



2024年春季学期

1

# 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](mailto:huangzhy@ustc.edu.cn)

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



## 7.3 概念结构设计

113

### 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

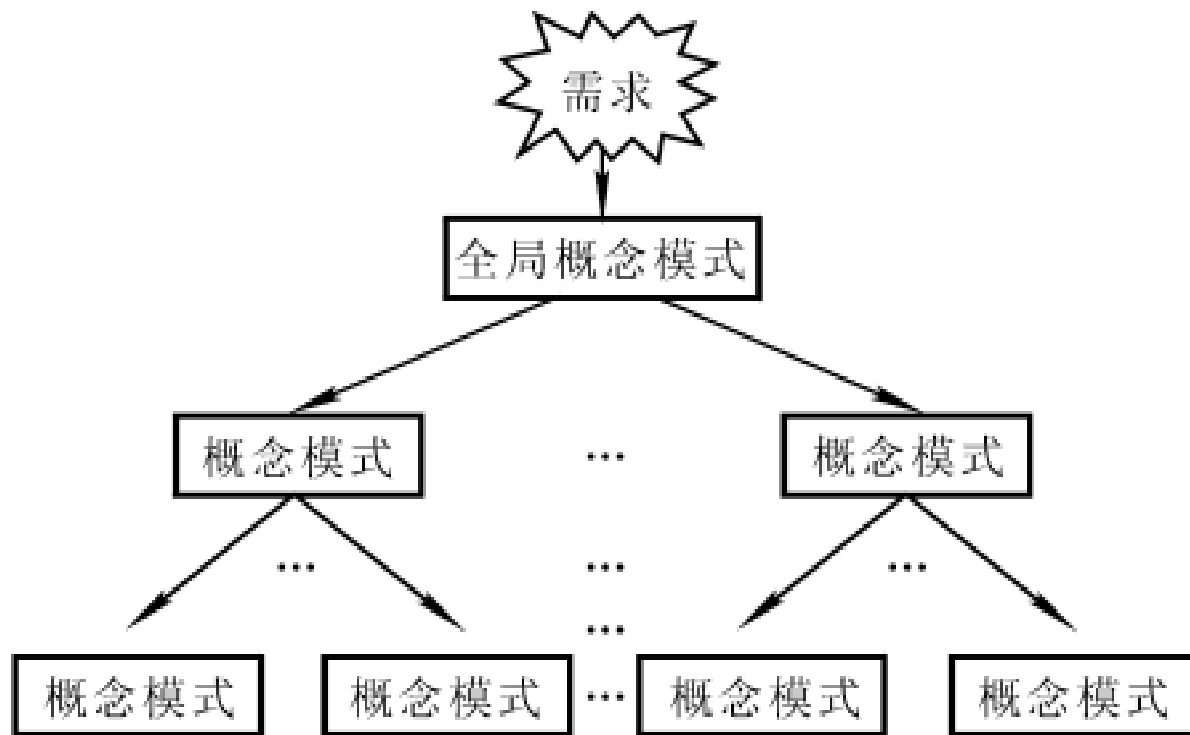


# 7.3.5 概念结构设计

- 概念结构设计方法与步骤（四类方法）

- 1. 自顶向下

- 首先定义全局概念结构的框架，然后逐步细化



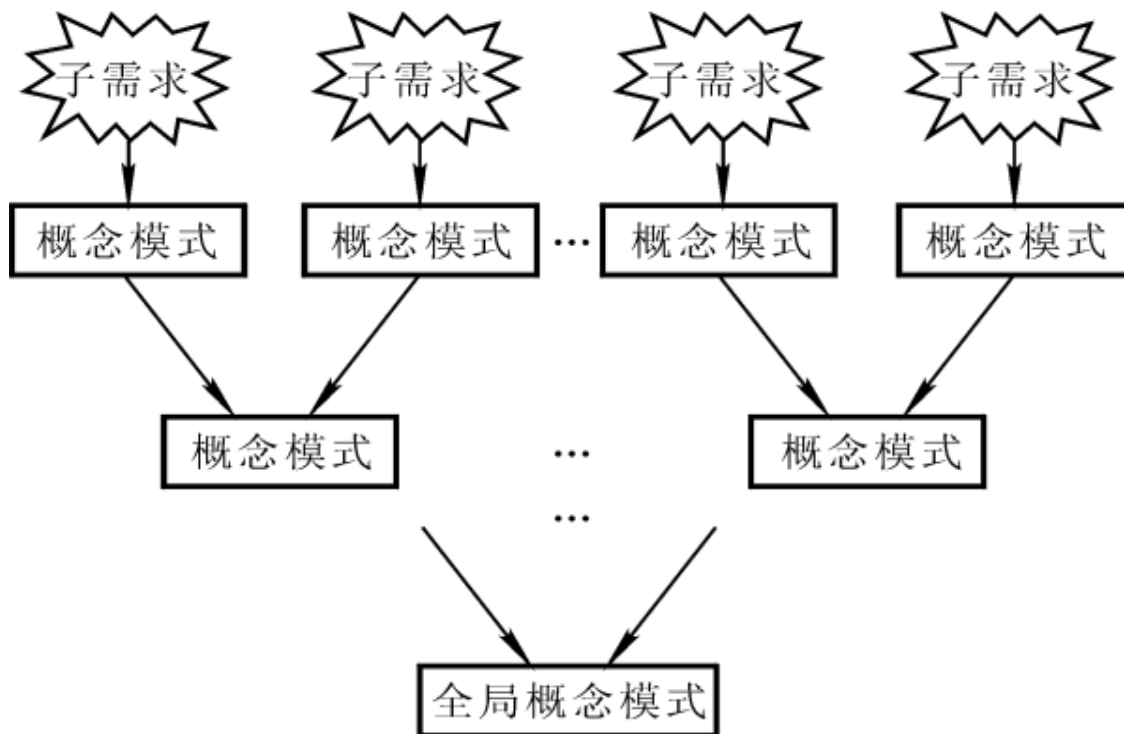


## 7.3.5 概念结构设计

### □ 概念结构设计方法与步骤（四类方法）

#### □ 2. 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



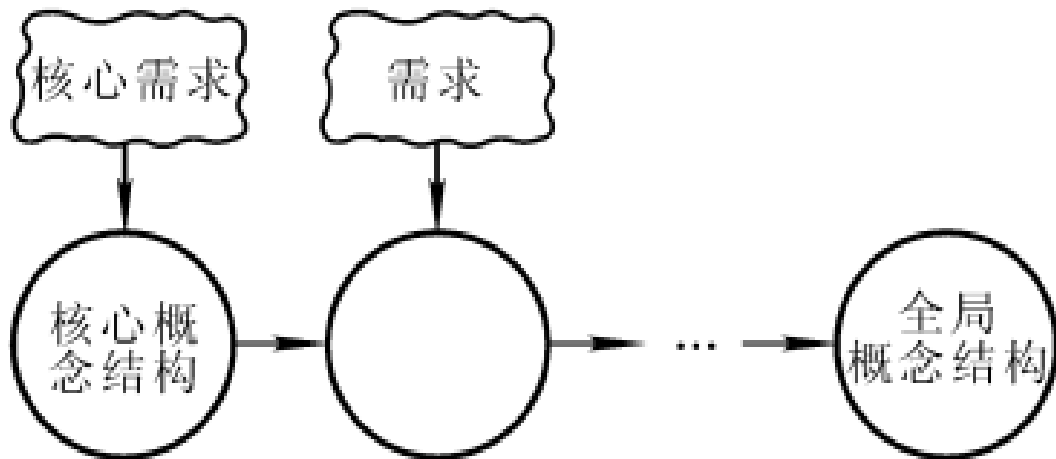


## 7.3.5 概念结构设计

- 概念结构设计方法与步骤（四类方法）

- 3. 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



## 7.3.5 概念结构设计

117

### □ 概念结构设计方法与步骤（四类方法）

#### □ 4. 混合策略

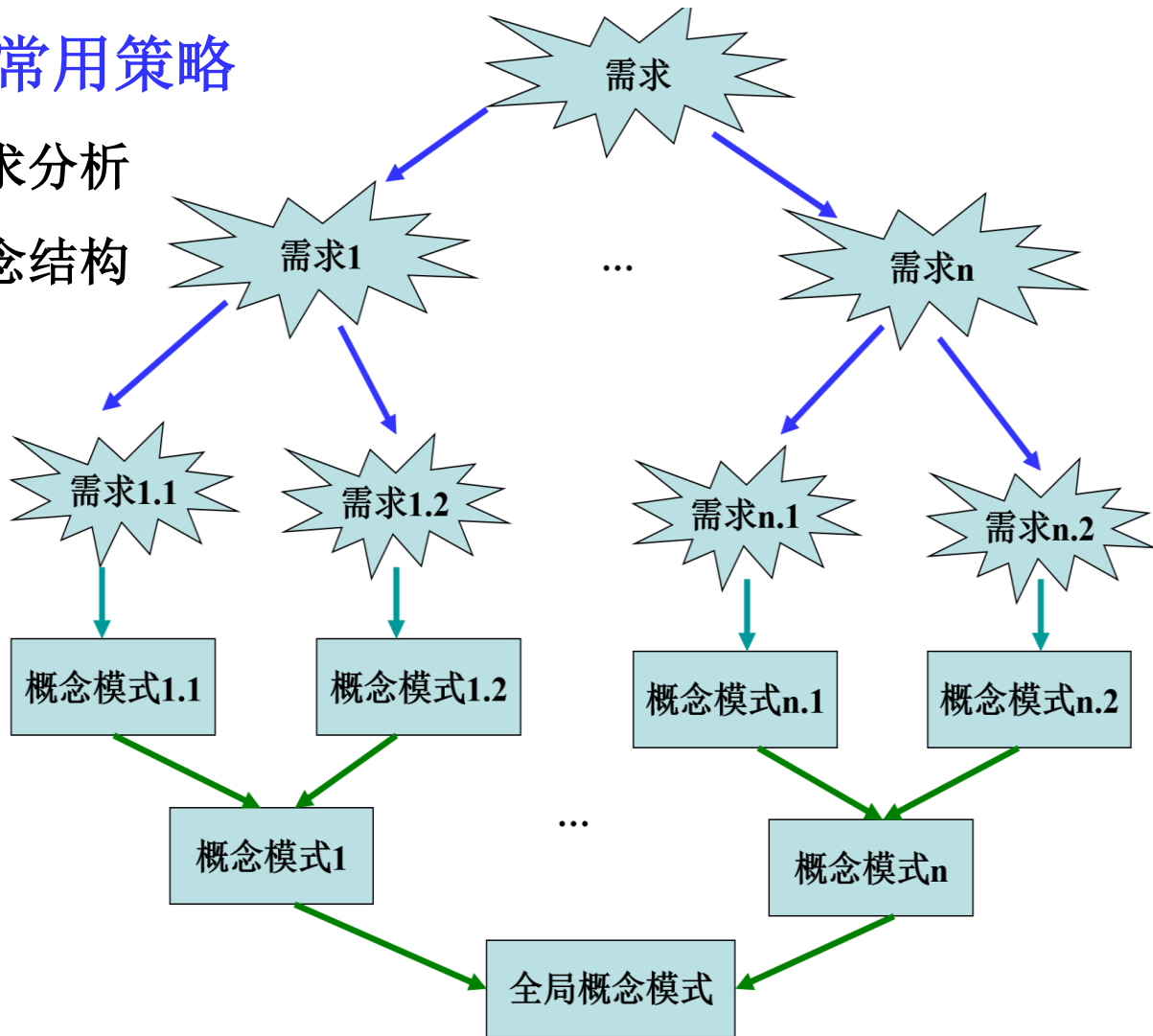
- 将自顶向下和自底向上相结合，用自顶向下策略设计一个全局概念结构的框架，以它为骨架集成由自底向上策略中设计的各局部概念结构。



# 7.3.5 概念结构设计

## 开发大型系统时的常用策略

- 自顶向下地进行需求分析
- 自底向上地设计概念结构



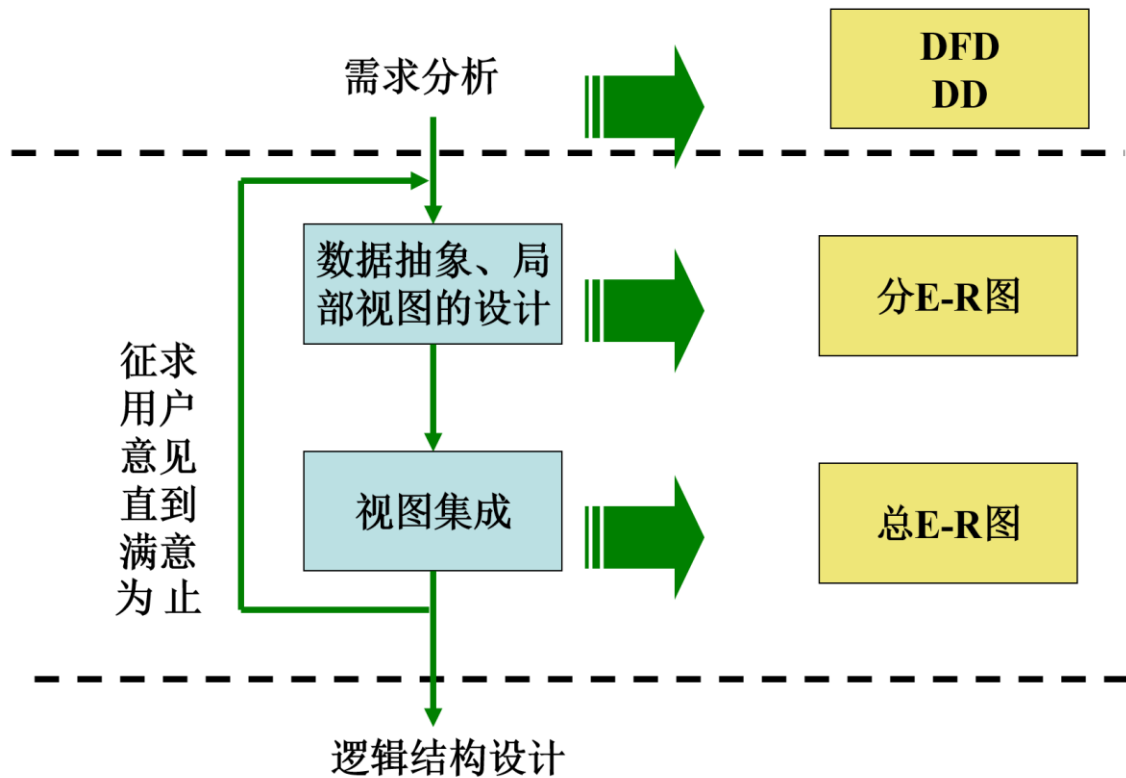


# 概念结构设计的方法与步骤（续）

## □ 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，形成子系统，即分E-R图

第2步：集成局部视图，得到全局概念结构，即总E-R图



DD数据字典  
DFD用户需求规格说明书





## 7.3.5 概念结构设计

120

- 第1步：局部设计，形成（子系统）分 E-R图
  - 问题：如何确定实体和属性？
    - 没有形式上可以截然划分的界限
- 1. 实体与属性的划分原则
  - 初步：现实世界应用环境有大体划分
  - 调整原则：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

