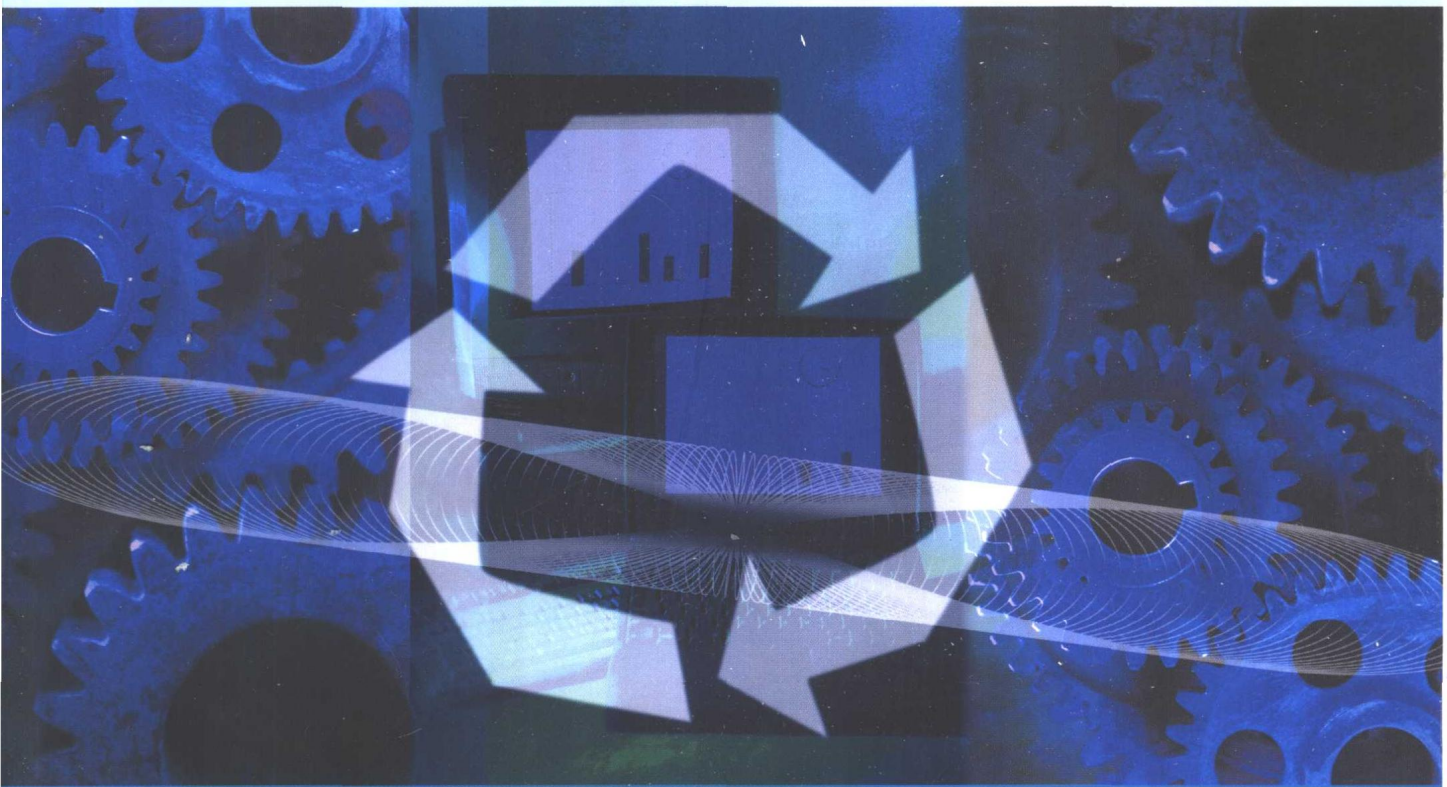
阿 惟 编著

Visual Basic.NET 进销存程序设计



在无法阻挡的.NET的潮流前，想利用Visual Basic .NET设计数据库系统，想要找一本能带您从无到有实现可以真正联机使用的商用数据库应用系统的书，本书将是您的最佳选择。

精彩主题：

- 系统分析
- ADO .NET开发技术
- 使用Visual Basic .NET设计符合ADO .NET数据访问对象结构的数据库应用程序
- 使用Crystal Report设计精美的报表
- 零售业进销存管理系统的实现，包括进货、出货、库存管理、应收账款、应付账款的管理……

清华大学出版社

Visual Basic .NET
进销存程序设计

阿 惟 编著

清 华 大 学 出 版 社

内 容 简 介

Visual Basic .NET，在众多 Windows 应用程序开发工具中，可以算是普及率相当高的程序开发工具之一。要利用 Visual Basic .NET 来开发符合 Microsoft 最新标准的数据存取对象结构——ADO.NET 的数据库应用系统，在寻觅开发工具的时候可以列为最佳选择。

