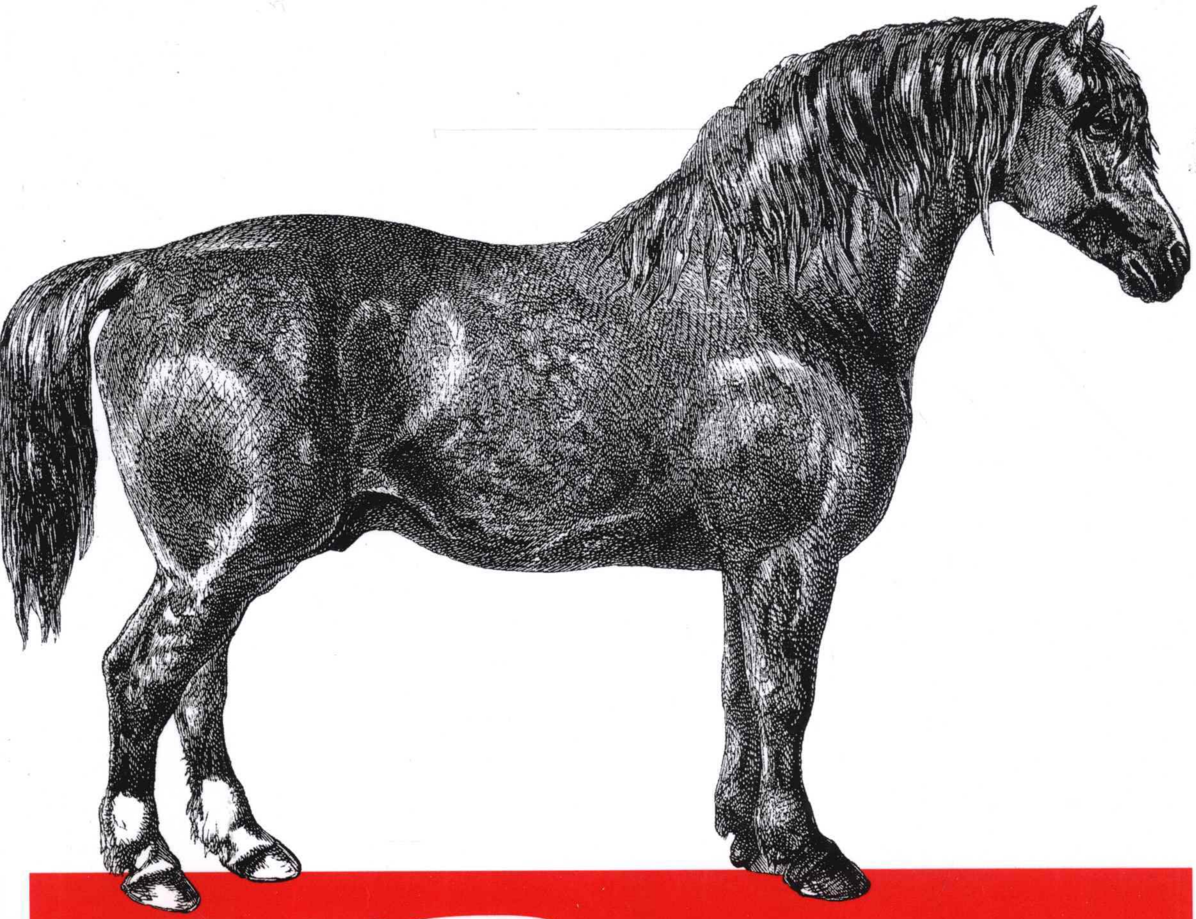
HBase权威指南（影印版）



HBase

The Definitive Guide

O'REILLY®

東南大學出版社

Lars George 著

HBase权威指南（影印版）

HBase: The Definitive Guide

Lars George

O'REILLY®

Beijing · Cambridge · Farnham · Köln · Sebastopol · Tokyo

O'Reilly Media, Inc.授权东南大学出版社出版

东南大学出版社

图书在版编目（CIP）数据

HBase权威指南：英文/（比）乔治（George, L.），著．—影印本．—南京：东南大学出版社，2012.4

书名原文：HBase: The Definitive Guide

ISBN 978-7-5641-3392-4

I. ① H…　II. ①乔…　III. ①计算机网络－信息存贮－指南－英文　IV. ① TP393-62

中国版本图书馆CIP数据核字（2012）第050017号

江苏省版权局著作权合同登记
图字：10-2011-420号

HBase权威指南（影印版）

出版发行：	东南大学出版社
地　　址：	南京四牌楼2号　　邮编：210096
出 版 人：	江建中
网　　址：	http://www.seupress.com
电子邮件：	press@seupress.com
印　　刷：	扬中市印刷有限公司
开　　本：	787毫米×980毫米　16开本
印　　张：	34.5
字　　数：	676千字
版　　次：	2012年4月第1版
印　　次：	2012年4月第1次印刷
书　　号：	ISBN 978-7-5641-3392-4
定　　价：	72.00元（册）

本社图书若有印装质量问题，请直接与营销部联系。电话（传真）：025-83791830

HBase权威指南（影印版）

HBase: The Definitive Guide

For my wife Katja, my daughter Laura, and son Leon. I love you!

Foreword

The HBase story begins in 2006, when the San Francisco-based startup Powerset was trying to build a natural language search engine for the Web. Their indexing pipeline was an involved multistep process that produced an index about two orders of magnitude larger, on average, than your standard term-based index. The datastore that they'd built on top of the then nascent Amazon Web Services to hold the index intermediaries and the webcrawl was buckling under the load (Ring. Ring. "Hello! This is AWS. Whatever you are running, please turn it off!"). They were looking for an alternative. The Google BigTable paper[*] had just been published.

Chad Walters, Powerset's head of engineering at the time, reflects back on the experience as follows:

> Building an open source system to run on top of Hadoop's Distributed Filesystem (HDFS) in much the same way that BigTable ran on top of the Google File System seemed like a good approach because: 1) it was a proven scalable architecture; 2) we could leverage existing work on Hadoop's HDFS; and 3) we could both contribute to and get additional leverage from the growing Hadoop ecosystem.