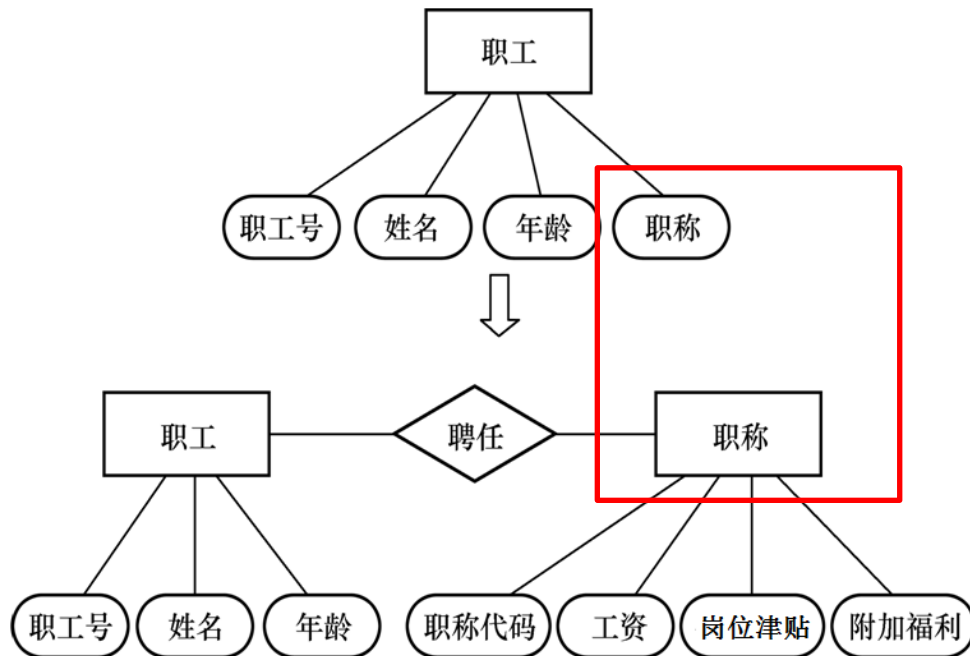


# 概念结构设计（续）

[例1] 职工是一个实体：

职工号、姓名、年龄是职工的属性，如何设计“职称”？

- 职称如果没有与工资、福利挂钩，无进一步描述的特点，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

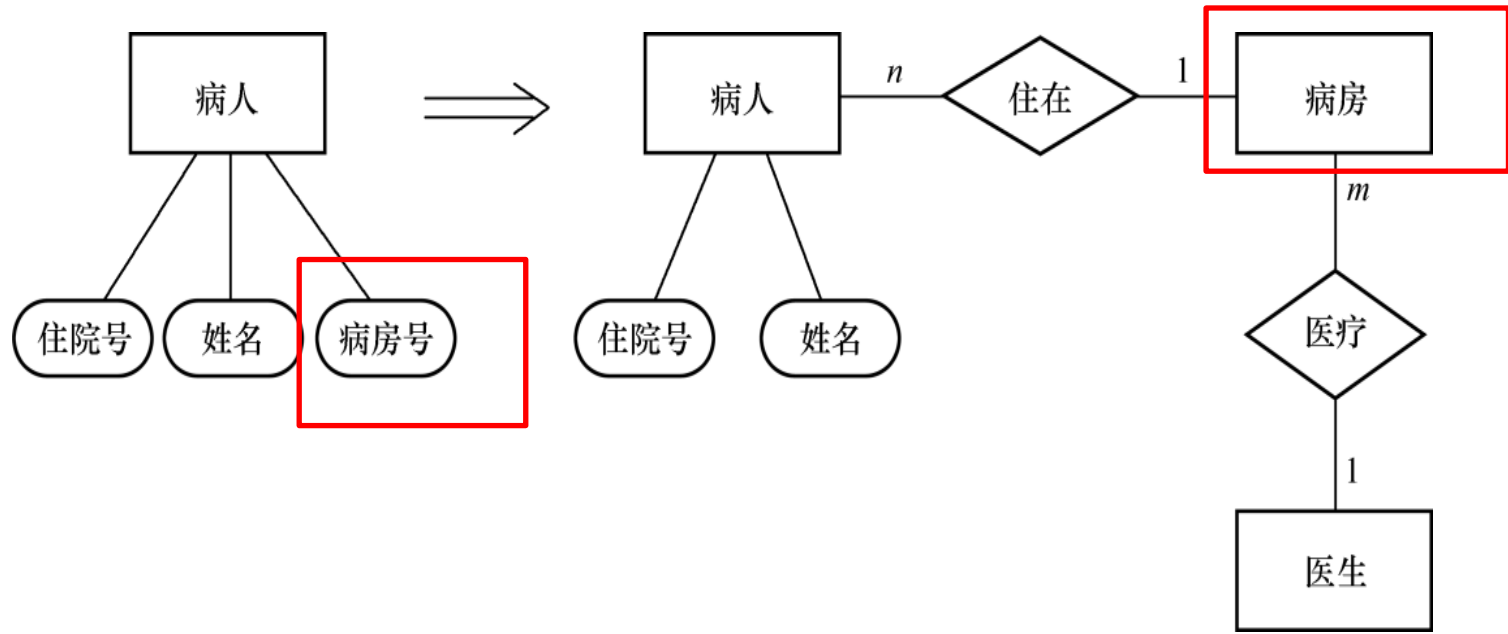




# 概念结构设计（续）

[例2] 在医院中，如何设计“病房号”？

- 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

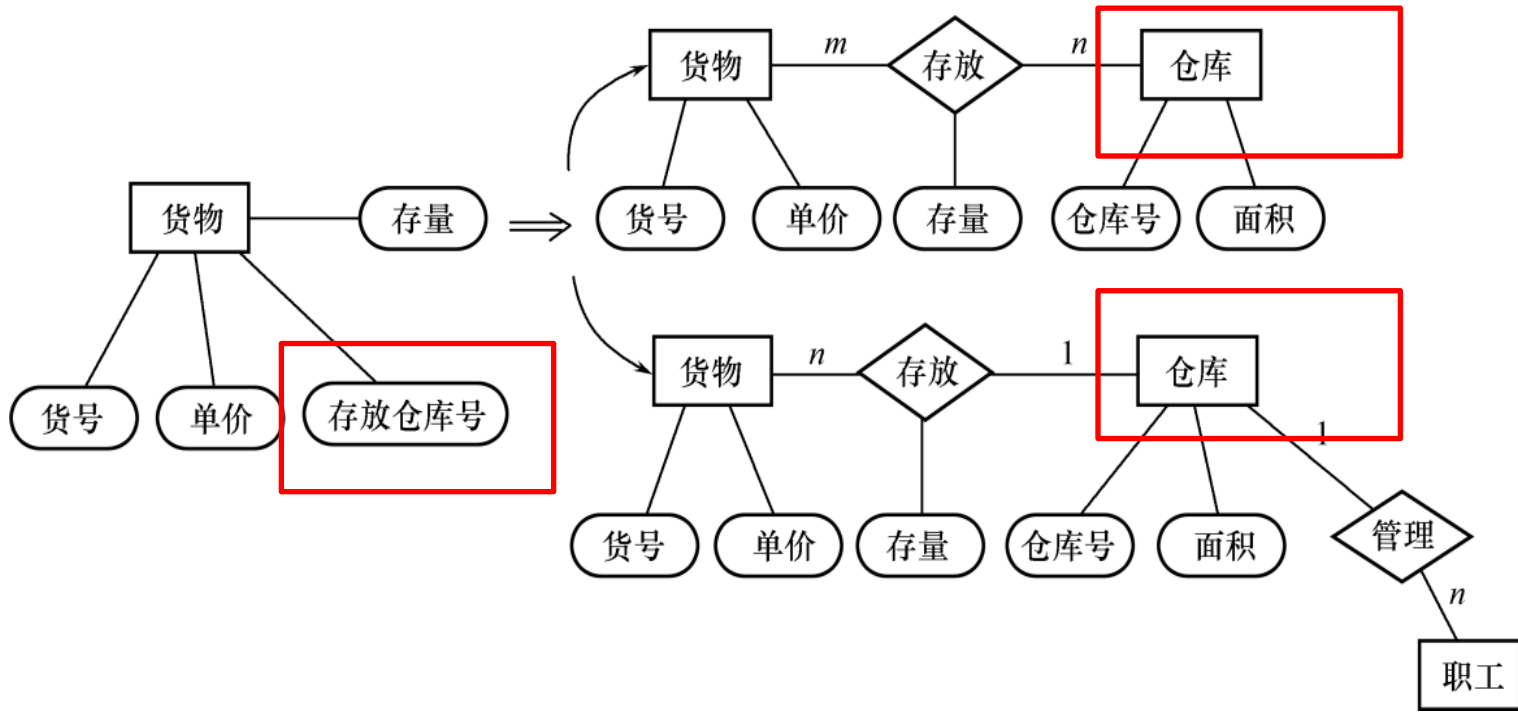




# 概念结构设计（续）

## [例3] 货物实体，如何设计“仓库”？

- 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。

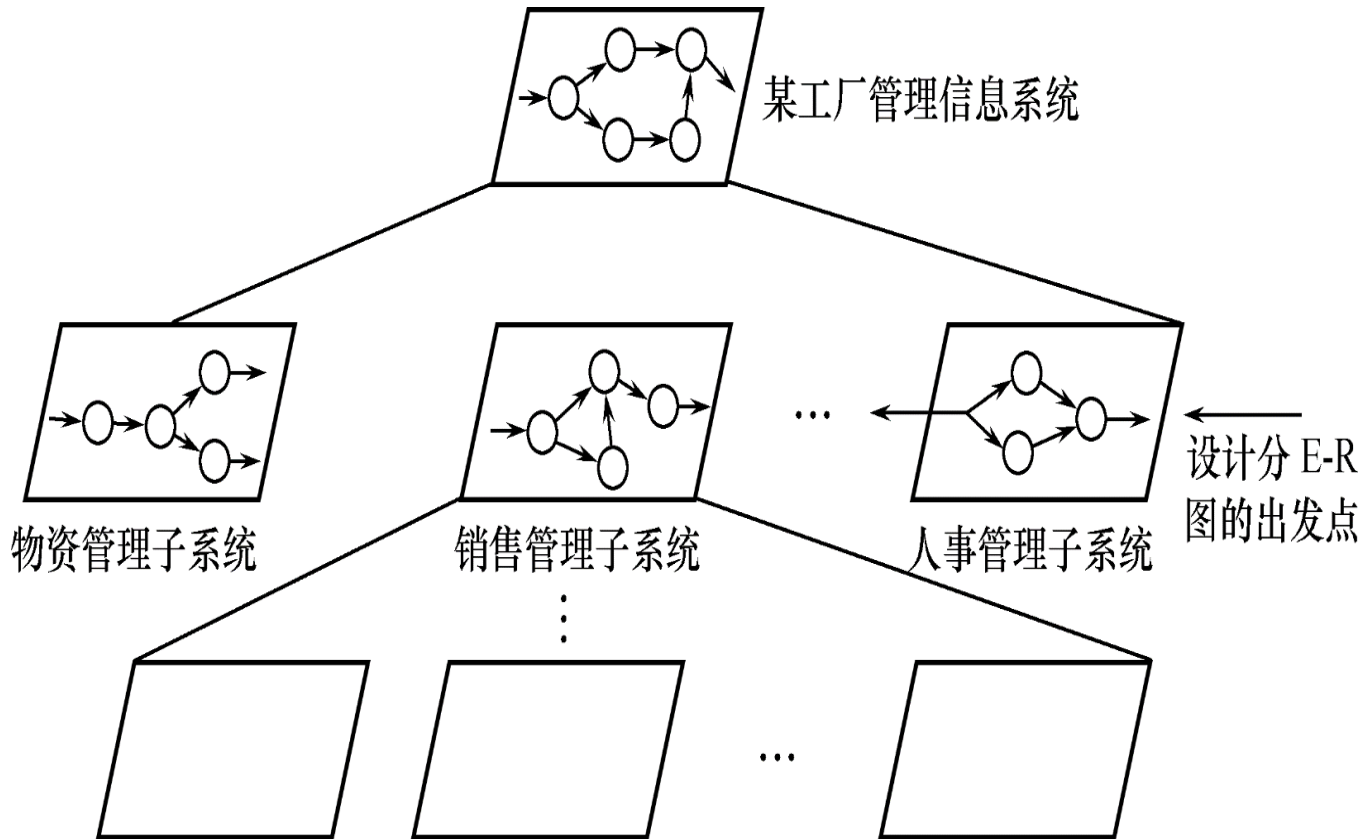




# 概念结构设计 (续)

## [例7.1] 工厂管理信息系统：多个子系统

物资管理、**销售管理**、劳动人事管理等





# 概念结构设计（续）

125

工厂管理信息系统：物资管理、销售管理、劳动人事管理等

## [例7.1] 销售管理子系统E-R图的设计

### □ 主要功能：

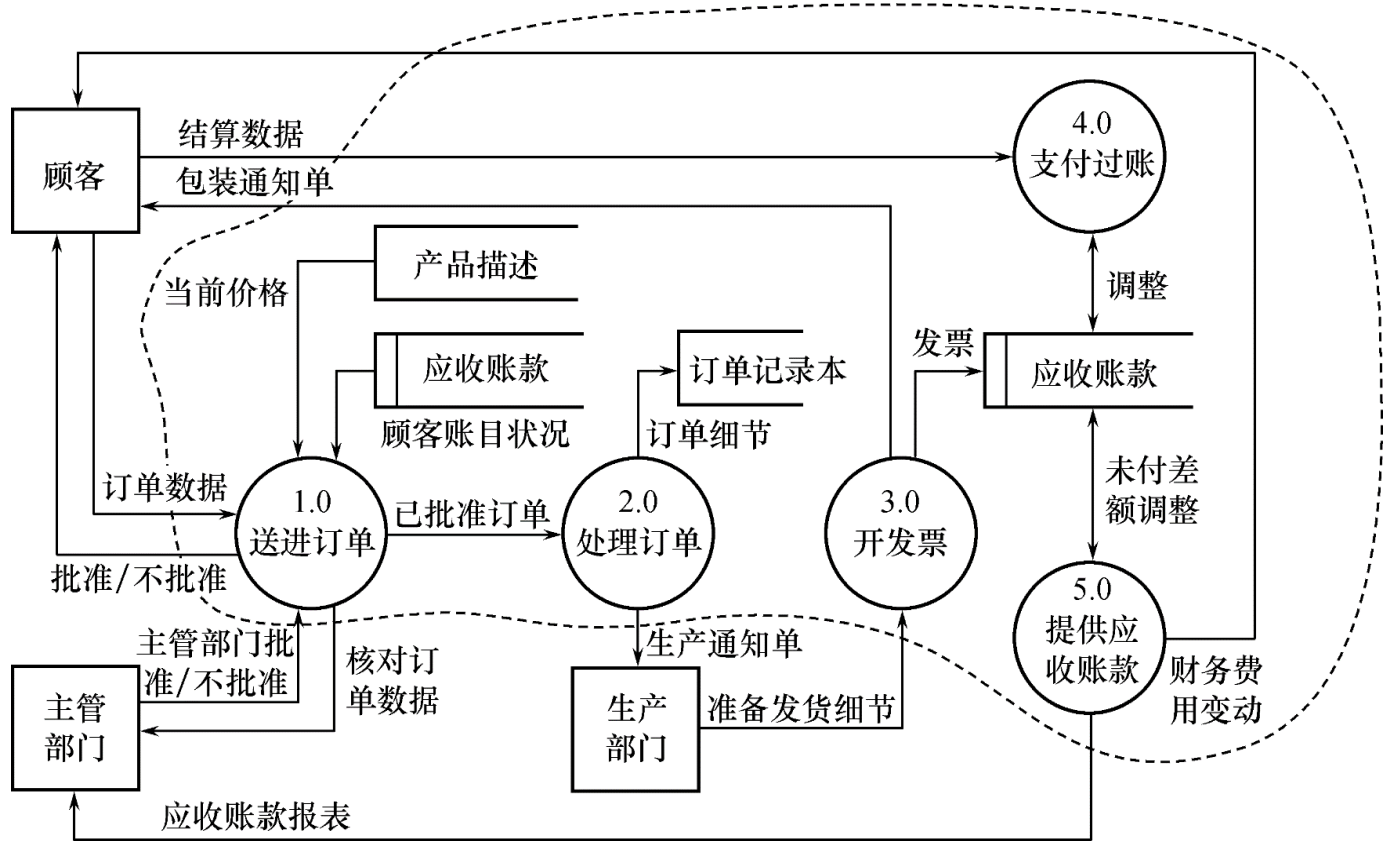
- 处理顾客和销售员送来的订单
- 工厂是根据订货安排生产的
- 交出货物同时开出发票
- 收到顾客付款后，根据发票存根和信贷情况进行应收款处理



## 2. 逐一设计分E-R图（续）（补充）

### 销售管理子系统

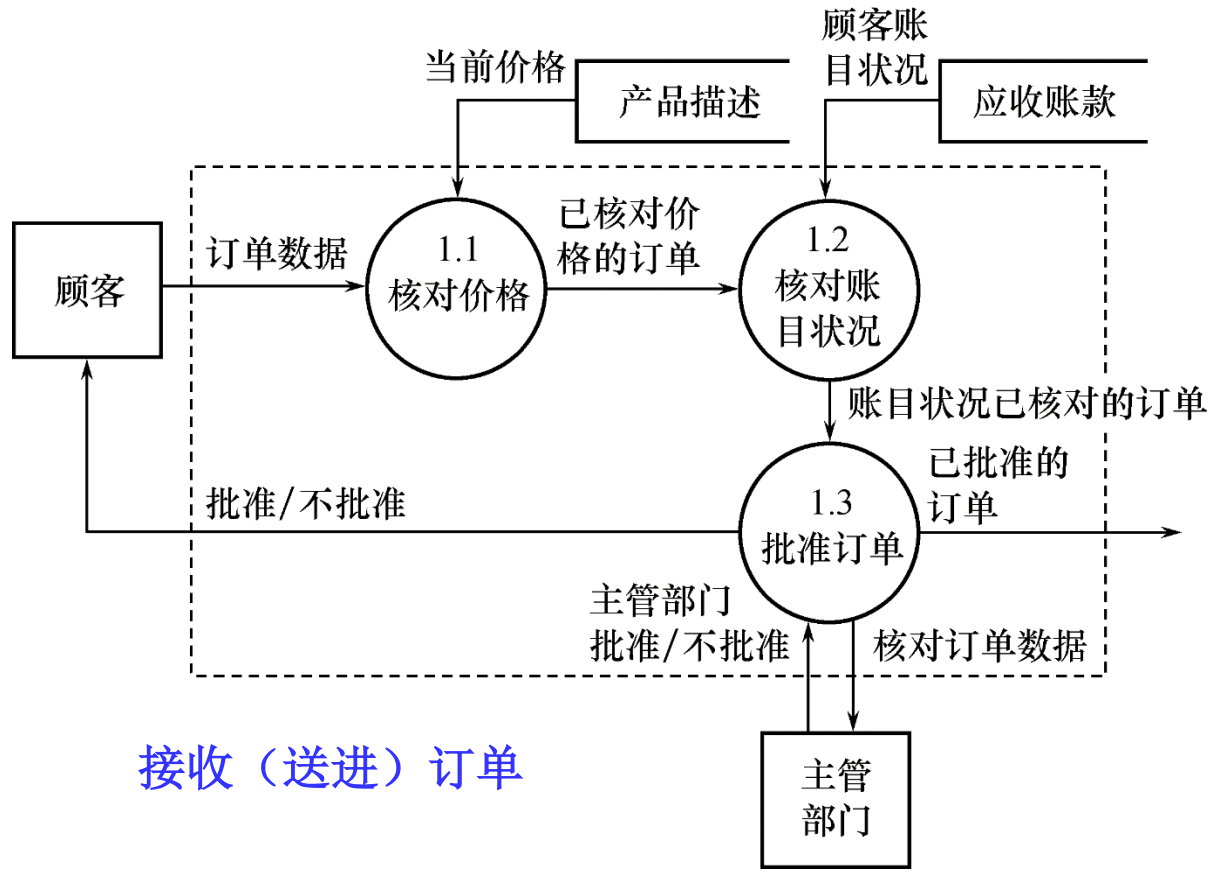
- 第一层数据流图，虚线部分划出了系统边界
  - 接收订单、处理订单、开发票、支付过账，提供应收账款





