有道翻译 支持免费文档翻译240万字符/年 立即下载→ https://fanyi.youdao.com/download

数据库设计与E-R模型

关继红教授
电子邮件:jhguan@tongji.edu.cn
计算机科学与技术系
同济大学

课程大纲

- **第0部分:概述**
  - Ch1:介绍
- **Part 1关系数据库**
  - Ch2:关系模型(数据模型，关系代数)
  - Ch3&4: SQL(结构化查询语言)
  - Ch5:高级SQL

第2部分:数据库设计

- **Ch6基于E-R模型的数据库设计**
- Ch7:关系型数据库设计

- **第三部分:应用程序设计与开发**
  - Ch8:复杂数据类型
  - Ch9:应用开发
- **Part 4大数据分析**
  - Ch10:大数据
  - Ch11:数据分析

- **第5部分:数据存储和索引**
  - Ch12:物理存储系统
  - Ch13:数据存储结构
  - Ch14:索引
- **第6部分:查询处理与优化**
  - Ch15:查询处理
  - Ch16:查询优化
- **第7部分事务管理**
  - Ch17:交易
  - Ch18:并发控制
  - Ch19:恢复系统
- **第8部分:并行和分布式数据库**
  - Ch20:数据库系统架构
  - Ch21-23:并行和分布式存储，查询处理和事务处理
- **第9部分**
  - **DB平台:OceanBase、MongoDB、Neo4J**

大学数据库

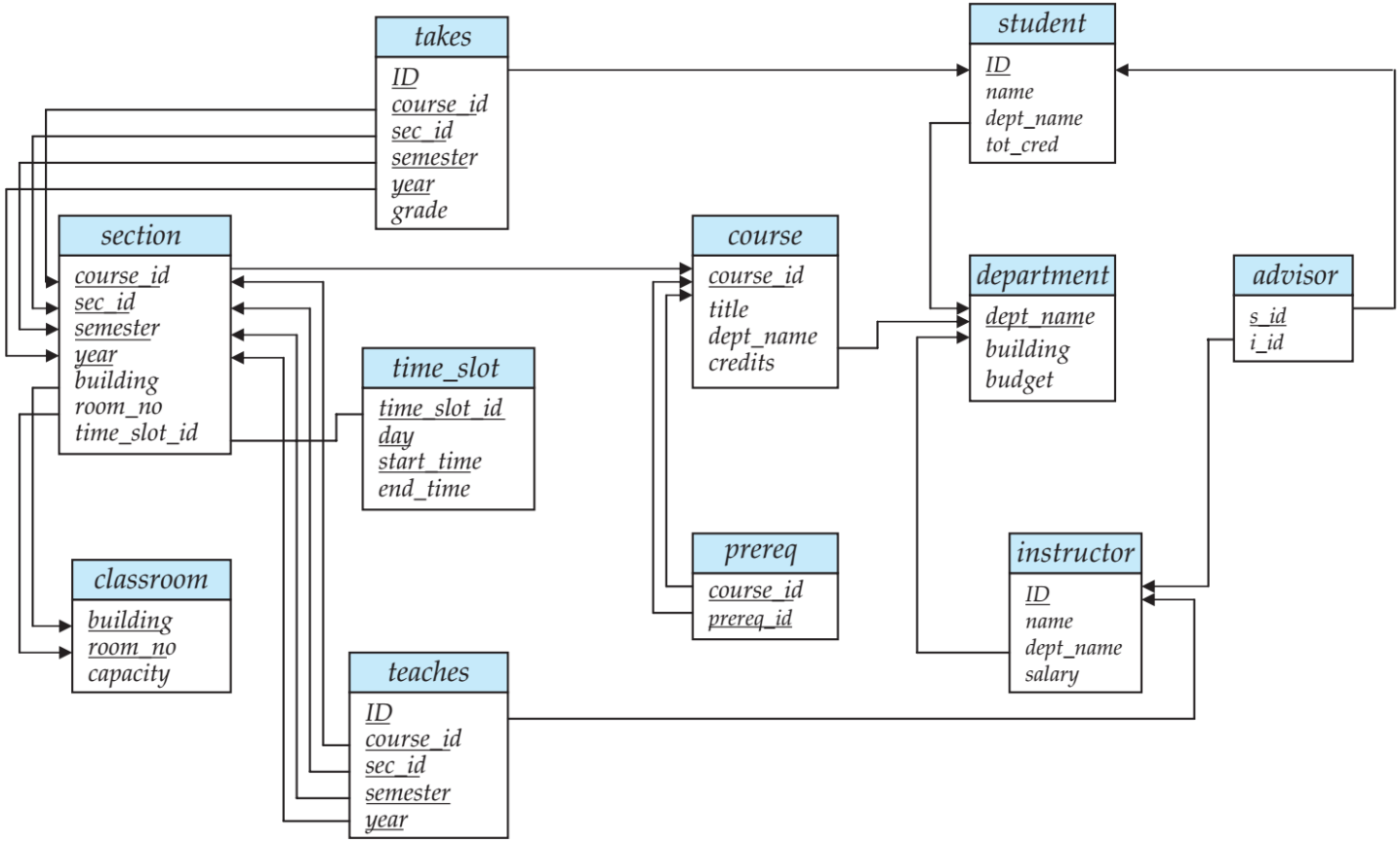
ID	*name*	*dept_name*	*salary*
22222	Einstein	Physics	95000
12121	Wu	Finance	90000
32343	El Said	History	60000
45565	Katz	Comp. Sci.	75000
98345	Kim	Elec. Eng.	80000
76766	Crick	Biology	72000
10101	Srinivasan	Comp. Sci.	65000
58583	Califieri	History	62000
83821	Brandt	Comp. Sci.	92000
15151	Mozart	Music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

教练表

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

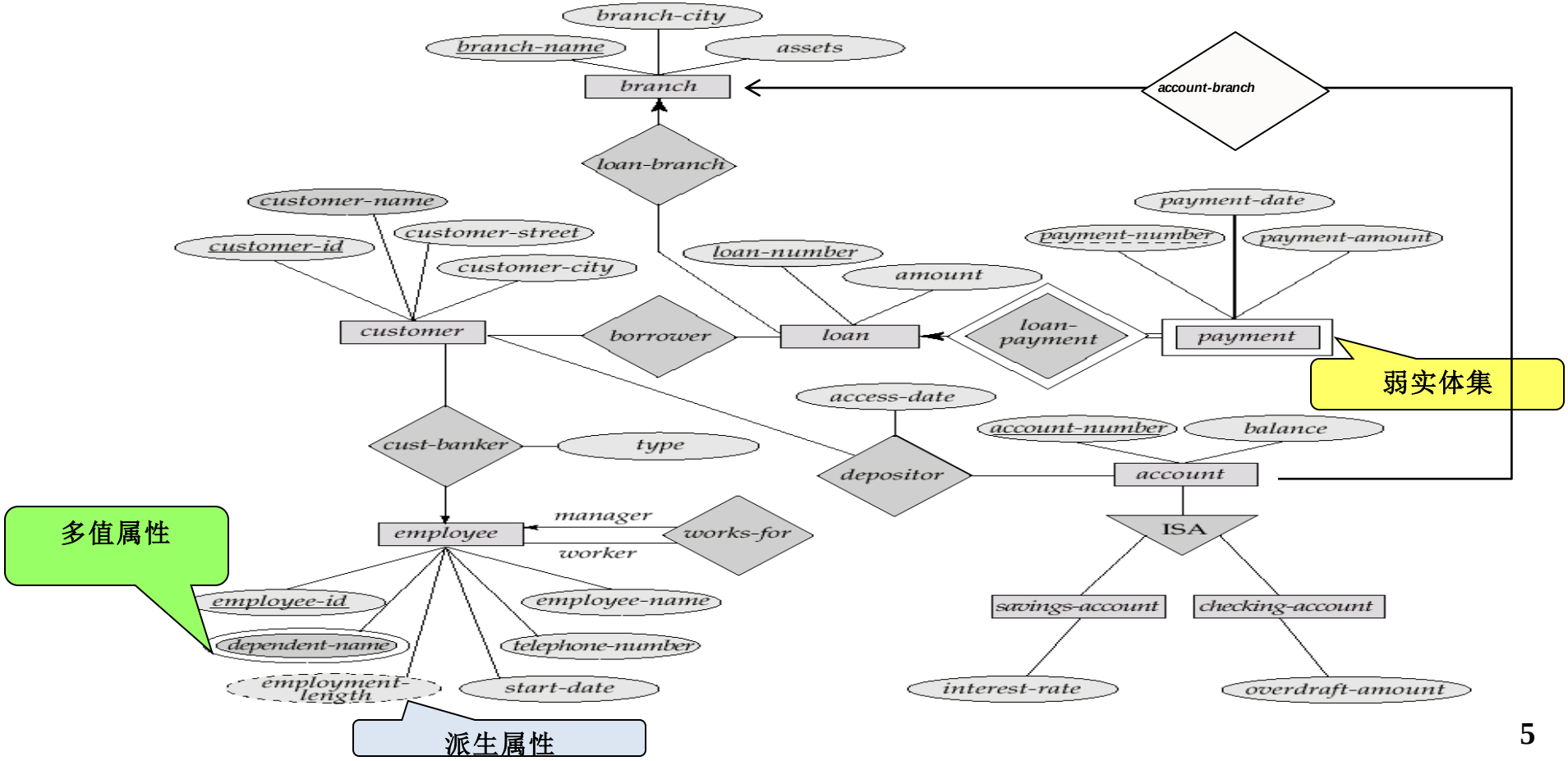
学生表

大学数据库



- 关系模型
- SQL
- 数据库设计
- 存储与查询
- 交易

银行企业的E-R图


branch-city
branch-name
assets
branch
account-branch
loan-branch
customer-name
customer-id
customer-street
customer-city
loan-number
amount
payment-date
payment-number
payment-amount
customer
borrower
loan
loan-payment
payment
弱实体集
access-date
account-number
balance
cust-banker
type
depositor
account
多值属性
manager
employee
works-for
worker
ISA
employee-id
employee-name
dependent-name
telephone-number
employment-length
start-date
savings-account
checking-account
interest-rate
overdraft-amount
派生属性

银行架构

- 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)(派生自多值属性)

- Account_branch = (account_number, branch_name)
- Loan_branch = (loan_number, branch_name)
- 借款人= (customer_id, loan_number)
- 存款人= (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)

大纲

设计过程概述

- 实体-关系模型(imagesini)
- 约束
- 实体-关系图(绘图)
- 还原为关系图式
- 总结

开发数据库应用包含的任务

设计数据库模式

- 关系（表）
- 关系的属性
- 约束

设计访问和更新数据的程序

设计控制数据访问的安全模式

数据抽象

信息世界

概念模型

机器世界

支持的数据模型
DBMS

真实世界概念模型 ⇨
数据库设计师

概念模型逻辑模型 ⇨
数据库设计师

逻辑模型物理模型 ⇨
DBMS

数据库设计(英语:Database Design)

- **概念设计()**
  - 将现实世界的组织映射到概念模型
- **逻辑设计**
  - 将概念模型转化为逻辑模型
- **物理设计**
  - 将逻辑模型实例化为物理组织和存储

数据库设计(续)

- 了解正在建模的现实世界领域
- 使用数据库设计模型指定它
  - 设计模型对于模式设计特别方便，但不一定由DBMS实现
    - **实体/关系(E/R)模型**
    - 对象定义语言(ODL)
- 将规范转换为DBMS的数据模型
  - **关系型、XML、面向对象等。**
- **创建DBMS模式**