本书除了讨论 Visual Basic .NET 在数据库应用系统的设计方式之外，更以一般中小企业最为广泛使用的数据库应用系统结构——客户机/服务器结构作为讨论的主题。除了深入讨论它的设计方法、编写技巧外，更详细地说明在开发数据库应用系统的过程中，程序设计人员所不能忽略的细节。此外，更以进销存管理系统作为本书的专题，理论与实际相结合，让读者快速掌握 Visual Basic .NET 开发客户机/服务器结构应用程序的精髓。

本书没有艰涩的程序设计技巧和深奥的技术讨论，只要稍具 Visual Basic 基础，都能在本书有计划的指引下，设计出令人刮目相看的优良的数据库管理系统。本书适合进行中小企业数据库应用系统程序设计的程序开发人员使用或参考。

图书在版编目（CIP）数据

Visual Basic .NET 进销存程序设计/阿惟编著.—北京：清华大学出版社，2003

ISBN 7-302-06731-7

Ⅰ.V…Ⅱ.阿…　Ⅲ.BASIC 语言—程序设计　Ⅳ.TP312

中国版本图书馆 CIP 数据核字（2003）第 045578 号

出 版 者：清华大学出版社(北京清华大学学研大厦，邮编 100084)
http://www.tup.com.cn
http://www.tup.tsinghua.edu.cn

责任编辑：桑任松

印 刷 者：北京牛山世兴印刷厂

发 行 者：新华书店总店北京发行所

开　　本：787×1092 1/16　**印张**：26.5　**字数**：630 千字

版　　次：2003 年 7 月第 1 版　2003 年 7 月第 1 次印刷

书　　号：ISBN 7-302-06731-7/TP・5017

印　　数：0001～5000

定　　价：38.00 元

序

Basic 语言的易学易用的特性，让它在早期的 DOS 时代，就已是学习程序设计的学生们所必修的入门科目。随着操作系统的发展，Basic 也从微软的 Visual Basic 1.0，一直发展到今天最新版本的 Visual Basic .NET。

今日的 Visual Basic .NET 已非昔日的 Visual Basic。它从过去的面向过程的语言，变为今日完整的面向对象语言。不仅如此，它在数据库的存取技术上，又是直接遵循 Microsoft 所提的 ADO.NET 的结构。因此，若现在要评比在 Windows 下的数据库开发工具，我想，Visual Basic .NET 的强劲功能，是可以让过去认为 Visual Basic 无法满足要求，或者是对 Visual Basic 期待很殷切的程序设计人员，重新评估 Visual Basic 的能耐，而让它能在众多的数据库开发工具的选择中，拥有让我们选择的一席之地。

Windows 环境与早期的 DOS 环境相比较，Windows 环境的复杂度绝对远超过 DOS。相当多在 DOS 时期即踏入数据库应用程序研发行列的程序设计人员，在转换到现在的 Windows 环境时，仍旧有一定程度的不适应感，导致他们所开发出来的系统，感觉很像 DOS 下的系统。

Visual Basic 的书籍相当多，但是几乎都是从入门开始，每本书的最后都是希望读者能自己编写出一套优质的系统来；要不然就是从比较高级的角度去切入，去讨论比较深入的主题。但是，却很少看到这样类型的书：从入门开始谈起，然后逐步带领读者进行实践，在不知不觉中走到了高级的领域。

上述的这些原因，是引起我撰写本书的动机。这本书以很浅显的数据库应用程序的系统分析为起点，让您对于“系统分析”这个深奥的名词有一个基本的概念。接着，逐步介绍 ADO.NET 的结构，并用简单但实际却用得到的案例，来实现书中每个章节的主题。最后，我们以进销存管理系统为主题，用 Visual Basic .NET 来把这个项目予以实现。

如果您是 Visual Basic .NET 初学者，这本书可能还不适合您，因为本书不是介绍 Visual Basic .NET 如何操作的入门书籍。所以，您若对 Visual Basic .NET 的操作、基本的结构还不是很熟悉的话，建议您摆一本 Visual Basic .NET 的入门书在身旁，并利用它来当做辅助的工具，可能会让学习的过程中比较顺利。

如果您是非常有经验的程序设计人员，但是对用 Visual Basic .NET 来当做数据库应用程序的开发工具，还不是很熟悉的话，这本书应该可以给您一些经验上的分享，或者是设计上的参考。

最后，感谢您购买本书。也希望本书能真的带给您一些实质上的收获。

作者

2002 年 7 月

目　录

第 1 章　数据库概念

本章将介绍如何由数据库规范化的方法，来学习设计出一个好的数据库。另外，我们也将认识各种不同的数据库运算方式，来了解在各种不同运算方式的数据库之间它们的处理方式与差异性。

