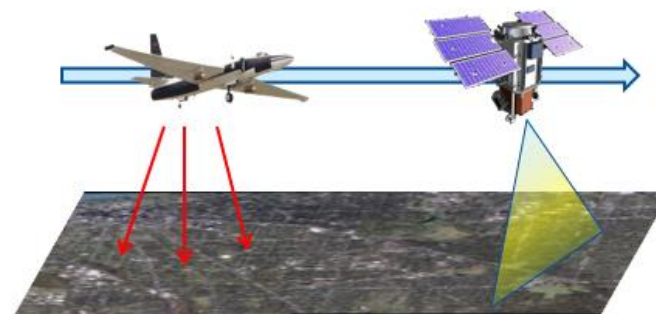


2012  
LiDAR



2016  
光学高分



个人主页:

<http://grzy.cug.edu.cn/luohui1>

# 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

- 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（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

➤ 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（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

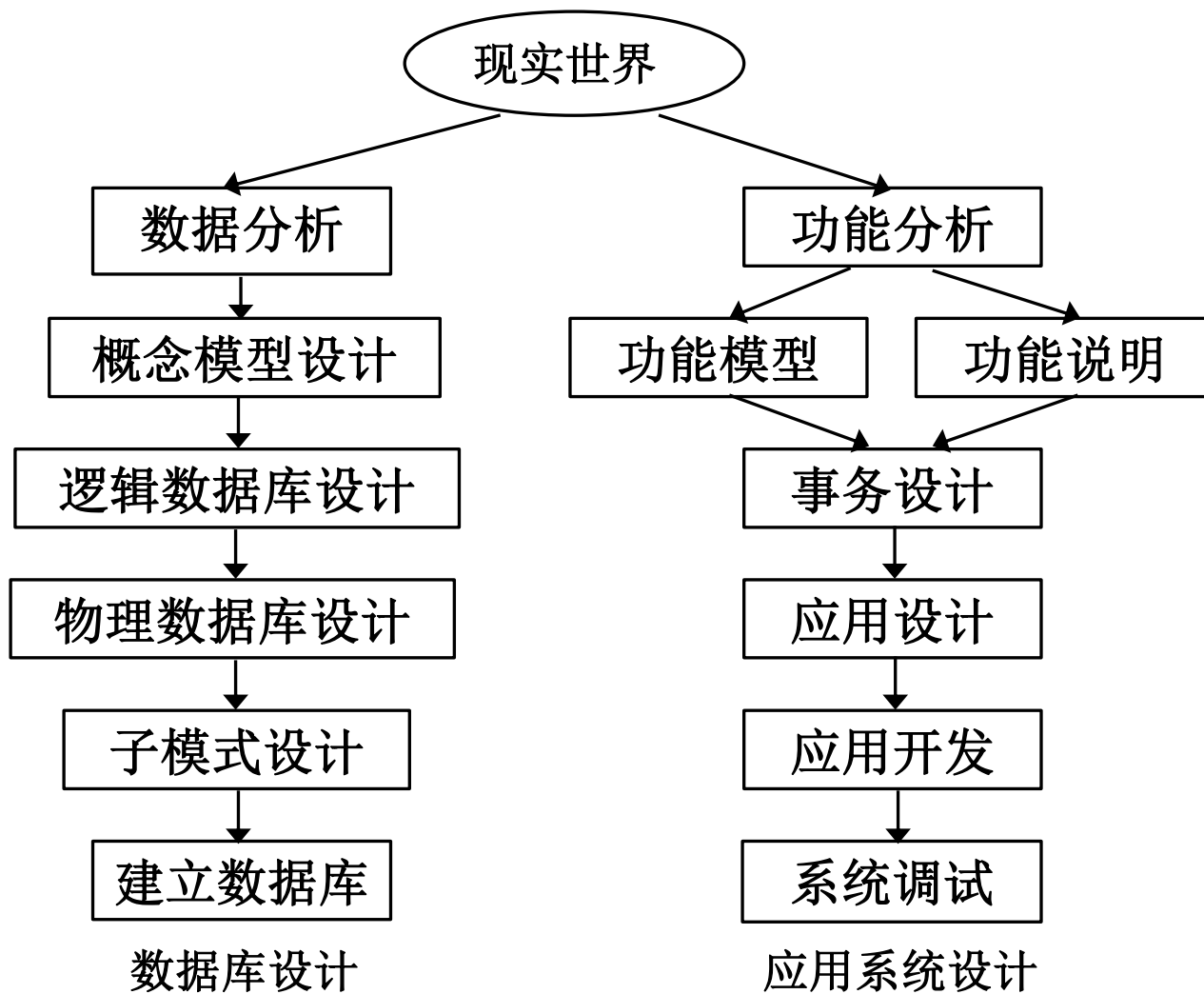
- 传统的软件工程：重行为设计

- ✓ 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）



结构和行为分离的设计



## 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

## 7.1 数据库设计概述

---

### 7.1.1 数据库设计的特点

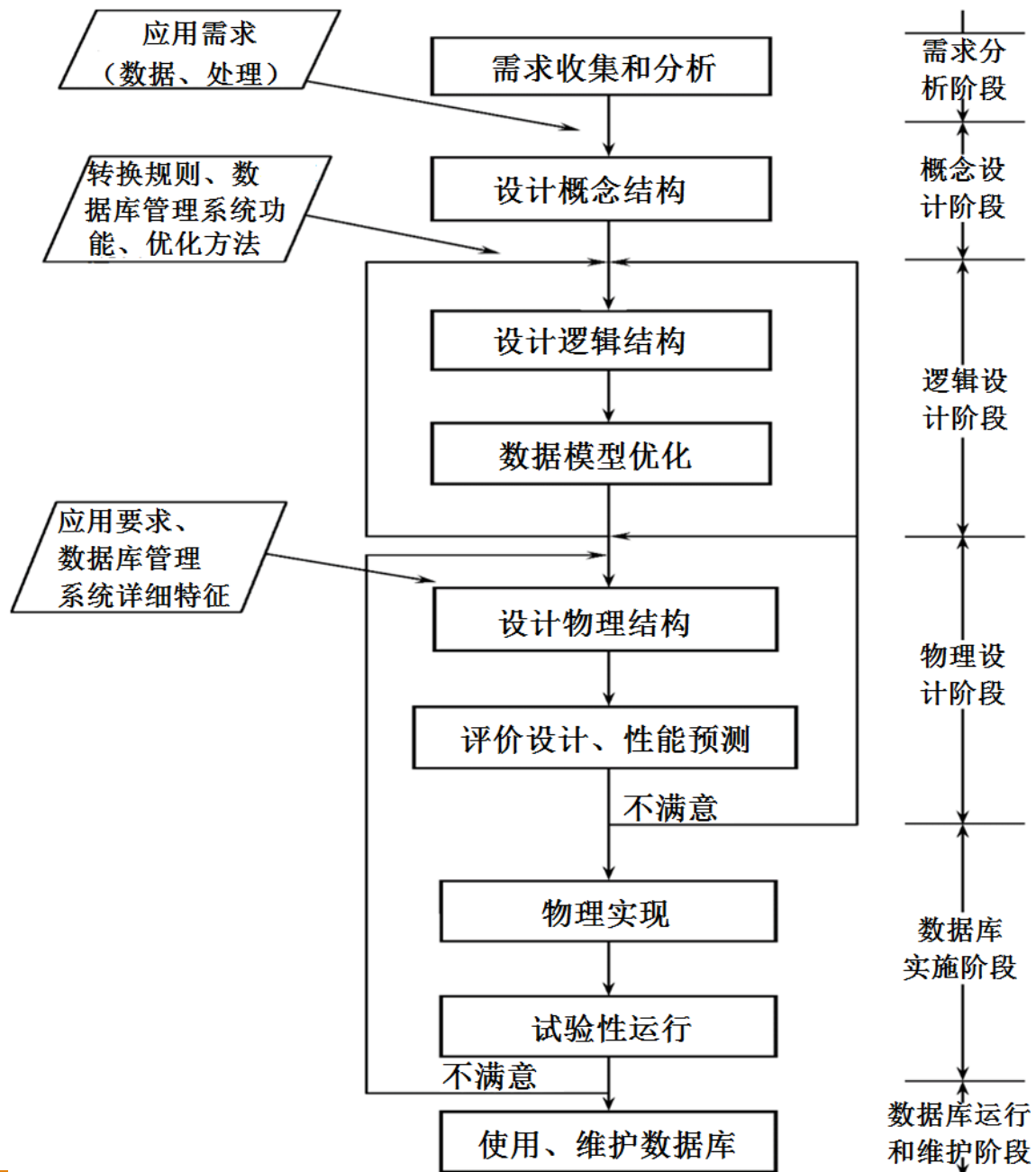
### 7.1.2 数据库设计方法

### 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）

## 1. 需求分析阶段

- 是否做得充分与准确，决定了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，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



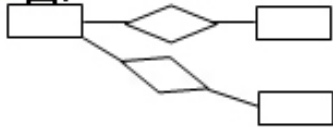
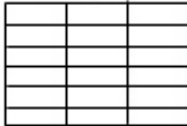
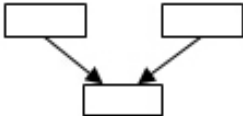
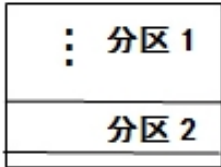

设计阶段	设计描述
需求分析	数字字典、全系统中数据项、数据结构、数据流、数据存储的描述
概念结构设计	概念模型 (E-R 图)  数据字典
逻辑结构设计	某种数据模型 <div style="display: flex; justify-content: space-around; align-items: flex-start;"> <div style="text-align: center;">             关系              </div> <div style="text-align: center;">             非关系              </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

