

[Main Page](#)

[Table of content](#)

[Copyright](#)

[Dedication](#)

[Foreword](#)

[Preface](#)

[Objectives of This Book](#)

[Audience for This Book](#)

[Structure of This Book](#)

[Conventions Used in This Book](#)

[Comments and Questions](#)

[Acknowledgments](#)

[Chapter 1. Introduction](#)

[1.1 Why Tune SQL?](#)

[1.2 Who Should Tune SQL?](#)

[1.3 How This Book Can Help](#)

[1.4 A Bonus](#)

[1.5 Outside-the-Box Solutions](#)

[Chapter 2. Data-Access Basics](#)

[2.1 Caching in the Database](#)

[2.2 Tables](#)

[2.3 Indexes](#)

[2.4 Uncommon Database Objects](#)

[2.5 Single-Table Access Paths](#)

[2.6 Calculating Selectivity](#)

[2.7 Joins](#)

[Chapter 3. Viewing and Interpreting Execution Plans](#)

[3.1 Reading Oracle Execution Plans](#)

[3.2 Reading DB2 Execution Plans](#)

[3.3 Reading SQL Server Execution Plans](#)

## Chapter 4. Controlling Execution Plans

### 4.1 Universal Techniques for Controlling Plans

### 4.2 Controlling Plans on Oracle

### 4.3 Controlling Plans on DB2

### 4.4 Controlling Plans on SQL Server

## Chapter 5. Diagramming Simple SQL Queries

### 5.1 Why a New Method?

### 5.2 Full Query Diagrams

### 5.3 Interpreting Query Diagrams

### 5.4 Simplified Query Diagrams

### 5.5 Exercises (See Section A.1 for the solution to each exercise.)

## Chapter 6. Deducing the Best Execution Plan

### 6.1 Robust Execution Plans

### 6.2 Standard Heuristic Join Order

### 6.3 Simple Examples

### 6.4 A Special Case

### 6.5 A Complex Example

### 6.6 Special Rules for Special Cases

### 6.7 Exercise (See Section A.2 for the solution to the exercise.)

## Chapter 7. Diagramming and Tuning Complex SQL Queries

### 7.1 Abnormal Join Diagrams

### 7.2 Queries with Subqueries

### 7.3 Queries with Views

### 7.4 Queries with Set Operations

### 7.5 Exercise (See Section A.3 for the solution to the exercise.)

## Chapter 8. Why the Diagramming Method Works

### 8.1 The Case for Nested Loops

### 8.2 Choosing the Driving Table

### 8.3 Choosing the Next Table to Join

### 8.4 Summary

## Chapter 9. Special Cases

[9.1 Outer Joins](#)

[9.2 Merged Join and Filter Indexes](#)

[9.3 Missing Indexes](#)

[9.4 Unfiltered Joins](#)

[9.5 Unsolvable Problems](#)

[Chapter 10. Outside-the-Box Solutions to Seemingly Unsolvable Problems](#)

[10.1 When Very Fast Is Not Fast Enough](#)

[10.2 Queries that Return Data from Too Many Rows](#)

[10.3 Tuned Queries that Return Few Rows, Slowly](#)

[Appendix A. Exercise Solutions](#)

[A.1 Chapter 5 Exercise Solutions](#)

[A.2 Chapter 6 Exercise Solution](#)

[A.3 Chapter 7 Exercise Solution](#)

[Appendix B. The Full Process, End to End](#)

[B.1 Reducing the Query to a Query Diagram](#)

[B.2 Solving the Query Diagram](#)

[B.3 Checking the Execution Plans](#)

[B.4 Altering the Database to Enable the Best Plan](#)

[B.5 Altering the SQL to Enable the Best Plan](#)

[B.6 Altering the Application](#)

[B.7 Putting the Example in Perspective](#)

[Glossary](#)

[Colophon](#)

[Index](#)

[Index SYMBOL](#)

[Index A](#)

[Index B](#)

[Index C](#)

[Index D](#)

[Index E](#)

[Index F](#)  
[Index G](#)  
[Index H](#)  
[Index I](#)  
[Index J](#)  
[Index K](#)  
[Index L](#)  
[Index M](#)  
[Index N](#)  
[Index O](#)  
[Index P](#)  
[Index Q](#)  
[Index R](#)  
[Index S](#)  
[Index T](#)  
[Index U](#)  
[Index V](#)  
[Index W](#)



- [Table of Contents](#)
- [Index](#)
- [Reviews](#)
- [Examples](#)
- [Reader Reviews](#)
- [Errata](#)
- [Academic](#)

## SQL Tuning

By [Dan Tow](#)

Start Reading ►

Publisher : O'Reilly

Pub Date : November 2003

ISBN : 0-596-00573-3

Pages : 336

*SQL Tuning* author Dan Tow outlines a timesaving method he's developed for finding the optimum execution plan--rapidly and systematically--regardless of the complexity of the SQL or the database platform being used. You'll learn how to understand and control SQL execution plans and how to diagram SQL queries to deduce the best execution plan for a query. Key chapters in the book includes exercises to reinforce the concepts you've learned. *SQL Tuning* concludes by addressing special concerns and unique solutions to

"unsolvable" problems.