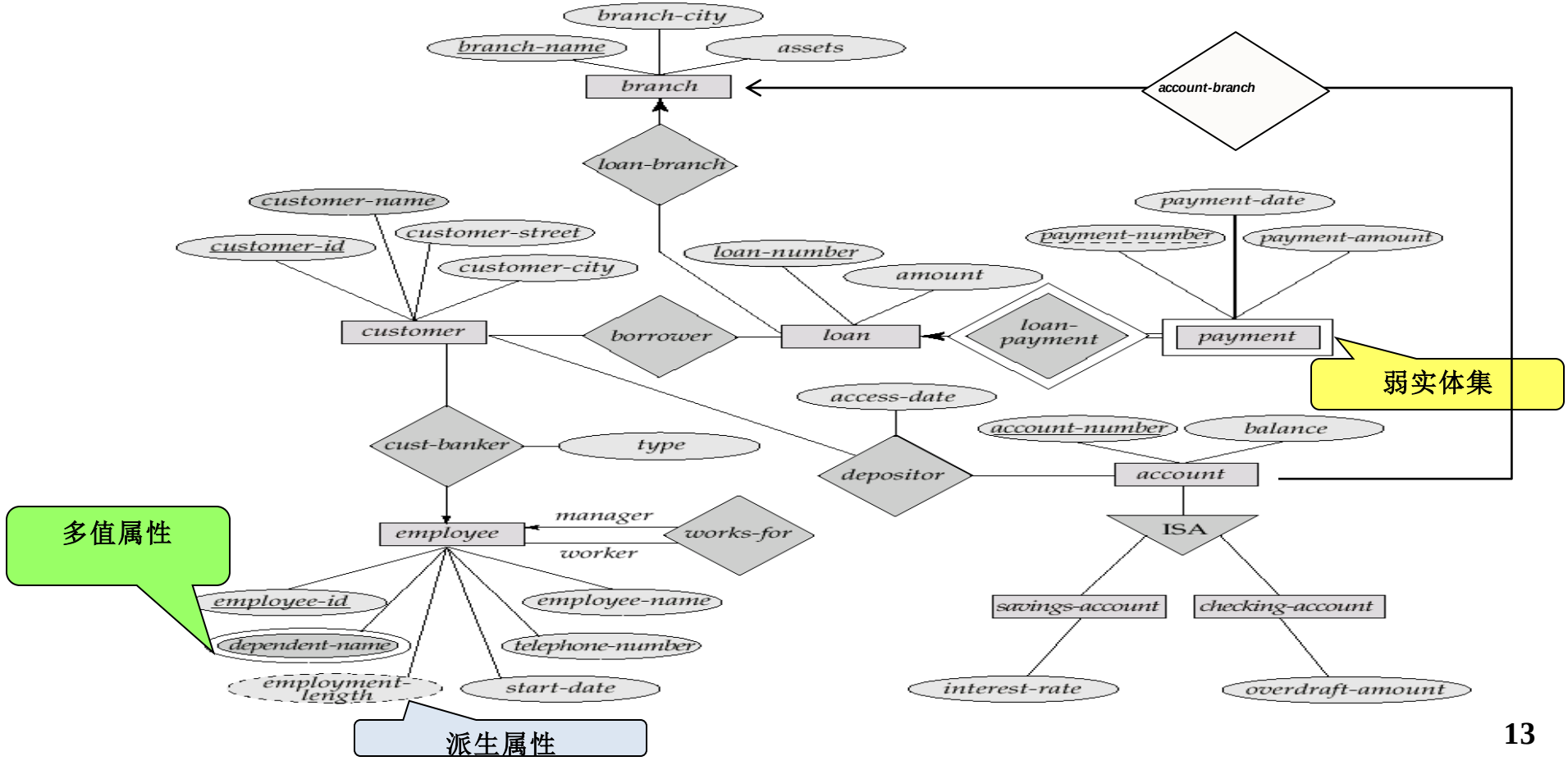
大纲

- 设计过程概述

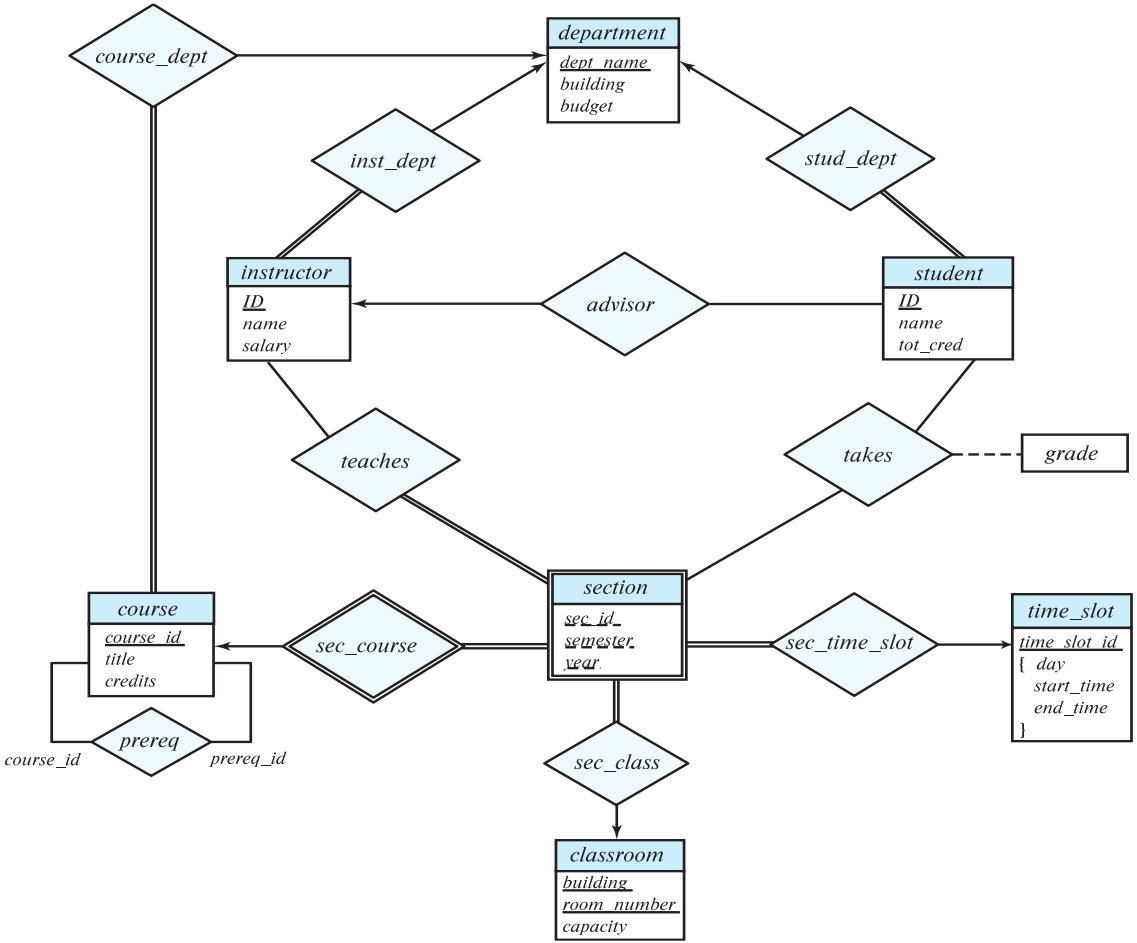
<s:1>实体-关系模型(cfr)

- 约束
- 实体-关系图(绘图)
- 还原为关系图式
- 总结

银行企业的E-R图


branch-city
branch-name
assets
branch
account-branch
loan-branch
customer-name
customer-id
customer-street
customer-city
loan-number
amount
payment-date
payment-number
payment-amount
customer
borrower
loan
loan-payment
payment
弱实体集
access-date
account-number
balance
cust-banker
type
depositor
account
多值属性
manager
employee
works-for
worker
ISA
employee-id
employee-name
dependent-name
telephone-number
employment-length
start-date
savings-account
checking-account
interest-rate
overdraft-amount
派生属性

E-R


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

数据库概念设计

- **概念设计(此阶段使用ER模型)**
  - **实体和关系是什么?**
  - **我们应该在数据库中存储关于这些实体和关系的哪些信息?**
  - **应该保存哪些完整性约束或业务规则?**

  - ER模型中的数据库"模式"可以使用ER图图形化地表示
  - 可以将ER图映射到关系模式中吗

ER模型：通用视图

- 历史上非常流行
- “打了折扣”的面向对象设计模型
- ER图表示设计
- 主要是一个设计模型——没有被任何主要的DBMS实现
- **三个概念**
  - **实体集**
  - **属性**
  - **关系集**

陈品山(Peter pinshan Chen)

- Peter P.陈博士是实体-关系模型(ER模型)的创始人，也是ER国际会议的创始人
- ER模型作为许多系统分析和设计方法、计算机辅助软件工程(CASE)工具和存储库系统的基础

Peter陈，实体-关系模型——迈向数据的统一视图，*ACM*数据库系统事务，第1卷，第1期，1976年3月，第9 - 36页

实体集

- **数据库可以建模为**
  - 实体的集合
  - 实体之间的关系
- **实体是一个存在并与其他对象区分开来的对象**
  - 例如，特定的人、公司、事件、大学
- 实体有属性
  - 例如，人有名字和地址
- **一个是一组共享相同属性的相同类型的实体**
  - 例如，所有人、公司、树、假期的集合
- **实体集的扩展是属于实体集的实体的实际集合**

实体集客户和贷款

customer_ customer_ customer_贷款额id名称街道城市编号

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

L-17	1000
L-23	2000
L-15	1500
L-14	1500
L-19	500
L-11	900
L-16	1300

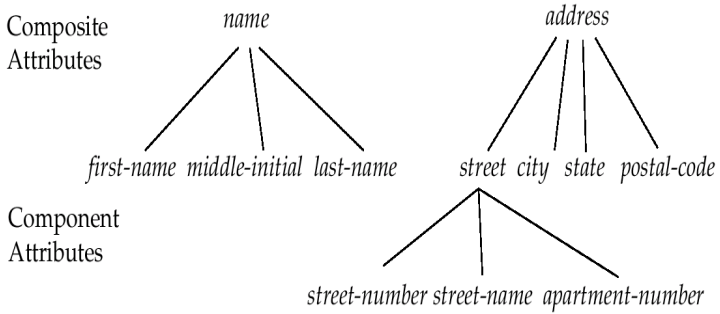
loan

属性(全称)

- 一个实体由一组属性表示，这些属性是实体集的所有成员所拥有的描述性属性

  Customer (customer_id, customer_name, customer_street, customer_city)
  loan (loan_number, amount)

- **Domain() -每个属性的允许值的集合**
- **属性类型**
  - 简单属性和复合属性
  - 单值属性和多值属性
  - 派生属性



关系集

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

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

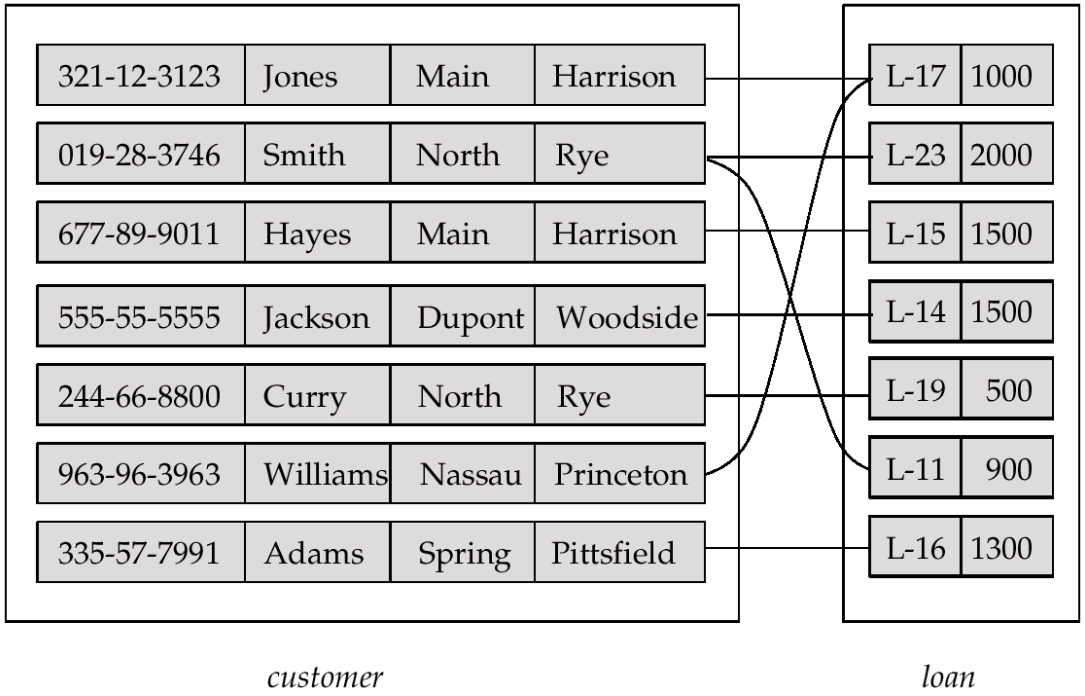
- **A relationship set** is a mathematical relation among $n \geq 2$ entities, each taken from entity sets

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

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

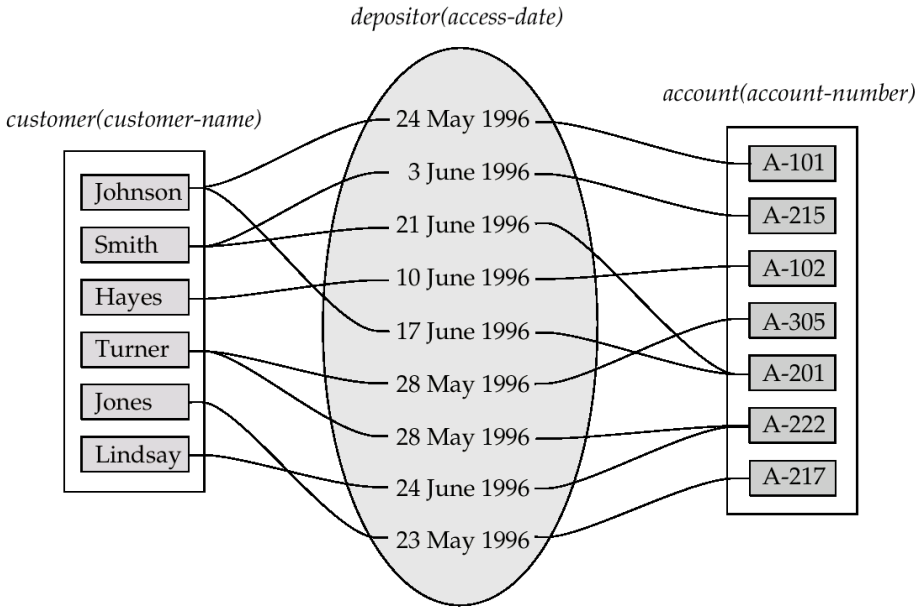
(Hayes, A-102) ∈ depositor

关系集借款人


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
L-17 1000
L-23 2000
L-15 1500
L-14 1500
L-19 500
L-11 900
L-16 1300
customer
loan

