Optimizing Oracle Code

Includes a Quick-Reference Card

Oracle PL/SQL Best Practices

O'REILLY®

Steven Feuerstein

O'REILLY®

Oracle PL/SQL Best Practices

In this compact book, Steven Feuerstein, widely recognized as one of the world's experts on the Oracle PL/SQL language, distills his many years of programming, writing, and teaching about PL/SQL into a set of PL/SQL "best practices"—rules for writing code that is readable, maintainable, and efficient.

Oracle PL/SQL Best Practices offers practical answers to some of the hardest questions faced by PL/SQL developers, including:

- What is the best way to write the SQL logic in my application code?
- How should I write my packages so they can be leveraged by my entire team of developers?
- How can I make sure that all my team's programs handle and record errors consistently?

This book summarizes approximately 120 PL/SQL best practices in nine major categories: PL/SQL program development, coding style, data structures, control structures, exception handling, writing SQL in PL/SQL, program construction, package construction, and built-in packages. The book comes with a pull-out quick-reference card that lists all the best practices covered in the book. Code examples demonstrating many of the best practices are also available at the O'Reilly web site.

Oracle PL/SQL Best Practices is a concise, easy-to-use reference that PL/SQL developers will turn to again and again, and a book that no serious developer can afford to be without.

> *"O'Reilly's PL/SQL texts are the definitive guides to the language, and this new volume is an exceptional addition to their ranks. Containing valuable advice for both experts and beginners alike,* Oracle PL/SQL Best Practices *is a must-read for any Oracle developer interested in improving the quality of his or her PL/SQL applications. As an instructor, I have recommended Steven's books to my students many times, and this book will be placed at the top of that list of recommendations!"*
>
> —Miriam Moran, Instructor, ThinkSpark

www.oreilly.com

US $19.95 **CAN $29.95**

ISBN-10: 0-596-00121-5

ISBN-13: 978-0-596-00121-6

9 780596 001216 51995

Oracle PL/SQL Best Practices

Oracle PL/SQL Best Practices

Steven Feuerstein

O'REILLY®

Beijing · Cambridge · Farnham · Köln · Paris · Sebastopol · Taipei · Tokyo

Oracle PL/SQL Best Practices
by Steven Feuerstein

Published by O'Reilly Media, Inc., 1005 Gravenstein Highway North, Sebastopol, CA 95472.

Editor: Deborah Russell

Production Editor: Mary Anne Weeks Mayo

Cover Designer: Ellie Volckhausen

Printing History:

April 2001: First Edition.

This book uses RepKover™, a durable and flexible lay-flat binding.

ISBN: 0-596-00121-5
ISBN13: 978-0-596-00121-6
[M] [03/07]

To the many Israeli and Palestinian women who reject the violence of Israel's military occupation and have dedicated their lives to work for a just and lasting peace

—Steven Feuerstein

Table of Contents

Preface

When I first started writing about the Oracle PL/SQL language back in 1994, the only sources of information were the product documentation (such as it was) and the occasional paper and presentation at Oracle User Group events. Today, there are at least a dozen books that focus exclusively on PL/SQL, numerous products that help you write PL/SQL code (integrated development environments, knowledge bases, etc.), training classes, and web sites. And the community of PL/SQL developers continues to grow in size and maturity, even with the advent of Java.

Access to information about PL/SQL is no longer the challenge. It can, on the other hand, be difficult to make sense of all the new features, the numerous resources, the choices for tools, and so on. When it comes to writing a program or an entire application, developers have, over and over again, expressed the desire for advice. They ask:

- How should I format my code?
- What naming conventions, if any, should I use?
- How can I write my packages so that they can be more easily maintained?
- What is the best way to query information from the database?
- How can I get all the developers on my team to handle errors the same way?

So many questions, so much burning desire to write code well, and so few resources available to help us do that.

So I decided to write a book that offers a concentrated set of "best practices" for the Oracle PL/SQL language. The objective of this book is to provide concrete, immediately applicable, quickly located advice that will assist you in writing code that is readable, maintainable, and efficient.

