



# 数据库系统概论

## An Introduction to Database System

### 第七章 数据库设计

刘淇

Email: qiliuql@ustc.edu.cn

课程主页：

<http://staff.ustc.edu.cn/~qiliuql/DB2020HF.html>



# 第七章 数据库设计

2

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 数据库的物理设计

7.6 数据库实施和维护

7.7 小结



## 7.3 概念结构设计

3

### 7.3.1 概念结构

### 7.3.2 概念结构设计的方法与步骤

### 7.3.3 数据抽象与局部视图设计

### 7.3.4 视图的集成



## 7.3.1 概念结构

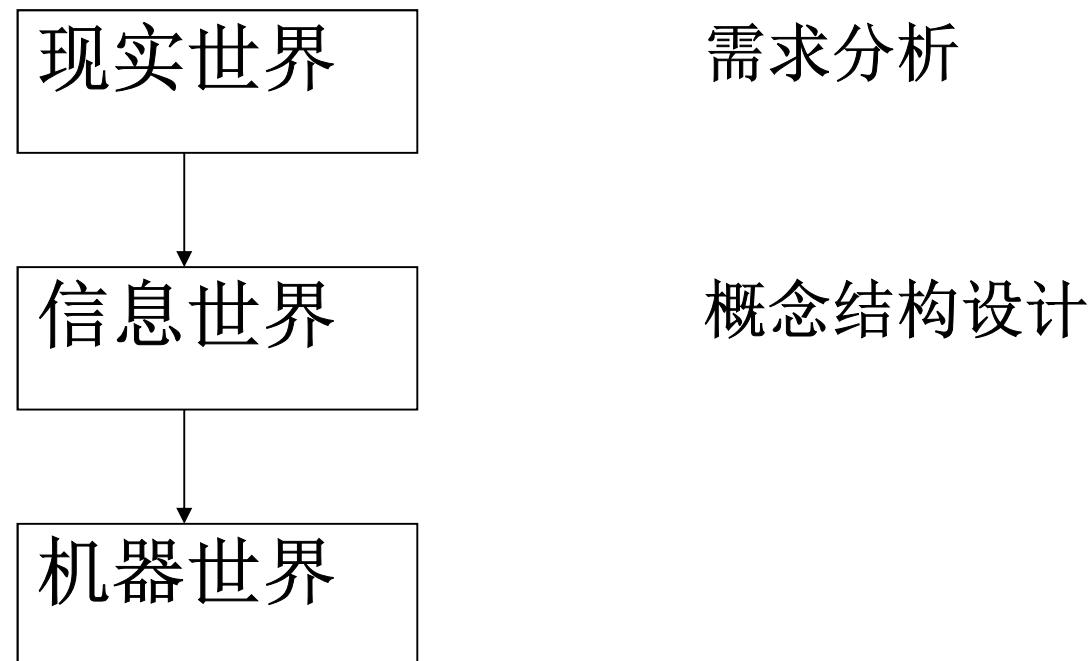
4

- 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



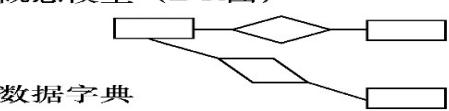
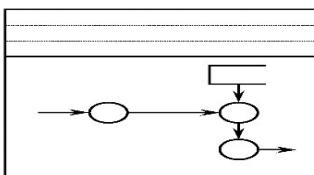
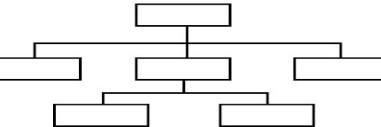
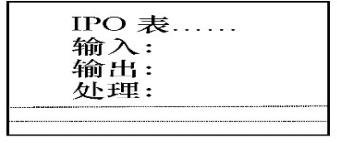
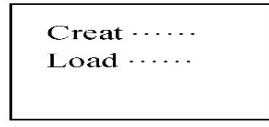
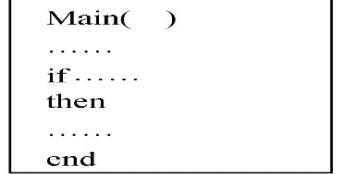
# 概念结构（续）

5





# 数据库设计各个阶段的设计描述

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）

7

## □ 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
- (3) 易于更改
- (4) 易于向关系、网状、层次等各种数据模型转换



# 概念结构（续）

8

## □ 描述概念模型的工具

### □ E-R模型



## 7.3 概念结构设计

9

### 7.3.1 概念结构

### 7.3.2 概念结构设计的方法与步骤

### 7.3.3 数据抽象与局部视图设计

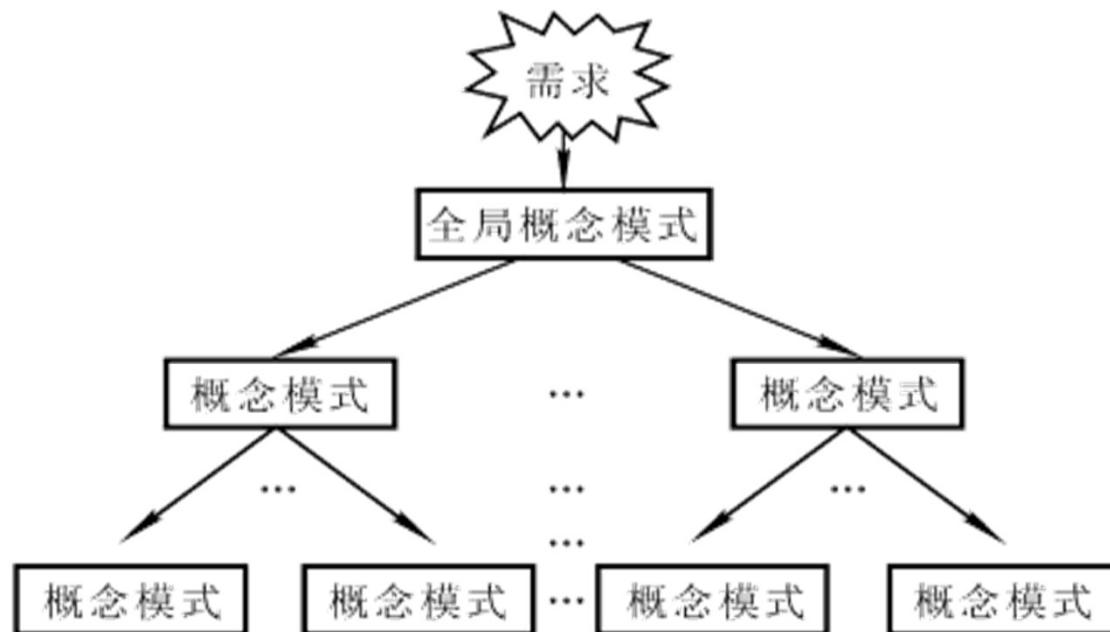
### 7.3.4 视图的集成



## 7.3.2 概念结构设计的方法与步骤

10

- 设计概念结构的四类方法
- 自顶向下
  - 首先定义全局概念结构的框架，然后逐步细化



自顶向下策略

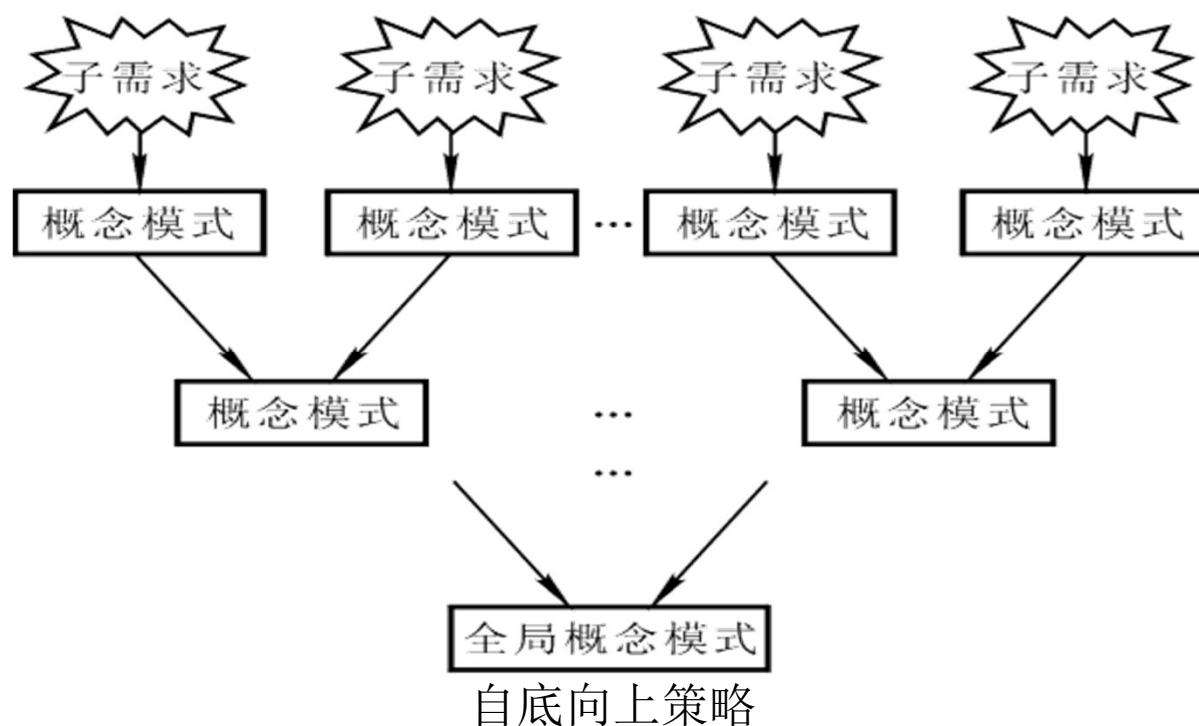


## 7.3.2 概念结构设计的方法与步骤

11

### □ 自底向上

- 首先定义各局部应用的概念结构，然后将它们集成起来，得到全局概念结构



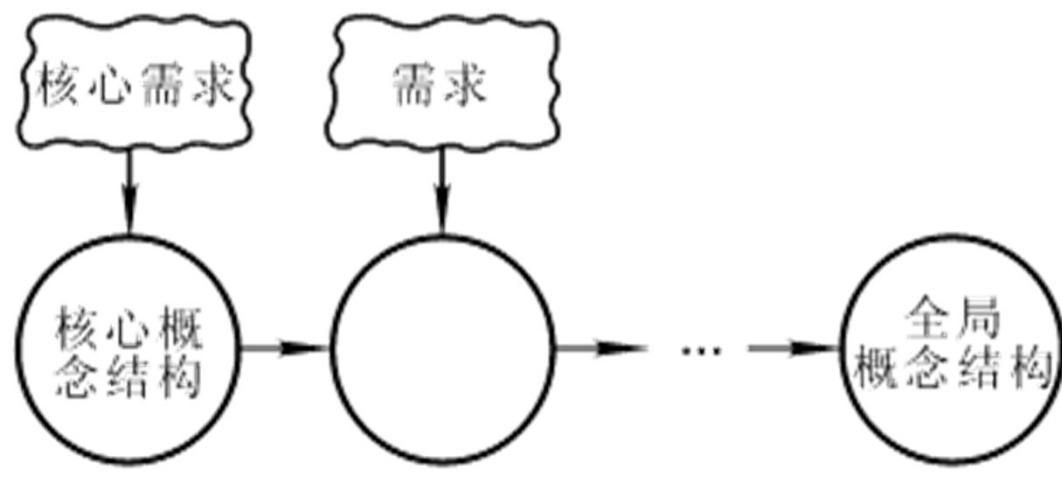


# 概念结构设计的方法与步骤（续）

12

## □ 逐步扩张

- 首先定义最重要的核心概念结构，然后向外扩充，以滚雪球的方式逐步生成其他概念结构，直至总体概念结构



逐步扩张策略



# 概念结构设计的方法与步骤（续）

13

## □ 混合策略

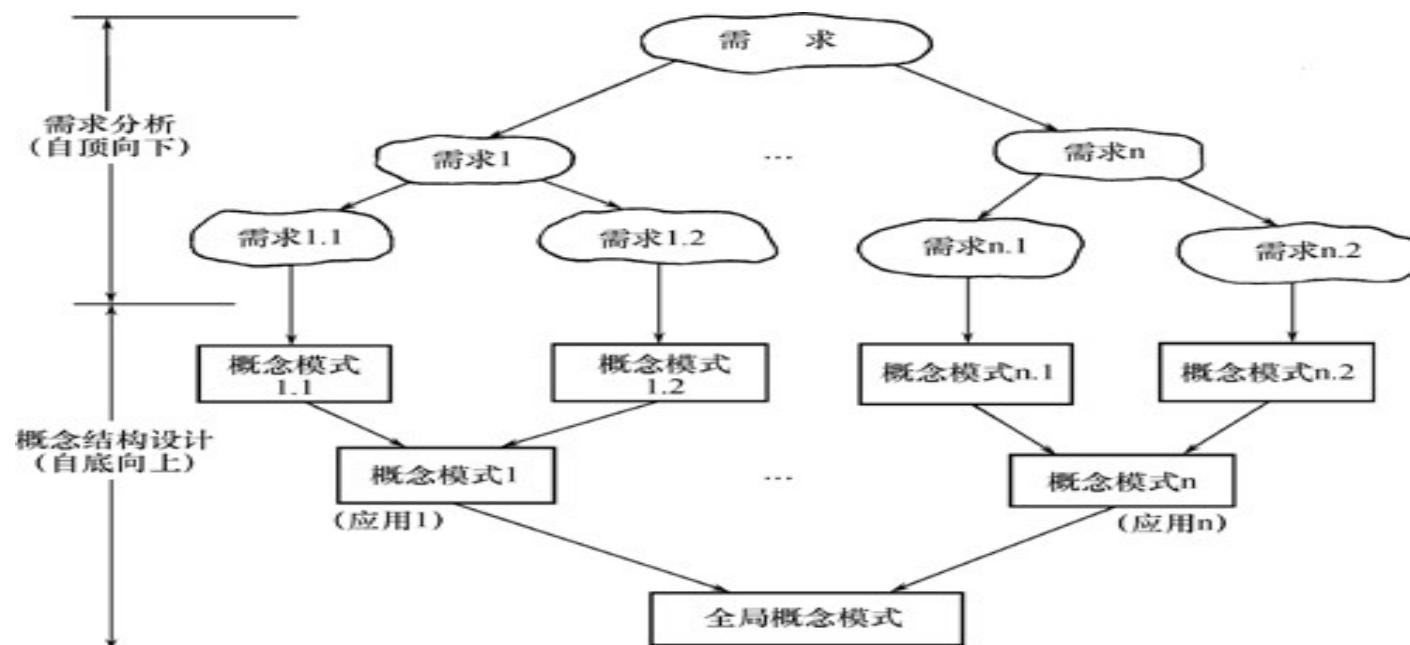
- 将自顶向下和自底向上相结合，用自顶向下策略设计一个全局概念结构的框架，以它为骨架集成由自底向上策略中设计的各局部概念结构。



