Introduction to Databases
《数据库引论》

Lecture 5: Database Design and E-R Model
第5讲：数据库设计与E-R模型

周水庚 / Shuigeng Zhou

邮件: sgzhou@fudan.edu.cn 网址：admis.fudan.edu.cn/sgzhou

复旦大学计算机科学技术学院

Content of the Course

- **Part 0: Overview**
  - Lect. 0/1 (Feb. 20) - Ch1: Introduction
- **Part 1 Relational Databases**
  - Lect. 2 (Feb. 27) - Ch2: Relational model (data model, relational algebra)
  - Lect. 3 (Mar. 6) - Ch3: SQL (Introduction)
  - Lect. 4 (Mar. 13) – Ch4 & 5: Intermediate & Advanced SQL
- **Part 2 Database Design**
  - Lect. 5 (Mar. 20) - Ch6: Database design based on E-R model
  - Lect. 6 (Mar. 27) - Ch7: Relational database design (Part I)
  - Lect. 7 (Apr. 3) - Ch7: Relational database design (Part II)
- **Midterm exam: Apr. 10**
- **Part 3 Data Storage & Indexing**
  - Lect. 7 (Apr. 17) - Ch12/13: Storage systems & structures
  - Lect. 8 (Apr. 24) - Ch14: Indexing
- **Part 4 Query Processing & Optimization**
  - May 1, holiday, no classes
  - Lect. 9 (May 8) - Ch15: Query processing
  - Lect. 10 (May 15) - Ch16: Query optimization
- **Part 5 Transaction Management**
  - Lect. 11 (May 22) - Ch17: Transactions
  - Lect. 12 (May 29) - Ch18: Concurrency control
  - Lect. 13 (Jun. 5) - Ch19: Recovery system
  - Lect. 14 (Jun. 5) – Course review

Final exam: 13:00-15:00, Jun. 18

University Database

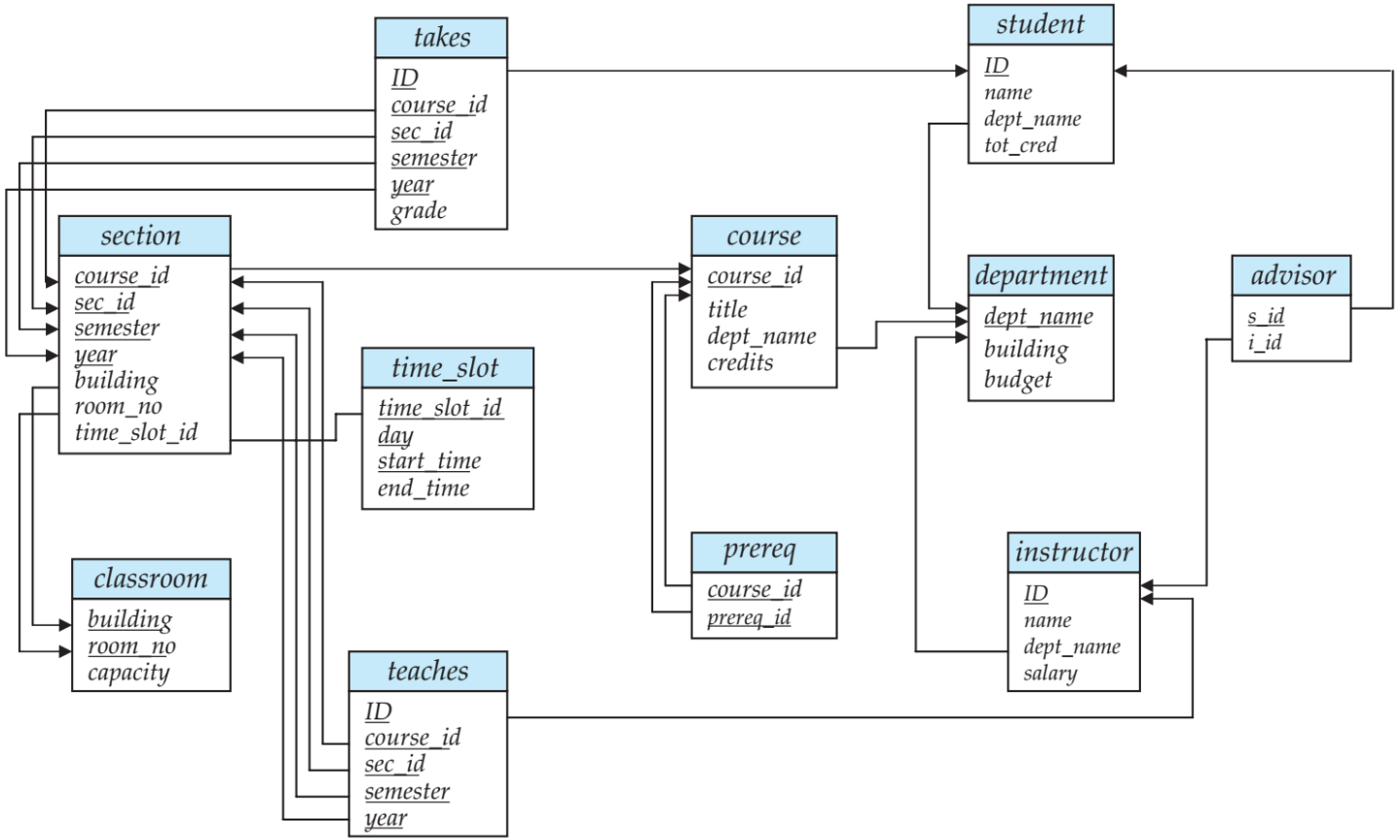
ID	*name*	*dept_name*	*salary*
22222	Einstein	Physics	95000
12121	Wu	Finance	90000
32343	El Said	History	60000
45565	Katz	Comp. Sci.	75000
98345	Kim	Elec. Eng.	80000
76766	Crick	Biology	72000
10101	Srinivasan	Comp. Sci.	65000
58583	Califieri	History	62000
83821	Brandt	Comp. Sci.	92000
15151	Mozart	Music	40000
33456	Gold	Physics	87000
76543	Singh	Finance	80000

Instructor table

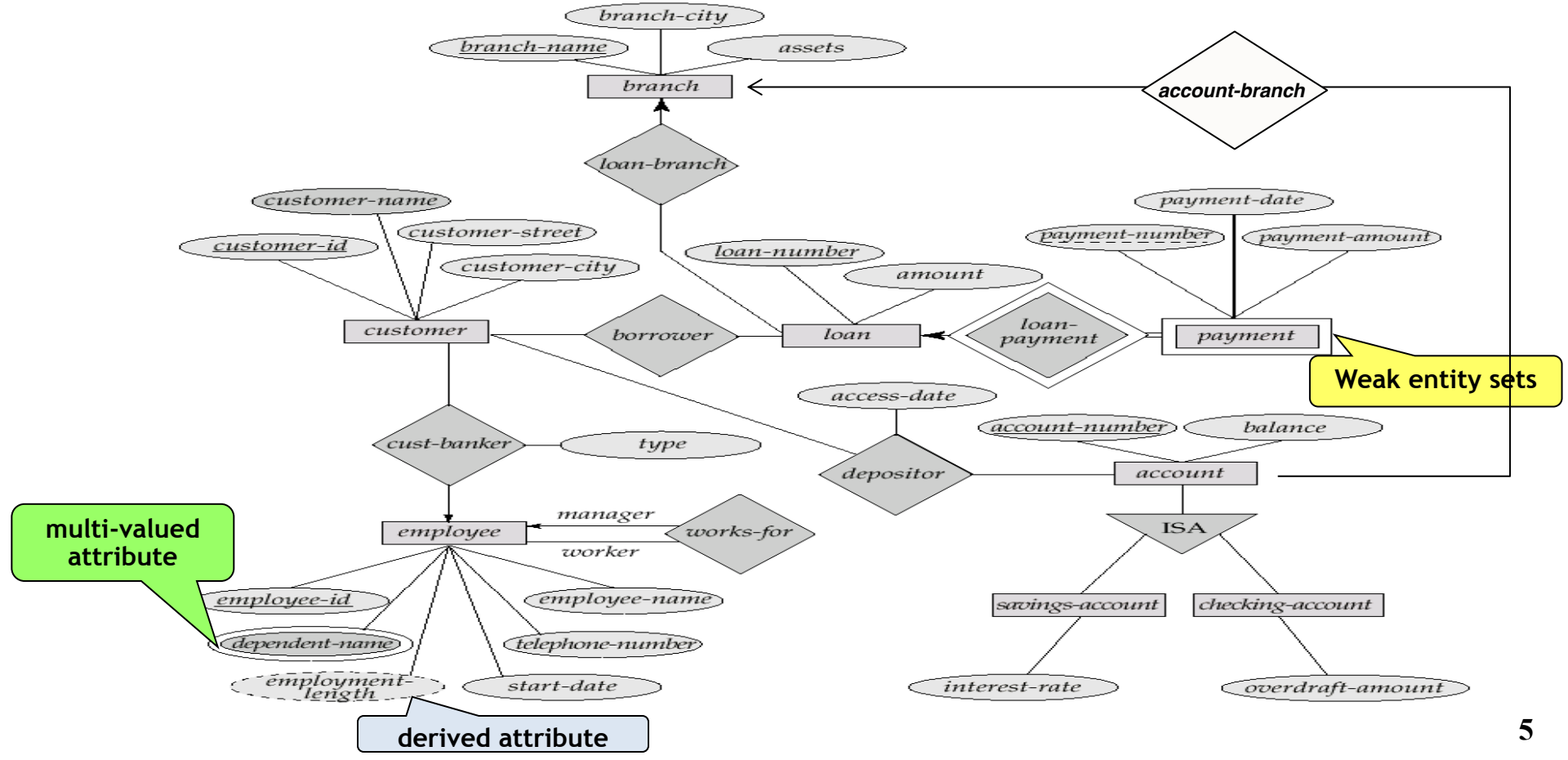
ID	*name*	*dept_name*	*tot_cred*
00128	Zhang	Comp. Sci.	102
12345	Shankar	Comp. Sci.	32
19991	Brandt	History	80
23121	Chavez	Finance	110
44553	Peltier	Physics	56
45678	Levy	Physics	46
54321	Williams	Comp. Sci.	54
55739	Sanchez	Music	38
70557	Snow	Physics	0
76543	Brown	Comp. Sci.	58
76653	Aoi	Elec. Eng.	60
98765	Bourikas	Elec. Eng.	98
98988	Tanaka	Biology	120

Student table

University Database



E-R Diagram for a Banking Enterprise



The Banking Schema

- *branch* = (*branch_name*, *branch_city*, *assets*)
- *customer* = (*customer_id*, *customer_name*, *customer_street*, *customer_city*)
- *loan* = (*loan_number*, *amount*)
- *account* = (*account_number*, *balance*)
- *employee* = (*employee_id*, *employee_name*, *telephone_number*, *start_date*)

- *dependent_name* = (*employee_id*, *dname*) (derived from a multivalued attribute)

- *account_branch* = (*account_number*, *branch_name*)
- *loan_branch* = (*loan_number*, *branch_name*)
- *borrower* = (*customer_id*, *loan_number*)
- *depositor* = (*customer_id*, *account_number*, *access_date*)
- *cust_banker* = (*customer_id*, *employee_id*, *type*)
- *works_for* = (*worker_employee_id*, *manager_employee_id*)

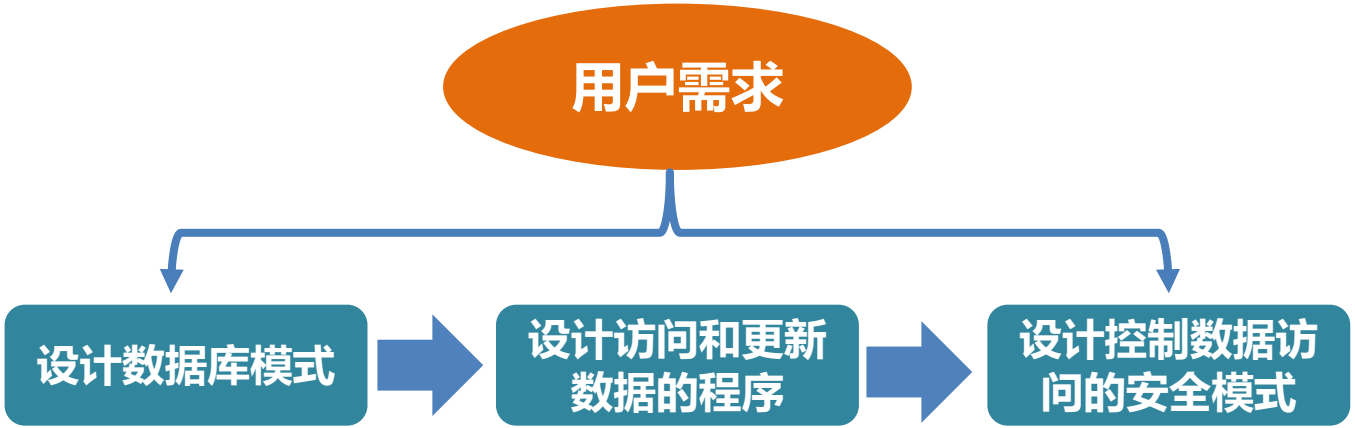
- *payment* =(*loan_number*,*payment_number*,*payment_date*,*payment_amount*)

- *savings_account* = (*account_number*, *interest_rate*)
- *checking_account* = (*account_number*, *overdraft_amount*)