You will undoubtedly find recommendations in this book that also appear in some of my other books; I hope you will not be offended by this repetition. It's simply impossible to offer in a single book everything that can and should be written about the Oracle PL/SQL language. While I plan to reinforce these best practices in the appropriate places in my other texts, I believe that we will all benefit from also having them available in one concise book, a book designed, beginning to end, to give you quick access to my recommendations for excellent PL/SQL coding techniques.

Structure of This Book

Oracle PL/SQL Best Practices is composed of nine chapters and one appendix. Each chapter contains a set of best practices for a particular area of functionality in the PL/SQL language. For each best practice, I've provided as many of the following elements as are applicable:

Title

A single sentence that describes the best practice and provides an identifier for it in the form *XXX-nn* (where *XXX* is the type of best practice—for example, EXC for exception handling—and *nn* is the sequential number within this set of best practices); see the section "About the Code" for how to use this identifier online. I have, whenever possible, sought to make this title stand on its own. In other words, you should be able to glance at it and understand its impact on how you write code. This way, after you've read the entire best practice, you can use Appendix A, *Best Practices Quick Reference* (or the pull-out quick-reference card), to instantly remind you of best practices as you write your code.

Description

A lengthier explanation of the best practice. It's simply not possible to cover all the nuances in a single sentence!

Example

We learn best from examples, so just about every best practice illustrates, through code and/or anecdote, the value of the best practice. Whenever it makes sense, I put the example code in a file that you can use (or learn from) in your own programming environment. You'll find these files on the O'Reilly web site (see "About the Code" later in this Preface).

Benefits

Why should you bother with this best practice? How crucial is it for you to follow this particular recommendation? This section offers a quick review of the main benefits you will see by following the best practice.

Challenges

Wouldn't it be great if we lived in a world in which following a best practice was all-around easier than the "quick and dirty" approach? That is, unfortunately, not always the case. This element warns you about the challenges, or drawbacks, you might face as you implement the best practice.

Resources

In the world of the Internet, everything is connected; no programmer stands alone! This section recommends resources, ranging from books to URLs to files containing code, that you can use to help you successfully follow this best practice. Where filenames are shown in this section, they refer to files available on, or referenced by, the O'Reilly web site.

Here are brief descriptions of the chapters and appendix:

Chapter 1, *The Development Process*, steps back from specific programming recommendations. It offers advice about how to improve the overall process by which you write code.

Chapter 2, *Coding Style and Conventions*, offers a series of suggestions on how to format and organize your code so that it is more readable and, therefore, more maintainable.

Chapter 3, *Variables and Data Structures*, takes a close look at how you ought to declare and manage data within your PL/SQL programs.

Chapter 4, *Control Structures*, is a "back to basics" chapter that talks about the best way to write IF statements, loops, and even the GOTO statement! Sure, these aren't terribly complicated constructs, but there are still right and wrong ways to work with them.

Chapter 5, *Exception Handling*, covers another critical aspect of robust application development: exception handling, or what to do when things go wrong.

Chapter 6, *Writing SQL in PL/SQL*, focuses on the most crucial aspect of PL/SQL development: how you should write the SQL statements in your programs.

Chapter 7, *Program Construction*, offers advice on how best to build your procedures, functions, and triggers—the program units that contain your business logic. It also includes best practices for parameter construction.

Chapter 8, *Package Construction*, steps back from individual program units to present recommendations for packages, the building blocks of any well-designed PL/SQL-based application.

Chapter 9, *Built-in Packages*, focuses on how to take best advantage of a few of the most often used of the packages provided to us by Oracle Corporation.

Appendix A, *Best Practices Quick Reference*, compiles the best practice titles across all the chapters into a concise resource. Once you have studied the individual best practices, you can use this appendix as a checklist, to be reviewed before you begin coding a new program or application. You'll also find a removable version of this appendix on the quick-reference card bound into the back of the book.

