

THE EXPERT'S VOICE® IN SQL SERVER PROGRAMMING

Expert SQL Server 2005 Development

*Advanced SQL Server techniques
for database professionals*

"The authors of this book are well-known in the SQL Server community for their in-depth architectural analysis and attention to technical detail. I recommend this book to anyone who wants to explore SQL Server solutions to some common and some not-so-common data storage and access problems."

—Bob Beauchemin, Director of Developer Skills, SQLskills



Adam Machanic
with Hugo Kornelis and Lara Rubbelke

Foreword by AP Ward Pond

Technology Architect, Microsoft SQL Server Center of Excellence

apress®

Expert SQL Server 2005 Development



Adam Machanic
with Hugo Kornelis and Lara Rubbelke

Expert SQL Server 2005 Development

Copyright © 2007 by Adam Machanic, Hugo Kornelis, Lara Rubbelke

All rights reserved. No part of this work may be reproduced or transmitted in any form or by any means, electronic or mechanical, including photocopying, recording, or by any information storage or retrieval system, without the prior written permission of the copyright owner and the publisher.

ISBN-13 (pbk): 978-1-59059-729-3

ISBN-10 (pbk): 1-59059-729-X

Printed and bound in the United States of America 9 8 7 6 5 4 3 2 1

Trademarked names may appear in this book. Rather than use a trademark symbol with every occurrence of a trademarked name, we use the names only in an editorial fashion and to the benefit of the trademark owner, with no intention of infringement of the trademark.

Lead Editor: James Huddleston

Technical Reviewer: Greg Low

Editorial Board: Steve Anglin, Ewan Buckingham, Gary Cornell, Jason Gilmore, Jonathan Gennick,

Jonathan Hassell, James Huddleston, Chris Mills, Matthew Moodie, Jeffrey Pepper, Dominic Shakeshaft, Matt Wade

Senior Project Manager: Tracy Brown Collins

Copy Edit Manager: Nicole Flores

Copy Editor: Ami Knox

Assistant Production Director: Kari Brooks-Copony

Senior Production Editor: Laura Cheu

Compositor and Artist: Kinetic Publishing Services, LLC

Proofreader: Elizabeth Berry

Indexer: Beth Palmer

Cover Designer: Kurt Krames

Manufacturing Director: Tom Debolski

Distributed to the book trade worldwide by Springer-Verlag New York, Inc., 233 Spring Street, 6th Floor, New York, NY 10013. Phone 1-800-SPRINGER, fax 201-348-4505, e-mail orders-ny@springer-sbm.com, or visit <http://www.springeronline.com>.

For information on translations, please contact Apress directly at 2560 Ninth Street, Suite 219, Berkeley, CA 94710. Phone 510-549-5930, fax 510-549-5939, e-mail info@apress.com, or visit <http://www.apress.com>.

The information in this book is distributed on an “as is” basis, without warranty. Although every precaution has been taken in the preparation of this work, neither the author(s) nor Apress shall have any liability to any person or entity with respect to any loss or damage caused or alleged to be caused directly or indirectly by the information contained in this work.

The source code for this book is available to readers at <http://www.apress.com> in the Source Code/Download section. A companion web site for this book, containing updates and additional material, can be accessed at <http://www.expertsqserver2005.com>.

To Kate: Thanks for letting me disappear into the world of my laptop and my thoughts for so many hours over the last several months. Without your support I never would have been able to finish this book. And now you have me back . . . until I write the next one.

—Adam Machanic

Contents at a Glance

Foreword	xiii
About the Authors	xv
About the Technical Reviewer	xvii
Acknowledgments	xix
Introduction	xxi
■ CHAPTER 1 Software Development Methodologies for the Database World	1
■ CHAPTER 2 Testing Database Routines	23
■ CHAPTER 3 Errors and Exceptions	47
■ CHAPTER 4 Privilege and Authorization	73
■ CHAPTER 5 Encryption	91
■ CHAPTER 6 SQLCLR: Architecture and Design Considerations	133
■ CHAPTER 7 Dynamic T-SQL	169
■ CHAPTER 8 Designing Systems for Application Concurrency	209
■ CHAPTER 9 Working with Spatial Data	251
■ CHAPTER 10 Working with Temporal Data	315
■ CHAPTER 11 Trees, Hierarchies, and Graphs	375
■ INDEX	439

