TURING 图灵程序设计丛书 数据库系列

Apress®

Pro SQL Server 2008
Relational Database Design and Implementation

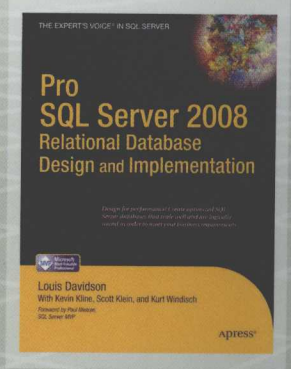
SQL Server 2008 数据库设计与实现

[美] Louis Davidson
Kevin Kline
Scott Klein
Kurt Windisch 著

程桦 张绪业 等译

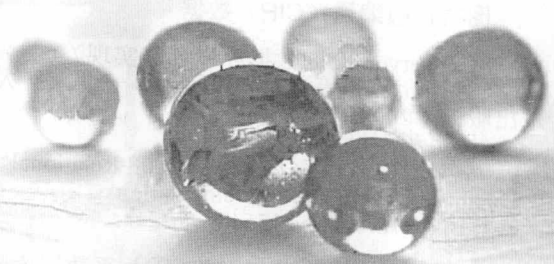
MVP Microsoft Most Valuable Professional

- 资深数据库专家的心血力作
- SQL Server设计思想的独到解析
- 关系数据库实现的通关宝典



人民邮电出版社
POSTS & TELECOM PRESS

TURING 图灵程序设计丛书 数据库系列



Pro SQL Server 2008
Relational Database Design and Implementation

SQL Server 2008 数据库设计与实现

人民邮电出版社
北　京

图书在版编目（CIP）数据

SQL Server 2008数据库设计与实现 /（美）戴维森（Davidson, L.）等著；程桦等译. —北京：人民邮电出版社，2009.11
（图灵程序设计丛书）
书名原文：Pro SQL Server 2008 Relational Database Design and Implementation
ISBN 978-7-115-21554-3

Ⅰ.①S… Ⅱ.①戴… Ⅲ.①关系数据库-数据库管理系统，SQL Server 2008-程序设计 Ⅳ.①TP311.138

中国版本图书馆CIP数据核字（2009）第177595号

内 容 提 要

本书深入浅出地介绍了目前世界上最受欢迎的数据库管理系统之一——SQL Server。全书共分三个部分：第一部分阐释了数据库的基本概念，讲解了数据库建模语言；第二部分展示了从概念建模到在SQL Server 2008上真正实现数据库的过程；第三部分深入探讨了SQL Server若干方面的技术细节，如数据保护、索引、并发访问等。通过将理论融入数据库实践，清晰地讲解了关系型数据库的设计原则，完整地展示了如何进行良好的关系型数据库设计，深入揭示了SQL Server 2008的技术细节。

本书浓缩了作者作为SQL Server数据库架构师多年来丰富的实践经验，适合各类数据库开发和管理人员学习参考。

图灵程序设计丛书

SQL Server 2008数据库设计与实现

◆ 著 [美] Louis Davidson Kevin Kline
Scott Klein Kurt Windisch
译 程 桦 张绪业 等
责任编辑 傅志红
执行编辑 马晓燕

◆ 人民邮电出版社出版发行 北京市崇文区夕照寺街14号
邮编 100061 电子函件 315@ptpress.com.cn
网址 http://www.ptpress.com.cn
北京顺义振华印刷厂印刷

◆ 开本：800×1000 1/16
印张：36.25
字数：1040千字 2009年11月第1版
印数：1-3 000册 2009年11月北京第1次印刷

著作权合同登记号 图字：01-2009-2893号

ISBN 978-7-115-21554-3

定价：89.00元

读者服务热线：(010)51095186 印装质量热线：(010)67129223
反盗版热线：(010)67171154

译 者 序

如果将数据库称为现代数字生活的基础，那么这样的说法一点也不夸张——大型企业应用、网站，所有这些东西的背后，都是数据库在支撑。并且，最终说来，产生价值的并不是绚丽的界面和现代化的输入方式，而是存放在数据库中的数据。不幸的是，虽然关系型数据库历经了约30年的发展，有成熟的理论和大量的实践基础，但是，大多数设计、开发人员在设计数据库结构时仍然是“跟着感觉走”，根据业务的需要和编程的方便，把字段这张表放几个那张表放几个完事。对这样设计出来的数据库，只需多问几个为什么——业务模型是如何体现的？数据完整性如何保证？性能是如何权衡的？——恐怕设计者就该崩溃了。