## 2. 逐一设计分E-R图（续）（补充）

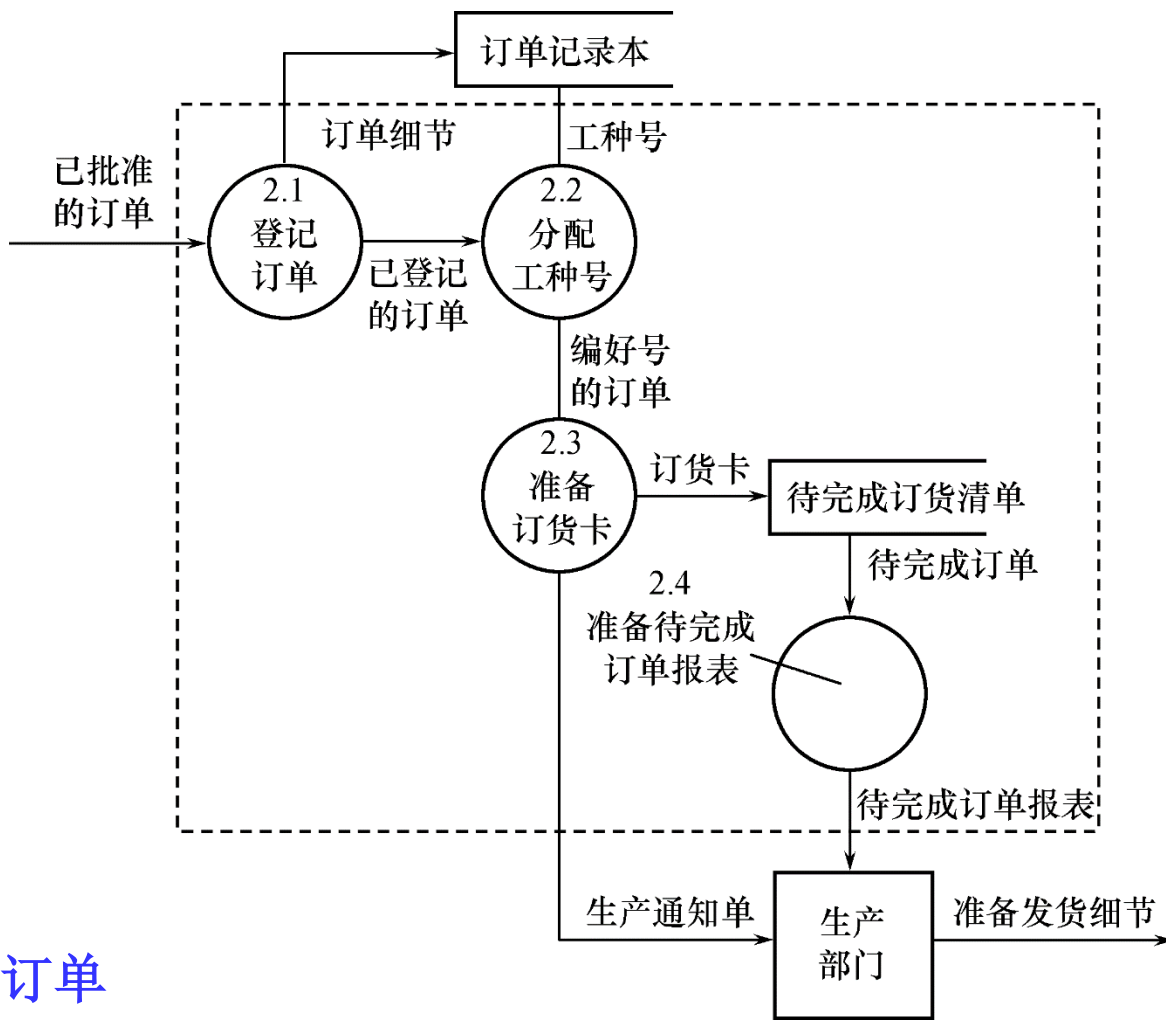
- 上图中把系统功能又分为4个子系统
- 下面四个图是第二层数据流图







## 2. 逐一设计分E-R图（续）（补充）

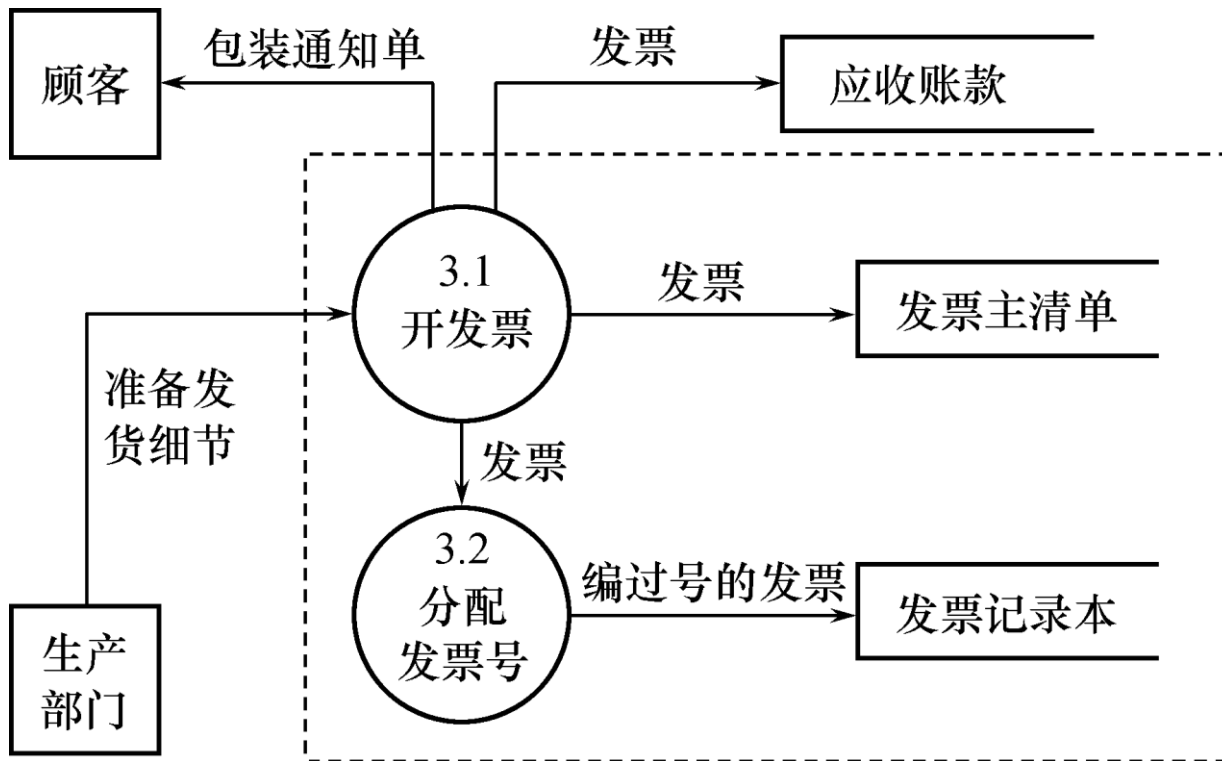


处理订单



## 2. 逐一设计分E-R图（续）（补充）

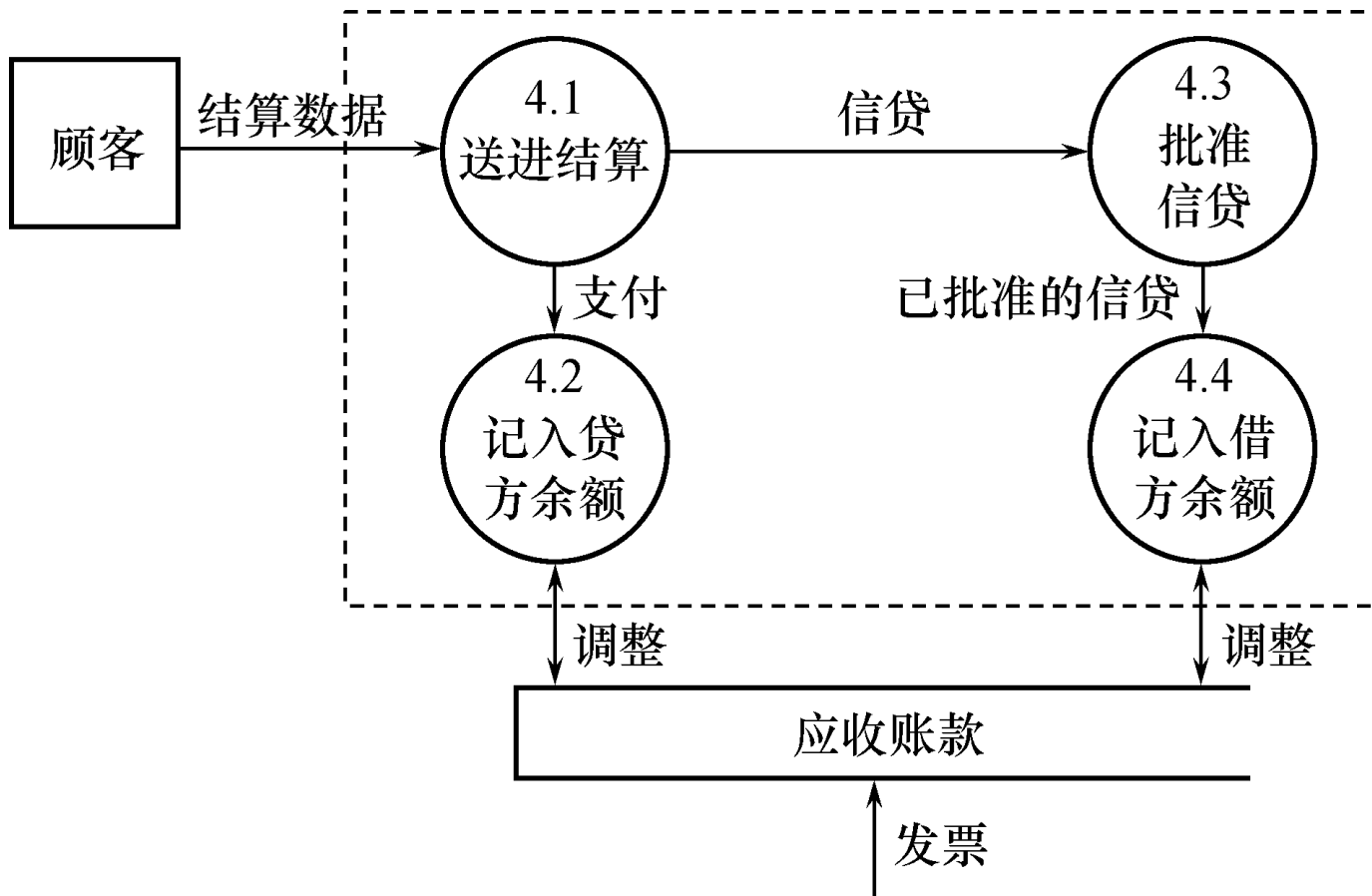
129



开发票



## 2. 逐一设计分E-R图（续）（补充）



支付过账



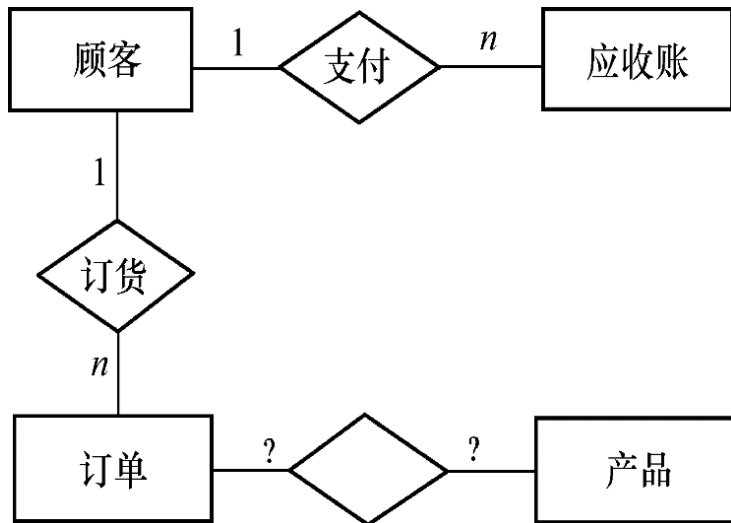
# 概念结构设计（续）

工厂管理信息系统：物资管理、销售管理、劳动人事管理等

## [例7.1] 销售管理子系统E-R图的设计

### ● 1. 画草图7.22

- 需求分析：接收订单、处理订单、开发票、支付过账
- 数据结构：订单、顾客，应收账款目，产品





# 概念结构设计（续）

132

- 2. 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**，E-R图进行了如下**调整**：图7.23
  - （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（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
