

# 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 数据库的物理设计

## \*7.6 数据库实施和维护

## 7.7 小结

# 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。

# 数据库设计的基本步骤

## ❖ 数据库设计分6个阶段

- 需求分析
- 概念结构设计
- 逻辑结构设计
- 物理结构设计
- 数据库实施
- 数据库运行和维护

## ❖ 需求分析和概念设计独立于任何数据库管理系统

## ❖ 逻辑设计和物理设计与选用的DBMS密切相关

# 数据库设计的基本步骤（续）

## 二、数据库设计的过程(六个阶段)

### 1.需求分析阶段

- 准确了解与分析用户需求（包括数据与处理）
- 最困难、最耗费时间的一步

### 2.概念结构设计阶段

- 整个数据库设计的关键
- 通过对用户需求进行综合、归纳与抽象，形成一个独立于具体DBMS的概念模型

# 数据库设计的基本步骤（续）

## 3.逻辑结构设计阶段

- 将概念结构转换为某个DBMS所支持的数据模型
- 对其进行优化

## 4.数据库物理设计阶段

- 为逻辑数据模型选取一个最适合应用环境的物理结构（包括存储结构和存取方法）

# 数据库设计的基本步骤（续）

## 5.数据库实施阶段

- 运用**DBMS**提供的数据库语言（如**SQL**）及宿主语言，根据逻辑设计和物理设计的结果
  - 建立数据库
  - 编制与调试应用程序
  - 组织数据入库
  - 进行试运行

## 6.数据库运行和维护阶段

- 数据库应用系统经过试运行后即可投入正式运行
- 在数据库系统运行过程中必须不断地对其进行评价、调整与修改

# 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 数据库的物理设计**

**7.6 数据库实施和维护**

**7.7 小结**

# 需求分析的任务和重点

- ❖ 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）
  - 明确用户的各种需求
  - 确定新系统的功能
  - 充分考虑今后可能的扩充和改变
  
- ❖ 调查的重点是“数据”和“处理”，获得用户对数据库要求
  - 信息要求
  - 处理要求
  - 安全性与完整性要求

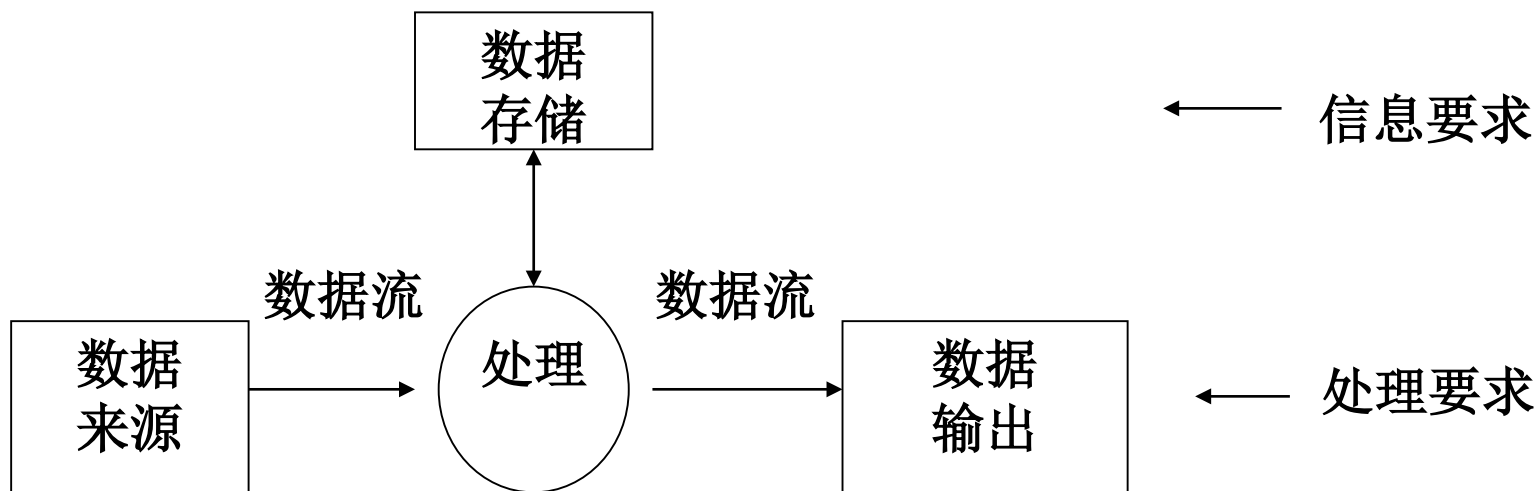


# 需求分析的方法

## ❖ 结构化分析方法（Structured Analysis，简称SA方法）

- 从最上层的系统组织机构入手
- 自顶向下、逐层分解分析系统

1. 首先把任何一个系统都抽象为：



# 进一步分析和表达用户需求（续）

## 2. 分解处理功能和数据

### (1) 分解处理功能

- 将处理功能的具体内容分解为若干子功能

### (2) 分解数据

- 处理功能逐步分解同时，逐级分解所用数据，形成若干层次的数据流图

### (3) 表达方法

- 处理逻辑：用判定表或判定树来描述
- 数据：用数据字典来描述

## 3. 将分析结果再次提交给用户，征得用户的认可

# 数据字典

## ❖ 数据字典的用途

- 进行详细的数据收集和数据分析所获得的主要结果

## ❖ 数据字典的内容

- 数据项
- 数据结构
- 数据流
- 数据存储
- 处理过程

# 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 数据库的物理设计**

**7.6 数据库实施和维护**

**7.7 小结**

# 概念结构

## ❖ 什么是概念结构设计

- 将需求分析得到的用户需求抽象为信息结构即概念模型的过程就是概念结构设计
- 概念结构是各种数据模型的基础，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）