这也难怪，设计、开发人员在学校中学习数据库时，理论书籍离实际开发较远——试问有几个人能够以可实践的方式把规范化的几条原则阐述清楚？在工作时，使用的数据库资料和书籍又往往是“手册型”，大多仅仅讲解特定数据库提供的功能。

正如作者所说——“我的书目标很简单，那就是填补这个空白，架起学院教科书与通常针对SQL Server所写的，纯粹面向实践的书之间的桥梁”。翻译的过程中，译者感到此言不虚：作者从数据库的基本概念到数据库建模，从如何运用规范化原则到如何做成实际的数据库表，从如何保护数据库完整性到如何提高数据库的性能，从数据库的安全机制到并发事务控制，从数据库设计开发的常用模式到应用程序的数据访问策略，既有理论又紧扣实践。阅读本书的过程，就是把一些以前模糊地知道，但又觉得很难运用的理论实实在在地运用到实践中的过程。

虽然本书是针对SQL Server这个特定数据库平台来阐述理论和实践的，然而，本书所体现的思想和方法，完全可以运用到其他关系数据库平台上。

翻译的时间很紧，如果没有家人的支持和理解，很难想象如何完成这项工作。

感谢人民邮电出版社图灵公司很好地推动了这本书的翻译工作。另外，图灵公司的论坛上丰富的资料和活跃的讨论也使我们眼界大开，受益良多。

翻译工作并非阐述自己的思想，翻译的第一要务是忠实地传达原著者的思想。虽然无法自由地表达自己的想法，然而，翻译的快乐就在于：使另一个人的好想法能让更多的人了解。当然，由于时间紧张，译者水平有限，错误粗糙之处相信不在少数，敬请广大读者批评指正。

感谢我的妻子魏萍，你容忍我一吃完饭就坐到计算机前。谢谢女儿程灵馨，在爸爸不能陪你玩时，5岁的你居然也表示理解。谢谢作者Louis Davidson，你耐心地回答了我的问题。

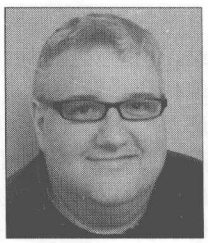
程桦

感谢我的未婚妻张艳，是你的鼓励和体贴让我在寒冷的冬日伏案不辍。还要感谢我的家人，你们殷殷的期盼让我产生了对知识的不渝追求。

张绪业

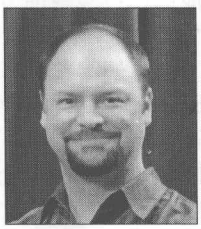
2009年5月

关 于 作 者



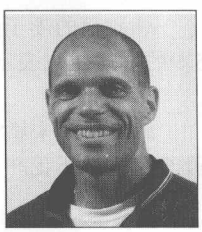
Louis Davidson，作为企业数据库开发人员和架构师，他拥有超过15年的工作经验。目前他是田纳西州Nashville的Christian广播网络和NorthStar工作室的数据架构师。对于Louis而言，他全部的职业经验几乎都与微软的SQL Server有关，从早期版本一直到当前最新版本的Beta版。Louis是一本讲数据库设计的书的4个版本的主要作者。Louis主要的兴趣领域是数据库架构和用T-SQL编码，并且，他设计过许多数据库，在这许多年中编写过数以千计的存储过程和触发器。

