O'REILLY®

第2版

MySQL High Availability

高可用性MySQL（影印版）

Charles Bell, Mats Kindahl,
Lars Thalmann 著
Pinterest technologists 序

東南大學出版社

第2版

高可用性MySQL（影印版）

MySQL High Availability

Charles Bell, Mats Kindahl,
Lars Thalmann 著

Beijing · Cambridge · Farnham · Köln · Sebastopol · Tokyo O'REILLY®

O'Reilly Media, Inc.授权东南大学出版社出版

南京 东南大学出版社

图书在版编目(CIP)数据

高可用性 MySQL：第 2 版：英文/(美)贝尔(Bell, C.),(美)肯德尔(Kindahl, M.), (瑞典)塞尔曼(Thalmann, L.)著. —影印本. —南京：东南大学出版社，2015.2

书名原文：MySQL High Availability, 2E

ISBN 978-7-5641-5386-1

Ⅰ.①高… Ⅱ.①贝… ②肯… ③塞… Ⅲ.①关系数据库系统—英文 Ⅳ.①TP311.138

中国版本图书馆 CIP 数据核字(2014)第 294383 号

江苏省版权局著作权合同登记
图字：10-2013-376 号

高可用性 MySQL　第 2 版(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：常州市武进第三印刷有限公司
开　　本：787 毫米×980 毫米　16 开本
印　　张：47.75
字　　数：935 千字
版　　次：2015 年 2 月第 1 版
印　　次：2015 年 2 月第 1 次印刷
书　　号：ISBN 978-7-5641-5386-1
定　　价：98.00 元

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

Foreword for the Second Edition

In 2011, Pinterest started growing. Some say we grew faster than any other startup to date. In the earliest days, we were up against a new scalability bottleneck every day that could slow down the site or bring it down altogether. We remember having our laptops with us everywhere. We slept with them, we ate with them, we went on vacation with them. We even named them. We have the sound of the SMS outage alerts imprinted in our brains.

When the infrastructure is constantly being pushed to its limits, you can't help but wish for an easy way out. During our growth, we tried no less than five well-known database technologies that claimed to solve all our problems, but each failed catastrophically. Except MySQL. The time came around September 2011 to throw all the cards in the air and let them resettle. We re-architected everything around MySQL, Memcache, and Redis with just three engineers.

MySQL? Why MySQL? We laid out our biggest concerns with any technology and started asking the same questions for each. Here's how MySQL shaped up:

- Does it address our storage needs? Yes, we needed mappings, indexes, sorting, and blob storage, all available in MySQL.
- Is it commonly used? Can you hire somebody for it? MySQL is one of the most common database choices in production today. It's so easy to hire people who have used MySQL that we could walk outside in Palo Alto and yell out for a MySQL engineer and a few would come up. Not kidding.
- Is the community active? Very active. There are great books available and a strong online community.
- How robust is it to failure? Very robust! We've never lost any data even in the most dire of situations.
- How well does it scale? By itself, it does not scale beyond a single box. We'd need a sharding solution layered on top. (That's a whole other discussion!)

- Will you be the biggest user? Nope, not by far. Bigger users included Facebook, Twitter, and Google. You don't want to be the biggest user of a technology if you can help it. If you are, you'll trip over new scalability problems that nobody has had a chance to debug yet.
- How mature is it? Maturity became the real differentiator. Maturity to us is a measure of the blood, sweat, and tears that have gone into a program divided by its complexity. MySQL is reasonably complex, but not nearly so compared to some of the magic autoclustering NoSQL solutions available. Additionally, MySQL has had 28 years of the best and the brightest contributing back to it from such companies as Facebook and Google, who use it at *massive* scale. Of all the technologies we looked at, by our definition of maturity, MySQL was a clear choice.
- Does it have good debugging tools? As a product matures, you naturally get great debugging and profiling tools since people are more likely to have been in a similar sticky situation. You'll find yourself in trouble at 3 A.M. (multiple times). Being able to root cause an issue and get back to bed is better than rewriting for another technology by 6 A.M.

