Pro SQL Server 2000 Database Design: Building Quality OLTP Databases

Louis Davidson

Apress™

Pro SQL Server 2000 Database Design: Building Quality OLTP Databases

ISBN (pbk): 1-59059-302-2

Printed and bound in the United States of America 12345678910

Technical Reviewer: Craig Weldon

Editorial Board: Dan Appleman, Craig Berry, Gary Cornell, Tony Davis, Steven Rycroft, Julian Skinner, Martin Streicher, Jim Sumser, Karen Watterson, Gavin Wray, John Zukowski

Assistant Publisher: Grace Wong

Project Manager: Kylie Johnston

Copy Editor: Mark Nigara

Production Manager: Kari Brooks

Production Editor: Kelly Winquist

Proofreader: Patrick Vincent

Compositor: Katy Freer

Artist: April Milne

Cover Designer: Kurt Krames

Manufacturing Manager: Tom Debolski

Distributed to the book trade in the United States by Springer-Verlag New York, Inc., 175 Fifth Avenue, New York, NY 10010 and outside the United States by Springer-Verlag GmbH & Co. KG, Tiergartenstr. 17, 69112 Heidelberg, Germany.

In the United States: phone 1-800-SPRINGER, email orders@springer-ny.com, or visit http://www.springer-ny.com. Outside the United States: fax +49 6221 345229, email orders@springer.de, or visit http://www.springer.de.

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, email info@apress.com, or visit http://www.apress.com.

The source code for this book is available to readers at http://www.apress.com in the Downloads section. You will need to answer questions pertaining to this book in order to successfully download the code.

Contents at a Glance

Introduction

If you're standing in your favorite bookseller, flipping thorough this book because it's in the technical book section, I know you're probably thinking, "Hey, where is all of the code and settings and such?" Well, this isn't exactly that kind of book. (Not that there is anything wrong with that kind of book; I alone have a gaggle of them around my desk.) This book intends to balance the thin line between the very implementation-oriented SQL Server books–which are concerned solely with DBCC settings, index options, and all of the knobs and handles on SQL Server–and the truly academic tomes that go deep into the theory of databases, but provide little or no practical information.

Database design can be thought of as an interesting blend of art and science. The science of database design is well established, with a number of mature methods for designing database structures, the most common of which is normalization. In fact these methods are almost ancient in computing terms, and it isn't terribly difficult to follow these rules when designing your database structures. As you'll see, it's also relatively straightforward to translate such a design into SQL Server tables.

This book presents straightforward techniques that will help you to design better databases in a way that is clear enough for novices, but at the same time helpful to even the most seasoned professional. One thing should be made clear before you start reading. Working with the design and architecture of a database requires a very different approach than performing database setup and administration. For example, in the role of data architect, I seldom create users, perform backups, or set up replication or clustering. This is the role of the DBA.

Though it isn't uncommon to wear both of these hats (and others such a programmer, tester, and so on), I think you'll find that your designs will generally be far better thought out if you can divorce your mind from the more implementation-bound roles.

In this book you'll look in depth at how to design, architect, and implement the SQL Server database tables and their accompanying access methods; you'll also examine the physical hardware, system software, and other aspects of a database system. You'll start from the point where someone mentions to you that they want to store some data, and begin generating tables and implementing access to these tables.

The real mission isn't so straightforward though. I've reviewed lots of people's code over the years, and no matter how well the object models, supporting code, or even documentation was put together, the database generally ranged from bad to horrible (there were good examples too!). My desire is to provide the information that database architects need to build databases properly. For anyone who has had the pleasure of reading college database textbooks, you know that they can be a bit on the dry side–as dry as the Mojave Desert in a drought. This is actually too bad, because much of the information contained between the covers is useful and relevant.

So my restated mission is to end the communication breakdown between the egghead geniuses and you, the working programmer, in order to provide the necessary knowledge in a way that's applicable to the real world. If you've ever done any programming, you'll undoubtedly disagree with some of the ideas presented in this book. I fully accept that this book is hardly the Gospel of St. Louis of Databases. My ideas and opinions have grown from 12 years of working with, and learning about, databases, and as such I've supplemented them with knowledge from many disparate persons, books, college classes, and seminars. The design methodology presented in this book is a conglomeration of these ideas, with as much credit given to those other folks as I can remember. I hope it proves a useful learning tool, and that by reading other people's works and trying out your own ideas, you'll develop a methodology that will make you a successful database designer.

What's Covered in This Book

This is a book of two halves. The first covers the logical design of databases, and the second looks at the physical design and implementation. The following is a summary of the areas covered chapter by chapter.

- **Chapter 1 Introduction to Database Concepts**: As its name implies, a brief introduction to the different database methodologies that are commonly used to implement a fully featured database system, such as OLTP databases, Data Warehouses, Operation Data Stores, and Data Marts.
- **Chapter 2 Gathering Information for a Database Project**: In this chapter we give an overview of the process of determining the requirements that the users will have of the database system, by looking at some of the more obscure places where important data hides.
- **Chapter 3 Fundamental Database Concepts**: A basic understanding of the concepts of relational theory is fundamental to the process of database design and is considered here. This will provide a basis for the development of our design.
- **Chapter 4 Entities, Attributes, Relationships, and Business Rules**: In this chapter we will begin the process of turning the information gathered in Chapter 2 into a logical design of our relational database, in particular by devising the entities we will require.
- **Chapter 5 Data Modeling**: Once we have discovered objects, we need to have a way to display and share the information with programmers and users. The data model is the most effective tool for depicting database design information.