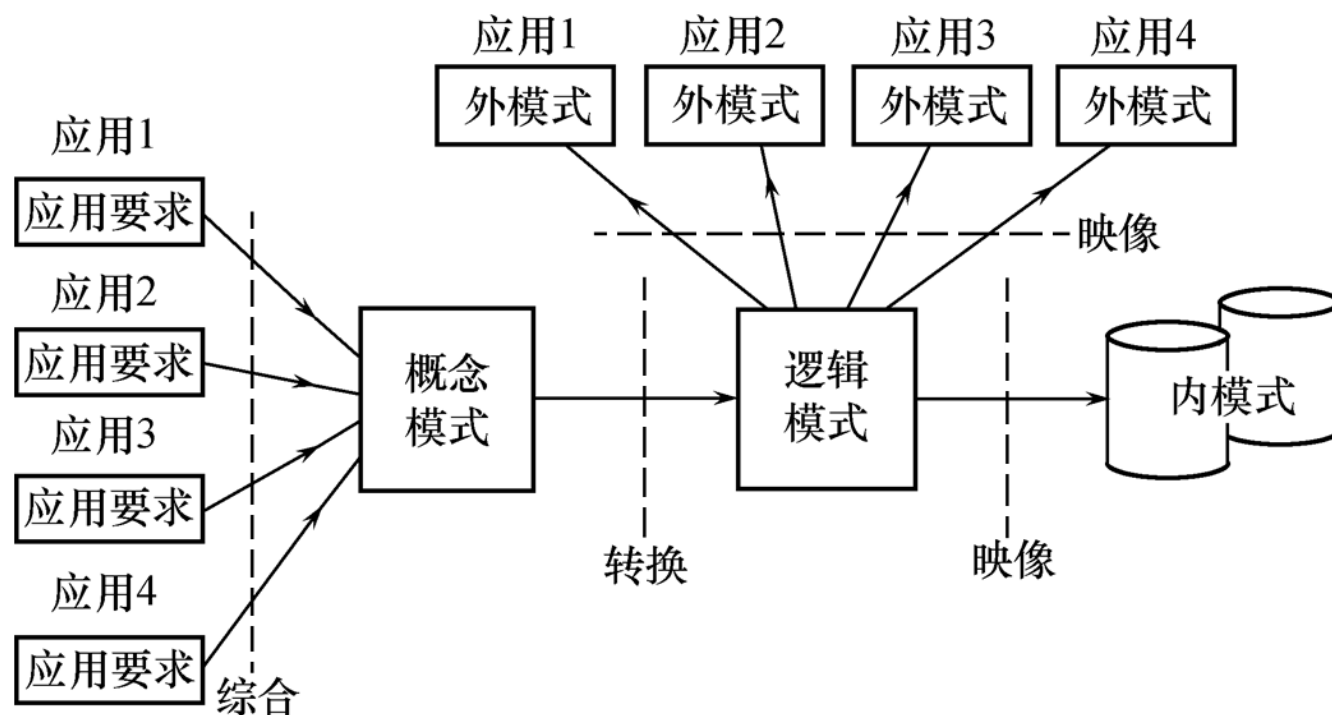
### 7.1.2 数据库设计方法

### 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

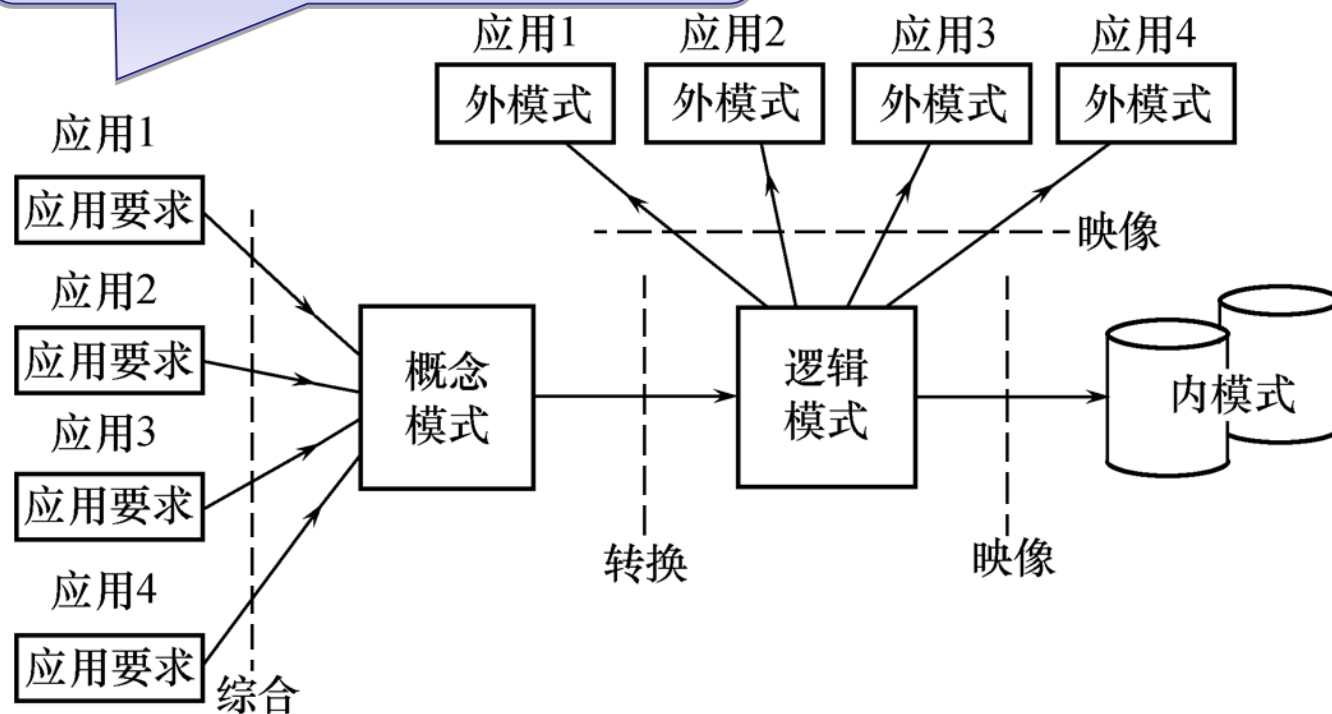


数据库的各级模式

# 数据库设计过程中的各级模式（续）

需求分析阶段：  
综合各个用户的应用需求

成的数据库各级模式



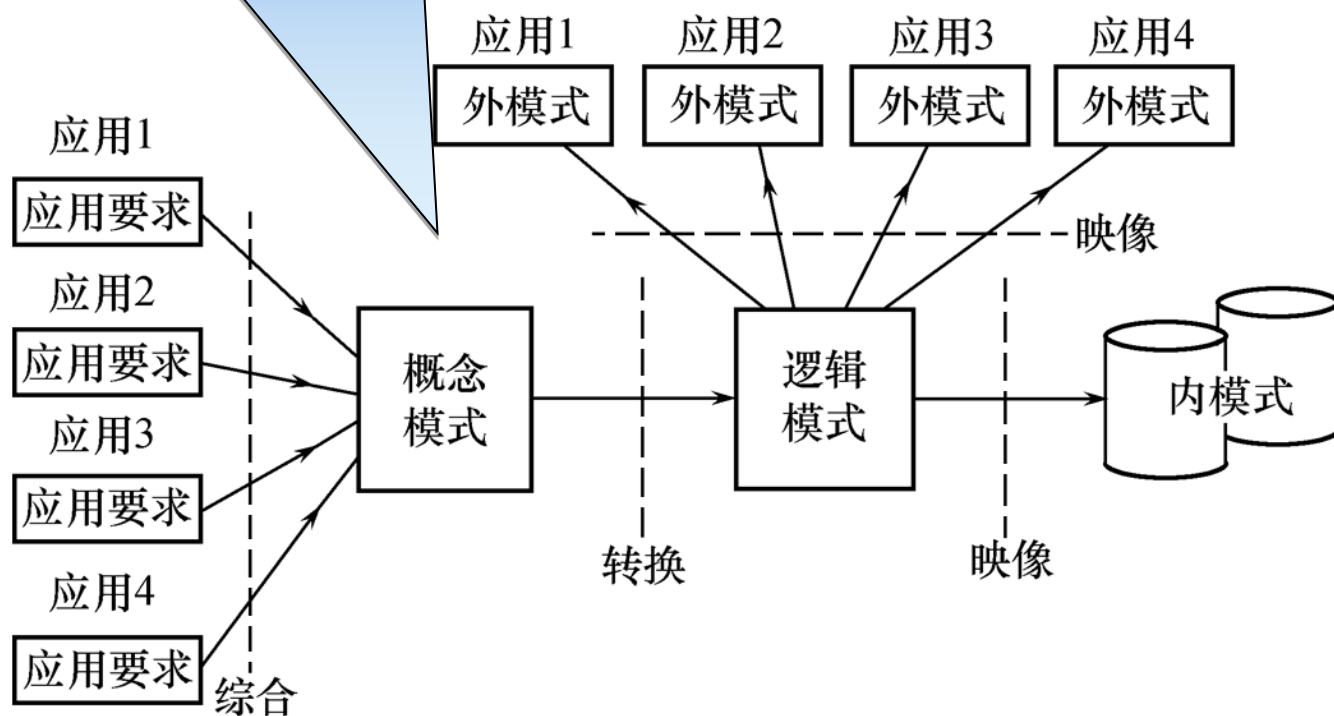
数据库的各级模式

# 数据库的各级模式 (续)

概念设计阶段:

形成独立于机器特点, 独立于各个数据库管理系统产品的**概念模式 (E-R图)**

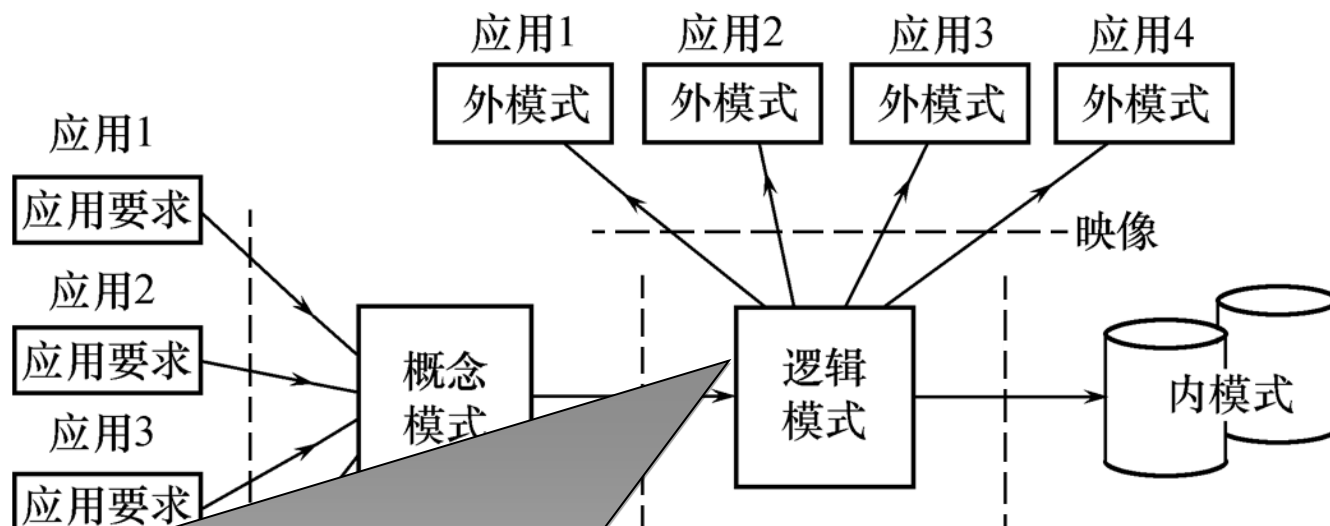
数据库各级模式



数据库的各级模式

# 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：

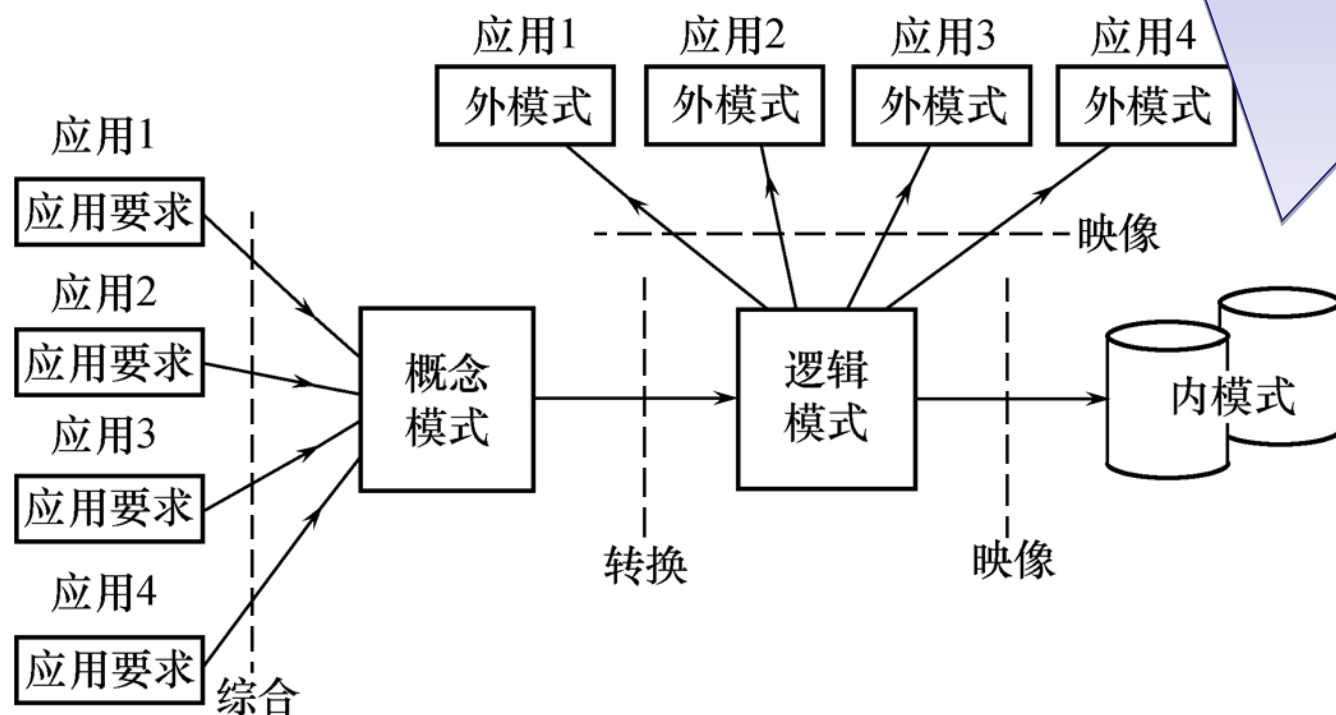
1. 首先将E-R图转换成具体的数据库产品支持的数据模型，如关系模型，形成数据库**逻辑模式**
2. 然后根据用户处理的要求、安全性的考虑，在基本表的基础上再建立必要的视图（**View**），形成数据的**外模式**