## ❖ 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

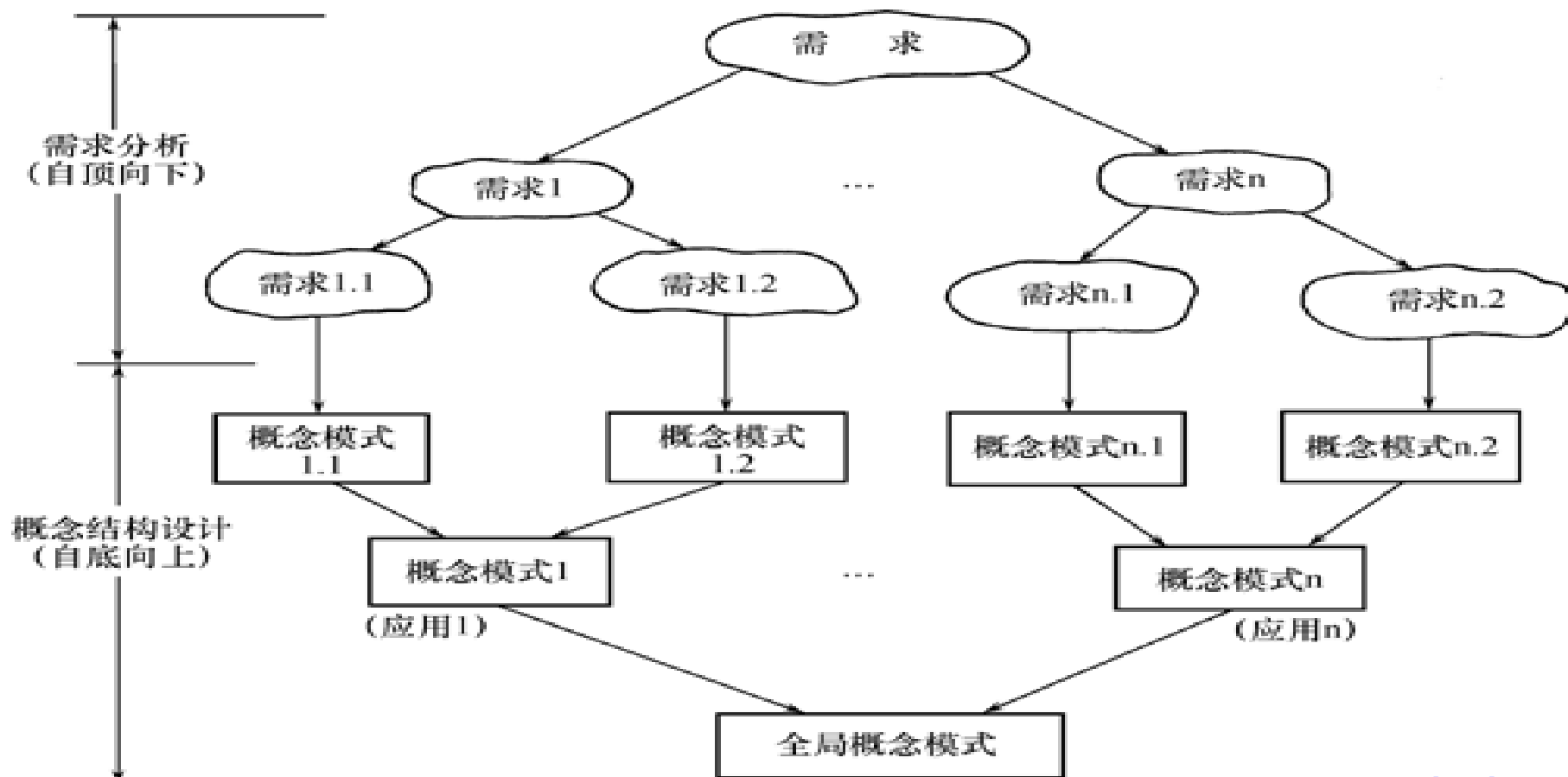
## ❖ 描述概念模型的工具

- E-R模型

# 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

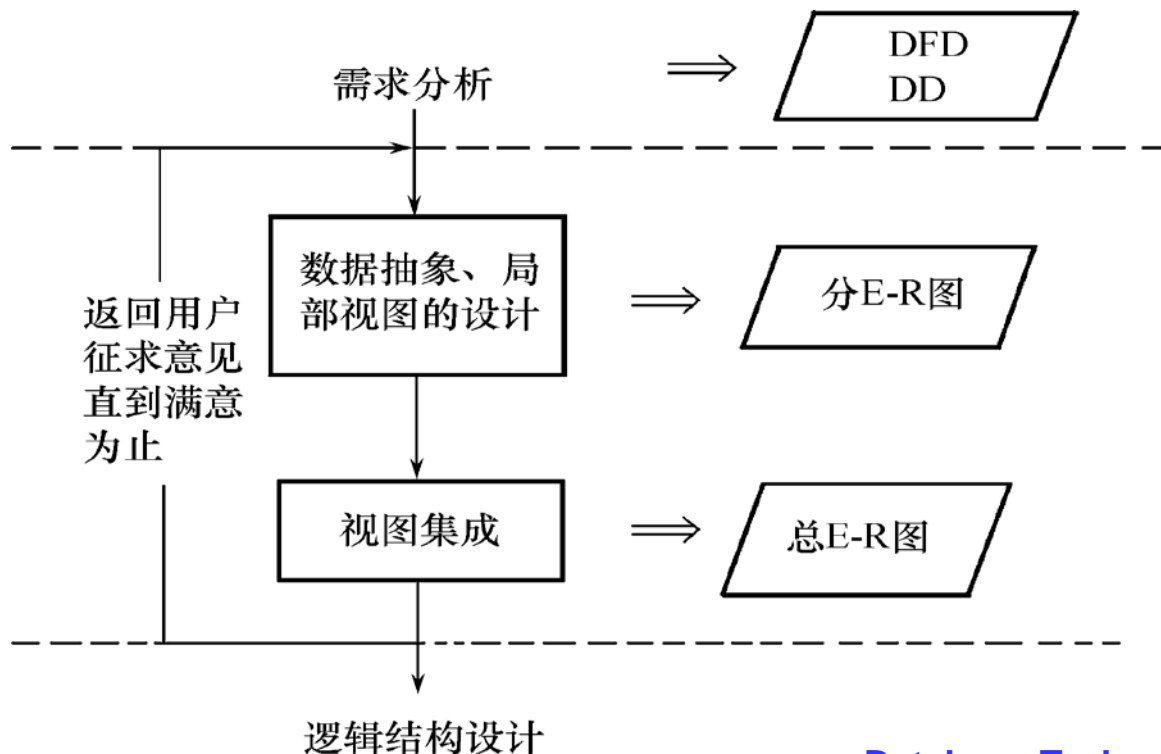


# 概念结构设计的方法与步骤（续）

## ❖ 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





# 数据抽象

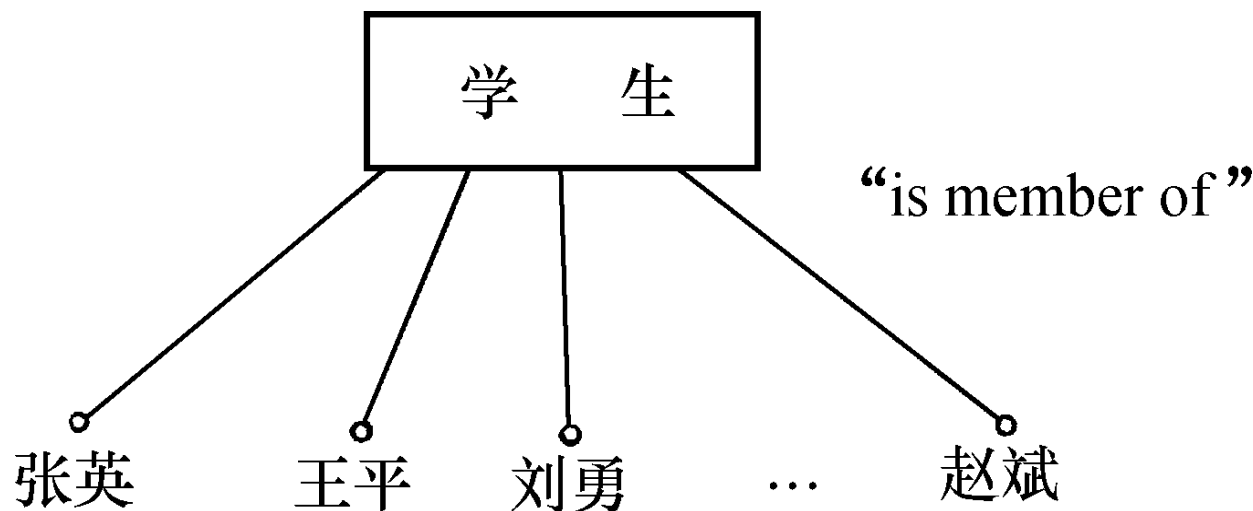
- ❖ 抽象是对实际的人、物、事和概念中抽取所关心的共同特性，忽略非本质的细节，并把这些特性用各种概念精确地加以描述。
  - 概念结构是对现实世界的一种抽象

# 数据抽象（续）

## ❖ 三种常用抽象

### 1. 分类（Classification）

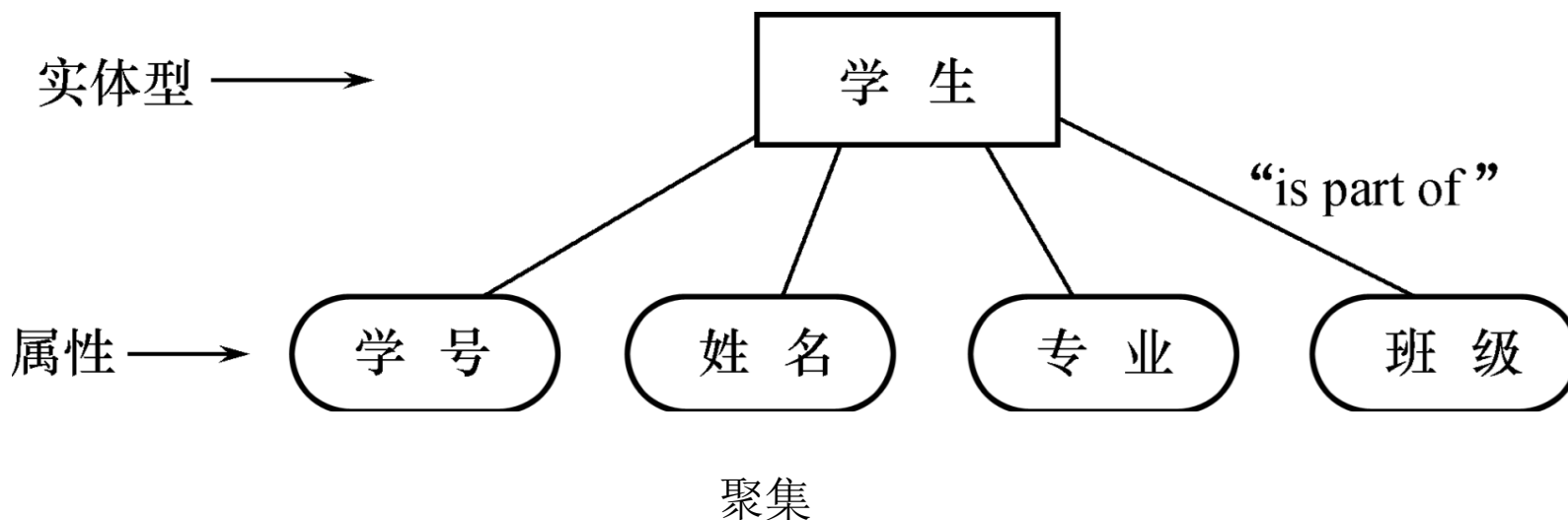
- 定义某一类概念作为现实世界中一组对象的类型
- 抽象了对象值和型之间的“is member of”的语义



# 数据抽象（续）

## 2. 聚集（Aggregation）

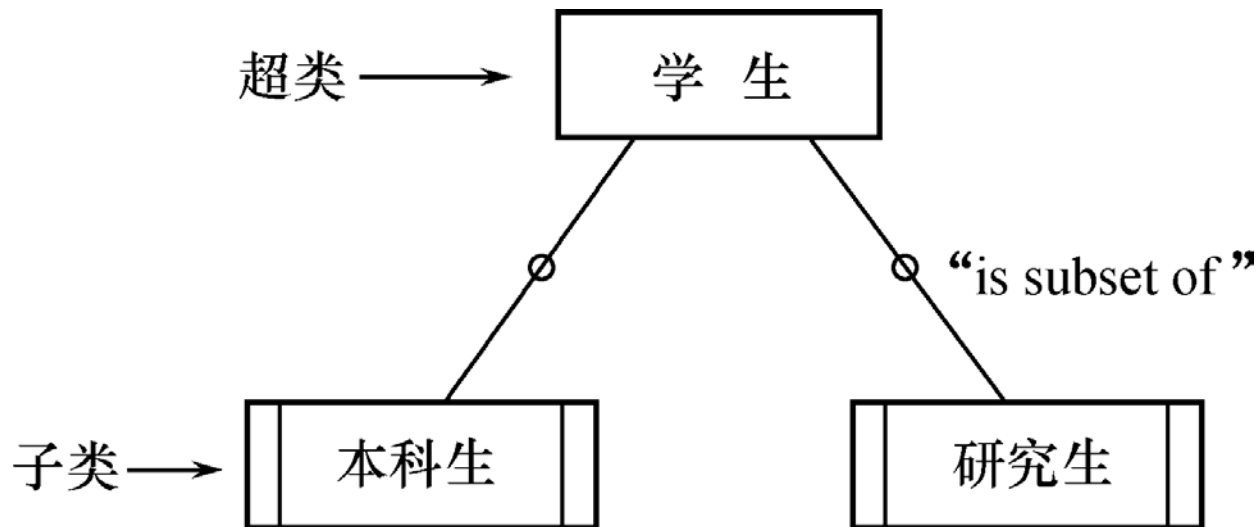
- 定义某一类型的组成成分
- 抽象了对象内部类型和成分之间 “is part of”的语义



# 数据抽象（续）

## 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



概括

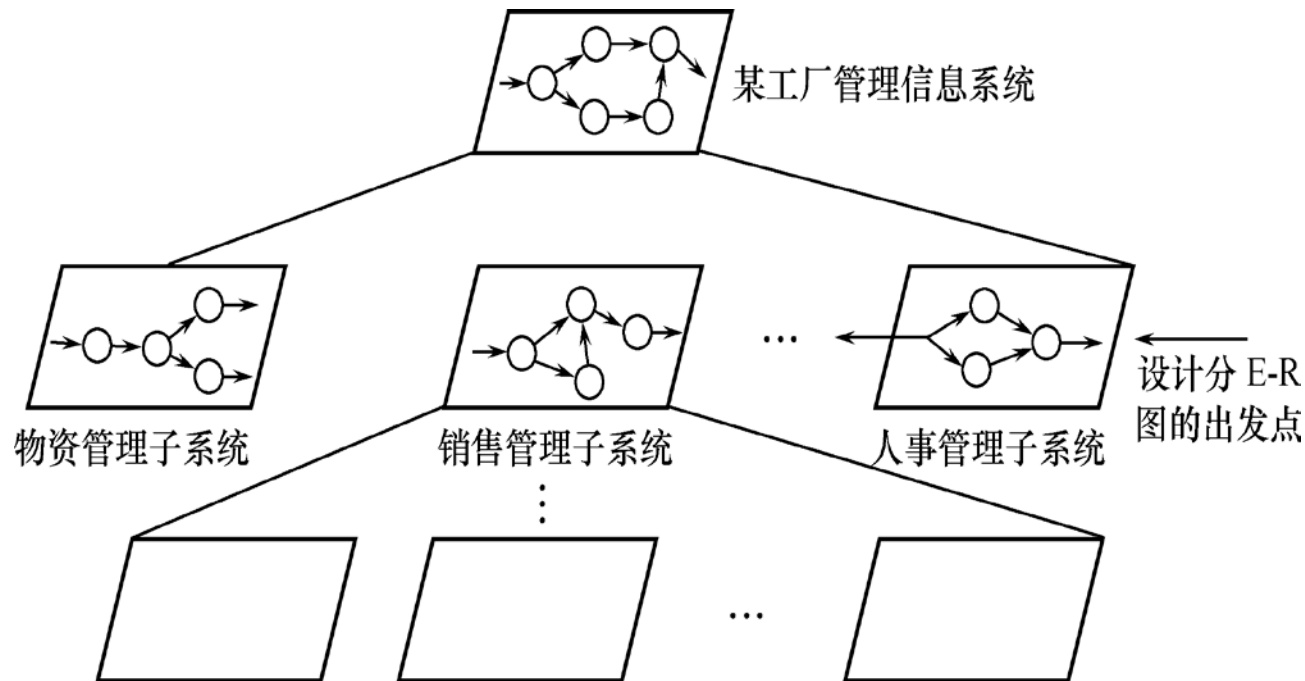
# 局部视图设计

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

- ❖ 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



## 2. 逐一设计分E-R图

### ❖ 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

## ❖ 如何抽象实体和属性

### ■ 实体

- ⑩ 现实世界中一组具有某些共同特性和行为的对象就可以抽象为一个实体。对象和实体之间是“**is member of**”的关系。
- ⑩ 例：在学校环境中，可把张三、李四等对象抽象为学生实体。