关系集(续)

- 一个也可以是关系集的属性
- 例如，实体集customer和account之间的存款人关系集可能具有属性access-date



关系集的程度(/)

- 参与关系集的实体集的数量
  - 涉及两个实体集的关系集是二进制的。
  - 关系集可能涉及两个以上的实体集
  - 两个以上实体集之间的关系很少，大多数关系是二元的

大纲

- 设计过程概述
- 实体-关系模型

☞**约束**

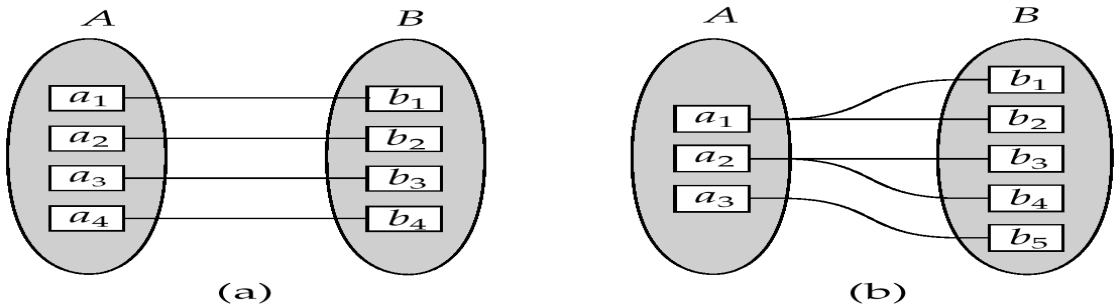
- 实体-关系图
- 还原为关系图式
- 总结

映射基数(Mapping Cardinalities)

- 表示另一个实体可以通过关系集与之关联的实体的数量
- 对于二元关系集，映射基数必须是以下类型之一
  - **一对一(1 ~ 1):A中的一个实体最多与B中的一个实体关联，B中的一个实体最多与A中的一个实体关联**
  - **一对多(1):A中的一个实体与B中的任何数量(零个或多个)实体相关联。然而，B中的一个实体最多只能与A中的一个实体相关联**
  - **多对一(多对一):A中的一个实体至多与一个B中的任何数量(零个或多个)实体相关联。然而，B中的一个实体可以与A中的任意数量的实体相关联(零个或多个)**
  - **多对多:A中的一个实体与B中的任意数量(零个或多个)实体相关联，B中的一个实体与A中的任意数量(零个或多个)实体相关联**

映射基数(续)

- 映射基数类型:
  - 一对一(1 ~ 1):A中的一个实体最多与B中的一个实体关联，B中的一个实体最多与A中的一个实体关联。
  - 一对多:A中的一个实体与B中的任意数量(零个或多个)实体相关联。但是，B中的一个实体最多只能与A中的一个实体相关联。

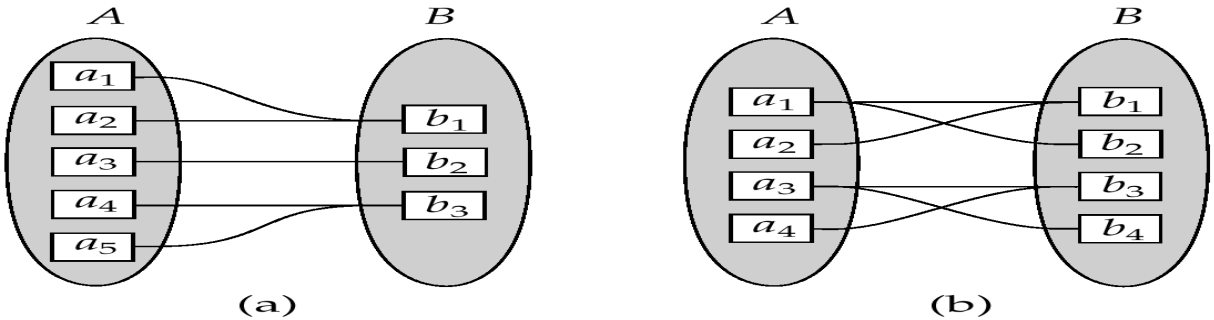


一对一　　一对多

注意:A和B中的某些元素可能无法映射到另一个集合中的任何元素

映射基数(续)

- 映射基数类型：
  - 多对一：A中的一个实体至多与B中的一个实体相关联。然而，B中的一个实体可以与A中的任何数量(零个或多个)实体相关联。
  - 多对多：A中的一个实体与B中的任意数量(零个或多个)实体相关联，B中的一个实体与A中的任意数量(零个或多个)实体相关联。

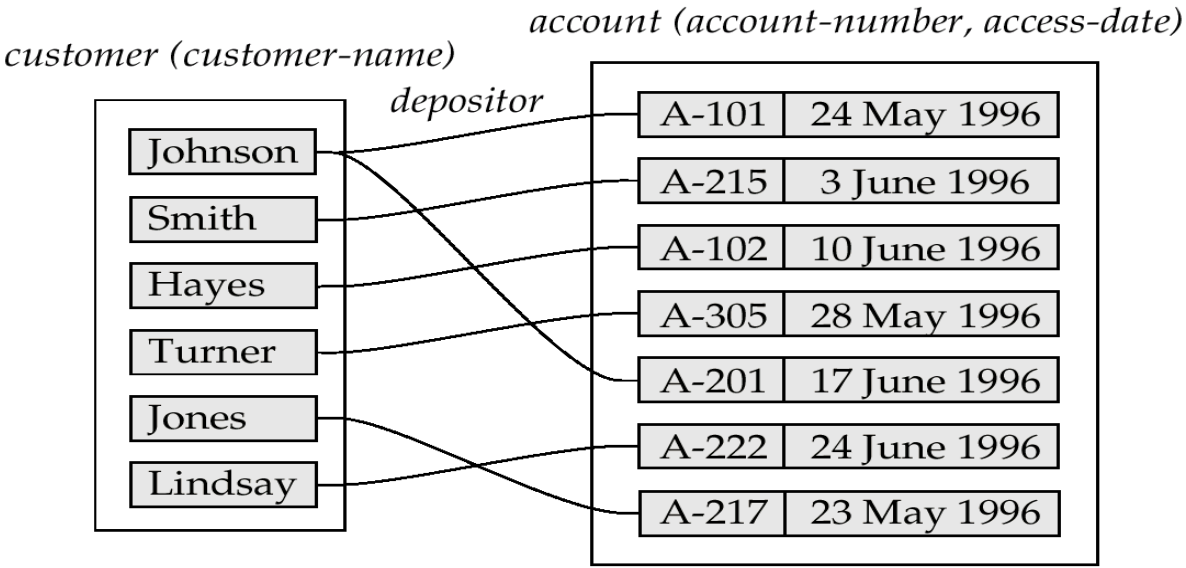


多对一　　多对多

注意：A和B中的某些元素可能无法映射到另一个集合中的任何元素

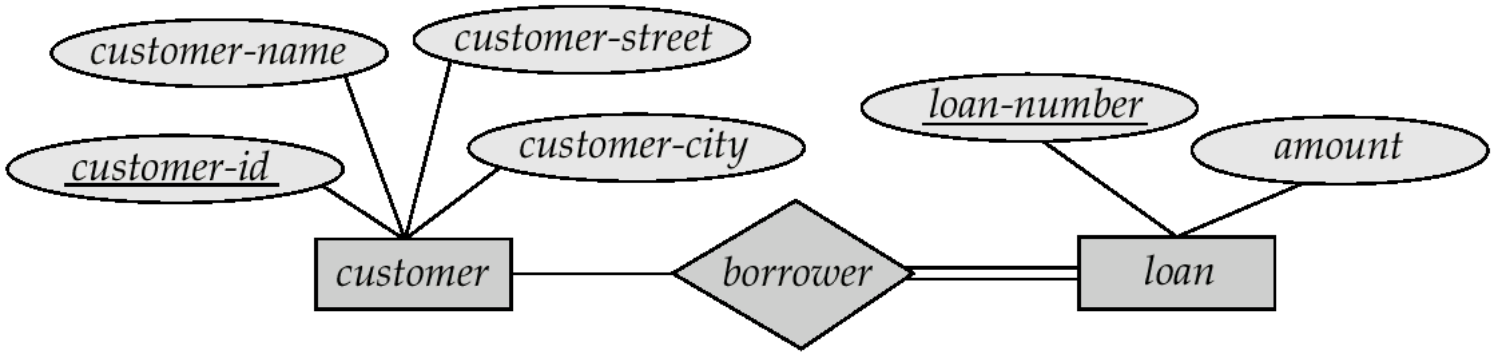
映射基数影响ER设计

- 如果每个帐户只能有一个客户，是否可以将access-date作为帐户的属性，而不是关系属性



参与约束(中文版)

- **总参与(双线表示):**
  - 实体集中的每个实体至少参与了关系集中的一个关系
  - **例如,每个学生实体通过顾问联系同至少一名教师相联系,学生在联系集顾问中是全部参与**
- **部分参与**
  - 某些实体可能不参与关系集中的任何关系
  - **如有的教练可能不指导学生,所以教练在联系集顾问中是部分参与**



大纲

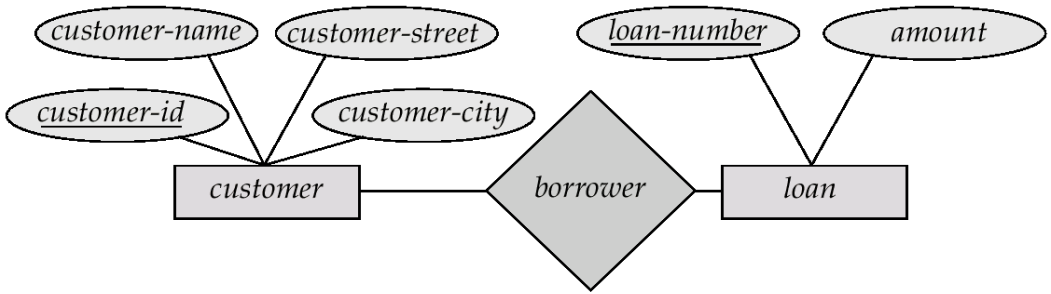
- 设计过程概述
- 实体-关系模型(英语:Entity-Relationship Model)
- 约束

实体-关系图(绘图)

- 还原为关系图式
- 总结

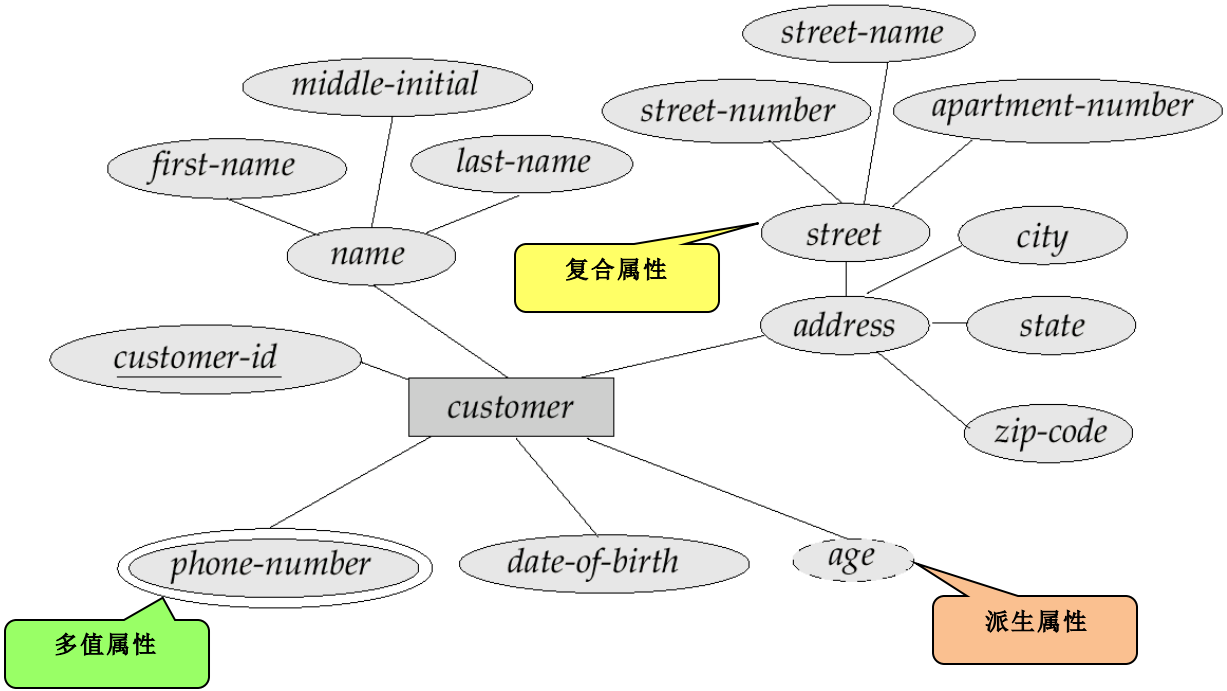
实体关系图

- 矩形表示实体集
- 菱形表示关系集。
- 线条将属性链接到实体集，将实体集链接到关系集。
- 椭圆表示属性
  - 双省略号表示多值属性
  - 虚线椭圆表示派生属性
- 下划线表示主键属性