  - （2）**原订单和产品的联系实际上是订单细节和产品的联系**。**增加联系—参照2**：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**增加实体—“折扣规则”**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。

订单细节上升实体：  
一张订单可订多个产品，  
可以由订单号、细节  
信息和产品号组成

增加折扣实体，和参照：  
工厂对大宗订货给予优惠

增加参照：  
原订单和产品的联系  
实际上是订单细  
节和产品的联系

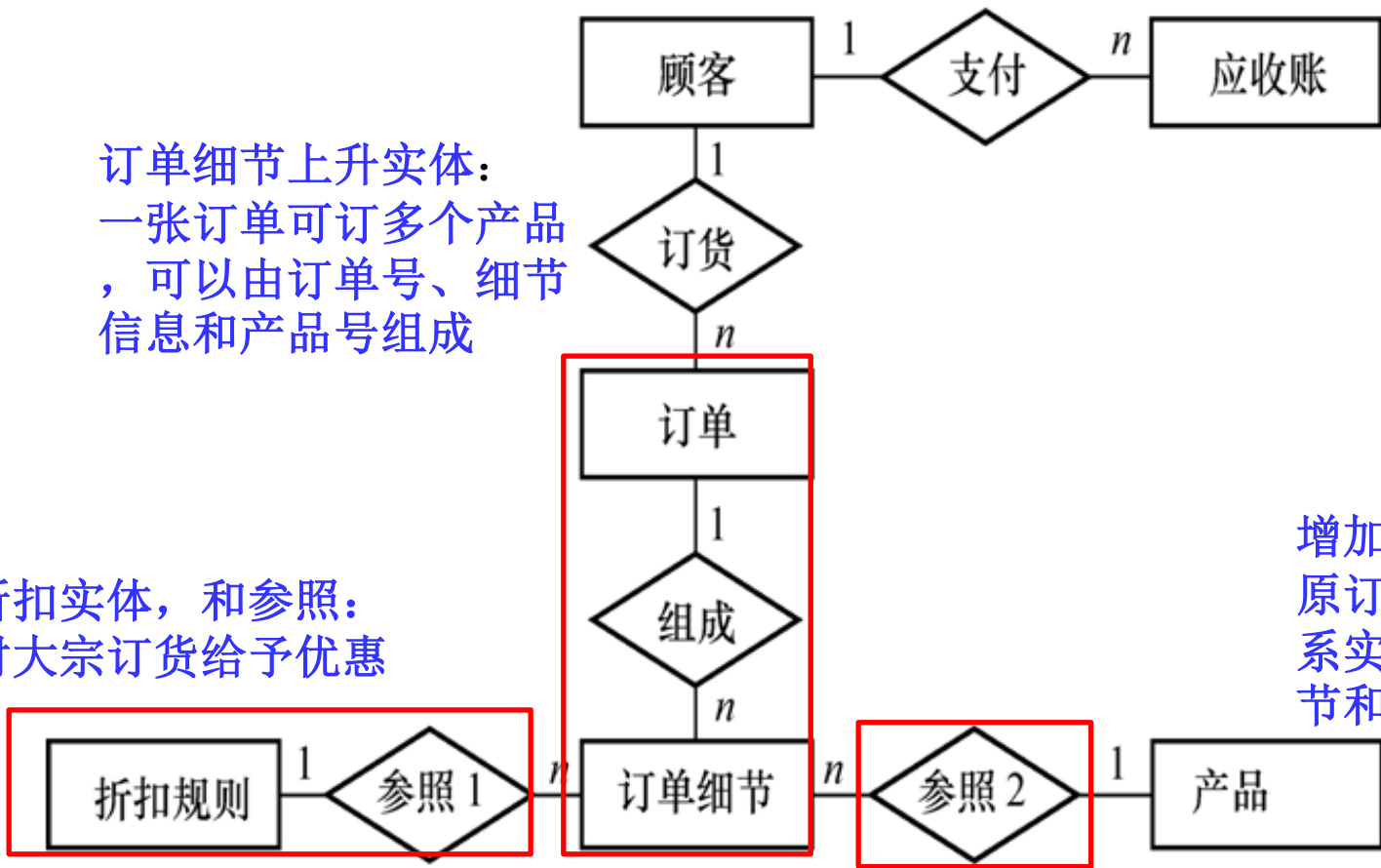


图7.23 销售管理子系统的E-R图



# 概念结构设计（续）

134

- 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，贷款限额}
  - 产品：{产品号，产品名，单价，重量}
  - 折扣规则：{产品号，订货量，折扣}

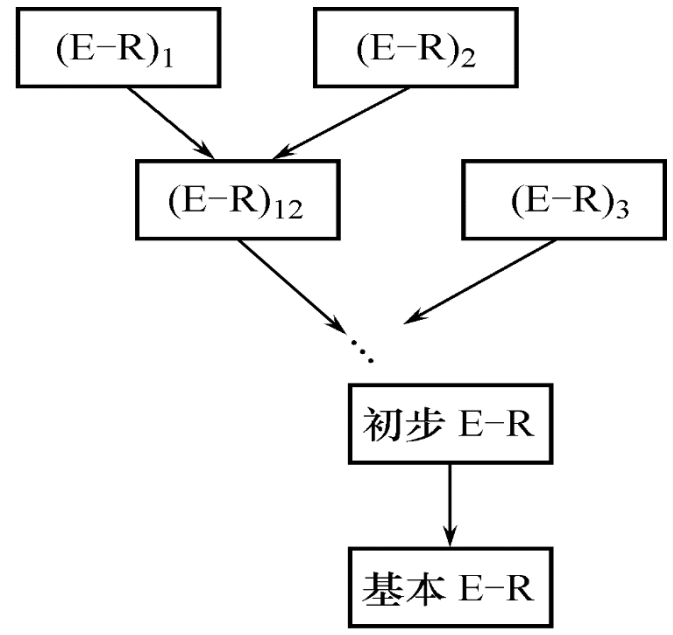
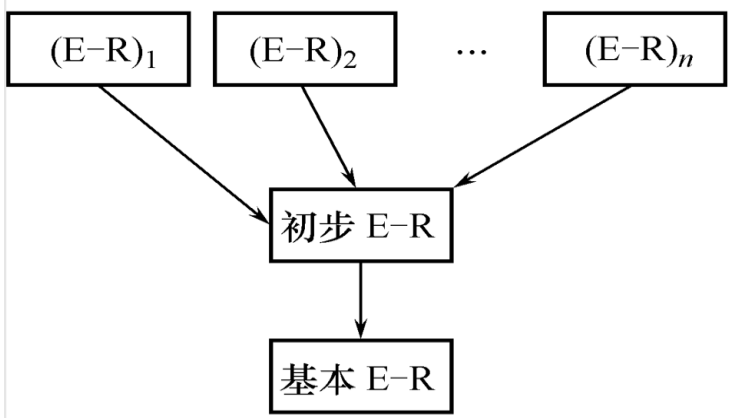
课后画实体-属性图？



# 概念结构设计（续）

## 2. E-R图的集成

- 一次集成：多个分E-R图一次集成（局部视图简单时）
- 逐步集成：用累加的方式一次集成两个分E-R图





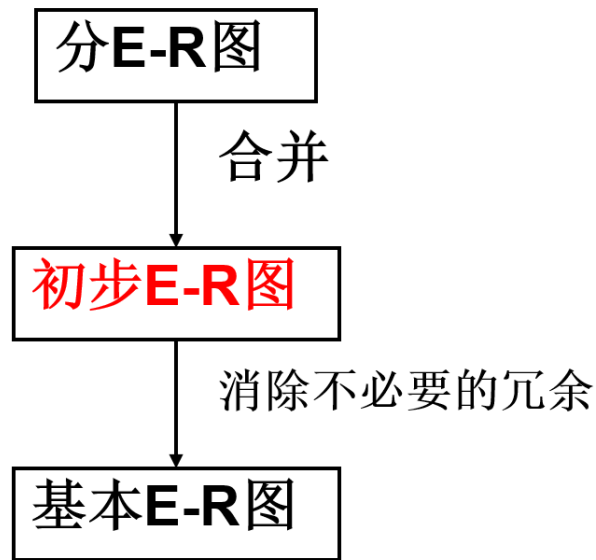
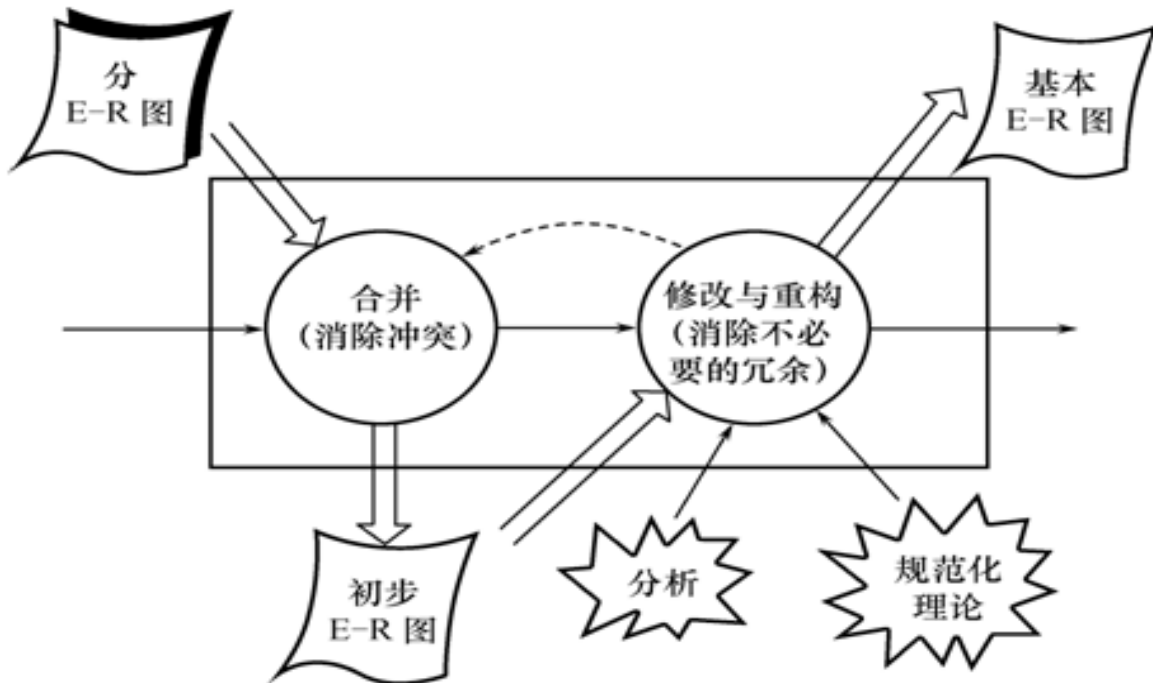


