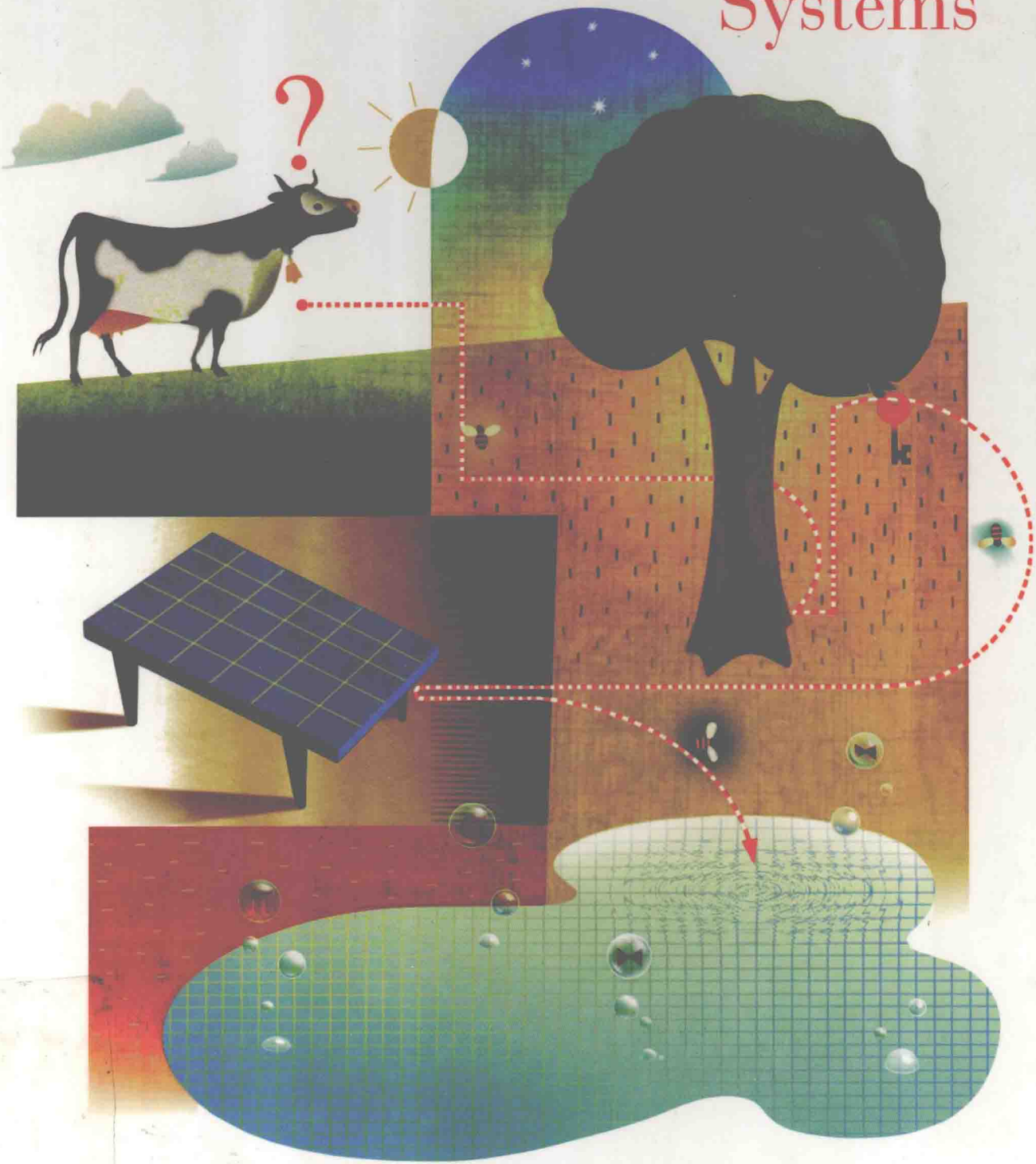
Database Management Systems



Raghu Ramakrishnan

DATABASE MANAGEMENT SYSTEMS

Raghu Ramakrishnan
University of Wisconsin
Madison, WI, USA

Boston, Massachusetts • Burr Ridge, Illinois
Dubuque, Iowa • Madison, Wisconsin
New York, New York • San Francisco, California
St. Louis, Missouri

WCB/McGraw-Hill
*A Division of the **McGraw-Hill** Companies*

DATABASE MANAGEMENT SYSTEMS

This book is printed on acid-free paper.

