The Guru's Guide to SQL Server Stored Procedures, XML, and HTML

SQL Server 存储过程、XML 和 HTML 高级指南

Covers .NET!

Ken Henderson 著　康博 译

清华大学出版社
http://www.tup.com.cn
http://www.tup.tsinghua.edu.cn

Pearson Education

SQL Server 存储过程、XML 和 HTML 高级指南

Ken Henderson 著
康 博 译

清华大学出版社

(京) 新登字 158 号
北京市版权局著作权合同登记号：01-2002-3016

内 容 简 介

本书对 SQL Server 存储过程程序设计做了全面的概述，并结合具体示例，阐述了这种程序设计理念，给出实际程序设计问题的解决方案。全书分为 4 部分，共 26 章。具体而言，除主要对用户定义函数、视图、触发器、扩展过程、错误处理、OLE 自动化、数据库设计和 HTML、XML 等做了详细讲解外，还对 SQL Server 支持的.NET 特性进行了介绍。

本书包含大量的 SQL 脚本，高质量的范例代码，非常实用，本书适合于数据库开发人员及数据库管理员阅读。

图书在版编目(CIP)数据

SQL Server 存储过程、XML 和 HTML 高级指南/ (美)享得森著；康博译. —北京：清华大学出版社，2002
书名原文：The Guru's Guide to SQL Server Stored Procedures，XML，and HTML
ISBN 7-302-05824-5

Ⅰ.S... Ⅱ. ①享... ②康... Ⅲ. ①关系数据库-数据库管理系统，SQL Server ②可扩充语言，XML-程序设计 ③超文本标记语言，HTML-程序设计 Ⅳ.TP311.13②TP312

中国版本图书馆 CIP 数据核字(2002)第 062555 号

出 版 者：清华大学出版社(北京清华大学学研大厦，邮政编码：100084)
http://www.tup.com.cn　　http://www.tup.tsinghua.edu.cn
责任编辑：郭东青
印 刷 者：北京昌平环球印刷厂
发 行 者：新华书店总店北京发行所
开　　本：787×1092 1/16　**印张**：36.75　**字数**：940 千字
版　　次：2002 年 10 月第 1 版　2002 年 10 月第 1 次印刷
书　　号：ISBN 7-302-05824-5/TP · 3448
印　　数：0001 ~ 4000
定　　价：73.00 元

序

对于任何立志成为一名专业的 SQL Server 应用程序开发人员的读者而言，这是一本非常重要的书。

我荣幸地担任了 Microsoft SQL Server 的首任开发经理。在供职于该团队的 11 年多的时间中，我聘用了该产品的许多核心开发人员，并与他们一起工作。我从中认识到杰出开发人员的许多共同特征：热情。软件开发天才是 1%的灵感加 99%的努力(套用爱因斯坦语)。他们中存在一种永不满足的狂热，对完美的不懈追求。专业开发人员编写高效的、符合业界高标准的、文档完备的代码，不只允许开发人员自己，也允许其他人在此后若干年内能够实施维护、理解和升级。许多刚从计算机科学系毕业且绝顶聪明的学生需要几年的时间才会成熟，从一个热情且聪明的程序员到一个真正专业的开发人员(有些人永远都不能完成这种转变，而只能维持短暂职业生涯——至少在我们的团队中是如此)。他们熟悉自己专长领域内的重要算法，掌握开发语言与工具。成为一名真正专业的开发人员不存在捷径。在提交代码之前，每一种临界条件都必须考虑到，有应对措施且经过测试。代码中的断言是不可或缺的。返回代码都必须得到检查。大量的注释是必要的。代码重用非常关键。代码检查和遍历深受欢迎，也是一种例行事务。当你提交代码时，就如同给一件艺术品签名。它符合您自己的严格标准，而且需要不断地学习。所谓专业人员是其职业技能的终生学徒。在该行业中，不进则退。

