第 3 章 数据库设计

在数据库领域内，通常把使用数据库的各类信息系统称为数据库应用系统。例如，以数据库为基础的各种管理信息系统、办公自动化系统、地理信息系统、电子政务系统、电子商务系统等都可以称为数据库应用系统。

数据库应用系统设计是指创建一个性能良好的、能满足不同用户使用要求的、又能被选定的 DBMS 所接受的数据库以及基于该数据库上的应用程序，而其中的核心问题是数据库的设计。

3.1 数据库设计概述

数据库设计是指对于给定的应用环境，在关系数据库理论指导下构造（设计）出最优的数据库逻辑模式和物理结构，并在此基础上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。

3.1.1 数据库设计目标和方法

1．数据库设计目标

数据库设计目标是为用户和各种应用系统提供一个较好的信息基础设施和高效率运行环境。高效率的运行环境包括数据库的存取效率、数据库存储空间的利用率、数据库系统运行管理的效率等。

数据库设计的目标主要包括如下几个方面的内容：

（1）最大限度地满足用户的应用功能需求。主要是指用户可以将当前与可预知的将来应用所需要的数据及其联系，全部准确地存放在数据库中。

（2）获得良好的数据库性能。即要求数据库设计保持良好的数据特性以及对数据的高效率存取和资源的合理使用，并使建成的数据库具有良好的数据共享性、独立性、完整性及安全性等。对关系数据库而言主要有：

① 数据要达到一定的规范化程度，避免数据重复存储和异常操作。

② 保持实体之间连接的完整性，避免数据库的不一致性。

③ 满足对事务响应时间的要求。

④ 尽可能减少数据的存储量和内外存间数据的传输量。

⑤ 便于数据库的扩充和移植，使系统有更好的适应性。

（3）对现实世界模拟的精确度要高。

（4）数据库设计应充分利用和发挥现有 DBMS 的功能和性能。

（5）符合软件工程设计要求，因为应用程序设计本身就是数据库设计任务的一部分。

上述目标中的某些内容有时候是相互冲突的。通常要对数据库的存取效率、维护代价及用户需求等方面全面考虑，权衡折中，以获得更好的设计效果。

2．数据库设计方法

大型数据库设计是涉及多学科的综合性技术，也是一项庞大的工程项目。它要求从事数据库设计的专业人员具备多方面的技术和知识。主要包括：

- 计算机的基础知识。
- 软件工程的原理和方法。
- 程序设计的方法和技巧。
- 数据库的基本知识和设计技术。
- 应用领域的相关知识。

这样，才能设计出符合具体领域要求的数据库及其应用系统。

要成功、高效地设计一个结构复杂、应用环境多样的数据库系统，仅仅靠手工的方法是很难的，必须在科学的设计理论和工程方法的支持之上，采用非常规范的设计方法，否则，就很难保证数据库设计的质量。近年来，人们将软件工程的思想和方法应用于数据库设计实践中，提出了许多优秀的数据库设计方法。

（1）新奥尔良（new orleans）法。最初于 1978 年 10 月提出，其后由 S. B. Yao 和 I. R. Palmer 等人对该方法进行了改进。是目前公认的比较完整和权威的一种规范设计方法。它将数据库设计分为四个阶段：需求分析（分析用户要求）、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，如图 3.1 所示。

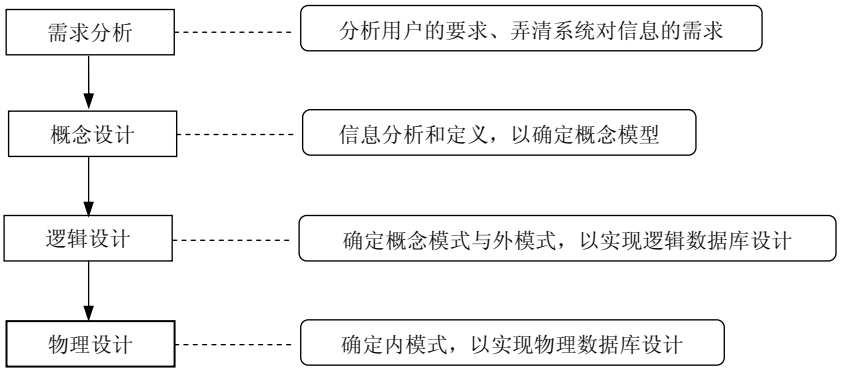


图 3.1　新奥尔良法设计过程示意图

（2）基于 E-R 模型的数据库设计方法。由 P. P. S. Chen 于 1976 年提出，该方法是数据库概念设计阶段广泛采用的方法。其基本思想是在需求分析的基础上用 E-R 图构造一个反映现实世界客观事物及其联系的概念模式。它完成了将现实世界向概念世界的转换过程。

（3）3NF 设计方法。由 S. Atre 提出的结构化设计方法，其思想是在需求分析的基础上首先确定数据库的模式、属性及属性间的依赖关系，然后将它们组织在一个单一的关系模式中，再分析模式中不符合 3NF 的约束条件，进行模式分解，最后规范成若干个 3NF 关系模式的集合。

（4）对象定义语言（Object Definition Language，ODL）方法。这是面向对象的数据库设计方法，该方法用面向对象的概念和术语来说明数据库结构。用 ODL 描述面向对象数据库结构设计，可以将其直接转换为面向对象的数据库。

利用一些数据库设计工具或自动建模软件辅助完成数据库设计的某些过程也是十分重要的方法。目前不少数据库厂商都设计和开发了一些很有特色的数据库设计工具和建模软件，如 Sybase 公司的 PowerDesign、Rational 公司的 Rose、CA 公司的 E-Rrwin 和 Bpwin 以及 Oracle 公司的 Oracle Designer 等。

扫一扫

微课视频

3.1.2 数据库设计的基本步骤

在数据库设计过程中，需求分析和概念设计可以独立于任何数据库管理系统，逻辑设计和物理设计与具体的数据库管理系统密切相关。数据库设计的步骤可以分为需求分析、概念结构设计、逻辑结构设计、物理结构设计、数据库实施、数据库运行和维护六个阶段。

1．需求分析阶段

需求分析是对用户提出的各种要求加以分析，对各种原始数据加以综合、整理，是形成最终设计目标的首要阶段，也是整个数据库设计过程中最困难的阶段。该阶段任务的完成，将为以后各阶段任务打下坚实的基础。该阶段的结果是需求分析报告，在需求分析报告中，需要列出目标系统所涉及的全部数据实体、每个数据实体的属性名一览表以及数据实体间的联系等。

2．概念结构设计阶段

概念结构设计是对用户需求进行进一步抽象、归纳，并形成独立于 DBMS 和有关软、硬件的概念数据模型的设计过程，这是对现实世界中具体数据的首次抽象，实现了从现实世界到信息世界的转化过程。概念结构设计是数据库设计的一个重要环节，通常用 E-R 模型等来描述。

3．逻辑结构设计阶段

逻辑结构设计是将概念结构转化为某个 DBMS 所支持的数据模型，并进行优化的设计过程。关系数据库的逻辑结构由一组关系模式组成。

4．物理结构设计阶段

物理结构设计是将逻辑结构设计阶段所产生的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）。

5．数据库实施阶段

数据库实施是设计人员利用所选用的 DBMS 提供的数据定义语言（DDL）来严格定义数据库，包括建立数据表、定义数据表的完整性约束等，编制与调试应用程序，组织数据入库，并进行试运行。

6．数据库运行和维护阶段

数据库试运行合格后，数据库开发工作就基本完成，可投入正式运行了。但是，由于应用环境在不断变化，数据库运行过程中物理存储也会不断变化，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。

综上所述，数据库设计流程可以用图 3.2 表示。较详细的过程将在后面各节中逐一讲解。

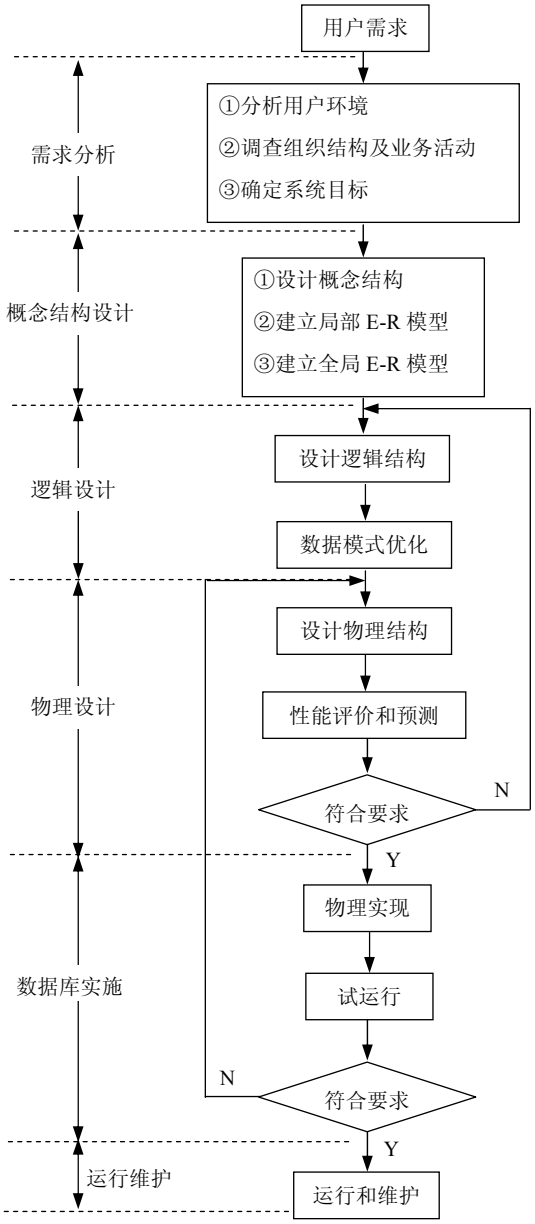


图 3.2　数据库设计流程图示

3.2　需 求 分 析

目前，数据库应用非常广泛、非常复杂，整个企业可以在同一个数据库上运行。此时，为了支持所有用户的运行，数据库设计就变得异常复杂。要是没有对数据信息进行充分的事先分析，这种设计将很难取得成功。因此，需求分析工作就被置于数据库设计过程的前沿。

3.2.1 需求分析的任务和目标

需求分析的任务是对系统的整个应用领域的情况做全面、详细的调查，确定企业组织的目标，收集支持系统总的设计目标的基础数据和对这些数据的要求，确定用户的需求，并把这些需求写成用户和数据库设计者都能够接受的文档。

需求分析的目标是给出应用领域中的数据项、数据项之间的关系和数据操作任务的详细定义，为数据库的概念设计、逻辑设计和物理设计奠定坚实的基础，为优化数据库的逻辑结构和物理结构提供可靠的依据。

设计人员还应该了解系统将来要发生的变化，收集未来应用所涉及的数据，充分考虑到系统可能的扩充和变动，使系统设计更符合未来发展的趋向，并且易于改动，以减少系统维护的代价。

这一阶段的任务如图 3.3 所示。

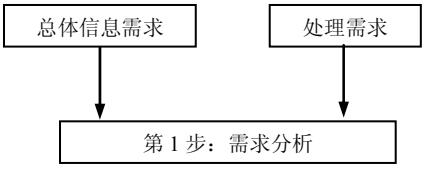


图 3.3　需求分析阶段图示

总体信息需求定义了未来系统用到的所有信息，描述了数据之间本质上和概念上的联系，描述了实体、属性及联系的性质。

处理需求定义了未来系统的数据处理的操作，描述了操作的先后次序、操作执行的频率和环境、操作与数据之间的联系。

在总体信息需求和处理需求定义说明的同时还应定义安全性和完整性约束。

这一阶段的结果是“系统需求分析报告”，其主要内容是系统的数据流图和数据字典。需求分析报告应是一份既切合实际，又具有远见的文档，是一个描述新系统的轮廓图。

3.2.2 需求分析的步骤

需求分析阶段的工作主要由下面三个步骤组成。

1．调查分析用户活动

调查未来系统所涉及用户的当前职能、业务活动及其流程等。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，包括各部门用户在业务活动中要输入什么数据，对这些数据的格式、范围有何要求。另外还需了解用户会使用什么数据，如何处理这些数据，经过处理的数据的输出内容、格式是什么。最后还应明确处理后的数据该送往何处等。其结果可以用数据流程图等图表表示出来。

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、安全性和完整性需求等，并确定哪些功能由计算机或将来由计算机完成，哪些活动由人工完成。

（1）信息需求：主要明确用户在数据库中需要存储哪些数据，以此确定各实体集以及实体集的属性，各属性的名称、别名、类型、长度、值域、数据量、实体之间的联系及联系的类型等。

（2）处理需求：指用户要对得到的数据完成什么处理功能，对处理的响应时间有何要求，处理的方式是联机处理还是批处理等。

（3）安全性和完整性需求：在定义信息需求和处理需求的同时必须确定相应的安全性和完整性约束等。

3．编写系统需求分析报告

作为需求分析阶段的一个总结，也是为了使用户和系统开发者双方对该系统的初始规定有一个共同的理解，使之成为整个开发工作的基础和依据，设计者最后要编写系统需求分析报告。作为需求分析阶段的一个总结，该报告应包括系统概况、系统的原理和技术、对原系统的改善、经费预算、工程进度、系统方案的可行性等内容。

随系统需求分析报告一起，还应提供下列附件：

（1）系统的硬件、软件支持环境的选择及规格要求（如操作系统、计算机型号及网络环境等）。

（2）组织机构图、业务流程图、各组织之间的联系图等。

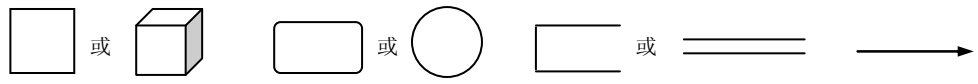
（3）数据流图、功能模块图及数据字典等图表。

系统需求分析报告一般经过设计者与用户多次讨论与修改后才能达成共识，并经过双方签字后生效，具有一定的权威性，同时也是后续各阶段工作的基础。

3.2.3 数据流图

数据流图（Data Flow Diagram，DFD）从数据传递和加工角度，以图形方式来表达系统的逻辑功能、数据在系统内部的逻辑流向和逻辑变换过程，是结构化系统分析方法的主要表达工具及用于表示软件模型的一种图示方法。

数据流图有四种基本表示符号：正方形（或正方体）表示数据的源点或终点、圆角矩形（或圆形）表示数据处理、开口矩形（或两条平行横线）表示数据存储、箭头表示数据的流向，如图 3.4 所示。



（a）数据的源点或终点　（b）数据处理　（c）数据存储　（d）数据线

图 3.4　数据流基本表示符号

一个数据处理框可以代表一系列程序、单个程序或者程序的一个模块，甚至可以代表人工处理过程；一个数据存储可以表示一个文件、文件的一部分、数据库的对象或记录的一部分，存储在磁盘等外存储器上；数据存储和数据流都是数据，数据存储是处于静止状态的数据，数据流是处于运动状态中的数据。