Contents

Foreword	xiii
About the Authors	xv
About the Technical Reviewer	xvii
Acknowledgments	xix
Introduction	xxi

CHAPTER 1	Software Development Methodologies for the Database World	1
	Architecture Revisited	2
	Coupling, Cohesion, and Encapsulation	2
	Interfaces	5
	The Central Problem: Integrating Databases and Object-Oriented Systems	8
	Where Should the Logic Go?	8
	The Object-Relational Impedance Mismatch	12
	ORM: A Solution That Creates Many Problems	17
	Introducing the Database-as-API Mindset	18
	The Great Balancing Act	19
	Testability	19
	Maintainability	19
	Security	20
	Performance	21
	Creeping Featurism	21
	Summary	22

CHAPTER 2	Testing Database Routines	23
	Introduction to Black Box and White Box Testing	23
	Unit and Functional Testing	24
	Unit Testing Frameworks	26
	The Importance of Regression Testing	29

Guidelines for Implementing Database Testing Processes and Procedures	30
Why Is Testing Important?	30
What Kind of Testing Is Important?	31
How Many Tests Are Needed?	31
Will Management Buy In?	32
Performance Testing and Profiling Database Systems	33
Capturing Baseline Metrics	33
Profiling Using Traces and SQL Server Profiler	34
Evaluating Performance Counters.	36
Big-Picture Analysis.	37
Granular Analysis	38
Fixing Problems: Is Focusing on the Obvious Issues Enough?	40
Introducing the SQLQueryStress Performance Testing Tool	40
Summary	45
 CHAPTER 3 Errors and Exceptions	 47
Exceptions vs. Errors.	47
How Exceptions Work in SQL Server	48
Statement-Level Exceptions	48
Batch-Level Exceptions	49
Parsing and Scope-Resolution Exceptions	50
Connection and Server-Level Exceptions	52
The XACT_ABORT Setting	52
Dissecting an Error Message	53
SQL Server's RAISERROR Function.	56
Monitoring Exception Events with Traces	60
Exception Handling	60
Why Handle Exceptions in T-SQL?	60
Exception "Handling" Using @@ERROR	61
SQL Server's TRY/CATCH Syntax.	62
Transactions and Exceptions	68
The Myths of Transaction Abortion	68
XACT_ABORT: Turning Myth into (Semi-)Reality	69
TRY/CATCH and Doomed Transactions.	71
Summary	72

CHAPTER 4	Privilege and Authorization	73
	The Principle of Least Privilege	74
	Creating Proxies in SQL Server	74
	Data Security in Layers: The Onion Model	75
	Data Organization Using Schemas	76
	Basic Impersonation Using EXECUTE AS	79
	Ownership Chaining	81
	Privilege Escalation Without Ownership Chains	83
	Stored Procedures and EXECUTE AS	83
	Stored Procedure Signing Using Certificates	85
	Summary	89
CHAPTER 5	Encryption	91
	What to Protect	92
	Encryption Terminology: What You Need to Know	93
	SQL Server 2005 Encryption Key Hierarchy	94
	Service Master Key	95
	Database Master Key	95
	SQL Server 2005 Data Protection	97
	HashBytes()	97
	Asymmetric Key and Certificate Encryption	98
	Symmetric Key Encryption	101
	EncryptByPassphrase	108
	Securing Data from the DBA	109
	Architecting for Performance	111
	Setting Up the Solution and Defining the Problem	112
	Searching Encrypted Data	116
	Summary	131
CHAPTER 6	SQLCLR: Architecture and Design Considerations	133
	Bridging the SQL/CLR Gap: the SqlTypes Library	134
	Wrapping Code to Promote Cross-Tier Reuse	135
	A Simple Example: E-Mail Address Format Validation	135
	SQLCLR Security and Reliability Features	137
	The Quest for Code Safety	140
	Selective Privilege Escalation via Assembly References	141
	Granting Cross-Assembly Privileges	148
	Enhancing Service Broker Scale-Out with SQLCLR	151