- [Table of Contents](#)

- [Index](#)

- [Reviews](#)

- [Examples](#)

- [Reader Reviews](#)

- [Errata](#)

- [Academic](#)

## SQL Tuning

By [Dan Tow](#)

Start Reading ►

Publisher : O'Reilly

Pub Date : November 2003

ISBN : 0-596-00573-3

Pages : 336

[Copyright](#)

[Dedication](#)

[Foreword](#)

[Preface](#)

[Objectives of This Book](#)

[Audience for This Book](#)

[Structure of This Book](#)

[Conventions Used in This Book](#)

[Comments and Questions](#)

[Acknowledgments](#)

[Chapter 1. Introduction](#)

[Section 1.1. Why Tune SQL?](#)

[Section 1.2. Who Should Tune SQL?](#)

[Section 1.3. How This Book Can Help](#)

[Section 1.4. A Bonus](#)

[Section 1.5. Outside-the-Box Solutions](#)

[Chapter 2. Data-Access Basics](#)

[Section 2.1. Caching in the Database](#)

[Section 2.2. Tables](#)

[Section 2.3. Indexes](#)

[Section 2.4. Uncommon Database Objects](#)

[Section 2.5. Single-Table Access Paths](#)

[Section 2.6. Calculating Selectivity](#)

[Section 2.7. Joins](#)

[Chapter 3. Viewing and Interpreting Execution Plans](#)

[Section 3.1. Reading Oracle Execution Plans](#)

[Section 3.2. Reading DB2 Execution Plans](#)

[Section 3.3. Reading SQL Server Execution Plans](#)

[Chapter 4. Controlling Execution Plans](#)

[Section 4.1. Universal Techniques for Controlling Plans](#)

[Section 4.2. Controlling Plans on Oracle](#)

[Section 4.3. Controlling Plans on DB2](#)

[Section 4.4. Controlling Plans on SQL Server](#)

[Chapter 5. Diagramming Simple SQL Queries](#)

[Section 5.1. Why a New Method?](#)

[Section 5.2. Full Query Diagrams](#)

[Section 5.3. Interpreting Query Diagrams](#)

[Section 5.4. Simplified Query Diagrams](#)

[Section 5.5. Exercises \(See Section A.1 for the solution to each exercise.\)](#)

[Chapter 6. Deducing the Best Execution Plan](#)

[Section 6.1. Robust Execution Plans](#)

[Section 6.2. Standard Heuristic Join Order](#)

[Section 6.3. Simple Examples](#)

[Section 6.4. A Special Case](#)

[Section 6.5. A Complex Example](#)

[Section 6.6. Special Rules for Special Cases](#)

[Section 6.7. Exercise \(See Section A.2 for the solution to the exercise.\)](#)

[Chapter 7. Diagramming and Tuning Complex SQL Queries](#)

[Section 7.1. Abnormal Join Diagrams](#)

[Section 7.2. Queries with Subqueries](#)

[Section 7.3. Queries with Views](#)

[Section 7.4. Queries with Set Operations](#)

[Section 7.5. Exercise \(See Section A.3 for the solution to the exercise.\)](#)

[Chapter 8. Why the Diagramming Method Works](#)

[Section 8.1. The Case for Nested Loops](#)

[Section 8.2. Choosing the Driving Table](#)

[Section 8.3. Choosing the Next Table to Join](#)

[Section 8.4. Summary](#)

[Chapter 9. Special Cases](#)

[Section 9.1. Outer Joins](#)

[Section 9.2. Merged Join and Filter Indexes](#)

[Section 9.3. Missing Indexes](#)



[Section 9.4. Unfiltered Joins](#)

[Section 9.5. Unsolvable Problems](#)

[Chapter 10. Outside-the-Box Solutions to Seemingly Unsolvable Problems](#)

[Section 10.1. When Very Fast Is Not Fast Enough](#)

[Section 10.2. Queries that Return Data from Too Many Rows](#)

[Section 10.3. Tuned Queries that Return Few Rows, Slowly](#)

[Appendix A. Exercise Solutions](#)

[Section A.1. Chapter 5 Exercise Solutions](#)