### ■ 属性

- ⑩ 对象类型的组成成分可以抽象为实体的属性。组成成分与对象类型之间是“**is part of**”的关系。
- ⑩ 例：学号、姓名、专业、年级等可以抽象为学生实体的属性。其中学号为标识学生实体的码。



# 逐一设计分E-R图

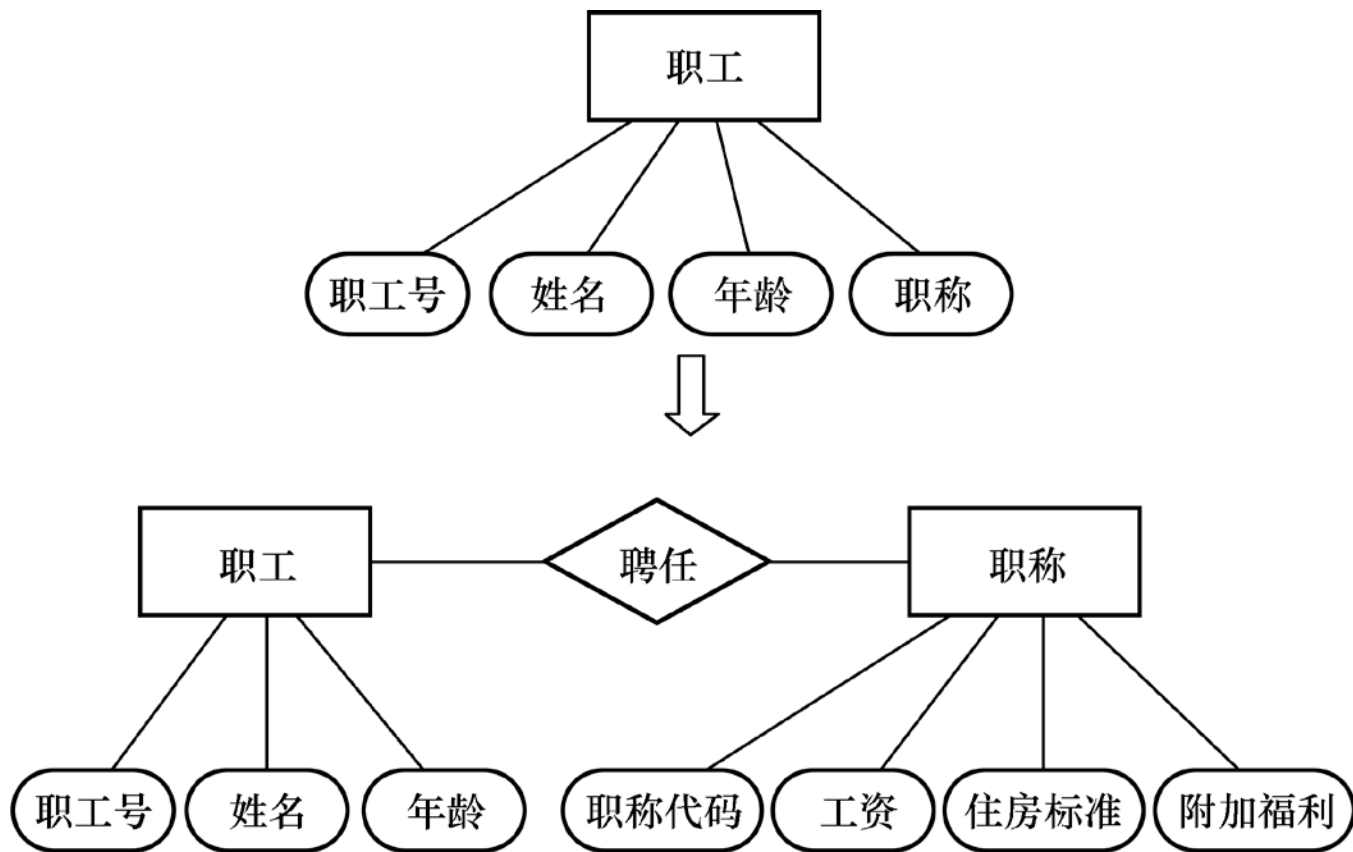
## ❖ 如何区分实体和属性

- 实体与属性是相对而言的。同一事物，在一种应用环境中作为“属性”，在另一种应用环境中就必须作为“实体”。

## ❖ 一般原则

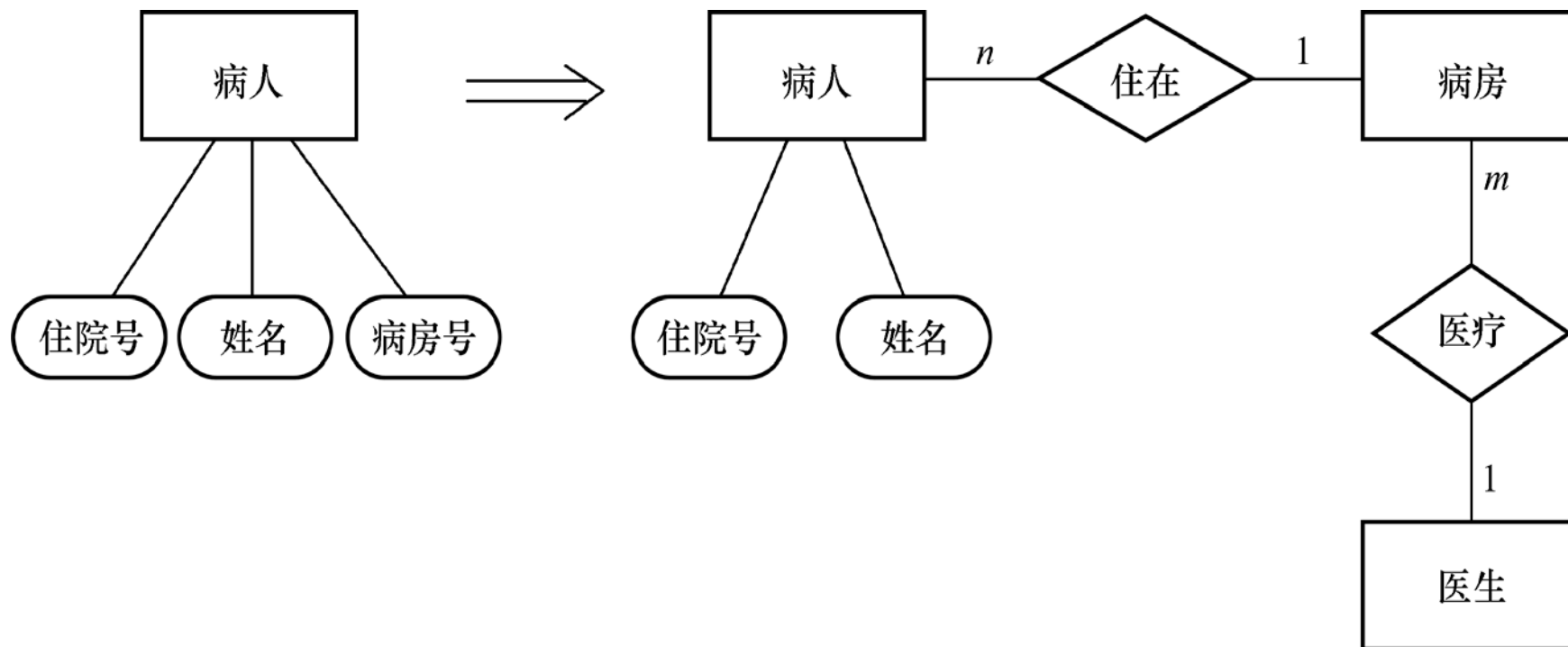
- **属性不能再具有需要描述的性质。**即属性必须是不可分的数据项，不能再由另一些属性组成。
  - **属性不能与其他实体具有联系。**联系只发生在实体之间。
- ❖ 为了简化E-R图的处置，**现实世界中的事物凡能够作为属性对待的，应尽量作为属性。**

