Lecture 5
Entity-Relationship Model(实体-联系模型)

March 26, 2014

Shuigeng Zhou
School of Computer Science
Fudan University

Outline

- Introduction to ER Model
- Entity Sets (实体集)
- Relationship Sets (联系集)
- Design Issues
- Mapping Constraints (映射约束)
- Keys (键)
- E-R Diagram (E-R图)
- Design of an E-R Database Schema
- Reduction of an E-R Schema to Tables

Database Design(数据库设计)

- Conceptual design（概念设计）
  - Mapping a real world organization to a conceptual model
- Logical design（逻辑设计）
  - Transforming the conceptual model to a logical model (database model)
- Physical design（物理设计）
  - Instantiating the logical model to physical organization and storage

Database Design (cont'd)

- Understand the real-world domain being modeled
- Specify it using a database design model
  - Design models are especially convenient for schema design, but are not necessarily implemented by DBMS
  - Popular ones include
    - Entity/Relationship (E/R) model
    - Object Definition Language (ODL)
- Translate specification to the data model of DBMS
  - Relational, XML, object-oriented, etc.
- Create DBMS schema

Database Conceptual Design

- *Conceptual design* (*ER Model is used at this stage*)
  - What are the *entities* and *relationships* in the enterprise?
  - What information about these entities and relationships should we store in the database?
  - What are the *integrity constraints* or *business rules* that hold?
  - A database `schema' in the ER Model can be represented pictorially (*ER diagrams*)
  - Can map an ER diagram into a relational schema

ER Model: A General View

- Historically very popular
- Can think of as a "watered-down" object-oriented
- Design model
- E/R diagrams represent designs
- Primarily a design model—not implemented by any major DBMS

Peter Pin-Shan Chen（陈品山）

- Dr. Peter P. Chen is the originator of the Entity-Relationship Model (ER Model), and the founder of ER international conference
- The ER Model serves as the foundation of many systems analysis and design methodologies, computer-aided software engineering (CASE) tools, and repository systems

Peter Chen, *The Entity-Relationship Model--Toward a Unified View of Data* *ACM Transactions on Database Systems, Vol. 1, No. 1, March 1976, Pages 9 - 36*

What is Data Model ?

- A collection of conceptual tools for describing
  - Data
  - Data relationships
  - Data semantics
  - Consistency constraints

Entity Sets

- A database can be modeled as (**ER Model**):
  - 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, holidays

Entity Sets customer and loan

customer-id	customer-name	customer-street	customer-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 number	amount
L-17	1000
L-23	2000
L-15	1500
L-14	1500
L-19	500
L-11	900
L-16	1300

loan

Attributes（属性）

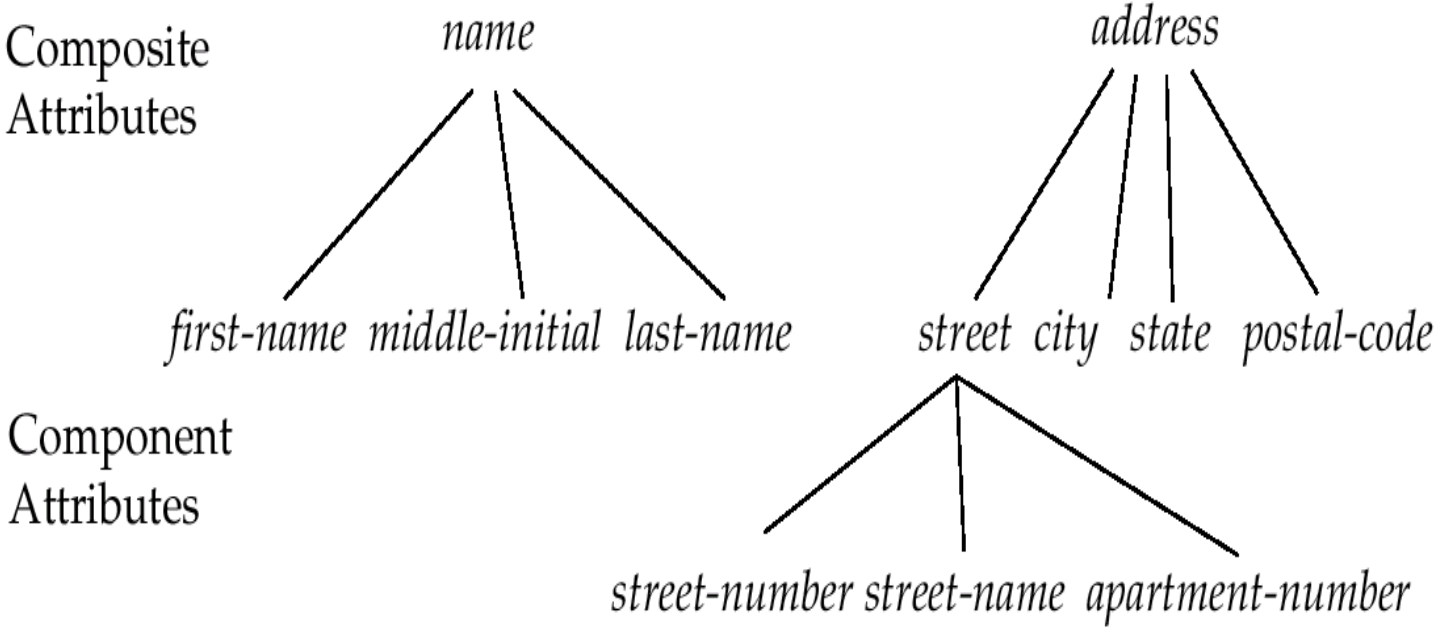
- An entity is represented by a set of attributes, that is descriptive properties possessed by all members of an entity set

  Example:

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

- Domain （域）- the set of permitted values for each attribute
- Attribute types（属性类型）
  - Simple and composite attributes.
  - Single-valued and multi-valued attributes
  - Derived attributes

Composite Attributes（复合属性）



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

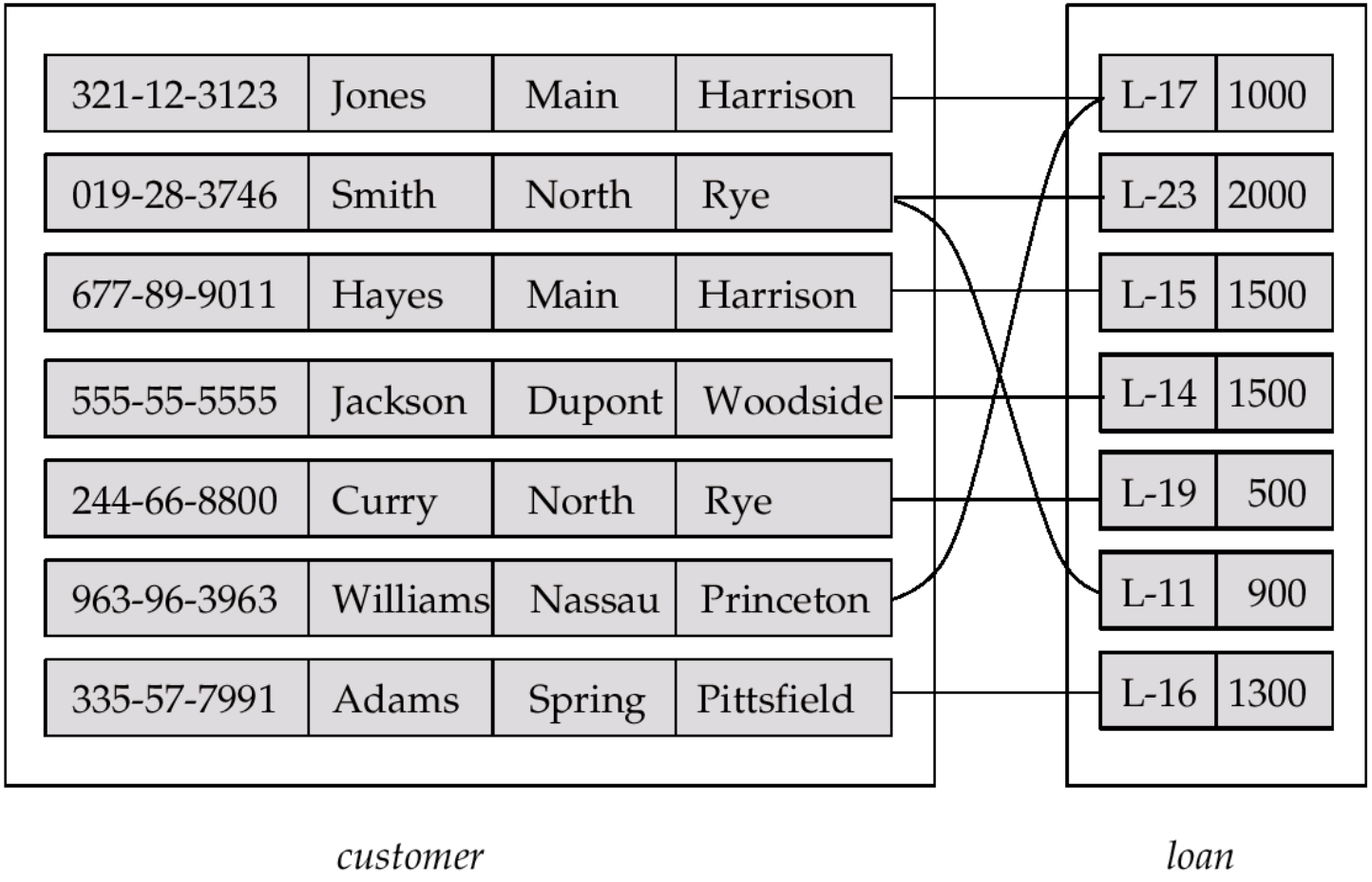
  $$\{(e_1, e_2, \ldots e_n) \mid e_1 \in E_1, e_2 \in E_2, \ldots, e_n \in E_n\}$$

  where $(e_1, e_2, \ldots, e_n)$ is a relationship

  - Example:

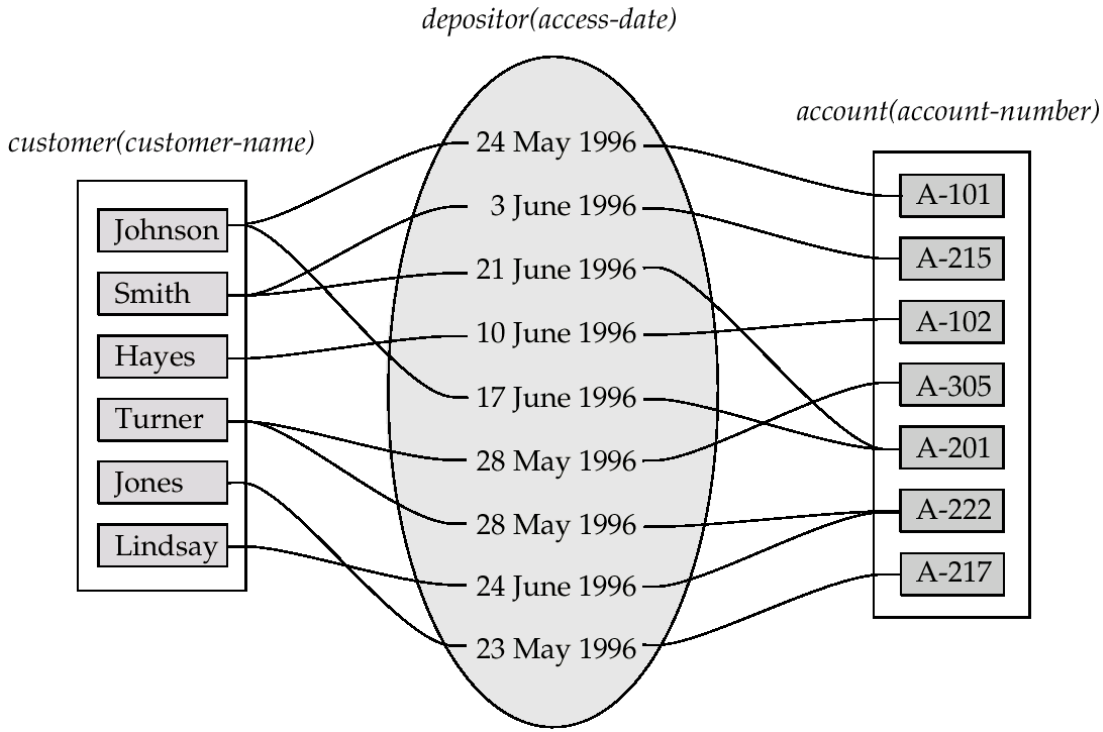
    $$(\text{Hayes}, \text{A-102}) \in \text{depositor}$$

Relationship Set borrower



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



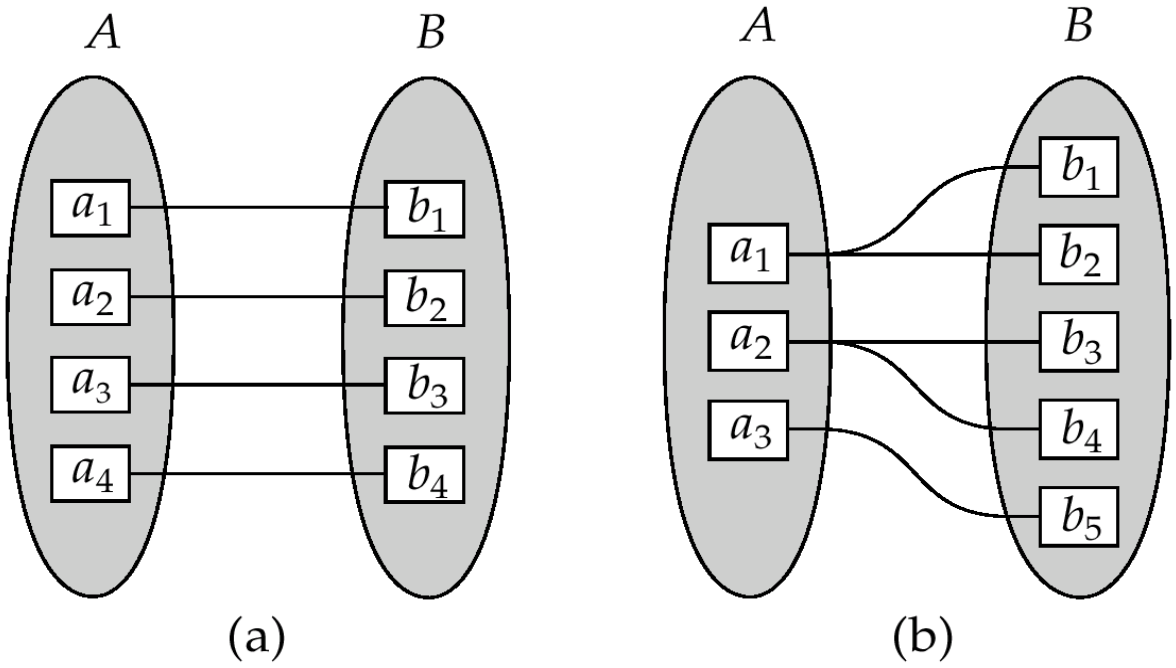
Degree （度/阶）of a Relationship Set

- Refers to number of entity sets that participate in a relationship set
- Relationship sets that involve two entity sets are **binary** (or degree two)
- Relationship sets may involve more than two entity sets
- Relationships between more than two entity sets are rare. Most relationships are binary

Mapping Cardinalities （映射的秩/基数）

- 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（1对1）
  - One to many（1对多）
  - Many to one（多对1）
  - Many to many（多对多）