# 概念结构设计（续）

## 2. E-R图的集成

□ E-R图的集成一般需要分两步

- (1) 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
- (2) 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）

137

## 2. E-R图的集成

### □（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。
- 合并E-R图：消除E-R图中的不一致，全系统可理解
- 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）

138

## 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
  - 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）

139

## ② 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
  - 例如，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



# 概念结构设计（续）

140

## ③结构冲突

- 1. 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。（变换遵循7.3.5的两个准则）
- 2. 同一实体在不同子系统的E-R图中所包含的属性个数和属性排列次序不完全相同
  - 常见：原因是不同的应用关心实体的不同侧面
  - **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）

141

## ③结构冲突（续）

- 3. 实体间的联系在不同的E-R图中为不同的类型
  - 例如，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：**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”

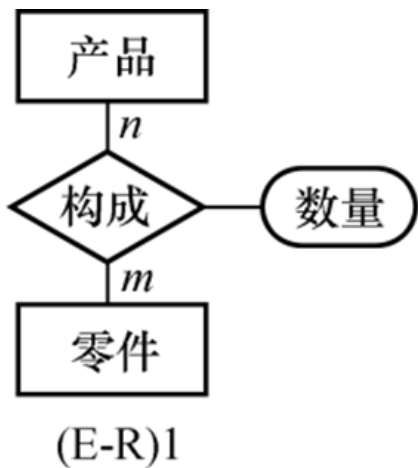


图7.25 (b) 中产品、零件与供应商三者之间还存在多对多的联系“供应”

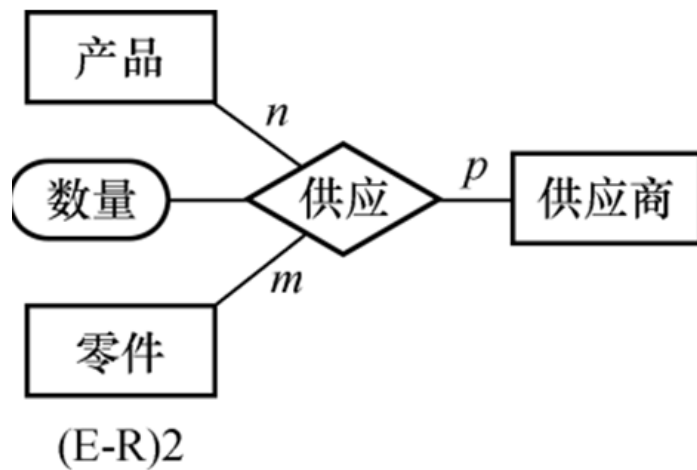
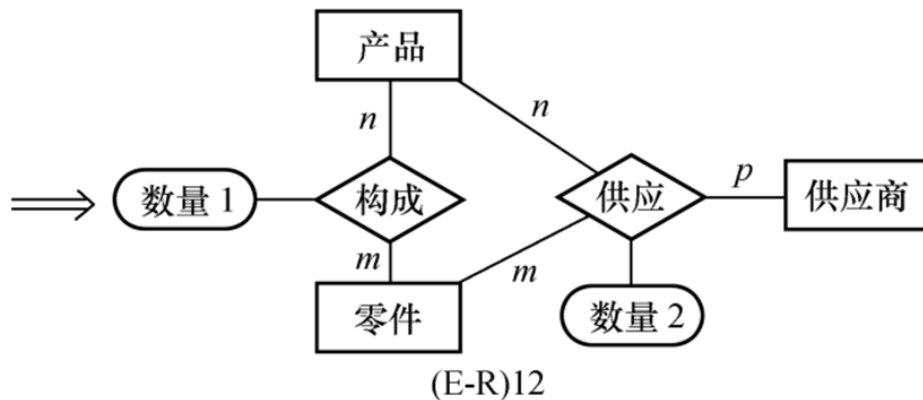


图7.25 (c), 合并两个E-R图





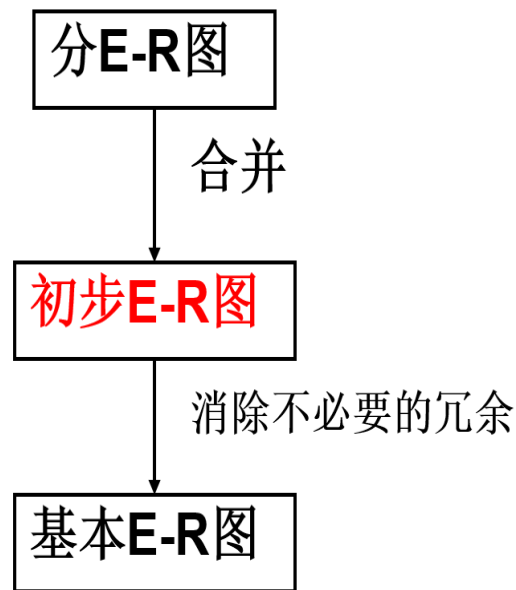
# 概念结构设计（续）

## E-R图的集成

在(1)初步E-R图中，仍存在一些冗余的数据和实体间冗余的联系

### （2）消除不必要的冗余，设计基本E-R图

- 破坏完整性，给数据库维护增加困难
  - 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
- 消除冗余方法
  - 主要采用分析方法
  - 规范化理论



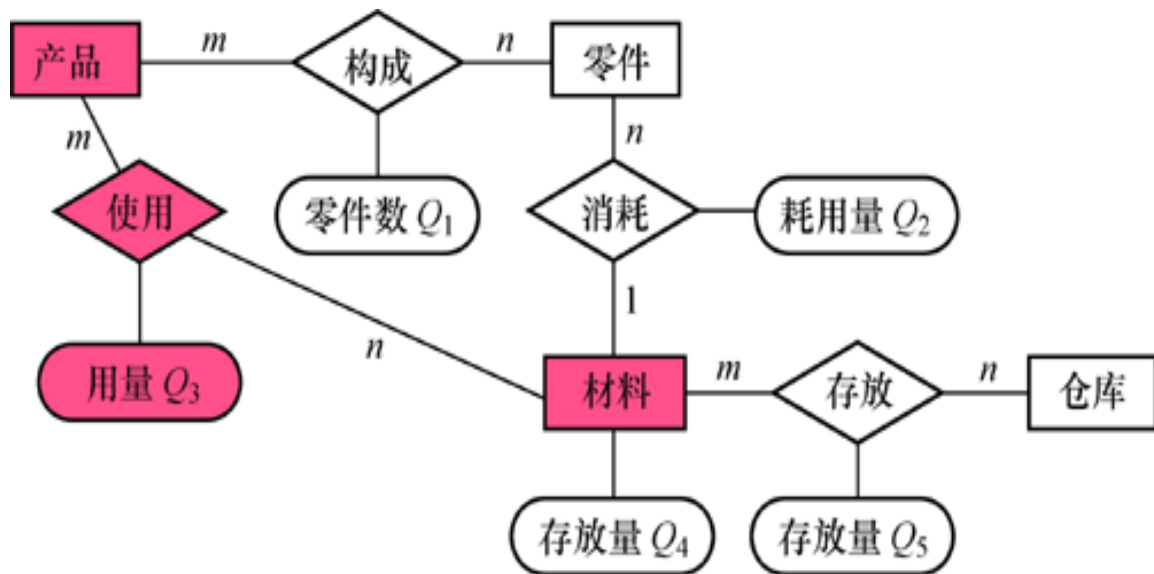




# 概念结构设计 (续)

## 消除冗余方法：采用分析方法

- 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逻辑关系的说明来消除冗余



$Q_3=Q_1 \times Q_2$ ,  $Q_4=\sum Q_5$ 。  
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 ( $Q_3=Q_1*Q_2$ 是冗余,  $Q_4$ 是 $Q_5$ 的合也冗余)

注意：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。



# 消除冗余的方法（续）

145

- 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

此处如何实现？



# 概念结构设计（续）

## 消除冗余的方法：

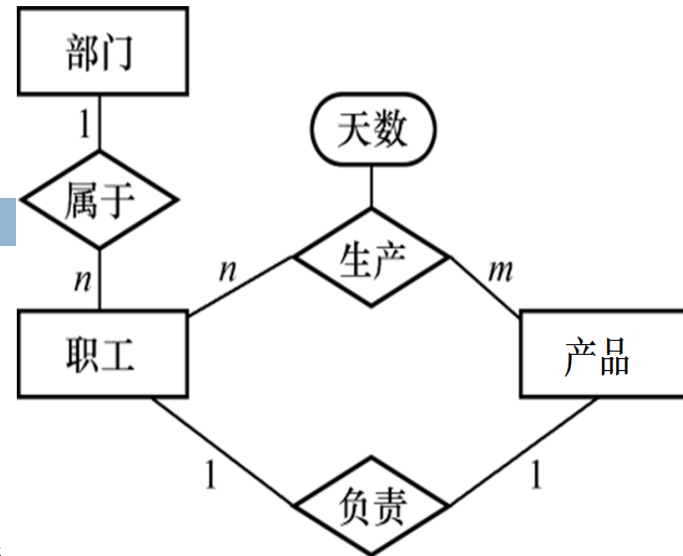
用规范化理论来消除冗余

### 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：有函数依赖集 $F_L$

- 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。

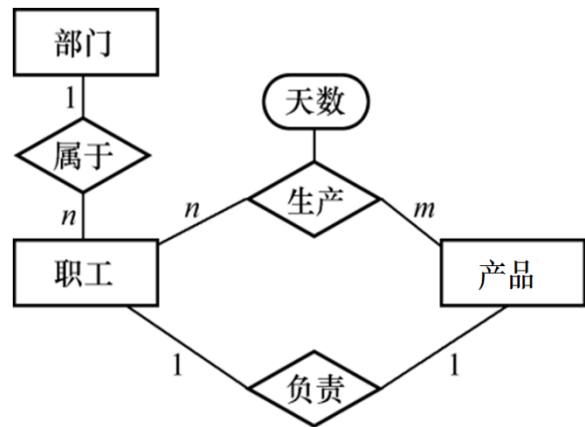




# 概念结构设计（续）

- 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
- 规范化理论受到泛关系假设的限制，应注意两个问题
  - 冗余的联系一定在D中，而D中的联系不一定是冗余
  - 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
    - 图7.27中，**职工和产品之间**存在另一个**1对1**联系表示为