# 概念结构设计的方法与步骤（续）

## □ 常用策略

- 自顶向下地进行需求分析
- 自底向上地设计概念结构





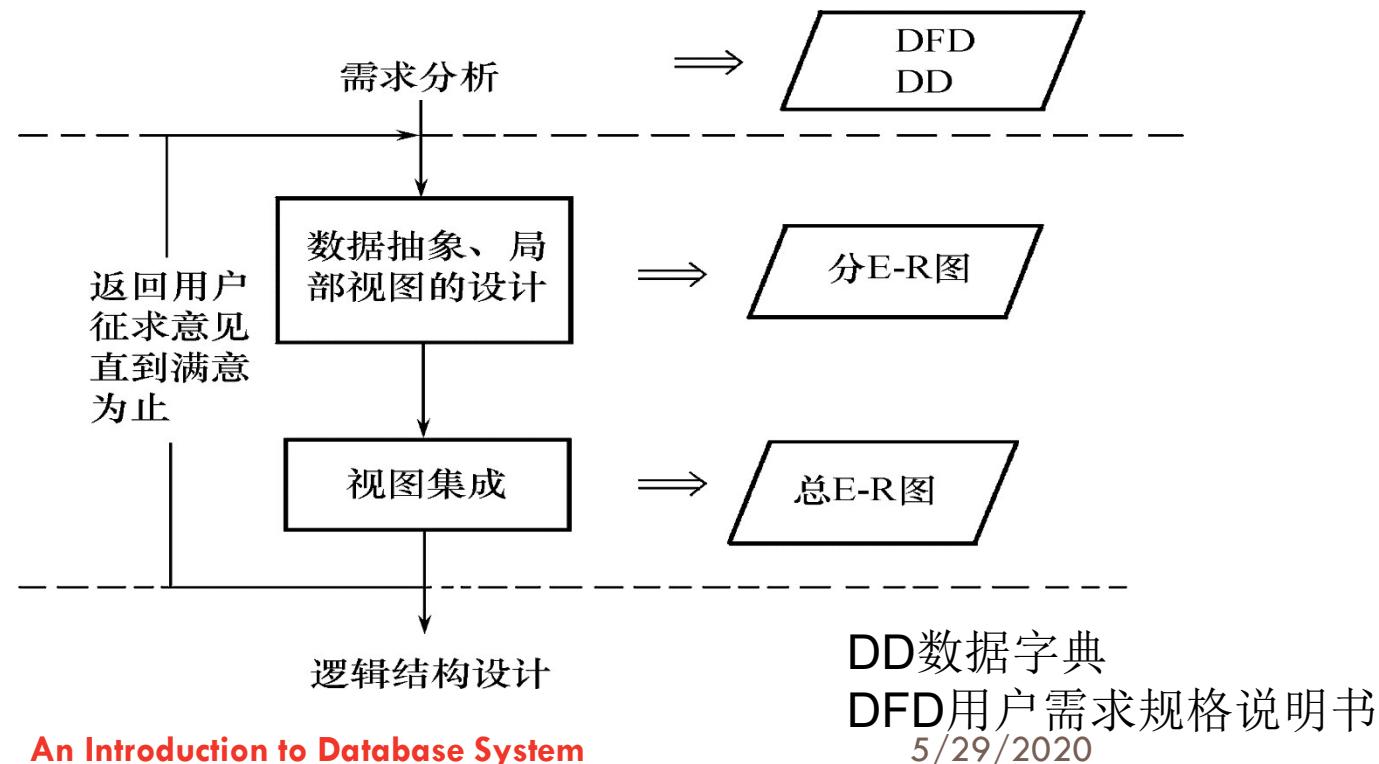
# 概念结构设计的方法与步骤（续）

15

## ❖ 自底向上设计概念结构的步骤

第1步：抽象数据并设计局部视图

第2步：集成局部视图，得到全局概念结构





## 7.3 概念结构设计

16

### 7.3.1 概念结构

### 7.3.2 概念结构设计的方法与步骤

### 7.3.3 数据抽象与局部视图设计

### 7.3.4 视图的集成



## 7.3.3 数据抽象与局部视图设计

17

- 数据抽象
- 局部视图设计



# 数据抽象

18

- 抽象是对实际的人、物、事和概念中抽取所关心的共同特性，忽略非本质的细节，并把这些特性用各种概念精确地加以描述。
- 概念结构是对现实世界的一种抽象



# 数据抽象（续）

19

## □ 三种常用抽象

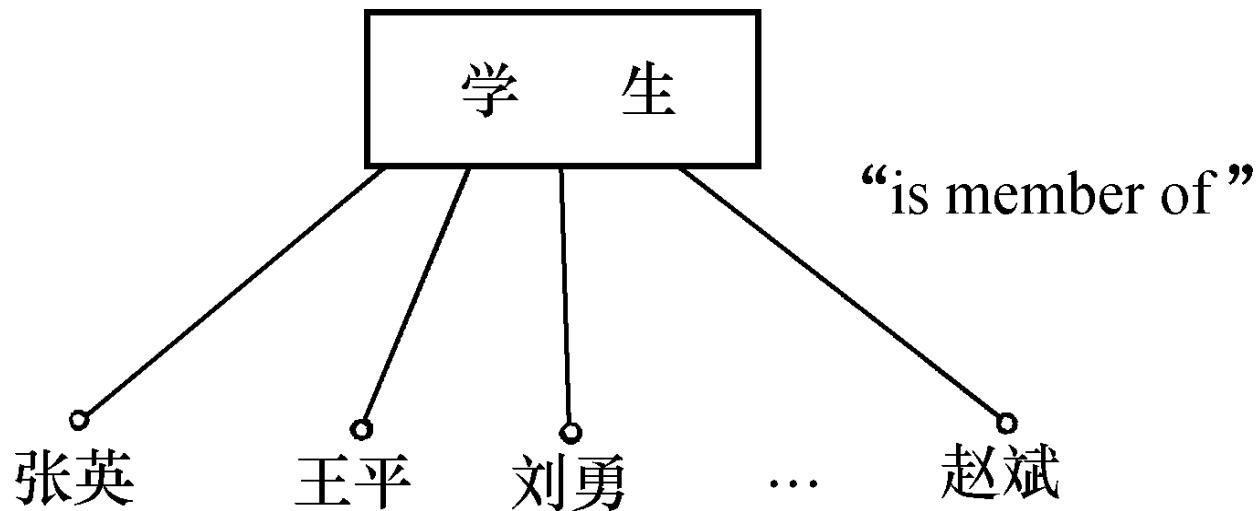
### 1. 分类 (Classification)

- 定义某一类概念作为现实世界中一组对象的类型
- 抽象了对象值和型之间的“is member of”的语义



# 数据抽象（续）

20





# 数据抽象（续）

21

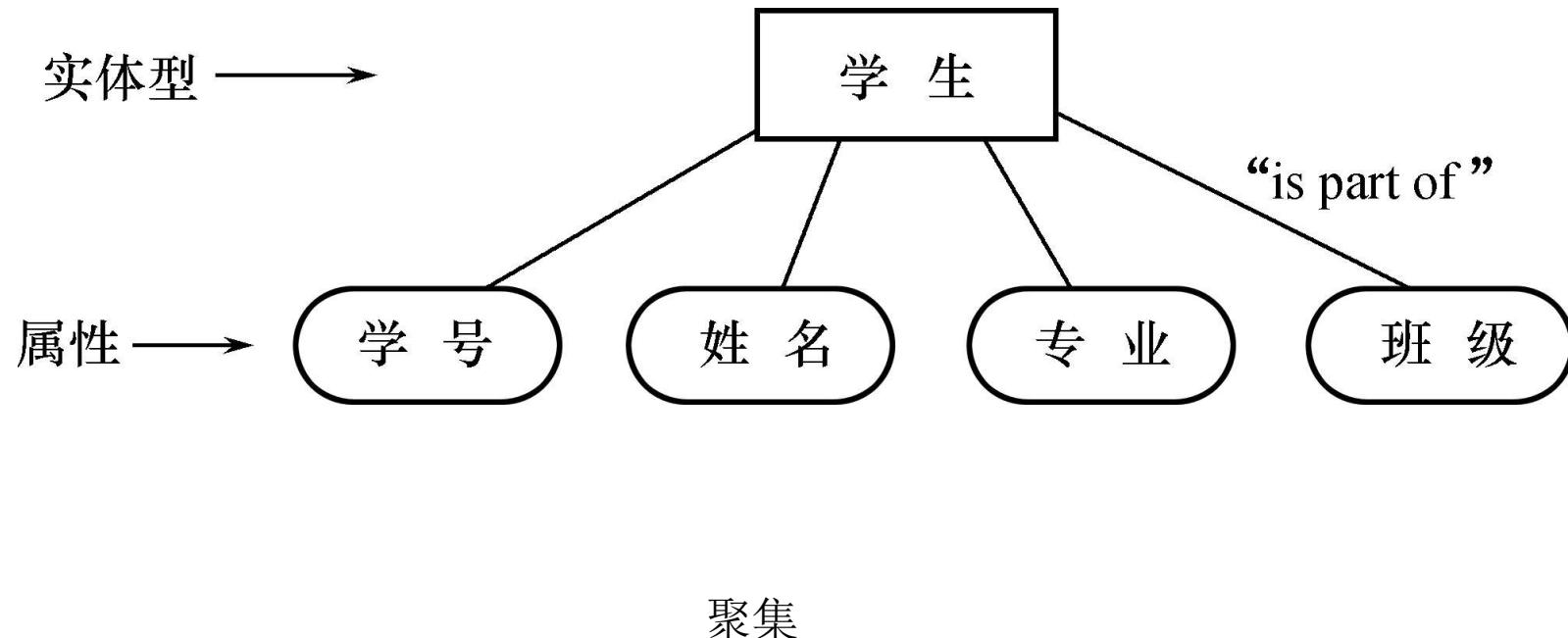
## 2. 聚集 (Aggregation)

- 定义某一类型的组成成分
- 抽象了对象内部类型和成分之间 “is part of”的语义



# 数据抽象（续）

22

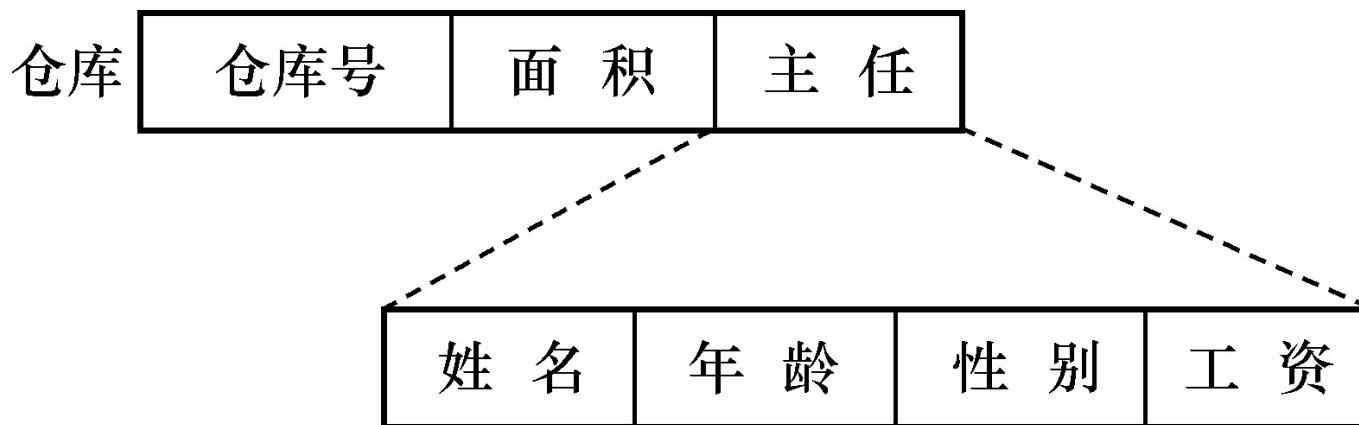




# 数据抽象（续）

23

- 复杂的聚集，某一类型的成分仍是一个聚集



更复杂的聚集



# 数据抽象（续）

24

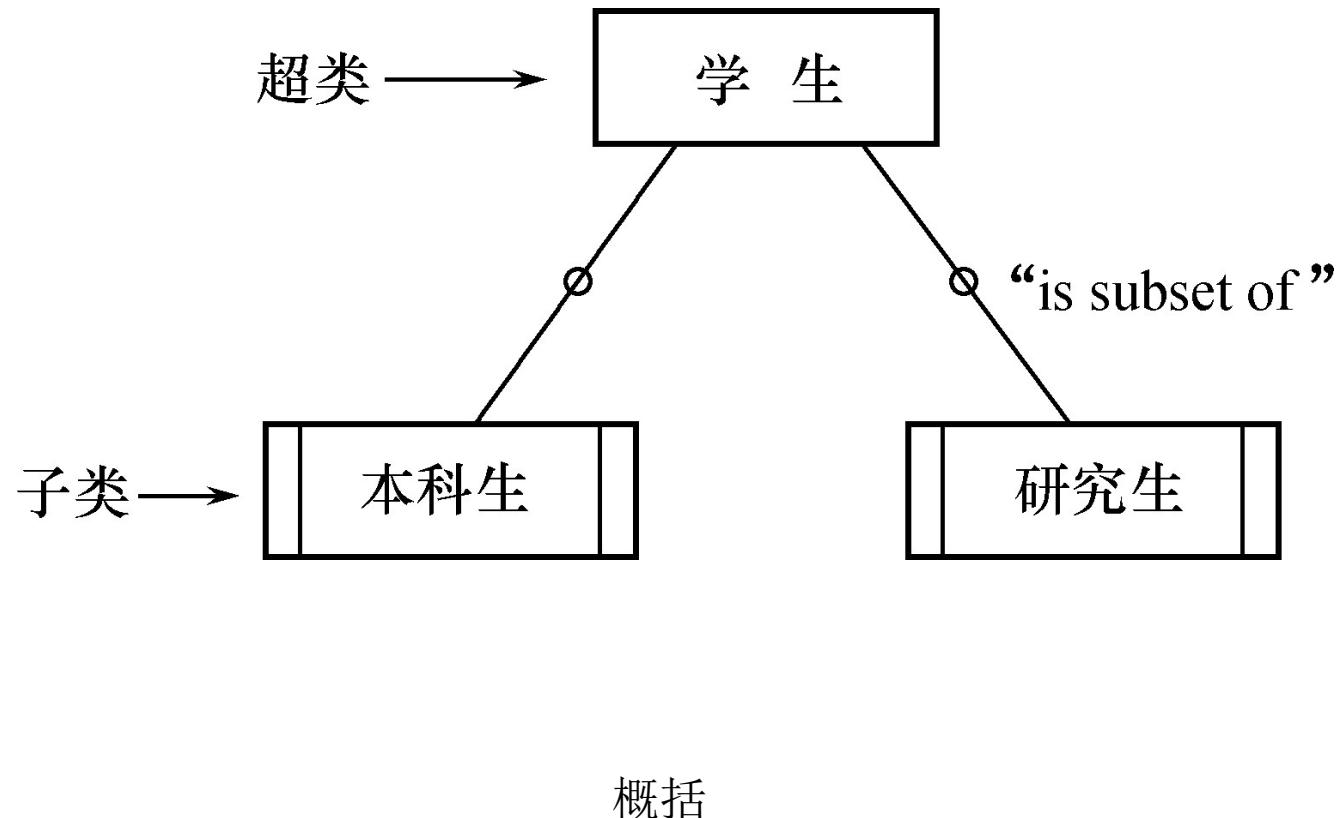
## 3. 概括 (Generalization)

- 定义类型之间的一种子集联系
- 抽象了类型之间的“is subset of”的语义
- 继承性



# 数据抽象 (续)

25





# 局部视图设计

26

设计分E-R图的步骤:

- 1.选择局部应用
- 2.逐一设计分E-R图



# 1. 选择局部应用

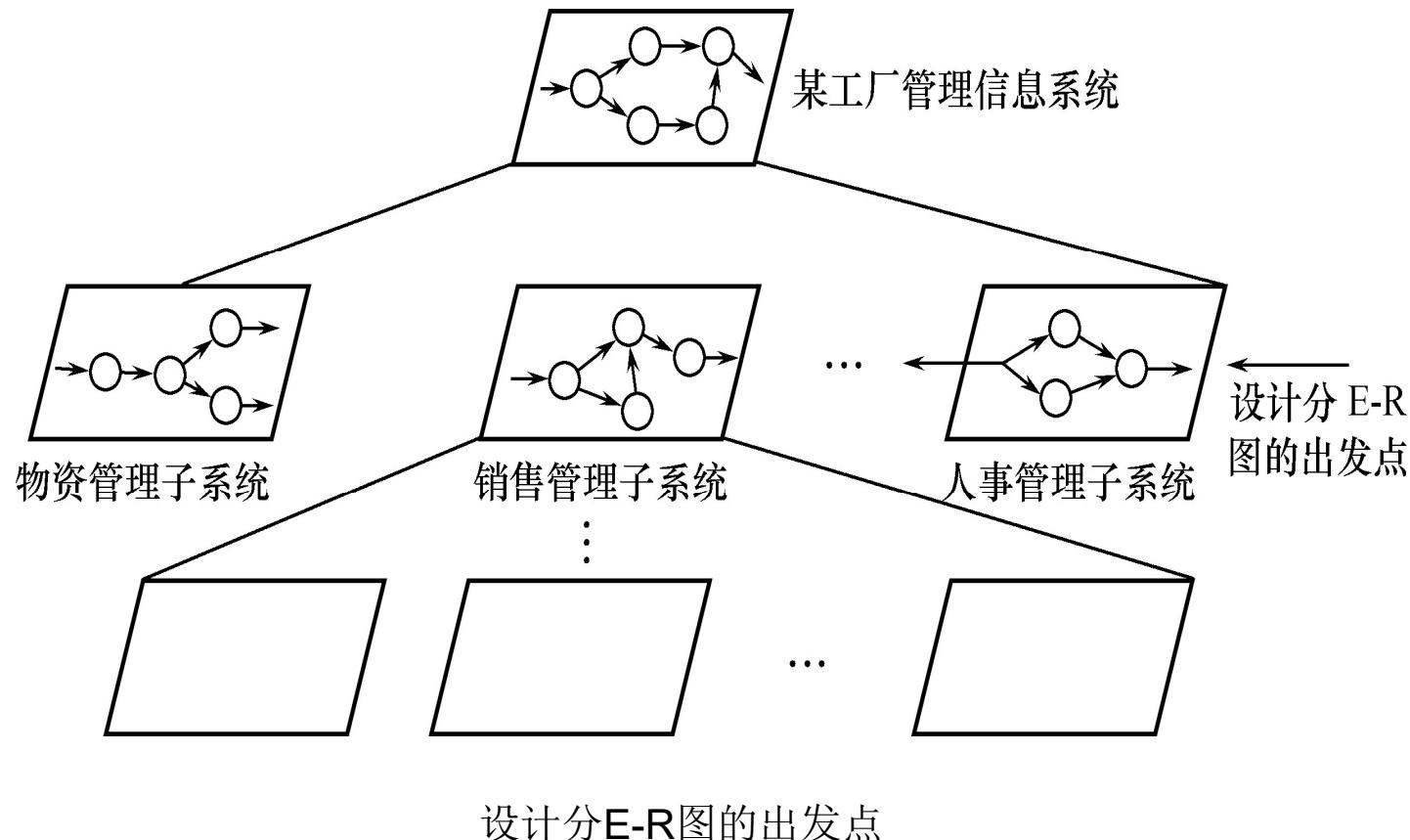
27

- 在多层的数据流图中选择一个适当层次的数据流图，作为设计分E-R图的出发点
- 通常以中层数据流图作为设计分E-R图的依据



# 选择局部应用（续）

28





## 2. 逐一设计分E-R图

29

- 任务
- 将各局部应用涉及的数据分别从数据字典中抽取出来
- 参照数据流图，标定各局部应用中的实体、实体的属性、标识实体的码
- 确定实体之间的联系及其类型（1:1， 1:n， m:n）