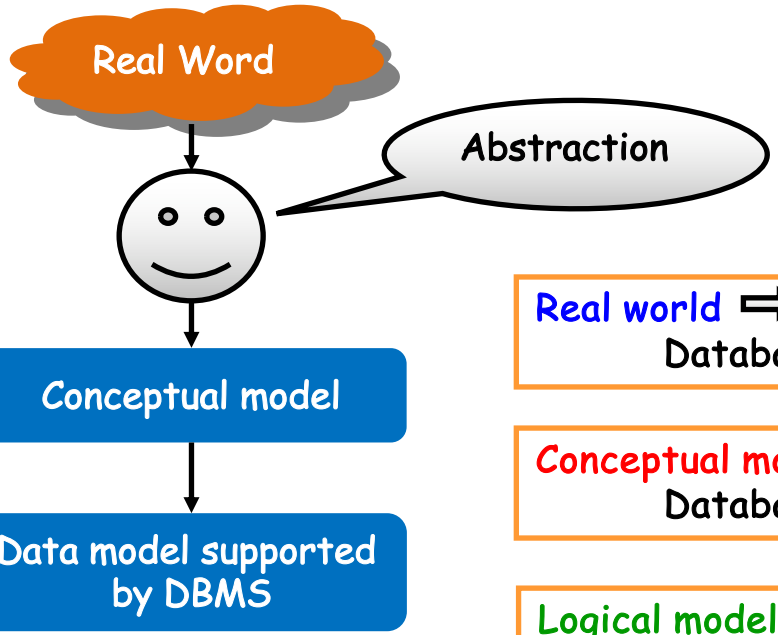
Outline

- ☞ Overview of the DB Design Process
- Entity-Relationship Model (**实体联系模型**)
- Constraints
- Entity-Relationship Diagrams (**实体联系图**)
- Reduction to Relation Schemas
- Summary

开发数据库应用包含的任务



Data Abstraction



Information world

Machine World

Real world ⇨ Conceptual model
Database Designers

Conceptual model ⇨ Logical model
Database Designers

Logical model ⇨ Physical model
DBMS

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
- **Physical design (物理设计)**
  - Instantiating the logical model to physical organization and storage

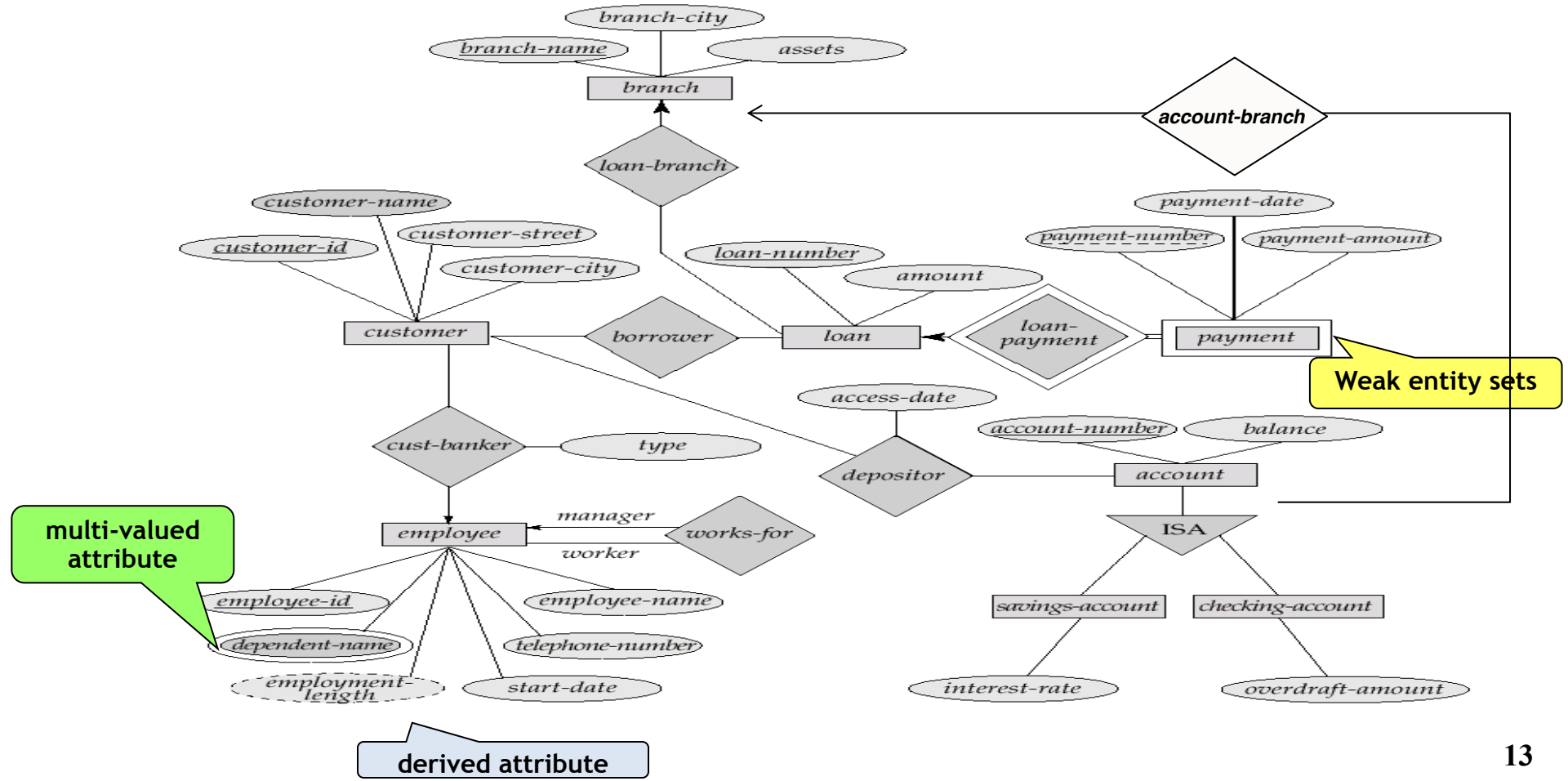
Database Design (Cont.)

- 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
    - **Entity/Relationship (E/R) model**
    - Object Definition Language (ODL)
- Translate specification to the **data model of DBMS**
  - **Relational**, XML, object-oriented, etc.
- **Create the DBMS schema (by DDL)**

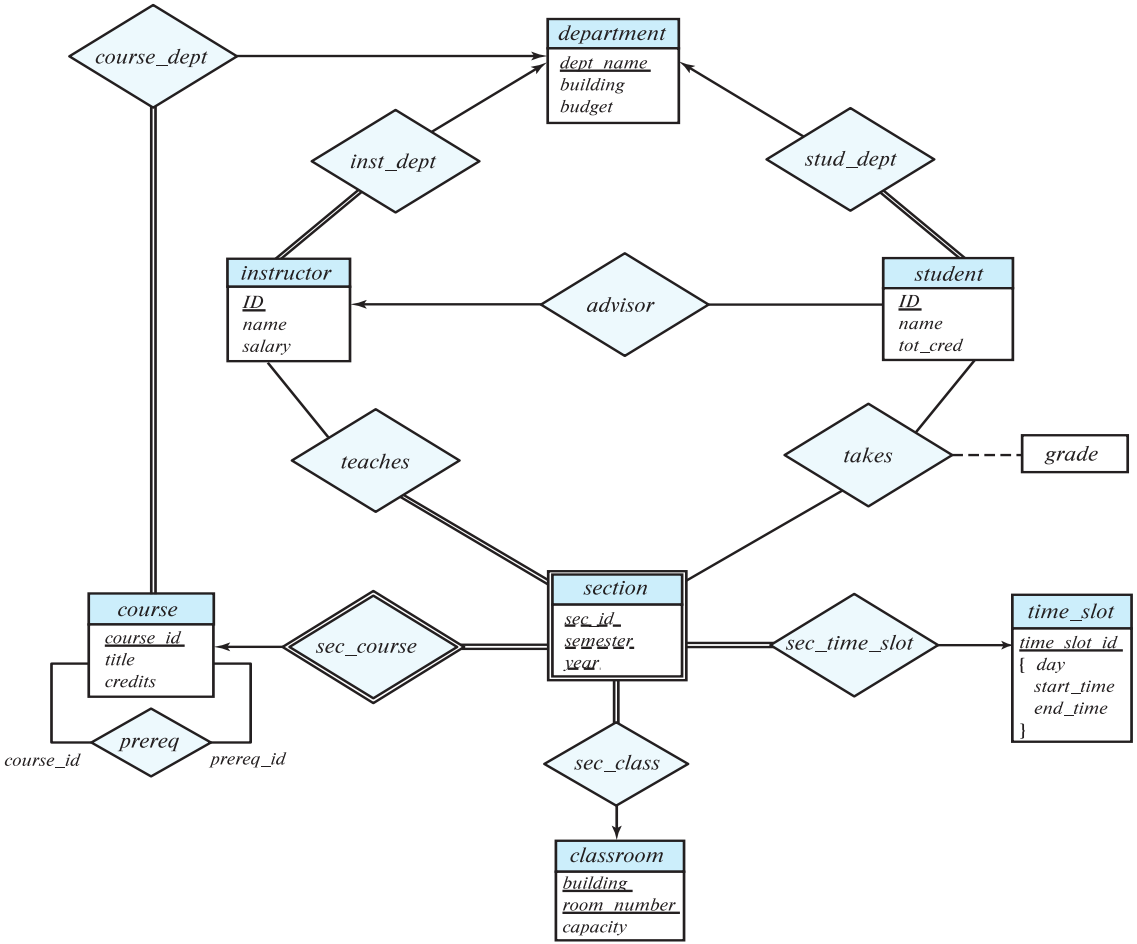
Outline

- Overview of the Design Process
- ☞ **Entity-Relationship Model (实体联系模型)**
- Constraints
- Entity-Relationship Diagrams (**实体联系图**)
- Reduction to Relation Schemas
- Summary

E-R Diagram for a Banking Enterprise



大学E-R图


course_dept
department
dept_name
building
budget
inst_dept
stud_dept
instructor
ID
name
salary
advisor
student
ID
name
tot_cred
teaches
takes
grade
course
course_id
title
credits
sec_course
section
sec_id
semester
year
sec_time_slot
time_slot
time_slot_id
{ day
start_time
end_time
}
prereq
course_id
prereq_id
sec_class
classroom
building
room_number
capacity

Database Conceptual Design

- **Conceptual design (ER Model is used at this stage)**
  - **What** are the **entities** and **relationships**?
  - **What information** about these entities and relationships should we store in the database?
  - **What** are the **integrity constraints** or **business rules** that should hold?

  - A database 'schema' in the ER Model can be represented pictorially using **ER diagram**
  - Can map an ER diagram into a relational schema

ER Model: A General View

- Historically very popular
- A "watered-down" object-oriented design model
- ER diagrams represent designs
- Primarily a design model—not implemented by any major DBMS
- **Three concepts**
  - **Entity set**
  - **Attributes**
  - **Relationship sets**

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. ER 2004 was held in Fudan
- 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

More about Peter Chen

- **Education**
  - B.S. in Electrical Engineering, National Taiwan University, 1968
  - M.S. and Ph.D. in Computer Science/Applied Math., Harvard University, 1970/1973
- **Academic Experience**
  - Assistant Professor at MIT Sloan School, 1974 -78
  - Associate Professor at UCLA Management School, 1978 - 84
  - Murphy J. Foster Professor at Computer Science Department, Louisiana State University (LSU), 1983 – 2011; Adjunct Professor, 2011-Present

Entity Sets（实体集）

- **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
  - E.g., specific person, company, event, university, instructor, student, course, classroom….
- Entities have **attributes**
  - E.g., people have names and addresses
- **An entity set** is a set of entities of the same type that share the same properties
  - E.g., the set of all persons, companies, trees, holidays, students
- **Extension of the entity set** (实体集的外延) is the actual collection of entities belonging to the entity set
  - An entity set vs. extension of the entity set ; a relation vs. a relation instance

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

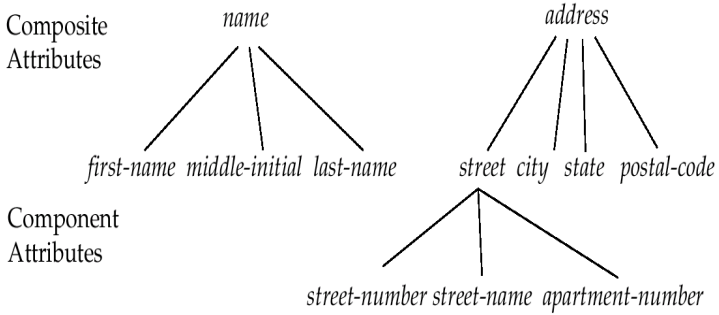
loan

Attributes（属性）

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

  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**
  - Simple and composite attributes (复合属性)
  - Single-valued and multi-valued attributes
    - phone_number
  - Derived attributes
    - age (derived from birthday)



Relationship Sets (联系集)

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

Hayes *depositor* *A-102*
customer entity *relationship set* *account entity*

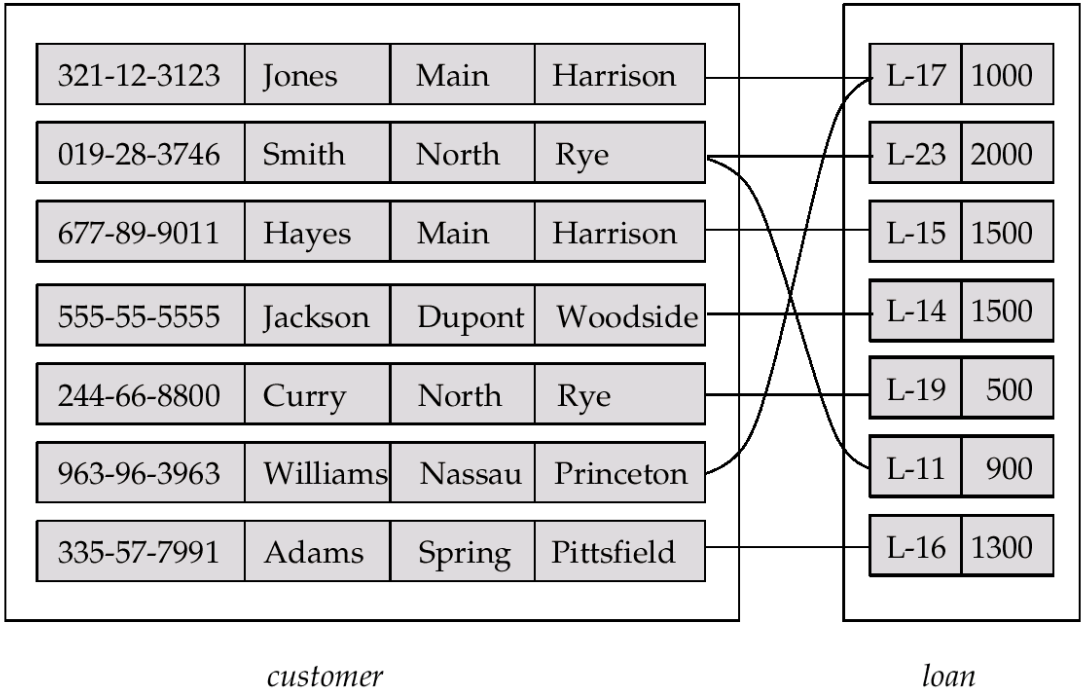
- **A relationship set** is a mathematical relation among $n \geq 2$ entities, each taken from entity sets

