UNIX® Relational Database Management

Application Development in the UNIX® Environment

ROD MANIS
EVAN SCHAFFER
ROBERT JØRGENSEN

UNIX®
Relational Database Management

Application Development in the UNIX Environment

Rod Manis
Evan Schaffer
Robert Jørgensen

PRENTICE HALL
Englewood Cliffs, New Jersey 07632

Library of Congress Cataloging-in-Publication Number 87-36103

Cover design: Lundgren Graphics, Inc.
Manufacturing buyer: Mary Ann Gloriande

This book was typeset by the authors. UNIX programs formatted the text and printed the final copy. We acknowledge the special assistance of Barbara Wright, who is responsible for the description of ve in Chapter 5.

Prentice Hall Software Series
Brian W. Kernighan, Advisor

Printed in the United States of America

10 9 8 7 6 5 4 3 2 1

ISBN 0-13-938622-X

Prentice-Hall International (UK) Limited, *London*
Prentice-Hall of Australia Pty. Limited, *Sydney*
Prentice-Hall Canada Inc., *Toronto*
Prentice-Hall Hispanoamericana, S.A., *Mexico*
Prentice-Hall of India Private Limited, *New Delhi*
Prentice-Hall of Japan, Inc., *Tokyo*
Simon & Schuster Asia Pte. Ltd., *Singapore*
Editora Prentice-Hall do Brasil, Ltda., *Rio de Janeiro*

UNIX®
Relational Database Management

YES—I'd like \rdb for my UNIX system

Cut out this form and fill in all necessary information. Then enclose this form in an envelope and mail to:

Robinson Schaffer Wright
711 California Street
Santa Cruz, CA 95060
(408) 429-6229

Please send me pricing information for **\rdb** for my **UNIX** system. I understand that **\rdb** runs on *all* UNIX based computers, micro to mainframe, and most UNIX derivitives (XENIX, Microport System V/AT, 4.3 BSD, AIX, ULTRIX, etc.). Prices for **\rdb** vary by size of system from $795 to $2,495.

My computer is ________________, it runs the following version of UNIX ____________, and supports _____users.

Name_____________________________________
Firm/Department______________________________
City________________State_______Zip___________
Daytime telephone number(_____) _____ _________

NOTE: Prices subject to change without notice.

YES—I'd like /rdb and/or MKS Toolkit for DOS for my PC-DOS or MS-DOS system

The MKS Toolkit provides over 100 UNIX like commands and is required to take advantage of /rdb in a DOS environment (version 2.0 and later releases).

Cut out this form and fill in all necessary information. Then enclose this form with your check or money order *only* in an envelope and mail to:

Robinson Schaffer Wright
711 California Street
Santa Cruz, CA 95060
(408) 429-6229

Please send the items checked below. PAYMENT ENCLOSED (check or money order *only*). Robinson Schaffer Wright will pay all shipping and handling charges.

_______/rdb and MKS Toolkit for DOS—$275
_______/rdb for DOS only. I have MKS Toolkit—$139
_______MKS Toolkit for DOS only (I don't need /rdb yet)—$139
California residents please add 7% sales tax

Name_____________________________________
Firm/Department______________________________
City________________State_______Zip_________
Daytime telephone number (_____) _____ _________

NOTE: Prices subject to change without notice.

This book is dedicated to the people who have made
/rdb
possible

Thomas Arnow
Robert Berkley
Andy Chang
Michael J. Defazio
Steve Eberhart
Keith Eisenstark
Raymond Eisenstark
David Fiedler
Tom Johnson
Ray Jones
Brian Kernighan
Paul S. Kleppner
Mark Krieger
Gig Graham
William Hamilton
Joel Harrison
Beth Harsaghy
Chris Lynch
John Mashey
Marc Meyer
Phoebe Miller
Eli Nielsen
Fred Pack
Durk Pearson
Ron Posner
Pat Rafter
William Robinson
Pat Sarma
David Schmidt
Tom Slezak
William P. Spencer
Shawn Steel
Ed Taylor
Roy Takai
Zhang Yuchao
Richard Vento
Peter Vizel
Jerry Walker
Colin Watanabee
Barbara Wright
Myron Zimmerman

Preface

This book shows you how to develop software applications in a UNIX environment, in a small fraction of the time and effort needed with other systems. The UNIX environment is so different that it makes possible a new approach to computing.

