第7章　数据库设计

7.1 数据库设计概述

7.1.1 数据库设计的任务、内容和特点

7.1.1.1 数据库设计的任务

- 数据库设计是指根据用户需求研制数据库结构的过程，具体地说，是指对于一个给定的应用环境，构造最优的数据库模式，建立数据库及其应用系统，使之能有效的存储数据，满足用户的信息要求和处理要求。
- 也就是把现实世界中的数据，根据各种应用处理的要求，加以合理地组织，满足硬件和操作系统的特性，利用已有的DBMS来建立能够实现系统目标的数据库。

数据库设计的任务如图7.1所示。

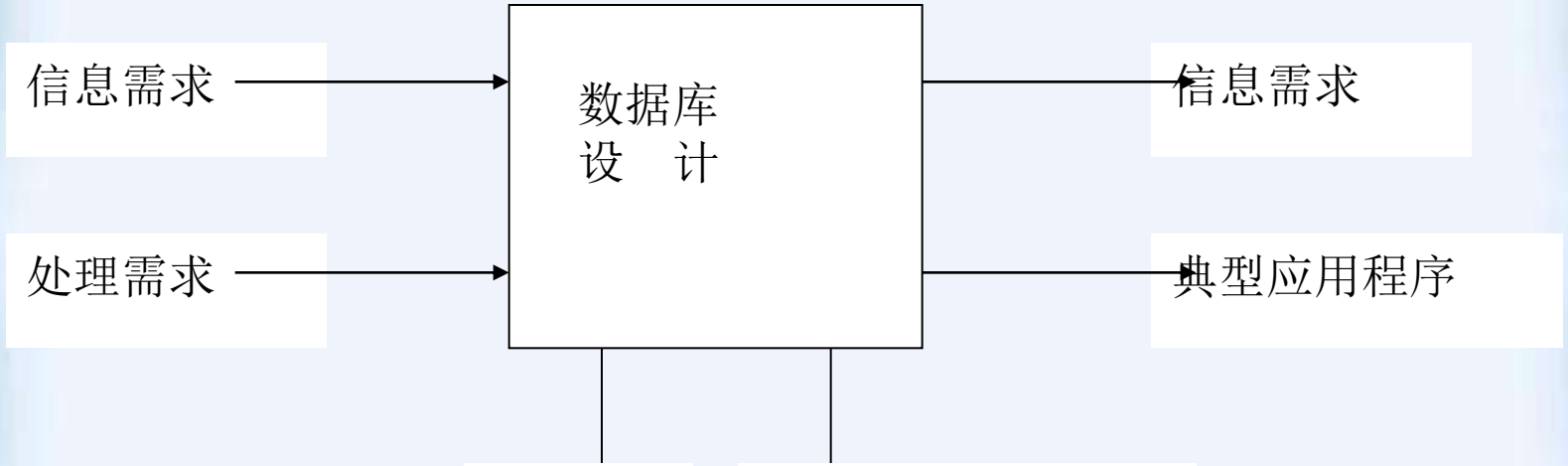


图7.1　数据库设计的任务

7.1.1.2 数据库设计的内容

- 数据库设计包括数据库的结构设计和数据库的行为设计两方面的内容。

1. 数据库的结构设计

- 数据库的结构设计指是根据给定的应用环境，进行数据库的模式或子模式的设计。
- 它包括数据库的概念设计、逻辑设计和物理设计。
- 数据库模式是各应用程序共享的结构，是静态的、稳定的，一经形成后通常情况下是不容易改变的，所以结构设计又称为静态模型设计。

2. 数据库的行为设计

- 数据库的行为设计是指确定数据库用户的行为和动作。而在数据库系统中，用户的行为和动作指用户对数据库的操作，这些要通过应用程序来实现，所以数据库的行为设计就是应用程序的设计。
- 用户的行为总是使数据库的内容发生变化，所以行为设计是动态的，行为设计又称为动态模型设计。

7.1.1.3 数据库设计的特点

- 在70年代末80年代初，人们为了研究数据库设计方法学的便利，曾主张将结构设计和行为设计两者分离，随着数据库设计方法学的成熟和结构化分析、设计方法的普遍使用，人们主张将两者作一体化的考虑，这样可以缩短数据库的设计周期，提高数据库的设计效率。

- 现代数据库的设计的特点是强调结构设计与行为设计相结合，是一种“反复探寻，逐步求精”的过程。首先从数据模型开始设计，以数据模型为核心进行展开，数据库设计和应用系统设计相结合，建立一个完整、独立、共享、冗余小、安全有效的数据库系统。
- 图7.2给出了数据库设计的全过程。

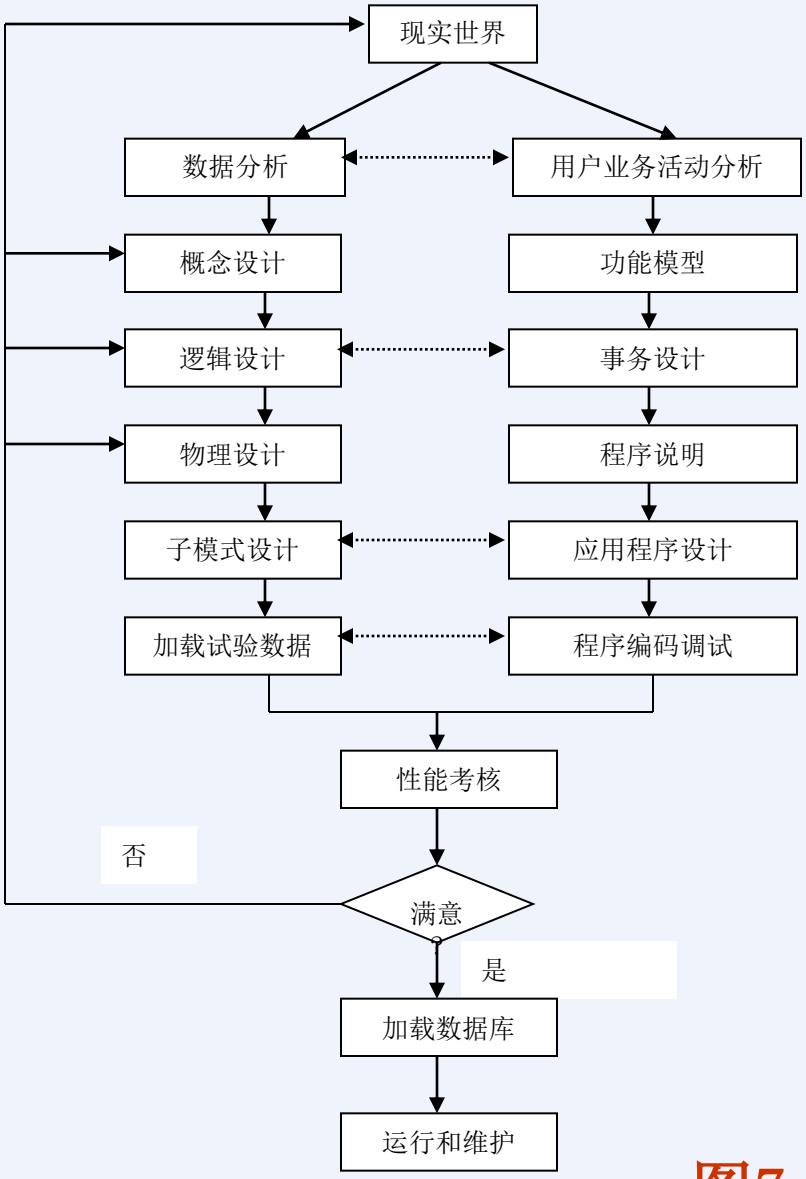
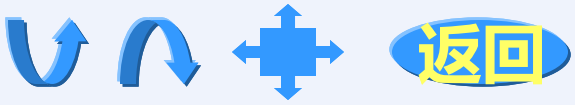


图7.2　数据库设计的全过程

返回

7.1.2 数据库设计方法简述

- 数据库设计方法目前可分为四类：直观设计法、规范设计法、计算机辅助设计法和自动化设计法。
- 直观设计法也叫手工试凑法，它是最早使用的数据库设计方法。这种方法依赖于设计者的经验和技巧，缺乏科学理论和工程原则的支持，设计的质量很难保证，常常是数据库运行一段时间后又发现各种问题，这样再重新进行修改，增加了系统维护的代价。因此这种方法越来越不适应信息管理发展的需要。
- 为了改变这种情况，1978年10月，来自三十多个国家的数据库专家在美国新奥尔良（New Orleans）市专门讨论了数据库设计问题，他们运用软件工程的思想和方法，提出了数据库设计的规范，这就是著名的新奥尔良法，它是目前公认的比较完整和权威的一种规范设计法。新奥尔良法将数据库设计分成需求分析（分析用户需求）、概念设计（信息分析和定义）、逻辑设计（设计实现）和物理设计（物理数据库设计）。目前，常用的规范设计方法大多起源于新奥尔良法，并在设计的每一阶段采用一些辅助方法来具体实现。
- 下面简单介绍几种常用的规范设计方法。



1. 基于E-R模型的数据库设计方法

- 基于E-R模型的数据库设计方法是由P.P.S.chen于1976年提出的数据库设计方法，其基本思想是在需求分析的基础上，用E-R（实体—联系）图构造一个反映现实世界实体之间联系的企业模式，然后再将此企业模式转换成基于某一特定的DBMS的概念模式。

2. 基于3NF的数据库设计方法

- 基于3NF的数据库设计方法是由S·Atre提出的结构化设计方法，其基本思想是在需求分析的基础上，确定数据库模式中的全部属性和属性间的依赖关系，将它们组织在一个单一的关系模式中，然后再分析模式中不符合3NF的约束条件，将其进行投影分解，规范成若干个3NF关系模式的集合。

➢ 其具体设计步骤分为五个阶段：

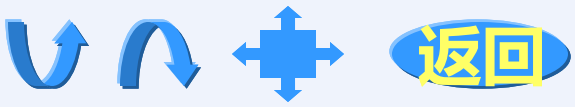
(1) 设计企业模式，利用规范化得到的3NF关系模式画出企业模式；

(2) 设计数据库的概念模式，把企业模式转换成DBMS所能接受的概念模式，并根据概念模式导出各个应用的外模式；

(3) 设计数据库的物理模式（存储模式）；

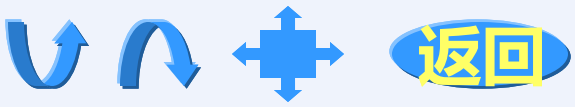
(4) 对物理模式进行评价；

(5) 实现数据库。

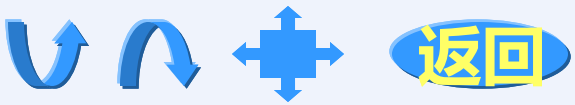


3．基于视图的数据库设计方法

- 此方法先从分析各个应用的数据着手，其基本思想是为每个应用建立自己的视图，然后再把这些视图汇总起来合并成整个数据库的概念模式。合并过程中要解决以下问题：
  (1) 消除命名冲突；
  (2) 消除冗余的实体和联系；
  (3) 进行模式重构，在消除了命名冲突和冗余后，需要对整个汇总模式进行调整，使其满足全部完整性约束条件。



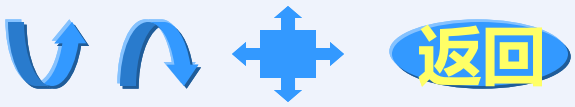
- 除了以上三种方法外，规范化设计方法还有实体分析法、属性分析法和基于抽象语义的设计方法等，这里不再详细介绍。
- 规范设计法从本质上来说仍然是手工设计方法，其基本思想是过程迭代和逐步求精。
- 计算机辅助设计法是指在数据库设计的某些过程中模拟某一规范化设计的方法，并以人的知识或经验为主导，通过人机交互方式实现设计中的某些部分。
- 目前许多计算机辅助软件工程（Computer Aided Software Engineering，CASE）工具可以自动或辅助设计人员完成数据库设计过程中的很多任务.比如SYSBASE公司的PowerDesigner和Oracle公司的Design 2000。



7.1.3　数据库设计的步骤

- 和其他软件一样，数据库的设计过程可以使用软件工程中的生存周期的概念来说明，称为“数据库设计的生存期”，它是指从数据库研制到不再使用它的整个时期。
- 按规范设计法可将数据库设计分为六个阶段（如图7.3所示）：
  （1）系统需求分析阶段
  （2）概念结构设计阶段
  （3）逻辑结构设计阶段
  （4）物理设计阶段
  （5）数据库实施阶段
  （6）数据库运行与维护阶段

- 该方法是分阶段完成的，每完成一个阶段，都要进行设计分析，评价一些重要的设计指标，把设计阶段产生的文档组织评审，与用户进行交流。如果设计的数据库不符合要求则进行修改，这种分析和修改可能要重复若干次，以求最后实现的数据库能够比较精确地模拟现实世界，能较准确地反映用户的需求，设计一个完善的数据库应用系统往往是六个阶段的不断反复的过程。
- 数据库设计中，前两个阶段是面向用户的应用要求，面向具体的问题；中间两个阶段是面向数据库管理系统；最后两个阶段是面向具体的实现方法。前四个阶段可统称为“分析和设计阶段”，后两个阶段称为“实现和运行阶段”。
- 六个阶段的主要工作各有不同。



1. 系统需求分析阶段

- 需求分析是整个数据库设计过程的基础，要收集数据库所有用户的信息内容和处理要求，并加以规格化和分析。这是最费时、最复杂的一步，但也是最重要的一步，相当于待构建的数据库大厦的地基，它决定了以后各步设计的速度与质量。需求分析做得不好，可能会导致整个数据库设计返工重做。在分析用户需求时，要确保用户目标的一致性。

2. 概念结构设计阶段

- 概念设计是把用户的信息要求统一到一个整体逻辑结构中，此结构能够表达用户的要求，是一个独立于任何DBMS软件和硬件的概念模型。

3. 逻辑结构设计阶段

- 逻辑设计是将上一步所得到的概念模型转换为某个DBMS所支持的数据模型，并对其进行优化。

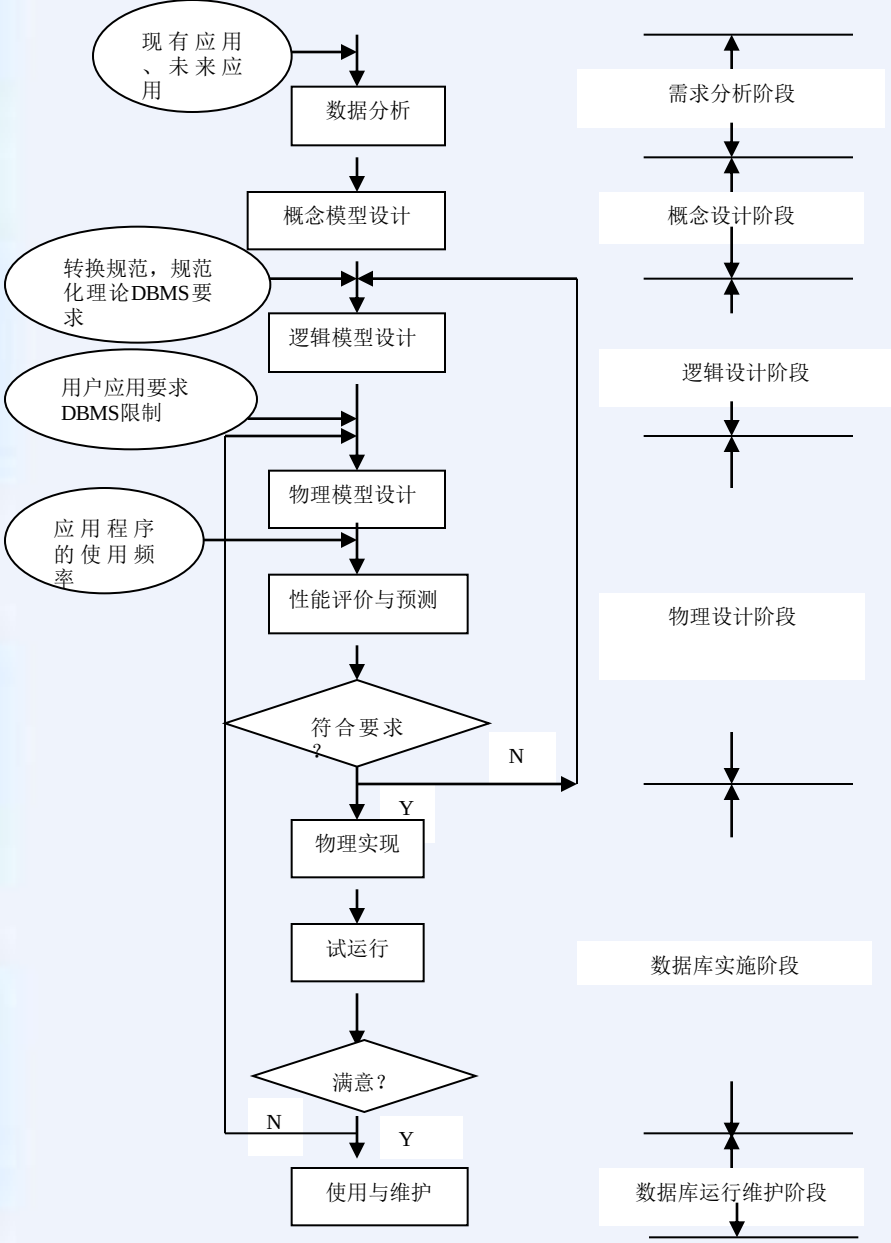


图7.3　数据库的设计步骤

返回

4. 物理设计阶段

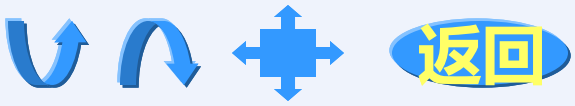
- 物理设计是为逻辑数据模型建立一个完整的能实现的数据库结构，包括存储结构和存取方法。
- 上述分析和设计阶段是很重要的，如果做出不恰当的分析或设计，则会导致一个不恰当或反应迟钝的应用系统。

