第 3 章 数据库设计技术

本章主要介绍数据库设计的方法和步骤，包括从数据库设计的需求分析开始，到数据库概念结构设计、逻辑结构设计、物理结构设计以及数据库的实施、运行和维护的整个过程。如果说第 2 章学的是数据库设计理论，那么本章介绍的就是对这些理论的应用方法。通过本章的学习，读者应该了解和掌握下列内容。

- 了解数据库设计的一般步骤，掌握需求分析的过程和方法。
- 掌握数据库结构设计的方法，包括概念结构设计、逻辑结构设计和物理结构设计。
- 理解数据库的实施、运行和维护方法。

3.1 数据库设计概述

在应用数据库技术解决实际问题时，需要针对给定应用环境，构造最优的数据库模式，然后基于该模式创建相应的数据库及其应用系统，使得形成的数据库系统能够有效地进行各种数据存储和管理任务，满足用户的各种应用需求，而这个过程就是数据库设计。

一般地，数据库设计是指在现有的应用环境下，从建立问题的概念模型开始，逐步建立和优化问题的逻辑模型，最后建立其高效的物理模型，并据此建立数据库及其应用系统，使之能够有效地收集、存储和管理数据，满足用户的各种应用需求。简单而言，数据库设计是数据库及其应用系统的设计。本章介绍的数据库设计主要是针对数据库本身的设计，较少涉及应用系统的设计。

数据库设计是数据库应用系统开发的关键技术之一，其最终目的可以归结为两点：①满足用户的需求；②简化应用程序的编程设计，实现系统协同、高效的开发，减少开发成本。从过程看，数据库设计主要分为 6 个步骤：系统需求分析、概念结构设计、逻辑结构设计、物理结构设计、数据库实施、数据库系统运行和维护，各步骤的先后关系如图 3.1 所示。其中，对每一个步骤，如果设计结果不满足要求，都可以返回前面的任一步骤，直到满足要求为止。

由于实际问题的时空复杂性，数据库设计过程中也存在诸多的不确定因素，加上应用程序运行环境的制约，这就使得数据库设计变得异常复杂。一般来说，数据库设计不是“一次到位”，而是“认识-设计-纠正-认识”的一种反复并逐步求精的过程。但是经过长期的积累，人们总结了数据库设计有关理论和方法，形成了数据库设计的一些基本规律，可为实际的数据库设计提供理论和经验参考。

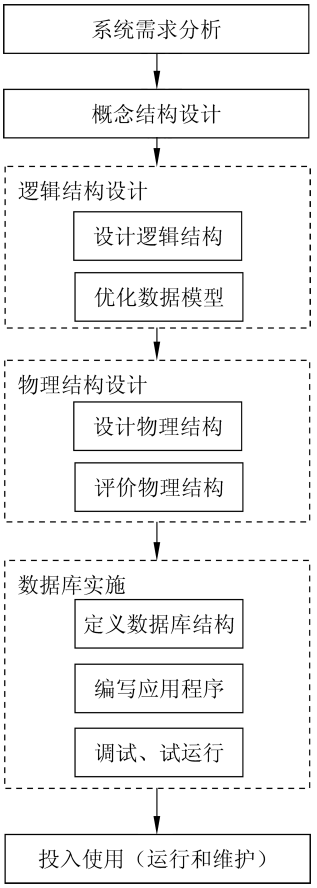


图 3.1　数据库设计的基本步骤

3.2　需 求 分 析

需求分析是了解用户需求，然后明确用户需求，最后形成需求文字表达（需求分析说明书）的一个过程。需求分析的最终结果就是形成一份有效的需求分析说明书。本节将从系统调研方法、需求分析所需要的技术和方法，以及数据字典的形成等方面来阐述需求分析的过程。

3.2.1　系统调研过程

系统调研也称项目调研，即把系统开发当作项目来运作，其主要目的是通过接触用户以了解并最终明确用户的实际需求。这个过程是一个系统分析人员理解和掌握用户业务流程的过程，是一个需要不断与用户进行沟通和磋商的过程。系统调研方法比较灵活，因人、因系统而异。但大致的调研过程基本一样，可以分为以下几个步骤来完成。

（1）充分了解项目背景以及开发的目的。

(2) 深入用户单位(指使用该系统的机构和组织)进行调查。包括了解单位的组织结构、运作方式,了解各部门的职责和功能。然后从数据流的角度分析各个部分的特性以及它与其他部门之间的关系,如各部门的输入(输出)数据及其格式是什么,这些数据来自哪里、去向何方等,并做相应的记录。

这个步骤是调查的重点,而且难度比较大,难点在于如何建立与用户的沟通渠道。因为用户与系统分析人员一般都具有不同的技术背景,所以经常导致这种情况的出现:用户认为已经说清楚了的东西而分析人员也许对之还不能理解,或者用户提出的要求过高,超出了计算机能够处理的范围等。当出现这种情况时,需要分析人员不断地询问或说明,时间久了就会导致用户的厌倦。因此,在进行这项调查前分析人员应该做好充分的准备,例如,拟好调查方案,设计合理而简洁的调查表等。

下面给出了一种调查方法,可供读者参考和选用。

① 单位情况及其运作方式的介绍。第一次到用户单位做现场调查时,系统分析人员都应同时到场,并邀请单位的相关负责人及各部门负责人就单位情况和各部门运作方式等方面做简要的介绍,并回答系统分析人员的一些问题。目的是使系统分析人员对整个单位及其各部门的关系有一个初步的理解。

② 对部门工作职能的深入了解。在总体介绍之后,系统分析人员应分头深入各个部门进行现场观摩,请专人具体介绍,允许的话可进行跟班作业,以准确理解用户的需求。

③ 召开调查会。通过对部门的观察后,应该有一个初步的书面总结,这时分析人员也已经对部门的职责和运作方式有一个较为深入的理解。为验证这个理解的正确程度,通过询问、讨论、填调查表等方式来召开一次调查会是必要的。但这种调查会召开的次数要严格控制,一般1～2次为宜,否则会令用户厌倦。

④ 如果与用户还没有就需求达成共识,相关分析人员可有选择地重复第②和③步。

需要提醒的是,不管在什么时候,保持与用户良好的关系都是系统顺利开发的必要条件。

(3) 确定用户需求、明确系统功能和边界。综合各个分析人员的调查结果,形成系统的功能说明,确定哪些功能是系统要实现的,哪些是不应该实现的,或者是不能实现的。所有这些结果都应该跟用户确认后以书面形式确定下来。

以上调研结果是下一步设计的前提和基础,任何不准确的结果都会导致开发工作的返工,增加开发成本。所以,在系统调研时应尽最大的努力做出具有最小误差的调研结果。

3.2.2　需求分析的方法

1. SA方法

在已有调研结果的基础上,运用需求分析的各种方法,形成高质量的系统需求说明书。但这是针对整个系统的设计而言。如果仅针对数据库设计来说,那就是形成用户需求的有效表达,这种表达在说明书中多以数据流图、数据字典等形式来描述。