# 逐一设计分E-R图（续）

30

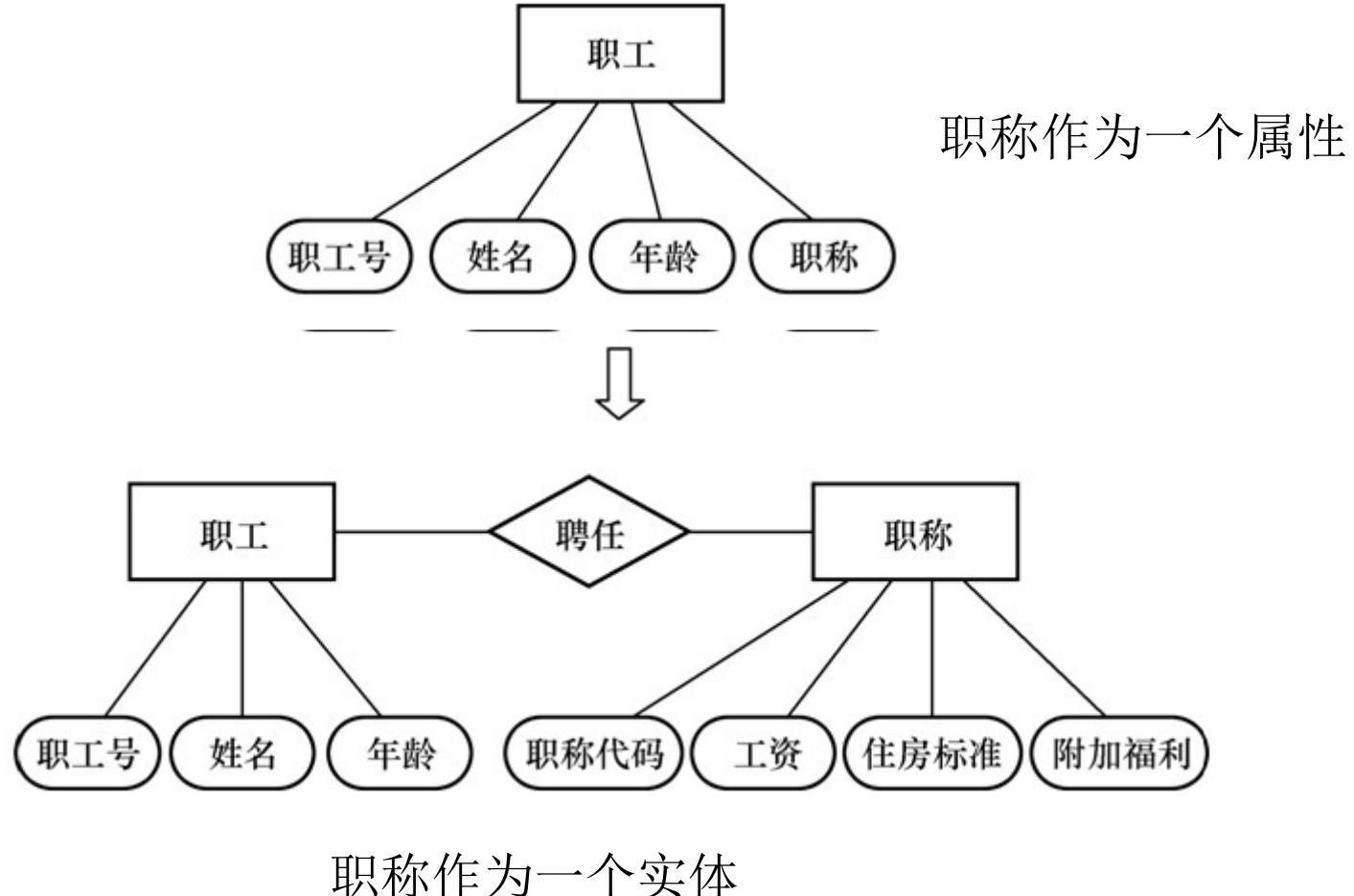
## □ 两条准则：

- (1) 属性不能再具有需要描述的性质。即属性必须是不可分的数据项，不能再由另一些属性组成
- (2) 属性不能与其他实体具有联系。联系只发生在实体之间



# 逐一设计分E-R图（续）

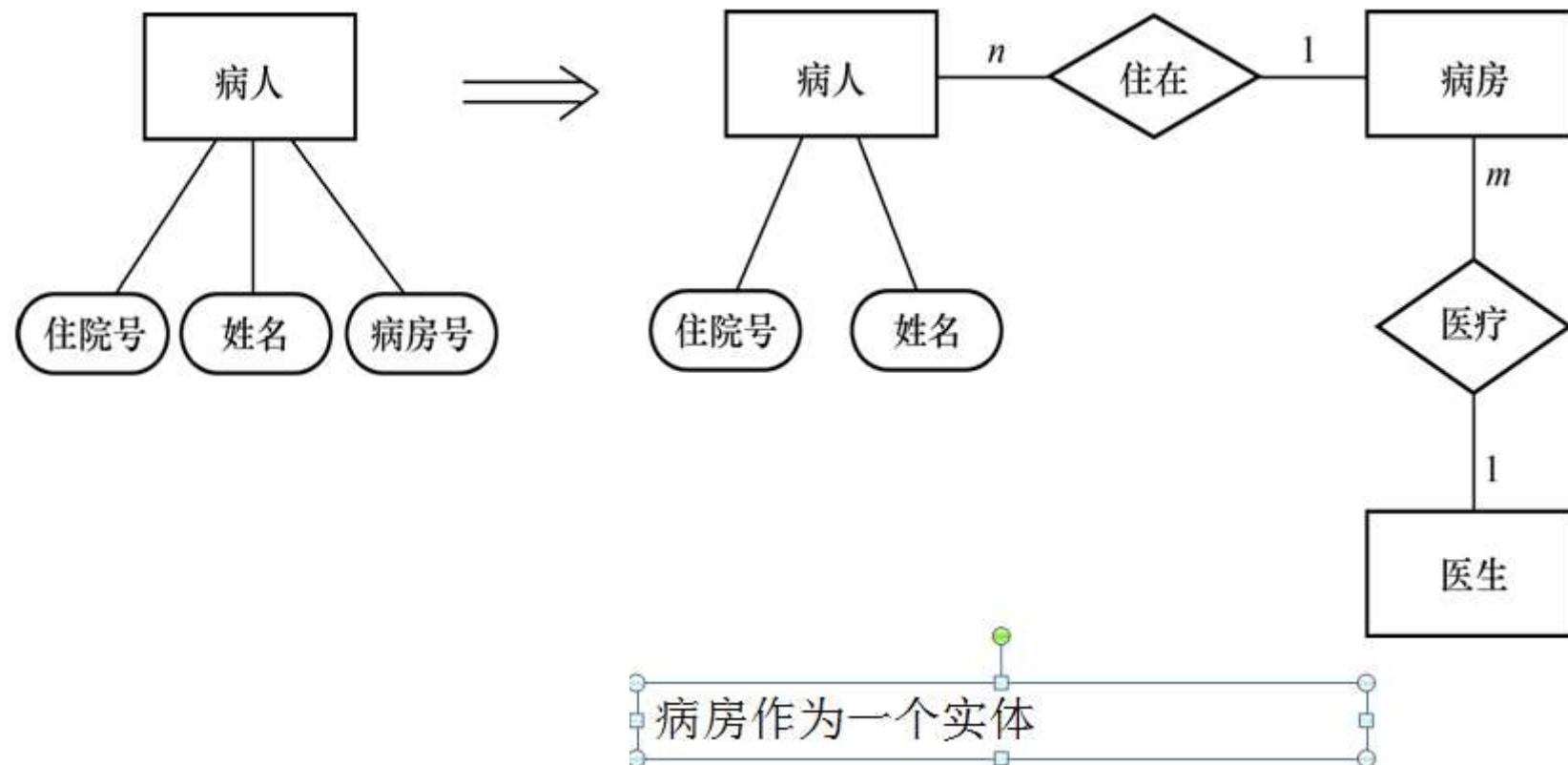
31





# 逐一设计分E-R图（续）

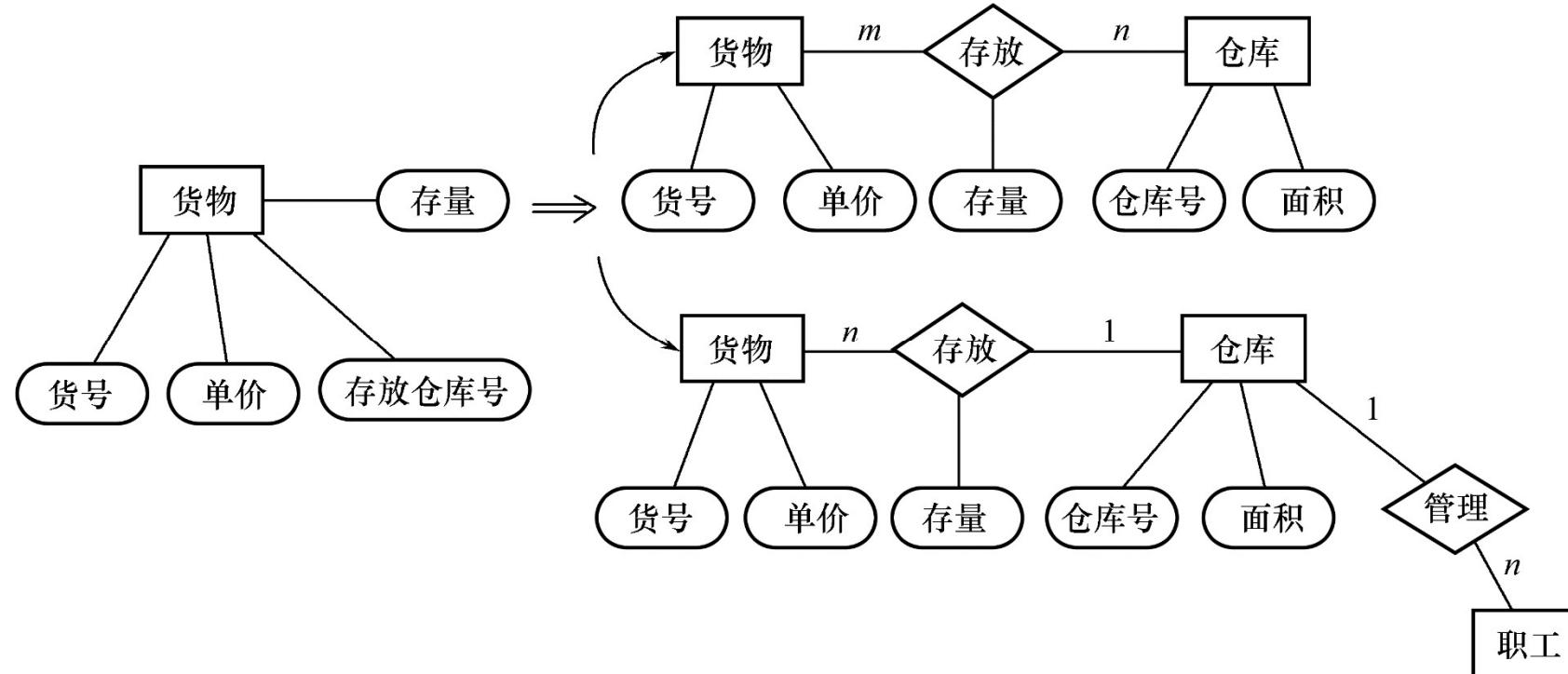
32





# 逐一设计分E-R图（续）

33



仓库作为一个实体



# 逐一设计分E-R图（续）

34

## [实例] 销售管理子系统分E-R图的设计

- ❖ 销售管理子系统的主要功能（需求分析）：
  - 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