5. 数据库实施阶段

- 根据物理设计的结果把原始数据装入数据库，建立一个具体的数据库并编写和调试相应的应用程序。应用程序的开发目标是开发一个可依赖的有效的数据库存取程序，来满足用户的处理要求。

6. 数据库运行与维护阶段

- 这一阶段主要是收集和记录实际系统运行的数据，数据库运行的记录用来提高用户要求的有效信息，用来评价数据库系统的性能，进一步调整和修改数据库。在运行中，必须保持数据库的完整性，并能有效地处理数据库故障和进行数据库恢复。在运行和维护阶段，可能要对数据库结构进行修改或扩充。



- 可以看出，以上六个阶段是从数据库应用系统设计和开发的全过程来考察数据库设计的问题。因此，它既是数据库也是应用系统的设计过程。在设计过程中，努力使数据库设计和系统其他部分的设计紧密结合，把数据和处理的需求收集、分析、抽象、设计和实现在各个阶段同时进行、相互参照、相互补充，以完善两方面的设计。按照这个原则，数据库过程各个阶段的设计可用图7.4描述。
- 在上图有关处理特性的描述中，采用的设计方法和工具属于软件工程和管理信息系统等课程中的内容，本书不再讨论，这里重点介绍数据特性的设计描述以及在结构特性中参照处理特性设计以完善数据模型设计的问题。
- 以下各节分别详细介绍数据库设计的六个阶段。

7.2 系统需求分析

- 需求分析是数据库设计的起点，为以后的具体设计作准备。
- 需求分析的结果是否准确的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。
- 经验证明，由于设计要求的不正确或误解，直到系统测试阶段才发现许多错误，则纠正起来要付出很大代价。
- 因此，必须高度重视系统的需求分析。

7.2.1　需求分析的任务

- 从数据库设计的角度来看，需求分析的任务是：对现实世界要处理的对象（组织、部门、企业）等进行详细的调查，通过对原系统的了解，收集支持新系统的基础数据并对其进行处理，在此基础上确定新系统的功能。

具体地说，需求分析阶段的任务包括以下三项：

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) 反映新系统信息的数据流图
逻辑结构设计	某种数据模型 关系模型	系统结构图 非关系模型（模块结构图）
物理设计	存储安排 存取方法选择 存取路径建立	模块设计 IPO表
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
运行维护	性能测试，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）

图7.4　数据库各个设计阶段的描述

返回

1．调查分析用户的活动

- 这个过程通过对新系统运行目标的研究，对现行系统所存在的主要问题的分析以及制约因素的分析，明确用户总的需求目标，确定这个目标的功能域和数据域。具体做法是：
  （1）调查组织机构情况，包括该组织的部门组成情况，各部门的职责和任务等。
  （2）调查各部门的业务活动情况，包括各部门输入和输出的数据与格式、所需的表格与卡片、加工处理这些数据的步骤、输入输出的部门等。

2．收集和分析需求数据，确定系统边界

- 在熟悉业务活动的基础上，协助用户明确对新系统的各种需求，包括用户的信息需求、处理需求、安全性和完整性的需求等。
  - （1）信息需求指目标范围内涉及的所有实体、实体的属性以及实体间的联系等数据对象，也就是用户需要从数据库中获得信息的内容与性质。由信息要求可以导出数据要求，即在数据库中需要存储哪些数据。
  - （2）处理需求指用户为了得到需求的信息而对数据进行加工处理的要求，包括对某种处理功能的响应时间，处理的方式（批处理或联机处理）等。
  - （3）安全性和完整性的需求。在定义信息需求和处理需求的同时必须相应确定安全性和完整性约束。
- 在收集各种需求数据后，对前面调查的结果进行初步分析，确定新系统的边界，确定哪些功能由计算机完成或将来准备让计算机完成，哪些活动由人工完成。由计算机完成的功能就是新系统应该实现的功能。

3．编写需求分析说明书

- 系统分析阶段的最后是编写系统分析报告，通常称为需求规范说明书。需求规范说明书是对需求分析阶段的一个总结。编写系统分析报告是一个不断反复、逐步深入和逐步完善的过程，系统分析报告应包括如下内容：
  （1）系统概况，系统的目标、范围、背景、历史和现状；
  （2）系统的原理和技术，对原系统的改善；
  （3）系统总体结构与子系统结构说明；
  （4）系统功能说明；
  （5）数据处理概要、工程体制和设计阶段划分；
  （6）系统方案及技术、经济、功能和操作上的可行性。
- 完成系统的分析报告后，在项目单位的领导下要组织有关技术专家评审系统分析报告，这是对需求分析结构的再审查。审查通过后由项目方和开发方领导签字认可。

随系统分析报告提供下列附件：

（1）系统的硬件、软件支持环境的选择及规格要求（所选择的数据库管理系统、操作系统、汉字平台、计算机型号及其网络环境等）。

（2）组织机构图、组织之间联系图t 各机构功能业务一览图。

（3）数据流程图、功能模块图和数据字典等图表。

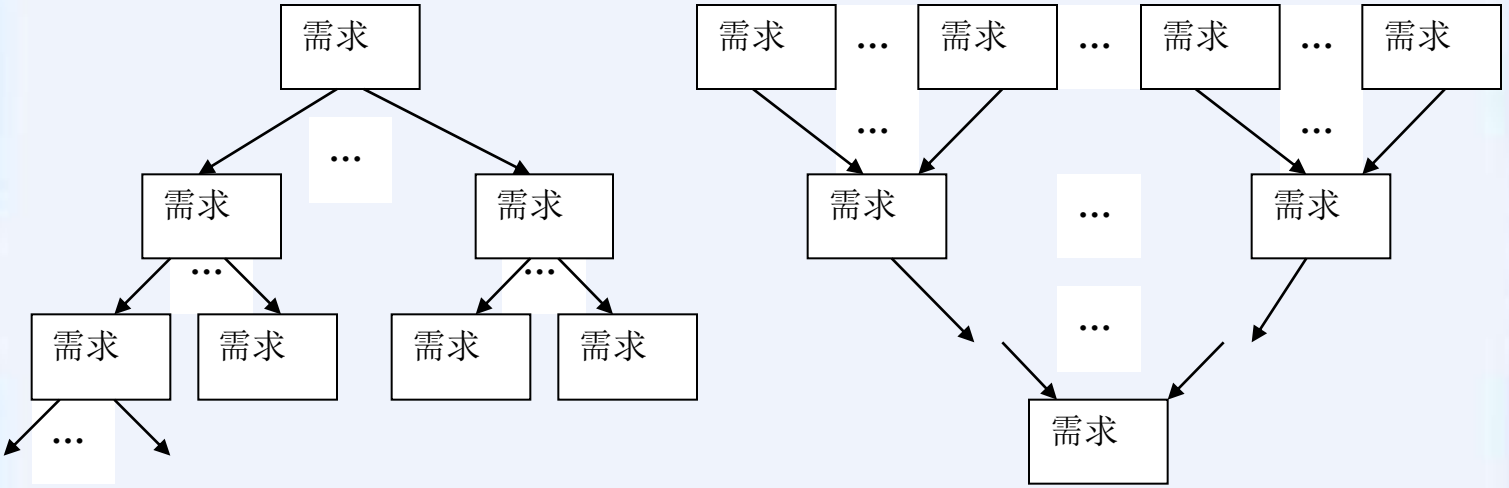
- 如果用户同意系统分析报告和方案设计，在与用户进行详尽商讨的基础上，最后签订技术协议书。
- 系统分析报告是设计者和用户一致确认的权威性文献，是今后各阶段设计和工作的依据。

7.2.2 需求分析的方法

- 用户参加数据库设计是数据应用系统设计的特点，是数据库设计理论不可分割的一部分。
- 在数据需求分析阶段，任何调查研究没有用户的积极参加是寸步难行的，设计人员应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，所以这个过程中不同背景的人员之间互相了解与沟通是至关重要的，同时方法也很重要。
- 用于需求分析的方法有多种，主要方法有自顶向下和自底向上两种，如图7.5所示。
- 其中自顶向下的分析方法（**Structured Analysis**，简称**SA**方法）是最简单实用的方法。**SA**方法从最上层的系统组织机构入手，采用逐层分解的方式分析系统，用数据流图（**Data Flow Diagram**，**DFD**）和数据字典（**Data Dictionary**，**DD**）描述系统。
- 下面对数据流图和数据字典作些简单的介绍。

1．数据流图

- 使用SA方法，任何一个系统都可抽象为图7.6所示的数据流图。
- 在数据流图中，用命名的箭头表示数据流，用圆圈表示处理，用矩形或其他形状表示存储。
- 图7.7是一个简单的数据流图。一个简单的系统可用一张数据流图来表示。当系统比较复杂时，为了便于理解，控制其复杂性，可以采用分层描述的方法。一般用第一层描述系统的全貌，第二层分别描述各子系统的结构。如果系统结构还比较复杂，那么可以继续细化，直到表达清楚为止。在处理功能逐步分解的同时，它们所用的数据也逐级分解，形成若干层次的数据流图。数据流图表达了数据和处理过程的关系。
- 在SA方法中，处理过程的处理逻辑常常借助判定表或判定树来描述，而系统中的数据则是借助数据字典来描述



(a) 自顶向下的需求分析

(b) 自底向上的需求分析

图7.5 需求分析的方法

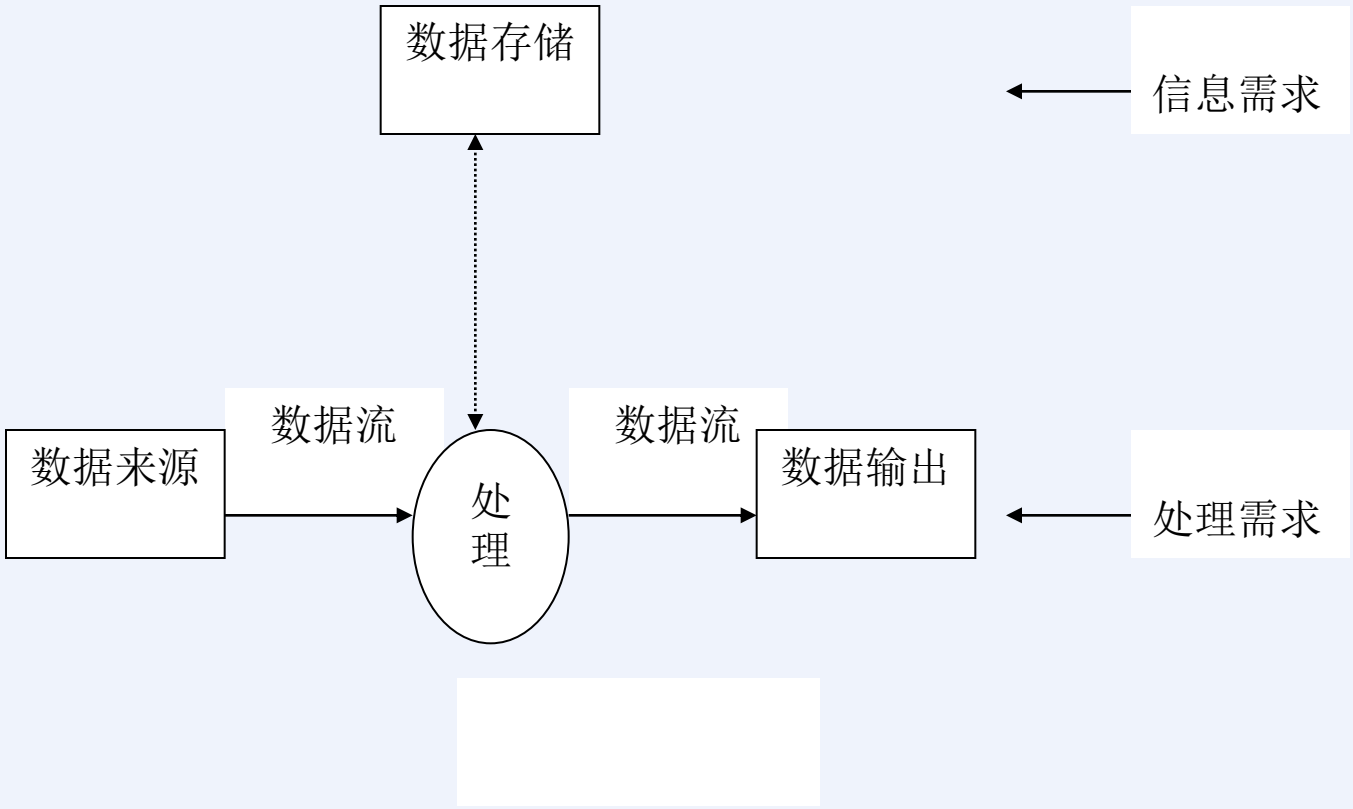


图7.6 数据流图

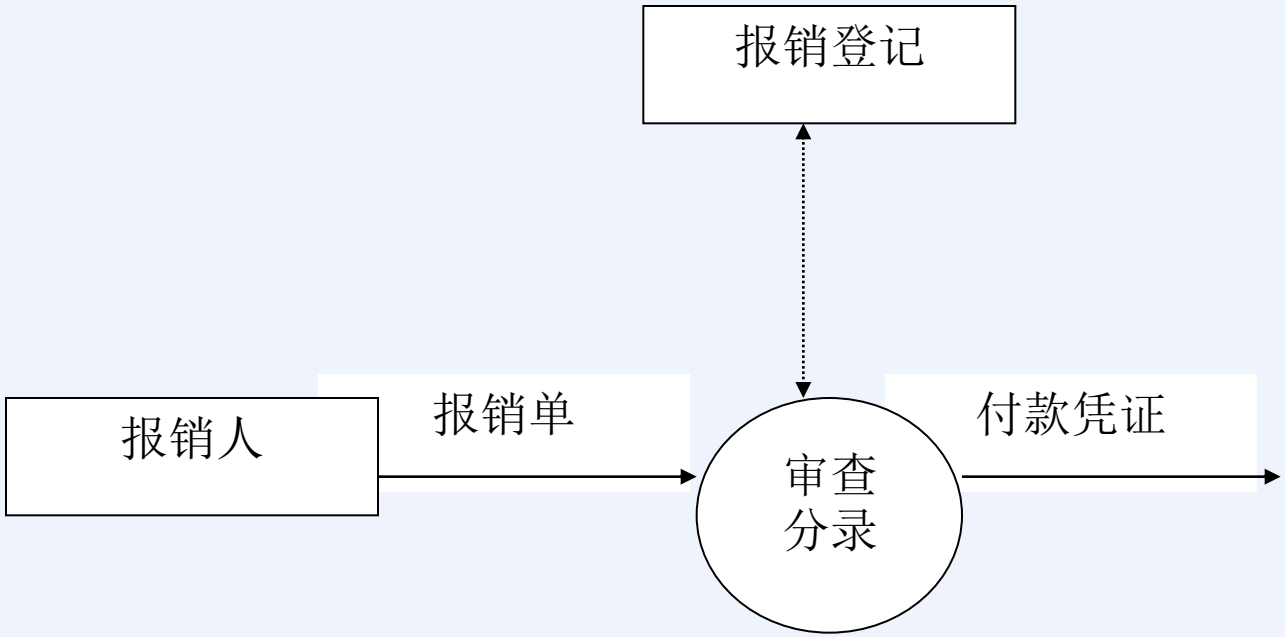


图7.7 数据流图示例

2．数据字典

- 数据字典是对系统中数据的详细描述，是各类数据结构和属性的清单。它与数据流图互为注释。
- 数据字典贯穿于数据库需求分析直到数据库运行的全过程，在不同的阶段其内容和用途各有区别。
- 在需求分析阶段，它通常包含以下五部分内容。