用户需求的表达一般是面向系统分析员的。为建立用户需求的表达,可以采用多种分析方法来完成。这些方法主要包括自顶向下和自底向上两种方法,其中常采用的方法

是自顶向下的结构化分析方法(Structured Analysis,SA)。这种方法的分析过程符合人类对问题的认识并最终解决的一般过程,其分析过程简单、实用,现已在众多领域中得到应用。其特点可以归结为一棵树的生长过程:先创建树根,然后创建树根节点的子节点,接着创建各子节点的子节点,直到创建所有的树叶节点为止。在这棵树中,树根节点相当于整个系统(第一层次上的系统),其子节点相当于第二层次上的系统,……,最后层次上的系统由叶子节点表示,它对系统分析人员是可认知的(认为已经清楚而不必再分解了)。可见,自顶向下的SA方法是从整个系统开始,采用逐层分解的方式对系统进行分析的方法。

例如,当初次面对一个复杂的系统时,只能对它有一个初步的了解,比如这个系统的功能是什么,输入和输出分别是什么;也可能认识得比这个层次稍微细一点儿,如这个系统分为几个大的子系统,每个子系统的作用及其关系,以及子系统输入/输出情况等。为寻求对整个系统的全面认识,需要对各子系统做进一步的分解,明确各个子系统的作用、系统之间的数据流向关系等。这个分解、认识的过程要持续到产生的每个子系统都能够被分析人员认知为止。基于这种分解的分析方法就是典型的自顶向下的结构化分析方法。

2. 数据流图

实际上,SA方法只是对问题分析的一种思想,在具体的分析过程中还需要借助其他的分析工具,这样才能完成对分析过程和结果的记录、对用户需求的表达等。其中,数据流图就是最为常用的辅助分析工具和描述手段。

数据流图是以图形的方式来刻画数据处理系统中信息的转变和传递过程,是对现实世界中实际系统的一种逻辑抽象表示,但又独立于具体的计算机系统。以下对数据流图常采用的符号做一个简要介绍。

1) 数据流

顾名思义,数据流是流动中的数据。所以数据流图是用有方向的曲线或直线来表示,用箭头表示数据流的方向,其旁边标以数据流的名称。其格式如图3.2所示。

图3.2 数据流格式

其中,数据流名不是随意取的,而是应该能够简要地概括数据流的含义,且易于理解。下文提到的数据名、加工名、文件名等都有同样的要求。

数据流可以来自数据的源点、加工和数据文件,可以流向数据的终点、加工和数据文件。但是,当数据取自文件或者流向文件时,相应的有向直线或曲线可以不命名,因为从相应的文件中即可知道流动的是什么数据。

2) 数据的源点和终点

显然,系统中的数据是来自系统以外的其他数据对象,其最终的流向也是系统以外的有关数据对象。这种向系统提供数据的数据对象统称为系统的**数据源点**,而系统数据所流向的数据对象则统称为**数据终点**。这两个概念的引入是为了帮助用户对系统接口界面

的理解。在数据流图中，**数据源点**和**数据终点**都是用方框来表示，方框中标以数据的名称。其格式如图 3.3 所示。

图 3.3　数据源点和数据终点格式

3）加工

加工是对数据处理的一个抽象表示。如果这种“加工”还不为系统分析员所理解，则需要 SA 方法对其进行分解，直到所得到的加工已经足够简单、不必再分时为止。这时的加工也称为**基本加工**。在数据流图中，加工是用圆圈（或椭圆）表示，圆圈中标以加工的名称。其格式如图 3.4 所示。

4）数据文件

数据文件是表示数据临时存放的地方，“加工”能够对其进行数据读取或存入。在数据流图中，数据文件通常用平行的双横线表示，旁边标以数据文件名。其格式如图 3.5 所示。

图 3.4　加工格式

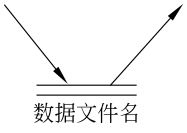
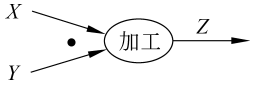
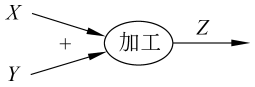
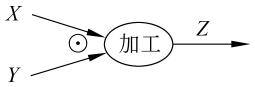
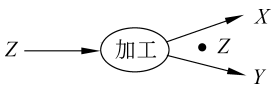


图 3.5　数据文件格式

此外，对于数据流图中的一个节点，可能有几条表示数据流的有向线出自或指向该节点。那么，该节点对数据流的影响方式（流出情况）或这几股数据流对该节点的作用方式是多种的。这种影响和作用方式说明如表 3.1 所示。

表 3.1　数据流的关系

符号表示	说　明
	表示在收到 X 和 Y 后才进行加工，然后产生 Z
	表示在收到 X 或者收到 Y 后即可进行加工，结果产生 Z
	表示从数据流 X 和 Y 中选择其中之一来进行加工，结果产生 Z
	表示在对 Z 进行加工后同时产生 X 和 Y
	表示在对 Z 进行加工后，产生 X 或者 Y，或者两者都产生

续表

符号表示	说　明
Z → 加工 ⊙ → X, Y	表示在对 Z 进行加工后,仅产生 X 和 Y 的其中之一

以上是数据流图中常用的符号,除这些外,还有其他一些符号,读者可参考有关书籍。

3. 基于数据流图的 SA 方法

自顶向下的 SA 分析方法可以与数据流图有机地结合起来,将对系统的分析过程和结果形象地表示出来。在数据流图中,SA 分析方法主要体现在对"加工"进行分解的过程。对数据流图的绘制和分解过程就是用户需求的分析及其表达的形成过程。

一个系统的数据流图是由多个子图构成的,如果加上子图之间的分解关系,就可以形成一棵树。但由于所有的子图通过表示分解关系的边连在一起而形成的树将是很庞大的,无法在同一平面中画出,所以在绘制数据流图时要分为多个子图来画。绘制的原则一般是,先绘制树根节点对应的子图,然后绘制根节点的子节点所对应的子图,一直绘制到所有叶子节点对应的子图为止。

下面以某石化集团的样品分析管理系统(一个子系统)的开发为例,介绍数据流图的基本绘制方法。

(1) 绘制根节点图。从加工粒度上看,根节点图(即根节点对应的数据流图)是最大的数据流图,它是将整个应用系统当作一个加工。例如,对于样品分析管理系统(YPFXMS),其根节点子图如图 3.6 所示。