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

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

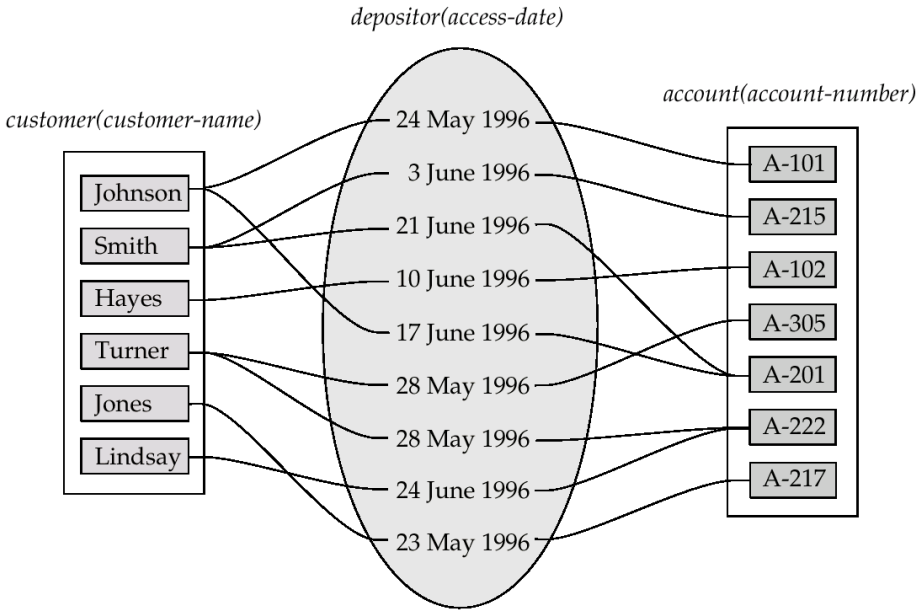
(Hayes, A-102) ∈ *depositor*

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

- The number of entity sets that participate in a relationship set
  - Relationship sets that involve two entity sets are **binary (二元的)**
  - Relationship sets may involve more than two entity sets
  - Relationships between more than two entity sets are rare, and most relationships are binary

Outline

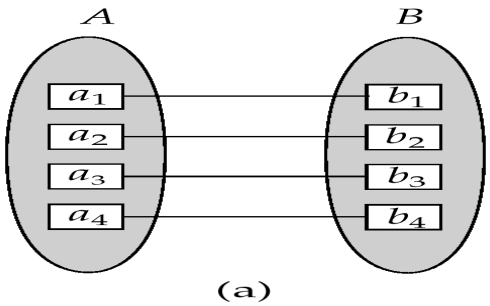
- Overview of the Design Process
- Entity-Relationship Model (**实体联系模型**)
- ☞ **Constraints**
- Entity-Relationship Diagrams (**实体联系图**)
- Reduction to Relation Schemas
- Summary

Mapping Cardinalities (映射基数)

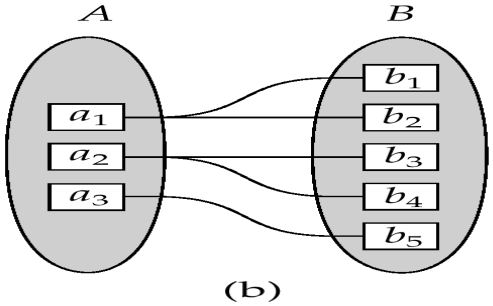
- Express the number of entities to which another entity can be associated via a relationship set
- For a binary relationship set, the mapping cardinality must be one of the following types
  - **One to one (1对1):** An entity in *A* is associated with **at most one entity** in B, and an entity in B is associated with **at most one entity** in *A*
  - **One to many (1对多):** An entity in *A* is associated with **any number (zero or more) of entities** in B. An entity in B, however, can be associated with **at most one entity** in *A*
  - **Many to one (多对1):** An entity in *A* is associated with **at most one entity** in B. An entity in B, however, can be associated with **any number (zero or more) of entities** in *A*
  - **Many to many (多对多):** An entity in *A* is associated with **any number (zero or more) of entities** in B, and an entity in B is associated with **any number (zero or more) of entities** in *A*

Mapping Cardinalities (cont.)

- Mapping cardinality types:
  - One to one(1对1): An entity in A is associated with at most one entity in B, and an entity in B is associated with at most one entity in A.
  - One to many(1对多): An entity in A is associated with any number (zero or more) of entities in B. An entity in B, however, can be associated with at most one entity in A.



One to one

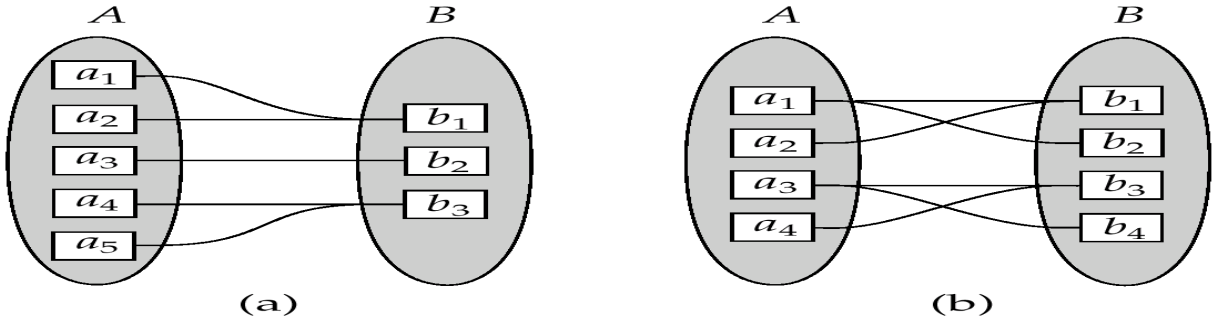


One to Many

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

Mapping Cardinalities (cont.)

- Mapping cardinality types:
  - Many to one(多对1): An entity in A is associated with at most one entity in B. An entity in B, however, can be associated with any number (zero or more) of entities in A.
  - Many to many(多对多): An entity in A is associated with any number (zero or more) of entities in B, and an entity in B is associated with any number (zero or more) of entities in A.

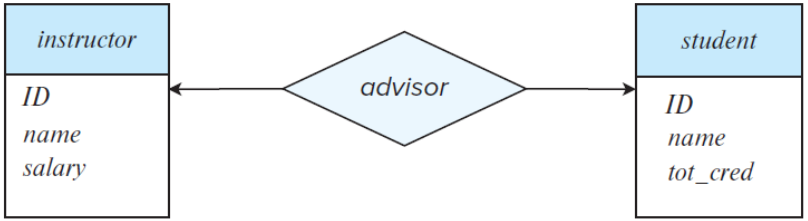


Many to one

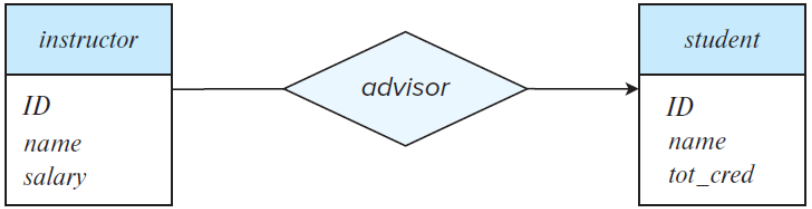
Many to many

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

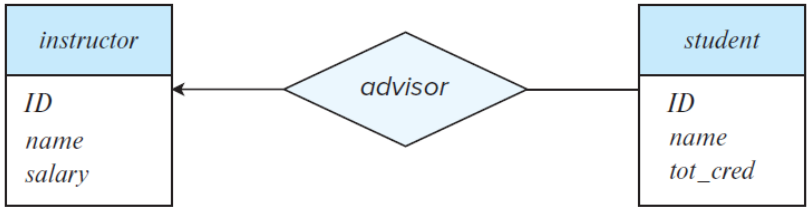
Mapping Cardinalities (cont.)



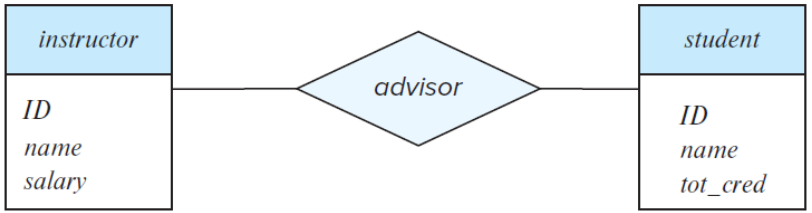
(a) One-to-one



(c) Many-to-one



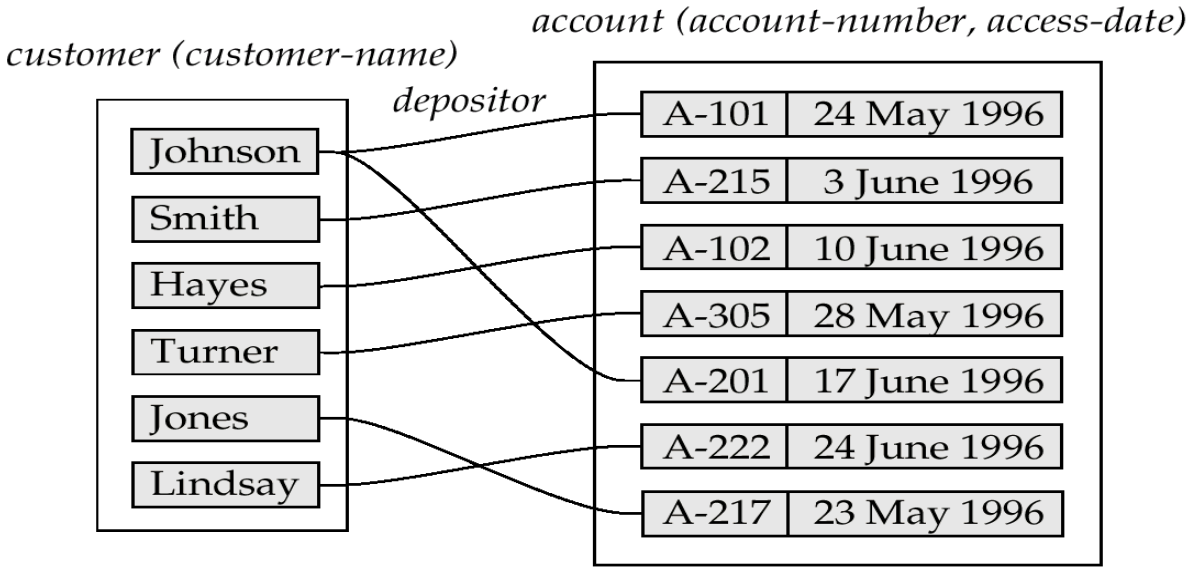
(b) One-to-many



(d) Many-to-many

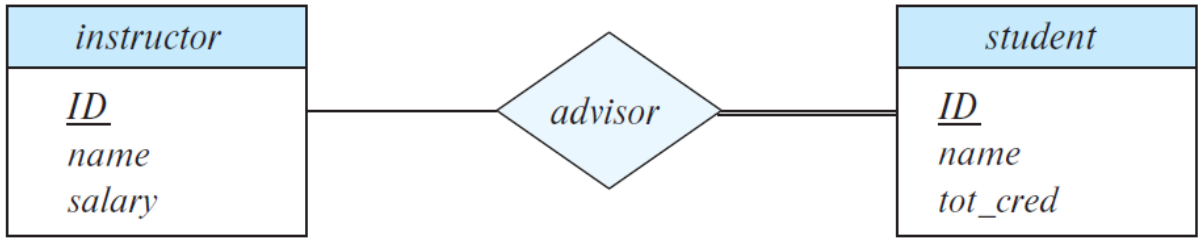
Mapping Cardinalities affect ER Design

- Can make access-date an attribute of account, instead of a relationship attribute, if each account can have only one customer
  - Each account participates a relationshop, i.e., the number of relationships is the number of account



Participation Constraints (参与约束)

- **Total participation** (indicated by **double line**):
  - Every entity in the entity set participates in at least one relationship in the relationship set
  - E.g., 每个student实体通过advisor联系同至少一名教师相联系，student在联系集advisor中是全部参与
- **Partial participation**
  - Some entities may not participate in any relationship in the relationship set
  - E.g., 有的instructor可能不指导学生，所以instructor在联系集advisor中是部分参与

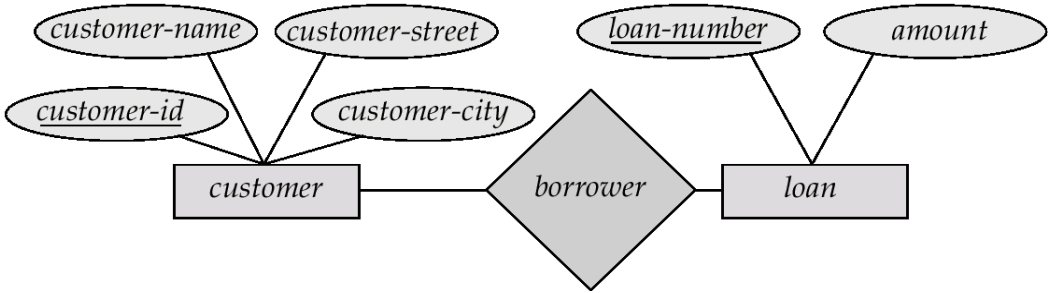


Outline

- Overview of the Design Process
- Entity-Relationship Model (**实体联系模型**)
- Constraints
- ☞ **Entity-Relationship Diagrams (实体联系图)**
- Reduction to Relation Schemas
- Summary