## 逐一设计分E-R图（续）



职称作为一个实体

# 逐一设计分E-R图（续）



病房作为一个实体

# 逐一设计分E-R图（续）

## 〔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：

- 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）

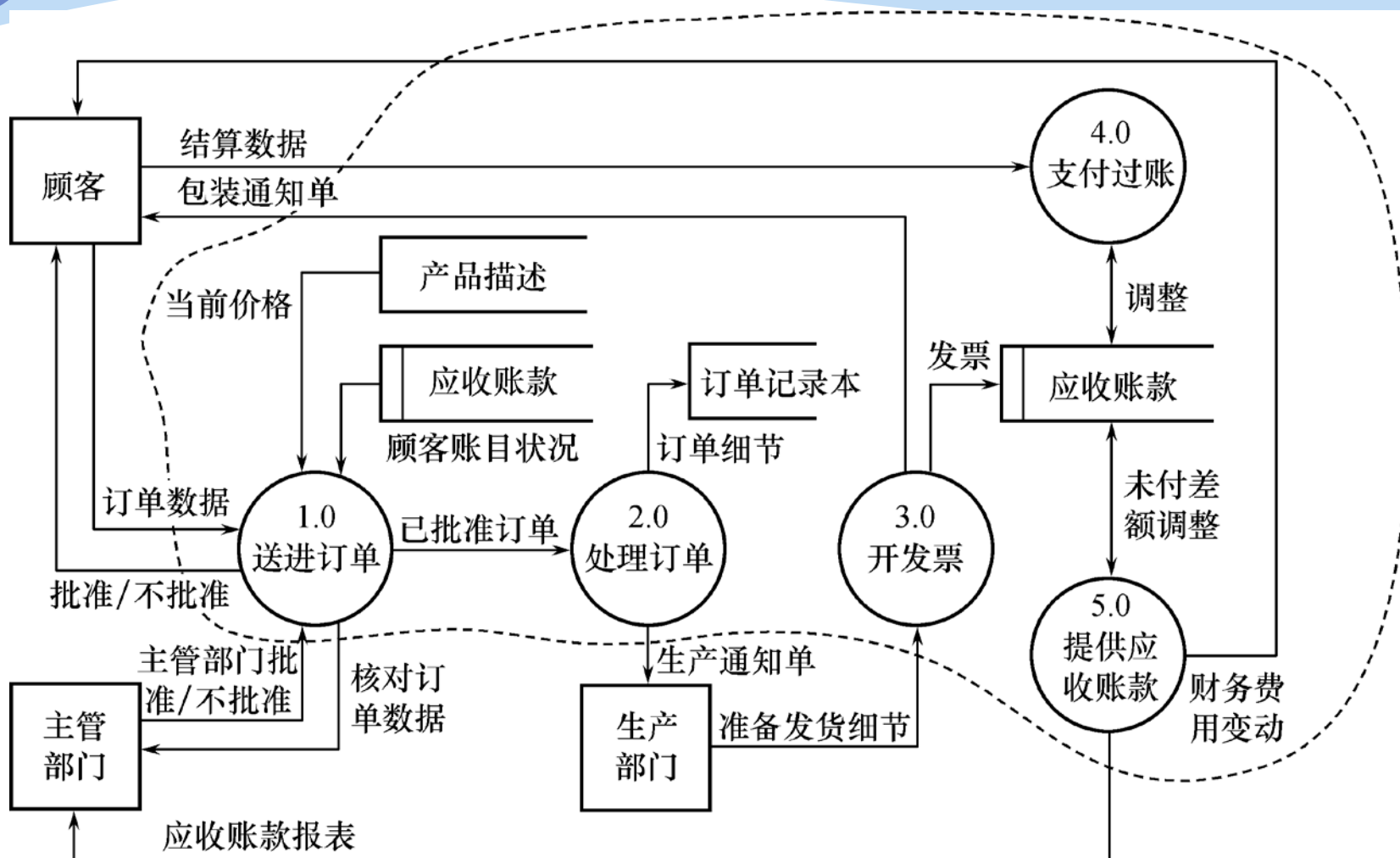


图7.18 销售管理子系统第一层数据流图

## 逐一设计分E-R图（续）

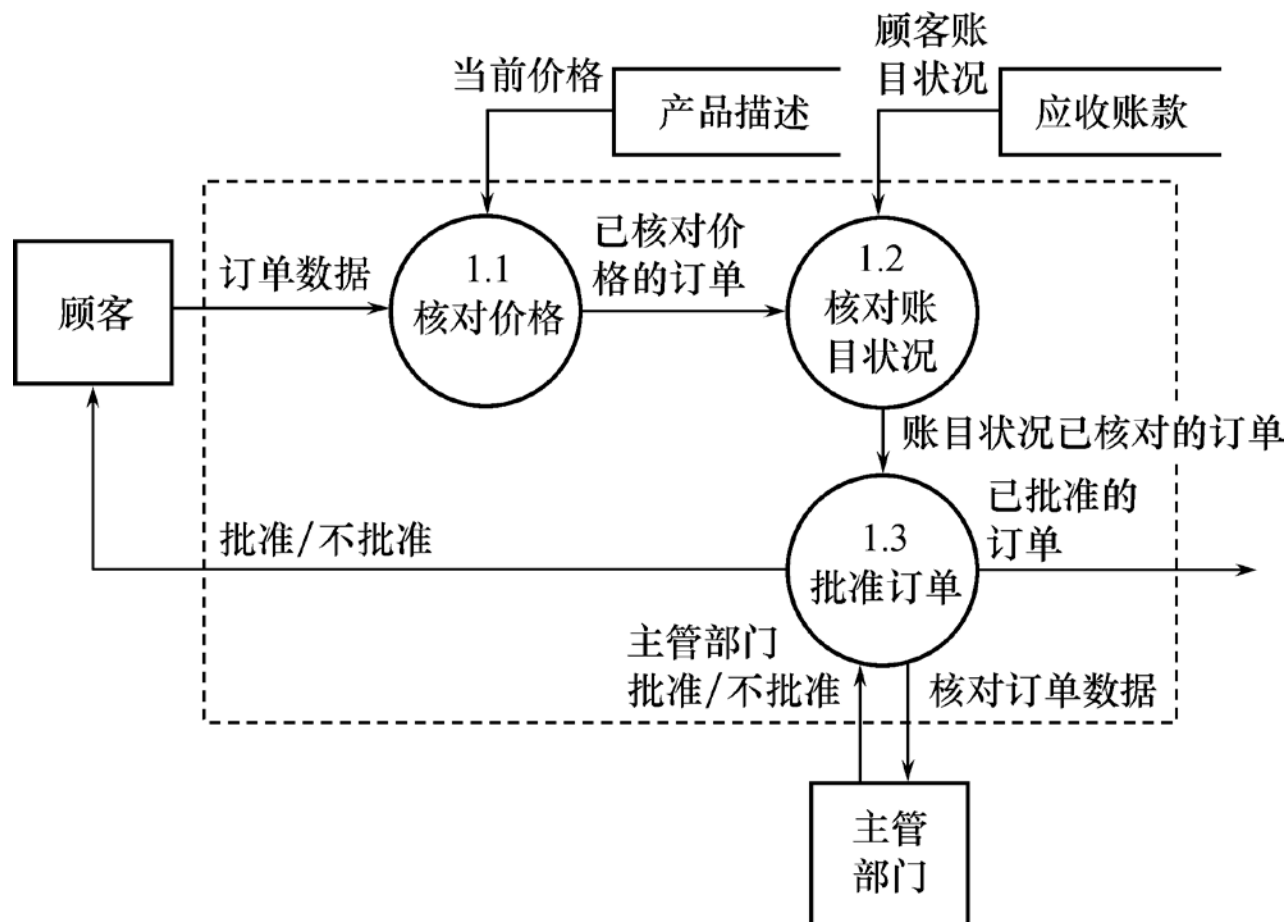
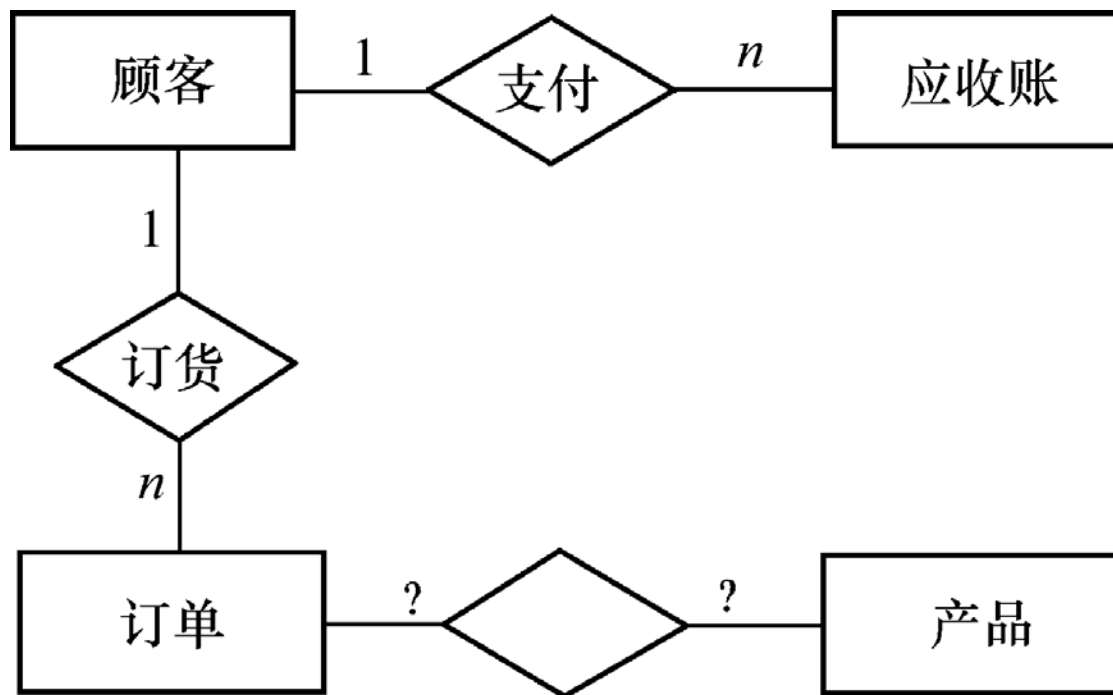


图7.19 接收订单

## 逐一设计分E-R图（续）

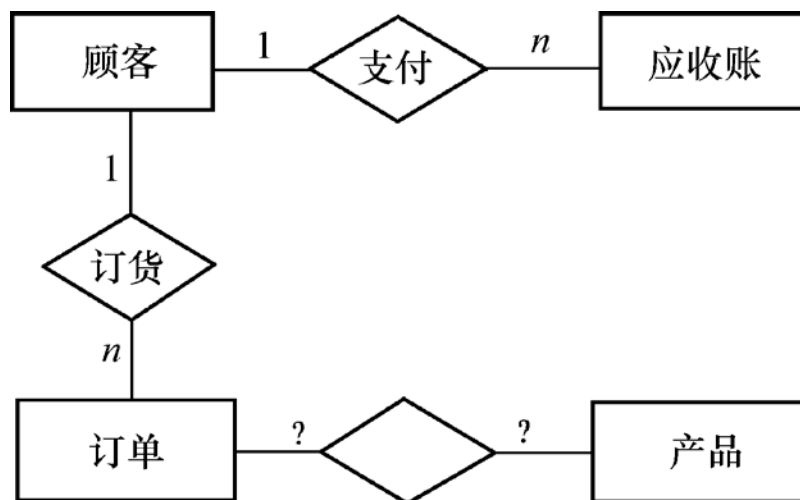


分E-R图的框架

## 逐一设计分E-R图（续）

❖ 遵循两个准则，进行如下调整：

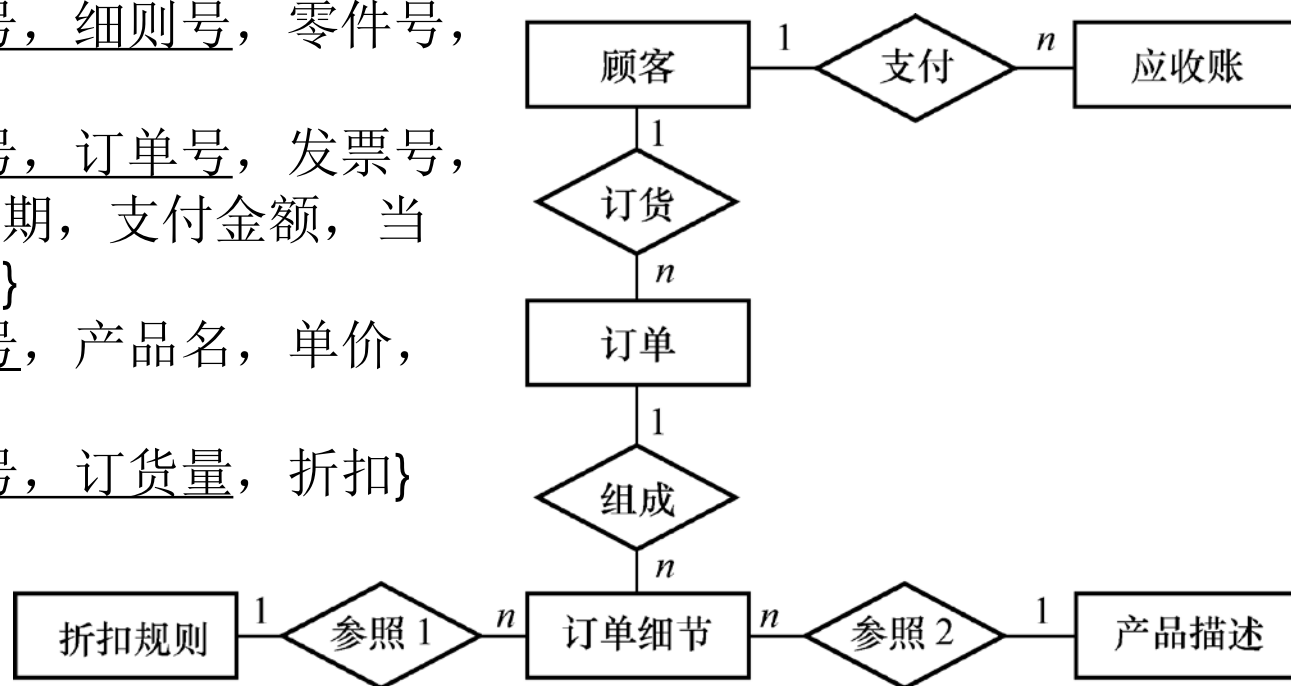
- 订单与订单细节是1 :  $n$ 的联系，订单细节上升为实体
- 原订单和产品的联系实际上是订单细节和产品的联系。
- 工厂对大宗订货给予优惠





