

数据库设计



China university of
Mining and Technology-Beijing

什么是数据库设计

- 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。



为什么需要设计数据库？

- 良好的数据库设计：

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。

- 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

- 糟糕的数据库设计：

- ☐ 数据冗余、存储空间浪费
- ☐ 内存空间浪费
- ☐ 数据更新和插入的异常



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



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



数据库设计专业人员应具备哪些知识？

- 数据库的基本知识和数据库设计技术
- 计算机科学的基础知识和程序设计的方法和技巧
- 软件工程的原理和方法
- 应用领域的知识



数据库设计方法

- 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问题，增加了维护代价



数据库设计方法

- 规范设计法

- 基本思想：过程迭代和逐步求精
- 典型方法
 - 新奥尔良（New Orleans）方法
 - 将数据库设计分为四个阶段
 - 基于E-R模型的数据库设计方法
 - 概念设计阶段广泛采用
 - 3NF（第三范式）的设计方法
 - 逻辑阶段可采用的有效方法
 - ODL（Object Definition Language）方法
 - 面向对象的数据库设计方法

- 计算机辅助设计

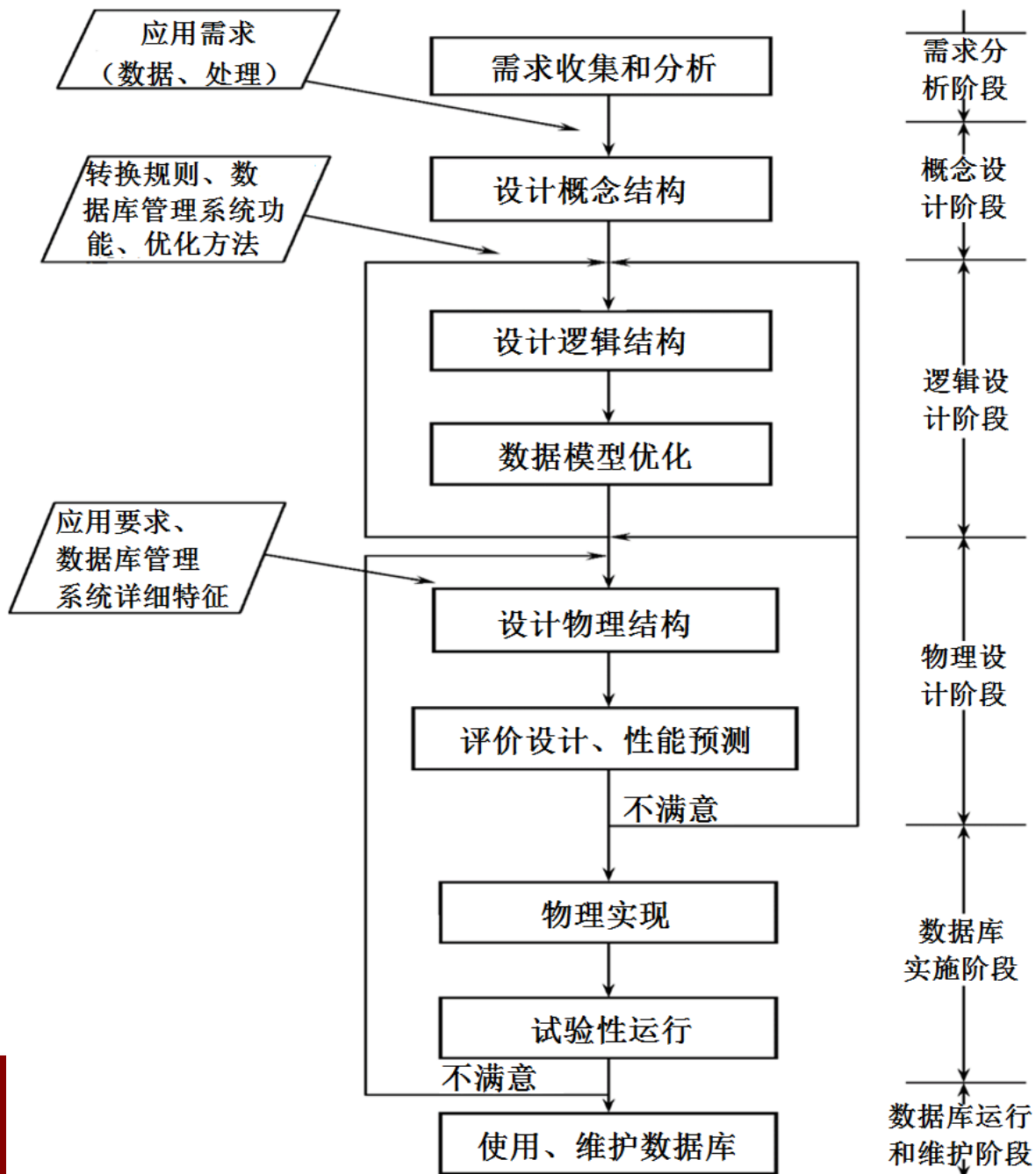
- ORACLE Designer 2000
- SYBASE PowerDesigner



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





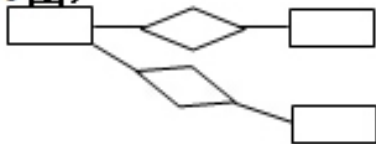
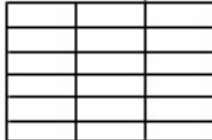
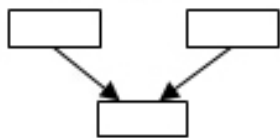


设计阶段	设计描述
需求分析	数字字典、全系统中数据项、数据结构、数据流、数据存储的描述
概念结构设计	概念模型 (E-R 图)  数据字典
逻辑结构设计	某种数据模型 关系  非关系 
物理结构设计	存储安排 存取方法选择 存取路径建立 
数据库实施	创建数据库模式 装入数据 数据库试运行 
数据库运行和维护	性能监测、转储/恢复、数据库重组和重构

图 数据库设计各个阶段的数据设计描述



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



需求分析

- 需求分析就是分析用户的需要与要求
 - 需求分析是设计数据库的起点
 - 需求分析的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
- 需求分析的任务
 - 通过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，不能仅仅按当前应用需求来设计数据库。



需求分析-重点

- 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-调查步骤

- (1) 调查组织机构情况
- (2) 调查各部门的业务活动情况。
- (3) 在熟悉业务活动的基础上，协助用户明确对新系统的各种要求。
- (4) 确定新系统的边界。



需求分析-调查方法

- 做需求调查时，往往需要同时采用多种方法
 - 无论使用何种调查方法，都必须有用户的积极参与和配合
 - 设计人员应该和用户取得共同的语言，帮助不熟悉计算机的用户建立数据库环境下的共同概念，并对设计工作的最后结果共同承担责任
- 常用调查方法
 - (1)跟班作业
 - (2)开调查会
 - (3)请专人介绍
 - (4)询问
 - (5)设计调查表请用户填写
 - (6)查阅记录



需求分析-难点

- 确定用户最终需求
 - 用户缺少计算机知识
 - 设计人员缺少用户的专业知识
- 解决方法
 - 设计人员必须不断深入地与用户进行交流



需求分析-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
- 进一步分析与表达这些需求



需求分析-分析方法

- 分析和表达用户的需求的常用方法
 - 自顶向下的结构化分析方法（Structured Analysis，简称SA方法）
- SA方法从最上层的系统组织机构入手，采用自顶向下、逐层分解的方式分析系统，并用数据字典描述系统。



需求分析-数据字典

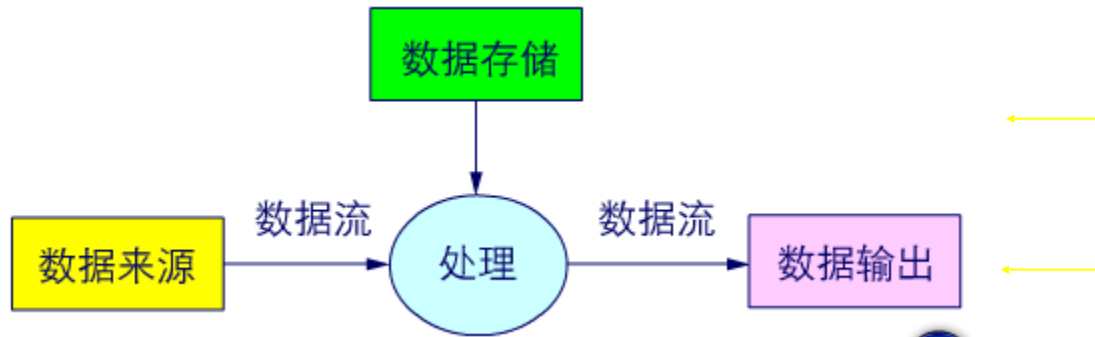
- 数据字典的用途
 - 是各类数据描述的集合
 - 进行详细的数据收集和数据分析所获得的主要结果
- 数据字典的内容
 - 数据项
 - 数据结构
 - 数据流
 - 数据存储
 - 处理过程



- 各项的描述：
 - 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, 与其它数据项的逻辑关系, 数据项之间的联系}
 - 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}}
 - 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}
 - 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}}



1. 任何一个系统都抽象为：



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，再将每个子功能继续分解，直到把系统的工作过程表达清楚为止。

(2) 分解数据

- 在处理功能逐步分解的同时，其所用的数据也逐级分解，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

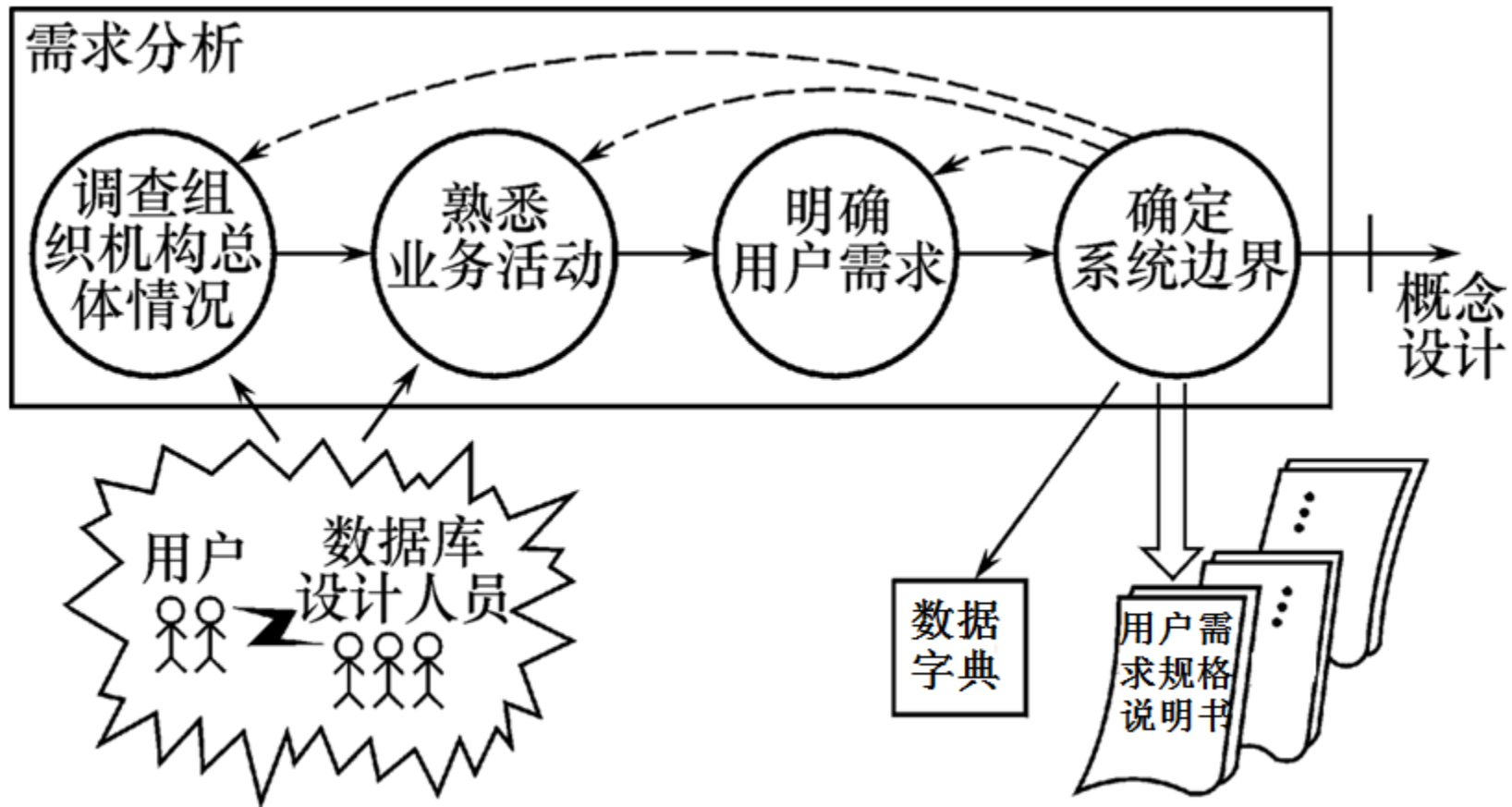


需求分析-数据字典

- 数据字典是关于数据库中数据的描述，是元数据，而不是数据本身
- 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

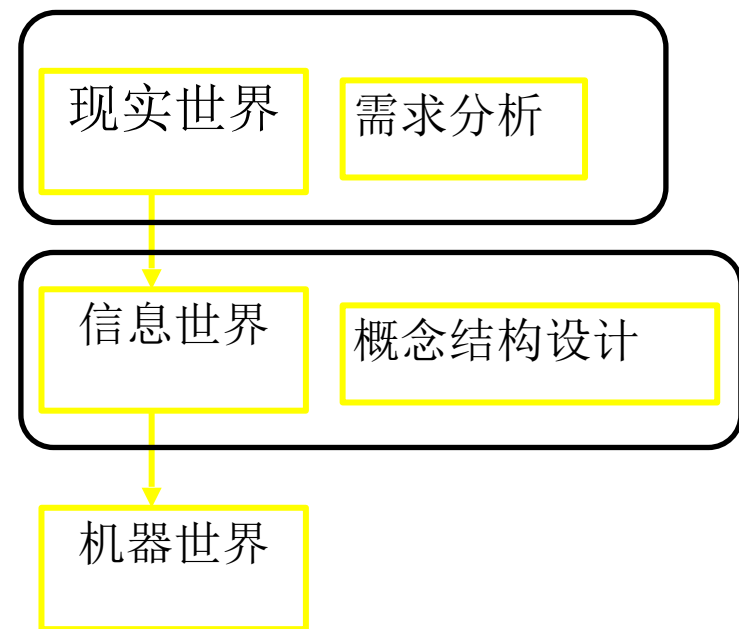


需求分析-小结



概念结构设计

- 需求分析阶段描述的用户应用需求是现实世界的具体需求
- 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
- 概念结构设计是整个数据库设计的关键



概念结构设计

- 概念结构设计的特点

- 能真实、充分地反映现实世界
- 易于理解
- 易于更改
- 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

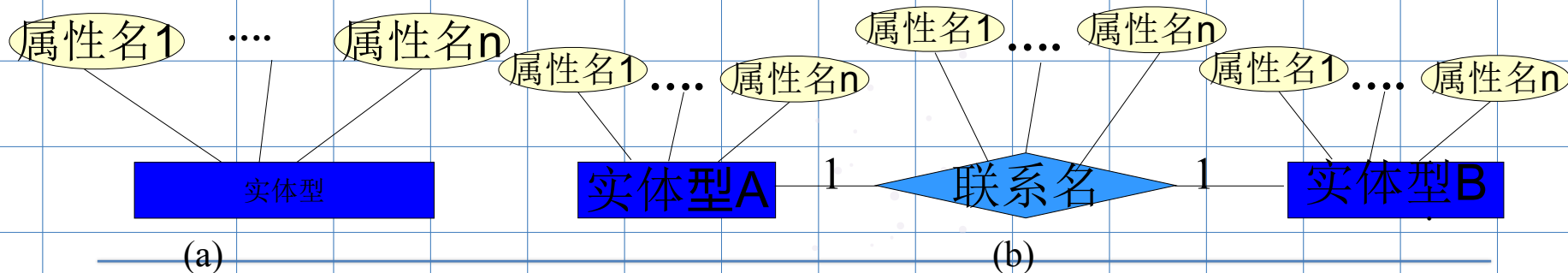
- E-R模型



概念模型的表示方法: 实体联系表示法——E-R图

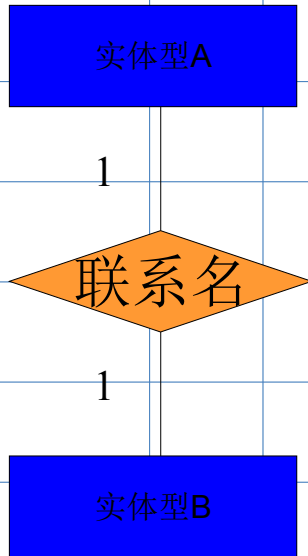
E-R图的表示方法:

- ❖ 实体用方框表示。
- ❖ 联系用菱形表示, 并且用边将其与有关的实体连接起来, 并在边上标有联系的类型;
- ❖ 属性用椭圆表示, 并且用边将其与相应的实体连接起来。
- ❖ 对于有些联系, 其自身也会有某些属性, 同实体与属性的连接类似, 将联系与其属性用边连接起来。

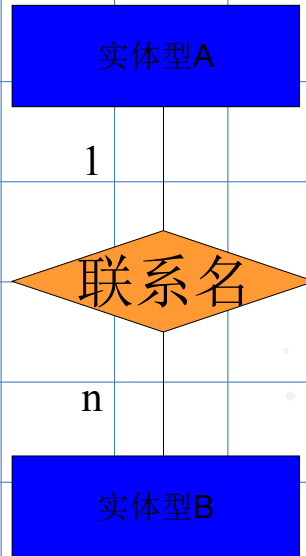


(1) 两个不同实体集之间联系的画法

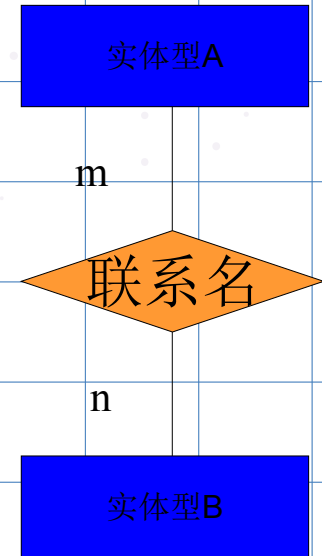
两个不同实体集之间存在1:1、1:n、m:n联系。



(a) 1: 1联系



(b) 1: n联系



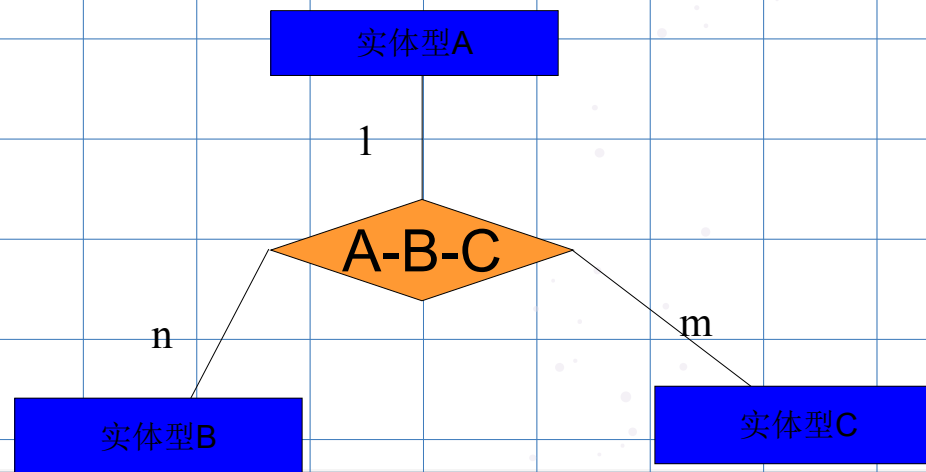
(c) m: n联系



(2) 两个以上不同实体集之间联系的画法

两个以上不同实体集之间可能存在各种关系，以3个不同实体集A、B、C为例，它们之间的典型关系有： $1:n:m$ 和 $r:n:m$ 联系。

❖ **$1:n:m$ 联系**：表示A和B之间是 $1:n$ （一对多）联系；A和C之间是 $1:m$ （一对多）联系；B和C之间是 $n:m$ （多对多）联系。



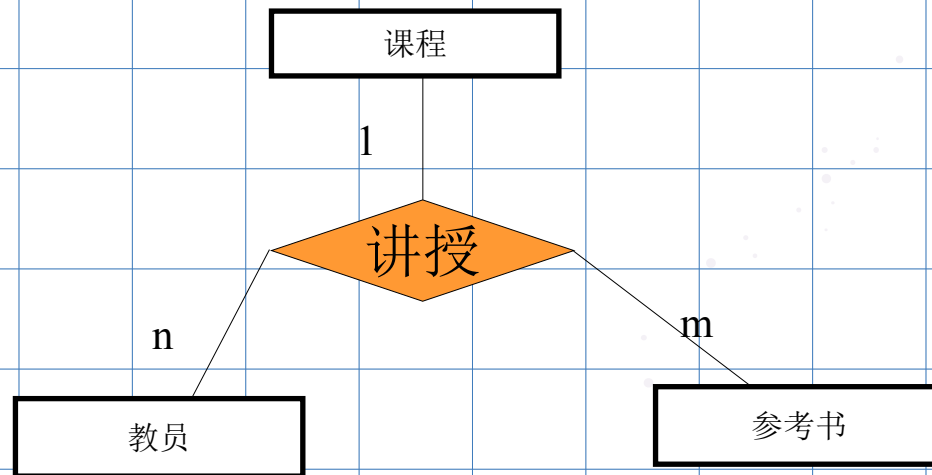
$1:n:m$ 联系的图形表示

示



China university of
Mining and Technology-Beijing

例如：语义描述：对于课程、教师与参考书三个实体型，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的联系。

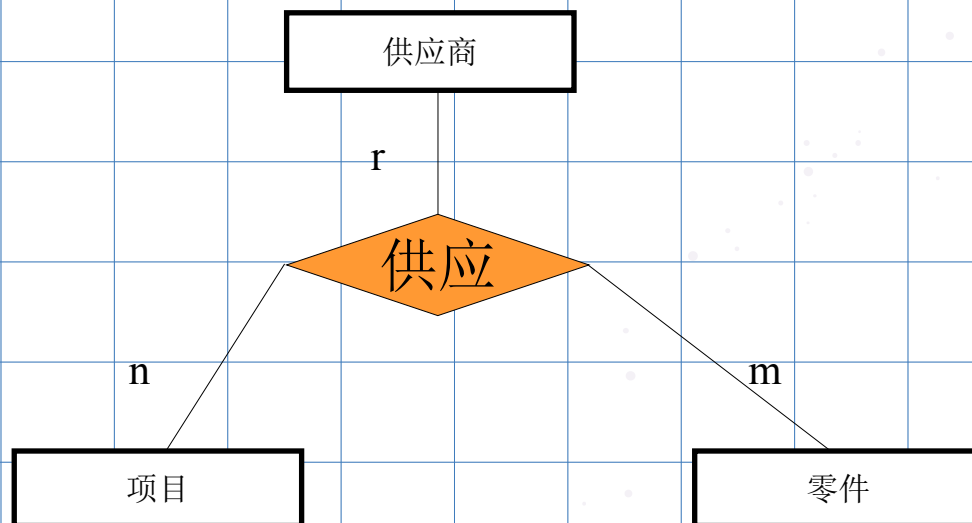


课程与教师、参考书之间的一对多的联系的图形表示



China university of
Mining and Technology-Beijing

例如有语义描述：有三个实体，供应商、项目、零件，一个供应商可以供给多个项目多种零件，而每个项目可以使用多个供应商供应的零件，每种零件可以由不同的供应商供应。则称供应商、项目、零件三者之间是多对多的联系。

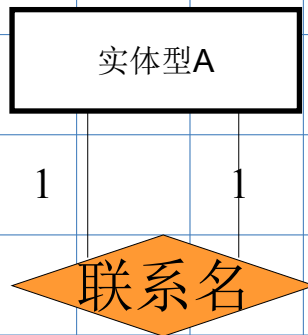


供应商、项目、零件三者之间多对多联系的图形表示

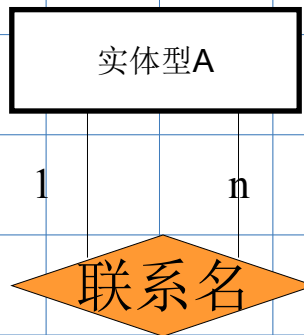


(3) 同一实体集内的二元联系的画法

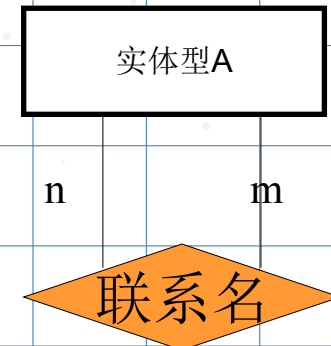
同一实体集内的二元联系：表示同一实体集内实体之间相互联系。有1:1、1:n、n:m联系。



(a) 1: 1联系



(b) 1: n联系



(c) n: m联系

例如：职工实体集中，领导与被领导的联系是1:n；婚姻联系是1:1.



怎样设计E-R图

根据实际问题的语义描述，设计E-R图的基本步骤：

- (1) 抽取实体并定义实体名，定义实体包含的属性名。
- (2) 定义实体间的联系名、联系类型、定义联系包含的属性名，绘制实体间联系的E-R图。
- (3) 将实体间联系的简单的E-R图与实体的属性、联系的属性合并在一起，绘制形成总体E-R图。

➤ 例 某大学选课管理中，学生可根据自己的情况选修课程。每名学生可同时选修多门课程，每位老师可讲授多门课程，每门课程可由多位老师讲授。画出对应的E-R图。

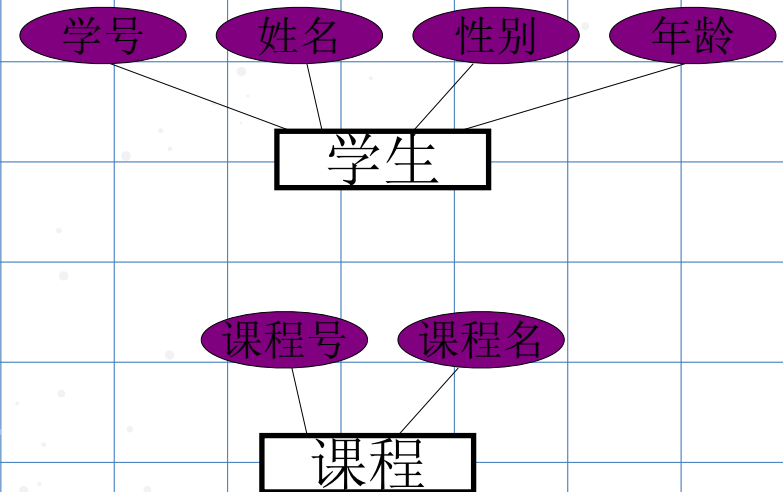
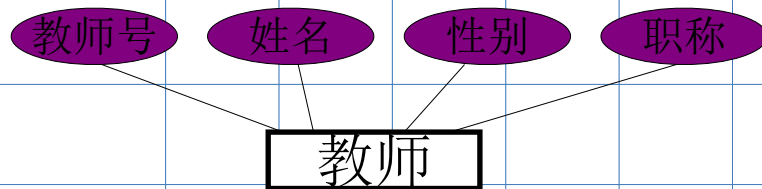


(1) 定义学生选课管理中涉及的实体及属性:

学生: 属性有学号、姓名、性别、年龄

教师: 属性有教师号、姓名、性别、职称

课程: 属性有课程号、课程名

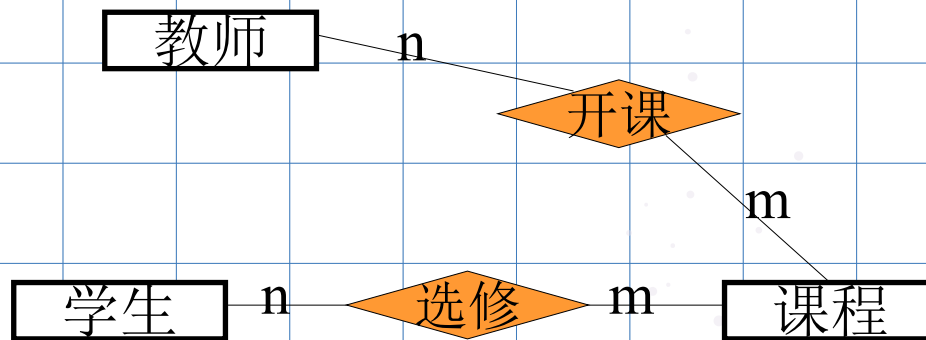


某大学选课管理中，学生可根据自己的情况选修课程。每名学生可同时选修多门课程，每位老师可讲授多门课程，每门课程可由多位老师讲授。

(2) 定义各实体之间的联系：

学生与课程之间是n:m的“选修”联系，“选修”联系包括“分数”属性。

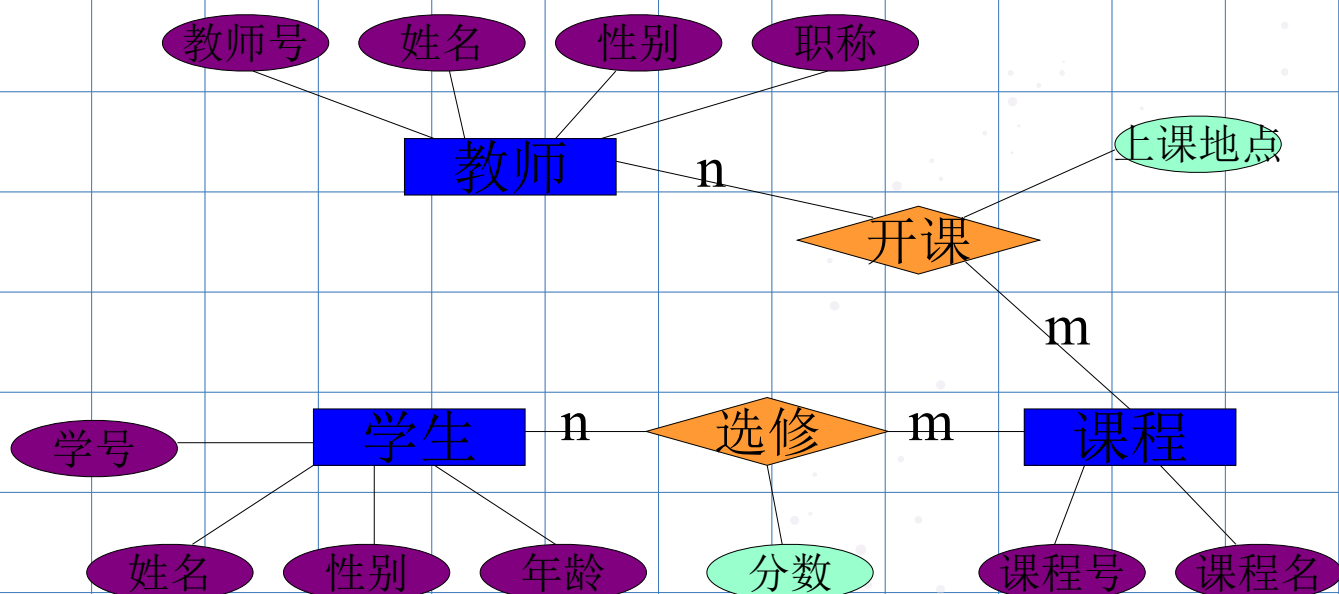
教师与课程之间是n:m的“开课”联系，“开课”联系包括“上课地点”属性。



某大学选课管理的实体联系E-R图



- 例 某大学选课管理中，学生可根据自己的情况选修课程。每名同学可同时选修多门课程，每位老师可讲授多门课程，每门课程可由多位老师讲授。



某大学选课管理总体E-R图



E-R图的表示方法（补充）：

➤对数据对象（实体、属性、联系）的命名

尽可能采用用户熟悉的名字，力求清晰、明了、便于记忆。尽可能反映数据对象的特点。

➤联系的识别与定义

实体间的联系主要包括：

- 存在性联系：系里有学生，系里有老师，产品有顾客。
- 功能性联系：教师讲授课程，学生选修课程，教师参加项目，机床加工零件。

定义联系就是识别联系，并确定联系的名字，并根据语义分析，确定联系的类型（一对一、一对多、多对多），若联系具有某种属性，定义属性名字。



概念结构设计—关键技术

- 实际上实体与属性是相对而言的，很难有截然划分的界限。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”。一般说来，在给定的应用环境中：