After the publication of the Google BigTable paper, there were on-again, off-again discussions around what a BigTable-like system on top of Hadoop might look. Then, in early 2007, out of the blue, Mike Cafarela dropped a tarball of thirty odd Java files into the Hadoop issue tracker: "I've written some code for HBase, a BigTable-like file store. It's not perfect, but it's ready for other people to play with and examine." Mike had been working with Doug Cutting on Nutch, an open source search engine. He'd done similar drive-by code dumps there to add features such as a Google File System clone so the Nutch indexing process was not bounded by the amount of disk you attach to a single machine. (This Nutch distributed filesystem would later grow up to be HDFS.)

Jim Kellerman of Powerset took Mike's dump and started filling in the gaps, adding tests and getting it into shape so that it could be committed as part of Hadoop. The first commit of the HBase code was made by Doug Cutting on April 3, 2007, under

[*] "BigTable: A Distributed Storage System for Structured Data" (*http://labs.google.com/papers/bigtable.html*) by Fay Chang et al.

the *contrib* subdirectory. The first HBase "working" release was bundled as part of Hadoop 0.15.0 in October 2007.

Not long after, Lars, the author of the book you are now reading, showed up on the #hbase IRC channel. He had a big-data problem of his own, and was game to try HBase. After some back and forth, Lars became one of the first users to run HBase in production outside of the Powerset home base. Through many ups and downs, Lars stuck around. I distinctly remember a directory listing Lars made for me a while back on his production cluster at WorldLingo, where he was employed as CTO, sysadmin, and grunt. The listing showed ten or so HBase releases from Hadoop 0.15.1 (November 2007) on up through HBase 0.20, each of which he'd run on his 40-node cluster at one time or another during production.