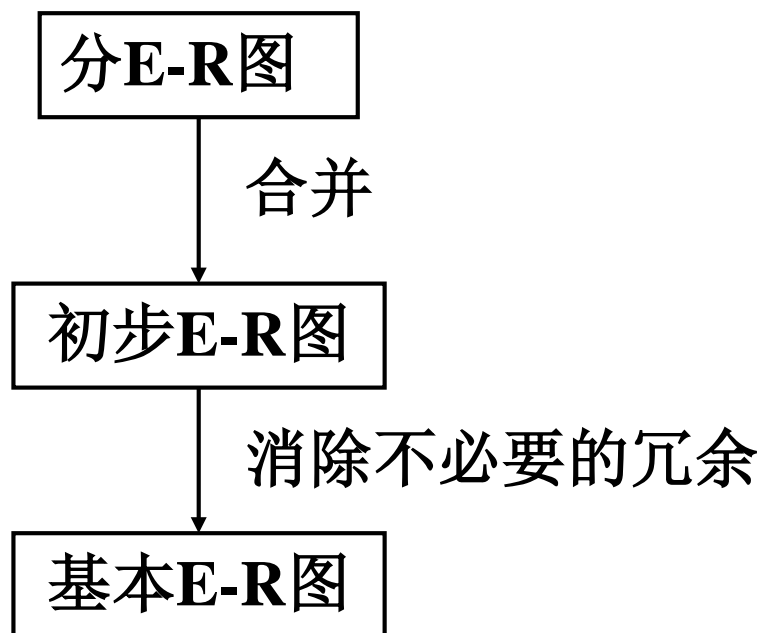
## 逐一设计分E-R图（续）

- ❖ **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}



# 视图的集成

- ❖ 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。



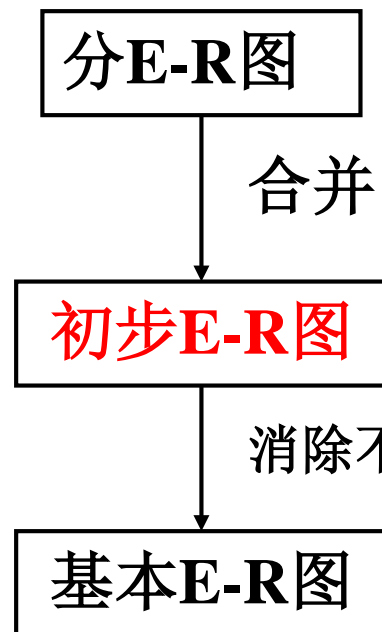
# 合并分E-R图，生成初步E-R图

- ❖ 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
- ❖ 冲突的种类
  - 属性冲突
  - 命名冲突
  - 结构冲突

# 消除不必要的冗余，设计基本E-R图

## ❖ 基本任务

- 消除不必要的冗余，设计生成基本E-R图



可能存在冗余的数据  
和冗余的实体间联系

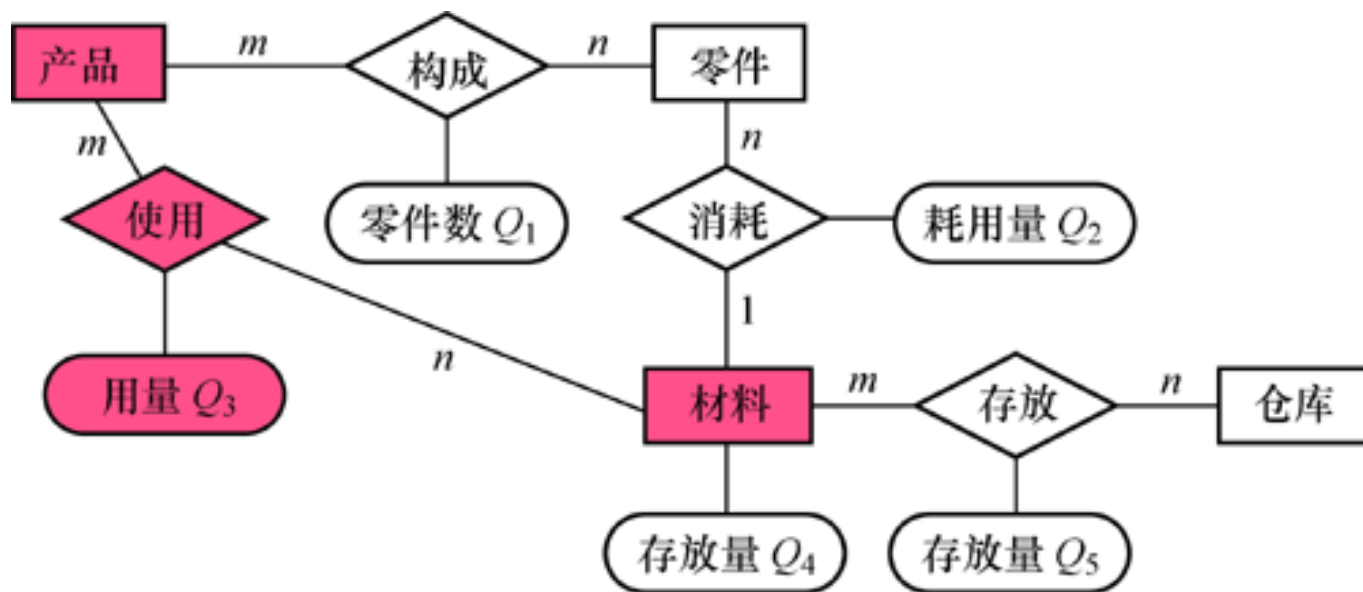
# 冗余

- ❖ **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

## ❖ 分析方法

- 根据数据字典中关于数据项之间的逻辑关系
- $Q3 = Q1 \times Q2$ ,  $Q4 = \sum Q5$ ,  $Q3$ 和 $Q4$ 冗余, “使用”联系冗余



# 消除冗余的方法（续）

- 效率VS冗余信息

- ⑩ 需要根据用户的整体需求来确定

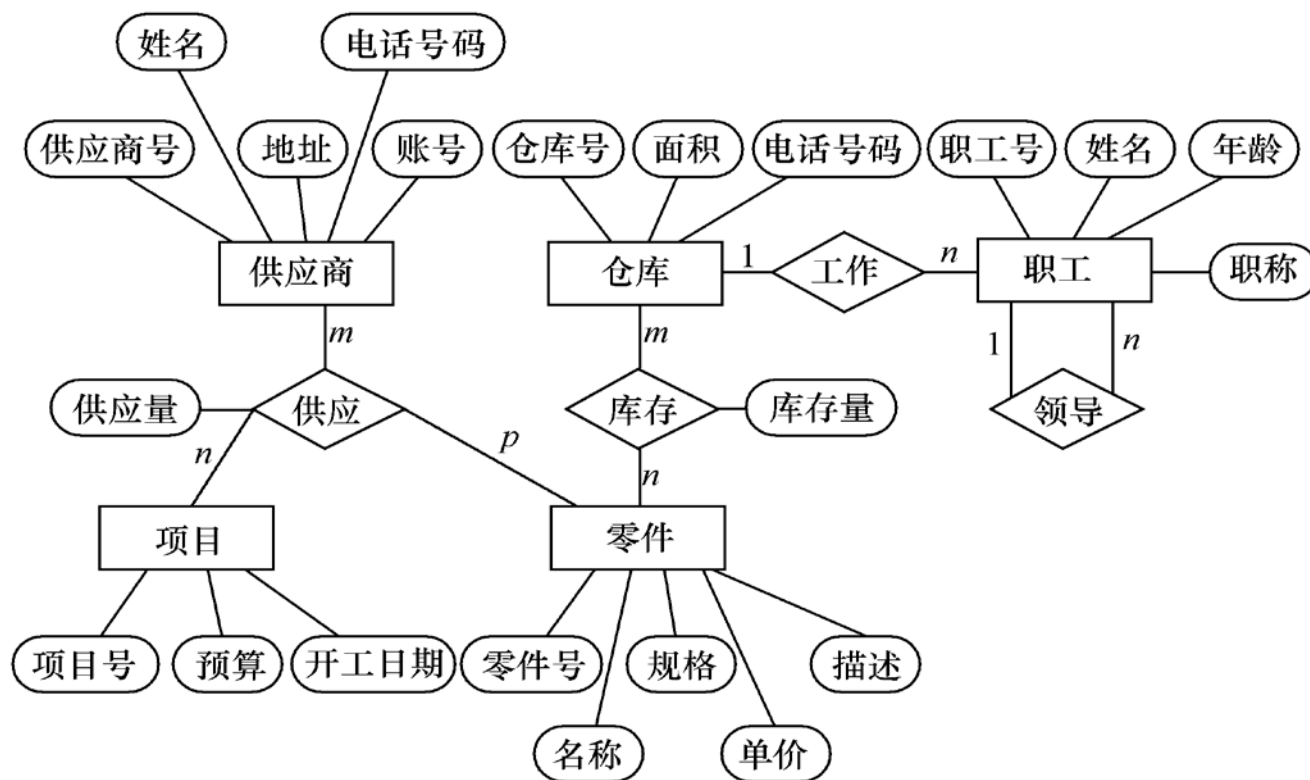
- 若人为地保留了一些冗余数据，则应把数据字典中数据关联的说明作为完整性约束条件