两条准则：

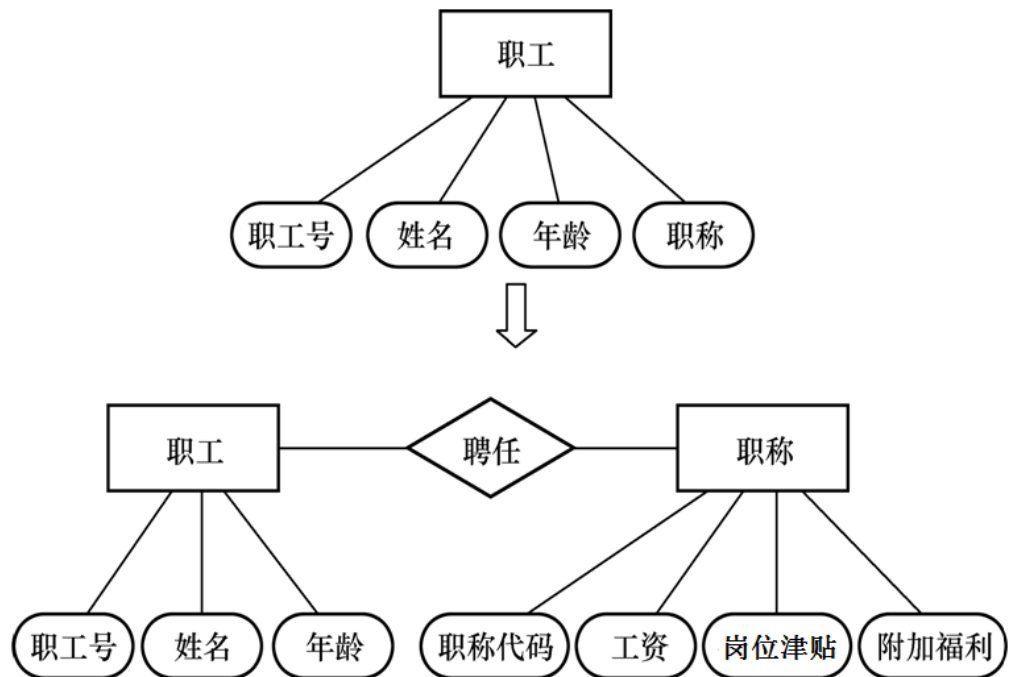
- (1) 属性不能再具有需要描述的性质。即属性必须是不可分的数据项，不能再由另一些属性组成
- (2) 属性不能与其他实体具有联系。联系只发生在实体之间



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。

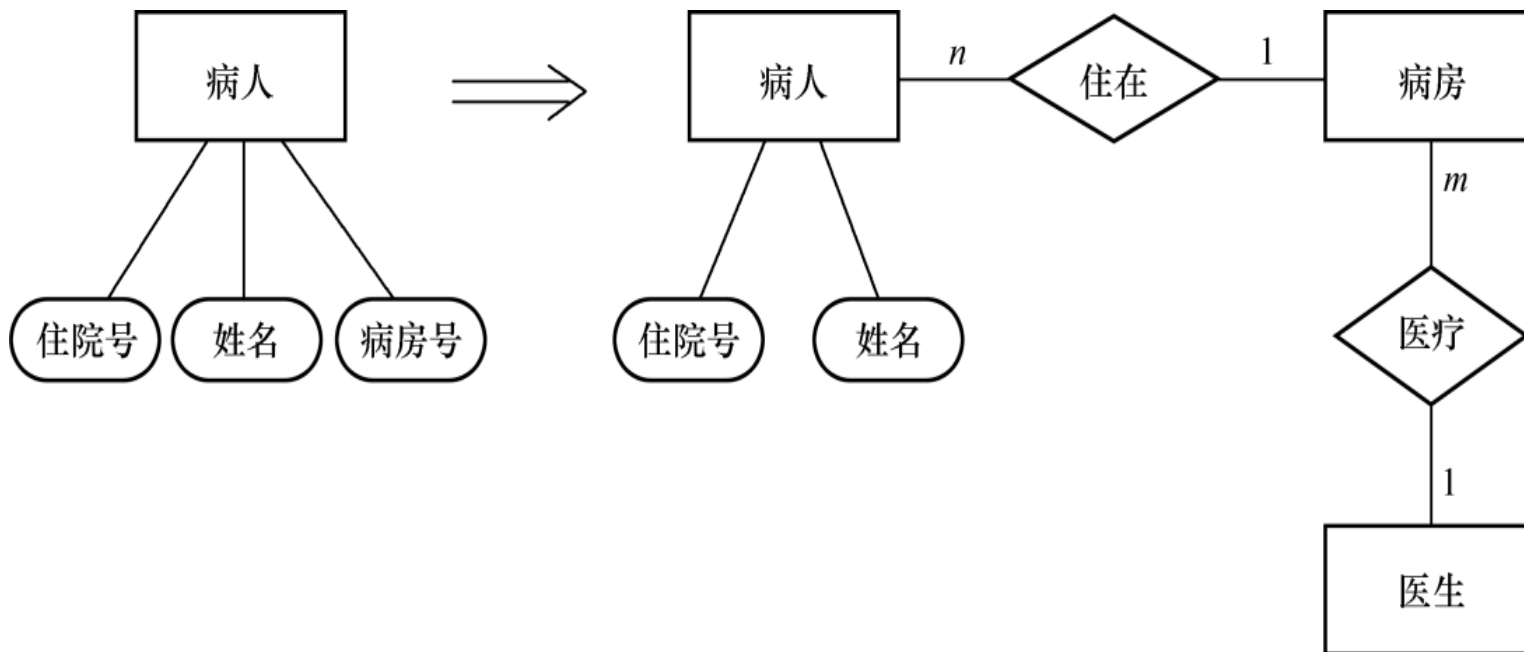
- 职称如果没有与工资、福利挂钩，根据准则（1）可以作为职工实体的属性
- 如果不同的职称有不同的工资、住房标准和不同的附加福利，则职称作为一个实体更恰当



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；

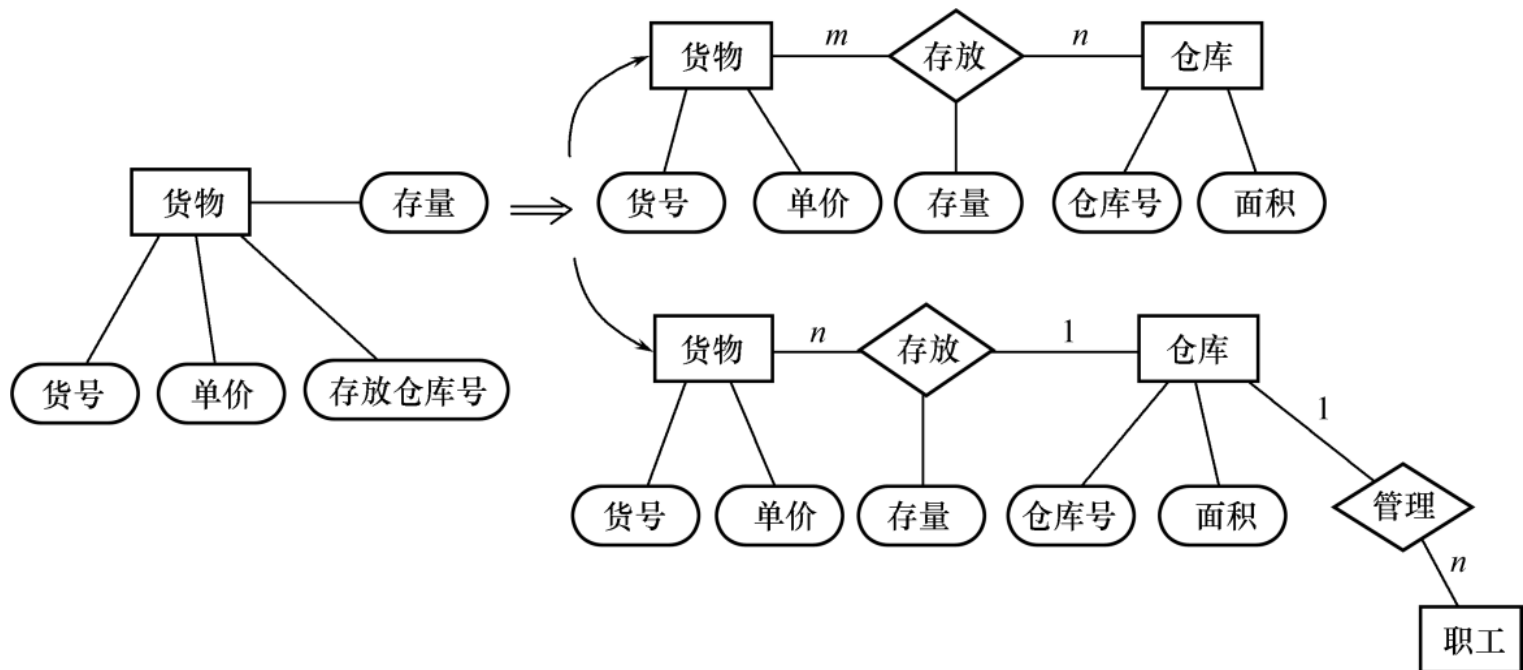
如果病房还要与医生实体发生联系，即一个医生负责几个病房的病人的医疗工作，则根据准则（2）病房应作为一个实体。



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。

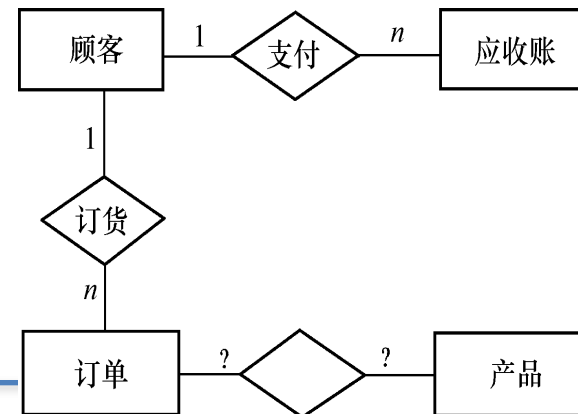


概念结构设计（续）

- [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



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（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。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。

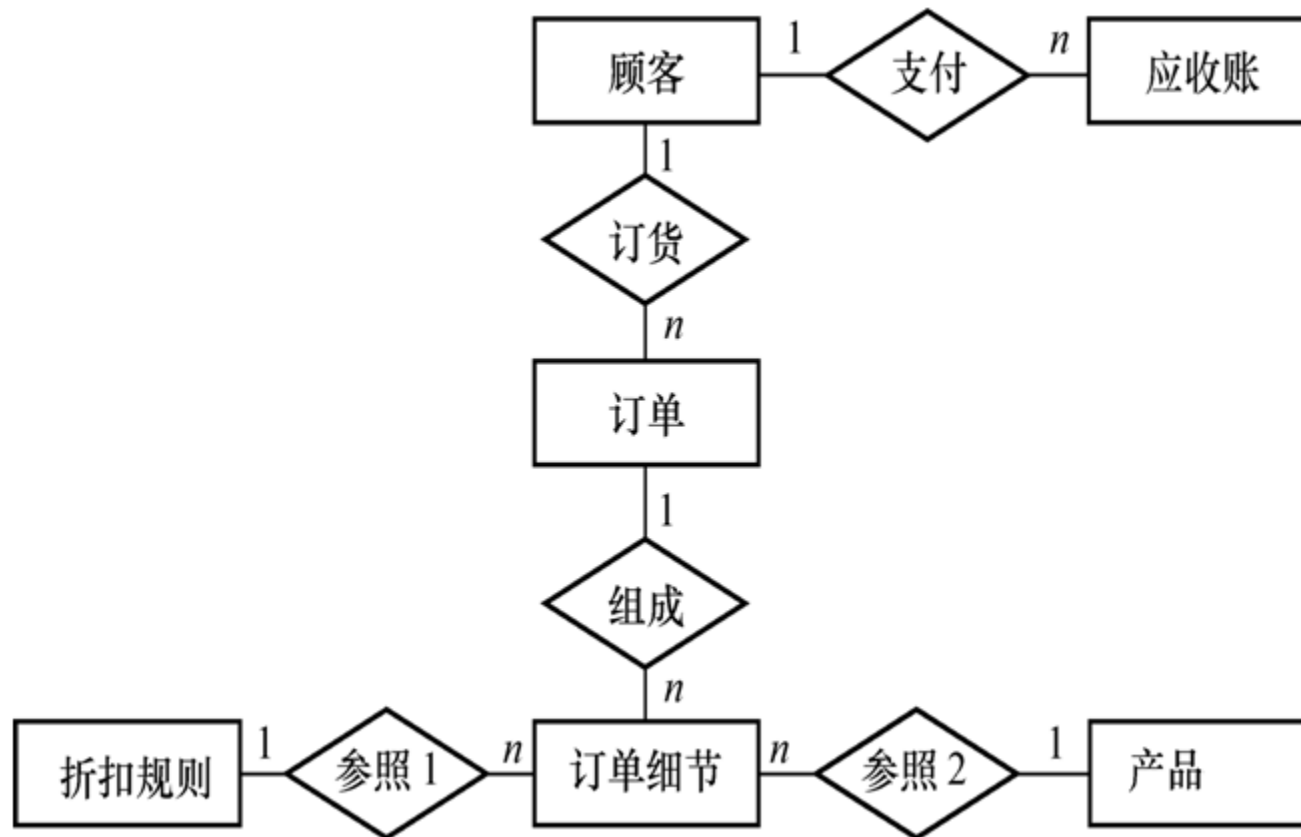


图7.23 销售管理子系统的E-R图



概念结构设计（续）

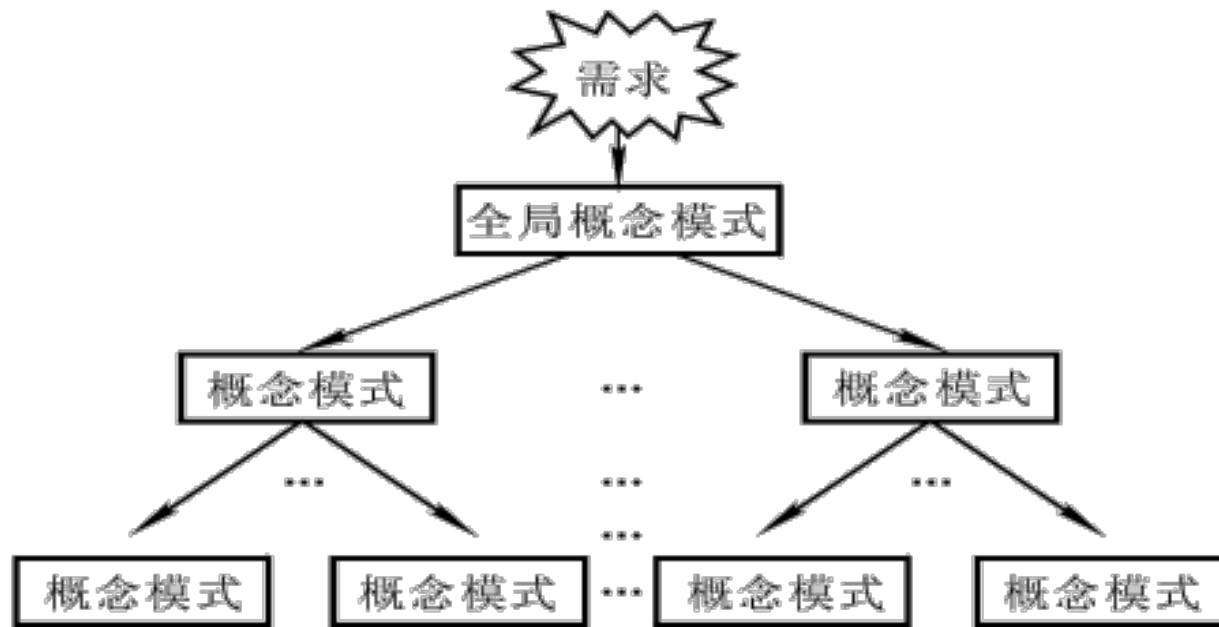
- 对每个实体定义的属性如下：
 - 顾客：{顾客号，顾客名，地址，电话，信贷状况，账目余额}
 - 订单：{订单号，顾客号，订货项数，订货日期，交货日期，工种号，生产地点}
 - 订单细则：{订单号，细则号，零件号，订货数，金额}
 - 应收账款：{顾客号，订单号，发票号，应收金额，支付日期，支付金额，当前余额，货款限额}
 - 产品：{产品号，产品名，单价，重量}
 - 折扣规则：{产品号，订货量，折扣}