1.1　概　　念

我们每天的生活都与数据息息相关。我们有很多的亲朋好友，如果没有一个通讯簿，可能无法牢记所有好朋友的电话、地址或生日。

对于工薪阶层而言，开源与节流是必要的处世之道。每个月花在伙食、交通、娱乐的费用各有多少？如果没有把这些支出做一记录，如何能知道这些费用各花了多少？是不是在娱乐的费用比例偏高而需要加以限制？

通常我们会在年终岁尾时，应客户的要求及考虑到客情的维持，都会赞助一些礼品，当做联欢晚会的摸彩礼品。但是，要赠送多少价值的礼品呢？我们总不能将平均月交易额仅一万元与高达一百万元的客户都送等值的礼品吧？这时，我们就必须去查出客户的交易记录了。

门市的店面寸土寸金，如果陈列架所摆设的商品不是销售最好的，或者是极为畅销的热卖商品，因为补货不及而造成缺货，都会对门市的营运造成影响。这时，我们就必须有完整的销售资料来供我们统计分析各商品的销售金额；同时也必须有一个最新、最正确的库存资料来随时掌握最新的存货状况。如果当前仓库的库存低于安全存量，就必须赶紧向供货商订货，以避免陈列架上缺货。

每一位亲友的电话、地址、生日，每一笔零用钱的支出明细，每一家客户的销售记录，每一项商品的销售状况，所有仓库的商品存货信息等，这些组合起来就是数据。自通讯簿中快速找出高中同学的数据，对我们而言可能是轻而易举的，然而我们可能无法快速在成堆的销售数据中，分别统计每家客户的销售金额。要同时记着所有商品的库存状况，对我们的记忆力而言，可能更是一项挑战。

数据库，简单地说，就是要来存储这些大量的数据，而且要能正确地存储。因为如果我们的销售数据库存储着一笔出货日期为“2001/9/31”的销售记录，那么我们对于这个数据库所存储的数据，大概就没有办法多信赖。

数据会随着时间的流逝而日积月累，导致数据量日渐庞大。如果不能快速在大量的数据中找出某客户、在某段期间的销售记录与合计的销售金额，那么，我们只要把数据放在档案柜就好了，而不用花费了大笔的金钱与庞大的精力，来把数据“换个地方放”。

我们也希望，数据能有一定的保密性。譬如说，公司的人事数据里，会有我们的薪水记录。我们大概都不太愿意让自己的薪水数据被别人知道。

而数据的最终目的，就是我们希望它能提供决策分析的信息。如果我们不能由大量数据的分析，找出最有潜力的客户或商品，则数据也还只是数据而已。

由最基本的元素组合成的一笔数据叫记录(Record)。由相同性质的记录组合起来叫数据表(Table)，相关的数据表组合起来就叫做数据库(Database)。处理数据的建立、查询、更新等工作的程序就叫做数据库管理系统(Database Management System，DBMS)。以表 1.1 所列的销售数据表一为例，纵向的每一个字段叫做元素，横向组合起来就成为一笔记录，所有记录的集合就是数据表，与其他相关的数据表组合起来就是数据库；而用来管理这些数据库的软件，就叫做数据库管理系统。

一个数据库系统中，会有很多的数据库；一个数据库中，会有很多的数据表；每个数据表中，会有为数不等的记录。数据库中，除了有数据表外，还会有其他的对象，如存储过程(Stored Procedure)、触发器(Trigger)、或者是视图(View)与包括可以存取这个数据库的用户等。在图 1.1 中，以 Microsoft 的 SQL Server 为例，说明了一个数据库管理系统应具备哪些成员。