- ⑩  $Q4 = \sum Q5$

- ⑩ 一旦Q5修改后就应当触发完整性检查，对Q4进行修改

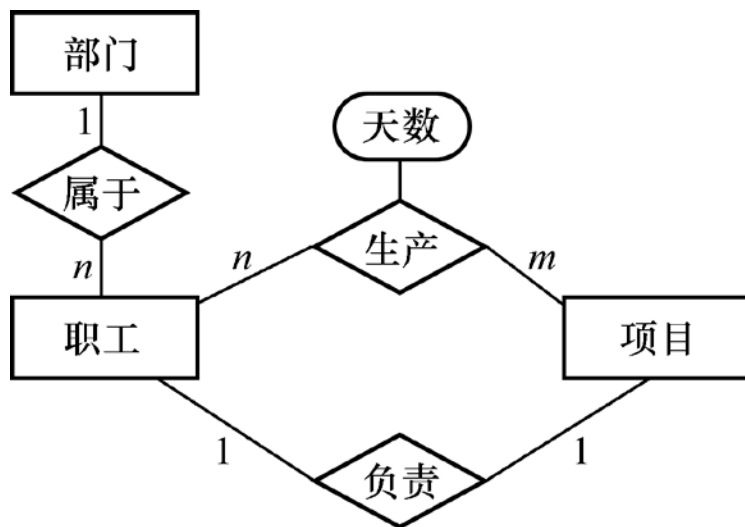
# 消除冗余，设计生成基本E-R图实例

## [实例] 某工厂管理信息系统的视图集成

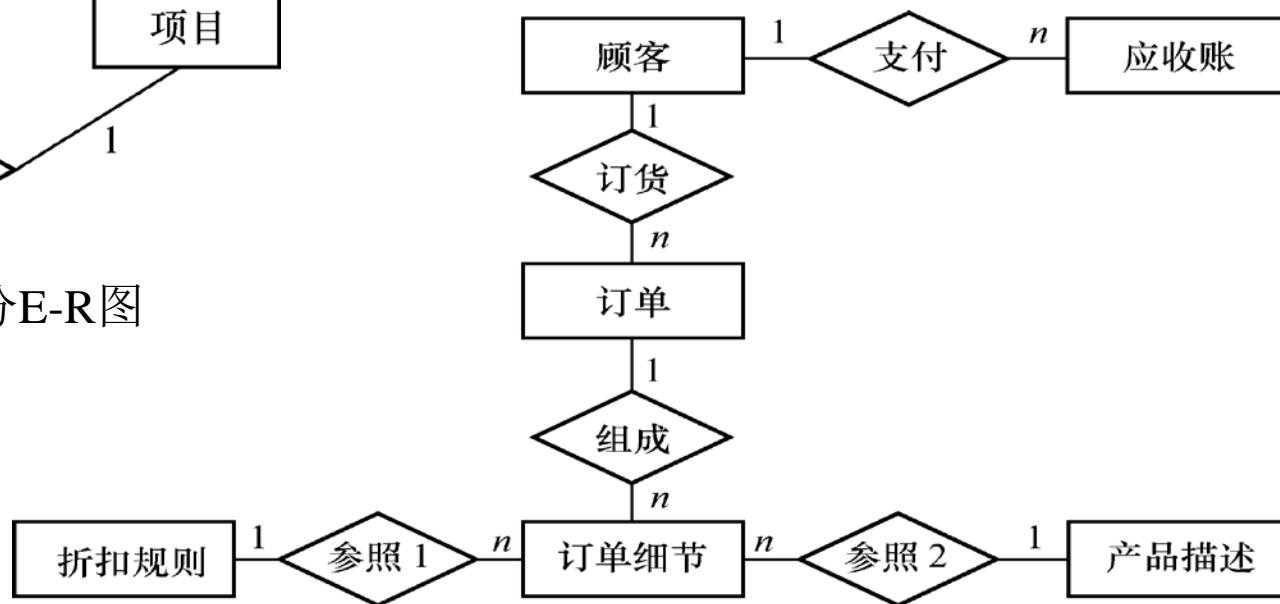




## 消除冗余，设计生成基本E-R图实例（续）

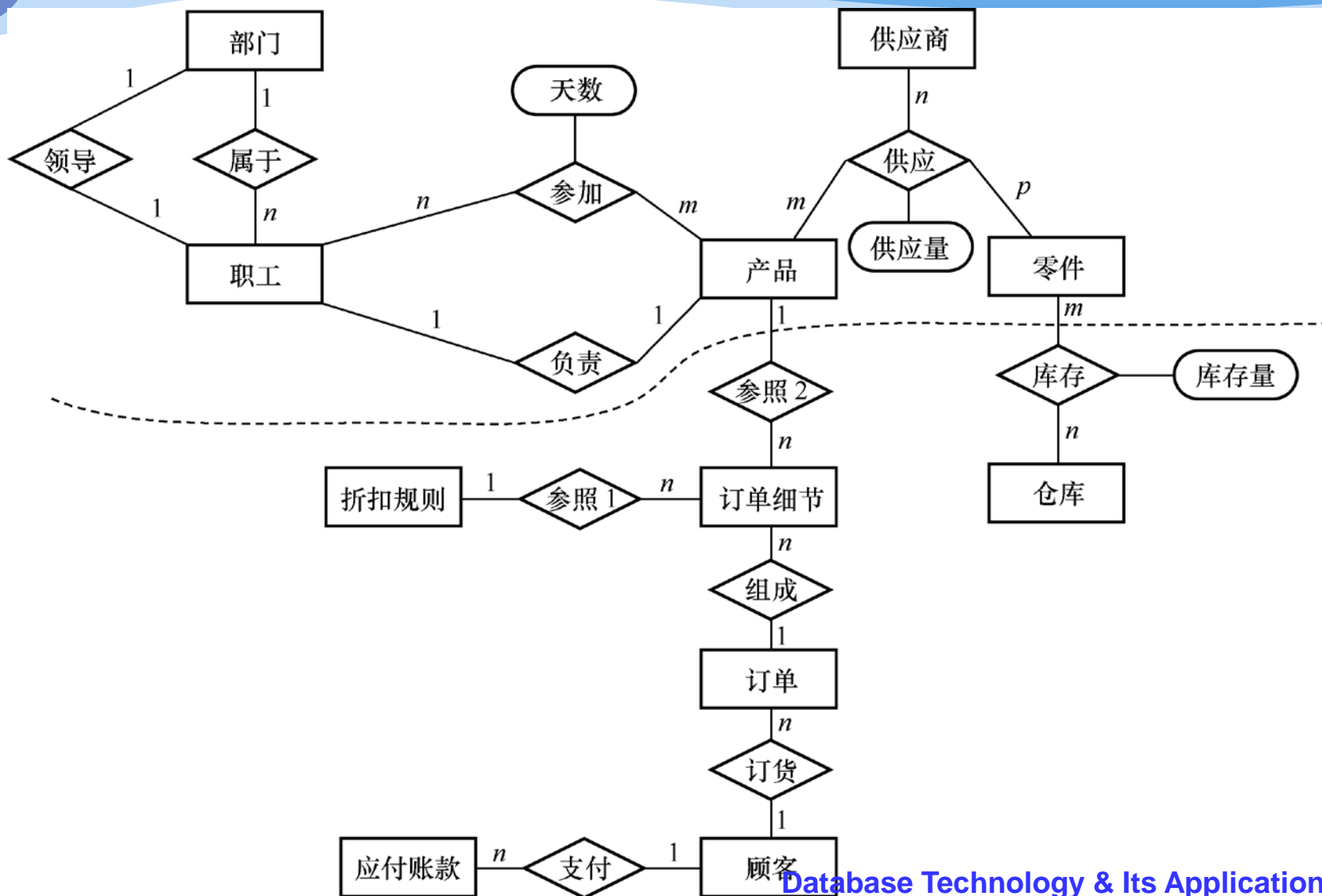


劳动人事管理的分E-R图



销售管理子系统的分E-R图

# 系统的基本E-R



## 消除冗余，设计生成基本E-R图实例（续）

集成过程，解决了以下问题：

- ❖ 异名同义，项目和产品含义相同
- ❖ 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

# 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 数据库的物理设计**

**7.6 数据库的实施和维护**

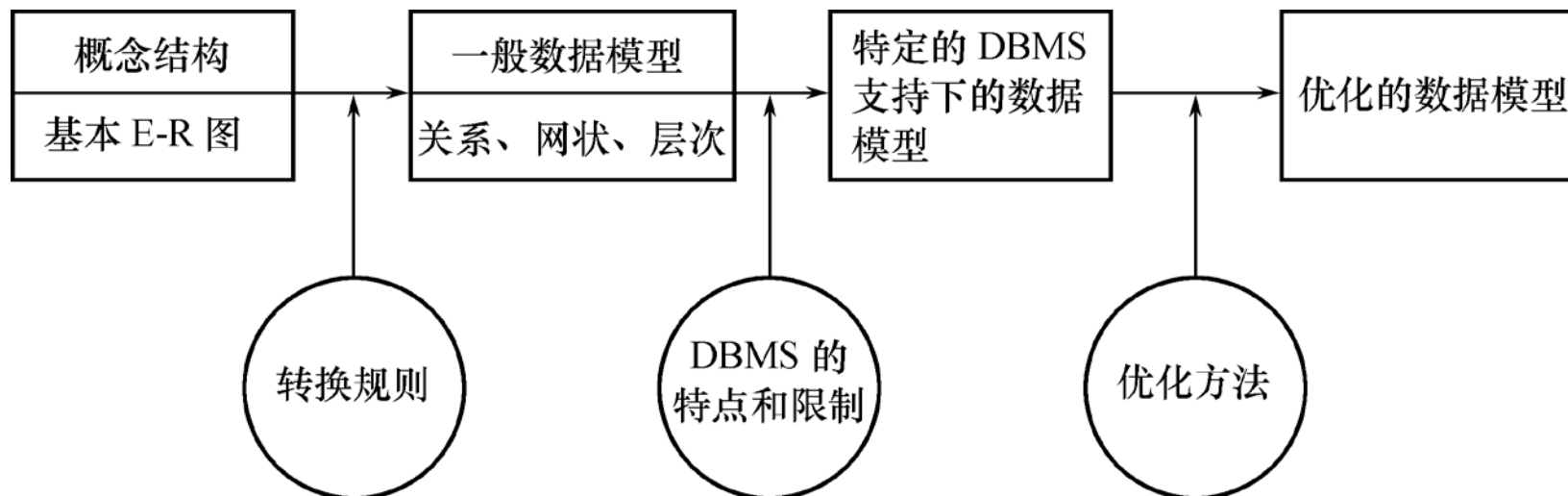
**7.7 小结**

# 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



# E-R图向关系模型的转换（续）

## ❖ 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）

实体型间的联系有以下不同情况：

## (1) 1:1 联系

- 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) 1:n 联系

- 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）

(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）

(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）

(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

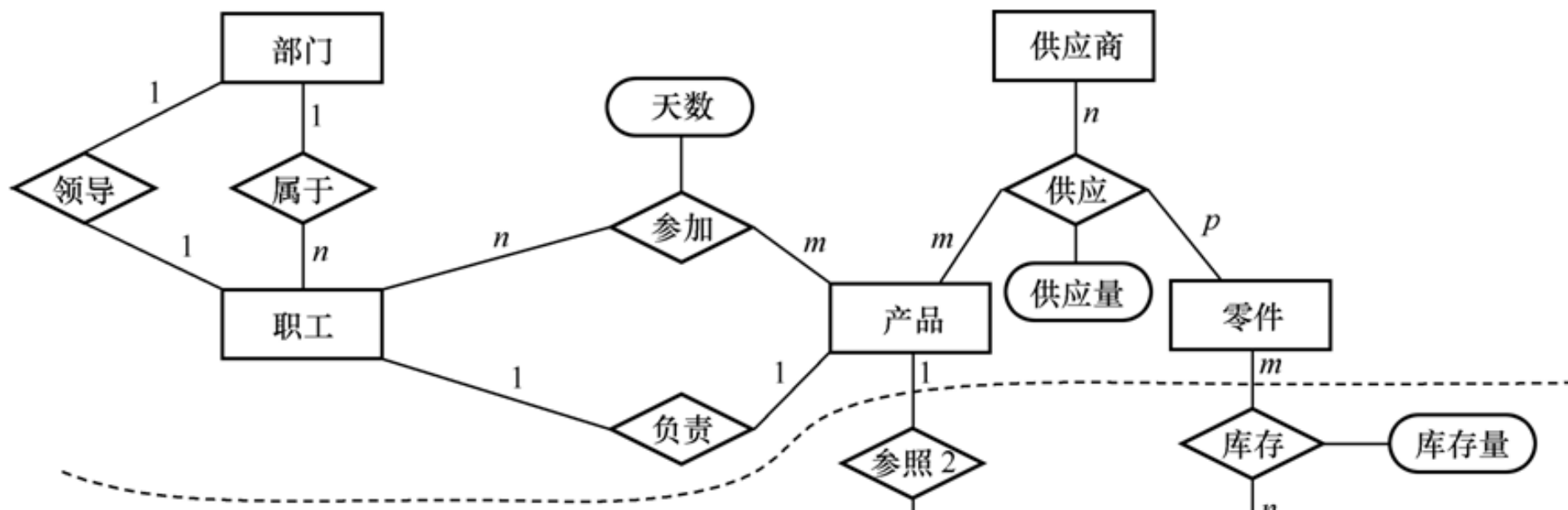
# E-R图向关系模型的转换（续）

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）

[例] 把图7.30中虚线上部的E-R图转换为关系模型