关于特约作者



Kevin Kline是Quest软件公司SQL Server解决方案的技术战略经理。Kevin从2004年开始就是微软的SQL Server MVP，他是国际SQL Server专家联盟（PASS）创立委员会的成员和上一届总裁。他独自写作或参与合著了好几本书，包括*SQL in a Nutshell*（O'Reilly，2004）、*Pro SQL Server 2005 Database Design and Optimization*（Apress，2006）以及*Database Benchmarking: Practical Methods for Oracle & SQL Server*（Rampant，2007）。Kevin给*SQL Server Magazine*和*Database Trends and Applications*杂志供稿，他的博客可以在SQLBlog.com和SQLMag.com上找到。在世界范围内的会议上，比如微软的Tech Ed、PASS社区峰会、微软IT论坛、DevTeach以及SQL Connections，Kevin也是人气最高的演讲者。从1986年开始Kevin就活跃在IT工业中。

Scott Klein是一位独立咨询师，对SQL Server、.NET和XML相关的所有知识都充满热情。他是几本书的作者，这包括*Professional SQL Server 2005 XML*（Wrox，2006）和*Professional LINQ*（Wrox，2008），他还为*SQL PASS Community Connector*写每两周一次的特辑文章。他也为好几个网站供稿，这包括Wrox（http://www.wrox.com）和TopXML（http://www.topxml.com）。他经常同佛罗里达附近的SQL Server和.NET用户群体交流。Scott住在佛罗里达的Wellington，当他不坐在计算机前时，你会发现他和自己的家人在一起，或者骑着他的雅马哈摩托车轰鸣在当地的摩托车越野赛赛道上。你可以通过ScottKlein@SqlXml.com联系他。



Kurt Windisch是Levi, Ray, and Shoup, Inc. 公司内部IT部门的应用程序监管，该公司是一家技术方案的全球供应商，总部位于伊利诺伊州的Springfield。Kurt在IT工业中的经验超过了17年。他在SQL Server专家联盟的董事会中服务了5年，为很多SQL Server杂志供过稿，也在讨论SQL Server数据库编程的国际会议上发过言。

序

数据库设计是Louis和我都热衷的事情。今年秋季，我将幸运地和我的朋友Louis Davidson登上同一个舞台。我们将在DevLink会议上演讲，地点是Louis的家乡Nashville，然后，在西雅图的PASS上我们也将同台演讲。在这两次演讲中，我们都将针对数据库设计这个题目开展讨论和辩论，并探讨对以数据为中心的应用来说数据库设计有多关键。

数据库设计是门科学也是门艺术。规范化应该说是门科学，而确定实体的范围则是一门艺术，需要具有与各种数据库打交道的经验才能学到手。优雅的解决方案都有一种简单的美。Louis不止是位作者和数据库设计人员，也是一位大师级技师。在本书中，你会发现字里行间都隐藏着作者的呼吁——改善你的技能，精心雕刻出能够经受时间考验的数据库，创造一个能让开发人员充分施展拳脚的、虚拟的数据世界。

数据库设计是每个以数据为中心的应用的基石。一个优雅的数据库设计使得数据一目了然、容易查询，并且使用有效的、基于集的查询将开发人员武装起来，使他们获得成功。但是，如果数据库设计就很糟糕，则无论多少代码也无法补偿，也无法为数据库加上缺失的功能。对任何以数据为中心的应用来说，没有什么角色比数据建模人员这个角色更为重要。用户界面换来换去，但是，即使经历了好几代应用编程语言，数据仍然存在着。数据库架构上犯下的错误迟早会被未来的程序员诅咒，即使他现在还未出生，即使他用的是现在还没发明的语言和工具。在我的好朋友Louis的指导下，多花一点时间来润色你的数据库设计，绝对是件值得的事。

因此，欢迎你阅读本书，这是Louis针对软件世界最重大的工作所写的战地指南的第三版。能够为本书作序我感到无比愉快。

Paul Nielsen
SQL Server MVP

前　言

我经常扪心自问，“为什么要做这件事？为什么还要写新版本？这值得吗？如果我抛开书去打任天堂游戏机，我是不是能帮超级玛丽快点把公主救出来？”在微软MVP 2008年峰会上，这些问题都由一位MVP帮我回答了。他感谢我写了这本书，说他曾经试着读点这方面的大学教材，但那些书都太难了，读不太懂。