我喜欢阅读本书的手稿。当您阅读此书时，我前面提及的全部特性跃然纸上。在供职于 Microsoft 期间，我经常要与 SQL Server 的客户打交道。常常出现某个应用程序不能正常运行或者表现不如开发人员预期的情况。因为诸如此类的应用程序，我赢得了“补丁先生”的美誉，并且帮助将某些濒临失败的应用程序扭转为巨大的成功。我喜欢自己这部分工作。我享受与客户打交道的乐趣，尽管这通常是始于剑拔弩张的局面。SQL Server 往往成为欠完善应用的替罪羔羊。的确有时 SQL Server 本身存在某种错误或引发问题的缺陷，但是，多年来，伴随每个新版本的问世，该产品的成熟度、健壮性和速度都呈指数增长，并且这种态势将一如继往。然而，即使在 1.x 版本的草创时期，通过更好地利用 SQL Server 存储过程，也可以极大地改善性能或提高简明性。由于对存储过程的更巧妙应用，成百倍或更多的性能增长也并非鲜见。一旦我们让那些抱怨的经理们离开那些正在检查存在问题的应用程序的某个房间，就意味着可能产生了重大进展。尽管我管理着一个大型开发团队，甚至还怀揣着顶尖大学的 MBA 证书，但是，我从不把自己看作是管理者。我总是以一名开发人员自居——只有开发人员才知道如何与其他开发人员交流，并且尊重彼此的绝妙创意和超群的技艺。这也正是您将从本书中领略到的：一位经验丰富的开发人员跟其他开发人员的倾心交谈。Ken 以清晰的、第一人称的方式编写本书，此举在技术书籍中实不多见。他敢于坦露自己的主张，而所有真正的开发人员知道，在软件开发中，主张与信念是至关重要的，因为很少有让人们无师自通的真知灼见。以我过去的经验，在存在问题的应用中，问题的症结往往在于这样简单的事实：开发人员在对待 SQL Server 时，

并不像他们对待 C、C++或其他程序设计语言那般认真和重视，他们的应用程序中其余的代码往往是用这些语言编写的。在许多情况下，他们根本上只对 SQL Server 和数据库应用浅尝辄止。而这一切对于那些才思敏捷的人往往是司空见惯的，但是，至少按照我的观点，他们还远未达到可以声称精通一种不熟悉的领域，或者成为真正的行家里手的水平。现实是，如果开发人员在投身于编写他们的第一个 SQL Server 应用程序之前，已经阅读、研究并真正领会了本书，这些状况或遭遇中的大多数就永远也不会发生。请花时间阅读并研究此书吧！成为一名专业 SQL Server 应用开发人员，这是必由之路。

Ken，祝贺您，而且还要感谢您编写了这本书。就像您的前一本书 The Guru's Guide to Transact-SQL，我希望 10 年前就已经有了本书。

——Ron Soukup

2001 年 9 月

前　言

本书的要旨是，用 Transact-SQL 创建存储过程就如同用任何其他语言开发程序。如同用其他程序设计语言成功开发所需要的那样，创建存储过程需要同样的技巧、规划、体察入微和对技术的总体把握。为了掌握 Transact-SQL，人们必须首先掌握软件开发的基础，然后在这种基础上，将 Transact-SQL 作为一种程序设计语言使用。本书教您如何实现该任务。

如果您是一位 SQL Server 初学者，也许有比本书更好的书可以向您介绍 Transact-SQL 存储过程程序设计。本书实际上适合中-高级开发人员。它假设您已经知道如何编写 Transact-SQL 查询程序，如何创建存储过程。除了某些讨论开始时的介绍性说明，本书很少提供入门级的指导。它针对具备中高级技术的开发人员，他们希望成为更好的存储过程程序员——希望学习更高级别的与 Transact-SQL、存储过程程序设计和 XML 相关的高级软件技术。