(1) 数据项

- 数据项是数据的最小单位，其具体内容包括：数据顶名、含义说明、别名、类型、长度、取值范围、与其他数据项的关系。
- 其中，取值范围、与其他数据项的关系这两项内容定义了完整性约束条件，是设计数据检验功能的依据。

(2) 数据结构

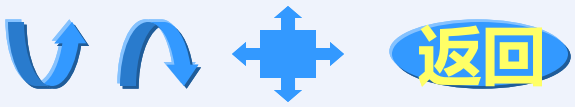
- 数据结构是数据项有意义的集合。内容包括：数据结构名、含义说明，这些内容组成数据项名。

(3) 数据流

- 数据流可以是数据项，也可以是数据结构，它表示某一处理过程中数据在系统内传输的路径。
- 内容包括：数据流名、说明、流出过程、流入过程，这些内容组成数据项或数据结构。
- 其中，流出过程说明该数据流由什么过程而来；流入过程说明该数据流到什么过程。

(4) 数据存储

- 处理过程中数据的存放场所，也是数据流的来源和去向之一。可以是手工凭证，手工文档或计算机文件。
- 包括｛数据存储名，说明，输入数据流，输出数据流，组成：｛数据项或数据结构｝，数据量，存取频度，存取方式｝。
- 其中，存取频度是指每天（或每小时、或每周）存取几次，每次存取多少数据等信息。存取方法指的是批处理，还是联机处理；是检索还是更新；是顺序检索还是随机检索等。



(5) 处理过程

- 处理过程的处理逻辑通常用判定表或判定树来描述，数据字典只用来描述处理过程的说明性信息。
- 处理过程包括｛处理过程名，说明，输入：｛数据流｝，输出：｛数据流｝，处理，｛简要说明｝｝。
- 其中，简要说明主要说明处理过程的功能及处理要求。
- 功能是指该处理过程用来做什么（不是怎么做），处理要求指该处理频度要求，如单位时间里处理多少事务、多少数据量、响应时间要求等，这些处理要求是后面物理设计的输入及性能评价的标准。

➢ 最终形成的数据流图和数据字典为“需求分析说明书”的主要内容，这是下一步进行概念设计的基础。

7.3 概念结构设计

7.3.1 概念结构设计的必要性

- 在需求分析阶段，设计人员充分调查并描述了用户的需求，但这些需求只是现实世界的具体要求，应把这些需求抽象为信息世界的结构，才能更好地实现用户的需求。
- 概念设计就是将需求分析得到的用户需求抽象为信息结构，即概念模型。
- 在早期的数据库设计中，概念设计并不是一个独立的设计阶段。当时的设计方式是在需求分析之后，接着就进行逻辑设计。这样设计人员在进行逻辑设计时，考虑的因素太多，既要考虑用户的信息，又要考虑具体**DBMS**的限制，使得设计过程复杂化，难以控制。为了改善这种状况，**P.P.S.chen**设计了基于**E-R**模型的数据库设计方法，即在需求分析和逻辑设计之间增加了一个概念设计阶段。在这个阶段，设计人员仅从用户角度看待数据及处理要求和约束，产生一个反映用户观点的概念模型，然后再把概念模型转换成逻辑模型。这样做有三个好处：

(1) 从逻辑设计中分离出概念设计以后，各阶段的任务相对单一化，设计复杂程度大大降低，便于组织管理。

(2) 概念模型不受特定的DBMS的限制，也独立于存储安排和效率方面的考虑，因而比逻辑模型更为稳定。

(3) 概念模型不含具体的DBMS所附加的技术细节，更容易为用户所理解，因而更有可能准确反映用户的信息需求。

- 设计概念模型的过程称为概念设计。概念模型在数据库的各级模型中的地位如图7.8所示。

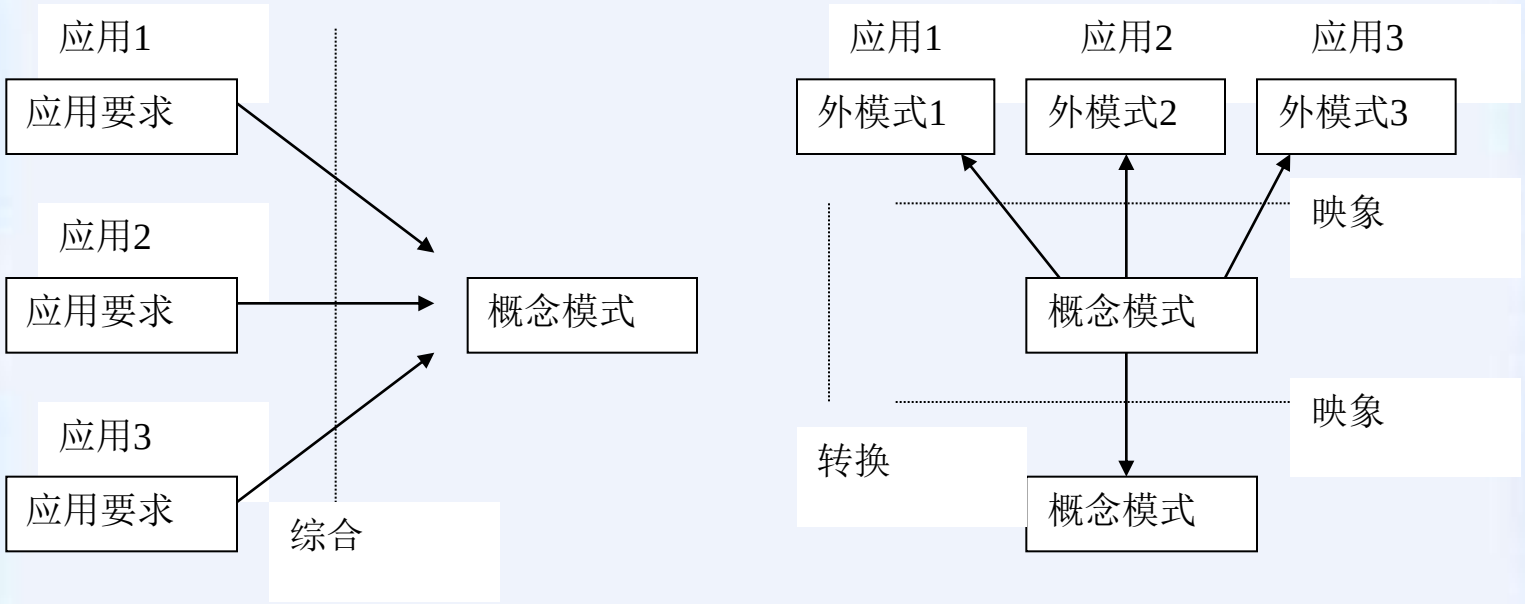


图7.8 数据库各级模型的形成

6.3.2 概念模型的特点

- 概念模型作为概念设计的表达工具，为数据库提供一个说明性结构，是设计数据库逻辑结构即逻辑模型的基础。因此，概念模型必须具备以下特点：
  - (1) 语义表达能力丰富。概念模型能表达用户的各种需求，充分反映现实世界，包括事物和事物之间的联系、用户对数据的处理要求，它是现实世界的一个真实模型。
  - (2) 易于交流和理解。概念模型是DBA、应用开发人员和用户之间的主要界面，因此，概念模型要表达自然、直观和容易理解，以便和不熟悉计算机的用户交换意见，用户的积极参与是保证数据库设计和成功的关键。
  - (3) 易于修改和扩充。概念模型要能灵活地加以改变，以反映用户需求和现实环境的变化。
  - (4) 易于向各种数据模型转换。概念模型独立于特定的DBMS，因而更加稳定，能方便地向关系模型、网状模型或层次模型等各种数据模型转换。
- 人们提出了许多概念模型，其中最著名、最实用的一种是**E-R**模型，它将现实世界的信息结构统一用属性、实体以及它们之间的联系来描述。

6.3.3 概念结构设计的方法与步骤

1. 概念结构设计的方法

- 设计概念结构的E-R模型可采用四种方法。

(1) 自顶向下。先定义全局概念结构E-R模型的框架，再逐步细化。如图7.9（a）所示。

(2) 自底向上。先定义各局部应用的概念结构E-R模型，然后将它们集成，得到全局概念结构E-R模型。如图7.9（b）所示。

(3) 逐步扩张。先定义最重要的核心概念E-R模型，然后向外扩充，以滚雪球的方式逐步生成其他概念结构E-R模型。如图7.9（c）所示。

(4) 混合策略。该方法采用自顶向下和自底向上相结合的方法，先自顶向下定义全局框架，再以它为骨架集成自底向上方法中设计的各个局部概念结构。

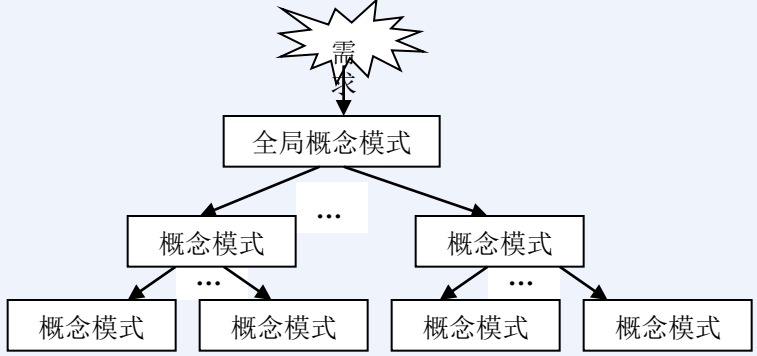
- 其中最常用的方法是自底向上。即自顶向下地进行需求分析，再自底向上地设计概念结构。

2．概念结构设计的步骤

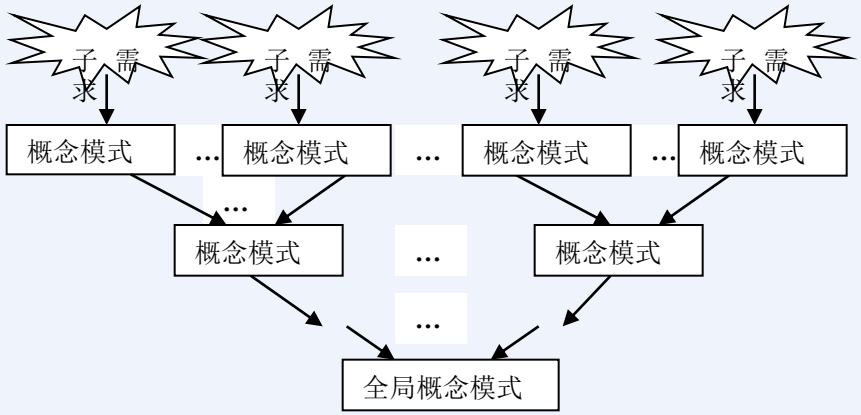
- 自底向上的设计方法可分为两步，如图7.10所示：
  （1）进行数据抽象，设计局部E-R模型，即设计用户视图。
  （2）集成各局部E-R模型，形成全局E-R模型，即视图的集成。

3．数据抽象与局部E-R模型设计

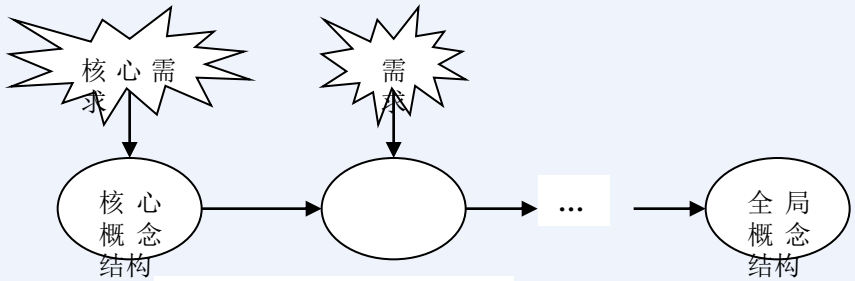
- 概念结构是对现实世界的一种抽象。
- 所谓抽象是对实际的人、物、事和概念进行人为处理，它抽取人们关心的共同特性，忽略非本质的细节，并把这些特性用各种概念精确地加以描述，这些概念组成了某种模型。
- 概念结构设计首先要根据需求分析得到的结果（数据流图、数据字典等）对现实世界进行抽象，设计各个局部E-R模型。



(a) 自顶向下的设计方法



(b) 自底向上的设计方法



(c) 逐步扩张的设计方法

图6.9 概念结构设计的方法

返回

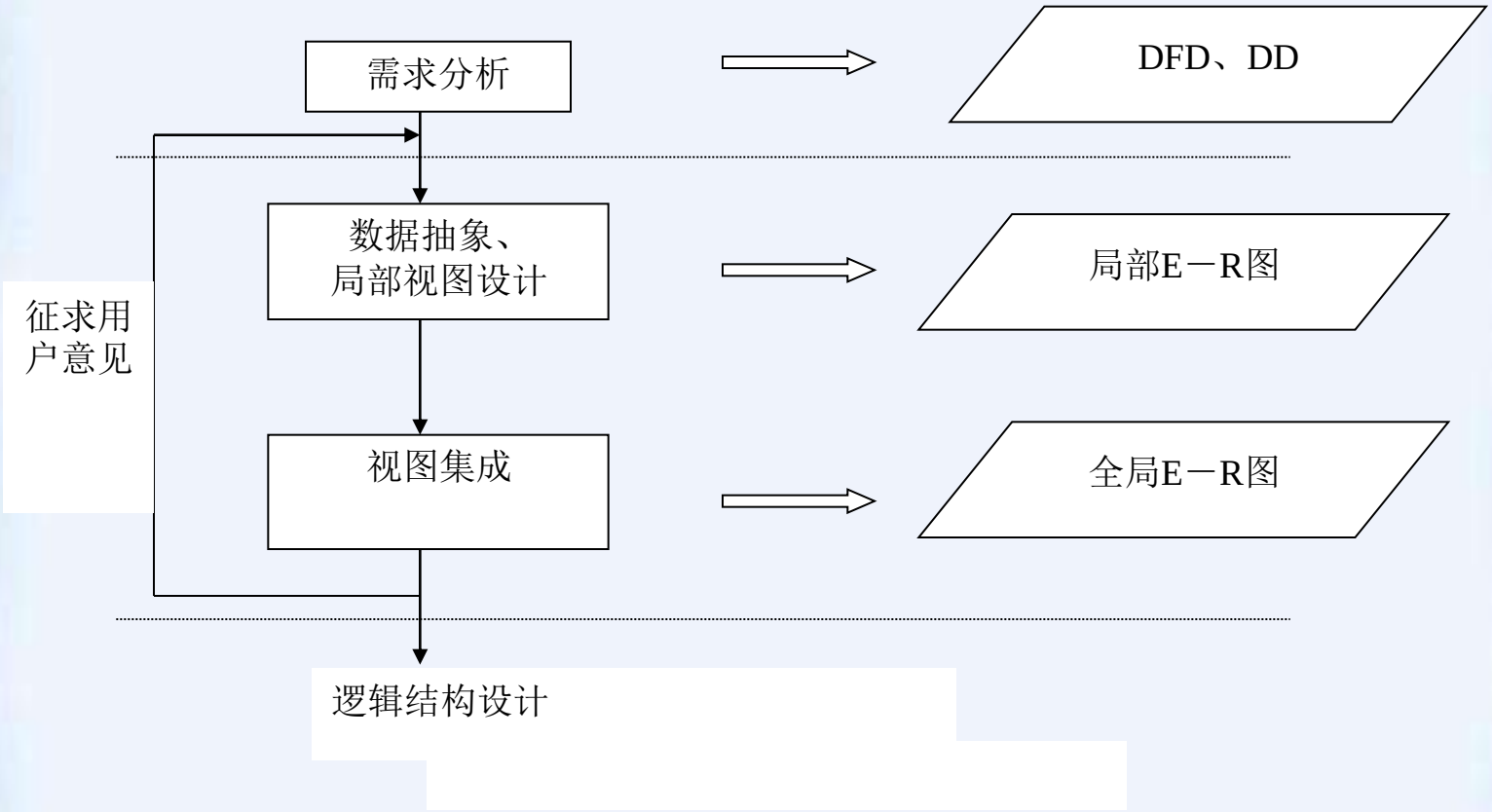


图7.10 自底向上方法的设计步骤

(1) E-R方法

- E-R方法是“实体-联系方法”（Entity-Relationship Approach）的简称。它是描述现实世界概念结构模型的有效方法。用E-R方法建立的概念结构模型称为E-R模型，或称为E-R图。
- E-R图基本成分包含实体型、属性和联系。（在第1章已经介绍过它们的基本概念，这里只给出它们的表示方法。）

