



哈爾濱工業大學

Harbin Institute of Technology

数据库系统

万晓珑 博士
大数据计算研究中心

wxl@hit.edu.cn



哈爾濱工業大學

Harbin Institute of Technology

数据库系统

第七章 数据库设计

第七章 数据库设计

7.1 数据库设计概述

7.2 需求分析

7.3 概念结构设计

7.4 逻辑结构设计

7.5 物理结构设计

7.6 数据库的实施和维护

7.7 小结

7.1 数据库设计概述

❖ 数据库设计（广义定义和狭义定义）

- 广义范围：数据库及其应用系统（使用数据库的各类信息系统）的设计
- 狭义范围：数据库本身的设计

❖ 本课程的重点在于数据库自身的设计

数据库设计概述（续）

❖ 数据库设计

- 数据库设计指对于一个给定的应用环境，构造优化的数据库**逻辑模式**和**物理结构**，据此建立数据库及其应用系统，使之能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（续）

❖ 数据库设计

- 数据库设计的**目标**是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（特点1）

- 三分技术，七分管理，十二分基础数据
- 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
 - 数据的收集、整理、组织和不断更新

数据库设计的特点（续）

2. 结构设计和行为设计相结合（特点2）

- 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，着重于过程处理的特性，只要有可能就尽量推迟数据库结构设计的决策
- 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

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的技术，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

数据库设计方法（续）

❖ 手工试凑法

- 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又不同程度地发现各种问题，增加了维护代价

数据库设计方法（续）

❖ 规范设计法

- 基本思想: 过程迭代和逐步求精 (迭代是重复反馈过程的活动, 每一次对过程的重复称为一次迭代, 而每一次迭代得到的结果会作为下一次迭代的初始值)
- 典型方法
 - 新奥尔良 (**New Orleans**) 方法
 - 基于**E-R**模型的数据库设计方法
 - **3NF** (第三范式) 的设计方法
 - 面向对象的数据库设计方法
 - 统一建模语言 (**UML**) 方法

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数据库管理系统密切相关

数据库设计的基本步骤（续）

❖ 参加数据库设计的人员

■ 系统分析人员和数据库设计人员

- 自始至终参与数据库设计，其水平决定了数据库系统的质量

■ 数据库管理员和用户代表

- 主要参加需求分析与数据库的运行和维护

■ 应用开发人员

- 包括程序员和操作员
- 在实施阶段参与进来，分别负责编制程序和准备软硬件环境

数据库设计的基本步骤（续）

1. 需求分析阶段

- 是否做得充分与准确，决定构建数据库的速度和质量

2. 概念结构设计阶段

- 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数据库管理系统的**概念模型**

3. 逻辑结构设计阶段

- 将概念结构转换为某个数据库管理系统所支持的**数据模型（例如关系模型）**，并对其进行优化

数据库设计的基本步骤（续）

4. 物理结构设计阶段

- 为逻辑数据结构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

5. 数据库实施阶段

- 根据逻辑设计和物理设计的结果构建数据库
- 编写与调试应用程序
- 组织数据入库并进行试运行

6. 数据库运行和维护阶段

- 经过试运行后即可投入正式运行
- 在运行过程中必须不断对其进行评估、调整与修改

数据库设计的基本步骤（续）

- ❖ 设计一个完善的数据库应用系统，往往是上述6个阶段的不断反复
- ❖ 这个设计步骤既是数据库设计的过程，也包括了数据库应用系统的设计过程
- ❖ 把数据库的设计和**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

数据库设计的基本步骤（续）

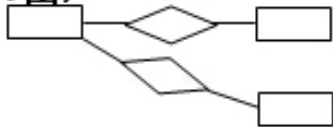
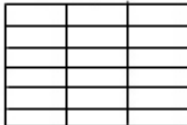
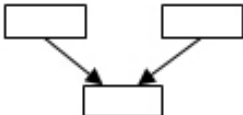
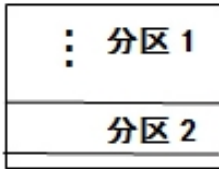

设计阶段	设计描述
需求分析	数字字典、全系统中数据项、数据结构、数据流、数据存储的描述
概念结构设计	概念模型（E-R 图）  数据字典
逻辑结构设计	某种数据模型 <div> <div>关系</div>  </div> <div> <div>非关系</div>  </div>
物理结构设计	存储安排 存取方法选择 存取路径建立 
数据库实施	创建数据库模式 装入数据 数据库试运行 
数据库运行和维护	性能监测、转储/恢复、数据库重组和重构