笔者的前一本书 The Guru's Guide to Transact-SQL 的卷首题词引自我的一位朋友——知名作家和演讲者 Joe Celko。为了掌握诸如 SQL 之类的非过程化语言，抛弃过程化程序设计是至关重要的。同时，我赞成 Joe 的观点：按照过程化方式编写 Transact-SQL 代码是编写理想的 Transact-SQL 代码的最大的惟一的障碍。当笔者写作第一部 Guru's Guide 时，我坚信，之所以那些使用其他语言的称职开发人员在尝试编写 Transact-SQL 代码时会遇到困难的主要原因是，试图与使用 C++的相同方式编写 Transact-SQL。他们的整个方法都错了，我推理这也是他们会遇到问题的原因。我认为他们没有像数据库程序员那样思考；相反，他们是像传统程序员那样思考，这种思维方式不适于数据库程序设计。我认为是如此。

从那时开始，我改变了看法。我曾看过一个访谈，其中 Edward Van Halen 声称，一个乐队的音乐唱片是该乐队在特定时空(音乐上和其他方面)的快照，书籍也如此。The Guru's Guide to Transact-SQL 就是我大约在 1998 年和 1999 年的全部写照，其间我写作了该书。此后，我对过程化程序设计与 Transact-SQL 之间关系的思考有了些许进展(或者我倾向于这样思考)。为什么？还是让我给大家讲一则小故事吧……

在我写作 The Guru's Guide to Transact-SQL 一书的两年期间的某个时候，该书的某位技术审阅者给我写信，询问几年前我为自己在 Sybase Developer's Journal 的专栏所写的一篇介绍 Transact-SQL 中某些 bitmask 技巧的文章。他想要知道我是否给他寄该文的一个副本，因为他正在从事某些与 bitmask 相关的工作，并且想要使用我曾介绍过的某项技术。我在那篇文章中找遍了也没有在我所使用的各种计算机上发现相关内容。我写那些内容的机器已经很久不用了，该机器我进行了磁带备份——所以那些内容一定还在最初的位置上。

最终通过在互联网上的搜索找到了这篇老古董，并将它转寄给需要它的同事。出于某种兴趣，我在自己的桌前坐下并阅读这篇文章片刻(大多数作者真正喜欢阅读自己曾写过的东西，不管它是多么古老，也不管它到底讲什么)。我独自思忖：是什么使我对 Transcact-SQL 中这种 bit-twiddling 技术着迷？为什么开始时我要考虑这些技术？我想要知道对这篇文章中这种技术

进行探讨的原因。我的答案是，如果我能指出探讨这种技术的原因和方式，也许我就能揭开进行这种创新的秘密，或至少找出这种困惑的原因。也许我就会作为一名更高级的 Transact-SQL 程序员。

我为此考虑了几天，最后想到了提出这种 bit-twiddling 技术的原因。我的结论是：我尽可能地要使自己相信这种想法都来自我自己，而我从 Transact-SQL 所获得的许多“发现”都是学术界间接传授的结果。这主要是源自我使用其他语言的经验，这种经验使我经过多年研究提出了许多具有创新的编码技术。我对 Transact-SQL 所做的大部分发现源自我使用传统的程序设计语言(诸如 Pascal、C/C++、汇编语言和其他语言)进行工作时在我的大脑中形成的雏形。我意识到在 Transact-SQL 程序设计这个相对较新的领域内，很少有真正具有创新意义的技术产生。毕竟像 C 和 Pascal 之类的语言比 Transact-SQL 早出现许多年——而像 COBOL 和 BASIC 之类的语言甚至更早。我们看到的不是计算领域的新问题，而是对同样的旧问题的新的解决方案。在 Transact-SQL 或 SQL Server 出现以前，人们已经解决这些问题很长时间了。当然，在软件工程领域内的大部分发现已经出现，而我们对 Transact-SQL 的创新成果只是在前人基础上努力的结果。