Extending User-Defined Aggregates	162
Summary	167
■ CHAPTER 7 Dynamic T-SQL	169
Dynamic T-SQL vs. Ad Hoc T-SQL	169
The Stored Procedure vs. Ad Hoc SQL Debate	170
Why Go Dynamic?	171
Compilation and Parameterization	172
Auto-Parameterization	174
Application-Level Parameterization	175
Performance Implications of Parameterization and Caching	177
Supporting Optional Parameters	180
Optional Parameters via Static T-SQL	180
Going Dynamic: Using EXECUTE	186
SQL Injection	192
sp_executesql: A Better EXECUTE	195
Dynamic SQL Security Considerations	204
Permissions to Referenced Objects	204
Interface Rules	205
Summary	207
■ CHAPTER 8 Designing Systems for Application Concurrency	209
The Business Side: What Should Happen When	
Processes Collide?	210
A Brief Overview of SQL Server Isolation Levels	211
Concurrency Control and SQL Server's Native	
Isolation Levels	216
Preparing for the Worst: Pessimistic Concurrency	217
Enforcing Pessimistic Locks at Write Time	222
Application Locks: Generalizing Pessimistic Concurrency	224
Hoping for the Best: Optimistic Concurrency	234
Embracing Conflict: Multivalue Concurrency	239
Extending Scalability Through Queuing	243
Summary	249

CHAPTER 9	Working with Spatial Data	251
	Representing Geospatial Data by Latitude and Longitude	251
	Setting Up Sample Data	253
	Calculating the Distance Between Two Points	254
	Moving from Point to Point	259
	Searching the Neighborhood	263
	The Bounding Box	269
	Finding the Nearest Neighbor	281
	The Dynamic Bounding Box	284
	Conclusion	293
	Representing Geospatial Data by Using the Hierarchical	
	Triangular Mesh	294
	A Simplified Description of HTM	294
	Implementing the HtmID	298
	Functions in the Spatial Database	300
	Conclusion	311
	Other Types of Spatial Data	312
	Three-Dimensional Data	312
	Astronomical Data	312
	Virtual Space	312
	Representing Regions As Polygons	313
	Summary	313
CHAPTER 10	Working with Temporal Data	315
	Representing More Than Just Time	315
	SQL Server's Date/Time Data Types	316
	Input Date Formats	316
	Output Date Formatting	318
	Efficiently Querying Date/Time Columns	320
	Date/Time Calculations	323
	Defining Periods Using Calendar Tables	329
	Designing and Querying Temporal Data Stores	340
	Dealing with Time Zones	341
	Working with Intervals	348
	Modeling Durations	368
	Managing Bitemporal Data	370
	Summary	373

CHAPTER 11	Trees, Hierarchies, and Graphs	375
	Terminology: Everything Is a Graph	375
	The Basics: Adjacency Lists and Graphs	377
	Constraining the Edges	378
	Basic Graph Queries: Who Am I Connected To?	380
	Traversing the Graph	381
	Adjacency List Hierarchies	391
	Querying Adjacency List Hierarchies: The Basics	392
	Finding Direct Descendants	393
	Traversing down the Hierarchy	395
	Traversing up the Hierarchy	404
	Inserting New Nodes and Relocating Subtrees	405
	Deleting Existing Nodes	406
	Constraining the Hierarchy	407
	Persisting Materialized Paths	409
	Finding Subordinates	411
	Navigating up the Hierarchy	412
	Optimizing the Materialized Path Solution	413
	Inserting Nodes	418
	Relocating Subtrees	419
	Deleting Nodes	422
	Constraining the Hierarchy	422
	Nested Sets Model	422
	Finding Subordinates	426
	Navigating up the Hierarchy	428
	Inserting Nodes	428
	Relocating Subtrees	430
	Deleting Nodes	435
	Constraining the Hierarchy	436
	Summary	437
INDEX		439

Foreword

Databases are software. I've based the second half of a software development career that began in 1978 on this simple idea.

If you've found this book, chances are you're willing to at least entertain the possibility that databases and their attendant programmability are worthy of the same rigor and process as the rest of an application. Good for you! It's a great pleasure for me to join you on this journey, however briefly, via this foreword.

There is a good possibility that you've grown as skeptical as I have of the conventional wisdom that treats the "back end" as an afterthought in the design and budgeting process. You're now seeking actionable insights into building or improving a SQL Server 2005 design and development process.

