2024年春季学期

数据库系统概论

An Introduction to Database Systems

第七章　数据库设计

中国科学技术大学 大数据学院

黄振亚，huangzhy@ustc.edu.cn

http://staff.ustc.edu.cn/~huangzhy/Course/DB2024.html

第七章 数据库设计

数据库设计概述

- 数据库设计
  - 数据库设计是指对于一个给定的应用环境，构造（设计）优化的数据库逻辑模式和物理结构，并据此建立数据库及其应用系统，使之能够有效地存储和管理数据，满足各种用户的应用需求，包括信息管理要求和数据操作要求。
  - 信息管理要求：在数据库中应该存储和管理哪些数据对象
  - 数据操作要求：对数据对象需要进行哪些操作，如查询、增、删、改、统计等操作

数据库设计概述

- 数据库设计
  - 目标：为用户和各种应用系统提供一个信息基础设施和高效率的运行环境
  - 高效率的运行环境
    - 数据库数据的存取效率高
    - 数据库存储空间的利用率高
    - 数据库系统运行管理的效率高

7.1 数据库设计概述

7.1.1 数据库设计的特点

7.1.2 数据库设计方法

7.1.3 数据库设计的基本步骤

7.1.4 数据库设计过程中的各级模式

7.1.1 数据库设计的特点

- 1. 数据库建设的基本规律
  - **三分技术，七分管理，十二分基础数据**
  - 技术
    - 设计工具
    - 开发技术工具
  - 管理
    - 数据库建设项目管理
    - 企业（即应用部门）的业务管理
  - **基础数据**
    - **数据收集、整理、组织**
    - **数据不断更新**

7.1.1 数据库设计的特点

- 2. 结构（数据）设计和行为（处理）设计相结合
  - 将数据库结构设计和数据处理设计密切结合

- 结构和行为分离的设计
  - 传统的软件工程：**重 行为设计**
    - 忽视对应用中数据语义的分析和抽象，只要有可能就尽量推迟数据结构设计的决策
  - 早期的数据库设计：**重 结构设计**
    - 致力于数据模型和数据库建模方法研究，忽视了行为设计对结构设计的影响

数据库设计的特点（续）

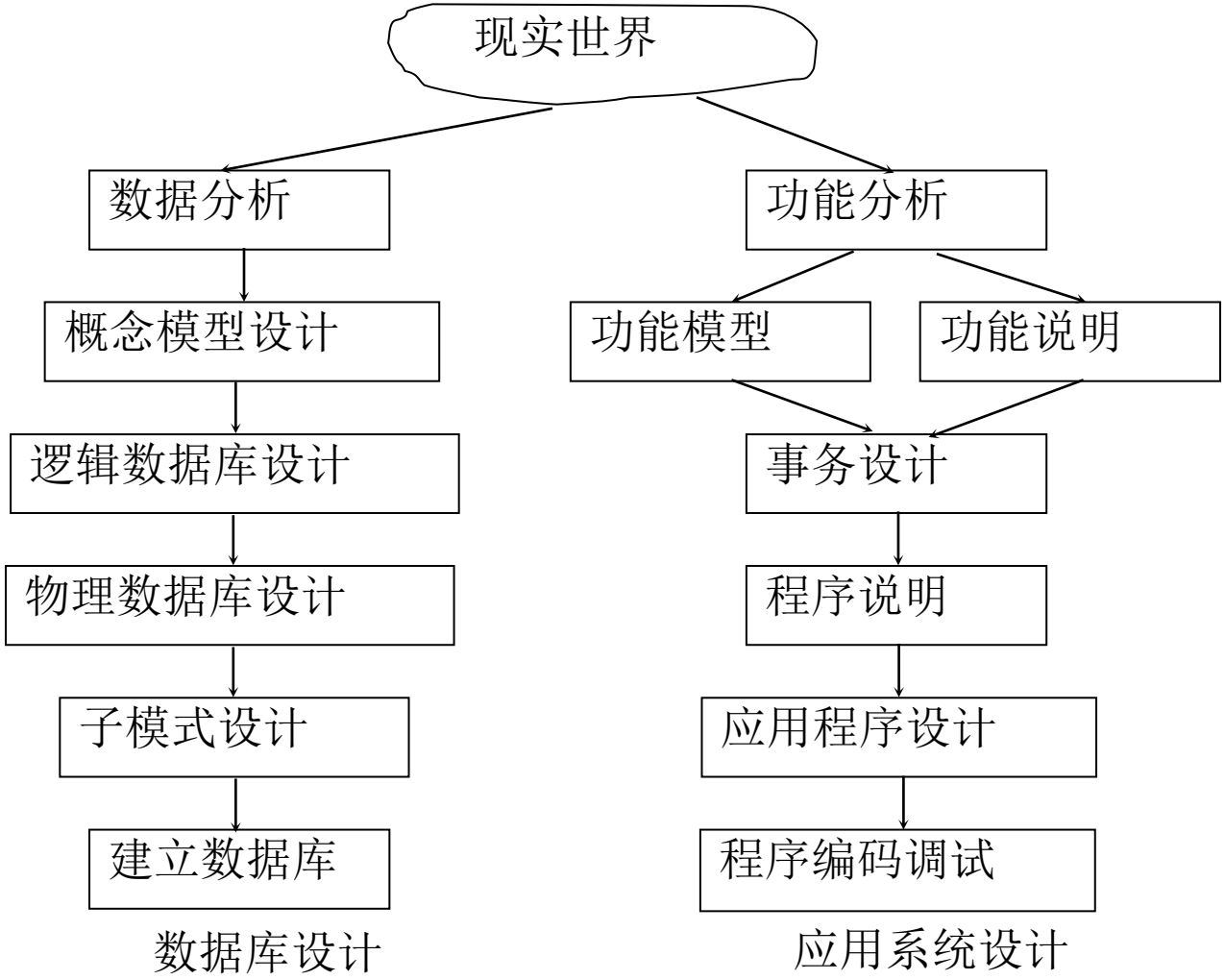


图7.1 结构（数据库结构）和行为（数据处理）分离的设计

7.1 数据库设计概述

7.1.1 数据库设计的特点

7.1.2 数据库设计方法

7.1.3 数据库设计的基本步骤

7.1.4 数据库设计过程中的各级模式

7.1.2 数据库设计方法

- 大型数据库设计是涉及多学科的综合性技术，又是一项庞大的工程项目。
- 它要求多方面的知识和技术。主要包括：
  - 计算机的基础知识
  - 软件工程的原理和方法
  - 程序设计的方法和技巧
  - 数据库的基本知识
  - 数据库设计技术
  - **应用领域的知识**

7.1.2 数据库设计方法

- 手工与经验相结合方法
  - 设计质量与设计人员的经验和水平有直接关系
  - 缺乏科学理论和工程方法的支持，工程的质量难以保证
  - 数据库运行一段时间后常常不同程度地发现各种问题，增加了维护代价
- 规范设计法
  - 基本思想：过程迭代和逐步求精

数据库设计方法（续）

- 规范设计法
  - 新奥尔良（New Orleans）方法
    - 将数据库设计分为若干阶段和步骤
    - 需求分析、概念设计、逻辑设计和物理设计
  - **基于E-R模型的数据库设计方法**
    - **概念设计阶段广泛采用**
  - 3NF（第三范式）的设计方法
    - 逻辑阶段可采用的有效方法
  - ODL（Object Definition Language）方法
    - 面向对象的数据库设计方法
  - 统一建模语言（UML）方法

数据库设计方法（续）

- 计算机辅助设计（CASE工具--Computer Aided/Assisted Software Engineering）
  - ORACLE Designer 2000
  - SYBASE PowerDesigner
    - 数据库建模UML工具
  - erwin Data Modeler
  - Navicat

7.1 数据库设计概述

7.1.1 数据库设计的特点

7.1.2 数据库设计方法

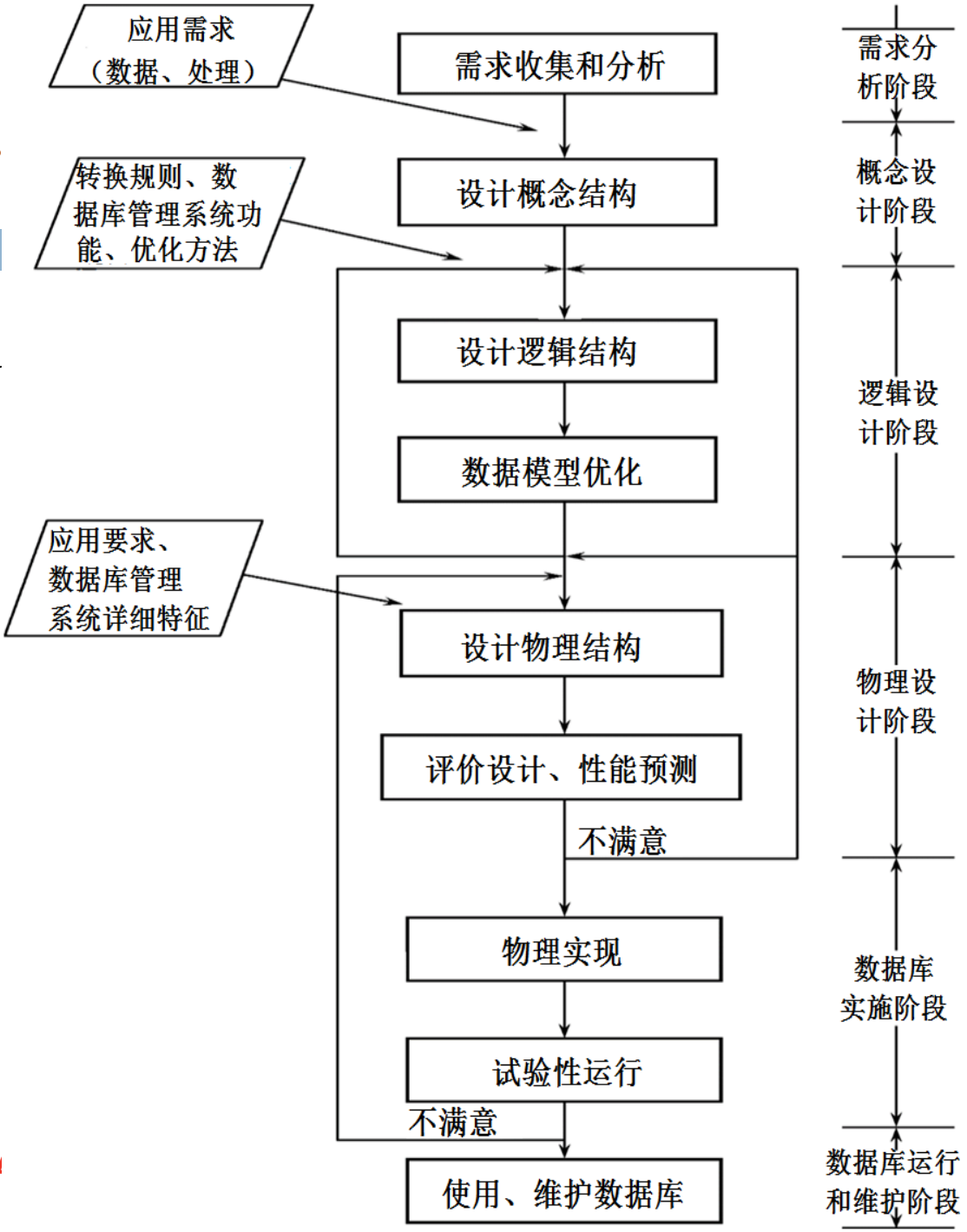
7.1.3 数据库设计的基本步骤

7.1.4 数据库设计过程中的各级模式

7.1.3 数据库设计的基本步骤

- 数据库设计分6个阶段
  - 需求分析
  - 概念结构设计
  - 逻辑结构设计
  - 物理结构设计
  - 数据库实施
  - 数据库运行和维护
- 需求分析和概念设计独立于任何数据库管理系统
- 逻辑设计和物理设计与选用的DBMS密切相关

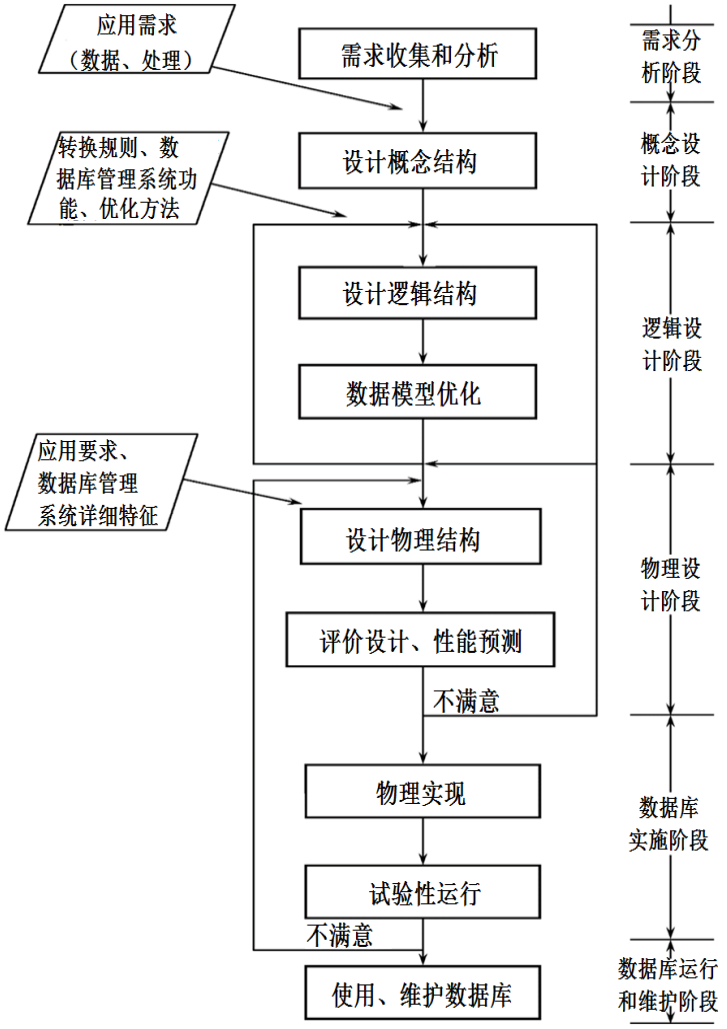
- 数据库设


应用需求
（数据、处理）
需求收集和分析
需求分析阶段
转换规则、数据库管理系统功能、优化方法
设计概念结构
概念设计阶段
设计逻辑结构
数据模型优化
逻辑设计阶段
应用要求、数据库管理系统详细特征
设计物理结构
评价设计、性能预测
物理设计阶段
不满意
物理实现
试验性运行
数据库实施阶段
不满意
使用、维护数据库
数据库运行和维护阶段