Of all those who have contributed to HBase over the years, it is poetic justice that Lars is the one to write this book. Lars was always dogging HBase contributors that the documentation needed to be better if we hoped to gain broader adoption. Everyone agreed, nodded their heads in ascent, amen'd, and went back to coding. So Lars started writing critical how-tos and architectural descriptions inbetween jobs and his intra-European travels as unofficial HBase European ambassador. His Lineland blogs on HBase (*http://www.larsgeorge.com*) gave the best description, outside of the source, of how HBase worked, and at a few critical junctures, carried the community across awkward transitions (e.g., an important blog explained the labyrinthian HBase build during the brief period we thought an Ivy-based build to be a "good idea"). His luscious diagrams were poached by one and all wherever an HBase presentation was given.

HBase has seen some interesting times, including a period of sponsorship by Microsoft, of all things. Powerset was acquired in July 2008, and after a couple of months during which Powerset employees were disallowed from contributing while Microsoft's legal department vetted the HBase codebase to see if it impinged on SQLServer patents, we were allowed to resume contributing (I was a Microsoft employee working near full time on an Apache open source project). The times ahead look promising, too, whether it's the variety of contortions HBase is being put through at Facebook—as the underpinnings for their massive Facebook mail app or fielding millions of of hits a second on their analytics clusters—or more deploys along the lines of Yahoo!'s 1k node HBase cluster used to host their snapshot of Microsoft's Bing crawl. Other developments include HBase running on filesystems other than Apache HDFS, such as MapR.

But plain to me though is that none of these developments would have been possible were it not for the hard work put in by our awesome HBase community driven by a core of HBase committers. Some members of the core have only been around a year or so—Todd Lipcon, Gary Helmling, and Nicolas Spiegelberg—and we would be lost without them, but a good portion have been there from close to project inception and have shaped HBase into the (scalable) general datastore that it is today. These include Jonathan Gray, who gambled his startup streamy.com on HBase; Andrew Purtell, who built an HBase team at Trend Micro long before such a thing was fashionable; Ryan Rawson, who got StumbleUpon—which became the main sponsor after HBase moved on from Powerset/Microsoft—on board, and who had the sense to hire John-Daniel Cryans, now a power contributor but just a bushy-tailed student at the time. And then there is Lars, who during the bug fixes, was always about documenting how it all worked. Of those of us who know HBase, there is no better man qualified to write this first, critical HBase book.

—Michael Stack, HBase Project Janitor

Preface

You may be reading this book for many reasons. It could be because you heard all about *Hadoop* and what it can do to crunch petabytes of data in a reasonable amount of time. While reading into Hadoop you found that, for random access to the accumulated data, there is something called *HBase*. Or it was the hype that is prevalent these days addressing a new kind of data storage architecture. It strives to solve large-scale data problems where traditional solutions may be either too involved or cost-prohibitive. A common term used in this area is *NoSQL*.

No matter how you have arrived here, I presume you want to know and learn—like I did not too long ago—how you can use HBase in your company or organization to store a virtually endless amount of data. You may have a background in relational database theory or you want to start fresh and this "column-oriented thing" is something that seems to fit your bill. You also heard that HBase can scale without much effort, and that alone is reason enough to look at it since you are building the next web-scale system.

I was at that point in late 2007 when I was facing the task of storing millions of documents in a system that needed to be fault-tolerant and scalable while still being maintainable by just me. I had decent skills in managing a MySQL database system, and was using the database to store data that would ultimately be served to our website users. This database was running on a single server, with another as a backup. The issue was that it would not be able to hold the amount of data I needed to store for this new project. I would have to either invest in serious RDBMS scalability skills, or find something else instead.

Obviously, I took the latter route, and since my mantra always was (and still is) "How does someone like Google do it?" I came across Hadoop. After a few attempts to use Hadoop directly, I was faced with implementing a random access layer on top of it—but that problem had been solved already: in 2006, Google had published a paper titled "Bigtable"* and the Hadoop developers had an open source implementation of it called HBase (the *H*adoop Data*base*). That was the answer to all my problems. Or so it seemed...