图 3.6　YPFXMS 的根节点数据流图

(2) 绘制子节点图。对系统开发来说,数据流图的根节点不提供任何有用的信息,需要对加工"样品分析管理系统"做进一步的分解。在调研中发现,不是每一种样品都需要分析,能够分析的是那些已经有计算公式或分析方案的样品,而且送样(被送用于分析的样品)要先存在样品分析员那里,样品分析员则按照某一个原则一次对这些送样进行分析。在送样被分析后,还需要对分析的结果进行检查,如果发现分析结果不合格(是指分析手段和方法出错),则返回给样品分析员重新进行分析,分析合格则返回给送样人员。于是,分解后得到如图 3.7 所示的数据流图。

图 3.7 中有 3 个节点,分别以"1""2""3"标记。如果某一个节点对应的加工内部还包含数据流,则对该加工进一步分解,直到不能再分而形成基本加工为止。例如,节点"2"对应的加工"样品分析实验"还可再分解,从而可以按照上述方法绘制相应的数据流图。这时数据流图中所有的加工节点都以"2."开头,如"2.1""2.2"等,表明这些加工是由加工"2"分解得到的。这样能使得数据流图层次清楚,一目了然。

当一个系统的数据流图绘制完了以后,应该和用户做进一步的交流,以确认数据流图

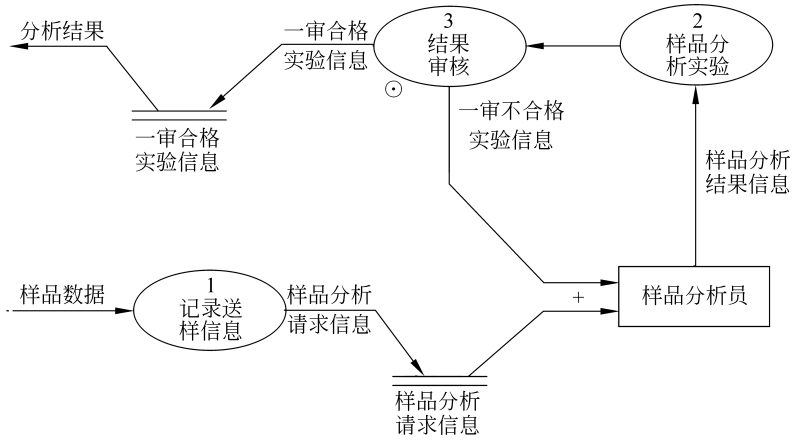


图 3.7　根节点 0 的子节点数据流图

是能够充分表达用户需求的。

3.2.3　形成数据字典

数据流图主要是表示数据和处理之间的关系，但缺乏对数据流、数据文件、加工等图中各个元素进行描述的能力。实际上，数据流图是将用户头脑中的需要转换为机器能够接受的表达的一个中转站，但数据流图表示的信息离机器能够接受的信息还比较远。如果把用户需求和机器表示放在两头，数据流图放在两者之间，那么数据流图更靠近用户需求一些，而相对远离机器表示。为此，需要引入数据字典的概念，通过数据字典可以加强数据流图的信息表达能力，同时这种表达拉近了与机器表示的距离，使得用户需求从纯粹的逻辑表达逐步转向机器表示，为数据库的实施奠定基础。

与数据流图一样，数据字典也是 SA 方法中一种有力的工具。通常情况下，数据字典与数据流图结合使用，主要是用于对数据流图中出现的各种元素进行描述，给出所有数据元素的逻辑定义。简而言之，数据字典是数据流图中数据元素的描述。这种描述是由一系列的条目组成，但不同的应用、不同的系统其组成的条目可能有所不同。一般来说，至少应该包括数据流、数据文件、加工和数据项 4 种条目，这 4 种条目的组成格式有较大的差别。现对这些条目及其格式说明如下。

1. 数据项条目

数据项是数据构成的最小组成单位，它不能再分割。数据项条目用于说明数据项的名称、类型、长度、取值范围等。例如，在课题管理系统中数据项“课题申请代码”条目可描述如下。

数据项名：课题申请代码。

类型：字符型。

长度：12。

取值范围：000000000000～999999999999。

取值说明：前 4 位为年份，第 5 和 6 位、第 7 和 8 位分别表示月份和日期，后 4 位表示

当天的课题序号。

2. 数据流条目

数据流条目主要用于说明数据流的组成(由哪些数据项组成),数据流的来源和流向以及数据流量等信息。例如,数据流"样品分析请求信息"条目描述如下。

数据流名称:样品分析请求信息。

组成:申请表编号、申请表名称、分析项目代码、样品编号、样品名称、送样日期、送样人员。

来源:记录送样信息(加工)。

去向:样品分析请求信息(文件)。

流量:10～20个每天。

3. 数据文件条目

数据文件条目用于说明数据文件是由哪些数据项组成、组织方式、存储频率如何等信息。例如,数据文件"一审合格实验信息"条目如下。

文件名:一审合格实验信息。

数据组成:实验记录表编号、实验记录表名称、实验日期、实验环境、实验目的、操作人员、原料规格、实验配方与工艺、操作过程与现象、实验结果与讨论、记录人员、实验组长、课题代码。

组织方式:按实验记录表编号递增排列。

存储频率:1次/天。

4. 加工条目

加工条目主要用于说明加工的逻辑功能,指明输入数据和输出数据等信息。其中,逻辑功能项用于指出该加工用来做什么、对加工处理的一些要求等。例如:

加工编号:1。

加 工 名:记录送样信息。

输入数据:样品数据。

输出数据:样品分析请求信息。

逻辑功能:对送检的样品数据进行登记,并由此转换成样品分析请求信息。

数据字典是对系统调研所收集的大量数据进行详细分析所得到的主要结果。它是数据库设计阶段的一项主要成果,因而它的创建是一项重要的工作,但同时也是一项费时的工作。

对数据字典的维护也是一项艰巨的任务。对于中型规模以上的系统来说,数据字典都是比较庞大的,可能包含成千上万个条目。在数据字典中需要保持这些条目的准确性、一致性,要按照一定的顺序来排列这些条目,以方便查找。这些工作量是非常可观的。幸运的是,现在有很多用于创建和维护数据字典的软件,使人们的工作量大大地减少。但是,软件是代替不了人的,创建和维护的许多工作还需要人工的参与。因此,深入理解数据字典的形成过程及其维护方法,都是学习数据库设计的主要任务。

3.3　数据库结构设计

在需求分析后，将形成系统的数据流图和数据字典。在此基础上，可以对数据库的结构进行设计。数据库结构包括概念结构设计、逻辑结构设计和物理结构设计3个部分。

3.3.1　概念结构设计

1. 概念结构及其设计思想

需求分析的成果是数据流图和数据字典，这是对用户需求在现实世界中的一次抽象。但这种抽象还只是停留在现实世界当中，而概念结构设计的目的就是将这种抽象转换为信息世界中基于信息结构表示的数据结构——概念结构，即概念结构是用户需求在信息世界中的模型。