# 数据库设计过程和数据库各级模式 (续)

## ● 数据库设计不

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



数据库的各级模式

# 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

## 7.2 需求分析

---

### 7.2.1 需求分析的任务

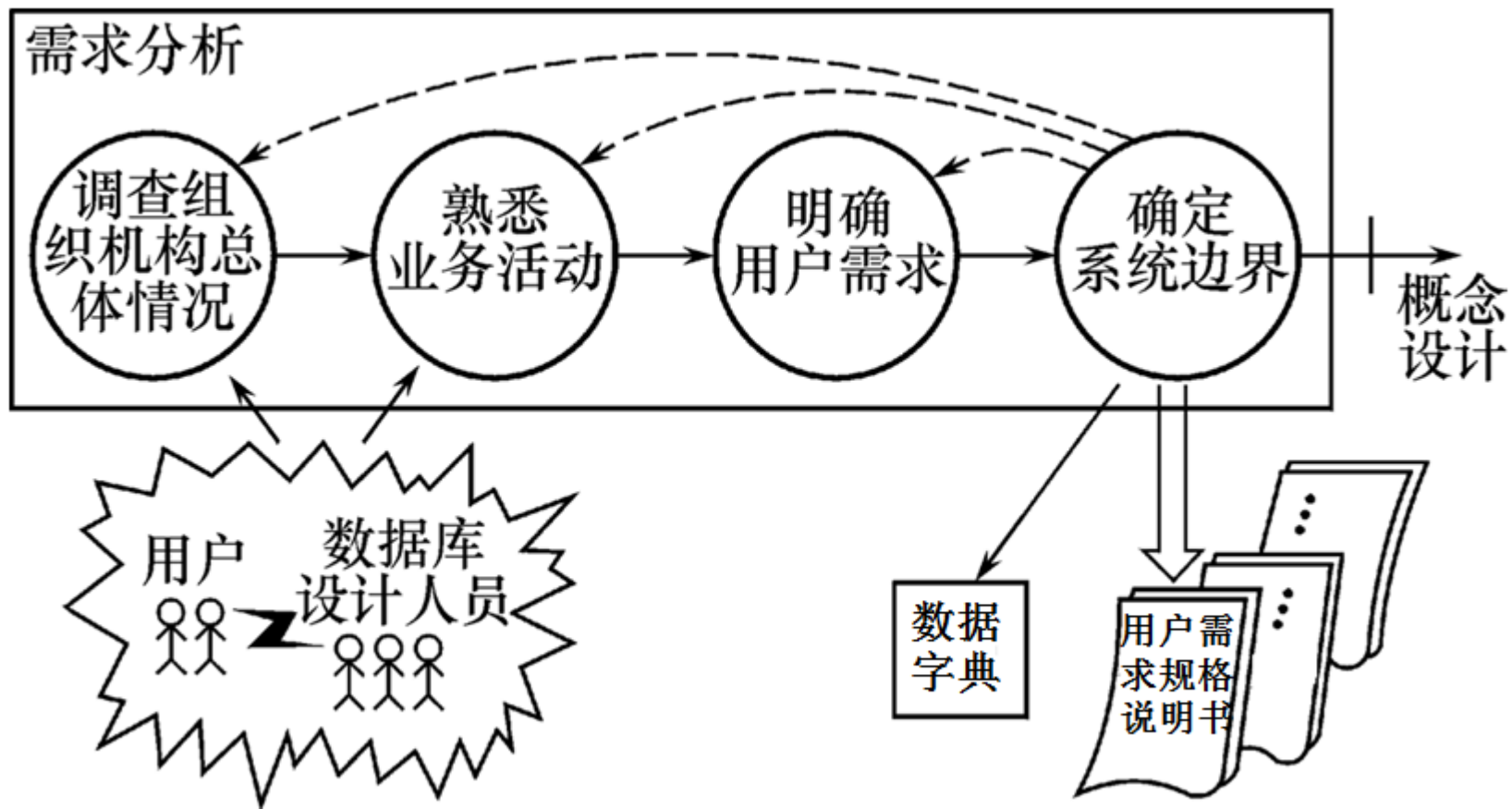
### 7.2.2 需求分析的方法

### 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

# 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

- 数据字典是关于数据库中数据的描述，即元数据，不是数据本身
- 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：

和关系数据库管理系统中数据字典（P89）的区别和联系

# 数据字典（续）

- 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}

- “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
- 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。
- 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}}

### 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

## 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

## 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



## 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： .....

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

数据流去向： 批准

组成： .....

平均流量： .....

高峰期流量： .....

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： .....

流出数据流： .....

组成： .....

数据量： 每年3000张

存取方式： 随机存取

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3平方米。安排新生宿舍其处理时间应不超过15分钟。

# 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信息结构（即概念模型）的过程就是概念结构设计
- 概念模型的特点
  - （1）能真实、充分地反映现实世界，是现实世界的一个真实模型。
  - （2）易于理解，从而可以用它和不熟悉计算机的用户交换意见。
  - （3）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。
  - （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

## 7.3.2 E-R模型

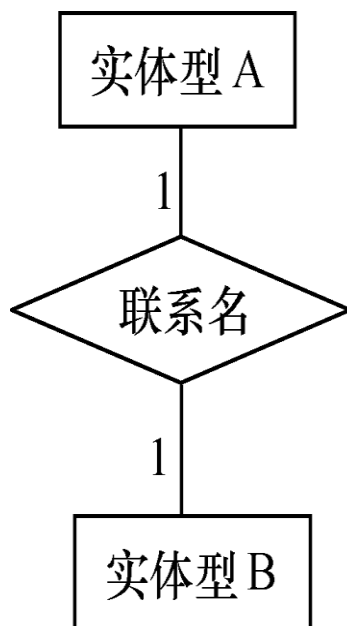
- 实体之间的联系

① 一对一联系 ( $1 : 1$ )

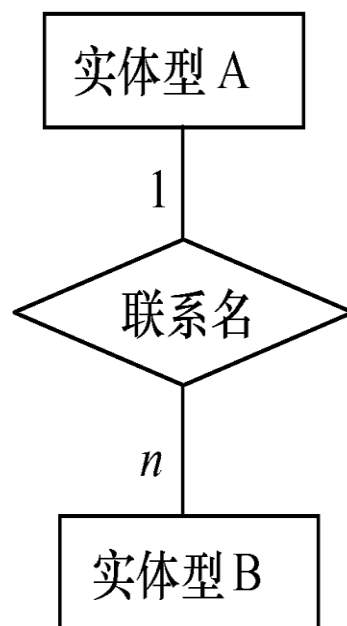
② 一对多联系 ( $1 : n$ )

③ 多对多联系 ( $m : n$ )

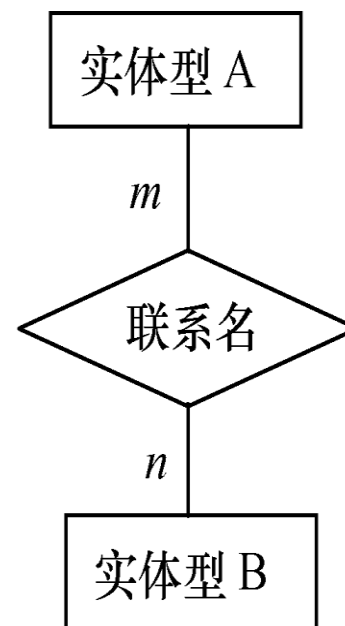
## E-R模型（续）



(a) 1:1 联系



(b) 1:n 联系



(c) m:n 联系

图7.6 两个实体型之间的三类联系

# E-R模型（续）

## ● E-R图

➤ 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

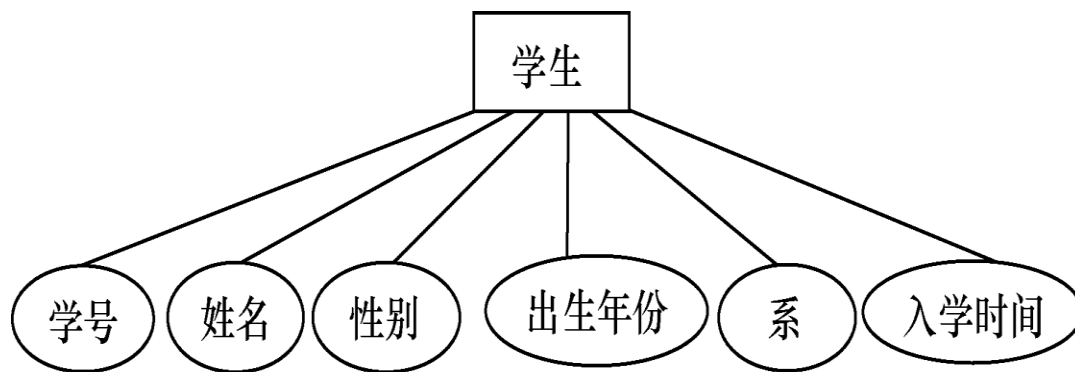


图7.9 学生实体及属性

## E-R模型（续）

- ✓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

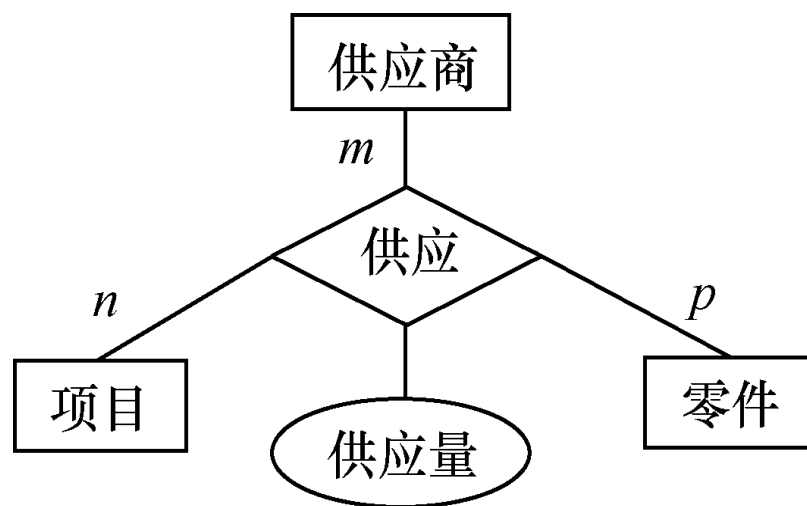


图7.10 联系的属性

## 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。

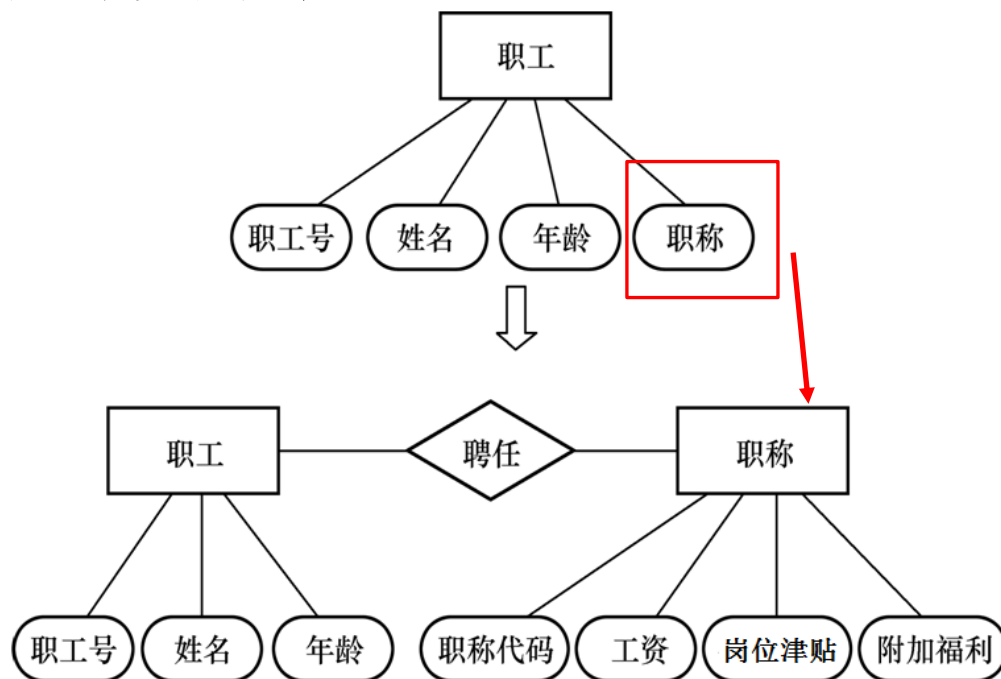
(2) **属性不能与其他实体具有联系**，即E-R图中所表示的联系是实体之间的联系。



# 概念结构设计（续）

[例1] 职工是一个实体，职工号、姓名、年龄是职工的属性。

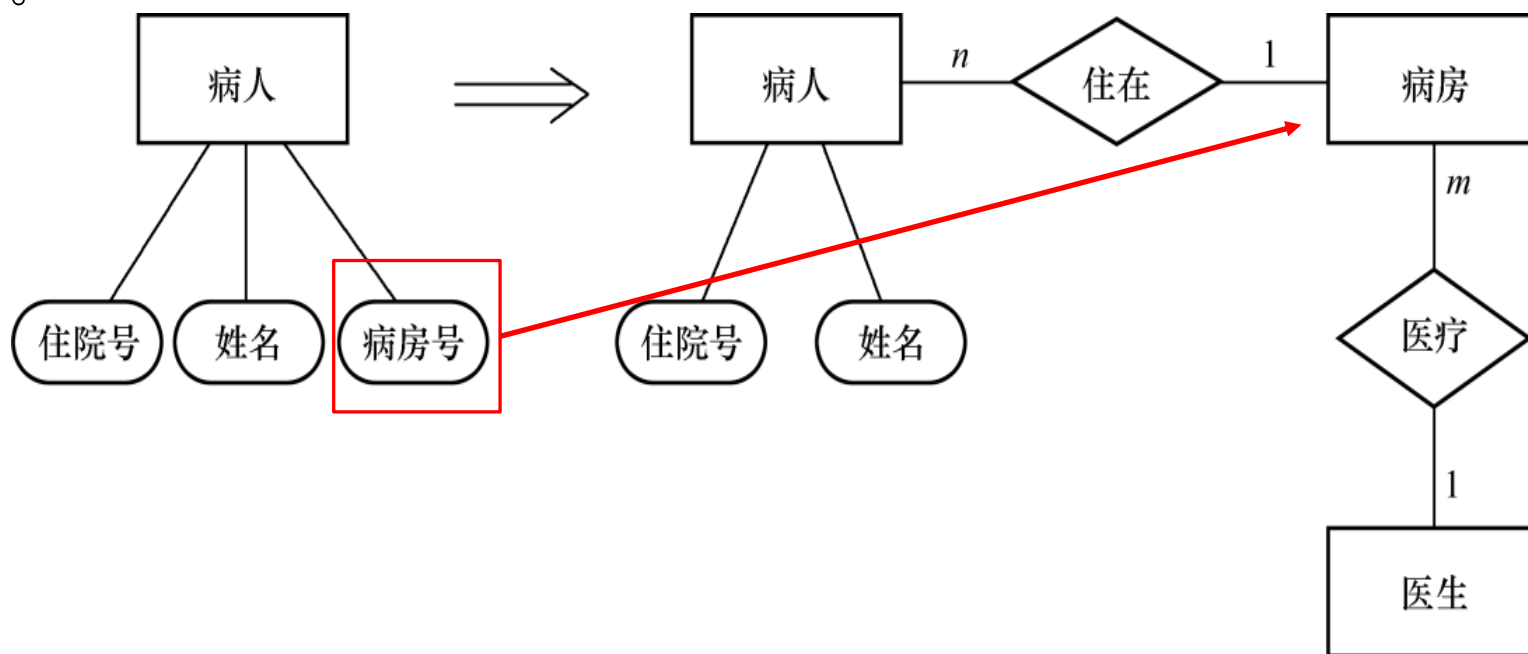
- 职称如果没有与工资、福利挂钩，根据**准则（1）**可以作为职工实体的属性
- 如果不同的职称有不同的工资、住房标准和不同的附加福利，则职称作为一个实体更恰当