* See *http://labs.google.com/papers/bigtable-osdi06.pdf* for reference.

These days, I try not to think about how difficult my first experience with Hadoop and HBase was. Looking back, I realize that I would have wished for this customer project to start today. HBase is now mature, nearing a 1.0 release, and is used by many high-profile companies, such as Facebook, Adobe, Twitter, Yahoo!, Trend Micro, and StumbleUpon (as per *http://wiki.apache.org/hadoop/Hbase/PoweredBy*). Mine was one of the very first clusters in production (and is still in use today!) and my use case triggered a few very interesting issues (let me refrain from saying more).

But that was to be expected, betting on a 0.1x version of a community project. And I had the opportunity over the years to contribute back and stay close to the development team so that eventually I was humbled by being asked to become a full-time committer as well.

I learned a lot over the past few years from my fellow HBase developers and am still learning more every day. My belief is that we are nowhere near the peak of this technology and it will evolve further over the years to come. Let me pay my respect to the entire HBase community with this book, which strives to cover not just the internal workings of HBase or how to get it going, but more specifically, how to apply it to your use case.

In fact, I strongly assume that this is why you are here right now. You want to learn how HBase can solve *your* problem. Let me help you try to figure this out.

General Information

Before we get started, here a few general notes.

HBase Version

While writing this book, I decided to cover what will eventually be released as 0.92.0, and what is currently developed in the trunk of the official repository (*http://svn.apache.org/viewvc/hbase/trunk/*) under the early access release 0.91.0-SNAPSHOT.

Since it was not possible to follow the frantic development pace of HBase, and because the book had a deadline before 0.92.0 was released, the book could not document anything after a specific revision: 1130916 (*http://svn.apache.org/viewvc/hbase/trunk/?pathrev=1130916*). When you find that something does not seem correct between what is written here and what HBase offers, you can use the aforementioned revision number to compare all changes that have been applied *after* this book went into print.

I have made every effort to update the *JDiff* (a tool to compare different revisions of a software project) documentation on the book's website at *http://www.hbasebook.com*. You can use it to quickly see what is different.

Building the Examples

The examples you will see throughout this book can be found in full detail in the publicly available GitHub repository at *http://github.com/larsgeorge/hbase-book*. For the sake of brevity, they are usually printed only in parts here, to focus on the important bits, and to avoid repeating the same boilerplate code over and over again.

The name of an example matches the filename in the repository, so it should be easy to find your way through. Each chapter has its own subdirectory to make the separation more intuitive. If you are reading, for instance, an example in Chapter 3, you can go to the matching directory in the source repository and find the full source code there.

Many examples use internal helpers for convenience, such as the `HBaseHelper` class, to set up a test environment for reproducible results. You can modify the code to create different scenarios, or introduce faulty data and see how the feature showcased in the example behaves. Consider the code a petri dish for your own experiments.

Building the code requires a few auxiliary command-line tools:

Java
: HBase is written in Java, so you do need to have Java set up for it to work. "Java" on page 46 has the details on how this affects the installation. For the examples, you also need Java on the workstation you are using to run them.

Git
: The repository is hosted by GitHub, an online service that supports *Git*—a distributed revision control system, created originally for the Linux kernel development.† There are many binary packages that can be used on all major operating systems to install the Git command-line tools required.

: Alternatively, you can download a static snapshot of the entire archive using the GitHub download (*https://github.com/larsgeorge/hbase-book/archives/master*) link.

Maven
: The build system for the book's repository is Apache Maven.‡ It uses the so-called *Project Object Model* (POM) to describe what is needed to build a software project. You can download Maven from its website and also find installation instructions there.

Once you have gathered the basic tools required for the example code, you can build the project like so:

```
~$ cd /tmp
/tmp$ git clone git://github.com/larsgeorge/hbase-book.git
Initialized empty Git repository in /tmp/hbase-book/.git/
remote: Counting objects: 420, done.
```

† See the project's website (*http://git-scm.com/*) for details.