7.1.3 数据库设计的基本步骤（续）

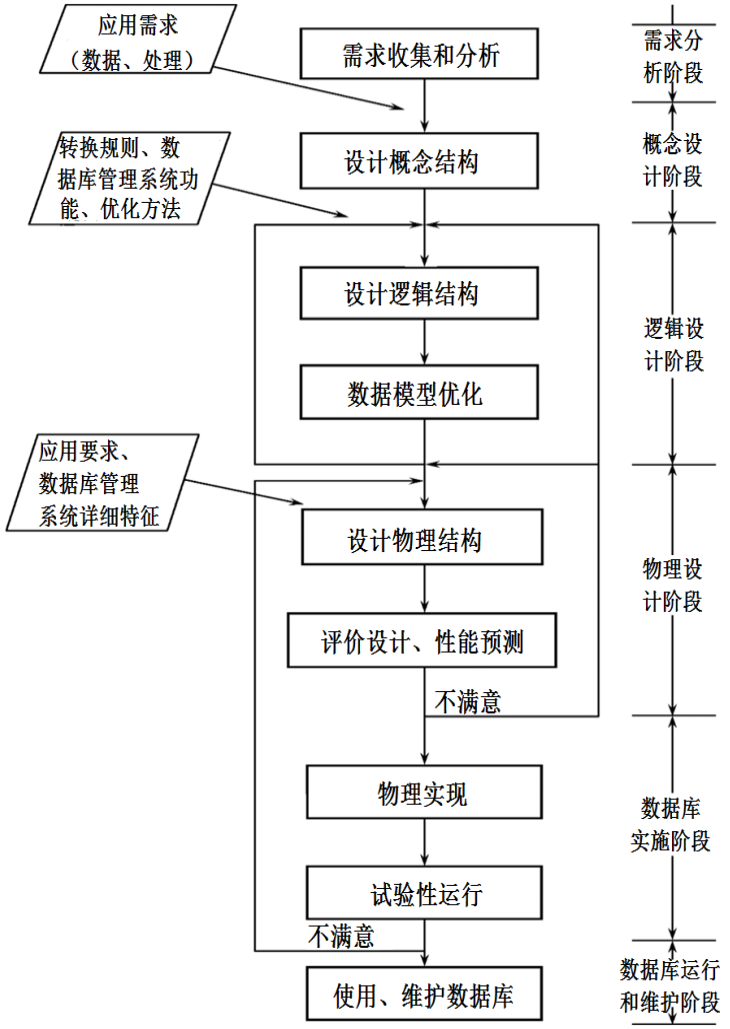
二、数据库设计的过程(六个阶段)

- **1. 需求分析阶段**
  - 准确了解与分析用户需求
    - 包括数据与处理
  - 是否做得充分与准确，决定了构建数据库的速度和质量
  - 最困难、最耗费时间的一步



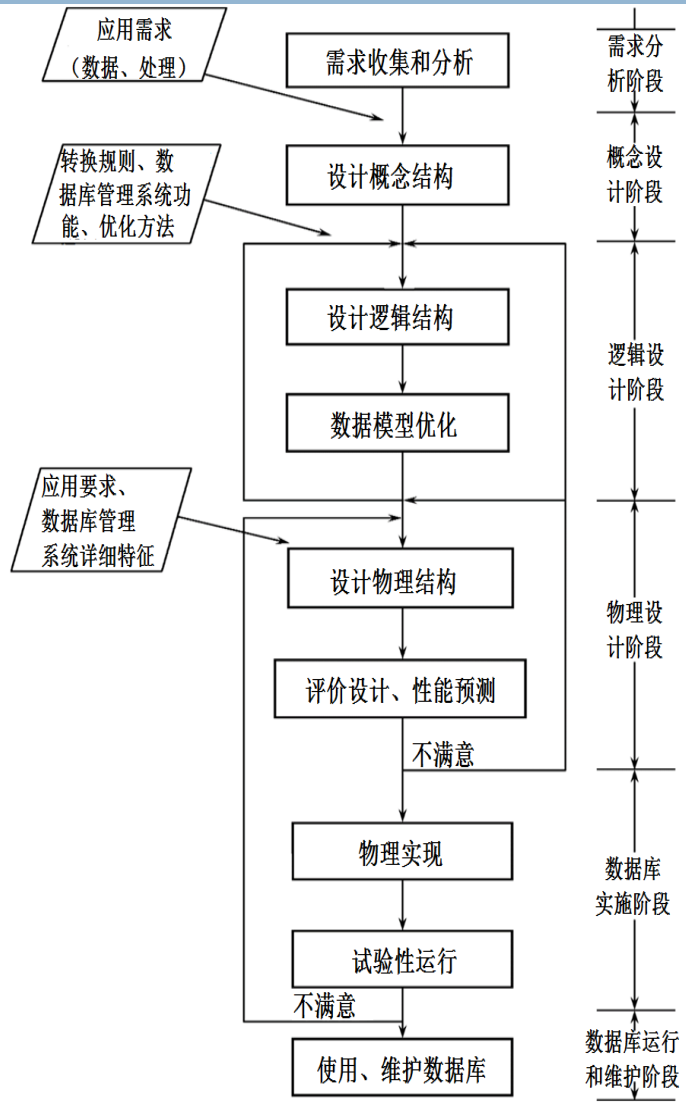
7.1.3 数据库设计的基本步骤（续）

- 2.概念结构设计阶段
  - 整个数据库设计的关键
  - 通过对用户需求进行综合、归纳与抽象，形成一个独立于具体DBMS的概念模型
- 3.逻辑结构设计阶段
  - 将概念结构转换为某个DBMS所支持的数据模型
  - 对其进行优化
- 4.数据库物理设计阶段
  - 为逻辑数据模型选取一个最适合应用环境的物理结构
    - 包括存储结构和存取方法



7.1.3 数据库设计的基本步骤（续）

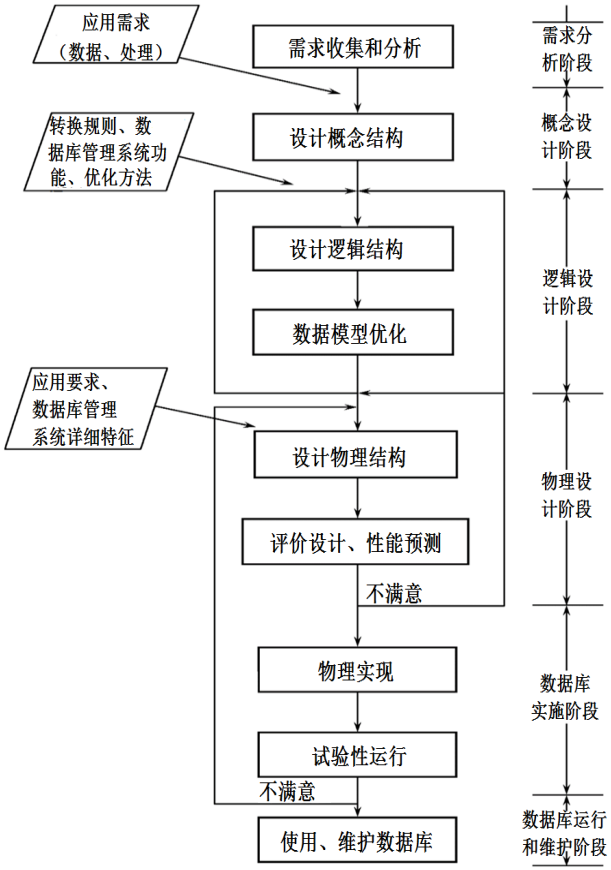
- 5.数据库实施阶段
  - 运用DBMS数据库语言（如SQL）及应用开发语言，根据逻辑设计和物理设计的结果，构建数据库
    - 建立数据库
    - 编制与调试应用程序
    - 组织数据入库
    - 进行试运行
- 6.数据库运行和维护阶段
  - 数据库应用系统经过试运行后即可投入正式运行
  - 在DBS运行过程中必须不断地对其进行评价、调整与修改



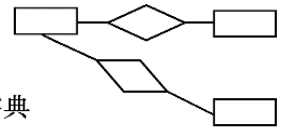
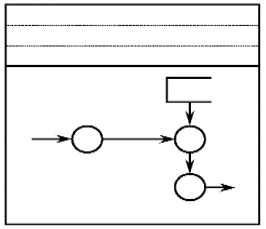
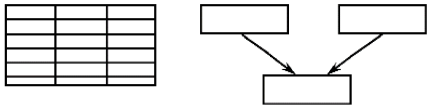
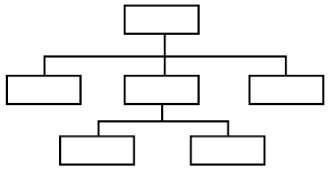
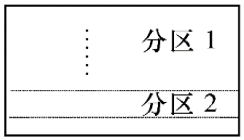
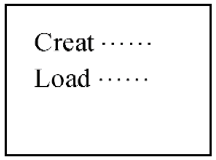
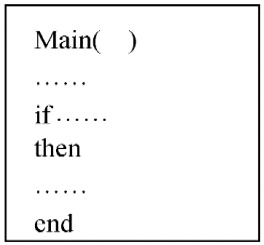
7.1.3 数据库设计的基本步骤（续）

完善的数据库应用系统需要上述六个阶段的不断反复（P208 图7.2）

- 把**数据库设计**和对**数据库中数据处理**的设计紧密结合起来
- 将这两个方面的需求分析、抽象、设计、实现在各个阶段同时进行，相互参照，相互补充，以完善两方面的设计



7.1.3

设计阶段	设计描述	
	数据	处理
需求分析	数据字典、全系统中数据项、数据流、数据存储的描述	数据流图和判定表（判定树）、数据字典中处理过程的描述
概念结构设计	概念模型（E-R图）  数据字典	系统说明书包括： ① 新系统要求、方案和概图 ② 反映新系统信息流的数据流图 
逻辑结构设计	某种数据模型 关系 非关系 	系统结构图 （模块结构） 
物理设计	存储安排 方法选择 存取路径建立 	模块设计 IPO 表 
数据库实施阶段	编写模式 装入数据 数据库试运行 	程序编码、编译联结、测试 
数据库运行和维护	性能监测、转储 /恢复 数据库重组和重构	新旧系统转换、运行、维护（修正性、适应性、改善性维护）

7.1 数据库设计概述

7.1.1 数据库设计的特点

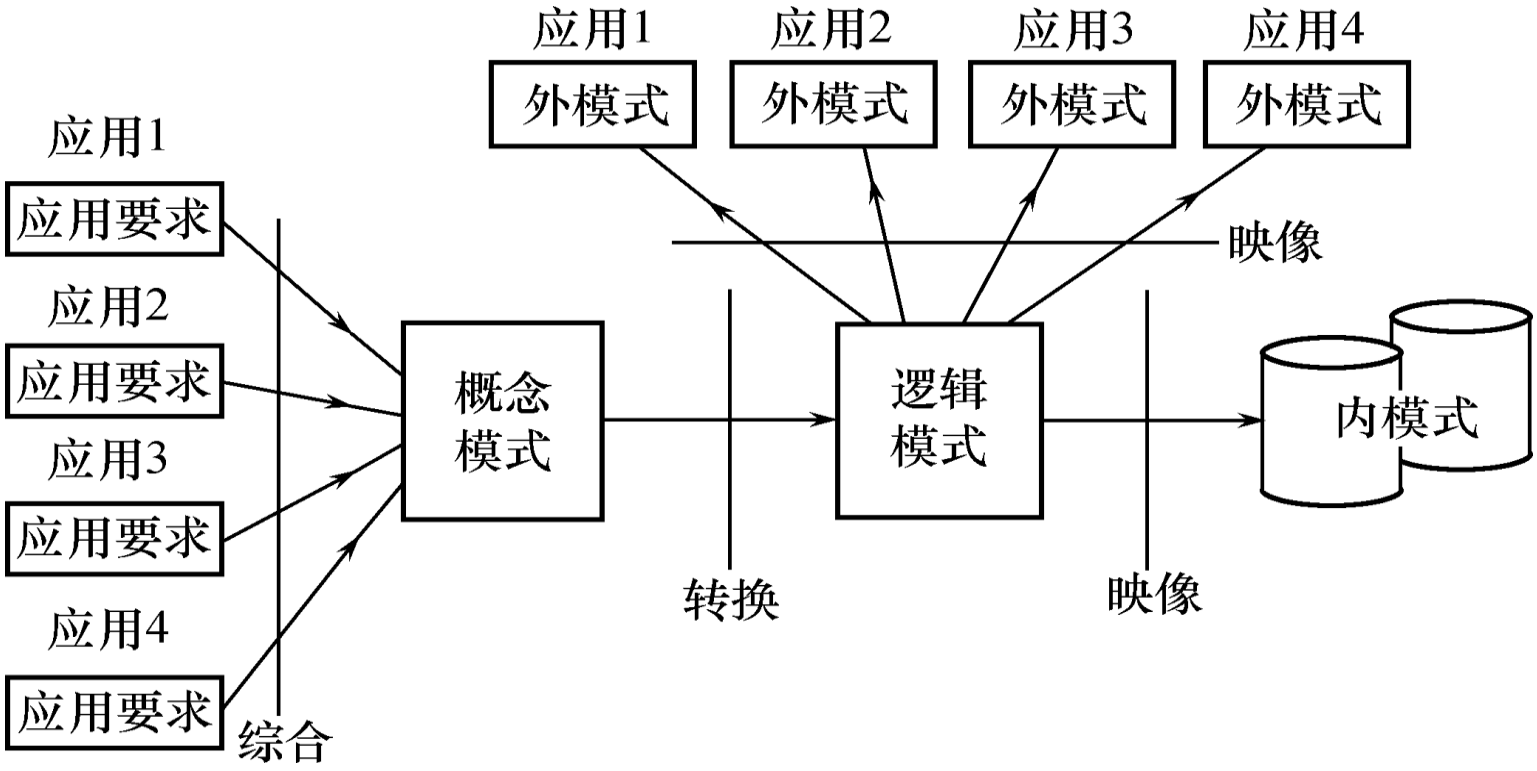
7.1.2 数据库设计方法

7.1.3 数据库设计的基本步骤

7.1.4 数据库设计过程中的各级模式

7.1.4 数据库设计过程中的各级模式

数据库设计不同阶段形成的数据库各级模式



需求分析阶段
综合用户应用需求

概念结构设计阶段
E-R图

逻辑结构设计阶段
逻辑模式（数据模型）
外模式（视图）

物理结构设计阶段
内模式（存储、索引）

第七章 数据库设计

7.2 需求分析

7.2.1 需求分析的任务

7.2.2 需求分析的方法

7.2.3 数据字典（output之一）

7.2 需求分析

- **需求分析：分析用户的要求**
  - 是设计数据库的起点
  - 结果是否准确地反映了用户的实际要求，将直接影响到后面各个阶段的设计，并影响到设计结果是否合理和实用
  - **需求分析常常被忽视**
    - 设计（开发）人员急于具体设计（开发）
    - 用户嫌麻烦