## 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；

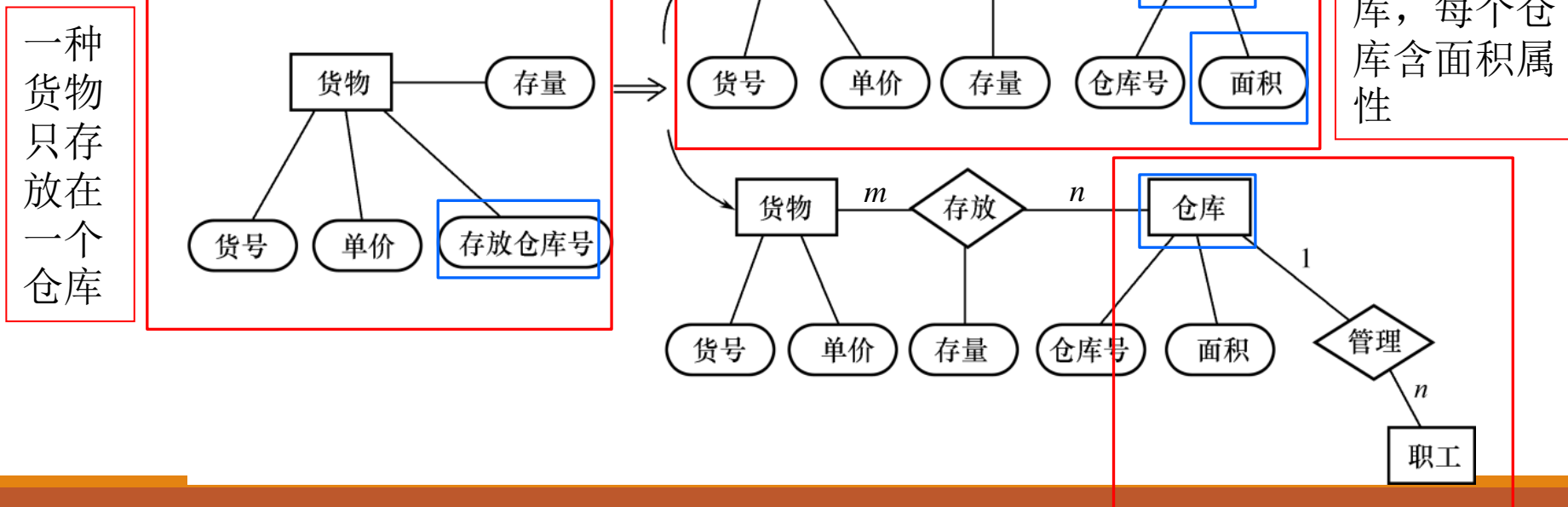
如果病房还要与医生实体发生联系，即一个医生负责几个病房的病人的医疗工作，则根据**准则（2）** 病房应作为一个实体。



# 概念结构设计（续）

[例3] 如果一种货物只存放在一个仓库，那么就可以把存放货物的仓库的仓库号作为描述货物存放地点的属性。

如果一种货物可以存放在多个仓库中，或者仓库本身又用面积作为属性，或者仓库与职工发生管理上的联系，那么就应把仓库作为一个实体。



# 概念结构设计（续）

## ●[例7.1] 销售管理子系统E-R图的设计。

➤该子系统的主要功能是：

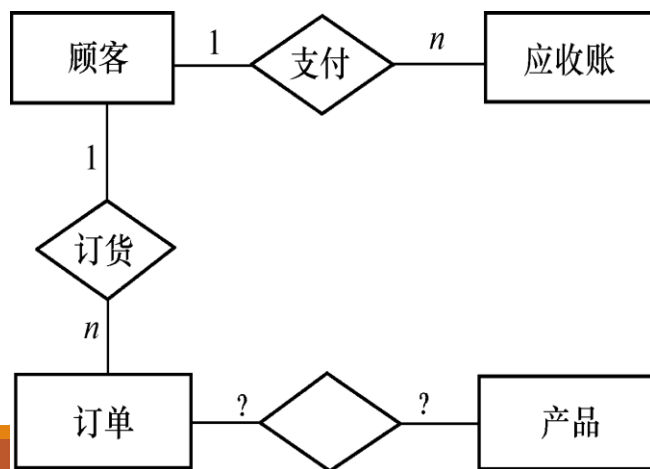
- ✓处理顾客和销售员送来的订单
- ✓工厂是根据订货安排生产的
- ✓交出货物同时开出发票
- ✓收到顾客付款后，根据发票存根和信贷情况进行应收款处理

围绕“订单”和“应收账款”  订单、顾客、应收账目

# 概念结构设计（续）

- 参照需求分析和数据字典中的详尽描述，遵循前面给出的两个准则，进行了如下调整：

（1）每张订单由订单号、若干头信息和订单细节组成。订单细节又有订货的零件号、数量等来描述。按照**准则（1）与（2）**，**订单细节就不能作订单的属性处理而应该上升为实体。**一张订单可以订若干产品，所以订单与订单细节两个实体之间是 $1:n$ 的联系。



## 概念结构设计（续）

- （2）原订单和产品的联系实际上是订单细节和产品的联系。每条订货细节对应一个产品描述，订单处理时从中获得当前单价、产品重量等信息。
- （3）工厂对大宗订货给予优惠。每种产品都规定了不同订货数量的折扣，应增加一个“折扣规则”实体存放这些信息，而不应把它们放在产品实体中。

## 概念结构设计（续）

- 最后得到销售管理子系统E-R图如图7.23所示。

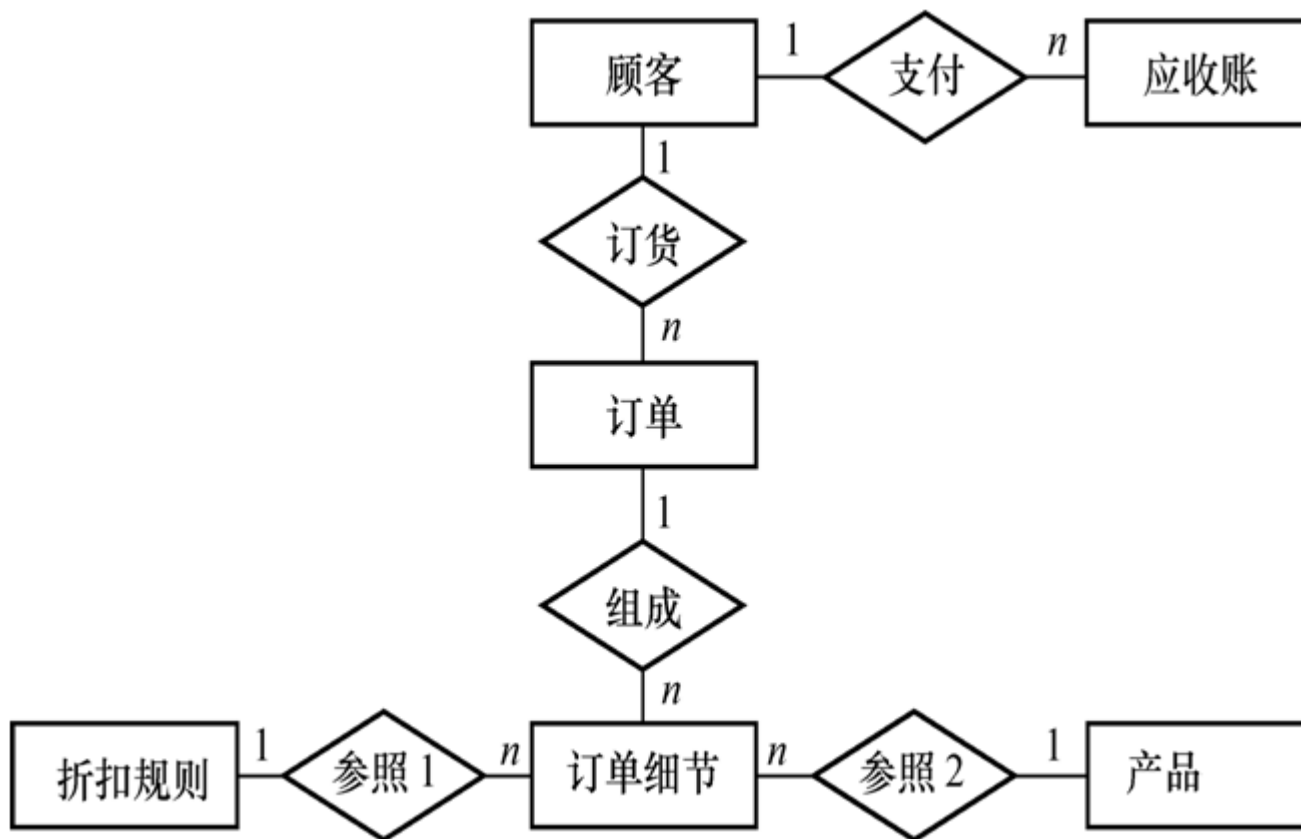


图7.23 销售管理子系统的E-R图

# 概念结构设计（续）

- 对每个实体定义的属性如下：

- 顾客：{顾客号，顾客名，地址，电话，信贷状况，账目余额}
- 订单：{订单号，顾客号，订货项数，订货日期，交货日期，工种号，生产地点}
- 订单细则：{订单号，细则号，零件号，订货数，金额}
- 应收账款：{顾客号，订单号，发票号，应收金额，支付日期，支付金额，当前余额，货款限额}
- 产品：{产品号，产品名，单价，重量}
- 折扣规则：{产品号，订货量，折扣}

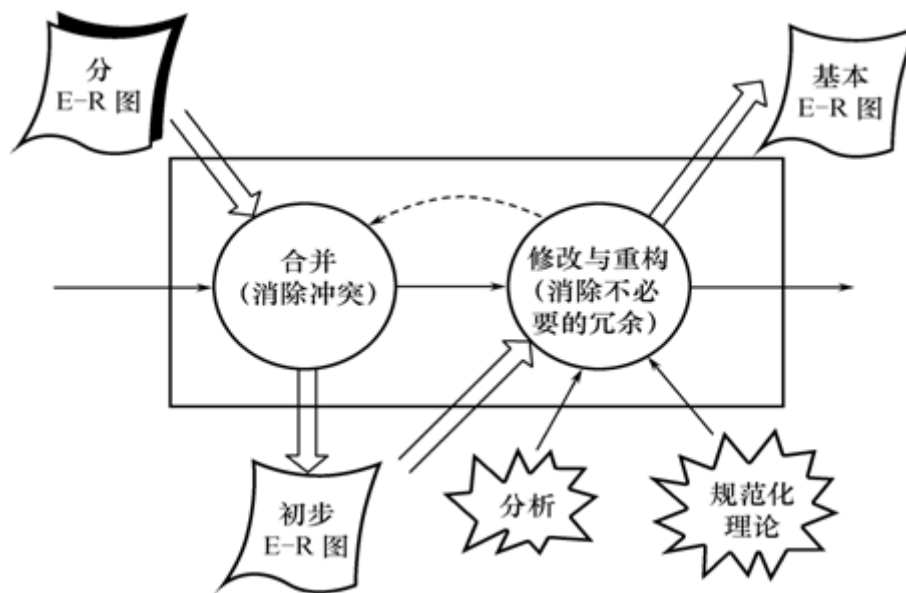


# 概念结构设计（续）

## 2. E-R图的集成

### ➤ 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、取值范围或取值集合不同。
  - 例如零件号，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的年龄，而另一些部门用整数表示职工的年龄。
- 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。
- **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，财务科称为项目，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、联系一级上
  - 也可能发生在属性一级上（**常见现象**）
  - 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”

