



Beginning T-SQL with Microsoft SQL Server 2005 and 2008

T-SQL编程入门经典

(涵盖SQL Server 2008 & 2005)

(美) Paul Turley 著
Dan Wood
吴伟平 译



清华大学出版社

T-SQL 编程入门经典

(涵盖 SQL Server 2008 & 2005)

Paul Turley
(美) Dan Wood 著
吴伟平 译

清华大学出版社

北 京

Paul Turley, Dan Wood
Beginning T-SQL with Microsoft SQL Server 2005 and 2008
EISBN: 978-0-470-25703-6
Copyright © 2008 by Wiley Publishing, Inc.
All Rights Reserved. This translation published under license.

本书中文简体字版由 Wiley Publishing, Inc. 授权清华大学出版社出版。未经出版者书面许可, 不得以任何方式复制或抄袭本书内容。

北京市版权局著作权合同登记号 图字: 01-2009-0506

本书封面贴有 Wiley 公司防伪标签, 无标签者不得销售。
版权所有, 侵权必究。侵权举报电话: 010-62782989 13701121933

图书在版编目(CIP)数据

T-SQL 编程入门经典(涵盖 SQL Server 2008 & 2005)/(美)图雷(Turley, P.), (美)伍德(Wood, D.)著; 吴伟平 译.
—北京: 清华大学出版社, 2009.8

书名原文: Beginning T-SQL with Microsoft SQL Server 2005 and 2008

ISBN 978-7-302-20535-7

I.T… II.①图… ②伍… ③吴… III.关系数据库—数据库管理系统, SQL Server 2005、2008 IV.TP311.138

中国版本图书馆 CIP 数据核字(2009)第 111073 号

责任编辑: 王 军 李 阳

装帧设计: 孔祥丰

责任校对: 成凤进

责任印制: 何 芊

出版发行: 清华大学出版社

地 址: 北京清华大学学研大厦 A 座

<http://www.tup.com.cn>

邮 编: 100084

社 总 机: 010-62770175

邮 购: 010-62786544

投稿与读者服务: 010-62776969, c-service@tup.tsinghua.edu.cn

质 量 反 馈: 010-62772015, zhiliang@tup.tsinghua.edu.cn

印 刷 者: 北京密云胶印厂

装 订 者: 三河市新茂装订有限公司

经 销: 全国新华书店

开 本: 185×260 印 张: 35.25 字 数: 858 千字

版 次: 2009 年 8 月第 1 版 印 次: 2009 年 8 月第 1 次印刷

印 数: 1~4000

定 价: 69.80 元

本书如存在文字不清、漏印、缺页、倒页、脱页等印装质量问题, 请与清华大学出版社出版部联系调换。联系电话: (010)62770177 转 3103 产品编号: 030072-01

前言

欢迎进入 SQL Server 2005 和 2008 的 Transact-SQL 编程世界。Transact-SQL 简称为 T-SQL，是微软公司为 ANSI 标准 SQL 数据库查询语言提供的强大的实现方案，可以在关系型数据库管理系统(RDBMS)中检索、处理和添加数据。

读者可能对 SQL 的用途有基本的了解，但不理解关系数据库的概念和 SQL 的作用。本书将帮助读者理解并开始使用核心的关系数据库概念，最后还要在真实的 T-SQL 查询应用程序中进一步强化这些概念。

如果读者熟悉关系数据库概念，但没有接触过 Microsoft SQL Server 或 T-SQL 语言，本书将从基础内容开始讲起。如果读者熟悉 SQL Server 的以前版本，本书可以使读者快速熟悉最新的特性。如果读者了解 SQL Server 2005，就可以通过本书学到 SQL Server 2008 的新功能。

一本流行的在线百科全书列出了目前在用的约 800 种不同的编程语言。这些语言可以为不同类型的计算机系统和专业化设备开发不同类型的应用程序。毋庸置疑，在我们所生活的这个信息时代，有非常多的软件。而且编程语言演化得也非常快，但软件业不变的特性之一是大多数商务应用程序都要读取、存储和处理数据——存储在关系数据库中的数据。要使用 Microsoft SQL Server，就必须学习和使用 T-SQL。

的确，SQL 是业界几十年不变的几个标准语言之一。T-SQL 的功能也随着每个 SQL Server 新版本增加的特性而不断扩展，本书的概念和练习将帮助读者理解和使用其核心语言及其最新功能。

本书读者对象

许多不同领域的 IT 人士都在使用 T-SQL。本书的目的是给这些 IT 人士提供可操作的数据库解决方案设计、数据库应用程序开发、报表和商务智能解决方案方面的指导和参考。

对于数据库解决方案设计人员，本书全面介绍了数据库建模、设计、对象管理、查询设计和高级查询概念，是一本内容丰富的参考资料。

对于编写代码来管理和使用 SQL Server 数据的应用程序开发人员而言，本书涵盖了基本的数据管理、简单和高级的查询设计等内容。还通过几个示例提供了易用的代码，使读者很容易地开始编写内嵌了 T-SQL 查询的应用程序。