图7.3 数据库设计各个阶段的数据设计描述

7.1 数据库设计概述

7.1.1 数据库设计的特点

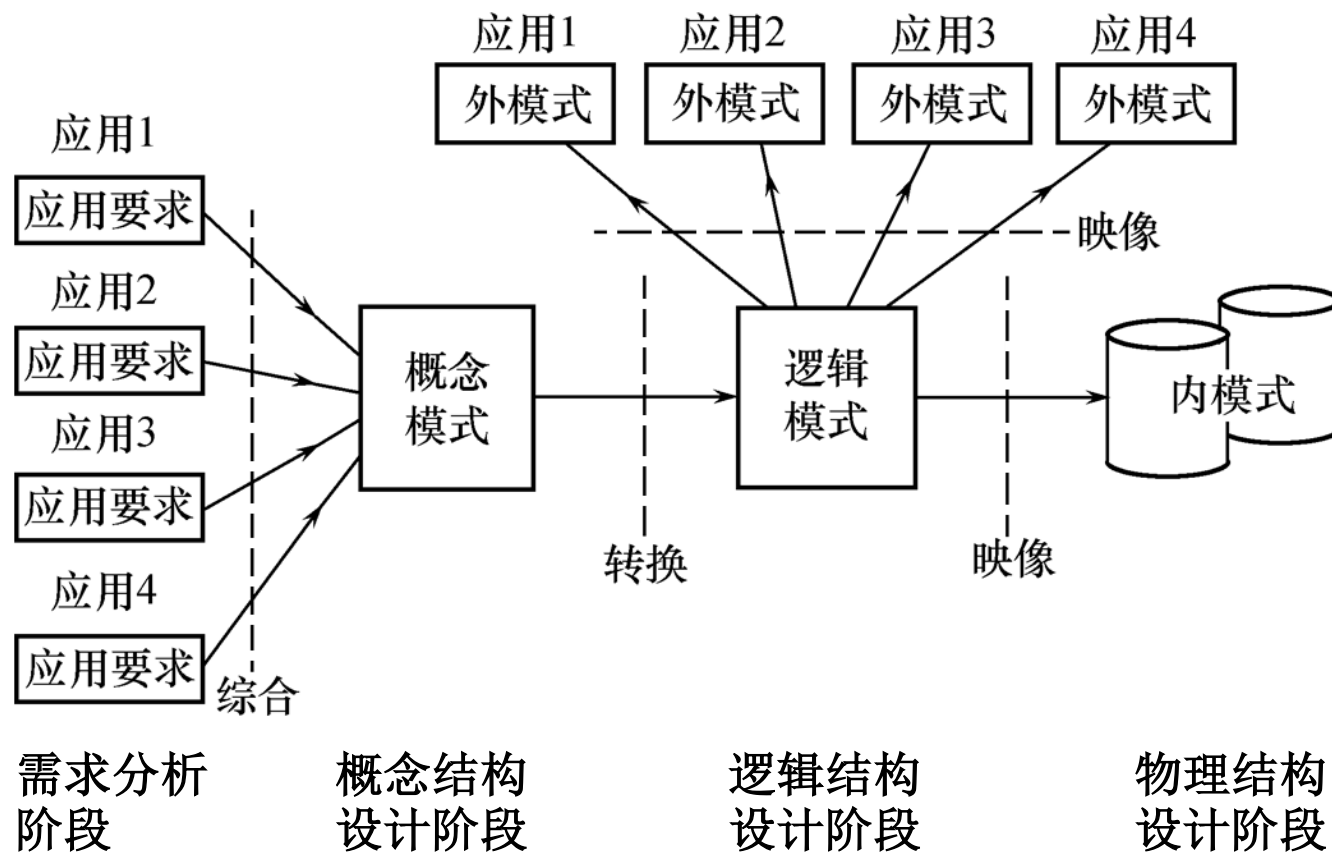
7.1.2 数据库设计方法

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

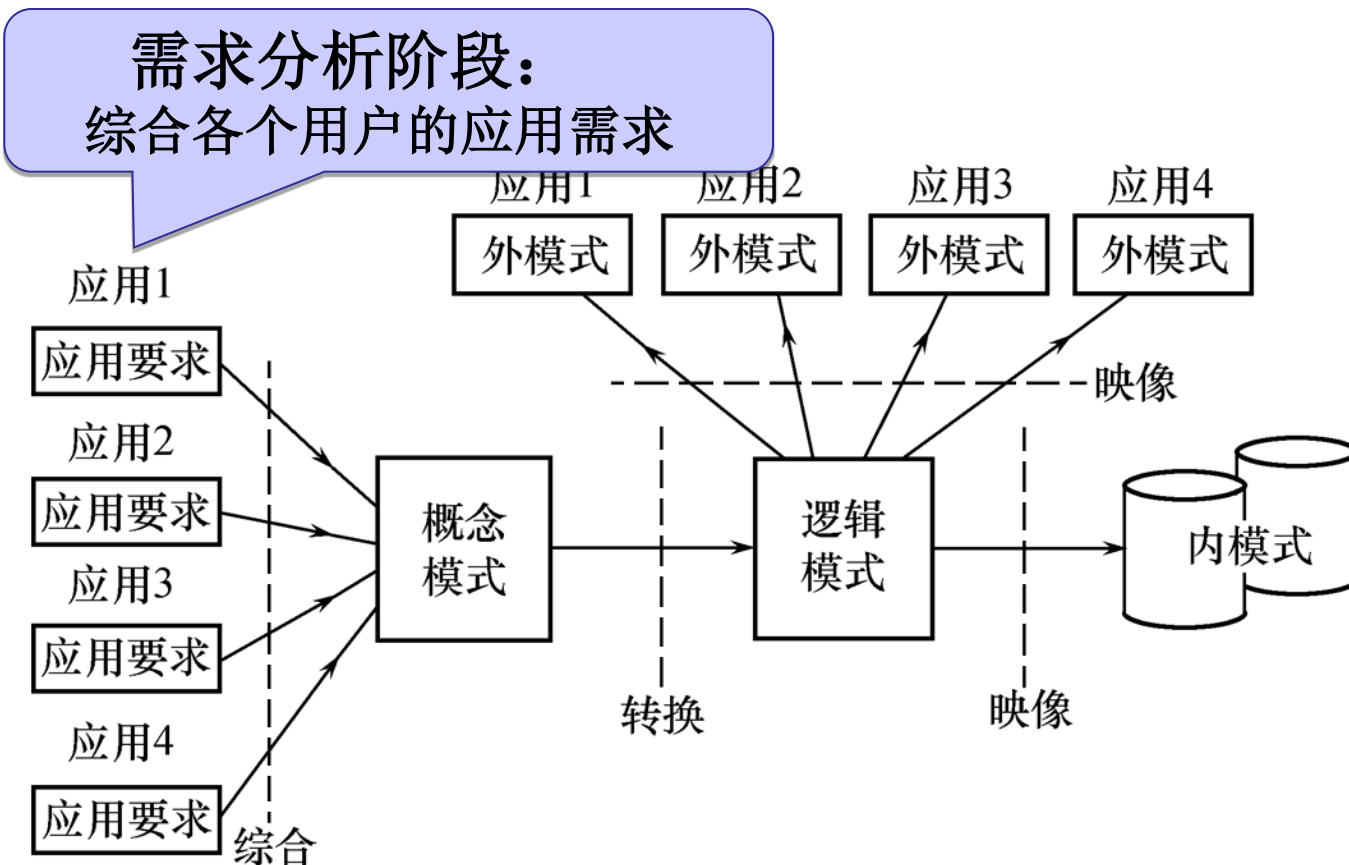
7.1.4 数据库设计过程中的各级模式

❖ 数据库设计不同阶段形成的数据库各级模式



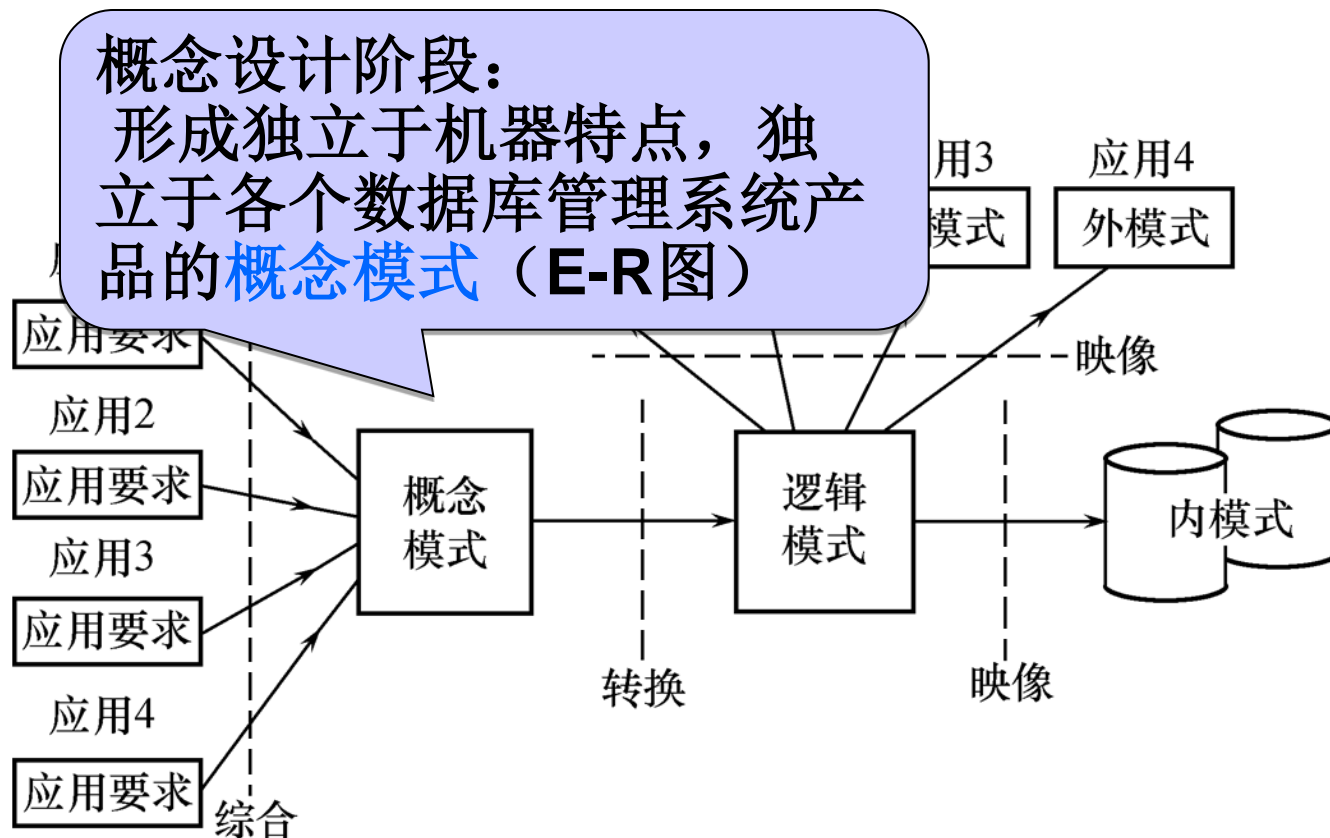
数据库设计过程中的各级模式（续）

❖ 数据库设计不同阶段形成的数据库各级模式



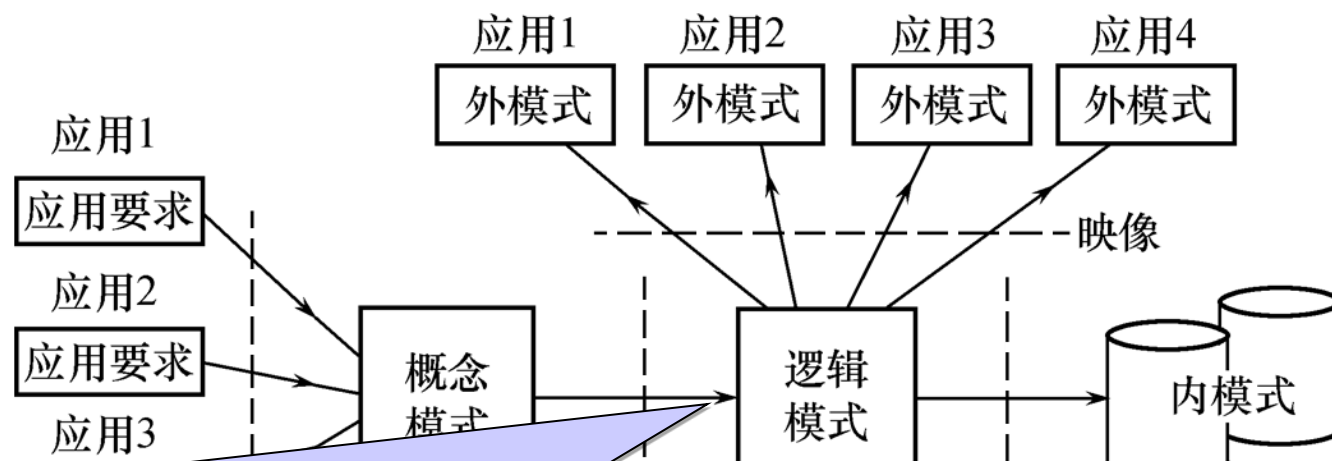
数据库设计过程中的各级模式（续）

❖ 数据库设计不同阶段形成的数据库各级模式



数据库设计过程中的各级模式（续）

❖ 数据库设计不同阶段形成的数据库各级模式



逻辑设计阶段：

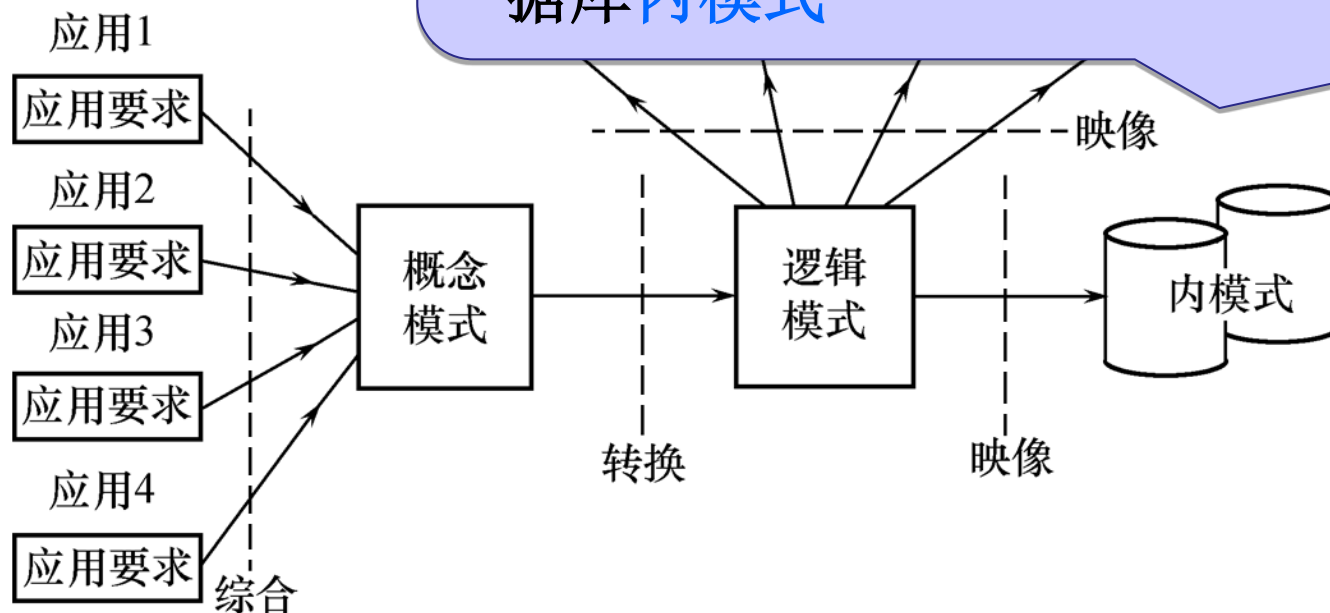
1. 首先将**E-R图**转换成具体的数据库产品支持的数据模型，如关系模型，形成数据库**逻辑模式**
2. 然后根据用户处理的要求、安全性的考虑，在基本表的基础上再建立必要的视图（**View**），形成数据的**外模式**

数据库设计过程中的各级模式（续）

❖ 数据库设计不同阶段形成的数据库各级模式

物理设计阶段：

根据数据库管理系统特点和处理的需要，进行物理存储安排，建立索引，形成数据库内模式



第七章 数据库设计

7.1 数据库设计概述

7.2 需求分析

7.3 概念结构设计

7.4 逻辑结构设计

7.5 物理结构设计

7.6 数据库的实施和维护

7.7 小结

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（续）

❖ 需求分析就是分析用户的要求

- 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

7.2.1 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）工作概况
- ❖ 明确用户的各种需求
- ❖ 在此基础上确定**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

需求分析的任务（续）

❖ 调查的重点是**数据**和**处理**，获得用户对数据库的要求

（1）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

（2）处理要求

- 用户要完成的处理功能
- 对处理性能的要求

（3）安全性与完整性要求

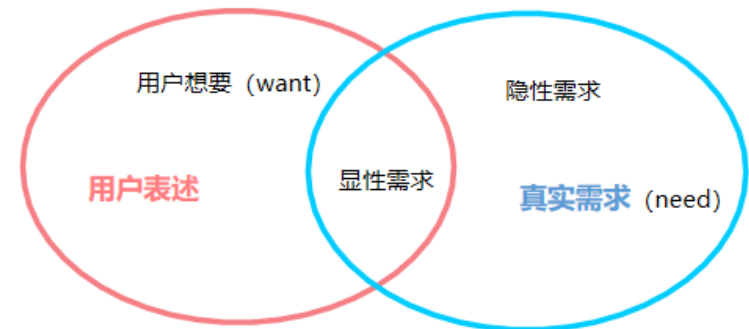
需求分析的任务（续）

❖ 确定用户最终需求的难点

- 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

❖ 解决方法

- 设计人员必须不断深入地与用户进行交流，才能逐步确定用户的实际需求



发明汽车以前，问人们需要什么样的交通工具，人们会回答：“我需要一匹更快的马”，而不是“我需要一辆汽车”

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

7.2.2 需求分析的方法

- ❖ 调查清楚用户的实际需求并进行初步分析
- ❖ 与用户达成共识
- ❖ 分析与表达这些需求

调查用户需求的步骤

- (1) 调查组织机构情况
- (2) 调查各部门的业务活动情况
- (3) 协助用户明确对新系统的各种要求，包括信息要求、处理要求、完全性与完整性要求
- (4) 确定新系统的边界

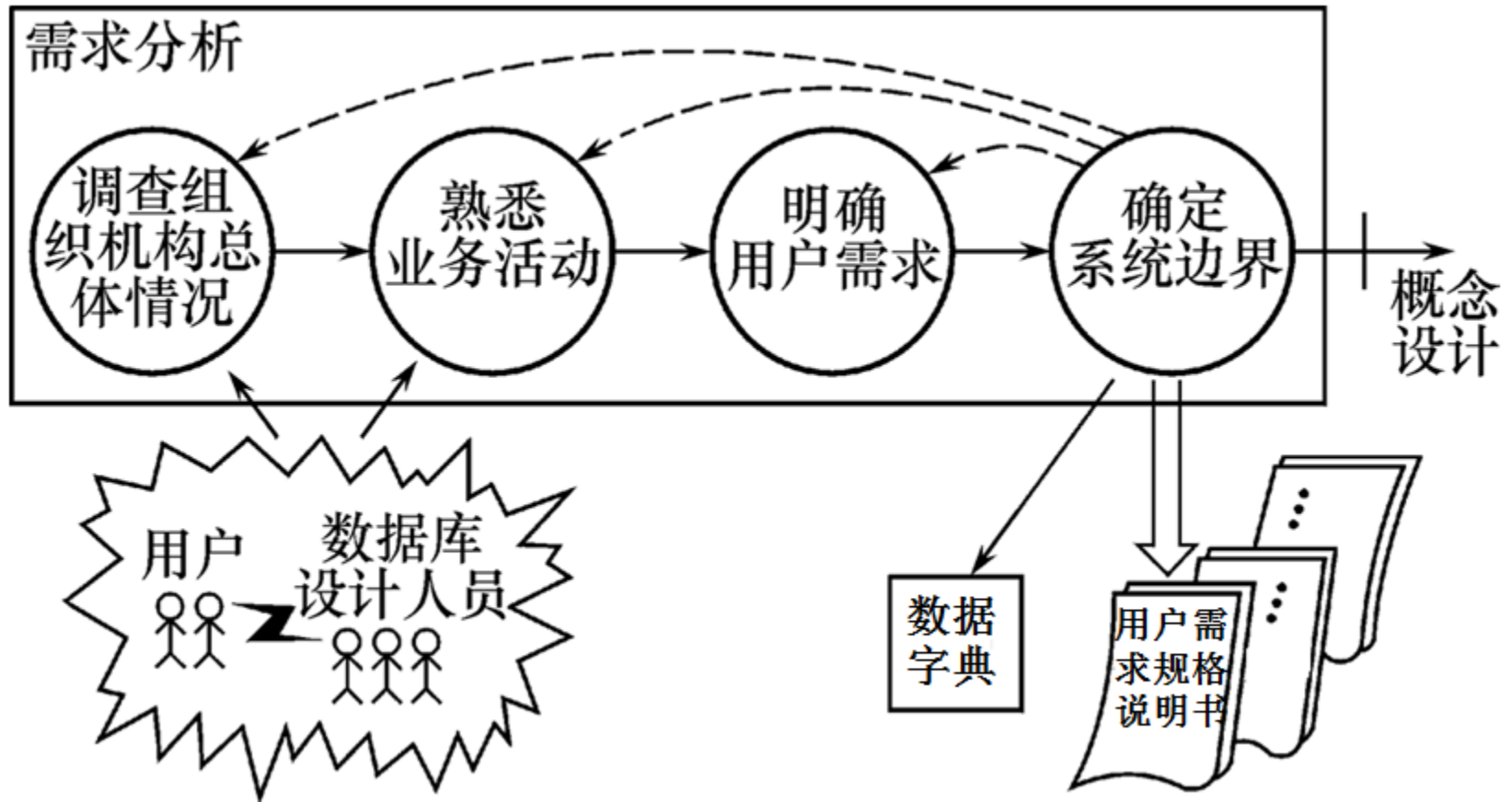
常用调查方法

- (1) **跟班作业**: 通过亲身参加业务工作了解业务活动的情况
- (2) **开调查会**: 通过与用户座谈来了解业务活动情况及用户需求
- (3) **请专人介绍**
- (4) **询问**: 对某些调查中的问题, 可以找专人询问
- (5) **设计调查表请用户填写**: 如果调查表设计合理, 则很有效
- (6) **查阅记录**: 查阅与原系统有关的数据记录

进一步分析和表达用户需求

- ❖ 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

需求分析过程



需求分析过程

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

- ❖ 数据字典在需求分析阶段建立，是进行详细的数据收集和数据分析所获得的主要结果
- ❖ 注意和关系数据库管理系统中数据字典的区别和联系

数据字典（续）

❖ 数据字典的内容

- **数据项**：不可再分的数据单位（看做属性）
- **数据结构**：由若干个数据项组成（看作二维表）
- **数据流**：数据结构在系统内传输的路径
- **数据存储**：数据结构停留或保存的地方
- **处理过程**：需要描述处理过程的说明性信息

1. 数据项

- ❖ 数据项是不可再分的数据单位
- ❖ 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}
- “取值范围”、“与其他数据项的逻辑关系”定义数据的完整性约束条件，是设计数据检验功能的依据
- 可以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

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。
- ❖ 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。
- ❖ 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
- 平均流量：在单位时间（每天、每周等）的传输次数
- 高峰期流量：在高峰时期的数据流量