概念结构设计-方法

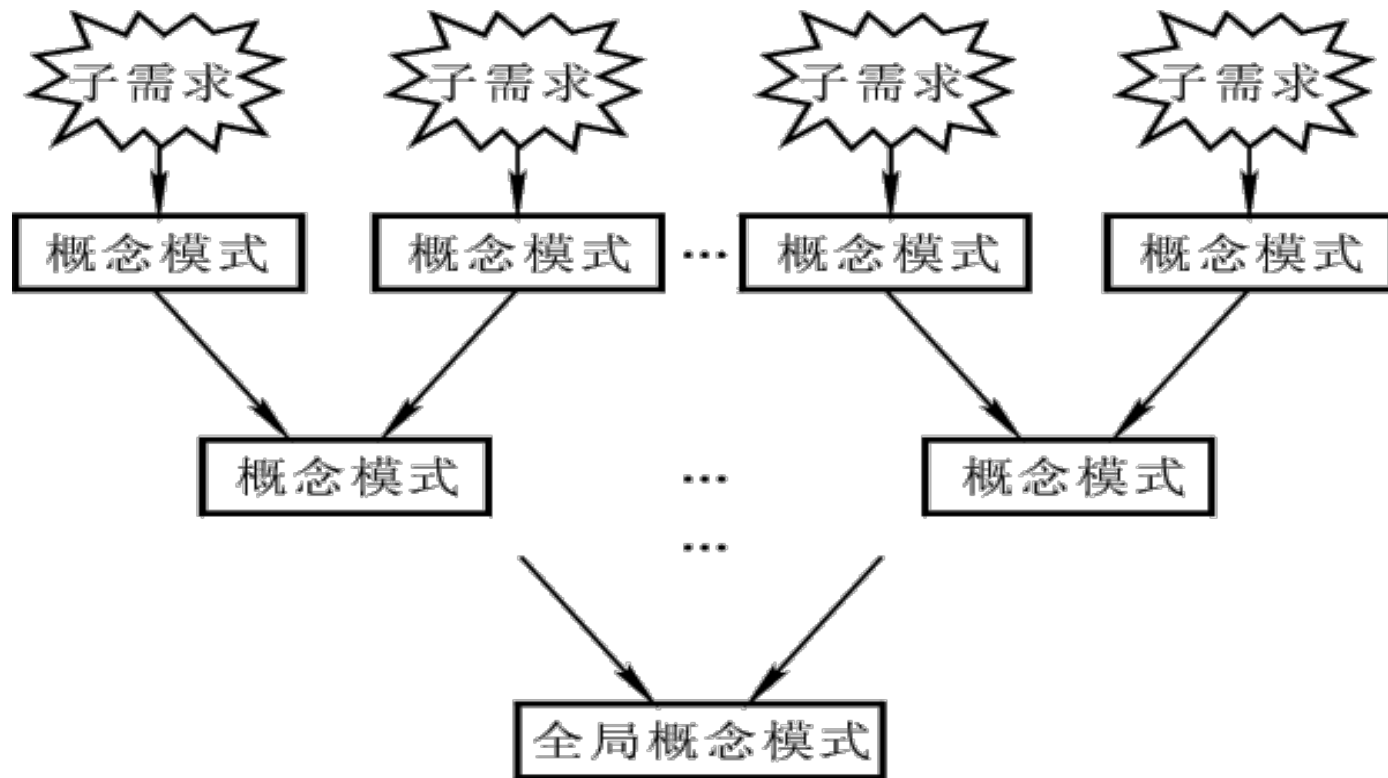
- 设计概念结构的四类方法
 - 自顶向下

➤ 首先定义全局概念结构的框架，然后逐步细化



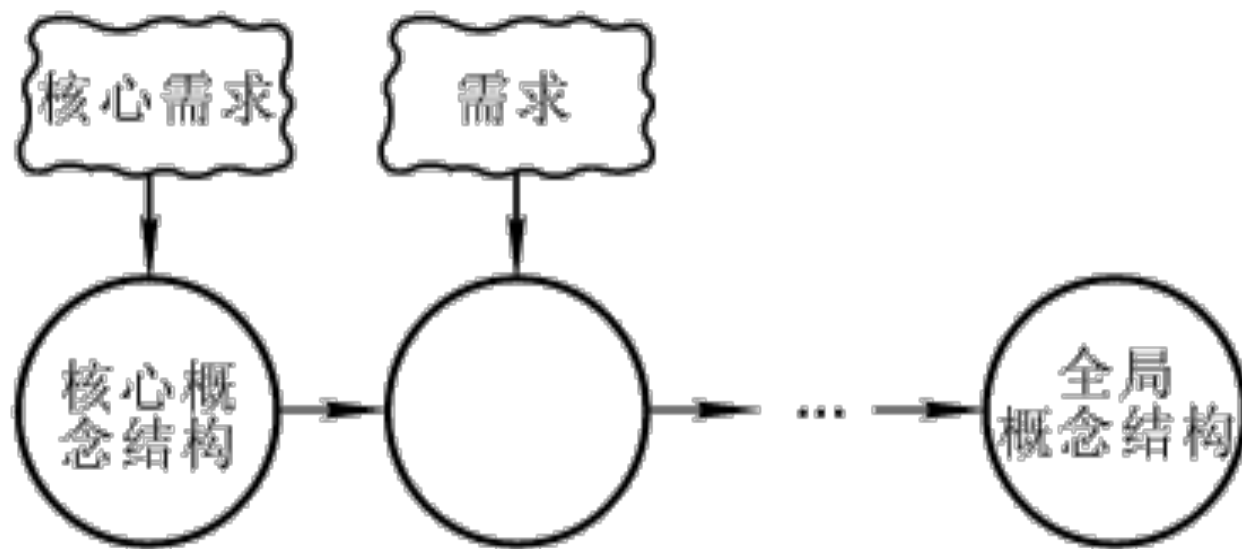
—自底向上

- 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



—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



一 混合策略

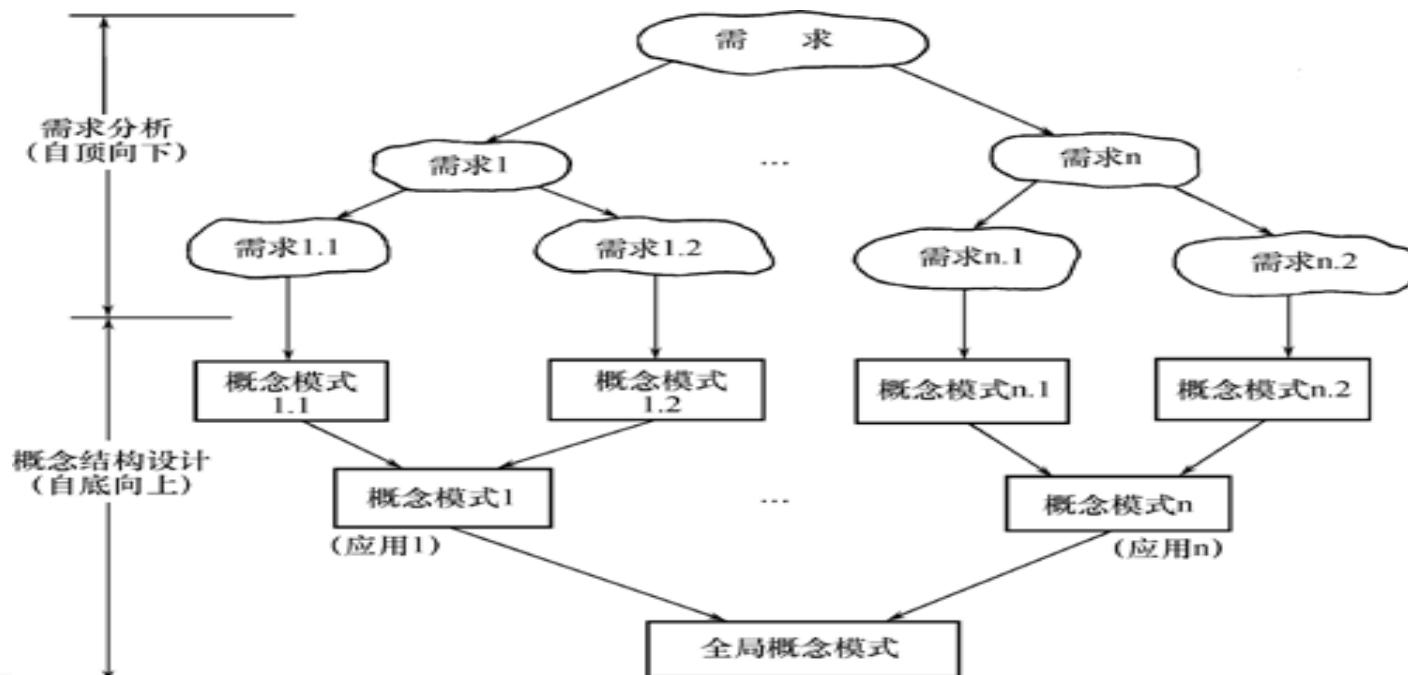
- 将自顶向下和自底向上相结合，用自顶向下策略设计一个全局概念结构的框架，以它为骨架集成由自底向上策略中设计的各局部概念结构。



概念结构设计-方法

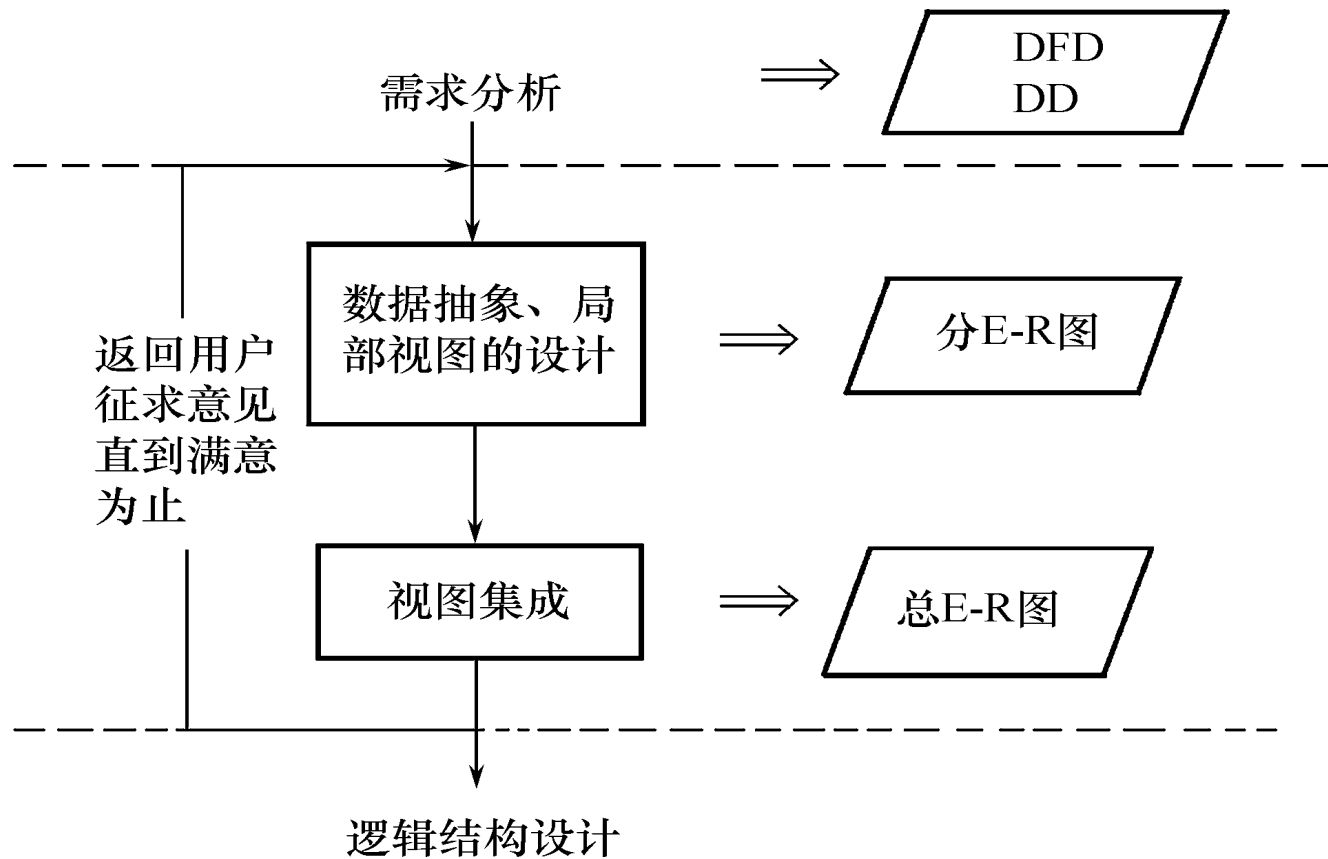
- 常用策略

- 自顶向下地进行需求分析
- 自底向上地设计概念结构



❖ 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

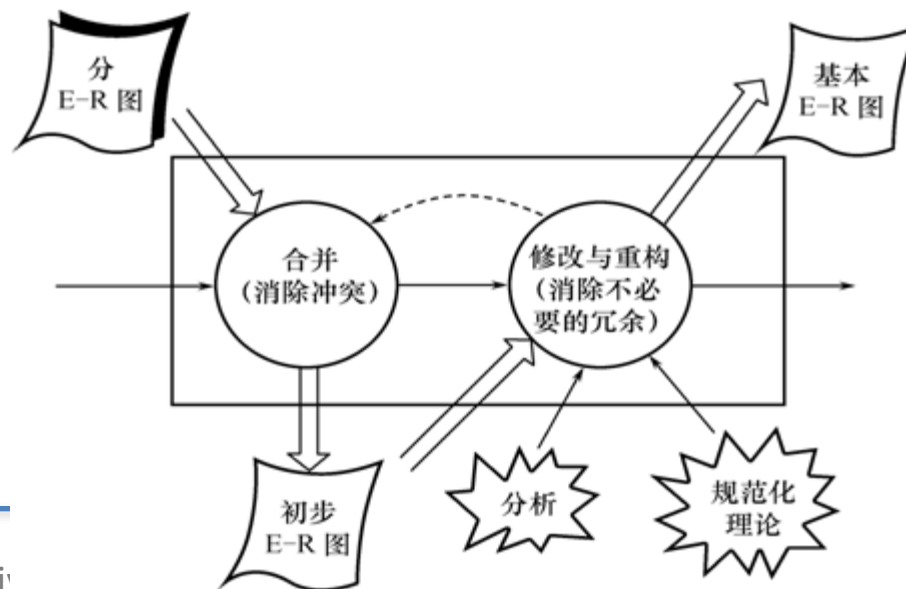


概念结构设计（续）

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
 - 通过讨论、协商等行政手段加以解决



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图中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。



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。



图7.25(a)中零件与产品之间存在多对多的联系“构成”

