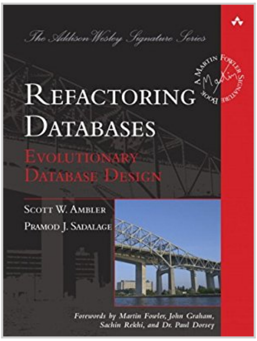
The Addison Wesley Signature Series
REFACTORING DATABASES
EVOLUTIONARY DATABASE DESIGN
SCOTT W. AMBLER
PRAMOD J. SADALAGE
A Martin Fowler Signature Book
Forewords by Martin Fowler, John Graham, Sachin Rekhi, and Dr. Paul Dorsey

Refactoring Databases: Evolutionary Database Design

By Scott W. Ambler, Pramodkumar J. Sadalage

Refactoring Databases: Evolutionary Database Design By Scott W. Ambler, Pramodkumar J. Sadalage

Refactoring has proven its value in a wide range of development projects—helping software professionals improve system designs, maintainability, extensibility, and performance. Now, for the first time, leading agile methodologist Scott Ambler and renowned consultant Pramodkumar Sadalage introduce powerful refactoring techniques specifically designed for database systems.

Ambler and Sadalage demonstrate how small changes to table structures, data, stored procedures, and triggers can significantly enhance virtually any database design—without changing semantics. You’ll learn how to evolve database schemas in step with source code—and become far more effective in projects relying on iterative, agile methodologies.

This comprehensive guide and reference helps you overcome the practical obstacles to refactoring real-world databases by covering every fundamental concept underlying database refactoring. Using start-to-finish examples, the authors walk you through refactoring simple standalone database applications as well as sophisticated multi-application scenarios. You’ll master every task involved in refactoring database schemas, and discover best practices for deploying refactorings in even the most complex production environments.

The second half of this book systematically covers five major categories of database refactorings. You’ll learn how to use refactoring to enhance database structure, data quality, and referential integrity; and how to refactor both architectures and methods. This book provides an extensive set of examples built with Oracle and Java and easily adaptable for other languages, such as C#, C++, or VB.NET, and other databases, such as DB2, SQL Server, MySQL, and Sybase.

Using this book’s techniques and examples, you can reduce waste, rework, risk, and cost—and build database systems capable of evolving smoothly, far into the future.

Refactoring Databases: Evolutionary Database Design

By Scott W. Ambler, Pramodkumar J. Sadalage

Refactoring Databases: Evolutionary Database Design By Scott W. Ambler, Pramodkumar J. Sadalage

Refactoring has proven its value in a wide range of development projects—helping software professionals improve system designs, maintainability, extensibility, and performance. Now, for the first time, leading agile methodologist Scott Ambler and renowned consultant Pramodkumar Sadalage introduce powerful refactoring techniques specifically designed for database systems.
Ambler and Sadalage demonstrate how small changes to table structures, data, stored procedures, and triggers can significantly enhance virtually any database design—without changing semantics. You'll learn how to evolve database schemas in step with source code—and become far more effective in projects relying on iterative, agile methodologies.
This comprehensive guide and reference helps you overcome the practical obstacles to refactoring real-world databases by covering every fundamental concept underlying database refactoring. Using start-to-finish examples, the authors walk you through refactoring simple standalone database applications as well as sophisticated multi-application scenarios. You'll master every task involved in refactoring database schemas, and discover best practices for deploying refactorings in even the most complex production environments.
The second half of this book systematically covers five major categories of database refactorings. You'll learn how to use refactoring to enhance database structure, data quality, and referential integrity; and how to refactor both architectures and methods. This book provides an extensive set of examples built with Oracle and Java and easily adaptable for other languages, such as C#, C++, or VB.NET, and other databases, such as DB2, SQL Server, MySQL, and Sybase.
Using this book's techniques and examples, you can reduce waste, rework, risk, and cost—and build database systems capable of evolving smoothly, far into the future.