There are many books available on writing C programs, administering a UNIX system, and specific software like spreadsheets and word processors. But this book tells you how to put them all together. System integration is the hardest part of software development. Therefore, we have filled this book with examples, solutions, source code, tricks, hints, and useable theory to help you.

Each of the authors has over 20 years of experience in computing in which we have had to put together an enormous number of applications on many different systems for a wide range of users. We have tried to put as much of what we have learned into this book as possible.

We emphasize database management because almost all applications require data to be entered, manipulated, searched, or reported. By learning simple database operations, you can approach any application with a powerful tool kit.

Model Business System

We have even included, as an example to get you started, a model business system with general ledger, tax, accounts receivable and payable, inventory, payroll, and operations. Included is a tutorial, manual pages, and source code of the UNIX shell scripts. They show you how to use the system, and also how to manage a business with a computer.

New Approach to Relational Database Management

Most books on relational database theory take a very mathematical approach and use a difficult language. They talk of relations, tuples, attributes, degrees, cardinality, and normalization. We've translated these terms into familiar words like tables, rows, and columns. We also show a new approach to normalizing tables which anyone can understand. We have tried to make relational database theory accessible to everyone.

The UNIX System

The UNIX system, of course, is among the greatest software tools available. It is portable, multi-user, and multi-tasking. It runs on hundreds of computers, from the IBM PC/XT to the Cray 2. It has been adopted as the standard operating system by the U.S. federal government (POSNIX), and many corporations and universities. But the most important features of the UNIX system are the UNIX shell, the software tools, and the *pipe and filter* approach.

To get the full power of the UNIX system, one must relearn and rethink how to do software development. Most software people are not aware of what they can do with the UNIX system. They use the same approach that they learned on other systems. It is as if they had a Porsche, and hitched a horse to it, not knowing that they could start its engine and roar away. In this book we will teach you how to drive, but you may have to give up your attachment to old fashioned horse and buggy styles of programming.

UNIX Shell is the Best Fourth Generation Programming Language

It is the UNIX shell that makes it possible to do applications in a small fraction of the code and time it takes in third generation languages. In the shell you process whole files at a time, instead of only a line at a time. And, a line of code in the UNIX shell is one or more programs, which do more than pages of instructions in a 3GL. Applications can be developed in hours and days, rather than months and years with traditional systems. Most of the other 4GLs available today look more like COBOL or RPG, the most tedious of the third generation languages.

Flat File Database

There are two approaches to database management. Traditionally, a large database management program had to be written to run on the old mainframe and mini computer operating systems which provided only a minimum of assistance. Therefore, all of the functions needed had to be coded. Each system had its own language and data formats. They seldom talked to each other. It was hard to pass data from one to the other. It took a lot of work to learn and code them. Then when you moved to another system, you had to learn a new language and figure out how to transfer and convert your data and rewrite your programs.

UNIX solved most of these problems by providing a huge set of tools and facilities and the ability to link them together with pipes and shell programs. Therefore, the burden on a database manager is significantly reduced. To take advantage of the UNIX environment, *flat file* database management systems were developed, including **/rdb** (pronounced slash-r-d-b) from Robinson Schaffer Wright, **Prelude** from VenturCom, **Unity** from AT&T, among others. These database managers work with UNIX and keep their data in flat ASCII UNIX files. You can mix UNIX and database commands in the same shell scripts and pipe commands. Therefore, you do not lose the power of UNIX by going into the database program. You use your UNIX knowledge, and/or learn UNIX skills once. Then you can move to most computers without having to learn *yet another system and language*.

/rdb Relational Database Management System for UNIX

We use **/rdb** for the examples in this book because it is so simple to learn and use, and does not get in the way of your understanding the principles of UNIX database management and applications development. It makes a good teaching aid. Examples with **/rdb** show you how to do basic tasks, without having to spend time learning complex systems. So even if you choose other software packages, you have learned the principles in the easiest way. You can then focus on the more cumbersome syntax of other software packages. **/rdb** is also one of the least expensive of the database managers and is available on any UNIX computer.

At the end of this book is a manual of the **/rdb** commands including examples of their use, and the source code of all of the UNIX shell programs. They will give you many ideas about how to handle specific problems in more detail than is appropriate for the tutorial part of this book.