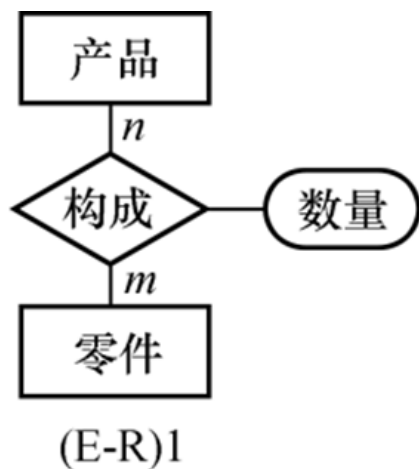
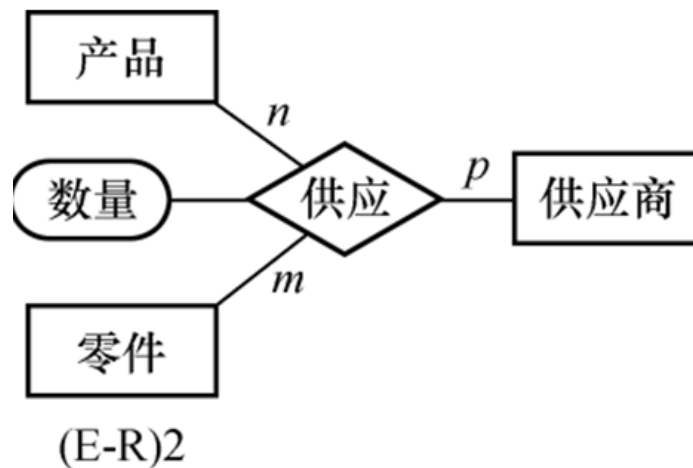
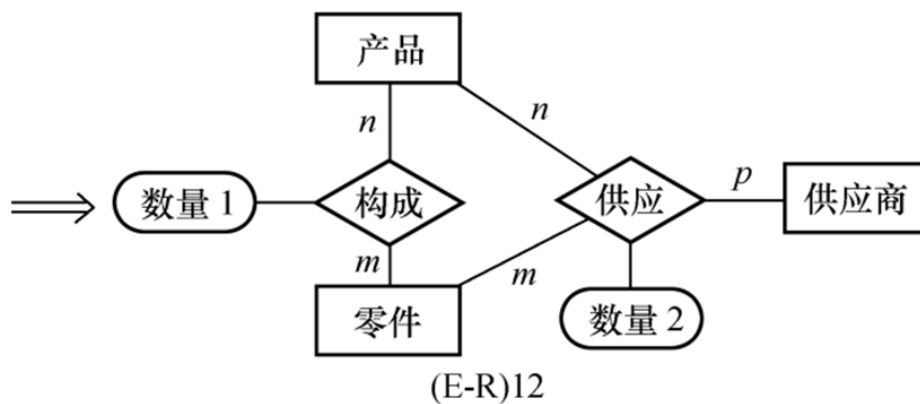


图7.25(b)中产品、零件与供应商三者之间还存在多对多的联系“供应”



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。

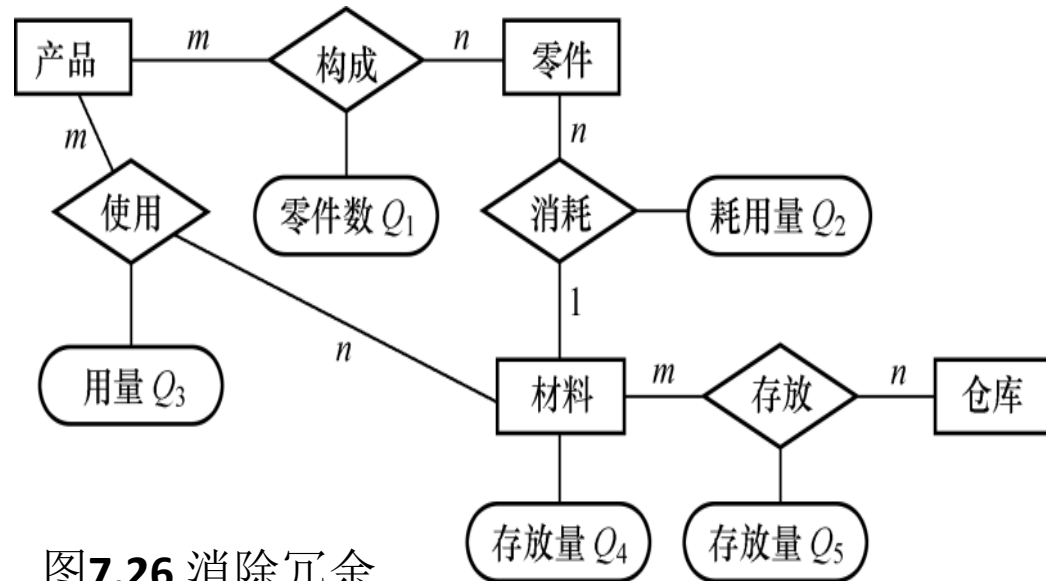


图7.26 消除冗余

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）

（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）

- 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。



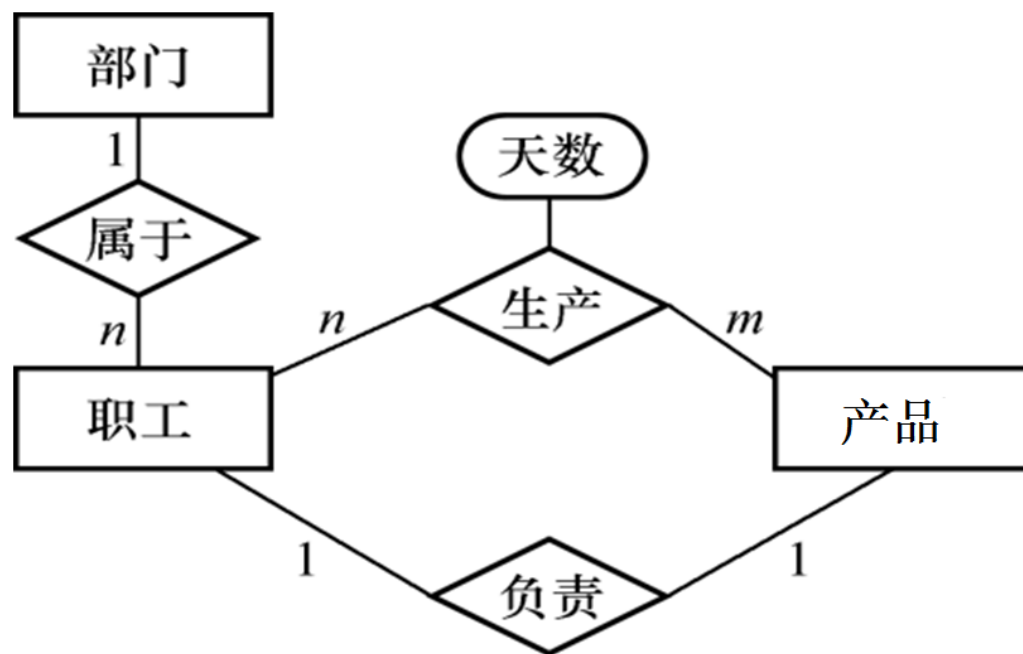


图7.27 劳动人事管理的分E-R图

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。



- [例7.2] 某工厂管理信息系统的视图集成。

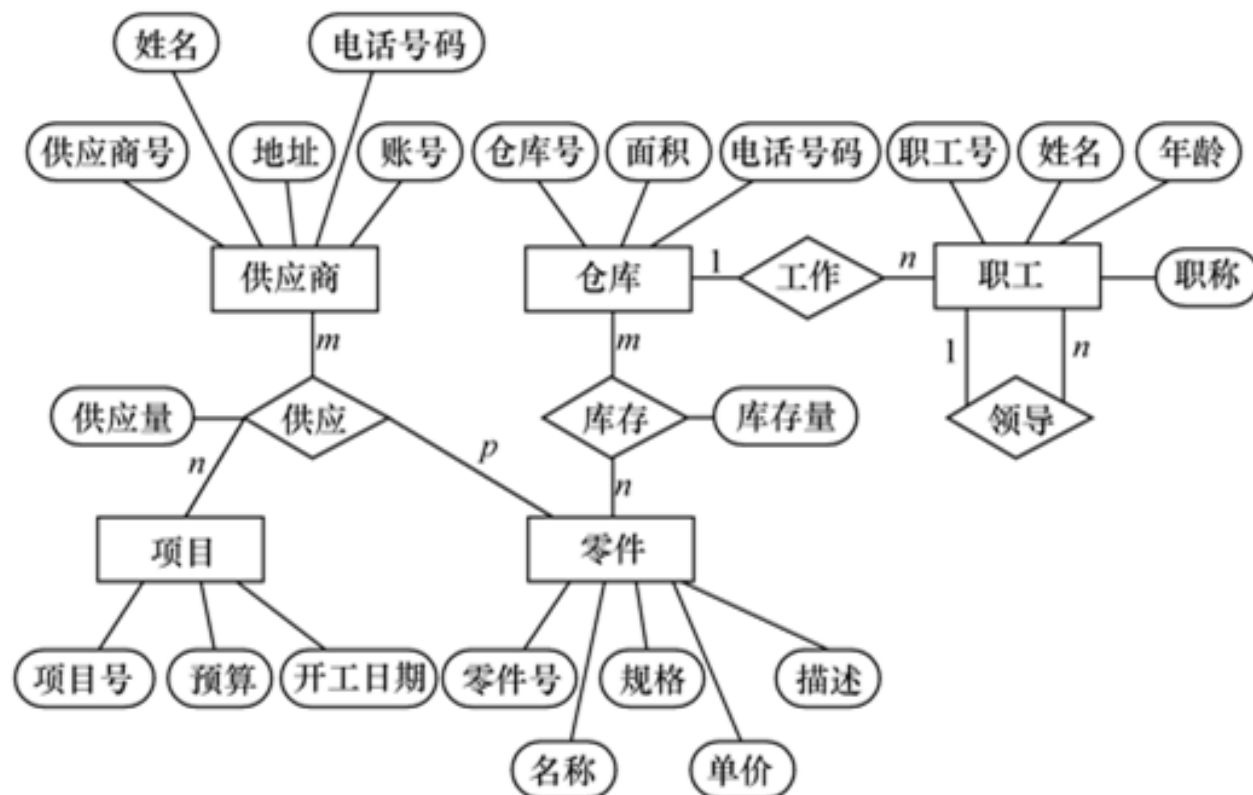


图7.11 工厂物资管理E-R图



- [例7.2] 某工厂管理信息系统的视图集成。

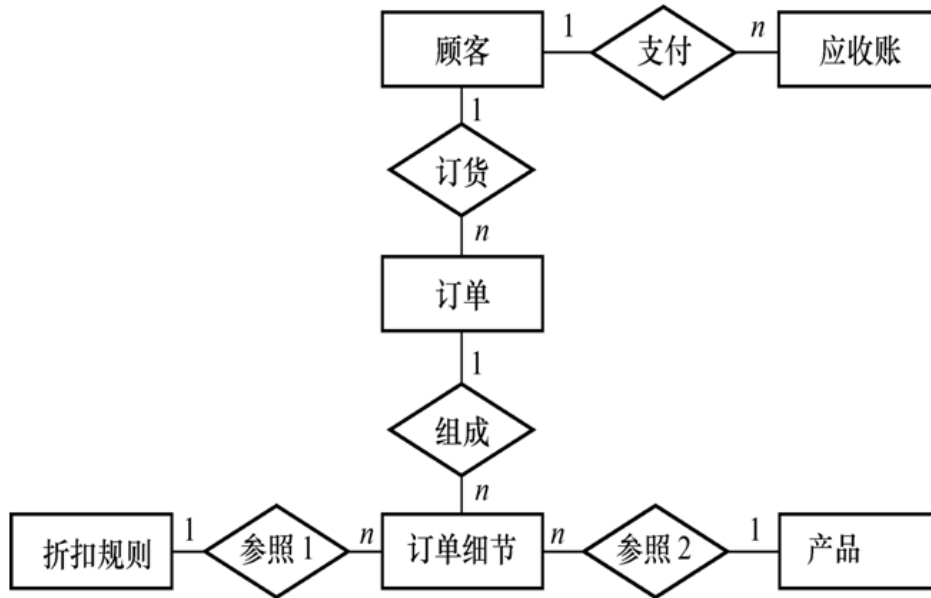


图7.23 销售管理子系统的E-R图

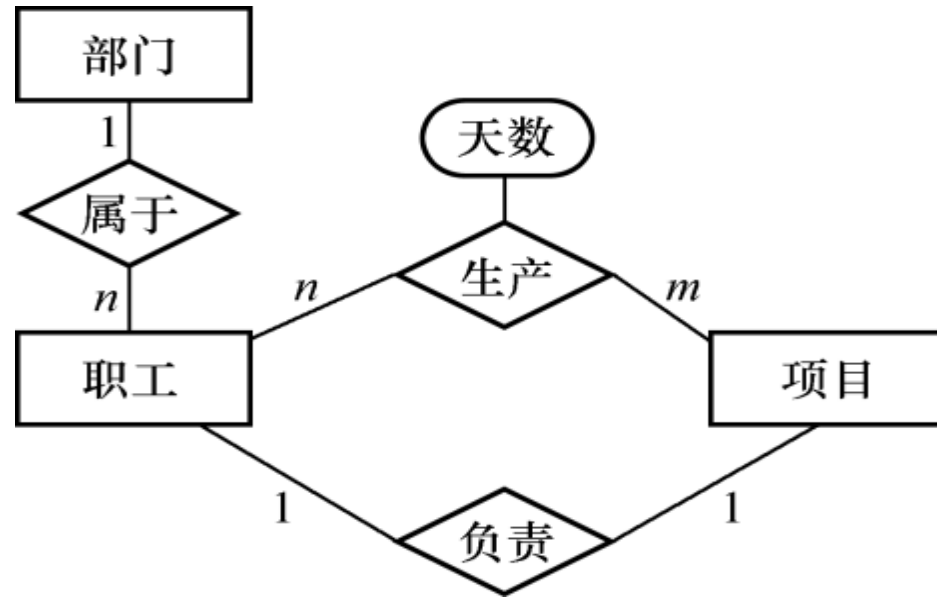
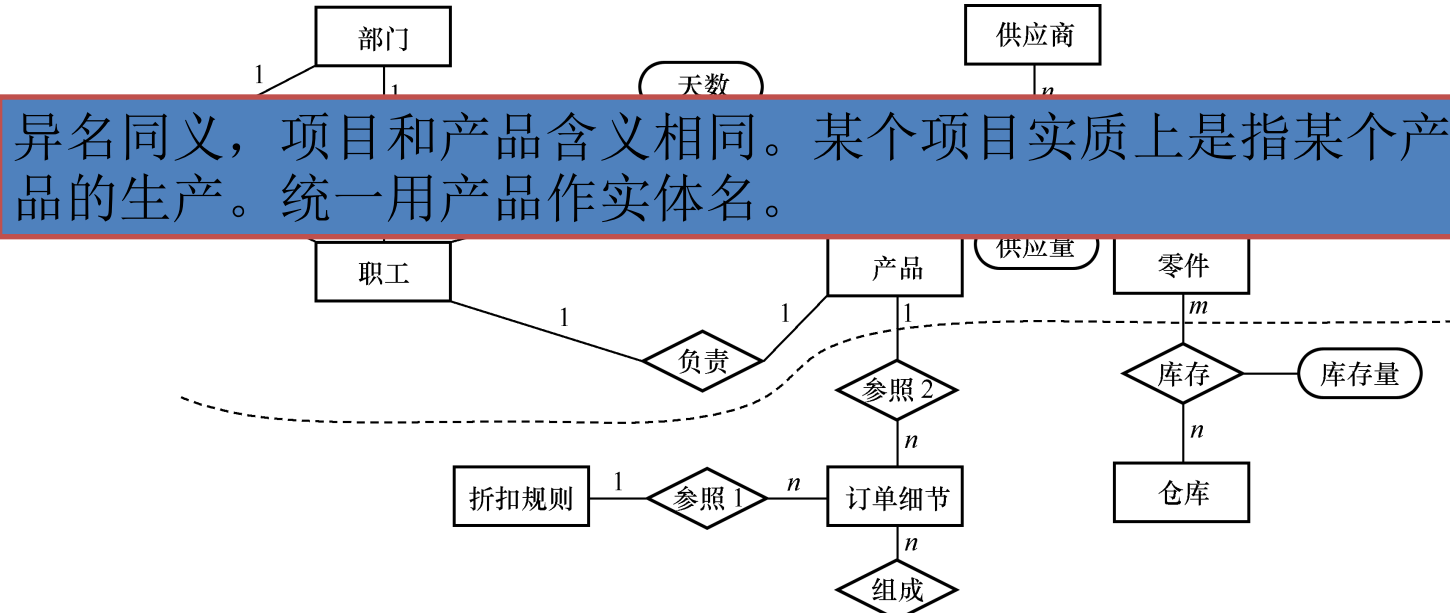