例 3.1　假设某学校教材科每学期需要一张征订教材报表，报表按课程编号排序，表中列出需要征订的新学期教材信息，对于每本新教材需要列出下列数据：教材编号、教材名称、征订册数、定价、出版社。教材出库或入库称为事务，书库管理员通过书库中的网络客户端把事务报告提交给教材征订系统，当某种教材的库存册数少于下学期学生需求册数时就应再次征订。

第一步从问题描述中提取源点或终点、数据处理、数据存储和数据流四种成分：由问

题描述可知“教材科”是数据终点，“书库管理员”是数据源点。接下来考虑问题描述：“教材科需要报表”，因此必须有一个用于产生报表的处理。事务的后果是改变书库中教材的存储量，然而任何改变数据的操作都是处理，因此对事务进行的加工是另一个处理。这里应注意，在问题描述中并没有明确提到需要对事务进行处理，但是通过分析可以看出这种需要。最后，考虑数据流和数据存储：系统把征订教材报表送给教材科，因此征订教材报表是一个数据流；事务需要从书库送到教材征订系统中，显然事务是另一个数据流。产生报表和处理事务这两个处理在时间上明显不匹配——每当有一个事务发生时立即处理它，然而每学期只产生一次征订教材报表。因此，用来产生征订教材报表的数据必须存放一段时间，也就是应该有一个数据存储。

注意，并不是所有数据存储和数据流都能直接从问题描述中提取出来。例如，“当某种教材的库存量少于下学期学生用书量时就应该再次征订”，这个事实意味着必须在某个地方有教材库存量和下学期学生用书量这样的数据。因为这些数据元素的存在时间应该比单个事务的存在时间长，所以认为有一个数据存储保存教材库存清单数据是合理的。

表 3.1 总结了上面分析的结果，其中加“*”号标记了问题描述中的隐含成分。

表 3.1　组成数据流图元素从描述问题信息中提取

源点/终点	数据处理	数据流	数据存储
教材科	产生报表	征订报表	教材征订信息
书库管理员	处理事务	教材编号	（见征订报表）
		教材名称	库存清单*
		征订册数	教材编号
		定价	库存量
		出版社	库存量与需求量比较
		事务	
		教材编号*	
		事务类型	
		册数*	

第二步画出数据流图的基本系统模型。计算机系统本质上都是把输入数据处理成输出数据，因此任何系统的基本模型都是由若干个数据源点/终点以及一个数据处理组成，数据处理就代表了系统对数据加工的基本功能，如图 3.5 所示。

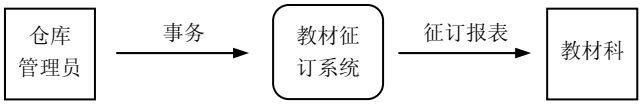


图 3.5　教材征订系统的基本系统模型

然而，从图 3.5 中对教材征订系统所能了解到的信息非常有限。

第三步把基本系统模型细化，描绘系统的主要功能。从表 3.1 可知，“产生报表”和“处理事务”是系统必须完成的两个主要功能，它们将代替图 3.5 中的“教材征订系统”，如图 3.6 所示。

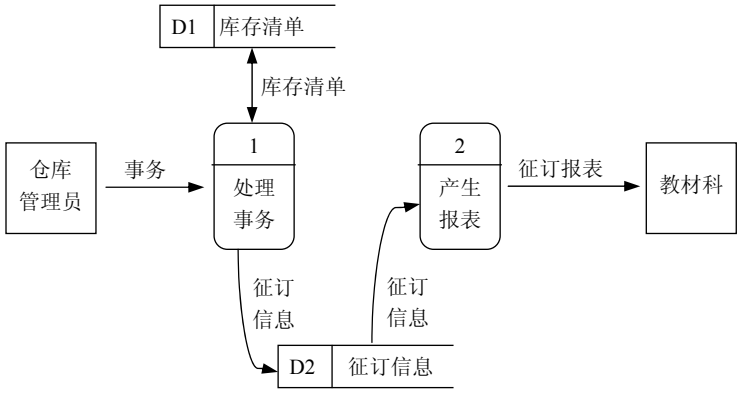


图 3.6　教材征订系统的功能级数据流图

从图 3.6 可以看出，细化后的数据流图中增加了两个数据存储：处理事务需要“库存清单”数据，产生报表和处理事务在不同时间进行，因此需要存储“征订信息”。除了表 3.1 中列出的两个数据流之外还有另外两个数据流，它们与数据存储相同。这是因为从一个数据存储中取出来的或放进去的数据通常和原来存储的数据相同，也就是说，数据存储和数据流只不过是同样数据的两种不同形式。

在图 3.6 中给处理和数据存储都加了编号，这样做的目的是便于引用和追踪。

第四步对功能级数据流图中描绘的系统主要功能进一步细化。考虑通过系统的逻辑数据流：当发生一个事务时必须首先接收它；随后按照事务的内容修改库存清单；最后如果更新后的某教材库存量少于学生需求量时，则应该再次征订，也就是需要处理教材征订信息。因此，把“处理事务”这个功能分解为下述三个步骤：“接收事务”“更新库存清单”和“处理教材征订”，这从逻辑上是合理的，如图 3.7 所示。

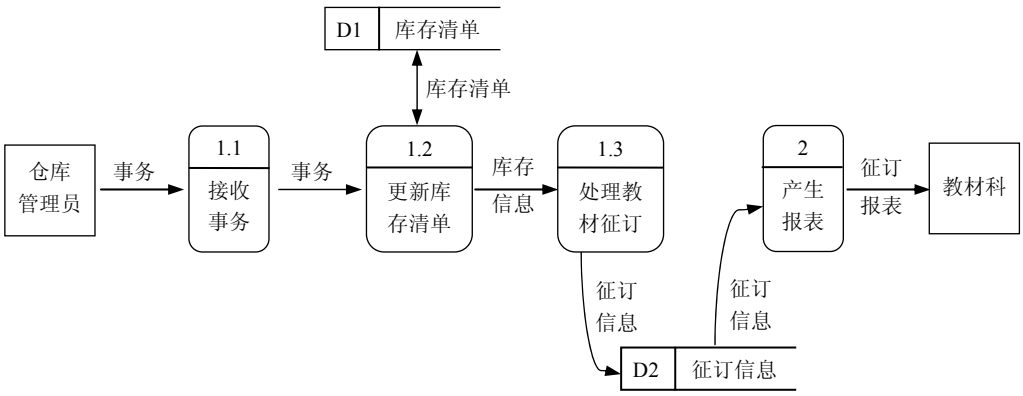


图 3.7　把处理事务的功能进一步分解后的数据流图

由于教材征订报表中需要的数据在存储的教材征订信息中全都有，产生报表只不过是按一定顺序排列这些信息，再按一定格式打印出来，因此“产生报表”这个功能不需要进一步分解。然而这些考虑纯属具体实现的细节，不应该在数据流图中表现。同理，对“接

收事务”或“更新库存清单”等功能也没有必要进一步细化。总之，若进一步分解涉及如何具体地实现一个功能时就不必再分解了。

当对数据流图分层细化时必须保持信息连续性，也就是说，当把一个处理分解为一系列处理时，分解前和分解后的输入输出数据流必须相同。例如，图 3.5 和图 3.6 的输入输出数据流都是“事务”和“征订报表”；图 3.6 中“处理事务”这个处理框的输入输出数据流是“事务”“库存清单”和“征订信息”，分解成“接收事务”“更新库存清单”和“处理教材征订”三个处理之后（见图 3.7)，它们的输入输出数据流仍然是“事务”“库存清单”和“征订信息”。

此外还应该注意在图 3.7 中对处理进行编号的方法。处理 1.1、1.2 和 1.3 是更高层次的数据流图中处理 1 的组成元素。如果处理 2 被进一步分解，它的组成元素的编号将是 2.1，2.2，…；如果把处理 1.1 进一步分解，则将得到编号为 1.1.1，1.1.2，…，的处理。

通常，用数据流程图描述一个系统时所涉及的系统结构比较复杂，这时可以进行细化和分解，形成若干层次的数据流程图，直到表达清楚为止。

3.2.4 数据字典

数据字典（Data Dictionary，DD）是指对数据流图中的数据项、数据流、数据存储和数据处理等数据元素进行定义和描述，其目的是对数据流图中各个元素做出详细的说明。没有数据字典，数据流图就不严格，然而没有数据流图，数据字典也难以发挥作用。只有将数据流图和对数据流图中每个数据元素的精确定义放在一起，才能共同构成全面的系统需求分析报告。

一般系统开发时有数据字典的存储程序，描述元素的数据字典应该包含这样的一些信息：名字、别名、描述、定义、位置。

图 3.8 给出了例 3.1 数据流图中几个数据元素的数据字典内容的含义。

名字：征订报表
别名：征订信息
描述：每学期提供一次给教材科需要征订教材的报表
定义：征订报表=教材编号+教材名称+征订册数+定价+出版社
位置：输出到打印机

（a）教材征订报表数据字典

名字：教材编号
别名：
描述：唯一地标识库存清单中一本特定教材的主键
定义：教材编号=字符型，宽度 8
位置：征订报表 征订信息 库存清单 事务

（b）教材编号数据字典

图 3.8　几个数据元素的数据字典内容的含义

名字：征订册数
别名：
描述：某课程教材一次征订的数量
定义：征订册数=整型，宽度 5
位置：征订报表
　　征订信息

（c）教材征订册数数据字典

图 3.8（续）

扫一扫

微课视频

3.3 概念结构设计

概念结构设计是将需求分析得到的用户需求抽象为反映用户观点的信息结构，它是对信息世界进行建模，是整个数据库设计的关键。

3.3.1 概念结构设计任务和 E-R 模型的特点

概念结构设计的任务是在需求分析阶段产生的系统需求分析报告基础上，按照特定方法把它们抽象为一个不依赖于计算机硬件结构和具体 DBMS 的数据模型，即概念模型。概念模型使设计者的注意力能够从对现实世界具体要求的复杂细节中解脱出来，而只集中在最重要的信息组织结构和处理模式上。

数据库设计最困难的一个方面是设计人员、编程人员以及最终用户看待数据的方式不同，这就给共同理解数据带来不便。E-R 模型是一个能够在设计人员、编程人员以及最终用户之间进行交流的模型。该模型能够描述现实世界，表达一定的语义信息且与技术实现无关，它具有以下特点：

（1）能真实、充分地反映现实世界，包括事物和事物之间的联系，并能满足用户对数据的处理要求。

（2）易于理解。可以利用它在设计人员、编程人员以及最终用户之间进行交流，使得用户能够积极参与，保证数据库设计的成功。

（3）易于更改。当应用环境和应用要求发生改变时，容易对模式进行修改和扩充。

（4）易于向关系、网状、层次等各种数据模型转换。

本节主要介绍如何利用 E-R 模型进行概念建模。

3.3.2 概念结构设计的基本方法

自从数据库技术广泛应用以来，出现了不少数据库概念结构设计的方法。尽管具体做法各异，但就其基本思想而言可归纳为以下两种：

（1）自底向上的设计方法，有时也称为属性综合法。这种方法的基本点是将前面需求分析中收集到的数据元素作为基本输入，通过对这些元素的分析，先是把它们综合成相应的实体或联系，进而构成局部概念模式，最后组合成全局概念模式，如图 3.9 所示。

自底向上的设计方法适合于较小且较为简单的系统设计，而对于中等规模以上的系统设计，数据元素常常多到几百甚至几千个。此时要对这么多的数据元素进行分析，再综合

成全局的系统设计是一件非常困难的事情。

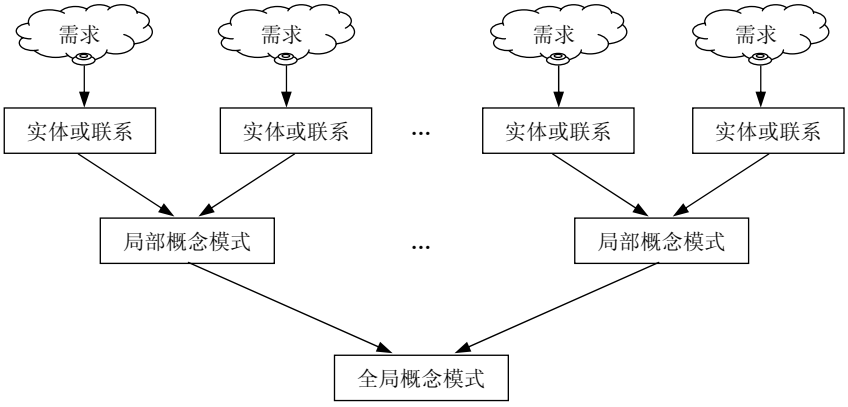


图 3.9　自底向上的设计方法

（2）自顶向下的设计方法。它是从分析组织的事务活动开始，首先识别用户所关心的实体及实体间的联系，建立一个初步的数据模型框架，然后再以逐步求精的方式加上必需的描述属性形成一个完整的局部 E-R 模型，最后再将这些局部 E-R 模型集成为一个统一的全局 E-R 模型，如图 3.10 所示。

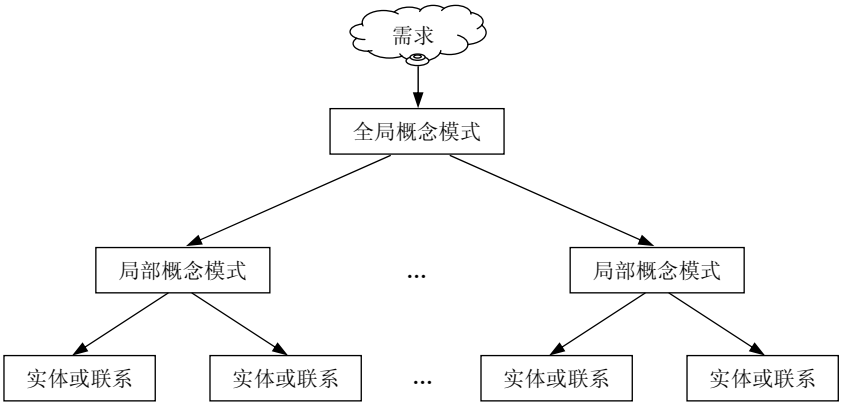


图 3.10　自顶向下的设计方法

自顶向下的设计方法是一种实体分析方法，它从总体概念入手，以实体作为基本研究对象。与自底向上的设计方法相比，其实体的个数远远少于属性的个数，因此以实体作为分析对象可以大大减少分析中所涉及的对象数，从而简化了分析过程。另外，自顶向下的设计方法通常使用图形表示法，因此更加直观、更易理解，有利于设计人员与用户的交流。