## 逐一设计分E-R图（续）

35

- 下图是第一层数据流图，虚线部分划出了系统边界
  - 接收订单、处理订单、开发票、支付过账

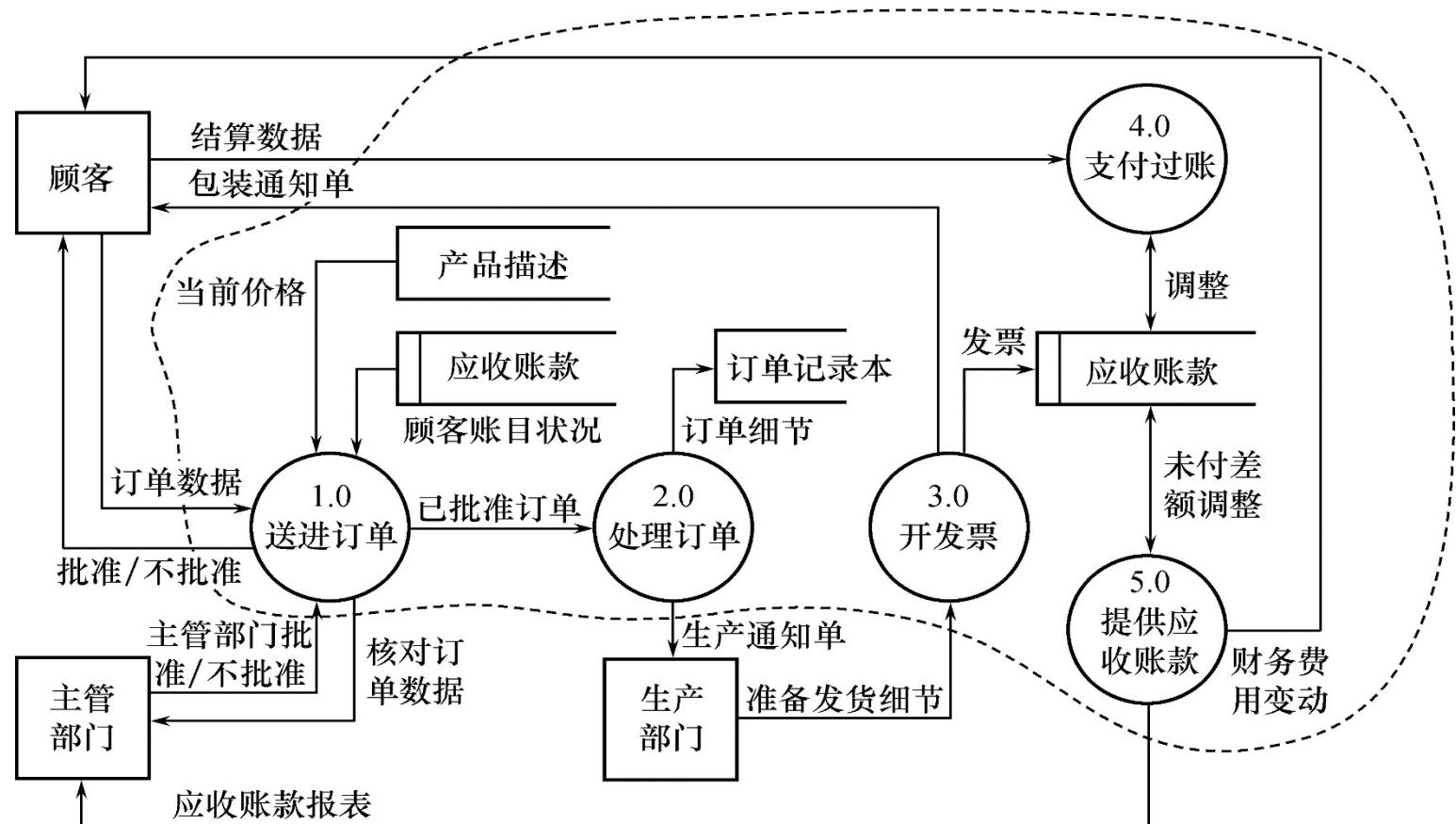


图7.18 销售管理子系统第一层数据流图



## 逐一设计分E-R图（续）

36

- 上图中把系统功能又分为4个子系统，下面四个图是第二层数据流图

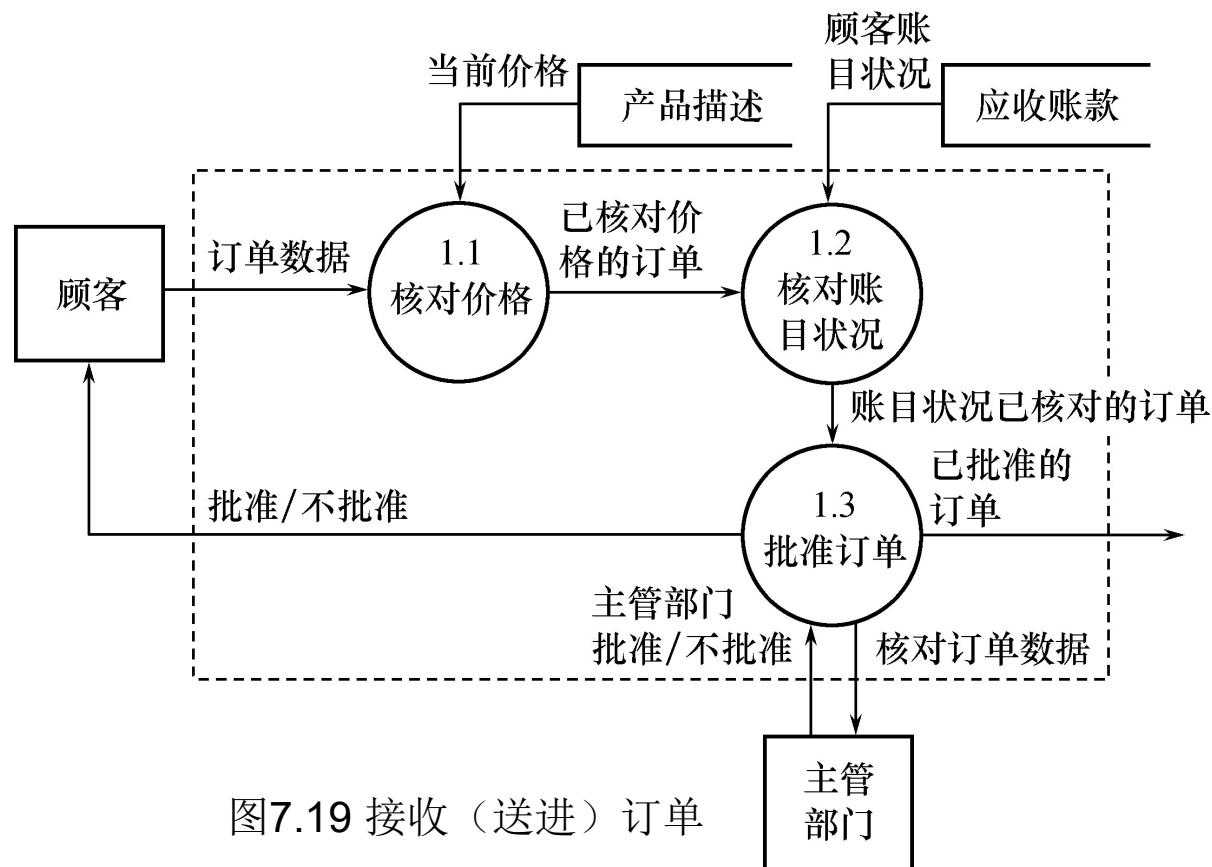


图7.19 接收（送进）订单



# 逐一设计分E-R图（续）

37

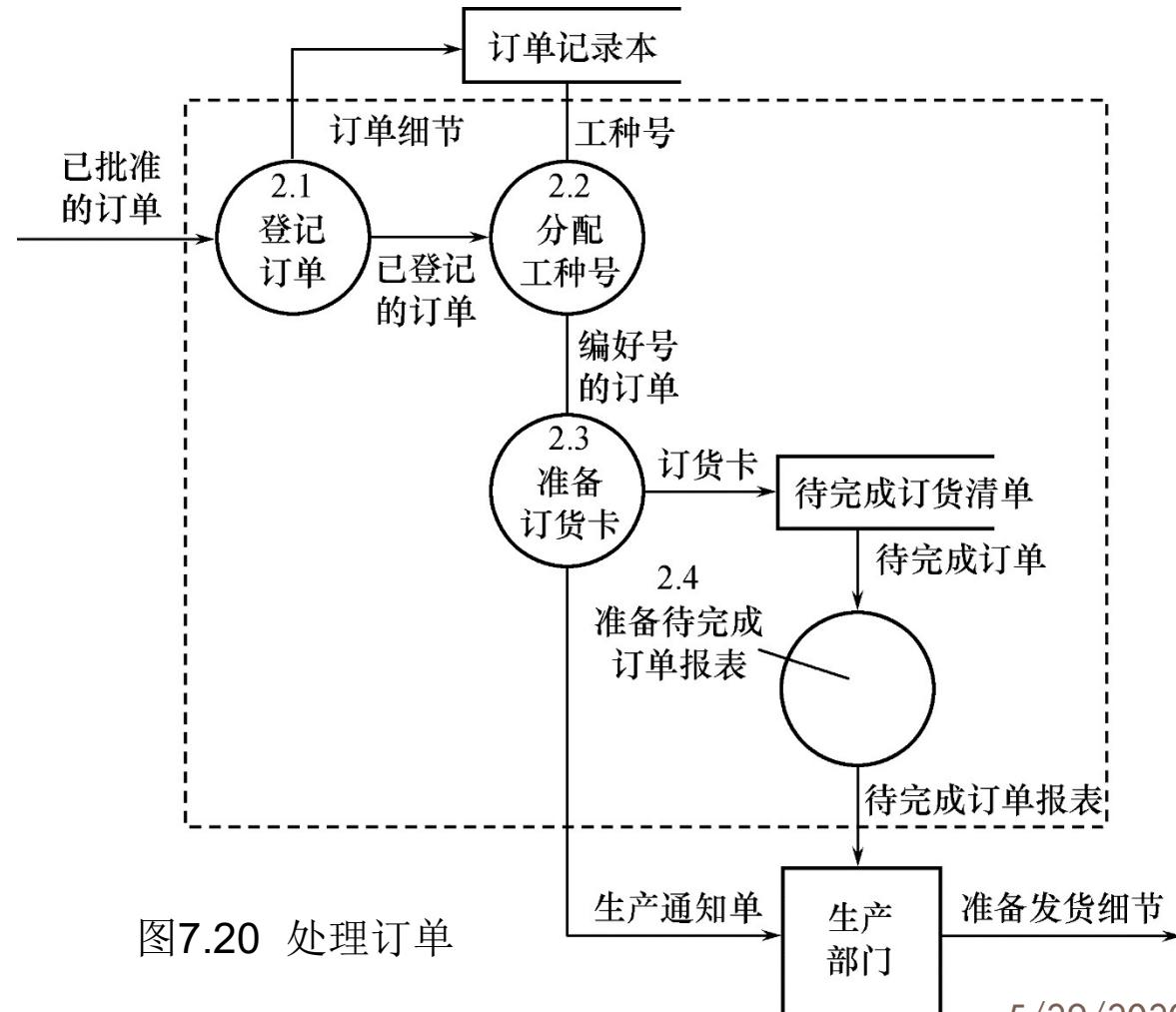


图7.20 处理订单



## 逐一设计分E-R图（续）

38

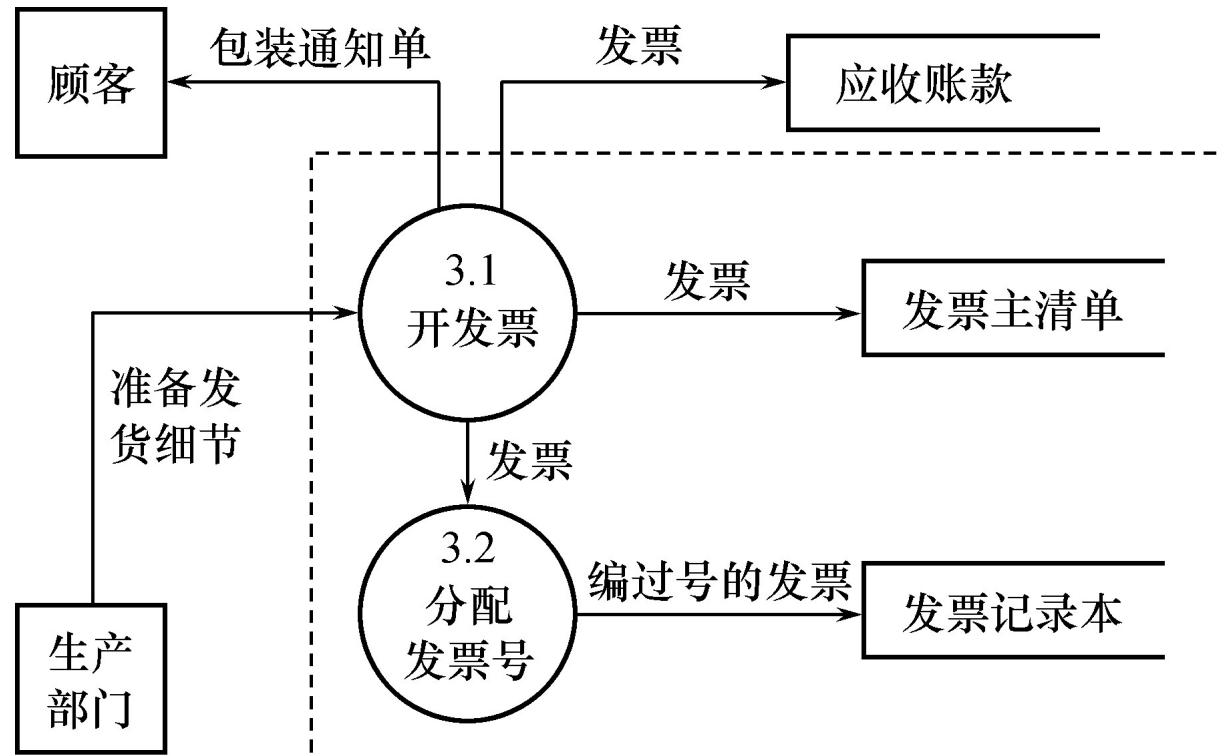
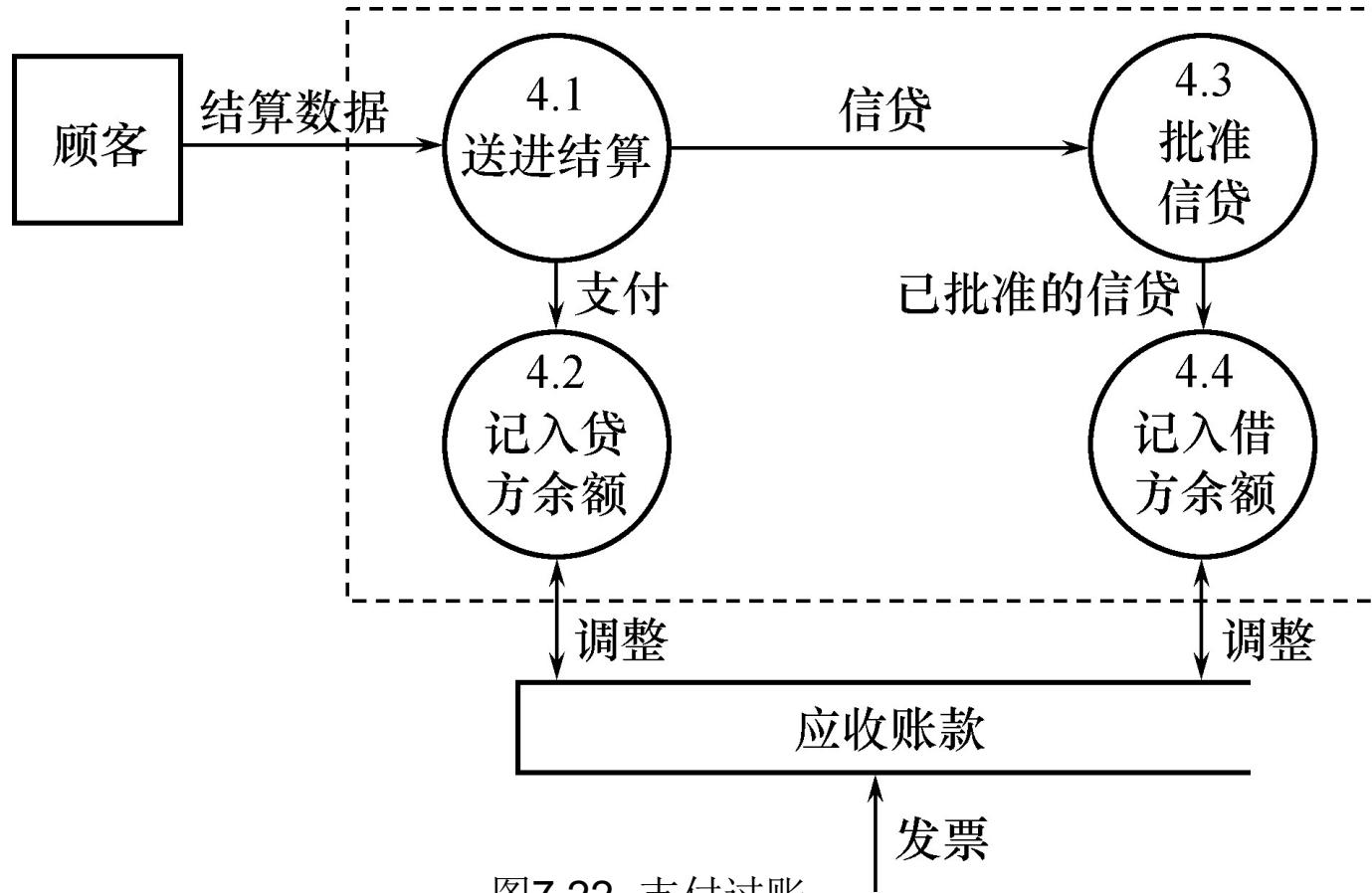


图7.21 开发票



## 逐一设计分E-R图（续）

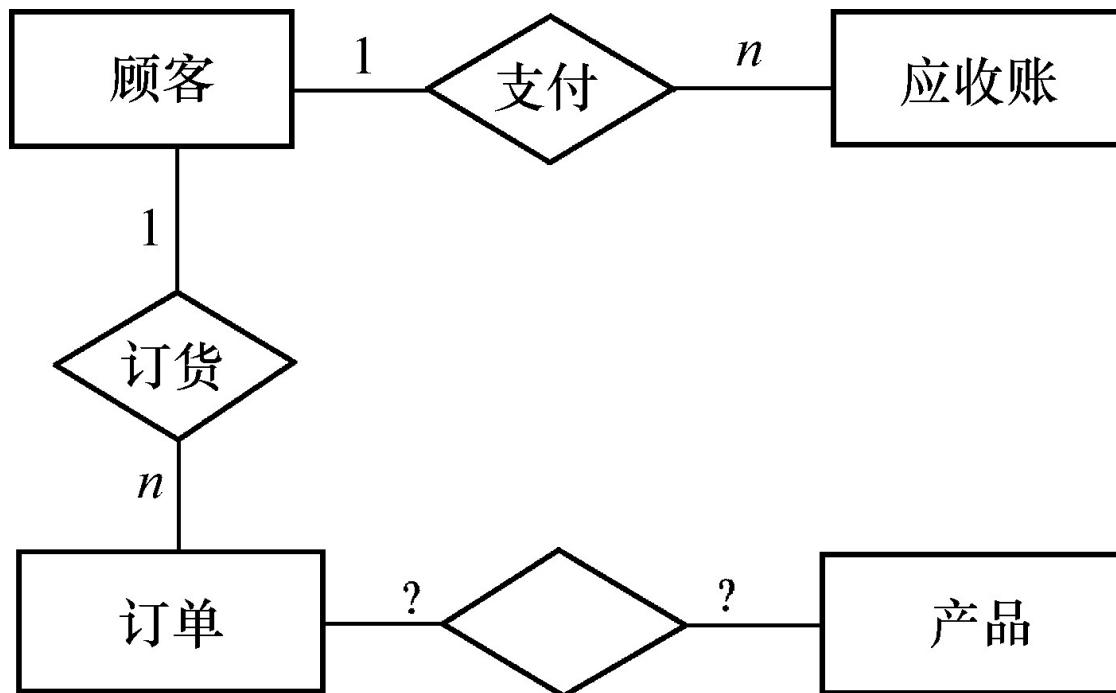
39





## 逐一设计分E-R图 (续)

40



订单、顾客、应收账款相关的分E-R图框架



## 逐一设计分E-R图（续）

41

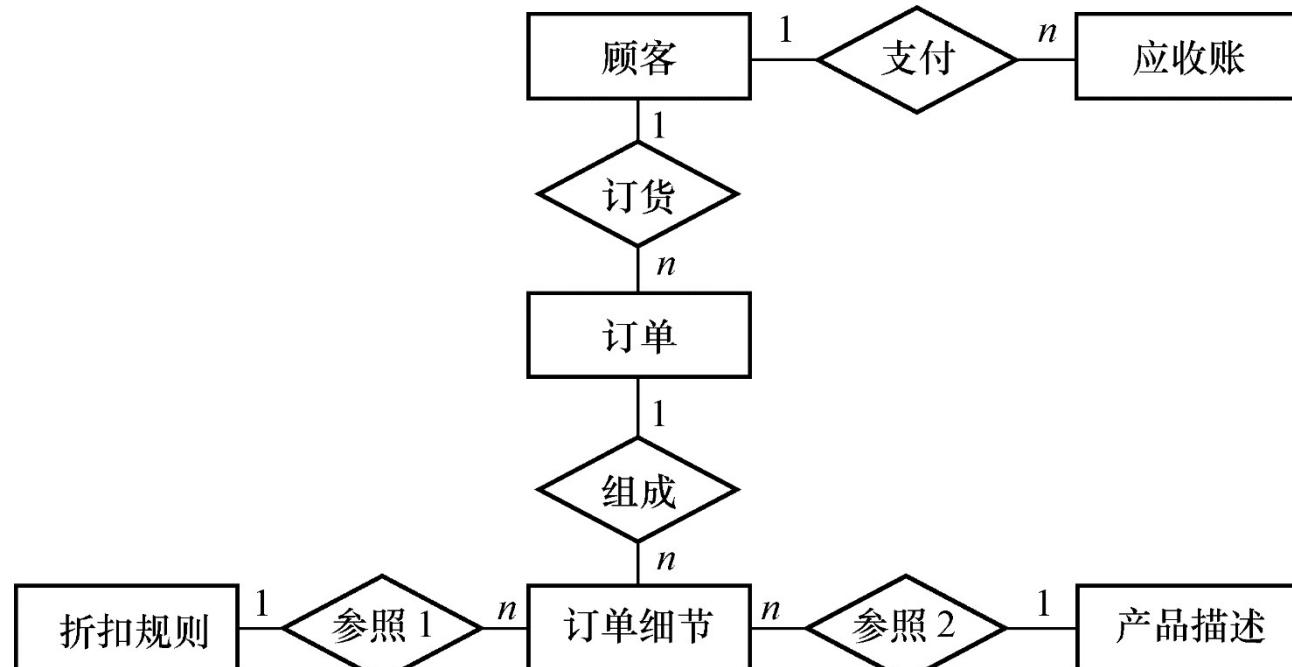
- 参照第二层数据流图和数据字典，遵循两个准则，进行如下调整：
  - (1) 订单与订单细节（零件号，订货数等）是 $1:n$ 的联系
  - (2) 原订单和产品的联系实际上是订单细节和产品的联系。
  - (3) 图7.21中“发票主清单”是一个数据存储，不必作为实体加入分E-R图
  - (4) 工厂对大宗订货给予优惠