‡ See the project's website (*http://maven.apache.org/*) for details.

```
remote: Compressing objects: 100% (252/252), done.
remote: Total 420 (delta 159), reused 144 (delta 58)
Receiving objects: 100% (420/420), 70.87 KiB, done.
Resolving deltas: 100% (159/159), done.
/tmp$ cd hbase-book/
/tmp/hbase-book$ mvn package
[INFO] Scanning for projects...
[INFO] Reactor build order:
[INFO]   HBase Book
[INFO]   HBase Book Chapter 3
[INFO]   HBase Book Chapter 4
[INFO]   HBase Book Chapter 5
[INFO]   HBase Book Chapter 6
[INFO]   HBase Book Chapter 11
[INFO]   HBase URL Shortener
[INFO] ------------------------------------------------------------------------
[INFO] Building HBase Book
[INFO]    task-segment: [package]
[INFO] ------------------------------------------------------------------------
[INFO] [site:attach-descriptor {execution: default-attach-descriptor}]
[INFO] ------------------------------------------------------------------------
[INFO] Building HBase Book Chapter 3
[INFO]    task-segment: [package]
[INFO] ------------------------------------------------------------------------
[INFO] [resources:resources {execution: default-resources}]
...
[INFO] ------------------------------------------------------------------------
[INFO] Reactor Summary:
[INFO] ------------------------------------------------------------------------
[INFO] HBase Book ............................................ SUCCESS [1.601s]
[INFO] HBase Book Chapter 3 .................................. SUCCESS [3.233s]
[INFO] HBase Book Chapter 4 .................................. SUCCESS [0.589s]
[INFO] HBase Book Chapter 5 .................................. SUCCESS [0.162s]
[INFO] HBase Book Chapter 6 .................................. SUCCESS [1.354s]
[INFO] HBase Book Chapter 11 ................................. SUCCESS [0.271s]
[INFO] HBase URL Shortener ................................... SUCCESS [4.910s]
[INFO] ------------------------------------------------------------------------
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESSFUL
[INFO] ------------------------------------------------------------------------
[INFO] Total time: 12 seconds
[INFO] Finished at: Mon Jun 20 17:08:30 CEST 2011
[INFO] Final Memory: 35M/81M
[INFO] ------------------------------------------------------------------------
```

This *clones*—which means it is downloading the repository to your local workstation—the source code and subsequently *compiles* it. You are left with a *Java archive file* (also called a *JAR* file) in the *target* directory in each of the subdirectories, that is, one for each chapter of the book that has source code examples:

```
/tmp/hbase-book$ ls -l ch04/target/
total 152
drwxr-xr-x  48 larsgeorge  wheel   1632 Apr 15 10:31 classes
drwxr-xr-x   3 larsgeorge  wheel    102 Apr 15 10:31 generated-sources
```

```
-rw-r--r--   1 larsgeorge  wheel  75754 Apr 15 10:31 hbase-book-ch04-1.0.jar
drwxr-xr-x   3 larsgeorge  wheel    102 Apr 15 10:31 maven-archiver
```

In this case, the *hbase-book-ch04-1.0.jar* file contains the compiled examples for Chapter 4. Assuming you have a running installation of HBase, you can then run each of the included classes using the supplied command-line script:

```
/tmp/hbase-book$ cd ch04/
/tmp/hbase-book/ch04$ bin/run.sh client.PutExample
/tmp/hbase-book/ch04$ bin/run.sh client.GetExample
Value: val1
```

The supplied *bin/run.sh* helps to assemble the required Java *classpath*, adding the dependent JAR files to it.

Hush: The HBase URL Shortener

Looking at each feature HBase offers separately is a good way to understand what it does. The book uses code examples that set up a very specific set of tables, which contain an equally specific set of data. This makes it easy to understand what is given and how a certain operation changes the data from the before to the after state. You can execute every example yourself to replicate the outcome, and it should match exactly with what is described in the accompanying book section. You can also modify the examples to explore the discussed feature even further—and you can use the supplied helper classes to create your own set of proof-of-concept examples.