The book you're holding is chock-full of such insights. And before turning you over to Adam, Hugo, and Lara, I'd like to offer one of my own.

I suggest that we stop calling the database the "back end." There is a dismissive and vaguely derogatory tone to the phrase. It sounds like something we don't want to pay much attention to, doesn't it? The "front end," on the other hand, sounds like the place with all the fun and glory. After all, it's what everybody can *see*. The back end sounds like something you can safely ignore. So when resources must be trimmed, it might be easier and safer to start where people can't see ... right?

Wrong. Such an approach ignores the fact that databases are software—important, intricate software. How would our outlook change if we instead referred to this component as the "foundational layer"? This term certainly sounds much weightier. For instance, when I consider the foundational layer of my family's house, I fervently hope that the people who designed and built it knew what they were doing, especially when it comes to the runoff from the hill in our backyard. If they didn't, all of the more obvious, fancy stuff that relies on the proper architecture and construction of our home's foundational layer—everything from the roof to the cable modem to my guitars—is at risk. Similarly, if the *foundational layer* of our application isn't conceived and crafted to meet the unique, carefully considered needs of our customers, the beauty of its user interface won't matter. Even the most nimble user interface known to mankind will fail to satisfy its users if its underlying foundational layer fails to meet any of the logical or performance requirements.

I'll say it again: Databases are software. Stored procedures, user-defined functions, and triggers are obviously software. But schema is software, too. Primary and foreign keys are software. So are indexes and statistics. The *entire* database is software. If you've read this far, chances are that you know these things to your core. You're seeking a framework, a mindset with which to approach SQL Server 2005 development in an orderly fashion. When you've completed this incredibly readable book, you'll have just such a context.

My work at Microsoft since 1999 has led me to become an advocate for the application of rigorous quality standards to all phases of database design and construction. I've met several

kindred spirits since I went public with this phase of my work in 2005, including Adam and Hugo. If you apply the advice that the authors offer in the pages that follow, you'll produce more scalable, maintainable databases that perform better. This will then lead to applications that perform better and are more maintainable, which will make your customers happier. This state of affairs, in turn, will be good for business.

And as a bonus, you'll be both a practitioner and a proponent of an expert-level tenet in the software and IT industries: Databases are software!

Ward Pond

Technology Architect, Microsoft SQL Server Center of Excellence

<http://blogs.technet.com/wardpond>

sqlwriter@comcast.net

About the Authors



ADAM MACHANIC is an independent database software consultant, writer, and speaker based in Boston, Massachusetts. He has implemented SQL Server solutions for a variety of high-availability OLTP and large-scale data warehouse applications, and also specializes in .NET data access layer performance optimization. Adam has written for *SQL Server Professional* and *TechNet* magazines, serves as the SQL Server 2005 Expert for SearchSQLServer.com, and has contributed to several books on SQL Server, including *Pro SQL Server 2005* (Apress, 2005). He regularly speaks at user groups, community events, and conferences on a variety of SQL Server and .NET-related topics. He is a Microsoft Most Valuable Professional (MVP) for SQL Server and a Microsoft Certified IT Professional (MCITP).

When not sitting at the keyboard pounding out code or code-related prose, Adam tries to spend a bit of time with his wife, Kate, and daughter, Aura, both of whom seem to believe that there is more to life than SQL.

Adam blogs at <http://www.sqlblog.com>, and can be contacted directly at amachanic@datamanipulation.net.

HUGO KORNELIS has a strong interest in information analysis and process analysis. He is convinced that many errors in the process of producing software can be avoided by using better procedures during the analysis phase, and deploying code generators to avoid errors in the process of translating the analysis results to databases and programs. Hugo is cofounder of the Dutch software company perFact BV, where he is responsible for improving analysis methods and writing a code generator to generate complete working SQL Server code from the analysis results.

When not working, Hugo enjoys spending time with his wife, two children, and four cats. He also enjoys helping out people in SQL Server-related newsgroups, speaking at conferences, or playing the occasional game.

In recognition of his efforts in the SQL Server community, Hugo was given the Most Valuable Professional (MVP) award by Microsoft in January 2006 and January 2007. He is also a Microsoft Certified Professional.

Hugo contributed Chapter 9, “Working with Spatial Data.”