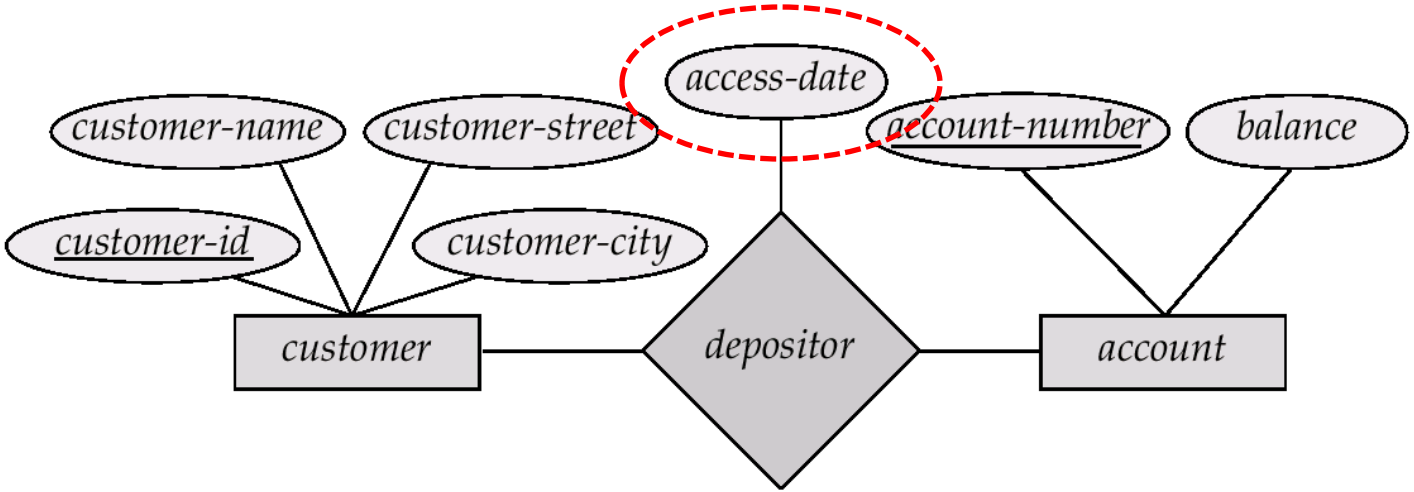


E-R图

- 具有复合、多值和派生属性的E-R图

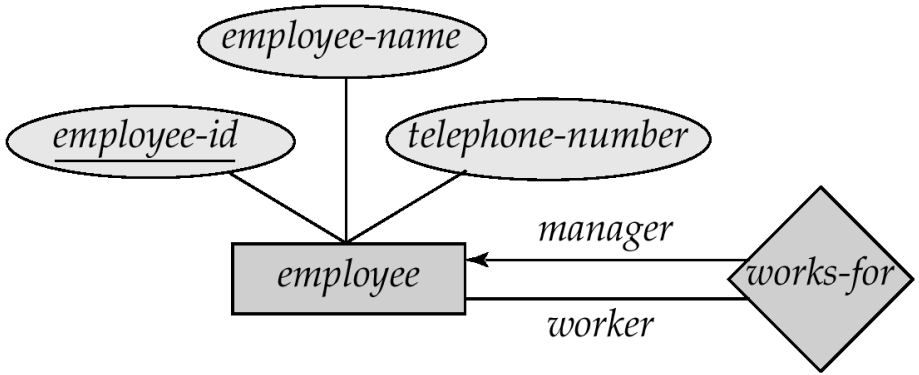


带有属性的关系集


access-date
customer-name
customer-street
account-number
balance
customer-id
customer-city
customer
depositor
account

角色()

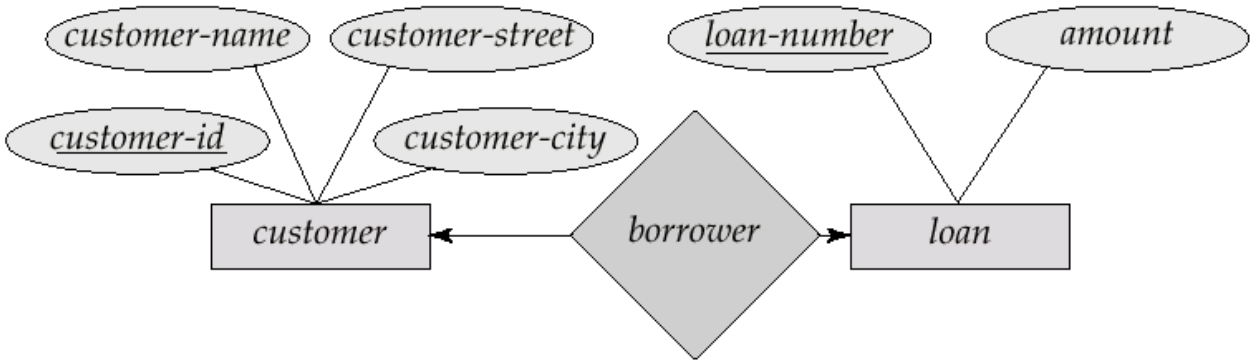
- **关系的实体集不需要是不同的**
  - 标签"管理者"和"工作者"被称为角色;它们指定了员工实体如何通过works-for关系集进行交互
  - 角色标签是可选的，用于澄清关系的语义



基数约束

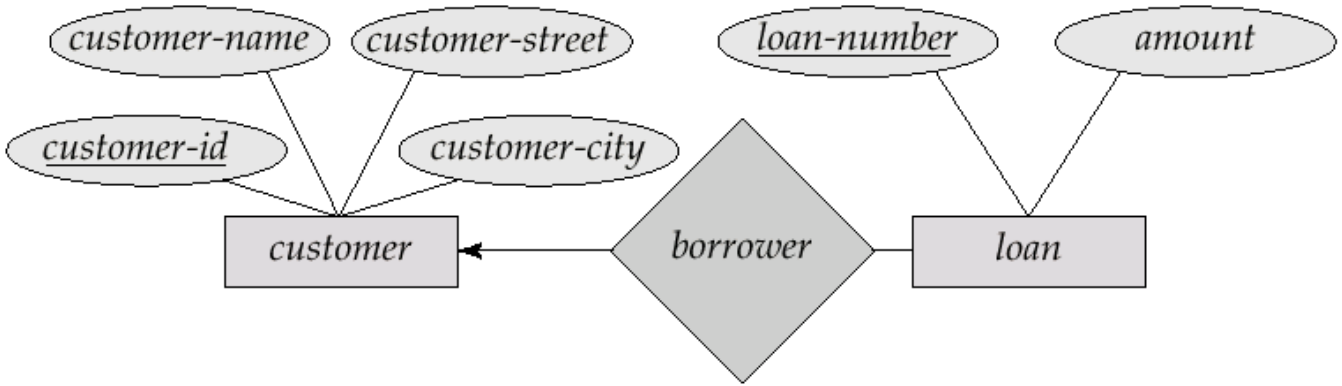
- 我们通过在关系集和实体集之间画一条有向线(→)，表示"一"，或一条无向线(-)，表示"许多"来表达基数约束。

- **一对一的关系**
  - 客户通过关系借款人最多与一笔贷款相关联
  - 一笔贷款最多只能通过借款人与一个客户关联



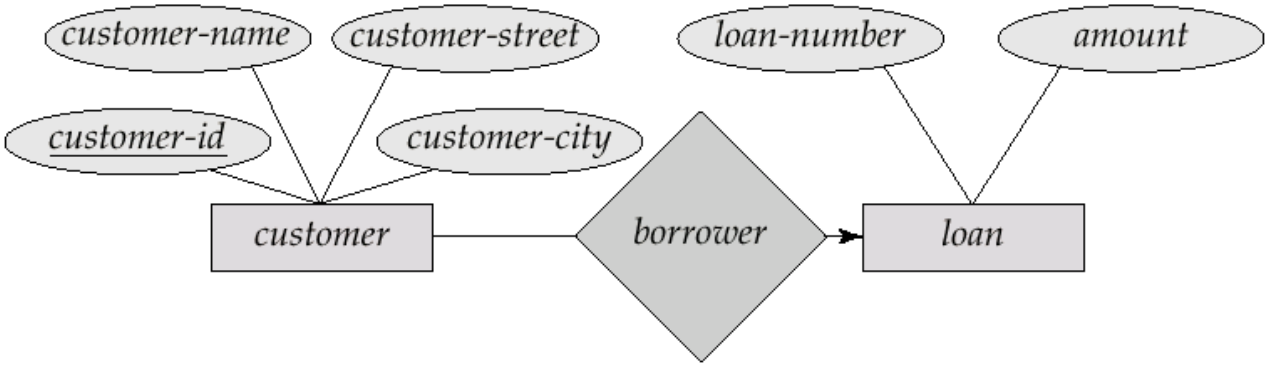
一对多关系

- **一对多关系**
  - 一笔贷款通过借款人最多与一个客户关联
  - 一个客户通过借款人与几笔(包括0笔)贷款相关联



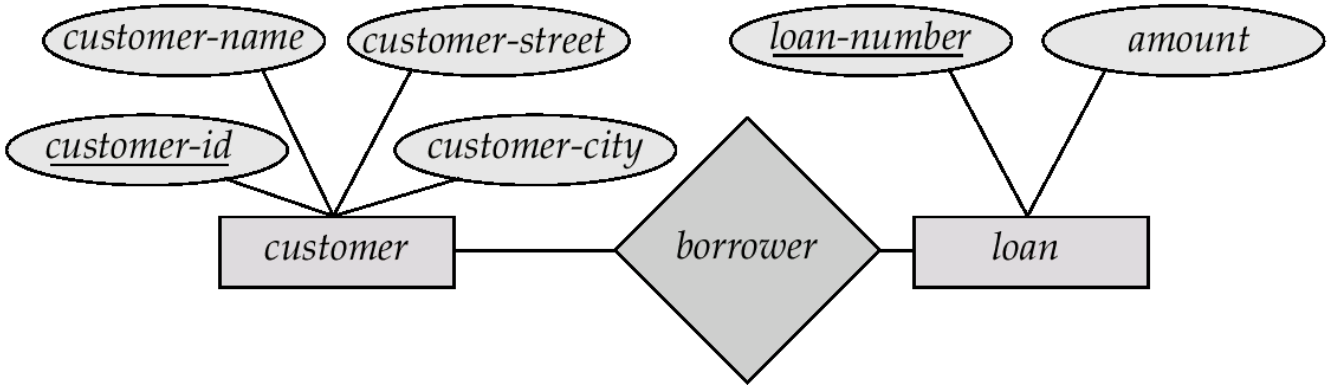
多对一的关系

- **多对一的关系**
  - 一笔贷款通过借款人与多个(包括0个)客户关联
  - 一个客户通过借款人最多与一笔贷款相关联



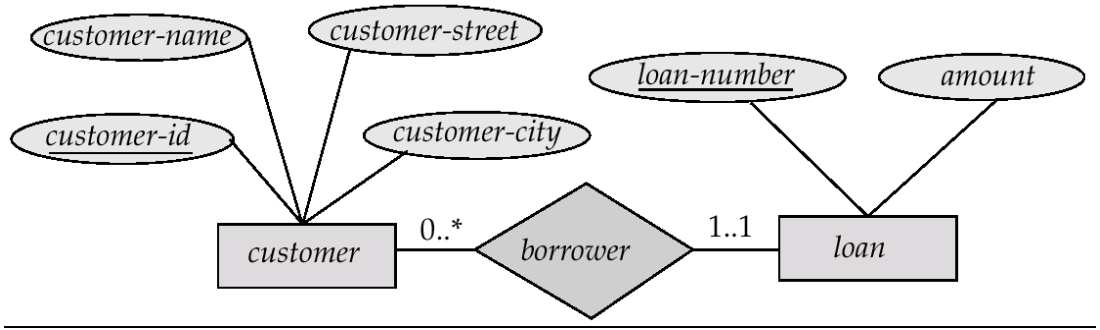
多对多关系

- **多对多关系**
  - 一个客户通过借款人与几笔(可能是0笔)贷款相关联
  - 一笔贷款通过借款人与多个(可能为0)客户关联



基数限制的替代表示法

- Cardinality limits can also express participation constraints
- Notation: $m..n$, where m and n are the minimum and maximum cardinalities respectively