7.2.1 需求分析的任务

- 需求分析的任务
- 需求分析的重点
- 需求分析的难点

需求分析的任务

- 详细调查现实世界要处理的对象（组织、部门、企业等）
- 充分了解原系统（手工系统或计算机系统）工作状况
- 明确用户的各种需求
- **确定新系统的功能**
  - **新系统必须充分考虑今后可能的扩充和改变**

需求分析的重点

- 调查的重点是“数据”和“处理”，获得用户对数据库要求
  - 信息要求
    - 用户需要从数据库中获得信息的内容与性质
    - 由信息要求可以导出数据要求，即在数据库中需要存储哪些数据
  - 处理要求
    - 用户要完成的处理功能
    - 对处理性能的要求
  - 安全性与完整性要求

需求分析的难点

- 确定用户最终需求是困难的
  - 用户缺少计算机知识，不能准确地表达自己的需求，他们所提出的需求往往不断地变化
  - 设计人员缺少用户的专业知识，不易理解用户的真正需求，甚至误解用户的需求

7.2 需求分析

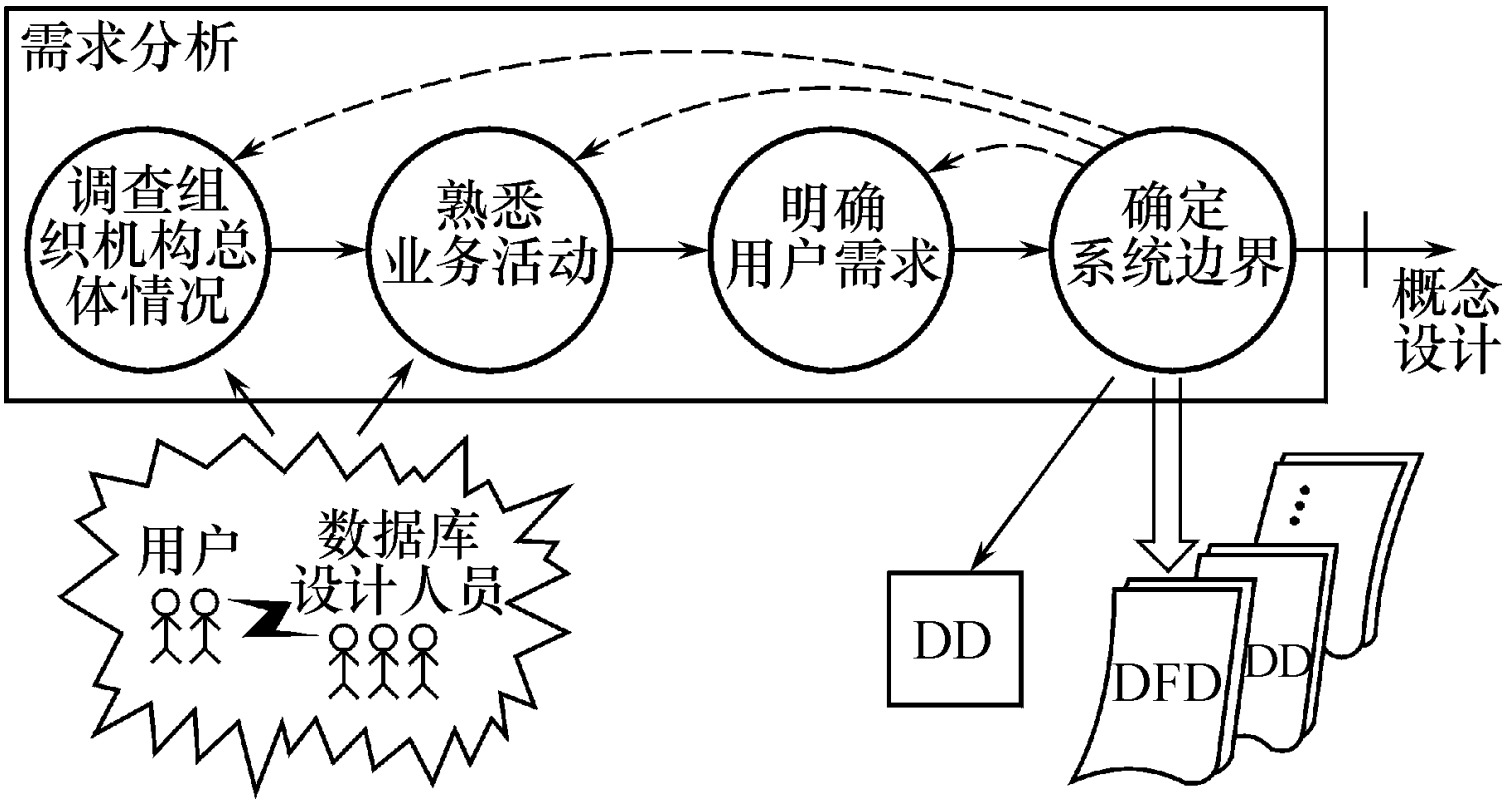
7.2.1 需求分析的任务

7.2.2 需求分析的方法

7.2.3 数据字典

需求分析过程

需求分析过程



DD数据字典
DFD用户需求规格说明书

7.2.2 需求分析的方法

需求分析的步骤

1. 调查（用户）需求，（与用户）达成共识

2. 分析与表达这些需求

调查用户需求

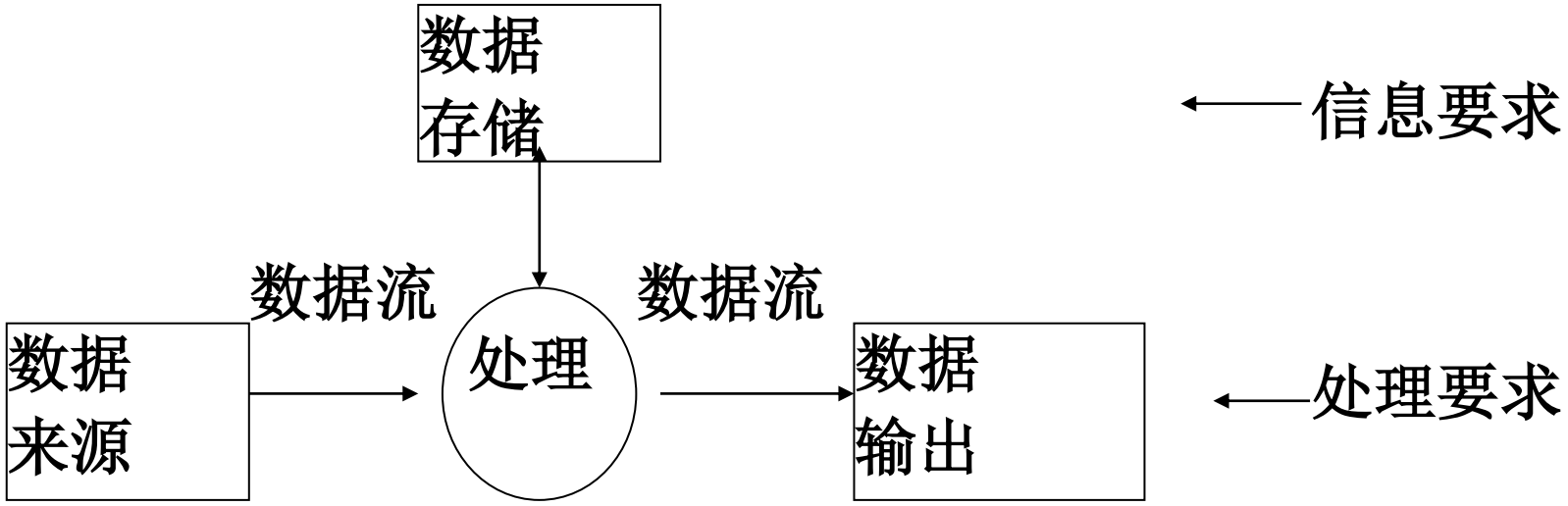
- 调查用户需求的具体步骤
  - (1) 调查组织机构情况
  - (2) 调查各部门的业务活动情况
  - (3) 在熟悉业务活动的基础上，协助用户明确对新系统的各种要求
  - **(4) 确定新系统的边界**
- 常用的调查方法
  - (1) 跟班作业
  - (2) 开调查会
  - (3) 请专人介绍
  - (4) 询问
  - (5) 设计调查表请用户填写
  - (6) 查阅记录

进一步分析和表达用户需求

- 结构化分析方法（Structured Analysis，简称SA方法）
  - 从最上层的系统组织机构入手
  - 自顶向下、逐层分解分析系统
- 对用户需求进行分析与表达后，需求分析报告必须提交给用户，征得用户的认可

进一步分析和表达用户需求（续）

- 1. 首先把任何一个系统都抽象为：



进一步分析和表达用户需求（续）

- 2．分解处理功能和数据

  (1)分解处理功能

  - 将处理功能的具体内容分解为**若干子功能**

  (2)分解数据

  - 处理功能逐步分解同时，逐级分解所用数据，形成若干层次的**数据流图**

  (3)表达方法

  - 处理逻辑：用判定表或判定树来描述
  - 数据：用数据字典来描述

- 3．将分析结果再次提交给用户，征得用户的认可

7.2 需求分析

7.2.1 需求分析的任务

7.2.2 需求分析的方法

7.2.3 数据字典

7.2.3 数据字典

- **数据字典**是关于数据库中数据的描述，即元数据，不是数据本身，是数据的数据
- 数据字典在**需求分析阶段建立**，在数据库**设计过程中不断修改、充实、完善**
- 数据字典是进行详细的数据收集和数据分析所获得的主要结果

注意：和DBMS中数据字典的区别和联系

7.2.3 数据字典

- 数据字典的用途
  - 进行详细的数据收集和数据分析所获得的主要结果
- 数据字典的内容
  - 数据项
  - 数据结构
  - 数据流
  - 数据存储
  - 处理过程
- 数据项是数据的最小组成单位
- 若干个数据项可以组成一个数据结构
- 数据字典通过对数据项和数据结构的定义来描述数据流、数据存储的逻辑内容

1. 数据项

- 数据项是不可再分的数据单位
- 对数据项的描述

数据项描述＝｛ 数据项名，数据项含义说明，别名，数据类型，长度，取值范围，取值含义，与其他数据项的逻辑关系，数据项之间的联系 ｝

  - 用**关系规范化理论**为指导，用**数据依赖**的概念分析和表示数据项之间的联系 —函数依赖
  - “取值范围”、“与其他数据项的逻辑关系”
    - 定义了**数据的完整性约束条件**，是设计数据检验功能的依据，包括模式设计、完整性检查条件、触发器，存储过程

数据字典举例

例：学生学籍管理子系统的数据字典。

- 数据项（以“学号”为例）

  数据项：　学号

  含义说明：唯一标识每个学生

  别名：　学生编号

  类型：　字符型

  长度：　8

  取值范围：00000000至99999999

  取值含义：前两位标别该学生所在年级，
  后六位按顺序编号

  与其他数据项的逻辑关系：学号确定其他数据项

数据项描述＝

｛ 数据项名，数据项含义说明，别名，数据类型，长度，取值范围，取值含义，与其他数据项的逻辑关系，数据项之间的联系 ｝

2. 数据结构

- 数据结构反映了数据之间的组合关系。
  - 一个数据结构可以由若干个数据项组成
  - 也可以由若干个数据结构组成
  - 或由若干个数据项和数据结构混合组成。
- 对数据结构的描述

  数据结构描述＝｛数据结构名，含义说明，
  组成:｛数据项或数据结构｝｝

数据字典举例（续）

- 例：数据结构（以“学生”为例）

“学生”是该系统中的一个核心数据结构：

数据结构：　学生

含义说明：　是学籍管理子系统的主体数据结构，
定义了一个学生的有关信息

组成：　学号，姓名，性别，年龄，所在系，年级

数据结构描述＝｛数据结构名，含义说明，
组成:｛数据项或数据结构｝｝

3. 数据流

- **数据流**是数据结构在系统内传输的路径。
- 对数据流的描述

  数据流描述＝｛ 数据流名，说明，数据流来源，
  数据流去向，组成:｛数据结构｝，
  平均流量，高峰期流量｝

  - 数据流来源：说明该数据流来自哪个过程
  - 数据流去向：说明该数据流将到哪个过程去
  - 平均流量：在单位时间（每天、每周、每月等）里的传输次数
  - 高峰期流量：在高峰时期的数据流量

数据字典举例（续）

- 例：数据流（“体检结果”可如下描述）

数据流：　　体检结果

说明：　　学生参加体格检查的最终结果

数据流来源：体检（处理过程）

数据流去向：批准（处理过程）

组成：　　{学号，{血常规}，{尿常规}，……}

平均流量：　平均每天200

高峰期流量：　平均每天400

数据流描述＝｛数据流名，说明，数据流来源，
数据流去向，组成:｛数据结构｝，
平均流量，高峰期流量｝

4. 数据存储

- 数据存储是数据结构停留或保存的地方，也是数据流的来源和去向之一
- 对数据存储的描述

  数据存储描述＝｛数据存储名，说明，编号，

  输入的数据流，输出的数据流，

  组成:｛数据结构｝，数据量，存取频度，存取方式｝

  - 存取频度：每小时、每天或每周存取次数，每次存取的数据量等信息
  - 存取方法：批处理 / 联机处理；检索 / 更新；顺序检索 / 随机检索
  - 输入的数据流：数据来源
  - 输出的数据流：数据去向

数据字典举例（续）

- 例：数据存储（“学生登记表”可如下描述）

数据存储：　学生登记表
说明：　记录学生的基本情况
流入数据流：　每学期5000
流出数据流：　每学期5000
组成：　{学号，姓名，性别，年龄，院系，年级，
{成绩单}，{体检单}，{获奖情况}}
数据量：　每年6000张
存取方式：　随机存取+按院系/班级打印