## 逐一设计分E-R图（续）

42

□ 得到分E-R图如下图所示



销售管理子系统的分E-R图



## 逐一设计分E-R图（续）

43

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}
- 产品描述：{产品号，产品名，单价，重量}
- 折扣规则：{产品号，订货量，折扣}



## 7.3 概念结构设计

44

### 7.3.1 概念结构

### 7.3.2 概念结构设计的方法与步骤

### 7.3.3 数据抽象与局部视图设计

### 7.3.4 视图的集成



## 7.3.4 视图的集成

45

- 各个局部视图即分E-R图建立好后，还需要对它们进行合并，集成为一个整体的数据概念结构即总E-R图。

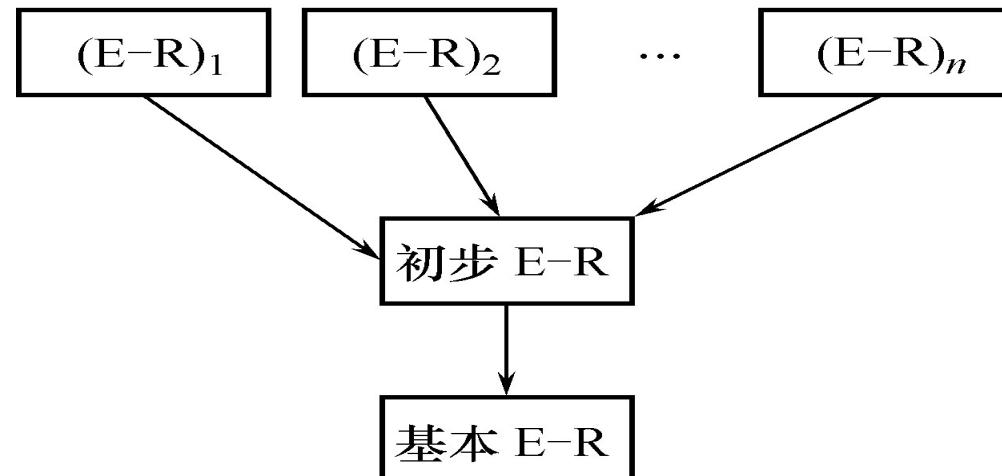


# 视图集成的两种方式

46

## □ 多个分E-R图一次集成

- 一次集成多个分E-R图
- 通常用于局部视图比较简单时



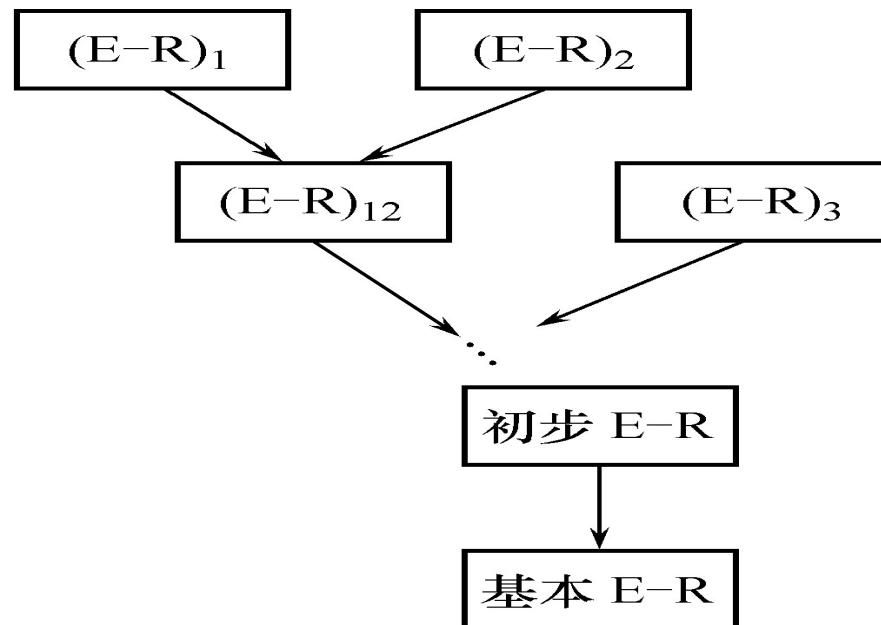


# 视图的集成（续）

47

## □ 逐步集成

- 用累加的方式一次集成两个分E-R图





# 视图的集成（续）

48

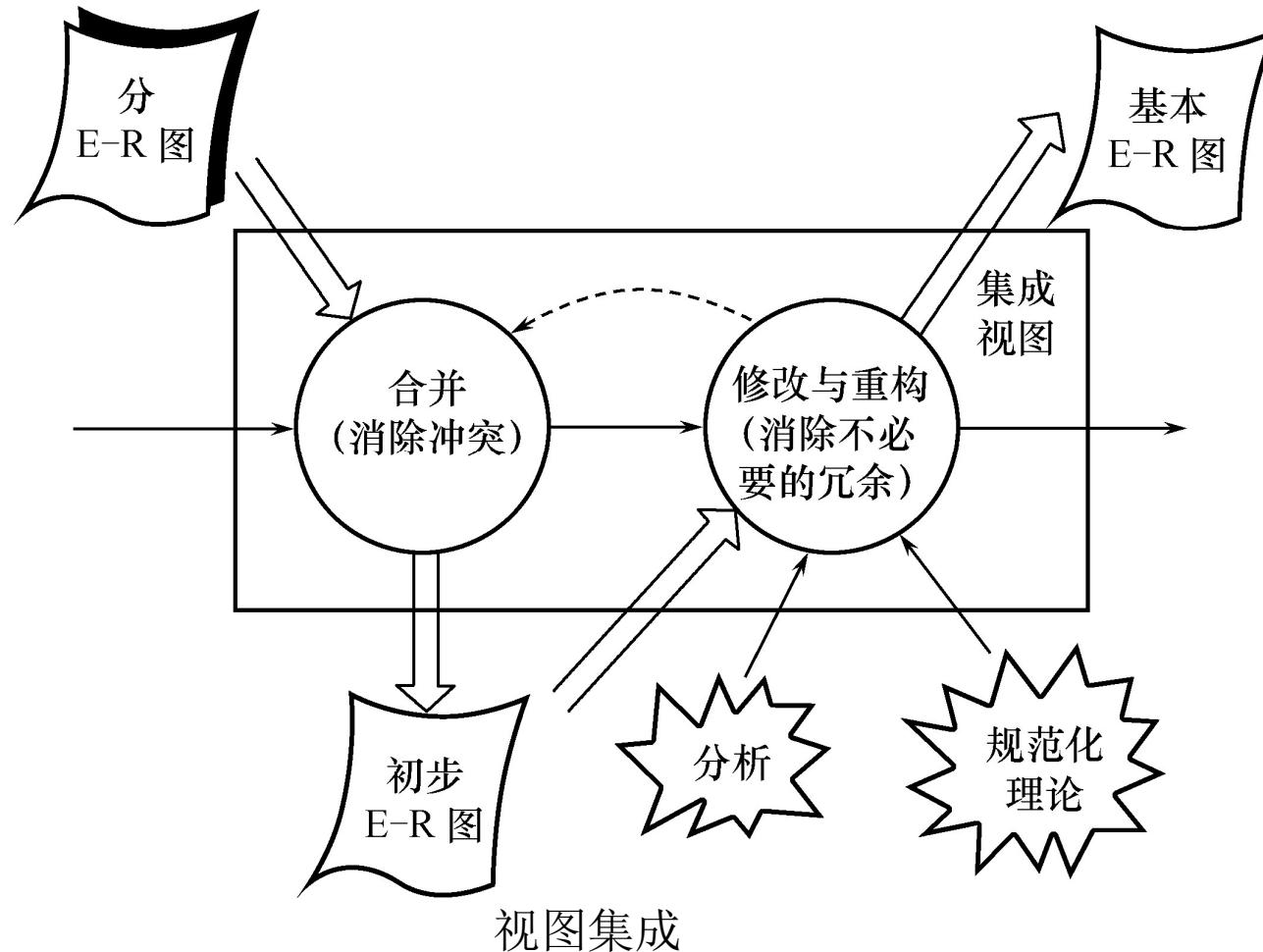
## □ 集成局部E-R图的步骤

1. 合并
2. 修改与重构



# 视图的集成（续）

49





## 合并分E-R图，生成初步E-R图

50

- 各分E-R图存在冲突
  - 各个分E-R图之间必定会存在许多不一致的地方
- 合并分E-R图的主要工作与关键
  - 合理消除各分E-R图的冲突



## 合并分E-R图，生成初步E-R图（续）

51

- 冲突的种类
  - 属性冲突
  - 命名冲突
  - 结构冲突



# 1. 属性冲突

52

## □ 两类属性冲突

### □ 属性域冲突

- 属性值的类型
- 取值范围
- 取值集合不同

### □ 属性取值单位冲突



## 2. 命名冲突

53

### □ 两类命名冲突

- 同名异义：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）：同一意义的对象在不同的局部应用中具有不同的名字



### 3. 结构冲突

54

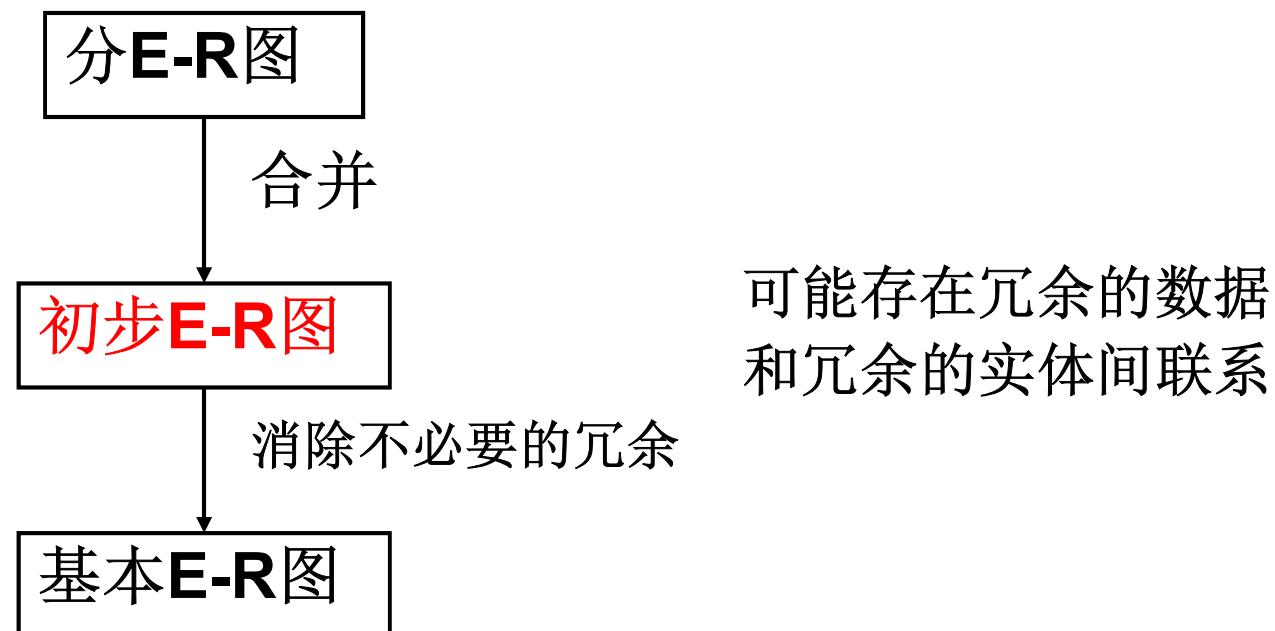
- 三类结构冲突
  - 同一对象在不同应用中具有不同的抽象
  - 同一实体在不同分E-R图中所包含的属性个数和属性排列次序不完全相同
  - 实体之间的联系在不同局部视图中呈现不同的类型



# 消除不必要的冗余，设计基本E-R图

55

- 基本任务
  - 消除不必要的冗余，设计生成基本E-R图





## 消除不必要的冗余，设计基本E-R图（续）

56

- 冗余
- 消除冗余的方法



# 1. 冗余

57

- 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
- 冗余数据和冗余联系容易破坏数据库的完整性，给数据库维护增加困难
- 消除不必要的冗余后的初步E-R图称为基本E-R图



# 消除冗余的方法

58

## □ 分析方法

- 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的逻辑关系来消除冗余



# 消除冗余的方法（续）

59

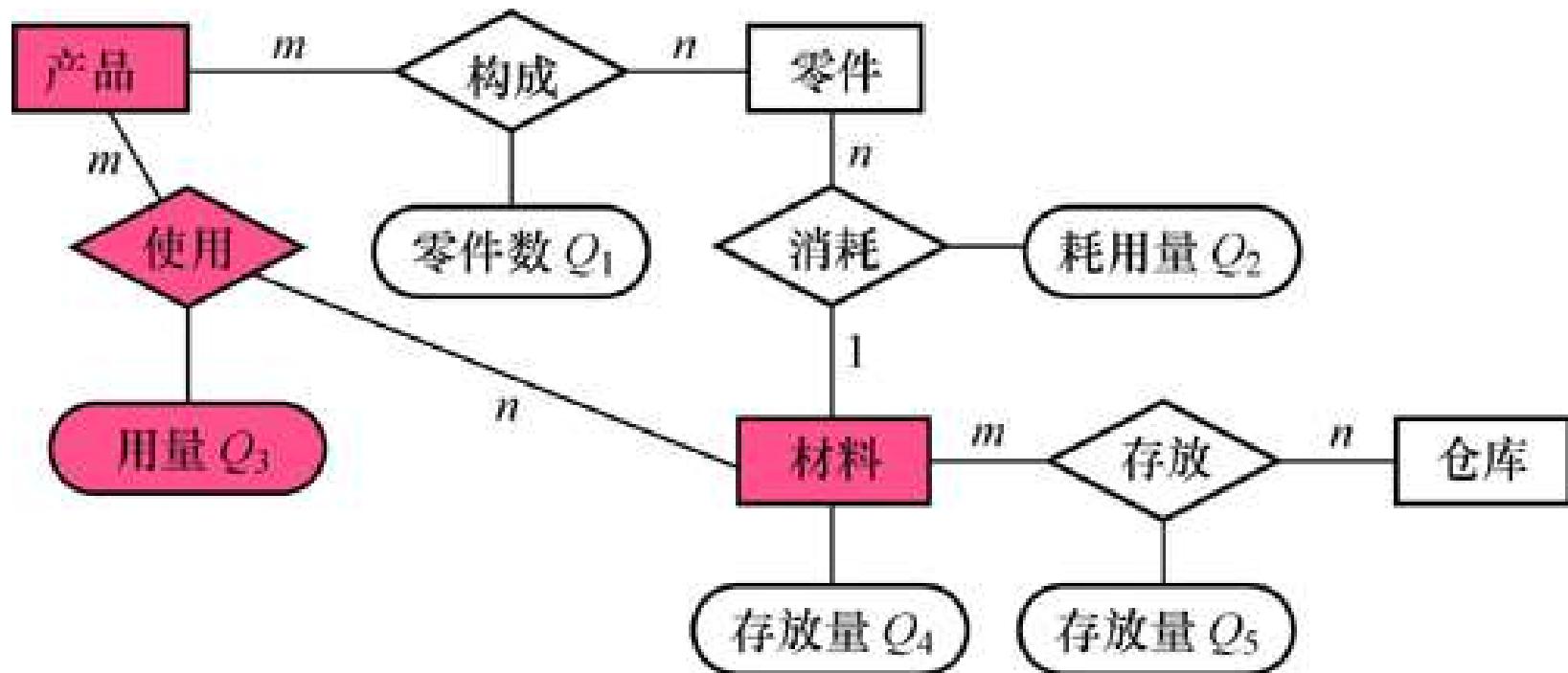


图7.26消除冗余 ( $Q_3=Q_1 \times Q_2$  和  $Q_4$  是冗余)



# 消除冗余的方法（续）

60

- 效率VS冗余信息
  - 需要根据用户的整体需求来确定
- 若人为地保留了一些冗余数据，则应把数据字典中数据关联的说明作为完整性约束条件
  - $Q4 = \sum Q5$
  - 一旦Q5修改后就应当触发完整性检查，对Q4进行修改