Mapping Cardinalities

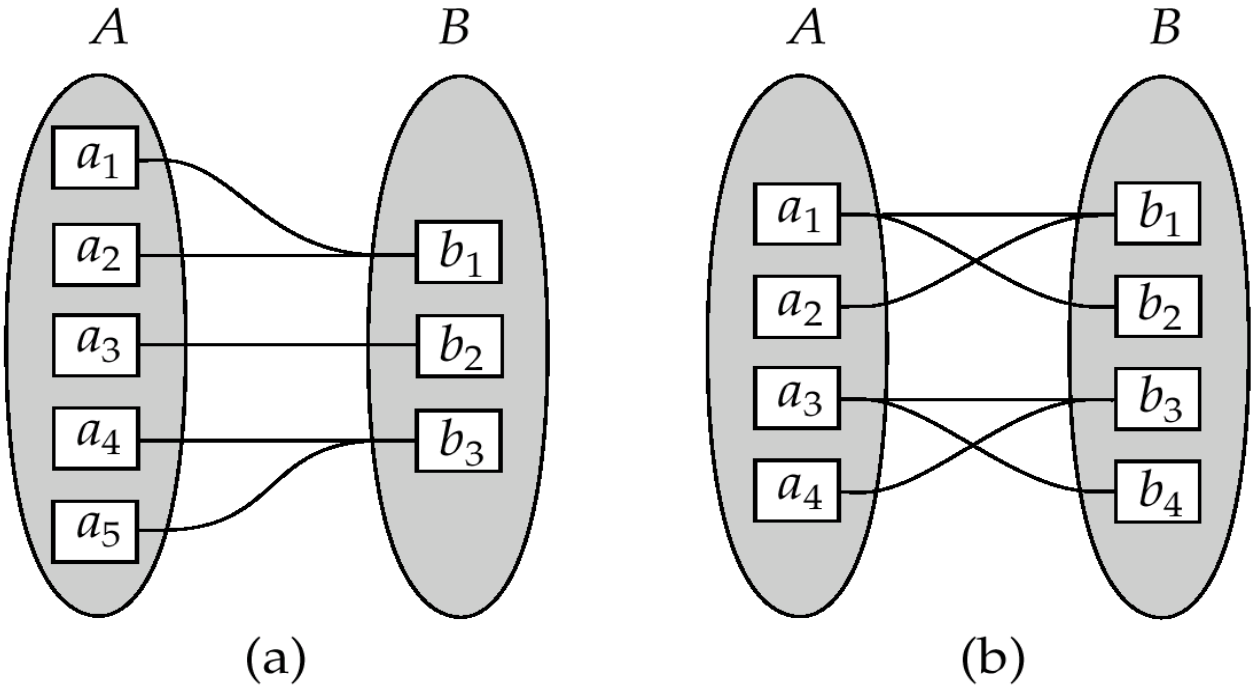


One to one

One to Many

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

Mapping Cardinalities



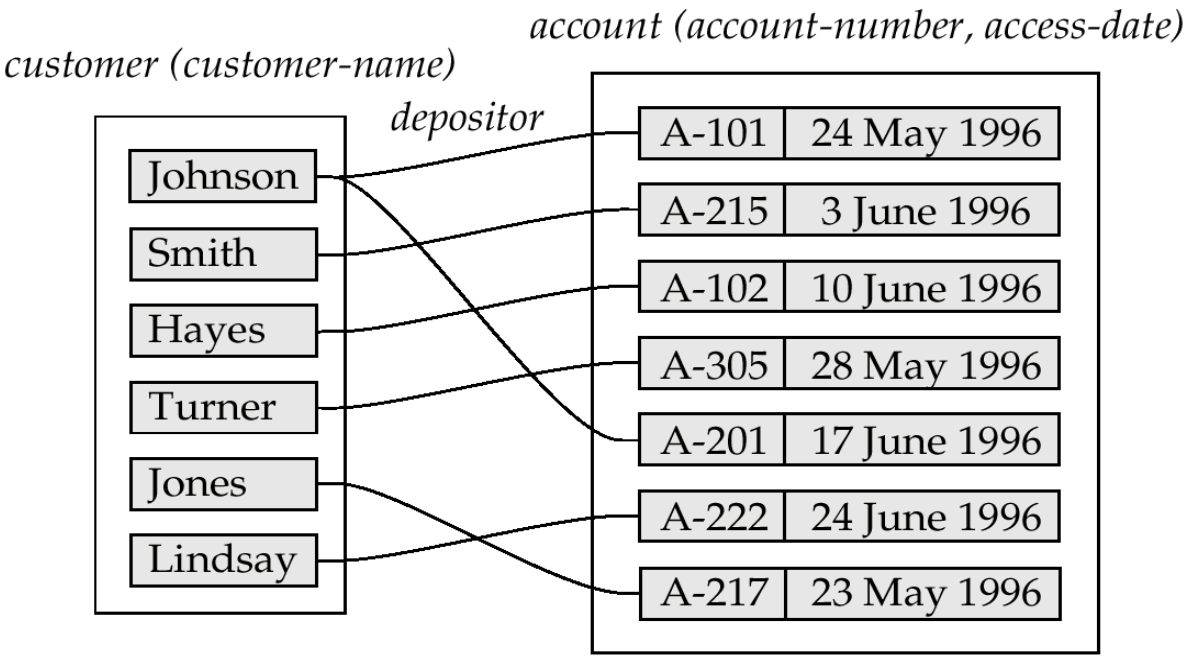
Many to one

Many to many

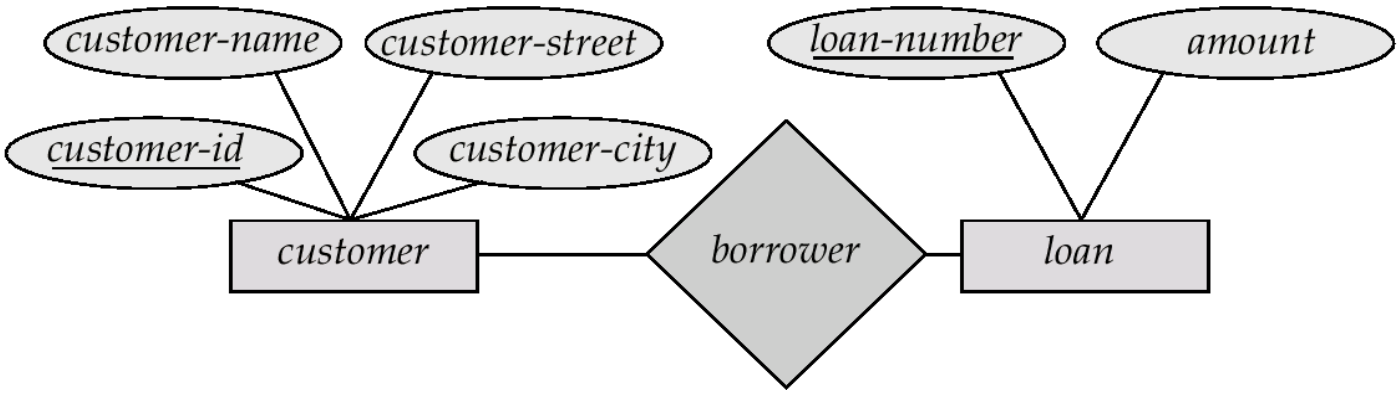
Note: Some elements in A and B may not be mapped to any elements in the other set

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

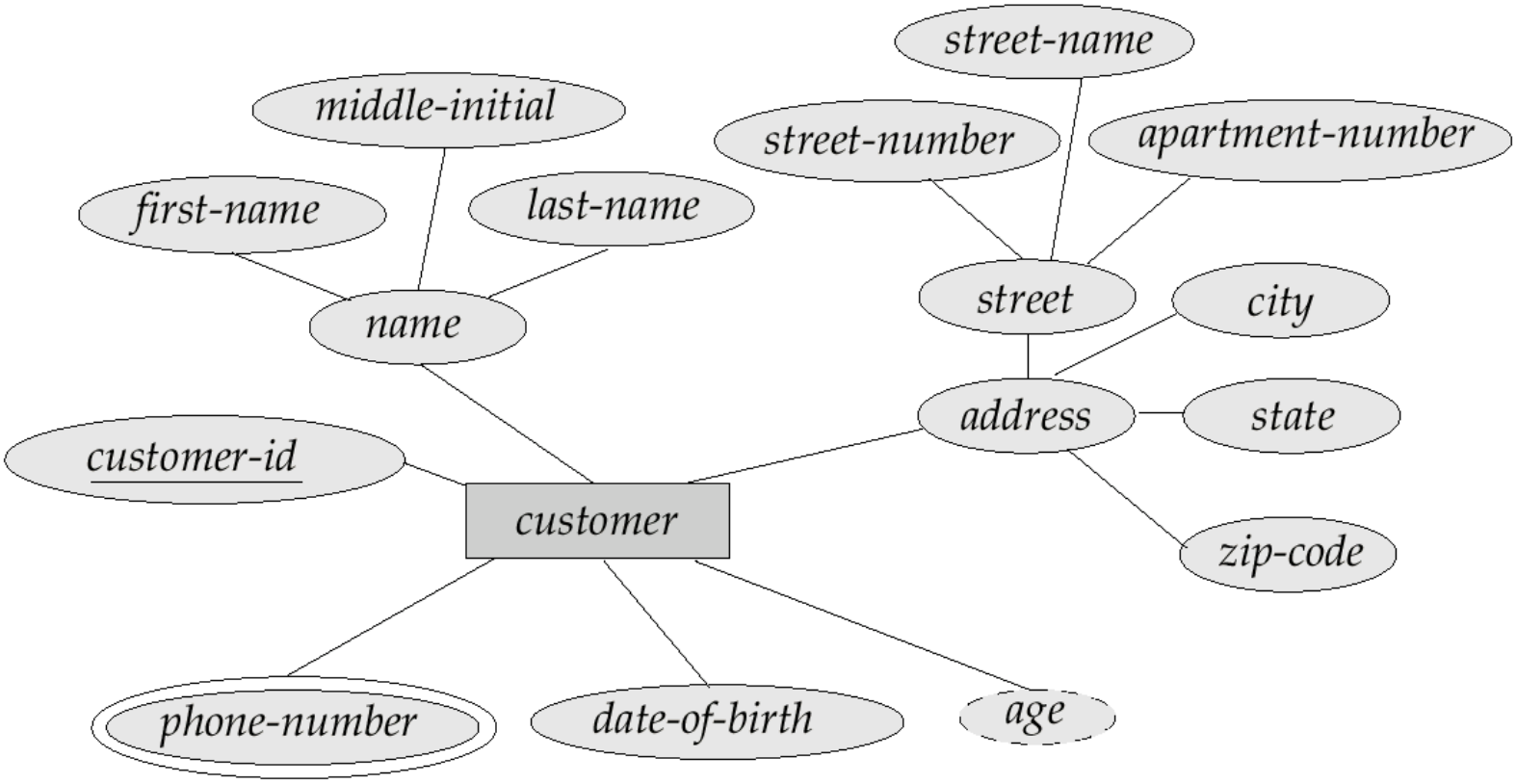


E-R Diagrams

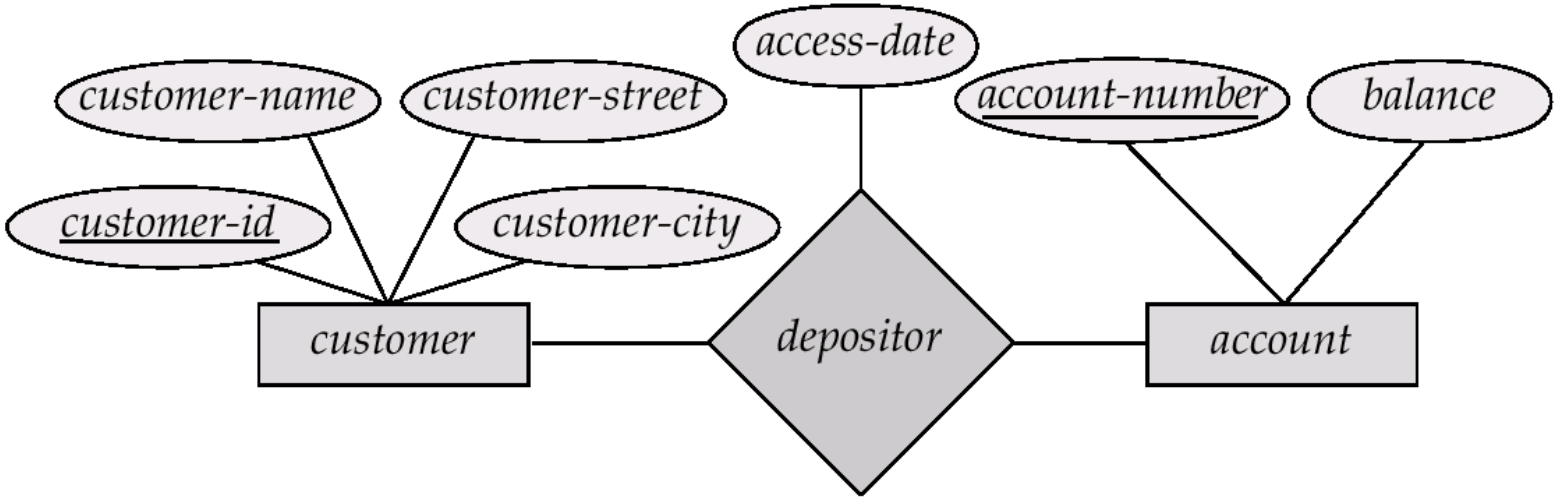


- Rectangles represent entity sets.
- Diamonds represent relationship sets.
- Lines link attributes to entity sets and entity sets to relationship sets.
- Ellipses represent attributes
  - Double ellipses represent multi-valued attributes
  - Dashed ellipses denote derived attributes
- Underline indicates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