LARA RUBBELKE is a service line leader with DigiNeer in Minneapolis, Minnesota, where she consults on architecting, implementing, and improving SQL Server solutions. Her expertise involves both OLTP and OLAP systems, ETL, and the Business Intelligence lifecycle. She is an active leader of the local PASS chapter and brings her passion for SQL Server to the community through technical presentations at local, regional, and national conferences and user groups. Lara’s two beautiful and active boys, Jack and Tom, and incredibly understanding husband, Bill, are a constant source of joy and inspiration.

Lara contributed Chapter 5, “Encryption.”

About the Technical Reviewer



GREG LOW is an internationally recognized consultant, developer, author, and trainer. He has been working in development since 1978, holds a PhD in computer science and MC*.* from Microsoft. Greg is the lead SQL Server consultant with Readify, a SQL Server MVP, and one of only three Microsoft regional directors for Australia. He is a regular speaker at conferences such as TechEd and PASS. Greg also hosts the SQL Down Under podcast (<http://www.sqldownunder.com>), organizes the SQL Down Under Code Camp, and co-organizes CodeCampOz.

Acknowledgments

Imagine, if you will, the romanticized popular notion of an author at work. Gaunt, pale, bent over the typewriter late at night (perhaps working by candlelight), feverishly hitting the keys, taking breaks only to rip out one sheet and replace it with a blank one, or maybe to take a sip of a very strong drink. All of this, done alone. Writing, after all, is a solo sport, is it not?

While I may have spent more than my fair share of time bent over the keyboard late at night, illuminated only by the glow of the monitor, and while I did require the assistance of a glass of Scotch from time to time, I would like to go ahead and banish any notion that the book you hold in your hands was the accomplishment of just one person. On the contrary, numerous people were involved, and I hope that I have kept good enough notes over the last year of writing to thank them all. So without further ado, here are the people behind this book.

Thank you first to Tony Davis, who helped me craft the initial proposal for the book. Even after leaving Apress, Tony continued to give me valuable input into the writing process, not to mention publishing an excerpt or two on <http://www.Simple-Talk.com>. Tony has been a great friend and someone I can always count on to give me an honest evaluation of any situation I might encounter.

Aaron Bertrand, Andrew Clarke, Hilary Cotter, Zach Nichter, Andy Novick, Karen Watterson, and Kris Zaragoza were kind enough to provide me with comments on the initial outline and help direct what the book would eventually become. Special thanks go to Kris, who told me that the overall organization I presented to him made no sense, then went on to suggest numerous changes, all of which I ended up using.

James Huddleston carried me through most of the writing process as the book's editor. Sadly, he passed away just before the book was finished. Thank you, James, for your patience as I missed deadline after deadline, and for your help in driving up the quality of this book. I am truly saddened that you will not be able to see the final product that you helped forge.

Tracy Brown Collins, the book's project manager, worked hard to keep the book on track, and I felt like I let her down every time I delivered my material late. Thanks, Tracy, for putting up with schedule change after schedule change, multiple chapter and personnel reorganizations, and all of the other hectic interplay that occurred during the writing of this book.

Throughout the writing process, I reached out to various people to answer my questions and help me get over the various stumbling blocks I faced. I'd like to thank the following people whom I pestered again and again, and who patiently took the time out of their busy schedules to help me: Bob Beauchemin, Itzik Ben-Gan, Louis Davidson, Peter DeBetta, Kalen Delaney, Steven Hemingray, Tibor Karaszi, Steve Kass, Andy Kelly, Tony Rogerson, Linchi Shea, Erland Sommarskog, Roji Thomas, and Roger Wolter. Without your assistance, I would have been hopelessly stuck at several points along the way.

Dr. Greg Low, the book's technical reviewer, should be granted an honorary PhD in SQL Server. Greg's keen observations and sharp insight into what I needed to add to the content were very much appreciated. Thank you, Greg, for putting in the time to help out with this project!

To my coauthors, Hugo Kornelis and Lara Rubbelke, thank you for jumping into book writing and producing some truly awesome material! I owe you both many rounds of drinks for helping me to bear some of the weight of getting this book out on time and at a high level of quality.