Andrew Hunt 和 Dave Thomas 在 Pragmatic Programmer 一书中强烈推荐渴望成为好的程序员的人每年应至少学习一门新的程序设计语言。我这里也作此推荐。如果您想精通 Transact-SQL 存储过程程序设计，就应该首先精通程序设计本身。程序设计、编码、软件工程——无论您如何称呼，都需要很长时间和需要掌握许多语言才能精通。像一个学习军事艺术的人在能获得高级地位前必须首先精通各方面的军事艺术一样。一个要精通 Transact-SQL 的程序员通常在希望能精通 Transact-SQL 前必须精通程序设计的各个方面。

各学科共同努力的结果所产生的美好前景和交叉性知识的传播，促使大学为学生讲授除主要研究领域外其他领域的知识。通过研究其他人做事的方式，您会看出在您的领域和其他领域间的许多相似点和差异性，您会对这些相似和差异有深刻的洞察，您会学会可能是以一种以前没有尝试过的方式把您在某方面所发现的东西应用到完全在您的领域外的其他领域。换句话说，您学会了创新。通过规纳各大学所持的观点，您对自己领域的理解会更加具有整体性：您开始更深刻地理解其哲学意义，并领会它适应事物实质性的时机。

我认为通过研究 SQL Server 之外的语言和技术也可获得同样的洞察力。即使我没有在汇编语言方面做过什么工作，即使没有对类似 Steve Gibson 这样的高手的工作进行过研究，我也不会对我最初写的那篇文章中的 bit-twiddling 技术目瞪口呆。但是如果没有我在 Pascal 和 Delphi 方面的工作，如果没有对如 Anders Hejlsberg 和 Kim Kokkonen 这样高手的代码进行过研究，我就不会在研究多年的 Transact-SQL 方面提出许多好的技术，包括您在本书中所看到的许多数据操作过程。我对 Transact-SQL 中一般设计模式的研究，受到 Erich Gamma 及其同事的 Design Patterns 一书的鼓舞，在我使用 C++和 Object Pascal 进行工作时总把该书放在手边。Brian Kernighan 和 Rob Pike 的 The Parctice of Programming 一书对我的程序设计习惯产生很大的影响。因为 Steve McConnell 的 After the Gold Rush 一书和 Kent Beck 的 Extreme Programming Explained 一书的影响，我也是一贯支持测试的。另外，由于受 Martin Fowler 的 Refactoring:Improving the Design of Existing Code 一书的影响，我也非常赞同再评估的价值。本书讨论的许多算法受到 Donald Knuth 的三卷书 The Art of Computer Programming 和 Jon Bentley 的 Programming Pearls

一书以及许多其他书籍的启发。

这些书本身没有一本是关于 Transact-SQL 甚至是 SQL Server 的。他们也都未能阐述很容易就被转换成类似 SQL 的面向集合的语言的技术。然而，它们是关于程序设计的，我在其他语言方面的工作从中获得了许多知识。我从各学科间的工作中受益、我也从我在使用 Transact-SQL 和使用其他语言的交叉工作中受益。而且从这项工作为程序员提供的观点中受益。我认为您也会如此。

因此，不是说为了精通 Transact-SQL 就必须放弃那种沉重的过程性程序设计，我反而鼓励您尝试采用其他的语言和工具。每年学一种——它可以是您还没掌握的任何一种语言或工具——如Visual Basic、Delphi、Ruby、C#，C++或 Java 等等。使用新语言来完成几个项目，理想情况是(但不是必需的)以某种方式将之跟 SQL Server 结合起来一起进行。购买所需的书籍，阅读新闻报导，进行研究，构建自己的软件。您会很惊奇地发现您竟学会了这么多的程序设计知识，作为开发人员的经验也在不断增长。

然后，有时通过这些研究项目，考虑一下如何把所学到的知识应用到 Transact-SQL 开发工作中。SQL Server 如何采纳您正在研究的以工具为特色的语言元素呢？它又是如何实现您在新语言中发现的特别有用的功能呢？区别在哪里？比如说，OLE 自动化在 Transact-SQL 和 Delphi 中的差别何在？假定 Transact-SQL 像所有的 SQL Server 一样，都由 C 和 C++编写，哪种语言声明会更符合其最初状态？