3 4 5 7 8 9 0 DOC/DOC 9 0 9 8

ISBN 0-07-050775-9

Publisher: Tom Casson
Executive editor: Elizabeth A. Jones
Developmental editor: Bradley Kosirog
Editorial assistant: Emily Gray
Marketing manager: John Wannemacher
Project manager: Jim Labeots
Production supervisor: Heather D. Burbridge
Designer: Larry J. Cope
Cover illustration: Mike Wiggins
Typeface: Times-Roman
Printer: R.R. Donnelley & Sons Company

Library of Congress Cataloging-in-Publication Data

Ramakrishnan, Raghu.
Database management systems / Raghu Ramakrishnan.
p. cm.
Includes bibliographical references and index.
ISBN 0-07-050775-9
1. Database management. I. Title.
QA76.9.D3R237 1997
005.74–dc2d1 97-23748

http://www.mhhe.com

To Apu with love

PREFACE

> The advantage of doing one's praising for oneself is that one can lay it on so thick and exactly in the right places.
>
> —Samuel Butler

One of the virtues of a relational database system is that it provides a high-level, declarative interface to users. In practice, however, sophisticated users, database administrators and application programmers must have a thorough grasp of what the system does in response to user-level requests. To make the most effective use of a database system, one must have insight into topics such as logical and physical schema design, how a query optimizer chooses an execution plan for a given query, the impact of indexing on performance, and the importance of memory available for buffering. From a performance perspective, all these factors are related, and I believe that they are best understood through a hands-on approach.

This book covers the fundamentals of modern database management systems, in particular relational database systems. It is intended as a text for an introductory database course for undergraduates, and I have attempted to present the material in a clear, simple style. A quantitative approach is used throughout and detailed examples abound. An extensive set of exercises (for which solutions are available online to instructors) accompanies each chapter, and reinforces students' ability to apply the concepts to real problems. The book contains enough material to support a second course, ideally supplemented by selected research papers. It can be used, with the accompanying software, in two distinct kinds of courses:

1. A course that aims to present the principles of database systems, with a practical focus but without any implementation assignments. Optionally, the software can be used to create various exercises and experiments that involve no programming.

2. A course that has a strong systems emphasis and assumes that students have good programming skills in C and C++. In this case the software can be used as the basis for projects in which students are asked to implement various parts of a relational DBMS. Several central modules in the project software (e.g., heap files, buffer manager, B+ trees, hash indexes, various join methods, concurrency control and recovery algorithms) are described in

sufficient detail in the text to enable students to implement them, given the (C++) class interfaces.

Many instructors will no doubt teach a course that falls between these two extremes with a mix of experiment-oriented and implementation assignments.

Note: Exercises using the Minibase software are outlined in several chapters. These exercises should be suitably expanded by instructors, as described in Appendix A. Detailed Minibase exercises and solutions are available online for instructors.

Choice of Topics

The main pedagogical objective of this book is to provide clear and thorough discussions of the topics covered. The choice of material has been influenced by these considerations:

- To concentrate on issues central to the *design, tuning, and implementation of relational database applications.* However, many of the issues discussed (e.g., buffering and access methods) are not specific to relational systems, and additional topics such as decision support and object-database systems are covered in later chapters.
- To provide adequate coverage of implementation topics to support a concurrent laboratory section or course project. For example, implementation of relational operations has been covered in more detail than is necessary in a first course. However, the variety of alternative implementation techniques permits a wide choice of project assignments. An instructor who wishes to assign implementation of sort-merge join might cover that topic in depth, whereas another might choose to emphasize index nested loops join.
- To provide in-depth coverage of the state of the art in currently available commercial systems, rather than a broad coverage of several alternatives. For example, we discuss the relational data model, B+ trees, SQL, System R style query optimization, lock-based concurrency control, the ARIES recovery algorithm, the two-phase commit protocol, asynchronous replication in distributed databases, and object-relational DBMSs in detail, with numerous illustrative examples. This is made possible by omitting or briefly covering some related topics such as the hierarchical and network models, B tree variants, Quel, randomized and semantic query optimization, view serializability, the shadow-page recovery algorithm, and the three-phase commit protocol.