数据存储描述＝｛数据存储名，说明，编号，输入的数据流，输出的数据流，组成:｛数据结构｝，数据量，存取频度，存取方式｝

5. 处理过程

- 处理过程的具体处理逻辑一般用判定表或判定树来描述
- 数据字典中只需要描述处理过程的说明性信息
- 处理过程说明性信息的描述

  处理过程描述＝｛处理过程名，说明，输入:｛数据流｝，
  输出:｛数据流｝，处理:｛简要说明｝｝

  - 简要说明：说明该处理过程的功能及处理要求
    - 功能：该处理过程用来做什么
    - 处理要求：处理频度要求，如单位时间里处理多少事务，多少数据量、响应时间要求等
    - 处理要求是后面物理设计的输入及性能评价的标准

数据字典举例（续）

- 例：处理过程（“分配宿舍”可如下描述）

处理过程：分配宿舍

说明：　　为所有新生分配学生宿舍

输入：　　学生，宿舍

输出：　　宿舍安排

处理：　　在新生报到后，为所有新生分配学生宿舍。

- 要求
  - 同一间宿舍只能安排同一性别的学生
  - 同一个学生只能安排在一个宿舍中。
  - 每个学生的居住面积不小于6平方米。
  - 安排新生宿舍其处理时间应不超过15分钟。

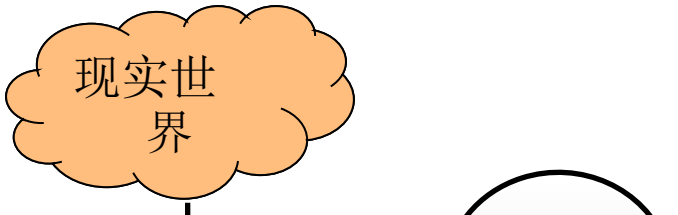
第七章 数据库设计

7.3.1 概念结构

- 什么是概念结构设计
  - 将需求分析得到的用户需求抽象为信息结构即概念模型的过程就是概念结构设计
  - 概念结构是各种数据模型的共同基础，它比数据模型更独立于机器、更抽象，从而更加稳定
  - 概念结构设计是整个数据库设计的关键

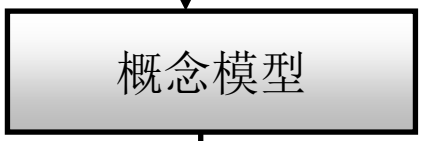
回顾：两大类数据模型 (续)

现实世界中客观对象的抽象过程



第一步

信息世界



现实世界 ⇨ 概念模型
数据库设计人员完成

第二步

概念模型 ⇨ 逻辑模型
数据库设计人员完成

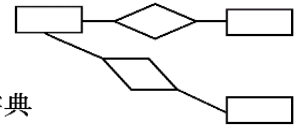
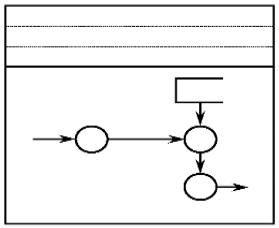
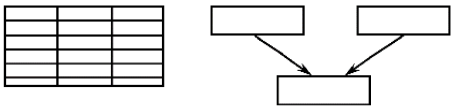
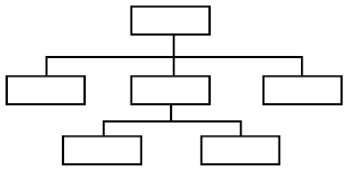
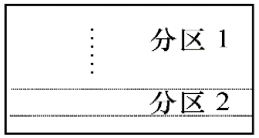
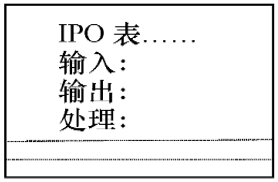
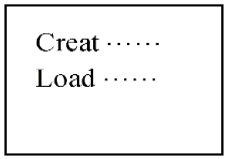
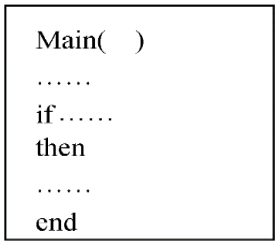
机器世界

逻辑模型 ⇨ 物理模型
由DBMS完成

第三步（自动）

看见一个动物（现实世界），命名为狗（概念模型），怎么在计算机里存（逻辑模型），存在哪里（物理模型）

数

设计阶段	设计描述	
	数据	处理
需求分析	数据字典、全系统中数据项、数据流、数据存储的描述	数据流图和判定表（判定树）、数据字典中处理过程的描述
概念结构设计	概念模型（E-R图）  数据字典	系统说明书包括： ① 新系统要求、方案和概图 ② 反映新系统信息流的数据流图 
逻辑结构设计	某种数据模型 关系 非关系 	系统结构图 （模块结构） 
物理设计	存储安排 方法选择 存取路径建立 	模块设计 IPO 表 
数据库实施阶段	编写模式 装入数据 数据库试运行 	程序编码、编译联结、测试 
数据库运行和维护	性能监测、转储 /恢复 数据库重组和重构	新旧系统转换、运行、维护（修正性、适应性、改善性维护）

概念结构（续）

- 概念结构设计的特点
  - 1.能真实、充分地反映现实世界
  - 2.易于理解: 与用户交换意见（关键：用户参与）
  - 3.易于更改：应用环境与应用要求改变时，
  - 4.易于向关系、网状、层次等各种数据模型转换
- 描述概念模型的工具
  - E-R模型

7.3 概念结构设计

7.3.1 概念结构

7.3.2 E-R模型

*7.3.3 扩展的E-R模型

*7.3.4 UML

7.3.5 概念结构设计

7.3 概念模型

- ER模型(Entity-Relationship Model)
  - 1976， Peter .P. Chen（陳品山）提出的概念设计方法
  - 以ER图的方式表达现实世界实体及实体间的联系

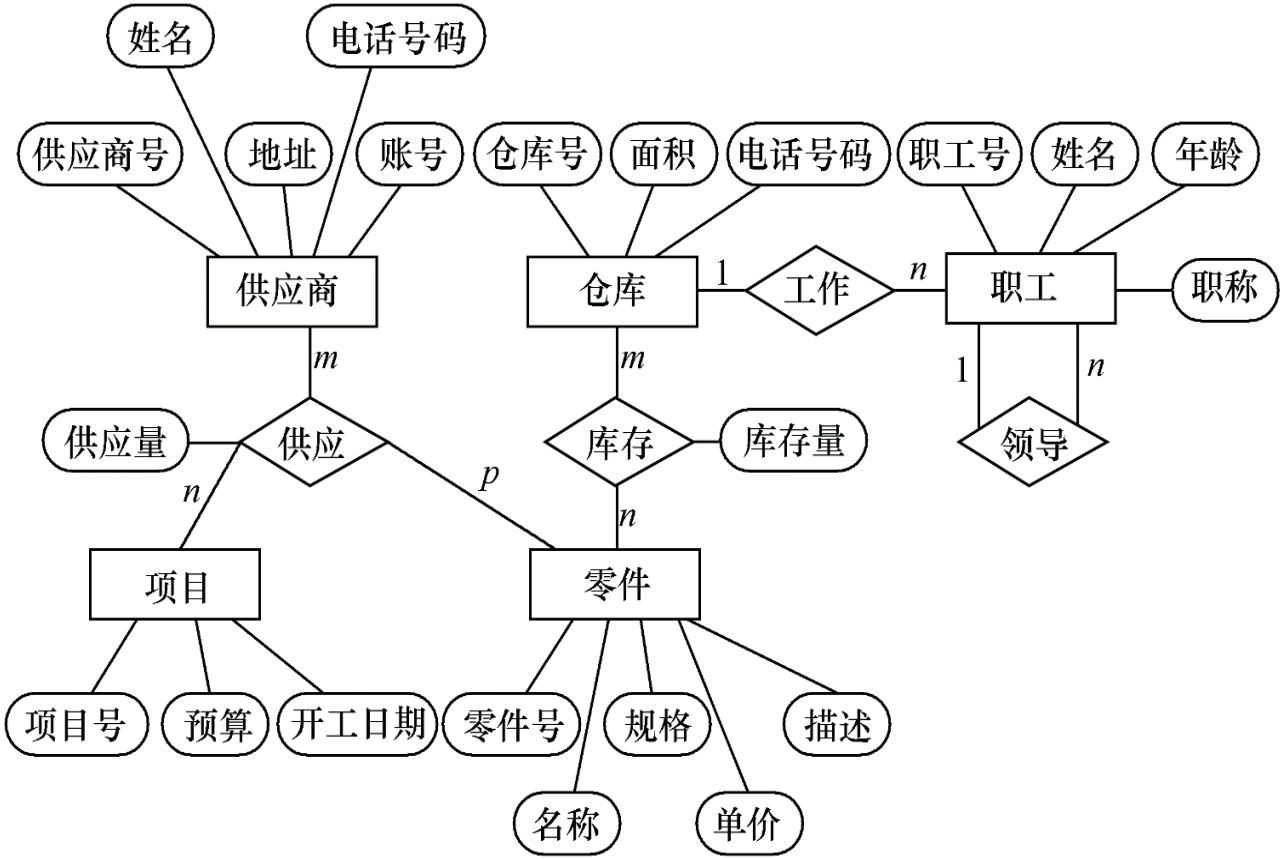
Louisiana State University

Peter Chen. The Entity-Relationship Model--Toward a Unified View of Data. *ACM Transactions on Database Systems*, Vol. 1(1), p.9-36,1976

One of the 38 most influential papers in Computer Science

7.3 概念模型

- 一个实例：工厂物资管理系统



E-R模型基本概念

(1) 实体（Entity）

客观存在并可相互区别的事物称为实体。

可以是具体的人、事、物或抽象的概念。

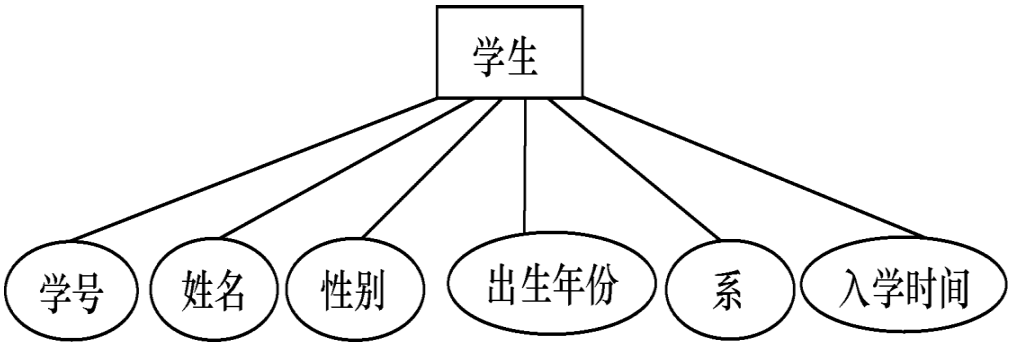
(2) 属性（Attribute）

实体所具有的某一特性称为属性。

一个实体可以由若干个属性来刻画。

(3) 码（Key）

唯一标识实体的属性集称为码。



E-R模型基本概念

(4) 域（Domain）

属性的取值范围称为该属性的域。

(5) 实体型（Entity Type）

用实体名及其属性名集合来抽象和刻画同类实体称为实体型

(6) 实体集（Entity Set）

同一类型实体的集合称为实体集

E-R模型基本概念

(7) 联系（Relationship）

- 现实世界中事物内部以及事物之间的联系在信息世界中反映为实体内部的联系和实体之间的联系
- **实体内部的联系：组成实体的各属性之间的联系**
- **实体之间的联系：不同实体集之间的联系**
- **实体之间的联系有一对一、一对多和多对多等类型**