“喔，是的，”我想，“那是我最开始要写这本书的原因。”当我最早开始设计数据库时，我从几位伟大的导师那里学到了不少东西，但是，当我想进一步深入研究时，我开始寻找数据库设计方面的材料，但没找着多少。我找到的最好的一本书是Chris Date的*An Introduction to Database Systems*（Addison Wesley，2003），然后我读了自己能读懂的部分。然而，我很快就有点晕头转向，没法做到一旦理解了概念之后，就能很好地将关于设计的理论转换到真正的、同时较为简单的设计过程中去。在我所使用的Chris和其他人的教材中，很明显，创建关系模型这件事包含了太多理论，甚至是数学知识。

如果你想要成为一名理论家，那Chris的书就是必读的，当然，除此之外还有很多可以参考的书（最好去看看http://www.dbdebunk.com/books.html，你会找到更多的书）。问题在于，这些书大多数都包含了太多的理论，超出了一般实践工作者所需了解的（或者愿意花时间去读的）内容，并且它们没有真正地深入到实际数据库系统的实现中去。我的书的目标很简单，那就是填补这个空白，在大学教材与通常针对SQL Server所写的、纯粹面向实践的书之间架起一座桥梁。我的目的不是顶替这些书，完全不是，在我的书架上有很多这样的书。本书更像是一本面向技术的书，而不是讲解SQL Server功能的使用手册。我将介绍关系引擎的大多数典型功能，教给你使用它们的技术。然而，我不能说这是你书架上需要的唯一一本书。

如果你有本书以前的版本，你可能会问为什么还需要这个新版本，我了解你的感受。我花了大量时间找出应该买这个新版本的理由，而这个理由并不那么明显——那就是，我现在涵盖了2008的所有功能。很明显，这只是部分原因，最重要的是，我一直在努力加入更多新内容让你的工作更轻松。我加入了一章（第7章）介绍开发模式，每一章中也都加入了大量新材料，从而能帮你改进数据库设计。

诗人和剧作家奥斯卡·王尔德曾经说过：“只有年轻人才会无所不知。”现在回头看时有点后悔，自己在写第一本书*Professional SQL Server 2000 Database Design*之前，以为自己什么都知道。正是那种无知和放纵不羁的热情，才让我有勇气写下第一本书。最后，我确实完成了第一本书，它之所以还不错，主要归功于技术编辑给予的批评。并且，如果我没有最初那种驱使我写完书的热情的话，我也不可能写这本书的第四版。无论如何，如果你花几周的时间逐章逐节地比较了这些书的每个版本的话，你会发现这是个内容逐步丰富的过程，而且作者也在逐步成熟。

过程的逐步丰富和作者的逐渐成熟都有其原因。一个原因是过去两个版本以来我所碰到的编辑：首先是Tony Davis，现在是Jonathan Gennick。他们两人对我的文风都多有指摘，并且他们把本书的结构调整得很棒。另一个原因可以简单地归结为经验，因为从我开始写第一版以来已经过了8年。但是，

书中所用的材料之所以有进步，最大的原因还是它们经过了检验。一方面我在对别人贡献着自己的好评论，另一方面我也收到了大量关于如何改进的反馈（其中一些评论可不那么友善）。我非常用心地倾听，从书发布那天开始就记一系列笔记。对于我能用得上的任何反馈，我都会感到很高兴。可以用电子邮件联系我（louis@drsql.org），如果你愿意也可以在我的网站（drsql.org）留下匿名反馈。你可以在网站上找到一个附录，里面含有我希望自己在写这本书时就了解了的材料。