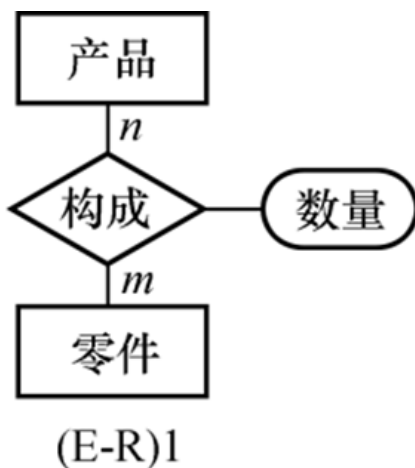
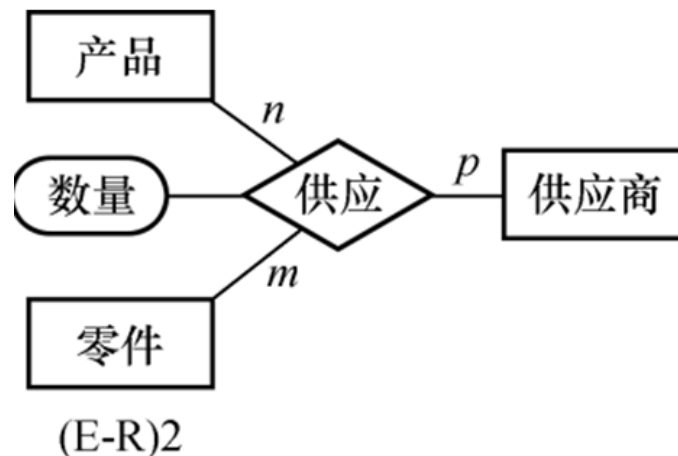
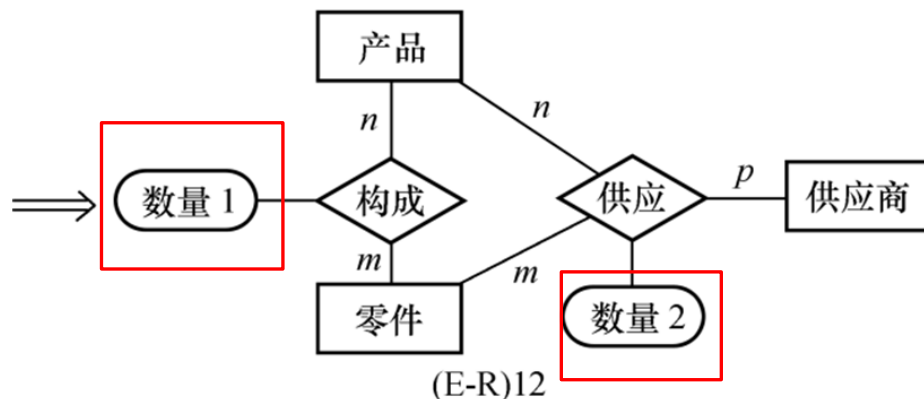


图7.25(b)中产品、零件与供应商三者之间还存在多对多的联系“供应”



合并两个E-R图，  
如图7.25(c)



# 概念结构设计（续）

（2）消除不必要的冗余，设计基本E-R图

- 所谓冗余的数据是指可由基本数据导出的数据，冗余的联系是指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。

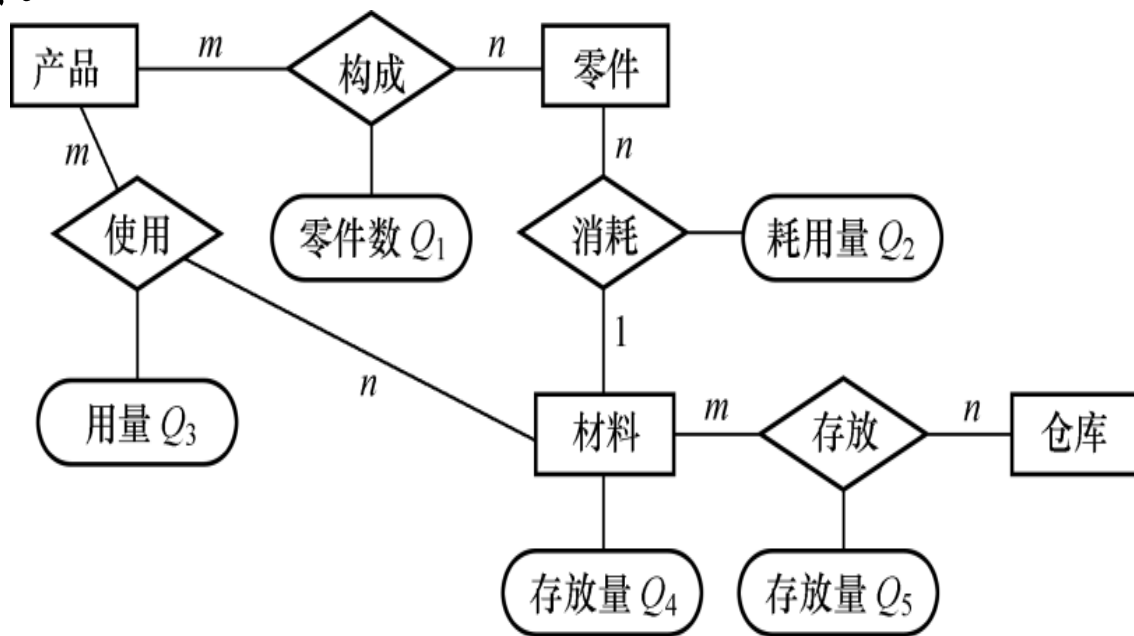


图7.26 消除冗余

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、多对多的联系可以用实体码之间的函数依赖来表示。于是有函数依赖集 $F_L$ ，如图7.27中：
- ✓ 部门和职工之间一对多的联系可表示为职工号 $\rightarrow$ 部门号
- ✓ 职工和产品之间多对多的联系可表示为（职工号，产品号） $\rightarrow$ 工作天数等。