UNIX and DOS

There is a joke going around that in the future there will be only two operating systems, *UNIX running DOS* and *DOS running UNIX*. It has already happened. The latest UNIX operating systems run DOS applications. And you can get the MKS Toolkit of over 100 UNIX commands, including the latest UNIX korn shell, vi, awk, grep, etc., to run on your DOS. **\rdb** (pronounced backslash-r-d-b), is the DOS version of **/rdb.** Therefore, most of the programs described in this book will also run on DOS (MS-DOS, PC-DOS, OS/2) with the MKS Toolkit and **\rdb.** See the tear out card in this book, which offers a discount on these packages.

Since UNIX and DOS run on most of the world's computers, you can develop an application on one computer and run it on many others. You can also network computers together and run the same programs on each, transferring data in a standard format. It is an old dream come true. But there are lots of details to make it all work, which is what this book is about.

Macintosh

We see no conflict between the UNIX/DOS approach and the Macintosh approach. In an integrated computing environment, we'll be able to 'point and click' on, for example, a shell script, or any arbitrary collection of commands. The concept of relations as flat files and relational operators as programs is the common ground.

Required Knowledge

You should know how to log on to UNIX, to use its commands, to use a text editor like **vi.** You should be aware of the method of using UNIX programs, often piped together, to get things done without having to write C or other language programs. You do not have to know how to program, although it helps.

There are many introductory books on UNIX and a few books on advanced UNIX and shell programming: *The UNIX™ Shell Programming Language* by Rod Manis and Marc H. Meyer [Manis 1986]. For more advanced UNIX, you should read Stephen Prata's *Advanced UNIX™ - A Programmer's Guide* [Prata 1985], and Brian Kernighan and Rob Pike's book *The UNIX™ Programming Environment* [Kernighan 1984], and the new Kernighan book on **awk.** See the bibliography for more complete citations.

This book starts where those leave off, and covers the details of putting together real applications with databases. One or more of these books should be read because they teach the UNIX and shell programming basics that we build upon. We assume that basic knowledge so that we can focus on application development. But even if you are new to UNIX, you should try to get started. If you get stuck, you can look up ideas in the other books and in your system's UNIX documentation.

To help overcome the old beliefs, we show how to develop and use software applications in the UNIX environment, covering all of the important tasks of programming and setting up computer systems to do real work for users. A beginner will learn the basics of database design and fourth generation programming. Experienced programmers will learn how to work in this new environment. Many UNIX gurus understand in principle that UNIX is a far better environment, but need to know more of the details of how to handle certain problems.

Computer Revolution

This book aims at nothing less than a revolution in computing! It challenges some of the most encrusted assumptions and proposes a simple, elegant approach which solves many of the major problems in computing today.

1. Data should be kept in flat ASCII tables (files), not binary, so that we can always see what we are doing, and do not have to depend upon some special program to decode our data for us.
2. Programming should be done in a fourth generation programming language, the most powerful of which is the UNIX shell, not in lower level languages, except in extreme cases.
3. Programs should be small and should pass data on to other programs. Software prisons, or large programs with self-contained environments, must be avoided because they require learning and they make extracting data difficult.
4. All programs should be integrated with a common interface with both users and data. All of our tools should look more or less the same for quick learning and easy use.
5. We should build software and systems to meet interface standards so that we can share software and stop dreaming that any individual or company can do it all from scratch.
6. We should use networks to tie together computers and software so that simple tables of data can be passed from one to another.
7. Industrial strength artificial intelligence and expert systems will have to be developed in the same environment previously described, and not in toy languages like LISP and PROLOG.
8. The UNIX system embodies the most advanced facilities for implementing these principles.

Rod Manis
Evan Schaffer
Robert Jørgensen

Foreword

I stumbled into using UNIX in late 1978, in a Bio-Medical Research environment. The machine was a PDP 11-34 running V6 UNIX, with custom automated microscopy gear used for image processing of mammalian cells. In those days there was the Kernighan and Ritchie C book and the UNIX manuals themselves; nothing more. Two degrees in computer science prepared me to tackle UNIX but I was still perhaps understandably shy about investigating all of the 100+ tools, including such strange ones as **awk**, **lex**, and **yacc**.