这样困惑了几年后，在您获得了 SQL Server 之外世界的实际经验后，您就会在掌握 Transact-SQL 和存储过程程序设计过程中很顺利地获得必要的工具。您会把软件工程视作一项纪律；您会因此而爱上程序设计这项工作。这是精通任何程序设计语言包括 Transact-SQL 的关键所在。

因此，尽管我对 Joe Celko 仍有歉意，但我也不再相信过程性程序设计是好的 Transact-SQL 编码的最大障碍。正好相反，没有真正掌握一门语言的缺点和优点——这些使事物具有独特性的东西——才是用它构建好软件的最大障碍。您只能通过学科间的交叉工作获得才能正确评估这些软件的缺点和优点。对于 Transact-SQL，它的优点是面向集合的开发，基本缺点是自上到下的程序设计方式。这不意味着您使用 Transact-SQL 只能编写面向集合的程序，编写过程 Transact-SQL 代码是一种蛮干的方式。毕竟我们称之为存储过程。这只意味着您的编码风格和 Transact-SQL 不同，就像和编写 Visual Basic 的不同一样。不止 Transact-SQL 是这样，许多语言也是如此。许多语言具有使其具有惟一性的细微差别和习惯用法，您不会像编写 Visual Basic 一样编写 C++。针对具体工作使用正确的工具，发挥工具的优点，避免它的缺点，不但要掌握工具本身，还应进行更一般的软件开发，这样才会对其优缺点越来越熟悉。您的目的是要成为一位高明的程序员，不止是一个存储过程专家。

我和 Joe 在 San Francisco 共进晚餐时曾说过：我仍然认为在很长一段时间内 C#是程序设计的最理想语言。

Ken Henderson

2001 年 1 月

引　言

本书不仅提供编码技巧和语法等方面的内容，还讲授 Transact-SQL 程序设计的基本理念，并介绍如何将此理念应用于提升编码技巧和解决实际问题。本书持如下观点："为什么做"至少与"如何做"同等重要，以及学习 Transact-SQL 的一种折衷方法——既强调理论也强调实践的方法——优于非折衷方法。

您将会注意到，本书深入研究许多传统上不直接与存储过程程序设计相关的主题。本书论及 XML、HTML、.NET 和许多其他似乎为辅助性的主题。这样做的原因非常简单：在创建现实中应用的软件时，人们倾向使用许多不同的技术。创建存储过程不是在真空中进行的。本书认可这种事实，并从 SQL Server 存储过程开发人员的角度探讨许多这样的相关技术。

XML 在其中也起着重要的作用，因为人们往往借助存储过程和 T-SQL 查询访问 SQL Server 的 XML 特性。HTML 也是如此。本书讨论.NET，因为 Microsoft 已宣布，下一个版本的 SQL Server 将允许使用新的.NET 语言 C#和 VB.NET 开发存储过程。笔者感觉一本关于 SQL Server 开发的书(即使它只集中于存储过程开发)至少应该提及.NET，并讨论它提供给 SQL Server 开发的许多激动人心的特性。也许 Transact-SQL 程序员在将自己的代码移植到.NET 和公共语言运行时(Common Language Runtime)时会存在某些阻力。通过提供广泛的技术综述，以及向开发人员提供他们可以对此预期些什么的忠告，笔者认为，像这样的一本书应该有助于提供示范。

本书在较广的软件开发背景下讨论 Transact-SQL 开发。它深入研究许多基础性软件工程概念：最佳实践、代码管理策略、设计模型和范例、软件测试等等。使用任何程序设计语言，包括 Transact-SQL 时，这些对于有效的开发都是重要的。本书试图把 Transact-SQL 提升到传统语言(如 Visual Basic 和 C++)的地位。它向读者讲授如何将 Transact-SQL 作为一门程序设计语言来掌握，而不是仅仅作为一种查询语言或一种 SQL Server 脚本工具。