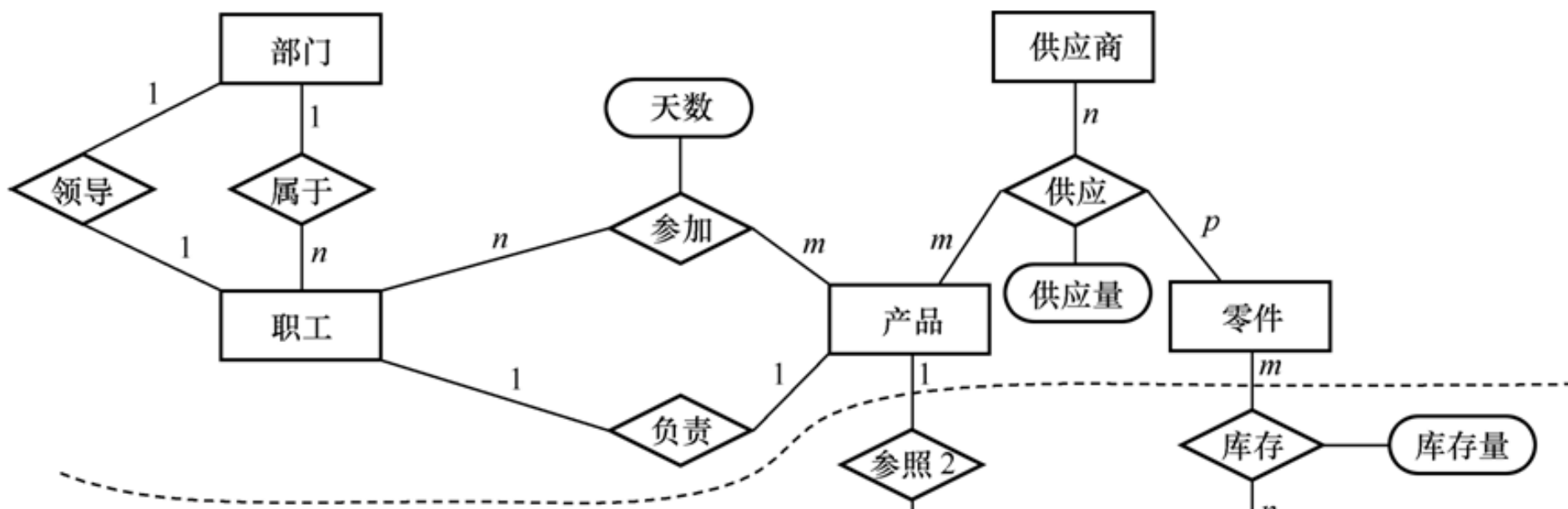
# E-R图向关系模型的转换（续）

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

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

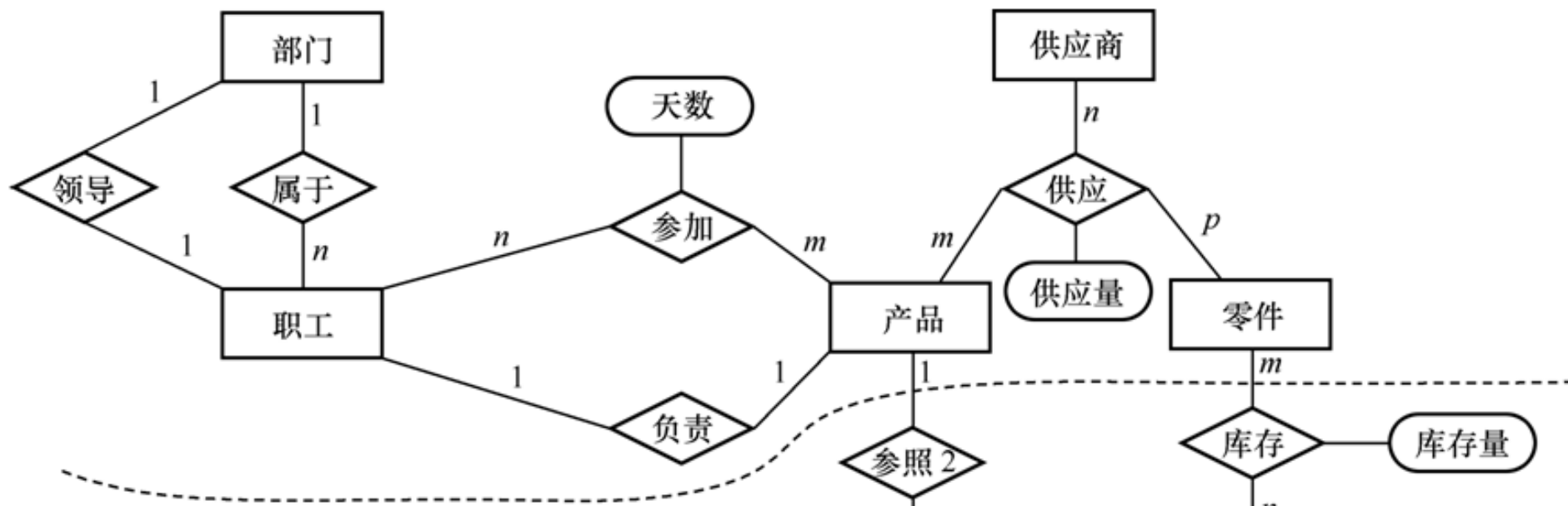


# E-R图向关系模型的转换（续）

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，...）

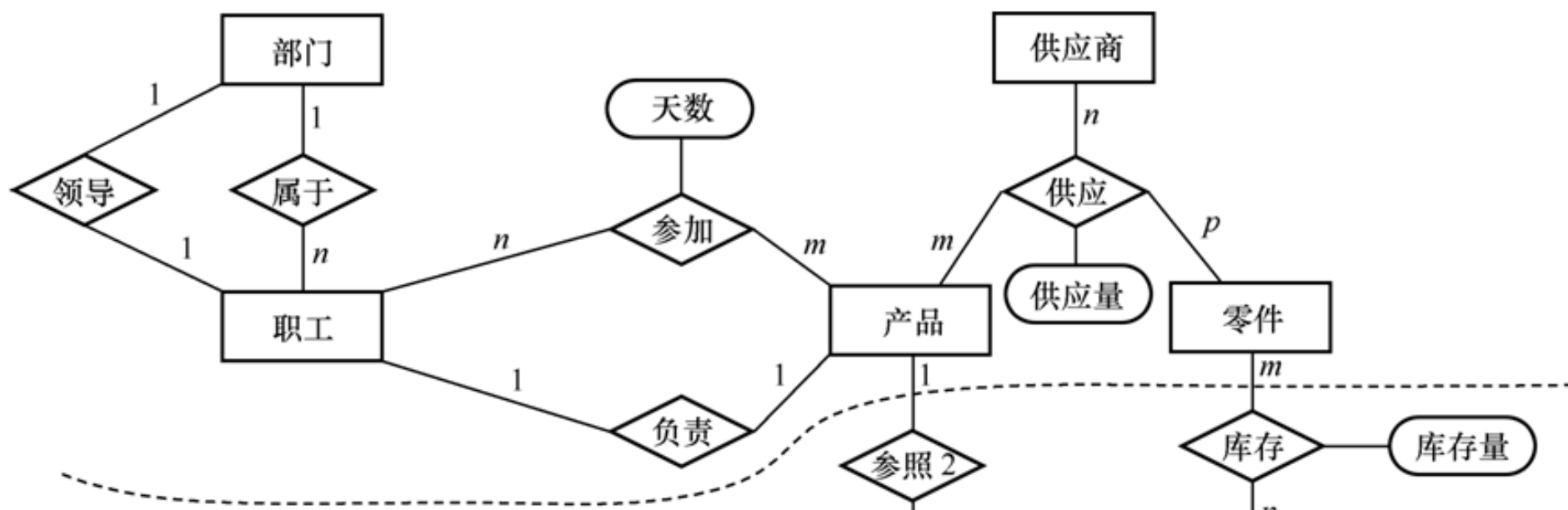
零件（零件号，零件名，...）



# E-R图向关系模型的转换（续）

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

**E-R图向关系模型的转换**

数据模型的优化

# 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，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

### ■ 确定数据依赖

- ⑩ 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

### ■ 消除 冗余的联系

- ⑩ 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）

## ■ 确定所属范式

- ⑩ 按照数据依赖的理论对关系模式逐一进行分析
- ⑩ 考查是否存在部分函数依赖、传递函数依赖、多值依赖等
- ⑩ 确定各关系模式分别属于第几范式

## ■ 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

- ⑩ 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）

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 $\rightarrow$ 英语

学号 $\rightarrow$ 数学

学号 $\rightarrow$ 语文

学号 $\rightarrow$ 平均成绩

(英语, 数学, 语文) $\rightarrow$ 平均成绩


# 数据模型的优化（续）

显然有：

学号 $\rightarrow$ (英语,数学,语文)