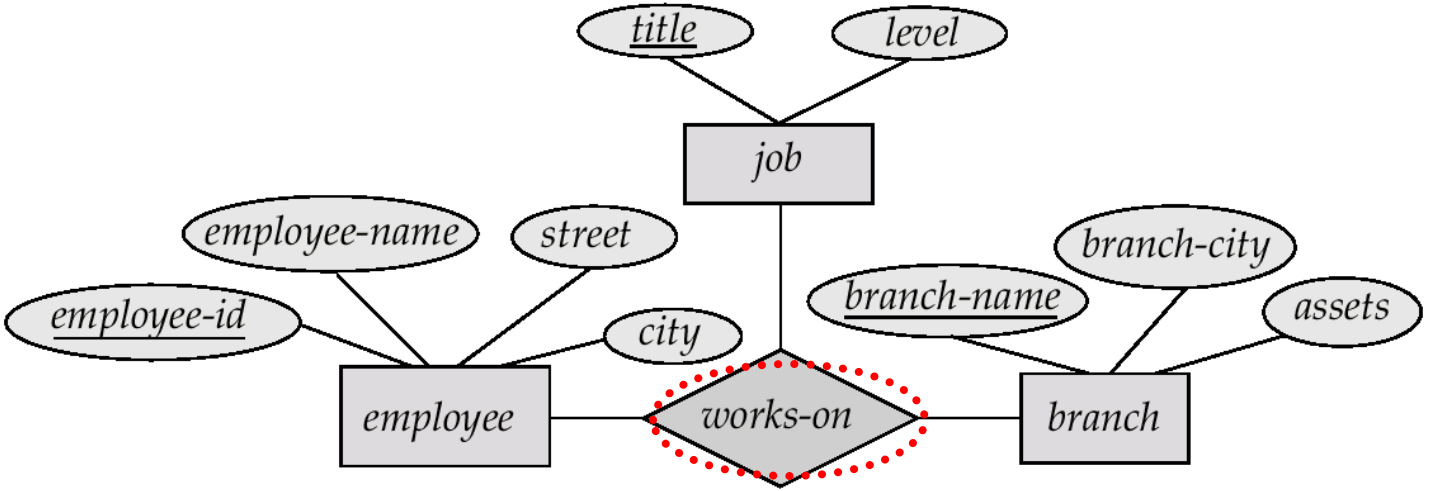
键(/)

- **实体集的超级键()是一个或多个属性的集合，其值唯一地决定了每个实体**
- **一个实体集的候选键()是一个最小超级键**
  - Customer_id是customer的一个候选键
  - Account_number是account的候选键
- 虽然可能存在多个候选键，但选择其中一个候选键作为主键()

关系集的键

- 参与实体集的主键组合形成关系集的超级键
  - (customer_id, account_number)是存款人的超级键
- 在决定哪些是候选键时必须考虑关系集的映射基数
- 在有多个候选键的情况下，选择主键时需要考虑关系集的语义

三元关系的E-R图


title
level
job
employee-name
street
branch-city
branch-name
assets
employee-id
city
employee
works-on
branch

三元关系上的基数约束

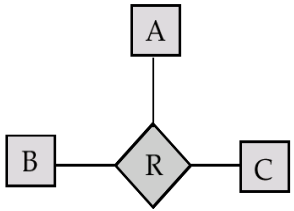
- 我们最多允许一个箭头脱离三元关系
  - **例如，从work -on到job的箭头表示每个员工在任何分支最多只从事一项工作**
- 如果有一个以上的箭头，则有两种方式来定义其含义
  - 例如，a, B和C之间的三元关系R带有指向B和C的箭头
- **为了避免混淆，我们禁止使用多个箭头**

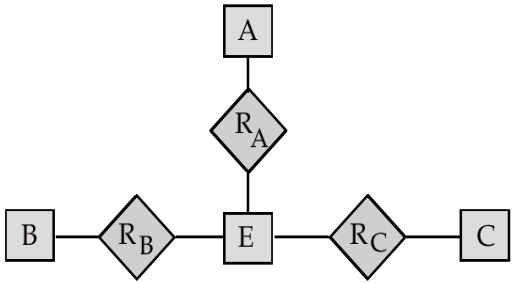
二元关系与非二元关系

- 一些看似非二元的关系可能用二元关系更好地表示
  - 例如，一个三元关系父母，将孩子与他/她的父亲和母亲联系起来，最好用两个二元关系，父亲和母亲来代替
    - 使用两个二元关系允许部分信息(例如，只知道母亲)。
  - 但也有一些关系自然是非二元的
    - 如工作

转换非二元关系

- 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





转换非二元关系(续)

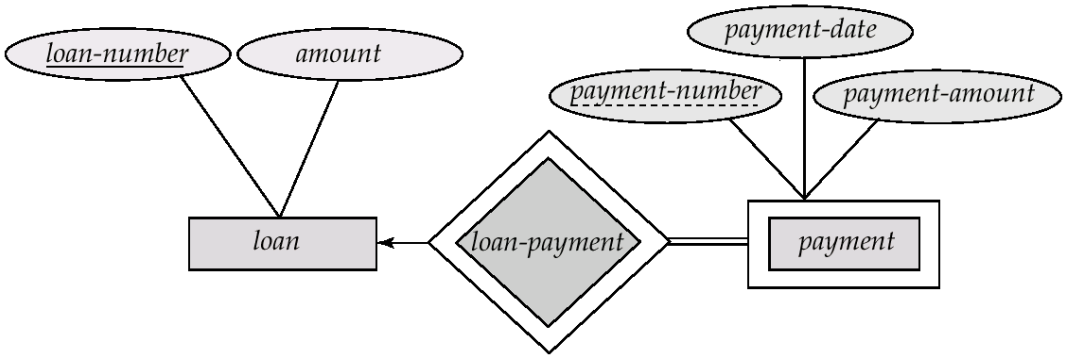
- **翻译的约束**
  - 翻译所有约束可能是不可能的
  - 翻译后的模式中可能存在不能对应于R的任何实例的实例
  - 我们可以通过使E成为由三个关系集识别的弱实体集(简要描述)来避免创建识别属性

弱实体集(Weak Entity Sets)

- 没有主键的An称为弱实体集
- 弱实体集的存在依赖于标识实体集的存在
  - **用双菱形描绘的识别关系**
- 鉴别器(部分键，)
- 弱实体集的主键
  - **鉴别符加标识实体集的主键**

弱实体集(续)

- 用双矩形来描绘弱实体集。
- 用虚线在弱实体集的鉴别符()下面划线。
  - Payment_number -支付实体集的鉴别符
  - 支付主键- (loan_number, payment_number)



弱实体集(Cont.)

- 注意:强实体集的主键没有显式地与弱实体集存储在一起，因为它在标识关系中是隐式的。
- 如果loan_number被显式存储，则可以将支付作为强实体进行

更多弱实体集示例

- 在大学里，一门课程是一个强实体，而课程设置可以被建模为一个弱实体
- course_offering的鉴别符为学期(包括学年)和section_number(如果有多个section)。
- 如果我们将course_offering建模为一个强实体，我们将把course_number建模为一个属性。然后，与course的关系将隐式地包含在course_number属性中

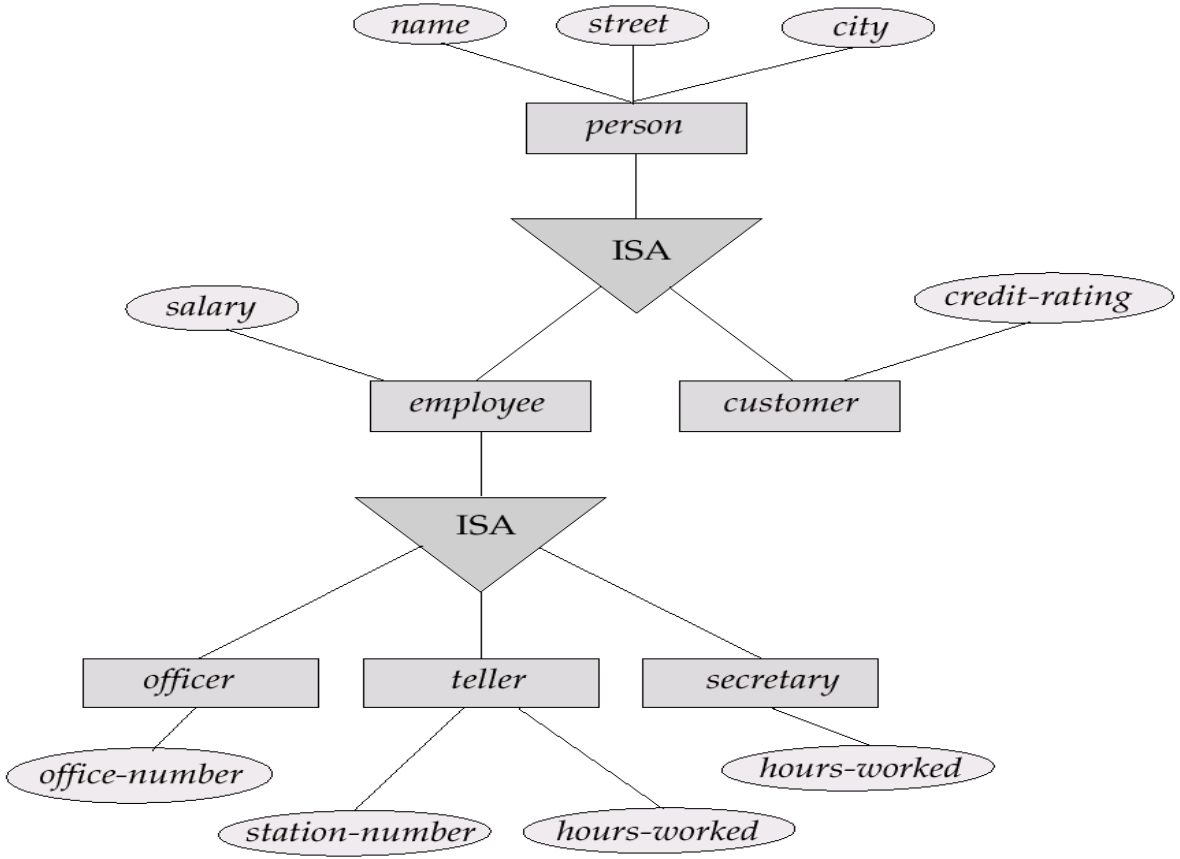
ER D design Issues

- 实体集与属性的使用
  - 选择主要取决于被建模企业的结构，以及与所讨论的属性相关的语义
- 使用实体集vs.关系集
  - 可能的指导方针是指定一个关系集来描述实体之间发生的动作
- 二元与n元关系集
- 关系属性的放置

专业化()

- 自顶向下的设计过程
  - 在实体集中指定与集合中的其他实体不同的子分组
  - 这些子分组成为具有不适用于高级实体集的属性或参与关系的低级实体集。
  - 由标记为ISA的三角形组件描述，例如，客户"是一个"人
- **属性继承**
  - 低级实体集继承它所链接的高级实体集的所有属性和关系参与

例子


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

泛化()

- 自下而上的设计过程
  - 将许多具有相同特征的实体集组合成一个更高级的实体集

- **专门化()，泛化()**
  - 专门化和泛化是彼此的简单倒置
  - 它们在E-R图中以相同的方式表示

专门化与泛化(续)

- 可以有基于不同特征的实体集的多个专门化。
  - **例如，永久雇员vs.临时雇员，以及官员vs.秘书vs.出纳员**
  - 每个特定的雇员都是
    - 永久雇员或临时雇员之一的成员，
    - 职员、秘书或出纳员的成员

- ISA关系也称为超类-子类关系

专门化/泛化上的设计约束

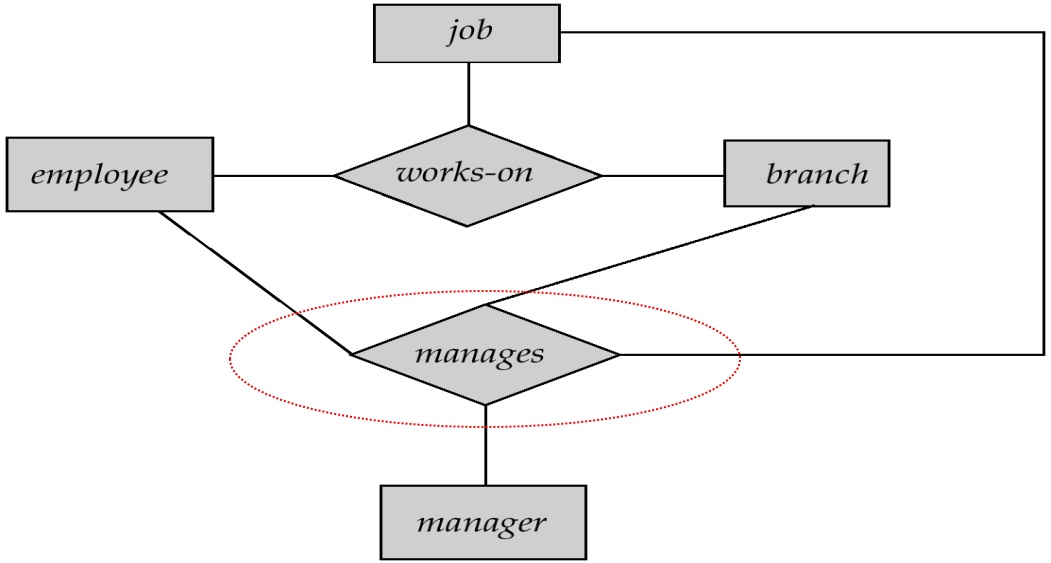
- 约束哪些实体可以成为给定低层实体集的成员
  - condition-defined (attribute-defined)
  - 用户定义的
- 在单个泛化中，实体是否可以属于多个较低级实体集的约束
  - 不相交的
  - 重叠