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。
- ❖ 可以是手工文档，也可以是计算机文档
- ❖ 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
 - **存取方式**: 批处理/联机处理, 检索/更新, 顺序/随机检索
 - 输入的数据流: 数据来源
 - 输出的数据流: 数据去向

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

需求分析小结

- ❖ 把需求收集和分析作为数据库设计的第一阶段是十分重要的。
- ❖ 第一阶段收集的基础数据（用数据字典来表达）是下一步进行概念设计的基础。
- ❖ 强调两点
 - （1）设计人员应充分考虑到可能的扩充和改变，使设计易于更改，系统易于扩充
 - （2）必须强调用户的参与

第七章 数据库设计

7.1 数据库设计概述

7.2 需求分析

7.3 概念结构设计

7.4 逻辑结构设计

7.5 物理结构设计

7.6 数据库的实施和维护

7.7 小结

7.3 概念结构设计

7.3.1 概念模型

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

7.3.1 概念模型

- ❖ 将需求分析得到的用户需求抽象为概念模型的过程就是概念结构设计
- ❖ 概念模型的特点
 - 真实、充分地反映现实世界，现实世界的一个真实模型
 - 易于理解，可以用它和不熟悉计算机的用户交换意见。
 - 易于更改，当应用环境和应用要求改变时，容易对概念模型修改和扩充。
 - 易于向关系、网状、层次等各种数据模型转换

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

7.3.2 E-R模型

❖ E-R模型：用E-R图来描述现实世界的概念模型

陈品山(Peter Pin-Shan Chen)，路易斯安纳州立大学教授，1976年3月在ACM Transactions on Database Systems上发表了The Entity-Relationship Model--Toward a Unified View of Data的论文

❖ 实体（**Entity**）表示一个离散对象。实体可以被认为是名词，如雇员、学生、教师、课程等

7.3.2 E-R模型

1. 实体之间的联系

(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 ($m:n$)

E-R模型（续）

①一对一联系（1 : 1）

- 例如，学校里一个班级只有一个班长，而一个班长只在一个班中任职，则班级与班长之间具有一对一联系。
- 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**。

E-R模型（续）

②一对多联系（1： n ）

- 例如，一个班级中有若干名学生，而每个学生只在一个班级中学习，则班级与学生之间具有一对多联系。
- 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（ $n \geq 0$ ）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 。

E-R模型（续）

③多对多联系（ $m:n$ ）

- 例如，一门课程同时有若干个学生选修，而一个学生可以同时选修多门课程，则课程与学生之间具有多对多联系。
- 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（ $n \geq 0$ ）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（ $m \geq 0$ ）与之联系，则称实体集**A**与实体集**B**具有多对多联系，记为 **$m:n$** 。

E-R模型（续）

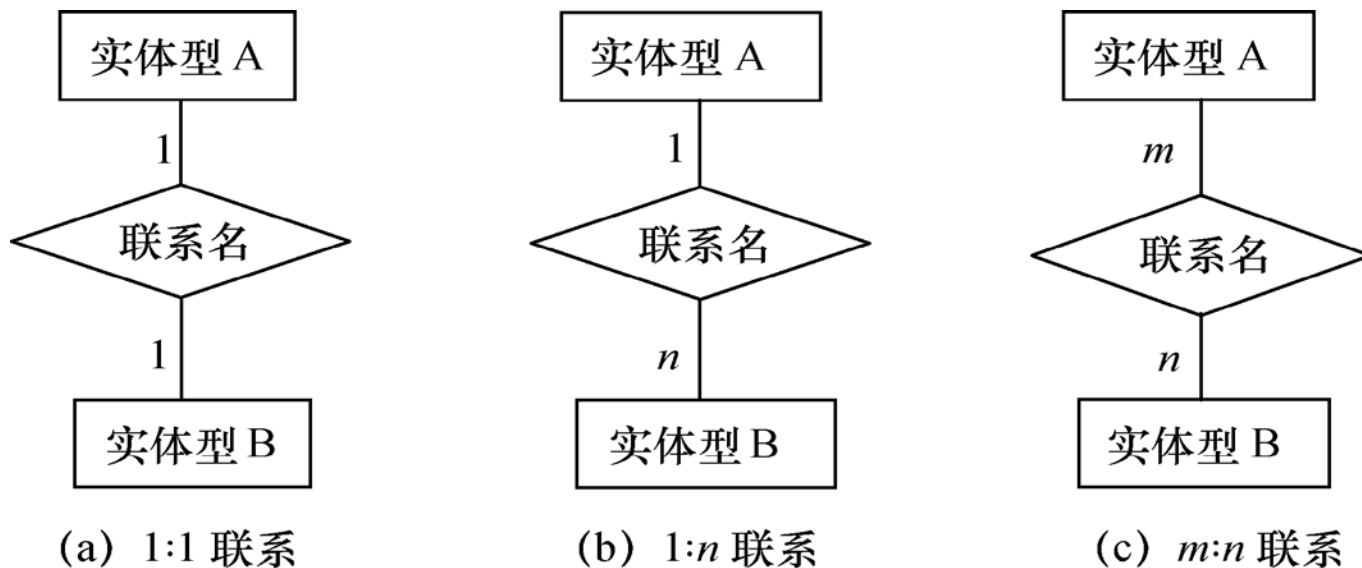
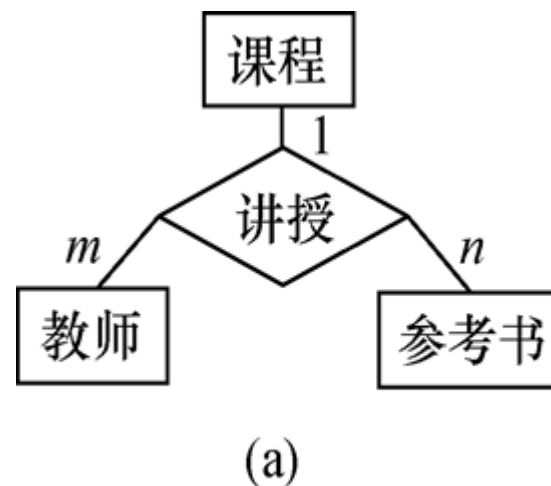


图7.6 两个实体型之间的三类联系

E-R模型（续）

（2）两个以上的实体型之间的联系

- 一般地，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。
- 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，而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。

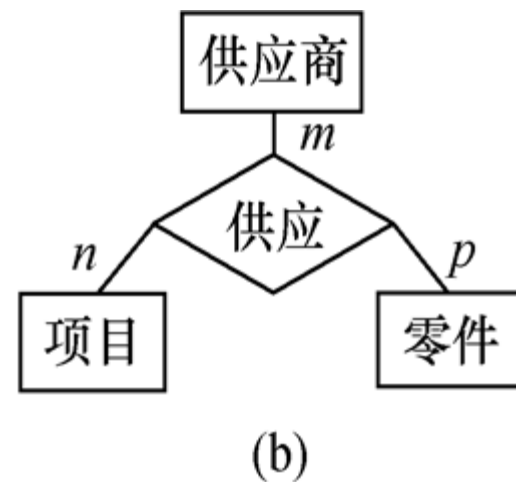


三个实体型之间的联系示例，
但教师和参考书之间没有联系

E-R模型（续）

（2）两个以上的实体型之间的联系

- 一般地，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。
- 供应商、项目、零件3个实体型，一个供应商可以给多个项目提供多种零件，每个项目可以使用多个供应商提供的零件，每种零件可由不同供应商提供，这三个实体间是多对多联系。

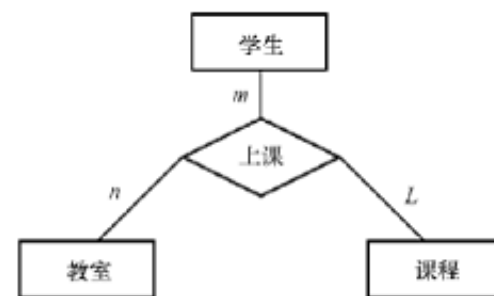


三个实体型之间的联系示例，
但项目和零件之间没有联系

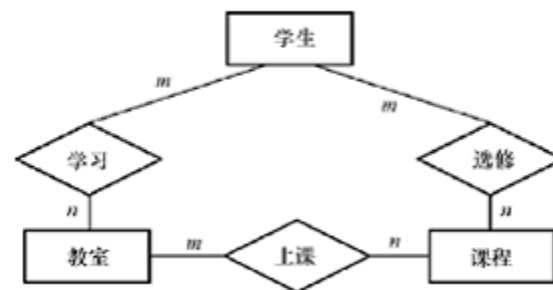
E-R模型（续）

（2）两个以上的实体型之间的联系

- 在两个以上实体集之间，当一个实体集与其它实体集之间均存在多对多联系，**而其它实体集之间没有联系**，这种联系才称之为多个实体集之间的多对多联系。
- 两个以上实体型之间的多对多联系和两个以上实体型两两之间的多对多联系是不等价。



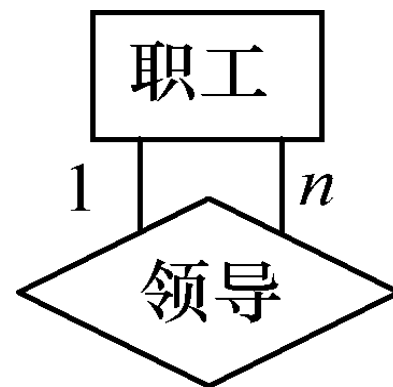
教室和课程没有关系



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。



单个实体型内的一对多联系示例

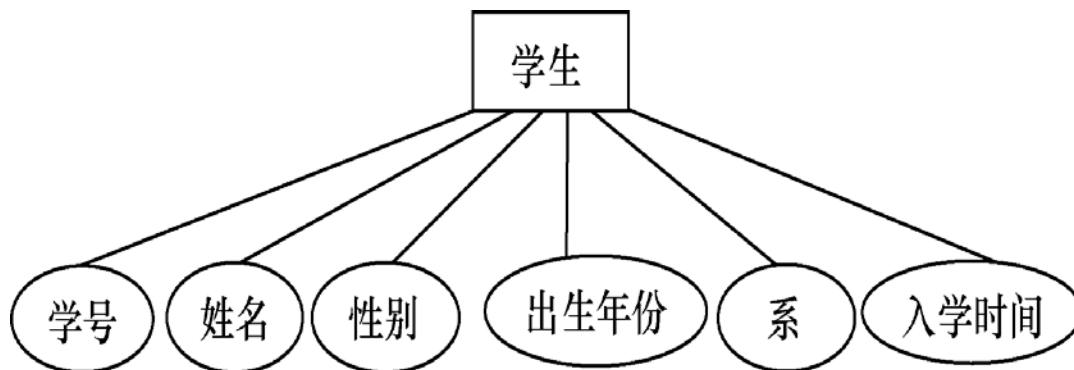
E-R模型（续）

- 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，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，称为N元联系

E-R模型（续）

2. E-R图：提供表示实体型、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。
 - 学生实体有学号、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



E-R模型（续）

- 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

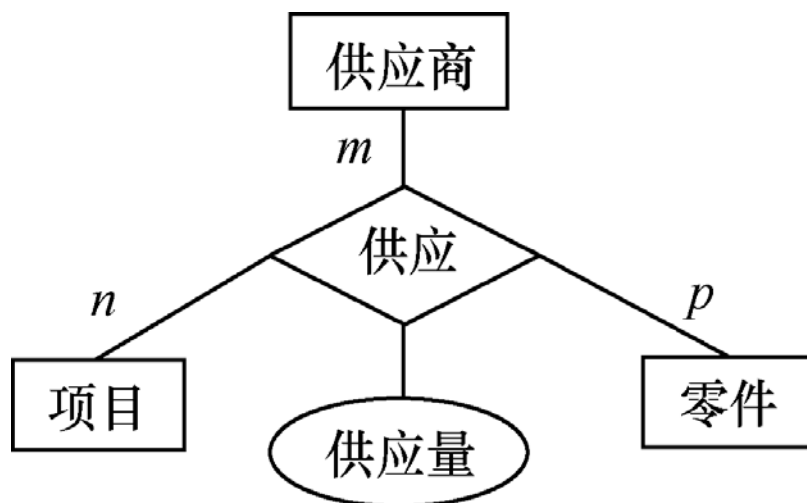


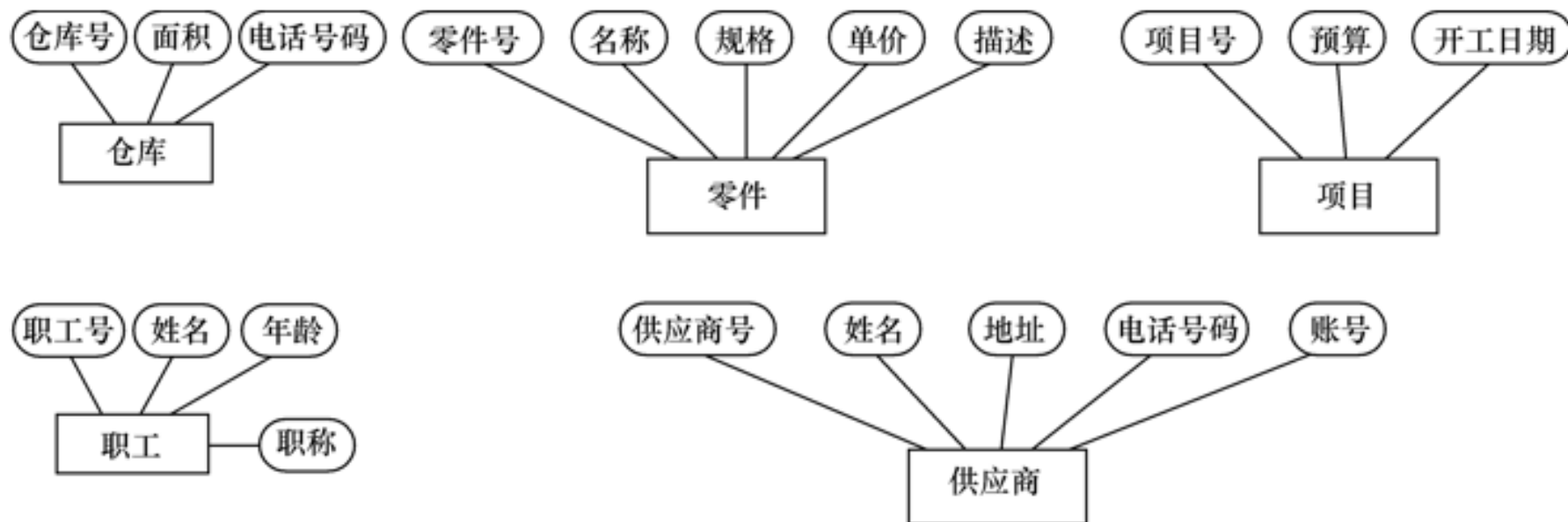
图7.10 联系的属性

E-R模型（续）

❖ 3. 一个实例

- 某个工厂物资管理的概念模型，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，**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。