Yet, sometimes it is important to see all the features working in concert to make the final leap of understanding their full potential. For this, the book uses a single, real-world example to showcase most of the features HBase has to offer. The book also uses the example to explain advanced concepts that come with this different storage territory—compared to more traditional RDBMS-based systems.

The fully working application is called *Hush*—short for *HBase URL Shortener*. Many services on the Internet offer this kind of service. Simply put, you hand in a URL—for example, for a web page—and you get a much shorter link back. This link can then be used in places where real estate is at a premium: Twitter only allows you to send messages with a maximum length of 140 characters. URLs can be up to 4,096 bytes long; hence there is a need to reduce that length to something around 20 bytes instead, leaving you more space for the actual message.

For example, here is the Google Maps URL used to reference Sebastopol, California:

```
http://maps.google.com/maps?f=q&source=s_q&hl=en&geocode=&q=Sebastopol, \
+CA,+United+States&aq=0&sll=47.85931,10.85165&sspn=0.93616,1.345825&ie=UTF8& \
hq=&hnear=Sebastopol,+Sonoma,+California&z=14
```

Running this through a URL shortener like Hush results in the following URL:

```
http://hush.li/1337
```

Obviously, this is much shorter, and easier to copy into an email or send through a restricted medium, like Twitter or SMS.

But this service is not simply a large lookup table. Granted, popular services in this area have hundreds of millions of entries mapping short to long URLs. But there is more to it. Users want to shorten specific URLs and also track their usage: how often has a short URL been used? A shortener service should retain counters for every shortened URL to report how often they have been clicked.

More advanced features are vanity URLs that can use specific domain names, and/or custom short URL IDs, as opposed to auto-generated ones, as in the preceding example. Users must be able to log in to create their own short URLs, track their existing ones, and see reports for the daily, weekly, or monthly usage.

All of this is realized in Hush, and you can easily compile and run it on your own server. It uses a wide variety of HBase features, and it is mentioned, where appropriate, throughout this book, showing how a newly discussed topic is used in a production-type application.

While you could create your own user account and get started with Hush, it is also a great example of how to import legacy data from, for example, a previous system. To emulate this use case, the book makes use of a freely available data set on the Internet: the Delicious RSS feed. There are a few sets that were made available by individuals, and can be downloaded by anyone.

Use Case: Hush

Be on the lookout for boxes like this throughout the book. Whenever possible, such boxes support the explained features with examples from Hush. Many will also include example code, but often such code is kept very simple to showcase the feature at hand. The data is also set up so that you can repeatedly make sense of the functionality (even though the examples may be a bit academic). Using Hush as a use case more closely mimics what you would implement in a production system.

Hush is actually built to scale out of the box. It might not have the prettiest interface, but that is not what it should prove. You can run many Hush servers behind a load balancer and serve thousands of requests with no difficulties.

The snippets extracted from Hush show you how the feature is used in context, and since it is part of the publicly available repository accompanying the book, you have the full source available as well. Run it yourself, tweak it, and learn all about it!

Running Hush

Building and running Hush is as easy as building the example code. Once you have cloned—or downloaded—the book repository, and executed

```
$ mvn package
```

to build the entire project, you can start Hush with the included start script:

```
$ hush/bin/start-hush.sh
=====================
 Starting Hush...
=====================
 INFO [main] (HushMain.java:57) - Initializing HBase
 INFO [main] (HushMain.java:60) - Creating/updating HBase schema
 ...
 INFO [main] (HushMain.java:90) - Web server setup.
 INFO [main] (HushMain.java:111) - Configuring security.
 INFO [main] (Slf4jLog.java:55) - jetty-7.3.1.v20110307
 INFO [main] (Slf4jLog.java:55) - started ...
 INFO [main] (Slf4jLog.java:55) - Started SelectChannelConnector@0.0.0.0:8080
```

After the last log message is output on the console, you can navigate your browser to *http://localhost:8080* to access your local Hush server.