Organization

The material can be divided into roughly seven parts, as indicated in Figure 0.1, which also shows the dependencies between chapters. An arrow from Chapter I to Chapter J means that I depends on material in J. The broken arrows indicate a weak dependency, which can be ignored at the instructor's discretion. The first two chapters cover material that is basic to the rest of the book, and we assume that they are covered first. Chapter 2 presents the relational model and uses SQL-92 constructs to make these concepts concrete. The early introduction of SQL facilitates assignments using a relational DBMS and also offers instructors the choice of covering the ER model early—the discussion of how to translate ER diagrams into tables assumes knowledge of SQL constructs for creating tables, and this prerequisite material is included in Chapter 2.

Each of the remaining six parts is described below, along with its dependence, if any, on the other parts.

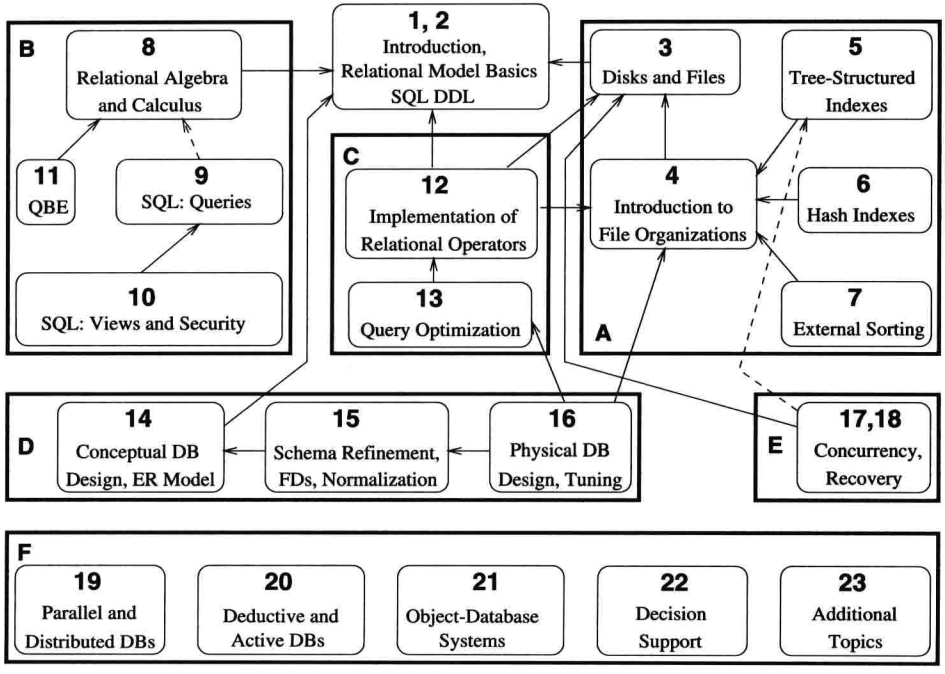


Figure 0.1 Dependencies between Chapters

- **Part A (File Organizations):** Storage media, buffer management, record and page formats, file organizations, various kinds of indexes, and external sorting are discussed in detail.

- **Part B (Query Languages):** Relational algebra and calculus, QBE and SQL. It is worth noting that the coverage of SQL, which follows the SQL-92 standard, is quite extensive. In addition to the data retrieval features of SQL, many important concepts, such as authorization, views and referential integrity, are discussed in the context of SQL.

  Chapter 9 can be taught before Chapter 8. This order may be desirable if an instructor wants students to write SQL queries early. However, an understanding of algebra and calculus will enable students to appreciate the foundations of SQL. Similarly, QBE can be taught without a prior discussion of DRC, but briefly covering DRC first is recommended.

- **Part C (Query Optimization and Evaluation):** Implementation of relational operators and generation of query plans. A basic understanding of indexes is assumed in the discussion of query evaluation. If implementation projects are assigned, covering Chapter 4 before Part C is recommended.

- **Part D (Database Design and Tuning):** Conceptual design, normalization, physical design and performance tuning. The discussion in Chapter 16 assumes a good understanding of query optimization, and in particular, of the use of indexes. In an important sense, Parts A, B and C lead up to the material on database design and tuning. I discuss these topics from a very practical perspective, with several examples. A reader who studies this material carefully will quickly realize that a thorough understanding of several basic issues is essential for good design. The goal is to ensure that the earlier chapters provide the reader with the necessary foundation.

- **Part E (Transaction Processing):** Concurrency control and recovery are covered with an emphasis on locking and logging techniques. The discussion of concurrency control in tree indexes depends on a knowledge of tree indexes.