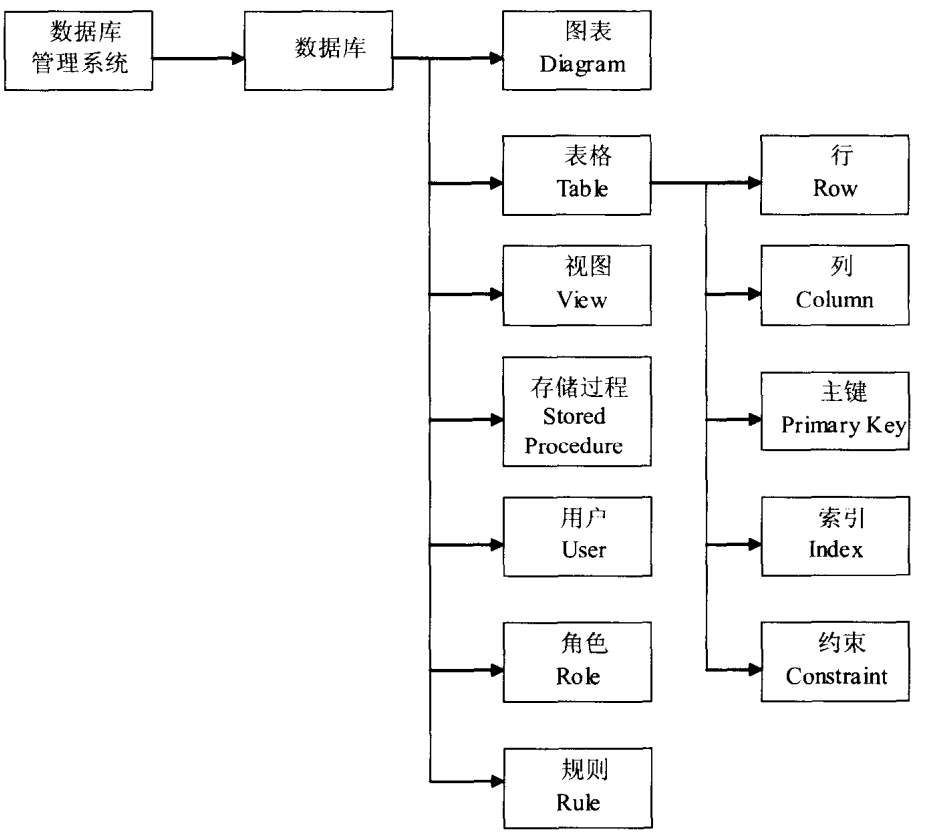


图 1.1 数据库系统的成员

为什么要使用数据库呢？我们可以把这些理由归纳成下列几点：

- 它可以存储大量的数据：数据库可以把大量的表格或单据(出货单、进货单、客户数据表)的数据，或者把我们头脑里记载的大量信息(亲友的通讯簿、管理者的交办事项、工作日程等)，存储在一个很小的空间中，例如：计算机的硬盘、手机的内存等。我们可以利用适当的应用程序，就能取阅这些数据。例如可以用 Access 来存取客户数据、用 Excel 来计算销售业绩等。

- 它是一个有系统的结构：它可以让我们用合理的逻辑来把数据存储起来。譬如说，当产品数据库存在着重复的产品编号；或是销售明细的产品数据在产品数据文件里找不到；甚至我们无法在客户数据里找到出货单的客户数据。这些很奇怪、又非常不合理的现象，在数据库系统中是不允许存在的。(前提当然是数据库设计要正确，或存取它的应用程序不能写错。)
- 它的数据可以让我们很巧妙地处理：现在的手机大概都具有能存储 100 个以上号码的电话簿。假如我们没有把这些电话簿输入到手机，而是一本随身的小记事本，那么要在这 100 笔电话簿数据中，找到想要的电话号码，会用什么方式？若是在手机的电话簿功能，我们可以用名字来寻找，或是由分组的功能来把这 100 笔电话数据给分门别类。这两种方式，明显地分出是否使用数据库来存取数据的优劣。再如一个工厂的零件、原料、成品、半成品等等，动辄数以千计。把这些存货想像成我们的通讯簿，若没有数据库系统的协助，则要找出螺丝的信息，要花多少功夫？数据库系统可以让我们拥有很多的方式来输入这些数据、寻找这些数据。另外，还有一个重要的优势，就是当业务人员跨国洽商时，可能不必再带一迭厚厚的数据前往了，而只要携带笔记本电脑，或是 PDA，就像是把一整个仓库的信息带着走。至于输入与寻找数据的亲切性之好坏，则全取决于数据库的设计是否恰当，及存取这些数据的应用程序设计得是否优良。
- 它具有良好的效率与正确性：一个好的数据库，存取数据的时间与数据量的多寡，并不完全成等比例增加，如图 1.2 所示。数据量愈多，存取数据的时间所增加的比例并不会增加太多。这与用传统的方式在成堆的文件中找所需的数据相比，当然具有无法抗衡的优势。