- **Chapter 6 Normalization Techniques**: Normalization is the process of taking the information we gathered in Chapter 2, and developed in Chapters 4 and Chapter 5, and turning it into a well-structured draft of the data model. In this chapter we consider the normalization rules we must follow in designing a well-structured model for our system.
- **Chapter 7 Advanced Normalization Topics**: This chapter builds on the previous one by extending the basic normalization techniques beyond those familiar to most programmers. In this way we are able to fine-tune our logical design so as to avoid, as far as is possible, any data anomalies.
- **Chapter 8 Ending The Logical Design Phase**: Once we have designed the "perfect" database, we need to return to the original specifications to ensure that the data we expect to store will serve the data needs of the users. By this point many programmers are ready to code away, but it is important to finish off the logical design phase, by double-checking our model and its documentation to try and minimize the level of changes required when we come to physically implementing it.
- **Chapter 9 Planning the Physical Architecture**: In this chapter you'll look at making decisions about how you'll implement the architecture. Here you'll begin to take into consideration the number of users, system size, and how the data will be used.
- **Chapter 10 Building the Basic Table Structures**: In this chapter you'll go through the mechanics of choosing datatypes, building tables, and creating indices.
- **Chapter 11 Declarative Data Protection**: There are many different issues that govern the data integrity of your system. In this chapter you'll look at how to build data protection into your databases using the base techniques using constraints.
- **Chapter 12 Programmatic Data Protection**: In this chapter you go beyond the declarative protections described in Chapter 11 to include methods that are extremely flexible, using triggers and stored procedures.
- **Chapter 13 Advanced Data Access and Modification**: In this chapter you'll look at some of the different methods of accessing the data in the databases you've created, in particular through the use of views and stored procedures.
- **Chapter 14 Determining Hardware Requirements**: One of the most difficult activities for many data architects is to translate the system requirements into actual hardware requirements. What disk format do I need? How much disk space do I need, and how will these requirements change in the future? How much RAM? How many processors? Too many questions. These are generally questions for the database administrator, but for many architects there may not be an administrator around to help make proper hardware decisions.
- **Chapter 15 Completing the Project**: In this chapter you'll look at the endgame. Once you have your data, structures, and queries built, the project is over, isn't it? To wrap up your project you'll consider the fine-tuning you'll need to carry out, including performance issues concerning reporting needs, especially in high usage systems, and the system testing and deployment you must undertake. You'll also look at drawing up disaster plans, in order to protect your data from all those mishaps that Murphy (in his famous law) predicted would occur.

In addition to the material directly in this text of this book, there are several appendices that will be located on the Apress website that supplement the text. These appendices cover material that was not exactly a great fit for a database design book, but useful to explain a concept or a feature of SQL Server or the RDBMS that will extend your experience with the book.

These appendices are:

- **Appendix A: Codd's 12 Rules for an RDBMS** A listing of Codd's rules, and a brief discussion of how SQL Server meets these standards that have stood as a measuring stick of relational databases since 1985.
- **Appendix B: Indexes** Explains the basic structure of indexes to give you a feel for the best uses of indexes.
- **Appendix C: User Defined Functions** An overview of how to code user defined functions.
- **Appendix D: Data Type Reference** An introduction to the many different intrinsic datatypes provided by SQL Server. This appendix gives an overview of how they are implemented and how best to use the different datatypes.
- **Appendix E: Triggers** An overview of how to code triggers to implement complex business rules.
- **Appendix F: Transactions** An overview of how transactions are used in SQL Server code to promote proper concurrency and data integrity.
- **Appendix G: Cursors** An overview of the features and settings used in Cursors.

I have moved most of the sections out of the book and onto the website to focus completely on database design, and very little on the mechanics of SQL Server coding. However, these references give you what you need to know to understand and code these objects, if you do not currently possess these skills.

Who Should Read This Book?

Of course it is hard to not say "everyone on earth," but each chapter of this book isn't necessarily relevant for everyone and parts of it will not appeal to every programmer. I wish it did, because there is stuff in each chapter that will enrich every programmer's ability to design and implement databases. However, this is a breakdown of what will be valuable to whom.

Database Architect

If you're already a database architect who is responsible for gathering requirements and designing databases with involvement or responsibility for implementation, then read the entire book. You can skip the third chapter on a first reading.

Database Programmer

If you're only involved in the implementation of database code, you'll be interested in a good part of this book, which will help you understand the reasons why the "insane" data architect wants to implement a database with 50 tables when it seems like it needs only 3.

A first reading might include Chapter 5 on data modeling, and Chapters 6 and 7 on normalization, followed by the entire Part Two of the book. This section describes all of the techniques for implementing database systems.

What You Need to Use This Book

For the first half of the book, you'll be learning about logical data modeling. There are no software requirements for working through this part of the book.

In the latter half that deals with physical design, the only requirement (other than your thinking cap) for working through the examples is an installed copy of SQL Server 2000 and the Query Analyzer tool that comes with it. This can be any edition of SQL Server (Personal Edition upwards), as long as you can connect with Query Analyzer. You'll need a database that you can access with a user who is a member of the `db_owner` role, because you'll be creating all objects as the database owner.

If you don't have a copy of SQL Server, an Evaluation Edition is available on Microsoft's SQL Server website at http://www.microsoft.com/sql.

Conventions Used

You're going to encounter different styles as you're reading through this book. This has been done to help you easily identify different types of information and make sure that you don't miss any key points. These styles are as follows:

Important information, key points, and additional explanations are displayed like this to make them stand out. Be sure to pay attention to these when you find them.

General notes, background information, and brief asides look like this.

- If you see something like, `BackupDB`, you'll know that it's a file name, object name, or function name.
- The first time you encounter an **important word**, it's displayed in bold text.
- Words that appear on the screen, such as menu options, are in a similar font to the one used on screen, for example, the File menu.