（3）混合设计方法。它是自底向上和自顶向下方法的结合。

实际项目开发中的概念设计常用的方法是自顶向下地进行需求分析，然后再自底向上地设计概念结构。

3.3.3　概念结构设计的主要步骤

概念结构设计是对用户需求综合、归纳与抽象，形成概念模型，并用 E-R 图表示。一

般可分为三个步骤来完成：进行数据抽象，设计局部概念模式；将局部概念模式综合成全局概念模式；评审。

1．进行数据抽象，设计局部概念模式

局部用户的信息需求是构造全局概念模式的基础。因此，需要先从个别用户的需求出发，为每个用户建立一个相应的局部概念结构。在建立局部概念结构时，常常要对需求分析的结果进行细化、补充和修改，如有的数据项要分为若干子项，有的数据定义要重新核实等。

2．将局部概念模式综合成全局概念模式

在综合过程中，主要处理各局部模式对各种对象定义的不一致问题，包括同名异义、异名同义和同一事物在不同模式中被抽象为不同类型的对象（例如，有的作为实体，有的又作为属性）等问题。把各个局部结构合并时，有时还会产生冗余（属性冗余、联系冗余等）问题，或导致对信息需求的再调整与分析，以确定确切的含义。消除了所有冲突后，综合各局部概念结构就可得到反映所有用户需求的全局概念结构。

3．评审

评审分为用户评审、DBA 与开发人员评审两部分。用户评审的重点放在确认全局概念模式是否准确完整地反映了用户的信息需求和现实世界事物属性间的固有联系；DBA 和开发人员评审则侧重于确认全局概念结构是否完整，各种成分划分是否合理，是否存在不一致性，以及各种文档是否齐全等。

3.3.4 局部 E-R 模型的设计

通常，一个数据库系统是为多个不同用户服务的。各个用户对数据的观点可能不一样，信息处理需求也可能不同。在设计数据库概念结构时，为了更好地模拟现实世界，一个有效的策略是“分而治之”，即先分别考虑各个用户的信息需求，形成局部概念结构，然后再综合成全局结构。在 E-R 方法中，局部概念结构设计又称为局部 E-R 模型。局部 E-R 模型的设计过程如图 3.11 所示。

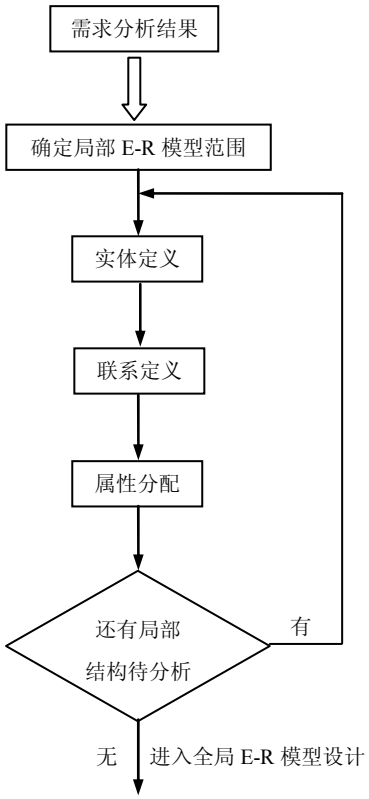


图 3.11　局部 E-R 模型设计

1．确定局部结构范围

根据需求分析阶段所产生的文档可以确定每个局部 E-R 图描述的范围。通常采用的方法是将整体的功能划分为几个系统，每个系统又分为几个子系统。设计局部 E-R 模型的第一步就是划分适当的系统或子系统，在划分时过细或过粗都不太合适。划分过细将造成大量的数据冗余和不一致，过粗有可能漏掉某些实体。一般遵循以下两条原则进行功能划分：

（1）独立性原则。划分在一个范围内的应用功能具有独立性与完整性，与其他范围内的应用有最少的联系。

（2）规模适度原则。局部 E-R 图规模应适度，一般以六个左右实体为宜。

为了说明概念结构的设计过程，本章给出某大学教学管理的简单应用实例。

例 3.2 某大学教学管理系统根据系统功能分为选课管理、学籍管理和教师开课管理三个子系统。选课管理子系统涉及学生选课，有以下语义约束：

（1）一个学院可以开设多门课程，不同学院开设的课程必须不同；

（2）一个学生可以选修多门课程，一门课程可以为多个学生选修。

学籍管理子系统涉及学院、专业、班级、学生等信息，有以下语义约束：

（1）一个学院开设有多个专业，一个专业只能属于一个学院；

（2）一个专业有多个班级，一个班级只属于一个专业；

（3）一个班级有多个学生，一个学生只属于一个班级。

教师开课管理子系统包含以下语义约束：

（1）一个部门可以有多名教师，一名教师只能属于一个部门；

（2）一个部门只有一个负责人；

（3）一名教师可以讲授多门课程，一门课程可由多名教师讲授。

2．确定实体及实体的主键

确定了局部结构设计范围后，接着进一步确定局部应用范围内的所有实体以及实体的主键。在信息系统中，实体和实体的属性通常都是指数据对象。

1）确定实体和属性

在教学管理信息系统的学生选课子系统的局部应用中，学生是一个实体，学生王丽萍、李芸的信息是学生实体中的两个实例。课程是一个实体，操作系统、数据库原理课程信息是课程实体中的两个实例。一个学生选修一门课程并参加了考试，就会有这门课程的成绩。因此，可以把成绩视为选课联系的一个属性。

学籍管理子系统的局部应用中，学院是一个实体，计算机学院、电信学院的信息就是学院实体中的两个实例。班级是一个实体，每个班级的信息是班级实体的一个实例。

教师开课管理子系统的局部应用中，教师是一个实体，每位上课教师的信息都是教师实体的一个实例。部门是一个实体，科技处、计算机学院的信息是部门实体中的两个实例。

属性具有原子性，实体是依赖属性描述的，两个实体相同或相异是指两个实体的属性集相同或相异。如学生实体的属性集（学号，姓名，性别，年龄，学院）、课程实体的属性集（课程号，课程名，课程类型，学分）、学院实体的属性集（学院号，学院名，地址，电话）等。

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”。例如，某大学中的学院，在某种应用环境中，它只是作为“学生”实体的一个属性，表明一个学生属于哪个学院；而在另一种应用环境中，由于需要考虑一个学院的院长、办公地址、联系电话等，这时学院就需要作为实体了。

在需求分析阶段，已收集了许多数据对象。概念结构设计时，需要区分这些数据对象究竟是实体还是属性，下面给出区分实体与属性的一般原则：

（1）实体一般需要描述信息，而属性不需要。例如，学生需要描述属性（学号、姓名、性别、出生年月等），所以学生是实体。而性别不需要描述信息，所以性别是属性。

（2）多值属性可考虑作为实体。例如，教师职务是一个多值属性，即一个教师可能担任多个职务。此时职务可考虑作为一个独立的实体，否则数据库表中就会出现大量空值。

为了说明这个问题，假设有一个教师基本信息表，其格式如表 3.2 所示。

表 3.2　教师基本信息表

教师号	教师姓名	性别	出生年月	工作部门	职务 1	…	职务 5	职称	…
…	…	…	…	…	…	…	…	…	…

从表 3.2 中可以看出，教师担任的职务最多可以有 5 个。因为多数教师的职务只有一个，那么其他职务项就是空值。这样不仅浪费空间，而且由于空值是一个特殊的值，它表明该值为空缺或未知。空值对数据库用户来说可能会引起混淆，应该尽量避免。因此，表 3.2 中的“职务”属性应该分离出来作为一个独立的实体，如图 3.12 所示。

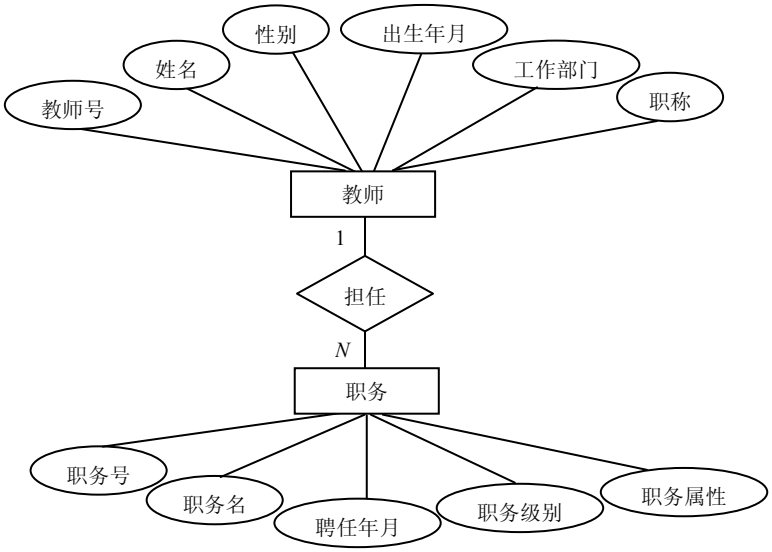


图 3.12　多值属性分离成独立实体示意图

2）确定实体的主键

学生实体的主键是学号，学院实体的主键是学院号，专业实体的主键是专业号，班级实体的主键是班级号，课程实体的主键是课程号，教师实体的主键是教师号，部门实体的主键是部门号，负责人的主键是职工号。

确定完所有的实体和实体的主键，再对实体进行归类，把具有共性的实体归为一类。例如，学校中的专科生、本科生以及研究生都是学生实体，它们之间具有共性，可以把它们归为学生一类，然后用泛化关系表示出来。

3．确定实体间的联系

联系是实体之间关系的抽象表示，即对现实世界中客观事物之间关系的描述。在局部 E-R 模型设计时，需要对已识别出的实体确定不同实体间的联系是属于什么类型的联系，是二元联系还是多元联系？然后再确定这些实体的联系方式，即一对一、一对多，还是多对多？这些问题的解决往往是根据问题的语义或者一些事务的规则确定的。本节主要讨论

常见的实体间的二元联系，它是现实世界中大量存在的联系。

在例 3.2 中，选课管理子系统的“学生”实体与“课程”实体之间存在多对多的“选修”联系，“课程”实体与“学院”实体之间存在多对一的“开设”联系；学籍管理子系统的“学院”实体与“专业”实体之间存在一对多的“开设”联系，“专业”实体与“班级”实体间存在一对多的“拥有”联系，“班级”实体与“学生”实体之间存在一对多的“含有”联系；教师开课管理子系统的“课程”实体与“教师”实体之间存在多对多的“讲授”联系，“教师”实体与“部门”实体之间存在多对一的“属于”联系，“部门”实体与“负责人”之间存在一对一的“聘任”联系。

关于实体间联系的确定，需要有几点说明：

1）联系的属性

联系本身也可以有属性，当一个属性不能归并到两个实体上时，就可以定义为联系的属性。在例 3.2 中，选课管理子系统的“学生”实体与“课程”实体之间存在“选修”联系，学生每学一门课程并参加考试，便可获得该门课程的成绩。如果把“成绩”属性放在“学生”实体中，由于一个学生的成绩属性有多个值（每门课一个成绩），所以不合适；如果把“成绩”属性放在“课程”实体中，也会因为一门课有多个学生选修而不易确定是哪个学生的成绩。因此，成绩一般作为选课联系的属性较为合适，如图 3.13 所示。

同样，在教师开课管理子系统中，教师讲授课程的“地址”可以作为“讲授”联系的属性，部门负责人的聘任“日期”可以作为“聘任”联系的属性。

2）冗余联系

冗余联系是指可以由基本联系导出的联系。图 3.14 是一个冗余联系的例子。假设每名教师可以担任多门课程的教学，一门课程可以有多名教师讲授，一个学生可以选修多门课程，一门课程有多名学生学习。由于联系具有传递性，因此，隐含了教师和学生多对多的授课联系。

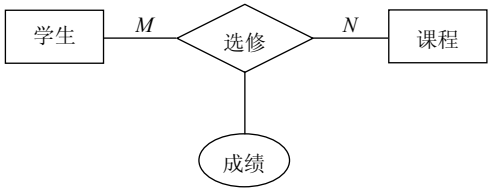


图 3.13　成绩作为选课联系的属性 E-R 图

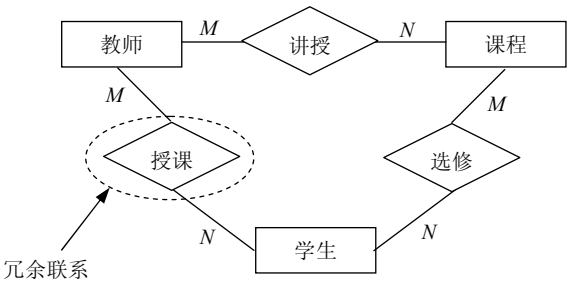


图 3.14　冗余联系的例子

冗余联系也会破坏数据库的完整性。因此，如果存在冗余联系，尽量消除它，以免将问题遗留在全局的 E-R 模式阶段。

3）实体间的多个联系

实体间可能存在多个联系。例如，教师实体和项目实体，有的教师作为项目主持人负责管理项目，而有些教师作为项目成员参与项目开发。假设一个项目只有一个项目负责人但可以由多名教师参与，一名教师可以管理或参与多个项目，如图 3.15 所示。

4．给实体及联系加上描述属性

当已经在一个局部应用中识别了实体、实体的主键以及实体间的联系时，便形成了一个局部的 E-R 图。然后，再为每个实体和联系加上所有必需的描述属性，以描述局部结构中的语义信息。如例 3.2 中的教师开课管理子系统中实体“部门”、联系“聘任”加上描述属性，如图 3.16（a）、（b）所示。

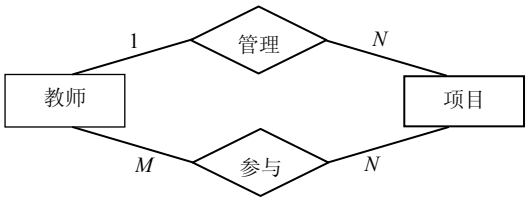
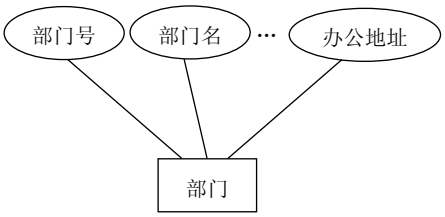
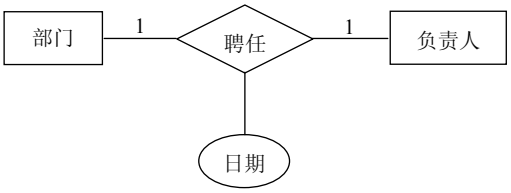


图 3.15　两个实体间存在多个联系示例



（a）实体“部门”加上描述属性



（b）联系“聘任”加上描述属性

图 3.16　给实体和联系加上描述属性

在需求分析阶段，已收集了所有的数据对象。除了主键属性外，还需将其他属性分配给有关的实体或联系。为使这种分配更合理，必须研究属性之间的函数依赖关系并考虑其他一些准则，而这些不易于一般用户理解。因此在概念结构设计阶段，应该避免涉及这类问题，而主要应从用户需求的概念上去识别实体或联系应该具有哪些描述属性。