## 概念结构设计（续）

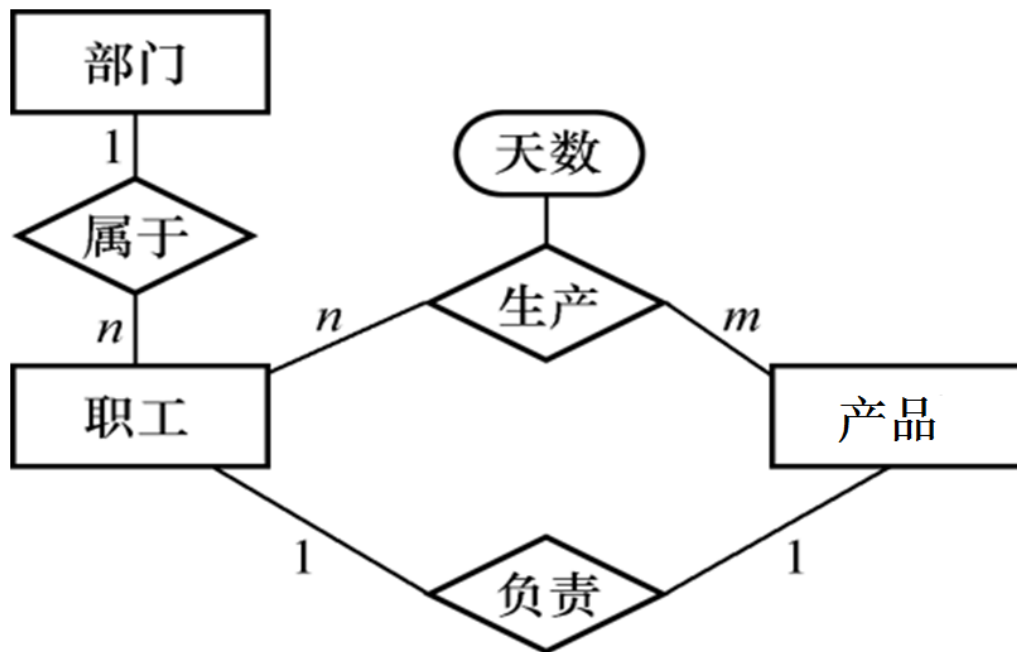


图7.27 劳动人事管理的分E-R图

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，确定是否是冗余的联系，若是，就把它去掉。

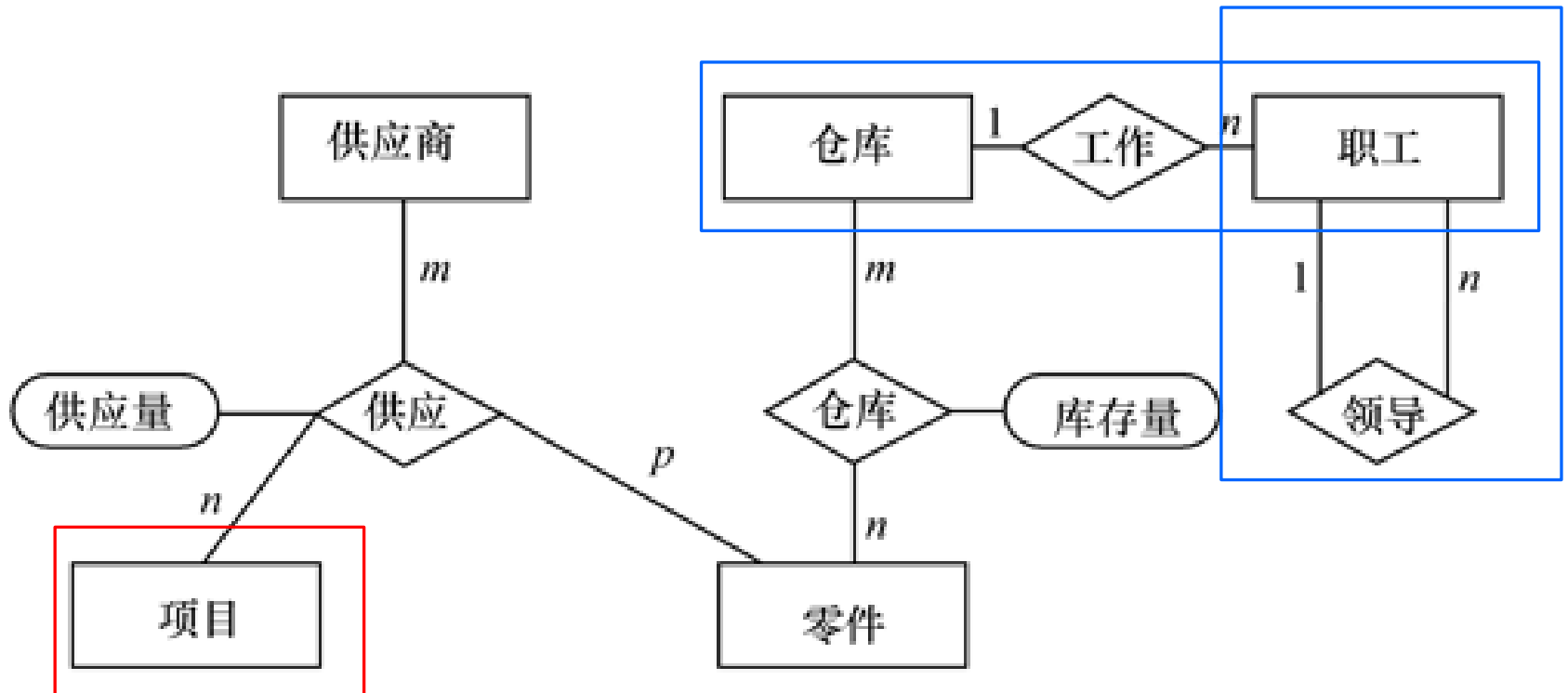
# 概念结构设计（续）

➤ 由于规范化理论受到泛关系假设的限制，应注意下面两个问题

- ✓ 冗余的联系一定在D中，而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。



(b) 实体及其联系图

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。

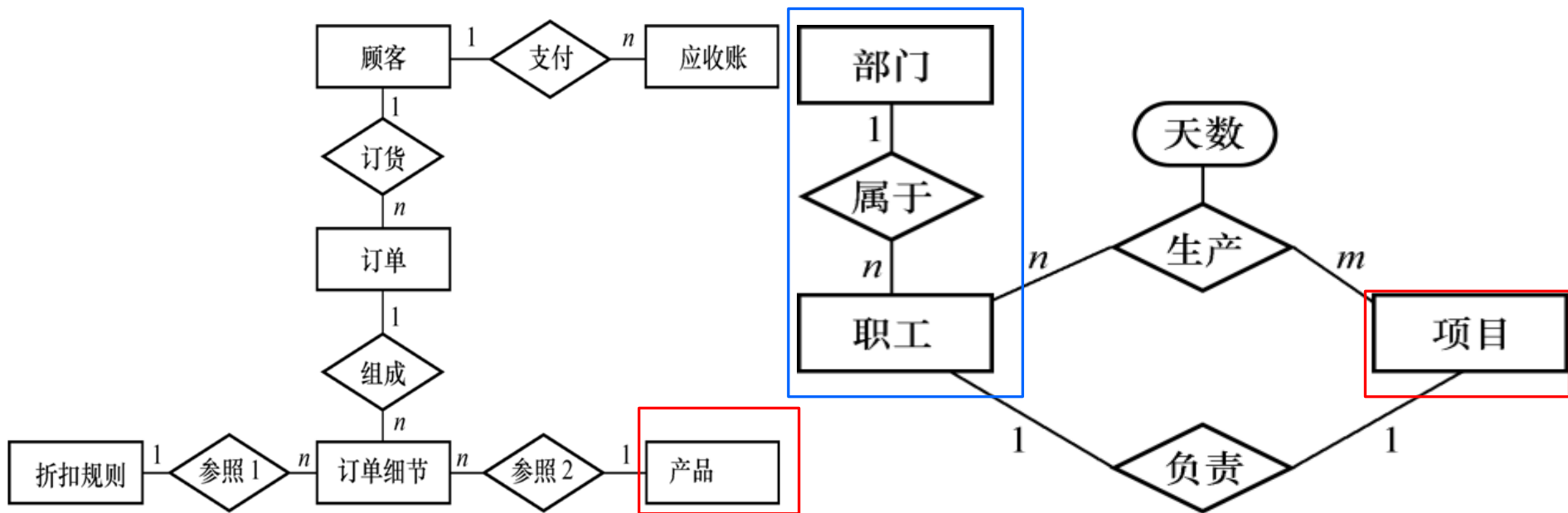


图7.23 销售管理子系统的E-R图

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。

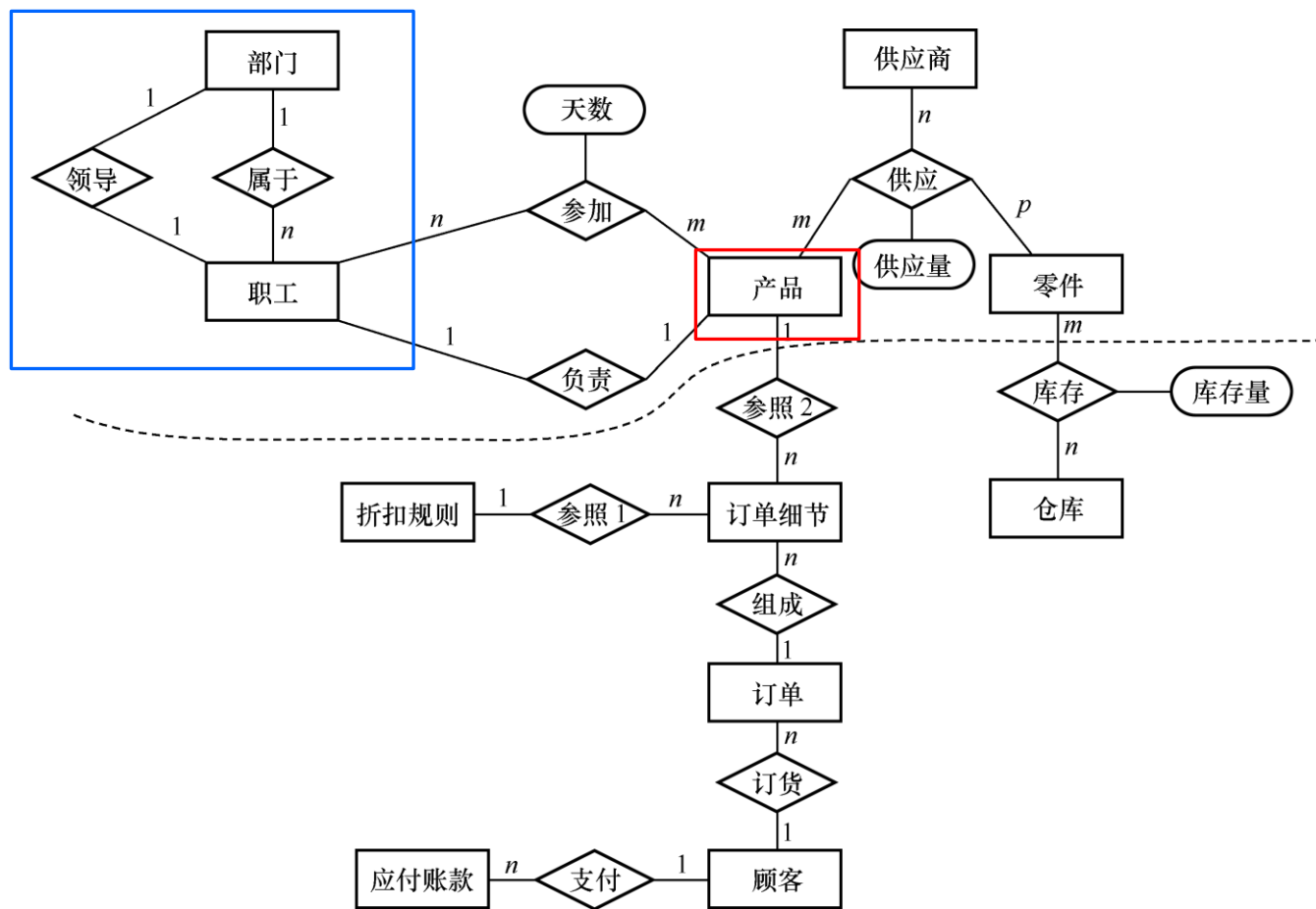
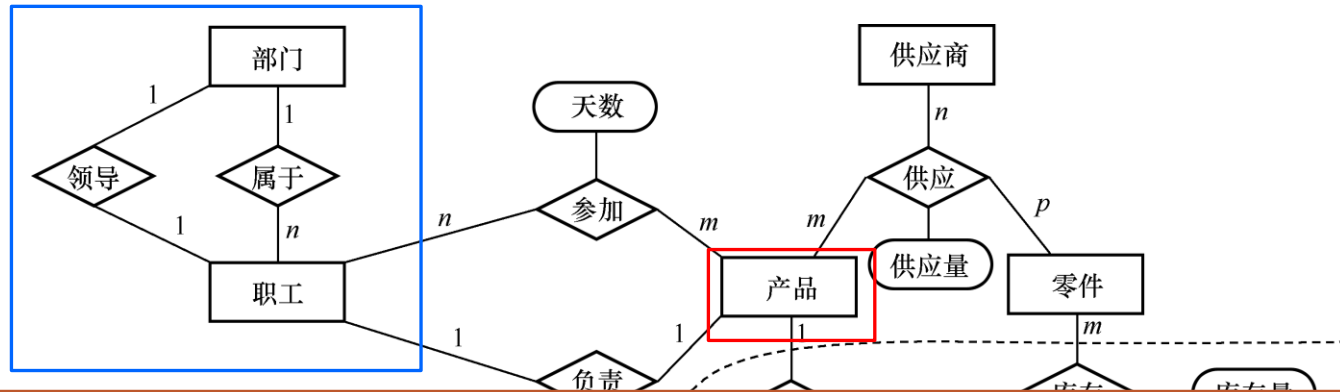


图7.28 某工厂管理信息系统的基本E-R图

# 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

## 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。

### ① 转换为一个独立的关系模式

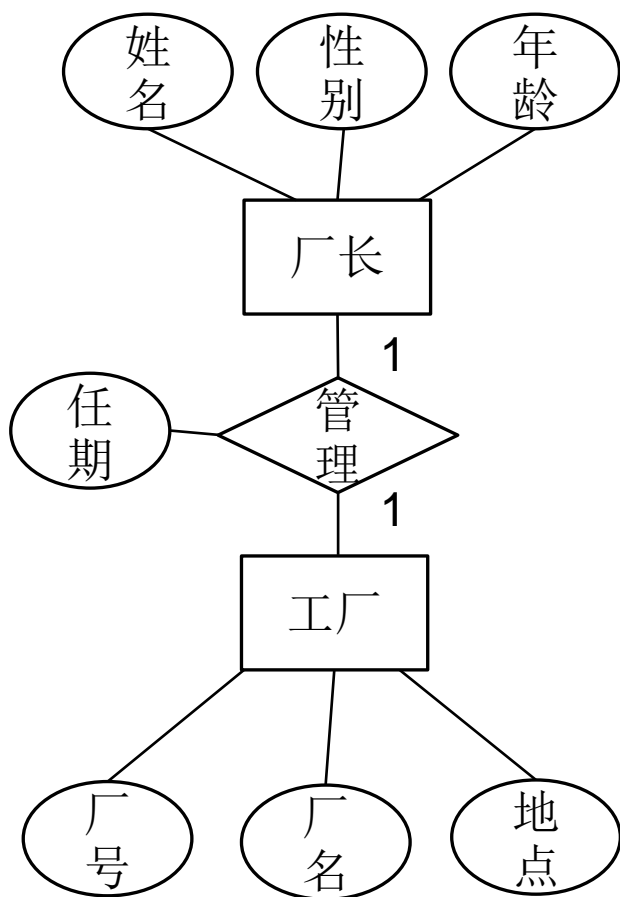
- 关系的属性：与该联系相连的各实体的码，联系本身的属性。
- 关系的候选码：每个实体的码均是该关系的候选码

# 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



合并:

工厂 (厂号, 厂名, 地点, 姓名, 任期)

厂长 (姓名, 性别, 年龄)

或

工厂 (厂号, 厂名, 地点)

厂长 (姓名, 性别, 年龄, 厂号, 任期)

独立:

工厂 (厂号, 厂名, 地点)

厂长 (姓名, 性别, 年龄)

管理 (厂号, 姓名, 任期) 或 (厂号, 姓名, 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。

## ①转换为一个独立的关系模式

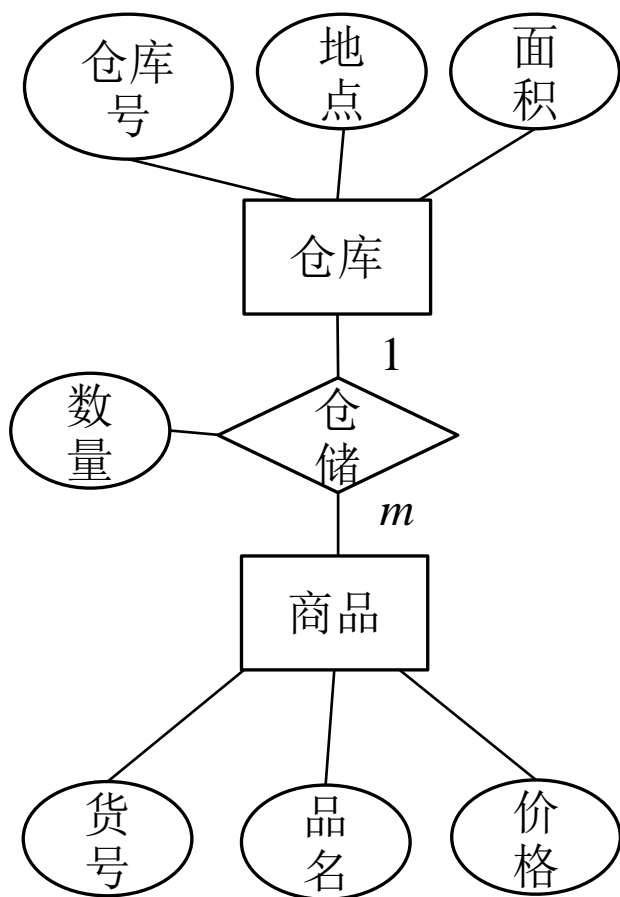
- 关系的属性：与该联系相连的各实体的码以及联系本身的属性
- 关系的码： $n$ 端实体的码

# 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
- 可以减少系统中的关系个数，一般情况下更倾向于采用这种方法



合并:

仓库 (仓库号, 地点, 面积)

商品 (货号, 品名, 价格, 仓库号, 数量)

独立:

仓库 (仓库号, 地点, 面积)

商品 (货号, 品名, 价格)

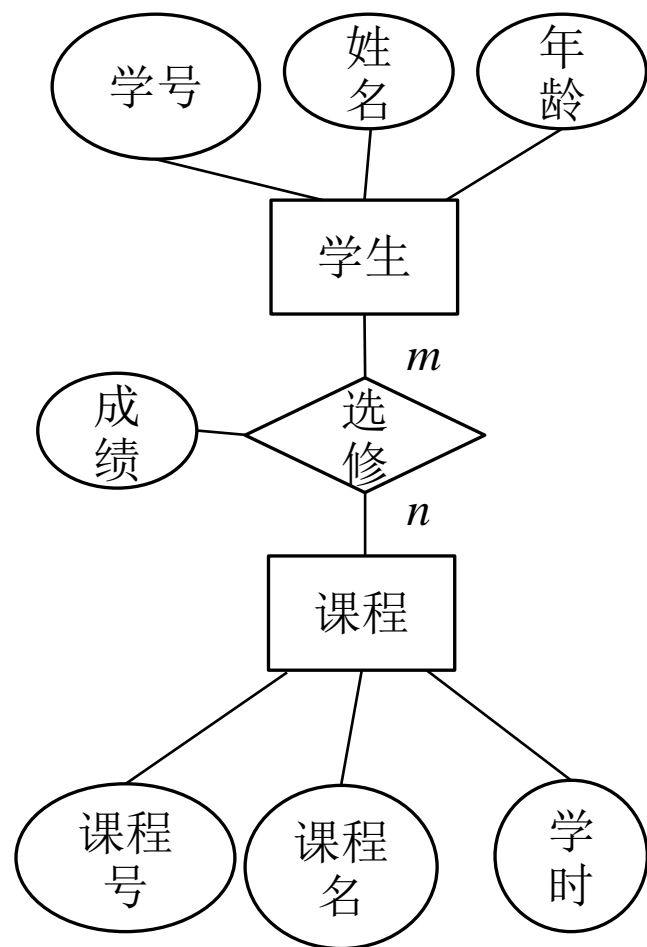
仓储 (货号, 仓库号, 数量)

# 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

必须对联系单独建立一个关系。

- 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



独立:

学生 (学号, 姓名, 年龄)