An indirect thanks goes out to Ken Henderson and Joe Celko, whose books inspired me to get started down the writing path to begin with. When I first picked up Ken's *Guru's Guide* books and Joe's *SQL for Smarties*, I hoped that some day I'd be cool enough to pull off a writing project. And while I can't claim to have achieved the same level of greatness those two managed, I hope that this book inspires a new writer or two, just as theirs did me. Thanks, guys!

Last, but certainly not least, I'd like to thank my wife, Kate, and my daughter, Aura. Thank you for understanding as I spent night after night and weekend after weekend holed up in the office researching and writing. Projects like these are hard on interpersonal relationships, especially when you have to live with someone who spends countless hours sitting in front of a computer with headphones on. I really appreciate your help and support throughout the process. I couldn't have done it without you!

Aura, some day I will try to teach you the art and science of computer programming, and you'll probably hate me for it. But if you're anything like me, you'll find some bizarre pleasure in making the machine do your bidding. That's a feeling I never seem to tire of, and I look forward to sharing it with you.

Adam Machanic

I'd like to thank my wife, José, and my kids, Judith and Timon, for stimulating me to accept the offer and take the deep dive into authoring, and for putting up with me sitting behind a laptop for even longer than usual.

Hugo Kornelis

I would like to acknowledge Stan Sajous for helping develop the material for the encryption chapter.

Lara Rubbelke

Introduction

Working with SQL Server on project after project, I find myself solving the same types of problems again and again. The solutions differ slightly from case to case, but they often share something in common—code patterns, logical processes, or general techniques. Every time I work on a customer’s software, I feel like I’m building on what I’ve done before, creating a greater set of tools that I can apply to the next project and the next after that. Whenever I start feeling like I’ve gained mastery in some area, I’ll suddenly learn a new trick and realize that I really don’t know anything at all—and that’s part of the fun of working with such a large, flexible product as SQL Server.

This book, at its core, is all about building your own set of tools from which you can draw inspiration as you work with SQL Server. I try to explain not only the *hows* of each concept described herein, but also the *whys*. And in many examples throughout the book, I attempt to delve into the process I took for finding what I feel is the optimal solution. My goal is to share with you how I think through problems. Whether or not you find my approach to be directly usable, my hope is that you can harness it as a means by which to tune your own development methodology.

This book is arranged into three logical sections. The first four chapters deal with software development methodologies as they apply to SQL Server. The next three chapters get into advanced features specific to SQL Server. And the final four chapters are more architecturally focused, delving into specific design and implementation issues around some of the more difficult topics I’ve encountered in past projects.

Chapters 1 and 2 aim to provide a framework for software development in SQL Server. By now, SQL Server has become a lot more than just a DBMS, yet I feel that much of the time it’s not given the respect it deserves as a foundation for application building. Rather, it’s often treated as a “dumb” object store, which is a shame, considering how much it can do for the applications that use it. In these chapters, I discuss software architecture and development methodologies, and how to treat your database software just as you’d treat any other software—including testing it.

Software development is all about translating business problems into technical solutions, but along the way you can run into a lot of obstacles. Bugs in your software or other components and intruders who are interested in destroying or stealing your data are two of the main hurdles that come to mind. So Chapters 3 and 4 deal with exception handling and security, respectively. By properly anticipating error conditions and guarding against security threats, you’ll be able to sleep easier at night, knowing that your software won’t break quite as easily under pressure.

Encryption, SQLCLR, and proper use of dynamic SQL are covered in Chapters 5, 6, and 7. These chapters are not intended to be complete guides to each of these features—especially true of the SQLCLR chapter—but are rather intended as reviews of some of the most important things you’ll want to consider as you use these features to solve your own business problems.

Chapters 8 through 11 deal with application concurrency, spatial data, temporal data, and graphs. These are the biggest and most complex chapters of the book, but also my favorite.

Data architecture is an area where a bit of creativity often pays off—a good place to sink your teeth into new problems. These chapters show how to solve common problems using a variety of patterns, each of which should be easy to modify and adapt to situations you might face in your day-to-day work as a database developer.

Finally, I'd like to remind readers that database development, while a serious pursuit and vitally important to business, should be *fun*! Solving difficult problems cleverly and efficiently is an incredibly satisfying pursuit. I hope that this book helps readers get as excited about database development as I am.