因此该关系模式中存在传递函数信赖，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



例，设一个海军基地要建立一个舰队管理信息系统，它包括如下两个方面的信息：

### 1. 舰队方面

舰队：舰队名称，基地地点，舰艇数量


舰艇：编号，舰艇名称，舰队名称

### 2. 舰艇方面

舰艇：舰艇编号，舰艇名，武器名称

武器：武器名称，武器生产时间，舰艇编号

官兵：官兵证号，姓名，舰艇编号

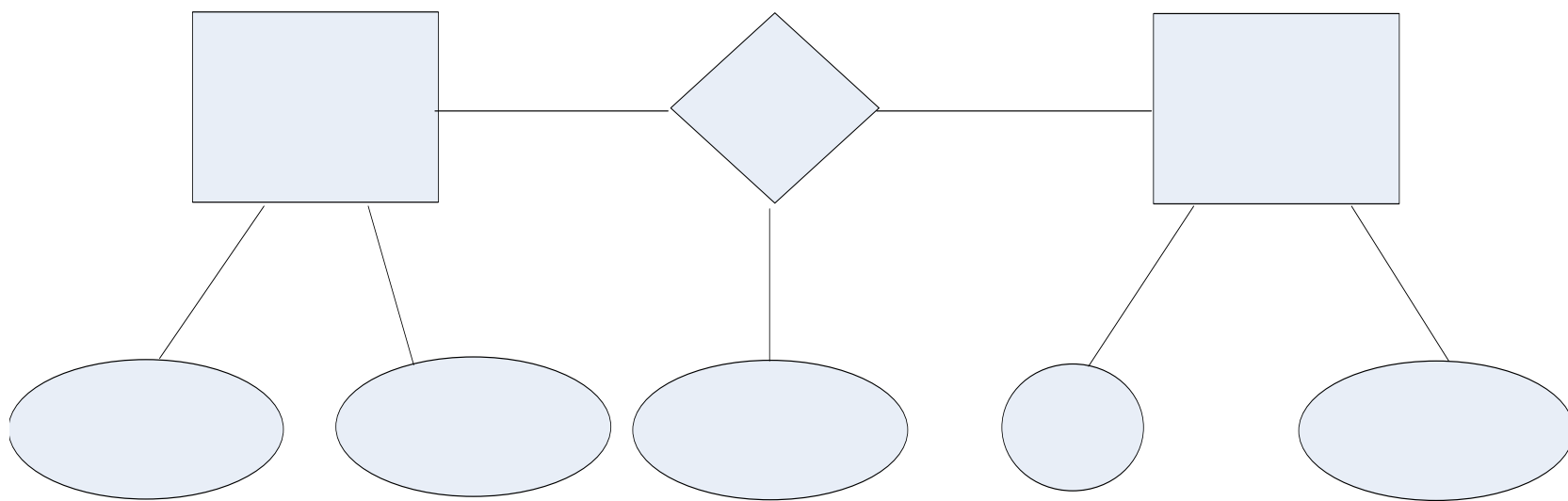


其中，一个舰队拥有多艘舰艇，一艘舰艇属于一个舰队；一艘舰艇安装多种武器，一种武器可安装于多艘舰艇上；一艘舰艇有多个官兵，一个官兵只属于一艘舰艇。

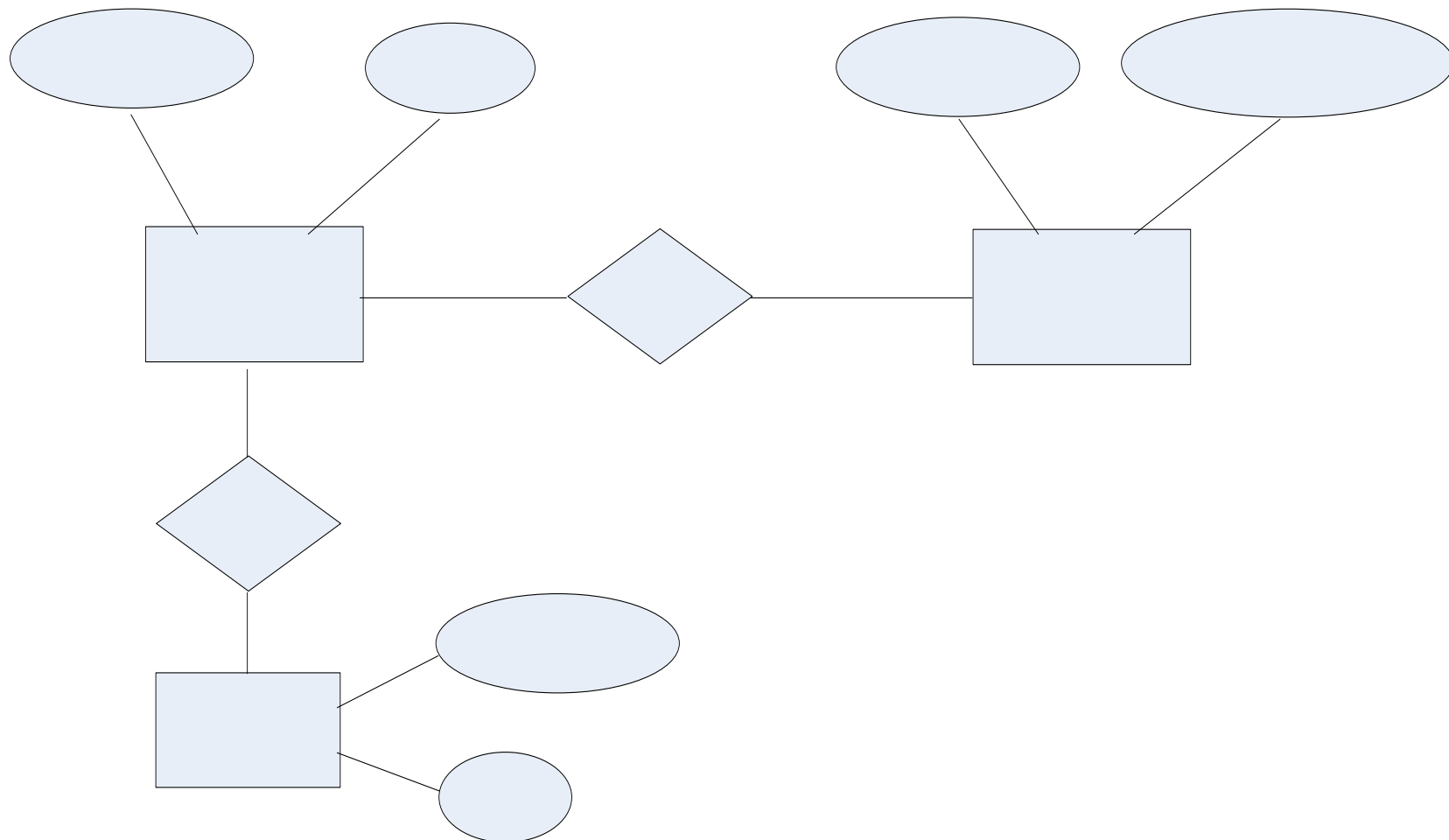
请完成如下设计：

- (1) 分别设计舰队和舰艇两个局部E-R图。
- (2) 将上述两个局部E-R图合并为一个全局E-R图。
- (3) 将该全局E-R图转换为关系模式。

## (1) 舰队局部E-R图如下面图所示

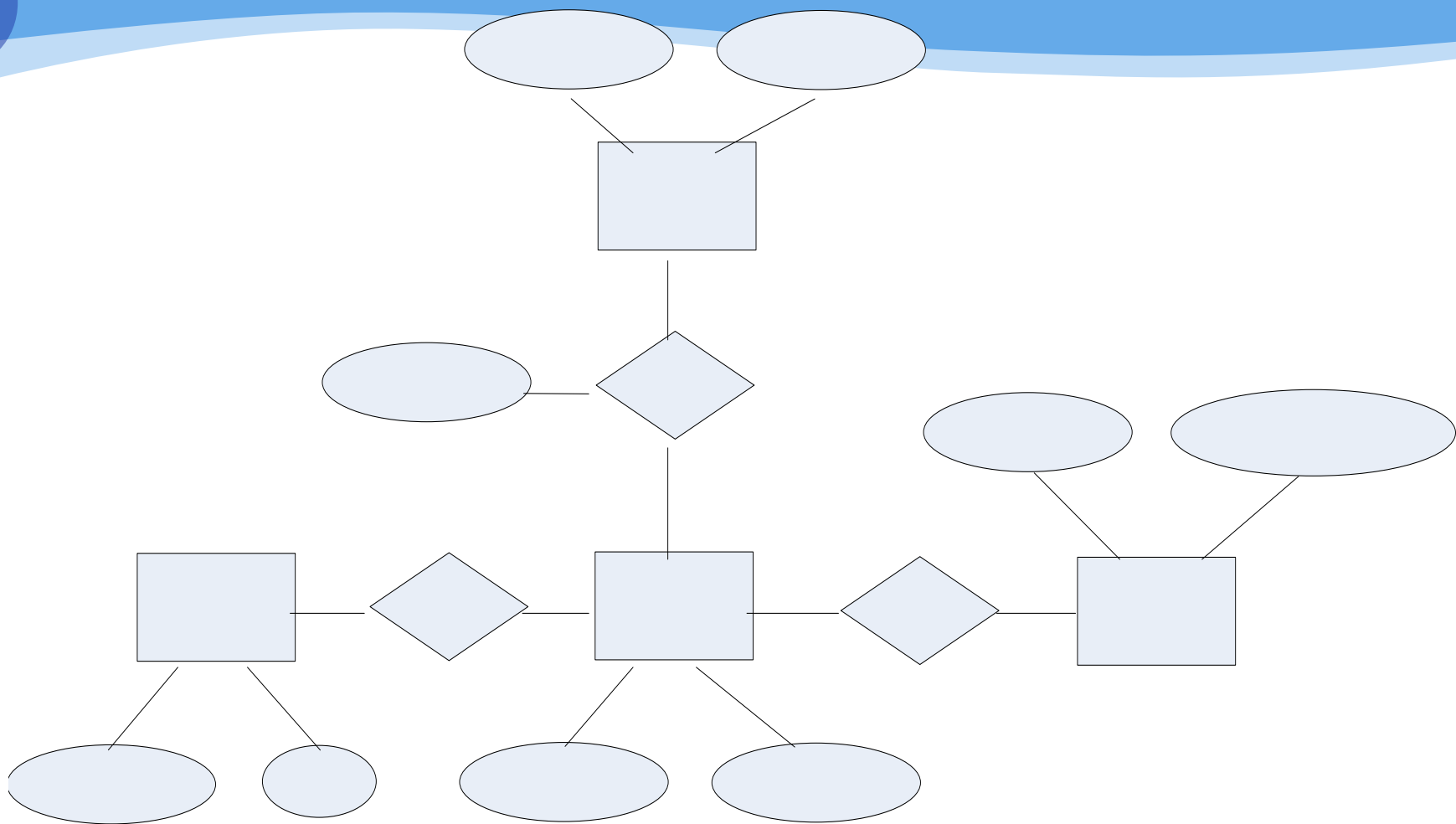


舰艇局部E-R图如下面图所示。





(2) 合并结果如图所示。



(3) 转换的关系模式如下：

舰队 (舰队名称, 基地地点)

舰艇 (舰艇编号, 舰艇名称, 舰队名称, 舰艇数量)

官兵 (官兵证号, 姓名, 舰艇编号)

武器 (武器名称, 武器生产时间)

安装 (舰艇编号, 武器名称)

# 练习

针对QQ应用，设计ER图和关系表

提示 应包含如下信息（但不限于）：

用户

群

聊天记录

# QQ

## ❖ 实体

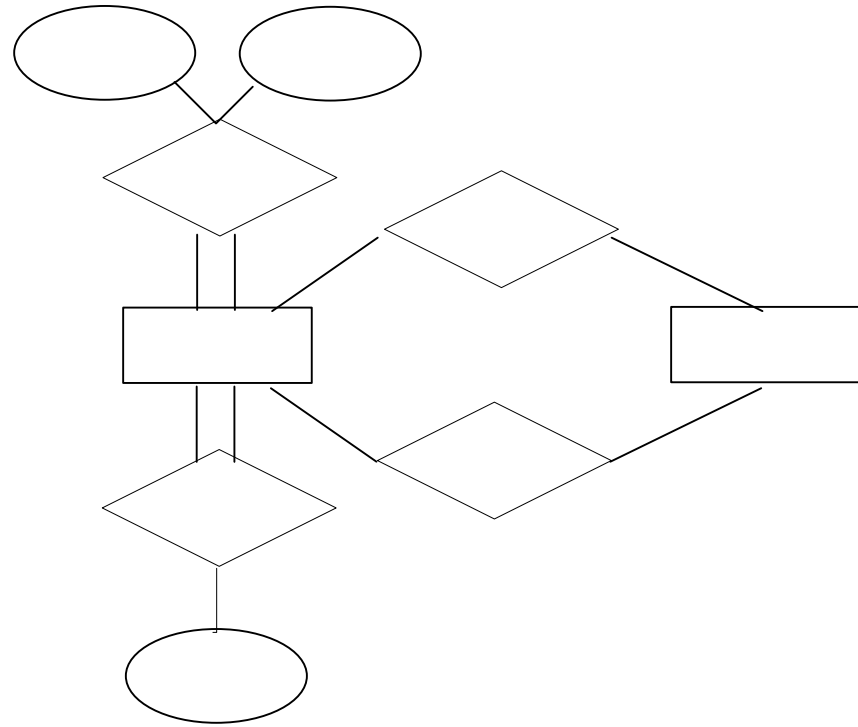
- 用户
- 群

## ❖ 联系

- 关系（用户-用户）
- 聊天（用户-用户）
- 加入（用户-群）
- 管理（用户-群）

# QQ

- ❖ 用户（qq号，昵称，.....）
- ❖ 群（群号，群名称，管理员qq号，.....）
- ❖ 关系（用户A qq号，用户B qq号，关系类型，.....）
- ❖ 聊天（发送用户qq号，接受用户qq号，发送时间，消息内容，.....）
- ❖ 加入（用户qq号，群号，.....）



# 作业

## ❖ 设计数据库模式（ER图、关系表）

- 淘宝网
- 新浪微博