给实体及联系添加描述属性可分为两步：一是确定属性；二是把属性分配给有关的实体和联系。

确定属性的原则如下：

（1）属性应该是不可再分解的语义单位。

（2）实体与属性之间的关系只能是 1:*N* 的。

（3）不同实体类型的属性之间应无直接关联关系。

属性分配的原则如下：

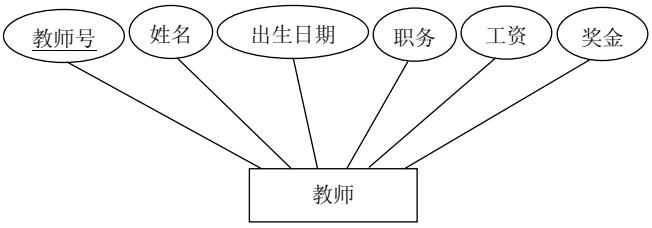
当多个实体类型用到同一属性时便会导致数据冗余，从而可能会影响存储效率和完整性约束，因此，通常的做法是把属性分配给那些使用频率最高的实体类型。

有些属性不宜归属于任一实体类型时，可以作为实体之间联系的属性。

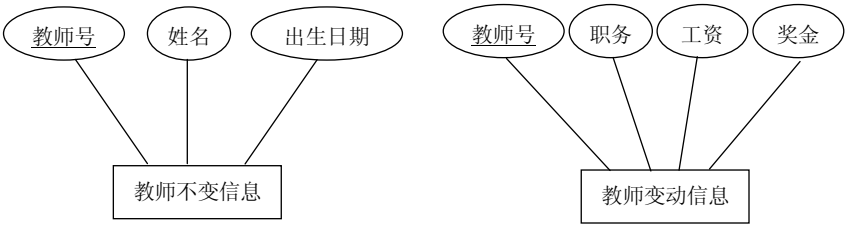
5．E-R 模型的操作

在数据库设计过程中，常常要对 E-R 图进行种种变化，这种变化称为 E-R 模型的操作。它包括实体类型、联系类型和属性的分裂、合并、增删等。

例 3.3　E-R 图分裂操作有水平分裂和垂直分裂两种。把教师分裂成男教师与女教师两个实体类型，这是水平分裂。也可把教师中经常变化的属性组成一个实体类型，而把固定不变的属性组成另一个实体类型，这是垂直分裂，如图 3.17 所示。但应注意，在 E-R 图垂直分裂中，键必须在分裂后的每个实体类型中都出现。



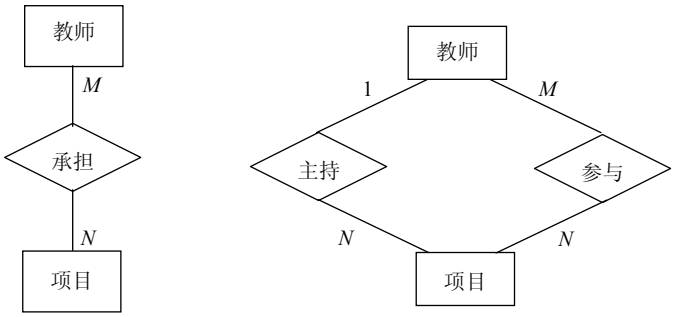
（a）分裂前的实体



（b）分裂后的两个实体

图 3.17　实体类型垂直分裂的 E-R 图

联系类型也可分裂。图 3.18（a）是教师承担项目的 E-R 图，而“承担”联系类型可以分裂为“主持”和“参与”两个新的联系类型，如图 3.18（b）所示。



（a）分裂前的联系　　（b）分裂后的联系

图 3.18　联系类型的分裂 E-R 图

合并是分裂操作的逆过程。例如，有一个“产品销售”实体，其属性有“产品号”和“销售量”，另一个“产品生产”实体，其属性有“产品号”和“产量”，把它们合并操作，如图 3.19 所示。

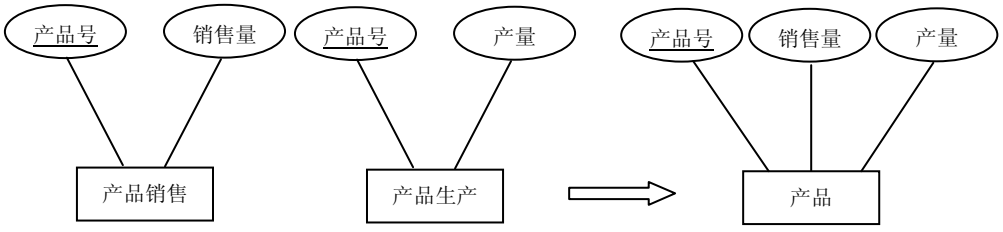
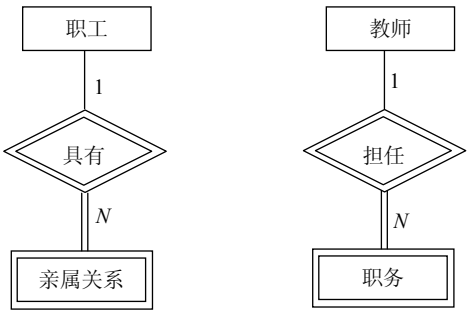


图 3.19　两个实体合并的 E-R 图

6．弱实体与弱联系

在现实世界中，有时某些实体对于另一些实体具有很强的依赖联系，例如一个实体的存在必须以另一实体的存在为前提。比如，一个职工可能有多个亲属关系，亲属关系是多值属性，为了消除冗余，设计两个实体：职工与亲属关系。在职工与亲属关系中，亲属关系的信息是以职工信息的存在为前提，所以职工与亲属关系是一种依赖联系。一个实体对于另一些实体具有很强的依赖联系，而且该实体主键的部分或全部从其依赖实体中获得，称该实体为弱实体。在 E-R 模型中，弱实体用双线矩形框表示。与弱实体的联系，称为弱联系，用双线菱形框表示。

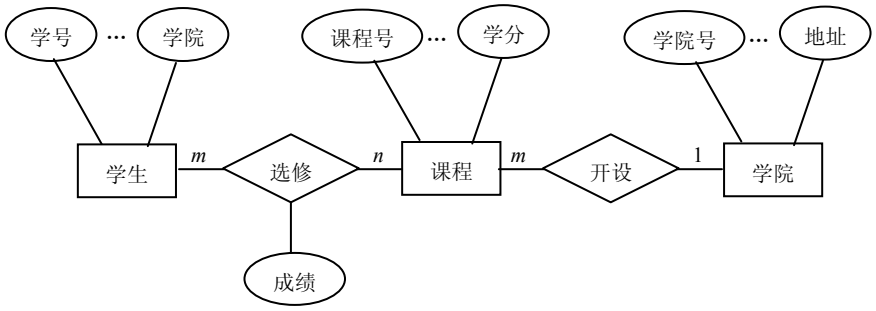


（a）亲属关系　　（b）职务关系

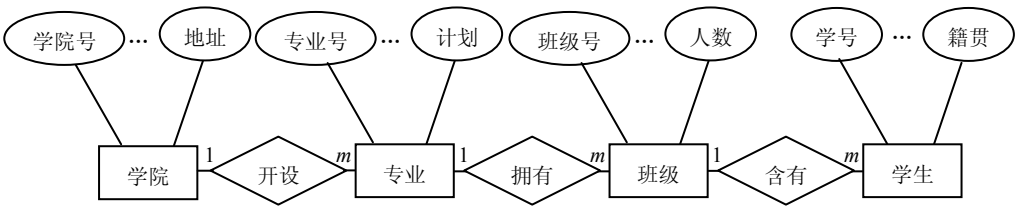
图 3.20　弱联系的表示方法

例 3.4　在人事管理系统中，亲属关系对于职工具有弱依赖联系，所以说，亲属关系是弱实体，如图 3.20（a）所示。又如图 3.12 中教师基本信息的“教师”和“职务”实体也是弱实体与弱联系的关系，如图 3.20（b）所示。

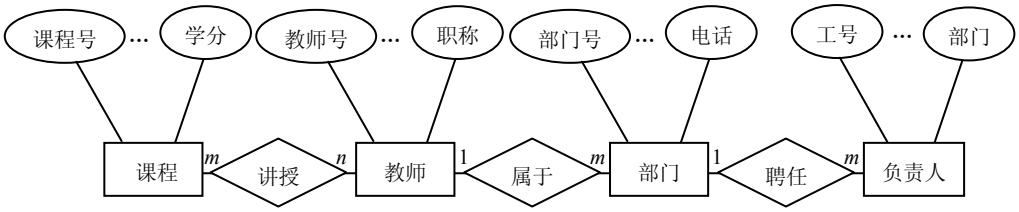
例 3.2 中选课管理、学籍管理、教师开课子系统的局部 E-R 图分别如图 3.21（a）、（b）、（c）所示。



（a）选课管理子系统局部 E-R 图



（b）学籍管理子系统局部 E-R 图



（c）教师开课子系统局部 E-R 图

图 3.21　学生管理系统的局部 E-R 图

3.3.5 全局 E-R 模型的设计

所有局部 E-R 模型都设计好后，接下来就是把它们综合成为单一的全局概念结构。全局概念结构不仅要支持所有局部 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。

全局 E-R 模型的设计过程如图 3.22 所示。

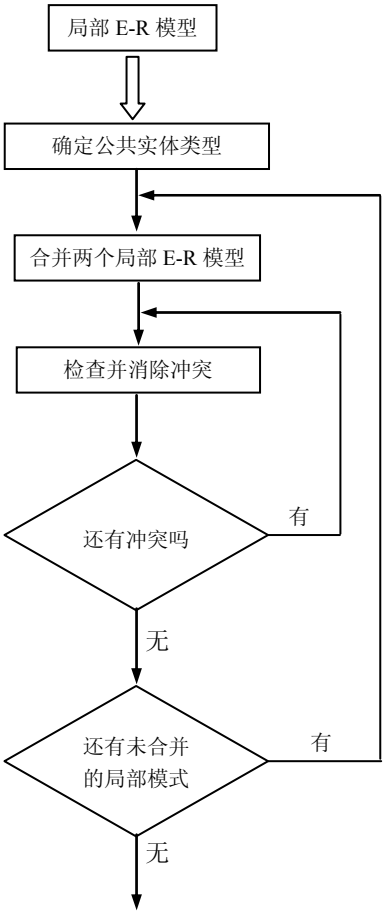


图 3.22　全局 E-R 模型设计

1．确定公共实体类型

为了给多个局部 E-R 模型的合并提供合并的基础，首先要确定各局部结构中的公共实体类型。

公共实体类型的确定并非一目了然。特别是当系统较大时，可能有很多局部模式，这些局部 E-R 模型是由不同的设计人员确定的，因此对同一现实世界的对象可能给予不同的描述。有的作为实体类型，有的又作为联系类型或属性。即使都表示成实体类型，实体类型名和键也可能不同。在这一步中，将仅根据实体类型名和键来认定公共实体类型。一般把同名实体类型作为公共实体类型的一类候选，把具有相同键的实体类型作为公共实体类型的另一类候选。

2．局部 E-R 模型的合并

合并的顺序有时会影响处理效率和结果。建议的合并原则是：首先进行两两合并；先合并那些现实世界中有联系的局部结构；合并从公共实体类型开始，最后再加入独立的局部结构。

进行两两合并是为了减少合并工作的复杂性。

3．消除冲突

由于各种类型应用的不同，不同的应用通常又由不同的设计人员设计成局部 E-R 模型，因此局部 E-R 模型之间不可避免地会有不一致的地方，一般称之为冲突。通常可以把冲突分为三种类型：

（1）属性冲突。主要表现在属性域冲突和属性取值单位冲突两种类型。

① 属性域冲突是指同一属性在不同局部 E-R 模型中有着不同的数据类型、取值范围或取值集合。如学生考查课成绩在某一局部 E-R 模型中用“优秀、良好、中等、及格、不及格”5 分制表示，而在另一个局部 E-R 模型中用百分制来表示，前者的数据类型为字符型，后者的数据类型为数值型。

② 属性取值单位冲突是指同一属性在不同局部 E-R 模型中具有不同的单位。如在卫星轨道设计时，某一局部 E-R 模型中用“角度制”表示，而在另一个局部 E-R 模型中用“弧度制”来表示。

（2）结构冲突。主要表现在如下几个方面：

① 同一对象在不同应用中的不同抽象。如选课管理子系统中“学生”实体有“学院”

属性，而学籍管理子信息系统中，“学院”又作为“实体”。一般地，对于同一对象在不同的局部 E-R 模型中产生不同的抽象，其解决方式是：把属性变为实体或把实体变为属性，使同一对象要具有相同的抽象。

例 3.5 在选课管理子系统中有实体：学生（学号，姓名，性别，出生年月，学院），学籍管理子信息系统中有实体：学院（学院号、名称、地址、电话）。其实“学生”和“学院”之间存在从属关系，应该调整、合并为如图 3.23 所示。

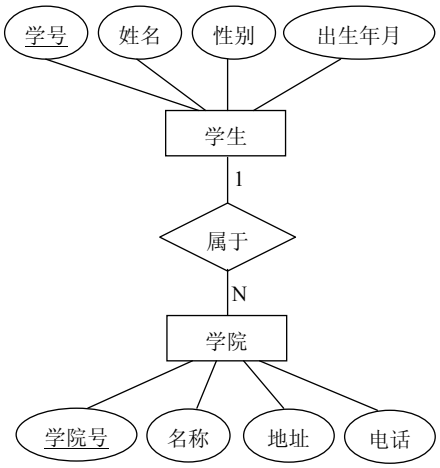
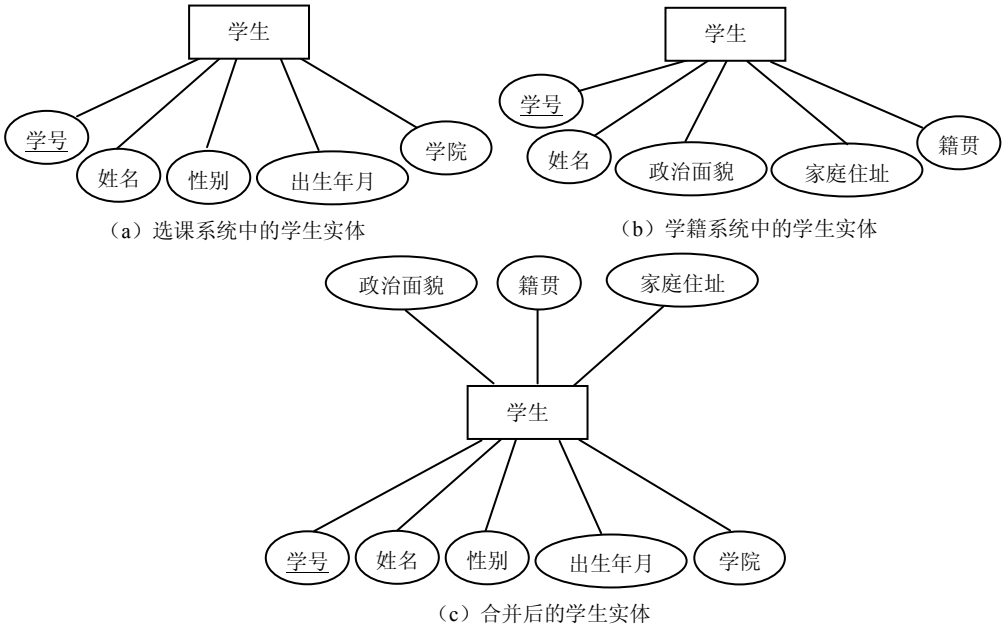