对于报表设计人员，本书是一个报表查询设计的参考资料，它全面介绍了基本查询概念，学习编写高效的查询，来支持商务报表和高级分析。

最后，对于不熟悉 SQL Server 的数据库管理员，本书介绍了数据库管理的主要论题，

以支持其他团队成员的工作。要了解数据库对象管理和安全概念等基础知识之外的其他内容,建议读者阅读本书作者与他人合作编著的另外两本书 *Beginning SQL Server 2005 Administration* 和 *Beginning SQL Server 2008 Administration*(Wrox 出版社出版)。

本书内容

本书介绍了 T-SQL 语言及其许多用法,初级和中级用户可以把它当作一个内容全面的参考资料。编著本书的目的是介绍 T-SQL 的所有基础知识,并较深入地探讨 T-SQL 最常见的应用。根据读者的职位和技术水平,可以分别查阅 Wrox 出版社出版的有关 Microsoft SQL Server 入门和高级编程经典系列的其他图书。

本书将学习:

- 使用 T-SQL 创建工具,来管理不同规模、不同范围和不同作用的数据库
- 使用视图、用户定义函数和存储过程的各种编程技巧
- 优化查询性能的方法
- 创建数据库,作为以后开发应用程序的基础

本书结构

由于本书的每个部分都把主题组织为逻辑组,所以本书很容易阅读,读者也可以把本书用作特殊主题的参考指南。

本书首先介绍了 T-SQL 语言和数据管理系统,然后讨论了 SQL Server 的产品体系。第一部分包含第 1~3 章,探讨了 SQL Server 产品体系结构的基本知识和关系数据库设计原则,最后讨论了 SQL Server 管理员和开发工具。

第二部分包含第 4~9 章,介绍了 T-SQL 语言,并讲述了数据检索、SQL 函数、聚合和分组、以及多表查询的核心组件。首先介绍基础知识和 SQL SELECT 语句的核心结构,之后讨论 SELECT 查询的高级形式。

第 10 章介绍事务处理和数据处理,读者将学习 INSERT、UPDATE 和 DELETE 语句如何与关系数据库引擎及事务日志交互操作,以锁定和修改数据行,并保证数据的完整性。我们不仅要学习使用正确的 SQL 语法,还要用简单的术语说明这个过程的工作原理。

总结部分的高级主题阐述如何创建和管理 T-SQL 编程对象,包括视图、函数和存储过程,学习优化查询的性能,在应用程序设计中使用 T-SQL,把查询设计的基础知识应用于真实的事务解决方案。第 15 章包含一个完整的教程,介绍了如何使用 SQL Server 2008 Reporting Services 可视化我们创建的 T-SQL 查询中的数据。

本书最后是一个参考附录,讨论了命令语法、系统存储过程、信息模式视图、文件系统命令和系统管理命令等内容。

使用本书的要求

本书的内容可应用于 Microsoft SQL Server 2005 和 2008 的所有版本。要使用本书讨论的所有特性,建议安装开发版,也可以使用企业版、标准版或工作组版。

SQL Server 2005 或 2008 开发版也可以安装在运行 Windows 2000、Windows XP 或 Windows Vista 系统的台式计算机上。还可以在 Windows Server 2000、Windows Server 2003 或 Windows Server 2008 上使用企业版或标准版。必须在台式计算机上安装 SQL Server 客户工具,台式计算机还必须安装 SQL Server 关系数据库服务器,或者在带有网络连接和访问权限的远程服务器上安装 SQL Server 关系数据库服务器。

访问 www.microsoft.com/sql, 可以获得最新服务包、特定兼容包的信息,了解建议的最低系统需求。

本书的示例可使用从 Microsoft 下载的示例数据库:用于 SQL Server 2005 的示例数据库是 AdventureWorks,用于 SQL Server 2008 的示例数据库是 AdventureWorks2008。这两个数据库的结构大不相同,所以本书为这两个版本的数据库提供了不同的代码示例。

第 15 章还使用了 SQL Server 2008 示例数据库 AdventureWorks2008DW。

要下载和安装这两个示例数据库,可以访问 www.codeplex.com。

源代码

在读者学习本书中的示例时,可以手工输入所有的代码,也可以使用本书附带的源代码文件。本书使用的所有源代码都可以从本书合作站点 <http://www.wrox.com/> 或 www.tupwk.com.cn/downpage 上下载。登录到站点 <http://www.wrox.com/>, 使用 Search 工具或使用书名列表就可以找到本书。接着单击本书细目页面上的 Download Code 链接,就可以获得所有的源代码。

注释:

由于许多图书的标题都很类似,所以按 ISBN 搜索是最简单的,本书英文版的 ISBN 是 978-0-470-25703-6。

在下载了代码后,只需用自己喜欢的解压缩软件对它进行解压缩即可。另外,也可以进入 <http://www.wrox.com/dynamic/books/download.aspx> 上的 Wrox 代码下载主页,查看本书和其他 Wrox 图书的所有代码。