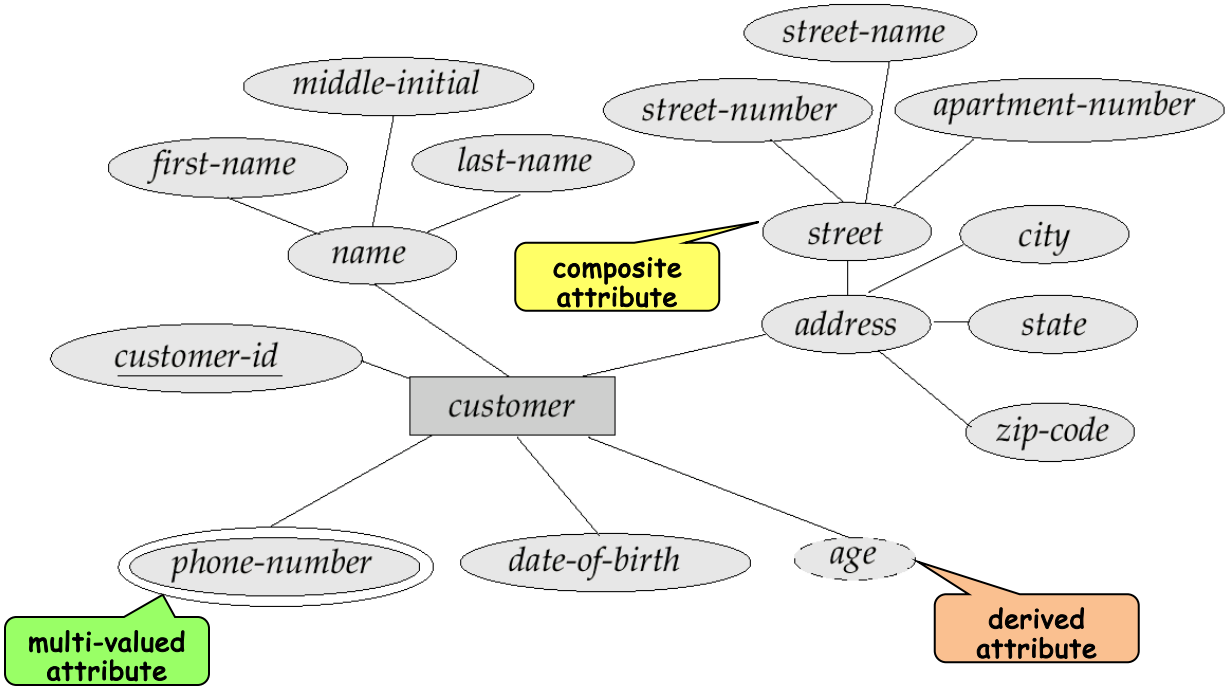
Entity-Relationship Diagrams

- **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**

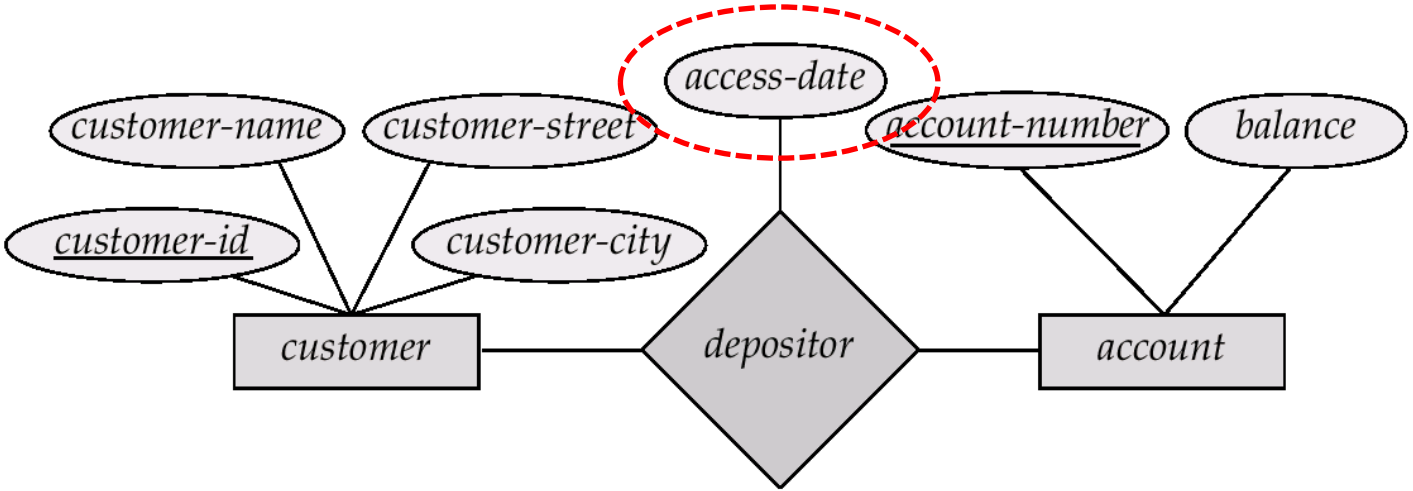


E-R Diagram

- E-R Diagram with composite, multivalued, and derived attributes

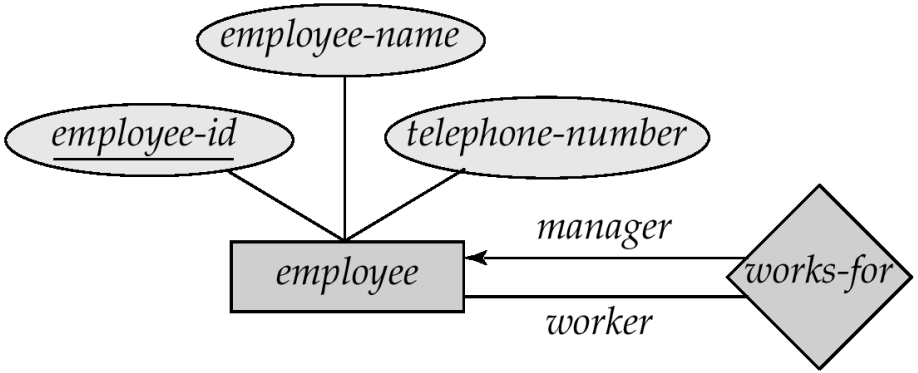


Relationship Sets with Attributes



Roles (角色)

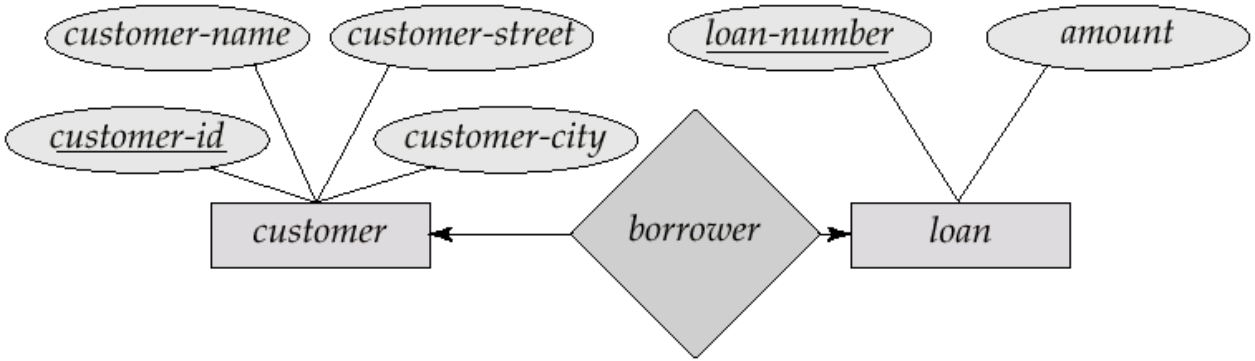
- **Entity sets of a relationship need not be distinct**
  - The labels "manager" and "worker" are called **roles**; they specify how employee entities interact via the works-for relationship set
  - Role labels are optional and used to clarify semantics of the relationship



Cardinality Constraints

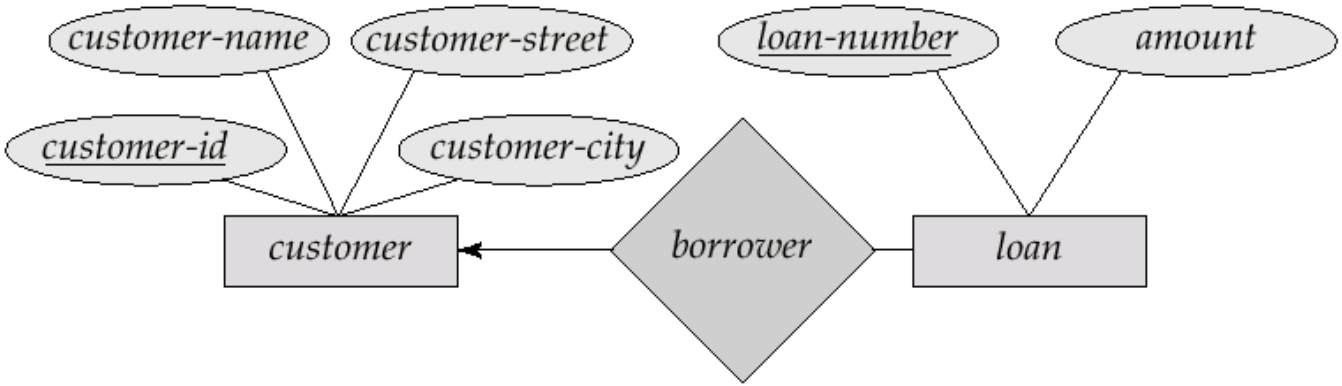
- We express cardinality constraints by drawing either **a directed line (→)**, signifying "**one**," or **an undirected line (—)**, signifying "**many**," between the relationship set and the entity set.

- **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



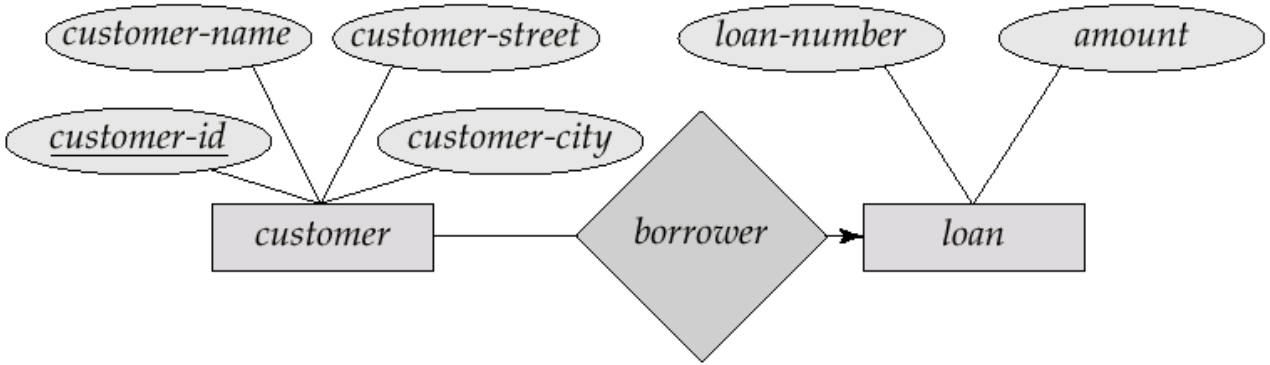
One-To-Many Relationship

- **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



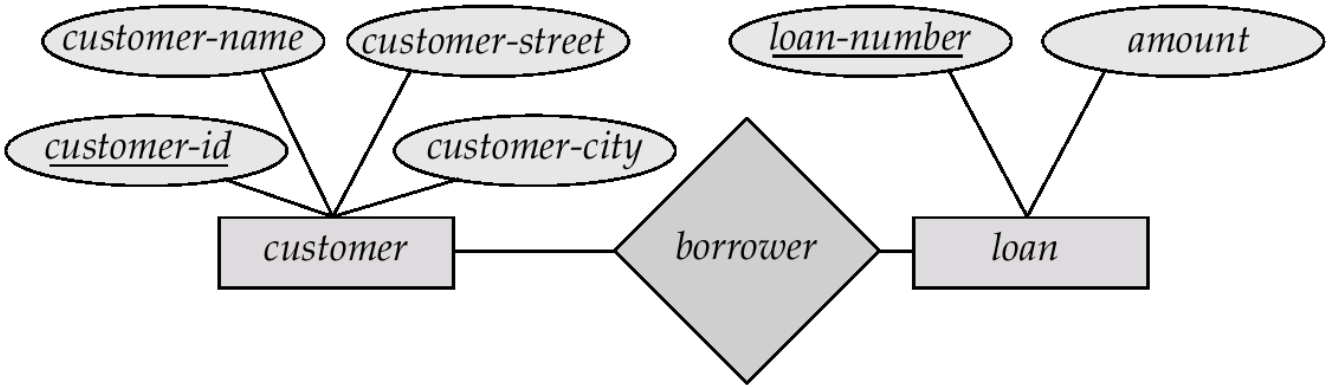
Many-To-One Relationships

- **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



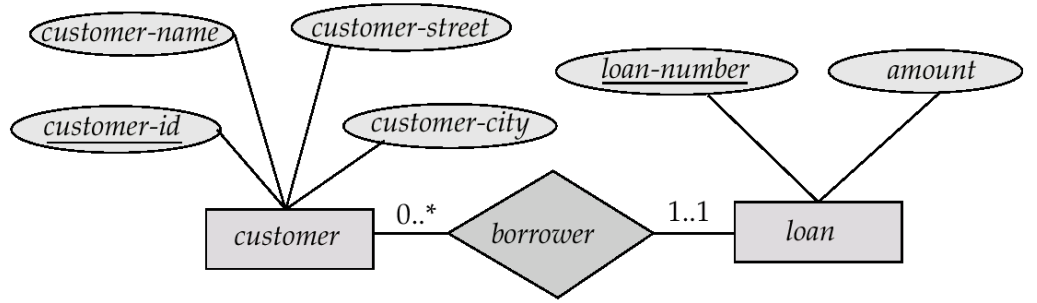
Many-To-Many Relationship

- **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

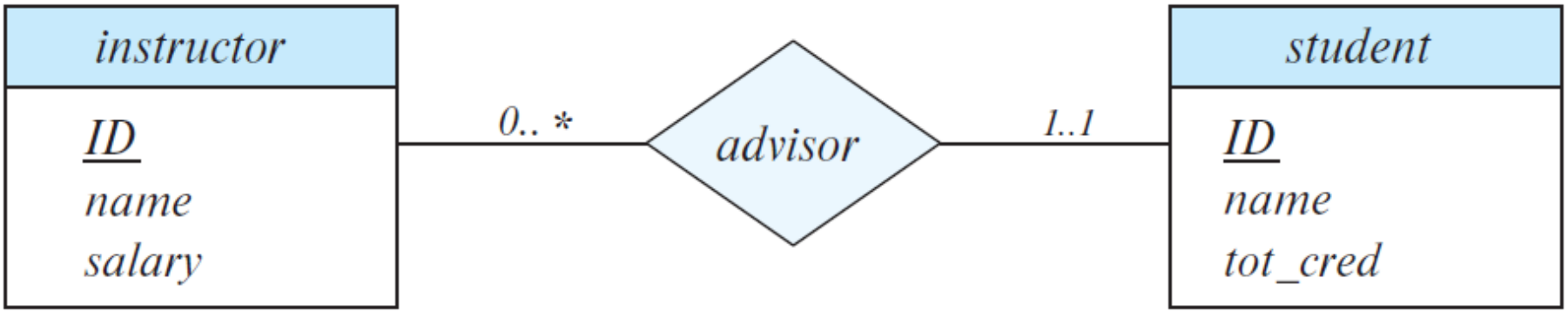


Alternative Notation for Cardinality Limits

- Cardinality limits can also express participation constraints
- Notation: ***m..n***, where ***m*** and ***n*** are the minimum and maximum cardinalities respectively
  - m=0: each entity may not participate the relationship set (partial participation)
  - m=1: each entity participates at least one relationship of the relationship set (full participation)
  - n=1: each entity participates at most one relationship of the relationship set
  - n=*: each entity participates many relationships of the relationship set