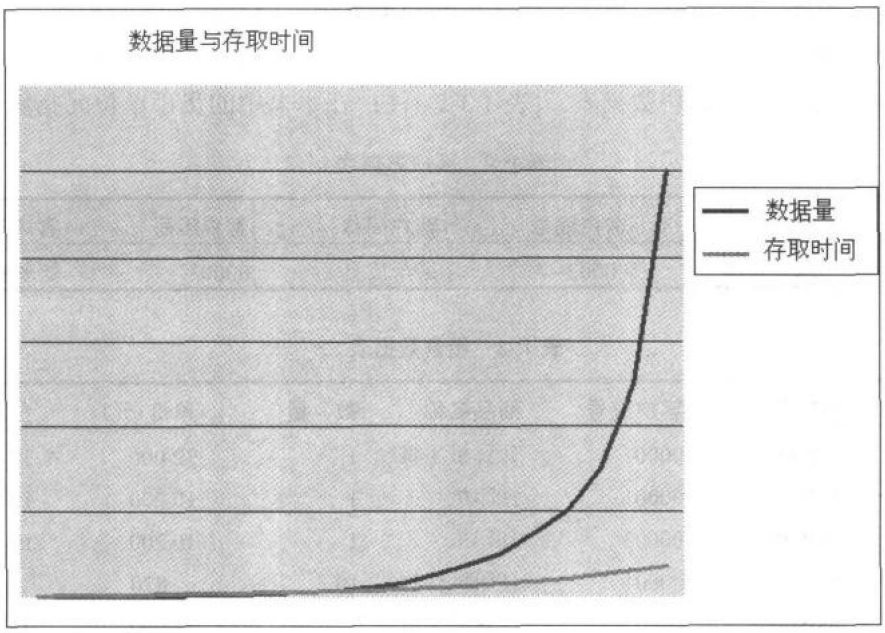


图 1.2　数据量与存取时间的关系

1.2 数据库的规范化

为了提高数据的存取效率与维护的便利性，数据库的设计就占有很重要的地位。数据库规范化的目的，就是要让我们设计出来的数据库能够得到很有效率的执行与合乎逻辑的维护。让我们以一家计算机公司的销售数据来了解数据库规范化的方法。

表 1.1 销售数据表一

销售日期	销货单号	客户简称	商品名称	数 量	单价(元)	金额(元)
90/05/02	90050001	永业贸易	计算机主机	1	32 000	32 000
90/05/02	90050001	永业贸易	打印机	1	10 200	10 200
90/05/02	90050002	德昶食品	打印机	1	10 200	10 200
90/05/08	90050003	永业贸易	打印机耗材	10	870	8 700
90/05/09	90050004	家革皮鞋	计算机主机	3	31 000	93 000
90/05/15	90050005	德昶食品	打印机耗材	15	870	13 050

表 1.1 是某计算机公司的销售记录。我们发现，在这份销售记录中，客户简称的数据中是有重复的。也就是说，一家客户可能会有一笔以上的销售记录；如果每笔销售记录都存储客户简称，那是不是浪费许多存储空间？所以，我们把客户简称独立出来成一个客户资料表，并给每家客户一个独一无二的编号。为每家客户编号的目的是为了输入方便与保持数据的一致性。譬如说，像永业与永业贸易，实际上应该是同一家公司，但是这对计算机而言却是不同的公司，只是因为数据输入人员对简称的取名方式的不同而有不同的结果。如果我们统一输入客户编号，就不会有这样的问题发生。而且，在实际作业中，输入客户编号比输入客户简称要来得方便、迅速多了。下列是我们把销售记录数据表一(表 1.1)分割成客户数据表(表 1.2)与销售数据表二(表 1.3)。(注：在本书中的货币单位元指新台币。)

表 1.2 客户数据表

客户编号	客户简称	客户编号	客户简称	客户编号	客户简称
00001	永业贸易	00002	家革皮鞋	00003	德昶食品

表 1.3 销售数据表二

销售日期	销货单号	客户编号	商品名称	数 量	单价(元)	金额(元)
90/05/02	90050001	00001	计算机主机	1	32 000	32 000
90/05/02	90050001	00001	打印机	1	10 200	10 200
90/05/02	90050002	00003	打印机	1	10 200	10 200
90/05/08	90050003	00001	打印机耗材	10	870	8 700
90/05/09	90050004	00002	计算机主机	3	31 000	93 000
90/05/15	90050005	00003	打印机耗材	15	870	13 050

在客户数据表中，客户编号是不会允许重复的。像客户编号这种不允重复的字段，我们称它为键值字段。当我们在设计每个数据表时，一定要尽可能找出像客户编号这种不允许重复的字段。也就是说，在同一个数据表中，绝对不会有两笔以上完全相同的记录存在。换句话说，键值也是可以由两个以上的字段组合而成的。

客户数据表与销售数据表二之间的关系，是通过客户编号这个字段来连结的。在销售数据表二中，客户编号这个字段就叫做关联字段，它参考了客户数据表中的客户编号这个字段。也就是说，销售数据表二的客户编号，必须存在于客户数据表。换言之，如果客户编号 00001 已有销售数据，那么我们就不能从客户数据表中，把客户编号为 00001 的数据给删除，这样才不会发生销售数据表中找不到客户编号为 00001 的客户资料。也可以这样说，客户可以不必拥有销售记录，但是有销售记录的客户必须存在于客户数据表中，而这就是关系型数据库名称的由来。

我们来看看销售数据表二。我们发现产品名称仍有重复的情形发生，也就是说，每项产品都不会只有一笔的销售记录。若比照分割客户数据表的方式，我们把销售数据表拆成商品数据表(表 1.4)与销售数据表三(表 1.5)。商品数据表中的商品编号是键值字段，也就是说，商品编号是不能重复的。再看销售数据表三与商品数据表，是通过商品编号这个个字段来说明它们的关系的，意思是说，销售数据表三的商品编号是关联字段，参考了商品数据表的商品编号这个字段。