图7.27 劳动人事管理的分E-R图



概念结构设计（续）

- [例7.2] 某工厂管理信息系统的视图集成。



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

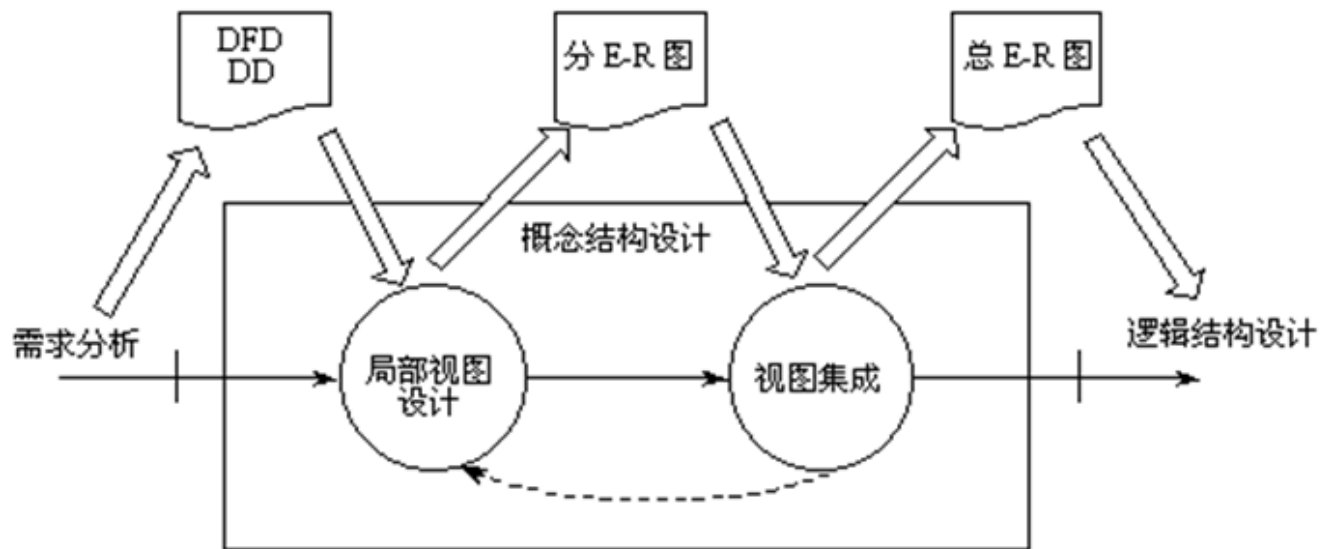
概念结构设计-验证整体概念结构

- 局部结构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局部结构，包括属性、实体及实体间的联系
 - 整体概念结构能满足需求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。



概念结构设计-小结

- 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



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

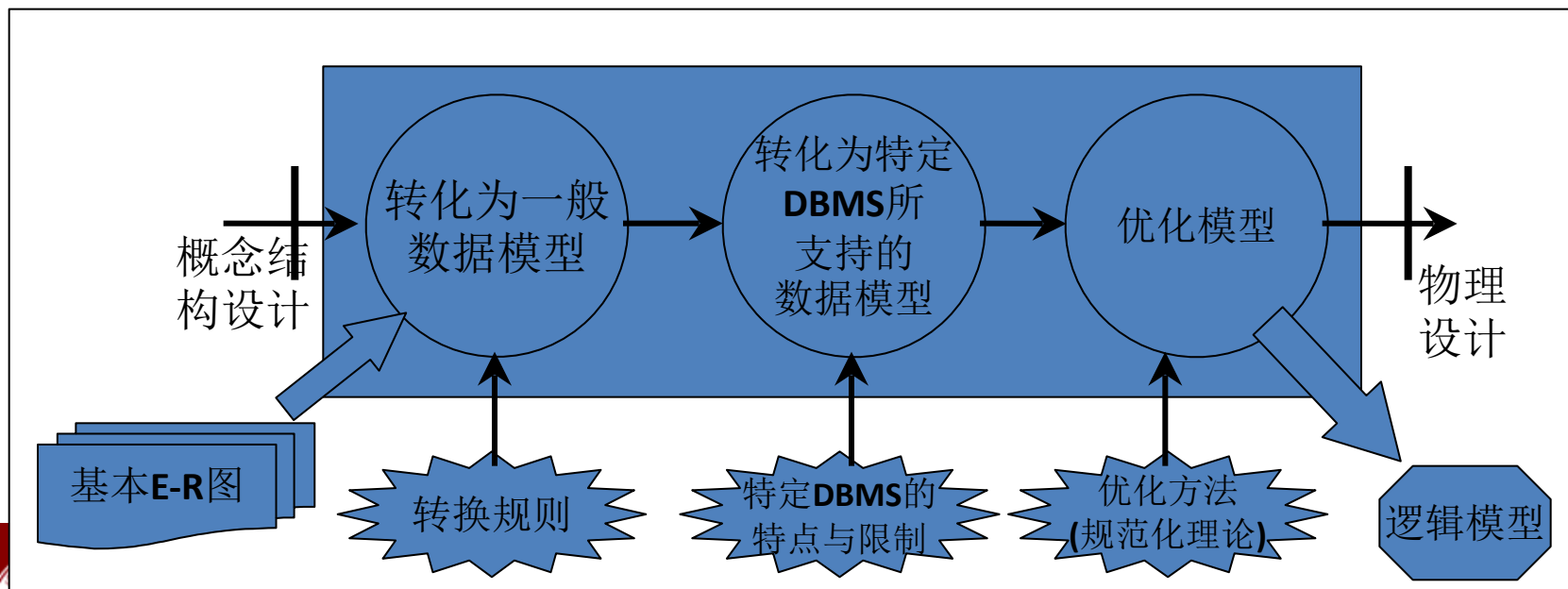


逻辑结构设计

- 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

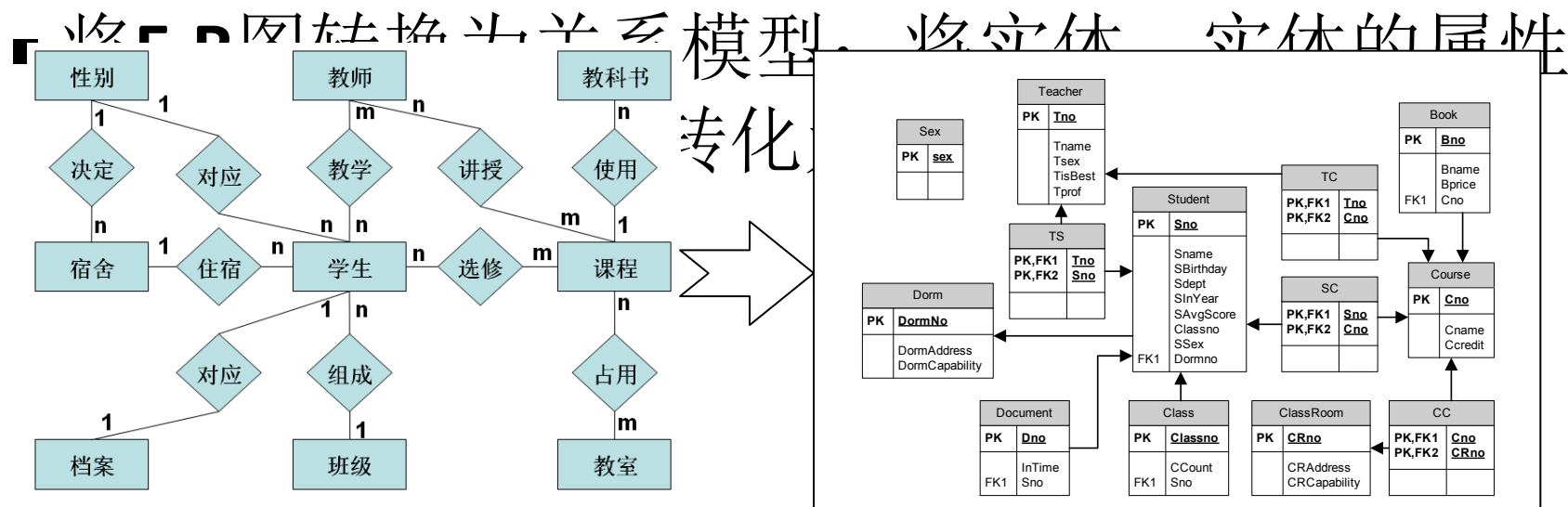
- 逻辑结构设计的步骤



逻辑结构设计-向关系模型的转换

❖ 转换内容

- **E-R图**由实体、实体的属性和实体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



转换规则：

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

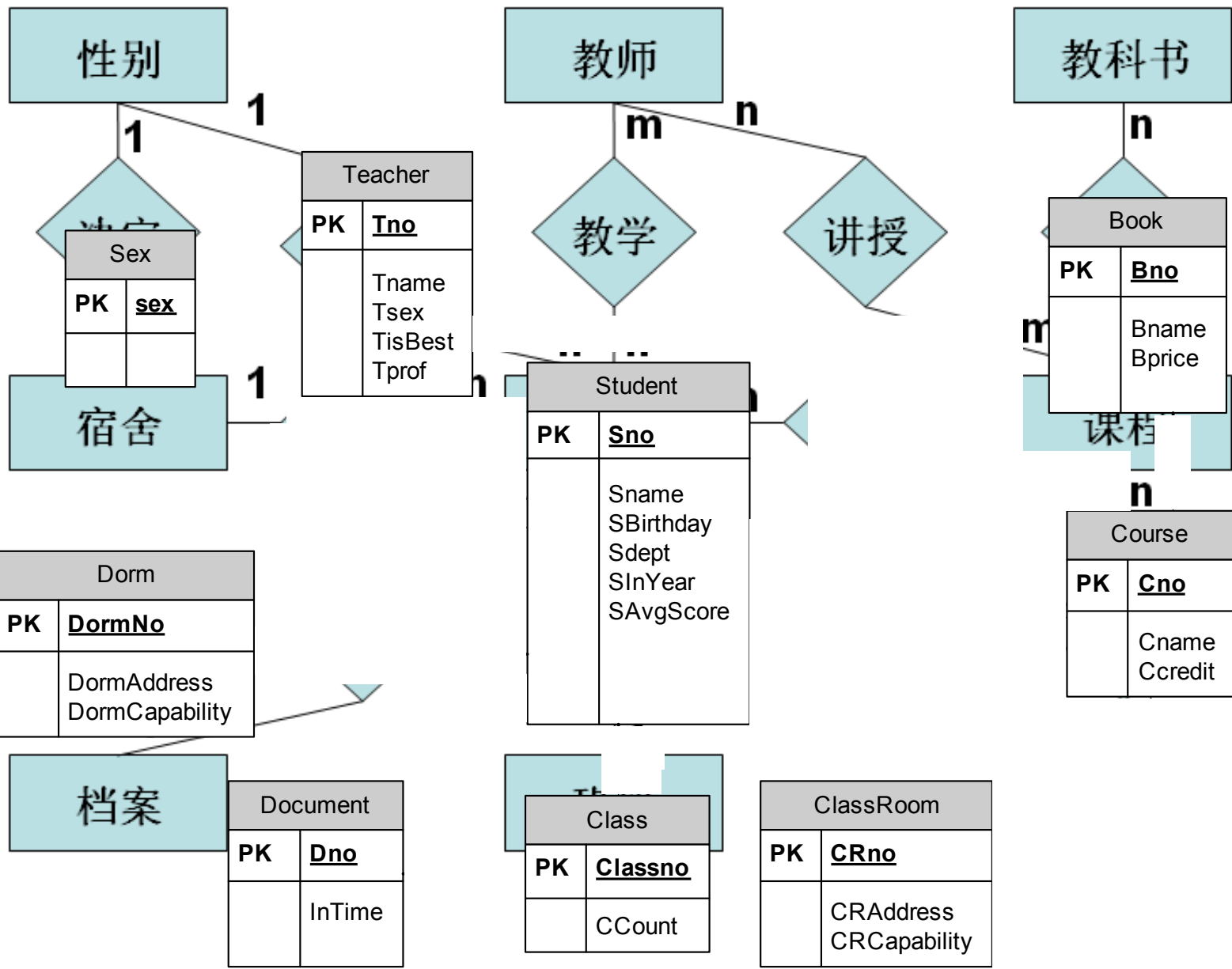
例，学生实体可以转换为如下关系模式：

学生（学号，姓名，出生日期，所在系，年级，平均成绩）

性别、宿舍、班级、档案材料、教师、课程、教室、教科书

都分别转换为一个关系模式。





2. 实体型间的联系有以下不同情况

(1) 一个 $m:n$ 联系转换为一个关系模式

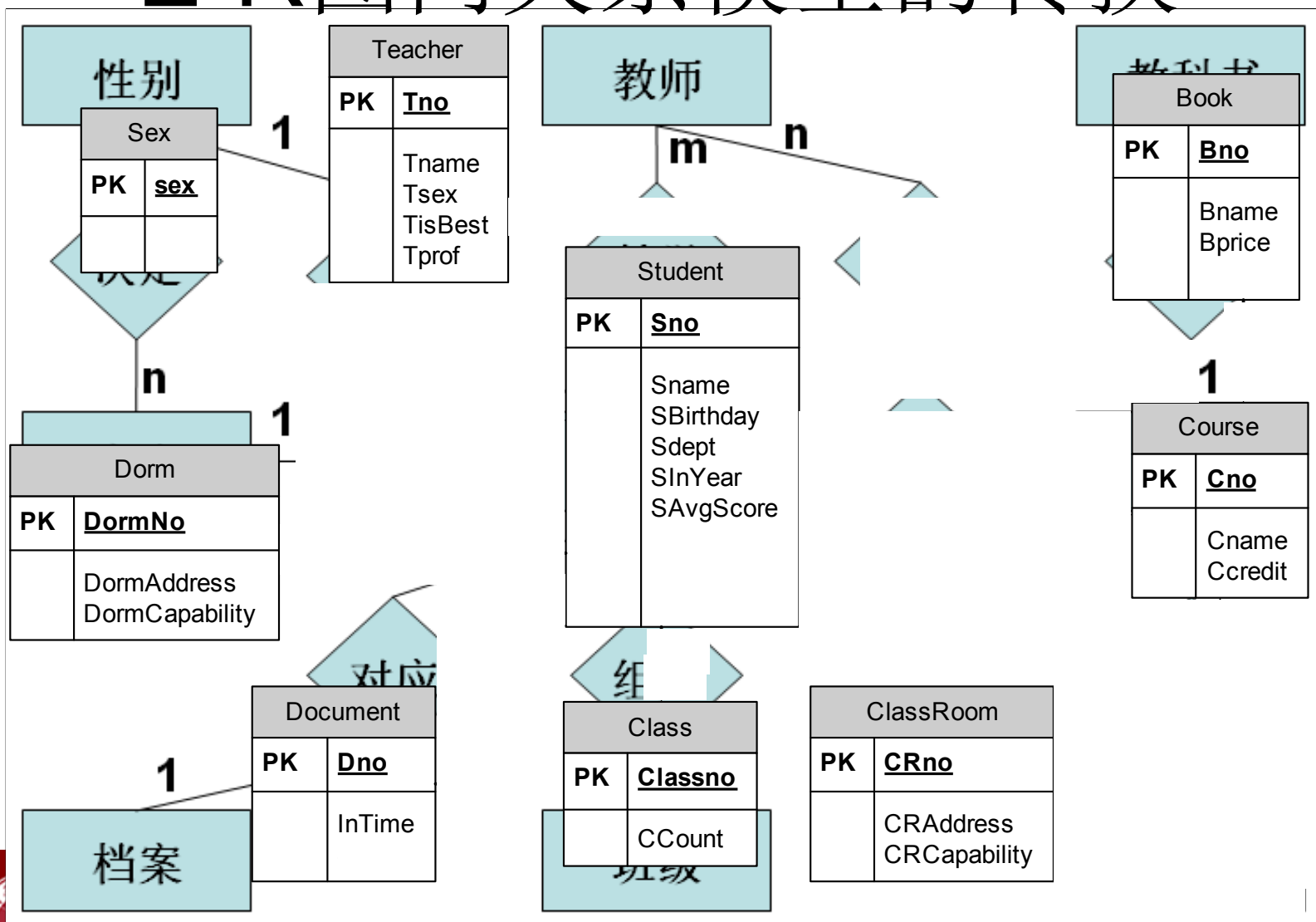
- 关系的属性：与该联系相连的各实体的码以及联系本身的属性
- 关系的码：各实体码的组合

