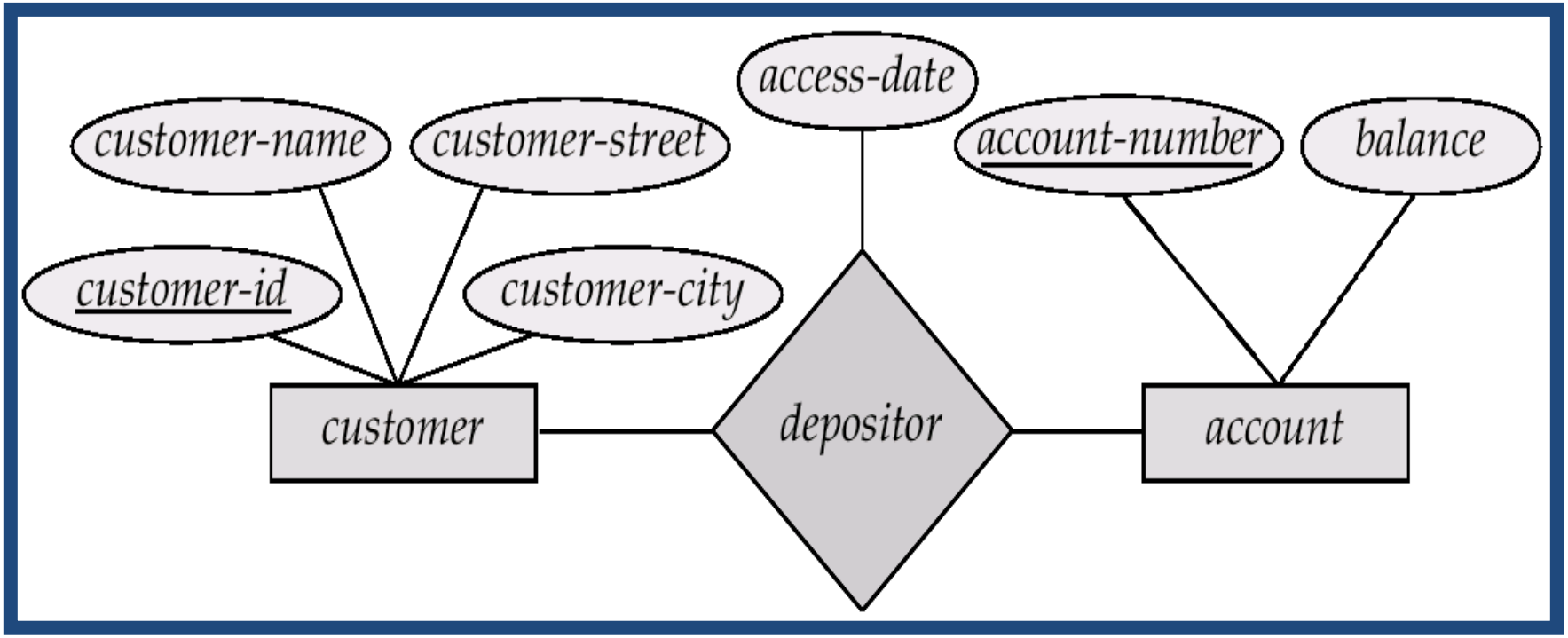

| **payment** | | | |
|---|---|---|---|
| payment_number | payment_date | payment_amount | loan_number |

# Representation of Relationship Sets

- Let R be a relationship set involving entity sets $E_1$, $E_2$, ... $E_m$. Let attribute(R) consists of *n* attributes. Then:
  - R is represented by a table called R with *n* distinct columns

# Representation of Relationship Sets associating Strong Entity Sets

you may have to depict primary key of the relationship set, when you do this.

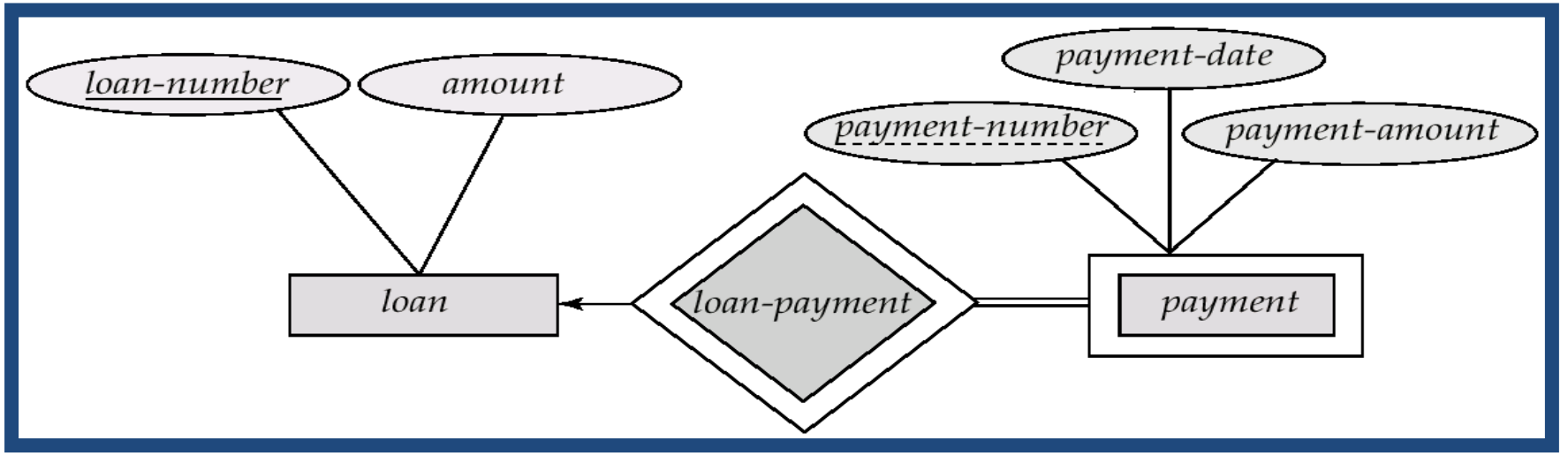



attribute(depositor) = {customer_id, account_number, access_date}

| **depositor** | | |
|---|---|---|
| customer_id | account_number | access_date |

# Representation of Relationship Sets

## Associating Strong and Weak Entity Sets




| payment | | | |
|---|---|---|---|
| payment_number | payment_date | payment_amount | loan_number |

**attribute(loan-payment) = {loan_number, payment_number**

| loan-payment | |
|---|---|
| loan_number | payment_number |

**Redundant information, already present in payment table. Therefore no**