- **Part F (Advanced Topics):** The discussion of query optimization in parallel and distributed databases assumes an understanding of query optimization in a centralized DBMS. In general, the chapters in Part F should be covered after material from the other parts has been covered. There is little dependence between chapters in Part F, and they can be covered in any order. Chapters on deductive and active databases, object-oriented database extensions, and decision support (data warehousing, data mining and OLAP) are included.

Order of Presentation

The material in the text is presented in the order in which I usually cover it, and is influenced by the order of the accompanying programming assignments, and by my preference for covering all database design topics together *after* covering all

optimization related material. Other instructors may want to cover some topics in a different order, based on the needs of their courses. For example, some instructors will want to cover SQL, and perhaps get students to use a relational database, before discussing file organizations or indexing. As another example, in a course that emphasizes database design many of the implementation-oriented chapters might well be skipped altogether, and the ER model may be covered before SQL. The book has therefore been designed to be flexible with respect to the ordering of material, and inter-chapter dependencies have been kept to a minimum. In particular, Parts A through E, regarded as units, can be covered in pretty much any order, although it is best to cover Part A before Part C and before Chapter 16. The remaining inter-part dependencies are minor.

Some additional points to note:

- Several section headings contain an asterisk. This symbol does not necessarily indicate a higher level of difficulty. Rather, omitting all asterisked sections leaves about the right amount of material in Chapters 1 through 18 for a broad introductory one-quarter or one-semester course (depending on the depth at which the remaining material is discussed and the nature of the course assignments).
- It is not necessary to cover all the alternatives for a given operator (e.g., various techniques for joins) in Chapter 12 in order to cover Chapter 13 adequately.

The book contains more material than can be covered in a one-semester course. It can be used in several kinds of introductory or second courses by choosing topics appropriately, or in a two-course sequence by supplementing the material with some advanced readings in the second course. Examples of appropriate introductory courses include courses on file organizations, and introduction to database management systems, especially if the course focuses on relational database design or implementation. Advanced courses can be built around the later chapters, which contain detailed bibliographies with ample pointers for further study.

Supplementary Material

Each chapter contains several exercises designed to test and expand the reader's understanding of the material. Instructors can obtain a set of lecture slides for each chapter through the Web in Postscript and Adobe PDF formats. Slides in Microsoft Powerpoint and a solution manual for the exercises are also freely available online upon request to the author (`raghu@cs.wisc.edu`).

Supplementary project software with sample assignments and solutions is available through the Internet, as described in Appendix A. The text itself does not refer to the project software, however, and can be used independently in a course that presents the principles of database management systems from a practical perspective, but without a project component.

For More Information

The home page for this book is at URL:

`http://www.cs.wisc.edu/~dbbook`

This page is frequently updated and contains information about the book, past and current users, and the software. *This page also contains a link to all known errors in the book, the accompanying slides, and the software.* Instructors are advised to visit this site periodically; they can also register at this site to be notified of important changes by email.

Acknowledgments

This book grew out of lecture notes for CS564, the introductory (senior/graduate level) database course at UW-Madison. David DeWitt developed this course and the Minirel course project, in which students wrote several well-chosen parts of a relational DBMS. My thinking about this material was shaped by teaching CS564, and Minirel was the inspiration for Minibase, which is more comprehensive (e.g., it has a query optimizer and includes visualization software) but tries to retain the spirit of Minirel. Mike Carey and I jointly designed much of Minibase. My lecture notes (and in turn many chapters in this book) were influenced by Mike's lecture notes and by Yannis Ioannidis's lecture slides.

Joe Hellerstein used the beta edition of the book at Berkeley, and provided invaluable feedback, assistance on slides, and hilarious quotes. Writing the chapter on object-database systems with Joe was a lot of fun.

C. Mohan provided invaluable assistance, patiently answering a number of questions about implementation techniques used in various commercial systems, in particular indexing, concurrency control and recovery algorithms. Moshe Zloof answered numerous questions about QBE semantics and commercial systems based on QBE. Ron Fagin, Krishna Kulkarni, Len Shapiro, Jim Melton, Dennis Shasha and Dirk Van Gucht reviewed the book and provided detailed feedback, greatly improving the content and presentation. Michael Goldweber at Beloit College, Matthew Haines at Wyoming, Michael Kifer at SUNY StonyBrook, Jeff

Naughton at Wisconsin, Praveen Seshadri at Cornell and Stan Zdonik at Brown also used the beta edition in their database courses and offered feedback and bug reports. In particular, Michael Kifer pointed out an error in the (old) algorithm for computing a minimal cover and suggested covering some SQL features in Chapter 2 to improve modularity. Gio Wiederhold's bibliography, converted to Latex format by S. Sudarshan, and Michael Ley's online bibliography on databases and logic programming were a great help while compiling the chapter bibliographies. Shaun Flisakowski and Uri Shaft helped me frequently in my never-ending battles with Latex.