专门化/泛化的设计约束(续)

- 完整性约束——指定高级实体集中的实体是否必须至少属于泛化内的低级实体集中的一个。
  - Total:一个实体必须属于低层实体集中的一个
  - Partial:实体不必属于较低级别的实体集

聚合(Aggregation)

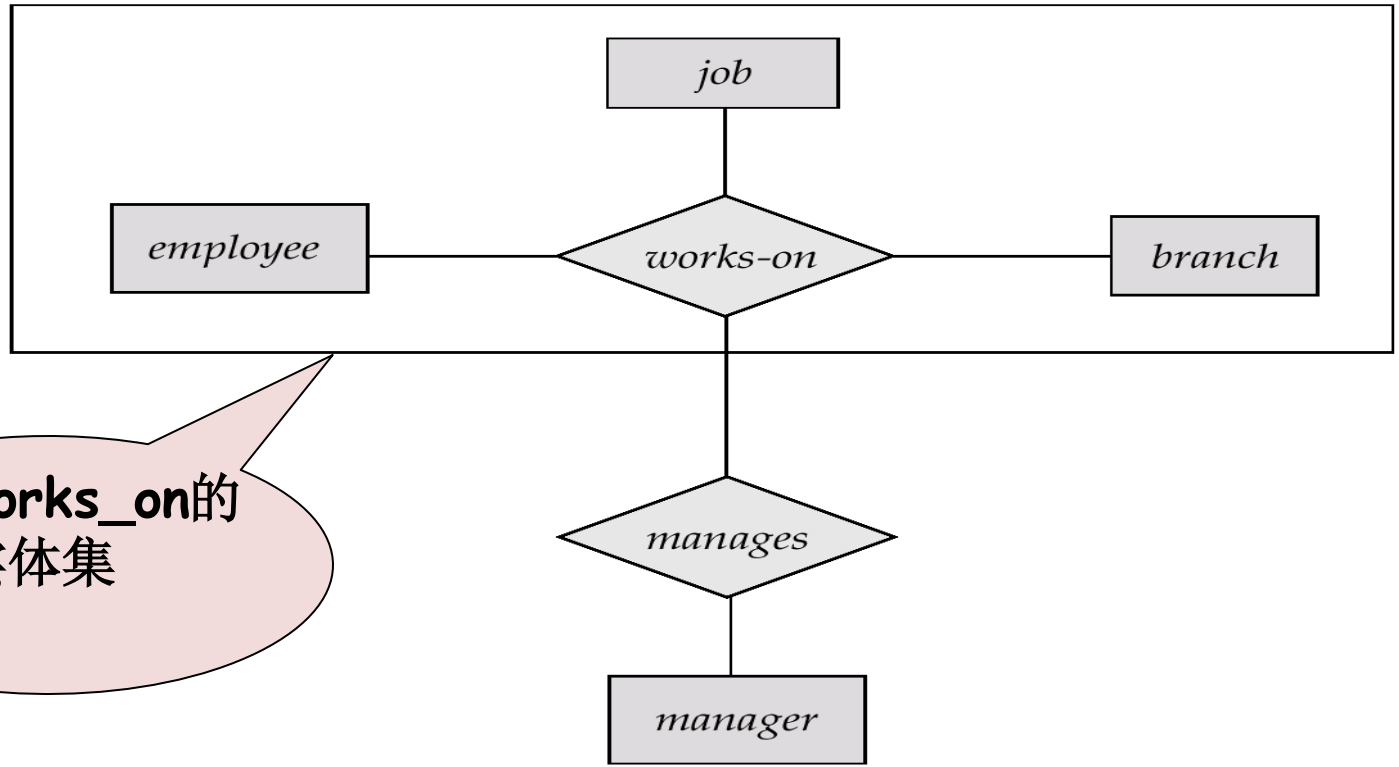
- 考虑我们前面看到的三元关系
- 假设我们想要记录某个分支的员工执行的任务的经理



聚合(续)。

- 关系集-工作和管理表示重叠的信息
- 通过聚合消除这种冗余
  - 将关系视为抽象实体
  - 允许关系之间的关系
- 在不引入冗余的情况下，下图表示:
  - 一名员工在某一分支机构从事某一工作
  - 员工、分支机构、工作组合可能有一个关联的经理

带聚合的E-R图


job
employee
works-on
branch
manages
manager
称为works_on的
高级实体集

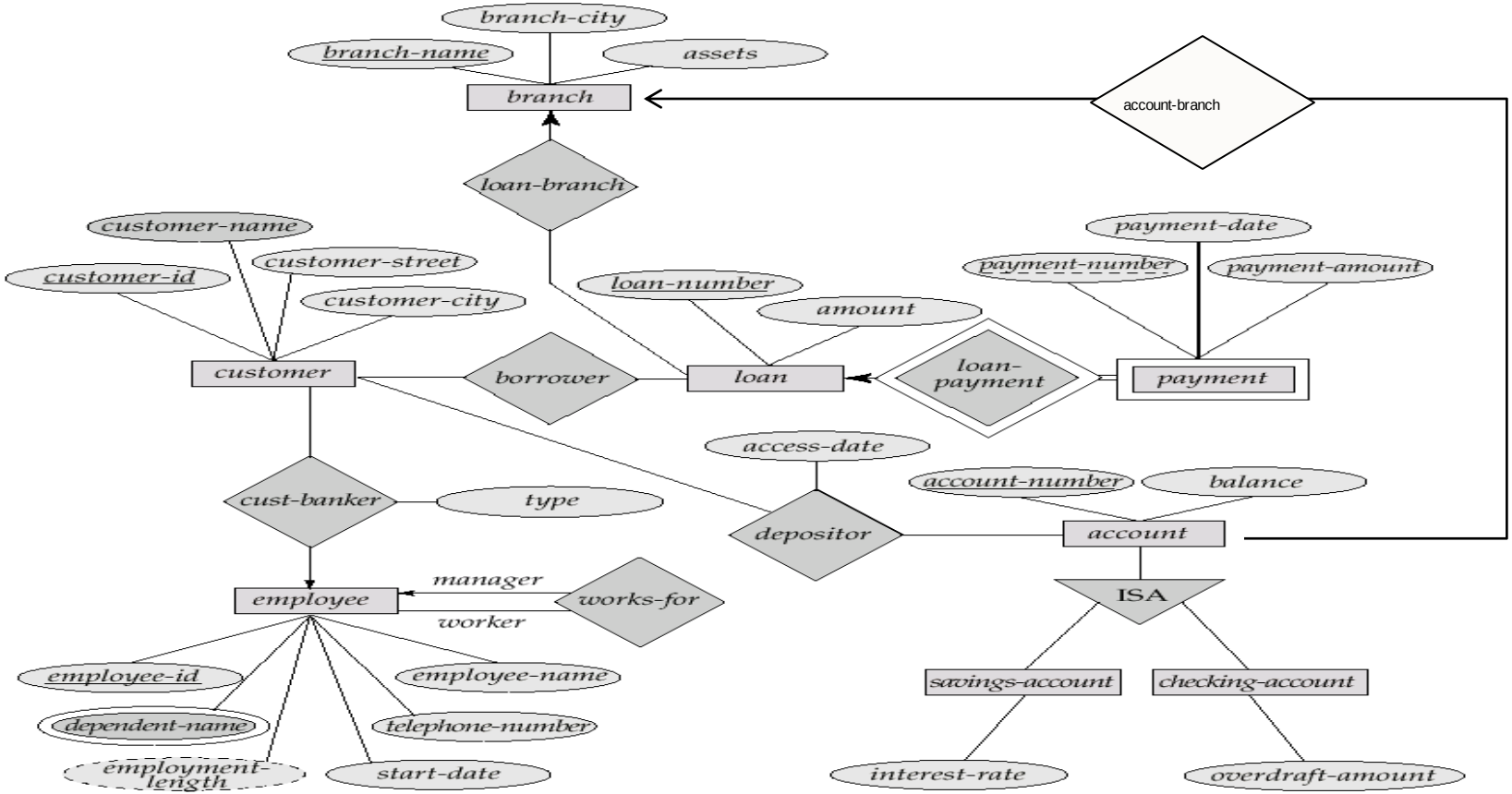
E-R设计决策

- 使用属性或实体集来表示对象
- 一个现实世界的概念是用实体集还是关系集来最好地表达
- 使用三元关系还是一对二元关系
- 强实体集或弱实体集的使用
- 专门化/泛化的使用——有助于设计中的模块化
- 聚合的使用-可以将聚合实体集视为单个单元，而不关心其内部结构的细节

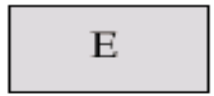
数据库设计阶段

- 需求分析
- 概念设计(E-R模型)
- 功能需求分析
  - 描述将对数据执行的操作
  - 审查设计
- 逻辑实现
  - 从概念模型到实现模型的映射
  - 比如关系模型、OO模型
- 物理实现
  - 指定数据库的物理特性
  - 缓冲区大小、索引……

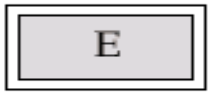
银行企业的E-R图


branch-city
branch-name
assets
branch
account-branch
loan-branch
customer-name
customer-id
customer-street
customer-city
loan-number
amount
payment-date
payment-number
payment-amount
customer
borrower
loan
loan-payment
payment
access-date
account-number
balance
cust-banker
type
depositor
account
manager
employee
works-for
worker
ISA
employee-id
employee-name
dependent-name
telephone-number
employment-length
start-date
savings-account
checking-account
interest-rate
overdraft-amount

E-R表示法中使用的符号摘要



Entity Set



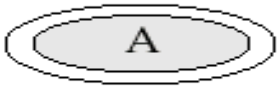
Weak Entity Set

Relationship Set

Identifying Relationship Set for Weak Entity Set

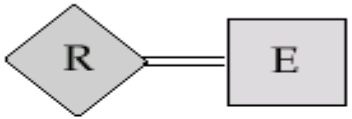
Primary Key

Attribute

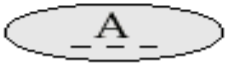


Multivalued Attribute

Derived Attribute

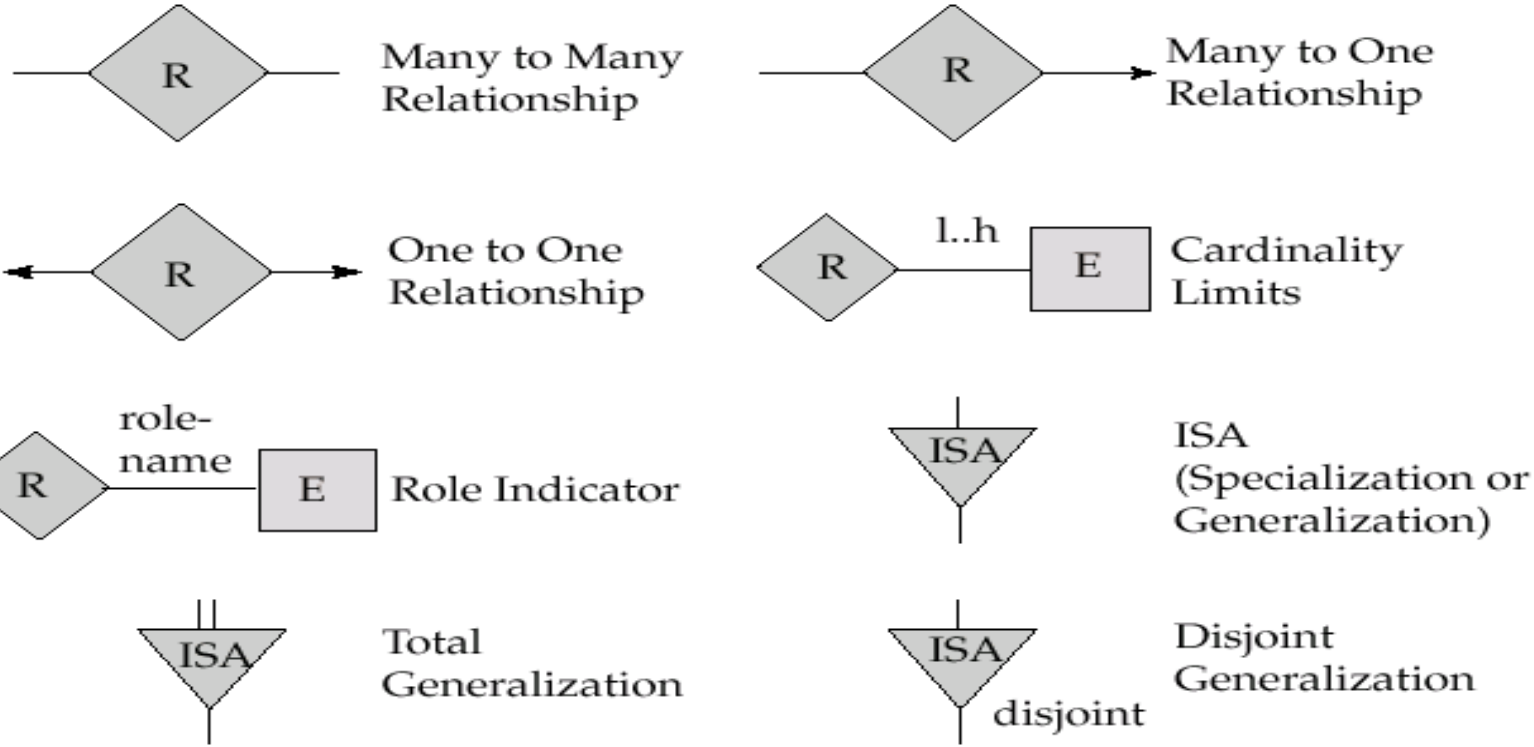


Total Participation of Entity Set in Relationship



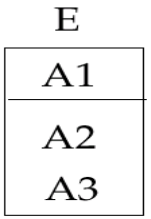
Discriminating Attribute of Weak Entity Set