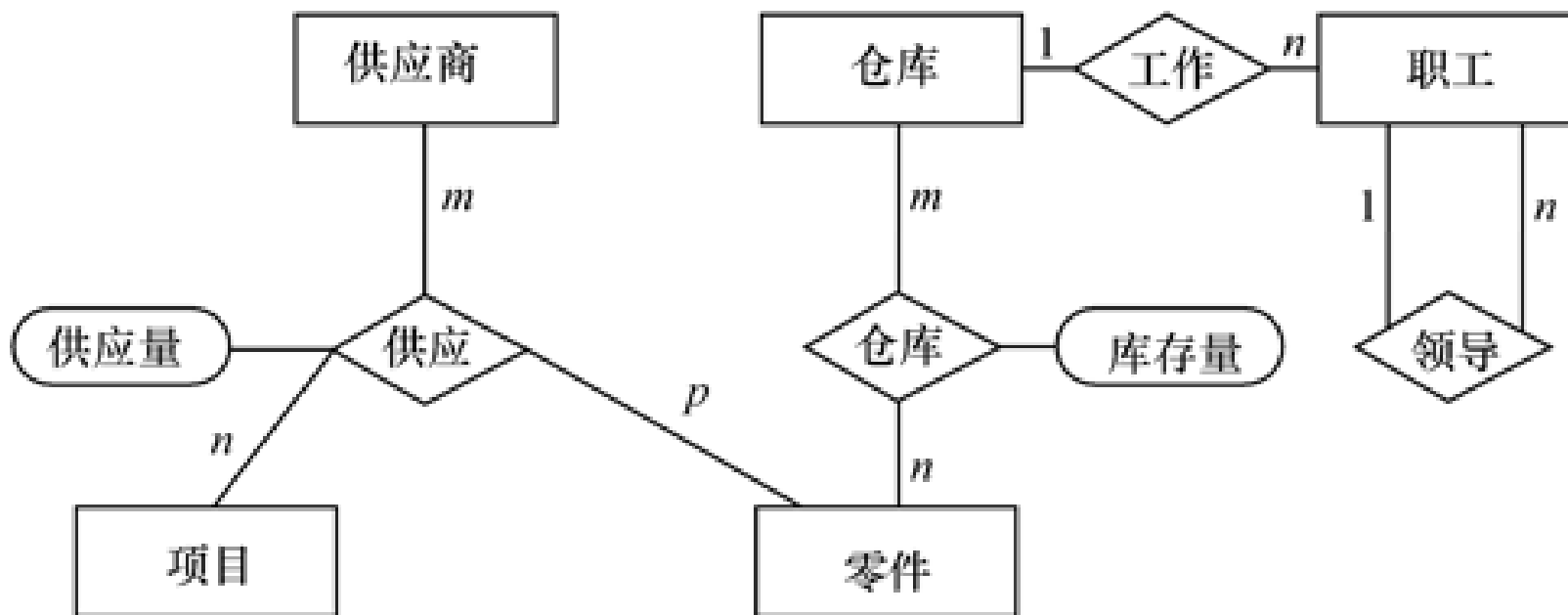
E-R模型（续）

- 这些实体之间的联系如下（续）：

（3） 职工之间具有领导与被领导关系，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。

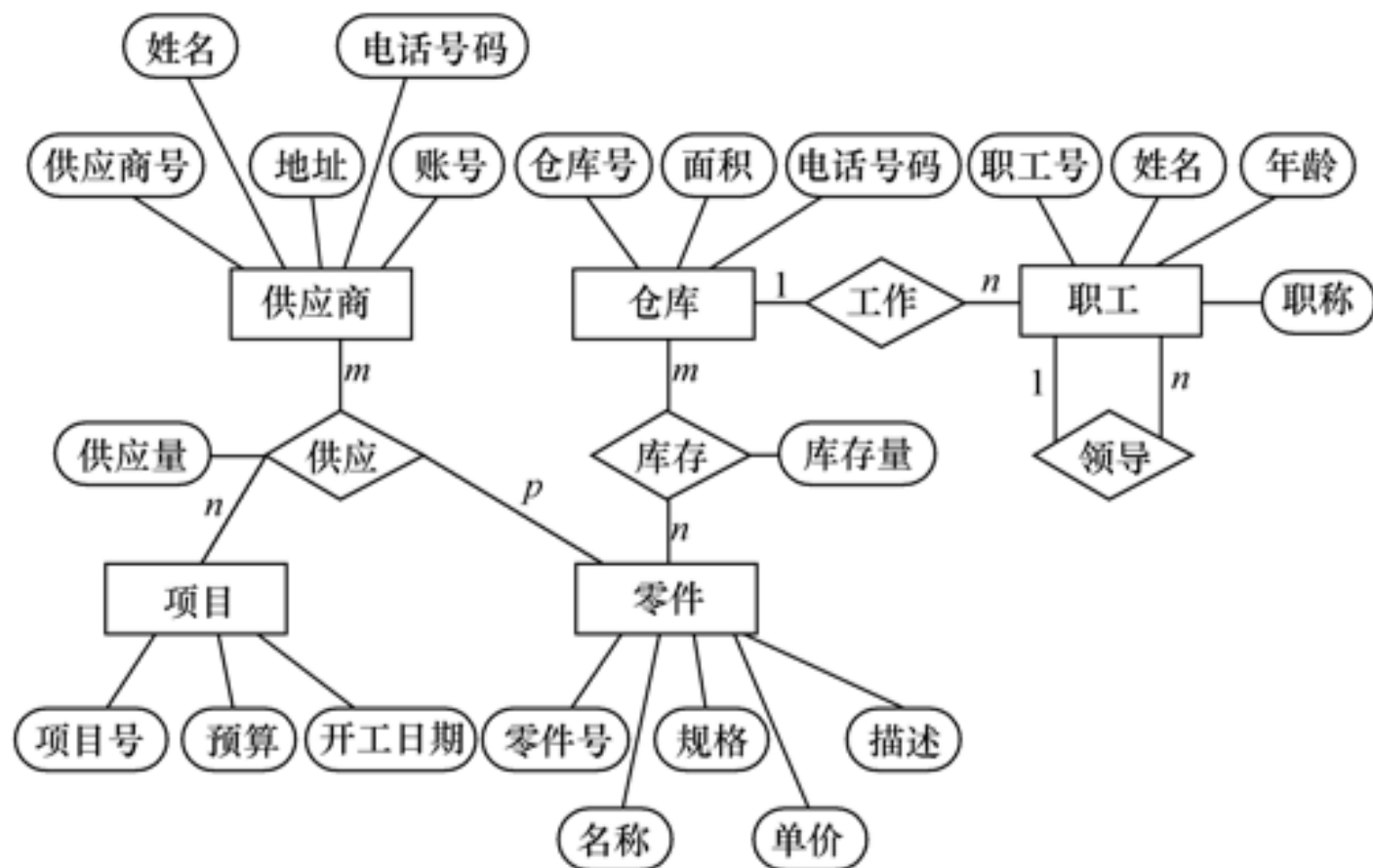
（4） 供应商、项目和零件三者之间具有**多对多**的联系，一个供应商可以供给若干项目多种零件，每个项目可以使用不同供应商供应的零件，每种零件可由不同供应商供给。

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

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

7.3.5 概念结构设计

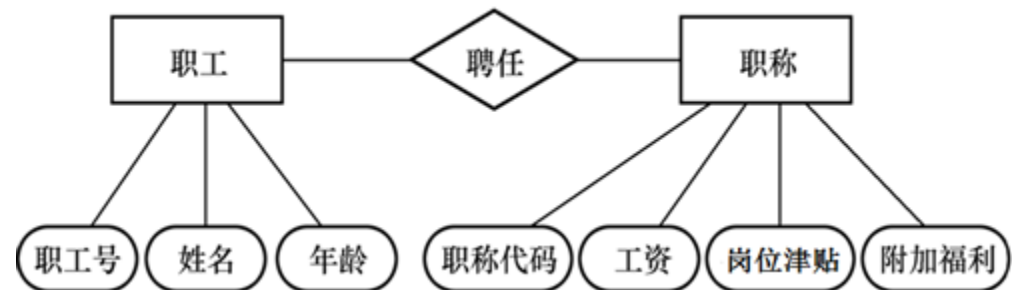
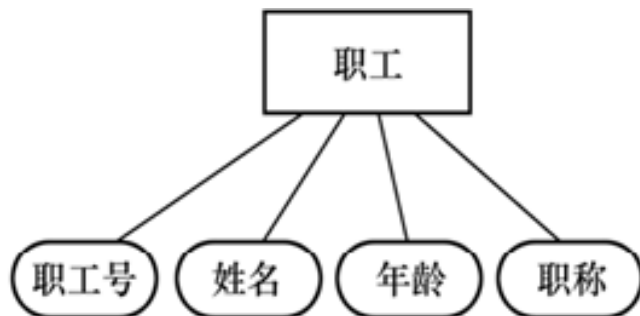
1. 实体与属性的划分原则

- 为了简化E-R图的处置，现实世界的事物**能作为属性对待的，尽量作为属性对待。**
- 两条准则：
 - (1) 作为属性，不能再具有需要描述的性质。属性必须是不可分的数据项，不能包含其他属性。
 - (2) 属性不能与其他实体具有联系，即E-R图中所表示的联系是实体之间的联系。

概念结构设计（续）

[例1] 职工是实体，职工号、姓名、年龄是职工的属性。

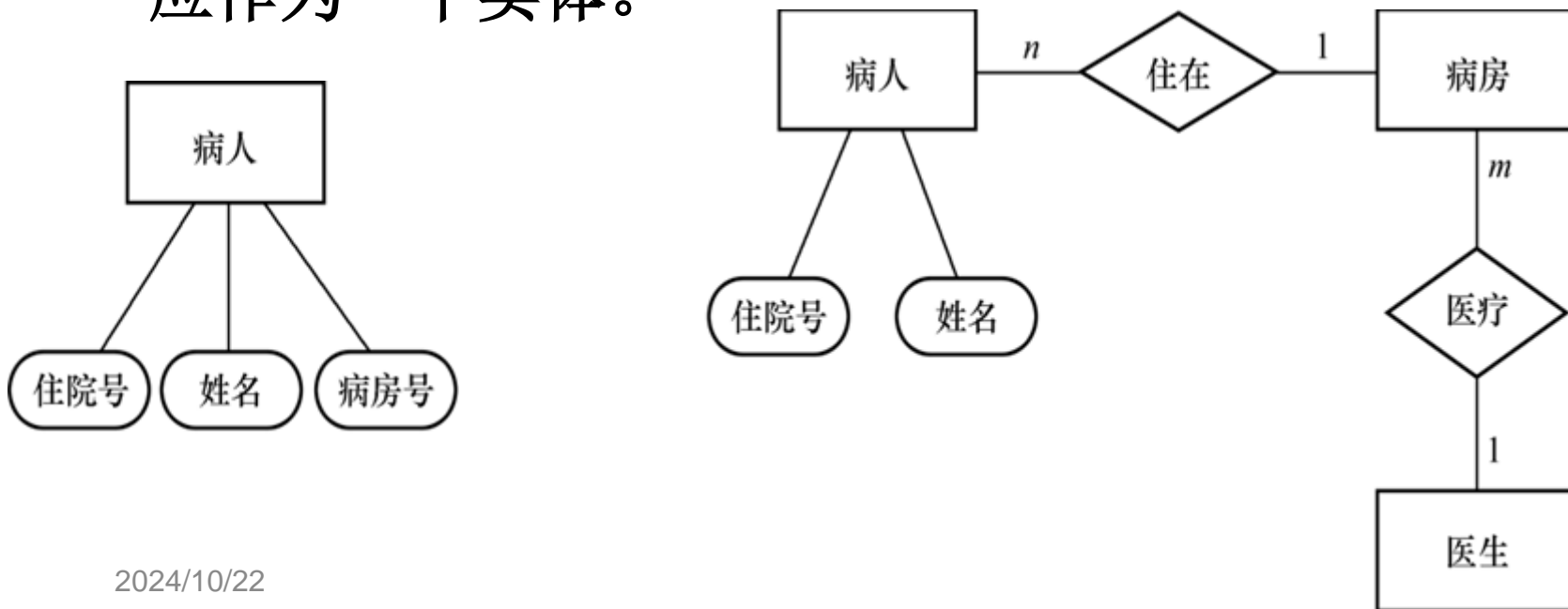
- 职称如果没有与工资、福利挂钩，根据准则（1）可以作为职工实体的属性
- 如果不同的职称有不同的工资、住房标准和不同的附加福利，则职称作为一个实体更恰当