How to Use This Book

My primary goal in writing this book was to create a resource that would make a concrete, noticeable difference in the quality of the PL/SQL code you write. To accomplish this, the book needs to be useful and usable not just for general study, but also for day-to-day, program-to-program tasks. It also needs to be concise and to the point. A 1,000-page text on best practices would be overwhelming, intimidating, and hard to use.

The result is this relatively brief (I consider any publication under 200 pages a major personal accomplishment!), highly structured book. I recommend that you approach *Oracle PL/SQL Best Practices* as follows:

1. Read the next section, "Not All Best Practices Are Created Equal." Some of the best practices in this book—whole chapters, in fact—will have a much higher impact than others on the quality and efficiency of your code. If you find that your current practices (or those of your organization) are far from the mark, then you will have identified your priorities for initial study.
2. Skip to Appendix A and peruse the best practice titles from each chapter. If you have been programming for any length of time, you will probably find yourself thinking: "Yes, I do that," and "Uh-huh, we've got that one covered." Great! I would still encourage you to read what I've got to say on those topics, as you might be able to deepen your knowledge or learn new techniques. In any case, a quick review of the appendix will allow you to identify areas that are

new to you, or perhaps strike a chord, as in "Oh my gosh, that program I wrote last week does exactly what Steven says to avoid. Better check that out!"

3. Dive into individual chapters or best practices within chapters. Read a best practice, wrestle with it, if necessary, to make sure that you really, truly agree with it. *And then apply that best practice.* This isn't an academic exercise. You will only truly absorb the lesson if you apply it to your code—if you have a problem or program that can be improved by the best practice.

If you are new to programming or new to PL/SQL, you will certainly also benefit greatly from a cover-to-cover reading of the text. In this case, don't try to fully absorb and test out every best practice. Instead, read and think about the best practices without the pressure of applying each one. When you are done, try to picture the best practices as a whole, reinforcing the following themes:

- I want to write code that I—and others—can easily understand and change as needed.
- The world is terribly complex, so I should strive to keep my code simple. I can then meet that complexity through carefully designed interaction between elements of my code.

Then you will be ready to go back to individual chapters and deepen your understanding of individual best practices.

The other crucial way to take advantage of this book is to *use the code* provided on the companion web site. See the later section "About the Code," for detailed information on the software that will help you bring your best practices to life.

Not All Best Practices Are Created Equal

This book contains about 120 distinct recommendations. I could have included many, many more. In fact, I filled up a Rejects document as I wrote the book. Following the proven, "top-down" approach, I first came up with a list of best practices in each area of the language. Then I went through each area, filling in the descriptions, examples, and so on. As I did this, I encountered numerous "best practices" that surely were the right way to do things. The reality, however, is that few people would ever bother to remember and follow them, and if they did bother, it would not make a significant difference in their code.

I had realized, you see, that not all best practices are created equal. Some are much, much more important than others. And some are just better left out of the book, so that readers aren't distracted by "clutter." I hope that the result—this book—has an immediate and lasting impact. But even among the best practices I didn't reject, some stand out as being especially important—so I've decided to award these best practices the following prizes:

Grand Prize

SQL-00: Establish and follow clear rules for how to write SQL in your application. (See Chapter 6.)

First Prize

MOD-01: Encapsulate and name business rules and formulas behind function headers. (See Chapter 7.)

Second Prize: Two Winners

EXC-00: Set guidelines for application-wide error handling before you start coding. (See Chapter 5.)

PKG-02: Provide well-defined interfaces to business data and functional manipulation using packages. (See Chapter 8.)

Third Prize: Four Winners

MOD-03: Limit execution section sizes to a single page using modularization. (See Chapter 7.)

DAT-15: Expose package globals using "get and set" modules. (See Chapter 3.)

DEV-03: Walk through each other's code. (See Chapter 1.)

STYL-09: Comment tersely with value-added information. (See Chapter 2.)

If you follow each of these "best of the best" practices, you will end up with applications that are the joy and envy of developers everywhere!