数据库的概念结构独立于它的逻辑结构，更与数据库的物理结构无关。它是现实世界中用户需求与机器世界中机器表示之间的中转站。它既有易于用户理解、实现分析员与用户交流的优点，也有易于转换为机器表示的特点。当用户的需求发生改变时，概念结构很容易做出相应的调整。所以，概念结构设计是数据库设计的一个重要步骤。

概念模式描述的经典工具是E-R图，由E-R图表示的概念模型就是所谓的E-R模型。E-R模型的创建和设计过程就是概念结构的创建和设计过程，所以概念结构的设计集中体现为E-R模型的设计。

E-R模型的优点主要体现在：它具有较强的表达能力，可以充分表示各种类型数据及数据之间的联系；数据表达形式简单，没有过多的概念，定义严格，无二义性等；E-R模型是以图形的形式出现，表示直观。

E-R图的基本画法在第1章中已经进行了介绍，那么如何基于E-R图进行概念结构设计呢？实际上，对于概念结构的设计，人们已经总结了以下4种设计指导思想。

（1）自顶向下：首先根据用户需求定义全局概念结构的E-R模型，然后对其分解，逐步细化。

（2）自底向上：首先根据各个部门的需求定义局部概念结构的E-R模型，然后将这些局部的E-R模型拼接成为全局的E-R模型，从而形成全局概念结构。

（3）先主后次：分析各种子需要的“轻重”，首先设计最重要的概念结构，形成它的E-R模型，然后定义次要概念结构的E-R模型，接着按照类似的方法定义其他所有概念结构的E-R模型，最后将这些模型继承起来，形成全局概念结构。

（4）上下混合：这是指将自顶向下和自底向上这两种方法结合起来使用的一种设计方法。

在实际应用当中，概念结构设计通常采用的是自底向上的设计方法（而需求分析一般是采用自顶向下的方法），即这种方法分为两步：①先建立局部概念结构的E-R模型；②然后将所有的局部E-R模型集成起来形成全局概念结构。以下主要介绍这种自底向上的概念结构设计方法。

2. 局部 E-R 模型的设计

E-R 模型的设计是基于需求分析阶段产生的数据流图和数据字典来进行的。一个系统的数据流图是按分层来绘制,是由多张数据流图构成。基于一张数据流图及其对应的数据字典部分进行的 E-R 模型设计得到的是一个局部概念。所以,采用从局部到全局的概念结构设计方法也就是很自然的事了。

局部应用所涉及的数据都已经收集在数据字典中,但如何从中进一步抽象出系统的实体以及实体间的联系却是基于局部数据流图进行 E-R 模型设计的主要困难。实体和实体间联系的划分并无统一的标准,一般采用的划分原则是:先凭经验,后做调整。

经验是指在一般情况下对于那些具有共同特征和行为的对象,可以将其抽象为实体;对象的共同特征和行为可以抽象为实体的属性。调整是指在凭经验做出抽象后,根据具体的应用和建模环境对实体与其属性之间的关系以及实体与实体之间的关系做出相应的更改,有可能使得原来的属性变为实体,原来是实体的变为属性等,从而也导致了实体间关系的改变。

对于实体及实体间关系的抽象还应注意以下三点:①在同一应用环境中,被抽象为属性的事物就不能再被抽象为实体了,否则会导致"属性又包含属性"的错误,这违反第一范式;②属性具有不可再分性,所以具有不可再分性的事物一般都应抽象为属性,而具有可再分性的事物一般不能抽象为属性;③一个事物不能同时被抽象为两个实体的属性,即一个属性只能隶属于一个实体。

例如,对于一个企业信息管理系统来说,企业中的工作人员可以抽象为"职工"实体,而工作人员的姓名、性别、年龄、职称、部门等可以抽象为"职工"实体的属性,如图 3.8 所示。

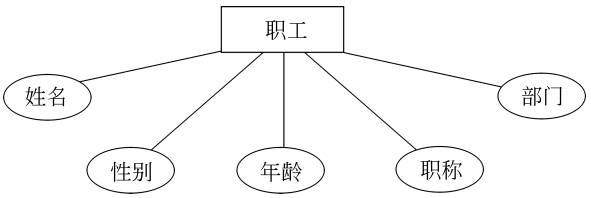


图 3.8 "职工"实体属性图

这是凭普遍经验做出的抽象(第一感觉就认为应该做出这样的抽象),但这可能不正确或不全面,需要做进一步的调整。例如,如果在这个系统中除了"职工"信息以外还需要考虑部门的一些信息,如部门的名称、人数、经理、位置等信息,那就应该对前面的抽象做调整:应该将"部门"由原来作为"职工"的属性改为一个新的实体——"部门"实体,同时原来作为属性的"部门"被删除(这避免了一个事物既作为属性又作为实体的情况出现)。这两个实体的关系是"部门"拥有"职工"(或者"职工"隶属于"部门"),即"拥有"是这两个实体之间的联系。这样,原来的 E-R 图进一步得到调整和扩充,变为如图 3.9 所示。

假设从需求分析中还发现,部门中的职工所做的工作是开发项目,用于说明项目的信息包括项目名称、项目性质、项目开发的起始时间、项目经费、项目经理等。于是,将项目抽象为实体,形成"项目"实体,如图 3.10 所示。

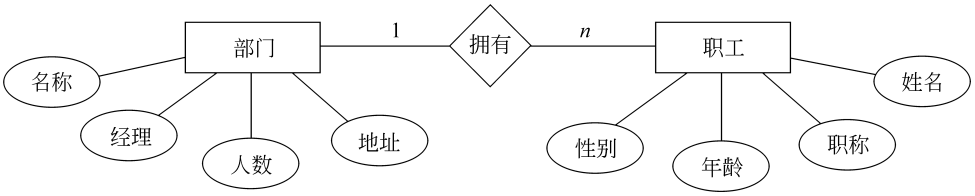


图 3.9　“部门”和“职工”的 E-R 图

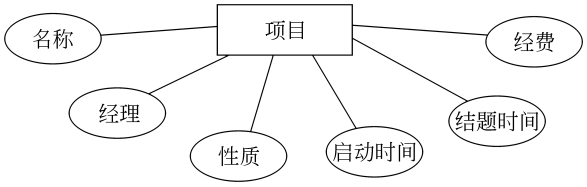


图 3.10　“项目”实体属性图

3. 全局 E-R 模型的集成

显然，相对于一个整体而言，以上画出的 E-R 图是局部的，这些 E-R 图应该能够合成一张总的 E-R 图。对概念结构来说，就是将局部概念结构集成为全局的概念结构。

例如，根据需求分析结果可以进一步发现，在这个企业中每个部门都有承接多个项目的可能，每个职工只参加一个项目。于是，将如图 3.9 和图 3.10 所示的 E-R 图拼接起来，结果得到了整个系统的 E-R 图，如图 3.11 所示。