Alternative Notation for Cardinality Limits



Partial participation

One to many

Full participation

Keys (键/码)

- 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 a candidate key of customer
  - account_number is a 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 of Entity Sets

- We must to specify how entities within a given entity set and relationships within a given relationship set are distinguished
- A key for an entity is a set of attributes that suffice to distinguish entities from each other
- The concepts of superkey, candidate key, and primary key are applicable to entity sets just as they are applicable to relation schemas

Attributes for Relationship Sets

- Let R be a relationship set involving entity sets E_1, E_2,…, E_n. Let primary-key(E_i) denote the set of attributes that forms the primary key for entity set E_i. Assume all the primary-key(E_i) are unique
- If the relationship set R has no attributes associated with it, then the set of attributes describes an individual relationship in set R

$$primary\text{-}key(E_1) \cup primary\text{-}key(E_2) \cup \cdots \cup primary\text{-}key(E_n)$$

- If the relationship set R has attributes a1, a2,…, am associated with it, then the set of attributes describes an individual relationship in set R

$$primary\text{-}key(E_1) \cup primary\text{-}key(E_2) \cup \cdots \cup primary\text{-}key(E_n) \cup \{a_1, a_2, \ldots, a_m\}$$

- If the attribute names of primary keys are not unique across entity sets, the attributes are renamed to distinguish them

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
  - 1 to 1, 1 to many, many to 1, many to many
- Need to consider **semantics of relationship set** in selecting the primary key in case of more than one candidate key

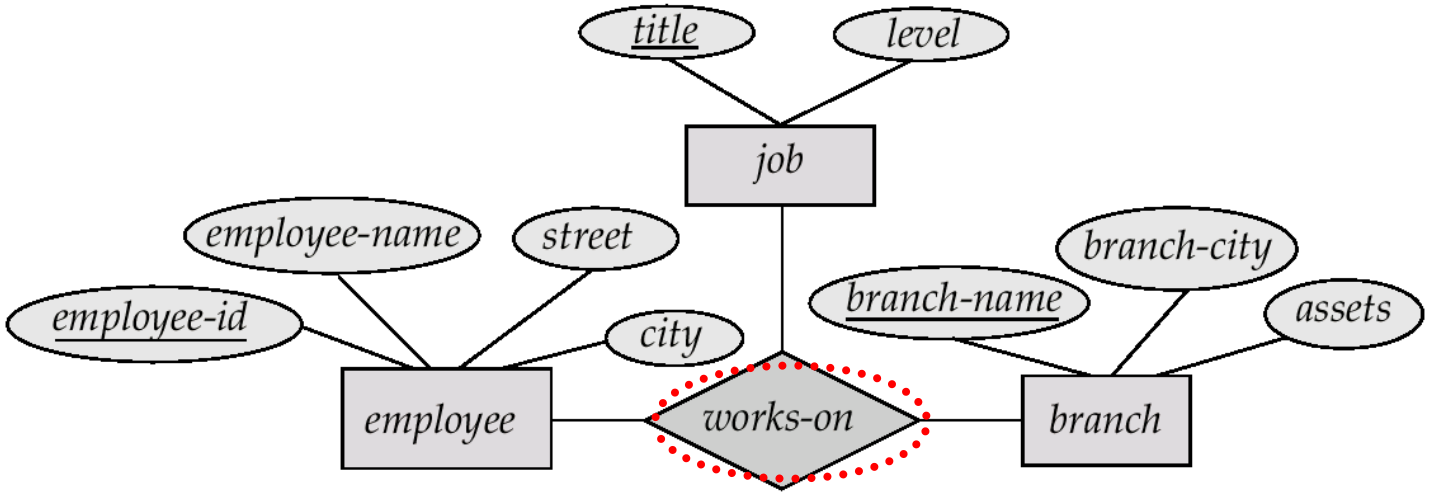
Keys for Relationship Sets

- Now consider binary relationships
  - For many-to-many relationships, the preceding union of the primary keys is a minimal superkey and is chosen as the primary key
  - For one-to-many and many-to-one relationships, the primary key of the "many" side is a minimal superkey and is used as the primary key
- For example, consider the entity sets instructor and student, and the relationship set advisor
  - If the relationship set is many-to-many, then the primary key of advisor consists of the union of the primary keys of instructor and student
  - if the relationship is many-to-one from student to instructor, then the primary key of advisor is simply the primary key of student

Keys for Relationship Sets

- For nonbinary relationships
  - If no cardinality constraints are present, then the superkey formed as described earlier is the only candidate key, and it is chosen as the primary key
  - If cardinality constraints are present, the choice of the primary key is more complicated
    - If only one arrow from the relationship set to an entity set, the combination of primary leys of entity sets with no arrow pointed forms the primary of the relationship set
    - If there are more arrows, there may different explanations, leading to different primary keys
      - A solution to this case is to transform the relationship set to a virtual entity set, and construct relationship set with each existing entity set

E-R Diagram with a Ternary Relationship



Cardinality Constraints on Ternary Relationship

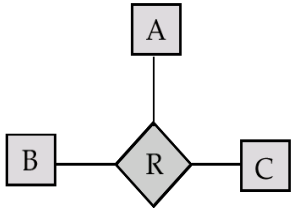
- We allow **at most one arrow out of a ternary relationship**
  - **E.g.,** an arrow from works-on to job indicates that each employee works on at most one job at any branch
- If there is more than one arrow, there are two ways of defining the meaning
  - E.g., a ternary relationship R between A, B and C with arrows to B and C
- **To avoid confusion we outlaw more than one arrow**

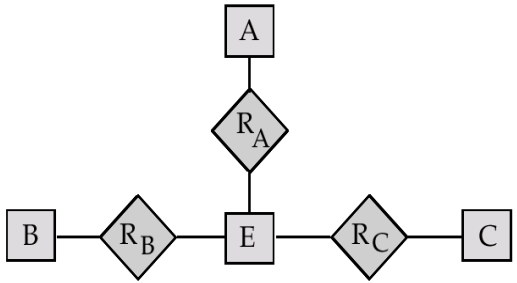
Binary vs. Non-Binary Relationships

- 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

- 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:
    - R_A, relating E and A
    - R_B, relating E and B
    - R_C, relating E and C
  - Create a special identifying attribute for **E**, and add any attributes of **R** to **E**
  - For each relationship (a_i , b_i , c_i) in **R**, create
    - add a new entity e_i in the entity set E
    - add (e_i, a_i) to RA
    - add (e_i, b_i) to RB
    - add (e_i, c_i) to RC





Converting Non-Binary Relationships (Cont.)

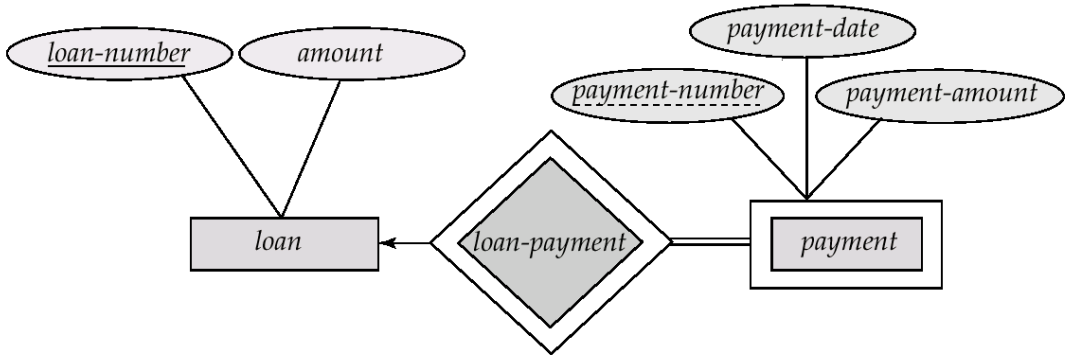
- **Translate constraints**
  - Translating all constraints may not be possible
  - There may be instances in the translated schema that cannot correspond to any instance of R
  - We can avoid creating an identifying attribute by making E a weak entity set (described shortly) identified by the three relationship sets

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, 分辨属性)**
- The primary key of a weak entity set
  - **Discriminator plus primary keys of identifying entity sets**

Weak Entity Sets (Cont.)

- Depict a weak entity set by **double rectangles**.
- Underline the **discriminator (分辨属性)** 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

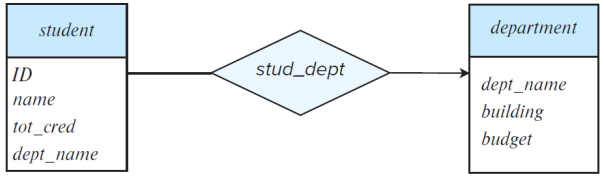
- If loan_number were explicitly stored, payment could be made a strong entity

More Weak Entity Set Examples

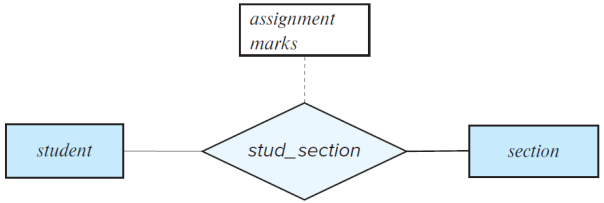
- In a university, a course is a strong entity and a course_offering can be modeled as a weak entity
- The discriminator of course_offering would be semester (including year) and section_number (if there is more than one section)
- If we model course_offering as a strong entity we would model course_number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

ER Design Issues

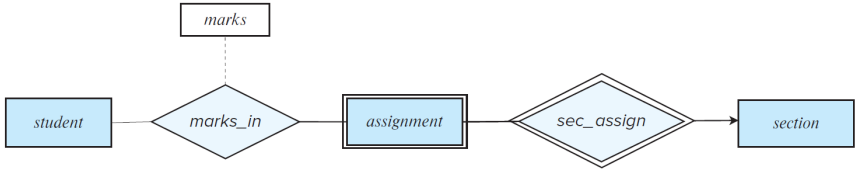
- **Common mistakes in E-R Diagrams**
  - Using the primary key of an entity set as an attribute of another entity set, instead of using a relationship
  - Designating the primary key attributes of the related entity sets as attributes of the relationship set
  - Using a relationship with a single-valued attribute in a situation that requires a multivalued attribute



(a) Incorrect use of attribute



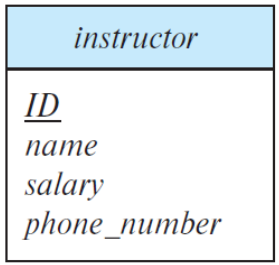
(b) Erroneous use of relationship attributes



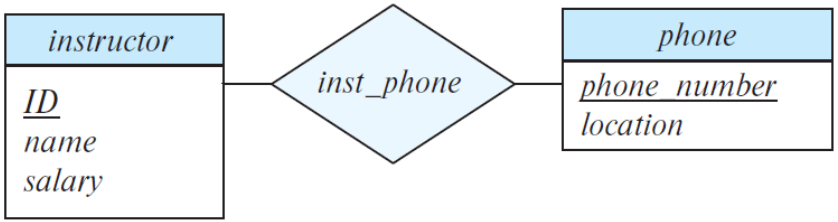
(c) Correct alternative to erroneous E-R diagram (b)

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
  - Entities is more complex and general than attributes
  - General, attributes are atomic



(a)



(b)

ER Design Issues

- Use of entity sets vs. relationship sets
  - Possible guideline is to designate a relationship set to describe an action that occurs between entities

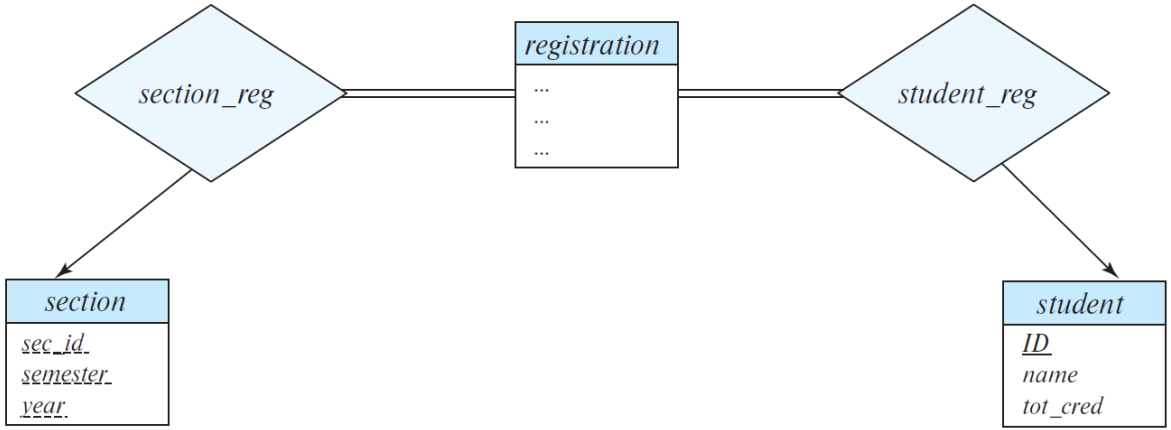


Figure 6.24 Replacement of *takes* by *registration* and two relationship sets.