表 1.4　商品数据表

商品编号	商品名称	商品编号	商品名称	商品编号	商品名称
P0001	计算机主机	P0002	打印机	P0003	打印机耗材

表 1.5　销售数据表三

销售日期	销货单号	客户编号	商品编号	数　量	单价(元)	金额(元)
90/05/02	90050001	00001	P0001	1	32 000	32 000
90/05/02	90050001	00001	P0002	1	10 200	10 200
90/05/02	90050002	00003	P0002	1	10 200	10 200
90/05/08	90050003	00001	P0003	10	870	8 700
90/05/09	90050004	00002	P0001	3	31 000	93 000
90/05/15	90050005	00003	P0003	15	870	13 050

我们再来看看销售数据表三，是否有重复的字段。我们发现销货单号有重复的情形。也就是说，同一张销货单会有多笔的销货明细，而且同样的销货单，它的销货日期与客户编号是相同的。于是我们把销售数据表三的销货日期、销货单号与客户编号分割成销货主表(表 1.6)，其余字段就是销货明细表(表 1.7)。销货主表的销货单号是独一的，所以销货单号是销货主表的键值字段。销货主表与销货明细表是通过销货单号来说明它们之间的关系的，所以销货明细表的销货单号是关联字段，参考了销货主表的销货单号。

表 1.6 销货主表

销售日期	销货单号	客户编号	销售日期	销货单号	客户编号
90/05/02	90050001	00001	90/05/09	90050004	00002
90/05/02	90050002	00003	90/05/15	90050005	00003
90/05/08	90050003	00001			

表 1.7 销货明细表

销货单号	商品编号	数 量	单价(元)	金额(元)
90050001	P0001	1	32 000	32 000
90050001	P0002	1	10 200	10 200
90050002	P0002	1	10 200	10 200
90050003	P0003	10	870	8 700
90050004	P0001	3	31 000	93 000
90050005	P0003	15	870	13 050

销货主表虽然在销售日期上有重复的情形，但是请参考表 1.8 即销售主表——日期与表 1.9 即销售主表——单号，可知如果我们把销售日期分割出来，这对数据存储空间而言是不减反增的，而且维护上也不见得会增添便利性，同时更看不出什么好处，所以我们不打算这么做。

表 1.8 销货主表——日期

销售日期	销售日期	销售日期	销售日期
90/05/02	90/05/08	90/05/09	90/05/15

表 1.9 销货主表——单号

销售日期	销货单号	客户编号	销售日期	销货单号	客户编号
90/05/02	90050001	00001	90/05/09	90050004	00002
90/05/02	90050002	00003	90/05/15	90050005	00003
90/05/08	90050003	00001			

销货明细表的金额是单价乘上数量，所以我们可以视情况来决定是否要省略这个字段。此处我们选择省略。表 1.10 是把金额字段省略后得到的销货明细表二。

表 1.10 销货明细表二

销货单号	商品编号	数 量	单价(元)
90050001	P0001	1	32 000
90050001	P0002	1	10 200
90050002	P0002	1	10 200

续表

销货单号	商品编号	数　量	单价(元)
90050003	P0003	10	870
90050004	P0001	3	31 000
90050005	P0003	15	870

现在，看看表 1.2 的客户数据表、表 1.4 的商品数据表、表 1.6 的销货主表、表 1.10 的销货明细表二，似乎找不到可以再分割的字段了。所以对于表 1.1 的销售数据的规范化工作，到此告一个段落。

要如何得知数据库的规范化能告一段落呢？此处有一个非常重要的原则，而只要遵循着这个原则，那么我们所设计出来的数据库就至少达到了一定的水准。这个原则就是：在数据表与数据表间，只可以存在着一对一、或者是一对多的关系。当我们发现在两个数据表间存在着多对多的关系时，就代表我们数据库的规范化工作还没完成。我们必须试着在造成多对多关系的两数据表间，再分割出来一个以上的数据表，以达到一对多或一对一的目的。

总之，数据库的规范化，主要有下列目的：

- 节省磁盘驱动器的存储空间。
- 数据维护的便利性。例如：当我们要把商品编号 P0001 的商品名称由计算机主机改成商用计算机主机时，如果没有表 1.4 的商品数据表，我们是不是就得在销售数据表二中，逐一修改呢？反之，我们只要在商品数据表中把商品名称改为我们要的“商用计算机主机”，再通过销货数据表三的关联字段商品编号，来参考到商品数据表中的商品编号，我们就能自商品数据表中带出这个新的商品名称了。

要做好数据库的规范化，其实并不难。只要把握最重要的三项要领：

- 数据表与数据表之间必须是以“一对一”或“一对多”的关系存在。如果发现有多对多的关系，则势必要再做进一步的分割。
- 每个数据表中都不能有重复的记录。
- 相同性质的字段放在同一个数据表。