本书去掉了过多的屏幕截图和其他在计算机书籍中常见的补白图案。笔者已有意避免因不必要的插图、框图和类似的内容而增加本书的篇幅。相反，笔者只在需要的地方包含了一些有用的插图，其他的则都被删除。笔者已在全书中尽量减少查询结果，但仍然尽可能地保留了完整的代码清单。人们讨厌臃肿的技术书籍，因此，我尽最大努力避免自己编写这样的书籍。

在开始编写本书时，我在心中拥有下列设计目标：

- 从我的上一本书 The Guru's Guide to Transact-SQL 还没来得及讲的地方开始，为那些掌握了该书的读者提供一个进一步学习的途径。
- 讲述了 Transact-SQL 程序设计理念，而不仅仅是介绍语法、技巧或隐藏的窍门。
- 说明 Transact-SQL 程序设计和用其他计算机语言进行软件开发之间的异同。
- 尽量避免成为在线 SQL Server 书籍的汇编材料。
- 完整地探讨与 Transact-SQL 存储过程程序设计相关的主题(如可扩展过程、数据库设计和 XML)，在与此类似的书籍中，这些主题往往被忽略或只简单讨论。

- 在每章之内和整本书中，都循着由简单到高级的方式向下进行。
- 编写一本内容均衡的书籍，Transact-SQL 存储过程程序设计的理论部分与实践部分内容篇幅相当。
- 在编写每章时都保证其独立性。为此，要尽量少地依赖其他章节的内容。
- 避免在计算机书籍中常见的、过多的屏幕截图和其他类型的补白图案。
- 创新性。给出以前没讨论过的 Transact-SQL 程序设计的技术、方法和思考方式。加强特定的 Transact-SQL 程序设计和更具一般性的软件工程方面的艺术性。
- 给出大量示例代码，不但要告诉读者如何做，还要给出其过程。
- 在各章文字内给出完整的代码范例。这样阅读本书就不再需要计算机或 CD-ROM。
- 如果格式化必要，且这样做又不会减损对本书素材的讨论，则精简各章内的代码列表。
- 给出具有实际意义的代码范例，它们脱离本书也具有内在价值。给出读者如果必要可以在其产品服务中真实使用的示例代码。
- 阐明计算机书籍也可以写得很好和易于理解，同时还要包括可用的技术信息。说明好的读物和好的技术不是互相排斥的。
- 使用流行的程序设计技术，特别强调跟 ANSI 的兼容性，当前版本的特性以及增强的内容。
- 给出容易理解的轻松的解说词，不强调太严肃。读者读本书时它就像您忠实的伙伴。编写方式就像以口语化方式在交谈。

这些是我在写作本书时制定的目标。一个作者的工作有其盛衰期，每天我都有些要完成的目标，有时我完成了，有时没完成。关键是失败时要不断尝试——找出错误在哪里，以及如何能做得更好。挑战是要不断地摆脱沮丧心情和克服其他障碍，始终踌躇满志。当您克服了这些障碍，看到自己完成的伟大事业时，这就是对您的奖励，它会使您的才能上升到更高的层次。

关于范例数据库

为使读者更容易理解，本书使用了大量的 SQL Server 的 Northwind 和 pubs 范例数据库。使用已成产品的范例数据库，将有助于避免只是为了理解那些很容易通过书中图框就可以讲解的概念而创建许多定制对象。您几乎总是能通过周围的注释和代码本身判断出某特定范例使用的是哪个数据库。Northwind 数据库的使用比 pubs 频繁，因此，除非特别声明，或有疑问，使用的都是 Northwind 数据库。

通常，对这些数据库的修改是在事务内进行，以便可以将之恢复。但为安全起见，在每章改动后应尽可能重新创建它们。重新创建的脚本(instnwnd.sql 和 instpubs.sql)在 SQL Server 根目录下的\install 子目录中可以找到。

目　录

第 1 部分　基　础

第2部分　目　　标