7.3.2 E-R模型

- 现实世界，事物内部及事物之间有联系
  - **实体内部**的联系：组成实体的各属性的联系
  - **实体之间**的联系：不同实体型的实体集之间的联系

1. **实体型之间的联系**

（1）两个实体型之间的联系：

①一对一联系（1：1）

②一对多联系（1：*n*）

③多对多联系（*m*：*n*）

7.3.2 E-R模型

1. 实体型之间的联系

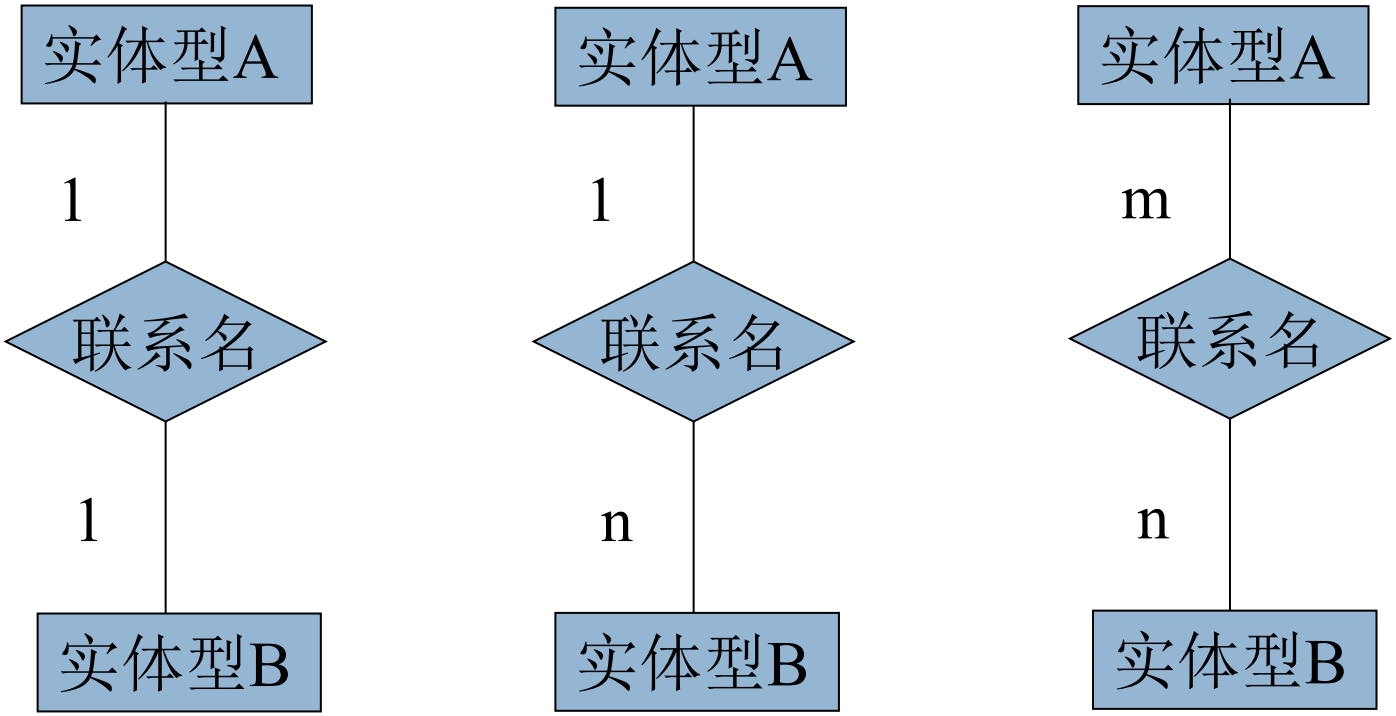


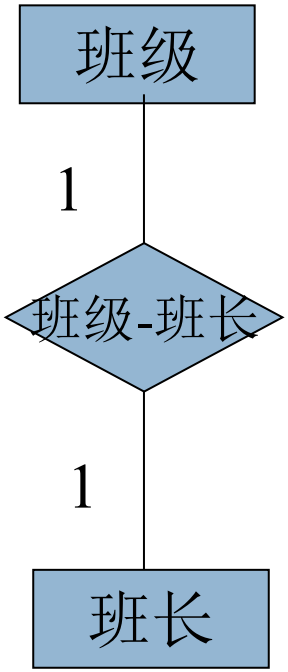
图7.6 两个实体型之间的三类联系

E-R模型（续）

① 一对一联系（1：1）

- 如果对于实体集A中的每一个实体，实体集B中至多有一个（也可以没有）实体与之联系，反之亦然，则称实体集A与实体集B具有一对一联系，记为1：1

- 例如，班级-班长联系
  - 一个班级只有一个班长
  - 一个班长只在一个班中任职

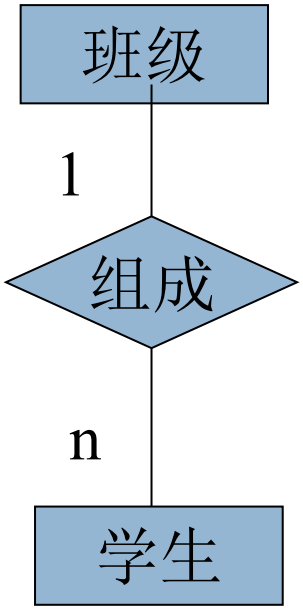


1:1联系

E-R模型（续）

② 一对多联系（1：n）

- 如果对于实体集A中的每一个实体，实体集B中有n个实体（n≥0）与之联系，反之，对于实体集B中的每一个实体，实体集A中至多只有一个实体与之联系，则称实体集A与实体集B有一对多联系，记为1：n



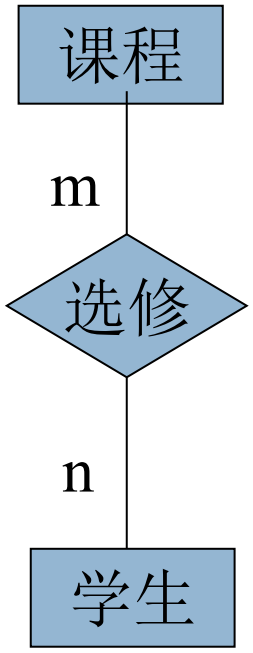
1:n联系

- 例如，班级-学生联系
  - 一个班级中有多名学生
  - 每个学生只在一个班级中学习

E-R模型（续）

③ 多对多联系（m：n）

- 如果对于实体集A中的每一个实体，实体集B中有n个实体（n≥0）与之联系，反之，对于实体集B中的每一个实体，实体集A中也有m个实体（m≥0）与之联系，则称实体集A与实体集B具有多对多联系，记为m：n
- 例如，选修联系
  - 一门课程同时有多名学生选修
  - 一个学生可以同时选修多门课程



m:n联系

E-R模型（续）

（2）两个以上的实体型之间的联系

- 两个以上的实体型之间也存在着一对一、一对多、多对多联系
- 例如，对于课程、教师与参考书3个实体型，如果
  - 一门课程可以有若干个教师讲授，使用若干本参考书
  - 每一个教师只讲授一门课程，
  - 每一本参考书只供一门课程使用，
  - 则课程与教师、参考书之间的联系是一对多的，图7.7(a)
- 例如，供应商、项目、零件3个实体型
  - 图7.7(b)

E-R模型（续）

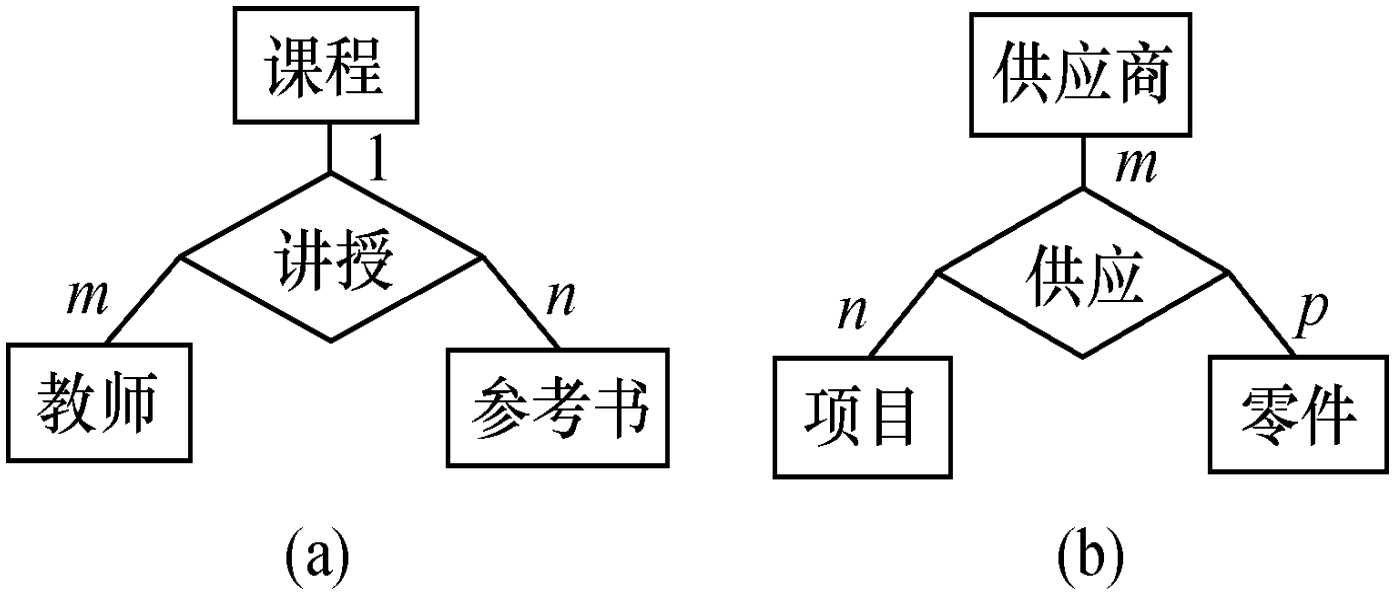


图7.7　三个实体型之间的联系示例

E-R模型（续）

（3）单个实体型内的联系

- 同一个实体集内的各实体之间也可以存在一对一、一对多、多对多的联系。
- 例如，职工实体型内部具有领导与被领导的联系，即某一职工（干部）“领导”若干名职工，而一个职工仅被另外一个职工直接领导，因此这是一对多的联系，如图7.8所示。

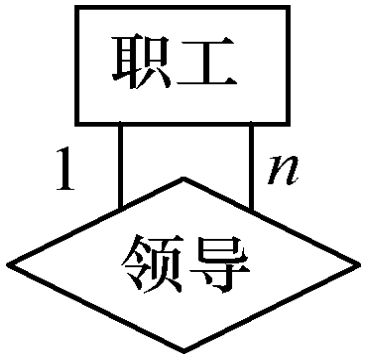
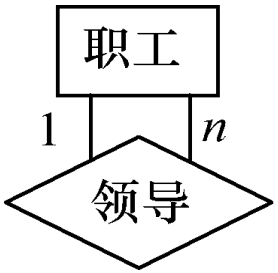
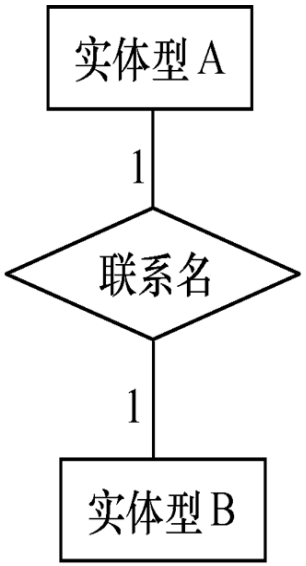


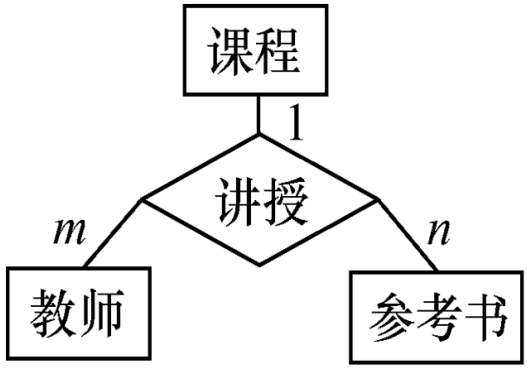
图7.8　单个实体型内的一对多联系示例

E-R模型（续）

- 联系的度：参与联系的实体型的数目
  - 2个实体型之间的联系度为2，也称为二元联系；
  - 3个实体型之间的联系度为3，称为三元联系；
  - N个实体型之间的联系度为N，也称为N元联系







E-R模型（续）

2. E-R图：表示方法

- E-R图提供了表示实体型、属性和联系的方法：
  - **实体型**：用矩形表示，矩形框内写明实体名。
  - **属性**：用椭圆形表示，并用无向边将其与相应的实体型连接起来
    - 例如，学生实体具有学号、姓名、性别、出生年份、系、入学时间等属性，用E-R图表示如图7.9所示

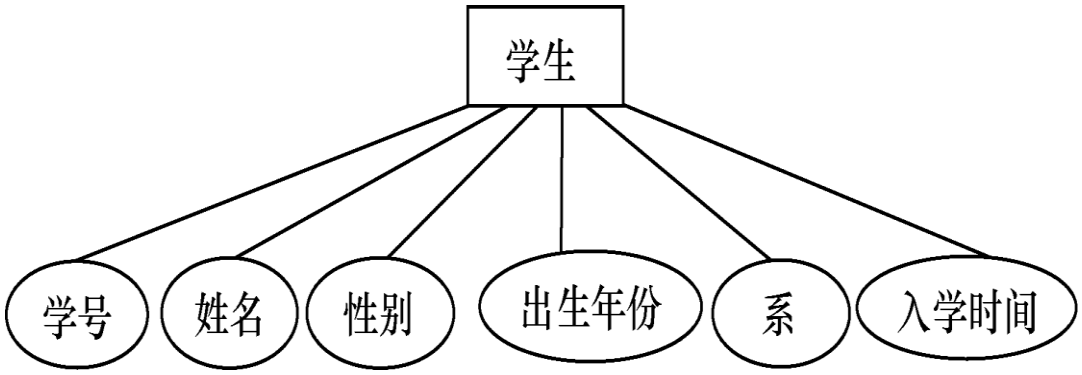


图7.9　学生实体及属性

E-R模型（续）

2. E-R图

- **联系**：用菱形表示，菱形框内写明联系名，并用无向边分别与有关实体型连接起来，同时在无向边旁标上联系的类型（1∶1，1∶n 或 m∶n）。
- 联系可以具有属性

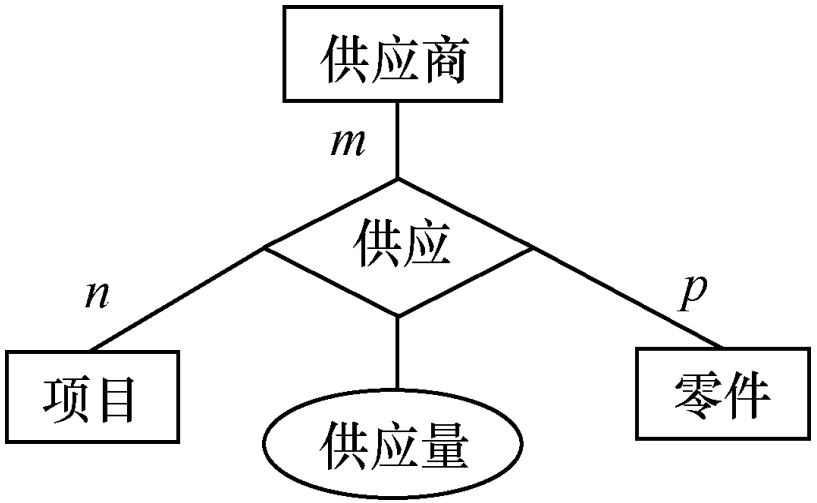
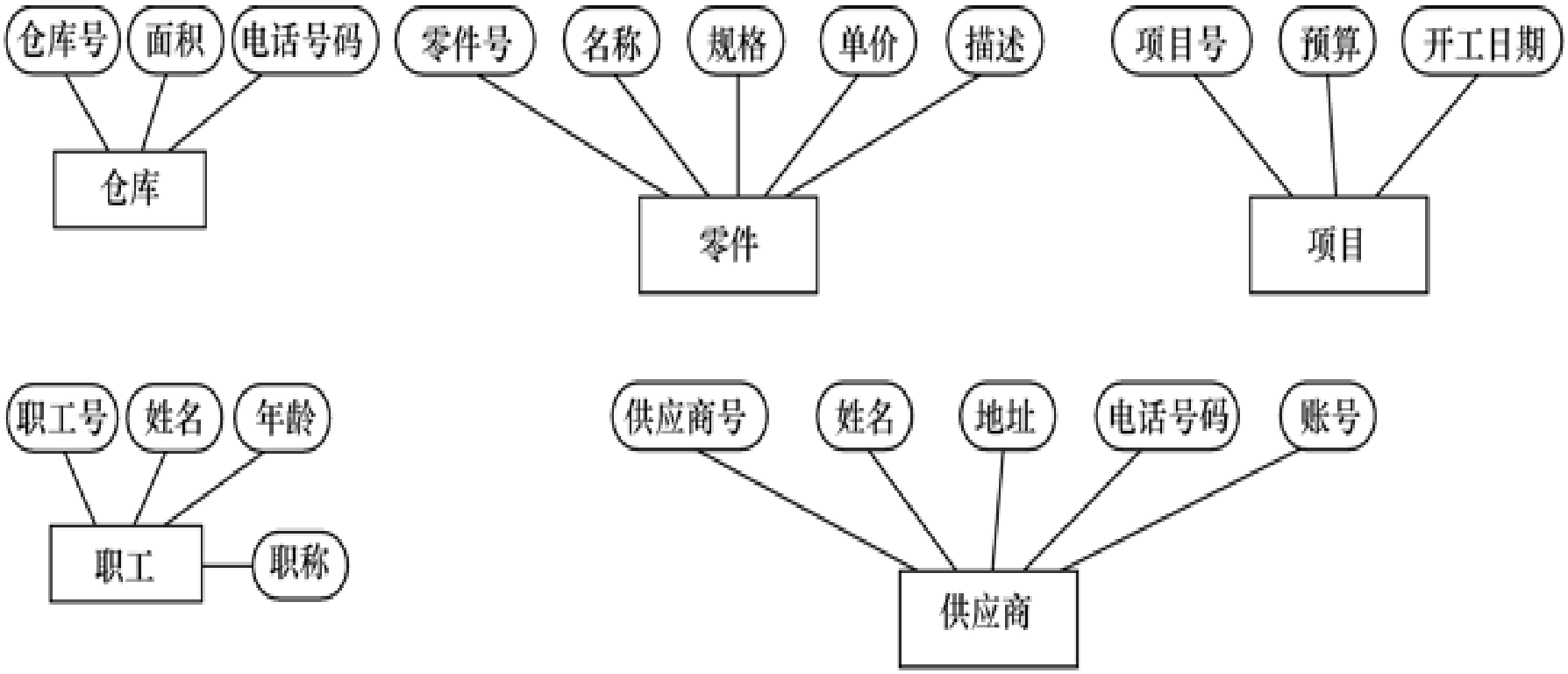