勘误表

尽管我们已经尽了各种努力来保证文章或代码中不出现错误,但是错误总是难免的,如果您在本书中找到了错误,例如拼写错误或代码错误,请告诉我们,我们将非常感激。通过勘误表,可以让其他读者避免受挫,当然,这还有助于提供更高质量的信息。

请给 wkservice@vip.163.com 发电子邮件, 我们会检查您的反馈信息, 如果是正确的, 我们将在本书的后续版本中采用。

要在网站上找到本书英文版的勘误表, 可以登录 <http://www.wrox.com>, 通过 Search 工具或书名列表查找本书, 然后在本书的细目页面上, 单击 Book Errata 链接。在这个页面上可以查看到 Wrox 编辑已提交和粘贴的所有勘误项。完整的图书列表还包括每本书的勘误表, 网址是 www.wrox.com/misc-pages/booklist.shtml。

P2P.WROX.COM

要与作者和同行讨论, 请加入 p2p.wrox.com 上的 P2P 论坛。这个论坛是一个基于 Web 的系统, 便于您张贴与 Wrox 图书相关的消息和相关技术, 与其他读者和技术用户交流心得。该论坛提供了订阅功能, 当论坛上有新的消息时, 它可以给您传送感兴趣的论题。Wrox 作者、编辑和其他业界专家和读者都会到这个论坛上来探讨问题。

在 <http://p2p.wrox.com> 上, 有许多不同的论坛, 它们不仅有助于阅读本书, 还有助于开发自己的应用程序。要加入论坛, 可以遵循下面的步骤:

- (1) 进入 p2p.wrox.com, 单击 Register 链接。
- (2) 阅读使用协议, 并单击 Agree 按钮。
- (3) 填写加入该论坛所需要的信息和自己希望提供的其他信息, 单击 Submit 按钮。
- (4) 您会收到一封电子邮件, 其中的信息描述了如何验证账户, 完成加入过程。

注释:

不加入 P2P 也可以阅读论坛上的消息, 但要张贴自己的消息, 就必须加入该论坛。

加入论坛后, 就可以张贴新消息, 响应其他用户张贴的消息。可以随时在 Web 上阅读消息。如果要让该网站给自己发送特定论坛中的消息, 可以单击论坛列表中该论坛名旁边的 Subscribe to this Forum 图标。

关于使用 Wrox P2P 的更多信息, 可阅读 P2P FAQ, 也可以在此了解论坛软件的工作情况以及 P2P 和 Wrox 图书的许多常见问题。要阅读 FAQ, 可以在任意 P2P 页面上单击 FAQ 链接。

目 录

第 1 章 T-SQL 和数据管理系统简介	1		
1.1 T-SQL 语言	1		
1.1.1 T-SQL: 是编程语言还是查询语言	2		
1.1.2 SQL Server 2008 中的新特性	3		
1.1.3 数据库管理系统(DBMS)	4		
1.2 作为关系数据库管理系统的 SQL Server	5		
1.2.1 表	5		
1.2.2 关系	12		
1.2.3 RDBMS 与数据完整性	14		
1.3 SQL Server 与其他产品	17		
1.3.1 Microsoft SQL Server	17		
1.3.2 Oracle	18		
1.3.3 IBM DB2	18		
1.3.4 Informix	18		
1.3.5 Sybase SQLAnywhere	18		
1.3.6 Microsoft Access(Jet)	19		
1.3.7 MySQL	19		
1.4 小结	19		
第 2 章 SQL Server 基础	21		
2.1 谁使用 SQL Server	21		
2.2 SQL Server 版本和功能	23		
2.2.1 SQL Server 压缩版	23		
2.2.2 SQL Server Express 版	23		
2.2.3 SQL Server 工作组版	24		
2.2.4 SQL Server 标准版	24		
2.2.5 SQL Server 企业版	24		
2.2.6 关系数据库引擎	24		
2.3 语义	25		
2.3.1 变化的术语	25		
2.3.2 关系	27		
2.3.3 主键	28		
2.3.4 外键	29		
2.4 规范化规则	30		
2.5 应用规范化规则	33		
2.5.1 想一想	35		
2.5.2 多重关联	37		
2.5.3 多值列	38		
2.5.4 规范化还是去规范化	39		
2.5.5 质疑权威	40		
2.6 查询处理的机制	40		
2.7 AdventureWorks 数据库	42		
2.8 小结	42		
第 3 章 SQL Server 工具	45		
3.1 常见的 SQL Server 任务	45		
3.2 SQL Server Management Studio	47		
3.2.1 工具窗口	50		
3.2.2 工具栏	58		
3.2.3 SQL Server Management Studio 配置	64		
3.3 SQL Server Business Intelligence Development Studio	66		
3.4 SQL Server Profiler	67		
3.5 Database Tuning Advisor	67		
3.6 SQL Server Configuration Manager	68		
3.7 命令行工具	68		
3.8 编写查询	70		
3.8.1 脚本选项	72		
3.8.2 使用图形化的查询设计器	73		
3.8.3 使用模板	77		