# 消除冗余的方法（续）

61

- 规范化理论
- 函数依赖的概念提供了消除冗余联系的形式化工具

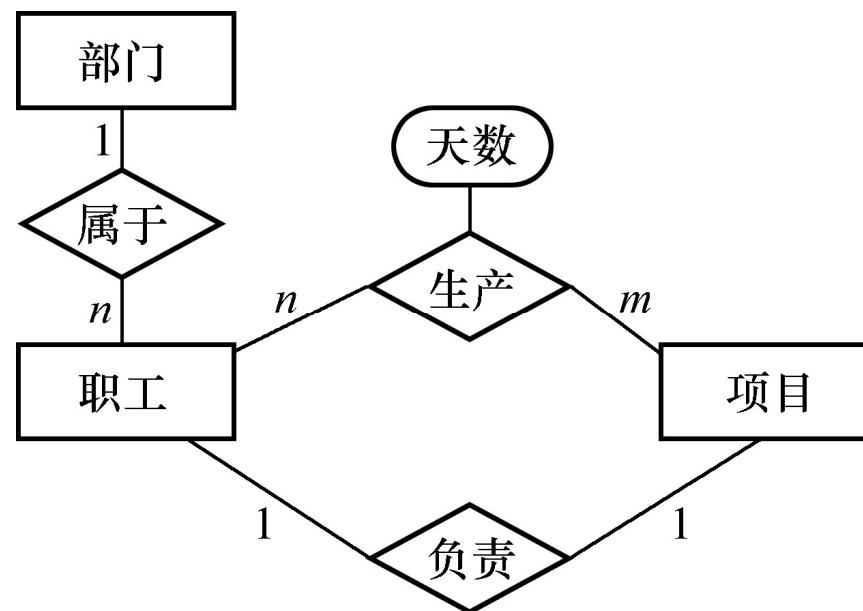


# 消除冗余的方法（续）

62

## □ 方法

1. 确定分E-R图实体之间的数据依赖，并用实体码之间的函数依赖表示。



劳动人事管理的分E-R图



# 消除冗余的方法（续）

63

上图中，

- 部门和职工之间一对多的联系可表示为：  
职工号 → 部门号
- 职工和产品之间多对多的联系可表示为：  
(职工号, 项目号) → 工作天数
- 得到函数依赖集  $F_L$



## 消除冗余的方法（续）

64

2. 求 $F_L$ 的最小覆盖 $G_L$ ，差集为 $D = F_L - G_L$ 。

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。

- (1) 冗余的联系一定在 $D$ 中，而 $D$ 中的联系不一定是冗余的；
- (2) 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。



## 消除冗余，设计生成基本E-R图实例

65

[实例] 某工厂管理信息系统的视图集成。

图1.14(c)、图7.24、图7.29分别为该厂物资、销售和劳动人事管理的分E-R图

图7.30为该系统的基本E-R图



## 消除冗余，设计生成基本E-R图实例（续）

66

### 该厂物资管理分E-R图

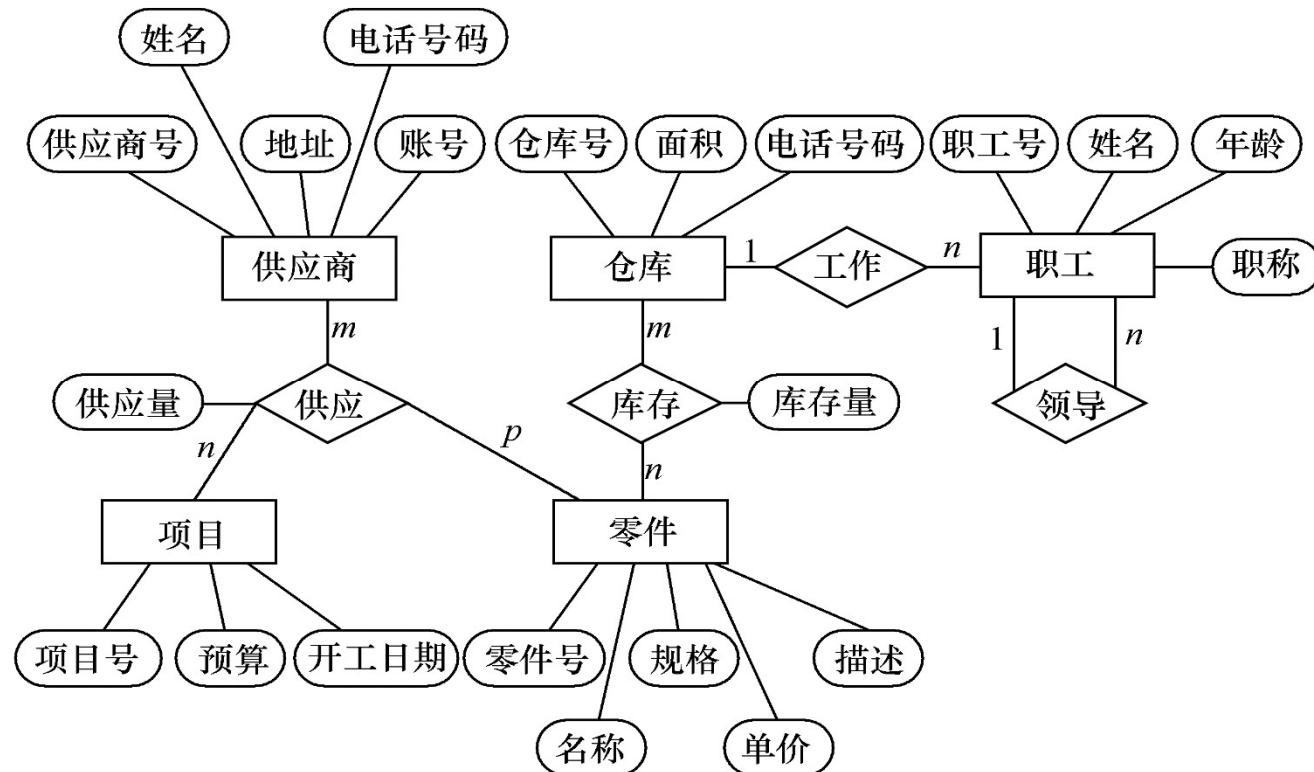


图1.14(c) 工厂物资管理E-R图



## 消除冗余，设计生成基本E-R图实例（续）

67

该厂销售管理分E-R图

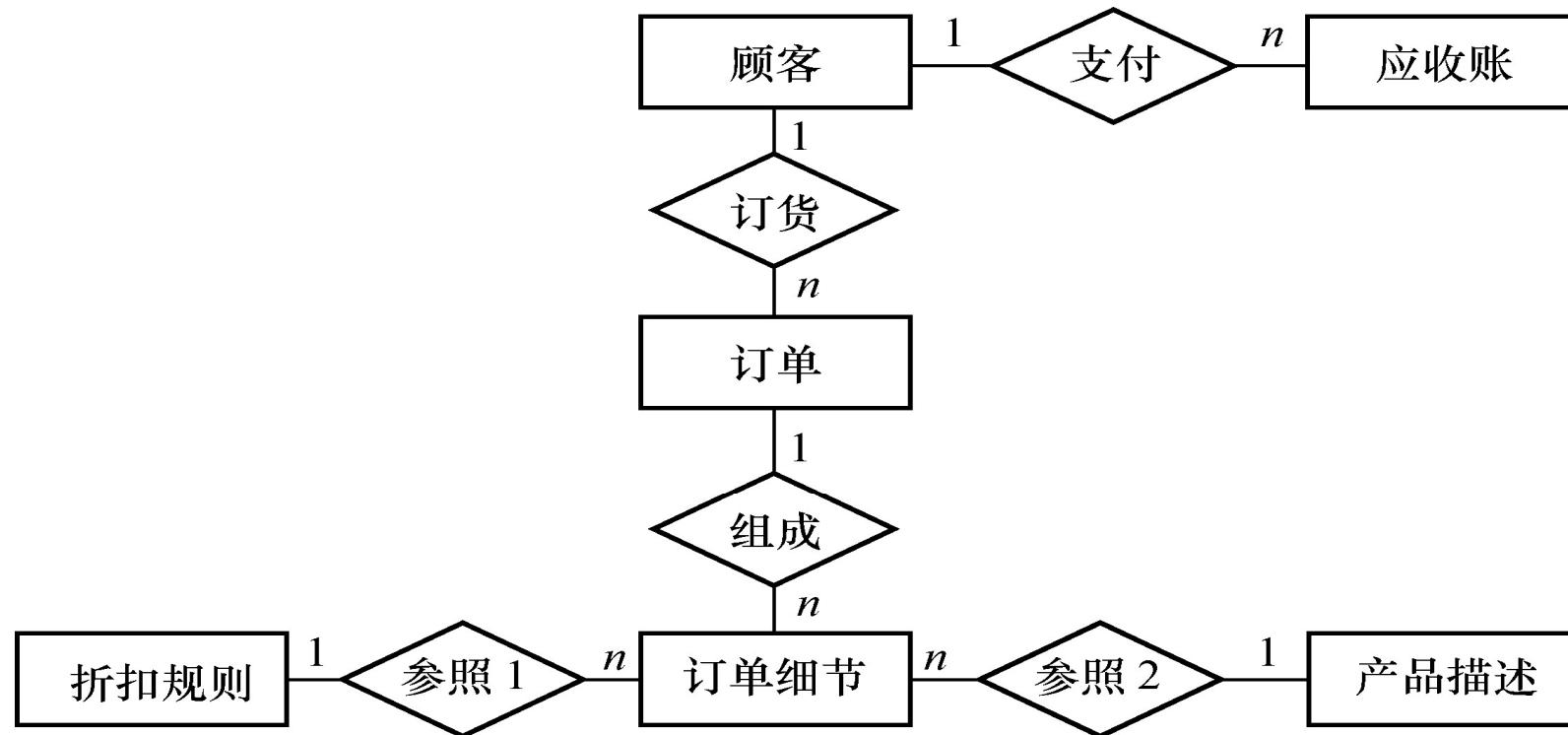


图7.24 销售管理子系统的分E-R图



## 消除冗余，设计生成基本E-R图实例（续）

68

该厂劳动人事管理分E-R图

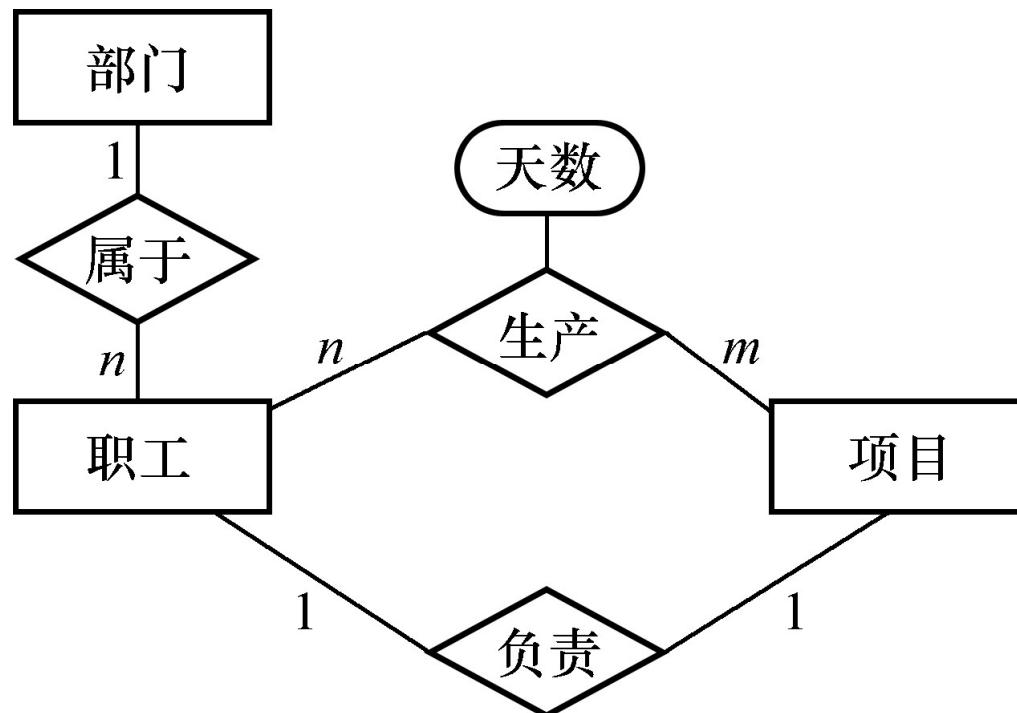


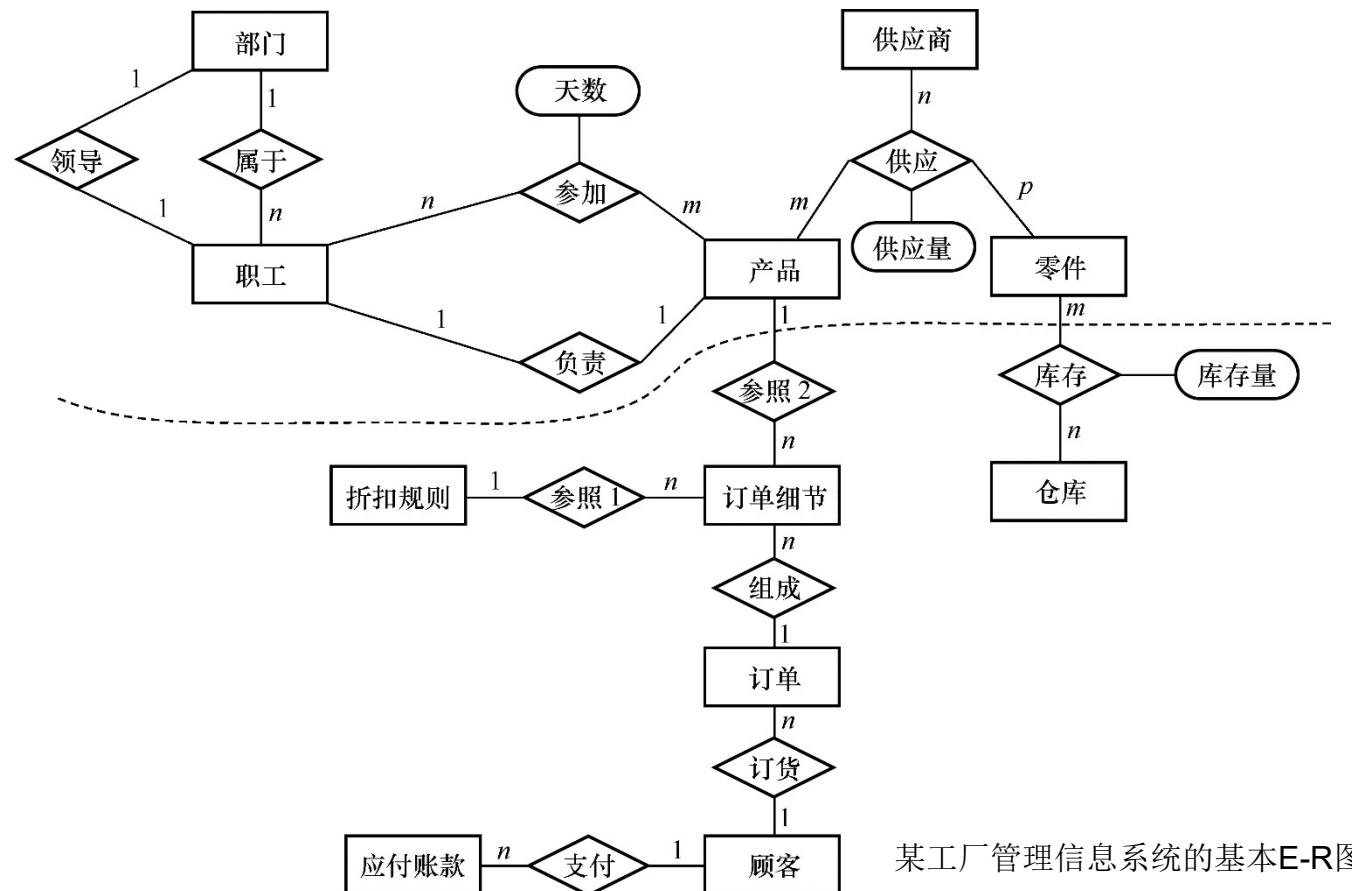
图7.29 劳动人事管理的分E-R图



# 消除冗余，设计生成基本E-R图实例（续）

69

## 系统的基本E-R(图7.30)





## 消除冗余，设计生成基本E-R图实例（续）

70

集成过程，解决了以下问题：

- 异名同义，项目和产品含义相同（统一为产品）
- 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



# 验证整体概念结构

71

- 视图集成后形成一个整体的数据库概念结构，对该整体概念结构还必须进行进一步验证，确保它能够满足下列条件：
  - 整体概念结构内部必须具有一致性，不存在互相矛盾的表达
  - 整体概念结构能准确地反映原来的每个视图结构，包括属性、实体及实体间的联系
  - 整体概念结构能满足需要分析阶段所确定的所有要求