图7.10　联系的属性

E-R模型（续）

- 案例一
  - 某个工厂物资管理的概念模型。物资管理涉及的实体有
    - 仓库：属性有仓库号、面积、电话号码
    - 零件：属性有零件号、名称、规格、单价、描述
    - 供应商：属性有供应商号、姓名、地址、电话号码、账号
    - 项目：属性有项目号、预算、开工日期
    - 职工：属性有职工号、姓名、年龄、职称

E-R模型（续）

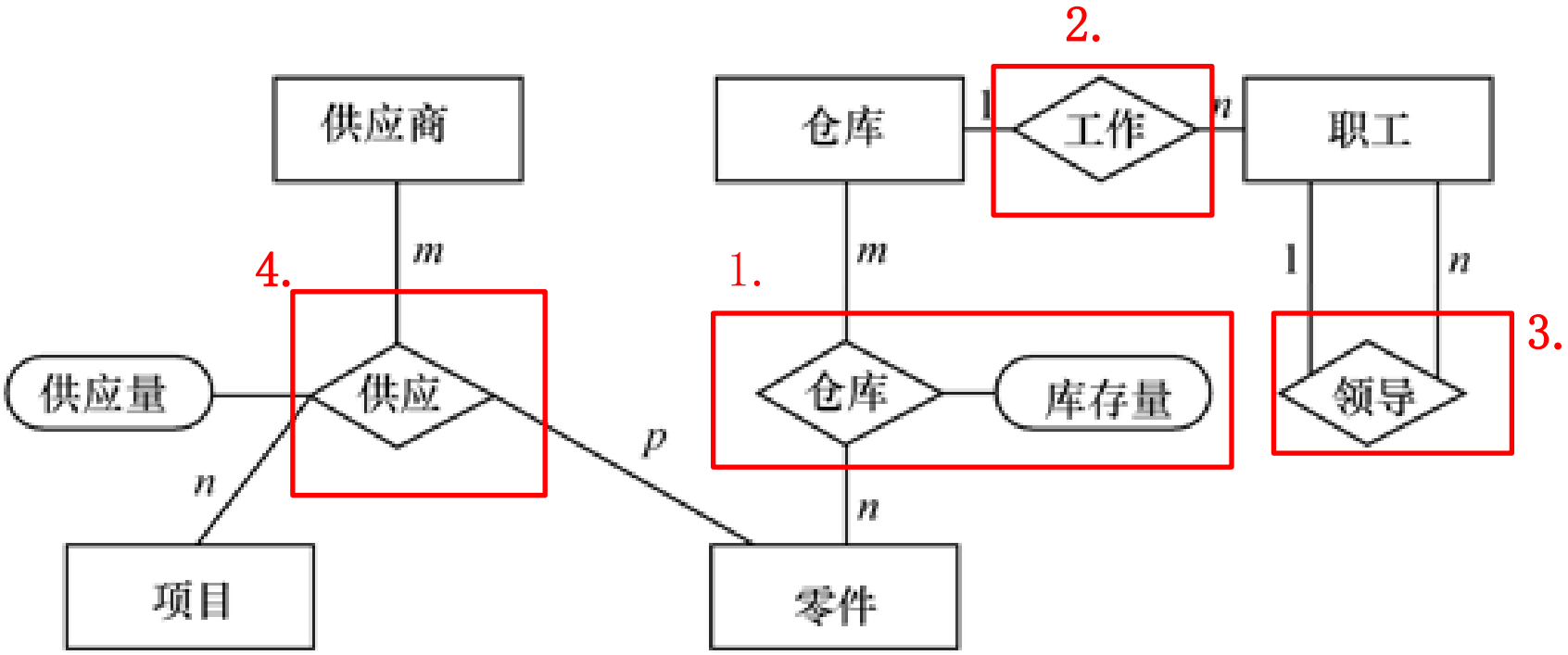


(a) 实体及其属性图

E-R模型（续）

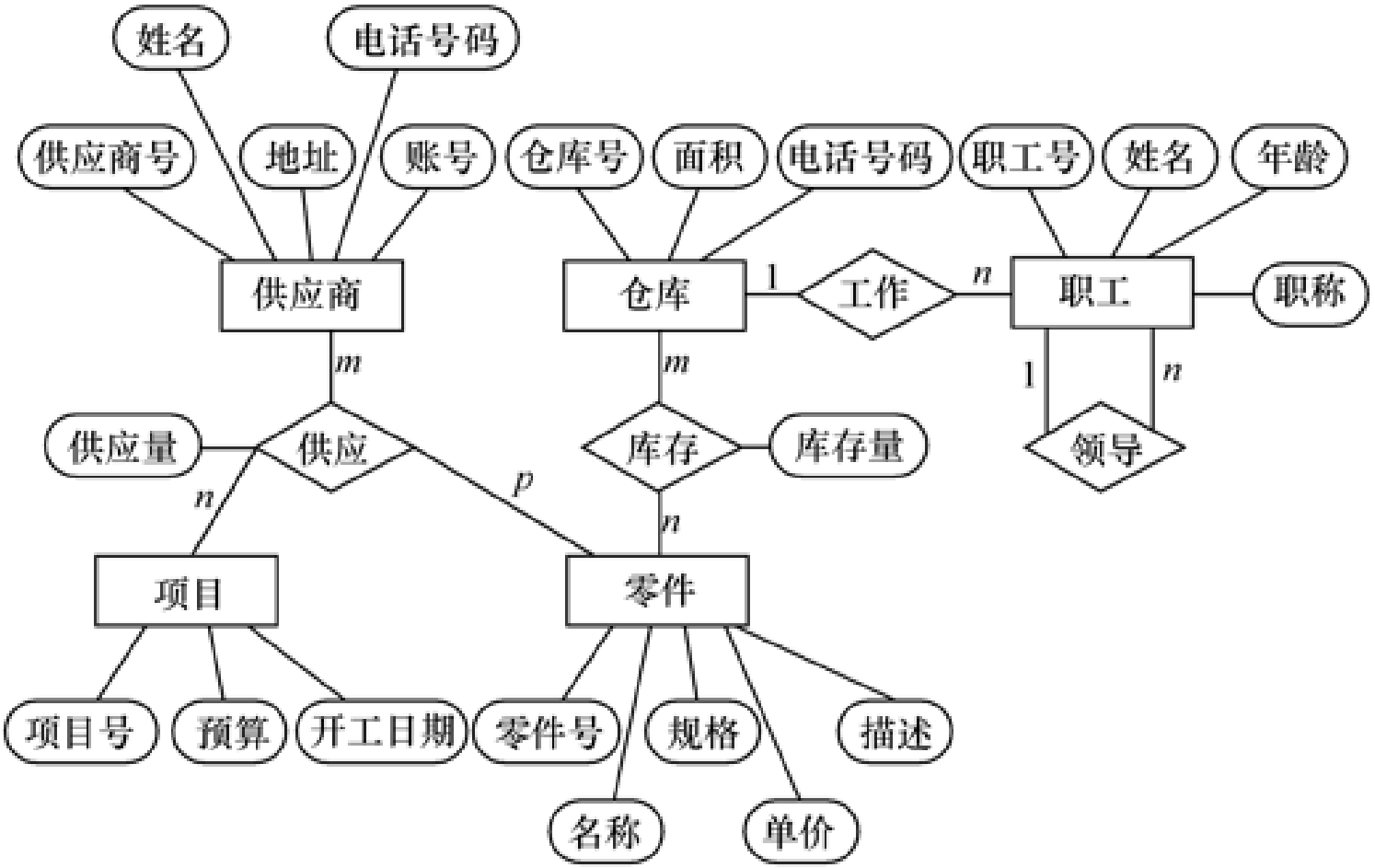
- 这些**实体之间的联系**如下
  - 1. 一个仓库可以存放多种零件，一种零件可以存放在多个仓库中，因此**仓库和零件具有多对多的联系**。用**库存量**来表示某种零件在某个仓库中的数量
  - 2. 一个仓库有多个职工当仓库保管员，一个职工只能在一个仓库工作，因此**仓库和职工之间是一对多的联系**
  - 3. 职工之间具有领导与被领导关系。即仓库主任领导若干保管员，因此**职工实体型中具有一对多的联系**
  - 4. **供应商、项目和零件三者之间具有多对多的联系**。即一 个供应商可以供给若干项目多种零件，每个项目可以使用不同供应商供应的零件，每种零件可由不同供应商供给

E-R模型（续）



(b) 实体及其联系图

E-R模型（续）



(c) 完整的实体-联系图

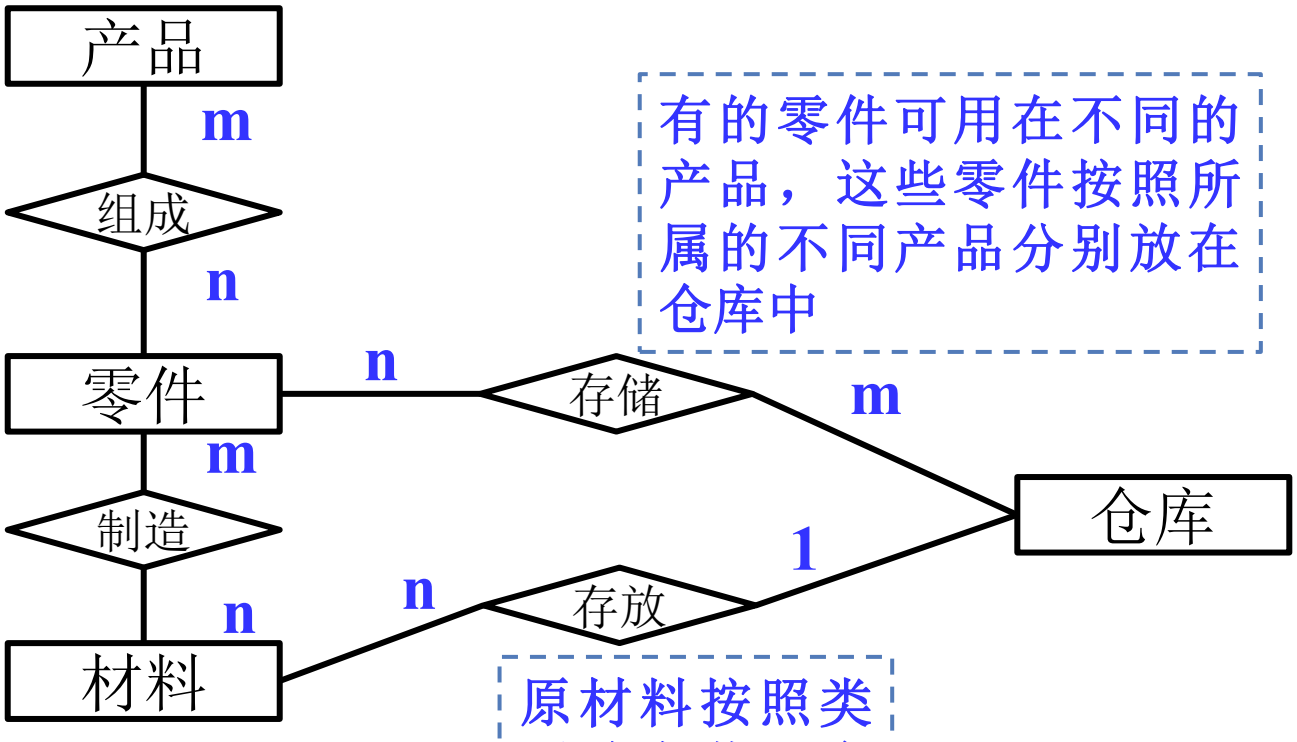
E-R模型（续）

- 案例二：
  - 某工厂生产若干产品，每种产品由不同的零件组成，有的零件可用在不同的产品。这些零件由不同的原材料制成，不同零件所用的材料可以相同。有的零件可用在不同的产品，这些零件按照所属的不同产品分别放在仓库中，原材料按照类别放在若干仓库中。
  - 请用E-R图画出此工厂产品、零件、材料、仓库的概念模型。