Based on our survey of 10 or so database technologies, MySQL was the clear choice. MySQL is great, but it kinda drops you off at your destination with no baggage and you have to fend for yourself. It works very well and you can connect to it, but as soon as you start using it and scaling, the questions starting flying:

- My query is running slow, now what?
- Should I enable compression? How do I do it?
- What are ways of scaling beyond one box?
- How do I get replication working? How about master-master replication?
- REPLICATION STOPPED! NOW WHAT?!
- What are options for durability (fsync speeds)?
- How big should my buffers be?
- There are a billion fields in *mysql.ini*. What are they? What should they be set to?
- I just accidentally wrote to my slave! How do I prevent that from happening again?
- How do I prevent running an `UPDATE` with no `where` clause?
- What debugging and profiling tools should I be using?
- Should I use InnoDB, MyISAM, or one of several other flavors of storage engine?

The online community is helpful for answering specific questions, finding examples, bug fixes, and workarounds, but often lacks a strong cohesive story, and deeper discussions about architecture are few and far between. We knew how to use MySQL at small scale, but this scale and pace were insane. *High Availability MySQL* provided insights that allowed us to squeeze more out of MySQL.

One new feature in MySQL 5.6, Global Transaction Handlers, adds a unique identifier to every transaction in a replication tree. This new feature makes failover and slave promotion far easier. We've been waiting for this for a long time and it's well covered in this new edition.

During our grand re-architecture to a sharded solution, we referred to this book for architectural decisions, such as replication techniques and topologies, data sharding alternatives, monitoring options, tuning, and concerns in the cloud. It gave us a deeper understanding of how MySQL works underneath the hood, which allowed us to make better informed choices around the high level queries, access patterns, and structures we'd be using, as well as iterate on our design afterward. The resulting MySQL architecture still serves Pinterest's core data needs today.

—Yashwanth Nelapati and Marty Weiner
Pinterest
February 2014

Foreword for the First Edition

A lot of research has been done on replication, but most of the resulting concepts are never put into production. In contrast, MySQL replication is widely deployed but has never been adequately explained. This book changes that. Things are explained here that were previously limited to people willing to read a lot of source code and spend a lot of time—including a few late-night sessions—debugging it in production.

Replication enables you to provide highly available data services while enduring the inevitable failures. There are an amazing number of ways for things to fail, including the loss of a disk, server, or data center. Even when hardware is perfect or fully redundant, people are not. Database tables will be dropped by mistake. Applications will write incorrect data. Occasional failure is assured. But with reasonable preparation, recovery from failure can also be assured. The keys to survival are redundancy and backups. Replication in MySQL supports both.

But MySQL replication is not limited to supporting failure recovery. It is frequently used to support read scale-out. MySQL can efficiently replicate to a large number of servers. For applications that are read-mostly, this is a cost-effective strategy for supporting a large number of queries on commodity hardware.

And there are other interesting uses for MySQL replication. Online data definition language (DDL) is a very complex feature to implement in a relational database management system. MySQL does not support online DDL, but through the use of replication, you can implement something that is frequently good enough. You can get a lot done with replication if you are willing to be creative.

Replication is one of the features that made MySQL wildly popular. It is also the feature that allows you to convert a popular MySQL prototype into a successful business-critical deployment. Like most of MySQL, replication favors simplicity and ease of use. As a consequence, it is occasionally less than perfect when running in production. This book explains what you need to know to successfully use MySQL replication. It will help you to understand how replication has been implemented, what can go wrong, how to prevent problems, and how to fix them when—despite your best attempts at prevention—they crop up.

MySQL replication is also a work in progress. Change, like failure, is also assured. MySQL is responding to that change, and replication continues to get more efficient, more robust, and more interesting. For instance, row-based replication is new in MySQL 5.1.

While MySQL deployments come in all shapes and sizes, I care most about data services for Internet applications and am excited about the potential to replicate from MySQL to distributed storage systems like HBase and Hadoop. This will make MySQL better at sharing the data center.

I have been on teams that support important MySQL deployments at Facebook and Google. I've encountered many of the problems covered in this book and have had the opportunity and time to learn solutions. The authors of this book are also experts on MySQL replication, and by reading this book you can share their expertise.