[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）



E-R图向关系模型的转换



(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

- 例，“组成”联系为1:n联系。将其转换为关系模式的两种方法：
 - 使其成为一个独立的关系模式：
组成（学号，班级号）



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

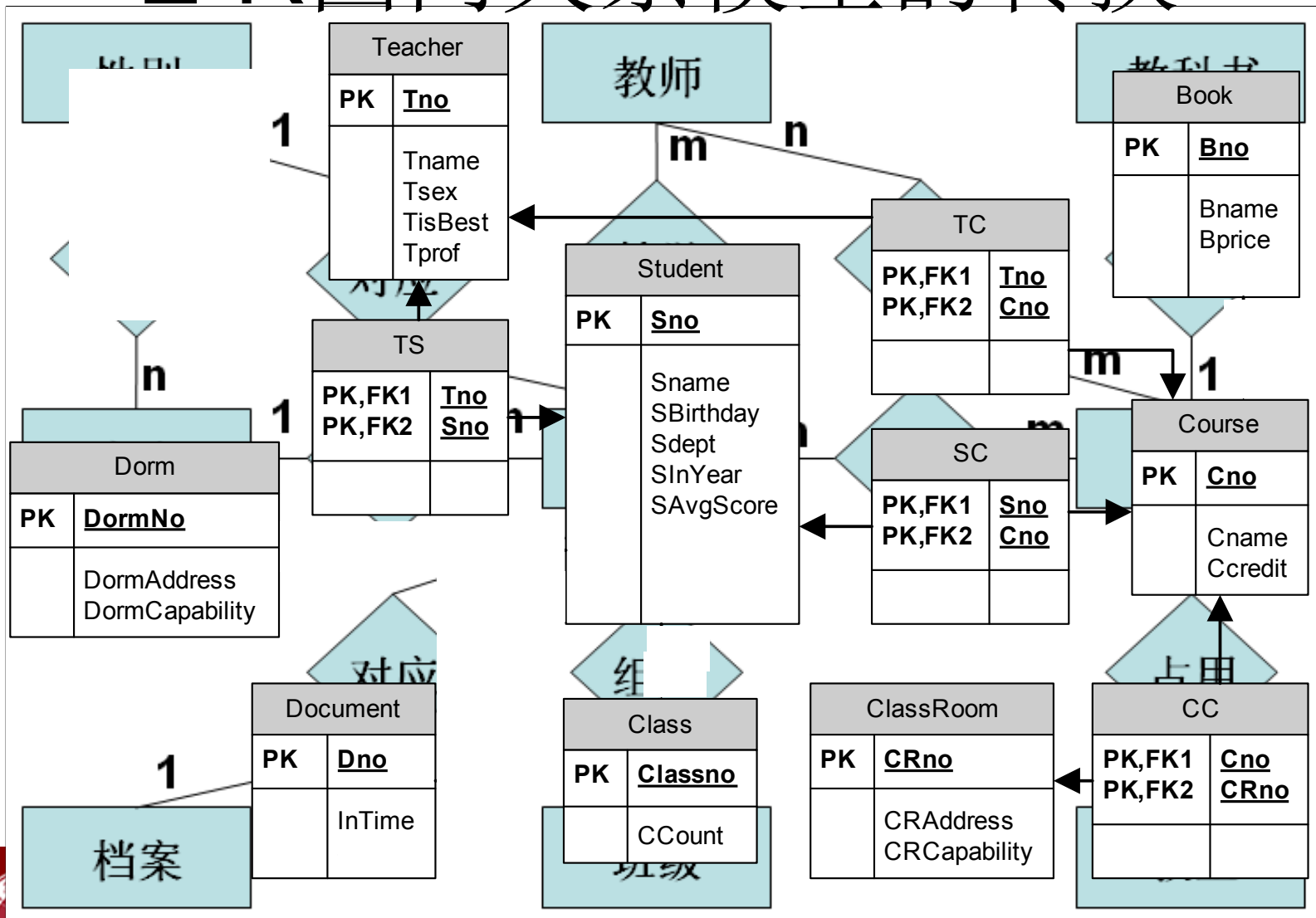


E-R图向关系模型的转换

- 例，“组成”联系为1:n联系。将其转换为关系模式的两种方法：
 - 使其成为一个独立的关系模式：
组成（学号，班级号）
 - 将其学生关系模式合并：
学生（学号，姓名，出生日期，所在系，年级，
班级号，平均成绩）



E-R图向关系模型的转换



(3) 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

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



E-R图向关系模型的转换

- 例，“管理”联系为1:1联系，可以有三种转换方法：
 - 转换为一个独立的关系模式：
 管理 (职工号, 班级号)
或管理 (职工号, 班级号)
 - “管理”联系与班级关系模式合并，则只需在班级关系中加入教师关系的码，即职工号：
 班级： (班级号, 学生人数, 职工号)
 - “管理”联系与教师关系模式合并，则只需在教师关系中加入班级关系的码，即班级号：
 教师： (职工号, 姓名, 性别, 职称, 班级号, 是否为优秀班主任)

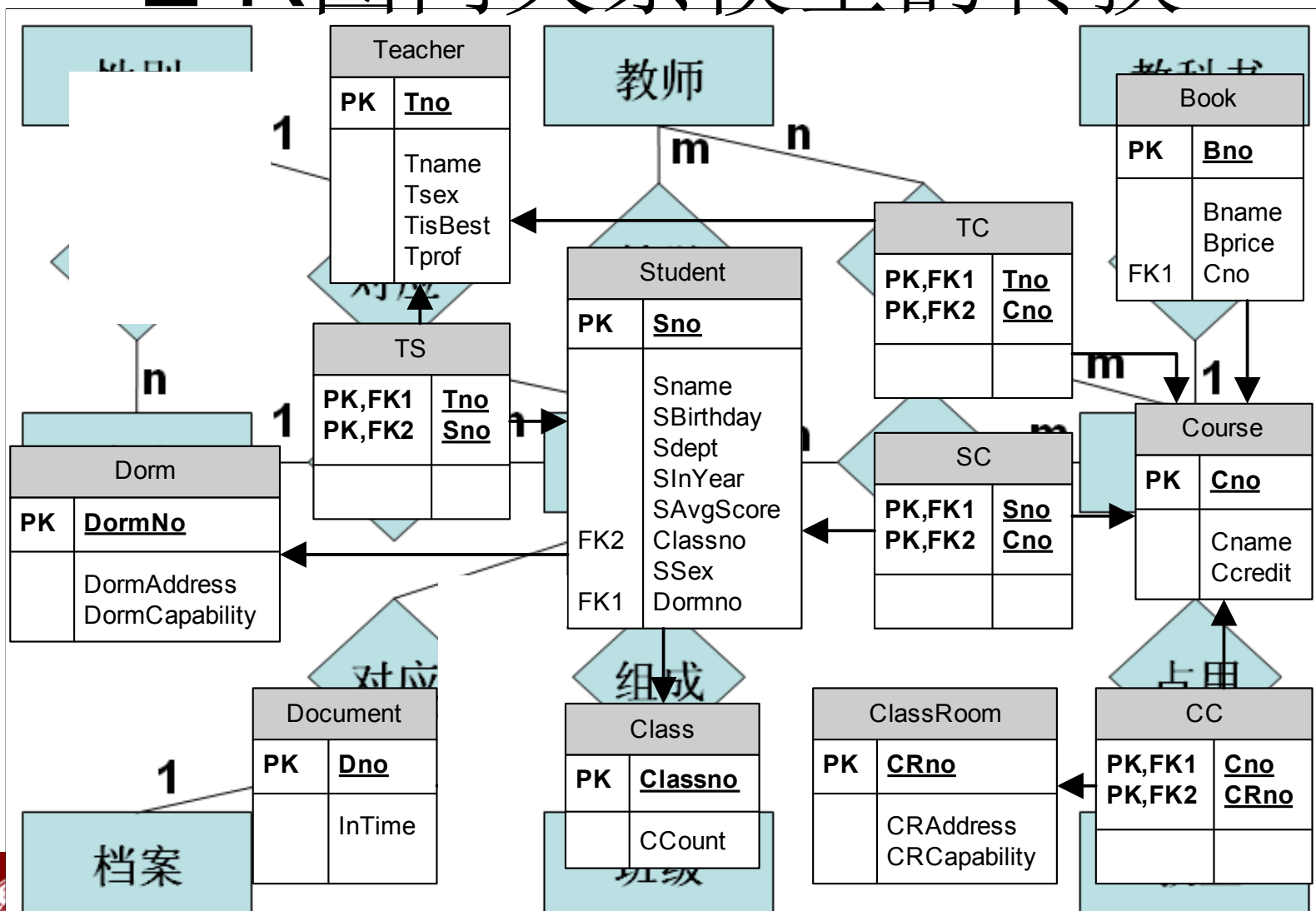


E-R图向关系模型的转换

- 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。



E-R图向关系模型的转换



(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)



(5) 同一实体集的实体间的联系，即自联系，也可按上述1:1、1:n和m:n三种情况分别处理。

•例，如果教师实体集内部存在领导与被领导的1:n自联系，我们可以将该联系与教师实体合并，这时主码职工号将多次出现，但作用不同，可用不同的属性名加以区分：

教师：{职工号，姓名，性别，职称，系主任}



(6) 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



逻辑结构设计-优化

- 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。



逻辑结构设计-规范化和性能的关系

- 为满足某种商业目标，数据库性能比规范化数据库更重要
 - 通过在给定的表中添加额外的字段，以大量减少需要从中搜索信息所需的时间
 - 在给定的表中插入计算列（如成绩总分），以方便查询
- 进行规范化的同时，还需要综合考虑数据库的性能。



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

显然有： 学号→(英语,数学,语文)

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，我们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）

（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）

— 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）



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。

- ✓ 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。



逻辑结构设计-设计用户子模式

- 根据局部应用需求设计用户子模式
- 设计用户子模式的目的
 - 使用更符合用户习惯的别名
 - 对不同级别的用户定义不同的视图，保证系统安全性
 - 简化用户对系统的使用
- DBMS中一般采用视图(View)机制



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



数据库物理设计

- 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



数据库物理设计-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**DBMS**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-选择参数

- 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

- 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



数据库物理设计-关系模式存取方法

- 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。
- 存取方法是快速存取数据库中数据的技术。DBMS一般提供多种存取方法。



数据库物理设计-关系模式存取方法

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



数据库物理设计-索引

- 索引存取方法的选择
- 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



数据库物理设计-索引

- 索引存取方法的选择

- 选择B+树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



数据库物理设计-索引

- 索引存取方法的选择
- 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



数据库物理设计-HASH存取

- 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中
 - 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



数据库物理设计-聚簇

- 聚簇存取方法的选择
- 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。



数据库物理设计-聚簇

- 聚簇的用途

- 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。

- 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。



数据库物理设计-聚簇

- 聚簇的用途
- 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



数据库物理设计-聚簇

- 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。



数据库物理设计-聚簇

- 聚簇的适用范围
- 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，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



数据库物理设计-聚簇

- 选择聚簇存取的方法

- 1.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组合聚簇；
- 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；
- 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。



数据库物理设计-聚簇

- 选择聚簇存取的方法
- 2.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；
 - 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。



数据库物理设计-聚簇

- 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，聚簇码值要相对稳定，以减少修改聚簇码值所引起的维护开销。



数据库物理设计-确定物理存储结构

- 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



数据库物理设计-物理存储位置

- 影响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。



数据库物理设计-物理存储位置

- 基本原则

- 根据应用情况将

- 易变部分与稳定部分
 - 存取频率较高部分与存取频率较低部分
 - 分开存放，以提高系统性能



数据库物理设计-物理存储位置

- 例如：

- 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，可以考虑将表和索引分别放在不同的磁盘上，在查询时，由于两个磁盘驱动器分别在工作，因而可以保证物理读写速度比较快。
- 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。



数据库物理设计-系统配置

- **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
 -



数据库物理设计-评价物理结构

- 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



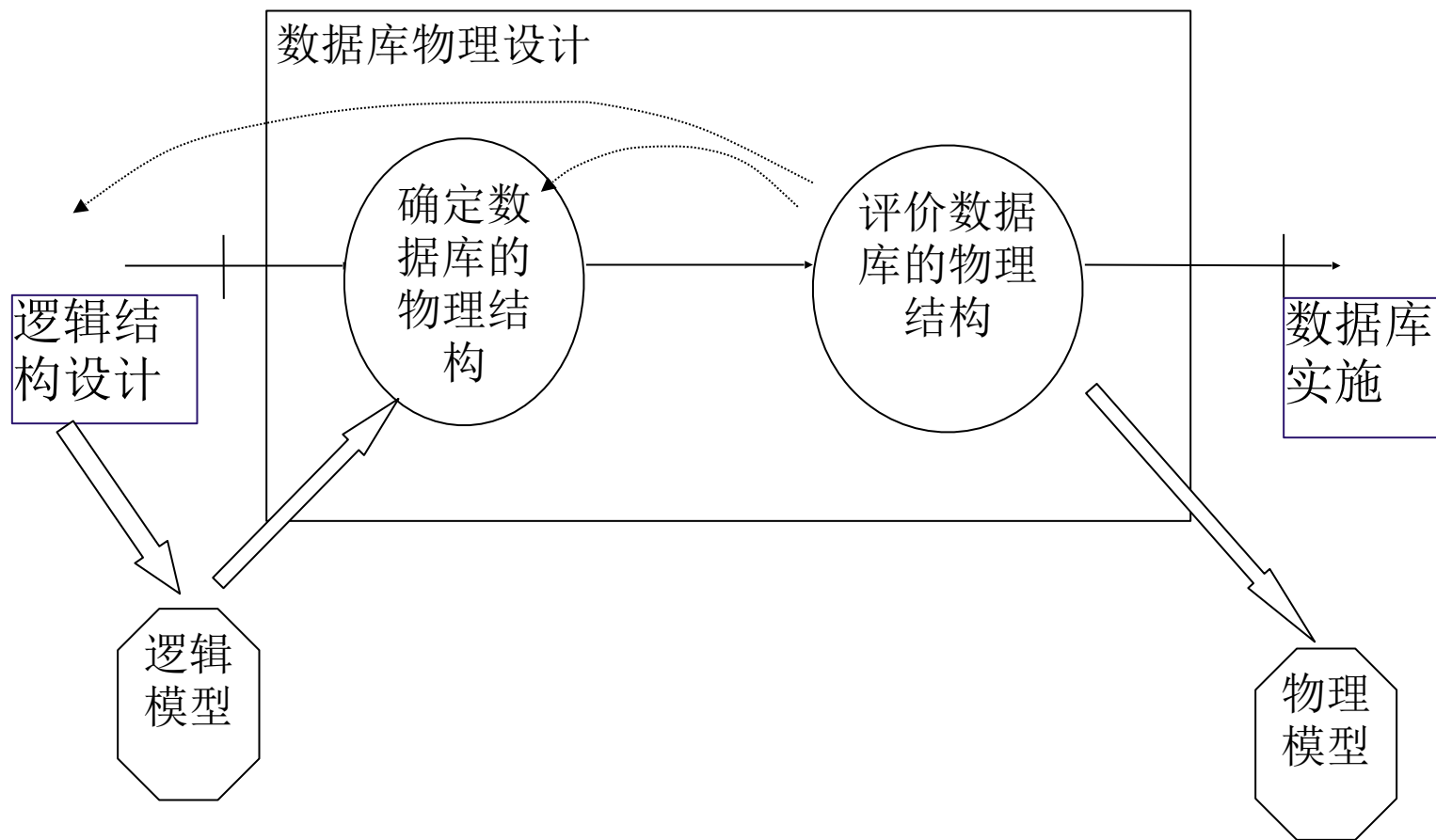
数据库物理设计-评价物理结构

- 评价方法（依赖于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



数据库物理设计-步骤



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



数据库的实施与维护-数据装入

- 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



数据库的实施与维护-应用程序的 编码和调试

- 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



数据库的实施与维护-试运行

- 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，直到达到设计要求阶段



数据库的实施与维护-试运行

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



数据库的实施与维护-试运行

- 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太，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



数据库的实施与维护-试运行

- 强调两点：
- 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。



数据库的实施与维护-运行与维护

- 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



数据库的实施与维护-运行与维护

- 在数据库运行阶段，对数据库经常性的维护工作主要是由**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. 数据库的重组和重构