图 3.23 学院与学生实体间联系的 E-R 图

② 同一实体在不同局部 E-R 图中属性组成不完全相同，包括属性个数、次序等。解决的方式是：取其在不同局部 E-R 模型中实体属性的并集，并适当设计好属性的次序。

例 3.6 在选课管理子系统中有实体：学生（学号，姓名，性别，出生年月，学院）；在学籍管理子系统中有实体：学生（学号，姓名，政治面貌，家庭住址，籍贯），将两个学生实体进行合并如图 3.24 (a)、(b)、(c)所示。



（a）选课系统中的学生实体

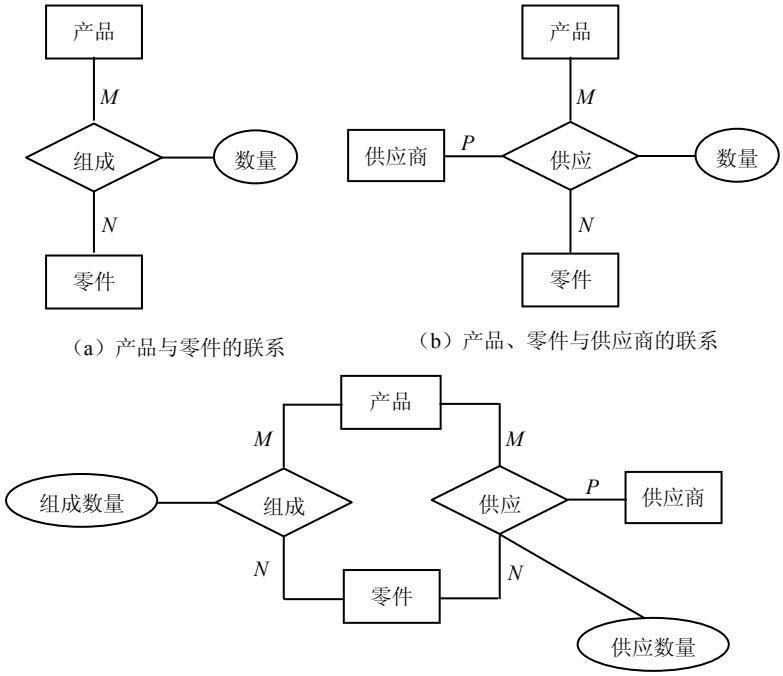
（b）学籍系统中的学生实体

（c）合并后的学生实体

图 3.24 合并后的 E-R 模型

③ 实体之间的联系在不同的局部 E-R 图中呈现出不同的类型。如实体 E_1 与 E_2 在某一应用中是多对多联系，而在另一应用中是一对多联系；又如在某一应用中实体 E_1 与 E_2 二者之间发生联系，而在另一应用中，实体 E_1、E_2、E_3 三者之间发生联系等。其解决方式为：根据具体应用的语义，对实体联系的类型进行适当的综合或调整。

例 3.7　在工程管理信息系统中，产品与零件之间的多对多联系如图 3.25（a）所示。产品、零件和供应商三者实体间多对多联系如图 3.25（b）所示。因为它们的语义不同，所以不具有包含关系。将它们综合起来合并成的 E-R 模型如图 3.25（c）所示。



（a）产品与零件的联系　　（b）产品、零件与供应商的联系

（c）综合后的联系

图 3.25　不同类型 E-R 图的综合

（3）命名冲突。主要表现在属性名、实体名或联系名之间的冲突。同名异义，即不同意义的对象具有相同的名字；异名同义，即同一意义的对象具有不同的名字。

命名冲突通常采用讨论、协商等行政手段解决。图 3.21 局部 E-R 图中的“学院”和“部门”是异名同义。事实上，各“学院”“处”“室”都是学校的一个“部门”，本例中统一命名为“部门”。

对于结构冲突，则要认真分析后才能消除。

设计全局 E-R 模型的目的不在于把若干局部 E-R 模型形式上合并为一个 E-R 模型，而在于消除冲突，使之成为能够被全系统中所有用户共同理解和接受的统一概念模式。

将图 3.21 的三个子系统的局部 E-R 模型消除冲突后，进行两两合并，可以得到如图 3.26 所示的初步的全局 E-R 模型。

4．全局模式的优化

在得到初步全局 E-R 模型后，为了提高数据库系统的效率，还应进一步地依据处理需求对 E-R 模型进行优化。一个好的全局 E-R 模型，除了能准确、全面地反映用户功能需求外，还应满足下列条件：

- 实体类型的个数尽可能少。
- 实体类型所含属性个数尽可能少。
- 实体类型间无冗余联系。

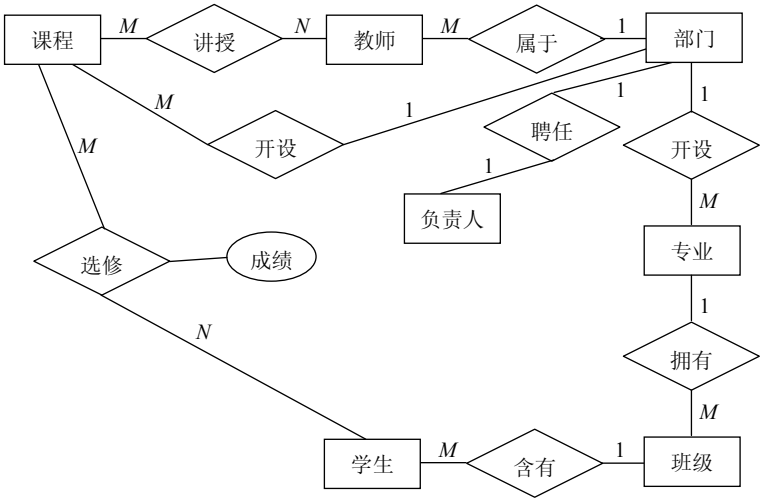


图 3.26　例 3.2 教学管理系统初步的全局 E-R 模型

但是，这些条件不是绝对的，要视具体的信息需求与处理需求而定。下面给出几个全局 E-R 模型的优化原则。

1）相关实体类型的合并

这里的合并不是前面的“公共实体类型”的合并，而是指相关实体类型的合并。在全局模式中，实体类型最终转换成关系模式，涉及多个实体类型的信息要通过连接操作获得。减少实体类型个数，就是减少连接的开销，以提高处理效率。

一般在权衡利弊后可以把 1:1 联系的两个实体类型进行合并。

具有相同键的实体类型常常是从不同角度刻画现实世界，如果经常需要同时处理这些实体类型，那么也有必要合并成一个实体类型。但这时可能会产生大量空值，因此，要对存储代价、查询效率进行权衡。

例 3.2 教学管理系统全局 E-R 模型中，只有“部门”和“负责人”实体间联系是 1:1 的，考虑到在学科发展、专业建设中常常需要较多的负责人个人信息，因此本例中两个实体不适宜合并。

2）冗余属性的消除

通常在各个局部结构中不允许存在冗余属性。但在综合成全局 E-R 模型后，可能产生全局范围内的冗余属性。例如，在教育统计数据库的设计中，一个局部结构 E_1 含有“毕业生数”“招生数”“在校学生数”和“预计毕业生数”等属性；另一局部结构 E_2 中含有“分年级在校学生数”“各专业学生数”和“各专业预计毕业生数”等属性。E_1 和 E_2 自身都无冗余，但综合成一个全局 E-R 模型时，E_1 的属性“在校学生数”和“预计毕业生数”可以从 E_2 的“分年级在校学生数”和“各专业预计毕业生数”推出，即为冗余属性。

一般地，同一非键的属性出现在几个实体类型中，或者一个属性值可从其他属性值导出时，理论上应该把这些冗余属性从全局模式中去掉。但在实际应用中，冗余属性消除与否，也取决于它对存储空间、访问效率和维护代价的影响。有时为了兼顾访问效率，有意保留冗余属性。

譬如 E_1 的属性“在校学生数”和“预计毕业生数”可以从全局模式中去掉。但是，如

果系统经常需要查询 E_1 的毕业生数、招生数、在校学生数及预计毕业生数，而此时 E_1 又消除了“在校学生数”和“预计毕业生数”两个属性，这样就需要频繁地将 E_1 和 E_2 进行连接，造成存储空间、访问效率和维护代价的低效。因此，可以在 E_1 中保留“在校学生数”和“预计毕业生数”这两个冗余属性。

3）冗余联系的消除

在全局模式中可能存在有冗余的联系，通常利用规范化理论中函数依赖的概念消除冗余联系。

在图 3.26 所示的教学管理系统初步的 E-R 模型中，“部门”实体与“课程”实体之间的“开设”联系可以由“部门”与“教师”实体之间“属于”联系、“教师”与“课程”之间的“讲授”联系推导出来，所以它属于冗余的联系，消除后即得教学管理系统优化的全局 E-R 模型，如图 3.27 所示。

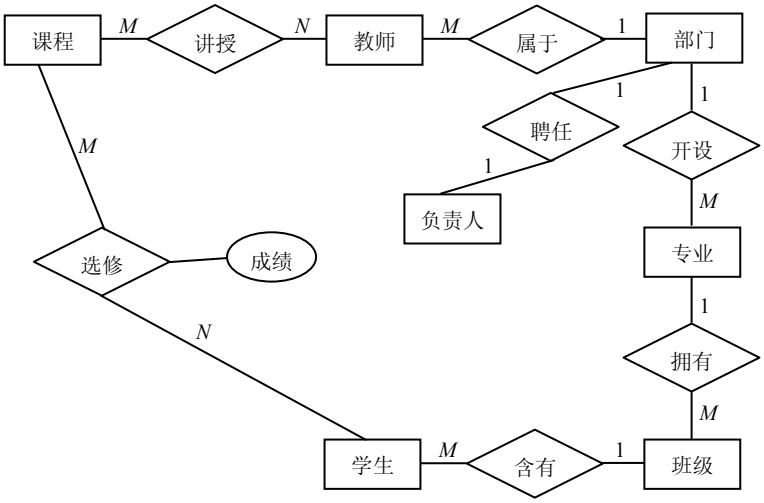


图 3.27　例 3.2 教学管理系统优化的全局 E-R 模型

5．最终全局 E-R 模型

最终全局 E-R 模型应满足如下要求：

（1）内部必须具有一致性，不再存在各种冲突。

（2）准确地反映原各局部 E-R 结构，包括属性、实体及实体间的联系。

（3）满足需求分析阶段所确定的所有需求。

全局 E-R 模型还应该提交给用户，征求用户和有关人员的意见，进行评审、修改和调整，最后确定的全局 E-R 模型为数据库的逻辑设计提供依据。

概念结构设计结果的文档资料一般包括整个组织的全局 E-R 图、有关说明及经过修订、充实的数据字典等。

3.3.6　概念结构设计实例

例 3.8　红星塑料厂产品生产综合信息管理系统的概念结构设计。塑料产品是先将各种形态的塑料原料（粉、粒料、黏合剂或分散体等）制成所需形状的塑料产品或塑料坯件。

对于塑料坯件还需要进行二次加工，将塑料坯件装配成为塑料产品。

为了简化该信息管理系统的设计，假设实例中只涉及产品设计、产品生产和材料存储三个模块。

根据概念结构设计的步骤，先确定局部范围，再进行局部概念结构设计，然后对各个局部概念结构进行综合。

1）确定局部范围

该系统的局部范围为产品设计、产品生产和材料存储三个子系统。

2）局部概念结构设计

（1）识别实体与实体的主键。

产品设计子系统：产品（主键：产品号）、坯件（主键：坯件号）、材料（主键：材料名）。

产品生产子系统：该部分生产的产品有成品和坯件，坯件还需要进一步装配，为了问题的简化，这里只涉及实体：产品（主键：编号）、材料（主键：材料名）。

仓库管理子系统：仓库存储分为成品存储、坯件存储、原材料存储，为了问题的简化，这里只涉及原材料的存储，涉及的实体有：原材料（主键：编号）、仓库（主键：仓库号）。

（2）定义实体间的联系。

在产品设计子系统中，有些“产品”可由“材料”直接生成，则“产品”实体和“材料”实体通过“使用”发生联系，是多对多联系；还有些“产品”由“坯件”装配而成，而“坯件”由“材料”生成，则实体“产品”和实体“坯件”通过“装配”发生联系，实体“坯件”和实体“材料”通过“消耗”发生联系，而且都是多对多联系。可得产品设计子系统的局部模型，如图 3.28 所示。

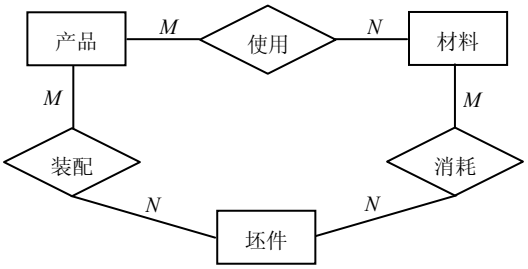


图 3.28　产品设计子系统局部 E-R 图

在产品生产子系统中，“产品”实体与“材料”实体通过“使用”联系在一起，而且是多对多联系。可得产品生产子系统的局部 E-R 模型，如图 3.29 所示。

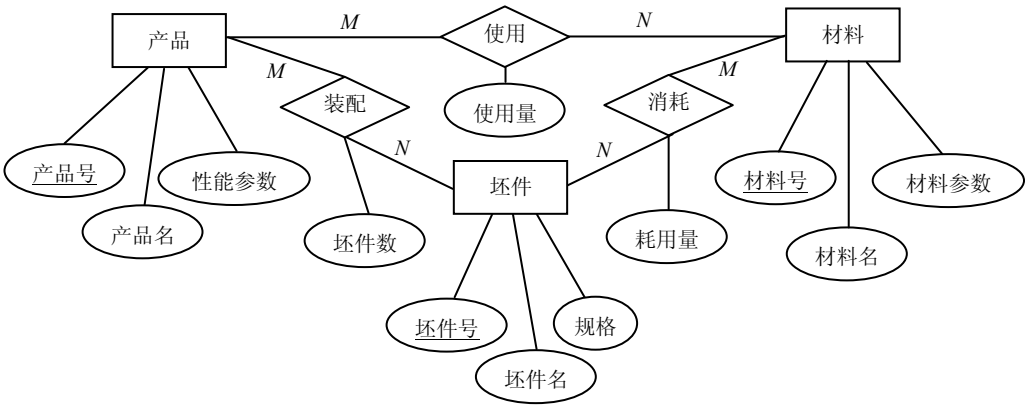
在材料存储子系统中，“材料”实体与“仓库”实体通过“存放”联系在一起，是多对多联系。可得材料存储子系统的局部 E-R 模型，如图 3.30 所示。