—Mark Callaghan

Preface

The authors of this book have been creating parts of MySQL and working with it for many years. Dr. Charles Bell is a senior developer leading the MySQL Utilities team. He has also worked on replication and backup. His interests include all things MySQL, database theory, software engineering, microcontrollers, and three-dimensional printing. Dr. Mats Kindahl is a principal senior software developer currently leading the MySQL High Availability and Scalability team. He is architect and implementor of several MySQL features. Dr. Lars Thalmann is the development director and technical lead of the MySQL Replication, Backup, Connectors, and Utilities teams, and has designed many of the replication and backup features. He has worked on the development of MySQL clustering, replication, and backup technologies.

We wrote this book to fill a gap we noticed among the many books on MySQL. There are many excellent books on MySQL, but few that concentrate on its advanced features and applications, such as high availability, reliability, and maintainability. In this book, you will find all of these topics and more.

We also wanted to make the reading a bit more interesting by including a running narrative about a MySQL professional who encounters common requests made by his boss. In the narrative, you will meet Joel Thomas, who recently decided to take a job working for a company that has just started using MySQL. You will observe Joel as he learns his way around MySQL and tackles some of the toughest problems facing MySQL professionals. We hope you find this aspect of the book entertaining.

Who This Book Is For

This book is for MySQL professionals. We expect you to have basic knowledge of SQL, MySQL administration, and the operating system you are running. We provide introductory information about replication, disaster recovery, system monitoring, and other key topics of high availability. See Chapter 1 for other books that offer useful background information.

How This Book Is Organized

This book is divided into two parts. Part I encompasses MySQL high availability and scale-out. Because these depend a great deal on replication, a lot of this part focuses on that topic. Part II examines monitoring and performance concerns for building robust data centers.

Part I, High Availability and Scalability

Chapter 1, *Introduction*, explains how this book can help you and gives you a context for reading it.

Chapter 2, *MySQL Replicant Library*, introduces a Python library for working with sets of servers that is used throughout the book.

Chapter 3, *MySQL Replication Fundamentals*, discusses both manual and automated procedures for setting up basic replication.

Chapter 4, *The Binary Log*, explains the critical file that ties together replication and helps in disaster recovery, troubleshooting, and other administrative tasks.

Chapter 5, *Replication for High Availability*, shows a number of ways to recover from server failure, including the use of automated scripts.

Chapter 6, *MySQL Replication for Scale-Out*, shows a number of techniques and topologies for improving the read scalabilility of large data sets.

Chapter 7, *Data Sharding*, shows techniques for handling very large databases and/or improving the write scalability of a database through sharding.

Chapter 8, *Replication Deep Dive*, addresses a number of topics, such as secure data transfer and row-based replication.

Chapter 9, *MySQL Cluster*, shows how to use this tool to achieve high availability.

Part II, Monitoring and Managing

Chapter 10, *Getting Started with Monitoring*, presents the main operating system parameters you have to be aware of, and tools for monitoring them.

Chapter 11, *Monitoring MySQL*, presents several tools for monitoring database activity and performance.

Chapter 12, *Storage Engine Monitoring*, explains some of the parameters you need to monitor on a more detailed level, focusing on issues specific to MyISAM or InnoDB.

Chapter 13, *Replication Monitoring*, offers details about how to keep track of what masters and slaves are doing.

Chapter 14, *Replication Troubleshooting*, shows how to deal with failures and restarts, corruption, and other incidents.

Chapter 15, *Protecting Your Investment*, explains the use of backups and disaster recovery techniques.

Chapter 16, *MySQL Enterprise Monitor*, introduces a suite of tools that simplifies many of the tasks presented in earlier chapters.

Chapter 17, *Managing MySQL Replication with MySQL Utilities*, introduces the MySQL Utilities, which are a new set of tools for managing MySQL Servers.

Appendixes

Appendix A, *Replication Tips and Tricks*, offers a grab bag of procedures that are useful in certain situations.

Appendix B, *A GTID Implementation*, shows an implementation for handling failovers with transactions if you are using servers that don't support GTIDs.

Conventions Used in This Book

The following typographical conventions are used in this book:

Plain text
: Indicates menu titles, table names, options, and buttons.

Italic
: Indicates new terms, database names, URLs, email addresses, filenames, and Unix utilities.