①实体型：用矩形框表示，框内标注实体名称。如图7.11（a）所示。

②属性：用椭圆形框表示，框内标注属性名称。如图7.11（b）所示。

③联系：指实体之间的联系，有一对一（1：1），一对多（1：*n*）或多对多（*m* ：*n*）三种联系类型。例如系主任领导系，学生属于某一系，学生选修课程，工人生产产品，这里“领导”、“属于”、“选修”、“生产”表示实体间的联系，可以作为联系名称。联系用菱形框表示，框内标注联系名称。如图7.11（c）所示。

- 现实世界的复杂性导致实体联系的复杂性。表现在E-R图上可以归结为图7.12所示的几种基本形式：

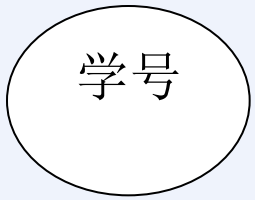
  ①两个实体之间的联系，如图7.12（a）所示。

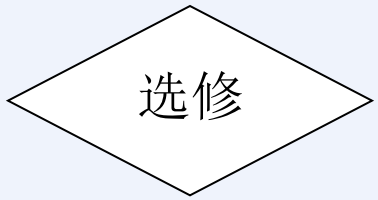
  ②两个以上实体间的联系，如图7.12（b）所示。

  ③同一实体集内部各实体之间的联系，例如一个部门内的职工有领导与被领导的联系，即某一职工（干部）领导若干名职工，而一个职工（普通员工）仅被另外一个职工直接领导，这就构成了实体内部的一对多的联系。如图7.12（c）所示。

- 需要注意的是，因为联系本身也是一种实体型，所以联系也可以有属性。如果一个联系具有属性，则这些联系也要用无向边与该联系连接起来。例如，学生选修的课程有相应的成绩。这里的“成绩”既不是学生的属性，也不是课程的属性，只能是学生选修课程的联系的属性。图7.12（b）中“供应数量”是“供应”联系的属性。

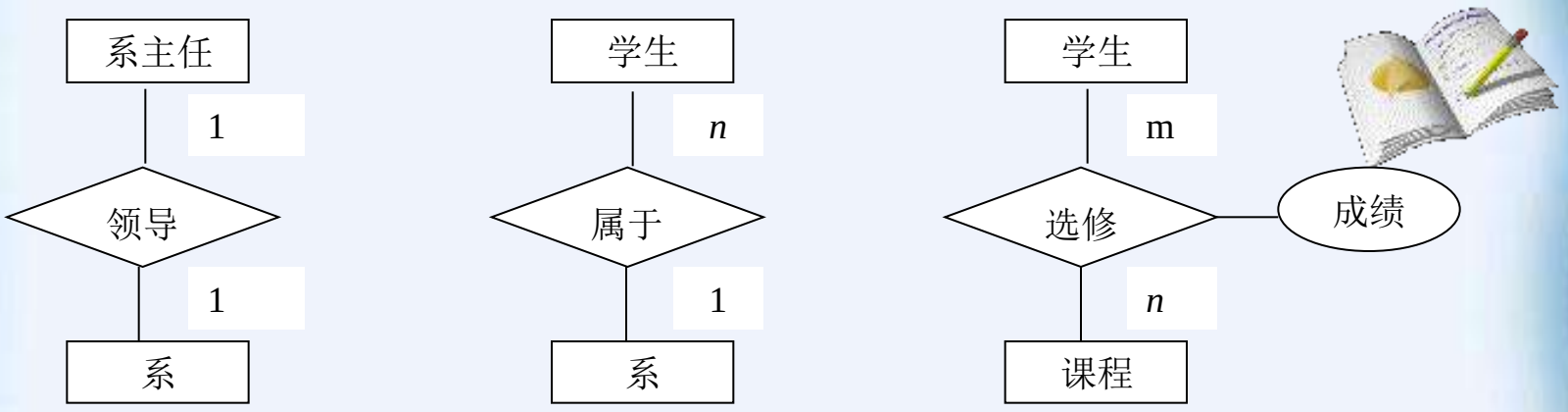
- E-R图的基本思想就是分别用矩形框、椭圆形框和菱形框表示实体、属性和联系，使用无向边将属性与其相应的实体连接起来，并将联系分别和有关实体相连接，注明联系类型。
- 图7.12为几个E-R图的例子，只给出了实体及其E-R图，省略了实体的属性。
- 图7.13为一个描述学生与课程联系的完整的E-R图。



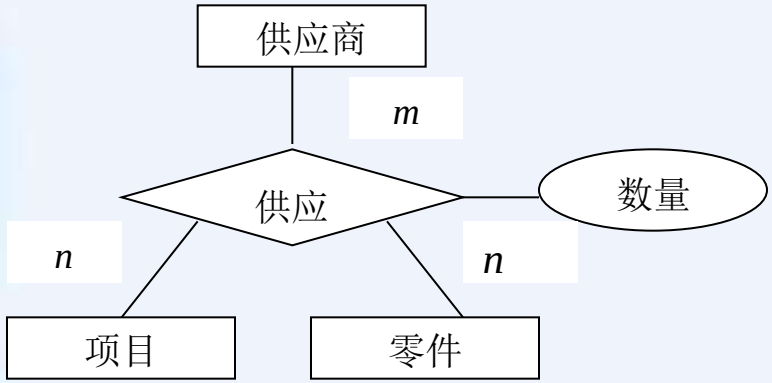


(a)实体　(b)属性　(c)联系

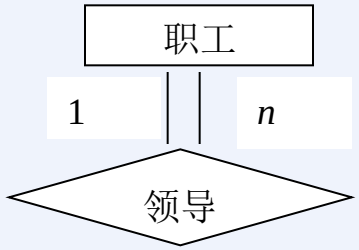
7.11 E-R图的三种基本成份及其图形的表示方法



(a)两个实体之间的联系



(b)多个实体之间的联系



(c)实体集内部的联系

m

图7.12 实体及其联系图

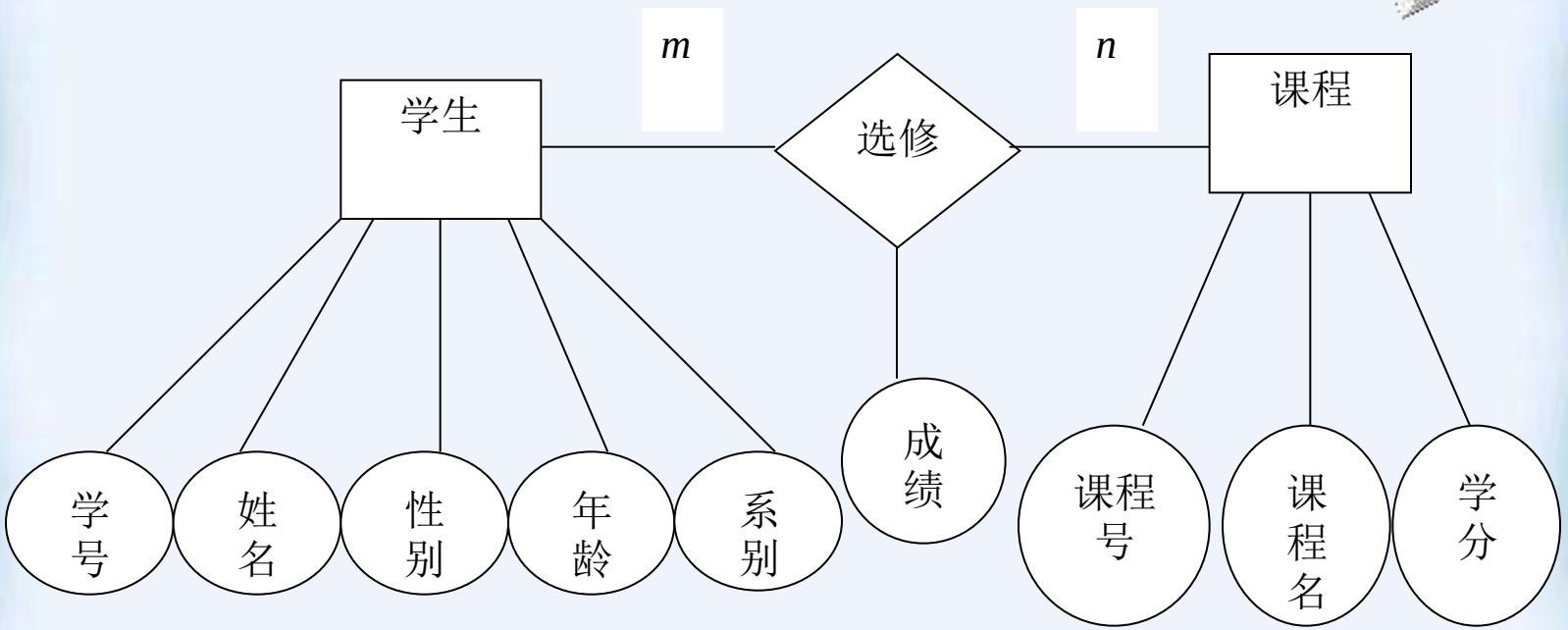


图6.13 学生与课程联系的完整的E－R图

(2) 数据抽象

- 在系统需求分析阶段，最后得到了多层数据流图、数据字典和系统分析报告。建立局部E-R模型，就是根据系统的具体情况，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，让这组图中每一部分对应一个局部应用。在前面选好的某一层次的数据流图中，每个局部应用都对应了一组数据流图，局部应用所涉及的数据存储在数据字典中。现在就是要将这些数据从数据字典中抽取出来，参照数据流图，确定每个局部应用包含哪些实体，这些实体又包含哪些属性，以及实体之间的联系及其类型。
- 设计局部**E-R**模型的关键就是正确划分实体和属性。实体和属性之间在形式上并无可以明显区分的界限，通常是按照现实世界中事物的自然划分来定义实体和属性，将现实世界中的事物进行数据抽象，得到实体和属性。
- 一般有3种数据抽象：分类、聚集、概括

①分类（Classification）

- 分类定义某一类概念作为现实世界中一组对象的类型，将一组具有某些共同特性和行为的对象抽象为一个实体。对象和实体之间是“is member of”的关系。
- 例如，在教学管理中，“赵亦”是一名学生，表示“赵亦”是学生中的一员，她具有学生们共同的特性和行为。

②聚集（Aggregation）

- 聚集定义某一类型的组成成份，将对象类型的组成成份抽象为实体的属性。组成成份与对象类型之间是“**is part of**”的关系。
- 例如，学号、姓名、性别、年龄、系别等可以抽象为学生实体的属性，其中学号是标识学生实体的主键。

➢ **概括: 类型间的子集联系.**

(2) 局部E-R模型设计

- 数据抽象后得到了实体和属性，实际上实体和属性是相对而言的，往往要根据实际情况进行必要的调整。在调整中要遵循两条原则：

①实体具有描述信息，而属性没有。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，联系只能发生在实体之间。

- 例如：学生是一个实体，学号、姓名、性别、年龄、系别等是学生实体的属性，系别只表示学生属于哪个系，不涉及系的具体情况，换句话说，没有需要进一步描述的特性，即是不可分的数据项，则根据原则①可以作为学生实体的属性。但如果考虑一个系的系主任、学生人数、教师人数、办公地点等，则系别应看作一个实体。如图7.14所示。
- 又如，“职称”为教师实体的属性，但在涉及住房分配时，由于分房与职称有关，即职称与住房实体之间有联系，则根据原则②，职称应作为一个实体。如图7.15所示。