图 3.29　产品生产子系统的局部 E-R 图

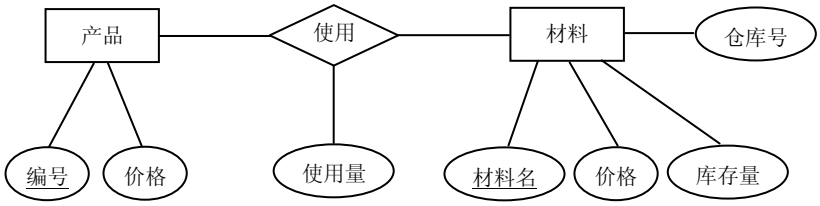
图 3.30　材料存储子系统的局部 E-R 图

（3）给实体及联系加上描述属性。

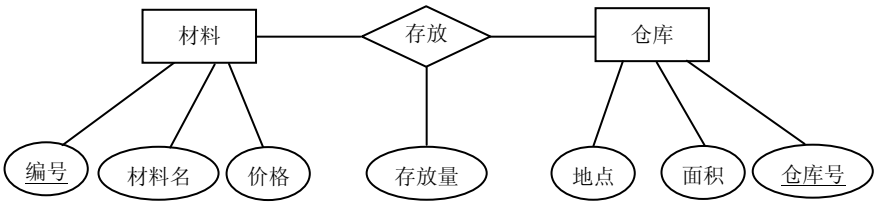
给实体和联系加上描述属性应根据具体的应用需求而定，实例中的内容是简化的，在具体的系统设计中根据需求分析来确定。如图 3.31（a）、（b）、（c）分别为产品设计子系统、产品生产子系统、材料存储子系统的实体及联系加上描述属性。



（a）产品设计子系统 E-R 图属性描述



（b）产品生产子系统 E-R 图属性描述



（c）材料存储子系统 E-R 图属性描述

图 3.31　各子系统实体和联系加描述属性

3）全局概念结构设计

分析和解决产品设计、产品生产和材料存储三个子系统局部 E-R 模型合并成全局 E-R 模型时的冲突问题。

对于属性冲突，因为该例中没有涉及具体企业应用对象和实际数据，所以在这里不需要讨论。但在实际应用时，可通过各子系统或不同应用设计人员间相互讨论、协商的方式加以解决。

对于命名冲突，在产品设计子系统局部 E-R 图中，实体“产品”有一个“产品号”属性，而在产品生产子系统的局部 E-R 图中，实体“产品”有一个“编号”属性，它们都是

实体“产品”的标识符，这里统一成“产品号”。

对于结构冲突，本例中第一个结构冲突，是“产品”实体在两个分 E-R 模型中属性组成部分不同的问题，取分 E-R 模型产品实体属性的并，然后统一属性名称，形成对“产品”实体新的描述，如图 3.32 所示。

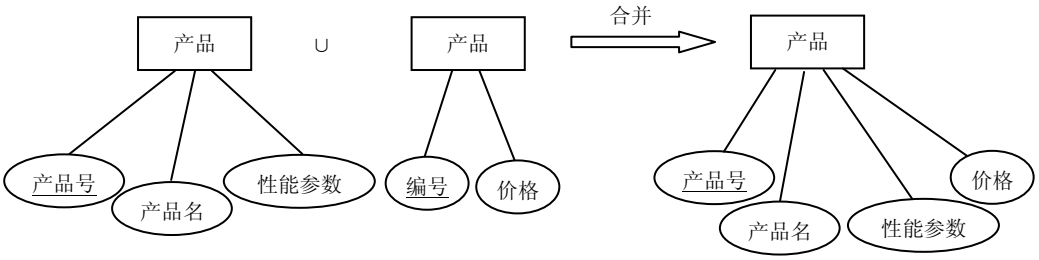


图 3.32　不同子系统的“产品”实体合并

本例中的第二个结构冲突，是在“仓库”对象在两个局部应用中具有不同的抽象，在产品生产子系统中作为“材料”实体的属性，而在材料存储子系统中它是一个单独的实体，为使同一对象仓库具有相同的抽象，必须在合并时把仓库统一作为实体加以处理。

本例中的第三个结构冲突是三个局部 E-R 图中的“材料”实体的信息描述。综合上面两个冲突的处理方法，“材料”实体合并如图 3.33 所示。

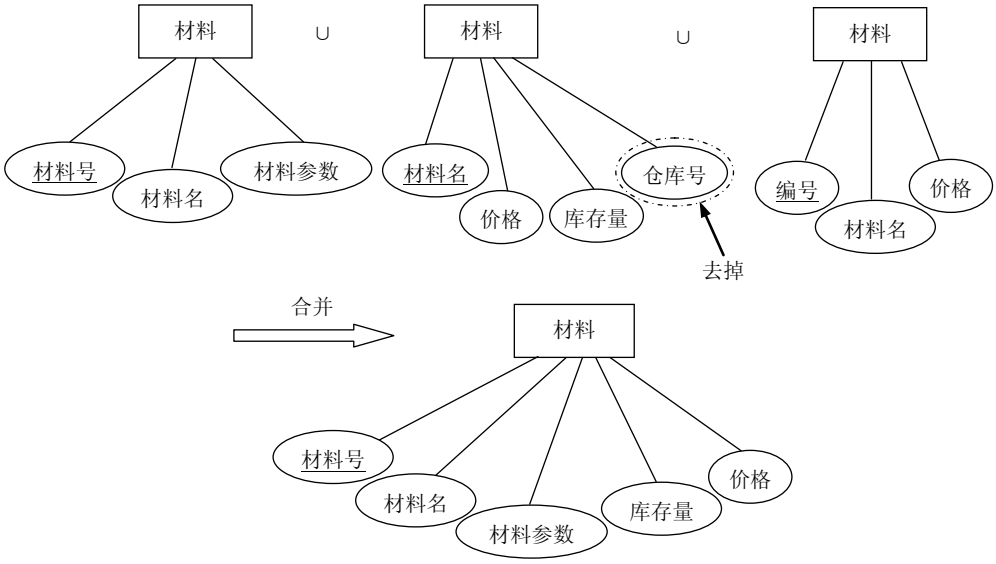


图 3.33　材料实体合并图示

在解决上述有关冲突后，综合各局部 E-R 模型可形成如图 3.34 所示初步的全局 E-R 模型。

4）全局 E-R 模型的优化

分析该 E-R 模型的数量属性可知，初步 E-R 模型存在着存放量、库存量等属性冗余问题。消除这些冗余后，可以得到如图 3.35 所示优化的全局 E-R 模型。

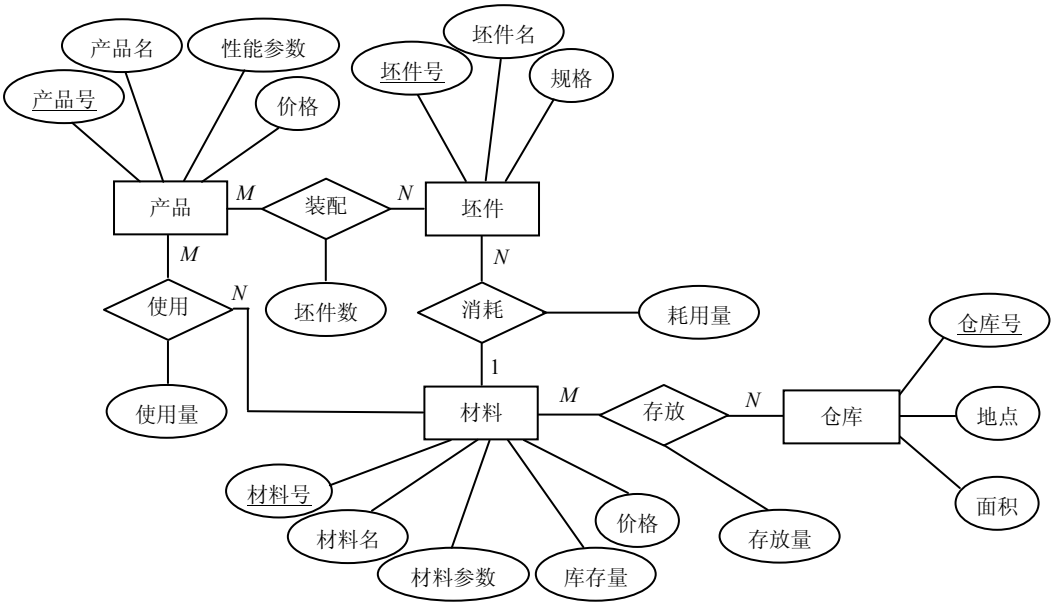


图 3.34　综合信息管理系统的初步全局 E-R 图

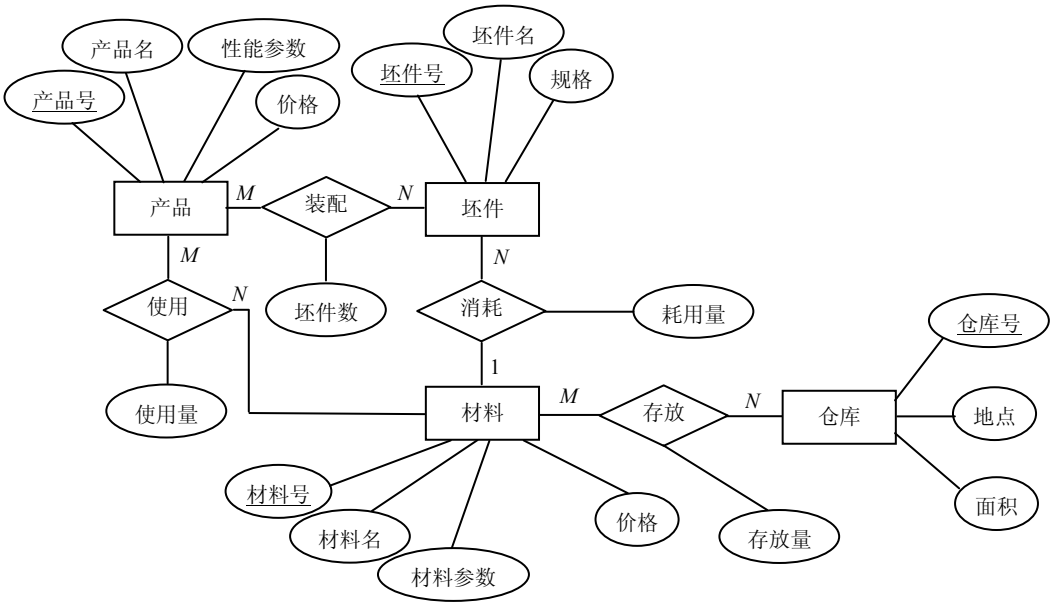


图 3.35　综合管理系统优化的全局 E-R 图

目前我们所产生的 E-R 模型，仅仅是红星塑料厂产品生产综合信息管理系统的一个基本概念模式，它表示了用户的数据处理要求，是沟通用户需求和系统设计的桥梁。但是，要想把它确定下来作为最终概念模式，设计者还应提交给用户，并与用户反复讨论、研究，同时征求用户和有关人员的意见，进行评审、修改和优化等工作。在用户确认这一模式已正确无误地反映了他们的需求后，才能作为最终的数据库概念结构，进行下一阶段的数据库设计工作。

扫一扫

微课视频

3.4 逻辑结构设计

概念结构设计的结果是得到一个与计算机硬件、软件和 DBMS 无关的概念模式。而逻辑设计的目的是把概念结构设计阶段设计好的全局 E-R 模型转换成与选用的具体机器上的 DBMS 所支持的数据模型相符合的逻辑模型（如网状、层次、关系或面向对象模型等）。如果选用的是关系型 DBMS 产品，逻辑结构设计是指设计数据库中所应包含的各个关系模式的结构，其中有各关系模式的名称、各属性的名称、数据类型、取值范围等内容。

从理论上讲，设计逻辑结构应该选择最适于相应概念结构的数据模型，然后对支持这种数据模型的各种 DBMS 进行比较，从中选出最合适的 DBMS。但实际情况往往是已给定了某种 DBMS，设计人员没有挑选的余地。本章只讨论目前比较流行的关系型数据库的逻辑结构设计，即如何把全局 E-R 模型转换为关系模型的原则和方法。

关系数据库逻辑设计的结果是一组关系模式的定义。逻辑设计过程可分为 E-R 图向关系模式的转换、关系规范化处理、对关系模式进行评价与修正等几个步骤，如图 3.36 所示。

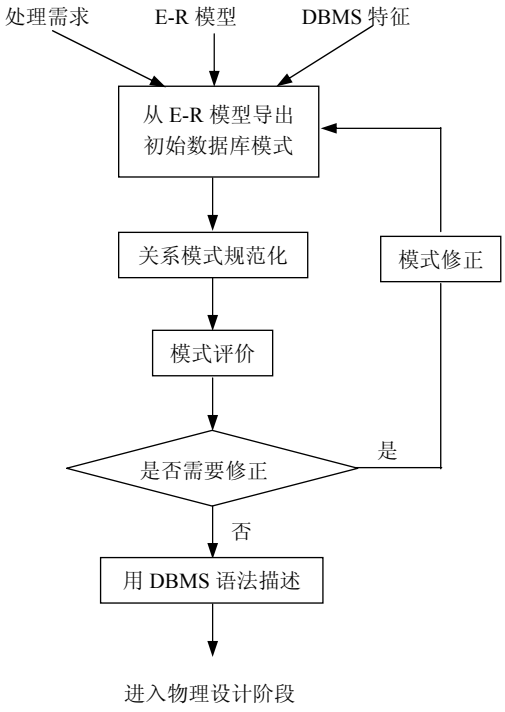


图 3.36 逻辑结构设计过程示意图

从图 3.36 可以看出，概念结构设计的结果直接影响到逻辑结构设计的复杂性和效率。

3.4.1 E-R 模型向关系模式的转换

关系模式由一组关系（二维表）组成，而 E-R 模型则是由实体、实体所对应的属性、

实体间的相互联系三个要素组成。所以将 E-R 模型转换为关系模式实际上就是要将实体、实体的属性和实体间的联系转换为关系模式的过程。这种转换一般遵循如下规则。

1．实体类型

将每个实体类型转换成一个关系模式。实体的属性转换为关系的属性，实体标识符转换为关系模式的键。

2．二元联系类型

联系类型转换成关系模式是根据不同的情况做不同的处理。两个实体的联系类型转换为关系模式的原则如下：

（1）若实体间的联系是 1:1 的，可以在两个实体类型转换成的两个关系模式中的任意一个关系模式的属性中加入另一个关系模式的键和联系类型的属性。

例 3.9 教育管理信息系统中的实体“校长”与“学校”之间存在着 1:1 的联系，如图 3.37 所示。

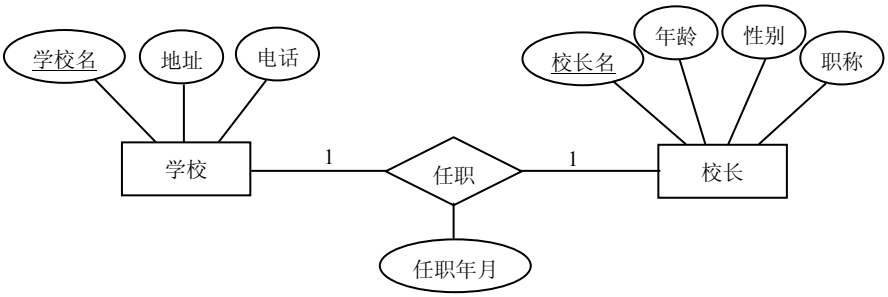


图 3.37 “学校”和“校长”1:1 联系图

在将其转化为关系模式时，“校长”与“学校”各为一个关系模式。如果用户经常要在查询学校信息时同时查询校长信息，那么就可以在学校关系模式中加入校长名和任职年月，其关系模式设计如下（加下画线者为主键，加波浪线者为外键）：