I owe a special thanks to the many, many students who have contributed to the Minibase software. Emmanuel Ackaouy, Jim Pruyne, Lee Schumacher and Michael Lee worked with me when I developed the first version of Minibase (much of which was subsequently discarded, but which influenced the next version). Emmanuel Ackaouy and Bryan So were my TAs when I taught CS564 using this version and went well beyond the limits of a TAship in their efforts to refine the project. Paul Aoki struggled with a version of Minibase and offered lots of useful comments as a TA at Berkeley. An entire class of CS764 students (our graduate database course) developed much of the current version of Minibase in a large class project that was led and coordinated by Mike Carey and me. Amit Shukla and Michael Lee were my TAs when I first taught CS564 using this version of Minibase, and developed the software further.

Several students worked with me on independent projects, over a long period of time, to develop several Minibase components. These include visualization packages for the buffer manager and B+ trees (Huseyin Bektas, Harry Stavropoulos and Weiqing Huang); a query optimizer and visualizer (Stephen Harris, Michael Lee and Donko Donjerkovic); an ER diagram tool based on the Opossum schema editor (Eben Haber); and a GUI-based tool for normalization (Andrew Prock and Andy Therber). In addition, Bill Kimmel worked to integrate and fix a large body of code (storage manager, buffer manager, files and access methods, relational operators and the query plan executor) produced by the CS764 class project. Ranjani Ramamurty considerably extended Bill's work on cleaning up and integrating the various modules. Luke Blanshard, Uri Shaft and Shaun Flisakowski worked on putting together the release version of the code, and developed test suites and exercises based on the Minibase software. Krishna Kunchithapadam worked on testing the optimizer and developed part of the Minibase GUI front-end.

Clearly, the Minibase software would not exist without the contributions of a great many talented people. With this software available freely in the public domain, I hope that more instructors will be able to teach a systems-oriented database course with a blend of implementation and experimentation to complement the lecture material.

I'd like to thank the many students who helped in developing and checking the solutions to the exercises and provided useful feedback on draft versions of the book. In alphabetical order: X. Bao, S. Biao, M. Chakrabarti, C. Chan, W. Chen, N. Cheung, D. Colwell, C. Fritz, V. Ganti, J. Gehrke, G. Glass, V. Gopalakrishnan, M. Higgins, T. Jasmin, M. Krishnaprasad, Y. Lin, C. Liu, M. Lusignan, H. Modi, S. Narayanan, D. Randolph, A. Ranganathan, J. Reminga, A. Therber, M. Thomas, Q. Wang, R. Wang, Z. Wang and J. Yuan. James Harrington and Martin Reames at Wisconsin and Nina Tang at Berkeley provided especially detailed feedback.

Charlie Fischer, Avi Silberschatz and Jeff Ullman gave me invaluable advice on working with a publisher. My editors at McGraw-Hill, Betsy Jones and Eric Munson, obtained extensive reviews and shepherded this book through two beta editions, greatly improving its form and content. Emily Gray and Brad Kosirog were there whenever problems cropped up. At Wisconsin, Ginny Werner really helped me to stay on top of things.

Finally, this book was a thief of time, and in many ways it was harder on my family than on me. My sons expressed themselves forthrightly. From my (then) five-year-old, Ketan: "Dad, stop working on that silly book. You don't have any time for *me*." Two-year-old Vivek: "You working *boook*? No no no come play basketball me!" All the seasons of their discontent were visited upon my wife, and Apu nonetheless cheerfully kept the family going in its usual chaotic, happy way all the many evenings and weekends I was wrapped up in this book. (Not to mention the days when I was wrapped up in being a faculty member!) As in all things, I can trace my parents' hand in much of this; my father, with his love of learning, and my mother, with her love of us, shaped me. My brother Kartik's contributions to this book consisted chiefly of phone calls in which he kept me from working, but if I don't acknowledge him, he's liable to be annoyed. I'd like to thank my family for being there and giving meaning to everything I do. (There! I knew I'd find a legitimate reason to thank Kartik.)

CONTENTS