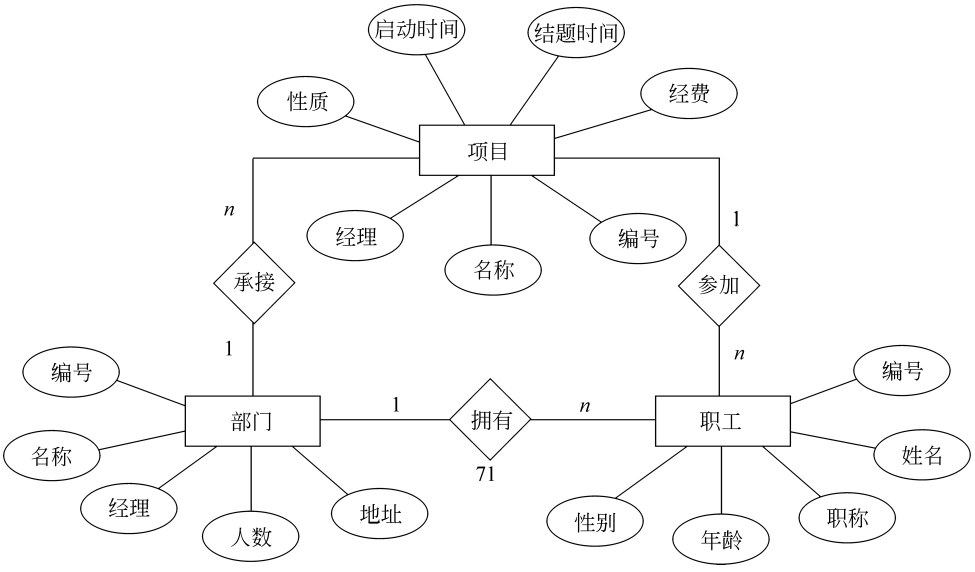


图 3.11　企业管理信息系统的 E-R 图

从以上 E-R 图的创建过程中可以看出，这种创建方法基本上是按照先局部后全局的自底向上的设计思想来实现的，E-R 图的创建过程是一个不断修正的反复过程。当然，实

际系统不可能这么简单,但从这个设计过程中读者可以体会到一个系统 E-R 图设计的一般方法,由此可以融会贯通,举一反三。

一般来说,局部 E-R 图可能是由多人设计的,即使是由同一个人设计,但由于设计是在不同的时间、不同的条件下完成的,这都有可能造成各个局部 E-R 图的不一致,从而使得在局部 E-R 图的拼接过程中产生许多问题。这些问题主要体现为各个局部 E-R 图之间的冲突,其中包括命名冲突、属性冲突和结构冲突等。

1) 命名冲突

命名冲突是指意义不同的元素在不同的局部 E-R 图中有相同的名字,或者是有相同意义的元素在不同的局部 E-R 图中具有不相同的名字。名字冲突的解决比较容易,只要开发人员进行充分的协商,制定统一的命名规则即可解决。

2) 属性冲突

属性冲突是指同义同名的属性在不同的局部 E-R 图中的取值类型、范围、所使用的单位等却完全不一样。例如,有的将职工编号定义为长度为 12B 的字符串类型,有的则定义为长度为 8B 的字符串类型,有的可能将职工编号定义为整型,等等。一般来说,在一个 E-R 图中不应该存在属性冲突。这种冲突也主要是通过协商、加强沟通来解决的。

3) 结构冲突

结构冲突是指一个事物在一个局部 E-R 图中被抽象为实体,而在另一个局部 E-R 图中又被抽象为属性。这时不能直接将这两个局部 E-R 图拼接为一个 E-R 图,首先要解决结构冲突问题。解决的办法是,视具体情况将相应的属性改为实体,或者将相应的实体改为属性。但这种操作又可能引起别的问题,更改时要慎重。

还有一种结构冲突是相同的实体在不同的局部 E-R 图中有不同的属性或不同的联系。对于前一种情况,一种简单的解决方法是:使该实体的属性集为它在各 E-R 图中的属性集的并;对于后一种情况,解决方法相对复杂,要视具体情况对联系进行分解,或者进行其他的调整。

另外,在构建的 E-R 图中,最好不要包含环形结构,因为这容易出现“死循环”参照关系。例如,假设实体 X、Y 和 Z 以及它们之间的联系 a、b 和 c 构成如图 3.12 所示的 E-R 图,则该 E-R 图出现“死循环”问题,因为在据此图建立数据表时,将出现 X 参照 Y、Y 参照 Z、Z 参照 X 的“死循环”参照关系,从而无法创建数据表。因此,如果一个 E-R 图包含环形结构,则需要进一步确认对概念结构的建模是否正确,重新修改 E-R 图,或者直接将环中的一条边(关联)去掉,以破坏“死循环”结构。

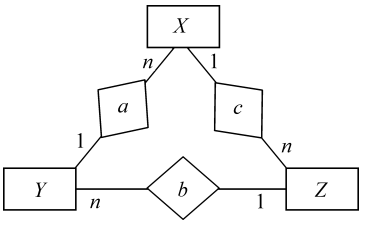


图 3.12 带环结构的 E-R 图

显然,读者可能注意到,如图 3.10 所示的 E-R 图就包含环结构。但该环结构并非是“死循环”结构,因此不会出现无法建表的情况,其表达的实体参照关系是正确的。

解决结构冲突所需要的工作量比较大,解决过程比较烦琐。所以在设计局部 E-R 图前,分析人员应尽可能地加强沟通,达成较为全面的共识,尽量避免出现结构冲突的问题。

3.3.2　逻辑结构设计

数据库的逻辑结构设计就是将以 E-R 图表示的概念结构转换为 DBMS 支持的数据模型并对其进行优化的过程。如今，绝大多数的数据库都是关系数据库，所以在此主要介绍概念结构到关系模型的转换方法和相关技术。

概念结构是由 E-R 图来描述的，所以这种转换问题可以归结为 E-R 图到关系模型的转换问题。E-R 图的基本元素是实体、属性和联系等，于是 E-R 图到关系模型的转换就变成了实体、属性和联系等基本元素到关系模式的转换问题了。

1. 实体和属性的转变

这种转变比较简单、直观，即一个实体转换为一个关系模式，其中实体名变成了关系模式的名称，实体属性相应地变成了关系的属性。例如，图 3.11 中的三个实体分别转换为以下三个关系模式。

项目(编号，名称，经理，性质，启动时间，结题时间，经费)
部门(编号，名称，经理，人数，地址)
职工(编号，姓名，性别，职称，年龄)