图 1.3 就是经我们规范化后的数据库的销售数据库关联图。

客户数据表与销售主表是一对多的关系；商品数据表与销售明细表是一对多的关系；销售主表与销售明细表是一对多的关系。由这个关联图来看，我们不难发现：关系型数据库与实际的操作比较接近。这家计算机公司内部一定会有客户数据卡(客户数据表)，也一样会有出售的商品目录(商品数据表)，当然销售单更不可少。而销售单不就是有单头(销售主表)与单身(销售明细表)吗？

到当前为止，我们只做到数据库的三阶的规范化。其实，在数据库的领域中，还有所谓的第四阶、第五阶、第六阶等规范化。只不过是我们不必被这些专业的名词给吓唬住，而在感到十分困惑之余，觉得数据库的规范化是一门高深莫测的学问。

若要以实例来说，我们就以销售主表来进一步说明。通常我们与客户的交易，付款条件与方式不尽相同。也许这笔交易的毛利较低，我们会要求客户直接用现金付款；也许客

户的付款习惯就是货到开立 30 天的期票；甚至是月结汇款到我们的银行账户等。若我们在销售主表摆一个付款条件的字段，则销售主表的数据可能会像表 1.11 的结果(假设都是货到马上收现金)：

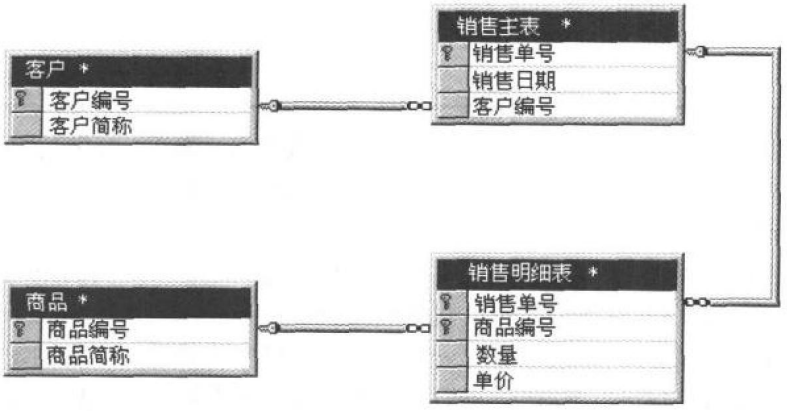


图 1.3 销售数据库关联图

表 1.11 销售主表(变更一)

销售日期	销货单号	付款方式	客户编号
90/05/02	90050001	现金	00001
90/05/02	90050002	下货收现	00003
90/05/08	90050003	现金	00001
90/05/09	90050004	现金付款	00002
90/05/12	90050005	货到现金票	00002
90/05/15	90050006	货到收现	00003

今天，我们要从这些销售数据里，找出是属于下货收现的销售单，该用什么方式呢？我们或许会觉得找出付款条件里有“现”这个字就行了！那么，“90050003”这笔销货单算不算呢？而对商业习惯来讲，开票日与到期日差距在 7 天内的支票都算是现金票，那么这与我们要找出下货收现的定义是否相符呢？

有鉴于此，我们可能会建立一个付款方式代码文件。这个文件的数据如表 1.12 所列。

表 1.12 付款方式代码表

付款方式代码	说 明	付款方式代码	说 明
01	货到现金	31	月结 30 天期票
02	月结现金	32	月结 45 天期票
21	货到现金票	33	月结 60 天期票
22	货到 30 天期票		

则表 1.11 可变成像表 1.13 所示的结果。

表 1.13　销售主表(变更二)

销售日期	销货单号	付款方式代码	客户编号
90/05/02	90050001	01	00001
90/05/02	90050002	01	00003
90/05/08	90050003	01	00001
90/05/09	90050004	01	00002
90/05/12	90050005	21	00002
90/05/15	90050006	01	00003