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

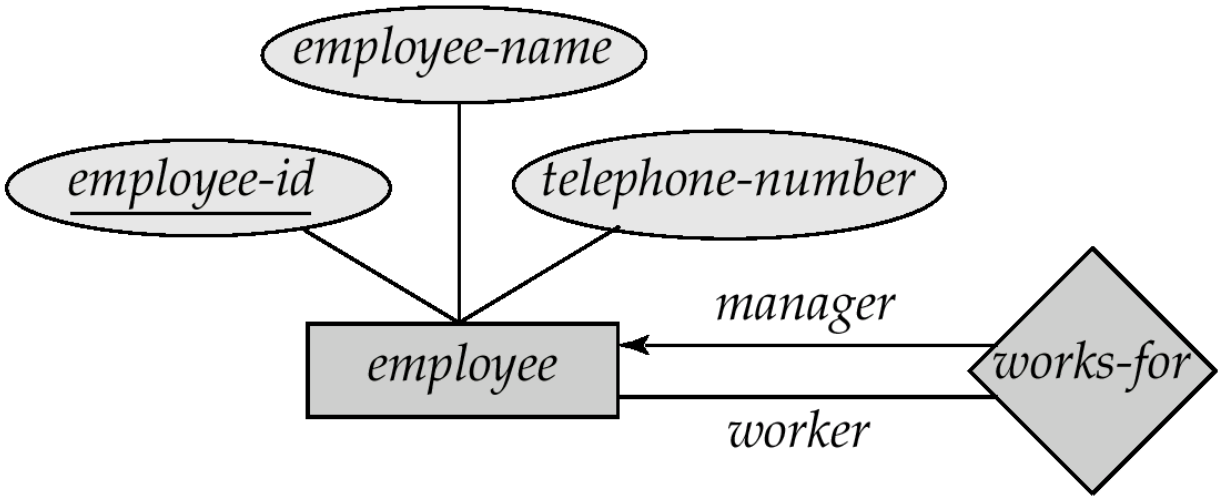


Relationship Sets with Attributes



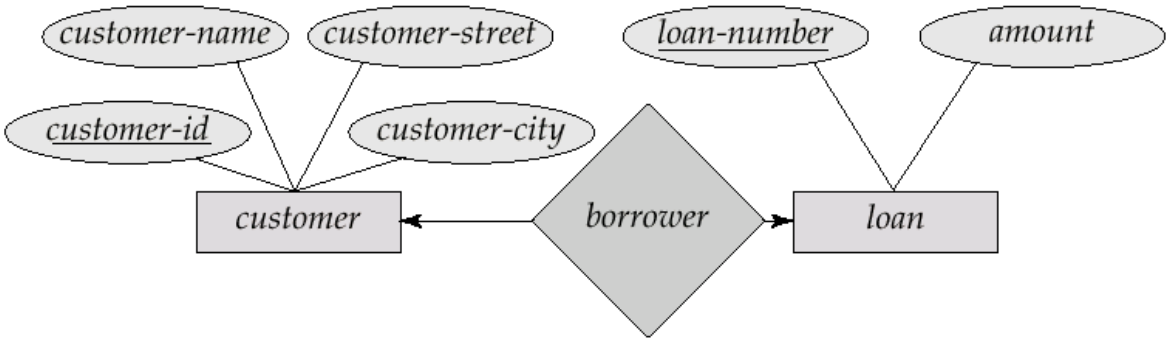
Roles

- Entity sets of a relationship set need not be distinct
  - The labels "manager" and "worker" are called roles; they specify how employee entities interact via the works-for relationship set.
  - Role labels are optional, and are used to clarify semantics of the relationship



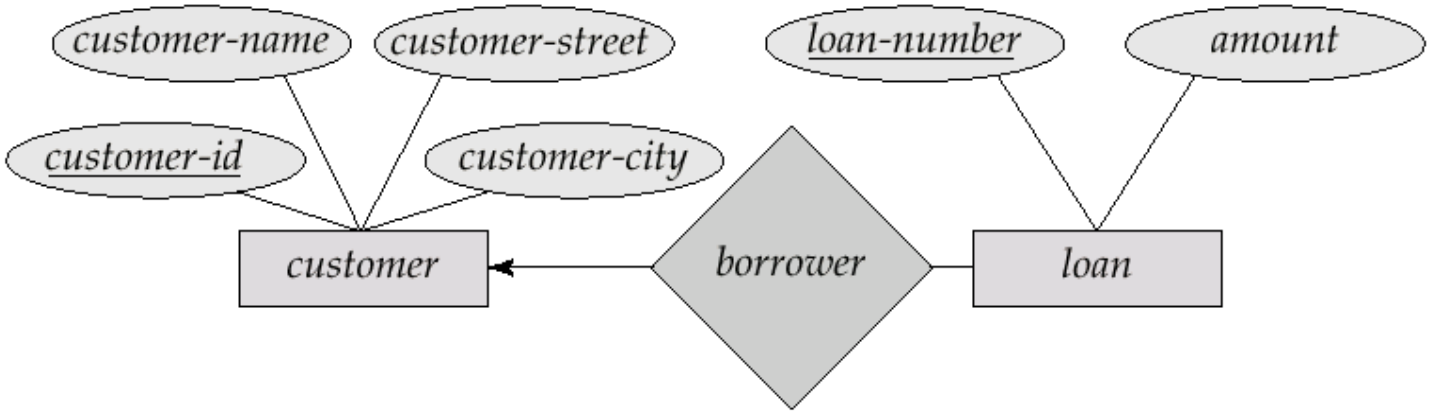
Cardinality Constraints

- 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.
- 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