数据库设计的目的

数据库设计的目的是什么？你到底为什么要关心这事？主要的原因是设计良好的数据库用起来很简单，因为所有东西都在其应处于的逻辑位置上。这非常像一个收拾得很好的橱柜，如果你需要红辣椒粉，直接到调味品搁架的红辣椒粉那格就能取到，这就比到处乱找要强得多。但是，许多系统组织得一团糟。即使每件东西都有一个分好的地方，如果找起来很困难，那件东西也就没啥价值了。想象如果一本电话簿完全没有排序会如何？或者，编排字典的时候把词任意放在文字中能放下的地方又会如何？数据库如果组织良好的话，即使要写一两个联结，你也可以靠本能就知道去哪里获取需要的数据。我的意思是，说到底，这难道不是件很有趣的事吗？

你也可能吃惊地发现，数据库设计是件相当简单的任务，不像听起来那么困难。与随着项目进行而拼凑数据存储相比，要做好这件事需要在项目一开始多花些时间，但是，在项目的整个生命周期中，这样做能获得相应回报。要想正确地进行数据库设计，我们面对的最具挑战性的问题是：与不做设计相比，正确地设计数据库会花更多的时间（在项目计划会议上，关于时间的争论经常发生）。因为数据库设计没有什么可见的东西来让客户感到赏心悦目，因此这一阶段经常会被压缩，目的就是让事情看起来进行得较快。对于普通客户而言，即使是最没有挑战性、最不让人感兴趣的用户界面也比最漂亮的数据模型要吸引人得多。虽然一般说来，数据才是投资创建系统的最终原因，但是，用户界面编程往往占据了中心地位。并不是说你的同事没有注意到一个糟糕的数据模型与一个漂亮的数据模型之间的区别，他们确实注意到了，但是，当程序员需要编码时，往往会顾不上考虑如何能正确地存储数据。我希望自己知道这个难题的答案，因为要是有这个答案，我的书会卖出一百万本。本书将努力提供一些技术和过程，帮助你进行数据库设计，力求对新手清晰易懂，对老练的专家也有所助益。

这个设计和架构数据存储的过程与数据库安装和管理的角色并不相同。例如，作为数据架构师的角色，我很少去创建用户、执行备份，或者去设置复制与集群。我很少谈及这些任务，它们被认为是管理工作，由DBA角色担当。既是开发人员又是DBA的人并不罕见（实际上，如果你在小公司工作，你会发现自己担当了多个角色，以至于不堪重负），但是，如果你能将自己的思维与偏重实现的角色相分离，从而使你能更多地思考数据有多难用，那么一般说来，你设计出的东西会好很多。对许多情况而言，数据库设计没有看起来那样困难。

注解 为安全起见，我要澄清一件事：如果你做过任何编程工作的话，毫无疑问，你会反对本书中的某些观点和意见。我完全同意本书不是什么Katmai[①]的圣路易斯[②]福音书。我的意见和观点来自于超过16年对数据库的学习、在数据库上的工作，并且由来自于许多不同的人、书、大学课堂和研讨会的知识所补充。我在“致谢”一节已经感谢了其中的许多人，但是，我忘掉

① Katmai：SQL Server 2008的代号。——译者注

② St. Louis：作者开玩笑，称自己为圣路易斯。——译者注

的人名也数以百计，我脑中铭刻的某些知识精华就来自于他们。本书所展示的设计方法是所有这些思想的集合。我希望本书能被看作是个有用的学习工具。通过阅读本书和其他人的著作，再实践你自己的想法，你将会总结出自己的方法论，它会适合你自己，并使你成为成功的数据库设计人员。

本书的结构

本书由如下各章组成。

第1章：数据库概念简介。该章提供了对关键术语和概念的简要概览。

第2章：数据建模语言。该章的作用是介绍数据架构师的主要工具——模型。该章详细介绍了一种建模语言——IDEF1X，因为它是本书中用于展示数据库设计的建模语言。还介绍了其他几种常见的建模语言，因为某些读者或者出于偏爱、或者由于公司的要求会使用这些类型的模型。