学校关系模式（学校名，地址，电话，校长名，任职年月）

校长关系模式（校长名，年龄，性别，职称）

（2）若实体间的联系是 1:*N* 的，则在 *N* 端实体类型转换成的关系模式中加入 1 端实体类型转换成的关系模式的键和联系类型的属性。

在例 3.5 中，选课管理子系统中的实体“学院”与“学生”之间存在着 1:*N* 的联系，其转换成的关系模式如下：

学生关系模式（学号，姓名，性别，出生年月，学院号）

学院关系模式（学院号、名称、地址、电话）

弱实体：弱实体间的联系是 1:*N* 的，而且在 *N* 端实体类型为弱实体，转换成关系模式时，将 1 端实体类型（父表）的键作为外键放在 *N* 端的弱实体（子表）中。弱实体的主键由父表的主键与弱实体本身的候选键组成。也可以为弱实体建立新的独立的标识符 ID。

例 3.10 某单位职工管理信息系统中的实体“职工”与弱实体“亲属关系”之间存在着 1:*N* 的联系，其 E-R 图如图 3.38 所示。

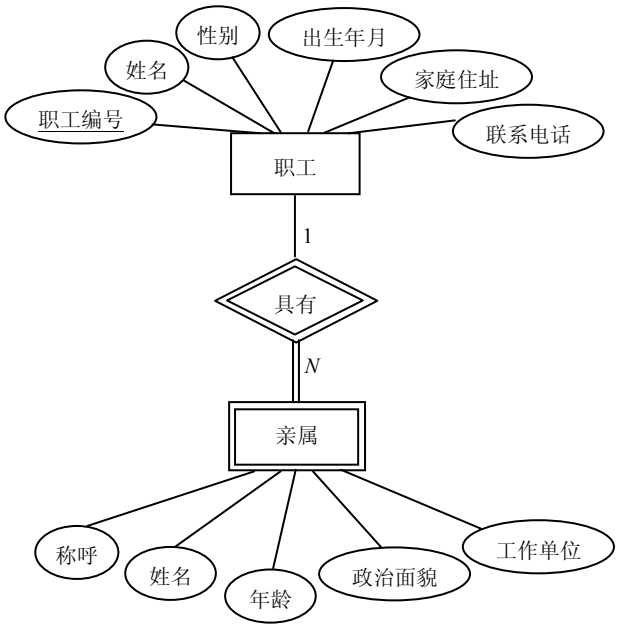


图 3.38　弱实体联系图

转换成的关系模式如下：

职工关系模式（职工编号，姓名，性别，出生年月，家庭住址，联系电话）。

亲属关系模式（职工编号，称呼，姓名，年龄，政治面貌，工作单位）。

（3）若实体间的联系是 *M*:*N* 的，则除两端实体分别转化为两个关系模式外，其联系类型也转换成关系模式，它的属性为两端实体类型的键加上联系类型的属性，而键是包含两端实体键的组合。

例 3.11　在例 3.2 的选课管理子系统中，实体“学生”与“课程”之间存在着 *M*:*N* 的联系，其 E-R 图如图 3.39 所示。

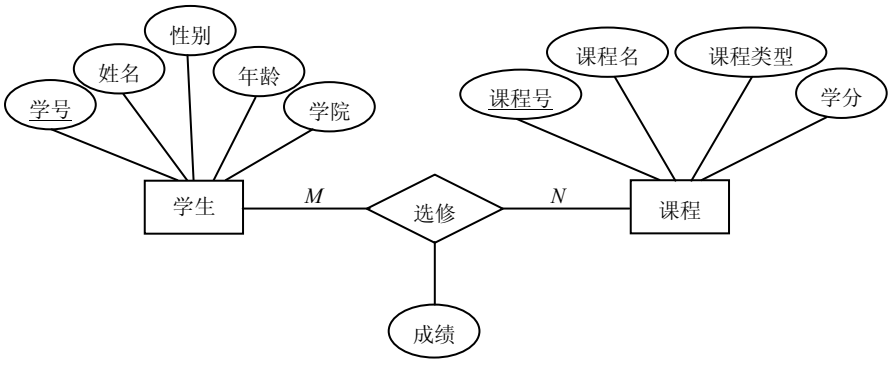


图 3.39　实体“学生”与“课程”联系图

转换为如下关系模式：

学生关系模式（学号，姓名，性别，年龄，学院）

课程关系模式（课程号，课程名，课程类型，学分）

选修关系模式（学号，课程号，成绩）

3．三元联系类型

三个实体间联系类型转换成关系模式与二元联系类似。三个实体都转换为一个关系模式，其联系类型转换为关系模式的原则如下：

（1）若实体间的联系是 1:1:1，可以在三个实体类型转换成的三个关系模式中任意一个关系模式的属性中加另两个关系模式的键和联系类型的属性。

（2）若实体间的联系是 1:1:*N*，则在 *N* 端实体类型转换成的关系模式中加入 1 端实体类型的键和联系类型的属性。

（3）若实体间的联系是 1:*M*:*N*，则将联系类型也转换成关系模式，属性为 1 端、*M* 端和 *N* 端实体类型的键加上联系类型的属性，而键是包含 *M* 端和 *N* 端实体键的组合。

（4）若实体间的联系是 *M*:*N*:*P*，则将联系类型也转换成关系模式，属性为三端实体类型的键加上联系类型的属性，而键是包含三端实体键的组合。

例 3.12　实体“供应商”“项目”和“零件”三元联系的 E-R 图如图 3.40 所示。

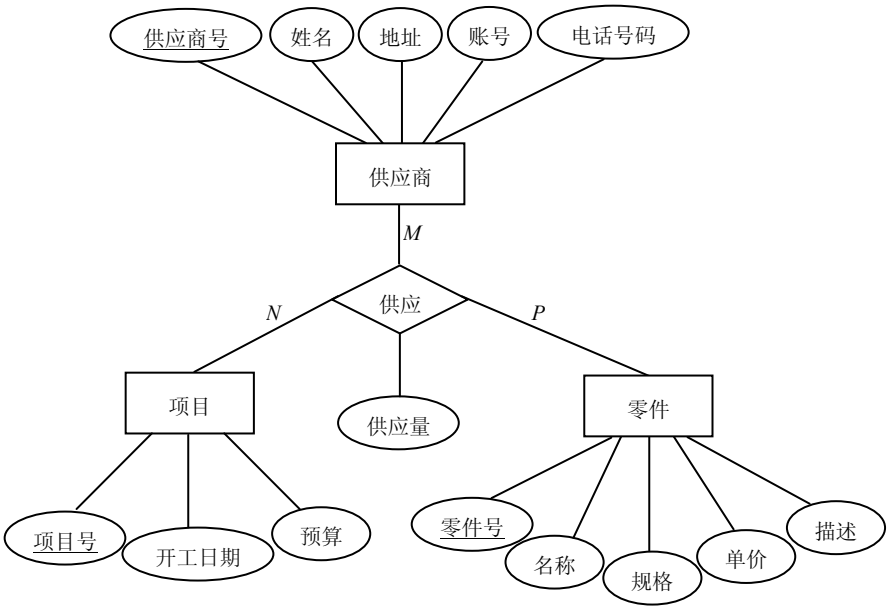


图 3.40　三元联系的 E-R 图

转化的关系模式如下：

实体“供应商”“项目”和“零件”转化为关系模式为：

供应商关系模式（供应商号，姓名，地址，账号，电话号码）

项目关系模式（项目号，开工日期，预算）

零件关系模式（零件号，名称，规格，单价，描述）

联系“供应”转化的关系模式为：

供应关系模式（供应商号，项目号，零件号，供应量）

3.4.2 关系模式的优化

在关系数据库的逻辑设计中，先是利用 E-R 模型向关系模式转换规则初步得到一组关系模式集后，还应该再适当地修改、调整关系模式的结构，以进一步提高数据库应用系统的性能，这个过程称为关系模式的优化。

关系模式的优化通常以规范化理论为指导。优化关系模式的方法如下：

1．确定函数依赖

根据需求分析阶段所得到的数据语义，分别写出每个关系模式内部各属性之间的函数依赖以及不同关系模式属性之间的函数依赖。

例如，在图 3.39 转换的关系模式中，学生关系模式内部存在下列函数依赖：

学号→（姓名，性别，年龄，学院）

课程关系模式内部存在下列函数依赖：

课程号→（课程名，课程类型，学分）

选修关系模式中存在下列函数依赖：

（学号，课程号）→ 成绩

学生关系模式的“学号”与选修关系模式的“学号”之间存在下列函数依赖：

学生.学号→选修.学号

课程关系模式的“课程号”与选修关系模式的“课程号”之间存在下列函数依赖：

课程.课程号→选修.课程号

2．关系模式的规范化

根据规范化理论对关系模式的函数依赖逐一进行分析，检查是否存在部分函数依赖、传递函数依赖等，确定各关系模式分别属于第几范式。进一步考查关系模式的规范程度在应用环境中是否合适，以确定是否要对它们进行合并或分解。要注意到，在对关系模式进行分解时，除了考虑数据等价和函数依赖等价以外，还要考虑到应用系统的效率。

关于关系模式的规范化问题，做如下两点说明：

- 并不是规范化程度越高的关系就越好。当一个应用的查询中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，而连接运算的代价是相当高的，可以说关系模式操作低效的主要原因就是做连接运算引起的。在这种情况下，第二范式甚至第一范式也许是最好的。
- 如果一个关系模式在实际应用中只是提供查询，并不提供更新操作，或者很少提供更新操作，此时不会存在更新异常问题或更新异常不是主要问题，可以不对关系模式进行分解。

例如，在关系模式学生成绩单（学号，英语，数学，语文，总分）中存在下列函数依赖：

学号→（英语，数学，语文）

（英语，数学，语文）→ 总分

因此，“学号→平均成绩”是传递函数依赖。由于关系模式中不存在局部函数依赖而存在传递函数依赖，所以是 2NF 关系。

虽然“总分”可以由其他属性推算出来，但如果某应用中需要经常查询学生的总分，

为了提高查询效率，关系模式中仍然可保留该冗余数据，对关系模式不再做进一步分解。

对于一个具体应用来说，规范化应进行到什么程度，需要根据具体情况而定。一般来说，关系模式达到第三范式就能获得比较满意的效果。

3．关系模式进行必要的分解

在一些具体应用中，常常需要对关系模式进行必要的分解，以提高数据操作的效率和存储空间的利用率。

常用的分解方法有两种：水平分解和垂直分解。

1）水平分解

所谓水平分解，是指把一个关系模式 *R* 中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，一个关系很大（这里指元组数多），而实际应用中，经常使用的数据只是一部分（通常至多占元组总数的 20%），此时可以将经常用到的这部分数据分解出来，形成一个子关系，这样可以减少查询的数据量。

另外，如果关系 *R* 上具有 *n* 个并发事务，而且多数事务存取的数据不相交，则 *R* 可分解为少于或等于 *n* 个子关系，使每个事务存取的数据对应一个子关系。

例如，有一个产品关系模式，其中包含有出口产品和内销产品两类数据。由于不同的应用对应不同类型的产品，如一个应用只对应出口产品，而另一个应用只对应内销产品。因此，可将产品关系模式进行水平分解。分解为两个关系模式：一个存放出口产品数据，另一个存放内销产品数据，如图 3.41 所示。这样可以提高应用存取的效率。

出口产品

产品号	产品名	型号规格	…
…	…	…	…
…	…	…	…

内销产品

产品号	产品名	型号规格	…
…	…	…	…
…	…	…	…

图 3.41　关系模式水平分解举例

2）垂直分解

所谓垂直分解，是把一个关系模式 *R* 的属性分解为若干子集合，形成若干子关系模式。

例如有一个职工关系模式，其中含有“职工号”“职工名”“性别”“职务”“职称”“出生日期”“地址”“邮编”“电话”“所在部门”等描述属性。如果应用中经常存取的数据是职工号、职工名、性别、职务等信息，而其他数据很少使用，则可以对职工关系模式进行垂直分解，即分解为两个关系模式：一个存放经常使用的数据，另一个存放不常使用的数据，如图 3.42 所示。这样也可以减少应用存取的数据量。

职工 1

职工号	职工名	性别	职务	…
…	…	…	…	…
…	…	…	…	…

职工 2

职工号	出生日期	地址	邮编	…
…	…	…	…	…
…	…	…	…	…

图 3.42　关系模式垂直分解举例

一般来说，凡是经常在一起使用的属性应从 *R* 中分解出来形成一个子关系模式，这样也可以提高数据操作的效率。

垂直分解的好处是可以提高某些事务的效率；不足之处是可能会使得另一些事务不得不执行连接操作，从而降低效率。是否需要垂直分解，取决于分解后 *R* 上的所有事务的总效率是否得到了提高。

垂直分解的方法可以采用简单的 E-R 模型分裂操作（如例 3.3），也可以用关系模式分解算法进行分解。需要注意的是，垂直分解必须以不损失关系模式的语义（保持无损分解和保持函数依赖分解）为前提。

下面通过一个例子来说明关系模式优化的过程。

例 3.13 假设有一个从 E-R 图直接转化过来的选修课程关系模式（学号，姓名，年龄，课程名称，成绩，学分）。请分析该关系属于第几范式？如果应用中需要常常对选修课程关系进行增、删、改操作，该关系存在什么问题？并对其设计进行优化。