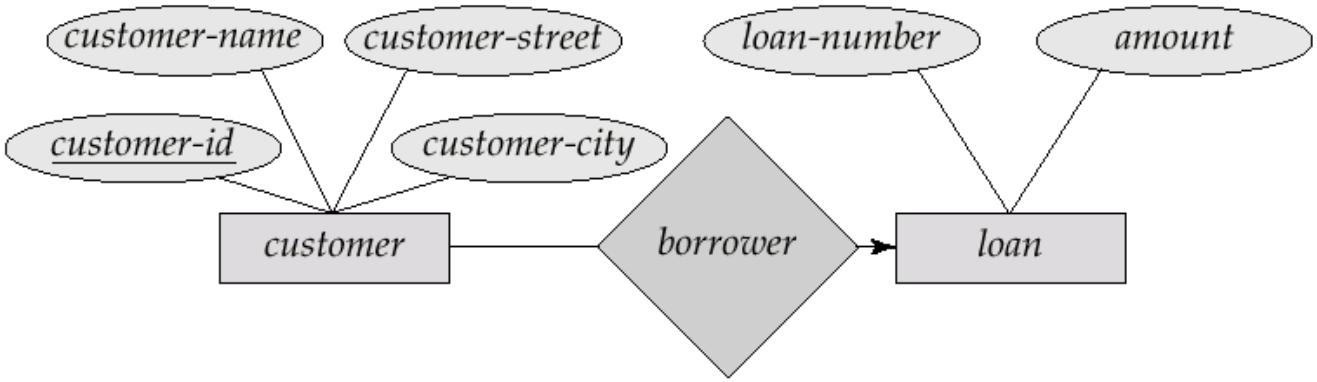
One-To-Many Relationship

- 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

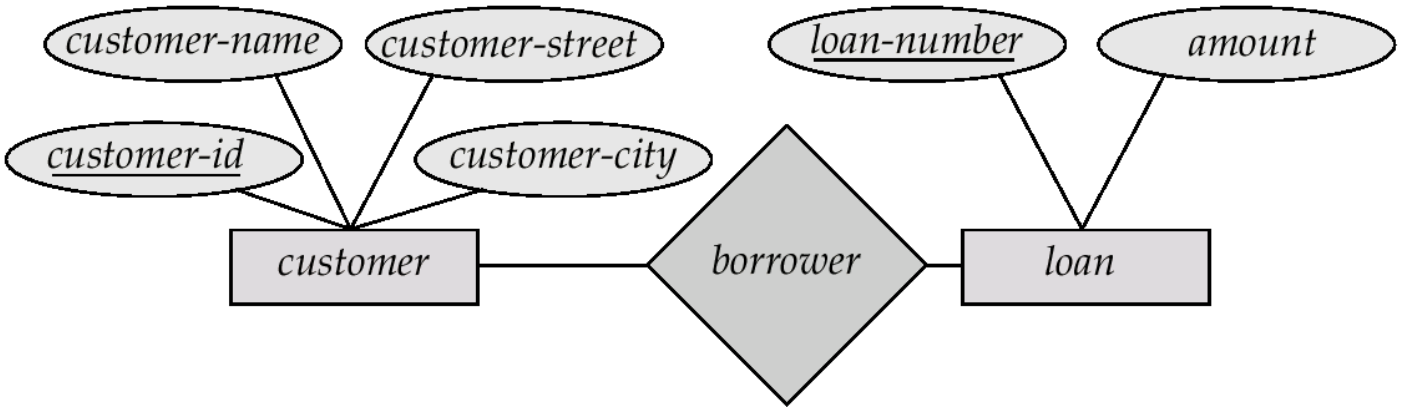


Many-To-One Relationships

- 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



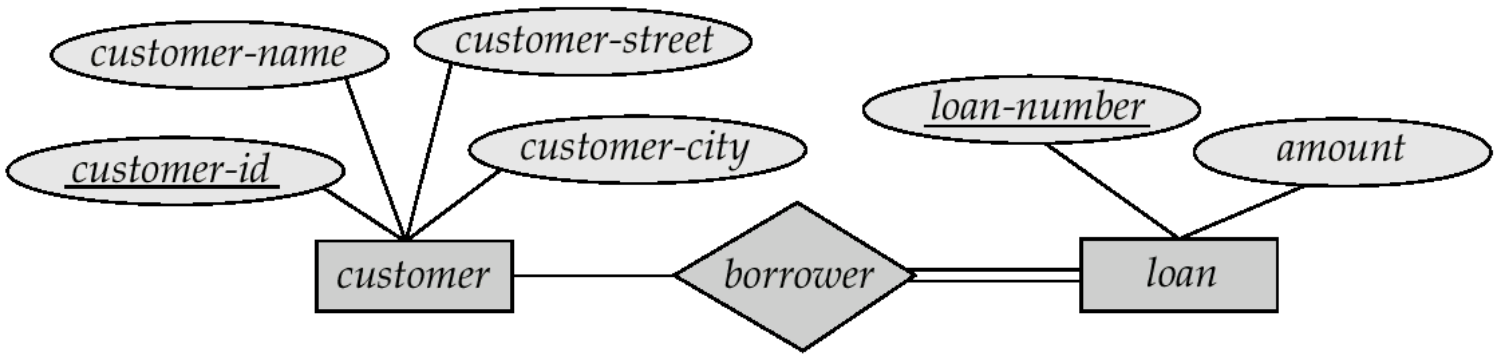
Many-To-Many Relationship



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

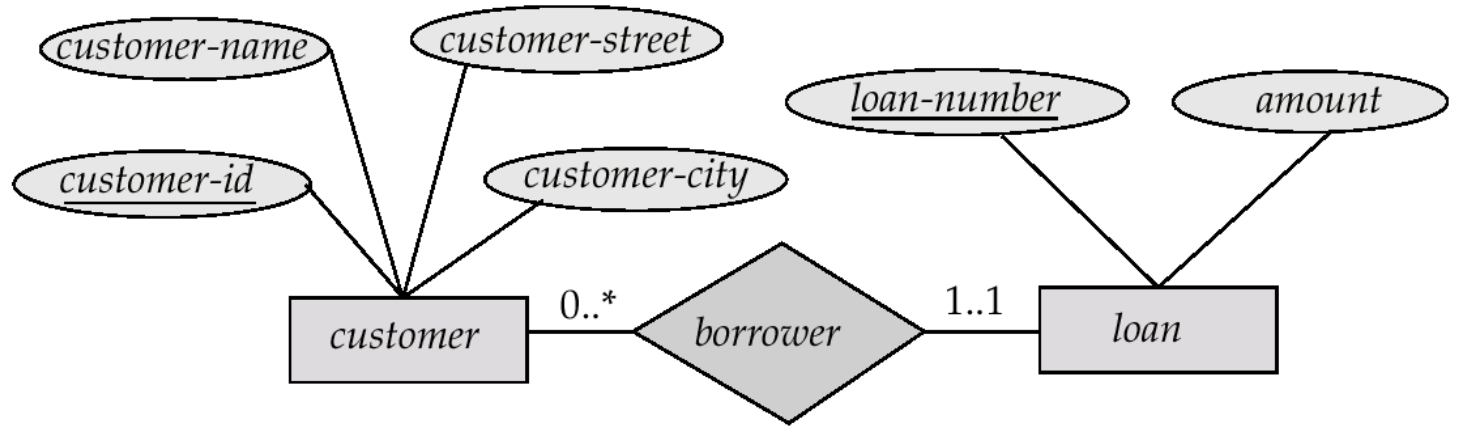
Participation of an Entity Set in a Relationship Set

- Total participation (indicated by double line): every entity in the entity set participates in at least one relationship in the relationship set
- Partial participation: some entities may not participate in any relationship in the relationship set



Alternative Notation for Cardinality Limits

- Cardinality limits can also express participation constraints
- Notation: l:h, here l and h are the minimum and maximum cardinalities respectively



One to Many

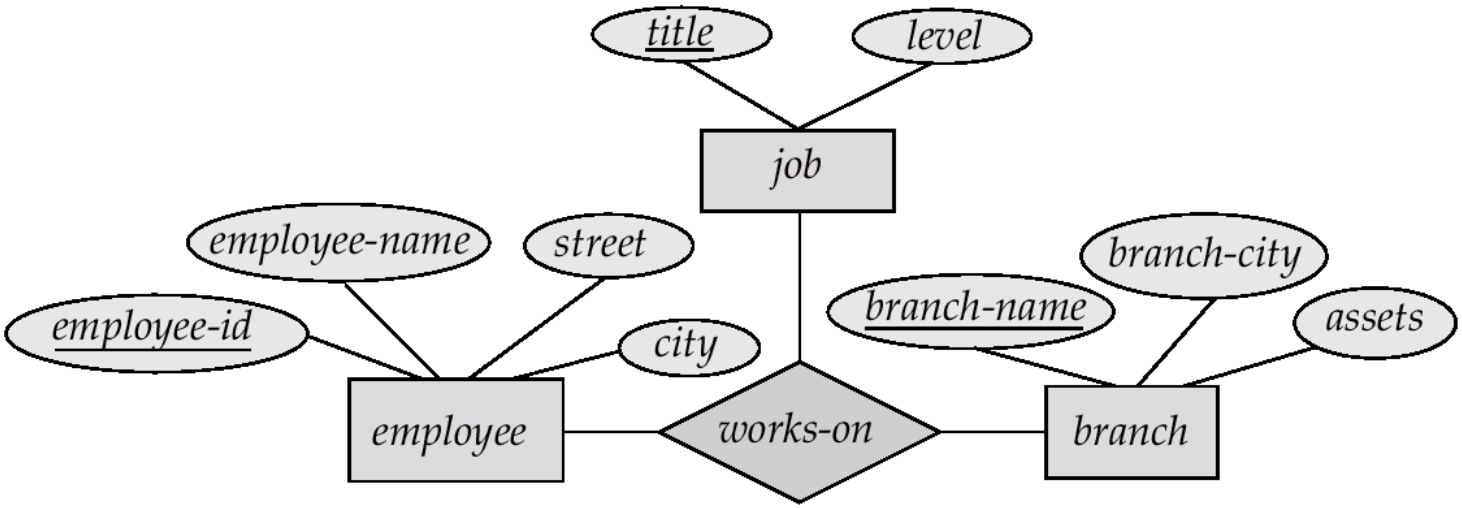
Keys（键）

- A super key（超键）of an entity set is a set of one or more attributes whose values uniquely determine each entity.
- A candidate key（候选键）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（主键）

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
- 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 relationship
- 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

To avoid confusion we outlaw more than one arrow

Binary Vs. Non-Binary Relationships

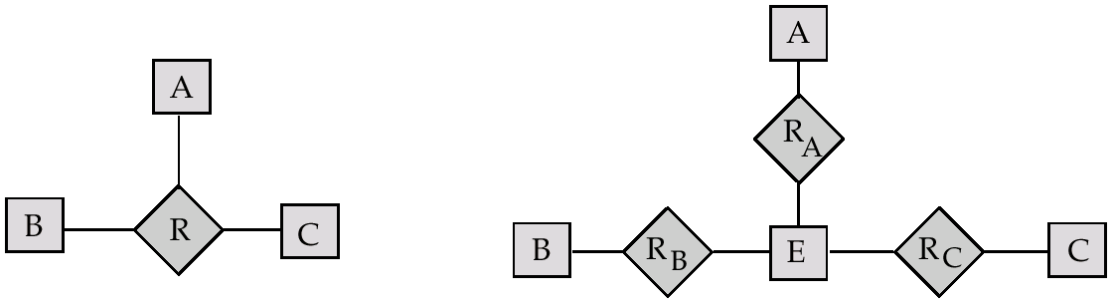
- Some relationships that appear to be non-binary may be better represented using binary relationships
  - E.g. A ternary relationship parents, relating a child to his/her father and mother, is best replaced by two binary relationships, father and mother
    - Using two binary relationships allows partial 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. RA, relating E and A
    2. RB, relating E and B
    3. RC, 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 RA
    3. add (e_i , b_i) to RB
    4. add (e_i , c_i) to RC



Converting Non-Binary Relationships (Cont.)

- 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
  - We can avoid creating an identifying attribute by making E a weak entity set (described shortly) identified by the three relationship sets

ER Design Issues

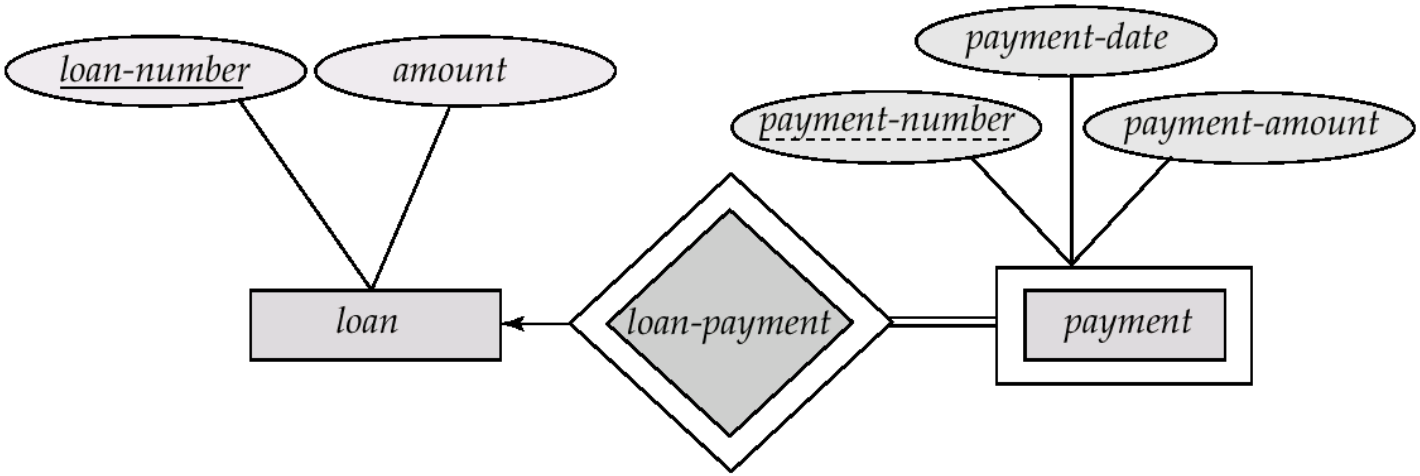
- Use of entity sets vs. attributes
  Choice mainly depends on the structure of the enterprise being modeled, and on the semantics associated with the attribute in question
- Use of entity sets vs. relationship sets
  Possible guideline is to designate a relationship set to describe an action that occurs between entities
- Binary versus n-ary relationship sets
- Placement of relationship attributes

Weak Entity Sets（弱实体集）

- 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
  - Identifying relationship depicted using a double diamond
- The discriminator (partial key)
- The primary key of a weak entity set
  - Discriminator plus primary keys of identifying entity sets

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)



Weak Entity Sets (Cont.)