第3章：概念阶段数据建模。在概念阶段数据建模中，我们的目标是讨论这样的过程：引入顾客的需求集合，将表、列、关系和业务规则放入数据模型格式中相应的地方。

第4章：规范化过程。数据库设计过程的下一步是规范化。规范化的目标是将表、列、关系和业务规则的集合格式化为每个值只在一个地方存储、每张表只代表一个单一实体的形式。在最初几次进行规范化时，这个过程会感觉不太自然，因为你不用琢磨如何使用数据，却必须琢磨数据本身以及结构会如何影响数据的质量。然而，一旦你掌握了规范化，那么，不以规范化的方式存储数据就会让人觉得不舒服。

第5章：实现基础的表结构。在数据库设计过程中，这是第一个要启动SQL Server，开始写脚本来构建数据库对象的时间点。该章讨论了构造表（包括为列选择数据类型）及关系，其中提到了实现后的结构可能与我们在规范化过程中得到的模型不一致的问题。

第6章：保护数据的完整性。除了要将数据安排在表和列中之外，也需要实现其他业务规则。在SQL Server中，保证数据完整性条件得以满足的第一条防线由CHECK约束与触发器构成，因为用户无法轻易绕开由约束和触发器进行的验证。这一章还讨论了其他几种保护数据的方法，它们可以用存储过程和客户端代码来实现。

第7章：模式和查询技术。除了设计表时的各种基本技巧之外，我们还利用几种技巧来构成常见的数据/查询接口，从而方便未来的查询和使用。该章讨论了几种常见的、有用的模式，同时也观察了某些人需要使用的一些模式——有些查询需求的接口实现对你来说非常困难，而这些模式能使它们变得更简单些。

第8章：数据访问安全。这些日子以来，安全问题几乎在每个程序员的头脑中都占据了很重要的位置。该章讨论了用来在系统中实现数据安全的某些策略，比如使用视图、触发器、加密，甚至使用SQL ServerProfiler等。

第9章：表结构与索引。该章展示了如何在SQL Server中建立表结构的基础知识，以及某些对数据进行索引以获得更好性能的策略。

第10章：并发编程。数据库设计和实现过程的一部分是不仅要考虑结构，而且还要考虑如何在多个用户之间最大化资源的利用率。该章描述了几种策略，它们都与如何在你的数据访问和修改代码中实现并发相关。

第11章：数据访问策略。该章讨论了关于编写代码来访问SQL Server的许多概念和问题。比较了即席SQL与存储过程（包括两者面对的所有风险和挑战，比如计划的参数化、性能、开发投入、可选

参数、SQL注入等），同时也讨论了T-SQL还是CLR对象更好，其中包括用CLR来编码不同类型的对象的示例。有关CLR的材料由Kurt Windisch提供，可以下载的CLR示例代码也由他提供。

附录A：Codd的RDBMS十二法则。本附录介绍了Codd关于数据库应该如何实现的创新的十二条原则。随着硬件的发展，我们离实现他的梦想越来越近。

附录B：标量数据类型参考。本附录介绍了可以被合法地当作标量类型的所有类型，为什么要用它们，以及它们的实现信息和其他细节。

再一次强调，给我反馈时不用犹豫，任何时候都不要犹豫（只要你不在早上3点给我打电话就行）。对任何章节，我都将努力改善大家发现还有所欠缺的地方，并将改善后的内容发布到我的博客上（http://sqlblog.com/blogs/louis_davidson），其标签为DesignBook，同时也会发布到我的网站上（http://drsql.org/ProSQLServerDatabaseDesign.aspx）。如果有什么新的想法或者找到了一些错误，或者有额外的资料，我都会将其添加在这两个地方。因为这是一本书，而不再是零零星星的资料，这两处是我能想到并选来发布信息的地方。

致　谢

如果说我看得更远，那是因为我站在巨人的肩上。

——伊萨克·牛顿

我并非什么天才，也不是数据库设计界的什么先驱。我承认下面这些人对本书的出版都给予了极大的帮助。有些人直接帮助了我，而有些人也许甚至不知道有这么一本书存在。无论如何，他们对本书的出版都有重要影响。