3.8.4 使用调试功能	79	5.2.8 排在前面的值	140
3.9 小结	84	5.3 小结	142
3.10 习题	85	5.4 习题	142
第4章 T-SQL 语言	87	第6章 SQL 函数	145
4.1 SQL 的本质	88	6.1 函数的组成	146
4.2 从哪儿开始	89	6.1.1 一个论点	146
4.3 数据操纵语言(DML)	89	6.1.2 确定性函数	147
4.3.1 查询是有层次的	89	6.1.3 在函数中使用用户变量	147
4.3.2 基于集合的操作	90	6.1.4 在查询中使用函数	148
4.3.3 基于行的操作	90	6.1.5 嵌套函数	149
4.3.4 查询语法基础	90	6.2 聚合函数	149
4.3.5 命名规范	94	6.2.1 AVG()函数	150
4.3.6 对象的界定	96	6.2.2 COUNT()函数	151
4.3.7 注释脚本	97	6.2.3 MIN()与 MAX()函数	151
4.3.8 使用模板	100	6.2.4 SUM()函数	152
4.3.9 生成脚本	100	6.3 配置变量	152
4.3.10 管理脚本	100	6.3.1 @@ERROR 变量	152
4.3.11 版本控制	101	6.3.2 @@SERVICENAME 变量	154
4.4 数据定义语言	101	6.3.3 @@TOTAL_ERRORS 变量	154
4.4.1 创建表	102	6.3.4 @@TOTAL_READ 变量	154
4.4.2 创建视图	102	6.3.5 @@VERSION 变量	154
4.4.3 创建存储过程	103	6.3.6 错误函数	155
4.4.4 创建触发器	103	6.4 转换函数	156
4.4.5 创建用户自定义函数	104	6.4.1 CAST()函数	157
4.4.6 脚本编写实践	105	6.4.2 CONVERT()函数	159
4.5 数据控制语言	109	6.4.3 STR()函数	161
4.6 小结	111	6.5 游标函数与变量	162
4.7 习题	111	6.5.1 CURSOR_STATUS()函数	162
第5章 数据检索	113	6.5.2 @@CURSOR_ROWS 全局变量	162
5.1 存储与检索	113	6.5.3 @@FETCH_STATUS 全局变量	163
5.2 SELECT 语句	113	6.6 日期函数	163
5.2.1 列的选择	115	6.6.1 DATEADD()函数	163
5.2.2 列的别名	120	6.6.2 DATEDIFF()函数	164
5.2.3 被计算和继承的列	121	6.6.3 DATEPART()与 DATENAME()函数	167
5.2.4 行的过滤	123		
5.2.5 WHERE 子句	123		
5.2.6 使用括号	136		
5.2.7 结果排序	138		

6.6.4	GETDATE()与 GETUTCDATE()函数	167
6.6.5	SYSDATETIME()和 SYSUTCDATETIME() 函数	168
6.6.6	DAY()、MONTH()和 YEAR()函数	168
6.7	字符串操纵函数	168
6.7.1	ASCII()、CHAR()、UNICODE() 和 NCHAR()函数	169
6.7.2	CHARINDEX()和 PATINDEX()函数	171
6.7.3	LEN()函数	172
6.7.4	LEFT()与 RIGHT()函数	172
6.7.5	SUBSTRING()函数	173
6.7.6	LOWER()与 UPPER()函数	175
6.7.7	LTRIM()与 RTRIM()函数	177
6.7.8	REPLACE()函数	177
6.7.9	REPLICATE()与 SPACE() 函数	178
6.7.10	REVERSE()函数	178
6.7.11	STUFF()函数	179
6.7.12	QUOTENAME()函数	179
6.8	数学函数	179
6.9	元数据函数	180
6.10	排列函数	183
6.10.1	ROW_NUMBER()函数	183
6.10.2	RANK()与 DENSE_RANK()函数	183
6.11.3	NTILE(n)函数	184
6.11	安全函数	185
6.12	系统函数与系统变量	186
6.12.1	COALESCE()函数	188
6.12.2	DATALength()函数	188
6.13	系统统计变量	189
6.14	小结	190
6.15	习题	190

第 7 章	聚合与分组	193
7.1	分组还是不分组	193
7.2	使用聚合函数	194
7.2.1	COUNT()函数	195
7.2.2	SUM()函数	195
7.2.3	AVG()函数	196
7.3	理解统计函数	197
7.3.1	STDEV()函数	197
7.3.2	STDEVP()函数	198
7.3.3	VAR()函数	198
7.3.4	VARP()函数	199
7.3.5	用户自定义的聚合函数	199
7.4	数据分组	199
7.4.1	GROUP BY 子句	200
7.4.2	HAVING 子句	204
7.4.3	总计与小计	207
7.4.4	子分组	208
7.4.5	ROLLUP 子句	209
7.4.6	CUBE 子句	210
7.4.7	GROUPING()函数	210
7.4.8	COMPUTE 与 COMPUTE BY 子句	212
7.5	小结	215
7.6	习题	215
第 8 章	多表查询	217
8.1	理解子查询与连接	219
8.1.1	在 WHERE 子句中连接表	219
8.1.2	在 FROM 子句中连接表	220
8.1.3	连接的类型	221
8.1.4	内连接	221
8.1.5	外连接	221
8.1.6	多列连接	225
8.1.7	非同等连接	226
8.1.8	特殊目的的连接运算	227
8.1.9	合并查询	234
8.2	小结	239
8.3	习题	239