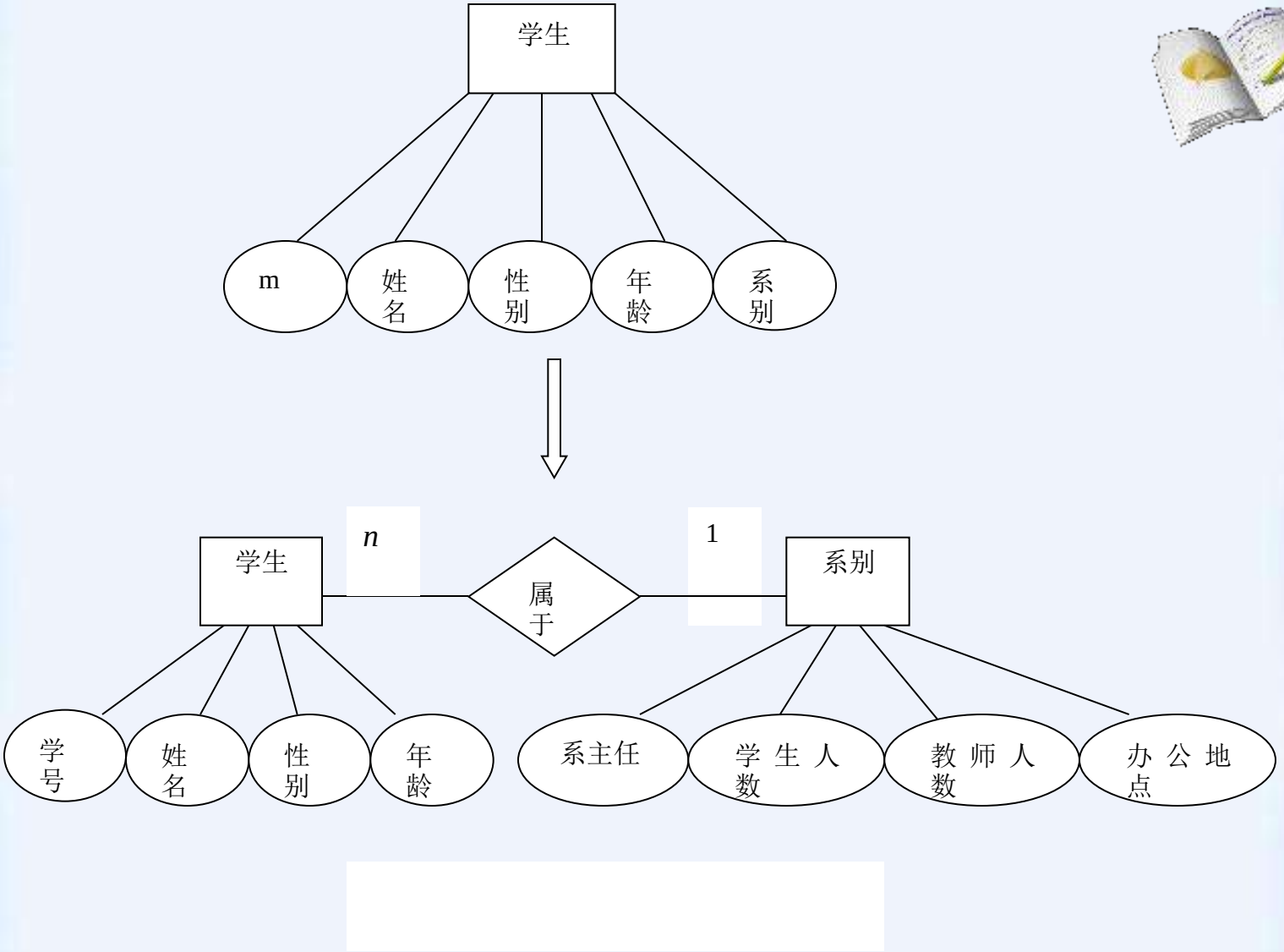


图 7 .14　系别作为一个属性或实体

返回

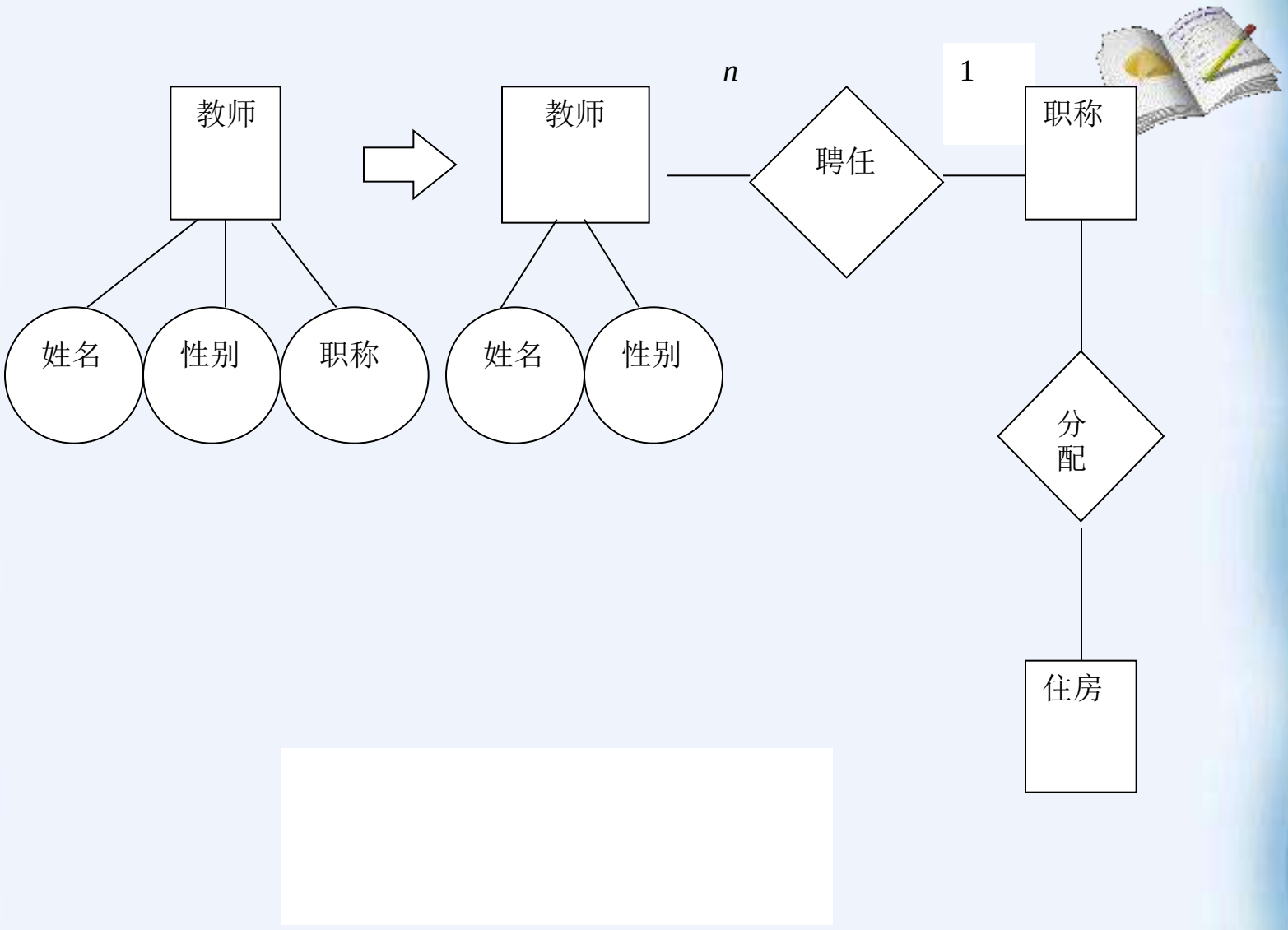


图 7 .15　职称作为一个属性或实体

- 此外，我们可能会遇到这样的情况，同一数据项，可能由于环境和要求的不同，有时作为属性，有时则作为实体，此时必须根据实际情况而定。一般情况下，凡能作为属性对待的，应尽量作为属性，以简化E-R图的处理。
- 下面举例说明局部E-R模型设计。
- 在简单的教务管理系统中，有如下语义约束。

  ①一个学生可选修多门课程，一门课程可为多个学生选修，因此学生和课程是多对多的联系；

  ②一个教师可讲授多门课程，一门课程可为多个教师讲授，因此教师和课程也是多对多的联系；

  ③一个系可有多个教师，一个教师只能属于一个系，因此系和教师是一对多的联系，同样系和学生也是一对多的联系。
- 根据上述约定，可以得到如图6.16所示的学生选课局部Ｅ－Ｒ图和如图７.17所示的教师任课局部Ｅ－Ｒ图。形成局部E-R模型后，应该返回去征求用户意见，以求改进和完善，使之如实地反映现实世界。
- E-R图的优点就是易于被用户理解，便于交流。

返回

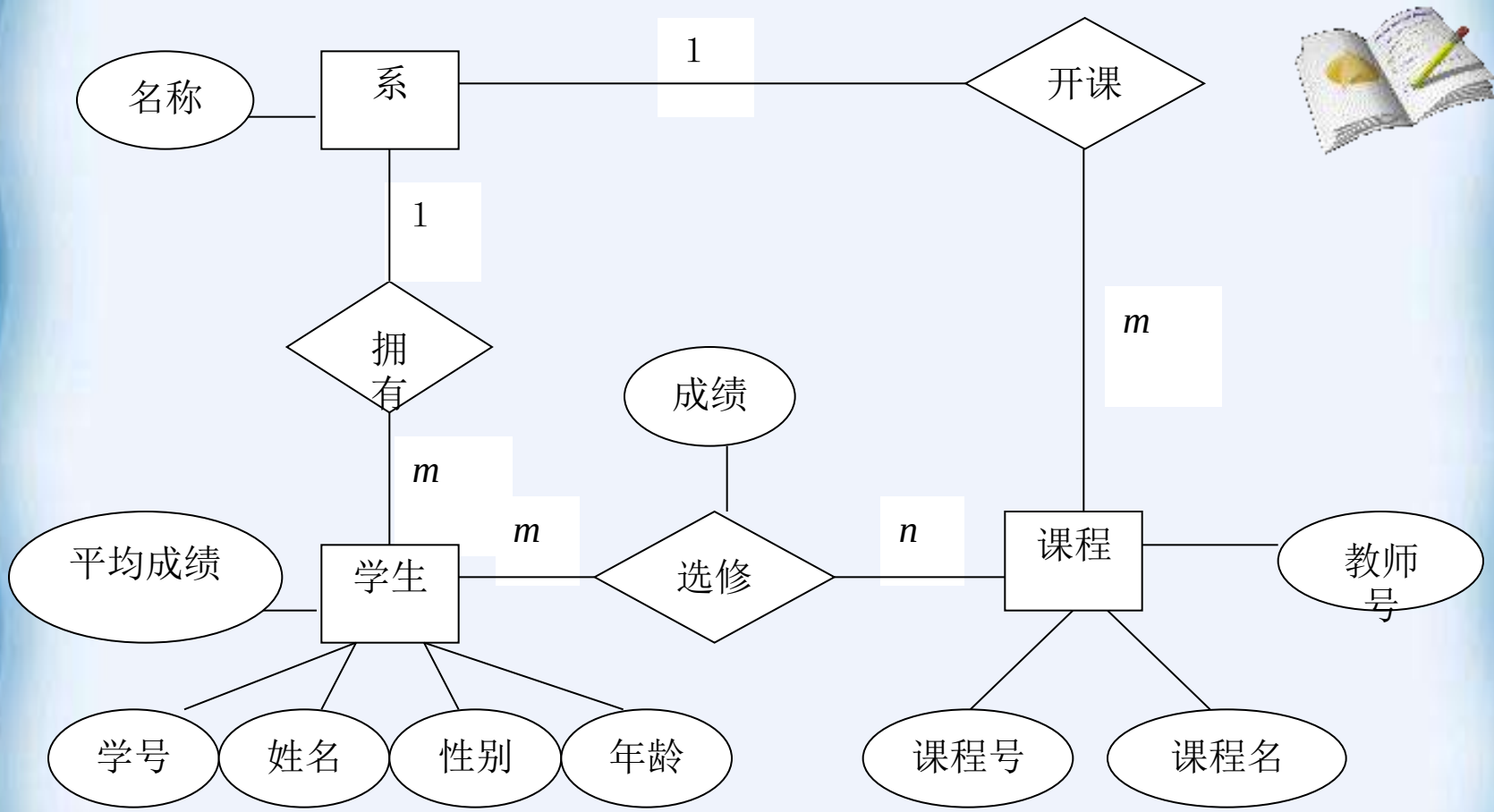


图 7 .16　学生选课局部 E — R 图

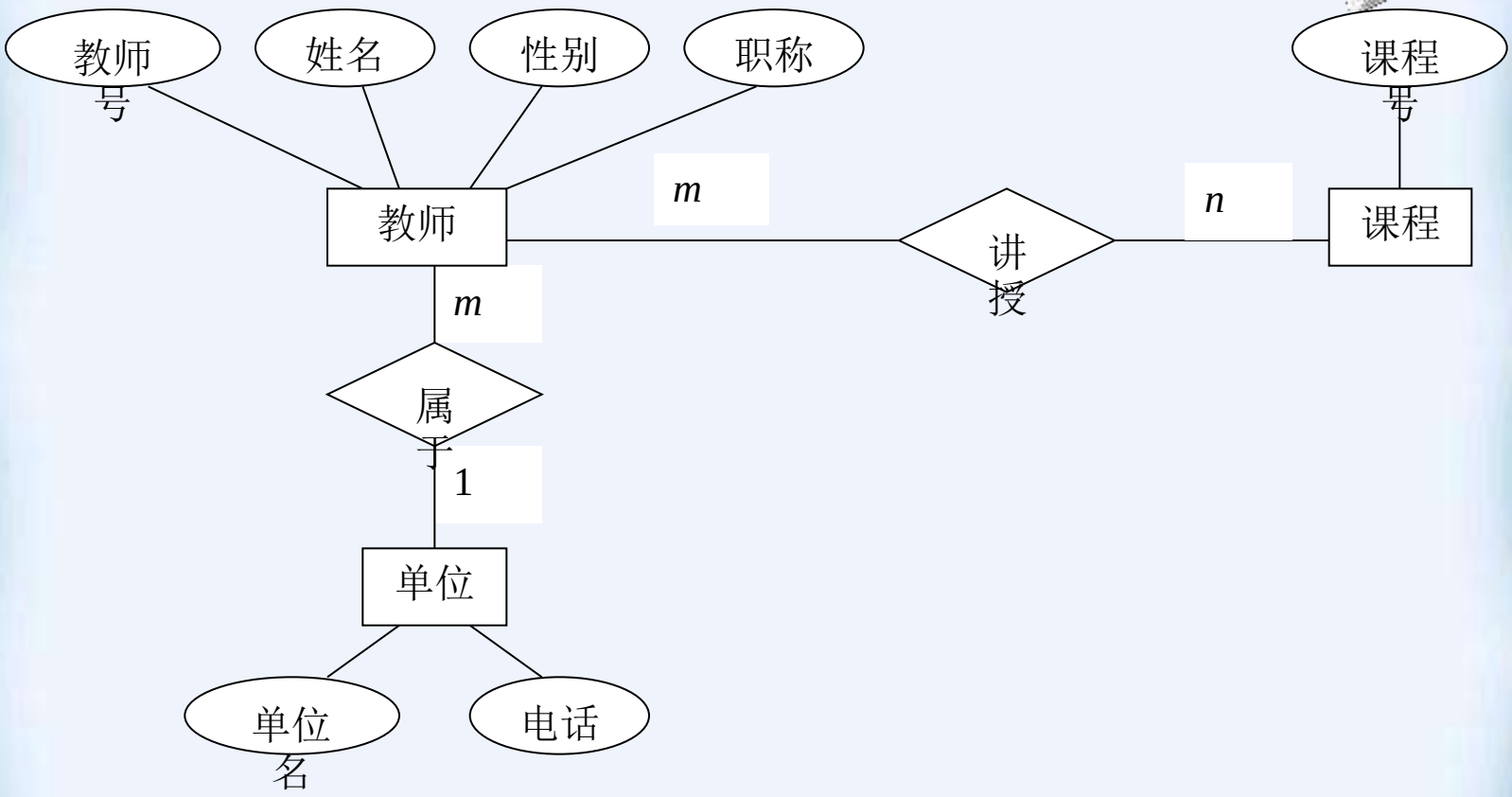


图6.17 教师任课局部E－R图

4．全局E-R模型设计

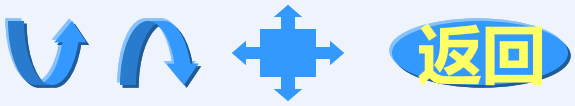
- 局部E-R模型设计完成之后，下一步就是集成各局部E-R模型，形成全局E-R模型，即视图的集成。视图集成的方法有两种：

①多元集成法，一次性将多个局部E-R图合并为一个全局E-R图，如图7.18（a）所示。

②二元集成法，首先集成两个重要的局部视图，以后用累加的方法逐步将一个新的视图集成进来，如图7.18（b）所示。在实际应用中，可以根据系统复杂性选择这两种方案。一般采用逐步集成的方法，如果局部视图比较简单，可以采用多元集成法。一般情况下，采用二元集成法，即每次只综合两个视图，这样可降低难度。无论使用哪一种方法，视图集成均分成两个步骤，如图7.19所示。