符号摘要(续)

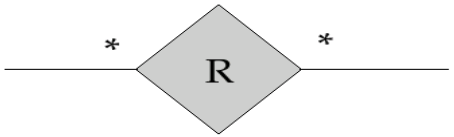

R
Many to Many Relationship
R
Many to One Relationship
R
One to One Relationship
R
l..h
E
Cardinality Limits
role-name
R
E
Role Indicator
ISA
ISA (Specialization or Generalization)
ISA
Total Generalization
ISA
disjoint
Disjoint Generalization

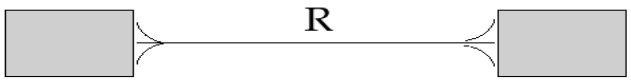
备选E-R符号

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

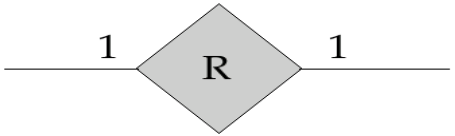


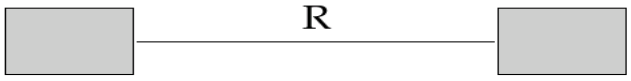
Many to Many Relationship



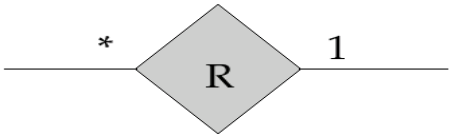


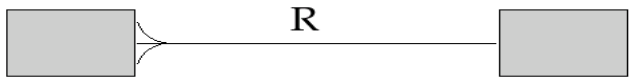
One to One Relationship





Many to One Relationship





大纲

- 设计过程概述
- 实体-关系模型(英语:Entity-Relationship Model)
- 约束
- 实体-关系图(绘图)

→**关系图式还原**

- 总结

还原为关系模式

- **E-R图到表的简化**
  - 对于每个实体集和关系集，都有一个唯一的表。
  - 每个表都有若干列
  - 将E-R图转换为表格式是从E-R图派生关系数据库设计的基础

- 主键允许实体集和关系集统一表示为表示数据库内容的表

将实体集表示为表

- 强实体集简化为具有相同属性的表

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

复合和多值属性

- **复合属性通过为每个组件属性创建一个单独的属性来实现扁平化**

- **实体E的多值属性M由单独的表EM表示**
  - 表EM具有与E的主键对应的属性和与多值属性M对应的属性
  - 多值属性的每个值映射到表EM的单独一行

表示弱实体集

- 弱实体集变成一个表，其中包含用于标识强实体集的主键的列

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

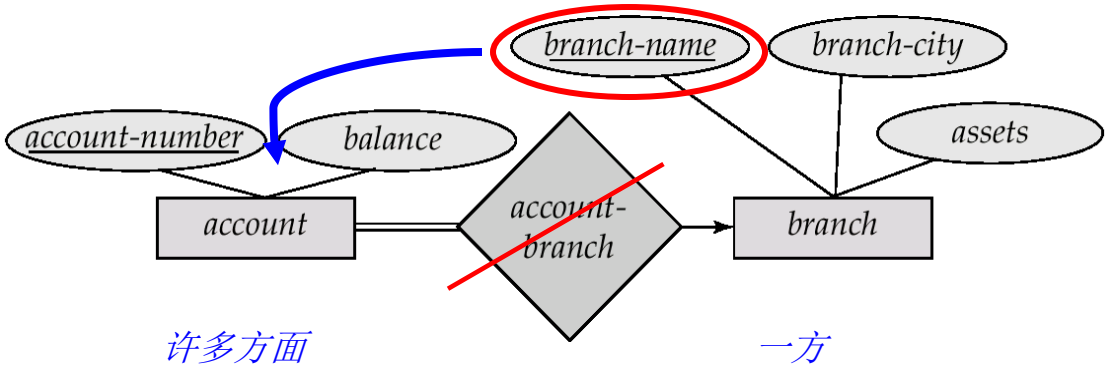
将关系集表示为表

- **多对多关系集**
  - 表示为一个表，其中列为两个参与实体集的主键，以及关系集的任何描述性属性。
  - **例如:关系集借方的表**

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

将关系集表示为表

- **多对一和一对多关系集**
  - 可以通过向多侧添加一个额外的属性，包含一侧的主键来表示吗
  - **例如:不为关系account-branch创建一个表，而是在实体集account中添加一个属性branch-name**



将关系集表示为表

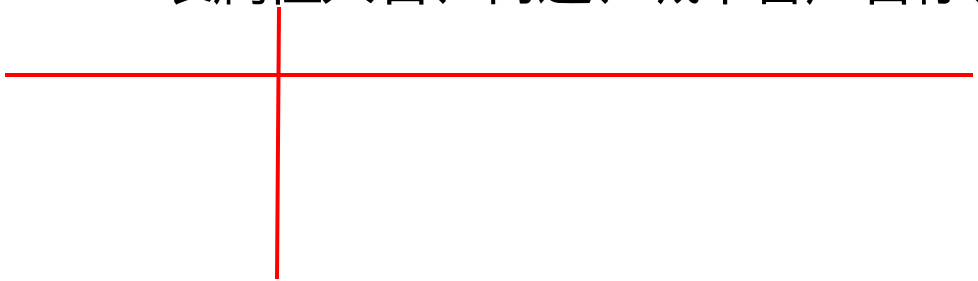
- **一对一的关系集**
  - **任意一方都可以选择充当“多”方**

- 如果参与是“多”方的部分，则可能导致空值
- 连接弱实体集与其识别强实体集的关系集对应的表是冗余的

将专门化表示为表

- **方法1：**
  - 为上级实体形成表格
  - 为每个低级实体集形成一个表，包括高级实体集的主键和本地属性

Table表属性人名、街道、城市客户名称、信用评级员工名称、工资



  - **缺点:查询实体的信息，例如employee，需要访问两个表**

将专门化表示为表(续)

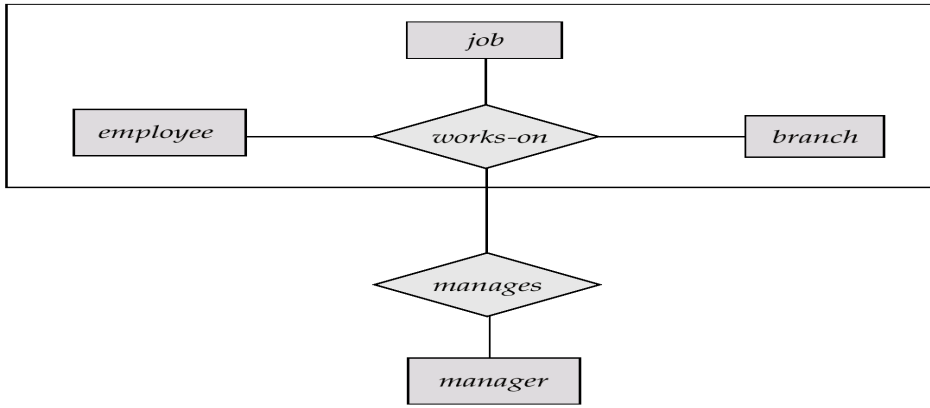
- **方法2:**
  - 为每个包含所有本地属性和继承属性的实体集形成一个表

    Table表属性人名、街道、城市客户名、街道、城市、信用评级员工名、街道、城市、工资

  - **缺点:对于既是客户又是员工的人，可能会冗余存储街道和城市**
  - 如果专门化是总的，表对于广义实体(人)不需要存储信息

与Aggregation相对应的关系

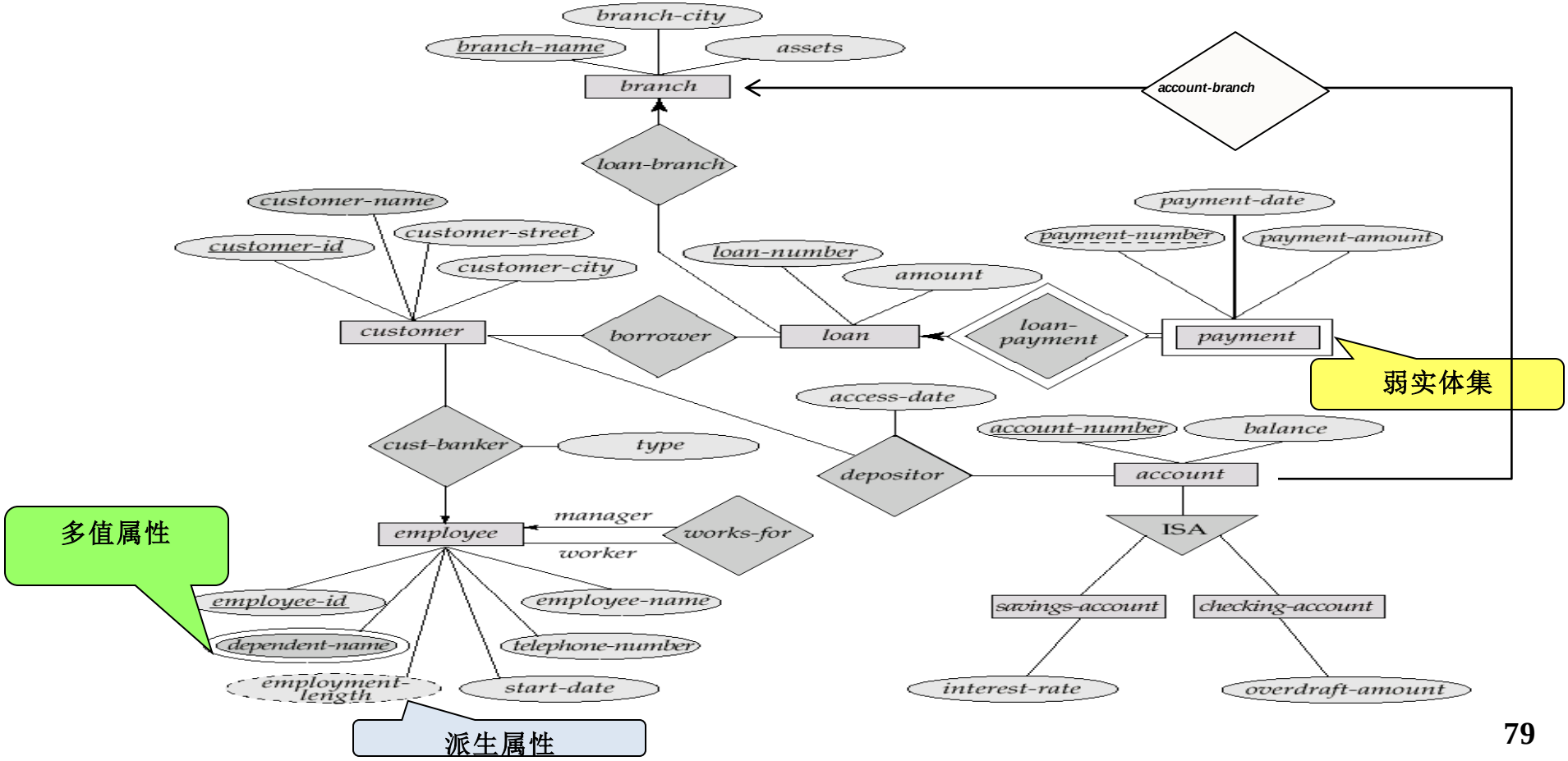
- 为了表示聚合，创建一个包含
  - 聚合关系的主键，
  - 关联实体集的主键
  - 任何描述性属性



- 例如，为了表示关系工作和实体集管理器之间的聚合管理，创建一个表
  管理(employee_id, branch_name, title, manager_name)

- 如果我们愿意在表管理中存储属性manager_name的空值，那么表工作是多余的

银行企业的E-R图


branch-city
branch-name
assets
branch
account-branch
loan-branch
customer-name
customer-id
customer-street
customer-city
loan-number
amount
payment-date
payment-number
payment-amount
customer
borrower
loan
loan-payment
payment
弱实体集
access-date
account-number
balance
cust-banker
type
depositor
account
多值属性
manager
employee
works-for
worker
ISA
employee-id
employee-name
dependent-name
telephone-number
employment-length
start-date
savings-account
checking-account
interest-rate
overdraft-amount
派生属性

银行架构

- 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)(派生属性不包括在内)