概念结构设计（续）

[例2] 在医院中，一个病人只能住在一个病房，病房号可以作为病人实体的一个属性；

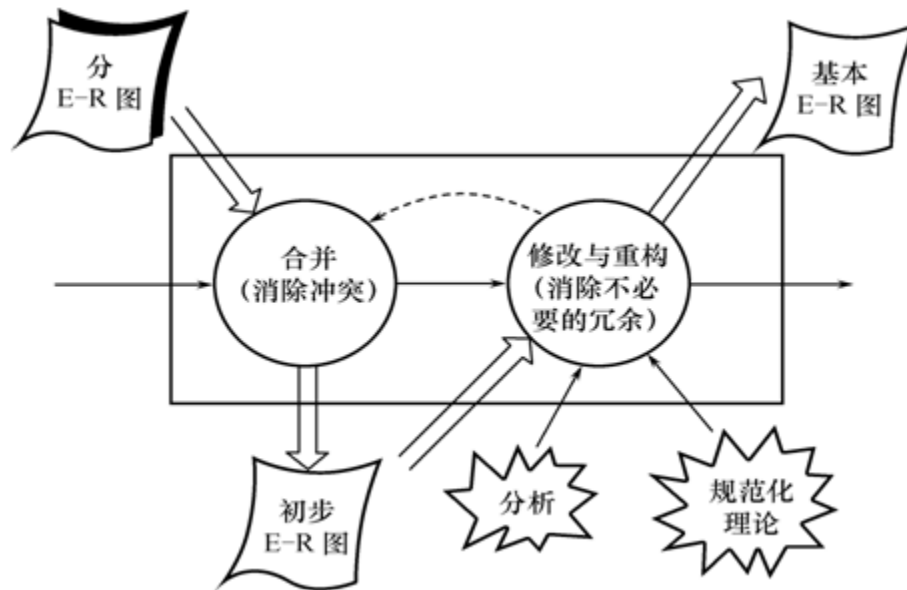
如果病房还要与医生实体发生联系，即一个医生负责几个病房的病人的医疗工作，则根据准则（2）病房应作为一个实体。



概念结构设计（续）

2. E-R图的集成(一般需要分两步)

- 合并。解决各分E-R图之间的冲突，将分E-R图合并起来生成初步E-R图。
- 修改和重构。消除不必要的冗余，生成基本E-R图。



概念结构设计（续）

（1）合并E-R图，生成初步E-R图

- 各个局部应用所面向的问题不同，各个子系统的E-R图之间必定会存在许多不一致的地方，称之为冲突。
- 子系统E-R图之间的冲突主要有三类：
 - ①属性冲突
 - ②命名冲突
 - ③结构冲突

概念结构设计（续）

①属性冲突

- 属性域冲突，即属性值的类型或取值范围不同
 - ✓ 例如产品号，有的部门把它定义为整数，有的部门把它定义为字符型。
 - ✓ 年龄，某些部门以出生日期形式表示职工年龄，而另一些部门用整数表示职工年龄。
- 属性取值单位冲突
 - ✓ 例如，零件的重量有的以公斤为单位，有的以斤为单位，有的以克为单位。

概念结构设计（续）

②命名冲突

- **同名异义**，即不同意义的对象在不同的局部应用中具有相同的名字，单位：所在部门 **vs.** 长度等度量。
- **异名同义**（一义多名），即同一意义的对象在不同的局部应用中具有不同的名字。
 - ✓ 如对科研项目，财务处称为项目，科研处称为课题，生产管理处称为工程。
- 命名冲突
 - ✓ 可能发生在实体、联系、属性级别
 - ✓ 通过讨论、协商等**行政手段**加以解决

概念结构设计（续）

③结构冲突

- 同一对象在不同应用中具有不同的抽象
 - ✓ 例如，职工在某一局部应用中被当作实体，而在另一局部应用中则被当作属性。
 - ✓ 解决方法：把属性变换为实体或把实体变换为属性，使同一对象具有相同的抽象。
- 同一实体在不同子系统的**E-R**图所包含的属性个数和属性排列次序不完全相同
 - ✓ 解决方法：使该实体的属性取各子系统的**E-R**图中属性的**并集**，再适当调整属性的次序。

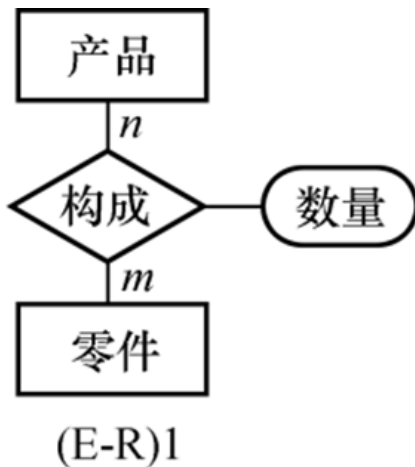
概念结构设计（续）

③结构冲突（续）

- 实体间的联系在不同的**E-R**图中为不同的类型。
 - ✓ 实体**E1**与**E2**在一个**E-R**图中是多对多联系，在另一个**E-R**图中是一对多联系
 - ✓ 解决方法是根据应用的语义对实体联系的类型进行综合或调整。

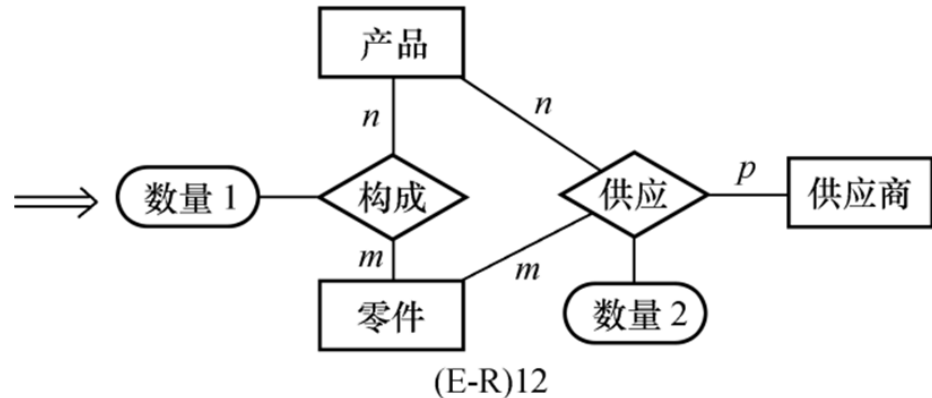
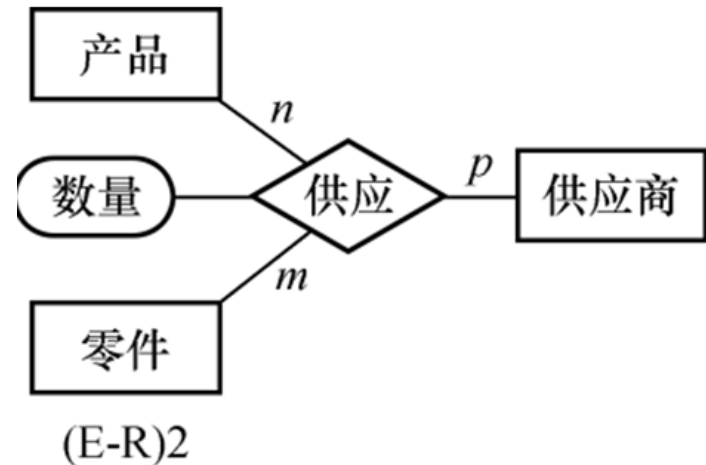
概念结构设计（续）

图7.25(a)中零件与产品之间存在多对多的联系“构成”



合并两个E-R图，
如图7.25(c)

图7.25(b)中产品、零件与供应商三者之间还存在多对多的联系“供应”



概念结构设计（续）

（2）修改与重构，消除不必要的冗余，设计基本E-R图

- 冗余的数据：可由基本数据导出的数据
- 冗余的联系：可由其他联系导出的联系。
- 消除冗余主要采用分析方法，即以数据字典和数据流图为依据，根据数据字典中关于数据项之间逻辑关系的说明来消除冗余。

概念结构设计（续）

- 如图7.26中， $Q_3=Q_1 \times Q_2$ ， $Q_4=\sum Q_5$ 。所以 Q_3 和 Q_4 是冗余数据，可以消去。并且由于 Q_3 消去，产品与材料间 $m:n$ 的冗余联系也应消去。

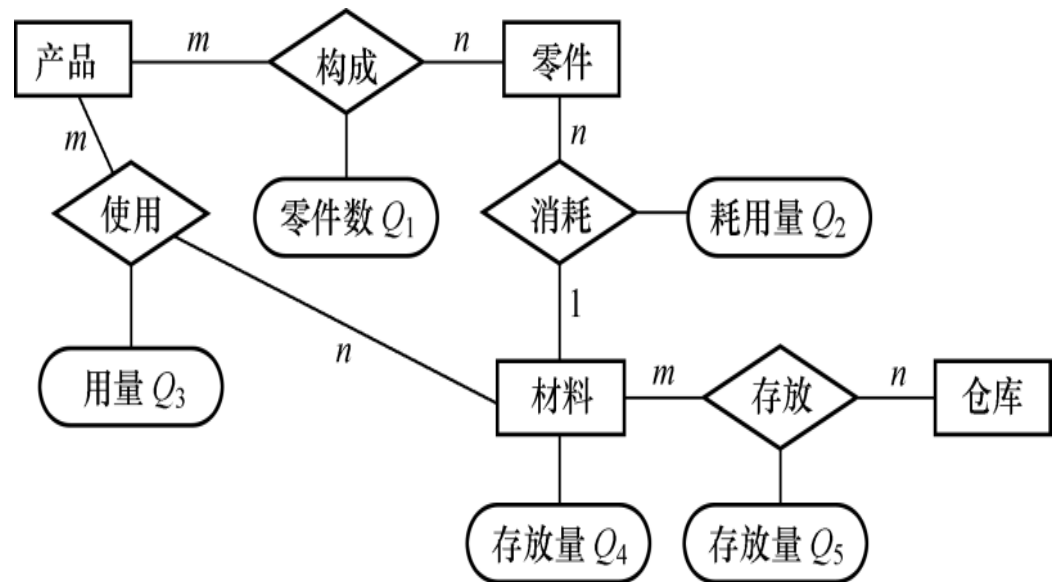


图7.26 消除冗余

并不是所有的冗余数据与冗余联系都必须加以消除，有时为了提高效率，不得不以冗余信息作为代价。

概念结构设计（续）

❖ 用规范化理论来消除冗余

①确定分E-R图实体之间的数据依赖。

- 实体之间一对一、一对多、多对多联系可以用实体码之间的函数依赖来表示，得到函数依赖集FL。

②求FL的最小覆盖GL，差集为 $D=FL-GL$ 。

- 逐一考察D中的函数依赖，确定是否冗余联系，若是，就把它去掉。

概念结构设计（续）

- 规范化理论的应用需要注意下面两个问题：
 - 冗余联系一定在D中，D中的联系不一定是冗余的；
 - 当实体之间存在多种联系时，要将实体之间的联系在形式上加以区分。