现在，我们要找出销售单的付款条件是现金的，是不是只要找出付款方式代码为“01”就行了呢？

我们只要能把握数据库规范化的三大要领，就能做出具有专业水准的数据库了。

1.3　数据库系统的结构

数据库系统的结构可依其数据运算及存取的方式，分为四大类型：即主机集中型数据库系统、文件型数据库系统、客户机/服务器结构型数据库系统以及多层式结构型数据库系统。

1.3.1　主机集中型数据库系统

主机集中型数据库系统的组成如图 1.4 所示。

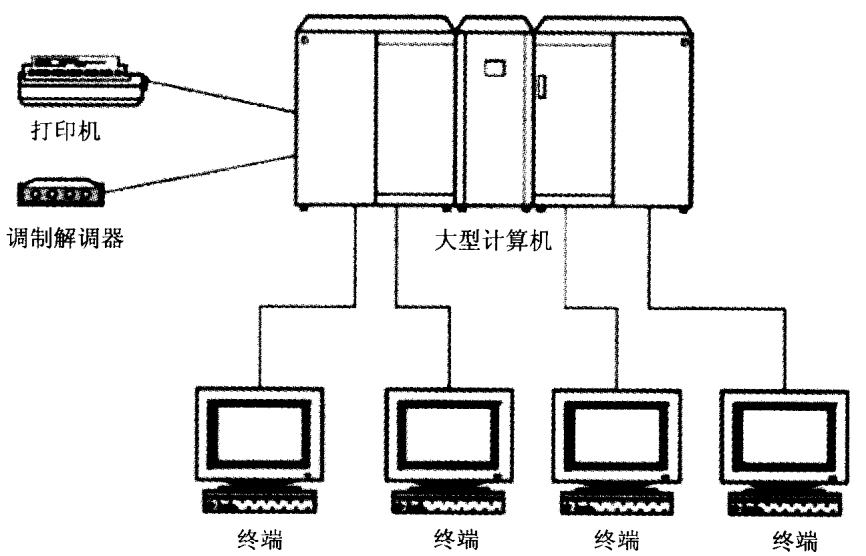


图 1.4　主机集中型数据库系统

主机集中型数据库系统主要发展于早期个人计算机还不普及时。这种集中型数据库系统，是由一组大型计算机或小型计算机，再加上一群终端机组合而成。数据库的存储、计

算、读取与应用程序的执行，全部集中在后端的主机上进行。前端的终端机只负责显示主机所传来的信息，或把输入的信息回传至主机处理，有点类似家里的电视机接上游戏主机。

主机集中型数据库系统的优点是：信息人员只要专心管理好这台主机，不太需要去前端的终端机进行维护，所以管理上很方便；同时，所有的用户可以共享一些外围设备，如打印机、调制解调器等。在计算机还不普及的早期，像打印机等计算机外设是相当昂贵的东西，而不是一般人、公司及商家所能负担的。

它的缺点是：当处理的数据越来越多，或同时联机到主机的终端用户越来越多时，主机的性能可能就无法满足需求。这时，就必须更换、添加设备或升级主机。通常这类型主机及其所有的外围设备，其购置的价格都非常昂贵。因为这类型的设备都是由少数的大型制造商所研发生产的，所以量少价高，而且更因为生产数量不多，故各家厂商间的硬件、软件并不完全兼容。换句话说，如果要更换的主机的生产厂商与原来所使用的主机的制造厂不同时，这时就不光是换硬件而已，连所有的软件、应用程序都有可能要一并重新来过。这样的负担并不是一般企业所能承受的。

1.3.2 文件型数据库系统

文件型数据库系统的组成如图 1.5 所示。

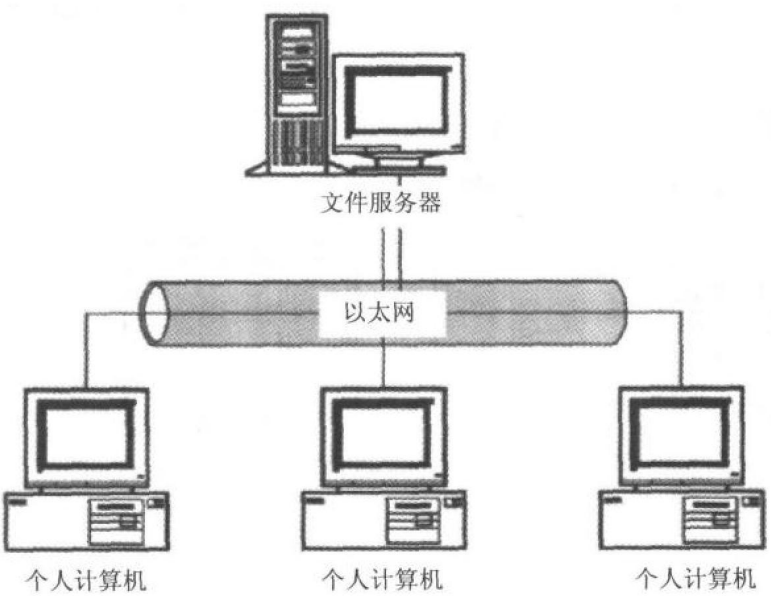


图 1.5 文件型数据库系统

随着苹果计算机、IBM 个人计算机的诞生，其开放性的结构、日渐平易近人的价格，以及愈来愈强的执行性能，已为一般企业所能负担，而文件型数据库系统也就在此时趁势崛起。这种类型的数据库系统，是由一台性能较好(或说较高档)的计算机当主机，即文件服务器，然后连上几台工作站。这台文件服务器，只负责文件的存储工作而已，而所有的数据运算、应用程序的处理都摆在前端的工作站。如早期的 dBase III，到今日的 Access，就是一些拥有高知名度的文件型数据库系统。