Refactoring Databases: Evolutionary Database Design By Scott W. Ambler, Pramodkumar J. Sadalage Bibliography

- Rank: #278806 in Books
- Published on: 2006-03-13
- Ingredients: Example Ingredients
- Original language: English
- Number of items: 1
- Dimensions: 9.50" h x 1.17" w x 7.25" l, 1.90 pounds
- Binding: Hardcover
- 384 pages

Refactoring Databases

Evolutionary Database Design

Preface

Evolutionary, and often agile, software development methodologies, such as Extreme Programming (XP), Scrum, the Rational Unified Process (RUP), the Agile Unified Process (AUP), and Feature-Driven Development (FDD), have taken the information technology (IT) industry by storm over the past few years. For the sake of definition, an evolutionary method is one that is both iterative and incremental in nature, and an agile method is evolutionary and highly collaborative in nature. Furthermore, agile techniques such as refactoring, pair programming, Test-Driven Development (TDD), and Agile Model-Driven Development (AMDD) are also making headway into IT organizations. These methods and techniques have been developed and have evolved in a grassroots manner over the years, being honed in the software trenches, as it were, instead of formulated in ivory towers. In short, this evolutionary and agile stuff seems to work incredibly well in practice.

In the seminal book *Refactoring*, Martin Fowler describes a refactoring as a small change to your source code that improves its design without changing its semantics. In other words, you improve the quality of your work without breaking or adding anything. In the book, Martin discusses the idea that just as it is possible to refactor your application source code, it is also possible to refactor your database schema. However, he states that database refactoring is quite hard because of the significant levels of coupling associated with databases, and therefore he chose to leave it out of his book.

Since 1999 when *Refactoring* was published, the two of us have found ways to refactor database schemas. Initially, we worked separately, running into each other at conferences such as Software Development (http://www.sdexpo.com) and on mailing lists (http://www.agiledata.org/feedback.html). We discussed ideas, attended each other's conference tutorials and presentations, and quickly discovered that our ideas and techniques overlapped and were highly compatible with one another. So we joined forces to write this book, to share our experiences and techniques at evolving database schemas via refactoring.

The examples throughout the book are written in Java, Hibernate, and Oracle code. Virtually every database refactoring description includes code to modify the database schema itself, and for some of the more interesting refactorings, we show the effects they would have on Java application code. Because all databases are not created alike, we include discussions of alternative implementation strategies when important nuances exist between database products. In some instances we discuss alternative implementations of an aspect of a refactoring using Oracle-specific features such as the SE,T UNUSED or RENAME TO commands, and many of our code examples take advantage of Oracle's COMMENT ON feature. Other database products include other features that make database refactoring easier, and a good DBA will know how to take advantage of these things. Better yet, in the future database refactoring tools will do this for us. Furthermore, we have kept the Java code simple enough so that you should be able to convert it to C#, C++, or even Visual Basic with little problem at all.

Why Evolutionary Database Development?

Evolutionary database development is a concept whose time has come. Instead of trying to design your database schema up front early in the project, you instead build it up throughout the life of a project to reflect the changing requirements defined by your stakeholders. Like it or not, requirements change as your project progresses. Traditional approaches have denied this fundamental reality and have tried to "manage change," a euphemism for preventing change, through various means. Practitioners of modern development techniques instead choose to embrace change and follow techniques that enable them to evolve their work in step with evolving requirements. Programmers have adopted techniques such as TDD, refactoring, and AMDD and have built new development tools to make this easy. As we have done this, we have realized that we also need techniques and tools to support evolutionary database development.

Advantages to an evolutionary approach to database development include the following:

1. **You minimize waste**. An evolutionary, just-in-time (JIT) approach enables you to avoid the inevitable wastage inherent in serial techniques when requirements change. Any early investment in detailed requirements, architecture, and design artifacts is lost when a requirement is later found to be no longer needed. If you have the skills to do the work up front, clearly you must have the skills to do the same work JIT.
2. **You avoid significant rework**. As you will see in Chapter 1, "Evolutionary Database Development," you should still do some initial modeling up front to think major issues through, issues that could potentially lead to significant rework if identified late in the project; you just do not need to investigate the details early.
3. **You always know that your system works**. With an evolutionary approach, you regularly produce working software, even if it is just deployed into a demo environment, which works. When you have a new, working version of the system every week or two, you dramatically reduce your project's risk.
4. **You always know that your database design is the highest quality possible**. This is exactly what database refactoring is all about: improving your schema design a little bit at a time.
5. **You work in a compatible manner with developers**. Developers work in an evolutionary manner, and if data professionals want to be effective members of modern development teams, they also need to choose to work in an evolutionary manner.
6. **You reduce the overall effort**. By working in an evolutionary manner, you only do the work that you actually need today and no more.