## 验证整体概念结构（续）

72

- 整体概念结构最终还应该提交给用户，征求用户和有关人员的意见，进行评审、修改和优化，然后把它确定下来，作为数据库的概念结构，作为进一步设计数据库的依据。



# 概念结构设计小结

73

- 概念结构设计的步骤
- 抽象数据并设计局部视图
- 集成局部视图，得到全局概念结构
- 验证整体概念结构



# 概念结构设计小结

74

- 数据抽象
  - 分类
  - 聚集
  - 概括



# 概念结构设计小结

75

- 设计局部视图
  - 1. 选择局部应用
  - 2. 逐一设计分E-R图
- 标定局部应用中的实体、属性、码，实体间的联系
- 用E-R图描述出来



# 概念结构设计小结

76

## □ 集成局部视图

### □ 1. 合并分E-R图，生成初步E-R图

#### ➤ 消除冲突

- 属性冲突
- 命名冲突
- 结构冲突

### □ 2. 修改与重构

#### ➤ 消除不必要的冗余，设计生成基本E-R图

- 分析方法
- 规范化理论



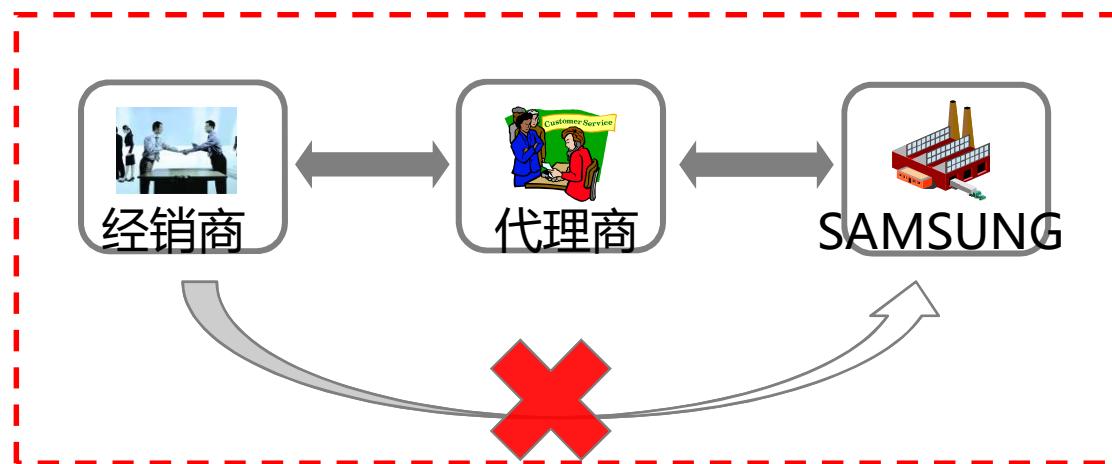
# 三星公司渠道管理系统

77

## □ 背景与意义

- **目前渠道结构复杂，渠道间客户恶性竞争：**经销商试图越级直接接收来自三星公司的产品，而不是按照三星—代理商—经销商的顺序，这将会导致市场秩序混乱。针对上述弊端，可以利用信息技术进行管理，以管理系统的形式规划设计三星渠道管理工作，从而实时掌握渠道和终端的有效信息。与此同时，以系统的方式也可以优化物流，库存，订单和其他信息，以便数据流更加准确透明。

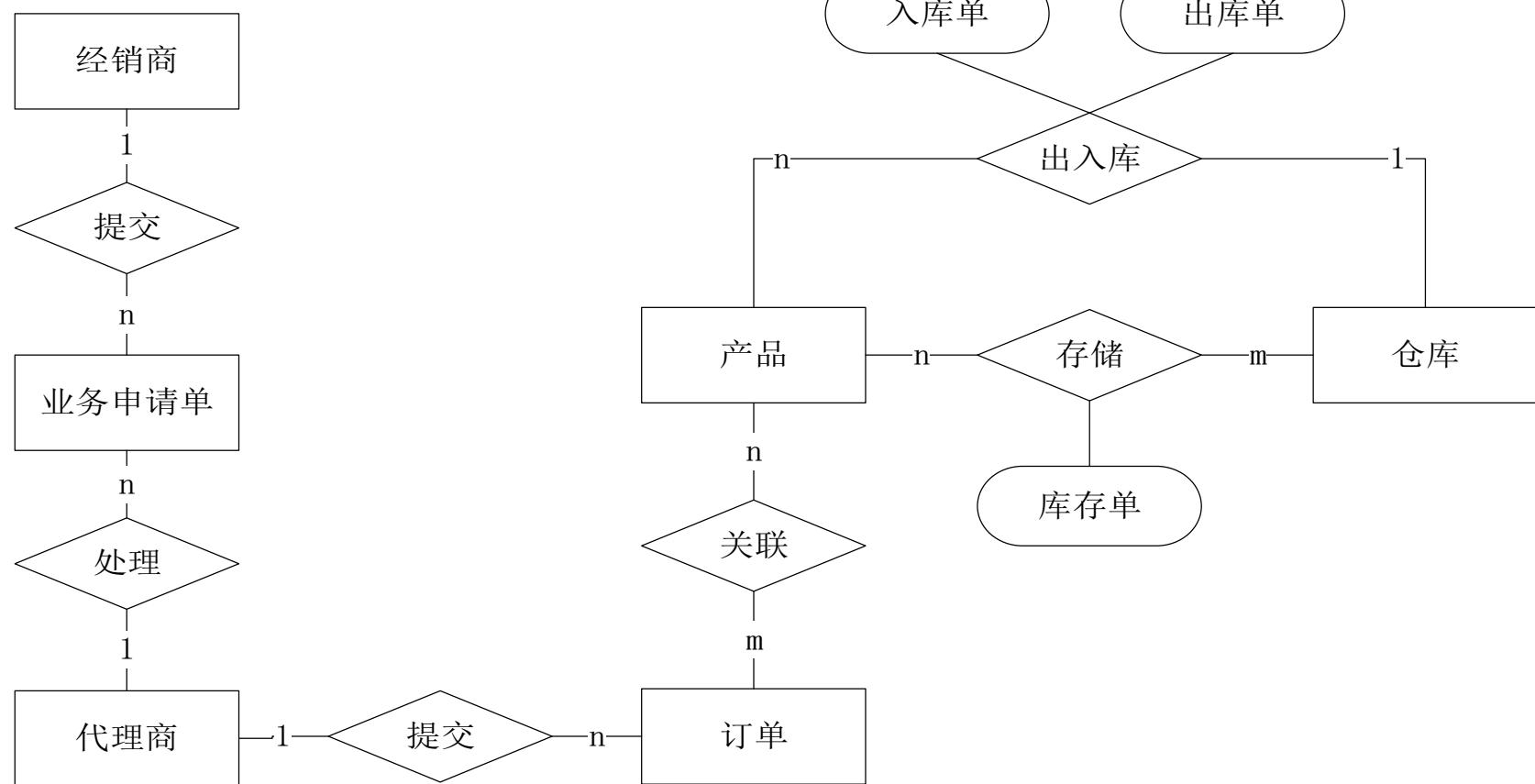
- 越级拿货
- 恶性竞争
- 秩序混乱





# 三星公司渠道管理系统E-R模型图

78





# 第七章 数据库设计

79

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 数据库的物理设计

7.6 数据库的实施和维护

7.7 小结



## 7.4 逻辑结构设计

80

### □ 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DBMS产品所支持的数据模型相符合的逻辑结构

### □ 逻辑结构设计的步骤

- 将概念结构转化为一般的关系、网状、层次模型
- 将转换来的关系、网状、层次模型向特定DBMS支持下的数据模型转换
- 对数据模型进行优化



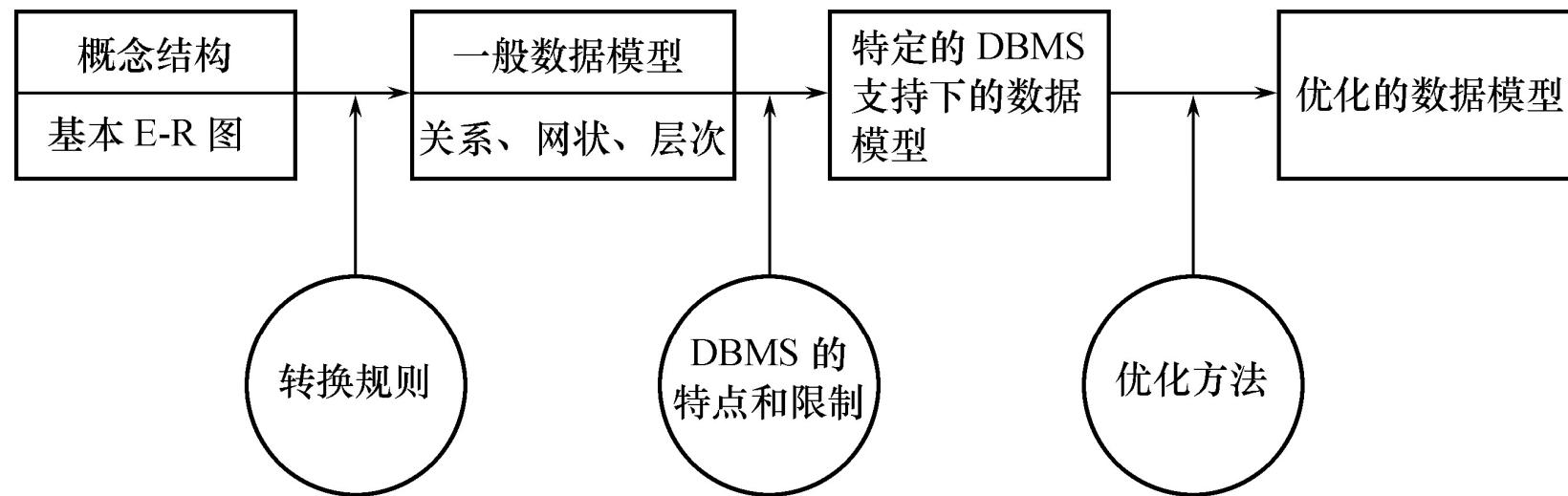
# 数据库设计各个阶段的设计描述

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）



# 逻辑结构设计(续)

82



逻辑结构设计时的3个步骤



## 7.4 逻辑结构设计

83

### 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



## 7.4.1 E-R图向关系模型的转换

84

- 转换内容
- 转换原则



# E-R图向关系模型的转换（续）

85

- E-R图向关系模型的转换要解决的问题
  - 如何将实体型和实体间的联系转换为关系模式
  - 如何确定这些关系模式的属性和码
- 转换内容
  - 将E-R图转换为关系模型：将实体、实体的属性和实体之间的联系转换为关系模式。



# E-R图向关系模型的转换（续）

86

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。

- 转换为一个独立的关系模式
- 与某一端实体对应的关系模式合并

(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。

- 转换为一个独立的关系模式
- 与n端对应的关系模式合并



## E-R图向关系模型的转换（续）

87

(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）

88

(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。

例，“讲授”联系是一个三元联系，可以将它转换为如下关系模式，其中课程号、职工号和书号为关系的组合码：

讲授（课程号, 职工号, 书号）



# E-R图向关系模型的转换（续）

89

## (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），并适当调整属性的次序



# E-R图向关系模型的转换（续）

90

注意：

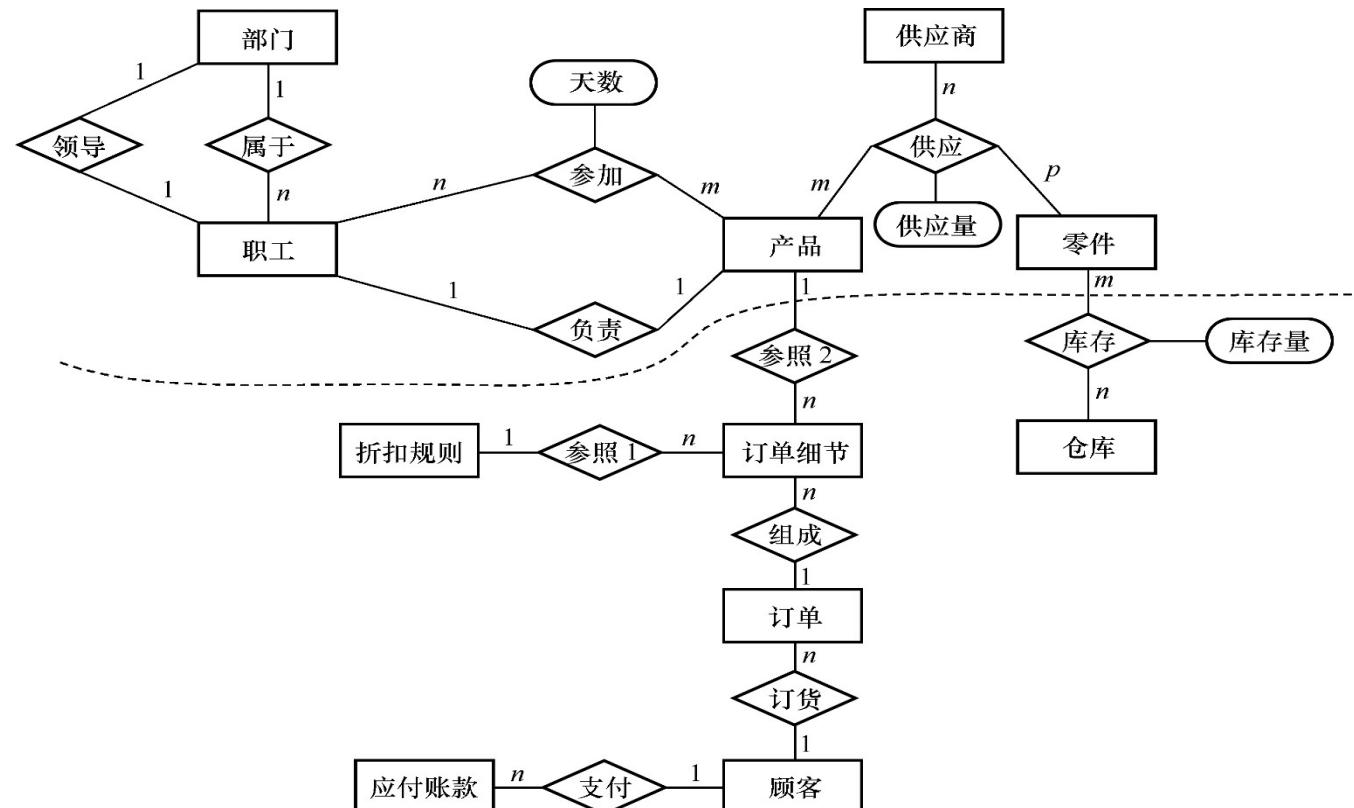
- 从理论上讲，1:1联系可以与任意一端对应的关系模式合并
- 但在一些情况下，与不同的关系模式合并效率会大不一样。因此究竟应该与哪端的关系模式合并需要依应用的具体情况而定。
- 由于连接操作是最费时的操作，所以一般应以尽量减少连接操作为目标。
- 例如，如果经常要查询某个班级的班主任姓名，则将管理联系与教师关系合并更好些。



# E-R图向关系模型的转换（续）

91

[例] 把图7.30中虚线上部的E-R图转换为关系模型





# E-R图向关系模型的转换（续）

92

[例] 把图7.30中虚线上部的E-R图转换为关系模型

□ 部门实体对应的关系模式

部门（部门号, 部门名, 经理的职工号, ...）

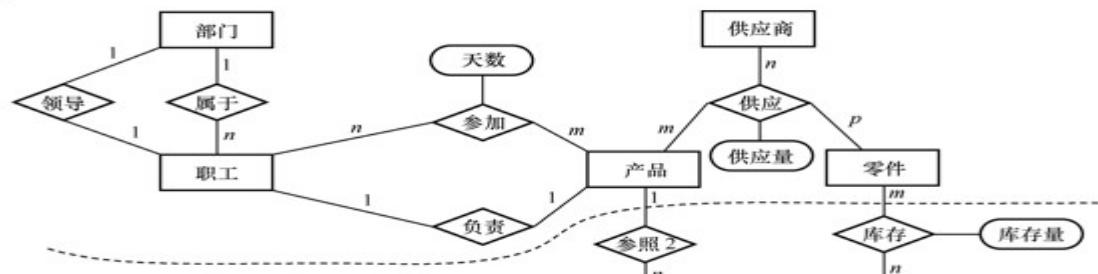
➢ 此关系模式已包含了联系“领导”所对应的关系模式

➢ 经理的职工号是关系的候选码

□ 职工实体对应的关系模式

职工（职工号、部门号, 职工名, 职务, ...）

➢ 该关系模式已包含了联系“属于”所对应的关系模式





# E-R图向关系模型的转换（续）

93

[例] 把图7.30中虚线上部的E-R图转换为关系模型 (续)

- #### □ 产品实体对应的关系模式

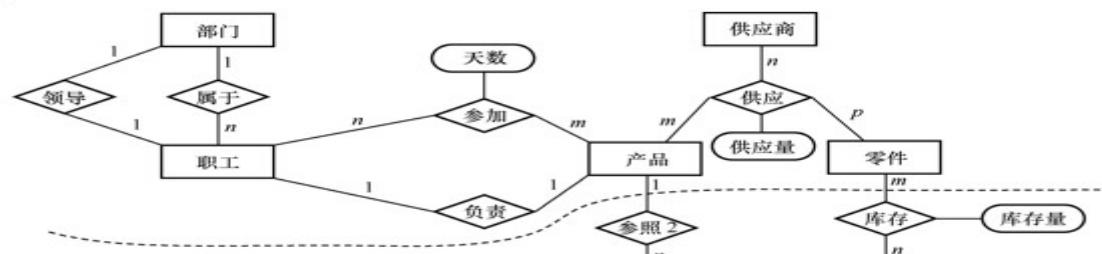
产品 (产品号, 产品名, 产品组长的职工号, ...)

- #### □ 供应商实体对应的关系模式

供应商 (供应商号, 姓名, ...)

- #### □ 零件实体对应的关系模式

零件 (零件号, 零件名, ...)