- Note: 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.
- If loan-number were explicitly stored, payment could be made a strong entity

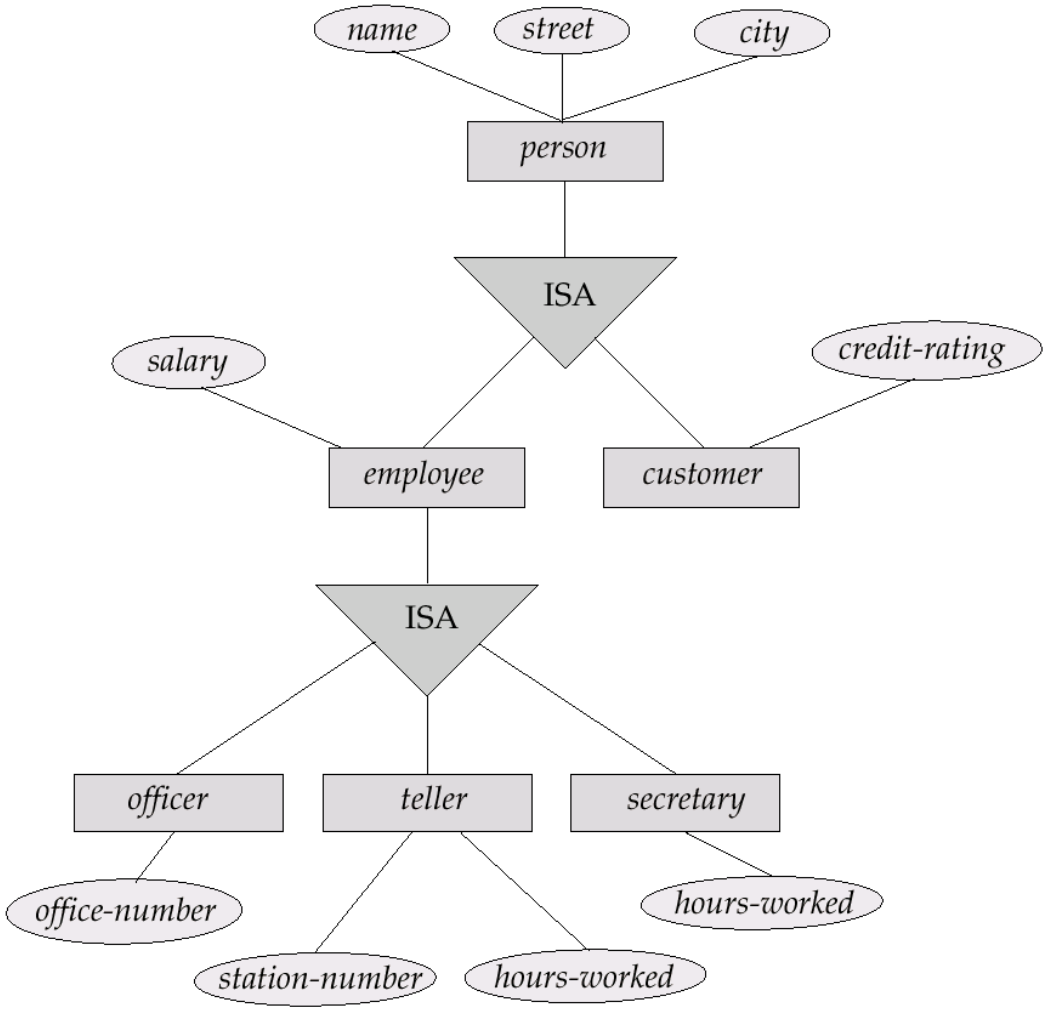
More Weak Entity Set Examples

- In a university, a course is a strong entity and a course-offering can be modeled as a weak entity
- The discriminator of course-offering would be semester (including year) and section-number (if there is more than one section)
- If we model course-offering as a strong entity we would model course-number as an attribute. Then the relationship with course would be implicit in the course-number attribute

Specialization（特化）

- 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（泛化）

- 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 (Contd.)

- 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

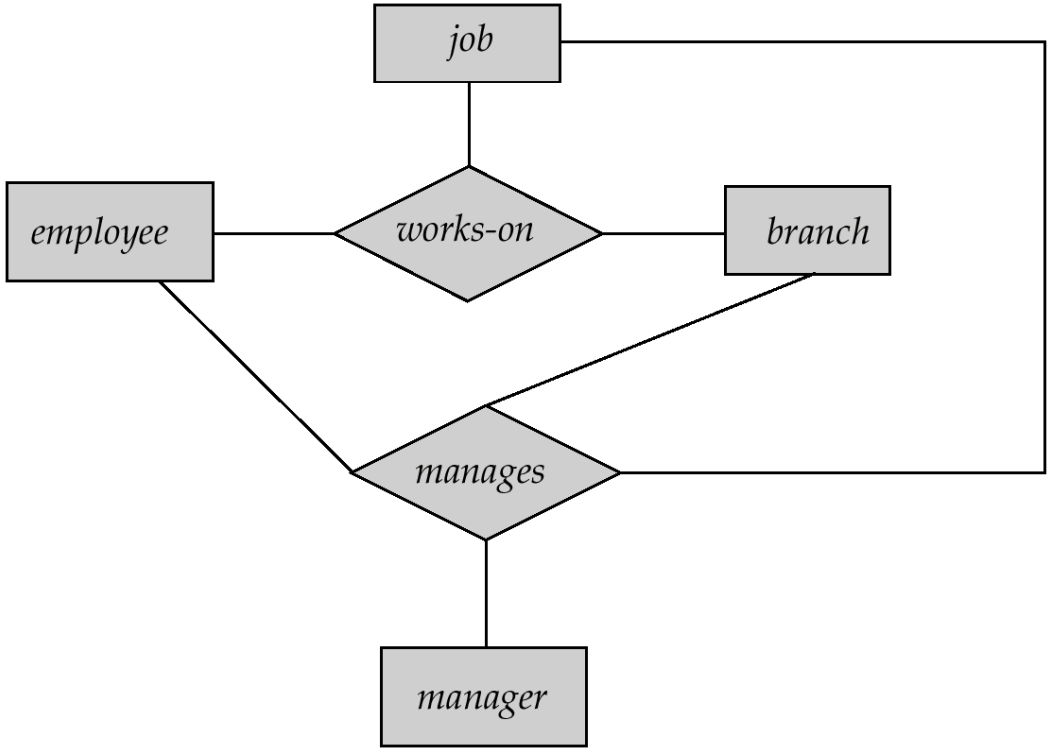
- Constraint on which entities can be members of a given lower-level entity set
  - condition-defined (attribute-defined)
  - user-defined
- Constraint on whether or not entities may belong to more than one lower-level entity set within a single generalization
  - Disjoint
  - Overlapping

Design Constraints on a Specialization/Generalization (Contd.)

- 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 (聚合)

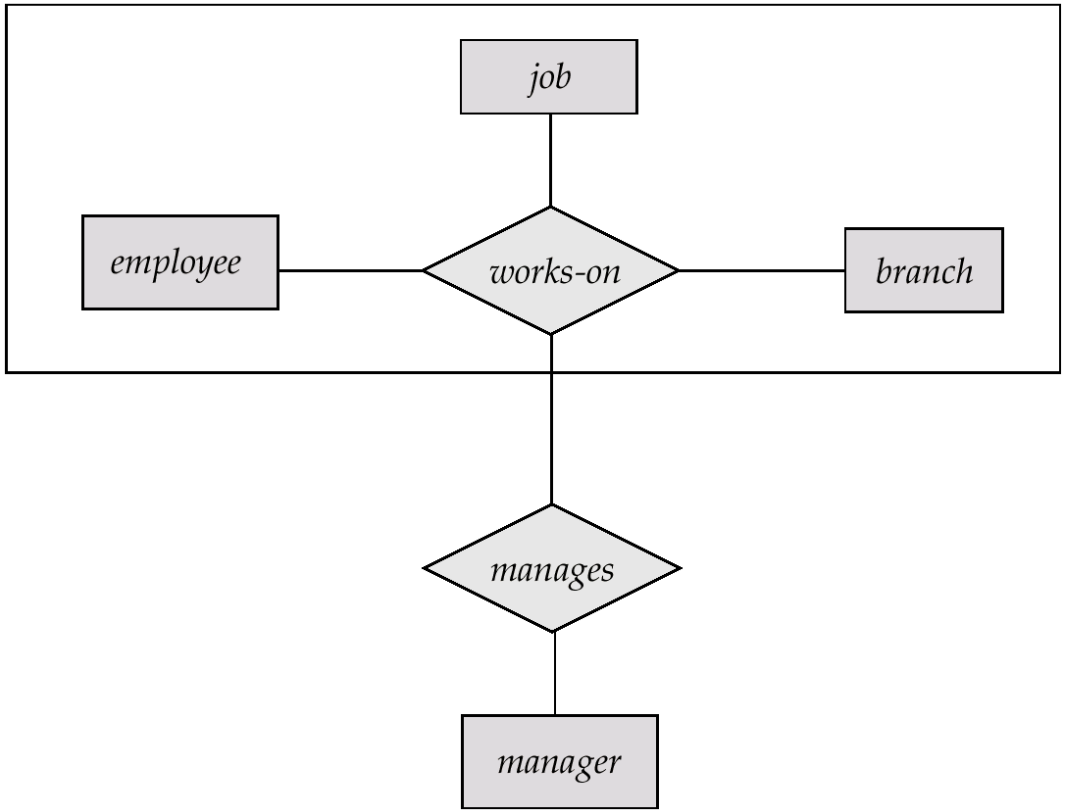
- Consider the ternary relationship works-on, which we saw earlier
- Suppose we want to record managers for tasks performed by an employee at a branch



Aggregation (Cont.)