解：关系的每个属性都具有原子性，因此属于第一范式。

由于每个学生可能选修多门课程，而每门课程对应一个成绩。因此，该关系的候选键为（学号，课程名称）。

根据数据的语义，该关系上存在的函数依赖集为：

（学号，课程名称）→（姓名，年龄，成绩，学分），课程名称→学分，学号→（姓名，年龄）。

由于（学号，课程名称）→（姓名，年龄），学号→（姓名，年龄）。该关系存在非主属性对候选键的部分函数依赖，因此，选修课程关系属于第一范式，且存在以下问题：

（1）数据冗余。如果同一门课程由多个学生选修，“学分”就会重复多次；如果同一个学生选修了多门课程，该学生的姓名和年龄就会重复多次。

（2）更新异常。若调整了某门课程的学分，则关系中选修该门课程所有学生元组的“学分”属性值都要更新，否则会出现同一门课程学分不同的情况。

（3）插入异常。假定要开设一门新的课程，暂时还没有学生选修。此时，由于候选键中“学号”没有值，所以课程名称和学分也无法插入到关系中。

（4）删除异常。假设有一批学生已经完成课程的选修，这些学生元组就应该从选修课程关系中删除。但与此同时，课程名称和学分信息也有可能被删除。

由于选修课程关系中的数据需要经常更新，所以必须解决上述可能出现的操作异常问题。

通过对选修课程关系模式的函数依赖进行逐一分析，可将选修课程关系模式分解为以下三个子关系模式：

学生（学号，姓名，年龄）

课程（课程名称，学分）

选课（学号，课程名称，成绩）

其中，学生关系模式上的候选键为“学号”，函数依赖集为：

{学号→姓名，学号→年龄}

课程关系上的候选键为“课程名称”，函数依赖集为：

{课程名称→学分}

选课关系上的候选键为（学号，课程名称），函数依赖集为：

{（学号，课程名称）→成绩}

由于不存在非主属性对候选键的部分函数依赖和传递函数依赖，因此，学生子关系模式、课程子关系模式和选课子关系模式均属于第三范式。因此，如果需要增加、删除以及修改相关数据信息，只需要对相关子关系模式进行操作即可。

另外，如果应用中的查询常常是统计学生的选课情况，则分解后带来的自然连接操作很少。因此，这样的设计是合理的。

以上通过对选修课程关系模式的分解，使各子关系模式达到了 3NF，基本上消除了数据冗余和操作异常。因此，关系模式得到了优化。

3.5 物理结构设计

将逻辑设计阶段中产生的数据库逻辑模型结合指定的 DBMS 设计出最适合应用环境的物理结构的过程，称为物理结构设计。它的任务是为数据库选择合适的存储结构与存取方法，也就是设计数据库的内模式。物理结构设计阶段一般分为设计物理结构和评价物理结构两部分，如图 3.43 所示。

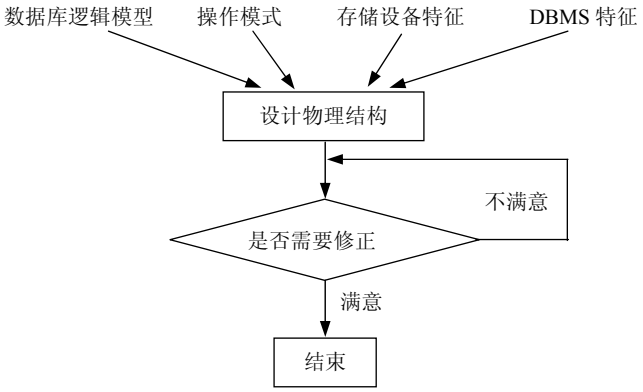


图 3.43　物理设计过程示意图

3.5.1 设计物理结构

由于用户最终是通过某一特定的 DBMS 使用数据库，因此，数据库的物理设计必须结合具体的 DBMS 进行，主要包括选择数据库的存储结构和存取方法两个方面。

1．确定存储结构

数据库的物理设计与特定的硬件环境、DBMS 及实施环境都密切相关，因此，在确定数据库的物理结构时，必须仔细阅读、参考具体 DBMS 的规定。一般说来，基本的存储结构（如顺序、散列）已有具体的 DBMS 确定，无须做太多考虑，设计人员主要考虑的因素是存储时间、存储空间和维护代价等方面。

数据库的配置也是确定数据库存储结构的重要内容，包括数据库空间的分配、日志文件的大小、数据字典空间的确定以及相关参数设置（如并发用户数、超时限制）等。一般

的 DBMS 产品都提供了一些有效存储分配的参数，供设计者在进行物理优化时选择。设计者在进行数据库的配置时也要仔细参考具体的 DBMS 手册。

2．选择存取方法

存取方法有索引、聚簇等方法。目前的 DBMS 一般都支持索引、聚簇等方法。

1）索引的选择

索引的选择是数据库物理设计的基本问题之一。物理设计中一般要解决对关系的哪些属性列建立索引、建立何种类型的索引等问题。一般说来，常常需要对下列情况的属性列建立索引：

（1）查询很频繁的属性列。

（2）经常出现在连接操作中的属性列。

（3）WHERE、ORDER、GROUP BY 等子句中的属性列。

不宜建索引的属性列一般有：

（1）不出现或很少在查询条件中出现的属性列。

（2）属性值很少的属性列，如“性别”属性列（“男”“女”）。

（3）属性值严重分布不均匀的属性列。

（4）经常需要更新的属性列。

（5）经常需要更新或含有记录较少的数据表的属性列。

（6）属性值内容过长的属性列，如人事管理信息中的“简历”等。

关系上定义的索引并不是越多越好，多建索引虽然可以缩短存取时间，但是增加了索引文件占用的存储空间及维护代价。

2）聚簇的选择

聚簇是改进系统性能的另一种技术。聚簇技术就是把有关的元组集中在一个物理块内或物理上相邻的区域内，以提高某些数据的访问速度。例如，要查询某地今年参加英语四级考试的学生信息（设有 30 万名），在极端的情况下，这 30 万名学生数据元组分散存储在 30 万个不同的物理块上。在做这种查询时，即使不考虑访问索引的次数，要访问这 30 万个学生的数据也需 30 万次 I/O 操作才能完成。如果将参加四级考试的学生按学校集中存放，则每存取一个物理块，就可以得到多个符合条件的学生元组，这样就会明显减少访问磁盘的次数。现代的 DBMS 一般都支持聚簇存放技术。聚簇分为三种情况：

分段——按属性分组，将文件在垂直方向进行分解。例如，将经常使用的属性域较少存取的属性分开存储到不同的存储设备或者存储区域上。

分区——将文件进行水平分解，按照记录存取频度进行分组。即将访问频率高的记录和访问低的记录分开并存储到不同的存储设备或者存储区域上。

聚簇——从不同的关系中取出某些属性物理地存储在一起，以改变连接查询的效率。

3.5.2 评价物理结构

数据库物理结构设计实际完成后，还应该进行评价，以确定物理设计结构是否满足设计要求。评价物理结构包括评价内容、评价指标和评价方法。

评价内容：存取方法选取的正确性、存取结构设计的合理性、文件存放位置的规范性和存取介质选取的标准性等。

评价指标：存取空间的利用率、存取数据的速度和维护费用等。

评价方法：根据物理结构的评价内容，统计存储空间的利用率、数据的存取速度和维护费用等指标。

如果物理设计结果满足了用户和设计的需求，则可以进入数据库实施阶段，否则需要修正甚至重新考虑物理结构的设计。一般说来，数据库的物理设计都需要反复测试、不断优化。

3.6 数据库的实施

当前面各阶段的数据库设计工作圆满完成后，就进入建立数据库的工作阶段。数据库的实施就是根据前面逻辑设计与物理设计的结果，利用 DBMS 工具和直接利用 SQL 命令在计算机上建立其实际的数据库结构、整理并装载数据，编制和调试应用程序。

（1）建立数据库结构。利用给定的 DBMS 所提供的命令，建立数据库的结构、外模式、内模式。对关系型数据库来说，就是创建数据库及数据库中所包含的基本表、视图、索引等。

（2）将原始数据装入数据库。装入数据的过程非常复杂，这是因为原始数据一般分散在企业的各个不同部门，而且它们的组织方式、结构和格式都与新设计数据库系统中的数据有不同程度的区别。因此，必须将这些数据从各个地方抽取出来，输入计算机，并经过分类转换，使它们的结构与新系统的数据库结构一致，然后才能输入到数据库中。

一般调试程序时需要将少量的、适合程序调试用的数据装入到数据库，系统运行正常后则需要将所有的原始数据装入到数据库。一般地，装入大批量数据应设计输入子系统进行数据输入。

（3）应用程序的编制和调试。与数据装载同时进行的工作是应用程序的编制和调试。在所编写的应用程序中都需要嵌入 SQL 语句来进行数据库数据的查询和更新。应用程序的设计、编码和调试的方法请参考有关软件工程的书籍。

3.7 数据库运行和维护

数据库试运行结果符合设计目标后，数据库就可以真正投入运行了。数据库系统正式运行，标志着数据库设计与应用开发工作的结束和维护阶段的开始。运行维护阶段的主要任务有如下几个方面：

（1）数据库的转储和恢复。DBA 应定期对数据库进行备份，将其转储到磁盘或其他存储设备上。这样，一旦数据库遭到破坏时可以及时地将其恢复。

（2）数据库的安全性和完整性控制。按照设计阶段规定的安全和故障恢复规则，经常监督系统的安全性，及时调整授权或密码等，如果数据库系统的完整性约束发生了变化，DBA 应该及时调整和修正。

（3）数据库性能的监督、分析和改造。数据库的设计成功和运行并不意味着数据库性能是最优的、最先进的。在数据库系统的运行过程中，DBA 需要密切关注系统的性能，监督系统的运行，并对监督数据进行分析，不断改进系统的性能。

（4）数据库的重组织与重构造。数据库系统的运行过程中，经常会对数据库进行插入、删除和修改等更新操作，这些操作会破坏数据库的物理存储，也会直接影响存储效率和系统性能。例如，由于多次的插入、删除和修改等更新操作，可能会使逻辑上属于同一记录类型或同一关系的数据被分散到不同的文件或文件的多个碎片上，就会降低数据的存取效率。此时，DBA 要负责对数据库进行重新组织，即按原设计要求重新安排数据的存储位置、回收垃圾、减少指针链等，以提高数据的存取效率和系统性能。

另外，数据库的应用环境也是不断变化的，经常会出现一些新的应用和消除一些旧的应用，这将导致出现新实体而淘汰旧实体，同时原先实体的属性和实体间的联系也会发生变化。因此，需要局部地调整数据库的逻辑结构，增加一些新的关系，删除一些旧的关系，或在某些关系中增加（删除）一些属性等，这就是数据库的重构造。当然，数据库的重构造是十分有限的，如果应用环境变化太大，重构造已无法满足用户的要求，就应该淘汰旧的系统，设计新的数据库系统。

习　题　3

3-1　名词解释

数据库设计　　数据流图　　数据字典　　弱实体　　概念结构设计　　逻辑结构设计　　物理结构设计

3-2　什么是数据库设计目标？数据库设计的基本步骤有哪些？

3-3　数据库设计的需求分析阶段是如何实现的？任务是什么？

3-4　概念设计的具体步骤是什么？

3-5　简述采用 E-R 方法的数据库概念设计过程。

3-6　逻辑设计的目的是什么？试述逻辑设计过程的输入和输出环境。

3-7　规范化理论对数据库设计有什么指导意义？

3-8　什么是数据库结构的物理设计？主要包含哪几方面的内容？

3-9　数据库实施阶段主要做哪几件事情？

3-10　数据库系统投入运行后，有哪些维护工作？

3-11　设某商业集团数据库中有三个实体集：一是“商店”实体集，属性有商店编号、商店名、地址等；二是“商品”实体集，属性有商品号、商品名、规格、单价等；三是“职工”实体集，属性有职工编号、姓名、性别、业绩等。

商店与商品间存在“销售”联系，每个商店可销售多种商品，每种商品也可放在多个商店销售，每个商店销售的每一种商品有月销售量记录；商店与职工间存在着“聘用”联系，每个商店有许多职工，每个职工只能在一个商店工作，商店聘用职工有聘期和月薪。

试画出 E-R 图，并在图上注明属性、联系的类型。再转换成关系模式集，并指出每个关系模式的主键和外键。

3-12　设某商业集团数据库中有三个实体集：一是“公司”实体集，属性有公司编号、公司名、地址等；二是“仓库”实体集，属性有仓库编号、仓库名、地址等；三是“职工”实体集，属性有职工编号、姓名、性别等。

公司与仓库间存在“隶属”联系，每个公司管辖若干仓库，每个仓库只能属于一个公

司管辖；

仓库与职工间存在“聘用”联系，每个仓库可聘用多个职工，每个职工只能在一个仓库工作，仓库聘用职工有聘期和工资。

试画出 E-R 图，并在图上注明属性、联系的类型。再转换成关系模式集，并指出每个关系模式的主键和外键。

3-13　设某商业集团数据库中有三个实体集：一是“商品”实体集，属性有商品号、商品名、规格、单价等；二是“商店”实体集，属性有商店号、商店名、地址等；三是“供应商”实体集，属性有供应商编号、供应商名、地址等。

供应商与商品间存在“供应”联系，每个供应商可供应多种商品，每种商品可向多个供应商订购，供应商供应每种商品有月供应量；商店与商品间存在“销售”联系，每个商店可销售多种商品，每种商品可在多个商店销售，商店销售商品有月计划数。

试画出 E-R 图，并在图上注明属性、联系的类型。再转换成关系模式集，并指出每个关系模式的主键和外键。

3-14　假设要为银行的储蓄业务设计一个数据库，其中涉及储户、存款、取款等信息，试设计 E-R 模型。

3-15　假设某超市公司要设计一个数据库系统来管理该公司的业务信息。该超市公司的业务管理规则如下：

（1）该超市公司有若干仓库，若干连锁商店，供应若干商品。

（2）每个商店有一个经理和若干收银员，每个收银员只在一个商店工作。

（3）每个商店销售多种商品，每种商品可在不同的商店销售。

（4）每个商品编号只有一个商品名称，但不同的商品编号可以有相同的商品名称。每种商品可以有多种销售价格。

（5）超市公司的业务员负责商品的进货业务。

试按上述规则设计 E-R 模型。

3-16　假设要根据某大学的系、学生、班级、学会等信息建立一个数据库。一个系有若干专业，每个专业每年只招一个班，每个班有若干学生；一个系的学生住在同一宿舍区；每个学生可以参加多个学会，每个学会有若干学生，学生参加某学会有入会年份。试为该大学的系、学生、班级、学会等信息设计一个 E-R 模型。