第 9 章 高级查询与脚本	241	11.6 习题	318
9.1 子查询	241	第 12 章 T-SQL 编程对象	319
9.1.1 标量表达式	242	12.1 视图	320
9.1.2 交换连接运算	244	12.1.1 虚表	320
9.1.3 互相关联的子查询	250	12.1.2 创建视图	320
9.1.4 子查询业务案例	253	12.1.3 保护数据	331
9.2 Common Table Expressions (CTE)	256	12.1.4 隐藏复杂性	331
9.3 游标	258	12.2 存储过程	334
9.3.1 行集(rowset)与游标操作	258	12.2.1 用存储过程实现带参 数的视图	334
9.3.2 创建与遍历游标	259	12.2.2 使用参数	335
9.4 小结	261	12.2.3 返回值	336
9.5 习题	262	12.2.4 记录的维护	338
第 10 章 数据事务	265	12.2.5 处理和引发错误	342
10.1 事务简介	265	12.2.6 错误消息	343
10.1.1 事务类型	266	12.2.7 处理业务逻辑	352
10.1.2 ACID 测试	267	12.2.8 循环	357
10.1.3 事务日志	267	12.3 用户自定义函数	359
10.1.4 记入日志的操作	268	12.3.1 标量函数	359
10.2 对数据执行 CRUD	268	12.3.2 内嵌表值函数	362
10.2.1 添加记录	268	12.3.3 多语句表值函数	362
10.2.2 修改记录	277	12.4 事务管理	364
10.2.3 删除记录	282	12.5 小结	366
10.2.4 用 MERGE 命令自动完成 插入、更新和删除操作	288	12.6 习题	367
10.3 显式事务	290	第 13 章 创建和管理数据库对象	369
10.4 小结	293	13.1 数据定义语言	369
10.5 习题	293	13.1.1 创建对象	370
第 11 章 高级功能	295	13.1.2 修改对象	370
11.1 数据的透视	295	13.1.3 删除对象	370
11.1.1 PIVOT 操作符	297	13.1.4 对象的命名	370
11.1.2 UNPIVOT 操作符	300	13.1.5 创建 DDL 脚本	372
11.2 全文索引和近似匹配	302	13.1.6 创建表	373
11.3 Microsoft 搜索服务	303	13.1.7 唯一标识符	380
11.3.1 音素匹配	303	13.1.8 约束	381
11.3.2 DIFFERENCE() 函数	305	13.1.9 创建视图	387
11.4 管理并填写目录	305	13.1.10 带索引的视图	390
11.5 小结	317	13.1.11 创建存储过程	391
		13.1.12 CREATE FUNCTION	395

13.2	IF EXISTS	395	15.3	SQL Server 2008 报表服务	444
13.3	数据库对象的安全	396	15.3.1	报表服务的体系结构	444
13.3.1	管理安全性对象	397	15.3.2	SQL Server 2008 报表设计器	445
13.3.2	数据控制语言	398	15.3.3	部署报表	464
13.4	小结	398	15.3.4	用报表管理器查看报表	465
13.5	习题	399	15.3.5	报表数据缓存	465
第 14 章	分析和优化查询性能	401	15.2.6	商业智能和业务报表	466
14.1	数据检索	401	15.3.7	报表应用程序的集成	467
14.2	分析查询	403	15.4	小结	470
14.2.1	会话选项	404	附录 A	命令语法参考	473
14.2.3	图形化执行计划	415	附录 B	系统变量与函数参考	493
14.3	编写高效的 T-SQL (最佳实践方式)	423	附录 C	系统存储过程参考	509
14.4	小结	427	附录 D	信息模式视图参考	525
14.5	习题	428	附录 E	FileStream 对象和语法	537
第 15 章	T-SQL 在应用程序与报表中 的应用	429	附录 F	习题答案	541
15.1	应用程序编程模型	429			
15.2	选择模型	430			
15.2.1	桌面数据库应用程序	432			
15.2.2	客户/服务器数据库 解决方案	433			
15.2.3	n 层组件解决方案	436			
15.2.4	Web 服务器应用程序	441			
15.2.5	多层 Web 服务解决 方案	442			
15.2.6	多系统集成解决方案	442			

第 1 章

T-SQL 和数据管理系统简介

本书的第 1 章将介绍关系数据库在设计和体系结构上的一些基本原理,并对 SQL 语言进行简要描述。如果用户以前从未接触过 SQL 和数据库技术,本章将提供一些基础知识,以确保用户能有效地阅读本书的其余内容。如果用户对关系数据库的概念及 Microsoft 的关系数据库实现和产品已经较为熟悉,可以直接跳到第 2 章: SQL Server 基础,或者第 3 章: SQL Server 工具。这两章介绍了 SQL Server 2005 和 2008 的特性和工具,以及如何使用它们编写 T-SQL。