- 负责人. 职工号 → 产品号
- 产品号 → 负责人. 职工号





# 概念结构设计（续）

□ [例7.2] 某工厂管理信息系统的视图集成。

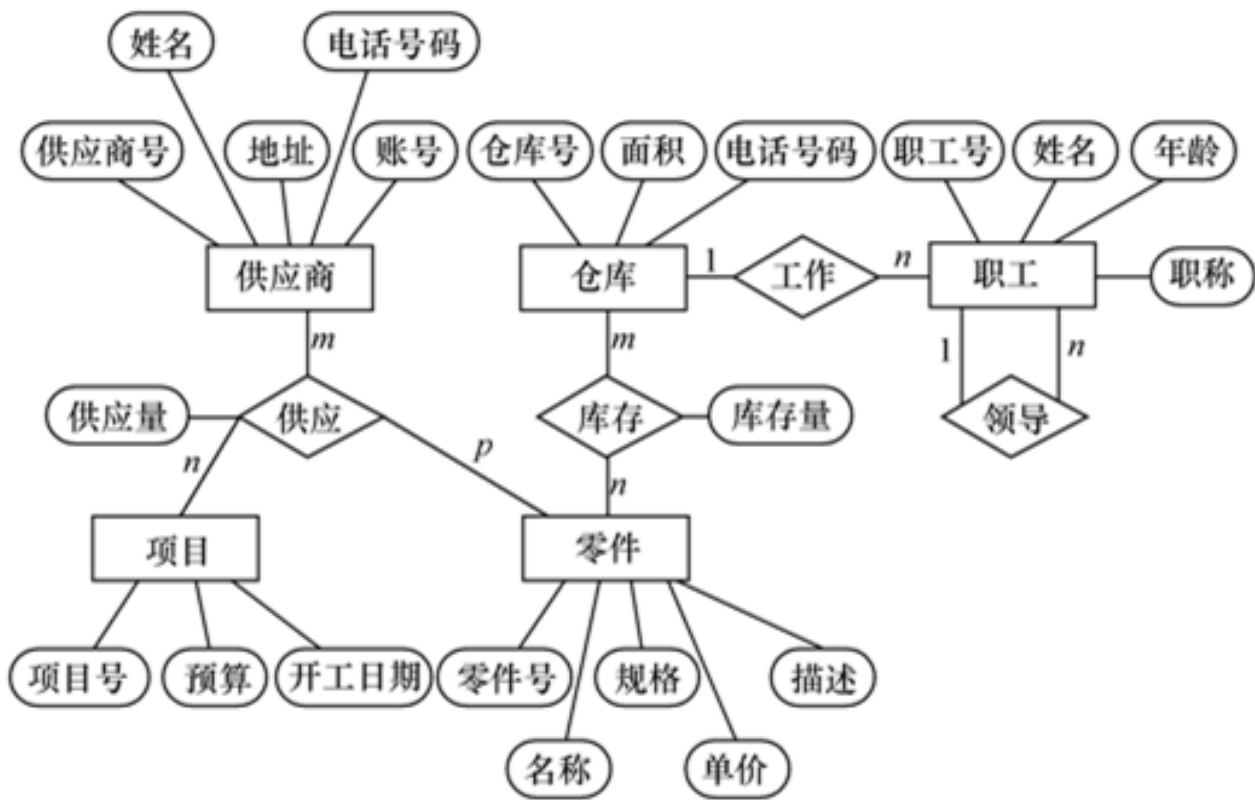


图7.11 子系统：工厂物资管理E-R图



# 概念结构设计（续）

□ [例7.2] 某工厂管理信息系统的视图集成

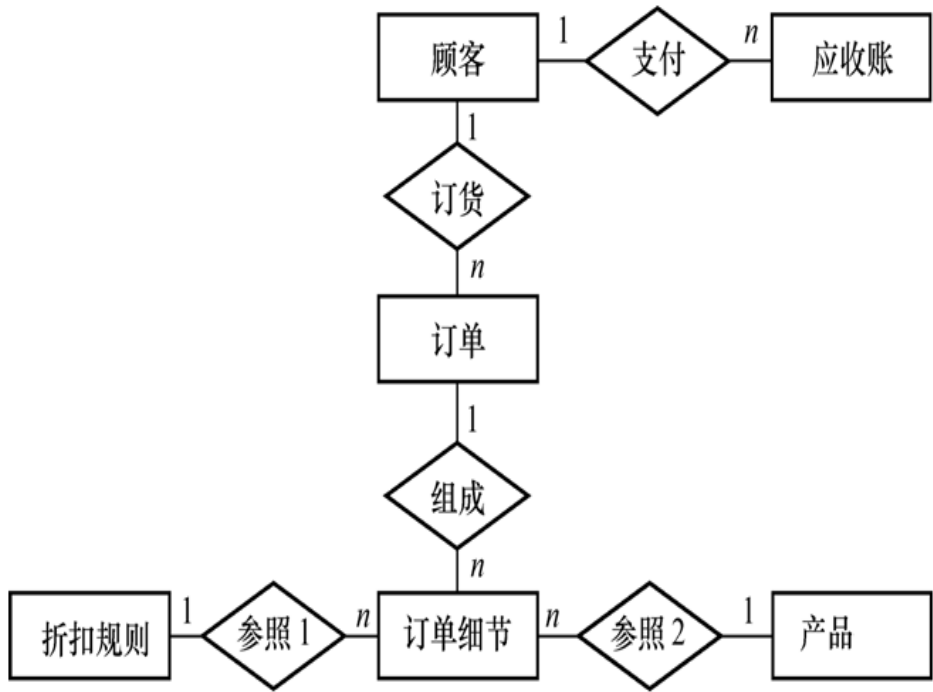


图7.23 销售管理子系统的E-R图

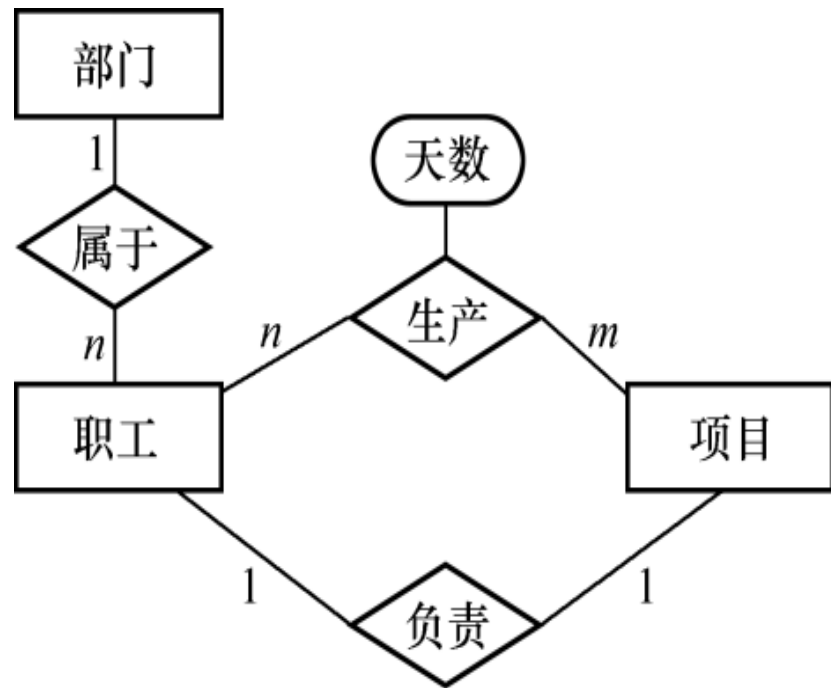


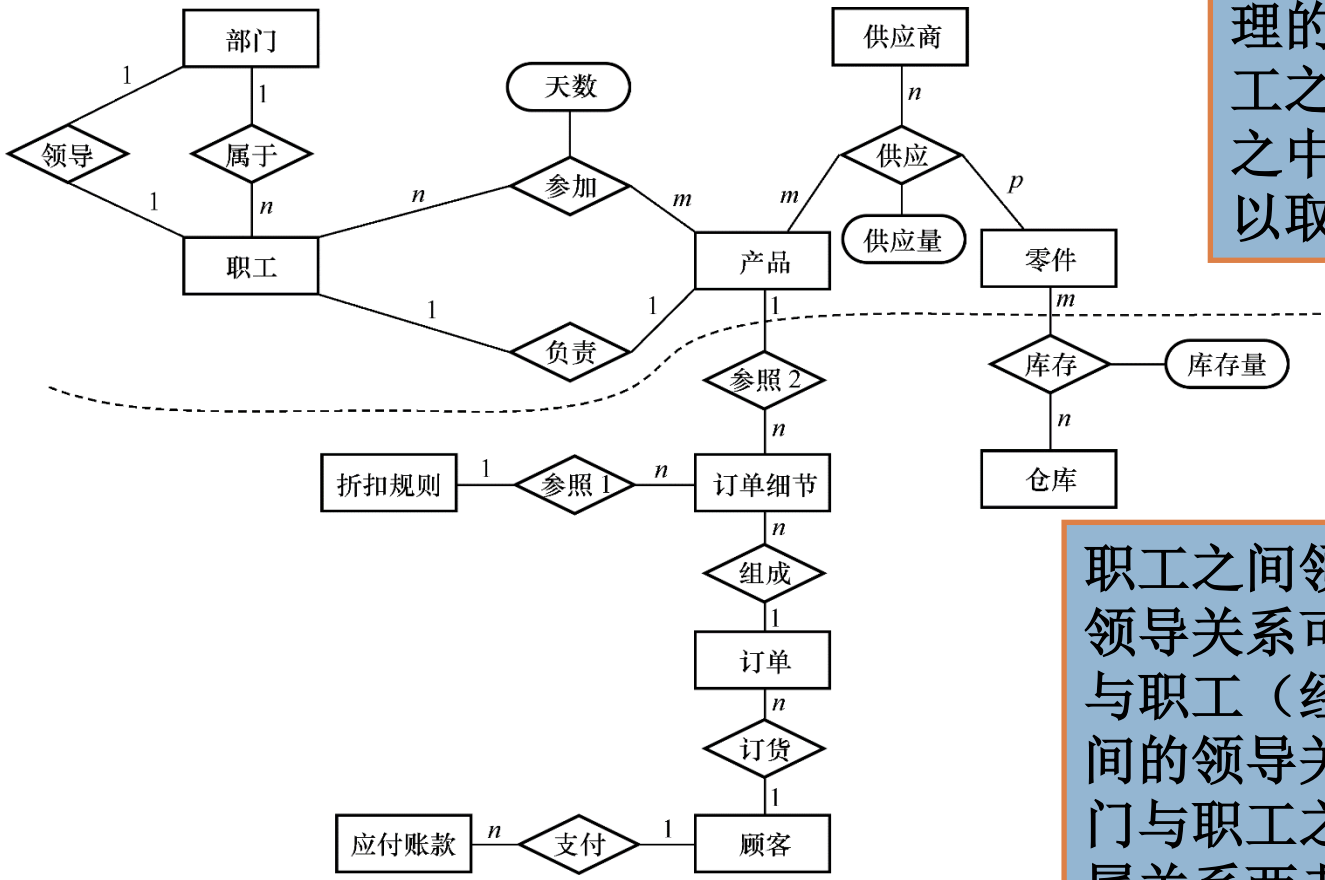
图7.27 劳动人事管理的分E-R图



# 概念结构设计 (续)

□ [例7.2] 某工厂管理信息系统的视图集成。

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...



## 消除冗余，设计生成基本E-R图实例（续）

151

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

152

- 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**
- 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。



# 第七章 数据库设计

155

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

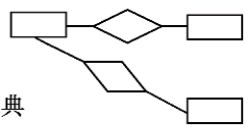
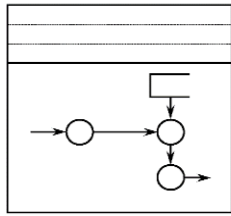
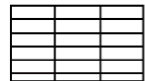
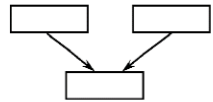
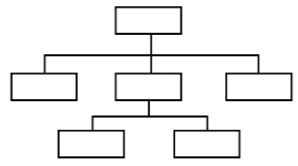
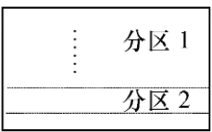
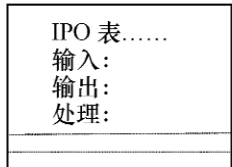
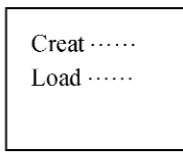
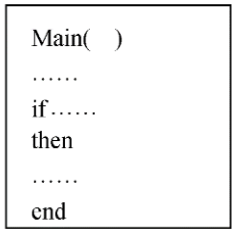
156

- 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
  - 关系模型：一组关系模式
- 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