概念结构设计（续）

❖[例7.2] 某工厂管理信息系统的视图集成。

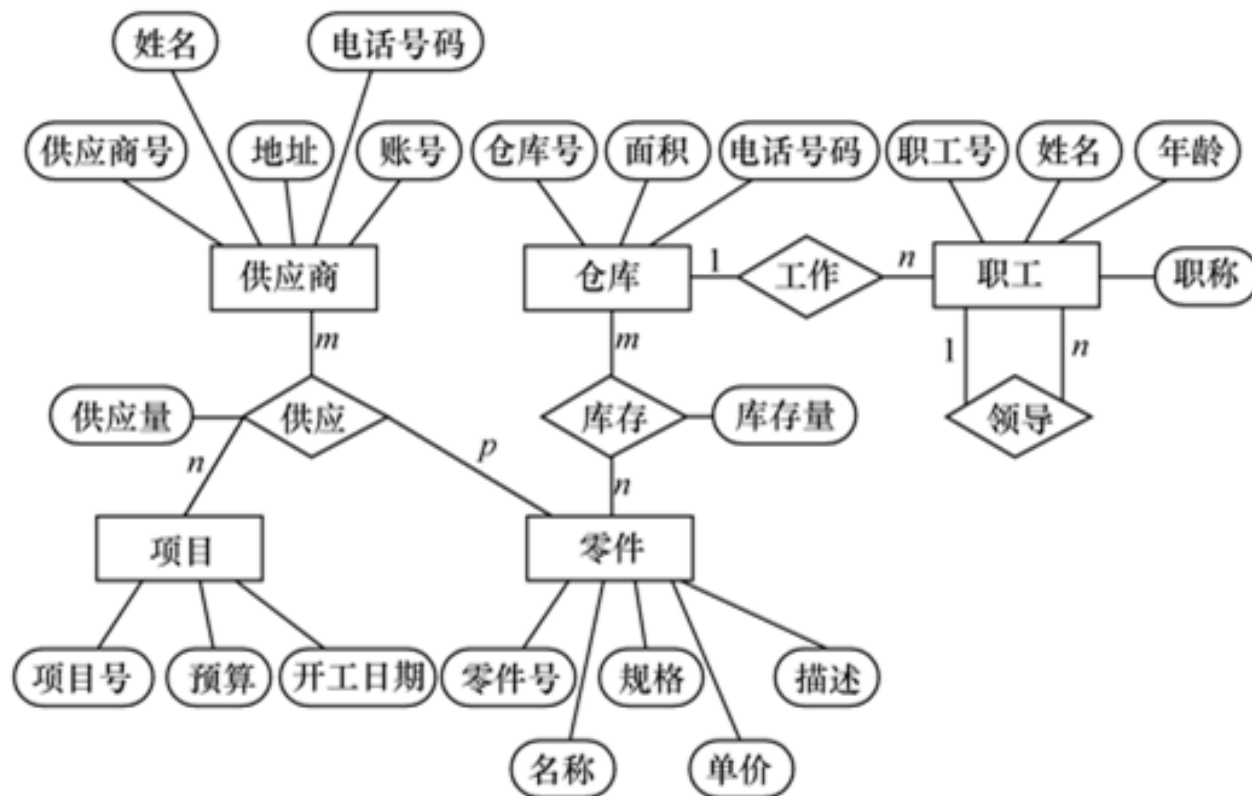


图7.11 工厂物资管理E-R图

概念结构设计（续）

❖ [例7.2] 某工厂管理信息系统的视图集成。

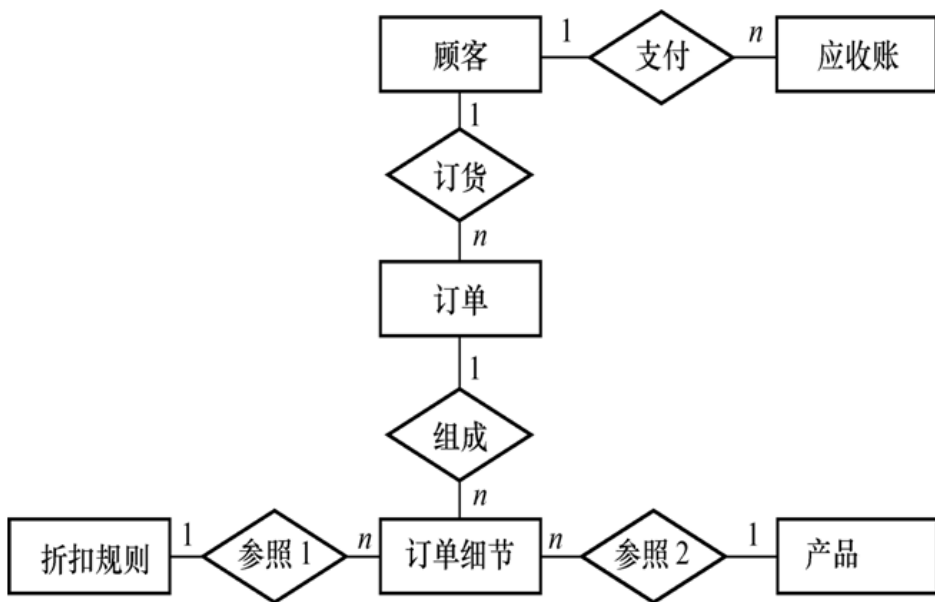


图7.23 销售管理系统的E-R图

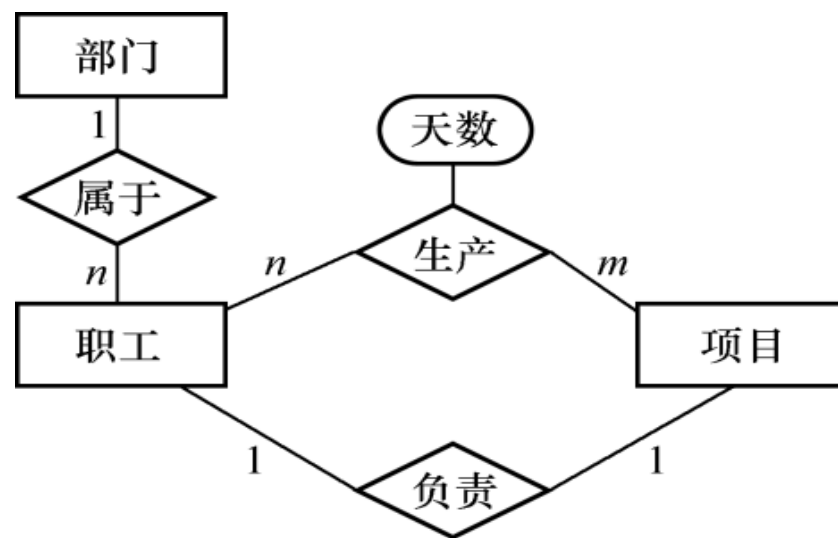
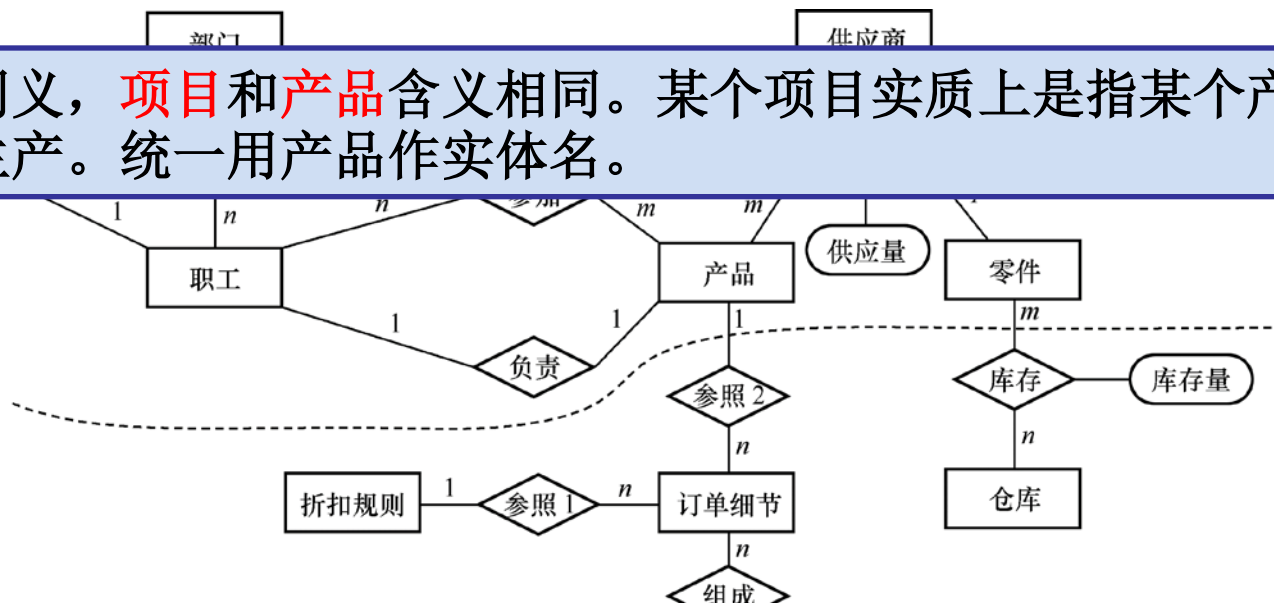


图7.27 劳动人事管理的分E-R图

概念结构设计（续）

❖ [例7.2] 某工厂管理信息系统的视图集成。

异名同义，**项目**和**产品**含义相同。某个项目实质上是指某个产品的生产。统一用产品作实体名。



库存管理中**职工与仓库的工作关系**已包含在劳动人事管理的**部门与职工**之间的联系之中，所以可以取消。职工之间**领导与被领导关系**可由部门与职工（经理）之间的**领导关系**、**部门与职工之间的从属关系两者**导出，所以也可以取消。

图7.28 某工厂管理信息系统的基本E-R图

第七章 数据库设计

7.1 数据库设计概述

7.2 需求分析

7.3 概念结构设计

7.4 逻辑结构设计

7.5 物理结构设计

7.6 数据库的实施和维护

7.7 小结

7.4 逻辑结构设计

❖ 逻辑结构设计的任务

- 把概念结构设计阶段设计好的**基本E-R图**转换为与选用数据库管理系统产品所支持的**数据模型相符合的逻辑结构(即关系模式)**

7.4 逻辑结构设计

7.4.1 E-R图向关系模型的转换

7.4.2 数据模型的优化

7.4.3 设计用户子模式

E-R图向关系模型的转换（续）

❖ 转换内容

- E-R图由实体型、实体的属性和实体型之间的联系三个要素组成
- 关系模型的逻辑结构是一组关系模式的集合
- 将E-R图转换为关系模型：将实体型、实体的属性和实体型之间的联系转化为关系模式

E-R图向关系模型的转换（续）

转换原则

1. 一个实体型转换为一个关系模式。

- 关系的属性：实体的属性
- 关系的码：实体的码

E-R图向关系模型的转换（续）

2. 实体型间的联系有以下不同情况

（1）一个1:1联系可以转换为一个独立的关系模式，也可以与任意一端对应的关系模式合并。

E-R图向关系模型的转换（续）

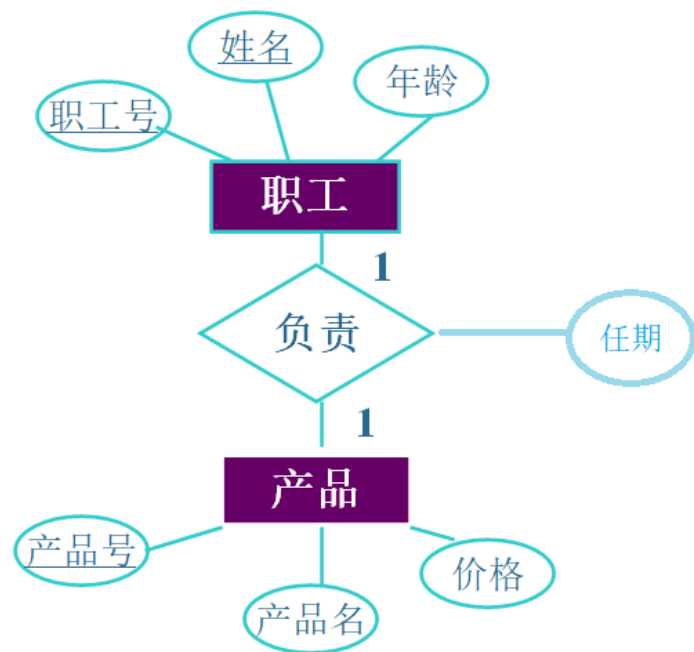
2. 实体型间的联系有以下不同情况

（1）一个1:1联系的转换

① 1:1联系转换为一个独立的关系模式

➤关系的属性：与该联系相连的**各实体的码**以及**联系本身的属性**

➤关系的候选码：每个实体的码均是该关系的候选码



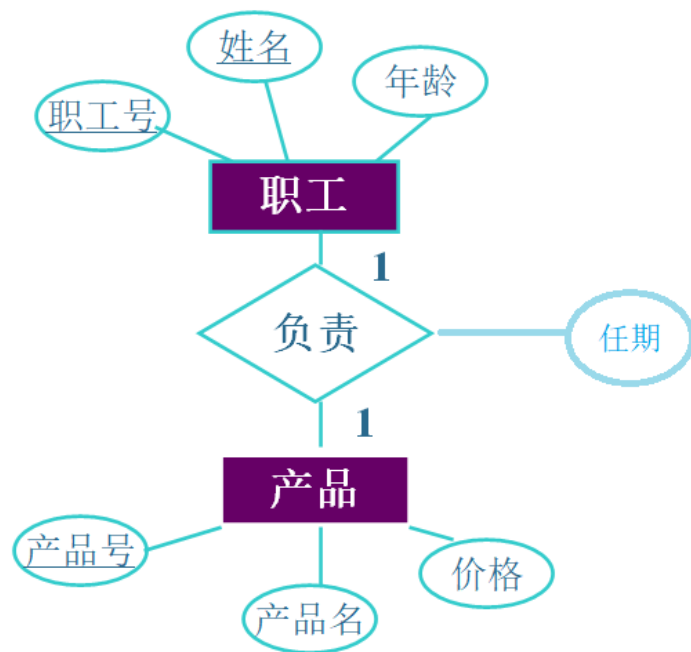
负责(职工号, 产品号, 任期)

E-R图向关系模型的转换（续）

（1）一个1:1联系的转换（续）

②与某一端实体对应的关系模式合并

- 合并后关系的属性：加入对应关系的码和联系本身的属性
- 合并后关系的码：不变



职工(职工号, 姓名, 年龄, 产品号, 任期)
或
产品(产品号, 产品名, 价格, 职工号, 任期)

E-R图向关系模型的转换（续）

（2）一个 $1:n$ 联系可以转换为一个独立的关系模式，也可以与 n 端对应的关系模式合并。

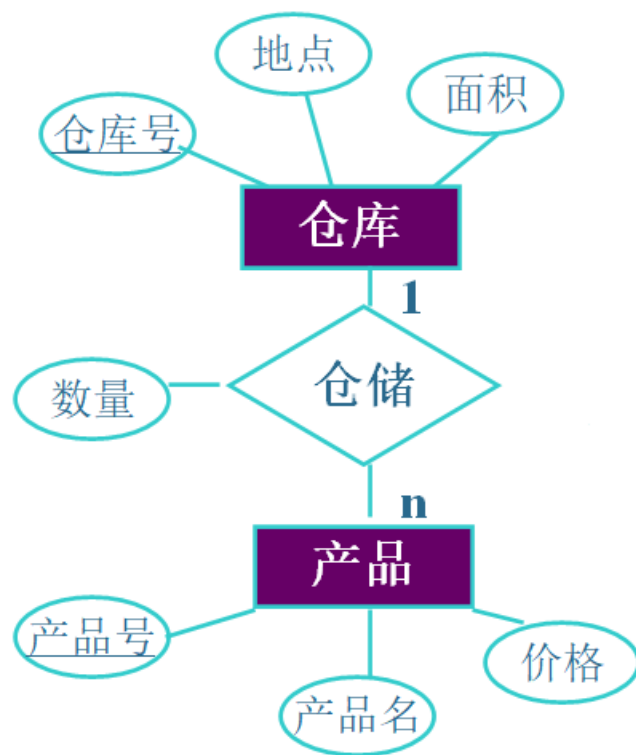
E-R图向关系模型的转换（续）

（2）一个1:n联系的转换

①转换为一个独立的关系模式

✓关系的属性：与该联系相连的各实体的码以及联系本身的属性

✓关系的码：**n**端实体的码



仓储(产品号, 仓库号, 数量)

E-R图向关系模型的转换（续）

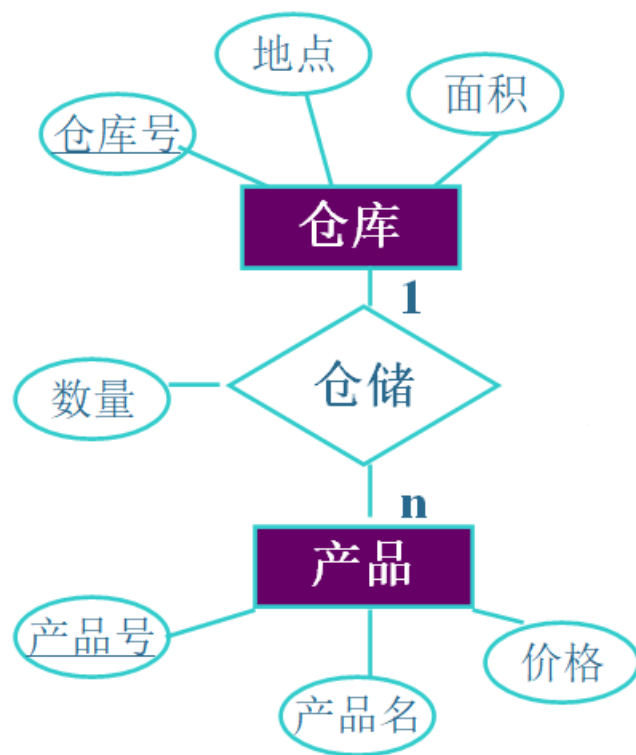
（2）一个1:n联系的转换（续）

②与n端对应的关系模式合并

✓合并后关系的属性：在n端关系中加入1端关系的码和联系本身的属性

✓合并后关系的码：不变

✓可以减少关系个数，一般更倾向于采用这种方法



产品(产品号, 产品名, 价格, 仓库号, 数量)