ER Design Issues

- **Binary versus n-ary relationship sets**
  - Some relationships that appear to be nonbinary could actually be better represented by several binary relationships

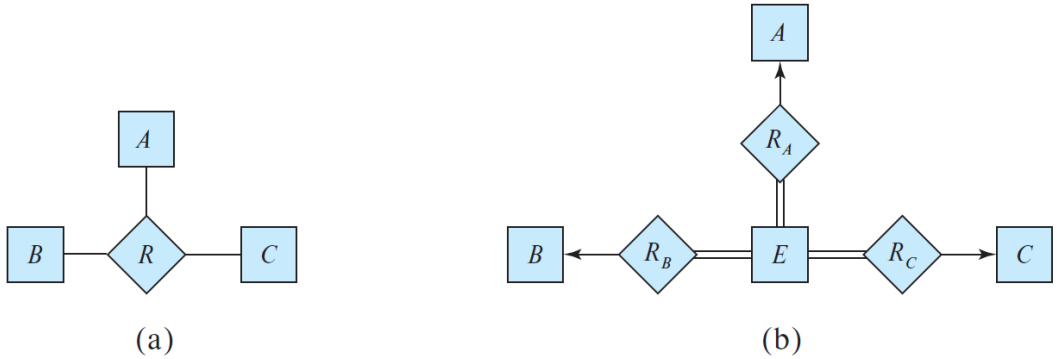
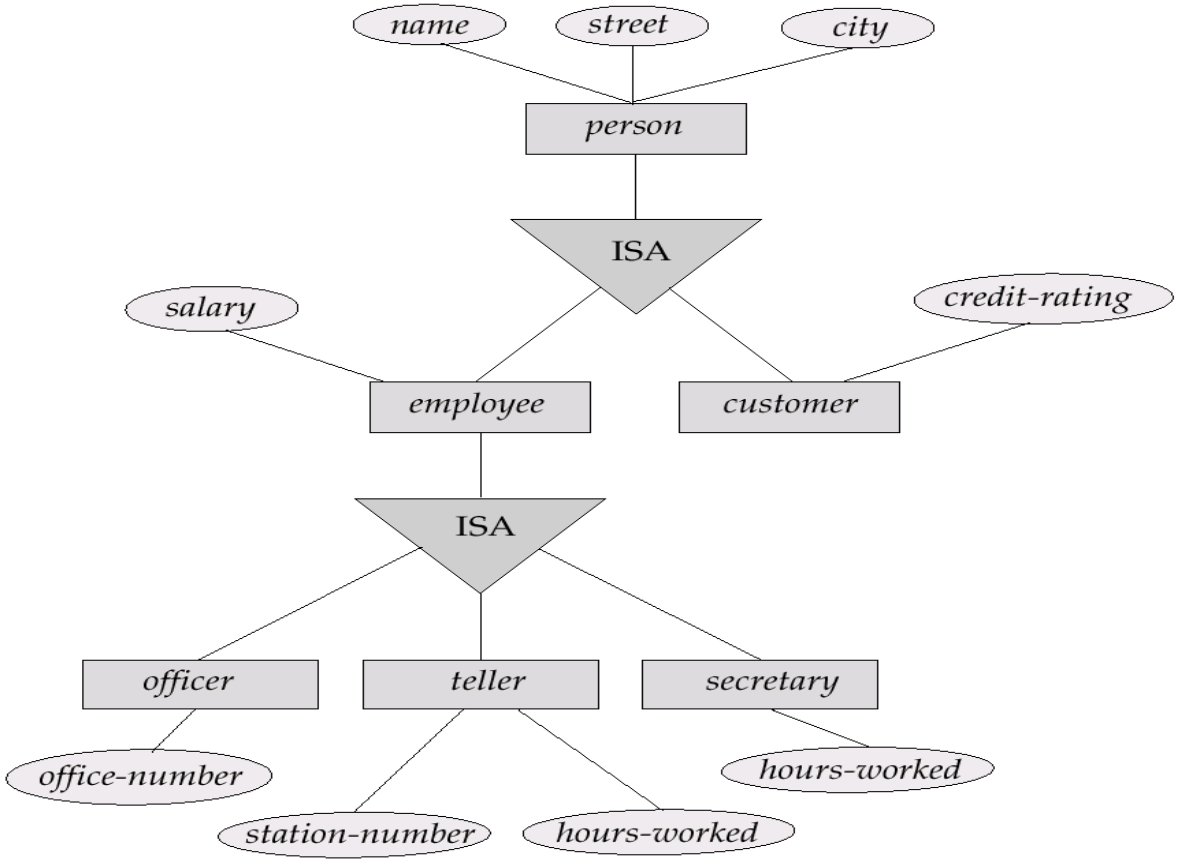


Figure 6.25 Ternary relationship versus three binary relationships.

Specialization（特化）

- Top-down design process (自上而下的设计过程)
  - 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（属性继承）
  - A lower-level entity set inherits all the attributes and relationship participation of the higher-level entity set to which it is linked

Example


name
street
city
person
ISA
salary
credit-rating
employee
customer
ISA
officer
teller
secretary
office-number
station-number
hours-worked
hours-worked

Generalization（泛化）

- A bottom-up design process (自下而上的设计过程)
  - Combine a number of entity sets that share the same features into a higher-level entity set

- Specialization (特化), Generalization (泛化)
  - Specialization and generalization are simple inversions of each other
  - They are represented in the same way in an E-R diagram

Specialization & Generalization (Cont.)

- 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

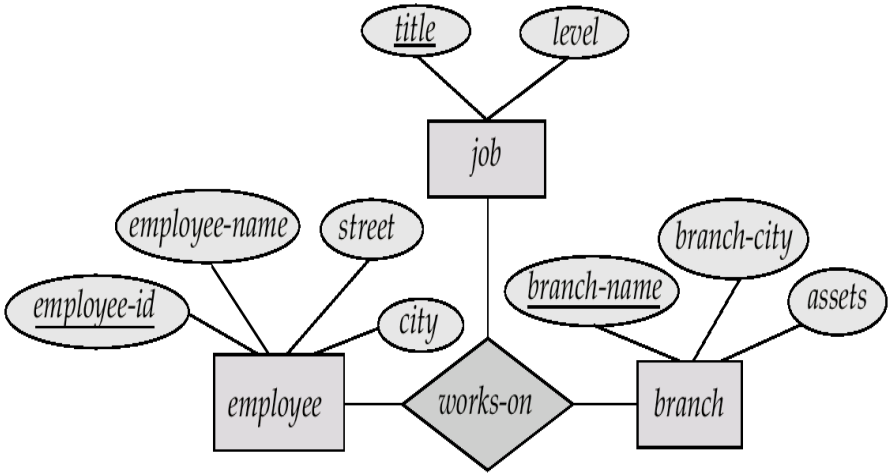
- Constraint on which entities can be members of a given lower-level entity set
  - condition-defined (attribute-defined)
  - user-defined
- Constraint on whether or not entities may belong to more than one lower-level entity set within a single generalization
  - Disjoint
  - Overlapping

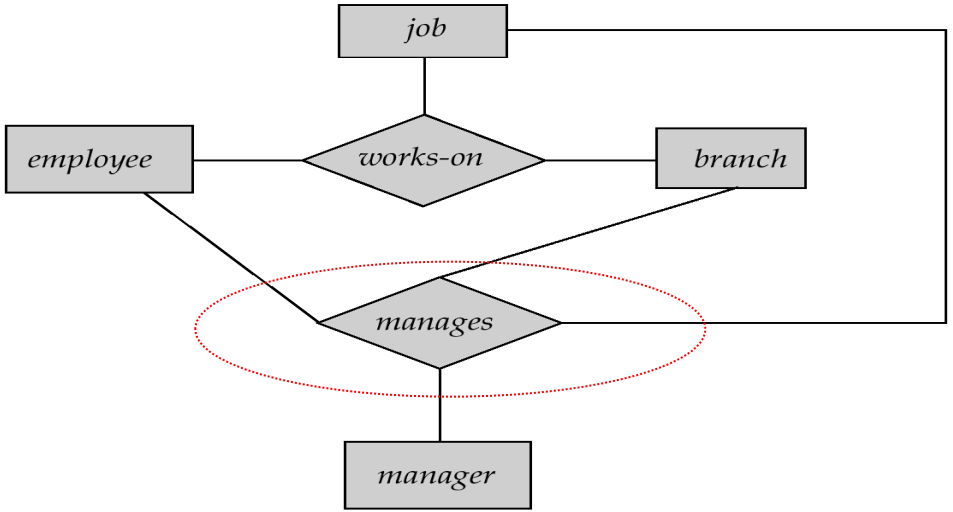
Design Constraints on a Specialization/Generalization (Cont.)

- 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 (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

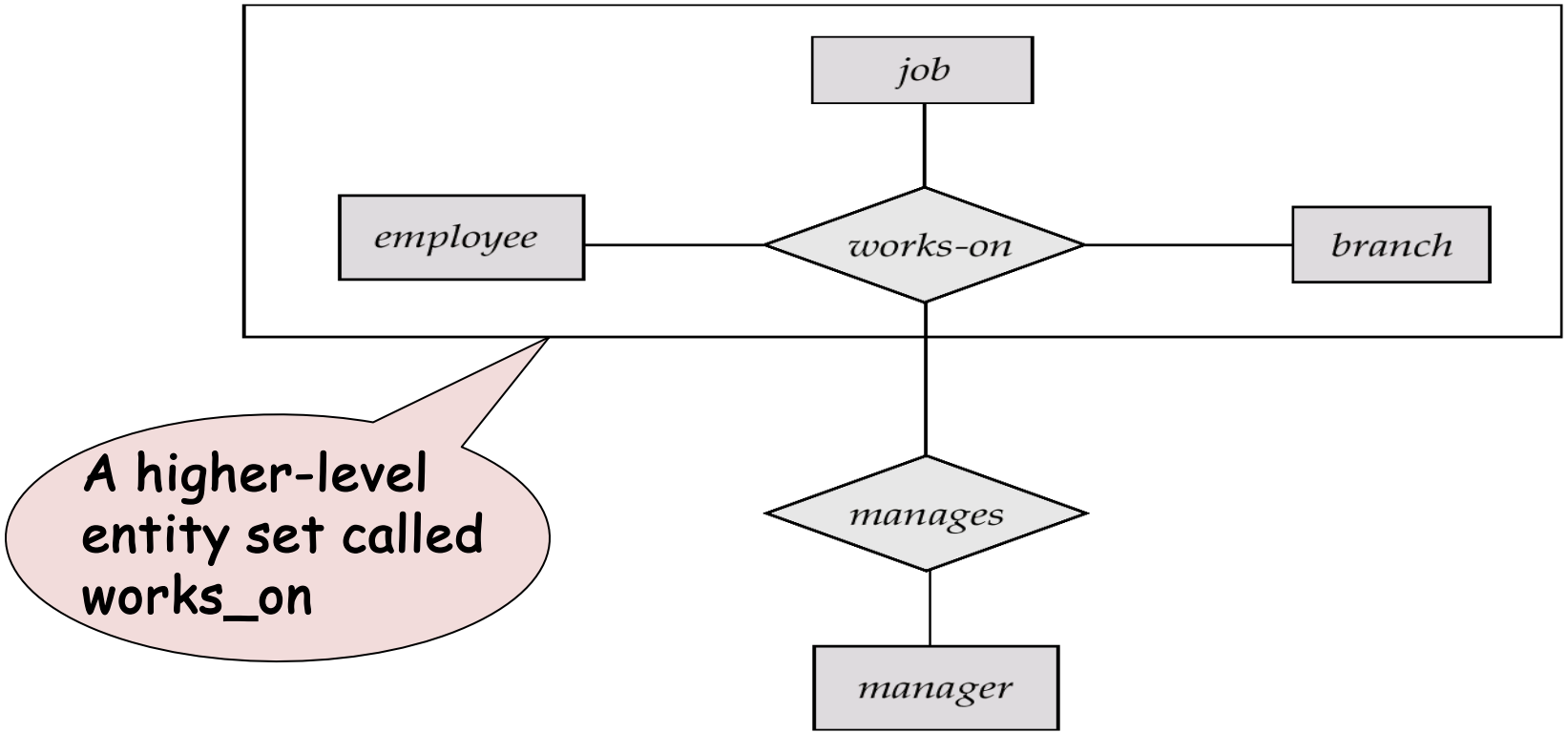




Aggregation (Cont.)