数据库的实施与维护-重组织和重构造

- 重组织的工作
 - 按原设计要求
 - 重新安排存储位置
 - 回收垃圾
 - 减少指针链
 - 数据库的重组织不会改变原设计的数据逻辑结构和物理结构



数据库的实施与维护-重组织和重构造

◎ 数据库重构造

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

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

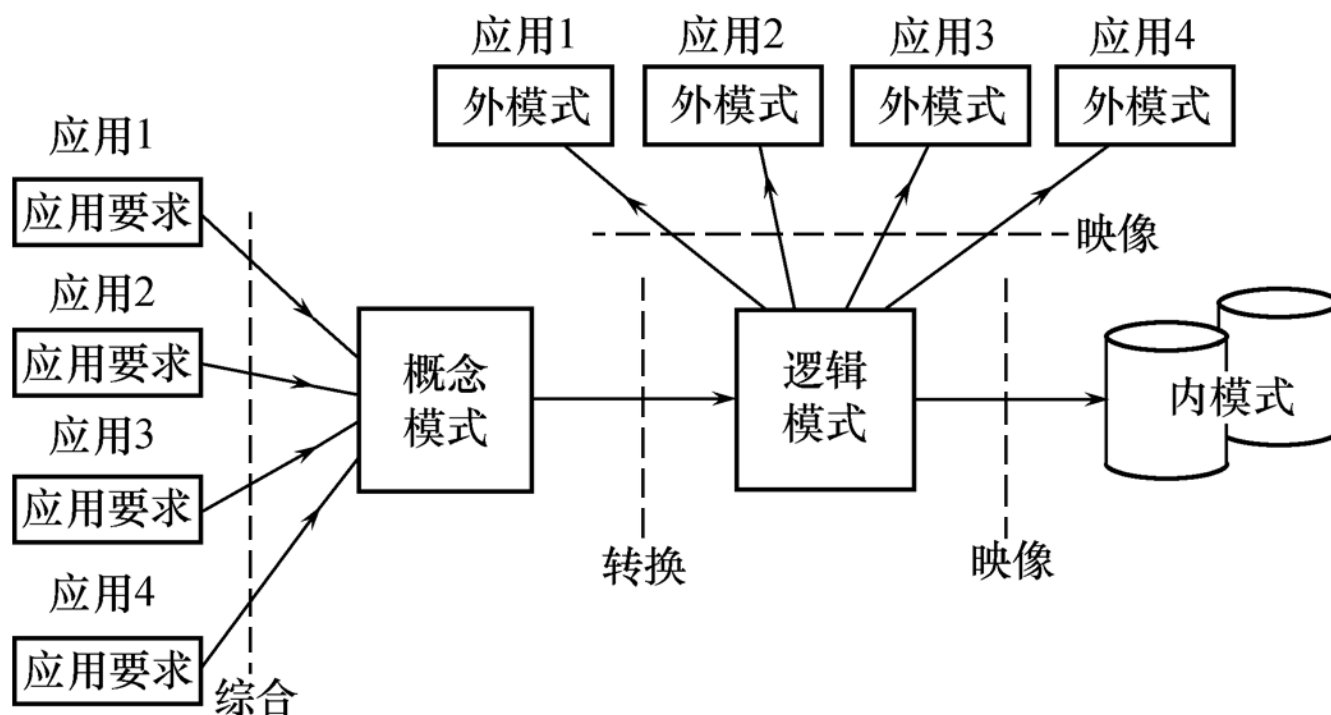
7.6 数据库的实施和维护

7.7 小结



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



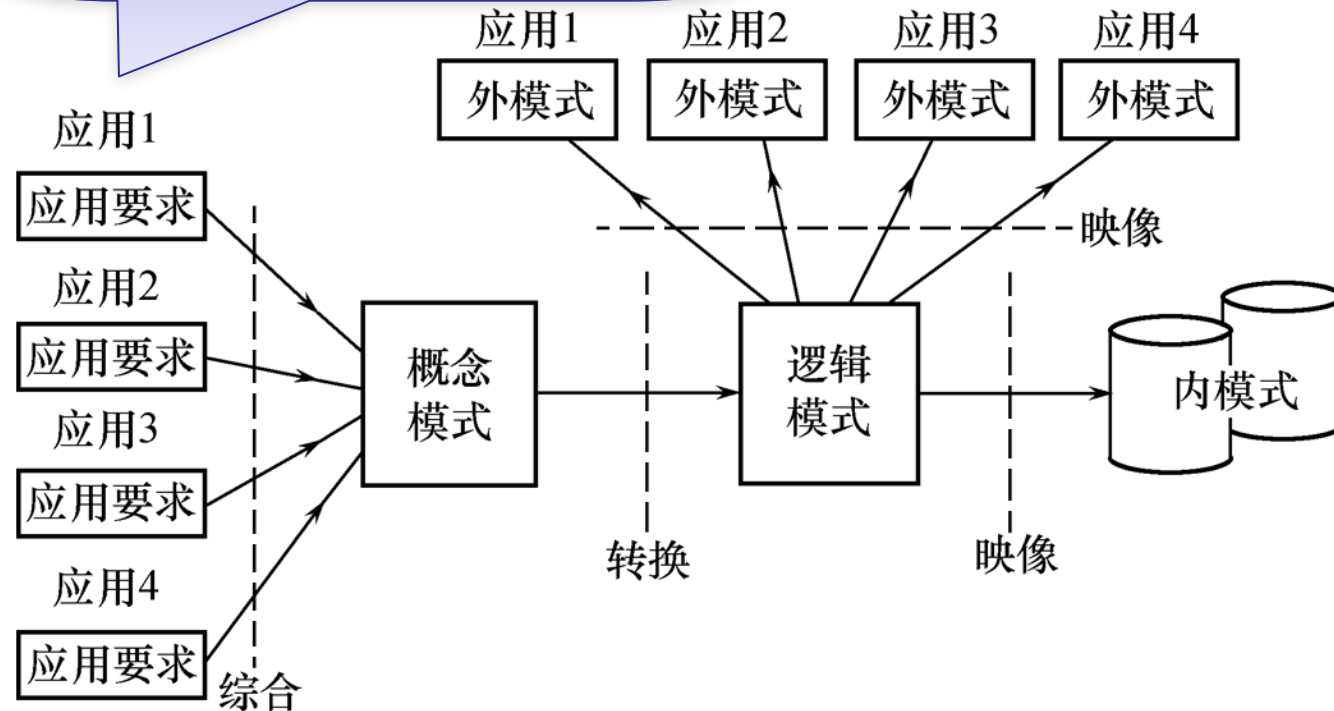
数据库的各级模式



数据库设计过程中的各级模式（续）

- 数据库设计不同阶段形成的数据库各级模式

需求分析阶段：
综合各个用户的应用需求



数据库的各级模式

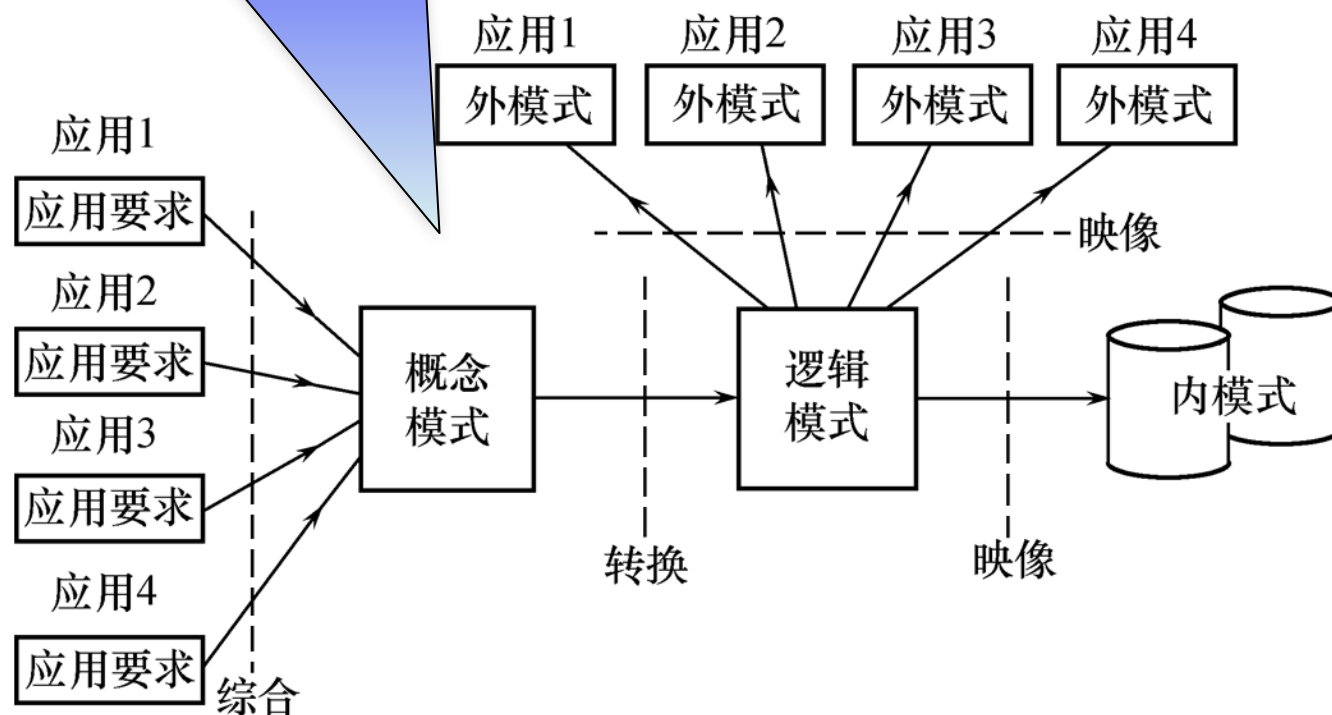


数据库设计过程中的各级模式（续）

• 数据库设计过程中的各级模式

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）



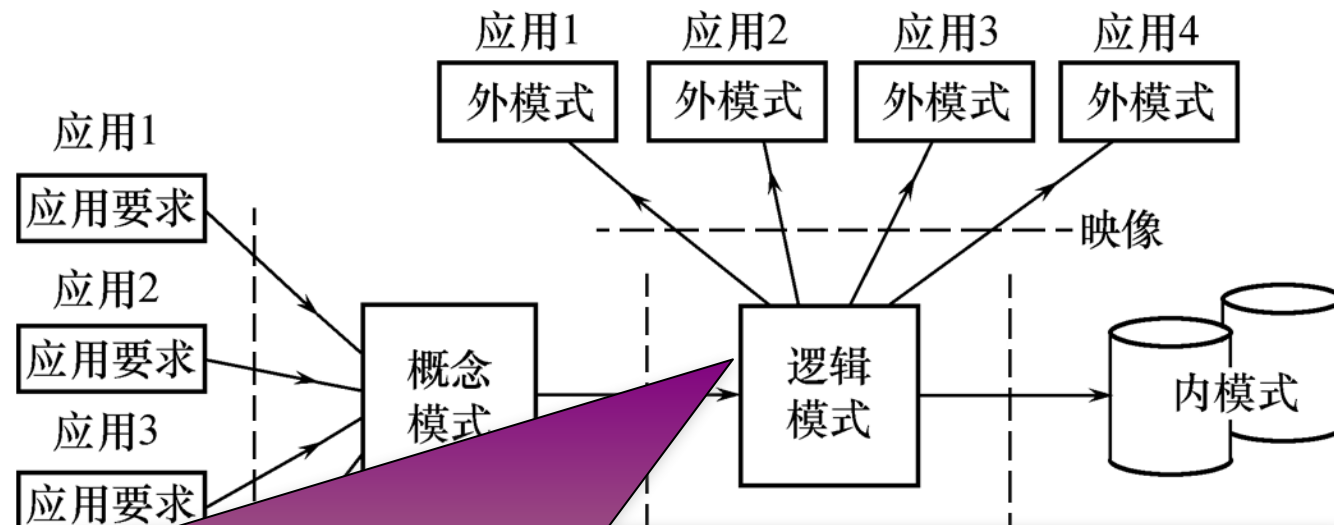
数据库的各级模式



China university of
Mining and Technology-Beijing

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：

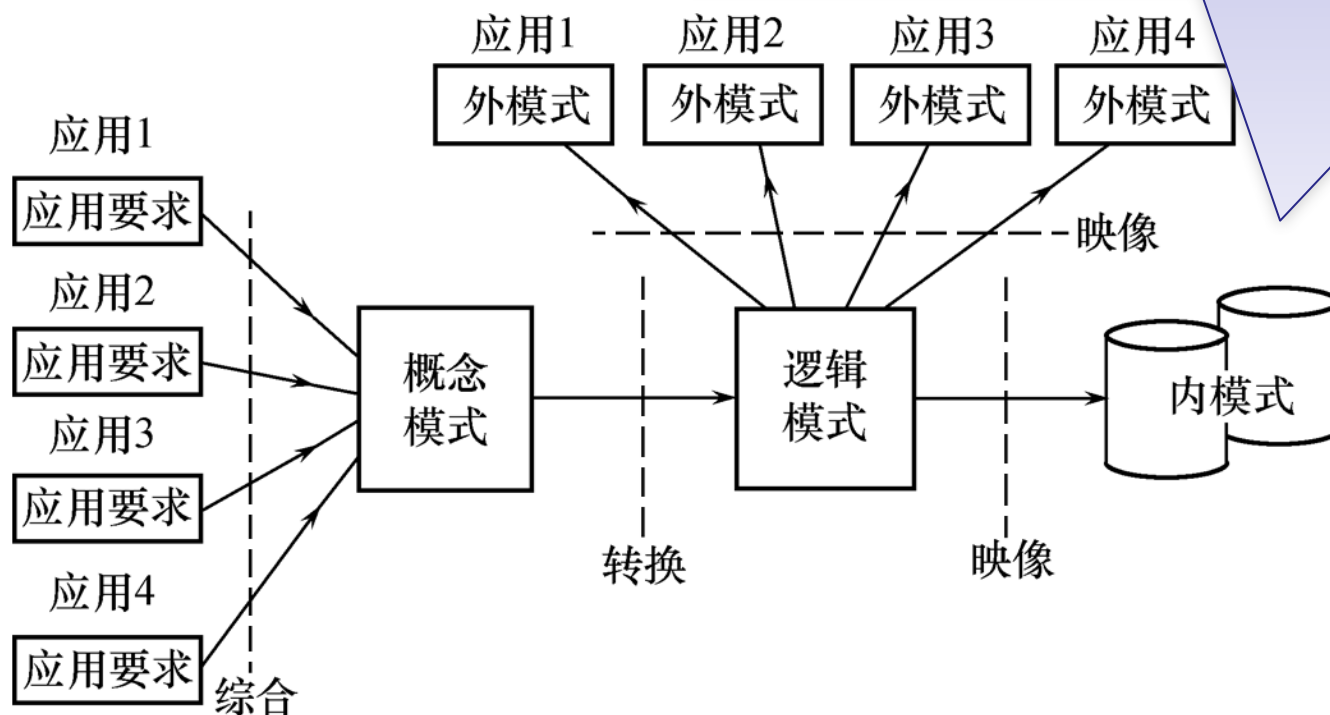
1. 首先将**E-R**图转换成具体的数据库产品支持的数据模型，如关系模型，形成数据库**逻辑模式**
2. 然后根据用户处理的要求、安全性的考虑，在基本表的基础上再建立必要的视图（**View**），形成数据的**外模式**

数据库设计过程中的各级模式（续）

- 数据库设计不

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



China university of
Mining and Technology-Beijing

1. 一个人存在于社会中，会有各种各样的身份，和不同的人相处会有不同的关系。请自行设计数据库（表结构，个数不限），保存一个人的名字和关系（比如父亲，朋友）。

2. 天猫“双十一”有个积分换魔盒的活动，总共有50万台天猫魔盒(box)，每个用户量(user)可以用 99 个天猫积分(point)兑换一台魔盒，且每人限换一台。参考(但不局限于)下面的下单逻辑：

- (1) 创建订单
- (2) 扣减用户积分。
- (3) 扣减盒库存。
- (4) 下单成功。

请用E-R图画此数据库概念模型并转换为关系模型？





密码:fpbw



China university of
Mining and Technology-Beijing