About the Code

The best way to learn how to write good code is by analyzing and following examples. Almost every best practice offered in this book includes a code example, both in the text and in downloadable form from the Oracle PL/SQL Best Practices site, available through the O'Reilly & Associates site at:

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

As a rule, I will follow my own best practices in these examples (unless the point of the code is to demonstrate a "bad practice"!). So, for example, you will rarely see me using DBMS_OUTPUT.PUT_LINE, even though this "show me" capability is needed in many programs. As I mention in **BIP-01**, you should avoid calling this procedure directly; instead, build or use an encapsulation *over* DBMS_OUTPUT.PUT_LINE. So rather than seeing code like this:

```
DBMS_OUTPUT.PUT_LINE (l_date_published);
```

you will instead encounter a call to the "pl" or "put line" procedure:

```
pl (l_date_published);
```

I also make many references to PL/Vision packages. PL/Vision is a code library, consisting of more than 60 packages that offer 1,000-plus procedures and functions to perform a myriad of useful tasks in PL/SQL applications. I have deposited much of what I have learned in the last five years about PL/SQL into PL/Vision, so I naturally return to it for examples. Any package mentioned in this book whose name starts with "PLV" is a PL/Vision package.

A completely free, "lite" version of PL/Vision is available from the PL/SQL Pipeline Archives at:

http://www.quest-pipelines.com/pipelines/dba/PLVision/plvision.htm

Select the "RevealNet Active PL/SQL Knowledge Base" from the list. (You might also like to download and try out the other code you'll find there.) A commercial version of PL/Vision (with more packages and functionality than the lite version) is currently available inside the RevealNet Active PL/SQL Knowledge Base (*http://www.revealnet.com*).

Whenever possible, the code I provide for the book can be used to generate best-practice-based code and as prebuilt, generalized components in your applications, code that you can use without having to make any modifications.

The code examples offer programs that you can use to both generate and directly implement those best practices. In some cases, the programs are rather simple "prototypes"; they work as advertised, but you will probably want to make some changes before you put them into production applications.

And you should most certainly test every single program you use from *Oracle PL/SQL Best Practices*! I have run *some* tests, and my wonderful technical reviewers have also exercised the code. In the end, however, if

the code goes into your application, you are responsible for making sure that it meets your needs.

Other Resources

This book is intended to complement numerous other resources for PL/SQL developers. It is, to my knowledge, the first collection of best practices specifically for the Oracle PL/SQL language. On the other hand, it doesn't stand on its own as a comprehensive resource, either for PL/SQL, in particular, or for Oracle application development, in general.

What follows is by no means an exhaustive list of resources for developers. However, I find that a 15-page bibliography is more intimidating than it is helpful. So I offer this short list of the resources that I have recently found most useful and interesting:

Code Complete by Steven McConnell (Microsoft Press)
: A classic text, this "practical handbook of software criticism" should be on the bookshelf of every developer or at least in your team's library. Chock-full of practical advice for constructing code, it shows examples in many languages, including Ada, which is enough like PL/SQL to make learning from McConnell a breeze. Don't start coding without it! The web site for Steven McConnell's consulting practice, *www.construx.com*, is also packed with lots of good advice.

Refactoring by Martin Fowler (Addison Wesley)
: According to this book, "refactoring is the process of changing a software system in such a way that it doesn't alter the external of the code, yet improves its internal structure." Sound great, or *what*? This excellent book uses Java as its example language, but the writing is clear and the Java straightforward. There is much to apply here to PL/SQL programming.

Extreme Programming Explained, by Kent Beck (Addison Wesley)
: This book is a highly readable and concise introduction to Extreme Programming (XP), a lightweight software development methodology. Visit *http://www.xprogramming.com* or *http://www. extremeprogramming.org* for a glimpse into the world of this interesting approach to development.

And then, of course, there is my own oeuvre, the Oracle PL/SQL Series from O'Reilly & Associates, which includes:

Oracle PL/SQL Programming, with Bill Pribyl
: The complete language reference for Oracle PL/SQL.