感谢我的妻子Valerie Davidson以及女儿Amanda Davidson，因为她们已经是第四次容忍了这种疯狂的行为（还帮我挑选了封面图片）。感谢我的妈妈，她用她的唠唠叨叨，让我获得了很好的娱乐。感谢我的岳母，她也时不时过来帮帮忙。

感谢我在世界上最好的朋友，还是在上大学那会儿，他让我开始对计算机产生兴趣，而那时我还一心想当个数学家呢。我想念他的程度也许超出他的想象，未来某一天，我一定要再找到他并当面致谢。

感谢我的良师Mike Farmer、Chip Broecker和Don Plaster，感谢他们早年给我的指导。

感谢Gary Cornell，他给了我机会写自己想写的书。

感谢Frank Castora和Don Watters，他们抽出时间来读本书的Beta版。（Don后来成了本书的技术编辑！）感谢Evan Terry和Wayne Snider所做的技术编辑工作，尽管这个过程有时有点偏离中心，并且结果和开始想的不太一样。

感谢我现在的经理Rob Murdoch，他允许我参加了几个会议，这对我确实很有帮助，让我能够写好这本书。感谢所有在CBN以及已经倒闭了的Compass公司以前的同事，他们为本书提供了许多例子。

感谢Scott Klein、Kevin Kline和Kurt Windisch，他们详细介绍了我不想（其实是没有足够能力）碰的主题。

感谢Paul Nielsen，他主动提出并花时间来为本书作序。

感谢我遇到过的最出色的编辑人员。这包括Jonathan Gennick，由于我对英语运用得如此之差，以至于有几次他忍不住（象征性地）要打我，没有他帮忙的话，我的东西有时看起来会像是没文化的乡巴佬写的。大多数编辑都写在版权页上了，但是，我想对Tony Davis（他为本书的上一版出力不小）特别说声谢谢，因为他使这本书变得很棒，尽管我的文风往往是杂乱无章的。

感谢Raul Garcia，他在微软SQL Server引擎组工作，给我提供了有关使用EXECUTE AS和基于证书的安全方面的信息。

感谢Isaac Kunen，与他在Tech Ed上的讨论帮助我更好地理解了空间数据类型。

感谢James Manning，他对READ COMMITTED SNAPSHOT那部分内容提出了建议。

感谢Chuck Heinzelman，他是如此好的一位朋友，他给了我机会，在这本书快要完成时为SQL Server标准写文章。

感谢所有过去一年半中与我一起工作过的MVP们，他们这些人真是绝无仅有。Steven Dybing、Ben Miller和现在的Ali Brooks，和你们一起合作太棒了。我还想单独列出其他一些人，他们在一些具体事情上对我帮助不小：Dejan Sarka和Andrew Watt，他们以不可思议的精力审阅了本书的上一个版本，没有让我漏过任何细节。Hugo Kornelis对本书的上一版给出了最负面的批评，他让我看到了自己的不足之处（要是我能够让他成为技术审稿人该多好）。Steve Kass给我提供了代码，用于展示货币数据类型有怎样的问题，他还针对新闻组中的难题提出了良好的解决方案，也促使我进行了更多的思考。Erland Somarskog帮助我更多地理解了错误处理是如何工作的，当然，还有其他主题（更不用说他那个很棒的网站了，http://www.sommarskog.se）。Adam Machanic对我的博客和新闻组中的许多话题都有帮助。感谢Aaron Bertrand很棒的网站http//www.aspfaq.com，以及那些关于鞋子的记忆[①]。感谢Kalen Delaney为我和社区所做的一切，感谢Greg Low博士将我放到了他的http://www.sqldownunder.com播客中。Kim Tripp为SNAPSHOT隔离级别提供了很棒的论文，Arnie Rowland给了我微软的两个课件，对我写作本书的某些部分很有帮助。感谢Allen White花在Tech Ed上的时间及可怕的采访，感谢Jason Follas在Tech Ed上讲述了空间类型。我还想感谢Tony Bain、 Hillary Cotter、Mike Epprecht、Geoff Hiten、Tom Moreau、Andrew Kelly、Tony Rogerson、Linchi Shea、Paul Nielson、Tibor Karaszi、Greg Linwood、Peter Debetta、Tom Moreau博士、Dan Guzman、Jacco Schalkwijk、Anith Sen、Jasper Smith、Ron Talmage、Christian Lefter和Kent Tegels，因为你们所有人在过去的日子里，在新闻组中都对我有过特别的帮助，教会了我新东西，让我的书能够写得更好。