[Section A.2. Chapter 6 Exercise Solution](#)

[Section A.3. Chapter 7 Exercise Solution](#)

[Appendix B. The Full Process, End to End](#)

[Section B.1. Reducing the Query to a Query Diagram](#)

[Section B.2. Solving the Query Diagram](#)

[Section B.3. Checking the Execution Plans](#)

[Section B.4. Altering the Database to Enable the Best Plan](#)

[Section B.5. Altering the SQL to Enable the Best Plan](#)

[Section B.6. Altering the Application](#)

[Section B.7. Putting the Example in Perspective](#)

[Glossary](#)

[Colophon](#)

[Index](#)

# Copyright

Copyright © 2004 O'Reilly & Associates, Inc.

Printed in the United States of America.

Published by O'Reilly & Associates, Inc., 1005 Gravenstein Highway North, Sebastopol, CA 95472.

O'Reilly & Associates books may be purchased for educational, business, or sales promotional use. Online editions are also available for most titles (<http://safari.oreilly.com>). For more information, contact our corporate/institutional sales department: (800) 998-9938 or [corporate@oreilly.com](mailto:corporate@oreilly.com).

Nutshell Handbook, the Nutshell Handbook logo, and the O'Reilly logo are registered trademarks of O'Reilly & Associates, Inc. Many of the designations used by manufacturers and sellers to distinguish their products are claimed as trademarks. Where those designations appear in this book, and O'Reilly & Associates, Inc. was aware of a trademark claim, the designations have been printed in caps or initial caps. The association between the image of salamander and the topic of SQL tuning is a trademark of O'Reilly & Associates, Inc.

While every precaution has been taken in the preparation of this book, the publisher and authors assume no responsibility for errors or omissions, or for damages resulting from the use of the information contained herein.

# Dedication

*To Prarva, and Abe,  
for a dream of more,  
and life surpassing the dream*

# Foreword

During my years as a developer, and later as a database administrator, the only performance tuning area in which I developed any expertise and had any level of success was that of tuning individual SQL statements. Because of that background, I was very interested when Dan Tow came to me with his idea for a book on SQL tuning.

The problem with SQL tuning, at least from my own perspective, is that it was often reasonably easy to pinpoint badly performing SQL statements that needed to be tuned, and it was reasonably easy to determine the execution plan currently being used for those badly performing statements, but then came the hard problem: finding a better execution plan, to make an offending SQL statement execute faster. Many is the time I've stared at a poorly performing SQL statement, reviewing its poorly performing execution plan, and wondered just what I should do next. For that matter, was there even any room for improvement? Perhaps the execution plan that performed so badly from a user's perspective was, in fact, the best possible execution plan. Perhaps I was wasting my time trying to guess at a better one.

There, I used that word *guess*, and that's at the heart of what sometimes made tuning SQL statements a frustrating activity. It all came down to my looking at a SQL statement and trying to guess at a better plan. Of course, I'd attempt to factor in my experience, my intuition, and my knowledge of the data being queried, and I'd pull out tips and tricks that I'd read about in books and magazine articles, but in the end I'd make a guess, try a new plan, make a guess, try a new plan, and so forth. I'd stop when one of two things happened:

- I got lucky and guessed a plan that was a good-enough improvement over the old plan to satisfy my client.
- I ran out of ideas.

It always bothered me when I ran out of ideas, because I never knew for sure whether the current plan really was optimal or whether I was just too thick-headed to intuit a better-performing execution plan than the current one. To be honest, I was always pretty hard on myself, chalking up any failure to improve a SQL statement as a personal inadequacy.

Dan doesn't guess. Let me boldface that and underline it. He doesn't execute an iterative guess-loop like I did, trying one idea after another in the hope of stumbling across an improvement. Instead, Dan uses an innovative and

mathematically based diagramming method to derive the optimal, or near-optimal, execution plan for a SQL statement. Then he puts that plan into effect. And that's it. There's no guesswork, and there's no uncertainty as to whether further improvement is possible.

At first, I was skeptical of Dan's approach. But the more I read his chapters, the more I began to see the logic behind his method. Math doesn't lie, and experience is a good indicator. Dan has over 10 years of tuning experience, and he's been very successful using the method described in this book.

There are three legs that any SQL tuning effort needs to stand on. You need to know how to identify a badly performing SQL statement. You then need to be able to see the execution plan used for that statement. Finally, you need to somehow come up with an improved plan. I'm convinced that Dan's method represents a viable third leg on which your SQL tuning efforts can rest. Read his book, apply his method, and save yourself hours of wondering what to do next.

Jonathan Gennick, author, editor, Oracle DBA