Oracle PL/SQL Programming: Guide to Oracle8i Features
: A companion volume describing the Oracle8*i* additions to the PL/SQL language.

Oracle PL/SQL Developer's Workbook, with Andrew Odewahn
: A workbook containing problems (and accompanying solutions) that will test your knowledge of Oracle PL/SQL language features.

Oracle Built-in Packages, with Charles Dye and John Beresniewicz
: A complete reference to the many built-in packages provided by Oracle Corporation.

Advanced Oracle PL/SQL Programming with Packages
: A description of how to write your own PL/SQL packages, including a large number of packages you can use in your own programs.

Oracle PL/SQL Language Pocket Reference, with Bill Pribyl and Chip Dawes
: A quick reference to the PL/SQL language syntax.

Oracle PL/SQL Built-ins Pocket Reference, with John Beresniewicz and Chip Dawes
: A quick reference to the calls to the Oracle built-in functions and packages.

Conventions Used in This Book

The following typographical conventions are used in this book:

Italic
: Indicates filenames, directory names, and URLs. It's also used for emphasis and for the first use of a technical term.

Bold
: Used when referring, by number, to a best practice described in this book (e.g., **BIP-04**).

`Constant width`
: Indicates examples and to show the contents of files and the output of commands.

`Constant width bold`
: Indicates code entered by a user (e.g., via SQL*Plus) or to highlight code lines being discussed.

UPPERCASE
: In code examples, indicates PL/SQL keywords.

lowercase
: In code examples, indicates user-defined items (e.g., variables).

The owl icon designates a note, which is an important aside to the nearby text. For example, I'll use this icon when suggesting the use of an alternative feature.

The turkey icon designates a warning relating to the nearby text. For example, I'll use this icon when a particular feature might affect performance or preclude use of some other feature.

Comments and Questions

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

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

There is a web page for this book, which lists errata, examples, or any additional information. You can access this page at:

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

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

bookquestions@oreilly.com

For more information about books, conferences, software, Resource Centers, and the O'Reilly Network, see the O'Reilly web site at:

http://www.oreilly.com

Acknowledgments

Thanks go, first of all, to my editor of six years at O'Reilly & Associates, Deborah Russell. She got me off the dime on this project and helped me turn it around in record time (I started doing serious writing on this book in October 2000 and finished it up in January 2001). It was, once again, a real pleasure working with you, Debby!

Thanks as well to the other O'Reilly people who turned this book into a finished product: Mary Anne Weeks Mayo, the production editor; Ellie Volckhausen, who designed the cover; and Caroline Senay, the editorial assistant who helped in many ways throughout the project.

Many outstanding Oracle developers and DBAs contributed their time and expertise, through technical review, code samples, or writing. My deep-felt gratitude goes out to: John Beresniewicz, Rohan Bishop, Dick Bolz, Dan Clamage, Bill Caulkins, Dan Condon-Jones, Fawwad-uz-Zafar Siddiqi, Gerard Hartgers, Edwin van Hattem, Dwayne King, Darryl Hurley, Giovanni Jaramillo, Vadim Loevski, Pavel Luzanov, Matthew MacFarland, Jeffrey Meens, James "Padders" Padfield, Rakesh Patel, Bill Pribyl, Andre Vergison (the brains behind PL/Formatter), and Solomon Yakobson. This book benefited, in particular, from a reworking of best practice titles by John Beresniewicz, close readings of many chapters by Dan Clamage (whose excellent comments on certain best practices I've included as sidebars in the text), and the contribution of trigger best practices by Darryl Hurley.

Oracle PL/SQL Best Practices is a much improved text as a result of all of your assistance, my friends. Any errors, on the hand, are entirely my fault and responsibility.

I would also like to thank my wife, Veva, for volunteering to pick up Eli from Jordan's house so that I could stay behind and write these acknowledgments (oh, and also for adding layer upon layer of meaning and happiness to my life).