2. 1∶1 联系的转变

一对一联系的转变比较简单。其中，最直观和最简单的方法就是创建一个独立的关系模式，该关系模式的属性是由该联系本身的属性以及与之相连的实体的候选码(每个实体中取一个候选码)组成。例如，一个仓库仅由一个仓库管理员管理，而一个仓库管理员也只能管理一个仓库，所以仓库管理员和仓库之间的联系是管理，管理时间为 8h(一天)，其 E-R 图如图 3.13 所示。

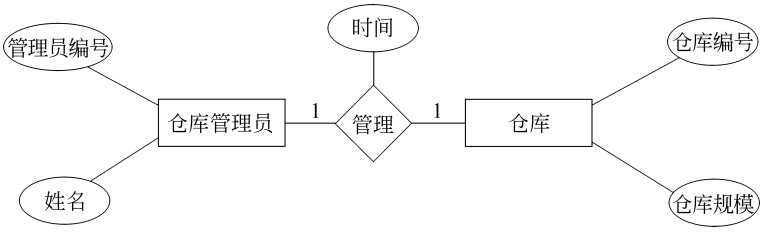


图 3.13　“仓库管理员”和“仓库”及其联系的 E-R 图

可见，“仓库管理员”和“仓库”之间的联系是 1∶1，该联系转换后形成如下的关系模式。

管理(管理员编号，仓库编号，时间)

其中，管理员编号和仓库编号分别为“仓库管理员”实体和“仓库”实体的候选码，时间是“管理”联系的属性。

但是，有时为了减少数据冗余或者其他原因，也可以将联系对应的关系合并到与之相连的某一个实体对应的关系中去。合并的方法是，将一个实体的候选码以及联系的属性添加到另一个实体对应的关系中。例如，对于以上例子，易知“仓库管理员”实体对应的关

系如下。

仓库管理员(管理员编号,姓名)

这时,只需将“仓库”实体的候选码“仓库编号”以及联系的属性“时间”一起添加到仓库管理员关系中即可,从而实现对联系“管理”的转换,结果得到的关系模式如下。

仓库管理员(管理员编号,姓名,仓库编号,时间)

当然,也可以将“仓库管理员”的候选码和联系的属性“时间”一起添加到仓库关系中,结果得到如下的关系模式。

仓库(仓库编号,仓库规模,管理员编号,时间)

一般来说,联系可以合并到任意与之相连的实体对应的关系中。但在实际应用中,往往从效率的角度来考虑如何进行合并。例如,对于上面的例子,如果查询仓库关系比查询仓库管理员关系要频繁得多,则应将联系合并到仓库管理员关系中去,这样可以提高整个系统的效率。

3. 1∶*n* 联系的转变

与一对一联系类似,一对多联系可以转换为一个独立的关系模式,也可以将联系合并到 n 端对应的关系模式中。例如,对于如图 3.11 所示的 E-R 图,“拥有”联系是一对多联系,当把这个联系转换为独立的关系模式时,则得到如下的关系模式。

拥有(职工.编号,部门.编号)

如果用合并的方法对该联系进行转换,则得到如下的关系模式。

职工(职工.编号,姓名,性别,年龄,职称,部门.编号)

其中,以上的“编号”属性都是相应实体的主码。

4. *m*∶*n* 联系的转变

多对多联系只能转换为一个独立的关系模式,其属性集是由与该联系相连的实体的属性(码)以及该联系本身的属性转换而得到的。例如,一个仓库可以存放多种零件,一种零件也可以存放在多个仓库中,可见仓库和零件之间的联系——“存放”是(m ∶ n)联系。假设其 E-R 图如图 3.14 所示。

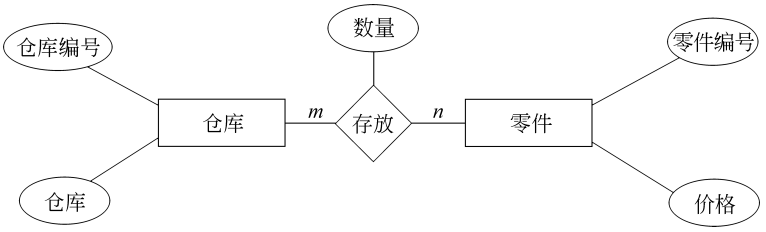


图 3.14 “仓库”和“零件”及其联系的 E-R 图

那么,“存放”联系转换为独立的关系模式后,结果如下。

存放(仓库编号,零件编号,数量)

以上讨论的是与两个实体相连的联系(称为二元联系)的转换问题。对于多元联系的转换问题,不难从二元联系转换方法的推广中获得解决,不再赘述。

5. 应用规范化理论实现逻辑结构的优化

逻辑结构设计的结果是数据模型。以上主要介绍了如何将以 E-R 图表示的概念结构转换为以关系模型表示的逻辑结构。在形成关系模式后,还需要对其进行优化处理,以尽可能地减少数据冗余、删除冲突和插入冲突等问题。对关系模式的优化处理主要是基于规范化理论进行的,具体操作方法可参见第 2 章的相关内容。

6. 用户子模式的设计

用户子模式也称为外模式,它是面向用户的,是用户可见的数据模型部分。它可以屏蔽概念模式,有助于实现程序与数据的独立,可以满足不同用户对数据的个性化需求,同时也有利于数据库的管理。

用户子模式的设计主要是利用局部 E-R 图,因为每一张 E-R 图一般都是表示局部概念结构。现在流行的 DBMS 一般都提供了视图功能,支持用户的虚拟视图。可以利用这个功能设计符合不同局部应用需要的用户子模式。

3.3.3　物理结构设计

在逻辑结构设计阶段得到的数据模型只是一个理论上的概念,与具体的计算机系统无关。但是,基于数据模型的数据库最终还是必须存放到某个具体的物理设备中,由特定的 DBMS 来管理。所以,如何选取合理的存储结构和有效的存储路径,以充分利用系统资源、提高数据库的性能,这就是物理结构设计需要完成的任务。

简而言之,物理结构设计就是为既定的数据模型选取特定的、有效的存储结构和存储路径的过程。特定性是指跟具体的计算机系统有关,包括操作系统和 DBMS 等;有效性是指以尽可能少的系统资源获取数据库尽可能高的运行效率。可以看出,物理结构设计的内容主要包括数据库存储结构的确定和数据库存取方法的确定。

1. 数据库存储结构的设计

数据库存储结构设计的任务是确定数据的存放位置和使用的存储结构,具体讲,确定如何在磁盘空间中存储关系、索引、日志、备份等数据库文件,以及如何设置系统存储参数,目的是使得以最小的系统资源获取最高的系统性能。

存储结构的设计是在已选定的 DBMS 和硬件条件下进行的,主要从以下两个方面考虑。

1) 确定数据的存放方式

在大多的关系 DBMS 中,数据的分类和指定存储是通过数据文件的划分和存储来实现的。因为在 DBMS 中,不能直接指定数据的存放位置,而只能通过一定的机制实现数据文件的指定存放,从而实现将数据存放在指定的位置。所以,确定了数据文件的存放位置也就确定了数据的存放位置。