1.1 T-SQL 语言

我曾经对同事说过,我打算编写 Parker 兄弟的 Trivial Pursuit 的一个新版本: Trivial Pursuit: Geek Edition。本节首先将介绍 T-SQL 语言的一些背景知识,使读者了解 Trivial Pursuit: Geek Edition 中“数据库历史”主题的一些信息。

ANSI(American National Standards Institute,美国国家标准协会)制订了结构化查询语言(SQL)标准,而 T-SQL 是 Microsoft 公司对此标准的一个实现。SQL 最初是由 IBM 的研究员开发的。在 SQL 的正式版本推出之前,SQL 称为 SEQUEL,它是 Structured English QUery Language(结构化英语查询语言)的缩写。第一个正式版本推出后,SEQUEL 重命名为 SQL。原名称中的 English 去掉了,但发音保留了下来,以便识别。在 SQL Server 2008 版本推出时,数据库市场上许多不同的风险承担商提供了 SQL 的多种实现。在用户研究这些有时令人迷惑的数据库技术时,肯定会面对这些不同的 SQL 变体。这些不同版本的相似之处是对 ANSI 标准的实现。其中,IBM 比其他供应商更坚定地支持 ANSI 标准,几乎到了僵化的程度。然而,由于不同的 SQL 实现者自定义了不同的编程对象,并对语言进行了扩展,这就使不同的 SQL 在特定平台上具有唯一性。

Microsoft SQL Server 2008 实现了 ANSI 2003 标准。“实现”这个词很重要,因为 T-SQL 的所有实现方案都不完全兼容 ANSI 标准。不仅 T-SQL 如此,Oracle 的 P/L SQL、Sybase 的 SQLAnywhere 和开放源代码的 MySQL 也一样。每个实现方案都在既定标准上有所偏离,有各自不同的扩展和变化。ANSI 定义了三个兼容级别:入门级(Entry)、中级(Intermediate)

和完整级(Full), T-SQL 遵从 ANSI 的入门级标准。理论上说,如果在使用 SQL 时严格遵从 ANSI 所定义的功能,则在 Microsoft SQL Server 上写的代码也能在任何兼容 ANSI 的平台上工作。在编写跨平台的查询时需要加倍小心,以保证所写的代码在语法上完美地符合所有可能被涉及到的平台。实际上,这个问题的背后有个很简单的现实:需要编写可在多个平台上运行的查询的人毕竟只是少数。所以,ANSI 标准的作用只是提供指导方针,以保证查询语言是用来操作数据的,而不是其他形式的程序设计。或许,让关系数据库发展得慢一些,就足以避免我们精神崩溃。

1.1.1 T-SQL: 是编程语言还是查询语言

T-SQL 没有被设计为功能全面的编程语言。多年来,随着 ANSI 标准的扩展,SQL 加入了越来越多结构化编程语言的元素。然而,SQL 仍旧缺乏真正的编程语言具备的能力与灵活性。我的朋友 Antoine 是一位很有才华的程序员,他形容 SQL 是“吃了安眠药的 Visual Basic”。我提及此事并不因为我赞成这个说法,而是因为我觉得这种说法很有趣。我也认为这代表了许多应用软件开发者对这种通用语言的看法。

T-SQL 的唯一设计目标是实现数据的检索和操纵。虽然 T-SQL 和其他兼容 ANSI 的语言一样,可执行许多类似于编程的操作,然而执行这些操作的效率有高低。如果仅为了避免被人称为“SQL 查询员”而不是“SQL 程序员”,那么把 T-SQL 称为编程语言还是不错的。当然,T-SQL 作为编程语言还有诸多不足,这仍旧是不可否认的事实。而作为一种数据检索与集合操纵语言,SQL 是很优秀的。T-SQL 程序员想把 T-SQL 用作编程语言时,会不可避免地编写高效处理与执行的代码的最佳实践方式发生冲突。这是因为最适合 T-SQL 的是操作数据集。在编写 T-SQL 代码时这一点请牢记在心。

在 SQL Server 2005 版本中,Microsoft 在编程语言(如 C#或 VB.NET)中添加了编写数据库调用的功能,而不是在纯 SQL 中实现该功能。SQL Server 2008 也支持这个非常灵活的功能,但要小心使用。在数据访问中这是一个非常好的革新,但对数据库引擎的几乎所有调用都必须处理为基于 T-SQL。

T-SQL 颇能完成像多重递归行操作或者复杂的算术运算这样的任务。当然,也可以在记事本里编写 .NET 应用程序。我父亲曾经对我说过:“会做并不意味着必须做”。SQL 程序员在代码中时常建立定制对象,因为这是最简单快捷的编码方法,但就内存和 CPU 的消耗而言,这会降低效率。有时快速解决问题是最好的方案,但必须考虑将来的性能。