`Constant width`
: Indicates command-line options, variables and other code elements, the contents of files, and the output from commands.

`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.

This element signifies a tip or suggestion.

This element signifies a general note.

This element indicates a warning or caution.

Using Code Examples

Supplemental material (code examples, exercises, etc.) is available for download at at http://bit.ly/mysqllaunch.

This book is here to help you get your job done. In general, if example code is offered with this book, you may use it 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: "*MySQL High Availability*, by Charles Bell, Mats Kindahl, and Lars Thalmann. Copyright 2014 Charles Bell, Mats Kindahl, and Lars Thalmann, 978-1-44933-958-6."

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

Safari® Books Online

Safari Books Online (*www.safaribooksonline.com*) is an on-demand digital library that delivers expert content in both book and video form from the world's leading authors in technology and business.

Technology professionals, software developers, web designers, and business and creative professionals use Safari Books Online as their primary resource for research, problem solving, learning, and certification training.

Safari Books Online offers a range of product mixes and pricing programs for organizations, government agencies, and individuals. Subscribers have access to thousands of books, training videos, and prepublication manuscripts in one fully searchable database from publishers like O'Reilly Media, Prentice Hall Professional, Addison-Wesley Professional, Microsoft Press, Sams, Que, Peachpit Press, Focal Press, Cisco Press, John Wiley & Sons, Syngress, Morgan Kaufmann, IBM Redbooks, Packt, Adobe Press, FT Press, Apress, Manning, New Riders, McGraw-Hill, Jones & Bartlett, Course Technology, and dozens more. For more information about Safari Books Online, please visit us online.

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://bit.ly/mysql_high_availability*.

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*.

Acknowledgments

The authors would like to thank our technical reviewers of this and the previous edition: Mark Callahan, Morgan Tocker, Sveta Smirnova, Luis Soares, Sheeri Kritzer Cabral, Alfie John, and Colin Charles. Your attention to detail and insightful suggestions were invaluable. We could not have delivered a quality book without your help.

We also want to thank our extremely talented colleagues on the MySQL team and in the MySQL community who have provided comments, including Alfranio Correia, Andrei Elkin, Zhen-Xing He, Serge Kozlov, Sven Sandberg, Luis Soares, Rafal Somla, Li-Bing Song, Ingo Strüwing, Dao-Gang Qu, Giuseppe Maxia, and Narayanan Venkateswaran for their tireless dedication to making MySQL the robust and powerful tool it is today. We especially would like to thank our MySQL customer support professionals, who help us bridge the gap between our customers' needs and our own desires to improve the product. We would also like to thank the many community members who so selflessly devote time and effort to improve MySQL for everyone.

Finally, and most important, we would like to thank our editor, Andy Oram, who helped us shape this work, for putting up with our sometimes cerebral and sometimes over-the-top enthusiasm for all things MySQL. A most sincere thanks goes out to the entire O'Reilly team and especially our editor for their patience as we struggled to fit so many new topics into what was already a very large book.

Charles would like to thank his loving wife, Annette, for her patience and understanding when he was spending time away from family priorities to work on this book. Charles would also like to thank his many colleagues on the MySQL team at Oracle who contribute their wisdom freely to everyone on a daily basis. Finally, Charles would like to thank all of his brothers and sisters in Christ who both challenge and support him daily.

Mats would like to thank his wife, Lill, and two sons, Jon and Hannes, for their unconditional love and understanding in difficult times. You are the loves of his life and he cannot imagine a life without you. Mats would also like to thank his MySQL colleagues inside and outside Oracle for all the interesting, amusing, and inspiring times together —you are truly some of the sharpest minds in the trade.

Lars would like to thank his amazing girlfriend Claudia; he loves her beyond words. He would also like to thank all of his colleagues, current and past, who have made MySQL such an interesting place to work. In fact, it is not even a place. The distributed nature of the MySQL development team and the open-mindedness of its many dedicated developers are truly extraordinary. The MySQL community has a special spirit that makes working with MySQL an honorable task. What we have created together is remarkable. It is amazing that it started with such a small group of people and managed to build a product that services so many of the Fortune 500 companies today.

Table of Contents

Part I. High Availability and Scalability