E-R模型：实例2（续）

每种产品由不同的零件组成，有的零件可用在不同的产品

这些零件由不同的原材料制成，不同零件所用的材料可以相同



有的零件可用在不同的产品，这些零件按照所属的不同产品分别放在仓库中

原材料按照类别放在若干仓库中

7.3 概念结构设计

7.3.3 扩展的E-R模型

- **E-R模型的不足**
  - 有些现实语义无法表达
  - 为了增强和丰富基本E-R模型的表达能力
- **对E-R模型进行扩展：**

1. ISA联系

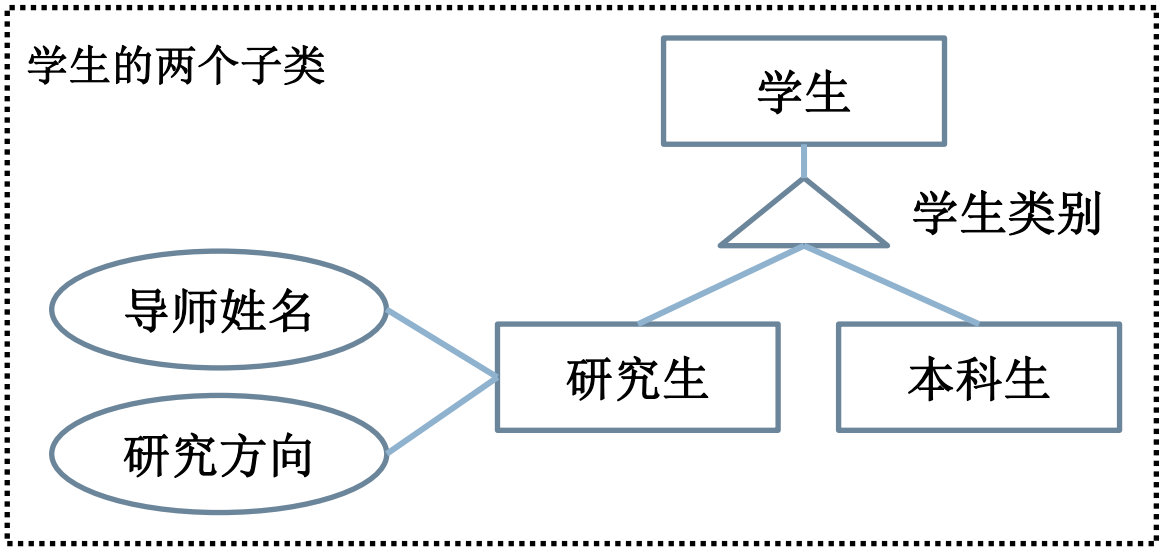
2. 基数约束

3. PART-OF联系

1. ISA 联系

1. ISA联系

- 有的实体型是某个实体型的子类型，这种父类-子类联系称为ISA联系， 表示“is a”语义。用△ 表示。
- ISA联系的性质：子类继承了父类的所有属性，子类也可以有自己的属性。



1. ISA 联系

（1）分类属性

- 分类属性是父实体型的一个属性
- 分类属性的值把父实体型中的实体分派到子实体型中

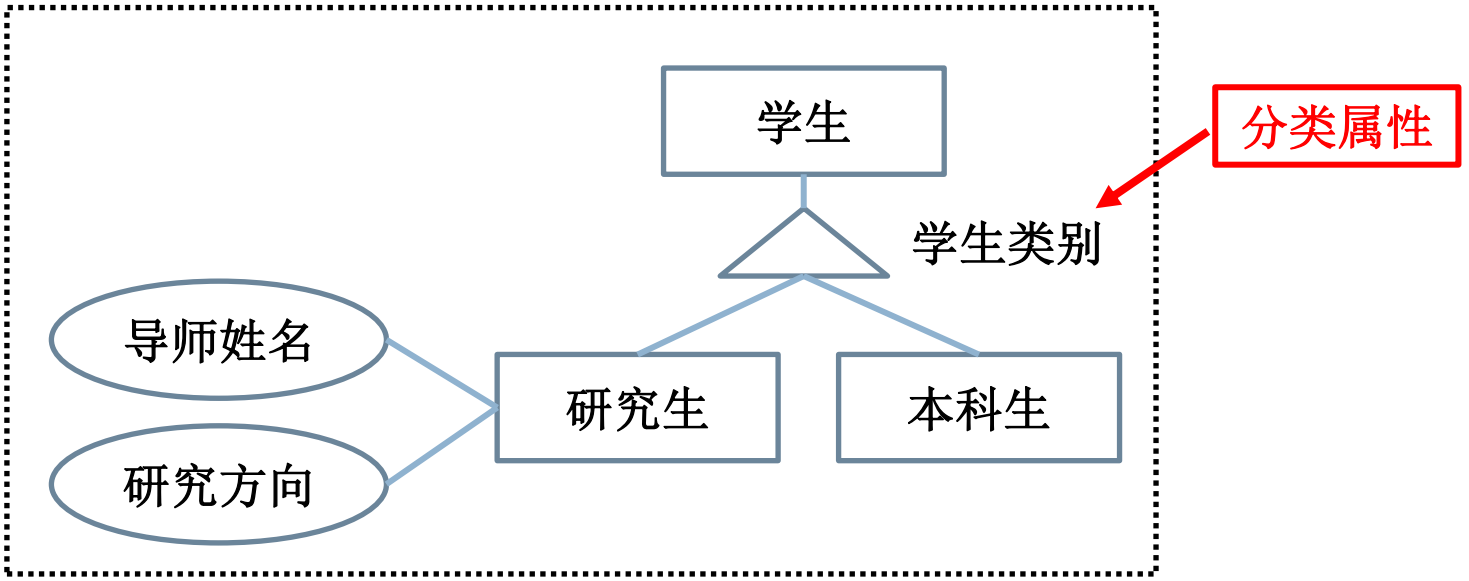


图7.12 学生的2个子类和分类属性

1. ISA 联系

（2）不相交约束与可重叠约束

- 不相交约束：描述父类中的一个实体不能同时属于多个子类中的实体集。即一个父类中的实体最多属于一个子类实体集。
  - 用ISA联系符号 三角形的一个叉号“X”来表示。
- 可重叠约束：父类中的一个实体能同时属于多个子类中的实体集。子类符号中没有叉号表示是可重叠的。

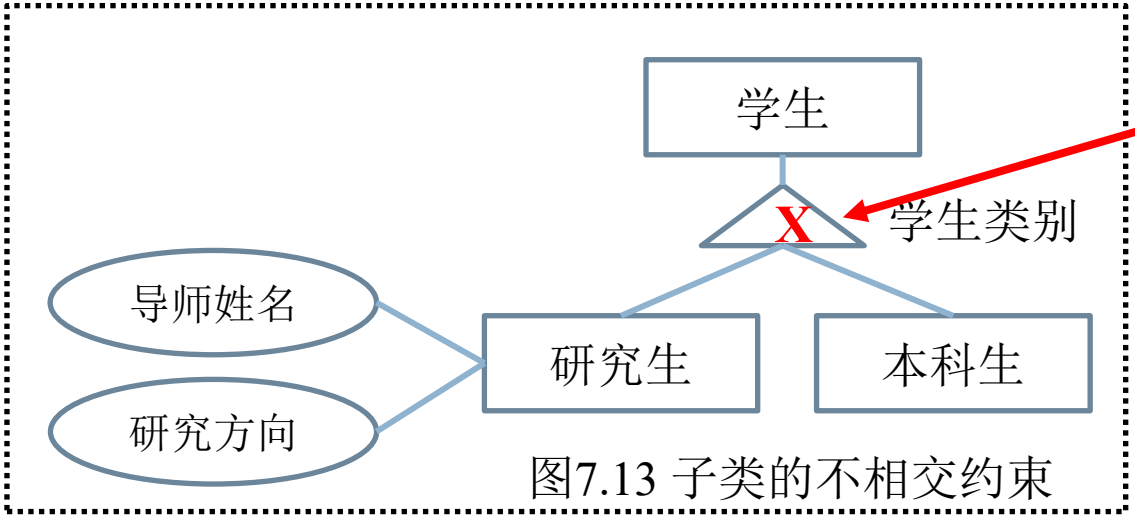


图7.13 子类的不相交约束

1. ISA 联系

（3）完备性约束

- 描述父类中的一个实体是否必须是某一个子类中的实体。
  - 如果是，则叫做完全特化（total specialization）
  - 否则，叫做部分特化（partial specialization）
- 完全特化用父类到子类的双线连接来表示
- 部分特化用父类到子类的单线连接来表示

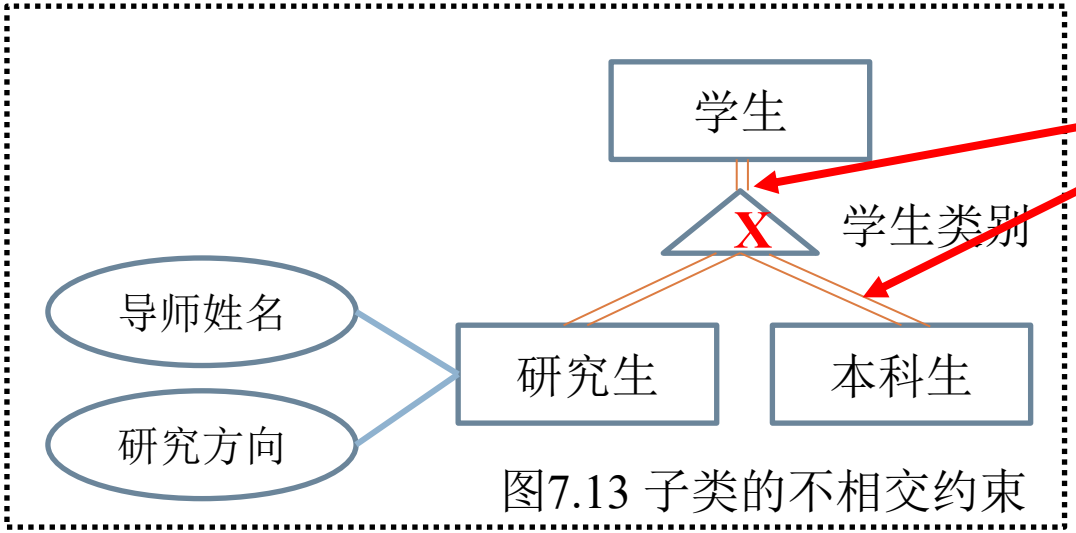


图7.13 子类的不相交约束

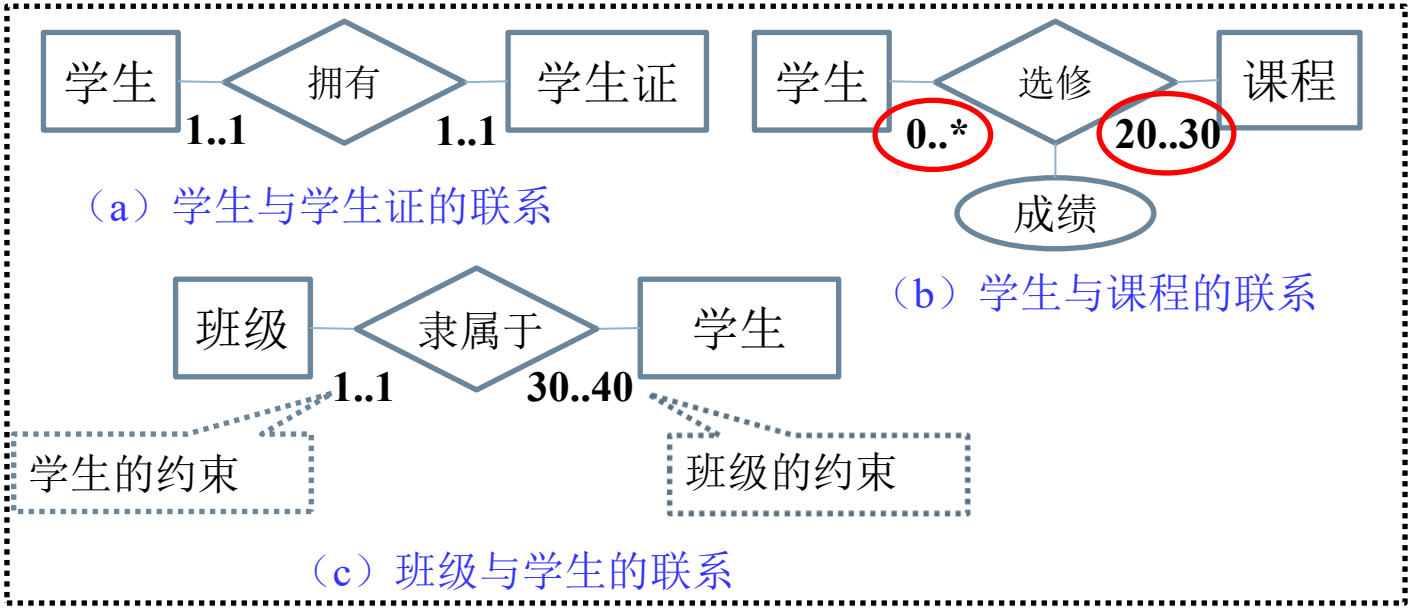
2. 基数约束

- 对E-R模型进行扩展
  - 为了增强和丰富基本E-R模型的表达能力。
- E-R模型的3种扩展：

1. ISA联系
2. 基数约束
3. PART-OF联系

2. 基数约束（续）

- 什么是基数约束？
  - 说明实体型中的任何一个实体可以在联系中出现的最少次数和最多次数
  - 对实体之间一对一、一对多、多对多联系的细化。
  - 约束用一个数对 min..max 表示，0≤min≤max。例如，0..1，1..3，1..*，其中 *代表无穷大