①合并，消除各局部E-R图之间的冲突，生成初步E-R图。

②优化，消除不必要的冗余，生成基本E-R图。



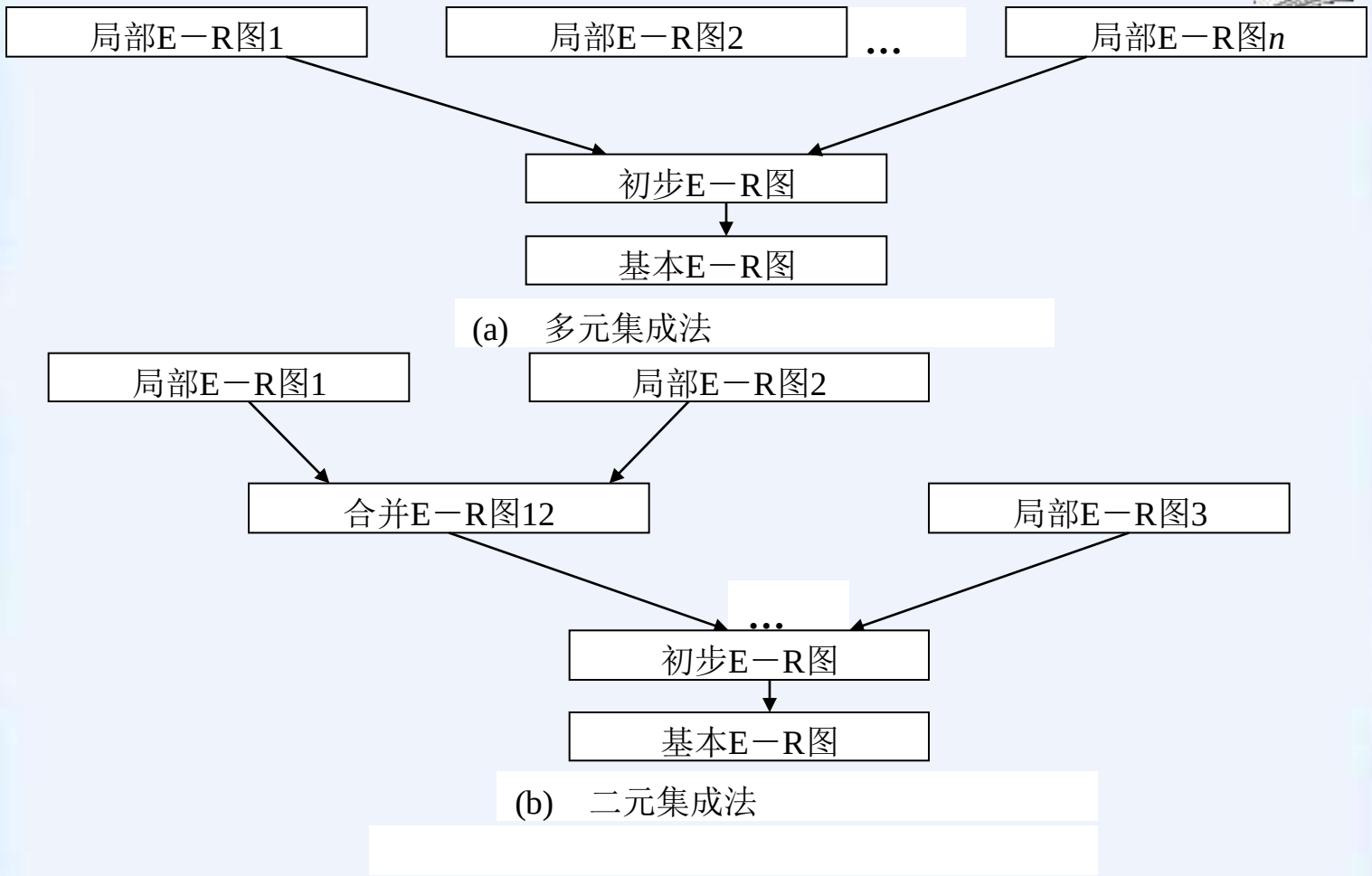


图7.18 局部视图合并成全局视图

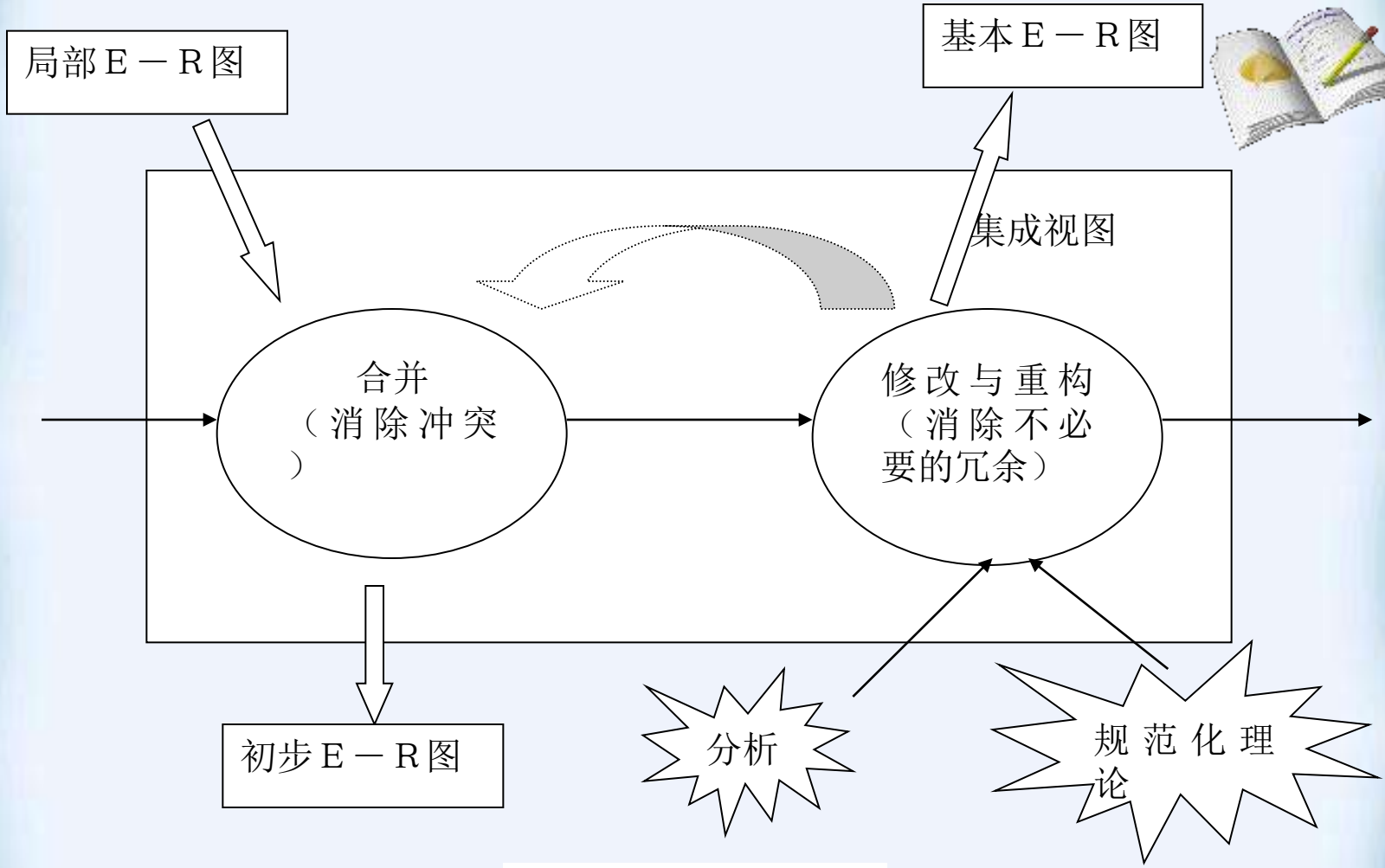


图7.19 视图集成

(1) 合并局部E-R图，生成初步E-R图

- 这个步骤将所有的局部E-R图综合成全局概念结构。
- 全局概念结构它不仅要支持所有的局部E-R模型，而且必须合理地表示一个完整、一致的数据库概念结构。
- 由于各个局部应用不同，通常由不同的设计人员进行局部E-R图设计，因此，各局部E-R图不可避免地会有许多不一致的的地方，我们称之为冲突。
- 合并局部E-R图时并不能简单地将各个E-R图画到一起，而必须消除各个局部E-R图中的不一致，使合并后的全局概念结构不仅支持所有的局部E-R模型，而且必须是一个能为全系统中所有用户共同理解和接受的完整的概念模型。
- 合并局部E-R图的关键就是合理消除各局部E-R图中的冲突。

- E-R图中的冲突有三种：属性冲突、命名冲突和结构冲突。

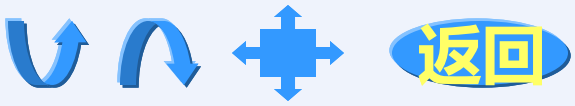
①属性冲突

- 属性冲突又分为属性值域冲突和属性的取值单位冲突。

  a. 属性值域冲突，即属性值的类型、取值范围或取值集合不同。比如学号，有些部门将其定义为数值型，而有些部门将其定义为字符型。又如年龄，有的可能用出生年月表示，有的则用整数表示。

  b. 属性的取值单位冲突。比如零件的重量，有的以公斤为单位，有的以斤为单位，有的则以克为单位。

- 属性冲突属于用户业务上的约定，必须与用户协商后解决。



②命名冲突

- 命名不一致可能发生在实体名、属性名或联系名之间，其中属性的命名冲突更为常见。
- 一般表现为同名异义或异名同义（实体、属性、联系名）。

  a. 同名异义，即同一名字的对象在不同的部门中具有不同的意义。

  比如，“单位”在某些部门表示为人员所在的部门，而在某些部门可能表示物品的重量、长度等属性。

  b. 异名同义，即同一意义的对象在不同的部门中具有不同的名称。

  比如，对于“房间”这个名称，在教务管理部门中对应着为教室，而在后勤管理部门对应为学生宿舍。

- 命名冲突的解决方法同属性冲突，需要与各部门协商、讨论后加以解决。

③结构冲突

a. 同一对象在不同应用中有不同的抽象，可能为实体，也可能为属性。例如，教师的职称在某一局部应用中被当作实体，而在另一局部应用中被当作属性。

这类冲突在解决时，就是使同一对象在不同应用中具有相同的抽象，或把实体转换为属性，或把属性转换为实体。但都要符合7.3.3中所介绍的准则。

b. 同一实体在不同应用中属性组成不同，可能是属性个数或属性次序不同。

解决办法是，合并后实体的属性组成为各局部E-R图中的同名实体属性的并集，然后再适当调整属性的次序。

c. 同一联系在不同应用中呈现不同的类型。比如E1与E2在某一应用中可能是一对一联系，而在另一应用中可能是一对多或多对多联系，也可能是在E1、E2、E3三者之间有联系。

这种情况应该根据应用的语义对实体联系的类型进行综合或调整。

- 下面以教务管理系统中的两个局部E-R图为例，来说明如何消除各局部E-R图之间的冲突，进行局部E-R模型的合并，从而生成初步E-R图。
  - 首先，这两个局部E-R图中存在着命名冲突，学生选课局部Ｅ－Ｒ图中的实体“系”与教师任课局部Ｅ－Ｒ图中的实体“单位”，都是指“系”，即所谓的异名同义，合并后统一改为“系”，这样属性“名称”和“单位”即可统一为“系名”。
  - 其次，还存在着结构冲突，实体“系”和实体“课程” 在两个不同应用中的属性组成不同，合并后这两个实体的属性组成为原来局部E-R图中的同名实体属性的并集。解决上述冲突后，合并两个局部E-R图，生成如图7.20所示的初步的全局E-R图。

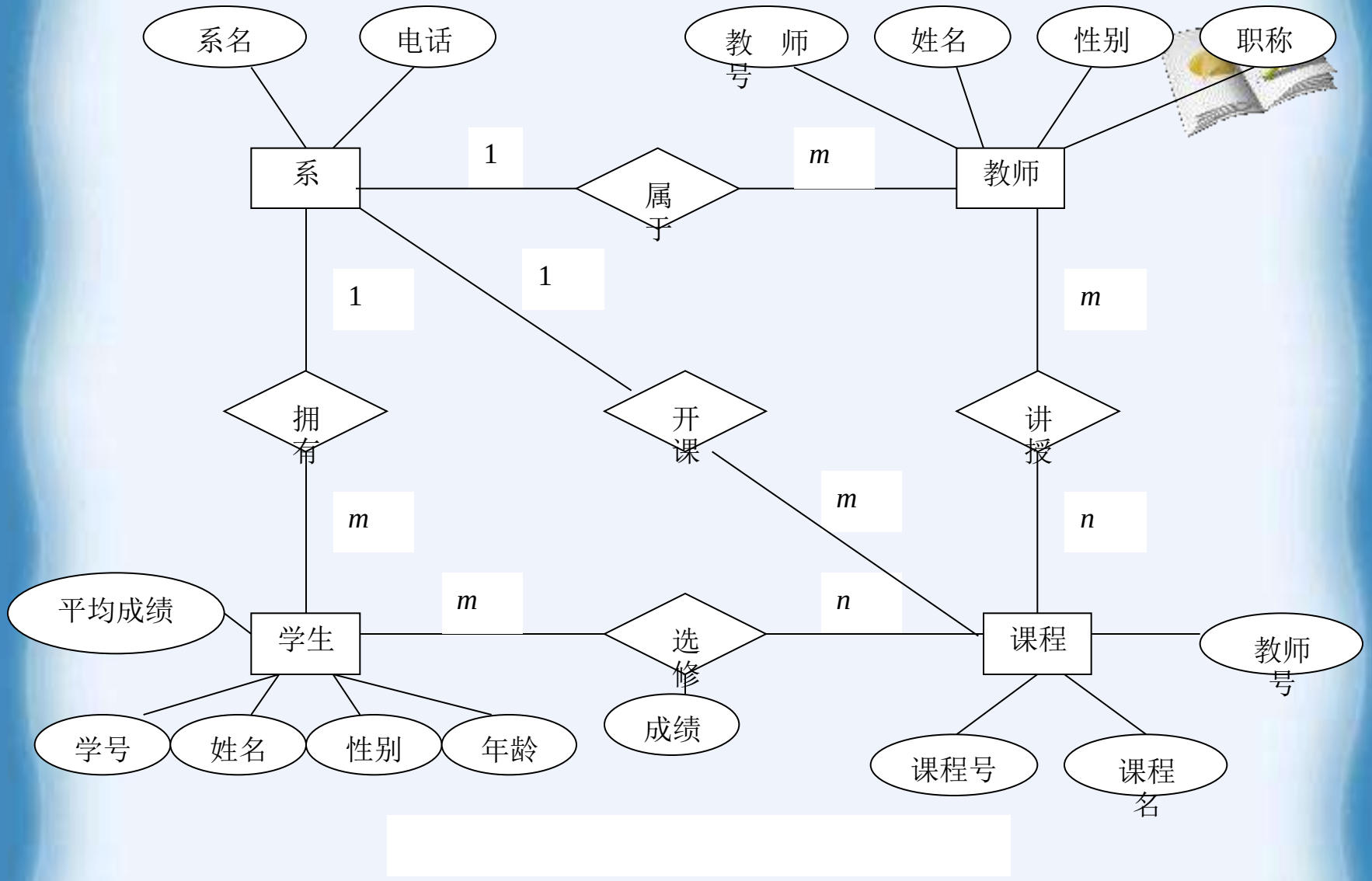


图6.20　教务管理系统的初步E－R图

(2) 消除不必要的冗余，生成基本E-R图

- 所谓冗余，在这里指冗余的数据和实体之间冗余的联系。冗余的数据是指可由基本的数据导出的数据，冗余的联系是由其他的联系导出的联系。在上面消除冲突合并后得到的初步E—R图中，可能存在冗余的数据或冗余的联系。冗余的存在容易破坏数据库的完整性，给数据库的维护增加困难，应该消除。我们把消除了冗余的初步E-R图称为基本E-R图。
- 通常采用分析的方法消除冗余。数据字典是分析冗余数据的依据，还可以通过数据流图分析出冗余的联系。
- 如在图7.20所示的初步E-R图中，“课程”实体中的属性“教师号”可由“讲授”这个教师与课程之间的联系导出，而学生的平均成绩可由“选修”联系中的属性“成绩”中计算出来，所以“课程”实体中的 “教师号” 与“学生”实体中的“平均成绩”均属于冗余数据。

- 另外，“系”和“课程”之间的联系“开课”，可以由“系”和“教师”之间的“属于”联系与“教师”和“课程”之间的“讲授”联系推导出来，所以“开课”属于冗余联系。
- 这样，图7.20的初步E-R图在消除冗余数据和冗余联系后，便可得到基本的E-R模型，如图7.21所示。
- 最终得到的基本**E-R**模型是企业的概念模型，它代表了用户的数据要求，是沟通“要求”和“设计”的桥梁。它决定数据库的总体逻辑结构，是成功建立数据库的关键。如果设计不好，就不能充分发挥数据库的功能，无法满足用户的处理要求。
- 因此，用户和数据库人员必须对这一模型反复讨论，在用户确认这一模型已正确无误的反映了他们的要求后，才能进入下一阶段的设计工作。

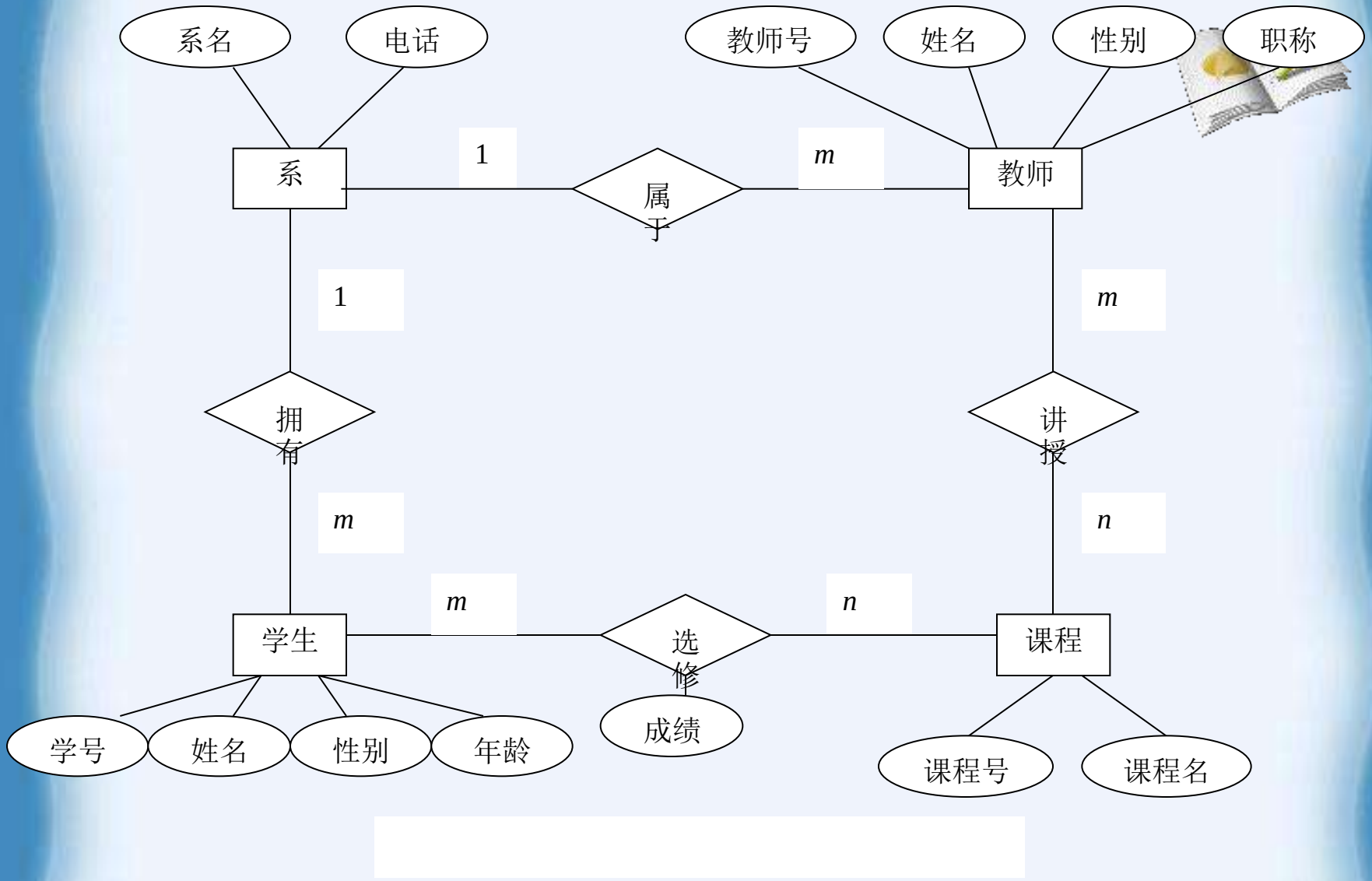


图7.21　教务管理系统的基本E－R图

6.4 逻辑结构设计

6.4.1 逻辑结构设计的任务和步骤

- 概念结构设计阶段得到的E-R模型是用户的模型，它独立于任何一种数据模型，独立于任何一个具体的DBMS。为了建立用户所要求的数据库，需要把上述概念模型转换为某个具体的DBMS所支持的数据模型。数据库逻辑设计的任务是将概念结构转换成特定DBMS所支持的数据模型的过程。从此开始便进入了“实现设计”阶段，需要考虑到具体的DBMS的性能、具体的数据模型特点。
- 从E-R图所表示的概念模型可以转换成任何一种具体的DBMS所支持的数据模型，如网状模型、层次模型和关系模型。这里只讨论关系数据库的逻辑设计问题，所以只介绍E-R图如何向关系模型进行转换。