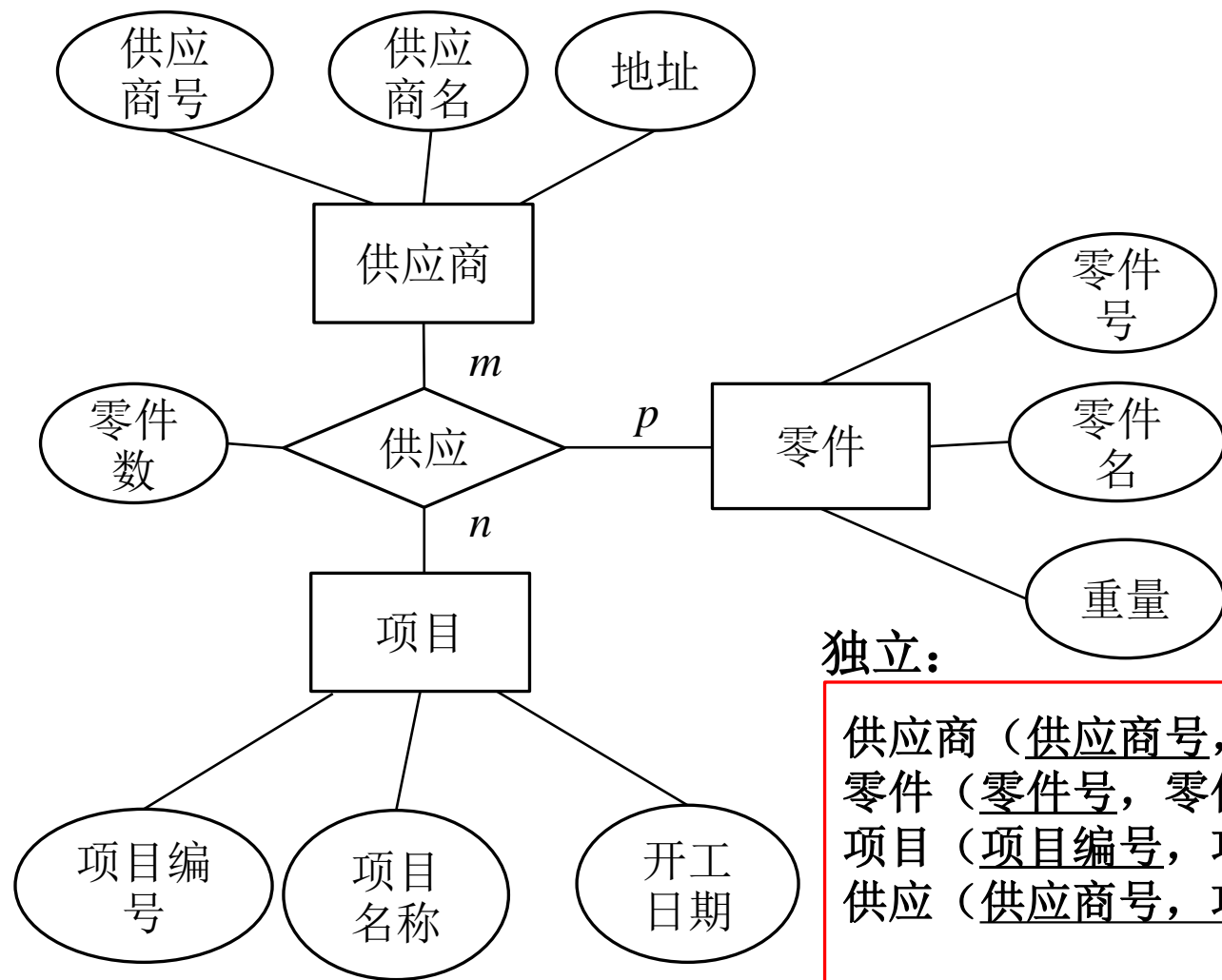
课程 (课程号, 课程名, 学时)

选修 (学号, 课程号, 成绩)

## E-R图向关系模型的转换（续）

（4）三个或三个以上实体间的一个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



独立：

供应商（供应商号，供应商名，地址）

零件（零件号，零件名，重量）

项目（项目编号，项目名称，开工日期）

供应（供应商号，项目编号，零件号，零件数）

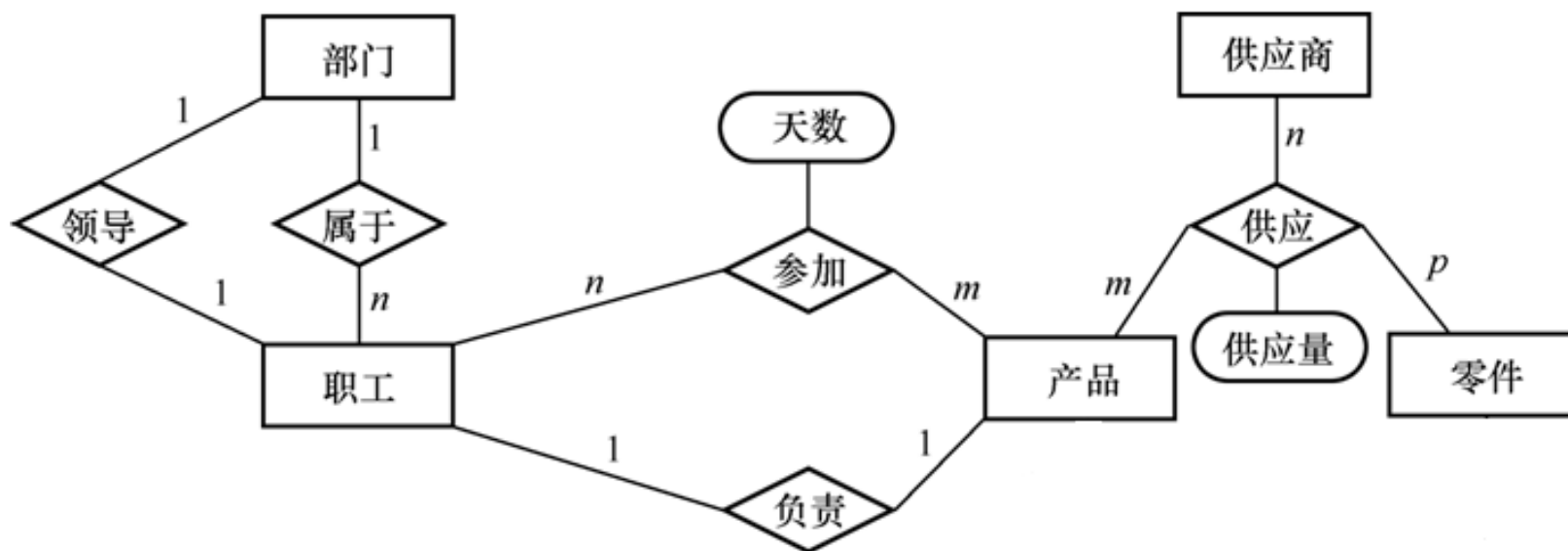
# E-R图向关系模型的转换（续）

（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（可能同名也可能不同名）
  - 适当调整属性的次序



## E-R图向关系模型的转换（续）



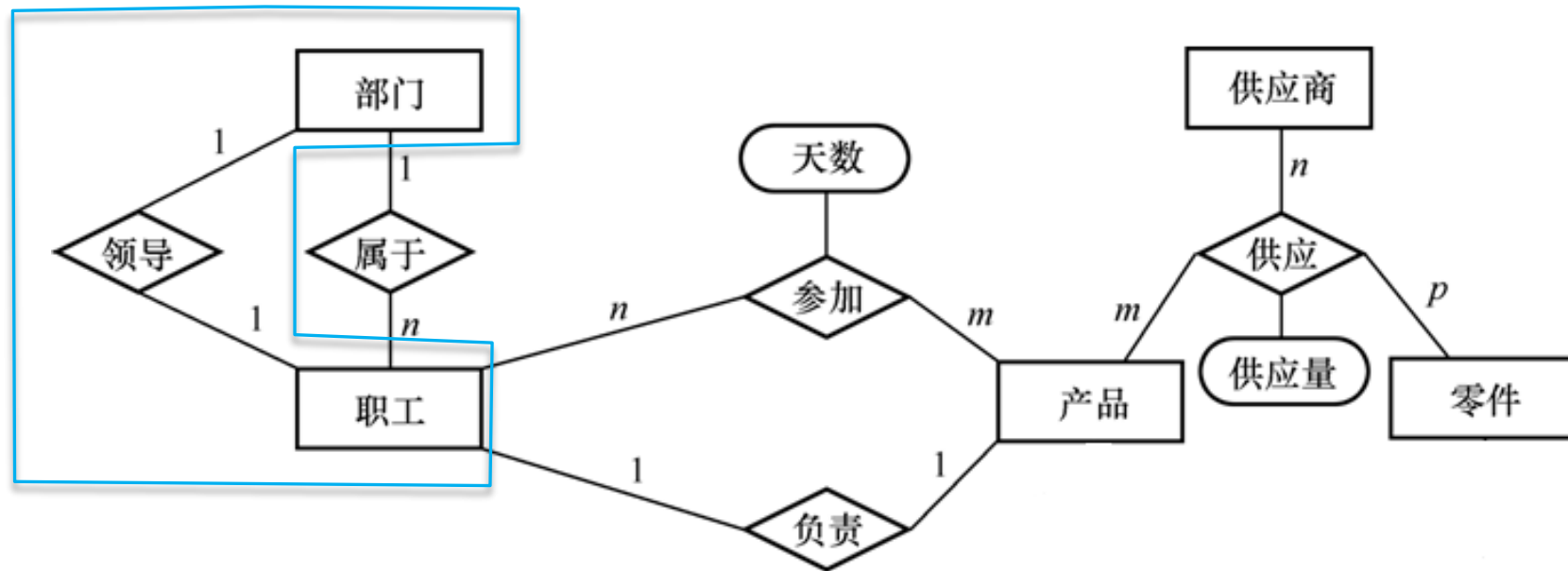
按照上述多条转换规则，将图7.28中虚线上部的E-R图转换为关系模型。

关系的码用下横线标出。

# 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，...）
- 供应商（供应商号，姓名，...）
- 零件（零件号，零件名，...）
- 职工工作（职工号，产品号，工作天数，...）
- 供应（产品号，供应商号，零件号，供应量）

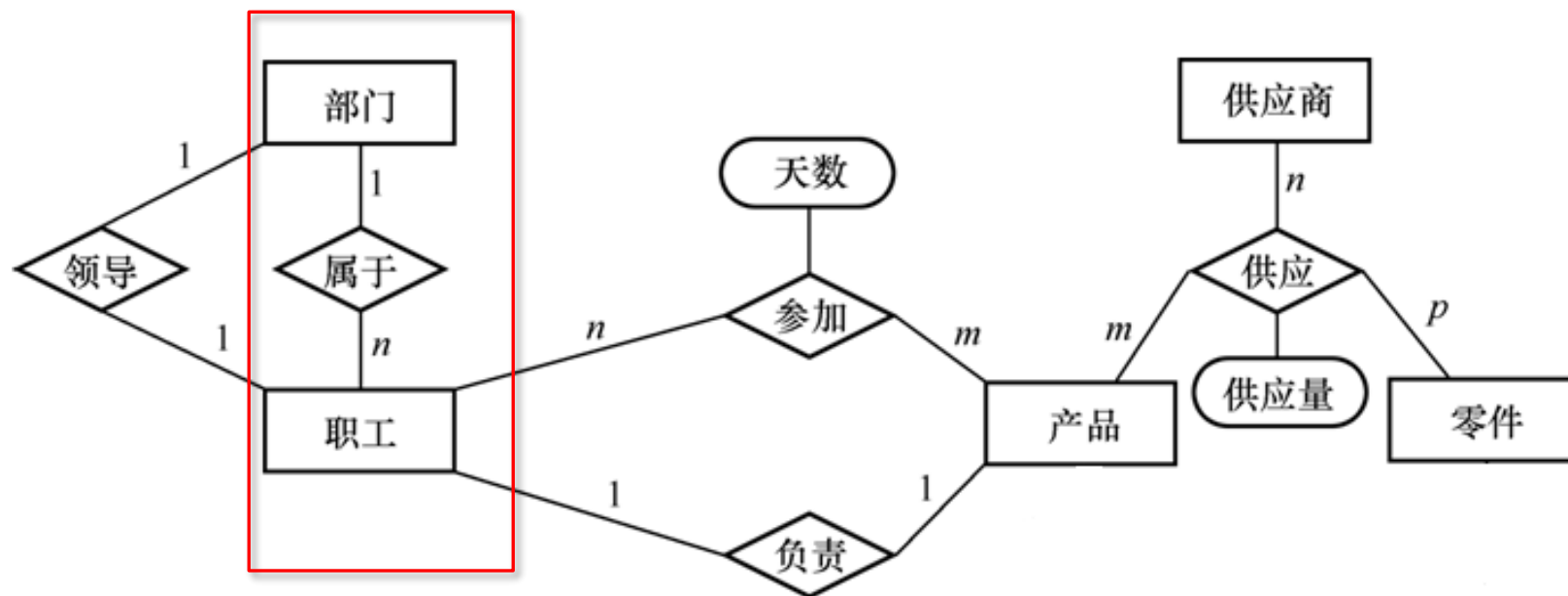
## E-R图向关系模型的转换（续）



# 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**，...）
- 供应商（供应商号，姓名，...）
- 零件（零件号，零件名，...）
- 职工工作（职工号，产品号，**工作天数**，...）
- 供应（产品号，供应商号，零件号，**供应量**）