# E-R图向关系模型的转换（续）

94

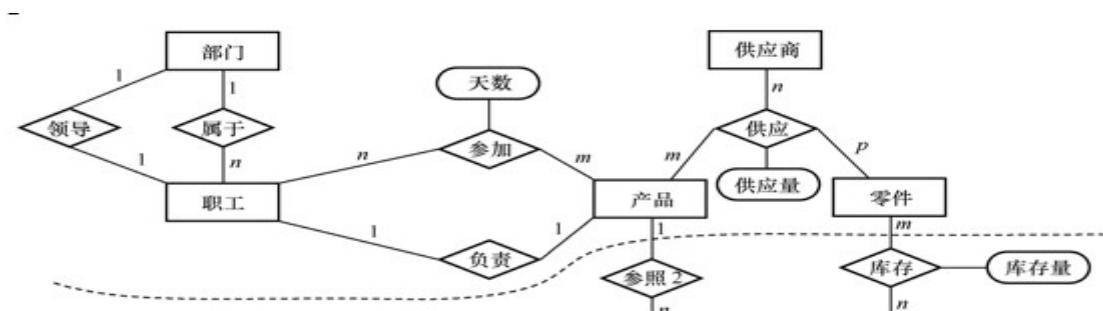
[例] 把图7.30中虚线上部的E-R图转换为关系模型（续）

- 联系“参加”所对应的关系模式

职工工作（职工号, 产品号, 工作天数, ...）

- 联系“供应”所对应的关系模式

供应（产品号, 供应商号, 零件号, 供应量）





## 7.4 逻辑结构设计

95

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

96

- 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）

97

## □ 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. 消除冗余的联系

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）

98

## 3. 确定所属范式

- 按照数据依赖的理论对关系模式逐一进行分析
- 考查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

### ❖ 关系模式规范化的基本步骤

1NF

↓ 消除非主属性对码的部分函数依赖

消除决定属性 2NF

集非码的非平凡函数依赖 ↓ 消除非主属性对码的传递函数依赖

凡函数依赖 3NF

↓ 消除主属性对码的部分和传递函数依赖

BCNF

↓ 消除非平凡且非函数依赖的多值依赖

4NF



# 数据模型的优化（续）

99

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。

**注意：**并不是规范化程度越高的关系就越优，

一般说来，第三范式就足够了



# 数据模型的优化（续）

100

例：在关系模式

学生成绩单(学号,英语,数学,语文,平均成绩)

中存在下列函数依赖：

学号→英语

学号→数学

学号→语文

学号→平均成绩

(英语, 数学, 语文)→平均成绩



# 数据模型的优化（续）

101

显然有：

学号→(英语,数学,语文) →平均成绩

因此该关系模式中存在传递函数依赖，是2NF关系

虽然平均成绩可以由其他属性推算出来，但如果应用中需要经常查询学生的平均成绩，为提高效率，仍然可保留该冗余数据，对关系模式不再做进一步分解



# 数据模型的优化（续）

102

5. 按照需求分析阶段得到的各种应用对数据处理的要求，对关系模式进行必要的分解，以提高数据操作的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）

103

## □ 水平分解

### ➤ 什么是水平分解

⑩ 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

### ➤ 水平分解的适用范围

- 满足“80/20原则”的应用
- 并发事务经常存取不相交的数据



# 数据模型的优化（续）

104

## □ 垂直分解

### ➤ 什么是垂直分解

- ⑩ 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



## 7.4 逻辑结构设计

105

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

106

□ 定义用户外模式时应该注重的问题

包括三个方面：

- (1) 使用更符合用户习惯的别名
- (2) 针对不同级别的用户定义不同的View，以满足系统对安全性的要求。
- (3) 简化用户对系统的使用



# 设计用户子模式（续）

107

**[例]** 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）

- 顾客视图中只包含允许顾客查询的属性
- 销售部门视图中只包含允许销售部门查询的属性
- 生产领导部门则可以查询全部产品数据
- 可以防止用户非法访问不允许他们查询的数据，保证系统的安全性



# 作业16—数据库设计

108

- 1: 科大有若干学院，每个学院有若干系和实验室，每个实验室有若干教师，其中有的教授和副教授每人各带若干研究生，每个系有若干学生，每个学生选修若干门课，每门课可由若干学生选修，每门课程也可由多个教师讲授，每个教师可以讲授多门课程。请用E-R图画出科大的概念模型。
2. 试把习题 1 中的E-R图转化成关系模型。
3. 试用规范化理论中的有关范式的概念分析习题 2 设计的关系模型中各个关系模式的候选码，它们属于第几范式？会产生什么更新异常？



# 第七章 数据库设计

109

## 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 数据库的物理设计

## 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

110

- 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环境的物理结构的过程，就是数据库的物理设计



# 数据库的物理设计(续)

111

## □ 数据库物理设计的步骤

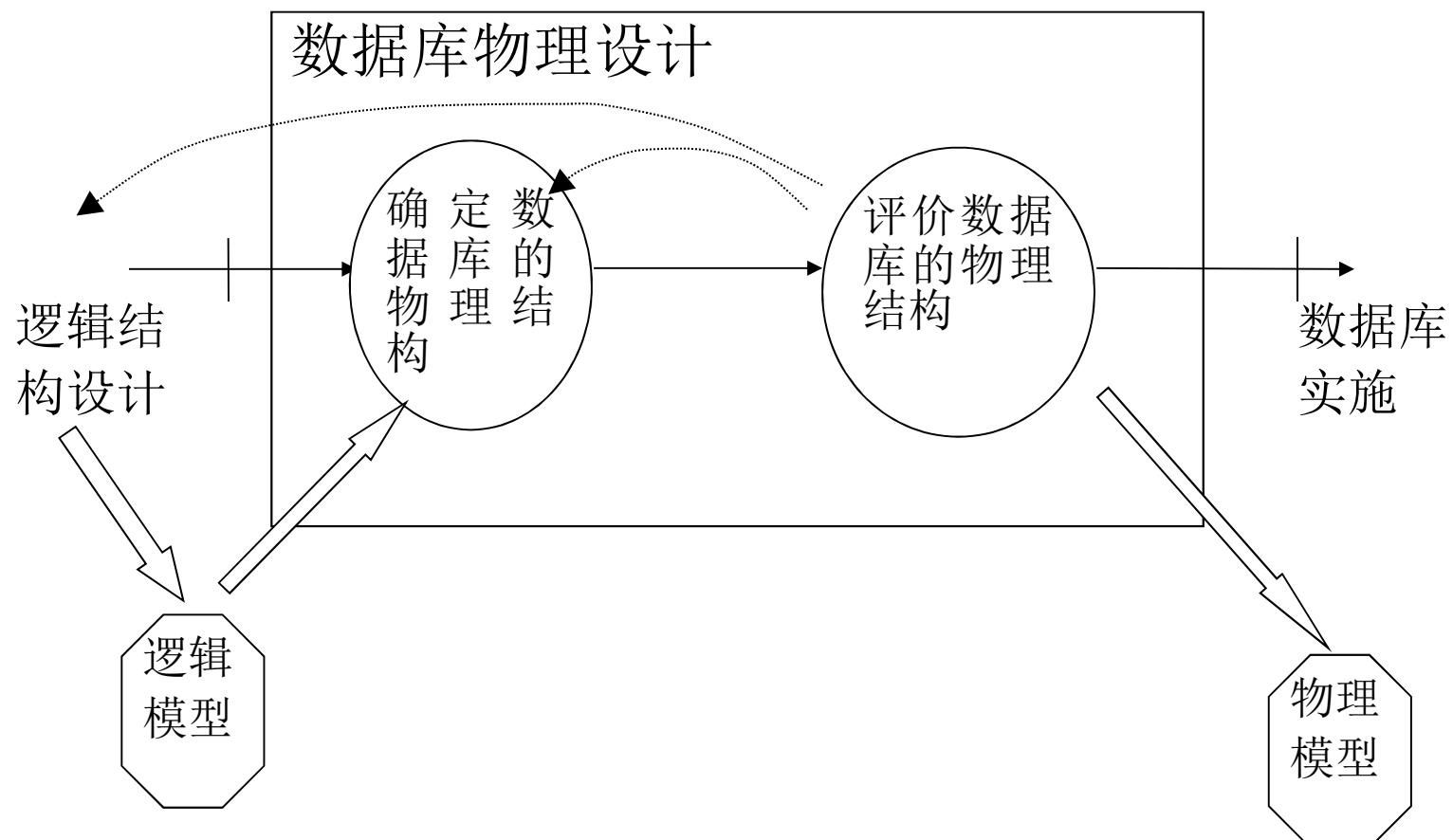
- 确定数据库的物理结构，在关系数据库中主要指存取方法和存储结构
- 对物理结构进行评价，评价的重点是时间和空间效率

如果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)

112





## 7.5 数据库的物理设计

113

### 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

114

- 设计物理数据库结构的准备工作
- 对要运行的事务进行详细分析，获得选择物理数据库设计所需参数
- 充分了解所用RDBMS的内部特征，特别是系统提供的存取方法和存储结构



# 数据库的物理设计的内容和方法（续）

115

- 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



## 数据库的物理设计的内容和方法（续）

116

- 选择物理数据库设计所需参数(续)
  - 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



## 数据库的物理设计的内容和方法（续）

117

- 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

118

### 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

119

- 数据库系统是多用户共享的系统，对同一个关系要建立多条存取路径才能满足多用户的多种应用要求
- 物理设计的任务之一就是要确定选择哪些存取方法，即建立哪些存取路径



# 关系模式存取方法选择（续）

120

- DBMS常用存取方法
  - 索引方法
    - 目前主要是B+树索引方法
    - 经典存取方法，使用最普遍
  - 聚簇(Cluster)方法
  - HASH方法



# 一、索引存取方法的选择

121

- 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



# 索引存取方法的选择（续）

122

- 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中出现，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
- 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



## 二、聚簇存取方法的选择

123

### □ 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（称为聚簇码）上具有相同值的元组集中存放在连续的物理块称为聚簇



# 聚簇存取方法的选择（续）

124

## □ 聚簇的用途

### □ 1. 大大提高按聚簇码进行查询的效率

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。

- 信息系的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）

125

## □ 2. 节省存储空间

- 聚簇以后，聚簇码相同的元组集中在一起了，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）

126

- 聚簇的局限性
  - 1. 聚簇只能提高某些特定应用的性能
  - 2. 建立与维护聚簇的开销相当大
    - 对已有关系建立聚簇，将导致关系中元组移动其物理存储位置，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移动



# 聚簇存取方法的选择（续）

127

- 聚簇的适用范围
- 1. 既适用于单个关系独立聚簇，也适用于多个关系组合聚簇

例：假设用户经常要按系别查询学生成绩单，这一查询涉及学生关系和选修关系的连接操作，即需要按学号连接这两个关系，为提高连接操作的效率，可以把具有相同学号值的学生元组和选修元组在物理上聚簇在一起。这就相当于把多个关系按“预连接”的形式存放，从而大大提高连接操作的效率。



# 聚簇存取方法的选择（续）

128

- 2. 当通过聚簇码进行访问或连接是该关系的主要应用，与聚簇码无关的其他访问很少或者是次要的时候，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等子句或短语时，使用聚簇特别有利，可以省去对结果集的排序操作



# 聚簇存取方法的选择（续）

129

## □ 设计候选聚簇

- 对经常在一起进行连接操作的关系可以建立聚簇
- 如果一个关系的一组属性经常出现在相等比较条件下，则该单个关系可建立聚簇
- 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。即对应每个聚簇码值的平均元组数不太少。太少了，聚簇的效果不明显



# 聚簇存取方法的选择（续）

130

## □ 优化聚簇设计

- 从聚簇中删除经常进行全表扫描的关系；
- 从聚簇中删除更新操作远多于连接操作的关系；
- 不同的聚簇中可能包含相同的关系，一个关系可以在某一个聚簇中，但不能同时加入多个聚簇
  - 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



### 三、HASH存取方法的选择

131

- 选择HASH存取方法的规则
  - 当一个关系满足下列两个条件时，可以选择HASH存取方法
    - 该关系的属性主要出现在等值连接条件中或主要出现在相等比较选择条件下
    - 该关系的大小可预知，而且不变；  
或  
该关系的大小动态改变，但所选用的DBMS提供了动态HASH存取方法



## 7.5 数据库的物理设计

132

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

133

- 确定数据库物理结构的内容
  - 1. 确定数据的存放位置和存储结构
    - 关系
    - 索引
    - 聚簇
    - 日志
    - 备份
  - 2. 确定系统配置



# 1. 确定数据的存放位置

134

- 确定数据存放位置和存储结构的因素
  - 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

例：消除一切冗余数据虽能够节约存储空间和减少维护代价，但往往会导致检索代价的增加

必须进行权衡，选择一个折中方案



# 确定数据的存放位置（续）

135

- 基本原则
- 根据应用情况将
  - 易变部分与稳定部分分开存放
  - 存取频率较高部分与存取频率较低部分，分开存放



# 确定数据的存放位置（续）

136

例：

- 数据库数据备份、日志文件备份等由于只在故障恢复时才使用，而且数据量很大，可以考虑存放在磁带上
- 如果计算机有多个磁盘或磁盘阵列，可以考虑将表和索引分别放在不同的磁盘上，在查询时，由于磁盘驱动器并行工作，可以提高物理I/O读写的效率



# 确定数据的存放位置（续）

137

例（续）：

- 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

138

- DBMS产品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
  - 使用的缓冲区长度、个数
  - 存储分配参数
  - .....



## 7.5 数据库的物理设计

139

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

140

- 评价内容
- 对数据库物理设计过程中产生的多种方案进行细致的评价，从中选择一个较优的方案作为数据库的物理结构



# 评价物理结构(续)

141

- 评价方法（完全依赖于所选用的DBMS）
- 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
- 如果该结构不符合用户需求，则需要修改设计



# 第七章 数据库设计

142

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 数据库的物理设计

7.6 数据库的实施和维护

7.7 小结



# 7.6数据库实施和维护

143

## 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



## 7.6.1 数据的载入和应用程序的调试

144

- 数据的载入
- 应用程序的编码和调试



# 数据的载入

145

- 数据库结构建立好后，就可以向数据库中装载数据了。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



# 应用程序的编码和调试

146

- 数据库应用程序的设计应该与数据设计并行进行
- 在组织数据入库的同时还要调试应用程序



## 7.6数据库实施和维护

147

### 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

148

- 在原有系统的数据有一小部分已输入数据库后，就可以开始对数据库系统进行联合调试，称为数据库的试运行
- 数据库试运行主要工作包括：
  - 1) 功能测试
    - 实际运行数据库应用程序，执行对数据库的各种操作，测试应用程序的功能是否满足设计要求
    - 如果不满足，对应用程序部分则要修改、调整，直到达到设计要求
  - 2) 性能测试
    - 测量系统的性能指标，分析是否达到设计目标
    - 如果测试的结果与设计目标不符，则要返回物理设计阶段，重新调整物理结构，修改系统参数，某些情况下甚至要返回逻辑设计阶段，修改逻辑结构



# 数据库的试运行（续）

149

强调两点：

- 分期分批组织数据入库
- 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大，费时、费力，所以应分期分批地组织数据入库
- 先输入小批量数据供调试用
  - 待试运行基本合格后再大批量输入数据
  - 逐步增加数据量，逐步完成运行评价



# 数据库的试运行（续）

150

- 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，误操作也无法避免
  - 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 数据库实施和维护

151

## 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 数据库的运行与维护

152

- 数据库试运行合格后，数据库即可投入正式运行。
- 数据库投入运行标志着开发任务的基本完成和维护工作的开始
- 对数据库设计进行评价、调整、修改等维护工作是一个长期的任务，也是设计工作的继续和提高。
  - 应用环境在不断变化
  - 数据库运行过程中物理存储会不断变化



# 数据库的运行与维护（续）

153

- 在数据库运行阶段，对数据库经常性的维护工作主要是由DBA完成的，包括：
  1. 数据库的转储和恢复
  2. 数据库的安全性、完整性控制
  3. 数据库性能的监督、分析和改进
  4. 数据库的重组织和重构造



# 数据库的运行与维护（续）

154

- 数据库的重组织和重构造
  - 重组织的形式
    - 全部重组织
      - 索引重组、单表重组、表空间的重组
    - 部分重组织
      - 只对频繁增、删的表进行重组织
  - 重组织的目标
    - 提高系统性能



# 数据库的运行与维护（续）

155

## □ 重组织的工作

### ➤ 按原设计要求

⑩ 重新安排存储位置

⑩ 回收垃圾

⑩ 减少指针链

### ➤ 数据库的重组织不会改变原设计的数据逻辑结构和物理结构



# 数据库运行与维护（续）

156

## □ 数据库重构

根据新环境调整数据库的模式和内模式

- 增加新的数据项
- 改变数据项的类型
- 改变数据库的容量
- 增加或删除索引
- 修改完整性约束条件



# 第七章 数据库设计

157

- 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 数据库的物理设计
- 7.6 数据库的实施和维护
- 7.7 小结



## 7.7 小结

158

- 数据库的设计过程
  - 需求分析
  - 概念结构设计
  - 逻辑结构设计
  - 物理设计
  - 实施和维护



## 小结（续）

159

- 数据库各级模式的形成
- 数据库的各级模式是在设计过程中逐步形成的
- 需求分析阶段综合各个用户的应用需求（现实世界的需求）
- 概念设计阶段形成独立于机器特点、独立于各个DBMS产品的**概念模式**（信息世界模型），用E-R图来描述



## 小结（续）

160

- 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（VIEW）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**