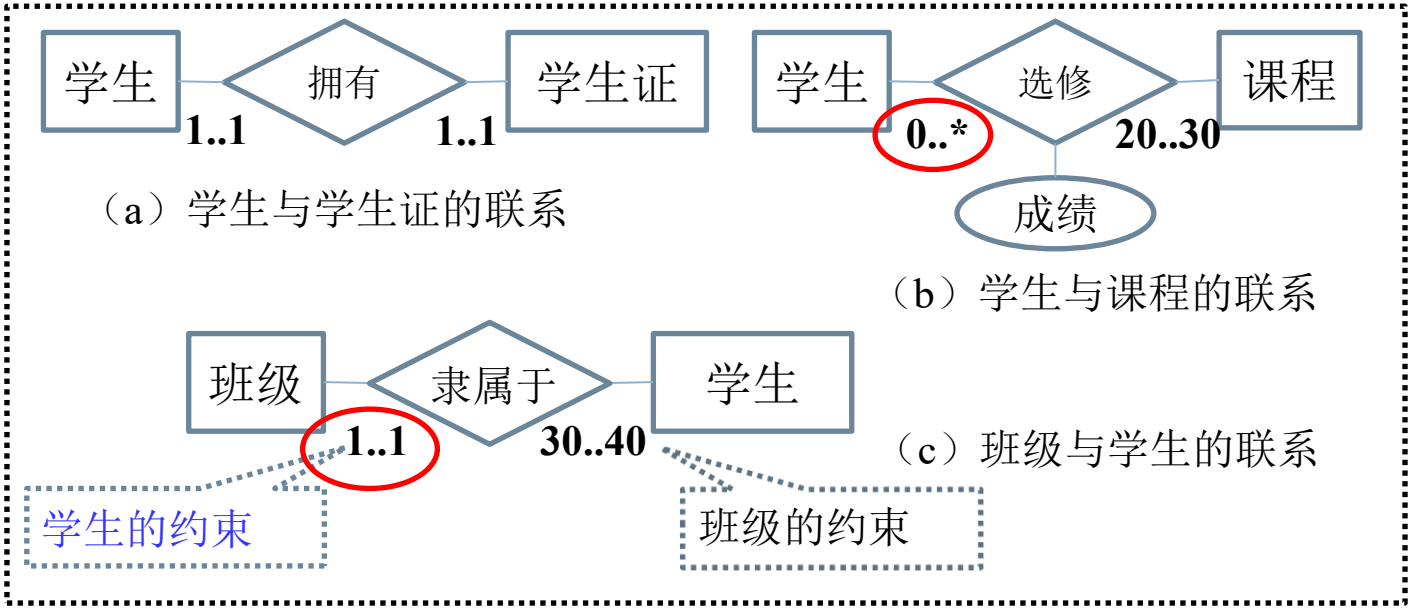
学生实体型的基数约束是20..30，表示每个学生必须选修20～30门课程；

课程的一个基数约束是0..*，即 一门课程可以被很多同学选修也可能还没有同学选修，如新开课。

图7.14 一对一、一对多、多对多的基数约束示例

2. 基数约束（续）

- **基数约束的类别**
  - **min=1的约束叫做强制参与约束，即被施加基数约束的实体型中的每个实体都要参与联系；**
  - **min=0的约束叫做非强制参与约束，被施加基数约束的实体型中的实体可以出现在联系中，也可以不出现在联系中**



允许课程可以没有学生选修

图7.14 一对一、一对多、多对多的基数约束示例

3. PART-OF联系

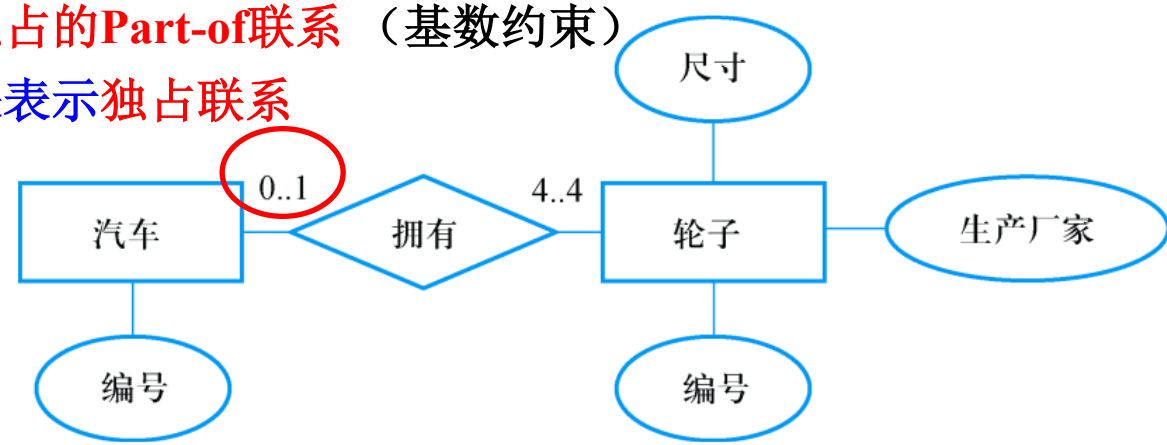
- E-R模型进行
  - 为了增强和丰富基本E-R模型的表达能力。
- 3种扩展：
  - 1. ISA联系
  - 2. 基数约束
  - 3. PART-OF联系

3. PART-OF联系（续）

- 什么是Part-of 联系
  - 描述某个实体型是另外一个实体型的一部分。
- Part-of 联系可以分为两种情况：
  - 非独占的Part-of联系，简称非独占联系

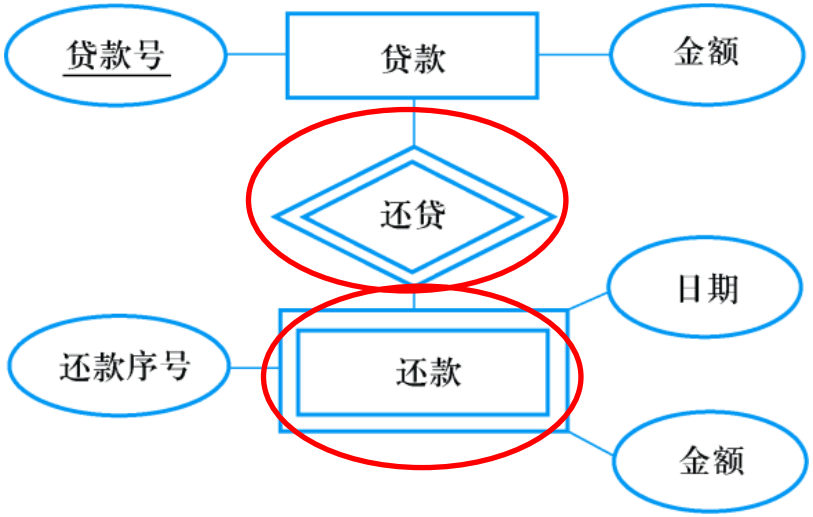
    整体实体如果被破坏，另一部分实体仍然可以独立存在
  - 独占的Part-of联系，简称独占联系

    整体实体如果被破坏，部分实体不能存在
- Part-of 联系如何表示？
  - 用非强制参与联系表示非独占的Part-of联系 （基数约束）
  - 用弱实体类型和识别联系来表示独占联系



3. PART-OF联系（续）

- 什么是弱实体型？
  - 如果一个实体型的存在依赖于其它实体型的存在，则这个实体型叫做弱实体型，否则叫做强实体型。
- 用弱实体类型和识别联系来表示独占联系
  - 双矩形表示弱实体型，用双菱型表示识别联系。



7.3 概念结构设计

7.3.4 UML

- UML称为统一建模语言（Unified Modeling Language），是对象管理 组织（Object Management Group，OMG）的一个标准。
- UML是为软件开发的各个阶段提供模型化和可视化支持的规范语言， 从需求规格描述到系统完成后的测试和维护。
- UML可以用于数据建模，业务建模，对象建模，组件建模等。
- UML提供了多种类型的模型描述图（diagram），借助这些图可以使应用程序更易理解。

用UML的类图建立概念模型

- UML中的类（class）大致对应E-R图中的实体。
- 用UML的类图表示E-R图
  - 实体型：用类表示，矩形框上部记上实体名，下面列出属性名。
  - 实体的码：在类图中在属性后面加“PK”（primary key）来表示码属性。
  - 联系：用类图之间的“关联”来表示。

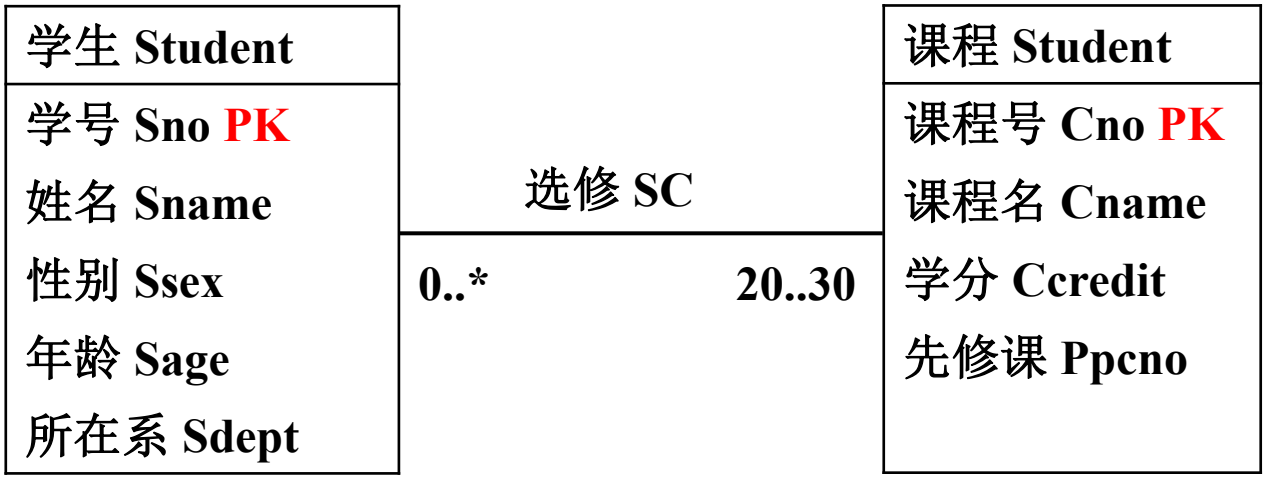


图7.18 用UML的类图表示E-R图示例

SYBASE PowerDesigner

数据库建模UML工具

用UML的类图建立概念模型（续）

- 基数约束
  - UML中关联类之间的基数约束和 E-R图中基数约束的概念类似。
- UML中的子类
  - **面向对象技术**支持超类-子类概念，子类可以继承超类的属性，也可以有自己的属性。
  - 这些概念和E-R图的父类-子类联系（ISA联系）是一致的

SYBASE PowerDesigner

数据库建模UML工具

UML工具

- SYBASE PowerDesigner
  - https://sybase-powerdesigner.informer.com/

Sybase PowerDesigner 16.6

Visualize the impact of changes before they are applied

User rating ★★★★☆ 4 (153 votes)

Your vote ☆☆☆☆☆

Latest version: 16.6 (See all)

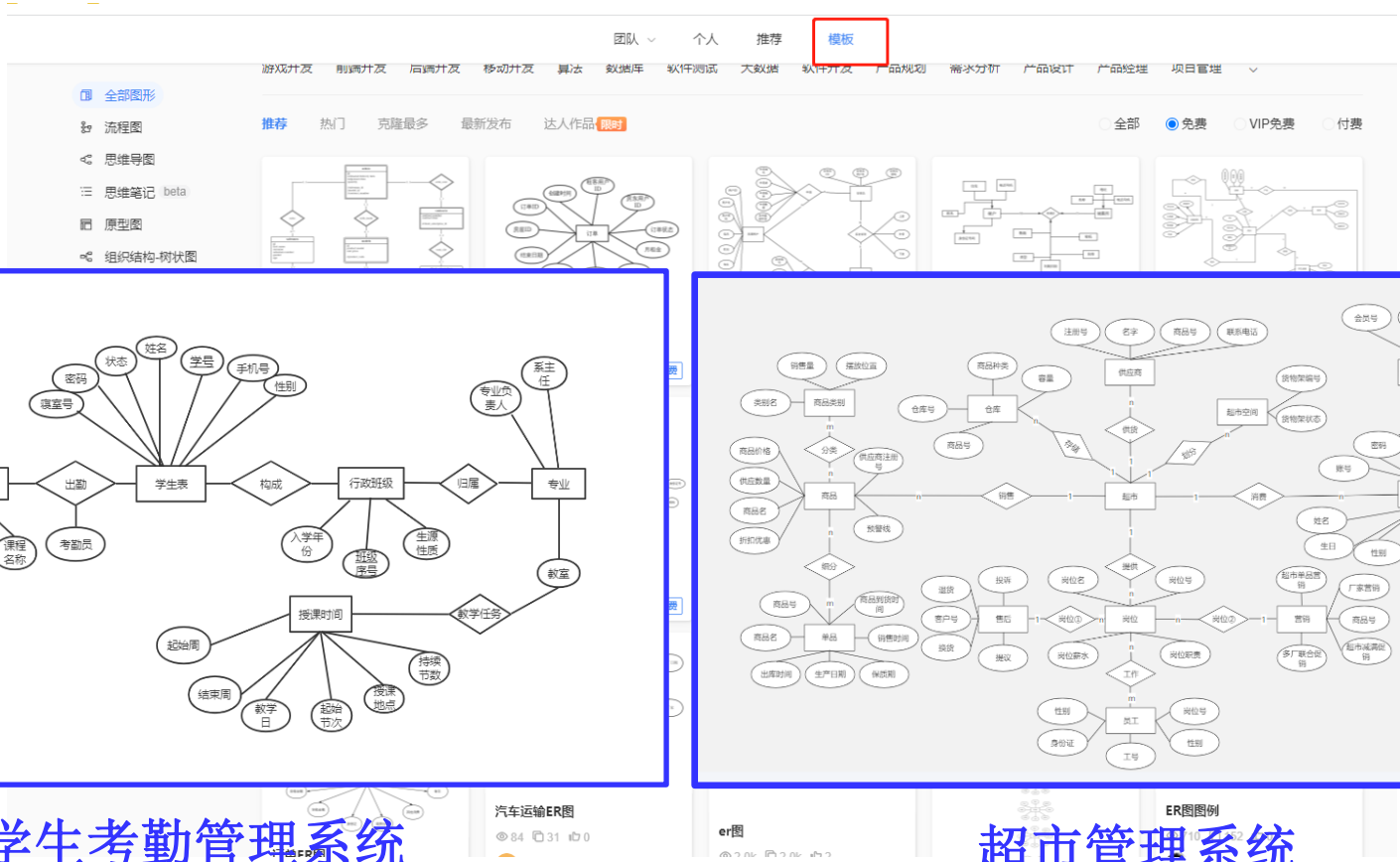
Developed by: SAP

Power Designer has integrated support for:

- Business Process Modeling supporting Business Process Modeling Notation
- Code generation for Java, C#, VB, .NET, EJB3, JSF, WinForm, and others.
- Data Warehouse Modeling
- Eclipse plug-in
- Object modeling (UML 2.0 diagrams)
- Report generation
- XML Modeling supporting XML Schema and DTD standards
- Visual Studio 2005 add-in

E-R图设计软件

- https://processon.com/
  - 含有大量ER图模板



学生考勤管理系统

超市管理系统