There are also several disadvantages to evolutionary database development:

1. **Cultural impediments exist**. Many data professionals prefer to follow a serial approach to software development, often insisting that some form of detailed logical and physical data models be created and baselined before programming begins. Modern methodologies have abandoned this approach as being too inefficient and risky, thereby leaving many data professionals in the cold. Worse yet, many of the "thought leaders" in the data community are people who cut their teeth in the 1970s and 1980s but who missed the object revolution of the 1990s, and thereby missed gaining experience in evolutionary development. The world changed, but they did not seem to change with it. As you will learn in this book, it is not only possible for data professionals to work in an evolutionary, if not agile, manner, it is in fact a preferable way to work.
2. **Learning curve**. It takes time to learn these new techniques, and even longer if you also need to change a serial mindset into an evolutionary one.
3. **Tool support is still evolving**. When *Refactoring* was published in 1999, no tools supported the technique. Just a few years later, every single integrated development environment (IDE) has code-refactoring features built right in to it. At the time of this writing, there are no database refactoring tools in existence, although we do include all the code that you need to implement the refactorings by hand. Luckily, the Eclipse Data Tools Project (DTP) has indicated in their project prospectus the need to develop database-refactoring functionality in Eclipse, so it is only a matter of time before the tool vendors catch up.

Agility in a Nutshell

Although this is not specifically a book about agile software development, the fact is that database refactoring is a primary technique for agile developers. A pro...

Users Review

From reader reviews:

Bruce Zimmerman:

What do you think about book? It is just for students because they are still students or that for all people in the world, the particular best subject for that? Only you can be answered for that question above. Every person has various personality and hobby for every single other. Don't to be compelled someone or something that they don't desire do that. You must know how great in addition to important the book Refactoring Databases: Evolutionary Database Design. All type of book could you see on many methods. You can look for the internet sources or other social media.

Alice Hille:

Now a day folks who Living in the era where everything reachable by match the internet and the resources inside can be true or not require people to be aware of each details they get. How people have to be smart in getting any information nowadays? Of course the answer then is reading a book. Reading a book can help folks out of this uncertainty Information specially this Refactoring Databases: Evolutionary Database Design book because this book offers you rich data and knowledge. Of course the information in this book hundred per cent guarantees there is no doubt in it you know.

Irene Robertson:

Information is provisions for individuals to get better life, information nowadays can get by anyone on everywhere. The information can be a expertise or any news even a problem. What people must be consider whenever those information which is inside former life are hard to be find than now is taking seriously which one is suitable to believe or which one the actual resource are convinced. If you find the unstable resource then you get it as your main information there will be huge disadvantage for you. All those possibilities will not happen within you if you take Refactoring Databases: Evolutionary Database Design as your daily resource information.

Shelly Sampson:

You may get this Refactoring Databases: Evolutionary Database Design by look at the bookstore or Mall. Just simply viewing or reviewing it may to be your solve trouble if you get difficulties for ones knowledge. Kinds of this publication are various. Not only simply by written or printed but can you enjoy this book through e-book. In the modern era just like now, you just looking because of your mobile phone and searching what your problem. Right now, choose your personal ways to get more information about your guide. It is most important to arrange yourself to make your knowledge are still revise. Let's try to choose suitable ways for you.

Download and Read Online Refactoring Databases: Evolutionary Database Design By Scott W. Ambler, Pramodkumar J. Sadalage #NKP86SXB4W5

Read Refactoring Databases: Evolutionary Database Design By Scott W. Ambler, Pramodkumar J. Sadalage for online ebook

Refactoring Databases: Evolutionary Database Design By Scott W. Ambler, Pramodkumar J. Sadalage Free PDF d0wnl0ad, audio books, books to read, good books to read, cheap books, good books, online books, books online, book reviews epub, read books online, books to read online, online library, greatbooks to read, PDF best books to read, top books to read Refactoring Databases: Evolutionary Database Design By Scott W. Ambler, Pramodkumar J. Sadalage books to read online.

Online Refactoring Databases: Evolutionary Database Design By Scott W. Ambler, Pramodkumar J. Sadalage ebook PDF download

Refactoring Databases: Evolutionary Database Design By Scott W. Ambler, Pramodkumar J. Sadalage Doc

Refactoring Databases: Evolutionary Database Design By Scott W. Ambler, Pramodkumar J. Sadalage Mobipocket

Refactoring Databases: Evolutionary Database Design By Scott W. Ambler, Pramodkumar J. Sadalage EPub