- 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



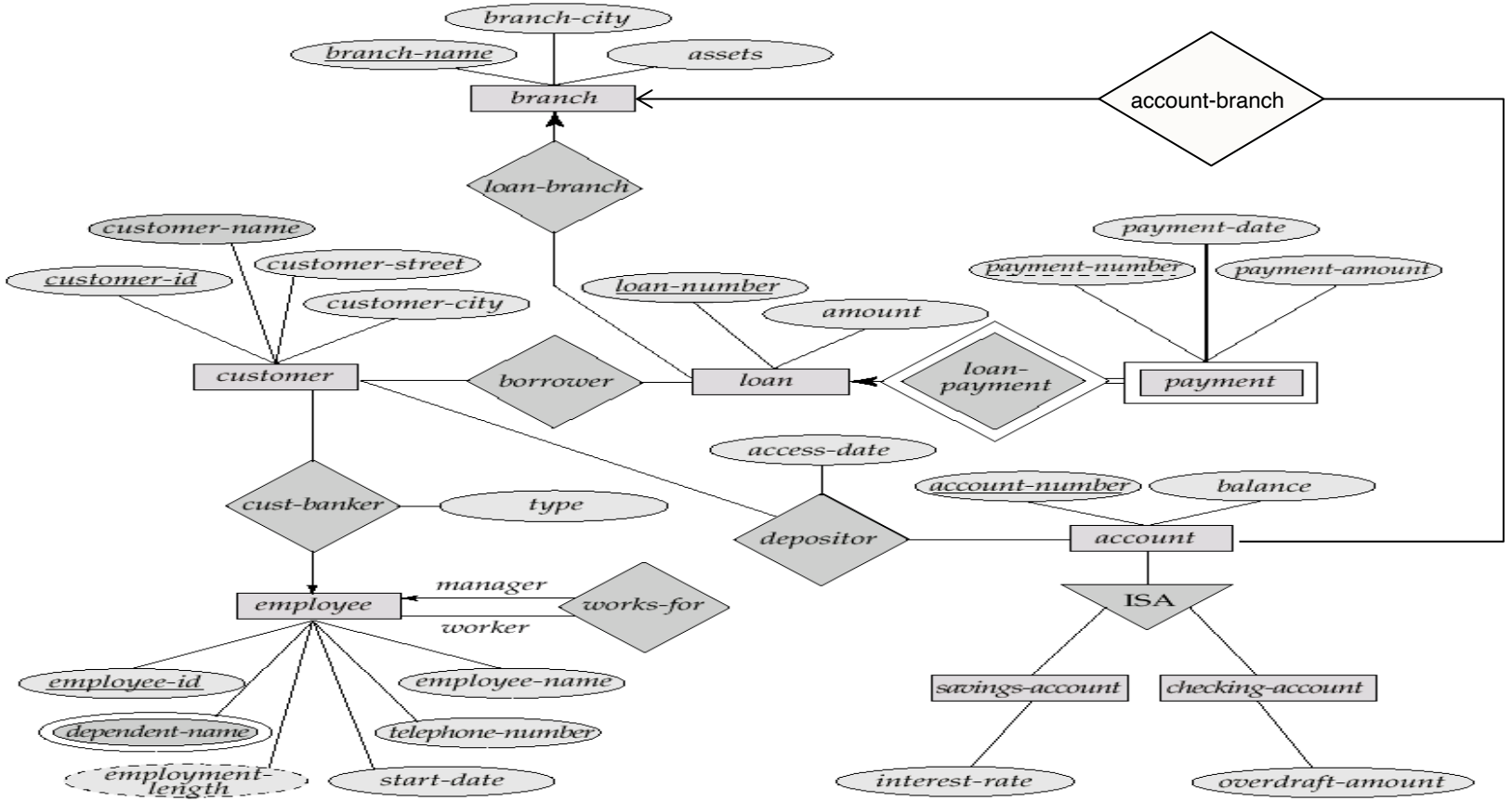
E-R Design Decisions

- 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

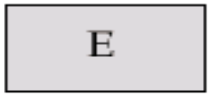
Database Design Phases

- 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

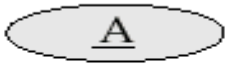


Entity Set

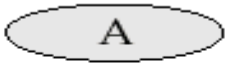
Weak Entity Set

Relationship Set

Identifying Relationship Set for Weak Entity Set



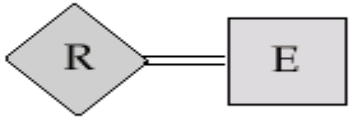
Primary Key



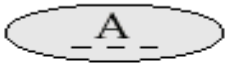
Attribute

Multivalued Attribute

Derived Attribute

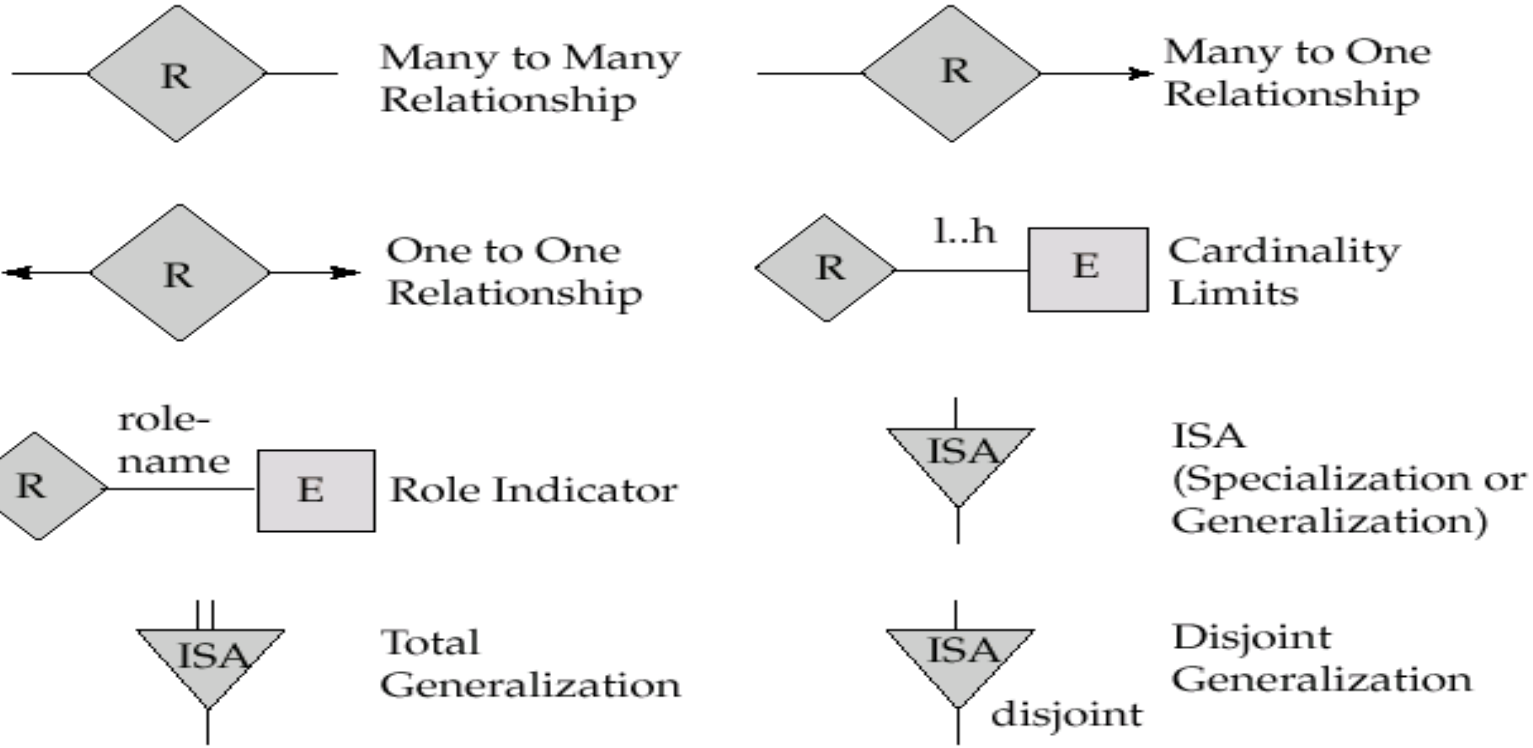


Total Participation of Entity Set in Relationship



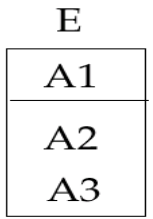
Discriminating Attribute of Weak Entity Set

Summary of Symbols (Cont.)

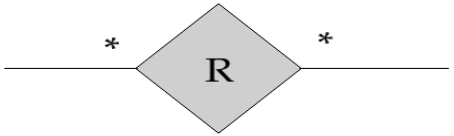


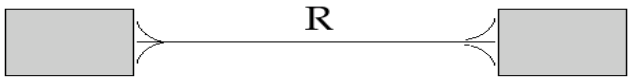
Alternative E-R Notations

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

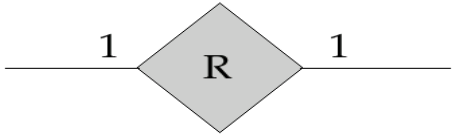


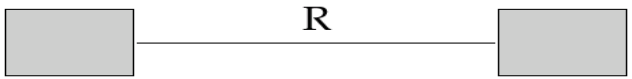
Many to Many Relationship



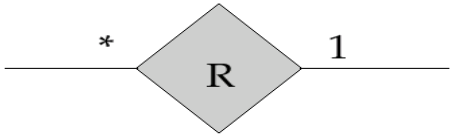


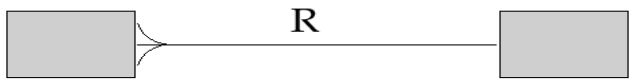
One to One Relationship





Many to One Relationship





Outline

- Overview of the Design Process
- Entity-Relationship Model (**实体联系模型**)
- Constraints
- Entity-Relationship Diagrams (**实体联系图**)
- ☞ **Reduction to Relation Schemas**
- Summary

Reduction to Relational Schemas

- Reduction of an E-R Diagram to Tables
  - 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

- Primary keys allow entity sets and relationship sets to be expressed uniformly as tables which represent the contents of the database

Representing Entity Sets as Tables

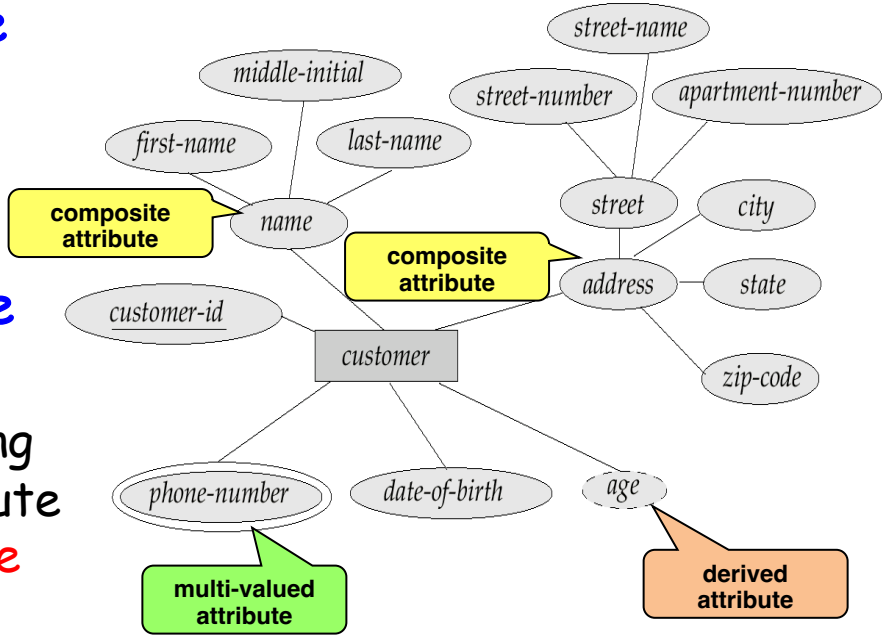
- **A strong entity set** reduces to a table with the same attributes

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

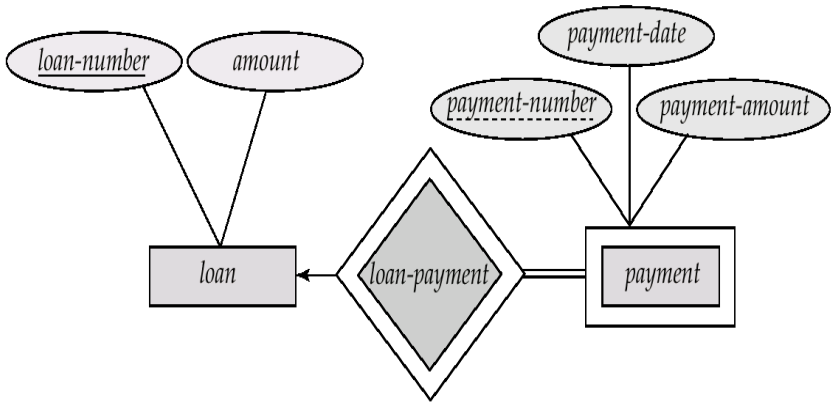
- Composite attributes are flattened out by creating a separate attribute for each component attribute

- 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 (primary key of EM? Full key)
  - 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

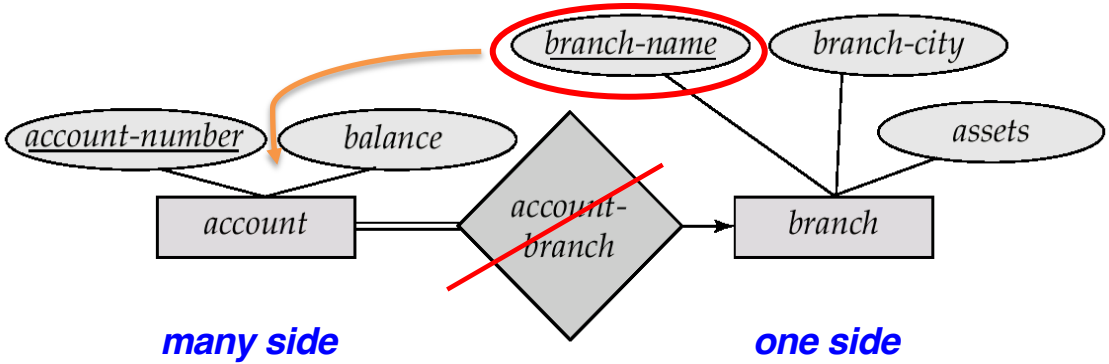
Representing Relationship Sets as Tables

- **Many-to-many relationship set**
  - 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

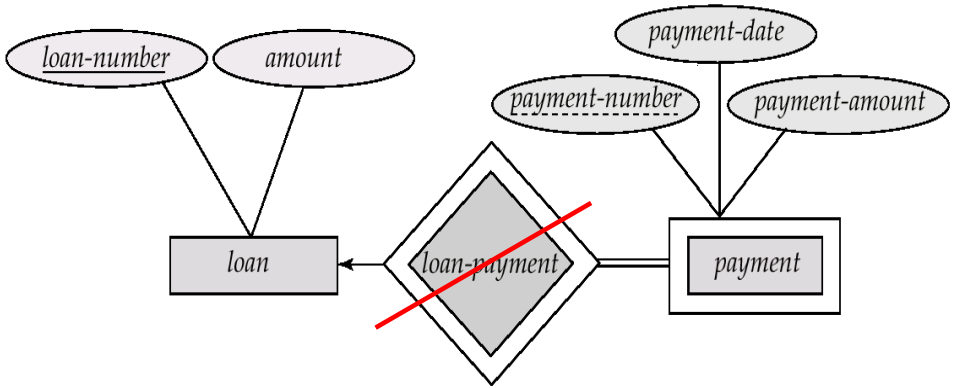
Representing Relationship Sets as Tables

- Many-to-one and one-to-many relationship sets
  - 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



Representing Relationship Sets as Tables

- 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:** Querying information about entities, e.g., employee, requires accessing two tables

Representing Specialization as Tables (Cont.)