- Dependent_name = (employee_id, dname)(派生自多值属性)

- Account_branch = (account_number, branch_name)
- Loan_branch = (loan_number, branch_name)
- 借款人= (customer_id, loan_number)
- 存款人= (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)(弱实体集)

- savings_account = (account_number, interest_rate) (ISA 专门化)
- checking_account = (account_number, overdraft_amount) (ISA 专门化)

大纲

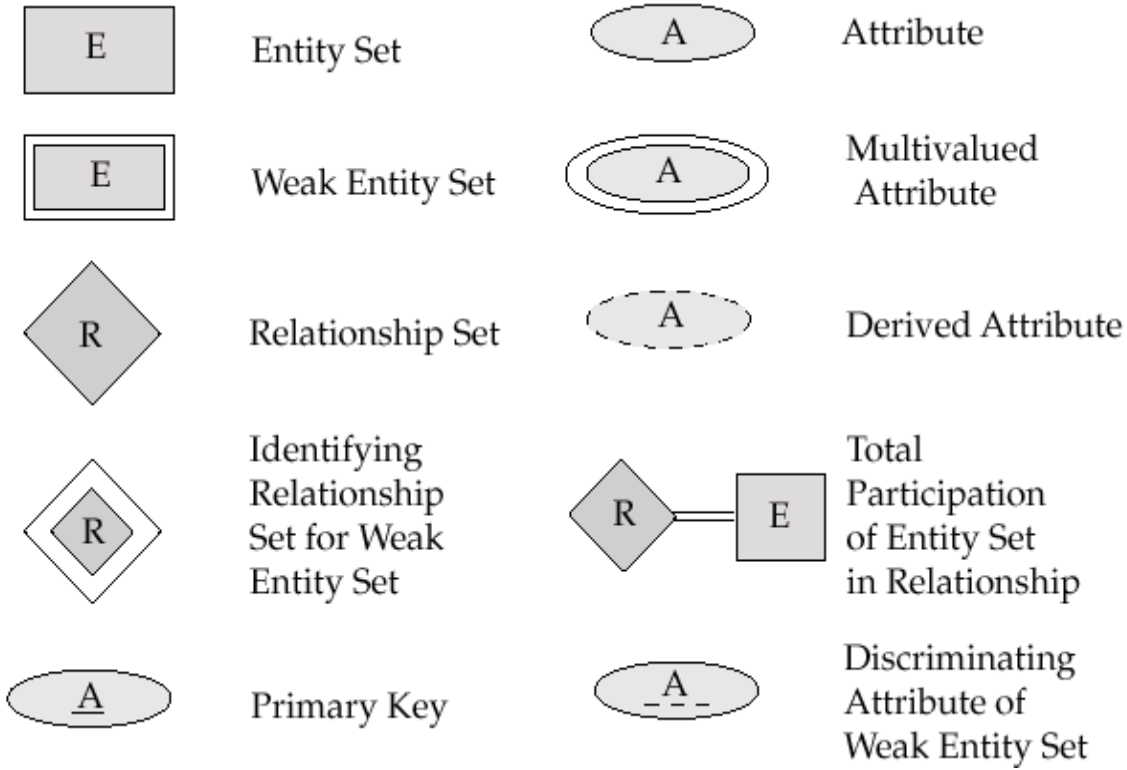
- 设计过程概述
- 实体-关系模型(英语:Entity-Relationship Model)
- 约束
- 实体-关系图(绘图)
- 还原为关系图式

☞**总结**

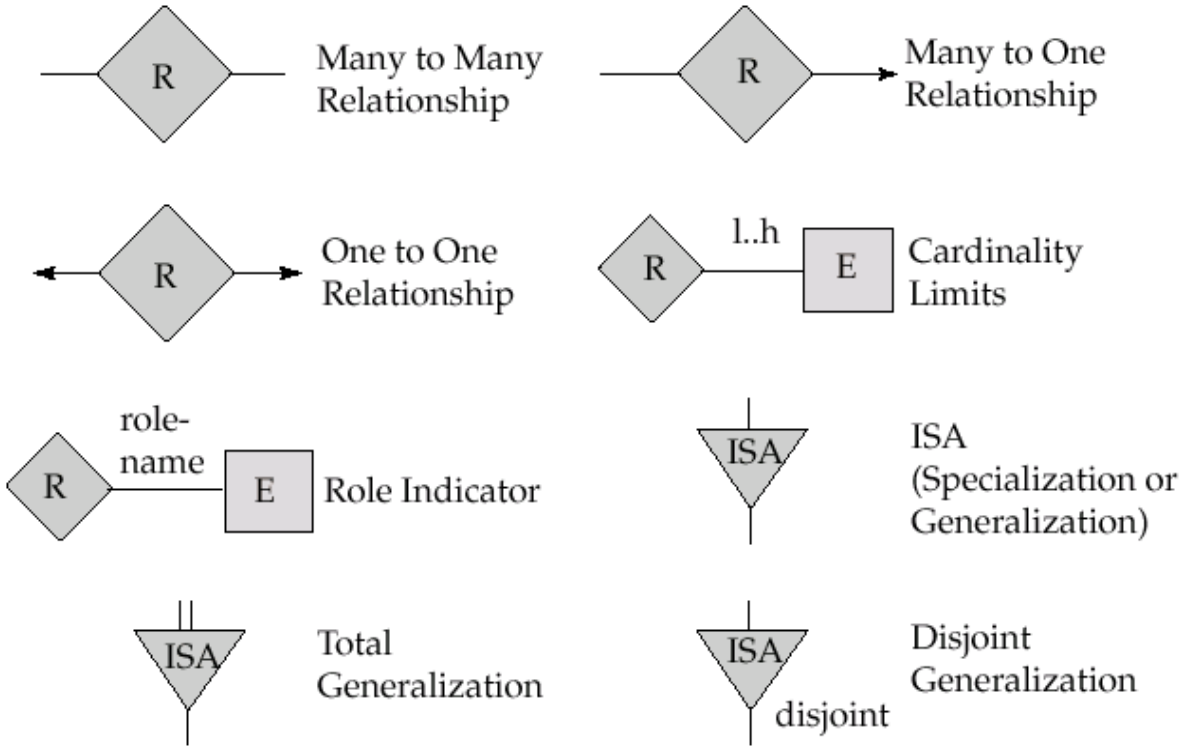
数据库设计阶段

- **需求分析**
- **概念设计(E-R模型)**
- **功能需求分析**
  - 描述将对数据执行的操作
  - 审查设计
- **逻辑实现**
  - 从概念模型到实现模型的映射
  - 例如，关系模型，OO模型
- **物理实现**
  - 指定数据库的物理特性
  - 缓冲区大小、索引……

E-R图中使用的符号


E
Entity Set
A
Attribute
E
Weak Entity Set
A
Multivalued Attribute
R
Relationship Set
A
Derived Attribute
R
Identifying Relationship Set for Weak Entity Set
R
E
Total Participation of Entity Set in Relationship
A
Primary Key
A
Discriminating Attribute of Weak Entity Set

E-R图中使用的符号(续)


R
Many to Many Relationship
R
Many to One Relationship
R
One to One Relationship
R
l..h
E
Cardinality Limits
R
role-name
E
Role Indicator
ISA
ISA (Specialization or Generalization)
ISA
Total Generalization
ISA
disjoint
Disjoint Generalization

设计工具

- **Rational Rose**
  - http://www-306.ibm.com/software/rational/
- **矢量绘图软件企业**
  - http://www.microsoft.com/china/office/xp/visio/default.asp
- **欧文**
  - http://www3.ca.com/Solutions/Product.asp?ID=260
- **电力设计**
  - http://www.sybase.com/products/developmentintegration/powerdesigner

ER模型总结

- **概念设计遵循需求分析**
  - 生成要存储的数据的高级描述
- **E-R模型在概念设计中比较流行**
  - 构念是表达性的，接近于人们对其应用程序的思考方式
- 基本构念:实体、关系和属性(实体和关系)
- 一些附加结构:弱实体、ISA层次结构
- 注意:ER模型有许多变体

ER模型总结(续)

- **E-R模型中的完整性约束**
  - **键约束、参与约束和ISA层次结构的重叠/覆盖约束。一些外键约束也隐含在关系集的定义中**
  - 一些约束(特别是功能依赖)不能在E-R模型中表示
  - 约束在确定企业的最佳数据库设计方面发挥着重要作用

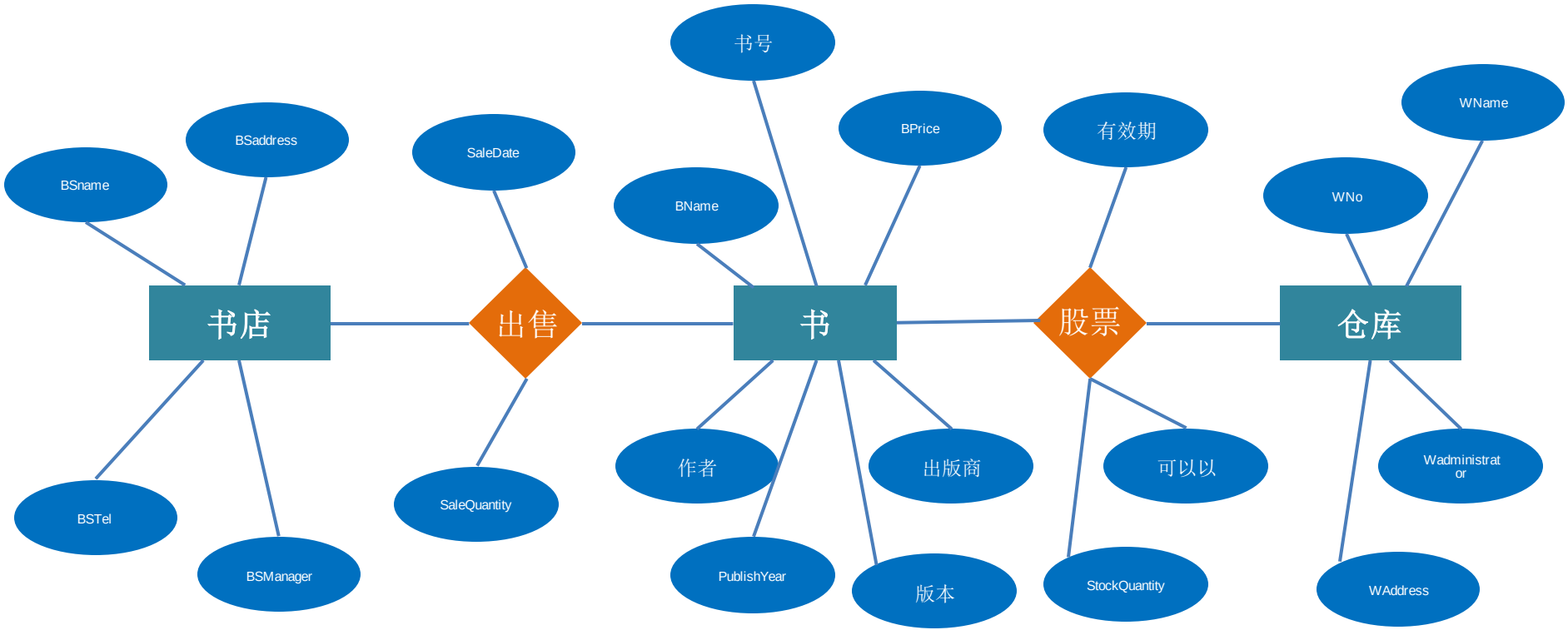
ER模型总结

- **ER设计是主观的**
  - 通常有很多方法来为给定的场景建模!分析备选方案可能会很棘手，尤其是对大型企业而言。常见的选择包括:
    - **实体vs.属性，实体vs.关系，二元关系还是n元关系，是否使用ISA层次结构**
- **确保良好的数据库设计**
  - 生成的关系模式应该进一步分析和细化
  - FD信息和归一化技术特别有用(“英文”)

补充练习

- **书店管理应用包含三个实体集**
  - 书店:BSName, BSaddress, BSTel, BSManager
  - 图书:书号，书名，价格，作者，出版商，出版年份，版本
  - 仓库:WNo、WName、WAddress、Wadministrator
- **两个关系集**
  - 销售(书店和图书):SaleDate和salequty
  - 库存(书和仓库):InDate, InPrice和StockQuantity
- **问题:给出对应的ER图和关系模式,并指出主码和外码**

ER图图


书店
出售
书
股票
仓库
BSname
BSaddress
BSTel
BSManager
SaleDate
SaleQuantity
书号
BName
BPrice
作者
出版商
PublishYear
版本
有效期
可以以
StockQuantity
WName
WNo
Wadministrat
or
WAddress

关系模式

- **关系模式&主键**
  - 书店(BSName，BSaddress，BSTel，BSManager)
  - 图书(书号，书名，价格，作者，出版商，出版年份，版本)
  - 仓库(WNo，WName，WAddress，Wadministrator)
  - 销售(BSName，书号，销售日期，销售数量)
  - 库存(书号，WNo，InDate，InPrice，StockQuantity)
- **外键**
  - 销售参考BSName和书号上的Bookstore和Book
  - 库存参考书和仓库分别在书号和WNo上

第6讲结束