E-R图向关系模型的转换（续）

（3）一个 $m:n$ 联系转换为一个关系模式

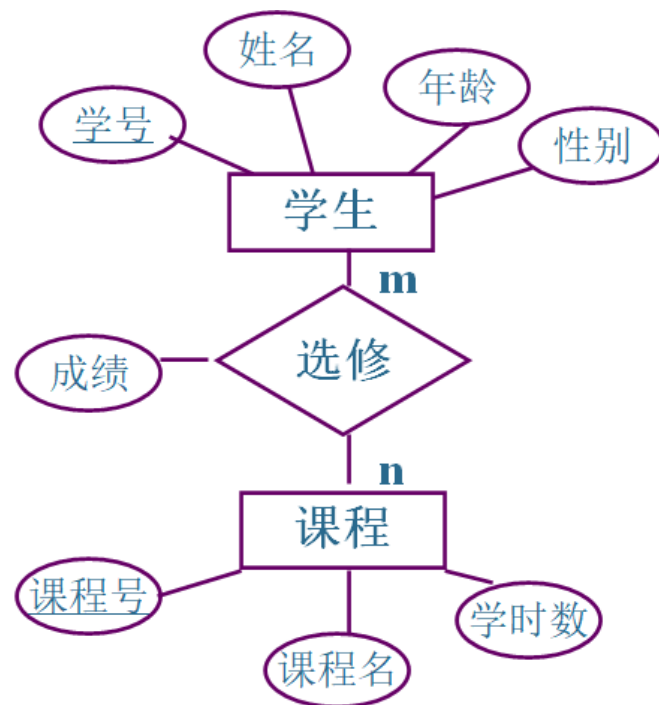
- 关系的属性：与该联系相连的各实体的码以及联系本身的属性
- 关系的码：各实体码的组合

E-R图向关系模型的转换（续）

(3) 一个 $m:n$ 联系转换为一个关系模式

[例] “选修”联系是一个 $m:n$ 联系，可以将它转换为如下关系模式，其中学号与课程号为关系的组合码：

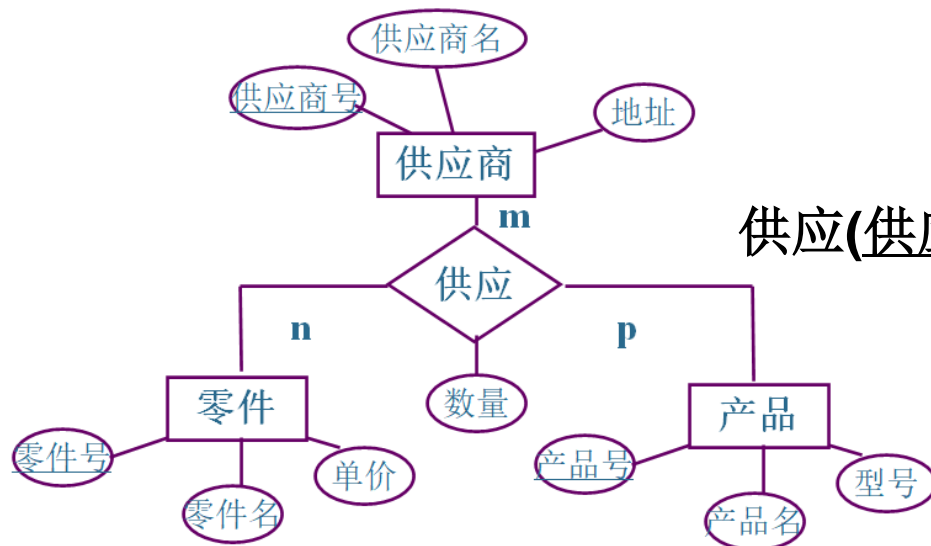
选修（学号， 课程号， 成绩）



E-R图向关系模型的转换（续）

（4）两个以上实体间多元联系转换为一个**关系模式**

- 关系的属性：与该多元联系相连的各实体的码以及联系本身的属性
- 关系的码：各实体码的组合



供应(供应商号, 零件号, 产品号, 数量)

E-R图向关系模型的转换（续）

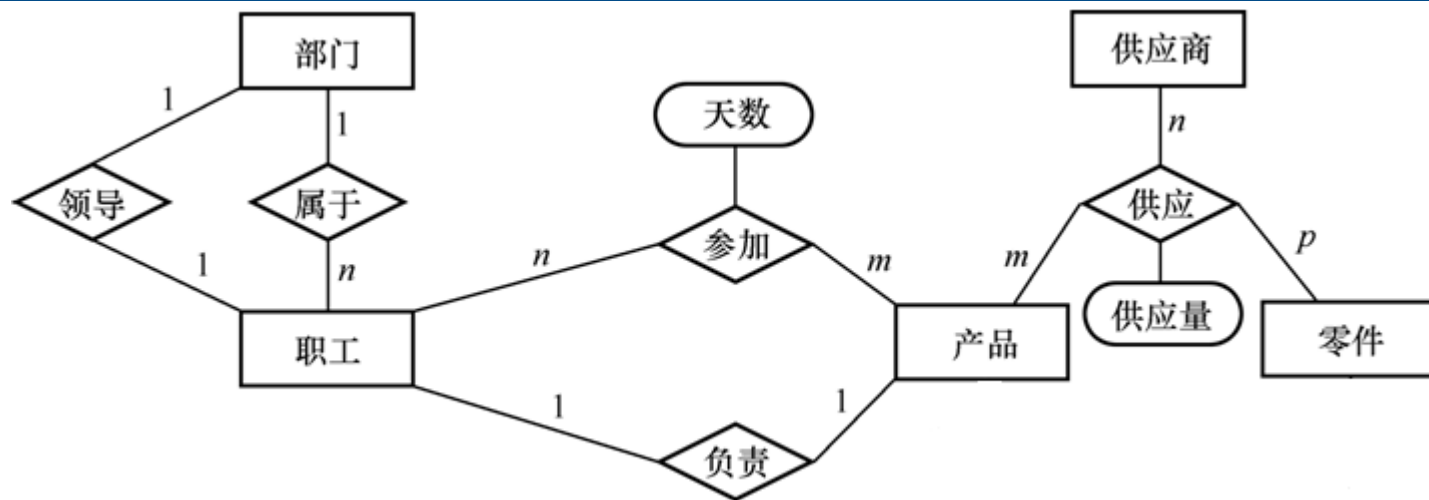
（5）具有相同码的关系模式可合并

- 目的：减少系统中的关系个数
- 合并方法：
 - ✓ 将其中一个关系模式的全部属性加入到另一个关系模式中
 - ✓ 然后去掉其中的同义属性（不一定同名）
 - ✓ 适当调整属性的次序

Student(Sno, Sname, Sage, Sdept, Ssex)

StuFoodCard(Sno, Amount)

E-R图向关系模型的转换（续）



- 部门（部门号，部门名，经理的职工号，...）
- 职工（职工号、部门号，职工名，职务，...）
- 产品（产品号，产品名，产品组长的职工号，...）
- 职工工作（职工号，产品号，工作天数，...）
- 供应商（供应商号，姓名，...）
- 零件（零件号，零件名，...）
- 供应（产品号，供应商号，零件号，供应量）

7.4 逻辑结构设计

7.4.1 E-R图向关系模型的转换

7.4.2 数据模型的优化

7.4.3 设计用户子模式

7.4.2 数据模型的优化

- ❖ 一般的数据模型还需要向特定数据库管理系统规定的模型进行转换。
- ❖ 转换的**主要依据是所选用的数据库管理系统的功能及限制**，没有通用规则。
- ❖ 对于关系模型来说，这种转换通常都比较简单。

数据模型的优化（续）

- ❖ 数据库逻辑设计的结果不是唯一的。
- ❖ 数据模型的优化：得到初步数据模型后，还应该适当地修改、调整数据模型的结构，以进一步提高数据库应用系统的性能。
- ❖ 关系数据模型的优化通常以规范化理论为指导。

数据模型的优化（续）

优化数据模型的方法：

（1）确定数据依赖

- 按需求分析阶段所得到的语义，分别写出每个关系模式内部各属性之间的数据依赖以及不同关系模式属性之间数据依赖。

（2）对于各个关系模式之间的数据依赖进行极小化处理，消除冗余的联系。

数据模型的优化（续）

- （3）按照数据依赖的理论对关系模式进行分析，考察是否存在部分函数依赖、传递函数依赖、多值依赖等，确定各关系模式分别属于第几范式。
- （4）按照需求分析阶段得到的各种应用对数据处理的要求，分析对于这样的应用环境这些模式是否合适，确定是否要对它们进行合并或分解。

数据模型的优化（续）

- 并不是规范化程度越高的关系就越优
 - 当查询经常涉及两个或多个关系模式的属性时，系统必须经常地进行连接运算
 - 连接运算的代价是相当高的
 - 因此在这种情况下，第二范式甚至第一范式也许是适合的。

数据模型的优化（续）

- 非BCNF的关系模式虽然会存在不同程度的更新异常，但如果在实际应用中对此关系模式只是查询，并不执行更新操作，就不会产生实际影响。
- 对于一个具体应用来说，到底规范化进行到什么程度，需要权衡响应时间和潜在问题两者的利弊才能决定

数据模型的优化（续）

（5）对关系模式进行必要分解，提高数据操作效率和存储空间的利用率。

- 常用分解方法

- 水平分解
- 垂直分解

数据模型的优化（续）

■ 水平分解

- 什么是水平分解：把关系的元组分为若干子集合，定义每个子集合为一个子关系，以提高系统的效率。
- 如何分解
 - ✓ 对符合**80/20**规则的，经常被使用的数据（约**20%**）水平分解出来，形成一个子关系。
 - ✓ 水平分解为若干子关系，使每个事务存取的数据对应一个子关系。

数据模型的优化（续）

■ 垂直分解

- 什么是垂直分解：把关系模式R的属性分解为若干子集合，形成若干子关系模式。
- 垂直分解的原则：经常在一起使用的属性从R中分解出来形成一个子关系模式
- 垂直分解的优点：可以提高某些事务的效率
- 垂直分解的缺点：可能使另一些事务不得不执行连接操作，降低了效率

数据模型的优化（续）

■ 垂直分解的适用范围

- 取决于分解后R上所有事务总效率是否得到提高

■ 进行垂直分解的方法

- 简单情况：直观分解
- 复杂情况：用第6章中的模式分解算法
- 垂直分解必须不损失关系模式的语义（保持无损连接性和保持函数依赖）

7.4 逻辑结构设计

7.4.1 E-R图向关系模型的转换

7.4.2 数据模型的优化

7.4.3 设计用户子模式

7.4.3 设计用户子模式

- ❖ 定义数据库模式主要是从系统的**时间效率**、**空间效率**、**易维护**等角度出发。
- ❖ 定义用户外模式时应该更注重考虑**用户的习惯与方便**。包括三个方面。

设计用户子模式（续）

（1）使用更符合用户习惯的别名

- 合并各分E-R图时消除命名冲突的工作，以使数据库系统中**关系和属性具有唯一的名字**。这在设计数据库整体结构时是非常必要的。
- 用视图机制在设计用户视图时可以**重新定义某些属性名**，使其与用户习惯一致，以方便使用。

设计用户子模式（续）

（2）针对不同级别的用户定义不同的视图，保证系统的安全性。

- 关系模式：产品（产品号，产品名，规格，单价，生产车间，生产负责人，产品成本，产品合格率，质量等级），可以在产品关系上建立两个视图：
 - ✓ 为一般顾客建立视图：产品1（产品号，产品名，规格，单价）
 - ✓ 为产品销售部门建立视图：产品2（产品号，产品名，规格，单价，车间，生产负责人）

设计用户子模式（续）

（3）简化用户对系统的使用

- 如果某些局部应用中经常要使用某些很复杂的查询，为了方便用户，可以将这些复杂查询定义为视图。

第七章 数据库设计

7.1 数据库设计概述

7.2 需求分析

7.3 概念结构设计

7.4 逻辑结构设计

7.5 物理结构设计

7.6 数据库的实施和维护

7.7 小结

7.5 数据库的物理设计

❖ 什么是数据库的物理设计

- 数据库在物理设备上的**存储结构与存取方法**称为数据库的**物理结构**，它依赖于选定的数据库管理系统。
- 为一个给定的逻辑数据模型选取**一个最适合应用要求的物理结构的过程**，是数据库的物理设计。

数据库的物理设计（续）

❖ 数据库物理设计的步骤

■ 确定数据库的物理结构

- 在关系数据库中主要指存储结构和存取方法;

■ 对物理结构进行评价

- 评价的重点是时间和空间效率

■ 若评价结果满足原设计要求，则可进入到物理实施阶段。否则，就需要重新设计或修改物理结构，有时甚至要返回逻辑设计阶段修改数据模型。

7.5 数据库的物理设计

7.5.1 数据库物理设计的内容和方法

7.5.2 关系模式存取方法选择

7.5.3 确定数据库的存储结构

7.5.4 评价物理结构

7.5.1 数据库物理设计的内容和方法

❖ 设计物理数据库结构的准备工作

- 充分了解应用环境，详细分析要运行的事务，以获得选择物理数据库设计所需参数。
- 充分了解所用关系型数据库管理系统的内部特征，特别是系统提供的存取方法和存储结构。

数据库物理设计的内容和方法（续）

❖ 选择物理数据库设计所需参数

■ 数据库查询事务

- 查询的关系、查询条件所涉及的属性、连接条件所涉及的属性、查询的投影属性

■ 数据更新事务

- 被更新的关系、每个关系上的更新操作条件所涉及的属性、修改操作要改变的属性值

■ 每个事务在各关系上运行的频率和性能要求

数据库物理设计的内容和方法（续）

❖ 关系数据库物理设计的内容

- 为关系模式选择存取方法（建立存取路径）
- 设计关系、索引等数据库文件的物理存储结构

7.5 数据库的物理设计

7.5.1 数据库物理设计的内容和方法

7.5.2 关系模式存取方法选择

7.5.3 确定数据库的存储结构

7.5.4 评价物理结构

7.5.2 关系模式存取方法选择

- ❖ 数据库系统是多用户共享的系统，对同一个关系建立多条存取路径才能满足多用户的多种应用要求。
- ❖ 物理结构设计任务之一是根据关系数据库管理系统支持的存取方法确定选择哪些存取方法。