感谢那些数据库理论的大师们，他们将理论知识传播到了我的头脑中，比如E. F. Codd、Chris Date、Fabian Pascal和Joe Celko，还有我在田纳西大学Chattanooga分校的教授，以及其他许多人：没有你们，我知道的东西会连现在的一半都不到。感谢Date先生对第1章的评审；也许等到本书的下一版，你会愿意审得更多。

还要感谢Jim Gray和Ken Henderson，你们俩是了不起的SQL Server宣传员，在过去这些年里一直启发着我。

即使我已经在这里提到了这么多人，我仍然担心会漏掉一些人。果真如此的话，我要补一句：感谢你们！

Louis Davidson

① Shoe memories，作者开玩笑，他为被感谢者买过鞋。——译者注

目　录

第1章 数据库概念简介

这下她明白了，敌人如魔鬼般的狡诈，正在于给谎言掺入一点真话，如此一来，谎言就更以假乱真。

——《纳尼亚传奇：最后的战役》，C. S. Lewis著

在作为数据架构师的职业生涯中，从最开始到昨天为止（不管你什么时候看到这些句子，我这句话基本上都不可能会变化），我无数次遭遇一个完全难以克服的现实。向一个以墙上的钟点考评所有项目的管理体系兜售“做对事情”的概念从来都不轻松。不光如此，通常情况下，实现功能的程序员人数往往比做数据库的多若干倍，他们边等着用数据库，边对管理层抱怨说，不到数据库设计完成或至少开始实现，他们什么事都干不了。如果这还不够，现在介绍一下项目组的思维定势，具体用词可能略有偏差：“数据库设计不那么重要。”

这话往往并非明确地用这些词说出来（虽然也确实有这样说的时候）。一般说来，在人们大谈用户界面（UI）应该如何显示某个东西，以及某按钮应该放到哪儿的时候，类似的话就混杂于其中。系统设计从一开始就让人感觉像是追求“照片般真实”的艺术家在画一幅画，而不是在进行一个需要遵循坚实工程实践的项目，先回答“需要做什么”再回答“如何做”的问题。更糟糕的是，由于数据库是几乎所有软件项目的骨干成分，这使得情况更加复杂且不易看清，往往快到项目开发阶段结束时，事情才暴露出狰狞的真面目。如果你在参与数据库设计，那么非常重要的是，你需要知道如何回答一个基本问题：“为什么？”

这样来看问题：你愿意开车在一座由某位不懂物理的工程师设计出来的桥上行驶吗？或者，你愿意乘坐一架由某位不懂飞行基本原理的人士所设计出来的飞机吗？听起来非常荒谬，是不是？那么，你愿意把自己的重要数据存放在一个由某位不懂数据库设计基本原理的人士所设计出来的数据库中吗？

本书的前4章致力于讨论关系数据库设计的不同阶段，以及如何有效地执行每个阶段的工作，从而能够获得一个良好的最终设计，既能满足业务需求又能保证数据库中数据的完整性。然而，在满腔热情地投入设计过程之前，需要探讨几个核心的关系数据库概念。因此，本章讨论如下主题。

- 数据库设计阶段：1.1节是对关系数据库设计中4个不同设计阶段的概览：概念阶段、逻辑阶段、实现阶段和物理阶段。出于时间和预算方面的考虑，通常人们都想跳过数据库设计的前期阶段，直接进入实现阶段。本章将解释为什么跳过任何一个阶段，或者跳过所有阶段会造成不完整和/或不正确的设计。同时，这样做将无法形成一个能支持高性能查询和报表功能的设计。