数据文件的划分和存储主要是基于数据访问的稳定性、安全性、效率等方面考虑的,相应的指导性规则包括:

(1) 数据库文件和日志文件应该分开存放在磁盘中。

(2) 如果计算机系统中有多个磁盘,可以将数据库文件分为多个文件,并分布在不同的磁盘中。

(3) 将数据表和索引等分开存放在不同的数据库文件中。

(4) 大的数据对象要分散存储在不同的数据库文件中。

以上操作,对不同的 DBMS 有不同的操作方式,但大多都提供这些操作功能。而对设计人员来说,他必须熟悉 DBMS 提供的这些功能及其操作方法。

2) 确定系统参数的配置

系统参数是指 DBMS 提供的设置参数。这些参数主要包括数据库的大小、同时连接的用户数、缓冲区个数和大小、索引文件的大小、填充因子等。DBMS 一般都对这些参数设置了初始值,但这些设置并不一定适应每一种应用环境,这需要设计人员重新设计。

这些参数的配置操作一般都可以在 DBMS 提供管理工具中完成。例如,SQL Server 2008 提供的管理工具是 SQL Server Management Studio(SSMS),SSMS 可以管理 SQL Server 2008 的所有组件,包括访问、配置、控制和开发这些组件。

2. 确定存取方法

存取方法即关系模式的存取方法,其目的是实现数据的快速存取。每一种 DBMS 都提供了多种不同的存取方法,其中索引法是最常用的一种,在实际开发当中用得最多。以下重点介绍这种存取方法。

索引为什么可以提高数据库中数据的存取速度呢? 这个道理与目录可以提高书的查阅速度的道理一样,即可以将索引形象地比喻为目录。实际上,索引正是基于目录的原理来设计的,它是“数据标题”和数据内存地址的列表。通过索引可以从部分数据检索中实现数据的快速查找,从而提高数据的查询效率。

但是索引的创建并不是无代价的。索引本身也是一种数据表,同样占用存储资源,而且要保持与数据表的同步,这要求在进行数据更新操作(包括添加、删除和修改操作)时也要对索引进行相应的更新操作。如果索引很大时,其占用的空间资源以及对其更新维护所需要付出的代价同样是非常可观的。所以对索引的创建与否,应该慎重考虑。

在创建索引时,要考虑以下几个经验性的指导原则。

(1) 在经常用于检索的列上创建索引,特别是要对主码创建索引(一般是由 DBMS 自动完成)。

(2) 在外键上创建索引,因为它经常用于与其他关系进行连接查询。

(3) 多在以读为主或者经常需要排列的列上创建索引。因为索引已经排序,它可以加快读取速度和排序效率。

(4) 多在经常用于条件查询的列上创建索引,特别是对那些常常出现少量元组满足条件的列。

而对具有以下性质的列,则不宜对其创建索引。

(1) 对于不经常用于检索的列,则不宜在其上创建索引。因为其上有无索引是无关紧要的,创建了反而浪费存储空间。

(2) 对于那些值域很小的列不应该创建索引。例如,不宜在“性别”列上创建索引,因

为“性别”只有两个值：“男”和“女”，索引对这种列并无作用。

(3) 对于值域严重分布不均匀的列不宜在其上创建索引。

(4) 对于更新操作非常频繁的列，不宜在其上创建索引。因为在进行更新操作时，不但要更新数据表的内容，而且还要更新索引表中的索引项，这会降低系统的效率。

(5) 对于长度超过30B的列，一般不要在其上创建索引。因为在过长的列上创建索引，索引所占的存储空间就大，索引级数也随之增加，从而会消耗系统资源、降低系统效率。如果非要创建不可，最好能够采取索引属性压缩措施。

3.4　数据库的实施、运行和维护

在数据库的逻辑结构和物理结构设计完了以后，就可以将这些设计结果付诸实践，并创建相应应用程序，形成实际可运行的系统。系统运行之后还需要对其进行日常维护。这些工作就是数据库实施、运行和维护要讨论的内容。

3.4.1　数据库实施

数据库的实施是在数据库逻辑设计和物理设计的基础之上进行的。与前面设计不同的是，数据库实施后将形成一种能够实际运行的系统，这种系统是将前面设计结果付之实践而形成的，是动态的；而前面的设计(包括需求分析、概念、逻辑设计和物理设计等)是在纸上进行的，停留在文档阶段，是静态的，但它们是数据库实施的基础，是数据库系统能够稳定、高效运行的前提。

数据库实施包括以下几方面的内容。

1. 建立数据库结构

根据物理结构的设计结果，选定相应的DBMS。然后在该DBMS中利用其提供的DDL建立数据库结构。

例如，在SQL Server中，可用下列的SQL语句来分别创建数据库和数据表。

```
CREATE DATABASE database_name
CREATE TABLE table_name
```

关于它们的使用格式，将在第4章中详细介绍。此外，在SQL Server中，还为数据库结构的创建提供了可视化的图形界面操作方法，极大地提高工作效率。

在数据库结构定义以后，通过DBMS提供的编译处理程序编译后即可形成实际可运行的数据库。但这时的数据库还仅仅是一个框架，内容是空的。要真正发挥它的作用，还需要编写相应的应用程序，将数据保存在其中，形成一个“有血有肉”的动态系统。

2. 装载测试数据，编写调试应用程序

应用程序设计与数据库设计可以同时进行，但应用程序的代码编写和调试则是在数据库结构创建以后进行的。应用程序的设计、编写同样是一个复杂的过程，相关的设计技术可参见软件工程方面的书籍。

应用程序的编写和调试是一个反复进行的过程，其中需要对数据库进行测试性访问。

所以,这时应该在数据库中装载一些测试数据。这些数据可以随机产生,也可以用实际数据作为测试数据(但这些实际数据要留有副本)。但不管用什么产生方法,使用的测试数据都应该能充分反映实际应用中的各种情况,以使得可以充分测试应用程序是否符合实际应用的要求。

3. 试运行

在应用程序调试完了以后,给数据库加载一些实际数据并运行应用程序,但还没有正式投入使用,而只是想查看数据库应用系统各方面的功能,那么这种运行就称为试运行。试运行也称联合调试,与调试的目的基本一样,但侧重点有所不同。调试主要是为了发现系统中可能存在的错误,以便及时纠正;试运行虽然也需要发现错误,但它更注重于系统性能的检测和评价。所以,试运行的主要工作如下。