关系模式存取方法选择（续）

❖ 数据库管理系统常用存取方法

1. **B+**树索引存取方法
2. **Hash**索引存取方法
3. 聚簇存取方法

1. B+树索引存取方法的选择

❖ 选择索引存取方法的主要内容

- 根据应用要求确定
 - 对哪些属性列建立索引
 - 对哪些属性列建立组合索引
 - 对哪些索引要设计为唯一索引

B+树索引存取方法的选择（续）

❖ 选择索引存取方法的一般规则

- 如果一个（或一组）属性**经常在查询条件**出现，则考虑在这个（或这组）属性上建立索引（或组合索引）
- 如果一个属性经常作为最大值和最小值等聚集函数的参数，则考虑在这个属性上建立索引
- 如果一个（或一组）属性经常在连接操作的连接条件中出现，则考虑在这个（或这组）属性上建立索引

B+树索引存取方法的选择（续）

❖ 关系上定义的索引数过多会带来较多的额外开销

- 维护索引的开销
- 查找索引的开销

2. HASH存取方法的选择

❖ 选择Hash存取方法的规则

- 如果一个关系的属性主要出现在等值连接条件中或主要出现在等值比较选择条件中，而且满足下列两个条件之一
 - 该关系的大小可预知，而且不变；
 - 该关系的大小动态改变，但所选用的数据库管理系统提供了动态Hash存取方法。

3. 聚簇存取方法的选择

❖ 什么是聚簇

- 为了提高某个属性（或属性组）的查询速度，把这个或这些属性上具有相同值的元组集中存放在连续的物理块中称为聚簇。
- 该属性（或属性组）称为聚簇码（**cluster key**）

聚簇存取方法的选择（续）

❖ 聚簇索引

- 建立聚簇索引后，基表中数据也需要按指定的聚簇属性值的升序或降序存放，即聚簇索引的索引项顺序与表中元组的物理顺序一致。
- 在一个基本表上最多只能建立一个聚簇索引

❖ 聚簇索引的适用条件

- 很少对基表进行增删操作
- 很少对其中的变长列进行修改操作

聚簇存取方法的选择（续）

❖ 聚簇的用途：对于某些类型的查询，可以提高查询效率

1. 大大提高按聚簇属性进行查询的效率

[例] 假设学生关系按所在系建有索引，现在要查询信息系的所有学生名单。

✓ 计算机系的**500**名学生分布在**500**个不同的物理块上时，至少要执行**500**次I/O操作。

✓ 如果将同一系的学生元组集中存放，则每读一个物理块可得到多个满足查询条件的元组，从而显著地减少了访问磁盘的次数。

聚簇存取方法的选择（续）

2. 节省存储空间

- ✓ 聚簇以后，聚簇码相同的元组集中在一起，因而聚簇码值不必在每个元组中重复存储，只要在一组中存一次就行了。

聚簇存取方法的选择（续）

❖ 聚簇的局限性

- 聚簇只能提高某些特定应用的性能
- 建立与维护聚簇的开销相当大
 - 对已有关系建立聚簇，将导致关系中元组的物理存储位置移动，并使此关系上原有的索引无效，必须重建。
 - 当一个元组的聚簇码改变时，该元组的存储位置也要做相应改变。

聚簇存取方法的选择（续）

❖ 聚簇的适用范围

- 既适用于单个关系独立聚簇，也适用于多个关系组合聚簇
- 当通过聚簇码进行访问或连接是该关系的主要应用，且与聚簇码无关的其他访问很少或是次要的时，可以使用聚簇
 - 尤其当SQL语句中包含有与聚簇码有关的**ORDER BY, GROUP BY, UNION, DISTINCT**等子句时，聚簇特别有利，可以省去或减化对结果集的排序操作

聚簇存取方法的选择（续）

❖ 选择聚簇存取方法

■ 设计候选聚簇

- （1）常在一起进行连接操作的关系可以建立组合聚簇
- （2）如果一个关系的一组属性经常出现在相等比较条件中，则该单个关系可建立聚簇；
- （3）如果一个关系的一个（或一组）属性上的值重复率很高，则此单个关系可建立聚簇。

聚簇存取方法的选择（续）

- 检查候选聚簇中的关系，取消其中不必要的关系
 - （1）从聚簇中删除经常进行全表扫描的关系
 - （2）从聚簇中删除更新操作远多于连接操作的关系
 - （3）从聚簇中删除重复出现的关系

当一个关系同时加入多个聚簇时，必须从这多个聚簇方案（包括不建立聚簇）中选择一个较优的，即在这个聚簇上运行各种事务的总代价最小。

7.5 数据库的物理设计

7.5.1 数据库物理设计的内容和方法

7.5.2 关系模式存取方法选择

7.5.3 确定数据库的存储结构

7.5.4 评价物理结构

7.5.3 确定数据库的存储结构

- ❖ 确定数据库物理结构主要指确定数据的**存放位置**和**存储结构**，包括：确定关系、索引、聚簇、日志、备份等的存储安排和存储结构，确定系统配置等。
- ❖ 确定数据的存放位置和存储结构要综合考虑**存取时间、存储空间利用率和维护代价**3个方面的因素。

确定数据库的存储结构（续）

❖ 影响数据存放位置和存储结构的因素

- 硬件环境
- 应用需求
 - 存取时间
 - 存储空间利用率
 - 维护代价

这三个方面常常是相互矛盾的， 必须进行权衡， 选择一个折中方案

1. 确定数据的存放位置

❖ 基本原则

- 根据应用情况，易变部分与稳定部分分开存放，经常存取部分与存取频率较低部分分开存放

❖ [例]

- 可以将比较大的表分别放在两个磁盘上，以加快存取速度，这在多用户环境下特别有效。
- 可以将日志文件与数据库对象（表、索引等）放在不同的磁盘以改进系统的性能。

2. 确定系统配置

❖ 数据库管理系统一般都提供了一些存储分配参数

- 同时使用数据库的用户数
- 同时打开的数据库对象数
- 内存分配参数
- 缓冲区分配参数（使用的缓冲区长度、个数）
- 存储分配参数
- 物理块的大小
- 物理块装填因子
- 时间片大小
- 数据库的大小
- 锁的数目等

确定系统配置（续）

- ❖ 系统都为这些变量赋予了合理的缺省值，在进行物理设计时需要根据应用环境确定这些参数值，以使系统性能最优。
- ❖ 在物理设计时对系统配置变量的调整只是初步的，要根据系统实际运行情况做进一步的调整，以切实改进系统性能。

7.5 数据库的物理设计

7.5.1 数据库物理设计的内容和方法

7.5.2 关系模式存取方法选择

7.5.3 确定数据库的存储结构

7.5.4 评价物理结构

7.5.4 评价物理结构

- ❖ 对数据库物理设计过程中产生的多种方案进行评价，从中选择一个较优的方案作为数据库的物理结构。
- ❖ 评价方法
 - 定量估算各种方案
 - 存储空间
 - 存取时间
 - 维护代价
- ❖ 对估算结果进行权衡、比较，选择出一个较优的合理的物理结构。

第七章 数据库设计

7.1 数据库设计概述

7.2 需求分析

7.3 概念结构设计

7.4 逻辑结构设计

7.5 物理结构设计

7.6 数据库的实施和维护

7.7 小结

7.6 数据库的实施和维护

7.6.1 数据的载入和应用程序的调试

7.6.2 数据库的试运行

7.6.3 数据库的运行和维护

数据的载入

❖ 数据库结构建立好后，可以向数据库中装载数据。

组织数据入库是数据库实施阶段最主要的工作。

❖ 数据装载方法

- 人工方法

- 计算机辅助数据入库

应用程序的调试

- ❖ 数据库**应用程序的设计**应该与**数据库设计**并行进行
- ❖ 在组织数据入库的同时还要调试应用程序
- ❖ 应用程序的设计、编码和调试的方法、步骤在软件工程等课程中有详细讲解，这里不赘述

7.6 数据库的实施和维护

7.6.1 数据的载入和应用程序的调试

7.6.2 数据库的试运行

7.6.3 数据库的运行和维护

7.6.2 数据库的试运行

- ❖ 应用程序调试完成，且已有一小部分数据入库后，就可以开始对数据库系统进行联合调试，也称数据库的**试运行**。
- ❖ 主要工作包括：
 - 功能测试：实际运行应用程序，执行对数据库的各种操作，测试应用程序的各种功能。
 - 性能测试：测量系统的性能指标，分析是否符合设计目标。

数据库的试运行（续）

❖ 数据库性能指标的测量

- 数据库物理设计阶段在评价数据库结构估算时间和空间指标时，作了许多简化和假设，忽略了许多次要因素，因此结果必然很粗糙。
- 数据库试运行则是要实际测量系统的各种性能指标（不仅是时间、空间指标），如果结果不符合设计目标，则需要返回物理设计阶段，调整物理结构，修改参数；有时甚至需要返回逻辑设计阶段，调整逻辑结构。

数据库的试运行（续）

❖ 数据的分期入库

- 重新设计物理结构甚至逻辑结构，会导致数据重新入库
- 由于数据入库工作量实在太太大，所以可以采用分期输入数据的方法
 - 先输入小批量数据供先期联合调试使用
 - 待试运行基本合格后再输入大批量数据
 - 逐步增加数据量，逐步完成运行评价

数据库的试运行（续）

❖ 数据库的转储和恢复

- 在数据库试运行阶段，系统还不稳定，硬、软件故障随时都可能发生
- 系统的操作人员对新系统还不熟悉，误操作也不可避免
- 因此必须做好数据库的转储和恢复工作，尽量减少对数据库的破坏

7.6 数据库的实施和维护

7.6.1 数据的载入和应用程序的调试

7.6.2 数据库的试运行

7.6.3 数据库的运行和维护

7.6.3 数据库的运行和维护

❖ 在数据库运行阶段，对数据库经常性的维护工作主要是由数据库管理员完成的，包括：

1. 数据库的转储和恢复

- 数据库管理员针对不同的应用要求制定不同的转储计划，定期对数据库和日志文件进行备份。
- 一旦发生介质故障，利用数据库备份及日志文件备份，尽快将数据库恢复到某种一致性状态。

数据库的运行和维护（续）

2. 数据库的安全性、完整性控制

- 初始定义

- ✓ **DBA**根据用户实际需要授予不同的操作权限，
根据应用环境定义不同的完整性约束条件

- 修改定义

- ✓ 当应用环境发生变化，对**安全性**的要求也会发生变化，需要根据实际情况修改原有的安全性控制
- ✓ 由于应用环境发生变化，数据库的**完整性**约束条件也会变化，也需要不断修正，以满足用户要求

数据库的运行和维护（续）

3. 数据库性能的监督、分析和改进

- 数据库运行过程中，**DBA**必须监督系统运行，对监测数据进行分析，找出改进系统性能的方法。
 - ✓ 利用监测工具获取系统运行过程中一系列性能参数的值
 - ✓ 通过仔细分析这些数据，判断当前系统是否处于最佳运行状态
 - ✓ 如果不是，则需要通过调整某些参数来进一步改进数据库性能

数据库的运行和维护（续）

4. 数据库的重组与重构造

（1）数据库的重组

- 为什么要重组数据库

- ✓ 数据库运行一段时间后，由于记录不断增、删、改，会使数据库的物理存储变坏，从而降低数据库存储空间的使用率和数据的存取效率，使数据库的性能下降。

数据库的运行和维护（续）

■ 重组组织的形式

- 全部重组组织
- 部分重组组织

✓ 只对频繁增、删的表进行重组组织

■ 重组组织的目标

- 提高系统性能

数据库的运行和维护（续）

- 重组组织的工作
 - 按原设计要求
 - ✓ 重新安排存储位置
 - ✓ 回收垃圾
 - ✓ 减少指针链
 - 数据库的重组组织不会改变原设计的数据逻辑结构和物理结构
- 数据库管理系统一般都提供重组组织数据库使用的实用程序，帮助**DBA**重新组织数据库。

数据库的运行和维护（续）

（2）数据库的重构造

■ 为什么要进行数据库的重构造

- 数据库应用环境发生变化，会导致**实体及实体间的联系也发生相应的变化**，使原有的数据库设计不能很好地满足新的需求
 - ✓ 增加新的应用或新的实体
 - ✓ 取消某些已有应用
 - ✓ 改变某些已有应用

数据库的运行和维护（续）

- 数据库重构造的主要工作
 - 根据新环境调整数据库的模式和内模式
 - ✓ 增加或删除某些数据项
 - ✓ 改变数据项的类型
 - ✓ 增加或删除某个表
 - ✓ 改变数据库的容量
 - ✓ 增加或删除某些索引

数据库的运行和维护（续）

- 重构造数据库的程度是有限的
 - 若应用变化太大，已无法通过重构数据库来满足新的需求，或重构数据库的代价太大，则表明现有数据库应用系统的生命周期已经结束，应该重新设计新的数据库应用系统。

第七章 数据库设计

7.1 数据库设计概述

7.2 需求分析

7.3 概念结构设计

7.4 逻辑结构设计

7.5 物理结构设计

7.6 数据库的实施和维护

7.7 小结

7.7 小结

❖ 数据库的设计过程

- 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
- 数据库运行维护
- 设计过程中往往还会有许多反复

小结（续）

❖ 数据库各级模式的形成

- 需求分析阶段：综合各个用户的应用需求（现实世界的需求）
- 概念设计阶段：**概念模式**（信息世界模型），用**E-R图**来描述
- 逻辑设计阶段：**逻辑模式**、**外模式**
- 物理设计阶段：**内模式**

小结（续）

❖ 概念结构设计

- E-R模型的基本概念和图示方法
- E-R模型的设计
- 把E-R模型转换为关系模型的方法

小结（续）

- ❖ 在逻辑设计阶段将**E-R**图转换成具体的数据库产品支持的数据模型(关系模型)，形成数据库逻辑模式
- ❖ 然后根据用户处理的要求和安全性的考虑，在基本表的基础上建立必要的视图，形成数据的外模式
- ❖ 在物理设计阶段根据**DBMS**特点和处理的需要，进行物理存储安排，设计索引，形成数据库内模式