Stopping the server requires a Ctrl-C to abort the start script. As all data is saved on the HBase cluster accessed remotely by Hush, this is safe to do.

Conventions Used in This Book

The following typographical conventions are used in this book:

Italic
: Indicates new terms, URLs, email addresses, filenames, file extensions, and Unix commands

`Constant width`
: Used for program listings, as well as within paragraphs to refer to program elements such as variable or function names, databases, data types, environment variables, statements, and keywords

`Constant width bold`
: Shows commands or other text that should be typed literally by the user

`Constant width italic`
: Shows text that should be replaced with user-supplied values or by values determined by context

This icon signifies a tip, suggestion, or general note.

This icon indicates a warning or caution.

Using Code Examples

This book is here to help you get your job done. In general, you may use the code in this book in your programs and documentation. You do not need to contact us for permission unless you're reproducing a significant portion of the code. For example, writing a program that uses several chunks of code from this book does not require permission. Selling or distributing a CD-ROM of examples from O'Reilly books does require permission. Answering a question by citing this book and quoting example code does not require permission. Incorporating a significant amount of example code from this book into your product's documentation does require permission.

We appreciate, but do not require, attribution. An attribution usually includes the title, author, publisher, and ISBN. For example: "*HBase: The Definitive Guide* by Lars George (O'Reilly). Copyright 2011 Lars George, 978-1-449-39610-7."

If you feel your use of code examples falls outside fair use or the permission given here, feel free to contact us at *permissions@oreilly.com*.

Safari® Books Online

Safari Books Online is an on-demand digital library that lets you easily search over 7,500 technology and creative reference books and videos to find the answers you need quickly.

With a subscription, you can read any page and watch any video from our library online. Read books on your cell phone and mobile devices. Access new titles before they are available for print, and get exclusive access to manuscripts in development and post feedback for the authors. Copy and paste code samples, organize your favorites, download chapters, bookmark key sections, create notes, print out pages, and benefit from tons of other time-saving features.

O'Reilly Media has uploaded this book to the Safari Books Online service. To have full digital access to this book and others on similar topics from O'Reilly and other publishers, sign up for free at *http://my.safaribooksonline.com*.

How to Contact Us

Please address comments and questions concerning this book to the publisher:

O'Reilly Media, Inc.
1005 Gravenstein Highway North
Sebastopol, CA 95472
800-998-9938 (in the United States or Canada)
707-829-0515 (international or local)
707-829-0104 (fax)

We have a web page for this book, where we list errata, examples, and any additional information. You can access this page at:

http://www.oreilly.com/catalog/9781449396107

The author also has a site for this book at:

http://www.hbasebook.com/

To comment or ask technical questions about this book, send email to:

bookquestions@oreilly.com

For more information about our books, courses, conferences, and news, see our website at *http://www.oreilly.com*.

Find us on Facebook: *http://facebook.com/oreilly*

Follow us on Twitter: *http://twitter.com/oreillymedia*

Watch us on YouTube: *http://www.youtube.com/oreillymedia*

Acknowledgments

I first want to thank my late dad, Reiner, and my mother, Ingrid, who supported me and my aspirations all my life. You were the ones to make me a better person.

Writing this book was only possible with the support of the entire HBase community. Without that support, there would be no HBase, nor would it be as successful as it is today in production at companies all around the world. The relentless and seemingly tireless support given by the core committers as well as contributors and the community at large on IRC, the Mailing List, and in blog posts is the essence of what open source stands for. I stand tall on your shoulders!

Thank you to the committers, who included, as of this writing, Jean-Daniel Cryans, Jonathan Gray, Gary Helmling, Todd Lipcon, Andrew Purtell, Ryan Rawson, Nicolas Spiegelberg, Michael Stack, and Ted Yu; and to the emeriti, Mike Cafarella, Bryan Duxbury, and Jim Kellerman.