- Relationship sets works-on and manages represent overlapping information
- Eliminate this redundancy via aggregation
  - Treat relationship as an abstract entity
  - Allows relationships between relationships
- 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



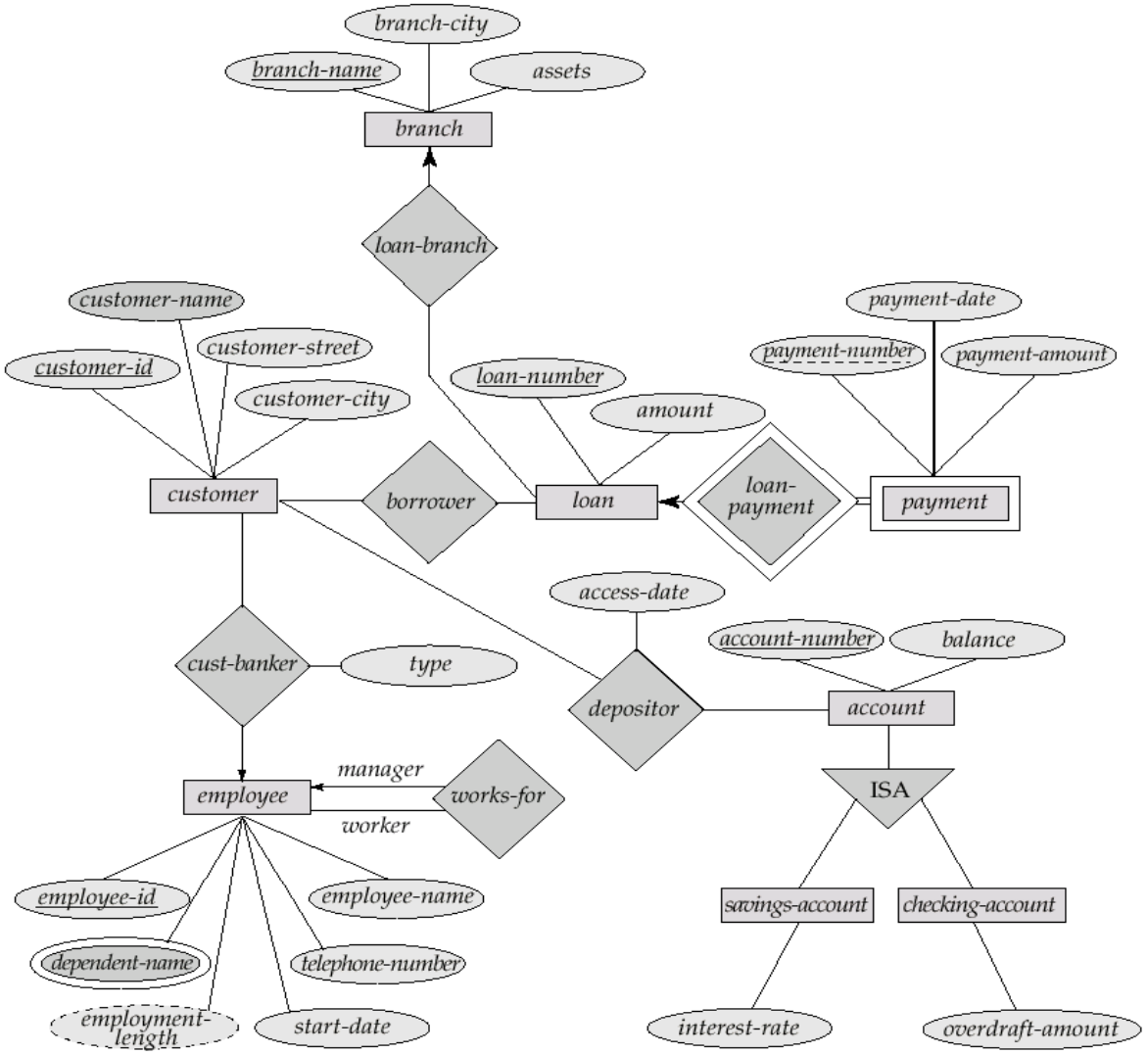
E-R Design Decisions

- 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

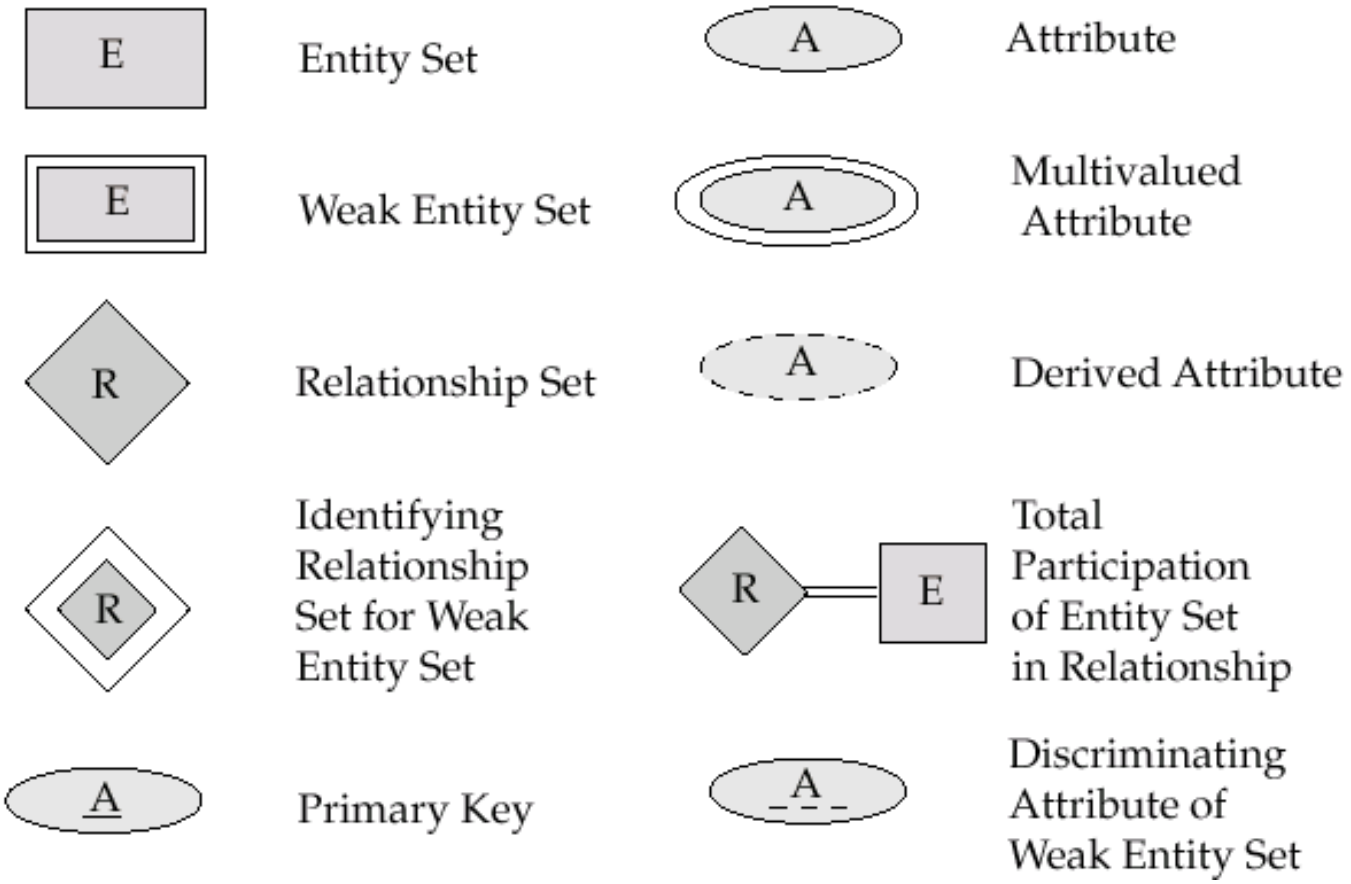
Database Design Phases

- Requirements Analysis
- Conceptual Design (E-R Model)
- Functional Requirements Analysis
  - Describe the operations that will be performed on the data
  - Review the design
- Logical Implementation
  - Mapping from conceptual model to implementation model
  - Such as relational model, OO model
- Physical Implementation
  - Specify physical features of the database
  - buffer size, index…