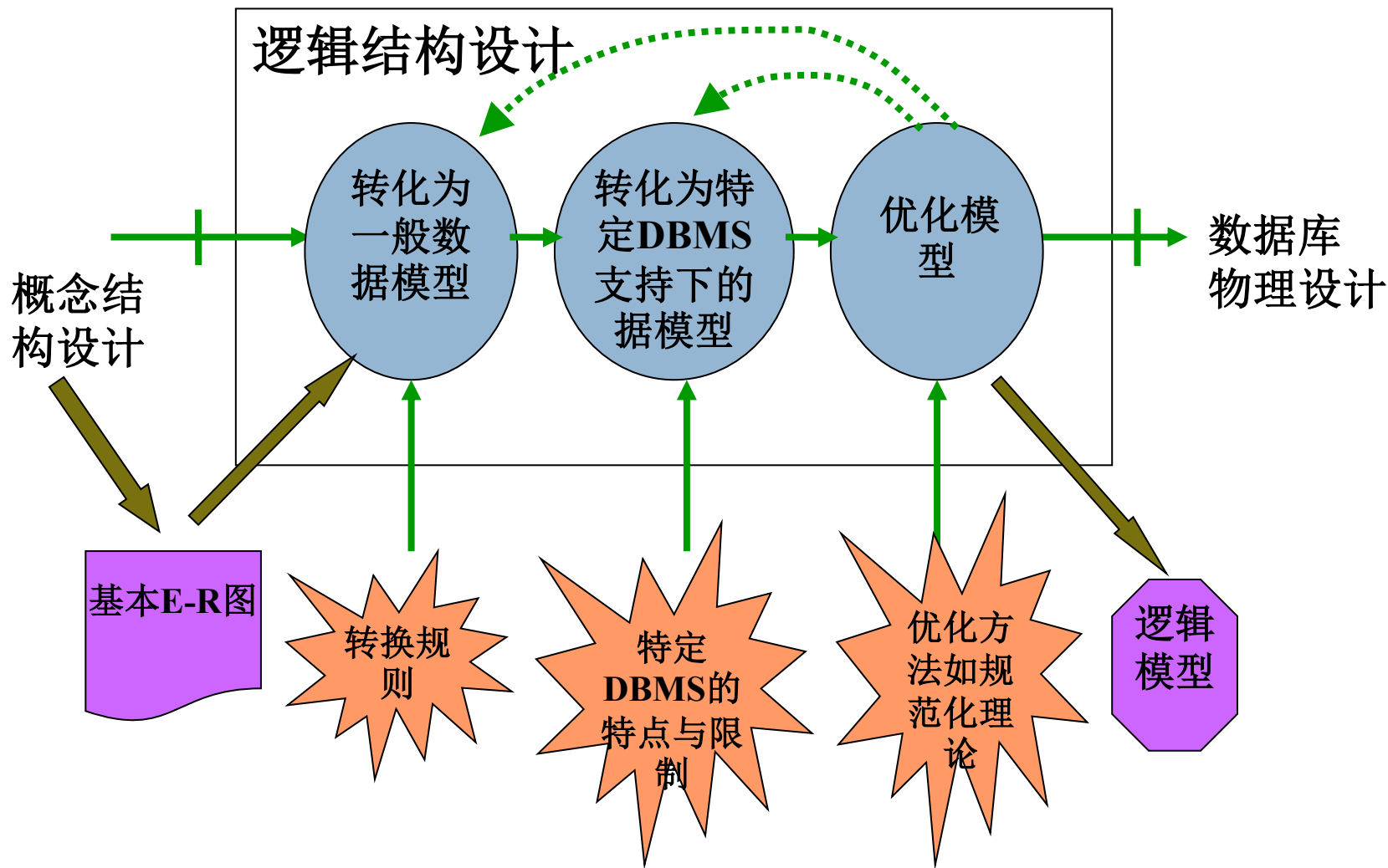
# 数据库

# 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)





## 7.4 逻辑结构设计

159

### 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

160

- 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



# E-R图向关系模型的转换（续）

161

- 转换内容
- 转换原则





# 转换内容

162

- 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、实体的属性和实体之间的联系转换为关系模式。



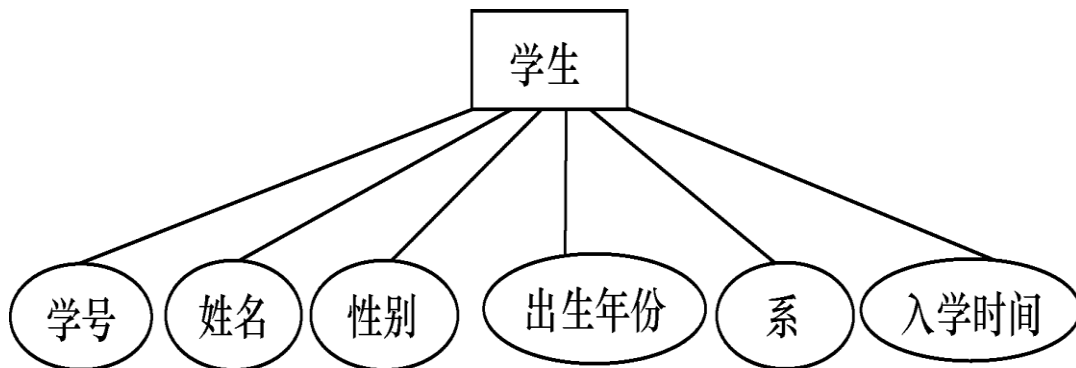
# 转换原则

163

## □ 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



学生（学号，姓名，性别，出生年份，系别，入学时间）



# 转换原则（续）

164

## 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

- 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

- 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



# 转换原则（续）

[例] “管理”联系为1: 1的联系

(1) 转换为一个独立的关系模式

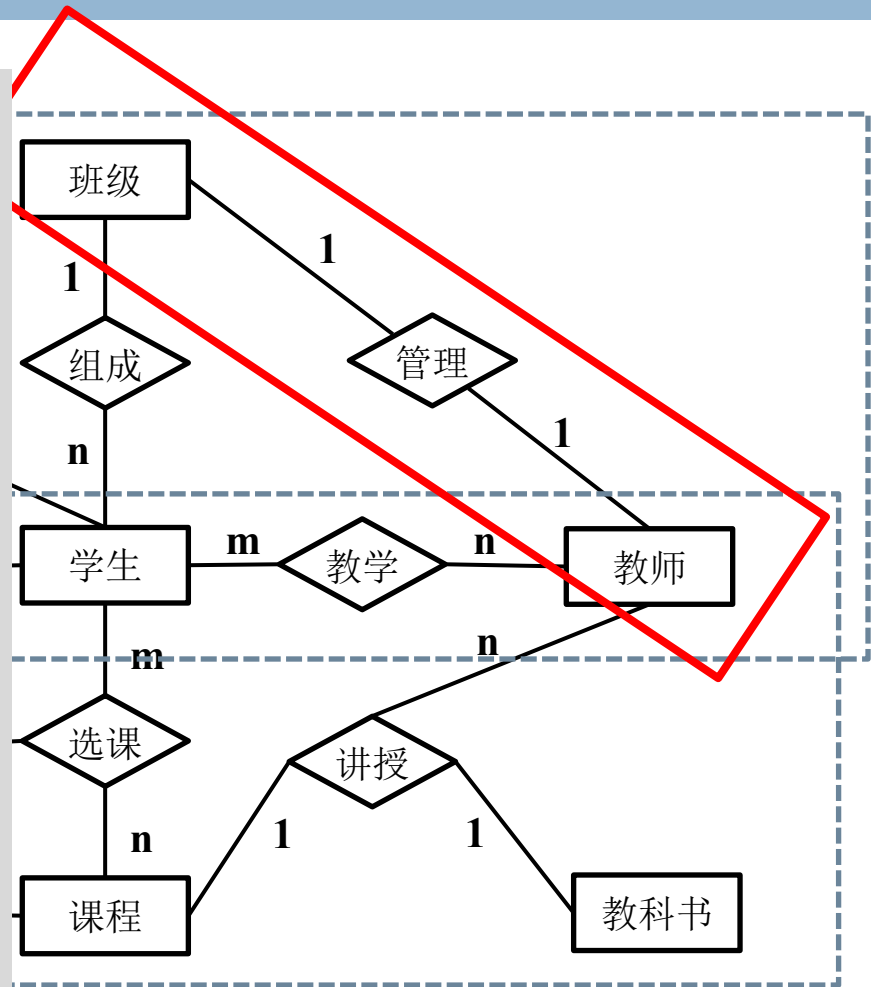
- 管理 (职工号, 班级号)
- 管理 (职工号, 班级号)

(2) “管理”与“班级”关系模式合并

- 班级 (班级号, 学生人数, **职工号**)
- 在“班级”中加入教师的码, 即职工号

(3) “管理”与“教师”关系模式合并

- 教师 (职工号, 姓名, 性别, 职务, **班级号**, 是否为优秀班主任)
- 在“教师”中加入“班级”的码, 即班级号



学生管理子系统E-R图



# 转换原则（续）

166

2. 实体型间的联系有以下不同情况：

(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

- 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

□ 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



# 转换原则 (续)

[例] “组成”联系为1: n的联系

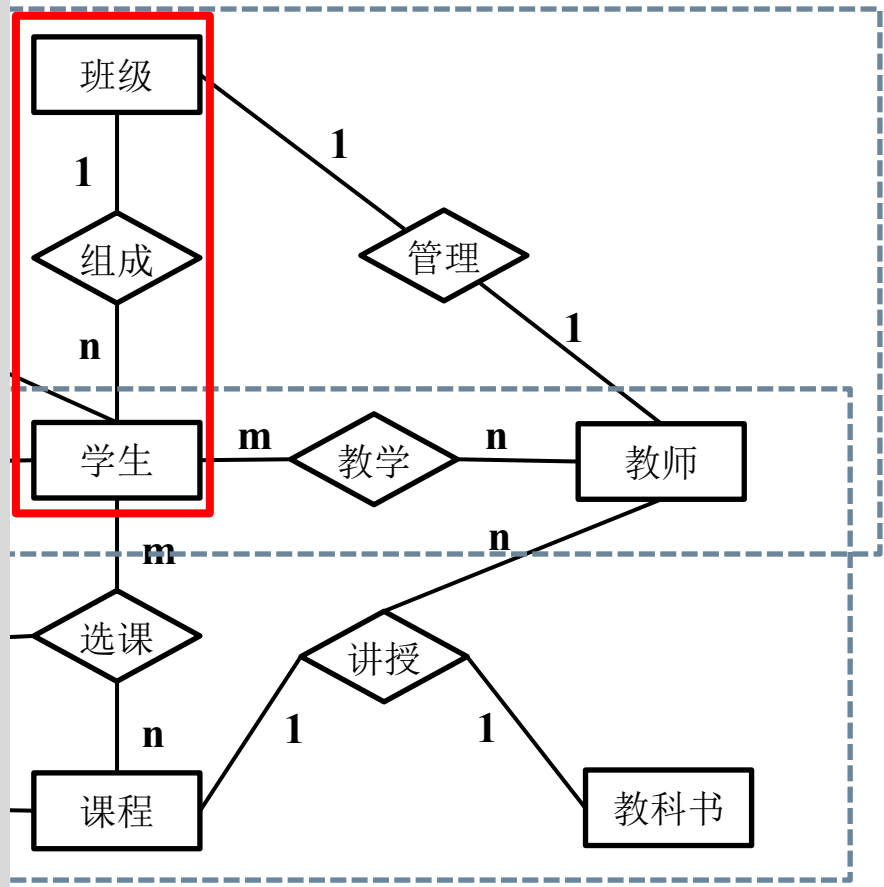
将其转换为关系模式的两种方法

(1) 使其成为一个独立的关系模式:

- 组成 (学号, 班级号)

(2) 将其与“学生”合并

- 学生 (学号, 姓名, 性别, 出生日期, 所在系, 年级, **班级号**, 平均成绩)



学生管理子系统E-R图



# 转换原则（续）

168

## 2. 实体型间的联系有以下不同情况：

### (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
- 不能合并：部分函数依赖

[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）



# 转换原则（续）

169

2. 实体型间的联系有以下不同情况：

(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

[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：

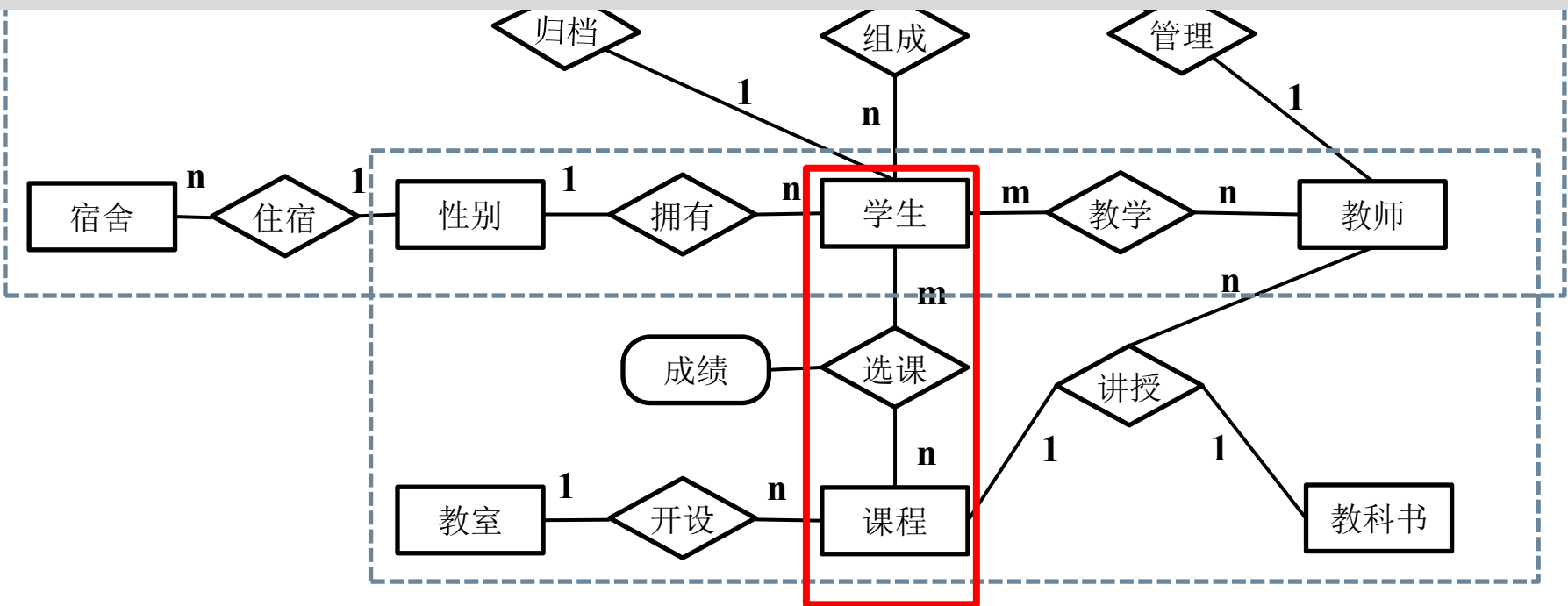
选修（学号，课程号，成绩）





# 转换原则 (续)

[例] “选修”联系为m: n的联系, 可以将它转换为选修 (学号, 课程号, 成绩), 其中, 学号和课程号为关系的组合码



学生管理子系统E-R图



# 转换原则（续）

171

2. 实体型间的联系有以下不同情况：

(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

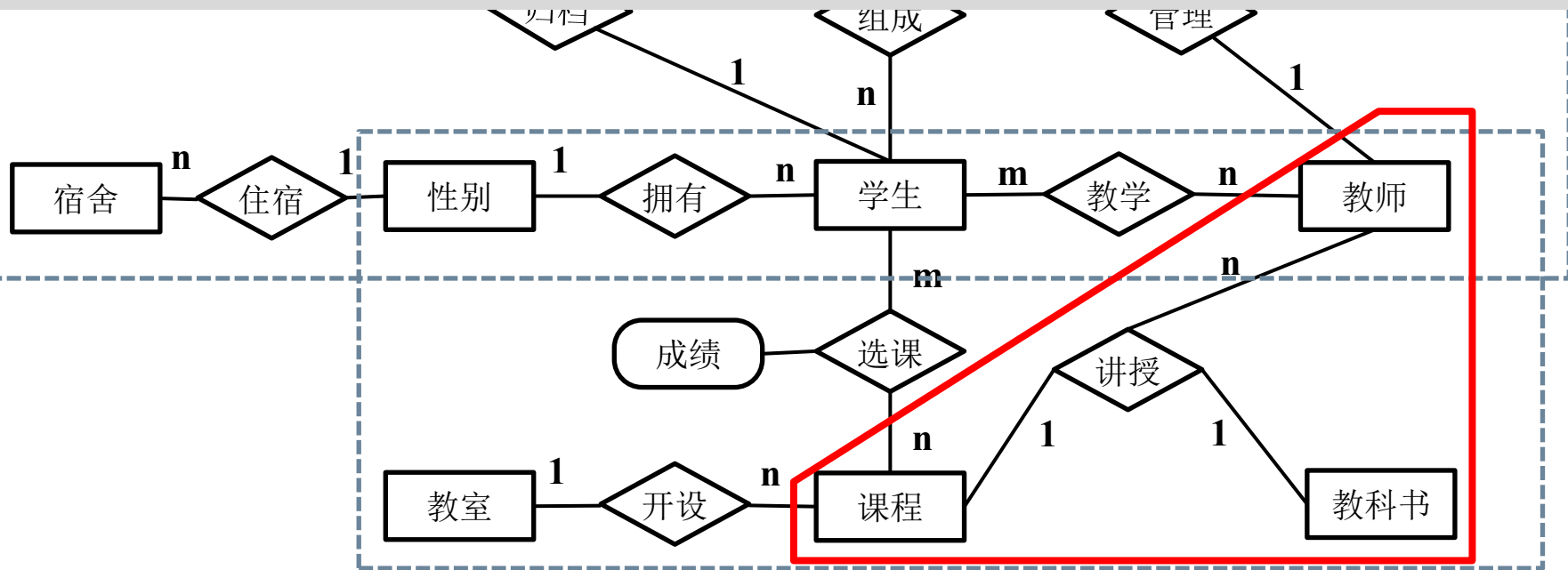
[例] “讲授”联系是一个三元联系，可以将它转换为如下关系模式，其中课程号、职工号和书号为关系的组合码：

讲授（课程号，职工号，书号）



# 转换原则 (续)

[例] “讲授”联系是一个三元联系，可以将它转换为  
讲授 (课程号, 职工号, 书号)  
其中，课程号、职工号和书号为关系的组合码



学生管理子系统E-R图



# 转换原则（续）

173

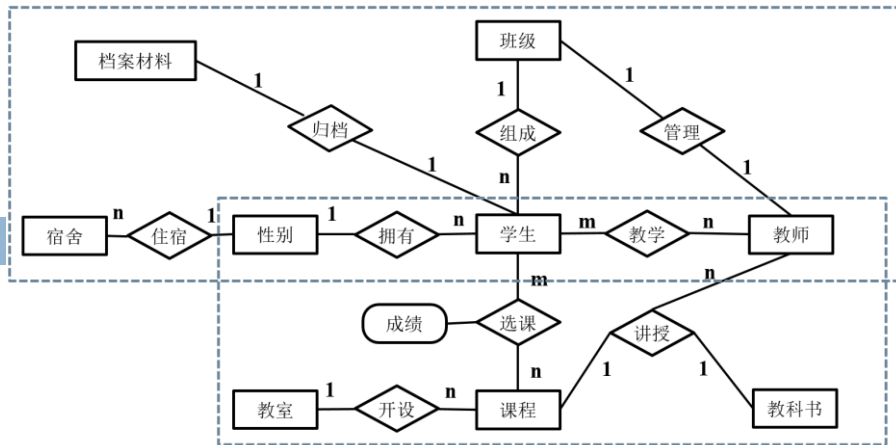
## 2. 实体型间的联系有以下不同情况：

### (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



# 转换原则 (续)



按照上述转换原则，学生管理子系统中的实体和联系转换为下列关系模型

## 实体与联系

- **学生** (学号, 姓名, 性别, 出生日期, 所在系, 年级, 班级号, 平均成绩, 档案号)
- 性别 (性别, 宿舍楼)
- 宿舍 (宿舍编号, 地址, 性别, 人数)
- 班级 (班级号, 学生人数)
- **教师** (职工号, 姓名, 性别, 职称, 班级号, 是否优秀班主任)

- 教学 (职工号, 学号)
- **课程** (课程号, 课程名, 学分, 教室号)
- 选修 (学号, 课程号, 成绩)
- 教科书 (书号, 书名, 价格)
- 教室 (教室编号, 地址, 容量)
- 讲授 (课程号, 教师号, 书号)
- 档案材料 (档案号, ...)