- 一般的逻辑设计分为以下三步（如图7.22所示）：
  (1) 初始关系模式设计；
  (2) 关系模式规范化；
  (3) 模式的评价与改进。

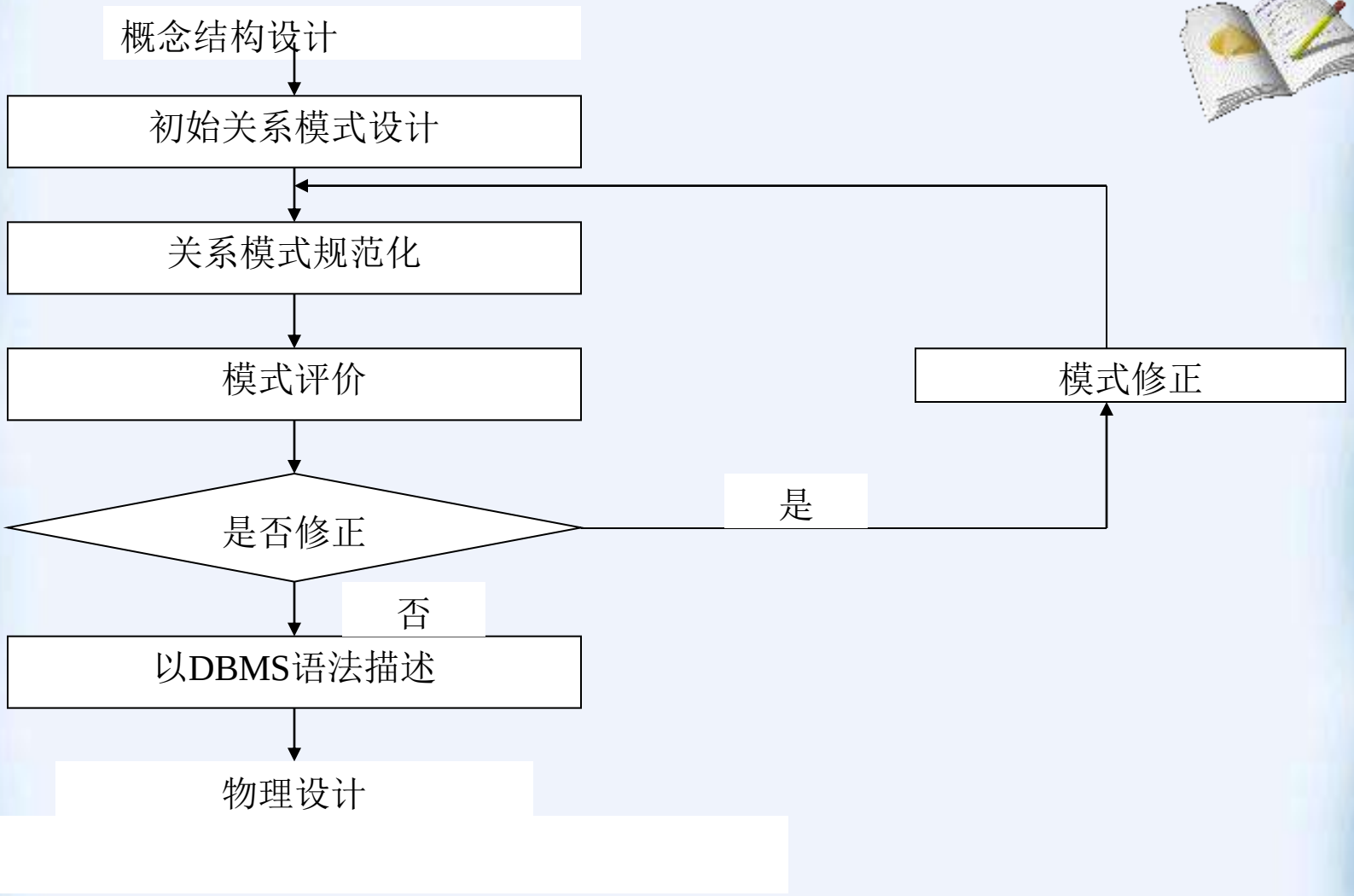


图6.22 关系数据库的逻辑设计

7.4.2 初始关系模式设计

1. 转换原则

- 概念设计中得到的E-R图是由实体、属性和联系组成的，而关系数据库逻辑设计的结果是一组关系模式的集合。所以将E-R图转换为关系模型实际上就是将实体、属性和联系转换成关系模式。在转换中要遵循以下原则：

  (1) 一个实体转换为一个关系模式，实体的属性就是关系的属性，实体的键就是关系的键。

  (2) 一个联系转换为一个关系模式，与该联系相连的各实体的键以及联系的属性均转换为该关系的属性。该关系的键有三种情况：

  ①如果联系为1:1，则每个实体的键都是关系的候选键；

  ②如果联系为1: n ，则n端实体的键是关系的键；

  ③如果联系为$n:m$，则各实体键的组合是关系的键。

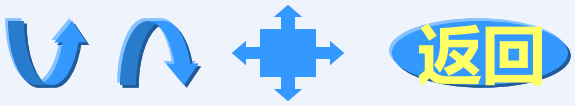
2．具体做法

(1) 把每一个实体转换为一个关系

- 首先分析各实体的属性，从中确定其主键，然后分别用关系模式表示。例如，以图7.21的E-R模型为例，四个实体分别转换成四个关系模式：
  - 学生（学号，姓名，性别，年龄）
  - 课程（课程号，课程名）
  - 教师（教师号，姓名，性别，职称）
  - 系（系名，电话）
- 其中，有下划线者表示是主键。

(2) 把每一个联系转换为关系模式

- 由联系转换得到的关系模式的属性集中，包含两个发生联系的实体中的主键以及联系本身的属性，其关系键的确定与联系的类型有关。
- 例如，还以图7.21的E-R模型为例，四个联系也分别转换成四个关系模式：
  - 属于（教师号，系名）
  - 讲授（教师号，课程号）
  - 选修（学号，课程号，成绩）
  - 拥有（系名，学号）



(3) 特殊情况的处理

- 三个或三个以上实体间的一个多元联系在转换为一个关系模式时，与该多元联系相连的各实体的主键及联系本身的属性均转换成为关系的属性，转换后所得到的关系的主键为各实体键的组合。
- 例如，图7.23表示供应商、项目和零件三个实体之间的多对多联系，如果已知三个实体的主键分别为“供应商号”，“项目号”与“零件号”，则它们之间的联系“供应”可转换为关系模式，其中供应商号，项目号，零件号为此关系的组合关系键。
  - ❖供应（供应商号，项目号，零件号，数量）

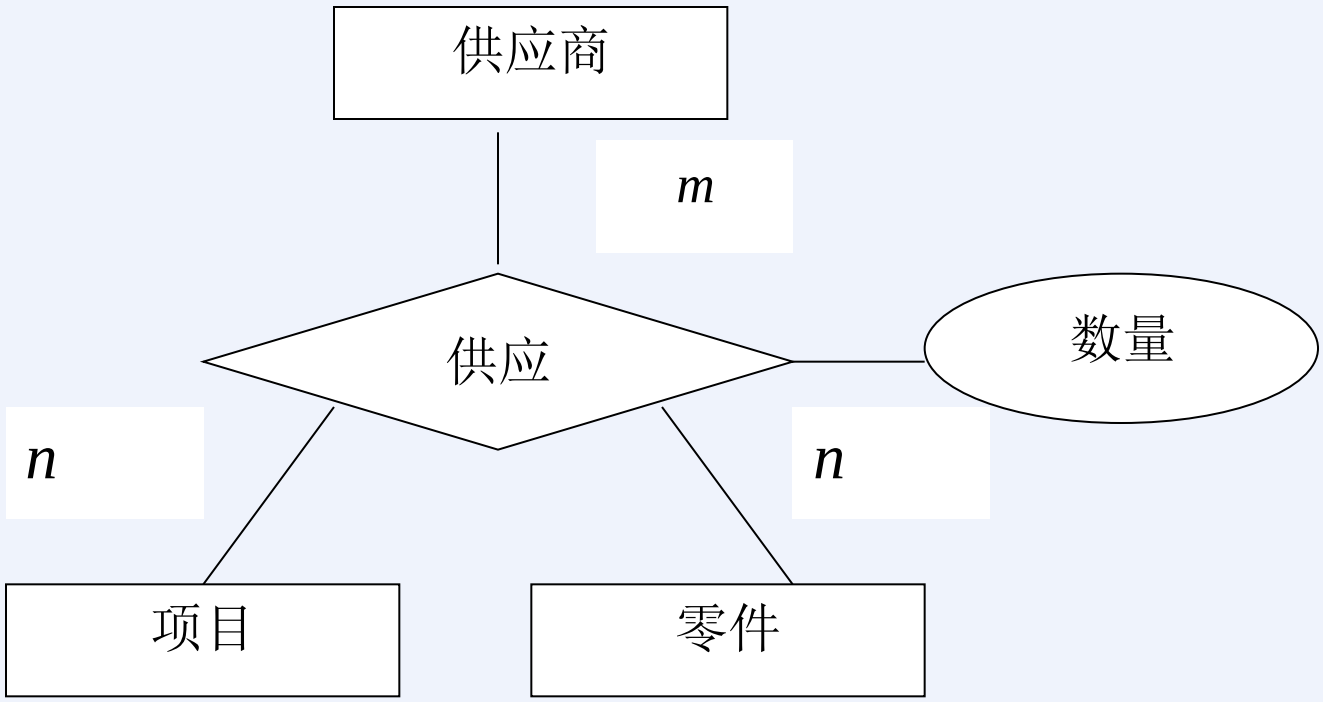


图7.23 多个实体之间的联系

7.4.3 关系模式规范化

- 应用规范化理论对上述产生的关系的逻辑模式进行初步优化，以减少乃至消除关系模式中存在的各种异常，改善完整性、一致性和存储效率。
- 规范化理论是数据库逻辑设计的指南和工具，规范化过程可分为两个步骤：确定范式级别，实施规范化处理。

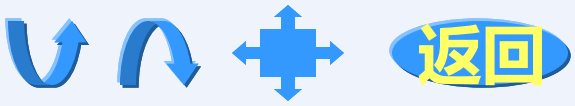
1. 确定范式级别
   - 考查关系模式的函数依赖关系，确定范式等级，逐一分析各关系模式，考查是否存在部分函数依赖，传递函数依赖等，确定它们分别属于第几范式。
2. 实施规范化处理
   - 确定范式级别后，利用第6章的规范化理论，逐一考察各个关系模式，根据应用要求，判断它们是否满足规范要求，可用已经介绍过的规范化方法和理论将关系模式规范化。

- 综合以上数据库的设计过程，规范化理论在数据库设计中有如下几方面的应用：

  (1) 在需求分析阶段，用数据依赖概念分析和表示各个数据项之间的联系。

  (2) 在概念结构设计阶段，以规范化理论为指导，确定关系键，消除初步E-R图中冗余的联系。

  (3) 在逻辑结构设计阶段，从E-R图向数据模型转换过程中，用模式合并与分解方法达到规范化级别。



7.4.4 模式评价与改进

1. 模式评价

- 关系模式的规范化不是目的而是手段，数据库设计的目的是最终满足应用需求。因此，为了进一步提高数据库应用系统的性能，还应该对规范化后产生的关系模式进行评价、改进，经过反复多次的尝试和比较，最后得到优化的关系模式。
- 模式评价的目的是检查所设计的数据库模式是否满足用户的功能要求、效率，确定加以改进的部分。模式评价包括功能评价和性能评价。
- (1) 功能评价
  - 功能评价指对照需求分析的结果，检查规范化后的关系模式集合是否支持用户所有的应用要求。关系模式必须包括用户可能访问的所有属性。在涉及多个关系模式的应用中，应确保联接后不丢失信息。如果发现有的应用不被支持，或不完全被支持，则应该改进关系模式。发生这种问题的原因可能是在逻辑设计阶段，也可能是在需求分析或概念设计阶段。是哪个阶段的问题就返回到哪个阶段去，因此有可能对前两个阶段再进行评审，解决存在的问题。

- 在功能评价的过程中，可能会发现冗余的关系模式或属性，这时应对它们加以区分，搞清楚它们是为未来发展预留的，还是某种错误造成的，比如名字混淆。如果属于错误处置，进行改正即可，而如果这种冗余来源于前两个设计阶段，则也要返回重新进行评审。

(2) 性能评价

- 对于目前得到的数据库模式，由于缺乏物理设计所提供的数量测量标准和相应的评价手段，所以性能评价是比较困难的，只能对实际性能进行估计，包括逻辑记录的存取数、传送量以及物理设计算法的模型等。
- 美国密执安大学的T.Teorey和J.Fry于1980年提出的逻辑记录访问（Logical Record Access，LRA）方法是一种常用的模式性能评价方法。LRA方法对网状模型和层次模型较为实用，对于关系模型的查询也能起一定的估算作用。
- 有关LRA方法本书不详细介绍，读者可以参考有关书籍。

2．模式改进

- 根据模式评价的结果，对已生成的模式进行改进。
  - 如果因为需求分析、概念设计的疏漏导致某些应用不能得到支持，则应该增加新的关系模式或属性。
  - 如果因为性能考虑而要求改进，则可采用合并或分解的方法。

(1) 合并

- 如果有若干个关系模式具有相同的主键，并且对这些关系模式的处理主要是查询操作，而且经常是多关系的查询，那么可对这些关系模式按照组合使用频率进行合并。
- 这样便可以减少联接操作而提高查询效率。

(2) 分解

- 为了提高数据操作的效率和存储空间的利用率，最常用和最重要的模式优化方法就是分解，根据应用的不同要求，可以对关系模式进行垂直分解和水平分解。
- 水平分解是把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。
- 对于经常进行大量数据的分类条件查询的关系，可进行水平分解，这样可以减少应用系统每次查询需要访问的记录数，从而提高了查询性能。

- 例如，有学生关系（学号，姓名，类别……），其中类别包括大专生、本科生和研究生。如果多数查询一次只涉及其中的一类学生，就应该把整个学生关系水平分割为大专生、本科生和研究生三个关系。

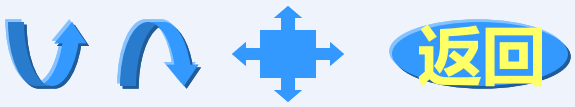
➢ 垂直分解是把关系模式的属性分解为若干子集合，形成若干子关系模式。垂直分解的原则是把经常一起使用的属性分解出来，形成一个子关系模式。

- 例如，有教师关系（教师号，姓名，性别，年龄，职称，工资，岗位津贴，住址，电话），如果经常查询的仅是前六项，而后三项很少使用，则可以将教师关系进行垂直分割，得到两个教师关系：

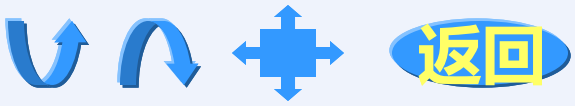
  教师关系1（教师号，姓名，性别，年龄，职称，工资）

  教师关系2（教师号，岗位津贴，住址，电话）

➢ 这样，便减少了查询的数据传递量，提高了查询速度。



- 垂直分解可以提高某些事务的效率，但也有可能使另一些事务不得不执行连接操作，从而降低了效率。因此是否要进行垂直分解要看分解后的所有事务的总效率是否得到了提高。垂直分解要保证分解后的关系具有无损连接性和函数依赖保持性。相关的分解算法已经在第4章进行了详细介绍。
- 经过多次的模式评价和模式改进之后，最终的数据库模式得以确定。逻辑设计阶段的结果是全局逻辑数据库结构。对于关系数据库系统来说，就是一组符合一定规范的关系模式组成的关系数据库模型。
- 数据库系统的数据物理独立性特点消除了由于物理存储改变而引起的对应程序的修改。标准的DBMS例行程序应适用于所有的访问，查询和更新事务的优化应当在系统软件一级上实现。这样，逻辑数据库确定之后，就可以开始进行应用程序设计了。

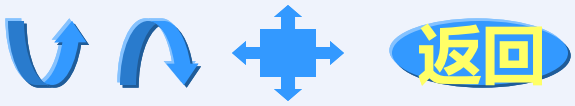


6.5 数据库物理设计

- 数据库最终要存储在物理设备上。对于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，称为数据库物理设计。物理设计的任务是为了有效地实现逻辑模式，确定所采取的存储策略。此阶段是以逻辑设计的结果作为输入，结合具体DBMS的特点与存储设备特性进行设计，选定数据库在物理设备上的存储结构和存取方法。
- 数据库的物理设计可分为两步：