I quickly discovered that the use of the UNIX tools freed me from the great bulk of the drudgery of supporting real end users on a computer system. I could string tools together quickly instead of writing small *one shot* programs to do the myriad of data manipulation tasks that characterize real life in the scientific computing world. In the course of my work I acquired a relational database package, but found that it did not mesh well with the UNIX tool-kit philosophy. Like the mammoth database systems I had previously used on large IBM systems, one had to *drop into* the database package, where a whole new command language applied and the UNIX tools could only be reached with extreme difficulty, if at all.

By 1983 I had added an early 16-bit microprocessor-based UNIX timesharing system running V7 UNIX to my computer shop. I had a growing number of users who were doing general purpose data processing and needed a variety of tools to handle their needs. I was fortunate enough to be in attendance at the Usenix conference where Rod Manis first described his **/rdb** database, that existed as a set of UNIX filters. As he presented his paper I realized, along with numerous other members of the audience, that Rod had grasped and implemented the central abstraction I had been toying with but never formalized: a UNIX database should use ASCII files and operate at the shell level as a series of filters. This simple but brilliant insight of Rod's, brought about by his mastery of a UNIX tool (**awk**) that I had ignored to that point, provided my researchers with the single most powerful tool I have been able to acquire to this date. Along with many others who remained after his talk to discuss his ideas, I was one of the first users of the **/rdb** package. The medical research environment is characterized by chronic lack of funding for computing equipment and scarcity of programming expertise. Without mega-funding, I had no choice but to find ways to let end users solve their own problems as much as possible, without making them all *programmers*.

To my mind, the power of this tool is the freedom it gives to the end users. For the first time I could sit down with a novice and create a working sample database using real data in a matter of minutes. Queries could be demonstrated and stored in *user-friendly* shell scripts. Within hours the end users were fully capable of creating their own databases. Their creativity was amazing: One researcher who had had a single Fortran class eight years earlier wrote a two-page shell script using **/rdb** filters to completely computerize her laboratory data processing and analysis. Other users discovered that the innocent-appearing simple report writer was in reality a powerful meta-tool that allowed them to write shell scripts that wrote shell scripts, yielding in effect a *two-dimensional* program. By word of mouth it was demanded to be bought for at least a dozen other machines at this site. The database for the world's largest melanoma (skin cancer) epidemiology study runs on **/rdb,** as do many other applications.

I have found this abstraction of a database (regular ASCII files, filter programs, normal file access) to serve well for moderate size databases and the frequency of update and query that are common in the scientific research fields. Like most UNIX utilities and indeed UNIX itself, it is always possible to write a single-point solution that would be faster. I have found, however, that the ability to apply the full power of the UNIX toolkit to *database problems* far outweighs any speed penalties that I might be paying on the current generation of super-micros and super-minis. Even in situations where a "TRADITIONAL DATABA$E" is required, the rapid prototyping of the **/rdb** approach is often used to provide the early insights into the proper way to tackle problems requiring exceptional speed or size. I find that writing a custom program to manipulate or calculate data is virtually unnecessary, since such problems can nearly always be solved with the **/rdb** database and regular UNIX filters.

A completely unexpected benefit of this tool-kit approach to databases has been the education of users, many of whom have had no prior computer training or exposure. The early confidence gained in putting up their own simple databases often led to users taking the plunge and developing more complex shell scripts using those tools. In numerous cases, their continued interest led to them delving into **awk** and writing custom front or back end programs. A few adventurous souls then began writing C code in the awk scripts, as well as learning to use **lex**, which after all is merely a C program-generator cleverly (?) disguised as a lexical-analyzer. The net result is that I have had several novice users bootstrap themselves up to be excellent applications programmers in C in an amazingly short time, all due to the good first experience with the database tools!

The evolution of personal computers into machines with sufficient power to run UNIX well (IBM AT, Mac II, etc.) puts the ability to use the UNIX tool-kit philosophy into the reach of every research lab and small business. A book like this is as much a guide to using UNIX itself and the UNIX philosophy of problem solving as it is to being a guide to a specific database. The authors have quietly effected a revolution in the use of databases as a problem-solving tool that is as startling in its clarity and simplicity as the development of the spreadsheet program.

Tom Slezak
Computer Scientist, UNIX Support
Lawrence Livermore National Laboratory

Contents