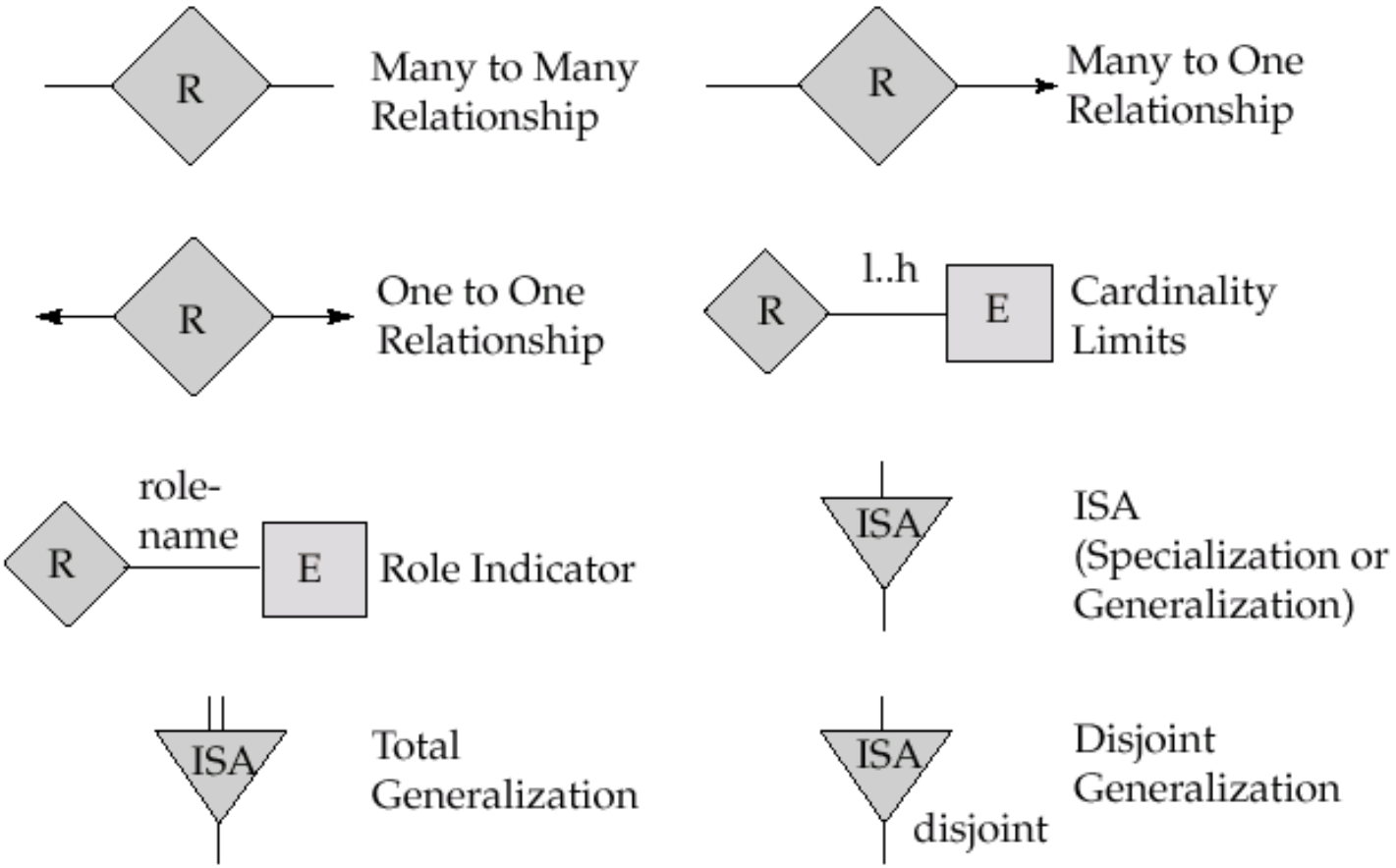
E-R Diagram for a Banking Enterprise



Summary of Symbols Used in E-R Notation

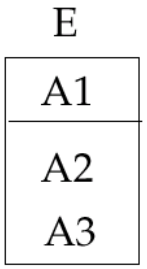


Summary of Symbols (Cont.)

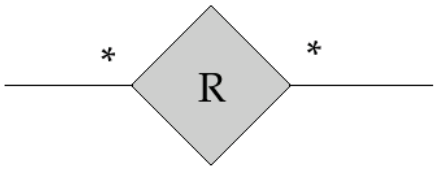


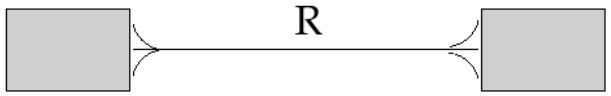
Alternative E-R Notations

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

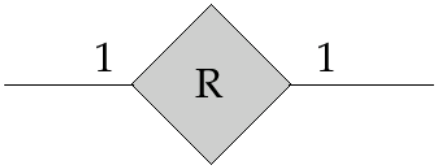


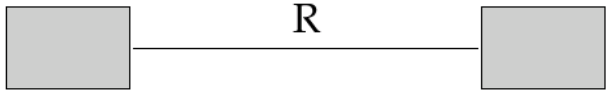
Many to Many Relationship



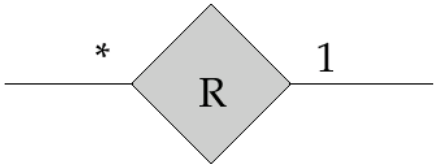


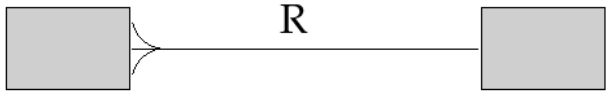
One to One Relationship





Many to One Relationship





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
- For each entity set and relationship set there is a unique table.
- Each table has a number of columns
- 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-id	*customer-name*	*customer-street*	*customer-city*
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 Multi-valued Attributes

- Composite attributes are flattened out by creating a separate attribute for each component attribute
- A multi-valued 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
  - 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

loan-number	*payment-number*	*payment-date*	*payment-amount*
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

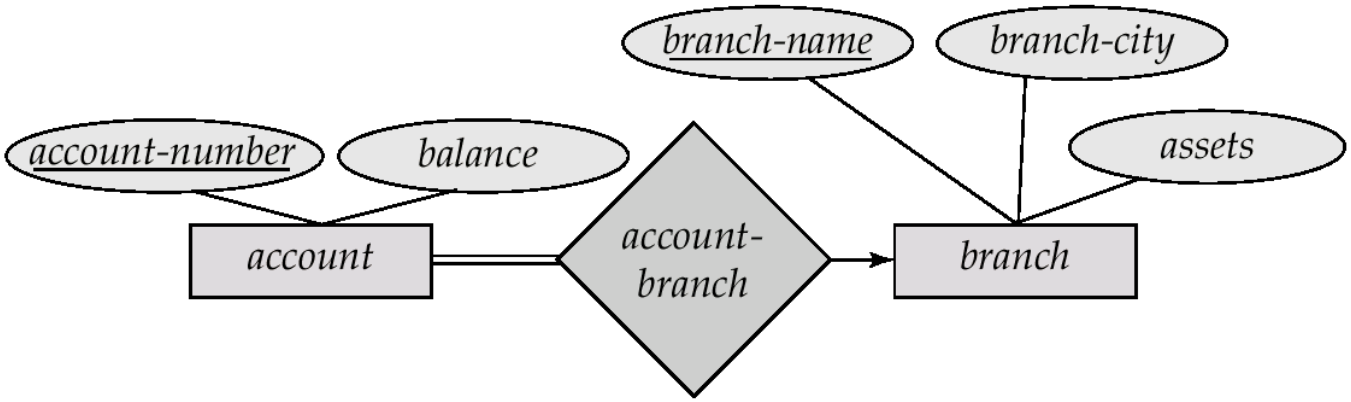
Representing Relationship Sets as Tables

- 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

customer-id	*loan-number*
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-name 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
- If participation is partial on the many side, it 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

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
person	name, street, city
customer	name, credit-rating
employee	name, salary

  - 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
person	name, street, city
customer	name, street, city, credit-rating
employee	name, street, city, salary

  - If specialization is total, table for generalized entity (person) not required to store information
  - Drawback: street and city may be stored redundantly for persons who are both customers and employees

Relations Corresponding to Aggregation

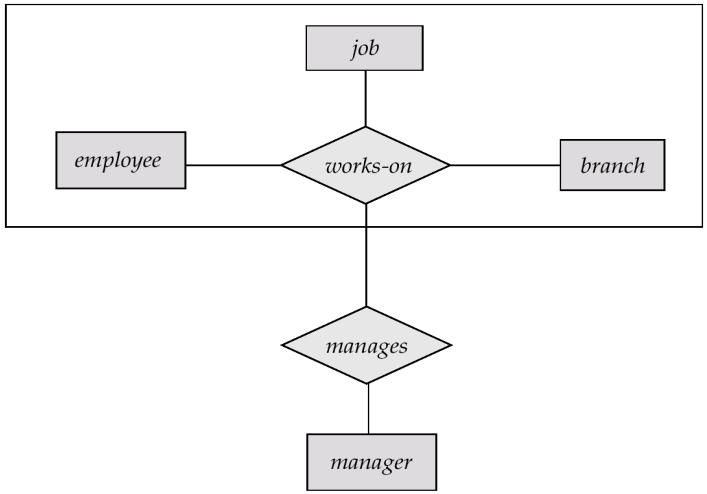
- To represent aggregation, create a table containing
  - primary key of the aggregated relationship,
  - the primary key of the associated entity set
  - Any descriptive attributes

Relations Corresponding to Aggregation (Cont.)

- E.g. to represent aggregation manages between relationship works-on and entity set manager, create a table

  manages(employee-id, branch-name, title, manager-name)

- Table works-on is redundant provided we are willing to store null values for attribute manager-name in table manages



Design Tools

- Rational Rose
  - http://www-306.ibm.com/software/rational/
- Visio Enterprise
  - http://www.microsoft.com/china/office/xp/visio/default.asp
- Erwin
  - http://www3.ca.com/Solutions/Product.asp?ID=260
- Power Designer
  - http://www.sybase.com/products/developmentintegration/powerdesigner

Summary of ER Model

- *Conceptual design* follows *requirements analysis*
  - Yields a high-level description of data to be stored
- ER model popular for conceptual design
  - Constructs are expressive, close to the way people think about their applications
- Basic constructs: *entities*, *relationships*, and *attributes* (of entities and relationships)
- Some additional constructs: *weak entities*, *ISA hierarchies*, and *aggregation*
- Note: There are many variations on ER model

Summary of ER Model (Contd.)

- Several kinds of integrity constraints can be expressed in the ER model: *key constraints*, *participation constraints*, and *overlap/covering constraints* for ISA hierarchies. Some *foreign key constraints* are also implicit in the definition of a relationship set.
  - Some constraints (notably, *functional dependencies*) cannot be expressed in the ER model.
  - Constraints play an important role in determining the best database design for an enterprise.

Summary of ER Model (Contd.)

- ER design is *subjective*. There are often many ways to model a given scenario! Analyzing alternatives can be tricky, especially for a large enterprise. Common choices include:
  - Entity vs. attribute, entity vs. relationship, binary or n-ary relationship, whether or not to use ISA hierarchies, and whether or not to use aggregation.
- Ensuring good database design: resulting relational schema should be analyzed and refined further. FD information and normalization techniques are especially useful

Homework

- Practice Exercises: 6.2
- Exercises: 6.21

End of Lecture 5