(1) 系统性能检测。包括测试系统的稳定性、安全性和效率等方面的指标,查看是否符合设计时设定的目标。

(2) 系统功能检测。运行系统,按各个功能模块逐项检测,检查系统的各个功能模块是否能够完成既定的功能。

如果检测结果不符合设计目标,则返回相应的设计阶段,重新修改程序代码或数据库结构,直到满足要求为止。如果不符合要求就强行投入使用,可能会产生意想不到的灾难性后果。对此,用户和开发方都应该慎重考虑。

总之,试运行是系统交付使用的最后一道“门槛”,能否在这一关中正确而充分地检测一个系统对以后的正式运行有着非常重要的意义。

3.4.2 数据库系统的运行和维护

试运行结束并被证实符合设计要求后,数据库就可以正式投入使用。数据库的正式使用标志着数据库开发阶段的基本结束,同时意味着数据库运行和维护阶段的开始。数据库的运行和维护并不是数据库设计的终点,而是数据库设计的延续和提高。

数据库的日常运行和维护也是一项专业性很强的工作,需要很强的专业技术。维护工作不是普通的用户就能够胜任的,一般是由系统管理员(DBA)完成。这种工作就是软件产品的售后服务。

在数据库的运行和维护阶段,DBA 的主要工作如下。

1. 数据库的转储和恢复

一旦数据库正式投入使用,企业的相关数据将全部存入数据库(一般不会另记在纸质材料中)。如果数据库发生故障,可能会导致这些数据的丢失,从而造成企业的重大损失。所以,为了尽量避免在数据库发生故障时造成数据丢失,DBA 应当根据应用的具体要求指定相应的备份和恢复方案,保证一旦发生故障,能够尽快将数据库恢复到某一种一致性的最近状态,尽量减少损失。

数据库转储正是为了解决上述问题而提出一种数据库恢复技术,它是指定期地把整个数据库复制到磁盘或者其他存储设备上保护起来的过程。实际上,数据库的转储和恢复是数据库运行维护中最重要的工作之一。

2. 数据库性能的检测、分析和改善

随着运行时间的增加，数据库的物理存储不断发生改变，加上数据量和用户的不断增加，这都使得数据库的运行性能不断下降。为此，DBA 必须利用 DBMS 提供的性能监控和分析工具定期地对数据库的各种性能指标进行检测，以便及早地发现问题并采取相应的优化和改善措施。

3. 数据库的安全性和完整性维护

不管是从企业内部还是从企业外部来讲，数据库的安全性和完整性都是至关重要的。作为数据库的管理者，DBA 必须对数据库的安全性和完整性负责。所以，DBA 应该认真审核每一个用户的身份，并正确授予相应的权限；随着时间的推移和应用环境的改变，对安全性的要求也随之发生变化，这要求 DBA 对数据库的安全性控制做出相应的调整，以适应新的情况。类似地，数据库的完整性约束条件也会发生变化，这同样要求 DBA 做出相应的修正，以满足新的要求。

4. 数据库的重组和重构

数据的插入、修改和删除是数据库的基本操作。这些操作的多次使用会使得数据在磁盘上的存储分布越来越散，导致数据的存储效率降低，整个系统性能下降。这时应该对数据库进行重新组织（即重组），以提高系统的性能。现在流行的 DBMS 一般都提供重组功能。

随着应用发展的需要，用户可能要求增加某些属性或实体，也可能要求删除某些属性或实体，或者要求修改某些实体之间的联系等。为满足这种要求，需要对数据库的模式和内模式进行调整，如增加或删除某些列和表、增加或删除某些索引、修改数据库的完整性约束条件等，这种调整就是对数据库进行重新构造的过程，即数据库的重构。现在流行的 DBMS 也提供数据库重构功能。

数据库重组和数据库重构有着本质的区别，这主要体现在：数据库重组的目的是为了提高系统的性能，它通过 DBMS 提供的功能对数据库在磁盘上的存储分布进行调整来达到重组的目的，重组不会改变数据库的模式和内模式；数据库重构的目的则是为了实现新的用户需求，它需要修改数据库结构，从而使得数据库的概念模式和内模式也被修改。

数据库重构不但使数据库结构发生了改变，而且在多数情况下也要求应用程序做出相应的修改。这会导致“牵一发而动全身”的后果，所以由数据库重构而引起的修改工作量是非常大的。因此，不是在迫不得已的情况下，请不要使用数据库重构。

虽然数据库重构可以实现新的用户需求，但这种需求的变化幅度必须限制在一定的范围内。如果超过这个范围，数据库重构可能无法实现，也可能是实现的代价太高而失去重构的意义。所以数据库重构并不是“无所不能”的。如果在一个数据库系统中无法进行数据库构成，则表明这个数据库系统已经被淘汰了，需要设计一个新的系统来取代它。

习　　题

一、简答题

1. 数据库设计主要分为哪几个步骤？每个设计步骤的主要目的以及获得的结果是什么？

2. 数据库结构设计包含哪几个部分?

3. 什么是 E-R 图? 它在数据库设计中有何作用?

4. 需求分析主要采用什么方法?

5. 什么是概念结构? 其设计思想是什么? 有哪些特点?

6. E-R 模型的集成需要注意什么问题?

7. 简述数据字典的结构及其作用。

8. 什么是逻辑结构设计?

二、设计题

1. 已知系统 a 的局部 E-R 图(概念结构)如图 3.15 所示。

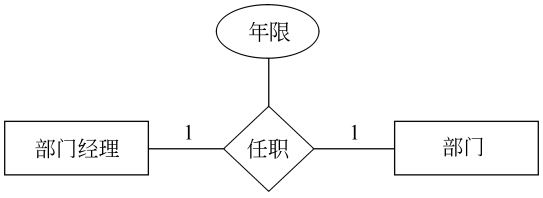


图 3.15 系统 a 的局部 E-R 图

其中,各实体的属性说明如下(为了简化 E-R 图,属性没有在图中标出)。

部门:部门编号, 名称, 地址, 人数
部门经理:工号, 姓名, 性别, 职称, 年龄

请将该 E-R 图表示的概念结构转换为相应的关系模式(逻辑结构)。

2. 已知系统 b 的局部 E-R 图(概念结构)如图 3.16 所示,请据此给出它合理的关系模式(逻辑结构)。

图 3.16 系统 b 的局部 E-R 图

其中,各实体的属性如下:

学院:学院代号, 名称,年科研经费, 专业数, 教师人数
班级:班级代号, 名称, 专业, 人数
学生:学号, 姓名, 性别, 专业, 籍贯

3. 已知系统 c 的局部 E-R 图如图 3.17 所示,请据此给出它合理的关系模式。

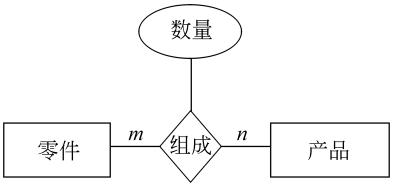


图 3.17 系统 c 的局部 E-R 图