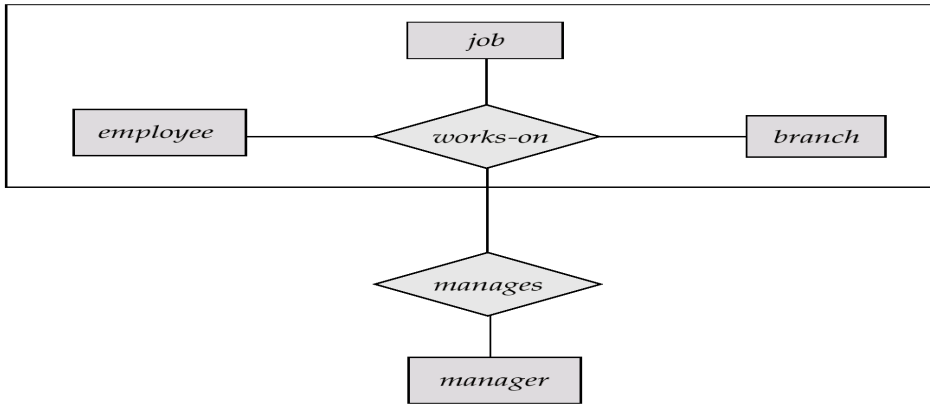
- **Method 2:**
  - Form a table for each entity set with all local and inherited attributes

table	table attributes
person	***name, street, city***
customer	***name, street, city,*** *credit-rating*
employee	***name, street, city,*** *salary*

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

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

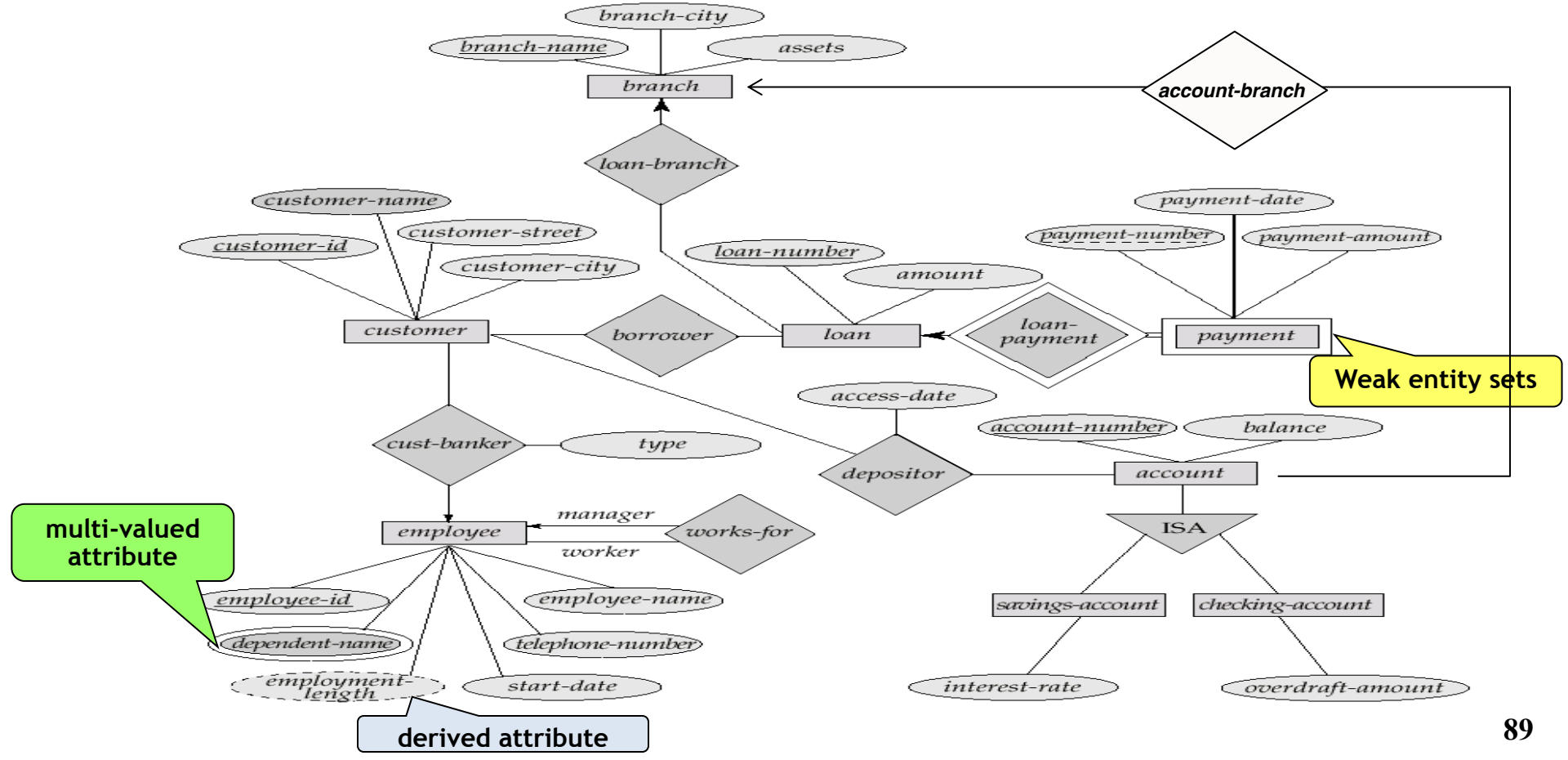


- E.g. to represent aggregation manages between relationship works-on and entity set manager, create a table

  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

E-R Diagram for a Banking Enterprise



The Banking Schema

- *branch* = (*branch_name*, *branch_city*, *assets*)
- *customer* = (*customer_id*, *customer_name*, *customer_street*, *customer_city*)
- *loan* = (*loan_number*, *amount*)
- *account* = (*account_number*, *balance*)

- *employee* = (*employee_id*, *employee_name*, *telephone_number*, *start_date*) (derived attribute not included)

- *dependent_name* = (*employee_id*, *dname*) (derived from a multivalued attribute)

- *account_branch* = (*account_number*, *branch_name*) (many to one)
- *loan_branch* = (*loan_number*, *branch_name*) (many to one)
- *cust_banker* = (*customer_id*, *employee_id*, *type*) (many to one)
- *borrower* = (*customer_id*, *loan_number*) (many to many)
- *depositor* = (*customer_id*, *account_number*, *access_date*) (many to many)
- *works_for* = (*worker_employee_id*, *manager_employee_id*)

- *payment* =(*loan_number*,*payment_number*,*payment_date*,*payment_amount*) (weak entity set)

- *savings_account* = (*account_number*, *interest_rate*) (ISA specialization)
- *checking_account* = (*account_number*, *overdraft_amount*) (ISA specialization)

The Banking Schema

- *branch* = (*branch_name*, branch_city, assets)
- *customer* = (*customer_id*, customer_name, customer_street, customer_city)
- *loan* = (*loan_number*, amount)
- *account* = (*account_number*, balance)

loan = (*loan_number*, *branch_name*, amount)
account = (*account_number*, *branch_name*, balance)

- *employee* = (*employee_id*, employee_name, telephone_number, start_date) (derived attribute not included)

- *dependent_name* = (*employee_id*, *dname*) (derived from a multivalued attribute)

- *account_branch* = (*account_number*, *branch_name*) (many to one)
- *loan_branch* = (*loan_number*, *branch_name*) (many to one)
- *cust_banker* = (*customer_id*, *employee_id*, type) (many to one)
- *borrower* = (*customer_id*, *loan_number*) (many to many)
- *depositor* = (*customer_id*, *account_number*, access_date) (many to many)
- *works_for* = (*worker_employee_id*, *manager_employee_id*)

- *payment* =(*loan_number*,*payment_number*,payment_date,payment_amount) (weak entity set)

- *savings_account* = (*account_number*, interest_rate) (ISA specialization)
- *checking_account* = (*account_number*, overdraft_amount) (ISA specialization)

Outline

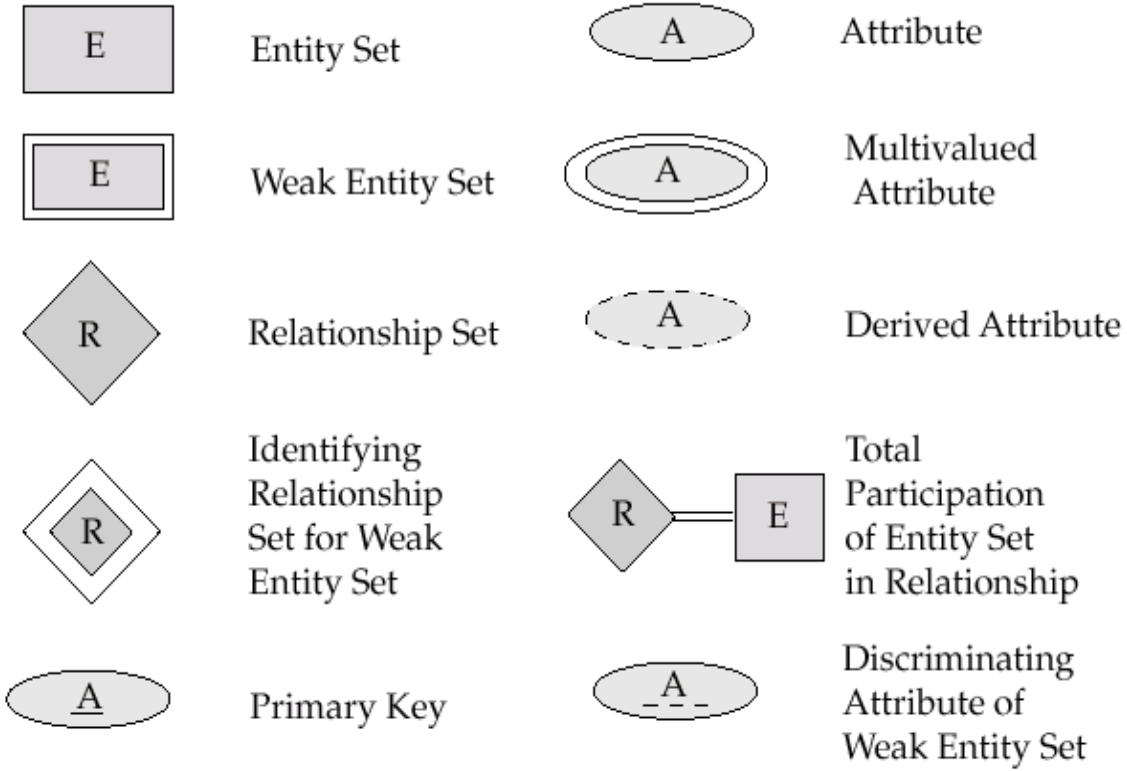
- Overview of the Design Process
- Entity-Relationship Model (**实体联系模型**)
- Constraints
- Entity-Relationship Diagrams (**实体联系图**)
- Reduction to Relation Schemas

☞ **Summary**

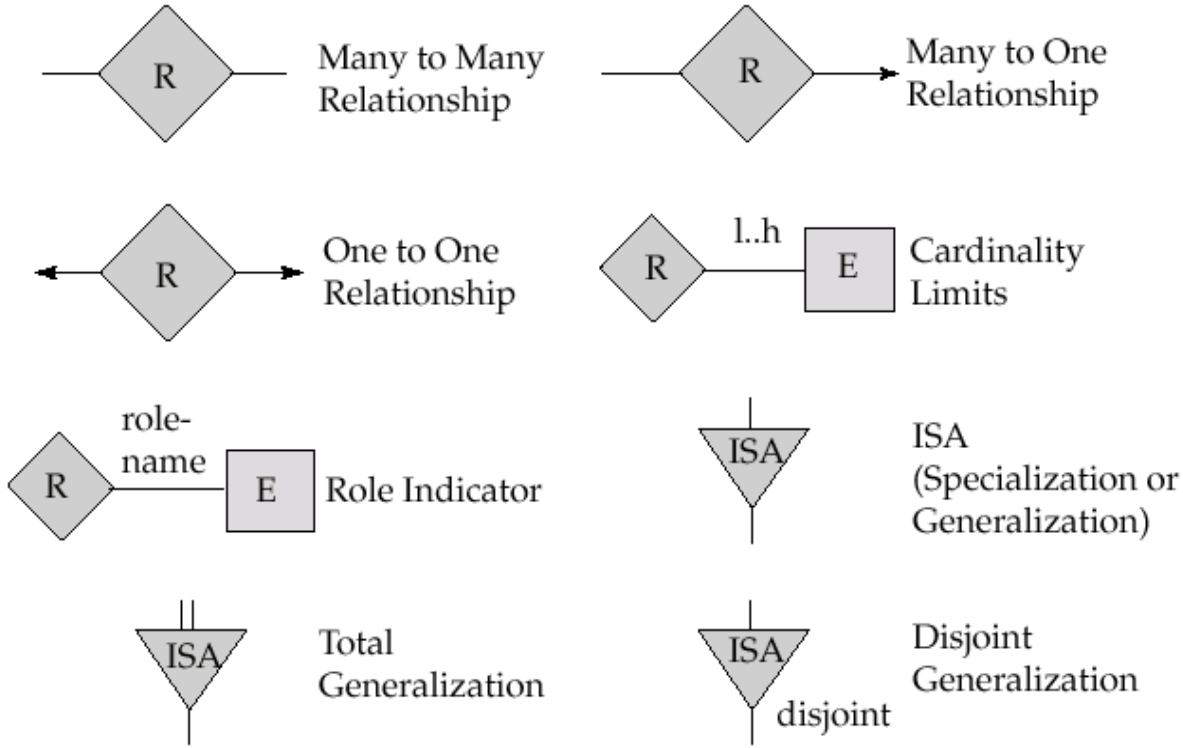
Database Design Phases

- **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
  - E.g., relational model, OO model
- **Physical Implementation**
  - Specify physical features of the database
  - buffer size, index…

Symbols Used in E-R Diagrams



Symbols Used in E-R Diagrams (Cont.)



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
- **E-R model** is 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
- Note: There are many variations on ER model

Summary of ER Model (Cont.)

- **Integrity constraints** in E-R 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 E-R model
  - Constraints play an important role in determining the best database design for an enterprise

Summary of ER Model

- 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

- Ensuring good database design
  - The generated relational schema should be analyzed and refined further
  - FD information and normalization techniques are especially useful (See Chp. 7 of the text book)

Assignment

- Select two the following three to finish
  - 6.22, 6.23, 6.24

- Submission DDL
  - 12:00pm, March 26, 2025

End of Lecture 5