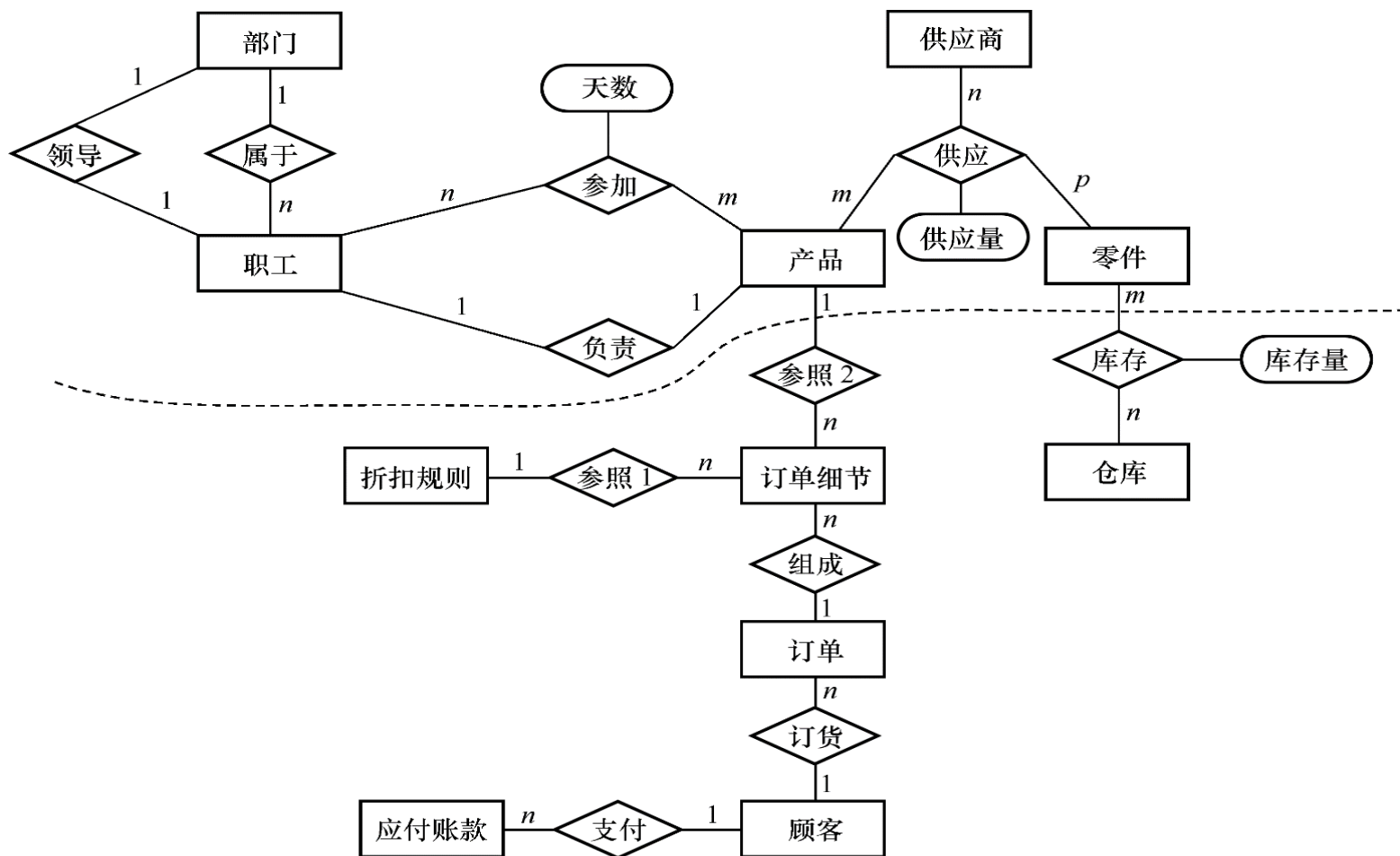
该关系模型由12个关系模式组成，其中：

- 学生关系模式包含了“拥有”、“组成”和“归档”联系所对应的关系模式
- 教师关系模式包含了“管理”联系所对应的关系模式
- 宿舍关系模式包含了“住宿”联系所对应的关系模式
- 课程关系模式包含了“开设”联系所对应的关系模式



# E-R图向关系模型的转换（续）

[例] 把图7.28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）

[例] 把图7.28中虚线上部的E-R图转换为关系模型

## □ 部门实体的关系模式

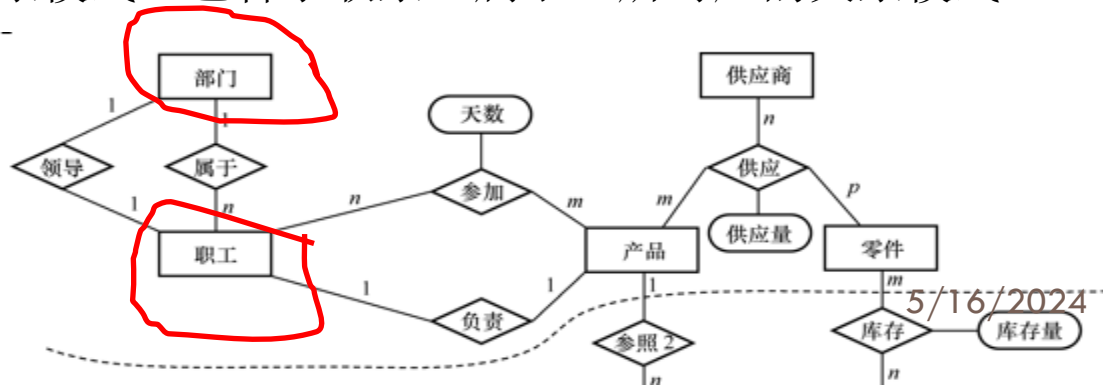
部门（部门号，部门名，经理的职工号，...）

- 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）

[例] 把图7.28中虚线上部的E-R图转换为关系模型

- 产品实体的关系模式

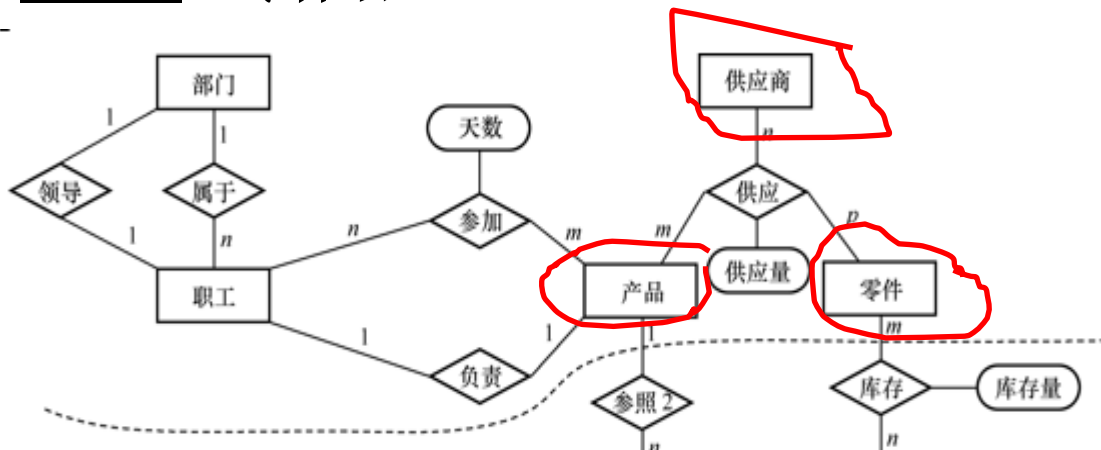
产品（产品号，产品名，产品组长的职工号，...）

- 供应商实体的关系模式

供应商（供应商号，姓名，...）

- 零件实体的关系模式

零件（零件号，零件名，...）







# E-R图向关系模型的转换（续）

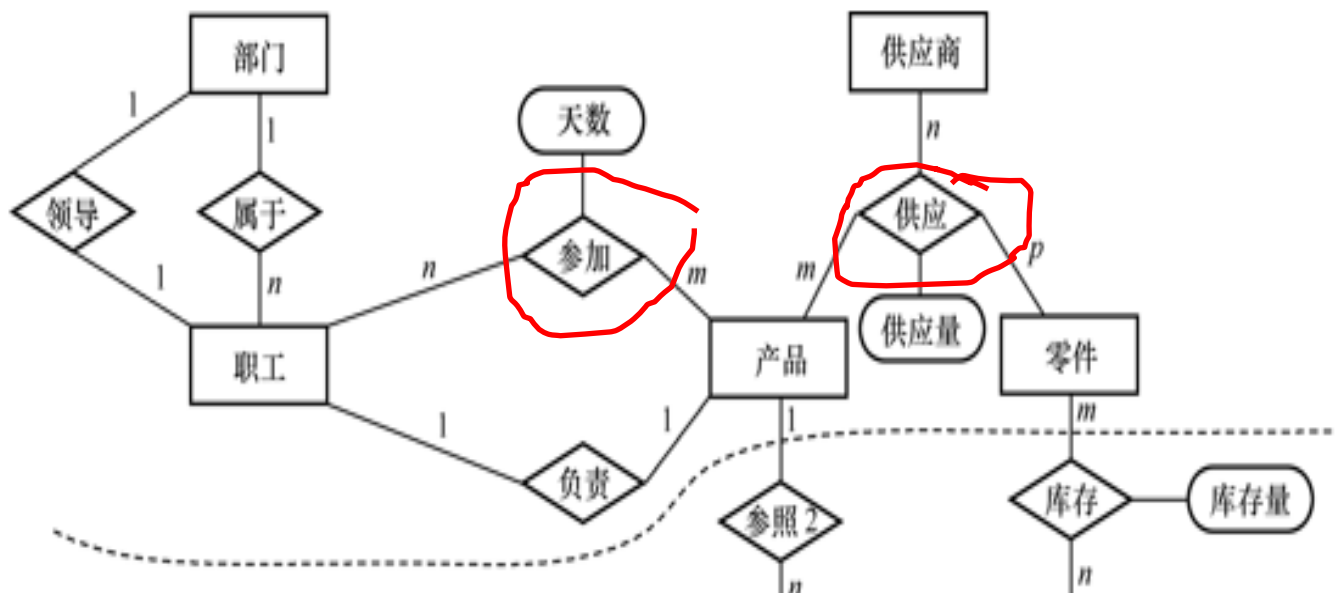
[例] 把图7.28中虚线上部的E-R图转换为关系模型

- “参加”联系的关系模式

职工工作（职工号，产品号，工作天数，...）

- “供应”联系的关系模式

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）

179

[例] **总结** 把图7.28中E-R图转换为关系模型

- 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）

180

## 一些TIP:

- 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



## 7.4 逻辑结构设计

181

### 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

182

- 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**
- 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。
- 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）

183

## □ 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）

## 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

	<b>1NF</b>
	↓ 消除非主属性对码的部分函数依赖
消除决定属性	<b>2NF</b>
集非码的非平	↓ 消除非主属性对码的传递函数依赖
凡函数依赖	<b>3NF</b>
	↓ 消除主属性对码的部分和传递函数依赖
	<b>BCNF</b>
	↓ 消除非平凡且非函数依赖的多值依赖
	<b>4NF</b>

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）

185

**注意：**并不是规范化程度越高的关系就越优

- 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
- 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。
- 非BCNF的关系模式虽然会存在不同程度的更新异常。若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。若有更新操作，需要考虑完整性约束（如触发器）
- 一般说来，第三范式就足够了





# 数据模型的优化（续）

186

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）

187

观察“平均成绩”：

学号 $\rightarrow$ (英语,数学,语文) $\rightarrow$ 平均成绩

因此该关系模式中存在传递函数依赖，是2NF关系

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）

188

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）

189

## □ 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
- 例如，物流系统：跨域快递，同城快递

### ➤ 水平分解的适用范围

- 满足“80/20原则”的应用。把经常被使用的数据（约20%）水平分解出来，形成一个子关系
- 并发事务经常存取不相交的数据
- 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）

190

## □ 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
- 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
  - 优点：可以提高某些事务的效率
  - 缺点：可能使另一些事务不得不执行连接操作，降低了效率



## 7.4 逻辑结构设计

191

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

192

- 定义数据库模式—全局模式
  - 考虑系统应用的**时间效率**、**空间效率**、**易维护**等
- 定义用户子模式（外模式）—视图机制
  - 考虑局部应用的特殊需求和用户体验
    - (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视图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）

193

## (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视图，保证系统安全性

- 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）

- 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



# 设计用户子模式（续）

195

## （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。