我目前使用的系统就是与上述问题相关的一个例子。由于最初的数据库非常小,开发团队和使用数据库的用户群也很小,因此数据库工作得很好。但数据库不会总是这么小,随着越来越多的用户开始使用这个系统,事务处理和代码执行的数量以几何级数增加,不久低效的代码就消耗掉了所有可用的 CPU 资源。这是编写权益代码而不是高效代码的陷阱。我父亲喜欢的另一个格言是“为什么总是没有足够的时间正确地完成工作,而总是有时间把工作做两遍?”本书试图展示编写 T-SQL 代码的最佳方法,避免用户写出让服务器可能崩溃的代码。不要试图编写很随便的代码,只是因为这是“一锤子买卖”。这种一次性的查询常常成为应用程序中业务逻辑的核心。

1.1.2 SQL Server 2008 中的新特性

SQL Server 2005 推出的时间与上一个版本 SQL Server 2000 相隔了 5 年, 这段时间对该产品的改进非常重要。许多书籍以及数以百计的网站都讨论“SQL Server 2005 中的新特性”这样的主题。而在 SQL Server 2008 推出时, 就没有这么热闹了, 对该平台的改进也不是很大。但 2008 版中仍有许多非常杰出的改进, 引入了许多 T-SQL 和应用程序开发人员一直期待的改进。这些改进都是通过 SQL Server 实现的, 所以, 本书不准备耗费大量时间来描述这个新版本所有的变化, 而只讨论可应用于本书所描述内容有关的新特性。作为介绍性章节, 本章将简要讨论两个对 SQL 程序员有重要影响的 SQL 改进: .NET Framework 与 SQL Server 的结合, 以及 Microsoft 的语言集成查询(LINQ)。

与 T-SQL 告别?

多年来, 一直都有人在说, T-SQL 及其 ANSI 实现方案 SQL 是过时的语言, 即将寿终正寝了。但每个数据库供应商, 无论是大还是小, 都投入了数百万美元, 来改进这种通用语言的专用版本。如果这是一种即将死亡的语言, 那么这些供应商为什么要这么做呢? 一个简单的事实是, 数据库是为 SQL 语言提供的基于集合的操作而构建和优化的。访问和处理数据还有更好的方式吗? 也许有, 但对于把数据存储的关系数据库中的每个重要行业而言, SQL 的统治还远未结束。

一家 Microsoft 合作公司的主管与 Microsoft 签订了合同, 开发并发布了许多 SQL Server 和 Visual Studio 产品, 他有无无线电方面的销售和市场经验, 曾经对 SQL Server 和 .NET Framework 发过这样的议论: “有了 SQL Server 与 .NET, 就可以与 T-SQL 告别了”。但在摆明了很多事实后, 他很快就被团队里的其他成员劝服了。但是, 这位主管的观点并非完全错误。他偶得的这句口号如果这样改一下就精确了: “有了 SQL Server 与 .NET, 就可以与低效的、CPU 消耗很大的 T-SQL 代码告别了。”

在最近的两个 SQL Server 版本中, 提供了两个重要的数据访问改进技术, 点燃了“SQL 即将死亡”的火苗。前面提到, 这两个版本合并了 .NET Framework, 开发了 LINQ。LINQ 是 Microsoft 最新的应用程序数据访问技术, 允许 Visual Basic 和 C# 应用程序使用面向集合的查询, 这些查询在 C# 或 VB 中开发, 而不需要用 T-SQL 编写。在 .NET Framework 中建立 SQL Server 引擎, 将允许开发人员用任意 .NET 语言创建 SQL Server 编程对象, 如存储过程、函数和聚合, 把它们编译到数据库引擎可以直接引用的 CLR 程序集中。

如果在 SQL Server 2008 中引入了 LINQ, 在 SQL Server 2005 中集成了 CLR, 那么是否就意味着 T-SQL 即将寿终正寝了呢? 不, 并不完全是。关于 T-SQL 即将寿终正寝的报道都是不成熟的、过分夸大的。具备使用托管代码创建数据库编程对象来替代 SQL 语句的能力, 并不意味着 T-SQL 有被灭绝的危险。同样, 具备用 C# 和 VB 创建基于集合的查询的能力, 也并没有敲响 T-SQL 的丧钟。SQL Server 的内部语言仍是 T-SQL。LINQ 有助于快速开发数据库应用程序, 但这种技术仍要匹配在服务器上运行的内部 T-SQL 代码的性能。这是因为 LINQ 数据访问仍必须从应用层转换到数据库层, 对于大型的企业级应用程序, LINQ 与应用程序中以前的内嵌 SQL 代码一样, 在性能方面仍达不到纯 T-SQL 的水平。

以前的状况现在仍未改变。对于需要添加、提取和处理存储在 SQL Server 上的数据的

应用程序而言, T-SQL 仍是核心语言。在数据引擎没有完全重写之前(尽管这一天很快就会到来), T-SQL 仍是 SQL Server 的核心。

1.1.3 数据库管理系统(DBMS)

DBMS 是一套用于存储和维护数据的程序。DBMS 的作用是管理数据, 维护数据的一致性和完整性。数据库管理系统有几种类型和不同的实现方案:

- **层次结构数据库管理系统(HDBMS)**

层次结构数据库已经存在了很多年, 可能是所有数据库类型中最古老的了。层次结构数据库曾用于管理以层次结构组织的数据, 目前在一些个案中它仍旧在发挥作用。层次结构数据库有许多限制, 比如只能管理单个层次结构数据、不能有效地防止错误的、重复的数据等。现在, HDBMS 的实现越来越少了, 只用于专用的、通常是非商业的应用程序中。

- **网状结构数据库管理系统(NDBMS)**

NDBMS 大都废弃了。过去, 大型机构的数据库系统是以网状或者层次结构实现的。与层次结构模型不同, 网状结构数据库没有数据不一致的问题。但是, 复杂、僵化的结构决定了对网状结构数据库及其宿主应用程序进行改进的工作将非常困难。

- **关系数据库管理系统(RDBMS)**

RDBMS 是在多个关联的表中存储数据的应用软件, 它使用 SQL 作为创建、管理、修改数据与数据结构的工具。RDBMS 将数据存储在中, 一个表代表一个实体, 例如 Customer 表和 Sale 表。在另一些表中保存了表与表之间的关系信息, 例如定义 Customer 和 Sale 表之间的关系。最早提出关系数据库概念的是一位来自 IBM 的科学家 E.F. Codd, 他于 1970 年定义了关系模型。关系数据库经过了优化, 适合于记录事务以及相应的事务性数据(Transaction Data)。大多数的商业软件使用 RDBMS 存储数据。因为 SQL 是专门为 RDBMS 设计的, 所以本章后面将详细讲解 RDBMS 的基本结构。

- **面向对象的数据库管理系统(ODBMS)**

ODBMS 是几年前才出现的系统, 它将数据作为对象存储在数据库中。ODBMS 支持面向对象理论的方方面面, 包括类与对象、类的继承等。当前, 准确定义 ODBMS 的国际标准并不存在。由于 ODBMS 应用程序保存的是对象, 而不是相关的实体, 因此系统在处理复杂的数据对象, 采用像 Microsoft 的 .NET 语言、C 与 Java 这样的面向对象编程(OOP)语言编程时, 效率非常高。ODBMS 解决方案一推出便倍受吹捧, 被认为是数据库系统的终极产品, 并预言所有其他类型的数据库系统都将成为历史。然而, 它一直没能如预测所愿被广泛接受。虽然它在数据库市场上很快占有一席之地, 但基本上仅限于计算机辅助设计(CAD)与电信行业。

- **对象-关系数据库管理系统(ORDBMS)**

ORDBMS 是从 RDBMS 发展而来的。关系数据库系统的开发者意识到, 存储对象的能力越来越重要, 于是他们在关系模型中加入了能够存储类与对象的机制。ORDBMS 的实现方案基本上占领了原本属于 ODBMS 的目标市场, 个中有太多原因, 这里不详细阐述。而将具备了 XML 数据类型、结合了 .NET Framework、引入了 filestream 数据类型的 Microsoft SQL Server 2008 贴上了 ORDBMS 的标签, 则有些争议。filestream 数据类型详见本章后面

的内容和附录 E。

1.2 作为关系数据库管理系统的 SQL Server

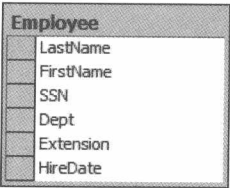
本节介绍关系数据库的概念，以及 Microsoft 实现它们的方式。本节的内容围绕数据库对象的创建展开，第 13 章将详细介绍这方面的内容。为了方便讨论，本节没有描述具体的机制，而把注意力集中在最终的结果上。

如前所述，关系数据库将所有数据都存储在表里。在理想情况下，每一个表都代表单一的实体或者对象。显然不应将风马牛不相及的两种数据(比如说，狗和汽车)保存在同一张表中。并不是说不能这样做，而是如果这样做的话，效率和可维护性就大打折扣了。

1.2.1 表

表可拆分成行与列。每一行必须能够独立存在，不依赖表中的其他行。行必须表示该表所代表的实体的一个完整的实例。行中的每一列包含特定的属性，它们有助于定义实例。这听起来可能有点复杂，但实际上很简单。为了便于说明，下面以现实世界中的实体——雇员为例。为了保存雇员的数据，需要创建一个表，其中的各种属性记录了雇员的数据。为了简单起见，将这个表命名为 Employee。

在创建雇员表时，还需要决定存储雇员的哪些属性。对于本例来说，假定需要存储雇员的姓氏、名字、社会保险号码(Social Security Number, SSN)、部门、电话分机号和雇佣日期。所得到的表如图 1-1 所示。



Employee					
LastName	FirstName	SSN	Dept	Extension	HireDate

图 1-1

表中的数据见图 1-2。

LastName	FirstName	SSN	Dept	Extension	HireDate
Flintstone	Fred	123456789	Operations	9876	11/12/2000
Slate	George	987654321	Management	3456	4/14/1999

图 1-2

1. 主键

为了有效地管理表中的数据，需要唯一地标识表中的每一行。如果没有一个特性，将不同的行区别开来，检索、更新或者删除一行数据就相当困难了。在很多情况下，这个