  (1) 确定物理结构，在关系数据库中主要指存取方法和存储结构；

  (2)评价物理结构，评价的重点是时间和空间效率。



6.5.1 确定物理结构

- 设计人员必须深入了解给定的DBMS的功能，DBMS提供的环境和工具、硬件环境,特别是存储设备的特征。另一方面也要了解应用环境的具体要求，如各种应用的数据量、处理频率和响应时间等。只有“知己知彼”才能设计出较好的物理结构。

1．存储记录结构的设计

- 在物理结构中，数据的基本存取单位是存储记录。有了逻辑记录结构以后，就可以设计存储记录结构，一个存储记录可以和一个或多个逻辑记录相对应。存储记录结构包括记录的组成、数据项的类型和长度，以及逻辑记录到存储记录的映射。某一类型的所有存储记录的集合称为“文件”，文件的存储记录可以是定长的，也可以是变长的。
- 文件组织或文件结构是组成文件的存储记录的表示法。文件结构应该表示文件格式、逻辑次序、物理次序、访问路径、物理设备的分配。物理数据库就是指数据库中实际存储记录的格式、逻辑次序和物理次序、访问路径、物理设备的分配。
- 决定存储结构的主要因素包括存取时间、存储空间和维护代价三个方面。设计时应当根据实际情况对这三个方面进行综合权衡。一般DBMS也提供一定的灵活性可供选择，包括聚簇和索引。

(1) 聚簇（Cluster）

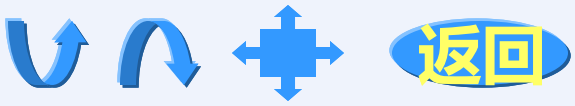
- 聚簇就是为了提高查询速度，把在一个（或一组）属性上具有相同值的元组集中地存放在一个物理块中。如果存放不下，可以存放在相邻的物理块中。其中，这个（或这组）属性称为聚簇码。
- 为什么要使用聚簇呢？聚簇有两个作用：

①使用聚簇以后，聚簇码相同的元组集中在一起了，因而聚簇值不必在每个元组中重复存储，只要在一组中存储一次即可，因此可以节省存储空间。

②聚簇功能可以大大提高按聚簇码进行查询的效率。例如，假设要查询学生关系中计算机系的学生名单，设计算机系有300名学生。在极端情况下，这些学生的记录会分布在300个不同的物理块中，这时如果要查询计算机系的学生，就需要做300次的I/O操作，这将影响系统查询的性能。如果按照系别建立聚簇，使同一个系的学生记录集中存放，则每做一次I/O操作，就可以获得多个满足查询条件和记录，从而显著地减少了访问磁盘的次数。

(2) 索引

- 存储记录是属性值的集合，主关系键可以惟一确定一个记录，而其他属性的一个具体值不能惟一确定是哪个记录。在主关系键上应该建立惟一索引，这样不但可以提高查询速度，还能避免关系键重复值的录入，确保了数据的完整性。
- 在数据库中，用户访问的最小单位是属性。如果对某些非主属性的检索很频繁，可以考虑建立这些属性的索引文件。索引文件对存储记录重新进行内部链接，从逻辑上改变了记录的存储位置，从而改变了访问数据的入口点。关系中数据越多索引的优越性也就越明显。
- 建立多个索引文件可以缩短存取时间，但是增加了索引文件所占用的存储空间以及维护的开销。因此，应该根据实际需要综合考虑。

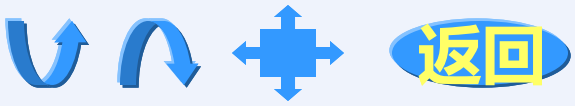


2．访问方法的设计

- 访问方法是为存储在物理设备（通常指辅存）上的数据提供存储和检索能力的方法。一个访问方法包括存储结构和检索机构两个部分。存储结构限定了可能访问的路径和存储记录；检索机构定义了每个应用的访问路径，但不涉及存储结构的设计和设备分配。
- 存储记录是属性的集合，属性是数据项类型，可用作主键或辅助键。主键惟一地确定了一个记录。辅助键是用作记录索引的属性，可能并不惟一确定某一个记录。
- 访问路径的设计分成主访问路径与辅访问路径的设计。主访问路径与初始记录的装入有关，通常是用主键来检索的。首先利用这种方法设计各个文件，使其能最有效地处理主要的应用。一个物理数据库很可能有几套主访问路径。辅访问路径是通过辅助键的索引对存储记录重新进行内部链接，从而改变访问数据的入口点。用辅助索引可以缩短访问时间，但增加了辅存空间和索引维护的开销。设计者应根据具体情况作出权衡。

3．数据存放位置的设计

- 为了提高系统性能，应该根据应用情况将数据的易变部分、稳定部分、经常存取部分和存取频率较低部分分开存放。
- 例如，目前许多计算机都有多个磁盘，因此可以将表和索引分别存放在不同的磁盘上，在查询时，由于两个磁盘驱动器并行工作，可以提高物理读写的速度。
- 在多用户环境下，可能将日志文件和数据库对象（表、索引等）放在不同的磁盘上，以加快存取速度。另外，数据库的数据备份、日志文件备份等，只在数据库发生故障进行恢复时才使用，而且数据量很大，可以存放在磁带上，以改进整个系统的性能。



4. 系统配置的设计

- DBMS产品一般都提供了一些系统配置变量、存储分配参数，供设计人员和DBA对数据库进行物理优化。系统为这些变量设定了初始值，但是这些值不一定适合每一种应用环境，在物理设计阶段，要根据实际情况重新对这些变量赋值，以满足新的要求。
- 系统配置变量和参数很多，例如，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、缓冲区分配参数（使用的缓冲区长度、个数）、存储分配参数、数据库的大小、时间片的大小、锁的数目等，这些参数值影响存取时间和存储空间的分配，在物理设计时要根据应用环境确定这些参数值，以使系统的性能达到最优。

6.5.2 评价物理结构

- 和前面几个设计阶段一样，在确定了数据库的物理结构之后，要进行评价，重点是时间和空间的效率。
- 如果评价结果满足设计要求，则可进行数据库实施。
- 实际上，往往需要经过反复测试才能优化物理设计。

6.6 数据库实施

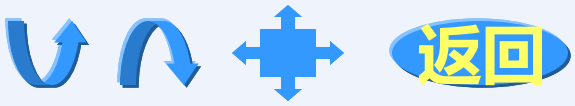
- 数据库实施是指根据逻辑设计和物理设计的结果，在计算机上建立起实际的数据库结构、装入数据、进行测试和试运行的过程。
- 数据库实施主要包括以下工作：
  - 建立实际数据库结构；
  - 装入数据；
  - 应用程序编码与调试；
  - 数据库试运行；
  - 整理文档。

6.6.1 建立实际数据库结构

- DBMS提供的数据定义语言（DDL）可以定义数据库结构。可使用第3章所讲的SQL定义语句中的CREATE TABLE语句定义所需的基本表，使用CREATE VIEW语句定义视图。

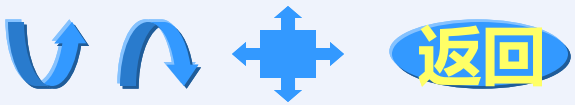
6.6.2 装入数据

- 装入数据又称为数据库加载（Loading），是数据库实施阶段的主要工作。在数据库结构建立好之后，就可以向数据库中加载数据了。
- 由于数据库的数据量一般都很大，它们分散于一个企业（或组织）中各个部门的数据文件、报表或多种形式的单据中，它们存在着大量的重复，并且其格式和结构一般都不符合数据库的要求，必须把这些数据收集起来加以整理，去掉冗余并转换成数据库所规定的格式，这样处理之后才能装入数据库。因此，需要耗费大量的人力、物力，是一种非常单调乏味而又意义重大的工作。



- 由于应用环境和数据来源的差异，所以不可能存在普遍通用的转换规则，现有的DBMS并不提供通用的数据转换软件来完成这一工作。
- 对于一般的小型系统，装入数据量较少，可以采用人工方法来完成。
  - 首先将需要装入的数据从各个部门的数据文件中筛选出来，转换成符合数据库要求的数据格式，
  - 然后输入到计算机中，
  - 最后进行数据校验，检查输入的数据是否有误。
- 但是，人工方法不仅效率低，而且容易产生差错。对于数据量较大的系统，应该由计算机来完成这一工作。通常是设计一个数据输入子系统，其主要功能是从大量的原始数据文件中筛选、分类、综合和转换数据库所需的数据，把它们加工成数据库所要求的结构形式，最后装入数据库中，同时还要采用多种检验技术检查输入数据的正确性。
- 为了保证装入数据库中数据的正确无误，必须高度重视数据的校验工作。在输入子系统的设计中应该考虑多种数据检验技术，在数据转换过程中应使用不同的方法进行多次检验，确认正确后方可入库。

- 如果在数据库设计时，原来的数据库系统仍在使用，则数据的转换工作是将原来老系统中的数据转换成新系统中的数据结构。同时还要转换原来的应用程序，使之能在新系统下有效地运行。
- 数据的转换、分类和综合常常需要多次才能完成，因而输入子系统的设计和实施是很复杂的，需要编写许多应用程序，由于这一工作需要耗费较多的时间，为了保证数据能够及时入库，应该在数据库物理设计的同时编制数据输入子系统，而不能等物理设计完成后才开始。



6.6.3 应用程序编码与调试

- 数据库应用程序的设计属于一般的程序设计范畴，但数据库应用程序有自己的一些特点。例如，大量使用屏幕显示控制语句、形式多样的输出报表、重视数据的有效性和完整性检查、有灵活的交互功能。
- 为了加快应用系统的开发速度，一般选择第四代语言开发环境，利用自动生成技术和软件复用技术，在程序设计编写中往往采用工具（CASE）软件来帮助编写程序和文档，如目前普遍使用的PowerBuilder、Delphi以及由北京航空航天大学研制的863/CMIS支持的数据库开发工具OpenTools等。
- 数据库结构建立好之后，就可以开始编制与调试数据库的应用程序，这时由于数据入库尚未完成，调试程序时可以先使用模拟数据。

6.6.4 数据库试运行

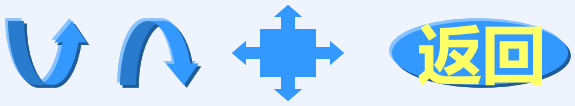
- 应用程序编写完成，并有了一小部分数据装入后，应该按照系统支持的各种应用分别试验应用程序在数据库上的操作情况，这就是数据库的试运行阶段，或者称为联合调试阶段。在这一阶段要完成两方面的工作。

  (1) 功能测试。实际运行应用程序，测试它们能否完成各种预定的功能。

  (2) 性能测试。测量系统的性能指标，分析系统是否符合设计目标。

- 系统的试运行对于系统设计的性能检验和评价是很重要的，因为有些参数的最佳值只有在试运行后才能找到。如果测试的结果不符合设计目标，则应返回到设计阶段，重新修改设计和编写程序，有时甚至需要返回到逻辑设计阶段，调整逻辑结构。

- 重新设计物理结构甚至逻辑结构，会导致数据重新入库。由于数据装入的工作量很大，所以可分期分批的组织数据装入，先输入小批量数据做调试用，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。
- 数据库的实施和调试不是几天就能完成的，需要有一定的时间。在此期间由于系统还不稳定，随时可能发生硬件或软件故障，加之数据库刚刚建立，操作人员对系统还不熟悉，对其规律缺乏了解，容易发生操作错误，这些故障和错误很可能破坏数据库中的数据，这种破坏很可能在数据库中引起连锁反应，破坏整个数据库。
- 因此必须做好数据库的转储和恢复工作，要求设计人员熟悉DBMS的转储和恢复功能，并根据调试方式和特点首先加以实施，尽量减少对数据库的破坏，并简化故障恢复。



6.6.5 整理文档

- 在程序的编码调试和试运行中，应该将发现的问题和解决方法记录下来，将它们整理存档作为资料，供以后正式运行和改进时参考。
- 全部的调试工作完成之后，应该编写应用系统的技术说明书和使用说明书，在正式运行时随系统一起交给用户。
- 完整的文件资料是应用系统的重要组成部分，但这一点常被忽视。必须强调这一工作的重要性，引起用户与设计人员的充分注意。

6.7 数据库运行和维护

- 数据库试运行结果符合设计目标后，数据库就投入正式运行，进入运行和维护阶段。数据库系统投入正式运行，标志着数据库应用开发工作的基本结束，但并不意味着设计过程已经结束。
- 由于应用环境不断发生变化，用户的需求和处理方法不断发展，数据库在运行过程中的存储结构也会不断变化，从而必须修改和扩充相应的应用程序。
- 数据库运行和维护阶段的主要任务包括以下三项内容：

  (1) 维护数据库的安全性与完整性；

  (2) 监测并改善数据库性能；

  (3) 重新组织和构造数据库。

6.7.1 维护数据库的安全性与完整性

- 按照设计阶段提供的安全规范和故障恢复规范，DBA要经常检查系统的安全是否受到侵犯，根据用户的实际需要授予用户不同的操作权限。
- 数据库在运行过程中，由于应用环境发生变化，对安全性的要求可能发生变化，DBA要根据实际情况及时调整相应的授权和密码，以保证数据库的安全性。
- 同样数据库的完整性约束条件也可能会随应用环境的改变而改变，这时DBA也要对其进行调整，以满足用户的要求。
- 另外，为了确保系统在发生故障时，能够及时地进行恢复，DBA要针对不同的应用要求定制不同的转储计划，定期对数据库和日志文件进行备份，以使数据库在发生故障后恢复到某种一致性状态，保证数据库的完整性。

6.7.2 监测并改善数据库性能

- 目前许多DBMS产品都提供了监测系统性能参数的工具，DBA可以利用系统提供的这些工具，经常对数据库的存储空间状况及响应时间进行分析评价；
- 结合用户的反应情况确定改进措施；及时改正运行中发现的错误；
- 按用户的要求对数据库的现有功能进行适当的扩充。
- 但要注意在增加新功能时应保证原有功能和性能不受损害。

6.7.3 重新组织和构造数据库

- 数据库建立后，除了数据本身是动态变化以外，随着应用环境的变化，数据库本身也必须变化以适应应用要求。
- 数据库运行一段时间后，由于记录的不断增加、删除和修改，会改变数据库的物理存储结构，使数据库的物理特性受到破坏，从而降低数据库存储空间的利用率和数据的存取效率，使数据库的性能下降。因此，需要对数据库进行重新组织，即重新安排数据的存储位置，回收垃圾，减少指针链，改进数据库的响应时间和空间利用率，提高系统性能。这与操作系统对“磁盘碎片”的处理的概念相类似。
- 数据库的重组只是使数据库的物理存储结构发生变化，而数据库的逻辑结构不变，所以根据数据库的三级模式，可以知道数据库重组对系统功能没有影响，只是为了提高系统的性能。

- 数据库应用环境的变化可能导致数据库的逻辑结构发生变化，比如要增加新的实体，增加某些实体的属性，这样实体之间的联系发生了变化，这样使原有的数据库设计不能满足新的要求，必须对原来的数据库重新构造，适当调整数据库的模式和内模式，比如要增加新的数据项，增加或删除索引，修改完整性约束条件等。
- DBMS一般都提供了重新组织和构造数据库的应用程序，以帮助DBA完成数据库的重组和重构工作。
- 只要数据库系统在运行，就需要不断地进行修改、调整和维护。一旦应用变化太大，数据库重新组织也无济于事，这就表明数据库应用系统的生命周期结束，应该建立新系统，重新设计数据库。从头开始数据库设计工作，标志着一个新的数据库应用系统生命周期的开始。

小　结

- 本章介绍了数据库设计的六个阶段，包括：系统需求分析、概念结构设计、逻辑结构设计、物理设计、数据库实施、数据库运行与维护。对于每一阶段，都分别详细讨论了其相应的任务、方法和步骤。
- 需求分析是整个设计过程的基础，需求分析做得不好，可能会导致整个数据库设计返工重做。
- 将需求分析所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所在，这一过程包括设计局部E-R图、综合成初步E-R图、E-R图的优化。

小　结

- 将独立于DBMS的概念模型转化为相应的数据模型，这是逻辑结构设计所要完成的任务。一般的逻辑设计分为三步：初始关系模式设计，关系模式规范化，模式的评价与改进。
- 物理设计就是为给定的逻辑模型选取一个适合应用环境的物理结构，物理设计包括确定物理结构和评价物理结构两步。
- 根据逻辑设计和物理设计的结果，在计算机上建立起实际的数据库结构，装入数据，进行应用程序的设计，并试运行整个数据库系统，这是数据库实施阶段的任务。
- 数据库设计的最后阶段是数据库的运行与维护，包括维护数据库的安全性与完整性，监测并改善数据库性能，必要时需要进行数据库的重新组织和构造。