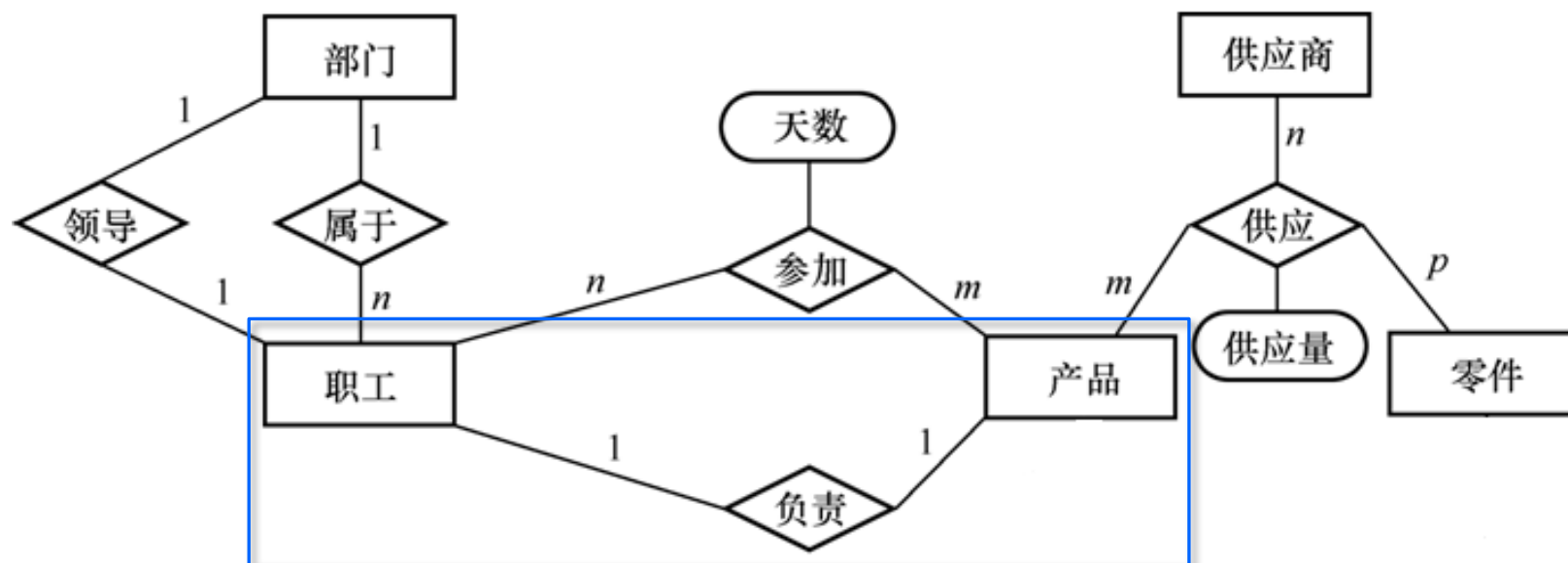
## E-R图向关系模型的转换（续）



# 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，...）
- 供应商（供应商号，姓名，...）
- 零件（零件号，零件名，...）
- 职工工作（职工号，产品号，工作天数，...）
- 供应（产品号，供应商号，零件号，供应量）

## E-R图向关系模型的转换（续）

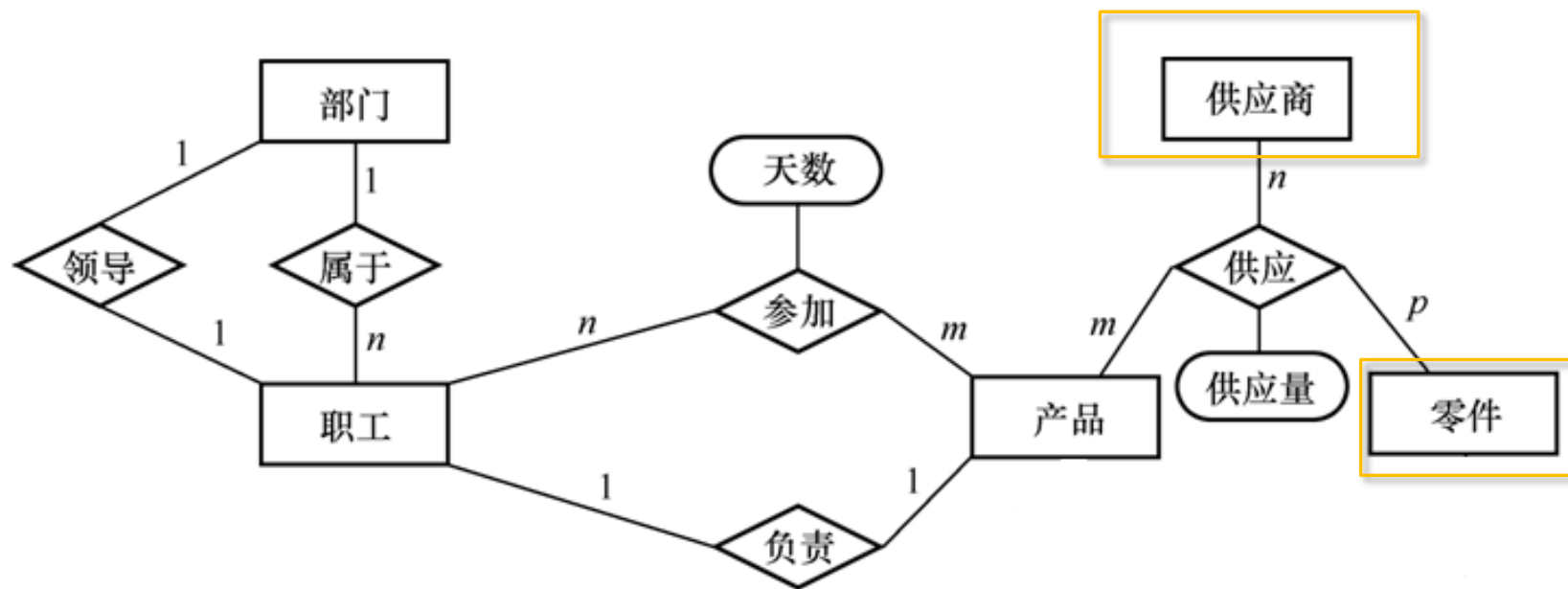


# E-R图向关系模型的转换（续）

- 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，...）
- 供应商（供应商号，姓名，...）
- 零件（零件号，零件名，...）
- 职工工作（职工号，产品号，工作天数，...）
- 供应（产品号，供应商号，零件号，供应量）



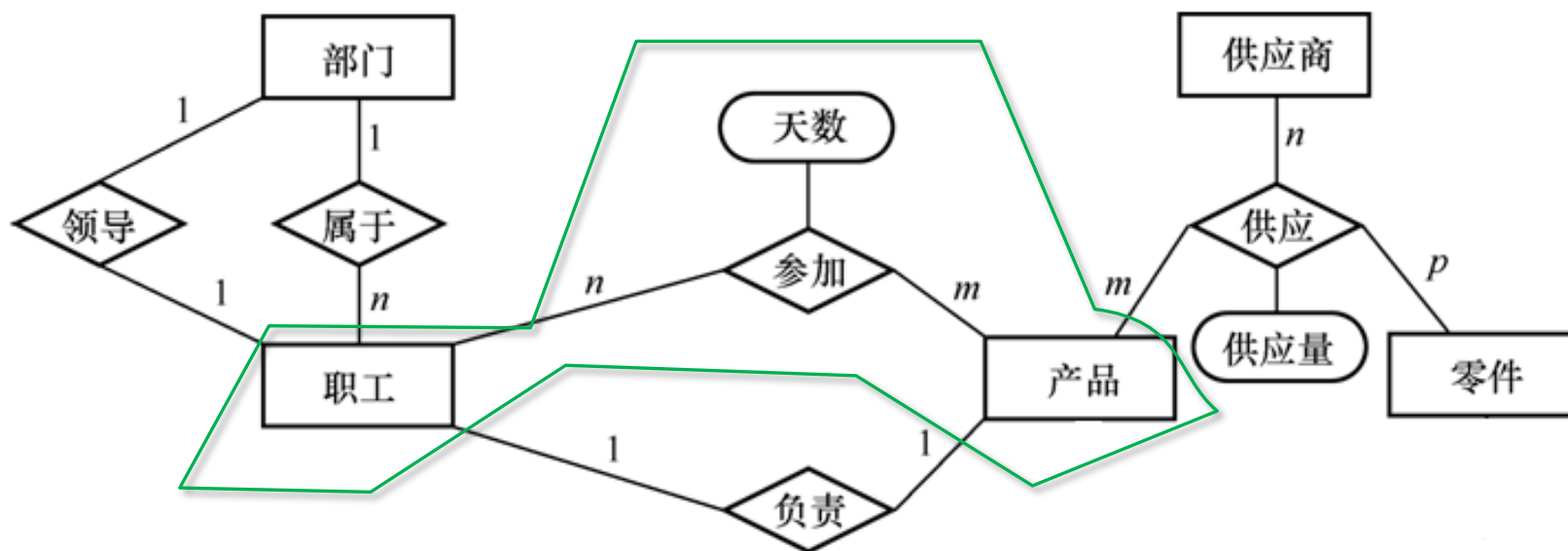
## E-R图向关系模型的转换（续）



# 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，...）
- 供应商（供应商号，姓名，...）
- 零件（零件号，零件名，...）
- 职工工作（职工号，产品号，工作天数，...）
- 供应（产品号，供应商号，零件号，供应量）

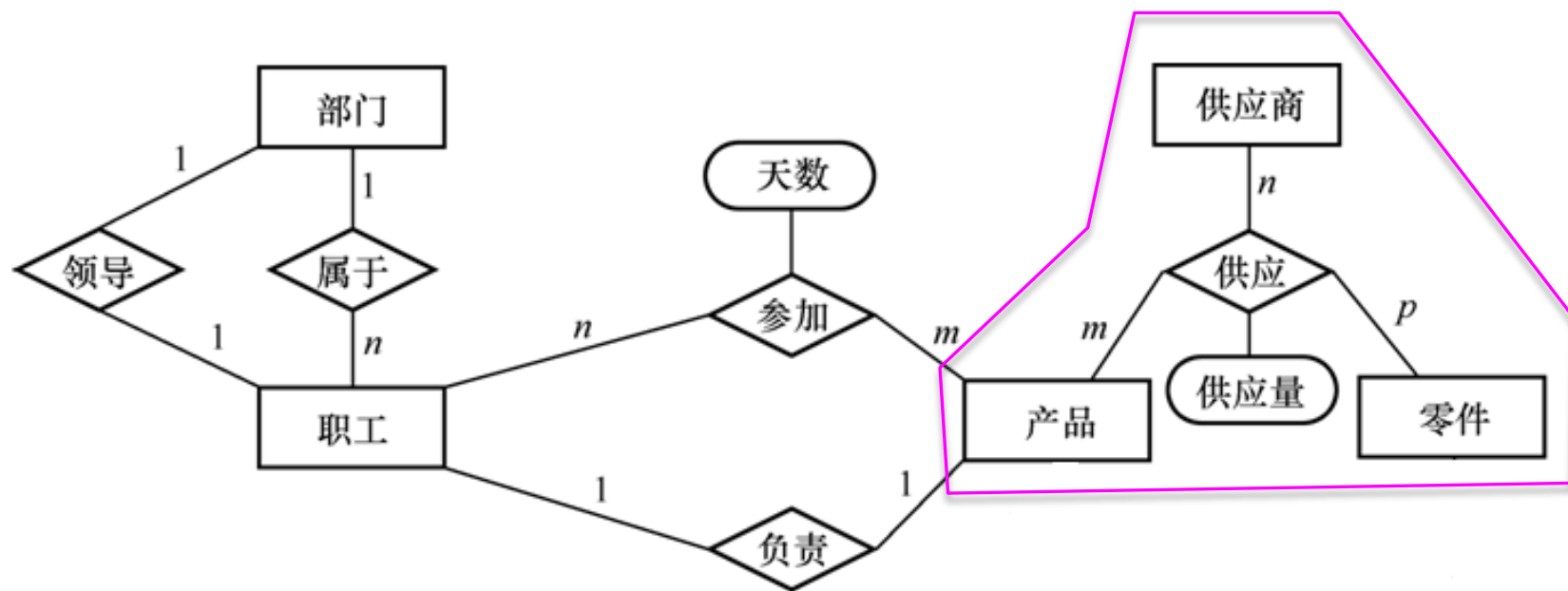
## E-R图向关系模型的转换（续）



# 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，...）
- 供应商（供应商号，姓名，...）
- 零件（零件号，零件名，...）
- 职工工作（职工号，产品号，工作天数，...）
- 供应（产品号，供应商号，零件号，供应量）

## E-R图向关系模型的转换（续）



# 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，...）
- 供应商（供应商号，姓名，...）
- 零件（零件号，零件名，...）
- 职工工作（职工号，产品号，工作天数，...）
- 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。
- 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，这就是数据模型的优化。
- 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(基本)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”原则的，把经常被使用的数据（约20%）水平分解出来，形成一个子关系。
- ✓ 若关系R上具有n个事务，而且多数事务存取数据不相交，则R可分解为少于或等于n个子关系，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上的所有事务的总效率是否得到了提高

## ➤ 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图曾做了消除命名冲突的工作，以使数据库系统中同一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。
- 用视图机制可以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，以保证系统的安全性。

- 假设有关系模式产品（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

- **数据库的物理结构：**数据库在物理设备上的存储结构与存取方法，数据库的物理结构依赖于选定的数据库管理系统。
- **数据库的物理设计：**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。



# 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取方法和存储结构;
- 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。

# 数据库的物理设计（续）

- 关系数据库物理设计的内容

- 为关系模式选择关系数据库管理系统支持的存取方法
- 设计关系、索引等数据库文件的物理存储结构

# 数据库的物理设计（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

# 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

# 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(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如关系模型，形成数据库逻辑模式。
- 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，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